

Data visualisation and machine learning web application with potential use in sports data analytics

Cathal Dooley

March 30, 2017



Student ID: 13533663

Class: 4ECE

Supervisor: Dr. Michael Schukat

Abstract

This project deals with the design and implementation of a machine learning and statistical analysis web application used to model user data. The application takes the user dataset as an input and can graphically display the data and generate statistical models to describe trends that exist in the dataset.

The application allows the user to investigate their datasets by visually describing the data through the implementation of box plots, density plots and scatter plots. This application also allows the user to test various classification algorithms to classify the data and provide useful information to the user about the performance of the classification algorithms. The application provides clustering capabilities to discover trends or patterns in the data that would otherwise be unidentifiable.

Machine learning and data mining are becoming increasingly popular topics in computing, which highlights the need for applications such as this today.

Provided with appropriate data, the use of this application extends to many facets of life including health, nature, economics and sport. This thesis delves into the inner-workings of the application and highlights the potential of its use.

Acknowledgments

I would like to thank my supervisor Dr. Michael Schukat, for his valuable support throughout this project.

I would like to express gratitude to my family by thanking my parents, Declan and Deirdre, my brothers, Stephen and Darren and my sisters Deanna and Lea for supporting me and keeping me in good spirits the whole way through.

I would also like to say a special thanks to my dad for staying up all night to proof read this thesis, not an easy job.

Contents

1	Introduction	1
1.1	Overview	1
1.2	Motivations	2
1.3	Research Questions and Objectives	2
1.3.1	Research Questions	2
1.3.2	Research Objectives	3
1.4	Thesis Structure	3
2	Literature Review	5
2.1	Data Visualisation Techniques	5
2.1.1	Box Plots	5
2.1.2	Density Plots	6
2.1.3	Scatter Plots	7
2.2	Machine Learning and Data Mining	8
2.2.1	Supervised Learning	8
2.2.2	Semi-Supervised Learning	8
2.2.3	Unsupervised Learning	8
2.3	Classification Analysis	9
2.3.1	Decision Tree Classification	9
2.3.2	Linear Discriminant Analysis Classification	11
2.3.3	K-Nearest Neighbours Classification	12
2.3.4	Support Vector Machine Classification	13
2.3.5	Naive Bayes Classification	14
2.4	Cluster Analysis	14
2.4.1	K-Means Clustering	15
2.5	Applications of Statistical Analysis in Sports Science	16
2.5.1	Biomarker Analysis	16
2.5.2	Case Study I	17
2.5.3	Case Study II	19
2.6	Machine Learning Software Currently Available	22
2.6.1	Orange	22
2.6.2	Weka	24
2.6.3	Python and Extended Libraries	25
2.6.4	R Programming	25
2.7	Project Influence	27

3	Developing the Application	28
3.1	Introducing the Application	28
3.2	Application Requirements	28
3.2.1	Graphical User Interface (GUI)	28
3.2.2	Reading and Describing the Dataset	29
3.2.3	Presentation and Visualisation	29
3.2.4	Machine Learning Algorithms	29
3.2.5	Analysis of Model Performance and Accuracy	29
3.3	Application Design	30
3.3.1	Phase I	30
3.3.2	Phase II	30
3.4	Application Components	33
3.4.1	Data Upload and Data Description	33
3.4.2	Missing Values and Data Plots	34
3.4.3	Classify	35
3.4.4	Cluster	36
3.4.5	Visualise	39
3.5	Testing, Verification and Comparative Analysis	40
3.5.1	Read CSV File and Separate Data	40
3.5.2	Verifying the Missing Values, Box Plots and Density Plots	40
3.5.3	Testing the Classification	44
3.5.4	Testing the Clustering	47
3.5.5	Scatter Plots	50
3.6	Improvements for the Application	51
4	Application Use in Sports Data Analysis	52
4.1	Implementation of the Application with Biomarker Data	52
4.2	Biomarker Dataset	52
4.3	Added Results Component	53
4.4	Experiment	54
4.4.1	Data Input and Visualisation	54
4.4.2	Classification	58
4.4.3	Cluster:	60
4.4.4	Other Visualisation:	63
5	Conclusions	64
5.1	Work Summary	64
5.2	Directions for Future Work	64
5.3	Concluding Remarks	65

List of Figures

1	Example of a labelled box plot.	6
2	Example density plots of Fisher's Iris dataset.	6
3	Example scatter plot with linear regression.	7
4	Dataset and corresponding decision tree. (Taken from 1617-CT475 Machine Learning and Data Mining, <i>Madden</i> [5])	10
5	KNN classification with $k = 3$ and $k = 6$	12
6	SVM Classification with optimal hyperplane and maximum margin.	14
7	Clustering with 3 clusters.	15
8	Plot of each biomarker over time with mean (black dot), smoothed trajectory (blue line) and 95% confidence interval displayed (dark shaded area). [10] . . .	18
9	Cophenetic correlation coefficient. [11]	20
10	Metagene matrix. [11]	21
11	Orange UI with detailed visualisations.	23
12	Weka UI.	24
13	RStudio console UI.	26
14	Front end of an example R Shiny web application.	27
15	Performance times using the DeployR driven application.	31
16	Design flow of the application.	32
17	GUI of Data Upload and Data Description panel.	33
18	GUI of Data Plots Panel with Box Plot Output.	34
19	GUI of Data Plots Panel with Density Plot Output.	35
20	GUI of Classify Panel.	36
21	GUI of Cluster panel with K-means cluster plot.	37
22	GUI of Cluster panel with hierarchical cluster plot.	38
23	GUI of Visualise panel.	39
24	Box plot of Petal Length by class. Created in Orange.	41
25	Box plot of Petal Length by class. Generated by the application.	42
26	Density plot of Sepal Length by class. Created in Orange.	43
27	Density plot of Sepal Length by class. Generated by the application.	44
28	Table of random sample of Iris data with associated classes.	46
29	Classification results of the unlabelled Iris sample.	47
30	K-means clustering of Sepal Length versus Petal Length with 3 clusters using Weka.	48
31	K-means clustering of Sepal Length versus Petal Length with 3 clusters using the application.	49
32	Scatter plot of Sepal Length versus Petal Length separated by class.	50
33	Biomarker classifications with associated attributes.	53
34	Risk and actions associated with the biomarker readings.	54
35	Percentage of missing values per variable.	55
36	Box plot representation of <i>vd</i> variable for all classes.	56

37	Density plot representation of vd variable for all classes.	57
38	Elbow curve for cluster number selection.	61
39	Cluster assignments for the cr and bt variables with $K=4$	62

List of Tables

1	Performance measures of the classification algorithms.	59
2	Predicted and actual class of the new sample athletes.	60
3	Cluster centres for the <i>vd</i> and <i>bt</i> variables	62

1 Introduction

The following thesis is concerned with the application of visualisation, classification and clustering of user data using a web application developed with open source technologies. The web application provides powerful statistical analytics to users and allows the user to grasp detailed knowledge of any dataset being investigated, tested and analysed.

The performance of the application in its ability to correctly plot, classify and cluster data is explored in this thesis. This section will outline a brief description of the main work carried out in this project.

1.1 Overview

This overview addresses the key aspects involved in the project.

The topic of machine learning data mining has become a prevalent part of computing in recent times with advancement in areas such as artificial intelligence. As a result, the importance of machine learning tasks, such as classification and cluster modelling, inspired the development of this application. The design and architecture of the classification and clustering models is followed by analysis and interpretation of results produced by the algorithms. The application requires data to be uploaded by the user which is then used for training, testing and validating the machine learning models. The aim of this application is to graphically represent this data and generate predictive results based on the classification or cluster models.

Firstly, the subject of data visualisation, machine learning, statistical analysis and data mining will be discussed. There is a great wealth of research being carried out in the area of data mining and machine learning today, with a broad range of highly useful applications. Some common applications include genetics based machine learning, which has been used in health research to detect genes that are indicative of diseases such as cancer in patients. Machine learning has also been applied to the area of genetics in sports, to detect biological traits that suggest fatigue, over training and injury in athletes. The scope of this machine learning application extends further than genetics to areas such as face detection and speech recognition also relying on common data mining and machine learning techniques. Existing technologies such as Orange and Weka are popular tools used for testing machine learning and data mining algorithms and provided inspiration in the development of the web application in this project and also acted as a comparative benchmark for testing the performance and results of the application. Much like Weka, the goal of the application is to provide users with the tools necessary to visualise their datasets and unlock hidden information by uncovering trends and patterns in the data.

Secondly, we look at the programming and computation involved in the application. For this project, the application was developed in R. R is an open-source programming language used for data analysis, statistical software and graphic visualisation. R allows users to deploy powerful web-based applications through the use of an R package called Shiny. The implementation of R's built-in visualisation, classification and clustering packages within the R Shiny

framework provided high performance statistical algorithms for the development of this application. R can be used for large scale data processing and performs quickly when processing heavy data loads so it was well suited for developing the web application. The potential of the application for use in sports data analysis is assessed by implementing the biomarker data of athletes with the application and testing, verifying and validating the results of this experiment.

The thesis closes with some remarks on future work which could be carried out in the wake of completing this project, along with some concluding words about the advantages of using the application for data visualisation and machine learning tasks.

1.2 Motivations

The motivation for this project stemmed from professional work experience I carried out during an internship at the Insight Centre for Data Analytics. At Insight, I was involved in a project relating to sport science and the investigation of biomarker data. I was part of a research team tasked with developing an application to investigate the biomarker data of athletes and draw conclusions from the investigations. This work interested me greatly and I saw the potential for me to follow up on all I had learned and implement this knowledge in my own project. The application ties into this area of analysis as it allows users to upload any data, including athlete biomarker data, to uncover trends in the data which could be used, in this case, to optimize athletic performance.

Machine learning is another interest of mine. I was exposed to basic machine learning and data mining concepts while working at Insight but it was not until I began studying machine learning and data mining under Dr. Michael Madden, NUIG, as part of my final academic year in university that my true interests in this area of computing were sparked. Exploring areas such as data manipulation, mathematics and computing also intrigue me so this project is a culmination of these interests.

1.3 Research Questions and Objectives

As discussed in Section 1.1, the aim of this project is to build a web-based application tool which can be used to model data, through classification and clustering. The following research questions and objectives have been identified based on this aim.

1.3.1 Research Questions

- 1. Can this application take a dataset as an input and render detailed plots of the data?*
- 2. Using this application, can users perform supervised and unsupervised learning techniques such as a range of classification and clustering algorithms on the data?*
- 3. Can a machine learning application, such as the one implemented in this project, be used in sports science to make predictions about athletic performance using biomarker data?*

1.3.2 Research Objectives

The following core research objectives were drawn up based on the preceding research questions.

- Develop an application to take user data as an input. Allow the user to learn the structure of the data by printing a table of information on the data. This data description will highlight the amount of observations, variables and classes per dataset. The user will also be able to view the dataset in its entirety. The option to visualise the data in detailed graphs will also be available to the user. Box plots and density plots will be used to get a descriptive view of the spread of the data and will visually represent key points in the dataset. The user will also be made aware of any missing values in the data. The completion of this research objective aims to answer research question 1 posed in the previous section.
- Design, develop and implement an application which can classify and cluster datasets. Carry out experimentation to validate the output of the classification and clustering to test the performance of the application for such tasks. This will determine the validity of the application in its ability to make predictions on new data based on the models that have been trained by the application. The completion of this research objective aims to answer research question 2 posed in the previous section.
- Research the use of predictive analysis applications in sports science. This involves understanding and implementing biomarker data with the application. In sports science, biomarkers are taken from athletes' blood samples and can indicate injury, health and ability to perform. This data is suitable for the application as the analysis of such data and the application of machine learning techniques can help make improvements in athletic performance. This analysis would lead to better decision making regarding when to train or rest or to determine the training load. The application will classify the data and will train different algorithms to create the most suitable model to classify new biomarker data at the highest level of accuracy. The application will also cluster the data to uncover hidden trends which may prove to be beneficial, given unlabelled biomarker data. The completion of this research objective aims to answer research question 3 posed in the previous section

1.4 Thesis Structure

This thesis is made up of five chapters in total, chapter 1 outlines the introduction to this project and provides an overview of the work.

Chapter 2 provides an extensive review of the previously stated research areas. It begins with by outlining common visualisation techniques that are employed in data analysis, followed by an introduction into the concepts of machine learning and data mining. The literature review then deals with understanding and implementing various machine learning techniques such as

classification and clustering algorithms. Next, some research into the use of statistical analysis in sports is considered with a description of biomarker data and its potential applications.

Finally, a technical review of existing technologies and programming languages and platforms is carried out.

Chapter 3 details the design, development and implementation process involved in creating the application. This chapter includes an introduction to the application and the requirements for the design. The key components of the application software are described in detail. The initial design phase is outlined which is followed by an improved design description which highlights added capabilities and improved features using a design flow diagram. The application is tested and validated by way of comparative analysis of the application output.

Chapter 4 is concerned with the implementation of the application in sports data analysis. This chapter includes the experimental research and testing of the application on a set of athlete biomarker data. Analysis of the results is carried out and conclusions are drawn from the experiment.

Finally, Chapter 5 provides concluding remarks on the project as a whole, including some recommendations for potential future work.

2 Literature Review

This chapter of the thesis deals with the gathering and analysing of key information required for this project. This section will delve into core research areas such as data visualisation and concepts of machine learning and data mining. Following a description of these topics, information on different classification and clustering algorithms will be presented. A detailed analysis of the uses of the different algorithms will be put forward to gain further insight into the inner-workings of the proposed application. Following this, some uses of machine learning and data mining in sports today will be discussed and reviewed. Finally, in the the last section, the options for choosing an open-source software suited to experimenting and implementing the information listed will be investigated.

2.1 Data Visualisation Techniques

Data visualisation is the graphical representation of data. This concept is important when dealing with any dataset. It can be difficult to get a grasp of trends in the data or understand the spread of different variables in the data when the information is listed numerically. The concept of data visualisation allows us to visually identify patterns in the data and, in turn, enables better decision making when analysing and experimenting with the data. There are a range of visualisation options which will be discussed in this section.

2.1.1 Box Plots

Box plots are a useful visualisation technique to get an idea of the spread of the data. They highlight, in particular, the skew of the distribution and indicate the presence of outliers in the data. Box plots make no assumption about the underlying statistical distribution so they are ideal for representing non-normal distributions. Some useful terms used in creating box plots are described below[1]:

- **Median:** The middle observation in data that has been arranged in ascending or descending numerical sequence.
- **Upper Quartile:** The median of the upper half of the data.
- **Lower Quartile:** The median of the lower half of the data.

The box plots can then be interpreted as follows:

- The ends of the box indicate the lower and upper quartiles. The length of the box indicates the inter-quartile range.
- The median value of the data is marked by a line within the box.
- The upper and lower tails extend to the higher and lower data observations. Values outside these tails are considered extreme values or outliers.

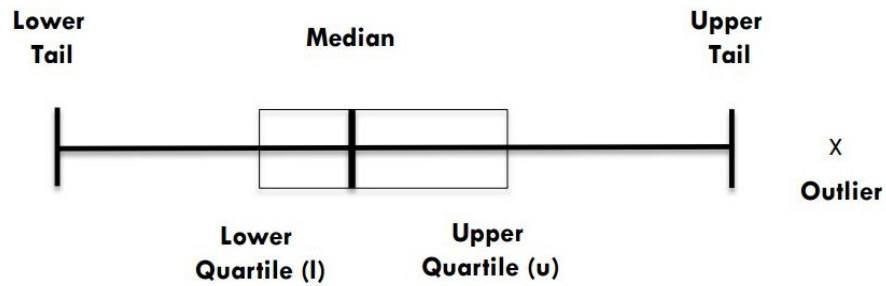


Figure 1: Example of a labelled box plot.

2.1.2 Density Plots

Another example of a data visualisation technique is the density plot. A density plot visualises the spread of the data over a continuous interval. This form of visualisation adapts kernel smoothing to the standard histogram charting format which allows for a smoother, more representative graph of the data. The density plot highlights which values of the data contains the most data points by determining and displaying the concentration of data points over a given interval.

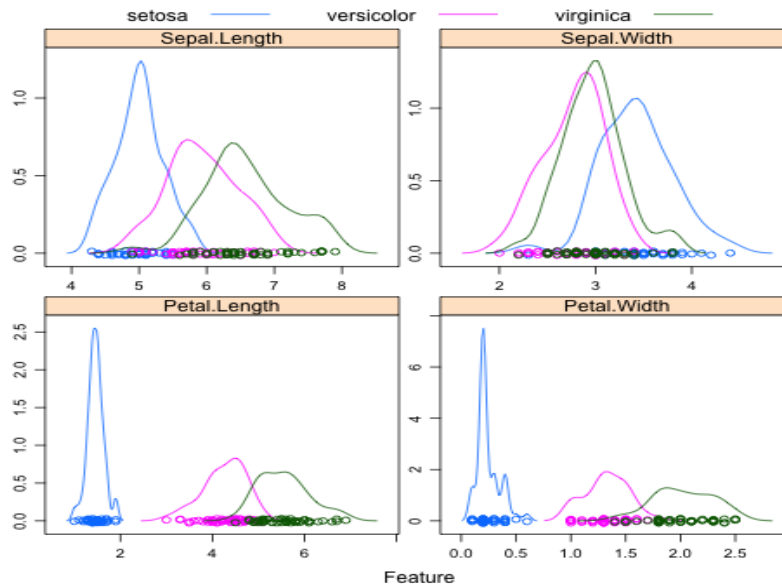


Figure 2: Example density plots of Fisher's Iris dataset.

2.1.3 Scatter Plots

Scatter plots are used in data representation to plot the data points of two variables against each other. It gives a good display of the relationship between two variables and helps understand their similarities and differences. The correlation can be made visible through the use of scatter plots where a positive upward trend between the variables indicates a positive level of association between the variables while a downward negative trend would suggest a negative association. When the points are scattered randomly across the plain it can be assumed that there is little or no association between the two variables.

Linear regression analysis can be introduced when creating scatter plots. This is a linear estimation of the relationship among the variables. The regression can be described by the equation[2]:

$$Y = a + bX$$

Where:

- Y is the dependent variable plotted on the y-axis,
- X is the independent variable plotted on the x-axis,
- a is the the y-intercept, given by

$$a = \frac{\sum y \sum x^2 - \sum x \sum xy}{n \sum x^2 - \sum x^2}$$

b is the slope of the line, given by

$$b = \frac{n \sum xy - \sum x \sum y}{n \sum x^2 - \sum x^2}$$

Figure 3 below shows an example of a scatter plot where a strong positive correlation exists between the X and Y values which is described by the line of regression.

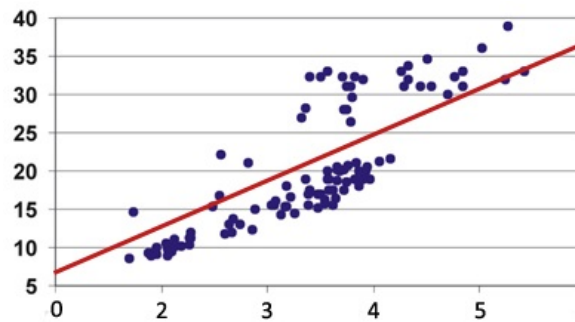


Figure 3: Example scatter plot with linear regression.

2.2 Machine Learning and Data Mining

Machine Learning is a facet of computing based on statistical techniques that deal with algorithms that are said to 'learn'. It was put forward by Arthur Samuel in 1959 that machine learning 'gives computers the ability to learn without being explicitly programmed'[3]. The idea of learning was depicted by Witten and Frank in 2005 who stated that 'learning is changing behaviour in a way that makes performance better in the future'[4].

This definition helps explain how the objective of machine learning can be seen as letting the computer improve its performance at a given task by inputting data for the computer to explore and, as a result, make predictions on. The computer learns to make predictions through the use of algorithms which model this input data and allows better decisions and predictions to be made.

Data mining describes the process of extracting interesting information from large sets of data with the intention of finding advantageous uses for the new found information. The application of this information, along with machine learning techniques, allows you to find and cultivate opportunities that may arise in new data. Data mining is especially necessary when dealing with big data, which are large and complex datasets and were traditionally unsuitable for data processing due to their size and complexity.

Machine learning tasks can be divided into supervised, semi-supervised and unsupervised learning tasks.

2.2.1 Supervised Learning

Supervised learning is a machine learning task which involves inferring a function from labelled input training data. Labelled data contains instances with attribute values as input feature variables and label values (classes) as output target variables of the model function. The computer must infer the model function based on these values. This function can then be used to make predictions on new data. Examples of the use of supervised learning include classification and regression tasks.

2.2.2 Semi-Supervised Learning

Semi-supervised learning can be seen as a class of supervised learning. It involves training a model based on a mixture of labelled and unlabelled data. The process of inferring a function to describe the training data with semi-supervised learning can often produce more accurate predictions for new data, since the additional unlabelled data allows the machine to learn the structure of new data without the presence of labelled classes.

2.2.3 Unsupervised Learning

Unsupervised learning tasks are concerned with inferring a function entirely based on unlabelled data. This means the model makes predictions on new data based on a function that

has been trained on data with input variables of which the corresponding output variables (labels) are unknown for each example. Examples of the use of unsupervised learning include clustering tasks.

2.3 Classification Analysis

As previously stated, classification is a form of supervised learning. Classification is concerned with identifying the class that a new observation belongs to, based on the function inferred from the labelled training set of data. This is achieved by feeding this input data into the specific classification algorithms and inferring the function or model. Different classification algorithms exist for a range of different tasks.

2.3.1 Decision Tree Classification

Decision trees build a classification model in the form of a tree structure. The idea of building the tree structure relies on partitioning the data at each decision node, beginning at the root node, which incrementally builds the tree levels. The leaf nodes of the tree describe the classification. A common decision tree algorithm for classification is the C4.5 algorithm. This algorithm is an extension of the ID3 algorithm originally developed by Ross Quinlan. The algorithm requires the calculation of the entropy and information gain of all attributes.

The entropy is a measure of the uncertainty of a random variable given by [5]:

$$Ent(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

For a dataset S with n different classes where the proportion of each class i is p_i .

The information gain of an attribute is the reduction in entropy from partitioning the data according to that attribute given by [5]:

$$Gain(S, A) = Ent(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Ent(S_v)$$

Where A is the attribute and S_v is the subset of S where A has value v .

Using the equations for entropy and information gain, the steps to inductively learning the decision tree are as follows [5]:

1. For all attributes that have not yet been used in the tree, calculate their entropy and information gain values for the training samples.
2. Select the attribute that has the highest information gain.
3. Make a tree node containing that attribute.
4. This node partitions the data. Apply the algorithm recursively to each partition.
5. Repeat step 4 until the entire tree is built.

Figure 4 below shows the dataset and corresponding decision tree for a classification example of deciding to play tennis (class) based on the weather (attribute).

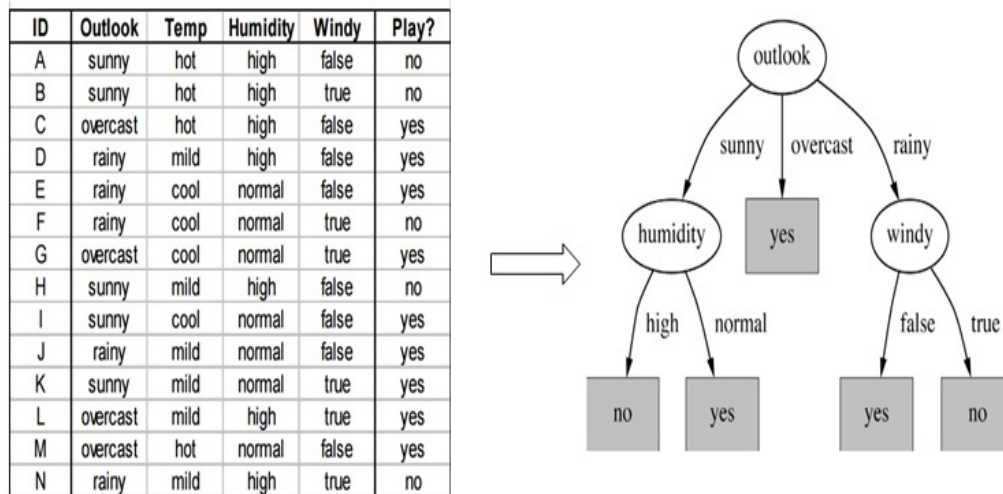


Figure 4: Dataset and corresponding decision tree.
(Taken from 1617-CT475 Machine Learning and Data Mining, *Madden* [5])

Another decision tree classification algorithm is the PART algorithm where the classification rules are constructed with partial decision trees. A random forest combines a multitude of decision trees. It operates by classifying the output class as the mode of the classes of each individual tree.

2.3.2 Linear Discriminant Analysis Classification

Linear Discriminant Analysis (LDA) is a form of linear classification is a technique used for linear classification with two or more classes. LDA relies on calculating statistical properties for each class in the data. The mean and variance of the variables for each associated class is calculated. This information is then fed into the LDA algorithm which forms a predictive model based on the calculations. The LDA assumes that the distribution of each variable is Gaussian (normally distributed) and that each attribute value has similar variance. The following equations are cited from *An Introduction to Statistical Learning with Applications in R* [6]:

The mean of each variable is given by the equation

$$\mu_k = \frac{1}{N_k} \sum x$$

Where μ_k is the mean value of each input x for the class k , N_k is the number of instances associated with the class k .

The variance is achieved by calculating the average squared difference of each mean value across every class given by:

$$\sigma^2 = \frac{1}{n-K} \sum (x - \mu^2)$$

Where σ^2 represents the variance of all inputs x , n is the total number of instances, K is the total number of classes and μ is the mean associated with the input x .

LDA makes predictions on new data based on determining probability that a new input belongs to a certain class. The classification prediction is based on the class that has the highest probability value. This class will be assigned as the output class.

Bayes' Theorem is used to estimate the probability of the output class k given the input x using the probability of each class and the probability of the data belonging to each class:

$$P(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum \pi_l f_l(x)}$$

In Bayes' Theorem π_k is called the prior probability and is expressed as

$$\pi_k = \frac{n}{n_k}$$

Where n_k is the number of classes.

Using the above equations, the discriminant function is achieved and given as

$$\delta_k(x) = x \cdot \frac{\mu_k}{2} \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Where the class that is calculated as having the largest value will be the output classification y and $\delta_k(x)$ is the discriminant function for class k given input x , the μ_k , σ^2 and π_k are all estimated from your data.

2.3.3 K-Nearest Neighbours Classification

K-Nearest Neighbours (KNN) classification is an Instance Based Learning (IBL) classification task. IBL algorithms rely on storing the training data in memory. The classification of new instances is based on similarity to these stored cases.

The KNN algorithm does not perform any generalization, meaning the algorithm keeps all the training data that has been stored. KNN classifies data by relying on the predictions of the nearest neighbours. k denotes the number of neighbours to be used as predictors. A *majority vote* is introduced where the neighbours vote on the classification of the new unlabelled instance based on a suitable distance metric. The number of neighbours is typically an odd number value when dealing with two classes while $k = 3$ is considered a minimum value for the majority of KNN applications. An example of KNN with $k = 3$ and $k = 6$ is shown below.

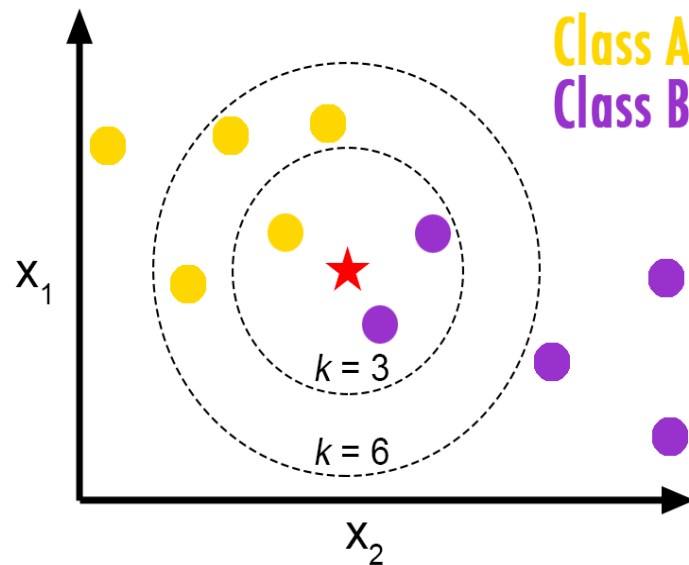


Figure 5: KNN classification with $k = 3$ and $k = 6$.

When dealing with KNN it is important to choose an appropriate distance metric. The equations of different distance metrics are described below when $p = (p_1, p_2, \dots, p_n)$ and $q = (q_1, q_2, \dots, q_n)$ [5]:

- Euclidean Distance: $\sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2}$
- Manhattan Distance: $|q_1 - p_1| + |q_2 - p_2| + \dots + |q_n - p_n|$
- Cosine Distance: $\cos\theta = \frac{\bar{p} \cdot \bar{q}}{|\bar{p}| \cdot |\bar{q}|}$
- Hamming Distance: Assign 0 for each attribute where both cases have the same value, assign 1 for each attribute where both cases are different.

2.3.4 Support Vector Machine Classification

Support Vector Machine (SVM) is a classification algorithm which operates by plotting the data points in p -dimensional space, where k is the number of instance features, with each instance value being expressed as individual co-ordinates. A hyper-plane is introduced to determine the difference in class of the instance features. The hyper-plane is linear and described, in p -dimensional space, by the function

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

The data points plotted in this p -dimensional space are known as support vectors. Since there is an infinite amount of different hyperplanes that could be introduced, the optimal hyperplane is chosen by the SVM algorithm by choosing the hyperplane that creates the biggest margin. The margin is given as: twice the distance between the hyperplane and the closest support vector to the hyperplane. Figure 6 on the next page illustrates a simple SVM classification with labelled hyperplane and margin.

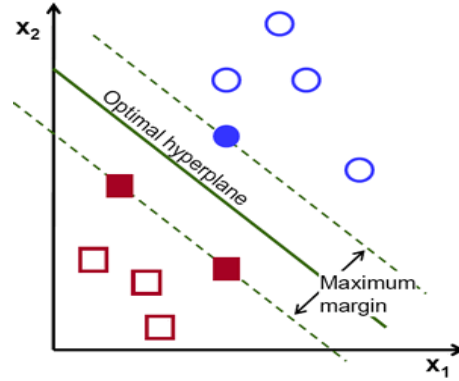


Figure 6: SVM Classification with optimal hyperplane and maximum margin.

2.3.5 Naive Bayes Classification

Naive Bayes is a probabilistic classification technique based on Bayes' Theorem. The algorithm assumes that all predictors are independent of each other. This means that, when predicting the class of a new instance, all predictor features are independent when finding the probability that the new instance belongs to a given class. Using Bayes' Theorem, the probability that an instance belongs to a class given the predictors is described by the equation

$$P(C|x) = \frac{P(x|C)P(C)}{P(x)}$$

Where $P(C|x)$ is the posterior probability of class c given predictor x , $P(x|C)$ is the posterior probability of predictor x given class c , $P(C)$ is the probability of class c and $P(x)$ is the probability of predictor x .

2.4 Cluster Analysis

Clustering or cluster analysis involves organising data objects into clusters with each cluster containing members more similar to each other than to members of other clusters. Clustering is an example of unsupervised learning and is used to uncover collections in the unlabelled data, which is useful for determining trends in the underlying structure of the data. Hence, it is possible to make decisions and draw conclusions from the clustered data which is the main advantage of cluster implementation. It can be difficult however, to determine the quality of the clusters since the data, in this case, is unlabelled. Figure 7 on the next page shows an example of clustering with 3 clusters.

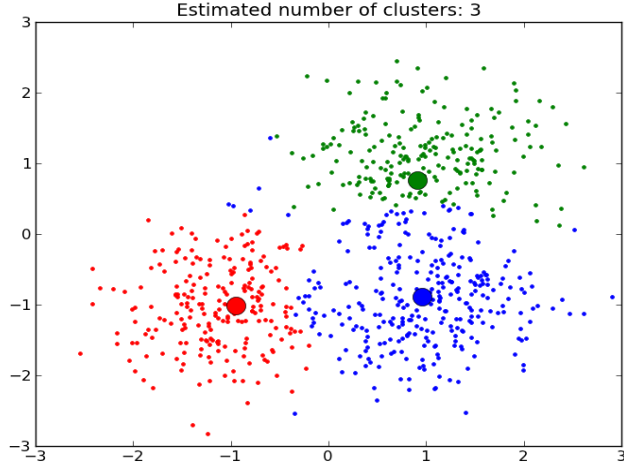


Figure 7: Clustering with 3 clusters.

2.4.1 K-Means Clustering

K-means Clustering is one of the most popular clustering algorithms in machine learning. The algorithm clusters the input data based on a number of input parameters including the number of clusters, K .

A useful explanation of the K-means algorithm cited from a paper by AK. Jain [7] on the topic of K Means clustering and is described below.

Let $X = \{x_i\}$, $i = 1, \dots, n$ be the set of n d -dimensional points to be clustered into a set of K clusters, $C = \{c_k, k = 1, \dots, k\}$. The K-means algorithm finds a partition such that the squared error between the empirical mean of a cluster and the points in the cluster is minimized. Let u_k be the mean of cluster c_k . The squared error between u_k and the points in cluster c_k is defined as

$$J(c_k) = \sum_{x_i \in c_k} \|x_i - u_k\|^2$$

The aim of the K-means clustering is to minimize the sum of the squared error over all K clusters,

$$J(C) = \sum_{k=1}^K \sum_{x_i \in c_k} \|x_i - u_k\|^2$$

Where $\|x_i - u_k\|^2$ is the Euclidean Distance between x_i and u_k .

Once the value of K has been chosen, the K-means clustering process is as follows:

1. Initialize the center of the clusters.
2. Attribute the closest cluster to each data point.
3. Set the position of each cluster to the mean of all data points belonging to that cluster.
4. Repeat steps 2-3 until convergence.

2.5 Applications of Statistical Analysis in Sports Science

The competitive nature of sport means it is necessary to generate solutions for improving athlete performance. Data mining and visualisation processes can be applied to sports players' datasets to determine trends in their performance or identify indicators which are tied to injury, and ability to perform. When data mining and visualisation techniques are applied correctly to such datasets, the information drawn from the process can prove vital for making decisions for how the players can improve performance and overall wellness. Machine learning processes can also be applied to the players' datasets to discover variations in the data and to construct models that adapt to accommodate new data. By merging the machine learning and data mining techniques one can attempt to predict and improve future performance of the players.

Orecco, a company based in Galway, Ireland, strive to optimize athletic performance by combining sports science with data science, as described above, in order to generate indicators of injury, optimal training load and recovery strategies. The analysis of biomarker data is a leading division of sport science which can be applied, in this case, to improve performance and reduce injury. Orecco focus on biomarker analysis in their work and this inspired me to research the application of this biomarker analysis in conjunction with statistical analysis techniques.

2.5.1 Biomarker Analysis

Biomarker data is gathered by taking blood samples from athletes. "A *biomarker, or biological marker, generally refers to a measurable indicator of some biological state or condition*"[8]. Interesting observations and analysis can be drawn from the data of an athlete such as key nutritional improvements or signs of injury. These blood samples undergo a series of tests to check the presence or absence of certain biomarkers or to compare the biomarkers found in the sample against control biomarkers. After carrying out such tests, the overall biomarker data results are compiled for each athlete.

Examples of common athletic biomarkers are listed below [9]:

- **Vitamin D:**

What it is: A substance that helps boost bone mass and immunity.

Why it matters: Low vitamin D, along with deficient calcium, can lead to stress fractures, breaks, and osteoporosis.

- **Creatine Kinase:**

What it is: An enzyme produced during exercise.

Why it matters: High levels can indicate over-training, leading to fatigue, muscle damage, and risk of injury.

- **Testosterone:**

What it is: A substance that helps boost bone mass and immunity.

Why it matters: A hormone that plays a key role in muscle mass, bone density, cholesterol levels, and VO₂ max.

- **C-Reactive Protein:**

What it is: A protein that correlates with muscle inflammation.

Why it matters: High levels can indicate heart disease or over-training, compromising the immune system and impeding recovery.

- **Vitamin B12:**

What it is: A substance that helps synthesize new cells, repair damaged cells, and deliver oxygen to muscles.

Why it matters: Low Vitamin B12 can decrease endurance and impede exercise intensity.

- **Ferritin:**

What it is: A protein that indicates blood iron levels.

Why it matters: Low iron levels can lead to injury, elevated heart rate, and weakened immunity. High levels can increase inflammation and cholesterol and decrease cardiovascular health.

2.5.2 Case Study I

Some interesting insights into biomarker analysis were put forward by Nathan A. Lewis et al. in an article titled *Critical Difference and Biological Variation in Biomarkers of Oxidative Stress and Nutritional Status in Athletes* in 2016 [10]. This study focuses on the knowledge of biological and critical difference values and how they can be used in data interpretation of individual athletes in order to monitor oxidative stress levels which indicate levels of fatigue and

over-training in the athletes. It was found that 'the application of oxidative stress biomarker monitoring programs in sport are hindered by reliability and repeatability of in-the-field testing tools, the turnaround of results, and the understanding of biological variation'[10]. This article outlines the introduction of the monitoring of oxidative stress levels in athletes coupled with knowledge of biological variations to prevent fatigue and injury in athletes. The work completed in this study provides interesting results in the analysis and visualisation of biomarker data.

The blood samples of twelve well-trained athletes was taken and analysed for specific biomarkers. Biomarkers such as serum alpha-tocopherol, gamma-tocopherol and serum lutein were extracted from the samples and biochemically analysed. This was followed by statistical analysis of the resulting data which consisted of numerical and graphical summaries of each biomarker. A linear mixed model was used to model the change in the biomarker readings over time. 'The time when the testing was recorded was modeled as a fixed effect, initially as a categorical variable in order to allow a comparison in the mean change at each time point and then as a continuous variable in order to compare the slopes for each biomarker over time.'[10]. The change in biomarker readings over time are described in figure 8 below.

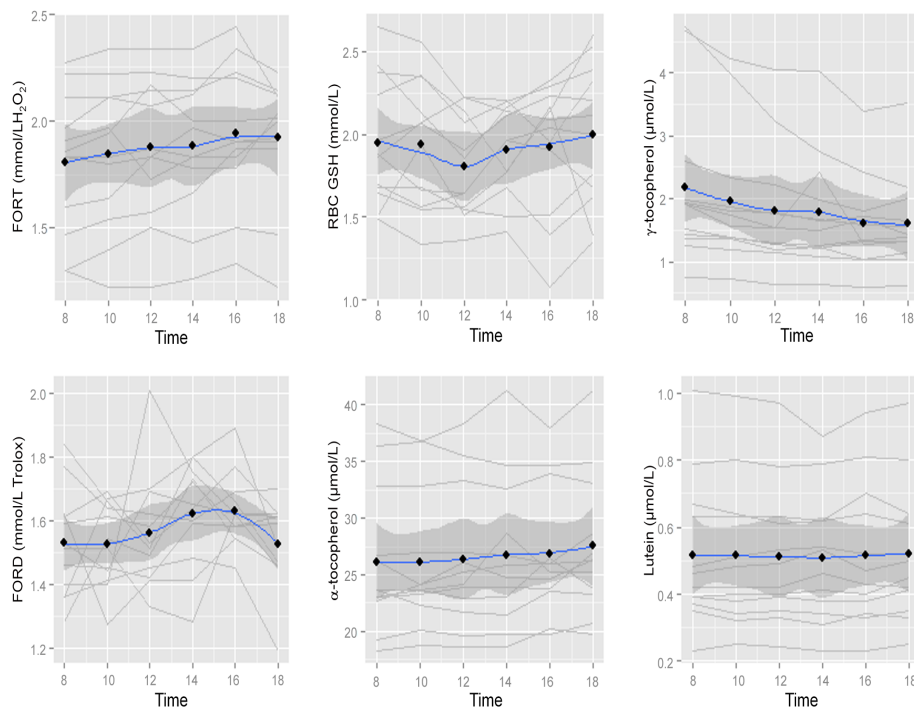


Figure 8: Plot of each biomarker over time with mean (black dot), smoothed trajectory (blue line) and 95% confidence interval displayed (dark shaded area). [10]

The constant monitoring of these biomarker results allow informed decisions to be made regarding the health and fitness of the athletes by uncovering signs of fatigue, over-performance or bad nutrition.

2.5.3 Case Study II

The application of machine learning techniques in the field of genetics and bioinformatics was investigated by Renaud Gaujoux and Cathal Seoighe in an article titled *A flexible R package for nonnegative matrix factorization* in 2010 [11]. This article explores the application of nonnegative matrix factorization (NMF), an unsupervised learning technique, in R to 'extract meaningful information from high-dimensional data such as gene expression microarrays'[11]. The development of an R package for NMF along with its potential implementations in bioinformatics studies is put forward in this article and 'commonly used benchmark data and visualization methods are provided to help in the comparison and interpretation of the results'[11].

The basis of NMF involves the 'factorization of matrices representing complex multidimensional datasets'[11] with applications in pattern recognition and unsupervised clustering. NMF has been implemented in bioinformatics and genetics research in areas such as the discovery of cancer types based on gene expression microarrays and has potential application in the area of biomarker analysis in sports science. NMF is particularly suited to the field of genetics since it allows for the overlap of input data components which is useful in 'the context of gene expression microarrays, where overlapping metagenes could identify genes that belong to multiple pathways or processes'[11]. This article primarily outlines the development and implementation of the R package, however, some interesting results were generated and visualised using the package with the Golub dataset which was referenced by Brunet et al. in *Metagenes and molecular pattern discovery using matrix factorization* [12]. The number of metagenes used to approximate the target matrix is determined by the specified parameter called the factorization rank, r . The optimum value for r is determined by the cophenetic correlation coefficient which indicates the robustness of the clusters created. Figure 9 below shows the output of the cophenetic correlation coefficient for ranging r using the NMF package.

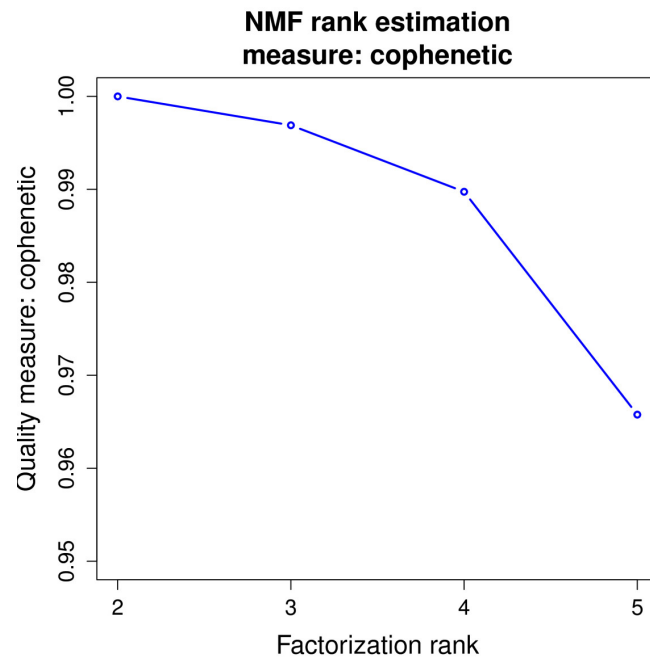


Figure 9: Cophenetic correlation coefficient. [11]

The metagene matrix can then be obtained from the factorization and visualised in the heatmap shown in figure 10 on the next page.

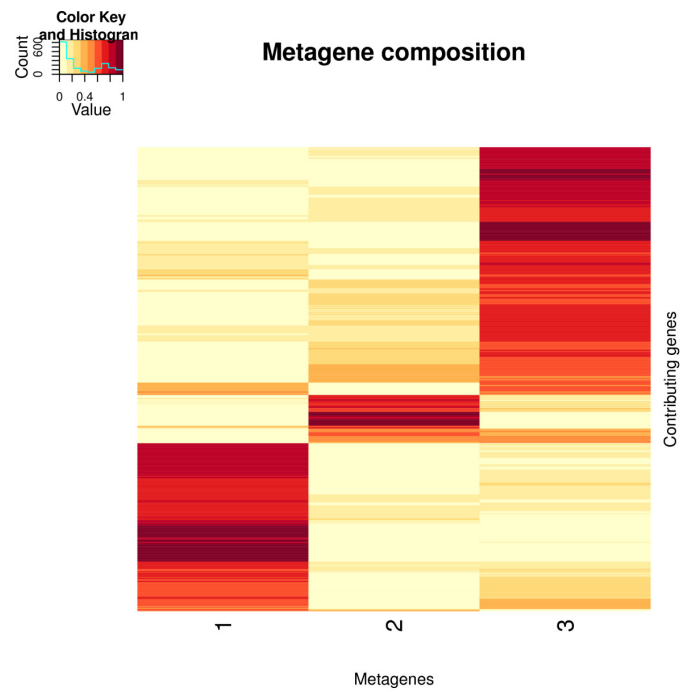


Figure 10: Metagene matrix. [11]

Valuable observations about the genetics expressions can be drawn from the resulting visualisations above, which can be used in the understanding and predictive analysis of genetic and biomarker datasets.

2.6 Machine Learning Software Currently Available

There is an abundance of software packages available for carrying out visualisation, classification and clustering tasks. Some examples of free, open source packages including Weka and Orange which provide powerful machine learning and data mining capabilities and are very useful when investigating important machine learning algorithms and processes. Programming languages such as R and Python are used for development of statistical analytics and machine learning software. Some examples of the software implemented in this project are outlined in this section. The application development in this project was completed in RStudio, an Integrated Development Environment (IDE) for the R programming language.

2.6.1 Orange

Orange is a free, open source toolkit, used for data visualisation, machine learning and data mining. Orange is a component-based visual programming environment used for exploratory data analysis. Orange relies on graphic visualisation to help users get a better understanding of the data and uncover hidden trends in the data. Users can interact with the various scatter plots, distributions and tree diagrams providing a hands-on exploration of the data in a user-friendly graphical interface. The Orange components are called widgets and the default widgets data, visualise, classify, regression, evaluate and unsupervised are described below [13]:

- **Data:** Data input, data filtering, sampling, imputation, feature manipulation and feature selection.
- **Visualise:** Common visualisations (box plot, histograms, scatter plot) and multivariate visualisations (mosaic display, sieve diagram).
- **Classify:** A set of supervised machine learning algorithms for classification
- **Regression:** A set of supervised machine learning algorithms for regression.
- **Evaluate:** Cross-validation, sampling-based procedures, reliability estimation and scoring of prediction methods.
- **Unsupervised:** Unsupervised learning algorithms for clustering (K-means, hierarchical clustering) and data projection techniques (multidimensional scaling, principal component analysis, correspondence analysis)..

The graphical user interface (UI) of Orange is displayed below with examples of box plots and density plots.

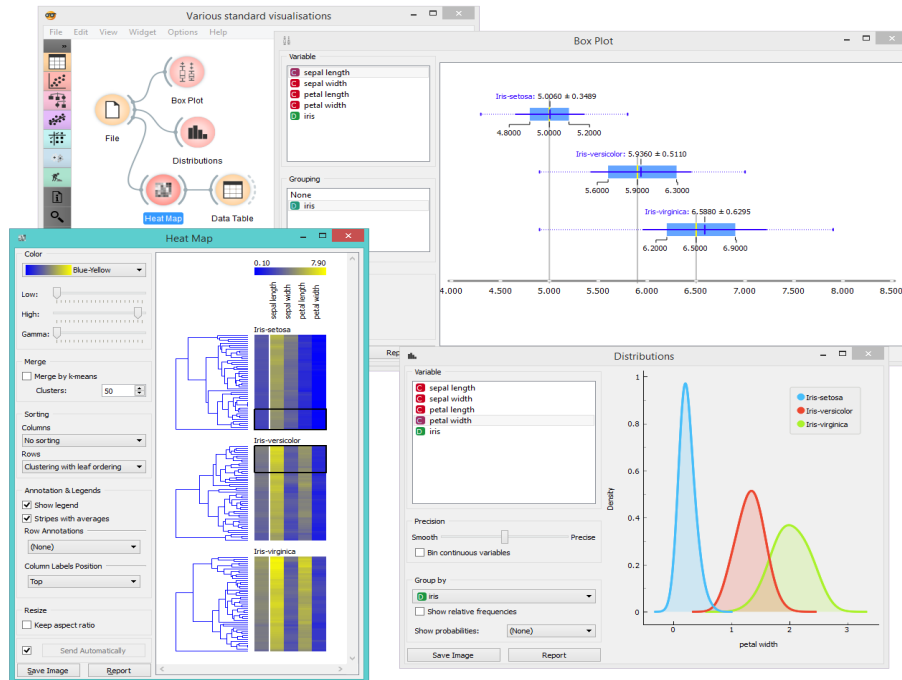


Figure 11: Orange UI with detailed visualisations.

2.6.2 Weka

Waikato Environment for Knowledge Analysis (Weka) is a free, open-source software which was developed in the Java programming language at the University of Waikato, New Zealand. Weka is a popular software tool used for computing a range of machine learning and data mining tasks. As shown in the image below, the software interface is divided into varied sections which serve different purposes.

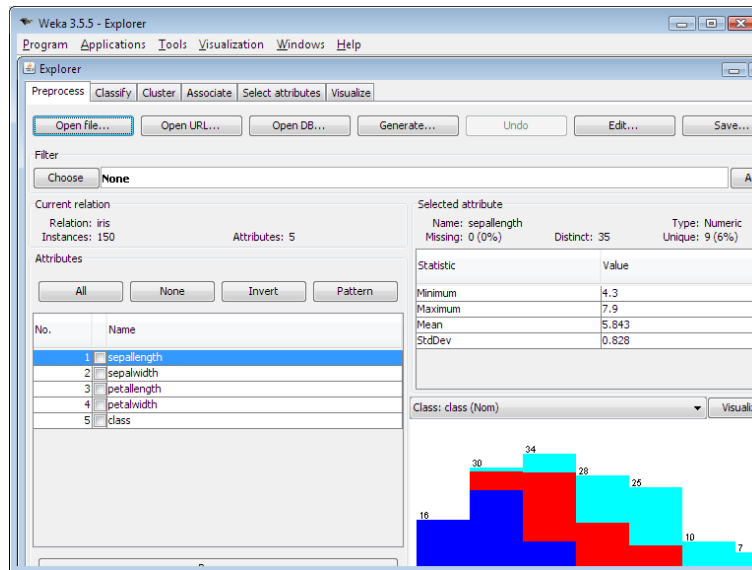


Figure 12: Weka UI.

The main features include:

- **Data Pre-Processing:** Load datasets, chart data, browse statistical information on the data.
- **Classification:** Apply and test various classification algorithms on the dataset and produce visualisations such as decision trees.
- **Clustering:** Apply and test clustering algorithms on the dataset and produce visualisations such as hierarchical cluster trees.
- **Visualisation:** Visualise the data in a scatter plot matrix and generate individual bi-variate scatter plots.
- **Association:** Data mining the dataset for various association rules.

Weka is an excellent tool for testing and analysing the results of various machine learning and data mining algorithms and was the main platform used for the comparative testing of the results for the application.

2.6.3 Python and Extended Libraries

Python is an open-source object-oriented, high-level programming language. Python is a general-purpose programming language with a wide variety of applications such as web and Internet development, network programming and scientific and numeric analysis. Python's built-in libraries for mathematical programming and data visualisation include NumPy, SciPy, and matplotlib. Scikit-learn is a library built for machine learning and data mining programming in Python based on the libraries previously mentioned. The key facilities offered in the scikit-learn package are classification, clustering, regression, dimensionality reduction, model selection and data pre-processing.

2.6.4 R Programming

R is a free, open-source programming language used for statistical computation and graphics. It is a popular language among statisticians and data miners as it provides extensive statistical capabilities to users such as linear and non-linear modelling, statistical testing, time-series analysis and machine learning capabilities such as classification and clustering. Users can render detailed graphics with enhanced colouring which are displayed in the user-friendly UI.

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes [14]:

- An effective data handling and storage facility.
- A suite of operators for calculations on arrays, in particular matrices.
- A large, coherent, integrated collection of intermediate tools for data analysis.
- Graphical facilities for data analysis and display either on-screen or on hardcopy.
- A well-developed, simple and effective programming language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

RStudio is the preferred IDE for R software development. From RStudio, users can develop R projects, and easily manage multiple working directories using these projects. RStudio also provides added features to help in the development of R code such as smart indentation, code completion, and syntax highlighting. The RStudio console contains windows to display the code, the command, environment variables and detailed plots as shown in figure 13 below.

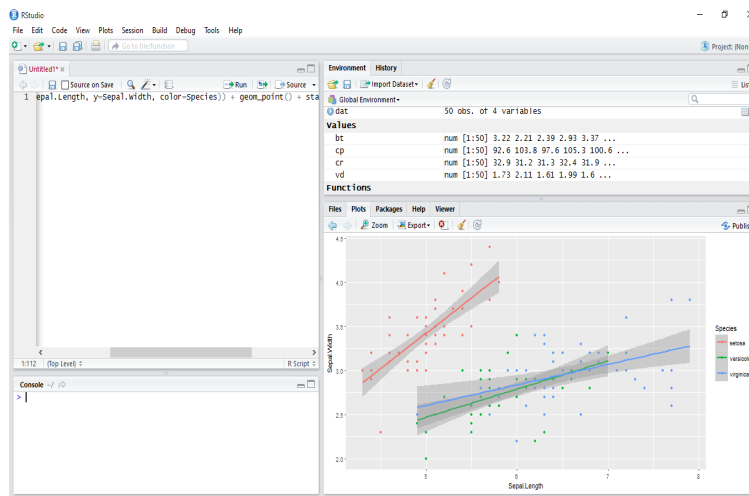


Figure 13: RStudio console UI.

R Shiny is a framework that is integrated with RStudio which allows users to build powerful R driven web applications for statistical analysis and graphics. This means that Shiny users can develop interactive web pages that harness the capabilities of R without knowing any peripheral web development languages such as HTML, CSS, or JavaScript. This opens the opportunity to deploy a host of machine learning and data mining potential on an interactive and easy to use web application. An example of an R Shiny application front end is shown in figure 14 on the next page.

Height and weight data

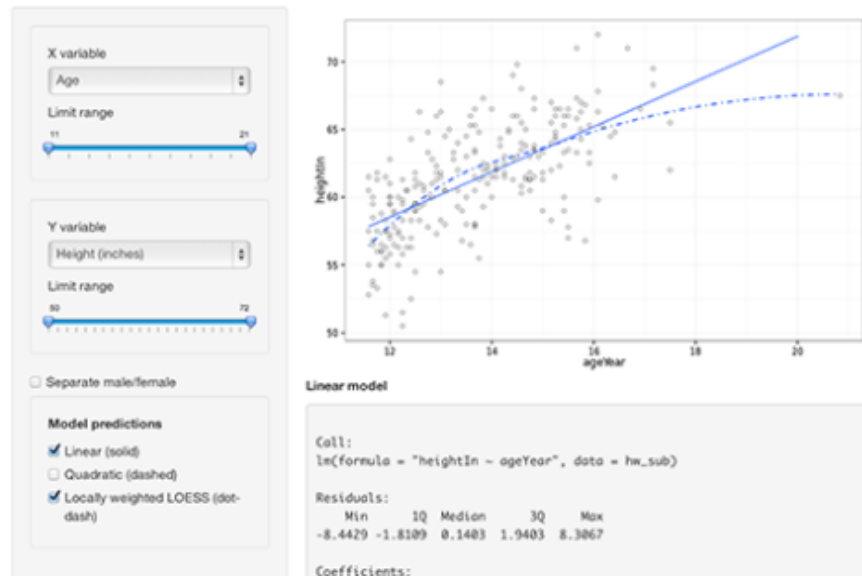


Figure 14: Front end of an example R Shiny web application.

2.7 Project Influence

The inspiration for this project stems from the influence of the methodologies, techniques and technologies outlined in the above section. The aim of this project is to develop an application that merges the detailed visualisations of Orange and the machine learning and data mining power of Weka through the implementation of R programming. The application will be capable of producing graphical representations of data with box plots, density plots and scatter plots and will also perform classification and cluster analysis on the data. The application will be geared towards the analysis of biomarker data for sports science by implementing the features described above. However, users will not be restricted to upload and analyse biomarker data only.

3 Developing the Application

This section of the thesis deals with developing the application for machine learning and data visualisation.

Firstly, an introduction of the application is outlined to give a brief overview followed by a description of the application requirements. The application design phases detail the initial design and improvements made on the design. A detailed summary of the application components is outlined along with testing, verification and comparative results.

3.1 Introducing the Application

The application is built for machine learning and data visualisation. The goal of the application is to allow users to input sample datasets to perform exploratory data analysis and render graphical representations of the data. Rich visualisations help users gain insight into complex datasets and uncover trends and patterns in their data. An appealing graphical user interface is introduced to provide ease of use and to optimise the user experience. The user can classify and cluster their data based on algorithms of their choice by training machine learning models. Experimentation results can be explored and analysed to determine which algorithms best classify and cluster the data and with this knowledge the user can make an informed decision when choosing a trained model to operate on new data. The application implements a range of machine learning algorithms including the PART decision tree, Random Forest, Linear Discriminant Analysis, Support Vector Machine, K-Nearest Neighbour and Naive Bayes for classification and K-means and hierarchical clustering for cluster analysis. These algorithms were implemented in R by downloading and installing the R package provided for each algorithm. These packages are available in the Comprehensive R Archive Network (CRAN).

3.2 Application Requirements

This section describes the requirements of the application which are categorized as the graphical user interface, reading and describing the dataset, presenting and visualising the data, applying machine learning algorithms and analysing of model performance and accuracy.

3.2.1 Graphical User Interface (GUI)

The application will rely on an appealing graphical user interface. This ensures the user can operate the application easily and will optimize the overall user experience. A well-designed GUI means that users will be capable of performing complex data analysis and machine learning tasks in a user-friendly environment. This will simplify daunting analytic and mathematical operations and will provide for visual and exploratory learning.

3.2.2 Reading and Describing the Dataset

The opening page of the application will present a data file upload window which will allow the user to upload a file of their choice. The application will deal mainly with CSV (comma-separated values) file input but the possibility to take in other data file types such as ARFF (Attribute-Relation File Format) files in future will be considered. Once the data has been uploaded some useful descriptive tables will be produced. These tables will summarise the dataset by presenting the number of observations and variables in the set and will detail the observations per class. The entire dataset will be printed in an elegant table.

3.2.3 Presentation and Visualisation

The user will be able to graph the data with aesthetic plotting options. The data will be visualised using box plots, density plots and scatter plots. These options will provide users with easily interpretable representations of the spread of the data and will allow the user to plot individual variables against other variables. The option of plotting the data based on class will also be presented and a percentage of missing data points will be described by a histogram.

3.2.4 Machine Learning Algorithms

The application will introduce machine learning algorithms for classification and clustering of input data.

Classification: The user will be able to choose from a range of classification algorithms to classify the data. A default option is chosen when the user navigates to the classification page. A drop-down menu will provide the option to change algorithms. The user can choose the percentage split for training and testing the data to model and test the performance of the data. Added parameters will also be chosen by the user, such as the number of neighbours for a K-nearest neighbours task. An added data upload option will be available so the user can upload new data to classify based on the trained model.

Cluster: The user will be able to perform a K-means clustering of the data. The user will specify the parameters, including the selection of variables to plot and the number of clusters. An elegant visual display of the assigned clusters will be plotted for the selected variables. The added feature of generating a hierarchical cluster tree will be made available to the user.

3.2.5 Analysis of Model Performance and Accuracy

The results of the various classification and clustering models will be calculated and presented in real-time. This allows for prompt experimentation since users can quickly summarise the output results and select optimum algorithms.

3.3 Application Design

3.3.1 Phase I

The initial design phase involves creating an application for data visualisation for sports player data only. The objective of the application is to allow users to upload sports data to the application and visualise their sports data using techniques such as box plots, density plots and scatter plots. The user can then use these visualisation techniques to understand and interpret large amounts of sports data and give them an idea of trends and patterns in the data. Some other data analysis features to be implemented in the application include a table of the data and the summary statistics of each variable in the dataset. The summary statistics include the minimum, maximum, mean and median of each variable.

It was decided early on that the application would be programmed using R. R would provide a good platform for both the statistical analysis and the graphics that were required for the application. The R code is written and tested in RStudio. This code produces the desired results, however, these R analytics had to be accessed by a user from a web application. After some research into deploying R scripts in a web application, a web framework called DeployR was discovered. DeployR is a web based technology that allows for integration between R analytics and web applications. DeployR allows for R files to be executed by web applications by turning R scripts into analytics web services once they have been uploaded to DeployR server. Employing this system requires knowledge of web development languages such as HTML, CSS and JavaScript and also requires the added monitoring, uploading and testing of the R files within the DeployR framework.

Design Drawbacks: Based on the initial design, the application can only take in sports datasets with specific variable names (headers). This restricts data input to sports data of a certain type only. The application features are limited, only performing simple visualisation tasks on the data. The use of DeployR meant programming in a range of different languages and had the added burden of monitoring, uploading and testing of the data with the DeployR framework. Also, DeployR relies on a web services API (Application Programming Interface) to deliver the R analytics. This meant that the constant API request and response calls made between the front-end web page and back-end DeployR server added extra computation which creates a substantial delay time when generating the output.

3.3.2 Phase II

The objective of the second design phase is to tackle the issues and drawbacks discovered in the initial phase. The option of taking any dataset as an input is introduced into the application. This means the application is no longer restricted to sports data only and greatly expands the scope of the application. The user can then upload a dataset of their choice and perform some statistical analysis and visual graphing of the data. It was decided that added features would be introduced into the application such as machine learning facilities. The addition of

classification and clustering tasks improves the application greatly and gives the user the option to explore different machine learning algorithms which, again, broadens the capability of the application and provides users with greater insight into their data. After carrying out extensive performance testing using Google's developer tools, Chrome DevTools, it was discovered that the web application put forward in phase 1 was greatly under-performing regarding run speed. The request and response times for a simple task using the DeployR driven web application are shown in figure 15.

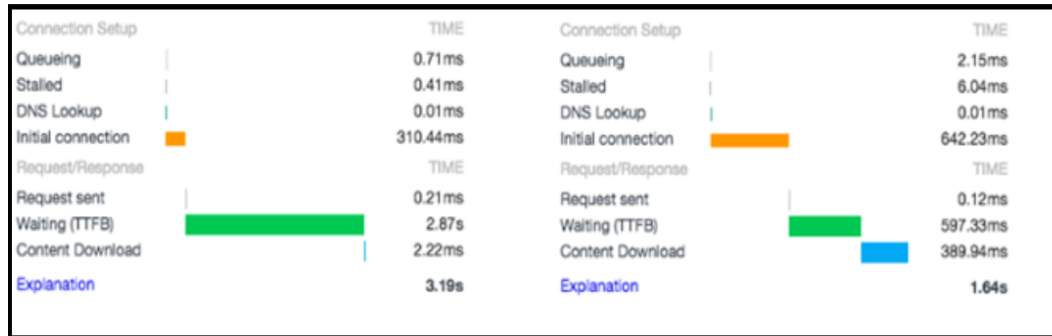


Figure 15: Performance times using the DeployR driven application.

It can be seen that the total request time is 3.19 seconds and the total response time is 1.64 seconds, taking a total of 4.83 seconds to generate the output. Following this discovery, the decision was made to begin developing the web application in R Shiny (explained in Section 2.6.4).

After testing the Shiny R application for the same task using Chrome DevTools, the improvements in performance were clear, taking only 244ms to generate the same output.

The final design flow diagram is shown on the next page.

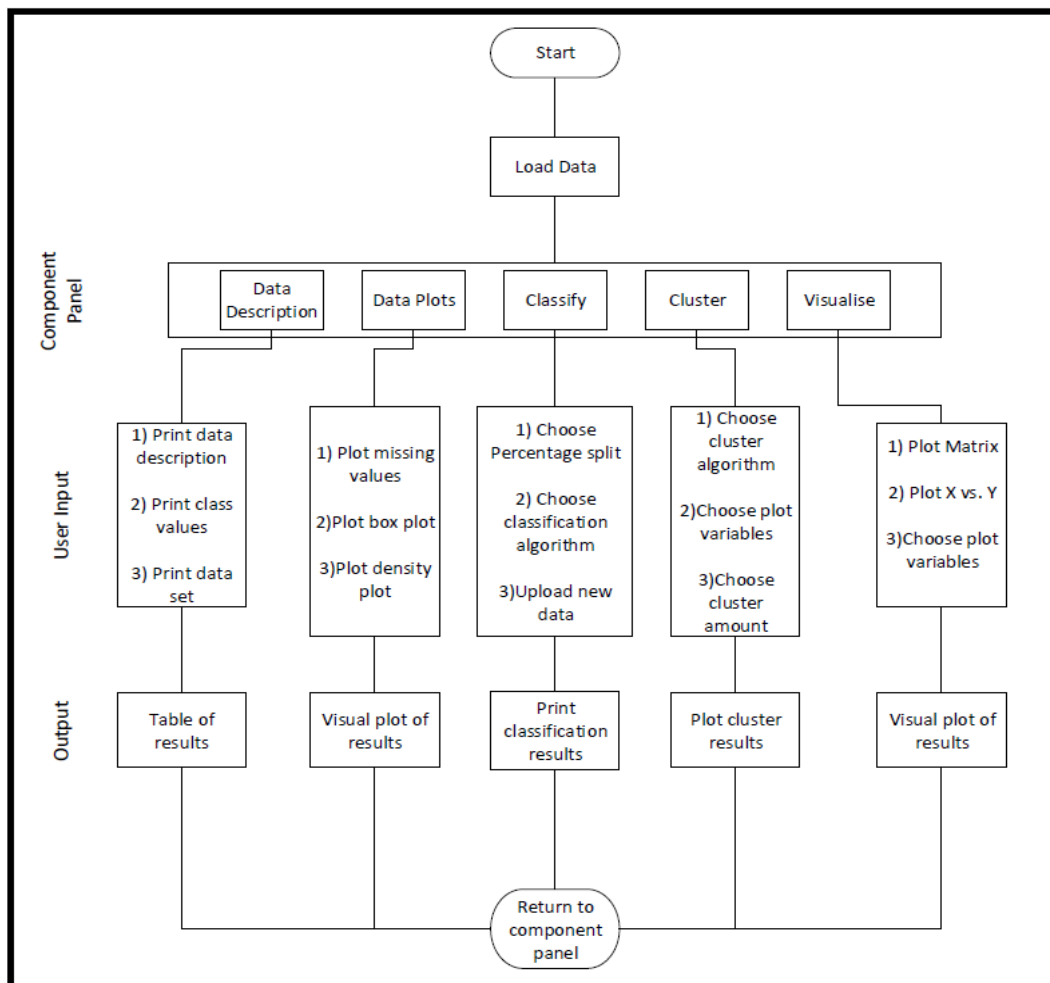


Figure 16: Design flow of the application.

3.4 Application Components

The application performs a range of different operations which are divided into individual component panel tabs in the application GUI. The different components are described in detail in this section.

3.4.1 Data Upload and Data Description

The user can upload a CSV file to the application. The upload window lets users browse their computer for a CSV file of their choice and a progress bar visualises the progression of the upload. Once the data has been uploaded, the user can choose a range of options from a drop-down menu. The options are:

- **Data Description:** Table showing the number of variables, observations and classes in the dataset.
- **Observations by Class:** Table showing the number of observations per class and the class names.
- **Table of Data:** Table showing the entire dataset.

The GUI display for this component is shown in figure 17 below

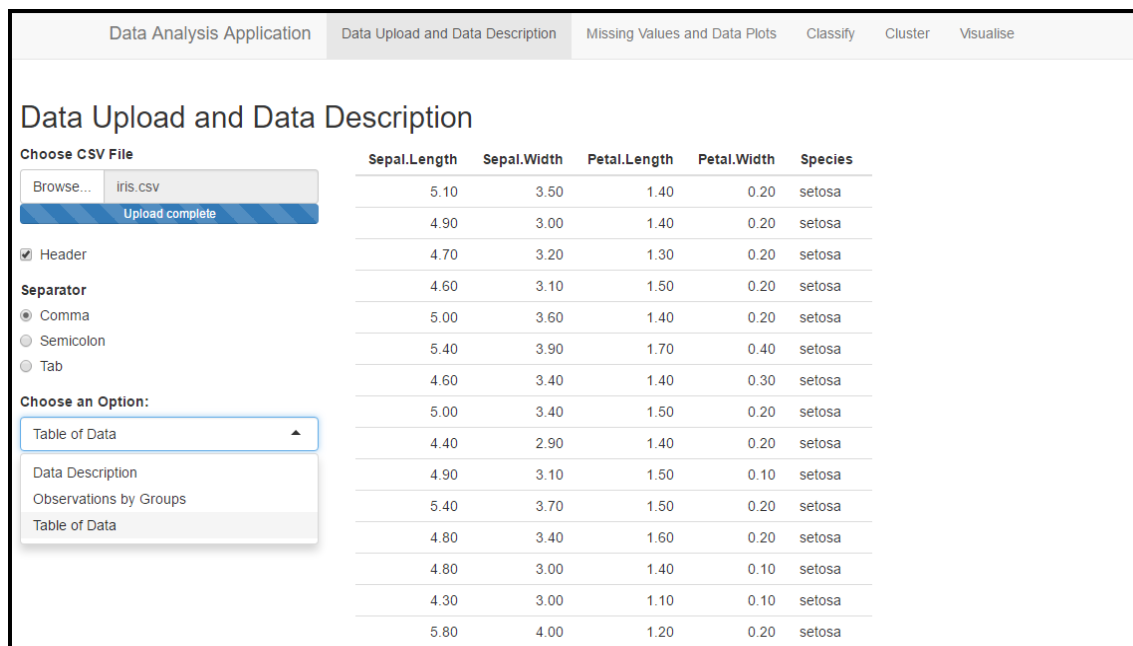


Figure 17: GUI of Data Upload and Data Description panel.

3.4.2 Missing Values and Data Plots

This component allows the user to perform a range of visualisations of the data. If missing values exist in the dataset the application will register the number of values missing and generate a histogram to display the percentage of missing values in the set. The option to plot the percentage of missing values per individual variable is also available to the user. The user can generate slick visualisations of the individual variables in the data using box plots and density plots which uncover the spread of the data in an elegant graphical display. Again, these options are made available to the user via a drop-down list. When generating the box plots and density plots users can choose to plot the data by class which gives insight into the underlying structure of the data, separated by class. The GUI display for this component is shown in the images below which highlight the box plot and density plot display.



Figure 18: GUI of Data Plots Panel with Box Plot Output.



Figure 19: GUI of Data Plots Panel with Density Plot Output.

3.4.3 Classify

The classify component is used to classify the input dataset. The user is prompted to select a percentage split using a slider widget. The percentage value chosen by the user determines the split of the data into training and testing data. For example, if the user selects a 0.7 split, this will allocate 70% of the data for training the classification model and the remaining 30% is used to test the model. A drop-down menu gives users access to a range of classification algorithms. The K-nearest neighbour algorithm requires an added parameter so users can choose the amount of neighbours to implement in the KNN classification. The selection of different algorithms produces varied experimentation results in the output panel to be compared and analysed. Summary statistics are printed with statistical values on the dataset including the minimum, maximum, mean, median and the first and third quartiles of each feature variable. The experimentation results include a confusion matrix and model accuracy, precision, f-measure and recall. After the model has been trained and tested the user is prompted to upload new data to classify. This new data can be unlabelled (no target output values) but must be compatible with the original dataset. This means that it must have the same feature variable names as the original dataset. The new data is classified using the model that has been trained.

The GUI display for this component is shown in figure 20 on the next page using the K-nearest neighbour algorithm as an example.

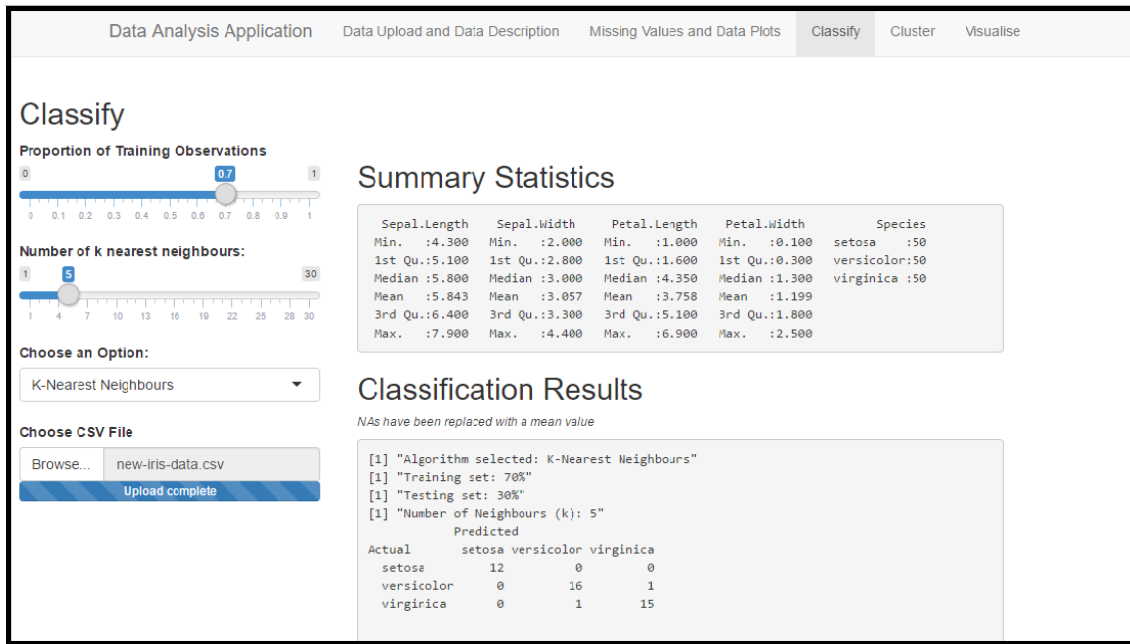


Figure 20: GUI of Classify Panel.

3.4.4 Cluster

The cluster component allows users to plot the results of K-means and hierarchical clustering of the input dataset. Two drop-down menus are dynamically populated with the feature variable names. The user chooses which variable to plot on the X and Y axes. The K-means algorithm implemented in this component requires the user to choose the number of clusters to pass as a parameter into the algorithm. A scatter plot of the selected feature variables with the coloured cluster assignments is generated in the output panel of this component. Since clustering is an unsupervised learning task, it is important to note that if the input data is labelled, the application separates the features from the target classes so the algorithm can perform clustering on the feature variables only. The user can choose between K-means and hierarchical clustering via a drop-down menu. The selection of the hierarchical clustering option will generate a hierarchical cluster tree, known as a dendrogram, which is a tree structure that builds a hierarchy of clusters.

The GUI display for this component is shown in figures 21 and 22 on the next pages using the K-means and hierarchical clustering examples.



Figure 21: GUI of Cluster panel with K-means cluster plot.

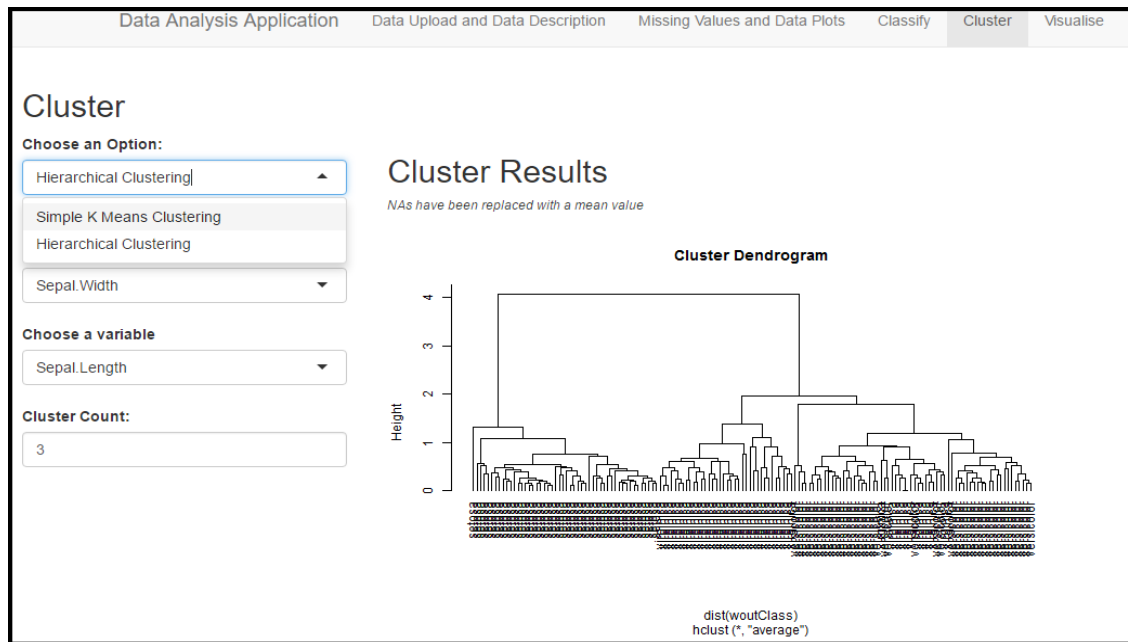


Figure 22: GUI of Cluster panel with hierarchical cluster plot.

3.4.5 Visualise

Finally, the visualise component provides added data visualisation facilities for the user. The user can choose to generate a scatter plot matrix populated by the scatter plots of each feature variable. The corresponding correlation coefficient and p-values are also generated for each bi-variate scatter plot. The user can also choose to plot the feature variables in a separate window by choosing which variable to plot on the X and Y axes from the drop-down menus which are dynamically populated with the feature variable names.

The GUI display for this component is shown in figure 23 below.

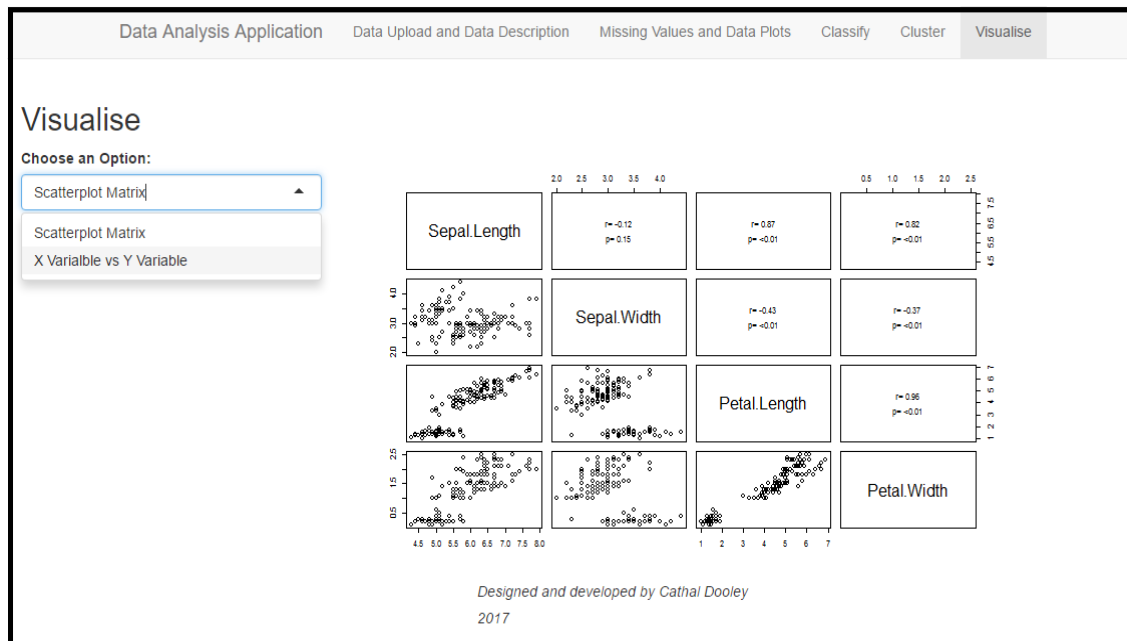


Figure 23: GUI of Visualise panel.

3.5 Testing, Verification and Comparative Analysis

This section outlines the process for testing and verifying the operation and results of the application. Existing applications such as Weka and Orange were used as analysis tools to compare and verify the visualisations and machine learning task outputs. The dataset used for the most of the testing of the application was R.A. Fisher's Iris dataset, 1936.

3.5.1 Read CSV File and Separate Data

This application requires a certain manipulation of the input data in order to carry out the data analysis and visualisation tasks. The data must be separated to produce information such as the number of variables and observations, the number of classes and class names and a list of the feature variable names for tasks such as populating drop-down lists and performing separate analysis tasks on the features and targets. To test the ability of the application to manipulate the data, smaller datasets were initially uploaded. This made it possible to compare the results of the data manipulation by printing different separations of the data into RStudio and comparing them to the original dataset by manually sifting through the data to verify.

3.5.2 Verifying the Missing Values, Box Plots and Density Plots

Sample datasets containing missing values were sourced to test the ability of the application in correctly generating a percentage of missing values histogram. Weka was used as a comparative tool to test this function. A simple dataset containing missing values was uploaded to Weka and, using the pre-process facility in Weka, a percentage of missing values per variable was produced. These values were compared to the values generated by the application and were verified.

The data visualisation functionality of the application was verified by comparing the box plots and density plots generated by the application to that generated in Orange. The comparative results of the box plot and density plot were created using the Iris dataset. Figures 25 and 26, both display box plot comparisons of the Petal Length variable plotted by class (setosa, versicolour and virginica). The first image was rendered using Orange and the second image was created using the application.

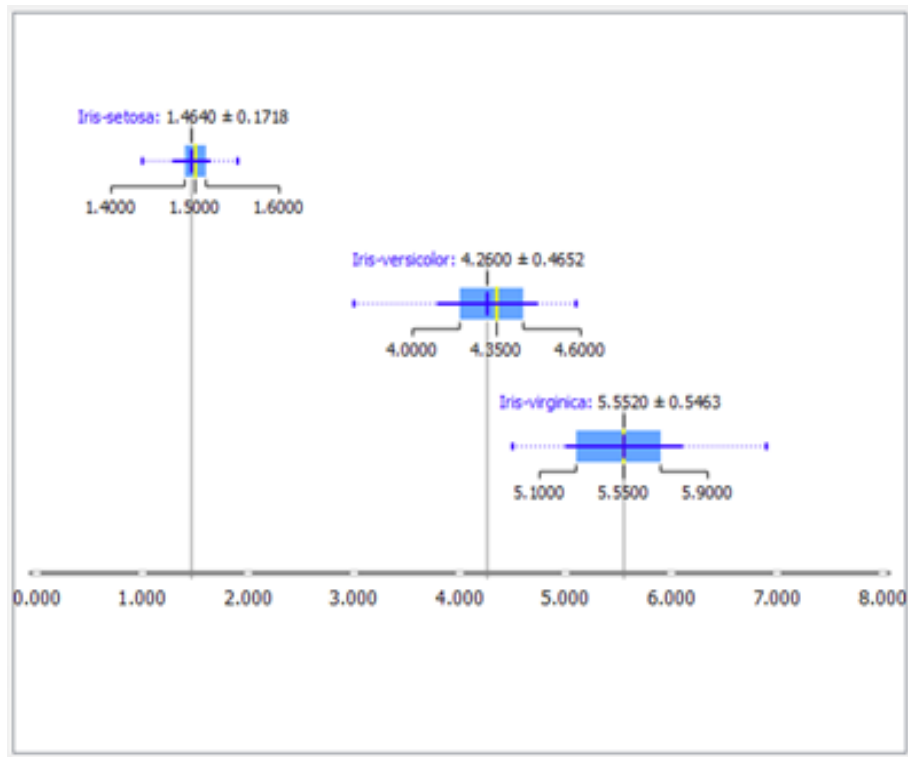


Figure 24: Box plot of Petal Length by class. Created in Orange.

It can be seen in the above box plots that the median value for the setosa class is 1.5, the versicolour class is 4.35 and the virginica class is 5.55.

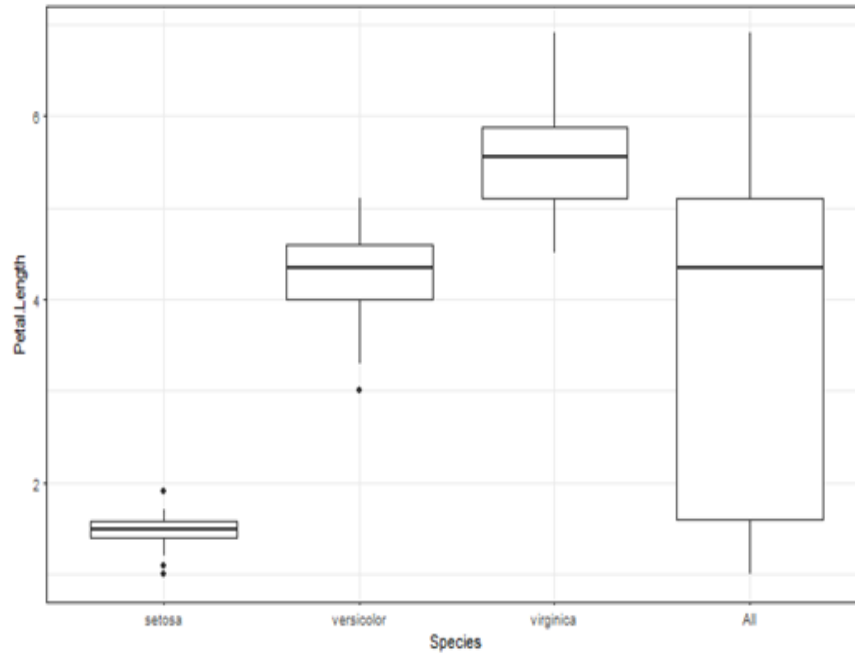


Figure 25: Box plot of Petal Length by class. Generated by the application.

It can be seen in the above image that the box plots created by the application accurately match those created in Orange. The application creates an added box plot which visualises all classes per specified variable. It can also be noted that the box plots generated by the application are flipped right by 90 degrees in comparison to the Orange box plots.

The same process described above was used to verify the density plots. The following two images in figures 27 and 28 are density plot comparisons of the Sepal Length variable plotted by class (setosa, versicolour and virginica). The first image was rendered using Orange and the second image was created using the application.

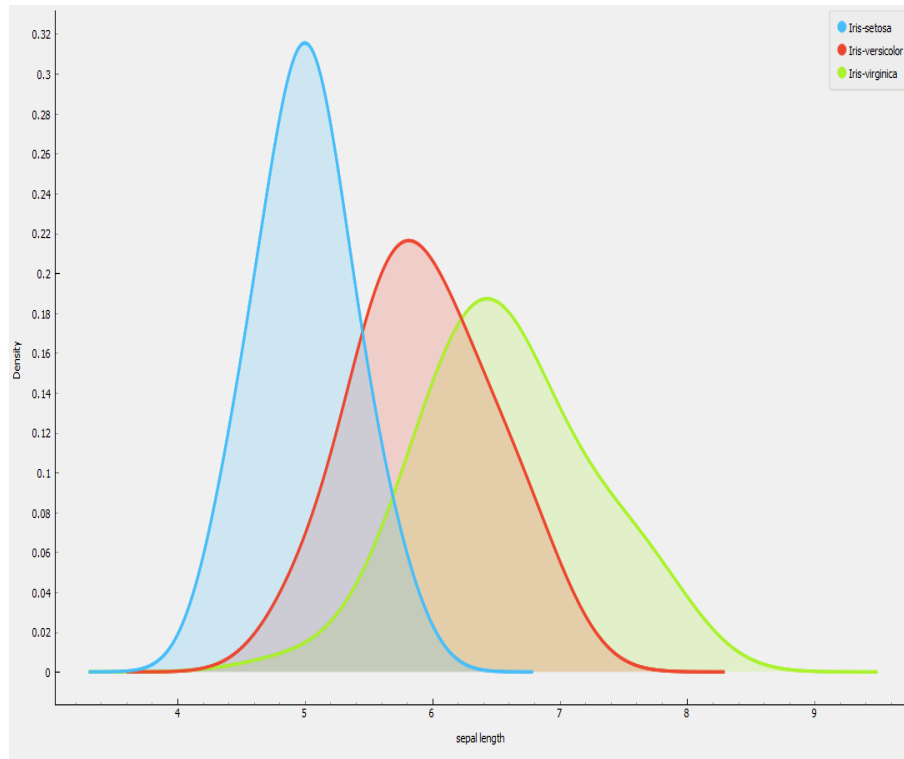


Figure 26: Density plot of Sepal Length by class. Created in Orange.

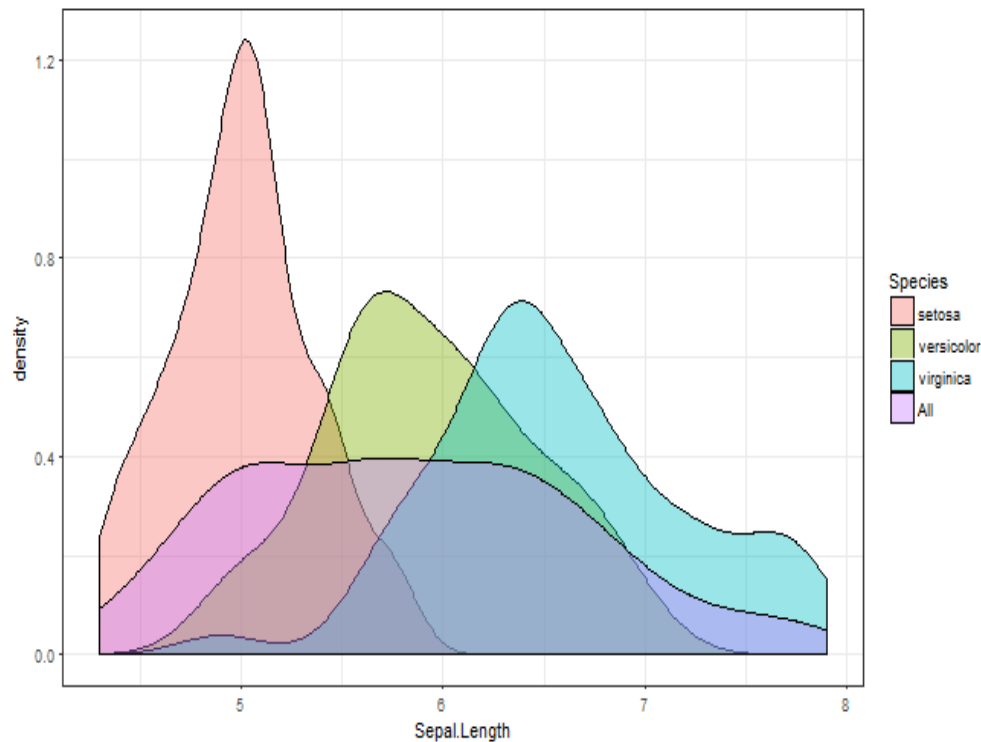


Figure 27: Density plot of Sepal Length by class. Generated by the application.

It can be seen in the above images that the density plots created accurately match those created in Orange. The application creates an added density plot which visualises all classes per specified variable. It can also be noted that the data distribution was normalized to range from zero to one in the Orange density plots, hence the difference in the Y axis values between both images.

3.5.3 Testing the Classification

The classification component of the application required input from the user to decide the percentage split of the data, the number of neighbours (for KNN) and which classification algorithm to employ. Appropriate testing and validation of these functions was carried out. There was also a need to test the ability of the application to classify new data based on the trained models.

The percentage split function was tested by running tests on different percentage splits and ensuring that the correct amount of data was allocated to the train and test sets. This function was tested on smaller datasets to check if the percentage split correctly rounded down

or rounded up according to the number of instances in the data.

The user input for the number of neighbours for K-nearest neighbours classification was tested using Weka. This involved running the KNN algorithms with both Weka and using the application for varying number of neighbours and comparing the results of each to verify the functionality of the number of neighbours input.

The main function of the application is to deliver accurate machine learning capabilities which meant it was imperative to ensure all classification algorithms were operating correctly on the data. Weka was used as a benchmark for comparative testing when running the different classification algorithms. The application generated classification experiment results for the various algorithms. The results were compared and verified with the results produced in Weka which included [15]:

- **Accuracy:** The rate at which the model correctly classifies the test data.
- **Confusion Matrix:** Table of prediction results.
- **Precision:** The number of positive predictions divided by the total number of positive class values predicted.
- **Recall:** (Sensitivity) The number of positive predictions divided by the number of positive class values in the test data.
- **F-Measure:** Conveys the balance between the precision and the recall.

The application facilitates classification of new data based on the model trained. The user can inspect the results as listed above for each classification algorithm. Using the information drawn from the experiment results, the optimum classification model can be chosen to classify new data. This function was tested by randomly sampling 20 instances of the Iris dataset. Initially, the class of each instance in the sample was noted to be used as a comparison after the data had been classified. Using the data description component of the application, the sample Iris data was printed to the output window and is shown in figure 28 on the next page.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
7.60	3.00	6.60	2.10	virginica
5.90	3.00	4.20	1.50	versicolor
6.80	3.20	5.90	2.30	virginica
5.70	3.00	4.20	1.20	versicolor
4.60	3.10	1.50	0.20	setosa
6.40	2.80	5.60	2.10	virginica
4.90	3.10	1.50	0.20	setosa
6.70	3.10	5.60	2.40	virginica
5.00	3.40	1.50	0.20	setosa
4.60	3.20	1.40	0.20	setosa
5.40	3.40	1.70	0.20	setosa
6.30	2.50	5.00	1.90	virginica
5.20	4.10	1.50	0.10	setosa
5.10	3.40	1.50	0.20	setosa
5.10	3.80	1.90	0.40	setosa
6.70	3.00	5.00	1.70	versicolor
4.80	3.00	1.40	0.10	setosa
5.70	2.90	4.20	1.30	versicolor
5.60	3.00	4.50	1.50	versicolor
6.30	3.30	6.00	2.50	virginica

Figure 28: Table of random sample of Iris data with associated classes.

The target classes of the sample data were then withheld and the sample instances were fed into the chosen classification model. The application then printed out the prediction of each sample based on the trained model. The new predictions were then compared to the original sample data containing the correct classes. The classification results were found to be accurate. An example output of this function using the PART classifier (Rpart in R) is shown in figure 29 on the next page.

```

[1] "Algorithm selected: rpart"
[1] "Training set: 70%"
[1] "Testing set: 30%"

```

Actual	Predicted		
	setosa	versicolor	virginica
setosa	12	0	0
versicolor	0	15	2
virginica	0	1	15

```

Correctly Classified Instances: 42
Accuracy: 0.9333333

Incorrectly Classified Instances: 3
Inaccuracy: 0.0666667

```

	Precision	Recall	F_Measure
setosa	1.0000000	1.0000000	1.0000000
versicolor	0.9375000	0.8823529	0.9090909
virginica	0.8823529	0.9375000	0.9090909

	1	2	3	4	5	6	7	8
virginica	versicolor	virginica	versicolor	setosa	virginica	setosa	virginica	
	9	10	11	12	13	14	15	16
	setosa	setosa	setosa	virginica	setosa	setosa	setosa	virginica
	17	18	19	20				
	setosa	versicolor	versicolor	virginica				

```

Levels: setosa versicolor virginica

```

Figure 29: Classification results of the unlabelled Iris sample.

By comparing figures 28 and 29, it can be determined that the model predicted 19 out of 20 of the sample cases correctly (the 16th case was predicted as virginica but was actually versicolor as seen in the table). These results verify the application's ability to classify new data based on a trained model.

3.5.4 Testing the Clustering

The clustering component of the application required input from the user to decide the variables to plot and the number of clusters for the k means clustering. Different K-means plots were generated in Weka for specified numbers of clusters and these plots were compared to the plots created with the application for the same number of clusters. Again, using Weka for comparative analysis, the cluster plots for varying cluster numbers were validated and verified. The sample K-means plots using Weka and the application are shown on the next pages with Sepal Length plotted on the X axis and Petal Length plotted on the Y axis. The cluster count, K, is 3 and the points in each cluster are coloured.

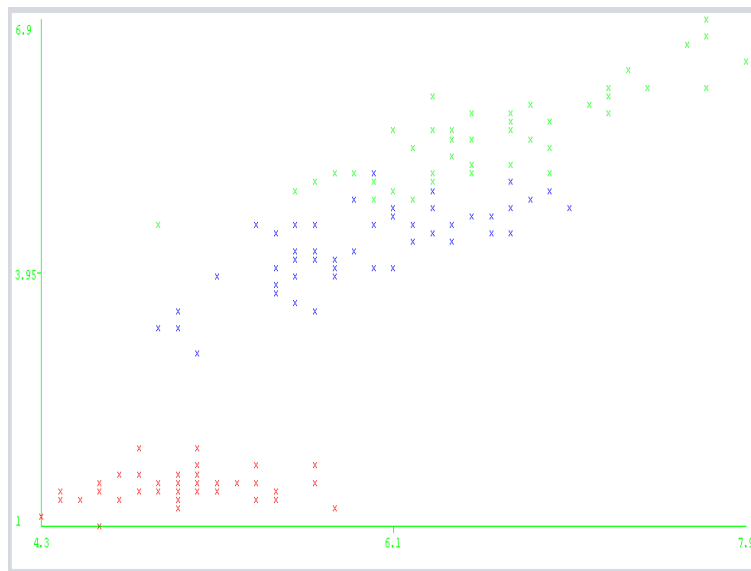


Figure 30: K-means clustering of Sepal Length versus Petal Length with 3 clusters using Weka.

Cluster Results

NAs have been replaced with a mean value

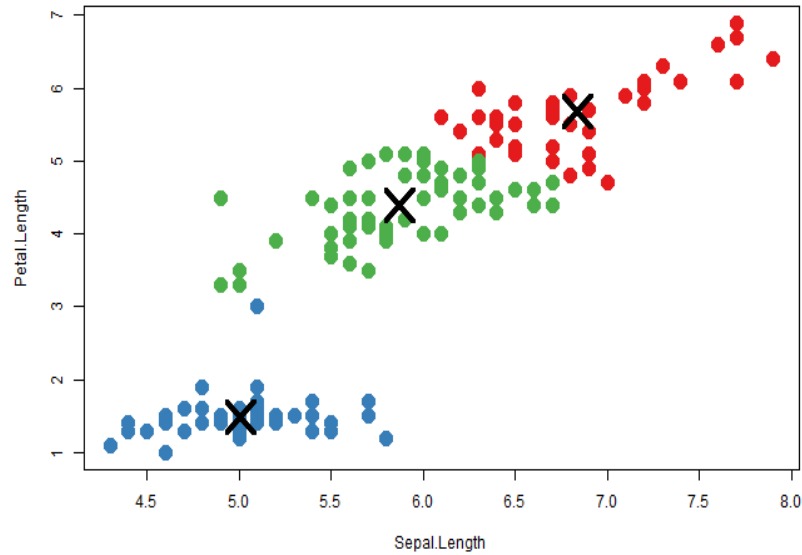


Figure 31: K-means clustering of Sepal Length versus Petal Length with 3 clusters using the application.

It can be seen in the above images that the cluster assignments are very similar with 3 clusters clearly plotted in each. It is also important to note that, since K-means clustering is an unsupervised learning task, the class of each instance is unknown to the algorithm. However, the 3 clusters created by the application outline correctly the different classes within the dataset. A scatter plot of Sepal Length versus Petal Length separated by class was created using Orange to highlight the accuracy of the above K-means clustering when uncovering the underlying trend in the data.

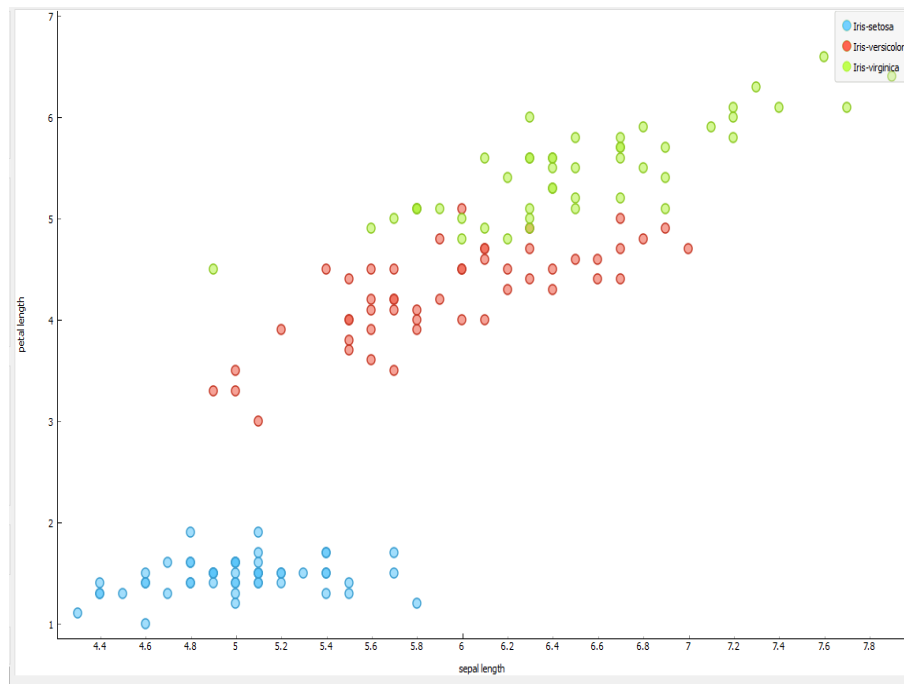


Figure 32: Scatter plot of Sepal Length versus Petal Length separated by class.

The hierarchical clustering tree generated by the application was validated by comparison with a hierarchical tree created in Orange.

3.5.5 Scatter Plots

Orange was used again as a comparative tool to verify the scatter plots created by the application. The scatter plot matrix outlines all bi-variate scatter plots which can be generated from the data along with values for the correlation coefficient and p-values. These values were verified by entering the individual scatter plot data into the built-in correlation coefficient and p-value functions in R.

3.6 Improvements for the Application

From the beginning of this project, the application has been changing and evolving. New improvements have been constantly implemented throughout the development process and some further improvements that could be considered are listed in this section.

- **Added Data Input Features:** At present, the application can only take CSV files as input. As previously discussed, the ability to take in other data file types such as ARFFs is a feature that would increase the range of the application's capabilities. It would also be highly beneficial to allow users to upload data with more than one target class variable for classification tasks. This would allow users to choose which variable they would like to be set as the output target and would relabel the remaining data as the input feature variables. Another development to the data structure compatibility of the application would be to program the application to perform classification tasks on non categorical target output variables.
- **Added Algorithms:** The application carries out a range of different classification algorithms. However, it would be possible to add more classification and clustering algorithms to the application and also add more input parameters for the user to fully experiment with the different algorithms. The addition of cross validation data splitting for the classifications would be an interesting added input parameter. This would allow users to vary the number of folds for the classification and analyse the different results.
- **Added Visualisations:** Some data visualisation options could be added to the application such as heat maps and Venn diagrams. This would further aid users in getting an overall understanding of the data. It would also be recommended to introduce tree visualisations for the decision tree classifiers.
- **Description of Processes:** This application aims to teach users to understand and interpret their data through data visualisation, machine learning and exploratory analysis. To further add to the learning experience, a description of the key processes and visualisation techniques could be introduced so users understand the underlying methodology and technicality that is involved in creating these plots, classifications and clusters.

4 Application Use in Sports Data Analysis

This section of the thesis outlines the potential use of the application in sports data analysis. A summary of the application implementation will be described and the sample biomarker dataset will be explained. An extra component is added to the application to aid users in analysing the results produced by the application.

4.1 Implementation of the Application with Biomarker Data

As mentioned in section 2.5.1, the analysis of biomarker data can uncover signs of fatigue, over-performance, injury and bad nutrition in athletes. The aim of this section is to implement the application with a set of biomarker data in order to generate visualisations of the dataset and employ some machine learning algorithms to make predictions about the data. After adequate biomarker data has been gathered for a number of athletes, different classes can be associated to the instances of the data based on the similarity of biomarker variable values. These classes categorise the players based on their biomarker readings and determine the health and state of each player. This enables informed decisions to be made regarding the future action of the players. The application of machine learning techniques, such as a classification algorithm, to the labelled biomarker data will infer a model to which new biomarker data can be applied and classified. This process would group the athletes and inform them of actions to take to improve future performance by highlighting indicators of over-training, injury and malnutrition. A user can test the ability of the different algorithms to classify the data based on the performance analysis of each algorithm. This means the user can then choose the optimum algorithm for classification of new data. The option of clustering the data is available to the user to uncover hidden patterns in the data which would allow them to draw interesting conclusions from the biomarker data of the athletes.

4.2 Biomarker Dataset

Biomarker data for a large number of athletes was required for this section of the thesis but unfortunately, real biomarker datasets are extremely difficult to source due to the high confidentiality of the data. An inquiry for a sample biomarker dataset was made to the Insight Centre for Data Analytics but due to the confidentiality of the information, a dataset could not be provided. However, Insight provided a sample structure for a biomarker dataset to which example biomarker readings could be applied.

Since the option to implement real biomarker data had been ruled out, it was decided to populate a biomarker dataset with example values based on research into the different readings of common biomarkers used in sports data analysis. Vitamin D, creatine kinase, c-reactive protein and vitamin B12, which are described in section 2.5.1, were chosen as adequate example biomarkers for this experiment. After researching the varying levels of each biomarker, a sample dataset was created with values ranging from low to high for each biomarker. The

biomarkers were entered under the variable names *vd* (vitamin D), *cr* (creatine kinase), *cp* (c-reactive protein) and *bt* (vitamin B12). These variables are all expressed as nanograms per millilitre of blood (ng/mL). The data consists of 480 observations, each representing a different athlete. Each observation is divided into four feature variables, the different biomarkers and one target output variable, the class. Sixteen classes labelled A-P were assigned to the data to group biomarker readings for different instances based on similarity. Each class provides information on the instance to which it is assigned and can be used to make decisions and predictions for the associated athletes.

4.3 Added Results Component

In order to aid users in the interpretation of their results, a new component was added to the application. This component contains information on the biomarker data which can help users make decisions based on the results produced by the classification of the biomarker data. The user has the option to print a table outlining the traits associated to each class in the biomarker data. This table is shown in figure 33 below.

About Biomarker Data

Choose an Option:

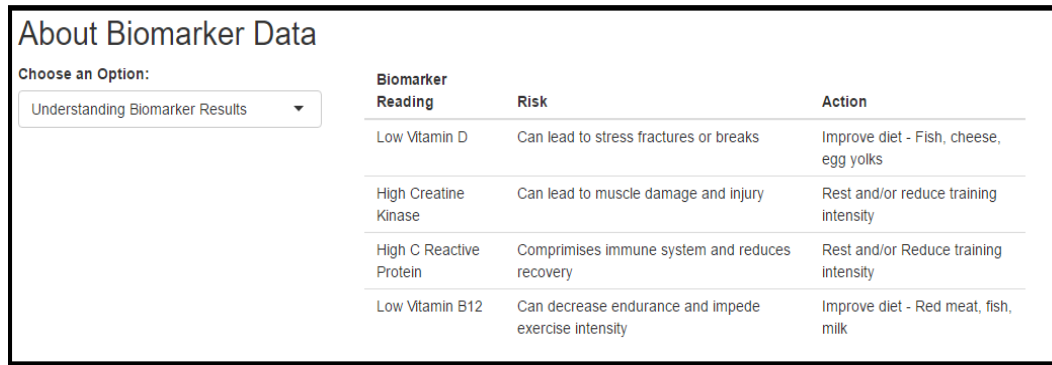
Biomarker Classifications

Class	Attributes
A	High Creatine Kinase, High C Reactive Protein
B	Low Vitamin D, High Creatine Kinase, High C Reactive Protein
C	High C Reactive Protein
D	High Creatine Kinase
E	High Creatine Kinase, High C Reactive Protein, Low Vitamin B12
F	Optimum
G	High Creatine Kinase, Low Vitamin B12
H	Low Vitamin D, High C Reactive Protein
I	Low Vitamin D, High Creatine Kinase
J	Low Vitamin D, High Creatine Kinase, High C Reactive Protein, Low Vitamin B12
K	High C Reactive Protein, Low Vitamin B12
L	Low Vitamin B12
M	Low Vitamin D, High Creatine Kinase, Low Vitamin B12
N	Low Vitamin D, High C Reactive Protein, Low Vitamin B12
O	Low Vitamin D
P	Low Vitamin D, Low Vitamin B12

Figure 33: Biomarker classifications with associated attributes.

It can be seen above that the different classes indicate different traits in the biomarker data. For example, if an athlete is classified into group A this means the athlete has high readings of both creatine kinase and c-reactive protein which is undesirable.

The user can then make informed decisions on the action to take to improve these readings for the athlete. To further aid the user in understanding the results and decide suitable actions to take, another table can be printed to describe the risk involved with each undesirable biomarker reading and the action to take to improve the readings. This table can be seen in figure 34 below.



Biomarker Reading	Risk	Action
Low Vitamin D	Can lead to stress fractures or breaks	Improve diet - Fish, cheese, egg yolks
High Creatine Kinase	Can lead to muscle damage and injury	Rest and/or reduce training intensity
High C Reactive Protein	Compromises immune system and reduces recovery	Rest and/or Reduce training intensity
Low Vitamin B12	Can decrease endurance and impede exercise intensity	Improve diet - Red meat, fish, milk

Figure 34: Risk and actions associated with the biomarker readings.

4.4 Experiment

To analyse the performance of the application in visualising, classifying and clustering the data, the following experiment was carried out and the results were analysed.

4.4.1 Data Input and Visualisation

The biomarker data described in section 4.2 was uploaded to the web application. The tables generated in the first component of the application were validated by comparing the data printed in the tables to the biomarker data. The application correctly determined the number of variables (5), the number of observations (480) and the number of classes (16). The application also correctly resolved the number of observations per class (30 observations per class). The dataset was printed in a table and verified by comparing it to the input biomarker dataset.

Some missing values exist in the dataset to verify the performance of the application in correctly identifying and plotting the percentage of missing values. There were 2 values missing from the *vd* variable, 1 value missing from *cr* variable and 3 values missing from *cp* variable. These values were presented as percentages of the number of observations of each variable in the histogram in figure 35 on the next page.

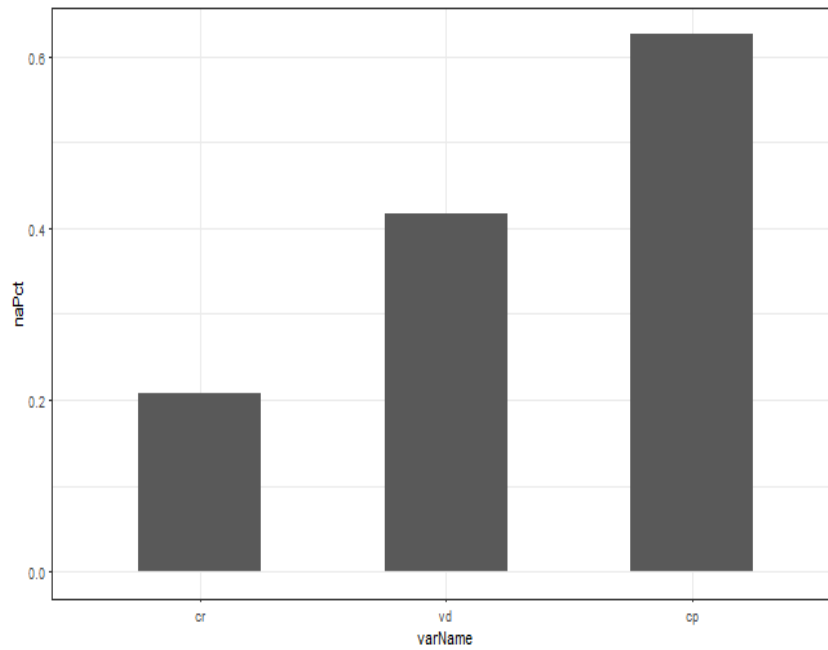


Figure 35: Percentage of missing values per variable.

The application produced detailed box plots and density plots of the data distribution which gives a good overall idea of the spread of the biomarker data, giving the user a reference of the range of values of each class per variable selected. Figure 36 and figure 37 on the next pages highlight the box plot and density plot outputs for the *vd* variable classes.

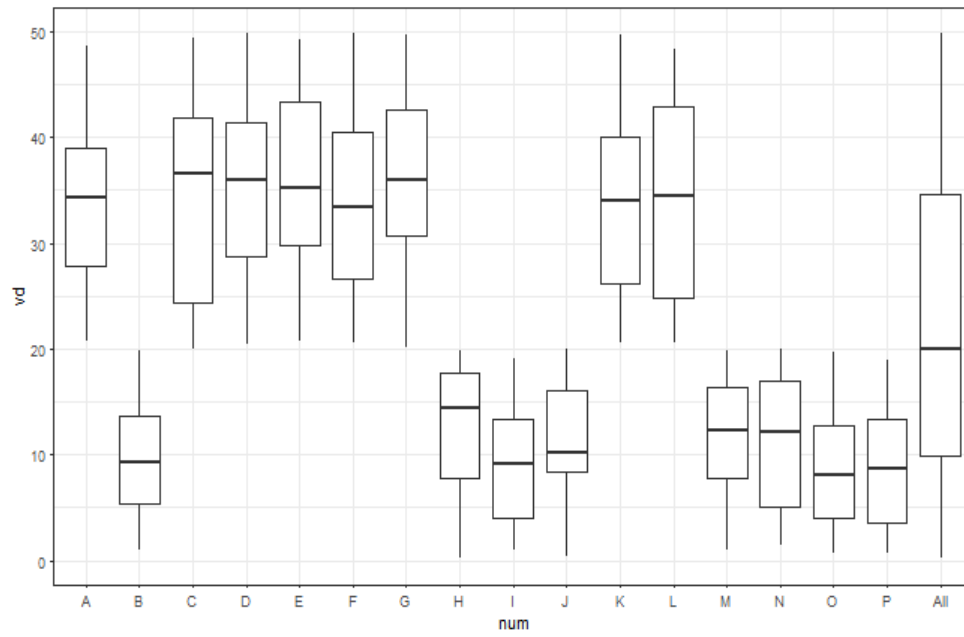


Figure 36: Box plot representation of *vd* variable for all classes.

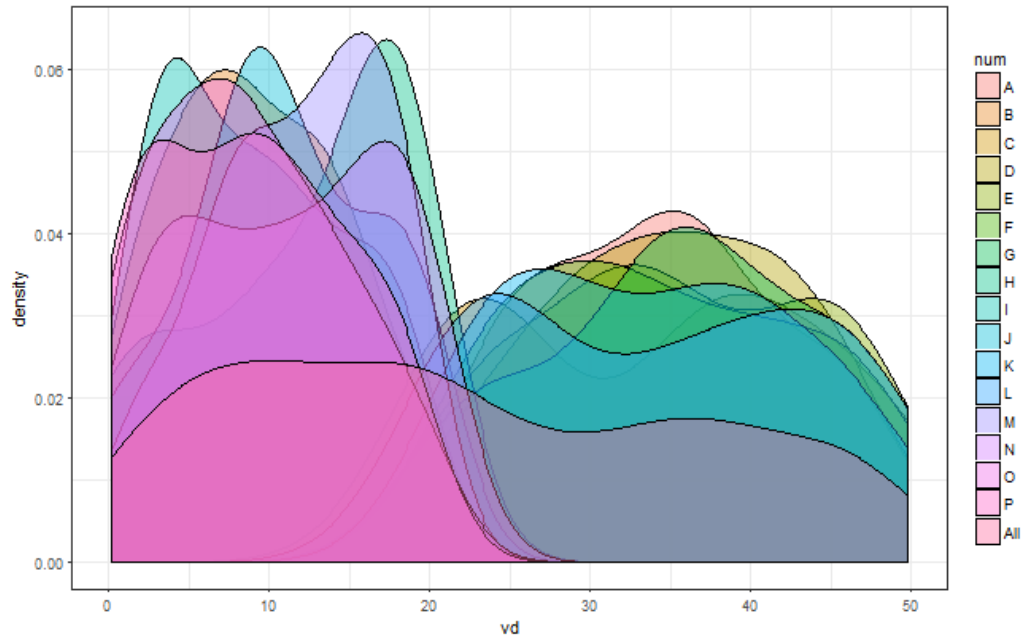


Figure 37: Density plot representation of *vd* variable for all classes.

It can be determined from the above visualisations that the readings for the *vd* variable for vitamin D range between approximately 0 and 50 ng/mL and there is a clear indication that classes are divided between the low and high vitamin D readings. Hence, it can be said that the difference in high and low values of the vitamin D readings is a strong factor in the classification of the athletes' biomarker data. These visualisations can be repeated for all variables in the dataset to further aid in understanding the structure and distribution of the data as a whole.

4.4.2 Classification

The aim of the classification of the biomarker data in this experiment was to test the different algorithms available and analyse the performance of each algorithm. After the best performing algorithm was determined, the model that had been created by the algorithm was used to classify a new set of unlabelled biomarker data. The results of this classification were analysed with reference to the new biomarker analysis component of the application and suitable actions were decided upon for each athlete.

Percentage Split: The data was split into 75% for training and 25% for testing which is considered a fair split for classification tasks. A range of different percentage splits were tested on the available algorithms and this 75:25 split was chosen since it provided enough training data to optimise the accuracy without compromising the precision of the classifiers. This split remained the same for all algorithms to ensure fair testing of algorithms.

Classification Results: The classification results were generated by the application and the key results were noted. These results can be seen in the table 1 which summarises the accuracy, precision, recall and f-measure of each algorithm used.

Algorithm	Accuracy	Precision	Recall	F-Measure
PART	0.9	0.88	0.89	0.885
Random Forest	0.9417	0.938	0.9373	0.9341
LDA	0.775	0.787	0.7748	0.7643
SVM	0.7833	0.7826	0.7897	0.7725
KNN (K=5)	0.3	0.311	0.312	NA
Naive Bayes	0.817	0.806	0.809	0.798

Table 1: Performance measures of the classification algorithms.

Based on the results above, it was determined that the Random Forest performed best when classifying the data followed closely by the PART classification algorithm. Clearly, the decision tree based classifiers perform well for this task which relates to the fact that decision trees are well suited to classification tasks involving categorical target variables. Decision tree performance values can often be misleading due to over-fitting, however, the ensemble technique of the Random Forest eliminates this issue. Hence, the Random Forest algorithm is confirmed to be the most suited algorithm to the task. The Linear Discriminant Analysis, Support Vector Machine and Naive Bayes classifiers similarly performed quite well on this task. The K-Nearest Neighbour algorithm performed very poorly on this task. The number of neighbours, K, was varied to generate the best possible performance results for the algorithm. The optimum value for K was determined to be 5, however, this still resulted in poor performance with accuracy of 30%. KNN is not suited to classification tasks involving large datasets such as this. This issue is commonly referred to as the 'curse of dimensionality'[16] which describes how KNN is better suited to datasets of smaller dimensions since added dimensions increase the input space exponentially. Another possible cause of the poor performance of KNN is due to the fact that KNN does not perform well with wide scale ranging data. Normalizing the dataset to scale the values from 0 to 1 would likely have increased the performance of KNN.

The Random Forest was chosen as the classifier for the new unlabelled biomarker data. As previously stated, it was not possible to source real biomarker readings so for this part of the experiment a random sample of 15 instances was extracted from the original input biomarker dataset to be used as the new data of 15 hypothetical athletes to predict their class. The class associated to each instance in the sample set was noted to be later compared to the predicted class of each athlete. The target classes were then removed from the new sample data. This unlabelled sample data was then fed into the trained Random Forest model to return the class predictions for each of the 15 athletes. Table 2 on the next page shows the results of the classification using the Random Forest algorithm.

Athlete	Predicted	Actual
1	C	C
2	D	D
3	G	G
4	B	B
5	D	D
6	J	J
7	G	G
8	I	I
9	N	N
10	K	K
11	G	G
12	L	L
13	B	B
14	E	E
15	J	J

Table 2: Predicted and actual class of the new sample athletes.

As expected, the Random Forest classified the new data extremely well. The model classified each instance correctly due to the high performance of the trained model. These results show that, in the case of a user attempting to classify new biomarker data where the class is unknown using the trained Random Forest model, the predictions are trustworthy and accurate. Using the predicted results, the user can refer to the biomarker analysis component of the application to determine what the results say about the athlete and help determine the action to take to improve the performance and health of the athlete. Based on the table of results above and taking athlete 1 as an example, the user can refer to the biomarker analysis page and find that class C indicates this athlete has high c-reactive protein readings which can compromise the immune system of the athlete and reduce recovery. The athlete is recommended to rest or reduce training intensity.

4.4.3 Cluster:

The clustering of the data in this experiment grouped the data together based on the similarity of the data points. The biomarker data was clustered by the application and the cluster number was varied for the different feature variables. The elbow curve below in figure 38 was produced in R and used to determine the optimum cluster number for the K-means clustering of variables *cr* and *bt* which represent the creatine kinase and vitamin B12 readings of the athletes.

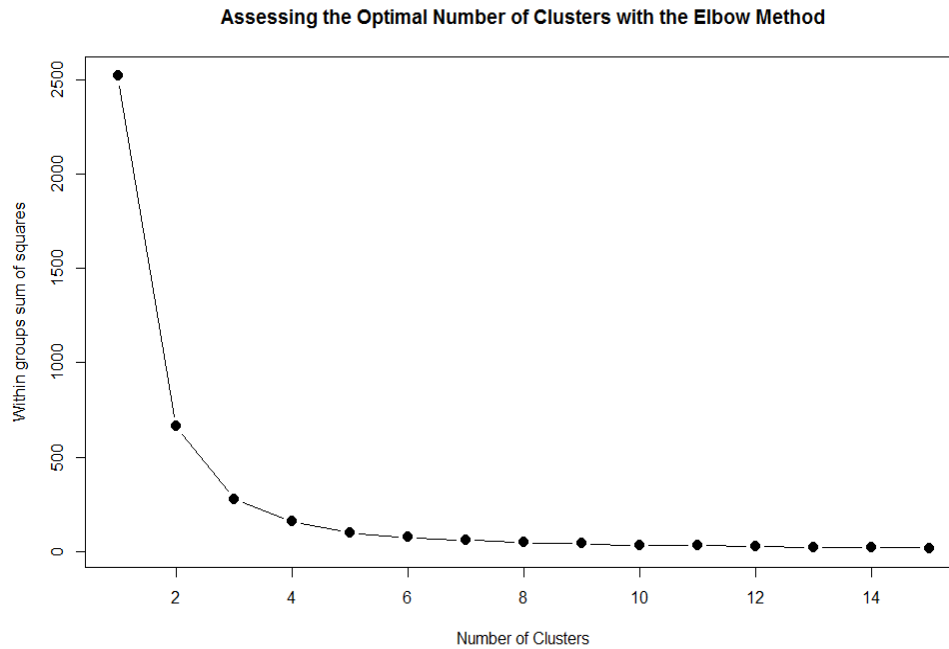


Figure 38: Elbow curve for cluster number selection.

The elbow curve shows that “the solution criterion value within groups sum of squares will tend to decrease substantially with each successive increase in the number of clusters” [17]. The optimum number of clusters can be confirmed by the 'elbow' of the curve. In this case, it can be said with reasonable confidence that 4 clusters is the optimum K value for clustering the data for the *cr* and *bt* variables. The cluster output of the application for this example is shown below in figure 39 on the next page.

Cluster Results

NAs have been replaced with a mean value

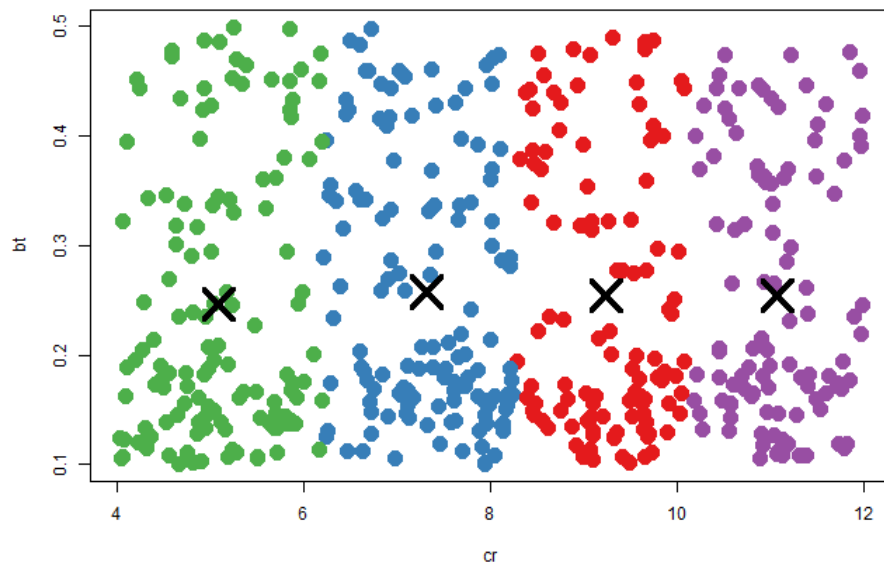


Figure 39: Cluster assignments for the *cr* and *bt* variables with K=4.

Based on the results above, it is clear that there are a well defined set of 4 distinct groups when it comes to understanding and analysing the readings of creatine kinase and vitamin B12. The cluster centres are marked on the plot with a black X and are listed in the table below.

Variable	Cluster 1	Cluster 2	Cluster 3	Cluster 4
cr	5.050094	7.279206	9.248431	11.082811
bt	0.2436545	0.2600982	0.2540872	0.2545578

Table 3: Cluster centres for the *vd* and *bt* variables

It is clear, from the table above, that the clusters are distinguished by varying creatine kinase levels with vitamin B12 levels remaining relatively stable at approximately 0.25. By implementing the box plots, as previously explained, to determine the range of the data values per variable it can be determined that 0.25 is a moderate vitamin B12 reading. By referring to the box plots for the *cr* variable, the levels of creatine kinase are shown to range between approximately 4 to 12 ng/mL. Hence, based on the readings displayed in table 3, the values for *cr* can be said to range from low to high across the clusters.

In this experiment, if an athlete was to be categorized into cluster 4 it can be said with a reasonable level of confidence that the athlete has a average level of vitamin B12 and a high

level of creatine kinase in his blood. Again, by referencing, the biomarker analysis component of this application, it can be found that this athlete would be at risk of muscle damage or injury and should rest or reduce training load.

4.4.4 Other Visualisation:

Using the Visualise component of the application, a further understanding of the biomarker was attained. Bi-variate scatter plots were generated for the various different variables in the dataset to highlight the relationship between the variables.

5 Conclusions

This section outlines the final conclusions drawn from the work and research carried out in this project. A summary of the work carried out is provided with some possibilities for future work. Finally, some concluding remarks will close out this thesis.

5.1 Work Summary

An in-depth literature review was accomplished in the early research stage of this project. Some useful visualisation techniques were introduced and explained. Common machine learning and data mining concepts were then outlined which involved insightful descriptions of a range of classification and algorithms clustering algorithms. Next, the application of statistical analysis in sport was explored. Research into specific use cases of data visualisation and statistics used in sports was carried out along with some valuable observations on the potential use of biomarker analysis in sports. This was followed by a technical review of some existing technologies and applications used for machine learning and data mining tasks. The literature review had a great influence on the work carried out in this project which was described in the closing section of the literature review.

After completing an extensive literature review, the application design and implementation was initiated. The first design phase described the initial plans for the application. This was followed by the next design phase which implemented improvements on the original design with added plotting, data manipulation and machine learning capabilities. The design was tested and validated and applied to a sample scenario involving biomarker data of a number of athletes.

5.2 Directions for Future Work

The following points describe some of the potential future work that can be carried out following this project:

- **Improvements to the Application:** There are many added features and improvements that can be applied to the existing application to exploit its full potential. Much of these improvements have been discussed in section 3.6. It was mentioned in section 3.6 that new algorithms and visualisations could be added as features of the application along with a description of all processes. Another potential addition to the application could be the introduction of more results for the K-means clustering component. A printed table summarising the cluster centres and a plot of the elbow curve would be desirable additions.
- **User Accounts:** It would be a good idea to introduce a user account system. This would require a database to store the account details of the users and would require a secure login page. The implementation of the account system would provide added security when

dealing with datasets of high confidentiality, especially information such as the biomarker data of athletes since this data is directly indicative of the athletes' form which is important and sensitive information. From their account, a user could save a log of their activity on the application which would allow the loading of old models and visualisations.

- **Further Application in Sports Data Analysis:** It would be interesting to carry out further research in to sports data analysis to determine the potential use of the application for non-biomarker related data in sports. Concussion determination and prevention is a topical issue in sports today. Some research into this area would provide insight into the determining signs of concussions in athletes and the possible prevention measures that can be taken. This data could then be loaded into the application and possible predictive analysis, based on the existing visualisation and machine learning facilities, could be carried out to determine whether the athlete has been concussed and provide measures to treat concussions and prevent future occurrences.

5.3 Concluding Remarks

Machine learning and data mining solutions are becoming increasingly widespread in many real-world problems including computer science, business, healthcare and sports. The growing need for such solutions means there is a great potential for the application put forward in this thesis. The application's simple user interface and detailed graphics make it suitable for learning complex statistical analytics in a user-friendly, visual environment. The possible improvements for the application set up a broad scope for future work.

References

- [1] Enda Barrett. *1617-CT417 Software Engineering III*. 2016.
- [2] Statistics How To. *The Linear Regression Equation*. 2017. URL: <http://www.statisticshowto.com/how-to-find-a-linear-regression-equation/> (visited on 01/10/2017).
- [3] Phil Simon. *Too Big to Ignore: The Business Case for Big Data*. Wiley, 2013. 258 pp.
- [4] Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.
- [5] Michael Madden. *1617-CT475 Machine Learning and Data Mining*. 2016.
- [6] Trevor Hastie Gareth James Daniela Witten and Robert Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Springer, 2013.
- [7] AK Jain. “Data clustering: 50 years beyond K-means”. In: (2010).
- [8] Wikipedia. *Biomarker*. 2017. URL: <https://en.wikipedia.org/wiki/Biomarker> (visited on 09/13/2016).
- [9] Outside. *What Some Common Athletic Biomarkers Say About Your Health*. 2016. URL: <https://www.outsideonline.com/1857911/what-some-common-athletic-biomarkers-say-about-your-health#slide-6> (visited on 02/23/2017).
- [10] Richard Burden Glyn Howatson Charles R. Pedlar Nathan A. Lewis John Newell. “Critical Difference and Biological Variation in Biomarkers of Oxidative Stress and Nutritional Status in Athletes”. In: (2016).
- [11] Renaud Gaujoux and Cathal Seoighe. “A flexible R package for nonnegative matrix factorization”. In: (2010).
- [12] TR Golub JP Mesirov JP Brunet P Tamayo. “Metagenes and molecular pattern discovery using matrix factorization. Proceedings of the National Academy of Sciences of the United States of America”. In: (2004).
- [13] Wikipedia. *Orange (software)*. 2017. URL: [https://en.wikipedia.org/wiki/Orange_\(software\)](https://en.wikipedia.org/wiki/Orange_(software)) (visited on 11/10/2016).
- [14] The R Foundation. *What is R?* 2017. URL: <https://www.r-project.org/about.html> (visited on 01/24/2017).
- [15] Machine Learning Mastery. *Classification Accuracy is Not Enough: More Performance Measures You Can Use*. 2014. URL: <http://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/> (visited on 02/15/2017).
- [16] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.
- [17] RPubS. *Example of K-Means Clustering with R*. 2015. URL: <https://rpubs.com/FelipeRego/K-Means-Clustering> (visited on 02/07/2017).