Supervised Learning Methods with Applications in X-ray Image Classification

Kabwe Mpundu

January 17, 2022

Abstract

This paper will describe how classification techniques build systems of labelling new data based on information learned from categorically labelled examples. Classification theory is discussed and used to describe traditional classifiers. These include the discriminant analyses, logistic regression and the nearest neighbour method. The classifiers are trained on data, tested with new data and their performance compared. From here methods in which to analyse images are demonstrated with the use of more modern classifiers. Tree-based classifiers and maximum margin classifiers are described. The famous MNIST data set of handwritten digits is then simplified to show how images can be processed and how these modern classifiers perform with simplified data. The last part of the paper will show how these methods can compare on a more complex set of images. The X-ray images of healthy people, Covid-19 infected patients and viral pneumonia patients will be used. Some general comments about classification methods' ability to learn will be presented considering the evidence from image classification.

Declaration

This piece of work is a result of my own work except where it forms an assessment based on group project work. In the case of a group project, the work has been prepared in collaboration with other members of the group. Material from the work of others not involved in the project has been acknowledged and quotations and paraphrases suitably indicated

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> *. . . I would say that fair play must be given to the machine. Instead of it giving no answer we could arrange that it gives occasional wrong answers. But the human mathematician would likewise make blunders when trying out new techniques. . . In other words then, if a machine is expected to be infallible, it cannot also be intelligent. There are several mathematical theorems which say almost exactly that. But these theorems say nothing about how much intelligence may be displayed if a machine makes no pretence at infallibility.*
> (Alan Turing)

Statistical analysis and modelling are the backbones of scientific study. This method of formulating knowledge from the real world requires a means of summarising and generalising data. The scientific method's basic idea is to isolate a phenomenon and then describe its workings mathematically. These mathematical formulations can be used in new scenarios. For example, an apple fell on Isaac Newton's head, which inspired his study of gravity and the laws of motion. With these insights, extraterrestrial travel is now possible. This ability to learn from a sample of measurements and then extend to new scenarios is called statistical learning (D. Michie & Taylor 1994).

Extending prior knowledge to novel cases in the statistical fashion was pioneered by the Reverend *Thomas Bayes*. He believed observational evidence could help describe some unknown parameter with conditional probabilities, hence the beginning of **classification theory**. Classification theory is a system of learning from labelled examples of data. At the point when a new classification is required, unlabelled data can be classified using the properties modelled in the labelled examples. Just the same way children are taught to identify letters by their sounds, statistical classification involves a teacher training a model. In the future, the child is expected to know the differences between letters based on their sound, and the teacher will test the students' ability and correct them where they are wrong. The same process of training, testing and correcting applies

to statistical learning and when this happens, it is called **supervised learning**.

There are many forms of learning and some of these include **regression** and **unsupervised learning**. They differ only slightly from classification. During training their labels may not exist but instead be a continuous scale as in regression, or the labels may be unknown entirely, as with the case of unsupervised learning. The main emphasis of this paper is classification; however, the use of some forms of regression and unsupervised learning are presented as useful in aiding the supervised learning process.

1.1 Aim

This paper will describe how classification techniques build systems of labelling new data based on information learned from categorically labelled examples. The first goal will be to describe the theory behind statistical learning. The second goal will be to demonstrate some applications of the theory mathematically and with some data examples. The third goal will be to introduce methods in which images can be modelled and later classified. The last goal will be to compare the different classification methods alongside the different image preparation methods.

1.2 Objectives

To accomplish the aim of the project the following objectives must be achieved.

1. Define Supervised Learning from a Bayesian perspective.
2. Show applications of classification with discriminant analyses, logistic regression and the K-nearest neighbours models.
3. Introduce the idea of image processing and classification using classification trees and the support vector machines.
4. Apply the knowledge learned to an X-ray image classification problem.
5. Comment on the suitability of statistical classification methods for use in the real world.

1.3 Data Methods and Notation

For classification in this paper, the labels will be called classes denoted by Π_i, $i \in 1, 2, 3, ..k$. A slight variation will appear in Chapter 5, where the classes are labelled

Π_2 and Π_1. The data used to build models will be called a feature space, $\mathbf{X}$ and an observation will be denoted as $\mathbf{x}$. A training subset will be used to build the models, and a testing subset will simulate new data for evaluation purposes.

1.4 The Organisation of Paper

Chapter 2 will cover the basic classification problem. It will follow by describing methods of measuring the performance of classifiers. Finally, the chapter will discuss ways and means of improving model performance.

Chapter 3 will discuss the traditional classification algorithms developed in the earlier parts of the 20^{th} century. The first section of the second chapter will discuss linear and quadratic discriminant analysis. It will follow this with a discussion of logistic regression and ultimately close with the k-nearest neighbour classifier.

Chapter 4 will cover tree-based classifiers as covered by Leo Breiman. The chapter will start will classification trees and show some examples. It will then describe the ideas behind the random forest classifiers. And there will be some data examples.

Chapter 5 will cover the support vector classifiers described by Vladimir Vapnik in 1992. The chapter will firstly, cover the basic principles. Followed by describing methods that can be used to deal with non-linearly separable data. Lastly, the chapter will include some examples with data.

Chapter 6 will cover X-ray image classification. In the beginning, the data will be described and analysed. Secondly, the classifiers discussed in the paper will be applied to different subsets of the data. These subsets will come from methods of dimension reduction covered in chapter 5 and in some level three university courses. Lastly, the results of these classifiers will be compared using tables of results.

Chapter 7 will provide an overall summary of the problems covered in this paper. Naturally, the solutions to these problems will be covered. To conclude some recommendation for future studies will be highlighted.

Chapter 2

Bayesian approach to Classification

All classification problems can be summarised as a Bayesian classification problem. Which are the means of extending prior knowledge to an unobserved distribution. In this chapter, we define the Bayesian rule, classifier and error. From this, the mathematical goals of classification will be defined. Finally, methods to measure the performance of classifiers will be discussed.

2.1 The basic classification problem

The basic problem here is to use the knowledge learnt in a training sample, to assign labels to unseen observations. A good classifier should be able to correctly label new observation by efficiently extracting insights from the training sample. However, the idea of the **Bayes error** arises, this error is equivalent to the true error of the population, it is when the training examples contain errors within themselves. This can arise from measurement errors, subjective data, messy data etc. Thus, the true error is common in nature, thus, it is the best any classifier can do (Fukunaga 1990).

However, it is possible that the error computed by a classifier is lower than the true error, thus, a means of validating the results is essential. Throughout this paper, the data will be randomly split into a train and test set, for training and evaluation, respectively. This will be done to ensure performance measurements are reliable and are not a result of overfitting. Overfitting is when the model is fitted with the errors in one data sample and thus, when presented with new data, the accuracy rate falls significantly as the errors were specific to the training sample. In Figure 2.1 the process in which data is handled to be used for both model building and evaluation is shown. In this diagram

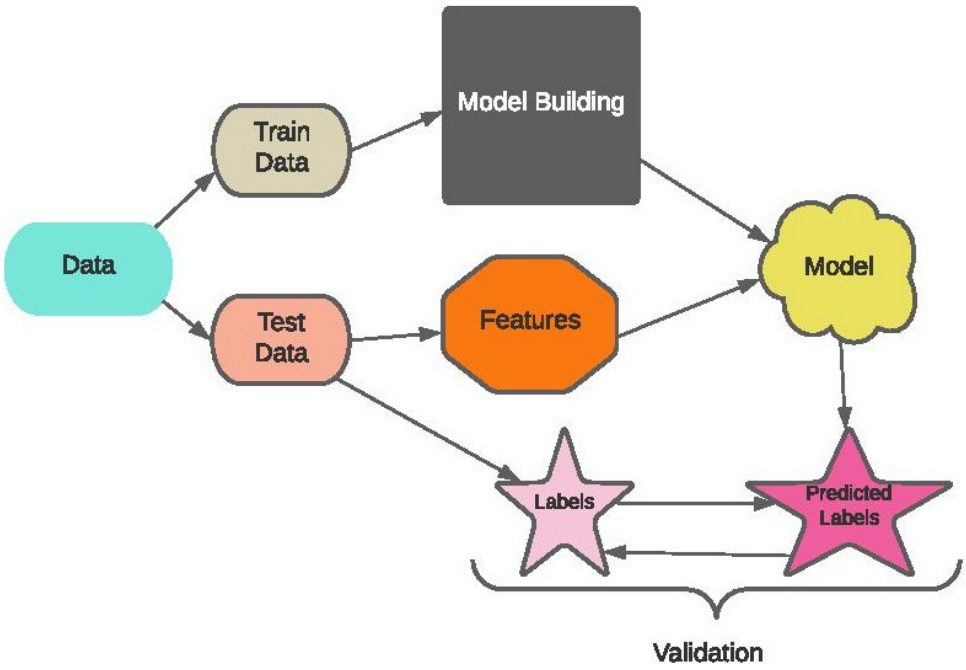

Figure 2.1: Classification process for model buildings and validation

data is randomly split into train and test set, the train set is used for model building and the resulting model is used to predict labels for the train set features. The predicted labels are then compared to the original labels as a means of validating the training of the model.

2.2 Model building using Bayes theorem

Using the Bayes theory, the Bayes rule can be defined. Models which use this rule are **Bayes rule classifiers**, all models in this paper will follow this principle. Basically, the models are all alternative means of estimating this Bayes rule classifier (Weiss & Kulikowski 1991). Thus, to begin our journey towards classifying medical images we must first describe the Bayes rule classifier.

The Bayes rule,

$$Prob(\Pi_i|\mathbf{x}) = \frac{Prob(\mathbf{x} \mid \Pi_i)Prob(\Pi_i)}{Prob(\mathbf{x})}, \quad i = 1, 2, ..., k, \tag{2.1}$$

maps the probabilities of a class label Π_i, $i = 1, 2, ..., k$, belonging to an observation $\mathbf{x}$ from the probability of observing that class and the probability of that observation given

it belongs to that class. For example, the probability of a smoker having cancer can be linked to the probability of any person having cancer and the probability a person with cancer is a smoker. In the example, the probability any person has cancer is called the prior probability,

$$Prob(\Pi_i) = Prob(\mathbf{X} \in \Pi_i) = \pi_i, \quad i = 1, 2, ...k, \tag{2.2}$$

and the probability that a cancer patient is a smoker is the conditional class probability distribution,

$$Prob(\mathbf{x} \,|\, \Pi_i) = Prob(\mathbf{X} = \mathbf{x} | \mathbf{X} \in \Pi_i) = f_i(\mathbf{x}), \quad i = 1, 2, ...k. \tag{2.3}$$

$Prob(\mathbf{x})$ will be the constant for any given observation, thus, the Bayes theorem,

$$Prob(\Pi_i|\mathbf{x}) = Prob(\mathbf{X} \in \Pi_i | \mathbf{X} = x) = \frac{f_i(\mathbf{x})\pi_i}{Prob(\mathbf{x})}, \quad i = 1, 2, ...k \tag{2.4}$$

can be used to create the *Bayes' rule classifier*,

$$f_i(\mathbf{x})\pi_i = \max_{1 \le j \le K} f_j(\mathbf{x})\pi_j, \tag{2.5}$$

which will assign the class Π_i to $\mathbf{x}$, given the conditions above are satisfied. In the case of our smoker, the probability of them having cancer versus the probability of them not having cancer are compared. If the probability of having cancer is higher than not having cancer a Bayes rule classifier will predict the person has cancer and vice versa. However, this may not always be right, as a person may seem to have a low risk of cancer but actually have it, this inherent risk is the Bayes error,

$$1 - Prob(\Pi_i|\mathbf{x}) = 1 - \max_{1 \le j \le K} \frac{f_i(\mathbf{x})\pi_i}{Prob(\mathbf{x})}. \tag{2.6}$$

Thus, classifiers all attempt to model the conditional class probability distributions from a feature set. These distributions can be weighted, and a new observation can be labelled based on the highest weighted probability density. Figuratively, this means a model will compare information about cancer patients it has observed to the information about the individual, based on this it can predict the person's condition. It is possible it for mistakes to happen, because of inherent errors in the data, maybe data about smoking is too general, i.e., smoking socially versus smoking a pack a day or living with a smoker, but identifying as a non-smoker. Errors are possible and should be expected when dealing with real world data, and hence section 2.3 discusses ways in which errors are recorded and how correct-observations can be treated.

2.3 Measuring Classification Power

As discussed, the classification goal is to learn from samples and use the model for new unseen cases. The error rate,

$$\text{error rate} = \frac{\text{number of errors}}{\text{number of cases}}, \tag{2.7}$$

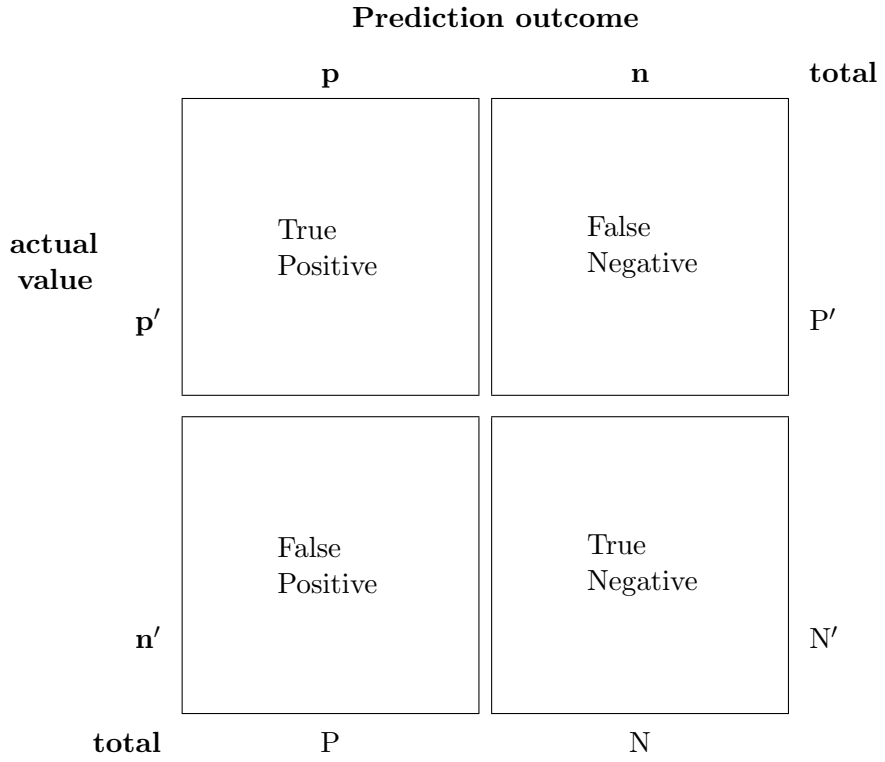

Table 2.1: A confusion matrix as would be expected in a binary classification problem.

is a standard way of measuring the suitability and performance of a model either as a ratio or a percentage. The model error and the true error will not always be the same. If the model error is higher it is reasonable to expect better results if more training is done, this problem is called underfitting or bias. If the model error is lower than would be expected this is called overfitting and the model has been influenced by noise or bad data. It is possible to compare these results when there exists a train and test set. However, during training it is not easy to estimate the true error rate, without revisiting the data collection sites, and expert consultations.

Not all errors are equal; if the cost of a mistake is high or the data is highly imbalanced, it is inappropriate to use the error rate. Using smokers as an example, it would be worse to predict good health to a smoker who actually has cancer versus predicting they have cancer, when they are healthy. A confusion matrix, as seen in table 2.1, tables different types of error. For example, it tables the number of true positives and negatives and the number of false positives and negatives in binary classification.

A confusion matrix can be any n x n matrices, with $(n-1)n$ types of errors. Where an Error E_{ij} will be the ith and jth element of the matrix, except along the diagonal.

Loss Functions

Loss functions measure the classifier's performance; they are very similar but give the model supervisor greater flexibility in controlling the weighting of errors. Cost functions are the most basic loss function, and others will build on it. Cost functions allow the weighting of the different kinds of errors described in a confusion matrix. The cost function,

$$Cost = \sum_{i}^{k} \sum_{j}^{k} E_{ij} C_{ij} \tag{2.8}$$

can then be used to evaluate the model.

The risk function

$$Risk = \sum_{i}^{k} \sum_{j}^{k} E_{ij} R_{ij} \tag{2.9}$$

considers the benefits of correct classification while still penalising the different kinds of misclassifications, thus, it weighs the benefits of correct classifications as well. Beyond this point, the impact of errors may still present some challenges, which is quite common in economic applications. For example, the cost of losing $10,000 will hurt people differently. Thus, some flexibility when it comes to discounting errors and weighting accurate predictions is required. For the sake of medical image classification, correctly identifying Covid-19 patients will be of more value than detecting healthy patients, especially in the elderly or vulnerable.

The utility function

$$Utility = \sum_{i}^{k} \sum_{j}^{k} E_{ij} U(R_{ij}) \tag{2.10}$$

extends on the cost and risk function by allowing the costs or risks to be variable and not constant, the exact methods will depend on the nature of the problem. The fundamental idea is that the same type of error may have different costs of benefits. For example, wrongly classifying a sick person to be healthy may be have a bigger penalty when the person belongs to a disadvantaged group and is less likely to receive medical treatment in the future. The choice of the best utility or loss function is often complex and will depend on the desired application and complexity of the problem (Houlding, Coolen & Bolger 2015).

Chapter 3

Classification Methods

This chapter will cover four traditional classification methods. The first two methods will use properties of normal distributions for classification, these are the *Linear Discriminant Analysis* and the *Quadratic Discriminant Analysis*. Following, this will be the *Logistic Regression* model which assumes a Bernoulli distribution. Finally, the *K-Nearest Neighbour* classifier will follow. In this section the Iris data set will be used to ensure consistency and the comparability of the results.

3.1 Data

The Iris data which has four numeric predictors, $\mathbf{X} \in \mathcal{R}^{nx4}$, and one categorical target variable, with three classes, where Π_i, can be *"Setosa", "Virginica"* and *"Versicolor"* (Fisher 1936). The four predictors are measurements of the height and length of the sepals and petals of the threes species of Iris flowers. In Figure 3.1 the features are grouped in pairs to produce scatterplots on the lower left of the image. On the diagonal we see the kernel density estimate of the class probability distribution. In the upper left the correlations of the features are shown. For class correlations the colored numbers show the within class correlation. On the right, on the figure are boxplots by class showing the distribution of observations for the specified feature. Histograms are also shown on the bottom row of the Figure. In this plot we see the data fits the required assumptions for the discriminant analysis. The estimated probability distribution appear bell-shaped indicating normality. The box plots show similar variances in sepal length and width, but the Setosa species appears to have a smaller variance than the other species. To highlight the differences in our methods we use the petal length and width measures to show the impact of the assumptions on variance.

Exploratory data analysis reveals the species have bell-shaped distributions, which

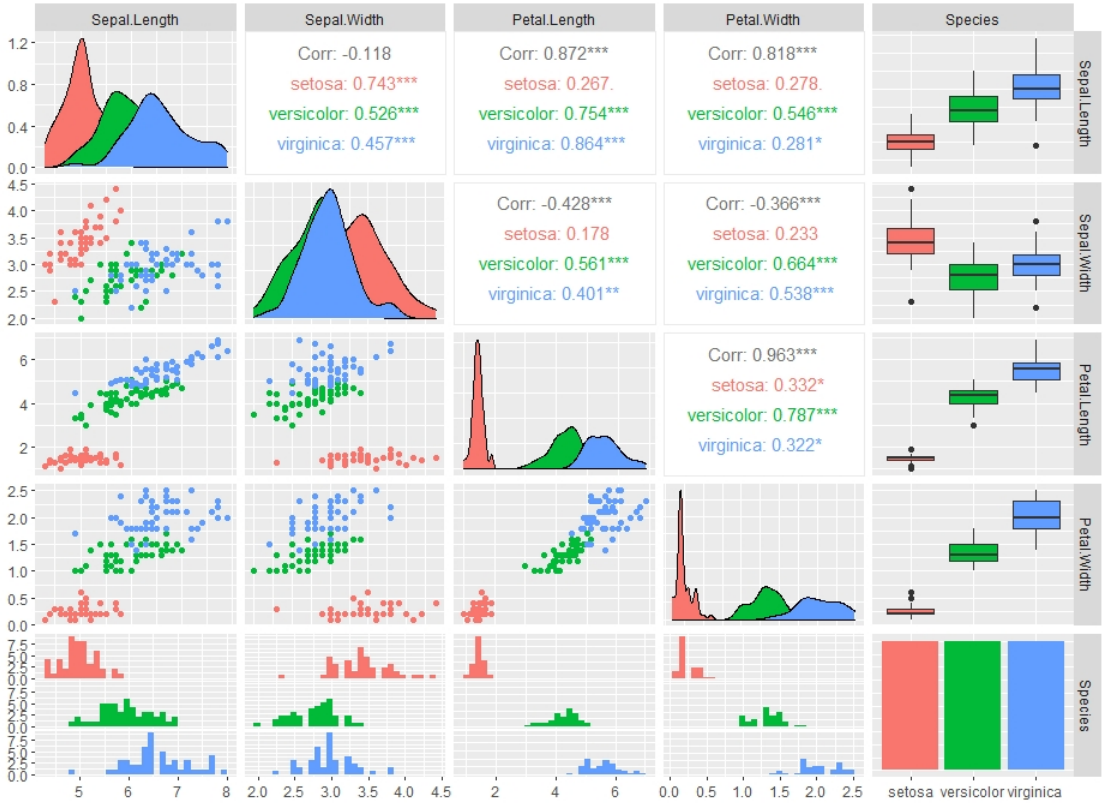

Figure 3.1: Iris Data Exploration- The figure shows the location of observations in the Iris dataset. The colours represent the different possible classes. The Histograms, Boxplots and Kernel-Density plots show the class distributions.

are associated with normal distributions, which make them suitable candidates for the discriminant analyses.

3.2 Linear Discriminant Analysis

The two discriminant analysis methods are similar in that both assume the conditional class distribution are normal (Lachenbruch & Goldstein 1979). However, the linear discriminant analysis is simpler in its assumptions of variance. These two methods are simple methods to learn the workings on supervised learning and classification.

Classification Rule

From the Bayes rule,

$$Prob(\mathbf{x}|\Pi_i) = \frac{f_i(\mathbf{x})Prob(\Pi_i)}{\sum f_j(\mathbf{x})Prob(\Pi_j)}, \tag{3.1}$$

described in Chapter 2, the weighted conditional probability distributions,

$$f_i(\mathbf{x})Prob(\Pi_i) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma^{-1}(\mathbf{x}-\mu_i)\right)}{\sqrt{(2\pi)^n}\|\Sigma\|}\pi_i, \tag{3.2}$$

can be estimated by assuming the normal distribution function given that the conditional class density functions share a common covariance matrix. Where the following parameters are calculated as follows:

1. $\pi_k = N_k/N$, where N_i is the number of class Π_k observations;
2. $\mu_k = \sum_{i\in\Pi_k} \mathbf{x}_i/N_k$;
3. $\Sigma = \sum_k^K \sum_{j\in\Pi_k}(\mathbf{x}_j-\mu_k)(\mathbf{x}_j-\mu_k)^T/(N-K)$.

Given the number of classes K, the total number of observations N in the feature space $\mathbf{X}$, for which $\mathbf{x} \in \mathbf{X}$. Using the Bayesian classification rule described earlier we assign the class Π_i to a new observation $\mathbf{x}$ given that the class Π_i satisfies the equation

$$\begin{aligned} f_i(\mathbf{x})\pi_i &= max_{1\le j\le K} \quad f_j(\mathbf{x})\ Prob(\Pi_j) \\ &= max_{1\le j\le K} \quad \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma^{-1}(\mathbf{x}-\mu_j)\right)}{\sqrt{(2\pi)^n}\|\Sigma\|}\pi_j, \\ &= max_{1\le j\le K} \quad \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma^{-1}(\mathbf{x}-\mu_j)\right)\pi_j. \end{aligned}$$

Decision Boundary

At the decision boundary the odds of selecting one class over another are one, that is the posterior probability distributions are equal, hence given two classes i and j we find

$$\frac{f_i(\mathbf{x})\ Prob(\Pi_i)}{f_j(\mathbf{x})\ Prob(\Pi_j)} = \frac{\pi_i \exp\{-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_x^{-1}(\mathbf{x}-\mu_i)\}}{\pi_j \exp\{-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_x^{-1}(\mathbf{x}-\mu_j)\}} = 1$$

$$\log\left(\frac{f_i(\mathbf{x})\ Prob(\Pi_i)}{f_j(\mathbf{x})\ Prob(\Pi_j)}\right) = \log\left(\frac{\pi_i}{\pi_j}\right) - \frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_x^{-1}(\mathbf{x}-\mu_i) + \frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_x^{-1}(\mathbf{x}-\mu_j)$$

It is now possible to reduce this function into a linear equation

$$
\begin{aligned}
&= \log\left(\frac{\pi_i}{\pi_j}\right) + \frac{1}{2}\left(-\mathbf{x}^T\Sigma^{-1}\mathbf{x} + \mathbf{x}^T\Sigma^{-1}\mu_i + \mu_i^T\Sigma^{-1}\mathbf{x} - \mu_i^T\Sigma^{-1}\mu_i\right) \\
&+ \frac{1}{2}\left(\mathbf{x}^T\Sigma^{-1}\mathbf{x} - \mathbf{x}^T\Sigma^{-1}\mu_j - \mu_j^T\Sigma^{-1}\mathbf{x} + \mu_j^T\Sigma^{-1}\mu_j\right) \\
&= \log\left(\frac{\pi_i}{\pi_j}\right) + (\mu_i - \mu_j)^T\Sigma^{-1}\mathbf{x} - \frac{1}{2}(\mu_i - \mu_j)^T\Sigma^{-1}(\mu_i + \mu_j),
\end{aligned}
$$

where the value of the output will determine which class to assign between Π_i and Π_j. Positive values would mean assigning class Π_i over class Π_j.

$$
Y = \begin{cases} \Pi_i & \text{if} \log(\frac{\pi_i}{\pi_j}) + (\mu_i - \mu_j)^T\Sigma^{-1}\mathbf{x} - \frac{1}{2}(\mu_i - \mu_j)^T\Sigma^{-1}(\mu_i + \mu_j) > 0 \\ \Pi_j & \text{otherwise} \end{cases}
$$

When the equation equals zero this is the decision boundary, the points where the odds of being one class over another are one. Expressing the function in the form $\beta_0 + \beta\mathbf{x}$, where the constant $\beta_0 = -\frac{1}{2}(\mu_i - \mu_j)\Sigma^{-1}(\mu_i + \mu_j) + \log(\frac{\pi_i}{\pi_j})$ and the weight vector $\beta = (\mu_i - \mu_j)\Sigma^{-1}$. Using these insights an observation $\mathbf{x}$ can be classified and linear decision boundaries can be constructed to generalise to future predictions.

Results

Figure 3.2 shows the expected classification regions for a Linear discriminant model trained with the Iris data. The classification regions vary in size, shape and colour. The regions were plotted my computing the Mahalanobis distances for all points to the class means. The colour of each region was assigned by considering which class distribution was closest. The lines are straight as a result of the shared co-variance matrix assumption. The **tidyverse** package with the **R-programming language** were used to prepare and plot the data (Wickham et al. 2019).

.

3.3 Quadratic Discriminant Analysis

From the Figure 3.2 of the classification regions some misclassifications for *'Versicolor'* and *'Virginica'* are noticeable. This is as, a straight line cannot perfectly separate the two regions. The variance of *'Virginica'* is also very large and more observations of *'Virginica'* are misclassified. Thus, revisiting the assumption of a shared covariance matrix is necessary. We now consider the covariance matrix Σ_i for each subset of $\mathbf{X}|\Pi_i$ and attempt to build a classification rule and decision boundary.

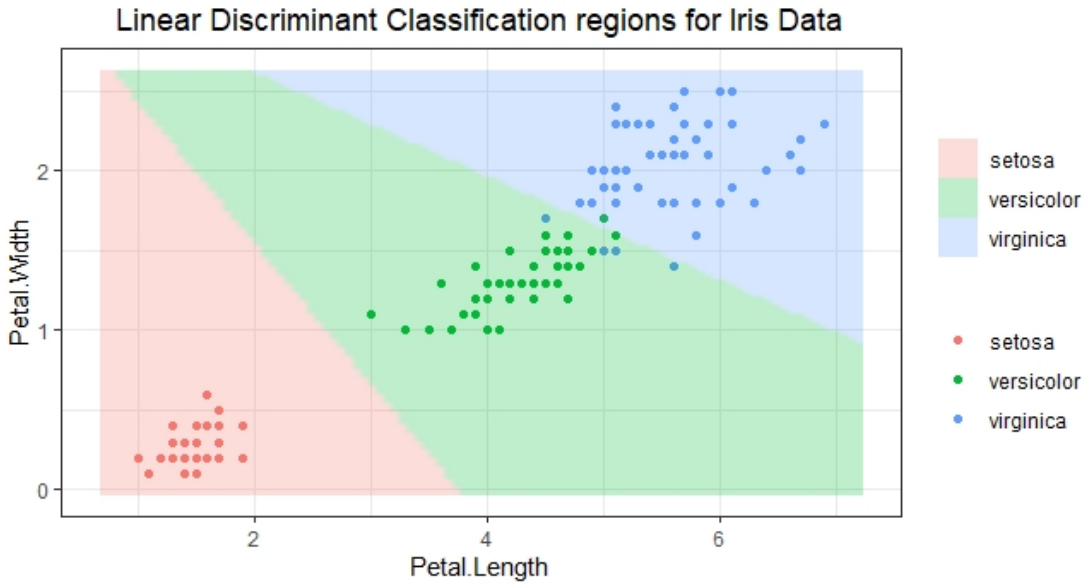

Figure 3.2: Classification regions and decision boundaries from a linear discriminant model trained on the Iris data.

For quadratic discriminant analysis (QDA) we define the classification rule; assign the class Π_i to an observation $\mathbf{x}$ given that it satisfies the condition,

$$f_i(\mathbf{x})\pi_i = \max_{1\leq j\leq K} \quad \pi_j \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_j^{-1}(\mathbf{x}-\mu_j)\right)$$

From this equation we can find the decision boundary between two classes Π_i and Π_j, as the point where they have the same weighted conditional probability density. Thus, the odds are one where we divide

$$\frac{f_i(\mathbf{x})Prob(\Pi_i)}{f_j(\mathbf{x})Prob(\Pi_j)} = \frac{\pi_i \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i)\right)}{\pi_j \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_j^{-1}(\mathbf{x}-\mu_j)\right)} = 1$$

$$\log\left(\frac{f_i(\mathbf{x})Prob(\Pi_i)}{f_j(\mathbf{x})Prob(\Pi_j)}\right) = \log\left(\frac{\pi_i}{\pi_j}\right) - \frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i) + \frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_j^{-1}(\mathbf{x}-\mu_j)$$

From this equation we can extract the log odds on the left-hand side to be used to create the decision boundary, and the right hand side can be simplified further,

$$= \log\left(\frac{\pi_i}{\pi_j}\right) + \frac{1}{2}\left(-\mathbf{x}^T\Sigma_i^{-1}\mathbf{x} + \mathbf{x}^T\Sigma_i^{-1}\mu_i + \mu_i^T\Sigma_i^{-1}\mathbf{x} - \mu_i^T\Sigma_i^{-1}\mu_i\right)$$

$$+ \frac{1}{2}\left(\mathbf{x}^T\Sigma_j^{-1}\mathbf{x} - \mathbf{x}^T\Sigma_j^{-1}\mu_j - \mu_j^T\Sigma_j^{-1}\mathbf{x} + \mu_j^T\Sigma_j^{-1}\mu_j\right)$$

$$= \log\left(\frac{\pi_i}{\pi_j}\right) - \frac{1}{2}\mathbf{x}^T\left(\Sigma_i^{-1} - \Sigma_j^{-1}\right)\mathbf{x} + \left(\mu_i\Sigma_i^{-1} - \mu_j\Sigma_j^{-1}\right)\mathbf{x} - \frac{1}{2}\left(\mu_i\Sigma_i^{-1}\mu_i - \mu_j\Sigma_j^{-1}\mu_j\right)$$

The resulting decision boundary is not linear but quadratic we can then test how well it performs on the Iris data.

Results

Figure 3.3 shows the expected classification regions for a Quadratic discriminant model trained with the Iris data. The classification regions vary in size, shape and colour. The regions were plotted my computing the Mahalanobis distances for all points to the class means. The colour of each region was assigned by considering which class distribution was closest. The blue region covers most of the plot, as a result of a larger co-variance. The Ggplott2 package with the R-programming language were used to make the plot (Wickham 2016).

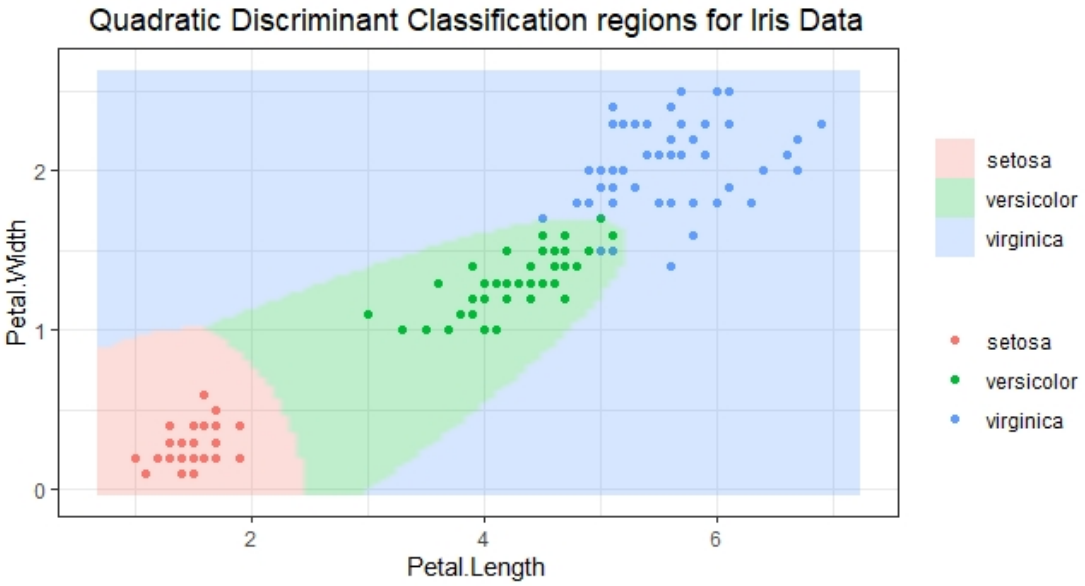

Figure 3.3: Classification regions and decision boundary after using Quadratic discriminant analysis

3.4 Logistic Discriminant Analysis

When presented with a binary classification problem. The conditional probability class distributions, $Prob(\Pi_i|\mathbf{x}) = \pi_i$ of the data must sum to one, $1 = \sum_i^2 Prob(\Pi_i|\mathbf{x})$ for an observation over all the classes. If the odds of observing Π_1 over Π_2 are

$$\text{odds} = \frac{\pi_1}{\pi_2}. \tag{3.3}$$

It follows that the log of the odds,

$$\begin{aligned} \text{log-odds} &= \log\left(\frac{\pi_1}{\pi_2}\right) && (3.4)\\ &= \log\left(\frac{\pi_1}{1-\pi_1}\right) && (3.5)\\ &= log\left(\frac{Prob(\Pi_1|\mathbf{x})}{1-Prob(\Pi_1|\mathbf{x})}\right) && (3.6) \end{aligned}$$

can be modelled as a linear equation. Thus, it can be re-written as

$$\begin{aligned} logit\big(Prob(\Pi_1|\mathbf{x})\big) = log\left(\frac{Prob(\Pi_1|\mathbf{x})}{1-Prob(\Pi_1|\mathbf{x})}\right) &= (\boldsymbol{\beta}_0 \ \boldsymbol{\beta})^T(1 \ \mathbf{x}) && (3.7)\\ &= \boldsymbol{\beta}_0 + \boldsymbol{\beta}^T\mathbf{x} && (3.8) \end{aligned}$$

, which is in the same form of a *logistic regression* model (Agresti 2007). Inversely, the conditional class probability,

$$\begin{aligned} Prob(\Pi_1|\mathbf{x}) &= \frac{e^{\boldsymbol{\beta}_0+\boldsymbol{\beta}^T\mathbf{x}}}{1+e^{\boldsymbol{\beta}_0+\boldsymbol{\beta}^T\mathbf{x}}} && (3.9)\\ &= \frac{1}{1+e^{-\boldsymbol{\beta}_0-\boldsymbol{\beta}^T\mathbf{x}}} && (3.10) \end{aligned}$$

can be represented as a function of $(\boldsymbol{\beta}_0 \ \boldsymbol{\beta})^T$ and $(1 \ \mathbf{x})$. Which is in the form of a sigmoid function, an S-shaped curve, as seen in Figure 3.4 (Leibovich-Raveh et al. 2018). From the Figure, it is obvious that the parameters will change the shape and location of the curve. The range on Y is between 0 and 1 and the curve resembles a step function. These are ideal properties of logistic regression, as it will allow more realistic estimates than would be observed under linear regression (G. James & Tibshirani 2013).

Maximum Likelihood estimation

Selecting the best possible vector of parameters $\boldsymbol{\beta}$ and the intercept $\boldsymbol{\beta}_0$, is possible through maximum likelihood estimation, unlike least-squares estimation which is used normal-linear regression. Generally, the maximum likelihood finds the best possible

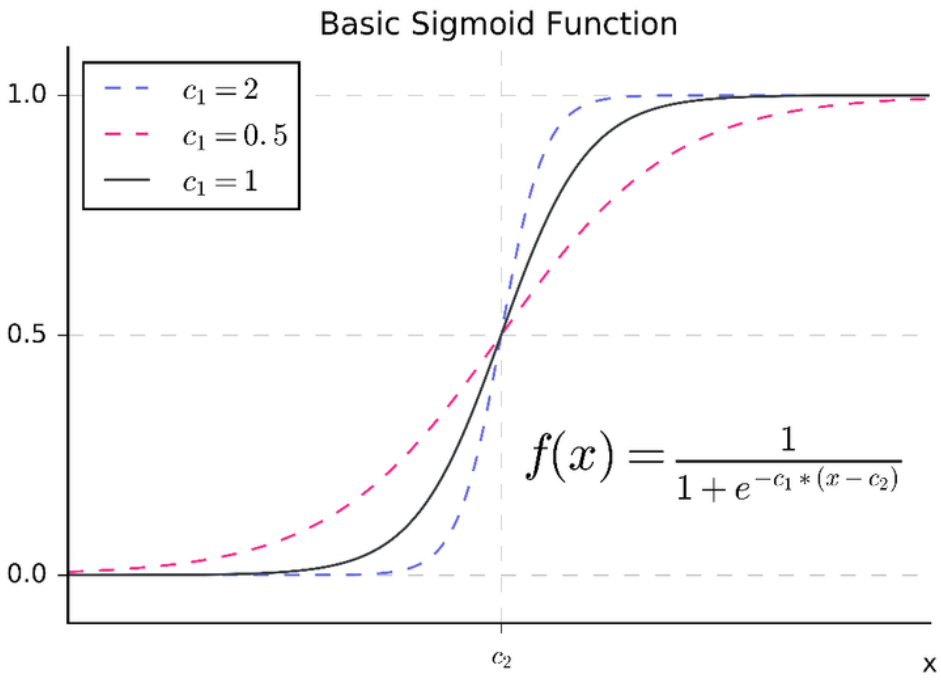

Figure 3.4: A sigmoid function, showing the characteristic S-shape. The parameters c_1 and c_2 determine the steepness of the curve and the location of the centre of the curve (Leibovich-Raveh et al. 2018).

values for the unknowns $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_0$. To begin the likelihood function $\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\beta}_0)$, which represents the probability that the observed odds was generated by the Bernoulli distribution with the unknown values $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_0$. The logistic regression can only be applied to a binary target variable; thus, it is assumed that Y follows a Bernoulli distribution. Thus, by selecting the values that maximise this probability, the logistic regression model can use the best parameters for classification (Albert & Anderson 1984). In this case,

$$\mathcal{L}(\boldsymbol{\beta}) = \prod_{i=1}^{n} Prob(\Pi_1|\mathbf{x}_i)^{y_i} Prob(\Pi_2|\mathbf{x}_i)^{1-y_i} \tag{3.11}$$

$$= \prod_{i=1}^{n} Prob(\Pi_1|\mathbf{x}_i)^{y_i} \left(1 - Prob(\Pi_1|\mathbf{x}_i)^{1-y_i}\right) \tag{3.12}$$

where y_i is in $\mathrm{Y} = y_1, y_2, y_3, ...y_n$, given

$$Y = \begin{cases} 1 & if X \in \Pi_1 \\ 0 & otherwise \end{cases}$$

In practice, it is easier to solve the logarithm of this function while still obtaining the same results (Kleinbaum & Klein 2010).

$$\begin{aligned}
\boldsymbol{l}(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \left(y_i \log\left(Prob(\Pi_1|\mathbf{x}_i)\right) + (1 - y_i) \log\left(1 - Prob(\Pi_1|\mathbf{x}_i)\right) \right) \\
&= \sum_{i=1}^{n} \left(y_i \log\left(\frac{Prob(\Pi_1|\mathbf{x}_i)}{1 - Prob(\Pi_1|\mathbf{x}_i)}\right) + \log\left(1 - Prob(\Pi_1|\mathbf{x}_i)\right) \right) \\
&= \sum_{i=1}^{n} \left(y_i(\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i) + \log\left(1 - \frac{e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}\right) \right) \\
&= \sum_{i=1}^{n} \left(y_i(\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i) + \log\left(\frac{1}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}\right) \right) \\
&= \sum_{i=1}^{n} \left(y_i(\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i) - \log\left(1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}\right) \right)
\end{aligned}$$

The log-likelihood function, $\boldsymbol{l}(\boldsymbol{\beta})$ will be at a maximum when its first partial derivative,

$$\frac{\delta \boldsymbol{l}(\boldsymbol{\beta})}{\delta \boldsymbol{\beta}} = S(\boldsymbol{\beta}) = \sum_{i-1}^{n} \left(y_i \mathbf{x}_i - \frac{\mathbf{x}_i e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}} \right) = 0 \tag{3.13}$$

with respect to $\boldsymbol{\beta}$ is equal to zero. To ensure the solution is a global maximum the

second order differential,

$$\frac{\delta S(\boldsymbol{\beta})}{\delta \boldsymbol{\beta}} = \left(\sum_{i-1}^{n} \left(y_i \mathbf{x}_i - \frac{\mathbf{x}_i e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}} \right) \right)' \tag{3.14}$$

$$= -\sum_{i=1}^{n} \frac{\mathbf{x}_i \mathbf{x}_i^T e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i} \left(1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}\right) - \mathbf{x}_i e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i} \left(\mathbf{x}_i e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}\right)}{\left(1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}\right)^2} \tag{3.15}$$

$$= -\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T \left(\frac{e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}} \right) \left(\frac{1}{1 + e^{\boldsymbol{\beta}_0 + \boldsymbol{\beta}^T \mathbf{x}_i}} \right) = F(\boldsymbol{\beta}) \leq 0 \tag{3.16}$$

must be negative. Using these constraints, numerical methods can be devised to find a solution.

Iteratively Re-weighted Least Squares

To find the $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_0$ which satisfy the equation, the *Newton-Raphson* algorithm is a means of improving estimates. It begins with an initial guess for the parameter values that would maximise the log-likelihood function. Placing these values into $S(\boldsymbol{\beta})$ and $F(\boldsymbol{\beta})$, it is possible to find better values that would maximise likelihood equation, as the resulting function has the shape of a concave parabola. The new values are used as a second guess for the parameter values. Repeating this process, until there is no further change in practical terms will yield the best result.

Results

Figure 3.5 shows the expected classification regions for a logistic regression model trained with the Iris data. The red coloured points, referring to the 'Setosa' species were taken as a reference point for which the odds of observing the other two classes were calculated. This was done using the vector generalised additive model in the R programming language (Yee & Wild 1996). The plot produced straight lines similar to those observed in LDA.

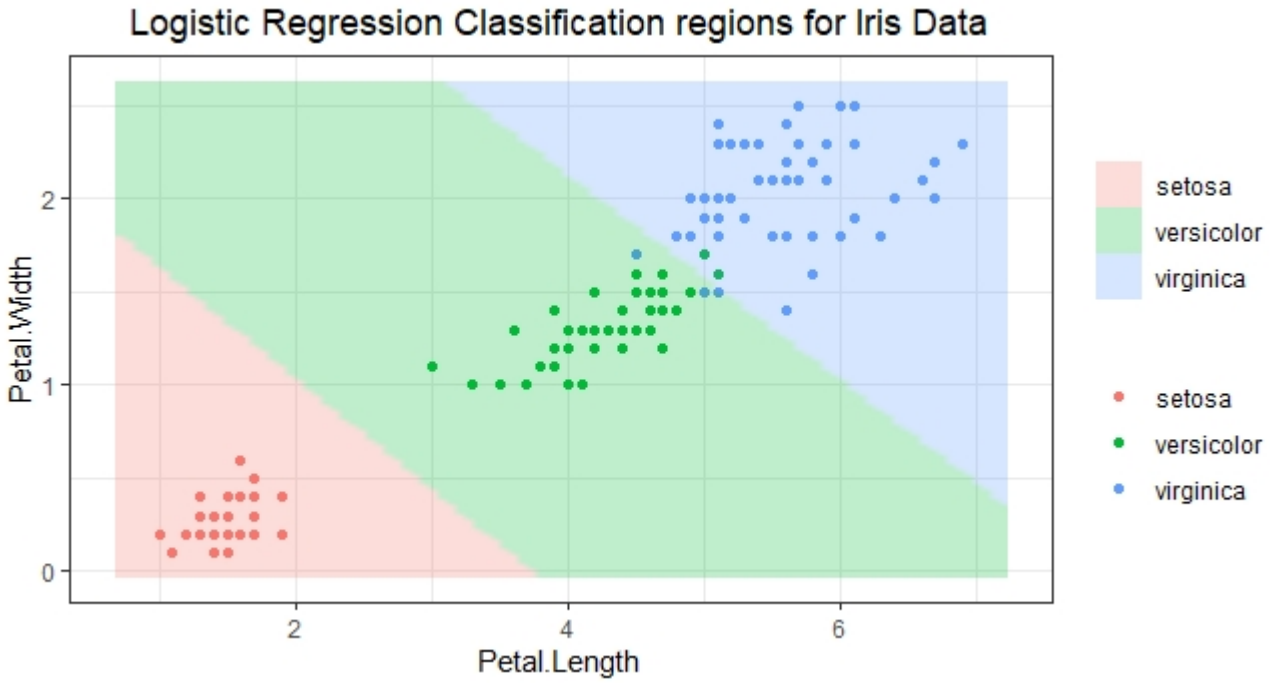

Figure 3.5: Classification regions for logistic regression model trained on Iris data.

3.5 K-Nearest Neighbours

The nearest neighbour is one of the most influential classifiers out there given it overcomes the disadvantages of the previously discussed classifiers. Firstly, traditional classifiers, like the discriminant analyses are limited by the assumptions they make, which do not always hold. Secondly, classifiers like logistic regression are simple but become complex when dealing with non-binary data or non-linearly separable data. The **Nearest neighbour** classifier makes no distributional assumptions and they are relatively easy to understand and use (Dudani 1976).

3.5.1 Introduction and Method

The nearest neighbour classifier works by comparing an unseen observation, to those observations from the training set (Cover & Hart 1967). Here is the algorithmic process;

1. A training set is learned by essentially recording it or committing it to memory.
2. When given a new observation, it uses a measure of similarity to rank the observations in the training set.

3. A set of size k, an odd number, with the k closest observations is then selected.

4. The modal class Π_i in this set is then assigned to the new observation.

The only parameters of control is the measure of similarity and the size of k. The size of k is often selected through a process called cross-validation (Browne 2000). This entails repeating the training on subsamples over different values of k. The performance of the trained models are then compared. The parameter's for the model achieving the greatest accuracy will be selected for the final model.

3.5.2 Example

In Figure 3.6, we have a new observation represented by the red dot and we wish to classify it between class **A** or class **B**, represented by the green triangles and the blue squares respectively. From the diagram we see circles representing the distances from our new observations, we can see that by considering the three nearest neighbours we have two instances of class **B** and only one instance of class **A**. Following the classification rule described earlier we can classify the new point as with the most frequent label in the neighbourhood $\mathcal{N}_0$ of size three which is class **B**.

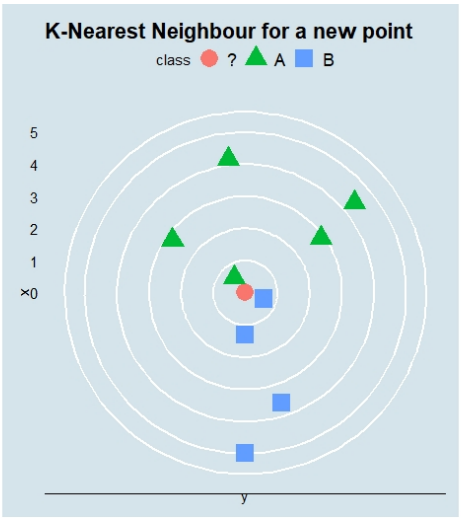

Figure 3.6: Neighbourhoods for K-nearest neighbour classifiers.

Distance Measures

Distance is used to rank the observations, so as to allow the most similar observations to be considered first. To compare the similarity of the new observation to those in the

learning set we need to define the distance measure. The Euclidean distance is the most convenient measure it takes the length of the linear segment between two points. We then consider the distance between the new observation and an observation $\mathbf{x}_i$ in the learning set.

$$D_{ij} = \sqrt{\sum (\mathbf{x}_i - \mathbf{x}_j)^2} \tag{3.17}$$

Other measures like the *Manhattan* distance are essential in areas like text analysis where the Euclidean distance would be irrational. The Manhattan distances is the distance between two points measured at right angles. The measure of distance used will depend on the nature of the data used. The Euclidean distance will work well for medical images analysis.

Normalisation

One issue does arise with using the nearest neighbour classifier this way; if the variance of one of the features is extremely large in comparison to the other feature parameters, the distances calculated will always be large and thus, the distance measure may be dominated by that feature. As result we standardise the distances by dividing by their variances. We take each feature $\mathbf{X}_i \in \mathbf{X}$, and find the variance σ_i and the mean $\bar{x}_i$ and the use the Z-score scaling method by

$$\mathbf{X}'_i = \frac{\mathbf{X}_i - \mu_i}{\sigma_i} \tag{3.18}$$

The new normalised feature space $\mathbf{X}'$ is then stored for comparing with new observation which are also scaled by the stored means and standard deviations.

3.5.3 Curse of Dimensionality

The major challenge nearest neighbour classifiers face is when dealing with highly dimensional data. As the number of dimensions increase the behavior of the function becomes unpredictable, as the distance to the nearest point will approach the distance to the furthest point (Beyer K. 1999). A thought experiment can be used to demonstrate this. Consider a one-dimensional space with 10 points, 10% of this space can be analysed with one point. Now consider a two dimensional space with 10 points on each axis and 100 combinations. If again 10% of the data needs to be analysed, 10 points in total will be required. However, $10\sqrt{0.1} \approx 3$ points in each axis will now be required. Thus, as the dimensions increase the number of points in each axis will approach the total number of points. Thus, local behaviour is difficult to isolate in higher dimensions (Rust 1997).

Iris Data Example

The K-nearest neighbour is trained on the Iris data. Using the model, the decision boundaries and the classification regions are plotted in Figure 3.7. The caret package with the R programming language are used to train this model and plot the figures (Kuhn 2008). The boundaries are flexible and capture all points, however, cross-validation revealed over-fitting with the 1-Nearest Neighbour.

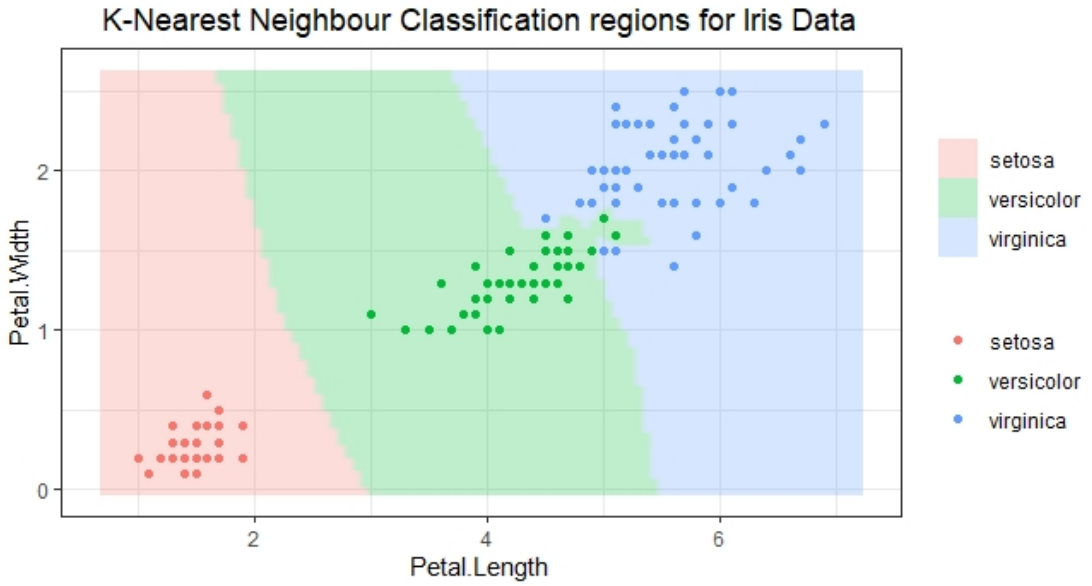

Figure 3.7: The Classification boundaries and regions for the Iris data after analysis with the 1-nearest neighbours classifier.

3.6 Summary and Conclusion

Table 3.1 shows the train and test accuracies of the various models. A distinction is made between testing on small and large portions of data. The Iris data was split in the first instance into 1:4 split and in the second it was split 1:1. The first reason for this was to validate models actually learned in the training phase. The second reason was to observe the model performance with different levels of training information.

Generally, all models performed well and there were no significant changes between training and testing. Overall, more training data improved the test results. Quadratic Discriminant Analysis performed the best during test in all rounds and Logistic regression may have performed the worst in this case, however,the differences are small.

Model performance on Iris data				
Model	Train(50% Split) Accuracy	Test(50% Split) Accuracy	Train(80% Split) Accuracy	Test(20% Split) Accuracy
Linear Discriminant Analysis	96.00%	97.33%	95.00%	100.00%
Quadratic Discriminant Analysis	96.00%	98.66%	97.50%	100.00%
Logistic Regression	97.33%	94.67%	95.00%	100.00%
1-Nearest Neighbour	98.67%	96.00%	99.17%	100.00%
3-Nearest Neighbour	97.33%	96.00%	95.83%	100.00%
5-Nearest Neighbour	96.00%	97.33%	95.00%	100.00%
7-Nearest Neighbour	96.00%	96.00%	95.83%	100.00%

Table 3.1: Table of testing and training performance for models trained on a portion of the Iris data. A distinction is made between the models trained with 80% of the data and those trained with 50% of the data.

In summary discriminant analysis is easy to perform, but the assumptions of normality are a limitation on the data which can be classified by this method. Another limitation to these methods comes when dealing with non-linearly separable data the solution to this maybe in choosing better predictors, but in general other classification techniques will show robust ways of dealing with this.

The biggest strength of logistic regression is that it is easy to perform and interpret. The binary nature is a serious limitation, even though solutions exist. Nonetheless, it is applicable to a wide range of data and has uses from weather forecasting to medical studies. This is mainly because of the probabilistic nature of the results and the odds it is capable of modelling.

The nearest neighbours classifier is called a lazy learner as it does not create any models of the data, instead it keeps a copy of the data for comparisons. As a result, this method can be quite accurate, but this can be resource intensive. It is does not require much supervision and is often used in online learning scenarios, where new models are needed frequently. It has been used extensively in loan applications as it can adapt quickly to changes in conditional class distributions this is achieved by discarding or re-weighting older observations and considering newer observations.

All together these classifiers are relatively simple to perform, and they are accurate as validated by test accuracies. The iris data did have some overlap as a subset of the data was used. However, with only two features out of four the results were impressive, especially when the models were presented with more data. Using the insights of feature extraction and model evaluation covered in this section and Chapter 2, the next section can begin to work with images as well as more complex models.

Chapter 4

Tree-based Classifiers

In the Chapter 3, we saw the decision boundaries produced by the discriminant analyses, logistic regression and the nearest neighbour methods. It was noted that in some areas there was overlap in the data and linear decision boundary was not ideal. The nearest neighbour method was able to work around this. But as given the curse of dimensionality the nearest neighbour may not be the ideal candidate for image classification. Thus, the tree-based classifiers provide a flexible, but yet robust methods for dealing with complex data. Tree-based classifiers have existed since the early 1960's, but it was not until 1984 that binary trees that used conditional class probabilities and recursive partitioning were invented (Breiman et al. 1984).

4.1 Overview

Classification trees are a system of conditions that split data depending on the characteristics. From Figure 4.1, it can be seen that they contain nodes. Nodes are the points at which classification happens. Beginning at the root node, which consists of all the data, a split is made into daughter nodes of which there are two types, *decision* and *terminal nodes*. On decision nodes, the data is further split into two daughter nodes; the splitting may carry on repeatedly until sufficient classification is achieved. For terminal nodes, no further splits occur, and the data is given a class Π_i.

4.2 Tree building Methods

For a tree to be informative and useful, the branching needs to be optimised and logical. Hence different splitting strategies are used to determine where and how to split a tree

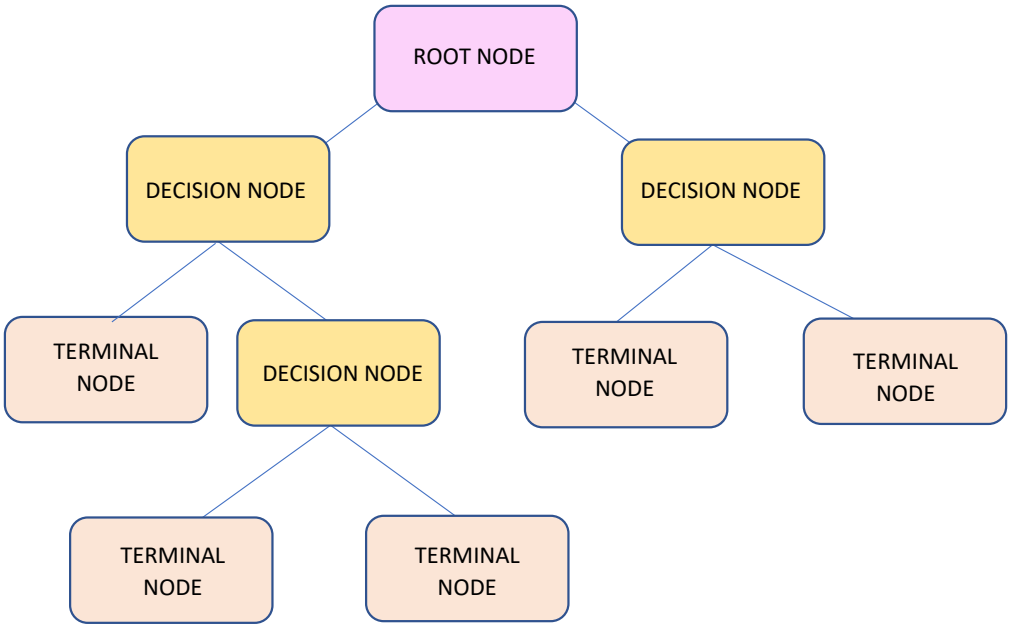

Figure 4.1: Classification Tree structure, showing root, decision and terminal nodes.

(Breiman 1996). The choice of a split at the root node is determined using a goodness of split measure.

4.2.1 Node Impurity Functions

Node impurity function can act as a goodness of split measure and they can be defined as follows. Let $\Pi_1, ..., \Pi_k$ be the k classes. At a node τ, the node impurity function will be as follows;

$$i(\tau) = \phi(Prob(1|\tau), ..., Prob(k|\tau))$$

where Prob$(k|\tau)$ is an estimate of $Prob(\mathbf{x} \in \Pi_k|\tau)$, the conditional probability that an observation $\mathbf{x}$ is in in Π_k given that it falls into node τ. ϕ is a symmetric function, defined on the set of all class probabilities which sum to one. As would be expected, ϕ is minimized when a node is pure. An example of ϕ is the *entropy function*,

$$i(\tau) = -\sum_{j=1}^{k} Prob(j|\tau) \ \log\big(Prob(j|\phi)\big), \tag{4.1}$$

and another example is the *Gini diversity function*,

$$i(\tau) = \sum_{j \neq j'} Prob(j|\tau)p(j'|\phi) = 1 - \sum_{j} Prob(j|\tau)^2, \tag{4.2}$$

where Prob$(j|\tau)$ is an estimate of $Prob(\mathbf{x} \in \Pi_j|\tau)$. The functions both produce convex

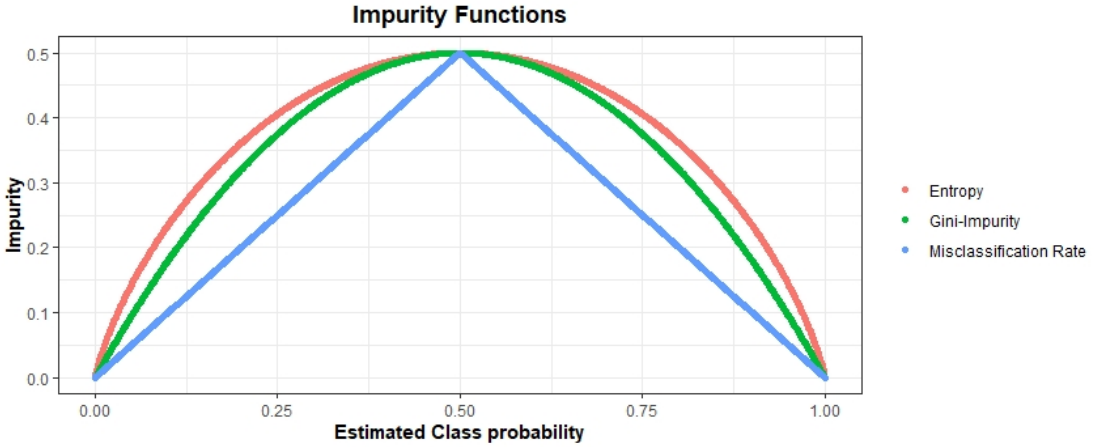

Figure 4.2: This figure shows the impurity functions and the misclassification rate. The entropy functions shown in red is convex and rounder than the Gini-impurity function is shown in green.

curves and penalise splits which are random. This can be seen in Figure 4.2, as random splits in the binary case will be centred around 0.5. Breiman found that the *entropy function* preferred splits that balanced the sample sizes of child nodes, whereas the *Gini criterion* preferred splits which put one pure class into one pure node (Breiman 1996). The different slopes of the curves in Figure 4.2 shows how Gini coefficient slopes faster than entropy function.

The **mis-classification rate**, $r(\tau) = 1 - \max(Prob(k|r))$, in Figure 4.2 is calculated by considering the chance of being wrong. In the two class case this is a linear function represented by the blue line in Figure 4.2 (Breiman et al. 1984).

An illustration of the node-impurity functions and the misclassification rate are shown in Figure 4.2. The Figure shows a scaled Gini-impurity function, this is scaled to align its peak with the entropy function and the misclassification rate. As can be seen in the Figure there is a heap in the centre, where the estimated class probability is closer to 50% or $\frac{1}{k}$ in the multi-class case. The convergence of these functions at this point shows how trees cannot discriminate from truly random data. however,the convergence of the functions at the edges shows how complete knowledge of the distribution reduces the misclassification rate to zero. More importantly the impurity functions also converge to zero. Thus, a system that reduces the misclassification rate and the node impurity rate will lead to more accurate generalisations.

4.2.2 Information Gain and the Goodness of Split

Given data of categorical, numerical or a mixed nature, it is possible to select subsets from these based on the qualities of individual features. However, the splitting mechanisms are different. Categorical variables are the easiest to split, the number of possible subset combinations are all possible, under binary splitting. Such that the number of splits, $2^{k-1}-1$, depend only on the number of classes k. For example, if given data of the letters A,B,C; three splits are possible $A\|B,C$; $B\|A,C$; and $C\|A,B$. For continuous numeric variables the splits can be any real value, but in the form of an inequality, i.e., $x \geq 5$.

With these formulas the goodness of a split measure is possible by calculating the difference in node impurities before and after the split. To do this we need to estimate the probability $Prob(j|\tau)$ at the node . This can be estimated by

$$Prob(j|\tau) = \frac{n_{+j}}{n_{++}} \quad \forall j = 1, ..., k,$$

where n_{++} represents the number of observations in the parent node and n_{+j} represents the number of observations belonging to class Π_j in the parent node. n_{Li} and n_{L+} will represent the number of observations belonging to class Π_i in the left daughter node and the total number of observations in the left daughter node, respectively. An illustration will be shown section 5.2.4. Thus, the estimated impurity function for entropy before the split is,

$$i(\tau) = -\sum_{i=1}^{K} \frac{n_{+i}}{n_{++}} \log \frac{n_{+i}}{n_{++}}, \tag{4.3}$$

$$i(\tau_L) = -\sum_{i=1}^{K} \frac{n_{Li}}{n_{L+}} \log \frac{n_{Li}}{n_{L+}}, \tag{4.4}$$

$$i(\tau_R) = -\sum_{i=1}^{K} \frac{n_{Ri}}{n_{R+}} \log \frac{n_{Ri}}{n_{R+}}. \tag{4.5}$$

The weighted difference,

$$\Delta i(\mathbf{s}, \tau) = i(\tau) - p_L i(\tau_L) - p_R i(\tau_R) \tag{4.6}$$

where $\mathbf{s}$ is the chosen split and τ is the node. p_L and p_R are the proportions of data in the left and right nodes respectively. The best split will provide the most information gain Δ.

4.2.3 Recursive Partitioning

Selecting the best split using the information gain is good, without repeating the process on daughter nodes the method would be unlikely to be informative. Recursive partitioning is the repetition of this procedure on all further nodes generated after further splitting. This method is completed layer by layer hence it is called *recursive partitioning*

1. On the learning set we use the goodness-of split criterion to split the root node. To do this we calculate the information gains for all splits on all variables.
2. We take the max information gain as the first split.
3. We then continue this process on all daughter nodes.
4. We stop either when the model is saturated or when a predefined limit is reached such as the number of splits, branches or level of information gain.

For the model to be saturated there will information gain will be equal to zero for all possible splits. Model saturation is ideal for small uncomplicated data sets. However, limits may need to be placed to prevent extremely large, complicated trees. There exist two main methods for limiting tree growth. The first is to impose a minimum information gain for a split to be made. The second is to add a minimum number of observations for daughter nodes when a split is made. It may be more practical, however, to split a tree to saturation, and then to remove unnecessary splits.

4.3 Titanic Data Example

The Titanic was a ship that crashed into an iceberg in 1912, there were a limited number of seats available on the lifeboats, thus, many people on board died. A summary of the passenger details are available in the R programming language (R Core Team 2017). Table 4.1 shows a sample of this summary of this. It shows the frequencies of passengers who survived or died controlling for their sex, age and passenger class. The data available in the full table will be used to illustrate the workings of classification trees. The aim is to predict a passenger's fate based on their sex, age and passenger class. To begin an analysis of the features and possible splits using the methods described in section 5.2.

In the feature space all the variables are categorical and ordinal. The sex and age variables are binary hence only one split is possible. For the Class variables there exist four possible outcomes, hence the number of unique splits is $2^{m-1} - 1$, where m is the number of unique cases. Hence there are nine possible splits at the root node. Following the steps above we calculate the information gain for all possible splits on all variables. The split with the highest rank or information gain is the optimal split at the root node.

Titanic data				
Passenger Class	Sex	Age	Frequency	Survived
First Class	Male	Child	5	Yes
Second Class	Female	Child	13	Yes
Third Class	Male	Adult	387	No
Crew	Female	Adult	20	Yes
First Class	Female	Adult	4	No

Table 4.1: A sample of rows in the Titanic Data

Given that there were 1731 males of which 367 survived, and there were 470 females of which 344 survived; and of the total 2201 passengers 711 survived. With this information the entropy before and after the split can be calculated. Using Equation 4.3, the entropy at the beginning,

$$= -\left(\frac{711}{2201}\log\left(\frac{711}{2201}\right) + \frac{1490}{2201}\log\left(\frac{1490}{2201}\right)\right) \tag{4.7}$$

$$= 0.2732 \tag{4.8}$$

If we use Equations 4.3 and 4.6 to calculate the information gain for the tops splits as shown in Table 4.2. From Table 4.2 the top 9 splits possible at the root node are shown. Repeating this process on the daughter nodes allows the creation of the tree shown Figure 4.3. Using the classification rule of assigning the majority class to a node the root node is blue indicating the majority of passengers died. At each branch a condition is met, using insights from Table 4.2, the first split is at sex. Each node is shown by a box and each condition for splitting the node is below it. If the condition is met the observations are partitioned to the left, if not they are partitioned to the right. In each node the colour intensity matches the purity of the node, with dark green indicating a large proportion of survivors and dark blue indicating a large portion of fatalities. Towards the bottom of the tree the colour intensities are darker indicating a good classification. In the nodes the text indicates the majority class, followed by the proportion of survivors in the node and finally the portion of the original data contained in the node. In the Figure 4.3, the groups which were predicted to survived were first class females, second class females, female crew, first class children and second class children. More splits were possible, however, five nodes were considered sufficient for illustrative purposes.

Information Gain by Split at the Root Node			
Variable	Split	Rank	Information Gain Δ
Sex	Male	1	0.0987
Class	First & Second Class	2	0.0348
Class	First	3	0.0336
Class	Crew	4	0.0110
Class	Third	5	0.0056
Age	Child	6	0.0044
Class	First & Third Class	7	0.0044
Class	Second	8	0.0027
Class	First Class & Crew	9	0.0011

Table 4.2: Information gain for nine different possible splits of the Titanic data.

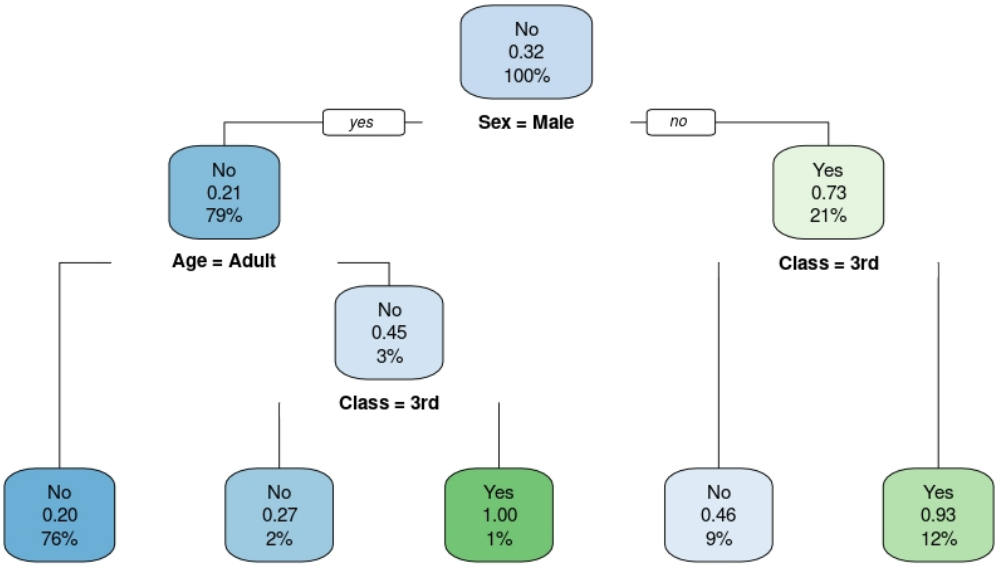

Figure 4.3: Classification Tree for Titanic Data. This tree shows how passengers are classified in the model.

4.4 Pruning and Cross-Validation

4.4.1 Misclassification Rate

To calculate the tree error rate, the misclassification rate R(τ) for an observation in a node is first defined.

$$r(\tau) = 1 - max_k\, p(k|\tau). \tag{4.9}$$

Thus, let T be the tree classifier and $\widetilde{T} = (\tau_1, \tau_2, ..., \tau_n)$ be the set of all terminal nodes of T, then the error rate eqn.4.10 of the tree can be estimated by summing the error for each observation at each node by the probability of that observation falling in that node $P(\tau_)$. In practice it is easier to estimate $P(\tau_l)$, by considering the proportion $p(\tau_l)$ of all observations that fall into a node

$$R(T) = \sum_{\tau \in \widetilde{T}} R(\tau)P(\tau) = \sum_{l=1}^{L} R(\tau_l)P(\tau_l) \tag{4.10}$$

Thus, the resubstitution estimate of R(T) is

$$R^{re}(T) = \sum_{l=1}^{L} r(\tau_l)p(\tau_l) = \sum_{l=1}^{L} R^{re}(\tau_l) \tag{4.11}$$

where $R^{re}(\tau_l) = r(\tau_l)p(\tau_l)$. It can be observed from this estimate that overfitting is highly possible for the following reasons;

1. Bigger trees will have smaller estimates of $R^{re}(T)$, as when trees get larger terminal nodes will tend to have fewer observations, and observations more paths to the desired class, thus, reducing the classification error to zero. $R^{re}(T^*) \leq R^{re}(T)$, where T^* is a larger tree and hence there are more points to distinguish classes.
2. The trees may grow to be large and complex and may even be bigger than the data as this estimate is sensitive to noise.

4.4.2 Pruning

A tree optimised on a given subset will be one that yielded the best accuracy. As has been shown, trees chosen this way fail to perform as expected. Thus, another means needs to be used to select the best tree. To solve this problem a penalty needs to be placed on the accuracy for larger trees. This means a tree with fewer branches is selected, but this is not at all obvious at the beginning. Thus, pruning is a means of selecting a sub-tree of the largest tree so as to maximise the average accuracy on different subsets of the data (Breiman et al. 1984).

To achieve this the resubstition estimate $R^{re}(T)$ is redefined, to accommodate a regularisation parameter, c $\geq$ 0, called the complexity parameter. Thus, for any node $\tau \in T$,

$$R_c(T) = R^{re}(\tau) + c \tag{4.12}$$

it follows that,

$$R_c(T) = \sum_{l=1}^{L} R_c(\tau_l) = R^{re}(T) + c|\widetilde{T}|, \tag{4.13}$$

where $|\widetilde{T}|$=L represents the subnodes of T_{max} which will be the terminal nodes of a smaller tree T. The magnitude of $c|\widetilde{T}|$ determines how much larger trees are penalised as it depends on the number of terminal nodes, as a result the tree that minimises this error is chosen as the best model.

$$R_c(T(c)) = min_T \, R_c(T) \tag{4.14}$$

As would be expected, if the value of c is small the tree will tend to over fit as observed in naive decision trees, increasing the value of c will penalise larger trees, but if this value is too large it may lead to underfitting where the tree is too small to provide any useful predictions. Thus, the value of c is chosen through cross-validation.

4.5 Random Forests

Overfitting in trees is quite common and pruning or bagging may help reduce this, however, **Random Forests** present as a highly accurate and informative alternative (Breiman 2001). Through the strong law of large numbers, they have been shown to be immune to overfitting. Secondly, they are able to produce a probabilistic output alongside information about the importance of features. In addition to this Random Forest work easily with highly dimensional data. They can work with images and text inputs and have no problems with multinomial problems like SVMs and logistic regression. Thus, they overcome the major disadvantages of the classifiers discussed so far. However, they have one disadvantage, they are "black box" models, this means they cannot provide an easily understandable model for researchers or operators to exploit as is seen with decision trees (Utkin, Kovalev & Coolen 2020).

4.5.1 Method

Random forests are an ensemble method, they combine many weak classifiers to form one strong classifier. The reason they perform better is as a result of the Strong Law of Large Numbers (Breiman 2001). The influence of any outliers or strong predictors is reduced through random sampling and averaging. Figure 4.4 shows N subsamples taken without replacement from the training set a tree is then trained on each of these samples (Machado, Recamonde-Mendoza & Corbellini 2015). In Figure 4.5 the unobserved value X is classified with all trees trained previously, the majority vote is the output of the random forest classifier (Machado, Recamonde-Mendoza & Corbellini 2015).

1. N subsamples of the data are taken as shown in Figure 4.4. These samples are randomly drawn without replacement from the features and observations.
2. A tree is then trained with each subsample to its fullest size.

3. The collection of trees are stored as a random forest.

4. New observation are classified by taking the modal vote from the trained trees as shown in Figure 4.5.

4.5.2 Handwritten digit data

In our goal of classifying images, we use images of handwritten digits for illustrative purposes. random forests and classification trees will be trained on the data. The handwritten digit data comes from the modified national institute of science and technology database (MNIST) (LeCun, Cortes & Burges 2010)(Chollet et al. 2015). For simplicity we consider vectors summarised from the images of the numbers 2 and 7. The simplification is done by taking the average pixel density in four quadrants of the image. For visualisation purposes only the top left and bottom right quadrants are considered.

Figure 4.6 shows a plot of the images seven and two. The image is split into four quadrants and will be summarised into four variables. The variables will represent the proportion of dark pixels in the respective quadrant. For visualisation purposes we consider only two variables x_1 and x_2. The variables x_1 and x_2 represent the top-left and bottom right corners, respectively. From the plots it is obvious that a large number of pixels are concentrated in x_1, which is the image of seven. Likewise, the image on the right has the smallest proportion of points in the top-left quadrant x_1.

For evaluation and computational reasons only a 1000 of the images were considered. Given the database has 70,000 images it is possible to estimate the true conditional distribution (Irizarry 2021). We use the true conditional probability to visually inspect model performance. 800 images are used for training and 200 are used for testing. An ensemble of classification trees will be compared against the random forest algorithm.

In Figure 4.7 the training distribution is shown on the left and the estimated true class distribution is shown on the right. The goal will be to visually inspect the effectiveness on random forests versus bagged trees. Bagged trees will involve training classification trees on subsamples of the data, similarly to the process shown in Figure 4.4. However, the difference between bagged trees and random forests will be on the use of features. Random forests will use a subset of the features, while bagged trees will use all features. From Figure 4.7 the training sample appears to be biased, as it has fewer observation in the top-left.

Figure 4.8 shows the decision boundaries for the bast bagged tree classifier and the best random forest classifier. This was achieved using methods from the caret package (Kuhn 2008) and data from the Keras and dslabs packages (Chollet et al. 2015) (Irizarry 2021). Resampling and cross-validation were the main methods used to determine the best model parameters. Cross-validation is a method of selecting the best parameters

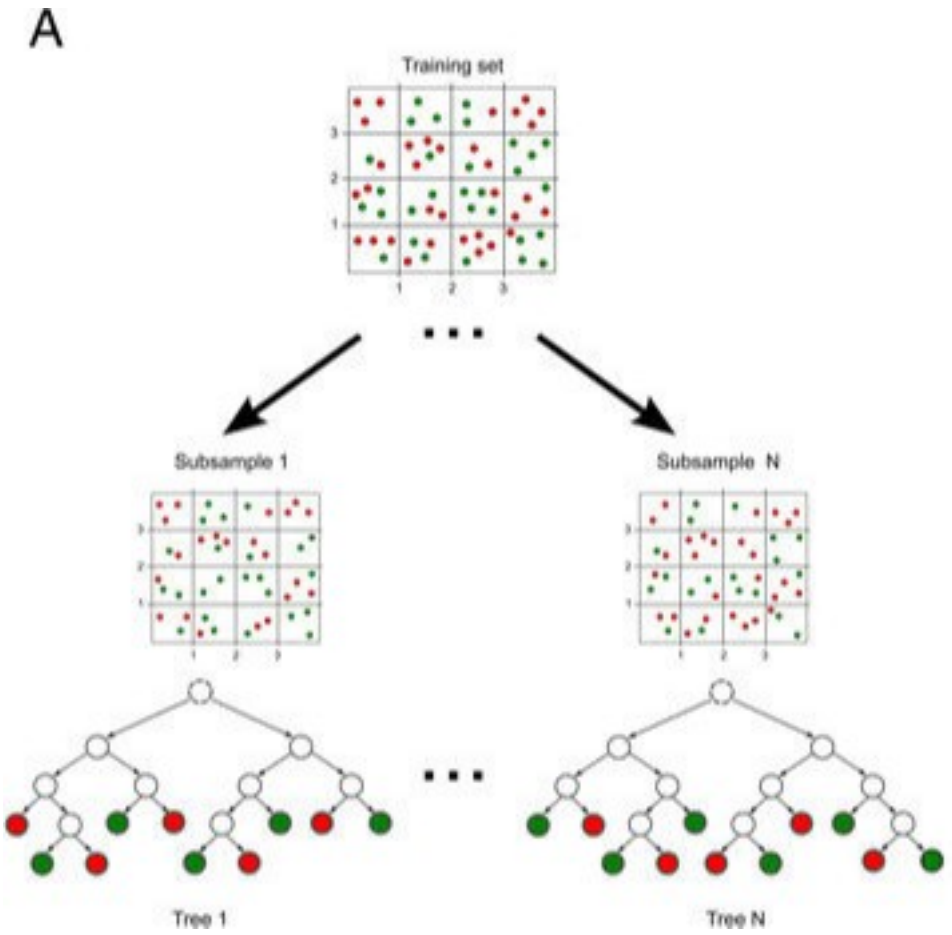

Figure 4.4: This figure shows the processes of training random forests classifiers and bagged trees. For random forests, samples of the features and observations are selecting for fitting a decision tree without pruning. For bagged trees, samples of the observations are selected to fit decision trees with pruning (Machado, Recamonde-Mendoza Corbellini 2015).

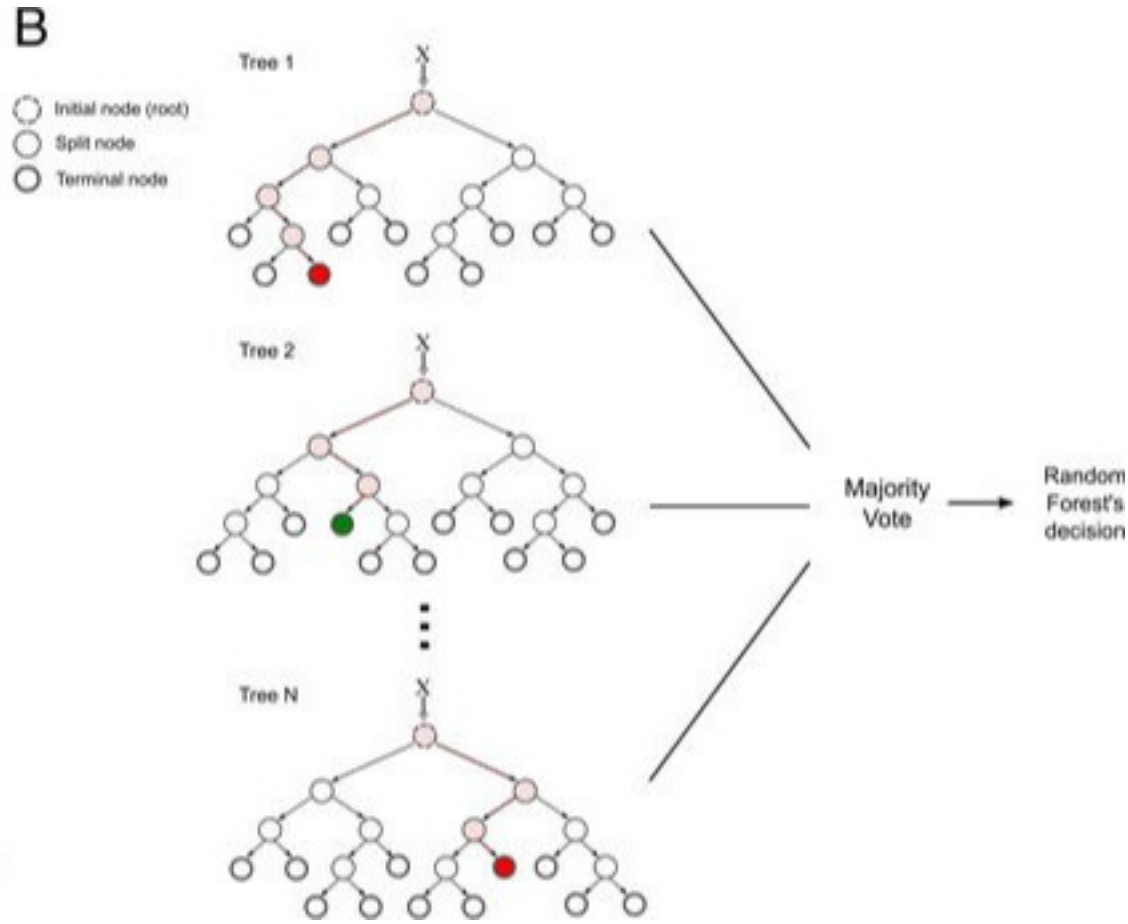

Figure 4.5: This figure shows the classification process for a new observation x when using random forests classifiers or bagged trees. The observation is run through the sub-models of these classifiers and the majority vote is assigned to the unlabelled observation (Machado, Recamonde-Mendoza Corbellini 2015).

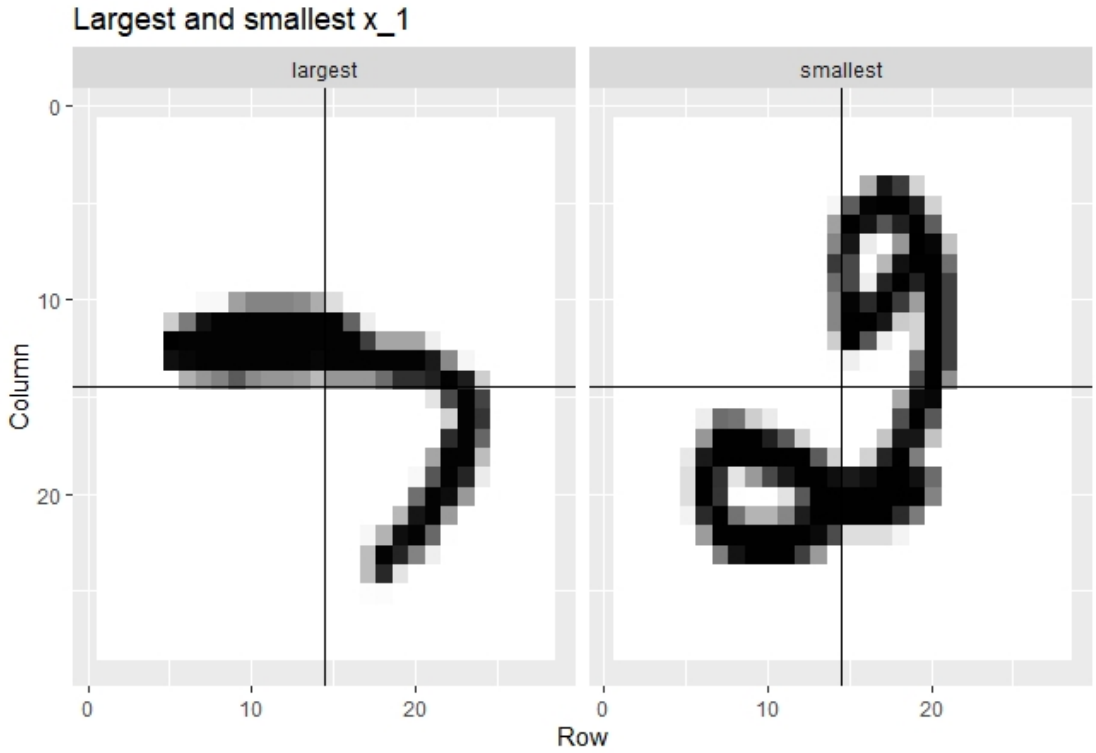

Figure 4.6: MNIST images of the number 2 and 7. The images are divided into four quadrants. The proportion of points in the top-left quadrant is x_1 and the proportion of points in the bottom right quadrant is x_2.

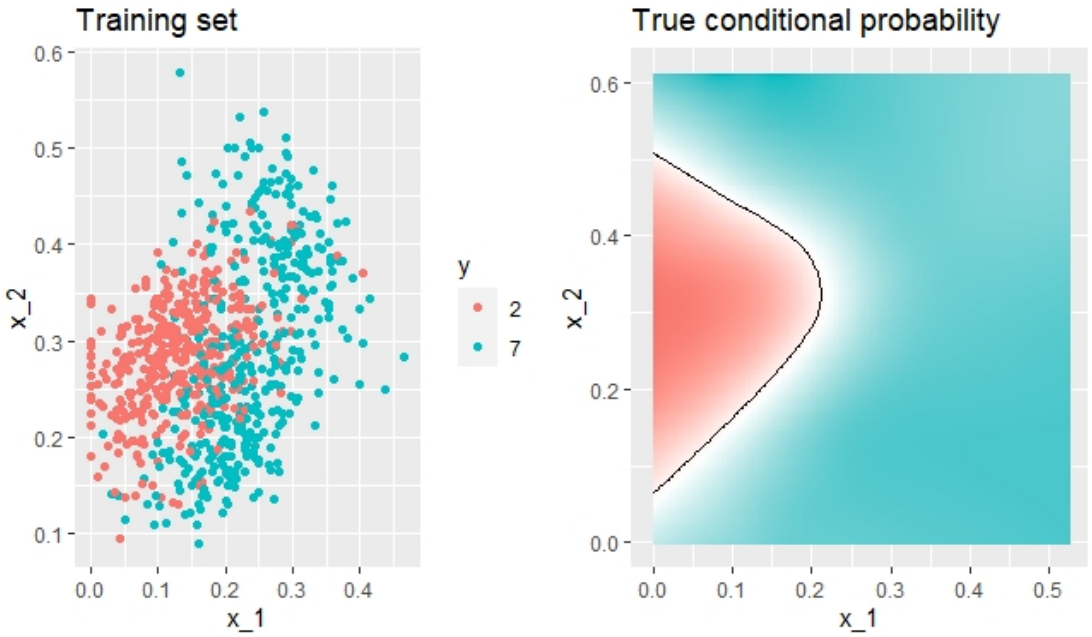

Figure 4.7: A scatterplot of training data and an estimate of the true conditional class probabilities. The red colour represents the number two and the blue colour represents the number 7. The white regions contain a mixture of probabilities and the black line shows the points of equal probability.

through some form or grid or range search. In these cases, the number of trees to be trained and the minimum split size for the tree were varied on different samples of the training data and verified with the unused samples.

In summary, both trees produce similar plots, however, some differences exist. The random forest boundary appears to be complex, but closer to the true distribution. The decision tree boundary is simple and covers the bulk of the true conditional class distribution. Given the bias in the data both models were biased, but for random forests the region of bias was smaller and fainter. Thus, using the probabilistic nature of random forests the classification thresholds can be adjusted to reduce bias. In Chapter 5 this example continues, in there the model performances will be compared the support vector machines.

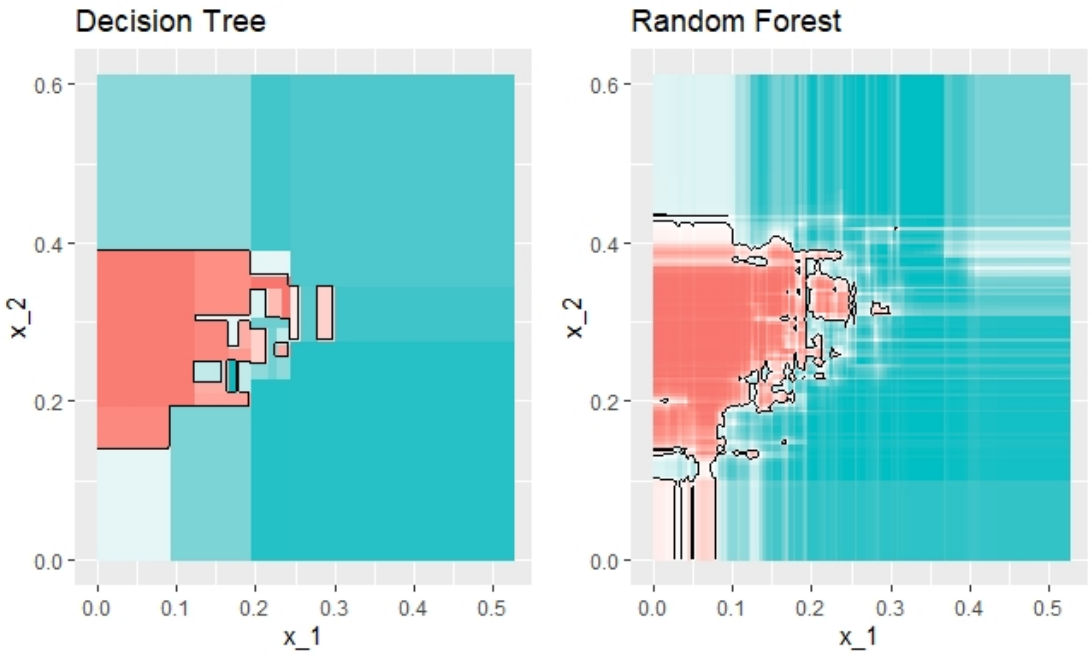

Figure 4.8: The decision boundaries for the bagged decision trees and random forests are shown above. The red colour represents observations that would be classified as images of the number two, and the blue regions represent observations to be classified as images of the number seven.

Chapter 5

The Support Vector Machine

Scientists working on image classification problems in the 1990s noticed neural networks had a problem converging on local maxima, to their rescue came Support Vector Machines (SVM) (Boser, Guyon & Vapnik 1996). SVMs are a discriminating method that use the dot product of vectors to find an optimal decision boundary which is also a global maximum boundary. Vapnik showed how transformations of the feature space allow non-linearly separable data to be classified (Boser, Guyon & Vapnik 1996).

5.1 Background and Introduction

The basic idea behind the support vector machine is that, given two classes of data Π_1 and Π_2, there exists a vector where the distance between two classes is maximised. The distance called the margin is measured between observations on the boundaries of each class. The observations are called support vectors. Thus, the support vector machine is a way of selecting boundary points in each class, in order to maximise the margin between them. A plane perpendicular to these points then acts as a decision boundary. Figure 5.1 shows four points selected as boundary points in order to maximise the margin.

5.1.1 Method

If we assume the training set can be separated by a hyperplane,

$$\mathbf{x} : f(\mathbf{x}) = \beta_0 + \mathbf{x}^T\beta = 0$$

where observations $\mathbf{x}_i$ closest to the plane are called support vectors and $f(\mathbf{x}) = 0$, then for points not on the plane, $f(\mathbf{x}) \neq 0$. Thus, for points where $f(\mathbf{x})$ is greater than zero,

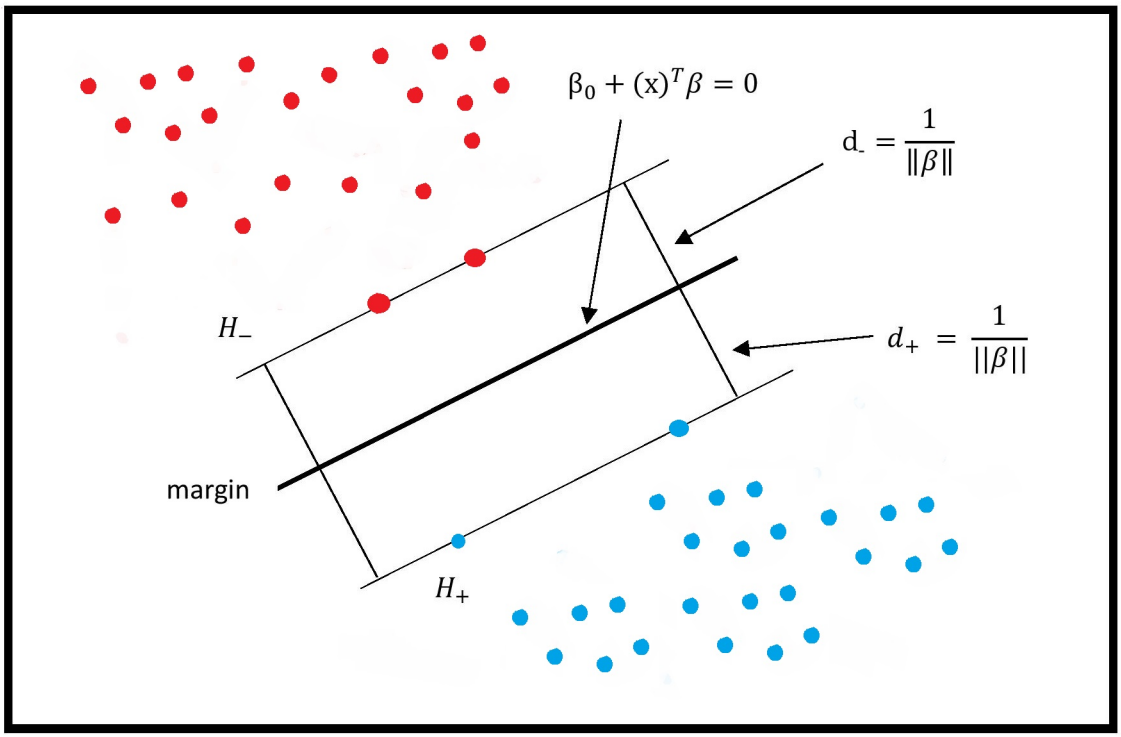

Figure 5.1: The margins of the separating hyperplane.

the class Π_2 is assigned and those where $f(\mathbf{x})$ is less than zero are assigned Π_1. Thus, the hyperplane divides the p-dimensional space in two as shown in Figure 5.1.

For any separating hyperplane. Let d_1 be the shortest distance from the separating hyperplane to the nearest observation in class Π_1, and let d_2 be the shortest distance from the same hyperplane to the nearest observation in class Π_2. As shown in Figure 5.1 where d_- is d_1 and d_+ is d_2. In Figure 5.1 the planes H_- and H_+ represent H_1 and H_2, as observations in Π_1 will have a negative dot product and those in Π_2 will have a positive dot product. Thus, we define Y,

$$y_i = \begin{cases} +1 \text{ if } \mathbf{x} \in \Pi_1 \\ -1 \quad \text{otherwise} \end{cases}$$

Then, the margin of the separating hyperplane is defined as d $= d_1 + d_2$. If, in addition, the distance between the hyperplane and its closest observation is maximized, we say that the hyperplane is an optimal separating hyperplane.

If the two classes in the training data are linearly separable, there exists β_0 and β such that

$$\beta_0 + \mathbf{x}_i^T\beta \geq +1, \; if \; y_i = +1, \tag{5.1}$$

$$\beta_0 + \mathbf{x}_i^T\beta \leq -1,\ if\ y_i = -1, \tag{5.2}$$

If there are data vectors in L such that equality holds in eqn. 5.1, then these data vectors lie on the hyperplane

$$\mathbf{H}_2 : (\beta_0 - 1) + \mathbf{x}^T\beta = 0 \tag{5.3}$$

similarly, if there are data vectors in L such that equality holds in eqn.5.2, then these data vectors lie on the hyperplane

$$\mathbf{H}_2 : (\beta_0 + 1) + \mathbf{x}^T\beta = 0 \tag{5.4}$$

Points in L that lie on either one of the hyperplanes $\mathbf{H}_1$ or $\mathbf{H}_2$, are said to be support vectors. Not all points will be support vectors. If x_1, and x_2 lie on the hyperplanes H_1, and H_2, respectively, then

$$\beta_0 + \mathbf{x}_1^T\beta = -1,\ \beta_0 + \mathbf{x}_2^T\beta = +1,$$

Using these equations the difference, $\mathbf{x}_2^T\beta - \mathbf{x}_1^T\beta = 2$, and the sum, $\beta_0 = 1/2(\mathbf{x}_2^T\beta + \mathbf{x}_1^T\beta)$ can be used to calculate the margin. By using the perpendicular distances of a support vector to the hyperplane, $\beta_0 + \mathbf{x}^T\beta = 0$.

$$d_1 = \frac{|\beta_0 + \mathbf{x}_1^T\beta|}{||\beta||} = \frac{1}{||\beta||} \tag{5.5}$$

$$d_2 = \frac{|\beta_0 + \mathbf{x}_2^T\beta|}{||\beta||} = \frac{1}{||\beta||} \tag{5.6}$$

$$d = d_1 + d_2 = \frac{2}{||\beta||} \tag{5.7}$$

So the margin of the separating hyperplane is $d = \frac{2}{||\beta||}$. The inequalities can be combined into a single set of inequalities,

$$y_i(\beta_0 + \mathbf{x}_i^T\beta) \geq +1,\ i = 1, 2, ..n. \tag{5.8}$$

The quantity $y_i(\beta_0 + \mathbf{x}_i^T\beta)$ is called the margin of $(\mathbf{x}_i, y_i)$ with respect to the hyperplane, $\mathbf{x}_i$ is a support vector if its margin equals one, $y_i(\beta_0 + \mathbf{x}_i^T\beta) = 1$. To find the best support vector we find the hyperplane that maximises the margin, $2/||\beta||$, but this is inconvenient as we intend to use Lagrangian multipliers and upon differentiation the solution will be complicated, instead we consider minimising $\frac{1}{2}||\beta||^2$ which differentiates perfectly to $\|\beta\|$. The minimisation constraint is subject to eqn.5.8.

$$\text{minimise}\ \ \frac{1}{2}||\beta||^2 \tag{5.9}$$

$$\text{subject to } y_i(\beta_0 + \mathbf{x}_i^T\beta) \geq +1, i = 1, 2, ..n. \tag{5.10}$$

The minimisation problem is more intuitive than the maximisation problem, as a quadratic function is being minimised against a linear function, thus, it is a convex optimisation problem. This is important to ensure any local minima will also be global minima and second order conditions are necessary, this solution was key to the adoption of support vector machine to classification problems as neural networks suffered from local minima and maxima when used for image and text classification.

The first part of optimisation is with the variables β_0 and β. Using Lagrangian multipliers we have the primal optimisation problem,

$$F_p(\beta_0, \beta, \lambda) = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{n} \lambda_i(y_i(\beta_0 + \mathbf{x}_i^T\beta) - 1) \tag{5.11}$$

where $\lambda = (\lambda_1, ..., \lambda_n)^T \geq 0$, first order conditions are taken.

$$\frac{\delta F_p(\beta_0, \beta, \lambda)}{\delta \beta_0} = -\sum_{i=1}^{n} \lambda_i y_i = 0 \tag{5.12}$$

$$\frac{\delta F_p(\beta_0, \beta, \lambda)}{\delta \beta} = \beta - \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i = 0 \tag{5.13}$$

$$y_i(\beta_0 + \mathbf{x}_i^T\beta) - 1 \geq 0 \tag{5.14}$$

$$\lambda_i \geq 0 \tag{5.15}$$

$$\lambda(y_i(\beta_0 + \mathbf{x}_i^T\beta) - 1) = 0 \tag{5.16}$$

eqn. 5.16 is the *Karush-Kuhn-Tucker* complimentarity condition, the theorem holds that the optimal point is a saddle-point and if it holds for β it also holds for β_0. With the conditions in eqn. 5.12 and eqn. 5.13,it is possible to derive,

$$\sum_{i=1}^{n} \lambda_i y_i = 0 \tag{5.17}$$

and

$$\beta^* = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i. \tag{5.18}$$

Replacing equations eqn.5.17 and eqn.5.18 back into eqn.5.11, we can factor out the primal variables it is easier to solve to the dual optimisation problem, where only λ is

estimated by,

$$\begin{aligned} F_d(\lambda) &= \frac{1}{2}||\beta^*||^2 - \sum_{i=1}^{n} \lambda_i (y_i(\beta_0^* + \mathbf{x}_i^T \beta^*) - 1) \\ &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=i}^{n} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j^T) - \sum_{i=1}^{n} \sum_{j=i}^{n} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j^T) + \sum_{i=1}^{n} \lambda_i \\ &= \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=i}^{n} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \end{aligned}$$

This is called the *Wolfe dual* or the dual optimisation problem; with it λ is found by maximising the equation subject to the constraints, $\lambda \geq \mathbf{0}$, $\lambda^T \mathbf{y} = \mathbf{0}$, where $\mathbf{y} = (y_i, ..., y_n)^T$ and $\mathbf{H} = (H_{ij}) = y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$ is a square (n x n) matrix. Thus, re-written as,

$$\text{maximise } F_D(\lambda) = 1_n^T \lambda - \frac{1}{2} \lambda^T \mathbf{H} \lambda. \tag{5.19}$$

The values of $\widetilde{\lambda}$ that solves the equation, can be replaced in eqn.5.18 to find, the optimal weight vector β^*,

$$\beta^* = \sum_{i=1}^{n} \widetilde{\lambda}_i y_i \mathbf{x}_i = \sum_{i \in sv} \widetilde{\lambda}_i y_i \mathbf{x}_i \tag{5.20}$$

If $\widetilde{\lambda}_i > 0$, from eqn. 5.16, the Karush-Kuhn-Tucker complimentarity condition it follows that $y_i(\beta_0 + x_i^T \beta) = 1$, thus, x should be a support vector. Then, let sv be the subset of indices for support vectors. Hence, for all observations that are not support vectors, $\tilde{\lambda}_i = 0$, thus, β is a linear function only for the support vectors.

Thus, the boundary will only depend on a few observations in the training set. However, the boundary may not be ideal if the data is linearly inseparable, thus, through cross-validation some observations are dropped. This will allow outliers or highly influential points to be ignored when selecting the best hyperplane. The cross-validation process will compare the test accuracies of subsamples top determine which boundary values can be dropped to improve accuracy. However, this method may not always apply.

When dealing with complex data structures, where a curve or complex boundary may be required kernels may be used. Kernels transform the features space to another dimension where it may be separable, through the methods discussed in section 5.1.1. Four popular kernels exist, the 'Linear', 'Sigmoid' 'Polynomial' and 'Radial Basis Function'. In fact the logistic regression discussed in section 3.4 is a special instance of the support vector classifier.

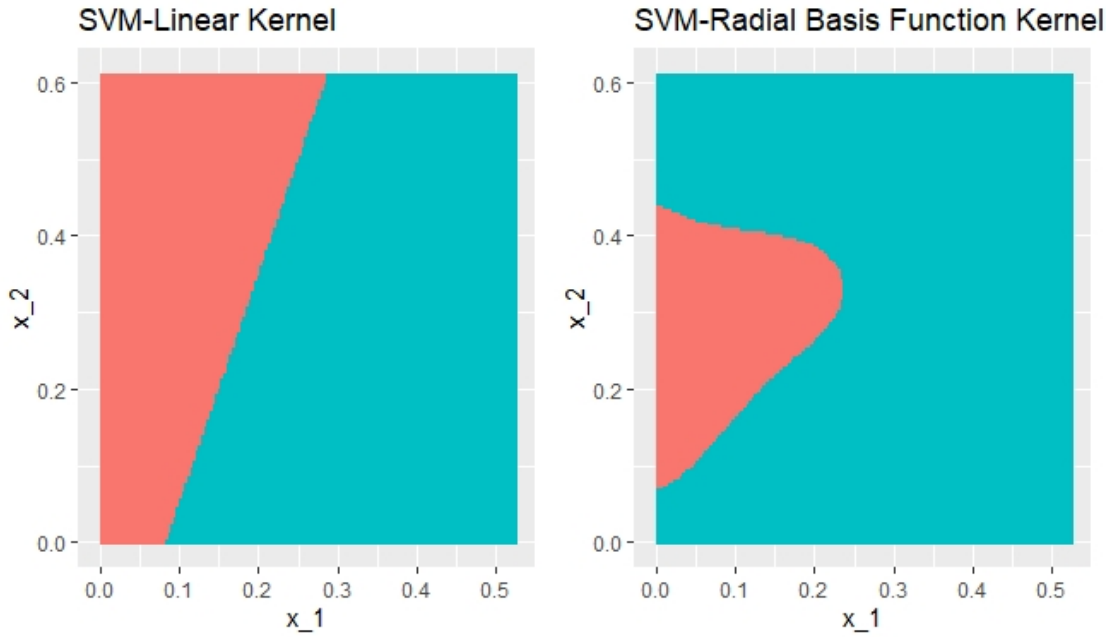

Figure 5.2: This figure shows the decision boundaries for support vector machine models trained on the MNIST data. On the left, the linear kernel produces a straight line. On the right, the radial basis function produces a curved boundary closer to the true conditional class probability function. The red regions represent images to be classified as the number two, whereas the red regions are classified as the number seven.

5.2 Handwritten digit data

Following section 4.5.2, the handwritten-digit data is used to train two support vector classifiers. The linear support vector classifier as shown is section 5.1.1 is used to create a decision boundary. The same is done for a radial basis function support vector classifier. The aim is to compare the test accuracies and decision boundaries between the tree-based classifiers and the support vector classifiers and form a critique.

Train and Test Accuracies for models trained on MNIST data.		
Model	Train Accuracy	Test Accuracy
Linear kernel - SVM	80.00%	75.50%
Radial Basis Function - SVM	84.63%	83.50%
Bagged Tree Classifier	84.63%	80.50%
Random Forest Classifier	99.65%	79.50%

Table 5.1: Train and test accuracies for classification models trained on MNIST data.

5.3 Summary

From the test results in Table 6.1, the best model was the support vector machine under the radial basis function. It had a significant lead on the other methods will strong consistency between train and test values. However, this may be as a result of poor cross-validation in the other models. The tree-based classifiers performed well in general, despite the possibility of overfitting in the random forests, as it had a high train score. The tree-based classifiers still have an advantage over support vector machines as bagging results in a probability distribution shown by the fainter regions in Figure 4.8. Overall, both methods have advantages which may aid the supervisor.

Chapter 6

X-ray image Applications

In this section we test the applicability of the classification methods to X-ray images. The data consists of 3886 chest X-ray images, they are a compilation of X-rays from machine learning competitions on Kaggle, a machine learning competition hosting service. The idea of analysing this database was inspired by the lack of statistical methods on the competition entries. Another motivation to solving this problem is the great benefit it may have to society, given that its effective implementation can improve the monitoring of the Covid-19 pandemic. The last inspiration for studying this topic is the work of Chowdhury et al. who use a smaller database and convolutionary neural networks to classify the images (Chowdhury et al. 2020). In solving this problem we hope to see an improvement or at a minimum comparable performance to the work of Chowdhury et al. To this end, we employ the methods discussed in this paper.

6.1 Data and Background

For the X-ray of normal lungs we found, 1341 images an example of the images are given in Figure 6.3. From this image the chest area, the ribs, heart and lungs are visible. Each of the normal images has 1024 x 1024 pixels. For X-ray images of lungs infected with Covid-19 there was a lot of variation in the 1200. Generally the Covid X-ray were of a lower quality than the other images and Chowdhury results may have been due to over-fitting rather can actual classification power as most Covid images were skewed and uncentered. However, the X-ray images of lungs infected with Covid-19 are anatomically similar to those of normal X-rays, however, the ribs are not as clearly visible possibly due to a lower penetration rate by the X-rays. As can be seen from these images, some clusters are visible in the viral pneumonia patients and Covid-19 patients. The images also differ in quality and perspective. There were 1345 images of viral pneumonia they act as a control against over-fitting between the Covid data and the normal data as the

Figure 6.1: An X-ray image of a healthy person taken from the front

Figure 6.2: An X-ray image of a person infected with Covid-19. The arrows show the characteristics of Covid-19 images

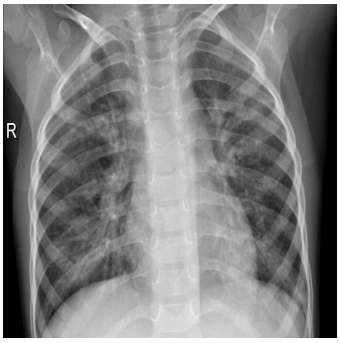

Figure 6.3: An X-ray image of a person infected with viral pneumonia taken from the front

images seem to follow no clear pattern by visual inspection.

6.1.1 Feature engineering

Given the larger number of dimensions 10^6 some dimension reduction is required (Cumming & Wooff 2007). Following the methods shown for analysing the handwritten digit data in Chapters 4 and 5, the images are reduced to fit a 32 x 32 grid. This is done with the **opencv package** in the **Python programming language** (van Rossum 1995) (Bradski 2000). We then flatten the data from a 32 x 32 grid into a list of 1024 features, this represents the feature space $\mathbf{X}$ and the condition represented in the X-rays represent the classes Π_i for i in 1,2,3, for "Normal", "Covid-19" and "Viral Pneumonia". We then also perform principal component analysis to gain some insights into the data, but also to reduce the dimensionality further.

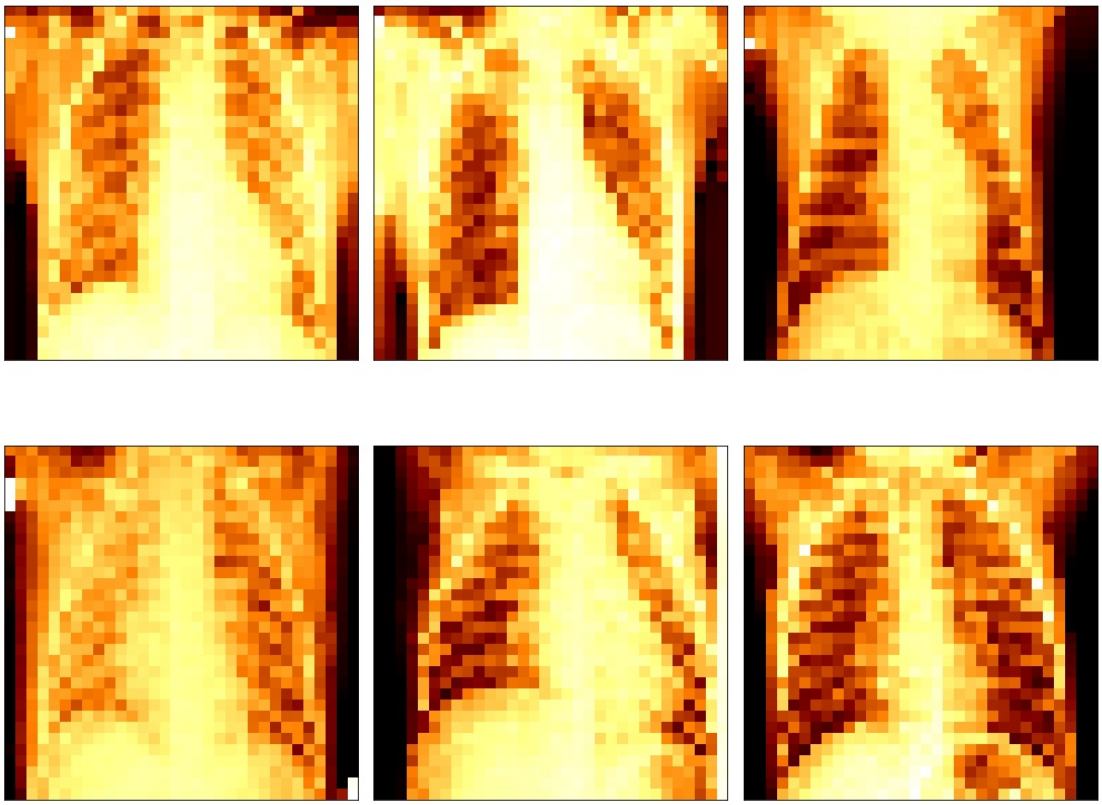

Figure 6.4: The images after being reduced to a 32 x 32 grid.

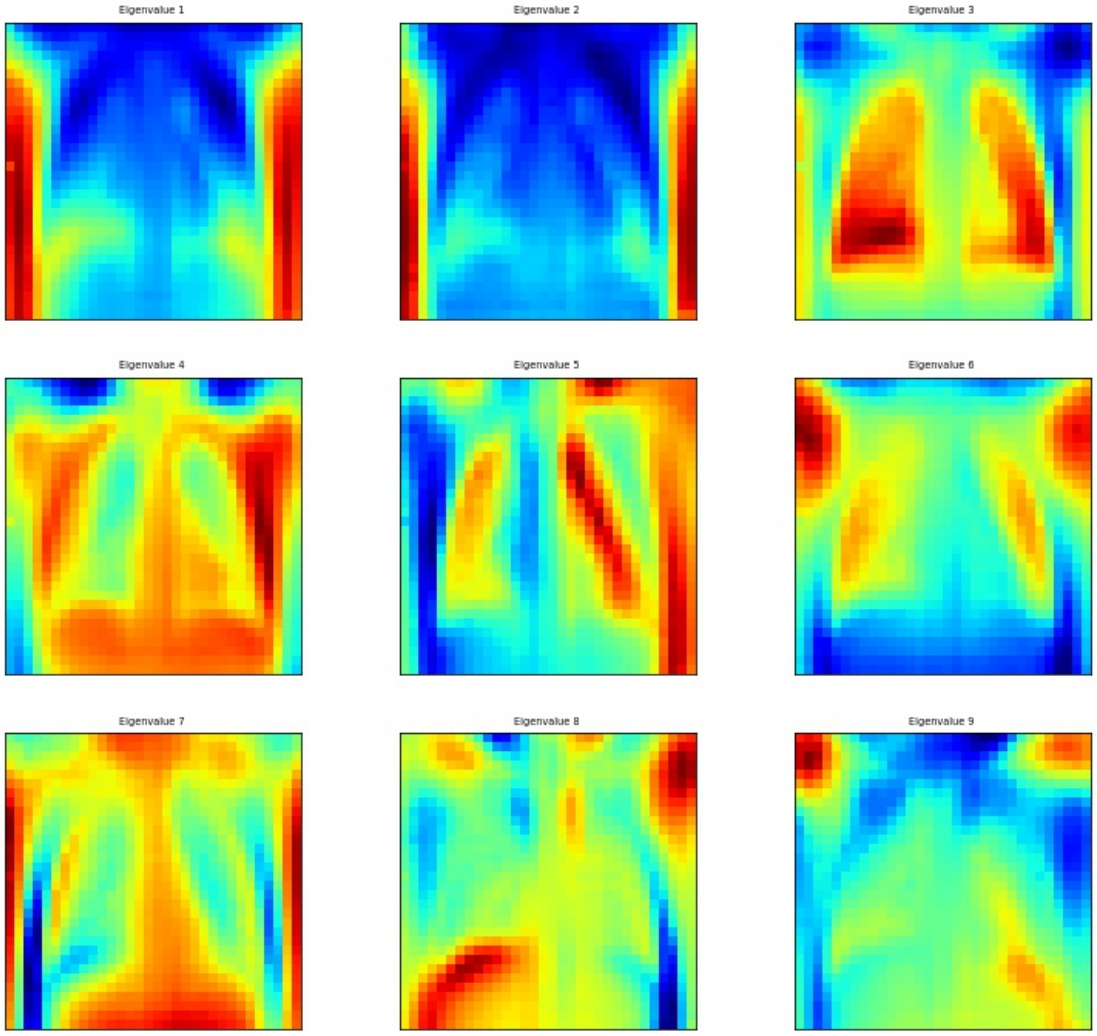

Figure 6.5: These are the top nine eigenvectors of the feature space. Cumulatively they accounted for 62% of the variance alone.

6.2 Analysis

Given the reduced feature space $\mathbf{X}$ and the list of classes Π_i, we train the models discussed earlier. The use of the **sci-kit learn** package in the Python programming language was used to train the models (Pedregosa et al. 2011).

Test accuracies for models trained on X-ray image data with an 80-20 split.			
Model	Full image 32 x 32 pixels	60 principal components (95% of variation)	10 principal components (65 % of variation)
Radial Basis Function - SVM	94.60%	95.76%	89.46%
Linear kernel - SVM	92.67%	93.70%	85.86%
Logistic Regression	91.90%	92.80%	84.70%
Classification Tree Classifier	80.59%	82.26%	82.26%
Random Forest Classifier	93.31%	93.70%	88.81%
K-Nearest Neighbour Classifier	92.80%	94.22%	87.53%
Linear discriminant Analysis	89.72%	92.29%	84.70%
Quadratic Discriminant Analysis	57.32%	94.85%	86.89%
Stacked Classifier	94.99%	95.78%	89.59%

Table 6.1: Test set accuracies for classification models trained on X-ray image data.

In Table 6.1, we can see the performance of different classifiers on different versions of the data. Across all the types of data the stacked classifier was the best performing classifier. Which was expected as it was created by combining the results of the other classifiers and using a logistic regression model to find the best means of combining them. The difference between the stacked classifier and the radial basis function support vector classifier was minimal. This would be expected as the radial basis function considers the distance between points in an infinite number of dimensions. And the stacked classifier may have given a larger weighting to the results from this model. It was interesting to note, the consistency of the classification tree across different levels of data. Insights from the tree may be useful in explaining the determinants of these disease when using the first 10 principal components. However, it was only able to produce an accuracy of $\approx 82\%$. Generally, all classifiers did well with the 32x32 image as well as with the different number of principal components.

Chapter 7

Conclusion and Recommendations

The methods explored in this paper have applied various supervised learning methods to image classification. Different challenges were covered through the use of traditional classifiers, maximum margin classifiers and ensemble classifiers. Each type of classifier had its own advantages, but the combined effect of all classifiers together was superior to any one classifier.

The first and second chapters introduced the problem of classification and supervised learning.

The third chapter covered the earliest classifiers to be defined. Using the Iris data the chapter also showed how the classifiers would provide similar insights. What was common was the ability of all classifiers to create a general classification rule, quite similar to one discussed in chapter 2. Given sufficiently large training data all models performed considerably well in predicting the classes for iris species based on the measurements alone.

The fourth chapter covered all tree-based classifiers. The classification trees and random forests. Using these methods an image classification problem was solved using features extracted from the data after dimension reduction. Both methods produced good decision boundaries based on the data, however, some bias was visually evident. Nonetheless both methods were able to produce results with high accuracy. Thus, this chapter demonstrated the ability to analyse images and the predictive power possible from combining many weak classifiers.

The fifth chapter introduced the support vector classifiers. Building on the example and data in chapter 4, some improvements were shown. The support vector classifiers

were able to produce a highly accurate decision boundaries and test accuracies when tested image data. In fact it was shown that the use of the radial basis function as a kernel produced the best results. However, this model was unable to produce a probabilistic response.

The sixth chapter building on the methods shown in previous chapters was able to compare all the methods discussed with x-ray images after applying various methods of dimension reduction. The results from all these models were then used to train a logistic regression classifier. The ensemble classifier was able to obtain the highest accuracy among all the models, while retaining the probabilistic nature expected from logistic regression.

7.1 Conclusion

All the classifiers discussed can experience problems of overfitting and underfitting. With little or biased data this may be difficult to determine, but by splitting data into train and test sets overfitting can be reduced. This is done by noting the any variance in performance. Underfitting on the other hand is difficult to notice, unless multiple models are compared, and some visual inspection is carried out. However, the use of bagging, random forests, logistic regression and support vector classifiers is unlikely to result in underfitting. Table 7.1 summarises the size of the problem for the different models.

Table 7.1 shows a comparison of the various models. What is immediately obvious is the need for some models to be cross-validated to control for over or under fitting. This is a key quality of supervised learning as the model is changed to meet some desired attributes. In the cases where cross-validation is not possible the use of probabilities can provide a means of adjusting classification thresholds to produce a desired output. From the table we see the ensemble classifiers like random forests, bagged trees and the stacked classifier achieve the sweet spot of allowing flexibility through cross-validation and probabilistic outputs. The other models share some ideal properties, but some challenges may arise. For example, the linear and quadratic discriminant analyses provide probabilistic outputs, but they are quite inflexible to adjusts when compared to the K-nearest neighbour as was shown in Chapter 3.

A strong observation was the performance of ensemble classifiers and margin classifiers. These were the random forests, the stacked classifiers; and the support vector machines and logistic regression respectively. The ensemble classifiers allowed the grouping of many week classifiers to create one strong classifier by using a system of voting. The margin based classifiers based themselves on finding the best means to split the data without actually considering the data. The support vector machine only used support vectors for classification and the selection of these was a global maximisation problem.

A comparison of model attributes				
Model	Over-fitting	Under-fitting	Cross-Validation	Probabilistic
K-nearest neighbour	Yes	Yes	Required	Possibly
Linear discriminant analysis	No	Yes	Not required	Yes
Quadratic discriminant analysis	No	Yes	Not required	Yes
Logistic Regression	No	No	Not required	Yes
Classification trees	Yes	Yes	Required	No
Bagged classification trees	No	Yes	Possibly	Possibly
Random forests	No	No	Not required	Possibly
Support Vector classifiers	Yes	No	Required	No
Stacked classifier classifiers	No	No	Possibly	Possibly

Table 7.1: A comparison of model attributes.

The same was true for the logistic regression model which used the Newton-Raphson method, a non-parametric method of determining the best parameters to model the log-odds. Thus, these methods are only focused on finding the best means of splitting the data, without relying on any distributional assumptions, or measures of similarity like seen with the discriminant analyses, K-nearest neighbour methods and the tree-based classifiers.

Using the strengths of many as seen with ensembles and independence of the non-parametric methods, some combination of the two can prove useful. Thus, out stacked classifier was shown to be the best here. But in the future, some form of adaptive boosting may be used. This is when the errors are re-weighted and used to train new models. This could provide the strengths in numbers required for unbiased analysis, but also maximising on any other unmodelled distributional changes.

However, the traditional classifiers and the tree-based classifiers remain useful for two reasons. Firstly, they provide simple explanations humans can understand. Secondly, the goal of complete accuracy should be avoided as noted in this paper emphasis of respecting the Bayes error as a true limit to classification problems. This can also help avoid spurious correlations. Thus, the use of these methods should be a multi-disciplinary approach encompassing, the means of data collection be it governance and economic bodies, but also include the means of interpreting results be it the various fields of academia, the public, governments and businesses.

In summary, supervised learning and statistical classification are useful tools for three reasons. Firstly, these methods can help extend human understanding by summarising hugely complex information with visual aids and mathematical relationships. Secondly, these methods can be extremely accurate, sometimes at the expense of explainability. Lastly, they can handle a wide range of problems when used skillfully and appropriately.

7.2 Recommendations

1. Only accuracy was used as the desired loss function, precision and recall would have been suitable alternatives given the intended use of the classifications.

2. The logistic regression and discriminant analysis methods were simplified, but yet proved equally useful in the final image classification. Further study and analysis of these methods could lead to fruitful insights.

3. Cross-validation techniques for the K-nearest neighbour, support vector machine and tree-based classifiers could have been discussed in more detail.

4. The dimension reduction methods were not discussed in great detail and more methodical approaches would have led to better noise reduction and information retrieval.

5. The final model could be used to provide probabilities of Covid-19 and Viral Pneumonia, thus, making it more useful to health authorities.

6. A support vector classifier could be used to create a stacked classifier with possible better results than the logistic regression stacked classifier.

7. Random classification trees created on samples of the data could provide more insights for a stacked classifier to work with.

8. Using a standardised dataset would help make models more useful. As the images used present with a large proportion of noise and the analysis using the first 10 principal components affirmed this.

Bibliography

Agresti, A. 2007. *An Introduction to Categorical Data Analysis.* Hoboken, NJ:John Wiley Sons.

Albert, A., and J. A. Anderson. 1984. "On the existence of maximum likelihood estimates in logistic regression models." *Biometrika*, 71(1): 1–10.

Beyer K., Goldstein J., Ramakrishnan R. Shaft U. 1999. "When Is "Nearest Neighbor" Meaningful?" 1540.

Boser, Bernhard, Isabelle Guyon, and Vladimir Vapnik. 1996. "A Training Algorithm for Optimal Margin Classifier." *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, 5.

Bradski, G. 2000. "The OpenCV Library." *Dr. Dobb's Journal of Software Tools.*

Breiman, L. 1996. "Bagging Predictors." *Machine Learning*, 24: 123–140.

Breiman, L. 2001. "Random Forests." *Machine Learning*, 45: 5–32.

Breiman, L., J. Friedman, C.J. Stone, and R.A. Olshen. 1984. *Classification and Regression Trees.* Taylor & Francis.

Browne, Michael W. 2000. "Cross-Validation Methods." *Journal of Mathematical Psychology*, 44(1): 108–132.

Chollet, Francois, et al. 2015. "Keras."

Chowdhury, Muhammad E. H., Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, and Mohammad Tariqul Islam. 2020. "Can AI Help in Screening Viral and COVID-19 Pneumonia?" *IEEE Access*, 8: 132665–132676.

Cover, T., and P. Hart. 1967. "Nearest neighbor pattern classification." *IEEE Transactions on Information Theory*, 13: 21–2.

Cumming, J. A., and D. A. Wooff. 2007. "Dimension reduction via principal variables." *Computational statistics & data analysis.*, 52(1): 550–565.

D. Michie, D.J. Spiegelhalter, and C.C. Taylor. 1994. *Machine Learning, Neural and Statistical Classification.* Leads, UK.

Dudani, S. A. 1976. "The Distance-Weighted k-Nearest-Neighbor Rule." *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(4): 325–327.

Fisher, R.A. 1936. "The of Multiple measurements in taxonomic problems." *Machine Learning.*

Fukunaga, K. 1990. *Introduction to Statistical Pattern Recognition.* San Francisco, CA:Morgan Kaufmann.

G. James, D. Witten, T. Hastie, and R. Tibshirani. 2013. *An Introduction to Statistical Learning.* New York, NY:Springer.

Houlding, Brett, Frank P. A. Coolen, and Donnacha Bolger. 2015. "A Conjugate Class of Utility Functions for Sequential Decision Problems." *Risk Analysis*, 35(9): 1611–1622.

Irizarry, Rafael A. 2021. "Introduction to Data Science."

Kleinbaum, David G., and Mitchel Klein. 2010. *Logistic Regression: A self learning text.* Verlag, NY:Spinger.

Kuhn, Max. 2008. "Building Predictive Models in R Using the caret Package." *Journal of Statistical Software, Articles*, 28(5): 1–26.

Lachenbruch, P. A., and M. Goldstein. 1979. "Discriminant Analysis." *Biometrics*, 35(1): 69–85.

LeCun, Yann, Corinna Cortes, and CJ Burges. 2010. "MNIST handwritten digit database." *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Leibovich-Raveh, Tali, Daniel Lewis, Saja Al-Rubaiey, Kadhim, and Daniel Ansari. 2018. "A new method for calculating individual subitizing ranges."

Machado, Gustavo, Mariana Recamonde-Mendoza, and Luís Corbellini. 2015. "What variables are important in predicting bovine viral diarrhea virus? A random forest approach." *Veterinary Research*, 46: 85.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, 12: 2825–2830.

R Core Team. 2017. "R: A Language and Environment for Statistical Computing." Vienna, Austria, R Foundation for Statistical Computing.

Rust, John. 1997. "Using Randomization to Break the Curse of Dimensionality." *Econometrica*, 65(3): 487–516.

Utkin, Lev V., Maxim S. Kovalev, and Frank P.A. Coolen. 2020. "Imprecise weighted extensions of random forests for classification and regression." *Applied Soft Computing*, 92: 106324.

van Rossum, G. 1995. "Python tutorial." Centrum voor Wiskunde en Informatica (CWI) CS-R9526, Amsterdam.

Weiss, S.M., and C.A. Kulikowski. 1991. *Computer Systems that Learn.* San Mateo, CA:Morgan Kaufmann.

Wickham, Hadley. 2016. *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, Max Kuhn, Thomas Lin Pedersen, Evan Miller, Stephan Milton Bache, Kirill Müller, Jeroen Ooms, David Robinson, Dana Paige Seidel, Vitalie Spinu, Kohske Takahashi, Davis Vaughan, Claus Wilke, Kara Woo, and Hiroaki Yutani. 2019. "Welcome to the tidyverse." *Journal of Open Source Software*, 4(43): 1686.

Yee, Thomas W., and C. J. Wild. 1996. "Vector Generalized Additive Models." *Journal of Royal Statistical Society, Series B*, 58(3): 481–493.

Appendix

1. Iris data analysis Rscripts `https://github.com/KBMP111/iris-data-classification`
2. Python notebooks for X-ray image analysis code `https://github.com/KBMP111/X-ray-Image-Analysis`