UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ROBUST WEIGHTED KERNEL LOGISTIC REGRESSION IN IMBALANCED AND RARE EVENTS DATA

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

MAHER MAALOUF
Norman, Oklahoma
2009

ROBUST WEIGHTED KERNEL LOGISTIC REGRESSION IN IMBALANCED AND RARE EVENTS DATA

A DISSERTATION APPROVED FOR THE
SCHOOL OF INDUSTRIAL ENGINEERING

BY

Theodore B. Trafalis, Chair

S. Lakshmivarahan

Hillel Kumin

Binil Starly

Amy McGovern

To my father, who instilled in me the love for science
To my mother, for her measureless love, support, and contagious Greek passion

Acknowledgements

Warmest thanks I give to Dr. Theodore B. Trafalis for the freedom to determine the path of my rare-events study and for having served as my advisor. I would also like to thank Dr. S. Lakshmivarahan for all his instruction and insightful guidance. Sincerest gratitude I express to Drs. Mary Court and Sridhar Radhakrishnan for their support in my last year. I would also like to thank the rest of my committee members, Drs. Amy McGovern and Binil Starly, as well as the director of the Industrial Engineering Department, Dr. Randa Shehab, for their encouragement and support. For the critical contribution of many useful ideas and the bootstrap code, I thank Robin Gilbert. Appreciation goes to Dr. Indra Adrianto for providing the code of the DDAG method. I would also like to thank Dr. Michele Eodice, the Industrial Engineering faculties, staff, and my friend and colleague Dr. Hicham Mansouri for all their support. For taking the time to edit this dissertation, I sincerely thank my cousin, Daniel E. Karim. My utmost thanks go to Dr. Hillel Kumin, without whose support throughout most of my doctoral years this dissertation would not have been possible.

Contents

List of Tables

List of Figures

Abstract

Recent developments in computing and technology, along with the availability of large amounts of raw data, have contributed to the creation of many effective techniques and algorithms in the fields of pattern recognition and machine learning. Some of the main objectives for developing these algorithms are to identify patterns within the available data or to make predictions, or both. Great success has been achieved with many classification techniques in real-life applications. Concerning binary data classification in particular, analysis of data containing rare events or disproportionate class distributions poses a great challenge to industry and to the machine learning community. This study examines rare events (REs) with binary dependent variables containing many times more non-events (zeros) than events (ones). These variables are difficult to predict and to explain as has been demonstrated in the literature. This research combines rare events corrections on Logistic Regression (LR) with truncated-Newton methods and applies these techniques on Kernel Logistic Regression (KLR). The resulting model, Rare-Event Weighted Kernel Logistic Regression (RE-WKLR) is a combination of weighting, regularization, approximate numerical methods, kernelization, bias correction, and efficient implementation, all of which enable RE-WKLR to be at once fast, accurate, and robust.

Chapter 1

Introduction

Politics, national security, weather forecasting, medical diagnosis, image and speech recognition, and bioinformatics, are but a few of the fields in which pattern recognition and machine learning have been applied. Predictive tasks whose outcomes are quantitative (*real* numbers) are called *regression*, and tasks whose outcomes are qualitative (*binary*, *categorical*, or *discrete*) are called *classification*. The most fundamental method to address regression problems is the *least squares* method, while *logistic regression* is the fundamental method for classification. The available data from which predictive tasks are constructed are referred to as the *training data*. The resulting model performance and accuracy are assessed using data called the *testing data*.

Most of the traditional models and algorithms are based on the assumption that the classes in the data are balanced or evenly distributed. However, in many real-life applications the data is imbalanced, and when the imbalance is extreme, this problem is termed the *rare events* problem or the *imbalanced data* problem. Logistic Regression (LR), has been proven to be a powerful classifier. The advantages of using LR are that it has been extensively studied [1], and recently it has been improved through the use of truncated-Newton's methods [2, 3]. Furthermore, with regard to rare events (REs), King and Zeng [4] applied the appropriate corrections on the LR method. Kernel Logistic Regression (KLR) [5, 6], which is a kernel version of LR, can perform as good as Support Vector Machines (SVM) [7], which is considered to be the state-of-the-art method. Furthermore, like LR, KLR can provide probabilities and extend to multi-class classification problems [8, 9]. Maalouf and Trafalis [10] recently demonstrated the effectiveness of truncated Newton's method when

applied to KLR.

Research Objectives

The primary objectives of this dissertation are the following:

- To develop a general classification algorithm that is fast, efficient, and accurate when applied to non-linearly separable datasets. The proposed algorithm is termed Truncated-Regularized Kernel Logistic Regression (TR-KLR) and is based on the Truncated-Regularized Iteratively Re-weighted Least Squares (TR-IRLS) algorithm [2].

- To develop fast and robust adaptations of TR-KLR in imbalanced and rare events data. The proposed algorithm is termed Rare-Event Weighted Kernel Logistic Regression (RE-WKLR).

- To gain significantly higher accuracy in predicting rare events with diminished bias and variance.

Research Contributions

The principal contributions of this dissertation are the following:

- The TR-KLR algorithm is the result of the combination of regularization, approximate numerical methods, kernelization and efficient implementation. When evaluated against SVM and TR-IRLS, using non-linearly separable binary and multiple class datasets, TR-KLR is as accurate as, and much faster than, SVM, as well as more accurate than TR-IRLS .

- Weighting, regularization, approximate numerical methods, kernelization, bias correction, and efficient implementation are critical to enabling RE-WKLR to be an

effective and powerful method for predicting rare events. Compared to SVM and TR-KLR, using non-linearly separable small and large binary rare-events datasets, RE-WKLR is as fast as TR-KLR and much faster than SVM. In addition, RE-WKLR is statistically significantly more accurate than both SVM and TR-KLR.

Scope of the Dissertation

Chapter 2 provides a summary of the literature on the current research and publications in the areas of learning from imbalanced and rare events data. Chapter 3 gives an overview and analysis of Logistic Regression models. Chapter 4 derives the Kernel Logistic Rregression model and implements the Truncated-Regularized Kernel Logistic Regression (TR-KLR) algorithm with some numerical results. Chapter 5 describes how Kernel Logistic Regression can be used to solve rare events and data imbalance problems through the Rare-Event Weighted Kernel Logistic Regression (RE-WKLR) algorithm. Numerical results are presented in Chapter 6, and Chapter 7 addresses the conclusion and future work.

Chapter 2

Rare Events and Imbalanced Datasets Research: An Overview

Summary

Rare events data, and imbalanced or skewed datasets are very important in data mining and classification. However, these types of data are difficult to predict and to explain as has been demonstrated in the literature. The problems arise from various sources. This chapter surveys the latest research on data mining in relation to rare events and imbalanced data.

Introduction

Rare events (REs), class imbalance, and rare classes are critical to prediction and hence human response in the field of data mining and particularly data classification. Examples of rare events include fraudulent credit card transactions [11], word mispronunciation [12], tornadoes [13], telecommunication equipment failures [14], oil spills [15], international conflicts [16], state failure [17], landslides [18, 19], train derailments [20], rare events in a series of queues [21] and other rare events.

By definition, rare events are occurrences that take place with a significantly lower frequency compared to more common events. Given their infrequency, rare events have an even greater importance when correctly classified. However, the imbalanced distribution of classes calls for correct classification. The rare class presents several problems and challenges to existing classification algorithms [4, 22].

King and Zeng [4] state that the problems associated with REs stem from two main sources. First, when probabilistic statistical methods, such as LR, are used, they underes-

timate the probability of rare events, because they tend to be biased towards the majority class, which is the less important class. Second, commonly used data collection strategies are inefficient for rare events data. A trade-off exists between gathering more observations (instances) and including more informational, useful variables in the data set. When one of the classes represents a rare event, researchers tend to collect very large numbers of observations with very few explanatory variables in order to include as much data as possible of the rare class. This in turn could significantly increase the data collection cost and not help much with the underestimated probability of detecting the rare class or the rare event.

In the machine learning literature, several problems associated with REs and imbalanced data have been identified. According to Weiss [22], the most common problems associated with rare events are the following:

- *Lack of Data: Absolute Rarity.* Absolute Rarity is where the number of examples associated with the minority class is small in the absolute sense. This makes it very difficult for any classifier to detect regularities within the rare events or rare classes [22].

- *Relative Lack of Data: Relative Rarity.* Sometimes rare events or minority classes, are not rare in the absolute sense, but they are rare relative to other events, objects, or classes. This also makes it difficult to detect patterns associated with rare events or classes [23]. Consider a certain cancer data with 10,000 examples and a 100:5 between-class imbalance. The majority class examples far outnumber those of the minority class, despite the fact that 500 examples in the minority class may not be considered “rare.”

- *Class Distribution.* When datasets are divided into training and testing, most classifiers assume that the distribution of the training set is the same as the testing set. However, the training set might be imbalanced while the testing set might not, and the other way around [24, 25]. This problem is always referred to as the *sample selection*

bias [26]. When this occurs, an inductive model constructed from a biased training set may not be as accurate on an unbiased testing set as one constructed without any selection bias in the training set [27].

- *Improper Evaluation Metrics.* The most commonly used evaluation metric is *classification accuracy*, which computes the fraction of correctly classified examples (instances). The problem with this metric is its bias towards the majority class (the class with output zero) at the expense of the minority class (the class with output one) [24, 25]. Consider for example a classifier classifying a dataset with 100 instances and 100:5 imbalance between the classes. Although this classifier may miss all of the five examples of the minority class, its *accuracy* would still be 95%.

- *Inappropriate Inductive Bias.* Inductive bias can be thought of as a predisposition for one explanation rather than another [28]. In machine learning, inductive bias is essential in the sense that it makes learning more efficient by constraining the search space [29]. An example of inductive bias would be the assumption of a linear function in linear regression [30]. Another example is the effect of prior probabilities on the classification outcome. However, when dealing with rare events, the generalization bias, such as the maximum-generality bias, can have a negative impact on learning rare events, because it selects the most general set of conditions that satisfy the majority class [22, 31].

- *Small Disjuncts.* In rule-based classifiers, such as decision trees, small disjuncts are inductive set of rules that correctly classify small training examples [31]. Jo and Japkowics [32] argue that class imbalance, per se, may not be the obstacle to the performance of classifiers, but rather class imbalance leads to small disjuncts that are more prone to errors. Some of the main reasons behind the poor performance of small disjuncts are the bias [31], attribute noise, missing attributes, and the size of the training set [33, 34].

- *Data Fragmentation.* Strategies such as divide-and-conquer, which partitions the data into small groups, can lead to data fragmentation. Data fragmentation can lead to absolute lack of data within a single partition [35].

- *Noise.* Noise within datasets can have a major negative impact on the detection of rare events due to its obstruction of the rare instances. Hence, forming decision boundaries around the rare classes would be very difficult [22].

Given these most common problems associated with REs, the following is a summary of the latest techniques for handling REs and imbalanced datasets.

2.1 Evaluation Metrics

Classification *accuracy* is the most commonly used method to assess the accuracy of the classifier. However, as stated earlier, in REs, accuracy places more weight on the majority class, and hence it should not be used as a measure of accuracy in REs and imbalanced data. For binary classification in REs, the rare class is considered the positive class while the majority class is considered either class zero or the negative class. Table 2.1 shows the *confusion matrix* (CM) for binary classification. In the matrix, *true positive* (TP) corresponds to the number of correctly classified positive instances, *false negative* (FN) corresponds to the number of positive instances classified as negative, *false positive* (FP) corresponds to the number of negative instances classified as positive, and *true negative* (TN) corresponds to the number of correctly classified negative instances. Tan et al. [36] list several widely

Table 2.1: Confusion matrix for binary classification.

	Predicted Class = 1	**Predicted Class = 0**
Actual Class = 1	TP	FN
Actual Class = 0	FP	TN

used metrics which can be derived from CM. According to the matrix, *accuracy* is defined

by

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN}. \tag{2.1}$$

The counts in CM can be expressed as percentages. The *true positive rate* (TPR), or *sensitivity*, is defined as the fraction of positive examples that are correctly predicted by the model, i.e.,

$$TPR = \frac{TP}{TP+FN}, \tag{2.2}$$

while the *true negative rate* (TNR), or *specificity*, is defined as the fraction of negative examples that are correctly predicted by the model, i.e.,

$$TNR = \frac{TN}{TN+FP}. \tag{2.3}$$

The *false positive rate* (FPR) is the fraction of negative examples predicted as a positive class, i.e.,

$$FPR = \frac{FP}{TN+FP}, \tag{2.4}$$

and finally, the *false negative rate* (FNR) is the fraction of positive examples predicted as a negative class, i.e.,

$$FNR = \frac{FN}{TP+FN}. \tag{2.5}$$

Precision (P) and Recall (R) are useful for applications in which the detection of one class is more important than the detection of the other class [36]. Precision measures the fraction of the predicted positive instances that are actually correct, while Recall measures the fraction of class instances that are correctly predicted. Higher P indicates lower FP, while higher R indicates lower FN. Precision and recall are defined by

$$P = \frac{TP}{TP+FP}, \tag{2.6}$$

$$R = \frac{TP}{TP+FN}. \tag{2.7}$$

In addition, Precision and Recall can be summarized by the F_1 metric, which is defined as

$$F_1 = \frac{2 \times TP}{2 \times TP + FP + FN}, \tag{2.8}$$

and it represents a harmonic mean between P and R. Recall is equivalent to *probability of detection* (*POD*), a metric widely used in meteorology [13]. Another important meteorology metric, similar to the F_1 measure is the Critical Success Index (*CSI*) [37], which is defined as

$$CSI = \frac{TP}{TP + FP + FN}, \tag{2.9}$$

and hence this metric is not affected by the number of non-REs predictions. Furthermore, Cohen's Kappa Index (κ) is another useful metric for REs prediction evaluation [18, 38]. The index determines the agreement between the model and reality. An index value of one indicates perfect prediction and a value of zero indicates no better prediction than mere chance.

Another alternative metric, widely used in the medical field, is the Receiver Operating Characteristic (ROC) analysis and the Area Under Curve (AUC) associated with it. The ROC is a trade off between false positive rate (FPR) and true positive rate (TPR) (sensitivity). Points correspond to FPR are plotted on the x axis and points correspond to TPR are plotted on the y axis. Therefore, a good classifier is one which generates points that are located on the upper left corner of the diagram. A random guessing would be located along the main diagonal [39]. AUC does not favor one class over the other, and hence it is not biased against the rare class [22].

2.2 Algorithm Level Techniques

2.2.1 Threshold Method

Many classifiers, such as logistic regression, Naive Bayes classifiers, and Neural Networks, produce a score, or a probability, that reflects the degree to which an instance or an example belongs to a certain class. Varying the threshold of the membership degree classification could improve the classification accuracy [22, 40]. The threshold method is related to the error costs and class distribution [25]. If the error costs and class distribution are found, then setting the appropriate threshold would be a straightforward task.

2.2.2 Learn Only The Rare Class

Sometimes, when classifiers learn classification rules for all classes, the rare classes may be ignored [22]. Therefore, under certain conditions, one-class approach (the rare class) may perform better than two-class approaches [41]. Techniques such as HIPPO [42] and RIPPER [43] are examples of such an approach. HIPPO uses neural networks to learn only the rare class by recognizing patterns within that class. RIPPER selects the majority class as its default class and learns the rules for detecting the minority class. It employs a general-to-specific strategy to iteratively grow a rule and a measure to choose the best conjunct to be added to the rules. The algorithm stops when the rule starts covering the majority class examples. Therefore, RIPPER generates rules from the rarest class to the most common class.

2.2.3 Cost-Sensitive Learning

Cost-Sensitive (CS) approaches are based on the fact that the value of a correctly classified rare (positive) example exceeds that of a majority (negative) class. Hence, greater costs are assigned to *false negatives* (misses) than to *false positives* (false alarms) [44]. CS learning then seeks to minimize the number of high-cost errors and the total misclassification cost.

Another related method is MetaCost [45], which makes error-based classifiers cost sensitive. MetaCost relabels the training examples with their estimated minimal-cost classes, and then applies the error-based classifier to the modified training set. Weighted Random Forest (WRF) [46] is another CS classifier. Random Forest (RF) [47] is an ensemble of decision trees, generated from bootstrap samples of the training data using random feature selection. The WRF method assigns more weights to the minority class (higher misclassification cost), thereby penalizing misclassification of the minority class. Chen et. al [46] claim that this algorithm is useful for extremely imbalanced data.

Iterative techniques, such as *boosting* are also related to CS learning. Iterative algorithms such as AdaBoost [48] assign different weights on the training distribution in each iteration. After each iteration, boosting increases the weights associated with incorrectly classified examples and decreases the weights associated with the correctly classified ones. A variant of AdaBoost, AdaCost [49], has been shown useful in addressing the problem of rarity and imbalance in data. Analysis of boosting techniques, however, shows that boosting is tied to the choice of the base learning algorithm [50, 51]. Thus, if the base learning algorithm is a good classifier without boosting, then boosting would be useful when that base learner is used in REs.

One problem with CS methods is that specific information on cost is difficult to obtain [22, 52]. Another problem is that covering more positive examples occur at the expense of generating more false alarms [36]. As a comparison with basic sampling techniques, both Maloof [53] and Weiss [54] found that both CS learning and sampling perform equally. Weiss [54] however found that CS learning has an advantage when datasets of size larger than 10,000 examples are used.

2.2.4 Other Methods

- *More Appropriate Inductive Bias.* Several attempts have been suggested to select an inductive bias that would perform well in rare events. Maximum specificity bias and

instance-based learning algorithms are examples of such methods. The suggested methods have shown only limited success. Weiss [22] posits that this may be the result of using overall classification accuracy rather than focusing on the benefits of small disjuncts.

- *Non-Greedy Search Techniques.* Genetic algorithms are increasingly used in data mining because of their ability to skip local minima in searching for the global minimum [14, 55]. Another method is the association-rule mining system which is able to find rare associations.

- *Utilize Knowledge/Human Interaction.* Knowledge and human interaction help improving the data mining process, especially for very difficult problems. In many rare event problems, decisions involve qualitative assessments and judgment. This includes a better description of examples, addition of more useful features, and discovery of results that warrant further investigation. Predicting international conflicts [16] is an example of data mining which requires both quantitative and qualitative assessments.

- *Two Phase Rule Induction.* Sometimes it is difficult to maximize both precision and recall. PNRule algorithm [56] uses two-phase rule induction and focuses on precision and recall separately. The first phase focuses on *recall* by inducing rules with high accuracy. The second phase focuses on *precision* through rules that remove false positives from the records covered by the first phase.

- *Biased Minimax Probability Machine (BMPM).* The Minimax Probability Machine (MPM) [57] is a novel classifier which estimates the worst-case bound on the probability of misclassification of future data points. The BMPM, proposed by Huang et al. [58], can control the decision hyperplane in favor of the more important class. However, the means and the covariance matrices have to be reliably estimated for good accuracy.

2.3 Data Level Techniques

2.3.1 Feature Selection

Zheng et al. [59] argue that existing feature selection measures are not appropriate for imbalanced data. The authors propose a feature selection framework, which selects features for positive and negative classes separately, and then explicitly combines them. The results show improvement on the performance of both Naive Bayes and regularized Logistic Regression methods.

2.3.2 Sampling

Sampling is undoubtedly one of the most important techniques in dealing with REs. The underlying concept behind sampling is minimizing the effect of rareness by changing the distribution of the training examples. Sampling techniques consist of basic sampling and advanced sampling. Van Hulse [60] provides a comprehensive survey on both random and intelligent data sampling techniques and their impact on various classification algorithms. Seiffert et al. [61] observed that data sampling is very effective in alleviating the problems presented by rare events.

Basic Sampling Methods

Basic sampling methods consist of *under-sampling* and *over-sampling* [36]. Under-sampling balances the training set by eliminating examples from the majority class. This strategy risks degrading the performance of the classifier because the examples eliminated may contain useful information. Over-sampling creates identical examples of the minority class in order to make the training set more balanced. Over-sampling thus can increase the computational time. In addition, over-sampling risks over-fitting, since it involves making identical copies of the minority class. Drummond and Holte [62] found that under-sampling using C4.5 (a decision tree algorithm) is most effective for imbalanced data.

Maloof [53] showed, however, that under-sampling and over-sampling are almost equivalent using Naive Bayes and C5.0 (a commercial successor to C4.5). Japkowicz [63] also came to similar conclusion, but found that under-sampling the majority class works better on large domains. Prati et al. [64] proposed over-sampling combined with data cleaning methods as a possible remedy, but without providing conclusive evidence. Weiss [54] found that there is no clear winner between under-sampling and over-sampling, and whether one should be chosen over the other is highly dependent on the dataset. King and Zeng [4] advocate under-sampling of the majority class when statistical methods such as logistic regression are employed, based on the dependent variable for handling rare events data. However, they state that such designs are only consistent and efficient with the appropriate corrections.

Advanced Sampling Techniques

Advanced Sampling Techniques use intelligence when adding or removing examples, or when they combine under-sampling and over-sampling. Barandela et al. [65] and Han et al. [66] examined the performance of intelligent sampling techniques, such as the Synthetic Minority Over-Sampling TEchnique (SMOTE) and Borderline-SMOTE. The SMOTE algorithm adds non-replicated minority-class examples from the line segments that join the *k* minority-class neighbors. Thus, while SMOTE is an over-sampling algorithm, it avoids the problem of over-fitting. However, in the presence of class overlap, SMOTE does not perform better than data editing techniques that focus on removing noisy instances and atypical patterns from the majority class [67]. Borderline-SMOTE over samples only the minority class instances that are near the borderline between the classes [66]. SMOTE-Boost [68] is another algorithm that uses boosting. The algorithm alters the distribution of the training data by adding new minority-class examples using SMOTE algorithm. Active learning techniques have been recently implemented in class imbalance data and show promising results. Ertekin et al. [69] proposed an efficient active learning method which

selects informative instances from the training dataset instead of using the entire training dataset. Such a strategy is useful for very large datasets.

Another strategy, suggested by Kubat and Matwin [70], is One-Sided Selection (OSS) which under-samples by removing majority class examples that are considered redundant or noisy. Laurikkala [71] argues that Hart's Condensed Nearest Neighbor (CNN) rule, used by OSS, is sensitive to noise, and proposed the Neighborhood Cleaning rule (NCL), which emphasizes more on data cleaning rather than data reduction. Another method, which involves hierarchical classification, is proposed by Li et al. [72]. The method consists of two stages. The first stage identifies most of the majority class examples and eliminates them. The second stage discriminates between the minority class and the greatly reduced majority class examples lying near the decision boundary.

Balanced Random Forest (BRF) [46] combines under-sampling with ensemble learning by artificially altering the class distribution, thus representing classes more equally in each tree with more computation efficiency. Another method, Cluster-Based Oversampling (CBO) [32] is shown to be effective in handling both class imbalance and small disjuncts simultaneously. The CBO algorithm utilizes re-sampling through clustering the training data of each class separately then performing random over-sampling, cluster by cluster. The advantage of this method is that it considers both between-class imbalance and within-class imbalance, then over-sampling the data to rectify these imbalances simultaneously. Another cluster-based algorithm, Classification using lOcal clusterinG (COG), recently proposed by Wu et al. [73] is also proving effective in rare events data, when applied using Support Vector Machines (SVM). The idea is to use clustering within each class to generate linearly separable balanced subclasses.

2.4 Kernel-Based Methods

In recent years, interest in kernel-based methods has been growing because they provide state-of-art techniques for many applications. Most of these kernel-based methods, how-

ever, are presented in the literature along the SVM method. SVM minimizes the total error while maximizing the margin between the support vectors (SV) and the separating hyperplane [7]. However, highly imbalanced data degrade the performance of SVM [74]. This degradation stems from various sources. Since SVM tries to minimize the total error, it is then biased towards the majority class, because the SV of the minority class may be positioned far from the separating hyperplane [74, 75]. Furthermore, in the case of absolute rarity, the number of SV of the minority class is simply inadequate to guarantee good classification performance [76].

A number of methods has been proposed in the literature to remedy this SVM problem. Akbani et al. [74] incorporated SMOTE with Different Costs (SDC) into SVM. The SDC algorithm uses different error costs for different classes in order to move the boundary away from the minority class. In addition SDC uses SMOTE to make the minority class instances more densely distributed, hence allowing the boundary to be more well defined. SVM ensembles method [77] is another example of the use of advanced sampling in SVM. The SVM ensembles method works by decomposing the majority class examples into K subsets, depending on the number of the minority class examples. All of the examples of the minority class are then combined with each subset from the majority class. Next, SVM is trained independently on each of these subsets. Finally, a majority vote is used on the combined SVM results. The advantage of SVM ensembles is that it preserves all of the examples that belong to the majority class without any loss of data. The only disadvantage is its assumption that a good class distribution is known. This estimation, however, increases the learning time.

Another example is the Granular Support Vector Machines-Repetitive Under-sampling (GSVM-RU) algorithm [78], which combines classification with under-sampling methods. The GSVM-RU algorithm is based on the Granular Support Vector Machines (GSVM) algorithm which combines the principles from statistical learning theory and the granular computing theory in a systematic and formal way [79]. The GSVM improves classifica-

tion effectiveness by establishing a trade-off between local significance of a subset of data and global correlation among different subsets of data. GSVM also improves efficiency by eliminating redundant data locally through parallel computation. The GSVM-RU method directly uses SVM for under-sampling. First, GSVM-RU retains all of the minority class examples and forms a positive information granule. Second, the majority class examples which are SV form a negative information granule, consisting of Negative Local Support Vectors (NLSVs). Then these NLSVs are extracted from the original training data, and combined with the positive granule, to form a smaller training dataset. This process is repeated several times until multiple negative information granules are formed. After that, the remaining majority examples in the original training set are simply discarded. An aggregation operation is then performed to selectively aggregate the examples in the negative granules with all the positive examples. Finally, SVM is modeled in the aggregate dataset for classification.

2.5 Conclusion

This chapter provides a summary of the literature on the most important investigations in the areas of learning from imbalanced and rare events data. Problems related to imbalance or REs result from many factors. The size of the dataset, the distribution of classes, data duplication, the choice of classifier, and class overlap are all relevant to the end products of classification. As Gu et al. [80] mentioned, deeper understanding of the basics would help in designing better methods for dealing with the problem of learning with imbalanced and REs data.

Chapter 3

Logistic Regression: An overview

Summary

Logistic Regression (LR) is one of the most important statistical procedures for the analysis of binary and proportional response data. This chapter presents a review of the LR method along with the recent developments on both the algorithmic level and on dealing with REs and imbalanced data.

3.1 Logistic Regression

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be a data matrix where n is the number of instances (examples) and d is the number of features (parameters or attributes), and $\mathbf{y}$ be a binary outcomes vector. For every instance $\mathbf{x}_i \in \mathbb{R}^d$ (a row vector in $\mathbf{X}$), where $i = 1 \ldots n$, the outcome is either $y_i = 1$ or $y_i = 0$. Let the instances with outcomes of $y_i = 1$ belong to the positive class (occurrence of an event), and the instances with outcomes $y_i = 0$ belong to the negative class (non-occurrence of an event). The goal is to classify the instance $\mathbf{x}_i$ as positive or negative. An instance can be thought of as a Bernoulli trial (the *random component*) with an expected value $E(y_i)$ or probability p_i.

A linear model to describe such a problem would have the matrix form

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \tag{3.1}$$

where ε is the error vector, and where

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix}, \text{and } \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}. \tag{3.2}$$

The vector β is the vector of unknown parameters such that $\mathbf{x}_i \leftarrow [1, \mathbf{x}_i]$ and $\beta \leftarrow [\beta_0, \beta^{\mathrm{T}}]$. From now on, the assumption is that the intercept is included in the vector β. Now, since $\mathbf{y}$ is a Bernoulli random variable with a probability distribution

$$P(y_i) = \begin{cases} p_i, & \text{if } y_i = 1; \\ 1 - p_i, & \text{if } y_i = 0; \end{cases} \tag{3.3}$$

then the expected value of the response is

$$E(y_i) = 1(p_i) + 0(1 - p_i) = p_i = \mathbf{x}_i\beta, \tag{3.4}$$

with a variance

$$V(y_i) = p_i(1 - p_i). \tag{3.5}$$

It follows from the linear model

$$y_i = \mathbf{x}_i\beta + \varepsilon_i \tag{3.6}$$

that

$$\varepsilon_i = \begin{cases} 1 - p_i, & \text{if } y_i = 1 \text{ with probability } p_i; \\ -p_i, & \text{if } y_i = 0 \text{ with probability } 1 - p_i; \end{cases} \tag{3.7}$$

Therefore, ε_i has a distribution with an expected value

$$E(\varepsilon_i) = (1 - p_i)(p_i + (-p_i)(1 - p_i) = 0, \tag{3.8}$$

and a variance

$$V(\varepsilon_i) = E(\varepsilon_i^2) - E(\varepsilon_i) = (1-p_i)^2(p_i) + (-p_i)^2(1-p_i) - (0) \tag{3.9}$$

$$= p_i(1-p_i). \tag{3.10}$$

Since the expected value and variance of both the response and the error are not constant (heteroskedastic), and the errors are not normally distributed, the least squares approach cannot be applied. In addition, since $y_i \in \{0,1\}$, linear regression would lead to values above one or below zero. Thus, when the response vector is binary, the logistic response function, as shown in Figure 3.1, is the appropriate one.

The logistic function commonly used to model each positive instance $\mathbf{x}_i$ with its expected binary outcome is given by

$$E(y_i = 1|\mathbf{x}_i, \beta) = p_i = \frac{e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} = \frac{1}{1+e^{-\mathbf{x}_i\beta}}, \qquad \text{for } i = 1,\ldots n. \tag{3.11}$$

The logistic (logit) transformation is the logarithm of the odds of the positive response, and is defined as

$$\eta_i = g(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \mathbf{x}_i\beta. \tag{3.12}$$

In matrix form, the logit function is expressed as

$$\eta = \mathbf{X}\beta. \tag{3.13}$$

The logit transformation function is important in the sense that it is linear and hence it has many of the properties of the linear regression model. In LR, this function is also called the *canonical link function*, which relates the linear predictor η_i to $E(y_i) = p_i$ through $g(p_i)$. In other words, the function $g(.)$ links $E(y_i)$ to $\mathbf{x}_i$ through the linear combination of $\mathbf{x}_i$ and β (the *systematic component*). Furthermore, the logit function implicitly places a separating hyperplane, $\beta_0 + \langle \mathbf{x}, \beta \rangle = 0$, in the input space between the positive and non-

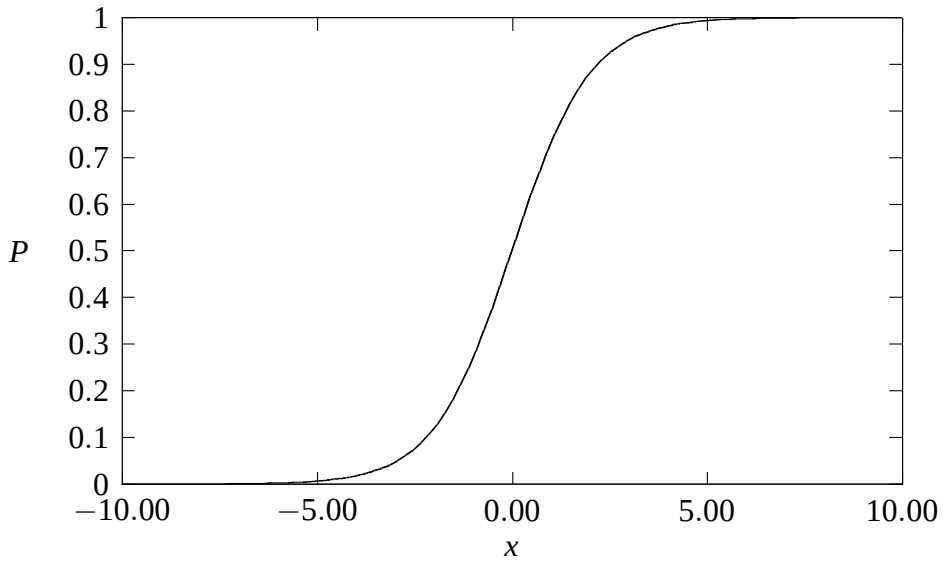

Figure 3.1: Logistic Response Function

positive instances.

The most widely used general method of estimation is the method of *maximum likelihood* (ML) (see Appendix A). The ML method is based on the joint probability density of the observed data, and acts as a function of the unknown parameters in the model [81].

Now, with the assumption that the observations are independent, the likelihood function is

$$\mathbb{L}(\beta) = \prod_{i=1}^{n} (p_i)^{y_i} (1-p_i)^{1-y_i} = \prod_{i=1}^{n} \left(\frac{e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right)^{y_i} \left(\frac{1}{1+e^{\mathbf{x}_i\beta}} \right)^{1-y_i}, \tag{3.14}$$

and hence, the log-likelihood is then

$$\ln \mathbb{L}(\beta) = \sum_{i=1}^{n} \left(y_i \ln \left(\frac{e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right) + (1-y_i) \ln \left(\frac{1}{1+e^{\mathbf{x}_i\beta}} \right) \right). \tag{3.15}$$

Amemiya [82] provides formal proofs that the Maximum Likelihood (ML) estimator for LR satisfies the ML estimators' desirable properties (see Appendix A). Unfortunately, there is no closed form solution to maximize $\ln \mathbb{L}(\beta)$ with respect to β. The LR *maximum likelihood estimates* (MLE) are therefore obtained using numerical optimization methods, which start with a guess and iterate to improve on that guess. One of the most commonly used

numerical methods is the Newton-Raphson method, for which, both the gradient vector and the Hessian matrix are needed:

$$\frac{\partial}{\partial \beta_j} \ln \mathbb{L}(\beta) = \sum_{i=1}^{n} \left(y_i \left(\frac{x_{ij}}{1+e^{\mathbf{x}_i\beta}} \right) + (1-y_i) \left(\frac{-x_{ij}e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right) \right) \tag{3.16}$$

$$= \sum_{i=1}^{n} \left(y_i x_{ij} \left(\frac{1}{1+e^{\mathbf{x}_i\beta}} \right) - (1-y_i) x_{ij} \left(\frac{e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right) \right) \tag{3.17}$$

$$= \sum_{i=1}^{n} (y_i x_{ij}(1-p_i) - (1-y_i) x_{ij}(p_i)) \tag{3.18}$$

$$= \sum_{i=1}^{n} (x_{ij}(y_i - p_i)) = 0, \tag{3.19}$$

where $j = 0, \ldots d$ and d is the number of parameters. Each of the partial derivatives is then set to zero. In matrix form, equation (3.19) is written as

$$g(\beta) = \nabla_\beta \ln \mathbb{L}(\beta) = \mathbf{X}^{\mathrm{T}}(\mathbf{y} - \mathbf{p}) = \mathbf{0}. \tag{3.20}$$

Now, the second derivatives with respect to β are given by

$$\frac{\partial^2}{\partial \beta_j \partial \beta_k} \ln \mathbb{L}(\beta) = \sum_{i=1}^{n} \left(\frac{-x_{ij}x_{ik}e^{\mathbf{x}_i\beta}}{(1+e^{\mathbf{x}_i\beta})(1+e^{\mathbf{x}_i\beta})} \right) \tag{3.21}$$

$$= \sum_{i=1}^{n} (-x_{ij}x_{ik}(p_i(1-p_i))). \tag{3.22}$$

If v_i is defined as $p_i(1-p_i)$ and $\mathbf{V} = diag(v_1, \ldots v_n)$ then the Hessian matrix can be expressed as

$$\mathbf{H}(\beta) = \nabla^2_\beta \ln \mathbb{L}(\beta) = -\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X}. \tag{3.23}$$

Since the Hessian matrix is negative definite, then the objective function is strictly concave, with one global maximum. The LR *information matrix* is given by

$$\mathbf{I}(\beta) = -E[\mathbf{H}(\beta)] = \mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X}. \tag{3.24}$$

The variance of β is then $\mathbf{V}(\beta) = \mathbf{I}(\beta)^{-1} = (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X})^{-1}$.

Over-fitting the training data may arise in LR [1], especially when the data are very high dimensional and/or sparse. One of the approaches to reduce over-fitting is through quadratic *regularization*, known also as *ridge regression*, which introduces a penalty for large values of β and to obtain better generalization [83]. The regularized log-likelihood can be defined as

$$\begin{aligned} \ln \mathbb{L}(\beta) &= \sum_{i=1}^{n} \left(y_i \ln \left(\frac{e^{\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right) + (1-y_i) \ln \left(\frac{1}{1+e^{\mathbf{x}_i\beta}} \right) \right) - \frac{\lambda}{2}||\beta||^2 \quad (3.25) \\ &= \sum_{i=1}^{n} \ln \left(\frac{e^{y_i\mathbf{x}_i\beta}}{1+e^{\mathbf{x}_i\beta}} \right) - \frac{\lambda}{2}||\beta||^2, \quad (3.26) \end{aligned}$$

where $\lambda > 0$ is the regularization parameter and $\frac{\lambda}{2}||\beta||^2$ is the regularization (penalty) term. For binary outputs, the loss function or the deviance (DEV) is the negative log-likelihood and is given by the formula [1, 2]

$$\mathbb{DEV}(\hat{\beta}) = -2 \ln \mathbb{L}(\beta). \quad (3.27)$$

Minimizing the deviance $\mathbb{DEV}(\hat{\beta})$ given in (3.27) is equivalent to maximizing the log-likelihood [1]. Recent studies showed that the *conjugate gradient* (CG) method, when applied to the method of *iteratively re-weighted least squares* (IRLS) provides better results to estimate β than any other numerical method [84, 85].

3.2 Iteratively Re-weighted Least Squares

One of the most popular techniques used to find the MLE of β is the iteratively re-weighted least squares (IRLS) method, which uses Newton-Raphson algorithm to solve LR score equations. Each iteration finds the *weighted least squares* (WLS) estimates for a given set of weights, which are used to construct a new set of weights [81]. The gradient and the

Hessian are obtained by differentiating the regularized likelihood in (3.26) with respect to β, obtaining, in matrix form

$$\nabla_\beta \ln \mathbb{L}(\beta) = \mathbf{X}^{\mathrm{T}}(\mathbf{y}-\mathbf{p}) - \lambda\beta = \mathbf{0}, \tag{3.28}$$

$$\nabla^2_\beta \ln \mathbb{L}(\beta) = -\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} - \lambda\mathbf{I}, \tag{3.29}$$

where $\mathbf{I}$ is a $d \times d$ identity matrix. Now that the first and second derivatives are obtained, the Newton-Raphson update formula on the $(c+1)-th$ iteration is given by

$$\hat{\beta}^{(c+1)} = \hat{\beta}^{(c)} + (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^{\mathrm{T}}(\mathbf{y}-\mathbf{p}) - \lambda\hat{\beta}^{(c)}). \tag{3.30}$$

Since $\hat{\beta}^{(c)} = (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})\hat{\beta}^{(c)}$, then (3.30) can be rewritten as

$$\hat{\beta}^{(c+1)} = (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}(\mathbf{V}\mathbf{X}\hat{\beta}^{(c)} + (\mathbf{y}-\mathbf{p})) \tag{3.31}$$

$$= (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}, \tag{3.32}$$

where $\mathbf{z}^{(c)} = \mathbf{X}\hat{\beta}^{(c)} + \mathbf{V}^{-1}(\mathbf{y}-\mathbf{p})$ and is referred to as the adjusted response [8].

Despite the advantage of the regularization parameter, λ, in forcing positive definiteness, if the matrix $(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})$ were dense, the iterative computation could become unacceptably slow [2]. This necessitates the need for a "trade off" between convergence speed and accurate Newton direction [86]. The method which provides such a trade-off is known as the truncated Newton's method.

3.2.1 TR-IRLS Algorithm

The WLS subproblem,

$$(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{I})\hat{\beta}^{(c+1)} = \mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}, \tag{3.33}$$

is a linear system of d equations and variables, and solving it is equivalent to minimizing the quadratic function $\frac{1}{2}\hat{\beta}^{(c+1)}(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X}+\lambda\mathbf{I})\hat{\beta}^{(c+1)}-\hat{\beta}^{(c+1)}(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)})$. Komarek and Moore [87] were the first to implement a modified linear CG to approximate the Newton direction in solving the IRLS for LR. This technique is called *truncated-regularized iteratively-reweighted least squares* (TR-IRLS). The main advantage of the CG method is that it guarantees convergence in at most d steps [86]. The TR-IRLS algorithm consists of two loops. Algorithm 1 represents the outer loop which finds the solution to the WLS problem and is terminated when the relative difference of deviance between two consecutive iterations is no larger than a specified threshold ε_1. Algorithm 2 represents the inner loop, which solves the WLS subproblems in Algorithm 1 through the linear CG method, which is the Newton direction. Algorithm 2 is terminated when the residual

$$\mathbf{r}^{(c+1)}=(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X}+\lambda\mathbf{I})\hat{\beta}^{(c+1)}-\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}$$

is no greater than a specified threshold ε_2. For more details on the TR-IRLS algorithm and implementation, see Komarek [2].

Algorithm 1: LR MLE using IRLS

Data: $\mathbf{X}, \mathbf{y}, \hat{\beta}^{(0)}$
Result: $\hat{\beta}$
begin
  $c = 0$
  while $|\frac{DEV^{(c)}-DEV^{(c+1)}}{DEV^{(c+1)}}| > \varepsilon_1$ **and** $c \leq$ *Max IRLS Iterations* **do**
    for $i \leftarrow 1$ **to** n **do**
      $\hat{p}_i = \frac{1}{1+e^{-\mathbf{x}_i\hat{\beta}}}$; /* Compute probabilities */
      $v_i = \hat{p}_i(1-\hat{p}_i)$; /* Compute weights */
      $z_i = \mathbf{x}_i\hat{\beta}^{(c)} + \frac{(y_i-\hat{p}_i)}{\hat{p}_i(1-\hat{p}_i)}$; /* Compute the adjusted response */
    $\mathbf{V} = diag(v_1, ..., v_n)$
    $(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X}+\lambda\mathbf{X})\hat{\beta}^{(c+1)} = \mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}$; /* Compute $\hat{\beta}$ via WLS */
    $c = c+1$
end

Default parameter values are given for both algorithms [87] and are shown to provide

Algorithm 2: Linear CG. $\mathbf{A} = \mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X} + \lambda\mathbf{X}$, $\mathbf{b} = \mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{z}$

Data: $\mathbf{A}, \mathbf{b}, \hat{\beta}^{(0)}$
Result: $\hat{\beta}$ such that $\mathbf{A}\hat{\beta} = \mathbf{b}$

begin
  $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\hat{\beta}^{(0)}$; `/* Initialize the residual */`
  $c = 0$
  while $||\mathbf{r}^{(c+1)}||^2 > \varepsilon_2$ **and** $c \leq$ *Max CG Iterations* **do**
    if $c = 0$ **then**
      $\zeta^{(c)} = 0$
    else
      $\zeta^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c+1)}}{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c)}}$; `/* Update A-Conjugacy enforcer */`
    $\mathbf{d}^{(c+1)} = \mathbf{r}^{(c+1)} + \zeta^{(c)}\mathbf{d}^{(c)}$; `/* Update the search direction */`
    $s^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c)}\mathbf{r}^{(c)}}{\mathbf{d}^{\mathrm{T}(c)}\mathbf{A}\mathbf{d}^{(c)}}$; `/* Compute the optimal step length */`
    $\hat{\beta}^{(c+1)} = \hat{\beta}^{(c)} + \zeta^{(c)}\mathbf{d}^{(c+1)}$; `/* Obtain approximate solution */`
    $\mathbf{r}^{(c+1)} = \mathbf{r}^{(c)} - s^{(c)}\mathbf{A}\mathbf{d}^{(c+1)}$; `/* Update the residual */`
    $c = c + 1$
end

adequate accuracy on very large datasets. For Algorithm 1, the maximum number of iterations is set to 30 and the relative difference of deviance threshold, ε_1, is set to 0.01. For Algorithm 2, the ridge regression parameter, λ, is set to 10 and the maximum number of iterations for the CG is set to 200 iterations. In addition, the CG convergence threshold, ε_2, is set to 0.005, and no more than three non-improving iterations are allowed on the CG algorithm.

Once the optimal MLE for $\hat{\beta}$ are found, classification of any given $i-th$ instance, $\mathbf{x}_i$, is carried out according to the following rules

$$\hat{y}_i = \begin{cases} 1, & \text{if } \hat{\eta}_i \geq 0 \text{ or } \hat{p}_i \geq 0.5 \text{ ;} \\ 0, & \text{otherwise.} \end{cases} \tag{3.34}$$

Aside from the implementation simplicity of TR-IRLS, the main advantage of the algorithm is that it can process and classify large datasets with little time, compared to other methods such as SVM. In addition, the TR-IRLS accuracy is comparable to that of SVM.

Furthermore, the algorithm does not require parameter tuning. This is an important characteristic when the goal is to classify large and balanced datasets.

Despite all of the aforementioned advantages of TR-IRLS, the algorithm is not designed to handle rare events data, and it is not designed to handle small-to-medium size datasets that are highly non-linearly separable [10].

3.3 Logistic Regression in Rare Events Data

3.3.1 Endogenous (Choice-Based) Sampling

Almost all of the conventional classification methods are based on the assumption that the training data consist of examples drawn from the same distribution as the testing data (or real-life data) [25, 26]. Likewise in *generalized linear models* (GLM), likelihood functions solved by methods such as LR are based on the concepts of random sampling or *exogenous* sampling [4, 88]. To see why this is the case [82, 89], under random sampling, the true joint distribution of $\mathbf{y}$ and $\mathbf{X}$ is $P(\mathbf{y}|\mathbf{X})P(\mathbf{X})$, and the likelihood function based on n binary observations is given by

$$\mathbb{L}_{Random} = \prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \beta)P(\mathbf{x}_i). \tag{3.35}$$

Under exogenous sampling, the sampling is on $\mathbf{X}$ according to a distribution $f(\mathbf{X})$, which may not reflect the actual distribution $P(\mathbf{X})$, and then $\mathbf{y}$ is sampled according to its true distribution probability $P(\mathbf{y}|\mathbf{X})$. The likelihood function would then be

$$\mathbb{L}_{Exogenous} = \prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \beta)f(\mathbf{x}_i). \tag{3.36}$$

As long as the ML estimator is not related to $P(\mathbf{X})$ or $f(\mathbf{X})$, then maximizing $\mathbb{L}_{Random}$ or $\mathbb{L}_{Exogenous}$ is equivalent to maximizing

$$\mathbb{L} = \prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \beta), \tag{3.37}$$

which is exactly the likelihood maximized by LR in (3.14) [26].

While the ML method is the most important method of estimation with a great advantage in general applicability, it is well-known that MLE of the unknown parameters, with exception to the normal distribution, are *asymptotically biased* in small samples. The ML properties are satisfied mainly asymptotically, meaning with the assumption of large samples [81, 90]. In addition, while it is ideal that sampling be either random or exogenous since it is reflective of the population or the testing data distribution, this sampling strategy has three major disadvantages when applied to REs. First, in data collection surveys, it would be very time consuming and costly to collect data on events that occur rarely. Second, in data mining, the data to be analyzed could be very large in order to contain enough REs, and hence computational time could be big. Furthermore, while the ML estimator is consistent in analyzing such data, it is asymptotically biased in the sense that the probabilities generated underestimate the actual probabilities of occurrence. In other words, the results are asymptotically biased. Cox and Hinkley [91] provided a general rough approximation for the asymptotic bias, developed originally by Cox and Snell [92], such that

$$E(\hat{\beta}-\beta) = -\frac{1}{2n}\frac{i_{30}+i_{11}}{i_{20}^2}, \tag{3.38}$$

where $i_{30} = E\left[\left(\frac{\partial L}{\partial \beta}\right)^3\right]$, $i_{11} = E\left[\left(\frac{\partial L}{\partial \beta}\right)\left(\frac{\partial^2 L}{\partial \beta^2}\right)\right]$, and $i_{20} = E\left[\left(\frac{\partial L}{\partial \beta}\right)^2\right]$, are evaluated at $\hat{\beta}$. Following King and Zeng [4], if $p_i = \frac{1}{1+e^{-\beta_0+x_i}}$, then the asymptotic bias is

$$\begin{aligned} E(\hat{\beta}_0-\beta_0) &= -\frac{1}{n}\frac{E[(0.5-\hat{p}_i)((1-\hat{p}_i)^2 y_i+\hat{p}_i^2(1-y_i))]^2}{(E[(1-\hat{p}_i)^2 y_i+\hat{p}_i^2(1-y_i)])} && (3.39)\\ &\approx \frac{\overline{p}-0.5}{n\overline{p}(1-\overline{p})}, && (3.40)\end{aligned}$$

where $\overline{p}$ is the proportion of events in the sample. Therefore, as long as $\overline{p}$ is less than 0.5 and/or n is small, the bias in (3.40) will not be equal to zero. Furthermore, the variance would be large. To see this mathematically, consider the variance matrix of the LR

estimator, $\hat{\beta}$, given by

$$\mathbf{V}(\hat{\beta}) = \left[\sum_{i=1}^{n} p_i(1-p_i)\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_i\right]^{-1}. \tag{3.41}$$

The variance given in (3.41) is smallest when the part $p_i(1-p_i)$, which is affected by rare events, is closer to 0.5. This occurs when the number of ones is large enough in the sample. However, the estimate of p_i with observations related to rare events is usually small, and hence additional ones would cause the variance to drop while additional zeros at the expense of events would cause the variance to increase [4, 93]. The strategy is to select on $\mathbf{y}$ by collecting observations for which $y_i = 1$ (the cases), and then selecting random observations for which $y_i = 0$ (the controls). The objective then is to keep the variance as small as possible by keeping a balance between the number of events (ones) and non-events (zeros) in the sample under study. This is achieved through *endogenous* sampling or *choice−based* sampling. Endogenous sampling occurs whenever sample selection is based on the dependent variable ($\mathbf{y}$), rather than on the independent (exogenous) variable ($\mathbf{X}$).

However, since the objective is to derive inferences about the population from the sample, the estimates obtained by the common likelihood using pure endogenous sampling are inconsistent. King and Zeng [4] recommend two methods of estimation for choice-based sampling, *prior correction* and *weighting*.

3.3.2 Correcting Estimates Under Endogenous Sampling

Prior Correction

Consider a population of N examples with τ as the proportion of events and $(1-\tau)$ as the proportion of non-events. Let the event of interest be $y = 1$ in the population with a probability $\tilde{p}$. Let n be the sample size with $\bar{y}$ and $(1-\bar{y})$ representing the proportions of events and non-events in the sample, respectively. Then, let $\hat{p}$ be the probability of the

event in the sample, and $s = 1$ be a selected event. By the Bayesian formula [93, 94],

$$\hat{p} = P(y = 1|s = 1) = \frac{P(s = 1|y = 1)P(y = 1)}{P(s = 1|y = 1)P(y = 1) + P(s = 1|y = 0)P(y = 0)} \tag{3.42}$$

$$= \frac{\left(\frac{\bar{y}}{\tau}\right)\tilde{p}}{\left(\frac{\bar{y}}{\tau}\right)\tilde{p} + \left(\frac{1-\bar{y}}{1-\tau}\right)(1-\tilde{p})}. \tag{3.43}$$

If the sample is random, then $\bar{y} = \tau$ and $1 - \bar{y} = 1 - \tau$, hence $\hat{p} = \tilde{p}$ and there is no inconsistency. When endogenous sampling is used to analyze imbalanced or rare events data, $\tau < (1 - \tau)$, and $\bar{y} \approx (1 - \bar{y})$, and hence $\hat{p} \neq \tilde{p}$, regardless of the sample size.

Now, assuming that $\hat{p}$ and $\tilde{p}$ are *logit* probabilities, and let $v_1 = \left(\frac{\bar{y}}{\tau}\right)$, and $v_0 = \left(\frac{1-\bar{y}}{1-\tau}\right)$, then equation (3.43) can be rewritten as

$$\hat{p} = \frac{v_1\tilde{p}}{v_1\tilde{p} + v_0(1-\tilde{p})}. \tag{3.44}$$

The odd of (3.44) is then

$$O = \frac{\hat{p}}{1-\hat{p}} = \frac{v_1\tilde{p}}{v_0(1-\tilde{p})}, \tag{3.45}$$

and the log odds is

$$\ln(O) = \ln\left(\frac{v_1}{v_0}\right) + \ln(\tilde{p}) - \ln(1-\tilde{p}), \tag{3.46}$$

which implies that

$$\mathbf{x}\hat{\beta} = \ln\left[\left(\frac{1-\tau}{\tau}\right)\left(\frac{\bar{y}}{1-\bar{y}}\right)\right] + \mathbf{x}\tilde{\beta}. \tag{3.47}$$

Prior correction is therefore easy to apply as it involves only correcting the intercept [4, 94], β_0, such that

$$\tilde{\beta}_0 = \hat{\beta}_0 - \ln\left[\left(\frac{1-\tau}{\tau}\right)\left(\frac{\bar{y}}{1-\bar{y}}\right)\right], \tag{3.48}$$

thereby making the corrected logit probability be

$$\tilde{p}_i = \frac{1}{1 + e^{\ln\left[\left(\frac{1-\tau}{\tau}\right)\left(\frac{\bar{y}}{1-\bar{y}}\right)\right] - \mathbf{x}_i\beta}}, \qquad \text{for } i = 1 \ldots n. \tag{3.49}$$

Prior correction requires knowledge of the fraction of events in the population, τ. The advantage of prior correction is its simplicity. However, the main disadvantage of this correction is that if the model is misspecified, then estimates on both $\hat{\beta}_0$ and $\hat{\beta}$ are less robust than weighting [4, 88].

Weighting

Under pure endogenous sampling, the conditioning is on $\mathbf{X}$ rather than $\mathbf{y}$ [89, 95], and the joint distribution of $\mathbf{y}$ and $\mathbf{X}$ in the sample is

$$f_s(\mathbf{y},\mathbf{X}|\beta) = P_s(\mathbf{X}|\mathbf{y},\beta)P_s(\mathbf{y}), \tag{3.50}$$

where β is the unknown parameter to be estimated. Yet, since $\mathbf{X}$ is a matrix of exogenous variables, then the conditional probability of $\mathbf{X}$ in the sample is equal to that in the population, or $P_s(\mathbf{X}|\mathbf{y},\beta) = P(\mathbf{X}|\mathbf{y},\beta)$. However, the conditional probability in the population is

$$P(\mathbf{X}|\mathbf{y},\beta) = \frac{f(\mathbf{y},\mathbf{X}|\beta)}{P(\mathbf{y})}, \tag{3.51}$$

but

$$f(\mathbf{y},\mathbf{X}|\beta) = P(\mathbf{y}|\mathbf{X},\beta)P(\mathbf{X}), \tag{3.52}$$

and hence, substituting and rearranging yields

$$f_s(\mathbf{y},\mathbf{X}|\beta) = \frac{P_s(\mathbf{y})}{P(\mathbf{y})}P(\mathbf{y}|\mathbf{X},\beta)P(\mathbf{X}) \tag{3.53}$$

$$= \frac{H}{Q}P(\mathbf{y}|\mathbf{X},\beta)P(\mathbf{X}), \tag{3.54}$$

where $\frac{H}{Q} = \frac{P_s(\mathbf{y})}{P(\mathbf{y})}$. The likelihood is then

$$\mathbb{L}_{Endogenous} = \prod_{i=1}^{n} \frac{H_i}{Q_i}P(y_i|\mathbf{x}_i,\beta)P(\mathbf{x}_i), \tag{3.55}$$

where $\frac{H_i}{Q_i} = \left(\frac{\bar{y}}{\tau}\right) y_i + \left(\frac{1-\bar{y}}{1-\tau}\right)(1-y_i)$. Therefore, when dealing with REs and imbalanced data, it is the likelihood in (3.55) that needs to be maximized [82, 88, 89, 96, 97]. Several consistent estimators of this type of likelihood have been proposed in the literature. Amemiya [82] and Ben Akiva and Lerman [98] provide an excellent survey of these methods.

Manski and Lerman [96] proposed the *weighted exogenous sampling maximum likelihood* (WESML), and proved that WESML yields a consistent and asymptotically normal estimator so long as knowledge of the population probability is available. More recently, Ramalho and Ramalho [99] extended the work of Manski and Lerman [96] to cases where such knowledge may not be available. Knowledge of population probability or proportions, however, can be acquired from previous surveys or existing databases. The log-likelihood for LR can then be rewritten as

$$\ln \mathbb{L}(\beta|\mathbf{y},\mathbf{X}) = \sum_{i=1}^{n} \frac{Q_i}{H_i} \ln P(y_i|\mathbf{x}_i,\beta) \tag{3.56}$$

$$= \sum_{i=1}^{n} \frac{Q_i}{H_i} \ln\left(\frac{e^{y_i \mathbf{x}_i \beta}}{1+e^{\mathbf{x}_i\beta}}\right) \tag{3.57}$$

$$= \sum_{i=1}^{n} w_i \ln\left(\frac{e^{y_i \mathbf{x}_i \beta}}{1+e^{\mathbf{x}_i\beta}}\right), \tag{3.58}$$

where $w_i = \frac{Q_i}{H_i}$. Therefore, in order to obtain consistent estimators, the likelihood is multiplied by the inverse of the fractions. The intuition behind weighting is that if the proportion of events in the sample is more than that in the population, then the ratio $\left(\frac{Q}{H}\right) < 1$ and hence the events are given less weight, while the non-events would be given more weight if their proportion in the sample is less than that in the population. This estimator, however, is not fully efficient, because the information matrix equality does not hold. This is

demonstrated as

$$-E\left[\frac{Q}{H}\nabla_{\beta}^{2}\ln P(\mathbf{y}|\mathbf{X},\beta)\right] \neq E\left[\left(\frac{Q}{H}\nabla_{\beta}\ln P(\mathbf{y}|\mathbf{X},\beta)\right)\left(\frac{Q}{H}\nabla_{\beta}\ln P(\mathbf{y}|\mathbf{X},\beta)\right)^{\mathrm{T}}\right], \quad (3.59)$$

and for the LR model it is

$$-\left[\frac{1}{n}\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)p_i(1-p_i)\mathbf{x}_i\mathbf{x}_j\right] \neq \left[\frac{1}{n}\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)^{2}p_i(1-p_i)\mathbf{x}_i\mathbf{x}_j\right]. \quad (3.60)$$

Let $\mathbf{A} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)p_i(1-p_i)\mathbf{x}_i\mathbf{x}_j$, and $\mathbf{B} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)^{2}p_i(1-p_i)\mathbf{x}_i\mathbf{x}_j$, then the asymptotic variance matrix of the estimator β is given by the *sandwich estimate*, such that $\mathbf{V}(\beta) = \mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1}$ [82, 88, 96].

Now that consistent estimators are obtained, finite-sample/rare-event bias corrections could be applied. King and Zeng [4] extended the small-sample bias corrections, as described by McCullagh and Nelder [100], to include the weighted likelihood (3.58), and demonstrated that even with choice-based sampling, these corrections can make a difference when the population probability of the event of interest is low. According to McCullagh and Nelder [100], and later Cordeiro and McCullagh [101], the bias vector is given by

$$bias(\hat{\beta}) = (\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{V}\xi, \quad (3.61)$$

where $\xi_i = Q_{ii}(\hat{p}_i - \frac{1}{2})$, and Q_{ii} are the diagonal elements of $\mathbf{Q} = \mathbf{X}(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}$, which is the approximate covariance matrix of the logistic link function η. The second-order bias-corrected estimator is then

$$\tilde{\beta} = \hat{\beta} - bias(\hat{\beta}). \quad (3.62)$$

As for the variance matrix $\mathbf{V}(\tilde{\beta})$ of $\tilde{\beta}$, it is estimated using

$$\mathbf{V}(\tilde{\beta}) = \left(\frac{n}{n+d}\right)^{2}\mathbf{V}(\hat{\beta}). \quad (3.63)$$

Since $\left(\frac{n}{n+d}\right)^2 < 1$, then $\mathbf{V}(\tilde{\beta}) < \mathbf{V}(\hat{\beta})$, and hence both the variance and the bias are now reduced.

The main advantage then of the bias correction method proposed by McCullagh and Nelder [100] is that it reduces both the bias and the variance [4]. The disadvantage of this bias correction method is that it is corrective and not preventive, since it is applied after the estimation is complete, and hence it does not protect against infinite parameter values that arise from perfect separation between the classes [102, 103]. Hence, this bias correction method can only be applied if the estimator, $\hat{\beta}$, has finite values. Firth [104] proposed a preventive second-order bias correction method by penalizing the log-likelihood such that

$$\ln \mathbb{L}(\beta|\mathbf{y},\mathbf{X}) = \sum_{i=1}^{n} \ln\left(\frac{e^{y_i \mathbf{x}_i \beta}}{1+e^{\mathbf{x}_i \beta}}\right) + \frac{1}{2}\ln|\mathbf{I}(\beta)|, \tag{3.64}$$

which leads to a modified score equation given by

$$\frac{\partial}{\partial \beta_j}\ln \mathbb{L}(\beta) = \sum_{i=1}^{n}(x_{ij}(y_i - p_i + h_i(0.5 - p_i))) = 0, \tag{3.65}$$

where h_i is the i-th diagonal element of the hat matrix

$$\mathbf{H} = \mathbf{V}^{\frac{1}{2}}\mathbf{X}(\mathbf{X}^{\mathrm{T}}\mathbf{V}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{V}^{\frac{1}{2}}. \tag{3.66}$$

A recent comparative simulation study by Maiti and Pradhan [105], however, showed that the bias correction of McCullagh and Nelder [100], provides the smallest *mean squared error* (MSE) when compared to that of Firth [104] and others using LR. Cordeiro and Barroso [106] more recently derived a third-order bias corrected estimator and showed that in some cases it could deliver improvements in terms of bias and MSE over the usual ML estimator and that of Cordeiro and McCullagh [101].

The challenge remains on finding the best class distribution in the training dataset. First, when both the events and non-events are easy to collect and both are available, then a sam-

ple with equal number of ones and zeros would be generally optimum [107, 108]. Second, when the number of events in the population is very small, the decision is then how many more non-events to collect in addition to the events. If collecting more non-events is inexpensive, then the general judgment is to collect as many non-events as possible. However, as the number of non-events exceed the number of events, the marginal contribution to the explanatory variables' information content starts to drop, and hence the number of zeros should be no more than two to five times the number of ones [4].

Applying the above corrections, offered by King and Zeng [4], along with the recommended sampling strategies, such as collecting all of the available events and only a matching proportion of non-events, could (1) significantly decrease the sample size under study, (2) cut data collection costs, (3) increase the rare event probability, and, (4) enable researchers to focus more on analyzing the variables.

Given all of the improvements made on the LR method, its underlying assumption of linearity, as evident in its logit function in (3.13), is often violated [8]. With the advancement of kernel methods, the search for an effective non-parametric LR model has become possible.

Chapter 4

Kernel Logistic Regression Using Truncated Newton Method

Summary

The combination of regularized Kernel Logistic Regression (KLR), truncated-Newton method, and Iteratively Re-weighted Least-Squares has led to a powerful classification method using small-to-medium size datasets. Compared to Support Vector Machines (SVM) and TR-IRLS on twelve benchmark publicly available datasets, my proposed algorithm is as accurate as, and much faster than, SVM, as well as more accurate than TR-IRLS. The proposed algorithm also has the advantage of providing direct prediction probabilities.

4.1 Introduction

Logistic Regression (LR) is an essential data mining technique for classifying binary datasets. Recently, there has been a revival of LR importance through the implementation of methods such as the truncated Newton. Truncated Newton methods have been effectively applied to solve large scale optimization problems. Komarek and Moore [2] were the first to show that the truncated-regularized iteratively re-weighted least squares (TR-IRLS) can be effectively implemented on LR to classify large datasets, and that it can outperform the support vector machine (SVM) algorithm. Later on, trust region Newton method [3], which is a type of truncated Newton, and truncated Newton interior-point methods [109] were also applied on LR to solve large scale problems. SVM [7] is considered a state-of-the-art algorithm for classifying binary data through its implementation of kernels (see Appendix B). Kernel Logistic Regression (KLR) [5, 6], which is a kernel version of LR has also proven

to be a powerful classifier [110]. Just like LR, KLR can naturally provide probabilities and extend to multi-class classification problems [8, 9].

Each one of aforementioned methods has a limitation. LR linearity may be an obstacle to handling highly nonlinearly separable small-to-medium size datasets [2]. SVM does not naturally extend to multi-class classification and does not provide probability estimates [110]. The SVM method also requires solving a constrained quadratic optimization problem with a time complexity of $O(n^3)$ [111] where n is the number of training instances. The KLR method is not sparse and requires all of the training instances in its model. Like SVM, KLR has a time complexity of $O(n^3)$. Its computation can be slow due to the density of its matrices [9]. Roth [112] was the first to apply the Conjugate Gradient (CG) on KLR and presented its efficiency on multi-class datasets. Zhu and Hastie [110] suggested the import vector machine (IVM) algorithm in order to take advantage of the SVM sparsity, thus reducing the time complexity to $O(n^2q^2)$ for binary classification, and $O(mn^2q^2)$ for the multi-class classification, where q is the number of import points and m is the number of classes. Keerthi et al. [113] incorporated the popular Sequential Minimal Optimization (SMO) algorithm in KLR and showed the results for binary classification. Karsmakers et al. [9] offered a fixed-size approach based on the number of support vectors to solve a multi-class KLR problems with the method of alternating descent. However, accuracy was compromised for faster computations.

The motivation for this study stems from the success and effectiveness of truncated Newton methods for solving large scale LR classification problems. In this chapter the speed of the TR-IRLS algorithm is combined with with the accuracy generated by the use of kernels for solving non-linear problems. The kernel version of the TR-IRLS algorithm (TR-KLR) is just as easy to implement and requires solving only an unconstrained regularized optimization problem. TR-KLR can also be extended to handle multi-class classification problems. To make the evaluation more thorough, the performance of TR-KLR is tested on twelve benchmark datasets, six of which are binary-class datasets and six are

multi-class datasets. In addition, for the multi-class datasets, the performance of TR-KLR is tested using One-Vs.-All (OVA) [7, 114], One-Vs.-One (OVO) [115], and Decision-Directed-Acyclic Graph (DDAG) [116] coding methods.

4.2 Kernel Logistic Regression

In the previous section, it was shown that the first-order conditions for LR are given by

$$\nabla_\beta \ln \mathbb{L}(\beta) = \mathbf{X}^{\mathrm{T}}(\mathbf{y}-\mathbf{p}) - \lambda\beta = \mathbf{0}. \tag{4.1}$$

By solving for β,

$$\begin{aligned} \beta &= \mathbf{X}^{\mathrm{T}}(\mathbf{y}-\mathbf{p})\lambda^{-1} && (4.2)\\ &= \mathbf{X}^{\mathrm{T}}\alpha = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i, && (4.3) \end{aligned}$$

where the vector α is known as the *dual variable*, and $\alpha = (\mathbf{y}-\mathbf{p})\lambda^{-1}$ with dimensions $n \times 1$. Therefore, the vector β can be expressed as a linear combination of the data points. Now, the logit vector η can be rewritten as

$$\begin{aligned} \eta &= \mathbf{X}\mathbf{X}^{\mathrm{T}}\alpha && (4.4)\\ &= \mathbf{K}\alpha, && (4.5) \end{aligned}$$

where $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$. The matrix $\mathbf{K}$ is referred to as the Gram matrix, which is symmetric positive semidefinite, with $n \times n$ dimensions.

Consider again the logit link function shown in the previous chapter,

$$\eta_i = \mathbf{x}_i\beta = \beta_0 + x_{i1}\beta_1 \ldots x_{id}\beta_d, \tag{4.6}$$

where the vector $\mathbf{x}_i$, given by $\mathbf{x}_i = [1, x_{i1}, \ldots x_{id}]$ with $i = 1, \ldots n$, is a d dimensional row

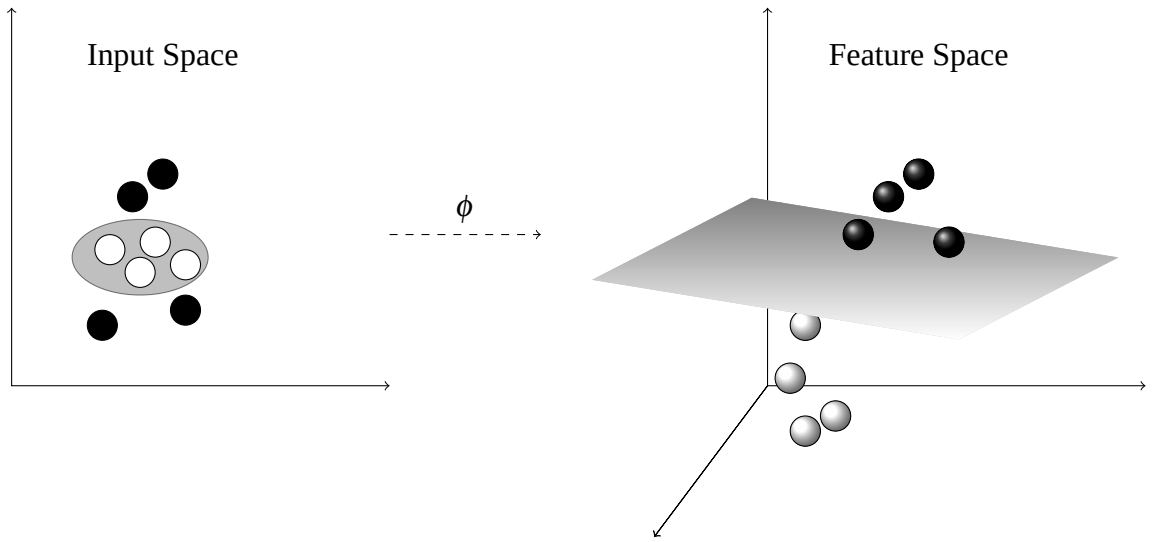

Figure 4.1: Mapping of non-linearly separable data from the input space to the feature space.

vector representing *d features*. The link function can be considered a simple linear model for regression, involving a linear combination of the input variables. Stated differently, this linear function represents the simplest form of an identity mapping polynomial basis function ϕ of the feature space such that $\phi(\mathbf{x}_i) = \phi[(1, x_{i1}, \ldots x_{id})] = \mathbf{x}_i$. Thus, the logit link function could be rewritten as

$$\eta_i = \phi(\mathbf{x}_i)\beta. \tag{4.7}$$

In general, the function $\phi(.)$ maps the data from a lower dimensional space into a higher one (Figure 4.1), such that

$$\phi : \mathbf{x} \in \mathbb{R}^d \rightarrow \phi(\mathbf{x}) \in \mathbb{F} \subseteq \mathbb{R}^\Lambda. \tag{4.8}$$

The goal for choosing the mapping ϕ is to convert nonlinear relations between the response (endogenous) variable and the independent (exogenous) variables into linear relations. However, the transformations $\phi(.)$ are often unknown but the dot product in the feature space can be expressed in terms of the input vectors through the kernel function. In

the case of KLR, the logit link function could be rewritten as

$$\begin{aligned} \eta_i &= \sum_{j=1}^{n} \alpha_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle && (4.9) \\ &= \sum_{j=1}^{n} \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) && (4.10) \\ &= \mathbf{k}_i \alpha, && (4.11) \end{aligned}$$

where $\mathbf{k}_i$ is the *i*-th row in the kernel matrix $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{K}$. The kernel is a transformation function that must satisfy Mercer's necessary and sufficient conditions, which state that a kernel function must be expressed as an inner product and must be positive semidefinite [117, 118].

Theorem 1 *(Mercer's Theorem) A kernel function κ can be expressed as an inner product*

$$\kappa(\mathbf{u}, \mathbf{v}) = \langle \phi(\mathbf{u}), \phi(\mathbf{v}) \rangle,$$

if and only if, for any function $f(\mathbf{u})$ such that $\int f(\mathbf{u})^2 dx$ is finite, then

$$\int \kappa(\mathbf{u}, \mathbf{v}) f(\mathbf{u}) f(\mathbf{v}) d\mathbf{u} d\mathbf{v} \geq 0.$$

Among the most well known kernels that satisfy Mercer's theorem are

- Linear Kernel: $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$.
- Polynomial Kernel: $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1))^p$ where p is the degree of the polynomial.
- Radial Basis Function (RBF) Kernel: $\kappa(\mathbf{x}_i, 0\mathbf{x}_j) = e^{(-\gamma ||\mathbf{x}_i - \mathbf{x}_j||)^2}$ where γ is the kernel parameter.

Now,

$$\eta_i = \mathbf{k}_i \alpha = \ln(\frac{p_i}{1 - p_i}), \quad (4.12)$$

which implies that

$$p_i = \frac{e^{\eta_i}}{1+e^{\eta_i}} = \frac{e^{\mathbf{k}_i\alpha}}{1+e^{\mathbf{k}_i\alpha}} = \frac{1}{1+e^{-\mathbf{k}_i\alpha}}, \tag{4.13}$$

and hence, the regularized log-likelihood can be rewritten with respect to α as

$$\ln\mathbb{L}(\alpha) = \sum_{i=1}^{n}(y_i \ln p_i + (1-y_i)\ln(1-p_i)) - \frac{\lambda}{2}\alpha^{\mathrm{T}}\mathbf{K}\alpha, \tag{4.14}$$

with a deviance

$$\mathbb{DEV}(\alpha) = -2\ln\mathbb{L}(\alpha). \tag{4.15}$$

4.3 Iteratively Re-weighted Least Squares

KLR models can also be fitted using IRLS [9]. The gradient and Hessian are obtained by differentiating $\ln\mathbb{L}(\alpha)$ with respect to α. In matrix form, the gradient is

$$\nabla_\alpha \ln\mathbb{L}(\alpha) = \mathbf{K}^{\mathrm{T}}(\mathbf{y}-\mathbf{p}) - \lambda\mathbf{K}\alpha. \tag{4.16}$$

The Hessian with respect to α is

$$\nabla^2_\alpha \ln\mathbb{L}(\alpha) = -\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} - \lambda\mathbf{K}, \tag{4.17}$$

where $\mathbf{V}$ is a diagonal matrix with diagonal elements $p_i(1-p_i)$ for $i = 1...n$. The Newton-Raphson update with respect to α on the $(c+1)-th$ iteration is

$$\hat{\alpha}^{(c+1)} = \hat{\alpha}^{(c)} + (\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})^{-1}(\mathbf{K}^{\mathrm{T}}(\mathbf{y}-\mathbf{p}) - \lambda\mathbf{K}\alpha^{(c)}). \tag{4.18}$$

Since $\hat{\alpha}^{(c)} = (\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})^{-1}(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})\hat{\alpha}^{(c)}$, then equation (11) can be rewritten as

$$\hat{\alpha}^{(c+1)} = (\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})^{-1}\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}, \tag{4.19}$$

where $\mathbf{z}^{(c)} = \mathbf{K}\hat{\alpha}^{(c)} + \mathbf{V}^{-1}(\mathbf{y} - \mathbf{p})$ is the adjusted dependent variable or the adjusted response.

4.4 TR-KLR Algorithm

The KLR WLS subproblem,

$$(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})\hat{\alpha}^{(c+1)} = \mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}, \tag{4.20}$$

is a systems of linear equations with a kernel matrix **K**, vector of adjusted responses **z**, and a weight matrix **V**. Both the weights and the adjusted response vector are dependent on $\hat{\alpha}^{(c)}$, which is the current estimate of the parameter vector. Thus, an initial estimate $\hat{\alpha}^{(0)}$ can be specified for $\hat{\alpha}$ and solved iteratively, giving a sequence of estimates that converges to the MLE of $\hat{\alpha}$. The linear CG method can also be applied here, minimizing the quadratic function,

$$\frac{1}{2}\hat{\alpha}^{(c+1)}(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})\hat{\alpha}^{(c+1)} - \hat{\alpha}^{(c+1)}(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}). \tag{4.21}$$

When applied to KLR, the CG method has a time complexity of $O(n^3)$ in the worst case, as it converges in at most n steps. To avoid the long computations that the CG may suffer from, a limit to the number of CG iterations can be placed, thus creating an approximate, or truncated Newton direction.

Similar to the TR-IRLS algorithm, Algorithm 3 represents the main (outer) loop of TR-KLR and it summarizes the IRLS for KLR and is terminated when the relative difference of deviance between two consecutive iterations is no greater than a specified threshold ε_1. As with TR-IRLS, the main problem to solve is the WLS in (4.20), which is a linear system of n equations and n variables. This is done through Algorithm 4, which represents the inner loop that approximates the Newton direction through the CG method. Algorithm 4 is the linear CG algorithm and is terminated when the CG residual is less than a threshold ε_2.

Algorithm 3: KLR MLE using IRLS

Data: $\mathbf{K}, \mathbf{y}, \hat{\alpha}^{(0)}$
Result: $\hat{\alpha}$

begin
  $c = 0$
  while $|\frac{DEV^{(c)} - DEV^{(c+1)}}{DEV^{(c+1)}}| > \varepsilon_1$ **and** $c \leq$ *Max IRLS Iterations* **do**
    for $i \leftarrow 1$ **to** n **do**
      $\hat{p}_i = \frac{1}{1+e^{-\mathbf{x}_i \hat{\alpha}}}$; /* Compute probabilities */
      $v_i = \hat{p}_i(1-\hat{p}_i)$; /* Compute weights */
      $z_i = \mathbf{k}_i \hat{\alpha}^{(c)} + \frac{(y_i - \hat{p}_i)}{\hat{p}_i(1-\hat{p}_i)}$; /* Compute adjusted response */
    $\mathbf{V} = diag(v_1, ..., v_n)$
    $(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K})\hat{\alpha}^{(c+1)} = \mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{z}^{(c)}$; /* Compute $\hat{\alpha}$ via WLS */
    $c = c + 1$
end

Algorithm 4: Linear CG. $\mathbf{A} = \mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K} + \lambda\mathbf{K}$, $\mathbf{b} = \mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{z}$

Data: $\mathbf{A}, \mathbf{b}, \hat{\alpha}^{(0)}$
Result: $\hat{\alpha}$ such that $\mathbf{A}\hat{\alpha} = \mathbf{b}$

begin
  $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\hat{\alpha}^{(0)}$; /* Initialize the residual */
  $c = 0$
  while $||\mathbf{r}^{(c+1)}||^2 > \varepsilon_2$ **and** $c \leq$ *Max CG Iterations* **do**
    if $c = 0$ **then**
      $\zeta^{(c)} = 0$
    else
      $\zeta^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c+1)}}{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c)}}$; /* Update **A**-Conjugacy enforcer */
    $\mathbf{d}^{(c+1)} = \mathbf{r}^{(c+1)} + \zeta^{(c)}\mathbf{d}^{(c)}$; /* Update the search direction */
    $s^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c)}\mathbf{r}^{(c)}}{\mathbf{d}^{\mathrm{T}(c)}\mathbf{A}\mathbf{d}^{(c)}}$; /* Compute the optimal step length */
    $\hat{\alpha}^{(c+1)} = \hat{\alpha}^{(c)} + \zeta^{(c)}\mathbf{d}^{(c+1)}$; /* Obtain approximate solution */
    $\mathbf{r}^{(c+1)} = \mathbf{r}^{(c)} - s^{(c)}\mathbf{A}\mathbf{d}^{(c+1)}$; /* Update the residual */
    $c = c + 1$
end

With exception to the value of ε_1, the default parameter values suggested by Komarek and Moore [87] are used for both algorithms and are shown to provide adequate accuracy. For Algorithm 3, the maximum number of iterations is set to 30 and for the relative difference of deviance threshold, ε_1, the value 2.5 is sufficient to reach the desired accuracy and at the same time maintain good convergence speed. By choosing the value 2.5 as a threshold, computational speed is improved while not affecting accuracy. Should the algorithm reach a certain desired accuracy with a low threshold, then by slightly modifying the parameters values (e.g. σ, λ) with a larger threshold, the same accuracy can be reached, hence the robustness of the algorithm. However, in some cases it may be advisable to make this threshold smaller to obtain better accuracy. As for Algorithm 4, the maximum number of iterations for the CG is set to 200 iterations. In addition, the CG convergence threshold, ε_2, is set to 0.005. Furthermore, no more than three non-improving iterations are allowed on the CG algorithm.

4.5 Computational Results & Discussion

The performance of the TR-KLR algorithm was examined using twelve benchmark datasets (see Table 4.1) found on the UC Irvine website [119], six of which are binary classification datasets, and the other six are multi-class classification datasets. The algorithm performance was then compared to that of both SVM and TR-IRLS. The binary datasets used are Wisconsin Breast Cancer Diagnostic (WBCD), Ionosphere, Bupa Liver Disorders, Haberman Survival, Pima Indians Diabetes, and Sonar datasets. The multi-class datasets consist of Wine, Glass, Iris, Dermatology, Thyroid, and Ecoli datasets. In addition, the multi-class classification performance was assessed using three methods; one vs. all, one vs. one, and DDAG. The Gaussian Radial Basis Function (RBF) kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{(-\frac{1}{2\sigma^2}||\mathbf{x}_i - \mathbf{x}_j||)^2} = e^{(-\gamma||\mathbf{x}_i - \mathbf{x}_j||)^2} \tag{4.22}$$

was used for both the TR-KLR and SVM methods, where σ is the kernel parameter. The values of these parameters that give the best generalization are usually chosen from a range of different values (generally user-defined), and tuned using ten-fold cross-validation (CV). The datasets were preprocessed using normalization of a mean of zero and standard deviation of one. All of the computations for TR-IRLS and TR-KLR were carried out using MATLAB version 2007a on a 512 MiB RAM computer for the binary datasets and on a 1.5 GiB RAM computer for the multi-class datasets. As for the SVM method, MATLAB SVM Toolbox [120] was used for the binary datasets and MATLAB LIBSVM toolbox [121] for the multi-class datasets.

Table 4.1: Datasets.

Dataset	**Instances**	**Features**	**Classes**
WBCD	569	30	2
Ionosphere	351	34	2
Liver	345	6	2
Survival	306	3	2
Sonar	208	60	2
Diabetes	768	8	2
Wine	178	13	3
Glass	214	10	6
Iris	150	4	3
Dermatology	358	34	6
Thyroid	215	5	3
Ecoli	336	7	8

4.5.1 Binary Classification

For the binary datasets, Tables 4.2 and 4.3 summarize the computation results for these three methods with their optimal parameters and accuracy, respectively. Table 4.4 shows a comparison of the total execution time with ten-fold CV using the three methods. Table 4.3 shows that the TR-KLR method scored as well as or slightly better than SVM on most of the datasets. In addition, both TR-KLR and SVM performed better than TR-IRLS on four out of the six datasets, namely, Ionosphere, Liver, and Sonar. As with all kernel methods,

parameter tuning is unavoidable as it involves two parameters, the regularization parameter (λ for TR-KLR and C for SVM), and the kernel parameter (σ or γ for both TR-KLR and SVM). For TR-IRLS, the only parameter that requires tuning is λ. While Komarek and Moore [2] observed that the value of λ did not affect the accuracy for large datasets, and hence they were able to use default values, the same cannot be said for smaller datasets. TR-IRLS challenged both TR-KLR and SVM on three datasets, WBCD, Survival, and Diabetes. This is probably because these datasets are more linearly separable than the others, on which TR-IRLS performed worse.

Table 4.2: Optimal parameter values found for the binary-class datasets.

	TR-KLR		**SVM**		**TR-IRLS**
	σ	λ	σ	C	λ
WBCD	5.4	0.1	5.0	10.0	30.0
Ionosphere	3.5	0.009	1.9	100.0	30.0
Liver	7.0	0.0009	5.0	10.0	0.05
Survival	5.0	0.01	1.8	1.0	10.0
Sonar	3.2	0.05	7.1	10.0	50.0
Diabetes	5.0	0.07	7.5	1.0	1.0

Table 4.3: Comparison of ten-fold CV accuracy (%) with 95 % confidence level.

	TR-KLR	**SVM**	**TR-IRLS**
WBCD	98.1 ± 1.1	**98.2** ± 1.1	98.1 ± 1.1
Ionosphere	**93.7** ± 2.5	90.6 ± 3.1	88.0 ± 3.4
Liver	**70.1** ± 4.8	**70.1** ± 4.8	65.3 ± 5.0
Survival	**75.4** ± 4.8	75.1 ± 4.9	73.8 ± 4.9
Sonar	**89.0** ± 4.3	87.9 ± 4.5	79.2 ± 5.5
Diabetes	**78.0** ± 2.9	77.2 ± 3.0	**78.0** ± 2.9

With regard to the execution time, Table 4.4 shows that while TR-IRLS is the fastest, TR-KLR is still significantly faster than SVM and yet identical to it with regard to accuracy while at the same time more accurate than the regular TR-IRLS, as shown in Table 4.3.

As mentioned earlier, the maximum number of iterations for IRLS was set to 30 while that of the CG method was set to 200. However, as Komarek and Moore [87] correctly

Table 4.4: Comparison of ten-fold CV time (in seconds).

	TR-KLR	**SVM**	**TR-IRLS**
WBCD	10.6	3305	5.8
Ionosphere	3.6	248	2.3
Liver	3.0	209	2.2
Survival	2.1	158	1.5
Sonar	1.3	58	1.5
Diabetes	17.0	7374	4.8

Table 4.5: Maximum number of iterations reached by IRLS and CG during the ten-fold CV on the binary-class datasets.

	IRLS	**CG**
WBCD	2	31
Ionosphere	2	32
Liver	1	19
Survival	1	14
Sonar	2	46
Diabetes	1	25

stated, these numbers should never be reached. This observation applies also to the TR-KLR algorithm, as shown in the empirical results in Table 4.5, where the maximum number of iterations reached by both algorithms during the ten-fold cross validation is displayed for the binary-class datasets. The maximum iterations reached on the binary-class datasets by IRLS in Algorithm 3 was 2 while the maximum CG iterations in Algorithm 4 was 46. Therefore, the number of iterations is relatively small compared to the size of the datasets.

4.5.2 Multi-class Classification

As mentioned earlier, for multi-class classification, the performance of TR-KLR was evaluated against both SVM and TR-IRLS using three methods: OVA, OVO, and DDAG. While LR and KLR can naturally extend to multi-class classification [9], the aforementioned multi-class coding schemes were applied in this study to make the comparison with SVM fair. The OVA approach [7, 114] constructs M classifiers for M classes. Classifier f_m is trained to discriminate between class m and all other classes. Then, the class of instance

$\mathbf{x}_i$ corresponds to the maximal value of the function $f_m(\mathbf{x}_i)$ such that $C_m = \max f_m(\mathbf{x}_i)$ for $m = 1,..,M$, where C_m is the class of $\mathbf{x}_i$. The OVO approach [115], constructs $\frac{M(M-1)}{2}$ binary discriminant functions, one for every pair of classes and proceeds as the OVA approach. As for the DDAG approach [116], it also constructs $\frac{M(M-1)}{2}$ classifiers (nodes) in the training phase. In the testing phase, it utilizes decision-directed-acyclic graph (DDAG), whereby at each node a binary classifier is constructed and the next node visited depends upon the results of this evaluation. If the value of the binary decision function is zero, the node exits from the left, otherwise, if the value is one, then the node exits from the right. The final answer is the class assigned by the leaf node visited at the final step as illustrated by Figure 4.2. The root node can be assigned randomly. The advantages of DDAG are fast computational time especially for large-scale problems.

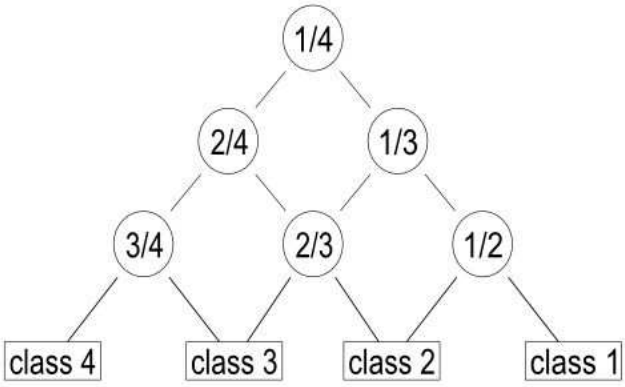

Figure 4.2: Illustration of DDAG for classification with four classes.

Table 4.6 lists the optimal parameters used to reach the desired accuracies, which are shown in Table 4.7. Table 4.6 shows that the parameters used to reach the desired accuracies are identical for all algorithms using both OVA and DDAG methods. In addition, it appears that the kernel parameter values (σ or γ) and the regularization parameter (λ) are more stable using OVO and DDAG as they do not vary much from one data set to another.

Table 4.7 shows the ten-fold CV accuracy with 95% confidence level reached by all algorithms using the three multi-class classification methods. With regard to OVA, TR-KLR

Table 4.6: Optimal parameter values for the multi-class datasets. C is the SVM regularization parameter, σ is the parameter (width) of the RBF kernel, and λ is the regularization parameter for both TR-IRLS and TR-KLR.

OVA	**TR-KLR**		**SVM**		**TR-IRLS**
	σ	λ	γ	C	λ
Wine	6.0	0.005	0.1	1.0	0.5
Glass	1.0	0.0005	0.1	100.0	0.09
Iris	3.0	0.001	0.02	10.0	0.01
Dermatology	5.0	0.1	0.01	1.0	0.3
Thyroid	1.6	0.004	7.1	0.09	0.005
Ecoli	6.0	0.01	7.5	0.01	0.5
OVO	**TR-KLR**		**SVM**		**TR-IRLS**
	σ	λ	γ	C	λ
Wine	4.0	0.01	0.01	1.0	0.5
Glass	1.0	0.01	0.1	1.0	0.5
Iris	5.0	0.01	0.03	10.0	0.004
Dermatology	5.0	0.02	0.01	1.0	0.3
Thyroid	1.2	0.004	0.1	10.0	0.005
Ecoli	5.0	0.05	0.3	1.0	0.5
DDAG	**TR-KLR**		**SVM**		**TR-IRLS**
	σ	λ	γ	C	λ
Wine	4.0	0.01	0.01	1.0	0.5
Glass	1.0	0.01	0.1	1.0	0.5
Iris	5.0	0.01	0.03	10.0	0.004
Dermatology	5.0	0.02	0.01	1.0	0.3
Thyroid	1.2	0.004	0.1	10.0	0.005
Ecoli	5.0	0.05	0.3	1.0	0.5

Table 4.7: Comparison of ten-fold CV accuracy (%) with 95 % confidence level.

OVA	**TR-KLR**	**SVM**	**TR-IRLS**
Wine	**99.47** $\pm$ 1.06	99.44 $\pm$ 1.09	98.36 $\pm$ 1.86
Glass	**74.98** $\pm$ 5.80	72.09 $\pm$ 6.01	65.35 $\pm$ 6.42
Iris	**98.00** $\pm$ 2.24	97.33 $\pm$ 2.58	96.00 $\pm$ 3.14
Dermatology	**97.45** $\pm$ 1.63	**97.45** $\pm$ 1.72	97.19 $\pm$ 1.71
Thyroid	**97.64** $\pm$ 2.03	96.73 $\pm$ 2.38	95.32 $\pm$ 2.82
Ecoli	**88.90** $\pm$ 3.36	88.08 $\pm$ 3.47	88.11 $\pm$ 3.46
OVO	**TR-KLR**	**SVM**	**TR-IRLS**
Wine	**100.0** $\pm$ 0.00	99.44 $\pm$ 1.09	98.36 $\pm$ 1.86
Glass	**75.53** $\pm$ 5.76	72.09 $\pm$ 6.01	65.05 $\pm$ 6.39
Iris	**98.00** $\pm$ 2.24	**98.00** $\pm$ 2.24	**98.00** $\pm$ 2.24
Dermatology	**98.00** $\pm$ 1.35	97.45 $\pm$ 1.63	97.73 $\pm$ 1.45
Thyroid	**97.66** $\pm$ 2.02	**97.64** $\pm$ 2.03	**97.64** $\pm$ 2.03
Ecoli	**88.46** $\pm$ 3.42	87.86 $\pm$ 3.49	88.02 $\pm$ 3.47
DDAG	**TR-KLR**	**SVM**	**TR-IRLS**
Wine	**100.0** $\pm$ 0.00	99.44 $\pm$ 1.09	98.36 $\pm$ 1.86
Glass	**75.53** $\pm$ 5.76	72.09 $\pm$ 6.01	65.05 $\pm$ 6.39
Iris	**98.00** $\pm$ 2.24	**98.00** $\pm$ 2.24	**98.00** $\pm$ 2.24
Dermatology	**98.00** $\pm$ 1.35	97.45 $\pm$ 1.63	97.73 $\pm$ 1.45
Thyroid	**97.66** $\pm$ 2.02	**97.64** $\pm$ 2.03	**97.64** $\pm$ 2.03
Ecoli	**88.46** $\pm$ 3.42	87.86 $\pm$ 3.49	88.02 $\pm$ 3.47

performed better than both SVM and TR-IRLS on all datasets. As for the OVO and DDAG methods, accuracies were identical on both methods. TR-KLR generally performed best using OVO and DDAG except for the Ecoli data, on which OVA performed slightly better. The performance of SVM appears consistent using all methods. On the other hand, TR-IRLS accuracies varied depending on the multi-class classification method used. TR-IRLS performed poorer using OVA than using OVO and DDAG, especially on Iris and Thyroid datasets. On the Glass data set, TR-IRLS performs poorest with a difference of almost 10 percentage point in accuracy compared to TR-KLR, and 7 percentage point compared to SVM.

Similar to the binary-class case, the maximum number of iterations reached by both algorithms during the ten-fold cross validation is displayed in Table 4.8 for the multi class

Table 4.8: Maximum number of iterations reached by IRLS and CG during the ten-fold CV on the multi-class datasets.

	OVA		**OVO**		**DDAG**	
	IRLS	**CG**	**IRLS**	**CG**	**IRLS**	**CG**
Wine	2	30	2	37	2	37
Glass	2	44	2	39	2	39
Iris	1	23	2	11	2	11
Dermatology	1	59	2	44	2	44
Thyroid	2	30	2	35	2	35
Ecoli	2	29	2	21	2	21

datasets. As can be observed, the maximum iterations reached by IRLS in Algorithm 3 is 2 while the maximum CG iterations in Algorithm 4 was 59, indicating that the number of iterations is also relatively small compared to the size of the datasets.

As can be seen, the TR-KLR algorithm is relatively easy to implement and is as effective as SVM on small-to-medium size datasets. The TR-KLR algorithm takes advantage of the speed of the TR-IRLS and the power of the kernel methods, particularly when the datasets are neither large nor linearly separable. Another benefit to using TR-KLR is that it uses unconstrained optimization methods whose algorithms are less complex than those with constrained optimization methods, such as SVM.

Chapter 5

Robust Weighted Kernel Logistic Regression in Imbalanced and Rare Events Data

Summary

Accurate prediction is important in data mining and data classification. This chapter develops my proposed Rare-Event Weighted Kernel Logistic Regression (RE-WKLR) algorithm in rare events data with binary dependent variables containing many times more non-events (zeros) than events (ones). It also applies the necessary corrections to improve the prediction accuracy.

5.1 KLR for Rare Events and Imbalanced Data

Like the LR model, the commonly used maximum likelihood formulation on KLR is not appropriate for classifying imbalanced and rare events data, especially when endogenous sampling is performed. The full likelihood function needs to be stated. The likelihood function should then be

$$\mathbb{L}(\alpha|\mathbf{y},\mathbf{K}) = f(\mathbf{y},\mathbf{K}|\alpha) = \frac{H}{Q}P(\mathbf{y}|\mathbf{K},\alpha)P(\mathbf{K}) \tag{5.1}$$

$$= \prod_{i=1}^{n} \frac{H_i}{Q_i}P(y_i|\mathbf{k}_i,\alpha)P(\mathbf{k}_i). \tag{5.2}$$

Now, following the same intuitive concept of Manski and Lerman [96], choice-based sampling can easily be dealt with so long as knowledge of the population probability is

available. The log-likelihood for KLR can then be rewritten as

$$\ln \mathbb{L}(\alpha|\mathbf{y},\mathbf{K}) = \sum_{i=1}^{n} \frac{Q_i}{H_i} \ln P(y_i|\mathbf{k}_i,\alpha) \tag{5.3}$$

$$= \sum_{i=1}^{n} \frac{Q_i}{H_i} \ln \frac{e^{y_i \mathbf{k}_i \alpha}}{1+e^{\mathbf{k}_i \alpha}} \tag{5.4}$$

$$= \sum_{i=1}^{n} w_i \ln \frac{e^{y_i \mathbf{k}_i \alpha}}{1+e^{\mathbf{k}_i \alpha}}, \tag{5.5}$$

where $w_i = \dfrac{Q_i}{H_i}$. As with LR, in order to obtain a consistent estimator, the likelihood is multiplied by the inverse of the fractions. This produces a Weighted Maximum Likelihood (WML), and the KLR model becomes a Weighted KLR (WKLR) model. It can be shown that the WML estimator for WKLR is consistent. From the regularity conditions (see Appendix A), the WML estimator solves the first-order conditions

$$\frac{Q}{H} \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha) = \mathbf{0}. \tag{5.6}$$

Taking the expectation of the score function with regard to the sample density yields

$$E\left[\frac{Q}{H} \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha)\right] = \int \frac{Q}{H} \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha) \frac{H}{Q} P(\mathbf{y}|\mathbf{K},\alpha) P(\mathbf{K}) d\mathbf{K} \tag{5.7}$$

$$= \int \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha) P(\mathbf{y}|\mathbf{K},\alpha) P(\mathbf{K}) d\mathbf{K} \tag{5.8}$$

$$= \int E\left[\nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha)\right] P(\mathbf{K}) d\mathbf{K} \tag{5.9}$$

$$= \mathbf{0}. \tag{5.10}$$

This estimator, however, like its LR counterpart, is not fully efficient, because the information matrix equality does not hold. This is demonstrated as

$$-E\left[\frac{Q}{H} \nabla_\alpha^2 \ln P(\mathbf{y}|\mathbf{K},\alpha)\right] \neq E\left[\left(\frac{Q}{H} \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha)\right)\left(\frac{Q}{H} \nabla_\alpha \ln P(\mathbf{y}|\mathbf{K},\alpha)\right)^{\mathrm{T}}\right], \tag{5.11}$$

because

$$-E\left[\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)p_i(1-p_i)\mathbf{k}_i\mathbf{k}_j\right] \neq E\left[\sum_{i=1}^{n}\left(\frac{Q_i}{H_i}\right)^2 p_i(1-p_i)\mathbf{k}_i\mathbf{k}_j\right]. \tag{5.12}$$

Now, as mentioned in Chapter 4, KLR regularization is used in the form of the ridge penalty $\frac{\lambda}{2}\alpha^{\mathrm{T}}\mathbf{K}\alpha$, where $\lambda > 0$, is a regularization parameter. When regularization is introduced, none of the coefficients is set to zero [122], and hence the problem of infinite parameter values is avoided. In addition, the importance of the parameter λ lies in determining the bias-variance trade-off of an estimator [123, 124]. When λ is very small, there is less bias but more variance. On the other hand, larger values of λ would lead to more bias but less variance [125]. Therefore, the inclusion of regularization in the WKLR model is very important to reduce any potential inefficiency. However, as regularization carries the risk of a non-negligible bias, even asymptotically [125], the need for bias correction becomes inevitable. In sum, the bias correction is needed to account for any bias resulting from regularization, small samples, and rare events.

5.1.1 Rare Events and Finite Sample Correction

Like the ordinary LR model, the method for computing the KLR probability,

$$Pr(Y_i = 1|\alpha) = \hat{p}_i = \frac{1}{1+e^{-\mathbf{k}_i\hat{\alpha}}}, \tag{5.13}$$

is affected by the problem of $\hat{\alpha}$, which is a biased estimate of α.

Bias Adjustment and Parameter Estimation

Following McCullagh and Nelder [100], the bias in large samples may be very small. However, for samples of smaller size, or for samples in which the number of parameters is large compared to the number of instances, the bias may not be so small. For the KLR model,

the approximate bias vector can be written as

$$\mathbf{b} = E(\hat{\alpha} - \alpha), \tag{5.14}$$

and by following the same methodology used by McCullagh and Nelder [100], it can be shown that the approximate asymptotic covariance matrix of η is given by

$$\mathbf{Q} = \mathbf{K}(\mathbf{K}^{\mathrm{T}}\mathbf{V}\mathbf{K})^{-1}\mathbf{K}, \tag{5.15}$$

where $\mathbf{V} = diag(p_i(1-p_i))$ for $i = 1...n$. Replacing the kernel matrix $\mathbf{K}$ with the matrix $\tilde{\mathbf{K}} = (\mathbf{K} + \delta\mathbf{I})$, for a very small $\delta > 0$, in order to make it invertible, would enable (5.15) to reduce into

$$\mathbf{Q} = (\mathbf{V})^{-1}. \tag{5.16}$$

Let Q_{ii} be the i^{th} diagonal element of $\mathbf{Q}$, and

$$\xi_i = -\frac{1}{2}\left(\frac{p_i^{''}}{p_i^{'}}\right)Q_{ii}, \tag{5.17}$$

where $p_i^{'} = \dfrac{\partial p_i}{\partial \eta_i}$ and $p_i^{''} = \dfrac{\partial^2 p_i}{\partial \eta_i^2}$ are the derivatives of the KLR logit function, then the bias vector $\mathbf{b}$ can be written in matrix form as

$$bias(\hat{\alpha}) = (\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{V}\tilde{\mathbf{K}})^{-1}\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{V}\xi, \tag{5.18}$$

which is obtained as the vector of regression coefficients with ξ as a response vector.

Applying now the formulation suggested by King and Zeng [4] on the weighted LR to the WKLR model in (5.5), the weighted likelihood can be rewritten as

$$\mathbb{L}_W(\alpha) = \prod_{i=1}^{n}(p_i)^{w_1 y_i}(1-p_i)^{w_0(1-y_i)}, \tag{5.19}$$

where $w_1 = \frac{\tau}{\bar{y}}$, and $w_0 = \frac{1-\tau}{1-\bar{y}}$. Now,

$$p_i = E(y_i) = \left(\frac{1}{1+e^{-\eta_i}}\right)^{w_1} \equiv p_i^{w_1}, \tag{5.20}$$

and hence,

$$p_i^{'} = w_1 p_i^{w_1}(1-p_i), \tag{5.21}$$

and

$$p_i^{''} = w_1 p_i^{w_1}(1-p_i)(w_1-(1+w_1)p_i). \tag{5.22}$$

Finally, the bias vector for WKLR can now be rewritten as

$$\mathbf{B}(\hat{\alpha}) = (\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}})^{-1}\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\xi, \tag{5.23}$$

where the i^{th} element of the vector ξ is now

$$\xi_i = 0.5Q_{ii}((1+w_1p_i-w_1), \tag{5.24}$$

with Q_{ii} as the diagonal elements of $\mathbf{Q}$, and $\mathbf{D} = diag(v_i w_i)$ for $i = 1...n$. The bias-corrected estimator becomes

$$\tilde{\alpha} = \hat{\alpha} - \mathbf{B}(\hat{\alpha}). \tag{5.25}$$

5.2 Iteratively Re-weighted Least Squares

For WKLR, the gradient and Hessian are obtained by differentiating the regularized weighted log-likelihood,

$$\ln \mathbb{L}_W(\alpha) = \sum_{i=1}^{n} w_i \ln \frac{e^{y_i \mathbf{k}_i \alpha}}{1+e^{\mathbf{k}_i \alpha}} - \frac{\lambda}{2}\alpha^{\mathrm{T}}\mathbf{K}\alpha, \tag{5.26}$$

with respect to α. In matrix form, the gradient is

$$\nabla_\alpha \ln \mathbb{L}_W(\alpha) = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{W}(\mathbf{y}-\mathbf{p}) - \lambda\tilde{\mathbf{K}}\alpha, \tag{5.27}$$

where $\mathbf{W} = diag(w_i)$ and $\mathbf{p}$ is the probability vector whose elements are given in (5.13). The Hessian with respect to α is then

$$\nabla^2_\alpha \ln \mathbb{L}_W(\alpha) = -\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} - \lambda\tilde{\mathbf{K}}. \tag{5.28}$$

The Newton-Raphson update with respect to α on the $(c+1)-th$ iteration is

$$\hat{\alpha}^{(c+1)} = \hat{\alpha}^{(c)} + (\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}})^{-1}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{W}(\mathbf{y}-\mathbf{p}) - \lambda\tilde{\mathbf{K}}\alpha^{(c)}). \tag{5.29}$$

Since $\hat{\alpha}^{(c)} = (\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\mathbf{K})^{-1}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\mathbf{K})\hat{\alpha}^{(c)}$, then (5.29) can be rewritten as

$$\hat{\alpha}^{(c+1)} = (\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}})^{-1}\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\mathbf{z}^{(c)}, \tag{5.30}$$

where $\mathbf{z}^{(c)} = \mathbf{K}\hat{\alpha}^{(c)} + \mathbf{D}^{-1}(\mathbf{y}-\mathbf{p})$ is the adjusted dependent variable or the adjusted response.

5.3 RE-WKLR Algorithm

For WKLR, the WLS subproblem is

$$(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}})\hat{\alpha}^{(c+1)} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\mathbf{z}^{(c)}, \tag{5.31}$$

which is a system of linear equations with a kernel matrix $\tilde{\mathbf{K}}$, vector of adjusted responses $\mathbf{z}$, and a weight matrix $\mathbf{D}$. Both the weights and the adjusted response vector are dependent on $\hat{\alpha}^{(c)}$, which is the current estimate of the parameter vector. Therefore, specifying an initial estimate $\hat{\alpha}^{(0)}$ for $\hat{\alpha}$ can be solved iteratively, giving a sequence of estimates that

converges to the MLE of $\hat{\alpha}$. That can be solved using the conjugate gradient (CG) method, which is equivalent to minimizing the quadratic problem

$$\frac{1}{2}\hat{\alpha}^{(c+1)}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}}+\lambda\tilde{\mathbf{K}})\hat{\alpha}^{(c+1)}-\hat{\alpha}^{(c+1)}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\mathbf{z}^{(c)}). \tag{5.32}$$

Similarly, the bias in (5.23) can be computed with CG by solving the quadratic problem

$$\frac{1}{2}\mathbf{B}(\hat{\alpha})^{(c+1)}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}}+\lambda\tilde{\mathbf{K}})\mathbf{B}(\hat{\alpha})^{(c+1)}-\mathbf{B}(\hat{\alpha})^{(c+1)}(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\xi^{(c)}). \tag{5.33}$$

Now, like the TR-KLR algorithm, in order to avoid the long computations that the CG may suffer from, a limit can be placed on the number of CG iterations, thus creating an approximate or truncated Newton direction.

Algorithm 5: WKLR MLE Using IRLS

Data: $\tilde{\mathbf{K}}, \mathbf{y}, \hat{\alpha}^{(0)}, w_1, w_0$
Result: $\hat{\alpha}, \mathbf{B}(\hat{\alpha}), \tilde{\alpha}, \tilde{p}_i$
begin
  $c = 0$
  while $|\frac{DEV^{(c)}-DEV^{(c+1)}}{DEV^{(c+1)}}| > \varepsilon_1$ **and** $c \leq$ *Max IRLS Iterations* **do**
    for $i \leftarrow 1$ **to** n **do**
      $\hat{p}_i = \frac{1}{1+e^{-\mathbf{k}_i\hat{\alpha}}}$; /* Compute probabilities */
      $v_i = \hat{p}_i(1-\hat{p}_i)$; /* Compute variance */
      $w_i = w_1 y_i + w_0(1-y_i)$; /* Compute weights */
      $z_i = \mathbf{k}_i\hat{\alpha}^{(c)} + \frac{(y_i-\hat{p}_i)}{\hat{p}_i(1-\hat{p}_i)}$; /* Compute adjusted response */
      $Q_{ii} = \frac{1}{v_i w_i}$; /* Compute weighted logit elements */
      $\xi_i = \frac{1}{2}Q_{ii}((1+w_1)\hat{p}_i - w_1)$; /* Compute the bias response */
    $\mathbf{D} = diag(v_i w_i)$; /* Obtain the $n \times n$ diagonal weight matrix */
    $(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}}+\lambda\tilde{\mathbf{K}})\hat{\alpha}^{(c+1)} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\mathbf{z}^{(c)}$; /* Compute $\hat{\alpha}$ via Algorithm 2 */
    $(\tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}}+\lambda\tilde{\mathbf{K}})\mathbf{B}(\hat{\alpha})^{(c+1)} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\xi^{(c)}$; /* Compute $\mathbf{B}(\hat{\alpha})$ via Algorithm 3 */
    $c = c+1$
  $\tilde{\alpha} = \hat{\alpha} - \mathbf{B}(\hat{\alpha})$; /* Compute the unbiased α */
  $\tilde{p}_i = \frac{1}{1+e^{-\tilde{\mathbf{k}}_i\tilde{\alpha}}}$; /* Compute the optimal probabilities */
end

Algorithm 5 represents the main (outer) loop of RE-WKLR, and it summarizes the

Algorithm 6: Linear CG for computing $\hat{\alpha}$. $\mathbf{A} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}}$, $\mathbf{b} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\mathbf{z}$

Data: $\mathbf{A}, \mathbf{b}, \hat{\alpha}^{(0)}$
Result: $\hat{\alpha}$ such that $\mathbf{A}\hat{\alpha} = \mathbf{b}$

begin
- $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\hat{\alpha}^{(0)}$; `/* Initialize the residual */`
- $c = 0$
- **while** $||\mathbf{r}^{(c+1)}||^2 > \varepsilon_2$ **and** $c \leq$ *Max CG Iterations* **do**
  - **if** $c = 0$ **then**
    - $\zeta^{(c)} = 0$
  - **else**
    - $\zeta^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c+1)}}{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c)}}$; `/* Update A-Conjugacy enforcer */`
  - $\mathbf{d}^{(c+1)} = \mathbf{r}^{(c+1)} + \zeta^{(c)}\mathbf{d}^{(c)}$; `/* Update the search direction */`
  - $s^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c)}\mathbf{r}^{(c)}}{\mathbf{d}^{\mathrm{T}(c)}\mathbf{A}\mathbf{d}^{c}}$; `/* Compute the optimal step length */`
  - $\hat{\alpha}^{(c+1)} = \hat{\alpha}^{(c)} + \zeta^{(c)}\mathbf{d}^{(c+1)}$; `/* Obtain an approximate solution */`
  - $\mathbf{r}^{(c+1)} = \mathbf{r}^{(c)} - s^{(c)}\mathbf{A}\mathbf{d}^{(c+1)}$; `/* Update the residual */`
  - $c = c + 1$

end

Algorithm 7: Linear CG for computing the bias. $\mathbf{A} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}}$, $\mathbf{b} = \tilde{\mathbf{K}}^{\mathrm{T}}\mathbf{D}\xi$

Data: $\mathbf{A}, \mathbf{b}, \mathbf{B}(\hat{\alpha})^{(0)}$
Result: $\mathbf{B}(\hat{\alpha})$ such that $\mathbf{A}\mathbf{B}(\hat{\alpha}) = \mathbf{b}$

begin
- $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{B}(\hat{\alpha})^{(0)}$; `/* Initialize the residual */`
- $c = 0$
- **while** $||\mathbf{r}^{(c+1)}||^2 > \varepsilon_3$ **and** $c \leq$ *Max CG Iterations* **do**
  - **if** $c = 0$ **then**
    - $\zeta^{(c)} = 0$
  - **else**
    - $\zeta^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c+1)}}{\mathbf{r}^{\mathrm{T}(c+1)}\mathbf{r}^{(c)}}$; `/* Update A-Conjugacy enforcer */`
  - $\mathbf{d}^{(c+1)} = \mathbf{r}^{(c+1)} + \zeta^{(c)}\mathbf{d}^{(c)}$; `/* Update the search direction */`
  - $s^{(c)} = \frac{\mathbf{r}^{\mathrm{T}(c)}\mathbf{r}^{(c)}}{\mathbf{d}^{\mathrm{T}(c)}\mathbf{A}\mathbf{d}^{(c)}}$; `/* Compute the optimal step length */`
  - $\mathbf{B}(\hat{\alpha})^{(c+1)} = \mathbf{B}(\hat{\alpha})^{(c)} + \zeta^{(c)}\mathbf{d}^{(c+1)}$; `/* Obtain approximate solution */`
  - $\mathbf{r}^{(c+1)} = \mathbf{r}^{(c)} - s^{(c)}\mathbf{A}\mathbf{d}^{(c+1)}$; `/* Update the residual */`
  - $c = c + 1$

end

IRLS for WKLR. The main problem to solve is the WLS in (5.31), which is a linear system of n equations and n variables. This is accomplished by Algorithm 6, which represents the inner loop that approximates the Newton direction through the linear CG method. Algorithm 7 is another linear CG for calculating the bias.

Similar to the TR-KLR formulation [10], for Algorithm 5, the maximum number of iterations is set to 30 and for the relative difference of deviance threshold, ε_1, the value 2.5 is found to be sufficient to reach the desired accuracy and at the same time maintain good convergence speed. As for Algorithms 6 and 7, the maximum number of iterations for the CG is set to 200 iterations. In addition, the CG convergence thresholds, ε_2 and ε_3, are set to 0.005 and no more than three non-improving iterations are allowed on the CG algorithm.

Chapter 6

Computational Results, Applications & Discussion

The performance of the RE-WKLR algorithm was examined using (1) seven benchmark binary class datasets (see Table 6.1) found on the UC Irvine Machine Learning Repository website [119] and (2) a real-life tornado dataset. Performance of the algorithm was then compared to that of both SVM and TR-KLR. In this analysis, the Gaussian Radial Basis Function (RBF) kernel,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = e^{(-\frac{1}{2\sigma^2}||\mathbf{x}_i - \mathbf{x}_j||)^2} = e^{(-\gamma||\mathbf{x}_i - \mathbf{x}_j||)^2}, \tag{6.1}$$

was used for all methods, where σ is the parameter of the kernel. The values of these parameters which give the best generalization were chosen from a range of different values (generally user-defined) and were tuned using the bootstrap method [126]. The bootstrap method was applied only to the testing sets. The idea behind the bootstrap method is to create hundreds or thousands of samples, called *bootstrap samples*, by re-sampling with replacement from the original sample. Each re-sample has the same size as the original sample [126].

For this study, the total number of bootstrap rounds (B) were set to 2500 rounds on all of the datasets except for the Spam and Tornado datasets. The bootstrap accuracy (A) had at most a half width of its 95% confidence interval equal to 0.25. The bootstrap sample size was chosen equal to the testing set size on all of the datasets. Due to the large size of both Spam and Tornado data, bootstrap rounds of 200 were found adequate to generate enough variations.

The overall bootstrap accuracy (A^*) was calculated according to the following. A se-

quence of sample accuracies,

$$a_1^{(\mathbf{1})}, \ldots, a_r^{(\mathbf{1})}, \ldots, a_B^{(\mathbf{1})}, a_1^{(\mathbf{0})}, \ldots, a_r^{(\mathbf{0})}, \ldots, a_B^{(\mathbf{0})},$$

was collected during the bootstrap procedure, where for a given round (r), $a_r^{(\mathbf{1})} = \frac{TP}{TP+FN}$ for class one, and $a_r^{(\mathbf{0})} = \frac{TN}{TN+FP}$ for class zero. After the bootstrap procedure was completed, the average accuracy of each class was computed. Then, for a given bootstrap procedure, the accuracy is

$$A = min\{a_{avg}^{(\mathbf{1})}, a_{avg}^{(\mathbf{0})}\}. \tag{6.2}$$

The overall accuracy reached, with different parameters, is considered to be $A^* = max\{A\}$. The interval between the 2.5^{th} and 97.5^{th} percentiles of the bootstrap distribution of a statistic is the non-parametric 95% bootstrap confidence interval. In addition, statistical significance was established using a multiple comparison paired t-test [127] single tailed with an adjusted $\alpha = 0.017$.

All of the datasets were preprocessed using normalization of a mean of zero and standard deviation of one. All of the computations for RE-WKLR and TR-KLR were carried out using MATLAB version 2007a on a 3 GiB RAM computer. As for the SVM method, MATLAB LIBSVM toolbox [121] was used.

6.1 Benchmark Datasets

The benchmark datasets were Ionosphere, Sonar, BUPA Liver Disorders, Haberman Survival, Pima Indians Diabetes, and SPECT Heart Diagnosis. The Ionosphere dataset describes radar signals targeting two types of electrons in the ionosphere: those that show some structure (good) and those that do not (bad) [128]. The Sonar dataset is composed of sonar signals detecting either mines or rocks [129]. The BUPA Liver Disorder dataset consists of blood tests that are thought to be sensitive to liver disorders that arising from

excessive alcohol consumption in single males (PC/BEAGLE User's Guide). The Haberman Survival dataset is information on the survival of patients who had undergone surgery for breast cancer [130]. The Pima Indian Diabetes dataset describes the onset of diabetes among Pima Indian patients [131]. SPECT Heart is data on cardiac Single Proton Emission Computed Tomography (SPECT) images. Each patient is classified into two categories: normal and abnormal [132]. Finally, the Spam dataset consists of email messages considered "spam," based on certain features [133].

The datasets were divided into training and testing sets. Two sampling schemes on the training datasets were applied. In the first, the training datasets were equally divided into 40 instances in each class, chosen randomly, but the same instances were applied to all of the methods. In the second scheme, the number of non-events remained 40 zeros, but the number of events was reduced to 15 instances. Due to the large size of the Spam dataset, it was treated separately, using balanced training samples of 40, 100, 200, and 300 instances, and imbalanced training samples with 40 non-events and 15 events, 100 non-events and 50 events, 200 non-events and 100 events, and 300 non-events with 150 events. To include rarity, the number of events (ones) in all the testing datasets, except for the SPECT Heart dataset, was randomly chosen and made 5% of the number of non-events (zeros). For the SPECT Heart dataset, the number of rare events remained unchanged, since the original data includes a rarity of 8% in the testing set.

Table 6.1: Datasets.

	Instances	**Features**	**Class**		**Rarity**
			0	**1**	in Testing Set
Ionosphere	351	34	225	126	5%
Sonar	208	60	111	97	5%
Liver	345	6	200	145	5%
Survival	306	3	225	81	5%
Diabetes	768	8	500	268	5%
SPECT	267	44	212	55	8%
Spam	4,601	57	2,788	1,813	5%

6.1.1 Balanced Training Data

For the balanced training dataset, Tables 6.2 and 6.3 summarize the computation results for the three methods, including their optimal parameters and accuracy, respectively. Table 6.3 and Figure 6.1 show that the RE-WKLR method scored much better than SVM and TR-KLR on all datasets except for the Ionosphere and BUPA liver data. When RE-WKLR performed better, the difference in accuracy was large, as shown with Sonar, Survival and Diabetes datasets. In addition, although TR-KLR achieved better accuracy on the Liver dataset, the difference was 2%.

A comparison of statistical significance is provided in Figure 6.2. Figure 6.2 shows the accuracy and 95% confidence level obtained by each method on the benchmark datasets. It can be observed that the accuracy of RE-WKLR is noticeably better than that of SVM on Sonar, Liver, and Survival datasets and only worse on the Ionosphere dataset. With respect to TR-KLR, the accuracy of RE-WKLR is better than that of TR-KLR on Sonar, Survival and SPECT datasets.

Table 6.2: Benchmark datasets optimal parameter values with balanced training sets.

	RE-WKLR		**SVM**		**TR-KLR**	
	σ	λ	σ	C	σ	λ
Ionosphere	2.5	0.07	3.0	10.0	6.0	0.04
Sonar	1.0	0.01	1.9	1.0	6.5	10
Liver	5.0	0.005	7.0	10.0	6.0	0.005
Survival	2.2	0.07	2.0	10.0	6.0	0.04
Diabetes	5.3	0.05	2.0	10.0	4.0	0.05
SPECT	7.4	0.7	1.3	10.0	8.0	0.001

Table 6.3: Benchmark datasets bootstrap accuracy (%) comparison using balanced training sets. Bold accuracy values indicate the highest accuracy reached by the algorithms being compared.

	RE-WKLR		SVM		TR-KLR	
Class	**0**	**1**	**0**	**1**	**0**	**1**
Ionosphere	96	89°	**94**	100	98	78
Sonar	**83**$^{\bullet}$	100	99	67	70	100
Liver	89	63$^{\diamond}$	56	75	**65**	75
Survival	**86**$^{\bullet}$	89	76	78	78	78
Diabetes	71	**70**$^{\bullet}$	68	61	73	61
SPECT	**72**$^{\bullet}$	73	69	73	51	93

$\bullet$ statistical significance using paired t-test with $\alpha = 0.017$ over both SVM and TR-KLR.
$\diamond$ statistical significance using paired t-test with $\alpha = 0.017$ over SVM.
$\circ$ statistical significance using paired t-test with $\alpha = 0.017$ over TR-KLR.

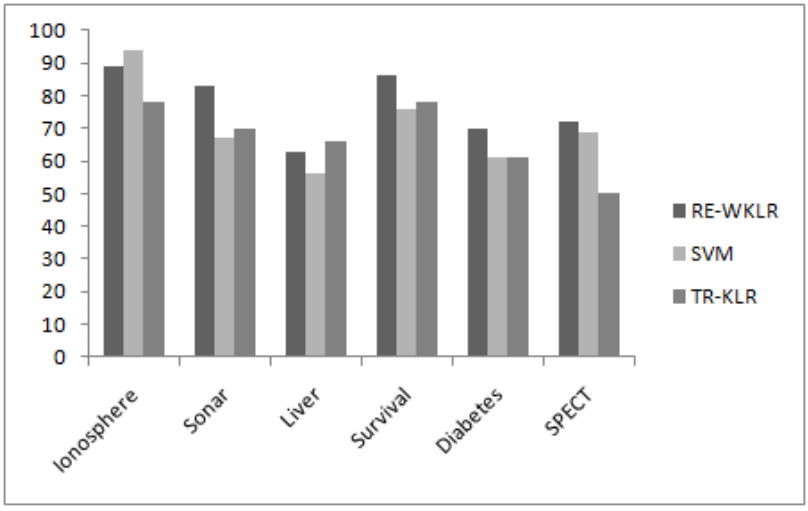

Figure 6.1: Benchmark datasets accuracy comparison with balanced training sets.

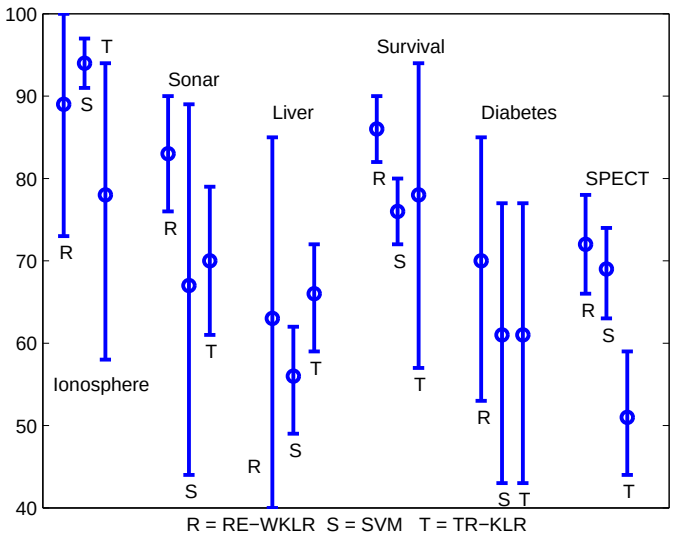

Figure 6.2: Benchmark datasets accuracy comparison using balanced training sets with 95% confidence.

6.1.2 Imbalanced Training Data

In order to appreciate the robustness and stability of RE-WKLR, the number of events in the training set was reduced to only 15 instances, again chosen randomly. It should be noted here that this is not an under-sampling scheme but rather an assumption that only 15 rare-event instances were available. Tables 6.4 and 6.5 summarize the results for these three methods with their optimal parameters and accuracy, respectively. Table 6.5 and Figure 6.3 show that RE-WKLR performed the best on all of the datasets except for Sonar and SPECT, on which it achieved equal accuracy with TR-KLR.

A comparison of statistical significance with imbalanced training data is provided in Figure 6.4. As can be observed from Figure 6.4, the accuracy of RE-WKLR is noticeably better than that of SVM on Ionosphere, Sonar, Survival, Diabetes, and SPECT datasets. With respect to TR-KLR, the accuracy of RE-WKLR is better than that of TR-KLR on Ionosphere and Survival datasets.

Except for the Ionosphere dataset, Figure 6.5 shows that despite the reduction, the RE-WKLR method retained almost the same level of accuracy as in the balanced training data, with some improvement on the Survival data. The accuracy of RE-WKLR reaches 100% on the Ionosphere dataset with the imabalanced training set. In comparison, SVM accuracy improved on the Liver data after the reduction but it became worse on both Ionosphere and Diabetes datasets. The accuracy of TR-KLR on the other hand improved on Sonar and SPECT datasets but degraded on both Ionosphere and Liver datasets.

Table 6.4: Benchmark datasets optimal parameter values with imbalanced training sets.

	RE-WKLR		**SVM**		**TR-KLR**	
	σ	λ	σ	C	σ	λ
Ionosphere	9.0	0.007	5.0	10.0	4.0	0.005
Sonar	8.0	0.01	4.0	10.0	20.0	0.005
Bupa	3.7	0.001	3.0	10.0	1.0	0.01
Haberman	2.5	0.002	1.0	100.0	1.0	0.01
Pima	2.7	0.0002	6.0	10.0	2.0	0.01
SPECT	5.7	0.09	4.0	10.0	5.1	0.3

Table 6.5: Benchmark datasets bootstrap accuracy (%) using imbalanced training sets.

	RE-WKLR		**SVM**		**TR-KLR**	
Class	**0**	**1**	**0**	**1**	**0**	**1**
Ionosphere	**100**$^{\bullet}$	**100**$^{\bullet}$	99	89	99	67
Sonar	**83**$^{\diamond}$	100	99	67	**83**	100
Liver	75	**63**$^{\bullet}$	81	62	76	50
Survival	**88**$^{\bullet}$	89	76	78	82	77
Diabetes	81	**70**$^{\bullet}$	90	52	83	61
SPECT	**74**$^{\diamond}$	80	69	73	76	73

$\bullet$ statistical significance using paired t-test with $\alpha = 0.017$ over both SVM and TR-KLR.
$\diamond$ statistical significance using paired t-test with $\alpha = 0.017$ over SVM.

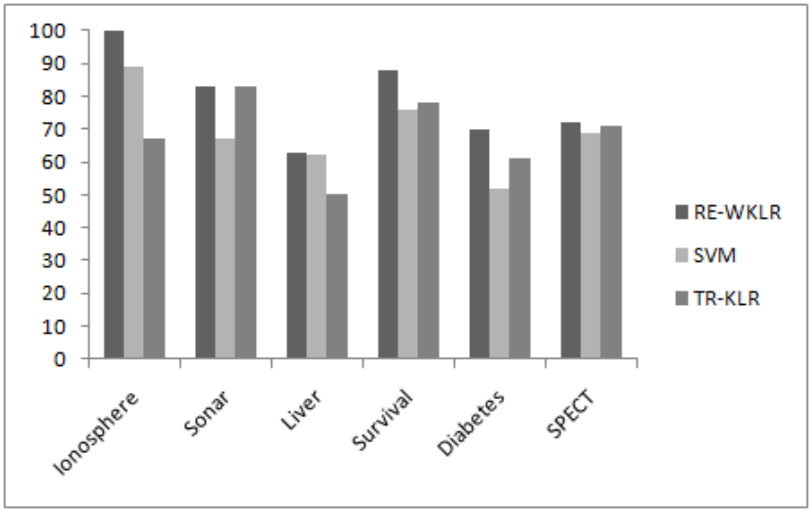

Figure 6.3: Benchmark datasets accuracy comparison with imbalanced training sets.

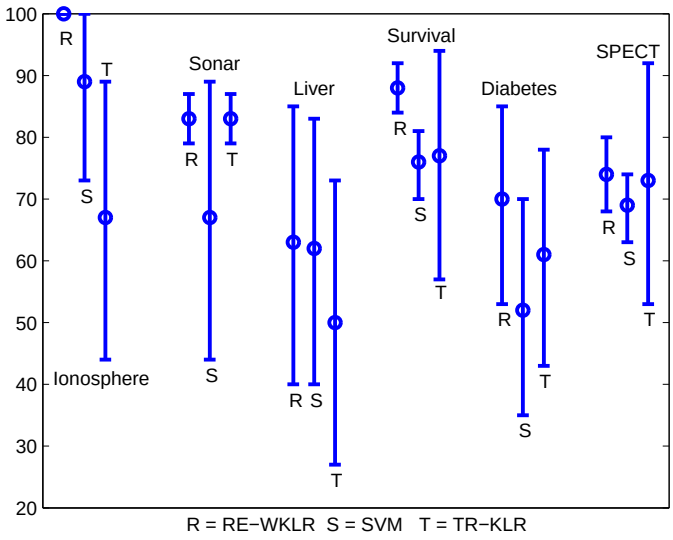

Figure 6.4: Benchmark datasets accuracy comparison using imbalanced training sets with 95% confidence level.

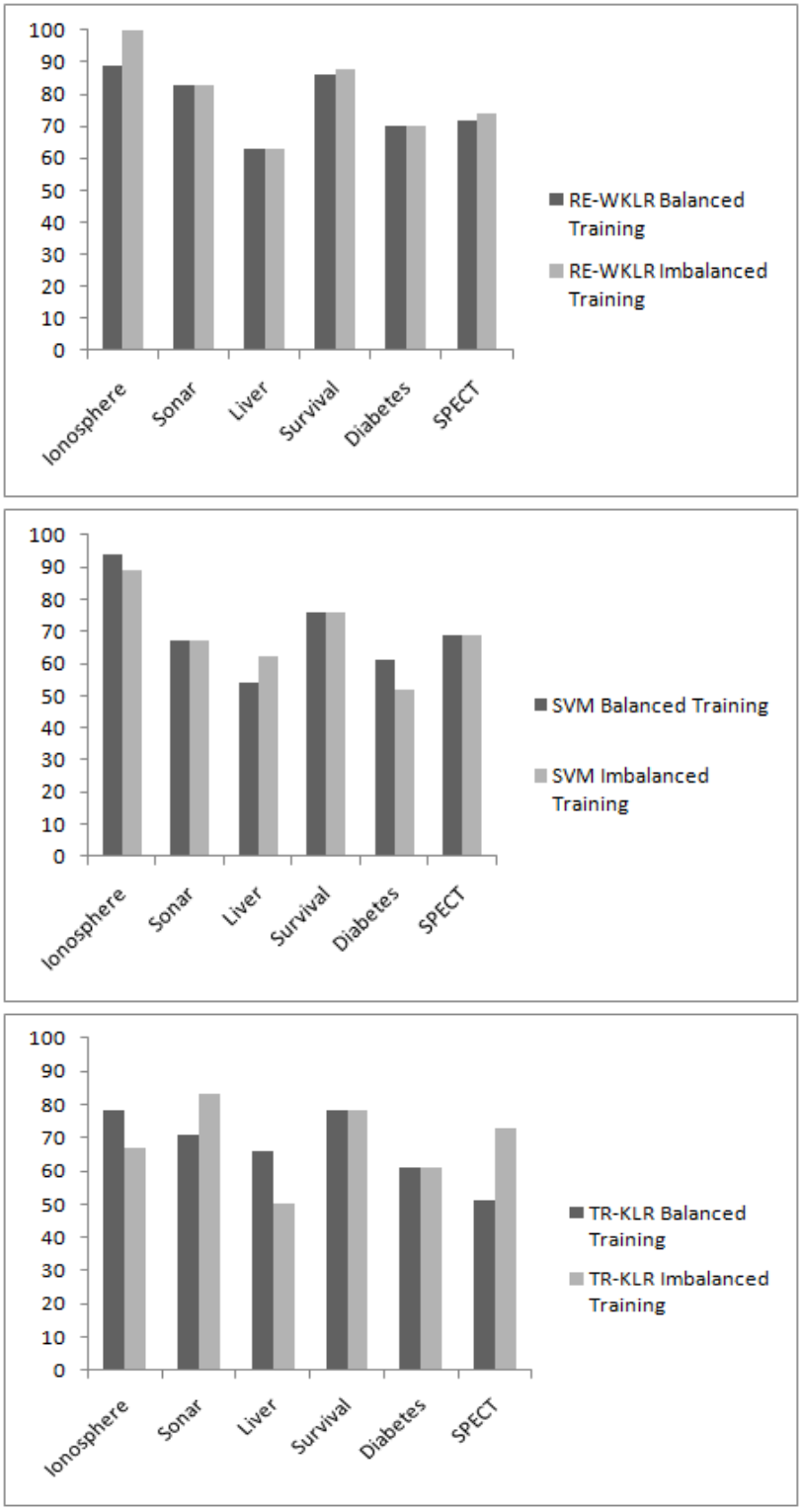

Figure 6.5: Comparison of algorithms on benchmark datasets with balanced and imbalanced training sets.

Balanced Training Set on Spam Data

For the balanced training Spam datasets, the optimal parameters and accuracies are shown in Tables 6.6 and 6.7, respectively. For samples of 40 instances, the accuracy of RE-WKLR was better than that of SVM and slightly less than that of TR-KLR, without any statistical significance. For samples of 100 instances, SVM performed slightly better than both RE-WKLR and TR-KLR, whose accuracy is equal. However, as the sample size increase to 200 and 300, the accuracy of RE-WKLR becomes noticeably greater than that of both SVM and TR-KLR, as indicated by Figures 6.6 and 6.7. Figure 6.7 shows a significant difference between the RE-WKLR accuracy and that of TR-KLR on sample size of 200 while a significant difference exists between RE-WKLR accuracy and that of SVM when the sample size is 300. In addition, notice the linear increase in the accuracy of RE-WKLR as the sample size increases, indicating consistency with ML asymptotic properties.

Table 6.6: Spam dataset optimal parameter values with balanced training datasets.

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	σ	λ	σ	C	σ	λ
40	40	6.0	0.3	7.0	1.0	5.0	0.01
100	100	7.0	0.5	18.0	10.0	5.0	0.1
200	200	7.0	0.5	18.0	100.0	7.0	0.5
300	300	3.0	1.0	18.0	10.0	7.0	0.01

Table 6.7: Spam dataset bootstrap accuracy (%) using balanced training sets.

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
40	40	$85^{\diamond}$	$85^{\diamond}$	89	83	88	**86**
100	100	89	86	92	**87**	91	86
200	200	90	**88**$^{\bullet}$	91	86	93	85
300	300	90	**89**$^{\bullet}$	94	84	93	88

$\bullet$ statistical significance using paired t-test with $\alpha = 0.017$ over both SVM and TR-KLR.
$\diamond$ statistical significance using paired t-test with $\alpha = 0.017$ over SVM.

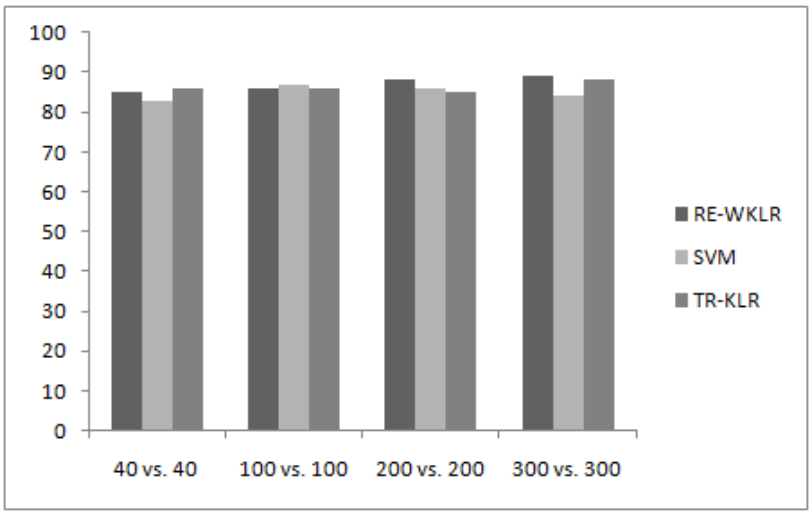

Figure 6.6: Spam dataset accuracy comparison with balanced training sets.

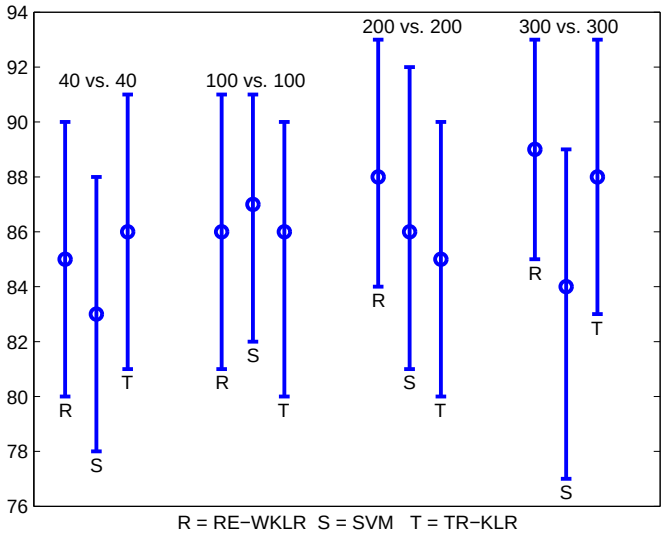

Figure 6.7: Spam dataset accuracy comparison using balanced training sets with 95% confidence level.

Imbalanced Training Set on Spam Data

Matters became different when imbalance was introduced to the training set of the Spam dataset, as shown in in Table 6.9 and Figure 6.8. The accuracies of SVM decreased with the imbalanced training data almost on all sample sizes except for a sample with 300 non-events and 150 events. The TR-KLR accuracy decreased on samples with 40 non-events and 15 events, and 100 non-events with 50 events. As shown in Figure 6.9, the accuracy of RE-WKLR is significantly higher than that of both SVM and TR-KLR. On the other hand the accuracy of RE-WKLR remained almost unchanged, as indicated by Figure 6.10.

Table 6.8: Spam dataset optimal parameter values with imbalanced training datasets.

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	σ	λ	σ	C	σ	λ
40	15	3.2	0.03	18.0	10.0	5.0	0.1
100	50	2.0	0.06	18.0	100.0	5.0	0.01
200	100	3.0	0.07	16.2	100.0	3.2	0.005
300	150	4.0	0.1	18.0	100.0	4.0	0.1

Table 6.9: Spam dataset comparison of bootstrap accuracy (%) using imbalanced training sets.

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
40	15	86	**85**$^{\bullet}$	95	70	94	75
100	50	85	**84**$^{\bullet}$	91	77	92	78
200	100	90	**89**$^{\bullet}$	93	79	91	84
300	150	91	**91**$^{\bullet}$	95	87	93	89

$\bullet$ statistical significance using paired t-test with $\alpha = 0.017$ over both SVM and TR-KLR.

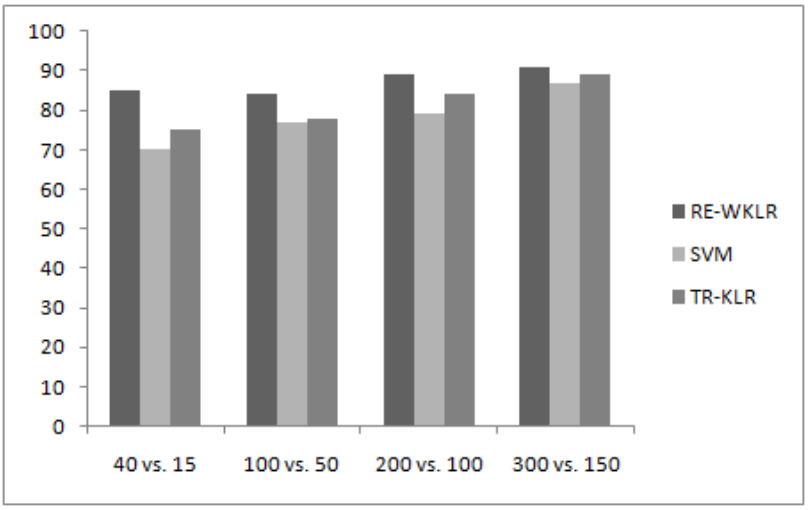

Figure 6.8: Spam dataset accuracy comparison with imbalanced training sets.

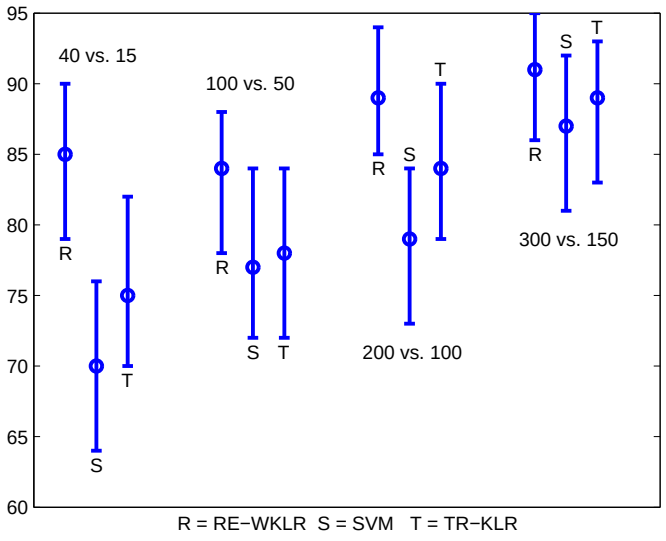

Figure 6.9: Spam dataset accuracy comparison using imbalanced training sets with 95% confidence level.

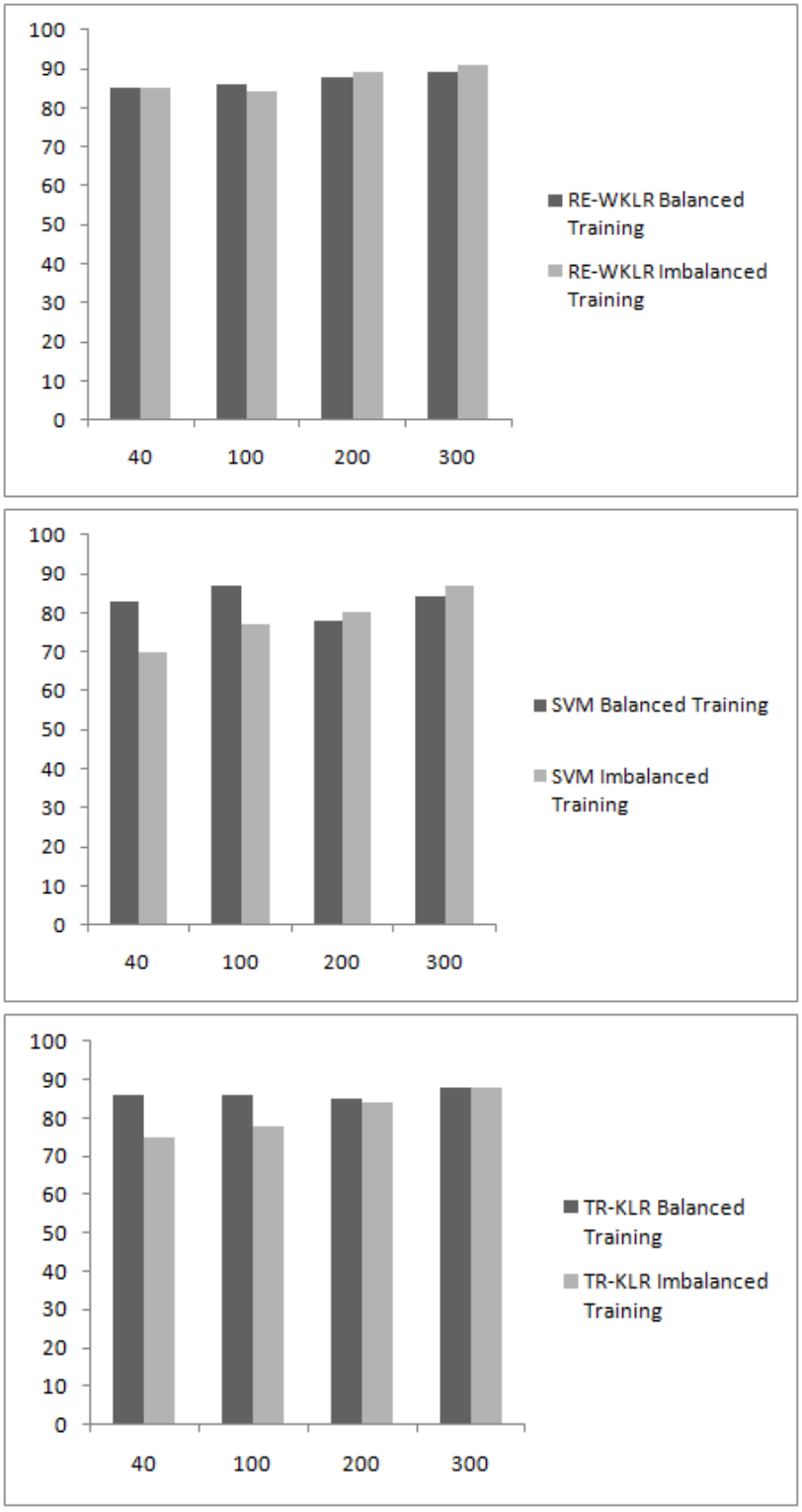

Figure 6.10: Comparison of algorithms on Spam dataset with balanced and imbalanced training sets.

6.2 RE-WKLR for Tornado Detection

The performance of RE-WKLR was evaluated on real-life tornado data and then compared to that of SVM and TR-KLR. The tornado dataset is based on the Doppler radar Mesocyclone Detection Algorithm (MDA) attributes, combined with the Near Storm Environment (NSE) dataset [134]. Application of SVM using the same dataset has been studied by Trafalis et al. [13, 135, 136], and Adrianto et al. [137] and found that SVM performed better than other methods such as Artificial Neural Networks (ANN) and Linear Discriminant Analysis (LDA).

6.2.1 Tornado Data

The Tornado dataset consists of 83 attributes, 24 of which are derived from the MDA data, measuring radar-derived *velocity* parameters that describe aspects of the Mesocyclone, in addition to the *month* attribute. The rest of the attributes are from the NSE data [134], which describes the pre-storm environment on a broader scale than MDA data. The attributes of the NSE data consist of *wind speed*, *direction*, *wind shear*, *humidity lapse rate*, and the *predisposition* of the atmosphere to explosively lift air over specific heights. In addition, the Tornado dataset consists of a training set and testing set. The training set has 387 tornado observations and 1,144 non-tornado observations. The testing set consists of 387 tornado observations and 11,872 non-tornado observations, and hence the rarity is 3%.

6.2.2 Experimental Results and Discussion

The same experimental setup used on the benchmark datasets in the previous section was also implemented on the tornado data for all algorithms. However, random under-sampling was used in the analysis. The optimal parameters and accuracies reached by all methods are shown in Tables 6.10 and 6.11, respectively. Table 6.11 shows the accuracies reached by the three methods using the original training data with 387 tornado observations and 1,144

non-tornado observations, in addition to two under-sampling schemes. In the first, the number of non-tornadoes was made to be twice the number of tornadoes, and in the second, the number of non-tornadoes was made equal to that of tornadoes, chosen randomly. In addition, the bootstrap sample size was made to consist of all of the testing set instances with 200 resampling rounds.

Table 6.10: Tornado dataset optimal parameter values.

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	σ	λ	σ	C	σ	λ
1,144	387	2.0	0.02	8.5	10.0	1.2	0.1
774	387	1.2	0.3	8.5	10.0	1.2	0.1
387	387	2.0	1.0	8.0	10.0	1.2	0.1

Table 6.11: Tornado dataset bootstrap accuracy (%).

Class Instances		**RE-WKLR**		**SVM**		**TR-KLR**	
0	**1**	**0**	**1**	**0**	**1**	**0**	**1**
1,144	387	**95**$^{\bullet}$	97	97	92	96	93
774	387	**95**	**95**	**95**	**95**	**95**	96
387	387	**95**$^{\bullet}$	95	93	98	93	98

$\bullet$ statistical significance using paired t-test with $\alpha = 0.017$ over both SVM and TR-KLR.

Table 6.12: Tornado dataset execution time in seconds.

Class Instances		**RE-WKLR**	**SVM**	**TR-KLR**
0	**1**			
1,144	387	**616**	807	632
774	387	**465**	784	482
387	387	**311**	615	322

As shown in the results, when the original dataset is used, RE-WKLR performs better than both SVM and TR-KLR. The difference between the accuracies is statistically significant as indicated Figure 6.11. However, when the non-tornado instances were reduced to be only twice the tornado instances, all three methods performed equally, with no

significant difference between accuracies. This is illustrated in Figure 6.12. Now, when the non-tornado instances were reduced further to be equal to the tornado instances, then RE-WKLR again performed better than SVM and TR-KLR, as shown in Figure 6.13. RE-WKLR maintained the same accuracy in all of the sampling schemes, while the accuracies of the other methods vary depending on the sample size and the degree of imbalanced in the training data. Assuming now that the original training data could not be reduced for some reasons, or there are no more than 387 instances of each class available for analysis, then the RE-WKLR method is the preferred choice. What is more, the computational speed of the RE-WKLR algorithm, measured by CPU time as shown in Table 6.12, is distinctly faster than that of SVM, despite the fact that LIBSVM is written mainly in C++ while both the TR-KLR and RE-WKLR algorithms are written purely in MATLAB. The time saving ranges between approximately 24% on the original training dataset and up to 50% on the equally distributed classes, as indicated by Table 6.12.

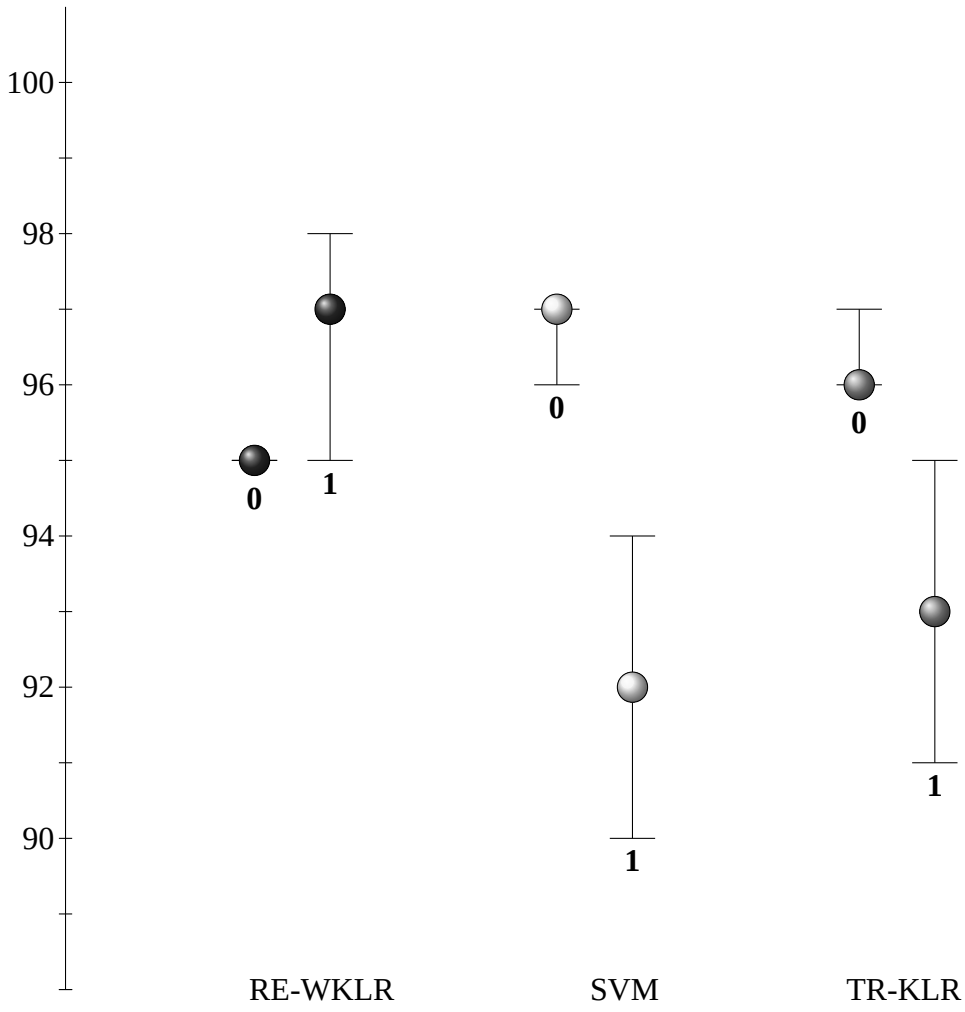

Figure 6.11: Tornado bootstrap accuracy (%) with 95 % confidence level using the original training set.

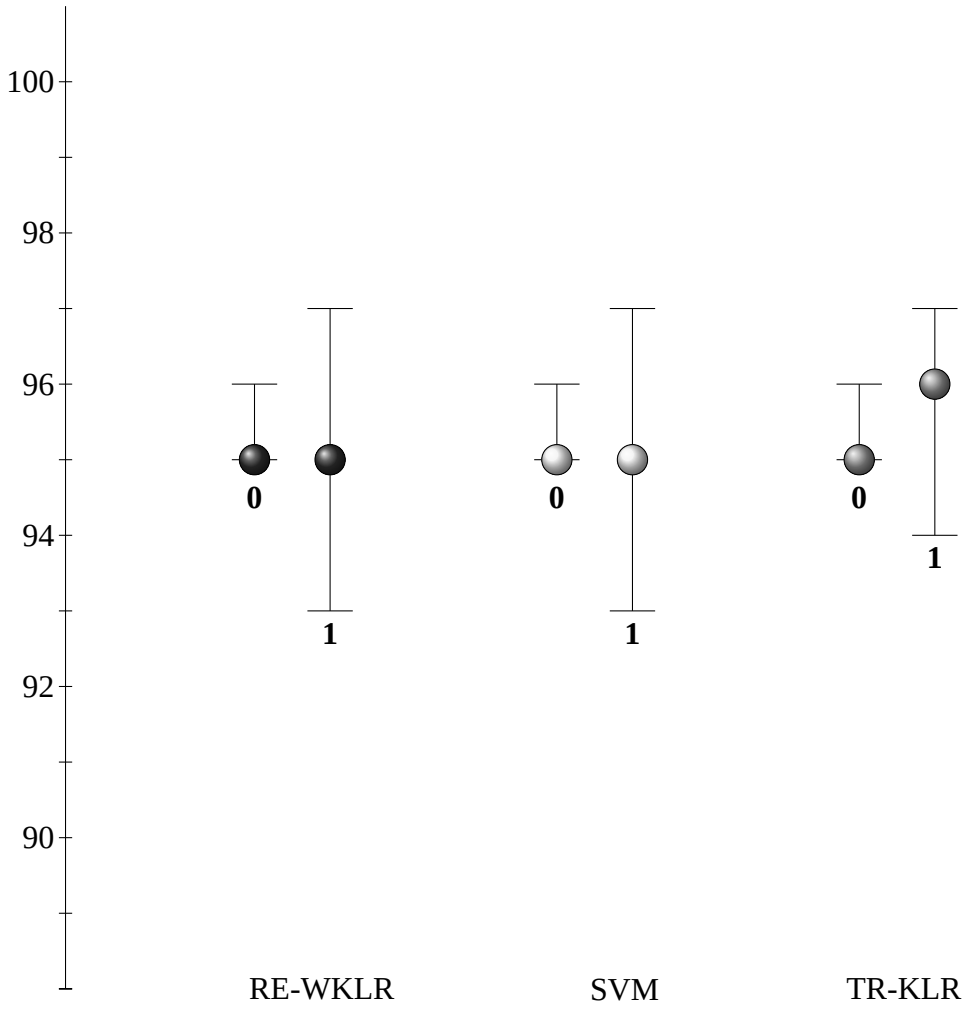

Figure 6.12: Tornado bootstrap accuracy (%) with 95 % confidence level. Training set includes 774 non-tornado instances and 387 tornado instances.

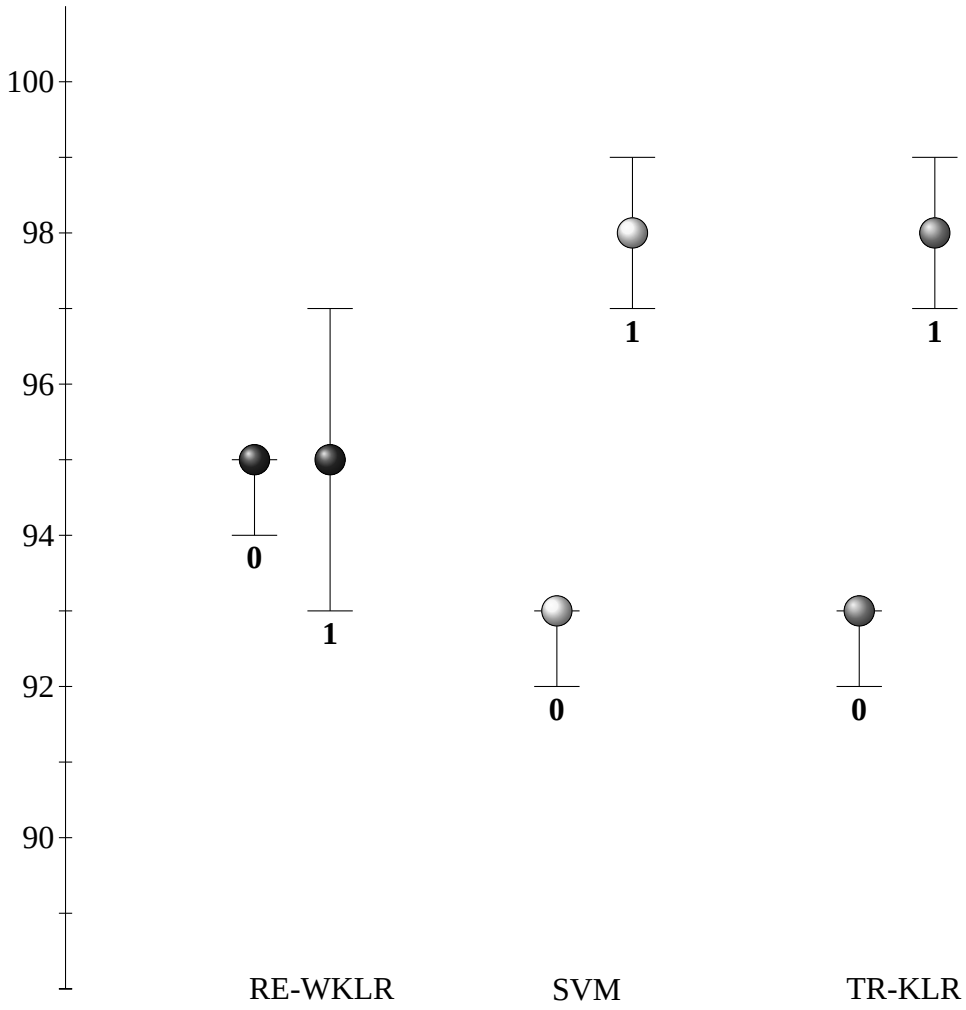

Figure 6.13: Tornado bootstrap accuracy (%) with 95 % confidence level. Training set includes 387 non-tornado instances and 387 tornado instances.

Chapter 7

Conclusion

In this study, two new powerful adaptations of classification algorithms are developed. The first is a general classification algorithm called the Truncated-Regularized Kernel Logistic Regression (TR-KLR). It is a direct adaptation of the TR-IRLS algorithm. The TR-KLR was demonstrated to be relatively easy to implement. It is mainly dependent upon the linear Conjugate Gradient (CG) method. It was shown to be as accurate as SVM when tested on twelve small-to-medium size datasets, half of which are binary-class and the other are multi-class datasets. The TR-KLR algorithm takes advantage of the speed of the TR-IRLS and the power of the kernel methods.

The second algorithm is the Rare-Event Weighted Kernel Logistic Regression (RE-WKLR). This algorithm is a further adaptation of the TR-KLR algorithm and is designed specifically for imbalanced and rare events data. It combines several concepts from the fields of statistics, econometrics and machine learning. Like TR-KLR, the RE-WKLR algorithm is relatively easy to implement and is robust when implemented on rare events and imbalanced data. It was shown that the RE-WKLR is very powerful when applied to both small and large datasets. The RE-WKLR algorithm takes advantage of bias correction and the power of the kernel methods, particularly when the data sets are neither balanced nor linearly separable.

Another benefit of RE-WKLR is that, also in common with TR-KLR, is that it uses unconstrained optimization methods whose algorithms are less complex than those with constrained optimization methods, such as SVM. As a rare-events and imbalanced data classifier, RE-WKLR is superior over both TR-KLR and SVM.

Future Work

Promising results have been demonstrated here, but future studies could lead to improved performance of the algorithms. Those studies can

- compare the methods used in this dissertation with different kernels and with different unconstrained optimization algorithms
- utilize and explore methods such as the trust-region Newton for further stability and robustness on both TR-KLR and RE-WKLR
- use intelligent sampling methods to improve the speed and accuracy of the algorithms
- compare RE-WKLR with methods developed recently on SVM, such as the Granular Support Vector Machines-Repetitive Undersampling (GSVM-RU) algorithm
- compare RE-WKLR with a kernel version of the Prior Correction method as it is more straightforward to implement
- implement feature-selection techniques, and methods such as Principal Component Analysis (PCA) to provide further data reduction
- apply RE-WKLR to imbalanced and rare-events multi-class datasets

Bibliography

[1] Hosmer, D.W., Lemeshow, S.: Applied Logistic Regression, second edn. Wiley (2000)

[2] Komarek, P.: Logistic regression for data mining and high-dimensional classification. Ph.D. thesis, Carnegie Mellon University (2004)

[3] Lin, C., Weng, R.C., Keerthi, S.S.: Trust region newton methods for large-scale logistic regression. Proceedings of the 24th International Conference on Machine Learning (2007)

[4] King, G., Zeng, L.: Logistic regression in rare events data. Political Analysis **9**, 137–163 (2001)

[5] Canu, S., Smola, A.J.: Kernel methods and the exponential family. In: ESANN, pp. 447–454 (2005)

[6] Jaakkola, T., Haussler, D.: Probabilistic kernel regression models (1999)

[7] Vapnik, V.: The Nature of Statistical Learning. Springer, NY (1995)

[8] Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Verlag (2001)

[9] Karsmakers, P., Pelckmans, K., Suykens, J.A.K.: Multi-class kernel logistic regression: a fixed-size implementation. International Joint Conference on Neural Networks pp. 1756–1761 (2007)

[10] Maalouf, M., Trafalis, T.B.: Kernel logistic regression using truncated newton method. In: C.H. Dagli, D.L. Enke, K.M. Bryden, H. Ceylan, M. Gen (eds.) Intelligent Engineering Systems Through Artificial Neural Networks, vol. 18, pp. 455–462. ASME Press, New York, NY, USA (2008)

[11] Chan, P.K., Stolfo, S.J.: Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, pp. 164–168. AAAI Press (1998)

[12] Busser, B., Daelemans, W., Bosch, A.: Machine learning of word pronunciation: the case against abstraction. In: Proceedings of the Sixth European Conference on Speech Communication and Technology, Eurospeech99, pp. 2123–2126 (1999)

[13] Trafalis, T.B., Ince, H., Richman, M.B.: Tornado detection with support vector machines. In: International Conference on Computational Science, pp. 289–298 (2003)

[14] Weiss, G.M., Hirsh, H.: Learning to predict extremely rare events. In: AAAI Workshop on Learning from Imbalanced Data Sets, pp. 64–68. AAAI Press (2000)

[15] Kubat, M., Holte, R.C., Matwin, S.: Machine learning for the detection of oil spills in satellite radar images. In: Machine Learning, pp. 195–215 (1998)

[16] King, G., Zeng, L.: Explaining rare events in international relations. International Organization **55**(3), 693–715 (2001)

[17] King, G., Zeng, L.: Improving forecast of state failure. World Politics **53**(4), 623–658 (2001)

[18] Eeckhaut, M.V.D., Vanwalleghem, T., Poesen, J., Govers, G., Verstraeten, G., Vandekerckhove, L.: Prediction of landslide susceptibility using rare events logistic regression: A case-study in the flemish ardennes (belgium). Geomorphology **76**(3-4), 392 – 410 (2006)

[19] Bai, S.B., Wang, J., Zhang, F.Y., Pozdnoukhov, A., Kanevski, M.: Prediction of landslide susceptibility using logistic regression: A case study in bailongjiang river basin, china. Fuzzy Systems and Knowledge Discovery, Fourth International Conference on **4**, 647–651 (2008)

[20] Quigley, J., Bedford, T., Walls, L.: Estimating rate of occurrence of rare events with empirical bayes: A railway application. Reliability Engineering & System Safety **92**(5), 619 – 627 (2007)

[21] Tsoucas, P.: Rare events in series of queues. Journal of Applied probability **29**, 168–175 (1992)

[22] Weiss, G.M.: Mining with rarity: a unifying framework. SIGKDD Explorations Newsletter **6**(1), 7–19 (2004)

[23] Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports. In: Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 337–341 (1999)

[24] Weiss, G.M., Provost, F.: Learning when training data are costly: the effect of class distribution on tree induction. Journal of Artificial Intelligence Research **19**, 315–354 (2003)

[25] Visa, S., Ralescu, A.: Issues in mining imbalanced data sets - a review paper. In: Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference, 2005, pp. 67–73 (2005)

[26] Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: ICML '04: Proceedings of the twenty-first international conference on Machine learning, p. 114. ACM, New York, NY, USA (2004)

[27] Fan, W., Davidson, I.: On sample selection bias and its efficient correction via model averaging and unlabeled examples. In: SDM (2007)

[28] Baum, E.B.: What is Thought? The MIT Press (2004)

[29] Chen, Z.: Computational Intelligence for Decision Support. CRC Press (1999)

[30] Alpaydin, E.: Introduction to Machine Learning. The MIT Press (2004)

[31] Holte, R., Acker, L.E., Porter, B.W.: Concept learning and the problem of small disjuncts. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 813–818 (1989)

[32] Jo, T., Japkowicz, N.: Class imbalances versus small disjuncts. SIGKDD Explor. Newsl. **6**(1), 40–49 (2004)

[33] Weiss, G.M.: Learning with rare cases and small disjuncts. In: In Proceedings of the Twelfth International Conference on Machine Learning, pp. 558–565. Morgan Kaufmann (1995)

[34] Weiss, G.M., Hirsh, H.: A quantitative study of small disjuncts. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, pp. 665–670. AAAI Press/ The MIT Press (2000)

[35] Friedman, J.H., Kohavi, R., Yun, Y.: Lazy decision trees. In: AAAI/IAAI, Vol. 1, pp. 717–724 (1996)

[36] Tan, P., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley (2005)

[37] Schaefer, J.T.: The critical success index as an indicator of warning skill. American Meteorological Society (AMS) **5**(4), 570–575 (1990)

[38] Hoehler, F.K.: Bias and prevalence effects on kappa viewed in terms of sensitivity and specificity. Journal of Clinical Epidemiology **53**, 499–503 (2000)

[39] Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006)

[40] Provost, F.: Machine learning from imbalanced data sets 101. In: In Proceedings of AAAI Workshop on Imbalanced Data Sets (2000)

[41] Kotsiantis, S., Kanellopoulos, D., Pintelas, P.: Handling imabalanced datasets: A review. GESTS International Transactions on Computer Science and Engineering **30**, 25–36 (2006)

[42] Japkowicz, N., Myers, C., Gluck, M.: A novelty detection approach to classification. In: Proceedings of the Fourteenth Joint Conference on Artificial Intelligence, pp. 518–523 (1995)

[43] Cohen, W.W.: Fast effective rule induction. In: Proceedings of the Twelfth International Conference on Machine Learning, pp. 115–123. Morgan Kaufmann (1995)

[44] Pazzani, M.J., Merz, C.J., Murphy, P.M., Ali, K., Hume, T., Brunk, C.: Reducing misclassification costs. In: ICML, pp. 217–225 (1994)

[45] Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining, pp. 155–164. ACM Press (1999)

[46] Chen, C., Liaw, A., Breiman, L.: Using random forest to learn imbalanced data. Tech. rep., University of California at Berkeley: Statistics Department (2004)

[47] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)

[48] Freund, Y., Schapire, R.E.: A brief introduction to boosting. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, pp. 1401–1406. Morgan Kaufmann (1999)

[49] Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K.: Adacost: misclassification cost-sensitive boosting. In: Prococeedings to the 16th International Conference on Machine Learning, pp. 97–105. Morgan Kaufmann (1999)

[50] Joshi, M.V., Kumar, V., Agarwal, R.C.: Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In: ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining, pp. 257–264. Washington, DC, USA (2001)

[51] Joshi, M.V., Agarwal, R.C., Kumar, V.: Predicting rare classes: can boosting make any weak learner strong? In: KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 297–306. ACM, New York, NY, USA (2002)

[52] Chawla, N.: C4.5 and imabalanced data sets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In: Workshop on Learning from Imbalanced Datasets II, ICML, Wahington D.C. (2003)

[53] Maloof, M.A.: Learning when data sets are imbalanced and when costs are unequal and unknown. In: ICML-2003 Workshop on Learning from Imbalanced Data Sets II (2003)

[54] Weiss, G.M., McCarthy, K., Zabar, B.: Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? In: International Conference on Data Mining (DMIN), pp. 35–41 (2007)

[55] Weiss, G.M., Hirsh, H.: Learning to predict rare events in event sequences. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, pp. 359–363. AAAI Press (1998)

[56] Joshi, M.V., Agarwal, R.C., Kumar, V.: Mining needle in a haystack: classifying rare classes via two-phase rule induction. In: SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, pp. 91–102. ACM, New York, NY, USA (2001)

[57] Lanckriet, G.R.G., Ghaoui, L.E., Bhattacharyya, C., Jordan, M.I.: Minimax probability machine. In: Advances in Neural Information Processing Systems 14. MIT Press (2002)

[58] Huang, K., Yang, H., King, I., Lyu, M.R.: Learning classifiers from imbalanced data based on biased minimax probability machine. In: CVPR (2), pp. 558–563 (2004)

[59] Zheng, Z., Wu, X., Srihari, R.: Feature selection for text categorization on imbalanced data. SIGKDD Explor. Newsl. **6**(1), 80–89 (2004)

[60] Hulse, J.V., Khoshgoftaar, T.M., Napolitano, A.: Experimental perspectives on learning from imbalanced data. In: ICML '07: Proceedings of the 24th international conference on Machine learning, pp. 935–942. ACM, New York, NY, USA (2007)

[61] Seiffert, C., Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A.: Mining data with rare events: A case study. In: ICTAI '07: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence - Vol.2 (ICTAI 2007), pp. 132–139. IEEE Computer Society, Washington, DC, USA (2007)

[62] Drummond, C., Holte, R.C.: C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. pp. 1–8 (2003)

[63] Japkowicz, N.: Learning from imbalanced data sets: A comparison of various strategies. Tech. rep., University of DalTech/Dalhousie (2000)

[64] Prati, R.C., Batista, G.E.A.P.A., Monard, M.C.: Learning with class skews and small disjuncts. In: SBIA, pp. 296–306 (2004)

[65] Barandela, R., Valdovinos, R.M., Sánchez, J.S., Ferri, F.J.: The imbalance training sample problem: under or over sampling. In: Structural, Syntactic, and Statistical Pattern Recognition, pp. 806–814. Springer-Verlag (2004)

[66] Han, H., Wang, W., Mao, B.H.: Borderline-smote: A new over-sampling method in imbalanced data sets learning. In: ICIC (1), pp. 878–887 (2005)

[67] García, V., Alejo, R., Sánchez, J.S., Sotoca, J.M., Mollineda, R.A.: Combined effects of class imbalance and class overlap on instance-based classification. In: Intelligent Data Engineering and Automated Learning (IDEAL), pp. 371–378 (2006)

[68] Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W.: SMOTEBoost: improving prediction of the minority class in boosting. In: Proceedings of the Principles of Knowledge Discovery in Databases, PKDD-2003, pp. 107–119 (2003)

[69] Ertekin, S., Huang, J., Bottou, L., Giles, C.L.: Learning on the border: active learning in imbalanced data classification. In: M.J. Silva, A.H.F. Laender, R.A. Baeza-Yates, D.L. McGuinness, B. Olstad, Øystein Haug Olsen, A.O. Falcão (eds.) CIKM, pp. 127–136. ACM (2007)

[70] Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: one-sided selection. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 179–186. Morgan Kaufmann (1997)

[71] Laurikkala, J.: Improving identification of difficult small classes by balancing class distribution. In: AIME '01: Proceedings of the 8th Conference on AI in Medicine in Europe, pp. 63–66. Springer-Verlag, London, UK (2001)

[72] Li, J.Q., Smith, D.R., Barford, L.A., Heumann, J.M.: Classification of rare events with high reliability. United States Patent Application 20030204507 (2003)

[73] Wu, J., Xiong, H., Wu, P., Chen, J.: Local decomposition for rare class analysis. In: KDD '07: Proceedings of the 13th ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining, pp. 814–823. ACM, New York, NY, USA (2007)

[74] Akbani, R., Kwek, S., Japkowicz, N.: Applying support vector machines to imabalanced datasets. Lecture Notes in Computer Science **3201**, 39–50 (2004)

[75] Wu, G., Chang, E.: Class-boundary alignment for imbalanced data sets. In: International Conference of Data Mining (ICDM), Workshop Learning from Imabalanced Data Sets II (2003)

[76] He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering **21**(9), 1263–1284 (2009)

[77] Yan, R., Liu, Y., Jin, R., Hauptmann, A.: On predicting rare classes with svm ensembles in scene classification. In: IEEE International Conference on Acoustic, Speech and Signal Processing, pp. 21–24 (2003)

[78] Tang, Y., Zhang, Y.Q., Chawla, N.V., Krasser, S.: SVMs modeling for highly imbalanced classification. IEEE Transactions on Systems, Man, and Cybernectics, Part B **39**(1), 281–288 (2009)

[79] Tang, Y., Jin, B., Zhang, Y.Q.: Granular support vector machines with association rules mining for protein homology prediction. Artificial Intelligence in Medicine **35**(1-2), 121–134 (2005)

[80] Gu, Q., Cai, Z., Zhu, L., Huang, B.: Data mining on imbalanced data sets. International Conference on Advanced Computer Theory and Engineering, **0**, 1020–1024 (2008)

[81] Garthwaite, P., Jolliffe, I., Byron, J.: Statistical Inference. Oxford University Press (2002)

[82] Amemiya, T.: Advanced Econometrics. Harvard University Press (1985)

[83] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)

[84] Malouf, R.: A comparison of algorithms for maximum etropy parameter estimation. In: Proceedings of Conference on Natural Language Learning (volume 6, 2002)

[85] Minka, T.P.: A comparison of numerical optimizers for logistic regression. Tech. rep., Deptartment of Statistics, Carnegie Mellon University (2003)

[86] Lewis, J.M., Lakshmivarahan, S., Dhall, S.: Dynamic Data Assimilation: A Least Squares Approach. Cambridge University Press (2006)

[87] Komarek, P., Moore, A.: Making logistic regression a core data mining tool: A practical investigation of accuracy, speed, and simplicity. Tech. rep., Carnegie Mellon University (2005)

[88] Xie, Y., Manski, C.F.: The logit model and response-based samples. Sociological Methods & Research **17**, 283–302 (1989)

[89] Cameron, A.C., Trivedi, P.K.: Microeconometrics: Methods and Applications. Cambridge University Press (2005)

[90] Collett, D.: Modelling Binary Data, 2nd edn. Chapman & Hall/CRC (2003)

[91] Cox, D.R., Hinkley, D.V.: Theoretical Statistics. Chapman & Hall/CRC (1979)

[92] Cox, D.R., Snell, E.J.: A general definition of residuals. Journal of the Royal Statistical Society **30**(2), 248–275 (1968)

[93] Palepu, K.: Predicting takeover targets: A methodological and empirical analysis. Journal of Accounting and Economic **8**, 3–35 (1986)

[94] Cramer, J.S.: Logit Models From Economics And Other Fields. Cambridge University Press (2003)

[95] Milgate, M., Eatwell, J., Newman, P.K. (eds.): Econometrics. W. W. Norton & Company (1990)

[96] Manski, C.F., Lerman, S.R.: The estimation of choice probabilities from choice based samples. Econometrica **45**(8), 1977–88 (1977)

[97] Imbens, G.W., Lancaster, T.: Efficient estimation and stratified sampling. Journal of Econometrics **74**, 289–318 (1996)

[98] Ben-Akiva, M., Lerman, S.: Discrete Choice Analysis: Theory and Application to Travel Demand. The MIT Press (1985)

[99] Ramalho, E.A., Ramalho, J.J.S.: On the weighted maximum likelihood estimator for endogenous stratified samples when the population strata probabilities are unknown. Applied Economics Letters **14**, 171–174 (2007)

[100] McCullagh, P., Nelder, J.: Generalized Linear Model. Chapman and Hall/CRC (1989)

[101] Cordeiro, G.M., McCullagh, P.: Bias correction in generalized linear models. Journal of Royal Statistical Society **53**(3), 629–643 (1991)

[102] Heinze, G., Schemper, M.: A solution to the problem of monotone likelihood in cox regression. Biometrics **57**, 114–119 (2001)

[103] Wang, S., Wang, T.: Precision of warm's weighted likelihood for a polytomous model in computerized adaptive testing. Applied Psychological Measurement **25**(4), 317–331 (2001)

[104] Firth, D.: Bias reduction of maximum likelihood estimates. Biometrika **80**, 27–38 (1993)

[105] Maiti, T., Pradhan, V.: A comparative study of the bias corrected estimates in logistic regression. Statistical Methods in Medical Research **17**(6), 621–634 (2008)

[106] Cordeiro, G., Barroso, L.: A third-order bias corrected estimate in generalized linear models. TEST: An Official Journal of the Spanish Society of Statistics and Operations Research **16**(1), 76–89 (2007)

[107] Cosslett, S.R.: Structural Analysis of Discrete Data and Econometric Applications. Cambridge: The MIT Press (1981)

[108] Imbens, G.W.: An efficient method of moments estimator for discrete choice models with choice-based sampling. Econometrica **60**(5), 1187–214 (1992)

[109] Koh, K., Kim, S., Boyd, S.: An interior-point method for large-scale ℓ_1-regularized logistic regression. Journal of Machine Learning Research **8**, 1519–1555 (2007)

[110] Zhu, J., Hastie, T.: Kernel logistic regression and the import vector machine. Journal of Computational and Graphical Statistics **14**, 185–205 (2005)

[111] Yu, Q.H.D., Daren, Xie, Z.: Selecting samples and features for SVM based on neighborhood model. In: RSFDGrC '07: Proceedings of the 11th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, pp. 508–517 (2007)

[112] Roth, V.: Probabilistic discriminative kernel classifiers for multi-class problems. In: Proceedings of the 23rd DAGM-Symposium on Pattern Recognition, pp. 246–253. Springer-Verlag, London, UK (2001)

[113] Keerthi, S.S., Duan, K.B., Shevade, S.K., Poo, A.N.: A fast dual algorithm for kernel logistic regression. Mach. Learn. **61**(1-3), 151–165 (2005)

[114] Rifkin, R., Klautau, A.: In defense of one-vs-all classification. Journal of Machine Learning Research **5**, 101–141 (2004)

[115] Kressel, U.H.G.: Pairwise classification and support vector machines. In: Advances in kernel methods: support vector learning, pp. 255–268. MIT Press, Cambridge, MA, USA (1999)

[116] Platt, J.C., Cristianini, N., Shawe-taylor, J.: Large margin DAGs for multiclass classification. In: Advances in Neural Information Processing Systems, pp. 547–553. MIT Press (2000)

[117] Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press (2000)

[118] Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)

[119] Asuncion, A., Newman, D.J.: UCI machine learning repository. university of california, irvine, school of information and computer sciences. http://www.ics.uci.edu/∼mlearn/MLRepository.html (2007)

[120] Gunn, S.R.: MATLAB support vector machine toolbox. (Internet) (1998). URL http://www.isis.ecs.soton.ac.uk/isystems/kernel

[121] Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm

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

[123] Cowan, G.: Statistical Data Analysis. Oxford University Press (1998)

[124] Maimon, O., Rokach, L. (eds.): Data Mining and Knowledge Discovery Handbook. Springer (2005)

[125] Berk, R.: Statistical Learning from a Regression Perspective, 1st edn. Springer (2008)

[126] Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. Chapman & Hall/CRC (1994)

[127] Jensen, D., Cohen, P.R.: Multiple comparison in induction algorithms. Machine Learning **38**, 309–338 (2000)

[128] Sigillito, V.G., Wing, S.P., Hutton, L.V., Baker, K.B.: Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Technical Digest **10**, 262–266 (1989)

[129] Gorman, R.P., Sejnowski, T.J.: Analysis of hidden units in a layered network trained to classify sonar targets. Neural Networks **1**, 75–89 (1988)

[130] Haberman, S.J.: Generalized residuals for log-linear models. In: Proceedings of the 9th International Biometrics Conference, Boston (1976)

[131] Smith, J.W., Everhart, J.E., Dickson, W.C., Johannes, R.S.: Using the adap learning algorithm to forecast the onset of diabetes mellitus. In: Proceedings of the Symposium on Computer Applications and Medical Care. IEEE Computer Society Press (1988)

[132] Kurgan, L.A., Cios, K.J., Tadeusiewicz, R., Ogiela, M., Goodenday, L.S.: Knowledge discovery approach to automated cardiac spect diagnosis. Artificial Intelligence in Medicine **32**(2), 149–169 (2001)

[133] Wang, Y., Witten, I.H.: Modeling for optimal probability prediction. In: ICML, pp. 650–657 (2002)

[134] Lakshmanan, V., Stumpf, G., Witts, A.: A neural network for detecting and diagnosing tornadic circulations using the mesocyclone detection and near storm environment algorithms. In: 21st International Conference on Information Processing Systems, San Diego, CA, American Meteorological Society, CD-ROM, J52.2 (2005)

[135] Trafalis, T.B., Santosa, B., Richman, M.B.: Bayesian neural networks for tornado detection. WSEAS Transactions on Systems **3**(10), 3211–3216 (2004)

[136] Trafalis, T.B., Santosa, B., Richman, M.B.: Learning networks in rainfall estimation. Computational Management Science **2**(3), 229–251 (2005)

[137] Adrianto, I., Trafalis, T.B., Richman, M.B., Lakshmivarahan, S., Park, J.: Machine learning classifiers for tornado detection: sensitivity analysis on tornado data sets. In: C.H. Dagli, D.L. Enke, K.M. Bryden, H. Ceylan, M. Gen (eds.) Intelligent Engineering Systems Through Artificial Neural Networks, vol. 16, pp. 679–684. ASME Press, New York, NY, USA (2008)

[138] Dekking, F.M., Kraaikamp, C., Lopuhaä, H.P., Meester, L.E.: A Modern Introduction to Probability and Statistics: Understanding Why and How. Springer (2007)

[139] Sorensen, D., Gianola, D.: Likelihood, Bayesian, and MCMC Methods in Quantitative Genetics. Springer (2007)

[140] Greene, W.H.: Econometric Analysis, 3rd edn. London, Prentice Hall (1997)

[141] Dobson, A.J., Barnett, A.G.: An Introduction to Generalized Linear Models. Chapman & Hall/CRC (2008)

[142] Pawitan, Y.: In All Likelihood. Oxford University Press (2001)

[143] Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery **2**, 121–167 (1998)

[144] Negoita, M.G., Reusch, B. (eds.): Real World Applications of Computational Intelligence. Springer (2005)

Appendix A

Maximum Likelihood Estimation

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be a data matrix, where n is the number of instances (examples) and d is the number of parameters, and an outcomes vector $\mathbf{y} \in \mathbb{R}$, or $\mathbf{y} \in \{0, 1\}$. The objective is to find an estimator, $\hat{\theta}$, for some unknown true parameter, θ, that would maximize the likelihood of observing the outcomes. This is the *principle of maximum likelihood* [138].

The joint probability density function, or the joint probability mass function, is then a function of θ, given the data $(\mathbf{X}, \mathbf{y})$. This function is called the *likelihood function*, and is denoted by $\mathbb{L}(\theta|\mathbf{X}, \mathbf{y})$. From Bayesian statistics, the likelihood can be expressed as

$$\mathbb{L}(\theta) = f(\mathbf{y}, \mathbf{X}|\theta) = P(\mathbf{y}|\mathbf{X}, \theta)P(\mathbf{X}), \tag{A.1}$$

where $P(\mathbf{y}|\mathbf{X}, \theta)$ is the conditional density of $\mathbf{y}$ given the data $\mathbf{X}$, and $P(\mathbf{X})$ is the marginal density of $\mathbf{X}$. Since the objective is to model the behavior of $\mathbf{y}$ by finding an estimate of θ which maximizes $\mathbb{L}(\theta)$, then the last term, $P(\mathbf{X})$, can be dropped without affecting the likelihood model. Hence, the likelihood is usually given as

$$\mathbb{L}(\theta) = P(\mathbf{y}|\mathbf{X}, \theta). \tag{A.2}$$

Maximizing the likelihood function is equivalent to maximizing the natural logarithm of the likelihood (log-likelihood), such that

$$\mathbb{L}(\theta) = \ln \mathbb{L}(\theta). \tag{A.3}$$

The log-likelihood function is usually analyzed and calculated because mathematically it is a monotonic function and the same value $\hat{\theta}$ maximizes both $\mathbb{L}(\theta)$ and $\ln\mathbb{L}(\theta)$.

A.1 Asymptotic Properties of Maximum Likelihood Estimators

The *maximum likelihood (ML) estimators* are *extremum* estimators that maximize the likelihood function. The ML estimators have some specific properties. There are four important properties for the ML estimators:

A.1.1 Asymptotic Consistency

Subject to some weak regularity conditions, ML estimators are consistent. The property of consistency essentially means that as the sample size increases, the expected value (mean) of the ML estimator, $\hat{\theta}$, gets closer in value to the true unknown population parameter θ. Mathematically [139],

$$\lim_{n\to\infty} \Pr(|\hat{\theta}_n - \theta| < \varepsilon) = 1. \tag{A.4}$$

In other words, as the sample size increases, the ML estimator lies within a small interval, $\varepsilon > 0$, of the true parameter θ with certainty (a probability of 1).

The Expected Score

Following Cameron and Trivedi [89], an essential consistency regularity condition is that

$$\mathrm{E}\left[\nabla_{\hat{\theta}} P(\mathbf{y}|\mathbf{X},\hat{\theta})\right]_{\hat{\theta}=\theta} = \mathbf{0}, \tag{A.5}$$

Proof:

$$\int P(\mathbf{y}|\mathbf{X},\hat{\theta})d\mathbf{y} = \mathbf{1}. \tag{A.6}$$

Differentiating both sides with respect to $\hat{\theta}$ yields

$$\nabla_{\hat{\theta}} \int P(\mathbf{y}|\mathbf{X}, \hat{\theta}) d\mathbf{y} = \mathbf{0}, \tag{A.7}$$

which implies that

$$\int \nabla_{\hat{\theta}} P(\mathbf{y}|\mathbf{X}, \hat{\theta}) d\mathbf{y} = \mathbf{0}, \tag{A.8}$$

if the range of $\mathbf{y}$ does not depend on $\hat{\theta}$. Now,

$$\nabla_{\hat{\theta}} \ln P(\mathbf{y}|\mathbf{X}, \hat{\theta}) = \nabla_{\hat{\theta}} P(\mathbf{y}|\mathbf{X}, \hat{\theta}) (P(\mathbf{y}|\mathbf{X}, \hat{\theta}))^{-1} \tag{A.9}$$

which implies that

$$\nabla_{\hat{\theta}} P(\mathbf{y}|\mathbf{X}, \hat{\theta}) = \nabla_{\hat{\theta}} \ln P(\mathbf{y}|\mathbf{X}, \hat{\theta}) P(\mathbf{y}|\mathbf{X}, \hat{\theta}), \tag{A.10}$$

Substituting now yields

$$\int \nabla_{\hat{\theta}} \ln P(\mathbf{y}|\mathbf{X}, \hat{\theta}) P(\mathbf{y}|\mathbf{X}, \hat{\theta}) d\mathbf{y} = \mathbf{0}, \tag{A.11}$$

which means that

$$\mathrm{E}\left[\nabla_{\hat{\theta}} \ln P(\mathbf{y}|\mathbf{X}, \hat{\theta})\right]_{\hat{\theta}=\theta} \quad = \quad \mathbf{g}(\hat{\theta}) = \int \nabla_{\hat{\theta}} \ln P(\mathbf{y}|\mathbf{X}, \hat{\theta}) P(\mathbf{y}|\mathbf{X}, \hat{\theta}) d\mathbf{y} = \mathbf{0}, \tag{A.12}$$

provided that the expectation is with respect to $P(\mathbf{y}|\mathbf{X}, \theta)$. What this essentially implies is that by the law of large numbers, the sample score converges in probability to its expected value as the sample size increases. The ML estimator is an extremum, and in the limit, the expected score is equal to zero. Since the expected score is equal to zero only at the true parameter θ, then in the limit, $\hat{\theta} = \theta$.

A.1.2 Asymptotic Normality

The normality property states that for large samples, ML estimators are normally distributed. Following Greene [140], consider a first-order Taylor series expansion of the score, $\mathbf{g}(\hat{\theta})$, around the true parameter vector, θ, then,

$$\mathbf{g}(\hat{\theta}) = \mathbf{g}(\theta) + \mathbf{H}(\theta)(\hat{\theta} - \theta), \tag{A.13}$$

where $\mathbf{H}(\theta) = \nabla^2_\theta \ln P(\mathbf{y}|\mathbf{X}, \theta)$ is the Hessian matrix. Now, from the consistency property, $\mathbf{g}(\hat{\theta}) = \mathbf{0}$, hence,

$$\mathbf{0} = \mathbf{g}(\theta) + \mathbf{H}(\theta)(\hat{\theta} - \theta), \tag{A.14}$$

and therefore,

$$\hat{\theta} - \theta = -\mathbf{H}(\theta)^{-1}\mathbf{g}(\theta), \tag{A.15}$$

or

$$\sqrt{n}(\hat{\theta} - \theta) = \left[-\frac{1}{n}\mathbf{H}(\theta)\right]^{-1} \frac{1}{\sqrt{n}}\mathbf{g}(\theta). \tag{A.16}$$

Now, if the quantity $\left[-\frac{1}{n}\mathbf{H}(\theta)\right]^{-1}$ is regarded as constant [141], and by the law of large numbers and the central limit theorem $\frac{1}{\sqrt{n}}\mathbf{g}(\theta) \to \mathbf{0}$ then $E(\hat{\theta} - \theta) = \mathbf{0}$, with the assumption that there is no bias. Therefore, $E(\hat{\theta}) = \theta$, which is another way to establish consistency. As for the variance of $\hat{\theta}$, it is just the outer product of A.16, such that

$$\begin{aligned} \mathbf{V}(\sqrt{n}(\hat{\theta} - \theta)) &= E[\sqrt{n}(\hat{\theta} - \theta)\sqrt{n}(\hat{\theta} - \theta)^{\mathrm{T}}] & \text{(A.17)} \\ &= E\left[\left[-\frac{1}{n}\mathbf{H}(\theta)\right]^{-1}\left(\frac{1}{n}\right)\mathbf{g}(\theta)\mathbf{g}(\theta)^{\mathrm{T}}\left[-\frac{1}{n}H(\theta)\right]^{-1}\right], & \text{(A.18)} \end{aligned}$$

which, if efficiency is established, reduces to

$$\mathbf{V}(\sqrt{n}(\hat{\theta} - \theta)) = E\left[-\frac{1}{n}\mathbf{H}(\theta)\right]^{-1} = \mathbf{I}^{-1}, \tag{A.19}$$

where **I** is the Fisher information matrix. Finally, following Cameron and Trivedi [89],

Definition 1 *(Asymptotic Distribution of $\hat{\theta}$) If*

$$\sqrt{n}(\hat{\theta} - \theta) \longrightarrow \mathcal{N}(\mathbf{0}, \mathbf{I}^{-1}), \tag{A.20}$$

then in large samples, $\hat{\theta}$ is asymptotically normally distributed with

$$\hat{\theta} \sim \mathcal{N}(\theta, \mathbf{I}^{-1}). \tag{A.21}$$

Usually, if asymptotic consistency is established, then the ML estimator converges in probability to the true parameter value [89].

A.1.3 Asymptotic Efficiency

The efficiency property states that as the sample size increases, the variance of the ML estimator approaches a minimum bound established by the Cramér-Rao theorem.

Theorem 2 *(Cramér-Rao Lower Bound) Assuming that the density of y_i satisfies the regularity conditions, the asymptotic variance of a consistent and asymptotically normally distributed estimator of the parameter vector θ will always be at least as large as*

$$[\mathbf{I}(\theta)]^{-1} = (-\mathrm{E}[\mathbf{H}(\theta)])^{-1} = \left(\mathrm{E}\left[\mathbf{g}(\theta)\mathbf{g}(\theta)^{\mathrm{T}}\right]\right)^{-1}. \tag{A.22}$$

A.1.4 Invariance

The following theorem describes the invariance property of ML estimators [142]:

Theorem 3 *If $\hat{\theta}$ is the ML estimator of the parameter vector θ and $\rho(\theta)$ is a function of θ, then $\rho(\hat{\theta})$ is the ML estimator of $\rho(\theta)$.*

Appendix B

Support Vector Machines

Let $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ be a set of training data where each $\mathbf{x}_i$ in $\mathbb{R}^n$ denotes a sample in the input space with a corresponding output $y_i \in \{1, 0\}$, for $i = 1, 2, ...n$ where n corresponds to the size of the training data. The goal is to find a separating hyperplane as far as possible from the nearest different instances while keeping all the instances in their correct side. In other words, the objective is to maximize the distance d between the separating hyperplane ($\langle \mathbf{x}, \beta \rangle + \beta_0 = 0$) and its nearest different instances, while placing the margin hyperplanes ($\langle \mathbf{x}, \beta \rangle + \beta_0 \pm 1$) into the separating margin. The vector β represents the normal of the hyperplane and β_0 is the offset from the origin. When all the instances are correctly classified, the problem is called *hard-margin* SVM. However, in real-life data, instances are not all correctly classified, and the problem is *soft-margin* SVM. The distance between the separating hyperplane and each of the margin hyperplane is $d = \frac{1}{||\beta||}$. Mathematically [118], the objective involves solving the following optimization problem:

$$\begin{aligned} \text{Minimize :} \quad & \tfrac{1}{2}||\beta||^2 + C\sum_{i=1}^{n} \xi_i, \\ \text{Subject to :} \quad & y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \end{aligned} \tag{B.1}$$

where ξ_i is nonnegative slack variable representing the errors. Thus, when an instance $\mathbf{x_i}$ is correctly classified by the hyperplane, and is outside of the margin, the corresponding slack variable $\xi_i = 0$. When the instance correctly classified, but is within the margin, then $0 < \xi_i < 1$. If the instance is misclassified, then $\xi_i > 1$. As for the constant C, it is also nonnegative and it represents the trade-off between maximizing the margin and minimizing

the errors. Figure B.1 illustrates the concept of SVM for classification.

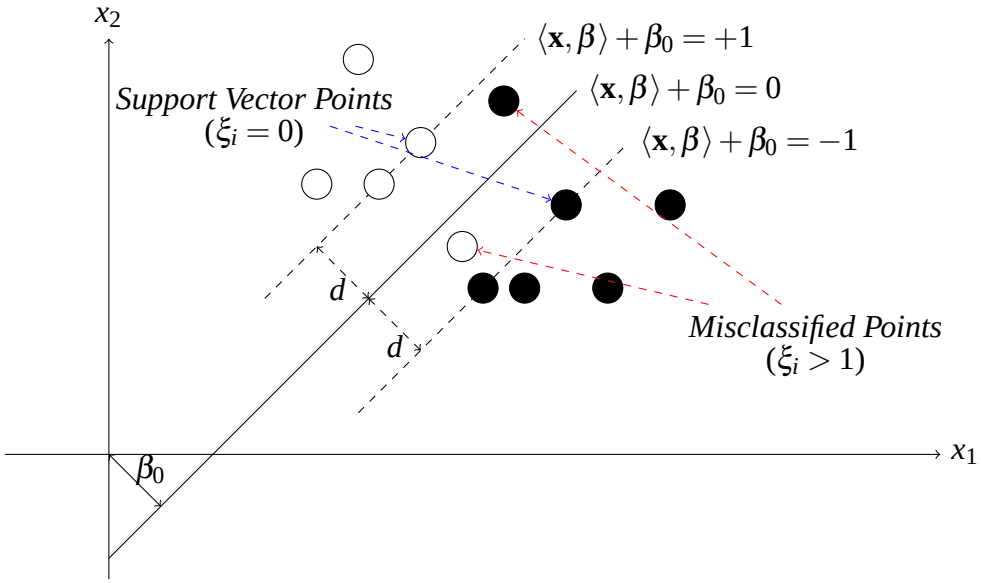

Figure B.1: SVM for classification.

Now, the Lagrangian (L_P) of the primal optimization program (B.1) can be expressed as

$$\begin{aligned} L_P = & \quad \tfrac{1}{2}||\beta||^2 + C\sum_{i=1}^{n}\xi_i \\ & - \sum_{i}^{n}\alpha_i(y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 + \xi_i) \\ & - \sum_{i}^{n}\mu_i\xi_i, \end{aligned} \tag{B.2}$$

where α_i, and μ_i are Lagrange multipliers associated with the constraints. The Karush-Kuhn Tucker (KKT) conditions can now be derived from the Lagrangian by taking the first-order derivatives of L_P with regard to β, β_0 and ξ_i, then setting them to zeros. The minimum to the optimization problem (B.2) is then reached when

$$\begin{aligned} \beta &= \sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i, \\ \sum_{i=1}^{n}\alpha_i y_i &= 0, \\ \alpha_i + \mu_i &= C, \end{aligned} \tag{B.3}$$

and the rest of the KKT conditions are [143]:

$$\begin{aligned}
\text{Constraint 1}: &\quad y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 + \xi_i \geq 0, \\
\text{Constraint 2}: &\quad \xi_i \geq 0, \\
\text{Multiplier Condition 1}: &\quad \alpha_i \geq 0, \\
\text{Multiplier Condition 2}: &\quad \mu_i \geq 0, \\
\text{Complementary Slackness 1}: &\quad \alpha_i[y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 + \xi_i] = 0, \\
\text{Complementary Slackness 2}: &\quad \mu_i \xi_i = 0,
\end{aligned} \tag{B.4}$$

for $i = 1 \ldots n$. If $\xi_i \geq 0$, then $\xi_i = 1 - y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0)$ and $\mu_i = 0$, hence $\alpha_i = C$. If $\xi_i = 0$, then $\mu_i > 0$, and hence $\alpha_i < C$. In addition, if $\xi_i = 0$, and $y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 = 0$, then $\alpha_i > 0$. Otherwise, if $y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 > 0$, then $\alpha_i = 0$.

Therefore, instances that are not on the support vector plane and are on the correct side have $\xi_i = \alpha_i = 0$. Instances on the support vector plane have $\xi_i = 0$, but $\alpha_i > 0$. Finally, data points on the wrong side of the support vector hyperplane have $\alpha_i = C$ and ξ_i balances this violation such that $y_i(\langle \mathbf{x}_i, \beta \rangle + \beta_0) - 1 + \xi_i = 0$. In other words, only a subset of the Lagrange multipliers would have a nonzero value in the solution, while others would vanish. Instances for which $\alpha_i = 0$ are called *support vectors* (SV).

Now, substituting the values in (B.3) back into (B.2) and replacing the dot product $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ gives the following dual optimization problem,

$$\begin{aligned}
\text{Maximize}: &\quad L_D = -\tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i}^{n} \alpha_i, \\
\text{Subject to}: &\quad \sum_{i=1}^{n} \alpha_i y_i = 0, \\
&\quad 0 \leq \alpha_i \leq C.
\end{aligned} \tag{B.5}$$

The dot product in (B.5) was replaced with a function $\kappa(\mathbf{x}_i, \mathbf{x}_j)$, called the kernel function. The kernel function maps the input vectors to a feature space that consists of the inner products of the mapped vectors [117]. Linear classification methods are then applied in that

feature space. Note that Complementary Slackness 1 and 2 show that $\xi_i = 0$ if $\alpha_i < C$. Thus, any training point for which $0 < \alpha_i < C$ can be taken to use Complimentary Slackness 1, with $\xi_i = 0$ to compute β_0. An alternative way to compute β_0 is by the following:

$$\beta_0 = \frac{1}{n_{sv}} \sum_{i=1}^{n_{sv}} (y_i - \langle \mathbf{x}_i, \beta \rangle), \tag{B.6}$$

where n_{sv} is the number of support vectors [144]. However, for kernel-based SVM, it is not necessary to calculate the value of β_0, as it is implicitly part of the kernel function [144].

Finally, classification of a new instance, $\mathbf{x}$, is then carried out based on the rule

$$f(\mathbf{x}) = sign \left[\sum_{i=1}^{n} y_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + \beta_0 \right], \tag{B.7}$$

for any $\alpha_i \neq 0$.