Learning from Asymmetric Models and Matched Pairs

by

Derek Koh

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved January 2013 by the
Graduate Supervisory Committee:

George Runger, Chair
  Tong Wu
  Rong Pan
  John Cesta

ARIZONA STATE UNIVERSITY

May 2013
ABSTRACT

With the increase in computing power and availability of data, there has never been a greater need to understand data and make decisions from it. Traditional statistical techniques may not be adequate to handle the size of today’s data or the complexities of the information hidden within the data. Thus knowledge discovery by machine learning techniques is necessary if we want to better understand information from data. In this dissertation, we explore the topics of asymmetric loss and asymmetric data in machine learning and propose new algorithms as solutions to some of the problems in these topics. We also studied variable selection of matched data sets and proposed a solution when there is non-linearity in the matched data.

The research is divided into three parts. The first part addresses the problem of asymmetric loss. A proposed asymmetric support vector machine (aSVM) is used to predict specific classes with high accuracy. aSVM was shown to produce higher precision than a regular SVM. The second part addresses asymmetric data sets where variables are only predictive for a subset of the predictor classes. Asymmetric Random Forest (ARF) was proposed to detect these kinds of variables. The third part explores variable selection for matched data sets. Matched Random Forest (MRF) was proposed to find variables that are able to distinguish case and control without the restrictions that exists in linear models. MRF detects variables that are able to distinguish case and control even in the presence of interaction and qualitative variables.
ACKNOWLEDGEMENTS

I would like to extend my greatest gratitude to my friends and colleagues through my Ph.D. journey that have helped, encouraged, and motivated me throughout the process. Without them, I would not have been able to produce this dissertation. Each of them have touched me in a different way and have helped made this experience enjoyable.

I would like to thank Dr. George Runger for his patience, insights, and guidance through this process. It is an honor to have such a distinguish professor serve as the chair of my dissertation committee. Throughout my graduate studies, when I was a full time student, and when I was working full time, Dr. Runger believed in my work and ability to get through the Ph.D. process. The wealth of knowledge and experience that he has imparted to me has been invaluable and will serve me throughout my life.

I would like to thank Dr. John Cesta, who has been a mentor to me from my undergraduate studies through my graduate studies. I am eternally grateful for his support and friendship. He provided motivation throughout my college career and helped me find direction along the way. Thanks for believing in me and helping me through it all.

Dr Teressa Wu and Dr Rong Pan have had significant impact on me and the classes that I’ve taken from them gave me the knowledge that was used in this dissertation. It is an honor to have them both as my committee members.

I appreciate the support given by Dr. Scott Menor, who was a committee member when I presented my proposal, and the School of Computing, Informatics and Decision Systems Engineering at Arizona State University.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Research Scope</td>
<td>4</td>
</tr>
<tr>
<td>1.2 Asymmetric Support Vector Machines (Chapter 3)</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Asymmetric Variable Selection (Chapter 4)</td>
<td>5</td>
</tr>
<tr>
<td>1.4 Matched Random Forest (Chapter 5)</td>
<td>7</td>
</tr>
<tr>
<td>2 BACKGROUND AND LITERATURE REVIEW</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>9</td>
</tr>
<tr>
<td>2.2 Asymmetric loss</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Quantile Regression</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Exploring the loss functions</td>
<td>11</td>
</tr>
<tr>
<td>2.5 Imbalance datasets</td>
<td>14</td>
</tr>
<tr>
<td>2.6 Linear Exponential Loss (LINEX)</td>
<td>15</td>
</tr>
<tr>
<td>2.7 Bayesian Asymmetric Loss</td>
<td>17</td>
</tr>
<tr>
<td>2.8 Bootstrapping and Bagging</td>
<td>18</td>
</tr>
<tr>
<td>2.9 Asymmetric Bagging</td>
<td>19</td>
</tr>
<tr>
<td>2.10 Asymmetric Boosting</td>
<td>21</td>
</tr>
<tr>
<td>2.11 Asymmetric variables</td>
<td>23</td>
</tr>
<tr>
<td>2.12 Classification and Regression Tree</td>
<td>26</td>
</tr>
<tr>
<td>2.13 Random Forest</td>
<td>27</td>
</tr>
</tbody>
</table>
## CHAPTER 3

### 3.1 Abstract

### 3.2 Introduction

### 3.3 Motivation and Background

- **3.3.1 Motivation**

- **3.3.2 Quantile Regression and Pinball Loss**

### 3.4 Methodology

- **3.4.1 Loss Optimization**

- **3.4.2 Parameter Values for Asymmetric Parameters**

- **3.4.3 Extensions to ASVM**

### 3.5 Experiments

- **3.5.1 ASVM on Simulated Data**

- **3.5.2 ASVM on Heart Disease Data**

- **3.5.3 ASVM on Class Imbalance Problem**

### 3.6 Conclusion and Future Work

### 3.7 Addendum to the Generalized Asymmetric SVM: A comparison with regular SVM

- **3.7.1 Comparison of Regular SVM to a Generalized Asymmetric SVM**

- **3.7.2 Precision Comparison Methodology**

- **3.7.3 Uneven dataset Comparison Methodology**

### 4 ASYMMETRIC RANDOM FOREST FOR FEATURE SELECTION

iv
# 4. Chapter 4

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>64</td>
</tr>
<tr>
<td>4.2</td>
<td>Background</td>
<td>65</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Asymmetric Features</td>
<td>67</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Loss functions in trees</td>
<td>68</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Random Forest</td>
<td>69</td>
</tr>
<tr>
<td>4.3</td>
<td>Asymmetric Variable Selection</td>
<td>70</td>
</tr>
<tr>
<td>4.4</td>
<td>Asymmetric Random Forest</td>
<td>74</td>
</tr>
<tr>
<td>4.5</td>
<td>Experiment</td>
<td>75</td>
</tr>
<tr>
<td>4.5.1</td>
<td>ARF on Simulated Asymmetric Features</td>
<td>75</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Asymmetric Forest on Simulated Asymmetric features that interacts</td>
<td>78</td>
</tr>
<tr>
<td>4.5.3</td>
<td>Asymmetric features in financial disclosure data</td>
<td>82</td>
</tr>
<tr>
<td>4.6</td>
<td>Conclusion</td>
<td>88</td>
</tr>
</tbody>
</table>

# 5. Chapter 5

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>89</td>
</tr>
<tr>
<td>5.2</td>
<td>Background</td>
<td>92</td>
</tr>
<tr>
<td>5.3</td>
<td>Conditional Logistic Regression</td>
<td>92</td>
</tr>
<tr>
<td>5.4</td>
<td>Classification and Regression Tree</td>
<td>94</td>
</tr>
<tr>
<td>5.5</td>
<td>Random Forest</td>
<td>97</td>
</tr>
<tr>
<td>5.6</td>
<td>Motivation</td>
<td>97</td>
</tr>
<tr>
<td>5.7</td>
<td>Matched Random Forest</td>
<td>98</td>
</tr>
<tr>
<td>5.7.1</td>
<td>Numerical Variables</td>
<td>100</td>
</tr>
<tr>
<td>5.7.2</td>
<td>Categorical Variables</td>
<td>102</td>
</tr>
<tr>
<td>5.7.3</td>
<td>Variable Importance</td>
<td>102</td>
</tr>
</tbody>
</table>
### CHAPTER 5.7.4 Probe Variables

5.8 Experiments

5.8.1 Simulated Dataset

5.8.2 Case 0: Null Case

5.8.3 Case 1: Linear Effect

5.8.4 Case 2: Non-linear Effect

5.8.5 Case 3: Two Variable Non-linear Effect with Interaction

5.8.6 Case 4: Three variables with complex interactions

5.8.7 Case 5: Categorical variables

5.8.8 Case Study

5.9 Conclusion

6 CONCLUSION

6.1 Future Work

REFERENCES
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The table shows the results of a simulated bi-variate normal data in two-dimensional space. The results are the average for 10 replicates of a five fold cross-validated experiment. Standard deviations are shown in the parenthesis. As shown from the results, varying the values of the ( \rho )'s and ( \epsilon )'s affects the precision and recall.</td>
<td>48</td>
</tr>
<tr>
<td>3.2</td>
<td>The table shows the results of the heart disease data set when evaluated by 10 replicates of a five-fold cross-validation for each learner. The values are the average of the replicates. Standard deviations are shown in the parenthesis.</td>
<td>51</td>
</tr>
<tr>
<td>3.3</td>
<td>Asymmetric paired SVM results (pASVM). The first 2 rows show the results for a regular and median SVM. The remaining rows have 2 sets of parameters that are for the two ASVMs in the pASVM. The values, which are the average for 10 replicates of the five-fold cross-validation. Standard deviations are shown in the parenthesis.</td>
<td>53</td>
</tr>
<tr>
<td>3.4</td>
<td>Results for the qASVM compared against the regular and median SVM. The values are the average for 10 replicates of the five-fold cross-validation. Standard deviations are shown in the parenthesis.</td>
<td>54</td>
</tr>
<tr>
<td>3.5</td>
<td>The results are the average of 10 replicates for a three-fold cross-validation. The counts are round to the nearest integer. Standard deviations are shown in the parenthesis.</td>
<td>55</td>
</tr>
</tbody>
</table>
4.1 Example dataset of an asymmetric variable $x$. When $x = 1$ then $y = 1$ but when $x = 0$, $y$ can be either 1 or 0. Thus $x = 1$ is a good predictor of class 1. 73

4.2 Simulated Features for Asymmetric Variable Selection. The parameters $\pi_1$ and $\pi_2$ corresponds to the accuracy of predicting class 1 and class 0 respectively. Feature 2 to Feature 7 are known as class 1 features as they are more accurate at predicting class 1 than class 0. Feature 8 to Feature 13 are known as class 0 features as they are more accurate predicting class 0 than class 1. 76

4.3 Average ranks of class 1 variables from 10 replicates. The rank is out of 30 where the lower the rank the more important the variable is. 78

4.4 Results from simulated asymmetric variables. Average class 1 rank is the average rank of class 1 features. $\Delta$ Class 1 and 0 is the difference of the average rank of class 1 features from class 0 features. $\Delta$ Class 1 and Noise is the difference of the average rank of class 1 features from the noise features. The asterisks indicates the difference is significant at $\alpha = 0.05$. The first row with Gini and $\rho$ is the standard RF algorithm. The ARFs showed significant differences in the rankings of class 1 features from class 0 and noise features. 80

4.5 Example dataset of an asymmetric variable $x$. When $x = 1$ then $y = 1$ but when $x = 0$, $y$ can be either 1 or 0. Thus $x = 1$ is a good predictor of class 1. 80
4.6 Average ranks of decomposed features from 10 replicates. The prefix of the decomposed feature is the label of the original feature. The rank is out of 14 where the lower the rank the more important the variable is. .......................... 82

4.7 Results from simulated asymmetric variables. Average class 1 rank is the average rank of class 1 features. ∆ Class 1 and 0 is the difference of the average rank of class 1 features from class 0 features. ∆ Class 1 and Noise is the difference of the average rank of class 1 features from the noise features. The asterisks indicates the difference is significant at $\alpha = 0.05$. The first row with Gini and $\rho$ is the standard RF algorithm. The ARFs showed significant differences in the rankings of class 1 features from class 0 and noise features. .......................... 84

4.8 Description of features used .......................... 86

5.1 Probability table for generating categorical variables. The categories of the control observations, $x_0$, are generated with equal probability. Based on the the values of the control, the cases are then generated with the following probabilities shown above. 118

5.2 Description of features used .......................... 122
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Loss function for absolute error and quadratic loss. The diamonds show the loss for absolute error and the squares show the quadratic loss. The cost of loss increases quadratically for the quadratic loss, from being equal with the absolute error when the loss is 1, to being 5 times the magnitude when the loss is 5.</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Loss function for LINEX. When the error is negative, the loss function is linear. When the error is positive, the loss is exponential. Thus the training is controlled by the outlier observations on the positive side and controlled equally by all observations on the negative side.</td>
<td>16</td>
</tr>
<tr>
<td>2.3</td>
<td>Recursive binary splitting. The left graph shows the tree that is grown by recursive splitting. The right graph shows the regions that is cased by the splits.</td>
<td>28</td>
</tr>
<tr>
<td>3.1</td>
<td>Quantile regression conducted on a data of sine wave with noise. Different values of ( \theta ) were used representing the different quantiles estimated.</td>
<td>39</td>
</tr>
<tr>
<td>3.2</td>
<td>The figure shows the penalty resulted from the error ( y - f(x) ). The parameters changes either the length or slope of the lines they are closest to. ( \epsilon_1 ) and ( \epsilon_2 ) changes the length of the lines and ( \rho_1 ) and ( \rho_2 ) changes the magnitude of the slopes.</td>
<td>40</td>
</tr>
</tbody>
</table>
3.3 Heat map of the classification space for various values of \( \epsilon \) and \( \rho \). The lighter regions are classified as positive and the darker regions are classified as negative.

3.4 The figure shows the penalty resulted from the error \( y - f(x) \). The parameters changes either the length or slope of the lines they are closest to. \( \epsilon_1 \) and \( \epsilon_2 \) changes the length of the lines and \( \rho_1 \) and \( \rho_2 \) changes the magnitude of the slopes.

3.5 The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of \( C \) and \( \gamma \). The line represents that of the regular SVM and the * are points of the proposed loss function with different values for \( \rho \). Except for Figure 3.5(c), the points of the proposed loss function generally are on top of the regular SVM.

3.6 The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of \( C \) and \( \gamma \). The line represents that of the regular SVM and the * are points of the proposed loss function with different values for \( \epsilon \). The points of the proposed loss function generally at least that of the regular SVM.
3.7 The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of C and $\gamma$ based on an uneven dataset where the negative class only represented 13% of the training data. The line represents that of the regular SVM and the * are points of the proposed loss function with different values for $\rho$. 

4.1 Impurity for different values of p. 

4.2 The graphs show the variable importance based on different ARF compared to the regular RF. In total, 10 replicates of 200 trees were grown for each simulation and the average results are reported. ARF is able to pick out the Class 1 features (feature 2 to 7) from class 0 and noise features while the regular RF is unable to differentiate between class 1 and class 0 features. 

4.3 The graphs show the variable importance based on different ARF compared to the regular RF. 200 trees were grown for each simulation. ARF is able to pick out the Class 1 features (feature 2 to 7) from class 0 and noise features while the regular RF is unable to differentiate between class 1 and class 0 features. 

4.4 Variable Importance for Form 4 data based on 200 Trees. The lower ranks are more important. 

5.1 Recursive binary splitting (Hastie et al., 2001). The left graph shows the tree that is grown by recursive splitting. The right graph shows the regions that is cased by the splits.
5.2 Illustration of bivariate predictors that distinguishes case and control non-linearly. Figure 5.2(a) shows the data without matching and no apparent relationship can be seen. Figure 5.2(b) shows the data with a matching. Figure 5.2(c) shows the matched data and subregions where the effect of the predictors can be easily identified visually.

5.3 MRF on the Case 0 noise variables. The graph gives the variable importance (VI) for each variable where the smaller the VI the better the variable is at distinguishing cases and controls. The variables $x_{11}$ to $x_{20}$ are probe variables.

5.4 p-values of MRF and CLR on the noise variables. In the presence of noise none of the variables are significant at $\alpha = 0.01$ when both MRF and CLR were used.

5.5 VI values from MRF on Case 1 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables.

5.6 Case 1 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. Both CLR and MRF were able to detect the $x_1$ at $\alpha = 0.01$.

5.7 VI values from MRF on Case 2 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{21}$ to $x_{60}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables.
5.8 Case 2 p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$ at $\alpha = 0.01$ while for the same level of significance, CLR was not able to detect $x_1$.  

5.9 VI values from MRF on Case 3 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ and $x_2$ are noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.  

5.10 Case 3 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$ and $x_2$ at $\alpha = 0.01$ while CLR was not able to detect both $x_1$ or $x_2$ at $\alpha = 0.01$.  

5.11 VI values from MRF on Case 4 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$, $x_2$, and $x_3$ are noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.  

5.12 Case 4 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$, $x_2$, and $x_3$ at $\alpha = 0.01$ while for the same level of significance, CLR was not able to detect $x_1$, $x_2$, and $x_3$.  

5.13 VI values from MRF on Case 5 categorical variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.

5.14 Case 5 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$ at $\alpha = 0.01$ at both levels of $\gamma$ while CLR was not able to detect $x_1$ at both levels of $\gamma$.

5.15 VI values from MRF on financial variables of top 500 profitable firms by market capitalization. The variables Net Income, Total Asset, Cashflow from operations, Cost of Goods Sold, Long Term Debt, and Capital Expenditure appear to have a noticeably lower VI values than the probe variables.

5.16 CLR and MRF on financial variables of top 500 profitable firms by market capitalization. CLR is not able to detect significance of variables at $\alpha = 0.01$. Apart from revenue and cash, MRF detects the other variables as significant at $\alpha = 0.01$. 
Chapter 1

INTRODUCTION

In an era of information technology there is an increasing amount of data that is becoming readily available. Such large data sets require the use of new tools and technologies to process and learn information present within the data. The field of statistical and machine learning is thus becoming more and more important in the research of such quantities of data. Designed to overcome the short comings of the older statistical methods, these new tools present an improvement over methods such as regression and ANOVA. They optimize knowledge discovery for large data sets, for higher-dimensional data sets, and data where the relationship between predictors and response is non-linear. The use of many of these advanced tools increases the need for better learners and variable selection algorithms.

The information that is needed from analyzing data differs throughout different application. For instance, a fund manager using data to predict stock prices differs greatly from a designer predicting foots sizes to design shoes. For a fund manager, to minimize his or her financial loses, he or she would be more averse to over estimating stock prices. The designer, on the other hand, does not care if he or she over estimates or underestimates the size of a person’s feet as long as the variance is small. Thus, specific algorithms need to be designed to cater to the different needs of the application.

In addition to using the right information from data sets for decisions, it is also necessary to understand what variables within a data set is important. Within data sets, characteristics of predictors can vary greatly. The data may
possess information that have unique properties that common machine learning variable selection algorithms may not be able to detect. It can be difficult to understand how predictors relate to the response as the relationship between the predictor and the response can be non-linear. It is the case for some predictors that only certain values of the predictor are useful in predicting the response. There are also cases when a predictor may only be good at predicting certain values of the response, and not all values of the response. The variety of characteristics in data sets call for customized methods of variable selection to discover such variables.

Training data used in machine learning have rows of a vector of predictors $\mathbf{x} = (x_1, ..., x_p)$ and a response value $y$. Each row corresponds to an observation in $p$-space. There is less focus in the machine learning literature on other kinds of data sets like matched data where the rows of the data may correspond to multiple points in $p$-space. However, such data sets are well studied in the statistics literature and have practical applications that span fields like healthcare and engineering. As machine learning techniques are apt at handling non-linear relationships in data sets and data sets with complex relationships, it would only be beneficial if these techniques are applied toward matched data sets.

In this dissertation, we contribute to the literature of machine learning by proposing a new learning algorithm that optimizes the precision for the desired class, a variable selection algorithm that finds asymmetric variables, and an algorithm that is able to detect complex relationships in matched data sets. This research contributes to the work in the current literature in three parts.

The first part of this research presents a generalized way of looking at asymmetric loss. This part contributes via presenting a general model for
asymmetric loss and applies it to the support vector machine binary classifier (Vapnik, 1998). To bridge the theoretical and the practical, this part also attempts to reconcile how such a loss function is useful by presenting ways in which such functions can be utilized in practice.

The next part strives to take the asymmetric ideas presented in the former research further by applying variables selection to find variables that are predictive of a subset of values in the response. Variable selection has traditionally been focused on searching for variables that are predictive for all values of the response. Consider a binary classification problem where \( y = 0, 1 \), a variable is usually considered predictive of the response if it is able to predict both the class 0 and class 1. Common variable selection techniques are apt at picking such variables. However, there could also be variables that are able to only predict class 1 well and not class 0. Such variables may often be missed by common variable techniques even though they are still useful, particularly if the user only cares about predicting class 1 accurately. The second part of the research focuses on finding variables that are predictive of a subset of values in the response, and not necessarily all response values. This allows variables that are asymmetrically predictive to be selected when in traditional variable selection algorithms, such variables might not be selected.

The third part of the research explores identifying useful variables that are able to distinguish case and control in matched studies. Matched data is a set of data observations that share a similar characteristics and thus can be grouped together. The most common type of matching is a 1-to-1 matching where each data point is matched with another point because they are similar. We usually denote one of the points as case and the other as control. The concept of matched data is used in longitudinal studies. For example, in a study
of fever medicine effectiveness, a patient has his temperature taken before and after treatment. The temperature before and after treatment can be matched together as it came from the same patient. Data does not have to be inherently matched as most data sets can be transformed to a matched data set. The third contribution of the research focuses on detecting relationships between variables in 1-to-1 matched data. Conditional logistic regression (CLR) is a traditional model for binary classification with matched data (Hosmer and Lemeshow, 2000). CLR provides information on which variables are important in distinguishing the case from the control. However, the method is based on a linear model. Non-linear relationships where high and low values of a variable may be attributed to the cases and values in between attributed to the control may not be detectable by CLR. The problem of addressing such non-linearity in CLR has not been addressed in the literature. Thus the third contribution introduces an algorithm to manage problems where the relationship between case and control is non-linear.

1.1 Research Scope

The research scope provides an overview of the research, its contribution to the existing literature, and its importance in real world applications. The research consists of three parts spread out into three different chapters. The chapters build on each other successively to define the overall thesis on asymmetric loss.

1.2 Asymmetric Support Vector Machines (Chapter 3)

In Chapter 3, a generalized asymmetric loss function is proposed and explored. There has been various studies of asymmetric loss in the literature
and these studies are explored in Chapter 2. No one study has adequately articulated the full benefits of asymmetric loss in a general fashion. This study contributes to the existing literature by formally presenting an algorithm that can be trained to increase precision in asymmetric prediction. It also extends the existing methods to form a generalized function for asymmetric loss.

The two main problems that this study argues would be better with asymmetric loss are the class imbalance problem, and the one sided optimization problem. The chapter considers the existing literature and argues that improvements can be made to the results if the papers considered using asymmetric loss. Certain loss functions such as that which is used in regression and that which exists in SVM are reviewed and the suitability of different types of loss functions is explored. The pinball loss (Koenker and Hallock, 2001), which is an important element of the generalized loss function proposed, is then presented to show that changing the penalty conditioned upon the sign of the residual creates quantile estimates, a result that can be useful in certain applications. After laying out the background for a generalized asymmetric loss, the proposed loss function is then derived and shown to have a close form solution within an SVM context. The operation of how this generalized SVM works is then shown graphically. The asymmetric SVM is applied to various datasets to show how it can benefit practical problems. In addition, combinations of these SVMs are introduced to demonstrate the ability for better performance when applied in an ensemble framework.

1.3 Asymmetric Variable Selection (Chapter 4)

Little has been done to use asymmetric learning in the area of variable selection. Variable selection algorithms are mostly designed to find variables
that can predict all possible values of the dependent variable. Consider a bi-
nary dependent variable with a positive and negative class. Variable selection
techniques are designed to pick variables that are predictive of when an ob-
servation is negative and when an observation is positive. In these techniques,
there is little value in picking variables that are able to predict one class ac-
curately. As a result, variables that are only able to predict a positive class
accurately tend to not be selected when using traditional algorithms. Chap-
ter 4 explores the use of asymmetry by using ensemble based methods and
in particular, random forests (Breiman, 2001), to discover variables that are
asymmetrically predictive.

This chapter modifies the models for variable selection such that it is more
attuned to a specific class of the dependent variable. This is done by modifying
the loss function within the tree. In trees, the loss function is the purity
function which is used to decide which split to make. More information on
how a tree works and the purity function can be found in Chapter 2. In this
research, we modify the purity function such that purity of one class is more
desired.

We adjust the purity function in two ways, by modifying the impurity mea-
sure (Gini index), and by modifying the weights given to the child nodes of a
split. We propose two different impurity measures that replaces the Gini index
for asymmetric variable selection. In addition we also propose two different
weighting schemes to weight the child nodes. Each of these modifications have
different benefits and is able to solicit asymmetric variables in their own way.

Simulated datasets that have variables that are predictive of both the posi-
tive and negative class, predictive of only the positive class, and not predictive
at all will be used as a test to see if the modification works. In addition, real
life financial data is used to test if the method is useful in real applications. The modified random forest is tested against options exercise and sell data which is known to be asymmetric in nature.

1.4 Matched Random Forest (Chapter 5)

Stratification, or blocking is commonly used in studies to reduce the variability of nuisance parameters from the model. A stratification parameter is added to the model to account for this nuisance variability in the model. For instance in the finance field it is well known that companies should be grouped into its respective peer group such that better analysis can be conducted. It would not make sense to compare the earnings of an energy company with that of a bank. The most rudimentary type of clustering is that of a 1:1 matched data. A 1:1 matching study is commonly called a case-control study where one of the matched data is labeled as the case and the other is labeled as control. In matched data experiments, we study whether the covariates have different values between the case and control and therefore is a distinguishing factor of the case and control.

Methods to find variables that distinguishes the case and control, such as conditional logistic regression (CLR) (Hosmer and Lemeshow, 2000), are available in the literature. CLR has been widely studied in the literature and also used in practice. However, the method lacks the ability to account for relationships that are non-linear. In addition, such linear models are not apt at managing categorical data. A different method would be to use random forest to learn relationships within matched data sets. There are many advantages to random forest over a linear model including accommodating categorical variables, auto-correlation, and most importantly, non-linear matched data sets.
In Chapter 5, we propose an algorithm to accommodate matched data sets for finding complex relationships between case and control. The objective of this research is to discover the variables that can distinguish the case from the control, even if the effect of the variable changes. The proposed algorithm, Matched Random Forest (MRF), allows for the relationship between case and control to be different in different regions of the variable’s space. Moreover, MRF is designed to detect relationships between the case and control in special cases on interactions between covariates. We show that MRF is able to detect variables that can distinguish case and control when the relationship is consistent throughout the variable space. This is the scenario where CLR is applicable and is widely used. We then introduce the problem where the ability for a variable to distinguish case and control depends on the region in the space the variable is located. For instance, an increase in debt of a firm that has low debt results in good stock performance, while an increase in debt when the firm has high debt results in poor stock performance (Campello, 2006). Thus, even though debt is able to predict stock performance, its relationship with stock performance depends on the level of debt. Such relationship is non-linear and is difficult to detect with regular statistical techniques. We showed that complex relationships such as non-linearity and multiple interactions could be discovered where as traditional statistical techniques such as conditional logistic regression is unable to pick the variables that are important. We apply MRF to financial data to show that MRF can be used for practical purposes.
Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Introduction

This chapter summarizes the background of the three parts in this dissertation. The background for each part is also introduced in each of the following four chapters.

2.2 Asymmetric loss

The notion of asymmetric loss is not something that is new. The method can be traced back to the 1980’s on the idea of estimating quantiles using quantile regression (Koenker and Bassett, 1978). Koenker and Bassett (1978) realized instead of a mean estimate, it may be useful to obtain a quantile estimate in certain situations. For instance, it is useful to know the 90th percentile of heights of males in clothing design. For such problems optimizing the squared error loss, even though it is computationally convenient, leads to bias estimates (Koenker and Hallock, 2001). We are able to get estimates of quantiles by using least squares regression. In a regression model, it is assumed that $f(Y|X) \sim N(E[Y|X], \sigma^2)$. Thus from the normal distribution, one can determine a quantile estimate based on the following equation

$$
\hat{\tau}_\theta = \Phi^{-1}(\theta)\sigma + E[Y|X] \tag{2.1}
$$

where $\theta$ is the quantile, $\Phi^{-1}(\cdot)$ is the inverse standard normal distribution, and $\sigma$ is the model’s standard deviation. This approach, although simple to
implement, is subjected to the assumptions of least squares which specifies that the errors are normally and identically distributed. A departure from such assumptions such as the presence of kurtosis results in a bias quantile estimate. Handling problems where the normality of residuals are violated is well documented in the literature. An example of a remedy is the use of Box-Cox transformations (Wei et al., 2006). However, implementing such a transformation in practice may be problematic if it violates the domain understanding of the problem. Thus using the $z$-estimate of a quantile is not a robust way of estimating the percentiles. To discover accurate quantile estimates, there needs to be a more robust model estimation that is able to estimate quantiles when the assumptions of normality are relaxed.

2.3 Quantile Regression

Quantile regression has been used successfully in many applications. Bassett (2007) used quantile regression to measure the handicap of football games. A decision to use quantile regression was appropriate to understand the probability of a score difference in a game which would then be used as a measure for a future game’s handicap. Hewson and Yu (2008) use quantile regression to assess local government based on a set of performance indictors. They try to match relative performance based on the upper and lower quartile. Good estimates required that quantile regression be used and modified to model the binary responses. Hewson and Yu (2008) also showed the viability of quantile regression in a Bayesian framework by trying to solve for a credible interval for the coefficients of covariates. These covariates provide insights into the reasons why local governments out-perform or under-perform their peers. Wei et al. (2006) use quantile regression for estimating the growth charts of children’s
Figure 2.1: Loss function for absolute error and quadratic loss. The diamonds show the loss for absolute error and the squares show the quadratic loss. The cost of loss increases quadratically for the quadratic loss, from being equal with the absolute error when the loss is 1, to being 5 times the magnitude when the loss is 5.

heights and weights. As presented in the work, traditional regression modeling lacks the robustness that quantile regression offers. Christmann and Steinwart (2008) took regular quantile regression and added a kernel. Wei et al. (2006) showed that the non-linear transformation of a radial basis function kernel is valid and consistent for the loss function. Using the radial basis function kernel and simulated results, the research showed how quantiles can be obtained from non-linear data.

2.4 Exploring the loss functions

The difference between least squares regression and quantile regression lies in the loss function of these two methods. The loss function quantifies the difference between the predicted and the actual result and is denoted as $L(Y, f(X))$. For a regular least squares regression, the loss function is as follows

$$L(Y, f(X)) = (Y - f(X))^2$$  \hspace{1cm} (2.2)

Minimizing the loss function reduces the risk of errors. If predictions are 100% accurate, the value of the loss function is 0. The magnitude of a loss function
is always non-negative. When used in machine learning algorithms, the loss function is made the objective function to be minimized. Observing equation 2.2, we see that the difference, $Y - f(X)$, is squared. As a result the marginal increase in the loss function gets larger as the difference between $Y$ and $f(X)$ increases. Thus, it is more likely that a model is susceptible to being heavily trained by the outliers of the dataset. An alternative is to have the marginal increase in loss be constant as the difference between $Y$ and $f(X)$ increases. An example of a loss function that achieves that is shown in the below equation

$$L(Y, f(X)) = |Y - f(X)|$$

(2.3)

Here the results has a constant marginal loss rate and the resulting model is less likely to be heavily influenced by outliers. We see that by changing the loss function, we can obtain models that exhibit different characteristics.

Steinwart (2007) has formalized the ways of comparing loss functions by using surrogate functions and identifying which functions are better in terms of computational requirements and robustness concerns. In the construction of many algorithms, the loss function may be either too computationally intensive or may not be robust. Steinwart (2007) suggests using a surrogate loss function, that is similar to the original loss function, in place of the original loss function so that the problem becomes less computationally intensive and the learning can be more robust. Steinwart (2007) provided criteria for suitable surrogate loss functions. The criteria allows for creativity in loss function designs to allow the learner to accomplish different objectives. For instance, the condition that the errors of difference signs has to be the same can be relaxed and the resulting model would be more accurate in predictions on one side of the error than the other. This kind of design is known as asymmetric loss.
Asymmetric loss has been explored in the literature of different disciplines. In the psychology literature, Weber (1994) showed that asymmetric cost is present in people’s decision making. Weber (1994) tries to quantify asymmetry cost psychologically. It discusses how people rank outcomes in a relative way instead of an absolute way, paying too much attention to a decision that has higher cost and too little attention to that which has little cost. Finally, stochastic dominance is used to justify the asymmetry in a decision where even though the expectations are the same, a stochastically dominant choice is preferred.

Huang (2001) proposed a new model that takes both quality and cost into account. The model is an extension of the classical Taguchi (Taguchi, 1995) quality model where only quality is controlled and it tries to find a balance between quality and cost. McCullough (2000) created a generalized loss function that can account for asymmetry to use for modeling of interest rates.

Koenker and Hallock (2001), in rationalizing quantile regression, understood that to build a model for estimating quantiles, the loss function would have to be asymmetric, penalizing residuals with different signs differently. Patton and Timmermann (2006) showed evidence that mean loss is not optimal for problems that are innately asymmetric, particularly in a non-linear model. When an artificial data set that is constructed to be imbalanced such that the proportion of class instances are either much less or much more than 50%, the use of mean loss often leads to bias results. Patton and Timmermann (2006) showed that modeling of stock prices are optimal only under certain strict restrictions and thus conventional modeling techniques such as squared error loss are not robust to changing conditions.
Changing conditions will exist when a predictor relationship with the response is non-symmetric. Patton and Timmermann (2006) highlighted conditional variance (see Hentschel (1995) for an explanation of conditional variance in time series modeling) as an example where the relationships between predictors are non-symmetric and the mean loss function fails to provide good estimations of the time-series being modeled. Patton and Timmermann (2006) presented a more generalized framework to accommodate asymmetric loss in stock price modeling which includes the use of transformation of the errors and using asymmetric loss functions such as linear-exponential loss. The paper stressed that given the non-symmetric nature of stock price movement, it is critical to create models that are more robust to deviations from the current stock forecast models.

Demetrescu (2007) presented an optimal criterion for forecast intervals under asymmetric loss functions. In forecasting times-series data, one is concern with the bounds of error. However, if one is concern with the direction of violation of the forecast, an optimal forecast interval should account for the risk associated with violations on different directions. Demetrescu (2007) showed how to construct optimal bounds with different risk expectations using asymmetric loss.

2.5 Imbalance datasets

In addition to minimizing risk in different directions, asymmetric loss can also be useful in classification problems where the proportion of response classes are significantly different. This is known as the imbalanced dataset problem. Examples of such problems spans many fields, from the detection of oil spills (Kubat and Matwin, 1998), to detecting fraud in mobile communi-
cations (Fawcett and Provost, 1997) and credits cards, to predicting when a manufacturing process will fail (Riddle and Etzioni, 1994), to diagnosing rare diseases (Laurikkala, 2001). This section provides an overview of how the class imbalance problem has been handled in the academic literature.

Daskalaki et al. (2006) explored the various ways that such imbalance datasets can be handled and showed that asymmetric learning is able to improve the results of detecting the class that has fewer samples. Daskalaki et al. (2006) showed that various learners such as neural network (Hopfield, 1982), C4.5 (Quinlan, 1993), and logistic regression (Hosmer and Lemeshow, 2000) all fail to adequately classify the class that has the least proportion in the training dataset. However, by using a voting algorithm described in the paper, the ability of correctly classifying the minority class can be improved.

Asymmetric prediction of imbalanced data can be managed by applying asymmetric SVMs (Hwang and Shim, 2005) (Huang and Du, 2005). Cohen et al. (2003) uses an asymmetric SVM to tackle the imbalance problem in the surveillance of nosocomial infections. The data set that Cohen et al. (2003) used has a percentage of positive cases of about 11% which results in the learner, which in this case is SVM, always predicting the negative class in the out-sample. Adjusting the loss function of the SVM such that there is asymmetry causes the SVM to be more sensitive to positive cases. The result of making such a modification causes the learner to be able to pick the positive class.

2.6 Linear Exponential Loss (LINEX)

The loss function used in quantile regression, known as the pinball (Takeuchi and Le, 2006), is a common asymmetric loss function that is used. However, it is not necessarily the only loss function that provides asymmetric loss. The
Figure 2.2: Loss function for LINEX. When the error is negative, the loss function is linear. When the error is positive, the loss is exponential. Thus the training is controlled by the outlier observations on the positive side and controlled equally by all observations on the negative side.

LINEX loss function is another proposed loss function with a desired property of being sensitive to outliers on one side and not on the other (Chang and Hung, 2007). Christoffersen and Diebold (1997) shows the use of LINEX loss in time series estimation where the errors may not necessarily be normally distributed. LINEX loss can be expressed as below.

\[ L(Y, F(X)) = e^{\alpha(Y - f(X))} - \alpha(Y - f(X)) - 1 \] (2.4)

where \( \alpha \) determines the linear and exponential portions. When \( \alpha > 0 \), the loss is approximately linear when \( f(X) > Y \) and exponential when \( f(X) < Y \). The converse is true when \( \alpha < 0 \). Statistical parameters have been estimated with the aid of LINEX when the underlying distribution is non-symmetric. In regression, the test for significance of the coefficients is done by using a scale parameter. The covariance matrix is commonly used as the scale parameter. As the covariance matrix is estimated and is used for coefficient estimates and t-statistics, an under-estimation of the covariance matrix may lead to misleading results of coefficient significance. This prompts the need to use a loss function that is less restrictive and can be manipulated to be robust in parameter estimation. Giles and Giles (1993) used the LINEX model to
predict the scale parameter for regression where we would like to be more conservative in its quantity. They show that the estimated results from LINEX are robust when the underlying errors in the data are mildly different from a normal distribution. However, in more extreme departures from a normal distribution, Giles and Giles (1993) concludes the resulting outcomes are less robust. The use of LINEX has been successfully applied to ridge regression Ohtani (1995) Akdeniz and Namba (2003).

2.7 Bayesian Asymmetric Loss

The work of asymmetric loss has been studied for use in Bayesian statistics, both in the likelihood function and in estimating credible intervals. Stuger (2006) used asymmetric loss to construct credible intervals in the estimation of the binomial parameter. The need for asymmetric loss for this estimation arose from the monitoring of public health systems where costs of overestimation and underestimation are different. The sample size is optimized based on an asymmetric approach applied to the likelihood function. Given an unspecified prior distributions, Kaminska and Porosinski (2009) proved that Bayesian estimators can be constructed with a proposed methodology: Bounded asymmetric Bayesian loss (ABL). Kaminska and Porosinski (2009) proposed the (ABL) as follows

\[ L(v, d) = K(1 - \left(\frac{v}{d} e^{1-(\theta/d)}\right)^{\gamma}) \] (2.5)

where \(K > 0, \gamma > 0\) are known parameters. \(K\) represents the maximum loss and \(\gamma\) determines the shape. The loss function has a property of robustness that winsorizes the extreme outliers. Winsorizing is done to control the influence of outliers in the model. This kind of winsorization has also been applied to LINEX in a method known as Bounded Asymmetric Loss (BLINEX). Wen
and Levy (2001) presented the BLINEX loss function as follows

\[
\frac{1}{\lambda} \left[ 1 - \frac{1}{1 + \lambda c(e^{ax} - ax - 1)} \right] \tag{2.6}
\]

Like ABL, BLINEX is also a robust estimator in that extreme outliers get winsorized. The paper showed that with BLINEX, the Bayes estimate of a parameter is unique and exists.

### 2.8 Bootstrapping and Bagging

Bootstrapping (Efron and Tibshirani, 1993) is a sampling methodology commonly used in machine learning. An example of a bootstrapping algorithm is as follows. Given a standard training set \( T \) of size \( n \), then \( B \) sets of data of size \( b \), \( b \leq n \) are generated by randomly selecting from the original training set \( T \) with replacement. As the core of the bootstrapping algorithm is sampling with replacement, one can get a customized bootstrap datasets \( B \), that has certain characteristics that are different than the original dataset, \( T \). An example of this is creating a balanced bootstrapped data set from an imbalanced training data set by oversampling the under-represented class. As oppose to learners where the algorithm itself is adjusted during training, bootstrapping provides a learning environment where the training data, because it is being manipulated, is also actively involved in the training process.

Bootstrapping is used in many ensemble methods. One of the most well-known ensemble methods that use bootstrapping is bagging (Breiman, 1996). Bagging reduces the variance of prediction and improves classification and regression models in terms of stability and classification accuracy. Bagging is a technique that repeatedly bootstraps from a data set according to a uniform probability distribution. Each bootstrap sample has the same size as the
original data. As bootstrapping is conducted, some observations may appear several times in the same training data set, while others may be omitted from the training data set. In each bootstrap sample, about two-thirds (\(\sim 63.2\%\)) of the training data are sampled while another one-third is unused. The unused data are called out-of-bag (OOB) data. For each of the \(B\) bootstrap samples, a learner is trained on that sample. When classifying a new observation, the observation is classified by each of the \(B\) learners and the results are combined by taking the majority vote from each of the learners.

2.9 Asymmetric Bagging

Asymmetric loss has been shown to be useful in various different applications, both in the literature as well as when applied practically. Work has also been done in the area of ensembles where asymmetric loss is applied on the ensemble method to the learners instead of within the learners. Tao et al. (2006) used bootstrapping with Support Vector Machine (SVM) learners to manage the problem of small datasets, and imbalanced dataset. Bagging was conducted in 3 ways: bootstrapping the cardinality of the independent variables to be trained in the classifiers, bootstrapping the dependent variable, and doing both of the aforementioned ways simultaneously. The first way takes care of having more predictors than observations, the second way takes care of the imbalanced dataset problem, and the third combines the benefits of the previous two ways. The results showed that SVM has an over-fitting problem and is biased in an uneven training dataset. Thus the bootstrapping methods mitigated these issues and produced compelling results.

Li et al. (2008) evaluates the efficacy of asymmetric methods applied within an algorithm such as SVM and also outside of an algorithm such as the use
of bagging (Breiman, 1996). Li et al. (2008) applied these methods to an imbalanced dataset of drug molecules activities. The results showed that, in the case of an asymmetric method applied within an algorithm, an asymmetric SVM does not provide any incremental value over a regular SVM. Asymmetric bagging on the other hand, provides considerable improvements to regular bagging of SVM learners. Li et al. (2008) also included their own modification to asymmetric bagging. Known as PRIFEAB bagging, the learner removes irrelevant and redundant features before performing the bagging algorithm to improve the performance of asymmetric bagging.

Random forests uses bagging to train each node of the trees in the forest. Asymmetry has been applied to random forest in the literature. One of the papers that explore the idea of asymmetry in random forest is Meinshausen (2006) where the idea of using quantile regression for getting prediction intervals for ozone levels and detecting outliers was proposed. The key difference between quantile regression forests and random forests is as follows: for each node in each tree, random forests takes the mean of the observations in the node for prediction and neglects all other information. In contrast, quantile regression forests takes into consideration the values of all observations in a node and not just its mean. With these observations, the quantile regression forest is able to find the conditional quantile given an observation. Quantile regression forests (QRF) is applied to various popular data sets from the machine learning literature and results are compared to various other quantile regression methods: linear quantile regression with interactions (QQR) and without interactions (LQR), and quantile regression trees with piecewise constraints (TRC), piecewise multiple linear (TRM), and piecewise second-degree polynomial form (TRP). The results showed QRF to be robust in various simulated datasets.
2.10 Asymmetric Boosting

Apart from bagging, boosting is also another type of ensemble where asymmetric learning can be applied (Hamed and Nuno, 2004). Mease et al. (2007) discusses quantile estimation and class probabilities in Logitboost (Friedmann et al., 2000) and Adaboost (Freund and Schapire, 1995). These algorithms were used to estimate probabilities of being in a particular class and these probabilities are used as thresholds to establish quantiles. Unlike bagging where each learner in the ensemble is trained independently from the other learners, algorithms like Adaboost trains a learner sequentially. Each successive learner trained is fed with training data that had been transforms by a cost factor. Achieving asymmetric classification involves changing the cost factor to something other than the median. Mease et al. (2007) noted that the boosting of asymmetric classifiers must stop early as they are prone to overfitting. This is because the longer the boosting process, the more likely the classification probability either limits to 1 or 0. Jittering over and under sampling (JOUS) with Adaboost was suggested as a method to preserve the advantages of boosting algorithms while protecting against overfitting. JOUS with Adaboost as applied to variable asymmetric data sets and the method performed better than the traditional methods of adjusting the cost of misclassification.

Viola and Jones (2006) describes cases where fast detection is more relevant than accurate detection. In using asymmetric Adaboost to detect faces, it was shown that asymmetric Adaboosting is better than regular Adaboost. Also the asymmetric Adaboost can be used as a speedy filter to filter through the false negative such that the dataset is vastly reduced and another more accurate
trainer can be used to identify the faces. Fenske et al. (2009) used gradient boosting of regular regression to estimate the bottom 5% of children nutritional scale for identifying children at risk of malnutrition. Let \((x_i, y_i), i = 1, ..., n\) be the training dataset and \(L(y, F(x))\) be a differentiable loss function. In gradient boosting, a number of iterations, \(M\), is set for the algorithm as follows

1. Initialize the model with a constant value: 
\[
F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i; \gamma)
\]

2. For \(m = 1, ..., M\)
   
   (a) Compute \(r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x) = F_{m-1}(x)}\)

   (b) Fit a base learner \(h_m(x)\) to \(r_{im}\).

   (c) Compute \(\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))\)

   (d) Update the model \(F_m(x) = F_{m-1}(x_i) + \gamma_m h_m(x_i)\)

3. Output \(F_m(x)\)

(Lee and Horowitz, 2005) modified the boosting algorithm by applying quantile regression as the learner and by changing the calculation of \(r_{im}\) to as follows

\[
r_{im} = \begin{cases} 
\tau & y_i - F(x_i) > 0 \\
0 & y_i - F(x_i) = 0 \\
\tau - 1 & y_i - F(x_i) < 0 
\end{cases}
\]

(2.7)

where \(\tau \in (0, 1)\) is an asymmetric parameter. Using this boosting algorithm allows for there to be non-linearity as oppose to just using a single least squares regression model. Chaudhuri and Loh (2002) used the GUIDE algorithm in Chaudhuri et al. (1995) to form quantile regression trees. The quantile estimation is done within the algorithm of the tree instead of in the boosting
process. Trees use the residuals of the previous tree as the response in the learning phase. After the splitting process in the tree algorithm, piecewise polynomials are used for each of the leaf node of the tree to predict observations. The size of piecewise polynomial regression tree models can be adjusted by changing the form of the polynomial fitted at the leaf nodes which allows for greater flexibility in the predictions at the leaf node as oppose to the common regression tree model where the prediction at the leaf nodes are constants.

2.11 Asymmetric variables

Choosing variables that are highly predictive the responses would generally lead to a better model. There is a great amount of research devoted to finding the best variables for use in model training and prediction in the data mining literature. There are three main ways the variable selection is conducted (Saeyes et al., 2007) : filter, wrapper, embedded. The filter selection techniques assesses the intrinsic properties of the predictor and assigns a score. The variables with the best scores are the picked. The simplest filters come in the form of t-stat or ANOVA (Jafari and Azuaje, 2006). Correlation based feature selection (CFS) is a method that picks variables with high correlation to the dependent variable and low correlation to the other independent variables (Hall, 1999). Similar to CFS is the minimum Redundancy Maximum Relevance Feature Selection (mRMR) algorithm which is based on the concept of gathering a subset of independent variables that are highly predictive of the dependent variable and highly dissimilar with other independent variables (Ding and Peng, 2003). The Markov blanket filter builds a Markov blanket that contains a minimal subset of relevant features that yields optimal classification (Zeng et al., 2009).
Wrappers utilize a learning algorithm as a black box to score subsets of variables according to their predictive power. The wrapper methodology was popularized by Kohavi et al. (1996) and is a simple and powerful way to address the problem of variable selection. Some popular known wrapper algorithms include sequential forward and backward selection (Kittler, 1978), simulated annealing (Kirkpatrick et al., 1983), randomized hill climbing (Skalak, 1994), genetic algorithms (Holland, 1975), and estimation of distribution algorithms (EDA) (Blanco et al., 2004). Sequential forward and backward selection uses regression as the learner to decide the best subset of variables that are relevant. Simulated annealing, randomized hill climbing and genetic algorithms both use the benefits of randomness to find a subset of variables that are pertinent for model prediction. EDA is a general version of the genetic algorithm. When iterating a genetic algorithm to find the best subsets, equal priors are placed on each of the variables in the current subset. In EDA, the priors are updated and represents a probability distribution of the variable’s pertinence.

For embedded methods, the search for a best subset of features is built into the classifier construction, and can be seen as a search in the combined space of feature subsets and hypotheses. This process may be more efficient in several respects. First, it makes use of the available data by not needing to split the training data into a training and validation set. Second, it reaches a solution faster by avoiding retraining a predictor from scratch for every variable subset investigated. Embedded methods are not new: decision trees such as CART (Breiman, 1984), for instance, have a built-in mechanism to perform variable selection. Random forest is an algorithm that is widely used in the literature as an embedded system for variable selection (Diaz-Uriarte and Alvarez de Andres, 2006) (Jiang et al., 2004). A variant of SVM known as SVM with
recursive feature selection is an SVM method that has an embedded feature selection technique (Guyon et al., 2002). Finally Ma and Huang (2005) tweaks the logistic regression method to produce an embedded method that eliminates variables with small weights.

The current research in variable selection does not consider unique variable types such as asymmetric variables. Asymmetric variables, when added to a model, improves the prediction accuracy of a subset of the responses as oppose to all response values. In asymmetric learning, we do not need predictors that can be used for predicting all values of the response, we only need the predictor to be effective at predicting the specific value that we want to focus on. Let \( y \in \{-1, 1\} \) be a binary response, we have a model \( f(x) = y \) where \( x = (x_1, x_2, \ldots, x_p) \) and \( x_i \in P \). Let \( P' \in P \) be a subset of all the possible variables \( P \). The variable selection techniques described previously tries to maximizes the function \( \max_{P'} P(Y = y | f(x)_{x \in P'} = y) \), which gives the optimal subset \( P' \) that when used as inputs to a learner for training produces a model that predicts all values of \( y \) with high accuracy. For asymmetric learning, we want to find \( P' \) such that \( \max_{P'} P(Y = 1 | f(x)_{x \in P'} = 1) \). We are only concerned about the accuracy of prediction when the predicted value is 1.

A review of the methods in the literature came out short on finding variables that are particularly good at asymmetric prediction. Asymmetric variables appear to be an area that has not been studied. However, asymmetric variables appear to be prevalent in the literature. Matzler et al. (2004) did a regression analysis to determine what attributes lead to better overall customer satisfaction. They showed that for variables of complaint handling, project management, and innovativeness, when the performance is low, there is a significant effect on overall satisfaction. However, when these same vari-
ables showed high performance, the effect on overall satisfaction is insignificant. Froyen et al. (1997) studied whether political pressures add significant explanatory power in monetary policy. They found that in white house administrations that pressure the Federal Reserve to tighten monetary policy results in higher interest rates whereas in white house administrations that promote loose monetary policy does not affect the interest rates. Finally Karras (1996) identified money-supply shocks and their effects on output for a panel of 18 European countries and found that many different specifications and estimation methods strongly support asymmetry: negative money-supply shocks are shown to have a statistically significant effect on output, whereas the effect of positive shocks is statistically insignificant.

2.12 Classification and Regression Tree

Tree based methods are commonly used in data mining. A popular method for tree-based classification and regression is CART (Classification and Regression Tree) (Breiman, 1984). Breiman (1984) showed that regression trees can be used to predict a continuous outcome, and classification trees can be used to predict a discrete outcome. The goal of tree-based methods is to partition the data into sub regions and then fit a simple model in each one (Hastie et al., 2001). A simple tree structure is a binary-split tree where data is recursively partitioned by a decision boundary that splits the data into two separate regions. The model predicts $Y$ by the following expression

$$f(x) = \sum_{m=1}^{M} c_m I\{x \in R_m\}$$  \hspace{1cm} (2.8)

where $R_m, m = 1, 2, ..., M$ are the disjoint regions of the predictor space, $c_m$ is the majority class in region $R_m$, and $I(\cdot)$ is an indicator function that gives a
value of 1 if the input condition is true and 0 otherwise. For a binary response dataset where $y \in \{0, 1\}$, the impurity of a region, $Q_m$, can be modeled with the Gini index as follows

$$Q_m = Gini(p) = 1 - p^2 - (1 - p)^2 \quad (2.9)$$

where $p$ is the proportion of observations that have $y = 1$ in region $R_m$. At every node the data is split based on a split point $s$. We define a pair of subregions as

$$R_1(j, s) = \{X | X_j \leq s\} \quad \text{and} \quad R_2(j, s) = \{X | X_j > s\} \quad (2.10)$$

Let $Q_1(j, s)$ and $Q_2(j, s)$ be the impurity for $R_1(j, s)$ and $R_2(j, s)$ respectively. We seek the splitting variable $j$ and the split point $s$ that minimizes the node impurity $v_m(j, s)$ by solving the following

$$\min_{j, s} v_m(j, s) = \min_{j, s} \left( \frac{N_1}{N_m} Q_1(j, s) + \frac{N_2}{N_m} Q_2(j, s) \right) \quad (2.11)$$

where $N_m, N_1, N_2$ are the number of observations in $R_m, R_1(j, s), R_2(j, s)$ respectively.

To illustrate how a classification tree works, consider a response $y$ and two predictor variables, $x_1$ and $x_2$. A value of $s_1$ that minimizes $v(j, s_1)$ is found. The space is divided into 2 sub regions and the process is continued. Figure 5.1, shows the resulting decision tree and the partitioned regions of $x_1$ and $x_2$. There are four separate regions that resulted from all the splits $s_1, s_2$ and $s_3$.

2.13 Random Forest

As individual tree classifier can be complex, unstable, and overfitted, ensemble methods, consisting of trained classifiers, were introduced for tree
classifiers to mitigate those issues (Opitz and Maclin, 1999). Random forest (Breiman, 2001) consists of a collection of tree predictors that each vote on the response of an observation. Let \( t \in T \) be the trees in the random forest, and \( f(x) \) be an individual tree predictor, the majority vote from all \( T \) trees are used as the predicted response of the random forest as follows

\[
\arg\max_y \sum_{t=1}^{T} I(f(x) = y)
\]  

(2.12)

Each tree in the random forest is constructed by a bootstrapped (Efron and Tibshirani, 1993) sample of the training data set. Let \( P \) be the set of all the variables used for training. For each node, the random forest algorithm only considers a random subspace \( P' \in P \) to use for splitting. The predictor variables \( j \in P' \) are randomly selected for each node and the splitting criteria becomes

\[
\min_{j \in P', s} \left( \frac{N_1}{N_m} Q(j,s) + \frac{N_2}{N_m} Q(j,s) \right)
\]  

(2.13)

In addition to classifying data, random forest can also be used for variable selection. Breiman (2001) discusses two ways in which variable selection can be conducted by random forest: variable importance, and decrease in Gini index. The reader can refer to Genuer and Poggi (2010) to get a better understanding for both of these measures.
Conditional logistic regression (CLR) is based on logistic regression and is used for matched studies (Hosmer and Lemeshow, 2000). The matched data used in these studies are typically formed by stratification which is the process of dividing members of the population into homogeneous subgroups. For the case of stratified data CLR is a more appropriate analysis than logistic regression because it accounts for the strata in the study within the analysis. CLR is used to determine if the value of the predictor is informative and is able to distinguish which subject is more likely to be the case (Hosmer and Lemeshow, 2000). For the purpose of this study, we only consider the specific case of 1:1 matching although other kinds of matching such as 1:m and n:m exists. Let \( x = (x_1, ..., x_p) \) be a vector of predictors and \( y \in \{0,1\} \) be the binary response that denote the case and control respectively. Denote

\[
E(Y) = P(Y = 1|x) = P(x)
\]

The logistic model is given by

\[
P(x) = \frac{1}{1 + e^{-\beta'x}}
\]

where \( \beta' = (\beta_1, ..., \beta_p) \) are the vector of coefficients of predictors \( x \). The logit model can be written as

\[
\log \left[ \frac{P(x)}{1 - P(x)} \right] = g(x, \beta) = \beta'x
\]

In CLR for matched case-control studies, two subjects are paired together to form matched units. For example, in the study by Berg et al. (2010), the observation of a patient with acromegaly and a patient from the general population with similar age and sex are matched to form a stratum.
For a 1:1 matched case, suppose there are $K$ strata with two subjects, a case and a control observation, in the $k^{th}$ stratum, where $k = 1,\ldots,K$. The conditional logistic regression model is

$$P(x) = \pi_k(x) = \frac{e^{\alpha_k + \beta'x}}{1 + e^{\alpha_k + \beta'x}}$$ (2.17)

where $\alpha_k$ is a stratification parameter with the contribution of all terms constant within the $k^{th}$ stratum. Given observations $(y_i, x_i), i = 1,\ldots,n$, the probability of the observed data conditioned on the stratum total and the total number of cases observed is denoted as the conditional likelihood for the $k^{th}$ stratum. The conditional likelihood for the $k^{th}$ stratum is (Hosmer and Lemeshow, 2000).

$$l_k(\beta) = \frac{e^{\beta'x_{1k}}}{e^{\beta'x_{1k}} + e^{\beta'x_{0k}}}$$ (2.18)

where $x_{1k}$ denote the case observation and $x_{0k}$ denote the control observation in the $k^{th}$ stratum. The full conditional likelihood over the $K$ strata of $l_k(\beta)$ with only $\beta$ as the only unknown parameter is the product of each $k^{th}$ conditional likelihood

$$l(\beta) = \Pi_{k=1}^{K} l_k(\beta)$$ (2.19)

The maximum likelihood estimators (MLE) of the conditional logistic likelihood function, $l_k(\hat{\beta})$, are obtained by finding the $\beta$ that maximizes the conditional likelihood function in Equation 5.6.

The slope coefficient, $\beta_j$ for $j = 1,\ldots,p$, gives the change in the log-odds for one unit increase in $x_j$ holding all other $x$’s constant. For example, $\beta_j > 0$ implies as $x_j$ increases so does the probability of the observation being a case. Like logistic regression, in CLR we test the significance of $\beta_j$ by conducting a Wald test (Rao, 1973) which compares $\hat{\beta}_j$ to an estimate of its standard error.
$\hat{SE}(\hat{\beta}_1)$. When $\beta_j$ is significant, there is a linear relationship between case and control in $x_j$ and thus we can use $x_j$ to distinguish case and control.
Chapter 3

GENERALIZED ASYMMETRIC SVM AND ITS APPLICATION

3.1 Abstract

In many data mining problems, the cost of misclassifying can be different. Such problems are known to be asymmetric and learners like Support Vector Machines (SVM) which uses the hinge loss function for training fails to adequately account for the asymmetric costs. In our research, we propose a modification to the SVM loss function such that would handle such asymmetric problems. A generalized asymmetric loss function is presented and applied to two kinds of problems: the optimization of precision, and the training of unbalanced data. Using the blood transfusion data from UCI, the asymmetric SVM was able to improve precision from 76% to 96%. To show the ability of training an uneven dataset, an unbalanced blood transfusion dataset was used where the number of true-positives was doubled without any loss of precision.

3.2 Introduction

The most common data mining techniques are trained to make predictions by minimizing the error rate. These techniques possess a loss function that penalizes errors by being impartial to whether the error is a false-positive (FP) or a false-negative (FN). There are a class of problems where the optimization of errors are focused solely on the ability to identify a positive class accurately. For such problems, the ability to obtain true-positives (TP) is more important than obtaining a true-negative (TN). For instance, the lawsuit that ensues
from an FP of wrongly accusing a patient of medical fraud is more punitive than the FN of allowing the fraud to happen (Liou et al., 2008). It is more important to identify a fraud case with high probability then to identify a large number of fraud cases. Such problems are better handled by optimizing the precision, defined as \((\frac{TP}{TP + FP})\), instead of the maximizing both TP and TN simultaneously.

This paper presents a generalized loss function that allows for asymmetric learning to occur. Not only would a generalized loss function result in better precision, it can also be applied to a common phenomena in many datasets where the training data has much more observations for one class. The imbalance dataset problem has often been mitigated in the literature with the utilization of an asymmetric resampling methodology (Hamed and Nuno, 2004). This paper attempts to tackle the challenges of precision optimization and the imbalance dataset problem by substituting the conventional hinge loss in support vector machines (SVM) with a generalized asymmetric loss function. This allows us to make adjustments to parameters that will either improve precision and/or increase recall. Allowing for different parameter values to be applied to the SVM algorithm such that an over prediction would have a different penalizing weight than that of an under prediction of the same magnitude, the generalized asymmetric loss presented is an extension of the pinball loss function Koenker and Hallock (2001) used in quantile regression (Steinwart, 2007). The pinball loss is not the only kind of asymmetric loss function in the literature. Other forms of asymmetric loss include Linex (Demetrescu, 2007) (Ohtani, 1995), and ramp loss (Takeuchi and Le, 2006). These alternative loss functions, although useful, have their shortcomings. Linex consists of a non-linear portion within the function which when applied to SVM will not
guarantee a sparse solution. The ramp loss penalizes data points far from the margin equally which results in a loss of information that those data points provide. The pinball loss not only resembles closest to the loss function in a support vector regression machine making it more natural to use, but it is linear in all regions which provides a sparse solution and it is able to accommodate the influence of extreme data points. With this asymmetric penalization, the SVM can be trained to be more selective in classifying a positive class to reduce the probability of a FP. It can also be used as alternative to the resampling approach of unbalanced datasets by allowing for greater penalization for the majority class when training an SVM. This would be an advantage over the resampling methodology as it maintains the integrity of the original data.

The paper is organized as follows. Section 3.3.1 presents the motivation of using an asymmetric loss function in the healthcare industry by presenting papers that would have been better off if asymmetric loss functions were introduced. Next, quantile regression is presented to establish the pinball loss function and how it is effective at optimizing losses asymmetrically. Section 3.4 presents a generalized asymmetric loss function and integrates this into the SVM algorithm. Section 3.5.1 illustrates how the generalized asymmetric loss function can be tuned to obtain different results with a simulated bi-variate normal distribution dataset. The remaining sections show how the generalized asymmetric SVM can be applied in practice by varying the parameters of the loss function to either obtain higher precision or higher hit rates of one class.
Machine learning learners generally focus on using symmetric loss functions in practical applications. However, for many of the applications, the objectives can benefit from an increase in precision of prediction. This is particularly useful in the medical field where the risks associated with misclassification is based on the misclassified prediction. For instance, which treatment gives a patient the highest probability of success, or which set of genes are most likely to cause cancer are examples of questions that would require the use of asymmetric loss as the risks of a FN and TN is different. Moreover, data obtained from the medical field often have disproportionate classes which makes it hard for many classifiers to learn to differentiate the data.

Liou et al. (2008) tried to solve the problem of increasing the precision of fraud detection in medical claims. High precision is needed as the false implication of fraud may bring forth lawsuits and consume resources. However, the paper focused on using symmetric learners such as logistic regression, neural network, and classification tree instead of directly optimizing precision. Although the results were compelling, the paper’s objective of maximizing the number fraud prediction while minimizing false positives were not addressed directly by the methods employed.

A lot of work have been done in the healthcare machine learning field to try and detect the differences in gene expressions between normal tissues and cancerous tissues (Ambroise and McLachlan, 2002)(Guyon et al., 2002). The ramifications for a FP and and FN are different but the algorithms employed by the literature lack the risk aversion feature which would minimize situations where expressions that are not cancerous are wrongly classified as cancerous.
To manage such problems, it would be beneficial to employ an asymmetric method that directly optimizes the precision of prediction.

The uneven dataset problem is often addressed by manipulating the training data. Liou et al. (2008) resampled its training data to produce a training dataset where the classes are even. Huang and Du (2005) managed the uneven dataset problem by using a weighted methodology for each training observation to mitigate the problem of the bias. The class with less observations is given more weight per observation. The weighted SVM technique was applied to the breast cancer diagnosis dataset available on the UCI database where only 10% of the training data was classified as malignant. Although the classification of malignant class increased, the precision did not improve over the regular SVM. Cohen et al. (2003) tackled the same problem in the healthcare field but instead of using a weighted approach, the authors used an asymmetric margin where different penalties were assigned to classifying the wrong class. The method was applied to the identification of nosocomial infections from Geneva University Hospital where the training data had 10% of the observations labeled as positive. The pinball loss employed was able to increase the accuracy of prediction by 5% over the baseline of a regular SVM. In this paper we take the idea of Cohen et al. (2003) use of asymmetric margins a step further by introducing a more general form of the pinball loss function to account for asymmetric data sets and noisy class boundaries.

3.3.2 Quantile Regression and Pinball Loss

In least square regression, the conditional expectation of the target, $Y$, is used as the solution that minimizes the error which is $E[(Y - f(X))^2]$. This
results in the estimate of the mean. We can decompose the problem as follows

\[
E[(Y - f(X))^2] = E[(Y - f(X))^2|Y < f(X)] P(Y < f(X)) \\
+ E[(Y - f(X))^2|Y \geq f(X)] P(Y \geq f(X)) \tag{3.1}
\]

To minimize \( E[(Y - f(X))^2|Y < f(X)] \), least squares may not yield the optimal solution. In this case, it is more important to find a \( y \) such that a proportion, \( \theta \), of \( Y \) has labels \( Y < y \). The solution to this problem results in the \( \theta \)-quantile of \( Y \) which we denote as \( \mu_\theta \). We find \( \mu_\theta \) by the following equation

\[
\mu_\theta = \arg \min_\mu \left[ P(Y \leq \mu) = \theta \right] \tag{3.2}
\]

When \( \theta = 0.5 \) the median is achieved. The conditional quantile \( \mu_\theta(x) \) for a pair of random variables \((X, Y) \in \mathbb{R}\) is defined as the function \( \mu_\theta(x) \) where the point \( \mu_\theta \) is the solution to.

\[
\mu_\theta(x) = \arg \min_\mu \left[ P(Y \leq \mu) = \theta | x \right] \tag{3.3}
\]

The basic strategy to obtain quantile estimates arises from the observation that minimizing the L1-loss function for a location estimator yields the median (Hao and Naiman, 2007). Denoting an indicator function as \( I(\cdot) \), observe that to minimize the following expression

\[
\min_\mu \sum_{i=1}^{N} |y_i - \mu| = \min_\mu \sum_{i=1}^{N} (y_i - \mu) I(y_i \geq \mu) - (y_i - \mu) I(y_i < \mu) \tag{3.4}
\]

by choice of \( \mu \), an equal number of observations have to lie on both sides of zero in order for the derivative with respect to \( \mu \) to vanish. Koenker and Hallock (2001) generalizes this idea to obtain a regression estimate for any quantile by proposing the pinball loss function. Estimates of the \( \theta \)-quantile can be obtained from the pinball loss.
\[ l_\theta = \begin{cases} 
\theta(y - \hat{y}) & \text{if } (y - \hat{y}) \geq 0, \\
(\theta - 1)(y - \hat{y}) & \text{if } (y - \hat{y}) < 0.
\end{cases} \tag{3.5} \]

When we apply loss function on data and attempt to minimize it, we find the number of observations with \( y_i < \mu_\theta \) is bounded from above by \( N\theta \) and the number of terms with \( y_i > \mu_\theta \) is bounded from below by \( N(1 - \theta) \). As \( N \) approaches infinity, the number of terms below \( \mu_\theta \) to the total number of terms converges to \( \theta \). Minimizing Equation 3.5 with \( \theta = 0.5 \) when applied on a training dataset results in the expression shown in Equation 3.4. Linear optimization programs such as CPLEX can be used to solve the minimization problem. A simulated Sine wave dataset with noise was simulated and Figure 4.1 gives a graphical example of quantile regression for different values of \( \theta \) estimated from the data. Quantile regression using support vector regression was studied by Hwang and Shim (2005) and has been successfully applied to linear and non-linear data. However little focus is given to the modification of the loss function for classification or even the generalization of such functions. As such functions are apt in asymmetric problems and imbalance data set problems, it is worthwhile to find a generalization to such functions when applied SVM so that better results can be achieved.

3.4 Methodology

Works in the literature have shown that the pinball loss described in the previous section can also be applied to a Support Vector Machine (Quadrianto et al., 2009). The hinge loss innate in SVM is replaced with a pinball loss function used in quantile regression. In this paper, we generalize the function by adding an \( \epsilon \)-sensitive tube commonly used in support vector regression
Figure 3.1: Quantile regression conducted on a data of sine wave with noise. Different values of $\theta$ were used representing the different quantiles estimated.

(SVR) to allow for a small magnitudes of errors to go unpenalized. The $\epsilon$-sensitive tube makes the classifier less sensitive to observations close to the margin by not penalizing errors within the $\epsilon$-sensitive tube. This can be useful for datasets that are noisy at the margin and allow for different penalization to occur for each of the classes. An asymmetric linear penalization property can be attained by allowing for different magnitudes for either sides of the $\epsilon$-sensitive tube. We included the $\epsilon$-sensitive tube to the pinball loss to create a generalized loss function

$$l_\theta = \begin{cases} 
-\rho_1(y - f(x)) & \text{if } (y - f(x)) < \epsilon_1 \\
0 & \text{if } \epsilon_1 \leq (y - f(x)) < 0 \\
0 & \text{if } 0 \leq (y - f(x)) < \epsilon_2 \\
\rho_2(y - f(x)) & \text{if } (y - f(x)) > \epsilon_2 
\end{cases} \quad (3.6)$$

The loss function is split into four distinct parts, each with different penalization magnitudes. Figure 3.2 shows the resulting loss function. The figure
Figure 3.2: The figure shows the penalty resulted from the error $y - f(x)$. The parameters changes either the length or slope of the lines they are closest to. $\epsilon_1$ and $\epsilon_2$ changes the length of the lines and $\rho_1$ and $\rho_2$ changes the magnitude of the slopes. depicts the different parts of the loss function that can be adjusted. One can change the magnitude of an SVM loss by specifying the linear function of the pinball loss or the size of either side of the $\epsilon$-sensitive tube. Both sides of the loss can be customized independently from each other.

We can modify each part of the proposed loss function by adjusting the respective magnitudes: $\epsilon_1, \epsilon_2, \rho_1, \rho_2 \in \mathbb{R}_{\geq 0}$ each of the sections have parameters that specify the magnitude in which to apply in the training process of the SVM. Varying the magnitude of $\epsilon_1$ and $\epsilon_2$ elongates or shortens the flat margins on either side of the loss. Increasing the magnitude of $\rho_1$ and $\rho_2$ increases the slopes for the non-zero gradient lines. The modifications provides the ability to customize loss functions for asymmetric problems. We denote the proposed asymmetric SVM as ASVM.

3.4.1 Loss Optimization

Training an ASVM, is similar to training a regular SVM where parameters $w$ and $b$ are found such that a maximal hyperplane is achieved between the
classes. To achieve that, the following QP problem is solved.

$$\min_w \frac{||w||^2}{2} + C \sum_{i=1}^{N} (\rho_1 \xi_i + \rho_2 \xi_i^*)$$

subject to

$$y_i - w'x_i - b \leq \xi_i + \epsilon_1$$

$$w'x_i + b - y_i \leq \xi_i^* + \epsilon_2$$

$$i = 1, 2, ..., N$$

$$\xi_i, \xi_i^* \geq 0$$

The QP problem is typically solved in the dual formulation. To obtain the dual form, we first find the Lagrangian primal,

$$L_p = \frac{||w||^2}{2} + C \sum_{i=1}^{N} (\rho_1 \xi_i + \rho_2 \xi_i^*)$$

$$- \sum_{i=1}^{N} \alpha_i (\xi_i + \epsilon_1 - y_i + w'x_i + b)$$

$$- \sum_{i=1}^{N} \alpha_i^* (\xi_i^* + \epsilon_2 + y_i - w'x_i - b)$$

$$- \sum_{i=1}^{N} (\eta_i \xi_i + \eta_i^* \xi_i^*)$$

The derivatives are

$$\frac{\partial L_D}{\partial b} = \sum_{i=1}^{N} \alpha_i^* - \sum_{i=1}^{N} \alpha_i \Rightarrow \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0$$

$$\frac{\partial L_D}{\partial w} = w - \sum_{i=1}^{N} \alpha_i x_i + \sum_{i=1}^{N} \alpha_i^* x_i \Rightarrow w = \sum_{i=1}^{N} x_i (\alpha_i - \alpha_i^*)$$

$$\frac{\partial L_D}{\partial \xi_i} = C \rho_1 - \eta_i - \alpha_i = 0 \Rightarrow C \rho_1 = \eta_i + \alpha_i$$

$$\frac{\partial L_D}{\partial \xi_i^*} = C \rho_2 - \eta_i^* - \alpha_i^* = 0 \Rightarrow C \rho_2 = \eta_i^* + \alpha_i^*$$
Substituting the partial derivatives into $L_p$ the Lagrangian dual formulation of the problem is obtained

$$
\max_{\alpha_i, \alpha_i^*} \sum_{i=1}^{N} y_i (\alpha_i - \alpha_i^*) - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i - \alpha_i^*) (\alpha_j - \alpha_j^*) x_i^t x_j \\
- \sum_{i=1}^{N} (\epsilon_3 \alpha_i + \epsilon_4 \alpha_i^*)
$$

subject to

$$\sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0$$

$$\alpha_i \in [0, \rho_1 C]$$

$$\alpha_i^* \in [0, \rho_2 C]$$

(3.8)

The dual formulation can now be solved using quadratic programing solvers.

To determine a class, the following expression is applied and the sign determines what the class is.

$$f(x) = \text{sign} \left( \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) x_i^t x + b \right)$$

(3.9)

The formulation retains the nice properties of SVM that allow for kernels to be used. As the objective function in (3.8) depends on the predictions only though the inner projects of the $x$’s, the kernel mapping of the original data can be substituted in place of the inner product. The resulting output can then be calculated via the below function.

$$f(x) = \text{sign} \left[ \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) K(x_i, x) + b \right]$$

(3.10)

### 3.4.2 Parameter Values for Asymmetric Parameters

Choosing asymmetric parameters depends on the user’s utility of recall versus precision. If heavy penalization is assigned to a class, that class will be predicted less by the ASVM and the predictions will be more precise. For
example, increasing the precision of the class + means reducing the magnitude of $\rho_1$ and increasing the magnitude of $\epsilon_1$. As we reduce $\rho_1$ and increase $\epsilon_1$, we will eventually achieve an ASVM that predicts class - for all cases. Conversely, by increasing $\rho_1$ and reduce $\epsilon_1$, we will eventually get an ASVM that predicts class + for all cases. We are only interested in the range of values where the ASVM is able to predict both classes. We call the range of these values the range of operability.

Before we find the range of operability for the $\rho$’s, we need to establish a relationship between $C$, $\rho_1$, and $\rho_2$ by observing the second term of the objective function of equation 3.7. It is trivial to show that $C$ is the magnitude of penalization and the $\rho$’s being the proportion of $C$ that is assign to each side of the loss. Thus we can assume the general constraint for $\rho_1$ and $\rho_2$

$$\rho_1 + \rho_2 = 1; \rho_1, \rho_2 \geq 0 \quad (3.11)$$

The range of operability for $\rho_1$ involves the finding $\rho_1^{(\text{max})}$ and $\rho_1^{(\text{min})}$. To do so, we set $\rho_1$ close to 0 to train the SVM. We should observe that the predicted values are undifferentiated (i.e. all predicted values are the same). We increase $\rho_1$ by an appropriate step size, retrain, and analyze the output again to check if there is differentiation. $\rho_1^{(\text{min})}$ is the min value when there is differentiation between the predicted values. A similar diagnostic can be conducted to find $\rho_1^{(\text{max})}$. Discovering the range of operability of $\rho_1$ also gives us the range of operability of $\rho_2$ by equation 3.11.

The $\epsilon$’s are less dependent on each other. The higher an $\epsilon$ is, the less penalization will occur on that respective side. To accommodate asymmetry, we set $\epsilon_1 \neq \epsilon_2$. We can implement an accounting for this by using the following constraint.
\[ \epsilon_1 + \epsilon_2 = k, \ k \geq 0 \quad (3.12) \]

The range of operability can also be achieved in a similar fashion of that of the \( \rho \)'s. By knowing the range of operability, it is then up to the user to decide the trade-off they want between precision and recall. Such a problem, which is not discussed in the research, can be thought of as the following equation

\[
\max_{c, \epsilon_1, \epsilon_2, \rho_1, \rho_2} \alpha \text{(Precision)} + \beta \text{(Recall)} \quad (3.13)
\]

\( \alpha \) and \( \beta \) are the quantitative benefit assigned to precision and recall respectively.

### 3.4.3 Extensions to ASVM

From Figure 3.2, we see that \( \rho \) penalizes a wrong observation linearly based on the distance of the observation to the hyperplane where as \( \epsilon \) penalizes data points the same independent of how far the data points are from the hyperplane. As these two parameters affect the final ASVM differently, we can, instead of training with both parameters simultaneously, train two separate ASVM, one for each parameter. By training two separate ASVMs, one that has \( \rho \) optimized for positive precision, and one that has \( \epsilon \) optimized for positive precision, we can then classify an observation as positive only if both ASVMs classified that observation as positive. We call such an ensemble paired ASVM (pASVM). pASVM can also be adjusted such that it can manage the imbalance data set problem. We can modify pASVM such that an observation is classified positive if either of the ASVMs classified the class as positive.

A further extension would be to introduce a quad ASVM (qASVM). Up to this point we cared only about the precision for classifying one of the classes
and thus have trained separate ASVMs for the positive class and negative class and have ignored the possibility that an observation can be classified both as positive and negative by the positive trained ASVM and negative trained ASVM simultaneously. We want to ensure that a data point $X_i$ is not classified as both positive with the positive ASVM and negative with the negative ASVM but that both ASVMs agree on its classification. The synergy of both a positive and negative precision optimized ASVM can potentially provide better precision when predicting both classes. With qASVM, four ASVMs are trained. Each of the four have one of the four parameters, $\rho_1, \rho_2, \epsilon_1, \epsilon_2$, optimized. This results in 2 ASVMs that are optimized for positive class precision and 2 ASVMs that are optimized for negative class precision. To classify we apply the following

$$f^{qASVM}(x) = \begin{cases} 
1 & \text{if } \sum_{i=1}^{4} f^{ASVM}_i(x) = 4, \\
0 & \text{if } -4 < \sum_{i=1}^{4} f^{ASVM}_i(x) < 4 \\
-1 & \text{if } \sum_{i=1}^{4} f^{ASVM}_i(x) = -4.
\end{cases}$$

The qASVM is able to predict both classes with high precision and as a result is a symmetric predictor unlike the pASVM and ASVM. When we apply qASVM for prediction, we obtain a neutral class (class 0) as one of its outputs. We label any observation in this class as observations that we cannot predict. We want our predictions to be accurate and so we want to have an output to denote observations that we have no confidence in classifying.
3.5 Experiments

3.5.1 ASVM on Simulated Data

Simulated data was fabricated to test the effectiveness of the asymmetric loss SVM. We simulated two bivariate normal distributions, one for the positive class and one for the negative class, on a two dimensional plane. A total of 400 observations were simulated for this experiment. As a control for ASVM, a regular SVM by Vapnik (1998) and a median ASVM where $\rho_1 = \rho_2 = 0.5$ and $\epsilon_1 = \epsilon_2 = 0$ were used as a benchmark and control respectively. The parameters, $\sigma$, for the Gaussian kernel and the penalizing parameter $C$ were held constant throughout the experiments at a value of 1 so as to show the marginal improvements that are achieved by adjusting the parameters of the loss function. We performed a five fold cross-validation and made predictions on the outsample for each fold. 10 replicates are done in total and the average results were tallied.

For this experiment, $\rho_1$ was adjusted to 0.4 and then 0.6 to make the model more prone to classifying a positive and positive class respectively. To show how the $\epsilon$-sensitive tube parameters ($\epsilon_1$ and $\epsilon_2$) can obtain better accuracy than that of the median SVM, each of the $\epsilon$’s was set to 1 and the other was set to 0. The trained SVMs utilize the Gaussian kernel function.

The heat map in Figure 3.3 show the classification space of the SVMs for one of the replicates. The light regions represent positive prediction while the darker regions represent negative prediction. The pluses (+) and the zeros (o) represents the training data of the positive and negative classes, respectively. From the results shown in Table 3.1, the median ASVM differs very little from the regular SVM. By adjusting $\rho_1$ to 0.4, the precision of
positive classification increases. In the other direction, when $\rho_1$ was adjusted to 0.6, the ASVM becomes more precise at identifying the negative class, with the precision increasing from 0.83 to 0.89. Comparing Figure 3.3(c) with Figure 3.3(d), a distinct change in area boundaries for classification can be observed. When $\rho_1 = 0.4$, there is a larger area in the space of the x’s where a point can be classified as a negative class. Changing $\rho_2$ to 0.6 decreases this area resulting in lesser area being classified as the negative class as well as improving the precision for negative classification.

Changing the values of $\epsilon_1$ and $\epsilon_2$ yield similar results. When $\epsilon_1 = 1$, the precision for the positive class increases. The heat maps of Figure 3.3(e) illustrates that the classification of positive classes are concentrated in the high density of positive class area in the $x$ space. The converse is true when $\epsilon_1 = 0$ and $\epsilon_2 = 1$ which is when the ASVM is optimized for the negative class. From Figure 3.3(f) we see that classification of the negative class is only concentrated in areas of high concentration of negative class observations and the rest of the area in the figure is classified positive. The resulting precision for the negative class increases from that of the baseline median ASVM. What is common with the heat maps of the ASVM is that the class where precision is optimized also had a higher recall. The fewer classified observation in the more risk adverse class is a trade off that one has to make in order to improve precision.

### 3.5.2 ASVM on Heart Disease Data

To show how asymmetric SVM can be applied to the healthcare industry, the statlog heart dataset obtained from the UCI website was used (Frank and Asuncion, 2010). The data consists of 270 observations of patients who may
$\rho_1 \quad \rho_2 \quad \epsilon_1 \quad \epsilon_2 \quad \text{TP} \quad \text{FN} \quad \text{FP} \quad \text{TN} \quad \text{Precision} \quad \text{Recall} \quad \text{Precision} \quad \text{Recall} \quad \text{Precision} \quad \text{Recall} \quad \text{Precision} \quad \text{Recall}$

<table>
<thead>
<tr>
<th></th>
<th>SVM</th>
<th>0.5</th>
<th>0.5</th>
<th>0</th>
<th>0</th>
<th>167</th>
<th>33</th>
<th>37</th>
<th>163</th>
<th>0.82(0.01)</th>
<th>0.83(0.01)</th>
<th>0.84(0.01)</th>
<th>0.82(0.01)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASVM</td>
<td>0.4</td>
<td>0.6</td>
<td>0.0</td>
<td>0.0</td>
<td>144</td>
<td>56</td>
<td>18</td>
<td>182</td>
<td>0.89(0.01)</td>
<td>0.77(0.01)</td>
<td>0.72(0.01)</td>
<td>0.91(0.01)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>0.4</td>
<td>0.0</td>
<td>0.0</td>
<td>182</td>
<td>18</td>
<td>60</td>
<td>140</td>
<td>0.75(0)</td>
<td>0.89(0.01)</td>
<td>0.91(0.01)</td>
<td>0.7(0.01)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>0.5</td>
<td>1.0</td>
<td>0.0</td>
<td>116</td>
<td>84</td>
<td>8</td>
<td>192</td>
<td>0.94(0.01)</td>
<td>0.7(0.01)</td>
<td>0.58(0.01)</td>
<td>0.96(0.01)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
<td>195</td>
<td>5</td>
<td>85</td>
<td>115</td>
<td>0.7(0)</td>
<td>0.96(0.01)</td>
<td>0.98(0)</td>
<td>0.58(0.01)</td>
</tr>
</tbody>
</table>

Table 3.1: The table shows the results of a simulated bi-variate normal data in two-dimensional space. The results are the average for 10 replicates of a five fold cross-validated experiment. Standard deviations are shown in the parenthesis. As shown from the results, varying the values of the $\rho$’s and $\epsilon$’s affects the precision and recall.
Figure 3.3: Heat map of the classification space for various values of $\epsilon$ and $\rho$. The lighter regions are classified as positive and the darker regions are classified as negative.
or may not have heart disease. The observations that have heart disease were assigned as the positive class and those that do not as the negative class. There are 13 different attributes that range from age and sex, to maximum heart rate and fasting blood sugar. Having a high precision in predicting heart disease is beneficial to health care providers as the cost of misdiagnosis is higher than that of failure to diagnose. By using a learner that predicts patients with heart disease or without heart disease with high precision, we are able to diagnose patients with high confidence. To assess the performance of the method, a regular SVM and a median SVM were used to establish a benchmark and a baseline for how asymmetric learning can add value. 10 replicates of five-fold cross-validations were conducted for each of the learners. The results for various parameter values of the asymmetric SVM are shown in Table 3.2.

The precision of the positive class compared to the median SVM improved from 0.80 to 0.91 as we decrease the value of $\rho_1$ and the precision of the negative class improved from 0.80 to 0.91 when we increased the value of $\rho_1$. Improvement in positive class precision was also achieved by adjusting $\epsilon_1$ to 1 which resulted in a 0.19 increase in precision. The negative class precision increased when we increased the value of $\epsilon_2$ to 0.95.

We next adjusted both $\rho$ and the $\epsilon$-sensitive tube of the loss function simultaneously. The simultaneous adjustment of two parameters offers greater flexibility to how errors are penalized. Setting $\rho_1 = 0.3$ and $\epsilon_1 = 0.5$ increased the positive class precision to 0.89. To optimize the precision of the negative class, $\rho_1$ was set to 0.7 and $\epsilon_2$ to 0.5, increasing the precision from the baseline to 0.91.

Table 3.2 show that when we optimized the ASVM precision, we are able to get precision values that out-perform the regular SVM. It must be noted
Table 3.2: The table shows the results of the heart disease data set when evaluated by 10 replicates of a five-fold cross-validation for each learner. The values are the average of the replicates. Standard deviations are shown in the parenthesis.
that although the accuracy got better, the improvements also resulted in lesser observations being classified for the class being optimized. As a result, when we increase precision, we inadvertently also increase the recall.

An alternative to adjusting both the $\rho$ and $\epsilon$ parameters in a single SVM, is to use the pASVM. Applying pASVM to the data set, we found that the results yield better accuracy than that of adjusting both sets of parameters within one SVM in both positive and negative precision. Table 3.3 shows the results for the pSVM experiment.

Finally, we apply qASVM to the data to see if we can achieve a symmetric learner that has optimized the precision for predicting both the positive and negative class. Table 3.4 shows the performance of the results for the qASVM. The precision was 0.87 for the positive class and 0.89 for the negative class. Thus the method shows improvement in both classes from the baseline of the median SVM.

### 3.5.3 ASVM on Class Imbalance Problem

To show how having an asymmetric loss function can be useful with datasets that have class imbalance, a dataset retrieved from the Blood Transfusion Service Center in Hsin-Chu City in Taiwan was used (Yeh et al., 2009). The set has five predictor variables that were used to classify whether or not a person will donate blood. The dataset, consisting of 748 observations, has only 23% of rows that are in the positive class (donated blood). For such problems, using a standard SVM might result in all the data being classified as negative. It is thus beneficial to apply ASVM to increase the number of observations being classified as positive. As we allow more observations to be classified as positive, we risk reducing the precision of prediction. However, the objec-
<table>
<thead>
<tr>
<th>$\rho_1$</th>
<th>$\rho_2$</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
<th>TP</th>
<th>FN</th>
<th>FP</th>
<th>TN</th>
<th>Precision (+)</th>
<th>Recall (−)</th>
<th>Precision (+)</th>
<th>Recall (−)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td></td>
<td></td>
<td></td>
<td>90</td>
<td>30</td>
<td>26</td>
<td>124</td>
<td>0.78(0.01)</td>
<td>0.81(0.01)</td>
<td>0.75(0.01)</td>
<td>0.83(0.01)</td>
</tr>
<tr>
<td>ASVM</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>89</td>
<td>31</td>
<td>24</td>
<td>126</td>
<td>0.79(0.02)</td>
<td>0.8(0.01)</td>
<td>0.74(0.02)</td>
<td>0.84(0.02)</td>
</tr>
<tr>
<td>pASVM</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
<td>16</td>
<td>92</td>
<td>1</td>
<td>148</td>
<td>0.96(0.03)</td>
<td>0.62(0)</td>
<td>0.13(0.02)</td>
<td>0.99(0.01)</td>
</tr>
<tr>
<td>(+)</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.58(0.01)</td>
<td>0.95(0.02)</td>
<td>0.95(0.01)</td>
<td>0.25(0.02)</td>
</tr>
<tr>
<td>pASVM</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
<td>114</td>
<td>2</td>
<td>84</td>
<td>38</td>
<td>0.58(0.01)</td>
<td>0.95(0.02)</td>
<td>0.95(0.01)</td>
<td>0.25(0.02)</td>
</tr>
<tr>
<td>(−)</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.58(0.01)</td>
<td>0.95(0.02)</td>
<td>0.95(0.01)</td>
<td>0.25(0.02)</td>
</tr>
</tbody>
</table>

Table 3.3: Asymmetric paired SVM results (pASVM). The first 2 rows show the results for a regular and median SVM. The remaining rows have 2 sets of parameters that are for the two ASVMs in the pASVM. The values, which are the average for 10 replicates of the five-fold cross-validation. Standard deviations are shown in the parenthesis.
<table>
<thead>
<tr>
<th></th>
<th>TP</th>
<th>FN</th>
<th>FP</th>
<th>TN</th>
<th>Precision (+)</th>
<th>Recall (+)</th>
<th>Precision (−)</th>
<th>Recall (−)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>90.3</td>
<td>29.7</td>
<td>25.6</td>
<td>124.4</td>
<td>0.78(0.01)</td>
<td>0.81(0.01)</td>
<td>0.75(0.01)</td>
<td>0.83(0.01)</td>
</tr>
<tr>
<td>ASVM (Median)</td>
<td>88.9</td>
<td>31.1</td>
<td>24</td>
<td>126</td>
<td>0.79(0.02)</td>
<td>0.8(0.01)</td>
<td>0.74(0.02)</td>
<td>0.84(0.02)</td>
</tr>
<tr>
<td>qSVM</td>
<td>45.5</td>
<td>10.8</td>
<td>6.8</td>
<td>83.5</td>
<td>0.87(0.03)</td>
<td>0.89(0.01)</td>
<td>0.38(0.03)</td>
<td>0.56(0.01)</td>
</tr>
</tbody>
</table>

**Table 3.4**: Results for the qASVM compared against the regular and median SVM. The values are the average for 10 replicates of the five-fold cross-validation. Standard deviations are shown in the parenthesis.
tive of this experiment is to show that ASVM and pASVM have the ability
to increase the number of TP without severe degradation of precision. The
predictor variables are first standardized to mean 0 and standard deviation 1
on the full dataset to prevent the variables with high magnitudes from dom-
inating the models. Like the heart disease data, the parameter of the kernel
and penalizing parameter $C$ were held constant at 1. To try and recreate the
testing scenario of Yeh et al. (2009) paper where 500 observations were ran-
domly picked to be insample and the remainder to be outsample, a three-folds
cross-validation was used and the results are presented in Table 3.5.

<table>
<thead>
<tr>
<th></th>
<th>$\rho_1$</th>
<th>$\rho_2$</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
<th>TP</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td>0.40(0.05)</td>
</tr>
<tr>
<td>ASVM</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0.64(0.23)</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>0.41(0.13)</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>13</td>
<td>0.43(0.09)</td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
<td>0.7</td>
<td>26</td>
<td>0.37(0.04)</td>
</tr>
<tr>
<td>pASVM</td>
<td>0.75</td>
<td>0.25</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0.40(0.04)</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.5: The results are the average of 10 replicates for a three-fold cross-
validation. The counts are round to the nearest integer. Standard deviations
are shown in the parenthesis.

As the objective is to increase the number of cases predicted positive while
maintaining reasonable precision, the loss function is setup to penalize FN
more. The median ASVM had a higher precision than that of the regular
SVM at the cost of lower TP counts. Setting $\rho_1 = 0.7$, the positive class
precision was 0.41 and the count of TP increased to 10 from the median SVM.
Setting $\epsilon_2$ to 0.5 resulted in an increase in the number of TP to 13, which is the
same as the regular SVM. Compared to the regular SVM, this configuration obtained better precision. Instead of just adjusting a single parameter, both the $\epsilon$ and $\rho$ parameters in the loss function can be adjusted simultaneously to increase the positive observations classified. $\rho_1$ was set to 0.6 and $\epsilon_2$ to 0.7. The adjustment of both sets of parameters simultaneously resulted in a much greater number of TP without significant degradation of precision. Next, we trained a pASVM to see if it can improve the number of observations classified as while maintaining high precision. The resulting precision is 0.40 and the number of TP increased to 12 from the baseline of the median ASVM.

3.6 Conclusion and Future Work

This paper explores the use of a generalized asymmetric loss function to achieve asymmetric loss and applies this loss function to SVM for classification problems. As an extension of the pinball loss function that is found in quantile regression, the generalized loss function is flexible enough to not only set different penalization standards for FP and FN, it is able to allow for the adjustment of an $\epsilon$-sensitive tube. This generalized loss function serves two different purposes: training classifiers when there is imbalanced data, and optimizing accuracy for a class of interest. This technique is useful in a variety of applications and the paper has shown that it is particularly beneficial in health care. In applying the method to heart disease where there are obvious advantages to doctors and insurance companies in finding a subset of patients with high or low risk of getting a heart disease, it was shown that the use of the loss function proposed in this paper improved the precision of the positive and negative class from that of the regular SVM. The trade off for the increase in accuracy came at the cost of a reduced number of observations being classified.
as the class that is being optimized. The use of blood transfusion data on the proposed loss function illustrate the benefits of using asymmetry to increase the number of observations classified as positive for an imbalanced dataset.

As the experiments presented served as examples to show how a change in the loss function could potentially improve the performance of accuracy and increase the number of classified observations, they do not present the optimized abilities of such loss functions. Much improvements can be achieved if a search is done to find a combination of the hyper-parameters of the kernel, $\epsilon$, $\rho$, and the penalizing parameter $C$, through cross validation. More importantly, using an asymmetric loss function requires the decision of the trade-off between the number of instances classified and accuracy. The general problem presented in Equation 3.13 could be further formalized and optimized by discovering the relationship between precision and classified instances and letting the user decide values of $\alpha$ and $\beta$. Finally, even though the paper has demonstrated the marginal benefit gained in the medical field by using asymmetric loss, the use of such methodology can serve a useful purpose in other disciplines and applications.
3.7 Addendum to the Generalized Asymmetric SVM: A comparison with regular SVM

3.7.1 Comparison of Regular SVM to a Generalized Asymmetric SVM

The proposed asymmetric SVM is valuable in problems where precision is required or when there is an uneven dataset. The difference between managing the problem endogenously versus tackling the problem exogenously is that an endogenous modification is expected to have non-linear optimizations that is taken into account by the QP that cannot be reproduced exogenously. The following results compare the performance of the regular hinge loss of an SVM and the proposed asymmetric loss function. With the hinge loss, the classification of a positive and negative class is based on whether the results of the SVM function (equation 3.15) is greater of lesser than a value \( \pi \) (typically \( \pi = 0 \)). Creating quantile loss exogenously, it is possible to favor one class over the other by just varying the value of \( \pi \).

\[
f(x) = \begin{cases} 
1 & \text{if } w'x_i + b \geq \pi \\
-1 & \text{if } w'x_i + b < \pi 
\end{cases}
\] (3.15)

Constructing a continuous ROC curve with the regular SVM is easy. Using the out-sample test dataset, the value of \( \pi \) is starts at a high level where all test data are classified as negative and is lowered until all the test data are classified as positive. Getting a continuous curve for the proposed loss function, on the other hand, is a little more tricky as each iteration would require the QP to be resolved. Figure 3.4 shows the proposed asymmetric loss function. While it would be computationally intensive to plot out the curve of the proposed loss function, various values of \( \rho \) and \( \epsilon \) can be used to find
points of on the ROC curve. With these points we can see if the proposed loss function actually performs better than that of the regular SVM by observing whether these points lie above or below the regular SVM curve.

3.7.2 Precision Comparison Methodology

A simulated dataset with a uniform random variable $x$ is constructed based on equation 3.16

$$f(x) = \text{sign}(0.95 \ast \cos(0.5 \ast (\exp(x) - 1)))$$ (3.16)

A training and test set of data is sampled and the machines are built on the training set and then tested on the test set. The SVM penalizing parameter $C$ and the kernel parameter $\gamma$ are each set to 3 different levels and all interaction of these levels are ran. The kernel used for this experiment is the gaussian kernel. The ROC curve is plotted exactly for the regular SVM based on the out-sample points. The proposed loss function is modified by changing either the values of $\rho$ or $\epsilon$ to get points on the ROC curve. Figure 3.5 shows the results by varying the values of $\rho$ to get different points on the graph. Figure 3.6 shows the results by varying the values of $\epsilon$ to get different points on the graph.

3.7.3 Uneven dataset Comparison Methodology

The same simulated data based on equation 3.16 is used to test the ability for both machines to be robust when training on uneven datasets. However, in this exercise, the in-sample dataset was sampled to produce a subset of in-sample data where only 13% of the observations had a class of -1. Both methods were then trained on this set of subset data and tested on the same
Figure 3.4: The figure shows the penalty resulted from the error $y - f(x)$. The parameters changes either the length or slope of the lines they are closest to. $\epsilon_1$ and $\epsilon_2$ changes the length of the lines and $\rho_1$ and $\rho_2$ changes the magnitude of the slopes.

test data used in section 3.7.2.
Figure 3.5: The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of $C$ and $\gamma$. The line represents that of the regular SVM and the * are points of the proposed loss function with different values for $\rho$. Except for Figure 3.5(c), the points of the proposed loss function generally are on top of the regular SVM.
Figure 3.6: The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of C and γ. The line represents that of the regular SVM and the * are points of the proposed loss function with different values for ε. The points of the proposed loss function generally at least that of the regular SVM.
Figure 3.7: The ROC curves above show the performance of a regular SVM and an proposed loss function for varying values of C and $\gamma$ based on an uneven dataset where the negative class only represented 13% of the training data. The line represents that of the regular SVM and the * are points of the proposed loss function with different values for $\rho$. 

(a) $C=1$ and $\gamma=10$  
(b) $C=100$ and $\gamma=10$  
(c) $C=200$ and $\gamma=10$
Chapter 4

ASYMMETRIC RANDOM FOREST FOR FEATURE SELECTION

4.1 Introduction

The increase use of machine learning has drove up the need to find relevant features to reduce dimensionality and improve model prediction. Irrelevant features may hinder the learner and result in poor prediction. Thus there is a need to focus on learning from only the important features. Discerning the right features to use may differ between applications. Consider the problem of yielding high accuracy in prediction made in the literature (Masnadi-Shirazi and Vasconcelos, 2011). A categorical response variable may consist of multiple classes. However, a user of the data may only be interested in the precise prediction of one of the classes. Thus the accuracy of predicting the other classes is not as important and the learners should be trained to be able to predict the class of interest with high accuracy. Examples of applications where high accuracy matters ranges from the discipline of fraud detection to that of medical diagnosis. For online fraud detection, it is important to be precise in classifying fraud transactions as misclassifying a non-fraud transaction as fraud can lead to high costs such as the lost of a customer and a bad reputation. This cost outweighs the cost of classifying a fraud transaction as non-fraud. Finding features that provide high accuracy in the prediction model can only result in a better model for such cases.

This work attempts to find features that are apt at predicting the class of interest but not necessarily apt at predicting the other classes. Such features
can be found by modifying the random forest variable selection algorithm Genuer and Poggi (2010) such that features that are apt at predicting the class of interest are identified.

### 4.2 Background

In supervised learning, there are input features $\mathbf{X} = (X_1, X_2, \ldots, X_p)$ that is mapped to an output $y \in (0, 1)$ with a function $f(\mathbf{X}) = y$. The model $f(\cdot)$ is trained using the training data set $(\mathbf{x}_i, y_i), i = 1, \ldots, N$. The number of features that can be used as input variables, $p$, can be large. With a large number of features, the resulting learner could suffer from overfitting, the curse of dimensionality problem, or multicollinearity between predictors (Verleysen and Franois, 2005). Thus it is necessary to find the best subset of features in that minimizes the prediction error of the model.

Feature selection is an important part of machine learning as it provides a subset of variables that are useful for prediction. Separating the variables that are noise from those that harbor information about the response ultimately result in a better trained learner that produces lower error rate. Feature selection also reduces the number of variables used in a supervised learner. For some learners, the objective is to predict a specific value of the response with higher accuracy. These learners are known as asymmetric learners. For asymmetric learners, the only the features that are capable in predicting a specified response value are selected.

For this research we are interested primarily in the error of misclassifying class 1 and thus the objective differs from the conventional variable selection goal of reducing the prediction error of the model to reducing the misclassification rate of class 1.
There are three main ways variable selection is conducted (Saeys et al., 2007): filter, wrapper, and embedded. The filter selection techniques assess the intrinsic properties of the predictor and assigns a score. The variables with the best scores are the selected. The simplest filters come in the form of t-statistic or ANOVA (Jafari and Azuaje, 2006). Another example of a filter is the information gain metric used in CART (Breiman, 1984). Variables that result in high information gain splits are selected as useful variables. Wrappers utilize a learning algorithm as a black box to score subsets of variables according to their predictive power. The wrapper methodology was popularized by Kohavi et al. (1996) and is a simple and powerful way to address the problem of variable selection, regardless of the chosen algorithm. Some popular known wrapper algorithms include sequential forward and backward selection (Kittler, 1978), simulated annealing (Kirkpatrick et al., 1983), and randomized hill climbing (Skalak, 1994). For embedded methods, the search for a best subset of features is built into the classifier construction, and can be seen as a search in the combined space of feature subsets. This process may be more efficient in several respects. Unlike wrapper methods, embedded methods do not require the splitting of the training data into a training and validation set. Second, it reaches a solution faster by avoiding retraining a predictor from scratch for every variable subset investigated. Embedded methods are not new: decision trees such as CART (Breiman, 1984), for instance, have a built-in mechanism to perform variable selection. Random forest is an algorithm that is widely used in the literature as an embedded system for variable selection (Diaz-Uriarte and Alvarez de Andres, 2006) (Jiang et al., 2004).
4.2.1 Asymmetric Features

A review of the feature selection methods in the literature failed to shed light on selecting variables that are particularly good at asymmetric prediction. However, asymmetric variables appear to be prevalent in many problems tackled in the machine learning literature. Matzler et al. (2004) did a regression analysis to find out what attributes lead to better overall customer satisfaction. They showed that for variables of complaint handling, project management, and innovativeness, when these variables have low values, there is a significant effect on overall satisfaction. However, when these same variables showed high values, the effect on overall satisfaction is insignificant. Froyen et al. (1997) studied whether political pressures add significant explanatory power in monetary policy. They found that white house administrations that pressure the Federal Reserve to tighten monetary policy results in higher interest rates whereas in administrations that promote loose monetary policy does not affect the interest rates. Finally Karras (1996) identified money-supply shocks and their effects on output for a panel of 18 European countries and found that many different specifications and estimation methods strongly support asymmetry: negative money-supply shocks are shown to have a statistically significant effect on output, whereas the effect of positive shocks is statistically insignificant.

These examples showed that asymmetric features have values that are associated with a high accuracy in the class of interest. Even though asymmetric features appear throughout many disciplines, little thought has been given to applying such variables to attain higher accuracy for the specific class. Moreover, there has not been any specific methodology in the literature that caters
to finding the subset of such features from the feature set. Thus having a variable selection technique that is catered towards finding features that are asymmetric will aid in the solutions of asymmetric problems in the literature.

4.2.2 Loss functions in trees

For this research, we will be focusing on modifying the random forest algorithm by Breiman (2001). The random forest (RF) algorithm consists of the bagging of decision trees. It is within the trees where the proposed method will be employed. We will be looking specifically at binary split decision trees. In general, like the CART algorithm by Breiman (1984), a binary decision tree divides the predictor space by recursively splitting the data. Each splitting rule splits the data into two separate nodes $\nu^L$ and $\nu^R$. Each rule forms a decision boundary, and a leaf node is the final partition after all the rules are enforced. The classification of data is done at each node, $\nu$, by the following loss $L(\cdot)$, for a classification tree

$$\begin{equation}
\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma) = \min_{\gamma} \sum_{i=1}^{N} 1 - I(y_i = \gamma)
\end{equation}\quad (4.1)$$

where $\gamma \in Y$

The function $I(\cdot)$ is an indicator function that returns a value of 1 if its input argument is true and 0 otherwise. The value of $\gamma$ is the classified output of $\nu$ and it can be shown that the value of $\gamma$ is $\arg \max_{y} P(y|\nu)$ for a classification tree. The rules for creating the partitions of the feature space is done in the following fashion. For all observations that fall into node $\nu$, an optimal split is found by finding that split that results in the greatest information gain. The information gain is calculated with an impurity function which in the case of CART is the Gini index. Let $k \in \mathbb{K}$ be a class of the dependent variable and
$p_k$ be the probability of class $k$ in node $\nu$, then the Gini is as follows.

$$Gini(\nu) = 1 - \sum_k p_k^2$$

(4.2)

Every possible split of the data within $\nu$ is considered in finding the optimal split. Splitting of regions continue until no split that increases the information gain can be made. Finding optimal regions in this fashion often results in a local optimum as the splitting decision is greedy. Thus a way to manage this problem is by using more than one tree and aggregating the prediction of all the trees. A method that does just that is the random forest algorithm.

### 4.2.3 Random Forest

Random forest was created by Breiman (2001). The method grows $T$—number of trees and takes the vote of all the trees grown to make a prediction. The trees are grown with the use of decision trees with one exception: a random subset of the total number of variables instead of all the variables is searched on to find the greatest change in purity. The selection process of using a random subset of the total number of variables is used in calculating the Gini index mediates the problem of obtaining a local minimum that the CART algorithm suffers from.

Breiman (2001) showed that there are two ways the random forest algorithm can be modified to be used for variable selection. The first way is by random permutation. For each tree grown in the forest, classify the out-of-bag (OOB) observations and take the accuracy of the results. Then, for each predictor $X$, permute the variable values and reclassify the OOB observations. Compare the results to the initial accuracy. The most important variables are the ones that result in the greatest drop in accuracy.
The second method for variable selection finds the variables that contributed the most information gain as the ones that are important. Let \( \nu^L \) and \( \nu^R \) be the left and right child nodes of the node \( \nu \). \( |\cdot| \) represents the cardinality function. When a predictor is split on some impurity function such as the Gini index, the information gain, \( IG(\cdot) \), is calculated as follows.

\[
IG(\nu) = G(\nu) - \rho^L G(\nu^L) - \rho^R G(\nu^R) \tag{4.3}
\]

where \( \rho^L = \frac{|\nu^L|}{|\nu|} \) and \( \rho^R = \frac{|\nu^R|}{|\nu|} \tag{4.4} \)

For equation 4.3 we see that there are 2 main drivers that affect the value of \( IG(\nu) \). The first is \( G(\cdot) \) which is an impurity function such as the Gini index. The second is \( \rho^L \) and \( \rho^R \), which are the proportions of data that are split into the left node and right node respectively.

For each variable, the total \( IG \) is calculated by summing the \( IG \) from each split that used \( x \). The most important variable are then the ones that delivered the greatest total \( IG \). For the work done in this paper, we focus on variable importance using this method.

### 4.3 Asymmetric Variable Selection

For binary classifying problems with two classes, \( y \in \{0, 1\} \), variable selection such as the random forest (RF) variable selection methods select variables that are apt at predicting both the positive and negative class. No consideration is given to the possibility that certain variables are better at predicting one class of the response over the other classes. If we are only interested in accurately predicting one of the classes, it would be better to have a variable selection algorithm that selects variables that are predictive of that class.

In this section, four different modifications to the calculation of \( IG(\cdot) \) in the
tree algorithm are introduced. Equation 4.3 is modified by either changing the impurity function $G(\cdot)$ or the weights, $\rho^L$ and $\rho^R$, assigned to the impurities of the child nodes. Two different asymmetric Gini functions, $aG1$ and $aG2$ and two weight functions $\rho_{lg}$ and $\rho_{bin}$ are proposed.

We use a selection technique that modifies the tree algorithm such that there is a bias toward splits that result in child nodes that are pure in class 1. This requires us to treat splits that decrease the impurity of class 1 more importantly. From equation 4.3, we see that $IG(\cdot)$ depends on the impurity function $G(\cdot)$, $\rho^L$, and $\rho^R$ which is the weight given to the $G(\cdot)$’s. Asymmetry is implemented by changing the impurity function $G(\cdot)$ such that it has a bias toward one class. We can also achieve asymmetry by adjusting $\rho^L$ and $\rho^R$ to be greater when the resulting class is 1.

For a binary problem, the Gini index is as follows

$$G(\nu) = 1 - p_0^2 - p_1^2$$

(4.5)

where $p_0$ is the proportion of data that are in class 0 and $p_1$ is the proportion of data that are in class 1. We modify the Gini as follows such that the resulting tree is more susceptible to splits that results in an increase in purity for class 1.

$$aG1(\nu) = \min(1 - p_0^2 - p_1^2, 1 - p_1)$$

(4.6)

From the equation, if class 0 is the majority class, then $aG1(\nu)$ yields the same magnitude as a regular Gini index. Otherwise, $aG1(\nu)$ applies a linear function for penalization. Figure 4.1 illustrates the impurity of both the regular Gini and aG1 for a binary response variable. For nodes with equal proportions of a majority class, we want the impurity of a node with class 0 as a majority class to be lesser than or equal to the impurity of a node with class 1 as a
Figure 4.1: Impurity for different values of p.

majority class. For instance, if we have two nodes, one with 70% class 1 and the other with 70% class 0, we want the impurity of the node with 70% class 0 to be lesser than or equal to the impurity of the node with 70% class 1. This ensures that splits that result in child nodes with higher proportion of class 1 will be chosen over splits that result in child nodes with a higher proportion of class 0. We are then able to grow trees that have leafs with higher purity for class 1 and as the variable selection algorithm sums up all IG caused by node splits from trees, a variable that is more capable at predicting class 1 yields a higher score.

We can take the idea of \( aG1(\cdot) \) a step further and completely disregard the impurity when the class is 0. Equation 4.7 shows how such a impurity function, \( aG2(\nu) \) is calculated

\[
aG2(\nu) = \begin{cases} 
1 - p_1^2 - (1 - p_1)^2 & \text{if } p_1 > 0.5 \\
\ aG2(\nu^0) & \text{otherwise}
\end{cases}
\]  

In Equation 4.7, \( \nu^0 \) represents the parent node of \( \nu \). When the majority class
Table 4.1: Example dataset of an asymmetric variable $x$. When $x = 1$ then $y = 1$ but when $x = 0$, $y$ can be either 1 or 0. Thus $x = 1$ is a good predictor of class 1.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

of $\nu$ is 1 only then do we calculate the Gini of $\nu$, otherwise we use the impurity of $\nu^0$. In this fashion, $aG_2(\cdot)$ only produces an increase in $IG(\cdot)$ if the child node has a higher purity in class 1.

Consider Table 4.1 with variable $x$ and response $y$. We see that most of the values of $x$ is 0. From this example, when $x = 1$ then $y = 1$ but when $x = 0$, $y$ can be either 1 or 0. Thus $x = 1$ is a good predictor of class 1. An important thing to notice is that the number of observations where $x = 1$ is sparse and $x = 0$ most of the time. Such asymmetric features have the ability to classify class 1 with high accuracy but may only be able to classify a small number as class 1. In addition, from Table 4.1, assume we examine a candidate split where data with $x = 1$ is partitioned to $\nu^L$ and $x = 1$ to $\nu^R$. The value of $\rho^R$ will be much lower than $\rho^L$ thus giving little weight to $\nu^R$.

In Equation 4.4, $\rho$ is the proportion of data that got split to either of the child node and can be thought as the weight assigned to nodes. A problem arises if one of the child node has a low proportion of data partitioned to it but has a low impurity for class 1, the resulting low weight may dampen the high reduction in impurity caused by the split. To mitigate this issue, we can reduce the disparity of $\rho$ between the child nodes by taking the logarithm of the values of $|\nu|$ as follows

$$
\rho^L_{ig} = \frac{\log(|\nu^L|)}{\log(|\nu^L|) + \log(|\nu^R|)}
$$

$$
\rho^R_{ig} = \frac{\log(|\nu^R|)}{\log(|\nu^L|) + \log(|\nu^R|)}
$$

(4.8)
Another solution would be to ignore the information gain from nodes that predicts class 0. Doing this ensures that we only take into account the splits that result in an increase in purity for class 1. Accounting for only class 1 splits can be done by as follows

$$\rho_{bin}^L = I(p_1^L \geq 0.5)$$

$$\rho_{bin}^R = I(p_1^R \geq 0.5)$$

(4.9)

where $p_1^L$ and $p_1^R$ are the proportions of class 1 in the left node and right node respectively and $I(\cdot)$ is an indicator function that returns 1 if the condition is true and 0 otherwise.

4.4 Asymmetric Random Forest

An Asymmetric Random Forest (ARF) is trained by making modifications to the random forest feature selection algorithm. The algorithm is modified to select features that can predict class 1 with high accuracy. For the experiments in this paper, we employ Algorithm 1, which is the algorithm used to select features in a random forest.

**Algorithm 1: Asymmetric Random Forest Variable Selection**

Let the number of training cases be $N$, and the number of variables in the classifier be $M$. For each tree do

- Run RF with $\mathcal{I}G(\cdot)$

For each variable do

- For each node do

  - If $m$ is the variable used for splitting then
    - Tally the $\mathcal{I}G(m)$

Rank the tallied variables in descending order

In the algorithm, we modify $\mathcal{I}G(\cdot)$ by either changing either $\rho^L$ and $\rho^R$ or the impurity function, $G(\cdot)$, to their asymmetric counterparts previously
mentioned. The results yield should then be bias towards obtaining features that possess information on classifying class 1.

4.5 Experiment

4.5.1 ARF on Simulated Asymmetric Features

We simulated data sets to test if ARF is capable in detecting asymmetric features. The features we generated are binary variables and the response, \( y \), is coded as either 0 or 1. As the features are binary, we can think of these features as predictors by using the binary value of the feature as a prediction for the value of \( y \). We can calculate \( P(y = 1|x = 1) \) and \( P(y = 1|x = 0) \) from the simulated data. The accuracy of a feature in predicting class 1 can be calculated by \( \max[(y = 1|x = 1), (y = 1|x = 0)] \). The function \( \max[\cdot] \) returns the greater value of the two inputs.

We define the parameters \( \pi_1 \) and \( \pi_0 \) as the accuracy in predicting class 1 and class 0 respectively. Higher values of \( \pi_1 \) results in the feature being more accurate at predicting class 1 and higher values of \( \pi_0 \) results in the feature being more accurate at predicting class 0. The accuracy values \( \pi_1 \) and \( \pi_0 \) for each of the features generated are shown in Table 4.2. In total we simulated 30 features: 1 symmetric feature, 6 asymmetric features with high accuracy for class 1 (class 1 features), 6 asymmetric features with high accuracy on class 0 (class 0 features), and 17 noise features that do not have any predictive power. Feature 1 has high accuracy for both the positive and negative class. Feature 2 to 7 have high accuracy for class 1 and feature 8 to 13 have high accuracy for class 0. The rest of the features are noise features. In an ideal situation, the asymmetric random forest will pick only the variables that have high accuracy in the positive class (feature 1 to feature 7).
Table 4.2: Simulated Features for Asymmetric Variable Selection. The parameters $\pi_1$ and $\pi_2$ corresponds to the accuracy of predicting class 1 and class 0 respectively. Feature 2 to Feature 7 are known as class 1 features as they are more accurate at predicting class 1 than class 0. Feature 8 to Feature 13 are known as class 0 features as they are more accurate predicting class 0 than class 1.

<table>
<thead>
<tr>
<th>Feature</th>
<th>$\pi_1$</th>
<th>$\pi_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetric Feature 1</td>
<td>0.9</td>
<td>0.89</td>
</tr>
<tr>
<td>Class 1 Features 2</td>
<td>0.83</td>
<td>0.53</td>
</tr>
<tr>
<td>3</td>
<td>0.84</td>
<td>0.64</td>
</tr>
<tr>
<td>4</td>
<td>0.72</td>
<td>0.62</td>
</tr>
<tr>
<td>5</td>
<td>0.75</td>
<td>0.66</td>
</tr>
<tr>
<td>6</td>
<td>0.63</td>
<td>0.59</td>
</tr>
<tr>
<td>7</td>
<td>0.56</td>
<td>0.55</td>
</tr>
<tr>
<td>Class 0 Features 8</td>
<td>0.53</td>
<td>0.82</td>
</tr>
<tr>
<td>9</td>
<td>0.65</td>
<td>0.84</td>
</tr>
<tr>
<td>10</td>
<td>0.61</td>
<td>0.7</td>
</tr>
<tr>
<td>Features 11</td>
<td>0.66</td>
<td>0.75</td>
</tr>
<tr>
<td>12</td>
<td>0.6</td>
<td>0.64</td>
</tr>
<tr>
<td>13</td>
<td>0.54</td>
<td>0.55</td>
</tr>
<tr>
<td>Noise Features 14-30</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>
We performed 10 replicates and averaged the ranks of the class 1 variables, class 0 variables, and noise variables. The most relevant variable is given the rank of 1 and the least is given the rank of 30. We run the Asymmetric Random Forest for each of these replicates using various combinations of the $G(\cdot)$, $\rho$, and number of trees in the forest.

We tried 50, 100, and 200 trees for the forest and found that the average ranks did not deviate much in general. Table 4.3 shows the change in average ranks for the regular RF algorithm. The average ranks when the number of trees is 200 is shown in Figure 4.2. Ideally, all class 1 features should have a low rank on average. Feature 1, a feature that has yields high accuracy for class 1 and 0, is consistently ranked first for almost all the simulations. The class 1 features were selected to be important for the regular RF algorithm and the ARF algorithms. The class 0 features were not selected by the ARF algorithms but were selected by the regular RF algorithm. Overall, the table shows that the modifications to the RF algorithm results in low ranks for only variables that are apt at predicting class 1. The Wilcoxon-Mann-Whitney test (Sprent, 2000) is used to test the significance of the class 1 variables being selected over the class 0 variables and the noise variables. The results are shown in Table 4.4. Based on $\alpha = 0.05$, with the exception of the regular RF variable selection, we find that the average rank of class 1 features are lower than that of the noise variables. $\rho_{lg}$ with Gini is found not to be effective in selecting the class 1 asymmetric variables. Combining the use of $\rho_{lg}$ with either aG1 or aG2 results in the ability to pick class 1 features over class 0 features. aG1 with $\rho$ does not yield an asymmetric result. Combining aG1 with $\rho_{lg}$ or $\rho_{bin}$ results in significant lower ranks for the class 0 features over the class 1 features. When applying aG2 or $\rho_{bin}$, we see significant lower ranks for class 0 features for all
Table 4.3: Average ranks of class 1 variables from 10 replicates. The rank is out of 30 where the lower the rank the more important the variable is.

<table>
<thead>
<tr>
<th>$G(\cdot)\rho$</th>
<th>No. of Trees</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Gini $\rho$</td>
<td>50</td>
<td>29.4</td>
</tr>
<tr>
<td>Gini $\rho$</td>
<td>100</td>
<td>29.5</td>
</tr>
<tr>
<td>Gini $\rho$</td>
<td>200</td>
<td>29.5</td>
</tr>
</tbody>
</table>

combinations. This shows the robustness of both these modifications.

4.5.2 Asymmetric Forest on Simulated Asymmetric features that interacts

Asymmetric accuracy in predictors can happen under interaction between multiple predictors. Consider the data in Table 4.5. We see that when $x_1 = 1$ and $x_2 = 1$ then $y = 1$. However, no other combination of $x_1$ and $x_2$ results in a good prediction for $y = 0$. Neither $x_1$ or $x_2$ alone give high accuracy for predicting $y = 1$. Only the interaction between both variables results in a good asymmetric variable. In this experiment, we try to find asymmetric predictors that are useful when they undergo interaction.

In the following experiment, we decompose variables from Table 4.2 into two variables that do not have high accuracy in predicting class 1. The decomposition for each predictor is done as follows. Let $x_i \in (0, 1)$ be a binary variable. To decompose $x_i$ into $z_{ai}$ and $z_{bi}$ we do the following. If $x_i = 1$, then we set $z_{ai} = 1$ and $z_{bi} = 0$ or $z_{ai} = 0$ and $z_{bi} = 1$ with equal probability. If $x_i = 0$, then we generate $z_{ai} = 1$ and $z_{bi} = 1$ or $z_{ai} = 0$ and $z_{bi} = 0$ with equal probability. Now the resulting $z_{ai}$ and $z_{bi}$ are noise variables that do not contain any information about the response by themselves. However, if we take into consideration the interaction of $z_{ai}$ and $z_{bi}$, we will get the original
Figure 4.2: The graphs show the variable importance based on different ARF compared to the regular RF. In total, 10 replicates of 200 trees were grown for each simulation and the average results are reported. ARF is able to pick out the Class 1 features (feature 2 to 7) from class 0 and noise features while the regular RF is is unable to differentiate between class 1 and class 0 features.
Table 4.4: Results from simulated asymmetric variables. Average class 1 rank is the average rank of class 1 features. $\Delta$ Class 1 and 0 is the difference of the average rank of class 1 features from class 0 features. $\Delta$ Class 1 and Noise is the difference of the average rank of class 1 features from the noise features. The asterisks indicates the difference is significant at $\alpha = 0.05$. The first row with Gini and $\rho$ is the standard RF algorithm. The ARFs showed significant differences in the rankings of class 1 features from class 0 and noise features.

<table>
<thead>
<tr>
<th>$G(\cdot)$</th>
<th>$\rho$</th>
<th>Average Class 1 Rank</th>
<th>$\Delta$ Class 1 and 0</th>
<th>$\Delta$ Class 1 and Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gini</td>
<td>$\rho$</td>
<td>16.92</td>
<td>-3.08</td>
<td>-0.48</td>
</tr>
<tr>
<td>Gini</td>
<td>$\rho_{lg}$</td>
<td>12.95</td>
<td>-1.45</td>
<td>5.71*</td>
</tr>
<tr>
<td>Gini</td>
<td>$\rho_{bin}$</td>
<td>7.93</td>
<td>12.75*</td>
<td>9.26*</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho$</td>
<td>14.65</td>
<td>-2.33</td>
<td>3.13*</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho_{lg}$</td>
<td>10.47</td>
<td>2.12*</td>
<td>8.69*</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho_{bin}$</td>
<td>13.40</td>
<td>4.77*</td>
<td>2.75*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho$</td>
<td>7.45</td>
<td>6.53*</td>
<td>12.28*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho_{lg}$</td>
<td>7.28</td>
<td>4.62*</td>
<td>13.24*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho_{bin}$</td>
<td>6.22</td>
<td>5.13*</td>
<td>14.81*</td>
</tr>
</tbody>
</table>

Table 4.5: Example dataset of an asymmetric variable $x$. When $x = 1$ then $y = 1$ but when $x = 0$, $y$ can be either 1 or 0. Thus $x = 1$ is a good predictor of class 1.
features in Table 4.2 that are predictive of the response $y$.

We decomposed Feature 1 in Table 4.2 to two different features. We also decomposed two features (Feature 2 and 3 in Table 4.2) that are class 1 asymmetric variables and two features that are class 0 asymmetric variables (Feature 8 and 9 in Table 4.2) to eight features that are individually unpredictable. Finally, we added four noise features (feature 14, 15, 16, and 17), resulting in a data set of 14 predictors in total. The ARF algorithm with various combinations of $G(\cdot)$ and $\rho$ were ran. Like the previous experiment, we used 50, 100, and 200 trees and we found that the average ranks did not deviate much in general for both ARF and RF and thus there was no need to further increase the number of trees.

Table 4.6 shows the change in average ranks for the regular RF algorithm. Having a low rank means that the model considers the variable important and thus we would like to see that all class 1 features have a low rank on average. Feature 1a and 1b, which are the decomposition of Feature 1, is selected by all the algorithms. With the exception of the noise features the regular RF noticeably selected all features as important. However, the regular RF failed to distinguish the class 1 features from the class 0 features.

Figure 4.3 shows the average ranks of 10 replicates for the RF and ARF with 200 trees. The graphs show that although the regular RF ranks both class 0 and class 1 variables lower than the noise variables, it is unable to differentiate between class 1 from class 0 variables. The ARF algorithms on the other hand ranks class 1 variables lower than both class 0 variables and noise variables, selecting only the variables we are interested in.

Table 4.7 shows the average class 1 feature rank and the rank difference between class 0 features and the noise features. The Wilcoxon-Mann-Whitney
Table 4.6: Average ranks of decomposed features from 10 replicates. The prefix of the decomposed feature is the label of the original feature. The rank is out of 14 where the lower the rank the more important the variable is.

<table>
<thead>
<tr>
<th>G(·)</th>
<th>ρ</th>
<th>No. of Trees</th>
<th>Decomposed Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1a 1b 2a 2b 3a 3b</td>
</tr>
<tr>
<td>Gini</td>
<td>ρ</td>
<td>50 1.2</td>
<td>1.8 12.7 12.4 4.0 4.2</td>
</tr>
<tr>
<td>Gini</td>
<td>ρ</td>
<td>100 1.4</td>
<td>1.6 12.0 13.3 4.0 5.2</td>
</tr>
<tr>
<td>Gini</td>
<td>ρ</td>
<td>200 1.4</td>
<td>1.6 13.0 12.9 4.9 4.9</td>
</tr>
</tbody>
</table>

test was used to test the significance of the difference. As shown from the table, the Δ between class 1 and class 0 features is not significant with the regular RF but significant for the ARFs at α = 0.05. This indicates that the ARFs are better able to pick up class 1 variables. The regular RF variable selection was neither able to pick class 1 features over class 0 features, nor was it able to distinguish class 0 features over the noise features. The ARFs with aG1 and aG2 were able to pick class 1 features over class 0 features and the noise features, regardless of which ρ was used. The ARF with ρlg was not able to pick class 1 features when used with Gini. It was however able to do so with aG1 and aG2. The top performing algorithm was aG1 with ρbin, which yielded the greatest difference between class 1 and class 0 variables. The experiment showed that ARF is useful in selecting out asymmetric variables that interact.

4.5.3 Asymmetric features in financial disclosure data

Public companies that trade in the United States must have its directors, officers, and shareholders owning more than 10% of the firm file with the United States Securities and Exchange Commission a statement of ownership any time they conduct a transaction with a company’s stock. The filing is done
Figure 4.3: The graphs show the variable importance based on different ARF compared to the regular RF. 200 trees were grown for each simulation. ARF is able to pick out the Class 1 features (feature 2 to 7) from class 0 and noise features while the regular RF is is unable to differentiate between class 1 and class 0 features.
Table 4.7: Results from simulated asymmetric variables. Average class 1 rank is the average rank of class 1 features. ∆ Class 1 and 0 is the difference of the average rank of class 1 features from class 0 features. ∆ Class 1 and Noise is the difference of the average rank of class 1 features from the noise features. The asterisks indicates the difference is significant at $\alpha = 0.05$. The first row with Gini and $\rho$ is the standard RF algorithm. The ARFs showed significant differences in the rankings of class 1 features from class 0 and noise features.

<table>
<thead>
<tr>
<th>$G(\cdot)$</th>
<th>$\rho$</th>
<th>Average Class 1 Rank</th>
<th>∆ Class 1 and 0</th>
<th>∆ Class 1 and Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gini</td>
<td>$\rho$</td>
<td>8.93</td>
<td>-0.73</td>
<td>-0.55</td>
</tr>
<tr>
<td>Gini</td>
<td>$\rho_{lg}$</td>
<td>8.45</td>
<td>-0.48</td>
<td>0.63*</td>
</tr>
<tr>
<td>Gini</td>
<td>$\rho_{bin}$</td>
<td>4.58</td>
<td>5.53*</td>
<td>6.25*</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho$</td>
<td>8.30</td>
<td>0.45*</td>
<td>0.15</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho_{lg}$</td>
<td>6.40</td>
<td>3.35*</td>
<td>2.95*</td>
</tr>
<tr>
<td>aG1</td>
<td>$\rho_{bin}$</td>
<td>4.75</td>
<td>6.08*</td>
<td>5.13*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho$</td>
<td>5.90</td>
<td>3.88*</td>
<td>3.73*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho_{lg}$</td>
<td>6.53</td>
<td>1.15*</td>
<td>4.65*</td>
</tr>
<tr>
<td>aG2</td>
<td>$\rho_{bin}$</td>
<td>4.23</td>
<td>4.25*</td>
<td>6.92*</td>
</tr>
</tbody>
</table>
by the individual trading the security with the Form-4 document and the filing is made available by the SEC on its website (SEC, 2010). Based on the research done by Cicero (2009), executive trade on privileged information and early exercise is correlated with the peak of a stock price run. In the dissertation written by Wei (2006), it was found that companies where executives that exercise early and deep in the money showed underperformance in the future. For this experiment, we take a variety of variables associated with executive option exercises and run the variable selection algorithms on them. A total of 21 features were used and their description can be found in Table 4.8.

The response was the future under-performance of the firm following the event of a form 4 release. The under-performance was based on the firm’s stock price under-performing its peers by more than 5%. We used the data from the Form-4 for mid-cap firms of 2005 to test ARF on. This gave us a total of 4803 observations. Like the previous experiments, we ran 10 replicates for each combination of \( G(\cdot) \) and \( \rho \).

Due to limited computing power with only the use of a MacBook 2GHz Core Duo computer, we sampled 1000 observations of the total number of observations for each replicate. The total time taken to run this experiment still exceeded ten and a half hours. The top seven variables, which makes a third of the total number of variables, were selected as important for the results. It was surprising to note that there was not significant difference in results between replicates. The selected features are shown in Figure 4.4.

From Wei (2006), we know the asymmetric variables are \( \text{gainratio} \) and \( \text{avginmon} \) which corresponds to the amount gain from exercise the option and the moniness of the option respectively. The regular RF variable selection algorithm was unable to pick up either of these variables. Figure 4.4 shows
Table 4.8: Description of features used

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>price_to_sales_rank</td>
<td>Price to Sales rank over the top 5000 firms</td>
</tr>
<tr>
<td>avgtimeleft</td>
<td>Ratio of time left when exercised to the total length of the vest period</td>
</tr>
<tr>
<td>change</td>
<td>Ratio between firm’s previous 6 months performance and size of exercised options</td>
</tr>
<tr>
<td>loginmon</td>
<td>Log of moniness</td>
</tr>
<tr>
<td>avginmon</td>
<td>The average moniness of all exercises</td>
</tr>
<tr>
<td>avgvestprice</td>
<td>Average vest price</td>
</tr>
<tr>
<td>diffavgshares</td>
<td>Total shares exercised - averaged exercised over the past year</td>
</tr>
<tr>
<td>gainratio</td>
<td>Ratio of the gain sold from the exercise</td>
</tr>
<tr>
<td>logmktcap</td>
<td>Log of market capitalization</td>
</tr>
<tr>
<td>prevmon3ret</td>
<td>Stock returns over the past 3 months</td>
</tr>
<tr>
<td>sellratio</td>
<td>Percentage of exercised shares sold</td>
</tr>
<tr>
<td>shares1</td>
<td>Total shares exercised</td>
</tr>
<tr>
<td>sharesoveryear</td>
<td>Total shares exercised over the year</td>
</tr>
<tr>
<td>sharessold</td>
<td>Total shares sold</td>
</tr>
<tr>
<td>sharessold_sellvalue</td>
<td>Total value of shares sold</td>
</tr>
<tr>
<td>ssprevmon3ret</td>
<td>Stock returns over past 3 months - benchmark returns over past 3 months</td>
</tr>
<tr>
<td>totalexervalue</td>
<td>Total exercise value</td>
</tr>
<tr>
<td>totalvestvalue</td>
<td>Total vest value</td>
</tr>
<tr>
<td>volatility</td>
<td>Volatility of stock over the past 3 years</td>
</tr>
<tr>
<td>momentum</td>
<td>Momentum of stock</td>
</tr>
<tr>
<td>momentum_diff</td>
<td>Change in momentum of stock</td>
</tr>
</tbody>
</table>
Figure 4.4: Variable Importance for Form 4 data based on 200 Trees. The lower ranks are more important.
the average feature importance rank for various combinations of $G(\cdot)$ and $\rho$. At 200 trees $aG1$ and $aG2$ both variables were ranked much lower than when it was ranked by the regular RF. Holding Gini constant and varying $\rho$ did not result in the asymmetric variables ranking lower. The results were similar to that of the regular RF. A reason for the lack of detection can be attributed to the nature of the data that makes the ARF insensitive to asymmetry. Further research needs to be done to understand the circumstances when this occurs.

4.6 Conclusion

We introduce the notion of asymmetric features where the feature is apt at predicting a particular response class but not necessarily apt at predicting the other response classes. An asymmetric RF for variable selection, constructed by modifying the information gain function, was proposed. We suggested two different $G(\cdot)$, aG1 and aG2, and two different $\rho$, $\rho_{lg}$ and $\rho_{bin}$ to account for asymmetry in the information gain function. These modifications were able to distinguish variables that are better at selecting out class 1 in our experiments. In addition, we show these changes had the ability to also pick up asymmetric feature interactions. The asymmetric RF was conducted on financial data and was shown to be able to pick up variables that are known to be asymmetric. Further research needs to be done to better understand the scenarios that work best for each combination of $G(\cdot)$ and $\rho$. In summary, the ARF proposed serves as a good tool to use to seek out asymmetric variables.
Chapter 5

MATCHED RANDOM FOREST

5.1 Introduction

Grouping of data is commonly done in statistics to analyze similar data together. The clustering of such data can reduce variance in statistical estimation and provide a better understanding of the data. For instance in the finance field it is well known that companies should be grouped into respective peer groups so that better analysis can be conducted (Daniel et al., 1997) (Faulkender and Yang, 2010) (Bizjak et al., 2008) (Marsili, 2002).

The most rudimentary type of clustering is that of a 1:1 matched data. A 1:1 matching study is commonly called a case-control study and is used in observational data where the untreated unit is typically denoted as the control and the treated unit is denoted as a case (Rubin, 1973). The application of 1:1 matched studies is widespread in clinical data sets (Berg et al., 2010) (Austin, 2008) (Pregibon, 1984). For instance, Berg et al. (2010) studied the risk of cardiovascular disease caused by acromegaly by conducting a 1:1 matched study. Subjects with acromegaly (cases) were matched with subjects from the general population (control). The control subjects were picked based on having similar demographic characteristics such as age and sex with their control counterparts. A matched experiment was conducted as oppose to a randomized experiment as it may not be reasonable to compare random subjects of acromegaly patients to random subjects of the general population. For example, factors such as age and sex are important variables that may
be distributed differently in the two groups causing bias in the data. Matching mitigates such bias in the statistical inference from the data by grouping similar subjects.

Berg et al. (2010) wanted to determine if variables such as blood pressure and cholesterol levels, which are indicators of heart disease, are different between the case and control. When studying matched data, discovering the variables that can distinguish the cases and controls provides a better understanding of which variables are important.

A common way to analyze matched studies is by using conditional logistic regression (CLR) (Hosmer and Lemeshow, 2000). CLR is used in matched data to detect features that can distinguish the case and control. Details of the method are discussed in Section 5.3. One can use CLR to determine if there is a significant difference in variable values between the cases and controls, the magnitude of the difference, and the direction of the difference (i.e. if cases have higher values than controls or vice-versa). However, the method is based on a linear model. Non-linear relationships where high and low values of a variable may be attributed to the cases and values in between attributed to the control may not be detectable by CLR. Such data sets are found in practice. For example, in a study of the link between cardiovascular death and myocardial infection, Salonen et al. (1982) found that serum selenium was associated with cardiovascular risk only at very low selenium concentrations and such a relationship was different in regular concentrations. Such phenomena exhibits a non-linear relationships between case and control where the relationship is dependent on the region where the observation is located. In addition to not being able to detect non-linear relationships, such linear models are not apt at managing categorical data.
One can add polynomial and cross-product terms to CLR models to attempt to handle non-linearities. Furthermore, one can add indicator variables to attempt to account for categorical variables. To learn general non-linear functions, linear modeling methods require a degree of complexity that grows exponentially with the number of inputs. That is, as the number of inputs increases, the number of interactions and polynomial terms required in a regression model grows exponentially (SAS, 2012). When transforming a categorical variable to indicator variables, the number of indicator variables required is dependent on the number of categories in a categorical variable. Hence, variables with a lot of categories results in a large number of indicator variables added to the model. The adding of more variables come at a cost of a less parsimonious model, which can lead to problems like overfitting and multicollinearity (Hawkins, 2004). This makes CLR impractical for data sets with a high number of variables.

To fully understand the effect a variable has on the response in the presence of non-linear relationships we need to first discover if the variable can distinguish a case from a control. We would then need to determine the sub-regions where the variable effect is linear. In addition, we should know if the effect is positive, negative, or possesses an interactive effect with other variables. Finally, we may also want to determine the magnitude of the effect that distinguishes a case and control. Fully understanding a variable’s effect is difficult and thus the problem is broken down into multiple steps. We attempt to address the first part of the problem which is discovering variables that can distinguish a case from a control.

The objective of this research is to discover the variables that can distinguish the case from the control, even if the relationship changes across the
variable space. An alternative to linear based models, tree-based methods possess many advantages that linear models lack. Trees are able to account for non-linear relationships, variable interactions, and categorical variables without a substantial increase in model complexity. The tree algorithms are scale invariant which makes them robust in the presence of outliers. Moreover, trees are able to handle missing values and its computation can be parallelized for fast learning (Srivastava et al., 2002).

In this research, we develop a tree-based ensemble algorithm to accommodate matched data sets for finding both linear and non-linear relationships between cases and controls. The proposed algorithm allows for the relationships between cases and controls to be different in different regions of the input variable space. Moreover, the proposed algorithm is designed to detect variables interactions that are able to distinguish cases and controls.

5.2 Background

5.3 Conditional Logistic Regression

Conditional logistic regression (CLR) is based on logistic regression and is used for matched studies (Hosmer and Lemeshow, 2000). The matched data used in these studies are typically formed by stratification which is the process of dividing members of the population into homogeneous subgroups. For the case of stratified data CLR is a more appropriate analysis than logistic regression because it accounts for the strata in the study within the analysis. CLR is used to determine if the value of the predictor is informative and is able to distinguish which subject is more likely to be the case (Hosmer and Lemeshow, 2000). For the purpose of this study, we only consider the specific
case of 1:1 matching although other kinds of matching such as 1:m and n:m exists. Let \( \mathbf{x} = (x_1, \ldots, x_p) \) be a vector of predictors and \( y \in \{0, 1\} \) be the binary response that denotes the control and case respectively. Denote

\[
E(Y) = P(Y = 1|\mathbf{x}) = P(\mathbf{x})
\] (5.1)

The logistic model is given by

\[
P(\mathbf{x}) = \frac{1}{1 + e^{-\beta'\mathbf{x}}}
\] (5.2)

where \( \beta' = (\beta_1, \ldots, \beta_p) \) are the vector of coefficients of predictors \( \mathbf{x} \). The logit model can be written as

\[
\log \left[ \frac{P(\mathbf{x})}{1 - P(\mathbf{x})} \right] = g(\mathbf{x}, \beta) = \beta'\mathbf{x}
\] (5.3)

In CLR for matched case-control studies, two subjects are paired together to form matched units. For example, in the study by Berg et al. (2010), the observation of a patient with acromegaly and a patient from the general population with similar age and sex are matched to form a stratum.

For a 1:1 matched case, suppose there are \( K \) strata with two subjects, a case and a control observation, in the \( k^{th} \) stratum, where \( k = 1, \ldots, K \). The conditional logistic regression model is

\[
P(x) = \pi_k(x) = \frac{e^{\alpha_k + \beta'\mathbf{x}}}{1 + e^{\alpha_k + \beta'\mathbf{x}}}
\] (5.4)

where \( \alpha_k \) is a stratification parameter with the contribution of all terms constant within the \( k^{th} \) stratum. Given observations \((y_i, \mathbf{x}_i), i = 1, \ldots, n\), the probability of the observed data conditioned on the stratum total and the total number of cases observed is denoted as the conditional likelihood for the \( k^{th} \) stratum. The conditional likelihood for the \( k^{th} \) stratum is (Hosmer and
Lemeshow, 2000).

\[ l_k(\beta) = \frac{e^{\beta' x_{1k}}}{e^{\beta' x_{1k}} + e^{\beta' x_{0k}}} \]  

(5.5)

where \(x_{1k}\) denote the case observation and \(x_{0k}\) denote the control observation in the \(k^{th}\) stratum. The full conditional likelihood over the \(K\) strata of \(l_k(\beta)\) with only \(\beta\) as the only unknown parameter is the product of each \(k^{th}\) conditional likelihood

\[ l(\beta) = \Pi_{k=1}^{K} l_k(\beta) \]  

(5.6)

The maximum likelihood estimators (MLE) of the conditional logistic likelihood function, \(l(\hat{\beta})\), are obtained by finding the \(\beta\) that maximizes the conditional likelihood function in Equation 5.6.

The slope coefficient, \(\beta_j\) for \(j = 1, ..., p\), gives the change in the log-odds for one unit increase in \(x_j\) holding all other \(x\)'s constant. For example, \(\beta_j > 0\) implies as \(x_j\) increases so does the probability of the observation being a case. Like logistic regression, in CLR we test the significance of \(\beta_j\) by conducting a Wald test (Rao, 1973) which compares \(\hat{\beta}_j\) to an estimate of its standard error \(\hat{SE}(\hat{\beta}_1)\). When \(\beta_j\) is significant, there is a linear relationship between case and control in \(x_j\) and thus we can use \(x_j\) to distinguish case and control. As \(\beta_j\) only describes the linear relationship between case and control in \(x_j\), it does not provide any information on whether a non-linear relationship between the case and control exists. Thus, to discover non-linear relationships between the case and control, a more robust model is needed.

5.4 Classification and Regression Tree

Tree based methods are commonly used in data mining. A popular method for tree-based classification and regression is CART (Classification and Regression Tree) (Breiman, 1984). Breiman (1984) showed that regression trees can
be used to predict a continuous outcome, and classification trees can be used to predict a discrete outcome. The goal of tree-based methods is to partition the observations into subregions and then fit a simple model in each one (Hastie et al., 2001). A simple tree structure is a binary-split tree where observations are recursively partitioned by a decision boundary that splits the observations into two separate regions. The model predicts $Y$ by the following expression

$$f(\mathbf{x}) = \sum_{m=1}^{M} c_{m} I(\mathbf{x} \in R_{m})$$

(5.7)

where $R_{m}, m = 1, 2, ..., M$ are the disjoint regions of the predictor space, $c_{m}$ is the majority class in region $R_{m}$, and $I(\cdot)$ is an indicator function that gives a value of 1 if the input condition is true and 0 otherwise. For a binary response dataset where $y \in \{0, 1\}$, the impurity of a region, $Q$, can be modeled with the Gini index as follows

$$Q = Gini(p) = 1 - p^2 - (1 - p)^2$$

(5.8)

where $p$ is the proportion of observations that have $y = 1$. The data is split based on a split value $s$ which is a numeric value if $x_{j}$ is numerical and a category of $x_{j}$ if $x_{j}$ is categorical. We define a pair of subregions as

$$R_{1}(j, s) = \{X|X_{j} \leq s\} \text{ and } R_{2}(j, s) = \{X|X_{j} > s\}$$

(5.9)

The pair of subregions for a categorical variable is defined as

$$R_{1}(j, s) = \{X|X_{j} = s\} \text{ and } R_{2}(j, s) = \{X|X_{j} \neq s\}$$

(5.10)

This kind of split is known as a one versus the rest split and we will only consider this split for this paper and note that there other split methods are possible. Let $Q_{1}(j, s)$ and $Q_{2}(j, s)$ be the impurity for $R_{1}(j, s)$ and $R_{2}(j, s)$
Figure 5.1: Recursive binary splitting (Hastie et al., 2001). The left graph shows the tree that is grown by recursive splitting. The right graph shows the regions that is cased by the splits.

respectively. We seek the splitting variable $j$ and the split value $s$ that solves the following

$$
\min_{j,s} \left( \frac{N_1}{N} Q_1(j, s) + \frac{N_2}{N} Q_2(j, s) \right)
$$

(5.11)

where $N_1, N_2$ are the number of observations in $R_1(j, s), R_2(j, s)$ respectively. The information gain is calculated as follows

$$
IG(j, s) = Q - \left( \frac{N_1}{N} Q_1(j, s) + \frac{N_2}{N} Q_2(j, s) \right)
$$

(5.12)

Let $j^*$ and $s^*$ be the values that solves Equation 5.11, a split is made only if $IG(j^*, s^*) > 0$.

To illustrate how a classification tree works, consider a response $y$ and two predictor variables, $x_1$ and $x_2$. A value of $s_1$ that results in a maximal information gain is found. The space is divided into two subregions and the process is continued. Figure 5.1 (Hastie et al., 2001), shows the resulting decision tree and the partitioned regions of $x_1$ and $x_2$. There are four separate regions that resulted from all the splits $s_1, s_2$ and $s_3$. These regions are also known as terminal nodes as they do not contain any sub regions delineated by the tree.
5.5 Random Forest

As individual tree classifier can be unstable, and overfitted, ensemble methods, consisting of trained classifiers, were introduced for tree classifiers to mitigate those issues (Opitz and Maclin, 1999). Random forest (Breiman, 2001) consists of a collection of tree predictors that each vote on the response of an observation. Let \( t \in T \) be the trees in the random forest, and \( f(x, t) \) be a predictor for tree \( t \), the majority vote from all \( T \) trees are used as the predicted response of the random forest as follows

\[
\arg\max_{\hat{y}} \sum_{t=1}^{T} I(f(x, t) = \hat{y})
\] (5.13)

Each tree in the random forest is constructed by a bootstrapped (Efron and Tibshirani, 1993) sample of the training data set. Let \( P \) be the set of all the variables used for training. For each node, the random forest algorithm only considers a random subspace \( P' \subseteq P \) to use for splitting. The predictor variables \( j \in P' \) are randomly selected for each node and the splitting criteria becomes

\[
\min_{j \in P', s} \left( \frac{N_1}{N} Q_1(j, s) + \frac{N_2}{N} Q_2(j, s) \right)
\] (5.14)

In addition to classifying data, a random forest can also be used for variable selection. Breiman (2001) discusses two ways in which variable selection can be conducted by a random forest: permutation importance indices, and decrease in node impurity measures. The reader can refer to Genuer and Poggi (2010) for a better understanding of both of these measures.

5.6 Motivation

The effect a variable has on the response is not always clear. A variable could have non-linear effects on the response and these effects may not be apparent when examining the data graphically.
Consider the artificial data set shown in Figure 5.2. We define the dark dots as cases and the light dots as controls. The objective is to determine the effect, if any, of $x_1$ or $x_2$. Graphically, if we do not analyze the data as a matched data set, we obtain the raw data plot shown in Figure 5.2(a). From the figure there appears to be no relationship between the light and dark dots. The data set is illustrated in Figure 5.2(b) as matched pairs. From Figure 5.2(b), there does not appear to be a consistent relationship between the case and control as we see that the effects of $x_1$ and $x_2$ change with the region the matched pairs are located. Being able to find the regions where the relationships are consistent is important and this can be illustrated by the regions shown in Figure 5.2(c). In the figure, we can easily visualize the complex relationship that exists between cases and controls and determine that effect $x_1$ and $x_2$ have on the observations.

5.7 Matched Random Forest

To discover non-linear relationships in high dimensional data sets, a robust method that is not restricted to only detecting linear relationships is needed. Random Forests are known to be good at handling data that are non-linear (Hastie et al., 2001). We propose an algorithm, Matched Random Forest (MRF), to account for 1:1 matched data by considering the observations of the data as paired strata instead of separate observations of cases and controls. The resulting strata data set is as follows
Figure 5.2: Illustration of bivariate predictors that distinguishes case and control non-linearly. Figure 5.2(a) shows the data without matching and no apparent relationship can be seen. Figure 5.2(b) shows the data with a matching. Figure 5.2(c) shows the matched data and subregions where the effect of the predictors can be easily identified visually.
As each row represents two points in \( p \)-space, we set the control observations \( \mathbf{x}_0 = (x_{0k1}, \ldots, x_{0kp}) \) as a proxy point for the entire stratum. In other words, we assume the entire strata to be identified and represented by the value \( \mathbf{x}_0 \).

### 5.7.1 Numerical Variables

We first consider numerical input variables and then later extend the method to handle categorical variables. In MRF, we want to find regions of consistent relationships between cases and controls where \( x_{0kj} > x_{1kj} \) or \( x_{0kj} < x_{1kj} \). We call such regions concordant with respect to \( j \). An example of such regions is illustrated in Figure 5.2(c). From the figure, we can determine concordancy projected on either \( j = 1 \) or \( j = 2 \). To find the \( j \) that results in maximal concordancy we need to inspect the concordancy projected on all \( j \)'s. We define

\[
 z_{ji} = \mathcal{I}(x_{1kj} > x_{0kj}) 
\]

as a dependent variable that we use to determine concordancy projected on \( j \). In multi-dimensional problems where \( p > 1 \), we account for the concordancy projected on each \( j \) by using a vector \( \mathbf{z} = (z_1, \ldots, z_p) \) for the dependent variables. The dependent data is constructed as follows

\[
 Z_{K \times p} = \begin{bmatrix} z_{11} & \cdots & z_{1p} \\ z_{21} & \cdots & z_{2p} \\ \vdots & \ddots & \vdots \\ z_{K1} & \cdots & z_{Kp} \end{bmatrix} 
\]

(5.17)
In MRF, the impurity of a region is described by how concordant the region is. Thus, the impurity of a region $R_m$ when we use response $z_j$ can be calculated as

$$Q_m(j) = 1 - \left(\frac{\sum_{k \in R_m} I(z_{kj} = 1)}{N_m}\right)^2 - \left(1 - \frac{\sum_{k \in R_m} I(z_{kj} = 1)}{N_m}\right)^2$$  \hspace{1cm} (5.18)$$

where $N_m$ is the number of strata in region $m$. Equation 5.18 is the Gini index applied to the 0, 1 variable $z_j$. We calculate $Q_m(j)$ for all $j$ which results in a total of $p$ impurity values. In MRF we find a split for partitioning the strata that results in regions $R_1(j, s)$ and $R_2(j, s)$ where

$$R_1(j, s) = \{X_0 | X_{0kj} \leq s\} \text{ and } R_2(j, s) = \{X_0 | X_{0kj} > s\}$$  \hspace{1cm} (5.19)$$

The values of $s$ and $j$ are found by solving the following

$$\min_{j, s} \left(\frac{N_1(j, s)}{N}Q_1^*(j, s) + \frac{N_2(j, s)}{N}Q_2^*(j, s)\right)$$  \hspace{1cm} (5.20)$$

where

$$Q_1^*(j, s) = \min_{j'} Q_1(j')$$  \hspace{1cm} (5.21)$$

and

$$Q_2^*(j, s) = \min_{j'} Q_2(j')$$  \hspace{1cm} (5.22)$$

The value $Q_1^*(j, s)$ is the minimum impurity value over all responses $z_{j'}$ in region $R_1(j, s)$ and the value $Q_2^*(j, s)$ is the minimum impurity value over all responses $z_{j'}$ in region $R_2(j, s)$. The values of $j'$ in region $R_1(j, s)$ and $R_2(j, s)$ can be different as it is based on the $z_{j'}$ that gives the best (minimum) impurity. The number of strata $N_1(j, s)$ and $N_2(j, s)$ depend on the split $j, s$. The split to partition the stratas is based on using the control value and comparing it to $s$. However, this need not be the case as we can use other values as the proxy point for partitioning such as the mean of the observations in the strata.
or the control. Like the regular CART algorithm (Breiman, 1984), this is done recursively until we are not able to find a split that results in an information gain.

5.7.2 Categorical Variables

Categorical variables are handled well in tree-based models as it does not require the use of indicator variables. Let \( x_j \in V_j \) be a categorical variable that have values in the set \( V_j \). The set categorical values for the dependent variable is then \( V_j \times V_j = W_j \). To give an example, if the values of variable \( x_j = \{A, B, C\} \), then if \( x_{0kj} = A \) and \( x_{1kj} = B \) then \( z_j = \{A, B\} \) and this is denoted more simply as \( AB \). It follows that the set of response values, \( W \), consist of all permutation of the values of \( x_{0kj} \) and \( x_{1kj} \). In our previous example, \( W = \{AA, AB, AC, BA, BB, BC, CA, CB, CC\} \). The impurity of the region \( R_m \) using variable \( j \) can then be calculated, for a categorical variable, as follows.

\[
Q_m(j) = 1 - \sum_{w \in W} \left( \frac{\sum_{i \in R_m} I(z_{ji} = w)}{N_m} \right)^2
\]  

(5.23)

Equation 5.10 is used for splitting categorical variables in MRF. Similar to continuous variables, the optimal split can be found by using Equation 5.20.

5.7.3 Variable Importance

We grow this modified random forest and use it to discover the important variables as follows. Let \( t = 1, ..., T \) be the trees that are grown, \( N_{mt} \) be the number of observations in terminal node \( m \) for tree \( t \) and \( Q_{mt}(j) \) be the impurity for region \( m \) in tree \( t \) using the response \( z_j \). Let \( M_t \) be the total number of regions in tree \( t \). After growing the forest, we define a variable
importance measure, $VI(j)$ for each $x_j$ from the forest as follows

$$
VI(x_j) = \frac{\sum_{t=1}^{T} \sum_{m=1}^{M_t} N_{mt} Q_{mt}(j)}{T \sum_{t=1}^{T} \sum_{m=1}^{M_t} N_{mt}}
$$

Equation 5.24 takes the average over the sum of impurities for $j$ in each terminal node for all the trees in the forest. This average describes how concordant a variable is across all the regions. If a variable is good at distinguishing case from control, it’s corresponding response $z_j$ would result in a low $Q(j)$ across multiple regions. The variables with lower $VI(j)$ are deemed to be more important.

### 5.7.4 Probe Variables

To be able make statistical decisions on whether a variable is important, we create probe variables. These probe variables are designed to be unrelated to the response so that their $VI$’s can be compared to that of the $x_j$’s. This use of probe variables is similar to the artificial variables applied in Tuv et al. (2009). Let

$$
\begin{align*}
x_{0j} &= \begin{bmatrix} x_{01j} \\ x_{02j} \\ \vdots \\ x_{0Kj} \end{bmatrix} \\
x_{1j} &= \begin{bmatrix} x_{11j} \\ x_{12j} \\ \vdots \\ x_{1Kj} \end{bmatrix}
\end{align*}
$$

We define $a_j = (x_{02j}, x_{1j}^T)^T$ where $a_j$ has dimensions $2K \times 1$. We then permute $a_j$ as follows

$$
\tilde{a}_j = \Pi a_j
$$

where $\Pi$ is a random permutation operator. We then need to convert $\tilde{a}_j$ to stratus so that we can compare it to the $x_j$’s. We can get the probe variables
from $\tilde{a}_j$ as follows

$$\tilde{a}_j = (\tilde{x}_{0j}, \tilde{x}_{1j}^T)^T \tag{5.27}$$

As this process is done for each $j$, the permutations applied to each $j$ is independent from the other. It is worth noting that permutations do not have to be applied to $a_j$ and can be done directly to the stratum variables $x_{0j}$ and $x_{1j}$ in the following fashion

$$\tilde{x}_{0j} = \Pi x_{0j} \quad \text{and} \quad \tilde{x}_{1j} = \Pi x_{1j} \tag{5.28}$$

In Equation 5.28, $\Pi$ gives a different permutation for the case and control.

MRF is conducted on the augmented data set, $X_{K \times 4p}$ as below

$$\begin{bmatrix}
x_{011} & \cdots & x_{11p} & \tilde{x}_{011} & \cdots & \tilde{x}_{11p} \\
x_{021} & \cdots & x_{12p} & \tilde{x}_{021} & \cdots & \tilde{x}_{12p} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x_{0K1} & \cdots & x_{1Kp} & \tilde{x}_{0K1} & \cdots & \tilde{x}_{1Kp}
\end{bmatrix}_{K \times 4p} \tag{5.29}$$

If $x_j$ does not have any effect that can distinguish cases and controls, we would expect $VI(x_j) = VI(\tilde{x}_{j'})$, $\forall j' \in P$. Conversely, if $x_j$ has an effect, we would expect $VI(x_j) < VI(\tilde{x}_{j'})$, $\forall j' \in P$. We would be able to test that by using the binomial sign test (Sprent, 2000). We use this test to verify if $VI(x_j)$ is statistically lesser than the $VI$ of the other probe variables $\tilde{x}_{j'}$. Let $c$ be the count of $VI(\tilde{x}_{j'})'$ that are less than $VI(x_j)$. The p-value based on a binomial distribution, $B(p + 1, 0.5)$, is calculated as follows

$$P(c) = \sum_{i=0}^{c} \binom{p+1}{i} 0.5^{p+1} \tag{5.30}$$

The p-value is compared to a preset $\alpha$ level. A significant p-value indicates that $x_j$ is important. This test procedure is done for each of the features.
5.8 Experiments

Simulated data were used to test the ability of MRF to discover relevant features. With simulated data, we are able to define the important variables and then test MRF’s ability in discovering them. To test MRFs effectiveness in a real-life application, MRF is tested against financial statement data from the top 500 firms in the United States to try and pick the important variables that are attribute to firms performing well. In the simulated experiments, five test cases were considered to test the efficacy of MRF under different scenarios. CRF was also conducted on these experiments as a benchmark for MRF.

5.8.1 Simulated Dataset

For our simulated data sets, unless otherwise stated, we set the range of the continuous variables $0 \leq x_{ji} \leq 10, \forall i$. To generate the strata, we first generate all the control observations from a uniform distribution over 0 to 10. We use a parameter $\gamma \in (0.5, 1)$ to generate variables with a positive effect as follows

$$x_1 = \begin{cases} 
  x_0 + \delta & \text{with probability } \gamma \\
  x_0 - \delta & \text{with probability } 1 - \gamma
\end{cases} \quad (5.31)$$

Variables with a negative effect are generated as follows

$$x_1 = \begin{cases} 
  x_0 - \delta & \text{with probability } \gamma \\
  x_0 + \delta & \text{with probability } 1 - \gamma
\end{cases} \quad (5.32)$$

We can create variables with no effect by setting $\gamma = 0.5$ in Equation 5.32. Variables that have no effect are essentially noise predictors that not predictive of the response. For variables with positive or negative effects, the higher the value of $\gamma$ the better the variable is at distinguishing cases from controls.
Further details of variables effects are explained in the respective test case sections. For each experiment, we set the number of strata, K, to 500 and we used 200 trees in the forest. These values were picked as a result of computation constraints with the hardware used (Macbook 2Ghz core processor) for simulation. We conducted five replicates for each experiment and averaged the ranks for the replicates. The p-value was then reported from the averaged values.

5.8.2 Case 0: Null Case

We simulated 10 random variables $x_1, ..., x_{10}$ as noise predictor variables that are independent and are not predictive of the response. We conducted this first study to ensure that both MRF and CLR will not detect any variable to be significant. Both MRF and CLR were applied to those variables as well as the probe variables. Figure 5.3 shows the average VI values from the results of the 5 replicates for the noise predictors and probe variables. Figure 5.4 shows the p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. From Figure 5.4, we see that no variables were significant at $\alpha = 0.01$ when both MRF and CLR were used. This is expected as the variables are all noise variables.

5.8.3 Case 1: Linear Effect

We simulated $x_1$ to have a positive effect and $x_2$ to $x_{10}$ to have no effect. Simulations were done with $\gamma = 0.9$ and $\gamma = 0.7$. For this experiment we used 10 probe variables ($x_{11}$ to $x_{20}$). To test the robustness of both MRF and CLR, five replicates of MRF and CLR were conducted on those variables and the average results are shown in Figure 5.5. The standard deviation of the VI for MRF is less than 3% of the VI value for all the variables. Both MRF
Figure 5.3: MRF on the Case 0 noise variables. The graph gives the variable importance (VI) for each variable where the smaller the VI the better the variable is at distinguishing cases and controls. The variables $x_{11}$ to $x_{20}$ are probe variables.

Figure 5.4: p-values of MRF and CLR on the noise variables. In the presence of noise none of the variables are significant at $\alpha = 0.01$ when both MRF and CLR were used.
and CLR detected $x_1$ at the significance level $\alpha = 0.01$. Figure 5.6 shows the p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. As this simulated case only consists of variables that can distinguish case and control linearly, we expect CLR to be able to accurately detect the right variable. As MRF is able to detect $x_1$ as important as well, it shows that MRF is able to detect linear relationships between case and control.

Figure 5.5: VI values from MRF on Case 1 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables.
Figure 5.6: Case 1 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. Both CLR and MRF were able to detect the $x_1$ at $\alpha = 0.01$. 
5.8.4 Case 2: Non-linear Effect

For this experiment we simulated 20 variables $x_1$ to $x_{20}$. In addition, we increased the number of probe variables to 40 ($x_{21}$ to $x_{60}$). We simulated $x_1$ to be able to distinguish case and control and the remaining 19 variables to have no effect. Variable $x_1$ was simulated to have a positive effect when $x_{01} < 5$ and a negative effect otherwise. Simulations were done with $\gamma = 0.9$ and $\gamma = 0.7$. MRF and CLR were conducted on those variables and the average results of five replicates are shown in Figure 5.7. The standard deviation of the VI for MRF is less than 3% of the VI value for all the variables. This gives us confidence that the VI values are stable. Figure 5.8 shows the p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$. MRF is able to detect $x_1$ at $\alpha = 0.01$ whereas CLR fails to detect the variable completely. The most significant variable from CLR is $x_{15}$ which has a p-value of 0.016. If we had increased our $\alpha$-level, we would potentially find this variable as significant which would be a false-positive. The result holds true when $\gamma = 0.9$ and $\gamma = 0.7$. MRF is thus able to detect the relevant variable even when the relationship changes over regions of the strata.

5.8.5 Case 3: Two Variable Non-linear Effect with Interaction

Case 2 dealt with the case where $x_1$ effect was different in two disjoint regions. For case 3, we evaluated the scenario where the effect for $x_1$ is different in three disjoint regions. Variable $x_1$ was simulated to have a positive effect when $x_{01} \leq 2.5$ or when $x_{01} > 7.5$ and a negative effect otherwise. We simulated $x_2$ to have a negative effect when $x_{03} < 5$ and a positive effect otherwise. This creates an interaction where the effect of $x_2$ is dependent on the value
Figure 5.7: VI values from MRF on Case 2 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{21}$ to $x_{60}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables.
Figure 5.8: Case 2 p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$ at $\alpha = 0.01$ while for the same level of significance, CLR was not able to detect $x_1$. 

112
of $x_3$. Variables $x_3$ to $x_{10}$ were simulated to have no effect. We simulated 10 probe variables $x_{11}$ to $x_{20}$. Simulations were done with $\gamma = 0.9$ and $\gamma = 0.7$. MRF and CLR were conducted on those variables and the results of the average of five replicates are shown in Figure 5.9. The standard deviation of the VI for MRF is less than 3% of the VI value for all the variables. The p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$ are shown in Figure 5.10. We find in the results for both $\gamma = 0.9$ and $\gamma = 0.7$ that CRF was unable to detect $x_1$ and $x_2$ to be significant at $\alpha = 0.01$ whereas MRF was able to detect both variables. MRF is thus able to detect the relevant variable even when the relationship between case and control is complex and when there is interaction between features.

5.8.6 Case 4: Three variables with complex interactions

In this case we set three variables to have effect and also included complex interactions in this scenario. A complex interaction is when we have the effect between case and control be dependent on the interaction of two other features. We simulated $x_1$, $x_2$ and $x_3$ to have effects and the remaining 17 variables to be noise. Variable $x_1$ was simulated to have a positive effect when $x_{01} < 5$ and a negative effect otherwise. Variable $x_2$ was simulated to have a positive effect when $x_{03} < 5$ and a negative effect otherwise. Variable $x_3$ was simulated to have a positive effect when $x_{01} < 5$ and $x_{02} \geq 5$ and a negative effect otherwise. Simulations were done with $\gamma = 0.9$ and $\gamma = 0.7$. MRF and CLR were conducted on those variables and the average results are shown in Figure 5.11. The standard deviation of the VI for MRF is less than 3% of the VI value for all the variables. The p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$ are shown in Figure 5.12. Despite the complex interactions, MRF is
Figure 5.9: VI values from MRF on Case 3 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ and $x_2$ are noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.
Figure 5.10: Case 3 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$ and $x_2$ at $\alpha = 0.01$ while CLR was not able to detect both $x_1$ or $x_2$ at $\alpha = 0.01$.
still able to detect $x_1$, $x_2$, and $x_3$ for both values of $\gamma$. CRF on the other hand is only able to detect $x_3$ as significant.

![MRF VI values for $\gamma = 0.9$ and $\gamma = 0.7$](image)

(a) $\gamma = 0.9$

(b) $\gamma = 0.7$

**Figure 5.11**: VI values from MRF on Case 4 variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$, $x_2$, and $x_3$ are noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.
Figure 5.12: Case 4 p-values for both CLR and MRF for the variables $x_1$ to $x_{10}$. The top graph shows p-values for $\gamma = 0.9$ and the bottom graph shows p-values for $\gamma = 0.7$. MRF was able to detect $x_1$, $x_2$, and $x_3$ at $\alpha = 0.01$ while for the same level of significance, CLR was not able to detect $x_1$, $x_2$, and $x_3$. 
<table>
<thead>
<tr>
<th></th>
<th>$x_1 = A$</th>
<th>$x_1 = B$</th>
<th>$x_1 = C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0 = A$</td>
<td>$\gamma$</td>
<td>$\frac{1-\gamma}{2}$</td>
<td>$\frac{1-\gamma}{2}$</td>
</tr>
<tr>
<td>$x_0 = B$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
</tr>
<tr>
<td>$x_0 = C$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{1}{3}$</td>
</tr>
</tbody>
</table>

**Table 5.1:** Probability table for generating categorical variables. The categories of the control observations, $x_0$, are generated with equal probability. Based on the the values of the control, the cases are then generated with the following probabilities shown above.

### 5.8.7 Case 5: Categorical variables

For this experiment, we simulated 10 categorical variables each with three levels. Let $x_j \in \{A, B, C\}$ be a three level categorical variable. We generate each of the three categories of the control observations with equal probability. The case variables were generated based on the probability matrix as shown in Table 5.8.7. From the table, if $\gamma$ is $\frac{1}{3}$ then the value of $x_1$ is independent of the value of $x_0$. For $\gamma$ values greater than $\frac{1}{3}$, $x_0 = A$ is indicative of $x_1 = A$. We set $\gamma$ for $x_1$ to 0.9 and 0.7. The remaining nine variables were simulated with $\gamma = \frac{1}{3}$. MRF and CLR were conducted on the data set and the average results of five replicates are shown in Figure 5.13. The standard deviation of the VI for MRF is less than 3% of the VI value for all the variables. From the graph, we see that MRF is able to detect $x_1$ as a relevant feature at $\alpha = 0.01$ for both $\gamma = 0.9$ and $\gamma = 0.7$ whereas CLR was not able to detect $x_1$ as a relevant feature at $\alpha = 0.01$ when $\gamma = 0.9$ or when $\gamma = 0.7$. The p-values for both CLR and MRF for the variables $x_1$ to $x_{20}$ are shown in Figure 5.14.
Figure 5.13: VI values from MRF on Case 5 categorical variables for $\gamma = 0.9$ and $\gamma = 0.7$. The variables $x_{11}$ to $x_{20}$ are probe variables. The VI for $x_1$ is noticeably smaller than the rest of the variables indicating the MRF is able to detect the relevant variables for this case.
Figure 5.14: Case 5 p-values for both CLR and MRF for the variables \(x_1\) to \(x_{10}\). The top graph shows p-values for \(\gamma = 0.9\) and the bottom graph shows p-values for \(\gamma = 0.7\). MRF was able to detect \(x_1\) at \(\alpha = 0.01\) at both levels of \(\gamma\) while CLR was not able to detect \(x_1\) at both levels of \(\gamma\).
5.8.8 Case Study

From the previous experiments we showed that MRF was able to detect variables that can distinguish case and control when the relationship changes across the variable space. We now apply the method to financial data to demonstrate its robust ability to discover relevant variables that distinguishes the case from the control. Understanding the features that move financial data is important in stock picking. With a plethora of financial information about a firm, investment managers need to screen the right variables for analyzing whether or not a company will outperform or underperform in the following year. It is well known that the relationships between financial variables changes across the variable space (Qi, 1999). For instance, an increase in capital expenditure for a manufacturing firm often yields higher stock prices whereas an increase in capital expenditure for a service firm often results in lower stock prices. MRF can be used to decipher the relevant variables from large set of financial variables. In our study, we took the top 500 US firms by market capitalization in 2010 that had an increase in market capitalization in 2011 as our observations for the study. From these firms we took 10 financial variables from 2010 and 2011 that are known to be important in stock price performance. The variables range from income statement variables, to the balance sheet variables, to financial ratios. The full list and a description of each of the variables are shown in Table 5.2. For each company, we assign the data from 2010 to be the case and the data on 2011 to be the control. We ran MRF and CLR on the data set. Our objective is to find the important variables that contributed to the increase in stock price from 2010 to 2011. We ran both CLR and MRF on this data set and the variable importance of each financial
Table 5.2: Description of features used

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net Income</td>
<td>The excess of revenues over outlays</td>
</tr>
<tr>
<td>Total Asset</td>
<td>Total liabilities + Stockholders equity</td>
</tr>
<tr>
<td>Revenue</td>
<td>Amount of money company receives from core operations</td>
</tr>
<tr>
<td>Cash</td>
<td>Amount of cash a company has</td>
</tr>
<tr>
<td>Cashflow Operations</td>
<td>Cash inflow or outflow as a result of operations</td>
</tr>
<tr>
<td>Cost of Goods Sold</td>
<td>The direct costs attributable to the production of the goods sold by a company</td>
</tr>
<tr>
<td>Long Term Debt</td>
<td>Loans and financial obligations lasting over one year</td>
</tr>
<tr>
<td>Capital Expenditure</td>
<td>Funds used by a company to acquire or upgrade physical assets such as property, industrial buildings or equipment</td>
</tr>
<tr>
<td>Profit Margin</td>
<td>A ratio of profitability calculated as net-income divided by revenues, or net profits divided by sales</td>
</tr>
<tr>
<td>Debt Ratio</td>
<td>A ratio that indicates what proportion of debt a company has relative to its assets.</td>
</tr>
</tbody>
</table>
The variables *Net Income*, *Total Asset*, *Cashflow from operations*, *Cost of Goods Sold*, *Long Term Debt*, and *Capital Expenditure* appear to have a noticeably lower VI values than the probe variables.

A search in the financial literature failed to find evidence that the cash a firm has on its balance sheet is a predictor of stock price. In addition, we know from Dechow *et al.* (1995) that accrual accounting variables are more significant than reported revenues in earnings management, which drive stock prices. MRF findings were consistent with Dechow *et al.* (1995) as the variable *revenue* was not detected as being good at differentiating profitable firms.
Figure 5.16: CLR and MRF on financial variables of top 500 profitable firms by market capitalization. CLR is not able to detect significance of variables at $\alpha = 0.01$. Apart from revenue and cash, MRF detects the other variables as significant at $\alpha = 0.01$.

5.9 Conclusion

In this research, we attempted a novel way of finding relevant variables that can distinguish case from control in matched studies, even if the relationship between case and control changes over the variable space. The current method for discovering relevant variables for match studies, CLR, while effective, is not suited for discovering variables that have non-linear relationships. Non-linearity and interactions are common in many matched data sets and thus a method that is able to discover relevant variables is important. We proposed a Matched Random Forest algorithm (MRF) which is designed to discover relevant variables even if the relationship between case and control are different in different regions of the variable space. We show that MRF is able to detect non-linear relationships as well as interactions between variables whereas CLR is unable to do so.

MRF is able to handle categorical variables and does not suffer from the curse of dimensionality problem that traditional linear models have when categorical variables are coded into indicator variables. MRF was able to pick
up features that are important in a data set of company financials, proving itself to be effective in practical applications. The success of the method opens up the impetus for more research into understanding matched variables when the relationships are non-linear. MRF can be improved to identify the regions where the relationship between case and control is linear. Research can also be done to improve MRF to understand the magnitude of the difference between case and control in the different regions. Overall, MRF provides a new perspective into relationship discovery in matched data sets.
Chapter 6

CONCLUSION

In this dissertation, we showed that the objectives of models can be very different in data sets across various disciplines. When there are non-linear relationships between predictors and response, traditional statistical techniques such as linear regression may not be effective for building a predictive model. In addition, the information we need from the data can be different. For example, in the field of finance, it may be more important to accurately pick out-performing stocks than under-performing stocks. This idea of asymmetry drove the majority of the research in this dissertation.

Chapter 3 contributed to the existing literature by introducing a generalized loss function with parameters to control for asymmetry. We introduced an Asymmetric Support Vector Machine (aSVM) by generalizing the pinball loss function Koenker and Hallock (2001). Until this work, asymmetric loss functions in the literature focused on regression and little research was done on asymmetric loss functions for classification. aSVM was able to produce asymmetric predictions based on a class of primary interest. In addition, ensembles of aSVMs helped increase the precision of the class of interest. Finally, we showed that practical applications of aSVM span from cost sensitive problems to class imbalance problems. This work contributes to the existing SVM literature by introducing a new type of loss function and contributes to the ensemble literature by showing that asymmetric predictions can be achieved by using aSVM.
In Chapter 4, we continued this idea of asymmetry and extended it to variable selection. Variable selection has traditionally focused on searching for variables that are predictive for all values of the response. The research in Chapter 4 brought to light the existence of variables that are more able to predict a particular class. Until this dissertation, there has been no study on these kinds of variables. This research simulated asymmetric variables to demonstrate that such properties in data can exist and showed real data where such variables are found in practice. Chapter 4 introduced the algorithm, Asymmetric Random Forest (ARF), which is able to select asymmetric variables. With ARF, not only were asymmetric variables discovered, but variables that produce asymmetric signals when interacting with other variables were found.

In Chapter 5, we provided a novel way to detect relevant variables that can distinguish case from control in matched studies, even if the relationship between cases and controls changes over the variable space. The current method for discovering relevant variables for match studies, conditional logistic regression (CLR), is not suited for discovering variables that have non-linear relationships. This work adds to the literature by proposing a solution for discovering relevant relationships in non-linear matched data sets. We proposed a Matched Random Forest algorithm (MRF) which is designed to discover relevant variables even if the relationship between case and control are different in different regions of the variable space. We show that MRF is able to detect non-linear relationships as well as interactions between variables whereas CLR is unable to do so.
6.1 Future Work

This dissertation provided groundwork for asymmetric learning and information discovery in non-linear matched data. Each chapter sets the stage for future research to take place. aSVM is just one of the many ways that asymmetric loss can be used for modeling. Some of the other machine learning algorithms that asymmetric loss functions can be adapted to include random forests, gradient boosted trees, and neural networks. As different machine learning techniques have different advantages, further research can be done to show how improved accuracy can be achieved by applying an asymmetric loss function to the training for the other machine learning methods.

As the concept of asymmetric variable selection has not been explored previously, Chapter 4 introduced this concept to the existing machine learning literature. As discussed in Chapter 4, asymmetric variables are ubiquitous and spans over almost all disciplines. Thus much more work is needed in identifying, analyzing, and using such asymmetric variables. Further research should be done in identifying such variables with existing variable selection techniques. Common variable selection techniques such as correlation based feature selection (CFS) (Hall, 1999) can be modified so that they can identify asymmetric variables. The affect of the parameters of ARF should also be further researched to improve the performance of identifying asymmetric variables.

Chapter 5 barely scratched the surface on non-linear matched data sets. The research done paves the way for the better understanding of non-linear matched data. Further research needs to be done to delineate the various regions where the relationship between the cases and controls are linear. This
requires a clustering algorithm, such as hierarchical clustering to group observations together. The regions where cases and controls are linear may be fragmented across the variable space and, thus, research in clustering techniques is necessary to achieve good results. The distance between a case and control should also be future researched. The question of how the distance between cases and controls affect the relevance of the variable is also important as it provides further insight into the variable. Overall, the research of non-linear matched data proves to be an exciting field for future research.
REFERENCES


Breiman, L., Classification and regression trees (Wadsworth International Group, Belmont, California, 1984).


Hao, L. and D. Naiman, Quantile Regression (SAGE Publications, 2007).


