Developing New Methodologies for Rank-Based Classifiers

by

Keith Uzelmann

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfilment of the requirements of the degree of

Master of Science

Department of Statistics
University of Manitoba
Winnipeg

Abstract

This thesis concerns a novel approach to addressing the issue of class imbalance in machine learning, in particular by modifying the sampling technique itself. Through the use of convex optimization, statistical learning theory, and numerical simulation, the technique of Nomination Sampling is applied to the Logistic regression and Support Vector Machine classifiers to obtain a better training set. Further, this sampling technique is based off of expert opinion, which until now has received only minimal attention in the machine learning community.

Efficient algorithms for solving the proposed learning problems are given, and numerical studies are performed to validate the efficacy of the methods presented.

Acknowledgment Page

I would like to thank foremost my supervisor, Dr. Mohammad Jafari-Jozani for this tireless guidance in the development of this thesis. The creation of this work was a long, meandering road with innumerable dead-ends and detours. Its completion was a challenge to say the least, and it could have never been realized without his direction and dedication.

I would also like to thank the Department's Graduate Assistant, Rosa Di Noto, who went above and beyond to support me during my program, especially in the final leg.

I would also like to thank the Department of Statistics for providing generous support, both through stipends and by affording me the opportunity to instruct throughout my program. My teaching experience deeply informs my relationship with the field of statistics and I am forever grateful for the opportunity.

Dedication Page

To my parents, without whom I would have nothing.

Contents

Contents iii

List of Figures vii

1 Introduction 1

1.1 Motivation 1

1.2 Nomination Sampling 6

1.3 Threshold Modelling 7

1.4 Loss and Statistical Learning Theory 9

1.5 Organization of the Thesis 11

2 Conditional Class Probabilities 13

2.1 Introduction 13

2.2 Developing The MaxNS and SRS Link 18

2.3 An Analytic Example 23

2.4 MaxNS Probabilities under Normality 27

2.5 Robustness against Model Assumptions 33

3 Logistic Regression with MaxNS Data **37**

3.1 Introduction 37

3.2 Logistic Regression with MaxNS Data 42

3.3 Solving the MNSLR Problem 43

3.4 Convergence Analysis 45

3.5 Regularization 52

3.6 Loss 55

4 The Support Vector Machine **57**

4.1 The Hard-Margin Support Vector Classifier 57

4.2 The Soft-Margin Support Vector Classifier 60

4.3 Duality and Computation of the SVC 62

4.3.1 The KKT Conditions 64

4.3.2 Applying Lagrange Duality to the Soft-Margin SVC 65

4.4 Nonlinear Classification via The Kernel Trick 68

4.5 The SVM and Loss Functions 72

4.6 The MaxNS SVM 74

4.7 Summary 82

5 Simulated Data Analysis **83**

5.1 Simulation: Logistic Regression 83

5.2 Simulation: SVM 88

6 Summary and Future Research **93**

6.1 Summary 93

6.2 Future Research 94

Bibliography **97**

List of Figures

1.1 The Sigmoid Function 3

1.2 SVM Example 4

1.3 Popular Loss Functions 10

2.1 Conditional Class Probabilities: An Analytic Example 26

2.2 Conditional Class Probability under Normality 28

2.3 Comparison of MaxNS and SRS Conditional Class Probability Functions . . 29

2.4 Comparison of Conditional Class Probability Function to Approximation . . 30

2.5 Further Comparisons to Approximation 31

2.6 Testing Conditional Class Probability for Robustness in X 34

2.7 Testing Conditional Class Probability for Robustness in $\mathcal{E}$ 35

2.8 Testing Conditional Class Probability for Robustness in X and $\mathcal{E}$ Simultaneously 36

3.1 Linear Probability Classification: Example 38

3.2 Logistic Regression Classification: Example 41

3.3 Gradient Descent: Example 44

3.4 Convex Functions: Example 46

3.5 Lipschitz Constant from Result 11 51

3.6 Ridge Regression: Example 54

4.1 Comparison of Several Separating Lines 58

4.2 Visualization of Margin in SVM 59

4.3 Nonlinear Classification Problem 68

4.4 Kernel Trick: Example 69

5.1 ROC Analysis: Example 85

5.2 ROC Analysis: MNSLR, $n = 15$ 87

5.3 ROC Analysis: MNSLR, $n = 40$ 88

5.4 ROC Analysis: SVM, $n = 15$ 89

5.5 Error Rate for MNSSVM, $d = 2$ 91

5.6 Error Rate for MNSSVM, $d = 5$ 92

Chapter 1

Introduction

In this chapter, we will motivate the work behind this thesis. We will introduce in simple terms the preliminary theory needed to approach the topics of this thesis, including Logistic regression, Support Vector Machines, threshold modelling, nomination sampling, and statistical learning theory. These topics will be expanded upon as they are needed.

1.1 Motivation

Broadly speaking, *classification* in statistics refers to the techniques by which we attempt to model and predict the class to which a datapoint belongs based on one or more input variables, known as *features*. Classifiers use an available dataset with known features and class labels, called a *training set*, and from this dataset build a *classifier* that may predict the class of a future observation using only its features. Classification techniques are used to address an ever-expanding range of real-world problems, from speech recognition to the identification of breast cancer from mammograms. An array of methods has been developed to address the goals of classification. These methods range from early approaches such as Fisher's Linear Discriminant Analysis (LDA) to more modern techniques such as the Perceptron algorithm [1], Logistic regression, and the Support Vector Machine (SVM), the latter two of which will

be the primary focus of this research.

Logistic regression, in its standard formulation, aims to fit a *sigmoid* function between a linear combination of the features $\boldsymbol{x} \in \mathbb{R}^d$ and a binary response $y \in \{0, 1\}$. In particular, the goal is to fit a function of the form

$$\sigma(\boldsymbol{x}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}$$

between the features and the response. See Figure 1.1 for an illustration when the dimension of the feature space is $d = 1$. The benefit of this technique is that the fitted values will be between 0 and 1 due to the nature of the sigmoid function. These values may then be interpreted as a modelling of the *conditional class probabilities* of the response Y given a value of $\boldsymbol{X}$. I.e.,

$$\sigma(\boldsymbol{x}) \approx \mathbb{P}(Y = 1 \,|\, \boldsymbol{X} = \boldsymbol{x}).$$

Observations of sampling unites with values of $\boldsymbol{x}$ such that $\sigma(\boldsymbol{x}) > 0.5$ will be classified as $Y = 1$, and those with $\sigma(\boldsymbol{x}) < 0.5$ will be classified as $Y = 0$.

Logistic regression is a popular technique in classification due to its ease of interpretation, and it is widely available across many common statistical software packages (e.g., SAS, STATISTICA, R, among others). Logistic regression finds modern applications within fields such as medicine [2, 3, 4, 5], finance [6, 7, 8], the social sciences [9, 10], and nutrition [11], among others. The application of Logistic regression may be extended through the use of kernels [12], and regularization techniques may be applied to Logistic regression to perform variable selection [13]. However, it it known [14] that Logistic regression will display bias in the case of *data imbalance*. That is, if there is a discrepancy between the proportions of 1's and 0's in the response variable, the Logistic model will display bias towards the majority class. This is a property which is shared with a more sophisticated statistical learning technique, the SVM.

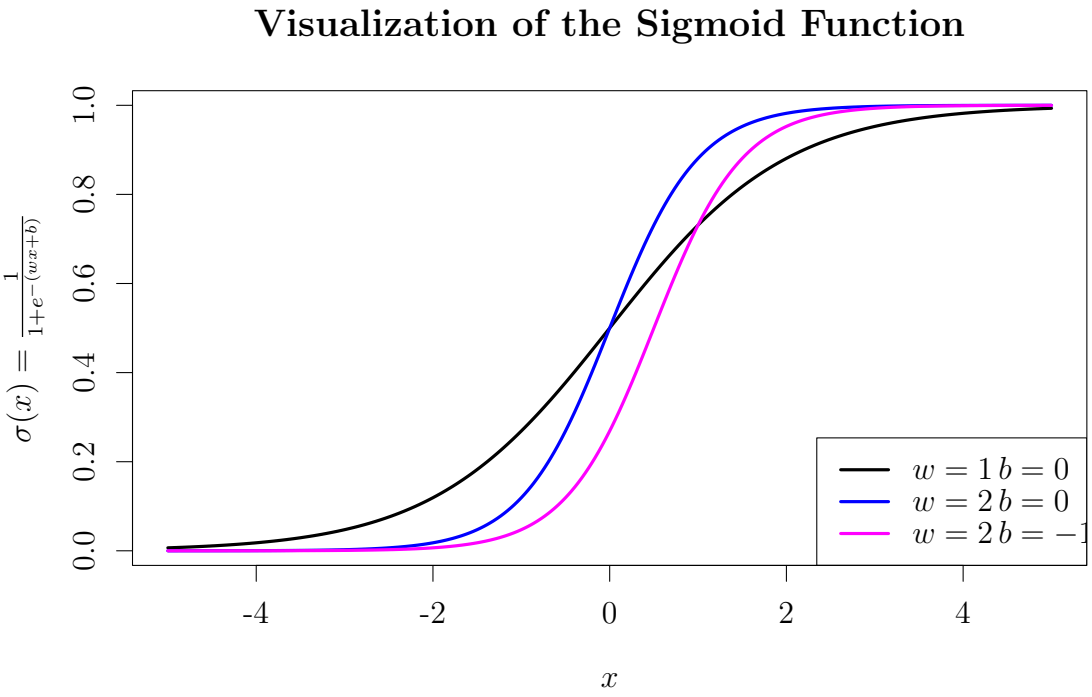

Figure 1.1: Shown here is the sigmoid function, with a linear rescaling in x. One can see that the function is a smooth and monotone mapping from $\mathbb{R}$ to $[0, 1]$, making it a candidate for modelling conditional probabilities.

The SVM, originally proposed by Boser et al. [15], achieves classification by determining a *decision boundary* in the feature space; data points that lie on one side the decision boundary are assigned to one class, and data points that lie on the other side of the decision boundary are assigned to another class. In particular, a decision boundary is determined that seeks to maximize the distance between the two classes of the dataset. Two examples of the SVM, one linear and one non-linear are given in Figure 1.2.

In recent years, SVMs have seen an explosion of interest within the machine learning community due to their strong mathematical foundation, but even more importantly their ability to solve a wide range of classification problems, even in high-dimensional and non-linear settings. Applications include facial recognition [16], text recognition [17, 18], bio-medicine [19], cancer recognition [20], and numerous engineering applications [21]. Further, the SVM has been shown to be robust against (though not immune to) the so-called *curse of*

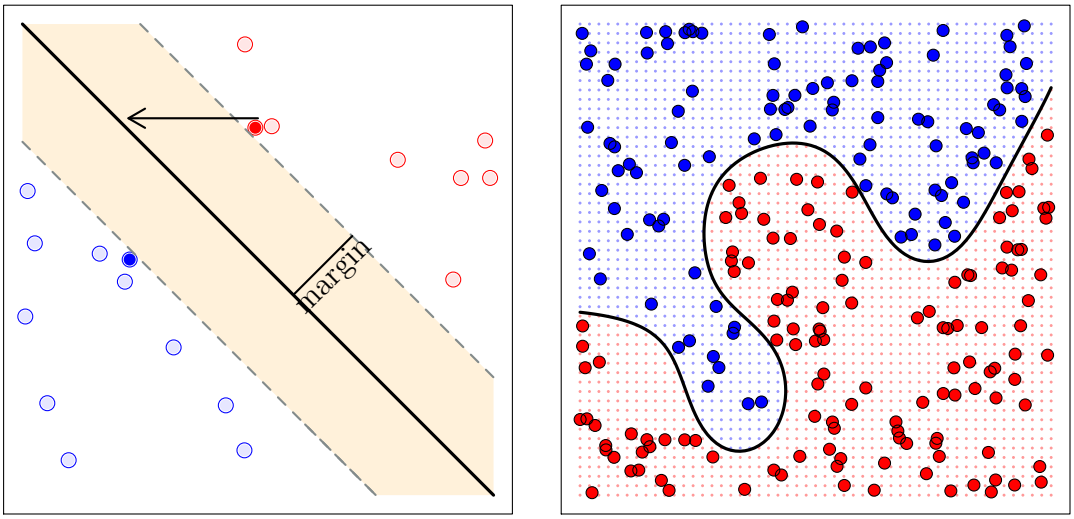

Figure 1.2: Two examples of the SVM classifier. In the left panel, we see a linear classifier between the two classes, with class represented by colour. The distance from the decision boundary to the closest data point is the margin of the classifier: the SVM seeks to maximize the margin of the classifier. The highlighted points are the so-called support vectors of the classifier. On the right panel, we see an example of a non-linear classifier. By casting the dataset into a higher dimensional feature space, the SVM is able to calculate a linear classifier in the modified space, and then project this classifier back down into the original feature space. These techniques will be discussed in detail in Chapter 4.

dimensionality [22], i.e., the tendency for statistical learning procedures to break down in high-dimensional settings. Finally, since only a subset of the datapoints contribute to the classifier, the SVM tends to be memory efficient.

However, SVMs are also subject to drawbacks, such as sensitivity to noise in the data (both in the features and the labels) [23], reliance on the use of tuning parameters [24], and computational inefficiency within large datasets [25]. Several modifications to the SVM algorithm have been proposed to address these deficiencies, such as [23, 24, 25, 26, 27, 28]. The shortcoming with which this research is concerned is that of *class imbalance*, as with Logistic regression. It has been observed in [29] and in subsequent research that when there is class imbalance in the SVM training dataset, the decision boundary found by the SVM algorithm will skew towards the minority dataset, and performance of the model with respect to the minority class will be degraded. This is an especially undesirable property, as a minority

class is often the class that we are most interested in detecting (such as in identification of defective products in quality control, or identification of diseases or risk factors in medical imaging).

An overview of the techniques which have thus far been applied to the issues stemming from class imbalance in SVMs may be broadly categorized into two approaches: *data pre-processing methods*, and *algorithmic modifications*. Data pre-processing methods consist primarily of re-sampling techniques, as seen in [30, 31]. These techniques have the advantage of being able to implement pre-existing SVM algorithms without a need to redesign the learning program. Another pre-processing method involves building classifiers from multiple subsets of the training data, and then collecting these smaller classifiers into an *ensemble* from which a final decision is made, e.g. in [32]. Algorithmic modifications, on the other hand, modify the SVM program itself. The most common strategy used is to assign class-specific costs to training examples, as seen in [29]. In particular, misclassification of members from the minority class is subject to a greater penalty than to misclassification of members from the majority class, and in this way, with correct tuning, a more desirable classifier may be obtained. Other modifications to the SVM algorithm have been developed, as seen in [33, 34, 31]. For a more complete and focused overview of these techniques, see [35].

Techniques have also been proposed to assuage the imbalance problem in Logistic regression models. In particular, a rare events correction is proposed in [36], while a cost-sensitive approach is utilized in [37]. Further, the resampling techniques discussed above (i.e. [30]) have also been applied to Logistic regression [38].

This research introduces a technique for addressing class imbalance issues in both Logistic regression and the SVM that lies in between re-sampling and algorithmic modification. In this thesis, we will apply an alternative *sampling* (not re-sampling) technique to the population in order to obtain a training dataset that is more biased towards the minority class, and then introduce a modified algorithm that deals with this biased training set. In particular, we

will apply the technique of *nomination sampling* to obtain a sample that has greater balance than what might normally occur when taking a simple random sample from the underlying population.

1.2 Nomination Sampling

Nomination sampling, introduced by Willemain [39] is explained below. Suppose that we have access to a population $U \coloneqq \{u_i : i = 1, \dots N\}$ of size N, where N is assumed to be very large, if not infinite in size. Rather than draw a single sample of size n from U, we draw n random subsamples of size k. Within each sample, we *nominate* for further consideration a *nominee*. We are left with a total of n nominees, forming our final sample. Formally, we set a function $\Psi : \{x_1, \dots, x_k\} \to x^*$, called a *nominator* that allows us to uniquely select a single element u^* from the sub-sample $\{u_1, \dots, u_k\}$. This nominee will often take the form of the element with the highest or lowest value of z_i, where z is some auxiliary variable associated with x that is easy to observe and can be used for an accurate ranking of sampling units with respect to a variable of interest that is hard to observe. The sub-sample $\{x_1, \dots, x_k\}$ is known as a *set* and k is called the *set size*. We outline the above procedure in the algorithm below:

1. Fix a set size k, and a nominator $\Psi : \{u_1, \dots, u_k\} \to u^*$.
2. Obtain a random sample $\{u_1, \dots, u_k\}$ of size k from the population U.
3. From this random sample, retain the element $u^* = \Psi(u_1, \dots, u_k)$.
4. Repeat steps 2 – 3 until n elements are obtained, called nominated samples.

For example, we may wish to study disease in some population where the occurrence of the disease is rare, say occurring in 2% of the population. Suppose that the measurements required

for this study are expensive (examples might include a bone mineral density measurement), so that we can only afford to measure 200 elements of the population. If we were to take a simple random sample of 200 members from this population, we would expect to only retain four members of the population who have this disease. Instead, if we take 200 sub-samples of size $k = 5$, and nominate from each sample the member who has the highest "likelihood" (according to an expert opinion, or some accessible concomitant) of having this disease, the probability of a nominee having this disease increases from 0.02 to $1 - (1 - 0.02)^5 = 0.096$ (assuming that our ranking is perfect), so our expected number of sample members having this disease increases from 4 to 19.2. This is all with only the additional cost of obtaining the sets and performing the nomination, which we assume to be cheap compared to the cost of the final measurements.

Returning to Logistic regression and SVMs, if we may obtain a training set through a nomination process that allows us to focus on members of the minority population (as in the example above), then we will in turn arrive at a more balanced sample. However, this sample will now have significant bias, which we also propose a method to account for.

We will take a detour here to briefly discuss *threshold modelling* here, which will be the primary way through which we model our "expert opinions".

1.3 Threshold Modelling

In this thesis, a binary variable Y will said to be from a *threshold model* if, for some variable Z and some constant m, Y can be represented as

$$Y = \begin{cases} 1, & \text{if } Z \geq m, \\ 0, & \text{if } Z < m. \end{cases}$$

The constant m will be referred to as the *threshold* of the model. Note that while this model assumes that Y follows a 1/0 coding, we may easily incorporate models with a ± 1 coding

(such as the SVM) with the mapping $Y \mapsto 2Y - 1$.

Many variables in practice do follow this model: for example, one description of obesity will classify an individual as obese if his/her BMI (Body Mass Index) is above 30 [40]; as another example, an individual will be classified as having osteoporosis if his/her BMD (Bone Mineral Density) is 2.5 or more standard deviations beyond the mean for the patient's reference group [41].

In this research we will assume that our response variables are coming from a threshold model, and we will use the variable Z to rank our sampling units. Although in practice it is rare to have access to Z, we will make the assumption here that our expert opinion is capable of achieving the same ranking as would be achieved if we ranked by Z. Thus, the developed models will contain an incorporation of the expert opinion. Indeed, the incorporation of expert opinion into a statistical model for the purpose of classification is no easy task, and we consider this to be a significant aspect of this work.

However, as previously mentioned, the use of Z to nominate sample members will result in a sample that is unrepresentative of the population, as we are intentionally trying to re-balance the training set. As our models require sample data that are representative of the population, using this nominated data directly in the algorithms will result in faulty predictions. Thus, we must make some modifications to the algorithms as they exist.

Before we modify the Logistic regression and SVM algorithms, we will need to understand their components from a statistical perspective. Thus, before we continue, a brief discussion on the basic concepts of Statistical Learning Theory will be appropriate.

1.4 Loss and Statistical Learning Theory

Consider the general situation of supervised learning: we have two sets, a feature space $\mathcal{X}$ and a class space $\mathcal{Y}$, and we wish to find some dependency between $\mathcal{X}$ and $\mathcal{Y}$, written as

$$Y = f(\boldsymbol{X}) + \varepsilon.$$

We call f a *decision function* and ε the *irreducible error*. The function f will take a (often vector-valued) input $\boldsymbol{x}_i$ and return a model $f(\boldsymbol{x}_i)$ of y_i. We wish to find the best such map f. However, a question immediately arises: what exactly do we mean by "best", and how do we measure/quantify this notion? We will do so by a function $L : \mathcal{Y} \times f[\mathcal{X}] \to \mathbb{R}_{\geq 0}$, called a *loss function*.

The map L will receive pairs $(y, f(\boldsymbol{x}))$, and it will return value that, in a sense, evaluates "how wrong" our prediction is. Loss functions play a central role in decision theory in general, and find many uses outside of statistics. In other fields, they are sometimes called "cost functions", or occasionally "regret" functions. In other words, we use the map f to decide how to predict y_i given $\boldsymbol{x}_i$, and L will quantify how much we should regret this decision. Common choices for regression include

$$L(y, f(\boldsymbol{x})) = \begin{cases} (y - f(\boldsymbol{x}))^2, & \text{the } \textit{squared error loss}, \text{ and} \\ |y - f(\boldsymbol{x})|, & \text{the } \textit{absolute error loss}, \end{cases}$$

while some common choices for binary classification (with $y = \pm 1$) include

$$L(y, f(\boldsymbol{x})) = \begin{cases} \mathbb{I}(y \neq \operatorname{sign}\{f(\boldsymbol{x})\}), & \text{the } \textit{0 – 1 loss}, \\ \max(1 - yf(\boldsymbol{x})), & \text{the } \textit{the hinge loss}, \text{ and} \\ \ln\bigl(1 + e^{-yf(\boldsymbol{x})}\bigr), & \text{the } \textit{the logistic loss}. \end{cases}$$

The choice of loss function will play a central role in the decision function found by the statistical learning method.

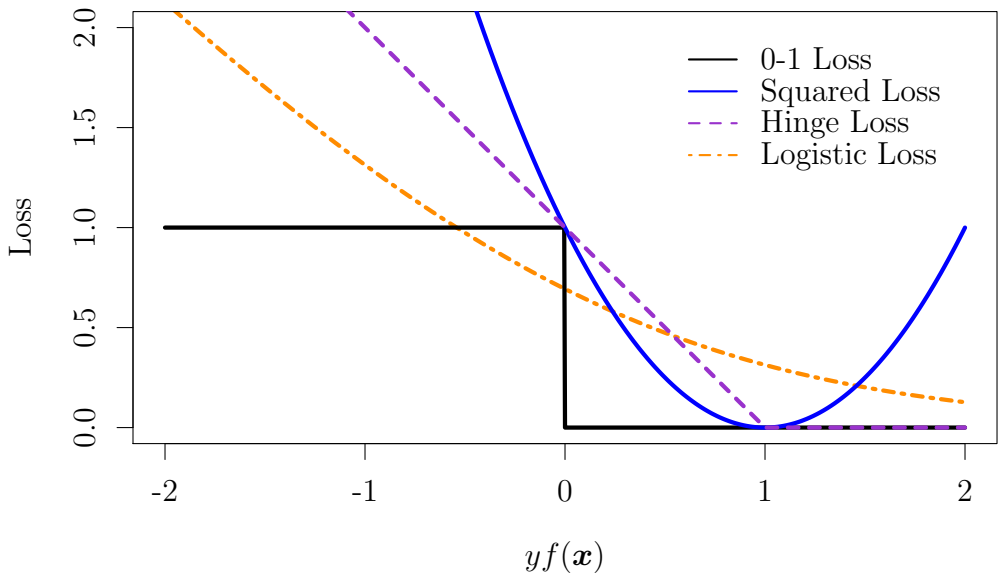

Figure 1.3: Shown here are multiple loss functions used in classification. In black is the 0 – 1 Loss Function, an ideal loss function from an intuitive perspective but often too algorithmically difficult to work with. The Squared Loss function is shown in blue, and is used in ordinary least-squares regression. The Logistic Loss (orange) is used in Logistic regression, and the Hinge Loss (purple) is used by SVMs. We will take special notice here of the asymptotic similarities of the Hinge and Logistic loss functions.

In a binary classification problem, with $y_i \in \{-1, 1\}$, we can often represent the loss function $L(y, f(\boldsymbol{x}))$ as a univariate function of $yf(\boldsymbol{x})$. For example, with a $y = \pm 1$ coding, we may represent the squared loss function as

$$(y - f(\boldsymbol{x}))^2 = (1 - yf(\boldsymbol{x}))^2.$$

This representation also provides a convenient interpretation: when the sign of $f(\boldsymbol{x})$ matches that of y, the product $yf(\boldsymbol{x})$ will be positive. Incorrect predictions will correspond to negative values of $yf(\boldsymbol{x})$. In Figure 1.3, some of the loss functions commonly used in statistical learning are compared.

Suppose that we have used a training set $\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \subseteq \mathcal{X} \times \mathcal{Y}$ to build f. The training error is defined to be

$$\frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i)),$$

and evaluates the average loss, or *empirical risk* associated with the chosen decision function f over the training set $\mathcal{T}$. We are, however, more interested in the *test error*, defined to be

$$\mathbb{E}[L(Y, f(\boldsymbol{X}))|\mathcal{T}],$$

where $\boldsymbol{X}$ and Y are drawn randomly from their populations, $\mathcal{X}$ and $\mathcal{Y}$, respectively. The test error, also known as the *generalization error* of the decision function f evaluates how well our decision function performs over the entire population, according to the loss function of choice.

Our final goal in statistical learning theory is to determine a decision function that will minimize the test error over all possible decision functions. In this research we will dedicate considerable time to developing a loss function appropriate for our purposes. We develop our theoretical results over a specific form of nomination sampling, Maxima Nominated Sampling (MaxNS), which has recently been studied extensively in the literature and has shown promising results in quantile regression analysis [42] and quality control [43]. However, the results of this work can easily be extended to Minima Nominated Sampling (MinNS) using a simple modifications.

1.5 Organization of the Thesis

The thesis will be organized as follows. In Chapter 2 we will explore the connection between the conditional class probabilities for SRS data to NS data, and highlight an error in the existing literature on the matter. In Chapter 3 we will take advantage of this connection to develop a modified Logistic regression algorithm that properly handles NS data. In Chapter 4, we will develop a modified SVM algorithm using this connection. In Chapter 5 we will perform simulation studies to illustrate the performance of our modified method to the standard methods. Finally, in Chapter 6 we will summarize the thesis and provide a discussion on the future of this line of research.

Chapter 2

Conditional Class Probabilities

In this chapter, using numerical and theoretical examples, we show that the relationship between MaxNS and SRS unconditional class probabilities does not carry into the conditional setting. Hence, we will derive a relationship between the conditional class probabilities for MaxNS data and those for SRS data. This relationship will be used to derive a rank-based surrogate loss function which then will be used to derive a appropriate models for Logistic regression and SVMs built upon MaxNS Data. We examine the robustness of our proposed relationship with respect to the distributional assumptions made on the model.

2.1 Introduction

Often researchers have access to primary assessments of area-specific experts, results of earlier surveys, or inexpensive measurements related to the problem of interest. These can be used to assign ranks or degrees of importance to already observed data, or identify more representative samples from the underlying population for further study. Rank-based sampling (RBS) designs provide a variety of data collection techniques for such situations with the help of expert knowledge and rank information. RBS designs, such as ranked set sampling (McIntyre), randomized nomination sampling (Jafari Jozani and Johnson) and judgement

post-stratification (MacEachern et al.), have been applied in many areas of application, including medical research, environmental studies and image processing. The ultimate goal is to actively embed the rank information into the learning process in order to either increase the efficiency or reduce the cost of study. However, most of the methods developed using RBS data are only applicable for situations in which both the training and test data are of the same nature. In practice, one might be able to obtain RBS data and train an efficient classification model, but if using the model for unseen data requires ranking the observation, most likely the developed method will have no practical use. This is simply because in many applications, test data forms a simple random sample (SRS). So, it is essential to develop efficient predictive or classification models using RBS training data that can be applied on SRS test data. A number of studies are done in the literature that tackle this problem. For example, Jafari Jozani et al (2018) in a quantile regression context used minima nominated samples to study the lower tail of the underlying population. Using an innovative approach, they built a new rank-based check function $\rho_{k,\tau}(\varepsilon)$ to incorporate the rank information of such samples into their study:

$$\rho_{k,\tau}(\varepsilon) = \left(\tau - \frac{1}{2}\right)\varepsilon + \frac{1}{2}|\varepsilon| - \frac{(k-1)}{k}\ln\left\{\frac{e^{(\tau-1)\varepsilon} - \tau}{1-\tau}\right\}\mathbb{I}(\varepsilon \leq 0),$$

where ε is an instant learning error when estimating the τ-th quantile of the population, and k is the set size. They developed an M-estimation algorithm for model fitting and used their proposed methodology to estimate the prevalence of osteoporosis, which is a growing international concern with aging populations. Developing predictive algorithms for osteoporotic fracture risk is pivotal in avoiding morbidity, mortality and economic burden from preventable fractures. One can use clinical risk factors such as body mass index (BMI), fracture risk assessment scores, smoking status and family history to identify (through a random process) more representative patients for the study and then take the more expensive bone mineral density (BMD) measurements from dual x-ray absorptiometry (DXA) images of this cohort for model building. They first biased the training sample towards patients

who have the lowest BMD values among a small number of randomly selected set of patients (of size k) from the study cohort and then adjusted for the selection bias using their new check function. They showed that their method is highly efficient and requires a significantly smaller number of training samples (about one tenth) to produce results that are comparable with commonly used methods in the literature.

Despite the vast literature on rank-based statistical methodologies, very little effort has been devoted to studying predictive models and developing classification techniques for rank-based data. The most pressing problem is how to incorporate the rank information into the analysis.

An important shortfall in the literature pertinent to RBS designs concerns rank-based predictive models for discrete response variables, in particular binary responses for classification. Consider a threshold model

$$Z = h(\boldsymbol{X}) + \mathcal{E},$$

where Z is a continuous response and $\mathcal{E}$ is a continuous "error" term with a symmetric distribution (e.g., Normal, Laplace) with expectation $\mathbb{E}[\mathcal{E}] = 0$ and variance $\mathbb{V}(\mathcal{E}) = \sigma^2$. We will assume that $\mathcal{E}$ and $\boldsymbol{X}$ are statistically independent. Suppose that the goal is to classify the binary response $Y = \mathbb{I}(Z \geq \eta)$, where $\eta \in \mathbb{R}$ is a fixed constant, called a threshold. In particular, we wish to use $\boldsymbol{X}$ to classify future observations of Y.

It is a well-documented phenomenon [44, 45] that if the distribution of interest is *imbalanced*, i.e., if $\mathbb{P}(Y = 0) >> \mathbb{P}(Y = 1)$, then the classification algorithm will bias towards the majority class. Further, in a case of extreme imbalance, a simple random sample will likely contain very few observations from the minority class. We aim to assuage this problem through the incorporation of expert opinions.

In particular we will assume that while we do not have access to Z, the pre-threshold response, we do have access to an expert opinion that can rank a small set of data according to

Z. E.g., perhaps we do not access to a patient's bone mineral density (BMD) measurements, and thus cannot classify a given patient as having osteoporosis outright. However, we may have access to an expert who, given the available data, could rank sets of 3 or 4 patients each and choose the one who is the "most likely" to have osteoporosis. Mathematically we will assume that this is equivalent to selecting the patient with the highest Z value.

By selecting our sample according to this technique, we will be able to reduce the imbalance in the data. For example, if our response variable Y takes the class $Y = 1$ with a probability of 0.13, we might expect a sample of size 50 to contain only only 6 or 7 observations where $Y = 1$. However, if we can nominate 50 sets of size 4 and select from each set the member who has the highest Z value, then we can expect the proportion of sample members with $Y = 1$ to be equal to $1 - (1 - 0.13)^5 \approx 0.5015$, thus leading to a balanced sample.

However, we are no longer taking a simple random sample of data. While a model build based upon this data will be well-posed to classify future observations taken according to this scheme (i.e., MaxNS test data), predictions will suffer for future observations taken randomly from the population – i.e., SRS test data. As the goal of the model is to predict the Y of a randomly selected population member, and not simply a MaxNS population member, this is problematic.

For ease of notation, we will let $Y_{[k]}$ and $\boldsymbol{X}_{[k]}$ denote our MaxNS sampling units. Further, we will simply write $\mathbb{P}\big(Y_{[k]} = 1 \,|\, \boldsymbol{x}\big)$ and $\mathbb{P}(Y = 1 \,|\, \boldsymbol{x})$ for the conditional class probabilities from the MaxNS and SRS schemes, respectively, and assume that it clear that we are conditioning on $\boldsymbol{X}_{[k]}$ and $\boldsymbol{X}$, respectively. Finally, we will sometimes use $\pi_{[k]}(\boldsymbol{x})$ and $\pi(\boldsymbol{x})$ to stand in for $\mathbb{P}\big(Y_{[k]} = 1 \,|\, \boldsymbol{x}\big)$ and $\mathbb{P}(Y = 1 \,|\, \boldsymbol{x})$ respectively, to emphasize that these probabilities are, in fact, functions of $\boldsymbol{x}$.

We wish to find a relationship between the MaxNS probabilities $\pi_{[k]}(\boldsymbol{x})$ and the SRS

probabilities $\pi(\boldsymbol{x})$. In particular, one might find a function g such that

$$\pi_{[k]}(\boldsymbol{x}) \approx g(\pi(\boldsymbol{x})),$$

and after finding $\widehat{\pi_{[k]}}(\boldsymbol{x})$, simply use $g^{-1}\big(\widehat{\pi_{[k]}}(\boldsymbol{x})\big)$ as an estimate of $\pi(\boldsymbol{x})$ given by

$$\hat{\pi}(\boldsymbol{x}) \approx g^{-1}\big(\widehat{\pi_{[k]}}(\boldsymbol{x})\big).$$

For example, in [42], the function g that is used is $g(t) = 1 - (1-t)^k$. I.e., the proposed link between $\pi_{[k]}$ and π is

$$\pi_{[k]}(\boldsymbol{x}) = 1 - (1 - \pi(\boldsymbol{x}))^k. \tag{2.1}$$

The rationale for (2.1) is as follows: if one ignores the conditioning variable, then one will quickly obtain that

$$\begin{aligned}
\mathbb{P}\big(Y_{[k]} = 1\big) &= 1 - \mathbb{P}\big(Y_{[k]} = 0\big) \\
&= 1 - \mathbb{P}(Y_1 = 0,\, Y_2 = 0, \ldots, Y_k = 0) \\
&= 1 - (\mathbb{P}(Y = 0))^k \\
&= 1 - (1 - \mathbb{P}(Y = 1))^k
\end{aligned}$$

Now, if one assumes that the relationship in the conditional case is similar, then it follows that

$$\mathbb{P}\big(Y_{[k]} = 1|\boldsymbol{x}\big) \approx 1 - (1 - \mathbb{P}(Y = 1|\boldsymbol{x}))^k.$$

while this approach led to positive results in [42], we will show in Section 2 through both theoretical and numerical results that this is a relationship that does not hold in general.

In light of this, we aim to discover the true nature of the relationship between $\pi_{[k]}(\boldsymbol{x})$ and $\pi(\boldsymbol{x})$, so as to obtain a more accurate procedure. This, however, is a very challenging problem. Hence, following some insightful theoretical derivations, we propose a new relationship between $\pi_{[k]}(\boldsymbol{x})$ and $\pi(\boldsymbol{x})$ and investigate its robustness.

2.2 Developing the MaxNS and SRS Link

Let $\boldsymbol{X} \sim F_{\boldsymbol{X}}$, and $\mathcal{E} \sim F_{\mathcal{E}}$ be independent continuous random variables. Set $Z = h(\boldsymbol{X}) + \mathcal{E}$, and define the variable Y as

$$Y = \begin{cases} 1, & \text{if } Z \geq m, \\ -1, & \text{if } Z < m, \end{cases}$$

where m is some real number. That is, we assume that $(\boldsymbol{X}, Y)$ follows a threshold model with parameter m. Now, assume that a MaxNS sample $\big(\boldsymbol{X}_{[k]i}, Y_{[k]i}\big)_{i=1}^{n}$ of sample size n and set size k is obtained, with Z as the ranking variable and $\boldsymbol{X}, Y, \mathcal{E}$ as concomitants.

Here we will explore the various marginal, joint, and conditional distributions of the variables $h(\boldsymbol{X})_{[k]}$, $\mathcal{E}_{[k]}$, $Y_{[k]}$, and $Z_{(k)}$. Our final goal is to obtain the conditional class probabilities

$$\pi_{[k]}(\boldsymbol{x}) \coloneqq \mathbb{P}\big(Y_{[k]} = 1 \,|\, \boldsymbol{x}\big)$$

associated with a MaxNS sample. In particular, we wish to explore the relationship between $\pi_{[k]}(\boldsymbol{x})$ and the conditional class probabilities $\pi(\boldsymbol{x}) \coloneqq \mathbb{P}(Y = 1 \,|\, \boldsymbol{x})$ associated with a simple random sample.

To obtain our results, we will make the simplifying assumption that $d = 1$ and that $h(t) = t$. I.e., we assume for now that $Z = X + \mathcal{E}$. Note however that all instances of X can simply be replaced by $h(\boldsymbol{X})$. As Z only depends on $\boldsymbol{X}$ through h, we have that $\mathbb{P}\big(Y_{[k]} = 1 \,|\, \boldsymbol{X}_{[k]} = \boldsymbol{x}\big) = \mathbb{P}\big(Y_{[k]} = 1 \,|\, h\big(\boldsymbol{x}_{[\boldsymbol{k}]}\big) = h(\boldsymbol{x})\big)$, which means that our results for the simplified model generalize immediately to the general case.

First, we will obtain the probability density function (PDF) of $X_{[k]}$, the feature variable under the MaxNS sampling procedure.

Result 1. *Let M_{k-1} denote the maximum of $k-1$ i.i.d. copies of Z. Then the distribution*

of $X_{[k]}$ is given by

$$\begin{aligned} f_{X_{[k]}}(x) &= k f_{X_i}(x) \mathbb{P}(M_{k-1} - \mathcal{E}_i \leq x) \\ &= k(k-1) f_{X_i}(x) \int_{-\infty}^{\infty} f_{Z_i}(t) [F_{Z_i}(t)]^{k-2} [1 - F_{\mathcal{E}_i}(t-x)] dt, \end{aligned}$$

where $\mathcal{E}_i$ and M_{k-1} are independent.

Proof. Let $W = \underset{j=1,\ldots,k}{\operatorname{argmax}} Z_j$. By continuity of Z, this value is unique w.p. 1. Note that W has a discrete uniform distribution on $\{1, 2, \ldots, k\}$. Applying Bayes' theorem,

$$f_{X_{[k]}}(x) \;=\; f_{X_i \,|\, W}(x \,|\, i) \;=\; \frac{P(W = i \,|\, X_i = x) f_{X_i}(x)}{P(W = i)} \;=\; k f_{X_i}(x) P(W = i \,|\, X_i = x).$$

Then

$$\begin{aligned} P(W = i \,|\, X_i = x) &= P\Big(\operatorname{argmin}(Z_1, \ldots, Z_{i-1}, \mathcal{E}_i + x, Z_{i+1}, \ldots, Z_k) = i \Big) \\ &= P\Big(\mathcal{E}_i + x \geq \max(Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_k) \Big) \\ &= P(M_{k-1} - \mathcal{E}_i \leq x). \end{aligned}$$

Note that by the independence of the sample, $\mathcal{E}_i$ and $\max(Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_k) = M_{k-1}$ are independent.

By the independent convolution formula for linear combinations,

$$\begin{aligned} P(M_{k-1} - \mathcal{E}_i \leq x) &= \int_{-\infty}^{x} f_{M_{k-1} - \mathcal{E}_i}(w) dw \\ &= \int_{-\infty}^{x} \int_{-\infty}^{\infty} f_{M_{k-1}}(t) f_{\mathcal{E}_i}(t-w) dt dw \\ &= (k-1) \int_{-\infty}^{x} \int_{-\infty}^{\infty} f_{Z_i}(t) \Big[F_{Z_i}(t) \Big]^{k-2} f_{\mathcal{E}_i}(t-w) dt dw \\ &= (k-1) \int_{-\infty}^{\infty} \int_{-\infty}^{x} f_{Z_i}(t) \Big[F_{Z_i}(t) \Big]^{k-2} f_{\mathcal{E}_i}(t-w) dw dt \end{aligned}$$

$$= (k-1)\int_{-\infty}^{\infty} f_{Z_i}(t)\Big[F_{Z_i}(t)\Big]^{k-2}\int_{-\infty}^{x} f_{\mathcal{E}_i}(t-w)dwdt$$

$$= (k-1)\int_{-\infty}^{\infty} f_{Z_i}(t)\Big[F_{Z_i}(t)\Big]^{k-2}\Big[1-F_{\mathcal{E}_i}(t-x)\Big]dt.$$

Combining this equation with the first equation of the proof brings us to the result. □

When an analytic solution to the given integral is not available, it may be easier to simply solve $P(M_{k-1} - \mathcal{E}_i \leq x)$ through numerical techniques.

Through a similar technique, we obtain the distribution of $\mathcal{E}_{[k]}$, the error term under a MaxNS sample.

Result 2. *Under the same conditions of Result 1, the distribution of $\mathcal{E}_{[k]}$ is given by*

$$f_{\mathcal{E}_{[k]}}(x) = kf_{\mathcal{E}_i}(x)\mathbb{P}(M_{k-1} - X_i \leq x)$$

$$= k(k-1)f_{\mathcal{E}_i}(x)\int_{-\infty}^{\infty} f_{Z_i}(t)[F_{Z_i}(t)]^{k-2}[1-F_{X_i}(t-x)]dt,$$

where X_i and M_{k-1} are independent.

The joint distribution of $X_{[k]}$ and $\mathcal{E}_{[k]}$ has a simple form, which will prove useful when obtaining conditional probabilities.

Result 3. *The joint distribution of $X_{[k]}$ and $\varepsilon_{[k]}$ is given by*

$$f_{X_{[k]},\mathcal{E}_{[k]}}(x,\varepsilon) = kf_{X_i}(x)f_{\mathcal{E}_i}(\varepsilon)[F_{Z_i}(x+\varepsilon)]^{k-1}.$$

Proof. Let $W = \underset{j=1,\ldots,k}{\operatorname{argmax}} Z_j$. By continuity of Z, this value is unique w.p. 1. Applying Bayes' theorem,

$$f_{X_{[k]},\mathcal{E}_{[k]}}(x,\varepsilon) = f_{X_i,\mathcal{E}_i\,|\,W}(x,\varepsilon\,|\,i)$$

$$= \frac{f_{X_i,\mathcal{E}_i}(x,\varepsilon)P(W=i\,|\,X_i=x,\mathcal{E}_i=\varepsilon)}{P(W=i)}$$

$$= k f_{X_i}(x) f_{\mathcal{E}_i}(\varepsilon) P(W = i \,|\, X_i = x, \mathcal{E}_i = \varepsilon).$$

Now,

$$\begin{aligned}
\mathbb{P}(W = i \,|\, X_i = x, \mathcal{E}_i = \varepsilon) &= \mathbb{P}(\text{argmax}(Z_1, \ldots, Z_{i-1}, x + \varepsilon, Z_{i+1}, \ldots, Z_k) = i) \\
&= \mathbb{P}(\max(Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_k) \leq x + \varepsilon) \\
&= \mathbb{P}(M_{k-1} \leq x + \varepsilon). \\
&= F_{M_{k-1}}(x + \varepsilon) \\
&= [F_{Z_i}(x + \varepsilon)]^{k-1}.
\end{aligned}$$

Thus,

$$f_{X_{[k]}, \mathcal{E}_{[k]}}(x, \varepsilon) = k f_{X_i}(x) f_{\mathcal{E}_i}(\varepsilon) [F_{Z_i}(x + \varepsilon)]^{k-1},$$

which results in the joint distribution of $X_{[k]}$ and $\mathcal{E}_{[k]}$. □

The conditional distribution of $\mathcal{E}_{[k]} \,|\, X_{[k]} = x$ may be obtained quickly.

Result 4. *The conditional distribution of* $\varepsilon_{[k]} \,|\, X_{[k]} = x$ *is given by*

$$\begin{aligned}
f_{\mathcal{E}_{[k]} \,|\, X_{[k]}}(\varepsilon \,|\, x) &= \frac{f_{\mathcal{E}_i}(\varepsilon)[F_{Z_i}(x + \varepsilon)]^{k-1}}{\mathbb{P}(M_{k-1} - \mathcal{E}_i \leq x)} \\
&= \frac{f_{\mathcal{E}_i}(\varepsilon)[F_{Z_i}(x + \varepsilon)]^{k-1}}{(k-1) \int_{-\infty}^{\infty} f_{Z_i}(t) \Big[F_{Z_i}(t)\Big]^{k-2} \Big[1 - F_{\mathcal{E}_i}(t - x)\Big] dt},
\end{aligned}$$

where M_{k-1} *and* $\mathcal{E}_i$ *are independent.*

Similarly, the conditional distribution of $Z_{(k)} \,|\, X = x$ follows.

Result 5. *The conditional distribution of* $Z_{(k)} \,|\, X_{[k]} = x$ *is given by*

$$f_{Z_{(k)} \,|\, X_{[k]}}(z \,|\, x) = \frac{f_{\mathcal{E}_i}(z - x)[F_{Z_i}(z)]^{k-1}}{\mathbb{P}(M_{k-1} - \mathcal{E}_i \leq x)}$$

$$= \frac{f_{\mathcal{E}_i}(z-x)[F_{Z_i}(z)]^{k-1}}{(k-1)\int_{-\infty}^{\infty} f_{Z_i}(t)\Big[F_{Z_i}(t)\Big]^{k-2}\Big[1-F_{\mathcal{E}_i}(t-x)\Big]dt},$$

where M_{k-1} and $\mathcal{E}_i$ are independent.

Proof. First,

$$f_{Z_{(k)}\,|\,X_{[k]}}(z\,|\,x) = f_{\mathcal{E}_{[k]}\,|\,X_{[k]}}(z-x\,|\,x).$$

Now, we simply apply the previous result. □

Finally, we obtain the conditional distribution of $Y_{[k]}\,|\,X_{[k]}=x$ from the above result.

Result 6. *The conditional distribution of $Y_{[k]}\,|\,X_{[k]}=x$ is given by*

$$\mathbb{P}\big(Y_{[k]}=1\,|\,X_{[k]}=x\big) = \mathbb{P}(\mathcal{E}>m-x\,|\,\mathcal{E}>M_{k-1}-x) \tag{2.2}$$

$$= \frac{\int_m^{\infty} f_{\mathcal{E}_i}(z-x)[F_{Z_i}(z)]^{k-1}dz}{(k-1)\int_{-\infty}^{\infty} f_{Z_i}(t)\Big[F_{Z_i}(t)\Big]^{k-2}\Big[1-F_{\mathcal{E}_i}(t-x)\Big]dt}, \tag{2.3}$$

where M_{k-1} and $\mathcal{E}_i$ are independent.

Proof. From Result 5,

$$\mathbb{P}\big(Y_{[k]}=1\,|\,X_{[k]}=x\big) = \mathbb{P}\big(Z_{(k)}\geq m\,|\,X_{[k]}=x\big)$$

$$= \frac{\int_m^{\infty} f_{\mathcal{E}_i}(z-x)[F_{Z_i}(z)]^{k-1}dz}{\mathbb{P}(M_{k-1}-\mathcal{E}_i\leq x)}.$$

Now, we observe that the numerator is equal to

$$\int_m^{\infty} f_{\mathcal{E}_i}(z-x)[F_{Z_i}(z)]^{k-1}dz = \int_{m-x}^{\infty} f_{\mathcal{E}_i}(z)[F_{Z_i}(z+x)]^{k-1}dz$$

$$= \mathbb{E}\big[F_{M_{k-1}}(\mathcal{E}+x)\mathbb{I}(\mathcal{E}>m-x)\big]$$

$$= \mathbb{E}\big[F_{M_{k-1}}(\mathcal{E}+x)\,|\,\mathcal{E}>m-x\big]\mathbb{P}(\mathcal{E}>m-x)$$

$$= \mathbb{P}(M_{k-1} \leq \mathcal{E} + x \,|\, \mathcal{E} > m - x)\mathbb{P}(\mathcal{E} > m - x)$$

$$= \mathbb{P}(M_{k-1} \leq \mathcal{E} + x,\, \mathcal{E} > m - x).$$

Hence,

$$\mathbb{P}\big(Y_{[k]} = 1 \,|\, X_{[k]} = x\big) = \frac{\int_m^\infty f_{\mathcal{E}_i}(z - x)[F_{Z_i}(z)]^{k-1} dz}{\mathbb{P}(M_{k-1} - \mathcal{E}_i \leq x)}$$

$$= \frac{\mathbb{P}(M_{k-1} \leq \mathcal{E} + x,\, \mathcal{E} > m - x)}{\mathbb{P}(M_{k-1} - \mathcal{E}_i \leq x)}$$

$$= \mathbb{P}\big(Z_{(k)} \geq m \,|\, X_{[k]} = x\big).$$

□

Note that Result 6 allows us to express the conditional class probability $\mathbb{P}\big(Y_{[k]} = 1 | X_{[k]} = x\big)$ in terms of the distributions of $\mathcal{E}$ and $Z = X + \mathcal{E}$. While this form is unfriendly, and for most distributions the resulting conditional class probability will not have a closed form, we may still employ numerical techniques to obtain the desired probabilities.

2.3 An Analytic Example

In this section we will use Result 6 to find the conditional class probabilities in a simplified case. In particular, we will assume that $X \sim U(0, 1)$, $\mathcal{E} \sim U(0, 1)$, $k = 2$, and $m = 2$.

First, note that $Z = X + \mathcal{E}$ has a triangular distribution. In particular, Z has the PDF

$$f_Z(z) = \begin{cases} z, & \text{if } 0 < z < 1, \\ 2 - z, & \text{if } 1 \leq z < 2, \end{cases}$$

and $f_Z(z) = 0$ otherwise, and a cumulative density function (CDF) of

$$F_Z(z) = \begin{cases} 0, & \text{if } z < 0, \\ \frac{1}{2}z^2, & \text{if } 0 \le z < 1, \\ -\frac{1}{2}z^2 + 2z - 1, & \text{if } 1 \le z < 2, \\ 1, & \text{if } z \ge 2, \end{cases}$$

We will determine the denominator $\mathcal{D}(x)$ in (2.3) next, as it is simpler to find than the numerator.

$$\begin{aligned} \mathcal{D}(x) &= (k-1)\int_{-\infty}^{\infty} f_Z(t)\Big[F_Z(t)\Big]\Big[1 - F_{\mathcal{E}}(t-x)\Big]dt \\ &= \int_{-\infty}^{\infty} f_Z(t)\Big[1 - F_{\mathcal{E}}(t-x)\Big]dt \\ &= \int_0^x t dt + \int_x^1 t(1-(t-x))dt + \int_1^{x+1}(2-t)(1-(t-x))dt \\ &= -\frac{2x^3 - 3x^2 - 3x - 1}{6} \end{aligned}$$

Next, we will find the numerator $\mathcal{N}(x)$ of (2.3. To this end, we will split this into the cases $0 \le m \le 1$ and $1 < m \le 2$, respectively. First, for the case $0 < m \le 1$,

$$\begin{aligned} \mathcal{N}(x) &= \int_m^{\infty} f_{\mathcal{E}}(z-x)F_Z(z)dz \\ &= \int_m^1 f_{\mathcal{E}}(z-x)\frac{1}{2}z^2 dz + \int_1^2 f_{\mathcal{E}}(z-x)\left(-\frac{1}{2}z^2 + 2z - 1\right)dz \\ &= \begin{cases} \displaystyle\int_x^1 \frac{1}{2}z^2 dz + \int_1^{x+1}\left(-\frac{1}{2}z^2 + 2z - 1\right)dz, & \text{if } 0 \le x < m \\ \displaystyle\int_m^1 \frac{1}{2}z^2 dz + \int_1^{x+1}\left(-\frac{1}{2}z^2 + 2z - 1\right)dz, & \text{if } m < x \le 1 \end{cases} \\ &= \begin{cases} -\frac{1}{6}(m^3 + x^3 - 3x^2 - 3x - 1), & \text{if } 0 \le x < m \\ -\frac{1}{6}(2x^3 - 3x^2 - 3x - 1), & \text{if } m < x \le 1. \end{cases} \end{aligned}$$

Alternatively, for the case $1 < m \leq 2$,

$$\begin{aligned}
\mathcal{N}(x) &= \int_m^\infty f_{\mathcal{E}}(z-x)F_Z(z)dz \\
&= \int_m^2 f_{\mathcal{E}}(z-x)\left(-\frac{1}{2}z^2+2z-1\right)dz \\
&= \begin{cases} 0, & \text{if } 0 \leq x < m-1, \\ \int_m^{x+1}\left(-\frac{1}{2}z^2+2z-1\right)dz, & \text{if } m-1 \leq x \leq 1 \end{cases} \\
&= \begin{cases} 0, & \text{if } 0 \leq x < m-1, \\ -\frac{1}{6}(x^3-3x^2-3x+1-m^3+6m^2-6m), & \text{if } m-1 \leq x \leq 1. \end{cases}
\end{aligned}$$

Thus, we obtain that the conditional class probability formula. In particular, for $0 \leq m \leq 1$, after some simplification we obtain that

$$\mathbb{P}\big(Y_{[k]}=1|x\big) = \begin{cases} 1-\dfrac{[x^3-m^3]_+}{2x^3-3x^2-3x-1}, & \text{if } 0 \leq m \leq 1, \\ \dfrac{x^3-3x^2-3x+1-m^3+6m^2-6m}{2x^3-3x^2-3x-1}, & \text{if } 1 < m < x+1 \leq 2, \\ 0 & \text{otherwise,} \end{cases} \tag{2.4}$$

where $[t]_+ = \max(t,0)$, the *positive part* of t.

We may easily obtain that the SRS class probabilities are given by

$$\begin{aligned}
\mathbb{P}(Y=1|X=x) &= \mathbb{P}(Z \geq m|X=x) \\
&= \mathbb{P}(X+\mathcal{E} \geq m|X=x) \\
&= \mathbb{P}(\mathcal{E}+x \geq m) \\
&= \mathbb{P}(\mathcal{E} \geq m-x) \\
&= \begin{cases} 0, & \text{if } x < m-1, \\ 1-(m-x), & \text{if } m-1 < x < m, \\ 1, & \text{if } x > m. \end{cases}
\end{aligned}$$

Using the link (2.1), one would obtain the approximation

$$\mathbb{P}(Y=1|X=x) \approx \begin{cases} 0, & \text{if } x < m-1, \\ 1-(m-x)^2, & \text{if } m-1 < x < m, \\ 1, & \text{if } x > m. \end{cases} \tag{2.5}$$

We compare the graphs of (2.4) and (2.5) for $m = 0.7, 1.0, 1.3$ in Figure 2.1. The true form is shown as a solid line, while the approximation based off of (2.1) is a dotted line.

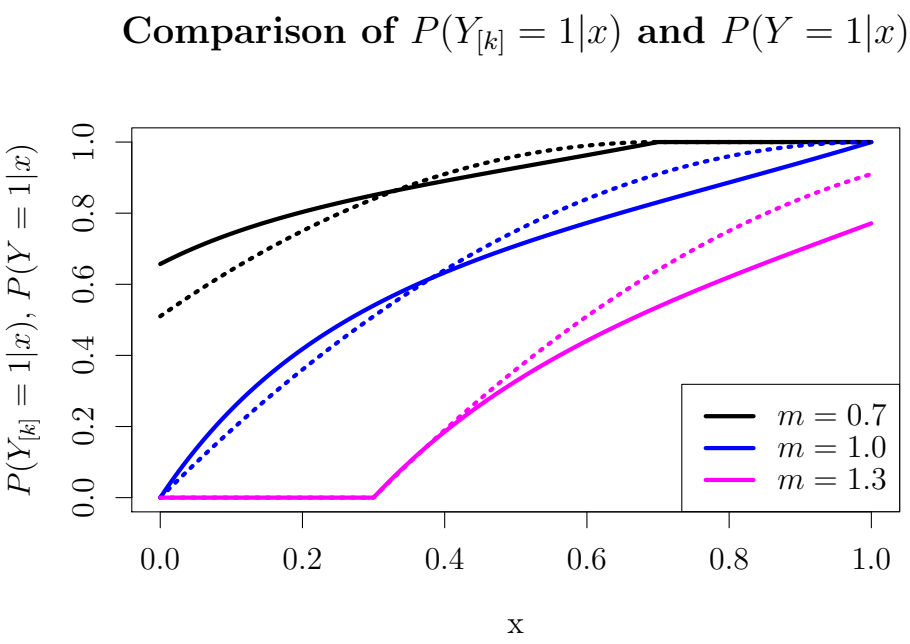

Figure 2.1: Compared here is the true form of the class conditional probability function for the problem established in Section 2.3 versus the approximation based on link (2.1). In particular, the function (2.4), representing the true form is compared to the function (2.5) representing the approximation based off of (2.1).

We can see that the two functions, in general, disagree. Thus, we obtain that the link (2.1) is not exact. In light of this, we will seek to discover a more accurate link. We will do so in the following section.

2.4 MaxNS Probabilities under Normality

In the previous section we worked out an explicit form for the class conditional probabilities within a MaxNS sample when X and $\mathcal{E}$ are both standard Uniform variables. However, this case is not applicable to real data scenarios. In this section we will assume instead that X and $\mathcal{E}$ are both Normally distributed. This is a much more realistic scenario: for example, the case of a linear model with a Normally-distributed predictor fits into this model.

In particular, let $X \sim N(\mu_X, \sigma_X^2)$, $\mathcal{E} \sim N(0, \sigma^2)$, $k = 2$, and let $m \in \mathbb{R}$ be the value such that $\mathbb{P}\big(Y_{[k]} = 1\big) = \frac{1}{2}$. For the remainder of this section, we will assume for simplicity that $\mu_X = 0$ and $\sigma_X^2 = 1$, though these same results will hold for any value of μ_X and σ_X, after an appropriate rescaling (see Result 8). Note that

$$\begin{aligned} 0.5 = \mathbb{P}\big(Y_{[k]} = 1\big) = \mathbb{P}\big(Z_{(k)} \geq m\big) &= 1 - \{1 - \mathbb{P}(Y = 1)\}^k \\ \implies \mathbb{P}(Y = 1) &= 1 - (0.5)^{1/k} \\ \implies \mathbb{P}(Z \leq m) &= (0.5)^{1/k} \end{aligned}$$

which, in turn, implies that $m = \sqrt{1 + \sigma^2}\Phi^{-1}\Big((0.5)^{1/k}\Big)$.

First, we obtain that

$$f(x) = \mathbb{P}(\mathcal{E} > m - x) = 1 - \Phi\left(\frac{m - x}{\sigma}\right).$$

We can obtain $\pi_{[k]}$ numerically, using the `integrate` function in the statistical software `R`. In Figure 2.2 we provide graphs of $\pi_{[k]}(x)$ for $\sigma = 0.33, 0.48$, and 0.88. These values correspond to a $\rho = \mathbb{C}\text{orr}(X, Z)$ values of 0.50, 0.75, and 0.90, respectively.

We wish to investigate if we can find some function g such that

$$\pi_{[2]}(x) = g(\pi(x)) \quad \forall x \in S_X.$$

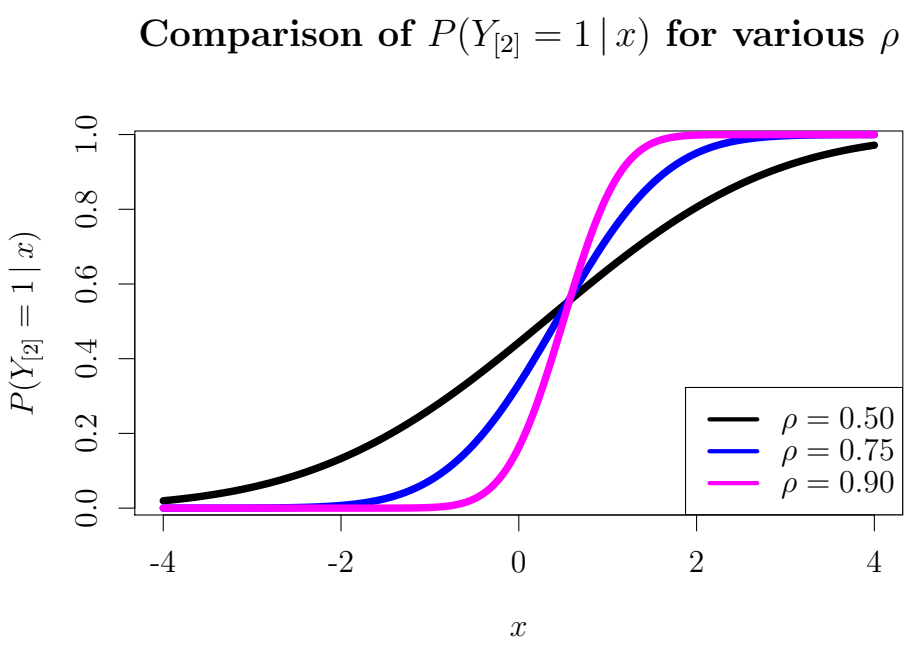

Figure 2.2: Comparison of $\pi(x) = P\big(Y_{[2]} = 1 \mid x\big)$ *for several values of* $\rho = \mathbb{C}orr(X, Z)$.

To this end, we produce Figure 2.3. In this figure, the horizontal axis represents $f(x)$, while the vertical axis represents $\pi_{[2]}(x)$.

We can see that the function g seems to have the approximate form $g(s) = s^{\omega_0}$, where $0 \leq \omega_0 \leq 1$. I.e., we find that

$$\pi_{[k]}(x) \approx \pi(x)^{\omega_0}, \quad \text{for some } 0 \leq \omega_0 \leq 1.$$

We run a simple loop to find the value ω_0 that minimizes

$$l(\omega) \coloneqq \int_{-\infty}^{\infty} \big[\pi(x)^{\omega} - \pi_{[k]}(x)\big]^2 dx.$$

In Table 2.1 we give the resulting values of ω_0.

To show how well this approximates the true function, in Figure 2.3 we again plot $\pi_{[2]}(x)$ for the three values of ρ, as solid lines, with the approximation $\pi(x)^{\omega_0}$ overlaid as dotted

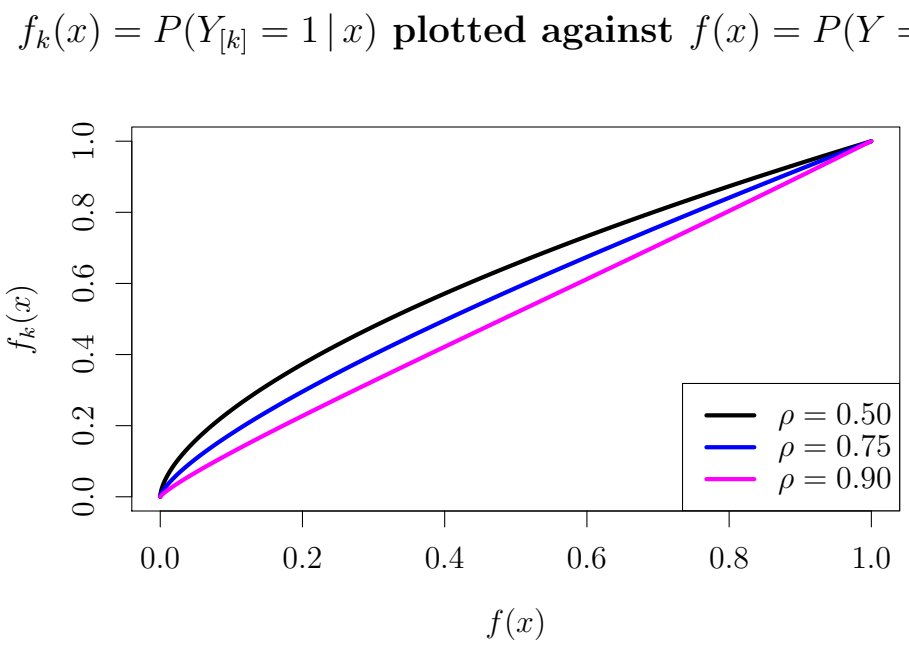

Figure 2.3: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \mid x\big)$ *plotted against* $f(x) = P(Y = 1 \mid x)$*, for various values of* $\rho = \mathbb{C}\text{orr}(X, Z)$*. A visual inspection reveals that there appears to be the approximate relationship* $\pi_{[k]}(x) = \pi(x)^{\omega_0}$ *for* $0 < \omega_0 < 1$*.*

	ω_0					
ρ	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 9$	$k = 16$
0.50	0.612	0.489	0.426	0.386	0.309	0.258
0.75	0.693	0.584	0.524	0.485	0.404	0.347
0.90	0.791	0.708	0.660	0.627	0.555	0.500
0.95	0.848	0.784	0.746	0.720	0.661	0.614
0.99	0.929	0.897	0.878	0.864	0.833	0.807

Table 2.1: This table gives the optimal value ω_0 *for several combinations of* ρ *and* k*.*

lines.

In summary, we can see that $\pi_{[2]}(x) \approx \pi(x)^{\omega_0}$ is a very good approximation, which depends only on σ, the standard deviation of $\mathcal{E}$.

In Figure 2.5 we repeat this analysis for $k = 3, 4, 5, 9$, and 16; we see that the $\pi_{[k]}(x) \approx \pi(x)^{\omega_0}$ approximation continues to hold.

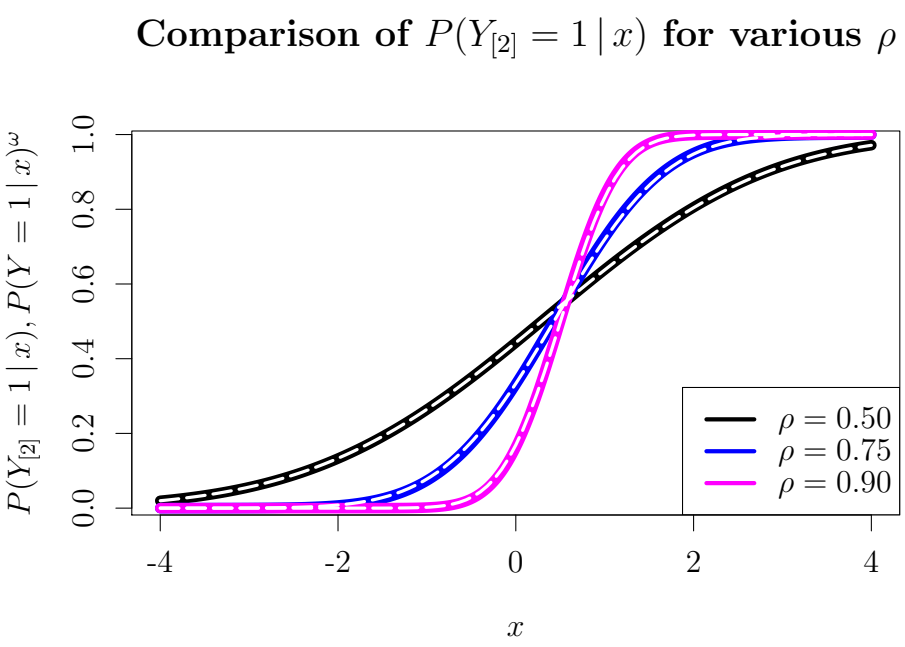

Figure 2.4: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \mid x\big)$ *plotted against* $\pi(x)^{\omega_0} = P(Y = 1 \mid x)^{\omega_0}$*, for various values of* $\rho = \mathbb{C}orr(X, Z)$*. The functions* $\pi_{[k]}(x)$ *are given as solid coloured lines, while* $\pi(x)^{\omega_0}$ *is a dotted white line.*

We summarize this finding below.

Result 7 (Conjecture)**.** *Let* $X \sim N(0, 1)$*,* $\mathcal{E} \sim N(0, \sigma^2)$*,* $k \geq 2$*, and* $m = 1 - \Phi\big(\frac{m-x}{\sigma}\big)$*. Then there exists some* $0 < \omega_0 < 1$ *such that, for any* $x \in \mathbb{R}$*,*

$$\mathbb{P}\big(Y_{[k]} = 1 \mid X_{[k]} = x\big) \approx \{\mathbb{P}(Y = 1 \mid X = x)\}^{\omega_0}.$$

We were unable to find an analytical justification for this approximation. We will rely on the numerical results given earlier in this section as our justification moving forward.

However, it is easy to show that Result 7 extends to any Normally distributed X variable, not only the Standard Normal.

Result 8. *Let* X*,* $\mathcal{E}$*,* k *and* m *be as in Result 7, and let* ω_0 *be the constant that optimizes this approximation. Now, set* $X' = \mu + \sigma_X X$*,* $\mathcal{E}' = \sigma_X \mathcal{E}$*,* $Z' = X' + \mathcal{E}'$*,* $m' = \mu + \sigma_X m$*, and*

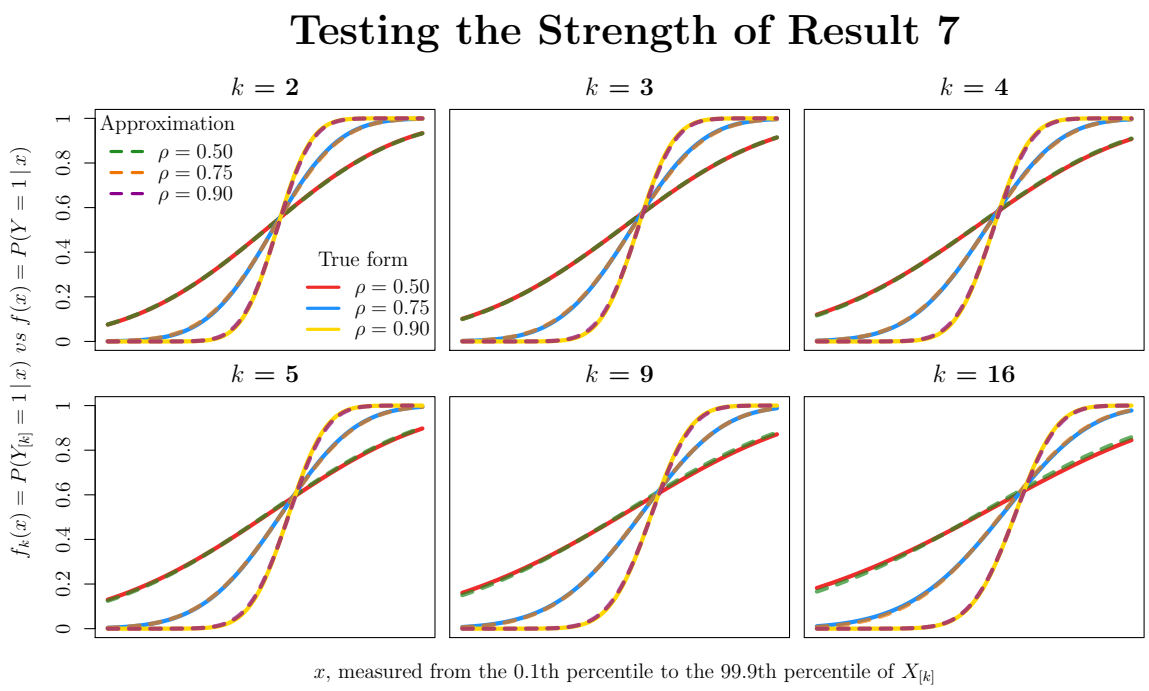

Figure 2.5: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \,|\, x\big)$ *compared to the function* $\pi(x)^{\omega_0} = P(Y = 1 \,|\, x)^{\omega_0}$*, for various values of* $\rho = 0.50, 0.75, 0.90$*, and* $k = 2, 3, 4, 5, 9, 16$*. The functions* $\pi_{[k]}(x)$ *are given as solid coloured lines, while* $\pi(x)^{\omega_0}$ *are given as dotted lines.*

$Y' = \mathbb{I}(Z' \geq m')$*. Then, for any* $x \in \mathbb{R}$*,*

$$\mathbb{P}\big(Y'_{[k]} = 1 \,|\, X'_{[k]} = x'\big) \approx \{\mathbb{P}(Y = 1 \,|\, X = x)\}^{\omega_0}.$$

Proof. First, note that $Z' = \mu + \sigma_X Z$, and $M'_{k-1} = \mu + \sigma_X M_{k-1}$. Now, from Result 5,

$$\begin{aligned} f_{Z'_{[k]} \,|\, X'_{[k]}}(z' \,|\, x') &= \frac{f_{\mathcal{E}'}(z' - x')[F_{Z'}(z')]^{k-1}}{\mathbb{P}\big(M'_{k-1} - \mathcal{E}' \leq x'\big)} \\ &= \frac{1}{\sigma_X} \frac{f_{\mathcal{E}}(z - x)[F_Z(z)]^{k-1}}{\mathbb{P}(M_{k-1} - \mathcal{E} \leq x)} \\ &= \frac{1}{\sigma_X} f_{Z_{[k]} \,|\, X_{[k]}}(z \,|\, x) \end{aligned}$$

Thus, we may integrate to obtain that

$$\mathbb{P}\big(Y'_{[k]} = 1 \,|\, X'_{[k]} = x'\big) = \mathbb{P}\big(Z'_{(k)} \geq m' \,|\, X'_{[k]} = x'\big)$$

$$
\begin{aligned}
&= \int_{m'}^{\infty} f_{Z'_{[k]} \,|\, X'_{[k]}}(t \,|\, x')dt \\
&= \int_{\mu+\sigma_X m}^{\infty} f_{Z'_{[k]} \,|\, X'_{[k]}}(t \,|\, x')dt \\
&= \int_{m}^{\infty} f_{Z'_{[k]} \,|\, X'_{[k]}}(\mu + \sigma_X z \,|\, x')dz \qquad \text{substitute } z = \frac{t-\mu}{\sigma_X} \\
&= \int_{m}^{\infty} f_{Z[k] \,|\, X[k]}(z \,|\, x)dz \\
&= \mathbb{P}\big(Y_{[k]} = 1 \,|\, X_{[k]} = x\big)
\end{aligned}
$$

By Result 7, there exists some $\omega_0 \in (0, 1)$ such that

$$\mathbb{P}\big(Y_{[k]} = 1 \,|\, X_{[k]} = x\big) \approx \{\mathbb{P}(Y = 1 \,|\, X = x)\}^{\omega_0} \quad \text{for all } x \in \mathbb{R}.$$

Since $\mathbb{P}\Big(Y'_{[k]} = 1 \,|\, X'_{[k]} = x'\Big) = \mathbb{P}\big(Y_{[k]} = 1 \,|\, X_{[k]} = x\big)$, the proof is complete. □

Note that, though we make the assumption that $X \in \mathbb{R}^1$, i.e., that we have access to only a single explanatory variable, this model expands easily to a multivariate case. In particular, assume that we have access to the vector of predictors $\boldsymbol{X} = (X_1, X_2, \ldots, X_d) \in \mathbb{R}^d$. Then, the results of this section require only that there exists a transformation $h : \mathbb{R}^d \to \mathbb{R}$ such that $h(\boldsymbol{X})$ is Normally distributed and $Z = h(\boldsymbol{X}) + \mathcal{E}$. Then we may simply switch all instances of X in this section with $h(\boldsymbol{X})$ and the results copy over with no further modifications necessary.

Thus, it is important to remember that the form of $\boldsymbol{X}$ is not directly relevant in Result 7; we need only for there to exist a transformation h such that $Z = h(\boldsymbol{X}) + \mathcal{E}$, where $h(\boldsymbol{X})$ and $\mathcal{E}$ are Normal. The variable $\mathcal{E}$ is simply the difference between $h(\boldsymbol{X})$ and Z, and ρ is a measurement of how "close" $h(\boldsymbol{X})$ and Z are.

As one example, a linear model has the form $Z = \boldsymbol{X}^\top \boldsymbol{\beta} + \mathcal{E}$, with the frequent additional assumption that $\mathcal{E}$ is Normally distributed. If we assume further that $h(\boldsymbol{X}) = \boldsymbol{X}^\top \boldsymbol{\beta}$ is

itself Normally distributed, then a threshold model in this circumstance will satisfy the requirements of Result 7.

In the following section, we will test how robust Result 7 is against the Normality assumption, both in the distribution of $h(\boldsymbol{X})$ and $\mathcal{E}$.

2.5 Robustness against Model Assumptions

Results 1 – 7 in the previous section have made the assumption that, in the model $Z = X + \mathcal{E}$, the variable X (or $h(\boldsymbol{X})$) has a Normal distribution. While this is often the case in real data scenarios, it is also often the case that the variable X will have a skewed distribution. In this section, we will test the strength of the approximation in Result 7 when X is non-Normal.

In particular, we will suppose instead that X follows a Weibull distribution. The Weibull distribution is a continuous probability distribution characterized by two positive parameters: the scale parameter, and the shape parameter. The Weibull distribution is a popular choice used when modelling a variable as it may take a wide array of (unimodal) shapes. We will fix the scale constant to 1 as this model is largely scale-invariant (given ρ) (Result 8). With a shape parameter of approximately 3.6, the distribution has a skewness of 0; shape parameters under 3.6 lead to a right-skewed distribution, while shape parameters above 3.6 lead to a left-skewed distribution.

We will consider the shape parameters of 2, 4, and 10. We display the resultant distributions of Z for the parameters $\rho = 0.5, 0.75, 0.90$ in Figure 2.6

From Figure 2.6, we can see that the approximation weakens as X moves further from Normality, as well as weakening as k grows. However, even in the worst-case shown, the error does not become appreciable. implying that this model displays some robustness in the Normality assumption on X.

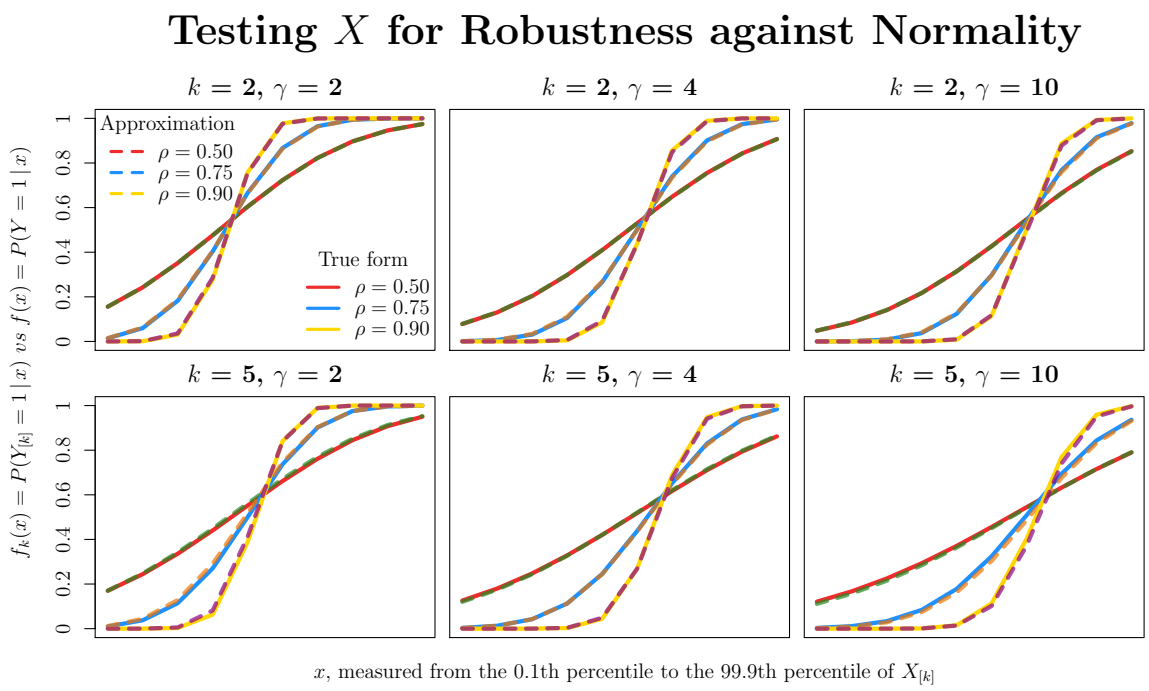

Figure 2.6: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \,|\, x\big)$ *compared to the function* $\pi(x)^{\omega_0} = P(Y = 1 \,|\, x)^{\omega_0}$*, for various values of* $\rho = 0.50, 0.75, 0.90$*, and* $k = 2, 5$*, in the case* $Z = X + \mathcal{E}$*, where* $X \sim$ *Weibull*(γ, θ)*. The functions* $\pi_{[k]}(x)$ *are given as solid lines (primary colours), while* $\pi(x)^{\omega_0}$ *are given as dotted lines (complementary secondary colours).*

We will also test here the robustness of the assumption of Normality in $\mathcal{E}$. In particular, we will now investigate the setting $Z = X + \mathcal{E}$, where $X \sim \mathrm{N}(0, 1)$ and $\mathcal{E}$ is drawn from the *Generalized Error Distribution*.

The Generalized Error Distribution is a generalization of the Normal Distribution, parametrized by three parameters: a location parameter μ, a positive scale parameter α, and a positive shape parameter β. This distribution has a PDF of

$$f(x;\, \alpha, \beta) = \frac{\beta}{2\alpha\Gamma(1/\beta)} \exp\left\{ -\left(\frac{|x - \mu|}{\alpha} \right)^{\beta} \right\}, \quad x \in \mathbb{R}.$$

This distribution contains the Normal distribution as a special case at $\beta = 2$ (which results in a variance of $\alpha^2/2$), and the Laplace distribution at $\beta = 1$. It converges to a uniform distribution on $(\mu - \alpha, \mu + \alpha)$ as $\beta \to \infty$.

We will test here the case $\mathcal{E} \sim \text{GED}(0, \alpha, \beta)$, with $\beta = 1, 2, 5$, with α set to attain the desired values of $\rho = \mathbb{C}\text{orr}(X, Z)$.

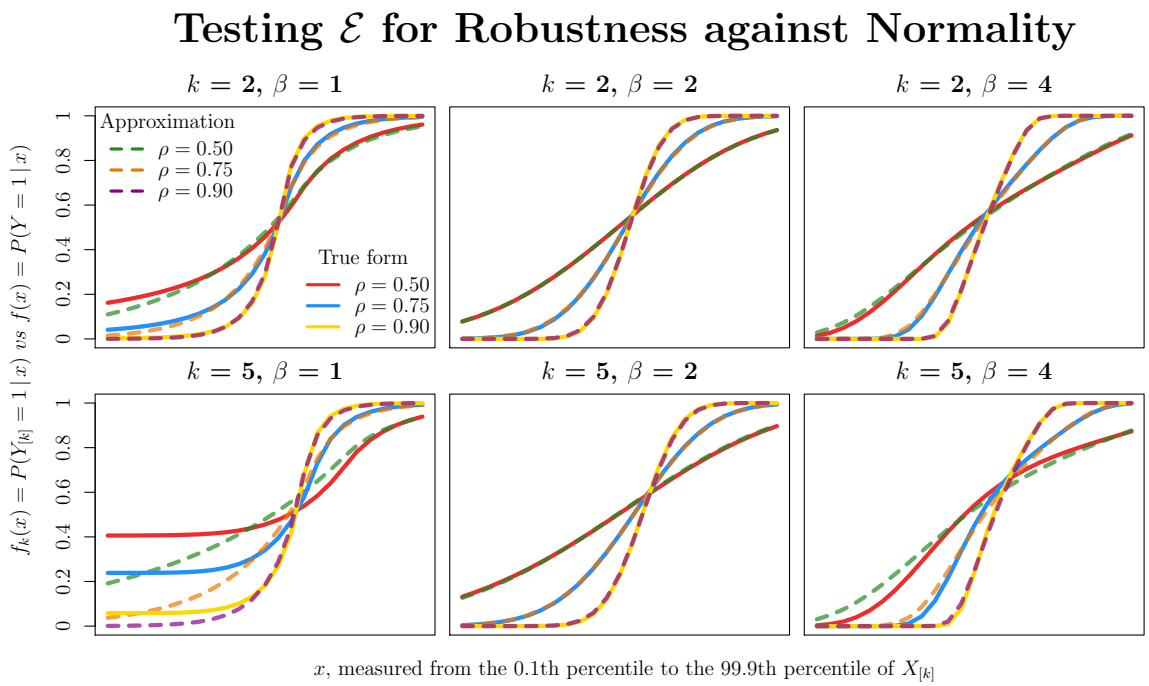

Figure 2.7: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \,|\, x\big)$ compared to the function $\pi(x)^{\omega_0} = P(Y = 1 \,|\, x)^{\omega_0}$, for various values of $\rho = 0.50, 0.75, 0.90$, and $k = 2, 5$, in the case $Z = X + \mathcal{E}$, where $\mathcal{E} \sim GED(0, \alpha, \beta)$. The functions $\pi_{[k]}(x)$ are given as solid lines (primary colours), while $\pi(x)^{\omega_0}$ are given as dotted lines (complementary secondary colours).

We can see that the violation of Normality in $\mathcal{E}$ leads to greater error in the approximation than violations of Normality in X. Again, the error increases with k. We notice in particular that leptokurtic distributions in $\mathcal{E}$ lead to a greater error than platykurtic $\mathcal{E}$ distributions. Further, this error is greater for small values of x than for large values of x. Further, we can see that as the correlation ρ between X and Z shrinks, the error in the approximation grows.

Finally, note that in the model $Z = X + \mathcal{E}$, while it may be helpful to think of $\mathcal{E}$ as an error term, this is not stipulated in the model in any way. Thus, we will test a situation where X and $\mathcal{E}$ are both non-Normal. In particular, we will let X and $\mathcal{E}$ each come from a Gamma distribution, with shape parameter α and scale parameter θ.

As the relationship between $\pi_{[k]}(x)$ and $\pi(x)$ is largely invariant of scale, we will keep the scale parameter fixed at $\theta = 1$. We will try shape parameters $\alpha = 2$, 4, and 9.

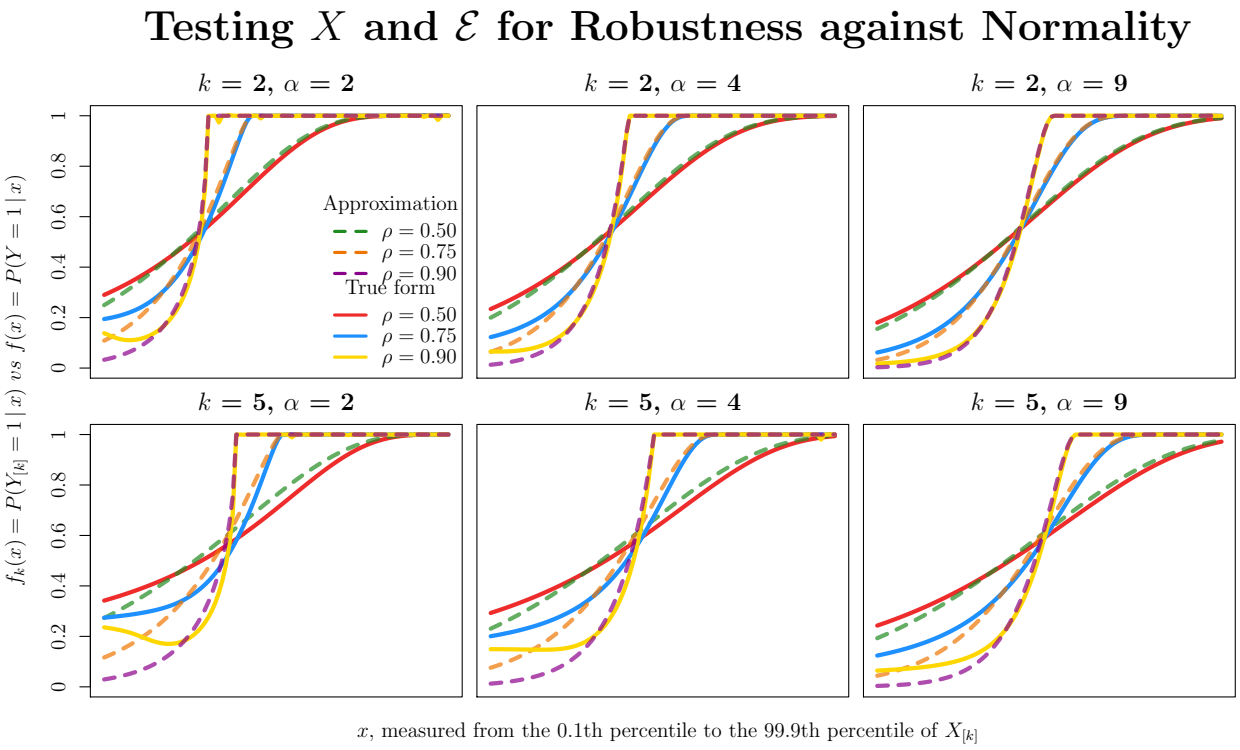

Figure 2.8: Shown here is the function $\pi_{[k]}(x) = P\big(Y_{[k]} = 1 \mid x\big)$ *compared to the function* $\pi(x)^{\omega_0} = P(Y = 1 \mid x)^{\omega_0}$*, for various values of* $\rho = 0.50, 0.75, 0.90$*, and* $k = 2, 5$*, in the case* $Z = X + \mathcal{E}$*, where* $X \sim Gamma(\alpha, \beta), \mathcal{E} \sim (\alpha_{\mathcal{E}}, \beta_{\mathcal{E}})$*. The functions* $\pi_{[k]}(x)$ *are given as solid lines (primary colours), while* $\pi(x)^{\omega_0}$ *are given as dotted lines (complementary secondary colours).*

We can see here again that the approximation grows weaker with k, while growing stronger with ρ. Once more, the accuracy is high for high values of x, and low for small values of x.

Overall, this battery of tests demonstrates that this model does display some sensitivity to violations of the Normality assumptions. In particular, while being moderately robust to violations in the Normality of X, the model seems sensitive to violations in the normality of $\mathcal{E}$. This sensitivity, in general, grows with k and shrinks with ρ, and mostly impacts small values of x.

Chapter 3

Logistic Regression with MaxNS Data

In this chapter, we develop a modified technique for constructing a Logistic regression model built upon a MaxNS training set, which we call the MNSLR problem. We then present a numerical technique for solving this MNSLR problem, as well as provide a proof that the algorithm given will in fact converge to a result. We also briefly discuss regularization and demonstrate how it applies to the MNSLR problem.

3.1 Introduction

Suppose that we have been presented with a vector of feature variables $\boldsymbol{X} \in \boldsymbol{\mathcal{X}} \subseteq \mathbb{R}^d$ and a binary response variable $Y \in \mathcal{Y} = \{0, 1\}$. For example, within a sample of mammograms, perhaps X is the radius of a cell nucleus, while Y marks whether this tumour is benign ($Y = 0$) or malignant ($Y = 1$). Suppose further that our goal is to model Y based on $\boldsymbol{X}$, using a (labelled) training set $(\boldsymbol{x}, y)_{i=1}^n \subseteq \boldsymbol{\mathcal{X}} \times \mathcal{Y}$.

One approach is to construct a simple linear regression model. That is, fit a curve of the form $\hat{Y} = \boldsymbol{w}^\top \boldsymbol{X} + b$, to the data, according to some optimization metric (e.g., minimizing the sum of the squared residuals, $\sum_{i=1}^n \big(\boldsymbol{w}^\top \boldsymbol{X} + b - y_i\big)^2$). Then, we could classify $Y = 1$

for all $\boldsymbol{X} = \boldsymbol{x}$ such that $\hat{y} > 0.5$, and classify $Y = 0$ otherwise. This technique is known as the *Linear probability model* [46]. We visualize this technique in Figure 3.1, using a random sample of $n = 50$ observations from the Wisconsin Breast Cancer Dataset, available on the UCI Machine Learning Repository [47].

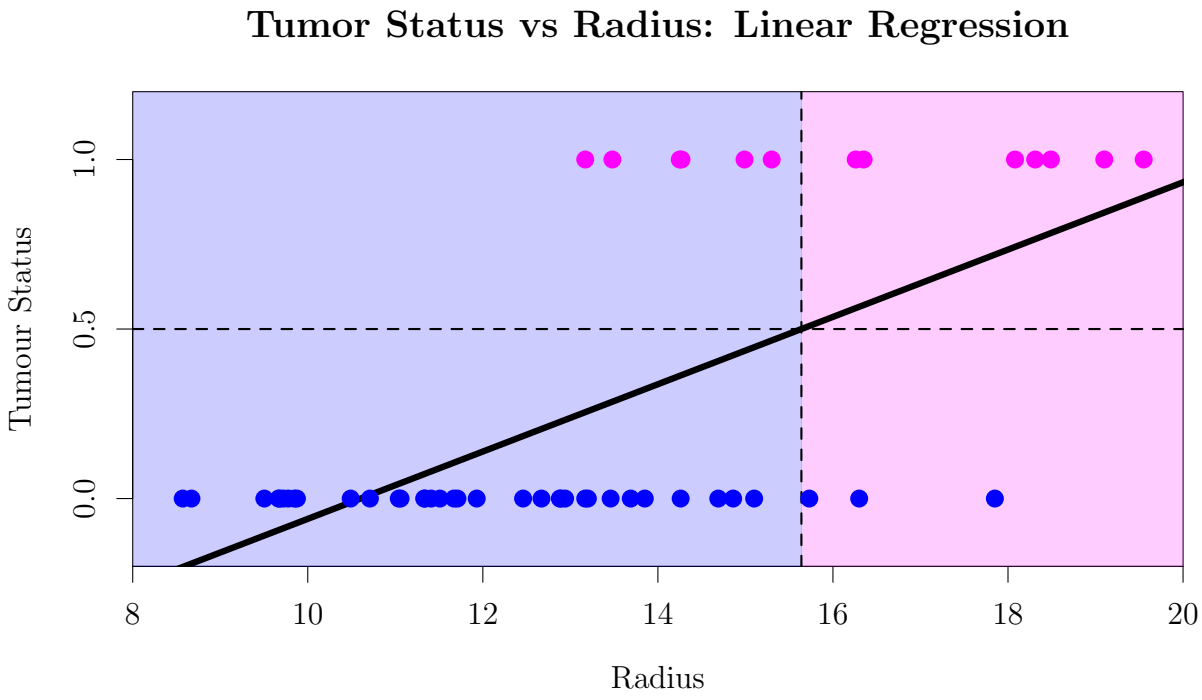

Figure 3.1: Shown here is the result of the linear regression classification technique on a random sample of $n = 50$ observations from the Wisconsin Breast Cancer Dataset. We restrict our feature space to a single variable X: the radius of the cell cluster. For all observations x_i such that $\hat{y}_i > 0.5$, we classify $Y = 1$, and we classify $Y = 0$ for all other observations.

However, this can lead $\hat{Y}$, our "model" of Y, to take values below 0 and above 1. This model may be seen as it presents a description of Y that is difficult to interpret. Further, if we recall that the goal of regression analysis is to fit $\mathbb{E}[Y \mid \boldsymbol{X} = \boldsymbol{x}]$, the mean of Y at $\boldsymbol{X} = \boldsymbol{x}$, and note that as $\mathcal{Y} = \{0, 1\}$, this is equivalent to fitting $\mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$, it seems that we should seek to force our $\hat{Y}$ to remain in the range $[0, 1]$.

We can achieve this by wrapping the linear function $\boldsymbol{w}^\top \boldsymbol{x} + b$ in the *sigmoid* function $\sigma(t)$,

defined as

$$\sigma(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t}, \quad t \in \mathbb{R}.$$

We choose this function as it is a monotone-increasing mapping from $\mathbb{R}$ to $[0,1]$. Thus, we arrive at a new curve to be fit; the *logistic curve* f, defined as

$$f(\boldsymbol{x}) = \frac{1}{1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}+b\right)}} = \frac{e^{\boldsymbol{w}^\top \boldsymbol{x}+b}}{1+e^{\boldsymbol{w}^\top \boldsymbol{x}+b}}.$$

That is, we make the model assumption that

$$\mathbb{P}(Y=1 \,|\, \boldsymbol{X}=\boldsymbol{x}) \approx \frac{1}{1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}+b\right)}}, \tag{3.1}$$

and we wish to find the constants $\boldsymbol{w}$ and b that optimize this approximation.

The technique used to fit the model parameters $\boldsymbol{w}, b$ can be viewed through the lens of *Maximum Likelihood Estimation* (MLE). In the MLE procedure, introduced by Fisher between 1912 and 1922 [48], one considers the likelihood of observing the sample data at hand under all possible parameter values of the joint distribution of the sample data, and picks the parameter values that gives the maximum possible likelihood (supposing that such a value exists and is unique). In particular, if we have a sample $(y_i)_{i=1}^n$ from a density $p(y;\boldsymbol{\theta})$ with a vector of parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, then we wish to find the vector $\boldsymbol{\theta}$ such that the *likelihood function* L, defined as

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} p(y_i;\boldsymbol{\theta}), \quad \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{3.2}$$

is maximized as a function of $\boldsymbol{\theta}$. However, working with (3.2) is often challenging. To simplify the problem, we will often instead consider the equivalent problem of minimizing the negative *log-likelihood* function,

$$\ell(\boldsymbol{\theta};\, \boldsymbol{y}) = -\ln\left(\prod_{i=1}^{n} p(y_i;\boldsymbol{\theta})\right) = -\sum_{i=1}^{n} \ln(p(y_i;\boldsymbol{\theta})). \tag{3.3}$$

The problem (3.3) is simpler as it is generally easier to work with general sums than products. Further, we will prefer phrasing our problem in terms of minimization as this is the standard optimization framework. Note that we will refer to the summands $-\ln(p(y_i;\boldsymbol{\theta}))$ by $\ell_i(\boldsymbol{\theta};\, y)$.

To the end of applying the MLE procedure to fitting $\boldsymbol{w}$ and b to find the optimal logistic curve for the observed data, define the density function for Y as below:

$$p_{Y|\boldsymbol{x}}(y;\boldsymbol{w},b) = \mathbb{P}(Y = y \,|\, \boldsymbol{X} = \boldsymbol{x};\boldsymbol{w},b).$$

According to the MLE procedure, our task now is to minimize

$$\ell(\boldsymbol{w},b;\,\boldsymbol{x},\boldsymbol{y}) = -\sum_{i=1}^{n} \ln\big(p_{Y_i|\boldsymbol{x_i}}(y_i;\boldsymbol{w},b)\big). \tag{3.4}$$

First, recall that y_i can only take the values 0 and 1. Hence, (3.4) can be represented as

$$-\sum_{i=1}^{n} \big[y_i \ln\big(p_{Y_i|\boldsymbol{x_i}}(1;\boldsymbol{w},b)\big) + (1-y_i)\ln\big(1-p_{Y_i|\boldsymbol{x_i}}(1;\boldsymbol{w},b)\big)\big]. \tag{3.5}$$

Now, applying model assumption (3.1), we have that (3.5) takes the form

$$-\sum_{i=1}^{n}\left[y_i \ln\left(\frac{1}{1+\exp(-(\boldsymbol{w}^\top \boldsymbol{x}_i + b))}\right) + (1-y_i)\ln\left(1-\frac{1}{1+\exp(-(\boldsymbol{w}^\top \boldsymbol{x}_i + b))}\right)\right]. \tag{3.6}$$

Finally, after some algebra, the log-likelihood function simplifies to

$$\ell(\boldsymbol{w},b;\,\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{n}\Big[y_i\big(\boldsymbol{w}^\top \boldsymbol{x}_i + b\big) - \ln\Big(1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big)\Big].$$

Minimizing this function over $\boldsymbol{w}$ and b is the Logistic regression problem, which we present below:

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \sum_{i=1}^{n}\Big[\ln\Big(1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big) - y_i\big(\boldsymbol{w}^\top \boldsymbol{x}_i + b\big)\Big]. \tag{3.7}$$

This problem may be solved through numerical techniques, such as the Newton-Raphson algorithm [49] – also known as the Iteratively Reweighted Least Square (IRLS) approach – or gradient descent [50]. A full exploration of such techniques falls outside of the scope of this research, but a thorough comparison may be found within [51].

The statistical software package R has several libraries which contain functions for performing Logistic regression. One of which is the `glm` function within the `stats` package. In Figure 3.2 we repeat the earlier analysis on the Wisconsin Breast Cancer Dataset, except now we will use Logistic regression instead of linear regression. Again, we will classify $Y = 1$ if $\hat{Y} > 0.5$ and $Y = 0$ otherwise, but the difference now being that we have an interpretable model for Y.

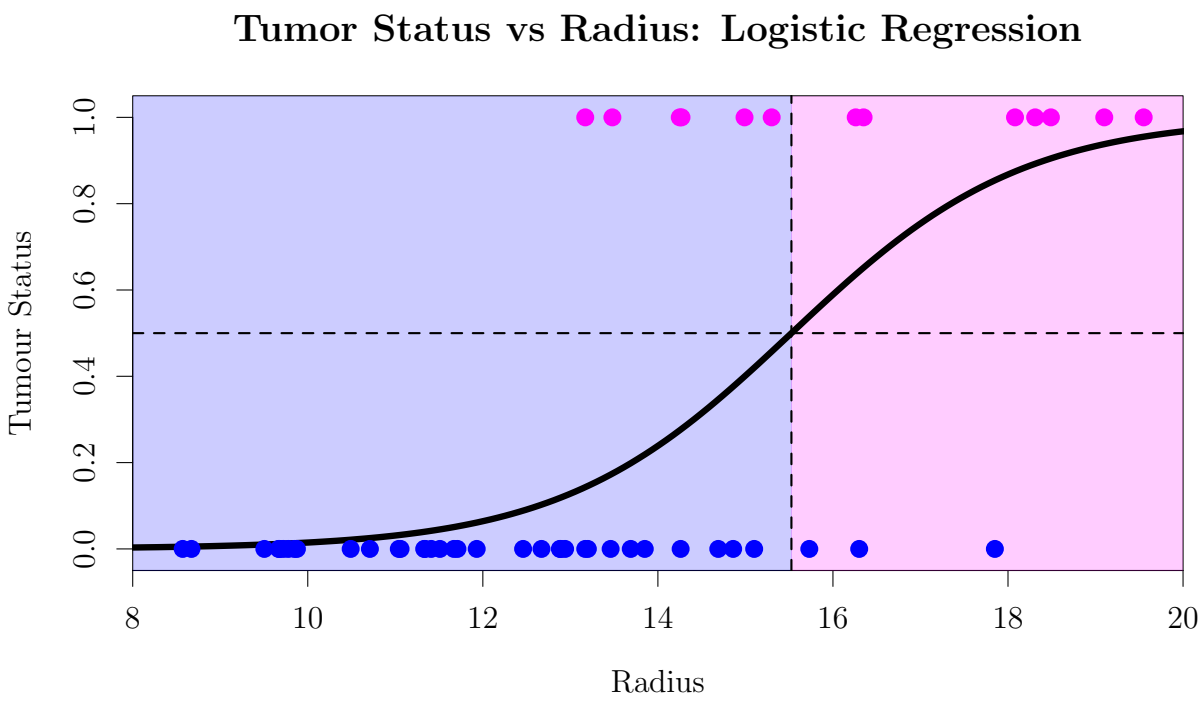

Figure 3.2: We repeat the analysis from Figure 3.1 here, but now using Logistic regression. We can see visually that the curve is monotonically increasing but still bounded between 0 and 1, allowing us to interpret the value of the curve at any point as the likelihood of the tumour being malignant. Further, we can see that the decision point has moved to the left

Note that if we instead allow Y to follow a ± 1 coding, then the Logistic regression problem takes the simpler form

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \sum_{i=1}^{n} \ln\Big(1 + e^{-y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big). \tag{3.8}$$

3.2 Logistic Regression with MaxNS Data

We will modify our task here. We still make the assumptions that, for some $\boldsymbol{w}, b$,

$$\mathbb{P}(Y = 1 \,|\, \boldsymbol{X} = \boldsymbol{x}) = \frac{1}{1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right)}},$$

and our goal is still to find $\boldsymbol{w}$ and b. However, assume now that we have been provided with MaxNS data $\big(\boldsymbol{x}_{[k]i}, y_{[k]i}\big)_{i=1}^{n}$.

Following Result 7 in Chapter 2, we may say that

$$\mathbb{P}\big(Y_{[k]} = 1 \,|\, \boldsymbol{X}_{[k]} = \boldsymbol{x}\big) \approx \left\{\frac{1}{1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right)}}\right\}^{\omega_0},$$

for some $0 \leq \omega_0 \leq 1$. We will repeat the technique of minimizing the negative log-likelihood to find $\boldsymbol{w}$ and b. We now have that

$$\mathbb{P}\big(Y_{[k]} = y \,|\, \boldsymbol{X}_{[k]} = \boldsymbol{x}\big) \approx \begin{cases} \left\{\frac{1}{1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right)}}\right\}^{\omega_0}, & \text{if } y = 1, \\ 1 - \left\{\frac{1}{1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right)}}\right\}^{\omega_0}, & \text{if } y = -1. \end{cases}$$

Thus, the summands of the negative log-likelihood are now

$$\ell_i(\boldsymbol{w}, b;\, \boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \omega_0 \ln\Big(1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big), & \text{if } y_i = 1. \\ -\ln\Big(1 - \left\{\frac{1}{1+e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}}\right\}^{\omega_0}\Big), & \text{if } y_i = -1. \end{cases} \tag{3.9}$$

Therefore, the program is

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \omega_0 \sum_{i:y_i=1} \ln\Big(1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big) - \sum_{i:y_i=-1} \ln\left(1 - \left\{\frac{1}{1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}}\right\}^{\omega_0}\right). \tag{3.10}$$

We will refer to this as the MaxNS Logistic Regression (MNSLR) problem. In particular, we note here that (3.9) is the MNSLR Loss Function.

3.3 Solving the MNSLR Problem

In this section we will provide a numerical technique for solving the MNSLR problem. In particular, we will develop a gradient descent algorithm for solving (3.10).

Gradient descent, first proposed by Curry in 1944 [52], is a popular optimization technique that has been applied to a wide array of problems [53]. The technique is intuitive: Suppose that we have a function $f : \mathbb{R}^d \to \mathbb{R}$ to be minimized. Starting from a certain point $\boldsymbol{x}_0$, we take iterative steps, always in the direction opposite of the gradient of f. I.e., the update step is $\boldsymbol{x}_{(t+1)} \leftarrow \boldsymbol{x}_{(t)} - \delta \nabla f\big(\boldsymbol{x}_{(t)}\big)$, where δ is our step size. We conclude the procedure when some stopping condition is met. A common stopping condition is of the form $||\nabla f(\boldsymbol{x})||_2 \leq \zeta$, where ζ is a small positive value and $||\boldsymbol{x}||_2 \coloneqq \sqrt{x_1^2 + \dots x_d^2}$ is the (L_2) norm of $\boldsymbol{x}$.

In Figure 3.3 we provide a visual example of the path taken by the gradient descent algorithm for a simple function.

For the purposes of this derivation, define $\boldsymbol{\beta} = (b, \boldsymbol{w})$, and $\boldsymbol{x}^* = (1, \boldsymbol{x})$. For brevity, set $z_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i^*$. Then, our goal is to minimize the function

$$f(\boldsymbol{\beta}) = \omega_0 \sum_{i:y_i=1} \ln\big(1 + e^{-z_i}\big) - \sum_{i:y_i=-1} \ln\left(1 - \left\{\frac{1}{1+e^{-z_i}}\right\}^{\omega_0}\right). \tag{3.11}$$

First, note that $f(\boldsymbol{\beta}) = \sum_{i=1}^n \ell_i\big(\boldsymbol{\beta};\, \boldsymbol{x}_i^*, y_i\big)$, where ℓ_i is as defined in (3.9). The gradient of ℓ_i can be shown to be

$$\nabla \ell_i\big(\boldsymbol{\beta};\, \boldsymbol{x}_i^*, y_i\big) = \begin{cases} -\dfrac{\omega_0}{1+e^{z_i}} \boldsymbol{x}_i^*, & \text{if } y_i = 1, \\ \dfrac{\omega_0}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0} - 1]} \boldsymbol{x}_i^*, & \text{if } y_i = -1. \end{cases}$$

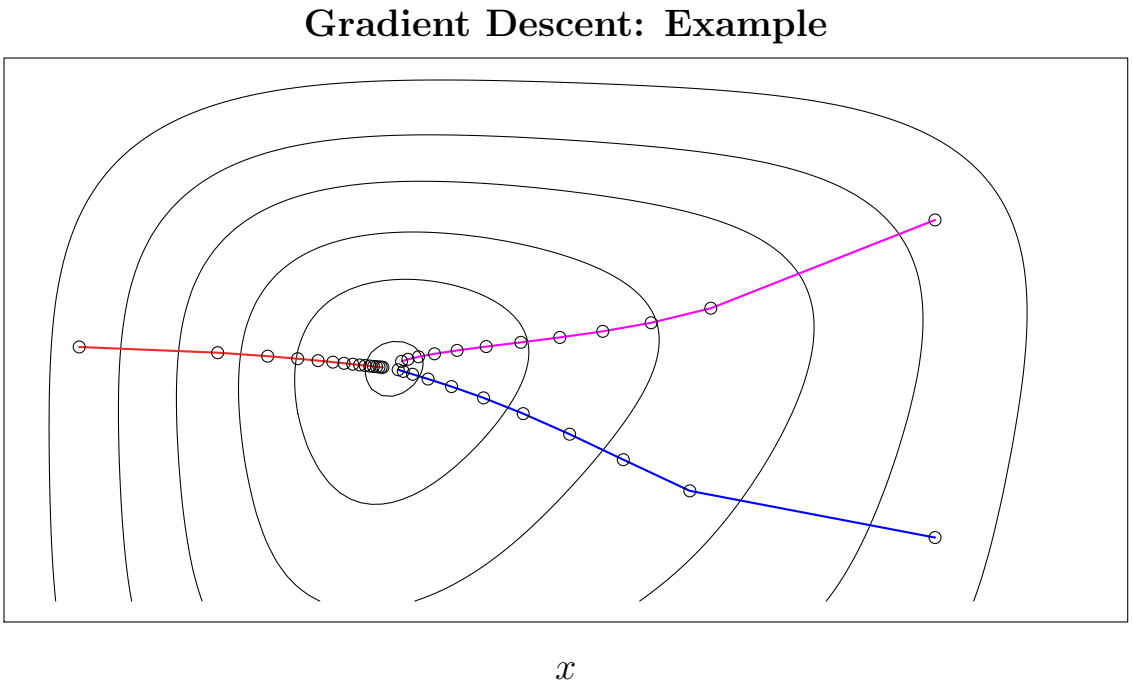

Figure 3.3: We demonstrate the Gradient Descent procedure here, for the objective function $f(x,y) = \frac{1}{2}\big(x^2+y^2\big) + \frac{1}{4}\big[(x-3)^4 + (y+3)^4\big] - 0.5(x-5)(y+3)$. *The black lines represent the contours of* f *at selected levels, while the blue, magenta, and red paths represent the paths taken by the algorithm for three different starting points. We can see that the algorithm quickly reaches the objective function in each case.*

Hence, the gradient of (3.11) is

$$\begin{aligned}\nabla f(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \nabla \ell_i\big(\boldsymbol{\beta};\, \boldsymbol{x}_i^*, y_i\big) \\ &= -\omega_0 \left(\sum_{i:y_i=1} \frac{1}{1+e^{z_i}} \boldsymbol{x}_i^* - \sum_{i:y_i=-1} \frac{1}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0}-1]} \boldsymbol{x}_i^* \right).\end{aligned}$$

Hence, we arrive at the following algorithm for solving the MNSLR problem:

Algorithm 1. *To solve the MNSLR problem, given a MaxNS training set* $\big(\boldsymbol{x}_{[k]i}, y_{[k]i}\big)_{i=1}^{n}$, *a step size* $\delta > 0$, *and a tolerance* $\zeta > 0$.

1. *Pick a starting point* $\big(b_{(0)}, \boldsymbol{w}_{(0)}\big) \in \mathbb{R} \times \mathbb{R}^d$

2. For $t = 0, 1, 2, \ldots$

(a) Calculate the directional updates for $\boldsymbol{w}$ *and* b*:*

$$\nabla_b f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big) = -\omega_0 \left(\sum_{i:y_i=1} \frac{1}{1+e^{z_i}} - \sum_{i:y_i=-1} \frac{1}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0} - 1]} \right),$$

$$\nabla_{\boldsymbol{w}} f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big) = -\omega_0 \left(\sum_{i:y_i=1} \frac{\boldsymbol{x}_i}{1+e^{z_i}} - \sum_{i:y_i=-1} \frac{\boldsymbol{x}_i}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0} - 1]} \right).$$

(b) Check $\big|\big|\nabla f\big(\boldsymbol{\beta}_{(t)}\big)\big|\big|_2 \leq \zeta$*. If so, stop. If not, continue.*

(c) Update $\boldsymbol{w}$ *and* b*:*

$$\boldsymbol{w}_{(t+1)} \leftarrow \boldsymbol{w}_{(t)} - \delta \nabla_{\boldsymbol{w}} f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big)$$

$$b_{(t+1)} \leftarrow b_{(t)} - \delta \nabla_b f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big)$$

3.4 Convergence Analysis

While there are several conditions under which one may prove convergence of the gradient descent algorithm, the standard conditions insist that the objective function f is (1) *convex* and (2) has a *Lipschitz continuous* gradient. We will establish here that the function f in the MNSLR problem satisfies both conditions.

First, a very short discussion of convex optimization is relevant. For a full exploration, see [54].

A function f is said to be *convex* on the set $A \subseteq \mathbb{R}^d$ if, for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$ and for all $t \in [0, 1]$,

$$f(t\boldsymbol{x}_1 + (1-t)\boldsymbol{x}_2) \leq tf(\boldsymbol{x}_1) + (1-t)f(\boldsymbol{x}_2).$$

Intuitively, this means that, for any two points within S, the straight line connecting these two points falls on or above the graph of the function along this line. If this inequality is strict on $t \in (0, 1)$, then we say that f is *strictly convex*. Examples are given in Figure 3.4.

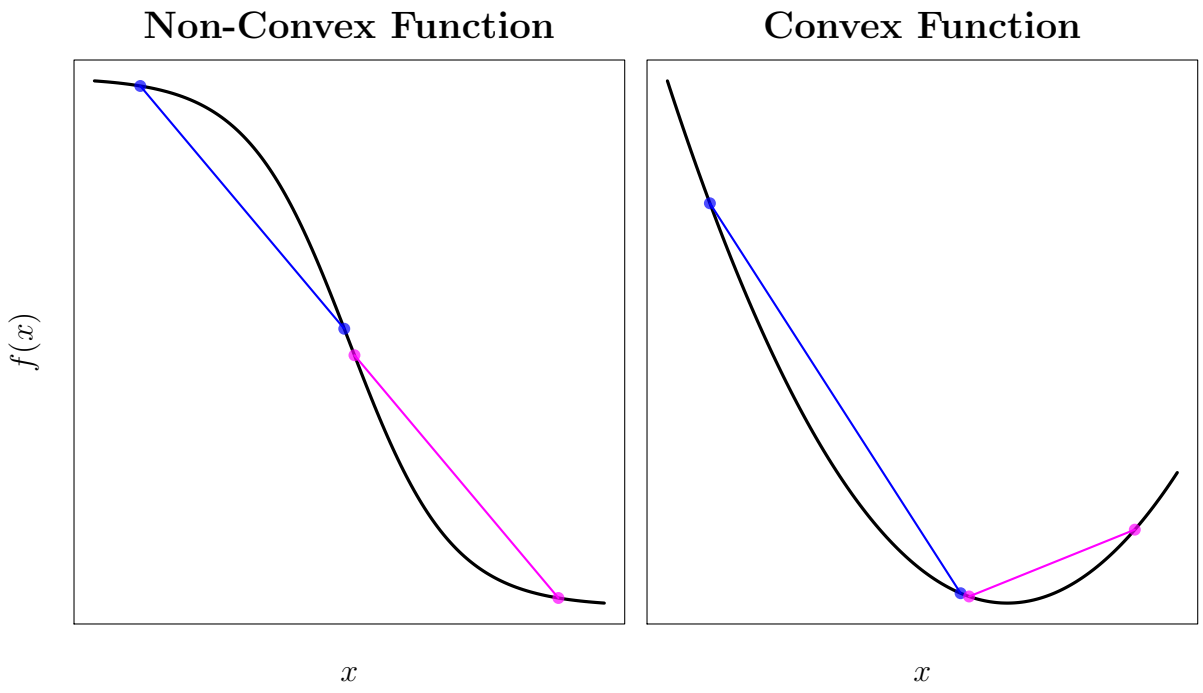

Figure 3.4: Here we give two functions. The function on the left is visibly non-convex, as the blue line falls above the curve of the function. The function on the right is strictly convex, and we can see that both lines are falling above the graph.

Some useful properties of convex functions are enumerated in the theorem below:

Theorem 1. *1. A twice-differentiable function f of one variable is (strictly) convex on $S \subseteq dom(f)$ if and only if its second derivative is (strictly) positive on S. This is known as the second-order condition.*

2. Let f be a real function of one variable. If f is (strictly) convex, then $g(\boldsymbol{x}) \coloneqq f\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right)$ is (strictly) convex for any $\boldsymbol{w} \in \mathbb{R}^d$ and any $b \in \mathbb{R}$. This is known as an affine composition.

3. Let $f(x) = \ln(1 + e^x)$. Then f is strictly convex. This is known as a log-sum-exp function.

4. Let f be a (strictly) convex function. Then af is (strictly) convex for any $a > 0$.

5. *Let f_1, f_2 be convex functions. Then $f_1 + f_2$ is convex. Further, if at least one of f_1, f_2 are strictly convex, then $f_1 + f_2$ is strictly convex.*

See [54] for more details.

An *optimization problem* is a problem of the form

$$\begin{aligned} &\underset{\boldsymbol{x}\in\boldsymbol{S}}{\text{minimize}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \quad g_1(\boldsymbol{x}) < 0,\, g_2(\boldsymbol{x}) < 0,\, \ldots,\, g_m(\boldsymbol{x}) < 0. \end{aligned}$$

The function $f : \boldsymbol{S} \to \mathbb{R}$ in the above formulation is referred to as the *objective function* of the optimization problem, while the g_i are the constraint functions. Any point $\boldsymbol{x} \in \boldsymbol{S}$ that satisfies the constraints $g_1(\boldsymbol{x}) < 0, \ldots, g_m(\boldsymbol{x}) < 0$ is called a feasible point, and any feasible point $\boldsymbol{x}^\star$ that minimizes f on $\boldsymbol{S}$ is referred to as an *optimal solution*. If the functions $f, g_1, g_2, \ldots, g_m$ are all convex on $\boldsymbol{S}$, then it is referred to as a convex optimization problem.

We have the following result concerning strictly convex optimization problems, again taken from [54].

Theorem 2. *Consider a convex optimization problem*

$$\underset{\boldsymbol{x}\in\boldsymbol{S}}{\textit{minimize}}\, f(\boldsymbol{x}).$$

If f is strictly convex, then the optimal solution (when it exists) is unique.

We will use the properties of convex functions listed above to prove that the objective function of the MNSLR problem is strictly convex.

Result 9. *Let f be defined as in (3.11). Then f is strictly convex for any $0 < \omega_0 < 1$.*

Proof. First, the function $g(x) \coloneqq \omega_0 \ln(1 + e^x)$ is strictly convex from (3) and (4). Next, we wish to show that the function h, defined as

$$h(x) \coloneqq -\ln\left(1 - \left[\frac{1}{1+e^{-x}}\right]^{\omega_0}\right)$$

is convex for any $0 < \omega_0 < 1$. Differentiating this function twice gives us the result

$$h''(x) = \frac{\omega_0(1+e^{-x})^{-\omega_0}\left[\omega_0 + e^x\left[(1+e^{-x})^{-\omega_0} - 1\right]\right]}{\left[(1+e^x)^2\left[(1+e^{-x})^{-\omega_0} - 1\right]\right]^2}.$$

To show positivity of h'', it will be sufficient to show that

$$\omega_0 + e^x\left[\left(1+e^{-x}\right)^{-\omega_0} - 1\right] > 0 \quad \text{for all } x \in \mathbb{R}.$$

After some algebraic manipulations, we find that the above condition is equivalent to

$$\left(1+e^{-x}\right)^{-\omega_0} > 1 - \omega_0 e^{-x} \text{ for all } x \in \mathbb{R}.$$

Which is nothing more than Bernoulli's Inequality, and hence it is true. Thus, h'' is strictly positive, which implies that h is strictly convex by the second-order condition.

Next, recall that affine compositions with strictly convex functions preserve strict convexity. I.e., $g_i^*(\boldsymbol{\beta}) \coloneqq g\left(-\boldsymbol{\beta}^\top \boldsymbol{x_i^*}\right)$ and $h_i^*(\boldsymbol{\beta}) \coloneqq h\left(-\boldsymbol{\beta}^\top \boldsymbol{x_i^*}\right)$ are strictly convex. Finally, since

$$f(\boldsymbol{\beta}) = \sum_{i:y_i=-1} g_i^*(\boldsymbol{\beta}) + \sum_{i:y_i=-1} h_i^*(\boldsymbol{\beta}),$$

i.e., since f is a sum of strictly convex functions, f is itself strictly convex. □

Hence, we have as a result that (3.10) has a unique solution, when it is feasible.

Result 10. *Suppose that a solution exists to the MNSLR problem, for a particular dataset* $(\boldsymbol{x}_i, y_i)_{i=1}^n$. *Then, this solution must be unique.*

To show that Algorithm 1 converges to a solution, we must also show that the gradient of f is Lipschitz continuous, where Lipschitz continuity refers to a particular strong form of uniform continuity. That is, a vector-valued function $\boldsymbol{f} : \mathbb{R}^{d_1} \to \mathbb{R}^d$ is said to be Lipschitz continuous with constant $L > 0$ if,

$$\text{for all } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \text{dom}(\boldsymbol{f}), \quad ||\boldsymbol{f}(\boldsymbol{x}_1) - \boldsymbol{f}(\boldsymbol{x}_2)||_2 \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2||_2^2$$

Below are some important properties of Lipchitz functions, taken from [55]

Theorem 3. *1. If f is a differentiable function of a single variable, and the derivative of f is bounded, then f is Lipschitz continuous. Further, the Lipschitz constant is $L = \max|f'(x)|$.*

2. If f and g are both Lipschitz continuous functions of a single variable, with Lipschitz constants L_f and L_g respectively, then $f + g$ is Lipschitz continuous with constant $L = \max(L_f, L_g)$.

3. If f and g are both Lipschitz continuous functions of a single variable, with Lipschitz constants L_f and L_g respectively, then $f \circ g$ is Lipschitz continuous with constant $L = L_f L_g$.

We will use these properties to show that the gradient of the objective function in the MNSLR problem is Lipschitz continuous.

Result 11. *Let f be defined as in (3.11). Then f has a Lipschitz-continuous gradient.*

Proof. This proof will require some preliminary results.

Let g and h be as in the proof of Result 9. We will first show that g' and h' are Lipschitz continuous.

To show that g' is Lipschitz continuous, note that its second derivative is given by

$$g''(x) = \frac{\omega_0 e^x}{(1+e^{-x})^2} = \frac{\omega_0 e^{-x}}{(1+e^x)^2} = g''(-x).$$

That is, g'' is symmetric about 0, and is equal to $\frac{\omega_0}{2}$ at 0. Since g is strictly convex, that makes $\frac{\omega_0}{2}$ an upper bound. Since g'' vanishes at positive and negative infinity, it is bounded below by 0. Thus, g' is Lipschitz continuous with constant $L_{g'} = \frac{\omega_0}{2}$.

Next, to show that h' is Lipschitz continuous, note that its derivative is given by

$$h''(x) = \frac{\omega_0 e^x (1+e^x)^{-(2+\omega_0)}\left[(1+e^x)^{-\omega_0} + \omega_0 e^x - 1\right]}{\left[1-(1+e^x)^{-\omega_0}\right]^2}.$$

It can be shown through elementary techniques that $\lim_{x\to-\infty} h''(x) = \lim_{x\to\infty} h''(x) = 0$. Further, by inspection, we see that h'' is continuous. Thus, it must be bounded by some upper value $L_{h'}(\omega_0)$. This upper bound must be found through numerical techniques (see Figure 3.5).

We can now continue with the proof. Recall that we seek to bound $\Delta \coloneqq ||\nabla f(\boldsymbol{\beta}_1) - \nabla f(\boldsymbol{\beta}_2)||$. For brevity, let $z_i = \boldsymbol{\beta}_1^\top \boldsymbol{x}_i^*$ and let $z_i' = \boldsymbol{\beta}_2^\top \boldsymbol{x}_i^*$. From Algorithm 1,

$$\Delta = \Bigg|\Bigg| -\omega_0 \Bigg\{ \sum_{i:y_i=1} \left(\frac{1}{1+e^{z_i}} - \frac{1}{1+e^{z_i'}} \right) \boldsymbol{x}_i^* + \sum_{i:y_i=-1} \left(\frac{1}{(1+e^{z_i})[(1+e^{-z_i})-1]} - \frac{1}{\left(1+e^{z_i'}\right)\left[\left(1+e^{-z_i'}\right)-1\right]} \right) \boldsymbol{x}_i^* \Bigg\} \Bigg|\Bigg|.$$

Now, using the triangle inequality,

$$\Delta \leq \omega_0 \Bigg(\sum_{i:y_i=1} \left| \frac{1}{1+e^{z_i}} - \frac{1}{1+e^{z_i'}} \right| ||\boldsymbol{x}_i^*|| + \sum_{i:y_i=-1} \left| \frac{1}{(1+e^{z_i})[(1+e^{-z_i})-1]} - \frac{1}{\left(1+e^{z_i'}\right)\left[\left(1+e^{-z_i'}\right)-1\right]} \right| ||\boldsymbol{x}_i^*|| \Bigg).$$

Using the results above,

$$\begin{aligned} \Delta &\leq \omega_0 \left(\sum_{i:y_i=1} \frac{1}{4} \left|\left|\boldsymbol{x}_i^*\right|\right|^2 ||\boldsymbol{\beta}_1 - \boldsymbol{\beta}_2|| + \sum_{i:y_i=-1} L(\omega_0) \left|\left|\boldsymbol{x}_i^*\right|\right|^2 ||\boldsymbol{\beta}_1 - \boldsymbol{\beta}_2|| \right) \\ &= \omega_0 \left(\sum_{i=1}^n L_i \left|\left|\boldsymbol{x}_i^*\right|\right| \right) ||\boldsymbol{\beta}_1 - \boldsymbol{\beta}_2||, \end{aligned}$$

i.e., ∇f is Lipschitz continuous with constant

$$L = \omega_0 \left(\sum_{i:y_i=1} \frac{1}{4} \left|\left| \boldsymbol{x}_i^* \right|\right|^2 + \sum_{i:y_i=-1} L(\omega_0) \left|\left| \boldsymbol{x}_i^* \right|\right|^2 \right).$$

□

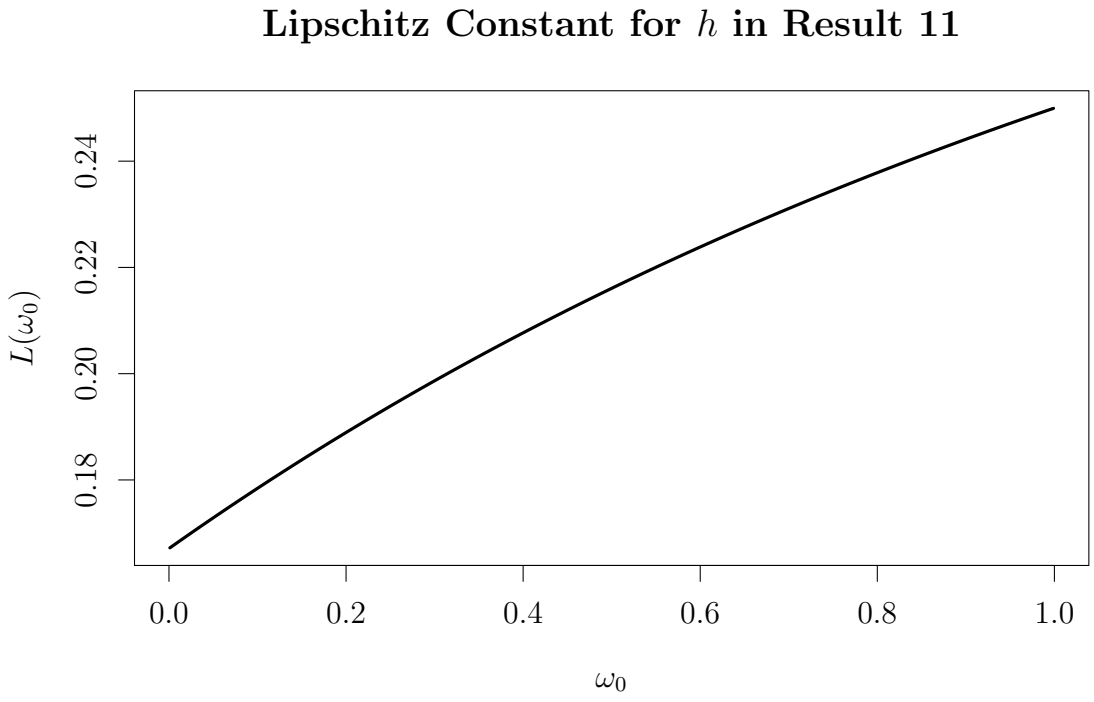

Figure 3.5: Here we present the Lipschitz constant $L(\omega_0)$ from the proof of Result 11. We see that this function is bounded above by $\frac{1}{4}$, and in fact converges to $\frac{1}{4}$ as ω_0 tends to 1. This is to be expected: the Lipschitz constant for the associated Logistic regression problem is $\frac{1}{4}$, and at ω_0, the MNSLR problem reduces to the standard Logistic regression problem.

The following theorem is taken from [56]:

Theorem 4. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$. Then, if we run gradient descent for I iterations with a fixed step size $\delta = 1/L$, it will yield a solution $f^{(I)}$ which satisfies*

$$f\left(\boldsymbol{x}^I\right) - f(\boldsymbol{x}^\star) - \frac{\left|\left|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\right|\right|}{2\delta I},$$

where $f(\boldsymbol{x}^\star)$ is the optimal value. I.e., this means that gradient descent is guaranteed to converge and that it does so with rate $O(1/I)$.

Hence, Results 9 and 11 together with Theorem 1 serve as a proof of the following result:

Result 12. *For a sufficiently small step size δ, Algorithm 1 for solving the MNSLR problem is guaranteed to converge with rate $O(1/I)$.*

3.5 Regularization

One issue of concern in classification methods based upon regression models is that of multicollinearity [57], i.e., the phenomenon wherein one or more explanatory variable in a multiple regression model can be linearly predicted from the the other explanatory variables with a considerable degree of accuracy. It is well-documented [58] that in the presence of multicollinearity, the coefficient estimates of the model will display extreme variance.

One popular solution to the concern of multicollinearity is the technique known as Ridge regression, first introduced to the linear regression context in 1970 [59], and later applied to logistic regression [60]. The Logistic ridge regression model is given below:

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \sum_{i=1}^{n} \ln\Big(1 + e^{-y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\Big) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2, \tag{3.12}$$

where $\lambda > 0$ is a tuning parameter of the model. This model is quite similar in nature to (3.8), with the addition of the new $\frac{\lambda}{2}||\boldsymbol{w}||^2$ term, sometimes referred to as a *penalty*. By encouraging the norm of $\boldsymbol{w}$ to be small, we control the variability of the coefficient estimates $\hat{w}_i$. However, this is not free, as it does result in some estimator bias [59].

In fact, this new penalty term does not just encourage the norm of $\boldsymbol{w}$ to be small: it forces $\boldsymbol{w}$ within a ball of radius t, centred at the origin. This is demonstrated in the theorem below, again taken from [54].

Theorem 5. *Let $\mathcal{L}_\lambda$ denote a convex optimization problem of the form*

$$\underset{\boldsymbol{x}\in\boldsymbol{S}}{\text{minimize}} \quad f(\boldsymbol{x}) + \lambda g(\boldsymbol{x}),$$

with f, g convex and $\lambda > 0$, and let $p^\star_\lambda$ denote a solution to $\mathcal{L}(\lambda)$. Let $\mathcal{C}_t$ denote a convex optimization problem of the form

$$\underset{\boldsymbol{x}\in\boldsymbol{S}}{\text{minimize}} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad g(\boldsymbol{x}) < t,$$

and let $d^\star_t$ represent the optimal value of this problem. Then, for any $\lambda > 0$, there exists a $t \geq 0$ such that the problems $\mathcal{L}_\lambda$ and $\mathcal{C}_t$ are equivalent in the sense that $p^\star_\lambda = d^\star_t$.

We may use Theorem 5 to help us visualize the Ridge regression problem. In particular, as the set $\left\{\boldsymbol{w} : ||\boldsymbol{w}||_2^2 < t^2\right\}$ corresponds to the d-dimensional ball of radius t, we may visualize the Ridge Regression problem as in Figure 3.6.

We may similarly apply the Ridge penalty to the MNSLR model, resulting in the model below:

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \omega_0 \sum_{i:y_i=1} \ln\left(1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}\right) - \sum_{i:y_i=-1} \ln\left(1 - \left\{\frac{1}{1 + e^{-\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)}}\right\}^{\omega_0}\right) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2. \tag{3.13}$$

We will refer to this model as the MaxNS Logistic Ridge Regression (MNSLRR) problem.

We may easily extend Algorithm 1 for solving the MNSLR problem to this new problem.

Algorithm 2. *To solve the MNSLRR problem, given a MaxNS training set $\left(\boldsymbol{x}_{[k]i}, y_{[k]i}\right)_{i=1}^n$, a step size $\delta > 0$, a tuning parameter $\lambda > 0$, and a tolerance $\zeta > 0$.*

1. *Pick a starting point $\left(b_{(0)}, \boldsymbol{w}_{(0)}\right) \in \mathbb{R} \times \mathbb{R}^d$*

2. *For $t = 0, 1, 2, \ldots$*

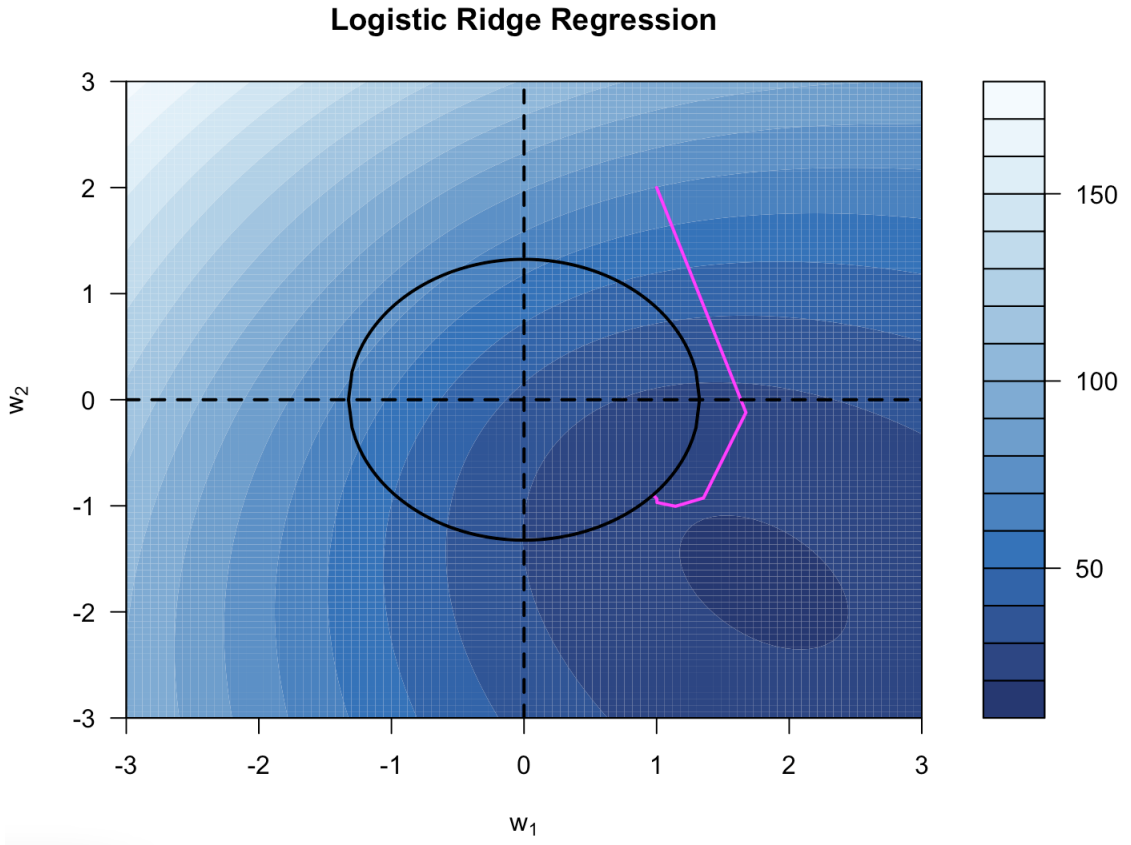

Figure 3.6: Above is a visualization of our gradient descent procedure applied to the Logistic Ridge regression problem, with $\lambda = 7$. While the loss function is minimized at $\boldsymbol{w} = (1.38, -1.79)$, the penalty term restricts β to a circle centred on the origin. The magenta line shows the gradient descent path.

(a) Calculate the directional updates for $\boldsymbol{w}$ and b:

$$\nabla_b f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big) = -\omega_0 \left(\sum_{i:y_i=1} \frac{1}{1+e^{z_i}} - \sum_{i:y_i=-1} \frac{1}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0} - 1]} \right),$$

$$\nabla_{\boldsymbol{w}} f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big) = \lambda \boldsymbol{w}_{(t)} - \omega_0 \left(\sum_{i:y_i=1} \frac{\boldsymbol{x}_i}{1+e^{z_i}} - \sum_{i:y_i=-1} \frac{\boldsymbol{x}_i}{(1+e^{z_i})[(1+e^{-z_i})^{\omega_0} - 1]} \right).$$

(b) Check $\big|\big|\nabla f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big)\big|\big|_2 \leq \zeta$. If so, stop. If not, continue.

(c) Update $\boldsymbol{w}$ and b:

$$\boldsymbol{w}_{(t+1)} \leftarrow \boldsymbol{\beta}_{(t)} - \delta \nabla_{\boldsymbol{w}} f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big),$$

$$b_{(t+1)} \leftarrow b_{(t)} - \delta \nabla_b f\big(\boldsymbol{w}_{(t)}, b_{(t)}\big).$$

Note here that since we have only added the term $\frac{\lambda}{2}||\boldsymbol{w}||_2^2$ to the objective, which is smooth and has a Lipschitz continuous gradient, Results 9 and 11 extend immediately to (3.13). Thus, we have that Algorithm 2 is also convergent. We state this as a separate result below.

Result 13. *For a sufficiently small step size δ, Algorithm 2 for solving the MNSLRR problem is guaranteed to converge with rate $O(1/I)$.*

3.6 Loss

Returning to the form of the Logistic Ridge regression problem, recall that it may be formulated as below (when Y follows a ± 1 coding):

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \sum_{i=1}^{n} L_{\text{LRR}}\big(y_i, \boldsymbol{w}^\top \boldsymbol{x}_i + b\big) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2,$$

where $L_{\text{LRR}}(y, f(\boldsymbol{x})) = \ln\big(1 + e^{-yf(\boldsymbol{x})}\big)$ is the associated loss function of the problem.

This is a general form of problem studied in Statistical Learning Theory, known as the Loss + Penalty form. This general form is instructive, as one may begin to examine the effects modifying either the Loss Function, or the Penalty. For example, by changing the penalty from an L_2-norm $||\boldsymbol{w}||_2 = \sqrt{\sum w_i^2}$ to an L_1-norm $||\boldsymbol{w}||_1 \coloneqq \sum|w_i|$, one arrives at the popular LASSO problem, introduced by Tibshrani in 1996 [61].

On the other hand, one may alter the loss function. As one example, the MNSLRR problem introduced in this chapter is

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \sum_{i=1}^{n} L_{\text{MNSLRR}}\big(y_i, \boldsymbol{w}^\top \boldsymbol{x}_i + b\big) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2,$$

where

$$L_{\text{MNSLRR}}(y, f(\boldsymbol{x})) = \begin{cases} \omega_0 \ln\big(1 + e^{-f(\boldsymbol{x})}\big), & \text{if } y = 1, \\ -\ln\Big(1 - \Big\{\frac{1}{1+e^{-f(\boldsymbol{x})}}\Big\}^{\omega_0}\Big), & \text{if } y = -1 \end{cases}$$

is the associated MNSLRR loss function.

In the next chapter, we will examine the similarity between Logistic Ridge regression and the SVM in terms of their associated loss functions. We will then take advantage of this relationship to modify the SVM to properly incorporate MaxNS training data.

Chapter 4

The Support Vector Machine

In this chapter, we develop in detail the Support Vector Machine (SVM), first introduced in Chapter 1. We then demonstrate the link between the SVM and Logistic regression. Using this link, we extend the MaxNS technique to the SVM, following off of the work done in Chapter 3.

4.1 The Hard-Margin Support Vector Classifier

Consider the (binary) classification problem presented in Figure 4.1. We have n training observations $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, with $\boldsymbol{x}_i \in \mathbb{R}^d$, $d = 2$, and $y_i = \pm 1$. Shown with the data are multiple straight lines that divide the data. These lines separate the data and induce a classification rule: if a test observation $\boldsymbol{x}_i$ falls on one side of the line, we classify $y_i = 1$, and if $\boldsymbol{x}_i$ falls on the other side of the line, we make the classification $y_i = -1$.

We can see, though, that there are multiple possible lines that separate the data, all of which perform equally well on the training data. The question arises: which of the lines does the best job, and how do we find it? While all of the lines above will define a classifier with no error on the training set, what we want is the separating line that will minimize our *test* error. Thus, we will seek the line creates the greatest level of distance between the two

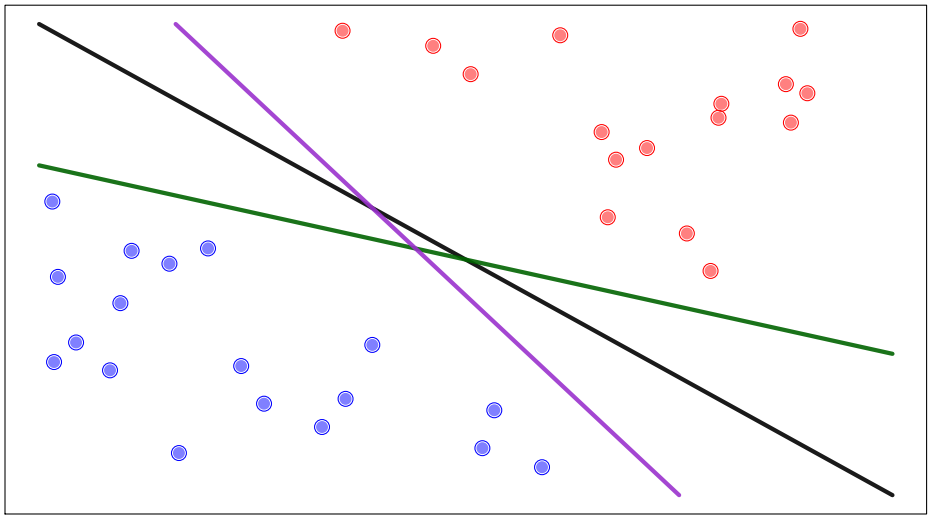

Figure 4.1: Shown here is a simple classification problem. Observations of one class are shown in blue, while observations of another class are shown in red. Three different lines are shown which separate the data. The Support Vector Classifier seeks to find the best such separating line.

classes.

Consider "thickening" the central black line in Figure 4.1, until contact is made with a data point. The perpendicular distance from the central black line to the outer boundary of the shaded area is called the *margin* of the line. We wish to find the separating line that has the largest corresponding margin. We visualize this technique in Figure 4.2.

In general, we allow the predictors $\boldsymbol{x}_i$ to be from $\mathbb{R}^d$, and we seek the *hyperplane* (for two dimensions, this is a line; for three dimensions this is a plane, and so forth) that has the largest corresponding margin. This hyperplane is called the *optimal separating hyperplane*.

Let $f(\boldsymbol{x}) = f_{\boldsymbol{w},b}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$. Recall from linear algebra that $f(\boldsymbol{x}_0)/||\boldsymbol{w}||$ gives the signed distance from $\boldsymbol{x}_0$ to the hyperplane $H = \{\boldsymbol{x} \in \mathbb{R}^d : f(\boldsymbol{x}) = 0\}$; hence, our prediction rule will be

$$G(\boldsymbol{x}) \coloneqq \text{sign}\{f(\boldsymbol{x})\}.$$

We will have a correct prediction when the true sign of y_i agrees with the predicted sign of y_i. In a more succinct form, we will have correct predictions when $y_i f(\boldsymbol{x}_i) > 0$.

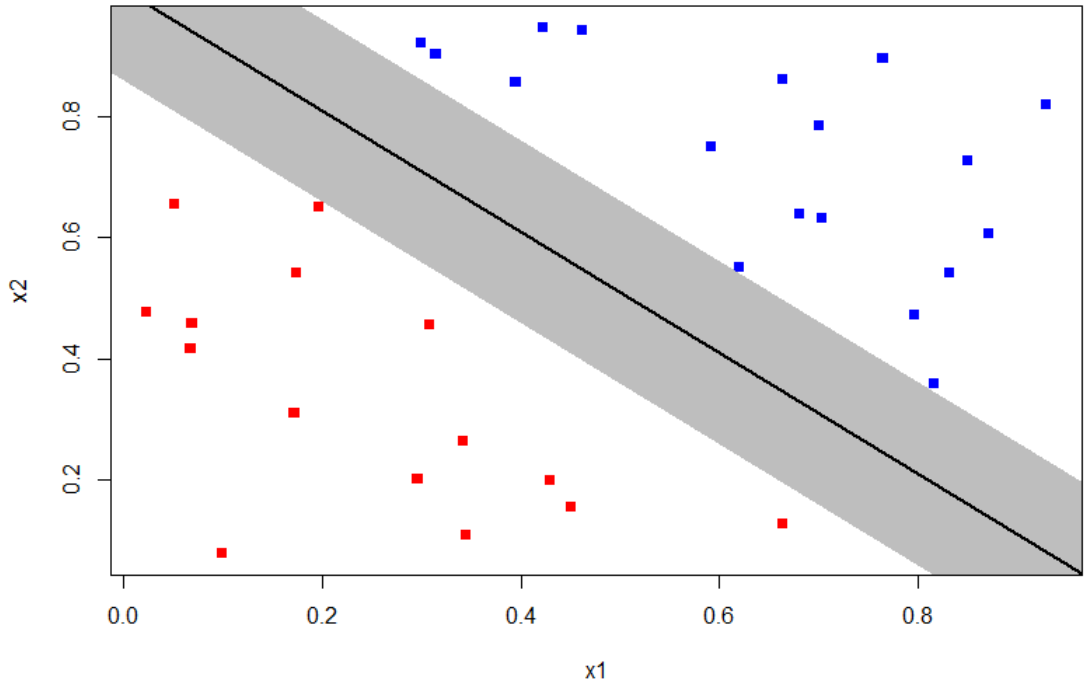

Figure 4.2: Another two-class classification problem with class denoted by colour, as in Figure 4.1. Here we consider one possible separating line, and imagine thickening it on their side until it reaches an observation. This is known as the margin of the separating line.

Let $M(\boldsymbol{w}, b)$ be the margin of H, defined precisely as

$$M(\boldsymbol{w}, b) \coloneqq \min_{i \leq n} \frac{\left|\boldsymbol{w}^\top \boldsymbol{x}_i + b\right|}{||\boldsymbol{w}||}.$$

Our test observation will fall outside (and to the correct side of) the margin exactly when $y_i f(\boldsymbol{x}_i)/||\boldsymbol{w}|| \geq M(\boldsymbol{w}, b)$.

Note that M will also depend on the training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, but we will consider this a fixed parameter of the model and will not include it in our notation.

Summarizing the above, we are seeking the $\boldsymbol{w}$ and b that give rise to the hyperplane $H = \left\{\boldsymbol{x} \in \mathbb{R}^d : f(\boldsymbol{x}) = 0\right\}$ with the largest margin $M(\boldsymbol{w}, b)$, subject to the condition $y_i f(\boldsymbol{x}_i) \geq$

$M(\boldsymbol{w}, b)||\boldsymbol{w}||$, for each $i \leq n$. We state this optimization problem in the below form:

$$\underset{\boldsymbol{w},b}{\text{maximize}} \quad M(\boldsymbol{w}, b) \tag{4.1}$$

$$\text{subject to} \quad y_i f(\boldsymbol{x}_i) \geq M(\boldsymbol{w}, b)||\boldsymbol{w}|| \quad \text{for all } i = 1, \ldots, n. \tag{4.2}$$

However, the form of this optimization problem is problematic from a computation perspective. Primarily, the objective (4.1) is not convex. However, we may take note of the fact that M is scale-invariant. That is, $M(\boldsymbol{w}, b) = M(\kappa\boldsymbol{w}, \kappa b)$ for any $\kappa \in \mathbb{R}$. Thus, for any particular $(\boldsymbol{w}, b)$, we can choose $\kappa = [||\boldsymbol{w}||M(\boldsymbol{w}, b)]^{-1}$, and thereby ensure that $||\boldsymbol{w}|| = 1/M(\boldsymbol{w}, b)$ without affecting the optimal value of M. Hence, condition (4.2) becomes $y_i f(\boldsymbol{x}_i) \geq 1$, and the maximization of M becomes equivalent to the minimization of $||\boldsymbol{w}||$. Thus, we have the equivalent problem below:

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||^2 \tag{4.3}$$

$$\text{subject to} \quad y_i f(\boldsymbol{x}_i) \geq 1 \quad \text{for all } i = 1, \ldots, n. \tag{4.4}$$

This is precisely the hard-margin support vector classifier (SVC) introduced in [62]. While it serves as an interesting and intuitive classification technique, the assumption of a hyperplane that perfectly separates the two classes is extremely strong. In the following sections, we will describe how to relax this assumption to allow not only for class overlap, but for nonlinear separations through the so-called *kernel trick*.

4.2 The Soft-Margin Support Vector Classifier

Let us suppose now that our dataset cannot be perfectly separated into two classes. We will allow for some points to lie on the "wrong" side of the margin, but we still wish to control the total degree of misclassification. To this end, define some variables $\xi_1, \ldots, \xi_n \geq 0$, called *slack*

variables. Re-examining our original optimization problem, condition (4.2) now becomes

$$y_i f(\boldsymbol{x}_i) \geq (1 - \xi_i) M(\boldsymbol{w}, b) \quad \text{for all } i = 1, \ldots n.$$

We also add a new condition to control the total error:

$$\sum_{i=1}^{n} \xi_i \leq K,$$

where $K \geq 0$ is some tuning parameter. We may follow the same steps as before to transform this into a more friendly optimization program

$$\underset{\boldsymbol{w}, b, \boldsymbol{\xi}}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||^2 \tag{4.5}$$

$$\text{subject to} \quad y_i f(\boldsymbol{x}_i) \geq 1 - \xi_i \quad \text{for all } i = 1, \ldots, n, \tag{4.6}$$

$$\xi_i \geq 0 \quad \text{for all } i = 1, \ldots, n, \tag{4.7}$$

$$\sum_{i=1}^{n} \xi_i \leq K, \tag{4.8}$$

which we transform one more final time into the equivalent problem below:

$$\underset{\boldsymbol{w}, b, \boldsymbol{\xi}}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{i=1}^{n} \xi_i \tag{4.9}$$

$$\text{subject to} \quad y_i f(\boldsymbol{x}_i) \geq 1 - \xi_i \quad \text{for all } i = 1, \ldots, n, \tag{4.10}$$

$$\xi_i \geq 0 \quad \text{for all } i = 1, \ldots, n. \tag{4.11}$$

The solution to the problem above will yield the soft-margin support vector classifier. Before we proceed with the development of the SVM, we will instead pause to discuss Lagrangian duality, which provides a simple way to obtain a support vector classifier and will set the stage for further development of the SVM.

4.3 Duality and Computation of the SVC

The optimization program (4.9) – (4.11) is convex, and thus may be solved through various known convex optimization techniques [63, 64]. However, we will introduce here a technique which allows us to transform (4.9) – (4.11) into a *quadratic problem*, a class of optimization programs that is known to be easy to solve (for a full discussion, see [65]). Further, this transformation will set the stage for the application of the so-called *kernel trick*, which will allow for non-linear decisions boundaries to be found.

Consider a convex optimization problem of the form

$$\underset{\boldsymbol{x}\in\boldsymbol{S}}{\text{minimize}} \quad f(\boldsymbol{x}) \tag{4.12}$$

$$\text{subject to} \quad g_i(\boldsymbol{x}) \leq 0, \quad \text{for all } i = 1, \ldots m, \tag{4.13}$$

where $f, g_1, \ldots, g_m$ are convex functions with a shared domain $\boldsymbol{S}$. One obvious (but naive) solution to (4.12) - (4.13) would be to minimize the following function

$$J(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}), & \text{if } g_i(\boldsymbol{x}) \leq 0 \quad \text{for all } i = 1, \ldots, m, \\ \infty, & \text{else.} \end{cases} \tag{4.14}$$

This function gives infinite penalty when a constraint is dissatisfied. This method will work, but the function J is both non-differentiable and discontinuous, and hence optimization of it is very difficult. Hence, we replace it with something nicer:

$$\mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}), \tag{4.15}$$

with $\lambda_i \geq 0$. This is known as the **Langrangian** of (4.12) - (4.13). The constants λ_i will act as penalizers on the functions g_i, depending on the degree to which they fail to satisfy condition 4.13. Thus, if $g_i(\boldsymbol{x}) \leq 0$, we set $\lambda_i = 0$. The Lagrangian is merely a refinement of the more rudimentary function J, as we will show here.

Theorem 6. *Let $\mathscr{L}$ and J be defined as in (4.14) and (4.15). If we take the maximum of $\mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{\lambda}$, we recover J.*

Proof. If (4.13) is satisfied, then $\sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) \leq 0$, since $\lambda_i \geq 0$ for each i. Thus,

$$\sup_{\boldsymbol{\lambda}} \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) = 0.$$

Hence,

$$\sup_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + 0 = J(\boldsymbol{x}).$$

Otherwise, we obtain that

$$\sup_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sup_{\boldsymbol{\lambda}} \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) = f_0(\boldsymbol{x}) + \infty = \infty = J(\boldsymbol{x}).$$

Hence, $\sup_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = J(\boldsymbol{x})$. □

Thus, the problem (4.12) – (4.13) is equivalent to

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}). \tag{4.16}$$

Under sufficient regularity conditions (discussed below), we may reverse maximum and the minimum in (4.16), I.e.,

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \max_{\boldsymbol{\lambda}} \min_{\boldsymbol{x}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) \coloneqq \max_{\boldsymbol{\lambda}} g(\boldsymbol{\lambda}). \tag{4.17}$$

The map $g(\boldsymbol{\lambda}) \coloneqq \min_{\boldsymbol{x}} \mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ is known as the *Langrange dual function*, or the *Lagrangian* of (4.12) – (4.13), and maximizing g subject to $\boldsymbol{\lambda} \geq 0$ is known as the *Lagrange dual problem*, in contrast to the original so-called *primal problem*.

4.3.1 The KKT Conditions

With Lagrange duality comes a set of necessary conditions for optimality of a solution, called the Karush-Kuhn-Tucker (KKT) conditions [65]. They are a central tool in the derivation of Lagrangian dual forms, and we will introduce them here.

Consider the problem (4.12) – (4.13), with Lagrangian

$$\mathscr{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}). \tag{4.18}$$

Suppose further that the functions $f, g_1, \ldots g_m$ are convex and differentiable. Then the KKT conditions are as below:

KKT 1: $g_i(\boldsymbol{x}) \leq 0 \quad \text{for all } i = 1, \ldots, m,$ *(Primal Feasibility)*

KKT 2: $\nabla_{\boldsymbol{x}} \left\{ f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) \right\} = 0,$ *(Stationarity)*

KKT 3: $\lambda_i \geq 0 \quad \text{for all } i = 1, \ldots, m,$ *(Dual Feasibility)*

KKT 4: $\lambda_i g_i(\boldsymbol{x}) = 0 \quad \text{for all } i = 1, \ldots m,$ *(Complementary Slackness)*

KKT 1, Primal Feasibility, is inherited from the primal problem. KKT 2, the Stationarity condition, is nothing more than Fermat's Interior Extremum Theorem, which will be found in any multivariate calculus textbook. KKT 3, Dual Feasibility, is inherited from the dual problem. Finally, KKT 4 is the Complementary Slackness condition and will help us to simplify our dual problem forms. As demonstrated in [54], these conditions are necessary for a solution of a dual problem to be the optimal, and under mild regularity conditions (such as convexity in f and continuous differentiability in g_i) are sufficient as well.

4.3.2 Applying Lagrange Duality to the Soft-Margin SVC

Here we will apply the theory of Lagrange duality to obtain an equivalent form of (4.9) – (4.11) that is easy to solve.

Recall that we are trying to solve the problem (4.9) – (4.11). The corresponding Lagrangian to this problem is

$$\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[y_i f(\boldsymbol{x}_i) - (1-\xi_i)] - \sum_{i=1}^{n}\eta_i\xi_i$$

First, we will satisfy the stationarity condition. We will differentiate with respect to $\boldsymbol{w}$, b, and ξ_i, $i = 1, \ldots, n$ and set the result to zero.

$$\nabla_{\boldsymbol{w}}\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = 0 \quad \Longrightarrow \quad \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i, \tag{4.19}$$

$$\nabla_{b}\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = 0 \quad \Longrightarrow \quad \sum_{i=1}^{n}\alpha_i y_i = 0, \tag{4.20}$$

$$\nabla_{\xi_i}\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = 0 \quad \Longrightarrow \quad \alpha_i = C - \eta_i. \tag{4.21}$$

Substituting (4.19) – (4.20) into $\mathscr{L}$, we arrive at the objective function of the Lagrange dual problem:

$$g(\boldsymbol{\alpha}) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j \tag{4.22}$$

We may combine (4.21) with dual feasibility to obtain that $0 \leq \alpha_i \leq C$ for all $i = 1, \ldots, n$. Thus, the Lagrange dual problem for the soft-margin SVC is

$$\underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{maximize}} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j \tag{4.23}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \ldots n \tag{4.24}$$

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \tag{4.25}$$

In particular, if we let $\mathbf{1} = (1, 1, \ldots, 1)^\top$, a vector of length n, and if we let $\boldsymbol{G}$ denote the $n \times n$ matrix with entries $G_{ij} = y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j$, then we may write (4.23) as

$$\mathbf{1}^\top \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^\top \boldsymbol{G} \boldsymbol{\alpha}.$$

Further, the conditions (4.24) – (4.25) are linear in $\boldsymbol{\alpha}$. Thus, the problem (4.23) – (4.25) is a *quadratic programming problem*, for which there exists a wide range of numerical solution techniques. As mentioned earlier, a full discussion of such problems may be found in [65], including proofs of existence and uniqueness of solutions. In particular, uniqueness of the solution to (4.23) – (4.25) may be found in [66].

Allow $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ to represent the solution to (4.23) – (4.25). Then, by (4.19), we obtain the solution for $\boldsymbol{w}$, given below:

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \boldsymbol{x}_i. \tag{4.26}$$

Note that by the nature of this equality, only observations $\boldsymbol{x}_i$ with corresponding dual variables $\alpha_i > 0$ will contribute to the solution of the primal problem. These vectors are known as the *support vectors* of the optimization program, from which the Support Vector Classifier takes its namesake. I.e., if we define $\mathcal{V}$ by

$$\mathcal{V} = \{i \leq n : \hat{\alpha}_i > 0\}, \tag{4.27}$$

then we can write

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \boldsymbol{x}_i = \sum_{i \in \mathcal{V}} \hat{\alpha}_i y_i \boldsymbol{x}_i. \tag{4.28}$$

The set $\mathcal{V}$ is referred to as the *support set* of the problem.

Invoking the complementary slackness KKT condition, we obtain the following:

$$\alpha_i [y_i f(\boldsymbol{x}_i) - (1 - \xi_i)] = 0 \quad \text{for all } i = 1, \ldots, n, \tag{4.29}$$

and

$$(C - \alpha_i)\xi_i = 0 \quad \text{for all } i = 1, \ldots, n. \tag{4.30}$$

From (4.29), we determine that

$$\alpha_i > 0 \quad \Longrightarrow \quad y_i f(\boldsymbol{x}_i) = 1 - \xi_i. \tag{4.31}$$

In particular, this means that the support vectors are precisely the elements $\boldsymbol{x}_i$ that lie *on*, *within*, or *on the wrong side of* the margin. Further, we derive from (4.30) that

$$\alpha_i < C \quad \Longrightarrow \quad \xi_i = 0. \tag{4.32}$$

Combining (4.31) and (4.32), we obtain that

$$0 < \alpha_i < C \quad \Longrightarrow \quad y_i f(\boldsymbol{x}_i) = y_i\big(\boldsymbol{w}^\top \boldsymbol{x}_i + b\big) = 1. \tag{4.33}$$

After obtaining $\hat{\boldsymbol{w}}$, we may choose any element i such that $0 < \alpha_i < C$ and solve (4.33) to obtain b. For stability, we will typically use an average of all such solutions. We will call this solution $\hat{b}$.

Now that we have obtained $\hat{\boldsymbol{w}}$ and $\hat{b}$, we obtain a separating hyperplane

$$\hat{f}(\boldsymbol{x}) = \hat{\boldsymbol{w}}^\top \boldsymbol{x} + \hat{b} \tag{4.34}$$

and a decision function

$$G(\boldsymbol{x}) = \text{sign}\Big\{\hat{f}(\boldsymbol{x})\Big\} \tag{4.35}$$

In this section we have derived the dual problem to the soft-margin SVC. Note however that we are still restricted to the class of linear decision boundaries. In the next section we will complete the development of the classical SVM by allowing it to generate non-linear decision boundaries.

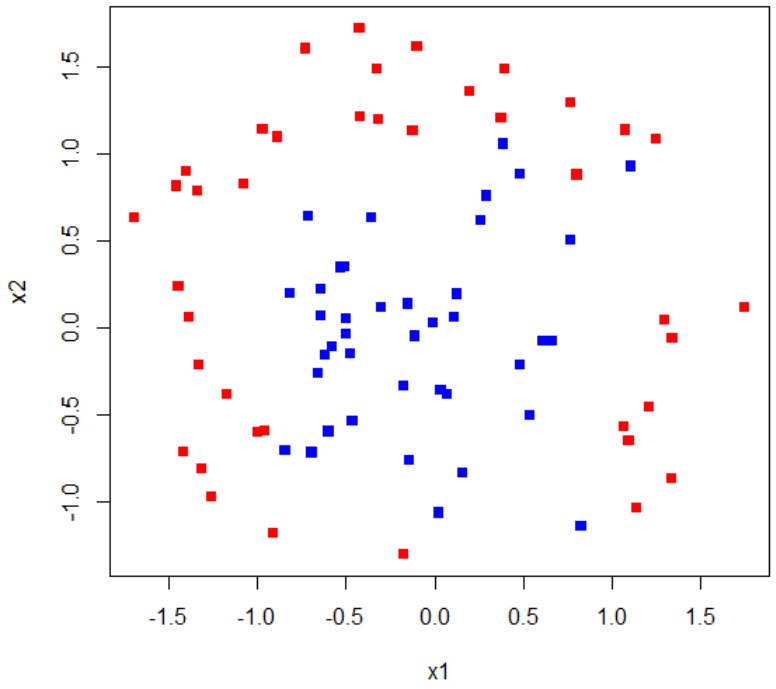

Figure 4.3: A binary classification problem: Class 1 corresponds to the blue points, while Class 2 corresponds to the red points. Though on visual inspection we can see that each class roughly corresponds to a unique connected region, no line $w_1x_1 + w_2x_2 + b = 0$ would achieve a proper separation of the two classes.

4.4 Nonlinear Classification via the Kernel Trick

Consider the classification problem in Figure 4.3. Even with softened margins, there is no meaningful or accurate way to separate these classes linearly. However, suppose we cast each datapoint (x_1, x_2) into $\mathbb{R}^3$ via the mapping $(x_1, x_2) \mapsto (x_1, x_2, x_1^2 + x_2^2)$, as in Figure 4.4.

This transformed dataset, which may trivially be projected back into the space of the original dataset, may be linearly separated at by the plane $z = 2$. This plane may then be projected back into the original feature space, to obtain a circular decision boundary on the original dataset.

This technique of casting a dataset into a transformed feature space and doing classification with the transformed dataset is known as the *kernel trick* and is central to the flexibility of the SVM. We will provide an overview of the theory for this technique in this section and

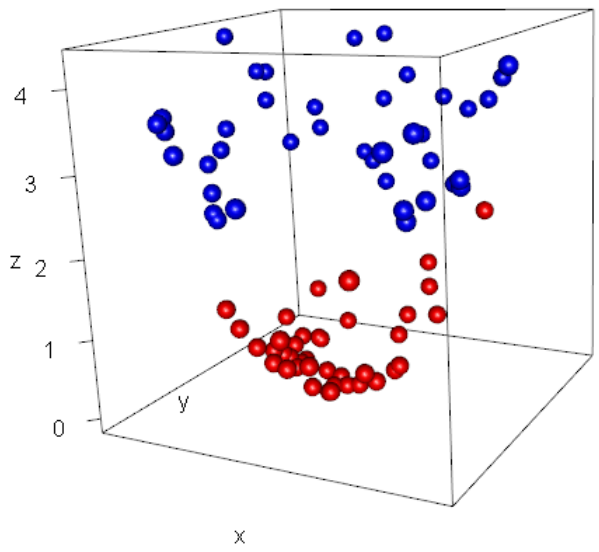

Figure 4.4: The two-dimensional data in Figure 4.3 has been now cast into an expanded space $\mathbb{R}^3$*, by way of the mapping* $(x_1, x_2) \mapsto \left(x_1, x_2, x_1^2 + x_2^2\right)$*. In this enlarged space, the hyperplane defined by* $z = 2$ *will separate the two classes. Projecting this down into* $\mathbb{R}^2$ *will result in a circular decision boundary.*

apply it to the soft-margin SVC to obtain the general SVM.

Let $\boldsymbol{\mathcal{X}} \subseteq \mathbb{R}^d$ be the feature space of the classification problem at hand. Allow $h : \boldsymbol{\mathcal{X}} \to \mathcal{H}$ be some mapping on $\boldsymbol{\mathcal{X}}$ where $\mathcal{H}$ is an inner product space with inner product $\langle \cdot, \cdot \rangle$. The space $\mathcal{H}$ may rather complicated (often infinite-dimensional) with a corresponding inner product that is not efficient to calculate. We will discover, however, that we may take advantage of the properties of the enlarged feature space without getting bogged down in its complexities.

In the space $\mathcal{H}$, the analog of a separating hyperplane will be some elements $\boldsymbol{w} \in \mathcal{H}$, $b \in \mathbb{R}$ such that

$$\langle \boldsymbol{w}, h(\boldsymbol{x}_i) \rangle + b > 0 \quad \text{for all } (\boldsymbol{x}_i, y_i) \text{ such that } y = 1,$$

$$\langle \boldsymbol{w}, h(\boldsymbol{x}_i) \rangle + b < 0 \quad \text{for all } (\boldsymbol{x}_i, y_i) \text{ such that } y = -1,$$

Let $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$. Then, our new optimization program is to solve

$$\underset{\boldsymbol{w} \in \mathcal{H}}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||_{\mathcal{H}}^2 + C \sum_{i=1}^{n} \xi_i \tag{4.36}$$

$$\text{subject to} \quad y_i f(\boldsymbol{x}_i) \geq 1 - \xi_i \quad \text{for all } i = 1, \dots n \tag{4.37}$$

$$\xi_i \geq 0 \quad \text{for all } i = 1, \dots n \tag{4.38}$$

Observe that we are now searching $\mathcal{H}$, a potentially infinite-dimensional space. However, if we take the dual of this problem (see [1] for details), we obtain the dual problem

$$\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{maximize}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4.39}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \dots n \tag{4.40}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0, \tag{4.41}$$

where $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle h(\boldsymbol{x}_i), h(\boldsymbol{x}_j) \rangle$ is the so-called *kernel* of the function (defined further on). Let $\hat{\boldsymbol{\alpha}}$ denote the solution to this problem, and let $\mathcal{V}$ again denote the support set, as in (4.27). Then, as in (4.28), we have the following solution for $\boldsymbol{w}$:

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i h(\boldsymbol{x}_i) = \sum_{i \in \mathcal{V}} \hat{\alpha}_i y_i h(\boldsymbol{x}_i). \tag{4.42}$$

For elements $0 < \alpha_j < C$,

$$\begin{aligned} y_j f(\boldsymbol{x}_j) &= y_j(\langle \boldsymbol{w}, h(\boldsymbol{x_j}) \rangle + b) \\ &= y_j \left(\sum_{i \in \mathcal{V}} \hat{\alpha}_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) + b \right) \\ &= 1, \end{aligned}$$

which may be used to solve for b. Hence, we arrive at a decision boundary function

$$\hat{f}(\boldsymbol{x}) = \sum_{i \in \mathcal{V}} \hat{\alpha}_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + \hat{b} \tag{4.43}$$

and a classification function

$$G(\boldsymbol{x}) = \text{sign}\Big\{\hat{f}(\boldsymbol{x})\Big\}. \tag{4.44}$$

Observe a remarkable fact about the dual formulation (4.39) – (4.41) and the decision boundary (4.43): all entries $h(\boldsymbol{x})$ in the enlarged feature space $\mathcal{H}$ are accessed through the kernel function K. This means that we may utilize the kernel function K without ever needing to explicitly define the mapping h or doing any manipulations in the space $\mathcal{H}$.

Formally, a *kernel* K is a mapping $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle h(\boldsymbol{x}_i), h(\boldsymbol{x}_j)\rangle \quad \text{for all } \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}, \tag{4.45}$$

for some function h, where $h[\mathcal{X}] = \mathcal{H}$ is a *Hilbert space*, and the inner product $\langle \cdot, \cdot \rangle$ is inherited from this space. For a full treatment of Hilbert spaces, see [67].

The explicit forms of h and $\mathcal{H}$ are not needed to confirm that K is a valid kernel either. The *Moore–Aronszajn Theorem* [67] guarantees that, so long as K is symmetric and *positive-definite*, i.e.,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$$

for all finite sequences (of any length) $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ in $\mathcal{X}$ and all scalar sequences $(c_1, \ldots, c_n)$ from $\mathbb{R}$, then K can be uniquely associated with a map h and a space $\mathcal{H}$ that satisfy condition (4.45).

Popular kernels within the SVM literature are given below:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \boldsymbol{x}_j \qquad \text{(Linear)},$$

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \big(1 + \boldsymbol{x}_i^\top \boldsymbol{x}_j\big)^d \qquad (d\text{th Degree polynomial}),$$

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\big(-\gamma ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2\big) \qquad \text{(Radial basis)}.$$

The linear kernel is the kernel used in the soft-margin SVC formulation. The dth degree polynomial kernel, $d \in \mathbb{Z}_{\geq 1}$ will cast the data into an enlarged, but finite space. Finally, the radial basis kernel, a.k.a. the RBF kernel will cast the data into an infinite dimensional space, and is the most widely-used kernel in practice [68]. The tuning parameter $\gamma > 0$ will control the flexibility of the decision boundary: lower values of γ will lead to a more rigid classifier and higher values of γ will lead to a more flexible classifier, though overfitting will become a risk.

Our discussion of the SVM so far has been especially mathematical and algorithmic to this point. In the next section we will provide a view of the SVM that is more statistical in nature.

4.5 The SVM and Loss Functions

It is not difficult to show [1] that the SVM may also be formulated in the Loss + Penalty form, as below:

$$\underset{\boldsymbol{w}, b}{\text{minimize}} \quad \frac{\lambda}{2}||\boldsymbol{w}||^2 + \sum_{i=1}^{n} L_{\text{Hinge}}(y_i, f(\boldsymbol{x}_i)), \tag{4.46}$$

where $L_{\text{Hinge}}(y, f(\boldsymbol{x})) = \max(0, 1 - yf(\boldsymbol{x}))$ denotes the *Hinge* loss function.

Analysis of the Hinge loss reveals that it is rather appropriate for binary classification. Its linearity adds an element of robustness to the optimization procedure, and the assignment of zero loss correct classifications is desirable. Further, we may consider how the loss function is behaving at a population level. In particular, we wish to minimize $\mathbb{E}[L(Y, f(\boldsymbol{X}))]$, the expected loss (i.e., the *risk*) associated with the learning function f according to the loss L. It may be shown that the minimizing function f of the risk associated with the hinge loss function is the map

$$f(\boldsymbol{x}) = \text{sign}\left\{\mathbb{P}(Y = +1|\boldsymbol{X} = \boldsymbol{x}) - \frac{1}{2}\right\}.$$

This is precisely the *Bayes learner*, which is the decision function that minimizes the likelihood of a misclassification. Thus, techniques that seek to minimize the hinge loss function will bring us closer to the minimum-error classifier.

Many variations on the SVM algorithm have been proposed that consider modified loss functions. Vapnik's ε-insensitive loss and the q-norm loss have been applied to increase sparsity [69], the ℓ^2-loss and the LINEX-loss have been applied in [70] and [71] to increase solution efficiency, Chapelle has applied Huber's loss to ease solvability in the primal as well as achieve some robustness [64], and an asymmetric "pinball" loss has been proposed in [72] to achieve noise insensitivity.

We note here the similarity in form between (3.13) and (4.46). We will investigate further here the similarities between the two, and ultimately propose our own modified loss function for training a SVM on MaxNS data.

Let $z = y\big(\boldsymbol{w}^\top \boldsymbol{x} + b\big)$, and let $t = yf(\boldsymbol{x})$ represent the margin of the classification problem. ThenL_{LRR}, the Logistic Ridge regression loss function, may be written in the marginal form $t \mapsto \ln(1 + e^{-t})$, while L_{SVM}, the hinge-loss for the SVM, corresponds to the marginal form $t \mapsto \max(0, 1 - t)$. These two functions are highly similar. By taking advantage of the approximation

$$\ln\big(e^a + e^b\big) \approx \max(a, b),$$

we observe that

$$\ln\big(1 + e^{-t}\big) = \ln\big(e^0 + e^{-t}\big) \approx \max(0, -t).$$

In particular, the two loss functions are asymptotically identical in the positive half of the t-plane, and asymptotically identical up to an additive constant in the negative half of the t-plane.

We will take this to be a fundamental similarity between regularized logistic regression and the hinge-SVM: *The hinge-SVM can be viewed as a "sharpening" of a regularized logistic*

regression program, up to an additive constant.

4.6 The MaxNS SVM

Derivation of the Loss Function

We will combine here the various observations made thus far, to obtain an SVM program that is appropriate for MaxNS data. To this end, we will seek to obtain a "sharpening" of the MaxNS regularized logistic regression program. In particular, we wish to find some hinge-like approximation of the MNSLR Loss Function (3.9).

In the $y = 1$ case, the approximation we seek is nothing more than

$$\omega_0 \ln\left(1 + e^{-z}\right) \approx \omega_0 \max(0, -z).$$

The $y = -1$ case is less straightforward. We shall investigate the function

$$\phi(z) \coloneqq -\ln\left(1 - \left\{\frac{1}{1 + e^{-z}}\right\}^{\omega_0}\right)$$

to find a hinge-like approximation. First, we observe that the function ϕ is monotone and positive. Next,

$$\lim_{z \to -\infty} \ln\left(1 - \left\{\frac{1}{1 + e^{-z}}\right\}^{\omega_0}\right) = 0.$$

So the function ϕ vanishes on the left. Finally, we will show that ϕ is asymptotically equal to $-\ln\omega_0 + z$ on the positive part of the plane. To this end, note that

$$\lim_{z \to \infty}\left[-\ln\left(1 - \left\{\frac{1}{1 + e^{-z}}\right\}^{\omega_0}\right) - z\right] = -\lim_{z \to \infty} \ln\left(e^z\left[1 - \left\{\frac{1}{1 + e^{-z}}\right\}^{\omega_0}\right]\right).$$

Now, let $\theta = 1 + e^{-z}$. Then, by L'Hospital's Rule

$$\lim_{z \to \infty} e^z\left[1 - \left\{\frac{1}{1 + e^{-z}}\right\}^{\omega_0}\right] = -\lim_{\theta \to 1^+} \frac{1 - \theta^{-\omega_0}}{1 - \theta}$$

$$= -\lim_{\theta \to 1^+} \frac{\omega_0 \theta^{-\omega_0 - 1}}{-1}$$

$$= \omega_0.$$

Thus,

$$\lim_{z \to \infty} \left[-\ln\left(1 - \left\{ \frac{1}{1+e^{-z}} \right\}^{\omega_0}\right) - z \right] = -\ln\omega_0.$$

In summary,

$$-\ln\left(1 - \left\{ \frac{1}{1+e^{-z}} \right\}^{\omega_0}\right) \approx \max(0, -\ln\omega_0 + z).$$

This is a hinge-like approximation of the $\phi(z)$, as desired. What is left is to add an additive constant. Let this constant be τ_{I} for $y = 1$ and τ_{II} for $y = -1$.

Hence, the loss function for the MaxNS SVM is

$$L_{\mathrm{MNSSVM}}(y, z) = \begin{cases} \omega_0 \max(0, \tau_{\mathrm{I}} - z), & \text{if } y = 1, \\ \max(0, \tau_{\mathrm{II}} - \ln\omega_0 + z), & \text{if } y = -1. \end{cases}$$

The Optimization Program

Suppose that we have been provided with MaxNS data $\left(\boldsymbol{x}_{[k]i}, y_{[k]i}\right)_{i=1}^{n}$. The optimization program for the MaxNS SVM is

$$\underset{\boldsymbol{w},b}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||_2^2 + C\sum_{i=1}^{n} L_{\mathrm{MNSSVM}}\big(y_{[k]i}, \boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big), \tag{4.47}$$

or, equivalently,

$$\underset{\boldsymbol{w},b,\boldsymbol{\xi}}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||_2^2 + C\sum_{i=1}^{n} \xi_i$$

$$\text{subject to} \quad \begin{cases} \xi_i \geq 0, & \text{for all } i \leq n, \\ \xi_i \geq \omega_0\big(\tau_{\mathrm{I}} - \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big)\big), & \text{for all } i : y_{[k]i} = 1, \\ \xi_i \geq \tau_{\mathrm{II}} - \ln\omega_0 + \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big), & \text{for all } i : y_{[k]i} = -1. \end{cases}$$

We will solve this by duality.

For brevity, set $I = \{i \leq n : y_{[k]i} = 1\}$, $II = \{i \leq n : y_{[k]i} = -1\}$. First, we obtain the Lagrangian $\mathscr{L} = \mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\delta}, \boldsymbol{\eta})$:

$$\begin{aligned}
\mathscr{L} = \frac{1}{2}||\boldsymbol{w}||_2^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\delta_i\xi_i \\
- \sum_{i\in I}\eta_i\big[\xi_i - \omega_0\big\{\tau_{\mathrm{I}} - \big(\boldsymbol{w}^\top\boldsymbol{x}_{[k]i} + b\big)\big\}\big] \\
- \sum_{i\in II}\eta_i\big[\xi_i - \big\{\tau_{\mathrm{II}} - \ln\omega_0 + \big(\boldsymbol{w}^\top\boldsymbol{x}_{[k]i} + b\big)\big\}\big] \\
= \frac{1}{2}||\boldsymbol{w}||_2^2 + \sum_{i=1}^{n}(C - \delta_i - \eta_i)\xi_i + \omega_0\tau_{\mathrm{I}}\sum_{i\in I}\eta_i + (\tau_{\mathrm{II}} - \ln\omega_0)\sum_{i\in II}\eta_i \\
- \omega_0\sum_{i\in I}\eta_i\big(\boldsymbol{w}^\top\boldsymbol{x}_{[k]i} + b\big) + \sum_{i\in II}\eta_i\big(\boldsymbol{w}^\top\boldsymbol{x}_{[k]i} + b\big).
\end{aligned}$$

To achieve stationarity, we set the derivatives of $\mathscr{L}$ with respect to $\boldsymbol{w}$, b, and ξ_i to 0. To this end,

$$\mathscr{L}_{\boldsymbol{w}} = \boldsymbol{w} - \omega_0 \sum_{i\in I} \eta_i \boldsymbol{x}_{[k]i} + \sum_{i\in II} \eta_i \boldsymbol{x}_{[k]i} = 0 \tag{4.48}$$

$$\implies \boldsymbol{w}^* = \omega_0 \sum_{i\in I} \eta_i \boldsymbol{x}_{[k]i} - \sum_{i\in II} \eta_i \boldsymbol{x}_{[k]i},$$

$$\mathscr{L}_b = -\omega_0 \sum_{i\in I} \eta_i + \sum_{i\in II} \eta_i = 0 \tag{4.49}$$

$$\implies \mathscr{L} \text{ is optimized w.r.t. } b \text{ at any point} \big(\boldsymbol{w}, b^*, \boldsymbol{\xi}, \boldsymbol{\delta}, \boldsymbol{\eta}\big) \text{ s.t. } \omega_0 \sum_{i\in I} \eta_i = \sum_{i\in II} \eta_i,$$

$$\mathscr{L}_{\xi_i} = C - \delta_i - \eta_i = 0 \tag{4.50}$$

$$\implies \mathscr{L} \text{ is optimized w.r.t. } \xi_i \text{ at any point} \big(\boldsymbol{w}, b, \boldsymbol{\xi}^*, \boldsymbol{\delta}, \boldsymbol{\eta}\big) \text{ s.t. } \delta_i = C - \eta_i.$$

Now, set

$$\alpha_i \coloneqq \begin{cases} \omega_0 \eta_i, & \text{if } y_{[k]i} = 1, \\ \eta_i, & \text{if } y_{[k]i} = -1. \end{cases}.$$

Under this notation, the Lagrangian simplifies to

$$\begin{aligned} \mathscr{L}^* &= \mathscr{L}\big(\boldsymbol{w}^*, b^*, \boldsymbol{\xi}^*, \boldsymbol{\delta}, \boldsymbol{\eta}\big) \\ &= \tau_{\mathrm{I}} \sum_{i\in I} \alpha_i + (\tau_{\mathrm{II}} - \ln\omega_0) \sum_{i\in II} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_{[k]i} y_{[k]j} \boldsymbol{x}_{[k]i}^\top \boldsymbol{x}_{[k]j}. \end{aligned}$$

We can further simplify the Lagrangian to

$$\sum_{i=1}^{n} \gamma_i \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_{[k]i} y_{[k]j} \boldsymbol{x}_{[k]i}^\top \boldsymbol{x}_{[k]j}, \tag{4.51}$$

where

$$\gamma_i \coloneqq \begin{cases} \tau_{\mathrm{I}}, & \text{if } y_{[k]i} = 1, \\ \tau_{\mathrm{II}} - \ln\omega_0, & \text{if } y_{[k]i} = -1. \end{cases}$$

It is of particular importance to note that (4.51) is a quadratic function of $\boldsymbol{\alpha}$.

We also have the KKT conditions

$$\delta_i \geq 0, \quad \text{for all } i \leq n, \tag{4.52}$$

$$\eta_i \geq 0, \quad \text{for all } i \leq n, \tag{4.53}$$

$$\delta_i \xi_i = 0, \quad \text{for all } i \leq n, . \tag{4.54}$$

$$\eta_i \big[\xi_i - \omega_0 \big\{\tau_{\mathrm{I}} - \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big)\big\}\big] = 0, \quad \text{for all } i \in \mathrm{I}, \tag{4.55}$$

$$\eta_i \big[\xi_i - \big\{\tau_{\mathrm{II}} - \ln\omega_0 + \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big)\big\}\big] = 0, \quad \text{for all } i \in \mathrm{II}. \tag{4.56}$$

Combining (4.49), (4.50), (4.52) and (4.53), we obtain that $0 \leq \eta_i \leq C$ for all $i \leq n$. Written with α_i, we say that

$$\begin{cases} 0 \leq \alpha_i \leq C\omega_0, & \text{if } i \in \mathrm{I}, \\ 0 \leq \alpha_i \leq C, & \text{if } i \in \mathrm{II}. \end{cases} \tag{4.57}$$

We can also rewrite (4.49) in terms of α_i. In particular,

$$\sum_{i=1}^{n} y_{[k]i} \alpha_i = 0. \tag{4.58}$$

If we combine (4.51), (4.57), and (4.58), we arrive at the completed dual problem:

$$\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{minimize}} \quad \sum_{i=1}^{n} \gamma_i \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_{[k]i} y_{[k]j} \boldsymbol{x}_{[k]i}^\top \boldsymbol{x}_{[k]j}$$

$$\text{subject to} \quad \begin{cases} 0 \leq \alpha_i \leq C\omega_0, & \text{if } i \in \mathrm{I}, \\ 0 \leq \alpha_i \leq C, & \text{if } i \in \mathrm{II}, \\ \sum_{i=1}^{n} y_{[k]i} \alpha_i = 0. \end{cases}$$

This is a *quadratic optimization problem.*

A solution to this problem will yield a vector $\boldsymbol{\alpha}^* \in \mathbb{R}^n$. To obtain $\boldsymbol{w}$ from this equation,

we rewrite (4.48) in terms of α_i as below:

$$\boldsymbol{w}^* = \sum_{i=1}^{n} \alpha_i^* y_{[k]i} \boldsymbol{x}_{[k]i}.$$

We may simplify this further by considering the set $\mathcal{V} = \{i \leq n : \alpha_i > 0\}$, known as the *support set*. We find that $\boldsymbol{w}^*$ may be represented in terms of elements of $\mathcal{V}$ alone:

$$\boldsymbol{w}^* = \sum_{i \in \mathcal{V}} \alpha_i^* y_{[k]i} \boldsymbol{x}_{[k]i}. \tag{4.59}$$

To obtain $b \in \mathbb{R}$, consider the set

$$\mathcal{V}^\circ \coloneqq \{i \leq n : 0 < \alpha_i < C\omega_0 \text{ for } i \in \mathrm{I},\ 0 < \alpha_i < C \text{ for } i \in \mathrm{II}\}, \tag{4.60}$$

the *interior* of $\mathcal{V}$. This is the set of all indices i such that $0 < \eta_i < C$. Note that for $i \in \mathcal{V}^\circ$, (4.50), (4.54), (4.55), and (4.56) give

$$\xi_i = 0 = \begin{cases} \omega_0\big(\tau_{\mathrm{I}} - \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big)\big), & \text{for all } \ i \in \mathrm{I}, \\ \tau_{\mathrm{II}} - \ln\omega_0 + \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big), & \text{for all } \ i \in \mathrm{II}. \end{cases}$$

We may further rearrange this to obtain that, for any $i \in \mathcal{V}^\circ$,

$$\big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big) = y_{[k]i}\gamma_i. \tag{4.61}$$

We may obtain b^* by solving (4.61) with respect to b, for any $i \in \mathcal{V}^\circ$.

We summarize this entire discussion in the following subsection.

The Optimization Program: Summarized

Suppose that, given feature observations $\boldsymbol{x} \in \mathbb{R}^d$, we wish to classify observations $y \in \{-1, 1\}$ with a support vector machine. Our training set is a MaxNS sample $\big(\boldsymbol{x}_{[k]i}, y_{[k],i}\big)_{i=1}^n$ of size

n. The parameter $\omega_0 \in \mathbb{R}$ is determined based off of set size and correlation between X and Z, the ranking variable; τ_{I}, τ_{II} are class-based bias terms; and C is a cost parameter. The optimization program for this task is

$$\underset{\boldsymbol{w},b,\boldsymbol{\xi}}{\text{minimize}} \quad \frac{1}{2}||\boldsymbol{w}||_2^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad \begin{cases} \xi_i \geq 0, & \text{for all } i \leq n, \\ \xi_i \geq \omega_0\big(\tau_{\mathrm{I}} - \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big)\big), & \text{for all } i : y_{[k]i} = 1, \\ \xi_i \geq \tau_{\mathrm{II}} - \ln\omega_0 + \big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big), & \text{for all } i : y_{[k]i} = -1, \end{cases}$$

and the dual problem is

$$\underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{minimize}} \quad \sum_{i=1}^{n}\gamma_i\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_{[k]i}y_{[k]j}\boldsymbol{x}_{[k]i}^\top \boldsymbol{x}_{[k]j}$$

$$\text{subject to} \quad \begin{cases} 0 \leq \alpha_i \leq C\omega_0, & \text{if } i \in \mathrm{I}, \\ 0 \leq \alpha_i \leq C, & \text{if } i \in \mathrm{II}, \\ \sum_{i=1}^{n} y_{[k]i}\alpha_i = 0, \end{cases}$$

where

$$\gamma_i = \begin{cases} \tau_{\mathrm{I}}, & \text{if } y_{[k]i} = 1, \\ \tau_{\mathrm{II}} - \ln\omega_0, & \text{if } y_{[k]i} = -1. \end{cases}$$

The dual is a convex optimization program, which may be solved through standard software. Once a minimizer $\hat{\boldsymbol{\alpha}}$ has been obtained, we calculate $\hat{\boldsymbol{w}}$ with the equation

$$\boldsymbol{w}^* = \sum_{i\in\mathcal{V}}\alpha_i^* y_{[k]i}\boldsymbol{x}_{[k]i}$$

and b by solving the equation

$$\big(\boldsymbol{w}^\top \boldsymbol{x}_{[k]i} + b\big) = y_{[k]i}\gamma_i$$

for any i such that

$$\begin{cases} 0 \leq \alpha_i^* \leq C\omega_0, & \text{if } y_{[k]i} = 1, \\ 0 \leq \alpha_i^* \leq C, & \text{if } y_{[k]i} = -1. \end{cases}$$

Applying the Kernel Trick

Finally, we may wish to apply the Kernel trick to this program, in order to obtain a non-linear classifier. By mirroring the discussion in Section 4.4, we arrive at the kernelized MaxNS SVC, which we will call the MNSSVM. We will jump directly into the dual problem, as it is the standard form for solving a SVM. We will assume again that we have been provided with MaxNS training data $\left(\boldsymbol{x}_{[k]i}, y_{[k],i}\right)_{i=1}^{n}$ and that our goal is to build a SVM to classify future observations. The optimization program for this task is

$$\begin{aligned} \underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{minimize}} \quad & \sum_{i=1}^{n}\gamma_i\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_{[k]i}y_{[k]j}K\left(\boldsymbol{x}_{[k]i},\boldsymbol{x}_{[k]j}\right) \\ \text{subject to} \quad & \begin{cases} 0\le\alpha_i\le C\omega_0, & \text{if } i\in \text{I}, \\ 0\le\alpha_i\le C, & \text{if } i\in \text{II}, \\ \sum_{i=1}^{n} y_{[k]i}\alpha_i = 0, \end{cases} \end{aligned}$$

where

$$\gamma_i = \begin{cases} \tau_{\text{I}}, & \text{if } y_{[k]i}=1, \\ \tau_{\text{II}} - \ln\omega_0, & \text{if } y_{[k]i}=-1. \end{cases}$$

The dual is a convex optimization program, which may be solved through standard software. Once a minimizer $\hat{\boldsymbol{\alpha}}$ has been obtained, obtain the set $\mathcal{V}^\circ$ as defined in (4.60). For any $j\in\mathcal{V}^\circ$,

$$y_j\left(\sum_{i\in\mathcal{V}}\hat{\alpha}_i y_i K(\boldsymbol{x}_i,\boldsymbol{x}_j)+b\right)=1,$$

which may be used to solve for b. Thus, we arrive at a decision boundary function

$$\hat{f}(\boldsymbol{x}) = \sum_{i\in\mathcal{V}}\hat{\alpha}_i y_i K(\boldsymbol{x}_i,\boldsymbol{x}) + \hat{b}$$

and a classification function

$$G(\boldsymbol{x}) = \text{sign}\Big\{\hat{f}(\boldsymbol{x})\Big\}.$$

We will note here that simulation studies lead us to believe that $\tau_{\mathrm{I}} = \tau_{\mathrm{II}} = 1$ are good choices. However, we have no analytic justification of this property, and thus we leave them in the model and push a deeper investigation of them to future research.

4.7 Summary

In this chapter, we introduced in detail the Support Vector Machine. An analysis of this problem through the lens of statistical learning theory revealed a deep connection with Logistic Ridge regression. Hence, we were able to exploit this link to extend our MNSLRR technique to the SVM, resulting in the MNSSVM model.

In the following section, we will conduct both simulated data analysis studies to test the efficacy of the methods developed in this Chapter and in Chapter 3.

Chapter 5

Simulated Data Analysis

In this chapter, we will perform a series of simulations to determine the efficacy of the models proposed in Chapters 3 and 4. We will make use of ROC and AUC analysis to show that the MNSLR model is superior to the standard Logistic regression model in terms of both sensitivity and specificity. By examining the error rates for the MNSSVM model, we will also see that in most cases it outperforms the standard SVM.

5.1 Simulation: Logistic Regression

We will first consider the MNSLR model proposed in Chapter 3. For the purposes of these simulations, we will assume the following parametric forms on our data: First, we will assume that our feature variables $\boldsymbol{X}$ are drawn from a Multivariate Normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ. Second, we will assume that our ranking variable Z is defined as $Z = h(\boldsymbol{X}) + \mathcal{E}$, where $h(\boldsymbol{X}) \coloneqq \boldsymbol{c}^\top \boldsymbol{X}$ for some d-dimensional vector of constants $\boldsymbol{c} \in \mathbb{R}^d$, and $\mathcal{E} \sim N(0, \sigma_\varepsilon^2)$ is an independent and Normally distributed "error term". In summary, we will assume a linear model on Z with Multivariate Normal feature variables. In particular, $\sigma_\varepsilon > 0$ will be defined to achieve a particular correlation $\rho = \mathbb{C}\text{orr}(h(\boldsymbol{X}), Z)$ between $h(\boldsymbol{X})$ and Z. Third, we will assume that $Y = 1$ if $Z \geq m$ and $Y = -1$ otherwise,

for some constant $m \in \mathbb{R}$. That is, we will assume that Y follows a threshold model with threshold m. Finally, we will set $m = m(k, \rho)$ such that the resultant MaxNS sample with set size k is expected to be balanced.

For our first battery of tests, we will assess the MNSLR technique to the standard SRS technique through so-called *ROC* (Receiver Operating Characteristic) analysis.

For a classification model, the ROC curve [73] is created by plotting the proportion of true negatives that are misclassified (also known as the *false positive rate*, which is equal to 1 minus the *specificity* of the model) against the proportion of true positives that are correctly classified (also known as the *true positive rate*, which is equal to the *sensitivity* of the model). In particular, all possible thresholds are considered for the decision boundary (which is, by default, 0.5 for a Logistic regression model), and for each possible threshold, the sensitivity and specificity of the model are determined.

By setting the decision boundary to its lowest possible value, all data points will be classified as "positive", and by setting the decision boundary to its highest possible value, all data points will be classified as "negative". As a result, the ROC curve will always pass through $(0, 0)$ and $(1, 1)$. Further, since increasing sensitivity can only decrease (or keep constant) the specificity, and vice versa, the ROC curve will always be non-decreasing. An optimal classifier will have an ROC curve that hugs the top-left of the graph, while a random classifier will have the curve $y = x$. We give an illustration of an ROC curve in Figure 5.1.

Associated with an ROC curve is a measure called AUC, or *Area Under the Curve*. The AUC of an ROC curve is nothing more than its integral. A curve that hugs the top-left will have a higher AUC and an inferior model will lead to a lower AUC. AUC analysis is important as it is a numerical technique, while the ROC is strictly graphical.

With ROC and AUC under our belt, we may now proceed to a simulation study on the MNSLR technique. In particular, we will follow the below steps:

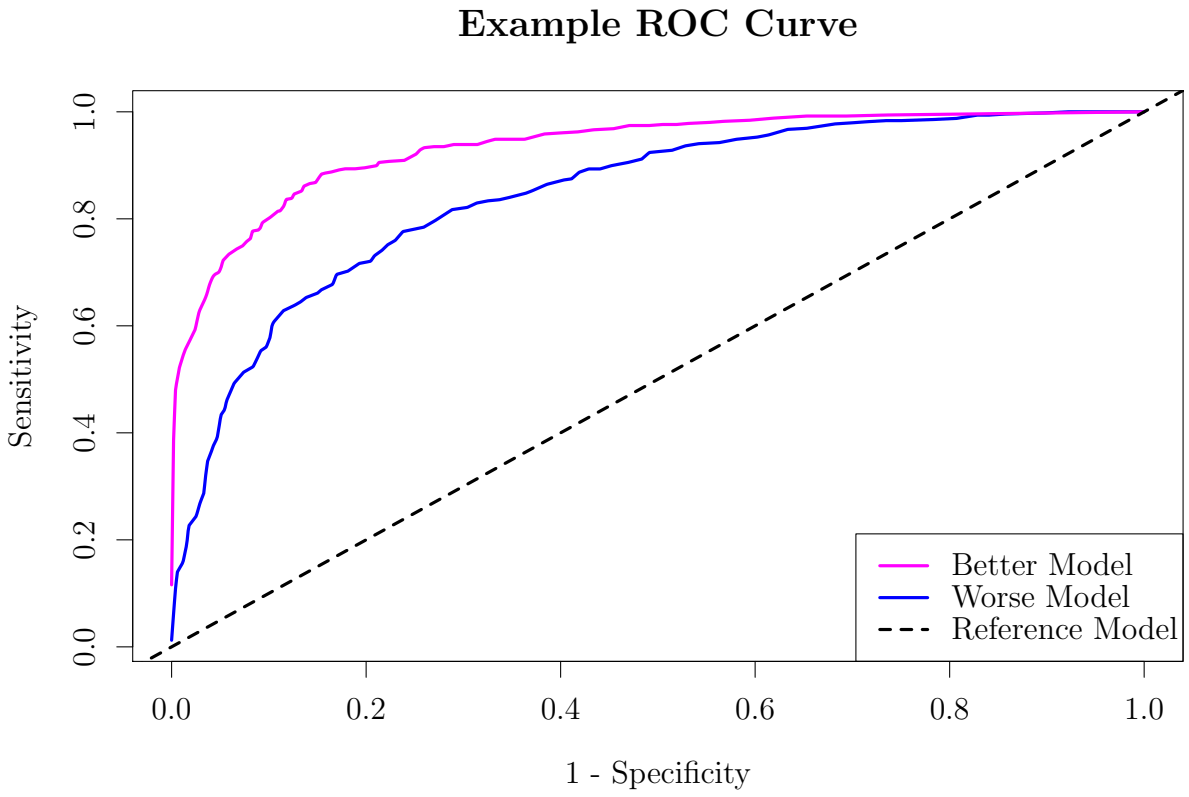

Figure 5.1: Shown here is an example of ROC analysis. The magenta curve shows a superior model, as it is closer to the top-left at all points. The dotted black line shows a reference model that would be obtained by randomly guessing at every point.

1. Establish a set size $k \in \mathbb{Z}$, a correlation $\rho \in (0, 1)$ and a feature space dimension $d \in \mathbb{Z}$. Determine a mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and a covariance matrix Σ for the multivariate normal feature space, and set a vector of coefficients $\boldsymbol{c} \in \mathbb{R}^d$.

2. Generate a population U of size $N \in \mathbb{Z}$. Specifically, draw N observations $\boldsymbol{X}$ from a d-dimensional multivariate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ. Set $\sigma_{\mathcal{E}} > 0$ so that the desired correlation $\rho = \mathbb{C}\mathrm{orr}\big(\boldsymbol{c}^\top \boldsymbol{X}, Z\big)$ is attained, and generate $\mathcal{E} \sim \mathrm{N}(0, \sigma_{\mathcal{E}}^2)$ as well as $Z = \boldsymbol{c}^\top \boldsymbol{X} + \mathcal{E}$. Set the threshold $m \in \mathbb{R}$ so that $\mathbb{P}\big(Y_{[k]} = 1\big) = \frac{1}{2}$.

3. Determine the Logistic regression model based on the population model. This is the

"true model" in the following simulations.

4. Determine a sample size $n \in \mathbb{Z}$ and a number of iterations $I \in \mathbb{Z}$. For $i = 1, 2, \ldots, I$, take MaxNS and SRS samples of size n from the population U. On each MaxNS sample generate a MNSLR model, and on each SRS sample generate a standard Logistic regression model.

5. For each MNSLR and Logistic regression model, determine the ROC curve and the AUC.

We will follow the procedure above with $I = 200$ to compare the MNSLR and LR techniques for $k = 2, 5$; $\rho = 0.75, 0.95$; $d = 2, 10$; and $n = 15, 40$. The results of these simulations are given in Figures 5.2 and 5.3

In general, we can see that the MaxNS method offers significant improvement across many parameter combinations, and in no case is it worse than the standard SRS method. We make a number of observations here and propose intuitive explanations for the results observed.

First, it appears that the improvement increases with set size k. This is expected, as the higher set sizes are used for populations with greater class imbalance. As the population displays greater class imbalance, there will be more bias within small simple random samples. Thus, the technique has more room to improve, so to speak.

Second, it appears that the improvement increases with correlation ρ. This is not surprising; a high correlation between $h(\boldsymbol{X})$ and Z means that the expert opinion contains a lot of information, and hence the usage of this information leads to a model improvement.

Next, we can see that the improvement seems to increase with the size of the feature space d. We believe this is due to the inherent complexity in higher-dimensional models. As the model is complicated, there is more error expected within a small sample size. Hence, like the relationship seen with k, there is more room for the model to improve.

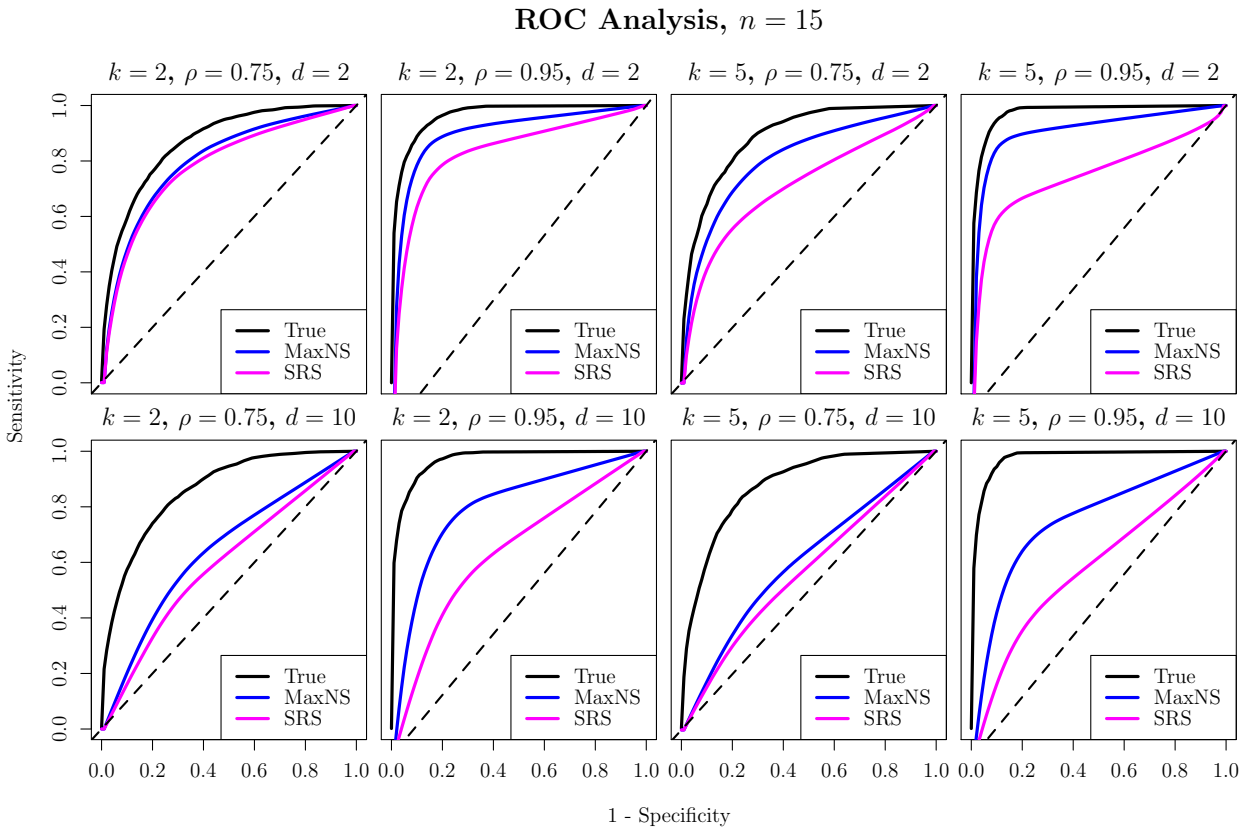

Figure 5.2: Shown here is the ROC analysis for the MNSLR model, compared to the standard SRS model, with a sample of size $n = 15$. One can see that across all tested parameter combinations, the MaxNS model is either better or no worse than the SRS model. In general, the improvement conferred seems to increase with k and ρ.

Finally, it seems like that improvement decreases with the sample size n. Again, this is not surprising. For larger sample sizes n, there is less bias within the SRS model, and hence there is less room to improve the model.

In Table (5.1) we give the AUC measurements associated with the ROC curves in Figures 5.2 and 5.3.

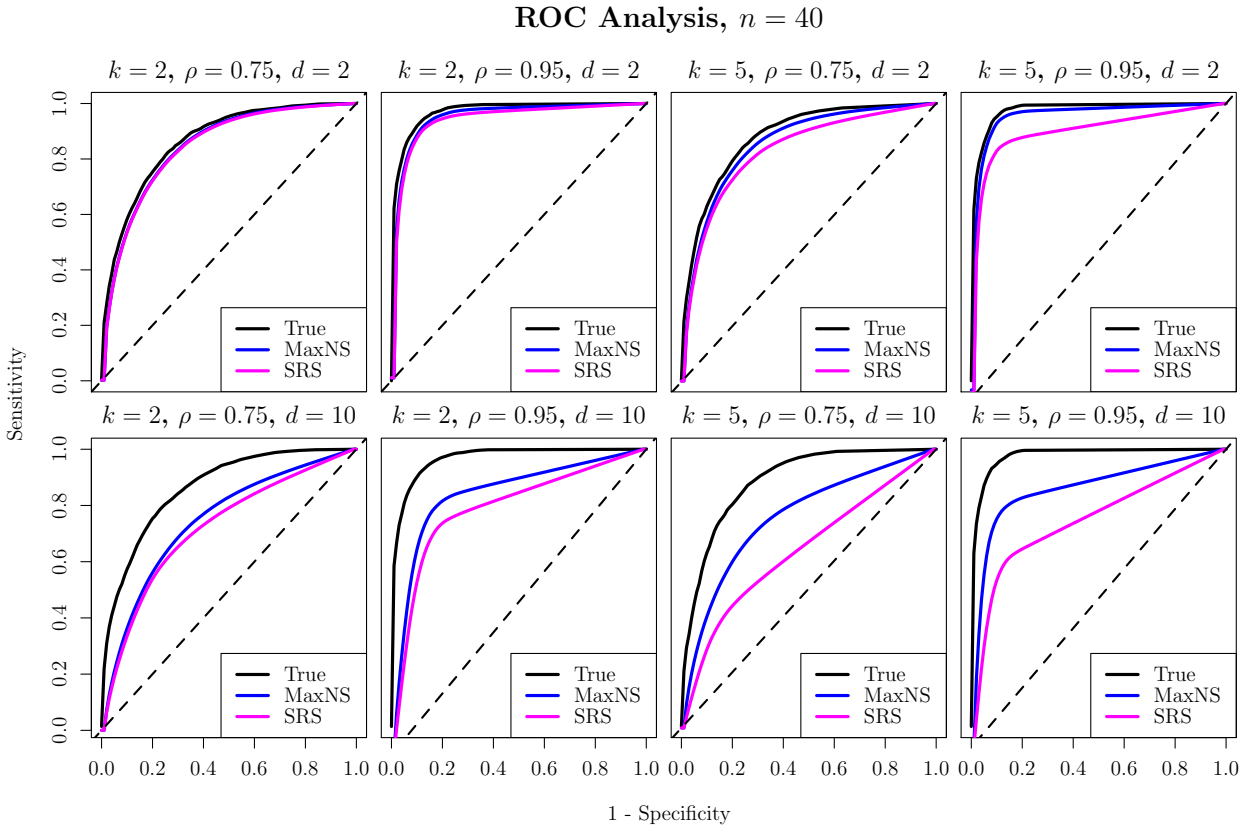

Figure 5.3: The analysis within Figure 5.2 is repeated here for a sample of size $n = 40$. The improvements conferred by the MaxNS method are lesser here, though the general patterns observed seem to persist.

	$n = 15$, $d = 2$		$n = 15$, $d = 10$		$n = 40$, $d = 2$		$n = 40$, $d = 10$	
	MaxNS	SRS	MaxNS	SRS	MaxNS	SRS	MaxNS	SRS
$k = 2$, $\rho = 0.75$	0.796	0.777	0.642	0.597	0.845	0.842	0.744	0.719
$k = 2$, $\rho = 0.95$	0.920	0.866	0.806	0.672	0.948	0.939	0.853	0.806
$k = 5$, $\rho = 0.75$	0.804	0.724	0.633	0.572	0.854	0.829	0.755	0.637
$k = 5$, $\rho = 0.95$	0.922	0.795	0.767	0.612	0.956	0.903	0.867	0.762

Table 5.1: This table gives the simulated AUC values for the parameter combinations considered in Figures 5.2 and 5.3.

5.2 Simulation: SVM

We will now turn our attention to the MNSSVM, developed in Chapter 4. We will first note that ROC analysis does not have a natural interpretation for the SVM as it does for Logistic

regression. Whereas the Logistic regression threshold is set somewhat arbitrarily at 0.5 and may be moved freely between 0 and 1 with a clear probabilistic interpretation, attempting to mirror this for the SVM is problematic.

The simplest way in which one may construct an ROC curve for a SVM is to retain the decision function $f(\boldsymbol{x})$ for each point. We traditionally use the sign of $f(\boldsymbol{x})$ as our classifier: hence, 0 is our natural thresholding point. This is the threshold that may be varied to construct an ROC curve. However, there is no simple interpretation of what any other threshold is, and in practice there is no clear way to choose any other threshold point. Thus, we do not emphasize an ROC analysis. However, for completion, a brief set of ROC curves for $n = 20$ is presented in Figure 5.4.

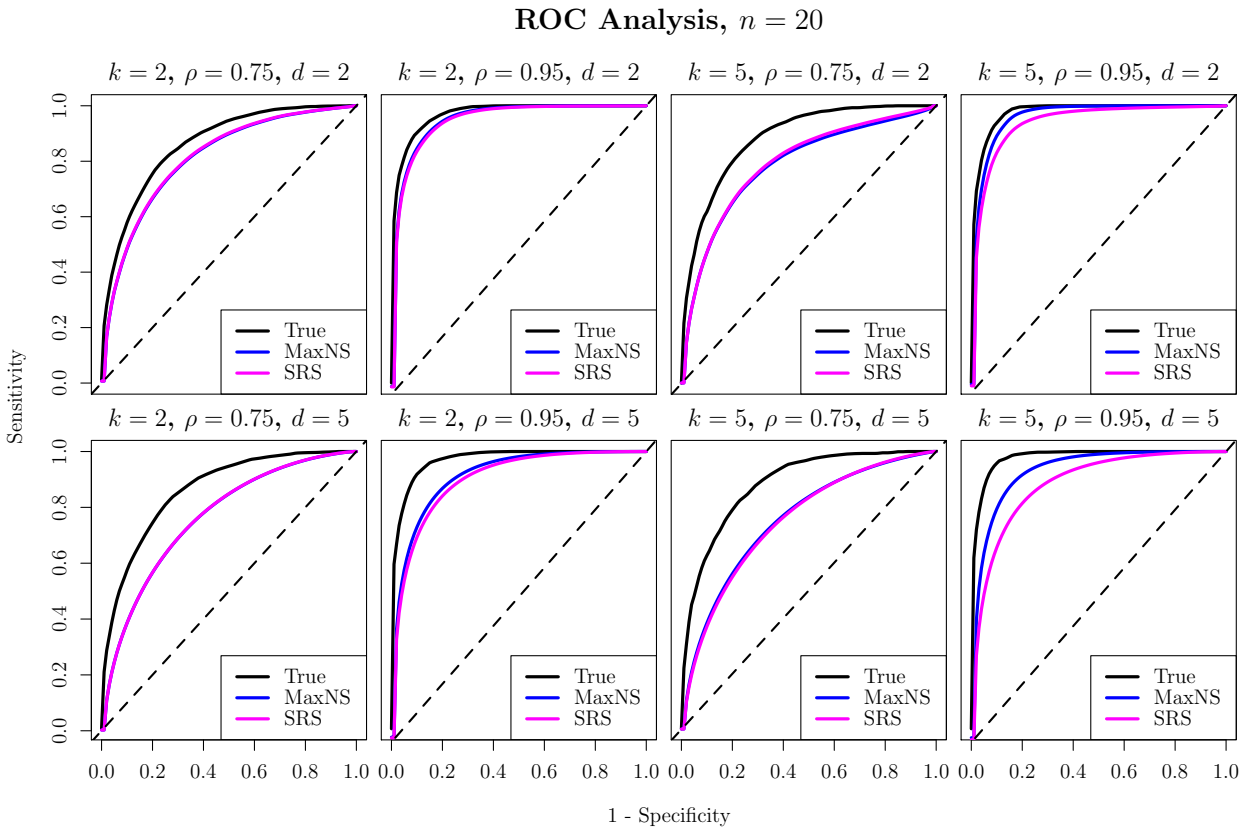

Figure 5.4: An ROC analysis comparing the MNSSVM to the standard SVM, for a sample of size $n = 20$. The MaxNS improvements conferred here are less prevalent than in the MNSLR model.

As can be seen, the improvements in the SVM in terms of ROC are much less substantial than the improvements seen for Logistic Regression. However, in general the MNSSVM does not seem to perform worse in general than the traditional SVM, and in some cases (particularly where k and ρ are high), there is some improvement.

The lack of the substantial improvement seen in Logistic regression is disappointing, but not unexpected. One primary reason that we fail to see a substantial gain is that the SVM is sparse; only a handful of points are required to construct the classifier, and the rest of the points end up playing no role whatsoever.

Thus, we will instead turn our attention to a more straightforward examination of the error rate (when classifying with a threshold of 0). In Figures 5.5 and 5.6 we compare the error rate of the MNSSVM to the traditional SVM. In particular, we examine both the overall error rate, and the error rate on the minority class. The procedure employed is highly similar to that employed when analysing the MNSLR model, except now classification errors are calculated instead of ROC curves / AUC.

As per [31], the primary point of concern when training with an imbalanced dataset is that, for small sample sizes, the classifier will bias towards the majority class. That is, the performance on the minority class suffers. This is especially unfortunate as the minority class is generally the class of interest (e.g., the "positives" in a medical study).

However, as can be seen in Figures 5.5 and 5.6, the MNSSVM provides an improved classifier without sacrificing performance on the minority class (in fact, in most cases, the performance on the minority class is improved).

One element not shown in these simulations is the frequency with which the algorithms fail to converge for SRS data. In particular, at least one observation is needed from each class to calculate a classifier, and of course even more than that are desired. With an imbalanced dataset, the likelihood of obtaining a sample with new or no observations from

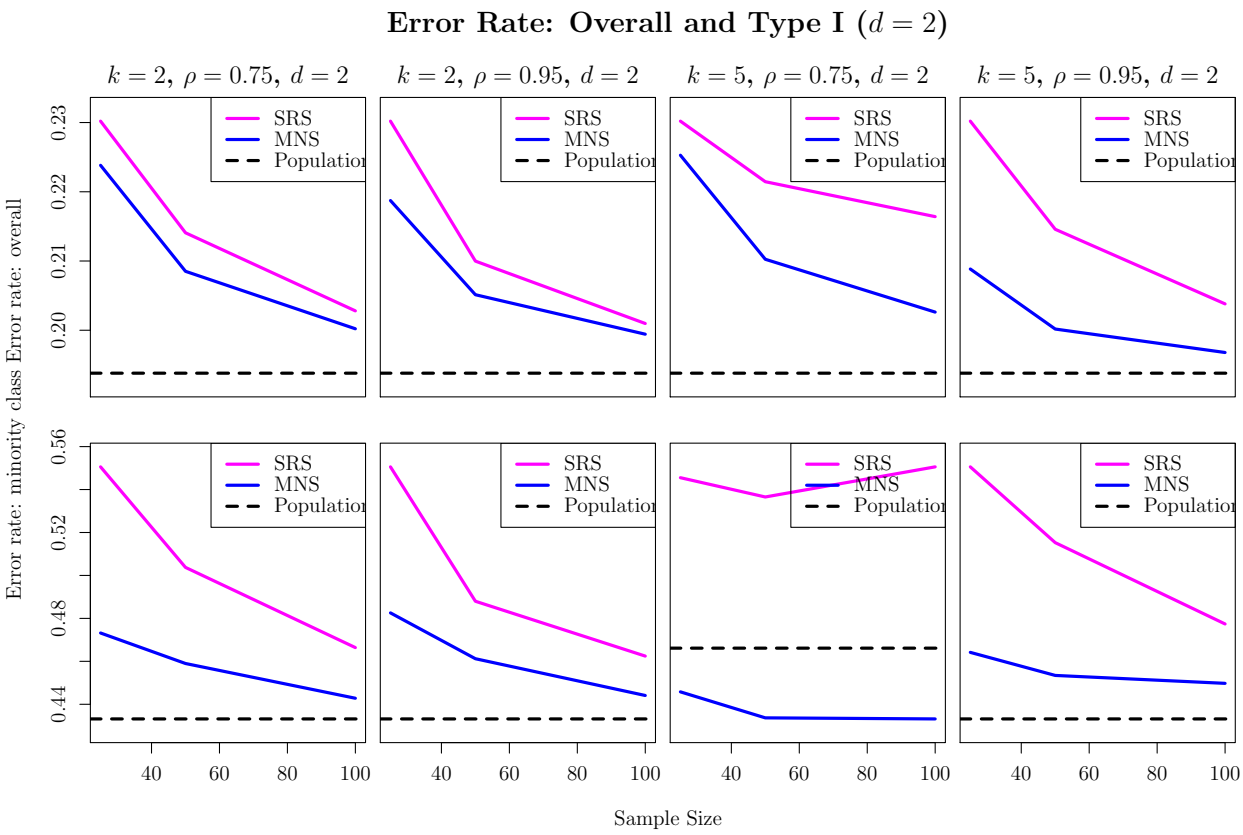

Figure 5.5: Shown here is the classification error rate for the MaxNS and standard SVM models, with the dimension of the feature space fixed at $d = 2$. The first row contains the overall classification error rates, while the second row contains the error rates on the Type I class. Once can see that in all cases, the error rate is lower both overall and on the minority class in the MNSSVM model than the standard SVM model. Note that each graph has been scaled to include all three lines, and thus the y-axes of each graph may in fact have different scales.

the minority class when taking a small sample is substantial. Thus, even in cases where the MaxNS techniques do not offer an improvement on average when compared to the standard techniques, they are still much more likely to converge.

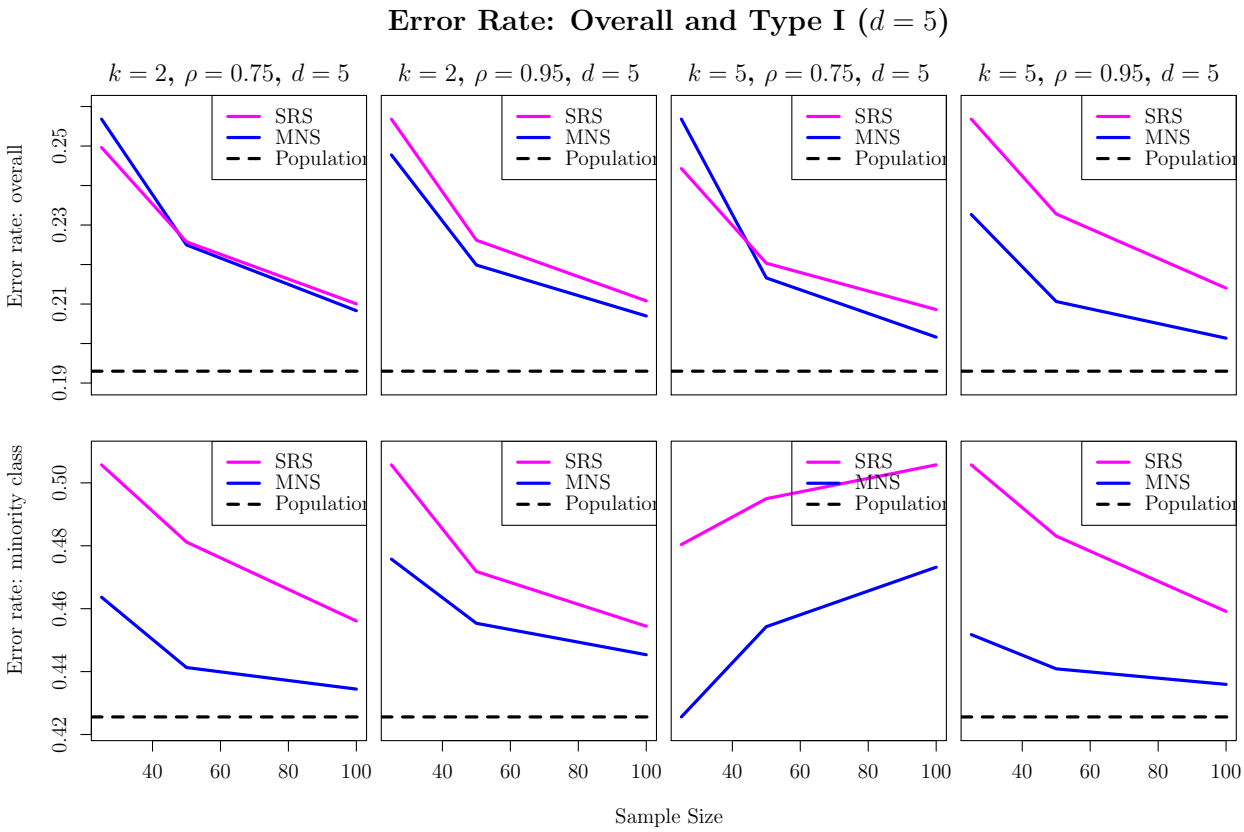

Figure 5.6: Repeated here is the analysis from Figure 5.5, with the dimension of the feature space now fixed at $d = 5$. The patterns observed in Figure 5.5 seem to persist here.

Chapter 6

Summary and Future Research

In this chapter, we provide a summary of the findings of this thesis. We then discuss the shortcomings of the work, and what work is to be done in the future to expand upon these findings.

6.1 Summary

We will summarize here the main findings of this thesis. First, in Chapter 2, we explored the relationship between the conditional class probabilities obtained within a MaxNS sampling scheme and those obtained in a SRS sampling scheme. We expressed the MaxNS conditional class probabilities in terms of the distributions of the population variables. Using this form, we derived analytically the explicit relationship between the MaxNS and SRS conditional class probabilities and compared this exact form to an approximation that exists in the literature, demonstrating an existing inconsistency in the literature.

However, the general form determined was somewhat intractable. Thus, we performed a small numerical study and made the observation that there is an approximate relationship of the form $\mathbb{P}\big(Y_{[k]} = 1 \,\boldsymbol{x}\big) \approx \mathbb{P}(Y = 1 \,|\boldsymbol{x})^{\omega_0}$, for some $0 < \omega_0 < 1$. We then performed a series of more in-depth numerical studies to show that, under certain distributional assumptions,

this approximation is strong. Finally, we conducted robustness studies to see how the approximation performs under violations of the model assumptions: we found that while it is robust to violations of the assumptions of the feature space, the assumption of normality within the error term cannot be violated without seeing poor approximation performance.

In Chapter 3, we discussed the Logistic regression technique. By examining the probabilistic properties of this technique, we were able to insert the approximation determined in Chapter 2 to develop a new model that appropriately incorporates MaxNS data. We then provided an algorithm to solve this problem numerically, and proved that the algorithm performs well. Finally, we introduced the notion of regularization through the Ridge regression technique. The Ridge Logistic regression model forms our link to SVMs.

In Chapter 4, we introduced the SVM. By examining the SVM through the lens of statistical learning theory, we obtained that the SVM and Logistic Ridge regression are in fact closely linked. By exploiting this link, we were able to extend our MaxNS Logistic Ridge regression model to the SVM to develop a MaxNS SVM model.

Finally, in Chapter 5, we performed a series of simulation studies to demonstrate the efficacy of the models developed in this thesis. We found that the MNSLR technique is notably superior to the standard Logistic regression technique when the model assumptions are met. We also found that the MNSSVM model is in many cases superior to the standard SVM model, though the gains seen for the SVM were less than that seen for Logistic regression.

6.2 Future Research

One of the main elements of this research that is lacking is a proper understanding of Result 7. This link was discovered through simulation, numerical study, and inspection. Though a significant effort was expended, we were unable to find any sort of analytic rationale for this relationship. As one result, the coefficient ω_0 in the model is something we obtain through a

purely numerical technique. We hope that obtaining an analytical justification for Result 7 will not only yield a better understanding of the bounds for this approximation, but it will allow us to derive ω_0 directly from the model.

Another shortcoming of this research is the restriction to a threshold model, with an approximately Normal transformed feature space $h(\boldsymbol{\mathcal{X}})$. While many real data scenarios will fit this model, many do not. In particular, a competing model is that the population at hand is in fact a mixture model, where observations such that $Y_i = 1$ form one class within the mixture and observations $Y_i = -1$ form another class. In terms of thresholding, it is more appropriate in this circumstance to consider a bimodal threshold variable Z. In fact, this is the more or less the model behind the standard SVM. We hope that further research can be done into this model.

In general, the models derived in this research are dependant on somewhat strong distributional assumptions. We hope that more work can be done to generalize these findings to incorporate a larger class of real data scenarios.

In the SVM model, we introduced the term τ_{I} and τ_{II}. Simulation studies lead us to believe that the optimal choice is to set both of these tuning parameters to 1. However, we have no strong justification for this. Future research will need to investigate these parameters further and determine if they have value as tuning parameters.

One large difficulty in this research is the assumption that our training data and our test data are of different forms. This prevents the use of popular tuning techniques such as cross-validation, as such techniques seek to minimize an estimate of the test error. As we cannot use our training data as surrogate test data, tuning parameters becomes difficult. One expansion to this technique would be to develop a model that is build on a mixture of MaxNS and SRS data. The SRS data could then also be used to tune the model parameters, as well as build the model itself.

Further, it is worth noting that this thesis does not contain a real data analysis, which

was cut from the project due to time constraints. Though we believe the model assumptions given are not too restrictive to be applicable in practice, it will be important to perform a real data analysis to demonstrate the value and applicability of these models.

Finally, though we decided to approach the SVM problem as a sharpening of a Logistic Ridge regression problem, this is by no means the standard approach for developing an SVM model. We believe that future research can be done that applies the MaxNS technique to the SVM model through a more direct approach.

Bibliography

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.

[2] Khalid S Khan, Patrick FW Chien, and Linga S Dwarakanath. Logistic regression models in obstetrics and gynecology literature. *Obstetrics & Gynecology*, 93(6):1014–1020, 1999.

[3] Murat Kologlu, Doruk Elker, Hasan Altun, and Iskender Sayek. Validation of mpi and pia ii in two different groups of patients with secondary peritonitis. *Hepato-gastroenterology*, 48(37):147–151, 2001.

[4] Kiarash Ghazvini, Shamsoddin Mansouri, Mohammad-Taghi Shakeri, Masoud Youssefi, Mohammad Derakhshan, and Masoud Keikha. Prediction of tuberculosis using a logistic regression model. *Reviews in Clinical Medicine*, 6(3):108–112, 2019.

[5] Raymond S Park, Sirirat Rattana-Arpa, James M Peyton, Jia Huang, Anna Kordun, Joseph P Cravero, David Zurakowski, and Pete G Kovatsis. Risk of hypoxemia by induction technique among infants and neonates undergoing pyloromyotomy. *Anesthesia & Analgesia*, 132(2):367–373, 2021.

[6] Ghita Bennouna and Mohamed Tkiouat. Scoring in microfinance: credit risk management tool–case of morocco. *Procedia computer science*, 148:522–531, 2019.

[7] Heri Kuswanto, Ayu Asfihani, Yogi Sarumaha, and Hayato Ohwada. Logistic regression

ensemble for predicting customer defection with very large sample size. *Procedia Computer Science*, 72:86–93, 2015.

[8] Sara Zahi and Boujemâa Achchab. Modeling car loan prepayment using supervised machine learning. *Procedia Computer Science*, 170:1128–1133, 2020.

[9] Pål Hartvig, Svein Alfarnes, Bjørn Østberg, Mona Skjønberg, and Tron A Moger. Brief checklists for assessing violence risk among patients discharged from acute psychiatric facilities: a preliminary study. *Nordic Journal of Psychiatry*, 60(3):243–248, 2006.

[10] John G Francis and Clive Payne. The use of the logistic model in political science: British elections, 1964-1970. *Political Methodology*, pages 233–270, 1977.

[11] Margaretha Ohyver, Jurike V Moniaga, Karina Restisa Yunidwi, and Muhamad Irfan Setiawan. Logistic regression and growth charts to determine children nutritional and stunting status: a review. *Procedia computer science*, 116:232–241, 2017.

[12] Gavin C Cawley and Nicola LC Talbot. Efficient approximate leave-one-out cross-validation for kernel logistic regression. *Machine Learning*, 71(2):243–264, 2008.

[13] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y Ng. Efficient l˜ 1 regularized logistic regression. In *Aaai*, volume 6, pages 401–408, 2006.

[14] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y Ng. Efficient l˜ 1 regularized logistic regression. In *Aaai*, volume 6, pages 401–408, 2006.

[15] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[16] O Déniz, M Castrillón, and M Hernández. Face recognition using independent component analysis and support vector machines. *Pattern Recognition Letters*, 24(13):2153–2157, 2003.

[17] Christopher Malon, Seiichi Uchida, and Masakazu Suzuki. Mathematical symbol recognition with support vector machines. *Pattern Recognition Letters*, 29(9):1326–1332, 2008.

[18] Dewi Nasien, Habibollah Haron, and Siti Yuhaniz. Support vector machine (svm) for english handwritten character recognition. *Computer Engineering and Applications, International Conference on*, 1:249–252, 01 2010.

[19] Hassan B Kazemian, Kenneth White, and Dominic Palmer-Brown. Applications of evolutionary svm to prediction of membrane alpha-helices. *Expert Systems With Applications*, 40(9):3412–3420, 2013.

[20] Giorgio Valentini, Marco Muselli, and Francesca Ruffino. Cancer recognition with bagged ensembles of support vector machines. *Neurocomputing*, 56:461–466, 2004.

[21] S Salcedo-Sanz, J. L Rojo-Álvarez, M Martínez-Ramón, and G Camps-Valls. Support vector machines in engineering: an overview. *Wiley interdisciplinary reviews. Data mining and knowledge discovery*, 4(3):234–267, 2014.

[22] Yanhui Guo, Xijie Yin, Xuechen Zhao, Dongxin Yang, and Yu Bai. Hyperspectral image classification with svm and guided filter. *EURASIP journal on wireless communications and networking*, 2019(1):1–9, 2019.

[23] Jinbo Bi and Tong Zhang. Support vector classification with input data uncertainty. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 161–168. MIT Press, 2005.

[24] Shih-Wei Lin, Kuo-Ching Ying, Shih-Chieh Chen, and Zne-Jung Lee. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert systems with applications*, 35(4):1817–1824, 2008.

[25] Jian xiong Dong, A Krzyzak, and C.Y Suen. Fast svm training algorithm with decomposition on very large data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):603–618, 2005.

[26] Xiaolin Huang, Lei Shi, and Johan A. K Suykens. Support vector machine classifier with pinball loss. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5):984–997, 2014.

[27] Glenn Fung and Olvi Mangasarian. Proximal support vector machine classifiers. KDD '01, pages 77–86. ACM, 2001.

[28] Jayadeva, R Khemchandani, and S Chandra. Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):905–910, 2007.

[29] Konstantinos Veropoulos, C. Campbell, and N. Cristianini. Controlling the sensitivity of support vector machines. *Proceedings of International Joint Conference Artificial Intelligence*, 06 1999.

[30] N. V Chawla, K. W Bowyer, L. O Hall, and W. P Kegelmeyer. Smote: Synthetic minority over-sampling technique. *The Journal of artificial intelligence research*, 16:321–357, 2002.

[31] Rukshan Batuwita and Vasile Palade. Efficient resampling methods for training support vector machines with imbalanced datasets. pages 1–8. IEEE, 2010.

[32] Yang Liu, Aijun An, and Xiangji Huang. Boosting prediction accuracy on imbalanced datasets with svm ensembles. volume 3918 of *Lecture Notes in Computer Science*, pages 107–118. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[33] Bhavani Raskutti and Adam Kowalczyk. Extreme re-balancing for svms: a case study. *ACM SIGKDD Explorations Newsletter*, 6(1):60–69, 2004.

[34] Tasadduq Imam, Kai Ming Ting, and Joarder Kamruzzaman. z-svm: An svm for improved classification of imbalanced data. volume 4304 of *Lecture Notes in Computer Science*, pages 264–273. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[35] Rukshan Batuwita and Vasile Palade. Class imbalance learning methods for support vector machines. pages 83–99. Wiley, Hoboken, NJ, USA, 1 edition, 2013.

[36] Gary King and Langche Zeng. Logistic regression in rare events data. *Political analysis*, 9(2):137–163, 2001.

[37] Hanwu Luo, Xiubao Pan, Qingshun Wang, Shasha Ye, and Ying Qian. Logistic regression and random forest for effective imbalanced classification. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 916–917. IEEE, 2019.

[38] Amirah Hazwani Abdul Rahim, Nurazlina Abdul Rashid, Asmahani Nayan, and Abd-Razak Ahmad. Smote approach to imbalanced dataset in logistic regression analysis. In *Proceedings of the Third International Conference on Computing, Mathematics and Statistics (iCMS2017)*, pages 429–433. Springer Singapore, Singapore, 2019.

[39] Thomas R Willemain. Estimating the population median by nomination sampling. *Journal of the American Statistical Association*, 75(372):908–911, 1980.

[40] Kathleen Y Wolin and Jennifer M Petrelli. *Obesity*. ABC-CLIO, 2009.

[41] John A Kanis, Eugene V McCloskey, Helena Johansson, Anders Oden, L Joseph Melton III, and Nikolai Khaltaev. A reference standard for the description of osteoporosis. *Bone*, 42(3):467–475, 2008.

[42] Mohammad Jafari Jozani, Olawale F. Ayilara, and William D. Leslie. Quantile regression with nominated samples: An application to a bone mineral density study. *Statistics in medicine*, 37(14):2267–2283, 2018.

[43] Mohammad Jafari Jozani and Sayed Jamal Mirkamali. Control charts for attributes with maxima nominated samples. *Journal of Statistical Planning and Inference*, 141(7):2386–2398, 2011.

[44] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[45] Haibo He and Yunqian Ma. *Imbalanced learning: foundations, algorithms, and applications*. Wiley-IEEE Press, 2013.

[46] Ottar Hellevik. Linear versus logistic regression when the dependent variable is a dichotomy. *Quality & Quantity*, 43(1):59–74, 2009.

[47] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[48] J Aldrich and RA Fisher. the making of maximum likelihood 1912-22. *Department of Economics, University of Southampton, United Kingdom*, 1995.

[49] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[50] Ziwei Ji and Matus Telgarsky. Risk and parameter convergence of logistic regression. *arXiv preprint arXiv:1803.07300*, 2018.

[51] Thomas P Minka. A comparison of numerical optimizers for logistic regression. *Unpublished draft*, pages 1–18, 2003.

[52] Haskell B Curry. The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics*, 2(3):258–261, 1944.

[53] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[54] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[55] Ştefan Cobzaş, Radu Miculescu, and Adriana Nicolae. *Lipschitz functions*, volume 10. Springer, 2019.

[56] Ryan Tibshirani. Convex analysis - lecture notes (chapter 6), year = 2013, publisher=Carnegie Mellon University.

[57] R. G. D. Allen and Ragnar Frisch. Statistical confluence analysis by means of complete regression systems. *The Economic Journal*, 45(180):741–742, 1935.

[58] BM Kibria, Kristofer Månsson, and Ghazi Shukur. Performance of some logistic ridge regression estimators. *Computational Economics*, 40(4):401–414, 2012.

[59] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[60] S. Cessie and J. C. Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992.

[61] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[62] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Annual Workshop on Computational Learning Theory: Proceedings of the fifth annual workshop on Computational learning theory; 27-29 July 1992*, COLT '92, pages 144–152. ACM, 1992.

[63] S Sathiya Keerthi, Dennis DeCoste, and Thorsten Joachims. A modified finite newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, 6(3), 2005.

[64] Olivier Chapelle. Training a support vector machine in the primal. *Neural computation*, 19(5):1155–1178, 2007.

[65] David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*, volume 2. Springer, 1984.

[66] Christopher J Burges and David Crisp. Uniqueness of the svm solution. *Advances in neural information processing systems*, 12, 1999.

[67] Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011.

[68] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[69] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[70] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[71] Yue Ma, Qin Zhang, Dewei Li, and Yingjie Tian. Linex support vector machine for large-scale classification. *IEEE Access*, 7:70319–70331, 2019.

[72] Xiaolin Huang, Lei Shi, and Johan AK Suykens. Support vector machine classifier with pinball loss. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):984–997, 2013.

[73] Viv Bewick, Liz Cheek, and Jonathan Ball. Statistics review 13: receiver operating characteristic curves. *Critical care*, 8(6):1–5, 2004.