National College of Ireland

BSc in Computing

2017/2018

James Redmond

X14544067

X14544067@student.ncirl.ie

Supervisor: Michael Bradford

Analysis on the horse racing industry

Technical Report

Table of Contents

Executive Summary 3

1 Introduction 4

1.1 Background 4
1.2 Aims 6
1.3 Technologies 6
1.4 Structure .... **Error! Bookmark not defined.**

2 System 9

2.1 Requirements 9
2.1.1 Functional requirements 9
2.1.2 Data requirements 22
2.1.3 User requirements .... **Error! Bookmark not defined.**
2.1.4 Environmental requirements .... **Error! Bookmark not defined.**
2.1.5 Usability requirements .... **Error! Bookmark not defined.**
2.2 Design and Architecture 24
2.3 Implementation 25
2.4 Testing 34
2.5 Graphical User Interface (GUI) Layout 34
2.6 Customer testing 39
2.7 Evaluation 40

3 Conclusions 49

4 Further development or research 50

5 References 51

6 Appendix 52

6.1 Project Proposal 52
6.2 Project Plan 56
6.3 Monthly Journals 56
6.4 Other Material Used 62

Executive Summary

This project involved the implementation of Betfairs API in order to obtain sufficient horse racing data. The primary aim of the project was to implement the data gathered from the API and create a horse rating system and to test the system based off results of races. This report describes and evaluates a range of different areas relating to the overall solution.

The main source of data for the project came from the Betfair API, problems were encountered with obtaining the data as the API is so broad and hard to understand, programs were written in Python to pull the data and automate the download and extraction process to speed up data extraction. The KDD approach was followed when analysing the data. Several algorithms were used for a predictive model. Some small concept programs were written in Python to test the code and application. R was used for the analysis of the data and creating the predictive models for the solution. R shiny was used to create an interactive application allowing the user to look and interact with the data. The visualization of the data allows users to understand the project and data much more and is also easier to pick up on trends within the data. Problems were encountered to obtain sufficient data from the API that would allow for a good solution, this had an impact on the overall project.

1 Introduction

1.1 Background

The reason for my idea came from my own personal interests as my Grandfather was very interested in horse racing, I became interested in it along with my sister being in show jumping competitions. I would attend a horse racing event once or twice a year since I turned 18. I have attended many outside of Dublin where I Live and would like to go to England as well. I was always interested in other things in regards to horse racing for example the trainers who train horses the jockeys, also my close friend and her family owned some race horses which always got me excited and talking and listening to their trainers talk about horses who people have probably never heard before.

So when I started back college this year I didn't really have an idea and was unsure what I could do for my Final year project, I immediately taught sports as that is what I am interesting in and I found that Football was almost a too broad area to do a project on. I then read an article about horse racing online and it came to my knowledge that there were only two official companies/organizations that rated horses being the Racing Post and the British Horse Racing Board, the Racing Post being the more popular option as it is more up to date. I then started to wonder how they rated the horses and I looked into it and they gave the factors that were considered when rating a horse and how it could be rated. I then decided I want to do a further analysis on how race horses get there ratings.

Most bookmakers give their starting odds for horses based of these racing systems but obviously not all the time do the best horses or the favourites win, so I want to do a study and see how these horses are rated and how many of the highest rated horses are successful and basically see how accurate these rating systems are.

I was also curious about making a predictive model to try and determine whether or not a horse won based of the data that was gathered. The horse racing industry is massive and i was interested to do a project based around it to see what kind of cool things I might discover that I didn't know before. Mainly as well just learn new things about the horse racing industry.

1.2 *Project Scope*

The scope of the project is to analyse data that contains various information about horse racing for example race results, race events, jockeys, race conditions etc. I plan to use this data to see what determines a race horses rating also known as an RP (racing post) rating. I will also use an algorithm that takes into account a range of variables to come up with my own rating of a horse and compare my results to the results the racing post got. I can compare this by: based of my ratings each race I can see if the horse who won was the highest rated out of the selected field and then compare them results to that of the racing post. The main scope of the project is to see what factors influence a racehorses rating which will then determine a rating and I can get a better understanding of how a race horse gets it rating, and the main challenge was to identify which factors I wanted to use for my own system.

There is no specific dataset for this project and so I have to get my data through an API, the API I will be using is Betfairs API this contains a lot of data spread across multiple sports, I will have to limit the data to just horse racing and take all the information I can get. Once I get the data I want and how I want it I will use R to run some statistical analysis on the new cleaned data that I have. The idea is to have a good visual representation of the data so that I can see clearly what the data is and how it is being used and the factors that are considered in a horses rating.

With the visualization of the data people will be able to see clearly my outcome from my research which will be how a racehorse gets it rating and some graphs on seeing how accurate a rating is for a horse and whether the best rated horses are the horses that are winning the most, if this gets back some sort of statistic like 80% of the highest rated horses in a race will win then a gambler or someone who just bets on horses could just simply pick the highest rated horse at statistically would win 80% of the time, and if it is a much lower number then it could be that the rating system isn’t that accurate and the so called best horses should be winning but aren’t, I will do more research and look into why the best rated horses aren’t winning and maybe it could be that other horses that are winning their rating could be very close to that of the highest rated horse in the race. So I would then hope to get a visual representation of the rating of all the winners in a random amount of races and compare the winners to the highest rated horse in each race and my output would be how many of the winners were close to the highest rated horse and how far away they were. I would use a line graph to show the results one line which will show the highest rated horses and the other would be the winner of the race.

1.3 Aims

The project aims is to do a statistical analysis on the data I retrieve form the Betfair API. This involves collecting, exploring and presenting large amounts of data from the API to identify and discover underlying trends and patterns. In this project the aim is to discover hidden trends and patterns in the data that as a result can predict horse ratings and so I will have my own system with coming up with horse racing systems, which can then be compared to the Racing Post to see if my system, is more accurate and better than their system. I will be able to test this by comparing my horse ratings to there's and taking individual horses and seeing how successful these horses where and whether my rating system predicted more winners than the Racing Post. Another aim is then if this system is successful I can use these results to make up my own predictive model and come up with my own betting odds and compare them to that of a bookmakers and in turn statistically analyse my chances of predicting winners. So an example of this if my system produced a horse at the odds of 4/1 and the bookmakers were given odds of 2/1 I would be able to identify that I actually have a 4 in 1 chance of winning not a 2 in 1 chance meaning I would have better statistics/chances of what horses I could pick to win a race.

1.4 Technologies

R Studios

I will use R Studios to build this project, which is an open source integrated development environment for R Language which itself is a programming language for statistical computing, graphics and will be the language I write this project in. R studios will be in conjunction with excel as this is where I will store my dataset and use it to retrieve information as needed such as horse racing results and information in regards to the horse racing industry and any data I pull from the API.

R Language

I will use R language to build my project and compute most of my analysis following the KDD approach, create predictive models and implement algorithms to try and solve my solution.

R Shiny

R shiny is an open source R package that provides beautiful and powerful web framework for building web applications. R Shiny will help turn my analysis into an interactive web application without requiring or the use of HTML, CSS, or JavaScript. This allows users to visualize and interact with my data. I will use many libraries from R throughout the project, some include (caret, mlbench, ISLR, ggplot2 etc).

Python

Python is a functional programming language that is easy to use and also easy to learn. Python has a design philosophy that emphasizes code readability and a syntax that allows programmers to express concepts in fewer lines of code than might be used in other languages. The first part of the project focused on extracting data from Betfairs API. I will be using Python to pull data from the API and I will also be using python scripts to convert the data to a csv file so I can use the data in R studio and run R on the data. Python is also used to automate the download and extraction phase in getting the data from the API which makes it much faster at obtaining data also makes it more convenient.

SPSS

SPSS is one of the most popular statistical packages which can perform highly complex data manipulation and analysis with simple instructions.

I will mainly use SPSS to backup what I am displaying using it as more as an end user friendly way of viewing the data so that a person with no technical ability will be able to manipulate the data with built in commands using it as a testing tool will help me solidify the information that I am looking to produce.

Excel

Excel is a spreadsheet tool with built in statistical and graphing commands that allow a user to manipulate information and data loaded onto it. I am going to use excel to hold information in a more structured format until I start to clean it and remove all the data that I do not want.

1.5 Definitions, Acronyms, and Abbreviations

RP – Racing post

2 System

2.1 User Requirements

2.1.1 Self-Intuitive

The system will not require any difficult manuals. A user should be able to easily interact with the application and won't need any previous knowledge about the application to navigate the UI (User Interface)

2.1.2 Friendliness

The system will be friendly for the end user. This should be achieved by having a simple but an attractive User Interface. Using R Shiny to create the interactive UI allows the user to be able to find everything they require and to easily navigate through the application.

2.1.3 Page loading speed

The system should ensure that the loading speed of each page within the application is fast and not slow. As users often get annoyed if they are left waiting for things to load. The data can be quite big so having the loading time of the application very fast can be difficult but effective.

2.1.4 Visualisation Interactions

The system shall allow Users to interact with the data by using R Shiny to allow the user to interact with graph visualisations by being able to change the inputs of the graphs which will then change the output and also being able to zoom in on graphs and get a further inside into the visualisations.

2.1.5 Functional requirements

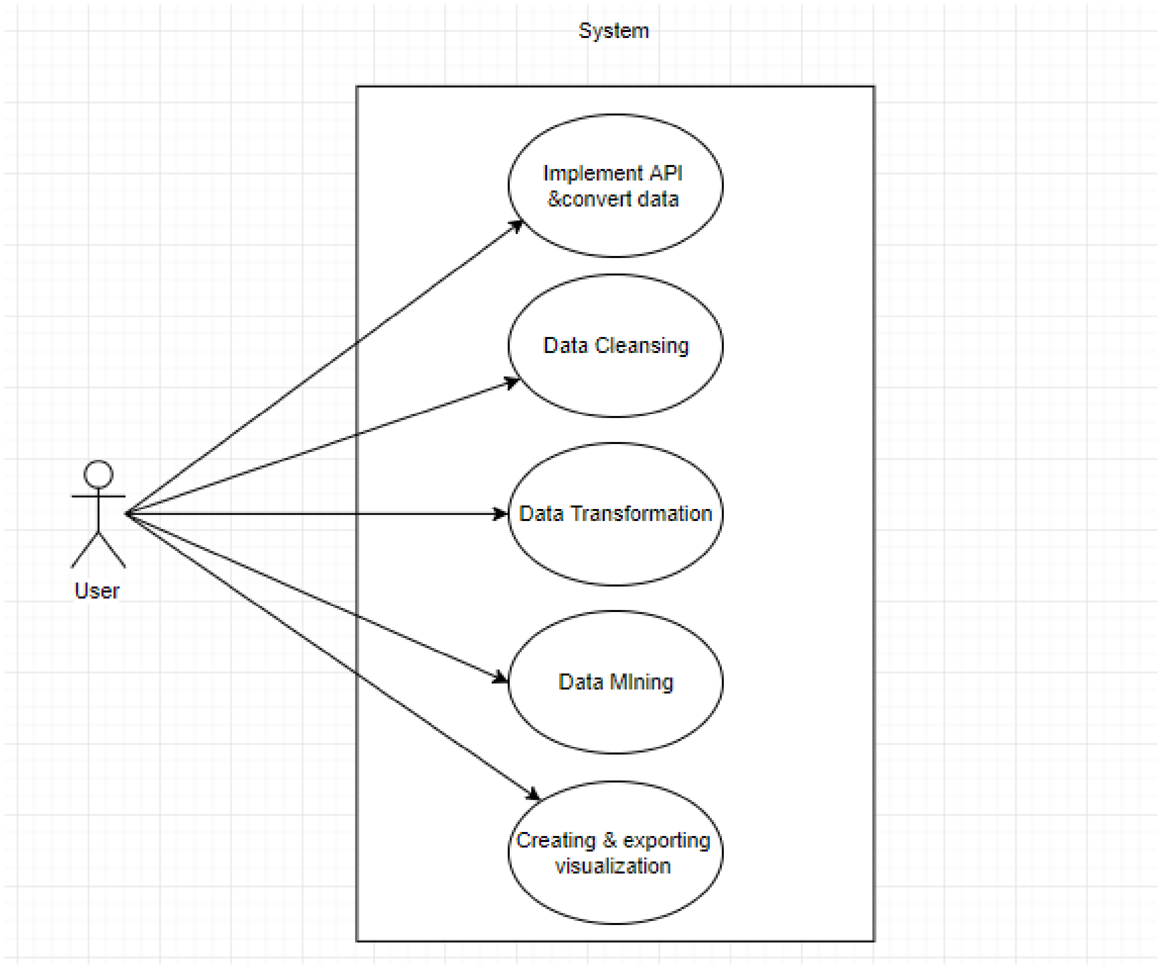
- Implementing API and converting data - the correct data must be obtained and automated from the Betfair API, the data comes in JSON format and must be converted to a csv file using Python. The extraction phase must be automated as well to make it more convenient with obtaining new data from the API.
- Cleaning of the data – The data must be setup correctly any unwanted data must be removed and any data that could make the results skewed must be removed.

- Data Transformation – This involves taking the now clean dataset and begin generating better data for the data mining stage methods for example transforming to normality which prevents data from being skewed, term frequency and dimensionality frequency which is to reduce the number of variables.
- Data Mining – This stage of the project involves using machine learning and statistical algorithms to discover patterns and trends and then output new information.
- Create and export visualization – This involves using R to code and display graphs to output the results of my project and to then use R Shiny to output results, this makes it easier to understand my results with aid of visualization. Most people can understand something easier if they can visualize it. R shiny also allows for users to play and interact with the data. Which makes the application a bit more for and not boring or static so it allows the user to do something rather than just look at graphs and read results.

2.1.6 Use Case Diagram

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

The Use Case Diagram provides an overview of all functional requirements.


System
Implement API &convert data
Data Cleansing
Data Transformation
Data MIning
Creating & exporting visualization
User

2.1.7 Requirement 1 <Implementing API and converting data>

2.1.7.1 Description & Priority

This requirement is the probably the most important and has the highest priority. I am using betfairs API to get my data, because the API is so big I have to obtain the right data and automate it so I don't have to manually download data, without this data I cannot do anything else with the system.

2.1.7.2 Use Case

This use will be representing requirement 1/Implementing API and converting data to a csv file

Sequence: (1)

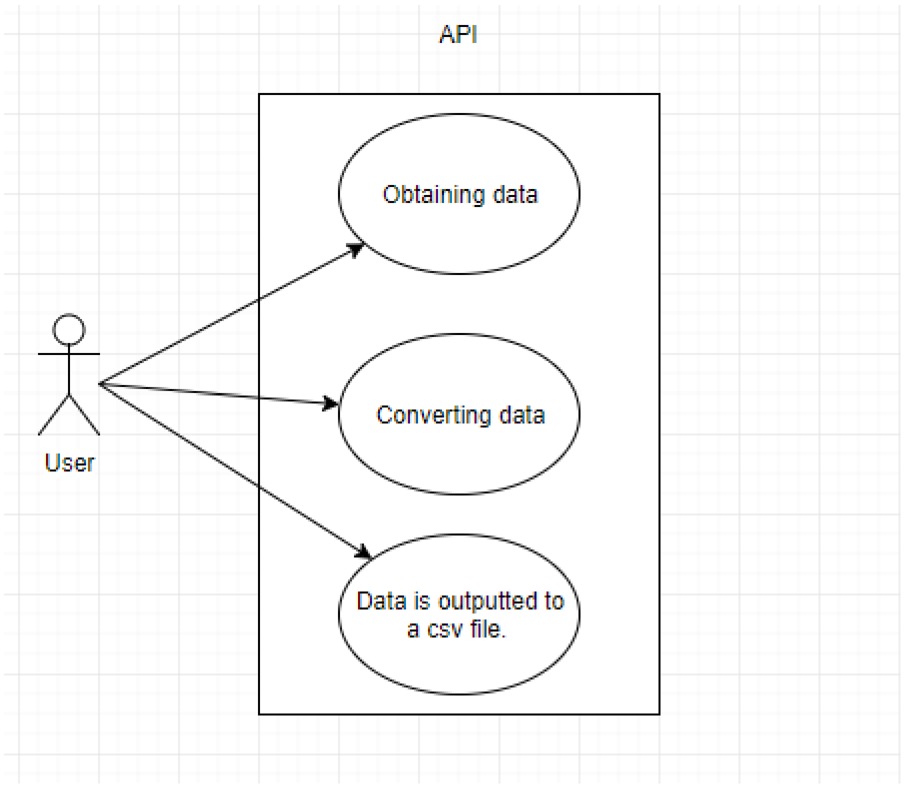
Scope

The scope of this use case is to obtain sufficient data from the API so that the user can analyse the data.

Description

This use case describes the process the user must take in order obtain the data from the API the data is to be automated rather than be downloaded manually in order to have sufficient data as the API is quite large and has data that is not necessary, the user cannot analyse the data without the correct data from the API.

Use Case Diagram



Flow Description

Precondition

The API must contain data.

Activation

This use case starts when a User pulls data from the API

Main flow

1. The use case begins when the user attempts to pull data from the API.
2. The User selects suitable data to be scraped from the API(See A1)
3. The User uses a Python request to retrieve data from the API(See E1)
4. The system will receive the data in JSON format.
5. The User uses Python to convert the JSON output.
6. The system receives the data in a csv file.
7. The data is now ready to be used.

Alternate flow

A1 : <Selecting and obtaining data>

1. The user looks at the data within the API.
2. The user selects the required/relevant data to be scraped.
3. The user creates a Python request to retrieve the data.
4. The system will now have the data in JSON format.
5. The use case continues at position 5 of the main flow

A2 : <Wrong data>

1. The data the user selected was not relevant to the project.
2. The user reselects sufficient data.
3. The use cases continues at position 3 of the main flow.

Exceptional flow

E1 : <Error obtaining data>

1. The system does not receive the data.
2. The Python request will give an error message.
3. The User will fix the error in the code for the Python request.
4. The use case continues at position 3 of the main flow

Termination

The data has been obtained and converted.

Post condition

The system goes into a wait state

2.1.8 Requirement 2 <Cleansing of the data>

2.1.8.1 Description & Priority

This use case describes the cleaning process of the data for the user. The user must clean the data correctly and efficiently so that the correct data can be used for analysis. This requirement has a high priority because if the data is not cleaned properly the dataset will be messy and data could be skewed.

2.1.8.2 Use Case

Sequence (2) <Data cleansing>

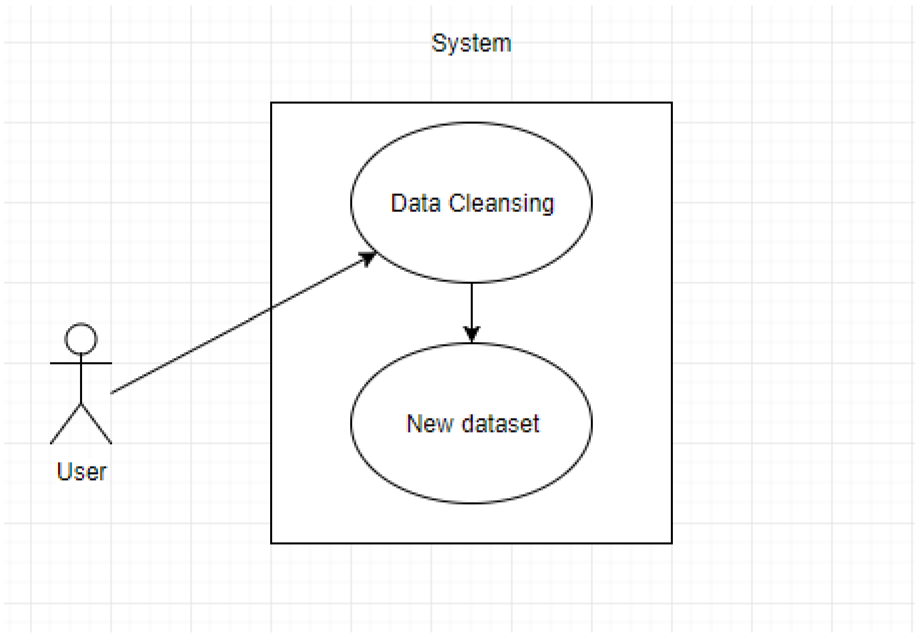
Scope

The scope of this use case is to clean the data that is retrieved from the API.

Description

This use case describes the cleaning of the data.

Use Case Diagram



Flow Description

Precondition

The data must be converted correctly into a csv file and ready for cleaning.

Activation

This use case starts when the User creates code to clean the data

Main flow

1. The use case begins when the user writes code in Rstudio to clean the data.
2. The User runs various lines of code to clean the dataset/data (See A1)
3. The system outputs and updates the data set, example (removing unwanted data) (See E1)
4. The user repeats position 1 of the main flow until the user is happy the data is cleaned and set up the way they want it.
5. The data is now cleaned and ready for data transformation.

Alternate flow

A1 : <Running code>

1. The user runs code to begin cleaning the data.
2. The system accepts the code and updates the dataset.
3. The user saves updated dataset.
4. The use case continues at position 3 of the main flow

Exceptional flow

E1 : <The new dataset>

1. The system failed to update the dataset
2. The User checks code for errors to identify why the data set didn't remove unwanted data or clean the data.
3. The User re runs the code.
4. The use case continues at position 3 of the main flow

Termination

The data has been prepared and is ready for analysis.

Post condition

The system goes into a wait state

2.1.9 Requirement 3 <Data Transformation>

2.1.9.1 Description & Priority

This use case describes the Data transformation process, this process involves taking the now clean dataset and begin generating better data for the data mining stage methods for example transforming to normality which prevents data from being skewed, term frequency and dimensionality frequency which is to reduce the number of variables, this use case also has a high priority as this process is essential for the data mining stage..

2.1.9.2 Use Case

Sequence (3) <Data Transformation>

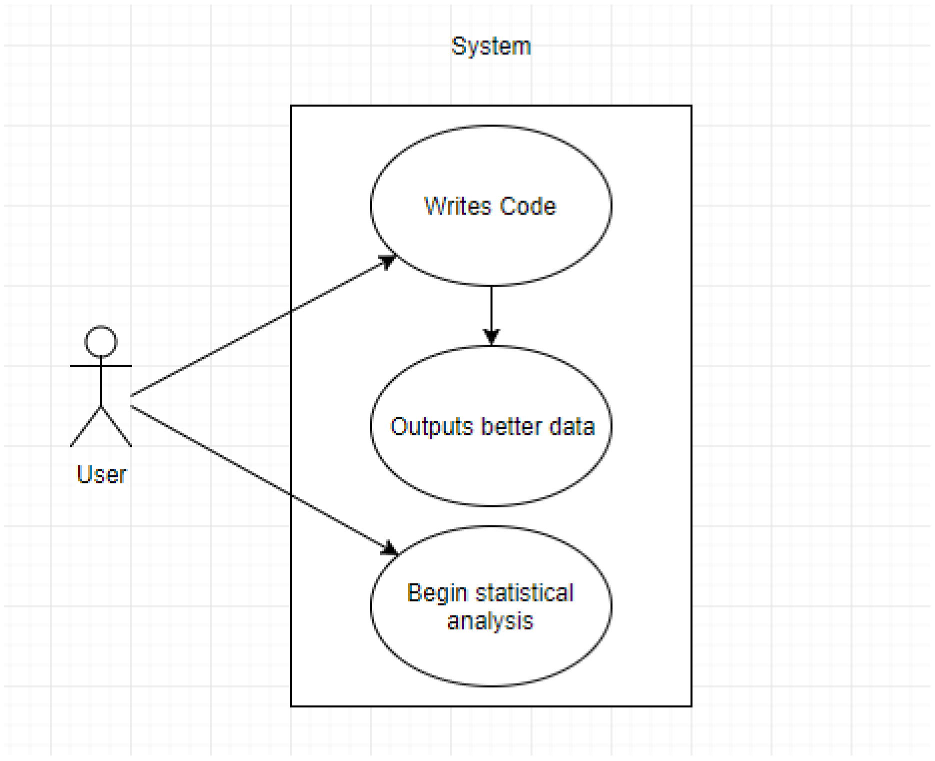
Scope

The scope of this use case is to transform the data and generate more efficient data for the data mining stage.

Description

This use case describes the process of data transformation.

Use Case Diagram



Flow Description

Precondition

The data must be cleaned.

Activation

This use case starts when the User begins to generate better data.

Main flow

1. The use case begins when the user writes code in Rstudio to generate more efficient data.
2. The User runs a number of different code samples to help produce better data.(See A1)
3. The system updates the dataset with the better data. (See E1)
4. The use case loops back and continues at position 1 on the main flow until the data is successfully transformed and ready.
5. The data is now transformed and will be more accurate when undergoing statistical analysis.

Alternate flow

A1 : <Running code>

1. The user runs code to start transforming the now cleaned data.
2. The system accepts the code and updates the dataset.
3. The system outputs better and more accurate information.
4. The use case continues at position 4 of the main flow.

Exceptional flow

E1 : <Error generating better data>

1. The system failed to update and generate better data for the dataset.
2. The User checks the sample of code for errors to identify why the system didn't generate better data.
3. The User fixes any errors and re runs the code.
4. The use case continues at position 4 of the main flow

Termination

The data has been transformed and better data has been generated.

Post condition

The system goes into a wait state

2.1.10 Requirement 4 <Data Mining>

2.1.10.1 Description & Priority

This use case describes the Data mining stage of the project which involves using machine learning and statistical algorithms to discover patterns and trends and then output new information.

2.1.10.2 Use Case

Sequence (4) <Data mining>

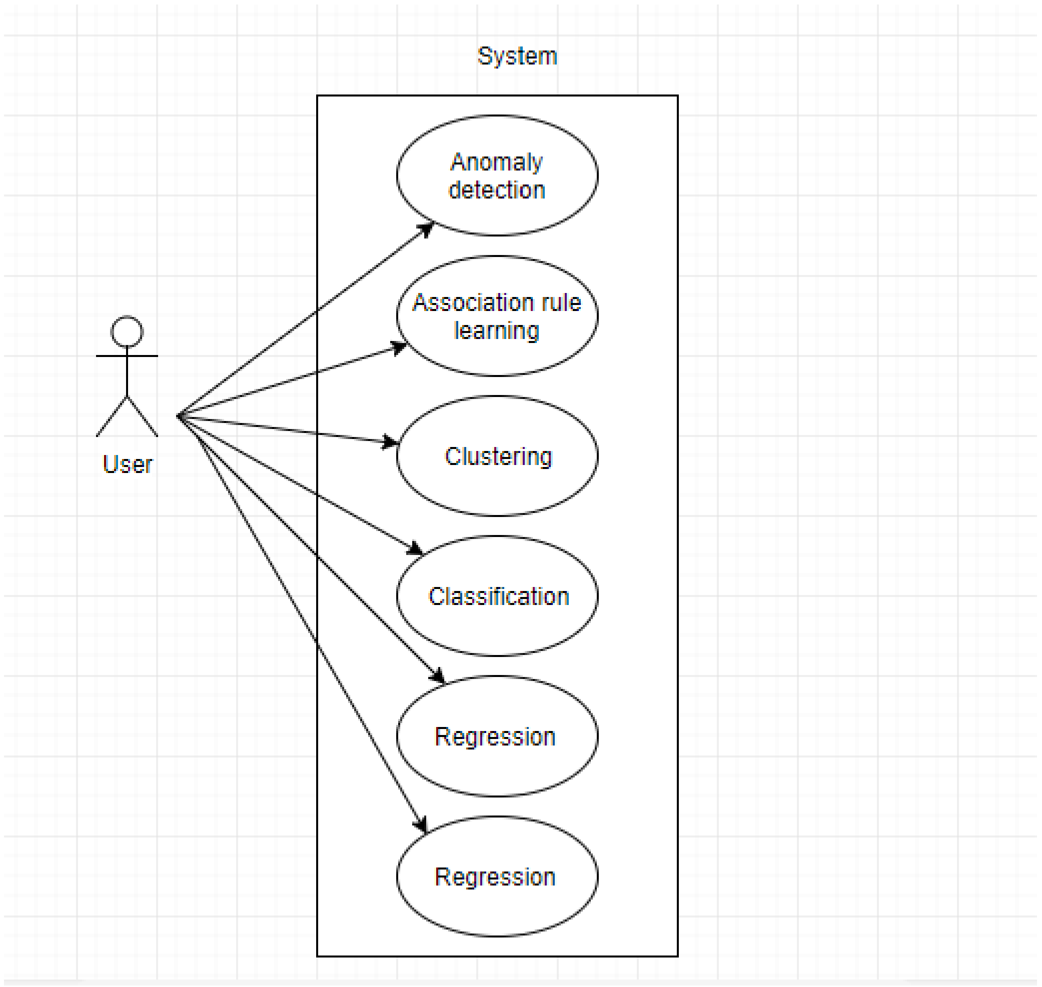
Scope

The scope of this use case is to begin the data mining process which involves 6 tasks.

Description

The use case describes the 6 tasks that must be completed in order to successfully complete the data mining process. The 6 tasks are (Anomaly detection, Association rule learning, clustering, classification, regression and summarization.

Use Case Diagram



Flow Description

Precondition

The data must be correctly set up and ready for the data mining process.

Activation

This use case starts when the User begins the data mining process of sorting through the data set to identify patterns and establish some relationships to solve problems.

Main flow

1. The use case begins when the user starts 1 of the 6 tasks mentioned above.
2. The user starts with Anomaly detection, by identifying unusual data records that might be interesting.(SeeA1)
3. The User runs statistical analysis algorithms to investigate further into this unusual data.
4. The system outputs the results of the code.
5. The User interprets the results and can see whether this data can help or it is a data error that requires further investigation.
6. The User decide what to do with this data.
7. The User repeats for each task and continues at position 1 of the main flow.
8. Data mining is complete and the user can see if this data can pick up on future trends in regards to the project.

Alternate flow

A1 : <another data mining task example Clustering>

1. The user runs statistical analysis algorithms to discover groups and structures in the data that are in some way similar.
2. The system outputs the results of the code.
3. The user can now see accurately what groups and structures in the data are in some way similar.
4. The use case continues at position 7 of the main flow

Exceptional flow

Termination

The data mining process is completed.

Post condition

The system goes into a wait state

2.1.11 Requirement 5 <Creating & exporting visualization>

2.1.11.1 Description & Priority

This use case describes the process of creating and exporting visualization for the project, which involves using R to code and display graphs to output the results of my project and to use tableau to output results, this requirement has a high priority because it is easier to understand my results with aid of visualization. Most people can understand something easier if they can visualize it.

2.1.11.2 Use Case

Sequence (5) <Creating & exporting visualization>

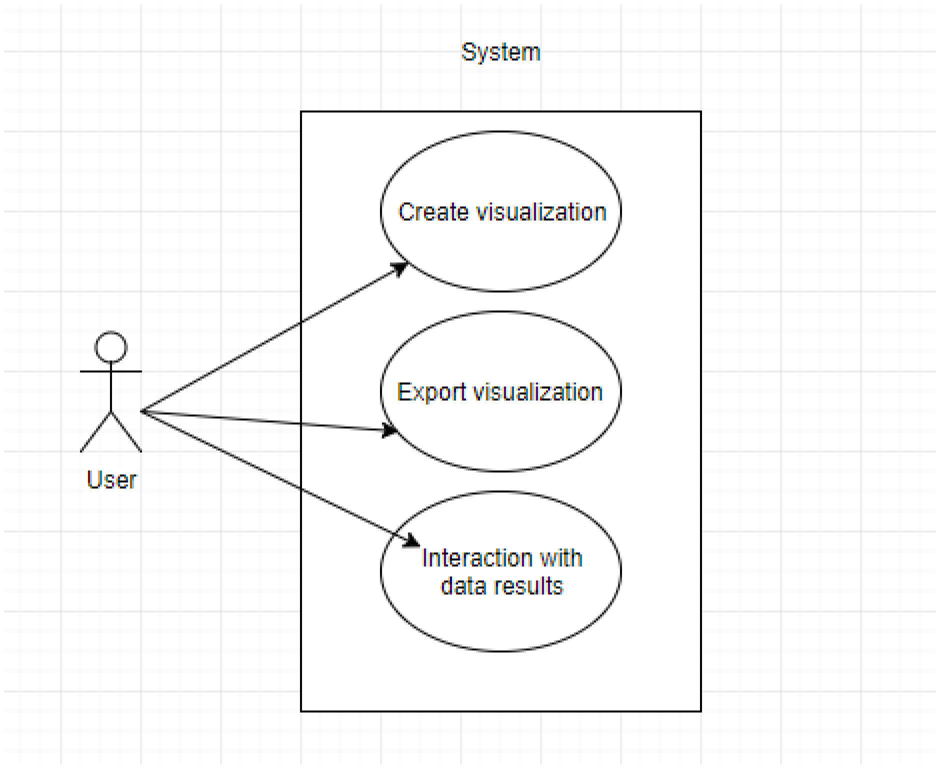
Scope

The scope of this use case is to list the steps involved in the creation of visualization and how to export it.

Description

The use case describes the steps that must be taken to create and export visualization of the data results developed by the user.

Use Case Diagram



Flow Description

Precondition

There must be data results in order to begin creating and exporting visualization.

Activation

This use case starts when the User starts to import results into tableau.

Main flow

1. The use case begins when the user starts representing the results in the form of a charts, diagrams, and pictures.
2. The User then imports results into tableau.
3. The User can detect patterns, trends and correlations that might go undetected in text-based data, as these may be exposed and recognized easier with data visualization software.(See A1)
4. This allows the end user to interpret the results easier and getting a better understanding of the results.(See E1)

Alternate flow

A1 : <Detecting patterns>

1. The User detects a pattern/trend in the data results through data visualization.

2. The User does more statistical analysis algorithms on these patterns.
3. The User interprets the results.
4. The use case continues at position 1 of the main flow

Exceptional flow

E1 : <Error outputting results>

1. The user could not interpret the results easily.
2. The results outputted incorrect information making the graphs messy and hard to understand for the end user.
3. The use case continues at position 1 of the main flow.

Termination

The data is visualized and therefore helps people understand the data much easier and there is a link to allow end users to view and interact with the graphs and the results.

Post condition

The system goes into a wait state

2.1.12 Data requirements

- The data should be automated from the Betfair API.
- The data must be converted to a csv file in order to clean the data and for data transformation to take place.
- The data should be exported using visualization.
- The data should display clearly the results in graphs and display patterns and trends within the data.
- The end user should be able to interact and interpret with the data results.

2.2 Non-Functional Requirements

2.2.1 Availability requirement

All of the results gathered as well as the system and the code used will be public through GitHub and made available for anyone so people can view and interpret the data at any time.

2.2.2 Performance requirement

The system will perform to a minimum set verge measured in response time. It is very important that the system or any system accommodates to certain performance requirements. Performance is a problem that required addressing as the system relies heavily on Betfair's API as a source of data. As a result it is critical that the system responds to a User within a reasonable amount of time.

2.2.3 Recovery requirement

All of the data gathered/datasets and any code used will be backed up locally also with cloud based storage providers like GitHub mentioned above and Google Drive. The latest version will always be stored locally and then will be pushed to GitHub after vital or important changes have been made. The coding will be done in Atom which is a code editor created by GitHub as well as Rstudio to run statistical analysis.

2.2.4 Reliability requirement

The system must meet all of the functional requirements. The scripts that are developed within the system will run perfectly as long as the machine has the necessary dependencies installed. The results must be allow users to interact with the data. A set of automated tests should be used to verify that the system can continue to meet the requirement as it grows.

2.2.5 Environmental requirement

The system will run across most platforms main ones being Windows, mac, and linux. The machine will be required to have Python 2.7 and Rstudio installed. The R scripts and Python scripts will contain dependencies which will also be required to be installed also.

2.2.6 Extendibility requirement

The system will be extendible if the data is increased and by applying more complex algorithms to find patterns and trends with the ratings compared to winners of races.

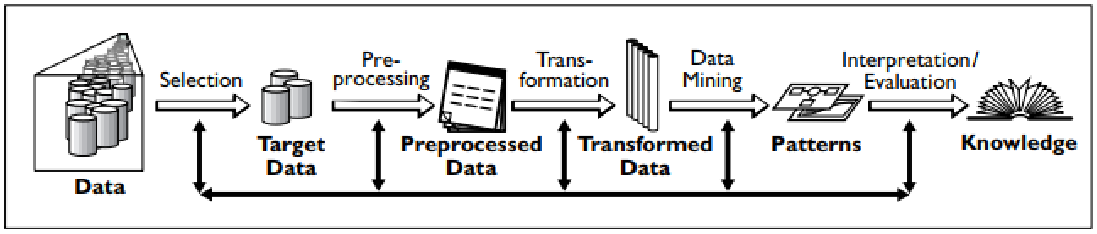
2.2.7 Reusability requirement

Reusability refers to that the system components can be re used to a similar composition to reduce time and lower code complexity. The system will be reusable meaning that the same algorithms and process can be implemented and re used on different parts of Betfairs API so for example different sports, so the algorithms and data processing can be implemented easily to be used on say football instead of horse racing..

2.3 Design and Architecture

2.3.1 Design Approach

2.3.2 KDD (Knowledge Discovery in Databases)



I will approach this project by using the KDD (Knowledge Discovery in Databases) approach which has some key steps in order to build a successful data analytics project. Below are the steps for this approach.

Selection- This involves finding the information and datasets that will be best for my project in terms of what I want my end result to be.

Pre-processing- This will be me cleaning the data so that I can get rid of all the unwanted data that I don’t need for my project.

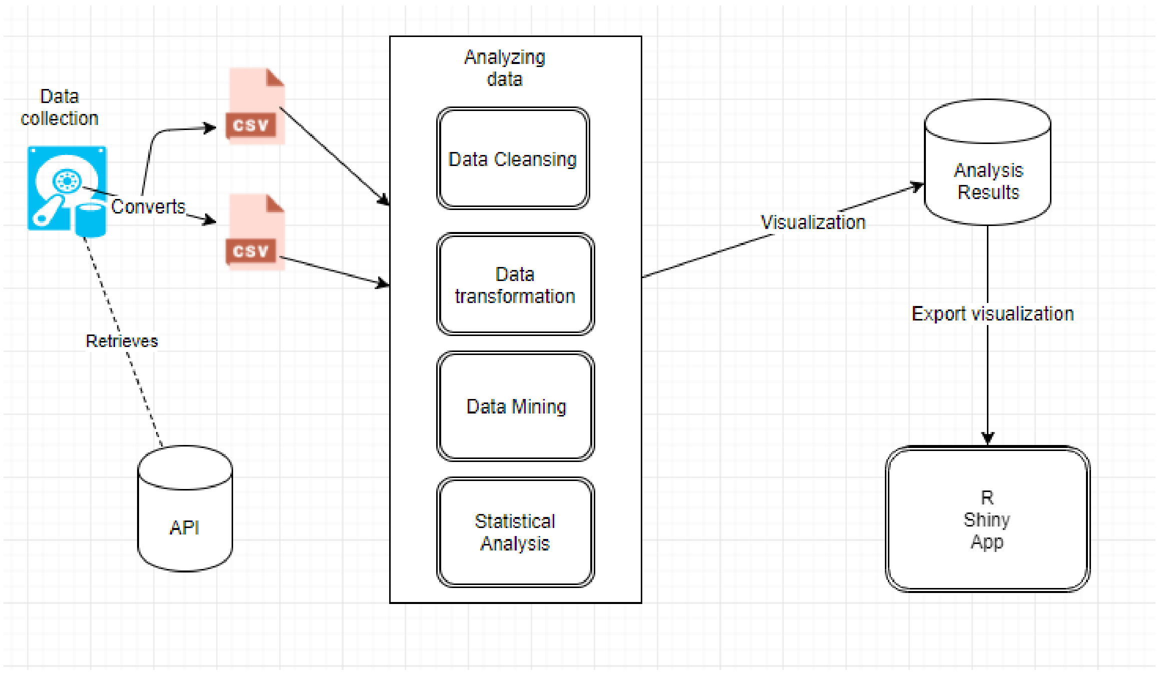
Transformation- This will involve using the clean dataset and generating better data, for the data mining stage methods which include dimension reduction, such as feature selection and extraction, record sampling, and attribute transformation such as discretization of numerical attributes and functional transformation.

Data Mining- This is where I will use the descriptive data mining technique which included unsupervised, and visualization aspects of data mining.

Interpretation- Here I will interpret the knowledge gained from the records I have acquired.

2.3.3 Architecture Design

2.3.3.1 System Architecture

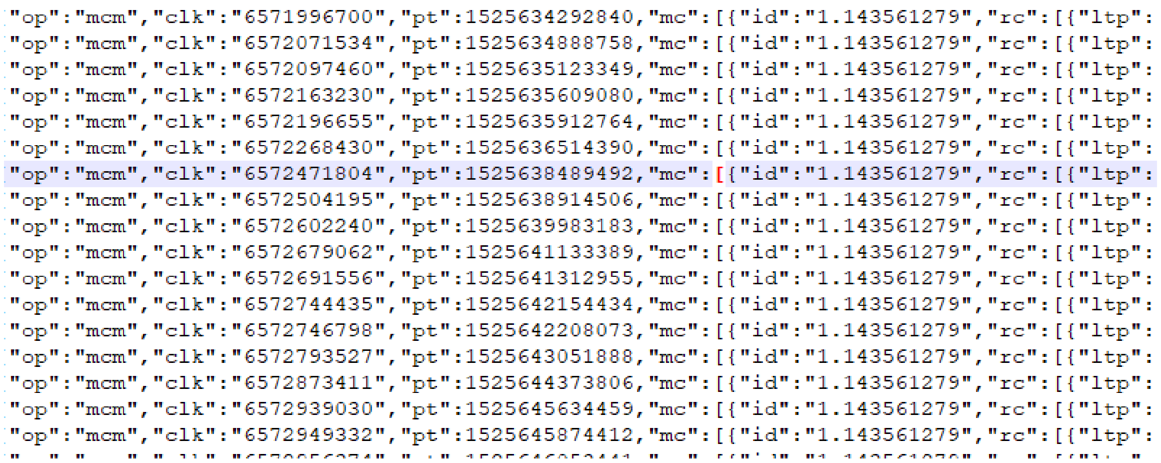


The figure above details a high level view of the system architecture. On the left shows how the extraction of data takes place and then the data is loaded and shows a step by step on how KDD approach is done on the data. On the right of the figure it shows then the data being visualised and then exported to R Shiny so the data can be seen visually and allow users to interact with it.

2.4 Implementation

2.4.1 Betfair API Crawler

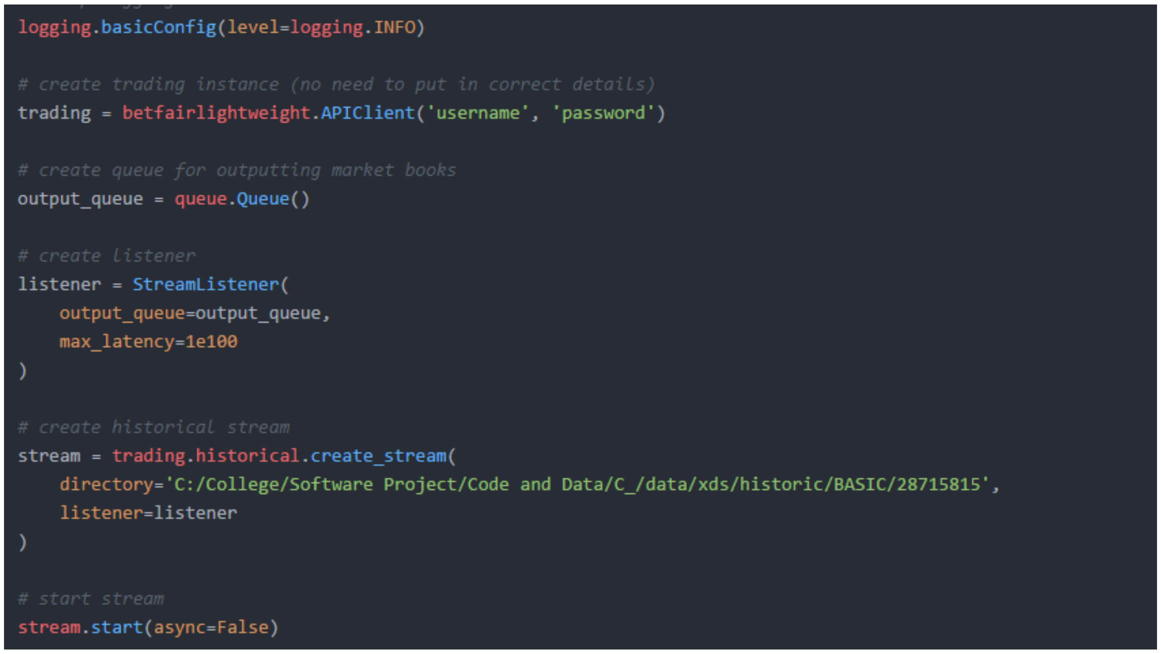
The Betfair crawler is part of the scraper and is written in Python. There was a huge amount of challenges encountered throughout the implementation phase. Some of these challenges were based around understanding the API as it is so large, so just trying to understand the appropriate API endpoints to use was very difficult and developing and writing code which could gather in the data in an efficient matter was also quite challenging. The first implementation was efficient but I found it very difficult and time consuming. Betfair recently changed their API so that there that is in JSON format, I wasn't very strong with JSON so it took me a lot of time to try understand the data in that format. Also because the data was in JSON format it was hard to read and look at it as if it were a csv, so by first looking at the data it looks just like a lot of number thrown together.



```
"op":"mcm","clk":"6571996700","pt":1525634292840,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572071534","pt":1525634888758,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572097460","pt":1525635123349,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572163230","pt":1525635609080,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572196655","pt":1525635912764,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572268430","pt":1525636514390,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572471804","pt":1525638489492,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572504195","pt":1525638914506,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572602240","pt":1525639983183,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572679062","pt":1525641133389,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572691556","pt":1525641312955,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572744435","pt":1525642154434,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572746798","pt":1525642208073,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572793527","pt":1525643051888,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572873411","pt":1525644373806,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572939030","pt":1525645634459,"mc":[{"id":"1.143561279","rc":[{"ltp":
"op":"mcm","clk":"6572949332","pt":1525645874412,"mc":[{"id":"1.143561279","rc":[{"ltp":
```

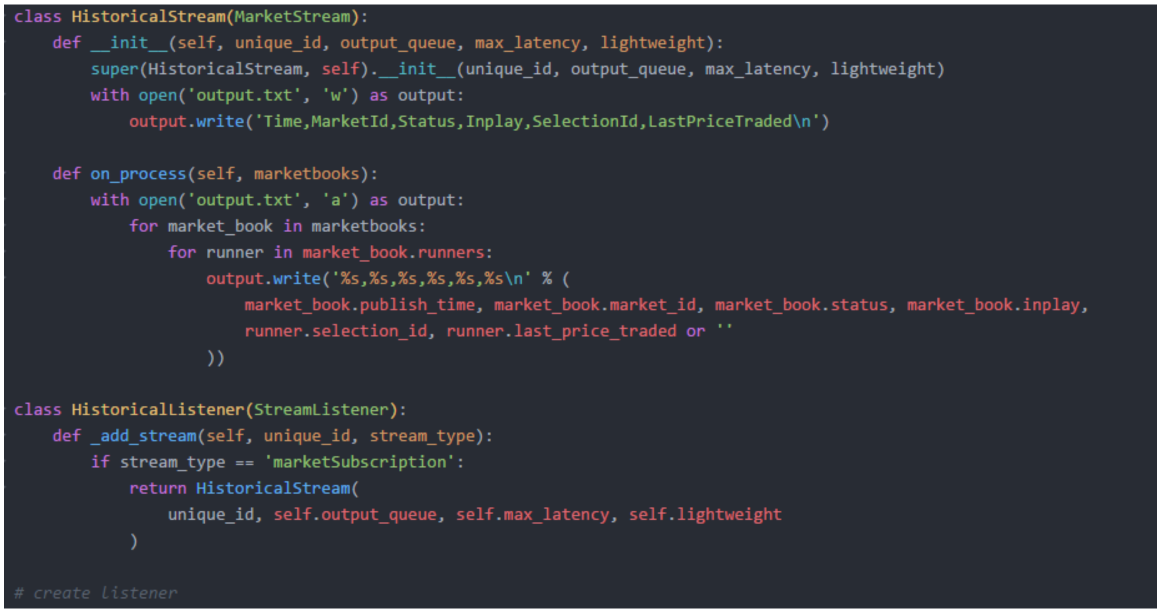
When I first extracted the data this was my output, I had no idea what any of this meant because all of the data columns are shortened down and are abbreviations. After ding research I fortunately found someone who created a word document of what all the abbreviations stood for which helped me significantly understand the data better. I spent a lot of time understanding the data and trying to get the grasps of the API. The next phase was to create code in Python to convert this data to a csv file so that I could load it into r and do analysis.

The below figures shows two Python scripts to convert one of the JSON files to a .txt file.



```python
logging.basicConfig(level=logging.INFO)

# create trading instance (no need to put in correct details)
trading = betfairlightweight.APIClient('username', 'password')

# create queue for outputting market books
output_queue = queue.Queue()

# create listener
listener = StreamListener(
    output_queue=output_queue,
    max_latency=1e100
)

# create historical stream
stream = trading.historical.create_stream(
    directory='C:/College/Software Project/Code and Data/C_/data/xds/historic/BASIC/28715815',
    listener=listener
)

# start stream
stream.start(async=False)
```

This code above is quite straight forward, I used the betfairlightweight package which helps access the data that you purchased on the https://historicdata.betfair.com/#/mydata so the data I purchased is stored In that directory on Betfairs website. Once you input username and password details to the code and give the directory of the file you are converting. The code will work.



```python
class HistoricalStream(MarketStream):
    def __init__(self, unique_id, output_queue, max_latency, lightweight):
        super(HistoricalStream, self).__init__(unique_id, output_queue, max_latency, lightweight)
        with open('output.txt', 'w') as output:
            output.write('Time,MarketId,Status,Inplay,SelectionId,LastPriceTraded\n')

    def on_process(self, marketbooks):
        with open('output.txt', 'a') as output:
            for market_book in marketbooks:
                for runner in market_book.runners:
                    output.write('%s,%s,%s,%s,%s,%s\n' % (
                        market_book.publish_time, market_book.market_id, market_book.status, market_book.inplay,
                        runner.selection_id, runner.last_price_traded or ''
                    ))

class HistoricalListener(StreamListener):
    def _add_stream(self, unique_id, stream_type):
        if stream_type == 'marketSubscription':
            return HistoricalStream(
                unique_id, self.output_queue, self.max_latency, self.lightweight
            )

# create listener
```

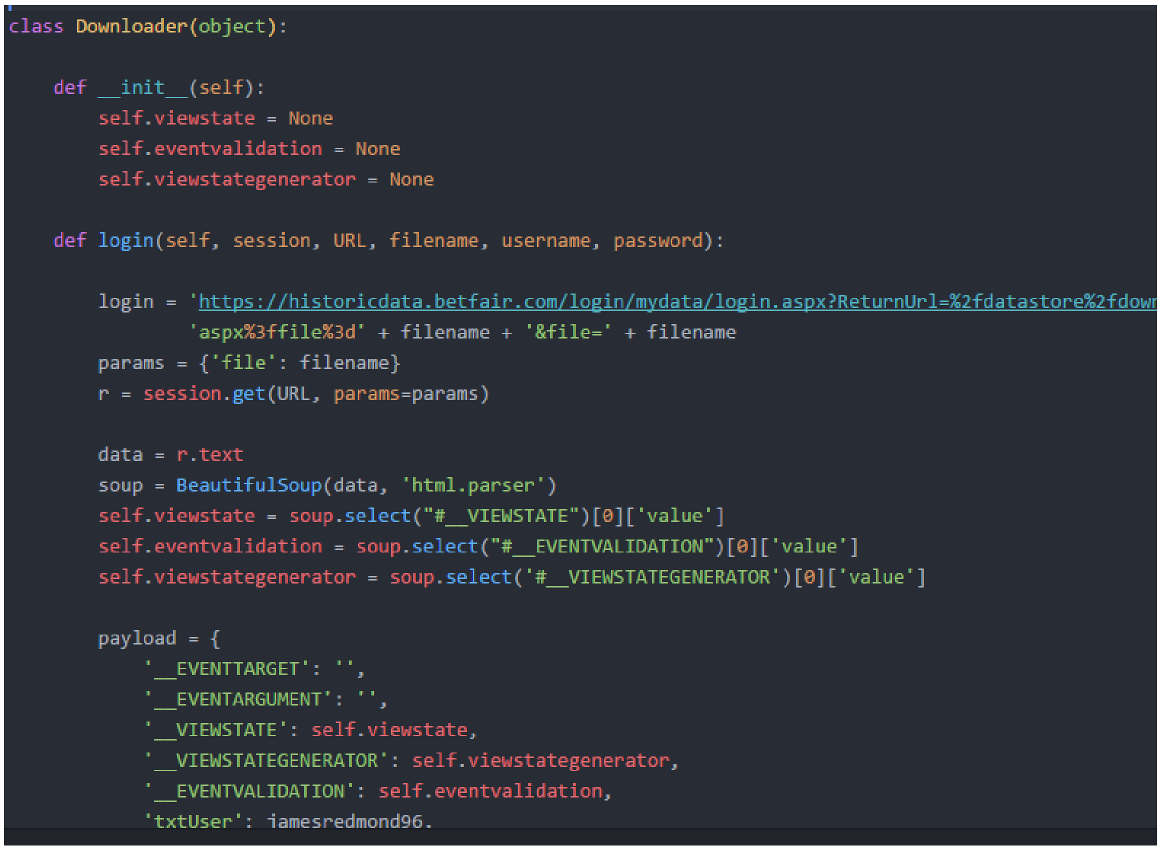
This figure shows the conversion to a .txt file, so basically what this Python script does is output the JSON file to a txt file and then from there I convert the txt file to a csv file and then clean the data. How it works is it uses the lightweight function to read in the data by matching the columns, so a column example s LastPriceTraded so this will pull the last traded price of a horse at a given time. This script can be customized and has to be customized for every file depending on what data I want to bring in. So if I only want to bring in 5 columns I can keep the script the same and just have to change the file directory and the output.txt file has to be given a new name for every bit of data that gets pulled in from the API otherwise the data will just override and just replace the txt file with new data before converting it.

2.4.2 Betfair API Automation

2.4.2.1 Download automation

The Betfair API crawler that I used to bring in the data was slow and manual. This meaning that it took a long time to obtain data from the API, which isn't ideal. So I made another Python Script that automated the extraction of the data. The idea for this was to automate the downloading process of the data as well as the unzipping process, meaning that I could get access to a bunch of JSON files much faster than having to manually download and unzip every file also to automate extraction of the data.

The below figure shows a snippet of code that automates the downloading process.

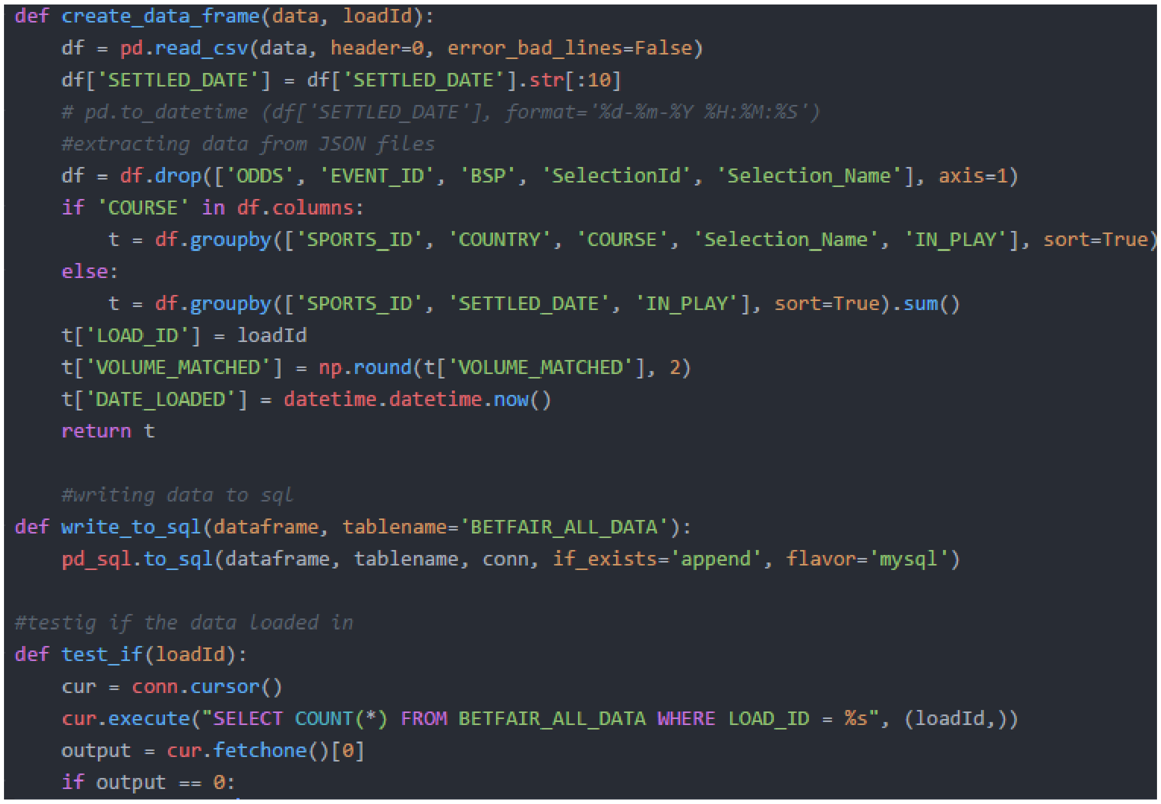


```
class Downloader(object):

    def __init__(self):
        self.viewstate = None
        self.eventvalidation = None
        self.viewstategenerator = None

    def login(self, session, URL, filename, username, password):

        login = 'https://historicdata.betfair.com/login/mydata/login.aspx?ReturnUrl=%2fdatastore%2fdown
                'aspx%3ffile%3d' + filename + '&file=' + filename
        params = {'file': filename}
        r = session.get(URL, params=params)

        data = r.text
        soup = BeautifulSoup(data, 'html.parser')
        self.viewstate = soup.select("#__VIEWSTATE")[0]['value']
        self.eventvalidation = soup.select("#__EVENTVALIDATION")[0]['value']
        self.viewstategenerator = soup.select('#__VIEWSTATEGENERATOR')[0]['value']

        payload = {
            '__EVENTTARGET': '',
            '__EVENTARGUMENT': '',
            '__VIEWSTATE': self.viewstate,
            '__VIEWSTATEGENERATOR': self.viewstategenerator,
            '__EVENTVALIDATION': self.eventvalidation,
            'txtUser': jamesredmond96,
```

The above code shows the Downloader object being created, first what happens is it gets the users login so that it can access the users data that has been purchased by the user, it finds the files through the url that's given above it searches through the url looking for files to download, the data is then requested and parsed using the Beautiful Soup library. Then it checks the view state of the file if it's valid and then generates the download of the file if found successfully. This is of course just a snippet of the code and what the rest of the code does is it makes a POST request to the login data, upon successful login the session cookie is stored in memory and the crawler can be requesting a User's connection, so the crawler can begin accessing the files that the user had downloaded. The script then gets the location of the file then it downloads the file and stores the file or the downloaded files in a directory.

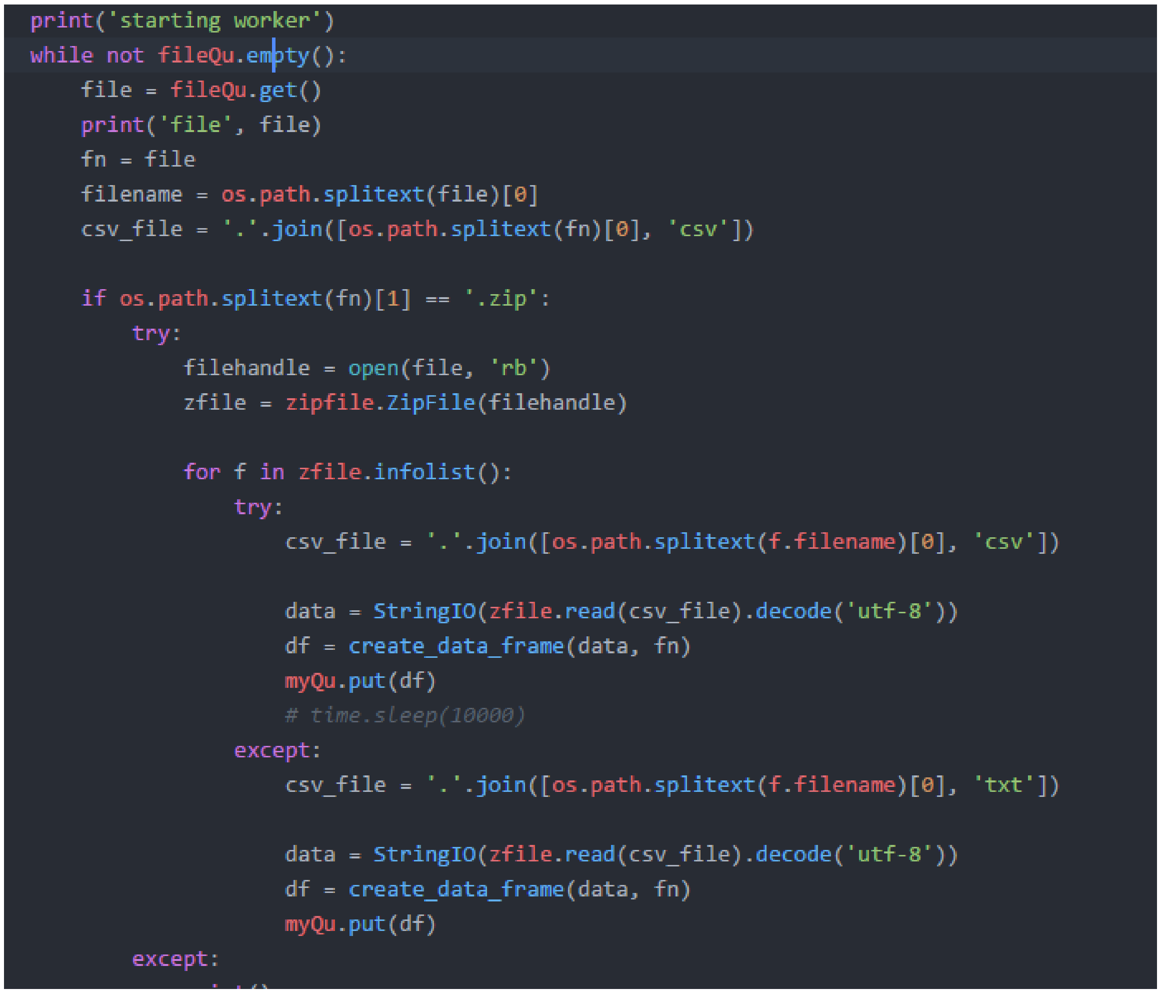
2.4.2.2 Unzip and extraction automation

Similar to the automation process above a Python script was created to speed up the process of obtaining data from the API, what the Python script does is it essentially accesses the downloaded data and it extracts the data into a table in a SQL database. So this helps speed up obtaining data hugely because it cuts out the long slow process of unzipping every file and the manually running code on each file once its unzipped, this code automates that process takes the file and stores into a SQL database from then the User can load the tables into csv files and then load them into R studio and begin analysis.



```
def create_data_frame(data, loadId):
    df = pd.read_csv(data, header=0, error_bad_lines=False)
    df['SETTLED_DATE'] = df['SETTLED_DATE'].str[:10]
    # pd.to_datetime (df['SETTLED_DATE'], format='%d-%m-%Y %H:%M:%S')
    #extracting data from JSON files
    df = df.drop(['ODDS', 'EVENT_ID', 'BSP', 'SelectionId', 'Selection_Name'], axis=1)
    if 'COURSE' in df.columns:
        t = df.groupby(['SPORTS_ID', 'COUNTRY', 'COURSE', 'Selection_Name', 'IN_PLAY'], sort=True)
    else:
        t = df.groupby(['SPORTS_ID', 'SETTLED_DATE', 'IN_PLAY'], sort=True).sum()
    t['LOAD_ID'] = loadId
    t['VOLUME_MATCHED'] = np.round(t['VOLUME_MATCHED'], 2)
    t['DATE_LOADED'] = datetime.datetime.now()
    return t

    #writing data to sql
def write_to_sql(dataframe, tablename='BETFAIR_ALL_DATA'):
    pd_sql.to_sql(dataframe, tablename, conn, if_exists='append', flavor='mysql')

#testig if the data loaded in
def test_if(loadId):
    cur = conn.cursor()
    cur.execute("SELECT COUNT(*) FROM BETFAIR_ALL_DATA WHERE LOAD_ID = %s", (loadId,))
    output = cur.fetchone()[0]
    if output == 0:
```

The above figure, shows the process of automation. A data frame is created, and it loads the downloaded data by the loadId of the downloaded file, then the User can create his data frame so whatever columns they want to pull from the files, then it matches the columns and writes the file to a table in SQL. A test is then run to check that the data loaded in the new table.



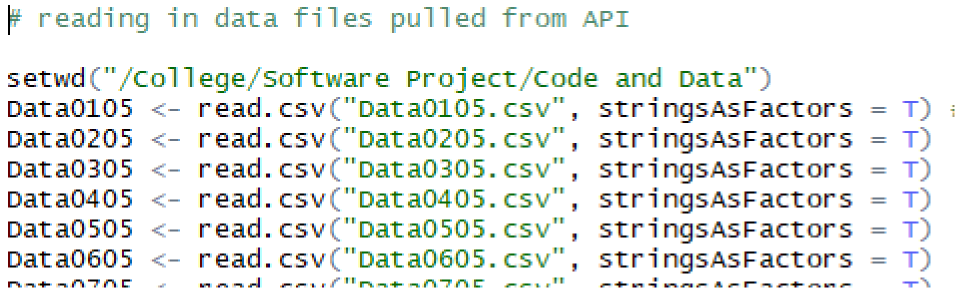
```python
print('starting worker')
while not fileQu.empty():
    file = fileQu.get()
    print('file', file)
    fn = file
    filename = os.path.splitext(file)[0]
    csv_file = '.'.join([os.path.splitext(fn)[0], 'csv'])

    if os.path.splitext(fn)[1] == '.zip':
        try:
            filehandle = open(file, 'rb')
            zfile = zipfile.ZipFile(filehandle)

            for f in zfile.infolist():
                try:
                    csv_file = '.'.join([os.path.splitext(f.filename)[0], 'csv'])

                    data = StringIO(zfile.read(csv_file).decode('utf-8'))
                    df = create_data_frame(data, fn)
                    myQu.put(df)
                    # time.sleep(10000)
                except:
                    csv_file = '.'.join([os.path.splitext(f.filename)[0], 'txt'])

                    data = StringIO(zfile.read(csv_file).decode('utf-8'))
                    df = create_data_frame(data, fn)
                    myQu.put(df)
        except:
```

Then the Python script converts the file to a csv. It does this by reading in the new table and printing it out as a csv file, user has to put in a directory on where they want the file to save. This can then be loaded into R studio and used for analysis.

The automation process helped speed up the process of obtaining data massively not just data but it helped signify the correct data and sped up the process of receiving the data. Once automation was complete the User is able to purchase data from the API and run the automation code to automatically download the data and extract the data and then convert the data to a csv while also being saved in a database for backup, once the user is happy with the amount of the data the data mining stage can begin in R.

2.4.3 Analysis in R

The analysis was performed following the KDD approach which was described earlier in the document. First step was cleaning the data so analysis could be performed. Then exploration of the data, once these two steps where done I could begin to implement my predictive models and algorithms to come up with my solution for the project. Then testing each algorithms accuracy to see how well they worked for the problem/solution at hand. The last step was to just show some visualization of the results of the algorithms comparing each one to each other based of accuracy levels.



```
# reading in data files pulled from API

setwd("/College/Software Project/Code and Data")
Data0105 <- read.csv("Data0105.csv", stringsAsFactors = T)
Data0205 <- read.csv("Data0205.csv", stringsAsFactors = T)
Data0305 <- read.csv("Data0305.csv", stringsAsFactors = T)
Data0405 <- read.csv("Data0405.csv", stringsAsFactors = T)
Data0505 <- read.csv("Data0505.csv", stringsAsFactors = T)
Data0605 <- read.csv("Data0605.csv", stringsAsFactors = T)
```

The above figure is just a code snippet of creating a data frame that holds data from a given date, these files were automated by the automation code above, after a sufficient amount of files were pulled in I combined all the files into one main data set so I could run my algorithms on the data. I then began exploring the dataset running multiple lines of code in the process. I done some random analysis such as looking whether or not a horse won or lost in a race I also ran code to check that the data is distributed correctly.

In total there was only around 100 lines of code for data exploration and data cleansing, this is largely due to the APIs data being cleaned already and ordered so that once the data was obtained not too much cleaning had to be done. After data exploration and data cleaning, I began to implement my algorithms. The algorithms I used were decision trees, KNN, and SVM below is a brief explanation of each. Decision Trees – A decision tree is a structure that includes a root node, branches and leaf nodes. Each subjective node denotes a test on an attribute, every branch stands for the outcome of a test, and then each leaf node contains a class label. The node at the very top of the tree is the root node.

KNN (k - Nearest Neighbour) –

K nearest neighbour is a quite simple algorithm that stores all available cases and classifies new cases based on a similarity measure. KNN has been around since the 70s and has been used in statistical estimation and pattern recognition. So to simplify how the algorithm works is a case is classified by a majority vote of all its neighbours, the case then being assigned to the class most common amongst its K nearest neighbour measured by a distance function. An example is if K=1, then the case is simply assigned to the class of its nearest neighbour.

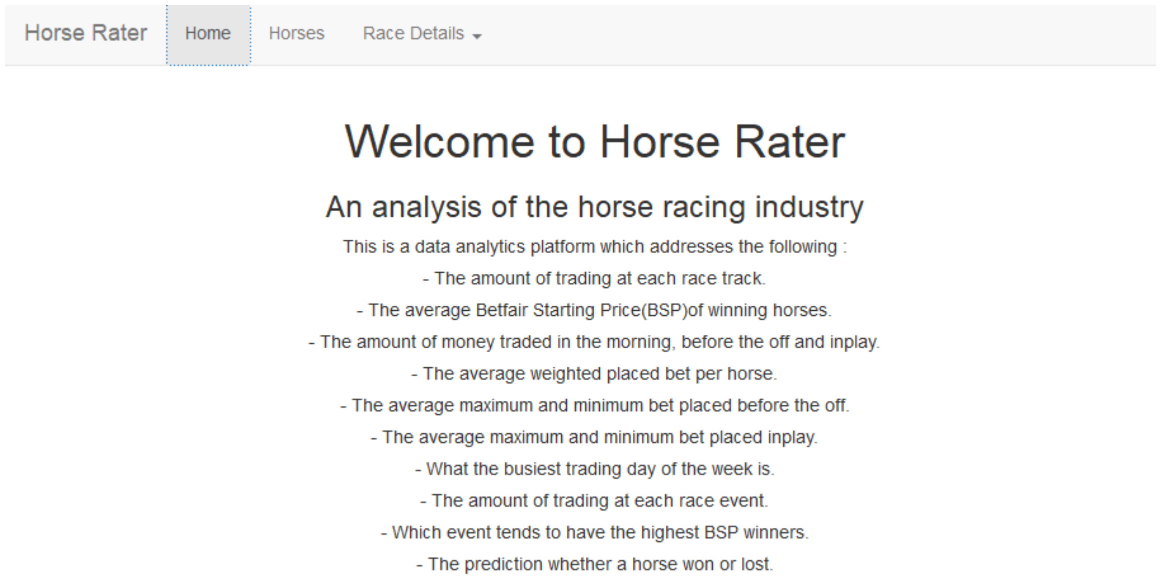
SVM (Support Vector Machine) –

This is the last algorithm I used to for my predictive model. A Support Vector Machine is a selective classifier formally defined by a separating hyperplane, this means that given labelled training data, the algorithm outputs an optimal hyperplane which categorizes new examples. I go into detail about how I used the above 3 algorithms for my predictive model in the evaluation section as they also give me my results.

2.4.4 Creating R Shiny application

I created the R shiny application for my project in r Studio, The layout of my application is two classes one that creates and holds the user interface, called ui.R and a server.R class which contains the code for displaying my interactive graphs about my data. I made a simple enough application with a home page, one main page and then a navigation panel to two sub-pages. The home page contains text about what the data is and the rest of the pages all contain interactive graphs.

2.5 Graphical User Interface (GUI) Layout



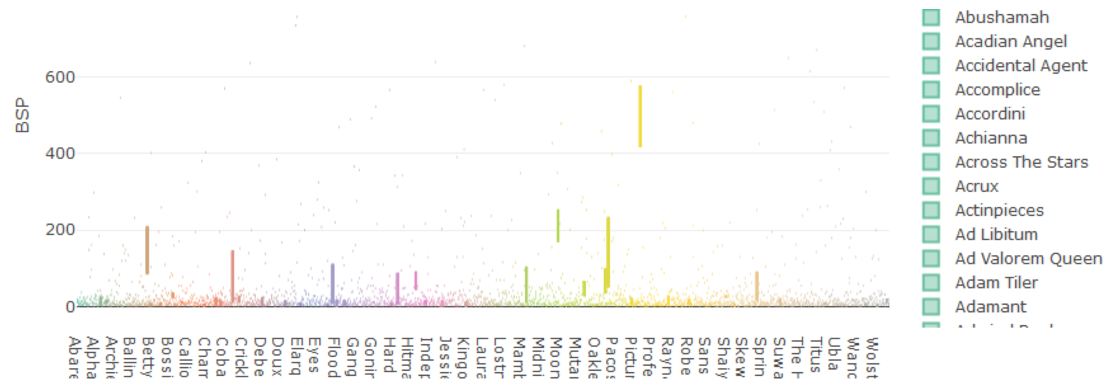
Welcome to Horse Rater

An analysis of the horse racing industry

This is a data analytics platform which addresses the following :

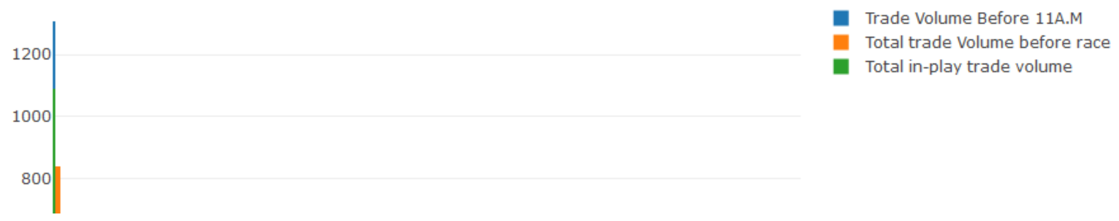
- The amount of trading at each race track.
- The average Betfair Starting Price(BSP)of winning horses.
- The amount of money traded in the morning, before the off and inplay.
- The average weighted placed bet per horse.
- The average maximum and minimum bet placed before the off.
- The average maximum and minimum bet placed inplay.
- What the busiest trading day of the week is.
- The amount of trading at each race event.
- Which event tends to have the highest BSP winners.
- The prediction whether a horse won or lost.

This is my r Shiny application. This is the home page contains information about the analysis and the data and some areas in which the platform addresses. As you can at the top of the page there is links to other pages which contain interactive graphs. Also the application is responsive.

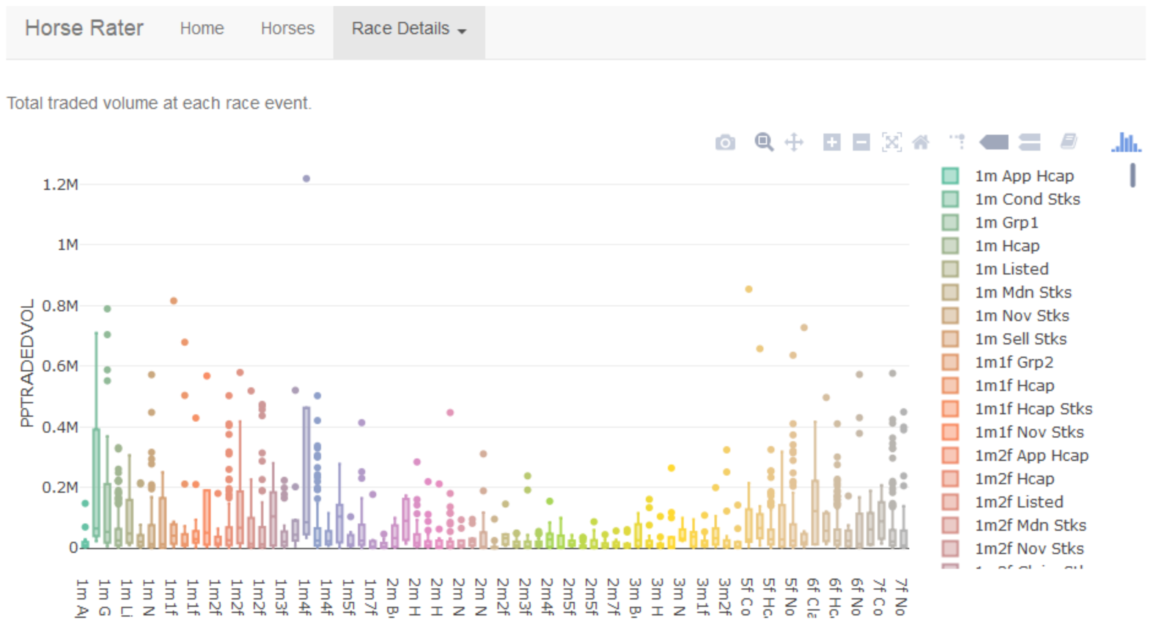


Total traded volume in the morning before the race and in-play.

Displaying horses who won and lost.

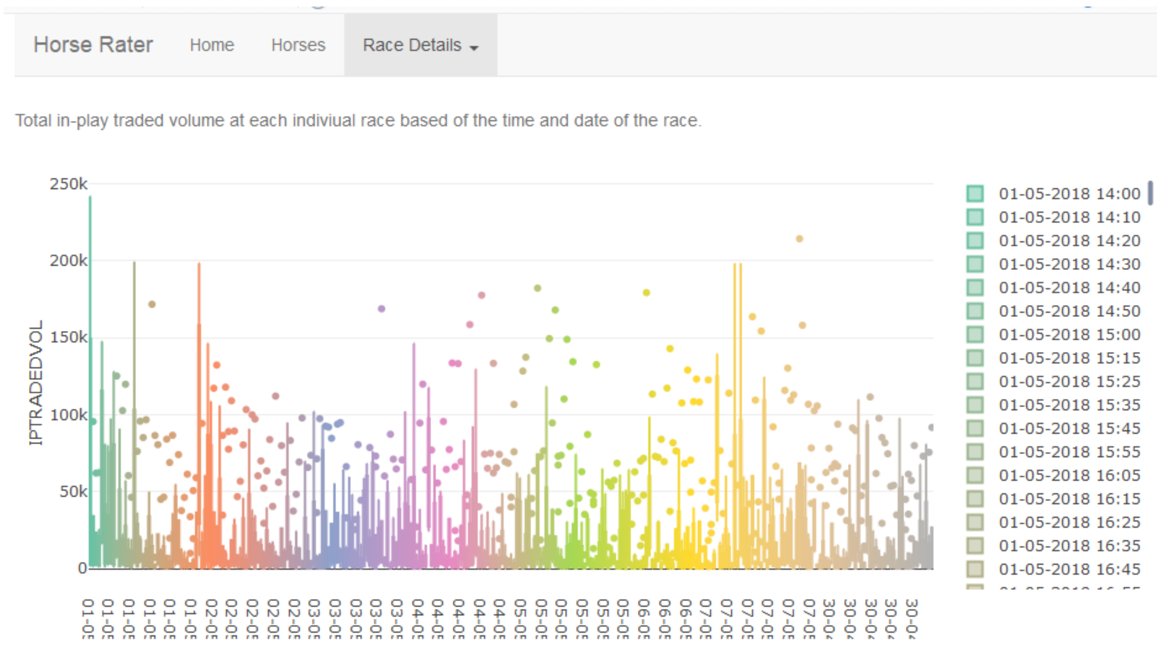


This figure above shows the main page called horses. This page contains a few interactive graphs that give an interesting insight into the data and the analysis.



Total traded volume at each race event.

The figure above, shows the Race details page, which then produces an interactive box chart, a user can interact with this data by looking at what type of race makes the most money. So meaning which race type has the highest trading volume of money.



The above figure shows the Race details page again only it's a different page selected within the drop down menu. This page contains an interactive graph that displays the total amount of in play trade volume in conjunction with the time of all races. So visually I can see a pattern already and when you interact with the data you learn there is a substantial difference in the amount of money being traded in-play in the earlier races, so at each race event the first and second race tend to have a very high amount of in-play trading volume compared to the rest of the races and as the races go on the lower it goes.

2.6 Testing

In most industries today, Software testing including automated and manual quality assurance takes up a considerable portion of the development lifecycle. Testing can be vital to ensure a system performs and behaves as it is supposed to.

2.6.1 Manual Test Cases

2.6.1.1 Test Scenario 1

Test Scenario –

Check Downloading of data functionality.

Test Case –

Check the response on entering valid user name and password.

Test Steps -

- Load python script called scraper.py
- Input username to the required fields.
- Input password to the required fields.
- Input correct url from user account within Betfair’s API
- Launch code in the command line to run script

Test Data -

User Name: mybetfairaccount

Password: My betfairpassword

url : “http://data.betfair.com/datastore/downloadfile.aspx?file =' + filename”

Expected Result –

Download file to directory

2.6.1.2 Test Scenario 2

Test Scenario –

Check Data Extraction functionality

Test Case –

Check that the data outputs to a csv.

Test Steps –

- Load python script called extraction.py
- Data frame must match column names with names in JSON file
- Sql code must have correct sql schema name.

- LOAD_ID must be a valid id

Test Data –

LOAD_ID must match id of the downloaded files

LOAD_ID = FILE_ID

Expected Result –

Data file should be loaded in database.

2.6.1.3 Test Scenario 3

Test Scenario –

Check KNN algorithm logic

Test Case –

Check that algorithm outputs results.

Test Steps –

- Load r script called ProjectAnalysis.R
- Run r code for knn Algorithm
- Plot knn algorithm

Test Data –

allData must be the dataset used

Expecting Result –

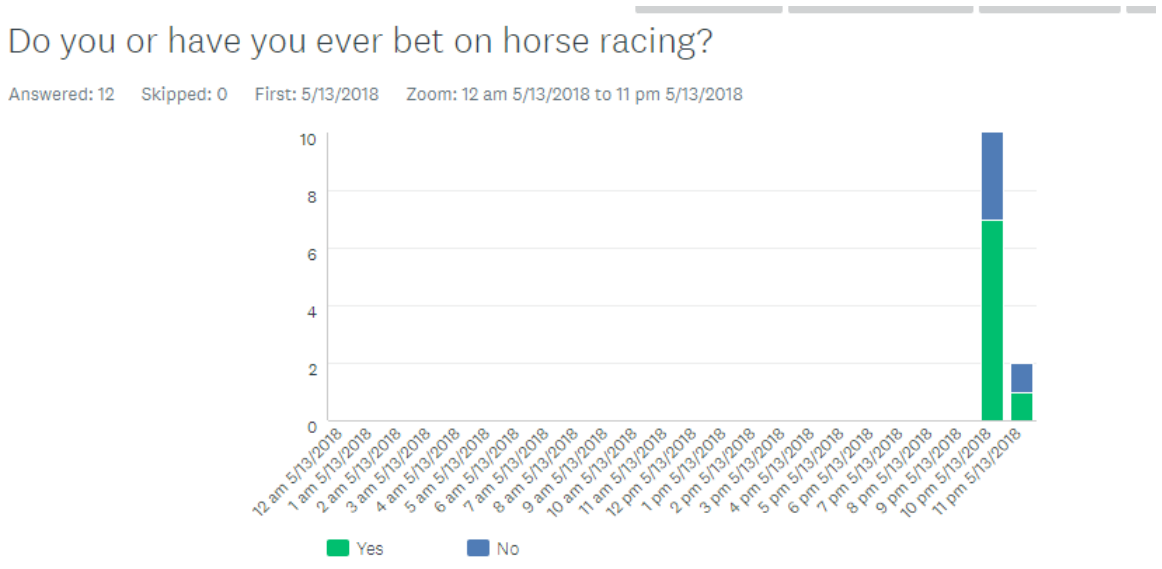
R plots knn algorithm

2.6.2 Unit Testing

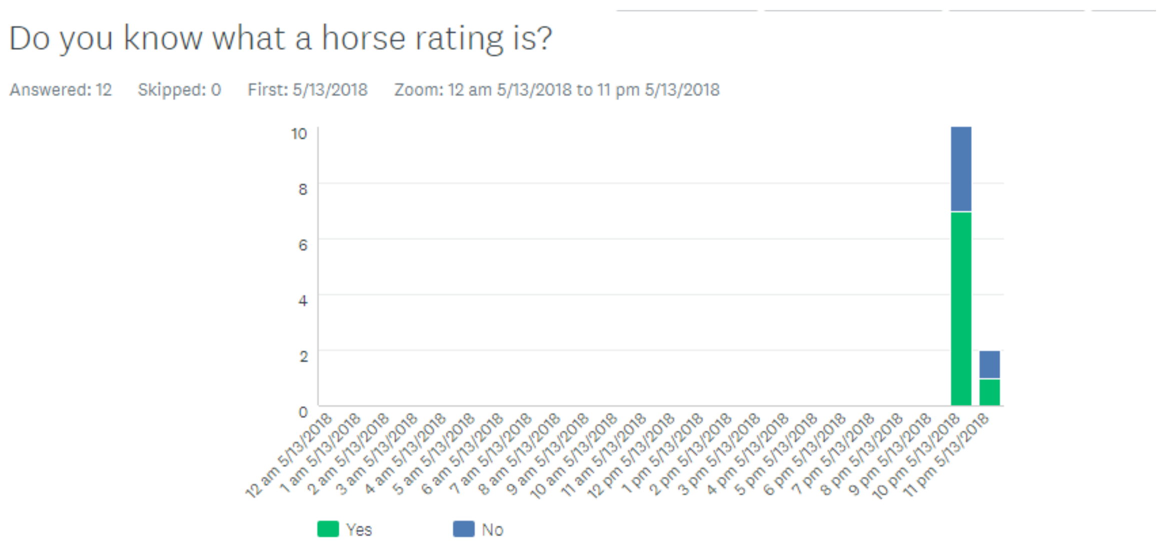
A Unit Testing methodology was employed for testing small components in confinement of factors such as dependencies and the environment that the code will be executed in. Every module has a collection of unit tests that are executed each time a build is ran to enable and ensure there were no undesirable changes since the most recent build or run. Data that is needed by each unit for testing is mocked while external dependencies are stopped. Extractor was used for testing python code it uses standard python libraries for unit testing. The same unit testing was used for UI Server testing which is the R shiny application. Some J unit tests were also done in R studio to test r code.

2.7 *Customer testing*

User Testing is taught to be one of the final stages in the development lifecycle. It is key to validating that a system can complete all the requirements from an end user’s perspective. For the purpose of User Testing a survey was conducted to see what people new about the horse racing industry and if my idea would interst people.


Do you or have you ever bet on horse racing?

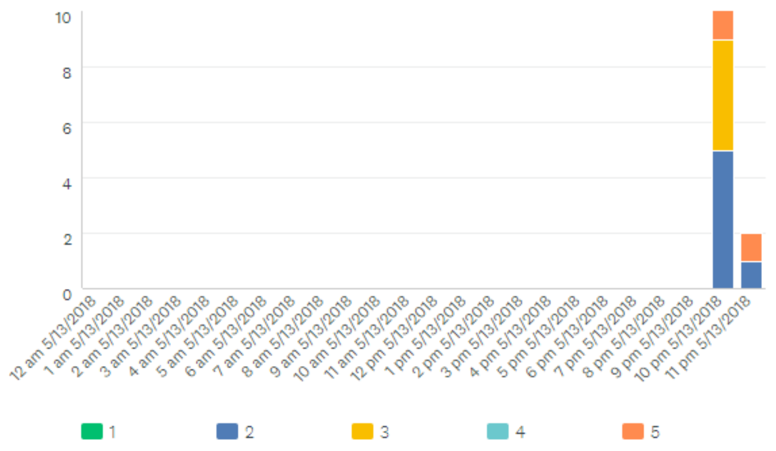
Answered: 12 Skipped: 0 First: 5/13/2018 Zoom: 12 am 5/13/2018 to 11 pm 5/13/2018


Do you know what a horse rating is?

Answered: 12 Skipped: 0 First: 5/13/2018 Zoom: 12 am 5/13/2018 to 11 pm 5/13/2018

If you were to bet on a horse race, how likely would you pick the favourite in the race? Scale 1-5

Answered: 12   Skipped: 0   First: 5/13/2018   Zoom: 12 am 5/13/2018 to 11 pm 5/13/2018

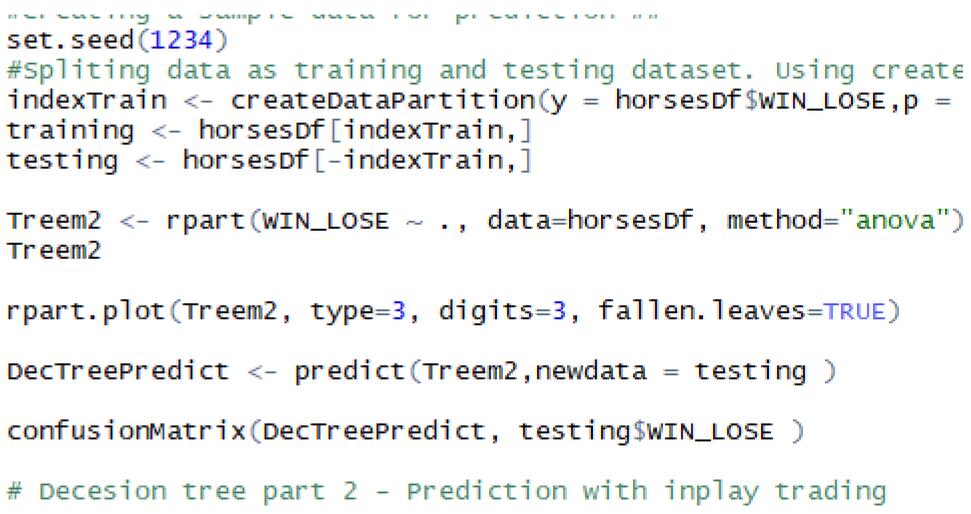


Above figures show some questions and responses from my survey I created. The data trends can be seen on the graphs.

2.8 Evaluation

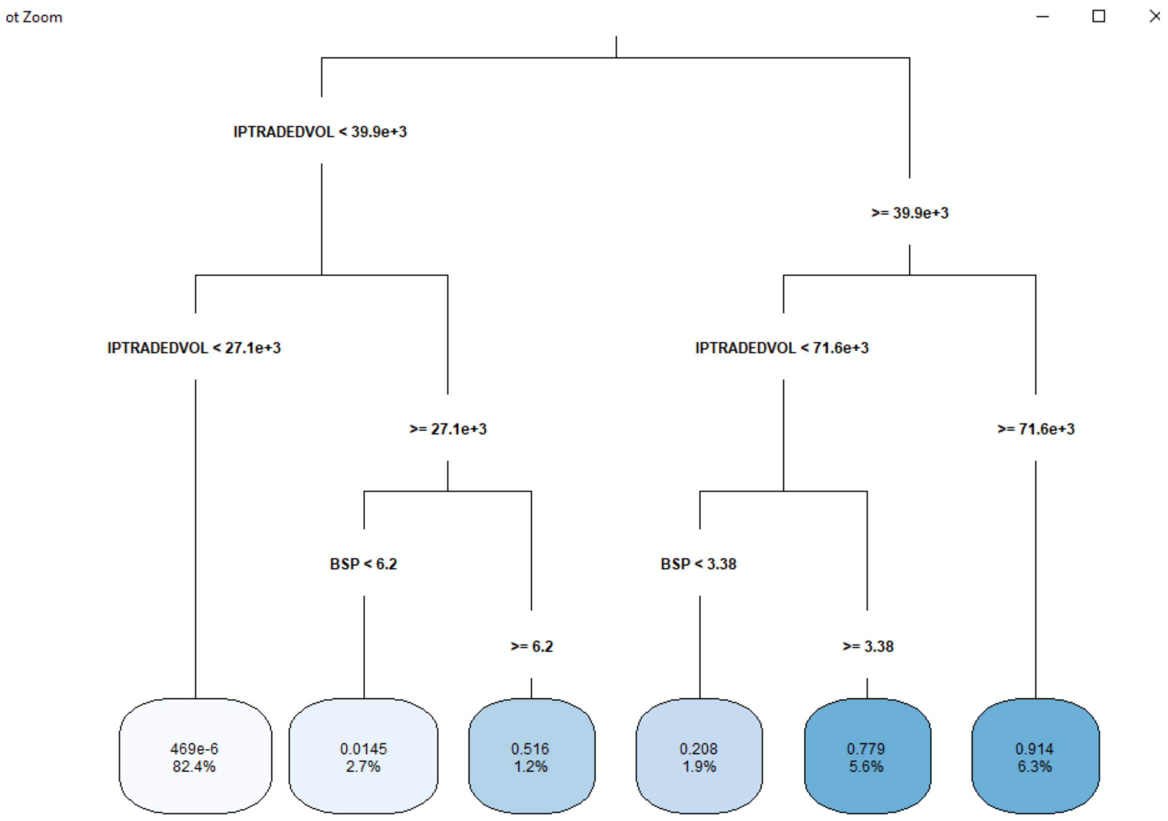
2.8.1.1 Algorithms used in R and their results

Decision Trees – A decision tree is a structure that includes a root node, branches and leaf nodes. Each subjective node denotes a test on an attribute, every branch stands for the outcome of a test, and then each leaf node contains a class label. The node at the very top of the tree is the root node. Below are snippets of code on how I implemented a decision tree algorithm in my predictive model and the output of the decision tree and its results.



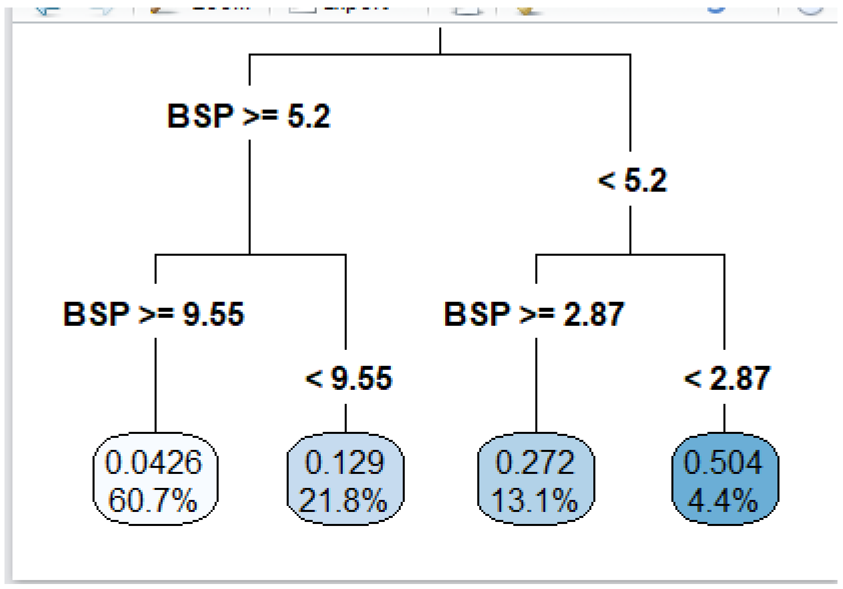
```
set.seed(1234)
#Spliting data as training and testing dataset. Using create
indexTrain <- createDataPartition(y = horsesDf$WIN_LOSE,p =
training <- horsesDf[indexTrain,]
testing <- horsesDf[-indexTrain,]

Treem2 <- rpart(WIN_LOSE ~ ., data=horsesDf, method="anova")
Treem2

rpart.plot(Treem2, type=3, digits=3, fallen.leaves=TRUE)

DecTreePredict <- predict(Treem2,newdata = testing )

confusionMatrix(DecTreePredict, testing$WIN_LOSE )

# Decesion tree part 2 - Prediction with inplay trading
```

The above code is running a decision on my dataset to try and predict whether or not a horse won or lost based off the columns within the dataset. The columns used are : BSP(Basic start price), WIN_LOSE(whether the horse won or lost), MORNINGTRADEDVOL(Amount traded before 11 am GMT on each horse), PPTRADEDVOL(Amount traded before the race went off), and IPTRADEDVOL(Amount traded in-play), I also done another decision tree without IPTRADEDVOL to see how accurate my predictive model is at predicting winners without the in-play traded volume.



The decision outputted these results. What these results mean:

The prediction on the left are all losers and that 82.4% of all horses lost, the second prediction is that all horses that have IPTRADEVOL < 39.8e+3 but greater than 27.1e+3 and have a BSP <6.2 have a 0.0145% chance of winning. Prediction number 6 is that all horses with an IPTRADEVOL >=71.6e+3 have a 0.91% chance of winning, prediction number 5 I find the most interesting as it is all horses that have an IPTRADEVOL >=39.e+3 and IPTRADEVOL <71.6e+3 and a BSP >= 3.38 have a 0.78% chance of winning.



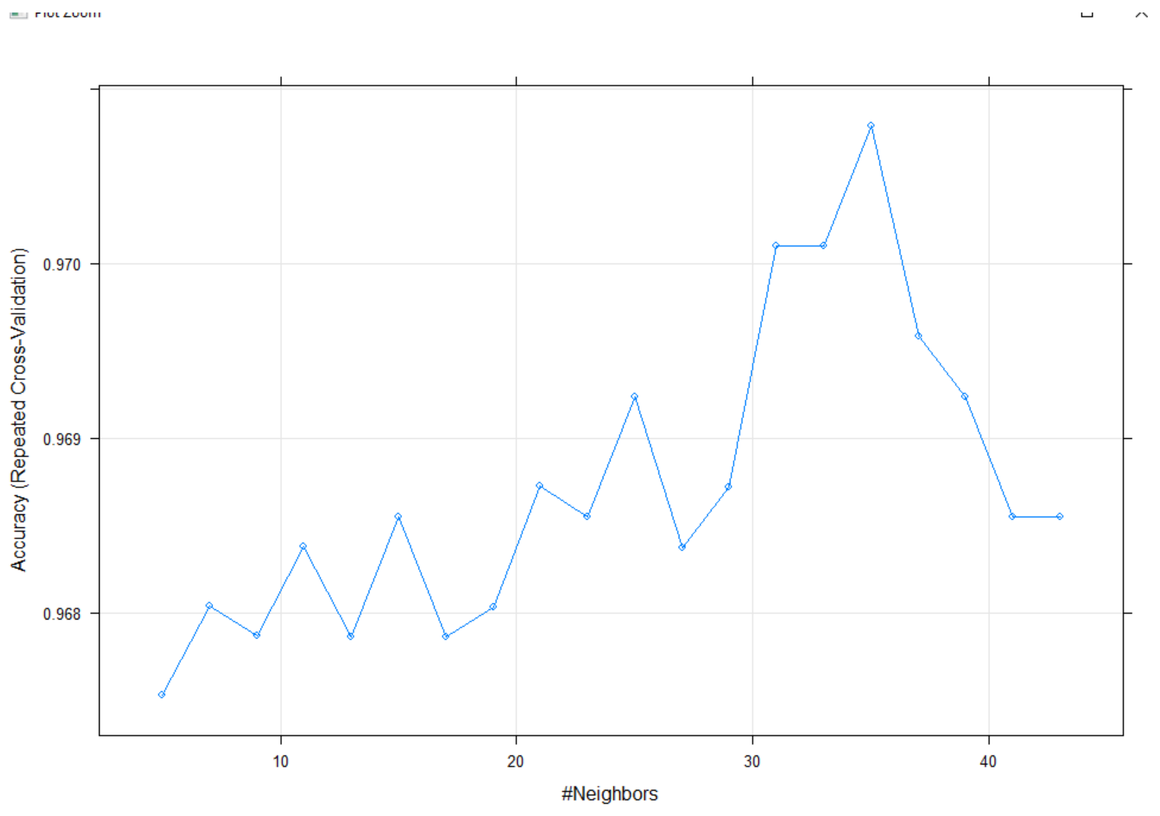
This was my second decision tree and as you can see is clearly not as good or as detailed as the first one. Basically gives 4 prediction with quite low percentages. The highest being prediction number 4 which states that all horses that have BSP <5.2 and < 2.87 have a 0.50% chance of winning. I could've probably done more with this tree but even at what it is it gives a good insight into the data because prediction 4 basically predicts that out of all the horses and horse races that is in my dataset 50% of all favourites with odds less than 2.87 will win, so if you are a gambling person it means you have a 1 in 2 chance of picking the winner of certain races based of the prediction you will win 50% of the time.

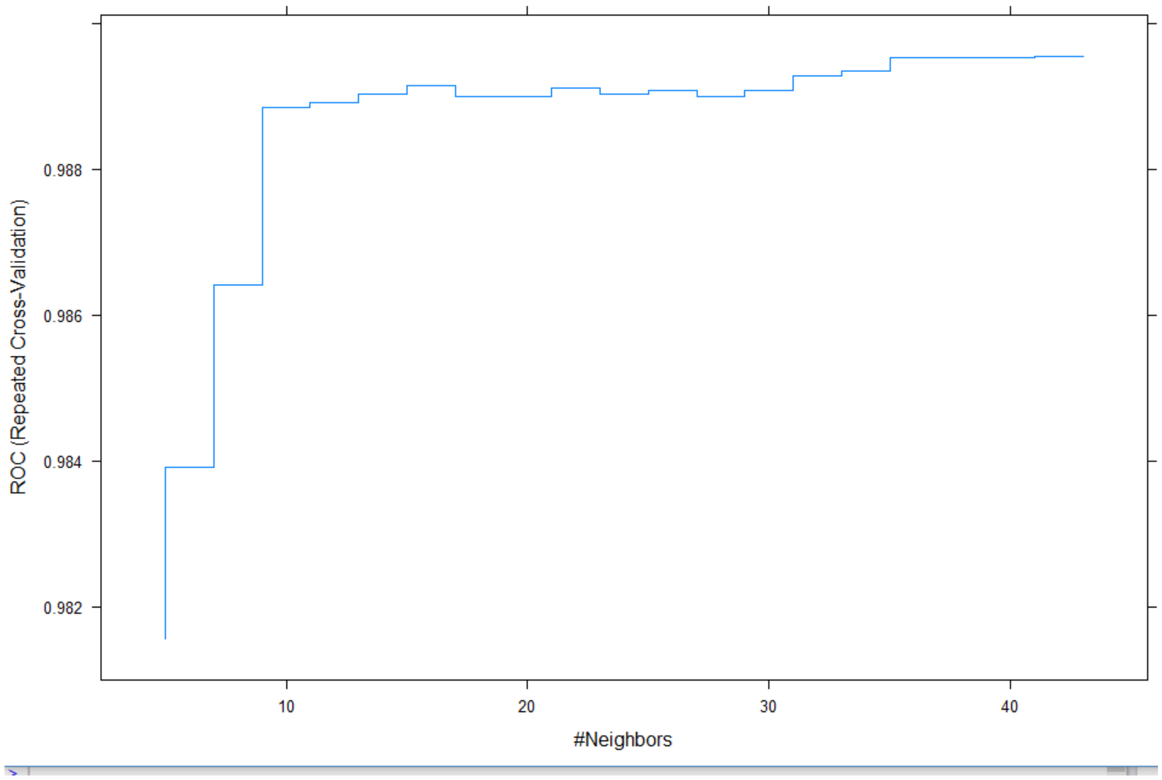
KNN (k - Nearest Neighbour) –

K nearest neighbour is a quite simple algorithm that stores all available cases and classifies new cases based on a similarity measure. KNN has been around since the 70s and has been used in statistical estimation and pattern recognition. So to simplify how the algorithm works is a case is classified by a majority vote of all its neighbours, the case then being assigned to the class most common amongst its K nearest neighbour measured by a distance function. An example is if K=1, then the case is simply assigned to the class of its nearest neighbour.

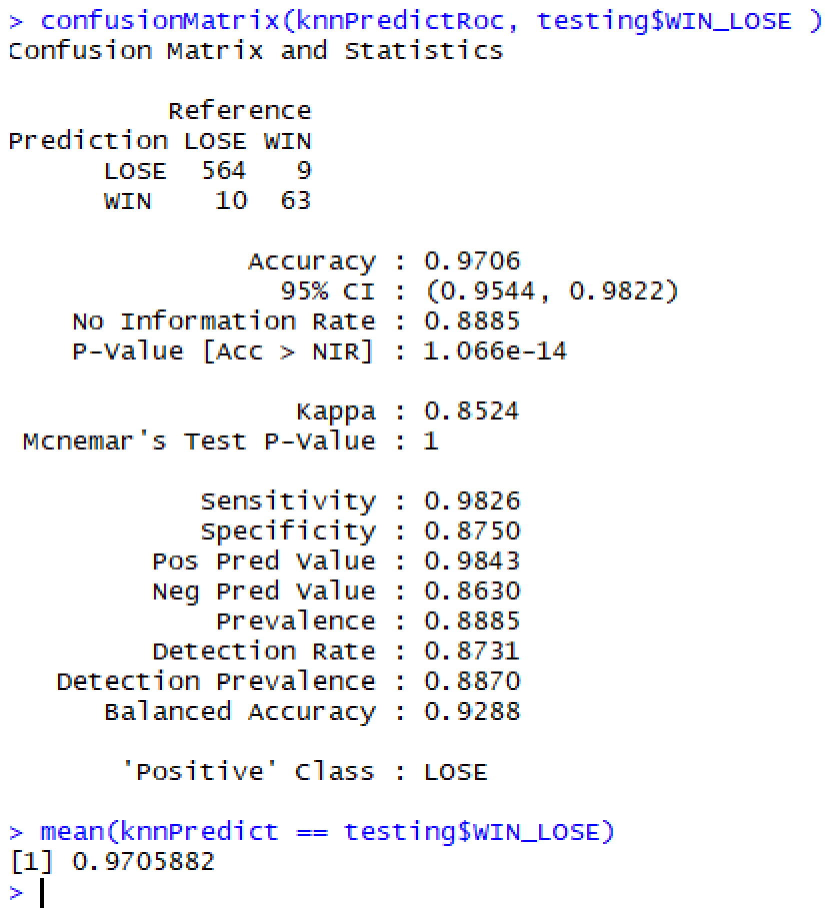
```
prop.table(table(horses$WIN_LOSE)) * 100

trainX <- training[,names(training) != "WIN_LOSE"]
preProcValues <- preProcess(x = trainX,method = c("center", "scale"))
preProcValues

set.seed(1234)
ctrl <- trainControl(method="repeatedcv",repeats = 3) #,classProbs=TRUE,summaryFunction = twoClassSum
knnFit <- train(WIN_LOSE ~ ., data = training, method = "knn", trControl = ctrl, preProcess = c("cent

# FOR ROC
ctrlRoc <- trainControl(method="repeatedcv",repeats = 3,classProbs=TRUE,summaryFunction = twoClassSum
knnFitRoc <- train(WIN_LOSE~ ., data = training, method = "knn", trControl = ctrlRoc, preProcess = c(

#Output of kNN fit
knnFit

#Output of KNN with ROC
knnFitRoc

plot(knnFit)
```

So like the decision tree the data was split into testing and training dataset. I used my algorithms all in the same way so for each one I'm trying to predict whether or not a horse won or lost and then testing the accuracy to see how accurate the algorithm was. Below are the outputted results from KNN.





By looking at the two plots above we can see the results of the KNN algorithm on this predictive model. The first plot shows where k is most accurate we can see that k = 35 has the highest percentage. The second plot shows that with this predictive model it seems that the more neighbours there is the higher the percentage is at predicting whether or not a horse won or lost a race.

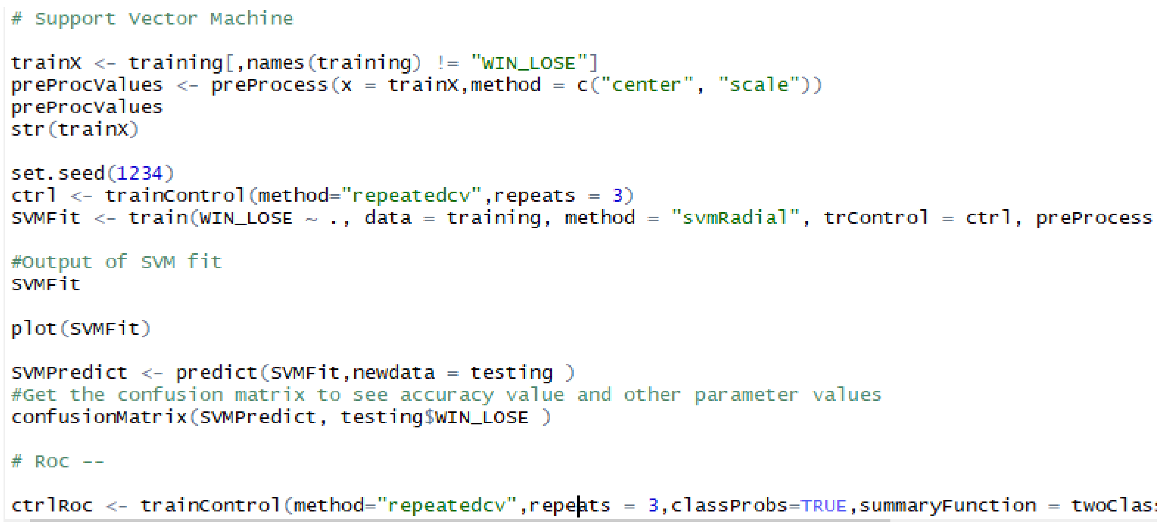


```
> confusionMatrix(knnPredictRoc, testing$WIN_LOSE )
Confusion Matrix and Statistics

          Reference
Prediction LOSE WIN
      LOSE  564   9
      WIN    10  63

               Accuracy : 0.9706
                 95% CI : (0.9544, 0.9822)
    No Information Rate : 0.8885
    P-Value [Acc > NIR] : 1.066e-14

                  Kappa : 0.8524
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9826
            Specificity : 0.8750
         Pos Pred Value : 0.9843
         Neg Pred Value : 0.8630
             Prevalence : 0.8885
         Detection Rate : 0.8731
   Detection Prevalence : 0.8870
      Balanced Accuracy : 0.9288

       'Positive' Class : LOSE

> mean(knnPredict == testing$WIN_LOSE)
[1] 0.9705882
> 
```

The above figure shows the code produces to test the accuracy of the KNN algorithm on my predictive model. A confusion matrix is used to test the accuracy of the algorithm. We can see that the accuracy of the algorithm is 97% which is quite high and is a very good accuracy reading. This means that the KNN algorithm is 97% sure with its predictions of whether a horse won or lost a race.

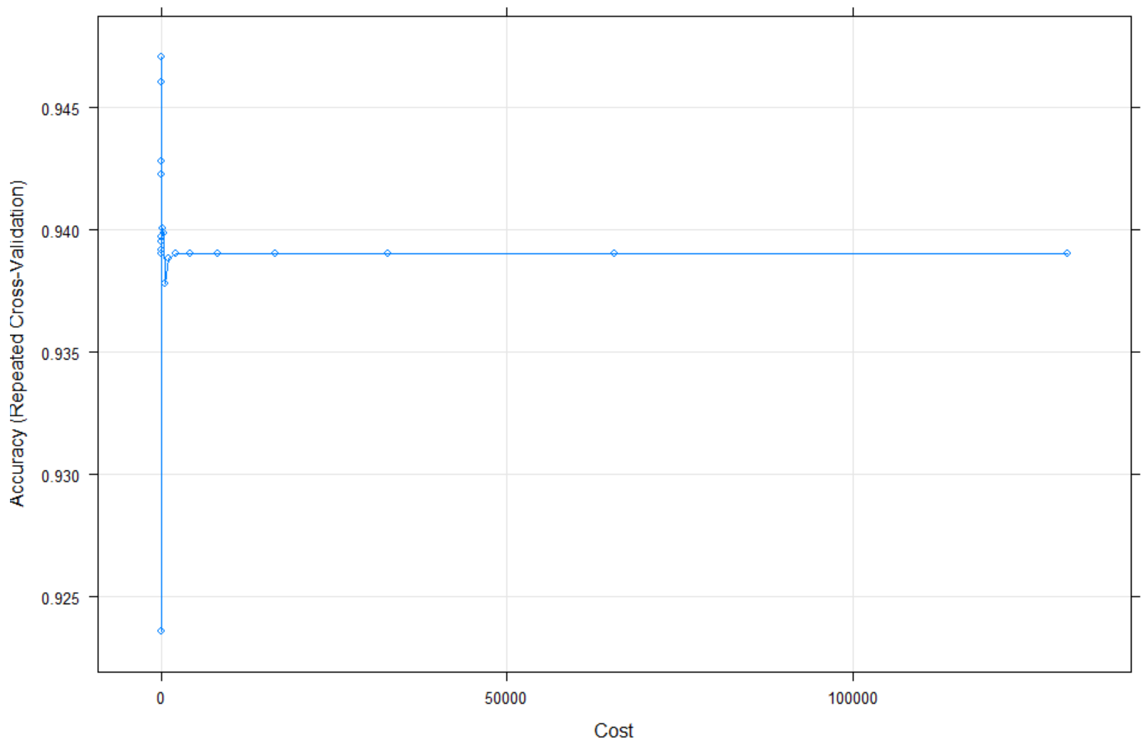
SVM (Support Vector Machine) –

This is the last algorithm I used to for my predictive model. A Support Vector Machine is a selective classifier formally defined by a separating hyperplane, this means that given labelled training data, the algorithm outputs an optimal hyperplane which categorizes new examples.

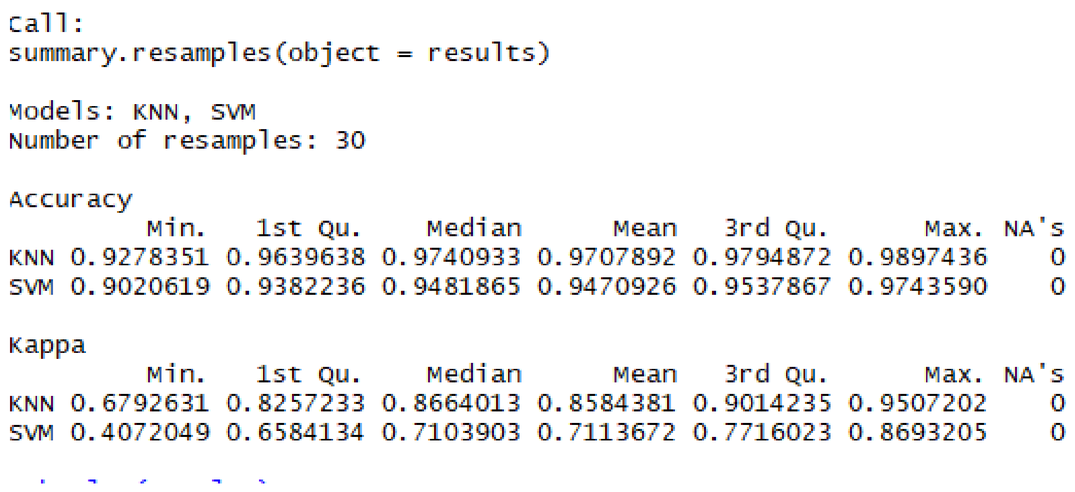


```
# Support Vector Machine

trainX <- training[,names(training) != "WIN_LOSE"]
preProcValues <- preProcess(x = trainX,method = c("center", "scale"))
preProcValues
str(trainX)

set.seed(1234)
ctrl <- trainControl(method="repeatedcv",repeats = 3)
SVMFit <- train(WIN_LOSE ~ ., data = training, method = "svmRadial", trControl = ctrl, preProcess

#Output of SVM fit
SVMFit

plot(SVMFit)

SVMPredict <- predict(SVMFit,newdata = testing )
#Get the confusion matrix to see accuracy value and other parameter values
confusionMatrix(SVMPredict, testing$WIN_LOSE )

# Roc --

ctrlRoc <- trainControl(method="repeatedcv",repeats = 3,classProbs=TRUE,summaryFunction = twoClas
```

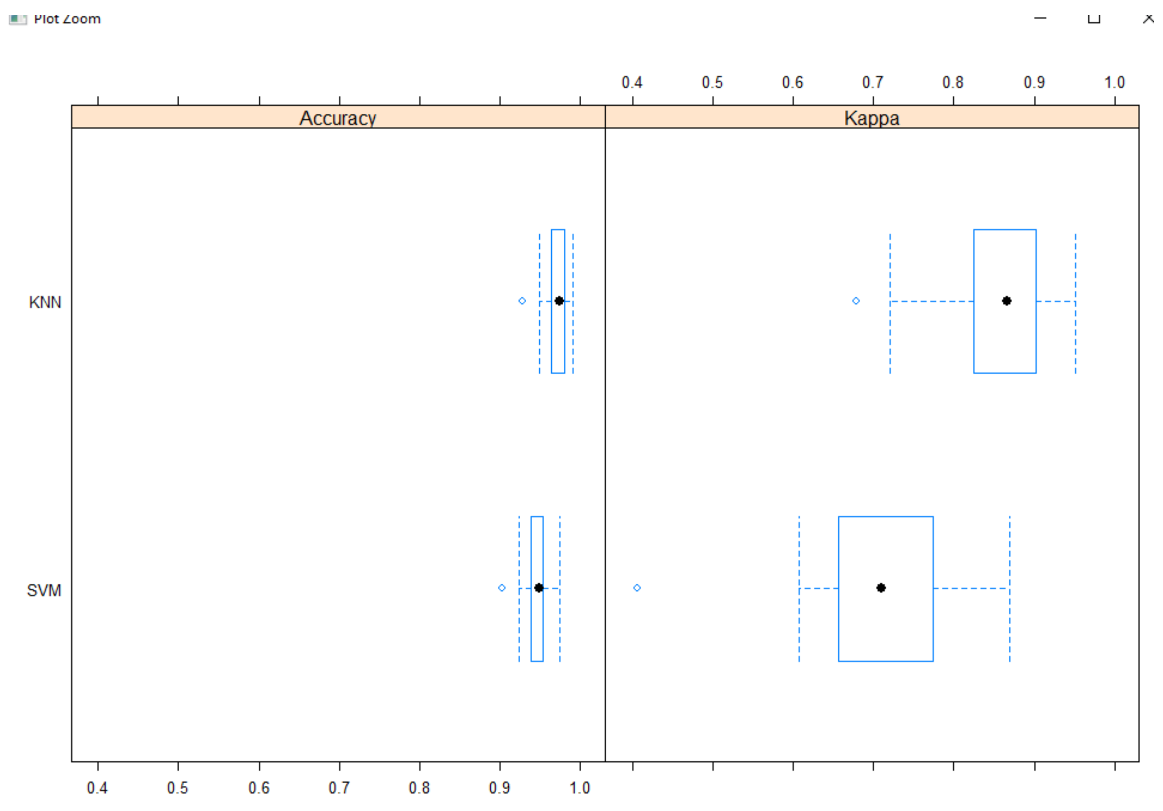
The above code is very similar to KNN only for pretty much replace the methods. The code itself look quite similar, just like KNN with SVM my predictive model is trying to predict whether or not a horse won or lost and then a confusion matrix is used to test the accuracy of the algorithm.

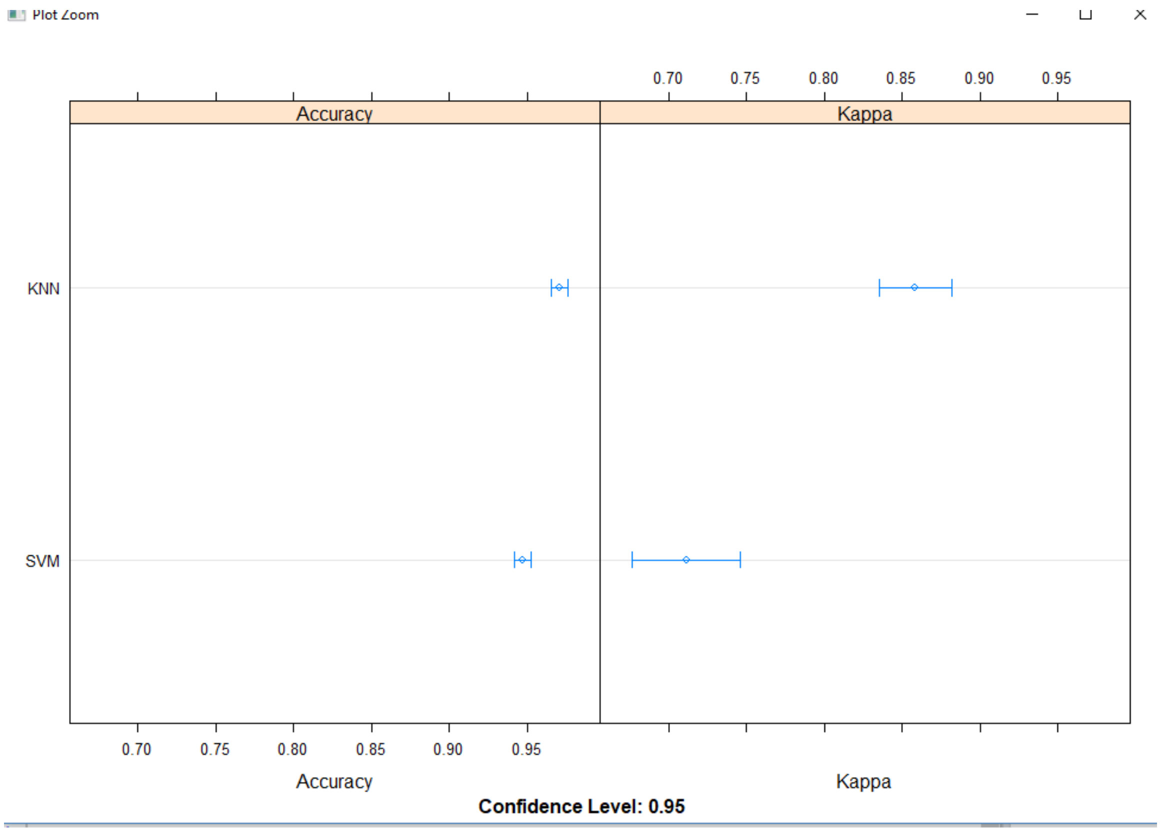


Comparing KNN and SVM against each other. The below graphs are comparing the accuracy of the KNN algorithm against the accuracy of the SVM algorithm of the predictive model, the results show which algorithm works best for this particular predictive model.



```
Call:
summary.resamples(object = results)

Models: KNN, SVM
Number of resamples: 30

Accuracy
         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
KNN 0.9278351 0.9639638 0.9740933 0.9707892 0.9794872 0.9897436    0
SVM 0.9020619 0.9382236 0.9481865 0.9470926 0.9537867 0.9743590    0

Kappa
         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
KNN 0.6792631 0.8257233 0.8664013 0.8584381 0.9014235 0.9507202    0
SVM 0.4072049 0.6584134 0.7103903 0.7113672 0.7716023 0.8693205    0
```





I have come to the conclusion that KNN works best out of the 3 algorithms I used for my predictive model.

3 Conclusions

In this report, the aims of the project were to design and develop a horse rating predictive model analysis system and compare the system with the Racing posts own system and compare results. The background of the project was outlined at the start of this report along with the reasons why I wanted to do this project and make this model. The technologies that were used to make this model and application were investigated in detail. Section 2 of the document mainly defines the system, it include all requirements that the system needs and uses. Unfortunately to obtain access to more detailed data from the API a 300€ payment had to be made, I am not financially capable to afford spending that type of money on college work, So I had to tweak the model slightly and instead of predicting horse rating my model predicted the winners and losers of past races based of all traded information, by traded information I mean all bets and money that were out on each horse. KNN was the best algorithm and gave a 97% accuracy at predicting weather a horse won or lost a race. I am happy still with the model I came up with but in the future I would definitely consider making that purchase and doing my analysis in more detail to come up with a horse rating prediction model. Testing was done at almost every stage of the KDD approach that was followed for the analysis of the project. R shiny application was created to allow and give the user access to my data and analysis, this then allowed users to interact with visualisation graphs. So to conclude my model was quite successful but not quite the model that I originally wanted to have for this project.

4 Further development or research

As of today the system is continuing to grow and evolve my adding more data to the project and as well as gaining more experience with R shiny to have a nice styled R shiny application. As mentioned in the conclusion in the near future I will look into purchasing the subscription so that I can have an even more detailed dataset which will allow me to make my horse rating prediction model, I can follow the same methods that I used for this project and pretty much use this project as a baseline or as a guide to follow.

5 References

BetFairs API

6 Appendix

6.1 Project Proposal

6.1.1 Objectives

Objective 1: My first objective is to find a dataset on Race Horses or horse racing results there is a few other datasets I could go for but I think these will work the best.

Objective 2: My second objective will be to clean the dataset to pull out the information I require, age, grades, weight, form, track conditions, weather, previous results, breed etc.

Objective 3: My third objective is to compare information based of previous races and see if there is a main contribution to the rating a horse gets after each race and check all of the other factors and see if and how they contribute towards the rating.

Objective 4: My fourth objective using some machine learning based on previous racing results to try and see if there is a trend in the winner of the races if the horse is majority the highest rated hose in the race.

Objective 5: Create an anonymous survey for a random selection of people to see if people would bet on a horse based of their horse ratings, if they know what a horse rating is and how a horse rating it is determined.

Objective 6: Complete documentation with my views on my study and weather people should take in to consideration horse rating and not just look at the odds of a horse.

6.1.2 Background

The reason for my idea came from my own personal interests as my Grandfather was very interested in horse racing, I became interested in it along with my sister being in show jumping competitions. I would attend a horse racing event once or twice a year since I turned 18. I have attended many outside of Dublin where I Live and would like to go to England as well. I was always interested in other things in regards to horse racing for example the trainers who train horses the jockeys, also my close friend and her family owned some race horses which always got me excited and talking and listening to their trainers talk about horses who people have probably never heard before.

So when I started back college this year I didn't really have an idea and was unsure what I could do for my Final year project, I immediately taught sports as that is what I am interesting in and I found that Football was almost a too broad area to do a project on. I then read an article about horse racing online and it came to my knowledge that there were only two official companies/organizations that rated horses being the RacingPost and the British Horse Racing Board, the Racing Post being the more popular option as it is more up to date. I then started to wonder how they rated the horses and I looked into it and they gave the factors that were considered when rating a horse and how it could be rated. I then decided I want to do a further analysis on how race horses get there ratings.

Most bookmakers give their starting odds for horses based of these racing systems but obviously not all the time do the best horses or the favourites win, so I want to do a study and see how these horses are rated and how many of the highest rated horses are successful and basically see how accurate these rating systems are.

6.1.3 Technical Approach

I will approach my project by using the KDD (Knowledge Discovery in Databases) approach which has some key steps in order to build a successful data analytics project. Below are the steps for this approach.

Selection- This involves finding the information and datasets that will be best for my project in terms of what I want my end result to be.

Pre-processing- This will be me cleaning the data so that I can get rid of all the unwanted data that I don't need for my project.

Transformation- This will involve using the clean dataset and generating better data, for the data mining stage methods which include dimension reduction, such as feature selection and extraction, record sampling, and attribute transformation such as discretization of numerical attributes and functional transformation.

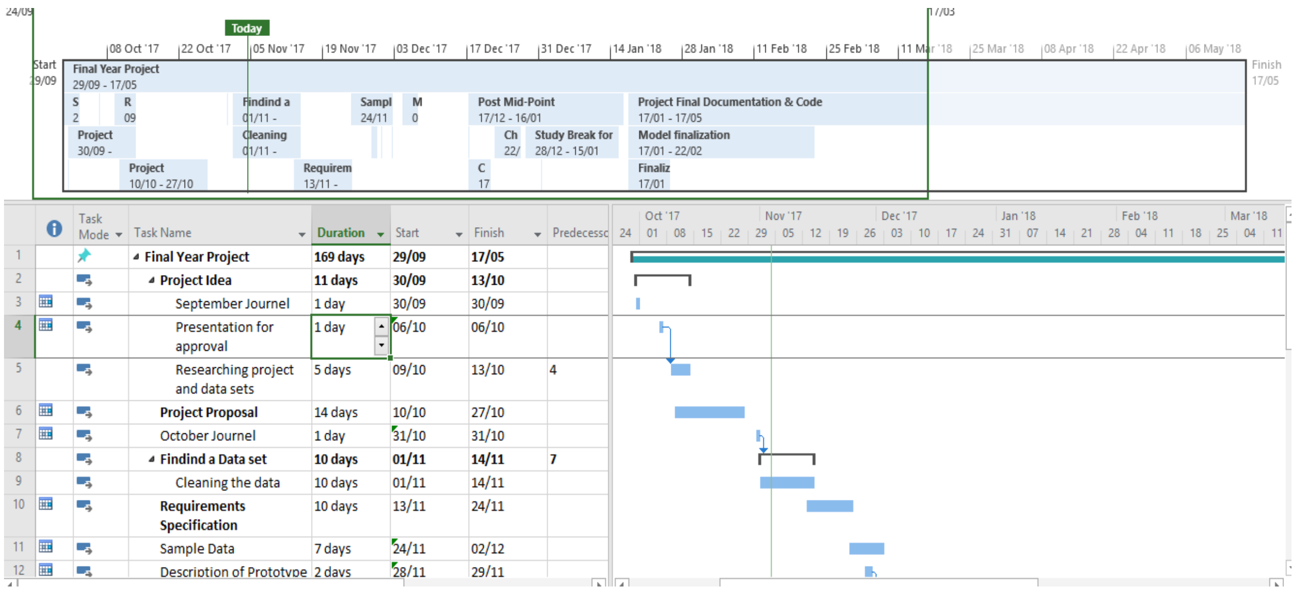
Data Mining- This is where I will use the descriptive data mining technique which included unsupervised, and visualization aspects of data mining.

Interpretation- Here I will interpret the knowledge gained from the records I have acquired.

6.1.4 Special resources required

I have no required special resources.

6.1.5 Project Plan



	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		**Final Year Project**	**169 days**	**29/09**	**17/05**	
2		**Project Idea**	**11 days**	**30/09**	**13/10**	
3		September Journel	1 day	30/09	30/09	
4		Presentation for approval	1 day	06/10	06/10	
5		Researching project and data sets	5 days	09/10	13/10	4
6		**Project Proposal**	14 days	10/10	27/10	
7		October Journel	1 day	31/10	31/10	
8		**Findind a Data set**	**10 days**	**01/11**	**14/11**	7
9		Cleaning the data	10 days	01/11	14/11	
10		**Requirements Specification**	10 days	13/11	24/11	
11		Sample Data	7 days	24/11	02/12	
12		Description of Prototype	2 days	28/11	29/11	

6.1.6 Technical Details

R Studios

I will use R Studios to build this project, which is an open source integrated development environment for R Language which itself is a programming language for statistical computing, graphics and will be the language I write this project in. R studios will be in conjunction with excel as this is where I will store my dataset and use it to retrieve information as needed such as horse racing results.

R Language

I will use R language to build my project and display information I have not decided which libraries I will use within R studios yet.

Python

Python has a design philosophy that emphasizes code readability and a syntax that allows programmers to express concepts in fewer lines of code than might be used in other languages. I will be using Python to pull data from the API and I will also be using python scripts to convert the data to a csv file so I can use the data in R studio and run R on the data.

SPSS

SPSS is one of the most popular statistical packages which can perform highly complex data manipulation and analysis with simple instructions.

I will mainly use SPSS to backup what I am displaying using it as more as an end user friendly way of viewing the data so that a person with no technical ability will be able to manipulate the data with built in commands using it as a testing tool will help me solidify the information that I am looking to produce.

Excel

Excel is a spreadsheet tool with built in statistical and graphing commands that allow a user to manipulate information and data loaded onto it. I am going to use excel to hold information in a more structured format until I start to clean it and remove all the data that I do not want.

Tableau

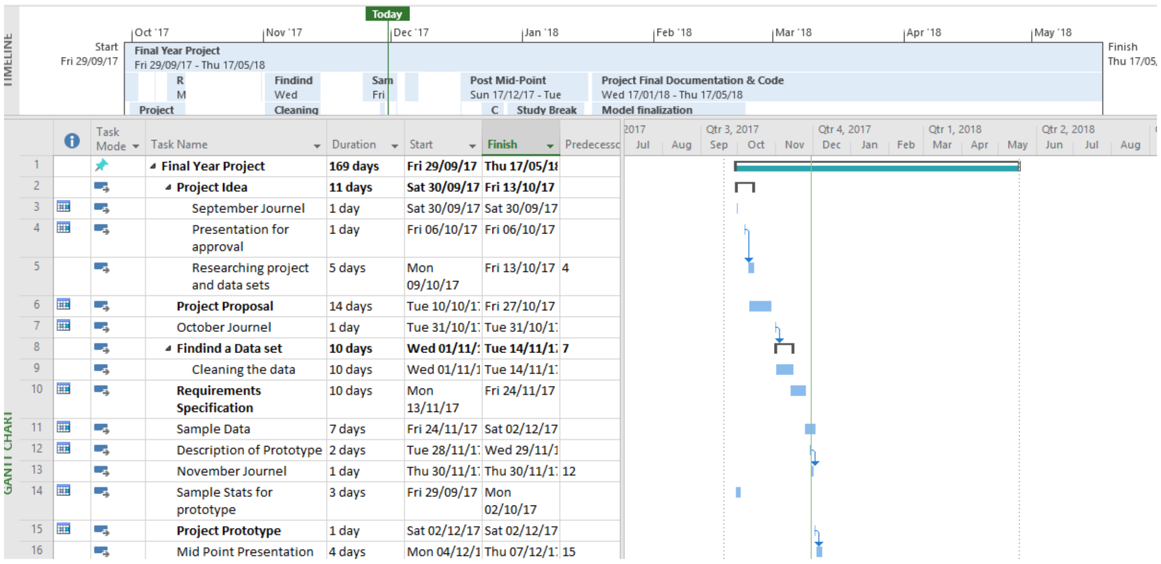
Tableau allows for instantaneous insight by transforming data into visually appealing, interactive visualizations called dashboards.

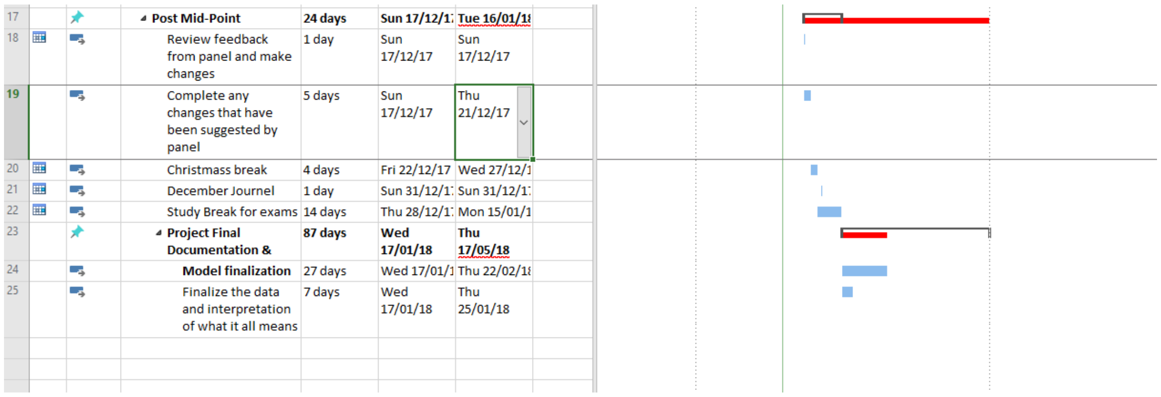
I will use this to Display all knowledge and interpretations of final figures I acquire through my datasets so an end user may easily understand the results which are being displayed.

6.1.7 Evaluation

I will use SPSS to test my data accuracy. I will also run various other tests such as API testing, reliability testing, conversion testing and storage testing. I will test my results by asking a user to navigate through my dashboard, making sure all KPIs (*Key Performance Indicator*) are fully functional.

6.2 Project Plan





	Task Mode	Task Name	Duration	Start	Finish	Predecessc
1		**Final Year Project**	**169 days**	**Fri 29/09/17**	**Thu 17/05/1ε**	
2		**Project Idea**	**11 days**	**Sat 30/09/17**	**Fri 13/10/17**	
3		September Journel	1 day	Sat 30/09/17	Sat 30/09/17	
4		Presentation for approval	1 day	Fri 06/10/17	Fri 06/10/17	
5		Researching project and data sets	5 days	Mon 09/10/17	Fri 13/10/17	4
6		**Project Proposal**	14 days	Tue 10/10/1:	Fri 27/10/17	
7		October Journel	1 day	Tue 31/10/1:	Tue 31/10/1:	
8		**Findind a Data set**	**10 days**	**Wed 01/11/:**	**Tue 14/11/1:**	7
9		Cleaning the data	10 days	Wed 01/11/1	Tue 14/11/1:	
10		**Requirements Specification**	10 days	Mon 13/11/17	Fri 24/11/17	
11		Sample Data	7 days	Fri 24/11/17	Sat 02/12/17	
12		Description of Prototype	2 days	Tue 28/11/1:	Wed 29/11/1	
13		November Journel	1 day	Thu 30/11/1:	Thu 30/11/1:	12
14		Sample Stats for prototype	3 days	Fri 29/09/17	Mon 02/10/17	
15		**Project Prototype**	1 day	Sat 02/12/17	Sat 02/12/17	
16		Mid Point Presentation	4 days	Mon 04/12/1	Thu 07/12/1:	15
17		**Post Mid-Point**	**24 days**	**Sun 17/12/1:**	**Tue 16/01/1ε**	
18		Review feedback from panel and make changes	1 day	Sun 17/12/17	Sun 17/12/17	
19		Complete any changes that have been suggested by panel	5 days	Sun 17/12/17	Thu 21/12/17	
20		Christmass break	4 days	Fri 22/12/17	Wed 27/12/1	
21		December Journel	1 day	Sun 31/12/1:	Sun 31/12/1:	
22		Study Break for exams	14 days	Thu 28/12/1:	Mon 15/01/1	
23		**Project Final Documentation &**	**87 days**	**Wed 17/01/18**	**Thu 17/05/18**	
24		**Model finalization**	27 days	Wed 17/01/1	Thu 22/02/1ε	
25		Finalize the data and interpretation of what it all means	7 days	Wed 17/01/18	Thu 25/01/18	

6.3 Monthly Journals

6.3.1 September

My Achievements

This month Is the first month I started back college, I was told to have a project idea by the end of week one so I began thinking of project ideas and researching on google looking for ideas as I originally wasn't too sure what my idea was. I wanted to do my final year project on something I am interesting in so I immediately taught of something sport related, I taught I would do something around formula 1 or football. I ended up coming up with an idea based around how racehorses are rated.

My Reflection

I came up with this idea as I am interested in Horse racing and I always wondered how the horses were rated, for example in football teams have rankings. I began looking into it and I found out that that every race horse has two rating an OR which is the BHB (British Horse Racing Board) and RPR which is the Racing Post Rating, the RPR is more accurate because they attach higher weight to recent performances. I found this interesting as they don't really specify what makes up these ratings well they do but not as in much detail as I would like to know, they specify the factors that contribute toward the ratings but that's really all, so when doing my project I plan do get Data on previous horse races and statistically analyse the data and make up a formula or program which determines the rating of the horses. From this I will be able to see for myself and as well document for others how the race horses are actually rated not just the factors that are involved in rating the horses I hope to be able to make it clear to see how all these factors make up the ratings the horses are given, some factors are : weight, ground surface, weather, grade, previous result/results, distance, age etc.

Intended Changes

Next month I will try to get an understanding in how I am going to obtain the data I need as well as plan out on what I need to do in order to complete my project and get the answers I am looking for.

I also want to know what I need to do what kind of technologies I need to do to and how I am going to display the data that I get as my output, I have an idea as what technologies I am going to use and I am going to draw a mock-up of what way I want my data to look like.

Supervisor Meetings

Date of Meeting: Not yet assigned

Items discussed:

Action Items:

6.3.2 October

My Achievements

Learning the API and exploring the data

My Reflection

I am getting a better understanding of the API and the data.

Intended Changes

Next month I will try to get an understanding in how I am going to obtain the data I need as well as writing code to pull data from the API

Supervisor Meetings

Date of Meeting: Every Wednesday at 12P.M

Items discussed: steps for project and next steps to take

Action Items: Tasks that I can try do for the following week.

6.3.3 November

My Achievements

Learning the API and exploring the data

My Reflection

I am getting a better understanding of the API and the data.

Intended Changes

Next month I will try to get an understanding in how I am going to obtain the data I need as well as writing code to pull data from the API

Supervisor Meetings

Date of Meeting: Every Wednesday at 12P.M

Items discussed: steps for project and next steps to take

Action Items: Tasks that I can try do for the following week.

6.3.4 January/February

My Achievements

Learning the API and exploring the data

My Reflection

I am getting a better understanding of the API and the data.

Intended Changes

Next month I will try to get an understanding in how I am going to obtain the data I need as well as writing code to pull data from the API

Supervisor Meetings

Date of Meeting: Every Wednesday at 12P.M

Items discussed: steps for project and next steps to take

Action Items: Tasks that I can try do for the following week.

6.3.5 March/April

My Achievements

Created application

Almost done report

Almost done project

My Reflection

I am getting a better understanding of the API and the data.

Intended Changes

Next month I will try to get an understanding in how I am going to obtain the data I need as well as writing code to pull data from the API

Supervisor Meetings

Date of Meeting: Every Wednesday at 12P.M

Items discussed: steps for project and next steps to take

Action Items: Tasks that I can try do for the following week.

6.4 Other Material Used

Any other reference material used in the project for example evaluation surveys etc.