An automated exact solution framework towards best subset selection in logistic regression

TK van Niekerk



orcid.org / 0000-0003-1379-9109

Thesis accepted in fulfilment of the requirements for the degree Doctor of Philosophy in Engineering with Industrial Engineering at the North-West University

Promoter:	Dr JV Venter
Co-promoter:	Prof SE Terblanche

Graduation: May 2024

Acknowledgements

I would first like to thank my study leaders, Prof. Terblanche and Dr. Venter, for their unwavering support and guidance throughout this Ph.D. journey. I have drawn immense value from our discussions concerning the Ph.D. topic and have come to realise that a solid academic support system is imperative to ensure success in completing one's Ph.D. studies. Thank you both for imparting some of your hard-earned knowledge and experience. Secondly, I would like to thank my friends and family for always lending an ear when times got tough, or motivation faltered. This long journey would not have been possible without your support.

To my wife, words can not express my gratitude when considering your loving kindness, understanding nature, motivation from the sidelines and determination to see me succeed. Thank you for keeping me focused and encouraging me to consistently put in the effort, even if it means writing one paragraph daily. Consistency is a vital attribute of a Ph.D. student; without it, failure would be inevitable.

Lastly, but most importantly, I would like to give all the glory and honour to the triune God (Father, Son and Spirit) since my talents and abilities are not of my own doing. I had to put in the work, but ultimately with His guidance, strength and wisdom, completing this Ph.D. study was possible. Glory be to God, my saviour.

Abstract

Logistic regression is a simplistic yet powerful modelling technique widely utilised to solve binary and multi-class classification problems. It provides interpretable and actionable insights about the relationship between independent variables and the target, which is not always possible when considering complex machine learning algorithms. As such, logistic regression is favoured as a modelling technique in the financial and healthcare industries where interpretability is paramount. Its interpretability relates to the fact that it can translate a linear combination of regression parameter estimates to a binary class domain while also providing statistically interpretable parameter estimates that could bring to light valuable underlying trends. Identifying a subset of predictive independent variables from a large set of potential inputs can drive much-needed value creation for any business. The NP-hard best subset selection problem is a well-known problem within literature, with multiple solution algorithms proposed to produce parsimonious predictive models. As part of this thesis, a novel exact automated solution framework is proposed to solve the best subset selection problem within the context of logistic regression. The framework uses a mixed integer programming piecewise linear underestimation of the log-likelihood function, as proposed by Venter (2020), while utilising explicit cardinality constraints to facilitate variable selection. The automated logistic regression solution framework (ALRSF) comprises four distinct steps: a cardinality selection step, a search space pruning step, an optimality step and a bound tightening step. The proposed framework also draws on techniques such as Benders decomposition and Lagrangian relaxation to reduce the computational burden associated with finding high-quality solutions while simultaneously making strides towards proving optimality. Empirical results show that the proposed solution framework significantly outperforms the standard branch-and-bound algorithm that is often used to perform best subset selection (which also serves as the benchmark during our comparison). Notable improvements in memory requirement, optimality gap size, problem size, and solution quality are observed. The results suggest that the proposed deterministic ALRSF provides a holistic and computationally efficient approach towards solving the NP-hard best subset selection problem.

Keywords: Best Subset Selection; Variable Selection; Logistic Regression; Mixed Integer Programming; Feature Subset Selection

Contents

1 Introduction **1**

1.1 Background and rationale 1

1.2 Research problem, research purpose and objectives 6

1.3 Research methods 8

1.4 Chapter Division 8

2 Logistic regression overview **10**

2.1 Introduction 10

2.2 Logistic regression parameter estimates 12

2.3 Maximising the log-likelihood function 14

2.4 Summary 16

3 Logistic regression subset selection approaches **17**

3.1 Introduction 17

3.2 Stepwise variable subset selection 17

3.2.1 Forward selection procedure 20

3.2.2 Backwards elimination procedure 21

3.2.3 Bi-directional stepwise procedure 22

3.2.4 Stepwise limitations 24

3.3 Regularised variable subset selection 26

3.3.1 The lasso, ridge regression and special case penalties 27

3.3.2 The elastic net penalty . . . 30
3.3.3 SCAD and MCP penalties . . . 33
3.3.4 Regularisation-based MIP formulations . . . 37
3.4 Best subset selection . . . 40
3.4.1 Best subset selection based MIP formulations . . . 44
3.4.2 Best subset selection limitations . . . 46
3.5 Summary . . . 47

4 Overview of relevant optimisation theory 48

4.1 Benders decomposition . . . 49
4.2 Lagrangian relaxation . . . 51
4.2.1 Towards solving the Lagrangian dual . . . 53
4.3 Summary . . . 55

5 Mathematical models 56

5.1 Model 1: Piecewise linearised log-likelihood MIP formulation - existing approach . . . 57
5.2 Model 2: Automated logistic regression solution framework (ALRSF) . . . 62
5.2.1 ALRSF - Proposed MIP formulation (Steps 0 - 2) . . . 63
5.2.2 ALRSF - Proposed Solution Framework (Steps 0 - 2) . . . 66
5.2.3 ALRSF dummy example (Steps 0 - 2) . . . 79
5.2.4 Proposed Lagrangian Reformulation (Step 3) . . . 83
5.2.5 Proposed sub-gradient solution algorithm (Step 3) . . . 85
5.3 Contributions of the proposed ALRSF . . . 89
5.4 Summary . . . 90

6 ALRSF model verification 91

6.1 Experimental assumptions . . . 91
6.2 Computational hardware and software . . . 93

6.3 Verification dataset . . . 93
6.4 ALRSF regression parameter estimates verification . . . 94
6.5 ALRSF cardinality selection verification . . . 96
6.6 Grid points estimation verification . . . 97
6.7 Summary . . . 98

7 ALRSF model validation **99**

7.1 Experimental datasets . . . 99
7.1.1 Superconductivity . . . 100
7.1.2 Relative location of CT slices . . . 100
7.1.3 Online news popularity . . . 101
7.1.4 Default of credit card clients . . . 102
7.2 Experimental results (Steps 0 to 2 of the ALRSF) . . . 103
7.2.1 Predictive performance . . . 103
7.2.2 Memory comparison . . . 108
7.2.3 Optimality gap and integer solution comparison . . . 110
7.2.4 Algorithmic execution time . . . 111
7.2.5 Additional observations made . . . 111
7.3 Experimental results (Step 3) . . . 112
7.3.1 Algorithmic bound tightening results . . . 114
7.3.2 Optimality Gap Improvements . . . 116
7.4 Summary . . . 117

8 Conclusion and recommendations **119**

8.1 Concluding remarks . . . 119
8.2 Future research initiatives . . . 120

A Peer-reviewed journal article published in the South African Statistical Journal (SASJ) **129**

List of Figures

1.1 Variable subset selection algorithmic categories 1

1.2 Filter methodology 2

1.3 Wrapper methodology 3

1.4 Hybrid methodology 3

1.5 Embedded methodology 3

2.1 Logistic Regression sigmoid function 13

3.1 Lasso, SCAD and MCP penalties 35

3.2 Lorenze curve 43

3.3 Piecewise Linear underestimation of the logistic loss function 45

5.1 Piecewise linear underestimation of the log-likelihood function 58

5.2 AUC performance measure as a function of cardinality (c) 61

5.3 Independent variable parameter estimate integer and decimal precision representation 66

5.4 ALRSF Logical process flow 67

7.1 Superconductivity Classification Problem 106

7.2 Relative location of CT slices Classification Problem 106

7.3 Online News Popularity Classification Problem 107

7.4 Credit Card Default Classification Problem 107

7.5 BandB vs ALRSF Memory comparison 109

7.6 BandB vs ALRSF Gap percentage comparison 110

7.7 BandB vs ALRSF Gap percentage comparison 111

7.8 Superconductivity classification problem - bound tightening 115

7.9 Relative location of CT slice classification problem - bound tightening . . . 115

7.10 Improvement in optimality gap - bound tightening 117

List of Tables

5.1 Integer to rescaled relational regression parameters 65

5.2 Novel ALRSF - Dummy Example 80

6.1 Independent variable description 94

6.2 Regression parameter estimate comparison 95

6.3 Influence of varying values for H - ALRSF 95

6.4 AUC performance measure comparison 96

6.5 AUC obtained at each iteration of the ALRSF that was applied to an eight-variable dataset 97

7.1 ALRSF vs B&B - Experimental Results 104

7.2 ALRSF vs B&B - Experimental Results 105

7.3 B&B LP relaxation (LPR) vs ALRSF LP relaxation (LPR) vs ALRSF Lagrangian relaxation (LR) - Experimental Results 113

Table of abbreviations

AIC	Akaike information criteria
ALRSF	Automated logistic regression solution framework
AIC	Akaike information criteria
API	Application programming interface
AUC	Area under the curve
BIC	Bayesian information criteria
BSS	Backward subset selection
CSV	Comma separated values
CT	Computed tomography
FIC	Focused information criteria
FSS	Forward subset selection
GB	Gigabytes
IV	Information value
LR	Lagrangian relaxation
LPR	Linear programming relaxation
MIP	Mixed integer programming
NP	Non-deterministic polynomial-time
RAM	Random Access Memory
ROC	Receiver operating characteristic
VIF	Variance inflation factor

Chapter 1

Introduction

1.1 Background and rationale

Variable subset selection follows the principle of Occam's razor, which relates to the model with the smallest number of independent variables that adequately describes the underlying data being preferred over a highly complex and convoluted model (Gheyas and Smith, 2009). As such, various methods have been proposed to try and find optimal or near-optimal solutions to the variable subset selection problem. The diagram in Figure 1.1 summarises the numerous algorithmic advances that have been made within the context of variable subset selection solution algorithms and classifies these into logical categories.

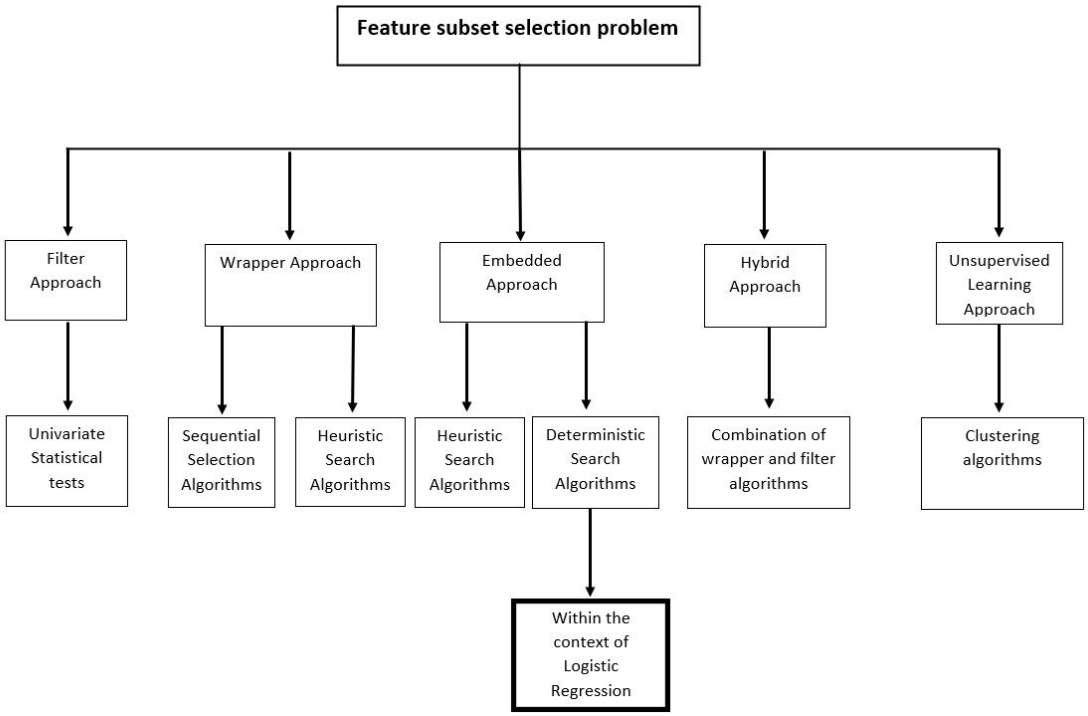


Figure 1.1: Variable subset selection algorithmic categories

With regards to Figure 1.1, variable subset selection algorithms can mainly be subdivided into filter, wrapper, embedded, hybrid and unsupervised approaches, the significant majority of which fall under the filter and wrapper categories [1]. These sub-categories will now be discussed in more detail in an attempt to clarify their distinct properties and to describe the literature gap that will be addressed as part of this research endeavour.

In general, filter methods can be defined as methods that are used as data pre-processing steps to remove irrelevant or redundant independent variables before model training or induction is initiated. These methods usually calculate univariate statistics that describe the predictive nature of each independent variable in relation to the response variable and exclude variables based on a predefined acceptance threshold. Frequently used approaches include the t-test, chi-squared test, correlation coefficients and mutual information [2], to name a few (Gheyas and Smith, 2009). The main drawback associated with most filter methods is the inadequacy of describing variable interactions and the influence of these on the final model's predictive performance (Lopez et al., 2004). The diagram in Figure 1.2 below presents an overview of the filter methodology.

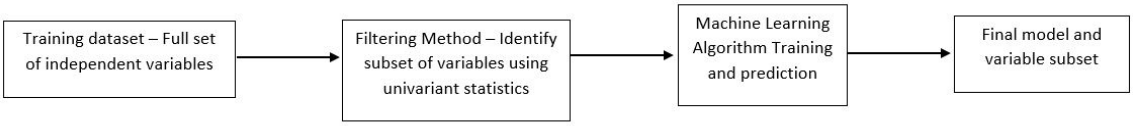


Figure 1.2: Filter methodology

Wrapper methods usually employ an iterative solution approach. The iterative solution approach attempts to find an near-optimal subset of variables by iteratively evaluating the influence of each subset of variables on the predictive performance of the generated model. The algorithm aims to find the subset of variables that will maximize the predictive performance of the final model selection. This approach is generally computationally expensive as the number of variable subsets to consider grows exponentially with an increase in the total number of independent variables. Additionally, the iterative nature of the training algorithms contribute to the already-exacerbated computational requirements. Some wrapper methods include forward subset (FSS) and backward subset (BSS) selection algorithms. While FSS and BSS are computationally efficient for relatively small subsets, they quickly become time-consuming when the dimensionality of the independent variable set becomes larger (approximately greater than 60 variables) (Gheyas and Smith, 2009; SAS, 2005). Figure 1.3 provides a graphical overview of the wrapper approach.

[1](Chandrashekar and Sahin, 2014; Civitelli et al., 2021; Sato et al., 2016a; Inza et al., 2000; Al-Ani, 2005; Hu et al., 2008; Karegowda et al., 2010; Li et al., 2011; Sun et al., 2004; Bell, 2000; Hall and Smith, 1998; Yang and Honavar, 1998; Kohavi and John, 1996; Hall and Smith, 2011; Song et al., 2011; Tan et al., 2008; Narendra and Fukunaga, 1977; Al-Ani, 2007; Lopez et al., 2004; Gheyas and Smith, 2009).

[2]Mutual information (MI) describes the amount of information that one variable gives about another. In general, mutual information is a dimensionless quantity. A large MI value relates to a reduction in uncertainty of one variable given information about the other.

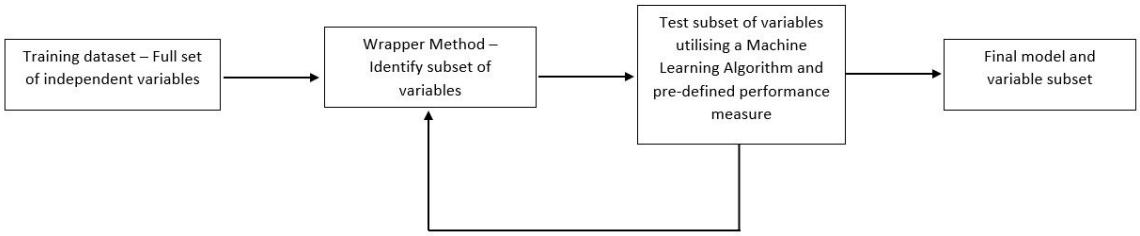


Figure 1.3: Wrapper methodology

Due to the expensive computational nature of wrapper methods, hybrid approaches have also been proposed that exhibit time complexity closer to filter methods (Song et al., 2011). A filter method is initially applied to reduce the search space, after which the remainder of the independent variables (which were not eliminated with the initial filter approach) are evaluated using a wrapper algorithm. This approach attempts to leverage off of the advantageous characteristics of both filter and wrapper methods. Figure 1.4 describes the methodology behind a general hybrid approach.

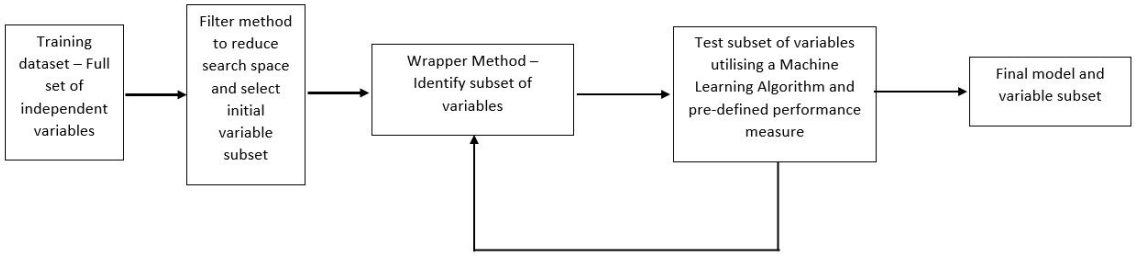


Figure 1.4: Hybrid methodology

In contrast to wrapper and filter methods, the main aim of the embedded approach is to eliminate the recursive identification of variable subsets (as is the case in wrapper methods) by incorporating variable selection as part of the algorithmic training process (Lopez et al., 2004). These methods are usually specific to a training algorithm and may reduce the computational time required to fit the model when compared to wrapper methods. Neural networks, decision trees, lasso regression (L_1-regularisation) and ridge regression (L_2-regularisation) are examples of embedded approaches (Song et al., 2011). Figure 1.5. provides a brief summary of embedded approaches.

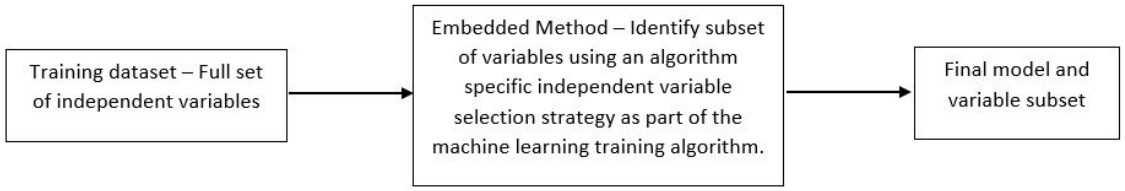


Figure 1.5: Embedded methodology

The research presented in this thesis will focus on expanding the knowledge base associated with embedded methods, with a key interest in best subset selection within the

context of logistic regression (binary classification) that utilises exact deterministic solution algorithms. The highlighted block in Figure 1.1 indicates where the specific research topic fits into the larger field of study. Compared to the vast variable subset selection literature, the former accounts for a small portion of research initiatives. However, a renewed interest in the use of exact deterministic solution algorithms to solve variable subset selection problems (within the context of regression modelling) has been observed in the past few years. This can mainly be attributed to performance improvements in computational software and hardware, and algorithmic advances made towards solving large-scale complex deterministic mixed integer programming (MIP) problems. The variable subset selection problem is known to be NP-hard, with the solution search space size exponentially increasing at a rate of 2^p with an increase in the number of independent variables (denoted by p) (Lopez et al., 2004; Gheyas and Smith, 2009). As such, finding high-quality solutions to expansive logistic regression variable subset selection problems by means of exact approaches is seen as improbable by some (Al-Ani, 2005).

From the numerous supervised algorithms available in literature, we choose to focus on logistic regression because it exhibits many preferable characteristics, which include ease of implementation and interpretability (Civitelli et al., 2021). A number of exact solution MIP formulations have recently been proposed to solve the variable subset selection problem within the context of logistic regression. These MIP formulations can be grouped into three main categories, as summarised below.

1. The first approach attempts to find the optimal set of independent variables by minimising the logistic loss objective function in conjunction with a regularisation term (or combination of regularisation terms) that makes use of one or more L_q-norms, where $q \in \{0, 1, 2\}$ (Dedieu et al., 2021; Insolia et al., 2021; Kamiya et al., 2019; Takano and Miyashiro, 2020). A formal definition of the L_q-norm notation will be provided as part of Chapter 3.

2. A second approach attempts to minimise a piecewise linear estimation of the logistic loss objective function in combination with instance-specific information criteria (IC), some of which include Akaike information criteria (AIC) or Bayesian information criteria (BIC) (Sato et al., 2016b; Civitelli et al., 2021).

3. Finally, the last approach aims to maximise a piecewise linear underestimation of the log-likelihood objective function in the presence of explicit cardinality constraints, which limit the number of independent variables that can be selected by the model (Venter, 2020).

Next we briefly mention a few high-level limitations associated with each of the aforementioned MIP approaches so as to provide more clarity on the contribution of the research contained within this thesis. Note that a detailed exposition on the limitations of each approach will be given in Chapter 3. These limitations are listed below.

1. For problems with low dimensionality, lasso regularisation introduces bias while adversely affecting statistical inference and interpretability of the regression parameter

estimates (Heinze et al., 2018). In other words, due to the shrinkage of regression parameters, the resulting model produced by lasso makes descriptive modelling and the comparison of log-odds ratios between independent variables difficult.

2. Like lasso, ridge regression also introduces bias into the model by shrinking parameter estimates. Unlike lasso, however, ridge regression does not truly select the best subset of independent variables. This is because the algorithm never sets any regression coefficients exactly equal to zero, but simply shrinks some of them such that they have near-zero values (Venter, 2020).

3. The choice of the penalty or regularisation parameters for best subset selection, lasso and ridge regularisation is usually determined by means of some form of heuristic-based solution algorithm such as grid-search. Within a MIP context, a two-step approach is usually employed. The first step attempts to select an appropriate value for each of the regularisation parameters, while the second step solves the MIP problem by using some deterministic solution algorithm, such as the straightforward branch-and-bound approach. Given that the first step of the algorithm is heuristic in nature, it is nearly impossible to determine the distance between the final solution (that uses a set of regularisation parameters) and the optimal solution (best possible solution). To summarise, heuristic-based techniques such as cross-validation and grid search are usually employed to estimate an acceptable value for the penalty parameters based on model performance and cannot be used simultaneously alongside deterministic solution algorithms to find the optimal solution over the full spectrum of regularisation parameters. This would result in a non-linear objective function which would lead to undue complexity, extensive computational requirements and unrealistic execution times [3].

4. Empirical evidence has shown that MIP formulations which attempt to select the best variable subset using information criteria such as AIC or BIC tend to favour less parsimonious models (Kutner et al., 2005; Venter, 2020). This is an undesirable model selection characteristic because we are interested in the smallest possible model that adequately describes the underlying population.

5. The MIP formulation proposed by Venter (2020) has a somewhat manual component as the extreme points of the linear splines need to be estimated through the solving of a linear relaxation of the original MIP, which excludes all cardinality constraints. Similarly, the proposed formulation also leaves the selection of the cardinality parameter to the model developer. A preferred characteristic of the proposed formulation is, however, the fact that it utilises explicit cardinality constraints without any regularisation being employed. This makes statistical inference with regards to the regression parameters possible and enhances model interpretability.

Considering the latter, the research presented in this thesis will aim to build on the work of Venter (2020) while attempting to address some of the limitations noted.

[3] Please refer to Chapter 3.3.4 for a formal definition of the optimality gap used throughout the thesis. Furthermore, Chapter 3.3.4 also clarifies the limitations associated with these MIP-based lasso and ridge-regression type formulations.

1.2 Research problem, research purpose and objectives

As motivated in the previous section, solving the best variable subset selection problem (within the context of MIP) can lead to more descriptive and parsimonious models with greater predictive power. Even a marginal increase in predictive performance can unlock significant financial gain from a business perspective. Parsimonious models also (in the absence of any independent variable parameter regularisation) make statistical inference more plausible by providing greater insight with respect to the relationship between the independent variables and the response variable. By utilising an exact solution framework, such as linear programming (LP) and MIP approaches, an optimality gap can be provided, which quantifies the quality of the solutions obtained. Alternatively, note that it is not possible to obtain an optimality gap when using heuristic-based variable selection methodologies, such as stepwise regression and approaches that make use of p-values for significance tests. Nevertheless, solving the best subset selection problem by means of a MIP framework is considered to be an NP-hard problem (Lopez et al., 2004; Gheyas and Smith, 2009). According to Miller (2002), variable subset selection problems that utilise MIP formulations can be seen as exponential-time procedures. This means that an exponential increase in execution time is observed with an increase in the number of independent variables. It has also been noted by Lund (2017) that solving variable subset selection problems using commercial software such as SAS becomes intractable when problem instances incorporate more than 75 independent variables. As such, greater advances from an algorithmic perspective are required to solve MIP variable subset selection problems that are relatively large in size. Given this background, the main aim of this thesis is to develop a novel automated solution framework (with explicit cardinality constraints) that enables improved adoption of MIP approaches during the solving of best subset selection problems.

The following research objectives are pursued in this thesis:

1. To conduct a thorough literature review with regards to:
    - Relevant variable subset selection techniques within the context of logistic regression.
    - Evaluate the advantages and limitations of each subset selection technique.
    - Expound on key statistical concepts used in the solution framework development process.
2. To provide an overview of relevant optimisation theory used to exploit the underlying structure of MIP formulations and how it can be applied to the logistic regression variable subset selection problem domain. Some of the techniques that will be considered include:
    - The Benders decomposition algorithm used to decompose a MIP with a specific block structure into master and sub-problems such that the problem has a reduced memory requirement.

- Lagrangian relaxation, which allows for the relaxation of complicating constraints into the objective function while penalising constraint violation through Lagrangian multipliers. The well-known sub-gradient method will also be considered as a solution algorithm to the Lagrangian dual.

3. To develop a novel solution framework that can be applied to the best subset selection problem within a MIP context. This solution framework will aim to:

   - Reduce the memory requirement associated with solving large-scale variable subset selection problems.
   - Push the boundaries with respect to the number of independent variables and training observations that can be considered as part of the model fitting process.
   - Automate cardinality selection by means of an explicit cardinality constraint that does not shrink regression parameter estimates and, in turn, allows for unbiased statistical inference.
   - Combine the concept of solution heuristics and MIP to prune the solution search space, which allows for the discovery of higher quality solutions closer to the optimal solution. The proposed solution framework also provides an optimality gap that can be used to quantify the quality of the final solution, which is not possible with most heuristic approaches (such as stepwise regression or procedures that make use of p-values for significance tests).
   - Largely automate hyper-parameter selection in an attempt to prevent the model developer from specifying the hyper-parameters.

4. Conduct verification and validation experiments to test whether:

   - The proposed MIP formulation and automated solution framework adequately capture the underlying logistic regression variable subset selection problem. This will be achieved by comparing the proposed framework with the standard branch-and-bound algorithm.
   - The regression parameter estimates and final set of independent variables (cardinality) are comparable with existing approaches.
   - The proposed ALSRF outperforms the most recent approaches proposed in literature from a memory, solution quality, optimality gap and problem instance size (number of independent variables and training observations) perspective.

5. Propose key areas of importance for future research endeavours that may allow for further development and progress within the area of MIP deterministic best subset selection solution frameworks.

6. Finally, produce a research article that is publishable in an peer-reviewed academic journal. This objective is satisfied as outlined in Appendix A. Our article titled, "An automated exact solution framework towards solving the Logistic regression best subset selection problem", has been published in the peer-reviewed South African Statistical Journal (van Niekerk et al., 2023).

1.3 Research methods

Currently, the limiting factor to the wide-scale use of MIP formulations when solving best subset selection problems relates to the memory constraint associated with large problem instances. As such, new algorithmic developments are required to address these limitations. To meet the aims and objectives (as set out in Section 1.2 of Chapter 1), a combined approach leveraging MIP, heuristics and advanced decomposition algorithms will be considered. The solution framework will be developed using the C++ concert technology API while also attempting to address most of the high-level limitations noted in Section 1.1 of Chapter 1.

1.4 Chapter Division

This thesis is subdivided into eight chapters, including the introductory chapter. The introductory chapter aims to clearly define the problem statement while also stating the main aims and objectives that will drive the research outputs. The concept of probabilistic modelling using the logistic regression modelling technique is introduced in Chapter 2 to set the scene before drilling down into the variable subset selection literature.

A detailed literature review will be presented in Chapter 3 and will focus on the different subset selection approaches available in literature. This is followed by an exposition of relevant optimisation theory in Chapter 4. The main idea behind these chapters is to provide an adequate overview of the theoretical and mathematical concepts that underpin variable subset selection. Chapters 3 and 4 will also highlight the litany of limitations associated with current variable selection approaches (available in literature), after which a novel logistic regression best subset selection solution algorithm will be formally introduced in Chapter 5, which aims to address these shortcomings.

Chapter 5 first introduces the piecewise linear approximation of the logistic regression best subset selection problem, which is solved using the standard branch-and-bound algorithm. Secondly, novel alterations to the former MIP formulation proposed by Venter (2020), along with the newly proposed ALRSF, are discussed.

Verification experiments are conducted in Chapter 6 to ensure that the proposed MIP framework accurately captures the underlying logistic regression best subset selection problem. The empirical results also aim to verify whether the regression parameter estimates and model selection proposed by the ALRSF is well within range of expected results when compared to the benchmark approach presented in Venter (2020).

The robustness of the proposed ALRSF is evaluated in Chapter 7 by solving four relatively large-scale real-world classification problems. The results are critically evaluated against the benchmark (standard branch-and-bound algorithm) through a comparative study that considers the memory requirement associated with each solution algorithm, the final optimality gap obtained, the quality of the integer solution discovered and the problem size

that could be considered. Finally, the key contributions made with regard to solving the best subset selection problem are summarised in Chapter 8, followed by recommendations for future research endeavours.

Chapter 2

Logistic regression overview

Logistic regression, a generalised linear model with a logit link function, is regularly used to solve binary classification problems (Czepiel, 2002). When faced with a categorical response variable, linear regression is considered an inappropriate modelling technique because the response variable is not measured on a ratio scale and lacks normally distributed error terms (Czepiel, 2002). Linear regression produces an output in the range $[-\infty, -\infty]$ while categorical variables can only take on a discrete set of values (Czepiel, 2002). Therefore, an alternative modelling technique is required to relate the output generated by the linear combination of independent variables (linear regression) to the categorical response variable domain. This chapter will provide an overview of the logistic regression model, followed by the derivation of the log-likelihood function, which, when maximised, allows for the determination of the logistic regression parameter estimates. We conclude by formalising the Newton-Raphson algorithm, which is most frequently used to maximise the log-likelihood function.

2.1 Introduction

As part of this thesis, the underlying principles that lead to the derivation of the logistic regression formulation will be considered. However, a broader overview of its evolution is available for the interested reader in Cramer (2002). According to Cramer (2002), the logistic function was developed in the 19th century with its origin traced back to two main areas of study. The first attempts to describe the growth rate of populations while the second gives insight into the progression of autocatalytic chemical reactions. The derivation of the logistic regression function (as we know it today) is based on the growth rate formulation presented in (2.1). Let $f(t)$ represent the time path of a quantity with a corresponding growth rate defined according to

$$\dot{f(t)} = \frac{df(t)}{dt}. \tag{2.1}$$

The equation in (2.1) describes the change of $f(t)$ based on t (where t is a measure of time), which leads to the rate of change $f(\dot{t})$ [1]. The rate of growth can also be formulated in a slightly different manner by assuming that $f(\dot{t})$ can be expressed as a function of $f(t)$ by using some quantity β which denotes a constant growth rate. The rate of growth $f(\dot{t})$ can then be described according to

$$f(\dot{t}) = \beta f(t), \tag{2.2}$$

which (in the context of population growth) ultimately leads to exponential growth in the form of

$$f(\dot{t}) = f(0)\exp(\beta t), \tag{2.3}$$

where $f(0)$ denotes the initial value of $f(t)$ at time $t = 0$. This implies that as t increases, the rate of change associated with $f(t)$ will follow an exponential trend. Cramer (2002) noted that while (2.3) would be a satisfactory formulation to quantify unopposed population growth, it would lead to population values that do not adequately describe reality. As noted by Pierre Verhulst (a Belgian mathematician and demographer), population growth will most certainly be faced with some form of resistance to further expansion. As such, a resistance term was added to (2.3) which led to

$$f(\dot{t}) = \beta f(t) - \Upsilon(f(t)), \tag{2.4}$$

with $\Upsilon(\cdot)$ representing a resistance function. The resistance term $\Upsilon(f(t))$ is dependent on the size of the population $f(t)$ and can take on many forms. When the resistance term is quadratic in nature, the rate of growth $f(\dot{t})$ can be modified to allow

$$f(\dot{t}) = \beta f(t)(\Phi - f(t)), \tag{2.5}$$

where Φ defines the upper bound of f. This creates a growth rate dependency on both the current population $f(t)$ and the margin for further growth $(\Phi - f(t))$. Cramer (2002) subsequently shows that the population size can be expressed as a proportion of the population upper limit Φ through $p(t) = \frac{f(t)}{\Phi}$. By dividing (2.4) with Φ, (2.4) can be rewritten such that

$$p(t) = \beta p(t)(1 - p(t)). \tag{2.6}$$

A solution to (2.6) can be described by

$$p(t) = \frac{\exp(\alpha + \beta t)}{1 + \exp(\alpha + \beta t)}. \tag{2.7}$$

The equation in (2.7) resembles the well-known logistic regression equation that is used extensively today where α is the intercept that represents some baseline (Cramer, 2002). In the next section, the approach that is used to estimate the logistic regression parameters will be discussed along with an explanation on how (2.7) slots into the logistic regression modelling framework.

[1] Let t denote a measure of time as part of Section 2.1 of Chapter 2 and an algorithmic counter variable throughout the remainder of the thesis such that t represents the t-th iteration out of T total possible iterations.

2.2 Logistic regression parameter estimates

This section will be focused on the derivation of the well-known log-likelihood function. The log-likelihood function forms the cornerstone of most solution algorithms when attempting to fit a logistic regression model to training data and is derived from the logistic function presented in Section 2.1 of Chapter 2. Let $i \in n = \{1, 2, ..., n\}$ denote the i-th observation from n records and $j \in p = \{0, 1, 2, ..., p\}$ denote the j-th independent variable from a total of p independent variables. The independent variables are defined according to $X_i \in \mathbb{R}^p$ with X_i a vector of size p. Alternatively, the independent variables can be defined as a matrix of constants $\mathbf{X}$ with size $n \times p$, where $\mathbf{X} \in \mathbb{R}^{n \times p}$. Allow $Y_i \in \{0, 1\}$ to describe the binary dependent variable associated with the i-th observation. Therefore, the dependent variable can be defined as a binary vector $\mathbf{Y}$ of size n with $\mathbf{Y} \in \{0, 1\}^n$. Since Y_i follows a Bernoulli distribution with $P(Y_i = 1|X_i) = (\frac{1}{1+\exp(-\beta^T X_i)})$, it can be stated that $P(Y_i = 1) = \vartheta_i$ and $P(Y_i = 0) = 1 - \vartheta_i$ where $\vartheta \in \mathbb{R}^n, 0 \leq \vartheta \leq 1$, is a vector of size n. As part of this thesis, independent variables refer to all variables X_i used as input to a machine learning model in an attempt to describe the dependent variable Y_i (also referred to as the response or target variable). Furthermore, it is assumed that the observations (X_i, Y_i) are independently sampled from the same population. First let us define

$$\nu_i = \beta^T X_i \ \ \forall i \in n, \tag{2.8}$$

with ν denoting a vector of size n such that $\nu = (\nu_1, \nu_2, ..., \nu_n)$. Each ν_i is defined as a linear combination of all independent variable values associated with the i-th observation (contained within the vector X_i) and a set of regression parameters $\beta \in \mathbb{R}^p$. $\beta^T X_i$ can be expressed as $\sum_{j \in p} \beta_j X_{ij}$, with $X_{i0}\beta_0$ denoting the intercept term of the linear equation and X_{i0} taking on a value of one for all observations. Let β_0 denote the regression parameter associated with the intercept term of the linear equation (β_0 denotes the regression parameter associated with the intercept term of the linear equation). The logistic regression function can then be denoted by $f(\nu) \in \mathbb{R}^n, 0 \leq f(\nu) \leq 1$, with $\nu \in \mathbb{R}^n$ representing a vector of size n. The logistic regression function $f(\nu_i)$ can then mathematically be formulated as

$$\vartheta_i = f(\nu_i) = \frac{1}{1 + \exp(-\nu_i)} \ \ \forall i \in n, \tag{2.9}$$

where ϑ_i represents the probability prediction based on the value of ν_i as defined in (2.8). The probability prediction ϑ_i can be interpreted as the probability of the i-th observation experiencing the event of interest captured by the binary response variable, or, alternatively, as the probability of having $Y_i = 1$. The function in (2.9) takes on the form of a sigmoid function which relates the linear combination of regression parameter estimates to the binary target domain (Kleinbaum and Klein, 2010). It is important to note that as $\nu_i \to \infty$ we have that $f(\nu_i) \to 1$. Similarly, when $\nu_i \to -\infty$ we will find that $f(\nu_i) \to 0$. Since the logistic regression function is a sigmoid function, it follows an s-shaped curve, which means that as the value for ν_i increases from $-\infty$ towards zero, the probability value $f(\nu_i)$ will sharply increase towards 0.5. As ν_i increases from zero to infinity, the same monotonically increasing trend is enforced whereby the probability will increase from 0.5 towards one. A graphical representation of the s-shaped logistic

regression curve (provided in Figure 2.1) will assist in clarifying the relationship between the linear function in (2.8) and the probability output produced by the logistic regression function $f(\nu_i)$ in (2.9).

To properly grasp the derivation of the log-likelihood formulation, we first need to consider the log-odds function. The log-odds function is derived from the logistic regression formulation by multiplying the right-hand side of (2.9) with $\frac{\exp(\nu_i)}{\exp(\nu_i)}$ and then taking the inverse of the result, which gives

$$\log(\frac{\vartheta_i}{1-\vartheta_i}) = \beta^T X_i. \tag{2.10}$$

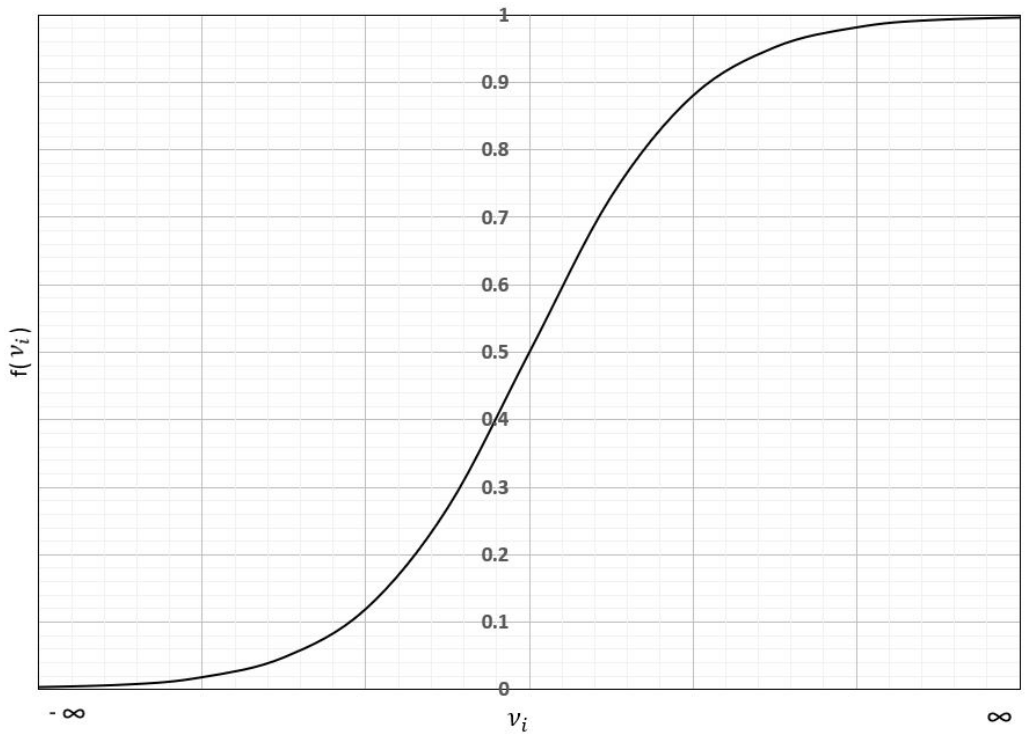


Figure 2.1: Logistic Regression sigmoid function

The equation in (2.10) relates the linear regression model to the log of the odds where the ratio $\frac{\vartheta_i}{1-\vartheta_i}$ is referred to as the odds ratio (Venter, 2020). The probability distribution of Y_i (the response variable) can then be defined according to

$$P(Y_i) = \vartheta_i^{Y_i}(1-\vartheta_i)^{1-Y_i}. \tag{2.11}$$

It is trivial to note that when $Y_i = 1$ the probability distribution $P(Y_i)$ will be equal to ϑ_i. Similarly, (2.11) will reduce to $1 - \vartheta_i$ when $Y_i = 0$. By considering all independent observations, the joint distribution function takes on the form

$$L(Y_1, ..., Y_n) = \prod_{i=1}^{n} P(Y_i) = \prod_{i=1}^{n} \vartheta_i^{Y_i}(1-\vartheta_i)^{1-Y_i}. \tag{2.12}$$

Taking the logarithm on both sides of (2.12) yields

$$\log(L(Y_1, ..., Y_n)) = \log(\prod_{i=1}^{n} \vartheta_i^{Y_i}(1-\vartheta_i)^{1-Y_i}). \tag{2.13}$$

The equation in (2.13) can then be simplified by applying the generic logarithmic rules $\log(x \cdot y) = \log(x) + \log(y)$ and $\log(x^n) = n\log(x)$, assuming that the logarithm base is identical on either side of the equation [2]. The simplified version of (2.13) can then be written as

$$\log(L(Y_1, ..., Y_n)) = \sum_{i \in n} Y_i \log(\vartheta_i) + (1 - Y_i) \log(1 - \vartheta_i). \tag{2.14}$$

Applying the logarithmic rule $\log(\frac{x}{y}) = \log(x) - \log(y)$ to (2.14) leads to

$$\log(L(Y_1, ..., Y_n)) = \sum_{i \in n} Y_i \log(\frac{\vartheta_i}{1 - \vartheta_i}) + \sum_{i \in n} \log(1 - \vartheta_i). \tag{2.15}$$

Following from (2.10) and considering the fact that $1 - \vartheta_i$ can be expressed as

$$1 - \vartheta_i = \frac{1}{1 + \exp(\beta^T X_i)}, \tag{2.16}$$

(2.15) can be modified to arrive at the well-known log-likelihood function

$$\log(L(Y_1, ..., Y_n)) = \sum_{i \in n} Y_i(\beta^T X_i) - \sum_{i \in n} \log(1 + \exp(\beta^T X_i)). \tag{2.17}$$

The log-likelihood function $g(\beta; \mathbf{X}, \mathbf{Y}) = \log(L(Y_1, ..., Y_n))$ in (2.17) can be maximised to estimate the regression parameters (denoted by the vector $\hat{\beta}$) that best describe the underlying binary classification problem. Alternatively, the logistic loss function, defined as

$$h(\beta; \mathbf{X}, \mathbf{Y}) = \sum_{i \in n} \log(1 + \exp(-Y_i(\beta^T X_i)), \tag{2.18}$$

can be minimised with respect to β to produce a result that is identical to the one that is found when $g(\beta; \mathbf{X}, \mathbf{Y})$ from (2.17) is maximised. Since the log-likelihood is concave, the logistic loss function will be convex, which explains why (2.17) should be maximised while (2.18) is minimised [3]. Therefore, the use of either (2.17) or (2.18) depends on the use case being considered, the accompanying constraints and the addition of possible regularisation terms to the objective functions. Section 2.3 of Chapter 2 presents the solution algorithm generally used to maximise the log-likelihood function in (2.17).

2.3 Maximising the log-likelihood function

The non-linear log-likelihood function in (2.17) is globally concave and can be maximised by using the iterative Newton-Raphson method as described in Arnerić et al. (2007). The development of the Newton-Raphson method can be traced as far back as the 16th century, with its evolution detailed in letters, extant notes and publications by Joseph Raphson,

[2] The variables x, y and n used as part of the logarithmic rules are only meant to supplement the reader's intuition and can be considered as placeholders with no inherent meaning. It gives insight into the use of these rules as part of the log-likelihood derivation.

[3] The negative of the logistic loss function can be used interchangeably with the log-likelihood function depending on the problem being considered. Maximising the log-likelihood will produce an identical result to the minimisation of the logistic loss function. Alternatively, one could also maximise the negative logistic loss function to obtain a result that is identical to the maximising of the log-likelihood function.

Isaac Newton and Thomas Simpson (Ypma, 1995). The main objective of maximising the log-likelihood function is to find the most appropriate values for $\hat{\beta}$ that will maximise (2.17). Given the concavity of (2.17), convergence of the Newton-Raphson algorithm towards a global maximum relates to "walking up" a figurative hill that is represented by a contour plot of the log-likelihood. Let T denote the total number of algorithmic iterations required as part of the Newton-Raphson method to ensure convergence is achieved and allow t to denote each individual iteration [4]. If $\hat{\beta}^t$ denotes the regression parameter vector at step t of the Newton-Raphson method, then $\hat{\beta}^{t+1}$ (the subsequent regression parameter vector at iteration $t+1$) can be estimated by a second order Taylor approximation according to

$$\ln(L(\hat{\beta}^{t+1})) = \ln(L(\hat{\beta}^t)) + (\hat{\beta}^{t+1} - \hat{\beta}^t)^T \aleph^t + \frac{1}{2}(\hat{\beta}^{t+1} - \hat{\beta} a^t) H^t (\hat{\beta}^{t+1} - \hat{\beta}^t), \tag{2.19}$$

which can then be used to find the value of $\hat{\beta}^{t+1}$ that will maximise the approximation of (2.19). This is achieved by

$$\frac{\partial ln(L(\hat{\beta}^{t+1}))}{\partial \hat{\beta}^{t+1}} = \aleph^t + H^t(\hat{\beta}^{t+1} - \hat{\beta}^t) = 0, \tag{2.20}$$

$$H^t(\hat{\beta}^{t+1} - \hat{\beta}^t) = -\aleph^t, \tag{2.21}$$

$$(\hat{\beta}^{t+1} - \hat{\beta}^t) = -H^{t^{-1}} \aleph^t, \tag{2.22}$$

$$\hat{\beta}^{t+1} = \hat{\beta}^t + (-H^{t^{-1}} \aleph^t), \tag{2.23}$$

where the value for $\hat{\beta}^{t+1}$ is determined by adding the gradient vector $\aleph^t$ multiplied by the negative of the inverse of the Hessian (denoted by $\{H^t\}^{-1}$). The gradient vector $\aleph^t$ can be defined as a vector of first order partial derivatives such that $\aleph^t = [\frac{\partial ln(L(\hat{\beta}^t))}{\partial \hat{\beta}_1^t}, \frac{\partial ln(L(\hat{\beta}^t))}{\partial \hat{\beta}_2^t}, ..., \frac{\partial ln(L(\hat{\beta}^t))}{\partial \hat{\beta}_p^t}]$. The vector $\aleph^t$ has a size equal to p which (as previously defined) represents the total number of independent variables considered [5]. The negative Hessian evaluates the log-likelihood's curvature by dividing the log-likelihood function's gradient by its curvature. The entries of the Hessian matrix can be defined as the set of second order partial derivatives (associated with the regression parameters at step t of the Newton-Raphson method) denoted by $\{H_{ij}^t\} = \frac{\partial^2 ln(L(\hat{\beta}^t))}{\partial \, hat \beta_i^t \partial \hat{\beta}_j^t}$. The curvature determines the size of the step taken towards the zero gradient, which coincides with the global maximum. The closer the solution produced by each iteration t to the maximum, the smaller the step size considered. Finally, to ensure the log-likelihood function increases at each iteration t, a scalar θ^t is introduced into (2.23). The constant θ^t assists with regulating the step size of each iteration t which leads to the amended Newton-Raphson formulation being expressed as

$$\hat{\beta}^{t+1} = \hat{\beta}^t + \theta^t(-H^{t^{-1}} \aleph^t). \tag{2.24}$$

Arnerić et al. (2007) noted that deriving the Hessian matrix analytically is rather computationally intensive, and, as such, calculating the inverse of the Hessian is possible by

[4] Please note that the superscript t associated with any variable will denote the value for that specific variable at iteration t. In contrast, superscript T refers to the sum of the variable over its specific indices. This convention will be used throughout the thesis.

[5] The Hessian matrix associated with iteration t is denoted by H^t as part of Section 2.3 of Chapter 2, while H (for the remainder of the thesis) will represent an integer conversion constant and will formally be defined in Chapter 5.

utilising quasi-Newton methods. For an overview of these quasi-Newton methods, please see Chapters 3 and 4 in Arnerić et al. (2007). As noted by Venter (2020), the Newton-Raphson method is a locally convergent method which means that the algorithm will converge when the initial regression parameter estimates $\hat{\beta}^0$ are close to the root. However, Hastie et al. (2009) states that overshooting is still a possibility and convergence cannot be guaranteed. Fortunately, since the log-likelihood function is globally concave, the Newton-Raphson method is generally capable of finding the optimal solution to (2.17).

2.4 Summary

An overview of the origin of the logistic regression function and how it is incorporated into the logistic regression modelling framework was provided. This was followed by an explanation of the well-known Newton-Raphson algorithm used to maximise the log-likelihood function in (2.17) in an attempt to fit the logistic regression model to the set of training observations.

Chapter 3

Logistic regression subset selection approaches

3.1 Introduction

This chapter will focus on presenting established variable subset selection algorithms (available in literature) with a key focus on wrapper and embedded selection methods that are used in logistic regression models. The limitations associated with each variable selection algorithm will be discussed in great detail in an attempt to further clarify the literature gap that will be considered as part of this thesis. Each limitation identified will play an integral part in developing the logistic regression variable subset selection algorithm in Chapter 5. The developed solution will aim to mitigate most of the limitations identified in this chapter in an attempt to increase model interpretability and parsimony. Variable selection (within the context of logistic regression) in its simplest form can be defined as maximising $g(\beta; \mathbf{X}, \mathbf{Y})$ in (2.17) or minimising $h(\beta; \mathbf{X}, \mathbf{Y})$ in (2.18) with respect to β (as defined in Chapter 2) in combination with some type of variable subset selection mechanism. Logistic regression subset selection approaches can be classified into stepwise, regularised, and best subset selection algorithms. The three overarching methodologies can be further subdivided into local, heuristic-based and exact approaches. Each methodology will be explained in more detail, starting with stepwise procedures.

3.2 Stepwise variable subset selection

Stepwise procedures are relatively simple to implement and widely used within the model development community. A study conducted in 2004 (which surveyed three leading ecological and behavioural journals) discovered that out of 65 journal papers which used multiple regression approaches, stepwise procedures made up 56% of the proposed algorithms even though there are numerous known and documented limitations associated with these types of procedures (Whittingham et al., 2006). In 2008 a similar survey was conducted

on four leading epidemiological journals, of which 20% used stepwise approaches (Walter and Tiemeier, 2009). Between 2004 and 2008, 44% of variable selection approaches (applied to multiple regression models) published in two leading Chinese epidemiological journals were based on stepwise procedures (Liao and Lynn, 2010). It is clearly evident that stepwise procedures are still regularly used and, as such, a detailed review within the broader context of variable subset selection algorithms is warranted. These procedures are iterative in nature and aim to select a subset of independent variables from a set of p regressors which best explain or describe the dependent variable (Draper and Smith, 1998). For a set of p regressors, a total of 2^p models are possible, which makes exhaustive search an infeasible strategy (Mao, 2004). Therefore, stepwise procedures utilise some form of statistical measure or selection criteria such as the Wald test statistic or information criteria to ascertain the statistical significance and predictive power of the independent variables (Draper and Smith, 1998). Only statistically significant independent variables or variables that improve the objective function (selection criteria) are allowed to enter or remain in the subset of variables. This subset of variables will ultimately be used for model inference. A summary of the underlying theory behind hypothesis testing (within the context of logistic regression) and model selection based on information criteria will be provided, followed by an overview of the most commonly used stepwise procedures available in literature.

P-values are used as part of hypothesis testing to inform the decision whether to reject the null hypothesis (Thiese et al., 2016). P-values are calculated from statistical tests (some of which include the chi-square test, t-test, F-test and KS-test) where the type of statistical test depends on the structure and distribution of the underlying data. Statistical tests (when used specifically to compare two variable distributions) assume that there is no difference between two groups under the null hypothesis and aim to provide a mechanism through which quantitative decisions can be made regarding the extent to which the observed data deviate from the null hypothesis (Thiese et al., 2016). From a practical perspective, p-values are used as a quantitative measure to determine which independent variable should enter or be removed from the model variable subset. A two-sided Wald test is used as the statistical test that underpins the p-value within the context of stepwise selection algorithms utilised in logistic regression. For a sufficiently large set of observations, the Wald test statistic takes the following form:

$$Z^* = \frac{\hat{\beta} - \beta}{s(\hat{\beta})},$$

where $\hat{\beta}$ denotes the maximum likelihood estimate and $s(\hat{\beta})$ the estimated standard deviation of the true regression parameter β. The null hypothesis associated with the test statistic is then given by

$$H_0 : \beta = 0,$$

while the alternative hypothesis H_A states that

$$H_A : \beta \neq 0.$$

The test statistic simplifies to

$$Z^* = \frac{\hat{\beta}}{s(\hat{\beta})}.$$

A well-known alternative to the Wald test is to simply take the square of the Wald test statistic (i.e. the test statistic becomes $(Z^*)^2$), which is then referred to as the Wald chi-squared test (Shahzad et al., 2021). The Wald chi-squared test conforms to a chi-squared distribution with degrees of freedom equal to one (Shahzad et al., 2021). As part of the hypothesis test, a confidence level α should be defined, which is also referred to as the p-value cut-off significance threshold (Venter, 2020). The null hypothesis H_0 can be rejected on the basis that the Wald chi-squared test satisfies

$$(Z^*)^2 > \chi^2_{df}(1-\alpha),$$

with $df = 1$ (where df denotes degrees of freedom) and $100(1-\alpha)$ referring to the percentile associated with the Wald chi-squared distribution. Conventionally, the significance threshold α is set such that $\alpha \in \{0.01, 0.05, 0.1\}$ (Venter, 2020; Shtatland et al., 2003). This is a relatively large range which may be open to interpretation. The choice of α is of a subjective and experimental nature and will influence the variable subset selected for model inference (Shtatland et al., 2003) - this limitation will be further discussed in Section 3.2.4 of Chapter 3. The p-value associated with the test statistic can be derived from the chi-squared distribution table evaluated at point $(Z^*)^2$, after which the null hypothesis H_0 is rejected on the basis that the p-value is less than or equal to α or accepted when the p-value is greater than α. It is important to note that as $(Z^*)^2 \to \infty$, we have that the associated p-values $\to 0$. As such, inference on whether to reject the null hypothesis can be made based on either the Wald chi-squared statistic or the p-values since the p-value is a one-to-one transformation of the Wald statistic. Therefore, the largest Wald statistic will be associated with the smallest p-value and vice versa. This implies that using either the Wald test statistic or the p-value generated by the Wald chi-square test will yield the same conclusion with respect to the inclusion or removal of a variable in/from the model.

Next, the theory behind using information criteria as part of stepwise procedures will be considered. We start with the Akaike information criterion (AIC), which is rooted in the Kullback-Leibler (K-L) information criterion (Burnham et al., 2011). K-L information represents the information that is lost when a candidate model $f_\iota(\cdot)$ is applied in an attempt to describe the true model $f(\cdot)$. The aim of using information theory in variable selection would be to identify the model with the least amount of information loss in relation to full reality or the true model $f(\cdot)$ (Yanagihara et al., 2012). Akaike (1998) proposed a few key contributions which relate the K-L information to the maximised log-likelihood function. This led to the derivation of the well-known AIC, which (for a relatively large number of observations) takes the form [1]

$$AIC = 2h(\beta; \mathbf{X}, \mathbf{Y}) + 2(\sum_{j \in p} z_j),$$

[1]The logistic loss function $h(\beta; \mathbf{X}, \mathbf{Y})$ (defined in Chapter 2) will be considered as part of the AIC formulation since minimising the AIC will produce the best model from the set of candidate models.

where $z \in \{0,1\}^p$ denotes a $p \times 1$ sized vector of binary decision variables with z_j the decision variable associated with the j-th regression parameter β_j. Note that index $j = 0$ represents the intercept term. As such allow z_0 to take on a value of one such that $z_0 = 1$ to ensure the intercept term is included in the model by default. When $z_j = 0$ the regression parameter β_j will be set equal to $\beta_j = 0$, while $\beta_j \in \mathbb{R}$ when $z_j = 1$. The term $2(\sum_{j \in p} z_j)$ is used as a correction for asymptotic bias by acting as a penalisation term which regulates the number of independent variables that can enter the final model (Burnham et al., 2011). If no bias control is added to the AIC formulation, the logistic loss term would monotonically decrease with an increase in the number of independent variables and, therefore, no variable selection would be possible. Even though the general AIC formulation is an asymptotically unbiased estimator, the possibility of bias still exits when the number of observations is relatively small (Yanagihara et al., 2012). Burnham et al. (2011) records an AIC second-order bias correction formulation referred to as AIC_c, which takes the form

$$AIC_c = AIC + \frac{(2(\sum_{j \in p} z_j)((\sum_{j \in p} z_j) + 1))}{n - (\sum_{j \in p} z_j) - 1},$$

where the bias correction term (second term) tends to zero as the number of observations n increases. By critically evaluating the AIC_c given above, it can be inferred that as $n \to \infty$, we have that $AIC_c \to AIC$. In general, stepwise procedures can be grouped into three main categories: forward selection, backward elimination and bi-directional stepwise regression. The type of significance measure or selection criteria employed will slightly influence the practical implementation of the various stepwise procedures as outlined in Sections 3.2.1 to 3.2.3 of Chapter 3. Each of these procedures will be summarised, after which a detailed exposition of the limitations associated with these variable selection methods will be provided.

3.2.1 Forward selection procedure

Let T denote the total number of algorithmic iterations associated with the forward selection stepwise procedure, while t represents each individual iteration. Let $t = 0$ denote the starting point of the stepwise procedure and $t = T$ the point at which the algorithm will conclude. Let C represent a subset of set p defined according to $C \subseteq p$ and can be referred to as the subset of variables from which model inference will be drawn. At iteration $t = 0$, start with an empty logistic regression model only consisting of an intercept term while excluding all of the predictors p. When considering a relatively large set of independent variables, the forward selection stepwise procedure could be advantageous since the procedure starts with the smallest possible model and expands based on inclusion criteria until convergence is reached. This reduces the computational burden associated with evaluating the model in totality (from the start of the stepwise procedure) by considering all independent variables. As part of the stepwise procedure, when an independent variable X_j enters the variable subset C, it cannot be removed in subsequent iterations. The algorithmic logic that underpins the forward selection algorithm is explained simplistically in Algorithm 3.1, while its associated limitations will be discussed in Section 3.2.4 of Chapter

3.

Algorithm 3.1 Forward selection procedure

1: Define an empty set $C \leftarrow \emptyset$.
2: **if** (p-value approach is used) **then**
3: Define significance threshold α.
4: **else if** (IC approach is used) **then**
5: No significance threshold is required.
6: **end if**
7: **for** ($t \leftarrow 0$ to T) **do**
8: **for** ($j \notin C$) **do**
9: At each iteration t build $p - t$ predictive logistic regression models.
10: Each model containing $t + 1$ independent variables consisting of all variables in subset C and one variable X_j from $j \notin C$.
11: Calculate Wald chi-squared statistic $(Z^*)^2$ associated with the generated regression parameter estimates for each logistic regression model.
12: Alternatively, determine the IC values associated with each predictive model.
13: **end for**
14: Select X_j with the highest Wald chi-squared statistic where $j \notin C$.
15: The independent variable X_j should also adhere to the significance threshold which states that the p-value must be less than α or similarly $(Z^*)^2 > \chi^2_{df}(1 - \alpha)$.
16: Alternatively, select variable $X_j \notin C$ which minimises the selected IC.
17: **if** ($(Z^*)^2 < \chi^2_{df}(1 - \alpha)$ or no improvement in the IC is observed) **then**
18: Terminate algorithm
19: **else**
20: Commence with iteration $t + 1$ and add identified independent variable X_j to set C.
21: **end if**
22: **end for**
23: Terminate algorithm and use identified subset of variables for model inference.

3.2.2 Backwards elimination procedure

The backwards elimination procedure will be discussed at the hand of Bursac et al. (2008). It is important to note that the following backwards elimination algorithm is only valid when $p < n$ and will need additional constraints for the case where $p \geq n$. It starts with all independent variables as part of the initial model and utilises a statistical test or selection criteria to assess the effects of each individual parameter. The aim of the backwards elimination procedure is to sequentially remove the least significant variable (which also satisfies the removal threshold) from the set C. This process will continue until no variable meets the removal criteria. It is also important to note that once a variable has been removed from the set C, it will remain excluded until the algorithm terminates, which means that it cannot enter the set C at some future step. Algorithm 3.2 explains the backwards elimination stepwise procedure when either hypothesis testing

or a selection criteria (such as AIC) are used to evaluate the predictive power of each independent variable.

Algorithm 3.2 Backwards elimination procedure

1: Define the set C such that C is of size p.
2: Set C starts out with all independent variables.
3: **if** (p-value approach is used) **then**
4: Define significance threshold α.
5: **else if** (IC approach is used) **then**
6: No significance threshold is required.
7: **end if**
8: **for** ($t \leftarrow 0$ to T) **do**
9: **for** ($j \in C$) **do**
10: At each iteration t build $p - t$ predictive logistic regression models.
11: Each model containing $p - (t + 1)$ independent variables.
12: Calculate the Wald chi-squared statistic $(Z^*)^2$ associated with the generated regression parameter estimates for each logistic regression model and all variables that are in the set C.
13: Alternatively, determine the IC values associated with each predictive model.
14: **end for**
15: Select X_j with the lowest Wald chi-squared statistic from the set C.
16: The variable X_j should also adhere to the significance threshold for p-value $\geq \alpha$ or similarly $(Z^*)^2 < \chi^2_{df}(1 - \alpha)$.
17: Alternatively, select variable X_j from the set C which minimises the selected IC when removed from set C.
18: **if** ($(Z^*)^2 > \chi^2_{df}(1 - \alpha)$ or no improvement in the IC is observed) **then**
19: Terminate algorithm
20: **else**
21: Commence with iteration $t + 1$ and remove identified X_j from set C.
22: **end if**
23: **end for**
24: Terminate algorithm and use identified subset of variables for model inference.

3.2.3 Bi-directional stepwise procedure

The bi-directional stepwise procedure is a combination of the forward selection and backward elimination stepwise procedures. Variables are added to or removed from set C in such a way that each forward stepwise pass is followed by a backward elimination step (Bursac et al., 2008). The algorithm is terminated when none of the variables meet either the inclusion or exclusion criteria or when the variable that entered at iteration t is the same variable removed in the subsequent iteration $t + 1$ (Bursac et al., 2008). The inclusion or removal of independent variables into or from the set C can be considered as being dynamic (based on the test statistic and significance threshold).

Algorithm 3.3 Bi-directional stepwise procedure

Define an empty set $C \leftarrow \emptyset$.
if (p-value approach is used) **then**
  Define significance threshold α_{in} and α_{out}.
else if (IC approach is used) **then**
  No significance threshold is required.
end if
for ($t \leftarrow 0$ to T) **do**
  for ($j \notin C$) **do**
    At each iteration t build $p - t$ predictive logistic regression models. Each model containing $t + 1$ independent variables consisting of all variables in subset C and one variable X_j that is not in set C.
    Calculate the Wald chi-squared statistic $(Z^*)^2$ associated with the generated regression parameter estimates for each logistic regression model. Alternatively, determine the IC values associated with each predictive model.
  end for
  Select X_j with the highest Wald chi-squared statistic from the set of variables not in the set C. It should also adhere to the significance threshold which dictates that the p-value should be less than or equal to α_{in} or similarly $(Z^*)^2 > \chi^2_{df}(1 - \alpha)$. Alternatively, select the variable X_j that is not in the set C which minimises the selected IC when added to set C.
  if ($(Z^*)^2 < \chi^2_{df}(1 - \alpha_{in})$ or no improvement in IC is observed) **then**
    Terminate algorithm
  else
    Commence with backwards elimination step for t and add identified X_j to set C.
  end if
  for ($j \in C$) **do**
    At each iteration t build $|C|$ predictive logistic regression models. Each model containing $C - 1$ independent variables. Calculate the Wald chi-squared statistic $(Z^*)^2$ associated with the generated regression parameter estimates for each logistic regression model. Alternatively, determine the IC values associated with each predictive model.
  end for
  Evaluate all variables in the set C.
  Select X_j with the lowest Wald chi-squared statistic from set C which also satisfies the significance threshold which dictates that the p-value must be greater than or equal to α_{out} or similarly $(Z^*)^2 < \chi^2_{df}(1 - \alpha)$. Alternatively, select variable $X_j \in C$ which minimises the selected IC when removed from set C.
  if ($(Z^*)^2 > \chi^2_{df}(1 - \alpha_{out})$ or no improvement in IC is observed or the variable that was added/removed in iteration $t - 1$ is the same as the variable that was removed/added in iteration t **then**
    Terminate algorithm
  else
    Commence with iteration $t + 1$ and remove identified X_j (if any) from set C.
  end if
end for
Terminate algorithm
Use identified subset of variables for model inference.

This procedure has the added benefit of accommodating changes in the test statistic based on some variable interactions. Algorithm 3.3 describes the bi-directional stepwise procedure while clarifying the amalgamation of the forward selection and backward elimination stepwise procedures.

3.2.4 Stepwise limitations

The limitations associated with stepwise procedures have been well documented in the literature and will be summarised in this section. One of the main drawbacks of stepwise regression is that it results in biased regression parameters which can lead to overfitting and incorrect significance tests (Whittingham et al., 2006). Stepwise selection suffers from a phenomenon known as model selection bias, which occurs when the inclusion or removal of variables is based on information relevant to individual variables without any referenence to the suite of possible models (Whittingham et al., 2006). Individual variables (although non-significant in isolation) may be jointly significant when considered alongside several variables in the variable set. As part of regression, it is, therefore, important to not only evaluate variables in isolation (which will lead to bias) but also to consider multivariate effects as well (Antonakis and Dietz, 2011). When a hypothesis test concludes that a specific variable is not significant, a value of zero is accepted, which dramatically underestimates the variable's contribution to the final model. Similarly, values in excess of the true value of the regression parameter estimate are accepted when the variable is considered to be significant. By not considering true variable interactions as part of the stepwise selection process, biased estimates can be produced as a result. Stepwise procedures also generate notoriously narrow confidence intervals around the parameter estimates, which may cause a false sense of security in the proposed model (Whittingham et al., 2006). Furthermore, Steyerberg et al. (1999) reports that the variance associated with regression parameter estimates is underestimated when model uncertainty is disregarded. This implies that hypothesis tests will draw incorrect conclusions.

Another widely recognised limitation of stepwise approaches is that the order in which the independent variables enter the stepwise selection procedure can influence the final model selected for inference. Whittingham et al. (2006) reports that this is especially prevalent when multicollinearity is present between independent variables. This highlights the fact that the solutions proposed by stepwise procedures cannot guarantee optimality. As such, the subset of variables contained in the final solution can potentially be far-removed from the optimal subset of variables (Smith, 2018). Stepwise procedures can be classified as variable selection heuristics and do not fit into the paradigm of deterministic exact solution algorithms. Placing a measure of confidence on the model selected by a stepwise procedure is not palatable since a different set of variables may be able to perform equally well on the selected training and holdout sets (Whittingham et al., 2006). A model that employs either the forward or backward selection stepwise procedure can also contain redundant variables that do not add significant predictive value. This is due to the selection procedures not allowing variables to re-enter and/or be removed from the variable subset when removed/included in subsequent iterations (Mao, 2004).

Another important consideration is that model selection via stepwise procedures is achieved by means of multiple hypothesis tests and not a single test (Whittingham et al., 2006). Stepwise procedures generally lead to underestimated p-values (Venter, 2020). Underestimated p-values exacerbate the occurrence of false positive results, also known as type-I errors (Wilkinson, 1979). This means that the null hypothesis is rejected too often when, in reality, it should be accepted. The selection of independent variables based on the hypothesis tests may therefore result in the inclusion of variables where the observed effect occurred by chance. Including variables that should have been rejected from the final variable subset will ultimately lead to overfitting on the training dataset and yield less parsimonious models (Whittingham et al., 2006).

Inferences based on data-driven selection procedures and the number of authentic variables included in the final subset of variables depend on the number of observations (Antonakis and Dietz, 2011; Derksen and Keselman, 1992). This is substantiated by the fact that as the number of observations increases, a hypothesis test is more likely to find a significant effect (Park, 2015; Thiese et al., 2016). This implies that the effects observed when utilising a large dataset are less likely to be the result of chance since a larger dataset better approximates the true underlying population. Given that the inclusion/removal of variables in stepwise procedures are largely dependent on the outcome of hypothesis tests, we note that differences in the final subset of variables chosen by the model may arise due to changes in the size of the modelling dataset (or number of observations considered). This is especially prevalent when the number of observations is small because the standard deviation of the estimated mean (true population mean) may increase. For a small number of observations, the danger of random error is also greater, which gives less certainty with respect to the effects observed.

Another point of contention is the subjective nature of the p-value significance threshold α. In general, the significance level is often selected in practice and in academia such that $\alpha \in \{0.01, 0.05, 0.1\}$ (Costanza and Afifi, 1979). Broadly applying the same threshold α to various scenarios is not advised and may lead to misleading results (Venter, 2020). The field of study or modelling scenario being considered may dictate the choice of α. When a type-I error could potentially result in catastrophic consequences, it would be advisable to opt for a more stringent threshold such as $\alpha < 0.01$. The choice of α will, however, greatly affect the final subset of variables proposed by the stepwise selection procedure. A relatively relaxed threshold will result in a large number of variables being classified as significant. This will yield an inflated variable subset which may lead to overfitting on the training set. Alternatively, when the threshold α is too strict, it may exclude predictors potentially correlated with the target variable. In either scenario, overfitting or losing predictive and explanatory power is an unwanted property of a variable selection algorithm. Variable selection should aim to select a parsimonious model with adequate generalisation performance on unseen data.

Halsey (2019) noted that p-values could be disregarded altogether and replaced with the AIC (as discussed in Section 3.2 of Chapter 3), which is an adequate measure to use when having to choose between multiple candidate models. The AIC measure is mathematically

related to the p-value in the sense that both the AIC and p-values are different transformations of the likelihood ratio (Halsey, 2019). However, one of the main limitations of AIC is that it provides a relative measure of model quality (Halsey, 2019). Halsey (2019) correctly noted that it is not always the case that the best model is a good model for the dataset under consideration. It is also well documented that the AIC measure tends to favour less parsimonious models (Vrieze, 2012). Vrieze (2012) proves this fact by comparing the log of the maximum likelihood estimate for a freely estimated model (denoted by MLE_1) with that of a restricted model (denoted by MLE_0) for which the covariance is set equal to zero. Vrieze (2012) noted that $(-2\log(MLE_0)) - (-2\log(MLE_1))$ follows a χ^2 distribution with a p-value of 0.05. This relates to a difference in likelihood of 3.84 with degrees of freedom equal to one. It is known beforehand that the restricted model MLE_0 has zero parameter estimates (since the covariance has been fixed to zero) while MLE_1 only has one. Considering this, Vrieze (2012) goes on to prove that for the restricted model to be selected over the complete model, the AIC of the restricted model should be less than that of the full model. This will only happen when $(-2\log(MLE_0) + 2(\sum_{j\in p} z_j)_0) - (-2\log(MLE_1) + 2(\sum_{j\in p} z_j)_1)$ is positive. Since $(\sum_{j\in p} z_j)_0 = 0$ and $(\sum_{j\in p} z_j)_1 = 1$, it can be shown that the second term associated with both reduces to $(-2\log(MLE_0)+2(0))$ and $-(-2\log(MLE_1) + 2(1))$ which leads to $(2(0) - 2(1)) = -2$. This suggests that the likelihood difference $(\log(MLE_0)) - (\log(MLE_1))$ should be greater than 2, which relates to a $\chi^2_1(84)$ distribution with a significance level equal to 0.16 (Vrieze, 2012). For the sake of clarity, $\chi^2_1(84)$ represents a Chi-square distribution with degrees of freedom equal to 1 and a significance level of 0.16 (as defined earlier in this Chapter by $100(1-\alpha)$, which represents the percentile associated with the chi-squared distribution) [2]. A p-value of 0.16 indicates that the AIC measure will tend to overestimate the model dimensions and may lead to overfitting on the training set (Karagrigoriou et al., 2011). In general, the AIC measure does not face all the same limitations as p-values. However, when used within the context of stepwise regression, it still faces some of the same inherent challenges as discussed throughout Section 3.2 of Chapter 3.

3.3 Regularised variable subset selection

An alternative to stepwise variable selection is to select the most predictive variables by regularisation or shrinkage of the regression parameter estimates (James et al., 2013). Regularisation techniques aim to find the most appropriate subset of variables by either maximising the log-likelihood or minimising the logistic loss while penalising the magnitude of the regression parameter estimates. In general, the procedures tend to shrink the irrelevant regression parameter estimates towards zero (Pereira et al., 2016). As a consequence of parameter shrinkage, model variance is reduced and variable selection is inherently enforced (Pereira et al., 2016). One of the main advantages of regularised variable subset selection methods is the reduced computational burden associated with finding an adequate subset of variables by not having to evaluate 2^p models. This is because reg-

[2] When evaluated on the Chi-squared table it is clear that a chi-squared distribution with degree of freedom equal to one and a significance level of 0.16 relates to a chi-square value of 2.

ularised variable selection methods solve a single model with a fixed penalty term λ with all predictors included as part of the variable set (James et al., 2013).

There are six main regularisation-based penalty terms that will be considered as part of this chapter, which include the least absolute shrinkage and selection operator (lasso) (also known as the L_1-norm penalty), ridge regression (also referred to as the L_2-norm penalty), the elastic net penalty, the $L_{\frac{1}{2}}$-penalty, the smoothly clipped absolute deviation (SCAD) penalty and, finally, the minmax concave penalty (MCP). Each of these will be discussed in detail in the remainder of this subsection.

3.3.1 The lasso, ridge regression and special case penalties

In general, lasso and ridge regression have a similar form with only slight differences associated with the regression parameter regularisation strategy (Pereira et al., 2016). Variable subset selection approaches that utilise either lasso or ridge regression parameter regularisation can fit a model to the data by minimising the objective function given by

$$\min_{\beta} \quad h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_q ||\beta||_{q_2}^{q_1}, \tag{3.1}$$

where $h(\beta; \mathbf{X}, \mathbf{Y})$ represents the logistic loss function (defined in Chapter 2) and λ_q a regularisation-specific penalty parameter. The subscript q is set equal to one when considering the lasso penalty and two for the ridge penalty. The notation λ_q is also used in the remainder of this chapter to denote other penalty terms and will be defined based on the choice of q. The choice of q_1 and q_2 determine the type of norm that $||\beta||_{q_2}^{q_1}$ represents and governs the extent of the regularisation. When $q_1 = 1$ and $q_2 = 1$, the L_1-norm is used during regularisation and we obtain a lasso regression model. Alternatively, when $q_1 = 2$ and $q_2 = 2$, the regularisation makes use of the L_2-norm, which yields a ridge regression model. Essentially, the lasso attempts to

$$\min_{\beta} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_1 ||\beta||_1, \tag{3.2}$$

where

$$||\beta||_1 = \sum_{j \in p} |\beta_j|, \tag{3.3}$$

while ridge regression attempts to

$$\min_{\beta} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_2 ||\beta||_2^2, \tag{3.4}$$

where

$$||\beta||_2^2 = \sum_{j \in p} \beta_j^2. \tag{3.5}$$

An equivalent formulation using the log-likelihood function can be defined by maximising $g(\beta; \mathbf{X}, \mathbf{Y})$ while subtracting the penalty terms in (3.2) and (3.4). Let $\lambda_1 \geq 0$ denote the lasso penalty and $\lambda_2 \geq 0$ the ridge regression penalty. The idea behind the penalty term is to regularise the magnitude of the regression parameter estimates. An increase in either λ_1 or λ_2 will cause the sum of the regression parameter estimates $\hat{\beta}$ to shrink towards zero.

Therefore, the formulation will simultaneously optimise for model fit and parsimony. This is substantiated by the fact that even though a single penalty term is applied to the sum of the regression parameter estimates, a varying degree of shrinkage will be observed for each individual estimate (Steyerberg et al., 2001). In both cases, the penalty term is only applied to the regression parameter estimates associated with each independent variable and will exclude the model intercept (Steyerberg et al., 2001). It is well known that the lasso formulation acts as a true variable subset selection algorithm by setting some of the regression parameter estimates exactly equal to zero. This is not the case when the ridge penalty is used. Although ridge regression tends to reduce model complexity, it does not function well as a variable selection algorithm since it will only minimise the regression parameter estimates without setting any estimate exactly equal to zero. Ridge regression can, therefore, not be used to discover parsimonious models from the set of 2^p candidate models. It only aims to reduce the influence of less important regressors on the final model used for inference.

Even though the lasso formulation reduces model complexity and selects a subset of variables from the set of p predictors, it is still plagued by several limitations.

1. The first is related to a scenario where the problem under consideration contains more independent variables p than the number of observations n such that $n < p$. In this scenario, the lasso formulation tends to select at most $\min(p, n)$ independent variables as part of the final subset of variables before it is saturated (Zhang et al., 2015). This limits lasso's application to sparse problem spaces (Fonti and Belitser, 2017).

2. The second limitation becomes evident when the set of variables is made up of a group of highly correlated variables. Lasso will tend to select only one variable from the group of correlated variables while setting the regression parameter estimates of the remaining variables in the group equal to zero (Zhang et al., 2015). While this is not necessarily a problem, it should be noted that lasso often makes this choice arbitrarily. This means that lasso will not necessarily select the variable from the group of correlated variables that best describes the target. As such, selecting a sub-optimal variable from the correlated group will cause a reduction in the overall model performance and may lead to the user erroneously concluding that the subset selected contains the most important variables.

3. Empirical experiments also show that when the number of observations n exceeds the number of independent variables p, lasso's performance may be compromised in the presence of severe collinearity between independent variables (Venter, 2020).

4. According to Herlands et al. (2015) the lasso or L_1-penalty is a convex approximation of the non-convex L_0-norm (which is used during best subset selection). The authors note that the L_1-norm penalty, when applied to real-world datasets, tends to select incorrect models due to the shrinkage effect of the norm on all variables (even those in the active set). This especially becomes a problem in sparse settings where the predictive risk of L_1-penalties is arbitrarily worse than that of the L_0-norm.

5. Another limitation of lasso and ridge regression relates to the subjective nature of manually selecting the penalty parameters λ_1 and λ_2. The subjectiveness of manually selecting these parameters can be mitigated by employing a combination of cross-validation and grid-search techniques to identify an adequate value for the penalty parameter based on the model's performance on unseen data. This approach is, however, an iterative heuristic by nature and cannot provide an optimality guarantee.

6. The choice of λ_1 for lasso controls the cardinality of the variable subset selected for model inference. Bertsimas et al. (2016), however, noted that there is no clear relationship between the value of λ_1 and the cardinality of the variable subset selected. The only correlation that is known is that a larger value for λ_1 corresponds to a sparser final model. However, it cannot explicitly control the number of variables in the final subset.

As an alternative to the lasso and ridge regression penalties, Liang et al. (2013) proposed the $L_{\frac{1}{2}}$ regularisation term for use in logistic regression (which, in turn, is derived from the formulation proposed by Xu et al. (2010)). The main idea behind the proposed penalty is to generate sparser results that are more closely related to the use of the L_0-norm while not being as computationally intractable (Xu et al., 2010). The $L_{\frac{1}{2}}$ formulation attempts to

$$\min_{\beta} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_{\frac{1}{2}} \sum_{j \in p} |\beta_j|^{\frac{1}{2}}. \tag{3.6}$$

The magnitude of the regularisation term is controlled by $\lambda_{\frac{1}{2}}$, with $\lambda_{\frac{1}{2}} > 0$. Empirical results presented by Liang et al. (2013) demonstrate the effectiveness of the $L_{\frac{1}{2}}$-penalty in solving small-scale non-linear logistic regression problems. The regularisation term aims to improve the sparsity of the final model selection in comparison to L_1-regularisation while reducing the expensive combinatory optimisation associated with L_0-regularisation (the L_0-penalty is NP-hard) and can be considered intractable for high-dimensional problems (Xu et al., 2010). Xu et al. (2010) showed that the $L_{\frac{1}{2}}$-penalty has some of the advantageous properties of both the L_0- and L_1-penalties while also noting that it possesses sparsity, unbiasedness and oracle properties, which are preferred characteristics of a variable selection algorithm. The $L_{\frac{1}{2}}$-penalty can be considered as a special case of L_q regularisation, with $(0 < q < 1)$ (Liang et al., 2013). The objective function in (3.6) is, however, non-convex due to the form of the $L_{\frac{1}{2}}$-penalty term. Xu et al. (2010) propose an iterative heuristic-based solution algorithm towards solving the $L_{\frac{1}{2}}$-penalty, which transforms (3.6) into a series of weighted lasso problems to which existing lasso solution algorithms can be applied. Since the solution algorithm is heuristic by nature, it will tend to converge towards local minima (instead of a global minimum) while also being incapable of providing a measure of optimality. Xu et al. (2010), however, showed through empirical results that the $L_{\frac{1}{2}}$-penalty produces solutions that outperform the L_1-penalty with increased model sparsity. As such, the added computational complexity associated with the non-convex objective function can be overlooked, considering the performance of the proposed regularisation formulation. In general, the limitations of the approach can be summarised as follows:

1. The $L_{\frac{1}{2}}$-penalty results in a non-convex objective function which requires specialised solution algorithms to ensure convergence towards a high-quality solution. The computational complexity of the $L_{\frac{1}{2}}$-penalty is, however, slightly less than that of L_0-regularisation.

2. The heuristic-based solution algorithm proposed by Xu et al. (2010) converges towards local minima, which suggests that better solutions exist within the solution search space that are not discovered by the algorithm.

3. The proposed solution framework is incapable of providing an optimality guarantee. As such, the user of the solution algorithm has no sense of the solution quality and how it compares to the optimal solution.

4. The $L_{\frac{1}{2}}$-penalty suffers from some of the same limitations as the lasso and ridge regression penalties (specifically limitations five and six on the previous page).

Even though the $L_{\frac{1}{2}}$-penalty seems to exhibit some favourable characteristics as a subset selection procedure (improving on both the L_1- and L_2-regularisation formulations with regards to model sparsity and overall predictive performance) it still falls short in some respects when compared to a true best subset selection methodology.

3.3.2 The elastic net penalty

The elastic net is a regularisation approach that linearly combines the penalty term of both the lasso and ridge regression formulations, where the objective function is formulated as

$$\min_{\beta} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_2 ||\beta||_2^2 + \lambda_1 ||\beta||_1. \tag{3.7}$$

The main aim of the elastic net formulation is to address some of the key limitations associated with the lasso, as described in Section 3.3.1 of Chapter 3, while retaining the beneficial characteristics associated with both the lasso and ridge regression penalties. This leads to a regularised formulation which is capable of

1. handling problems where $p > n$,

2. dealing with correlated predictor variables while also being able to include group structures present in the variable set (Venter, 2020),

3. solving a subset selection problem with $n > p$ while enjoying reduced computational complexity when compared to best subset selection techniques.

It is easy to note that when λ_2 is equal to one and λ_1 is equal to zero, the elastic net regularised formulation takes the form of the ridge regression penalty. Alternatively, the lasso penalty is obtained when λ_2 is equal to zero and λ_1 is equal to one. Venter (2020) and Zou and Hastie (2005) noted that the elastic net only performs well when it closely

resembles either lasso or ridge regression formulations. As a result, it does not add significantly more value (from a variable selection perspective) when compared to its lasso and ridge regession counterparts (Venter, 2020). As such, Zou and Hastie (2005) renamed the standard elastic net formulation the naive elastic net. Zou and Hastie (2005) also noted a double shrinkage effect (which introduces unnecessary parameter bias) due to the two-step solution algorithm used to solve the naive elastic net. The first step attempts to find the ridge regression parameters for a fixed value of λ_2, after which a lasso-type shrinkage is applied. To address this, Zou and Hastie (2005) proposed a transformation of the naive elastic net (within the context of linear regression), which will be discussed shortly. Let us first assume that the response vector is centred such that $\sum_{i\in n} Y_i = 0$ and the columns of the design matrix are standardised such that $\sum_{i\in n} X_{ij} = 0$ and $\sum_{i\in n} X_{ij}^2 = 0$ with λ_1 and λ_2 non-negative. The objective function of the elastic net (within the context of linear regression) is then given by

$$\min_{\beta} ||\mathbf{Y} - \mathbf{X}\beta||_2^2 + \lambda_2||\beta||_2^2 + \lambda_1||\beta||_1, \tag{3.8}$$

where

$$||\mathbf{Y} - \mathbf{X}\beta||_2^2 = \sum_{i\in n}(Y_i - \sum_{j\in p}\beta_j X_{ij})^2. \tag{3.9}$$

Finding the naive estimates $\hat{\beta}$ requires the minimising of (3.8), which is equivalent to a penalised least squares model (Zou and Hastie, 2005). Zou and Hastie (2005) defined a coefficient $\varrho = \frac{\lambda_2}{\lambda_1+\lambda_2}$ such that the substitution of ϱ in the place of λ_1 and λ_2 yields an equivalent problem that is defined as

$$\hat{\beta} = \min_{\beta} ||\mathbf{Y} - \mathbf{X}\beta||_2^2 + \varpi, \tag{3.10}$$

subject to

$$(1-\varrho)||\beta||_1 + \varrho||\beta||_2^2 \leq \varpi, \tag{3.11}$$

where the left-hand side of the constraint in (3.11) is referred to as the naive elastic net penalty and the right hand side of (3.11) (represented by the decision variable ϖ) is used as part of the objective function to ensure that regularisation is enforced. It is a convex combination of the well-known lasso and ridge regression penalties. As part of the derivation provided by Zou and Hastie (2005), ϱ is specified such that $0 < \varrho < 1$. This will ensure that both lasso and ridge regression penalties are consistently present within the formulation. When $\varrho = 0$ the ridge regression penalty is removed, while $\varrho = 1$ results in the exclusion of the lasso penalty. In an attempt to address the double shrinkage effect inherent to the naive elastic net, a transformation is proposed such that

$$\beta^{\text{elastic } net} = \beta^{\text{naive elastic net}}(1+\lambda_2). \tag{3.12}$$

The transformation is able to preserve the variable selection property associated with the naive elastic net while reducing regression parameter bias. Rescaling the naive elastic net by using $(1+\lambda_2)$ is justified when considering an exact solution of the naive elastic net where the predictors are orthogonal. In this case, it is well-known that the lasso is minimax optimal, implying that the naive elastic net is sub-optimal. However, after scaling with the help of $(1+\lambda_2)$, minmax optimality of the elastic net is achieved. Rescaling via $(1+\lambda_2)$

is motivated by a decomposition of the ridge regression operator. Given that the design matrix of predictors is standardised, it can be shown that

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 1 & \rho_{12} & \dots & \rho_{1p} \\ \vdots & \ddots & & \vdots \\ & & 1 & \rho_{p-1,p} \\ & & & 1 \end{bmatrix}_{p \times p}$$

where ρ_{ij} denotes the sample correlation that is given by $\rho = \mathbf{X}_i'\mathbf{X}_j$, with $\mathbf{X}'$ describing the transpose of the original design matrix. The ridge regression parameter estimate $\hat{\beta}^{\text{ridge}}$ for the penalty parameter λ_2 is then given by $\hat{\beta}^{\text{ridge}} = \mathbf{R}\mathbf{Y}$, with $\mathbf{R} = (\mathbf{X}'\mathbf{X} + \lambda_2\mathbf{I})^{-1}\mathbf{X}'$ and $\mathbf{I}$ the identity matrix. We can then rewrite $\mathbf{R}$ as

$$\mathbf{R} = \frac{1}{1+\lambda_2}\mathbf{R}^* = \frac{1}{1+\lambda_2} \begin{bmatrix} 1 & \frac{\rho_{12}}{1+\lambda_2} & \dots & \frac{\rho_{1p}}{1+\lambda_2} \\ \vdots & \ddots & & \vdots \\ & & 1 & \frac{\rho_{p-1,p}}{1+\lambda_2} \\ & & & 1 \end{bmatrix} \mathbf{X}'$$

with $\mathbf{R}^*$ describing the usual ordinary least squares estimate, where the sample correlations ρ_{ij} are shrunk by a factor of $\frac{1}{1+\lambda_2}$. Zou and Hastie (2005) refer to this as "decorrelation". The ridge regression operator can therefore be described as decorrelation followed by scaling shrinkage. It is noted by Zou and Hastie (2005) that when combining the ridge regression shrinkage step with the lasso, the direct shrinkage effect is not needed and is eliminated by rescaling. The proposed method, therefore, utilises the lasso for both parameter shrinkage and variable selection. This suggests that the elastic net penalty does not require the shrinkage step of ridge regression. As such, the stabilised elastic net solution vector of regression parameter estimates $\hat{\beta}^{\text{elastic net}}$ can therefore be derived by solving

$$\min_{\beta} \beta'(\frac{\mathbf{X}'\mathbf{X} + \lambda_2\mathbf{I}}{1+\lambda_2})\beta - 2\mathbf{Y}'\mathbf{X}\beta + \lambda_1||\beta||_1, \tag{3.13}$$

which is closely related to the lasso estimates β^{lasso} that is obtained by solving

$$\min_{\beta} \beta'(\mathbf{X}'\mathbf{X})\beta - 2\mathbf{Y}'\mathbf{X}\beta + \lambda_1||\beta||_1. \tag{3.14}$$

The elastic net in (3.13) can therefore be seen as a stabilised version of the lasso penalty given in (3.14), where $\mathbf{X}'\mathbf{X}$ in (3.14) is replaced with its shrunken version $\frac{\mathbf{X}'\mathbf{X}+\lambda_2\mathbf{I}}{1+\lambda_2}$. The proposed formulation in (3.13) addresses the double shrinkage effect of the naive elastic net while still promoting model sparsity. The elastic net formulation is solved by utilising a novel least angle regression elastic net (LARS-EN) algorithm that is derived from the well-known least angle regression (LARS) algorithm proposed by Efron et al. (2004). For a detailed overview of the proposed solution algorithm, please refer to Efron et al. (2004).

Although the elastic net formulation outperforms the individual L_1- and L_2-penalties, it does have a few limitations that should be considered, which are discussed next.

1. As noted by Zou and Hastie (2005), the naive elastic net has a double shrinkage effect due to its two-step solution approach. To mitigate this, a transformation of

the original formulation is required. However, even addressing the double shrinkage effect still results in the elastic net introducing bias into the regression parameter estimates, which impedes statistical inference. Post-hoc regression analysis can be applied by using the independent variables identified by the elastic net as inputs in a straightforward regression model that does not make use of penalty terms. This will ensure unbiased regression parameter estimates, which makes statistical inference possible. Nevertheless, such an approach neglects to acknowledge that variable selection is still based on biased estimates, which in turn affects the final variables included in the model.

2. The elastic net performs poorly in the absence of scaling, especially if its penalty parameters do not closely reflect the parameters used in ridge regression (Venter, 2020).

3. In some cases, the solution algorithms used to solve the regularised formulations are computationally intensive, making it intractable for large-dimensional problems. This is true in the case of the LARS-EN algorithm that is used to solve the elastic net proposed by Zou and Hastie (2005). The computational requirements of the solution algorithm increase significantly when $p > n$ and, in some cases, require early stopping (Venter, 2020).

4. Similar to lasso and ridge regression, the penalty parameters λ_1 and λ_2 can be estimated by employing a combination of cross-validation and grid search techniques. However, k-folds cross-validation would be more appropriate given that the elastic net requires a two-dimensional search. As such, the LARS-EN algorithm will require k-fold cross-validation over a two-dimensional surface, which greatly increases the number of regression models that need to be estimated for evaluation (Venter, 2020; Zou and Hastie, 2005).

5. The approach used to estimate the penalty parameters (similar to lasso or ridge regression) is an iterative heuristic by nature and cannot provide an optimality guarantee over the full range of both λ_1 and λ_2, as all possible combinations cannot be considered simultaneously.

Regardless of the limitations noted, it is clear that the elastic net penalty is able to select a subset of variables from a set of p predictors while exhibiting greater flexibility in comparison to the lasso penalty, given that it can be applied to problems that are sparse and contain highly correlated independent variables.

3.3.3 SCAD and MCP penalties

In an attempt to address the regression parameter bias associated with the lasso, ridge regression and elastic net penalties (especially when the regression parameter estimates are large in magnitude), multiple alternative regularisation approaches have been proposed in literature (see e.g. Fan and Li (2001); Zhang (2010)). These include the SCAD and

MCP penalties that are known to be non-convex penalty terms. The SCAD penalty will be explained at the hand of Fan and Li (2001). Firstly, let us consider a regularised linear regression problem with objective function given by

$$\min_{\beta} ||\mathbf{Y} - \mathbf{X}\beta||_2^2 + J_\lambda(\beta), \tag{3.15}$$

where $J_\lambda(\cdot) > 0$ represents a penalty function. It is important to note that even though the formulation was defined for a linear regression problem, it can equivalently be applied to logistic regression problems. Specifying $\varsigma > 2$ and $\lambda \geq 0$ (where λ denotes the penalty used as part of the SCAD or MCP function) yields a penalty function $J_\lambda(\cdot)$ that can be defined as the SCAD penalty and is given by

$$J_{\lambda,\varsigma}(\beta) = \begin{cases} \lambda|\beta|, & \text{if } |\beta| \leq \lambda, \\ \frac{(2\varsigma\lambda|\beta| - \beta^2 - \lambda^2)}{2(\varsigma-1)}, & \text{if } \lambda < |\beta| \leq \varsigma\lambda, \\ \frac{\lambda^2(\varsigma+1)}{2}, & \text{if } |\beta| > \varsigma\lambda. \end{cases} \tag{3.16}$$

The SCAD penalty relates to a quadratic spline function with knots at $[-\varsigma\lambda, \varsigma\lambda]$ and $[-\lambda, \lambda]$ (Fan and Li, 2001). The proposed penalty function prevents excessive penalisation of large β values, which makes the solution continuous. The SCAD derivative vanishes outside the range $[-\varsigma\lambda, \varsigma\lambda]$ which results in sparse solutions that produces unbiased estimates for large coefficients. The SCAD penalty corresponds to the lasso where $|\beta| = \lambda$, smoothly switches to a quadratic function for $|\beta| = \varsigma\lambda$ and remains constant for the remainder of the range where $|\beta| > \varsigma\lambda$. This means that the SCAD penalty reduces parameter bias for large regression coefficients and retains the bias and rate of regularisation for small regression coefficients. Xie and Huang (2009) observed that the SCAD penalty is continuously differentiable throughout the range $[-\infty, \infty]$ and is singular at zero. The complete derivation of the SCAD penalty is available in Fan and Li (2001). Next, we consider the minimax convex penalty (MCP) which is closely related to the SCAD penalty. The MCP, with $\varsigma > 1$ and $\lambda \geq 0$, is defined as

$$J_{\lambda,\varsigma}(\beta) = \begin{cases} \lambda|\beta| - \frac{\beta^2}{2\varsigma}, & \text{if } |\beta| \leq \varsigma\lambda, \\ \frac{1}{2}\varsigma\lambda^2, & \text{if } |\beta| > \varsigma\lambda. \end{cases} \tag{3.17}$$

The MCP, similar to the SCAD penalty, starts by applying a penalty that is equivalent to the lasso, after which the penalty rate is reduced to zero as the absolute value of the regression parameter $|\beta| \to \infty$. It is important to note that the MCP reduces the bias associated with the estimates while it increases concavity as the tuning parameter ς increases. For both SCAD and MCP penalties, λ controls the degree of regularisation (similar to lasso and ridge regression) while ς determines the rate at which the penalty term tends to zero (Venter, 2020). As $\varsigma \to \infty$, the MCP penalty starts to resemble the lasso penalty. Alternatively, when $\varsigma = 1$, MCP performs like a L_0-penalty (Venter, 2020). The same is true for SCAD when $\varsigma \to \infty$ (Venter, 2020). The regularisation effect of the lasso, SCAD and MCP penalties (on the regression parameter estimates $\hat{\beta}$) can be observed in Figure 3.1.

Figure 3.1 was generated by selecting $\varsigma = 2$ and $\lambda = 0.5$. Manually choosing appropriate values for the regularisation parameters is rather subjective in nature and can influence

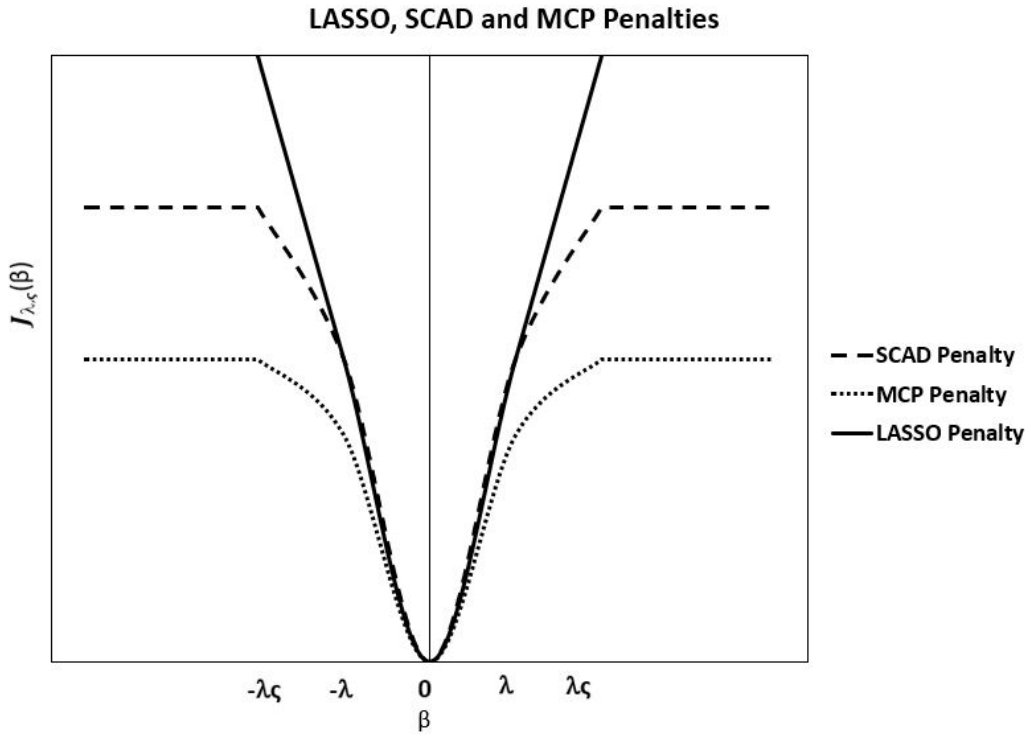


Figure 3.1: Lasso, SCAD and MCP penalties

the rate ς and magnitude of regularisation λ. Therefore, k-fold cross-validation and generalised cross-validation are used to estimate the most appropriate hyperparameters for the problem being considered (Breiman, 1995; Fu, 1998; Varoquaux et al., 2017; Park and Hastie, 2008). Generalised cross-validation can be used to determine λ for the lasso and ridge regression penalty terms, while k-fold cross-validation would be more appropriate for SCAD and MCP penalties. Since SCAD and MCP utilise two hyperparameters, a search over a two-dimensional grid that spans various (λ, ς) coordinates will be required (Breiman, 1995; Fu, 1998; Varoquaux et al., 2017; Park and Hastie, 2008; Fan and Li, 2001). Fan and Li (2001) proposed that the two-dimensional search should be conducted through five-fold cross-validation. The authors then state that a constant value of $\varsigma = 3.7$ seems to be a reasonable choice, especially for linear regression problems. However, as correctly noted by Venter (2020), this assumption might not hold for logistic regression problems. Simulations conducted by Fan and Li (2001) indicate that the SCAD penalty outperforms the lasso and ridge regression penalties when noise in the training dataset is reduced. Nevertheless, the lasso seems to perform best when the number of observations n is small and the noise level is high.

Even though the SCAD and MCP penalties address some of the regression parameter bias associated with lasso and ridge regression, it still falls short in some respects. Since the SCAD and MCP penalties are both non-convex, there is no guarantee that the algorithms will converge to a global optimum, especially in the case of logistic regression (Venter, 2020). Fan and Li (2001) proposed solving the non-convex optimisation problem by utilising a local quadratic approximation (LQA) algorithm. The algorithm attempts to locally approximate the penalty function with a quadratic function (Breheny and Huang,

2011). Fan and Li (2001), however, noted that the LQA algorithm suffers from the same drawback associated with backward elimination stepwise procedures in the sense that once an independent variable has been excluded from the variable set, it will no longer be considered in subsequent iterations. Some modifications to the LQA algorithm have been proposed, as noted by Zou and Li (2008), to address the limitations associated with the LQA algorithm. One of these includes solving a perturbed version of the LQA algorithm, as proposed by Hunter and Li (2005). Another alternative was proposed by Fan and Li (2001) in which the LQA solution is approximated by using one-step or k-step estimates. For the former, selecting the size of the perturbations becomes rather cumbersome while the latter lacks a sparse representation (Zou and Li, 2008). As such, Zou and Li (2008) proposed an iterative local linear approximation algorithm which improves on the lack of sparsity associated with the LQA one-step estimator.

Coordinate descent is another solution algorithm applied to solve the lasso regularisation problem. It was shown by Friedman et al. (2007) that the coordinate descent algorithm can be seen as an efficient competitor to the well-known LARS algorithm. As such, Breheny and Huang (2011) proposed the application of the coordinate descent algorithm to the SCAD and MCP regularisation problems. The numerical results reported by Breheny and Huang (2011) suggest that MCP should be favoured over the lasso and SCAD penalties. At the same time, the coordinate descent algorithm provides an efficient, fast and stable alternative to solving the non-convex problem formulations. Breheny and Huang (2011) also noted that the hyperparameters are generally determined by cross-validation or by utilising information criteria such as the AIC or BIC. The AIC and BIC, however, tend to select local minima when used in the non-convex region of the objective function, while cross-validation can be associated with an excessive computational burden when attempting to find the optimal hyperparameters. Therefore, Breheny and Huang (2011) proposed the use of a combination of information criteria (BIC), cross-validation and a convexity diagnostics to mitigate the limitations associated with information criteria and cross-validation when used in isolation. The proposed framework seems to outperform the local linear approximation (LLA) approach when considering computational efficiency and produces a similar solution with a 1000-fold reduction in computation time. The coordinate descent algorithm, however, experiences a slight decline in computational efficiency in the presence of highly correlated variables but still outperforms the LLA approach by generating a solution 100 times faster. Even though the coordinate descent algorithm, in combination with the proposed hybrid hyperparameter selection framework, seems to perform well, some level of subjectiveness is still inherent to the approach (Venter, 2020). The successful implementation of the proposed framework still depends on the evaluation thereof by the user, which may be vulnerable to the choices made by the user. Nevertheless, neither the coordinate descent nor the LLA solution algorithms can guarantee convergence to the optimal solution in the case of logistic regression. Another rather unwanted characteristic is that SCAD and MCP formulations tend to overfit on training observations, especially for logistic regression problems (Venter, 2020). This is especially prominent when utilising a MCP formulation where $\varsigma \rightarrow 1$ (Venter, 2020). In general, even though SCAD and MCP regularisation approaches produce slightly less biased esti-

mates, some limitations still adversely affect the final subset of variables selected for model inference.

3.3.4 Regularisation-based MIP formulations

In an attempt to provide an optimality guarantee, Dedieu et al. (2021) proposed an exact MIP regularised variable subset selection formulation that uses both the lasso and ridge regression penalties. An additional L_0-norm penalty term was also incorporated which attempts to enforce model sparsity by regulating the number of independent variables included in the variable subset. This is achieved by utilising binary decision variables $z \in \{0,1\}^p$ where z is a vector of size p. This MIP approach, therefore, utilises an $L_0 - L_q$-penalty as part of its formulation. The regression parameter estimate $|\hat{\beta}_j|$ is set equal to zero when $z_j = 0$ and may take on any value between $[0, M]$ when $z_j = 1$. The constant M refers to the upper bound of the regression parameter estimate and is enforced by a Big-M constraint. The value of M should be specified to be sufficiently large to ensure the complete MIP search space is considered as part of the branch-and-bound solution process. The objective function proposed by Dedieu et al. (2021) aims to

$$\min_{\beta} \quad \frac{1}{n} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_0 ||\beta||_0 + \lambda_1 ||\beta||_1 + \lambda_2 ||\beta||_2^2. \tag{3.18}$$

The equation in (3.18) can be further simplified by grouping the logistic loss function with the L_1- and L_2-norms, such that

$$q(\beta; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) = \frac{1}{n} h(\beta; \mathbf{X}, \mathbf{Y}) + \lambda_1 ||\beta||_1 + \lambda_2 ||\beta||_2^2. \tag{3.19}$$

Substituting (3.19) into (3.18) leads to

$$\min_{\beta, z} \quad q(\beta; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) + \lambda_0 \sum_{j \in p} z_j, \tag{3.20}$$

subject to

$$|\beta_j| \leq M z_j \quad \forall j \in p, \tag{3.21}$$

$$z_j \in \{0,1\} \quad \forall j \in p. \tag{3.22}$$

The formulation proposed in (3.20) - (3.22) is a variation of the general regularisation approach defined in (3.1). The L_0-norm which is defined by $\lambda_0 ||\beta||_0$ describes the number of non-zero regression parameters, i.e. the independent variables associated with $\hat{\beta}$ that are not equal to zero. This is enforced by the binary activation variable z_j as defined in (3.20). The L_0-norm can, therefore, be substituted with $\lambda_0 \sum_{j \in \mathbf{p}} z_j$ to produce an identical result.

A tighter formulation of the optimisation problem in (3.20) - (3.22) was proposed by Dedieu et al. (2021) and is summarised in (3.23) to (3.28). The tightened formulation is used to improve on or provide a bound of optimality to solutions generated by a two-step iterative local search algorithm. The first algorithm used is a coordinate descent approach followed by a local combinatorial search algorithm that seeks to improve the solutions generated as part of the first algorithmic iteration. We refer the reader to Dedieu et al. (2021) for

a detailed explanation of this iterative algorithmic approach. Finally, the tightened MIP formulation can be used to try and improve on the local solutions discovered while also attempting to provide an optimality guarantee. Let $A \subset p$ denote the indices of a subset of variables from all of the available variables p, with the compliment of A denoted by A^c. Then the support set can be defined as $A = \text{Supp}(\beta)$ where $\beta \neq 0$. Let $A^1 \subset A$ and $A^2 \subset A^c$ denote two sets with size of at most m. The function of m will become apparent when considering its influence on z_j, which will be discussed shortly. The superscript t associated with $\hat{\beta}^t$ represents the regression parameters generated at iteration t of the local search solution algorithm (a combination of coordinate descent with local combinatorial search). As part of the combinatorial search step, Dedieu et al. (2021) defines a $p \times p$ matrix B^A for which it's j-th row is equal to $\sum_{j \in p} e_j$ if $j \in A$ and zero otherwise. Let e_j represent the conical basis for $\mathbb{R}^p$. This means that for any $\hat{\beta} \in \mathbb{R}^p$, $(B^A\hat{\beta})_j = \hat{\beta}_j$ when $j \in A$ and $(B^A\hat{\beta})_j = 0$ when $j \notin A$. This swap-and-solve logic is captured by (3.25) - (3.27). The binary decision variable z_j facilitates this swap-and-solve logic. Particularly, for $j \in A$, when $z_j = 0$ it means that $j \in A^1$ and that variable X_j should be removed from the support set A. Alternatively, when $j \in A^c$ with $z_j = 1$, we have that for $j \in A^2$, variable X_j should be added to the support set A. The size of m controls the number of variables considered for inclusion into the support set A. A large value for m generally relates to lengthier execution times. The main objective of the MIP formulation is to

$$\min_{\beta, z, \omega} \; q(\omega; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) + \lambda_0 \sum_{j \in p} z_j, \tag{3.23}$$

subject to

$$|\beta_j| \leq M z_j, \;\; \forall j \in A^c, \tag{3.24}$$

$$\omega = \beta_j^t - \sum_{j \in A} e_j \beta_j^t (1 - z_j) + \sum_{j \in A^c} e_j \beta, \tag{3.25}$$

$$\sum_{j \in A} z_j \geq |A| - m, \tag{3.26}$$

$$\sum_{j \in A^c} z_j \leq m, \tag{3.27}$$

$$z_j \in \{0, 1\} \;\; \forall j \in p. \tag{3.28}$$

Dedieu et al. (2021) go on to develop another MIP formulation which is solved via an integrability generation algorithm (IGA). Let A represent the support and A^c the complement of A. Once again, the heuristic-based solution algorithm (a combination of coordinate descent with local combinatorial search) is used to prune the solution search space. The support A is then initialised with the variables identified by the initial heuristic solution framework. The proposed MIP formulation can be defined as

$$\min_{\beta, z} \; q(\hat{\beta}; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) + \lambda_0 \sum_{j \in p} z_j, \tag{3.29}$$

subject to

$$|\hat{\beta}_j| \leq M z_j, \;\; \forall j \in p, \tag{3.30}$$

$$z_j \in [0, 1], \;\; \forall j \in A^c, \tag{3.31}$$

$$z_j \in \{0, 1\}, \ \ \forall j \in A, \tag{3.32}$$

where z_j, $j \in A^c$, is allowed to be continuous and take on any value between zero and one, while integrality constraints are placed on z_j, $j \in A$, according to (3.32). Allow $\hat{\beta}^t$ and z^t to represent the regression parameters and binary selection variables for each algorithmic iteration t. At iteration $t+1$ update $A = A \cup \{j | z_j^t \neq 0, j \in A^c\}$ and then solve the MIP presented in (3.29) - (3.32) with warm-starting functionality of the MIP solver activated. This means that the solution in t is used as starting point for $t+1$. Since the MIP is a relaxation of (3.20) - (3.22), it provides a lower bound to the original problem. As such, at each iteration t, the MIP in (3.29) - (3.32) aims to improve the MIP lower bound while also allowing an optimality gap to be calculated. The solution algorithm proposed by Dedieu et al. (2021) is capable of solving high-dimensional problems with a variable set of up to 50 000 independent variables as a result of the novel integrability generation logic that is applied. In general, the empirical results presented in Dedieu et al. (2021) were focused on solving sparse classification problems where $n \leq 1000$.

Even though MIP formulations can provide an optimality guarantee, the formulations presented in Section 3.3.4 of Chapter 3 fall short in some respects. Since regularised MIP formulations still require derivation of the penalty parameters λ_0, λ_1 and λ_2, the proposed formulations cannot provide an optimality gap with regards to the complete search space over the full range of λ_0, λ_1 and λ_2. Thus, the optimality gap is related to a fixed value for each penalty parameter and does not give a true measure of optimality over the entire search space. In general, as discussed in previous chapters, the penalty parameters are determined through cross-validation and grid search, which is an iterative heuristic by nature. To clearly understand the point of discussion, it would be valuable to firstly define the optimality gap and clearly explain what it attempts to measure. The optimality gap (which is used to provide an optimality guarantee) can be expressed as

$$\text{optimality gap} = \frac{bb - bi}{|bi| + \alpha},$$

where bb denotes the best bound obtained by the MIP and bi the best integer solution. An arbitrary small value α is added to the denominator to prevent the gap from being undefined when $|bi|$ is zero. The best bound in a maximisation problem can be interpreted as the bound obtained by a LP relaxation of the original MIP problem. This means that all integer or binary variables are relaxed and allowed to take on any real value, which reduces the number of constraints that the problem needs to adhere to. The best integer solution is obtained by means of a branch-and-bound algorithm. The branch-and-bound algorithm is designed to solve combinatorial optimisation problems. The author refers the interested reader to Section 4.2.2 of Venter (2020) where a clear exposition of the branch-and-bound algorithm is provided. In essence, the optimality gap (as a performance measure) is applicable to all optimisation problems, whether a heuristic-based or deterministic-based solution approach is considered. The difference between the two approaches relates to the fact that heuristic-based approaches can only provide an upper or lower bound (depending on whether we are dealing with a minimisation or a maximisation problem), whereas a deterministic algorithm is able to enforce both upper and lower bounds for the problem space, thereby making it possible to calculate an

optimality gap. Returning to the initial discussion, it should be clear to the reader that these MIP-based lasso and ridge regression approaches can only provide an optimality gap for a fixed value of the penalty parameters as the full range of the penalty parameters cannot simultaneously be considered in combination with the proposed MIP. This will result in a non-linear objective function that would greatly complicate the MIP solution approach. In general, the regularisation parameters are determined first before the MIP is solved. This two-step approach is followed by Dedieu et al. (2021) where the first step utilises a heuristic-based approach towards the specification of the regularisation penalty path associated with each parameter. The dimensions of the grid that should be considered depend on the value of $q \in \{1, 2\}$ assigned to the $L_0 - L_q$-penalty term in the MIP formulation. A search over a two-dimensional space will be required when a $L_0 - L_1$-penalty or $L_0 - L_2$-penalty is considered. This substantiates the claim that the gap percentage generated as part of these MIP formulations does not adequately describe the complete solution search space.

3.4 Best subset selection

The logic behind a best subset selection approach that uses some form of statistical or probabilistic selection criteria is described in Algorithm 3.4. In essence, best subset selection performs an exhaustive search in pursuit of the best model by considering all possible combinations of independent variables. Let Ξ denote the model evaluation criteria which will be used to assess each model at iteration t amongst a total number of 2^p possible candidate models. Allow Ξ_b to denote the best criterion value obtained throughout the search procedure and $T = 2^p$ the total number of algorithmic iterations considered as part of the for-loop in Algorithm 3.4. Various selection criteria Ξ exist for use as part of best subset selection within the context of logistic regression. The most popular selection criteria include AIC, BIC, classification accuracy, the area under the receiver operator curve (AUC), sensitivity and specificity of the classifier and the Gini coefficient (Wah et al., 2018). Each of these selection criteria will be discussed in detail in this subsection. The AIC will be excluded from the discussion as it was defined in Section 3.2 of Chapter 3. Let us first consider the BIC. The BIC is similar to that of the AIC, with only a slight difference in how the number of variables present in the model is penalised.

Algorithm 3.4 Best subset selection algorithm

1: Select model evaluation criteria Ξ.
2: Set Ξ_b equal to 0.
3: **for** ($t \leftarrow 1$ to T) **do**
4: Calculate Ξ_t for each model at iteration t.
5: **if** (Ξ_t improves on Ξ_b) **then**
6: $\Xi_b = \Xi_t$
7: best model = model at iteration t.
8: **end if**
9: **end for**
10: End algorithm with the best model recorded out of 2^p candidates.

The BIC is derived within a Bayesian framework as an estimate to the Bayes factor between two competing models (Acquah, 2010). The BIC can be written as

$$BIC = 2h(\beta; \mathbf{X}, \mathbf{Y}) + \log(n)(\sum_{j \in p} z_j),$$

where $h(\beta; \mathbf{X}, \mathbf{Y})$ denotes the logistic loss function, n is the number of observations and $(\sum_{j \in p} z_j)$ the number of regression parameters that are included in the model. The BIC does not have a fixed constant associated with its penalty term but instead utilises the natural logarithm of the number of observations. This means that an increase in the number of observations will directly affect the magnitude of the penalty applied. The BIC penalises model complexity more harshly than the AIC and is, therefore, less likely to select a model with increased complexity. If a true model exists between a selection of candidate models, the BIC is more likely to select the true model with an increase in the number of observations n. The BIC's underlying definition is slightly different to that of the AIC because the AIC aims to find the best approximate model from the unknown true data. At the same time, the BIC is designed to find the most probable model from the set of possible candidate models (Acquah, 2010). Although the underlying theory on which the two measures are based differ, both methods are practically used in an attempt to identify good models from a set of candidate solutions (Acquah, 2010). In general, a reduction in BIC or AIC relates to better models. Following the logic from Algorithm 3.4, a set of 2^p models will be developed, after which the model with the smallest AIC or BIC will be considered the best model from the set of possible candidate models.

The AUC provides a measure that describes the degree of separability in a classification model (Bradley, 1997). The larger the AUC value of a model, the better the class separation between the target classes. Since the AUC refers to the area under the ROC curve, some underlying theory associated with the ROC curve needs to be discussed. We will first define sensitivity and specificity, also referred to as recall or true positive rate and true negative rate, respectively (Bradley, 1997). Sensitivity can be defined as

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \tag{3.33}$$

where TP denotes the true positives and FN the number of false negatives. In general, TP refers to instances for which the classifier correctly classified the response variable

as an event case. In contrast, FN refers to instances where the classifier classified the response variable as a non-event case, but in actual fact, it should have been equal to an event case. In essence, sensitivity provides a measure that indicates which proportion of the positive class has been correctly classified. Specificity can be written as

$$\text{Specificity} = \frac{TN}{TN + FP}, \tag{3.34}$$

where TN denotes the true negatives and FP the false positives. Specificity provides a view of the proportion of the negative class that has been correctly classified. True negatives refer to instances where the classifier correctly classified the negative class (non-event case), while FP describes the number of instances where the classifier incorrectly classified the response variable as a positive class. Another key concept to understand is the prediction probability threshold. Since logistic regression provides a prediction probability within the range of zero and one, some threshold should be selected to indicate a separation between classes. If a threshold of 0.5 is selected, it would mean that any instance with a prediction probability less than 0.5 will be classified as a non-event, while those with predicted probabilities larger than 0.5 will be classified as an event. Considering the above, it should be apparent that changing the prediction threshold will also result in a change in both sensitivity and specificity (Bradley, 1997). Instead of selecting a single predicted threshold, the ROC curve is used to generate a visual representation of how the true positive rate (TPR) and false positive rate (FPR) change with a change in the probability threshold. The ROC curve is a plot of the TPR against the FPR while the AUC measure is a single metric representing the area under the ROC curve (Bradley, 1997). An AUC value greater than 0.5 indicates a higher likelihood that the classifier can correctly distinguish between the positive and negative classes. This implies that the model is more proficient at producing true negatives and true positives. Let AUC_{area} denote the total area calculated by using the trapezoidal rule and K the total number of trapezoids or intervals used to approximate the area under the ROC curve, which is defined by $f^{ROC}(\gamma)$. Allow γ_k to describe the grid points on the x-axis (the false positive rate at point k) and $f^{ROC}(\gamma_k)$ the corresponding true positive rate at point k. The ROC curve can be defined as a plot of the TPR (3.33) against the FPR (3.34). The size of each interval k between two x-axis grid points (given by $\gamma_k - \gamma_{k-1}$) is denoted by $\Delta\gamma_k$. In some cases, equally-spaced intervals can be used, which will simplify $\Delta\gamma_k$ such that it becomes a single value. The AUC can then be calculated by employing the trapezoidal rule which yields

$$AUC_{area} = \int_a^b f^{ROC}(\gamma)d\gamma = \sum_{k \in K} \frac{f^{ROC}(\gamma_{k-1}) + f^{ROC}(\gamma_k)}{2} \Delta\gamma_k, \tag{3.35}$$

where a and b define the start and end points of the integral taken over the ROC function f^{ROC} while $d\gamma$ denotes the variable of integration, which in this case is γ. The AUC measure can be used as the selection criteria Ξ in Algorithm 3.4 such that the model with the highest AUC will be selected as the final model from the 2^p candidate models. The advantage of using the AUC instead of accuracy is its ability to consider class imbalance. Ultimately, the AUC provides a good overview of the model's discriminative power. An alternative to the AUC is the Gini coefficient, which provides the same information as the

AUC. The Gini coefficient (denoted by G) is related to the AUC such that

$$AUC = \frac{G+1}{2}. \tag{3.36}$$

The Gini coefficient can be calculated as the ratio of the area between the diagonal line and the Lorenz curve (area **A** in Figure 3.2) to the total area below the diagonal line (areas **A** and **B** in Figure 3.2) (Fellman et al., 2012) [3].

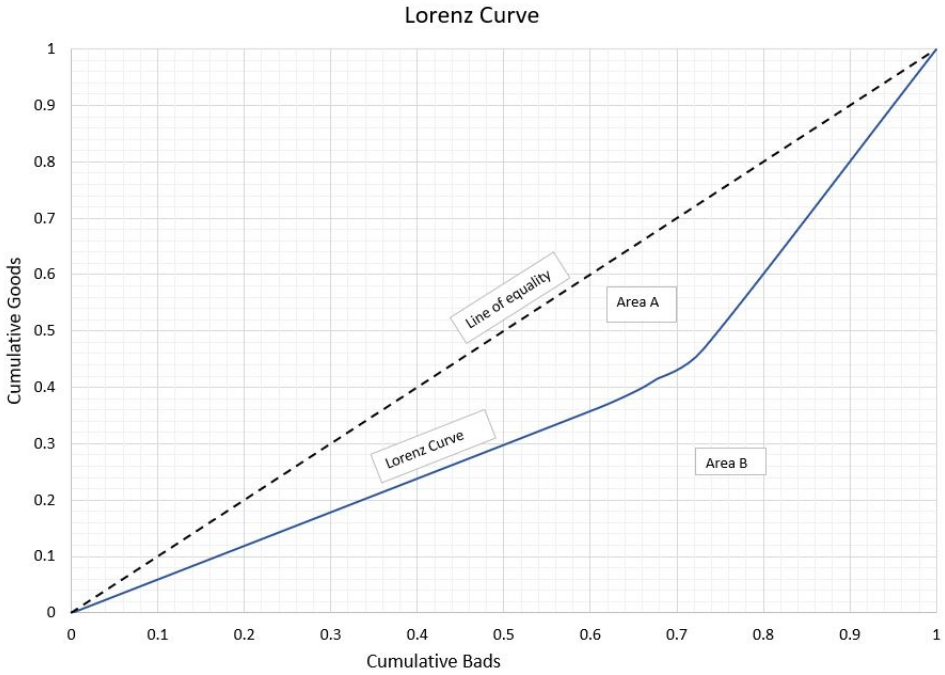


Figure 3.2: Lorenze curve

Mathematically, the Gini coefficient that is derived from the Lorenz curve can be defined as

$$G = \frac{\text{Area } A}{\text{Area } A + \text{Area } B}, \tag{3.37}$$

where Area A+Area B can be set equal to 0.5 (this is because the area below the diagonal line will always be equal to 0.5). This results in the Gini coefficient formula simplifying to 2Area A. Another alternative to the Gini coefficient calculation is given by

$$G = 1 - 2\text{Area } B. \tag{3.38}$$

Area B can be calculated by using

$$\text{Area } B = \int_0^1 f^L(\gamma)d\gamma, \tag{3.39}$$

where $f^L(\cdot)$ denotes the Lorenz curve (Chotikapanich and Griffiths, 2001). The Lorenz curve describes the differences between the cumulative event-case distribution (the "bads")

[3]The "bads" in Figure 3.2 refer to event cases where $Y = 1$ and the "goods" refer to the non-event cases where $Y = 0$. This type of terminology is often used within the context of the Lorenz curve because "bads" are used in the financial industry to describe defaulters in binary probability-to-default models, while "goods" are used to describe up-to-date customers.

and the cumulative non-event case distribution (the "goods")[4]. The area under the Lorenz curve can be estimated by using the trapezoidal rule presented in (3.35), after which the Gini coefficient can be calculated using (3.36). Similar to the AUC, an increase in the Gini coefficient represents an improvement in the predictive power of the selected model. Therefore, the model with the largest Gini coefficient will be considered the champion from the 2^p possible candidates.

Another simplistic model selection criterion that can be used as part of Algorithm 3.4 is classification accuracy. Accuracy can be defined as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3.40}$$

and relates to the ratio of the number of instances the classifier correctly identified in relation to the total number of instances present in the dataset. It is, however, important to note that accuracy does not function well with imbalanced datasets where one of the binary classes makes up the majority of the target variable. If the majority of response variable instances consist of e.g. non-events and the classifier classifies all instances as non-events, its classification accuracy will be high. This may lead to the user considering the model as a good classifier, when in reality, it performs rather poorly because the separability between the binary classes is inadequate. This is one of the main reasons why the AUC or Gini coefficient is preferred over accuracy as a measure of model performance. Therefore, using accuracy as a selection criterion should be reserved for cases where the dataset is balanced between the two binary classes.

3.4.1 Best subset selection based MIP formulations

In general, a best subset selection approach that utilises some form of model selection criteria (as discussed previously) can be solved iteratively by either the re-weighted least squares method, gradient descent method or Newton method (Maalouf, 2011; Gasso, 2019). These methods are local search algorithms and cannot provide a measure of optimality. Therefore, some researchers have also approached the best subset selection problem from an exact deterministic perspective.

Sato et al. (2016b) presents a MIP formulation that attempts to simultaneously optimise the logistic loss function while reducing model complexity. The objective function includes a model complexity term focused on minimising information loss. This is achieved by attempting to minimise the AIC or BIC information criteria. Let I denote an adjustment constant which facilitates switching between either the AIC or BIC. It is important to note that I does not represent a function. When $I = 2$, the model attempts to minimise the AIC, while $I = \log(n)$ results in the BIC being minimised. Let n denote the total number of observations in the training set. Sato et al. (2016b) employed a binary selection decision variable z_j which controls the value of the regression parameter β_j associated with the j-th independent variable X_j. Let $k \in K = \{1, 2, ..., K\}$ describe a set of symmetric grid point

[4]As part of Chapter 3, the definition of γ is made clear based on the section considered. However, as part of Chapter 3, γ represents the grid point values on the x-axis.

values such that γ_k denotes the line segment associated with point k. The piecewise linear approach proposed by Sato et al. (2016b) approximates the non-linear logistic loss function and can formally be defined according to (3.41). This relates to a series of linear functions that approximate the logistic loss function from below while excluding non-feasible regions from the solution search space. This relates to the pointwise maximisation of a set of linear inequalities such that

$$\max\{h'(\gamma_k)(Y_i(X_i^T\beta) + h(\gamma_k)\} \ \ \forall k = 1, ..., K. \tag{3.41}$$

These linear inequalities can be transformed for use in a minimisation problem by using

$$\min\{\varphi|\varphi \geq h'(\gamma_k)(Y_i(X_i^T\beta) + h(\gamma_k)\} \ \ \forall k = 1, ..., K. \tag{3.42}$$

Figure 3.3 is a graphical representation of the approximation approach defined by (3.42) with the dashed lines representing the linear functions used to approximate the non-linear logistic loss function from below. The number of linear functions K used to approximate the logistic loss function will influence both the accuracy of the approximation and the computational requirement of the MIP formulation. Therefore, a trade-off exists between model accuracy and computational demand. Figure 3.3 provides an example of $K = 2$ straight lines at points γ_0 and γ_K (shown as dashed lines) that are used to approximate the non-linear logistic loss function (shown as a solid black line).

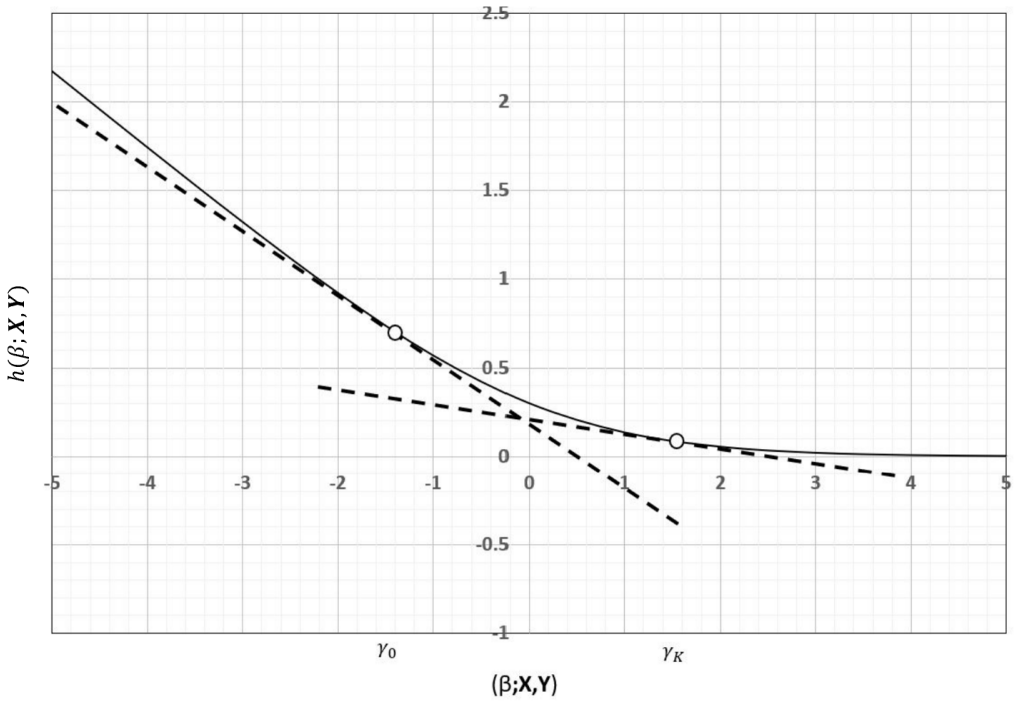


Figure 3.3: Piecewise Linear overestimation of the logistic loss function

The piecewise linear objective function proposed by Sato et al. (2016b) is given by

$$\min_{\beta,z,\varphi} \ \ 2\sum_{i\in n}\varphi_i + I(\sum_{j\in p} z_j). \tag{3.43}$$

subject to

$$\varphi_i \geq h'(\gamma_k)(Y_i(X_i^T\beta) - \gamma_k) + h(\gamma_k) \;\; \forall k \in K; \forall i \in n, \tag{3.44}$$

$$z_j = 0 \rightarrow \beta_j = 0 \;\; \forall j \in p, \tag{3.45}$$

$$z_j \in \{0, 1\} \;\; \forall j \in p, \tag{3.46}$$

where the constraint in (3.44) represents the piecewise linear overestimation of the log-likelihood function (as described by Sato et al. (2016b) and Venter (2020)). Recall from Chapter 2 that the negative of the logistic loss is equal to the log-likelihood function, which suggests that (3.44) would be an overestimation of the log-likelihood function. The binary decision variable z_j is used as part of (3.45) which is a special form of a Big-M constraint referred to as a SOS1 constraint. A SOS1 implies that only one element in the set $\{\hat{\beta}_j, 1 - z_j\} \forall j \in p$ can be non-zero such that $\hat{\beta}_j$ should be set equal to zero when $z_j = 0$ and non-zero otherwise. The objective function in (3.43) that utilises the AIC (with $I = 2$) attempts to

$$\min_{\beta,z,\varphi} \;\; 2\sum_{i \in n} \varphi_i + 2(\sum_{j \in p} z_j), \tag{3.47}$$

while the BIC formulation (with $I = \log(n)$) attempts to

$$\min_{\beta,z,\varphi} \;\; 2\sum_{i \in n} \varphi_i + \log(n)(\sum_{j \in p} z_j). \tag{3.48}$$

An alternative piecewise linear approximation approach is that of Venter (2020) which proposes a piecewise linear underestimation of the non-linear log-likelihood function. This approach will be formalised in Chapter 5 and forms the basis from which the ALRSF is derived.

3.4.2 Best subset selection limitations

In general, one of the main drawbacks assocaited with best subset selection is the approach's computational burden. Miller (2002) stated that the best subset selection problem is NP-hard and can be considered as an exponential-time procedure. This means that adding additional variables to the set of p independent variables exponentially increases the execution time associated with the solution algorithm. As mentioned previously, the number of possible candidate models that can be considered is 2^p (Mao, 2004). A consensus exists among researchers that best subset selection problems with more than 75 independent variables become intractable (Lund, 2017). As such, the feasibility of applying best subset selection to large variable sets diminishes with an increase in the number of independent variables, which makes scalability problematic.

Generally, the value of the logistic regression log-likelihood function will increase with the addition of more variables. This will affect selection criteria that makes use of the log-likelihood value as part of its formulation, such as AIC and BIC. However, an increase in the log-likelihood function does not nexessarily translate to a substantial increase in predictive performance. This monotonically increasing trend (associated with the log-likelihood function) can be attributed to the model finding meaningless correlations between the

independent variables and the target variable, which can lead to overfitting and a reduction in generalisation performance. In general, most of the selection criteria proposed in Section 3.4 of Chapter 3 will yield better results with the addition of more variables (Venter, 2020). Venter (2020) noted that performance measures such as the AUC or Gini coefficient will reach a plateau, after which no significant improvement will be observed by adding more predictors. Expanding the subset of variables, which will be used for model inference, without a notable performance improvement seems unjustified and can lead to a loss of generalisation performance. Therefore, using Algorithm 3.4 in conjunction with the AUC or Gini coefficient as the selection criteria may result in an appropriately-sized final subset of variables. This might only be valid when the order in which the independent variables are added is based on some rank order that is derived from predictive performance (i.e. adding highly predictive variables first and proceeding in this fashion until the variable with the lowest predictive performance is added last). Similarly, empirical evidence shows that even model selection based on information criteria (such as the AIC or BIC), which include a penalty term in an attempt to optimise for model complexity, tend to favour larger models that are less parsimonious (Venter, 2020; Kutner et al., 2005). As such, even though the best subset selection techniques presented in Section 3.4 of Chapter 3 have some favourable characteristics, it is evident that these approaches still fall short of the main objective of producing parsimonious models with good generalisation performance.

3.5 Summary

A clear overview of variable subset selection algorithms that are commonly found in literature was provided, along with discussions on the limitations associated with each. The identified limitations highlight the literature gap (within the context of variable subset selection where logistic regression is used as the underlying modelling framework) that the subsequent model development Chapters 5 to 7 will attempt to address or at least aim to improve on. The main limitations that will be considered as part of the novel solution framework's development process include:

1. Reducing the computational burden associated with solving high-dimensional variable best subset selection problems.

2. Providing a measure of optimality that can be used to verify the quality of the solution with respect to the "true" optimal solution (noting that many of the best subset selection algorithms discussed thus far cannot provide such a guarantee).

3. Providing a way by which overfitting on training observations can be assessed while limiting loss of generalisation performance on unseen data.

4. Preventing the inclusion of redundant variables into the final subset of variables that add no significant predictive power to the classification model.

Chapter 4

Overview of relevant optimisation theory

Generally, NP-hard MIP problems are solved using the well-known branch-and-bound algorithm (Clausen, 1999). The branch-and-bound algorithm prunes the solution search space to discover the best possible integer solution. Explicit enumeration of the total search space is impractical. As such, the branch-and-bound algorithm utilises the bounds of the problem being considered in combination with the current best solution discovered to implicitly search only parts of the solution search space (Clausen, 1999). This gives rise to an attractive property associated with these exact deterministic solution algorithms: they can provide a measure of optimality.

Depending on whether the MIP being considered is a maximisation or a minimisation problem, the branch-and-bound algorithm can either estimate a lower bound or upper bound for the MIP. The alternative bound (either lower or upper bound) of the MIP problem can then be estimated using a linear relaxation of the original MIP formulation. Linear relaxation, however, falls short of producing a tight bound when the MIP is loosely defined. This can result in large optimality gaps that require significant computational power to prove optimality. It is also well-known that MIP problems generally increase in difficulty as the number of constraints and variables increase (Taskın, 2010). This observation also applies to the logistic regression best subset selection problem, as highlighted in Chapter 3. The number of candidate solutions 2^p, grows as the number of independent variables p increases. Several decomposition and relaxation techniques have been developed throughout recent years in an attempt to solve increasingly difficult NP-hard problems by exploiting their underlying problem structure. Two of these methods, Benders decomposition and Lagrangian relaxation, will be discussed in this chapter and will play an integral part in developing the novel solution framework that is presented in Chapter 5.

4.1 Benders decomposition

The Benders decomposition algorithm was first proposed by Benders (1962) to solve mixed-variable optimisation problems. The idea behind Benders decomposition is to fix a set of complicating variables such that the problem becomes significantly easier to solve (Rahmaniani et al., 2017). This relates to the problem being partitioned into a master problem and several sub-problems, followed by delayed constraint generation known as cuts (Costa, 2005). According to Rahmaniani et al. (2017), Benders decomposition can alternatively be referred to as outer linearisation or variable partitioning. The approach has been widely applied to numerous problem formulations, including non-linear, linear, stochastic, integer, bilevel and multi-stage optimisation problems (Rahmaniani et al., 2017). The Benders decomposition approach will be explained at the hand of the general mixed integer linear programming (MILP) problem presented in Rahmaniani et al. (2017). The MILP aims to

$$\min_{x,y} \rho^T y + \sigma^T x, \tag{4.1}$$

subject to

$$Wy = \kappa, \tag{4.2}$$

$$Oy + Dx = \tau, \tag{4.3}$$

$$x \geq 0, \ \ \forall x \in \mathbb{R}^{n_2}, \tag{4.4}$$

$$y \geq 0, \ \ \forall y \in \mathbb{Z}^{n_1}, \tag{4.5}$$

where $W \in \mathbb{R}^{m_1 \times n_1}$ represents a known matrix of size $m_1 \times n_1$ and $\kappa \in \mathbb{R}^{n_1}$ a predetermined vector of size n_1. The variables $x \in \mathbb{R}$ and $y \in \mathbb{Z}$ are continuous and integer decision variables, respectively. Let both $O \in \mathbb{R}^{m_2 \times n_1}$ and $D \in \mathbb{R}^{m_2 \times n_2}$ denote a known matrix of size $m_2 \times n_1$ and $m_2 \times n_2$, respectively. Similar to κ, let $\tau \in \mathbb{R}^{n_2}$ denote a given vector of size n_2. Allow ρ and σ to describe the cost associated with each decision variable y and x, respectively, with (4.1) aiming to minimising the overall cost. The integer decision variables y can be considered as complicating variables. Let the constraint in (4.3) act as a connecting constraint which relates the continuous decision variables x to the complicating integer decision variables y, while (4.4) and (4.5) ensure that both x and y are non-negative decision variables. For any fixed value of y (denoted by y^*), the original optimisation problem (4.1) - (4.5) can be reformulated such that it becomes

$$\min\{\rho^T y^* + \min_{x \geq 0}\{\sigma^T x \ \mid \ Dx = \tau - Oy^*\}\}. \tag{4.6}$$

Since the inner minimisation problem is a continuous linear problem it can be dualised by adding dual variables $\theta_v, v \in V$, to the relaxed constraint set $Dx = (\tau - Oy^*)$, where $v \in V$ denotes a set of extreme points generated by solving the dualised form of the inner minimisation problem. The dualised form can be defined as

$$\max_{\theta_v, v \in V} \theta_v^T (\tau - Oy^*), \tag{4.7}$$

subject to

$$\theta_v^T D \leq \sigma \ \mid \ \forall v \in V. \tag{4.8}$$

According to duality theory, (4.7) can be used interchangeably with the inner minimisation problem in (4.6). This leads to (4.6) being refined and expressed as a min-max optimisation problem equivalent to

$$\min\{\rho^T y^* + \max_{\theta_v, v \in V}\{\theta_v^T(\tau - Oy^*)\}\}, \tag{4.9}$$

subject to

$$\theta_v^T D \leq \sigma \ \mid \ \forall v \in V. \tag{4.10}$$

Even though the feasible solution space of the inner maximisation problem $\{\theta \mid \theta_v^T D \leq \sigma\}$ is independent of the choice of y^*, it still affects the optimisation problem in two distinct ways. It can yield a solution for (4.9) that is either feasible or unbounded (Rahmaniani et al., 2017). Considering the latter, it is necessary to note that there exists some direction of unboundedness $r_g | g \in G$ for which $r_g^T(\tau - Oy^*) > 0$. Unboundedness should be avoided since it relates to the infeasibility of the generated solution (based on the fixed values for y^*). To prevent unboundedness, a feasibility cut of the form

$$r_g^T(\tau - Oy^*) \leq 0 \ \mid \ \forall g \in G, \tag{4.11}$$

is added to (4.9), which leads to

$$\min\{\rho^T y^* + \max_{\theta_v, v \in V}\{\theta_v^T(\tau - Oy^*)\}\}, \tag{4.12}$$

subject to

$$r_g^T(\tau - Oy^*) \leq 0 \ \mid \forall \ g \in G. \tag{4.13}$$

The optimisation problem defined by (4.12) can be linearised by substituting the dual maximisation problem (present in the min-max objective function) with a continuous decision variable ξ. This linearised form is known as the Benders decomposition master problem (MP) and can be defined as

$$\min_{y, \xi} \rho^T y + \xi, \tag{4.14}$$

subject to

$$Wy = \kappa, \tag{4.15}$$

$$r_g^T(\tau - Oy) \leq 0 \ \mid \ g \in G, \tag{4.16}$$

$$\theta_v^T(\tau - Oy) \leq \xi \ \mid \ v \in V, \tag{4.17}$$

$$y \geq 0 \ \mid \ y \in \mathbb{Z}, \tag{4.18}$$

The Benders decomposition sub-problem (SP) is given by (4.7) - (4.8). The MP attempts to generate suitable values for both y and ξ, after which the SP in (4.7) is solved by using the value of y that is generated by solving the MP such that $y = y^*$. The constraints in (4.16) and (4.17) can be referred to as feasibility and optimality cuts, respectively. The cuts are generated based on the solution of the SP. If the solution of the SP is unbounded, a feasibility cut of the form (4.16) is added to the MP. Alternatively, if the solution is feasible and bounded, an optimality cut of the form (4.17) is added to the MP. This iterative solve-and-cut approach continues until the algorithm converges to a solution. In general, only a subset of variables is considered as part of the MP. In contrast, the values

for the remaining variables are determined as part of the SPs (Taskın, 2010). The MP will largely contain all of the integer variables in the vector y, while the continuous variables in x form part of the SP (Taskın, 2010). Ultimately, Benders decomposition aims to exploit the underlying block structure of the MIP formulation (Taskın, 2010). An added benefit of the Benders decomposition algorithm is the ability to calculate an optimality gap. Since the MP is a relaxed form of the equivalent Benders decomposition, it provides a lower bound estimate to the optimal cost (Rahmaniani et al., 2017). The upper bound estimate can be obtained by combining the solution of y^* with the objective function of the SP. This is equivalent to solving the original problem with y fixed to a constant value y^*. The upper and lower bound can then be used to calculate an optimality gap. The algorithm is terminated when optimality is achieved or when the optimality gap is lower than a pre-defined threshold (Adulyasak et al., 2015).

4.2 Lagrangian relaxation

Lagrangian relaxation is a universal bounding technique used in combinatorial optimisation (Lemaréchal, 2001). It was developed by Held and Karp (1970) to solve the well-known travelling salesman problem. Held and Karp (1970) utilised a Lagrangian approach that is based on minimum spanning trees to successfully solve the travelling salesman problem. After noting the success of Held and Karp (1970), Lagrangian-based methods were subsequently used to solve several scheduling problems throughout the 1970s. It was not until 1974 that Geoffrion (1974) coined the name "Lagrangian relaxation". Since then, its application has grown significantly and has been applied to some of the most notoriously difficult combinatorial optimisation problems (Held and Karp, 1970). The mechanism behind Lagrangian relaxation can be described with the help of an optimisation problem that aims to [1]

$$\max \quad f(x), \tag{4.19}$$

subject to

$$g_j(x) = 0, \tag{4.20}$$

$$h_j(x) \leq 0, \tag{4.21}$$

where $j \in J = \{1, 2, ..., |J|\}$ denote the number of constraints associated with (4.20) and (4.21) and $x \in \mathbb{R}$ denotes a real value that is used as input to the function $f(x)$, $g_j(x)$ and $h_j(x)$ [2]. Assume that (4.20) represents a set of complicating constraints and (4.21) a set of constraints that are relatively easy to satisfy. The presence of the constraint set in (4.20) makes the optimisation problem increasingly difficult to solve. However, removing (4.20) from the optimisation problem would alter the underlying structure of

[1] The notation f, g and h (used as part of Section 4.2 in Chapter 4) differ in meaning from what has previously been defined. In this context, it is used as a general function definition to allow for a generalised introduction to Lagrangian relaxation.

[2] The index j in the context of Section 4.2 of Chapter 4 is not the same as j used to denote the j-th independent variable from the set of p variables. Rather, it serves as an index for the constraints defined by (4.20) and (4.21).

the problem and will influence the outcome. It is preferable to leave the MIP problem's underlying structure unaltered while finding an efficient way to solve (4.19). This is where Lagrangian relaxation is useful. It aims to incorporate relaxed versions of the complicating constraints into the objective function such that the MIP becomes much easier to solve. The Lagrangian relaxation function can be formulated with reference to the primal variable x and some dual variable ϕ such that

$$L(x;\phi) \quad = \max_{x\in\mathbb{R}} \quad f(x) - \sum_{j\in J} \phi_j g_j(x), \tag{4.22}$$

subject to

$$h_j(x) \leq 0. \tag{4.23}$$

A relaxed version of the constraint in (4.20) is added to the objective function in (4.22) and multiplied by what is known as Lagrangian multipliers, denoted by ϕ_j, where ϕ_j is a vector of multipliers (Sun et al., 2007). The purpose of the Lagrangian multipliers is to penalise any violation of the original constraints in (4.20). Since the relaxed constraint in (4.20) is an equality constraint, the Lagrangian multipliers are not restricted to a specific sign convention. However, in the case of relaxing an inequality constraint, the Lagrangian multipliers are usually restricted to $\phi_j^t > 0$, where T denotes the number of algorithmic iterations. The Lagrangian multipliers attempt to enforce the original problem structure onto the relaxed formulation while simultaneously attempting to satisfy ($\phi^T g(x) = 0$). Lemaréchal (2001) described Lagrangian multipliers as a linear price associated with each constraint $g_j(x)$ based on the sign of ϕ_j. Whether ϕ should be increased or decreased depends on the magnitude and sign of the constraint violation. Another valuable characteristic of the Lagrangian relaxation approach is its bounding property which relates to solving the dual of (4.22). The dual of the relaxed formulation is presented as

$$f(\phi) \quad = \quad \min_{\phi\in J} \quad L(x;\phi). \tag{4.24}$$

Any value for ϕ will satisfy the bounding inequality such that

$$f(\phi) \geq f^*(\phi) \geq f(x) \;\; \forall x \in \mathbb{R}, \tag{4.25}$$

since $\phi^T g(x) = 0$ when optimality is reached. This is known as weak duality, as noted by Lemaréchal (2001) [3]. Let $f^*(\phi)$ denote the optimal solution to (4.24) (referred to as the Lagrangian dual). In some cases, the optimal solution to the Lagrangian dual will be equal to the optimal solution of the original MIP (however, this is not always the case). This is true when the optimal value of ϕ is discovered with the primary variable x being feasible in the original optimisation problem in (4.19) such that $f^*(\phi) = f(x)$. From (4.24) it is clear that any value $f(\phi)$ will bound the original objective function in (4.19) from above.

[3] In general, weak duality relates to the optimal solution to the dual being greater than or equal to the optimal solution of the primal problem (Udriste et al., 2015). In contrast, strong duality relates to the optimal solution to the dual problem being equal to the optimal solution of the primal problem (Udriste et al., 2015). In other words, the duality gap is equal to zero. Strong duality cannot always be proven for all optimisation problems and, in the case of weak duality, the dual gap may take on a value greater than or equal to zero. Weak duality still allows us to tighten the upper/lower bound (depending on the type of optimisation problem considered) in comparison to linear relaxation (used as part of the standard branch-and-bound approach).

This means that the sharpest possible upper bound to (4.19) can be found by identifying the values for $\phi_j \in J$ that will solve (4.24). Bounding by means of Lagrangian relaxation allows a measure of optimality to be calculated. This bounding property associated with the Lagrangian relaxation approach is beneficial in practice and can be used to find tighter bounds to the original MIP problem. In general, Lagrangian relaxation can be used instead of linear relaxation to bound the optimisation problem from above or below, depending on the nature of the optimisation problem (Fisher, 1981).

To the best of the author's knowledge, no Lagrangian-relaxation-based approach has been used as part of any solution algorithm in an attempt to solve the logistic regression best subset selection problem. Gaudioso et al. (2017) has, however, applied the concept of Lagrangian relaxation to a L_1-regularised support vector machine (SVM) variable subset selection MIP formulation while utilising a generalised bundle method (GBM) as solution algorithm. Gaudioso et al. (2017) noted that the proposed relaxation algorithm was able to produce high-quality solutions and was associated with an affordable computational cost. As part of this thesis, an alternative, easy-to-implement Lagrangian relaxation solution algorithm known as the sub-gradient method will be considered. The solution algorithm aims to determine the Lagrangian multipliers ϕ_j^t at each algorithmic iteration t while guiding the solution towards convergence. The underlying theory associated with the sub-gradient algorithm will be explained in Section 4.2.1 of Chapter 4.

4.2.1 Towards solving the Lagrangian dual

Recall that the dual problem can be defined as

$$f^*(\phi) = \min_{\phi \in J} L(x; \phi), \tag{4.26}$$

which relates to finding the values for $\phi_j \in J$ that pertain to the optimal solution of the dual such that $f(\phi) = f^*(\phi)$. Note that the Lagrangian function can be defined according to the upper envelope of the set of hyperplanes $\{f(x) - \phi^T g(x) \mid x \in \mathbb{R}\}$ which allows reformulation of the optimisation problem such that the objective function becomes

$$\min \omega, \tag{4.27}$$

subject to

$$\omega \geq f(x) - \sum_{j \in J} \phi_j^t g_j(x) \mid x \in \mathbb{R}. \tag{4.28}$$

Solving problem (4.26) generally leads to an exponential growth in the number of constraints (represented by (4.28)) over the solution search space. If (4.22) was continuously differentiable, the well-known gradient descent algorithm can be used to solve (4.26). The function $L(x; \phi)$ is, however, a piecewise linear convex function and, therefore, non-differentiable at every point. This leads to the sub-gradient method being a more suitable algorithm to solve (4.26) (Zhao et al., 1999; Guignard, 2003). The sub-gradient method is an iterative solution algorithm wherein steps are taken in a direction of descent based on some sub-gradient δ_j^t (Guignard, 2003). At each iteration t, calculate a sub-gradient δ_j^t

(associated with the j-th relaxed constraint, $j \in J$) such that

$$\delta_j^t = (g_j(x)) \ \ \forall j \in J, \tag{4.29}$$

where $g_j(x)$ is obtained by optimising (4.22), with each $\phi_j \in J$ fixed to a constant value (as determined by the previous sub-gradient iteration) [4]. The value for $g_j(x)$ will differ at each subsequent iteration since the Lagrangian multipliers are adjusted to limit constraint violations. The magnitude and sign of each sub-gradient are directly proportional to the extent of the constraint violations. The step size θ^t, in the direction of descent (based on the sum of each sub-gradient δ_j^t), is usually calculated using the Polyak step size formula and is given by

$$\theta^t = \frac{UB - LB}{||\delta^t||_2^2} \tag{4.30}$$

where UB refers to the upper bound of the optimisation problem (which is calculated while solving (4.22)) and LB refers to the lower bound of the optimisation problem, which represents the best integer solution (obtained from solving (4.19)). The equation in (4.30) allows dynamic adjustment of the step size θ^t according to the difference between the upper and lower bound values. This will aid convergence by reducing the number of iterations required to reach an acceptable value for (4.26). Algorithm 4.1 describes the steps associated with the sub-gradient algorithm that can be used to solve (4.26).

Algorithm 4.1 Sub-gradient algorithm

1: Set $\phi_j^t \ \forall j \in J$ equal to 0.1.
2: **for** ($t \leftarrow 1$ to T) **do**
3: Compute $L(x; \phi^t)$ for fixed values of $\phi_j^t \ \forall j \in J$.
4: Determine the corresponding values for x based on the solution of $L(x; \phi^t)$ and the fixed values for $\phi_j^t \ \forall j \in J$.
5: Calculate a sub-gradient δ_j^t such that $\delta_j^t = g_j(x)^t$ at point $\phi_j^t \ \forall j \in J$.
6: **if** ($\sum_{j \in J} \delta_j^t = 0$) **then**
7: Lack of constraint violations means the optimal solution is obtained.
8: Terminate the algorithm.
9: **else**
10: Compute $\phi_j^{t+1} = \phi_j^t + \delta_j^t \theta^t \ \forall j \in J$ where θ^t is the step size and δ_j^t the sub-gradient.
11: If the relaxed constraint was an equality constraint, $\phi_j^{t+1} \ \forall j \in J$ is allowed to take on any sign such that $\phi_j^{t+1} \leq 0$ or $\phi_j^{t+1} \geq 0$.
12: If the relaxed constraint was an inequality constraint, $\phi_j^{t+1} \ \forall j \in J$ would be constrained according to $\phi_j^{t+1} \geq 0$ such that $\phi_j^{t+1} = \max(0, \phi_j^{t+1})$.
13: Start the next iteration $t + 1$.
14: **end if**
15: **end for**

It is important to note that every sub-gradient iteration does not always guarantee an improvement in the objective function in (4.26). However, the algorithm will converge as

[4] At iteration $t = 1$ of the sub-gradient algorithm, the Lagrangian multipliers $\phi_j \in J$ are initialised to an arbitrary value. In subsequent iterations, the multipliers will be calculated according to Algorithm 4.1.

$t \to \infty$. As such, the objective function value that is found at each iteration should be recorded to ensure the best objective function value (from the set of solutions generated as part of each algorithmic step) is discovered. This relates to the sharpest upper bound associated with the original optimisation problem in (4.19). The magnitude of the step size θ^t is important to facilitate convergence. A step size that is too small will lead to an inflated number of algorithmic iterations required for Algorithm 4.1 to converge. The opposite is also true when θ^t is too large - the objective function may converge to a sub-optimal value (Guignard, 2003) [5]. The equation in (4.30) provides an adequate trade-off between convergence speed and the discovery of new solutions that approximate the optimal solution $f^*(\phi)$.

4.3 Summary

The fundamental theory behind Benders decomposition and Lagrangian relaxation was discussed as part of this Chapter. These optimisation techniques will be used in subsequent chapters to develop a well-rounded novel best subset selection solution framework. The main aim of both optimisation techniques is to make use of the MIP problem's underlying structure in such a way that its computational burden is reduced while also improving the quality of the solutions generated. Improvement in solution quality ultimately leads to a reduction in the optimality gap. Since the best subset selection problem is NP-hard, even a slight improvement in solution quality (which relates to a reduction in the optimality gap) can be seen as a significant contribution to the field of study. Chapter 5 will shed some light on the integration of Benders decomposition and Lagrangian relaxation into a novel solution framework that attempts to improve current subset selection algorithms.

[5] Please note that the definition of θ in Sections 4.1 and 4.2 of Chapter 4 differ. In Section 4.1, θ is used as dual variables, while Section 4.2 uses it as a step size. Let θ denote a step size for the remainder of the thesis.

Chapter 5

Mathematical models

Authors such as Venter (2020) and Sato et al. (2016b) have proposed variations of MIP approaches that attempt to solve the best subset selection logistic regression problem. The authors' proposed methodologies aim to generate parsimonious models for large-scale logistic regression problems.

Venter (2020) solved the Superconductivity logistic regression problem instance, found on the UCI machine learning repository, which contains around 58 predictors and 6000 training samples. The approach proposed by Venter (2020) utilised a direct linear approximation of the log-likelihood objective function, with 401 piecewise linear segments, and a cardinality constraint that enforces variable selection. The resulting solution produced a final model that contained 10 independent variables and exhibited an optimality gap of 22.57%. The model was allowed to run for 24 hours and a train and test accuracy of around 85.10% and 84.49% was achieved, respectively. The MIP formulation proposed by Venter (2020) does not regularise regression parameter estimates and, therefore, allows for straightforward statistical inference. This is a favourable attribute that the framework proposed later in this chapter will incorporate as well.

Sato et al. (2016b) solved multiple logistic regression classification problems from the UCI machine learning repository, the largest of which was the spam classification problem. The problem instance contained a training set of 4610 observations and 58 independent variables. While a problem instance with 90 variables was also considered, we note that the training set consisted of a mere 360 observations, which can be seen as a quite small sample. The authors attempted to solve the subset selection problem by using deterministic MIP-based approaches (with the branch-and-bound algorithm serving as the core underlying component) and well-known stepwise algorithms. The MIP optimisation ran for more than 10 000 seconds while trying to minimise the AIC in the one instance and the BIC in the other. Refer to Chapter 3 of this document for a detailed overview of the limitations associated with regularised variable selection MIP formulations, such as those applied by Sato et al. (2016b).

Recall from Chapter 1 that the key focus areas of our research endeavour include.

- facilitating the use of larger training sets during model fitting,
- reducing the computational memory requirement associated with solving logistic regression subset selection MIP frameworks,
- automating the specification of the cardinality constraint and the selection of the linear grid points.

In this chapter, the MIP formulation proposed by Venter (2020) will be presented, after which the proposed novel ALRSF will be discussed. The chapter concludes with a discussion on the perceived value that is added by the solution framework that is proposed in this thesis (over and above other MIP-based best subset selection techniques that were discussed in previous chapters).

5.1 Model 1: Piecewise linearised log-likelihood MIP formulation - existing approach

Some of the notation that was defined previously will be defined again in this section for the sake of completeness. Let p represent the number of independent variables and j the j-th independent variable associated with the logistic regression problem instance such that $j \in p = \{0, 2, ..., p\}$. Notice that $p = 0$ refers to the intercept term while $j = 1, .., p$ represent the j-th independent variable. Furthermore, let $\beta \in \mathbb{R}^p$ denote a vector of size p representing the regression parameters associated with each of the independent variables. Let i denote the i-th observation in the training set such that $i \in n = \{1, 2, ..., n\}$ where n denotes the total number of observations. The dependent and independent variables for the i-th training observation are denoted by Y_i and $X_i \in \mathbb{R}^p$, respectively, where X_i is a vector of size p. It is important to note that the independent variables X_i are normalised such that they are between zero and one. This ensures that the variables are expressed on the same scale, which leads to regression parameter estimates that are more comparable with one another. Let $k \in K = \{1, 2, ..., K\}$ denote the number of piecewise linear grid point intervals and γ_k the grid point values associated with each interval, as shown in Figure 5.1. Note that $g(\beta; \mathbf{X}, \mathbf{Y})$ in Figure 5.1 denotes the log-likelihood objective function that is used in logistic regression.

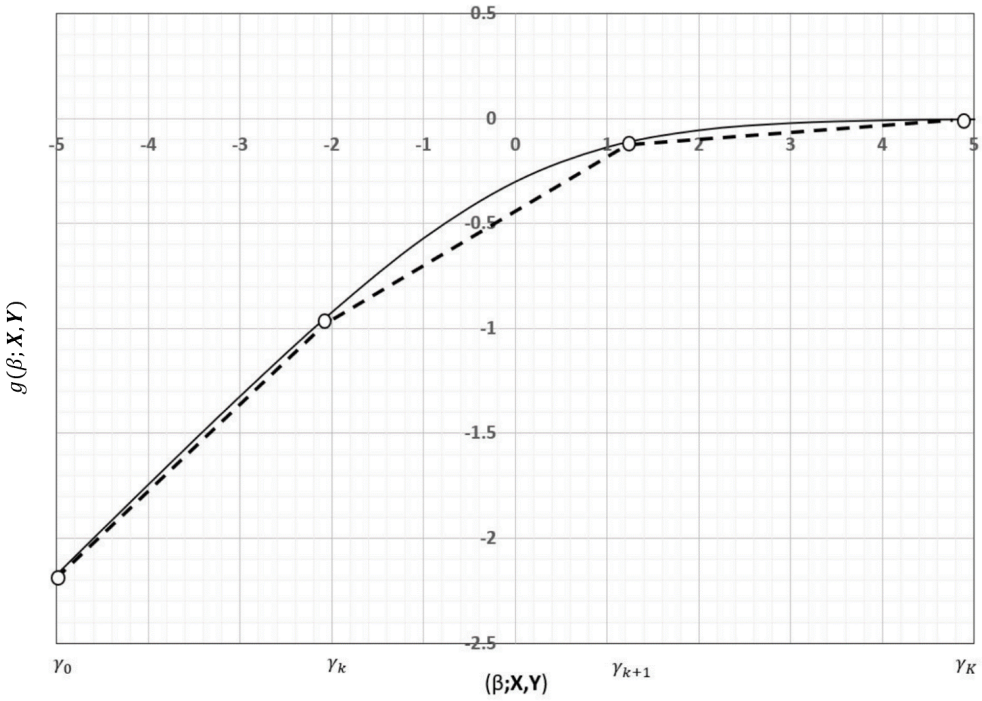


Figure 5.1: Piecewise linear underestimation of the log-likelihood function

At this point it is worthwhile to clarify the necessity of the piecewise linear approach proposed by Venter (2020). A piecewise linear approximation of the log-likelihood is considered for the simple fact that the log-likelihood is non-linear. If the log-likelihood were to be used as-is within a MIP formulation (which will be presented shortly), it would result in a non-linear objective function that, in turn, leads to a non-linear MIP. Non-linear MIPs can not be solved by using the well-known branch-and-bound algorithm and would therefore complicate the solution process significantly. As such, linearising the log-likelihood objective function (as visualised by the dashed lines in Figure 5.1) simplifies the solution process by enabling use of the standard branch-and-bound algorithm. Let the interval selection variable Γ_{ik} denote the k-th weight value associated with the k-th grid value for the i-th observation and $\{\Lambda_{ik} \in \mathbb{R} | 0 \leq \Lambda_{ik} \leq 1\}$ [1].

Mathematically, the piecewise linear approximation proposed by Venter (2020) can be defined as a convex combination of grid points that are expressed as

$$\Lambda_{ik}\gamma_k + \Lambda_{ik+1}\gamma_{k+1} = \beta^T X_i \;\; \forall i \in \{1, ..., n\}, \forall k \in \{1, ..., K\}.$$

The following proof is provided by Venter (2020), which shows that the proposed linear approximation of the log-likelihood function serves as an underestimation of the original non-linear log-likelihood.

[1] As mentioned in Chapter 3, the branch-and-bound algorithm will not be explained in detail as part of this thesis. However, a detailed explanation can be found in Venter (2020) Section 4.2.2 of Chapter 4.

Proof.

$$Y_i(\beta^T X_i) - \log[1 + \exp(\beta^T X_i)] = Y_i(\Lambda_{i1}\gamma_1 + \Lambda_{i2}\gamma_2) - \log[1 + \exp(\Lambda_{i1}\gamma_1 + \Lambda_{i2}\gamma_2)],$$
$$\leq Y_i(\Lambda_{i1}\gamma_1 + \Lambda_{i2}\gamma_2) - (\Lambda_{i1}\log[1 + \exp(\gamma_1)] + \Lambda_{i2}\log[1 + \exp(\gamma_2)]).$$

For a large enough value of K we have that $\gamma_{k+1} \approx \gamma_k$ such that

$$Y_i(\beta^T X_i) - \log[1 + \exp(\beta^T X_i)] \approx Y_i(\sum_{k \in K} \Lambda_{ik}\gamma_k) - \sum_{k \in K} \Lambda_{ik}\log[1 + \exp(\gamma_k)]$$

which yields

$$Y_i(\beta^T X_i) - \log[1 + \exp(\beta^T X_i)] = \sum_{k \in K} (\gamma_k Y_i - \log[1 + \exp(\gamma_k)])\Lambda_{ik}.$$

□

The value of K (which denotes the number of grid point intervals or linear splines) depends on the choice made by the model developer. A larger value for K would include more linear splines in the approximation, which would lead to a closer approximation of the non-linear log-likelihood. However, while a larger K yields a more accurate approximation, it produces a MIP that is computationally more expensive to solve. In contrast, a small value for K would lead to a greater underestimation of the log-likelihood function, but will reduce the computational effort required to solve the linearised MIP. There is therefore a trade-off between approximation accuracy and computational efficiency. For the remainder of this document, assume that $K = 401$. This means that the log-likelihood will be approximated by 401 linear splines, where the length of the k-th spline is defined by two grid points γ_k and γ_{k+1}. For example, observe from Figure 5.1 that the line segment between γ_0 and γ_k is defined by a single spline. Logically, the length of each spline will decrease as K increases such that we obtain the original non-linear log-likelihood when K tends to infinity. Let us now discuss the interval selection variable $\Lambda_{ik} \in \mathbb{R}$, which was introduced previously. The interval selection variable is used as a supporting decision variable to relate the predicted values associated with each observation (denoted by βX_i) to each linear spline (that spans γ_k and γ_{k+1}), which approximates the value of the log-likelihood function at the k-th grid point of the i-th training sample. In other words, the interval selection variable can be thought of as a variable that facilitates interpolation between two adjacent grid points γ_k and γ_{k+1}. This allows the MIP solution algorithm to select which spline would be appropriate to relate the x-axis values to the corresponding y-axis outcome. Since the independent variable vector X_i is fixed for each observation i, the choice of the regression parameter vector β and how it relates to the linearised objective function will be managed by Λ_{ik}. Furthermore, as part of the proposed linearised log-likelihood, extreme points should be defined which dictate the boundary conditions of the solution search space (i.e. the start and end points of the x-axis in Figure 5.1 should be specified). The extreme points of the grid will be denoted by $-\Gamma = -\gamma_0$ and $\Gamma = \gamma_K$. Venter (2020) proposed an approach wherein these extreme points are determined up front before the subset

selection MIP is solved. When the true underlying regression parameters are not known beforehand, a starting point would be to fit a full logistic regression model (which will be defined shortly) that excludes any subset selection. In other words, the logistic regression is solved in the absence of any variable selection. The model developer can then deduce the extreme points of the grid via an empirical analysis once the parameter estimates for the full logistic regression model are obtained. Firstly, let

$$M_1 = \min\{\sum_{j\in p} \hat{\beta}_j X_{1j}, ..., \sum_{j\in p+1} \hat{\beta}_j X_{nj}\},$$

and

$$M_2 = \max\{\sum_{j\in p} \hat{\beta}_j X_{1j}, ..., \sum_{j\in p+1} \hat{\beta}_j X_{nj}\},$$

where p denotes the number of independent variables and n the number of observations. The range of the grid is then given by

$$[M_1 - b_1; M_2 + b_2],$$

where b_1 and b_2 are margins of conservatism that are added to the minimum and maximum values of the grid points, respectively, to ensure that the complete search space is considered as part of the MIP solution process. Using $5 \leq b_1 \leq 20$ and $5 \leq b_2 \leq 20$ were found to be sufficient for real-world datasets, according to Venter (2020). This approach involves a manual (and sometimes iterative) process that needs to be executed first before the solving of the subset selection MIP can be initiated. When the proposed approach is applied to real-world problems, Venter (2020) found that a grid range of [-30;30] is sufficiently large such that it describes the complete solution search space associated with the UCI Junk-mail classification problem. It is important to note that the extreme points of the grid are problem-dependent and will be different for every model training set.

Let $z \in \{0, 1\}^p$ denote a cardinality selection variable vector of size p, which acts as an on-off switch to identify the specific independent variables that should be included as part of the final variable subset. Note that index $j = 0$ represents the intercept term. As such allow z_0 to take on a value of one such that $z_0 = 1$ to ensure the intercept term is included in the model by default. In conjunction with Big-M constraints, the cardinality selection variable for the j-th independent variable (denoted by z_j) governs the solution search space and defines the range in which β_j can operate. If the binary cardinality selection variable z_j takes on a value of 0, the repression parameter $\hat{\beta}_j$ will also assume a value of 0. Alternatively, when z_j is equal to 1, $\hat{\beta}_j$ can take on any value in the range $[-M, M]$ (note that M is not the same as M_1 and M_2). To ensure the complete solution search space is considered, sufficiently large Big-M lower and upper bound values must be utilised. As such, M can be set close or equal to the integer limit allowed by the solver. In the case of the computing software Cplex (the solver of choice for this thesis), the integer limit is equal to $2.15 \cdot 10^9$. Therefore, M can take on quite a large integer value [2]. While this might simplify the choice of M, it is worth noting that constraining M to a more modest value

[2] Cplex is a commercially available software package that is designed by IBM and used specifically for optimisation problems.

may be sufficient at describing the complete solution search space while also producing an identical final solution. It is therefore advisable that model developers consider the unique properties of each modelling problem and select the value of M based on industry knowledge. Alternatively, an iterative approach can be considered by first setting M equal to an arbitrarily large value (close to the integer limit) after which the size of M is reduced during iterative post-hoc analyses once initial regression parameter estimates have been identified in order to improve on the quality of the optimal solution.

Let c denote the cardinality parameter that limits the number of variables that can be included in the final MIP solution (over and above the intercept term since the intercept term is by default considered as part of the logistic regression model). As part of the experiments conducted by Venter (2020), values ranging from 6 to 15 were used for the cardinality selection variable c. These values were determined by studying incremental improvements in the AUC as variables were iteratively added to the model (starting with $c = 1$ and progressing until the full model, where $c = p$, is fitted). This was achieved by using SAS (statistical software) as a solution tool and ultimately selecting c based on the point at which the AUC starts to plateau, as depicted by the red coordinate in Figure 5.2. In other words, SAS sequentially fits a j-variable model, starting with $j = 1$ and ending with $j = p$, until the full range of possible independent variables have been considered for inclusion in the final model. If c represents the point on the x-axis in Figure 5.2 where the AUC start to level off, then we can argue that adding more than c variables to the final model does not contribute significantly to the predictive power of the model. Ultimately, the value of c can be informed by the saturation of some model performance measure (e.g. AUC, Gini coefficient, accuracy, etc.).

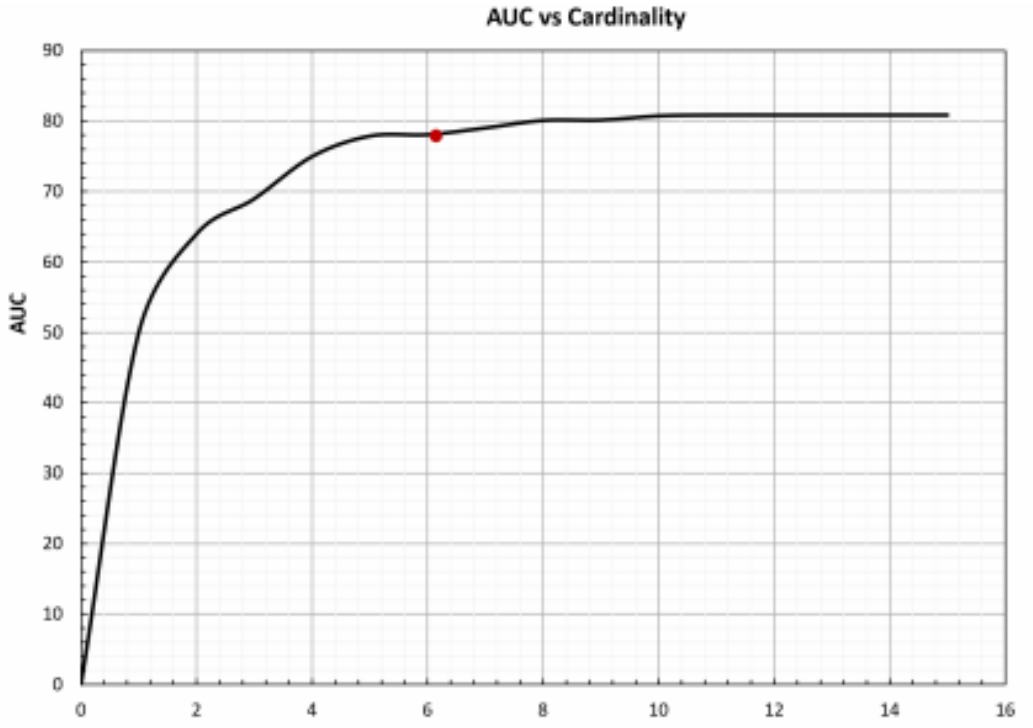


Figure 5.2: AUC performance measure as a function of cardinality (c)

Considering the background provided thus far, the linearised best subset selection MIP formulation proposed by Venter (2020) can now formally be introduced. The objective of the piecewise linear approximation of the log-likelihood function is to

$$\max_{\beta,\Lambda,z} \sum_{i\in n}\sum_{k\in K}(\gamma_k Y_i - \log[1+\exp(\gamma_k)])\Lambda_{ik}, \tag{5.1}$$

subject to

$$\sum_{k\in K}\Lambda_{ik}\gamma_k = \beta^T X_i \ \ \forall i \in n, \tag{5.2}$$

$$\sum_{k\in K}\Lambda_{ik} = 1 \ \ \forall i \in n, \tag{5.3}$$

$$0 \leq \Lambda_{ik} \leq 1 \ \ \forall k \in K; \forall i \in n, \tag{5.4}$$

$$|\beta_j| \leq M z_j \ \ \forall j \in p, \tag{5.5}$$

$$\sum_{j\in p} z_j \leq c+1. \tag{5.6}$$

The equation in (5.2) serves as a linearisation constraint which equates the summation of the regression parameters and the independent variables with each line segment that approximates a portion of the log-likelihood function. In conjunction with (5.2), (5.3) ensures that the sum of the weights assigned to the individual grid points between the line segments are equal to one for each observation so as to enforce convexity. It is evident from (5.4) that Λ_{ik} can only take on values between zero and one. As mentioned earlier, the decision variable Λ_{ik} can be thought of as an interpolation variable that selects the appropriate point on the spline between the two grid points γ_k and γ_{k+1} which is then used to relate the sum of the regression parameters and independent variables $\beta^T X_i$ to the approximated log-likelihood. The constraint in (5.5) acts as a binary switching constraint that sets the value of the j-th regression parameter equal to zero if the MIP determines that the inclusion of the j-th variable in the final model is not warranted. The total number of variables allowed in the final MIP solution is constrained by the sum of the binary switching variables, as expressed in (5.6). The constraint in (5.6) allows the MIP to select any number of independent variables as long as the total number is less than or equal to the cardinality parameter c. Note that the right hand side of the constraint is increased by one (by default) to accommodate for the addition of the intercept term. The inclusion of the intercept term is facilitated by $z_0 = 1$ which means that the value of c can be used to facilitate inclusion or exclusion of the remaining independent variables (without the need to consider the intercept term). The above convention is used throughout the remainder of the thesis. The MIP presented in (5.1) - (5.6) can be solved by using the conventional branch-and-bound algorithms that are native to Cplex.

5.2 Model 2: Automated logistic regression solution framework (ALRSF)

The ALSRF proposed in this chapter combines a heuristic that enforces regression parameter fixing, Benders decomposition and Lagrangian relaxation to allow for an improved

deterministic solution framework. The solution framework consists of four main steps, which include the cardinality step ($S = 0$), search space pruning step ($S = 1$), optimality step ($S = 2$) and bound tightening step ($S = 3$). The ALRSF is derived from, and based on, the MIP that was introduced in Section 5.1 of Chapter 5 previously. Some minor modifications to the constraints presented in Section 5.1 of Chapter 5 are introduced, which enable the solution framework to apply the Cplex native Benders decomposition algorithm. This allows for a reduction in memory requirement, an increase in the size of the problem that can be considered and enhanced numeric stability. Sections 5.2.1 to 5.2.3 of Chapter 5 will focus on the first three steps of the ALRSF (denoted by $S = \{0, 1, 2\}$) which attempts to improve the lower bound of the maximisation problem that maximises the log-likelihood. Sections 5.2.4 to 5.2.5 of Chapter 5 will provide an exposition on Step $S = 3$, which aims to tighten the upper bound by utilising a Lagrangian reformulation of the MIP presented in Section 5.2 of Chapter 5. The modified MIP (that is applied in the first three steps of the ALRSF) is presented first, after which the solution framework is described in Section 5.2.2 of Chapter 5. Section 5.2.4 of Chapter 5 will be focused on deriving a Lagrangian relaxation of the MIP presented in Section 5.2, after which the sub-gradient-based solution framework will be discussed in Section 5.2.5. The mathematical notation presented in Section 5.1 still applies to the subsequent sections.

5.2.1 ALRSF - Proposed MIP formulation (Steps 0 - 2)

Let p' denote a subset of indices contained in p such that $p' \subseteq p$. The subset p' contains the index of each independent variable that should be excluded from the best subset MIP formulation and can be obtained as part of a pre-processing step that analyses the variance inflation factor (VIF) of each independent variable. The VIF serves as a measure of multicollinearity between independent variables. Furthermore, it gives insights into the extent by which the regression parameter estimates will be inflated if the correlations between variables are not addressed. When a variable is associated with a large VIF, it indicates the presence of a severe multicollinearity problem and shows that the variable being considered is correlated with one or more of the remaining variables in the set. Specifically, the VIF of each variable provides an indication of how well the rest of the variables in the set will be able to describe the variable in question by means of a linear regression model. As part of a pre-processing step, a VIF threshold L can be defined that dictates whether the j-th variable should be included in the model, where $j \in p$ when $VIF_j \leq L$ and $j \in p'$ when $VIF_j > L$. Since subset p' is identified prior to execution of the solution algorithm, it is a static set of indices. It will only change when the exclusion threshold L is adjusted. The value of L can be selected based on industry experience or industry best practises. As a rule of thumb, a value of $L = 10$ generally relates to severe multicollinearity (Obrien, 2007). As such, assume that $L = 10$ for the remainder of the thesis. The decision variable $\beta' \in \mathbb{Z}^p$ is a vector of size p and denotes an integer representation of the rescaled relational regression parameter variable $\beta \in \mathbb{R}^p$. The integer representation of the regression parameter vector β is used to alter the structure of the constraints such that the default Cplex Benders decomposition algorithm can be applied

to solve the logistic regression MIP formulation. The constant H acts as a delimiter and facilitates the conversion of β' from an integer variable to β, which is a rescaled relational variable that is expressed as $\beta = \frac{\beta'}{H}$. An arbitrary value can be chosen for H. However, it is important to note that a larger value for H will ensure a greater decimal precision of the rescaled relational approximation associated with β but will simultaneously increasing the solution search space. The latter will be covered in detail later in this section. The piecewise linearised underestimation of the log-likelihood solves an objective function that is given by

$$\max_{\beta',z,\Lambda} \sum_{i\in n} \sum_{k\in K} (\gamma_k Y_i - \log[1 + \exp(\gamma_k)])\Lambda_{ik}, \tag{5.7}$$

Subject to:

$$\sum_{k\in K} \Lambda_{ik}\gamma_k = \frac{\beta'^T}{H} X_i \ \ \forall i \in n, \tag{5.8}$$

$$\sum_{k\in K} \Lambda_{ik} = 1 \ \ \forall i \in n, \tag{5.9}$$

$$0 \leq \Lambda_{ik} \leq 1 \ \ \forall k \in K; \forall i \in n, \tag{5.10}$$

$$\frac{\beta'_j}{H} = 0 \ \ \forall j \in p', \tag{5.11}$$

$$|\frac{\beta'_j}{H}| \leq M z_j \ \ \forall j \in p, \tag{5.12}$$

$$\sum_{j\in p} z_j \leq c + 1. \tag{5.13}$$

The equation in (5.11) ensures that all the regression parameter estimates associated with variables that exhibit $VIF \geq L$ be set equal to zero. The indices of the variables that should be excluded as a result of the VIF threshold are contained within the subset p' as mentioned earlier. The transformation $\beta = \frac{\beta'}{H}$ relates the integer regression parameter vector β' to the rescaled regression parameter vector β. The resulting regression parameters are rescaled relational parameters with the scale governed by H. This allows the solution framework to exploit the unique block-diagonal structure of (5.8) such that the Cplex native Benders decomposition algorithm can be invoked. A block-diagonal structure refers to a matrix structure where smaller sub-matrices are arranged along the main diagonal of the matrix. While the size of these matrices can differ, it is important to note that each matrix must be square and that the remainder of the larger matrix must be populated with zero values. Refer to Chapter 8 of Venter (2020) for a visual representation of the aforementioned block-diagonal matrix structure. Such a block-diagonal structure ensures that there is no interaction between the sub-matrices (or sub-problems) such that the overall MIP can be decomposed into a master problem and one or more sub-problems, which can be seen as a significant advantage (refer to Chapter 4, where an overview of master problems and sub-problems was provided). The constraint matrix can be divided into smaller and simpler parts that can be solved separately, which ultimately reduces the number of variables and constraints that are considered by each sub-problem. This gives rise to an iterative fix-and-solve approach that is utilised as part of Benders decomposition, which converges to a solution that is equal to or close to the optimal solution. Specifically, for the best subset selection logistic regression MIP, a valid decomposition would not be

possible if the continuous regression parameters (denoted by β) are not transformed to integer representations (denoted by β'). In other words, this means that the complete MIP will have to be solved with a standard branch-and-bound algorithm if the aforementioned transformation is not applied, which is more rigorous, less efficient and computationally more expensive than a Benders decomposition approach [3].

Table 5.1 seeks to explain the reasoning behind the regression parameter transformation that is proposed by the MIP, where the influence of various values for H are shown.

Table 5.1: Integer to rescaled relational regression parameters

Parameter Number	Integer parameter β'	Rescaled rational parameter β (H = 10)	Integer parameter β'	Rescaled rational parameter β (H = 100)	Integer parameter β'	Rescaled rational parameter β (H = 1000)
W_1	12	1.2	125	1.25	1257	1.257
W_2	15	1.5	158	1.58	1581	1.581
W_3	89	8.9	890	8.90	8901	8.901
W_4	500	50.0	5008	50.08	50089	50.089
W_5	6	0.6	68	0.68	689	0.689
W_6	1	0.1	11	0.11	113	0.113
W_7	78	7.8	783	7.83	7830	7.830

Observe from Table 5.1 that the choice of the constant delimiter H influences the decimal precision of the final regression parameter estimates. A larger H affords Cplex the flexibility to select larger integer values for β', which in turn allows for greater decimal accuracy. A trade-off however exists between parameter accuracy and the size of the search space. It is important to note that H can not be significantly large due to the integer size limitation of Cplex, which is $2.15 \cdot 10^9$. The choice of the delimiter H should therefore consider the variability of the integer decision variable β'. A fixed value of $H = 1000$ allows $|\beta'|$ to take on any value between 0 and $2.15 \cdot 10^6$. With $H = 1000$, β' can either take on larger values with less decimal precision or smaller values with greater decimal precision. As such, we should also take note that there is a trade-off between how large the regression parameter estimates can be (in absolute terms) and the parameters' decimal precision. A moderate value for H is proposed to allow for adequate variability of β' in either direction (integer vs decimal precision), as depicted in Figure 5.2.

In general, multicollinearity results in unstable and unreliable regression parameter estimates. This means that regression parameter estimates become very sensitive to small changes in the model. In turn, the parameter instability will have a large impact on the choice of the extreme points of the grid (discussed in Chapter 5.1) that is used to express the linearised log-likelihood. This will become clearer in the next sections of Chapter

[3] This is substantiated by the fact that Benders decomposition can divide the MIP into a master problem and numerous sub-problems, making the solution approach less computationally intensive.

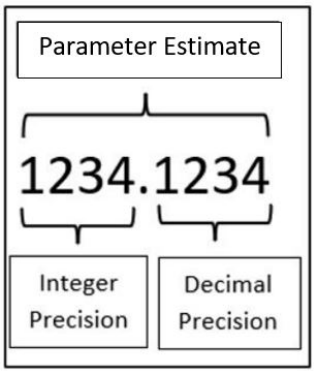


Figure 5.3: Independent variable parameter estimate integer and decimal precision representation

5. The VIF exclusion constraint in (5.11) therefore attempts to reduce collinearity and improve the numerical stability of the solution algorithm. As mentioned earlier, the VIF threshold can be informed by industry best practices, where $VIF = 10$ seems to be a popular choice.

5.2.2 ALRSF - Proposed Solution Framework (Steps 0 - 2)

As mentioned previously, the majority of MIPs (in both practice and academia) are solved by using the straightforward branch-and-bound technique. In most cases the branch-and-bound algorithm is able to produce acceptable results that are relatively close to the optimal solution. However, in some instances (especially when the MIP is NP-hard), more advanced techniques are required to find an acceptable solution. An adequate example of such an NP-hard problem is the best subset selection problem that was discussed in Section 5.2.1 of Chapter 5. The standard branch-and-bound algorithm may be able to produce adequate and feasible solutions for moderately-size problems, but lacks the ability to yield high-quality solutions (that are close to optimality) when the number of independent variables or size of the training set increases significantly. Additionally, a larger training set or independent variable set translates to extensive computational expenses. As such, an alternative solution approach is required that will ensure high-quality feasible solutions are still possible, even when large-scale variable selection problems are considered.

There are mainly four aspects of the best subset selection MIP in (5.7) - (5.13) that need to be improved upon to ensure that high-quality solutions (which are close to optimality) are produced by the solver. This includes improvements with respect to

1. the increase in computational requirement associated with an increase in both the number of observations n and the number of independent variables p,
2. the solution time associated with finding a high-quality feasible solution that is either optimal or close to the optimal solution,
3. the current need to execute manual pre-processing steps to determine the most

appropriate values for the cardinality variable c and the extreme points of the grid (denoted by $-\Gamma$ and Γ) before the actual MIP problem (as defined by Venter (2020)) can be solved, and

4. the standard branch-and-bound algorithm, which seems to be limited in the sense that it can mostly produce high-quality solutions for moderately-sized best subset selection problems.

In an attempt to address the above-mentioned limitations, an ALRSF is proposed that consists of four distinct steps: the cardinality selection step $S = 0$, search space pruning step $S = 1$, optimality step $S = 2$ and bound tightening step $S = 3$. Furthermore, the ALRSF also makes use of advanced decomposition techniques that will greatly improve the efficiency of the MIP solver. The algorithmic flow of the proposed ALRSF is depicted by Figure 5.3.

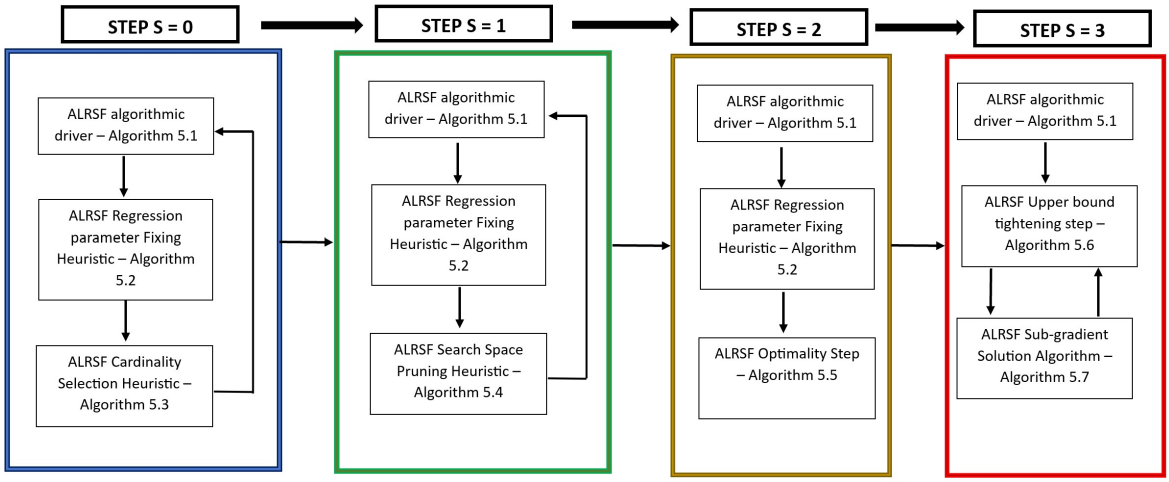


Figure 5.4: ALRSF Logical process flow

Before we continue with a detailed discussion of the ALRSF, it is worthwhile to explain why the aforementioned four steps of the solution framework will yield the desired improvements over the straightforward branch-and-bound algorithm (as listed at the start of this section). As discussed previously, standard branch-and-bound algorithms generally become impractical when the solution search space starts to increase for large NP-hard MIP problems. Throughout history, multiple different techniques have been proposed to alleviate the limitations of the branch-and-bound approach, one of which includes Benders decomposition. The fact that the overall problem can be divided into a master problem and several sub-problems is one of the main reasons why decomposition approaches are so efficient at finding high-quality solutions. This gives rise to a fix-and-solve approach, which does not require the MIP to evaluate the entire solution search space in a single attempt at finding the best solution. This means that it is no longer a requirement to consider the complete solutions search space in isolation in an attempt to finding the optimal solution. The development of the proposed ALRSF draws on the fundamental principle that dividing the problem into manageable sub-problems might greatly reduce the computational

requirements while allowing the discovery of solutions that are closer to optimality. To that effect, the first two steps of the ALRSF can be seen as heuristics that are used to identify the most appropriate cardinality value to use without solving p logistic regression MIP problems (which in turn reduces the time required to find c) after which the problem is divided into manageable sub-problems. The second step of the ALRSF is a search space pruning heuristic, which means that it will not be able to prove optimality, but instead is mainly aimed at pruning the search space such that the subsequent optimality step only needs to consider a subset of possible solutions as potential candidates. This suggests that the solving of the MIP in the third step (the optimality step) will be less computationally intensive. To summarise, the proposed solution is mainly motivated by the desire to divide the MIP into smaller, more manageable problems that are less complex to solve, thereby making the best subset selection problem more efficient. Furthermore, the ALRSF is also guided by the need to solve large MIP problem instances. The remainder of this section will be focused on the first three steps (denoted by $S = \{0, 1, 2\}$) of the ALRSF, while step four (denoted by $S = \{3\}$) will be discussed in Sections 5.2.4 and 5.2.5.

The MIP formulation that was provided in Section 5.2.1 is utilised in the first three steps $S = \{0, 1, 2\}$ and is solved using Cplex's built-in Benders decomposition algorithm. It is imperative to note that the MIP in Section 5.2.1 is intrinsically linked to the ALRSF, since the proposed solution framework attempts to solve the MIP in a more efficient manner. Each step of the ALRSF is therefore aimed at

1. reducing the time required to find the optimal solution, and
2. improving the quality of the final solution of the best subset selection MIP proposed by Venter (2020).

As discussed in Chapter 4, the MIP constraint matrix should adhere to a specific block structure for the problem to be solved by means of the Benders decomposition algorithm. Benders decomposition should be applied when a vector of complicating variables is present in the MIP problem instance such that fixing the complicating variables produces a problem that is easier to solve. The MIP in Section 5.2.1 is much simpler to solve when the complicating subset selection variables in the vector $z \in 0, 1^p$ and integer regression parameters β' are temporarily fixed. When the aforementioned constraint-fixing is applied, the MIP from Section 5.2.1 can easily fit within the confines of the Benders decomposition paradigm. Specifically, the constraints in (5.9) - (5.13) can be restructured such that their form is in line with the standard form of the Benders decomposition methodology. The equation in (5.8) is analogous to the connecting constraint (which is described by the equation in (4.3) from Chapter 4) since it contains both integer and continuous decision variables β' and Λ_{ik}, respectively. Cplex's default Benders decomposition algorithm automatically decomposes the MIP into master and sub-problems when Cplex's Benders decomposition property is set equal to 3. All integer variables are allocated to the master problem, while all continuous variables are assigned to the sub-problem. When bender's decomposition is applied to the MIP from Section 5.1, all decision variables (except for the cardinality selection variable) will automatically be allocated to the sub-problem. This

will cause an invalid decomposition in Cplex. To solve this, a simplistic adjustment to (5.1) - (5.6) is proposed by substituting the continuous regression parameters β with integer parameters β'. For the final result, the integer parameter estimates are divided by a moderately large constant H to produce rescaled relational regression estimates. While this continuous-to-integer conversion may marginally reduce the decimal precision of the final parameter estimates, it enables the use of the Cplex Benders decomposition for improved efficiency.

It is important for the reader to note that the first three steps of the ALRSF all make use of a Benders decomposition approach to solve the best subset selection MIP. Additional logic that relates to the heuristic nature of the first step, the identification of the cardinality parameter c and the pruning of the search space will be provided later in this chapter and does not detract from the fact that the Benders decomposition is used as the main algorithm - instead of the traditional branch-and-bound approach - to solve the MIP. This differs from other heuristic-based approaches (mentioned in Chapter 4) that make use of custom solution algorithms or algorithms that focus on finding high-quality local optima. It should be noted that each of the heuristic-based steps add greater value when a MIP-based solution approach is used. Alternatively, if an approach was employed that only finds local optima (which is often the case for many regression methodologies), each heuristic iteration will only reduce the search space by a small-to-insignificant margin as a result of locally (and not globally) optimal solutions. Since Benders decomposition identifies the global optimum at each iteration, the model developer can be assured that the heuristic steps prune the solution search space to a point where no further improvement is possible. This would then necessitate the use of a subsequent solution approach that can consider the remaining candidate solutions in totality so that an optimality gap can be provided and/or existing solutions can be improved upon. This is why the optimality step is executed after the search space pruning step has concluded. Each of the first three steps of the ALRSF will now be discussed in more detail.

Let p represent the number of independent variables present in the logistic regression problem instance while the total number of algorithmic iterations are denoted by $T = p$. Furthermore, let j be the index of the j-th independent variable in the set of p variables, where $j = 0$ denotes the intercept term of the logistic regression model. Let t describe each individual iteration for $t = 0$ to $t = T = p$. The variable T is an input parameter and will be problem-dependent. Let S denote the four distinct steps associated with the proposed solution framework such that $S = \{0, 1, 2, 3\}$. Algorithm 5.1 provides a holistic overview of how each step fits into the proposed solution framework and acts as the driver algorithm that controls the ALRSF logic. The index j is also used to denote an internal algorithmic iteration counter that allows the ALRSF to navigate between regression parameter decision variables (denoted by β_j), since the number of possible regression parameters will be equal to the number of variables under consideration at any given point in time. Let β_0 denote the intercept term of the logistic regression model, which is included in the summation $\beta^T X_i$ that is used throughout this section such that $X_{i0} = 1$ when the index $j = 0$. Allow $[-\Gamma, \Gamma]$ to denote the extreme points of the grid that defines the range over which splines are used to linearise the log-likelihood objective function. The extreme points of the grid

have the exact same interpretation as in Venter (2020). Note, however, that unlike Venter (2020) (where the extreme points are derived by means of a manual pre-processing step), the extreme points of the ALRSF are identified automatically and subsequently adjusted by using an adjustment factor ψ. For the remainder of this thesis assume that $\psi = 10$ is used for the interval adjustment factor for all empirical exercises. For the remainder of this thesis, also assume that the AUC will be used to evaluate model performance, i.e. we have that $AUC_t = P_t$ is the model performance measure at iteration t and $AUC_b = P_b$ is the best AUC that is obtained thus far at any given point in time. Let VIF_j denote the variance inflation factor associated with each independent variable X_j where, as previously mentioned, X_j is a vector of size p. Furthermore, let IV_j denote the rank of the information value (IV) of the j-th variable (in relation to all independent variable's IVs) and not the j-th variable's actual information value. The information value is a measure of the predictive power of the independent variables in relation to the dependent variable. This measure is generally used as part of classification problems (where the target variable is binary) to rank independent variables based on their predictive power and can assist in variable selection. The IV of the j-th independent variable can be written as

$$IV_j = \sum_{bin \in B} WoE_{bin}(PE_{bin} - PNE_{bin}),$$

Where WoE_{bin} is referred to as the weight of evidence associated to each individual bin and PE_{bin} and PNE_{bin} denotes the proportion of event cases per bin and proportion of non-event cases per bin, respectively. The independent variable is split into B number of bins after which the weight of evidence is calculated for each bin. This is achieved by

$$WoE_{bin} = \ln(\frac{PE_{bin}}{PNE_{bin}}).$$

The IV associated with the j-th independent variable is therefore the sum of the WoE of each bin that is multiplied with the distance between the event proportion and non-event proportion of the particular bin [4]. Variables with large IVs are considered to be more predictive than variables with smaller IVs. Note that a thorough discussion of the advantages and disadvantages of the IV (in comparison with other metrics that are used to rank independent variables with respect to their predictive power) are considered to be out of scope for this thesis.

With the mathematical notation now thoroughly defined, an overview of the ALRSF's algorithmic intuition can be given at the hand of a step-by-step description of the ALRSF logic, as presented in Algorithm 5.1. Algorithm 5.1 is used as the algorithmic diver behind the ALRSF and decides when each step should be initiated or terminated. This can also clearly be observed when studying Figure 5.4.

[4] An event can be seen as a observation that is associated with a target variable value of $Y = 1$, while a non-event can be seen as an observation that has $Y = 0$.

Algorithm 5.1 - ALRSF algorithmic driver

$S = 0$
for $t = 1$ to T **do**
  Execute Algorithm 5.2 (regression parameter fixing heuristic) based on the value of S.
  if $S = 0$ **then**
    Execute Algorithm 5.3 (cardinality selection step) where $S = 0$.
  else if $S = 1$ **then**
    Execute Algorithm 5.4 (search space pruning step) where $S = 1$.
  else if $S = 2$ **then**
    Execute Algorithm 5.5 (optimality step) where $S = 2$.
  end if
end for
Execute Algorithm 5.6 (Bound tightening step) where $S = 3$.
Solution Algorithm Completed.

Algorithm 5.2 - Regression parameter fixing heuristic

if $(t \geq T)$ & $(S = 1)$ **then**
  $S = S + 1$
  $t = T - 1$
end if
if $S = 0$ **then**
  for $j = 1$ to p **do**
    if $(IV_j \leq t)$ **then**
      $\beta'_j =$ free
    else
      $\beta'_j = 0$
    end if
  end for
else if $S = 1$ **then**
  for $j = 1$ to p **do**
    if $(|\beta'^{t-1}_j| > 0)$ or $(IV_j = t)$ **then**
      $\beta'_j =$ free
    else
      $\beta'_j = 0$
    end if
  end for
else if $S = 2$ **then**
  for $j = 1$ to p **do**
    $\beta'_j =$ free
  end for
  Return to Algorithm 5.1 - (driver algorithm).
end if

Algorithm 5.3 - Cardinality selection step - $S = 0$

$c = 0$, $k = 0$, $\zeta = 0$, $u = 0$
P_t and $P_b = 0$
${\beta'}^b = 0$
for $j = 1$ to p' **do**
  $\beta'_j = 0$ if $VIF_j > L$
end for
Each regression parameter β'_j associated with the j-th variable for which $VIF_j > L$ is set equal to zero.
Solve logistic regression MIP presented in (5.7) - (5.13) using Benders decomposition.
Cardinality constraints in (5.12) and (5.13) are excluded from the MIP.
for $i = 1$ to n **do**
  $\Omega_i = 0$
  $\Omega_i = \Omega_i + \frac{{\beta'}^T}{H} X_i$
  if $(\Omega_i \leq -\Gamma)$ or $(\Omega_i \geq \Gamma)$ **then**
    $|\Gamma| = |\Gamma| + \psi$
    $\Delta\gamma = \frac{2|\Gamma|}{K}$
    $\zeta = 1$
  end if
end for
if $\zeta = 0$ **then**
  Calculate and record training and test performance measure P_t.
  Record ${\beta'}^t$.
  if $P_t > P_b$ **then**
    for $j = 0$ to p **do**
      ${\beta'_j}^b = \beta'_j$
    end for
    $P_b = P_t$
    $c = c + 1$
    $u = 0$
  else
    $u = u + 1$
  end if
  if $u = U$ **then**
    $S = S + 1$
    $t = 0$
  end if
else
  $t = t - 1$
end if
Cardinality parameter c is obtained as a result of this algorithm.
Save MIP solution found at iteration t for use as warm-start during the execution of iteration $t + 1$.
Increment $t = t + 1$.
Return to Algorithm 5.1 - (driver algorithm).

To ease the flow of the remainder of this Chapter, the reader is referred back to Figure 5.4, as the explanation that follows will start with the blue block in Figure 5.4 and progress towards the red block in Figure 5.4 (which will be discussed in Section 5.2.4). This visual representation attempts to clearly define the logical flow of the proposed solution framework.

We start with Algorithm 5.1 of the ALRSF when $S = 0$ and $t = 1$ (as shown in the first and second lines of Algorithm 5.1). This relates to the blue block in Figure 5.4. The blue block highlights the sequential nature of the cardinality selection step where $S = 0$. The main objective of this step is to automatically identify the most appropriate value for the cardinality parameter c for the specific training dataset being considered. With $t = 1$ and $S = 0$, the third line of Algorithm 5.1 dictates that the ALRSF should first enter the regression parameter fixing heuristic (Algorithm 5.2) to identify the appropriate code logic for $S = 0$. Algorithm 5.2 is a novel regression parameter fixing heuristic which facilitates the inclusion and/or exclusion of parameters in/from the MIP formulation (as discussed in Section 5.2.1) based on the logic that is outlined in step S. The main contribution of the regression parameter fixing heuristic is realised during the execution of the cardinality selection and search space pruning steps (denoted by $S = 0$ and $S = 1$). Algorithm 5.2 comprises three if-else statements, each describing logic specific to the value assigned to S. Since $S = 0$, the appropriate code block to consider is described by lines 5 to 12. Lines 5 to 12 in Algorithm 5.2 indicate which regression parameters β' should be set free and which should be fixed to zero. Note that when a regression parameter is set free, it is allowed to take on any integer value in the range $(-\infty, \infty)$. Specifically, within the context of the particular software used by the model developer, this means that regression parameters that are allowed to take on any integer value that does not exceed the maximum limit that is set by the solver. Given that $S = 0$ relates to the cardinality selection step, lines 7 to 11 in Algorithm 5.2 describe the specific variables that should be considered as part of the cardinality selection step.

At any given point in time, the number of independent variables that are present in the cardinality selections step should be equal to the iteration counter t - this can be seen by studying the if-then logic in line 7 of Algorithm 5.2. After considering the regression parameter fixing logic for $S = 0$, line 25 in Algorithm 5.2 indicates that the ALRSF should return to Algorithm 5.1. Since S is still equal to zero, the if-statement described by lines 4 to 5 will be triggered. The if-statement now dictates that the ALRSF should enter Algorithm 5.3 (the cardinality selection step). In general, Algorithm 5.3 is made up of two for-loops. The first loop (lines 4 to 6) is used to exclude all variables that are associated with VIF values that exceed the threshold L, while the second (lines 10 to 18) is used to ensure the solution search space that is governed by the extreme points $-\Gamma$ and $\Gamma]$ is sufficiently large. Since the algorithm is still at iteration $t = 1$, lines 10 to 18 will verify whether the solution search space should be expanded or whether it sufficiently describes the problem at hand (after solving the MIP from Section 5.2.1). If the solution search space is found to be insufficient, the iteration variable t will be decremented by one such that $t = t - 1$ (according to line 37) and then incremented by one in line 41 of Algorithm 5.3. Essentially, this means that the same steps of the MIP (that were discussed up until

this point) will be re-executed with larger extreme grid points so that the solution search space can be increased. Specifically, the range of the grid is increased from $[-\Gamma, \Gamma]$ to $[-\Gamma - \psi, \Gamma + \psi]$ in an attempt to describe the MIP more adequately as part of iteration $t = 1$. After making the relevant adjustments, line 42 of Algorithm 5.3 will direct the ALRSF back to Algorithm 5.1. Since $t = 1$ and $S = 0$, the same logic as discussed previously will be executed, i.e. Algorithm 5.2 will once again be used to fix/free the appropriate regression parameters in the vector β', after which iteration $t = 1$ associated with Algorithm 5.3 will be re-executed. This process continues until the solution search space is sufficiently described by $[-\Gamma - \psi, \Gamma + \psi]$, which means that the iteration counter t can now be incremented by one (such that $t = 2$). This iterative fix-and-solve process is described by the blue block in Figure 5.4. For more clarity on why the extreme points of the grid need to be adjusted when the solution search space is not adequately described, refer back to Figure 5.1 (specifically the x-axis) that depicts the piecewise linear splines that are used to approximate the log-likelihood and are described by (5.8) - (5.10). The ALRSF makes use of a grid with range $[-\Gamma, \Gamma]$, where the grid points are spaces such that the size of the intervals are given by

$$\frac{|2\Gamma|}{K}, \tag{5.14}$$

where K is the number of piecewise linear splines that are used to approximate the non-linear log-likelihood. The extreme points of the grid are problem-specific and can be obtained in two ways. Firstly, the extreme points can be calculated manually by solving an LP representation of the MIP presented in Section 5.1, which excludes all cardinality selection constraints (as explained in Chapter 6 of Venter (2020)). Alternatively, a novel automated approach can be used where the algorithm determines whether

$$\beta^T X_i \leq -\Gamma, \ \ \forall i = 1, 2, ..., n, \tag{5.15}$$

or

$$\beta^T X_i \geq \Gamma, \ \ \forall i = 1, 2, ..., n, \tag{5.16}$$

in which case the extreme points $-\Gamma$ and Γ are adjusted in both directions by a constant value ψ to allow for dynamic alteration of the search space as necessitated by the algorithm. This will guarantee that a sufficiently large solution search space is maintained without the need for any user intervention. Note that a trade-off exists between execution time and the increase in the size of the search that is enforced by ψ. A small value for ψ ensures a tight solution space but increases the number of iterations and execution time required to obtain a solution.

Similarly, the size of K directly influences the memory requirement of the solution algorithm and the approximation accuracy of the linearised log-likelihood objective function. As noted in Venter (2020), a small value for K will bring about a more severe underestimation of the log-likelihood function. In contrast, a large value for K will lead to an increase in the number of algorithmic iterations required, execution time and computational memory needed by the solver to obtain a solution. Parameters K and ψ should therefore be fine-tuned to find a balance between accuracy, memory usage and execution time. Let ζ be a binary indicator that is used to indicate whether the solution search space sufficiently describes the characteristics of the problem instance. If the grid points

satisfy the conditions as set out by (5.15) and (5.16), the performance measures and regression parameters are recorded for iteration t and the algorithm can proceed to iteration $t+1$. Suppose the contrary is true and the grid does not adequately describe the problem search space according to (5.15) and (5.16). In such instance, the recording of the relevant performance measures and data products for iteration t is delayed while the grid points are modified by a constant value ψ, after which iteration t is re-executed. The adjustment of the extreme points of the grid is described by the second for-loop in lines 10 to 38 in Algorithm 5.3. By not adjusting the range of the grid in a dynamic fashion (by increasing the extreme points) we run the risk of obtaining untrustworthy results from the cardinality selection step. In a scenario where the grid range is extremely small, we may find that some of the variables cannot contribute to the model's predictive performance as their parameter estimates cannot be adjusted due to the constrained solution search space (which is constrained by the extreme points of the grid). As a result, the true predictive power of some variables cannot be adequately assessed and might be completely overlooked by the cardinality selection step. This is why it is paramount for the algorithm to incorporate the added functionality of being able to adjust the range of the grid to ensure that the entire search space is evaluated, even if the ALRSF starts off with a relatively restricted grid range (that is constrained to be quite tight in an attempt to reduce execution time).

As stated previously, the MIP from Section 5.2.1 is solved with a controlled or fixed set of regression parameter estimates contained within vector β' (as shown in Algorithm 5.2) at each iteration of Algorithm 5.3. For the cardinality selection step ($S = 0$) the variables are added to the model in a sequential fashion based on their individual IVs, where the variables are ranked from most predictive to least predictive. This sequential addition of variables to the MIP is captured by lines 5 to 12 in Algorithm 5.2. The IV analysis considers variables in isolation and does not account for variable interactions. Ultimately, it allows the MIP to assess the predictive contribution of each independent variable to the final model (based on some performance measure, such as the AUC), which, in turn, will be used to determine the cardinality parameter c (that governs the maximum number of variables that are allowed in the final model). In other words, the MIP proposed in (5.7) - (5.13) is solved at each iteration t while only considering a subset of the independent variables at iteration t. The cardinality step iteratively evaluates the performance of the model on the test set by comparing the performance measure P_t (which is the AUC at iteration t for the purpose of this thesis) in the current iteration with the best performance measure P_b found so far. For the remainder of this thesis, the reader will note that the cardinality parameter c is incremented by one such that $c_t = c_{t-1} + 1$ when $P_t \geq P_b$. Furthermore, when $P_t \geq P_b$, P_b is also updated with the value of the latest performance measure, after which the iterative process continues to determine whether additional variables should be added to the final subset by assessing subsequent values of P_t. In the case where $P_t \leq P_b$, the cardinality set remains unchanged and the next independent variable can be considered. This iterative process is followed until either the stopping criteria is reached or the total number of iterations approach $t = T$. Alternative cardinality selection schemes can also be considered, e.g. only updating c when $P_t \geq (P_b + P_b 0.1)$. When this is done, the ALRSF will tend to select more

parsimonious models with a slight reduction in model performance [5]. The stopping criteria adopted in the solution approach presented in Algorithm 5.3 is informed by the consecutive number of iterations for which the best-recorded performance measure P_b has not changed. This is governed by a consecutive performance increase threshold denoted by U, while the consecutive number of iterations for which no improvement in model performance is recorded is denoted by u. The consecutive performance threshold U is an adjustable hyper-parameter that is specified such that $U = 10$ for the purpose of this thesis, which means that the ALRSF will assess the AUC of a model that includes between $q + 1$ and $q + 10$ variables if the model in the current iteration includes q variables. Consider two scenarios for u and U, where

1. the algorithm will continue to sequentially add variables to the solution framework, one by one, while $u < U$, or
2. the cardinality step $S = 0$ will terminate and the solution search space pruning step $S = 1$ commences when $u \geq U$.

It is essential to note that Algorithm 5.3 does not have a for-loop that describes each iteration t. This is because Algorithm 5.1 is used as the driving algorithm that either initialises or terminates a specific algorithm based on the value of t and S. In other words, Algorithm 5.3 is invoked by 5.1 as and when required. This iterative switching between Algorithms 5.1, 5.2 and 5.3 continues until either $t = T$ or $u \geq U$. When the termination criteria associated with the cardinality selection algorithm (Algorithm 5.3) is met, the step counter S can be incremented by one such that $S = 1$. Similarly, the iteration counter t is reset to a value of $t = 1$ (considering the logic defined in lines 32 to 41 in Algorithm 5.3). The ALRSF can then return to Algorithm 5.1 based on line 42 in Algorithm 5.3. This concludes the cardinality selection step (the blue block in Figure 5.4).

The next step is the search space pruning step which is depicted by the green block in Figure 5.4. The main goal of this step is to prune the solution search space after the cardinality value c has been identified before an attempt is made to find the optimal solution. This step improves the original MIP by dividing it into manageable sub-problems, where each sub-problem prunes the search space as new solutions that are closer to optimality are discovered. With $S = 1$ and $t = 1$, the ALRSF will first enter Algorithm 5.2 to identify the relevant regression parameter fixing logic that needs to be enforced. The appropriate logic for $S = 1$ is described by lines 13 to 20 in Algorithm 5.2. In general, for $S = 1$, the total number of independent variables present at any given iteration should adhere to the constraint in (5.17) - line 13 to 20 of Algorithm 5.2 ensures that this is indeed the case. When line 25 of Algorithm 5.2 triggers a return to Algorithm 5.1, with $S = 1$ and $t = 1$, the if-statement in lines 6 and 7 of Algorithm 5.1 will be triggered. This means that the search space pruning step in Algorithm 5.4 will be initiated.

Algorithm 5.4 is almost identical to Algorithm 5.3, with the only difference relating to

[5] The influence of various cardinality selection criteria on the ALRSF's generated model output may be studied as part of future research endeavours.

the number of independent variables considered at each iteration and the inclusion of the cardinality selection constraints (5.12) and (5.13). The cardinality parameter c identified by Algorithm 5.3 is used as input in Algorithm 5.4, as seen in the second line of Algorithm 5.4. The idea of the search space pruning step is to sequentially add one independent variable from the set of available variables (based on IV rank order) to the set of independent variables that make up the current cardinality set (controlled by the parameter value c) in an attempt to improve the current best solution. Each iteration in Algorithm 5.4, therefore, tests the assumption that the current cardinality set is the best set of independent variables. During the search space pruning step $S = 1$, the regression parameter fixing heuristic ensures that a sub-problem of the MIP (in Section 5.2.1) is solved such that

$$\sum_{j \in p} z_j \leq (c + 1) + 1. \tag{5.17}$$

The main goal of the search space pruning step is to iteratively reduce the solution search space by testing various variable combinations that might improve the objective function value discovered as part of the cardinality selection step. Each solution obtained during iteration t is used as a warm-start in the subsequent iteration $t + 1$ to ensure that any progress made up to the most recent iteration was not in vain. At each iteration of the search space pruning step, a sub-problem that adheres to the constraint in (5.17) is solved and one of the following actions is taken with respect to the new independent variable being evaluated:

1. the variable is added to the cardinality set if the number of non-zero regression parameters is less than c and the objective function value is improved,
2. the variable replaces one of the variables in the current cardinality set if the objective function is improved and the number of allowable non-zero regression parameters is equal to c, or
3. the variable is rejected and not considered for inclusion in the cardinality set if the objective function does not improve.

This approach, in conjunction with the use of Cplex's Benders decomposition algorithm, significantly reduces the memory requirement of the ALRSF in comparison to the standard branch-and-bound solution approach proposed by Venter (2020). In general, the solution search space pruning step only takes one pass through all of the p variables. Executing multiple passes across all the variables would not add any value to the ALRSF. When the first pass is completed, the search space pruning step $S = 1$ concludes and the optimality step $S = 2$ is initiated. By iteratively evaluating different variable combinations, the solution search space can be pruned and the cardinality set updated before all independent variables are finally allowed to enter the MIP formulation at step $S = 2$. It is important to note that the third line in Algorithm 5.2 sets $t = T - 1$ since the optimality step is not iterative in nature and only one iteration is required to produce an optimality gap (this is because the last iteration considers the solution search space as a whole).

Algorithm 5.4 - Search space pruning step - $S = 1$

$\zeta = 0$
Cardinality parameter value c is generated by Algorithm 5.3.
for $j = 1$ to p' **do**
  $\beta'_j = 0$ if $VIF_j > L$
end for
Each regression parameter β'_j associated with j-th variable for which $VIF_j > L$ is set equal to zero.
Solve logistic regression MIP presented in (5.7) - (5.13) using Benders decomposition.
Cardinality constraints (5.12) and (5.13) included in the MIP.
Cardinality is set equal to c.
for $i = 1$ to n **do**
  $\Omega_i = 0$
  $\Omega_i = \Omega_i + \frac{\beta'^T}{H} X_i$
  if $(\Omega_i \leq -\Gamma)$ or $(\Omega_i \geq \Gamma)$ **then**
    $|\Gamma| = |\Gamma| + \psi$
    $\Delta\gamma = \frac{2|\Gamma|}{K}$
    $\zeta = 1$
  end if
end for
if $\zeta = 0$ **then**
  Calculate and record training and test performance measure P_t.
  Record β'^t.
else
  $t = t - 1$
end if
Save MIP solution found at iteration t for use as warm-start during the execution of iteration $t + 1$.
Increment $t = t + 1$.
Return to Algorithm 5.1 - (driver algorithm).

The optimality step $S = 2$ will follow an approach that is similar to the other algorithms that have been presented thus far and is visually depicted by the yellow block in Figure 5.4. The first part of the optimality step is to identify the appropriate code block in Algorithm 5.2 which relates to the step counter $S = 2$. Once the relevant regression parameter fixing heuristic code block has been identified, Algorithm 5.5 will be executed. As part of $S = 2$ in Algorithm 5.2, all regression parameters are set free such that $\beta' \in \mathbb{Z}^p$. This ensures that the complete search space is considered, which provides a true measure of optimality for the best subset selection problem. The solution in step $S = 1$ is used as a warm-start to the solution process in $S = 2$. This prevents any progress that was made during the search space pruning step from being discarded and reduces the overall memory requirement of the optimality step. The optimality step $S = 2$ will only terminate when optimality is guaranteed, an out-of-memory exception occurs or a predefined termination criterion is

met (e.g. a limit on the execution time). The optimality step allows a gap percentage to be calculated and aims to improve the best integer solution discovered in $S = 1$. All regression parameters are considered (set free) during the optimality step as per the logic presented in Algorithm 5.2. This ensures that the gap percentage is a true reflection of the distance between the best integer solution and the optimal solution.

Algorithm 5.5 - Optimality step - $S = 2$

1: Cardinality parameter value c is generated by Algorithm 5.3.
2: **for** $j = 1$ to p' **do**
3: $\beta'_j = 0$ if $VIF_j > L$
4: **end for**
5: Each regression parameter β'_j associated with the j-th variable for which $VIF_j > L$ is set equal to zero.
6: Use last MIP solution found in iteration t as warm-start to step $S = 2$.
7: Allow remaining regression parameters to be non-zero.
8: Solve logistic regression MIP presented in (5.7) - (5.13) using Benders decomposition.
9: Record final regression parameters, performance measures and optimality gap.
10: $S = S + 1$
11: Return to Algorithm 5.1 - (driver algorithm).

As a final consideration the proposed ALRSF provides a unique measure of analytical flexibility. The performance measures P_t and the influence of each independent variable on the MIP solution at each iteration t are available for tracking, evaluation, and analysis. Even though the solution algorithm allows one to calculate an optimality gap at step $S = 2$ the user can analyse the selected performance measure at each iteration t and terminate the algorithm based on predefined criteria if so inclined. The ability to iteratively analyse the results of the ALRSF at each iteration allows the model developer to obtain insights with respect to the adequacy of the size of the training set, the performance of the model, early detection of overfitting and the impact of various independent variables that would otherwise have been impossible if the standard branch-and-bound algorithm was applied. The last step, according to Algorithm 5.1, is the Lagrangian-based bound tightening step (denoted by $S = 3$). This step will, however, be discussed in detail as part of Sections 5.2.4 and 5.2.5.

5.2.3 ALRSF dummy example (Steps 0 - 2)

The iterative solution algorithm contained within the ALRSF will be applied to a small 6-variable dummy problem to clarify each of the three steps' contribution to the proposed solution framework. This can be used as supplementary material towards the understanding of the pseudo-code presented in Algorithms 5.1 to 5.5. Note that the dummy example (considered as part of this section) is a hypothetical example. The ALRSF will only be verified and validated against real-world datasets in Chapters 6 and 7.

Table 5.2: Novel ALRSF - Dummy Example

Iterations	β'_1	β'_2	β'_3	β'_4	β'_5	β'_6	Test AUC	c	Gap %
Cardinality Step - (S = 0, $c = 0$)									
$t = 1$	Free	Fixed	Fixed	Fixed	Fixed	Fixed	51.34	1	-
$t = 2$	Free	Free	Fixed	Fixed	Fixed	Fixed	65.41	2	-
$t = 3$	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	-
$t = 4$	Free	Free	Free	Free	Fixed	Fixed	87.30	3	-
$t = 5$	Free	Free	Free	Free	Free	Fixed	87.31	3	-
$t = 6$	Free	Free	Free	Free	Free	Free	87.29	3	-
Search Space Pruning Step - (S = 1, $c = 3$)									
$t = 1$	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	-
$t = 2$	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	-
$t = 3$	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	-
$t = 4$	Free	Free	Fixed	Free	Fixed	Fixed	88.50	3	-
$t = 5$	Free	Free	Fixed	Fixed	Free	Fixed	88.65	3	-
$t = 6$	Free	Free	Fixed	Fixed	Free	Fixed	88.65	3	-
Optimality Step - (S = 2, $\forall\beta \in \mathbb{R}^p$, $c = 3$)									
$t = 1$	Free	Free	Free	Free	Free	Free	88.65	3	42.56

When critically evaluating the first six iterations (cardinality selection step) of Table 5.2, it can be observed that as the iterations progress from $t = 1$ to $t = 6$, the number of free regression parameters is equal to the value of t at any given point in time. For instance, at iteration $t = 1$, only the first variable's regression parameter is allowed to take on any integer value $\beta' \in \mathbb{Z}$. In contrast, the remaining regression parameters are fixed to zero. Similarly, two regression parameters are set free at iteration $t = 2$. It should be intuitive to the reader that at each subsequent iteration $t + 1$, an additional independent variable is included in the MIP for testing until all the variables have been considered or the best-recorded performance measure P_b does not improve for U consecutive iterations. The independent variables are added sequentially based on the rank order of the variables' IVs. With regards to Table 5.2, it can be assumed that the first variable has the largest IV while the sixth variable has the lowest IV. As such, independent variables will be added to the cardinality selection step from left to right in Table 5.2 (X_1 to X_6). In other words, the first variable is expected to contribute the most to the model's predictive power, while the sixth variable is assumed to contribute the least. Variables with higher IVs are more likely to enter and remain in the final set of independent variables (constrained by the cardinality parameter c) than variables with smaller IVs. Furthermore, it can be postulated that the addition of more independent variables to the model does not significantly improve the predictive prowess of the model if the test AUC starts to reach a plateau. It should be noted, however, that when the training set is not a statistically representative sample of the population, the test AUC may deteriorate as new independent variables are added. This is a clear indication of overfitting on the training set. With this in mind, consider

that the cardinality selection step attempts to identify parsimonious models that generalise well on unseen data.

At each iteration $t = 1$ to $t = 6$, the MIP from Section 5.2.1 is solved using the Cplex native Benders decomposition algorithm. During each algorithmic step we are able to calculate some performance measure from the training and test set, respectively, for the model that is obtained. Both P_t (the AUC at iteration t) and P_b (the best AUC recorded up until iteration t) are initialised as zero when the ALRSF starts and are updated at each iteration t for $t = 1, ..., 6$. The best-recorded performance measure P_b is updated based on the selected cardinality selection criteria. The cardinality selection criteria (as part of this hypothetical example) was defined such that the cardinality parameter c will only be incremented by one when $P_t > P_b$. With reference to Table 5.2, note that a test AUC of 51.34% is found at iteration $t = 1$. This is greater than the initial test AUC $P_0 = 0$ and, therefore, the cardinality parameter c can be incremented by one such that $c = 1$. Similarly, since $P_0 > P_b$, P_b is updated by setting $P_b = P_1$. At iteration $t = 2$, the second variable enters the MIP based on its IV rank order. The goal of iteration $t = 2$ is to solve a MIP problem with six independent variables while the regression parameters of the first two variables are allowed to vary throughout the full range of $\beta' \in \mathbb{Z}$ and the remaining regression parameters set equal to zero. The test AUC (denoted by P_2) found at iteration $t = 2$ was 65.41%, which is larger than P_1 found at iteration $t = 1$ and suggests that the cardinality parameter c can be updated such that $c = 2$. The same logic is applied to subsequent iterations $t = 3$ to $t = 6$. Observe that from iteration $t = 3$ onwards, the cardinality parameter value remains constant. This is due to AUC values at iterations $t = 4$ to $t = 6$ that are found to be smaller than the best-recorded performance measure P_b that was found at $t = 3$. The cardinality step $S = 0$ has therefore concluded that a model with a cardinality $c = 3$ would be considered appropriate for this specific logistic regression problem instance, which implies that the step counter S can be incremented by one such that $S = 1$. When the cardinality parameter value remains constant for U iterations or when all the independent variables have been considered, the cardinality step is terminated and the search space pruning step is initiated. For the example in Table 5.2, a value of $U = 5$ was used. Since the cardinality set remained constant for four iterations (between $t = 3$ and $t = 6$) and the threshold $U = 5$ was not exceeded, note that all possible independent variables were considered for inclusion in the final model before the cardinality step $S = 0$ concluded.

The second set of iterations for $S = 1$ constitute the search space pruning step. Step $S = 1$ solve the MIP from Section 5.2.1 and includes the cardinality constraints from (5.12) and (5.13) since c has been identified in the previous step $S = 0$. Including the cardinality constraints results in an NP-hard problem. A regression parameter fixing heuristic is employed to iteratively prune the initial MIP solution search space to find high-quality solutions to the NP-hard problem. This approach reduces the large complex MIP into smaller and more manageable sub-problems. Table 5.2 shows that, at most, three independent variables are selected from a set of $c + 1$ variables. This relates to the constraint in (5.17) which defines the size of the sub-problems considered at each

iteration of the search space pruning step. Given that a cardinality parameter $c = 3$ was identified during step $S = 0$, note that the final number of variables included in the final model (in all subsequent steps of the ALRSF) cannot exceed $c = 3$. As part of the search space pruning step, the first three variables were identified as the most predictive inputs for the model. This is why the first three variables are set free at iterations $t = 1$ to $t = 3$. Similar to the cardinality selection step, one variable at a time is added to the search space pruning step (starting from the first variable) based on the ranking of its IV. Since the first three variables are already part of the cardinality set (constrained by the parameter c) identified in step $S = 0$, the first three iterations of step $S = 1$ will be nearly identical to those in step $S = 0$. A change is only observed at iteration $t = 4$ of $S = 1$ when the fourth variable is considered for possible inclusion in the cardinality set. During iteration $t = 4$ we observe that the combination of the first, second and fourth variables produces a model that performs better than a model that includes the first three variables. Essentially, this means that the fourth independent variable supplanted the third independent variable from the cardinality set. The search space pruning step therefore challenges the assumption that the variables that are currently in the cardinality set produce the best-performing model by iteratively testing other variable combinations (one by one based on their IV rank order) until an improvement (or no improvement) in model performance is recorded. At iterations $t = 5$ and $t = 6$, the cardinality set is once again updated, as the test performance is improved by considering alternative variable combinations. Table 5.2 shows that different 3-variable combinations have a noticeable impact on the model's performance. By iteratively testing variable combinations and updating the cardinality set based on improvements observed in the model's generalisation performance, solutions that are better than those provided by the standard branch-and-bound MIP proposed by Venter (2020) can be found. When all of the independent variables have been considered at least once, the search space pruning step (Algorithm 5.4) will conclude and the optimality step $S = 2$ can be initiated. No significant improvement in the objective function will be realised by implementing a second pass through the set of independent variables.

At step $S = 2$, we arrive at the conclusion of our example, where the optimality step of ALRSF is now invoked. The solution discovered at iteration $t = 6$ of the search space pruning step $S = 1$ is used as a warm-start in step $S = 2$. All independent decision variable regression parameters are allowed to take any value such that $\beta' \in \mathbb{Z}^p$. An attempt is then made (while utilising the Cplex native Benders decomposition algorithm) to further improve the best integer solution discovered in Step $S = 1$. Step $S = 2$ also allows the calculation of an optimality gap for the problem instance, given that the remaining search space is considered. It is important to note that the optimality gap in Table 5.2 is only hypothetical and not underpinned by any upper or lower bound values. Its only function is to inform the reader about the optimality step's ability to provide an optimality gap as part of the ALRSF logic. The solution framework will only terminate when optimality has been proven, available system memory is exceeded or a predefined stopping criteria is satisfied (such as time limit or number of integer solutions discovered). The gap from Table 5.2 indicates that the relative difference between the objective function of the best

integer solution that was found by the MIP (before the stopping criteria was reached) and the objective function of the MIP that has no integer constraints is 42.56%.

5.2.4 Proposed Lagrangian Reformulation (Step 3)

The final step of the proposed ALRSF is focused on tightening the upper bound of the maximisation problem, as discussed in Sections 5.1 and 5.2. The upper bound for a maximisation problem is determined by solving an LP relaxation of the MIP, after which the solver will attempt to tighten the bounds during the execution of the branch-and-bound algorithm. The optimality gap is then calculated by evaluating the distance between the best integer solution and the upper bound. Lagrangian relaxation is a decomposition technique that allows the relaxation of complicating constraints into the objective function (accompanied by Lagrangian multipliers that are denoted by ϕ), as formalised in Chapter 4. The relaxed form of the original MIP is much easier to solve and provides a unique bounding property that can be exploited to obtain bounds that are much tighter when compared to those produced by an LP relaxation.

The structure of the MIP that was proposed in Section 5.2.1 naturally lends itself to being decomposed into two separate problems, which include a straightforward logistic regression problem and a best subset selection problem. The former is easier to solve without the latter being considered. The separable nature of the best subset selection MIP makes it an ideal candidate to which Lagrangian relaxation can be applied (Zhao et al., 1999). The cardinality selection constraints in (5.12) and (5.13) can be seen as being complicating constraints because they add a binary selection decision variable z_j to the MIP formulation, which significantly increases the solution search space size. According to Lopez et al. (2004) and Gheyas and Smith (2009), the complexity of the best subset selection problem increases at a rate of 2^p as the number of independent variables p increases. The logistic regression MIP can be solved within a few seconds when excluding the cardinality selection constraints, whereas the inclusion of these constraints significantly increases the execution time of the model. Recall that the ALRSF MIP uses an integer conversion approach. As part of the Lagrangian relaxation reformulation, the integer conversion transformation is excluded and the regression parameters are allowed to take on any value such that $\beta \in \mathbb{R}^p$. It will be empirically proven (in Chapter 6) that the objective function of the proposed MIP in Section 5.2.1 and the original MIP in Section 5.1 will have the same optimal value, with only a slight difference in the decimal precision associated with the final regression parameters. Omitting the integer conversion transformation from the Lagrangian reformulation will therefore not have a significant effect on the final model that is obtained in the end.

Note that the VIF exclusion constraint in (5.11) is still included in the Lagrangian relaxation reformulation so as to exclude highly correlated variables, which improves the solution algorithm's numerical stability. Excluding specific independent variables from the solution search space will undoubtedly influence the final objective function value. Therefore, to ensure consistency between the ALRSF and the Lagrangian relaxation ap-

proach, only variables with a $VIF \leq L$ are allowed to participate in the solution process. The excluded regression parameters $\beta \in \mathbb{R} \ | \forall j \in p'$ along with their associated Lagrangian multipliers $\phi_j^{2t} \ | \forall j \in p'$ and $\phi_j^{3t} \ | \forall j \in p'$ are, therefore, set equal to zero at the start of the solution algorithm. With this in mind, a Lagrangian relaxation is proposed where the objective is to

$$\begin{aligned} \max_{\beta,z,\Lambda} \ f_{\text{ALRSF}_{LR}} = \sum_{i \in n} \sum_{k \in K} (\gamma_k Y_i - \log[1 + \exp(\gamma_k)]) \Lambda_{ik} - \phi^{1t} (\sum_{j \in p} z_j - c - 1) \\ - \sum_{j \in p} \phi_j^{2t} (\beta_j - M_j z_j) + \sum_{j \in p} \phi_j^{3t} (\beta_j + M_j z_j), \end{aligned} \tag{5.18}$$

subject to:

$$\sum_{k \in K} \Lambda_{ik} \gamma_k = \beta^T X_i \ \ \forall i \in n, \tag{5.19}$$

$$\sum_{k \in K} \Lambda_{ik} = 1 \ \ \forall i \in n, \tag{5.20}$$

$$\beta_j = 0 \ \ \forall j \in p', \tag{5.21}$$

$$0 \leq \Lambda_{ik} \leq 1 \ \ \forall k \in K; \forall i \in n, \tag{5.22}$$

where ϕ^{1t}, ϕ_j^{2t} and ϕ_j^{3t} denote the Lagrangian multipliers associated with the relaxed constraints in (5.12) and (5.13). Let the superscript t denote the t-th iteration of Algorithms 5.6 and 5.7, which will be presented shortly. The multipliers are employed to penalise constraint violation and are only allowed to take on values such that $\phi \geq 0$. The objective function value $f_{\text{ALRSF}_{LR}}$ represents an upper bound to the original problem f_{ALRSF} such that any solution will have the property $f_{\text{ALRSF}_{LR}} \geq f_{\text{ALRSF}}$. The optimal solution of the proposed relaxation, denoted by $f^*_{\text{ALRSF}_{LR}}$, will therefore be an upper-bound to the optimal solution of the original logistic regression MIP that was presented in Section 5.2.1 (Fisher, 1985). In order to find the tightest possible upper bound, we need to solve an optimisation problem of the form

$$\min_{\phi} f_{\text{ALRSF}_{LR}}, \tag{5.23}$$

which can be referred to as the Lagrangian dual of the primal optimisation problem (Lemaréchal, 2001). Various algorithms can be used to find the most appropriate values for ϕ that will produce an optimal solution for the Lagrangian dual. One of these solution algorithms is the sub-gradient algorithm. The sub-gradient algorithm is used to determine the value for ϕ^t at each iteration t such that the objective function $f_{\text{ALRSF}_{LR}}$ converges towards the optimal solution $f^*_{\text{ALRSF}_{LR}}$. Note that it is possible for the sub-gradient δ^t at iteration t to not follow a direction of descent for $f_{\text{ALRSF}_{LR}}$, which means that the upper bound that is found by solving the Lagrangian relaxation problem may not necessarily decrease in a monotonic fashion. It is therefore necessary to track the best upper bound for the objective function throughout all iterations T, which is denoted by ${f_{\text{ALRSF}_{LR}}}^t$. A modified version of the well-known sub-gradient algorithm is proposed in Section 5.2.5 (which accommodates the presence of Big-M values in the constraint in (5.12)) to solve the Lagrangian dual problem.

5.2.5 Proposed sub-gradient solution algorithm (Step 3)

Let t denote the individual algorithmic iterations of the solution algorithm presented in Algorithm 5.6 and Algorithm 5.7 while T denotes the total number of possible algorithmic iterations. Ideally, T should be sufficiently large to ensure convergence of the sub-gradient algorithm.

Algorithm 5.6 Upper Bound Tightening Step - $S = 3$

$t = 1$
$\phi^{1t} = 0.1$
$\phi_j^{2t}, \phi_j^{3t} = 0 \mid \forall j \in p'$
$\phi_j^{2t}, \phi_j^{3t} = 0.1 \mid \forall j \notin p'$
$\alpha = 2$
$M = \infty$
$\theta = 1$
$u = 0$
$\epsilon = 1 \cdot 10^{-10}$
for $t \leftarrow 1$ to T **do**
  for $j \leftarrow 1$ to p **do**
    $M_j = M$
  end for
  Solve MIP in (5.18) - (5.23) with ${M_j}^t \leftarrow \infty$.
  Use ${\beta_j}^t$ that is obtained from solving the MIP in Section 5.2.4.
  if $({M_j}^t - |{\beta_j}^t|) > D$ **then**
    for $j \leftarrow 1$ to p **do**
      ${M_j}^t = |{\beta_j}^t| + D$
    end for
  end if
  Solve MIP in (5.18) - (5.23) with tightened Big-M Values (${M_j}^t$).
  Record objective function value ${f_{\text{ALRSF}_{LR}}}^t$.
  if $f^t_{\text{ALRSF}_{LR}} < f^b_{\text{ALRSF}_{LR}}$ **then**
    $f^b_{\text{ALRSF}_{LR}} = f^t_{\text{ALRSF}_{LR}}$
  end if
  Execute Algorithm 5.7 which estimates new Lagrangian Multipliers.
end for
Terminate algorithm

Algorithm 5.7 Lagrangian Multipliers - (Sub-gradient solution algorithm)

1: Use solution obtained from Algorithm 5.6.
2: $\delta^{1^t} = (\sum_{j \in p} {z_j}^t - c - 1)^2$
3: $\delta_j^{2^t} = (\beta_j - {M_j}^t {z_j}^t)^2 \ \ \forall j \in p$
4: $\delta_j^{3^t} = (\beta_j + {M_j}^t {z_j}^t)^2 \ \ \forall j \in p$
5: $\theta^t = \frac{\alpha(UB^t - LB^t)}{\delta^{1^t} + (\sum_{j \in p} \delta_j^{2^t}) + (\sum_{j \in p} \delta_j^{3^t})}$
6: $\phi^{1^{t+1}} = \max(0, \phi^{1^t} + \theta^t \delta^{1^t})$
7: $\phi_j^{2^{t+1}} = \max(0, \phi_j^{2^t} + \theta^t \delta_j^{2^t}) \ \ \forall j \in p$
8: $\phi_j^{3^{t+1}} = \max(0, \phi_j^{3^t} - \theta^t \delta_j^{3^t}) \ \ \forall j \in p$
9: **if** $(f^b_{\text{ALRSF}_{LR}} \leq f^t_{\text{ALRSF}_{LR}})$ **then**
10: $u = u + 1$
11: **else**
12: $u = 0$
13: **end if**
14: **if** $(u \geq U)$ **then**
15: $\alpha = \frac{\alpha}{2}$
16: **end if**
17: **if** $(\alpha <= \epsilon)$ **then**
18: Terminate Algorithm 5.6 and Algorithm 5.7.
19: **end if**
20: Return to Algorithm 5.6 with updated Lagrangian multipliers for use as part of iteration $t + 1$.

The Lagrangian multipliers associated with the relaxed form of the constraints in (5.12) and (5.13) are initialised as arbitrary values that are greater than zero (from the second and fourth lines in Algorithm 5.6). The Big-M constant M, as defined in Section 5.1, is initially set equal to a sufficiently large value in an attempt to determine what the most appropriate value for M_j should be (for a specific set of multipliers ϕ^t) to ensure a tight solution search space is maintained throughout each iteration. After solving the Lagrangian relaxation MIP in (5.18) - (5.23) with a large value for M_j, the Big-M value for each independent variable is adjusted by lines 16 to 20 of Algorithm 5.6 to satisfy $M_j = |\beta_j| + D$, where D is an arbitrarily small value [6]. Observe that a slightly different notation is used for the Big-M bounds such that we now refer to $M_j \forall j \in p$ instead of M, as M is no longer considered to be the same universal value for all of the independent variables (as in Sections 5.1 and 5.2). The absolute value of each regression parameter estimate $|\hat{\beta}_j|$ is now bounded by the range $[0, M_j]$, which is determined by lines 11 to 20 in the pseudo-code of Algorithm 5.6. By forcing D to be relatively small, the true influence of each regression parameter estimate on the Lagrangian objective function can easily be ascertained. In contrast, a large value for D will result in an inflated number of sub-gradient iterations that are required to ensure convergence. After determining the most appropriate Big-M values (denoted by M_j^t) to use for each regression parameter estimate at iteration t, the Lagrangian MIP is solved using the standard branch-and-

[6] Note that D in Chapter 5 has an alternative definition to D in Chapter 4.

bound algorithm. The value for ${f_{\text{ALRSF}_{LR}}}^t$ is recorded and compared with the best upper bound found (denoted by $f^b_{\text{ALRSF}_{LR}}$) prior to iteration t. If $f^t_{\text{ALRSF}_{LR}}$ is smaller than $f^b_{\text{ALRSF}_{LR}}$, then $f^b_{\text{ALRSF}_{LR}}$ is updated. This check-and-update logic can be seen in lines 23 to 25 of Algorithm 5.6. The next step is to determine the most appropriate values for the Lagrangian multipliers at iteration $t+1$, denoted by ${\phi_j}^{t+1}$. This can be done by using Algorithm 5.7, which describes the well-known sub-gradient algorithm. Observe that Algorithm 5.7 is called by line 26 of Algorithm 5.6 (as needed) when an updated set of Lagrangian multipliers is required for iteration $t+1$.

The sub-gradient algorithm is generally made up of two distinct steps. Firstly, the gradients δ^{1t}_j, δ^{2t}_j and δ^{3t} associated with each relaxed term in (5.12) and (5.13) is determined based on the final solution of the MIP in iteration t, ${f_{\text{ALRSF}_{LR}}}^t$, which is obtained via Algorithm 5.6. Lines 2 to 4 of Algorithm 5.7 present the logic that is used to calculate these sub-gradients. Secondly, the Polyak step size is calculated as stipulated in line 5 of Algorithm 5.7. The Polyak step size is an adaptive step size which allows accelerated convergence based on the smoothness and local curvature of the optimisation landscape (Hazan and Kakade, 2019). It dynamically decreases or increases based on the distance between the upper bound (UB) and the lower bound (LB). The best integer solution that was obtained in Step $S = 1$ (the second step) of the ALRSF will be used as the lower bound for the sub-gradient solution algorithm. At each iteration, a step is taken in the opposite direction of sub-gradients $\delta^{1t}, \delta^{2t}_j$ and δ^{3t}_j in an attempt to minimise the Lagrangian dual [7]. It is important to note that the sign convention of the multiplier update rule $\phi^{t+1} = \max(0, \phi^t \pm \theta\delta)$ is dependant on the nature of the constraint (inequality or equality constraints) that are relaxed into the objective function (5.18). The relaxed form of the constraints in (5.12) and (5.13) can be defined as

$$\phi^{1t}(\sum_{j\in p} z_j - c - 1) \leq 0, \tag{5.24}$$

$$\sum_{j\in p} \phi^{2t}_j(\beta_j - M_j z_j) \leq 0, \tag{5.25}$$

$$\sum_{j\in p} \phi^{3t}_j(\beta_j + M_j z_j) \geq 0. \tag{5.26}$$

We now consider the logic behind the relaxed form of the constraint in (5.13) to gain a better understanding of the mechanism that is used by the Lagrangian relaxation approach towards limiting constraint violations. When $\sum_{j\in p} z_j - c - 1$ satisfies (5.24), the magnitude of the penalty enforced by ϕ^{1t} should be reduced. The opposite is true when (5.24) is violated. By increasing ϕ^{1t} when $\sum_{j\in p} z_j - c - 1 \geq 0$, a penalty is rightfully enforced on the

[7] The sign convention of the sub-gradients depends on how the relaxed constraints are formulated, e.g. the constraint in (5.24) can alternatively be defined as $\phi^{1t}(c+1-\sum_{j\in p} z_j) \leq 0$, which would result in a sub-gradient with an opposite sign convention to (5.24). In this case, a step in the direction of the sub-gradient would be appropriate (opposed to taking a step in the opposite direction). The change in sign convention should also be reflected as part of the equation in (5.18) to ensure that the relaxed constraints encourage feasibility while penalising constraint violations. This relates to changing the "-" sign associated with the second term of (5.18) to a "+" sign and will produce an identical result. Even though the formulation differences were explained at the hand of (5.24), similar changes can be applied to (5.25) and (5.26).

constraint in (5.24). Clearly, when (5.24) is violated (resulting in a positive sub-gradient) and the Lagrangian multiplier ϕ^{1^t} is increased, an adverse effect on the final objective function value will be observed if none of the decision variables take on alternative values. The negative sign that precedes the second term of the objective function in (5.18), a positive sub-gradient associated with $\sum_{j \in p} z_j - c - 1$ and an increase in $\phi^{1^t} \geq 0$ will result in a larger negative value being added to the final objective function. As a result, the MIP solver will attempt to maximise (5.18) while also attempting to reduce the magnitude by which (5.24) is violated (as a direct result of increasing the Lagrangian multiplier value associated with ϕ^{1^t}).

In an attempt to maximise (5.18) in the presence of the added penalty term, the total number of variables selected to be active by $\sum_{j \in p} z_j$ will be reduced, which enforces constraint compliance according to (5.24). In other words, the Lagrangian multipliers encourage solution feasibility for the primal optimisation problem. The same logic can be applied to the relaxed Big-M constraints in (5.12), with the only difference relating to a slight change in the sign convention for the greater-than inequality, as defined in (5.26). With more than one constraint relaxed into the objective function, selecting the most appropriate values for $\phi^{1^t}, \phi_j^{2^t}$ and $\phi_j^{3^t}$ are no longer trivial and, as such, the sub-gradient method provides a simplistic algorithm for solving the Lagrangian dual. This iterative solution process continues until α is arbitrarily small, where α is a constant value that is used as part of the Polyak step size given by θ. Let U denote a constant threshold at which α is halved. The threshold U can be specified to be relatively large to ensure that α is only reduced when no improvement or progress is made with respect to the best-recorded upper bound value $f^b_{\text{ALRSF}_{LR}}$ for a total of U consecutive iterations. As part of the empirical experiments conducted in Chapter 7, $U = 10$ was used, whereas α was initialised such that the algorithm starts of with $\alpha = 1$. This means that if the best-recorded upper bound does not improve within 10 algorithmic iterations, α can be halved according to the logic in lines 9 to 16 in Algorithm 5.7. The variable u keeps track of the number of consecutive iterations for which no improvement has been observed. If $f^b_{\text{ALRSF}_{LR}}$ improves within $t < 10$ algorithmic iterations, e.g. eight iterations, u will be once again reset to $u = 0$ to reinitialise the consecutive improvement counting process. Reducing α will directly affect the step size θ and, in turn, force the solution algorithm to converge. Let ϵ denote the threshold for α at which the solution algorithm concludes. If α is sufficiently small (e.g. $\alpha < \epsilon$ as specified in lines 17 to 19 in Algorithm 5.7), the step size will no longer affect the change to the Lagrangian multipliers and the algorithm can be terminated as no further progress will be possible.

In summary, the bound tightening step can be seen as an attempt to finding the most appropriate Lagrangian multipliers (using Algorithm 5.7) that will produce the best upper bound to the original maximisation problem defined in Section 5.2. Furthermore, the proposed bounding algorithm employs a novel Big-M tightening approach to try and reduce the number of algorithmic iterations while also pruning the size of the solution search space. This Lagrangian bounding approach ultimately aims to improve the optimality gap associated with the logistic regression best subset selection problem. The improvements in the optimality gap will be verified as part of Chapter 7.

5.3 Contributions of the proposed ALRSF

The main contributions put forward as part of the proposed ALSRF include the following:

1. The ALRSF provides an automated solution framework that requires minimal user intervention and reduces the specialised skills required to solve best subset selection problems.

2. The ALRSF reduces the amount of computational memory consumed when solving the exact deterministic best subset selection problem.

3. The ALRSF yields higher-quality solutions that can be proven to be closer to optimality by studying the optimality gap that is generated as part of the MIP's output.

4. The size of the problems that can be considered has drastically increased. This further bolsters the framework's capability to be utilised when solving large-sized industry problems.

5. Analytical flexibility is afforded to the model developer due to the iterative nature of the solution framework. It gives oversight of the holistic solution process and allows for early detection of overfitting, provide insights into the interactions between variables and the target and allows for early termination of the solution algorithm by using predefined evaluation criteria.

6. The solution approach blends the advantageous characteristics of both heuristic, deterministic and decomposition-based solution algorithms.

7. Minor transformations to the original MIP formulation (in Section 5.1 of Chapter 5) enable the use of Cplex's native Benders decomposition algorithm instead of the standard branch-and-bound algorithm. The reduction in memory requirement can mainly be attributed to exploiting the unique block structure of the proposed MIP formulation in combination with the use of the proposed search space pruning heuristic that trims the initial solution search space.

8. The best subset selection solution framework of the ALRSF naturally lends itself to applications to various supervised machine learning techniques that are not considered in this thesis, such as linear regression and neural networks. This can be considered as part of future research endeavours.

9. The framework makes use of Lagrangian relaxation in conjunction with a subgradient-based solution approach to tighten the maximisation problem's upper bound while also reducing the optimality gap. A lower optimality gap instils greater confidence in the quality of the solution produced by the framework.

The solution framework provides an automated approach towards solving the non-linear logistic regression best subset selection problem and requires minimal user intervention. In comparison, when considering the approach proposed in Venter (2020), the user should

manually determine the linearised grid points that are bounded by $[-\Gamma, \Gamma]$ by solving a linear programming problem while excluding all cardinality constraints. The ALRSF also incorporates an automated cardinality selection step. The only parameters that need to be specified before executing the automated algorithm include the number of independent variables present in the training set, the number of observations of both the training and test sets, the variable importance ranking of each independent variable (based on IV or some other measure of variable importance) and all variables that exceed the VIF threshold L. Additionally, the user also needs to specify the desired cardinality selection criteria.

Based on the information provided by the user, the algorithm will automatically determine the most appropriate problem-specific cardinality parameter value, iteratively reduce the solution search space (while pruning the solution space) and discover solutions that are closer to optimality. Finally, the algorithm will attempt to prove optimality and calculate an optimality gap which provides the user with a measure of the quality of the final solution. The fact that the framework iteratively reduces the solution search space implies that larger subset selection problems from industries such as finance, engineering, research and development, healthcare and agriculture can now be considered [8]. The ALRSF requires far less computational memory and resources when compared to the straightforward branch-and-bound algorithm, which increases the flexibility of its implementation.

In summary, the proposed solution framework provides a holistic approach towards solving larger-scale MIP logistic regression best subset selection problems. The automated solution framework also opens a whole new avenue to the field of deterministic solution algorithms and can provide a starting point from which numerous automated solution approaches can be developed.

5.4 Summary

This chapter was focused on introducing the reader to the proposed MIP formulation and algorithmic logic behind the novel ALRSF. The solution framework was mainly developed to solve large-sized real-world classification problems and to address some of the key limitations of the standard branch-and-bound solution algorithm that is used to solve the logistic regression MIP proposed by Venter (2020). The main contributions were reiterated and the proposed solution framework's value proposition was justified.

[8] Until recently, only moderately-sized datasets have been considered in literature for most NP-hard problems.

Chapter 6

ALRSF model verification

This chapter will be dedicated to verifying the solution framework presented in Chapter 5. The goal of model verification is to test each of the unique functionalities that make up the proposed solution framework. From the verification experiments, some level of confidence can be obtained as to whether the proposed ALRSF adequately describes the underlying logistic regression problem. Furthermore, the experiments in this chapter seek to determine whether the solutions generated by the framework are of a high quality and can be trusted as being correct and optimal.

Relatively small problem instances were considered for verification experiments as larger training sets are not guaranteed to achieve optimality, thereby making a proper comparative study impossible. Three major aspects of the proposed novel ALRSF will be verified, which include

1. the regression parameter estimates $\hat{\beta}_j$ produced by the final solution,
2. the subset of variables selected (from the total set of p independent variables) for inclusion in the final cardinality set (which is constrained by the parameter c),
3. the size of the grid that is used to linearise the log-likelihood objective function, which is constrained by the extreme points denoted by $-\Gamma$ and Γ.

6.1 Experimental assumptions

The assumptions will apply to both Chapter 6 and 7 and will allow for a comparative analysis between the proposed solution framework (applied to the MIP formulation in Section 5.2 of Chapter 5) and the standard branch-and-bound solution algorithm (the MIP formulation in Section 5.1 of Chapter 5). Any deviations from these assumptions (in subsequent chapters) will be clearly specified. The assumptions are listed below.

1. All experiments, except those applicable to the relative CT slice location classifica-

tion problem, were allowed to run for 345600 seconds. The relative CT slice location classification problem experiments were permitted to run for 691200 seconds. The allowed runtime is directly proportional to the size of the problem instance considered, with emphasis on the number of independent variables. A greater number of independent variables necessitates longer execution times.

2. To ensure that a comparative study is possible, the ALRSF was executed prior to the standard Cplex branch-and-bound algorithm to obtain the problem-specific cardinality parameter value c and the range $[-\Gamma, \Gamma]$ of the grid used to linearise the log-likelihood function. These values were used as input parameters in the standard branch-and-bound algorithm.

3. The number of grid point intervals was specified to be $K = 401$ for all experiments. A detailed study on the influence of K on the solution accuracy and execution time can be found in Venter (2020).

4. When the execution of a specific problem instance terminates (in Cplex) as a result of an out-of-memory exception, the problem instance will be re-executed with a predefined termination criteria (such as a runtime limit). The time limit will be informed by how long the first solution attempt took to produce a result before the out-of-memory exception occurred. This allows the user to extract the final solution obtained (before the available system memory is exceeded) as results cannot be generated when Cplex encounters an out-of-memory exception.

5. The influence of multicollinearity is not a key focus of this study. For all of the experiments considered from here-on out, we use a VIF threshold of $L = 10$ (unless stated otherwise).

6. The delimiter H, which is utilised to transform β' from integer variables to rescaled relational variables β, was selected as 1000 for all experiments in Chapter 6 (unless stated otherwise). The influence of H on model performance and the regression parameter estimates will be discussed in Section 6.4 of Chapter 6. In Section 5.2 of Chapter 5, a moderately-sized value for H was proposed to allow adequate variability in β' on either side of the decimal separator. As such, a value of $H = 100000$ was selected for all experiments that were conducted in Chapter 7.

7. Based on empirical evidence, a value of 10 was selected for the grid point adjustment constant ψ (with reference to the experiments conducted in Chapter 7). A value of $\psi = 10$ allowed for an adequate trade-off between the number of iterations required during the cardinality selection and solution pruning steps while also ensuring a relatively tight solution search space.

8. The consecutive incremental limit U was also set equal to a value of 10. This represents the total number of consecutive iterations for which the performance measure in P_t adheres to $P_t < P_b$ (where $P_t = AUC_t$ with AUC_t the AUC at iteration t), after which the cardinality selection step can be terminated and the search space pruning step can be initiated. A smaller value for U could lead to the selection of more parsimonious models. However, it may exclude some important

independent variables that could add to the predictive power of the final model. It is assumed that variables with low IVs do not necessarily contribute to the performance of the model and, as such, that a moderate value for U (even when the number of independent variables is large) will still produce acceptable results.

9. The cardinality selection criteria were defined in such a way that the cardinality parameter c is only incremented by one when $P_t \geq P_b$ and left unchanged when $P_t < P_b$. Alternative selection criteria are possible and can be defined by the model developer. The model developer should, however, be cognisant of the fact that the selection criteria may influence the final model selected for inference.

10. For the remainder of the thesis the value reported for c represents the number of independent variable c over and above the intercept term. In other words, the intercept term is included in the model by default and as such c will only constrain the number of independent variables (excluding the intercept term) that can be included in the model.

6.2 Computational hardware and software

The novel automated solution framework was developed in C++ and the logistic regression MIP formulations (from Sections 5.1 and 5.2 of Chapter 5) were solved by using the C++ Cplex concert technology API. The training and test datasets were saved as comma-delimited files (csv) and used as input to the C++ solution framework. The framework was implemented on a Ryzen Linux Mint machine which has the following specifications:

1. Processor: AMD Ryzen 5 1600 6 core processor.
2. Operating System: Linux Mint.
3. Installed Memory (RAM): 32 GB.

Data pre-processing, which includes the calculation of the VIF values associated with individual variables, was done by using built-in Python libraries, such as statsmodels. The normalisation of training and test sets and the calculation of variables IVs was done in Excel. The built-in ranking function available in Excel was used to rank the IV values in descending order.

6.3 Verification dataset

The dataset used for verification experiments contains the defaults of credit card clients from Taiwan and was published in 2016 on the UCI machine learning repository. The dataset consists of 24 explanatory variables and 30000 observations. Two sub-problems are derived from the dataset that serve as verification problems, the first of which includes only

six independent variables. The second problem instance consisted of eight independent variables and a training set of 200 observations. Table 6.1 provides more information on the independent variables that were considered.

Table 6.1: Independent variable description

Variable	Variable description
1	Amount of credit extended to the client
2	The client gender
3	Level of education
4	Marital status
5	Client age
6	Repayment status in September 2005
7	Repayment status in August 2005
8	Repayment status in July 2005

The binary target Y_i indicates whether the i-th client defaulted such that $Y_i = 1$ when a default occurred. The target variable event rate (for the training set) is 23% . The ALRSF was first executed to determine the value of the cardinality parameter c and grid range $[-\Gamma, \Gamma]$ for each problem instance under consideration. These values were then used as inputs in the standard branch-and-bound algorithm. This allows for a comparative study to be conducted where the same underlying problem is solved.

6.4 ALRSF regression parameter estimates verification

To prove the validity of the proposed solution framework, a comparison between the output generated by the ALRSF (defined by the MIP in Section 5.2 of Chapter 5) and the standard branch-and-bound algorithm (defined by the MIP in Section 5.1 of Chapter 5) should be made. The two main factors to consider include

1. the independent variables selected for inclusion into the cardinality set,
2. the size (in absolute terms) of the regression parameter estimates associated with each independent variable in the final model.

The final results should indicate whether the results that are generated by the ALRSF are accurate and a good representation of the underlying logistic regression subset selection problem. A conversion constant of $H = 1000$ was utilised. The influence of H on the accuracy and quality of the solutions (generated by the solution framework) was also evaluated, and the results are recorded in Table 6.3.

Table 6.2 provides a summary of the final solutions produced by the ALRSF and the standard branch-and-bound technique. At first glance, we can observe that the objective

function values (which are given in the LogL column of Table 6.2) are identical since both algorithms achieved optimality. This suggests that the solutions of the ALRSF can be trusted and that the algorithmic logic is sound.

For experiments 1 and 2, a cardinality parameter of $c = 4$ was used. Similarly, a value of $c = 3$ was employed for the eight-variable problem instance. Both of the algorithms were allowed to include any of the available independent variables into the cardinality set. Nevertheless, it is evident that both solution algorithms opted for the same variable set (which yielded identical log-likelihood values). When the regression parameter estimates between the two algorithms are compared, we can observe that the standard branch-and-bound algorithm allowed for greater decimal precision for individual regression coefficients. We note, however, that the differences between the parameter sets are negligible. The small difference between the parameter estimates of the two approaches can be attributed to the integer conversion coefficient H.

Table 6.2: Regression parameter estimate comparison

Logic	c	p	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	LogL
ALRSF	4	6	0.655	2.460	0	0	1.536	-0.350	-5.668	-	-	-98.63
B&B	4	6	0.654536	2.46055	0	0	1.53657	-0.34927	-5.66825	-	-	-98.63
ALRSF	3	8	0.466	2.347	0	0	1.691	0	-5.738	0	0	-98.67
B&B	3	8	0.465977	2.34726	0	0	1.69113	0	-5.73871	0	0	-98.67

Table 6.3 presents the results that are obtained when the aforementioned experiments are repeated for the ALRSF, but for varying H values. When considering the results for a larger value of H, it is clear that the decimal precision increases with the same order of magnitude associated with the increase in H. Comparing the results of the ALRSF experiments in Table 6.3 to the branch-and-bound results obtained in Table 6.2, a greater resemblance is observed (with regards to regression parameter decimal precision) when a larger value for H is used. Even though the value of H influences the decimal precision of the regression parameter estimates, the objective function values for both algorithms were identical.

Table 6.3: Influence of varying values for H - ALRSF

H	c	p	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	OBJF
1000	4	6	0.655	2.460	0	0	1.536	-0.350	-5.668	-	-	-98.63
10000	4	6	0.65475	2.46032	0	0	1.53659	-0.34926	-5.66828	-	-	-98.63
1000	3	8	0.466	2.347	0	0	1.691	0	-5.738	0	0	-98.67
10000	3	8	0.46597	2.34726	0	0	1.69113	0	-5.73871	0	0	-98.67

Consideration should also be given to the ability of similar models with slightly different parameter estimates to generalise well on unseen data. Table 6.4 provides an overview of

the AUC performance measures for each experiment. In most cases, the training AUC and test AUC are identical for all experiments, with only one exception: ALRSF models that are based on an eight-variable dataset that utilised a cardinality parameter of $c = 3$ and $H = 1000$. For these experiments, the test set AUC of the ALRSF is slightly larger. This can be attributed to a slight reduction in overfitting on the training set. The difference is, however, negligible and suggests that the use of a large H in the hopes of obtaining better results is unwarranted. In Chapter 5, it was proposed that a moderate value for H should be specified to allow for adequate variability in either direction of the decimal separator.

Table 6.4: AUC performance measure comparison

Logic	H	c	p	Train AUC	Test AUC
ALRSF	1000	4	6	71.56	74.36
ALRSF	1000	3	8	71.34	74.52
ALRSF	10000	4	6	71.56	74.35
ALRSF	10000	3	8	71.34	74.48
B&B	1000	4	6	71.56	74.35
B&B	1000	3	8	71.34	74.48

In general, the comparative study substantiates the validity of the ALRSF. It gives confidence that

1. the logic of the ALRSF generates satisfactory results which can be trusted and considered as correct, and
2. the ALRSF adequately approximates the true underlying logistic regression best subset selection problem.

6.5 ALRSF cardinality selection verification

The cardinality selection logic of the ALRSF was clearly defined in Chapter 5. This chapter aims to verify whether the aforementioned logic yields the described output. For completeness, a short recap of the cardinality selection algorithm will be provided, after which the experimental results will be evaluated.

In summary, the cardinality selection algorithm evaluates the test AUC at each iteration (denoted by P_t for each iteration t), starting at iteration $t = 1$. If the test AUC for iteration t is larger than the best-recorded test AUC recorded thus far such that $P_t > P_b$, then the cardinality parameter c is incremented with one. Alternatively, the cardinality parameter will remain unchanged and the next independent variable can be considered for inclusion in the final model in iteration $t + 1$. This process is repeated up until iteration $t = T$ and will only terminate when either the stopping criterion is met or the total number of independent variables p (available for testing) have been considered.

Table 6.5: AUC obtained at each iteration of the ALRSF that was applied to an eight-variable dataset

Logic	U	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$
ALRSF	2	72.04	72.24	73.96	73.86	73.70	75.32	75.34	75.32

The result in Table 6.5 indicate that an ALRSF execution on the eight-variable problem instance considers $c = 3$ as the most appropriate choice for the cardinality parameter (in this specific instance). A consecutive limit increase of $U = 2$ was employed. This means that a test AUC (denoted by P_t) which is below the best-recorded AUC (denoted by P_b) for two consecutive iterations leads to a termination in the cardinality selection step, and that the solution search space pruning step can be initiated. The cardinality parameter c (discovered at this point) can be used to constrain the size of the cardinality set of the final model. From Table 6.5, we can see that the test AUC increased slightly from iteration $t = 1$ to iteration $t = 2$, and again from $t = 2$ to $t = 3$. The same is true from iteration $t = 2$ to $t = 3$. Alternatively, the test AUC values that are found between iterations $t = 4$ and $t = 5$ are slightly lower than the best AUC that was recorded at $t = 3$. Since the consecutive increment limit was specified to be $U = 2$, the cardinality selection step was terminated when it reached the fifth iteration and the cardinality parameter was fixed at $c = 3$.

Note that if the incremental limit variable U was larger such that $U > 2$, the cardinality parameter would have been set equal to $c = 8$ as subsequent iterations after $t = 5$ yielded test AUC values that are higher than the test AUC recorded at iteration $t = 3$. It is clear that the consecutive incremental variable U plays an essential role in determining the final cardinality parameter value c. Given that the independent variables are considered by the ALRSF in an ordinal fashion according to their IVs, it is expected that independent variables that are assessed later in the solution algorithm will not lead to a significant improvement in the model's predictive performance. As such, specifying a value for U that is too large may be unnecessary because it will result in the execution of additional algorithmic iterations that do not contribute significantly to the performance of the final model. Alternatively, some key independent variables may be overlooked if U is too small. A delicate balance should be struck between independent variable exploration and model execution time.

6.6 Grid points estimation verification

The grid range selection method, as described in Section 5.6 of Venter (2020), was utilised to manually calculate the appropriate values for the extreme points of the grid (denoted by $-\Gamma$ and Γ) which define the range of the grid used in the MIP from Chapter 5.1 . Recall that the manual approach executes the logistic regression MIP without the inclusion of cardinality constraints. The exclusion of the cardinality constraints result in a relaxation

of the MIP formulation. The LP formulation regression parameter estimates can then be used to estimate the extreme points of the grid such that

$$[-\Gamma, \Gamma] = [\min(\sum_{i \in n} \beta^T X_i) - b_1, \max(\sum_{i \in n} \beta^T X_i) + b_2]. \tag{6.1}$$

The equation in (6.1) describes the procedure that is used to identify the lower and upper bounds over the range of observations $\forall i = 1, ..., n$. It is important to note that b_1 and b_2 are static numbers that act as a safety buffer to ensure that an adequate grid range, which contains the most relevant portion of the solution search space, is considered. Venter (2020) found that suitable values for b_1 and b_2 (when applied to real-world problems) are

$$5 \leq b_{\text{opt}} \leq 20, \tag{6.2}$$

where $b_{opt} = b_2 = b_1$. By applying the aforementioned logic to the six- and eight-variable problems from Chapter 6.3, extreme poits for the grid were found to be $-\Gamma = -8.26$ and $\Gamma = 5.98$. A minimum value of 5 was selected for both b_1 and b_2 to allow for a relatively tight solution search space.

In comparison, the ALRSF found $-\Gamma = -6$ and $\Gamma = 6$ as the extreme points of the grid. The grid range obtained from the manual procedure is comparable with the results produced by the ALRSF. The lower bound is slightly smaller for the manual procedure. However, this can be attributed to the value of the safety buffer assigned to b_1 and b_2. As is the case with the adjustment constant ψ, the size of the safety buffer b_{opt} influences the search space size. Nevertheless, the differences are negligible and we may conclude that the ALRSF is adept at dynamically selecting the end points of the gird in an automated fashion while simultaneously enforcing a tight solution search space. Given that ψ affects the size of the solution search space, one must ensure that the size of ψ does not result in a search space that is unnecessarily large. For this experiment, a value of $\psi = 1$ was used, which enforced a relative tight solution search space.

6.7 Summary

This chapter was focused on verifying the results that were generated by the proposed novel ALRSF. The verification experiments attempted to instil some confidence in the final set of parameters selected by the ALRSF (constrained by the cardinality parameter c), the regression parameter estimates that were produced by the ALRSF and the suitability of the grid range that is automatically selected by the framework. A comparative study was conducted between the novel ALRSF and the standard branch-and-bound algorithm. The experimental results indicate that the ALRSF aligns with the output that is expected from a MIP formulation of the logistic regression best subset selection problem. This was confirmed by noting that the ALRSF produces identical (or nearly identical) results when compared to those yielded by the standard branch-and-bound algorithm. The ALRSF, therefore, adequately represents the underlying logistic regression problem formulation.

Chapter 7

ALRSF model validation

The goal of Chapter 7 is to validate the proposed ALRSF (that was developed in Chapter 5) and to determine if the objectives that were set out in Chapter 1 have been satisfied. The novel solution framework will be compared against the standard branch-and-bound algorithm to assess any differences in execution time, memory requirement, size of the problem instances considered, quality of integer solutions discovered and robustness.

Both the novel solution framework and the standard branch-and-bound algorithm will be used to solve four diverse logistic regression classification problems. The classification problems originate from the financial, engineering, healthcare and information technology sectors. The four problem instances and their training and test datasets were sourced from the online UCI machine learning repository. These are the Superconductivity regression, Relative location of CT slices regression, online news popularity regression and credit approval classification problems. All regression problems were transformed to binary classification problems so that the ALRSF could be used.

Three experiments will be conducted per dataset (with a varying number of observations) in order to evaluate the robustness of both solution approaches when solving relatively large subset selection problems. Section 7.2 of Chapter 7 will be focused on the evaluation of the first three steps of the ALRSF, after which Section 7.3 of Chapter 7 will be dedicated to discussing the results obtained from the fourth step of the ALRSF, which aims to tighten the MIP bound and improve the optimality gap. For all experiments, emphasis was placed on solving problem instances with a large number of independent variables and a large number of training observations.

7.1 Experimental datasets

An overview of the datasets that were utilised is presented next. This will include a discussion on the number of observations considered, the number of independent variables present, the nature of the response variable, event rate distribution and problem context. Information on the derivation of the training set from the primary dataset will also be

provided.

7.1.1 Superconductivity

The superconductivity dataset consists of observations from 21263 superconductors and their associated variables. A total of 81 independent variables are provided, some of which describe the atomic mass, density, electron affinity, thermal conductivity, fusion heat and atomic radius of each superconductor. No independent variables were removed. All independent variables were allowed to enter the best subset selection problem such that the number of variables ensured that a relatively large problem instance was considered. The continuous response variable represents the critical temperature associated with the i-th superconductor and was transformed into a binary target by specifying $Y_i = 1 \; \forall i \in n$ where $\{\text{critical temperature} \geq 1400\}$. Let Y_{bin} represent the group of observations from the set of n observations for which $Y_i = 1$.

A total of three experiments were executed for the superconductivity classification problem. Three distinct training sets were derived from the set of 21263 observations which contain 6500, 8000 and 9000 observations, respectively. The generalisation performance of the selected model (at each iteration) was evaluated against a test set with 6000 observations. The first training set is slightly unbalanced with an event rate of 72.75%. In contrast, the second training set has an event rate of 59.11% and the third set an event rate of 47.46%. Event rate can be defined as the percentage of observations that belong to Y_{bin}.

This specific dataset was selected in response to the experiments conducted in Venter (2020) and will allow for a direct comparison between the proposed novel solution framework and the standard branch-and-bound MIP formulation proposed by Venter (2020). The derived datasets contain a relatively large number of independent variables and a sizeable number of training observations compared to other problem instances that are frequently addressed in literature (Venter, 2020). Therefore, the results produced by the ALRSF for the superconductivity dataset highlights the ability of the framework to produce high-quality solutions for large real-world problems.

7.1.2 Relative location of CT slices

The relative location of CT slices dataset (regression problem) serves as a summary of Computed Tomography (CT) images from 74 patients. The 74 patients can further be subdivided into 43 males and 31 females. The dataset was donated in 2011 and consists of 53500 observations with 385 independent variables. Variables 2 to 241 are histograms that describe the bone structure of the different patients, while variables 242 to 385 describe the air inclusions inside the patient's body. The target variable is an integer variable that ranges between 0 and 180, where 0 denotes the top of the patient's head and 180 the soles of the patient's feet. The dataset contains no missing values, which means that no

imputation was required.

From the 385 independent variables, a total of 18 unary variables were removed from the dataset, as these variables would add no value to the discriminatory power of the proposed classifier. As discussed in Chapter 5, the VIF for each independent variable was calculated to provide a measure of multicollinearity present in the training set. All variables with a VIF ≥ 10 were excluded from the classification problem (according to (5.11)). In total, 125 variables were removed from the dataset due to a violation of the VIF threshold. Removing the correlated variables produced a final training set with 242 variables.

Considering that the ALRSF can currently only be applied to solve binary classification problems, the integer response variable was transformed to a dichotomous variable such that $Y_i = 1 \ \forall i \in n$ where the original response variable had a value larger than 50. Three best subset selection experiments were carried out on the relative location of CT slices dataset which meant that the dataset of 53500 observations had to be subdivided into three distinct training sets. The first experiment utilised a train and test set of 5000 and 25000 observations, respectively. The response rate for the first training set was 45.06%.

Similarly, a train and test set of 6000 and 25000 observations, respectively, were used for the second experiments with a training set response rate of 49.08%. Finally, a train and test set consisting of 6500 and 25000 observations for the third experiment was employed with a training set response rate of 48.57%. In general, the dependent variable class split is relatively balanced and is not expected to impact the generalisation performance of the ALRSF.

The CT scan classification problem was selected based on its size. Applying the ALRSF to solve classification problems with many independent variables and a moderate number of training observations will provide insight into how well the framework performs for large best subset selection problems.

7.1.3 Online news popularity

The online news popularity dataset summarises a heterogeneous set of variables associated with articles published by Mashabale, a global multi-platform media and entertainment company. The dataset is publicly available from the UCI machine learning repository and was published in 2015. The continuous response variable describes the number of shares on social networks. The dataset contains 61 independent variables and 39797 observations. Some of the independent variables are used to describe the number of words present in the title, the number of videos, the date on which the article was published and the title's popularity. An exhaustive list that describes each of the 61 variables can be found on the UCI machine learning repository and will not be discussed here. The dataset contains no missing values, which means that no imputation is required.

From the 39797 observations, three training sets with 11000, 12000 and 13000 observations, respectively, were extracted. Additionally, a test dataset of 24000 observations was

compiled. From the 61 independent variables, 13 were removed. A total of 45 independent variables were available for model fitting. The target variable (number of shares) was modified to be suitable for a classification problem such that $Y_i = 1 \ \forall i \in n$ where the original response variable was greater or equal to 1400.

The first training set (with 11000 observations) had an event rate of 55.93%, while the second (with 12000 observations) had an event rate of 54.98%, which is marginally less skewed. Similarly, the final training set had a response rate of 54.67%. In general, the training sets were relatively balanced for all experiments.

By testing different classification problem scenarios, the robustness of the ALRSF can be evaluated and its value proposition to the industry can be justified. This particular classification problem contains a moderate number of independent variables and a large number of training observations (when compared with the other datasets considered on this chapter and in literature).

7.1.4 Default of credit card clients

In Section 6.3 of Chapter 6, a brief introduction was provided to the default credit card clients classification problem. The complete problem instance with all 24 independent variables is considered for the experiments that were conducted. The independent variables describe the clients' personal information and financial status (some of which were presented in Chapter 6). A detailed description of each explanatory variable can be found on the UCI machine learning repository website and will not be discussed here. The binary response variable represents the default status of a client. None of the variables were removed due to the VIF threshold and there were no missing values in the dataset.

Three experiments were conducted by using training datasets that consisted of 8000, 10000 and 11000 observations, respectively, and a test dataset that contained 10000 observations. For the first experiment, a response rate of 28.52% was observed. Similarly, unbalanced response rates of 22.59% and 18.83% were recorded for the second and third experiments, respectively.

Moderately unbalanced training sets were formulated for these experiments to evaluate the influence of unbalanced data on the generalisation performance of the final model proposed by both the ALRSF and standard branch-and-bound algorithms. The main purpose of this test is to further examine the robustness of the solution framework and to verify its performance on problems that are relevant to the financial industry (where logistic regression and variable selection are very topical).

7.2 Experimental results (Steps 0 to 2 of the ALRSF)

The experimental results (associated with steps $S = 0$ to $S = 2$ of the proposed ALRSF) are summarised in Table 7.1 and 7.2. The objective of these experiments is to benchmark the ALRSF against the standard branch-and-bound MIP (proposed by Venter (2020)) that is used for best subset selection. To this effect, key measures such as the memory usage associated with each solution algorithm, the final optimality gap percentage achieved, the test and train performance measures, the execution time and the cardinality set that was identified, were considered. Accuracy and AUC were selected as the preferred performance measures for these experiments. However, any appropriate statistical measure can be used when defined prior to the execution of the solution algorithm. All of the aforementioned metrics allow for a quantitative approach that can be used to determine if the objectives from Chapter 1 have been satisfied.

A few abbreviations to take note of (with reference to Tables 7.1 and 7.2) include ME and OPT, which are defined as follows:

1. The acronym ME is short for "memory exceeded" and is assigned to a problem instance when Cplex failed as a result of an out-of-memory exception before any feasible integer solution could be found.
2. The abbreviation OPT refers to optimality. Optimality is defined as the gap between the upper integer and lower integer bound that satisfies the optimality tolerance defined by Cplex. When the optimality tolerance is satisfied, the solution discovered can, for all intents and purposes, be considered as the best possible solution available.

In order for a comparative study to be possible, the same values need to be used for the cardinality parameter c and the extreme points of the grid (denoted by $-\Gamma$ and Γ). This was determined by executing the ALRSF's cardinality selection step (denoted by $S = 0$) and the grid point evaluation logic (defined as part of steps $S = 0$ and $S = 1$ of the ALRSF). The latter provides an automated approach that can be used to determine these hyper-parameters with limited-to-no user intervention. The final values (as per the ALRSF logic) were then used as inputs in the standard branch-and-bound MIP. This allowed a similar solution search space for both solution algorithms, thereby enabling a direct comparison of the results in Table 7.1 and Table 7.2.

7.2.1 Predictive performance

Figures 7.1 to 7.4 exhibit the train and test AUC recorded at each iteration of the ALRSF after the execution of steps $S = 0$ to $S = 2$ (for all four datasets that were discussed in Section 7.1 of Chapter 7). The iterative nature of the solution framework provides the model developer with a unique ability to analyse each independent variable's influence on the selected model's generalisation performance. The graphical analysis also provides a way to detect overfitting, as is evident in Figure 7.2.

Table 7.1: ALRSF vs B&B - Experimental Results

Instance Identifier/ Observations	Cardinality / Total Variables (After VIF Exclusion) / VIF Excluded	Memory Used (GB)	Integer Solution	Gap (%)	Training Accuracy (%)	Training AUC	Test Accuracy (%)	Test AUC	Duration (sec)
Superconductivity - Classification Problem									
1-ALRSF/ 6500	2/81/0	25.4	-808.75	42.48	95.31	98.34	83.77	86.47	345 600
1-B&B/ 6500	2/81/0	>32	-1819.12	75.40	86.11	92.77	77.93	86.10	40 000
2-ALRSF/ 8000	6/81/0	28.1	-823.64	30.73	96.35	98.78	84.95	87.17	345 600
2-B&B/ 8000	6/81/0	>32	-2260.73	74.47	89.49	94.05	82.32	87.54	31 000
3-ALRSF/ 9000	17/81/0	30.2	-773.22	23.52	97.11	99.07	83.45	87.21	345 600
3-B&B/ 9000	17/81/0	>32	ME	ME	ME	ME	ME	ME	3 285
Relative CT Slice Location - Classification Problem									
1-ALRSF/ 5000	32/242/125	26.4	-127.75	100.00	99.20	99.84	87.41	89.04	691 200
1-B&B/ 5000	32/242/125	>32	-1123.80	99.96	90.82	96.74	84.04	91.23	15 053
2-ALRSF/ 6000	23/242/125	31.1	-359.40	97.82	97.83	99.56	89.56	93.19	691 200
2-B&B/ 6000	23/242/125	>32	-4158.88	99.81	50.92	50.00	60.62	50.00	8 000
3-ALRSF/ 6500	23/242/125	>32	-439.70	95.92	97.45	99.47	90.16	94.34	542 611
3-B&B/ 6500	23/242/125	>32	-3558.65	99.69	77.32	81.79	71.82	77.22	2 304

Table 7.2: ALRSF vs B&B - Experimental Results

Instance Identifier/ Observations	Cardinality / Total Variables (After VIF Exclusion) / VIF Excluded	Memory Used (GB)	Integer Solution	Gap (%)	Training Accuracy (%)	Training AUC	Test Accuracy (%)	Test AUC	Duration (sec)
Online News Popularity - Classification Problem									
1-ALRSF/ 11000	41/45/13	20	-7100.34	OPT	63.06	66.25	63.63	68.89	116 805
1-B&B/ 11000	41/45/13	>32	-7107.75	0.11	62.90	66.13	66.13	68.89	4 900
2-ALRSF/ 12000	45/45/13	22	-7729.86	OPT	62.94	66.75	64.91	69.01	157 817
2-B&B/ 12000	45/45/13	30	-7729.86	OPT	62.94	66.75	64.91	69.01	11 759
3-ALRSF/ 13000	42/45/13	31.9	-8388.35	OPT	63.07	66.96	57.68	69.25	162 000
3-B&B/ 13000	42/45/13	>32	-8956.39	6.34	54.67	50.00	53.00	50.00	2 785
Credit Card Default - Classification Problem									
1-ALRSF/ 8000	13/23/0	25.2	-3825.34	0.06	80.13	70.31	81.33	71.59	345 600
1-B&B/ 8000	13/23/0	>32	-5266.13	0.21	79.95	70.25	82.25	71.56	3 800
2-ALRSF/ 10000	13/23/0	29.1	-4785.78	0.05	79.89	70.85	81.43	72.13	345 600
2-B&B/ 10000	13/23/0	>32	-4804.15	0.42	79.84	70.52	81.44	71.17	3 500
3-ALRSF/ 11000	13/23/0	31.2	-5254.68	0.04	80.03	70.58	81.49	71.84	345 600
3-B&B/ 11000	13/23/0	>32	-5261.57	0.13	79.94	70.59	81.44	71.79	3 218

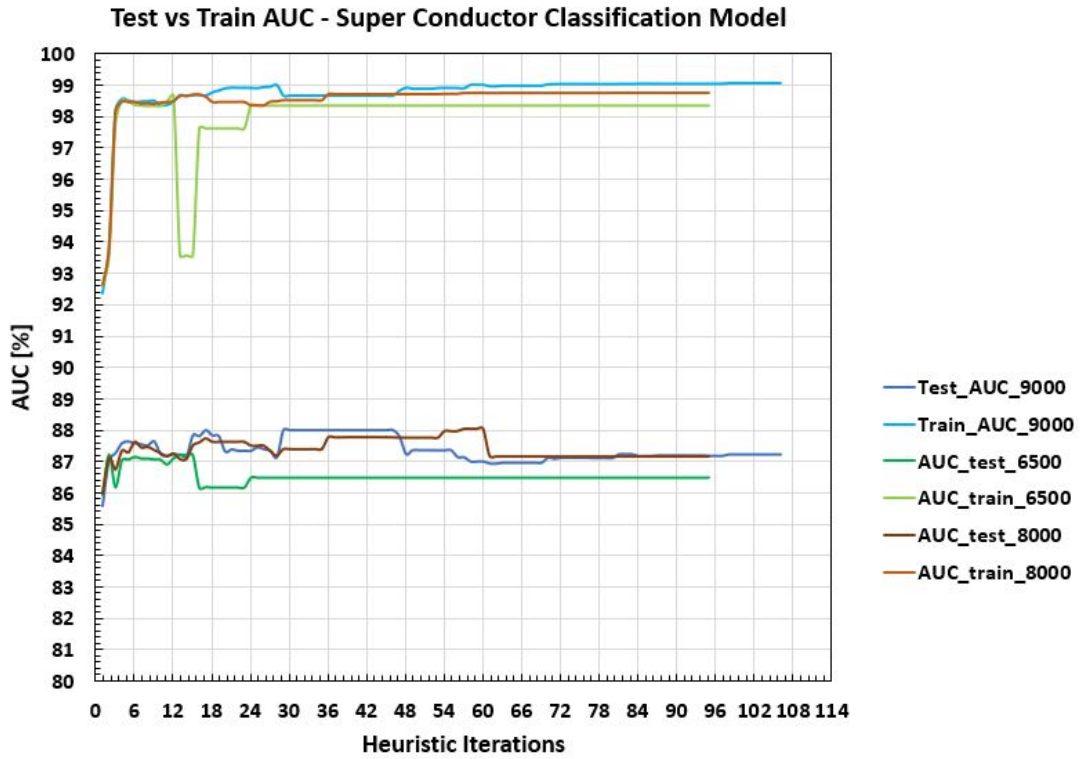


Figure 7.1: Superconductivity Classification Problem

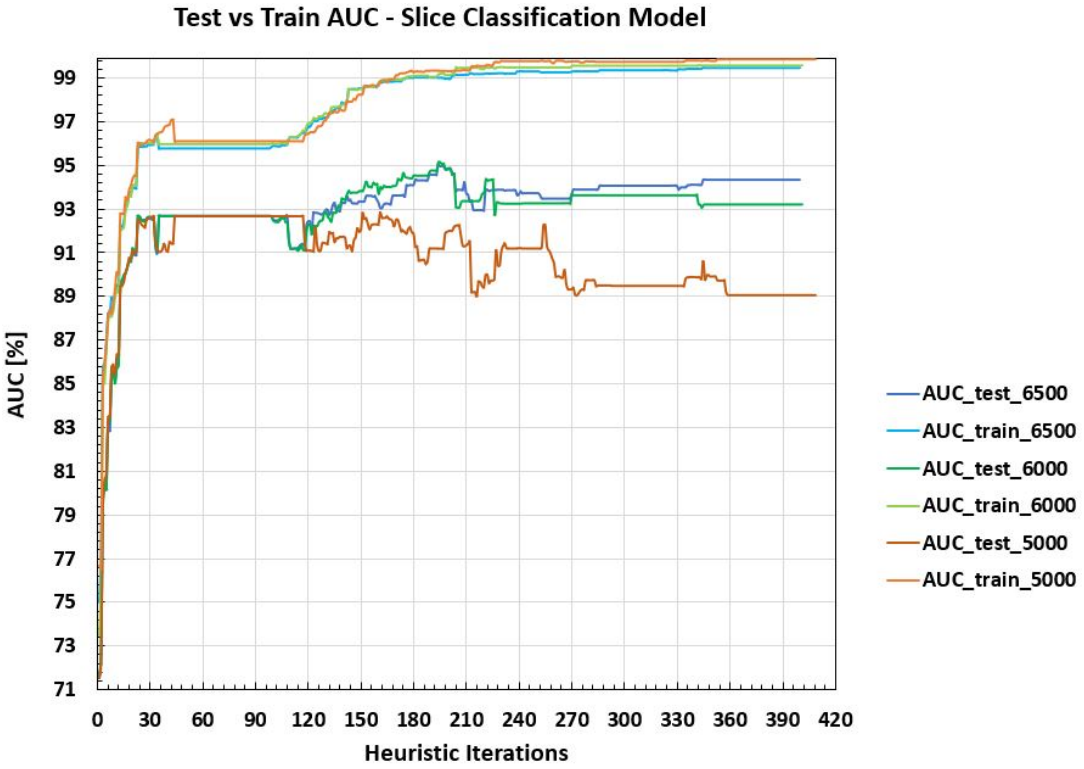


Figure 7.2: Relative location of CT slices Classification Problem

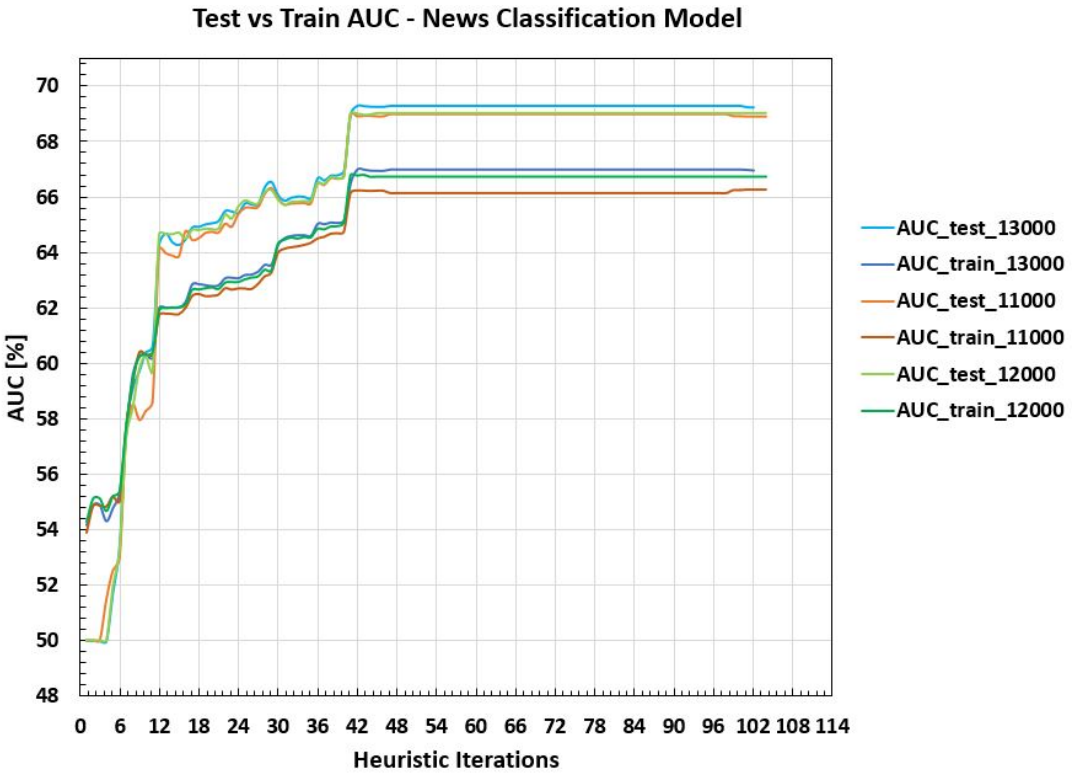


Figure 7.3: Online News Popularity Classification Problem

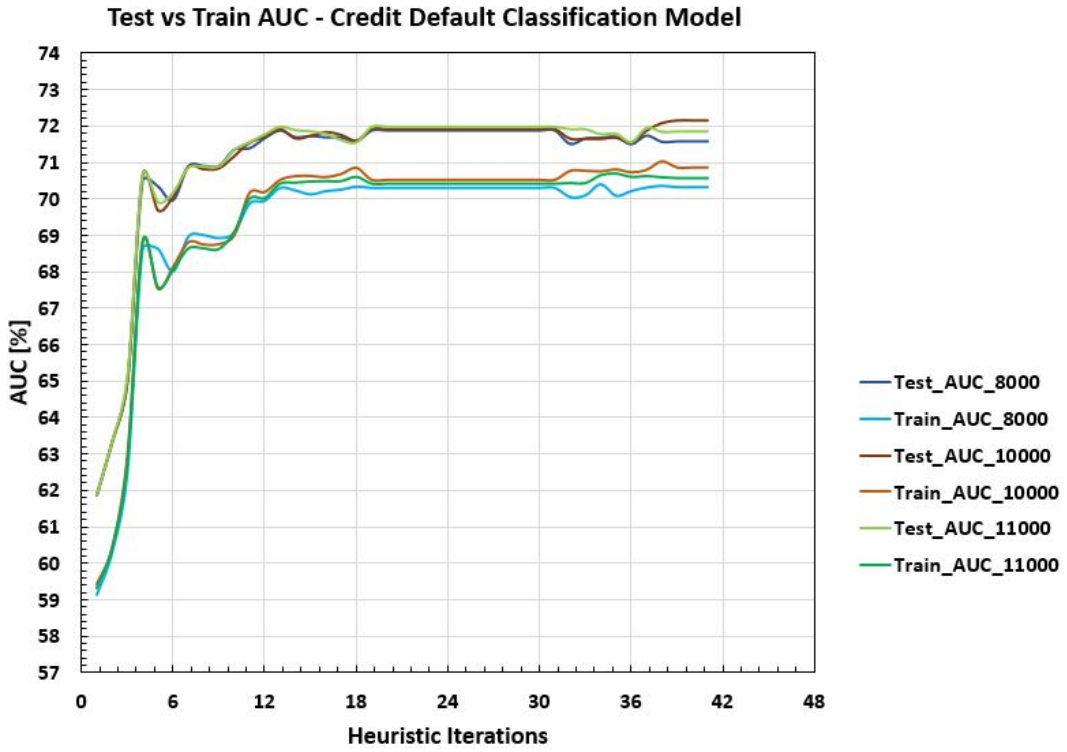


Figure 7.4: Credit Card Default Classification Problem

In Figure 7.2 (the CT slices classification problem) a clear drop in generalisation performance can be seen from iteration 150 onwards, which indicates that the model is overfitting

on the training set. An increase in the AUC of the training set further corroborates the presence of overfitting. This type of analysis is impossible with most deterministic MIP formulations that are solved with the branch-and-bound algorithm.

The three distinct algorithmic steps are also visible in the graphical representation of the performance measures. Since the independent variables are added to the solution framework sequentially (based on IV rank order), a clear spike in the AUC can be observed at the start of the cardinality selection step $S = 0$, after which the AUC starts to stabilise and achieve a plateau. The cardinality parameter values (which are denoted by c and derived as part of the first step of the ALRSF) are presented in Tables 7.1 and 7.2. Furthermore, Figures 7.1 to 7.4 show how the consecutive selection limit U and selection criteria $P_t > P_b$ influence the final model. From Figure 7.3, it is clear that a slightly smaller model would have been obtained if a different cardinality selection criteria such as $P_t > (1 + 0.1)P_b$ was used.

When the number of training observations is small, the ALRSF and standard branch-and-bound approach yield similar predictive performance on the test sets. However, the ALRSF outperforms the branch-and-bound technique, with respect to the quality of the final solution, when larger datasets are considered. Observe from Figure 7.1 that the branch-and-bound approach could not produce a solution for the third experiment on the superconductivity dataset. Alternatively, the ALRSF found a high-quality solution with a test AUC of 87.21%. This equates to a Gini coefficient of 74.42%. Furthermore, the ALRSF achieved a relative lift of 86.38% and 22.17% in the AUC for the second and third experiments of the relative CT slice location problem, respectively, when compared to the standard branch-and-bound MIP. This can be seen as a substantial improvement in predictive performance when the size of the problem instance is considered. A similar observation can be made for the online news popularity dataset, where a relative lift of 38.5% in the test AUC was recorded for the third experiment. These findings clearly support the notion that the proposed ALRSF can solve much larger best subset selection problems when compared to the standard branch-and-bound subset selection approach.

7.2.2 Memory comparison

Figure 7.5 provides a graphical summary of the memory requirements of each solution algorithm (ALRSF vs branch-and-bound). The machine on which the experiments were conducted had a total of 32 GB random access memory (RAM) available for computational purposes. In almost all cases, the standard branch-and-bound algorithm terminated because of out-of-memory exceptions and had to be executed for a second time with a predefined time limit enforced. This allows for the extraction of the last-generated solution from the Cplex concert technology API. The only exception to the latter was the second experiment that was conducted for the online news popularity dataset. The reason for the slightly reduced memory requirement for the online news popularity experiment can be ascertained by critically evaluating Table 7.2. All of the independent variables were allowed to enter the cardinality set, which meant that the cardinality constraints in (5.5)

and (5.6) could be excluded from the MIP. Ultimately, this meant that the optimisation problem did not have to consider complicating constraints.

In contrast, the ALRSF exhibits a reduced memory requirement for most of the experiments, with memory usage increasing alongside an increase in the number of training observations. The ALRSF boasted with a reduction in memory requirement that ranged between 0.9 and 10 GB, depending on the specific problem instance being considered. However, it is important to note that for instances where the memory usage was capped at 32 GB, the total memory gain (as a result of employing the ALRSF) cannot be ascertained as the algorithm could have required anything from 32 GB upwards. The reduction in memory requirements for problems that contain fewer observations therefore paints a clearer picture of the contributions made by the ALRSF. The rate at which the memory requirement increases (with reference to the ALRSF) depends on the size of the training set and the number of independent variables considered.

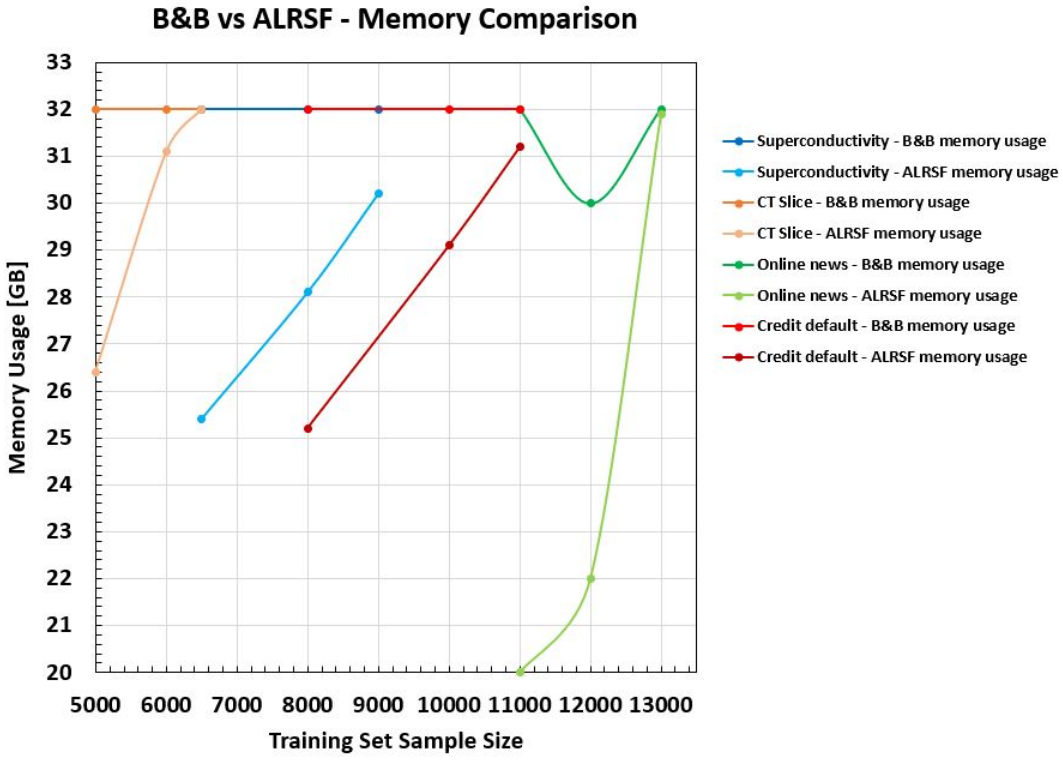


Figure 7.5: BandB vs ALRSF Memory comparison

The ALRSF outperforms the standard branch-and-bound approach with respect to the computational memory that is required to find high-quality solutions. The improvement can mainly be attributed to the reformulation of the underlying logistic regression MIP, which forms the core of this thesis. The newly proposed MIP allows the user to capitalise on the unique block structure of the MIP by harnessing the Benders decomposition solution algorithm native to Cplex. This provides greater implementation flexibility to the proposed ALRSF (with respect to the size of the problem instance considered).

7.2.3 Optimality gap and integer solution comparison

The optimality gap is a measure that is used to define by how much the best integer solution deviates from optimality. Figures 7.6 and 7.7 provide a graphical comparison of the optimality gaps that are obtained when either the ALRSF or standard branch-and-bound technique is used for each of the datasets discussed in Section 7.1 of Chapter 7.

In most cases, the ALRSF outperformed the standard branch-and-bound algorithm. The only instance for which the branch-and-bound algorithm provided a better gap percentage was the first experiment that was conducted on the relative CT slice location dataset. This may be attributed to a slightly tighter upper bound that is obtained by the branch-and-bound algorithm when compared to the Benders decomposition algorithm. The largest difference was observed for the superconductivity dataset, where a gap percentage difference of 75% was recorded. As the number of training observations increased, the standard branch-and-bound algorithm struggled to find high-quality solutions before terminating on out-of-memory exceptions. The proposed ALRSF consistently discovered high-quality solutions while simultaneously improving the optimality gap percentage. The quality of the integer solutions that were discovered by the ALRSF (as summarised in Tables 7.1 and 7.2), in comparison to those found by the branch-and-bound technique, also attest to the robustness and flexibility of the proposed ALRSF. In most cases, the ALRSF still had memory capacity available and was only terminated when a predefined runtime limit was triggered.

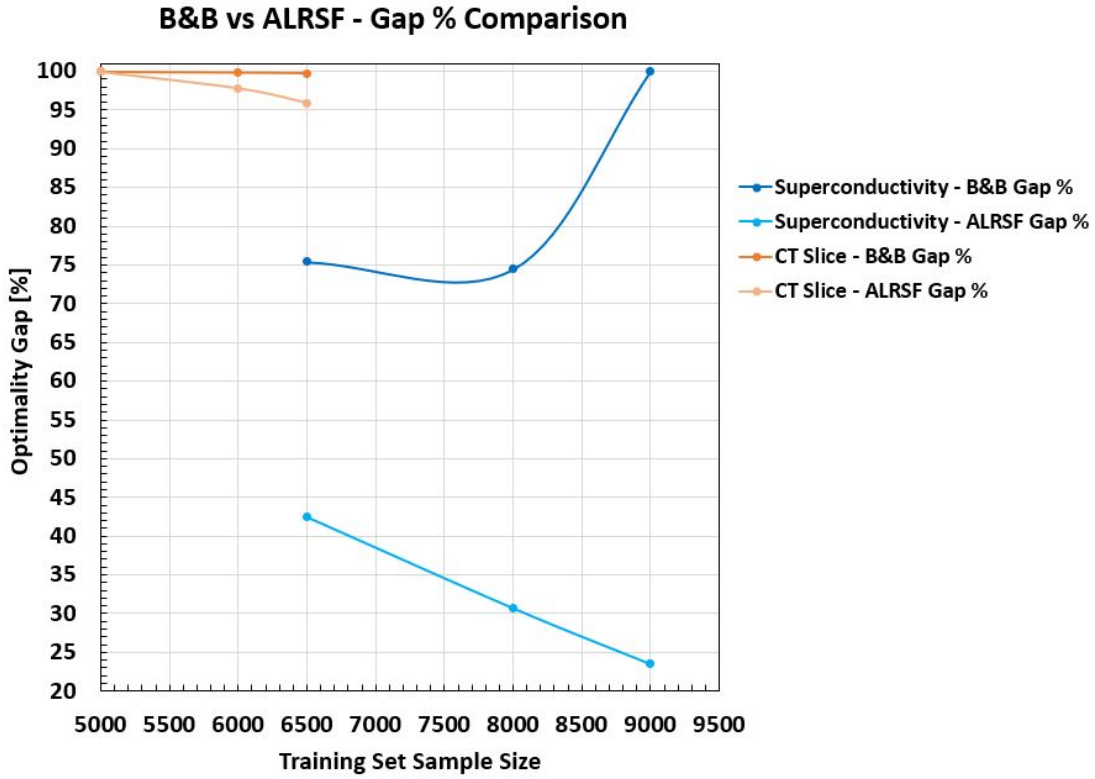


Figure 7.6: BandB vs ALRSF Gap percentage comparison

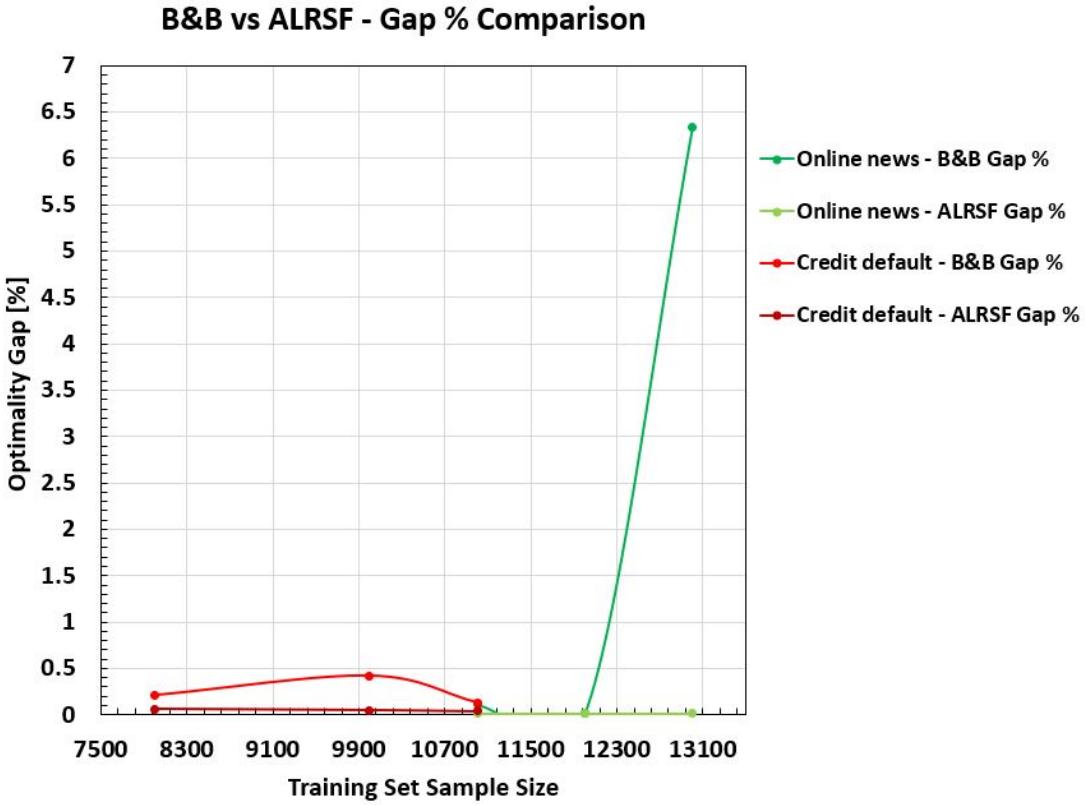


Figure 7.7: BandB vs ALRSF Gap percentage comparison

7.2.4 Algorithmic execution time

Tables 7.1 and 7.2 show that the ALRSF has much longer execution times when compared to the standard branch-and-bound algorithm. However, it is important to note that in most cases, the branch-and-bound algorithm terminated early in the solution process based on out-of-memory exceptions. If the memory resources were sufficiently large, the branch-and-bound algorithm would also exhibit extended execution times. Most of the time spent during the execution process of the ALRSF is mainly allocated to the search space pruning step. A greater number of algorithmic iterations is required when there are more independent variables in the training set. This is evident from the experiments that were conducted on the relative CT slice location classification problem. Nevertheless, the ability of the ALRSF to solve large-scale real-world best subset selection problems greatly outweighs the slightly longer execution times.

7.2.5 Additional observations made

Some additional inferences can be made from the visual trends depicted in Figures 7.1 through 7.7 and the results provided in Tables 7.1 and 7.2, which will be discussed next.

1. A reduction in overfitting on the training set is observed when the number of training observations increases. Larger training sets are generally a better representation

of the underlying population distribution and allow for improved generalisation performance on unseen data.

2. The cardinality selection step dynamically determines the appropriate cardinality parameter value to use based on the problem-specific information provided.

3. The linearised grid point values (which are bounded by $[-\Gamma, \Gamma]$) generally increase when the number of independent variables increases or when the size (in absolute terms) of the regression parameter estimates increases.

4. A slight increase in the predictive performance (AUC or accuracy) on the training set can have a relatively large influence on the generalisation capabilities of the selected model. Therefore, insights into independent variable interactions and their influence on the model's predictive power, along with continuous performance monitoring on both the test and train datasets, can add significant value to the final model selection. The ALRSF provides functionalities to that effect and attempts to generate descriptive and parsimonious predictive models.

5. In general, the larger the number of variables included in the cardinality step (in relation to the total number of independent variables available), the closer the gap percentage is to optimality. This can be attributed to the reduced number of independent variable combinations that the solution algorithm should evaluate before optimality is proven.

7.3 Experimental results (Step 3)

The experimental results presented in this chapter will focus on the final step of the ALRSF, which is referred to as the bound tightening step (denoted by $S = 3$). This step utilises the concept of Lagrangian relaxation along with a sub-gradient solution algorithm to tighten the upper bound of the final solution that is obtained in the third step of the ALRSF. Note that there is no real need for bound tightening in the case of the online news popularity and credit card default classification datasets as both problems yielded solutions that are close to optimality (as can be seen when studying the optimality gaps in Table 7.2). As such, step four of the ALRSF was only applied to the superconductivity and relative CT slice location classification problems because the gap percentages for these datasets range from 23.52% up to 100%, which means that a great deal of improvement can still be made. The value that is added by the proposed bound tightening step of the ALRSF will be demonstrated in this chapter.

Table 7.3 provides the results of a comparative study between the upper bound and gap percentage that are obtained when using the standard branch-and-bound algorithm with LP relaxation, the proposed ALRSF with benders decomposition or a Lagrangian relaxation bound tightening algorithm (as given in the fourth step of the ALRSF). The proposed Lagrangian relaxation bound tightening algorithm outperformed both the standard branch-and-bound algorithm and the standard ALRSF (that only consists of steps

$S = 0$, $S = 1$ and $S = 2$) by a large margin for all experiments considered. This is substantiated by evaluating the distance between the lower and upper bounds produced by each algorithm. For the superconductivity classification dataset, a reduction of up to 72% in the optimality gap was observed when the standard branch-and-bound results are compared with the ALRSF bound tightening algorithm (for training sets that contain 9000 observations). This can be seen as a significant gain and provides the model developer with greater confidence that the resulting solution is much closer to optimality.

Table 7.3: B&B LP relaxation (LPR) vs ALRSF LP relaxation (LPR) vs ALRSF Lagrangian relaxation (LR) - Experimental Results

Instance Identifier/ Observations	Cardinality / Total Variables (After VIF Exclusion) / VIF Excluded	Integer Solution (Lower Bound)	Upper Bound	Gap (%)
Superconductivity - Classification Problem				
1-B&B-LPR/ 6500	2/81/0	-1819.12	-447.54	75.40
1-ALRSF-LPR/ 6500	2/81/0	-808.75	-465.16	42.48
1-ALRSF-LR/ 6500	2/81/0	-808.75	-536.44	33.67
2-B&B-LPR/ 8000	6/81/0	-2260.73	-577.24	74.47
2-ALRSF-LPR/ 8000	6/81/0	-823.64	-570.57	30.73
2-ALRSF-LR/ 8000	6/81/0	-823.64	-653.69	20.63
3-B&B-LPR/ 9000	17/81/0	ME	ME	100
3-ALRSF-LPR/ 9000	17/81/0	-773.22	-591.38	23.52
3-ALRSF-LR/ 9000	17/81/0	-773.22	-680.62	11.98
Relative CT Slice Location - Classification Problem				
1-B&B-LPR/ 5000	32/242/125	-1123.80	-0.49	99.96
1-ALRSF-LPR/ 5000	32/242/125	-127.75	0.00	100.00
1-ALRSF-LR/ 5000	32/242/125	-127.75	-72.39	43.34
2-B&B-LPR/ 6000	23/242/125	-4158.88	-8.06	99.81
2-ALRSF-LPR/ 6000	23/242/125	-359.40	-7.82	97.82
2-ALRSF-LR/ 6000	23/242/125	-359.40	-185.33	48.43
3-B&B-LPR/ 6500	23/242/125	-3558.65	-10.90	99.69
3-ALRSF-LPR/ 6500	23/242/125	-439.70	-17.94	95.92
3-ALRSF-LR/ 6500	23/242/125	-439.70	-200.88	54.32

As with the superconductivity dataset, the optimality gap associated with the relative CT slice location experiments was improved considerably, with a reduction from 99% to values below 50% being recorded. As outlined in Sections 5.2.4 and 5.2.5 of Chapter 5, the proposed bound tightening step clearly achieves the objective of tightening the upper bound (which in turn reduces the optimality gap) of the final solution produced by the first three steps of the ALRSF. The largest improvement for the relative CT slice location experiments was observed for the dataset consisting of 5000 training observations, where a reduction of up to 56.67% in the optimality gap is achieved (compared to the standard

branch-and-bound and the standard three-step ALRSF approach).

Generally, the memory requirement associated with the fourth step of the ALRSF (based on the experiments that were conducted for Table 7.3) ranges between 11 and 16GB. Furthermore, the memory requirements are also noted to increase when the number of training observations increases. Therefore, the proposed bound tightening step will not limit the size of the problems that the ALRSF can consider since the memory requirement is far below the available memory (32 GB). All experiments were allowed to run for approximately 12 hours, after which no significant improvement in the upper bound was observed. It would be beneficial to test different step size rules (other than the Polyak step size) to assess how the choice of the step size will influence the algorithmic convergence of the proposed algorithm.

7.3.1 Algorithmic bound tightening results

Figures 7.8 and 7.9 provide a visual representation of the iterative bound tightening that occurs as part of the unique algorithmic formulation proposed by the novel ALRSF. The first few steps focus on pruning the lower bound, while step $S = 3$ aims to tighten the upper bound. A direct relationship is observed between the number of algorithmic iterations and the optimality gap. An increase in the number of iterations leads to a reduction in the final optimality gap, as can be seen by the convergence of the two bounds towards the optimal solution. This is, however, only true up to the point where the proposed algorithm starts to converge. In order to provide proof of optimality, step $S = 2$ of the ALRSF (optimality step) will have to be executed until the solution search space is sufficiently pruned such that no improvement in the best integer solution is possible. This may be highly time-consuming, depending on the complexity of the underlying problem.

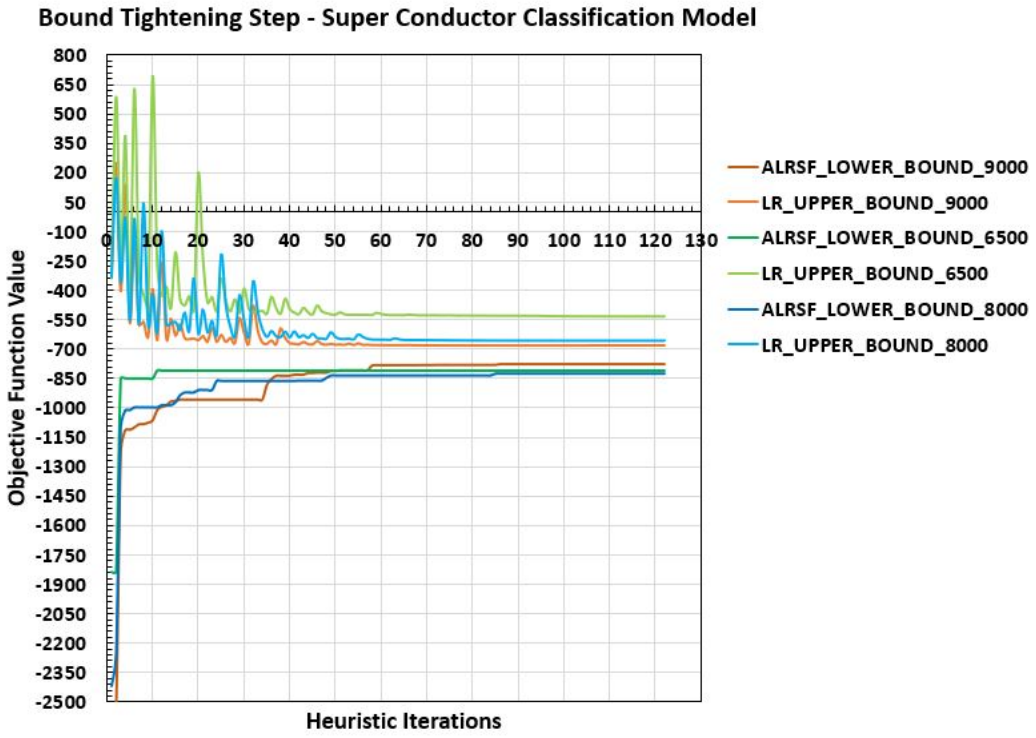


Figure 7.8: Superconductivity classification problem - bound tightening

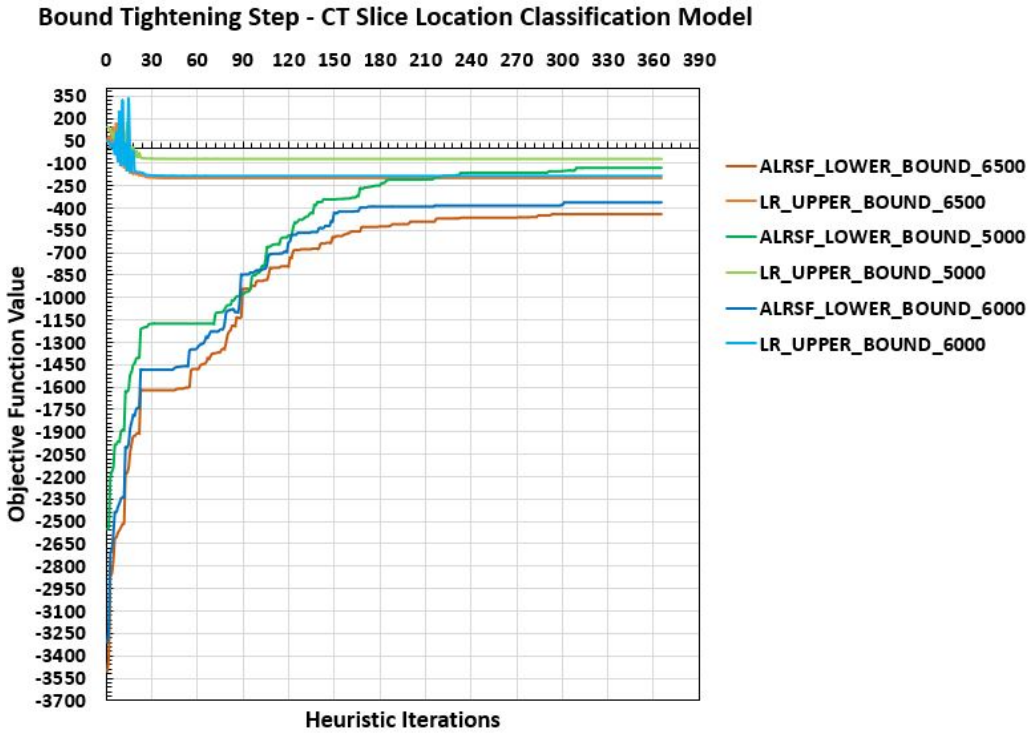


Figure 7.9: Relative location of CT slice classification problem - bound tightening

The gap between the upper bound (dual solution) and lower bound (primal solution) is known as the duality gap. As stated in Chapter 4, strong duality is not always possible. However, weak duality affords us the ability to improve the upper bound such that the

overall optimality gap is reduced. The upper bound that is obtained by using the Lagrangian relaxation approach is far better than the upper bound that is produced by a linear relaxation of the standard branch-and-bound technique.

It is also important to note from Figures 7.5 and 7.6 that the bound tightening step exhibits what seems to be a zigzagging effect. This is a known limitation of the sub-gradient algorithm and may lead to an increase in the amount of time that is required to ensure convergence. Future recommendations to address these limitations will be provided in Chapter 8. Nevertheless, the proposed framework significantly improves the final optimality gap that is obtained (as is evident from Figures 7.5 and 7.6) by approaching the solution search space from both the upper and lower bound. It also results in a reduction in the exorbitant computational resources that are demanded by the standard branch-and-bound approach. Ultimately, the solution framework yields superior solutions that are closer to optimality when compared to the standard branch-and-bound technique.

7.3.2 Optimality Gap Improvements

Figure 7.10 provides a graphical representation of the optimality gap that is obtained as part of a comparative study between the standard branch-and-bound algorithm, the stand-alone ALRSF ($S = \{0, 1, 2\}$) and finally, the ALRSF with the addition of the bound tightening step ($S = 3$). The bound tightening step (which is based on Lagrangian relaxation) adds significant value to the solution framework, especially when applied to large-scale problems with a moderate-to-large number of training observations and many independent variables. This is substantiated by critically evaluating the results presented in Table 7.3. For the relative CT slice location problem, the ALRSF ($S = \{0, 1, 2\}$) did not succeed in reducing the optimality gap when compared to the standard branch-and-bound algorithm. This is because the lower bound produced by the LP relaxation of the branch-and-bound approach was close to zero. This results in a relatively large gap percentage as a result of how the gap is calculated, as is evident in Figure 7.10. Alternatively, the bound tightening step (that utilises the proposed Lagrangian relaxation approach) reduced the gap percentage for all CT slice classification experiments to below the 55% mark. Similarly, the bound tightening step was able to improve the optimality gap associated with the superconductivity classification experiments to values below the 35% mark. This amounts to a relative gap improvement of up to 56.66% and 88.02%, respectively, for the two different problem instances. It is evident that the addition of the bound tightening step further bolsters the ALRSF's capability to solve large-scale classification problems while also providing a greater measure of confidence in the quality of the solution. The Lagrangian relaxation formulation (proposed in Section 5.2.4 of Chapter 5) provides a tighter relaxation when compared to an LP relaxation and, as such, is a key contribution that improves the flexibility and robustness of the solution framework.

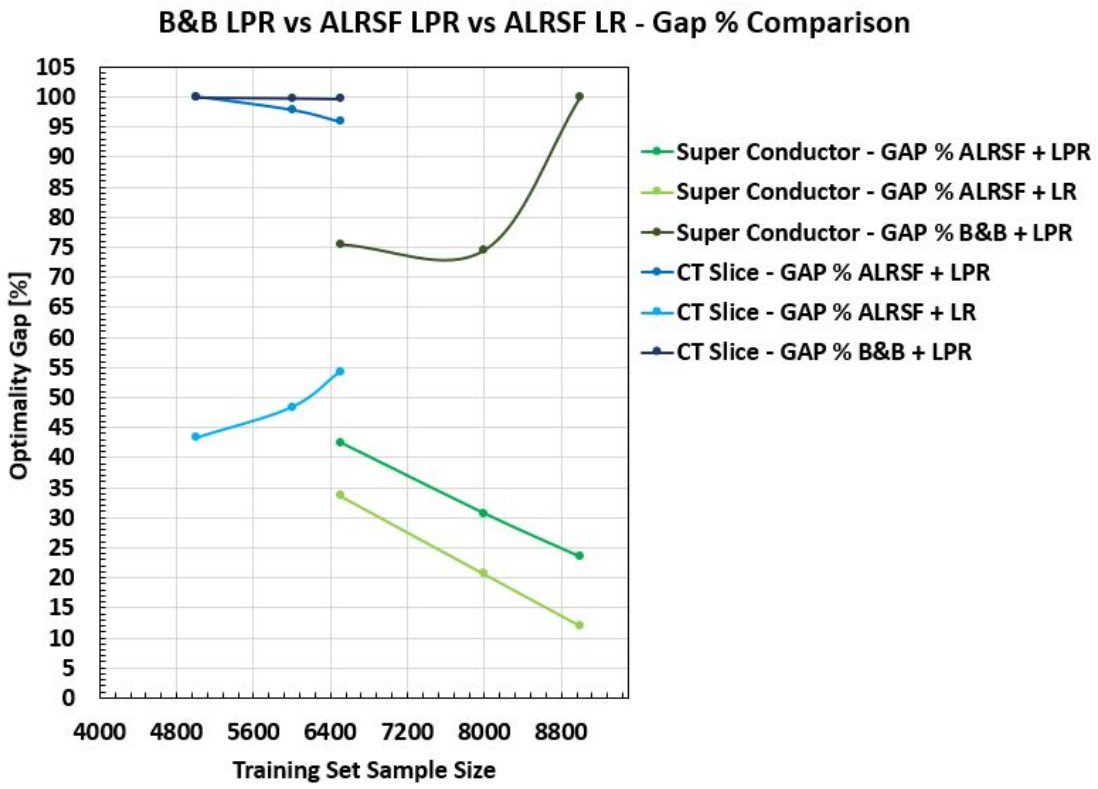


Figure 7.10: Improvement in optimality gap - bound tightening

7.4 Summary

A number of key observations can be made as part of a comparative study on the proposed ALRSF and the standard branch-and-bound best subset selection MIP, which will be discussed next.

1. Empirical evidence suggests that the ALRSF outperforms the branch-and-bound algorithm when considering the memory that is required to solve large-scale classification problems. The reduction in memory requirements can mainly be attributed to the novel heuristic that is employed during the search space pruning step (which includes the use of Cplex's native Benders decomposition algorithm) before proof of optimality is attempted.

2. Higher quality solutions were discovered when the ALRSF was applied to large-scale best subset selection problems. In contrast, an apparent degradation in solution quality was noted when the standard branch-and-bound algorithm is used.

3. The proposed novel framework provides an iterative solution approach that allows incremental monitoring of selected performance measures. This provides analytical insight into the solution algorithm logic and may assist with the early detection of overfitting on the training set.

4. For most experiments, the ALRSF was able to marginally outperform the branch-and-bound algorithm when considering the discrimination between the two binary target classes. This is substantiated by critically evaluating the test AUC for each experiment. There were, however, some instances where the branch-and-bound algorithm was slightly superior. This can mainly be attributed to overfitting by the ALRSF on the training set. Fitting a logistic regression model (using the best subset of variables identified by the ALRSF) to a training set that is not a representative sample of the actual underlying population can result in overfitting, which negatively impacts generalisation performance.

5. Incorporating a Lagrangian relaxation based bound tightening step greatly assisted in reducing the optimality gap (when considering large-scale problems). Being able to find solutions closer to the optimal solution further highlighted the quality of the solutions discovered by the ALRSF.

Chapter 7 focused on validating the proposed solution framework and justifying its contribution to the field of data science. The ALRSF provides a holistic approach to solving the well-known best subset selection problem and limits the manual intervention required to find parsimonious models. This is achieved by utilising the novel automated four-step solution approach put forward by the ALRSF, as discussed in Chapter 5.

Chapter 8

Conclusion and recommendations

8.1 Concluding remarks

A novel ALSRF was proposed to achieve the objectives that were outlined in Chapter 1. The main contributions of the proposed ALRSF can be attributed to its novel four-step model fitting approach, which is mentioned below.

1. Automatically selecting the most suitable cardinality set size (defined by the parameter value c) based on the specific problem instance being considered. Furthermore, the cardinality selection is also based on the predictive value added by each independent variable in relation to the overall generalisation capability of the model.
2. The implementation of a novel search space pruning heuristic. Additionally, the approach is able to leverage the Cplex native Benders decomposition algorithm as a result of the formulation modifications that were proposed in Chapter 5.
3. The calculation of an optimality gap (which is impossible when using variable subset selection techniques other than MIP formulations). The optimality step of the ALRSF also attempts to further improve the best integer solution that is discovered as part of the solution search space pruning step.
4. The final step of the ALRSF attempts to tighten the upper bound of the MIP maximisation problem by employing a Lagrangian relaxation approach while utilising a novel sub-gradient-based solution heuristic to identify the most appropriate Lagrangian multipliers. In addition, the proposed sub-gradient-based solution heuristic attempts to reduce the Big-M values at each algorithmic iteration which in turn reduces the solution search space size along with the number of iterations required to approach convergence.

Empirical results (presented in Chapter 7) highlight the superiority of the proposed ALRSF when solving high-dimensional problem instances with a relatively large number of independent variables and observations. The proposed algorithm reduces the memory require-

ment associated with finding high-quality solutions for large-scale problem instances and also results in a reduction in the final optimality gap (as is evident from the results of the experiments in Chapter 7). The capability of finding high-quality solutions (with a measure of optimality in relation to the optimal solution) for sizeable best subset selection problems may incentivise greater future adoption of exact deterministic solution algorithms.

The novel solution framework provides an automated holistic approach towards the solving of best subset selection problems and limits user intervention. As such, the requirement for highly skilled resources to facilitate the model-building process is reduced, which frees up time that can be directed towards alternative endeavours. The proposed framework pushes the boundaries of what is currently available in literature and may act as a stepping stone towards solving this interesting yet complex NP-hard problem.

8.2 Future research initiatives

The proposed automated solution framework naturally lends itself to other modelling techniques, such as linear regression and even possibly neural networks, since regression parameter pruning can reduce the initial search space before attempting to prove optimality. However, the underlying mathematical modelling structure will need to be revised, as this may change depending on the selected modelling technique. Future research endeavours may aim to achieve the following:

1. Develop new MIP formulations that are able to accommodate alternative modelling techniques, such as linear regression and neural networks (that can fit into the solution framework that was proposed in Chapter 5).
2. Expand the MIP formulation (that was presented in Section 5.2 of Chapter 5) to cater for multi-class classification problems.

The proposed solution framework mainly prunes the solution search space by utilising the novel search space pruning heuristic (presented in Section 5.2.2 of Chapter 5). Since the proposed MIP formulation is expressed as a maximisation problem, the branch-and-bound or Benders decomposition solution algorithm attempts to reduce the lower bound of the MIP instance. The upper bound is estimated by Cplex by using a linear relaxation of the original MIP formulation, which tries to tighten the bound as part of the algorithmic solution process. An alternative to the LP relaxation approach was proposed in Section 5.3 of Chapter 5, where a Lagrangian relaxation MIP formulation is derived that uses a novel sub-gradient-based solution framework to find tighter bounds. This is done by solving the Lagrangian dual problem. In general, the proposed Lagrangian relaxation approach, in conjunction with the sub-gradient solution framework, was able to further reduce the optimality gap that is obtained as part of the output produced by the standard ALRSF (which comprises of steps $S = 0$, $S = 1$ and $S = 2$). Some improvements can, however, be considered as part of future research endeavours, which include:

1. Improving the sub-gradient algorithm's convergence speed by reducing the zigzagging effect that is experienced by the algorithm as a result of the Lagrangian multipliers zigzagging across ridges of the dual function. This relates to the non-differentiability of the dual function and may increase the time required for the sub-gradient algorithm to converge (Bragin, 2023). As part of future research initiatives, interleaved, surrogate or incremental sub-gradient methods may be considered as possible solutions to speed up convergence (Bragin, 2023).

2. Attempt to find Lagrangian multipliers that yield an improved approximation of the optimal solution to the original MIP formulation within a smaller time frame. There are multiple alternative solution algorithms (apart from the sub-gradient algorithm) that can be employed to solve the Lagrangian dual problem. One of these include the well-known Conic bundle algorithm. This is a cutting-plane-based algorithm which might improve convergence by choosing the direction of descent based on information collected throughout previous iterations. The Conic bundle approach may be an efficient alternative to the standard sub-gradient algorithm.

3. Improving the approach used to select the Big-M constant values at each iteration of the solution algorithm. This will allow for a tighter solution search space which might, in turn, reduce the number of iterations required to converge to a desired solution. Reducing the Big-M constants may also improve numeric stability and the convergence speed of the MIP solution process.

4. Develop a heuristic that is based on Lagrangian relaxation to find feasible solutions to the original MIP formulation. Try and integrate the Lagrangian relaxation heuristic with the regression parameter pruning heuristic in an attempt to reduce the execution time of the ALRSF.

The work presented in this thesis provides a sufficient foundation for multiple possible research avenues that can be undertaken with respect to the logistic regression best subset selection problem. The contributions made, along with the empirical results presented, will hopefully inspire further research into automated deterministic solution frameworks.

Bibliography

Acquah, H. D.-G. (2010). Comparison of akaike information criterion (aic) and bayesian information criterion (bic) in selection of an asymmetric price relationship.

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2015). Benders decomposition for production routing under demand uncertainty. *Operations Research*, 63(4):851–867.

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer.

Al-Ani, A. (2005). Feature subset selection using ant colony optimization. *International journal of computational intelligence*.

Al-Ani, A. (2007). Ant colony optimization for feature subset selection. *International Journal of Computer and Information Engineering*, 1(4):999–1002.

Antonakis, J. and Dietz, J. (2011). Looking for validity or testing it? the perils of stepwise regression, extreme-scores analysis, heteroscedasticity, and measurement error. *Personality and Individual Differences*, 50(3):409–415.

Arnerić, J., Škrabić, B., and Babić, Z. (2007). Maximization of the likelihood function in financial time series models. In *Proceedings of the International Scientific Conference on Contemporary Challenges of Economic Theory and Practice*, pages 1–12.

Bell, D. (2000). A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41(2):175–195.

Benders, J. (1962). Partitioning procedures for solving mixed-variables programming problems . *Numerische mathematik*, 4(1):238–252.

Bertsimas, D., King, A., and Mazumder, R. (2016). Best subset selection via a modern optimization lens. *The annals of Statistics*, 44(2):813–852.

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159.

Bragin, M. A. (2023). Lagrangian relaxation for mixed-integer linear programming: Importance, challenges, recent advancements, and opportunities. *arXiv preprint arXiv:2301.00573*.

Breheny, P. and Huang, J. (2011). Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The annals of applied statistics*, 5(1):232.

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics*, 37(4):373–384.

Burnham, K. P., Anderson, D. R., and Huyvaert, K. P. (2011). Aic model selection and multimodel inference in behavioral ecology: some background, observations, and comparisons. *Behavioral ecology and sociobiology*, 65(1):23–35.

Bursac, Z., Gauss, C. H., Williams, D. K., and Hosmer, D. W. (2008). Purposeful selection of variables in logistic regression. *Source code for biology and medicine*, 3(1):1–8.

Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection method. *Computational Optimization and Applications*, 40(1):16–28.

Chotikapanich, D. and Griffiths, W. (2001). On calculation of the extended gini coefficient. *Review of Income and Wealth*, 47(4):541–547.

Civitelli, E., Lapucci, M., Schoen, F., and Sortino, A. (2021). An effective procedure for feature subset selection in logistic regression based on information criteria. *Computational Optimization and Applications*, 80(1):1–32.

Clausen, J. (1999). Branch and bound algorithms-principles and examples. *Department of Computer Science, University of Copenhagen*, pages 1–30.

Costa, A. M. (2005). A survey on benders decomposition applied to fixed-charge network design problems. *Computers & operations research*, 32(6):1429–1450.

Costanza, M. C. and Afifi, A. (1979). Comparison of stopping rules in forward stepwise discriminant analysis. *Journal of the American Statistical Association*, 74(368):777–785.

Cramer, J. S. (2002). The origins of logistic regression.

Czepiel, S. A. (2002). Maximum likelihood estimation of logistic regression models: theory and implementation. *Available at czep. net/stat/mlelr. pdf*, 83.

Dedieu, A., Hazimeh, H., and Mazumder, R. (2021). Learning sparse classifiers: Continuous and mixed integer optimization perspectives. *Journal of Machine Learning Research*, 22(135):1–47.

Derksen, S. and Keselman, H. J. (1992). Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology*, 45(2):265–282.

Draper, N. R. and Smith, H. (1998). *Applied regression analysis*, volume 326. John Wiley & Sons.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of statistics*, 32(2):407–499.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.

Fellman, J. et al. (2012). Estimation of gini coefficients using lorenz curves. *Journal of Statistical and Econometric Methods*, 1(2):31–38.

Fisher, M. L. (1981). The lagrangian relaxation method for solving integer programming problems. *Management science*, 27(1):1–18.

Fisher, M. L. (1985). An applications oriented guide to lagrangian relaxation. *Interfaces*, 15(2):10–21.

Fonti, V. and Belitser, E. (2017). Feature selection using lasso. *VU Amsterdam research paper in business analytics*, 30:1–25.

Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007). Pathwise coordinate optimization. *The annals of applied statistics*, 1(2):302–332.

Fu, W. J. (1998). Penalized regressions: the bridge versus the lasso. *Journal of computational and graphical statistics*, 7(3):397–416.

Gasso, G. (2019). Logistic regression.

Gaudioso, M., Gorgone, E., Labbé, M., and Rodríguez-Chía, A. M. (2017). Lagrangian relaxation for svm feature selection. *Computers & Operations Research*, 87:137–145.

Geoffrion, A. M. (1974). Lagrangean relaxation for integer programming. In *Approaches to integer programming*, pages 82–114. Springer.

Gheyas, Iffat, A. and Smith, Leslie, S. (2009). Feature subset selection in large dimensionality domains. *Elsevier Pattern recognition*, 43(1):5–13.

Guignard, M. (2003). Lagrangean relaxation. *Top*, 11(2):151–200.

Hall, M. and Smith, L. (1998). Practical feature subset selection for machine learning. pages 181–191.

Hall, M. and Smith, L. (2011). Feature subset selection: A correlation based filter approach. *IEEJ Transactions on Electrical and Electronic Engineering*, 6(2):173–179.

Halsey, L. G. (2019). The reign of the p-value is over: what alternative analyses could we employ to fill the power vacuum? *Biology letters*, 15(5):20190174.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hazan, E. and Kakade, S. (2019). Revisiting the polyak step size. *arXiv preprint arXiv:1905.00313*.

Heinze, G., Wallisch, C., and Dunkler, D. (2018). Variable selection - a review and recommendations for the practicing statistician. *Biometric Journal*, 60(3):431–449.

Held, M. and Karp, R. M. (1970). The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6):1138–1162.

Herlands, W., De-Arteaga, M., Neill, D., and Dubrawski, A. (2015). Lass-0: sparse non-convex regression by local search. *arXiv preprint arXiv:1511.04402*.

Hu, Q., Yu, D., Liu, J., and Wu, C. (2008). Neighborhood rough set based heterogeneous feature subset selection. *Information sciences*, 178(18):3577–3594.

Hunter, D. R. and Li, R. (2005). Variable selection using mm algorithms. *Annals of statistics*, 33(4):1617.

Insolia, L., Kenny, A., Calovi, M., and Chiaromonte, F. (2021). Robust variable selection with optimality guarantees for high-dimensional logistic regression. *Stats*, 4(3):665–681.

Inza, I., Larranaga, P., Etxeberria, R., and Sierra, B. (2000). Feature subset selection by bayesian network-based optimization. *Artificial intelligence*, 123(1–2):157–184.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Kamiya, S., Miyashiro, R., and Takano, Y. (2019). Feature subset selection for the multinomial logit model via mixed-integer optimization. In *The 22nd International conference on artificial intelligence and statistic*, pages 1254–1263, Okinawa, Japan. PMLR.

Karagrigoriou, A., Mattheou, K., and Vonta, I. (2011). On asymptotic properties of aic variants with applications. *American Open Journal of Statistics*, 1(2):105–109.

Karegowda, Asha, G., Jayaram, M, A., and Manjunath, A, S. (2010). Feature subset selection problem using wrapper approach in supervised learning. *International journal of Computer applications*, 1(7):13–17.

Kleinbaum, D. G. and Klein, M. (2010). Introduction to logistic regression. In *Logistic regression*, pages 1–39. Springer.

Kohavi, R. and John, G. (1996). Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324.

Kutner, M., Nachtsheim, C., Neter, J., and Li, W. (2005). *Applied Linear Statistical Models*. 5th Ed. McGraw-Hill.

Lemaréchal, C. (2001). Lagrangian relaxation. In *Computational combinatorial optimization*, pages 112–156. Springer.

Li, B., Wang, Q., and Hu, J. (2011). Feature subset selection: A correlation-based svm filter approach. *IEEJ Transactions on Electrical and Electronic Engineering*, 6(2):173–179.

Liang, Y., Liu, C., Luan, X.-Z., Leung, K.-S., Chan, T.-M., Xu, Z.-B., and Zhang, H. (2013). Sparse logistic regression with a l1/2 penalty for gene selection in cancer classification. *BMC bioinformatics*, 14(1):1–12.

Liao, H. and Lynn, H. S. (2010). A survey of variable selection methods in two chinese epidemiology journals. *BMC medical research methodology*, 10(1):1–5.

Lopez, Felix, G., Torres, Garcia, M., Batista, Belen, M., Perez, Jose, M., and Moreno-Vega, M. (2004). Solving feature subset selection problem by a parallel scatter search. *Elsevier Science Direct*, 169(2006):477–489.

Lund, B. (2017). Logistic model selection with sas proc's logistic, hplogistic hp-genselect-paper aa02. In *2017 Midwest SAS Users Group Conference*, page 3, St. Louis, Missouri, USA. ACM.

Maalouf, M. (2011). Logistic regression in data analysis: an overview. *International Journal of Data Analysis Techniques and Strategies*, 3(3):281–299.

Mao, K. Z. (2004). Orthogonal forward selection and backward elimination algorithms for feature subset selection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(1):629–634.

Miller, A. (2002). *Subset selection in regression*. CRC Press: Washington.

Narendra, P. and Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *IEEE Transactions on computers*, 26(9):917–922.

Obrien, R. M. (2007). A caution regarding rules of thumb for variance inflation factors. *Quality & quantity*, 41:673–690.

Park, H. M. (2015). Hypothesis testing and statistical power of a test.

Park, M. Y. and Hastie, T. (2008). Penalized logistic regression for detecting gene interactions. *Biostatistics*, 9(1):30–50.

Pereira, J. M., Basto, M., and da Silva, A. F. (2016). The logistic lasso and ridge regression in predicting corporate failure. *Procedia Economics and Finance*, 39:634–641.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817.

SAS, I. (2005). *Predictive modeling using logistic regression course notes*. SAS Institute, Cary.

Sato, T., Takano, Y., Miyashiro, R., and Yoshise, A. (2016a). Feature subset selection for logistic regression via mixed integer optimization. *Computational Optimization and Applications*, 64(3):865–880.

Sato, T., Takano, Y., Miyashiro, R., and Yoshise, A. (2016b). Feature subset selection for logistic regression via mixed integer optimization. *Computational Statistics and Data Analysis*, 64(3):865–880.

Shahzad, F., Abid, F., Obaid, A. J., Kumar Rai, B., Ashraf, M., and Abdulbaqi, A. S. (2021). Forward stepwise logistic regression approach for determinants of hepatitis b & c

among hiv/aids patients. *International Journal of Nonlinear Analysis and Applications*, 12:1367–1396.

Shtatland, E., Kleinman, K., and Cain, E. (2003). Stepwise methods in using sas proc logistic and sas enterprise miner for prediction. sugi'28 proceeding, paper 258-28, cary, nc: Sas institute.

Smith, G. (2018). Step away from stepwise. *Journal of Big Data*, 5(1):1–12.

Song, Q., Ni, J., and Wang, G. (2011). A fast clustering-based feature subset selection algorithm for high dimensional data. *IEEE transactions on knowledge and data engineering*, 25(1):1–14.

Steyerberg, E., Eijkemans, M., and Habbema, J. (2001). Application of shrinkage techniques in logistic regression analysis: a case study. *Statistica Neerlandica*, 55(1):76–88.

Steyerberg, E. W., Eijkemans, M. J., and Habbema, J. D. F. (1999). Stepwise selection in small data sets: a simulation study of bias in logistic regression analysis. *Journal of clinical epidemiology*, 52(10):935–942.

Sun, T., Zhao, Q., and Luh, P. (2007). On the surrogate gradient algorithm for lagrangian relaxation. *Journal of optimization theory and applications*, 133(3):413–416.

Sun, Z., Bebis, G., and Miller, R. (2004). Object detection using feature subset selection. *Pattern recognition*, 37(11):2165–2176.

Takano, Y. and Miyashiro, R. (2020). Best subset selection via cross-validation criterion. *Springer*, 28(2):475–488.

Tan, F., Fu, X., Zhang, Y., and Bourgeois, A. (2008). A genetic algorithm-based method for feature subset selection. *Soft computing*, 12(2):111–120.

Taskın, Z. C. (2010). Benders decomposition. *Wiley Encyclopedia of Operations Research and Management Science. John Wiley & Sons, Malden (MA).*

Thiese, M. S., Ronna, B., and Ott, U. (2016). P value interpretations and considerations. *Journal of thoracic disease*, 8(9):E928.

Udriste, C., Constantinescu, M., Tevy, I., and Dogaru, O. (2015). Lagrange and wolf dualities in nonholonomic optimization. *arXiv preprint arXiv:1506.04574*.

van Niekerk, T., Venter, J. V., and Terblanche, S. E. (2023). An automated exact solution framework towards solving the logistic regression best subset selection problem. *South African Statistical Journal*, 57(2):89–129.

Varoquaux, G., Raamana, P. R., Engemann, D. A., Hoyos-Idrobo, A., Schwartz, Y., and Thirion, B. (2017). Assessing and tuning brain decoders: cross-validation, caveats, and guidelines. *NeuroImage*, 145:166–179.

Venter, J. v. (2020). *Variable selection in logistic regression using exact optimisation approaches*. PhD thesis, North-West University school of Science with Business Mathematics.

Vrieze, S. I. (2012). Model selection and psychological theory: a discussion of the differences between the akaike information criterion (aic) and the bayesian information criterion (bic). *Psychological methods*, 17(2):228.

Wah, Y. B., Ibrahim, N., Hamid, H. A., Abdul-Rahman, S., and Fong, S. (2018). Feature selection methods: Case of filter and wrapper approaches for maximising classification accuracy. *Pertanika Journal of Science & Technology*, 26(1).

Walter, S. and Tiemeier, H. (2009). Variable selection: current practice in epidemiological studies. *European journal of epidemiology*, 24(12):733–736.

Whittingham, M. J., Stephens, P. A., Bradbury, R. B., and Freckleton, R. P. (2006). Why do we still use stepwise modelling in ecology and behaviour? *Journal of animal ecology*, 75(5):1182–1189.

Wilkinson, L. (1979). Tests of significance in stepwise regression. *Psychological bulletin*, 86(1):168.

Xie, H. and Huang, J. (2009). Scad-penalized regression in high-dimensional partially linear models. *The Annals of Statistics*, 37(2):673–696.

Xu, Z., Zhang, H., Wang, Y., Chang, X., and Liang, Y. (2010). L1/2 regularization. *Science China Information Sciences*, 53(6):1159–1169.

Yanagihara, H., Kamo, K.-i., Imori, S., and Satoh, K. (2012). Bias-corrected aic for selecting variables in multinomial logistic regression models. *Linear algebra and its applications*, 436(11):4329–4341.

Yang, J. and Honavar, V. (1998). Feature subset selection using a genetic algorithm. *Springer*, pages 117–136.

Ypma, T. J. (1995). Historical development of the newton–raphson method. *SIAM review*, 37(4):531–551.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942.

Zhang, S., Zhang, L., Qiu, K., Lu, Y., and Cai, B. (2015). Variable selection in logistic regression model. *Chinese Journal of Electronics*, 24(4):813–817.

Zhao, X., Luh, P. B., and Wang, J. (1999). Surrogate gradient algorithm for lagrangian relaxation. *Journal of optimization Theory and Applications*, 100(3):699–712.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320.

Zou, H. and Li, R. (2008). One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of statistics*, 36(4):1509–1533.

Appendix A

Peer-reviewed journal article published in the South African Statistical Journal (SASJ)

South African Statistical Journal
Vol. XX, No. X, 1–41
© 2023 South African Statistical Association

An automated exact solution framework towards solving the logistic regression best subset selection problem

Thomas K. van Niekerk[1], *Jacques V. Venter*[2] *and Stephanus E. Terblanche*[1]

[1]School of Industrial Engineering, North-West University, Potchefstroom, South Africa
[2]Centre for Business Mathematics and Informatics, North-West University, Potchefstroom, South Africa

An automated logistic regression solution framework (ALRSF) is proposed to solve a mixed integer programming (MIP) formulation of the well known logistic regression best subset selection problem. The solution framework firstly determines the optimal number of independent variables that should be included in the model using an automated cardinality parameter selection procedure. The cardinality parameter dictates the size of the subset of variables and can be problem-specific. A novel regression parameter fixing heuristic that utilises a Benders decomposition algorithm is applied to prune the solution search space such that the optimal regression parameter values are found faster. An optimality gap is subsequently calculated to quantify the quality of the final regression model by considering the distance between the best possible log-likelihood value and a log-likelihood value that is calculated using the current parameter values. Attempts are then made to reduce the optimality gap by adjusting regression parameter values. The ALRSF serves as a holistic variable selection framework that enables the user to consider larger datasets when solving the best subset selection logistic regression problem by significantly reducing the memory requirements associated with its mixed integer programming formulation. Furthermore, the automated framework requires minimal user intervention during model training and hyperparameter tuning. Improvements in quality of the final model (when considering both the optimality gap and computing resources required to achieve a result) are observed when the ALRSF is applied to well-known real-world UCI machine learning datasets.

Keywords: Best Subset Selection, Independent Variable Selection, Logistic Regression, Mixed Integer Programming

1. Introduction

Binary classification problems play an important role within the broader field of statistical modelling and can be solved by using algorithms such as support vector machines, neural networks, decision tree based methods, k-nearest-neighbours and logistic regression (Civitelli et al., 2021). Logistic regression is by far one of the most popular approaches which is supported by a wealth of literature and research (Civitelli et al., 2021). In general, logistic regression algorithms are straightforward to

Corresponding author: Thomas K. van Niekerk (thomasvn67@gmail.com)
MSC2020 subject classifications: 90-XX, 62-XX

implement, can be applied to relatively large machine learning problems and provide a measure of interpretability (Civitelli et al., 2021; Stoltzfus, 2011).

An important area of research (with reference to solving binary classification problems) is finding the optimal subset of independent variables from a set of independent variables $j \in p = \{1, 2, ..., |p|\}$ and observations $i \in n = \{1, 2, ..., |n|\}$ to include in the final model during the training phase. Let X_i denote a $p \times 1$ input vector of independent variable values for the ith observation in a dataset with n records, where the jth entry of X_i correspond to the jth variable. Furthermore, let $Y_i \in \{0, 1\}$ be a binary dependent or target variable that follows a Bernoulli distribution where $P(Y_i = 1|X_i) = (1 \quad \exp(-\beta^T X_i))^{-1}$. Within a regression model setting, the independent variable entries in X_i represent a vector of p predictors that are used as inputs in a machine learning model in an attempt to predict the dependent variable Y_i. It is assumed that each observation (X_i, Y_i) is independently sampled from the same population. The selected subset of input variables should accurately capture the nature of the problem instance while still allowing for adequate generalisation performance. Best subset selection (in its most fundamental form) describes a process that attempts to select the most relevant and statistically significant set of independent variables by optimising some objective function $f(\beta; \mathbf{X}, \mathbf{Y})$ with respect to a vector of regression parameters $\beta \in \mathbb{R}^p$, where β_j denotes the jth regression parameter associated with the jth input variable (with β_0 being the intercept term). Mathematically, the problem can be defined as

$$\max_{\beta} f(\beta; \mathbf{X}, \mathbf{Y}), \tag{1}$$

or

$$\min_{\beta} f(\beta; \mathbf{X}, \mathbf{Y}), \tag{2}$$

subject to

$$||\beta||_0 \leq c, \tag{3}$$

where $||.||_0$ is the L_0-norm, $||\beta||_0 = \sum_{j \in p} I(\beta_j > 0)$ and (3) is known as a cardinality selection constraint that allows the regression model to contain at most c of the p available independent variables (Zhang et al., 2018; Venter, 2020). The composition of $f(\beta; \mathbf{X}, \mathbf{Y})$ depends on the type of regression algorithm that is applied. In the case of logistic regression, the log-likelihood function is typically used as the objective function $f(\beta; \mathbf{X}, \mathbf{Y})$. Since the log-likelihood function is concave, the objective function in (1) can be modified according to

$$\max_{\beta} f(\beta; \mathbf{X}, \mathbf{Y}) = \log(L(Y_1, ..., Y_n)) = \sum_{i \in n} Y_i(\beta^T X_i) - \sum_{i \in n} \log(1 \quad \exp(\beta^T X_i)), \tag{4}$$

where $\beta^T X_i$ is equal to $\sum_{j \in p} \beta_j X_{ij}$. The choice between (1) or (2) depends on the objective function being used. As an alternative to the log-likelihood function, the modeller may employ the logistic loss function, which is defined as

$$\min_{\beta} f(\beta; \mathbf{X}, \mathbf{Y}) = \sum_{i \in n} \log(1 \quad \exp(-Y_i(\beta^T X_i)). \tag{5}$$

Since the log-likelihood is concave, the negative of the log-likelihood (logistic loss function) will

be convex. This explains why maximisation of (4) and minimisation of (5) will yield identical results. Different formulations of the best subset selection problem make use of either the log-likelihood or the logistic loss objective function.

It is well known that training a logistic regression classifier on too many independent variables may result in the model overfitting on the training set, which in turn degrades the predictive performance of the final selected model. This is especially true when the number of independent variables p greatly exceeds the number of training observations n such that $p > n$ (Naganuma et al., 2021). Numerous studies have attempted to solve (to optimality) the logistic regression best subset selection problem by using a variation of three main MIP logistic regression formulations that are commonly found in the literature. The first approach attempts to find the optimal subset of variables by minimising a logistic loss objective function that contains a regularisation term, such as the L_0-norm, L_1-norm (lasso) or L_2-norm (ridge regression) (Dedieu et al., 2021; Insolia et al., 2021; Kamiya et al., 2019; Takano and Miyashiro, 2020). A second approach involves a piecewise linear approximation of the non-linear logistic loss function, where the objective function is adjusted such that the best model is chosen based on a problem-specific information criterion, like the Akaike information criterion (AIC) or Bayesian information criterion (BIC) (Sato et al., 2016; Civitelli et al., 2021). Finally, we consider an alternative approach proposed by Venter (2020), who, similar to Sato et al. (2016), also makes use of a piecewise linear approximation of the logistic regression objective function. The difference, however, is that the use of an approximated log-likelihood by Venter (2020) leads to an underestimation of the optimal log-likelihood, while an overestimation is observed because of the logistic loss used by Sato et al. (2016). Furthermore, a strict and explicit cardinality constraint (instead of some information criterion) limits the number of variables included in the final model(s) produced by Venter (2020). These overarching formulations and the relevant underlying theory will be discussed in Section 2. A detailed account of the limitations associated with each approach will be provided to further highlight the contributions of the proposed automated logistic regression solution framework presented herein.

One of the main drawbacks of deterministic MIP solution frameworks relates to the computational burden and resource constraints imposed by these formulations, which in turn limits the size and dimensionality of the prediction problem that can be considered. According to Miller (2002), best subset selection MIP approaches are exponential-time procedures, which implies that an exponential increase in execution time is observed alongside an increase in the number of independent variables. In fact, best subset selection problems are NP-hard by nature (Natarajan, 1995; Venter, 2020). Some authors note that attempts to perform best subset selection procedures using commercially available statistical software become intractable when the input dataset contains approximately 40 - 75 independent variables (Lund, 2017).

The framework proposed in this paper mainly builds on the logistic regression MIP formulation presented by Venter (2020) because of the favourable statistical characteristics that it exhibits. With the above in mind, we set out to develop a novel automated solution framework that can be used to solve large-scale best subset selection logistic regression problems that are formulated as MIPs. The framework in question mainly builds on the formulation proposed by Venter (2020) because of its favourable statistical attributes. Additionally, substantial attempts are made to address some of the limitations inherent to logistic regression MIP problems which are commonly found when dealing with best subset selection. Firstly, the automated solution framework (dubbed the ALRSF) produces

interpretable and parsimonious models that generalise well on unseen data by avoiding the use of regularisation terms and biased regression parameters. This is done by incorporating an explicit cardinality constraint into the formulation. Secondly, the ALRSF requires minimal user intervention (unlike some of the other formulations found in literature) by automatically determining the most appropriate cardinality parameter and iteratively adjusting the parameter search space during model training. As such, the regression parameters of the model rely more on the underlying patterns that are present in the data and less on input from the model developer. Finally, and perhaps most importantly, the ALRSF exhibits significant improvements in computational memory requirements when presented with best subset selection problems. These improvements enable the user to consider larger datasets that would otherwise be intractable when solved with the MIP approaches mentioned earlier.

The remainder of this paper can be subdivided into six main sections. Section 2 provides a background on best subset selection and related work, accompanied by appropriate theoretical considerations that underpin the proposed ALRSF. The contribution of the ALRSF to existing literature is also presented. Sections 3 and 4 introduce the reader to the proposed best subset selection MIP formulation with relevant modifications that allow it to be used within an automated logistic regression setting. The logic of the ALRSF is then summarised in Section 4 followed by model verification in Section 5. Empirical results are presented in Section 6 where predictive models are applied to three real-world datasets in order to evaluate the performance of the proposed ALRSF and to compare it against existing best subset MIP approaches. Section 6 further highlights the contribution of the ALRSF to the intersecting fields of variable selection and computational efficiency when dealing with sizeable datasets. Concluding remarks and recommendations with respect to future research are discussed in Section 7.

2. Background and rationale

In this section, we elaborate on the concept of best subset selection within a MIP problem setting where regularisation terms, information criteria or explicit cardinality parameters are incorporated into the model objective function to select the optimal subset of variables from the set of p predictors. Firstly, MIP formulations that rely on regularisation techniques will be considered. In its most simplistic form, regularisation MIP formulations can be defined as[1]

$$\min_{\beta} \quad \frac{1}{n} f(\beta; \mathbf{X}, \mathbf{Y}) \quad \lambda ||\beta||_{q_2}^{q_1}, \tag{6}$$

where $||.||_{q_2}^{q_1}$ controls the type of regularisation used for subset selection. When $q_1 = q_2 = 1$, the popular lasso regression approach is followed. Alternatively, setting $q_1 = q_2 = 2$ forces the model to apply a ridge regression penalty. Unlike lasso and ridge regression (which are not NP-hard), we can specify a strict penalty by setting $q_1 = 1$ and $q_2 = 0$, which will use the L_0-norm and perform best subset selection. Incorporating a combination of the aforementioned regularisation terms into a single objective function can also be considered in an attempt to overcome the limitations associated

[1] Note that the objective function is minimised when the logistic loss is used. Alternatively, the objective function will be maximised when the log-likelihood is employed, with the "+" sign in front of the regularisation term (the second term of the objective function) changed to a "-" sign.

with each individual approach (see Dedieu et al. (2021) for more details). The regularisation-based MIP formulation proposed by Dedieu et al. (2021) utilises L_0-L_q-norms as penalty terms and is shown in (7), where $\lambda_0 \geq 0$ controls the number of non-zero regression parameters and enforces model sparsity, while $\lambda_q \geq 0$ controls the shrinkage of model parameters.

$$\min_{\beta} \quad \frac{1}{n} f(\beta; \mathbf{X}, \mathbf{Y}) \quad \lambda_0 ||\beta||_0 \quad \lambda_1 ||\beta||_1 \quad \lambda_2 ||\beta||_2^2. \tag{7}$$

Each regularisation term can be included/excluded based on the value(s) selected for q_1, q_2, λ_0, λ_1 and λ_2. Dedieu et al. (2021) reports that the function $f(\beta; \mathbf{X}, \mathbf{Y})$ is interchangeable between the logistic loss function (defined in Section 1) or the hinge loss function. A detailed overview of each type of loss function can be found in the author's work. From Equation (7) we can group the logistic loss function, lasso and ridge regression terms into a single term that can be written as

$$h(\beta; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) = \frac{1}{n} f(\beta; \mathbf{X}, \mathbf{Y}) \quad \lambda_1 ||\beta||_1 \quad \lambda_2 ||\beta||_2^2. \tag{8}$$

Substituting (8) into (7) leads to

$$\min_{\beta, z} \quad h(\beta; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) \quad \lambda_0 \sum_{j \in p} z_j, \tag{9}$$

subject to

$$|\beta_j| \leq M z_j \quad \forall j \in p, \tag{10}$$

$$z_j \in \{0, 1\} \quad \forall j \in p. \tag{11}$$

The formulation in (9)–(11) is a special case of the regularised regression model presented in (6) and allows for variable selection by optimising some inherent cardinality parameter based on the z_j decision variable. When $z_j = 0$, the jth independent variable is excluded from the final subset by setting $\beta_j = 0$. Alternatively, $|\beta_j|$ is allowed to take on any value in the range $[0, M]$ when $z_j = 1$. The constant M in constraint (10) governs the range of values that the regression parameters are allowed to take on and should be sufficiently large (ideally, in a perfect world where we have unlimited computational power M should tend to infinity). Given that the L_0-norm describes the number of non-zero regression parameters (the number of parameters where $\beta_j \neq 0$), we can replace $\lambda_0 ||\beta_0||$ with $\lambda_0 \sum_{j \in p} z_j$ to perform the same task. A tighter formulation of the optimisation problem in (8)–(11) was proposed by Dedieu et al. (2021) and is summarised in (12)–(17):

$$\min_{\beta, z, \omega} \quad h(\omega; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) \quad \lambda_0 \sum_{j \in p} z_j, \tag{12}$$

subject to

$$|\beta_j| \leq M z_j, \quad j \in A', \tag{13}$$

$$\omega = \beta_j^t - \sum_{j \in A} e_j \beta_j^t (1 - z_j) \quad \sum_{j \in A'} e_j \beta_j, \tag{14}$$

$$\sum_{j \in A} z_j \geq |A| - m, \tag{15}$$

$$\sum_{j \in A'} z_j \leq m, \tag{16}$$

$$z_j \in \{0, 1\} \quad \forall j \in p. \tag{17}$$

Dedieu et al. (2021) makes use of a two-step iterative local search algorithm to first prune the solution search space. The best solution discovered as part of the local search algorithm is then used as an input to the tightened MIP formulation presented in (12)–(17) in an attempt to either improve the best integer solution or provide an optimality guarantee. The first algorithm used (as part of the initial two-step local search algorithm) is a coordinate descent approach followed by a local combinatorial search algorithm. The reader is referred to Dedieu et al. (2021) for a detailed explanation of this iterative algorithmic approach. Finally, the tightened MIP formulation can be used to try and improve on the local solutions discovered while also attempting to provide an optimality guarantee. Let $A \subset p$ denote a subset of variables from all available independent variables p with the complement of A denoted as A'. Then the support set can be defined as $A = \text{Supp}(\beta)$ where $\beta \neq 0$. Let $A^1 \subset A$ and $A^2 \subset A'$ denote two sets with size of at most m (the value of m will be discussed shortly). Let β^t denote the vector of regression parameters that were generated at iteration t of the local search algorithm (the combination of the coordinate descent and combinatorial search approaches)[2]. As part of the combinatorial search step a $p \times p$ matrix B^A is defined, where the jth row of B^A is equal to $\sum_{j \in p} e_j$ if j element of A and zero otherwise. This means that for any $\beta \in \mathbb{R}^p$, $(B^A\beta)_j = \beta_j$ when $j \in A$ and $(B^A\beta)_j = 0$ when $j \notin A$. This swap-and-solve logic is captured by the constraints in (14)–(17) and is facilitated by the binary decision variables $z_j, j = 1, ..., p$. Particularly when $z_j = 0 \ \forall j \in A$ it means that the jth variable in A^1 should be removed from support set A. Alternatively, when $z_j = 1 \ \forall j \in A'$ it means that the jth variable in A^2 should be added to the support set A. The size of m controls the number of variables that can be considered for inclusion into the support A. A large value for m will generally result in extended model execution times.

Dedieu et al. (2021) go on to develop another MIP formulation which is solved using an integrability generation algorithm. Let A represent the support and A' the complement of A. Once again the heuristic-based solution algorithm (which is a combination of coordinate descent with local combinatorial search) is used to prune the solution search space. The support A is then initialised with the variables identified by the initial heuristic search algorithm. The proposed MIP formulation can be defined as

$$\min_{\beta, z} \quad h(\beta; \mathbf{X}, \mathbf{Y}, \lambda_1, \lambda_2) \quad \lambda_0 \sum_{j \in p} z_j, \tag{18}$$

subject to

$$|\beta_j| \leq M z_j, \quad \forall j \in p, \tag{19}$$

$$z_j \in [0, 1], \quad \forall j \in A', \tag{20}$$

$$z_j \in \{0, 1\}, \quad \forall j \in A, \tag{21}$$

[2]For the remainder of this paper, the reader should note that an uppercase superscript T (e.g. D^T) refers to the transpose of a vector/matrix, whereas a lowercase superscript t (e.g. β^t) refers to the tth iteration.

where $z_j \in A'$ in (20) is allowed to be continuous and take on any value between zero and one, while integrality constraints are placed on $z_j \in A$ in (21). Allow β^t and z^t to denote the regression parameters and binary selection variables at iteration t. At iteration t 1 update $A = A \cup \{j|z_j^t \neq 0, j \in A'\}$ and then solve the MIP in (18)–(21) with warm-starting functionality of the MIP solver activated. This means that the solution at iteration t is used as starting point at iteration t 1. Since the MIP is a relaxation of (9)–(11), it provides a lower bound to the original problem i.e. logistic loss that is produced by the regression model (9)–(11) will be greater than or equal to the logistic loss obtained when using (18)–(21). As such, at each iteration t, the MIP in (18)–(21) aims to improve the MIP lower bound while also allowing an optimality gap to be calculated. The solution algorithm proposed by Dedieu et al. (2021) is capable of solving high-dimensional problems with a variable set of up to 50 000 independent variables as a result of the novel integrability generation logic applied. In general, the empirical results presented by the authors focused on solving sparse classification problems where $n \leq 1000$.

The lasso or L_1-norm regularisation has multiple attractive properties, some of which include its computational efficiency and scalability in the case of large-scale problem instances (Bertsimas et al., 2016; Hastie et al., 2020). One of its most useful characteristics is its ability to act as a true independent variable selection algorithm by setting some regression parameter estimates exactly equal to zero. Lasso regression does, however, introduce model bias by shrinking parameter estimates towards zero. This is especially true when independent variables in the design matrix have different measurement scales, where the betas associated with variables that have large true underlying regression parameters experience a greater degree of shrinkage (and, therefore, bias) (Bertsimas et al., 2016). In turn, the resulting shrinkage can adversely impact statistical inferences that are carried out when assessing the model, such as the evaluation of log-odds ratios (Heinze et al., 2018). Ridge regression suffers from the same drawbacks as lasso, but has the added disadvantage of producing larger models that contain many independent variables. This is because ridge regression only shrinks the regression parameters without setting them exactly equal to zero (Venter, 2020). As such, ridge regression should not be considered a true subset selection technique. Parameter bias as a result of regularisation can, however, be alleviated by performing a second round of model fitting, where a logistic regression model that only uses the variables identified by lasso is considered for inclusion. During this second round, no variable selection is performed (i.e. no regularisation terms are present in the model) and the goal is simply to generate improved parameter estimates with less bias for the purpose of statistical inference.

Regularisation techniques usually employ a penalisation parameter λ to control the magnitude of regularisation. The choice of this penalisation parameter (when manually selected) is subjective in nature and leads to a trade-off between variance and bias (Venter, 2020). The subjective nature of manually selecting the penalisation parameter can, however, be mitigated by an automated approach that uses a combination of cross-validation and grid search. When such an approach is utilised, a predefined grid of penalisation parameters is considered during various iterations of model fitting, where the most appropriate parameter value that corresponds to the best model performance is selected. Similarly, Bertsimas et al. (2016) proposes a heuristic-based approach to generate a regularisation path for the penalty parameters. It should be noted, however, that cross-validation and the approach proposed by Bertsimas et al. (2016) are heuristics by nature and as such can not provide an optimality guarantee with respect to the quality of the final model. This is because the complete

solution search space over the full range of penelisation parameter values cannot be considered simultaneously, since the optimality gap yielded by (6), (9) or (12) is specific to some constant value of λ. The optimality gap (which is used to provide an optimality guarantee) can be defined according to

$$\text{optimality gap} = \frac{bb - bi}{|bi| \quad \alpha},$$

where bb denotes the best bound of the MIP obtained and bi the best integer solution. An arbitrary small value α is added to the denominator to prevent the gap from being undefined when $|bi|$ is zero. The best bound in a maximisation problem can be interpreted as the bound obtained by a linear programming (LP) relaxation of the original problem. This means that all integer or binary variables are relaxed and allowed to take on any real value, which reduces the number of constraints that the problem needs to adhere to. The best integer solution is obtained by means of a branch-and-bound algorithm. The branch-and-bound algorithm is designed to solve combinatorial optimisation problems and a detailed overview of the algorithm is available in Section 4.2.2 of Venter (2020).

An alternative to variable selection using regularisation terms is rooted in the concept of information theory. Some of the most well-known information criteria include AIC and BIC. An IC-based subset selection problem that utilises the AIC solves

$$\min_{\beta,z} \quad 2f(\beta; \mathbf{X}, \mathbf{Y}) \quad 2(\sum_{j \in p} z_j \quad 1), \tag{22}$$

whereas a model that utilises the BIC is fitted by solving

$$\min_{\beta,z} \quad 2f(\beta; \mathbf{X}, \mathbf{Y}) \quad \log(|n|)(\sum_{j \in p} z_j \quad 1). \tag{23}$$

Using (22) and (23), Sato et al. (2016) proposes a logistic regression model that minimises a piecewise linear approximation of the logistic loss function while simultaneously performing subset selection via the inclusion of the AIC or BIC in the linearised objective function. Let γ_k denote a set of symmetric grid values, where $k \in K = \{1, 2, ..., |K|\}$. The authors approximate the non-linear logistic loss function in (5) by creating a series of linear functions that track the logistic loss curve from below and cut away non-feasible regions of the solution space. This relates to the pointwise maximisation of a set of linear inequalities which can be written as

$$\max\{f'(\gamma_k)(Y_i(X_i^T\beta) \quad f(\gamma_k)\} \quad \forall k = 1, ..., K. \tag{24}$$

These linear inequalities can be reformulated as a minimisation problem according to

$$\min\{\varphi | \varphi \geq f'(\gamma_k)(Y_i(X_i^T\beta) \quad f(\gamma_k)\} \quad \forall k = 1, ..., K. \tag{25}$$

The methodology proposed by Sato et al. (2016) is summarised in Figure 1 with the dashed lines representing the piecewise linear approximations of the logistic loss function. Since the approximation is below the logistic loss function, it can be described as an overestimation of the log-likelihood function, since the negative of the logistic loss yields the log-likelihood. The number

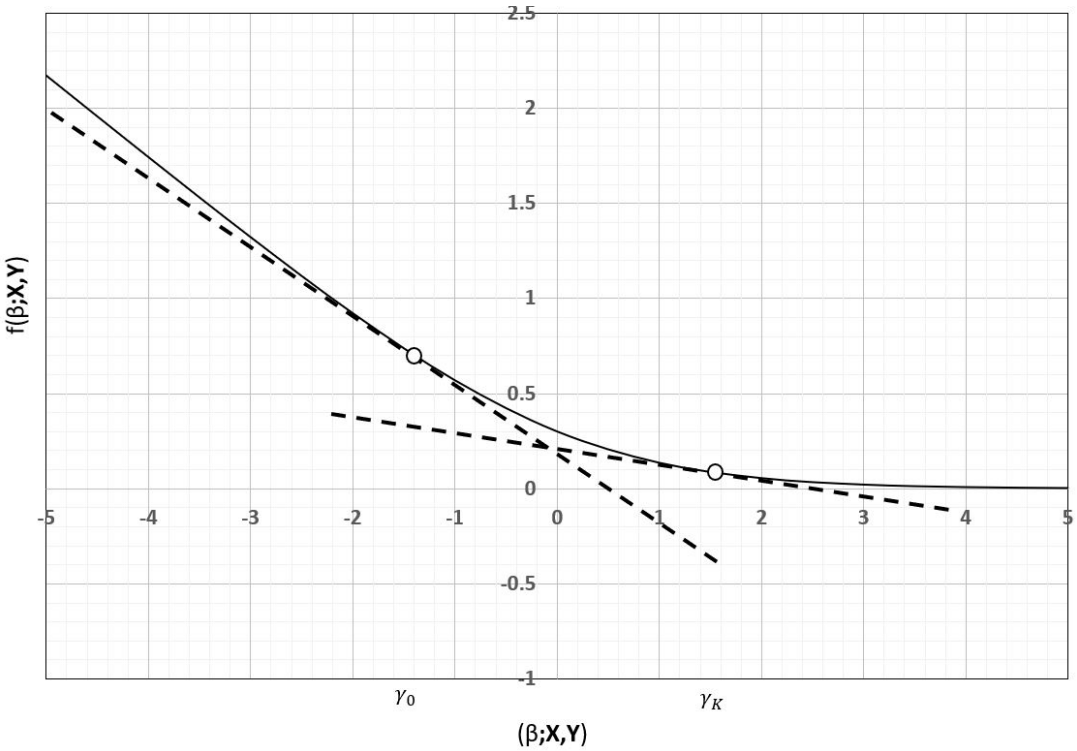


Figure 1. Piecewise linear overestimation of the logistic loss function

of tangent lines is defined by K (in Figure 1, $K = 2$ tangent lines are used). Larger numbers of linear splines result in an approximation that is closer to the non-linear logistic loss function.

The proposed piecewise linear approximation approach by Sato et al. (2016) can formally be defined as

$$\min_{\beta,z,\varphi} \quad 2\sum_{i\in n}\varphi_i \quad I(\sum_{j\in p} z_j \quad 1). \tag{26}$$

subject to

$$\varphi_i \geq f'(\gamma_k)(Y_i(X_i^T\beta) - \gamma_k) \quad f(\gamma_k) \quad \forall k \in K; \forall i \in n, \tag{27}$$

$$z_j = 0 \rightarrow \beta_j = 0 \quad \forall j \in p, \tag{28}$$

$$z_j \in \{0, 1\} \quad \forall j \in p, \tag{29}$$

where (27) represents a piecewise linear estimation of the logistic loss function as discussed in detail by Venter (2020) and Sato et al. (2016). Note that I in (26) does not denote an indicator function, but instead represents the information criterion that is applied during model fitting. When $I = 2$, the objective function in (22) is found and the AIC is used. Alternatively, when $I = \log(\cdot)$, the objective function (23) is used, which implies that the model is optimised with respect to the BIC.[3] The number of tangent lines K determines the accuracy of the linear approximation and, therefore, a trade-off exists between approximation accuracy and execution time (Sato et al., 2016). Variable

[3]It is important to note that I, used as part of the MIP formulation presented in (26)–(29), is a constant that facilitates switching between the use of either the Akaike or Bayesian information criterion. It does not represent an indicator function.

subset selection is facilitated by the selection decision variable z_j by means of (28) which is referred to as an indicator constraint or SOS1 (special ordered set), which implies that only one element in the set $\{1 - z_j, \beta_j\} \forall j \in p$ can be non-zero. This means that β_j should be set equal to exactly zero when $z_j = 0$ and non-zero otherwise. Constraint (28) is an alternative to a big-M formulation which can equivalently be defined as $\sum_{j \in p} |\beta_j| \leq M z_j$. It is important to note, however, that empirical evidence suggests that variable subset selection by means of minimising some information criterion (e.g. AIC or BIC) tends to favour larger models that are less parsimonious (Venter, 2020; Kutner et al., 2005).

Finally, we consider the piecewise linear approximation formulation for the logistic regression best subset selection problem that is proposed by Venter (2020). This formulation differs from that of Sato et al. (2016) in two noticeable areas, namely[4]

1. an explicit cardinality constraint is used instead of some information criterion to facilitate variable selection; and

2. instead of minimising the logistic loss (which results in an overestimation of the log-likelihood), a linear approximation of the log-likelihood is maximised (which results in an underestimation of the log-likelihood).

By critically evaluating the log-likelihood function defined in (4) it is clear that the first term is already linear. As such, only the second term $\sum_{i \in n} \log(1 \quad \exp(\beta^T X_i))$ needs to be linearised. The non-linear section is linearised by using $\Lambda_{ik} \in \mathbb{R}^{\{0,1\}}$ as an interval selection variable which relates the linearised grid point splines γ_k to the continuous estimates of $\beta^T X_i$. In other words, the decision variable Λ_{ik} identifies the appropriate line segment between two splines $[\gamma_k, \gamma_{k\ 1}]$ (used to approximate the log-likelihood function) that corresponds to the log-likelihood function value $f(\beta; \mathbf{X}, \mathbf{Y})$. The non-linear term of the log-likelihood function is therefore estimated by K linear line segments that connect at grid points γ_k, where $\min(\gamma_k) = -\Gamma$ and $\max(\gamma_k) = \Gamma$. The decision variable Λ_{ik} can alternatively be thought of as an interpolation variable. Figure 2 provides a visual representation of the piecewise linear underestimation of the log-likelihood function, where the dashed lines represent the piecewise linear approximation proposed by Venter (2020) and the solid black line denotes the non-linear log-likelihood.

The piecewise linear approximation of the log-likelihood attempts to fit a logistic regression model while simultaneously performing best subset selection by solving

$$\max_{\beta, \Lambda, z} \quad \sum_{i \in n} \sum_{k \in K} (\gamma_k Y_i - \log[1 \quad \exp(\gamma_k)]) \Lambda_{ik}, \tag{30}$$

subject to

$$\sum_{k \in K} \Lambda_{ik} \gamma_k = \beta^T X_i \quad \forall i \in n, \tag{31}$$

[4] From Figure 1, observe that the optimal solution is found where the two lines intersect. This means that the linear approximation of the logistic loss function underestimates the non-linear function (the solid line in Figure 1). However, given that $-f(\beta; \mathbf{X}, \mathbf{Y})$ is the logistic loss function and $f(\beta; \mathbf{X}, \mathbf{Y})$ the log-likelihood, we note that an underestimation of the logistic loss function results in an overestimation of the log-likelihood. The opposite is true for the formulation proposed by Venter (2020).

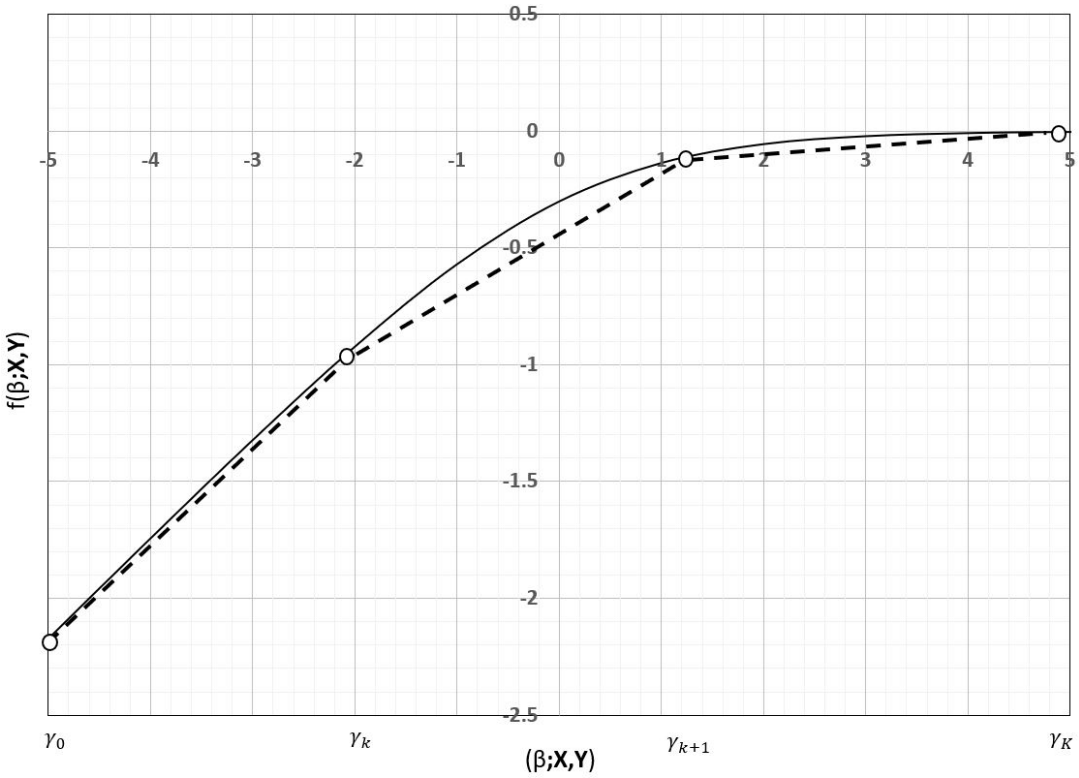


Figure 2. Piecewise linear underestimation of the log-likelihood function

$$\sum_{k \in K} \Lambda_{ik} = 1 \quad \forall i \in n, \tag{32}$$

$$0 \leq \Lambda_{ik} \leq 1 \quad \forall k \in K; \forall i \in n, \tag{33}$$

$$|\beta_j| \leq M z_j \quad \forall j \in p, \tag{34}$$

$$\sum_{j \in p} z_j \leq c. \tag{35}$$

Constraint (32) guides the selection of the appropriate linearised line segments in conjunction with (31) and (33) such that the log-likelihood is maximised. Constraints (34) and (35) are jointly referred to as cardinality selection constraints such that (35) ensures that a maximum of c independent variables are included in the model, while (34) allows the regression parameter $\beta_j \ \forall j \in p$ to default to 0 when $z_j = 0$ and range between $[-M, M]$ when $z_j = 1$. By using explicit cardinality constraints, the shrinkage effect (and, therefore the bias) introduced by regularisation is no longer present and the computational burden associated with the determination of the penalty term (e.g. cross-validation) is eliminated. In contrast to regularisation approaches, the MIP formulation proposed by Venter (2020) does not influence the interpretability of the regression parameters (Heinze et al., 2018). However, in all fairness, the proposed MIP formulation does require the specification of a cardinality value c that is used to limit the number of independent variables that can enter the model. In contrast to regularisation approaches, all possible variable combinations are considered for a specific cardinality value c, which means that the best subset MIP formulation can produce an optimality

gap and, therefore, a true measure of optimality (Hastie et al., 2020). This means that changing the value of c will only influence the number of independent variables included in the final subset and the final model performance, but (theoretically) for any given c, the best possible model is always obtained (the same cannot be said for regularised approaches or approaches that make use of p-values). Furthermore, an added benefit of the best subset MIP formulation is the possibility of explicitly controlling the sparsity of the model through c. Alternatively, no clear relationship exists between c and the penalty term λ that is used in regularisation approaches (Bertsimas et al., 2016).

Unfortunately, the approach in (30)–(35) requires some manual intervention for successful execution. The end-points of the grid $[-\Gamma, \Gamma]$ that describe the linearised approximation of the log-likelihood need to be manually calculated. This can be achieved by solving an LP relaxation of the original MIP problem formulation proposed by Venter (2020) that excludes all cardinality constraints. The linearised piecewise extreme points are then derived from the initial solution in a way which ensures that the complete search space is considered. According to the author, this type of manual intervention is undesirable if the objective is to automate the process of model fitting such that the optimal search space is determined by the algorithm and not the user. A detailed approach for finding the extreme points of the grid used to linearise the log-likelihood can be found in Venter (2020).

For the remainder of this paper, we use the formulation proposed by Venter (2020) to design our automated best subset logistic regression framework (our ALRSF). We address some of the drawbacks (as mentioned previously) associated with the model in (30)–(35) while attempting to automate the model fitting process and reduce the computational burden of best subset selection as much as possible.

3. Proposed MIP formulation

The proposed ALSRF combines the concept of a regression parameter fixing heuristic and Benders decomposition to allow for an improved deterministic solution framework that solves the best subset selection MIP problem within acceptable time frames while providing a guarantee of optimality. The solution framework is made up of three distinct steps which include the cardinality parameter selection step (automatically determining the most appropriate number of variables to include in the model), search space pruning step (automatically reducing the MIP search space that contains the range of possible values which regression parameters can take on in order to reduce run times) and an optimality gap calculation step (in order to provide a guarantee on the quality and optimality of the final regression model). This section presents the reader with the modifications made to the MIP formulation in (30)–(35), followed by a detailed exposition of the proposed novel ALRSF in Section 4.

First, let p' denote a subset of p such that $p' \subseteq p$. The subset p' can be defined as a set of independent variables for which the variance inflation factor (VIF) is larger than some threshold L. The subset p' therefore contains the indices of all the variables in the original set p that need to be excluded by the ALRSF based on the VIF criteria. Incorporating a VIF threshold into the ALRSF (which has not been done in any of the logistic regression best subset selection MIP formulations presented in Section 2) serves two purposes:

1. Highly correlated variables are barred from being included in the final model in an effort

to reduce multicollinearity. It is well-known that correlated predictors add no additional information to the model (as potential signals are already contained within the other inputs) and result in unstable parameter estimates.

2. By eliminating some of the variables from the start, the logistic regression problem instance now has to consider fewer variables in its search for the best variable subset, which can potentially result in shorter execution times and fewer computational resources being consumed.

Next, allow $\beta'_j \in \mathbb{Z} \mid \forall j \in p$ to be an integer representation of the continuous decision variable $\beta_j \in \mathbb{R} \mid \forall j \in p$. A constant H is used to facilitate the conversion of β'_j from an integer variable to a continuous variable β_j by setting $\beta_j = \beta'_j/H$. A value for H can be chosen arbitrarily, noting that larger values for H will result in greater accuracy when estimating β, but increases the solution search space (resulting in extended execution times). The choice of H also influences the decimal precision of β. The proposed transformation allows the unique block structure of the MIP problem to be exploited such that it can be divided between a master and a sub-problem for Benders decomposition.[5] A brief overview of Benders decomposition will follow later on in this section. When the default Benders decomposition algorithm is considered, we note that all integer decision variables are allocated to the master problem, while continuous variables are assigned to the sub-problems. By applying the standard Benders decomposition algorithm to the MIP formulation proposed by Venter (2020), all decision variables apart from the cardinality selection variables z_j will be allocated to the sub-problems which will result in an invalid decomposition. By using integer values for the continuous regression parameters, we are able to make use of a valid Benders decomposition in an attempt to increase computational efficiency. The complete proposed MIP formulation for the piecewise linearised log-likelihood function (with the aforementioned modifications) is presented as

$$\max_{\beta',z,\Lambda} \sum_{i\in n}\sum_{k\in K} (\gamma_k Y_i - \log[1 \quad \exp(\gamma_k)])\Lambda_{ik}, \tag{36}$$

subject to

$$\sum_{k\in K} \Lambda_{ik}\gamma_k = \frac{\beta'^T}{H} X_i \quad \forall i \in n, \tag{37}$$

$$\sum_{k\in K} \Lambda_{ik} = 1 \quad \forall i \in n, \tag{38}$$

$$0 \leq \Lambda_{ik} \leq 1 \quad \forall k \in K; \forall i \in n, \tag{39}$$

$$\frac{\beta'_j}{H} = 0 \quad \forall j \in p', \tag{40}$$

$$\left|\frac{\beta'_j}{H}\right| \leq M z_j \quad \forall j \in p, \tag{41}$$

$$\sum_{j\in p} z_j \leq c. \tag{42}$$

[5] The use of Benders decomposition greatly reduces the memory requirement associated with solving the best subset selection MIP problem (formulated in Section 3) in comparison to using the standard branch-and-bound algorithm.

By incorporating constraint (40), we allow the model to automatically discard highly correlated variables that breach a VIF threshold supplied by the user (recall that $VIF_j > L, \ \forall j \in p'$). This also reduces the problem search space of the MIP. Note that constraint (37) serves as the connecting constraint that distinguishes between the master problem and sub-problem for Benders decomposition, where β'_j/H relates the integer regression parameters to the real-valued parameters β_j. The idea behind Benders decomposition is to fix a set of complicating variables to yield an optimisation problem that is notably easier to solve. A high-level introduction to the approach will be provided at the hand of a mixed integer linear programming (MILP) problem (note that the best subset selection logistic regression problem presented in (36)–(42) also classifies as a MILP). The objective function of the Benders decomposition problem can be defined according to

$$\min_{x,y} \rho^T y \quad \sigma^T x, \tag{43}$$

subject to

$$Wy = \kappa, \tag{44}$$

$$Oy \quad Dx = \tau, \tag{45}$$

$$x \geq 0, \quad \forall x \in \mathbb{R}, \tag{46}$$

$$y \geq 0, \quad \forall y \in \mathbb{Z}, \tag{47}$$

where $W \in \mathbb{R}^{m_1 \times n_1}$ represents a known coefficient matrix of size $m_1 \times n_1$ and $\kappa \in \mathbb{R}^{n_1}$ a pre-determined vector of size n_1 (Rahmaniani et al., 2017). The variables $x \in \mathbb{R}$ and $y \in \mathbb{Z}$ are continuous and integer decision variables, respectively. Let $O \in \mathbb{R}^{m_2 \times n_1}$ and $D \in \mathbb{R}^{m_2 \times n_2}$ denote matrices of size $m_2 \times n_1$ and $m_2 \times n_2$, respectively. Similar to κ, let $\tau \in \mathbb{R}^{n_2}$ denote a predefined vector of size n_2 (Rahmaniani et al., 2017). Finally, let σ and ρ denote the cost associated with each decision variable in x and y, respectively, with the objective function in (43) aimed at minimising the overall cost with respect to x and y. The integer decision variables y can be considered as being a set of complicating variables (Rahmaniani et al., 2017). Constraint (45) acts as a connecting constraint which relates the continuous decision variables x to the complicating integer decision variables y. The same can be said for constraint (37) in the ALRSF formulation, where (37) can be rewritten such that $\sum_{k \in K} \Lambda_{ik} \gamma_k - (\beta'^T/H) X_i = 0 \ \ \forall i \in n$ resembles constraint (45) in a standard MILP. The right-hand side of (37) can be rewritten such that it represents the standard form of the Benders decomposition problem. For any fixed value of y (defined as y^*) the original optimisation problem (43)–(47) can be defined as

$$\min\{\rho^T y^* \quad \min_{x \geq 0}\{\sigma^T x \ \mid \ Dx = \tau - Oy^*\}\}. \tag{48}$$

Since the inner minimisation problem is a continuous linear problem, it follows from duality theory that the inner problem can be dualised by adding dual variables θ_v to the relaxed constraint set given by $Dx = (\tau - Oy^*)$ where $v \in V$ is a set of extreme points. (Rahmaniani et al., 2017). Each solution to the inner maximisation problem in (49) generates one of the extreme points $\theta_v, \forall v \in V$. The dual representation of (48) then aims to

$$\max_{\theta_v, v \in V} \theta_v^T (\tau - Oy^*), \tag{49}$$

subject to

$$\theta_v^T D \leq \sigma \quad \forall v \in V. \tag{50}$$

It follows that (49) can be substituted into the second term of the minimisation problem presented in (48) which leads to a minmax optimisation problem equivalent to

$$\min\{\rho^T y^* \quad \max_{\theta_v, v \in V}\{\theta_v^T(\tau - Oy^*)\}\}, \tag{51}$$

subject to

$$\theta_v^T D \leq \sigma \mid \forall v \in V. \tag{52}$$

The feasibility of the inner maximisation problem is dependent on θ (as captured by constraint (52) above) but independent of the choice of y^*. According to Rahmaniani et al. (2017), the choice of y^* can, however, result in either a feasible or unbounded solution of the optimisation problem (Rahmaniani et al., 2017). Considering the latter case, there exists some direction r_g (with $g \in G$ describing all possible directions of unboundedness) for which $r^T(\tau - Oy^*) > 0$. To prevent unboundedness, a feasibility cut-off of the form

$$r^T(\tau - Oy^*) \leq 0, \tag{53}$$

is added to (51) which leads to

$$\min\{\rho^T y^* \quad \max_{v \in V}\{\theta_v^T(\tau - Oy^*)\}\}, \tag{54}$$

subject to

$$r_g^T(\tau - Oy^*) \leq 0 \mid \forall g \in G. \tag{55}$$

The objective function in (54) can be linearised by replacing the dual maximisation problem (second term of (54)) with a decision variable ξ. This linearised form of the optimisation problem (54)–(55) is known as the Benders decomposition master problem (MP) and is defined as

$$\min_{y, \xi} \rho^T y \quad \xi, \tag{56}$$

subject to

$$Wy = \kappa, \tag{57}$$

$$r_g^T(\tau - Oy) \leq 0 \mid \forall g \in G, \tag{58}$$

$$\theta_v^T(\tau - Oy) \leq \xi \mid \forall v \in V, \tag{59}$$

$$y \geq 0 \mid y \in \mathbb{Z}. \tag{60}$$

The Benders decomposition sub-problem (SP) is captured by (49)–(50). Ultimately, the MP aims to find suitable values for y and ξ, after which the SP is solved by using a value of y that is generated when the MP yields $y = y^*$. Constraints (58) and (59) can be referred to as feasibility and optimality cuts, respectively. The cuts are generated based on the solution of the sub-problem. If the solution of the sub-problem is unbounded, a feasibility cut of the form (58) is added to the MP. Similarly,

if the solution is both feasible and bounded, an optimality cut of the form (59) is added to the MP. This iterative solve-and-cut approach continues until the algorithm converges to a solution. The generated cuts prune the solution search space such that the overall computational burden of the algorithm is reduced. The MP will generally contain all of the integer variables, while the remainder of the variables (which are the continuous variables) will form part of the SP (Taşkin, 2010). The Benders decomposition algorithm can therefore be applied to the logistic regression MIP presented in (36)–(42) because it can be decomposed into an MP and SP as a result of the integer transformation that is proposed for the regression parameters.

4. Novel automated logistic regression solution framework (ALRSF)

The ALRSF's goal is to solve the MIP presented in (36)–(42) to optimality while also producing high-quality solutions. This is no easy task given that the best subset selection problem is NP-hard. As such, this section will clearly outline the adopted approach which includes a combination of mixed integer programming, Benders decomposition and heuristic-based techniques. Let $t \in T = \{1, 2, ..., |T|\}$ denote an outer looping constant that is used to keep track of the number of iterations that are executed by the ALRSF, where T is equal to p, the number of independent variables under consideration in the training set. Next, assume that all of the independent variables are sorted according to their information value (IV) in descending fashion (where the variable with the highest IV is ranked first and the variable with the lowest IV is ranked last) and IV_j denote the ranking of the jth variable relative to the other variables. The ranking dictates the order in which the independent variables are included in the solution algorithm at each step S. IV is widely used within credit scoring applications to assess an independent variable's predictive power and its ability to differentiate between the discrete target variable classes (Kolacek and Rezac, 2010; Kvesic and Gordana, 2012). In general, a variable with an IV that is higher than 0.2 or 0.3 is an acceptable candidate for inclusion in a scorecard model (Kolacek and Rezac, 2010; Kvesic and Gordana, 2012). For the ALRSF, however, the IVs are only used to rank-order each independent variable (relative to the other variables) with respect to its perceived predictive power. This means that the absolute values of the IVs themselves have no meaning.[6] Let $\Delta\gamma_k$ denote the distance between two splines $[\gamma_{k\ 1} - \gamma_k]$ associated with the piecewise linear approximation of the log-likelihood function (depicted in Figure 2). In case of symmetric splines $\Delta\gamma_k$ can be simplified to $\Delta\gamma$. Let $\Omega_i \in n$ represent a grid range evaluation variable which is used to identify whether the selected regression parameters cause the current grid range $[-\Gamma, \Gamma]$ to be inadequate and is calculated as $\Omega_i = (\beta'/H^T)X_i \ \ \forall i \in n$. To illustrate, refer to Figure 2 where the extreme points of the grid are shown as $-\Gamma = \gamma_0$ and $\Gamma = \gamma_k$. The solution search space is deemed to be too small when $\Omega_i \geq \Gamma$ or $\Omega_i \leq -\Gamma$, which implies that the grid with range $[-\Gamma, \Gamma]$ must be expanded to ensure that all possible solutions are considered by the logistic regression MIP. Specifically, the size of the grid needs to be expanded by some value ψ such that the grid points now range between $[-\Gamma - \psi, \Gamma \quad \psi]$. The constant ψ is arbitrarily selected and influences the incremental size with which the search space is increased. A large value for ψ may reduce the number of iterations needed to identify a sufficiently large grid, but can potentially cause the search

[6]For the purpose of this article IV was used as a measure of variable importance, however, any appropriate variable importance statistic (e.g. univariate Gini coefficient) can be utilised in place of IV to establish the independent variable importance rank-order.

space to be larger than needed, which can increase execution time. A balance should thus be struck between execution speed and the tightness of the solution search space. Let $\zeta \in \{0, 1\}$ represent a binary variable which is used to identify whether the grid range $[-\Gamma, \Gamma]$ is sufficiently large or needs to be expanded. When $\zeta = 1$ the current iteration t will be re-executed with an increased grid range. This will continue until the grid range is sufficiently large and represents the complete search space before the ALRSF can move to iteration $t \quad 1$. When $\zeta = 0$, the grid range considered is deemed to be large enough and the relevant performance measures and solution progress recorded for use in iteration $t \quad 1$.

A detailed study by Venter (2020) showed that a large value for K (the number of piecewise linear splines used to approximate the non-linear log-likelihood) can dramatically increase the execution time of the algorithm, while smaller K values result in an underestimation of the non-linear log-likelihood function. The aforementioned underestimation is, however, only marginal (and at times negligible) for small values of K. In general, models exhibited similar predictive performance across multiple empirical experiments for a wide range of K values. Let P_t denote the performance measure recorded at iteration t based on a holdout test set[7]. The performance measure P_t is used to determine whether the cardinality of the model (denoted by c) and set of regression parameters associated with the best model identified thus far (denoted by β'^b) should be updated with the solutions (the variables selected and their associated parameters) generated in the latest iteration. Let P_b denote the best value of P_t that is recorded over all iterations such that $P_b = \max(P_t)$ for $t = \{1, 2, 3, ..., |T|\}$. To evaluate the ALRSF, the area under the curve (AUC) was used as a performance measure P_t (although, theoretically, any model fit statistic such as the Gini coefficient or accuracy can also be used). A list of possible performance measures for binary classification problems is available in Table 1 of Sokolova and Lapalme (2009). Since the ALRSF starts with an iterative cardinality selection step and uses a performance measure P_t to assess the relative increase/decrease in model performance based on the inclusion of the jth independent variable, some form of performance tracking is required to identify when the predictive power of the model starts to plateau in spite of the addition of more independent variables. Specifically, when the performance of the model on the test dataset either reaches a plateau or decreases at iteration t of the ALRSF, it indicates that the cardinality parameter c (which dictates the number of variables to include in the model) can be set equal to the size of the subset identified at iteration $t-1$. The variable $u \in \mathbb{Z}$ is used as the aforementioned tracking variable that keeps track of the number of consecutive iterations for which the performance measure P_t has not improved on the best-recorded performance measure P_b. The value of u is then evaluated against a predefined threshold U, which determines the maximum allowable consecutive iterations for which P_b does not improve. When the threshold U is exceeded the cardinality step will be terminated and the search space pruning step initiated.

A holistic overview of Algorithms 1 to 5 will be provided next, which are explained in conjunction with Algorithm 1. This is because Algorithm 1 serves as an overarching driving algorithm that controls the various steps $S = \{0, 1, 2\}$ contained within the ALRSF. Since Algorithm 1 functions as the driving force behind each step of the ALRSF, the reader can continuously return to Algorithm 1 to identify the subsequent step that should be executed. In other words, the algorithmic flow will

[7] Note that the ALRSF evaluates the performance of the model based on a test set so that models that overfit on the training data are not presented as the champion.

jump between Algorithm 1 and the remainder of the Algorithms 2 to 5 as required by the ALRSF logic. Next, we discuss the flow of each algorithm contained within Algorithm 1:

- At the start of Algorithm 1, the step counter is set equal to $S = 0$ and we start at iteration $t = 1$. When the for-loop is entered at $t = 1$, the logic contained within Algorithm 2 is initiated.
- As we enter Algorithm 2 at iteration $t = 1$ and step $S = 0$, the pseudo code in lines 6 to 12 is executed. The aforementioned pseudo code identifies which regression parameters contained within the vector β' should be set equal to zero and which are allowed to take on integer values. Once this is done, we return to Algorithm 1 as per line 25 of Algorithm 2.
- Entering Algorithm 1 for a second time at iteration $t = 1$ and step $S = 0$ invokes the if statement in line 4 of Algorithm 1. This if statement instructs the solver to now consider Algorithm 3.
- The goal of Algorithm 3 is to automatically determine the cardinality parameter c (which dictates the size of the best subset of predictor variables) by iteratively adding independent variables one-by-one to the logistic regression MIP in (36)–(41) (note that constraint (42) is omitted in this step as we have not determined a final value for c yet). The change in the performance measure P_b is also assessed during this step. The final value of c is determined at the point where no further improvement in P_b has been observed for U consecutive iterations. At each iteration t, the MIP will abide by an additional constraint which dictates that $\sum_{j \in p} z_j = t$ based on the regression parameters that were allowed to be non-zero in Algorithm 2. Note that Algorithm 3 does not have an outer loop counter that corresponds to each iteration $t \in T$. Therefore, Algorithm 3 is called as needed (and non-iteratively) by Algorithm 1.
- We now direct our attention to the two for-loops in lines 7 to 9 and 13 to 20 of the pseudo-code of Algorithm 3. The first for-loop removes highly correlated variables from the MIP formulation by setting their regression parameters equal to zero. It is assumed that the VIF associated with each variable and the VIF threshold is provided by the user. The second for-loop determines if the sum of the regression parameters and independent variable values of each record falls within the range $[-\Gamma, \Gamma]$ that was specified for grid points used to linearise the log-likelihood (see lines 13 to 20). If this is not the case, the range of the grid is increased to ensure that the solution search space is sufficiently large and the tth iteration is re-executed.
- Once Algorithm 3 concludes in iteration $t = 1$, we revert back to Algorithm 1, as indicated by line 45 of Algorithm 3. Since the cardinality selection step has not reached the consecutive performance improvement limit U as described in lines 35 to 38, the step counter S is not incremented and as such will remain fixed at $S = 0$.

Algorithm 1 – ALRSF algorithmic driver.

$S = 0$
for $t = 1$ to T **do**
  Execute Algorithm 2 (regression parameter fixing heuristic) based on the value of S.
  if $S = 0$ **then**
    Execute Algorithm 3 (cardinality selection step) where $S = 0$.
  else if $S = 1$ **then**
    Execute Algorithm 4 (search space pruning step) where $S = 1$.
  else if $S = 2$ **then**
    Execute Algorithm 5 (optimality step) where $S = 2$.
  end if
end for
Solution Algorithm Completed.

Algorithm 2 – Regression parameter fixing heuristic.

if $(t \geq T)$ & $(S = 1)$ **then**
  $S = S \quad 1$
  $t = T - 1$
end if
if $S = 0$ **then**
  for $j = 1$ to p **do**
    if $(IV_j \leq t)$ **then**
      β'_j = free
    else
      $\beta'_j = 0$
    end if
  end for
else if $S = 1$ **then**
  for $j = 1$ to p **do**
    if $(|\beta'^{t-1}_j| > 0)$ or $(IV_j = t)$ **then**
      β'_j = free
    else
      $\beta'_j = 0$
    end if
  end for
else if $S = 2$ **then**
  for $j = 1$ to p **do**
    β'_j = free
  end for
  Return to Algorithm 1 - (driver algorithm).
end if

Algorithm 3 – Cardinality selection step – $S = 0$.

$c = 0$
P_t and $P_b = 0$
${\beta'}^b = 0$
$k = 0$
$u = 0$
$\zeta = 0$
for $j = 1$ to p' **do**
  $\beta'_j = 0$ if $VIF_j > L$
end for
Each regression parameter β'_j associated with variable j for which $VIF_j > L$ is set equal to zero.

Solve logistic regression MIP presented in (36)–(40) using Benders decomposition.
Cardinality Constraints (41) and (42) are excluded from the MIP.
for $i = 1$ to n **do**
  $\Omega_i = 0$
  $\Omega_i = \Omega_i \quad {\beta'}^T X_i$
  if $(\Omega_i \leq -\Gamma)$ or $(\Omega_i \geq \Gamma)$ **then**
    $|\Gamma| = |\Gamma| \quad \psi$
    $\Delta\gamma = \frac{2*|\Gamma|}{K}$
    $\zeta = 1$
  end if
end for
if $\zeta = 0$ **then**
  Calculate and record training and test performance measures P_t.
  Record ${\beta'}^t$.
  if $P_t > P_b$ **then**
    for $j = 0$ to p **do**
      ${\beta'_j}^b = \beta'_j$
    end for
    $P_b = P_t$
    $c = c \quad 1$
    $u = 0$
  else
    $u = u \quad 1$
  end if
  if $u = U$ **then**
    $S = S \quad 1$
    $t = 0$
  end if
else
  $t = t - 1$
end if
Cardinality c is produced as a result of this algorithm.
Save MIP solution found at iteration t for use as warm-start during the execution of iteration $t \quad 1$.
increment $t = t \quad 1$.
Return to Algorithm 1 - (driver algorithm).

- The ALRSF will once again initiate Algorithm 2 at iteration $t = 2$ and step $S = 0$ so that it can identify the next sequence of regression parameters that are allowed to take on non-zero values, after which a version of the logistic regression MIP is solved such that a model is fitted that contains t independent variables. When the stopping criterion in lines 35 to 38 of Algorithm 3 is met, the step counter S is incremented and the iteration counter is reset such that $S = 1$ and $t = 0$. We then return to Algorithm 1.

- A step value of $S = 1$ and iteration counter value of $t = 0$ in Algorithm 1 indicates that the ALSRF should execute the portion of the parameter fixing heuristic algorithm (Algorithm 2) that applies when $S = 1$. This is contained in lines 13 to 20 of the pseudo-code for Algorithm 2. The logic is quite similar to that of lines 5 to 12 with only a few minor adjustments. Once the appropriate regression parameters have been set equal to zero or non-zero, respectively, the ALRSF returns to Algorithm 1 (as indicated by line 25 of Algorithm 2).

- Next, the ALRSF moves to lines 6 and 7 of Algorithm 1. In line 7, the search space pruning step is initiated, the logic of which is found in the pseudo-code of Algorithm 4. Similar to Algorithm 3, Algorithm 4 contains no iteration counter and is therefore called as and when needed by Algorithm 1 (based on the step value S).

- The only difference between Algorithm 3 and Algorithm 4 relates to the fact that the logistic regression MIP is now solved by incorporating the cardinality constraint in (42) in Section 3. When $S = 1$, the number of independent variables that are allowed to enter the final model at each iteration is allowed to be at most c 1. This cardinality constraint is enforced by the regression parameter fixing logic contained within Algorithm 2 where $S = 1$.

- Furthermore, Algorithm 4 also determines if the solution search space is sufficiently large enough by applying the logic contained within lines 10 to 18. If this is not the case, the grid that is used to linearise the log-likelihood is enlarged by adjusting the range as in line 14, after which iteration t is re-executed. In essence, Algorithm 4 attempts to iteratively include independent variables into the final subset of predictors (one by one in a sequential manner, where the sequence depends on the variables' IV) to determine if model performance can be improved when compared to a model that only contains the predictors that are already in the best subset of variables.

- Algorithm 4 will continue to execute until one pass through all of the independent variables has been completed, i.e. until iteration $t = T = p$ is reached. In doing so, the entire solution search space is pruned by solving sub-problems associated with the logistic regression MIP in (36)–(42). It is important to note that the solution produced by the model at iteration t is used as a warm-start for iteration t 1 to ensure that any incremental progress made by continuous search space pruning is not lost. In other words, when the model prunes the search space during an iteration, the pruned space is recalled in future iterations in an attempt to reduce execution times.

- At iteration $t = T = p$, Algorithm 4 concludes and lines 1 to 4 of Algorithm 2 are invoked (as per Algorithm 1) which sets $S = 2$ and initiates the optimality step. Since the optimality step

Algorithm 4 – Search space pruning step – $S = 1$.

$\zeta = 0$
Cardinality parameter value c is generated by Algorithm 3.
for $j = 1$ to p' **do**
  $\beta'_j = 0$ if $VIF_j > L$
end for
Each regression parameter β'_j associated with variable j for which $VIF_j > L$ is set equal to zero.

Solve logistic regression MIP presented in (36)–(40) using Benders decomposition.
Cardinality Constraints (41) and (42) included in the MIP.
Cardinality is set equal to c.
for $i = 1$ to n **do**
  $\Omega_i = 0$
  $\Omega_i = \Omega_i \quad \beta'^T X_i$
  if $(\Omega_i \leq -\Gamma)$ or $(\Omega_i \geq \Gamma)$ **then**
    $|\Gamma| = |\Gamma| \quad \psi$
    $\Delta\gamma = \frac{2*|\Gamma|}{K}$
    $\zeta = 1$
  end if
end for
if $\zeta = 0$ **then**
  Calculate and record training and test performance measures P_t.
  Record β'^t.
else
  $t = t - 1$
end if
Save MIP solution found at iteration t for use as warm-start during execution of iteration $t \quad 1$
increment $t = t \quad 1$
Return to Algorithm 1 - (driver algorithm)

Algorithm 5 – Optimality step – $S = 2$.

Cardinality parameter value c is generated by Algorithm 3.
for $j = 1$ to p' **do**
  $\beta'_j = 0$ if $VIF_j > L$
end for
Each regression parameter β'_j associated with variable j for which $VIF_j > L$ is set equal to zero.

Use last MIP solution found in iteration t as warm-start to step $S = 2$.
Allow remaining regression parameters to be non-zero.
Solve logistic regression MIP presented in (36)–(40) using Benders decomposition.
Record final regression parameters, performance measures and optimality gap.
Return to Algorithm 1 - (driver algorithm).

is not iterative, t can be set equal to $t = T - 1$ considering that $S = 2$ is the last step in the ALRSF.

- Lines 21 to 24 of Algorithm 2 are subsequently executed when $S = 2$ and $t = T - 1$. This allows all the integer versions of the regression parameters in the vector β' to take on non-zero values (if deemed significant by the model) since the entire solution search space needs to be evaluated to provide a guarantee of optimality. Once all regression coefficients are allowed to take on non-zero values, the ALRSF returns to Algorithm 1.
- With $S = 2$, lines 8 and 9 in Algorithm 1 are executed, which initiates the optimality step. Algorithm 5 is one of the more simplistic components of the ALRSF and simply solves the logistic regression MIP in (36)–(42), where the cardinality parameter c is specified based on the results generated by Algorithm 3 and any regression coefficient is allowed to take on non-zero values. Ultimately, Algorithm 5 solves the original best subset selection regression problem, while the other Algorithms and steps presented thus far assist in automatically determining the cardinality parameter and pruning the solution search space in an attempt to reduce memory requirements and provide an optimality guarantee.
- The ALRSF concludes when optimality is guaranteed, an out-of-memory exception occurs or an alternative stopping criterion is met.

Now that the algorithmic flow of the ALRSF has been presented, some of the key elements associated with each of the three algorithmic steps $S = \{0, 1, 2\}$ will be discussed in more detail. The goal of the regression parameter fixing heuristic logic in Algorithm 2 (which is executed at steps $S = 0$ and $S = 1$ of the framework) is to reduce the computational memory burden associated with solving the complete logistic regression MIP in (36)–(42). From Algorithm 2, observe that the logic of the parameter fixing heuristic is adjusted based on the value of S. At step $S = 0$, Algorithm 2 ensures that the independent variables are iteratively included in the model based on their relative IV rank-ordering (starting with the most predictive variables first). This implies that the regression parameters associated with the most predictive variables are iteratively allowed to be non-zero, while the remaining coefficients are set equal to zero. This allows the user to ascertain the influence of each independent variable on model performance before the cardinality step at $S = 0$ is terminated (based on the threshold U) or before the next independent variable is considered for inclusion. As stated previously, during the execution of the cardinality selection step, the total number of independent variables that are allowed to have non-zero regression parameters at any given iteration will be equal to t.

Similar to before, the independent variables are once again added to the model in an iterative manner when $S = 1$ in Algorithm 2. However, the regression parameters of all the independent variables that were added to the best predictor subset in step $S = 0$ are now allowed to be non-zero and take on any real value. When $S = 1$, it is important to note that the number of variables considered for inclusion in the model will be equal to c 1 during the execution of iteration t. However, at the end of iteration t, a maximum of c predictors may be present in the model. In essence, when an additional independent variable is considered for inclusion in the model in the current iteration t (given that there may or may not already be variables present in the model), one of the following can occur:

1. the variable is added to the final model if the number of non-zero regression parameters is less than c and the performance of the model is improved,

2. the variable replaces one of the other predictors that are already present in the model if the number of non-zero regression parameters is equal to c and if model performance is improved, or

3. if model performance does not improve (or deteriorates), the variable is rejected.

The solution search space is therefore iteratively reduced by testing the various regression parameter combinations. When $S = 2$, the final step of the parameter fixing heuristic (Algorithm 2) permits any regression parameter to be non-zero so that the entire solution search space is evaluated and an optimality gap is calculated. The solution from step $S = 1$ is used as a warm-start in step $S = 2$. This significantly reduces the size of the search space that needs to be considered before optimality can be proven.

Next, we consider Algorithm 3, which automatically determines the most appropriate cardinality parameter c based on the predictive power of the independent variables, the number of observations and performance of the model on the test set. In conjunction with the regression parameter fixing heuristic, Algorithm 3 also includes an additional functionality to allow for dynamic adjustment of the grid range $[-\Gamma, \Gamma]$ as explained earlier in this section. At each iteration t, the size of the grid range is evaluated to ensure that it is suitable and allows for the complete search space to be considered when a new independent variable is added to the model. The grid is updated by adding ψ to end-points such that the new range is $[-\Gamma - \psi, \Gamma \quad \psi]$ if the size of the grid is found to be insufficient (based on the value of Ω_i). This process of check-and-expand continues until the solution search space is sufficiently large. It should also be noted that the size of the grid will depend on the size of the regression parameters, the scale of each of the predictors and the cardinality parameter c. These factors may differ from one problem to the next and as such the proposed ALRSF allows for dynamic adjustment of the solution search space when required.

As part of Algorithm 3, the cardinality constraints presented in the logistic regression MIP in (36) -(42) are excluded from the model fitting step since the influence of each independent variable is evaluated without enforcing any restriction on the regression coefficients. The main output produced as part of the cardinality step is the final cardinality parameter c which will be utilised in steps $S = 1$ and $S = 2$, respectively. During each iteration, the MIP is solved by using the standard Benders decomposition approach (as discussed in Section 3). Furthermore, the integer solution found at iteration t is used as a warm-start in iteration $t \quad 1$. This prevents the model fitting algorithm from continuously resolving the same complete logistic regression MIP instance, which in turn should result in reduced computational memory demand and execution times.

When Algorithm 4 is invoked, the logistic regression MIP contained within each search space pruning iteration $t \in T = p$ is once more solved using Benders decomposition. The main difference between Algorithm 3 (executed at step $S = 0$) and Algorithm 4 (executed at step $S = 1$) is the inclusion of cardinality constraints in Algorithm 4, which limits the number of variables that can enter the model and facilitates best subset selection. The addition of the cardinality constraints to the MIP formulation introduces further complexity to the model, which leads to larger memory requirements because the model is tasked with filtering through numerous independent variable combinations.

This is where the search space pruning step is most useful as it iteratively prunes the search space by gradually testing the possibility of each independent variable entering the best variable subset. Algorithm 4, therefore, solves multiple sub-problems that only contain a subset of the available independent variables. As mentioned earlier, variables are iteratively added to the best subset (one at a time based on their IV rank-ordering) for testing and the Algorithm will only terminate when all independent variables have been tested at least once. Specifically, for Algorithm 4, the largest MIP that will be solved at any given point in time will be subject to the constraint $\sum_{j \in p} z_j \leq c \quad 1$, while the parameter c and iteration $t = 0$ is set equal to the value that was found in Algorithm 3. Similar to Algorithm 3, the search space pruning step also evaluates whether the size of the grid with range $[-\Gamma, \Gamma]$ is sufficiently large at iteration t. If found insufficient, the extreme points of the grid are increased by ψ until $\Omega_i \geq -\Gamma$ and $\Omega_i \leq \Gamma$ before continuing to iteration $t \quad 1$.

The final step of the ALRSF, which is executed by Algorithm 5 when $S = 2$, aims to improve on the solution generated by the search space pruning step by including all independent variables in the MIP solution process (except for those variables for which the VIF threshold was violated) while allowing their regression parameters β' to take on any integer value between the range $(-\infty, \infty)$. The best integer solution generated as part of step $S = 1$ using Algorithm 4 is used as a warm-start when executing Algorithm 5 in the final step where $S = 2$. This means that the optimality step $S = 2$ is not required to conduct a complete search of the entire solution space since a large portion of the search space has already been eliminated by Algorithm 4. The optimality step can therefore focus on improving the best integer solution while also attempting to prove optimality. The fact that the proposed solution framework is able to generate an optimality gap is a key advantage over normal heuristic-based variable selection algorithms, because it provides a guarantee on the optimality of the solution. As stated by Kutner et al. (2005), heuristic regression algorithms (such as stepwise selection, lasso and ridge regression) produce models that are "good enough". However, the user is not able to guarantee that the resulting model is the best possible model among all possible models and variable combinations. However, when an optimality gap is produced, the user is able to quantify a hypothetical distance between the predictiveness of the current model and the best possible model, with a gap of 0% indicating optimality. In other words, an optimality gap of 0% means that the final model (with at most c predictors) is the best possible model that could be obtained amongst all possible variable combinations.

4.1 Dummy example

We now attempt to clarify the three steps and five algorithms of the ALRSF using a practical logistic regression example that is based on a dataset containing six independent variables. It should be noted that the dummy example presented here is hypothetical and for illustrative purposes only. The performance and validity of the proposed ALRSF will be evaluated in Sections 5 and 6 using real-world datasets. The results of the dummy example are summarised in Table 1, which represents what a solution that is produced by the ALRSF would resemble. The first six iterations summarise the cardinality selection logic (Algorithm 3) while the following six relate to the search space pruning step (Algorithm 4). Finally, an attempt is made to prove optimality during the last iteration which relates to the optimality step described in Algorithm (5). Similar to what was presented in Algorithm 3, at iteration $t = 1$ the cardinality parameter is initialised to a value of $c = 0$. From iteration $t = 1$ to

Table 1. Novel ALRSF – Dummy Example.

Iterations	β'_1	β'_2	β'_3	β'_4	β'_5	β'_6	Test AUC	c	Gap %
Cardinality Step - (S = 0, c = 0)									
t = 1	Free	Fixed	Fixed	Fixed	Fixed	Fixed	51.34	1	–
t = 2	Free	Free	Fixed	Fixed	Fixed	Fixed	65.41	2	–
t = 3	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	–
t = 4	Free	Free	Free	Free	Fixed	Fixed	87.30	3	–
t = 5	Free	Free	Free	Free	Free	Fixed	87.31	3	–
t = 6	Free	Free	Free	Free	Free	Free	87.29	3	–
Search Space Pruning Step - (S = 1, c = 3)									
t = 1	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	–
t = 2	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	–
t = 3	Free	Free	Free	Fixed	Fixed	Fixed	87.60	3	–
t = 4	Free	Free	Fixed	Free	Fixed	Fixed	88.50	3	–
t = 5	Free	Free	Fixed	Fixed	Free	Fixed	88.65	3	–
t = 6	Free	Free	Fixed	Fixed	Free	Fixed	88.65	3	–
Optimality Step - (S = 2, $\forall\beta \in \mathbb{R}^p$, c = 3)									
t = 1	Free	Free	Free	Free	Free	Free	88.65	3	42.56

$t = 6$, the regression parameters[8] are sequentially added to the model based on IV rank-order.

For this example, we assume that the columns of Table 1 reflect the relative rank-ordering of the variables (based on their IVs) for $j = 1$ to 6. In other words, β'_1 corresponds to the variable with the highest IV, whereas β'_6 is associated with the variable with the lowest IV. Since the cardinality step tests the addition of one independent variable at a time, there will always be $c = t$ variables included in the model at iteration t. The regression parameters associated with the independent variables that enter the MIP for testing at each iteration are allowed to take on any integer value. This means that at iteration $t = 1$ regression parameter β'_1 is allowed to be non-zero such that $\beta'_1 \in \mathbb{Z}$, while the remaining regression parameters are fixed at zero. This essentially means that at iteration $t = 1$ a sub-problem with 6 independent variables is solved, utilising the standard Benders decomposition algorithm, with the regression parameter of one independent variable set allowed to be non-zero and the remaining parameters set equal to zero. Notice that the regression parameter β'_1 is allowed to be non-zero throughout the cardinality step (the first six iterations). This is because the variable associated with this parameter has the largest IV at iteration $t = 1$. Since variables are tested sequentially, the regression parameter β'_1 will always be allowed to be non-zero (throughout the first six iterations) as its predictor will have the highest IV, regardless of other variable combinations that are added. Furthermore, for iterations 1 to 6, note that the sequential inclusion of variables to the model means that one additional regression parameter is allowed to be non-zero at a time, as outlined by Algorithm 3.

[8] Each repression parameter $\beta'_j \ \forall j \in p$ is either set free or fixed. The term "Free" refers to the regression parameter $\beta'_j \in \mathbb{Z}$ being allowed to take on any integer value in $[-M, M]$, while "Fixed" relates to β_j being set equal to exactly zero.

Similarly, the test AUC P_t and highest recorded AUC P_b are both set equal to 0 when the cardinality step is initialised. At each iteration, P_t is recorded by solving the logistic regression MIP and fitting the model in a sequential manner (as discussed earlier). At iteration $t = 1$, P_1 is set equal to 51.34%, which is the AUC that was obtained at this step. Since P_1 is greater than the initial value of $P_b = 0$, we set $P_b = P_1$ and the cardinality parameter c is incremented by 1. The same is true for iteration $t = 2$ where the regression parameter β_2' is allowed to enter the model based on the IV rating of predictor variable X_2. A MIP with 6 independent variables is then solved, where the first two regression parameters are allowed to vary between $(-\infty, \infty)$, while the remaining regression parameters are set equal to zero. The test AUC P_2 after execution of the second iteration was found to be, 65.41% which is higher than P_b. As a result, we set $P_b = P_2$ and the cardinality parameter c is incremented once more such that $c = 2$. The same logic is applied for iterations 3 to 6 throughout the cardinality step (via the execution of Algorithm 3). At iteration 4, the test AUC that was found is smaller than P_b which means that the cardinality parameter c is not updated. In fact, for iterations 4, 5 and 6, none of the AUC values derived from the test set is greater than P_b, which means that the parameter c cannot be incremented. Ultimately, the cardinality step terminates at iteration 6 and the cardinality parameter c is set equal to the value of c that corresponds to the iteration where P_b was recorded. In this case, P_b was found at iteration 3 and we conclude that a cardinality parameter of $c = 3$ is most appropriate. This means that, for the specific logistic regression problem at hand, the best subset selection approach should produce the best 3-variable model.

By adding independent variables to the cardinality selection step in a sequential manner, the user is able to evaluate the incremental influence of each variable on performance of the model (as assessed on the test set so as to prevent overfitting). The first few variables that are added during the initial steps of the algorithm are expected to contribute the most to the predictive power of the model, as high IV values are assumed to be associated with high differentiation abilities. As independent variables are continuously added to the best variable subset, a point is reached where the model becomes saturated and the test AUC curve will either form a plateau (no improvement is obtained by adding additional variables to the model) or starts to drop off (adding additional variables to the model leads to overfitting). This implies that adding additional variables to the model beyond this point is therefore futile, which suggests that all remaining variables in the training set can be excluded. When the cardinality parameter remains constant for U iterations, the cardinality step can be terminated and the search space pruning step initiated. The cardinality parameter c can then be utilised in the pruning and optimality steps in conjunction with cardinality constraints, to enforce a limit on the number of independent variables that can be included in the final logistic regression model formulation. According to Kutner et al. (2005), the order in which variables are added to or removed from a model can exert a significant influence on the final model that is obtained. For example, during stepwise selection procedures, the variable that is added during iteration t (and its regression coefficient) is dependent on the variables that were included in the model at iteration $t - 1$. One might therefore think that the proposed ALRSF presented in this paper suffers from the same drawback due to the sequential addition of variables to the model during step $S = 0$. Note, however, that this is not the case. The iterative addition of variables to the model only takes place during the cardinality step and is simply used to determine the most appropriate value of the cardinality parameter c. During step $S = 2$ of the ALRSF, all combinations of variables are tested, which means that the order in which variables enter the model is no longer influential.

The search space pruning step $S = 1$ (which is facilitated by Algorithm 4) can be described by the second set of six iterations in Table 1. The main purpose of the pruning step is to reduce the solution search space, which ultimately reduced the optimality gap percentage in the final optimality step $S = 2$. This is done by iteratively testing independent variable combinations to determine which variable subset will improve the objective function value in (36) while conforming to constraint (42) in the logistic regression MIP. The total number of independent variables that are allowed to have non-zero regression parameters at any given iteration during the execution of step $S = 1$ may not exceed c 1. For the particular example presented in Table 1, a total of four regression parameters are therefore allowed to be non-zero during the search space pruning step $S = 1$, as the most appropriate cardinality parameter was determined to be $c = 3$ during the cardinality step $S = 0$. This is evident when we consider the third and fourth iterations of the search space pruning step, where the ALRSF determined that allowing β'_4 to be non-zero instead of β'_3 yields an improvement in model performance. As a result, the predictor variable associated with β'_3 was removed from the best variable subset and replaced with the variable associated with β'_4. In other words, the independent variables are iteratively added to the model and are either allowed to enter the best subset by replacing one of the current variables with a non-zero regression parameter or are rejected, at which point the next iteration t 1 is initiated with no change to the best subset. During the execution of the search space pruning step, note that the first three regression parameters are allowed to be non-zero during the first three iterations because the cardinality parameter was specified as $c = 3$ and the predictor variables associated with these coefficients had the highest IVs. The idea behind the search space pruning step is to challenge this assumption (that the first three variables produce the best-performing model) by iteratively adding another variable to the model and testing to determine if the added variable can replace one of the variables that are already included in the best variable subset. When $S = 1$, the first iteration tests whether the variable associated with β'_1 can improve the objective function when added to a subset that is allowed to contain at most three variables. However, since this variable is already part of the best subset and the size of the subset is smaller than $c = 3$, no change is recorded. The same can be said for the variables associated with β'_2 and β'_3 during iterations $t = 2$ and $t = 3$ of the search space pruning step. From Table 1, a change in the best subset is only visible at the fourth iteration of the search space pruning step since the variable associated with β'_4 improved the predictive performance of the model by replacing the variable associated with β'_3 in the best variable set. The same logic applies for every iteration contained in the search space pruning step, where we note that no significant improvement in model performance is recorded during iterations 5 and 6. Once all of the predictors have been tested at least once, the optimality step $S = 2$ can be initiated since more than one pass over the set of independent variables will not result in further improvements in the model's predictive performance.

The final step of the ALRSF is the optimality step (which is facilitated by Algorithm 5). The solution obtained during the final iteration of step $S = 1$ (search space pruning step) is used as a warm-start in the optimality step $S = 2$, which prevents the complete logistic regression MIP from being solved and leverages the reduced search space provided by the pruning heuristic. If the warm-start method was not utilised, the progress made in step $S = 1$ would be disregarded and step $S = 1$ would add no value to the solution framework. During step $S = 2$, all regression parameters are allowed to be non-zero to ensure that all possible independent variable combinations are considered by the logistic regression model and to determine the extent to which the final log-likelihood solution

deviates from optimality. The solution framework will only terminate when optimality has been proven, an out-of-memory error occurs or a predefined user-specific stopping criterion is met. The optimality gap presented in Table 1 is only hypothetical and shows that step $S = 2$ is capable of producing a measure of optimality for the final solution.

In summary, the proposed novel ALRSF provides a holistic approach towards fitting large-scale logistic regression models that are formulated as MIP problems to perform best subset selection. This is achieved by iteratively pruning the solution space and employing the default Benders decomposition approach (which is only possible if the model is formulated as the MIP in (36)–(42)) to find high-quality solutions with smaller memory requirements and providing an optimality gap for the final solution. The end-result is a proposed approach that aims to find high-quality solutions that impose less computational memory requirements, are executed faster than comparative formulations and provide a guarantee on the quality of the model in the form of an optimality gap.

5. Model verification

In this section, the proposed ALRSF is benchmarked against the standard branch-and-bound MILP formulation of the logistic regression best subset selection problem that was proposed by Venter (2020)[9]. Two small-scale datasets that contain six and eight variables, respectively, will be considered. The problems in question relate to default prediction for credit card customers and are solved using default data provided by Taiwanese credit providers that were published on the UCI machine learning repository. The original dataset consists of 24 independent variables and 30000 observations. For illustrative purposes, a training set of 200 observations is used. This is to ensure that both problems are solved such that optimality is guaranteed (i.e. an optimality gap of 0% is obtained) within an acceptable time frame so that a proper comparative study can be conducted. The binary target class was distributed such that 23% of the observations were assigned a value of one (default). Table 2 contains the descriptions of each of the independent variables that were considered for inclusion in the model. Three main aspects of the proposed solution framework are considered when the final models are evaluated:

1. The regression parameter estimates produced by each of the algorithms need to be inspected and compared in order to determine whether they are appropriate.

2. The best variable subset produced by each model needs to be examined to assess the variables that were selected.

3. The range of the grid that is used to linearise the non-linear log-likelihood (with extreme points $[-\Gamma, \Gamma]$) needs to be scrutinised to determine whether the solution space is sufficiently described.

For the verification of the grid range, the manual grid selection method proposed by Venter (2020) was utilised to obtain a suitable grid for the standard branch-and-bound algorithm. The ALRSF was

[9] Note that commercially available software (such as SAS) also makes use of variations of the standard branch-and-bound method to execute best subset selection within regression procedures. For more information, see Appendix C of Venter (2020) that summarises the Leaps and Bounds (LBA) method of Furnival and Wilson (2000) and the branch-and-bound algorithm (BBA) of Gatu and Kontoghiorghes (2006) which are used in SAS procedures.

Table 2. Independent variable description.

Independent variable	Variable description
1	Amount of credit extended to the client
2	Client gender
3	Level of education
4	Marital status
5	Client age
6	Repayment status in September 2005
7	Repayment status in August 2005
8	Repayment status in July 2005

developed in C++ and the MIP formulations were solved utilising the Cplex concert technology API. The experiments were conducted on a Ryzen Linux Mint machine with an AMD Ryzen 5 1600 6-core processor and 32 GB random access memory (RAM). Python (with Anaconda distribution) and Microsoft Excel were used for supplementary data processing, such as calculating the VIF and IV of each independent variable. The training and test sets were saved as Microsoft Excel comma-separated files (CSV) and used as input for both the standard branch-and-bound and ALRSF algorithms.

In Table 3, the regression parameters that were produced by the standard branch-and-bound algorithm and ALRSF are shown. It is evident that near-identical regression parameters are produced by both algorithms for two versions of the problem (one where six inputs are allowed to enter the model and one where eight inputs are allowed to enter the model). There are, however, some minor differences concerning the decimal precision that will be addressed later in this section. With reference to the objective function (which is the log-likelihood), an identical value was recorded for both algorithms. This provides confidence that the proposed ALRSF framework outputs accurate and trustworthy results that are in line with the more-established branch-and-bound approach (which serves as the default method for performing best subset selection).

Next, a set of experimental results are presented to explain the decimal differences observed in Table 3 by varying the integer conversion constant H in (40) which was formally introduced in Section 3. The integer conversion constant H, which facilitates the conversion between the integer regression parameters β' and continuous parameters β, is firstly selected to be a 1000 at first and then changed to 10000. This allowed the influence of H on the final parameter estimates to be evaluated. By comparing the results in Table 4 with those in Table 3, it is clear that the ALRSF parameter estimates are more closely aligned with those produced by the branch-and-bound algorithm when $H = 10000$. Therefore, a larger value for H ensures greater decimal precision and provides a better

Table 3. Regression parameter estimate comparison.

Logic	c	p	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	LogL
ALRSF	4	6	0.655	2.460	0	0	1.536	-0.350	-5.668	-	-	-98.63
B&B	4	6	0.654536	2.46055	0	0	1.53657	-0.34927	-5.66825	-	-	-98.63
ALRSF	3	8	0.466	2.347	0	0	1.691	0	-5.738	0	0	-98.67
B&B	3	8	0.465977	2.34726	0	0	1.69113	0	-5.73871	0	0	-98.67

Table 4. Variable conversion constant H – ALRSF.

H	c	p	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	OBJF
1000	4	6	0.655	2.460	0	0	1.536	-0.350	-5.668	-	-	-98.63
10000	4	6	0.65475	2.46032	0	0	1.53659	-0.34926	-5.66828	-	-	-98.63
1000	3	8	0.466	2.347	0	0	1.691	0	-5.738	0	0	-98.67
10000	3	8	0.46597	2.34726	0	0	1.69113	0	-5.73871	0	0	-98.67

Table 5. AUC performance measure comparison.

Logic	H	c	p	Train AUC	Test AUC
ALRSF	1000	4	6	71.56	74.36
ALRSF	1000	3	8	71.34	74.52
ALRSF	10000	4	6	71.56	74.35
ALRSF	10000	3	8	71.34	74.48
B&B	1000	4	6	71.56	74.35
B&B	1000	3	8	71.34	74.48

approximation of the true underlying regression parameters, but increases the size of the search space. The choice of H may also be problem-specific. It is important to consider the influence of the conversion constant H on the performance measures that are calculated from the training and test sets, respectively. A comparison of the AUC values that were achieved on the training and test sets, respectively, can be found in Table 5. Identical training AUC values were found for all experiments, regardless of the value of H, which might suggest that H has an insignificant impact on model fitting and training set performance. It is, however, interesting to note that when the test AUC is examined, the ALRSF with $H = 1000$ produces higher AUC values when compared to models produced by both the branch-and-bound algorithm and an ALRSF with $H = 10000$. This might be attributed to a marginal reduction in overfitting on the training set (because the regression parameters have lower decimal precision) which ultimately allows for slightly better generalisation performance. Nevertheless, the performance increase can be regarded as negligible. Ultimately, we can conclude that the proposed ALRSF produces results that are expected (when compared with default best subset selection techniques) and that the performance and generalisation capabilities of the model are not severely affected by the parameters of the proposed MIP.

Table 6 shows that the best subset consists of three independent variables when a logistic regression model is fitted to the eight-variable dataset using the ALRSF. The purpose of the results recorded in Table 6 is to evaluate the logic of the solution algorithm and to determine if a cardinality parameter of $c = 3$ is a sensible choice. Recall from Section 3 that a specific cardinality selection criteria can be set by the user to determine whether the improvement in the performance measure P_t between

Table 6. Cardinality selection verification.

Logic	c	p	U	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$	$t = 6$	$t = 7$	$t = 8$
ALRSF	3	8	2	72.04	72.24	73.96	73.86	73.70	75.32	75.34	75.32

iterations is sufficient enough to allow the cardinality parameter c to be incremented with one. For the purpose of the model shown in Table 6, the selection criterion was specified such that the cardinality parameter can only increase when the performance measure of the current iteration, P_t, is larger than, the best performance measure P_b achieved this far. When the results of Table 6 are assessed, it is clear that the AUC increased consistently from iterations $t = 1$ to $t = 3$, which implies that the cardinality parameter was set equal to $c = 3$ at iteration $t = 3$. The consecutive increment limit U for this experiment was specified as $U = 2$, which means that the model will also consider the four-variable and five-variable models produced in iterations 4 and 5 before terminating. However, given that no improvement in the test AUC is recorded for iterations 4 and 5, the cardinality selection step terminates at iteration 3 and it is concluded that the most suitable size for the best subset is three independent variables. If the incrementation limit U was specified to be larger than two, the model would have identified improvements in the test AUC at iterations 6 and 7, which may have indicated that the most suitable value for the cardinality parameter should either be $c = 6$ or $c = 7$. In general, the user should consider a sufficiently large value for U to ensure that the best subset selected by the final model is not too small, sufficiently describes the problem space and that most of the variables that add to the predictive power of the model are considered. However, there also exists a trade-off when using a value for U that is too large, as inflated values for U will result in extended model run times. Model developers should also consider the possibility of an AUC plateau forming after a certain number of iterations, where the addition of more variables to the model does not lead to significant improvements in AUC. In such circumstances, a large U value will produce a cardinality set that is similar to one that is obtained when a small U is used.

6. Empirical results and discussion

In this section, the added benefits of the ALRSF over the standard branch-and-bound best subset selection problem are demonstrated by applying the proposed approach to realistically-sized real-world classification problems. All problem instances considered were obtained from the UCI machine learning repository and cover a wide range of industries including health care, engineering and entertainment.

6.1 Experimental Datasets

Three problem instances are considered, the first of which is the superconductivity regression problem. The dataset is made up of 21263 observations and 81 independent variables. The independent variables describe the attributes of various superconductors including density, atomic mass, electron affinity, fusion heat and thermal conductivity. The real-valued response variable Y_i was transformed into a binary target variable by specifying $Y_i^{bin} = I(Y_i \geq 1400)$ where $I(\cdot)$ is an indicator function.

Three experiments were performed for the transformed superconductivity classification problem. The 21263 observations were subdivided into three training sets with 6500, 8000 and 9000 observations, respectively, while predictive performance was assessed using a hold-out set of 6000 observations. The goal of these three experiments is to demonstrate the flexibility of the ALRSF when solving large problem instances that consist of a large number of independent variables and sizeable training sets. When literature on best subset selection using MIP formulations is reviewed, one will often find that models are either fitted on datasets that contain a large number of independent

variables or a large number of observations, but rarely both. One of the largest problem instances in literature was solved by Venter (2020), with 6000 observations and 58 independent variables that originated from the superconductivity dataset. There is therefore potential to consider a best subset selection problem (within the context of a MIP formulation) that contains a training set that is large with respect to both n and p.

The second problem instance considered is known as the relative location of CT slices regression problem, which describes the CT images for 74 patients. The dataset in question consists of 53500 observations and 385 independent variables. The target variable (location of CT scans) is an integer-type variable and ranges from 0 to 180, where 0 references the soles of the patient's feet and 180 the top of the head. For the purpose of solving classification problems, the integer target variable was transformed to a binary target by specifying $Y_i^{bin} = I(Y_i \geq 50)$.

From the 385 independent variables, a total of 18 unary variables were removed. A further 125 independent variables were removed from the training and test sets based on a VIF threshold of 10. As discussed in Section 4, a functionality was added to allow for the removal of highly correlated independent variables in an attempt to improve numerical stability and statistical interpretability of the final model produced by the framework (Akinwande et al., 2015). This resulted in a final set of 242 independent variables from which the ALRSF can identify the best subset of variables.

Similar to the superconductivity problem instance, three experiments were derived from the total set of available observations, with each training set consisting of 5000, 6000 and 6500 observations, respectively. A test set of 25000 observations was utilised for model evaluation. This problem instance is focused on testing the proposed ALRSF's ability to solve classification problems with a large number of independent variables while still allowing for a moderate number of training observations to be considered.

The third and final round of experiments was conducted by utilising the online news popularity dataset, which consists of 39797 observations. Three training sets comprising 11000, 12000 and 13000 observations, respectively, were then derived. Furthermore, a test set consisting of 24000 observations was used to assess model performance. From the 58 independent variables available, 13 were excluded from the training set based on the VIF exclusion criteria. The continuous target variable (number of shares) was modified to be suitable for a classification problem such that all observations were divided between two classes. The target variable Y_i was transformed by specifying $Y_i^{bin} = I(Y_i \geq 1400)$.The online news popularity classification problem is a moderately sized problem for which a large set of training observations were employed to further validate the proposed solution framework. Additionally, the appropriateness of the ALRSF within application domains that relate to different fields of interest is also explored by considering three separate datasets that relate to vastly different prediction problems.

6.2 Modelling assumptions

The main assumptions associated with the modelling framework (with reference to the experiments in this section) can be summarised as follows:

1. An execution time limit of 345600 seconds for all experiments, except for the CT slice location classification problem, was enforced. For the CT slice location problem, a total of 691200 seconds were allotted due to the size of the problem instance. The execution time of the

ALRSF is directly affected by the number of independent variables and observations.

2. To allow for a comparative study between the proposed ALRSF and the standard branch-and-bound algorithm, the problem-specific parameters (as identified by the ALRSF) are also used for the branch-and-bound algorithm. This includes the cardinality parameter c and the grid range $[-\Gamma, \Gamma]$. It is important to note that for the standard branch-and-bound algorithm, the latter would have to be determined by means of a manual selection procedure while the proposed ALRSF specifies an appropriate grid in an automated fashion.

3. A total of $K = 400$ splines (or line segments) were used to approximate the non-linear log-likelihood objective function. A detailed study with regards to the influence of the value for K can be found in Venter (2020).

4. Cplex only outputs a solution when the problem either terminates based on a predefined termination criteria or when the algorithm has proven optimality. When a problem instance fails (e.g. in the case of an out-of-memory error), a solution cannot be extracted from the Cplex C++ concert technology API and the problem will have to be re-executed with a predefined stopping criteria. This was the case for most branch-and-bound experiments against which the ALRSF is benchmarked. When an experiment failed on an out-of-memory exception, the instance is re-executed, but with a set time limit to allow the user to extract the best solution that was found before termination.

5. The integer conversion constant H (as discussed in section 4) is arbitrarily set equal to 100000 for the experiments considered in this section. In Section 5 we noted that the value of H has a negligible effect on the performance of the final model, which means that a value for H can be arbitrarily set by the user. A larger value for H will, however, better approximate the true underlying logistic regression parameters.

6. The grid point adjustment constant ψ was selected to be equal to 10. This is to facilitate a gradual increase in the grid range $[-\Gamma, \Gamma]$ while simultaneously reducing the number of iterations required to accurately discover the appropriate solution search space.

7. The incremental threshold constant U, was set equal to a value of 10 based on preliminary verification experiments. A value of $U = 10$ implies that the cardinality selection step can terminate when no improvement in model performance is recorded with the consecutive inclusion of up to 10 independent variables to the best variable subset (in addition to the predictors that are already contained within the best subset). The empirical results that follow seem to suggest that a value of $U = 10$ is reasonable, even for large datasets.

6.3 Empirical results

Table 7 and Figures 3 to 5 provide a summary of the results that were obtained for the experiments that were described in Section 6.1. Table 7 contains key assessment criteria, such as memory usage, optimality gap, train and test AUC and best integer solution, that can be used to compare the proposed ALRSF with the standard best subset selection branch-and-bound algorithm.

In general, the memory usage required by the proposed ALRSF is much lower than the standard branch-and-bound algorithm. In some cases, a significant reduction in memory of up to 10 GB is

observed. This, in turn, allows for larger datasets to be considered when attempting to fit a logistic regression model following a best subset selection approach. For the superconductivity classification problem with a training size of 9000 observations, the standard branch-and-bound algorithm was not able to find a solution as a result of insufficient memory (the reader will note that an entry of "ME" in Table 7 indicates that a "memory exceeded" exception occurred). In contrast, the ALRSF was able to find a high-quality solution with a memory buffer of 1.8GB from the available memory of 32 GB still available. Similarly, for both the CT slices and online news popularity classification problems (6500 and 13000 training observations, respectively), much shallower solutions that exist within the solution tree were found when utilising the standard branch-and-bound algorithm, while high-quality solutions were obtained when applying the ALRSF. This is further corroborated by critically evaluating the optimality gap and best integer solution obtained for these two datasets in Table 7. A significant improvement with regard to the optimality gap was observed, especially for the superconductivity problem.

When the train and test AUC is considered, it can be seen that the performance between the two algorithms is relatively similar for smaller datasets. The superiority of the ALRSF with respect to model performance is, however, evident when using the solution framework to solve large-scale problem instances. The algorithmic improvements made to the standard branch-and-bound algorithm (as described in Sections 3 and 4) clearly add to the flexibility and robustness of the solution framework. For the superconductivity problem, the test AUC obtained by the ALRSF ranged between 86.47% and 87.21%. A slight increase in test AUC is observed with an increase in the set of training observations, which may be attributed to more training data that are more representative of the underlying population. A relatively similar trend is observed for the CT location and news popularity classification problems. For the CT slice location problem, the test AUC increased from 89.04% to 94.34% with an increase in the number of training observations from 5000 to 6500 records. Finally, we note that the ALRSF was able to yield AUC values between 68% and 69% for the online news popularity dataset, which are comparable to AUC values of the models that were produced by the branch-and-bound algorithm.

It is also worth noting that the ALRSF is able to dynamically adjust the best variable set based on the available training data and the problem instance considered. The cardinality parameter c varied between experiments, which suggests that the solution framework is able to detect changes in trends contained in the underlying data and adjust the proposed model accordingly to best describe the problem instance. The cardinality selection criteria will influence the size of the final model selected. For the online news popularity problem, almost all of the independent variables were included in the final model. This is because of consistent slight increases in the test AUC that are recorded with the addition of each new variable, as seen in Figure 5.

A possible solution that will yield more parsimonious models might be to use a less stringent cardinality selection criteria which stipulates that the increase in the AUC must be larger than some predefined percentage before a variable is added to the model, e.g. the AUC must increase by 10% from iteration $t-1$ to iteration t before an additional variable is considered for inclusion.

The ability of the ALRSF to solve a problem instance with 242 independent variables and 6500 training observations is considered to be a noteworthy achievement for the best subset MIP problem formulation, given that standard branch-and-bound methods will likely fail when presented with such a large number of predictors and observations. Similarly, for the superconductivity classification

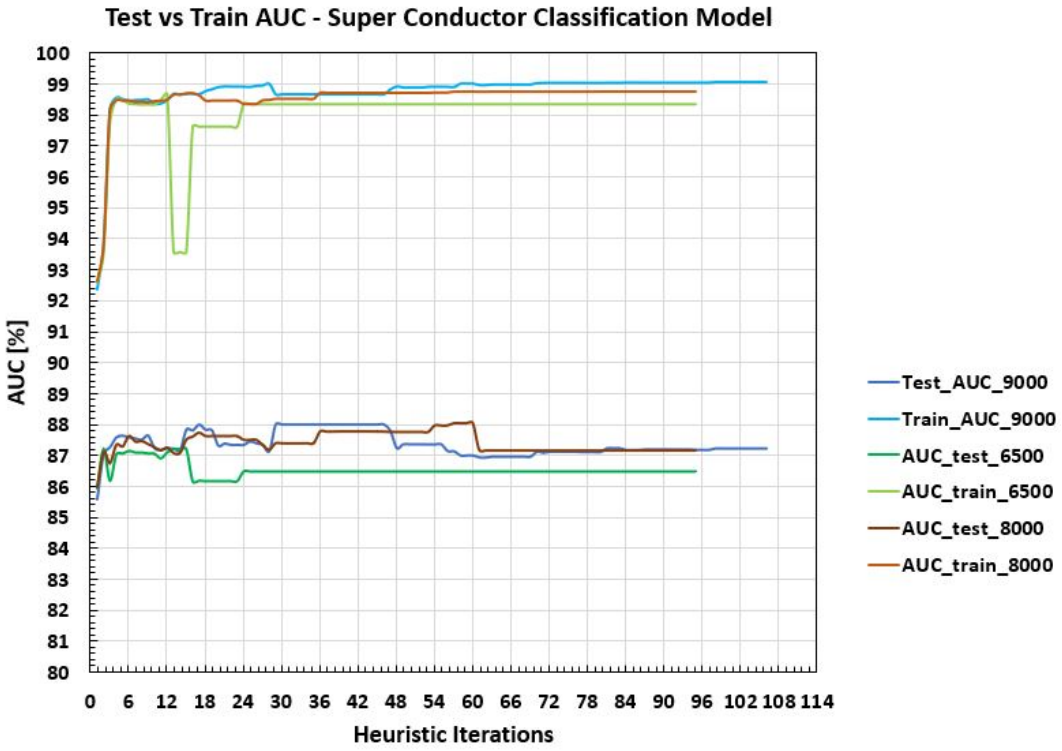


Figure 3. Superconductivity classification problem.

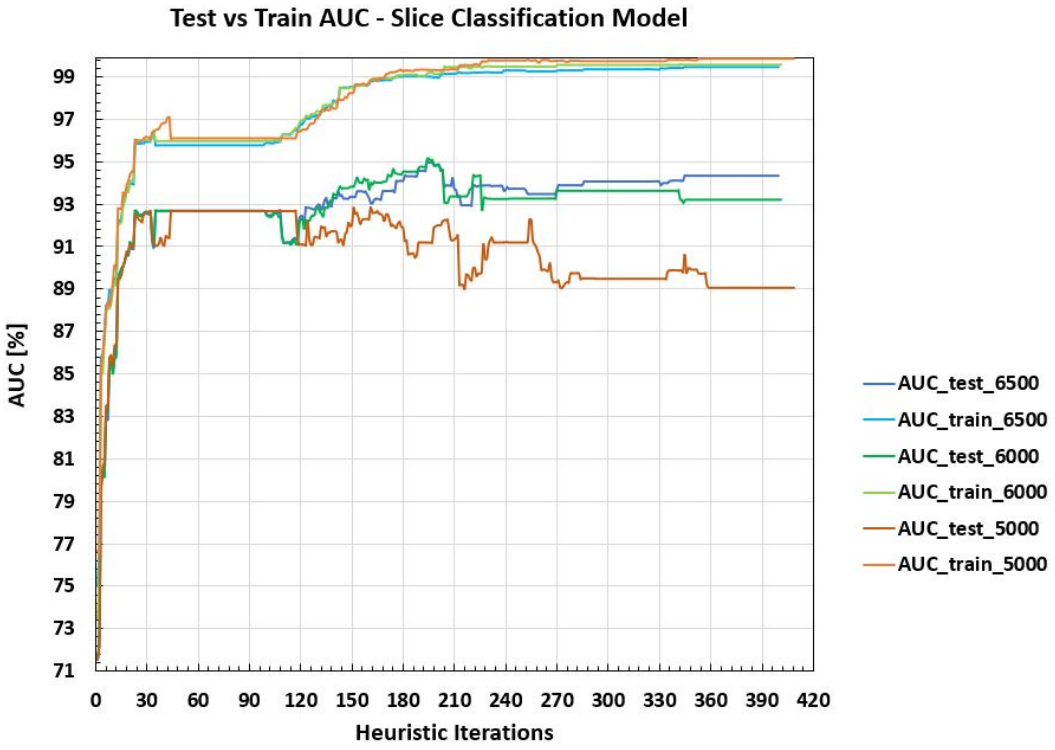


Figure 4. Relative CT slices location classification problem.

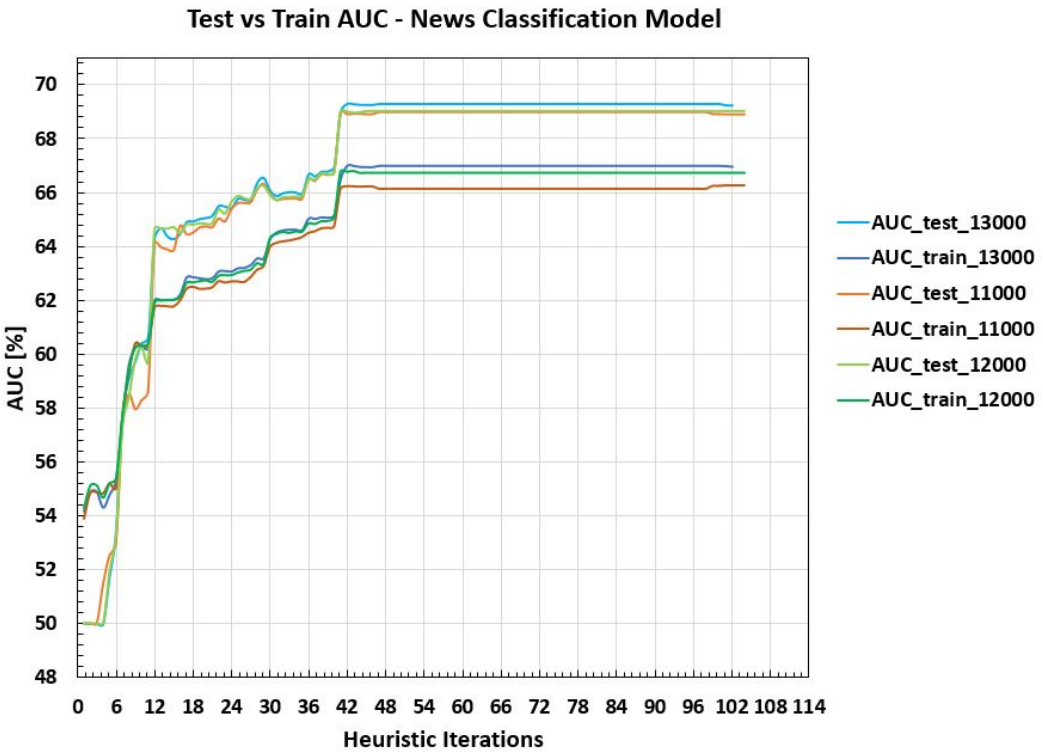


Figure 5. Online news popularity classification problem.

problem, a problem instance with 81 independent variables and 9000 training observations was successfully solved. The aforementioned solutions can be considered as some of the largest problems that were successfully executed within a best subset selection MIP framework (when compared to the available literature on similar formulations). The ALRSF pushes the boundaries concerning computational efficiency and increases the type and size of problems to which a best subset logistic regression solution framework can be applied. Figures 3 to 5 are a visual representation of the iterative solution process contained within the ALRSF that was described in Section 4. It is important to note that the iterative approach allows the user to continuously monitor the performance of the model as fitting progress, which in turn can also be used to detect overfitting on the training set. The cardinality parameter c for each experiment is listed in Table 7. Figures 3 to 5 illustrate the transition from the cardinality step $S = 0$ to the search space pruning step $S = 1$ when a value of $U = 10$ is added to the problem formulation. It is evident from the visual representations of the solution framework that adding more independent variables to the best variable subset will not necessarily lead to increased predictive performance. The aim of the solution framework is therefore to produce parsimonious models in an automated fashion that are still sufficiently predictive when applied to unseen data. Figure 4 clearly shows an instance of overfitting, where an increase in the training AUC is accompanied by a drop in the test AUC when 5000 training observations are considered. In general, the visual representation of the solution framework supports the notion that an increase in the amount of training data results in an increase in the model performance. Nevertheless, the ALRSF equips the user with analytical capabilities to terminate the solution framework at any given point in time when an acceptable solution is found, which can be used to address the problem of overfitting.

This means that the ALRSF is also more interpretable, controllable and transparent when compared to the standard branch-and-bound approach.

7. Conclusion and recommendations

A novel automated logistic regression solution framework was presented that allows for a holistic deterministic approach towards solving the best subset selection MIP problem formulation (presented in Section 3) of a binary logistic regression classification problem. The ALRSF is executed via a three-step approach, which involves the following:

1. The automatic specification of a cardinality parameter c and selection of the best variable subset based on model performance.

2. Pruning of the search space described by the MIP formulation, which is achieved via a Benders decomposition regression parameter fixing pruning algorithm.

3. The calculation of an optimality gap in order to provide a guarantee of optimality, which is not possible for heuristic variable selection methods (such as stepwise, lasso and ridge regression) that do not employ MIP formulations. Furthermore, the optimality step also attempts to improve the solution that was discovered by the pruning algorithm in the previous step.

Empirical results presented in Section 6 show that significant improvements in the final logistic regression solution are achieved when the ALRSF is applied instead of the standard branch-and-bound model. Specifically, the ALRSF allows for better solutions to be achieved for larger problems, requires less memory during execution, and, overall, obtains higher quality solutions with lower optimality gaps due to its efficiency and ability to effectively manage the solution search space. The ability to efficiently identify an optimal subset of independent variables from large-scale datasets that contain many observations and variables by using a deterministic solution framework may incentivise more users to consider exact methods for model-fitting exercises, as such methods are able to provide the user with a measure of optimality (which is not possible when using heuristic fitting algorithms). Another consideration is minimising the need for highly skilled resources to facilitate model development by employing a more automated framework.

The proposed solution framework naturally lends itself to other modelling techniques such as linear regression and even possibly neural networks. Future endeavours can, therefore, be concerned with the expansion of the proposed ALRSF to accommodate more than one model specification within an exact best subset selection MIP problem formulation. Expanding the proposed MIP formulation to allow for best subset selection within the context of multi-class classification problems can also be explored.

The search space pruning heuristic mainly focuses on reducing the lower bound of the MIP when formulated as a maximisation problem (which is the case for the log-likelihood function). As a possible area for improvement, the addition of Lagrangian relaxation during the optimality step ($S = 2$) to tighten the MIP upper bound can be considered. The cardinality constraints are seen as complicating constraints, the effect of which can be managed more appropriately by possibly including these constraints as a Lagrangian relaxation in the objective function. This might further reduce the optimality gap and the time required to prove optimality.

Table 7. ALRSF vs. B&B – experimental results.

Instance Identifier/ observations	Cardinality / Total independent variables (After VIF Exclusion) / VIF Excluded	Memory Used (Gb)	Integer Solution	Gap (%)	Training Accuracy (%)	Training AUC	Test Accuracy (%)	Test AUC
Superconductivity Classification Problem								
1-ALRSF/ 6500	2/81/0	25.4	-808.75	42.48	95.31	98.34	83.77	86.47
1-B&B/ 6500	2/81/0	>32	-1819.12	75.40	86.11	92.77	77.93	86.10
2-ALRSF/ 8000	6/81/0	28.1	-823.64	30.73	96.35	98.78	84.95	87.17
2-B&B/ 8000	6/81/0	>32	-2260.73	74.47	89.49	94.05	82.32	87.54
3-ALRSF/ 9000	17/81/0	30.2	-773.22	23.52	97.11	99.07	83.45	87.21
3-B&B/ 9000	17/81/0	>32	ME	ME	ME	ME	ME	ME
Relative CT Slice Location Classification Problem								
1-ALRSF/ 5000	32/242/125	26.4	-127.75	100.00	99.20	99.84	87.41	89.04
1-B&B/ 5000	32/242/125	>32	-1123.80	99.96	90.82	96.74	84.04	91.23
2-ALRSF/ 6000	23/242/125	31.1	-359.40	97.82	97.83	99.56	89.56	93.19
2-B&B/ 6000	23/242/125	>32	-4158.88	99.81	50.92	50.00	60.62	50.00
3-ALRSF/ 6500	23/242/125	>32	-439.70	95.92	97.45	99.47	90.16	94.34
3-B&B/ 6500	23/242/125	>32	-3558.65	99.69	77.32	81.79	71.82	77.22
Online News Popularity Classification Problem								
1-ALRSF/ 11000	41/45/13	20	-7100.34	OPT	63.06	66.25	63.63	68.89
1-B&B/ 11000	41/45/13	>32	-7107.75	0.11	62.90	66.13	66.13	68.89
2-ALRSF/ 12000	45/45/13	22	-7729.86	OPT	62.94	66.75	64.91	69.01
2-B&B/ 12000	45/45/13	30	-7729.86	OPT	62.94	66.75	64.91	69.01
3-ALRSF/ 13000	42/45/13	31.9	-8388.35	OPT	63.07	66.96	57.68	69.25
3-B&B/ 13000	42/45/13	>32	-8956.39	6.34	54.67	50.00	53.00	50.00

In conclusion, the proposed novel ALRSF exhibits multiple improvements with respect to the best subset selection problem (which is an NP-hard problem) within the context of an exact mathematical solution framework. The ALRSF was able to solve large-scale logistic regression classification problems in an efficient and automated manner while providing higher quality solutions when compared to other best subset selection MIP formulations that are commonly found in the literature.

References

Akinwande, O. M., Dikko, G. H., and Samson, A. (2015). Variance inflation factor: As a condition for the inclusion of suppressor variable(s) in regression analysis. *Open Journal of Statistics*, **5**, 754–767.

Bertsimas, D., King, A., and Mazumder, R. (2016). Best subset selection via a modern optimization lens. *Annals of Statistics*, **44**, 813–852.

Civitelli, E., Lapucci, M., Schoen, F., and Sortino, A. (2021). An effective procedure for feature subset selection in logistic regression based on information criteria. *Computational Optimization and Applications*, **80**, 1–32.

Dedieu, A., Hazimeh, H., and Mazumder, R. (2021). Learning sparse classifiers: Continuous and mixed integer optimization perspectives. *Journal of Machine Learning Research*, **22**, 1–47.

Furnival, G. M. and Wilson, R. W. (2000). Regressions by leaps and bounds. *Technometrics*, **42**, 69–79.

Gatu, C. and Kontoghiorghes, E. J. (2006). Branch-and-bound algorithms for computing the best-subset regression models. *Journal of Computational and Graphical Statistics*, **15**, 139–156.

Hastie, T., Tibshirani, R., and Tibshirani, R. (2020). Best subset, forward stepwise or lasso? Analysis and recommendations based on extensive comparisons. *Statistical Science*, **35**, 579–592.

Heinze, G., Wallisch, C., and Dunkler, D. (2018). Variable selection – a review and recommendations for the practicing statistician. *Biometric Journal*, **60**, 431–449.

Insolia, L., Kenny, A., Calovi, M., and Chiaromonte, F. (2021). Robust variable selection with optimality guarantees for high-dimensional logistic regression. *Stats*, **4**, 665–681.

Kamiya, S., Miyashiro, R., and Takano, Y. (2019). Feature subset selection for the multinomial logit model via mixed-integer optimization. *In The 22nd International Conference on Artificial Intelligence and Statistic*. PMLR, Okinawa, Japan, 1254–1263.

Kolacek, J. and Rezac, M. (2010). Assessment of scoring models using information value. *In 19th International Conference on Computational Statistics*. Elsevier, Paris, France, 1191–1198.

Kutner, M., Nachtsheim, C., Neter, J., and Li, W. (2005). *Applied Linear Statistical Models*. McGraw-Hill.

Kvesic, L. and Gordana, D. (2012). Risk management and business credit scoring. *In In Proceedings of the ITI 2012 34th International Conference on Information Technology Interface*. IEEE, Cavtat, Croatia, 47–52.

Lund, B. (2017). Logistic model selection with SAS PROC's LOGISTIC, HPLOGISTIC HPGENSELECT. *In 2017 Midwest SAS Users Group Conference*. ACM, St. Louis, MI.

Miller, A. (2002). *Subset Selection in Regression*. CRC Press, Washington DC.

Naganuma, M., Takano, Y., and Miyashiro, R. (2021). Feature subset selection for ordered logit model via tangent-plane-based algorithm. *IEICE Transactions on Information and Systems*, **80**, 1–2.

Natarajan, B. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, **24**, 227–234.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, **259**, 801–817.

Sato, T., Takano, Y., Miyashiro, R., and Yoshise, A. (2016). Feature subset selection for logistic regression via mixed integer optimization. *Computational Statistics and Data Analysis*, **64**, 865–880.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, **45**, 427–437.

Stoltzfus, J. C. (2011). Logistic regression: A brief primer. *Academic Emergency Medicine*, **18**, 1099–1104.

Takano, Y. and Miyashiro, R. (2020). Best subset selection via cross-validation criterion. *TOP*, **28**, 475–488.

Taşkin, Z. C. (2010). Benders decomposition. *In Wiley Encyclopedia of Operations Research and Management Science*. Wiley, Hoboken, NJ.

Venter, J. v. (2020). *Variable selection in logistic regression using exact optimisation approaches*. Ph.D. thesis, North-West University.

Zhang, Z., Trevino, V., Hoseini, S. S., Belciug, S., Boopathi, A. M., Zhang, P., Gorunescu, F., Subha, V., and Dai, S. (2018). Variable selection in logistic regression model with genetic algorithm. *Annals of Translational Medicine*, **6**, 45–57.

Manuscript received 2022-07-27, revised 2023-03-16, accepted 2023-04-17.