University of Texas at El Paso
ScholarWorks@UTEP

Open Access Theses & Dissertations

2020-01-01

Comparing predictive performance of statistical learning models on Medical data

Francis Biney
University of Texas at El Paso

Follow this and additional works at: https://scholarworks.utep.edu/open_etd

Part of the Applied Mathematics Commons, and the Artificial Intelligence and Robotics Commons

Recommended Citation

Biney, Francis, "Comparing predictive performance of statistical learning models on Medical data" (2020). *Open Access Theses & Dissertations*. 3083.
https://scholarworks.utep.edu/open_etd/3083

This is brought to you for free and open access by ScholarWorks@UTEP. It has been accepted for inclusion in Open Access Theses & Dissertations by an authorized administrator of ScholarWorks@UTEP. For more information, please contact lweber@utep.edu.

COMPARING PREDICTIVE PERFORMANCE OF STATISTICAL LEARNING MODELS ON MEDICAL DATA

FRANCIS BINEY

Master's Program in Computational Science

APPROVED:

Maria Christina Mariani, Ph.D., Chair

Thompson Sarkodie-Gyan, Ph.D.

Suneel Babu Chatla, Ph.D.

Elsa Villa, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

to my

FAMILY

with love

COMPARING PREDICTIVE PERFORMANCE OF STATISTICAL LEARNING MODELS ON MEDICAL DATA

by

FRANCIS BINEY, M.S.

THESIS

Presented to the Faculty of the Graduate School of
The University of Texas at El Paso
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

Computational Science Program
THE UNIVERSITY OF TEXAS AT EL PASO
August 2020

Acknowledgements

I am thankful to God Almighty for making it possible to undertake such a project. It simply wouldn't have been possible without his guidance.

Next, I would like to express my deepest gratitude to my advisor, Dr. Maria C. Mariani from the Mathematical Sciences Department at The University of Texas at El Paso, for her advice, encouragement and support.

I wish to thank the other members of my committee, Dr. T. Sarkodie-Gyan from Electrical and Computer Engineering Department, Dr. Suneel Babu Chatla from the Mathematical Sciences Department and Dr. Elsa Y. Villa from the Department of Education. Their comments and guidance helped in the completion of this work.

I also like to thank Dr. Ming-Ying Leung and Dr. Sharma, Natasha from the Mathematical Sciences Department at The University of Texas at El Paso for stimulating my knowledge. I sincerely thank all the other professors and staff of the Mathematical Sciences Department at The University of Texas at El Paso for all their support.

And finally, to my loving wife, Mrs. Utitofon P. Biney and my kids Lia Yaa Biney, Nnenna Paula Okoh and Uyai Francis Biney for sparing me family time to work on this project. I would like to say God bless you for all the wonderful things you have done for me.

NOTE: This thesis was submitted to my Supervising Committee on the July 30, 2020.

Abstract

This work investigates the predictive performance of 10 Machine learning models on three medical data including Breast cancer, Heart disease and Prostate cancer. Furthermore we use the models to identify risk factors that contribute significantly to these diseases.

The models considered include; Logistic regression with L_1 and L_2 penalties, Principal component logistic regression(PCR-LR), Partial least squares logistic regression(PLS-LR), Multivariate adaptive regression splines(MARS), Support vector machine with Radial Basis Kernel (SVM-RBK), Random Forest(RF), Gradient Boosting Machines(GBM), Elastic Net (Enet) and Feedforward Neural Network(FFNN). The models were grouped according to their similarities and learning style; i) Linear regularized models: LR-Lasso, LR-Ridge and LR-Enet. ii) Linear dimension reduction models: PCR and PLSR. iii) Non-Linear ensemble models : Random forest and Gradient Boosting. iv) Other Non-Linear models: FFNN, SVM and MARS. The methodology is not new, however the major contribution of this work comes in the realm of applications. The methodology was applied to three medical data set: Breast cancer, Heart disease and Prostate cancer. In all the applications the methodology provides insight into predictive performance of these model and the risk factors of these diseases.

The model selection and hyperparameter tuning were done using bias-variance tradeoff and cross-validation. The model's performance and generalization were improved for each method, by applying early stopping, dropout and removed non-significant variables to avoid overfitting. Different predictive performance measures were used including prediction accuracy, sensitivity and specificity depending of the nature of the response distribution whether balanced or imbalanced to compared the models.

The result show that the non-linear models; SVM-RBK, RF and FFNN gave the best predictive performance for the breast cancer data. The linear models; LR-PLS, LR-PC ,LR-Ridge and LR-Enet were preferred for heart disease and mixture of linear and non-linear models; LR-Lasso, LR-Enet, RF and GBM best describes the prostate cancer predictions.

Table of Contents

Page

Acknowledgements iv

Abstract v

Table of Contents vii

List of Tables xi

List of Figures xii

Chapter

1 Introduction 1

- 1.0.1 Problem Statement 3
- 1.0.2 Aims and Objectives 3
- 1.0.3 Thesis Overview 3

2 Statistical Learning Models 5

- 2.1 Logistic Regression Model 5
  - 2.1.1 Defining the Logistic Regression Model 5
  - 2.1.2 Interpreting the Logistic Regression Model 6
  - 2.1.3 Fitting the Logistic Regression Model 6
- 2.2 Regularization 8
  - 2.2.1 L_1 Regularization 9
  - 2.2.2 L_2 Regularization 9
  - 2.2.3 Logistic Regression with L_1 Regularization 10
  - 2.2.4 Logistic Regression with L_2 Regularization 11
- 2.3 Principal Component Analysis 11
  - 2.3.1 Explaining Variations in Data 12
- 2.4 Principal Component Regression 13
- 2.5 Random Forests 14

2.6 Multivariate Adaptive Regression Splines (MARS) 17
2.7 Support Vector Machine (SVM) 20
2.8 Gradient boosting 30
2.8.1 Algorithm for Gradient Boosting 32
2.9 Neural networks 33
2.9.1 Model definition and description 33
2.9.2 Model fitting 35
2.9.3 Issues in Training Nueral networks 36
3 Model Assessment and Selection 38
3.1 Model Assessment 38
3.1.1 Loss Function 38
3.2 Test Error 39
3.3 Estimating Test Error 39
3.3.1 Train Error 39
3.4 Adjusted Train Error 41
3.4.1 Mellow's C_p 42
3.4.2 Akaike information criterion 42
3.4.3 Bayesian information criterion (BIC) 43
3.4.4 Adjusted R^2 44
3.5 Cross Validation 44
3.6 Model Selection 46
3.7 Bias, Variance and Model Complexity 46
3.7.1 The Bias–Variance Tradeoff 46
3.7.2 illustration of Overfitting and Underfitting 47
3.7.3 Empirical illustration of Bias-Variance Tradeoff 48
3.7.4 The Bias-Variance Decomposition 49
3.8 Model Selection Methods 49
3.8.1 Cross-validation for model selection 49

3.8.2 Subset Selection 51
3.8.3 Stepwise selection 52
3.8.4 Regularization or Shrinkage Methods 54
3.8.5 Dimension Reduction Methods 59
4 Data Analysis and Result. 62
4.1 Analysis on Breast Cancer Data 62
4.1.1 Data background 62
4.1.2 Association of Data 63
4.1.3 Principal Component Analysis (PCA) 65
4.2 Application of Statistical learning methods 69
4.2.1 Logistic regression (LR): 70
4.2.2 Logistic Principal Component Regression(LR-PC): 75
4.2.3 Logistic Partial Least squares Regression(LR-PLS): 77
4.2.4 Multivariate Adaptive Regression Splines(MARS): 78
4.2.5 Random Forest(RF): 79
4.2.6 Gradient Boosting 81
4.2.7 Support Vector Machine 82
4.2.8 Deep Learning: Feed froward Network (FFNN) 84
4.2.9 Models Summary for Breast Cancer Data 85
4.3 Analysis on Heart Disease Data 86
4.3.1 Association of Data 87
4.3.2 Principal Component Analysis (PCA): 89
4.4 Application of Statistical learning methods 92
4.4.1 Logistic regression 92
4.4.2 Logistic Principal Component Regression(LR-PC): 94
4.4.3 Logistic Partial Least squares Regression(LR-PLS) 94
4.4.4 Multivariate Adaptive Regression Splines: MARS 95
4.4.5 Random Forest(RF): 96

4.4.6 Gradient Boosting(GBM): 98
4.4.7 Support Vector Machine(SVM): 99
4.4.8 Deep Learning: Feed froward Network(FFNN): 100
4.4.9 Models Summary 102
4.5 Analysis on Prostate Cancer Data 103
4.5.1 Association of Data 105
4.5.2 Principal Component Analysis (PCA): 107
4.6 Application of Statistical learning methods 109
4.6.1 Logistic regression(LR): 109
4.6.2 Logistic Principal Component Regression(LR-PC): 111
4.6.3 Logistic Partial Least squares Regression(LR-PLS) 111
4.6.4 Multivariate Adaptive Regression Splines(MARS): 112
4.6.5 Random Forest(RF): 114
4.6.6 Gradient Boosting 115
4.6.7 Support Vector Machine(SVM): 116
4.6.8 Deep Learning: Feed froward Network(FFNN): 117
4.6.9 Models Summary 118
4.6.10 ELASTIC NET for Breast Cancer Data 120
4.6.11 ELASTIC NET for Heart Disease Data 121
4.6.12 ELASTIC NET for Prostate Cancer Data 122
5 Concluding Remarks 123
References 126
Curriculum Vitae 131

List of Tables

4.1 Name of variables and description 62

4.2 % of variation explained by each Component 65

4.3 Cofficient,Pred. Accuracy of Lasso and Ridge LR model 73

4.4 Coefficient of Principal Components 76

4.5 Selecting the optimal model for FFNN using miss classification error 84

4.6 All model applied to the Breast Cancer Data 85

4.7 Features name and short Description 86

4.8 % variation explained by each component 89

4.9 Estimate Coefficient of Lasso and Ridge LR 93

4.10 LR-PC coefficient 94

4.11 Selecting the optimal model for FFNN using miss classification error 101

4.12 All model applied to the Heart Disease data 102

4.13 Variables name and brief description. 103

4.14 % variation explained by each components 107

4.15 Estimate Coefficient of LR-Lasso and Ridge 110

4.16 LR-PC coefficient 112

4.17 Selecting the optimal model for FFNN using miss classification error 117

4.18 All model applied to the Prostate Cancer data 118

List of Figures

2.1 Illustration of PCA in the 2-D scenario. 12
2.2 The Flow Chart of Random Forests. 16
2.3 The basis functions $(x - t)+$ (solid orange) and $(t - x)+$ (broken blue used by MARS. 18
2.4 The linear algebra of a hyperplane:Hyperplane in 2 Dimensions.[17] 21
2.5 Maximal Margin Classifier.[17] 22
2.6 Maximal Margin Classifier.[1] 23
2.7 Illustration of Soft-Margin SVM.[1] 25
2.8 Diagram of a single hidden layer, feed-forward neural network [27]. 33
2.9 Common Activation Functions [27]. 34

3.1 Expected training error vs expected test error [1] 40
3.2 5 fold cross validation partition. 45
3.3 Train, validation and Test set. 46
3.4 Problem of Overfitting and Underfitting Train set [27] 47
3.5 Prediction error (orange) and tenfold cross-validation curve(blue) estimated from a single training set. 50
3.6 Simulated data: $n = 50$, $p = 45$, all having nonzero coefficients.The bias increases as λ increases and variance decreases as λ increases 55
3.7 Lasso (left) and Ridge regression (right) contours of the error and constraint functions. 57

4.1 Scatter plot and Correlation for Breast Cancer Variables 63
4.2 Scatter plot and Correlation between Variables by response 64
4.3 % of variation explained by each Component 66

4.4 Correlation between variables and PCA. 67
4.5 Contributions of variables to PCA 67
4.6 Breast cancer variables PC 68
4.7 Breast Cancer PCA Bipot 69
4.8 Relation between, Coeff.size, number of features and $log(\lambda)$ 71
4.9 Relation b/n, Coeff.size, number of features and % Dev Explained 71
4.10 selecting λ using 10 fold cross validation 72
4.11 selecting λ using 10 fold cross validation 73
4.12 Best λ by 10 fold cross validation and coefficient of LR ridge 74
4.13 Best λ by 10 fold cross validation and coefficient of LR ridge 74
4.14 Variables importance plot for Lasso and Ridge regression 75
4.15 selecting λ and number of components using 10 fold cross validation 76
4.16 Selecting the number of component by CV 77
4.17 Selecting the number of component by CV 78
4.18 MARS Variables important plot 79
4.19 Selecting the split size 79
4.20 Selecting the Tree size 80
4.21 RF important variables in Predicting Cancer 80
4.22 Selecting the Tree size 81
4.23 GBM important Variable 82
4.24 Selecting the Cost parameter 83
4.25 SVM important variables 83
4.26 FFNN important variables 84
4.27 Scatter plot and Correlation for Heart Disease variables 87
4.28 Scatter plot and Correlation between Variables by response 88
4.29 All Variables-PC. 90
4.30 Variables-PCA correlation. 90
4.31 Heart disease PCA Biplot 91

4.32 Selecting λ using 10 fold cross validation 92
4.33 Important variable plot using Lasso and Ridge regression 93
4.34 Selecting the number of components by CV (left) and variables importance plot (right). 95
4.35 Selecting the number of component by CV 95
4.36 MARS variables importance plot 96
4.37 Selecting the split size 97
4.38 Selecting the Tree size 97
4.39 RF variables importance plot 98
4.40 Selecting the Tree size 98
4.41 GBM important Variable 99
4.42 Selecting the Cost parameter 100
4.43 SVM variables importance plot 100
4.44 NN important variables 101
4.45 An illustration of the Gleason grading process.[29] 104
4.46 Scatter plot and Correlation for Prostate Cancer variables 105
4.47 Scatter plot and Correlation between Variables by response. 106
4.48 Correlation between variables and PCA. 108
4.49 Prostate Cancer all Variances PC 108
4.50 Prostate Cancer PCA Biplot 109
4.51 Selecting λ using 10 fold cross validation 110
4.52 Lasso and Ridge regression variable important plot 111
4.53 Selecting the number of component by CV(left) and variables importance plot (right). 112
4.54 Selecting the number of component by CV 113
4.55 MARS variable importance plot 113
4.56 Selecting the tree size 114
4.57 RF variable importance plot 114

4.58 Selecting the Tree size 115
4.59 GBM variable importance plot 115
4.60 Selecting the Cost parameter 116
4.61 SVM variable important plot 117
4.62 FFN variable importance plot 118
4.63 Selecting the optimal hyper parameters by CV 120
4.64 LR-Enet variable importance plot 120
4.65 Selecting the optimal hyper parameters by CV 121
4.66 LR-Enet variable importance plot 121
4.67 Selecting the optimal hyper parameters by CV 122
4.68 LR-Enet variable importance plot 122

Chapter 1

Introduction

Statistical learning models have been applied to detect and classify many chronic diseases such as cancer, heart disease, tumor, diabetes, and several others (see [43],[45],[46],[30]). These methods aid practitioners and researchers to learn past given observations to detect trends and patterns in independent data set. Also, some of these models helps to identify variables that plays significant role in the cause of these diseases.

Prostate cancer (PC) is the second most common cancer among males worldwide that results in more than 350,000 deaths annually [29]. With more than 1 million new diagnoses reported every year, the key to decreasing mortality is developing more precise diagnostics. Diagnosis of PC is based on the grading of prostate tissue biopsies. These tissue samples are examined by a pathologist and scored according to the Gleason grading system [29]. Stamey et al.(1989) examined the correlation between the level of prostate specific antigen (PSA) and a number of clinical measures, in 97 men who were about to receive a radical prostatectomy. Using Bivariate and multivariate analyses they showed cancer volume to be the primary determinant of serum prostate specific antigen levels[30]. Hastie et al. applied different predictive models including PCR, PLS, Lasso, Ridge and best subset regression to assess risk factor of prostate cancer. They concluded that volume of prostate cancer to be a significant predictor prostate cancer[1].

The mortality rate of breast cancer (BC) has been increasing in the past decades [40]. In the United States, the risk of a woman having breast cancer at some point in her life is about 12%. It has been shown that, one in eight women has the risk of developing breast cancer in her life time [40]. Almost (80%) breast cancers are invasive and they break through the walls of the glands or ducts where they appear and then spread into surrounding breast tissue. However, the mortality rate has declined over the past years with the application of Statistical learning methods. A recent analysis shows that the survival rate is 91% after 5 years of diagnosis and 80% after 15 years of diagnosis [40]. Vikas et.al compares several supervised learning classifiers, such as Naive Bayes, support Vector Machine with Radial Basis Function (SVM-RBF) kernel, Radial Basis Function neural networks, and Decision trees to find the best classifier in breast cancer datasets [41]. Their studies show that SVM-RBF kernel is more accurate (96.84%) than the other classifiers. Joachims et al. used neuron-fuzzy model to achieve prediction accuracy of 95.06% [42]. Other researchers have worked on several data mining algorithms such as SVM, k-nearest neighbor algorithm (IBK), and Bloom Filter (BF) Tree on breast cancer data [43].

Heart disease has been concern among medical researchers for many years. A recent analysis has shown that every year about 735,000 Americans have heart attacks. Of these, 525,000 are first heart-attack victims and 210,000 have already had heart attacks [44]. One of the major challenges in heart disease is its correct detection in body human. Since a lot of parameters and technicality are involved for accurately predicting this disease, there is a vast scope of research including Statistical learning models. Dwivedi et al. used logistic regression, artificial neural network, and support vector machine techniques on heart disease data and concluded logistic regression as the best model with prediction accuracy of 85% [45]. Artificial neural network and fuzzy neural network used by Kahramanli et al. on Cleveland heart disease data and obtained 86.8% prediction accuracy [46]. Thus these models play a vital role in the diagnosis and prognosis of heart disease in patients.

In this work we investigated the predictive performance of 10 Machine learning models on three medical data including Breast cancer, Heart disease and Prostate cancer. We also use the models to identify risk factors that contribute significantly to these diseases. The models considered include: Logistic regression with L_1 and L_2 penalties, Principal component regression, Partial least squares regression, Multivariate adaptive regression splines, Support vector machine, Random Forest, Gradient Boosting Machines, Elastic Net and Feedforward Neural Network.

1.0.1 Problem Statement

With the importance of understanding and diagnosing disease, Statistical learning models have played a vital role, but due to the availability of many such models it sometimes difficult to find the right model for a given situation.

In this work, we investigated the predictive performance of 10 Machine learning models on three medical data including Breast cancer, Heart disease and Prostate cancer. We also use the models to identify risk factors that contribute significantly to these diseases. The models considered ranged from simple logistic regression model to complex Feedforward Neural Network.

1.0.2 Aims and Objectives

This thesis aims to compare the predictive performance of different statistical learning models on medical data to identify which models best fit the trends and patterns in these data. We also use the models to identify important risk factors of these diseases. The methodology is not new however the major contribution of this work comes in the realm of applications. The methodology is applied to three data set: Breast cancer, Heart disease and Prostate cancer.

1.0.3 Thesis Overview

The thesis is organized as follows:

1. Chapter 1 gives a preview of the thesis topic under considerations and Statistical Learning methods. This chapter also review the literature on statistical learning applications on Breast cancer, heart disease and Prostate cancer.

2. Chapter 2 looks at definition and different classes of statistical learning models.

3. Chapter 3 discusses ways of assessing and selecting a model for a given data.

4. Chapter 4 is on data analysis and results.
5. Chapter 5 on conclusion and recommendation.

Chapter 2

Statistical Learning Models

2.1 Logistic Regression Model

2.1.1 Defining the Logistic Regression Model

Logistic Regression is one of the most commonly used classification algorithm for predicting the probability of the class of a target variable. Consider data that contain n observations $\{(y_i, \mathbf{x}_i) : i = 1, \ldots, n\}$, where y_i is the binary $0, 1$ response for the i -th individual and $\mathbf{x}_i$ is its associated feature vector [1]. The response y_i follows a bernoulli trial with parameter $p_i = E(y_i) = \mathrm{P}\{y_i = 1\}$.

The probability distribution function of y_i can be written in the exponential form

$$\begin{aligned} f_Y(y_i) &= p_i^{y_i}(1-p_i)^{1-y_i} = (1-p_i)\left\{\frac{p_i}{1-p_i}\right\}^{y_i} \\ &= \exp\left\{y_i \log\frac{p_i}{1-p_i} + \log(1-p_i)\right\} \\ &= \exp\left\{\frac{y_i \log\{p_i/(1-p_i)\} - \{-\log(1-p_i)\}}{1} + 0\right\} \end{aligned}$$

If we let $\theta_i = \log\{p_i/(1-p_i)\}$, $a(\phi) = 1, c(y_i, \phi) = 0$, and

$$b(\theta_i) = -\log(1-p_i) = \log\left(\frac{1}{1-p_i}\right) = \log\left(1 + \frac{p_i}{1-p_i}\right) = \log\left(1 + e^{\theta_i}\right)$$

It follows that $E(y_i|\mathbf{x}_i) = p_i$ and $\operatorname{var}(y_i|\mathbf{x}_i) = p_i(1-p_i)$. The logistic regression model applies the canonical link $\theta_i = \eta_i$, which leads to the following formulation:

$$\operatorname{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \mathbf{x}_i^T \boldsymbol{\beta} \tag{2.1}$$

or, equivalently [1],

$$p_i = \text{logistic}\left(\mathbf{x}_i^T \boldsymbol{\beta}\right) = \frac{\exp\left(\mathbf{x}_i^T \boldsymbol{\beta}\right)}{1 + \exp\left(\mathbf{x}_i^T \boldsymbol{\beta}\right)} \tag{2.2}$$

2.1.2 Interpreting the Logistic Regression Model

The logistic regression coefficients $\boldsymbol{\beta}$ is interpreted similarly as in linear regression and has to do with the odds ratio [1]. The quantity

$$\frac{p_i}{1-p_i} = \frac{\mathrm{P}\left(Y=1|\mathbf{x}_i\right)}{\mathrm{P}\left(Y=0|\mathbf{x}_i\right)}$$

is often referred to as the odds of having $Y = 1$ conditioning on $\mathbf{x}_i$, which is a risk measure in many applications. The logistic model can be expressed in terms of odds

$$\log(\text{ odds }) = \mathbf{x}_i^T \boldsymbol{\beta} \tag{2.3}$$

With similar arguments in linear regression, model (2.3) implies that every one-unit increase in X_j, while holding other features fixed, would lead to an amount of β_j change in the logarithm of the odds [1]. That is

$$\beta_j = \log\left(\text{Odds}\, x_{j-x+1}\right) - \log\left(\text{Odds}\, x_{j-x}\right) = \log\left(OR_{(x+1)=x}\right)$$

In other words, the odds ratio comparing $X_j = x + 1$ vs. $X_j = x$, with other predictor fixed, is $OR_{(x+1)x} = \exp\left(\beta_j\right)$

2.1.3 Fitting the Logistic Regression Model

The log likelihood for the logistic model (2.1) can be written as:

$$\begin{aligned} L(\boldsymbol{\beta}) &= \sum_{i=1}^{n}\left\{y_i \log \frac{p_i}{1-p_i} + \log\left(1-p_i\right)\right\} \\ &= \sum_{i=1}^{n}\left\{y_i \beta^T x_i - \log\left(1 + e^{\beta^T x_i}\right)\right\} \end{aligned} \tag{2.4}$$

Here $\beta=\{\beta_0,\beta_1\}$, and we assume that the vector of inputs x_i includes the constant term 1 to accommodate the intercept[1].

To maximize the log-likelihood, we set its derivatives to zero. These score equations are

$$\frac{\partial L(\beta)}{\partial \beta}=\sum_{i=1}^{n} x_i\left(y_i-p_i\right)=0 \tag{2.5}$$

which are $p+1$ equations nonlinear in β[1].

To solve the score equations(2.5), we use the newton-Raphson algorithm, which requires the second-derivative or Hessian matrix

$$\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^T}=-\sum_{i=1}^{n} x_i x_i^T p_i\left(1-p_i\right). \tag{2.6}$$

Starting with β^{old} , a single newton update is

$$\beta^{\text{new}}=\beta^{\text{old}}-\left(\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^T}\right)^{-1} \frac{\partial L(\beta)}{\partial \beta} \tag{2.7}$$

where the derivatives are evaluated at β^{old} . It is convenient to write the score and Hessian in matrix notation. Let y denote the vector of y_i values, $\mathbf{X}$ the $n \times (p+1)$ matrix of x_i values, p the vector of fitted probabilities with i th element $p_i\left(\beta^{\text{old}}\right)$ and $\mathbf{W}$ a $n \times n$ diagonal matrix of weights with i th diagonal element $p_i\left(\beta^{\text{old}}\right)\left(1- p_i\left(\beta^{\text{old}}\right)\right)$ [1]. Then we have

$$\frac{\partial L(\beta)}{\partial \beta}=\mathbf{X}^T(\mathbf{y}-\mathbf{p}) \tag{2.8}$$

$$\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^T}=-\mathbf{X}^T \mathbf{W} \mathbf{X}. \tag{2.9}$$

The Newton step is thus

$$\begin{aligned}
\beta^{\text{new}} &= \beta^{\text{old}}+\left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^T(\mathbf{y}-\mathbf{p}) \\
&= \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{W}\left(\mathbf{X} \beta^{\text{old}}+\mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})\right) \\
&= \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{W} \mathbf{z}
\end{aligned} \tag{2.10}$$

In the second and third line we have re-expressed the Newton step as a weighted least squares step, with the response

$$\mathbf{z} = \mathbf{X}\beta^{\text{old}} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p}) \tag{2.11}$$

sometimes known as the *adjusted response*[1]. These equations get solved repeatedly, since at each iteration $\mathbf{p}$ changes, and hence so does $\mathbf{W}$ and $\mathbf{z}$. This algorithm is referred to as *iteratively reweighted least squares* or IRLS, since each iteration solves the weighted least squares problem [1]:

$$\beta^{\text{new}} \leftarrow \arg\min_{\beta}(\mathbf{z} - \mathbf{X}\beta)^T\mathbf{W}(\mathbf{z} - \mathbf{X}\beta) \tag{2.12}$$

In practice $\beta = 0$ is a good starting value for the iterative procedure, although convergence is never guaranteed. For the multiclass case ($K \geq 3$) the Newton algorithm can also be expressed as an iteratively reweighted least squares algorithm, but with a vector of $K - 1$ responses and a nondiagonal weight matrix per observation[1]. Logistic regression models are used mostly as a data analysis and inference tool, where the goal is to understand the role of the input variables.

2.2 Regularization

To obtain a parsimonious model and important features from the original model, two regularization methods was used L_1 and L_2. The regularization technique penalizes the magnitude of coefficients of features and possibly lower prediction error and can sometimes improve prediction accuracy. By retaining a subset of the features and discarding the rest, regularization produces a model that is interpretable. Common shrinkage methods optimize the least squares criterion while shrinking the size or length of the regression coefficients. One motivation for this approach is that $E\|\hat{\boldsymbol{\beta}}\|^2 \geq \|\boldsymbol{\beta}\|^2$ despite LSE $\hat{\boldsymbol{\beta}}$ is unbiased for $\boldsymbol{\beta}$[1].

2.2.1 L_1 Regularization

In general, a regularized or penalized estimator $\tilde{\boldsymbol{\beta}}$ in linear regression can be stated as follows

$$\tilde{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} g\left(|\beta_j|\right) \leq t \tag{2.13}$$

where n is the of observation and p the number of features for some convex function $g(\cdot)$ and constant t. This is called the Lasso penalty. The intercept term β_0 can be suppressed by working with centered data. This is the BRIDGE regression introduced by Frank and Friedman (1993)[2]. The Lagrangian form of (2.13) is:

$$\tilde{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} g\left(|\beta_j|\right) \right\} \tag{2.14}$$

where $\lambda > 0$ is a penalty or regularization parameter that controls the amount of the shrinkage. The functions $g(x) = x^q$ with $q \geq 0$ is most often used. $\tilde{\beta}$ corresponds to LSE when $q = 0$, lasso (Tibshirani, 1996)[3] estimator when $q = 1$; and the ridge estimator (Hoerl and Kennard,1970)[4] when $q = 2$.

The lasso estimator $\hat{\beta}_{L_1}$ does not have an explicit form but its entire solution path for any λ can be efficiently obtained via the LARS (Efron et al., 2004)[6] algorithm. The generalized cross-validation criterion is often used to determine the optimal λ in both ridge regression and lasso [Craven and Wahba (1979)][7].

2.2.2 L_2 Regularization

Ridge regression uses L_2 regularization to shrinks the regression coefficients by imposing a penalty on their size. The ridge coefficients minimize a penalized residual sum of squares:

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\text{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{2.15}$$

where the complexity parameter $\lambda \geq 0$ controls the amount of shrinkage: the larger the value of λ, the greater the amount of shrinkage[1]. The coefficients are shrunk toward zero. An equivalent way to write the ridge problem is

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \tag{2.16}$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 \leq t$$

which makes explicit the size constraint on the parameters [1]. There is a one-to-one correspondence between the parameters λ in (2.15) and t in (2.16). The ridge solutions are not equivalent under scaling of the inputs, and so the inputs are normally standardizes before solving (2.15). The matrix form of criterion in (2.15) [1] is:

$$\text{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta \tag{2.17}$$

with solution given by:

$$\hat{\beta}^{\text{ridge}} = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{2.18}$$

where I is the $p \times p$ identity matrix.

2.2.3 Logistic Regression with L_1 Regularization

The L_1 penalty used in the lasso (2.13) can be used for variable selection and shrinkage with any linear regression model. For logistic regression, we would maximize a penalized version of (2.4):

$$\hat{\beta}^{\text{lasso}} = \max_{\beta} \left\{ \sum_{i=1}^{N} \left[y_i \left(\beta^T x_i\right) - \log\left(1 + e^{\beta^T x_i}\right) \right] - \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{2.19}$$

As with the lasso, we typically do not penalize the intercept term, and standardize the features for the penalty to be meaningful. The solution is found using nonlinear programming methods (Koh et al., 2007)[8]. Alternatively, using the same quadratic approximations that were used in the Newton algorithm in (2.12), we can solve (2.19) by repeated application of a weighted lasso algorithm. The score equations (2.8) for the variables with non-zero coefficients have the form

$$\mathbf{x}_j^T(\mathbf{y}-\mathbf{p}) = \lambda \cdot \operatorname{sign}(\beta_j) \tag{2.20}$$

Path algorithms for lasso are more difficult, because the coefficient profiles are piecewise smooth rather than linear. The solution is obtained by using quadratic approximations [1].

2.2.4 Logistic Regression with L_2 Regularization

The ridge-regression with L_2 penalty technique can be used to overcomes the multicollinearity problem. Here we maximized the L_2 penalized versions (2.4) as follows:

$$\hat{\beta}^{\text{ridge}} = \max_{\beta}\left\{\sum_{i=1}^{N}\left[y_i\left(\beta^T x_i\right) - \log\left(1+e^{\beta^T x_i}\right)\right] - \lambda\sum_{j=1}^{p}\beta_j^2\right\} \tag{2.21}$$

where λ controls the amount of regularization [1].

From the perspective of predictive modeling, variable selection is not necessary with regularization, except for the determination of the penalty parameter λ. In applications where the effects of features are under study, inclusion of irrelevant variables complicate model interpretability, although their effects are shrunk. Both ridge regression and lasso usually provide competitive predictive performance. The difference between the ridge and the lasso estimators is that as λ increases, Lasso effectively does variable selection by setting some coefficients to be exactly zero [1].

2.3 Principal Component Analysis

PCA can be motivated and understood in several different yet related ways: 1) The first is to (linearly) transform correlated variables into a set of uncorrelated ones. Having uncorrelated predictors is useful in dealing with multicollinearity in regression. 2) The second is to seek mutually orthogonal directions (linear combinations) along which data show most variation.

2.3.1 Explaining Variations in Data

Why do we want to seek mutually orthogonal directions along which data show most variation? This question was explained in Gentle (2009 [9]. First of all, the information in data is presented as variation. That is why many statistical methods such as analysis of variance (ANOVA) are design to analyze the variation. Secondly, correlation among variables reduces the amount of information that the variables contain. Thus, PCA is aimed to seek transforms of variables that are uncorrelated to each other and at the same time explain the maximum variation in the original variables.

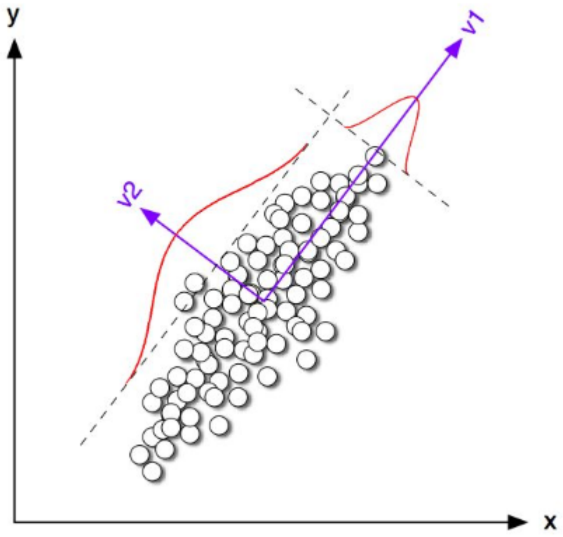


Figure 2.1: Illustration of PCA in the 2-D scenario.

Note that the straight line that the first PC form corresponds to the total LS line that minimized the total perpendicular distances from each point to the straight line, which can be contrasted with the LS fitted line.

2.4 Principal Component Regression

Principal Component (PC) regression, starts by using the Principal Components of the feature variables $\mathbf{Z}$ in place of the original variables $\mathbf{X}$. An important feature of principal components is that they reduce the dimension of data set to linear combination of the feature variable that explain most of the variability of the original data [Jolliffe,2002][10]. The regression equation can be expressed as:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.22}$$

where $\mathbf{y}$ is the response variable, a vector of n standardized observations, $\mathbf{X}$ is an $(n \times p)$ matrix of standardized features variables, $\boldsymbol{\beta}$ is a vector of p regression coefficients to be estimated and $\boldsymbol{\epsilon}$ is a vector of error terms that are independent of each other with the constant variance σ^2 [10]. The values of the PCs for each observation are given by

$$\mathbf{Z} = \mathbf{XA} \tag{2.23}$$

where the $(i, k)th$ element of $\mathbf{Z}$ is the score-value of the kth PC for the ith observation, and $\mathbf{A}$ is a $(p \times p)$ matrix whose kth column is the kth eigenvector of $\mathbf{X}'\mathbf{X}$. Because A is orthogonal, $\mathbf{X}\boldsymbol{\beta}$ can be rewritten as[10]:

$$\mathbf{X}\boldsymbol{\beta} = \mathrm{XAA}'\beta = \mathrm{Z}\gamma, \tag{2.24}$$

where $\gamma = \mathbf{A}'\beta$. Equation (2.22) can therefore be written as

$$\mathbf{y} = \mathbf{Z}\gamma + \boldsymbol{\epsilon} \tag{2.25}$$

which has simply replaced the predictor variables by their PCs in the regression model. Principal component regression can be defined as the use of the model (2.25) or of the reduced model

$$\mathbf{y} = \mathbf{Z}_m\gamma_m + \boldsymbol{\epsilon}_m \tag{2.26}$$

where γ_m is a vector of m elements that are a subset of elements of γ $\mathbf{Z}_m$ is an $(n \times m)$ matrix whose columns are the corresponding subset of columns of $\mathbf{Z}$, and ϵ_m is the appropriate error term. Since the columns of $\mathbf{Z}$ are orthogonal. The least square estimate of γ is obtained by regressing $\mathbf{y}$ on $\mathbf{Z}$ in equation (2.25)[10]. The vector $\hat{\gamma}$ is given by:

$$\hat{\gamma} = \left(\mathbf{Z}'\mathbf{Z}\right)^{-1}\mathbf{Z}'\mathbf{y} \tag{2.27}$$

$$= \mathbf{L}^{-2}\mathbf{Z}'\mathbf{y}$$

where $\mathbf{L}$ is the diagonal matrix whose k th diagonal element is $l_k^{1/2}$, and $l_k, k = 1, 2, \ldots, p$ is the eigenvalues of $\mathbf{X}'\mathbf{X}$ with k being the largest eigenvalue.

The main advantage of PC regression occurs when multicollinearities are present. In this case, by deleting a subset of the PCs, especially those with small variances, much more stable estimates of $\boldsymbol{\beta}$ can be obtained[10]. Furthermore, if the regression equation is calculated for PCs instead of the predictor variables, then the contributions of each transformed variable (PC) to the equation can be more easily interpreted than the contributions of the original variables. Because of uncorrelatedness, the contribution and estimated coefficient of a PC are unaffected by which other PCs are also included in the regression, whereas for the original variables both contributions and coefficients can change dramatically when another variable is added to, or deleted from, the equation. This is especially true when multicollinearity is present, but even when multicollinearity is not a problem, regression on the PCs, rather than the original predictor variables, may have advantages for computation and interpretation[10].

2.5 Random Forests

Random forest is a popular ensemble algorithm which is used for the classification and regression in machine learning. The method was developed by Leo Breiman in 2001[13]. It combines Breiman's bagging sampling approach ((1996)[14], and the random selection of features, introduced independently by Ho (1998)[15]. it is based on averaging a collection of decorrelated decision trees. This is done by a collection of of trees constructed from a training data set and internally validated to yield a prediction of the response given the predictors for future observations. For each tree grown on a bootstrap sample, the error rate for observations left out of the bootstrap sample is monitored. This is called the "out-of-bag" error rate. A few random samples of m out of the p features are considered for splitting. Typically $m = \sqrt{p}$ where p is the number of features[1].

Random forest overcome the overfitting problem by averaging high number of decision trees built out of randomly selected sets of features. This increase it prediction accuracy and generally popular algorithm but lacks intractability .

Algorithm

Below is the algorithm for random forest regression or classification [1]

1. For $b = 1$ to B :
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $n_{\min}$ is reached.
      i Select m variables at random from the p variables
      ii Pick the best variable/split-point among the m.
      iii Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x :
Regression: $\hat{f}_{\mathrm{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$
Classification: Let $\hat{C}_b(x)$ be the class prediction of the b th random-forest tree. Then $\hat{C}_{\mathrm{rf}}^B(x) = \text{majority vote } \left\{\hat{C}_b(x)\right\}_1^B$

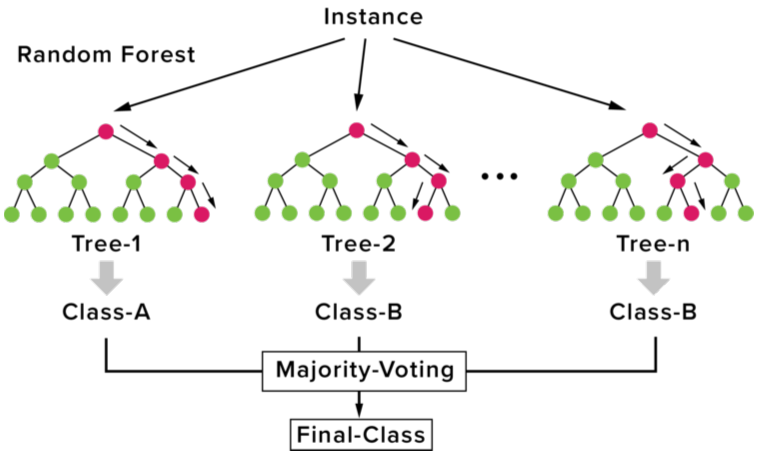


Figure 2.2: The Flow Chart of Random Forests.

Figure 2.2 is visual representation of the random forest algorithm [11].

Variable Importance

Variable importance measure is an attractive feature offered by random forest. Variable importance measure helps answer questions such as which features are important predictors of the target variable. The variable importance technique in random forests has been increasingly studied in its own right and applied as a tool for variable selection in various fields. This method generally belongs to the ”cost-of-exclusion” (Kononenko and Hong,1997)[12] feature selection category, in which the importance or relevance of a feature is determined by the difference in some model performance measure with and without the given feature included in the modeling process.

The standard random forest offers two different variable importance measures for each feature. The first measure is computed from permuting OOB data: For each tree, the prediction error on the out-of-bag portion of the data is recorded (error rate for classification, MSE for regression). Then the same is done after permuting each predictor variable. The difference between the two are then averaged over all trees, and normalized by the standard deviation of the differences. If the standard deviation of the differences is equal to 0 for a variable, the division is not done (but the average is almost always equal to 0 in that case)[1].

The second measure is the total decrease in node impurities from splitting on the variable, averaged over all trees. For classification, the node impurity is measured by the Gini index. For regression, it is measured by residual sum of squares. The Gini measure of a feature of interest is the sum of the decrease of Gini impurity criteria of the splits that are based on this feature, scaled by the total number of trees in the forest. An 'important' feature is often selected for splitting and yields a high decrease of Gini impurity when selected, leading to a high Gini measure [1] .

Partial Dependence Plot

The partial dependence plot provides a way to explore the overall effects of a feature on the response and extract interpretation from random forest. We briefly describe the idea in the regression setting. Given a prediction function, denoted by $f(\mathrm{x}) = f\left(\mathrm{x}_1, \mathrm{x}_2\right)$ by partitioning $\mathbf{x}$ into $(\mathbf{x}_1, \mathbf{x}_2)$ want to know the relative importance of feature $\mathbf{x}_1$ on the target. The partial dependence is defined as $f_1\left(\mathbf{x}_1\right) = E_{\mathbf{x}_1} f\left(\mathbf{x}_1, \mathbf{x}_2\right)$. In Regression, the estimation is given by

$$\hat{f}\left(\mathbf{x}_1\right) = \frac{1}{n} \sum_{i=1}^{n} f\left(\mathbf{x}_1, \mathbf{x}_{i2}\right) \tag{2.28}$$

In classification where the response y is categorical with levels $\{1, \ldots, C\}$, the logits (i.e., the log of the fraction of votes) are used so that

$$\hat{f}(\mathbf{x}) = \log \Pr(y = 1) - \frac{1}{C} \sum_{c=1}^{C} \log \Pr(y = c) \tag{2.29}$$

2.6 Multivariate Adaptive Regression Splines (MARS)

Multivariate adaptive regression splines was developed by Friedman [1991][16] as a non-parametric approach to regression and model-building. As such, it makes no assumptions about the relationship between the predictors and the response variable. MARS is an additive approach for regression, and is well suited for high dimensional problems, that is large number of inputs[16].

MARS uses piecewise linear basis functions of the form $(x-t)_+$ and $(t-x)_+$ known as hinge fuction shown in Figure 2.3. The hing fuction is defined as:

$$(x-t)_+ = \begin{cases} x-t, & \text{if } x > t \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad (t-x)_+ = \begin{cases} t-x, & \text{if } x < t \\ 0, & \text{otherwise} \end{cases}$$

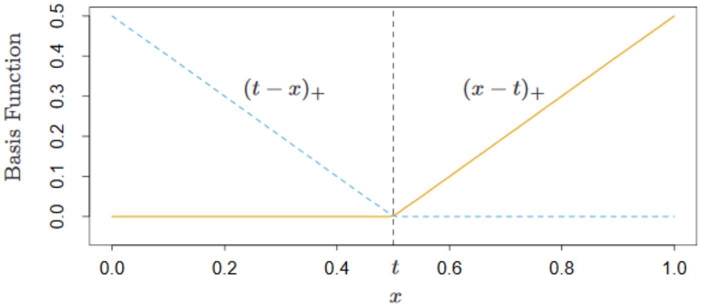


Figure 2.3: The basis functions $(x-t)+$ (solid orange) and $(t-x)+$ (broken blue used by MARS.

The key property of the functions of Figure 2.3 is their ability to operate locally; they are zero over part of their range. When they are multiplied together, the result is nonzero only over the small part of the feature space where both component functions are nonzero. As a result, the regression surface is built up parsimoniously, using nonzero components locally only where they are needed. The use of other basis functions such as polynomials, would produce a nonzero product everywhere, and would not work as well[1].

Each function is piecewise linear, with a knot at the value t. The idea is to form reflected pairs for each input X_j with knots at each observed value x_{ij} of that input. The result is a collection of basis functions given by:

$$\mathcal{C} = \left\{ (X_j - t)_+ , (t - X_j) + \right\}, t \in \{x_{1j}, x_{2j}, \ldots, x_{Nj}\} \tag{2.30}$$

and $j = 1, 2, \ldots, p$,

The model-building strategy is a forward stepwise linear regression, but uses functions from the set $\mathcal{C}$ and their products instead of using the original observations. Thus the model has the form

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X) \tag{2.31}$$

where each $h_m(X)$ is a function in $\mathcal{C}$, or a product of two or more such functions.

In the model building process of (2.31) the model typically overfits the data, and so a backward deletion procedure is applied. The term whose removal causes the smallest increase in residual squared error is deleted from the model at each stage, producing an estimated best model $\hat{f}_\lambda$ of each size (number of terms) λ. Generalized cross-validation [1] is used to estimate the optimal value of λ, the criterion is given by:

$$\mathrm{GCV}(\lambda) = \frac{\sum_{i=1}^{N} \left(y_i - \hat{f}_\lambda \left(x_i \right) \right)^2}{(1 - M(\lambda)/N)^2} \tag{2.32}$$

The value $M(\lambda)$ is the effective number of parameters in the model: this accounts both for the number of terms in the models, plus the number of parameters used in selecting the optimal positions of the knots [1].

Algorithm for MARS

Below is the algorithm for MARS [1]

```
Data: Training data $\mathcal{D} = \{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ for regression

1. begin
    (a) set $M_{\max}$, the maximum number of terms;
    (b) initialize $M = 0, b_0(\mathrm{x}_i) = 1$, and model $h(\mathrm{x}_i) = \beta_0 b_0(\mathrm{x}_i)$
    (c) repeat
```

i Find all allowable candidate terms:

$\left\{(x_{ij}-t)_+, x_{ij}\right\}$ for continuous x_{ij} and $I(x_{ij}\in A)$ for categorical x_j

ii Perform greedy search for the best term associated with model

$$h(\mathbf{x}_i)+\sum_{m=0}^{M}\alpha_m b_m(\mathbf{x}_i)\cdot x_{ij}+\sum_{m=0}^{M}\gamma_m b_m(\mathbf{x}_i)\cdot(x_{ij}-t)_+ \text{ for continuous } X_j$$

$$h(\mathbf{x}_i)+\sum_{m=0}^{M}\alpha_m b_m(\mathbf{x}_i)\cdot I(x_{ij}\in A) \qquad \text{for categorical } X_j$$

which may be further modified to conform to the constraints;

iii Let M' denote the number of added terms associated the "best' basis pair;

iv Update $M := M + M'$

v Update model $h(\mathbf{x}_i) := \sum_{m=0}^{M}\beta_m b_m(\mathbf{x}_i)$

(d) **Until** $M > M_{\max}$ or no more permissible candidate terms;

2. **end**.

Result: A large initial MARS model $y_i=\sum_{m=0}^{M_{\max}}\beta_m b_m(\mathbf{x}_i)+\varepsilon_i$

2.7 Support Vector Machine (SVM)

SVM is a supervised machine learning method that is used for both classification and regression problems. It finds a hyperplane that separates the classes in feature space and predicts the class of test observations using the separation boundary[17].

Hyperplane

A hyperplane in p dimensions is a flat affine subspace of dimension p − 1. In general, the equation for a hyperplane has the form $x^\top\beta+\beta_0=0$. The vector β is called the normal vector and it points in a direction orthogonal to the surface of a hyperplane. If $\beta_0 = 0$, the hyperplane goes through the origin. Example of a hyperplane in two dimension (line) is in Figure 2.4 [1] .

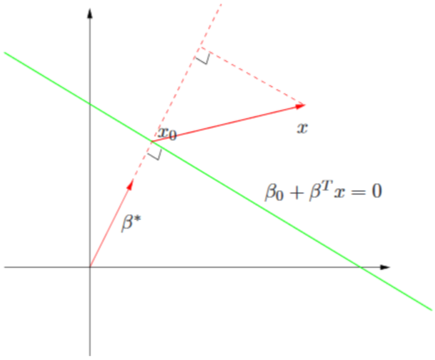


Figure 2.4: The linear algebra of a hyperplane:Hyperplane in 2 Dimensions.[17]

Separating Hyperplane

A separating hyperplane for two sets is a hyperplane for which one set lies on one side of the hyperplane, and the other set lies on the other side. Given training data consists of N pairs of observations $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, with $x_i \in \mathrm{R}^p$ and $y_i \in \{-1, 1\}$. If $f(x) = x^\top \beta + \beta_0$, then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other. If observations are coded such that for point on one side of the hyperplane $y_i = +1$ and $y_i = -1$ for the other side, then if $y_i f(x_i) > 0$ for all i $f(x) = x^\top \beta + \beta_0 = 0$ defines a separating hyperplane [1].

Maximal margin classifier

Since there may be infinitely many possible separating hyperplanes for a given data, the maximal margin classifier find the separating hyperplane that makes the biggest margin (gap) M between the two classes. The resulting optimization problem is given by [1]:

$$\max_{\beta, \beta_0, \|\beta\|=1} M : \quad \text{subject to } y_i \left(x_i^T \beta + \beta_0 \right) \geq M, i = 1, \ldots, N \tag{2.33}$$

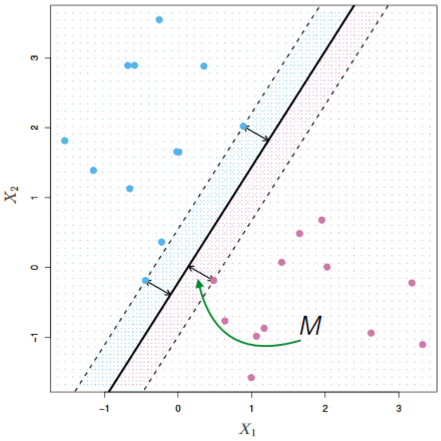


Figure 2.5: Maximal Margin Classifier.[17]

The band in the figure is M units away from the hyperplane on either side, and hence $2M$ units wide. The set of conditions in (2.33) ensure that all the points are at least a signed distance M from the decision boundary defined by β and β_0, and we seek the largest M and associated parameters.

The problem can be rephrases by replacing the conditions in (2.33) with

$$\frac{1}{\|\beta\|} y_i \left(x_i^T \beta + \beta_0 \right) \geq M \tag{2.34}$$

or equivalently:

$$y_i \left(x_i^T \beta + \beta_0 \right) \geq M \|\beta\| \tag{2.35}$$

where $\|\beta\| = 1/M$. Equation (2.33) is equivalent to:

$$\min_{\beta, \beta_0} \frac{1}{2} \|\beta\|^2 : \quad \text{subject to } y_i \left(x_i^T \beta + \beta_0 \right) \geq 1, i = 1, \ldots, N \tag{2.36}$$

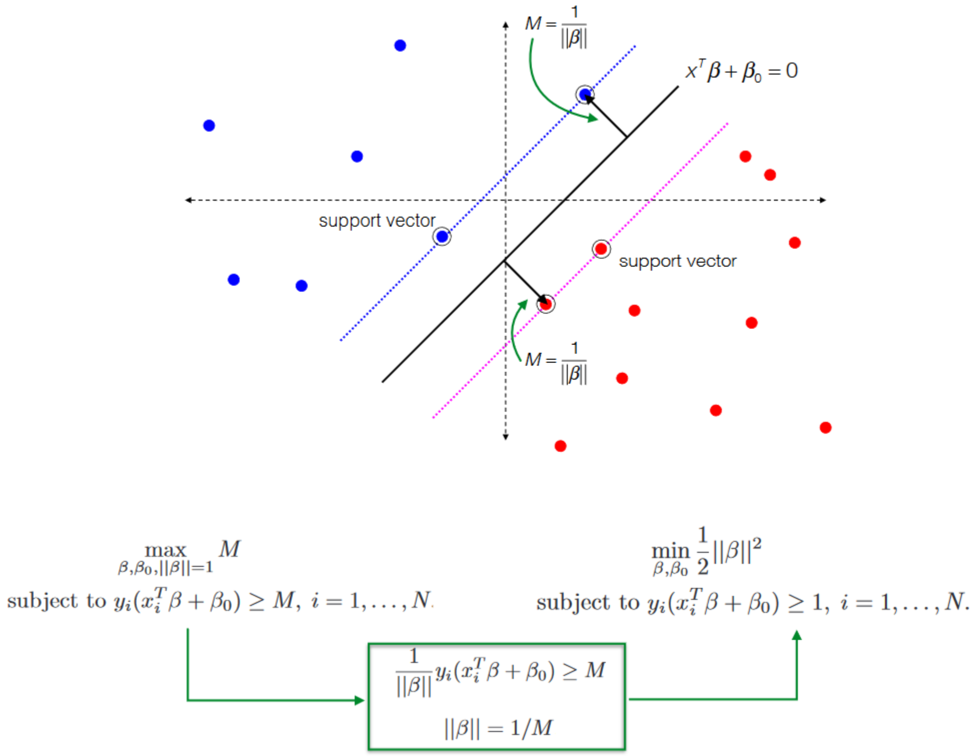

$M = \frac{1}{\|\beta\|}$
$x^T\beta + \beta_0 = 0$
support vector
support vector
$M = \frac{1}{\|\beta\|}$
$\max_{\beta,\beta_0,\|\beta\|=1} M$
subject to $y_i(x_i^T\beta + \beta_0) \geq M,\ i = 1, \ldots, N.$
$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$
subject to $y_i(x_i^T\beta + \beta_0) \geq 1,\ i = 1, \ldots, N.$
$\frac{1}{\|\beta\|} y_i(x_i^T\beta + \beta_0) \geq M$
$\|\beta\| = 1/M$

Figure 2.6: Maximal Margin Classifier.[1]

This is a convex optimization problem with quadratic criterion, linear inequality constraints. The Lagrange (primal) function, to be minimized w.r.t. β and β_0, is

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i \left(x_i^T\beta + \beta_0 \right) - 1 \right] \tag{2.37}$$

Setting the partial derivatives to zero w.r.t β and β_0, we obtain:

$$\beta = \sum_{i=1}^{M} \alpha_i y_i x_i \tag{2.38}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{2.39}$$

and substituting (2.38) and (2.39) into (2.37) gives the dual representation of the Lagrangian (primal) function:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k \tag{2.40}$$

To obtain optimal solution Kraus-Kuhn-Tucker (KKT) condition [19] must be satisfied for all $i = 1, .., N$:

$$\alpha_i \geq 0$$
$$\sum_{i=1}^{N} \alpha_i y_i = 0$$
$$\alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - 1 \right] = 0 \forall i$$

Using quadratic programming, we can find $\alpha_i s$ that minimize L_D and also satisfying KKT condition. The KKT condition then becomes

$$\hat{\alpha}_i \left\{ y_i (\hat{\boldsymbol{\beta}}^T \mathbf{x}_i + \hat{\beta}_0) - 1 \right\} = 0 \tag{2.41}$$

for $i = 1, \ldots, n$. This leads to two scenarios

$$\begin{cases} \text{if } \hat{\alpha}_i = 0 & y_i (\hat{\boldsymbol{\beta}}^T \mathbf{x}_i + \hat{\beta}_0) > 1 \\ \text{if } \hat{\alpha}_i > 0 & y_i (\hat{\boldsymbol{\beta}}^T \mathbf{x}_i + \hat{\beta}_0) = 1 \end{cases} \tag{2.42}$$

In the latter case of $\hat{\alpha}_i > 0$, the condition $y_i (\hat{\boldsymbol{\beta}}^T \mathbf{x}_i + \hat{\beta}_0) = 1$ implies that the points x_i, associated with y_i, must lie right on the margin. These points are called 'support vectors'. Thus the $\boldsymbol{\beta}$ is estimated by:

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{n} \hat{\alpha}_i \cdot y_i \mathbf{x}_i = \sum_{i:\hat{\alpha}_i \neq 0} \hat{\alpha}_i \cdot y_i \mathbf{x}_i \tag{2.43}$$

To estimate $\boldsymbol{\beta_0}$ the support vectors with $\hat{\alpha}_i > 0$ is used from KKT condition .

$$\left\{ y_i \left(\hat{\boldsymbol{\beta}}^T \mathbf{x}_i + \hat{\beta}_0 \right) - 1 \right\} = 0$$

It follows that

$$\hat{\beta}_0 = y_i - \hat{\boldsymbol{\beta}}^T \mathbf{x}_i$$

Given a new observation with $\mathbf{x}$, we predict its outcome as sign $\left\{ \hat{\boldsymbol{\beta}}^T \mathbf{x} + \hat{\beta}_0 \right\}$[1].

Soft-Margin SVM

The Soft Margin Classifier or the Support Vector Classifier is a generalization of the maximal margin classifier to the non-linearly separable classes or classes that overlap in feature space. It allow some observation to be miss-classified hence result in a more robust hyperplane. The Soft Margin SVM method choose a hyperplane that maximize M, but allow for some points to be on the wrong side of the margin. The method introduces slack variables, ξ_i, which measure the degree of misclassification of the i-th individual[1].

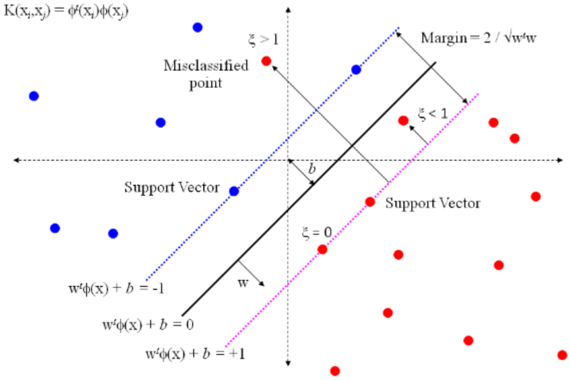


Figure 2.7: Illustration of Soft-Margin SVM.[1]

There are two ways to modify the constraint in (2.33):

$$\begin{gathered} y_i \left(x_i^T \beta + \beta_0 \right) \geq M - \xi_i \\ \text{or} \\ y_i \left(x_i^T \beta + \beta_0 \right) \geq M \left(1 - \xi_i \right) \end{gathered} \tag{2.44}$$

for all $\xi_i \geq 0, \sum_{i=1}^{N} \xi_i \leq$ constant.

The first option is more natural in the sense that ξ_i directly measures the distance from a wrongly classified observation $\mathbf{x}_i$ to its corresponding margin. However, it results in a non-convex programming program. Thus the second option is preferred. Thus the resulting optimization problem is given by:

$$\max_{\beta,\beta_0,M} M, \text{ subject to } \|\beta\| = 1, \text{ and } y_i\left(\beta^T \mathbf{x}_i + \beta_0\right) \geq M\left(1 - \xi_i\right) \tag{2.45}$$

with $\xi_i \geq 0$ and $\sum_{i=1}^n \xi_i \leq C$ for $i = 1, \ldots, n$. Dropping the constraint $\|\boldsymbol{\beta}\| = 1$ and then setting $M = 1/\|\boldsymbol{\beta}\|$, we re-express the problem as

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{2}\|\boldsymbol{\beta}\|^2, \text{ s.t. } y_i\left(\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0\right) \geq 1 - \xi_i, \xi_i \geq 0, \text{ and } \sum_{i=1}^n \xi_i \leq C \tag{2.46}$$

Using the penalization method, the above problem can be equivalently rewritten as

$$\begin{aligned} &\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^N \xi_i \\ &\text{subject to} \quad \xi_i \geq 0, y_i\left(x_i^T \beta + \beta_0\right) \geq 1 - \xi_i \forall i \end{aligned} \tag{2.47}$$

where C is the "cost" parameter which regulate the level of miss classifications allowed[1]. The separable case corresponds to $C = \infty$.

This is also a convex optimization problem with quadratic criterion and linear inequality constraints. The Lagrange (primal) function, to be minimized w.r.t. β and β_0, and ξ_i is

$$L_P = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i\left[y_i\left(x_i^T \beta + \beta_0\right) - \left(1 - \xi_i\right)\right] - \sum_{i=1}^N \mu_i \xi_i \tag{2.48}$$

Setting the respective derivatives to zero, we gives:

$$\beta = \sum_{i=1}^N \alpha_i y_i x_i \tag{2.49}$$

$$0 = \sum_{i=1}^N \alpha_i y_i \tag{2.50}$$

$$\alpha_i = C - \mu_i, \forall i \tag{2.51}$$

as well as the constraints $\alpha_i, \mu_i, \xi_i \geq 0$ Vi. By substituting (2.50)- (2.52) into (2.49), we obtain the Lagrangian (Wolfe) dual objective function

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'} \tag{2.52}$$

which gives a lower bound on the objective function (2.48) for any feasible point. We maximize L_D subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$. In addition to (2.50) – (2.52), the Karush-Kuhn-Tucker conditions [19] include the constraints

$$\begin{aligned} \alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - (1 - \xi_i) \right] &= 0 \\ \mu_i \xi_i &= 0 \\ y_i \left(x_i^T \beta + \beta_0 \right) - (1 - \xi_i) &\geq 0 \end{aligned} \tag{2.53}$$

for $i = 1, \ldots, N$. Together these equations (2.50) – (2.54) uniquely characterize the solution to the primal and dual problem.

From (2.50) we see that the solution for β has the form

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i \tag{2.54}$$

Maximizing the dual (2.53) is a simpler convex quadratic programming problem than the primal (2.49), and can be solved with standard techniques [Murray et al., 1981] [20].

Given the solutions $\hat{\beta}_0$ and $\hat{\beta}$, the decision function can be written as

$$\begin{aligned} \hat{G}(x) &= \text{sign}[\hat{f}(x)] \\ &= \text{sign}\left[x^T \hat{\beta} + \hat{\beta}_0 \right] \end{aligned}$$

The tuning parameter of this procedure is the cost parameter C[1].

Support Vector Machines (Kernels)

The support vector machine(SVM) is an extension of the support vector classifier. They are use when the decision boundary is non-linear and assuming linear boundary in the input space is not reasonable. The non-linearity is introduced through the use of kernels. The idea is to enlarge the feature space such that the data is linearly separable in the enlarged space [17].

SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data are transformed in such a way that the separator could be drawn as a hyperplane. Following this, characteristics of new data can be used to predict the group to which a new record should belong [18].

The optimization problem (2.45) and its solution for SVM are rephrased in a way that involves the input features through inner products. The Lagrange dual function (2.53) has the form

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \left\langle h\left(x_i\right), h\left(x_{i'}\right) \right\rangle \tag{2.55}$$

where $h\left(x_i\right)$ is a transformed feature vectors. From (2.50) the solution $f(x)$ can be written as

$$f(x) = h(x)^T \beta + \beta_0$$

$$= \sum_{i=1}^{N} \alpha_i y_i \left\langle h(x), h\left(x_i\right) \right\rangle + \beta_0 \tag{2.56}$$

where α_i, β_0 can be determined by solving $y_i f\left(x_i\right) = 1$ in (2.57) for any x_i for which $0 < \alpha_i < C$. Both (2.56) and (2.57) involve $h(x)$ only through inner products therefore the transformation $h(x)$ is not needed only knowledge of the kernel function [1]:

$$K\left(x, x'\right) = \left\langle h(x), h\left(x'\right) \right\rangle \tag{2.57}$$

that computes inner products in the transformed space. K should be a symmetric positive (semi-) definite function. Three popular choices for K in the SVM literature are [1]:

$$d \text{ th-Degree polynomial: } K\left(x, x'\right) = \left(1 + \left\langle x, x' \right\rangle\right)^d$$

$$\text{Radial basis: } K\left(x, x'\right) = \exp\left(-\gamma \left\| x - x' \right\|^2\right)$$

$$\text{Neural network: } K\left(x, x'\right) = \tanh\left(\kappa_1 \left\langle x, x' \right\rangle + \kappa_2\right)$$

Example: suppose a feature space with two inputs X_1 and X_2, and a polynomial kernel of degree 2. Then

$$\begin{aligned} K\left(X, X'\right) &= \left(1 + \left\langle X, X' \right\rangle\right)^2 \\ &= \left(1 + X_1 X_1' + X_2 X_2'\right)^2 \\ &= 1 + 2X_1 X_1' + 2X_2 X_2' + \left(X_1 X_1'\right)^2 + \left(X_2 X_2'\right)^2 + 2X_1 X_1' X_2 X_2' \end{aligned} \tag{2.58}$$

Then $M = 6$, and if we choose $h_1(X) = 1, h_2(X) = \sqrt{2} X_1, h_3(X) = \sqrt{2} X_2, h_4(X) = X_1^2, h_5(X) = X_2^2$, and $h_6(X) = \sqrt{2} X_1 X_2$, then $K\left(X, X'\right) = \left\langle h(X), h\left(X'\right)\right\rangle$. From (2.57) the solution can be written as

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i K\left(x, x_i\right) + \hat{\beta}_0 \tag{2.59}$$

The regularization parameter C is clearer in an enlarged feature space, since perfect separation is often achievable there. A large value of C will discourage any positive ξ_i, and lead to an overfit wiggly boundary in the original feature space; a small value of C will encourage a small value of $\|\beta\|$, which in turn causes $f(x)$ and hence the boundary to be smoother [1].

2.8 Gradient boosting

Gradient Boosting is one of many boosting methods. The motivation for boosting procedure is to combines the outputs of many weak classifiers to produce a powerful committee. The algorithm can be applied to both classification and regression settings. Boosting work by fitting a tree to the entire training set, but adaptively weight the observations to encourage better predictions for points that were previously misclassified. In boosting, trees are grown sequentially with each tree is grown using information from previously grown trees[1].

The basic principles of gradient boosting are as follows: given a loss function (e.g., squared error for regression) and a weak learner (e.g., regression trees), the algorithm seeks to find an additive model that minimizes the loss function. The algorithm is typically initialized with the best guess of the response (e.g., the mean of the response in regression). The gradient (e.g., residual) is calculated, and a model is then fit to the residuals to minimize the loss function. The current model is added to the previous model, and the procedure continues for a user specified number of iterations [21]. The different loss functions used for both numerical or categorical response are displayed in the table below.

Setting	Loss Function	$-\partial L\left(y_i, f\left(x_i\right)\right) / \partial f\left(x_i\right)$
Regression	$\frac{1}{2}\left[y_i - f\left(x_i\right)\right]^2$	$y_i - f\left(x_i\right)$
Regression	$\left\|y_i - f\left(x_i\right)\right\|$	$\operatorname{sign}\left[y_i - f\left(x_i\right)\right]$
Regression	Huber	$y_i - f\left(x_i\right)$ for $\left\|y_i - f\left(x_i\right)\right\| \leq \delta_m$ $\delta_m \operatorname{sign}\left[y_i - f\left(x_i\right)\right]$ for $\left\|y_i - f\left(x_i\right)\right\| > \delta_m$ where $\delta_m = \alpha$ th-quantile $\left\{\left\|y_i - f\left(x_i\right)\right\|\right\}$
Classification	Deviance	k th component: $I\left(y_i = \mathcal{G}_k\right) - p_k\left(x_i\right)$

A detailed description of the gradient boosting is as follows: a given tree can be represented as:

$$T(x;\Theta)=\sum_{j=1}^{J}\gamma_j I\left(x\in R_j\right) \tag{2.60}$$

the tree parameters $\Theta=\{R_j,\gamma_j\}$, where j is an index of the terminal node, $j,\ldots,J,R_j$ a predictor-space region defined by the j th terminal node, and γ_j is the value assigned to each observation in the j th terminal node. The goal is to construct values for the unknown parameters Θ so that the loss function is minimized [22]. At this point, no particular loss L is specified, and we seek

$$\hat{\Theta}=\arg\min_{\Theta}\sum_{j=1}^{J}\sum_{x_i\in R_j}L\left(y_i,\gamma_j\right) \tag{2.61}$$

Using stagewise algorithm [23] we can approximate (2.61) with

$$\hat{\Theta}_m=\arg\min_{\Theta_m}\sum_{i=1}^{N}L\left(y_i,f_{m-1}\left(x_i\right)+T\left(x_i;\Theta_m\right)\right) \tag{2.62}$$

where $f_{m-1}\left(x_i\right)$ are the results as of the previous tree. Given the results from the previous tree, the intent is to reduce the loss as much as possible using the fitted values from the next tree.

Equation (2.61) can be reformulated as a numerical optimization task, where:

$$g_{im}=-\left[\frac{\partial L\left(y_i,f\left(x_i\right)\right)}{\partial f\left(x_i\right)}\right]_{f\left(x_i\right)=f_{m-1}\left(x_i\right)} \tag{2.63}$$

is the gradient for the i th observation on iteration m, defined as the partial derivative of the loss with respect to the fitting function. One approach to obtain solution is to use "steepest descent," algorithm [1] in which a "step length" ρ_m is the solution to

$$\rho_m=\arg\min_{\rho}L\left(\mathbf{f}_{m-1}-\rho\mathbf{g}_m\right) \tag{2.64}$$

The current solution is then updated as: $\mathbf{f}_m = \mathbf{f}_{m-1} - \rho_m \mathbf{g}_m$ and the process repeated at the next iteration.

The gradient boosting has two tuning parameters: tree depth and number of iterations. Tree depth in this context is also known as interaction depth, since each subsequential split can be thought of as a higherlevel interaction term with all of the other previous split predictors. Given the loss function, the gradient boosting algorithm is given below:.

2.8.1 Algorithm for Gradient Boosting

Below is the algorithm for random forest regression or classification [1]

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L\left(y_i, \gamma\right)$.
2. For $m = 1$ to M :
   (a) For $i = 1, 21..., N$ compute

   $$r_{im} = -\left[\frac{\partial L\left(y_i, f\left(x_i\right)\right)}{\partial f\left(x_i\right)}\right]_{f=f_{m-1}}$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm}, j = 1, 2, \ldots, J_m$

   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}\left(x_i\right) + \gamma\right)$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I\left(x \in R_{jm}\right)$
3. Output $\hat{f}(x) = f_M(x)$.

2.9 Neural networks

Neural networks are powerful nonlinear regression techniques inspired by theories about how the brain works(Bishop 1995 [24]; Ripley 1996[25]; Titterington 2010 [26]). A neural network is a two stage regression or classification model, typically represented by a network diagram as in Figure 2.8. This network applies both to regression or classification [1].

2.9.1 Model definition and description

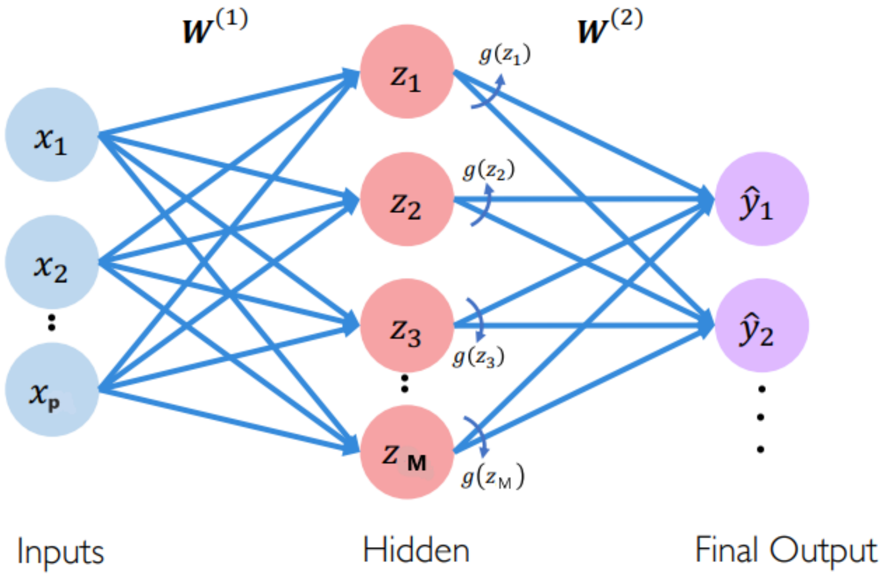


Figure 2.8: Diagram of a single hidden layer, feed-forward neural network [27].

For regression $K = 1$ and there is only one output unit Y_1. In classification setting with $K-$class, there are K output units, with the kth unit modeling the probability of class k. The units in the middle of the network compute the derived features Z_m, called hidden units because the values Z_m are not directly observed. In general there can be more than one hidden layer than illustrated in Figure 2.8 [1].

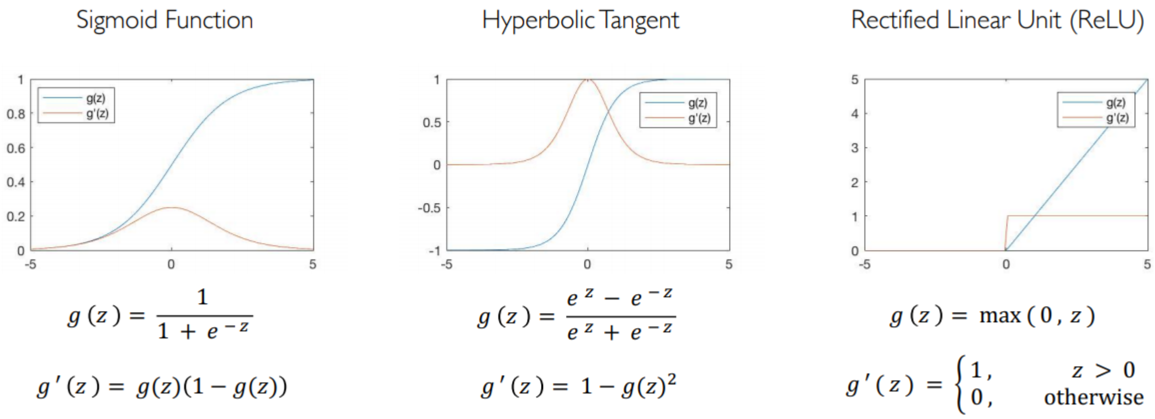
The derived features Z_m are created from linear combinations of the inputs, and then the target Y_k is modeled as a function of linear combinations of the Z_m as follows [1]:

$$Z_m = \sigma\left(\alpha_{0m} + \alpha_m^T X\right), m = 1, \ldots, M \tag{2.65}$$

$$T_k = \beta_{0k} + \beta_k^T Z, k = 1, \ldots, K \tag{2.66}$$

$$f_k(X) = g_k(T), k = 1, \ldots, K \tag{2.67}$$

where $Z = (Z_1, Z_2, \ldots, Z_M)$, and $T = (T_1, T_2, \ldots, T_K)$. The nonlinear function $\sigma()$ in (2.65) is called the activation function. In practice different activation functions are used for different problems, Figure 2.9 shows some of the most commonly used activation functions[27].



Sigmoid Function

$g(z) = \frac{1}{1 + e^{-z}}$

$g'(z) = g(z)(1 - g(z))$

Hyperbolic Tangent

$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$

$g'(z) = 1 - g(z)^2$

Rectified Linear Unit (ReLU)

$g(z) = \max(0, z)$

$g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases}$

Figure 2.9: Common Activation Functions [27].

The output function $g_k(T)$ allows a final transformation of the vector of outputs T. For regression we used the identity function $g_k(T) = T_k$ and for $K - class$ classification the softmax function defined in (2.68) is used.

$$g_k(T) = \frac{e^{T_k}}{\sum_{\ell=1}^{K} e^{T_\ell}} \tag{2.68}$$

2.9.2 Model fitting

The unknown parameters in a neural network model are called weights, and we seek values for them that make the model fit the training data well. We denote the complete set of weights by θ, which consists of [1]:

$$\begin{aligned} &\{\alpha_{0m}, \alpha_m; m = 1, 2, \ldots, M\}\; M(p+1) \text{ weights,} \\ &\{\beta_{0k}, \beta_k; k = 1, 2, \ldots, K\}\; K(M+1) \text{ weights.} \end{aligned} \tag{2.69}$$

For regression, we use sum-of-squared errors as our measure of fit (error function):

$$R(\theta) = \sum_{k=1}^{K} \sum_{i=1}^{N} \left(y_{ik} - f_k\left(x_i\right)\right)^2 \tag{2.70}$$

For classification we use either squared error or cross-entropy (deviance):

$$R(\theta) = -\sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log f_k\left(x_i\right) \tag{2.71}$$

The solution is usually obtained by minimizing $R(\theta)$, this may usually overfit the data. Instead some regularization is applied directly through a penalty term, or indirectly by early stopping[1].

The generic approach to minimizing $R(\theta)$ is by gradient descent, called back-propagation in this setting (Rumelhartet al. 1986)[28]. Because of the compositional form of the model, the gradient can be easily derived using the chain rule for differentiation. This can be computed by a forward and backward sweep over the network, keeping track only of quantities local to each unit [1].

Below is the detail of the back-propagation method for squared error loss [1]. Let $z_{mi} = \sigma\left(\alpha_{0m} + \alpha_m^T x_i\right)$, from (2.66)-(2.68) and let $z_i = \left(z_{1i}, z_{2i}, \ldots, z_{Mi}\right)$. Then we have

$$\begin{aligned} R(\theta) &\equiv \sum_{i=1}^{N} R_i \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} \left(y_{ik} - f_k\left(x_i\right)\right)^2 \end{aligned} \tag{2.72}$$

with derivatives

$$\begin{aligned} \frac{\partial R_i}{\partial \beta_{km}} &= -2\left(y_{ik} - f_k\left(x_i\right)\right) g_k'\left(\beta_k^T z_i\right) z_{mi} \\ \frac{\partial R_i}{\partial \alpha_{m\ell}} &= -\sum_{k=1}^{K} 2\left(y_{ik} - f_k\left(x_i\right)\right) g_k'\left(\beta_k^T z_i\right) \beta_{km} \sigma'\left(\alpha_m^T x_i\right) x_{i\ell} \end{aligned} \tag{2.73}$$

Given these derivatives, a gradient descent update at the $(r+1)$ st iteration has the form

$$\begin{aligned} \beta_{km}^{(r+1)} &= \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}^{(r)}} \\ \alpha_{m\ell}^{(r+1)} &= \alpha_{m\ell}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{m\ell}^{(r)}} \end{aligned} \tag{2.74}$$

where γ_r is the learning rate, discussed below. Now write (2.74) as

$$\begin{aligned} \frac{\partial R_i}{\partial \beta_{km}} &= \delta_{ki} z_{mi} \\ \frac{\partial R_i}{\partial \alpha_{m\ell}} &= s_{mi} x_{i\ell} \end{aligned} \tag{2.75}$$

The quantities δ_{ki} and s_{mi} are "errors" from the current model at the output and hidden layer units, respectively. From their definitions, these errors satisfy

$$s_{mi} = \sigma'\left(\alpha_m^T x_i\right) \sum_{k=1}^{K} \beta_{km} \delta_{ki} \tag{2.76}$$

known as the back-propagation equations. Using this, the updates in (2.75) can be implemented with a two-pass algorithm. In the forward pass, the current weights are fixed and the predicted values $\hat{f}_k\left(x_i\right)$ are computed from formula (2.66) – (2.68). In the backward pass, the errors δ_{ki} are computed, and then back-propagated via (2.77) to give the errors s_{mi}. Both sets of errors are then used to compute the gradients for the updates in (2.75) via (2.76). This two-pass procedure is what is known as $back-propagation$[1].

2.9.3 Issues in Training Nueral networks

The Neural Network model is generally over parametrized, and the optimization problem is nonconvex and unstable [1]. Also the backpropagation approach comes with several complications[22].

First, the start values can be a problem because the loss functions are not convex hence the search can get stuck in a local minimum. A common approach is to repeat the estimation several times with different sets of start values and then choose the result with the smallest value of the loss. Also, there is some reason to think that for very high dimensional data, the local minimums will not be all that different from the global minimum[22].

Second, with so many weights, overfitting can be a serious problem. Some form a regularization can help. Penalizing the fit using ridge regression is one option[22].

Third, the different units in which the inputs are measured can make a big difference. Standardizing the inputs is usually helpful[22].

Fourth, in Figure 2.8 each input is connected to each hidden variable, and each hidden variable is connected to the target. This makes the network saturated. No inputs are directly linked to the target. In short, there can be a large number of different network structures, which implies that some weights can be set to 0.0.[22]. Finally, the number of latent variables and hidden layers are tuning parameters typically arrived at through some combination of subject-matter knowledge and performance in cross-validation[22].

Chapter 3

Model Assessment and Selection

The generalization performance of a learning method relates to its prediction capability on independent test data. Assessment of this performance is extremely important in practice, since it guides the choice of learning method, and gives us a measure of the quality of the final chosen model [1]. The two objective for assessing the performance of a model, are: 1. Model selection which involves estimating the performance of different models in order to choose the best one and 2. Model assessment which involves estimating the prediction error (generalization error) of the best model on new data [1].

In this chapter we describe the key concepts and methods for performance assessment and how they are used to select models.

3.1 Model Assessment

3.1.1 Loss Function

For regression problem, with quantitative response variable Y, a vector of inputs X, and a prediction model $\hat{f}(X)$ that has been estimated from a training set $\mathcal{T}$. The loss function for measuring errors between Y and $\hat{f}(X)$ is denoted by $L(Y, \hat{f}(X))$[1]. Typical choices are

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \text{squared error} \\ |Y - \hat{f}(X)| & \text{absolute error} \end{cases} \tag{3.1}$$

3.2 Test Error

Model accuracy is measured using test error. Test error (also called generalized error) measures how well a model trained on a set $\mathcal{T}$ generalize to data that we have not seen before (test set). The test error is defined by

$$\mathrm{Err}_{\mathcal{T}} = \mathrm{E}[L(Y, \hat{f}(X))|\mathcal{T}] \tag{3.2}$$

where both X and Y are drawn randomly from their joint distribution (population). Expected prediction error is an average test error over all training data defined as.

$$\mathrm{Err} = \mathrm{E}[L(Y, \hat{f}(X))] = \mathrm{E}\left[\mathrm{Err}_{\mathcal{T}}\right] \tag{3.3}$$

The goal is to estimate Err $_{\mathcal{T}}$ but most methods effectively estimate the expected error Err as a measure of model accuracy or a measure of fit quality[1].

3.3 Estimating Test Error

3.3.1 Train Error

Train error is based on data points that have been used for training a model or estimate its parameters, that is the average loss over the training sample given by

$$\overline{\mathrm{err}} = \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, \hat{f}\left(x_i\right)\right) \tag{3.4}$$

The train error generally is not a good estimate of the test error because many statistical methods specifically estimate coefficients so as to minimize the train error. For these methods, the train error can be quite small, but the test error is often much larger. Therefore there is no guarantee that the method with the lowest training error will also have the lowest test error [17].

Also as a model becomes more and more complex, it is able to adapt to more complicated underlying structures in the train set. Hence there is a decrease in bias but an increase in variance which will result in overfitting the train set (low bias =low train error) and poor generalization/prediction accuracy (high variance in prediction = high test error)[1]. This is illustrated in Figure 1.1.

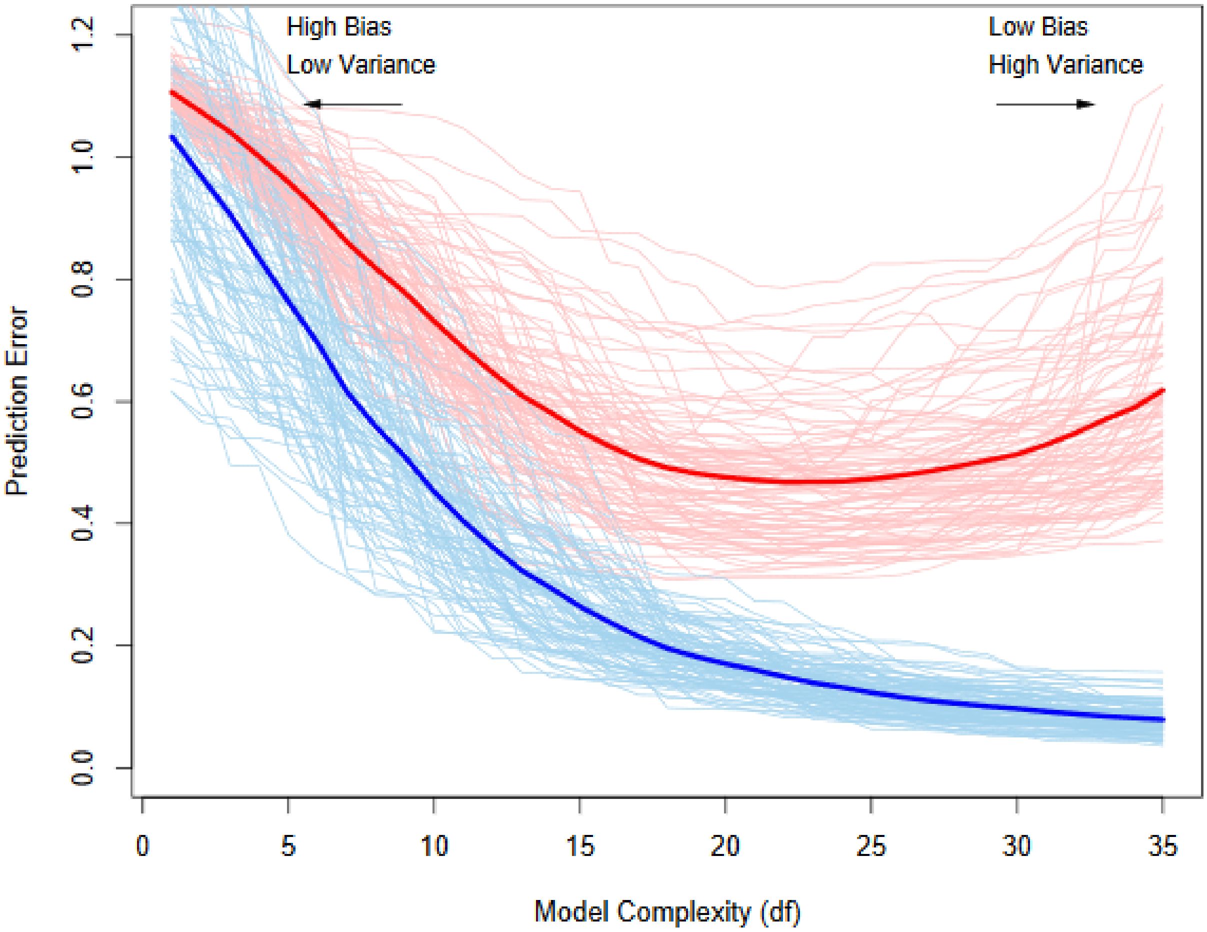


Figure 3.1: Expected training error vs expected test error [1]

Figure 1.1 explains the behavior of test sample and training sample error as the model complexity is varied. The solid curves show the expected test error Err and the expected training error E[err].

For classification setting a typical loss functions is given by:

$$\begin{aligned} L(G, \hat{G}(X)) &= I(G \neq \hat{G}(X)) \quad (0-1 \text{ loss }) \\ L(G, \hat{p}(X)) &= -2 \sum_{k=1}^{K} I(G=k) \log \hat{p}_k(X) \\ &= -2 \log \hat{p}_G(X) \quad (-2 \times \log - \text{ likelihood }) \end{aligned} \tag{3.5}$$

where G, is a categorical response with K classes. The estimated class of X, $\hat{G}(X) = \arg\max_k \hat{p}_k(X)$ if we model the probabilities $p_k(X) = \Pr(G = k|X)$, that is the probability that the predicted class of G is k for a given x value. The quantity $-2\times$ the log-likelihood is sometimes referred to as the deviance [1]. The test error here is define as:

$$\mathrm{Err}_{\mathcal{T}} = \mathrm{E}[L(G, \hat{G}(X))|\mathcal{T}] \tag{3.6}$$

which is the population misclassification error of the classifier trained on $\mathcal{T}$, and Err is the expected misclassification error. Training error is given by:

$$\overline{\mathrm{err}} = -\frac{2}{N} \sum_{i=1}^{N} \log \hat{p}_{g_i}(x_i) \tag{3.7}$$

The training error has been shown to be a poor estimate of the test error in Figure 1.1, in next section we discus two common approaches for estimating the test error. We begin with an indirectly estimate of test error which makes an adjustment to the training error to account for the bias due to overfitting and continue with a direct estimate of the test error, using either a validation set approach or a cross-validation approach [1].

3.4 Adjusted Train Error

The most common adjusted train error estimates of the test error are C_p, AIC, BIC, and Adjusted R^2 are briefly discussed below.

3.4.1 Mellow's C_p

Mallow (1973)[33] derived a C_p criterion by examining the expected model error. Suppose d parameters are fit under squared error loss, then C_p statistic is:

$$C_p = \overline{\mathrm{err}} + 2 \cdot \frac{d}{N} \hat{\sigma}_{\varepsilon}^2 \tag{3.8}$$

where $\hat{\sigma}_{\varepsilon}^2$ is an estimate of the noise variance, obtained from the mean squared error of a low-bias model. For a fitted least squares model containing d predictors, the C_p estimate of test MSE is computed using the equation

$$C_p = \frac{1}{N} \left(\mathrm{RSS} + 2d\hat{\sigma}_{\varepsilon}^2 \right) \tag{3.9}$$

where RSS is an regression sum of square associated with the fitted model. Essentially, the C_p statistic adds a penalty of $2d\hat{\sigma}^2$ to the training RSS in order to adjust for the fact that the training error tends to underestimate the test error [17].

Clearly, the penalty increases as the number of predictors in the model increases; this is intended to adjust for the corresponding decrease in training RSS. As a consequence, the C_p statistic tends to take on a small value for models with a low test error, so when determining which of a set of models is best, we choose the model with the lowest C_p value [17].

3.4.2 Akaike information criterion

Akaike (1974)[34] derived a criterion from information theories, known as Akaike information criterion (AIC). The AIC is a similar but more generally applicable estimate of Err when a log-likelihood loss function is used. In the regression setting, the standard linear model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon \tag{3.10}$$

with Gaussian errors, maximum likelihood and least squares are the same thing. In this case AIC is given by [17]

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + 2d\hat{\sigma}^2\right) \tag{3.11}$$

For the logistic regression model, using the binomial log-likelihood, we have

$$\mathrm{AIC} = -\frac{2}{N} \cdot \log \mathrm{lik} + 2 \cdot \frac{d}{N} \tag{3.12}$$

For the Gaussian model (with variance $\sigma_\varepsilon^2 = \hat{\sigma}_\varepsilon^2$ assumed known), the AIC statistic is equivalent to C_p, and so we refer to them collectively as AIC. To use AIC for model selection, we simply choose the model giving smallest AIC over the set of models considered. For nonlinear and other complex models, we need to replace d by some measure of model complexity. Given a set of models $f_\alpha(x)$ indexed by a tuning parameter α, denote by $\overline{\mathrm{err}}(\alpha)$ and $d(\alpha)$ the training error and number of parameters for each

$$\mathrm{AIC}(\alpha) = \overline{\mathrm{err}}(\alpha) + 2 \cdot \frac{d(\alpha)}{N}\hat{\sigma}_\varepsilon^2 \tag{3.13}$$

The function AIC (α) provides an estimate of the test error curve, and we find the tuning parameter $\hat{\alpha}$ that minimizes it. Our final chosen model is $f_{\hat{\alpha}}(x)$[1].

3.4.3 Bayesian information criterion (BIC)

BIC is derived from a Bayesian point of view by Schwarz (1978) [35] , but looks similar to C_p (and AIC) as well. For the least squares model with d predictors, the BIC is, up to irrelevant constants, given by

$$\mathrm{BIC} = \frac{1}{n}\left(\mathrm{RSS} + \log(n)d\hat{\sigma}^2\right) \tag{3.14}$$

Like C_p, the BIC will tend to take on a small value for a model with a low test error, and so generally we select the model that has the lowest BIC value. Notice that BIC replaces the $2d\hat{\sigma}^2$ used by C_p with a $\log(n)d\hat{\sigma}^2$ term, where n is the number of observations. since

$\log n > 2$ for any $n > 7$ the BIC statistic generally places a heavier penalty on models with many variables, and hence results in the selection of smaller models than C_p [**?**].

The Bayesian information criterion (BIC), like AIC, is applicable in settings where the fitting is carried out by maximization of a log-likelihood. The generic form of BIC is [1]

$$\mathrm{BIC} = -2 \cdot \log \operatorname{lik} + (\log N) \cdot d \tag{3.15}$$

3.4.4 Adjusted R^2

The adjusted R^2 statistic is another popular approach for selecting among a set of models that contain different numbers of variables. The coefficient of determination R^2 is defined as $1-\mathrm{RSS}/\mathrm{TSS}$, where $\mathrm{TSS} = \sum (y_i - \bar{y})^2$ is the total sum of squares for the response. since RSS always decreases as more variables are added to the model, the R^2 always increases as more variables are added. For a least squares model with d variables, the adjusted R^2 statistic is calculated as [17]:

$$\text{Adjusted } R^2 = 1 - \frac{\mathrm{RSS}/(n-d-1)}{\mathrm{TSS}/(n-1)} \tag{3.16}$$

Unlike C_p, AIC, and BIC, for which a small value indicates a model with a low test error, a large value of adjusted R^2 indicates a model with a small test error. Maximizing the adjusted R^2 is equivalent to minimizing $\frac{\mathrm{RSS}}{n-d-1}$. While RSS always decreases as the number of variables in the model increases, $\frac{\mathrm{RSS}}{n-d-1}$ may increase or decrease, due to the presence of d in the denominator [17].

3.5 Cross Validation

Cross Validation is an alternative to the approaches discussed above, it is the simplest and most widely used method for estimating prediction error. This method directly estimates the expected prediction error (average test error) over all training data [1].

$$\text{Err} = \text{E}[L(Y, \hat{f}(X))] \tag{3.17}$$

This is the average generalization error when the method $\hat{f}(X)$ is applied to an independent test sample from the joint distribution of X and Y.

This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size; for example, when $K = 5$, the scenario looks like Figure 1.2.

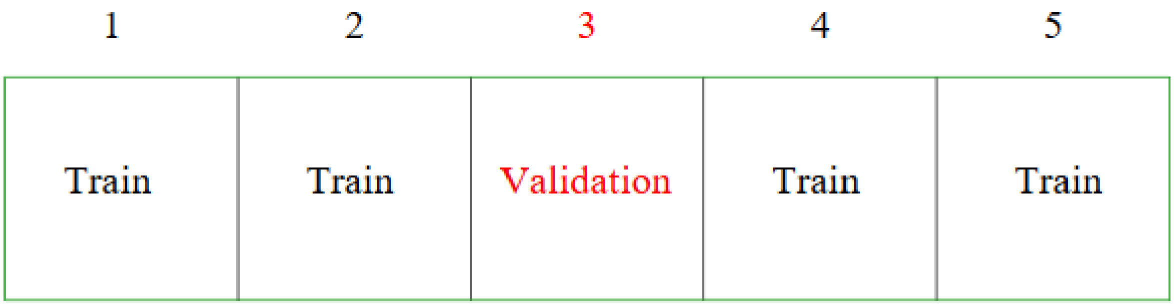


Figure 3.2: 5 fold cross validation partition.

The model is trained using $K-1$ parts of the K folds and calculate the prediction error of the fitted model with kth part of the data. This is done for $k = 1, 2, \ldots, K$ and average the K estimates of prediction error. The k-fold CV estimate of the prediction error is [1]:

$$\text{CV}(\hat{f}) = \frac{1}{K} \sum_{i=1}^{K} L\left(y_i, \hat{f}^{-k(i)}\left(x_i\right)\right) \tag{3.18}$$

where $\hat{f}^{-k}(x)$ is the model fitted with the kth part of the data removed. In practice $k = 5$ or $k = 10$ often gives lower variance and lower bias that is (provides a good compromise for) balance bias-variance tradeoff [1].

This procedure has an advantage relative to AIC, BIC, C_p, and adjusted R^2, in that it provides a direct estimate of the test error, and makes fewer assumptions about the true underlying model. It can also be used in a wider range of model selection tasks, even in cases where it is hard to pinpoint the model degrees of freedom (e.g. the number of predictors in the model) or hard to estimate the error variance σ^2[17].

3.6 Model Selection

Again the two objective for assessing the performance of a model, are Model selection which involves estimating the performance of different models in order to choose the best one and 2. Model assessment which involves estimating prediction error of the best model on new data [1].

In a data-rich situation, the best approach for both problems is to randomly divide the dataset into three parts: a training set, a validation set, and a test set. The training set is used to fit the models; the validation set is used to estimate prediction error for model selection; the test set is used for assessment of the generalization error of the final chosen model. Generally the split might be 50% for training, and 25% each for validation and testing [1].

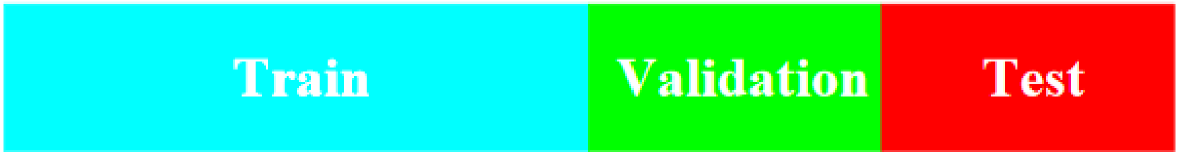


Figure 3.3: Train, validation and Test set.

In situations where there is insufficient data to split it into three parts an approximation of the validation (model selection) step can either be obtained analytically using AIC, BIC,Adjusted R^2 or cross validation as explained in sections 1.4 and 1.5. Next we discuss concept that are use in model complexity selection.

3.7 Bias, Variance and Model Complexity

3.7.1 The Bias–Variance Tradeoff

Here we explore the relation between test error and the bias-variance tradeoff. Bias and variance are statistical properties of predictive models. Bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model. This may result in poor generalization of the model on unseen data (ie a low prediction accuracy or high test error). An example is assuming simple linear relationship, when the true relation is non-linear, so no matter how many training observations we are given, it will not be possible to produce an accurate estimate using linear regression [17].

Variance refers to the amount by which $\hat{f}$ would change if we estimated it using a different training data set. Since the training data are used to fit the statistical learning method, different training data sets will result in a different $\hat{f}$. But ideally the estimate for f should not vary too much between training sets. However, if a method has high variance then small changes in the training data can result in large changes in $\hat{f}$. In general, more flexible statistical methods have higher variance [17].

3.7.2 illustration of Overfitting and Underfitting

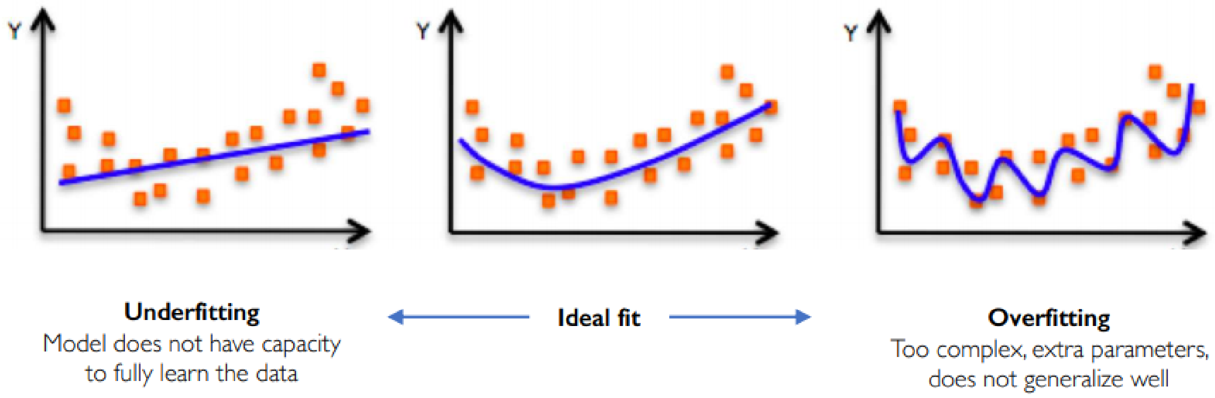


Figure 3.4: Problem of Overfitting and Underfitting Train set [27]

From Figure 1.4 least squares line on the left is too simple to capture trend in data, it has high bias and low variance, because moving any single observation will likely cause only a small shift in the position of the line. Thus underfitting the data which may not generalize well on unseen data. In contrast too flexible curve on the right follow the observations very closely. It has high variance and low bias because changing any one of these data points may cause the estimate $\hat{f}$ to change. Thus overfitting the data which may also not also generalize well. The middle figure has the right level of complexity, hence will provide a good generalization to unseen data. Thus finding the right/good level of complexity without overfitting or underfitting involves a trade off between bias and variance of a model. The fact overfitted model tends to provide prediction with a larger variance in spite of a smaller bias while an underfitted model tends to provide prediction with a smaller variance yet with a larger bias, is called the "bias-variance tradeoff" [17].

3.7.3 Empirical illustration of Bias-Variance Tradeoff

The Figure 1.1 explains the relation between bias ,variance and model complexity. Its shows the prediction error (light red curves) for 100 simulated training sets each of size 50. The lasso was used to produce the sequence of fits by increasing the model complexity from left to right. The solid red curve is the average prediction error [1].

From Figure 1.1 as the model becomes more and more complex, it uses the training data more and is able to adapt to more complicated underlying structures. Hence there is a decrease in bias but an increase in variance. The Training error consistently decreases with model complexity, typically dropping to zero if model complexity is increased enough. However, a model with zero training error is overfit to the training data and will typically generalize poorly. However, prediction error decreases as important variables are gradually added in, hits its minimum around the best model, and then starts to increase when irrelevant variables are included [1].

The graph also suggests that underfitting causes more concerns than overfitting if prediction is the primary goal. This is because the inflation amount in prediction error caused by slightly overfitting is relatively smaller than that caused by underfitting. Nevertheless, a simpler model is much easier to interpret. The goal of model selection is to find a parsimonious model that does reasonably well in prediction[1]. There are three groups of methods for this task, which are discussed in order.

3.7.4 The Bias-Variance Decomposition

Suppose we assume that $Y = f(X) + \varepsilon$ where $\mathrm{E}(\varepsilon) = 0$ and $\mathrm{Var}(\varepsilon) = \sigma_\varepsilon^2$, we can decomposed the expected prediction error of a regression fit $\hat{f}(X)$ at an input point $X = x_0$, into the sum of three fundamental quantities using squared-error loss [1]:

$$\begin{aligned}
\operatorname{Err}\left(x_0\right) &= E\left[\left(Y-\hat{f}\left(x_0\right)\right)^2 \mid X=x_0\right] \\
&= \sigma_\varepsilon^2+\left[\mathrm{E} \hat{f}\left(x_0\right)-f\left(x_0\right)\right]^2+E\left[\hat{f}\left(x_0\right)-\mathrm{E} \hat{f}\left(x_0\right)\right]^2 \\
&= \sigma_\varepsilon^2+\operatorname{Bias}^2\left(\hat{f}\left(x_0\right)\right)+\operatorname{Var}\left(\hat{f}\left(x_0\right)\right) \\
&= \text{ Irreducible Error } + \text{Bias}^2 + \text{ Variance.}
\end{aligned} \tag{3.19}$$

The first term is the variance of the target around its true mean $f\left(x_0\right)$, and cannot be avoided no matter how well we estimate $f\left(x_0\right)$, unless $\sigma_\varepsilon^2 = 0$. The second term is the squared bias, the amount by which the average of our estimate differs from the true mean; the last term is the variance; the expected squared deviation of $\hat{f}\left(x_0\right)$ around its mean. Typically the more complex we make the model $\hat{f}$, the lower the (squared) bias but the higher the variance [1].

Equation 3.19 tells us that in order to minimize the expected test error we need to select a statistical learning method that simultaneously achieves low variance and low bias.

3.8 Model Selection Methods

3.8.1 Cross-validation for model selection

Given a set of models $f(x, \alpha)$ indexed by a tuning parameter α, that determines the model complexity denote by $\hat{f}^{-k}(x, \alpha)$ the αth model fit with the kth part of the data removed. Then for this set of models we define

$$\mathrm{CV}(\hat{f}, \alpha) = \frac{1}{K} \sum_{i=1}^{K} L\left(y_i, \hat{f}^{-\kappa(i)}\left(x_i, \alpha\right)\right) \tag{3.20}$$

The function $\mathrm{CV}(\hat{f}, \alpha)$ provides an estimate of the test error curve, and we find the tuning parameter $\hat{\alpha}$ that minimizes it. The selected model is $f(x, \hat{\alpha})$, which is then fitted to all the data [1].

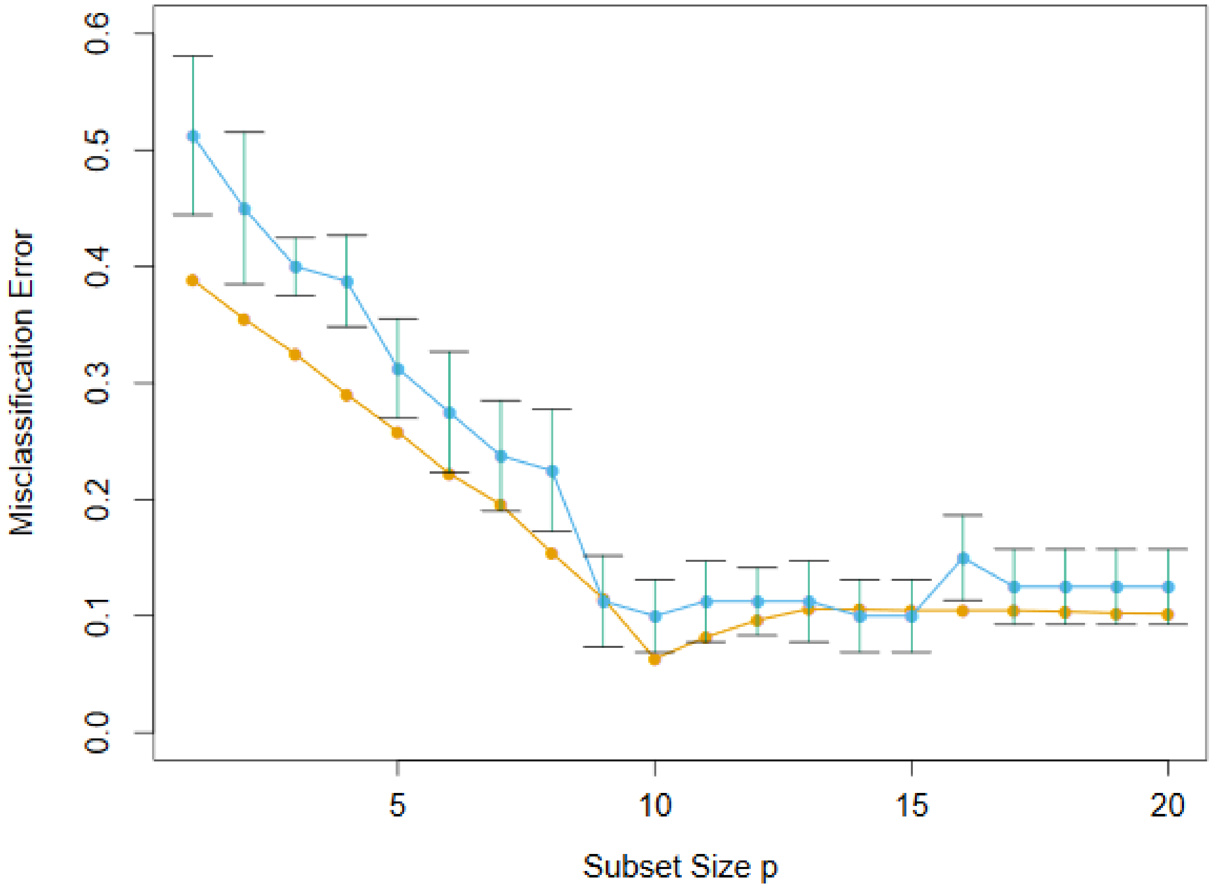


Figure 3.5: Prediction error (orange) and tenfold cross-validation curve(blue) estimated from a single training set.

One standard error rule is used in practice choose the most parsimonious model whose error is no more than one standard error above the error of the best model[1].

Cross-Validation for Classification Problems

In the classification setting, cross-validation works just as described above, except that misclassification error is used to quantify test error, instead mean squared error. The data is divided into K roughly equal-sized parts $C_1, C_2, \dots C_K$. Where C_k denotes the indices of the observations in part k. There are n_k observations in part k : if n is a multiple of K, then $n_k = n/K$. The k-fold CV estimate of the prediction error rate is given by[17]:

$$\mathrm{CV}_K = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{Err}_k \tag{3.21}$$

where $\mathrm{Err}_k = \sum_{i \in C_k} I\left(y_i \neq \hat{y}_i\right)/n_k$ the missclassification rate of the kth fold.

3.8.2 Subset Selection

In this section we consider some methods for selecting subsets of predictors. These include best subset and stepwise model selection procedure.

Best subset selection

The best subset selection is performed by fitting a separate least squares regression for each possible combination of the p predictors. That is all p models are fitted that contain exactly one predictor, all $\begin{pmatrix} p \\ 2 \end{pmatrix} = p(p-1)/2$ models that contain exactly two predictors, and so forth. Then the best model is identify from all the resulting models. The algorithm for the best subset selection is described below [1].

Algorithm for best subset selection

Below is the algorithm for the best subset selection :

1. For k $= 0, 1, \dots$ p

    (a) Fit all ${}_{\mathrm{p}}\mathrm{C}_{\mathrm{k}}$ models that contain exactly k predictors (Mo denote the null model, which contains no predictors)
    (b) Pick the best among these ${}_{P}\mathrm{C}_{\mathrm{k}}$ models, and call it M_{k}. The best model is defined as having the largest R^2

2. Select a single best model from among $\mathrm{M}_0, \ldots, \mathrm{M}_{\mathrm{p}}$ using cross-validated prediction error, C_{p}(AlC) BIC, or adjusted R^2

3.8.3 Stepwise selection

For computational reasons, best subset selection cannot be applied with very large p. An enormous search space can lead to overfitting and high variance of the coefficient estimates. For these reasons, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection [1].

Forward stepwise selection

Forward stepwise selection begins with the null model, and then adds prediotors to the model, one-at-a-time, until all the predictors are in the model. In particular, at each step the variable that gives the greatest additional improvement to the fit is added to the model. The algorithm for the forward stepwise selection is described below [1].

Algorithm for the forward stepwise selection:

1. M_0 denote the null model
2. For $\mathrm{k} = 0, \ldots, \mathrm{p} - 1$
    (a) Consider all $p-k$ models that augment the predictors in M_k with one additional predictor.
    (b) Choose the best among these $p-k$ models, and call it M_{k+1}. Here best is defined as having highest R^2

3. Select a single best model from among $\mathrm{M_0, \ldots, M_p}$ using cross-validated prediction error, $\mathrm{C_p(AlC), BIC}$, or adjusted $\mathrm{R^2}$

Backward stepwise selection

Backward stepwise selection begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time. The algorithm for the backward stepwise selection is described below [1].

Algorithm for the Backward stepwise selection

1. Let M_p denote the full model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$
    (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
    (b) Choose the best among these k models, and call it $\mathcal{M}_{k-1}$ Here best is defined as having smallest RSS or highest R^2

3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, $C_p(\text{AIC}), \text{BIC}$, or adjusted R^2

Comparing Forward and Backward selection

Backward stepwise selection requires that the number of samples n is larger than the number of variables p (so that the full model can be fit). In contrast, forward stepwise can be used even when $n < p$, and so is the only viable subset method when p is very large [1].

The subset selection methods described above involve using least squares to fit a linear model that contains a subset of the predictors. By retaining a subset of the predictors and discarding the rest, subset selection produces a model that is interpretable and has possibly lower prediction error than the full model from a regular least squares fit [17].

3.8.4 Regularization or Shrinkage Methods

The subset selection methods describe above, because of their discrete process (variables are either retained or discarded), it often exhibits high variance, and so doesn't reduce the prediction error of the full model. As an alternative, we can fit a model containing all p predictors using a technique that constrains or regularizes the coefficient estimates, or equivalently, that shrinks the coefficient estimates towards zero. Shrinkage methods are more continuous, and don't suffer as much from high variability. The two best-known techniques for shrinking the regression coefficients towards zero are ridge regression and the lasso[1].

Ridge regression

Ridge regression uses L_2 regularization to shrinks the regression coefficients by imposing a penalty on their size. The ridge coefficients minimize a penalized residual sum of squares[1]:

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{3.22}$$

where the complexity parameter $\lambda \geq 0$ controls the amount of shrinkage, the larger the value of λ, the greater the amount of shrinkage. The coefficients are shrunk toward zero. An equivalent way to write the ridge problem is[1]:

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \tag{3.23}$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 \leq t$$

which makes explicit the size constraint on the parameters. The ridge solutions are not equivariant under scaling of the inputs, and so the features are standardizes before solving (3.22). In addition, no penalty for β_0, it is estimated by the mean of $\mathbf{y}$. The matrix form of criterion in (3.22)is [1]:,

$$\mathrm{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta \tag{3.24}$$

with solution solutions given by:

$$\hat{\beta}^{\text{ridge}} = \left(\mathrm{X}^T\mathrm{X} + \lambda \mathrm{I}\right)^{-1}\mathrm{X}^T\mathrm{y} \tag{3.25}$$

where I is the $p \times p$ identity matrix. The larger $\lambda \geq 0$ is, the more penalty on the size of β. For $\lambda > 0$, the ridge estimate of β is shrunk towards zero. If $\lambda \approx \infty$, all coefficients except β_0 are zero. No penalty on β if $\lambda = 0$. In the case, the ridge estimate equivalent to the least square estimate[1].

Choosing a Tuning parameter λ

The turning or complexity parameter λ is determined in practice using bias-variance tradeoff of the model and cross-validation in the case of no validation set. The bias increases as λ (amount of shrinkage) increases. The variance decreases as λ increases.

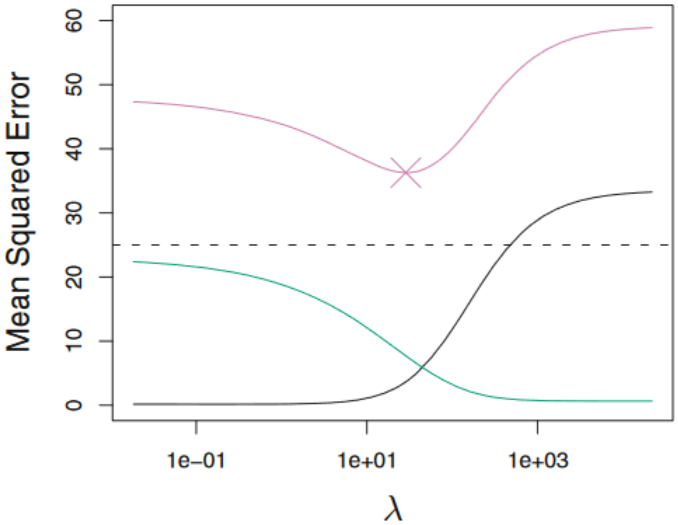


Figure 3.6: Simulated data: $n = 50$, $p = 45$, all having nonzero coefficients.The bias increases as λ increases and variance decreases as λ increases

Figure 3.6 is a simulated data with $n = 50$ observations, $p = 45$ predictors, all having nonzero coefficients. Squared bias (black), variance (green), and test mean squared error (purple) for the ridge regression predictions on a simulated data set, as a function of λ. The horizontal dashed lines indicate the minimum possible MSE. The purple crosses indicate the ridge regression models for which the MSE is smallest[17].

As λ increases, the complexity of the ridge regression fit decreases, leading to decreased variance but increased bias. This is illustrated in Figure 3.6. The test mean squared error (MSE), plotted in purple, is a function of the variance plus the squared bias. From Figure 3.6, for values of λ up to about 10, the variance decreases rapidly, with very little increase in bias, plotted in black. Consequently, the MSE drops considerably as λ increases from 0 to 10. Beyond this point, the decrease in variance due to increasing λ slows, and the shrinkage on the coefficients causes them to be significantly underestimated, resulting in a large increase in the bias. The minimum MSE is achieved at approximately $\lambda = 30$.[17].

Lasso Regression

The lasso is a shrinkage method like ridge, but uses uses L_1 regularization to penalize regression coefficients size. As with ridge regression, the lasso shrinks the coefficient estimates towards zero. However, in the case of the lasso, the L_1 penalty has the effect of forcing some of the coefficient estimates to be exactly zero when the tuning parameter λ is sufficiently 1arge. The lasso estimate is defined by [1]:

$$\begin{aligned}\hat{\beta}^{\text{lasso}} &= \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \\ &\text{subject to } \sum_{n=1}^{n} |\beta_j| \le t\end{aligned} \tag{3.26}$$

The Lagrangian form of the lasso problem is:

$$\hat{\beta}^{\text{lasso}} = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{3.27}$$

Compared to the ridge regression problem (3.24) or (3.25). The L_2 ridge penalty $\sum_1^p \beta_j^2$ is replaced by the L_1 lasso penalty $\sum_1^p |\beta_j|$. This latter constraint makes the solutions nonlinear in the y_i, and there is no closed form expression as in ridge regression. The lasso solution is a quadratic programming problem, although efficient algorithms are available for computing the entire path of solutions as λ is varied, with the same computational cost as for ridge regression[1].

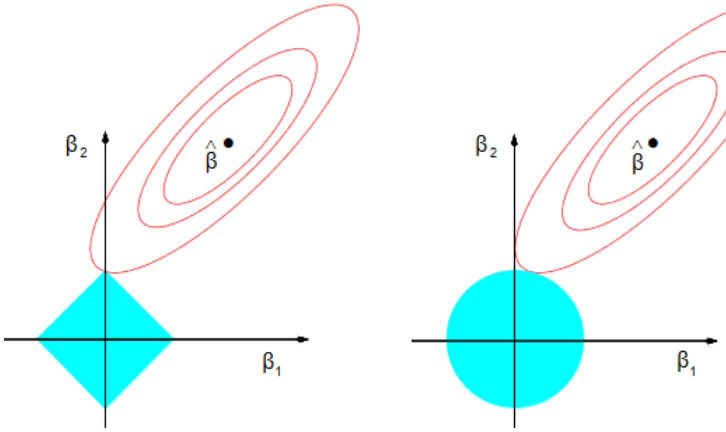


Figure 3.7: Lasso (left) and Ridge regression (right) contours of the error and constraint functions.

Figure 1.7 shows the lasso (left) and ridge regression (right) for two parameters. The residual sum of squares has elliptical contours, centered at the full least squares estimate. The constraint region for ridge regression is the disk $\beta_1^2 + \beta_2^2 \leq t$, while that for lasso is the diamond $|\beta_1| + |\beta_2| \leq t$. Both methods find the first point where the elliptical contours hit the constraint region[1].

The turning or complexity parameter λ is determined in practice using bias-variance tradeoff of the model and cross-validation in the case of no validation set[1].

Comparing Ridge and Lasso Regression

Lasso has a big advantage with respect to interpretation. It performs variable selection in the linear model. But neither the ridge regression nor the lasso universally dominate the other in terms of the prediction error. In general, one might expect the lasso to perform better when the response is a function of only a relatively small number of predictors[1].

Elastic net = lasso + ridge

For data with high degree of collinearity in the features, the lasso penalty is indifferent to the choice among a set of strong but correlated variables and the ridge penalty, on the other hand, tends to shrink the coefficients of correlated variables toward each other. Also, if there is a group of highly correlated variables, then the lasso tends to select one variable from a group and ignore the others. The elastic net regression includes lasso (L_1) and ridge (L_2) penalties (Zou and Hastie,2005)[36], and has the form

$$\sum_{j=1}^{p}\left(\alpha\left|\beta_j\right|+(1-\alpha)\beta_j^2\right) \tag{3.28}$$

The second term encourages highly correlated features to be averaged, while the first term encourages a sparse solution in the coefficients of these averaged features. The elastic net penalty can be used with any linear model, in particular for regression or classification. The optimization problem for linear regression is given by [1]:

$$\hat{\beta}^{\text{e-net}} = \min_{\beta}\|\mathbf{y}-\mathbf{X}\beta\|^2+\lambda\left[\alpha\|\beta\|_2^2+(1-\alpha)\|\beta\|_1\right] \tag{3.29}$$

$$= \text{argmin}_{\beta}(\text{y}-\text{X}\beta)'(\text{y}-\text{X}\beta)+\lambda\left[(1-\alpha)\|\beta\|_1+\alpha\|\beta\|_2^2\right] \tag{3.30}$$

$$= \underset{\beta}{\text{argmin}}\left\{\frac{1}{2}\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{p}x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^{p}\left(\alpha\left|\beta_j\right|+(1-\alpha)\beta_j^2\right)\right\} \tag{3.31}$$

where $\lambda \geq 0, 0 \leq \alpha \leq 1$

In classification, a multinomial logistic regression with elastic-net penalty becomes:

$$\max_{\{\beta_{0k}, \beta_k \in R^p\}_1^K} \left[\sum_{i=1}^{N} \log \Pr\left(g_i | x_i\right) - \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} \left(\alpha \left|\beta_{kj}\right| + (1-\alpha)\beta_{kj}^2\right) \right] \tag{3.32}$$

The parameter α determines the mix of the penalties and both α and λ chosen by cross-validation.

3.8.5 Dimension Reduction Methods

The methods that we have discussed so far in this section have involved fitting linear regression models, using least squares or a shrunken approach, with the original predictors, $X_1, X_2, \ldots, X_p$. Next we explore a class of approaches that transform the predictors and then fit a least squares model using the transformed variables. The idea behind the method is that in many situations large number of inputs are correlated. These methods find a small number of linear combinations $Z_m, m = 1, \ldots, M$ of the original inputs X_j, and the Z_m are then used in place of the X_j as inputs in the regression. The methods differ in how the linear combinations Z_m are constructed[1].

The $Z_1, Z_2, \ldots, Z_M$ represent $M < p$ linear combinations of our original p predictors. That is

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j \tag{3.33}$$

for some constants $\phi_{1m}, \phi_{2m} \ldots, \phi_{pm}, m = 1, \ldots, M$. We can then fit the linear regression model,

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i, \quad i = 1, \ldots, n \tag{3.34}$$

using least squares. The regression coefficients are given by $\theta_0, \theta_1, \ldots, \theta_M$ in (1.34). If the constants $\phi_{m1}, \ldots, \phi_{mp}$ are chosen wisely, then such dimension reduction approaches can often outperform least squares regression [1].

Principal components regression (PCR)

The linear combinations Z_m used are obtained using principal components analysis discussed under section 2.3 . Principal component regression forms the derived input columns $\mathbf{Z}_m = \mathbf{X}v_m$, using SVD on $\mathbf{X} = \mathbf{UDV}'$ where $\mathbf{v}_{\mathrm{m}}$ is mth column of $\mathbf{V}$ (principal component direction), and $\mathbf{u}_{\mathrm{m}}$ is *mth* column of $\mathbf{U}$ (normalized principal component). The response $\mathbf{y}$ is then regressed on $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_M$ for some $M \leq p$. Since the $\mathbf{Z}_m$ are orthogonal, this regression is just a sum of univariate regressions[1]:

$$\hat{\mathbf{y}}^{\mathrm{pcr}}_{(M)} = \bar{y}\mathbf{1} + \sum_{m=1}^{M} \hat{\theta}_m \mathbf{z}_m \tag{3.35}$$

where $\hat{\theta}_m$ is PCR regression coefficient. Since $\mathbf{Z}_m$ are each linear combinations of the original $\mathbf{x}_j$, we can express the solution (3.35) in terms of coefficients of the $\mathbf{x}_j$ [1]:

$$\hat{\beta}^{\mathrm{pcr}}(M) = \sum_{m=1}^{M} \hat{\theta}_m v_m \tag{3.36}$$

The first principal component is a normalized linear combination of the variables with the largest variance. The second principal component has largest variance, subject to being uncorrelated with the first component. Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation. The first few principal components can be thought as the low-dimensional approximation of the feature matrix[1].

Draw back of PCR

PCR identifies linear combinations, or directions, that best represent the predictors $X_1, \ldots, X_p$. These directions are identified in an unsupervised way, since the response Y is not used to help determine the principal component directions. That is, the response does not supervise the identification of the principal components. Consequently, PCR suffers from a potentially serious drawback, there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response[1].

Partial least squares (PLS)

This method also constructs a set of linear combinations of the inputs for regression, but unlike principal components regression it uses Y in addition to $\mathbf{X}$ for this construction. That is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response. The PLS approach attempts to find directions that help explain both the response and the predictors [1]. The algorithm for the PLS is described below:

Algorithm for Partial least squares

1. Standardize each $\mathbf{x}_j$ to have mean zero and variance one. Set $\hat{\mathbf{y}}^{(0)} = \bar{y}1$, and $\mathbf{x}_j^{(0)} = \mathbf{x}_j, j = 1, \ldots, p$
2. For $m = 1, 2, \ldots, p$
   - (a) $\mathbf{z}_m = \sum_{j=1}^{p} \hat{\varphi}_{mj}\mathbf{x}_j^{(m-1)}$, where $\hat{\varphi}_{mj} = \left\langle \mathbf{x}_j^{(m-1)}, \mathbf{y} \right\rangle$
   - (b) $\hat{\theta}_m = \langle \mathbf{z}_m, \mathbf{y} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle$
   - (c) $\hat{\mathbf{y}}^{(m)} = \hat{\mathbf{y}}^{(m-1)} + \hat{\theta}_m \mathbf{z}_m$
   - (d) Orthogonalize each $\mathbf{x}_j^{(m-1)}$ with respect to $\mathbf{z}_m : \mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - \left[\left\langle \mathbf{z}_m, \mathbf{x}_i^{(m-1)} \right\rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle \right] \mathbf{z}_m, j = 1, 2, \ldots, p$
3. Output the sequence of fitted vectors $\left\{ \hat{\mathbf{y}}^{(m)} \right\}_1^p$. since the $\{\mathbf{z}_\ell\}_1^m$ are linear in the original $\mathbf{x}_j$, so is $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{\text{pls}}(m)$.

Chapter 4

Data Analysis and Result.

4.1 Analysis on Breast Cancer Data

4.1.1 Data background

The analysis that follows is focused on the breast cancer [49] data sets, that are available in UCI machine learning repository. The breast cancer data sets has 699 observations with 11 variables. Table 4.1 gives a description of the breast cancer data set.

Table 4.1: Name of variables and description

Variable	**Description**
Id	Sample code number
Cl.thickness	Clump Thickness
Cell.size	Uniformity of Cell Size
Cell.shape	Uniformity of Cell Shape
Marg.adhesion	Marginal Adhesion
Epith.c.size	Single Epithelial Cell Size
Bare.nuclei	Bare Nuclei
Bl.cromatin	Bland Chromatin
Normal.nucleoli	Normal Nucleoli
Mitoses	Mitoses
Class	Class

The 'Class' column is the response variable that includes the status of a tumor as malignant (breast cancer) or benign (not breast cancer). Our objective is to predict the "Class" variable and to conclude whether a patient's tumor is malignant or benign.

In the next section, we performed some exploratory data analysis by studying the correlation, association and cluster of all variables in the data sets. This aids in understanding the relation between variables, gain insight in the underlying structure of the data and find patterns before modeling. It also help to uncover a parsimonious model, one which explains the data with a minimum number of predictor variables.

4.1.2 Association of Data

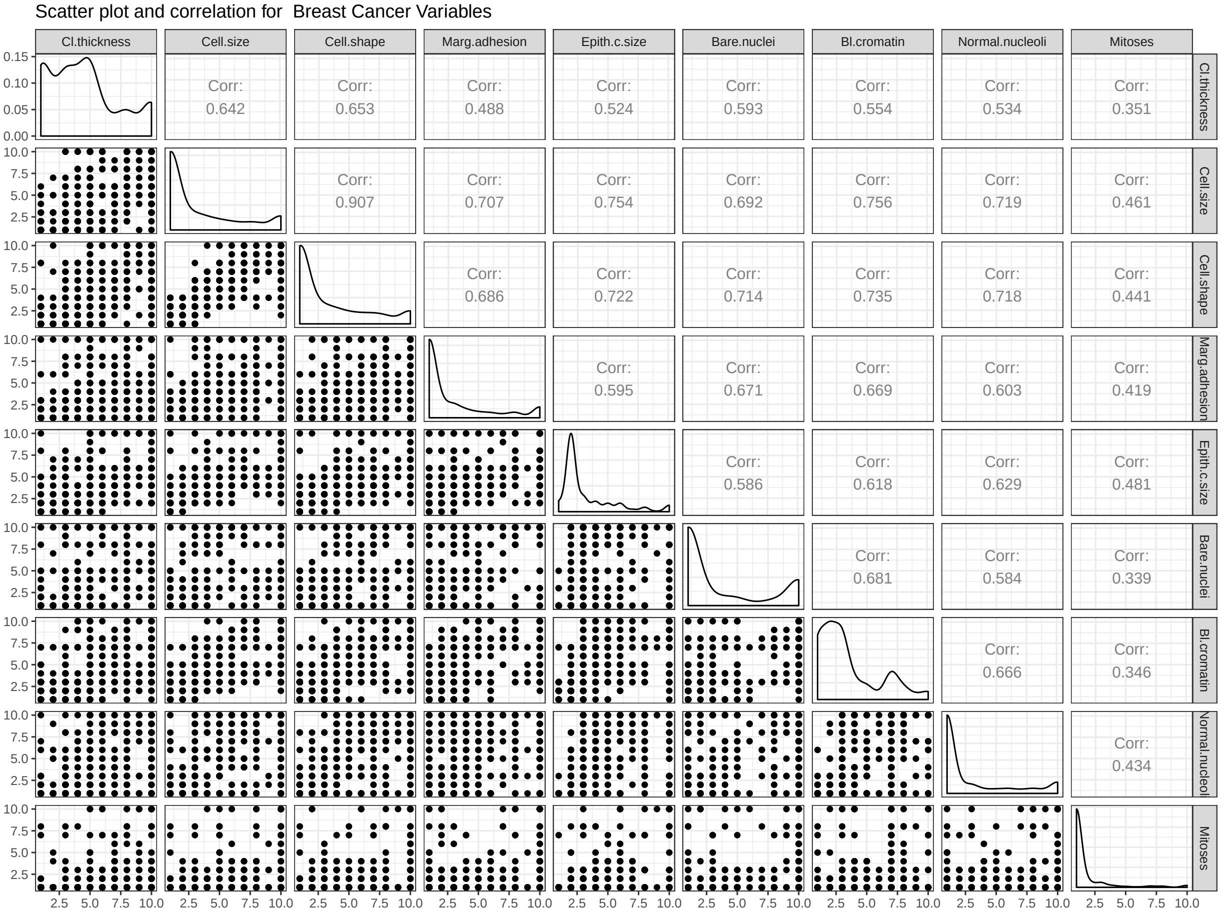


Figure 4.1: Scatter plot and Correlation for Breast Cancer Variables

Figure 4.1, shows the individual variables distribution, scatter plot and correlation between the variables. All the variable are right skewed (the diagonal) with Mitoses showing some potential outliers (left margin). Most of the variable shows strong positive association from the scatter plot with correlation greater the 0.5 except Mitoses. This may result in multicollinearity. The correlation between cells size and cell shape,cells size and marg adversion and cells size and normal nuclei are 0.907, 0.707 and 0.719 respectively.

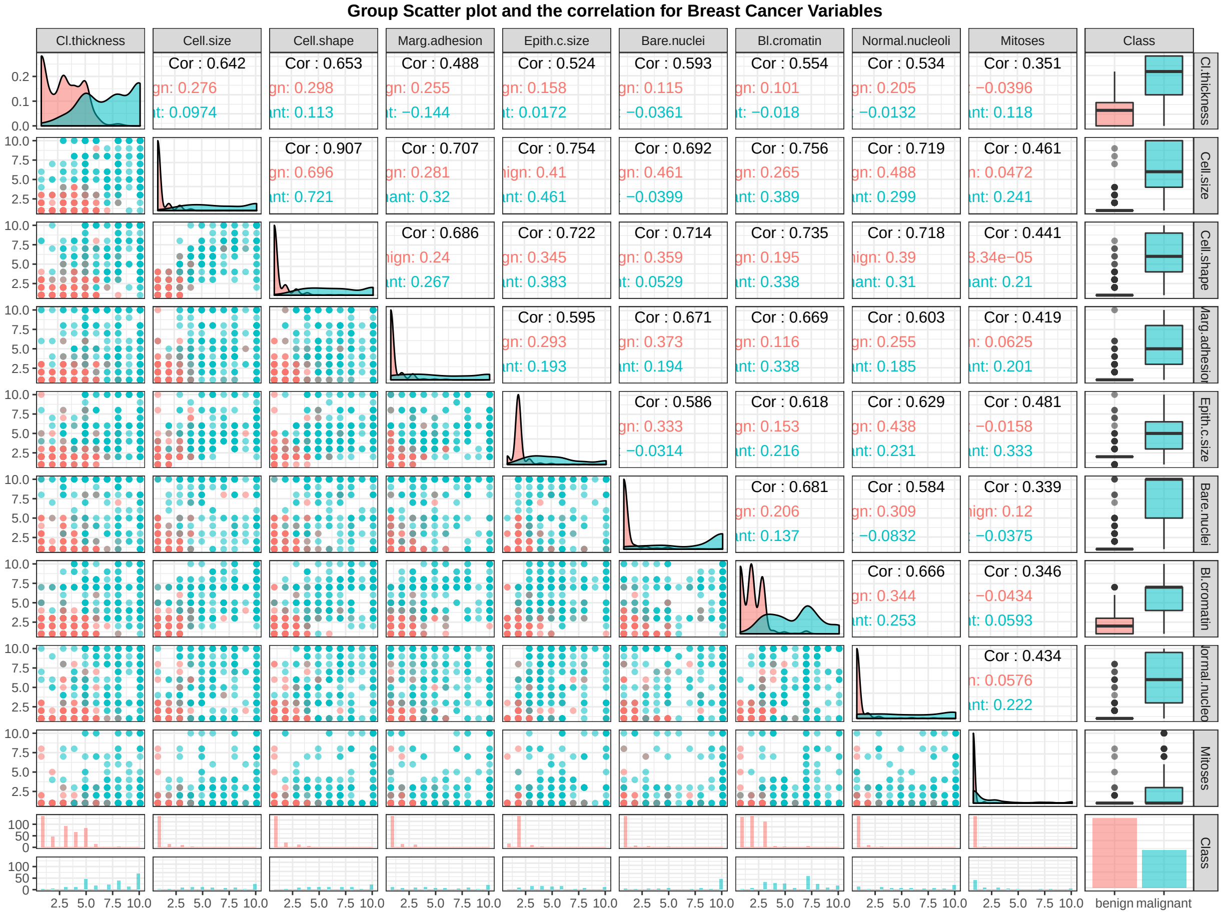


Figure 4.2: Scatter plot and Correlation between Variables by response

Figure 4.2 shows the group distribution, scatter plot and correlation between the variables. The density curve (diagomal) shows that the variables distribution of the Benign group turn to be right skewed compared to the Malignant group which is mostly symmetric. The boxplot (left margin) shows that the Benign group variable has a lot of outliers compared to the Malignant group which show outlier only in the Mitoses variable. Generally the scatter plot shows that the malign group turn to have high values in both coordinates (above 5 unit) while Benign has low values (below 5 unit) both coordinates except few outliers. That is tumor with large Clump thickness, cell size,and cell shape turned to be Malignant.

4.1.3 Principal Component Analysis (PCA)

In this section we use Principal Component analysis to investigate the relationship between variable and observations to get insight of the data. Also PCA is used to reduce the nine correlate feature to three decorrelated feature which will later be used for Principal component regression (PCR).

PCA Summary

Figure 4.3 is the scree plot which explain the the percentage on information explained (retained) by the each of the principal components. From the figure 66% , 8.6% and 6%of the information are retained in the first, second and third component respectively. The cumulative percentage of the first three component is 80.1%; meaning 80.1% of the information is retained this is shown is Table 4.2. This is an acceptably large percentage [1] hence we used these three component for our PCR.

Table 4.2: % of variation explained by each Component

STATISTICS	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
Std deviation	2.43	0.88	0.73	0.68	0.62	0.55	0.54	0.51	0.30
Prop of Variance	0.66	0.09	0.06	0.05	0.04	0.03	0.03	0.03	0.01
Cum Proportion	0.66	0.74	0.80	0.85	0.89	0.93	0.96	0.99	1.00

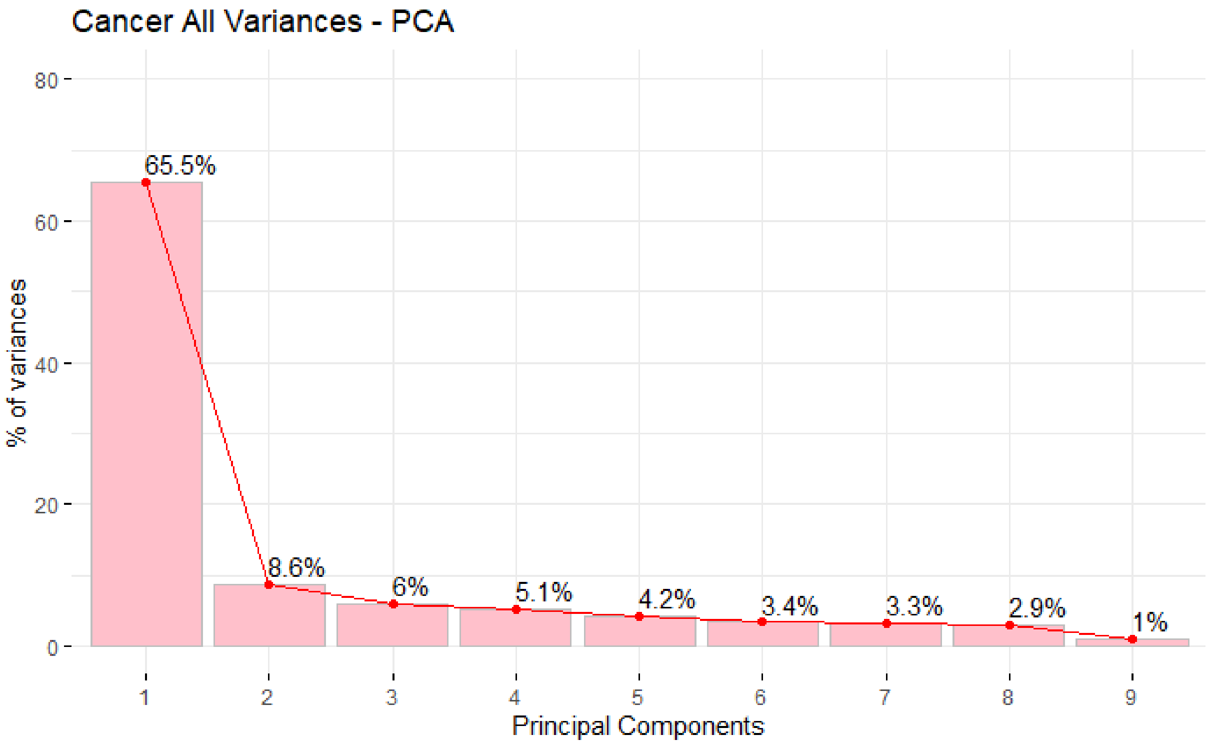


Figure 4.3: % of variation explained by each Component

Figure 4.4 shows the correlation or contribution of each feature to the dimensions of the principal component. It highlight the most contributing variables for each components. It can be seen that most of the features contribute highly to the first component and only Mitoses and CI.thickness contribute to the second and third components. An alternative variable contribution plot is shown in Figure 4.5.

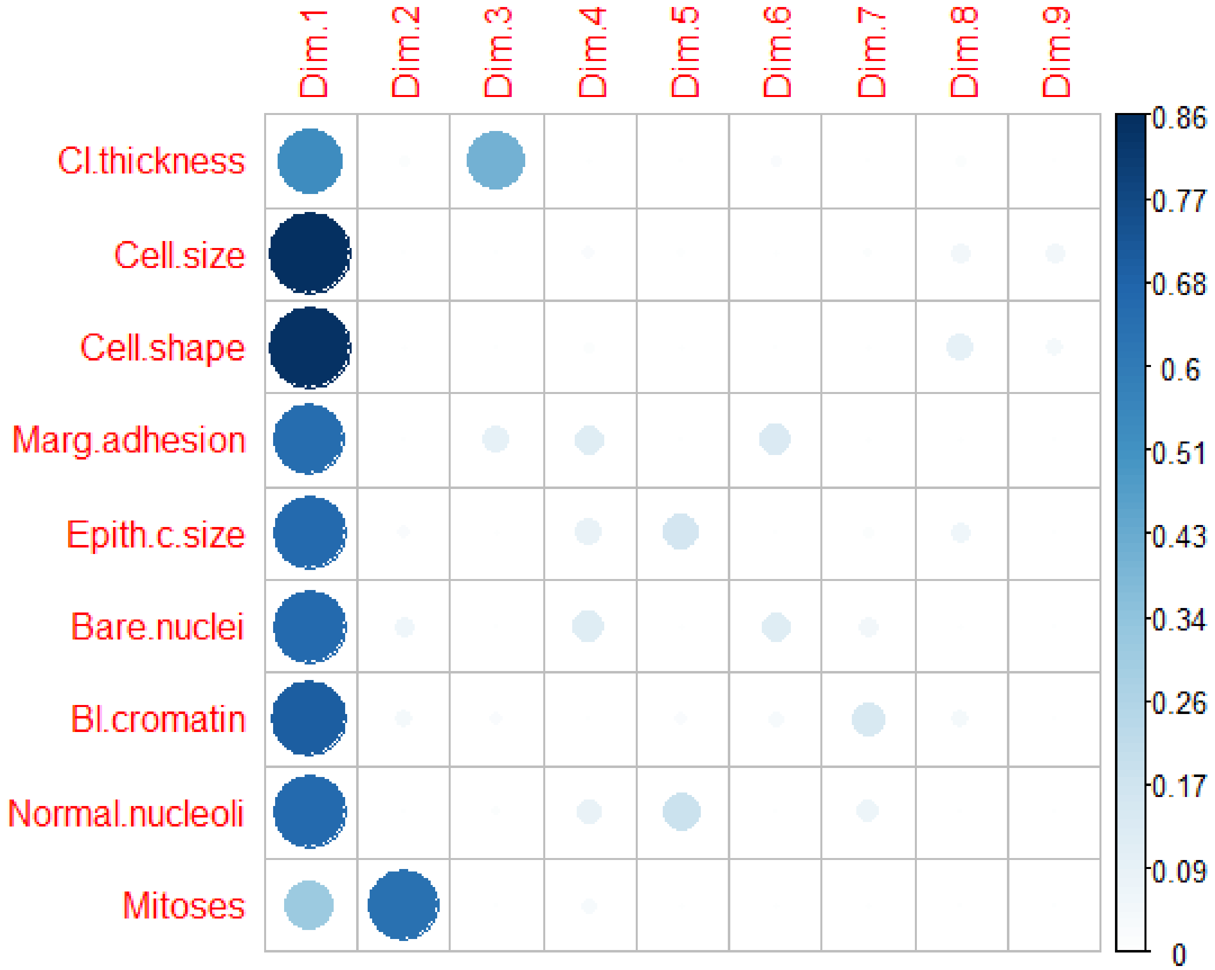


Figure 4.4: Correlation between variables and PCA.

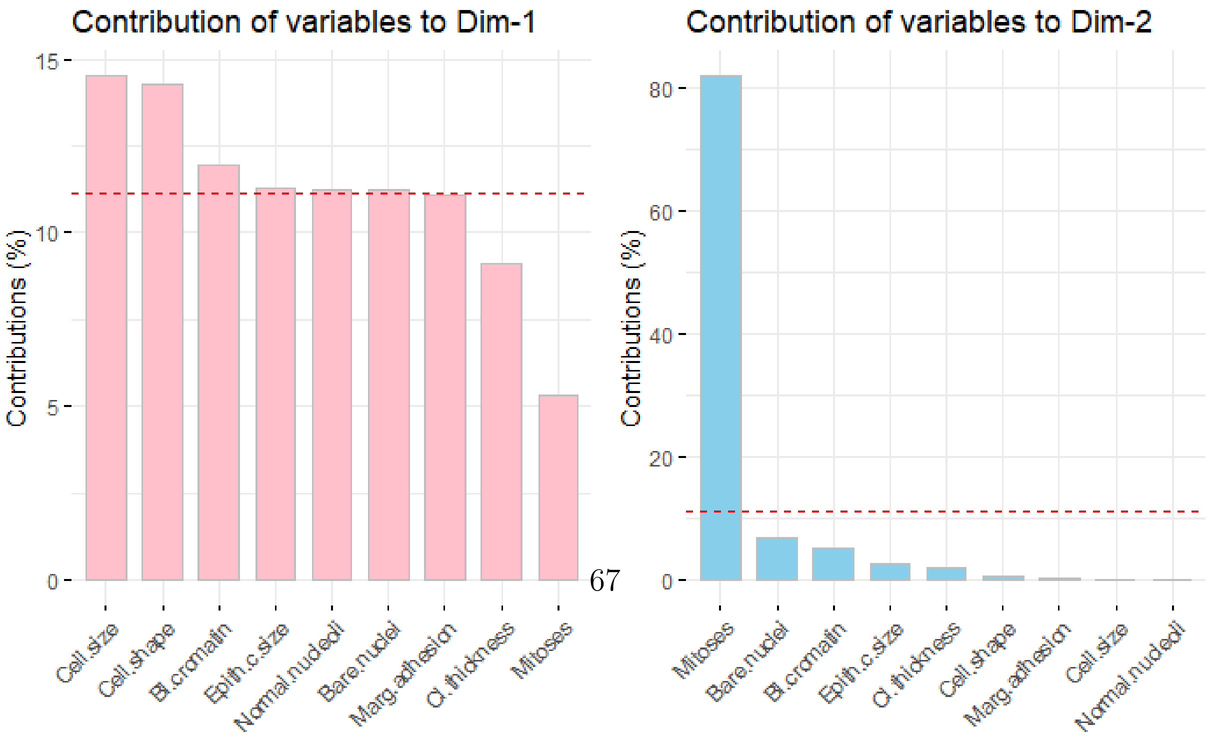


Figure 4.6 is the variable PCA plot which shows correlation between variables and the general meaning of the dimension of the components. Positive correlated variables point to the same side of the plot while negative correlated variables point to opposite sides of the graph. From the figure 4.6 we can see that all the variable are positively correlated and in the same but negative direction on PC1. About half of the variable are on the positive and half on the negative side of PC2 except Mitoses which appear very different. This figure consistent with correlation matrix plot in Figure 4.1. The interpretation we assign to PC1 is the average cell size and PC2 the rate of cell growth or division.

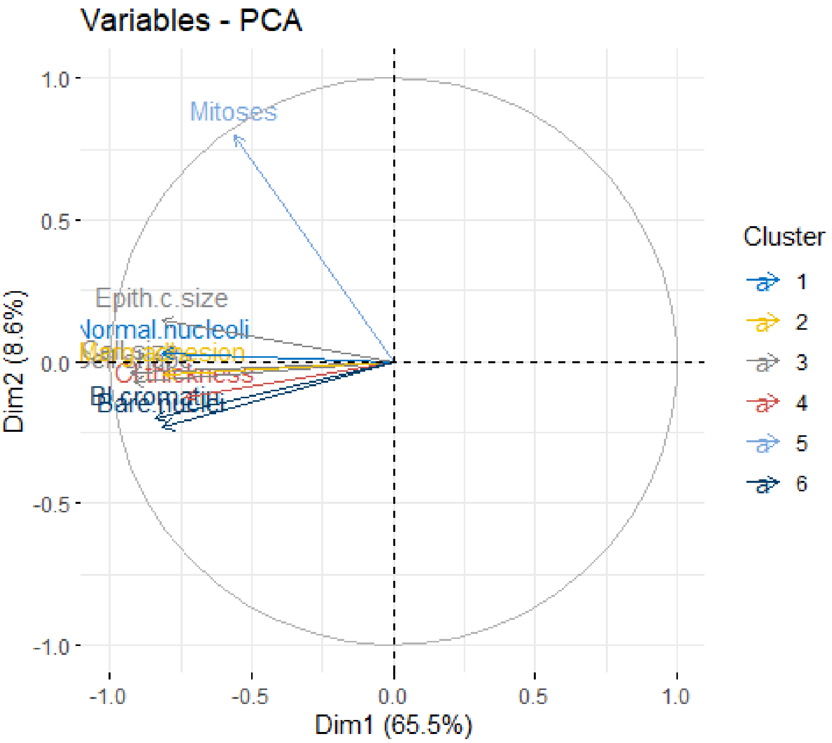


Figure 4.6: Breast cancer variables PC

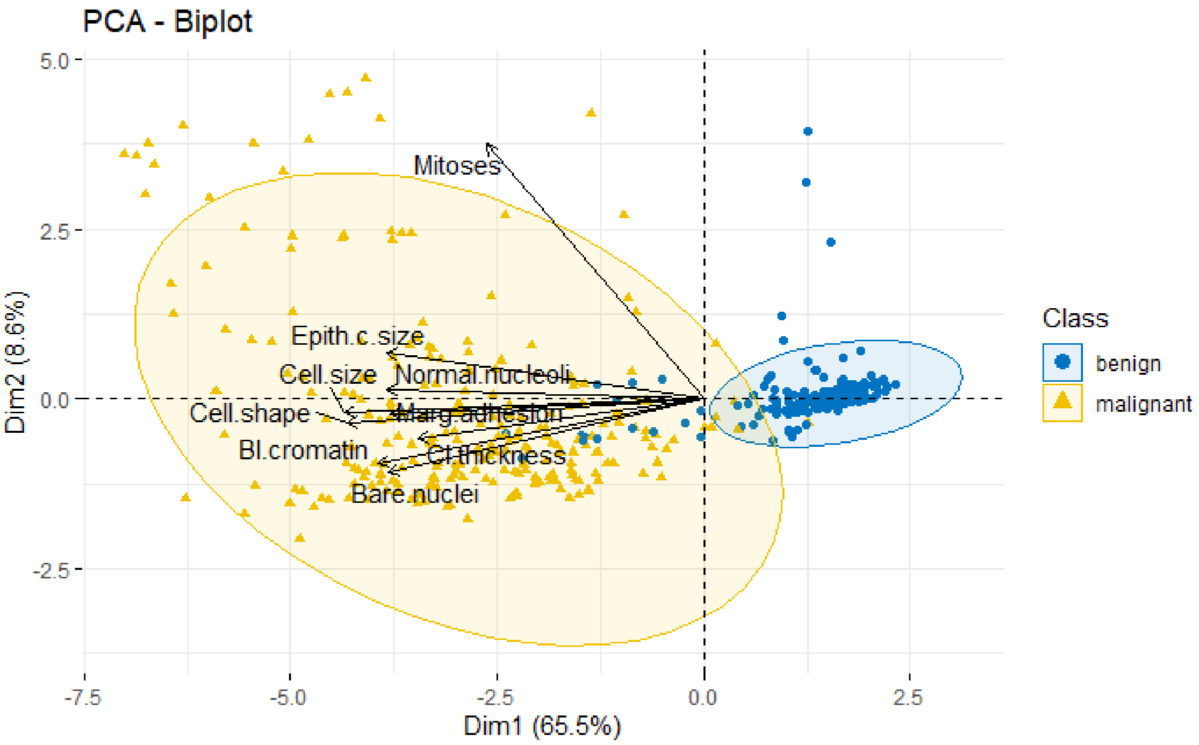


Figure 4.7: Breast Cancer PCA Bipot

Figure 4.7 is a biplot which combines both the features and the individual on the PC1 and PC2 plot with response of each individual in blue (bengin) or yellow (malignant) colored. The plot shows individuals with similar profile group together. From the figure 4.7 we can conclude that individuals with large (average) cell size and fast cell growth turn to be more cancerous(left in yellow) than those with relatively small cell size and slow growth rate.

4.2 Application of Statistical learning methods

In this section we apply 10 Machine Learning models to the breast cancer data sets, compare the predictive performance and interpretability with each other and chose best fits. We randomly divided the data into two part, 70% for building or training the model(train set) and 30% for evaluating the performance of the models(test set). We then compute the prediction accuracy and missclassification rate (MCR) to evaluate the models performance. R statistical programs was used for the data analyses.

4.2.1 Logistic regression (LR):

The logistic regression with L_1 (lasso) and L_2 (ridge) regularization are the first models we applied to build a predictive model. The model is given by:

$$P(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}} \tag{4.1}$$

where the coefficient $\beta's$ are estimate by :

$$\hat{\beta}^{\text{lasso}} = \max_{\beta}\left\{\sum_{i=1}^{N}\left[y_i\left(\beta^T x_i\right) - \log\left(1+e^{\beta^T x_i}\right)\right] - \lambda\sum_{j=1}^{p}|\beta_j|\right\} \tag{4.2}$$

for the Lasso Logistic regression and

$$\hat{\beta}^{\text{ridge}} = \max_{\beta}\left\{\sum_{i=1}^{N}\left[y_i\left(\beta^T x_i\right) - \log\left(1+e^{\beta^T x_i}\right)\right] - \lambda\sum_{j=1}^{p}\beta_j^2\right\} \tag{4.3}$$

for the Ridge Logistic regression. The complexity parameter λ is obtained with cross-validation. The L_1 penalty is also used for variable selection. Given the estimated parameter and the feature vector we predict class probability using (4.1).

Figure 4.8 shows all the coefficients of the lasso model, with each curve corresponds to a variable. It shows the path of the whole coefficient vector at as λ varies. The axis above indicates the number of nonzero coefficients at the current λ, which is the effective degrees of freedom (df) for the lasso.

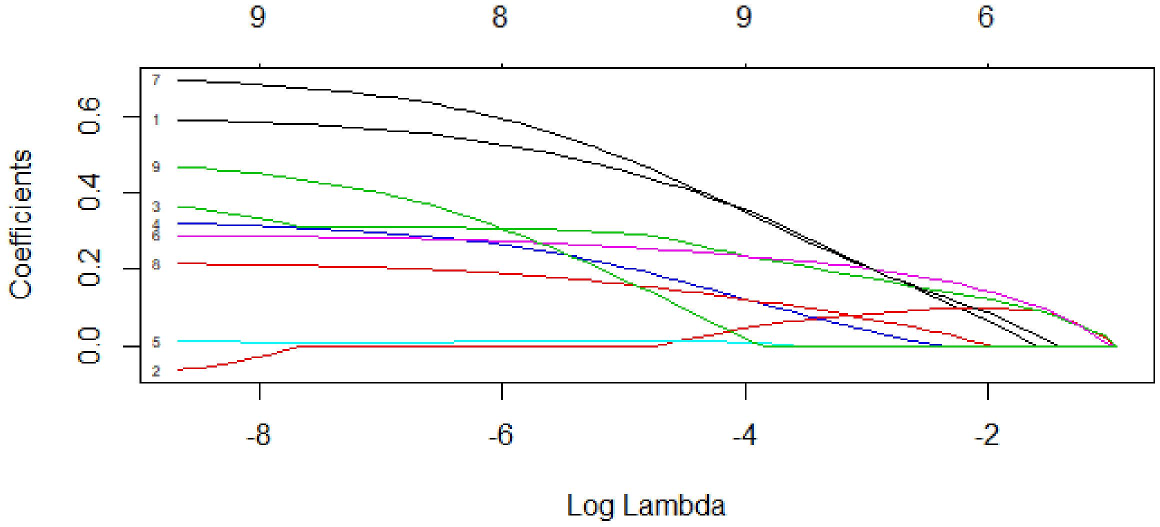


Figure 4.8: Relation between, Coeff.size, number of features and $log(\lambda)$

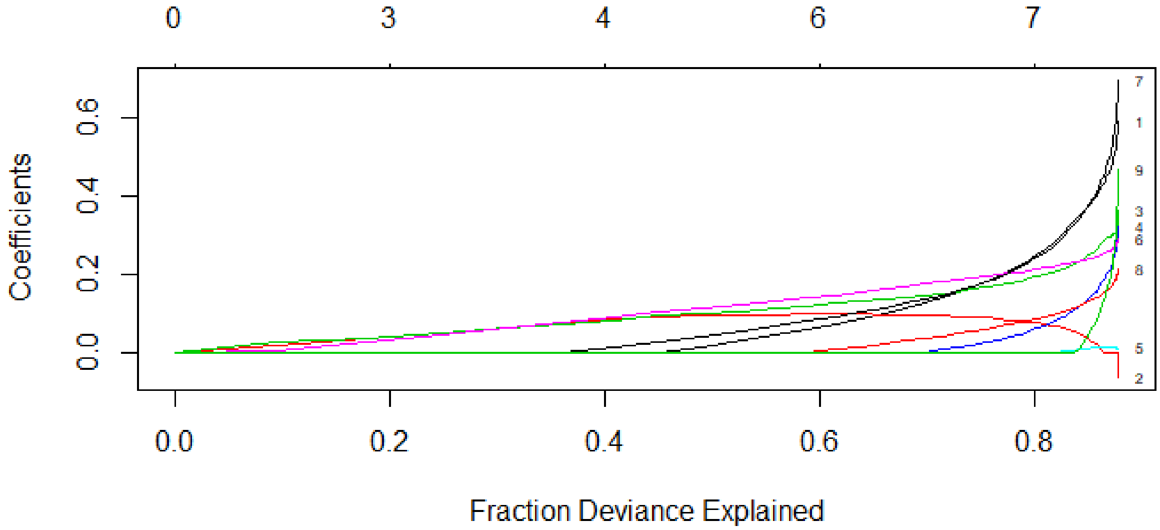


Figure 4.9: Relation b/n, Coeff.size, number of features and % Dev Explained

Figure 4.9 shows the fraction deviance explained by different model and their coefficient.

From figure 4.9 about 80% of the information in the data is explained by including seven features in the model. We choose λ to be 0.04170697 and 0.08187249 for lasso and ridge respectively using the crossvalidation and the 1-SE rule which are shown in Figures 4.10 and 4.11. The corresponding a selected model are in Figures 4.12 and 4.13 for lasso and ridge receptively. The selected seven features and their coefficients are shown in Table 4.3 column 2 for the Lasso Logistic regression model. Column 1 of Table 4.3 also shows the coefficients of Ridge Logistic regression model. Table 4.3 shows important predictors using the best predictive model with L_1 penalty. The features Epith.c.size and Mitoses were removed from the final model. The prediction accuracy of each model using the test data was 0.9659 for lasso and 0.9756 for Ridge.

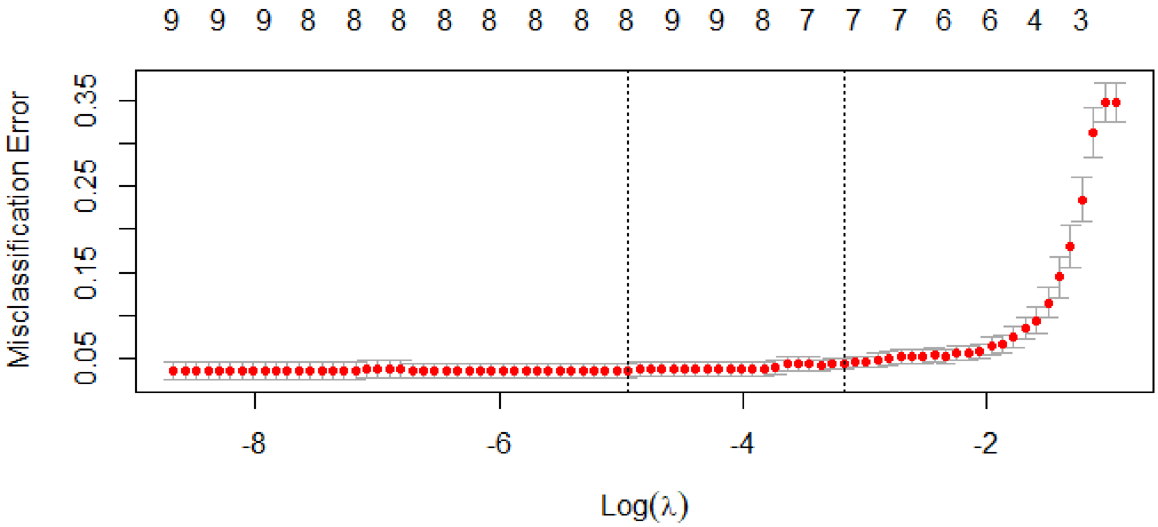


Figure 4.10: selecting λ using 10 fold cross validation

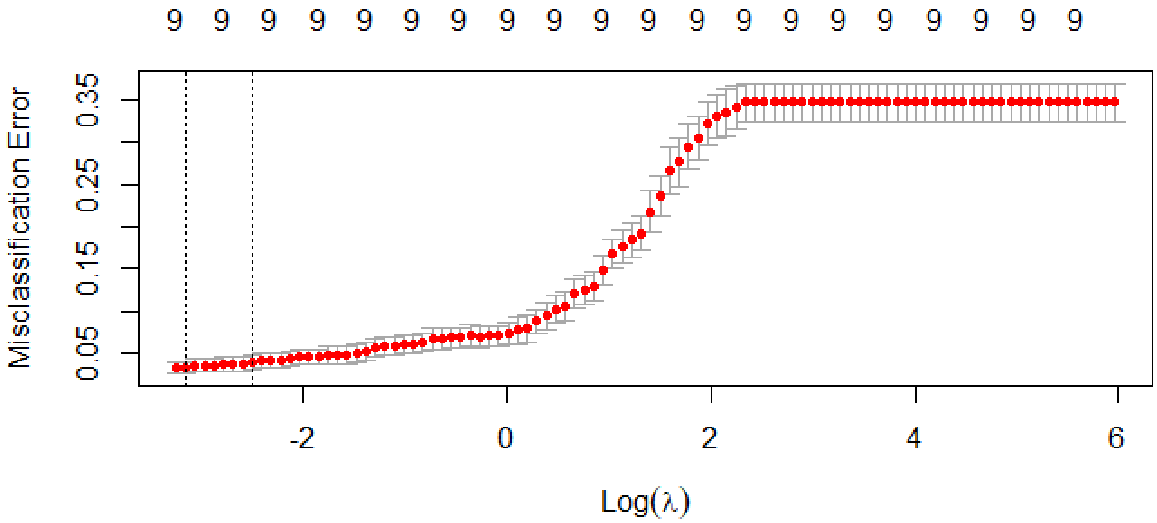


Figure 4.11: selecting λ using 10 fold cross validation

Table 4.3: Cofficient,Pred. Accuracy of Lasso and Ridge LR model

Features	**LR-Ridge Coefficient**	**LR-Lasso Coefficient**
Cl.thickness	0.1712177	0.23606
Cell.size	0.1253754	0.07934
Cell.shape	0.1434834	0.18897
Marg.adhesion	0.1140987	0.05437
Epith.c.size	0.1267267	*
Bare.nuclei	0.1421901	0.20966
Bl.cromatin	0.1804671	0.2299
Normal.nucleoli	0.1115392	0.08242
Mitoses	0.1274586	*
Pred. Accuracy	**0.9756**	**0.9659**

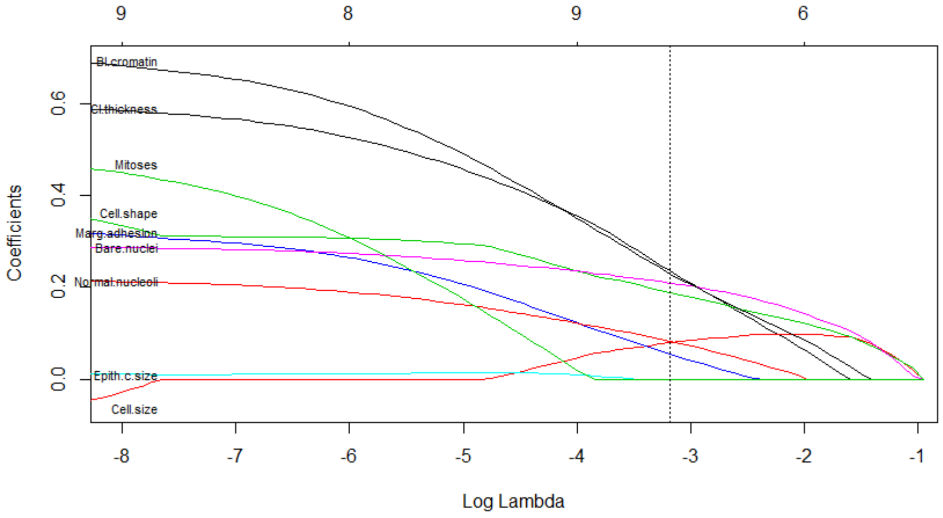


Figure 4.12: Best λ by 10 fold cross validation and coefficient of LR ridge

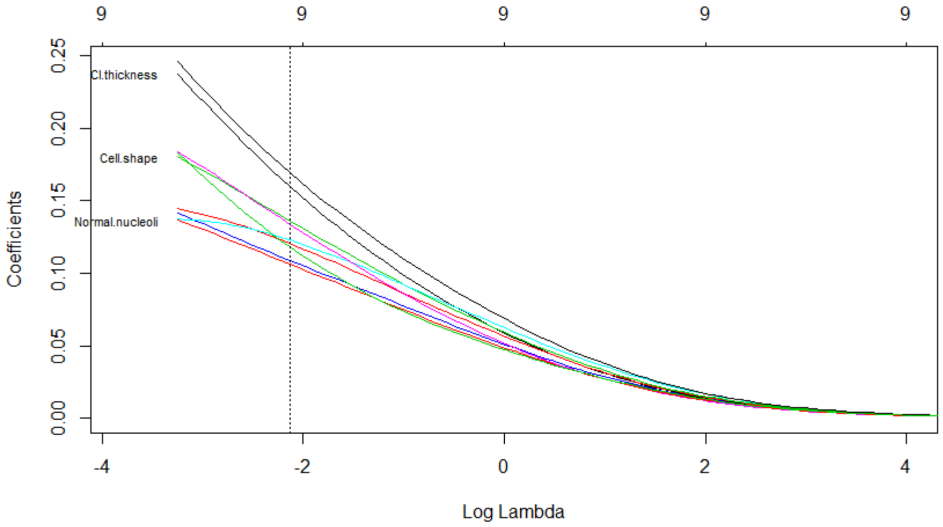


Figure 4.13: Best λ by 10 fold cross validation and coefficient of LR ridge

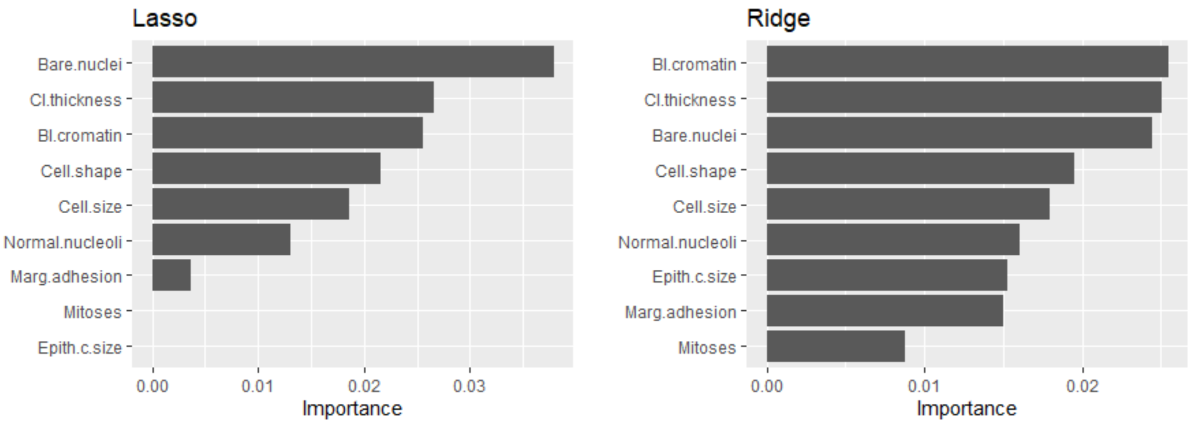


Figure 4.14: Variables importance plot for Lasso and Ridge regression

Figure 4.14 show the variables importance plot for both lasso and ridge regularized regression. The top three predictor for cancer predictions were Bare.nuclei, CI.thickness and BI.cromatin. The features Mitoses and Epith.c.size were dropped by the lasso model.

4.2.2 Logistic Principal Component Regression(LR-PC):

The second analysis is the principal component regression a dimension reduction method. We reduce the dimension of the original correlated data from nine to three uncorrelated as explain in Figure 4.3. This transformation help overcomes the multicollinearity problem of the cancer and heart disease datasets. The response data and cross validation were used to confirm the observed components in Figure 4.15. The transformed data was used to build predictive model for cancer and heart disease datasets. We compared the predictive performance of PCR to the other models in Table 4.6.

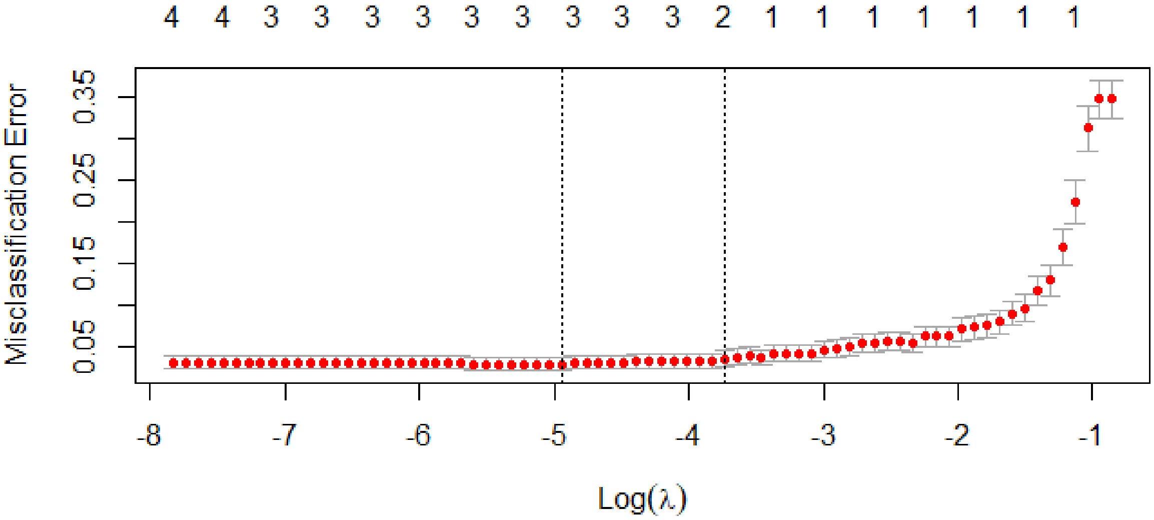


Figure 4.15: selecting λ and number of components using 10 fold cross validation

Table 4.4: Coefficient of Principal Components

	Estimate	**Std. Error**	**z value**	**P-value**
(Intercept)	-1.263	0.3278	-3.853	0.00011
PC1	-2.2708	0.2597	-8.746	0.00000
PC2	0.1489	0.387	0.385	0.07302
PC3	0.7754	0.3923	1.977	0.04809

Null deviance = 617.334 on 477 df and Residual deviance = 85.926 on 474 df

AIC: 93.926 and Pred. Accuracy: 0.9707

Table 4.4 column two show the parameter estimates of the principal components, all were significant at $\alpha = 0.05$ except PC2. The prediction accuracy on the test data was 0.9707.

4.2.3 Logistic Partial Least squares Regression(LR-PLS):

Next we fitted Logistic partial least square regression to the cancer data. Unlike the principal component regression were component are determine independent of the response, PLS uses the response variable to aid the construction of the principal components. Thus PLS ia a supervised dimension reduction method that finds new features that not only captures most of the information in the original features, but also are related to the response. The new features were used as predictor in the predictive model.

Similar to PCR, we fittted PLS model on the train set and used cross validation to select the number of principal components that maximizes predictive accuracy. Five component were used as shown in Figure 4.16. The prediction error of the final model on the test set was 0.961. We compared the predictive performance of LR-PLS to the other models in Table 4.6.

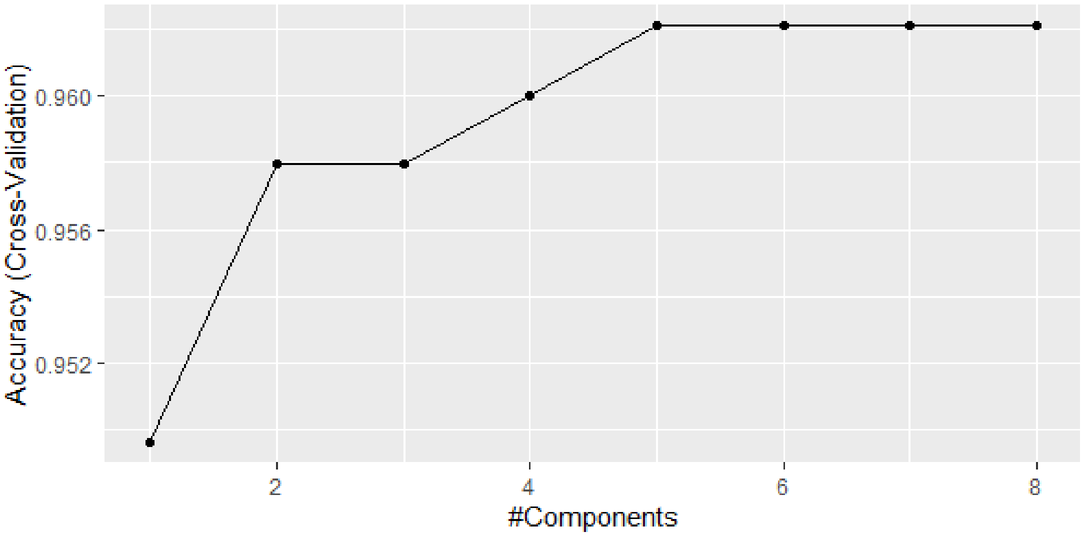


Figure 4.16: Selecting the number of component by CV

4.2.4 Multivariate Adaptive Regression Splines(MARS):

The multivariate adaptive regression splines (MARS) model was the next model we fitted to the Cancer data. This model search for, and discover, nonlinearities and interactions in the data that help maximize predictive accuracy. The two parameter: the degree of interactions and the number of terms retained in the final model were selected using 10-fold cross-validation. We perform a CV grid search to identify the optimal combination of these hyperparameters that minimize prediction error. The model that provides the optimal combination includes second degree interaction effects and retains 26 terms. The cross-validated prediction accuracy for these models is displayed in Figure 4.17. The optimal model's cross-validated prediction accuracy was 0.96 on the train set . The final model gave prediction accuracy of 0.9659 on the test data.

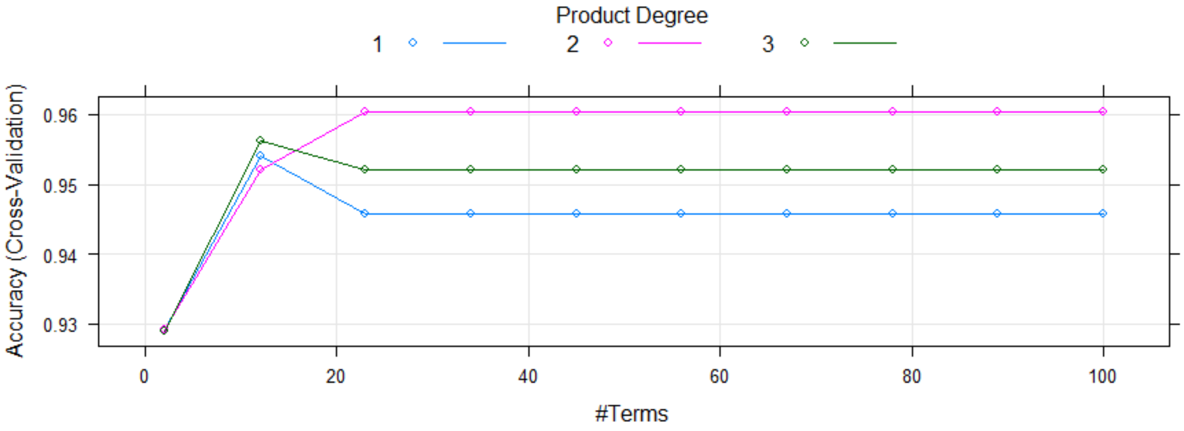


Figure 4.17: Selecting the number of component by CV

We ranked the predictors in terms of importance using the Generalized Cross-Validation (GCV) show in Figure 4.18. The GCV is a type of regularization technique that trades-off goodness-of-fit against the model complexity. From Figures 4.18, the Cell.size is most important predictor of cancer cancer. We compared the predictive performance of MARS to the other models in Table 4.6.

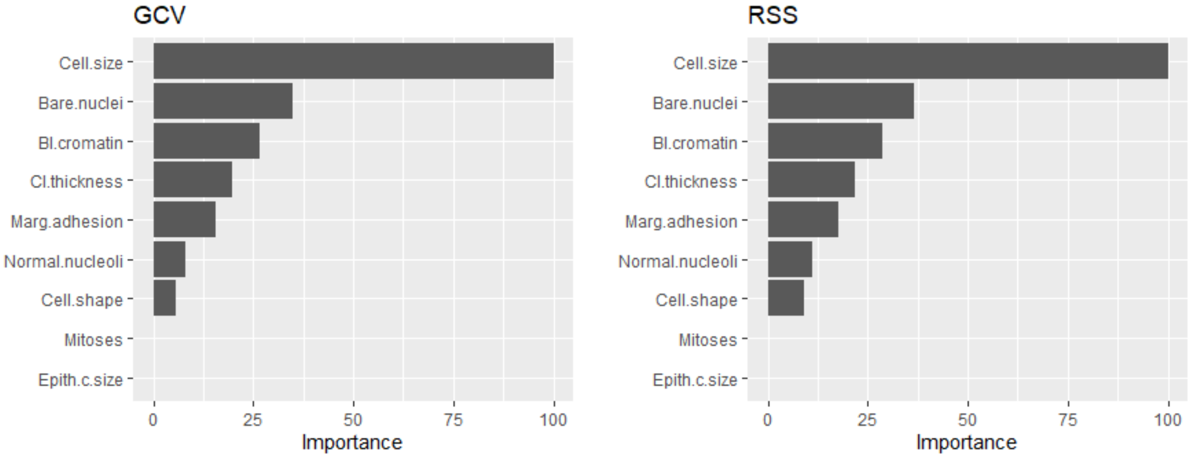


Figure 4.18: MARS Variables important plot

4.2.5 Random Forest(RF):

For random forests, we first train models with 1000 trees using the *randomForest* function in R. We search for the number of feature m randomly sampled as candidates at each split that gives the smallest OOB error to be 3 as shown in Figure 4.19. The OOB error of the random forest are stablized at $B = 500$ trees as shown in Figure 4.20.

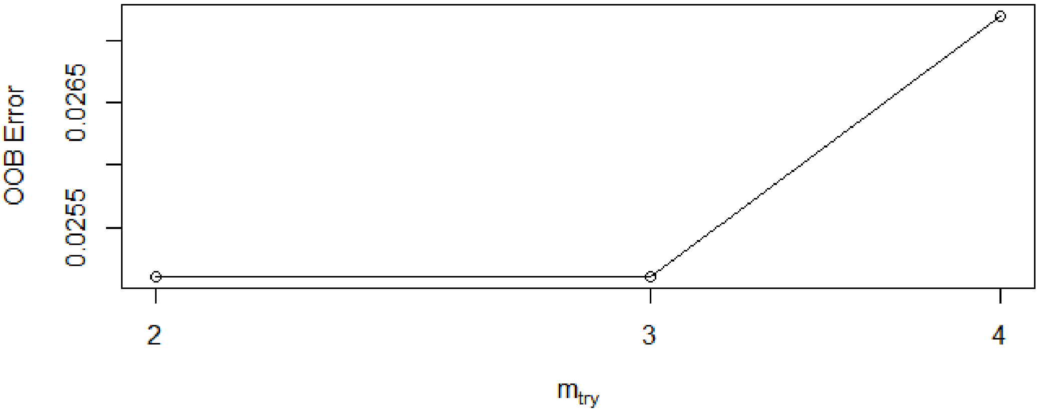


Figure 4.19: Selecting the split size

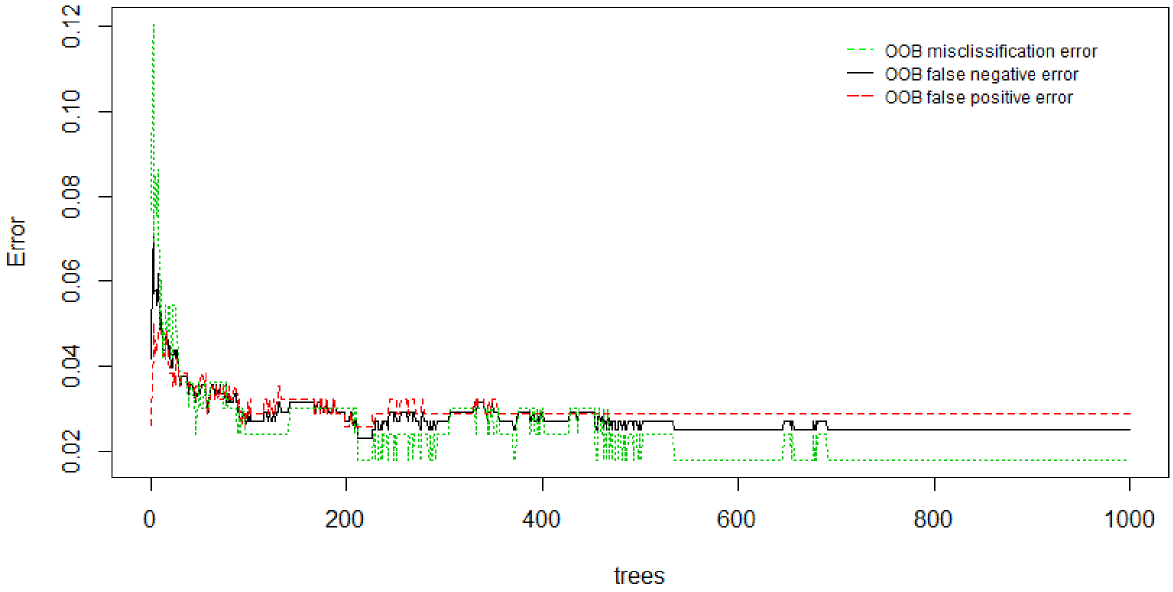


Figure 4.20: Selecting the Tree size

The random forest model with $m = 3$ and $B = 500$ on the Cancer test data gave a prediction accuracy of 0.9756.

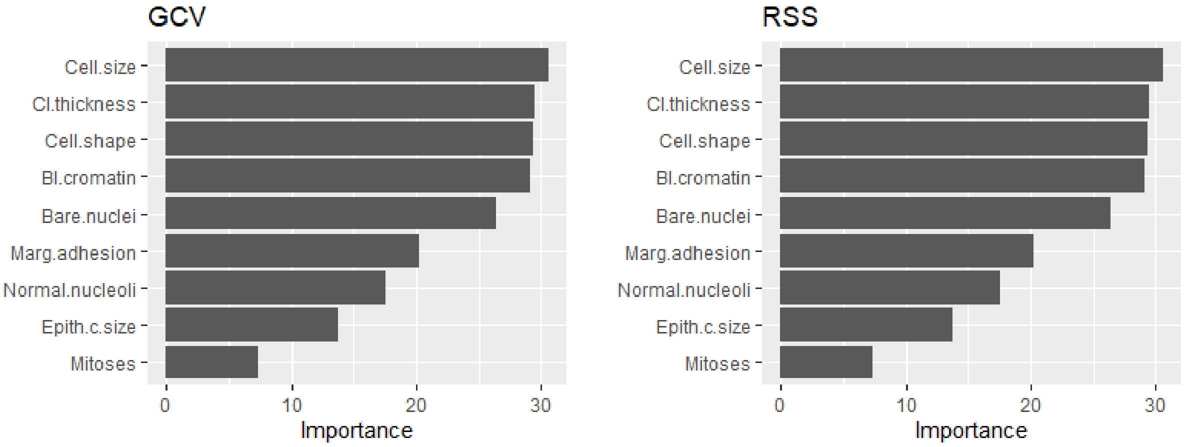


Figure 4.21: RF important variables in Predicting Cancer

Figure 4.21 shows variable importance ranked using the Mean Decrease Gini indices. The figure shows that Cell.size, CI.thickness and Cell.shape to be top three most important variable in predicting breast cancer. We compared the predictive performance of RF to the other models in Table 4.6.

4.2.6 Gradient Boosting

The gradient boosting algorithm with 1000 tree was fitted to to the cancer data. The optimal tree size by 10 fold cross validation was 120 as show in Figure 4.22. The blue dotted line indicates the best iteration, and the black and green curves indicate the training and cross validation error respectively.

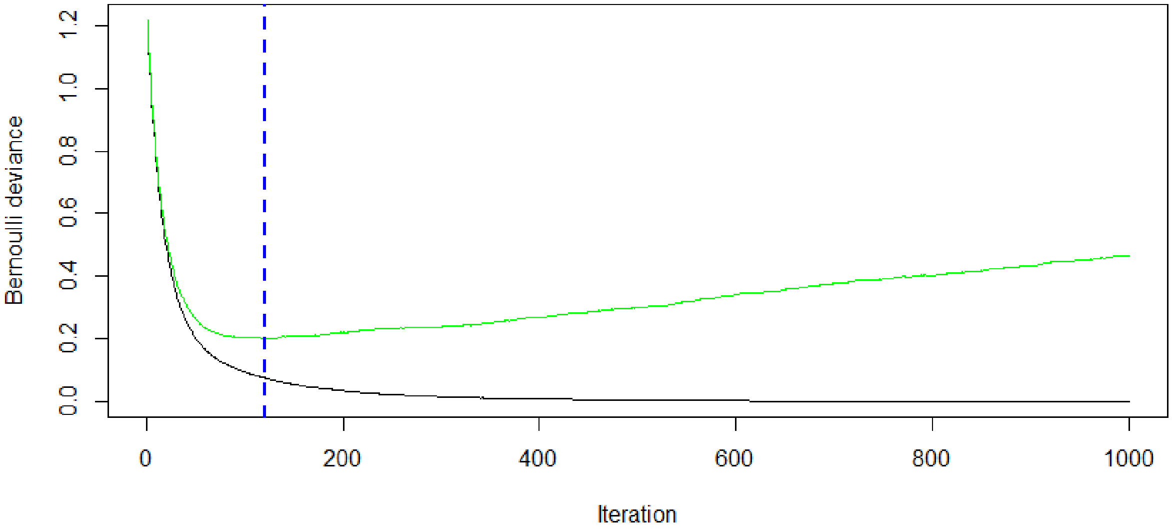


Figure 4.22: Selecting the Tree size

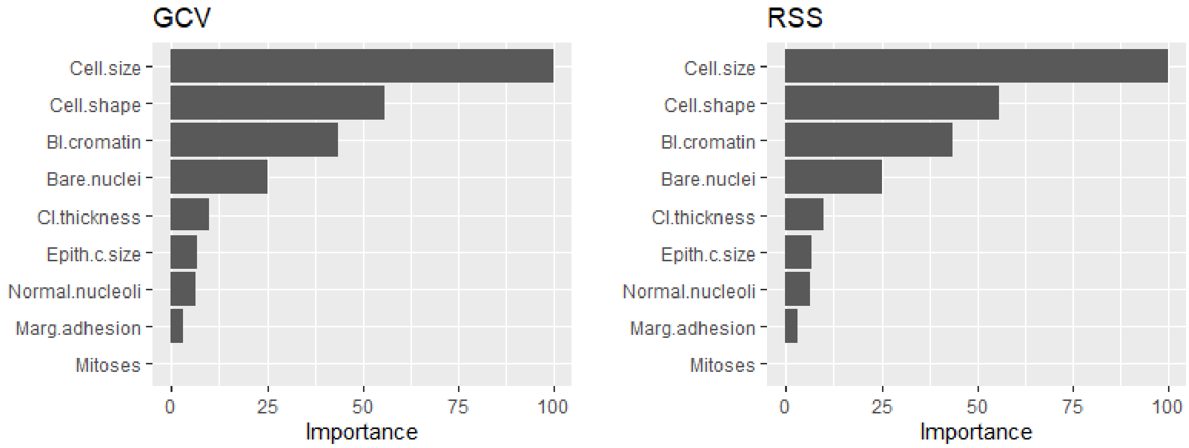


Figure 4.23: GBM important Variable

Again for the interpretation, we display the relative importance of variables in boosted trees. From the variable importance plot in Figure 4.23, Cell.size, Cell.shape and Bi.cromatin are the top three most important predictors of breast cancer. This is consistent with the result of RF. The prediction accuracy form test data and the selected model is 0.9707. We compared the predictive performance of GBM to the other models in Table 4.6.

4.2.7 Support Vector Machine

We applied the Gaussian/Radial basis kernel (SVM-GK) to the Cancer data and used 10-fold cross validation to tune two parameters γ and C over the search gird $\gamma \in 10\%, 50\%, 90\%$ quantiles of $\|\mathrm{x} - \mathrm{x}'\|$ and $C \in 2\hat{}(-2:7)$. The best tuning parameters on our search grid gave gamma $= 0.02257$ and cost $= 0.5$. shown in Figure 4.24.

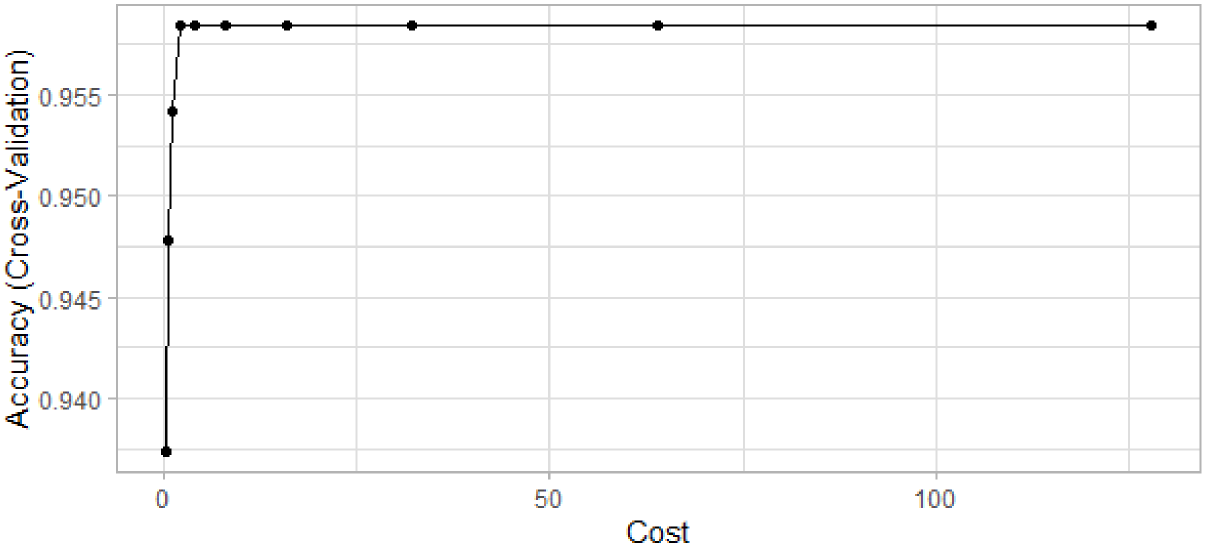


Figure 4.24: Selecting the Cost parameter

The resulting model on the test set gave a prediction accuracy of 0.9756. Figure 4.25 shows the variable importance plot from the SVM. Bare .nuclei is most important and Mitoses the lease important predictor of Cancer which is shown in Figure 4.25. We compared the predictive performance of SVM to the other models in Table 4.6.

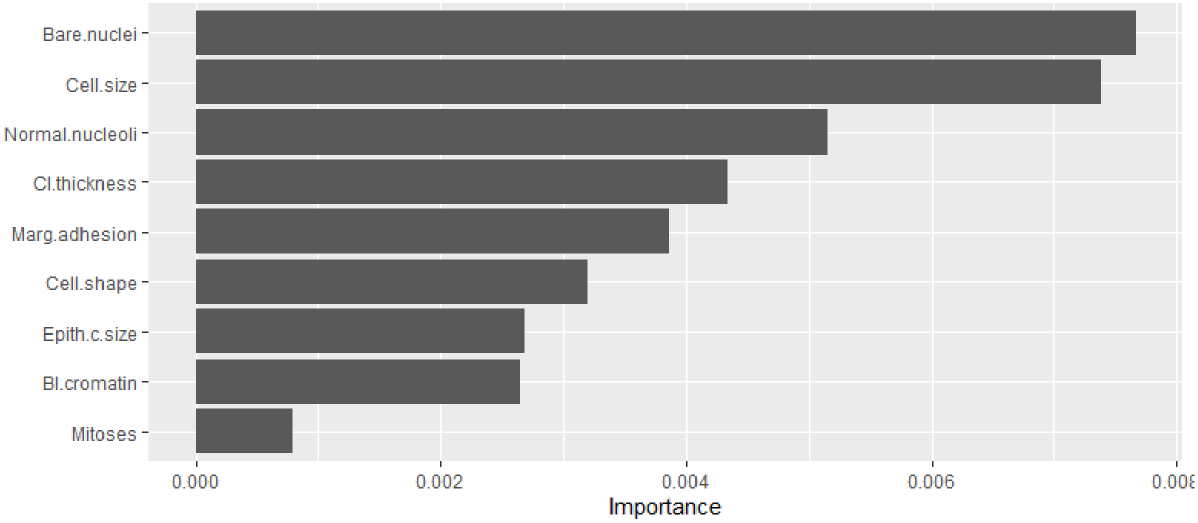


Figure 4.25: SVM important variables

4.2.8 Deep Learning: Feed froward Network (FFNN)

The final model applied to the Cancer data was Feed Forward Neural Network. To avid over-fitting due to the data size we used random-hyperparameter search and cross validation to determine the optimal network configuration because of the large parameter space. The parameter we optimized were the activation function, the number of hidden layers, the number of neurons(units) in each hidden layer, epochs,Learning rate and regularization λ for L_1 and L_2 penalties. Hundred models were bulid from the serach. Below are top five model ordered by miss classifications error in table 4.5.

Table 4.5: Selecting the optimal model for FFNN using miss classification error

Actva.	**Epochs**	**Hidden**	**IDR**	**L1**	**L2**	**L.Rate**	**Miss.Err**
Maxout	100	[9,3,2]	0.050	8E-05	4E-05	0.020	0.006
Maxout	100	[5,5,2]	0.000	9E-05	8E-05	0.010	0.015
Maxout	100	[9,2]	0.000	2E-05	1E-02	0.010	0.015
Maxout	50	[9,3,2]	0.000	3E-05	5E-05	0.020	0.015
Maxout	100	[9,2]	0.000	9E-05	2E-02	0.020	0.015

The best model in table 4.5 row 1, fitted to the test data gave prediction accuracy of 0.9707317. Figure 4.26 show the important predictors. We compared the predictive performance of FFNN to the other models in Table 4.6.

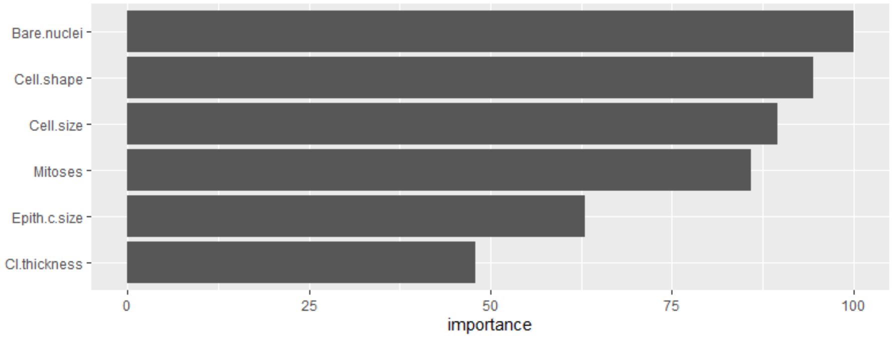


Figure 4.26: FFNN important variables

4.2.9 Models Summary for Breast Cancer Data

Table 4.6: All model applied to the Breast Cancer Data

Model	**Sensitivity**	**Specificity**	**Accuracy**	**95% CI**
LR-Ridge	0.9724	0.9452	0.9656	(0.9440, 0.9920)
LR-Lasso	0.9848	0.9315	0.9659	(0.9309, 0.9862)
LR-Enet	0.9848	0.9452	0.9707	(0.9374, 0.9892)
LR-PC	0.9848	0.9452	0.9707	(0.9374, 0.9892)
LR-PLS	0.9773	0.9315	0.9610	(0.9246, 0.9830)
MARS	0.9589	0.9621	0.9610	(0.9246, 0.9830)
SVM	0.9848	0.9589	0.9756	(0.9440, 0.9920)
RF	0.9773	0.9726	0.9756	(0.9440, 0.9920)
GBM	0.9773	0.9589	0.9707	(0.9374, 0.9892)
FFNN	0.9473	0.9920	0.97561	

Table 4.6 compares the result of 10 model applied to the breast cancer data. The models are grouped according to their similarities and learning style. i) Linear regularized models: LR-Lasso, LR-Ridge and LR-Enet. ii) Linear dimension reduction models: PCR and PLSR. iii) Non-Linear ensemble models : Random forest and Gradient Boosting. iv) Other Non-Linear models: FFNN, SVM and MARS. From Table 4.6, the non linear models: SVM, RF and FFNN gave the best prediction accuracy of 0.9756. From the variable importance plots in Figures 4.21, 4.25, 4.26 the top risk factors of breast cancer are uniformity of cellsize (Cell.size), uniformity of cell shape (Cell.shape), bare nuclei (Bare.nuclei) and bland chromatin (Bl.cromatin).

4.3 Analysis on Heart Disease Data

The heart disease data used in this analysis was also obtained from UCI machine learning repository[49], it has 14 features that play a role in explaining the cause of heart disease. The features include age of patients, sex, chest pain, resting blood pressure, fasting blood sugar, number of major vessel, and several others.

Table 4.7: Features name and short Description

Variables	**Description**
Age	Age of patientin years
sex	Sex, 1 for male
cp	Chest pain
trestbps	Resting blood pressure
chol	Serum cholesterol
fbs	Fasting blood sugar larger 120mg/dl (1true)
restec	Gresting electroc. result (1 anomality)
thalach	Maximum heart rate achieved
Exang	Exercise induced angina
Oldpeak	ST depression induced by exercise relative to rest
Slopethe	Slope of the peak exercise ST segment
ca	Number of major vessel
Thal	Thalassamia
num	Angiographic disease status

Table 4.7 is the summary and description of heart disease dataset. The target variable is num that contains the rate of diameter narrowing of coronary artery. It takes value 0 when the rate $< 50\%$, and value 1 when the rate $> 50\%$. We assume that the patient has no heart disease when *num* is 0 and the patient has heart disease when num is 1. The goal

is to predict the response variable *num* using ML method to determine whether a patient has heart disease.

4.3.1 Association of Data

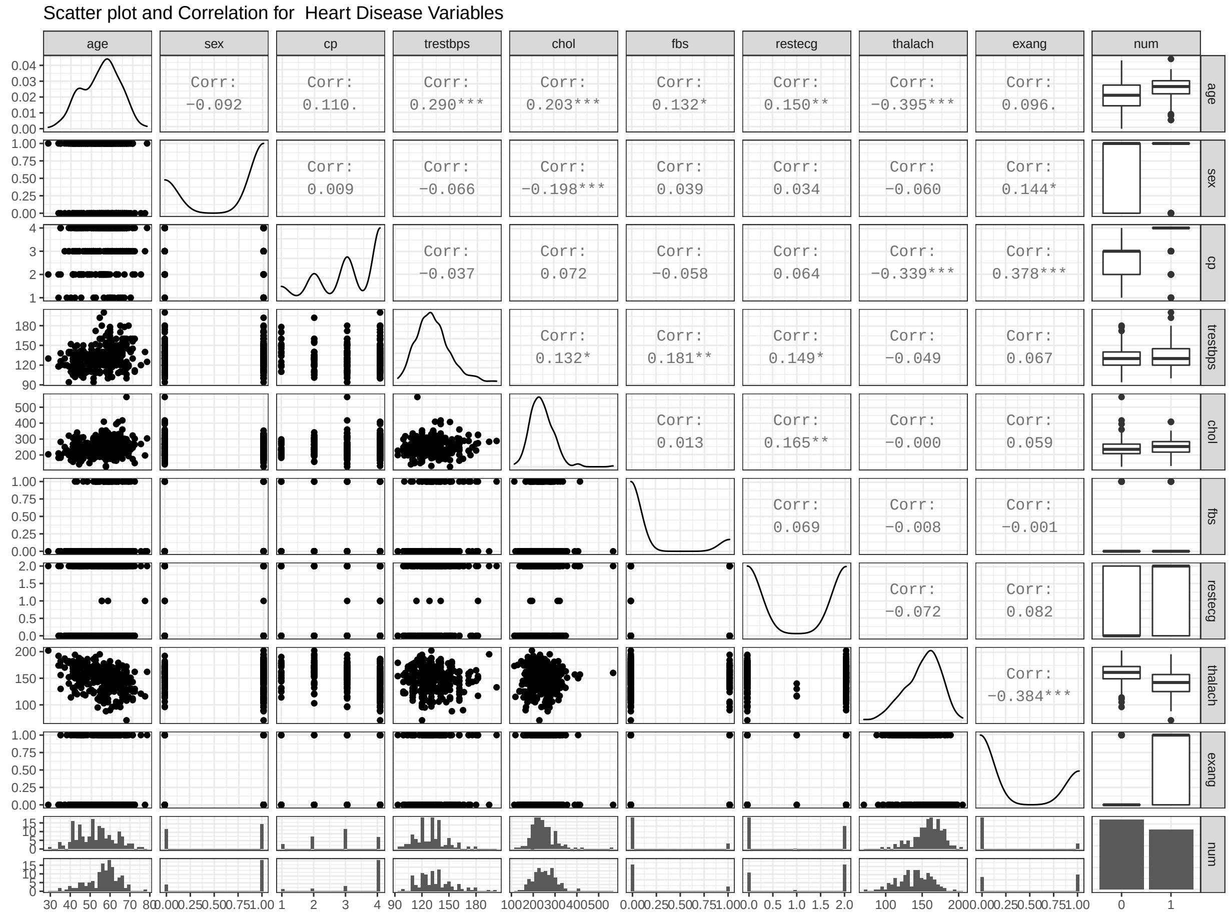


Figure 4.27: Scatter plot and Correlation for Heart Disease variables

Figure 4.27, shows the individual variables distribution, scatter plot and correlation between the Heart disease variables. Age, trestbps, chol and thalach appears approximately normal while fbs, restecg and esang appears bi-modal. Almost all the variable show weak association from the correlation value ranging from -0.395 to 0.378.

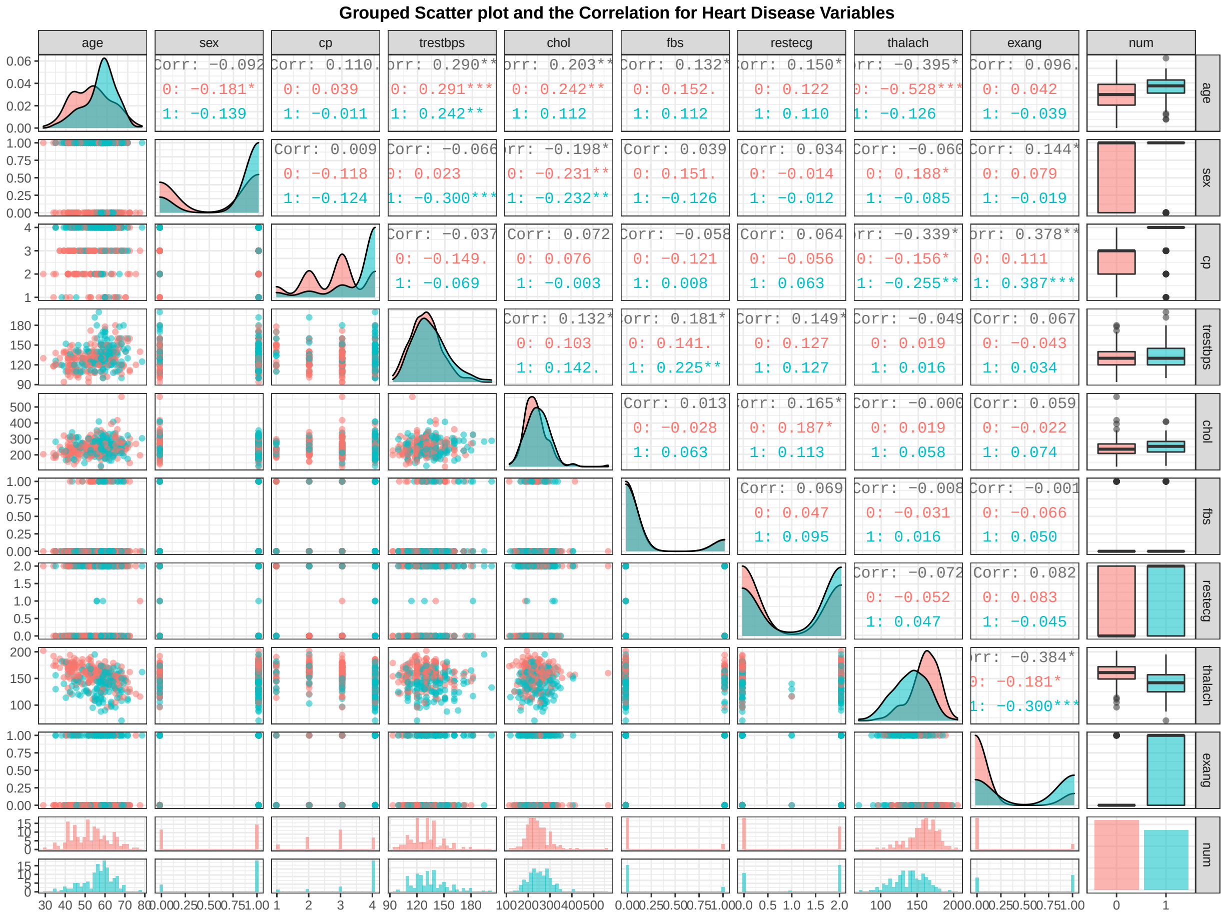


Figure 4.28: Scatter plot and Correlation between Variables by response

Figure 4.28 shows the group distribution, scatter plot and correlation between the variables and response. The density curve (diagonal) shows that the variables distribution of both heart disease and non-heart disease groups to be very similar. In both groups age,trestbps, chol and thalach appears approximately normal while fbs, restecg and esang appears bimodal.

The boxplot (left margin) shows that the heart disease group variable has a lot of outliers compared to the non-heart disease group which show outlier only in the trestbps and chol variables. Generally the scatter plots show no significant trends between both groups.

4.3.2 Principal Component Analysis (PCA):

In this section we use Principal Component analysis to investigate the relationship between variable and observations to get insight of the data. Table 4.8 shows the percentage on information explained (retained) by the each of the principal components. From the table about 81% of the information are retained by $PC1 - PC8$, out of the 14 components.

PCA Summary

Table 4.8: % variation explained by each component

	PC1	**PC2**	**PC3**	**PC4**	**PC5**	**PC6**	**PC7**	**PC8**	**PC9**
Std Dev	1.76	1.27	1.12	1.05	1.00	0.93	0.92	0.88	0.83
Prop. var	0.24	0.12	0.10	0.09	0.08	0.07	0.06	0.06	0.05
Cum. Prop.	0.24	0.36	0.46	0.54	0.62	0.69	0.75	0.81	0.86

Figure 4.29 shows the contribution of each feature to the dimensions of the principal component. It can be seen that eight of the features contribute to the first component and four to the second component.

Figure 4.29: All Variables-PC.

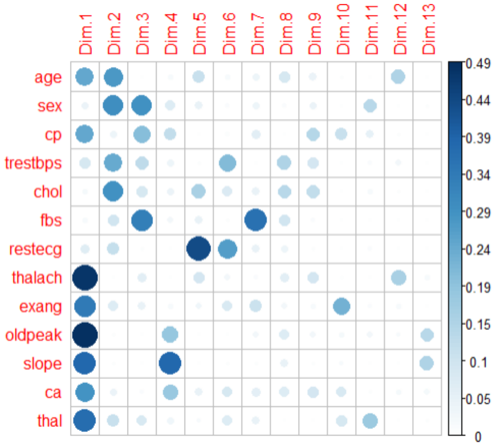


Figure 4.30: Variables-PCA correlation.

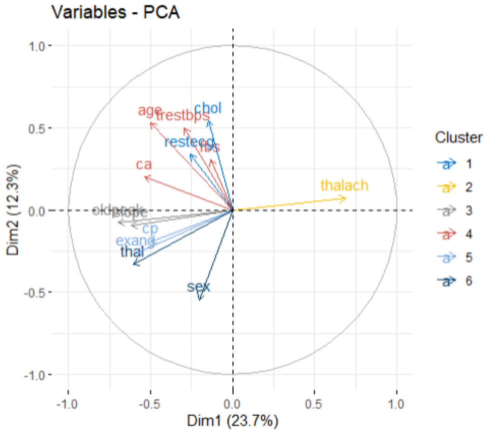


Figure 4.30 is the variable PCA plot which shows correlation between variables and the general meaning of the dimension of the components. Positive correlated variables point to the same side of the plot while negative correlated variables point to opposite sides of the graph. From figure 4.30 we can see that all the variable are positively correlated and in the same but negative direction on PC1 except thalach. About half of the variables are on the positive and half on the negative side of PC2 except thalach which appear very different. This figure consistent with correlation Matrix plot in Figure 4.27. The interpretation we assign to PC1 is the level of endurance from exercise before chest pain and PC2 is the cholesterol level given age and gender.

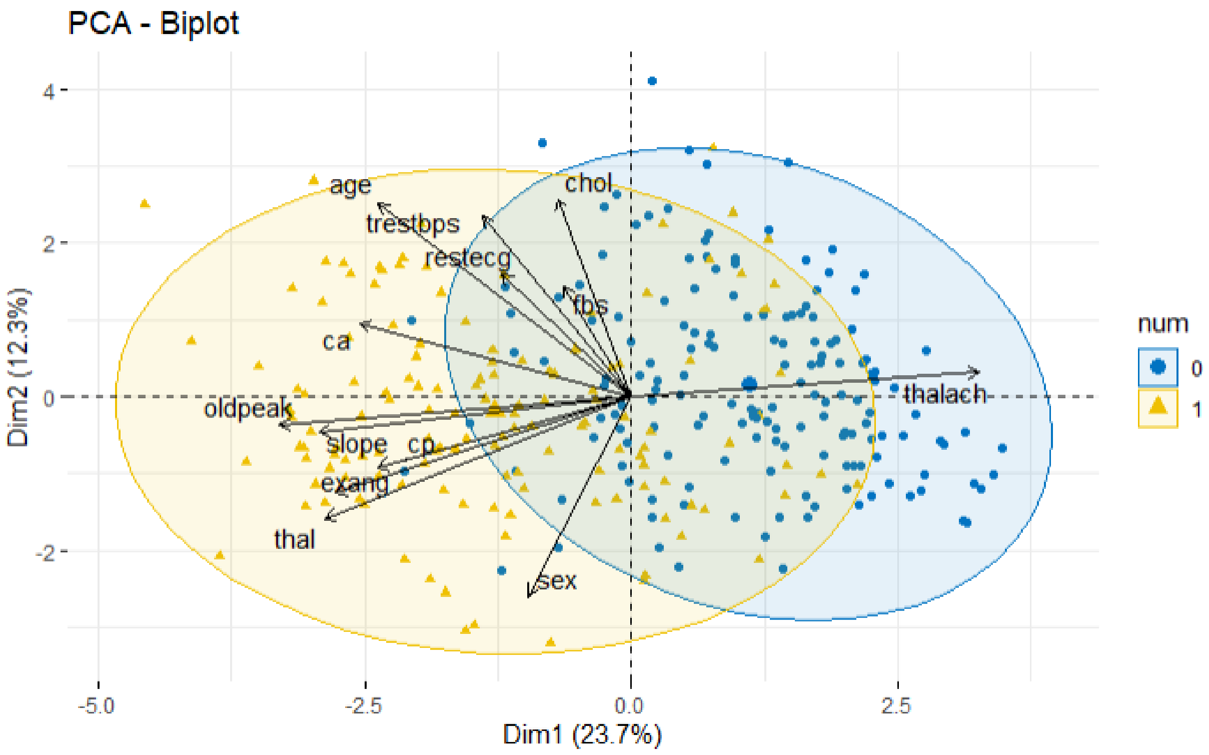


Figure 4.31: Heart disease PCA Biplot

Figure 4.31 is a biplot which combines both the features and individual on the PC1 and PC2 with response of each individual in blue (no heart disease) or yellow (heart disease) colored. The plot shows individuals with similar profile group together. From the figure we conclude that subject who tend to endure more exercise before chest pain and have low cholesterol turn to be at low risk of heart disease. Contrary subject who tend to endure less exercise before chest pain and have high cholesterol turn to be at high risk of heart disease.

4.4 Application of Statistical learning methods

4.4.1 Logistic regression

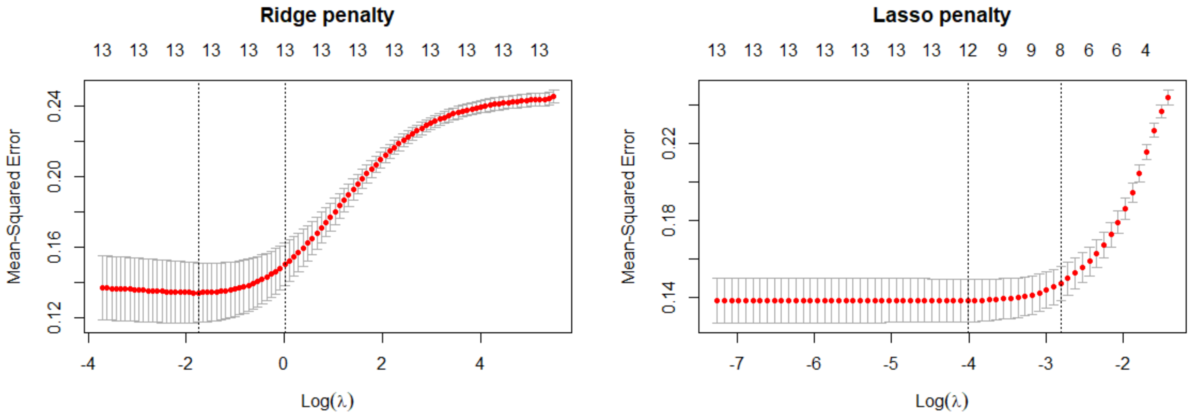


Figure 4.32: Selecting λ using 10 fold cross validation

Figure 4.32 shows the tuning parameter λ and the number of features selected by LR-ridge (left) and LR-lasso(right). We choose λ to be 0.05035599 and 0.363611 for lasso and ridge respectively using the crossvalidation and the 1-SE rule. The selected eight features and their coefficients are shown in Table 4.9 column 2 for the Lasso Logistic regression model. Column 1 of Table 4.9 show the coefficients of Ridge Logistic regression model.

Table 4.9: Estimate Coefficient of Lasso and Ridge LR

Features	LR-Ridge	LR-Lasso
age	0.001162064	
sex	0.073681097	0.037073089
cp	0.042463517	0.044754279
trestbps	0.000916134	
chol	0.00024749	
fbs	-0.015930603	
restecg	0.025136613	0.013670152
thalach	-0.001578803	-0.001014997
exang	0.096665553	0.135966412
oldpeak	0.036505923	0.050496442
slope	0.034953927	
ca	0.05829244	0.112452634
thal	0.025019609	0.040342951
Pred. Accuracy	**0.7576**	**0.8182**

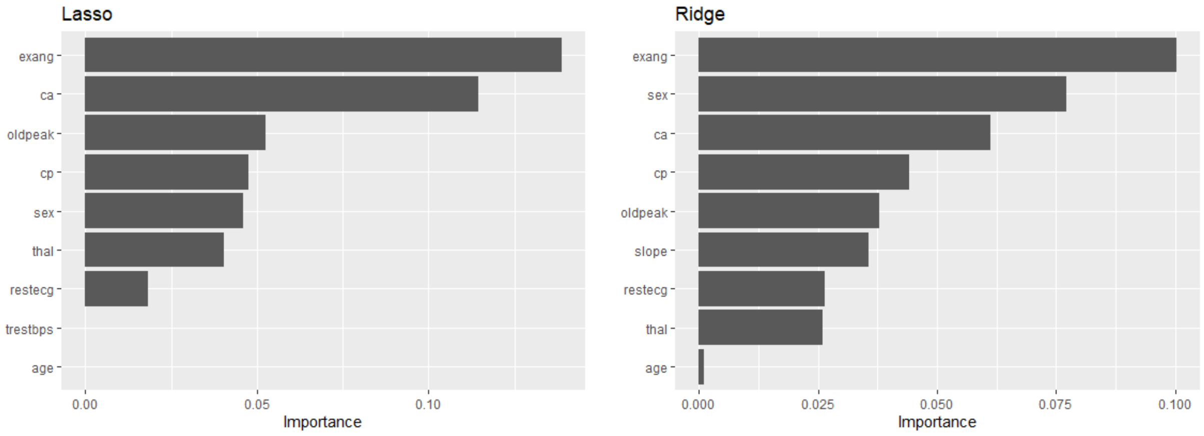


Figure 4.33: Important variable plot using Lasso and Ridge regression

Figure 4.33 shows important predictors using the best predictive model with L_1 and L_2 penalties. The features exang, ca, oldpeak and exang, sex, ca were the top three predictors for Lasso and Ridge LR respectively . The prediction accuracy of each model using the test data was 0.8182 for lasso and 0.8384 for Ridge.

4.4.2 Logistic Principal Component Regression(LR-PC):

The eight PC components obtained in section (4.3.2) were used as predictor to fit a logistic regression to the heart disease data. The coefficients of the resulting model are in Table 4.10. We compared the predictive performance of PCR to the other models in Table 4.12.

Table 4.10: LR-PC coefficient

Estimate	**Std.**	**Error**	**z**	**value**	**P-value**
(Intercept)	-0.4469	0.2103	-2.125	0.0336	
PC1	1.24853	0.16947	7.367	1.7E-13	***
PC2	0.19191	0.153	1.254	0.2097	
PC3	-0.0129	0.16888	-0.076	0.939	
PC4	-0.438	0.18799	-2.33	0.0198	*
PC5	-0.4096	0.19367	-2.115	0.0344	*
PC7	0.27663	0.22134	1.25	0.2114	
PC8	-0.0978	0.22904	-0.427	0.6694	

Null deviance = 269.92 on 197 df and Residual deviance = 148.97 on 190 df

AIC: 164.97 and Pred. Accuracy: 0.8384

4.4.3 Logistic Partial Least squares Regression(LR-PLS)

The Logistic partial least square regression to the Heart disease data. The PLS model was fitted to the train set and used cross validation to select the number of principal components that maximizes predictive accuracy. One component were used as shown in

Figure 4.34(left).The The prediction error of the final model on the test set was 0.8384. We compared the predictive performance of LR-PLS to the other models in Table 4.12.

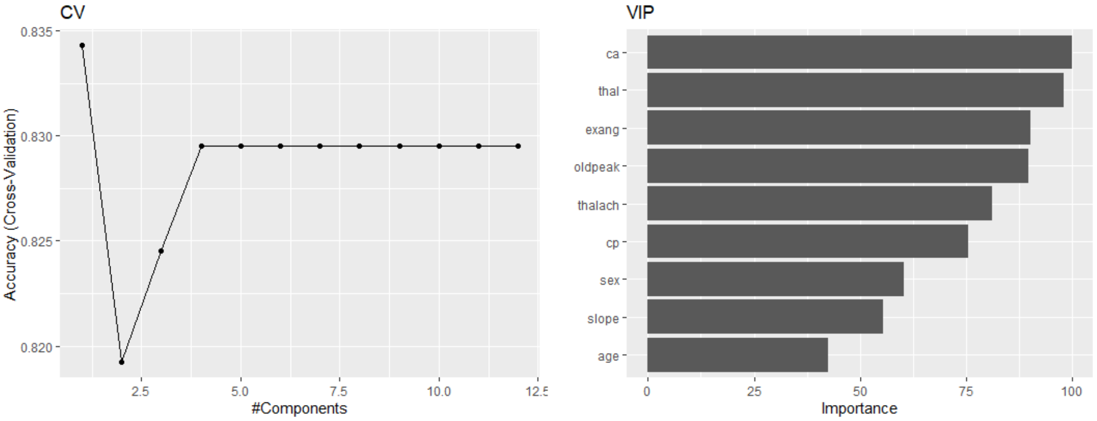


Figure 4.34: Selecting the number of components by CV (left) and variables importance plot (right).

4.4.4 Multivariate Adaptive Regression Splines: MARS

We perform a 10-fold cross-validation grid search to identify the optimal combination of these hyperparameters that minimize prediction error. The model that provides the optimal combination includes first degree interaction effects and retains 12 terms. The cross-validated prediction accuracy for these models is displayed in Figure 4.35.

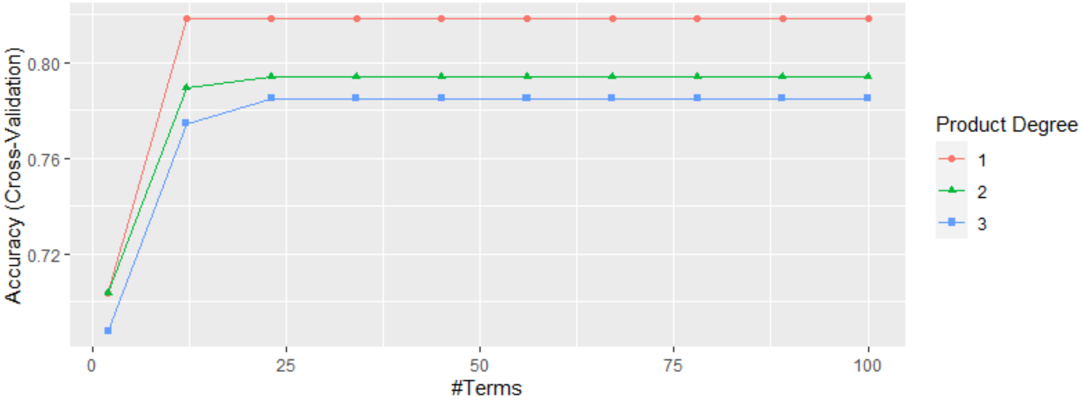


Figure 4.35: Selecting the number of component by CV

The optimal model's cross-validated prediction accuracy was 0.85 on the train set . The final model gave prediction error of 0.7374 on the test data.

We ranked the predictors in terms of importance using the generalized cross-validation (GCV) show in Figure 4.36. From Figures 4.36, the ca, cp and slope are the top three important predictor of Heart disease. We compared the predictive performance of MARS to the other models in Table 4.12.

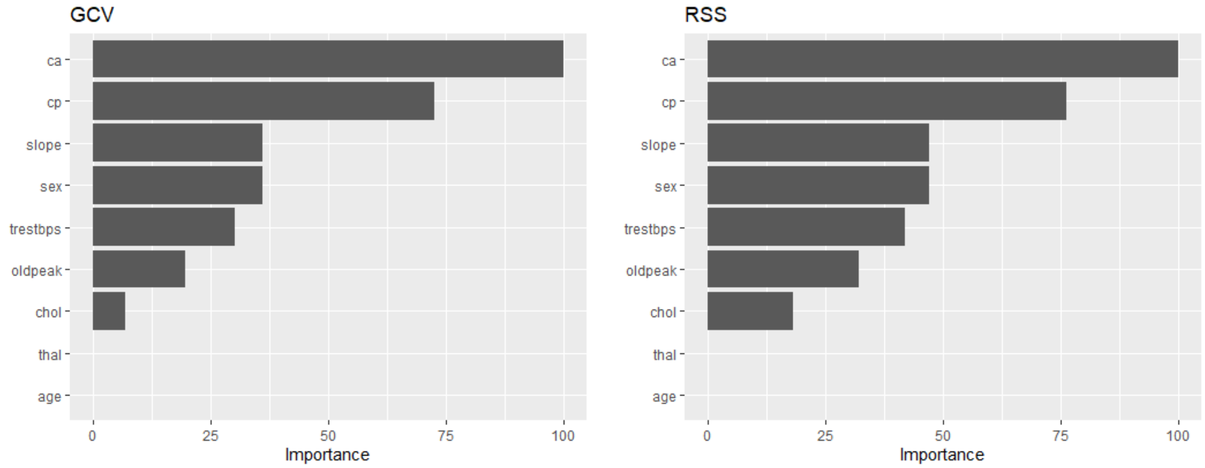


Figure 4.36: MARS variables importance plot

4.4.5 Random Forest(RF):

We train models with 1000 trees and search for the number of feature m to randomly sampled as candidates at each split. We obtained m with the smallest OOB error to be 6 as shown in Figure 4.37. The OOB error of the random forest are stablized at $B = 500$ trees as shown in Figure 4.38. The random forest model with $m = 3$ and $B = 500$ on the heart disease test data gave a prediction accuracy of 0.9756.

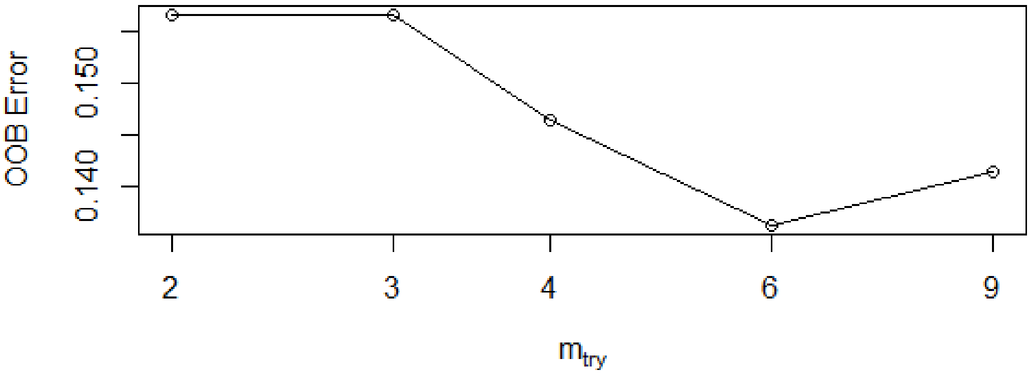


Figure 4.37: Selecting the split size

Figure 4.39 shows variable importance ranked using the Mean Decrease Gini indices. The figure shows that ca, oldpeak and cp to be top three most important predictors of Heart disease. We compared the predictive performance of RF to the other models in Table 4.12.

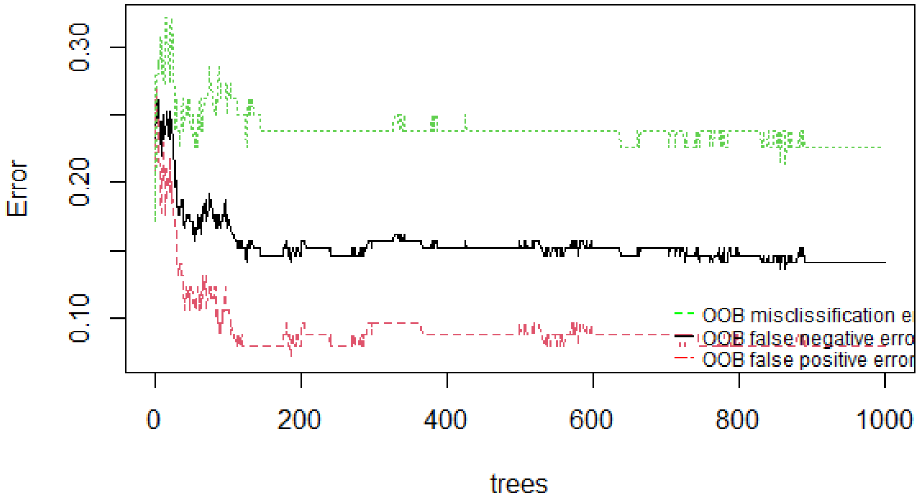


Figure 4.38: Selecting the Tree size

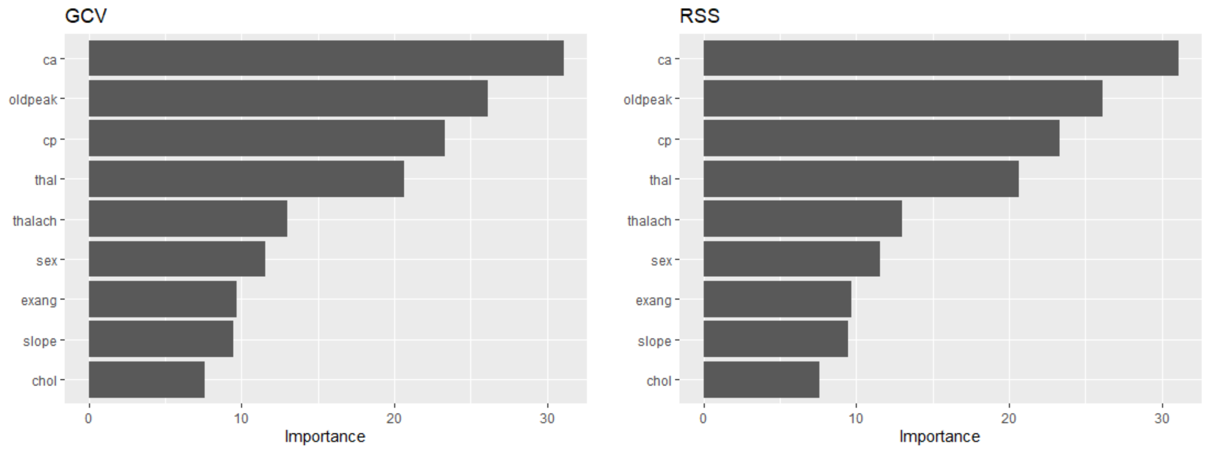


Figure 4.39: RF variables importance plot

4.4.6 Gradient Boosting(GBM):

The gradient boosting algorithm with 10000 tree was fitted to to the Heart disease data. The optimal tree size by 10 fold cross validation was 915 as show in Figure 4.40. The blue dotted line indicates the best iteration, and the black and green curves indicate the training and cross validation error respectively.

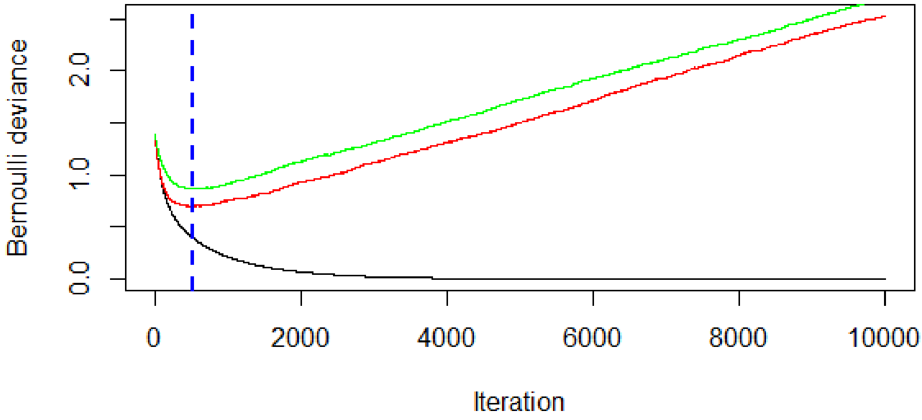


Figure 4.40: Selecting the Tree size

For the interpretation, we display the relative importance of variables in boosted trees. From the variable importance plot in Figure 4.41, chol, cp and ca are the top three most important predictors of Heart disease. The prediction accuracy form test data and the selected model is 0.8283. We compared the predictive performance of GBM to the other models in Table 4.12.

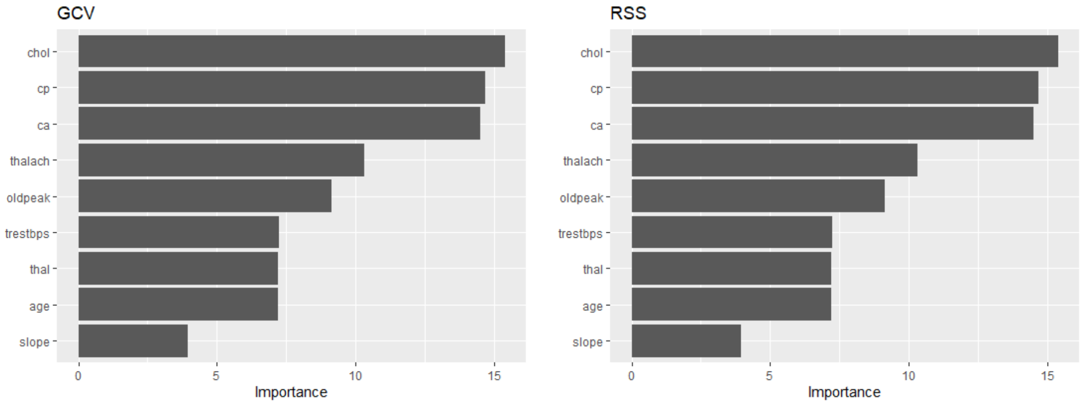


Figure 4.41: GBM important Variable

4.4.7 Support Vector Machine(SVM):

We applied the Gaussian kernel SVM to the Heart disease data and used 10-fold cross validation to tune two parameters γ and C over the search gird $\gamma \in 10\%, 50\%, 90\%$ quantiles of $\|\mathrm{x} - \mathrm{x}'\|$ and $C \in 2\hat{}(-2:7)$. The best tuning parameters on our search grid gave gamma $= 0.04194202$ and cost $= 0.25$. shown in Figure 4.43.

The resulting model on the test set gave a prediction accuracy of 0.8384. Figure **??** shows the variable importance plot for the SVM. ca, thal and oldpeak are the top three most important predictor of Heart disease. We compared the predictive performance of SVM to the other models in Table 4.12.

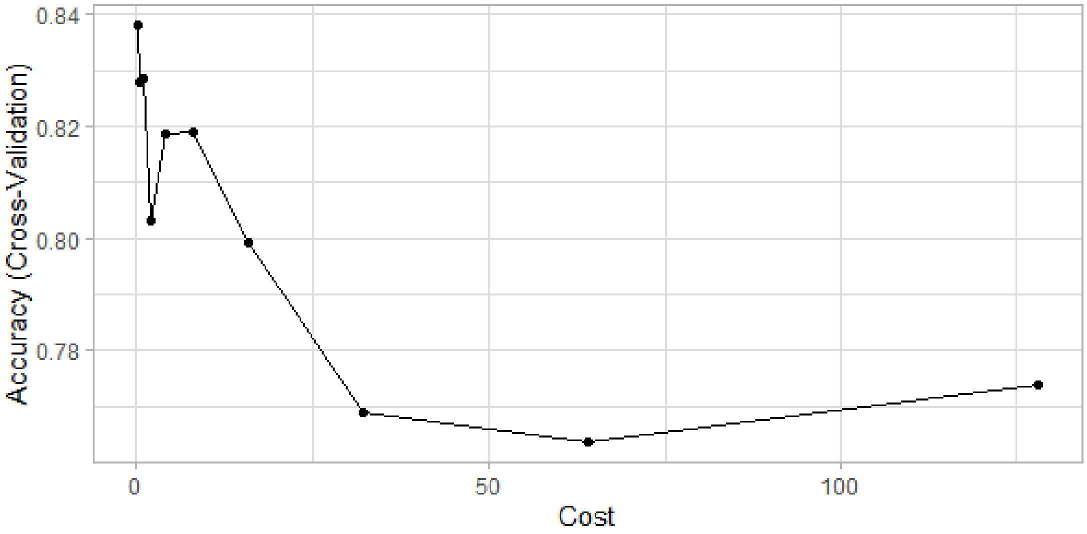


Figure 4.42: Selecting the Cost parameter

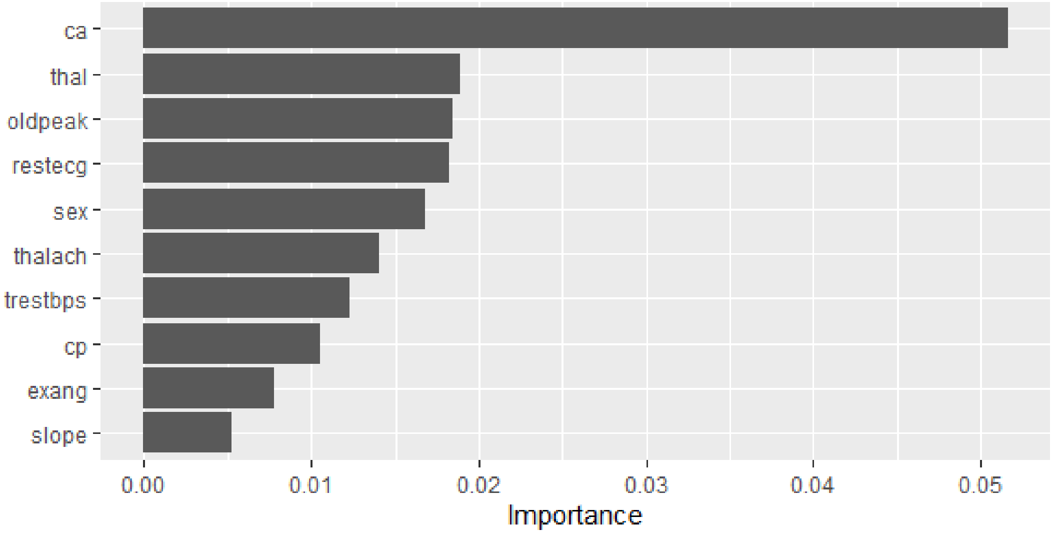


Figure 4.43: SVM variables importance plot

4.4.8 Deep Learning: Feed froward Network(FFNN):

The top five FFN model ordered by miss classifications error form the random hyper-parameter search using cross validation is shown in Table 4.11.

Table 4.11: Selecting the optimal model for FFNN using miss classification error

Actv.	**Epochs**	**Hidden**	**IDR**	**L1**	**L2**	**L.Rate**	**Miss. Err**
Maxout	100	[14,3,2]	0.000	9.4E-05	8E-05	0.010	0.015
Maxout	100	[14,3,2]	0.000	3.5E-05	9E-05	0.020	0.020
Maxout	100	[14,3,2]	0.050	1.2E-05	9E-05	0.020	0.025
Maxout	100	[14,2]	0.050	1.5E-05	0.01	0.010	0.030
Rectifier	100	[14,3,2]	0.000	3.1E-05	8E-05	0.010	0.045

The best model from Table 4.11 row 1, fitted to the test data gave prediction accuracy of 0.8383. Figure 4.44 show the important predictors. We compared the predictive performance of FFNN to the other models in Table 4.12.

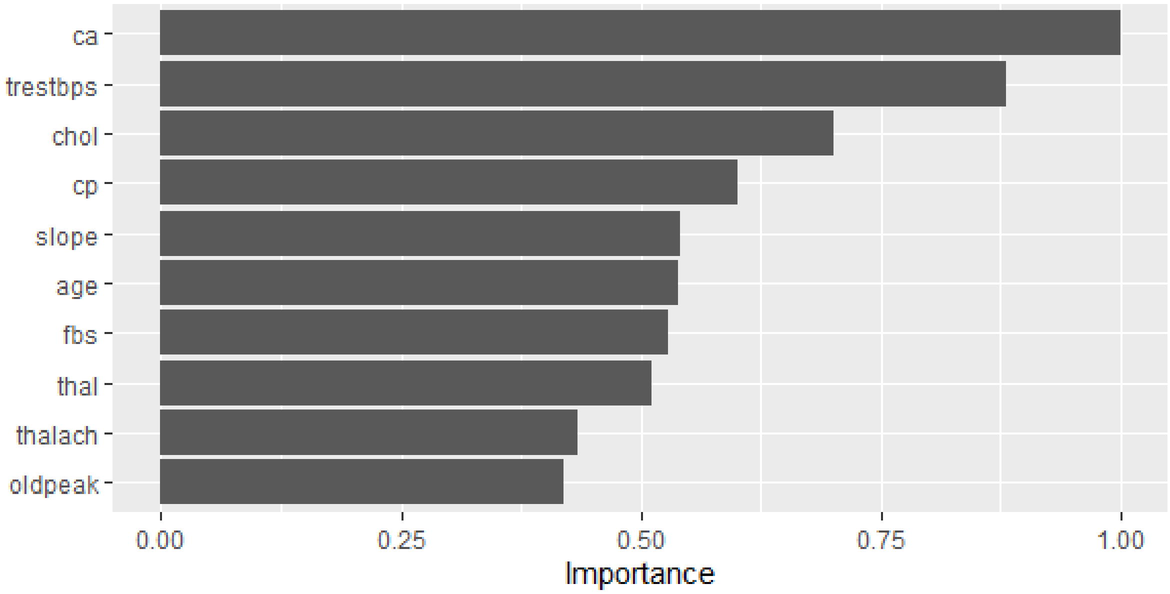


Figure 4.44: NN important variables

4.4.9 Models Summary

Table 4.12: All model applied to the Heart Disease data

Model	**Sensitivity**	**Specificity**	**Accuracy**	**95% CI**
LR-Ridge	0.9565	0.7358	0.8384	(0.7509, 0.9047)
LR-Lasso	0.9783	0.6792	0.8182	(0.7280, 0.8885)
LR-Enet	0.9565	0.7358	0.8384	(0.7509, 0.9047)
LR-PC	0.9565	0.7358	0.8384	(0.7509, 0.9047)
LR-PLS	0.9565	0.7358	0.8384	(0.7509, 0.9047)
MARS	0.6038	0.8913	0.7374	(0.6393, 0.8207)
SVM	0.9565	0.7358	0.8384	(0.7509, 0.9047)
RF	0.9130	0.6981	0.7980	(0.7054, 0.872)
GBM	0.9348	0.7358	0.8283	(0.7394, 0.8967)
FFNN	0.8000	0.8700	0.8383	

Table 4.12 compares the result of 10 model applied to the heart disease data. The models are grouped according to their similarities and learning style. i) Linear regularized models: LR-Lasso, LR-Ridge and LR-Enet. ii) Linear dimension reduction models: PCR and PLSR. iii) Non-Linear ensemble models : Random forest and Gradient Boosting. iv) Other Non-Linear models: FFNN, SVM and MARS. From Table 4.12, the linear models: LR-PLS, LR-PC ,LR-Ridge and LR-Enet gave the best prediction accuracy of 0.8384. From the variable important plots in Figures 4.33, 4.34, 4.65 the top risk factors of heart disease are number of major vessels (ca), Thalassamia (Thal) and exercise induced angina (Exang).

4.5 Analysis on Prostate Cancer Data

The final analysis is on the prostate cancer (PC), which is the second most common cancer among males worldwide that results in more than 350,000 deaths annually [29]. With more than 1 million (PC) new diagnoses reported every year, the key to decreasing mortality is developing more precise diagnostics. Diagnosis of PC is based on the grading of prostate tissue biopsies. These tissue samples are examined by a pathologist and scored according to the Gleason grading system [29]. In this analysis, we will develop models for predicting severity (or Gleason score) of prostate cancer using eight predictors.

The studied prostate cancer data came from a study by Stamey et al. (1989)[30]. The data consist of log cancer volume (lcavol), log prostate weight (lweight), age, log of the amount of benign prostatic hyperplasia (lbph), seminal vesicle invasion (svi), log of capsular penetration (lcp) Gleason score (gleason), and percent of Gleason scores 4 or 5(pgg45). The response we are predicting is the Gleason score (gleason). Table 4.13 has a brief description of the variable.

Table 4.13: Variables name and brief description.

Variable	**Description**
lcavol	log cancer volume
lweight	log prostate weight
age	Age
lbph	log of benign prostatic hyperplasia amount
svi	seminal vesicle invasion
lcp	log of capsular penetration
lpsa	log prostate specific antigen
pgg45	percent of Gleason scores 4 or 5
gleason	Gleason score

The grading process consists of finding and classifying cancer tissue into Gleason patterns (3, 4, or 5) based on the growth patterns of the tumor Figure 4.45 [29]. After the biopsy is assigned a Gleason score, it is converted into an ISUP grade on a 1-5 scale [31]. The Gleason grading system is the most important prognostic marker for PC, and the ISUP grade has a crucial role when deciding how a patient should be treated. There is both a risk of missing cancers and a large risk of over grading resulting in unnecessary treatment. However, the system suffers from significant inter-observer variability between pathologists, limiting its usefulness for individual patients. This variability in ratings could lead to unnecessary treatment, or worse, missing a severe diagnosis [29].

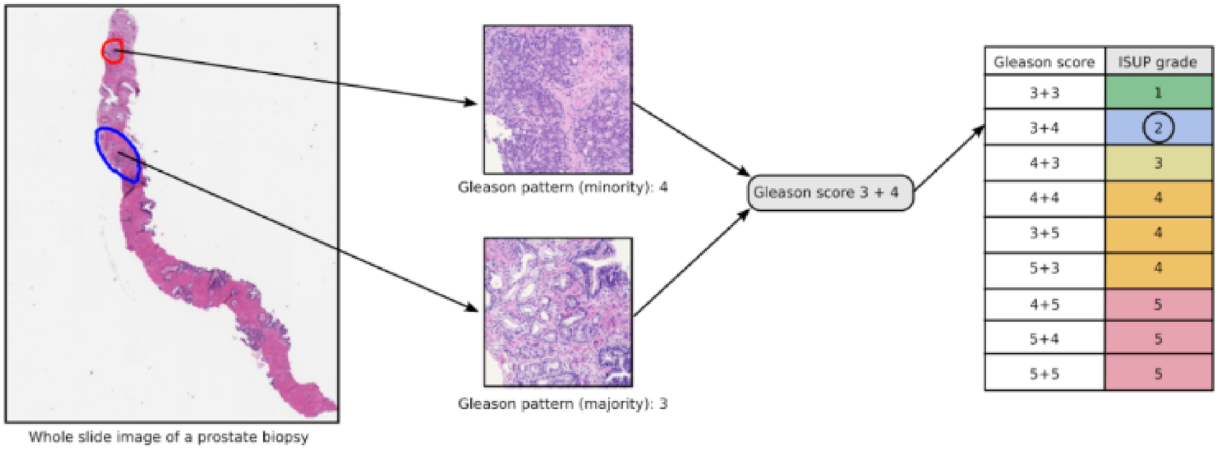

Whole slide image of a prostate biopsy
Gleason pattern (minority): 4
Gleason pattern (majority): 3
Gleason score 3 + 4
Gleason score
ISUP grade
3+3
1
3+4
2
4+3
3
4+4
4
3+5
4
5+3
4
4+5
5
5+4
5
5+5
5

Figure 4.45: An illustration of the Gleason grading process.[29]

Figure 4.45 is an illustration of the Gleason grading process for a biopsy containing prostate cancer. The most common (blue outline, Gleason pattern 3) and second most common (red outline, Gleason pattern 4) cancer growth patterns present in the biopsy dictate the Gleason score (3+4 for this biopsy), which in turn is converted into an ISUP grade (2 for this biopsy) following guidelines of the International Society of Urological Pathology. A Gleason score of 6 or less is considered low risk, 7 is intermediate risk, and a score of 8 to 10 is high risk cancer [32]. In our study we consider gleason score of 6 as low risk(LRPC) and score of 7 to 10 as high risk of prostate cancer(HRPC).

4.5.1 Association of Data

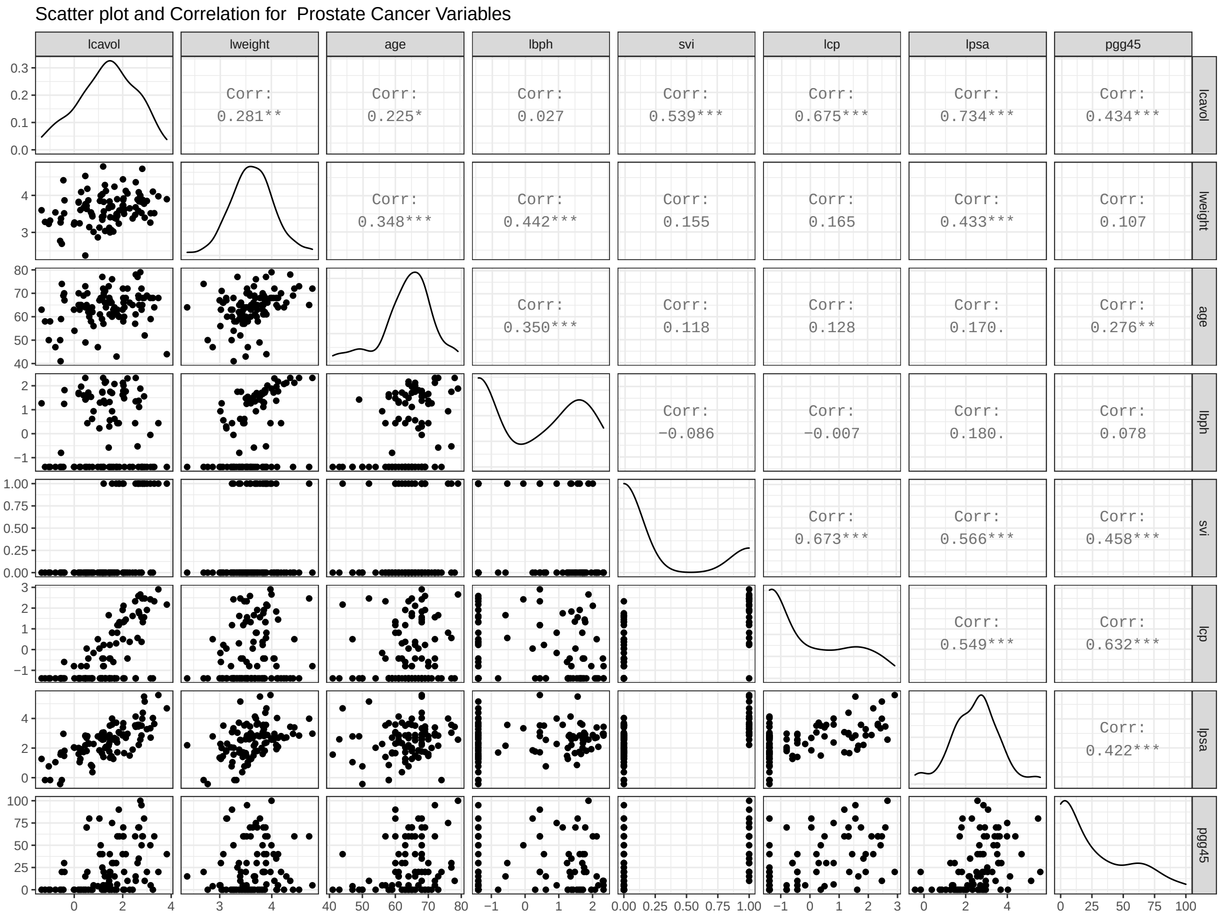


Figure 4.46: Scatter plot and Correlation for Prostate Cancer variables

The variables distribution and correlation matrix of the predictors given in Figure 4.46. The figure shows many strong correlations, example lcavol shows strong positive relationship with svi, lcp, and lpsa. The distribution of four of the variables are approximately normal with the remaining three either right-skewed or bi-modal.

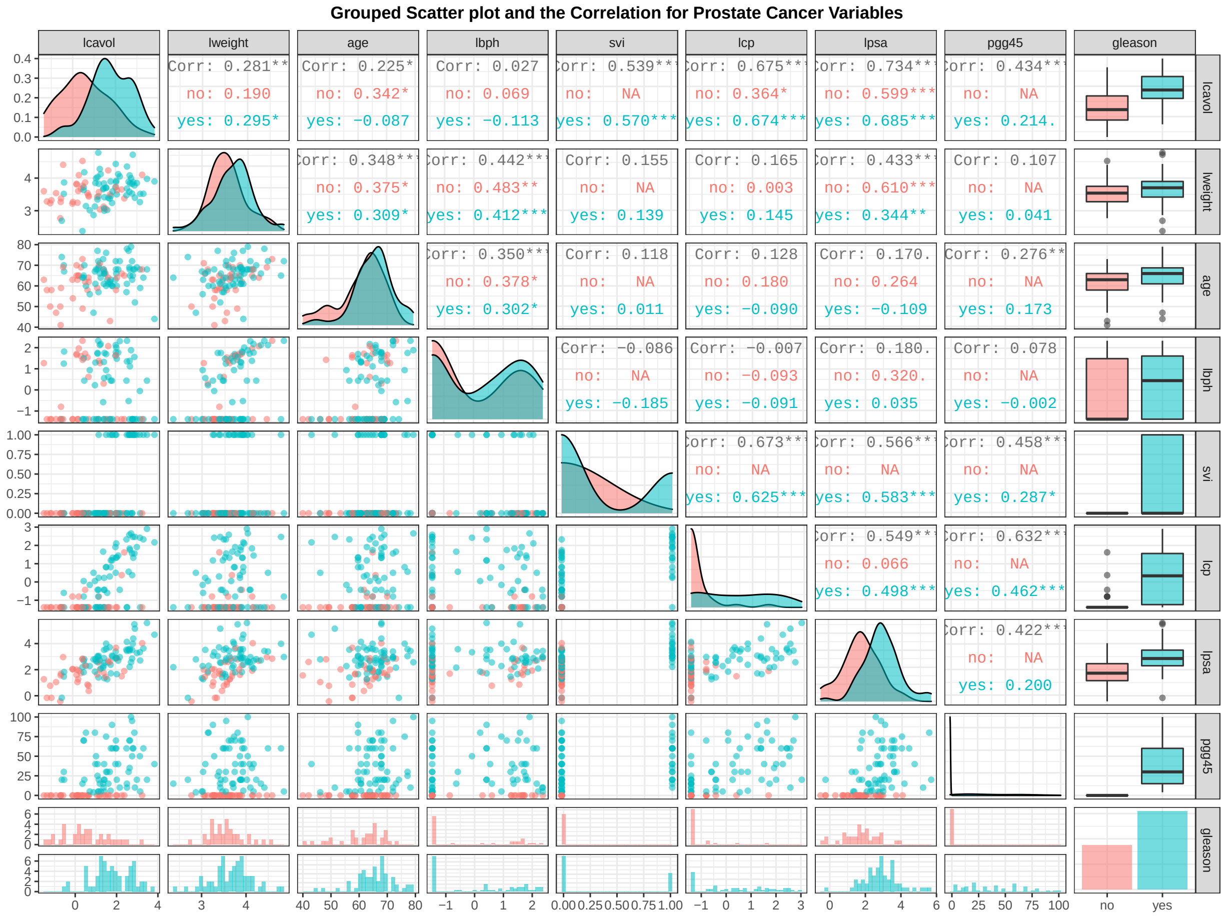


Figure 4.47: Scatter plot and Correlation between Variables by response.

Figure 4.47 shows the group distribution, scatter plot and correlation between the variables and response. The density curve (diagonal) shows the variables distribution for both high risk prostate cancer (green) and low risk prostate (red) cancer groups. From the density curves in Figure 4.47, the average lcavol, lweight, age and lpsa are higher in the high risk prostate cancer group compared to the low risk prostate cancer group. The boxplot (left margin) shows that the HRPC have higher median gleason score for all the variable except svi. Generally the scatter plots show positive linear trend in the HRPC compared to LRPC group which shows no specific trend between the variables.

4.5.2 Principal Component Analysis (PCA):

In this section we use Principal Component analysis to investigate the relationship between variable and observations to get insight of the data. Table 4.14 shows the percentage on information explained (retained) by the each of the principal components. From the table about 83% of the information are retained by $PC1 - PC4$, out of the 8 components.

PCA Summary

Table 4.14: % variation explained by each components

	PC1	**PC2**	**PC3**	**PC4**	**PC5**	**PC6**	**PC7**	**PC8**
Std. Dev.	1.87	1.28	0.93	0.78	0.68	0.65	0.57	0.40
Prop. Var.	0.44	0.21	0.11	0.08	0.06	0.05	0.04	0.02
Cum. Prop.	0.44	0.64	0.75	0.83	0.89	0.94	0.98	1.00

Figure 4.48 shows the contribution of each feature to the dimensions of the principal component. It can be seen that five of the features contribute to the first component and three to the second component.

Figure 4.49 is the variable PCA plot which shows correlation between variables and the interpretation of the dimension of the components. From the figure we can see that all the variable are positively correlated and in the same direction on PC1. About half of the variables are on the positive and half on the negative side of PC2. The interpretation we assign to PC1 is the volume of cancer (lcavol) given the amount of prostate specific antigen (lpsa) and PC2 is the level of prostate enlargement(lbph) and weight(lweight) given the patient age.

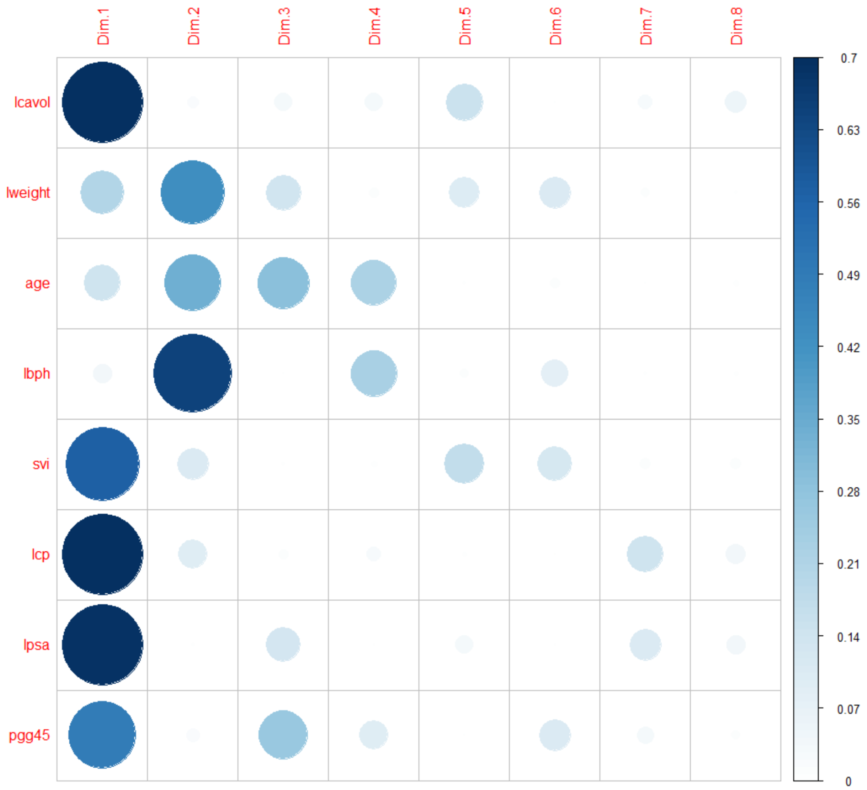


Figure 4.48: Correlation between variables and PCA.

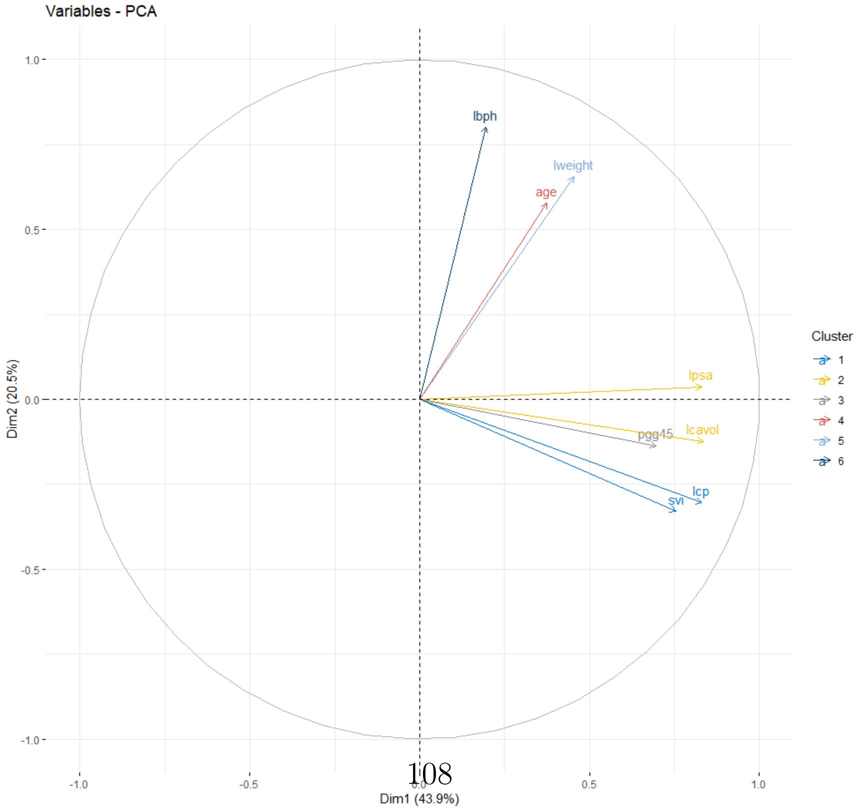


Figure 4.49: Prostate Cancer all Variances PC

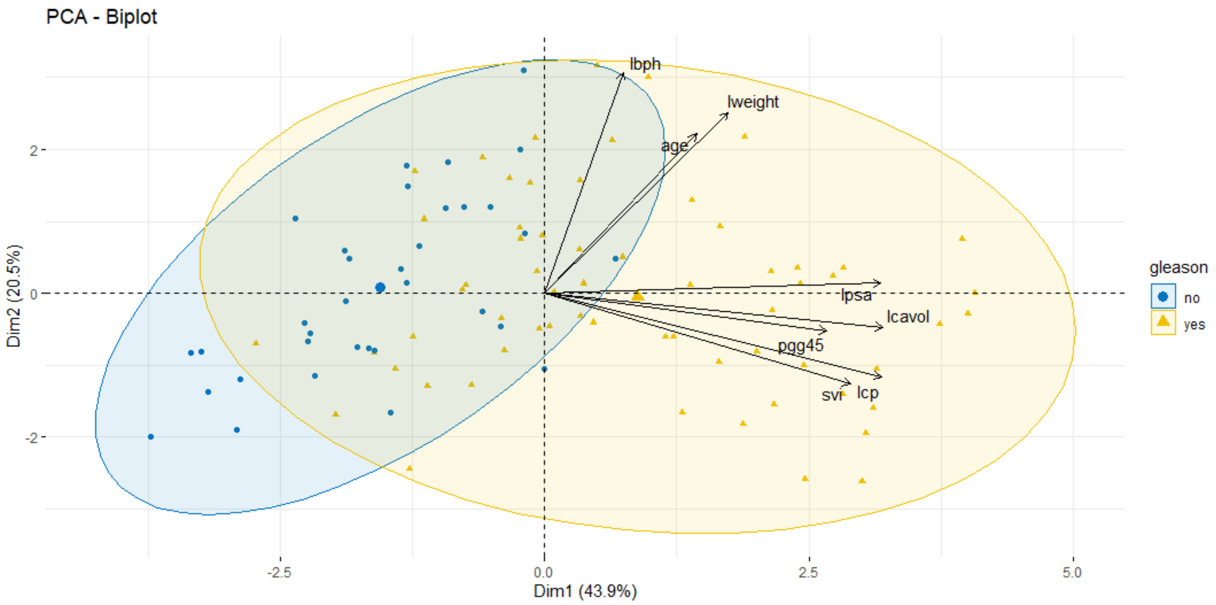


Figure 4.50: Prostate Cancer PCA Biplot

Figure 4.50 is a biplot which combines both the features and individual on the PC1 and PC2 with response of each individual in blue (low risk of prostate cancer) or yellow (high risk of prostate cancer) colored. The plot shows individuals with similar profile group together. From the figure we conclude that subject who have large cancer volume and large amount of prostate specific antigen turn to be at high risk of prostate cancer. Also old subject who have enlargement and heavy prostate turn to be at high risk of prostate cancer.

4.6 Application of Statistical learning methods

4.6.1 Logistic regression(LR):

Figure 4.51 shows the tuning parameter λ and the number of features selected by LR-ridge (left) and LR-lasso(right). We choose λ to be 0.0001020873 and 0.235835 for lasso and ridge respectively using the crossvalidation and the 1-SE rule. The selected six features and their coefficients are shown in Table 4.15 column 2 for the Lasso Logistic regression model. Column 1 of Table 4.15 show the coefficients of Ridge Logistic regression model.

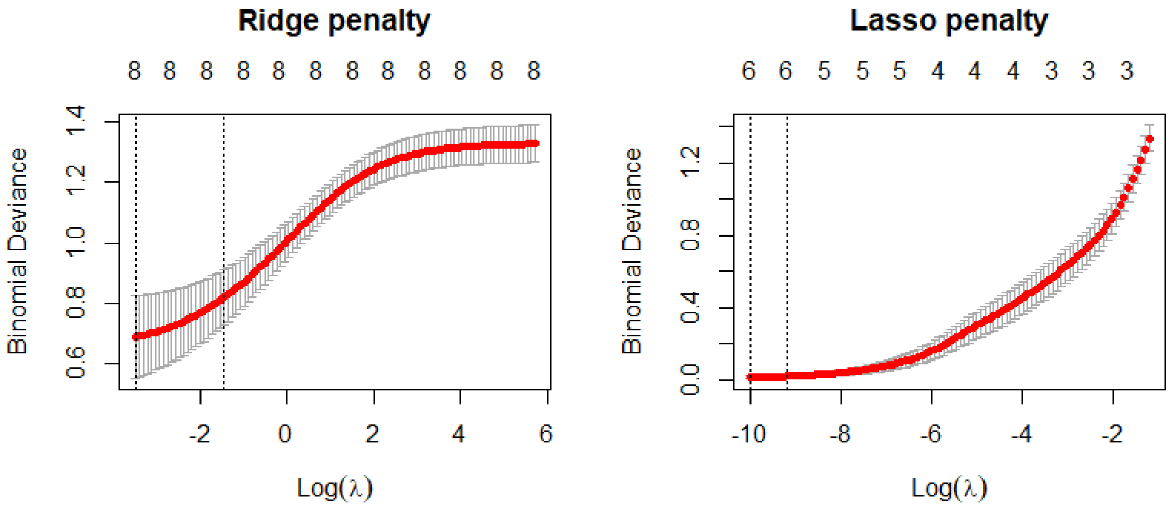


Figure 4.51: Selecting λ using 10 fold cross validation

Table 4.15: Estimate Coefficient of LR-Lasso and Ridge

Features	**LR-Ridge**	**LR-Lasso**
lcavol	0.239458698	0.37135653
lweight	0.115073840	-0.08676346
age	0.023167586	-0.01772586
lbph	0.007482273	
svi	0.474413962	
lcp	0.201077847	0.79941013
lpsa	0.291553599	0.23893327
pgg45	0.017668468	2.22722249
Pred. Accuracy	**0.8485**	**1.0000**

Figure 4.52 shows important predictors using the best predictive model with L_1 and L_2 penalties. The features pgg45, lcp, lcavol and svi, lpsa, lcavol were the top three predictors

for Lasso and Ridge LR respectively . The prediction accuracy of each model using the test data was 1.0000 for lasso and 0.8485 for Ridge.

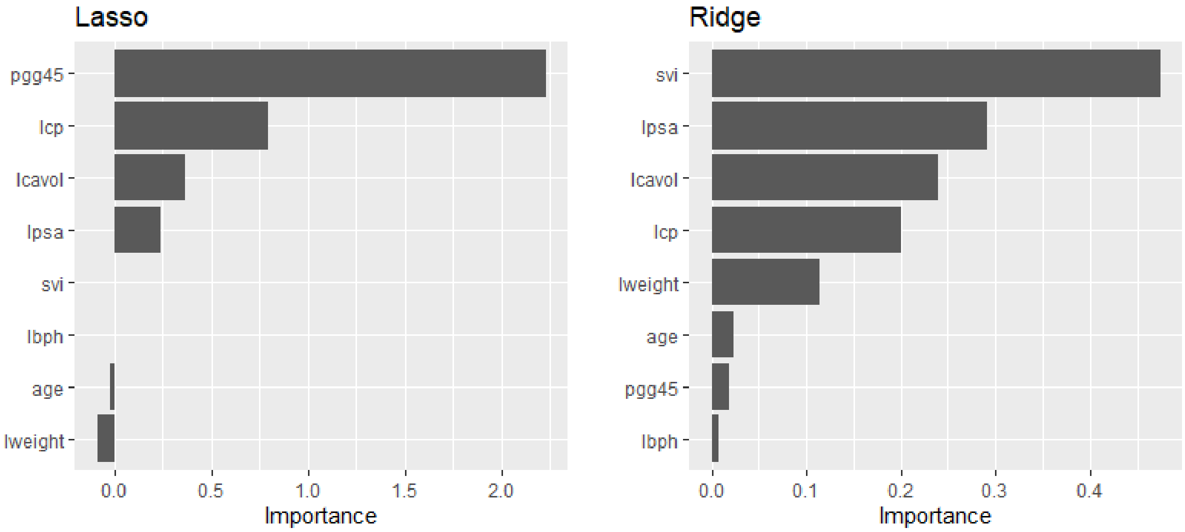


Figure 4.52: Lasso and Ridge regression variable important plot

4.6.2 Logistic Principal Component Regression(LR-PC):

The four PC components obtained in section (4.5.2) were used as predictor to fit a logistic regression to the prostate cancer data. The coefficients of the resulting model are in Table 4.16. We compared the predictive performance of PCR to the other models in Table 4.18.

4.6.3 Logistic Partial Least squares Regression(LR-PLS)

The Logistic partial least square(PLS) regression is the next model applied to the Prostate cancer data. The PLS model was fitted to the train set and used cross validation to select the number of principal components that maximizes predictive accuracy. Two component were used as shown in Figure 4.53(left). The prediction accuracy of the final model on the test set was 0.8182. We compared the predictive performance of LR-PLS to the other models in Table 4.18.

Table 4.16: LR-PC coefficient

Component	Estimate	Std. Err.	z	value	P-value
(Intercept)	7.924	2.862	2.769	0.00563	**
PC1	5.547	1.789	3.101	0.00193	**
PC2	-2.769	1.059	-2.614	0.00896	**
PC3	-3.069	1.217	-2.523	0.01164	*
PC4	2.941	1.26	2.333	0.01965	*

Null deviance = 83.591 on 63 df and Residual deviance = 24.526 on 59 df

AIC: 34.53 and Pred. Accuracy: 0.9697

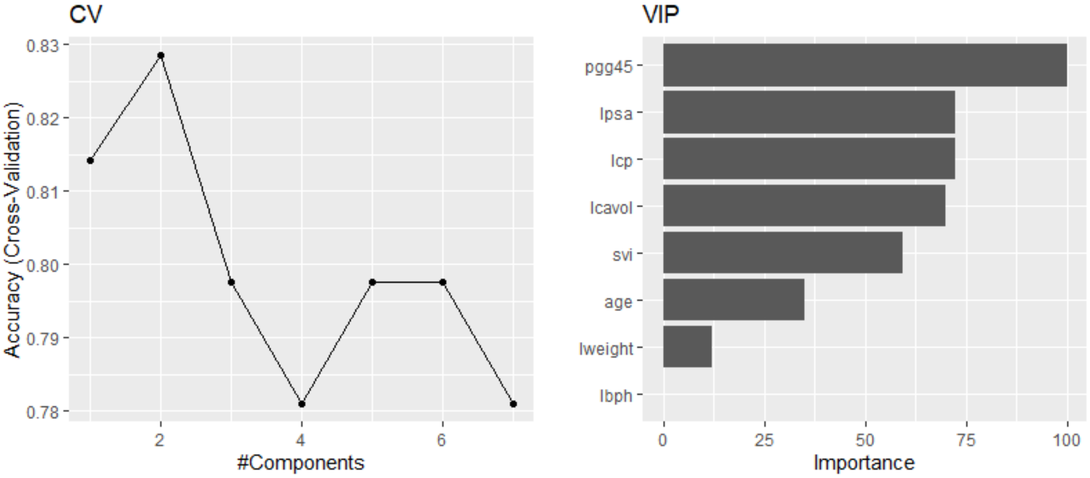


Figure 4.53: Selecting the number of component by CV(left) and variables importance plot (right).

4.6.4 Multivariate Adaptive Regression Splines(MARS):

We perform a 10-fold cross-validation grid search to identify the optimal combination of these hyperparameters that minimize prediction error. The model that provides the optimal combination includes first degree interaction effects and retains 2 terms. The cross-validated prediction accuracy for these models is displayed in Figure 4.54. The optimal model's cross-validated prediction accuracy was 1.000 on the train set . The final model gave prediction error of 1.000 on the test data.

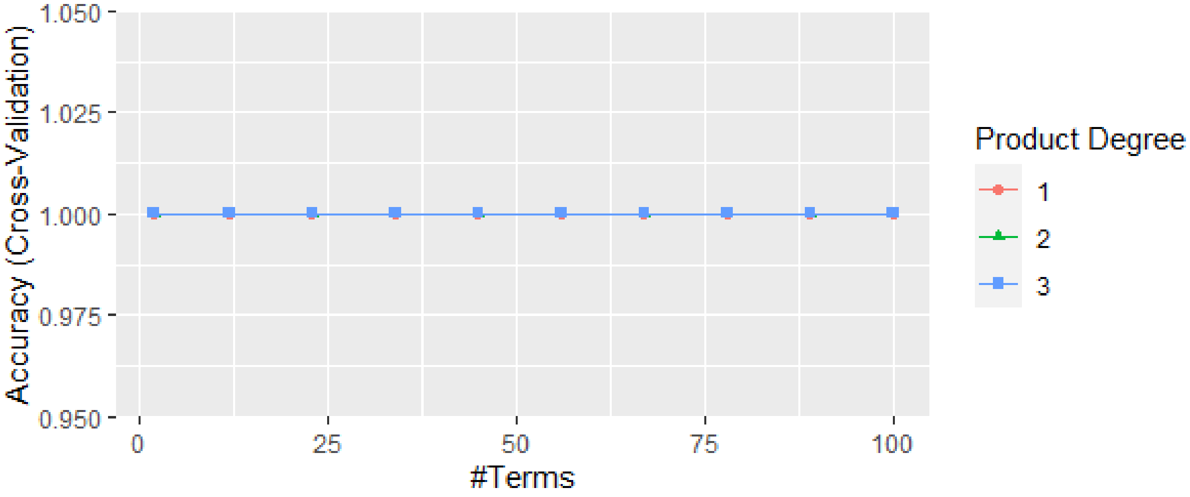


Figure 4.54: Selecting the number of component by CV

We ranked the predictors in terms of importance using the generalized cross-validation (GCV) show in Figure 4.55. From Figures 4.55, the pgg45 was the only predictor of prostate cancer. We compared the predictive performance of MARS to the other models in Table 4.18.

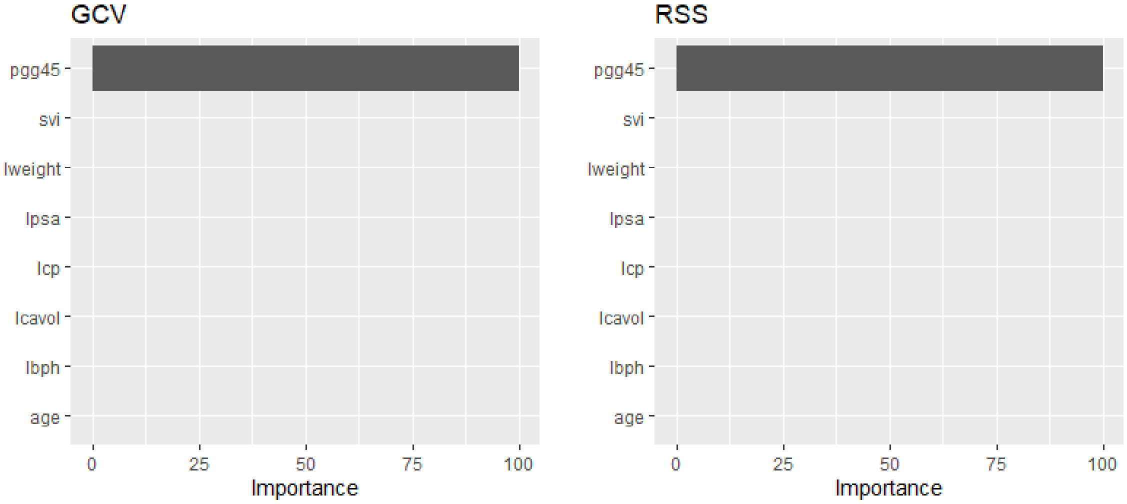


Figure 4.55: MARS variable importance plot

4.6.5 Random Forest(RF):

We trained the random forest models with 500 trees and chose $m = \sqrt{p} = 3$ as random sampled candidate at each split. The OOB error of the random forest stabilized at $B = 36$ trees as shown in Figure 4.56. The random forest model with $m = 3$ and $B = 36$ on the prostate test data gave a prediction accuracy of 1.000.

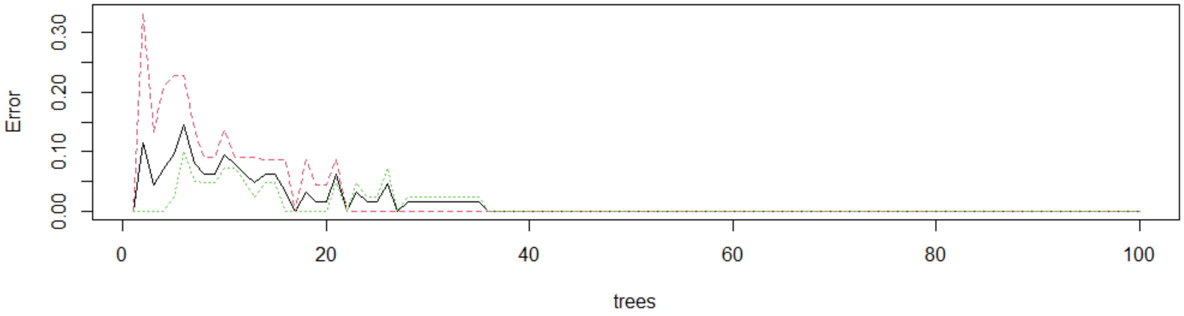


Figure 4.56: Selecting the tree size

Figure 4.57 shows variable importance ranked using the Mean Decrease Gini indices. The figure shows that pgg45, lcavol and lcp to be top three most important predictors of prostate cancer. We compared the predictive performance of RF to the other models in Table 4.18.

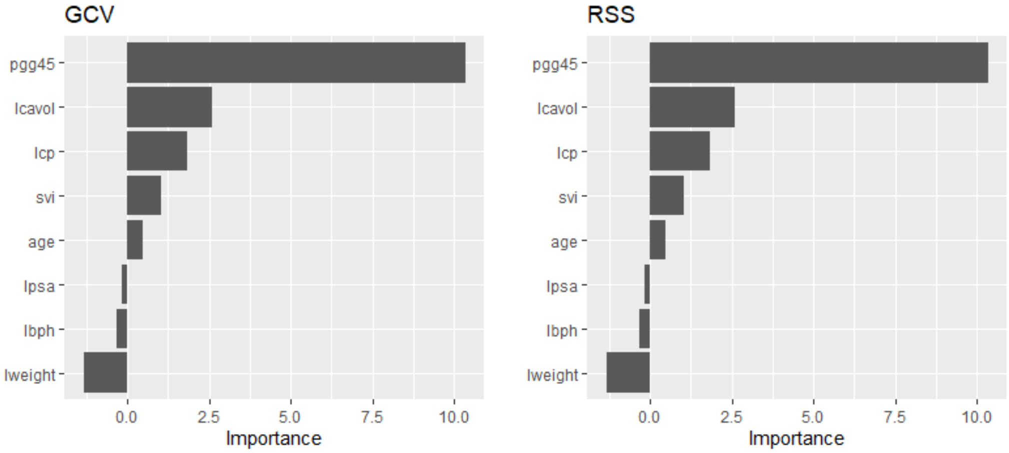


Figure 4.57: RF variable importance plot

4.6.6 Gradient Boosting

The gradient boosting algorithm with 1000 tree was fitted to to the Prostate cancer data. The optimal tree size by 10 fold cross validation was 483 as show in Figure 4.58. The blue dotted line indicates the best iteration, and the black and green curves indicate the training and cross validation error respectively.

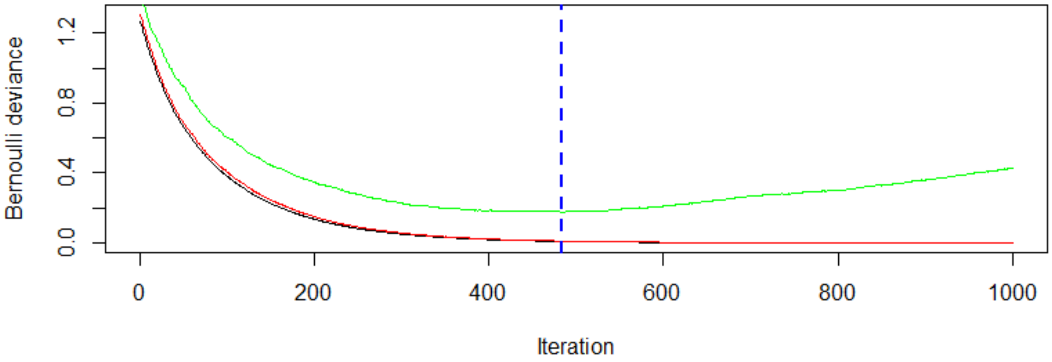


Figure 4.58: Selecting the Tree size

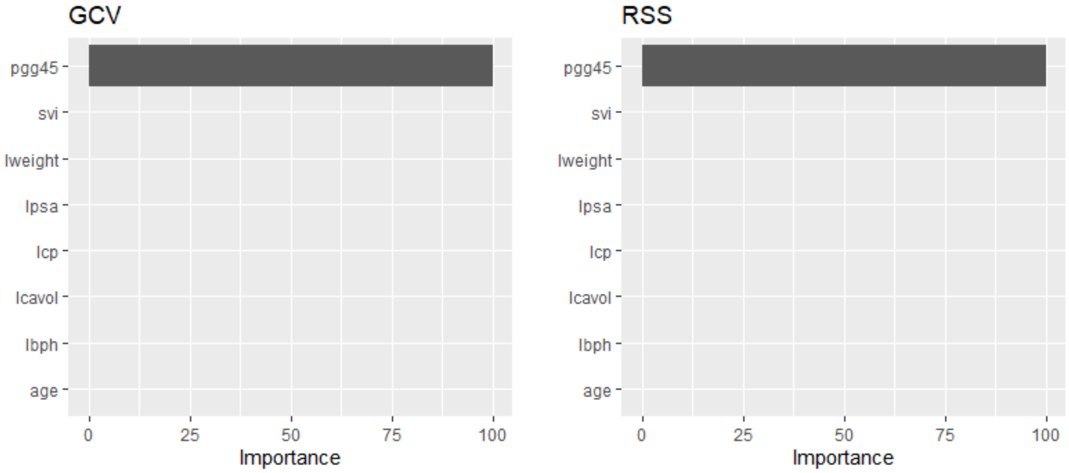


Figure 4.59: GBM variable importance plot

For the interpretation, we display the relative importance of variables in boosted trees.

From the variable importance plot in Figure 4.59, pgg45 was the most important predictors of the stage of prostate cancer. The prediction accuracy form test data and the selected model is 1.00. We compared the predictive performance of GBM to the other models in Table 4.18.

4.6.7 Support Vector Machine(SVM):

We applied the Gaussian kernel SVM to the Prostate cancer data and used 10-fold cross validation to tune two parameters γ and C. The best tuning parameters on our search grid gave gamma = 0.1215503 and cost = 2. shown in Figure 4.60.

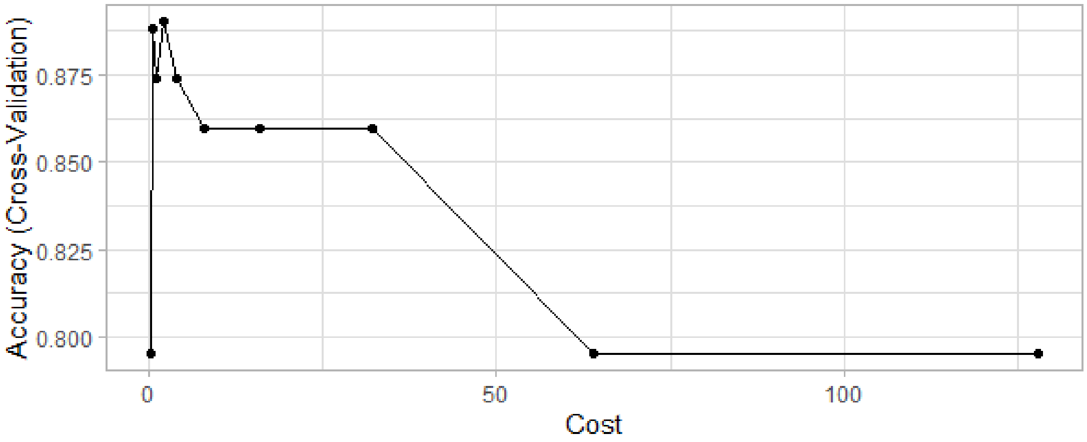


Figure 4.60: Selecting the Cost parameter

The resulting model on the test set gave a prediction accuracy of 0.8182. Figure 4.61 shows the variable importance plot from the SVM. The predictors pgg45, lcp and svi are the top three most important predictor of stage of prostate cancer. We compared the predictive performance of SVM to the other models in Table 4.18.

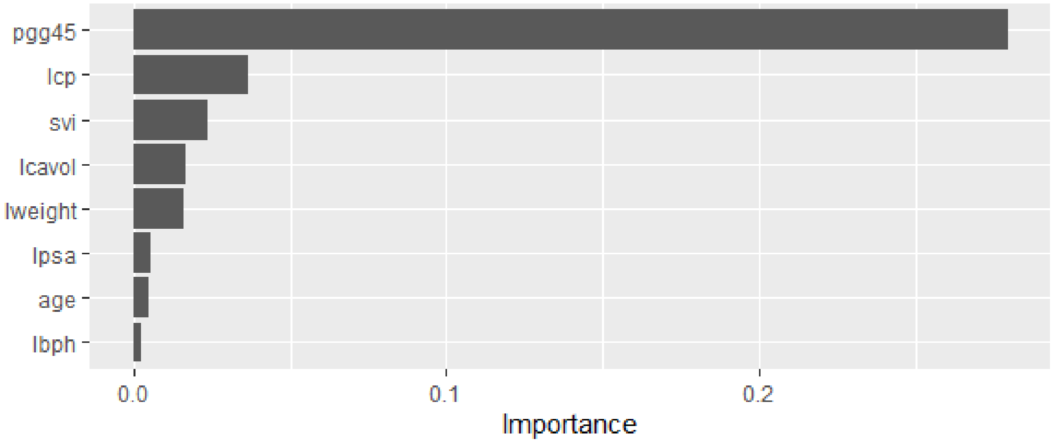


Figure 4.61: SVM variable important plot

4.6.8 Deep Learning: Feed froward Network(FFNN):

The top five FFNN model ordered by miss classifications error form the random hyper-parameter search using cross validation is shown in table 4.17.

Table 4.17: Selecting the optimal model for FFNN using miss classification error

Actv.	**Epochs**	**Hidden**	**IDR**	**L1**	**L2**	**L.Rate**	**Miss. Err**
Tanh	100	[8, 2, 2]	0.000	1.0E-6	7.7E-5	0.02	0.000
Tanh	50	[5, 5, 2]	0.000	2.1E-5	9.5E-5	0.02	0.000
Tanh	100	[8, 3, 2]	0.050	7.3E-5	7.4E-5	0.02	0.016
Tanh	100	[8, 5, 2]	0.050	4.0E-6	8.9E-5	0.02	0.016
Maxout	100	[8, 5, 2]	0.000	9.4E-5	7.8E-5	0.01	0.016

The best model from Table 4.17 row 1, fitted to the test data gave prediction accuracy of 0.909. Figure 4.62 show the pgg45 to be most important predictors of prostate cancer. We compared the predictive performance of FFNN to the other models in Table 4.18.

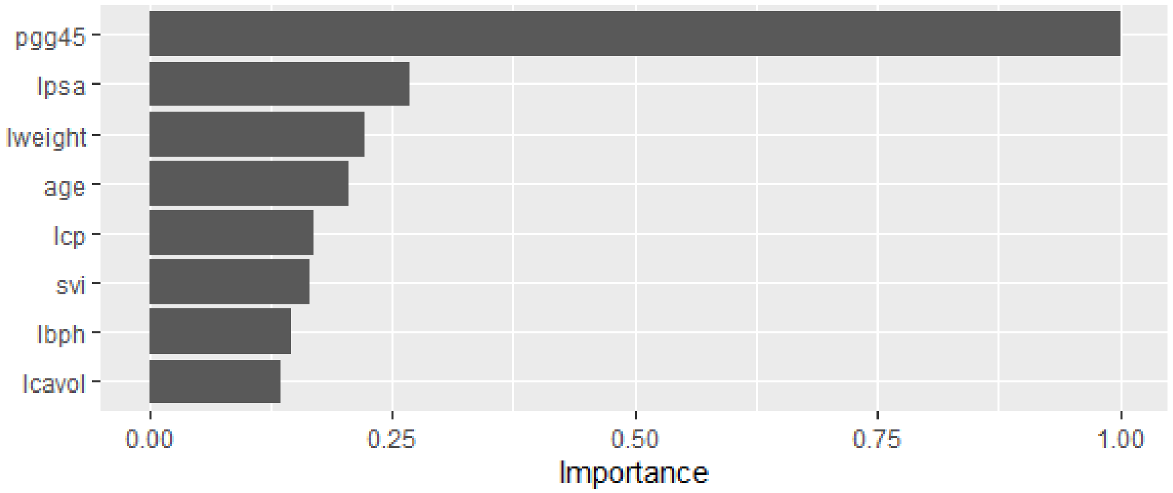


Figure 4.62: FFN variable importance plot

4.6.9 Models Summary

Table 4.18: All model applied to the Prostate Cancer data

Model	**Sensitivity**	**Specificity**	**Accuracy**	**95% CI**
LR-Ridge	0.7500	0.9048	0.8485	(0.681, 0.9489)
LR-Lasso	1.0000	1.0000	1.0000	(0.8942, 1.0000)
LR-Enet	1.0000	1.0000	1.0000	(0.8942, 1.0000)
LR-PC	0.9167	1.0000	0.9697	(0.8424, 0.9992)
LR-PLS	0.8333	0.8095	0.8182	(0.6454, 0.9302)
MARS	1.0000	1.0000	1.0000	(0.8942, 1.0000)
SVM	0.9167	0.7619	0.8182	(0.6454, 0.9302)
RF	1.0000	1.0000	1.0000	(0.8942, 1.0000)
GBM	1.0000	1.0000	1.0000	(0.8942, 1.0000)
FFNN	0.9091	0.9091	0.9091	

Table 4.18 compares the result of 10 model applied to the prostate cancer data. The models are grouped according to their similarities and learning style. i) Linear regularized models: LR-Lasso, LR-Ridge and LR-Enet. ii) Linear dimension reduction models: PCR and PLSR. iii) Non-Linear ensemble models : Random forest and Gradient Boosting. iv) Other Non-Linear models: FFNN, SVM and MARS. From Table 4.18, the linear models: LR-Lasso, LR-Enet and non linear ensemble models RF and GBM gave the best prediction accuracy of 1.00. From the variable important plots in Figures 4.52, 4.57, 4.59 and 4.68 the top risk factors of prostate cancer severity are percent of Gleason scores 4 or 5(pgg45), cancer volume (lcavol) and capsular penetration (lcp).

4.6.10 ELASTIC NET for Breast Cancer Data

CV: alpha = 0.3 lambda = 0.0273143 ,pred. accu.best model = 0.9663528

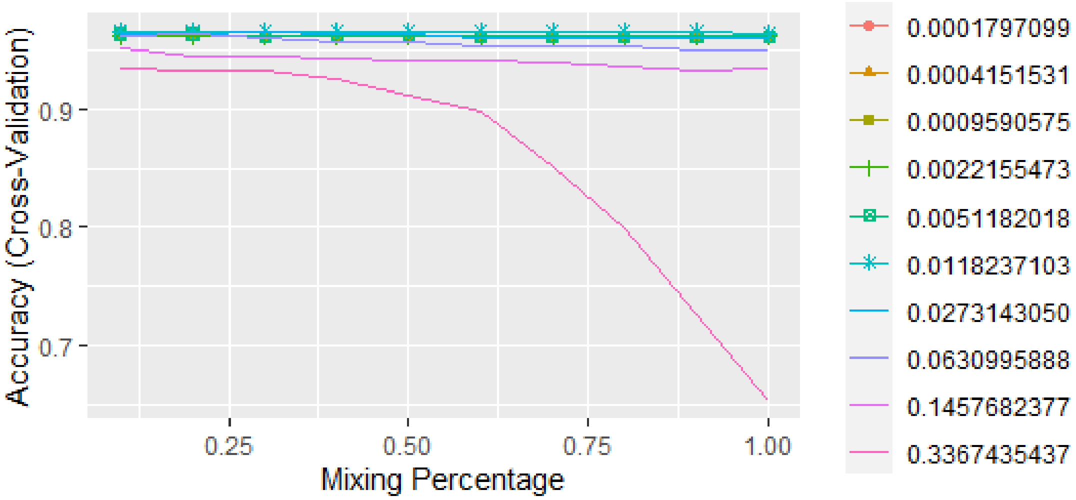


Figure 4.63: Selecting the optimal hyper parameters by CV

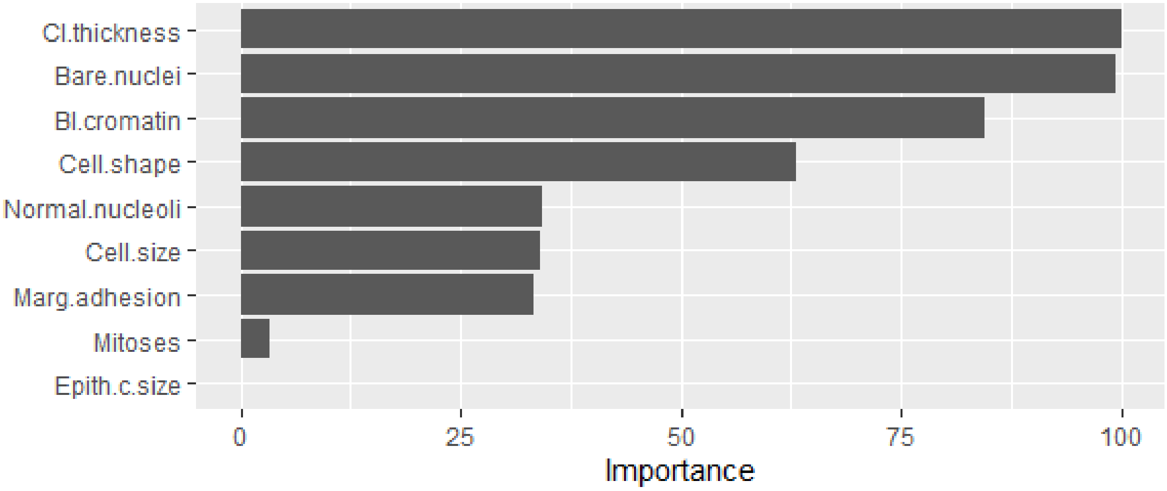


Figure 4.64: LR-Enet variable importance plot

Pred. accu.Test set =0.9707

4.6.11 ELASTIC NET for Heart Disease Data

CV alpha =0.1 lambda = 0.2119891 ,pred. accu. best model= 0.8342982

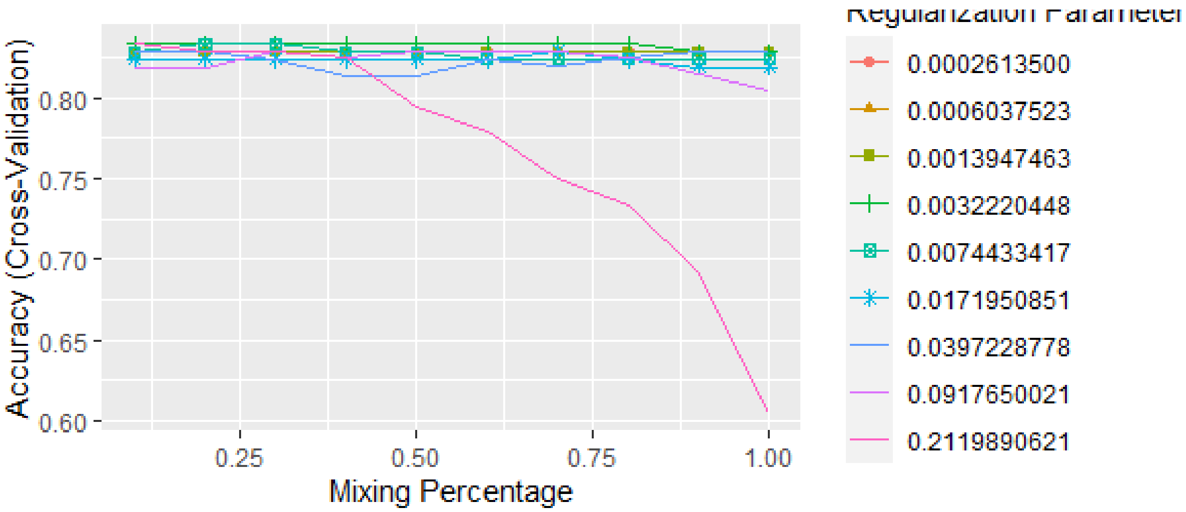


Figure 4.65: Selecting the optimal hyper parameters by CV

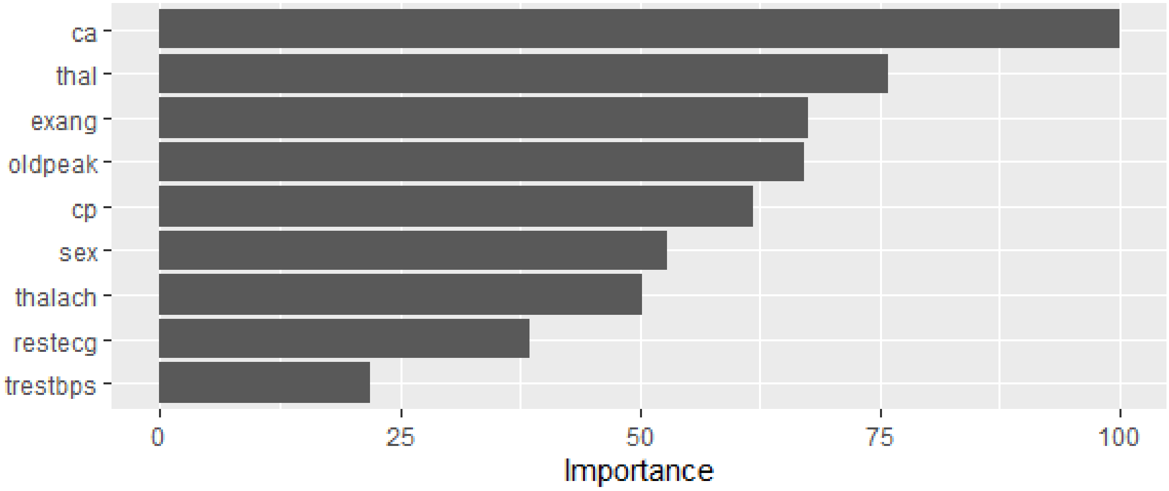


Figure 4.66: LR-Enet variable importance plot

Pred. accu.Test set = 0.8384

4.6.12 ELASTIC NET for Prostate Cancer Data

cv alpha =1 lambda = 0.0003251038 ,pred accu.best model = 0.9666667

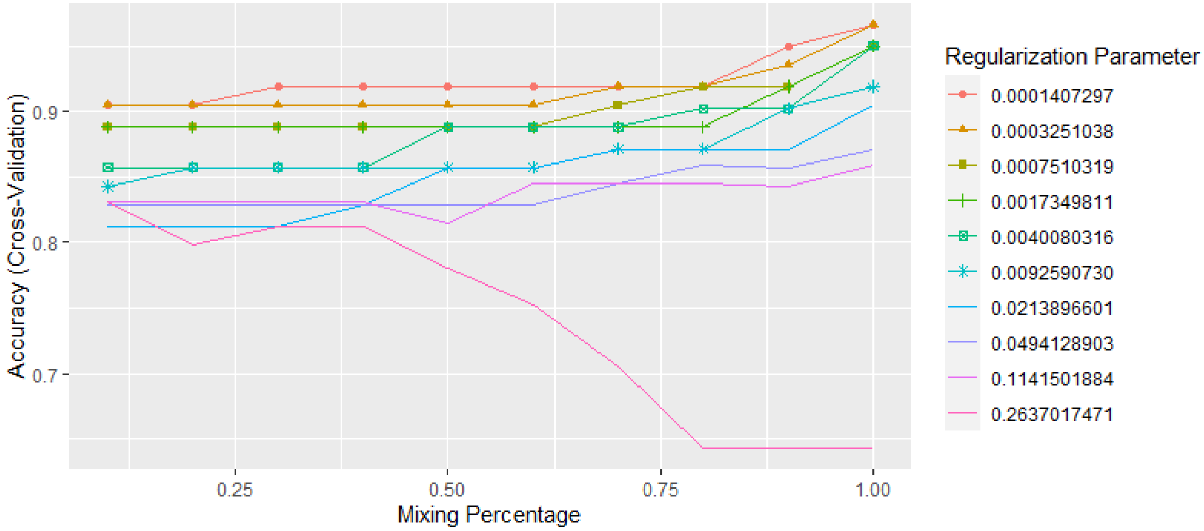


Figure 4.67: Selecting the optimal hyper parameters by CV

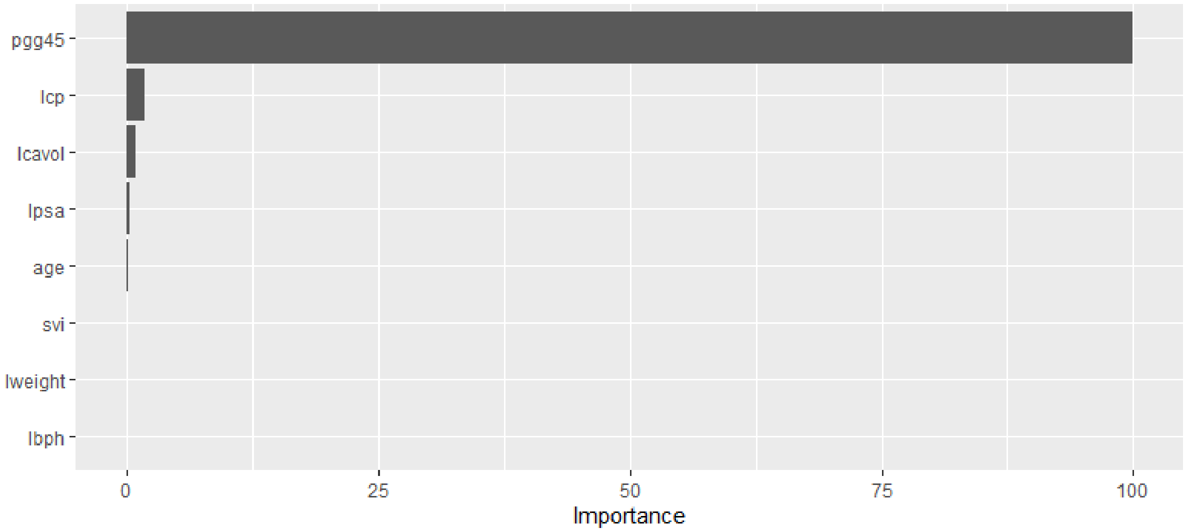


Figure 4.68: LR-Enet variable importance plot

Pred. accu.Test set = 1.000

Chapter 5

Concluding Remarks

The predictive performance of 10 statistical learning models were investigated using three medical data set, including breast cancer, heart disease and prostate cancer. The objective is to use these models to predict the presence of breast cancer, heart disease and severity of prostate cancer in patients in order to select the best predictive model(s) for these diseases. We also used the models to identify risk factors that contribute significantly to these diseases.

The models considered include; Logistic regression with L_1 and L_2 penalties, Principal component regression(PCR), Partial least squares regression(PLS), Multivariate adaptive regression splines(MARS), Support vector machine(SVM-GK), Random Forest(RF), Gradient Boosting Machines(GBM), Elastic Net(Enet) and Feed Forward Neural Network(FFNN).

The models are grouped according to their similarities and learning style. i) Linear regularized models: LR-Lasso, LR-Ridge and LR-Enet. ii) Linear dimension reduction models: PCR and PLSR. iii) Non-Linear ensemble models : Random forest and Gradient Boosting. iv) Other Non-Linear models: FFNN, SVM-GK and MARS. The methodology is not new, however the major contribution of this work comes in the realm of applications. The methodology was applied to three medical data set: Breast Cancer, Heart disease and Prostate cancer. In all the applications the methodology provides insight into predictive performance of these model and the risk factors of these diseases.

To improve models performance and generalization of each method, we applied early stopping, dropout and removed non-significant variables to avoid over fitting. We then used cross-validation to select the best tuning parameters for each model. Once the predictive models were built, we checked their efficiency in the diagnosis and prognosis of disease using test data. In order to obtain the most efficient model, we also compared the prediction accuracy, sensitivity, and specificity to find the best classifier (see Tables 4.6, 4.12 and 4.18).

The result show the non-linear models; SVM-GK, RF and FFNN gave the best prediction accuracy of 0.9756 (Table 4.6) for the breast cancer data. From the variable important plots in Figures 4.21, 4.25, 4.26 the top risk factors of breast cancer are uniformity of cell-size (Cell.size), uniformity of cell shape (Cell.shape), bare nuclei (Bare.nuclei) and bland chromatin (Bl.cromatin). This result is in line with previous work of Padmavathi, J.(2011) [48], Vikas et al.(2014) [41] and Mariani el at. (2019) [47] for their studies on breast cancer prediction. In their work Vikas et. al compared several supervised learning classifiers, such as Naive Bayes, Support Vector Machine (SVM-RBF), Neural Networks, and Decision trees to find the best classifier in breast cancer datasets. Their study showed SVM-RBF kernel was the most accurate classifier. Mariani el at. in their paper compared the predictive ability of five ML algorithms using breast cancer data set. Using prediction accuracy and the Receiver Operating Characteristic (ROC) curve as the perfomance criterion. They showed Random Forest (RF) to be the best predictive model of breast cancer. Finally in her work Padmavathi, J. compared feed forward neural network (FFNN) with one hidden layer to commonly used Multilayer Perceptron network model and the classical logistic regression. The sensitivity and specificity of both neural network models had a better predictive power compared to logistic regression. Hence our results of breast cancer prediction reinforced previous results by different methodology.

The linear models; LR-PLS, LR-PC ,LR-Ridge and LR-Enet was preferred for heart disease data with prediction accuracy of 0.8384 (Table 4.12). From the variable important plots in Figures 4.33, 4.34, 4.65 the top risk factors of heart disease are number of major vessels (ca), Thalassamia (Thal) and exercise induced angina (Exang). This result is consistent with previous work of Dwivedi AK.(2018)[45] and Mariani el at (2019)[47]. In the paper Dwivedi AK.(2018) evaluated the performance of six machine learning techniques for predicting heart disease. The methods were validated using tenfold cross validation and assessed the performance using receiver operative characteristic curve. The highest classification accuracy of 85 % was reported using logistic regression. Also Mariani el at. in their paper compared the predictive ability of five ML algorithms using heart disease data set. Using prediction accuracy and the Receiver Operating Characteristic (ROC) curve as the performance criterion. They showed Principal Component Regression (PCR) provided the best predictive performance for heart disease data set.

Finally a mixture of linear models and non-linear models; LR-Lasso, LR-Enet, RF and GBM gave the best prediction accuracy of 1.00 for the Prostate cancer data (Table 4.18). From the variable important plots in Figures 4.52, 4.57, 4.59 and 4.68 the top risk factors of prostate cancer severity are percent of Gleason scores 4 or 5(pgg45); cancer volume (lcavol) and capsular penetration (lcp).

In general no particular model, class of models or learning style dominated predictability for all the three data set. Non-linear models was prefered for Breast cancer data, a linear model for heart disease and a mixture of linear and non linear models for prostate cancer .

References

[1] Hastie T., Tibshirani, R., Friedman, J.(2009), *The Elements of Statistical Learning:Data Mining, Inference,and Prediction*, 2nd edition,Springer,2008.

[2] Frank, I. E. and Friedman, J. H. (1993), *A Statistical View of Some Chemometrics Regression Tools.*, Technometrics, $35 : 109 - 148$

[3] Tibshirani, R.(1996), *Regression shrinkage and selection via the Lasso*, Journal of the Royal Statistical Society, Series $B, 58 : 267 - 288$

[4] Hoerl, A. E. and Kennard, R.W. (1970), *Ridge regression: biased estimation for nonorthogonal problems.*, Technometrics, $12 : 55 - 67$.

[5] Casella, G. (1980), *Minimax Ridge Regression Estimation.* , Annals of Statistics, $8(5) : 1036 - 1056$

[6] Efron, B., Hastie, T.., Johnstone, I., and Tibshirani, R. (2004) , *Least Angle Regression (with discussion)* , Annals of Statistics, $407 - 499$

[7] CRAVEN, P. and WAHBA, G. (1979), *Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation.*, Numer. Math. $31\ 377 - 403$

[8] Koh, K., Kim, S.J. and Boyd, S. (2007), *An interior-point method for large-scale L_1-regularized logistic regression*, Journal of Machine Learning Research 8: 1519–1555.

[9] Gentle, J. E. (2009), *Computational Statistics.*, Fairfax, VA: Springer.

[10] Jolliffe, L.T. (2002), *Principal Component Analysis*, Second edition(2002), Springer, $167 - 195$.

[11] artnome.com, *https://www.artnome.com/news/2018/11/8/inventing-the-future-%of-art-analytics*

[12] Kononenko ,I.Hong, S.J. (1997), *Attribute selection for modelling*, Future Generation Computer Systems,Elsevie-1997

[13] Breiman, L. (2001), *Random forests*, Machine Learning,Kluwer Academic Publishers 45,5–32(2001)

[14] Breiman, L. (1996), *Bagging Predictors*, Machine Learning,Publishers:Kluwer Academic 24,123-140(1996)

[15] Ho, T.K., *The random subspace method for constructing decision forest*, Pattern Analysis and Machine Intelligence,Publisher:IEEE Transactions 20,8 (1998)

[16] Friedman J.H., *Multivariate adaptive regression splines*, The annals of statistics, 1991 - JSTOR 1-67

[17] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013)., *An Introduction to Statistical Learning with Applications in R.*, Springer()2013). ISBN-13: 978-1461471370

[18] IBM Knowledge Center, *https://www.ibm.com/support/knowledgecenter/ca/SS3RA7-sub/modeler-mainhelp-client-ddita/clementine/svm-howwork.html*

[19] Gordon G.,Tibshirani R. (2012), *Optimization, 2012: Karush-kuhn-tucker conditions.*,

[20] Murray, W., Gill, P. and Wright, M. (1981), *Practical Optimization.*, Academic Press.

[21] Max K., Johnson K.(2013), *Applied Predictive Modeling.*, springer Science Business Media New York (2013)

[22] Berk,R.A.(2016), *Statistical Learning from a Regression Perspective.*, Springer International Publishing Switzerland (2016).

[23] Tibshirani R.J (2015), *A General Framework for Fast Stagewise Algorithms.*, Journal of Machine Learning Research 16 (2015) 2543-2588.

[24] Bishop, C. (1995), *Neural Networks for Pattern Recognition.*, Oxford University Press, Oxford(1995).

[25] Ripley, B. (1996), *Pattern Recognition and Neural Networks.*, Cambridge University Press(1996).

[26] Titterington, M. (2010), *Neural Networks.*, Wiley Interdisciplinary Reviews:Computational Statistics,2(1), 1–8

[27] Amini, A. (2019), *Introduction to Deep Learning.*, MIT 6.S191

[28] Rumelhart, D, Hinton G., Williams, R. (1986), *Learning Internal Representations by Error Propagation.In Parallel Distributed Processing: Explorations in the Microstructure of Cognition.*, The MIT Press.

[29] kaggle (2020), *https://www.kaggle.com/c/prostate-cancer-grade-assessment/overview*

[30] Stamey, T., Kabalin, J., McNeal, J., Johnstone, I., Freiha,F., Redwine, E.and Yang, N. (1989), *Prostate specific antigen in the diagnosis andtreatment of adenocarcinoma of the prostate II radical prostatectomytreated patients.*, Journal of Urology16: 1076–1083

[31] ISUP (2020), *https://isupweb.org/isup/*

[32] PCF (2020), *https://www.pcf.org/about-prostate-cancer/diagnosis-staging-prostate-cancer/gleason-score-isup-grade/* Prostate, Cancer, Foundation

[33] Mallows, C. L. (1973), *Some comments on Cp* Technometrics, 15: 661–675

[34] Akaike, H. (1973)., *Information theory and an extension of the maximum likelihood principle.* In Procedings of Second International Symposium on Information Theory, (eds B. N. Petrov and F.Csaki). Budapest: Akademiai Kiado, 267–281.

[35] Schwarz, G. (1978), *Estimating the dimension of a model* The Annals of Statistics, 6: 461–464.

[36] Zou, H. and Hastie, T. (2005), *Regularization and variable selection viathe elastic net* Journal of the Royal Statistical Society Series B.67(2): 301–320.

[37] Pradesh,A.(2011)), *Analysis of Feature Selection with Classification, Breast Cancer Datasets* Indian J. Comput. Sci. Eng, 2(5), 756-763.

[38] Dwivedi, A.K. (2018), *Performance evaluation of different machine learning techniques for prediction of heart* disease, Springer, 29(10), 685-693.

[39] Kahramanli, H., Allahverdi, N. (2008), *Design of a hybrid system for the diabetes and heart diseases* Elsevier (Expert Systems with Applications), 35(2), 82-89

[40] ACS (2017), *American Cancer Society. Breast Cancer Facts and Figures 2017-2018* (https://www.cancer.org/)

[41] Chaurasia, V., Pal, S. (2014), *Data Mining Techniques : To Predict and Resolve Breast Cancer* Survivability,IJCSMC, 3(1), 10-22

[42] Thorsten J. (1999), *Transductive Inference for Text Classification Using Support Vector Machines* Icml, 99,200-209, doi:10.4218/etrij.10.0109.0425.

[43] Pradesh, A. (2011), *Analysis of Feature Selection with Classification : Breast Cancer Datasets* Indian J. Comput. Sci. Eng, 2(5), 756-763(2011)

[44] CDC (2020), *https://www.cdc.gov/heartdisease/facts.htm*

[45] Dwivedi, A.K. (2018), *Performance evaluation of different machine learning techniques for prediction of heart disease* Springer,29(10), 685-693.

[46] Kahramanli, H., Allahverdi, N. (2008), *Design of a hybrid system for the diabetes and heart diseases* Elsevier(Expert Systems with Applications),35(2), 82-89

[47] Mariani M. C.,Tweneboah O.K., Bhuiyan M. (2019), *Supervised machine learning models applied to disease diagnosis and prognosis* AIMS Public Health. 2019; 6(4): 405–423.

[48] Padmavathi, J. (2011), *A Comparative study on Breast Cancer Prediction Using RBF and MLP.* International Journal of Scientific & Engineering Research, Volume 2, Issue 1, January-2011 ISSN 2229-5518

[49] UCI machine learning repository, *https://archive.ics.uci.edu/ml/datasets/Breast +Cancer+Wisconsin+(Diagnostic)*

Curriculum Vitae

Francis Biney was born in Ghana on January 17, 1981. After graduating from St. Mary Secondary School in 2000, he entered Kwame Nkrumah University of Science and Technology in 2001 in Ghana. He graduated with his bachelors degree in Mathematics in May 2005 with honors. He worked temporary with the Statistics Department of Internal Revenue Service in Ghana and taught in St. Mary Secodary School until June 2010 .

In the fall of 2010 , he entered the Graduate School of The University of Texas at El Paso. While pursuing a masters degree in Statistics, he worked as a Teaching Assistant. He planing to continue studying for a PhD. after completing his masters degree.

Permanent address: 12973 Immanuel Vista Dr.
El Paso, Texas 79938