Predicting association football match outcomes using social media and existing models.

One Semester Individual Project
CM3203

Final Year Project

Author: Kiran Smith
Supervisor: Dr. Steven Schockaert
Moderator: Dr. Alia Abdelmoty

Abstract

Whilst predicting association football matches has historically been a popular topic and area of research, few models are genuinely diverse and innovative in their approach. Existing models have not been modernised to account for the 21st century phenomena of big data that could be used to raise the accuracy of such models. This project aims to build on existing probabilistic models by using data generated from Twitter, alongside the use of more traditional statistical models to achieve the paper title "Predicting association football match outcomes using social media and existing models".

With the combination of text classification, a Poisson Distribution implementation and betting odds provided by bookmakers, it was found that we can indeed increase the accuracy of existing methods, and that combining these three inputs achieved a higher accuracy than each method individually.

This paper provides an insight into the benefits of social media data in predicting sports events, and a basis from which this model could be incorporated into a balanced and profitable betting strategy.

Acknowledgements

I would like to thank Dr. Steven Schockaert for all of his help and guidance throughout the duration of this project.

Contents

1 - Introduction

1.1 Aims/Goals of Project

The emergence of the online sports betting markets in the UK has seen the football betting industry grow by 23% from 2008 to 2013 [17]. In the same period, the increase in popularity of social media websites has seen exponential user growth, with Facebook's users increasing by 1130% [18][19], and Twitter's users growing by 2948% [20][21].

Given that Twitter's 288 million active users collectively create 500 million tweets per day [1], social media is clearly a data rich resource that has great potential. One way in which text data generated from Twitter (a tweet) can be used to fulfil its potential, and one that this project aims to further explore, is in the field of sentiment analysis, i.e. attempting to determine whether a tweet can be classified as either positive or negative, so we can better understand the sentiment on a given subject.

This project aims to reliably predict association football match outcomes from fixtures in the Premier League. This will be achieved through a combination of inputs. Firstly, through performing a sentiment analysis on tweets; i.e. determining whether a tweet mentioning a team can predict whether that team will win, lose or draw; secondly, using the results from an implemented Poisson Distribution model for each fixture, and finally using betting odds for each outcome of matches in the Premier League.

1.2 Intended Audience/Beneficiaries of Work

Considering the subject matter and scope of this project, I would expect this project and its findings to be of interest to the gambling industry as a whole. Sports betting establishments may be interested in adopting or incorporating a solution that uses social media sentiment into algorithms that determine betting odds. Meanwhile, customers of these betting establishments may be interested in using the findings of this project to devise a profitable betting strategy against bookmakers.

More generically, organisations from all industries are likely to be interested in the results of this project, to justify an implementation of a solution, which uses tweet sentiment analysis that is appropriate for their needs. Whilst there has already been research into using tweet sentiment to predict events [2][3][5][6], it is possible that tweet sentiment analysis can be applied more generally in other fields, such as crime prevention; for example, detecting if there is an association between areas with high crime rates and tweets sent from the area, which contain negative sentiment, and devising an appropriate strategy to reduce these high crime rates.

1.3 Scope

Overall this project aims to collect, store, process and classify tweets, with the aim of using the results alongside a Poisson Distribution model and betting odds in order to predict football match outcomes. The tweets will be related to the 20 teams in the Premier League, by using keywords related to each team as a parameter in the request to the Twitter Streaming API. A list of these keywords is available in Appendix 1a at the end of this document.

Given that 277,000 tweets are created each minute alone on Twitter [22], this project will involve the efficient collection and storage of data from the Twitter API, into a MongoDB instance using the Twitter-Tap library for Python.

The collected tweets will then be separated into different time periods, which are associated with a set of matches in the Premier League. Further information on these time periods are available in Appendix 2. From here, this project will process and filter the tweets before using a Naïve-Bayes classification model provided by the Natural Language Tool Kit (NLTK) library for Python [16], to test whether it is possible to accurately classify a data set into the classes "Win", "Draw" or "Lose".

In order to test the suitability of the feature set used for the Naïve Bayes classification implementation, this project will also introduce the Chi-Square feature selection algorithm, which aims to make the classifications more accurate.

Taking into account the results produced by the Naïve Bayes Classifier, a Poisson

Distribution implementation that predicts how many goals a team will score (and thus, the likelihood of them winning), and incorporating the match's betting odds, this project will predict whether a team will win, draw or lose.

1.4 Assumptions

For the purposes of this project, I am going to make the following assumptions:

I shall ignore any other football matches that have taken place in the same time period, and thus, will assume that all tweets mentioning a team relate to Premier League matches, where appropriate. For example, I recorded data whilst matches from other competitions were ongoing, such as the FA Cup, the Champions League, the Europa League and the League Cup. However, I am not going to include an extensive data set from 8 days in late March where there was a break from Premier League football for international team matches.

Given that statistically there is a higher chance of a team winning or losing a match than drawing a match, it is accepted that there may be more data for instances of teams winning and losing, than drawing. As such, it may be more difficult to correctly classify tweets for draws as the feature set may not be as varied as it is for wins and losses.

Finally, I am going to assume that all tweets mentioning a team are related to football matches and activities, and not other news related items that do not impact upon a football match outcome. For example in mid-February, a group of Chelsea FC fans were involved in a racist abuse incident in Paris [44], and consequently some mentions of Chelsea here may impact the data set.

2 - Background

2.1 Literature Review

In the late 2000s, the emergence and subsequent high adoption rates of social media provided reliable platforms for the general public and organisations alike to publish opinion and fact. The breadth and availability of platforms such as Twitter, allows a wider public insight into a variety of different subject areas, and as such, provides the ability to use data generated from Twitter to produce future event insights. Research projects based on data primarily sourced from Twitter have shown the ability to predict future events in stock markets [2], political elections [3], the spread of diseases [5] and NFL matches [6] among others.

As Twitter has grown into a platform that has over 288 million active users that collectively generate 500m tweets each day [1], the vast amount of freely accessible data through the Twitter API make it an excellent and valuable source of data to use in text classification cases. Whilst the use of data from this platform has had success in the past in predicting future events as listed above, Tumasjan et al acknowledged the drawbacks from using Twitter as a basis for prediction, by noting that its users are not representative of the entire population and only those who are active on Twitter [3].

Gayo-Avello further noted the drawbacks of using Twitter data for prediction, by observing the inability of different implementations to detect sarcasm in text [7] among others. With respect to their findings, this project aims to use text classification from sentiment analysis upon Twitter data as only part of the formula that will determine the final prediction of whether a team will win, draw or lose.

Over time there have been many notable attempts to predict the results of association football matches, including Constantinou in 2012 [4], and the most popular methods used involve the Poisson Distribution model, as with Maher in 1982 [8], Dixon and Coles in 1997 [9] and Gardner in 2011 [10]. As such, the approach taken with this project will be using a Poisson Distribution model similar to Maher's, due to the relative ease and accuracy of this method.

Overall, there appears to be little research into any correlation between Twitter data and sports match outcomes, notwithstanding the report by Sinha et al [6]. Despite the aforementioned research into ANFL (American National Football League) results and Twitter data, association football and American football have many differences and so this project aims to provide varied and additional insights into sentiment analysis and sports predictions.

2.2 Twitter API

APIs (Application Programming Interfaces) are interfaces that allow applications to communicate with software systems, and are used by API providers and third parties alike. In recent years with the growth of big data [23], companies like Twitter use and promote their APIs to third party developers in order to consolidate their product, in exchange for access to their data. One such API provided by Twitter is their Streaming API which gives users access to a global stream of tweets, and can return a feed of tweets that contain references to user specified subjects, such as football teams.

I chose to use the Twitter Streaming API for this project, as it keeps a persistent HTTP connection open from the request origin to the Twitter Streaming API, and continually sends requests to retrieve tweets that can then be stored in a database. The Twitter Streaming API constantly sends a stream of tweets back to the request origin in a JSON format, and contains a variety of key-value fields [24]. This process is shown in the diagram below.

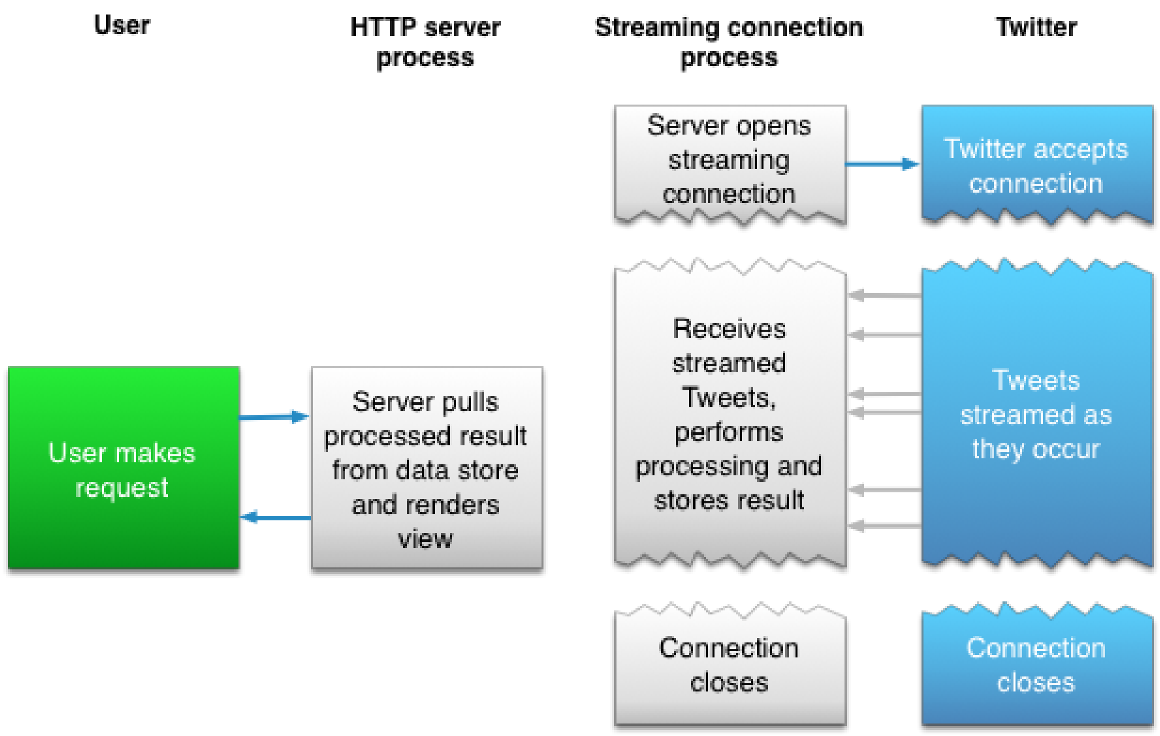
User
HTTP server process
Streaming connection process
Twitter
Server opens streaming connection
Twitter accepts connection
Receives streamed Tweets, performs processing and stores result
Tweets streamed as they occur
User makes request
Server pulls processed result from data store and renders view
Connection closes
Connection closes

Figure 1 - Diagram showing the process sending the request to Twitter Streaming API [25]

I am specifically interested in the "created_at" and "text" fields [24], primarily as I need to separate tweets into different time periods, so that I can perform a sentiment analysis upon the tweet text.

In the request to the Twitter Streaming API, the following application specific tokens will be included, and should be kept private:

- Twitter Consumer Key
- Twitter Consumer Secret
- OAUTH Access Token
- OAUTH Secret

This request shall also include keywords relating to the football teams in the Premier League as parameters in the request to the Twitter Streaming API; these are disclosed in the Appendix 1a.

2.3 MongoDB

To store tweets retrieved from Twitter, I chose to use MongoDB as the database for this project, primarily due to its ease of use and its compatibility with storing, retrieving and exporting JSON data. MongoDB is an open source document-based database, which works well with storing JSON-like data [26]. Finally, the ability to scale the database was particularly important for this project, as whilst I did not anticipate any issues, I did not want to suffer any issues relating to a lack of storage or efficient storage for tweets.

2.4 Twitter-Tap

Twitter-Tap is a Python library that simplifies the process to collect tweets from the Twitter Streaming API to store into a MongoDB instance [15]. I chose to use this library due to its simplicity, and because it supports both the retrieval of tweets from the Twitter Streaming API and storage of tweets into MongoDB, which are components I had already planned to use. Throughout the duration of my project, the Twitter-Tap library worked relatively successfully and saved 33.4 million tweets overall.

2.5 Natural Language Tool Kit (NLTK)

Natural Language Tool Kit, hereafter mentioned simply as NLTK, is a Python library that provides extensive language and text processing methods and tools, and most importantly, implements data classification through the Naïve Bayes classifier [16]. I chose to use NTLK due to its easy implementation and extensive documentation available online, where it is listed as an "amazing library to play with natural language" [16], and secondly, because as I was using Python elsewhere to retrieve and process tweets, consistently using one language reduces the likelihood of problems occurring.

2.6 Keyword Search Terms

In order to retrieve tweets from Twitter that were relevant to the 20 Premier League teams, I used keywords as parameters in the request to the Twitter Streaming API. Each team in the Premier League has an online Twitter presence [27], so for each team I used their Twitter username along with another keyword relating to that team. The other search term for each team was based on a team's name or nickname, taking into account popular Twitter nicknames for teams which are often shortened or abbreviated due to the 140 character constraint on tweets. An example of this is the use of "Man Utd" as the nickname keyword for Manchester United. I chose to use just 2 total search terms for each team, in an attempt to ensure I did not retrieve more tweets for some teams than others.

3 - Text Classification

In order to correctly analyse the sentiment of tweets for this project, with the aim of detecting any correlation between tweet content and football match outcomes, I will incorporate the process of data classification.

Data classification is the problem of identifying the class that unseen data should be categorised into and learning the relationship between data and class variables, specifically "given a set of training data points along with associated training labels, determine the class label for an unlabelled test instance" [11]. There are many different data classification models [28], however, for this project, I will implement a Naïve Bayes classification model; this is discussed further below.

For the purposes of this project, I am concerned with classifying tweets into one of three classes; "Win", "Draw" or "Lose", where the class label is a prediction of the outcome of the next game for the team mentioned in the tweet. In order to build an effective and accurate classifier, the model will need to be trained using data collected for this project.

Data collection started on 8th February, and I chose to use the data taken from this time until 7th March as training data for the first set of tests, which includes 5,386,124 tweets. Subsequent weeks and periods with Premier League matches in after this date are used as test data, where the test data is the data that this project aims to predict, and will incorporate the previous weeks data into the training data. By training the Naïve Bayes model with a months' worth of data, there will be a significant feature set which increases the classifiers accuracy; this is discussed in more depth below.

3.1 Naïve Bayes Classification

The Naïve Bayes text classification method is based on Bayes' theorem, which is detailed below in Figure 2 [29]:

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

The equation above is based on a and b both being events, as we aim to find out the probability of a occurring, given that b has occurred. The Naïve Bayes Classifier aims to find out the predicted classification of a tweet, given the words in that tweet and prior knowledge we have about the frequency of all words the model is trained on, with an associated class label [14].

Relating the equation in Figure 2 to a text classification problem;

$$a = class$$
$$b = terms\ in\ \ a\ tweet$$
$$MAP = maximum\ a\ posteriori$$
$$aMAP = argmax_{a \in A}\ P(a|b)$$

$$aMAP = argmax_{a \in A}\ \frac{P(b|a)\ P(a)}{P(b)}$$

In this example, *MAP* is the most likely classification for this tweet. We drop the denominator P(b) from the equation, as P(b) holds the same value for all classes, and thus will not affect the overall probability if we remove the denominator from all calculations. Therefore, we are left with:

$$aMAP = argmax_{a \in A}\ P(b|a)P(a)$$

At which point, we represent the tweet ***b***, as a set of words $\boldsymbol{x_1 \ldots x_n}$

$$aMAP = argmax_{a \in A}\ P(\boldsymbol{x_1}, \boldsymbol{x_2} \ldots \boldsymbol{x_n}|a)P(a)$$

Therefore, we want to know how often this class occurs given these words, performing the calculation for all classes, and using the most likely class as the classification.

The Naïve Bayes classifier is popular due to its simplicity and strong performance [30], and works particularly well when classifying documents such as tweets [11]. The Naïve Bayes classifier assumes that words (hereafter referred to as features) in a document (hereafter referred to as tweets) are independent of each other, and as such does not take into account sequences of words.

Using the equation in Figure 2 above, A is the class label, i.e. "Win", "Lose" or "Draw", and B is the tweet text. In order to train our model, the training data is labelled with the match outcome dependent upon whether the team mentioned won, lost or drew in the time period the tweet was created. Then, each tweet is split into a set of features, i.e. individual words, and the classifier takes account of the number of occurrences of a feature, alongside the class label across all tweets when training a model. When it comes to classifying and predicting the class label of test data, each tweet is again split into a set of features, and then based on our prior knowledge of feature occurrences and their associated class label, the model will classify this tweet.

Given the classifier's reliance upon feature occurrences in order to classify tweets, there will obviously be an issue if the classifier attempts to classify a tweet containing a feature with little or zero occurrences in the training data. Therefore, the concept of smoothing is introduced to the Naïve Bayes Classifier; this is discussed in more detail below.

The accuracy of the classifier is measured with the recall measure for Win, Lose and Draw, i.e. the likelihood of a tweet being given a class label of "Win", "Lose" or "Draw". It is worth noting that whilst this project aims to gain the highest possible accuracy, just by randomly guessing the match outcome, it is possible to guess correctly 33% of the time, and as such this should be the absolute minimum accuracy accepted.

Given two teams A and B, we want to see the classification probabilities for each team, i.e. how often the tweets for a team are classified as "win", "draw" or "lose". This is accomplished by running the classifier using training data labelled with "win", "draw" or "lose", and test data containing tweets mentioning team A, and then by running the classifier again with the test data mentioning team B. The values generated by these tests are then used to give the true classification value of an outcome; the formulae for these outcomes are listed below.

For sets of tweets mentioning teams A and B we are able to assert that they are independent events, as a tweet claiming that Team A will win does not impact the fact that a tweet might be claiming the contrary regarding Team B. Given this independence of tweets, we are able to multiply these values together to calculate the match outcomes as follows.

P(Team A Win) = Team A Win classification value x Team B Lose classification value

P(Draw) = Team A Draw classification value x Team B Draw classification value

P(Team B Win) = Team A Lose classification value x Team B Win classification value

These 3 calculated values are then used as inputs into the equation in section 5.1.4.

3.2 Smoothing

As previously mentioned, the Naïve Bayes Classifier outputs a predicted class for an inputted tweet based on the features within the tweet, the class labels, and the prior knowledge of features, their frequencies and associated class labels. Therefore, this project needs to account for a situation where a feature has no or low frequencies in the feature set that the model is trained on, so to reduce the risk of badly or inaccurately classifying a tweet this project uses smoothing.

Smoothing eliminates the possibility of getting a 0 classification value from our classification equation, by altering or removing noise in the data [45]. Specifically within the NLTK implementation of the Naïve Bayes classifier, Laplace Smoothing is used, where 1 is added to the numerator and denominator of the Naïve Bayes equation, thus eliminating the chance of getting a classification value of 0 [31].

The equation we use to incorporate Laplace Smoothing is given below in Figure 3:

Where;

$\boldsymbol{x_i}$ is a word in a tweet
$\boldsymbol{a}$ is a class label

$$P(\boldsymbol{x_i}|\boldsymbol{a}) = \frac{count(\boldsymbol{x_i}, \boldsymbol{a}) + 1}{\left(count(\boldsymbol{x}, \boldsymbol{a})\right) + 1}$$

Here, we calculate the classification value based on the number of times a word, $\boldsymbol{x_i}$, appears in the class, $\boldsymbol{a}$, compared to all words, $\boldsymbol{x}$, from all tweets in the class $\boldsymbol{a}$. Whilst Laplace Smoothing can be considered simplistic [43], it is an acceptable solution for this implementation due to the time scale of this project.

3.3 Feature Selection

When the Naïve Bayes classifier is run, it is trained with a minimum of 5386263 tweets, which increases each week as we aim to make our classifier more accurate. As each tweet is labelled with the class label "Win", "Lose" or "Draw", it is also broken down into a set of features, which the classifier then collates and notes for their frequency and associated class label. Therefore, it is highly likely that the feature set has high dimensionality, which may not be optimal when using test data for classification if there are low occurrences of features, however it will significantly reduce the amount of unseen data.

Feature selection is involved in attempting to maximise the accuracy and efficiency of our classifier, by using selected features from the entire feature set. Common applications of feature selection include removing punctuation from documents, as often training our model using features that include punctuation can cause a negative impact when using test data. Another method to implement feature selection can include the removal or modification of certain features, as they do not add any informative value in determining the classification. For example, I chose to replace all team name keywords with a single string "TEAMNAME", as for each team there are varying numbers of tweets that mention them, this negatively impacted upon the accuracy of my results, as shown in the results section.

By carefully filtering out low frequency terms, we are able to reduce our reliance on smoothing, thus helping to make an accurate classification. By sorting our features into their usefulness, that is to say features that are effective in distinguishing between class labels, we are able to select the first x features to see how the accuracy of the classifier is impacted. We are able to view a features usefulness with the show_most_informative_features() function in the NLTK library [32].

The show_most_informative_features() function is based off of the most_informative_features() function, which is used to display the 'most informative' features used by the classifier, where the informativeness of a feature is defined as "equal to the highest value of P(fname = fval|label), for any label, divided by the lowest value of P(fname = fval|label), for any label" [32].

As this function only shows us the usefulness of each feature, the Chi Square feature selection method is implemented to increase the performance of our classifier.

3.4 Chi Square

The Chi Square measure, also denoted as X^2, is used to test the independence of two events, and specifically for feature selection, it is used to test the independence of a term and the occurrence of an associated class label. With regards to my project, the Chi Square distribution is used to determine which features provide the most information gain, so that we can then run the Naïve Bayes classifier on test data using the first x features, where x is equal to 10, 100, 1000, 10000 and 15000.

To implement this feature selection method, we will then use a frequency distribution that takes into account all words, and then a conditional frequency distribution that takes into account words under each class label "win", "draw" or "lose".

To calculate the Chi Square score for each word, we then run the BigramAssocMeasures.chi_sq() function which takes in the following parameters:

(a, (b, c), d)

Where;

a = The frequency of the word for the label
b = The total frequency of the word across all labels
c = The total frequency of all words that occurred for the label
d = The total frequency for all words across all labels

O_i = The observed value, in this example, a

E_i = The expected value, in this example (b * c) / d

$$\mathbf{X^2} = \sum \frac{(O_i - E_i)^2}{E_i}$$

Using these values and for each class label, the BigramAssocMeasures.chi_sq() function calculates an association score between that word and the given class label, and each class label score is summed into one single score. Using this score, we rank the words from highest to lowest score, with the aim of running our classifier on the first x features given above.

3.5 Accuracy

When running our Naïve Bayes classifier implementation, files containing tweets that mention a single team are used as the test data. This project aims to see how many times tweets are correctly classified, and measure the exactness of the classifier using recall, with the aim to optimise the classifier for the future.

For example, to test a file containing tweets that mention Arsenal, we will measure the recall values for the labels “Win”, “Lose” and “Draw”, where;

recall(“win”) + recall(“draw”) + recall(“lose”) = 1

Therefore, the recall values for each label can be used as true percentage values that, in combination with other values, can be used to predict a match’s outcome.

Whilst this project strives to achieve the best possible results, we must consider and accept that Twitter is not a perfect source of data given misspellings, grammatical errors and slang words in the text. Given the extensive data set collected, which can be used as training data and the aforementioned limitations, realistically we can expect to achieve an accuracy of 70-80%. As previously mentioned, we are able to achieve a success rate of 33% purely by randomly guessing. Therefore, we are able to accept the recall value of any class label that has a value larger than the recall values for other class labels, as the outcome prediction.

4 - Predicting Football Outcomes

Association football is one of, if not the most popular sport in the UK [33] and throughout the world, due to its global appeal and universal access with 209 associations affiliated with FIFA today [34]. As such, it is has one of the UK's most popular betting markets [17], which makes it a popular topic to attempt to predict [4][8][9][10]; it is this combination of facts that serve as the motivating factors behind my project.

Alongside using text classification of tweets that mention teams in the Premier League, I am using data from the current 2014/2015 Premier League season to estimate the number of goals that each team in a match will score. I will then use this value in a Poisson Distribution to predict the likelihood of a match outcome for two teams. I will then combine the text classification values, the Poisson Distribution outcome values and betting odds for each match as reported by bookmakers, in order to provide a final prediction outcome for the Team A win, draw and Team B win scenarios.

4.1 Poisson Distribution

The Poisson Distribution model is used to estimate the chances of a variety of events happening, given that we know the average number of times that this event happens. Using this distribution model in order to predict football results has already been undertaken many times [8][9][10], and my work is mostly based on the model created by Ratcliffe [35]. Given two teams, Team A and Team B, their average goals scored and conceded for games played at home and away in the Premier League, we aim to output a percentage for the outcomes of Team A win/Team B lose, Team A and Team B draw and Team A lose/Team B win.

The Poisson Distribution is denoted by the following equation, displayed in Figure 4 below.

$$\mathrm{P}(x;\ \mu) = \frac{(e^{-\mu})(\ \mu^{x})}{x!}$$

Where:

x is the actual number of times an event happens, and

μ is the average number of times an event happens, and

e is Euler's number, equal to approximately 2.71828

With regards to my project;

x is the actual number of goals that are scored, and

μ is the average number of goals that a team will score

The value of x varies between 0 and 8, to calculate the probability of a team scoring that many goals. Whilst the 6 most common scores in the Premier League in the 4 seasons preceding the 2012-2013 season included no team scoring more than 3 goals in a match [36], this 2014-2015 season Southampton beat Sunderland 8-0 and so, therefore, the x value range is increased to represent this possibility.

The value of μ varies taking into account the goals that a team has scored and conceded both home and away, and given two teams A and B and assuming that Team A is at home and Team B is away, is defined as follows:

μ = Average Goals Scored

For Team A,

$$\mu = (Team\ A\ Home\ Attack\ Strength) * (Team\ B\ Away\ Defence\ Weakness) * (League\ Home\ Average\ Goals\ Scored)$$

For Team B,

$$\mu = (Team\ B\ Away\ Attack\ Strength) * (Team\ A\ Home\ Defence\ Weakness) * (League\ Away\ Average\ Goals\ Scored)$$

Team A Home Attack Strength = $\frac{Team\ A\ Average\ Home\ Goals\ Scored\ Per\ Game}{League\ Average\ Home\ Goals\ Scored\ Per\ Game}$

Team B Away Attack Strength = $\frac{Team\ B\ Average\ Away\ Goals\ Scored\ Per\ Game}{League\ Average\ Away\ Goals\ Scored\ Per\ Game}$

Team A Home Defence Weakness = $\frac{Team\ A\ Average\ Home\ Goals\ Conceded\ Per\ Game}{League\ Average\ Home\ Goals\ Conceded\ Per\ Game}$

Team B Away Defence Weakness = $\frac{Team\ B\ Average\ Away\ Goals\ Conceded\ Per\ Game}{League\ Average\ Away\ Goals\ Conceded\ Per\ Game}$

Taking into account the above equations, we calculate the Poisson Distribution using the μ, e and x values, for each x value from 0 to 8. These values are then summed together to give the summed percentages of the Team A win, draw and Team B win scenarios.

For example to calculate the probabilities for the match Man Utd vs Swansea, we compute the following:

League Average Home Goals Scored Per Game: 1.574

League Average Home Goals Conceded Per Game: 1.195

League Average Away Goals Scored Per Game: 1.195

League Average Away Goals Conceded Per Game: 1.574

Man Utd Average Home Goals Scored Per Game: 1.526

Man Utd Average Home Goals Conceded Per Game: 1.105

Swansea Average Away Goals Scored Per Game: 1.105

Swansea Average Away Goals Conceded Per Game: 1.474

Man Utd Home Attack Strength = $\frac{1.526}{1.574}$ = 0.97

Man Utd Home Defence Weakness = $\frac{1.105}{1.195}$ = 0.93

Swansea Away Attack Strength = $\frac{1.105}{1.195}$ = 0.93

Swansea Away Defence Weakness = $\frac{1.474}{1.574}$ = 0.94

Man Utd μ = 0.97 * 0.94 * 1.574 = 1.44

Swansea μ = 0.93 * 0.93 * 1.195 = 1.03

This project then implements a Poisson Distribution, where the value of *x* varies between 0 and 8, and using the μ value above for each team, we calculate the probability of that team scoring *x* goals, where P(X = *x*) = 1

We then predict the probability of a given match outcome occurring, depending on the value provided from the Poisson Distribution for each team scoring *x* goals. For example;

P(Man Utd 1-1 Swansea) = (Man Utd value of scoring *x* = 1 goal) * (Swansea value of scoring *x* = 1 goal)

Extending this for all match score scenarios, where *x* varies from 0-8 for both teams, it is then possible to predict the likelihood of any score with a maximum of 8 goals for each team.

Therefore, the probability of:

Man Utd winning = The summation of all score probabilities where Man Utd wins,
Man Utd and Swansea drawing = The summation of all score probabilities where the teams draw,

Swansea winning = The summation of all score probabilities where Swansea wins

All workings for this project are calculated and displayed in the attached Appendix 3.

4.2 Match Odds

This project will make use of betting odds for outcomes of each football match in the Premier League as a contributing factor in our final equation to predict football matches. Whilst betting companies lower the risk involved with offering bets on the outcomes of football matches by marginally increasing the odds on these outcomes, it is still acceptable to use these betting odds as a basis from which it is possible to predict the outcomes of matches. Although betting companies offer odds to customers that are in the betting companies' favour, the extensive underlying mathematics that produce the odds originally are representative of that betting companies' confidence in an outcome for a match.

The actual betting odds used for this project are taken from Odds Portal, and for each outcome there are average odds from each featured bookmaker on Odds Portal [37]. The full details of odds for each match are available in Appendix 4.

4.3 Combination of inputs

Once I have collected results from the Naïve Bayes Classifier and the implemented Poisson Distribution model and collected the match odds from the bookmakers, I will combine these three inputs to provide for each match outcome, a probability of that outcome occurring.

The overall equation used to predict the outcome of a football match incorporates calculations for win, lose and draw scenarios for both Team A and Team B from the Naïve Bayes classifier, the Poisson Distribution and collected betting odds. This equation applies a weighting to each of these three inputs, and then adds them together to calculate a probability. The calculation of this equation is displayed below.

P(Team A Win) = P(Team B Lose)

P(Team A Draw) = P(Team B Draw)

P(Team A Lose) = P(Team B Win)

P(Outcome) = ax + by + cz

Where;

a = Poisson Distribution weighting

b = Odds weighting

c = Classification result weighting

x = Poisson Distribution value for outcome

y = Odds value for outcome

z = Classification value for outcome

a + b + c = 1

The exact values for the weightings a, b and c are explained and justified in the Results section **7.1.4**.

5 - Specification, Design and Approach

5.1 Overview

This project consists of four main sections; a Twitter Scraper that collects tweets from the Twitter Streaming API, a Naïve Bayes classifier implementation that takes in tweets collected using the Twitter Scraper, a Poisson Distribution using Premier League data and then a final formula that uses the outputs of the Naïve Bayes classifier and the Poisson Distribution are combined in an overall equation to give a decisive prediction on whether a team will win, lose or draw.

These four sections are separate but sections can be dependent upon each other, as the Naïve Bayes classifier is dependent upon using tweets collected from the Twitter Scraper, and the results of the Naïve Bayes classifier and the Poisson Distribution are used in the overall prediction equation. The Twitter Scraper and Naïve Bayes classifier implementation are shown in Figure 5 below, with the Poisson Distribution and final formula shown in Figure 6.

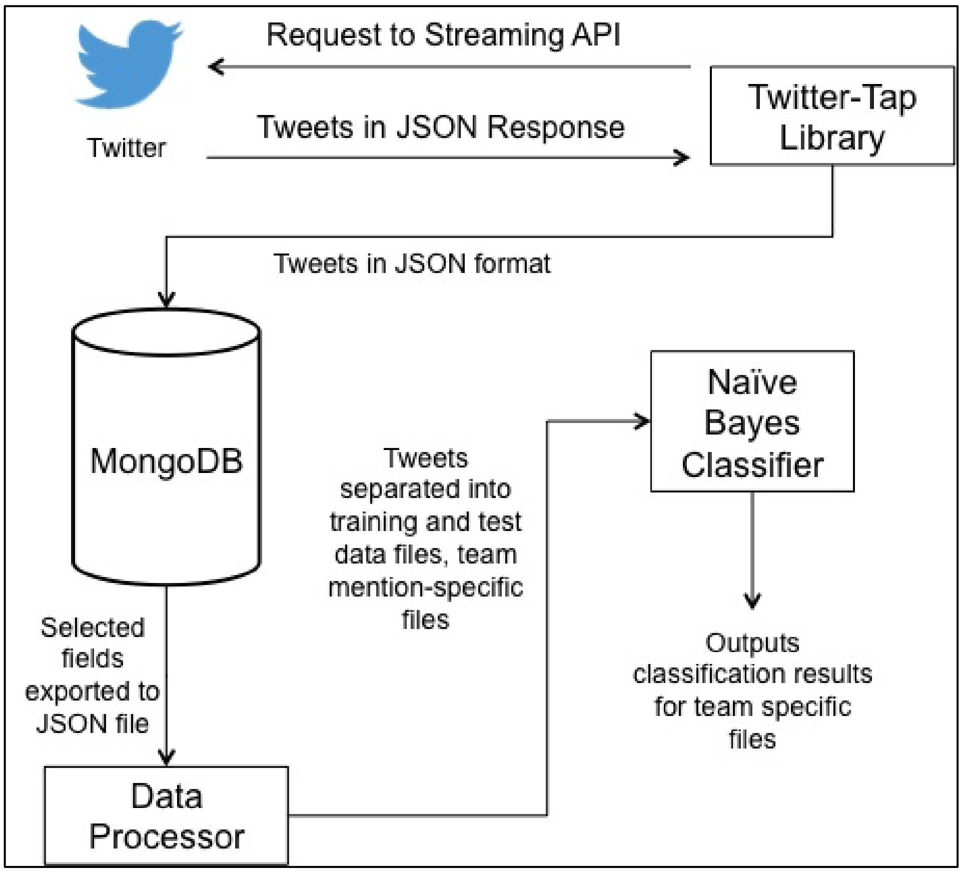
Request to Streaming API
Twitter
Tweets in JSON Response
Twitter-Tap Library
Tweets in JSON format
MongoDB
Tweets separated into training and test data files, team mention-specific files
Naïve Bayes Classifier
Outputs classification results for team specific files
Selected fields exported to JSON file
Data Processor

Figure 5 – Twitter Scraper and Naïve Bayes Implementation

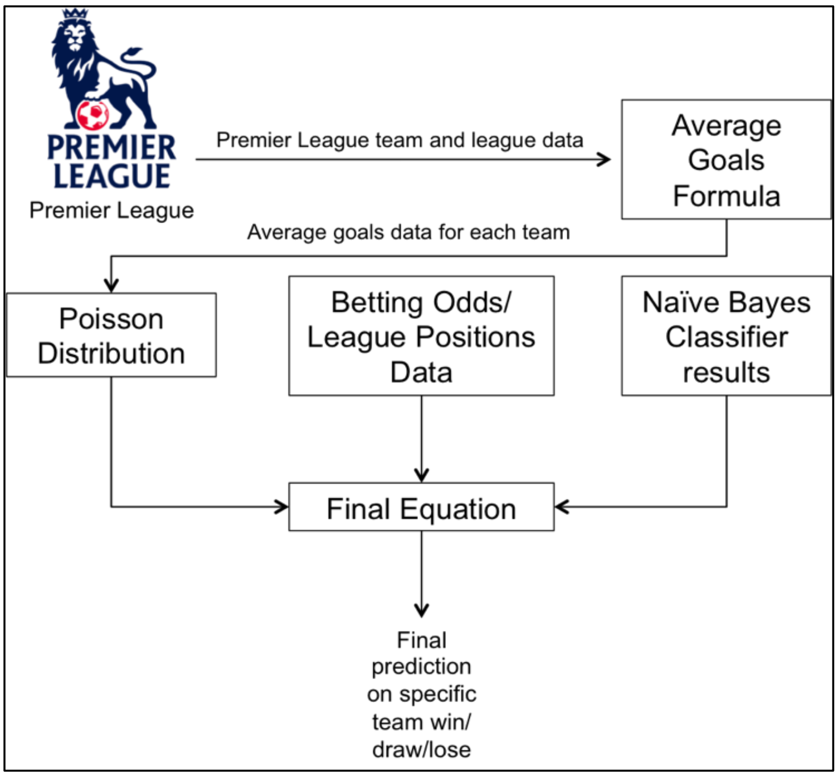
PREMIER LEAGUE
Premier League
Premier League team and league data
Average Goals Formula
Average goals data for each team
Poisson Distribution
Betting Odds/ League Positions Data
Naïve Bayes Classifier results
Final Equation
Final prediction on specific team win/ draw/lose

Figure 6 – Poisson Distribution and Input to Overall Equation

5.1.1 Twitter Scraper

The Twitter Scraper works by accessing the Twitter Streaming API through a request from the Twitter Tap library for Python, which then listens for tweets that contain any keyword from the list of keywords. These tweets are returned in JSON and then stored in a MongoDB instance. An example of all of the JSON fields returned for a single tweet can be found on Twitter's website [42].

Whilst this current implementation of my project requires only the creation date and text content of a tweet, I chose to store all JSON information relating to a tweet so that I could future proof any extension to my project. I then extracted the "created_at" and "text" attributes for each tweet from the MongoDB database, at which point these JSON files were processed. This processing stage involved separating the multiple JSON files into different files according to the values of the "created_at" for each JSON object, checking that words did not mention more than 2 teams by filtering through an extended list of keywords relating to teams (available in Appendix 1b at the bottom of this document), and then the JSON objects for each team that it mentioned. This produced a file for each team in each time period, which upon the results of early tests, replaced all mention of a team's name with "TEAMNAME"; this is discussed further below. This data is then used in the Naïve Bayes classifier.

5.1.2 Naïve Bayes Classifier

The Naïve Bayes Classifier implementation in this project used a total of 5 different files;

- TRAIN_WIN_FILE
- TRAIN_DRAW_FILE
- TRAIN_LOSE_FILE
- TEST_WIN_FILE/TEST_DRAW_FILE/TEST_LOSE_FILE
- FULL_WIN_FILE/FULL_DRAW_FILE/FULL_LOSE_FILE

The TRAIN_WIN_FILE, TRAIN_DRAW_FILE and TRAIN_LOSE_FILE references all referenced files that are used for training data. As noted earlier, the data collection for this project started 8th February, and the first set of data used for testing was collected after 7th March. These training data files were split into different time periods, and all tweets mentioning teams that won/drew/lost in that time were collated into a single file for wins, draws and losses. This collection and collation of data continued throughout the project lifecycle.

The TEST_WIN_FILE, TEST_DRAW_FILE and TEST_LOSE_FILE references all reference files that are used for test data; these files only relate to one time period. Similar to the training data files, all teams that won/drew/lost in this time period were collated into one file named "winningTeams", "drawingTeams" and "losingTeams".

The FULL_WIN_FILE, FULL_DRAW_FILE and FULL_LOSE_FILE references were used to reference the full documents of tweets that should be used for the Chi Square test on our feature set. Depending on whether the team that we are attempting to classify the tweets of won/lost/drew, the tweets mentioning that team are appended to the TRAIN_WIN/LOSE/DRAW_FILE, whilst the other two outcomes reference only the training data files.

For example, if Arsenal had won in a time period then we would append the Arsenal.txt file to the TRAINING_WIN_FILE, in order to make a FULL_WIN_FILE, whilst the FULL_DRAW_FILE and FULL_LOSE_FILE would reference the TRAINING_DRAW_FILE and TRAINING_LOSE_FILE documents accordingly.

The Naïve Bayes Classifier then took these files as an input, calculated the probabilities and then outputted classification values for the class labels "Win", "Draw" and "Lose". These values are then used in the final equation.

5.1.3 Poisson Distribution

The Poisson Distribution calculation uses data that has been collected from the Premier League competition over a 9 month period. This data was noted and collated in the attached Appendix 3. The Poisson Distribution then outputs predicted values for each teams, based on the number of goals it expects that team to score, and these values are used in the final equation.

5.1.4 Final Equation

As previously explained, the final equation takes inputs from the Naïve Bayes Classifier, the implemented Poisson Distribution model and the match odds for each match, and applies a weighting to each of these inputs to make an overall prediction on the outcome of each match. In order to ensure the optimal weighting values are selected, multiple tests were conducted with varying weighting values, where the accuracy of these weightings was measured by counting the number of games, where our models predicted outcome actually occurred; these tests are described in the Results section 7.4.

5.2 Project Requirements

This project has a set of different requirements that should be met, in order to be able to accurately predict football results using text classification, Premier League data and betting odds. These requirements have been split into functional and non-functional requirements; these are detailed below.

5.2.1 Functional Requirements

5.2.1.1 Twitter Scraper

- Needs to collect tweets mentioning teams, according to the keyword list.

- Needs to store these tweets in a MongoDB instance as they are collected.

- Needs to be stable to easily deal with simple errors over a long period of time (i.e. exceeding rate limits by sending a new request).

- Needs to be able to accurately process JSON files to distinguish between different teams, and to filter tweets that mention > 1 team.

5.2.1.2 Naïve Bayes Classifier

- Needs to use Chi Square test on features, to then run Naïve Bayes Classifier based on the top 10000 and 15000 ranked features.

- Needs to calculate and output classification probabilities for team-specific file inputs, for the class labels “Win”, “Draw” and “Lose”.

5.2.1.3 Poisson Distribution

- Needs to calculate average goals expected for home and away teams based on collected Premier League data.

- Needs to use and incorporate new data collected over a period of time.

- Needs to output likelihood percentages for Team A win, draw and Team B win, given any two teams, Team A and Team B.

5.2.1.4 Final Equation

- Needs to collect betting data on all Premier League matches over a 2 month period, for all outcomes in a match.

- Needs to incorporate betting data, Poisson Distribution output and Naïve Bayes Classifier output to predict match outcome given two teams, Team A and Team B.

5.2.2 Non-Functional Requirements

- The Twitter Scraper should collect, store and retrieve the tweets efficiently.

- The Naïve Bayes classifier should not take longer than 30 minutes to run each test for a set of data.

- All aspects of the program should be well documented and formatted.

6 - Implementation

This section of my report will cover all areas of the implementation of the project, explaining and analysing the different steps involved in data collection, processing, classification and combination to produce a final output.

6.1 Twitter Data Collection

In order to effectively classify tweets with "win", "draw" and "lose" labels, it was clear that I would be required to use an extensive data set to train my classifier with. Although the Twitter Searching API does allow users to view and collect past tweets, it is noted in the Twitter documentation that this should not be used for “completeness” [38] and recommends using the Twitter Streaming API to get a better snapshot of all tweets.

Therefore, given that I knew I needed to use the Streaming API to retrieve the tweets, and MongoDB to store the tweets, I set about finding a library that was compatible with both of these services and would simplify the process for myself. I used the Twitter-Tap library for Python [15], which required minimal installation of components and automated the process of retrieving and storing tweets. However, Twitter-Tap was optimised for versions of Python 2, and initially there were small problems configuring the program to work on my machine which was running Python 3. After making small adjustments to the underlying code, I was able to successfully run commands using Twitter-Tap.

The command to be run on my machine took the following format:

```
tap stream --consumer-key CONSUMERKEY --consumer-secret CONSUMERSECRET --access-token
ACCESSTOKEN --access-token-secret ACCESSTOKENSECRET --track "keywords" -v DEBUG
```

By creating an application request on Twitter’s website, I was supplied the following parameters by Twitter:

--consumer-key

--consumer-secret

--access-token

--access-token-secret

The parameter "track" allows the user to specify keywords that they want the retrieved tweets to contain. For my project, I used 40 keywords related to the 20 teams in the Premier League; these are listed below in Appendix 1a at the bottom of this document.

The data collection process was a continuous process that began on 8th February 2015 and finished on 19th April 2015. In this period there were some, but relatively few connection outages, which hindered my efforts to collect data from Twitter. However, I altered the code so that the program would send a new request to Twitter if the rate-limit of collected tweets was exceeded, and as such was unable to retrieve any more tweets. Altering this code was essential to the success of my project, as I was often away from my data collection machine for long periods and ensured the data collection process was very efficient.

6.2 Twitter Data Processing

From the beginning of the project, I was unaware of the final scope of what I would achieve in this project and chose to be proactive and not reactive. Therefore, I stored all fields for each tweet collected from Twitter in the MongoDB database. I quickly decided however, that the only information I would need to use would be the "created_at" datetime field which explains when the tweet was actually created and the "text" string field, which contained the text content of the tweet. I then exported these fields from MongoDB into a JSON file, and then sent this file to a different machine for processing.

To process the data effectively, I needed to separate the data using the "created_at" attribute into different time periods where each team only played once in this time period, so I could rely on the assumption that a tweet sent in this period mentioning a team was related to that team's outcome in the Premier League match this week.

The periods that I separated the data into are listed below. Please note that the times are listed in the format YYYYMMDD HHMMSS for organisational simplicity:

20150208 211934 to 20150211 200000
20150216 180350 to 20150222 161500
20150223 000000 to 20150301 140500
20140301 200000 to 20150304 200000
20150305 000000 to 20150307 150000
20150310 000000 to 20150316 200000
20150317 000000 to 20150322 160000
20150401 000000 to 20150406 200000
20150406 000000 to 20150407 194500
20150408 000000 to 20150413 200000
20150414 000000 to 20150419 160000

For simplicity relating to the remainder of my projects, I have renamed these time periods as follows:

20150208 211934 to 20150211 200000 = Week 1
20150216 180350 to 20150222 161500 = Week 2
20150223 000000 to 20150301 140500 = Week 3
20140301 200000 to 20150304 200000 = Week 4
20150305 000000 to 20150307 150000 = Week 5
20150310 000000 to 20150316 200000 = Week 6
20150317 000000 to 20150322 160000 = Week 7
20150401 000000 to 20150406 200000 = Week 8
20150406 000000 to 20150407 194500 = Week 9
20150408 000000 to 20150413 200000 = Week 10
20150414 000000 to 20150419 160000 = Week 11

Within each time period, there can be multiple different kick off times for teams, as matches are scheduled to commence at different time for TV broadcasts. As I wanted to detect sentiment in tweets from before each match commenced, I further separated the files according to the kick offs of different teams; these details are available in Appendix 2.

After separating these files into different time periods, I processed the text content of the tweets in an attempt to reduce noise and improve accuracy in my data. This data processing included removing new line characters, which stopped my program from assuming each new line was a new tweet, changing the character case of the tweets to lower case, and ensuring tweets only mentioned 1 team using a set of keywords used to refer to Premier League teams; these are available in Appendix 1b.

Each text file was then separated into an additional set of files, only containing tweets that mentioned different teams, and this was done for each Premier League team. I then further chose to detect the impact that the presence of team names in tweets had on classification results. As such, I created an additional set of files that replaced team names with "TEAMNAME", and ran the tests on these files to note any differences in results.

I then separated my data sets into both training and test data, where the training data was data collected from weeks 1-5, and the test data was data collected from weeks 6-11. The training data files for winning, losing and drawing were appended text files for each team that won, drew and lost in each period in the weeks 1-5. For each subsequent week I used for test data, I added the previous weeks data to my training data; this helped to improve the accuracy of my Naïve Bayes Classifier, and helped the classifier become a learning model.

I then ran tests for each team in each week from weeks 6-11, using the sentiment analysis project conducted by Andy Bromberg as a basis for my project [40]. Using Bromberg's project was useful as the underlying objectives achieved in his project were similar to mine and it made sense not to reinvent the wheel. These tests initially took too long a period of time to run, as I was running the classifier using all features collected from all tweets, and then after applying the Chi Square test to rank the features, ran the classifier using the first 10, 100, 1000, 10000 and 15000 words. As discussed further in the Results section, I quickly found I was able to achieve optimal accuracy from using the Chi Square top ranked 10000 features, reducing the run time from around 440 minutes to 25 minutes for each test. The results from the Naïve Bayes Classifier tests were then stored for later use in the overall equation.

6.3 Poisson Distribution

Alongside collecting data from Twitter, I also collected data on the Premier League table and match results for each week. This process was relatively straightforward, as a lot of the work could be automated and required little manual input. Issues relating to this section of the project were primarily to do with perceived inaccuracies of the data set. As some teams played fewer Premier League games than others because of matches in other competitions that had priority over the Premier League games, and the fact that the Poisson Distribution implementation is reliant upon using average goals scored and conceded for each team and the league as a whole, it is possible that the data set is not seen as being as fair as possible. However, as this data is based on averages that will not change greatly at this stage of the season, it could be justified as a non-issue.

This data was stored in an Excel document, and was used to calculate the values required in the Poisson Distribution, which were in turn used as inputs to the final equation.

6.4 Betting Odds

The final set of data that I collected were betting odds related to the matches in the Premier League. Whilst I initially collected data using the web-scraping platform Kimono [39] from the bookmakers William Hill, on occasion there were issues with the Kimono platform, which resulted in some odds not being collected. I decided to use an alternative source, Odds Portal, as they show the average odds at the time of kick off for each outcome for Premier League matches offered from its range of 17 bookmakers. Using average odds from 17 bookmakers offers my project the chance to use expert insight and prediction from a range of bookmakers, and Odds Portal made it easy to source historic data.

A key assumption that needs to be made regarding the odds is that we are assuming the betting companies have added the same profit margin across all outcomes, and haven't for example, added a bigger profit margin to "win" odds than "draw" odds. If these profit margins were different in any way, then we would not be able to detect what the actual probability was, as the results would be erroneous.

7 - Results

In this section, I am going to discuss the results from tests run throughout the course of, and for the purpose of this project. This section will be split into sub-sections relating to results from the Naïve Bayes Classifier, the Poisson Distribution, the Match Odds and then an equation that combines these three inputs to provide a definitive output prediction for win, draw and lose outcomes for matches in the Premier League.

7.1 Twitter Scraper

I was able to successfully set up the Twitter Scraper for my project, which collected a total of 33,464,517 tweets from 8th February to 20th April 2015, where teams in the Premier League were mentioned. As previously noted, data processing and filtering was required to remove tweets that did not meet the criteria for this project, and as such were unable to be used for classification. Therefore, a total of 13,651,906 tweets were used in this project as both training and test data. The number of tweets relating to each team varied wildly, and these are available in Appendix 5.

7.2 Naïve Bayes Classifier

Overall, this project's implementation of the Naïve Bayes Classifier worked very well and I was able obtain classification values for each team that played in each week, from weeks 1-11. For the first set of tests, where the test data was collected in week 6, I ran tests on tweets for each team that contained the teams name and were anonymised (i.e. each team's name mention was changed to "TEAMNAME"), and ran the Naïve Bayes Classifier using the following feature sets:

- Using all features from all collected tweets
- After running Chi Square test, the first 10 ranked features
- After running Chi Square test, the first 100 ranked features
- After running Chi Square test, the first 1000 ranked features
- After running Chi Square test, the first 10000 ranked features
- After running Chi Square test, the first 15000 ranked features

In Figures 7,8 and 9 below, using the knowledge that we now have in retrospect of each teams outcome in the Premier League, I have taken the predicted probability of that outcome occurring for each team, averaged it for each number of features used, and graphed the results. As we can see, where x is the number of different features used for classification, the varying values of x provide an insight into the fluctuating accurateness of the classifier. It is important to note that this is the likelihood of that match outcome occurring from one teams' data, and does not use two teams' data in combination to make a prediction.

Figure 7 below shows the results of tests run using week 6 data that include the teams name within the tweets. As we can see with the varying number of features, results are generally inconsistent and the average correct match outcome percentage is at its highest when using 10 features, sharply decreases when using 100 features, and slowly increases again when using 1000, 10000, 15000 and all features. When looking at the data set available in Appendix 4 however, we can also see that for some teams, the classifier achieved ~99% prediction of the correct outcome, which is an extremely unlikely scenario for a football match.

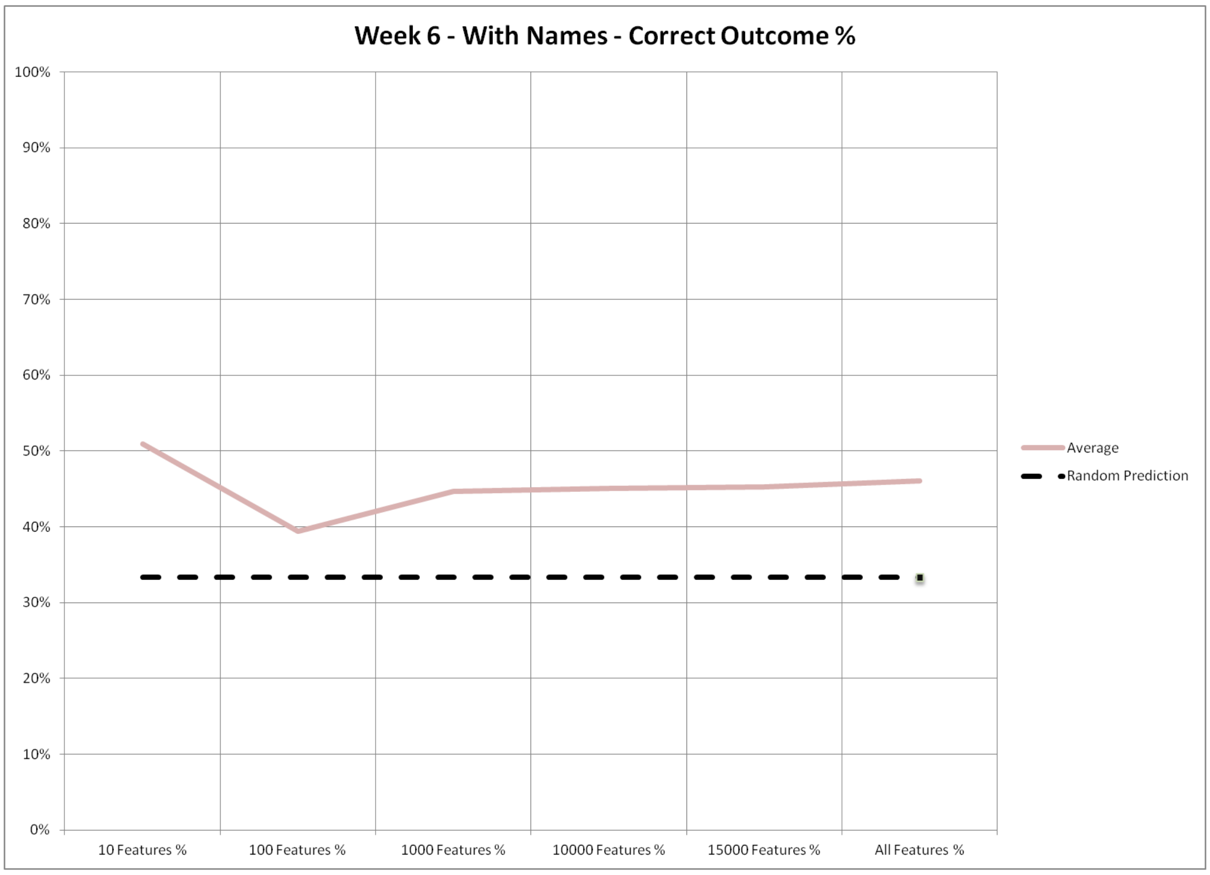

Figure 7 - Graph showing average correct outcome % likelihood in week 6 for all teams

In contrast Figure 8 below shows tests being run on the same week 6 data, which differs in that the tweets were made anonymous, and we can see a difference in the data generated by the Naïve Bayes Classifier. There is not such a high % prediction when using the top 10 ranked features to classify the data. However, after 100 features, when more features are used, a higher predicted percentage for that outcome occurring can be achieved. Again, when looking at the data set in Appendix 4, there are three teams who drew their match in week 6 and the predicted percentage of this outcome occurring was less than 10% for each of these teams. This could have significantly affected the average, and as these results are generally repeated throughout the course of this project, it highlights the inability of the Naïve Bayes Classifier to correctly classify matches that are drawn.

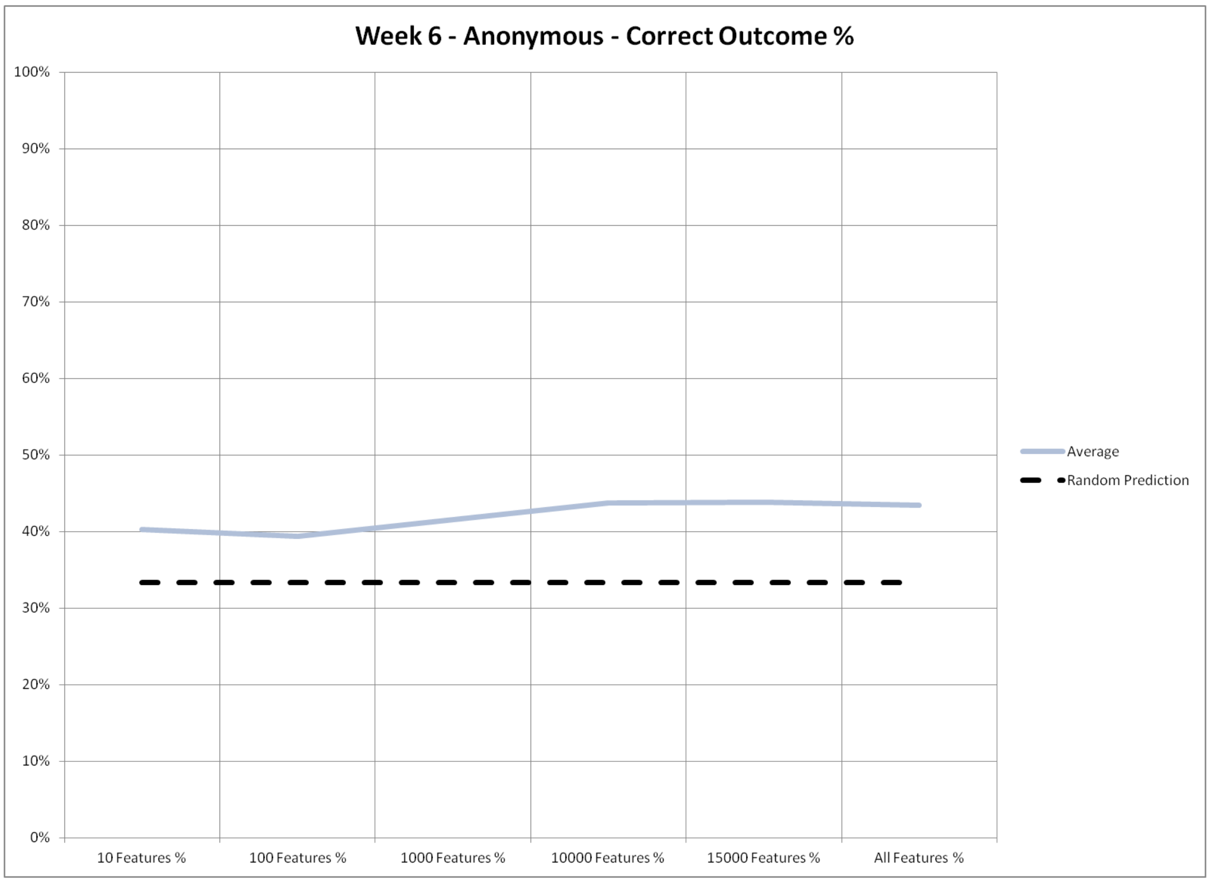

Figure 8 - Graph showing average correct outcome % likelihood in week 6 for all teams with anonymised data set

In order to run my tests more efficiently, I compared the results from the anonymised data set to those from the normal data set with the intention of only choosing one data set to use in these tests from here onwards. I compared the number of correct outcomes that we were able to get from each data set, that is to say for each team, whether the highest percentage outcome actually happened.

Features Used	Anonymous correct outcomes	Anonymous correct percentage	With name correct outcomes	With name correct percentage
All Features	11	55%	11	55%
Top 10 ranked	8	40%	10	50%
Top 100 ranked	9	45%	7	35%
Top 1000 ranked	9	45%	9	45%
Top 10000 ranked	11	55%	10	50%
Top 15000 ranked	11	55%	11	55%

Table 1 – Showing accuracy percentage to predict outcome for week 6 data

We can see that the data set with team names is noisier, and varies from 35-55% correctness, whilst the anonymous data set varies from 40-55% correctness. Whilst the correct outcome percentages in Figures 7 and 8 show that the data set with team names has a higher accuracy percentage, I attribute this to some teams being able to achieve a 99% outcome prediction, which as mentioned is extremely unlikely, and therefore I decided to run the remainder of tests for this project with anonymised data sets.

Additionally, due to the positive results from using 10000 and 15000 features to classify the data, I decided to further explore the accuracy levels from using these amounts; these results are detailed in Table 2. As we can see the results are generally the same for each week except for week 7, where using 10000 features yielded 60% accuracy as opposed to using 15000 features, which produced 50% accuracy; Figure 9 shows values from the classifier when using 10000 features. It is important to acknowledge that whilst 55% accuracy in predicting the correct match outcome is not as high as initially hoped, it is reasonable to expect the classifier to perform better as it learns from more data that it uses.

Week	Top 10000 features correct	Top 15000 features correct	Games Played
6	55%	55%	10
7	60%	50%	10
8	60%	60%	10
9	0%	0%	1
10	65%	65%	10
11	57%	57%	7

Table 2 - Accuracy in predicting correct outcomes in all weeks anonymised data

In Figure 9 below, we can see how the average classification predicted outcome % has changed over time as the model has consumed and learned from more data. This graph displays results from weeks 6-11 inclusive when using 10000 features to classify our anonymised data set. We can see that for week 7, the average % sits just above 50% when using both 10000 and 15000 features, and this % drops for the remaining weeks data. Given the classifiers poor performance in detecting draws in football matches, it could be argued that this highest average % in week 7 is a result of no draws occurring in this period, whereas in weeks 8, 9 and 10 there was at least 1 draw.

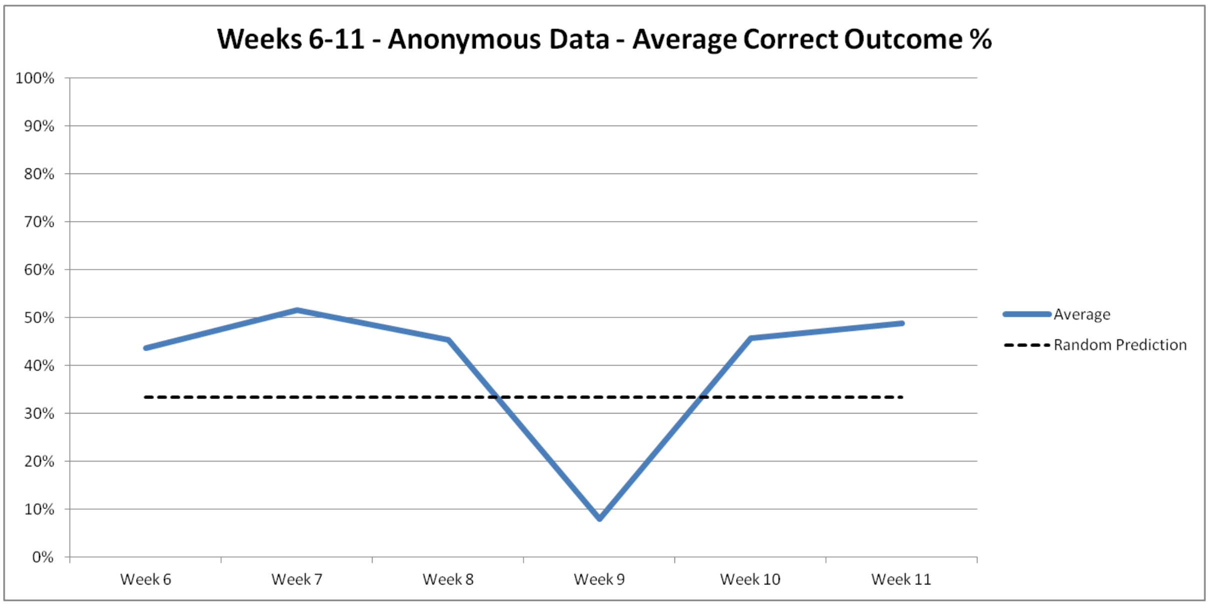

Figure 9 - Graph showing correct outcome % likelihood in week 7 for each team with anonymised data set

In the week 8 period this issue is emphasised; when looking at the correct outcome percentages, two of the three lowest values were for Burnley and Tottenham, who played each other and drew in that week; the average percentage subsequently decreased to ~45%. Similarly, in week 9, the only match played was a draw between QPR and Aston Villa, consequently producing the low average classification result. Likewise, in week 10, 4 of the bottom 5 teams with the lowest predicted classification % values drew in that week, resulting in a lower average classification.

In the period from weeks 6-11 there were only 6 draws in a total of 48 matches, which meant that it was difficult to gain additional features about tweets for teams that draw, which would help the accuracy of the classifier. These low classification results underline the need to use additional inputs in order to compute the actual likelihood of a draw occurring given two teams, such as the Poisson Distribution and betting odds.

As Figures 7, 8 and 9 show an average of the outcome prediction for each team that fluctuates around the 50% mark, it is not truly indicative of the number of match outcomes that this model is able to predict. Therefore, in Figure 10 below, the average values for the actual match outcome occurring has been graphed, alongside the percentage of correct outcomes that this model was able to predict, for each week.

As we can see, with the exception of week 9, where there was only 1 match played and it was not possible to predict the outcome given both teams data, the % of the games that can be predicted is higher than the average %, predicting the correct outcome each week. This indicates that whilst the average correct outcome % displayed in Figures 7, 8 and 9 above is indeed important, and could be described as indicative of the amount of team outcomes that can be correctly predicted, the number of outcomes that can actually be predicted is not related to the average correct outcome value. For example, in Week 10 we can see that it is possible to predict 65% of the teams outcomes, but the average correct outcome value is only 45.6%. However, it must again be noted that we are accepting the highest % of all outcomes for each teams classification results as the prediction for the match outcome, and that theoretically, this project can accept any value above 33.3%, given that the other outcome values are less than this.

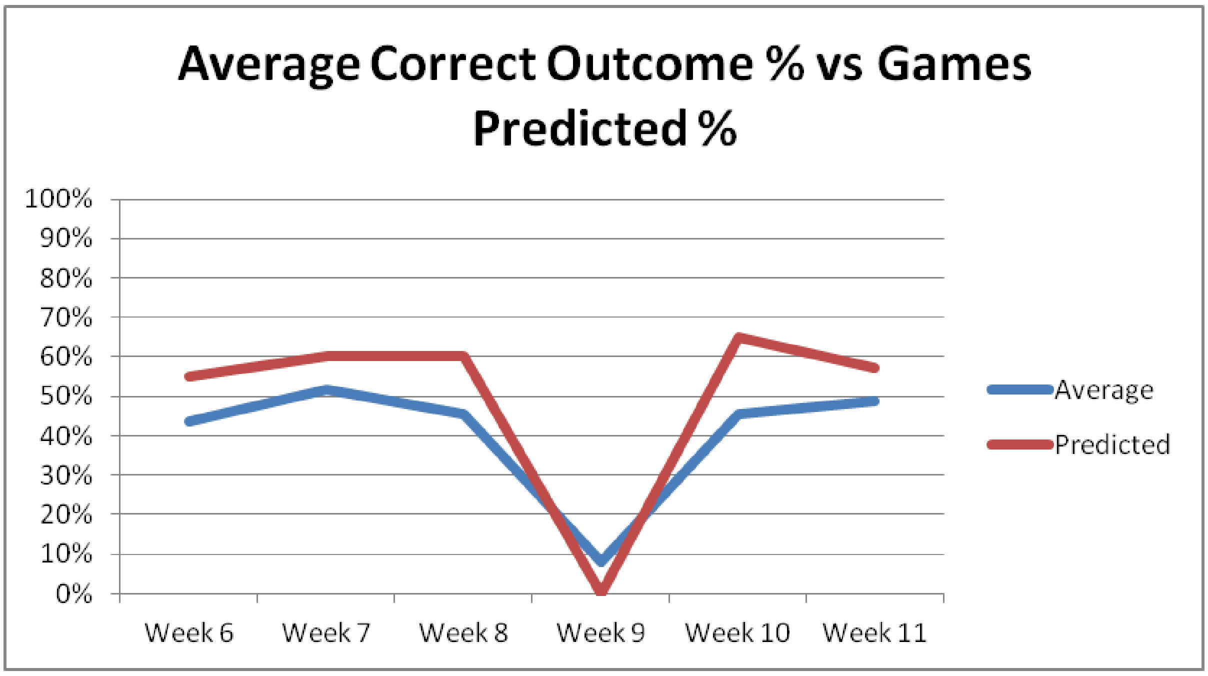

Figure 10 – Average correct outcome percentage vs correct prediction percentages using 10000 features

In addition to detecting the accuracy of "Win", "Draw" and "Lose" classifications for a set of tweets when running tests, I used a function described earlier in this report, the most_informative_features() function. This function outputs a list of features that are most important in distinguishing between the different class labels. Full outputs are available in Appendix 5, but below in Table 3 is a sample of the top 10 most informative features in the Naïve Bayes Classifier after applying the Chi Square test, when classifying data for Man Utd in week 11.

Most Informative Features	Class Labels Ratios	Ratio Values
085658595202	draw : lose	19226.6 : 1.0
Hargapromo	lose : draw	17360.1 : 1.0
Kiossoccer	draw : lose	16918.4 : 1.0
085763063588	draw : lose	16913.3 : 1.0
231322d3	draw : lose	16913.3 : 1.0
27d71eb9	win : draw	15631.6 : 1.0
25e47881	draw : lose	11323.0 : 1.0
085648746716	win : draw	10523.8 : 1.0
2ab09477	win : draw	10523.7 : 1.0
Juragansoccerid	draw : lose	8242.1 : 1.0

Table 3 – Output from most_informative_features() function on Man Utd week 11 data after Chi Square test

The table shows a list of features, along with class labels and the ratio values of those class labels. For example, the string '085658595202' appears in tweets that have been labelled with "draw" 19226.6 more times than tweets that have been labelled with "lose". By searching for these terms in my collection of tweets, it is clear that these terms appear in spam tweets, and interestingly, 6 of these 10 features are used to distinguish tweets with a "draw" label from other class labels. Therefore, it could be possible to associate the poor performance of labelling tweets that should be "draw" with the presence of these spam features. It is possible that these spam features could be associated with popular teams that may have drawn in weeks 1-5, and that popular teams have not drawn as much in weeks 6-11. Additionally, it seems a high number of the features associated with the "draw" label are spam. This is an area that I could conduct future research into, along with advanced feature selection and filtering to remove features or tweets that are spam laden.

The above results are encouraging signs that the Naïve Bayes Classifier can be detected as accurate, however they are all related to the individual teams outcome, not the match outcome.

I was able to find out the true prediction outcome of matches concerning two teams, where;

P (Team A Win) = P(Team A Win) * P(Team B Lose),
P (Team A/B Draw) = P(Team A Draw) * P(Team B Draw), and
P(Team A Lose) = P(Team A Lose) * P(Team B Win)

Using these values, this model was able to predict a total of 28 matches from a total of 48 Premier League matches, which equals 58.33%. This value is an acceptable value, and proves that it is possible to correctly predict the outcome of more games than not. Additionally it provides a strong basis from which the results of the Poisson Distribution and the Betting Odds can be added.

7.3 Poisson Distribution

Given the statistical nature of the Poisson Distribution, and that the results from this are based on average goals scored and conceded throughout the Premier League and for each team, these results were initially expected to be closely related to the football match outcomes. Throughout weeks 6-11, the Poisson Distribution model was able to correctly predict the match outcome of 27/48 matches, which equals 56.25%.

Similar to the results from the Naïve Bayes Classifier, the Poisson Distribution did not predict any outcomes of draws at all, as the prediction % for each match were generally quite low, such as those shown for Aston Villa and QPR in Week 9.

The graph in Figure 11 below shows for all 27 matches where the highest % value is used to predict the outcome of the match and displays the percentage value for this outcome. It is important to note that the graph is sorted in time order i.e. the markers read from left to right, oldest to most recent. We can see in this graph that it is possible to predict the outcome of these matches to a very high percentage, and that only 5 matches were predicted with a percentage of less than 45%, and that generally the results tend to get higher over time.

The full results from the Poisson Distribution are available in Appendix 3; the success rate of 56% again suggests that this could be a useful input in predicting football match outcomes.

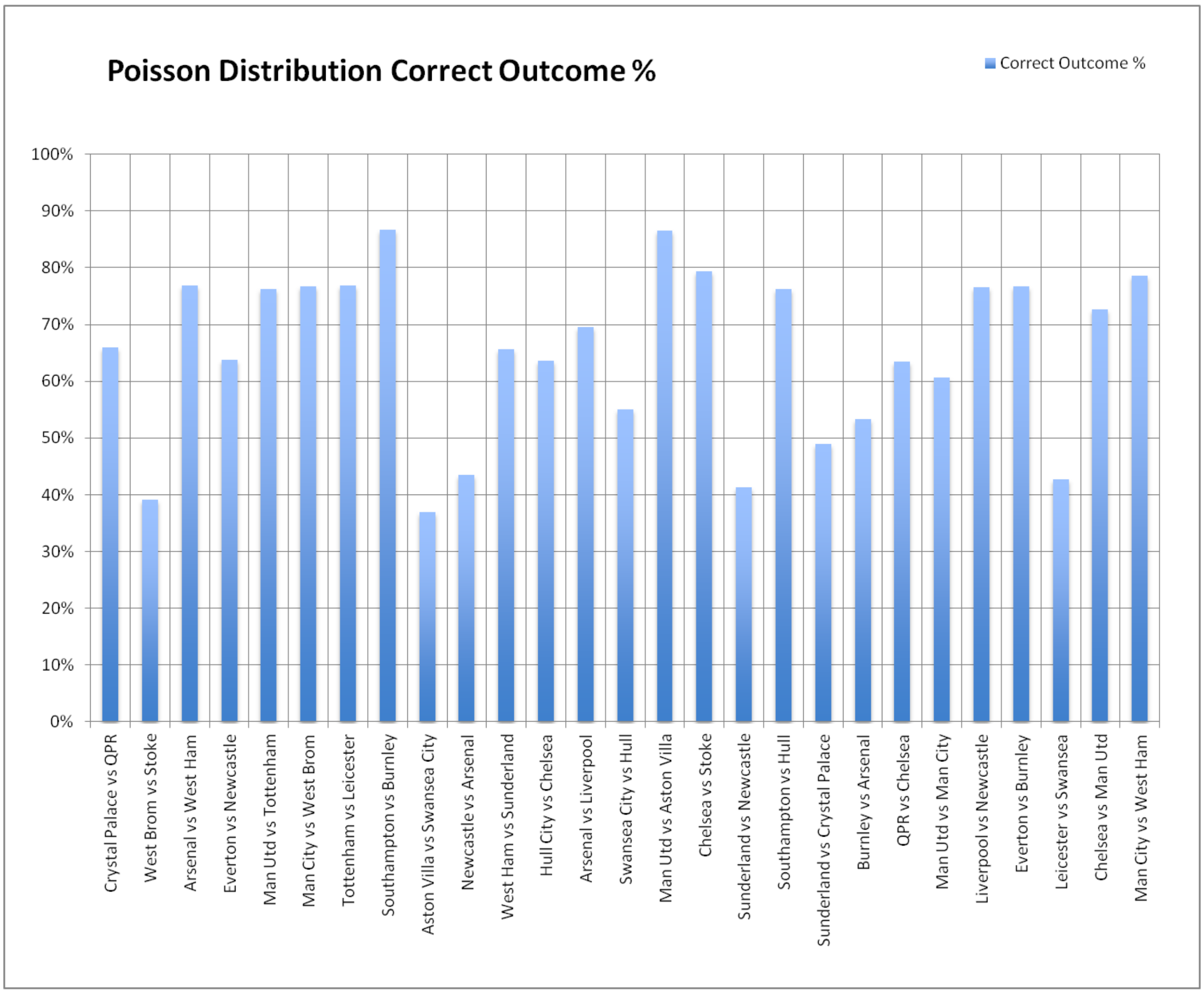

Figure 11 – Bar graph showing distribution of correct outcome prediction % for each match from Poisson Distribution

7.4 Odds

Using betting odds to predict football match outcomes, again seems like it should be able to produce correct outcomes to a high degree of reliability. As discussed previously, betting companies will want to make use of the best possible information and mathematical models in order to offer betting odds that are related to the outcomes of a football match. Indeed, by turning these odds into predictive percentages, this project correctly predicted the outcome of 29 of the 48 matches observed in weeks 6-11, which equates to 60.42%; these results are marginally better than the results from the Naïve Bayes Classifier and the Poisson Distribution. This is significant, as it shows that by using data sourced from social media we are able to achieve similar results to those devised from refined and tested models used by bookmakers.

Again, and similar to the results gained from the other input methods, this project once again was unable to gain an overall probability of any match ending in a draw, and therefore unable to predict any matches to be draws.

As Figure 12 displays below, the correct outcome prediction % from using the match odds data is high, and the two matches that we can see the lower % predictions for, are Sunderland vs Newcastle and Man Utd vs Man City. We need to take into account that these matches are local derbies, and that for a variety of different factors which are beyond the scope of this project, local derbies might have a more fair spread of outcome percentages, which means that there will be a lower percentage for the correct outcome.

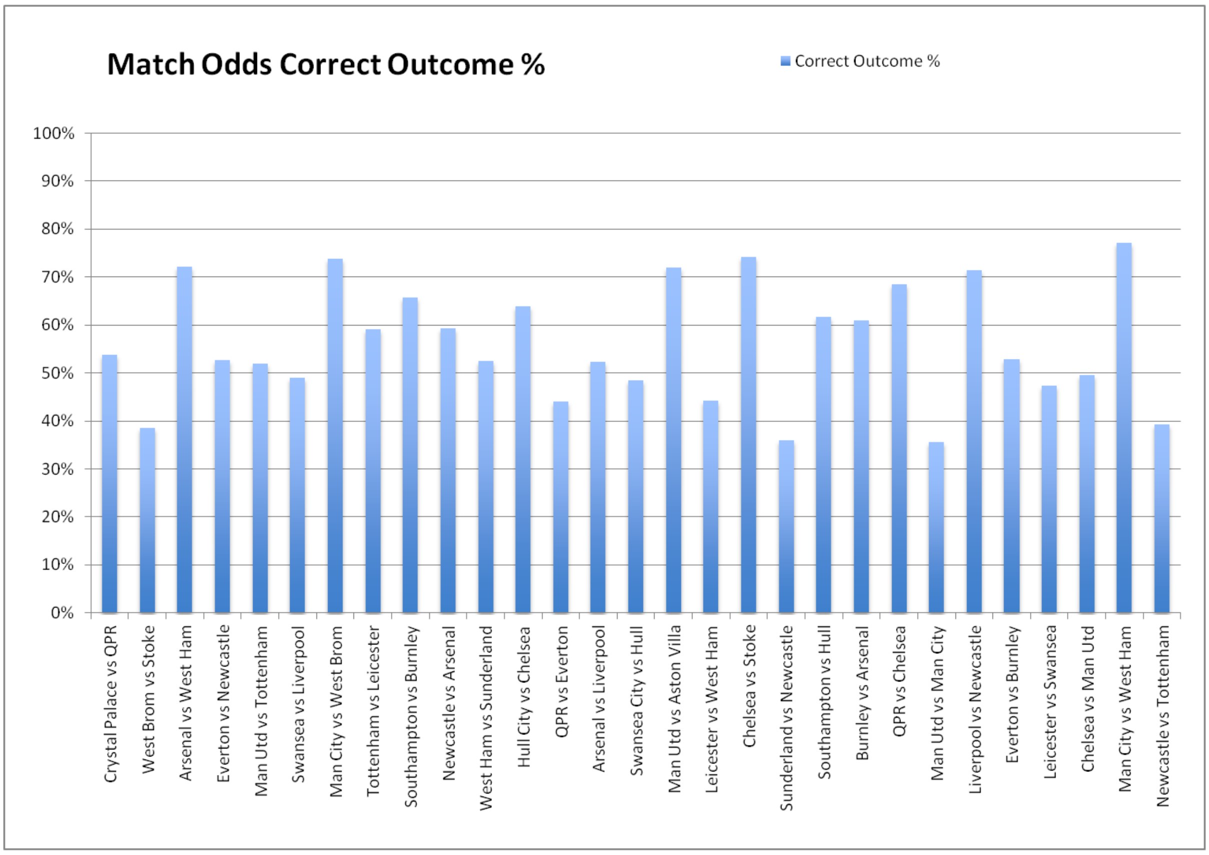

Figure 12 – Bar chart showing distribution of correct outcome prediction % for each match from Match Odds

7.5 Combining Inputs

In this section, I am going to discuss the results from combining the results of the Naïve Bayes Classifier, the Poisson Distribution and the match odds in order to build a model that is capable of consistently predicting match outcomes in the Premier League. To achieve this, the model described in section 5.1.4 will be used.

This project aims to find optimal values of the number of matches that can be predicted. To do this, the project will incorporate and apply weightings to the results from the Naïve Bayes Classifier, the Poisson Distribution implementation and the Match Odds. The weightings a, b and c will be applied to these three inputs respectively, where a + b + c = 1. Through extensive tests, I have been able to find 21 sets of optimal values for a, b and c, which achieve a 66.67% accuracy value in predicting the outcome of 32 matches out of a total of 48 matches.

These values are shown in the stacked bar chart in Figure 13 below.

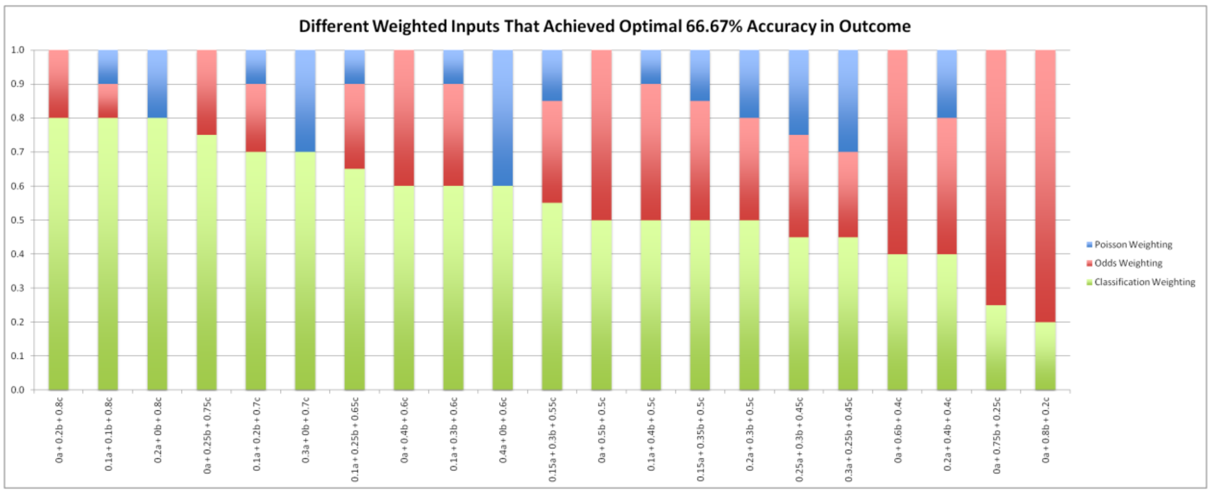

Figure 13 – Stacked bar chart showing differing values of weightings that all achieve optimal 66.67% accuracy

Using the combination of values shown in Figure 13 above, 32 matches from a total of 48 were predicted with the correct outcome, which equates to an accuracy of 66.67%. Simply by seeing these values, we can easily see that it is more beneficial to combine the Naïve Bayes Classifier, the Poisson Distribution and the Match Odds inputs rather than rely on these individual inputs by themselves, where they achieved 28/48, 27/48 and 29/48 respectively. It is interesting to note however from the tests, that by excluding the results from our classifier, it is not possible to achieve the optimal value of 32, and on the tests that this project ran, the maximum value achieved was 29 matches; these results are available in Appendix 4.

These results showed interesting results with regards to the different weightings of these inputs and their impact on the number of matches that can be correctly predicted. Figure 16 below shows a graph that displays the accuracy of each input method cumulatively for a varying number of matches, with the exception of week 6, where at least 50% accuracy was achieved for each method. In particular, we can see that the Match Odds show a great accuracy level, and that the Poisson Distribution seems to be less accurate after weeks 6 and 7. This insight into the data impacted upon my decision to use the prior mentioned weighting values for a, b and c.

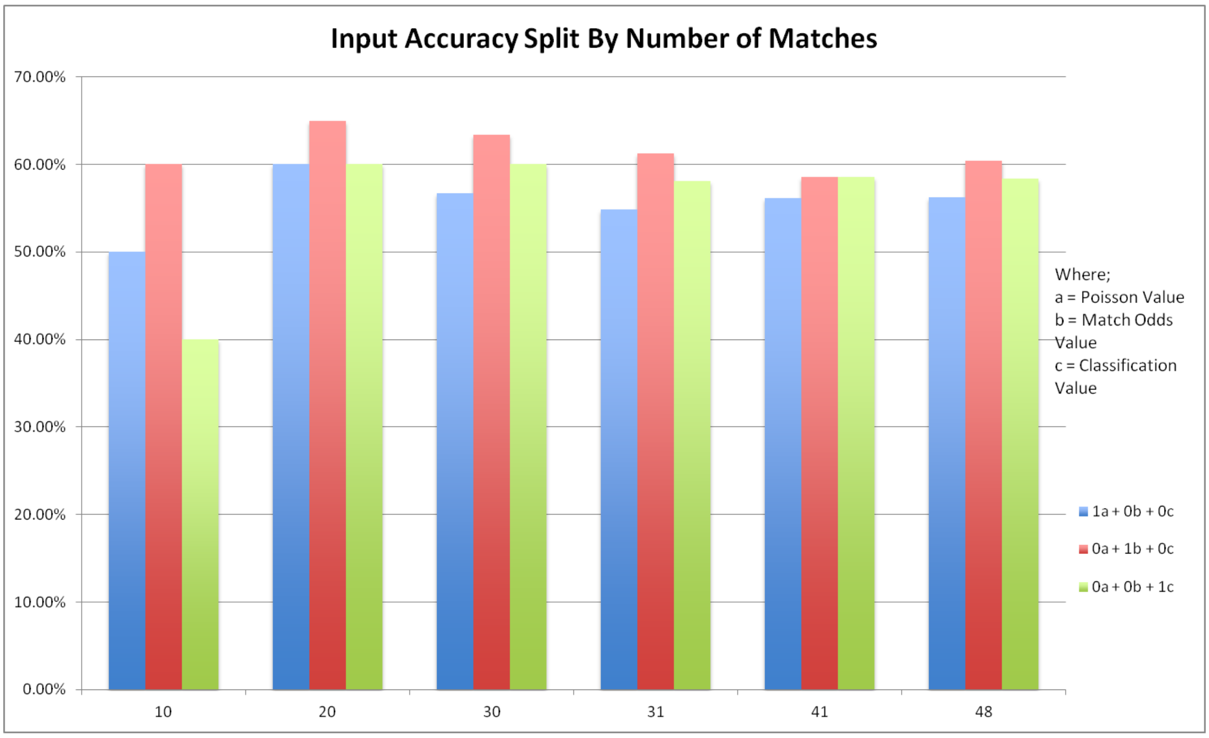

Figure 14 – Graph showing cumulative accuracy among each input method

The accuracy of each input method is also displayed in Figure 15 below, which shows the accuracy level on a week-by-week basis.

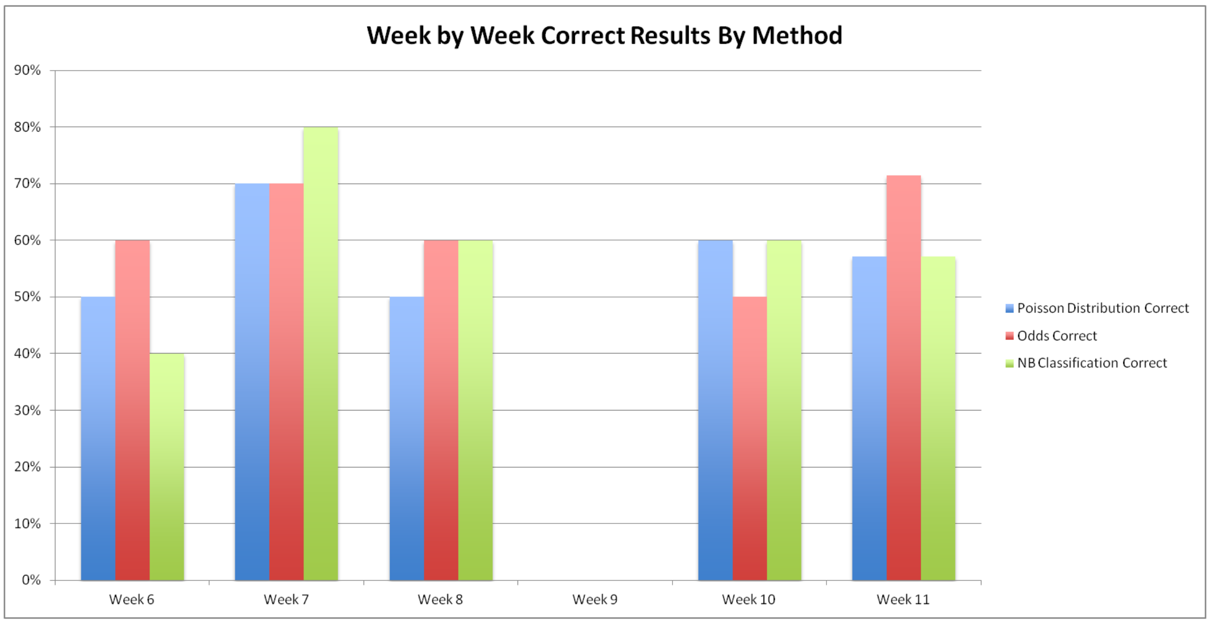

Figure 15 – Graph showing accuracy of each input method on week-by-week basis.

By testing different weightings on all 48 matches, I was able to achieve the optimal prediction value of 66.67% in 21 different cases; shown in Figure 16 below. Here, we can see that generally the inputs from the Naïve Bayes Classifier and the Match Odds are most important in helping to achieve this optimal value, and that minimising the weighting applied to the Poisson Distribution does not necessarily hinder the outcomes. Again, seeing the results from this graph that were able to achieve the 66.67% accuracy, contributed to my decision on the final weightings that I would apply to the different inputs.

In Figure 16, we can see that the weighting applied to the Naïve Bayes Classifier can vary from 0.2 to 0.8 and the Match Odds weighting can vary from 0.1 to 0.8, both whilst remaining effective. In contrast, the weighting applied to the Poisson Distribution only varies between 0.1 and 0.4, which highlights this input as not being as flexible or important in making classifications. Indeed, there are also a number of scenarios where either the Poisson Distribution or the Match Odds have 0 weighting, and the optimal accuracy can still be achieved, where in contrast there are no scenarios tested where the optimal accuracy level was achieved by applying 0 weighting to the Naïve Bayes Classifier.

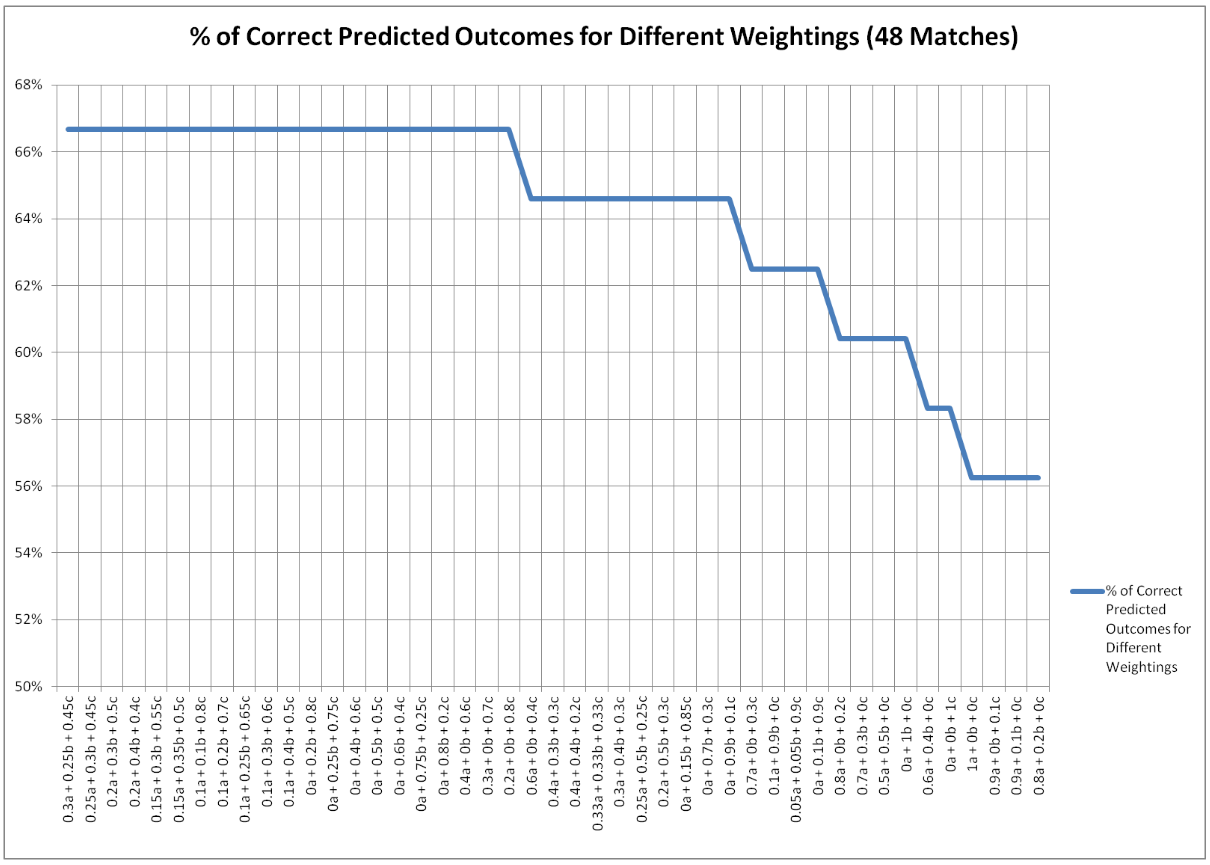

Figure 16 – Showing accuracy levels for tested weighting values when using 48 samples

In Figure 16 above, a variety of tests using different weightings for each of the inputs were run, using the full data set from weeks 6-11; we can see the varying levels of accuracy that were achieved. As shown, the best accuracy level that was achieved was 66.67%, whilst the worst was 56.25%.

As discussed previously, I found that whilst many different optimal weighting values for a, b and c were achieved, I have decided to propose the use of the weightings 0.2a + 0.4b + 0.4c for my project. I have chosen these values due to the relatively equal spread in weightings across the three inputs, which emphasises the importance of the match odds and classifier results.

As we have seen above, the inclusion of the Poisson Distribution data for each match did add some value, but not as much as the Match Odds or the Naïve Bayes Classification data; I have chosen to reflect this in the weighting values. I chose not to exclude the Poisson Distribution data altogether, as there always tends to be anomalies in football and results are achieved that are not expected, such as QPR beating West Brom 4-1, and Crystal Palace beating Man City 2-1, and whilst this information was not picked up by the Poisson Distribution, it may well be in the future and therefore the model should not be heavily reliant on few inputs.

Table 4 below displays the number of matches played and the percentage of these that could be predicted when using the 0.2a + 0.4b + 0.4c weightings. The percentage values that were achieved are in line with those that were obtained from other weightings that provided optimal results, and as we can see, the accuracy levels achieved are good; varying between 60 and 75%.

Using the 0.2a + 0.4b + 0.4c weighting values	
Total Matches Played	**Percentages**
10	60%
20	75%
30	70%
31	67.74%
41	65.85%
48	66.67%

Table 4 – When using the 0.2a + 0.4b + 0.4c weighting, the accuracy % at matches

Table 5 below shows the percentage of correct outcomes that are predicted using each of the Poisson Distribution, the Odds and the Naïve Bayes Classifier when the weighting 0.2a + 0.4b + 0.4c is applied, that the other inputs do not predict. This highlights the importance of including each input.

Match	**Correct Method**	**Correct Result**	**Wrong Method 1**	**Wrong Result 1**	**Wrong Method 2**	**Wrong Result 2**
Sunderland vs Aston Villa	NB Classifier	69%	Match Betting Odds	29%	Poisson Distribution	11%
Stoke City vs Crystal Palace	NB Classifier	74%	Poisson Distribution	29%	Match Odds	30%
Everton vs Southampton	NB Classifier	64%	Match Betting Odds	35%	Poisson Distribution	33%

Table 5 – Matches that one input predicted correct outcome of, but other two inputs failed to do so.

Table 5 above shows football matches where our equation using the weighting 0.2a + 0.4b + 0.4c was unable to predict the final match outcome, but one of the inputs was able to do so. As we can see, there were only 3 matches and on each occasion, the method that was able to predict the result was the Naïve Bayes Classifier. The classification results varied between 63-74% accuracy, whilst results from the other methods varied between 11 and 35% accuracy. These results show that the Naïve Bayes Classifier produces interesting results for football results, and can be investigated further in the future.

Further details regarding the results of the tests run, are available in Appendices 3, 4 and 5.

7.6 Evaluation

Using the data in the figures above and attached in the appendices, it seems fair to deduce that by performing sentiment analysis on data from Twitter, we are able predict football match outcomes in the Premier League to a better level of accuracy than traditional statistical models. By then combining this with betting data and a statistical model, the level of accuracy can be further improved. The method that has been developed has proven to outperform probabilities drawn from both betting data and a statistical model based on historical Premier League data, and could be a method that is further investigated.

Whilst there has already been work done regarding predicting events using Twitter as mentioned previously, this method proves that indeed it is applicable to Premier League football matches, and could be evaluated further to investigate the psychological impacts of Twitter on professional sports athletes.

Finally, it is worth considering that whilst the favourable results of 27, 28, 29 and 32 out of 48 matches in terms of accuracy were achieved, this project aimed to achieve correct predictions on football match outcomes 70-80% of the time, which was, unfortunately, not achieved. This can be improved in the future as more data is accumulated from Twitter, however, it is still valid that there does not appear to have been a consistent way to profit from football match prediction, and that both data from Twitter and football matches can be very volatile.

8 - Future Work

Given the broad nature of this project in terms of the topics that are included, there are a number of ways that this project could be extended further including classification, statistical analysis, and natural language problems.

With regards to classification problems, I would further consider the use of feature selection. Feature selection is vital, and namely relates to ensuring the selection of the most relevant and the filtering of ineffective features, so that we can accurately classify a document. I would also consider further filtering spam tweets, along using a stop word list which contains words in the English language that hold no particular sentiment. I would consider applying the Porter Stemming algorithm, where we would remove the inflexional endings of English words [41]. Finally, given the global attraction and support of football, I would also consider extending the development of this project to provide support for languages other than English.

If this project were taken on and developed to a more advanced level, I would also further research and consider the use of alternative classification models that could provide better accuracy.

For statistical analysis I would consider adapting the implemented Poisson Distribution model to make it more complex, to account for additional factors such as mental factors, fatigue and weather types. Indeed for this project, as shown in Figure 20 below, we can see that there were significantly more wins for home teams than away teams or draws; I could therefore incorporate this information in the Poisson Distribution to represent this likelihood.

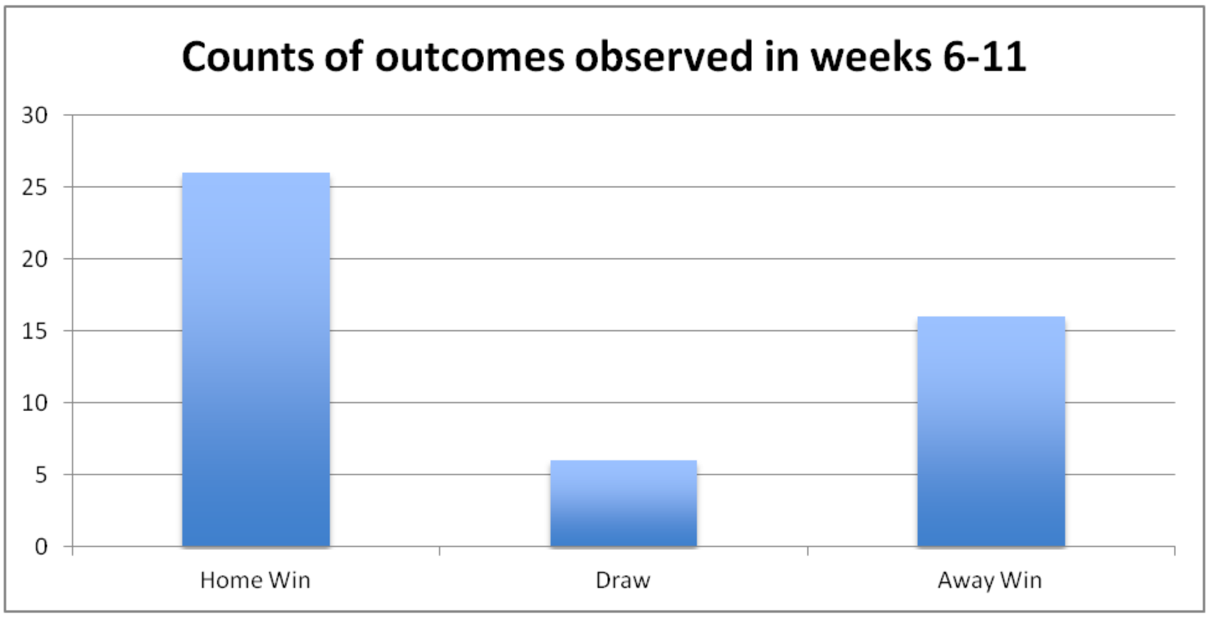

Figure 20 – Counts of different match outcomes observed in Premier League in weeks 6-11

9 - Conclusions

The conclusions that can be taken from this project are that Twitter data, historical data relating to football matches, and match odds can all be used to predict football matches to a greater level of accuracy than traditional statistical models. In combination with each other, these inputs have allowed this project to predict 66.67% of matches in the Premier League throughout March and April 2015. The additional insight provided by Twitter sentiment was displayed, as by using Twitter data alone, this project predicted the outcome of 3 matches that other existing models could not predict.

However, as we know, football is unpredictable to an extent and can be impacted by a wide range of factors. However, this project leaves an interesting introduction into the use of sentiment analysis upon publicly available data, which can be applied in a wide range of fields.

10 - Reflection on learning

Throughout this project, I have been able to learn disciplinary techniques on how to persevere whilst working, how to present work and findings, and how to work closely with my supervisor in a way to provoke further thought and interest in my project.

I have also learned time management and research skills, and learned how to properly search for scholarly articles, which may be related to this field of work.

Finally, I have technically advanced my knowledge of the Twitter API, MongoDB and the NLTK Python library, as well as my general knowledge of text classification algorithms, which I expect will be valuable in the future.

References

[1] -Twitter (2015) *About Twitter.* Available at: https://about.twitter.com/company (Accessed 26th April 2015).

[2] – Chen, R; Lazer, M. (2011) *Sentiment Analysis of Twitter Feeds for the Prediction of Stock Market Movement.* Available at: http://cs229.stanford.edu/proj2011/ChenLazer-SentimentAnalysisOfTwitterFeedsForThePredictionOfStockMarketMovement.pdf (Accessed 26th April 2015).

[3] - Tumasjan, A; Sprenger, T; Sandner, P; Welpe, I. (2010) Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment, *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media (ICWSM),* p178-185.

[4] – Constantinou, A. (2012) *Bayesian networks for prediction, risk assessment and decision making in an inefficient Association Football gambling market*. Ph.D Thesis, Queen Mary, University of London.

[5] - Paul, M; Dredze, M. (2011) You Are What You Tweet: Analyzing Twitter for Public Health, *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM)*, p265-272.

[6] – Sinha, S; Dyer, C; Gimpel, K; Smith, N; (2013) *"Predicting the NFL Using Twitter",* Presented at ECML/PKDD 2013 Workshop on Machine Learning and Data Mining for Sports Analytics. Available at: www.cs.cmu.edu/~nasmith/papers/sinha+dyer+gimpel+smith.mlsa13.pdf. (Accessed 26th April 2015).

[7] - Gayo-Avello, D. (2012) *A Balanced Survey on Election Prediction using Twitter Data.* Available at: http://arxiv.org/pdf/1204.6441v1.pdf. (Accessed 26th April 2015).

[8] - Maher, M.J. (1982). Modelling association football scores. *Statistica Neerlandica* 36, p109-118

[9] - Dixon, M; Coles, S. (1997) Modelling Association Football Scores and Inefficiencies In The Football Betting Market. *Applied Statistics* 46, p265-280.

[10] - Gardner, J. (2011) *Modeling and Simulating Football Results*. MMath Thesis, University of Leeds.

[11] – Aggarwal, C. (2014) An Introduction to Data Classification. In: *Data Classification: Algorithms and Applications*. Boca Raton: CRC Press. p2.

[12] – Chen, J; Huang, H; Tian, S; Qu, Y. (2009) Feature selection for text classification with Naïve Bayes. *Expert Systems with Applications: An International Journal*. 36, p5432-5435

[13] - Manning, C; Raghavan, P; Schütze, H. (2008). *Introduction to Information Retrieval*, Cambridge: Cambridge University Press.

[14] – OpenCourseOnline (2012) *6 – 3 – Formalizing the Naïve Bayes Classifier – Stanford NLP – Dan Jurafsky & Chris Manning*. [Online Video]. 04 April. Available at: www.youtube.com/watch?v=TpjPzKODuXo. (Accessed: 26 April 2015)

[15] - Kranjc, J. (2013). *Twitter Tap* [Computer Program]. Available at: http://janezkranjc.github.io/twitter-tap/. (Accessed 26th April 2015).

[16] - NLTK (2015). Available at: http://www.nltk.org. (Accessed 26th April 2015).

[17] – Gambling Commission (2014) *Gambling industry statistics April 2009 to March 2014*. Available at: www.gamblingcommission.gov.uk/docs/Industry-statistics-April-2009-to-March-2014.xlsx. (Accessed 9th April 2015).

[18] - Abram, C. (2008) *Our First 100 Million.* Available at: https://www.facebook.com/notes/facebook/our-first-100-million/28111272130. (Accessed 26 April 2015).

[19] - Facebook (2014) *Facebook Reports Fourth Quarter and Full Year 2013 Results.* Available at: http://investor.fb.com/releasedetail.cfm?ReleaseID=821954. (Accessed 26 April 2015).

[20] - eMarketer (2009) *US Twitter Usage Surpasses Earlier Estimates.* Available at: http://www.emarketer.com/Article.aspx?R=1007271. (Accessed 26th April 2015).

[21] - eMarketer (2014) *Emerging Markets Drive Twitter User Growth Worldwide.* Available at: http://www.emarketer.com/Article/Emerging-Markets-Drive-Twitter-User-Growth-Worldwide/1010874. (Accessed 26th April 2015).

[22] - James, J. (2014) *Data Never Sleeps 2.0.* Available at: http://www.domo.com/blog/2014/04/data-never-sleeps-2-0. (Accessed 26th April 2015).

[23] - Reiner, S. (2013) *Big Data and APIs Combine to Boost Business.* Available at: http://www.mashery.com/blog/big-data-and-apis-combine-boost-business. (Accessed 26th April 2015).

[24] - Twitter (2015) *Tweets.* Available at: https://dev.twitter.com/overview/api/tweets. (Accessed 26th April 2015).

[25] - Twitter (2015) *Streaming.* Available at: https://dev.twitter.com/streaming/overview. (Accessed 26th April 2015).

[26] - MongoDB (2015) *Introduction to MongoDB.* Available at: http://docs.mongodb.org/manual/core/introduction/. (Accessed 26th April 2015).

[27] - Twitter (2015) *List members.* Available at: https://twitter.com/TwitterUK/lists/premier-league-clubs/members. (Accessed 26th April 2015).

[28] – Tan, P; Steinbach, M; Kumar, V. (2005) Classification: Basic Concepts, Decision Trees and Model Evaluation. In: A *Introduction to Data Mining*. Boston: Pearson. p148.

[29] - StatTrek (2015) *Bayes' Theorem.* Available at: http://stattrek.com/probability/bayes-theorem.aspx. (Accessed 26th April 2015).

[30] - StatSoft (2015) *Naïve Bayes Classifier.* Available at: http://www.statsoft.com/textbook/Naïve-bayes-classifier. (Accessed 26th April 2015).

[31] - Ng, A. (2014) 'Generative Learning algorithms', lecture notes distributed in the topic CS229 Machine Learning. Stanford University, Stanford In September 2014.

[32] - NLTK (2015) *NLTK Chapter 6*. Available at: http://www.nltk.org/book/ch06.html. (Accessed 26th April 2015).

[33] – Ipsos MORI (2003) *Rugby Union 'Britain's Second Most Popular Sport'.* Available at: https://www.ipsos-mori.com/researchpublications/researcharchive/928/Rugby-Union-Britains-Second-Most-Popular-Sport.aspx. (Accessed 26 April 2015).

[34] - FIFA (2015) *Associations.* Available at: http://www.fifa.com/aboutfifa/organisation/associations.html. (Accessed 26th April 2015).

[35] - Ratcliffe, J. (2014) *Poisson Distribution: Predict a soccer betting winner.* Available at: https://www.pinnaclesports.com/en/betting-articles/sport/how-to-calculate-poisson-distribution. (Accessed 26th April 2015).

[36] - Fan Prediction League (2012) *What is the most common scoreline in the Premier League?.* Available at: http://www.fanpredictionleague.com/blog/what-is-the-most-common-scoreline-in-the-premier-league. (Accessed 26th April 2015).

[37] - Odds Portal (2015). *Premier League Results & Historical Odds.* Available at: http://www.oddsportal.com/soccer/england/premier-league/results/. (Accessed 26th

April 2015).

[38] - Twitter (2015) *The Search API.* Available at: https://dev.twitter.com/rest/public/search. (Accessed 26th April 2015).

[39] – Kimono Labs (2015) *Kimono Labs.* Available at: https://www.kimonolabs.com/. (Accessed 26th April 2015).

[40] - Bromberg, A. (2013) *Second Try: Sentiment Analysis in Python.* Available at: http://andybromberg.com/sentiment-analysis-python/. (Accessed 26th April 2015).

[41] - Porter, M. (2006) *The Porter Stemming Algorithm.* Available at: http://tartarus.org/martin/PorterStemmer/. (Accessed 26th April 2015).

[42] - Twitter (2015) *GET search/tweets.* Available at: https://dev.twitter.com/rest/reference/get/search/tweets. (Accessed 26th April 2015).

[43] - Hiemstra, D. (2009) *Encyclopedia of Database Systems*. London: Springer. p2169-2170.

[44] - Tran, M. (2015) *Chelsea fans allegedly involved in Paris Métro racist incident identified.* Available at: http://www.theguardian.com/world/2015/feb/22/chelsea-fans-paris-metro-racism-identified-police. (Accessed 3rd May 2015).

[45] - University of Maryland. (2015) *Smoothing.* Available at: http://terpconnect.umd.edu/~toh/spectrum/Smoothing.html. (Accessed 3rd May 2015).

Appendix 1

a) List of keywords used to search for tweets containing these words:

arsenal,@arsenal,aston
villa,@avfcofficial,burnley,@burnleyofficial,chelsea,@chelseafc,crystal
palace,@cpfc,everton,@everton,hull,@hullcity,leicester,@officialfoxes,liverpool,@lfc,man
city,@mcfc,man
utd,@manutd,newcastle,@nufc,qpr,@qprfc,southampton,@southamptonfc,stoke,@stokecit
y,sunderland,@sunderlandafc,swansea,@swansofficial,tottenham,@spursofficial,west
brom,@wbafcofficial,west ham,@whufc_official

b) List of keywords used to remove more than 1 mention of a team from tweets:

'arsenal', 'afc', 'gunners', '@avfcofficial', 'villa', 'burnley', 'clarets', 'chelsea', 'blues', 'cpfc', 'palace', 'everton', 'toffees', 'hull', 'tigers', 'leicester', 'foxes', 'liverpool', 'lfc', 'reds', 'city', 'mcfc', 'man utd', 'manutd', 'united', 'newcastle', 'magpies', 'nufc', 'qpr', 'queens park rangers', 'southampton', 'saints', 'stoke', 'potters', 'sunderland', 'black cats', 'swans', 'tottenham', 'spurs', 'west brom', 'wba', 'west ham', 'whu', 'hammers'

Appendix 2 - List of Time Periods and KO Times.zip

Appendix 3 - PoissonDistribution.xlsx

Appendix 4 - ResultsFromTests.xlsx

Appendix 5 - TeamResults.zip - File containing results from NB Classifier from weeks 6-11

Appendix 6 - Project Code.zip