Can Machine Learning Harness the "Wisdom of the Crowd" to Predict the Outcome of Football Matches?

Author: Liam Byrne

April 13, 2020

Contents

Can Machine Learning Harness the “Wisdom of the Crowd” to Predict the Outcome of Football Matches?

Liam Byrne, *St. John the Baptist School, Woking*

Abstract—The interest in predicting the results of football matches has never been so great due to the rewards available within the game and the business ecosystem built around it. Aristotle suggested a more informed decision can be made by combining the views of a large number of independent people. This concept is called ‘Wisdom of the Crowd’. Sentiment analysis is the process of recognising the affective state of a person based on their utterances. Modern social media platforms such as Twitter offer a vehicle for groups of football fans to express their opinions. This paper examines whether automated computer techniques can analyse the sentiment of fans and through applying the ‘Wisdom of the Crowd’ concept to predict the outcome of games. In order to do this, an extensive literature review has been made and an original experiment has been carried out. The experiment consisted of building software to collect Tweets from supporters of selected football clubs. The sentiment was determined using machine learning and lexicon techniques. This result was used to predict a football score and the score was compared to the actual result of the game. Overall, a prediction accuracy of 38.9% has been achieved which falls significantly short of other statistical techniques. Therefore, it was concluded that ‘Wisdom of the Crowd’ is not a good standalone method to predict football match outcomes. It is suggested that this is because football supporters are not a ‘wise crowd’ due to their inherent optimism and herd-like mentality. However, sentiment measured during games correlates with in-game key events and so there are some potentially interesting applications for this technique.

Index Terms—Machine Learning, Wisdom Of The Crowds, Twitter, Sport prediction, Data Science, NLP

1 INTRODUCTION

ASSOCIATION football has become increasingly popular globally and with this a huge business ecosystem has grown up around the sport. In England, the Premier League and the EFL Football League have become a business worth £ 5.5Bn in 2018 [1]. Subsequently, the football betting market has substantially grown amassing 45.9% of gross gambling yield of all internet accessed gambling in the United Kingdom [2]. As a result, there is unprecedented interest in the prediction of the outcome of football matches.

The art of prediction has been synonymous with human nature throughout recorded history. In ancient history, prediction was a divine art. Pythia, the Oracle of Delphi, in Ancient Greece was said to communicate with the god Apollo and would have visions of the future. Thousands travelled far and wide to Delphi and the Greek Empire would not make any major decision without consulting Pythia [3]. Today, prediction is a no longer an art but a complex science as a result of the massive growth of Mathematical techniques and Computer Science. This is powered by increasing processing power in computers from the exponential increase in the number of transistors on a microprocessor following Moore’s Law. This has enabled Data Science to progress and develop powerful techniques to make predictions.

Data science unifies statistics, data analysis, machine learning and related methods to understand and analyse actual phenomena [4]. A subset of this field is known by the buzzword: ‘Big Data’. Traditional analysis methods become overwhelmed when confronted with large data sets; ‘Big Data’ is defined when input values have a volume, velocity and variety so vast that it requires specific technology and methods to extract meaning and understanding [5].

Football has become an increasingly interesting topic for Data Science due to the mass gathering of data and its popularity in social media. This report investigates how some of that data can be used to predict football match outcomes, particularly focussing on the use of social media data to harness the so-called “wisdom of the crowd” a principle first proposed by Aristotle who wrote:

> **It is possible that the many, though not individually good men, yet when they come together may be better.**
> **- *Aristotle* [6]**

• *L.Byrne is with St. John the Baptist School, Woking.*

April 13, 2020

2 FOOTBALL DATA

2.1 The Role of Data in Sport

Predicting the outcome of a football match is not a trivial process. Data and statistics have become ubiquitous with any modern sport. Through the ongoing information age there has been development of measurement and high capacity storage of data relevant to performance in sport.

The expansion of sport data collection has revolutionised the game.

Data now plays a role for:

1) **Football Clubs;**
   - **Team Selection**
   - **Analysis of opposing teams**
   - **Team tactics**
   - **Scouting players for transfers**
2) **Peripheral Markets and Games;**
   - **Fantasy Football**: players now gain points based on calculated performances informed by in-game statistics rather than basic results of the game. Additionally, players have access to more complex statistics for their own team selection.
   - **Match Betting**, more appropriate odds can be calculated to ensure a profit for the bookmaker while the customer is still given reasonable odds.
   - **Sport journalism** and sport news reports

2.2 Access to Data

Data may give teams a competitive edge over opposition directly affecting results and therefore there has been a growing interest in accessing data across the football industry. Moreover, peripheral markets that are data-reliant will have the advantage over rivals if they have exclusive access to data. The hunger for unique and exclusive data has encouraged the emergence of data collection and analysis companies such as Opta Sports who sell sport data to clients.

The sheer volume of data collected has brought football into the field of Big Data. Conventional statistical methods are overwhelmed by the volume of data and so this requires specific data science techniques to be employed.

The demand for more data has widened the scope of data collection, indirect sources such as social media can be utilised to derive meaningful information. For example, data can be collected reflecting the sentiment of supporters related to a team or a specific player. This technique has also been demonstrated in a political context such as in elections, an example is the 2012 USA election between Obama and Romney [7].

2.3 Types of Data

Data analyst companies have techniques to collect vast amounts of data from different aspects of the game. Generally, this data can be separated into three groups: player data, match data and peripheral data.

Player data is further broken down into their performance in games and their theoretical ability ratings. Each player in a team can be assessed through their contributions to the game, these can include goals scored, pass completion percentage, saves (goalkeepers) and many more. Opta collects this data by staff analysing a live video feed and manually aggregating the different match contributions of a player. For more complex measures such as distance travelled or location heatmaps; video-based techniques are adopted to aggregate the more continuous data contributions. Theoretical ability ratings are metrics which define the strengths and playing style of a given player. A popular implementation of this is the Electronic Arts (EA) Sports FIFA data set used in their annual release of the FIFA video game. EA breaks down a player's ability into around 30 ratings such as their shot power, sprint speed and many other ability ratings. These metrics are quite subjective despite their quantitative appearance (0-100), despite this, they provide a balanced dataset as each rating is relevant and the better team to win the game is more likely to have higher ratings. However, using theoretical data on its own does not consider the chance of shock results where the underdog team wins the match.

Match data is the aggregation of all the team's contributions. Many of the metrics are simply the sum or the average of the individual performances or contributions, such as number of completed passes or total distance covered. Other metrics simply show whether the team are achieving the objectives of the game including the result or clean sheets (0 goals conceded). This data can allow recent form to be measured which is a very common choice for inferring the result of upcoming games.

Peripheral data is generally not directly extracted from a match. This can be hypothesised to reflect the outcome of a match. Examples of this are: home team advantage (team is playing at the stadium which belongs to their club), weather and social media reactions. Social media is a new addition due to platforms such as Twitter attracting groups of fans together through team hashtags (e.g. #FFC for Fulham Football Club). The focus of this report is to investigate the extent which prediction models based on social media performs against other prediction methods.

3 Wisdom of the Crowd

The so-called 'wisdom of the crowd' is a maxim postulated by Aristotle thousands of years ago [6] and analysed by Galton in 1907 where people attempted to guess the weight of an ox [8]. During this millennium the concept has been popularised by Surowiecki's book: 'The Wisdom of Crowds' (2004) [9].

Wisdom of the crowd can be defined as the collective intelligence of a group of individuals compared to a single expert. Today there has been a resurgence of interest into this principal due to the immense growth of social media. This media provides a platform for users to document and share their personal experiences and opinions within their friendship circles and then this propagates across the internet. The idea of 'six degrees of separation' where it takes six or fewer intermediary people to reach any specific person in the world is accelerated through social media - Facebook determined 3.57 steps between any pair of users in 2016 [10], this relationship illustrates that propagation is viral. Social media requires a large number of active users to run successfully, by containing potentially billions of active users in a single platform 'wisdom of the crowd' techniques can be demonstrated.

There is a multitude of social media to choose from, such as the specialised image and experience sharing such as Instagram or Snapchat, alternatively the specialised opinion sharing platform of Twitter which gives users 280 characters (previously 140) to talk about a subject or an event and view and enter a discussion using hashtags (e.g. #Brexit).

3.1 Features of a 'wise crowd'

According to Surowiecki: to be an effective platform to collect wisdom from users; the medium must meet certain requirements. Surowiecki emphasises the importance of key components of a 'wise crowd' these include:
'Diversity of opinion', 'Independence', 'Decentralisation', 'Aggregation' and 'Trust'.

3.1.1 Diversity of Opinion

The first element is 'Diversity of opinion'; A group should consist of a variety of opinions, otherwise it is no more useful than a single person. It is vital that there is a range of expertise in the area this includes 'eccentric' viewpoints where the less informed or controversial perspectives provide a diverse range of opinions. In the first quarter of 2019 Twitter hosted an average of 330 million active users per month [11], this population contains a variety of nationalities and personal backgrounds which provides a range of expertise and perspectives. Furthermore, Twitter is an attractive place for experts in a given field as it acts as personal career networking and marketing for themselves or a company, the distributed experts is a key point Surowiecki covers to prevent homogeneity. These qualities mean that Twitter undoubtedly satisfies the key element of diversity of opinion.

3.1.2 Independence

All individuals must not be influenced by others and so give an 'Independent' perspective. The physiological phenomenon of 'groupthink' must be eliminated; the phenomenon heavily documented by Irving Janis in 1972 who illustrated it using the failed CIA operation 'Bay of Pigs' in 1961 [12]: The operation was initiated by President Eisenhower's administration who left office in 20th January 1961, less than 3 months prior to the operation. President Kennedy's administration, which took office after Eisenhower, appeared subservient to the CIA representatives and suppressed the injection of new opinions from Arthur M. Schlesinger Jr. and Senator J. William Fulbright. The fear of a deteriorating harmony caused the members of the debate to restrain themselves from posing alternative viewpoints, especially when the Kennedy team explicitly rejected objections. This caused dysfunctional decision-making leading to the inevitable debacle. After the fiasco, Kennedy appeared to have an epiphany of horror as he exclaimed "All my life I've known better than to depend on the experts. How could I have been so stupid, to let them go ahead?" [13]. This primary example of groupthink has shaped decision-making to the present day. In the context of social media independence is debatable. The users of social media usually participate on their own accord as it merely requires an unrestricted access to the internet whom are all capable of making their own opinion. However, the 'timeline' feature of social media (especially Twitter) is likely is to influence an individual's opinion as they are more unlikely to go against the majority. In contrast, the illusion that social media is more anonymous and disconnected from reality may encourage people to convey their personal opinion without risking the backlash of physically announcing a controversial argument. Additionally, the confidence and self-esteem required to announce eccentric or controversial opinions on social media is negligible compared to reality [14]. Whether social media meets Surowiecki's 'wise crowd' independence is a grey area as people are undoubtedly influenced by what is presented to them on their timeline as it often contains the opinions of relatives and friends although there is still room for eccentric and controversial opinions due to the safety of feeling anonymous lack of self-disclosure.

3.1.3 De-centralised

Another key element is that the medium is 'decentralized' where each member of the population has their own knowledge and there is freedom from any hierarchy of authority. Essentially there is no single person or group which can dictate the answer of a population. An example of this is seen across open-source software, the source code of the Android operating system or the Linux operating system which has been developed and maintained by a community which has no hierarchy of expertise but simply peer-reviewed by other enthusiasts.

Surowiecki attributes the 2003 NASA Space Shuttle Columbia disaster to the lack of decentralization. According to a speech by NASA astronaut Mark Kelly there was a culture which rejected all opposing opinions by the less experienced engineers this suggests centralization of the high-level engineers [15], this is also revealed in a journal entry into the Project Management journal: Organizational behaviour and disaster [16]. The disaster was caused by dismissal of the concerns which less experienced engineers had about foam potentially rupturing the fuselage of the shuttle, the groupthink mentality gave a false sense of control. Consequently, the dismissed concerns unfolded at lift-off where the left wing of the orbiter was ruptured causing catastrophic damage. Mark Kelly re-affirms this groupthink sentiment with the counter-intuitive quote:

> “ **None of us is as dumb as all of us.**
> ***- Mark Kelly, NASA* [15]** ”

even a room filled with highly intelligent NASA engineers. Kelly's quote juxtaposes Aristotle's view of the principal wisdom of crowds being "It is possible that the many, though not individually good men, yet when they come together may be better", this highlights the importance of Surowiecki's criteria of 'wise-crowds'.

Social media such as Twitter should easily achieve decentralization as anyone with an account can voice their opinion with their own expertise. However, Twitter has an underlying hierarchy similar to all social media: followers and likes. The number of followers an account has is directly proportional to the regular viewers of a tweet meaning there is a larger catchment scope of users, often the experts in a field will have a greater number of followers. Furthermore, Twitter's verified account system of experts or celebrities to prove legitimacy of an account has had an effect where users who are verified (blue-tick by username) appear to carry more authority despite not necessarily being an expert in the field or pushing their own political agenda, this has been dubbed as 'Blue-tick culture' by bloggers [17]. This elite group of verified users prevent decentralization as there is an imbalance of authority. The number of likes a tweet gets also gives a tweet more authority as a user will believe a tweet is likely to be correct through herd-instinct and are more likely to place a like themselves through peer-pressure. With this considered Twitter appears to be a decentralised platform briefly after a tweet is posted however once a tweet receives many user impressions the underlying hierarchy distorts and centralises the medium.

3.1.4 Aggregation

Wisdom of the crowd requires a balanced system to combine the notion of a large sample of people. The methods of aggregation can be active or passive. Active methods require explicit user interaction to select their preference about a question, similar to a referendum. Passive methods usually involve suggesting the notion of the group of people without their knowledge but analysing their actions or speech.

Twitter again appears to be the prime candidate as there are integrated features to perform straw polls for active techniques and an enormous community of users who regularly submit tweets with text which can be analysed with sentiment analysis part of the passive techniques. Sentiment analysis can indicate the subject users are discussing through keywords, a polarity rating of how positive and the intensity of sentiment and the use of different grammar to indicate more subjective speech or more objective speech. Passive techniques may receive more honest answers as the user does not understand their participation. Furthermore, it is likely that the participation level of active techniques may be significantly lower than passive techniques. This can be performed anonymously, and the tweets received are accessible to any twitter user meaning there is no breach of privacy.

3.1.5 Trust

Finally, each member of the population must trust the system of aggregation and the platform itself. Any form of scepticism may change an individual's answer they place. The result of a minor lack of trust may prevent lack of honesty. A severe lack of trust may cause absence from the platform entirely.

In recent months, the trust between an end-user and a social media platform has become a hot-topic. A social media company hosts a platform for individuals to document their experiences in life and share media with others. A digital foot print can be formed by analysing a user's activity and suggesting personal interest. Access to an individual's digital footprint can allow reconstruction of the interests and personal information such as their political orientation. The Facebook-Cambridge Analytica scandal of early 2018 has acted as the moment of realisation for anyone with a digital footprint concerning how much of their personal data is stored and how it is used. The data scientist and lecturer of Psychology Aleksandr Kogan developed an application called "This Is Your Digital Life" which is a personality test. This application was given to Cambridge Analytica which offered the app to Facebook users and claimed the data and results were simply for academic reasons at Cambridge University. However, the application collected excessive personal account information about the user and all of the user's friends [18]. According to an estimation by Facebook, the dataset contained 87 million users [19]. Cambridge Analytica was exposed for using this information to target individuals with adverts and propaganda to sway their vote on elections, especially the 2016 Hilary Clinton Donald Trump election [20].

In the UK the previous Data Protection Act of 1998 (DPA1998 [21]) did not emphasise the same level of transparency of data usage compared to the recent Data Protection Act of 2018 (DPA2018) which complies with GDPR (General-Data Protection Regulation) which is mandatory to follow within the European Union. The current DPA2018 has a requirement that collection of specific data must have an explicit purpose and the introduction of 'the right to be forgotten' meaning an individual can demand to see all data stored about them and delete them [22].

In 2019 Ofcom reported that the public trust in Facebook is at just 31%, the lowest of the UK's top 10 websites [23]. Additionally, the liking and sharing mechanism was used 20% less suggesting less usage of the platform, however, their total user numbers continues to rise [24].

4 Review of Methods for Prediction

4.1 Conventional Machine Learning Approaches

Machine learning is the study of algorithms that can perform a specific task without being explicitly programmed. A model is fed raw data as an input, in supervised learning the model has an understanding about the desired outcome which the raw data should return, but in unsupervised learning the model is used to draw inferences. The model can learn these patterns to understand unseen data.

Supervised machine learning algorithms can be divided into two groups: classification and regression although, there is a large overlap of algorithms performing both. Classification is where a model can infer the qualitative class label of data provided based on analysing the features. Regression is where the model outputs a quantitative value based on the correlations of different variables.

Currently, machine methods are a popular method of prediction because of the ability to find patterns in large datasets without prior knowledge about specific correlations. In football there is a vast supply of data meaning machine learning is an appropriate method. Although machine learning is still sparse in usage, statistical methods using player and match data are the most common. Each current investigation has a different selection of data and different types of machine learning models. Pettersson and Nyquist from the University of Chalmers [25] demonstrated a classifier using neural networks to classify the outcome of football matches on their master's thesis. They experimented data related to different parts of the game:

- **Line-ups and positioning** of players
- **Substitutions**: Players leaving and entering the game
- **Goals**: goal-scorer and assist
- **Card**: which player receives the card
- **Penalties**: Penalty taker and success rate

This data was fed into different types of neural networks to assess the accuracy. An artificial neural network (ANN) is an approach based on the biological processes in the brain. The model has neurons which perform an operation and are connected to neurons in the adjacent layer. Each connection has a weighting which govern the amplitude of a connected between two neurons.

They determined different success rates based on different amounts of data provided, their main conclusion was that prediction accuracy increases later in the game due to more data being available later on. Interestingly, they attempt to compare their model with a human level of prediction accuracy. They assess the predictions of betting companies and newspapers in Gothenburg to be representative of a knowledgeable football fan. The accuracy was assessed by the proportion of correct outcome predictions. The most accurate newspaper 'Sydsvenska dagbladet' achieved 46.75% and the most accurate betting company achieved 50.79%. However, despite this they state that it is not suitable to compare human prediction with statistical models but simply a benchmark. Although, this is a debatable topic where some claim prediction models need a human element or participation to ensure no bizarre results [26].

A common comment in the conclusions of contemporary methods is that their model will never be perfect, but a suggestion would be to include analysis of social media on matchday to bring a human element back in. A report released by the department of Computer Science at STM IST in India [27] devises another prediction model and conclude that they should bring sentiment analysis of Twitter to further enhance the model.

5 Theory of Sentiment Analysis

5.1 Introduction to Sentiment Analysis

Sentiment analysis is the process of recognising affective states and opinions from speech or text using techniques from the fields of natural language processing (NLP) and computational linguistics. In Psychology, affective states are constructs on a two-dimension model comprised of valency (positive or negative words) and arousal (intensity of reaction to stimuli), this reflects a mood. This model can be applied to speech and text to identify the affective state of the end-user.

The goal of natural language processing and computational linguistics is for a computer to achieve human-like comprehension of text and speech. Popular successful implementations of sentiment analysis have been to capture the opinions and attitude of a customer through online discussion and reviews. Retailers and services can tailor products based on feedback to enhance the 'Voice of the Customer'.

Sentiment polarity is about determining if a piece of text is positive, negative or neutral. This can be calculated using sentiment scores generated by interpreting text, another option is discretely classifying the sentiment polarity of a text. There are a range of methods to reach a sentiment polarity, most prominently the use of machine learning classification or lexicon dictionary-based sentiment scores.

5.2 Machine Learning Approach

Generally, sentiment analysis using machine learning techniques use a classifier which has been trained with supervised learning. Supervised learning is where a model is given both the raw data and the desired outcome. For instance, the sentence "I am having a terrible day today" should be classified as negative polarity so the model is given the sentence and the negative polarity as a pair, this allows the model to be adjusted through 'training'. There are many options for the composition of the model which may alter the accuracy of the resulting classification.

5.2.1 *Text Pre-Processing*

Here is an example tweet on Twitter from a Tottenham Hotspur fan after conceding a goal against Chelsea, through this section this example will be used to illustrate how the techniques are applied:

"Embarrassing so far, toothless and no ideas at all from midfield. Llorente might as well be a traffic cone #COYS"

Textual data contains vast amounts of redundant data which has no significant impact on the information density of the sentence. To combat this, each sentence is tokenised meaning each individual word is separated and considered separate. Many of these words are called 'stop words' which is a set of common words with less information than keywords such as 'the' and 'so'. A subset of the original text should be taken which does not include these stop words and other filters such as 'part of speech tagging' which allows specification of word class.

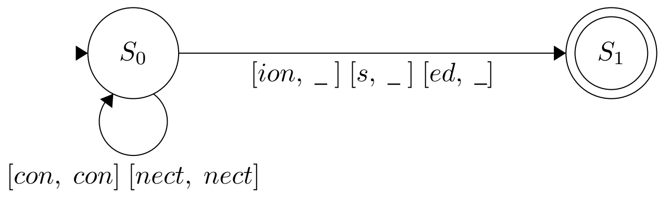
Fig. 1. Removal of stop words

["Embarrassing", "so", "far", "toothless", "and", "no", "ideas", "at", "all", "from", "midfield", "Llorente", "might", "as", "well", "be", "a", "traffic", "cone", "#COYS"]

["Embarrassing", "toothless", "ideas", "midfield", "Llorente", "traffic", "cone"]

Fig. 2. Finite State Transducer: Stemming Example

$$\Sigma \;=\; \{morpheme \mid morpheme \in English\_Morphemes\}$$


S_0
S_1
$[ion,\ \_]\ [s,\ \_]\ [ed,\ \_]$
$[con,\ con]\ [nect,\ nect]$

con	nect	ion	_	_	...
Input Tape: 'connection'					

con	nect	_	_	_	...
Output Tape: 'connect'					

Once the meaningless words are removed the remaining keywords must be normalised. Morphology is the study of morphemes which are linguistical units of the smallest meaningful section of a word. Normalisation can be achieved through stemming or lemmatisation where each inflected word (a word has changed form from the stem word) is substituted by their word stem or a common morpheme. This ensures that similar words can be grouped together for subsequent stages.

Stemming works by removing the suffix from inflected words to convert it to the stem form. Stemming is implemented using a Finite State Transducer (FST) which is a computational model used to show how an input string can be mapped onto an output string. A FST is a finite collection of states connected by transitions which have rules that must be met to reach another state. A 'tape', similar to a concept used in Turing Machines, holds different symbols of the alphabet (local available symbols) in cells along its infinite length (one or both directions). The tape is read by a read/write head which iterates along the tape in a single direction. The current symbol read by the read/write head is given to the current state, if the current symbol is the same as the input label found on a transition, the state changes and the output label on the transition is written to the output tape. The states and transitions can be arranged to fit the morphology of a given language allowing tasks such as stemming to be performed:

This is a crude heuristic process susceptible to over-stemming and under-stemming. For example, the words 'connection', 'connects' and 'connected' all stem to the word 'connect', this works perfectly as all instances are gathered and the information remains in the sentence. However, errors can occur with the amount of suffix being removed. Under-stemming is where not enough of the suffix is removed meaning the word does not completely reduce to stem form meaning a common word may not be reachable. Over-stemming is where too many characters are omitted causing loss of information or completely different words to be matched due to the common characters.

Lemmatisation literally means to resolve a word to its lemma meaning its form found in a dictionary. This is a simpler approach where a large lexicon of known words is stored and their associated morphology. WordNet created by Princeton University is a popular example of this. Words are looked up and converted to their stem form. This is a very clean approach which is guaranteed to keep the intended information. Despite lemmatisation being the preferred option for pre-processing, it is significantly slower and uses more memory. Additionally, it is unable to process unknown words where stemming can simply apply the same rules as usual.

Fig. 3. List of words after pre-processing

["Embarrassing", "toothless", "ideas", "midfield", "Llorente", "traffic", "cone"]

Finally, frequency analysis is performed to record the number of instances of each remaining word. This is called a 'Bag of Words' which is a key representation of long text passages in the NLP field due to being salient most important features retained for the problem domain and discrimination enough data remains to identify patterns for a classifier [28].

TABLE 1
Bag of Words

Word	Frequency
"embarrass"	1
"toothless"	1
"idea"	1
"midfield"	1
"llorente"	1
"traffic"	1
"cone"	1

5.2.2 Classifier Algorithm

There are many different choices for the internal structure of the classifier. A popular choice of algorithm in sentiment analysis is Naïve Bayes. Bayes' theorem is used to describe the probability of an event happening given that another prior event — potentially related — impacts the probability of the event. Bayes theorem can be expressed with the equation:

$$P(H|E) \ = \ \frac{P(E|H) \ P(H)}{P(E)} \tag{1}$$

$P(H|E)$: Posterior - how likely is the hypothesis (H) given all observed evidence (E)?
$P(E|H)$: Likelihood - how likely is the evidence (E) given that the hypothesis (H) is true?
$P(H)$: Prior - how likely was the hypothesis (H) before the evidence (E) was observed?
$P(E)$ Marginal - how likely is the evidence (E) occurring alone?
[29]

Naïve Bayes is an implementation of Bayes' theorem to act as a classifier to discriminate a class of an object based on features observed. 'Naïve' refers to the assumption that each feature is independent of each other, meaning the presence of a feature does not affect the appearance of another, in reality this is regularly violated [28]. However, this large assumption simplifies the mathematics.

For sentiment analysis we can treat the hypothesis as a polarity and individually determine the greatest probability of positive, neutral and negative text. The evidence is the different words or features in the bag of words created. Let's assign all our evidence to a single dimensional feature vector called E with a length of n.

$$E \ = \ (e_1, e_2, e_3, ..., \ e_n) \tag{2}$$

The independence assumption allows us to expand the original Bayes theorem to support many features as the probability of two independent events both occurring is the product of their respective probability.

$$P(A \cap B) \ = \ P(A)P(B) \tag{3}$$

Hence, Bayes theorem supports the feature vector:

$$P(H|e_1, \ ..., \ e_n) \ = \ \frac{P(e_1|H)P(e_2|H)... \ P(e_n|H)P(H)}{P(e_1)P(e_2)...P(e_n)} \tag{4}$$

Thus, the numerator can be simplified using a product operator:

$$P(H|e_1, \ ..., \ e_n) \ = \ \frac{P(H)\prod_{i=1}^{n} P(e_i|H)}{P(e_1)P(e_2)...P(e_n)} \tag{5}$$

As the evidence is the same for different hypotheses, the denominator is constant meaning the posterior probability is directly proportional to the numerator:

$$P(H|e_1, \ ..., \ e_n) \ \propto \ P(H)\prod_{i=1}^{n} P(e_i|H) \tag{6}$$

H - the hypothesis is the class variable. In order to determine the most probable class, the maximum probability of each hypothesis must be found. The arguments of the maxima (abbreviated to argmax) can be used to find the class with the maximum value of the posterior probability.

$$H = argmax_H P(H)\prod_{i=1}^{n} P(e_i|H) \tag{7}$$

This final output of the class variable will be the sentiment polarity of the bag of words. [30]

5.3 Lexicon Dictionary-based approach

A lexicon is a collection of words in a vocabulary. Different words have their associated polarity and intensity. By manually assigning a score of how positive or negative a word is, we can sum the score of each word in a sentence to calculate the polarity of the entire sentence.

Natural Language Toolkit released a sentiment analysis tool called Valence-Aware Dictionary sentiment Reasoner (VADER) [31]. VADER has a collection of nearly 8000 words commonly used on social media, this includes common abbreviations, slang terms and emoticons. This has all been manually entered and reviewed, these words are found in a large dictionary data structure as follows:

Fig. 4. VADER Lexicon format

```
}
debt       : -1.5
decay      : -1.7
decayed    : -1.6
decayer    : -1.6
decayers   : -1.6
}
```

Each word is rated a score between -4 and +4 representing negative to positive. Each score is an average of many human reviewers, another instance of instance of wisdom of the crowd. The sentiment polarity of a sentence is calculated by combining the score of all the meaningful words in the sentence. The sentiment analysis tool gives 4 different metrics: positive, neutral, negative and compound. Each of the metrics are calculated by summing the score of relevant words (e.g. positive metric only assesses positive words). The compound metric is the advised analysis metric as it combines all words in a sentence to a score which ranges from -1 to +1. Each of these metrics must be normalised as the sum of scores is likely out of the bounds of -1 and +1. Compound polarity scores are generated in the following way:

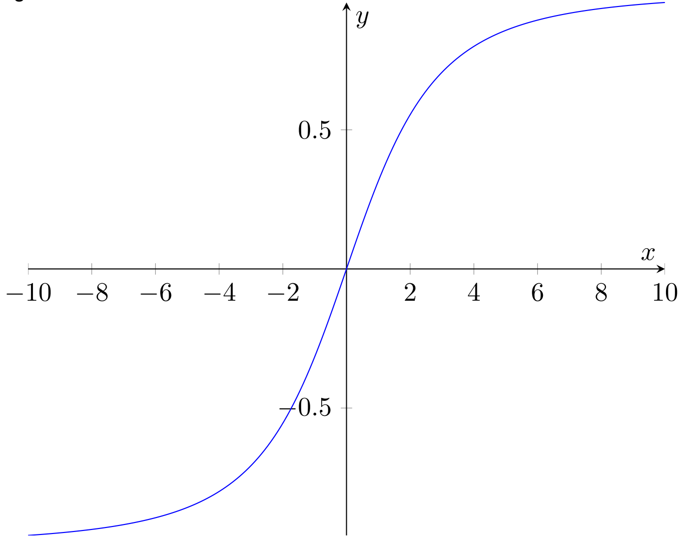
$$x \ = \ \sum p \tag{8}$$

p is the sentiment polarity of a single word. x is the sum of the polarity ratings in a sentence. For passages with multiple sentences, compound scores are calculated for each sentence and the mean compound score is used.

$$y = \frac{x}{\sqrt{x^2 + \alpha}} \tag{9}$$

To normalise the x values between -1 and 1, the x values are entered into a function. α is the normalising parameter which governs the horizontal stretch of the function. y is the compound polarity score; the normalising function is graphed in Figure 5.

Fig. 5. VADER Normalisation Function



The reviewers were also given sentences with varying punctuation and capitalisation to determine the polarity change with different punctuation and heavy usage of capitalisation. These features require explicit handling through five different heuristics [32].

The first heuristic is the inclusion of punctuation. Notably the inclusion of excessive punctuation which is a common feature of social media grammar. Exclamation and question marks often act as emphasisers of the emotion of a sentence. For example, the sentences “I hate this.” and “I hate this!!!” the latter clearly has more emotion and more negative. The heuristic accounts for this by emphasising the calculated compound score. An emphasiser makes a positive polarity even more positive and a negative polarity even more negative. Specifically, VADER amplifies the score by 0.292 for each exclamation point and 0.180 for every question mark. Capitalisation is the second heuristic. Heavy use of capitalisation brings more emotion into a sentence as it is understood as shouting. The polarity of each capital word is amplified by 0.733 which will increase or decrease the polarity of the sentence.

The third heuristic is less straightforward. Degree modifiers alter the polarity of a sentence depending on the intensity of words which precede adjectives. For instance, the phrase “this is somewhat fun” and “this is extremely fun” they both change the magnitude of the polarity of the word ‘fun’. VADER has dictionaries of degree modifiers called boosters increase magnitude of adjective and dampeners decrease the magnitude. VADER has many rules on different modifiers and their relative distance to the adjective. Multiple modifiers in the same sentence will affect the polarity of the entire sentence.

One obvious flaw of simply assessing the words in a sentence is the different intensities of certain clauses of a sentence. Sentences with contrasting clauses can make it ambiguous to find which clause carries the important information. The sentence: “Usually the service is brilliant but recently it has been terrible.” highlights the importance of recognising the important clause, every word after ‘but’ holds the vital information. VADER reduces the polarity of all words by 50% and increases the latter words by 150% for emphasis.

The final heuristic used is handing of negation. Placing worsts such as ‘isn’t’ or ‘not’ in proximity to other words will reverse the polarity. Bag of words models often remove negator words meaning they will just determine the polarity of the latter word which is not accurate to the sentiment. VADER handles negation by multiplying the negated word by -0.74.

5.4 Wisdom of the Crowd Approaches

The focus of this report is to assess the value of wisdom of the crowd in football prediction. Despite the desire for a more human or a realistic element in conventional methods, the evaluations of applying wisdom of the crowd techniques to football prediction are all very recent. All of the investigations use sentiment analysis of the Twitter social media platform.

A collaboration between the Computer Science department of the University of Texas and the Central Connecticut State University assessed the exact topic [33]. They used Twitter’s official streaming API to collect each tweet which contains certain keywords in real-time. Tracking football fans on Twitter is simple because of official club hashtags which each fan of a club will include in their tweet. Across their study period they collected 18,027,966 Tweets.

Each tweet was given a sentiment polarity rating and a subjectivity rating. A tone rating is a measure of the subjectivity or objectivity of a phrase. Subjectivity is where a phrase is based on opinion and emotion such as the reaction of a Newcastle United fan to their line-up against Liverpool.

Fig. 6. Subjective Tweet Example

```
``Disappointed with the line-up, set up too
   defensively at home and no Shelvey''
```

Objective phrases are unbiased and purely state facts. The following example is by a general football news Twitter account after the Newcastle line-ups were released.

Fig. 7. Objective Tweet Example

```
  ``Ex-Gateshead loan player @PaulDummett
 starts for #NUFC against #LFC tonight.''
```

Both of these metrics were combined in different proportions to create a prediction model. The following diagram displays how eight different models are produced from the metrics. The location of each numbered model defines the parameters of filtration where only tweets which satisfy the respective tone and polarity are taken.

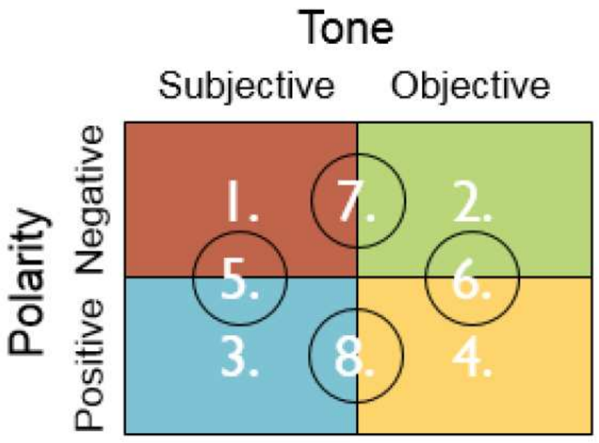


Fig. 8. Sample space diagram of models for Tweet analysis

To determine the tone and polarity they used the tool 'Opinion Finder' which utilises Naïve Bayes classifiers [34] to determine polarity and tone. They did not disclose whether the polarity classifier was trained on their football data, this may cause differences in score. By combining the sentiment polarity of Tweets about the involved club (e.g. #FFC for Fulham) and Tweets about the match being played (e.g. #FULCHE for a match were Fulham plays Chelsea at home) they evaluated different research questions.

The first research question was: "what signals exist in Twitter data that may provide match predictions?". They simply evaluated the accuracy of each of the 8 models and calculated the final pay-out of a series of hypothetical bets of $100 on the outcome of each match. Accuracy is simply the number of correct predictions divided by the total number of predictions. Each model was compared to a baseline model which is based on the odds of betting companies, these odds were also used to determine pay-out of the other models. The pay-out was determined by placing the $100 bet on the most likely team to win, if there are no tweets in a model then no bets were placed.

122 matches were recorded, the baseline model predicted 80 correct meaning an accuracy of 65.57% and a pay-out of $1,887.88. None of the 8 models exceeded this baseline accuracy, however these results were used to determine which sentiment features reflected the gameplay. The greatest accuracies came from model 1 (50.49%) and model 6 (50.00%) however this does not entirely reflect the pay-out. Model 1 only paid-out $46.82 more than the baseline while model 6 paid $816.75 over the baseline prediction. Only three of the eight produced less money than the baseline. Only model 5 gave a loss of $195.94, a substantial $2,083.42 less than the baseline.

The consistent high pay-out suggests that bets with higher odds were taken where the baseline model is reluctant to take risks. Perhaps, fans on Twitter give a better indication whether a shock result will occur where the underdog team wins.

The second research question was "what role does sentiment magnitude have on successful match prediction?". This can be inferred by sentiment polarity and the goal-difference of the game. It was expected that the greater difference in sentiment polarity would convey a greater goal difference. They found that model 7 and 8 both display this feature, however model 7 shows the goal difference increased when the fans were more negative. Although their tests use the difference between two teams rather than the polarity of a single club which will likely produce different results.

Overall there are many learnings which can be applied to an investigation later in this report. The first points are the process of collecting Tweets. The official Twitter API (application-programming interface) provides a suitable and customisable bridge to twitter based on keywords. They emphasise the importance of a vast number of Tweets. Determining sentiment polarity with a score is more useful than simple classification of a label as scores can show intensity, potentially relating to goal-difference. Also, there are many different features of sentiment analysis such as tone which also reflect the emotion of the fan. The pay-out calculation is relatively arbitrary because a single high-odds bet can distort the result significantly. The accuracies were lower than the baseline model, however the predictions made are unbiased as they have no prior knowledge about the football team. Perhaps combining the balanced approach of a conventional model which uses player and match data with the impartial sentiment analysis approach can enable a greater accuracy.

6 INVESTIGATION

6.1 Introduction to Investigation

In order to evaluate the success of a proposed prediction method, an original investigation was performed. In this investigation, the proposed prediction method was to predict football matches using the passive form of wisdom of the crowd using machine learning techniques. The objective was to determine the sentiment polarity of a fanbase and then use that score to predict the outcome of the game.

A machine learning model was trained and tested to determine its accuracy to evaluate how it compares to conventional methods. Additionally, any interesting observations between social media and in-game events were recorded.

The full listing of the Python program is available in appendix B.2 and GitHub repository:
"https://github.com/liamhbyrne/twitter-football-prediction"

6.2 Investigation Objectives

1) **Gather all tweets released by fans through a specific time-frame.**
    - 1.1. Create a link to Twitter's streaming API (application-programming-interface) to receive live tweets which contain certain keywords
    - 1.2. Aggregate all tweets collected and store persistently for later use
2) **Determine a sentiment polarity of a club's fans.**
    - 2.1. Extract text and tokenize each sentence from each tweet
    - 2.2. Calculate an average polarity rating of the sentences.
    - 2.3. Average all polarity ratings from the set of fans.
3) **Produce a range of graphs to demonstrate the relationship between sentiment polarity and actual events in the football match.**
    - 3.1. Produce a bar chart to visually compare average sentiment polarity between 2 competing teams
    - 3.2. Produce a line-graph of sentiment polarity against time to demonstrate a relationship to in game events.
    - 3.3. Analyse the vocabulary used with frequency analysis
    - 3.4. Plot commonly used words onto a word cloud to illustrate sentiment from fans.
4) **Correlate the goal difference and the average polarity rating of a club's fans.**
    - 4.1. Train a linear regression model with the independent (goal difference) and dependant (sentiment polarity) variables.
    - 4.2. Predict the goal difference of the football match by inputting the measured average sentiment polarity of a club.

6.2.1 Required Python Libraries

- **Tweepy** (v3.8.0): Provides access to the official Twitter restful API called 'Filter real-time Tweets' with standard access supporting up to 400 keywords on a single connection.
- **JSON** (built-in library): Allows manipulation of JSON data structures received from API.
- **NLTK** (Natural Language Tool-Kit, v3.4.5): Provides various natural language tools, specifically part of speech tagging is used.
- **TextBlob** (v0.15.2): Supports supervised learning of Naïve Bayes Classifiers specifically for sentiment analysis.
- **VaderSentiment** (v3.2.1): Large lexicon with sentiment polarities based on social media. Calculates polarity scores of phrases.
- **NumPy** (v1.17): Supports large multidimensional arrays of data. Hosts various mathematical functions. A constraint of TensorFlow.
- **TensorFlow** (v1.14): Open-source libraries of machine learning tools.
- **Matplotlib** (v3.1.1): Graph plotting library, used for line and bar graphs. Constraint of WordCloud library.
- **WordCloud** (v1.5): Plots words to a stencil on a Matplotlib graph to produce a custom word cloud.

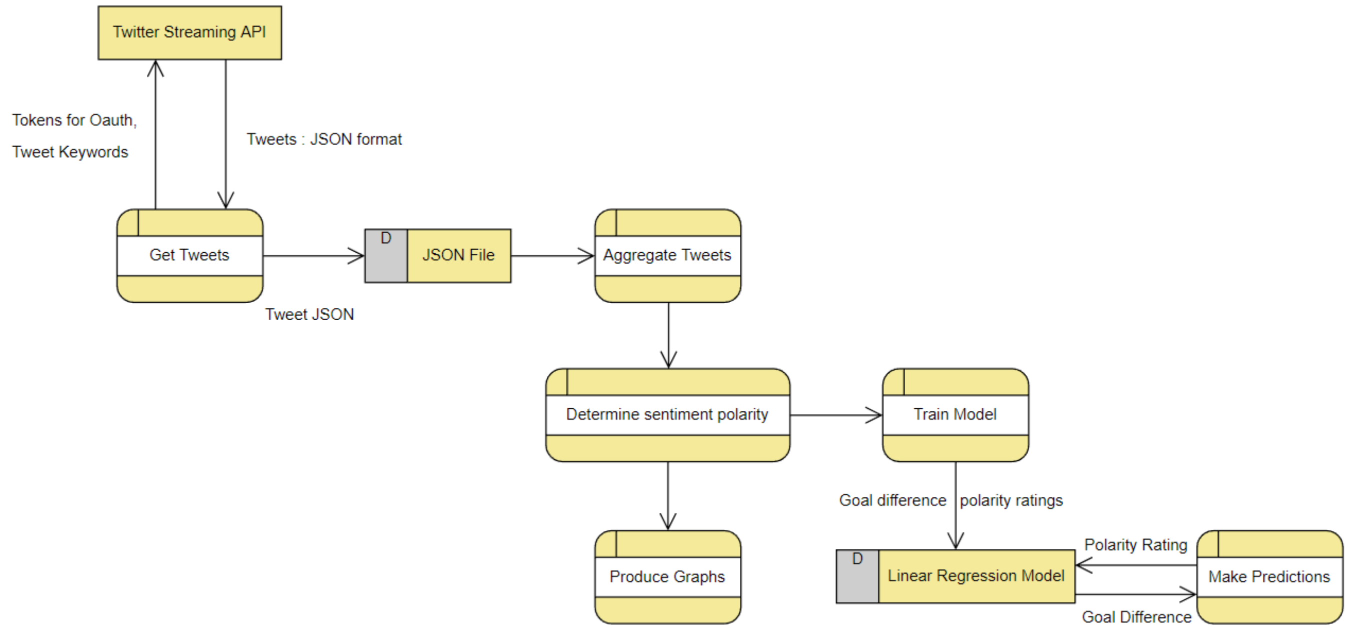


Fig. 9. Data Flow Diagram

6.3 System Overview

6.3.1 System Structure

The data-flow-diagram (DFD) in Figure 9 illustrates a system which allows each objective to be achieved. This was implemented in the Anaconda distribution of Python 3.7 programming language as it is well supported by external libraries which contain specialised tools for data-science.

6.3.2 Getting and Aggregating Tweets

Twitter developers offer a free version of a streaming API (Application-Programming-Interface) which gives developers access to the throughput of all newly posted Tweets which match a set of keywords specified such as '#CPFC' for Crystal Palace Football Club fans. The API provides a software intermediary between the server-side Twitter throughput and a client. The Python Library Tweepy provides access to the Twitter API. Each tweet is received in a JSON (Java-Script-Object-Notation) data structure which contains all of the details about the Tweet. An example of a received tweet is seen in Appendix A.1, the tweet was posted by a Tottenham Hotspur fan during the semi-final of the Carabao Cup against Chelsea.

The tweet is representative of the majority of the throughput, however extended tweets which exceed 140 characters have a slightly altered structure. For this investigation the only important structures are the "text" for the sentiment analysis and "created_at" to identify the individual tweet. Each tweet received, is written to a json file which will contain all incoming tweets. This file can be held persistently for later use. This file can be parsed to collect each tweet which can be abstracted to collect just the time stamp and the text content. The many pairs of timestamps and text can be stored in a dictionary data structure:

Fig. 10. Dictionary of Tweets, KEY: Timestamp, VALUE: Text

```
 {
'Thu Jan 24 19:48:29 +0000 2019' : 'Hearing
the spurs fans so loud is just incredible
#COYS',
'Thu Jan 24 19:48:30 +0000 2019' : 'I hope
@SpursOfficial can take the sting out of
marking #COYS'
}
```

6.3.3 Determine Sentiment Polarity

Before sentiment analysis is performed, general text pre-processing and filtering is required. The pre-processing task is to remove meaningless tags and references such as hashtags, hyperlinks and Twitter usernames. The popularity of Twitter has attracted mass marketing. In the eyes of sentiment analysis, marketing tweets offer no indicating features that would reflect the football match. Retweets are when a user posts the tweet of another user onto their profile usually to re-affirm what is being said in the tweet. However, retweets are simply duplicate tweets which can distort the sentiment of the whole audience. Both marketing and retweets are both omitted through common keywords and manually blacklisting accounts.

6.3.4 Choice of Sentiment Analysis Model

In the previous sentiment analysis section, a Naïve Bayes classifier and a VADER lexicon approach were detailed. A Naïve Bayes classifier was used to demonstrate how an accurate football-based sentiment classifier can be built. The VADER lexicon produces a quantitative score rather than a label (e.g. Positive) denoting intensity. The quantitative score allows precise analysis of sentiment, furthermore the hypothesis that sentiment polarity is directly proportional to goal-difference (report by Texas and Connecticut university [33]) can be tested. The issue with a Naïve Bayes classifier is that there may be an insufficient amount of training data to satisfy accurate sentiment analysis requirements, VADER provides a gold standard social media lexicon that can be trusted thus not affecting subsequent prediction accuracies. TextBlob is natural processing library that supports Naïve Bayes classifiers in linguistics. The training of the model is supervised meaning each phrase must be manually labelled. The tweets are trained in bags of words using all techniques detailed in the text-pre-processing section above. In the example in Appendix A.1, a limited number of tweets were used as it is simply a proof-of-concept. An accuracy of 65% was achieved in a 60/40 partition between training and testing data. Clearly 65% is not sufficient to attempt to analyse the sentiment of a large audience, this highlights the advantage of using VADER. All code used in the classifier investigation is reproduced in Appendix A.1.

VADER produced a compound score over each tweet, a positive polarity and negative partition can be made across the scores to classify the polarity. The positive partition is 0.0 and up and the negative partition is all negative numbers. Through this, all 8 were predicted correctly (100% accuracy). However, this is a very small test-set so accuracies lack precision but is still representative.

Through VADER a tweets compound score is assigned to each tweet:

Fig. 11. Dictionary values have text coupled with polarity score in tuple

```
 {
`Sun Feb 10 17:05:36 +0000 2019`:
(For this season, we will get into champions
league and maybe even nick a cup or two.`,
0.26335)
}
```

6.4 Prediction Model

A prediction model is needed to take all the features from the data collected and come to a decision about the most likely outcome of the football match. The output from the model will also be quantitative in order to predict the goal difference.

In 2019, the run-up to 130 games were recorded from January to May. In the 258MB of data captured - 68,360 tweets were recorded. The model was given 70% (91 games) of the matches with their respective polarity and final goal-difference for supervised training. The remaining 30% (39 games) were used to test the model.

Tweet collection started one hour prior to kick-off (at time of line-ups being released) until kick-off. Once the collection period is complete, sentiment analysis was performed then each tweet was assigned its polarity score. The mean polarity score was taken across the entire audience. The mean polarity score must be tested for correlation against the final goal-difference of the game. A linear regression model was chosen as it supports bivariate data. Linear regression is about adjusting the parameters on the equation of a straight line to fit the plotted data.

To illustrate this relationship, all of the captured games were placed on a scatter graph (Figure 12). To demonstrate a linear regression line, a least-square regression line is drawn to fit the model most appropriately. Least-squares works simply by adjusting the line until the square of each distance between the line and each point is at its minimum, squaring is used to account for negatives. The correlation is very weak, as expected, but there is a slight positive correlation which fits the assumption the relationship between polarity and goal difference.

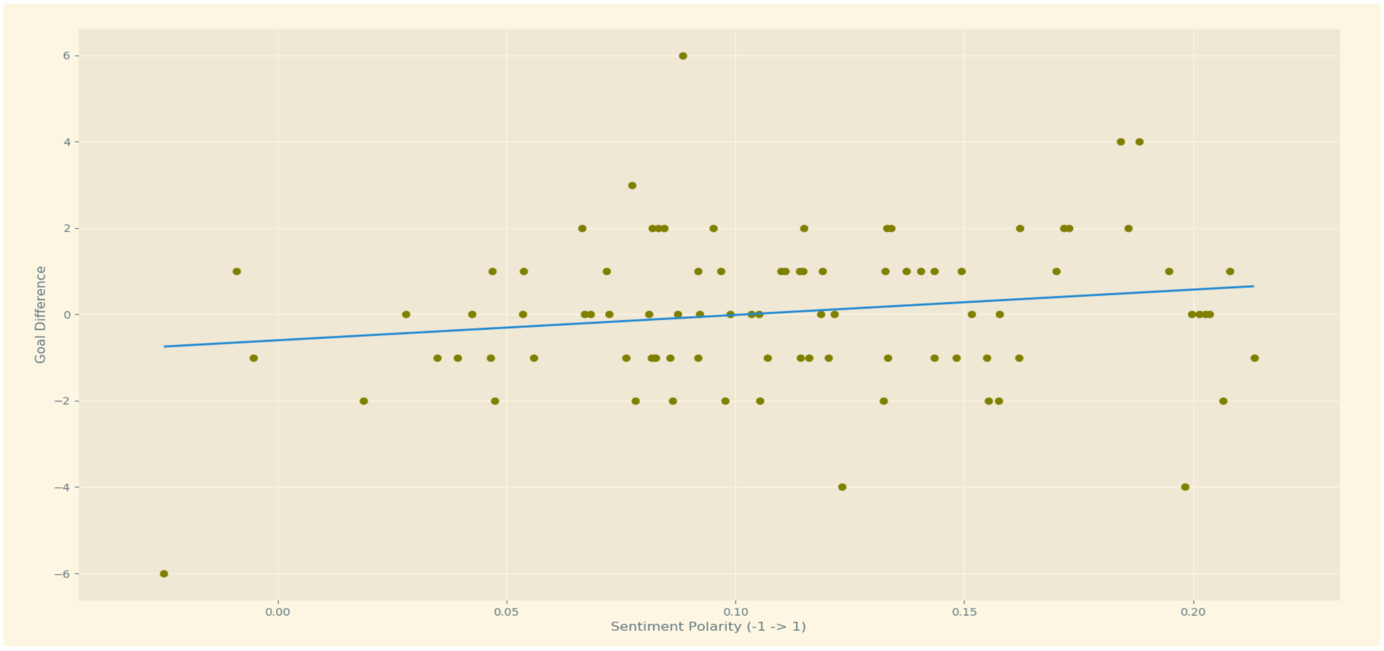


Fig. 12. Goal Difference plotted against Sentiment Polarity, illustrating the Linear Regression Model with a Least Squares Regression Line

However, the prediction model used does not calculate a regression line using the statistical method of least-squares instead it fits the data. Linear regression can be viewed as a statistical or as a machine learning technique it simply depends on how the regression line is made. Statistical Methods use the familiar equation of a straight line:

$$y = mx + c \tag{10}$$

Where the relationship between the feature x and the label y is governed by m which changes the gradient of the line and c which governs the point of interception on the y-axis.

Machine Learning methods use an almost identical equation but with different meanings [35].

$$y' = b + \omega_1 \tag{11}$$

Where y' is the predicted label, b is the bias (y-intercept), ω_1 is the weighting (gradient) of the first x feature. The features and weights are numbered as the equation supports multivariable regression. Each weight acts as a co-efficient to the feature, by adjusting the co-efficient of each weight the expression can be calibrated to reach the specific value, or y' value. The difference between the target value and current value is the error. A cost function is used to model the error during calibration. This model uses the MSE (mean-squared error) as a cost function which works by calculating the mean of the squared differences between a regression line and a point. Below shows the equation (12) to calculate the error, N is the number of data points.

$$MSE = \frac{1}{N} \sum_{i=1}^{n} \left(y\prime_i - (\omega x_i + b)\right)^2 \tag{12}$$

Gradient descent is a method of attempting to find the minimum or the optima of an unknown function by repetitively taking samples and stepping across the x-axis by altering parameters (weights) until the lowest result is achieved. This is applied to cost functions to minimise the error to best fit a regression line to the data.

6.5 Evaluation

To evaluate the success of the model, the test set was fed into the model and the predictions recorded. The polarity of each team was entered, and the expected goal-difference rounded to the nearest integer. An accuracy score was calculated by determining the success of outcome predictions and exact goal difference predictions.

In Appendix B.1, the full testing results are presented. The outcome predictions were correct 38.9% on each team, only 6% above random (33%). The exact goal-difference predictions were correct 27.8% on each team. From observing the behaviour of the model, it appears to favour the draw outcome. 86% of the test set was predicted as a draw, this is likely because of the regression line having a shallow gradient meaning very extreme polarities are needed to sway the model. Although, the goal difference predictions benefitted from the draw predictions as there is only one possible goal difference.

Despite the low accuracy of this model and the reviewed wisdom of the crowd model, sentiment analysis may still warrant a place in football prediction. Below is a collection of graphs and figures which reveal the connection between social media and football matches.

The first figure is a custom generated word-cloud to attempt to identify key topics and talking points of fans. Frequency analysis of words (excluding stop words) was performed on each tweet with positive or negative polarity. Part of speech tagging allows specific word classes to be displayed such as proper nouns which include names of players. Each word was plotted onto an aesthetic stencil.

The left word-cloud in Figure 13 is filled with words in negative tweets by Liverpool fans before a 3-2 win against Newcastle (04/05/19). Interesting words are 'Klopp', the Liverpool manager and 'Firmino' who is a regular forward in their line-up. Perhaps Liverpool fans were angry about Jürgen Klopp's decision to not include Roberto Firmino in their starting line-up.

On the right is a word-cloud filled with words from Tottenham fans, hence the stencil, during the Carabao Cup semi-final against Chelsea. There are 750 words entered meaning many words with less interest appear.

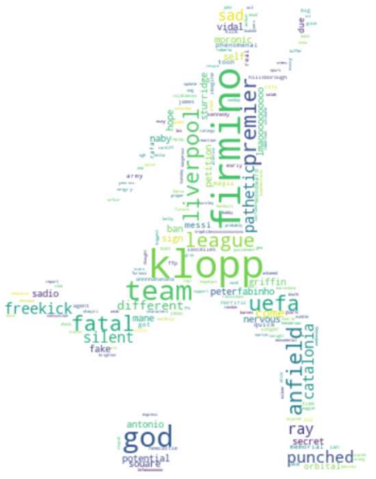


Fig. 13. Liverpool Wordcloud - Negative Tweet content

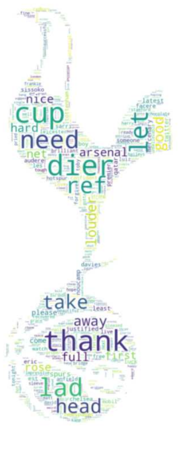


Fig. 14. Tottenham Wordcloud - illustrates the need to abstract words of lesser sentiment

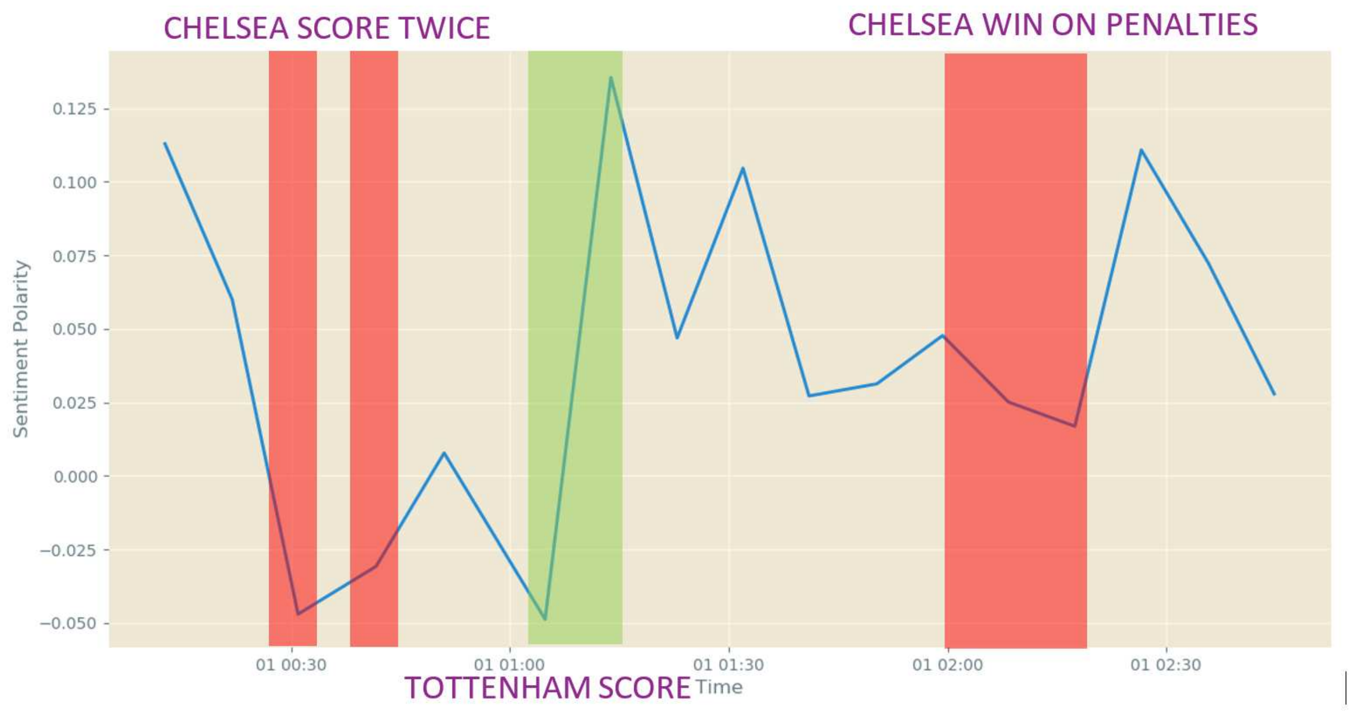


Fig. 15. Annotated Sentiment Polarity against Time graph with key in-game events

The polarity-time graph above in Figure 15 shows the average polarity by Tottenham fans in 10-minute intervals across the same Tottenham game as featured in the word-cloud. Tottenham appear to come into the game with confidence at a polarity of 0.1125; Tottenham conceded goals at minute 27′ and 38′, this is clearly seen with the dramatic drop in sentiment polarity to negative numbers. The polarity appears to neutralise over half-time reaching 0.0. Once the game starts after half time, Llorente scores for Tottenham (50‘). This goal explains the enormous climb as the polarity is inverted to positive. Gradually the polarity begins to neutralise as no goals are scored by either side and the prospect of deciding the game by penalties is harrowing. Penalties are clearly not easy viewing for the fans where they lose 4-2 reflected by the further drop. After penalties, fans appear to appreciate their progress in the competition and hope for redemption rather than anger or sadness hence the rapid increase in polarity.

The clear reflection of in-game events and polarity is very interesting. This could be used to suggest whether a goal is going to occur soon based on the current trend.

6.6 Limitations

The method used in the investigation had some drawbacks. Most limitations originate from the tweet collection process where the throughput of the Twitter streaming API is limited. Twitter only granted standard access to the API meaning that only 1 connection can be made at a time with a maximum of 400 keywords. Twitter prevents abuse of the API by preventing immediate re-connections (420 Error) using an adjustable timeframe, this time frame increases exponentially after each attempt. Random drops in internet connection and intermittent access to the API caused periods of no tweets being collected, this could prevent sentiment analysis being representative of the match.

Another drawback was that VADER fails to classify some important phrases correctly. For example, the phrases “What a goal!” or “Get in!” are both very common positive phrases although as there are no words in the phrases which hold any sentiment polarity meaning the phrases are classified as neutral.

7 Conclusion

The research question: "Can machine learning harness the 'Wisdom of the Crowd' to predict the outcome of football matches?" can be broken down into two constituent questions. The first question is "Can machine learning harness the 'Wisdom of the Crowd'?", and the second question is "Can 'Wisdom of the Crowd' predict the outcome of football matches?".

The first question relates to the investigation through the use of machine learning in sentiment analysis. While the approach taken in the detailed investigation was to use a lexicon-based solution for practical reasons (it would take a lot of time to gather and train a model) a shorter investigation using a Naïve Bayes classifier showed the viability of that approach. Precisely, an accuracy of 65% was achieved over a relatively small training and testing set. The theory of a Naïve Bayes classifier is extensively explored in the sentiment analysis section. Furthermore, implementations of the classifiers on other media such as movie reviews quote an accuracy of 84.84%. Despite not testing this question very extensively, there is enough supportive secondary data to say "yes".

The latter question is assessed in the investigation evaluation. A linear regression model achieved an accuracy of 38.9% on outcome predictions. This lower accuracy aligns broadly with the literature: the investigation by Texas and Connecticut universities achieved 50.49% which is also significantly below the 65.57% baseline accuracy. However there is a contribution to the low accuracy from the limitations of the investigation processes. Both investigations use Twitter as their analysis media, suggesting the use of passive 'Wisdom of the Crowd' on Twitter is not reliable and biased. However successful presidential election predictions have been made entirely through Twitter [7]. Therefore Twitter may not have a 'wise crowd' specifically for football despite behaving better than its other social media counterparts. The testing set documented in Appendix C shows how few negative average polarities of fanbases there are, even during matches where the team concedes a significant number of goals. Perhaps the emotions of football fans are not as fickle as expected, instead as supporters they are persistently positive during most occasions. The differences in volatility of sentiment polarity between each team after an in-game event makes it difficult to have a one fits all model for the Premier League, this is seen in the incredibly weak correlation in figure 6. Perhaps, the biased nature of the football supporter community prevents any diversity in opinion, hence the narrow range of sentiment polarity seen in testing. It can also be argued that the 'herd mentality' of supporter groups would promote the development of 'Group-think' within their communities. It can be concluded therefore that, passive 'Wisdom of the Crowd' techniques using Twitter do not perform well as standalone prediction models for football matches.

However some interesting discoveries during this investigation do back up the method as a form of prediction. Firstly, the sentiment polarity-time graph clearly reflects in-game events across many matches. Perhaps the trend of current sentiment polarity could predict goals. Secondly, frequency analysis of player names can highlight which players are being discussed, usually because of their performance. Potentially this could suggest players who are likely to score next or make other forms of contribution. Additionally, conventional prediction models which use player and match data may be able to provide balance. Finally, the pay-out of 'Wisdom of the Crowd' model in the literature review was often significantly greater, suggesting the model places high-odds bets whilst the conventional methods generally favour the favourite to win the match. Perhaps a new composite model with a fusion between the balance of conventional data prediction models and the divergent predictions of 'Wisdom of the Crowd' methods would be valuable. Overall the final answer to the full question is that, currently conventional statistical and machine learning methods yield a better accuracy than 'Wisdom of the Crowd' sentiment analysis, however for its ability to potentially foresee unexpected outcomes it should not be neglected.

LIST OF FIGURES

LIST OF TABLES

References

[1] Deloitte, “European football market worth a record € 25.5 billion (21.9bn) as Premier League leads the way in new era of financial stability,” Deloitte Press Releases, 7th June 2018. [Online]. Available: http://bit.ly/premierleaguefinance [Accessed 11th May 2019].

[2] Gambling Commission (Public body of U.K. government), “Industry statistics”, 19th December 2018. [Online]. Available: http://bit.ly/GamblingCommission [Accessed 11th May 2019].

[3] A. Barnes-Brown, “The Oracle of Delphi: How the Ancient Greeks Relied on One Womans Divine Visions,” History Answers, 28th June 2018. [Online]. Available: http://bit.ly/PythiaHistory [Accessed 11th May 2019].

[4] C. Hayashi, What is Data Science? Fundamental Concepts and a Heuristic Example, Springer Japan, 1998.

[5] M. G. M. G. Andrea De Mauro, “What is Big Data? A Consensual Definition and a Review,” Research Gate, p. 8, 2015.

[6] Aristotle, “Politics Book III,” in Politics, Loeb Classical Library, 350 BCE.

[7] D. C. A. K. Hao Wang, “A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle”, Computational Linguistics, vol. 50, pp. 115-120‘, 2012.

[8] F. Galton, “Vox populi”, Nature, pp. 450-451, 1907.

[9] J. Surowiecki, The Wisdom of Crowds, New York: Anchor Books, 2004.

[10] M. B. C. D. I. O. F. S. E. Smriti Bhagat, “Three and a half degrees of separation”, Facebook Research, 2016.

[11] Twitter, Inc., “EARNINGS PRESS RELEASE: Selected Company Financials and Metrics [Group]”, Twitter, Inc. Investor Relations, San Francisco, First Quarter, 2019.

[12] I. Janis, Victims of Groupthink, New York: Houghton Mifflin Company (July 1972), 1972.

[13] J. F. Kennedy, Interviewee, Concerning the Bay of Pigs. [Interview]. 6th April 1964.

[14] Kerry Flynn, IB Times, “STUDY: Teens Embrace Anonymous Social Networks To Discuss Awkward Topics, Build Confidence,” International Business Times, 8 August 2015. [Online]. Available: http://bit.ly/teensonsocialmedia [Accessed 28 August 2019].

[15] M. K. (NASA), Interviewee, LESSONS FROM NASA AND THE DANGERS OF ‘GROUPTHINK’. [Interview]. 6 July 2017.

[16] L. S. a. T. B. R. Dimitroff, “Organizational Behavior and Disaster: A study of conflict at NASA”, Project Management Journal, vol. 36, no. 2, pp. 28-38, 2005.

[17] P. Fairchild, “USER ARTICLE: The Toxicity Of Blue Tick Culture”, Medium, 1 October 2018. [Online]. Available: http://bit.ly/bluetickculture [Accessed 28 July 2019].

[18] UNITED STATES CONGRESS HEARING: Facebook, Social Media Privacy, and the Use and Abuse of Data, 10th April 2018.

[19] H. Kozlowska, “The Cambridge Analytica scandal affected nearly 40 million more people than we thought”, QUARTZ, 4 April 2018. [Online]. Available: http://bit.ly/cambridgeanalyticaa [Accessed 28 July 2019].

[20] H. Davies, “Ted Cruz using firm that harvested on millions of unwitting Facebook users”, The Guardian, 11 December 2015. [Online]. Available: http://bit.ly/guardianCambridgeAnalytica [Accessed 28 July 2019].

[21] Parliament of United Kingdom, Data Protection Act 1998, United Kingdom: Act of Parliament, 1998.

[22] Parliament of United Kingdom, Data Protection Act 2018, United Kingdom: Act of Parliament, 2018.

[23] Ofcom, “Online Nation 2019 Report”, 30 May 2019. [Online]. Available: http://bit.ly/ofcomNationalSummary [Accessed 29 July 2019].

[24] A. Hern, “Facebook usage falling after privacy scandals, data suggests,” The Guardian, 20 June 2019. [Online]. Available: http://bit.ly/guardianFacebook [Accessed 29 July 2019].

[25] R. N. Daniel Pettersson, “Football Match Prediction Using Deep Learning”, Chalmers University of Technology - Master’s Thesis, Gothenburg, Sweeden, 2017.

[26] E. O. Esumeh, “Using Machine Learning to Predict Winners of Football League for Bookies”, International Journal of Artificial Intelligence, 2015.

[27] H. M. Anand Ganesan, ENGLISH FOOTBALL PREDICTION USING MACHINE LEARNING CLASSIFIERS, International Journal of Pure and Applied Mathematics, vol. 118, no. 22, pp. 533-536, 2018.

[28] S. Raschka, “Naive Bayes and Text Classification I - Introduction and Theory,” Sebastian Raschka Jupyter Notebook, 2014.

[29] M. Rethana, “Bayesian Statistics and Naive Bayes Classifier Refresher”, Medium, 11 September 2018. [Online]. Available: http://bit.ly/bayestheoremexplained [Accessed 16 August 2019].

[30] S. Jain, “Naive Bayes Classifiers”, GeeksforGeeks, 3 March 2017. [Online]. Available: http://bit.ly/naivebayestheory [Accessed 17 August 2019].

[31] E. G. C.J Hutto, “VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text”, Research Gate Publication, 2015.

[32] P. Calderon, “VADER Sentiment Analysis Explained”, Data Meets Media, 10 April 2017. [Online]. Available: http://bit.ly/vadersentiment [Accessed 18 August 2019].

[33] A. T. J. a. C. S. L. J. Robert P. Schumaker, “Predicting Wins and Spread in the Premier League Using a Sentiment Analysis of Twitter”, Decision Support Systems, vol. 88, no. August 2016 Issue, pp. 76-84, 2016.

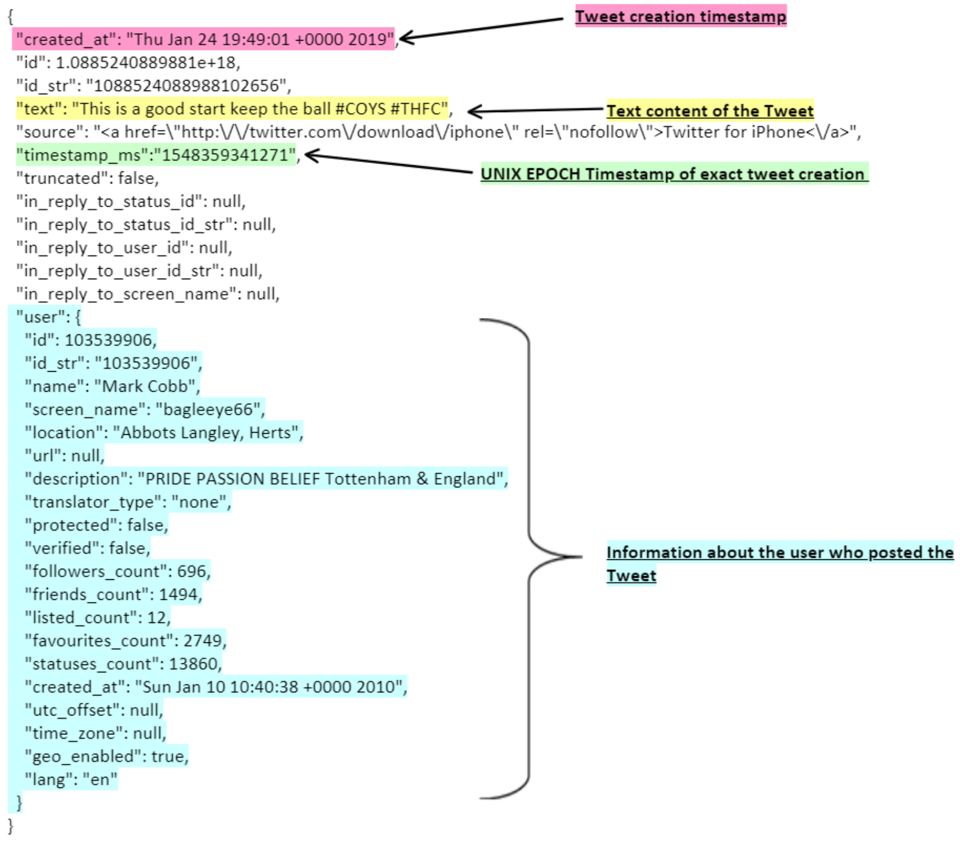
[34] “OpinionFinder: A system for subjectivity analysis”, in HLT/EMNLP on Interactive Demonstrations - Association for Computational Linguistics, Vancouver, British Columbia, Canada, 2005.

[35] Google Developers, “Descending into ML: Linear Regression”, 5 March 2019. [Online]. Available: http://bit.ly/linearRegression [Accessed 1 September 2019].

Appendix A
Twitter

A.1 JSON Format

A representative JSON received from the Twitter Streaming API

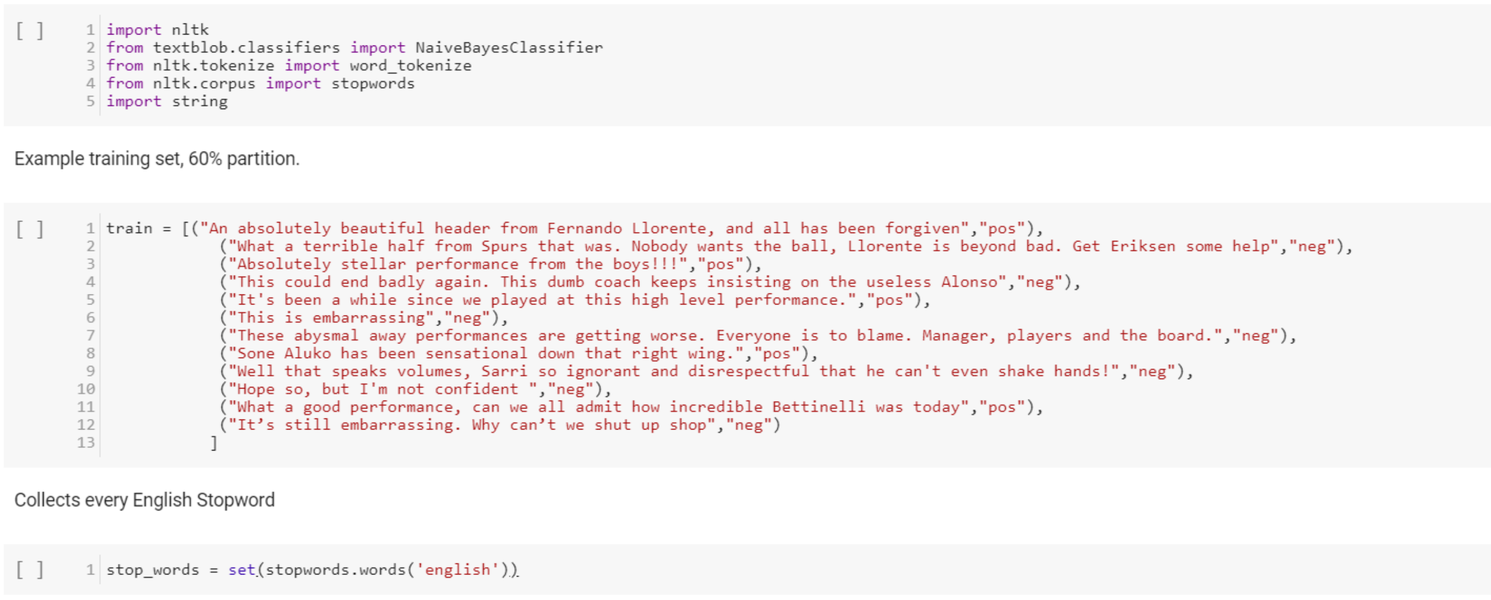

Tweet creation timestamp
Text content of the Tweet
UNIX EPOCH Timestamp of exact tweet creation
Information about the user who posted the Tweet

```
{
 "created_at": "Thu Jan 24 19:49:01 +0000 2019",
 "id": 1.0885240889881e+18,
 "id_str": "1088524088988102656",
 "text": "This is a good start keep the ball #COYS #THFC",
 "source": "<a href=\"http:\/\/twitter.com\/download\/iphone\" rel=\"nofollow\">Twitter for iPhone<\/a>",
 "timestamp_ms":"1548359341271",
 "truncated": false,
 "in_reply_to_status_id": null,
 "in_reply_to_status_id_str": null,
 "in_reply_to_user_id": null,
 "in_reply_to_user_id_str": null,
 "in_reply_to_screen_name": null,
 "user": {
  "id": 103539906,
  "id_str": "103539906",
  "name": "Mark Cobb",
  "screen_name": "bagleeye66",
  "location": "Abbots Langley, Herts",
  "url": null,
  "description": "PRIDE PASSION BELIEF Tottenham & England",
  "translator_type": "none",
  "protected": false,
  "verified": false,
  "followers_count": 696,
  "friends_count": 1494,
  "listed_count": 12,
  "favourites_count": 2749,
  "statuses_count": 13860,
  "created_at": "Sun Jan 10 10:40:38 +0000 2010",
  "utc_offset": null,
  "time_zone": null,
  "geo_enabled": true,
  "lang": "en"
 }
}
```

Appendix B
Python Listing

B.1 Naïve Bayes TextBlob Sentiment Analysis

A simple machine learning sentiment analysis notebook displayed using Google's Colab:



```
[ ]
import nltk
from textblob.classifiers import NaiveBayesClassifier
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
import string
```

Example training set, 60% partition.

```
[ ]
train = [("An absolutely beautiful header from Fernando Llorente, and all has been forgiven","pos"),
              ("What a terrible half from Spurs that was. Nobody wants the ball, Llorente is beyond bad. Get Eriksen some help","neg"),
              ("Absolutely stellar performance from the boys!!!","pos"),
              ("This could end badly again. This dumb coach keeps insisting on the useless Alonso","neg"),
              ("It's been a while since we played at this high level performance.","pos"),
              ("This is embarrassing","neg"),
              ("These abysmal away performances are getting worse. Everyone is to blame. Manager, players and the board.","neg"),
              ("Sone Aluko has been sensational down that right wing.","pos"),
              ("Well that speaks volumes, Sarri so ignorant and disrespectful that he can't even shake hands!","neg"),
              ("Hope so, but I'm not confident ","neg"),
              ("What a good performance, can we all admit how incredible Bettinelli was today","pos"),
              ("It’s still embarrassing. Why can’t we shut up shop","neg")
             ]
```

Collects every English Stopword

```
[ ]
stop_words = set(stopwords.words('english'))
```

Example testing set. 40% partition.

```
[ ]   test = [("That’s the best cross I’ve ever seen ","pos"),
               ("I have no confidence left","neg"),
               ("Tom Cairney has been placing beautiful passes all around the park","pos"),
               ("I feel sick","neg"),
               ("Regardless of what happens, so proud of that 2nd half. fingers crossed! Good luck boys!","pos"),
               ("WHAT A GOAL COME ON!!!!!","pos"),
               ("If Spurs win this, Pochettino - what a genius!!","pos"),
               ("We are dreadful without Kane, Dele and Son","neg")
          ]
```

Edits each tweet to lowercase and removes stopwords and punctuation

```
[ ]   for dataset in [train, test]:
          for tweet in dataset:
              tweet_words = word_tokenize(tweet[0])
              bagged_words = ([word.lower() for word in tweet_words if (word not in stop_words) and (word not in string.punctuation)], tweet[1])
              dataset[dataset.index(tweet)] = bagged_words
```

Model is given training set

```
[ ]   cl = NaiveBayesClassifier(train, format='csv')
```

Example classification

```
[ ]   print(cl.classify("Another incredible performance"))
```

```
pos
```

Model accuracy and most informative features

```
[ ]   cl.accuracy(test)
```

```
0.625
```

```
[ ]   cl.show_informative_features(30)
```

```
Most Informative Features
   contains(performance) = False             neg : pos    =      2.2 : 1.0
    contains(absolutely) = False             neg : pos    =      1.6 : 1.0
      contains(llorente) = True              pos : neg    =      1.3 : 1.0
          contains(this) = False             pos : neg    =      1.3 : 1.0
            contains(it) = True              pos : neg    =      1.3 : 1.0
  contains(embarrassing) = False             pos : neg    =      1.3 : 1.0
          contains(what) = True              pos : neg    =      1.3 : 1.0
         contains(aluko) = False             neg : pos    =      1.2 : 1.0
        contains(played) = False             neg : pos    =      1.2 : 1.0
         contains(admit) = False             neg : pos    =      1.2 : 1.0
         contains(since) = False             neg : pos    =      1.2 : 1.0
```

B.2 Full Investigation Python Listing

The entirety of the Python programs used in the investigation. Project can also be accessed on GitHub (URL in report)

```
// tweepy_streaming.py
from tweepy.streaming import StreamListener #Class that gathers tweets
from tweepy import OAuthHandler, Stream #Authenticator
import tweepy_creds
from abstract_tweets import eliminateSpam
import json

class TwitterStreamer():
    """
     Class for streaming and processing live tweets
     """

    def stream_tweets(self, hashtag_teams):
        listener = OverriddenListener(hashtag_teams) # Instance
        creds = OAuthHandler(tweepy_creds.CONSUMER_KEY, tweepy_creds.CONSUMER_SECRET) # Consumer keys given to authenticator
        creds.set_access_token(tweepy_creds.ACCESS_TOKEN, tweepy_creds.ACCESS_TOKEN_SECRET) # access token given
        stream = Stream(creds, listener) # Start Streaming, (authentication, object)        hashtags = ["#FFC", "#CFC",
     "#LUFC", "#NUFC"]  # Keywords
        stream.filter(track=[*hashtag_teams]) # Filter based on keywords

class OverriddenListener(StreamListener): #Class inherits the tweet gatherer
    """
     Handles incoming tweets
     """
    def __init__(self, hashtag_teams):
        self.hashtag_teams = hashtag_teams

    def on_data(self, data): #Takes data from listener
        datum = json.loads(data)
        if eliminateSpam(datum) == True:
            return True #Terminates addition
        datum_text = data.lower()
        print(data)
        for i in self.hashtag_teams:
            if i in datum_text:
                with open("" + self.hashtag_teams[i], ’a’) as file:
                    file.write(data)
        return True #Stop

    def on_error(self, status): #Print
        #encountered error
        if status == 420: #Catches error
            print("420 - LIMIT REACHED")
            return False
        if status == 401:
            print("UNAUTHORISED")
            return False
        print(status)

hashtag_teams = {’#ffc’:’ffctestsecondhalf.json’,
                 ’fulham’ : ’fulhamtestsecondhalf.json’} #keywords, LOWERCASE, Including hashtags

twitter_streamer = TwitterStreamer()
twitter_streamer.stream_tweets(hashtag_teams)
```

```python
\\abstract_tweets.py
import json
from typing import *
from basic_sentiment import *
from NaiveTextBlob import *

def gatherTweets(file_name: str) -> List:
    '''
     Moves tweets into List
     '''
    tweets = []
    with open("C:/Users/Liam/Documents/EPQ/json_data/data/" + file_name + '.json') as file:
        counter = 0
        for line in file: #Accounts for .JSON File spacing
            if counter % 2 != 0:
                counter += 1
                continue
            tweets.append(json.loads(line)) #Adds entire tweet information to list
            counter += 1
    return tweets


def abstractTweets(tweets: List) -> Dict:
    '''
     Adds to dictionary based on [date : text]
     '''
    abstracted_tweets = {}
    for tweet in tweets:
        if eliminateSpam(tweet) == True:
            continue
        if tweet["created_at"] in abstracted_tweets: #Accounts for duplicates
            attempt = tweet["created_at"] + '&' #Places symbols to prevent collisions
            while attempt in abstracted_tweets:
                attempt += '&'
            try:
                abstracted_tweets[attempt] = tweet["extended_tweet"]["full_text"]
                #generateTraining(tweet["extended_tweet"]["full_text"])
            except KeyError:
                abstracted_tweets[attempt] = tweet["text"]
                #generateTraining(tweet["text"])
        else:
            try:
                abstracted_tweets[tweet["created_at"]] = tweet["extended_tweet"]["full_text"]
                #generateTraining(tweet["extended_tweet"]["full_text"])
            except KeyError:
                abstracted_tweets[tweet["created_at"]] = tweet["text"]
                #generateTraining(tweet["text"])
    print(abstracted_tweets)
    return abstracted_tweets


def eliminateSpam(tweet : str) -> bool:
    '''
     Eliminate Irrelevant Tweets
     '''
    disallowed_accounts = []
    disallowed_terms = ['RT','stream','mobile','LIVE','Live','STREAM','HD','WATCH','watch']
    if tweet["lang"] != "en":  #Only accepts English
        return True
    if tweet["user"]["screen_name"] in disallowed_accounts:   #Removes if in dissallowed accounts
        return True
    for term in disallowed_terms:  #Removes if in dissallowed terms
        if term in tweet["text"]:
            return True
    return False
```

```python
\\ basic_sentiment.py
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from typing import *
import re
import matplotlib.pyplot as plt
import datetime
from matplotlib import style
import random
import numpy as np
import nltk
from nltk.corpus import stopwords
from nltk import RegexpTokenizer
from wordcloud import WordCloud
import tensorflow as tf
from NaiveTextBlob import *


def getSentiment(tweets : Dict):
    '''
     :param tweets: Dictionary of tweet text identified by their timestamp
     :return:       Dictionary of tweets with text and their sentiment polarity
     '''
    print("Getting sentiment...")
    analyzer = SentimentIntensityAnalyzer() #Creates a vaderSentiment object
    for t in tweets:
        averages = []
        sentences = nltk.sent_tokenize(extractText(tweets[t]))
        for i in sentences:
            vs = analyzer.polarity_scores(i) #Retrieves sentiment scores
            averages.append(vs["compound"]) #adds each sentence polarity to a list so an average can be taken.
        try:
            mean = sum(averages) / len(averages)
        except ZeroDivisionError:
            continue
        tweets[t] = (extractText(tweets[t]), mean) #Extracts compound score
    print(tweets)
    return tweets


def extractText(tweet: str)-> str: #Regular expression to remove hyperlinks and hashtags
    result = re.sub("http\S+", "", tweet)
    result = re.sub("#\S+", "", result)
    result = re.sub("@\S+", "", result)
    return result


def AverageByTime(tweets: Dict)-> Dict:
    print("Finding averages...")
    polarised_time = {}
    initial_time = datetime.datetime.strptime(list(tweets.keys())[0], "%a %b %d %H:%M:%S %z %Y") #Gets time of first tweet
    time_gap = int(input("Choose a time interval between values:\n>>>")) #Collects value from user for intervals
    next_time = initial_time + datetime.timedelta(minutes=time_gap)
    current_polarity = []
    for t in tweets:
        time = t
        while '&' in time:
            time = time.replace("&", "")
        time = datetime.datetime.strptime(time, "%a %b %d %H:%M:%S %z %Y")
        if time <= next_time:
            current_polarity.append(float(tweets[t][1]))
            continue
        else:
            total = sum(current_polarity)
            elements = len(current_polarity)
            result = total / elements
            formatted_time = time.strftime("%a %b %d %H:%M:%S %z %Y")
            polarised_time[formatted_time] = result
            initial_time = datetime.datetime.strptime(t, "%a %b %d %H:%M:%S %z %Y")
            next_time = initial_time + datetime.timedelta(minutes=time_gap)
            current_polarity = []
    return polarised_time


def findAverage(polarised_tweets: Dict) -> float:
    total : float = 0.0
    total_squared : float = 0.0
    for t in polarised_tweets:
        total += float(polarised_tweets[t][1])
        total_squared += float(polarised_tweets[t][1]**2)
    mean = total / len(polarised_tweets)
    standard_deviation = (total_squared / len(polarised_tweets)) - mean**2
    return mean
```

```python
def plotGraph(time_polarity: Dict, kick_off_hour : int, kick_off_minute : int, title : str) -> None:
    style.use('Solarize_Light2') #Sets style and labels
    plt.xlabel('Time, (\m)')
    plt.ylabel('Sentiment Polarity (-1 -> 1)')
    plt.title(title)
    x = []
    y = []
    for t in time_polarity:
        y.append(time_polarity[t])
        minute = datetime.datetime.strptime(t, "%a %b %d %H:%M:%S %z %Y") - datetime.timedelta(hours=kick_off_hour,
      minutes=kick_off_minute) #Sets time at minute 0
        minute = minute.strftime("%H:%M:%S")
        minute = datetime.datetime.strptime(minute, "%H:%M:%S") #Converts to needed datetime format
        x.append(minute)
    print("Graph Opening...")
    plt.plot(x, y, color=random.choice(['blue','olive','red','green','purple']), marker='.')


def plotBarGraph(averages: dict, title: str) -> None:
    style.use('Solarize_Light2') # Sets style and labels
    plt.xlabel('Team')
    plt.ylabel('Sentiment Polarity (-1 -> 1)')
    plt.title(title)
    plt.bar(averages.keys(), averages.values(), color=random.choice(['blue','olive','red','green','purple']))

def plotCorrelation(averages : List, results : List) -> None:
    style.use('Solarize_Light2') #Sets style and labels
    plt.ylabel('Goal Difference')
    plt.xlabel('Sentiment Polarity (-1 -> 1)')
    plt.scatter(averages, results, color=random.choice(['blue','olive','red','green','purple']))
    plt.plot(np.unique(averages), np.poly1d(np.polyfit(averages, results, 1))(np.unique(averages)))

def frequencyAnalyse(polarised_tweets : Dict):
    positive_words = {}
    negative_words = {}
    tokenizer = RegexpTokenizer(r'\w+')
    stop_words = list(stopwords.words('english'))
    for i in polarised_tweets:
        word_pit = tokenizer.tokenize(polarised_tweets[i][0])
        tags = nltk.pos_tag(word_pit)
        for word in tags:
            if word[0] in positive_words:
                positive_words[word[0]] += 1
                continue
            elif word[0] in negative_words:
                negative_words[word[0]] += 1
                continue
            if len(word[0]) < 3:
                continue
            if word[0].lower() in stop_words:
                continue
            if word[1] in ['JJ','NNP','JJS','JJR']:
                if polarised_tweets[i][1] > 0.2:  #Positive
                    positive_words[word[0].lower()] = 1
                elif polarised_tweets[i][1] < -0.2:  #Negative
                    negative_words[word[0].lower()] = 1
    for w in sorted(negative_words, key=negative_words.get, reverse=True):
        print(w, negative_words[w])
    return (positive_words, negative_words)


def wordCloudGenerator(word_group : List, mask):
    wc = WordCloud(background_color="white", max_words=2000, mask=mask)
    clean_string = ','.join(word_group)
    wc.generate(clean_string)
    plt.imshow(wc, interpolation='bilinear')
    plt.axis("off")
    plt.show()

def createModel(polarity : List, results: List):
    pol = np.array(polarity)
    gd = np.array(results)
    print(pol, gd)
    tf.logging.set_verbosity(tf.logging.ERROR)
    l0 = tf.keras.layers.Dense(units=1, input_shape=[1])
    model = tf.keras.Sequential([l0])
    model.compile(loss='mean_squared_error', optimizer=tf.keras.optimizers.Adam(0.1))
    model.fit(pol, gd, epochs=600, verbose=False)
    return model





def predictOutcome(model, match_polarity : float) -> float:
    result = model.predict([match_polarity])
    return result
```

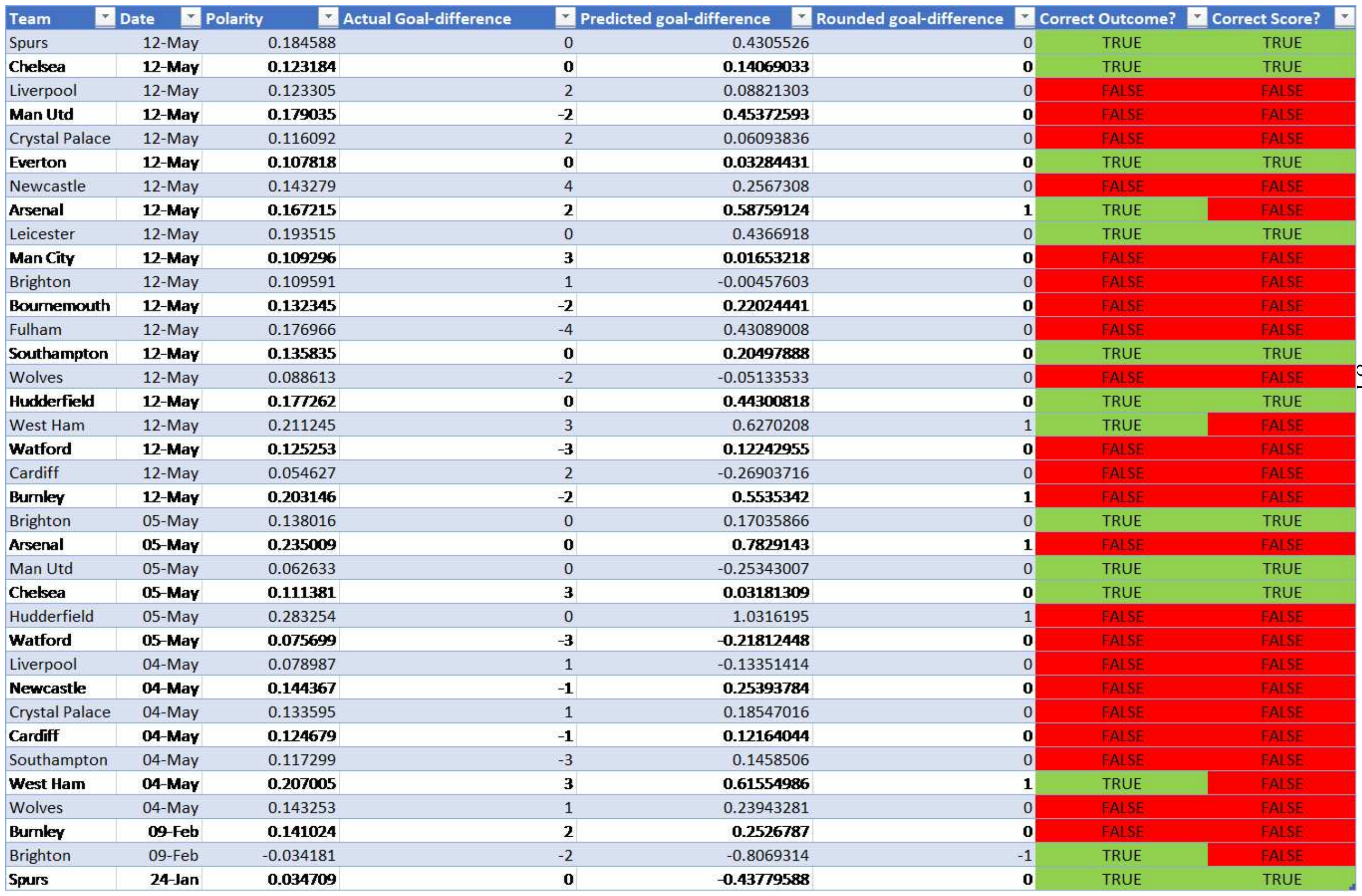
```
\\ control.py
from abstract_tweets import *
from basic_sentiment import *
from typing import *
from PIL import Image
import csv

class Analyse:
    def __init__(self):
        self._team_name : str
        self._file_name : str
        self._title : str
        self._kick_off_hour : int
        self._kick_off_minute: int
        self._result : int
        self._polarity : float
        self._image_file : str

    def setTeamName(self) -> None:
        self._team_name = input("TEAM NAME:\n>>>")

    def setFileName(self) -> None:
        self._file_name = input("Enter FILE NAME:\n>>>")

    def setGraphInfo(self) -> None:
        self._title = input("Enter the TITLE:\n>>>")
        self._kick_off_hour = int(input("Enter Kick-Off HOUR:\n>>>"))
        self._kick_off_minute = int(input("Enter Kick-Off MINUTE:\n>>>"))

    def setResult(self) -> None:
        self._result = int(input("Enter the goal difference (+x WIN, 0 DRAW ,-x LOSS):\n>>>"))

    def gatherTweets(self) -> Dict:
        tweets : List = gatherTweets(self._file_name)
        abstracted : Dict = abstractTweets(tweets)
        print(str(len(abstracted)) + " tweets accepted.")
        return abstracted

    def findSentiment(self, abstracted) -> Dict:
        polarised_tweets = getSentiment(abstracted)
        return polarised_tweets

    def calculateAverage(self, polarised_tweets : Dict) -> Tuple:
        return (self._team_name, findAverage(polarised_tweets))

    def createLineGraph(self, polarised_tweets : Dict) -> None:
        polarised_time = AverageByTime(polarised_tweets)
        plotGraph(polarised_time, self._kick_off_hour, self._kick_off_minute, self._title)
        plt.legend([self._team_name])

    def createBarChart(self, averages, title):
        plotBarGraph(averages, title)

    def setPolarity(self, polarised_tweets : Dict) -> None:
        self._polarity = findAverage(polarised_tweets)

    def getFrequency(self, polarised_tweets : Dict) -> Tuple:
        return frequencyAnalyse(polarised_tweets)

    def setImageFile(self):
        self._image_file = input("Enter name of the image:\n>>>")

def average():
    num_of_teams = int(input("Enter NUMBER of teams:\n>>>"))
    teams = [Analyse() for n in range(num_of_teams)]
    for t in teams:
        result = {}
        t.setTeamName()
        t.setFileName()
        while True:
            try:
                abstracted = t.gatherTweets()
                break
            except:
                print("file does not exist!")
                t.setFileName()
        polarised = t.findSentiment(abstracted)
        average = t.calculateAverage(polarised)
        result[average[0]] = average[1]
        t.createBarChart(result, title='PL')
    plt.show()
```

```python
def graph():
    num_of_teams = int(input("Enter NUMBER of teams:\n>>>"))
    teams = [Analyse() for n in range(num_of_teams)]
    for t in teams:
        t.setTeamName()
        t.setFileName()
        t.setGraphInfo()
        while True:
            try:
                abstracted = t.gatherTweets()
                break
            except FileNotFoundError:
                print("file does not exist!")
                t.setFileName()
        polarised = t.findSentiment(abstracted)
        t.createLineGraph(polarised)
    plt.title("PL")
    plt.show()

def correlation():
    num_of_teams = int(input("Enter NUMBER of teams:\n>>>"))
    teams = [Analyse() for n in range(num_of_teams)]
    results = []
    averages = []
    for t in teams:
        t.setFileName()
        t.setResult()
        while True:
            try:
                abstracted = t.gatherTweets()
                break
            except FileNotFoundError:
                print("file does not exist!")
                t.setFileName()
        polarised = t.findSentiment(abstracted)
        t.setPolarity(polarised)
        results.append(t._result)
        averages.append(t._polarity)
    plotCorrelation(averages, results)
    #write to .csv
    file = open('C:/Users/Liam/desktop/correlation.csv','a')
    for i in range(len(averages)):
        text = str(averages[i]) + ',' + str(results[i]) + '\n'
        file.write(text)
    file.close()
    plt.show()

def frequencyAnalysis():
    team = Analyse()
    try:
        team.setFileName()
    except FileNotFoundError:
        print("file does not exist!")
        team.setFileName()
    abstracted = team.gatherTweets()
    polarised = team.findSentiment(abstracted)
    frequencies = team.getFrequency(polarised)
    positive_frequency = frequencies[0]
    negative_frequency = frequencies[1]
    print(positive_frequency)
    print(negative_frequency)
    print(len(positive_frequency))
    print(len(negative_frequency))
    return (positive_frequency, negative_frequency)

def wordCloud():
    team = Analyse()
    all_words = frequencyAnalysis()
    choice = input("Positive [p] or Negative [n] wordcloud?\n>>>")
    selected = False
    while not selected:
        if choice == 'p':
            words = all_words[0]
            selected = True
        elif choice == 'n':
            words = all_words[1]
            selected = True
    team.setImageFile()
    while True:
        try:
            mask = np.array(Image.open("C:/Users/Liam/Desktop/" + team._image_file))
            print("accepted!")
            break
        except FileNotFoundError:
            print("File not found!")
            team.setImageFile()
    wordCloudGenerator(words, mask)
```

```python

def regression():
    polarity = []
    results = []
    with open('C:/Users/Liam/desktop/correlation.csv','r') as file:
        reader = csv.reader(file)
        for row in reader:
            polarity.append(row[0])
            results.append(row[1])
    model = createModel(polarity, results)
    value = float(input("Enter the pre-match polarity:\n>>>"))
    prediction = predictOutcome(model, value)
    print(prediction)

def classification():
    pass

if '__name__' == '__main__':
    choice = input("AVERAGE            : '1'\n"
                   "GRAPH              : '2'\n"
                   "CORRELATE          : '3'\n"
                   "FREQUENCY ANALYSIS : '4'\n"
                   "WORDCLOUD          : '5'\n"
                   "REGRESSION         : '6'\n"
                   ">>>")
    if choice == '1':
        average()
    elif choice =='2':
        graph()
    elif choice == '3':
        correlation()
    elif choice == '4':
        frequencyAnalysis()
    elif choice == '5':
        wordCloud()
    elif choice == '6':
        regression()
    elif choice == '7':
        trainClassifier()
```

Appendix C
Results Tables

Table which documents the testing process.



Team	Date	Polarity	Actual Goal-difference	Predicted goal-difference	Rounded goal-difference	Correct Outcome?	Correct Score?
Spurs	12-May	0.184588	0	0.4305526	0	TRUE	TRUE
Chelsea	**12-May**	**0.123184**	**0**	**0.14069033**	**0**	TRUE	TRUE
Liverpool	12-May	0.123305	2	0.08821303	0	FALSE	FALSE
Man Utd	**12-May**	**0.179035**	**-2**	**0.45372593**	**0**	FALSE	FALSE
Crystal Palace	12-May	0.116092	2	0.06093836	0	FALSE	FALSE
Everton	**12-May**	**0.107818**	**0**	**0.03284431**	**0**	TRUE	TRUE
Newcastle	12-May	0.143279	4	0.2567308	0	FALSE	FALSE
Arsenal	**12-May**	**0.167215**	**2**	**0.58759124**	**1**	TRUE	FALSE
Leicester	12-May	0.193515	0	0.4366918	0	TRUE	TRUE
Man City	**12-May**	**0.109296**	**3**	**0.01653218**	**0**	FALSE	FALSE
Brighton	12-May	0.109591	1	-0.00457603	0	FALSE	FALSE
Bournemouth	**12-May**	**0.132345**	**-2**	**0.22024441**	**0**	FALSE	FALSE
Fulham	12-May	0.176966	-4	0.43089008	0	FALSE	FALSE
Southampton	**12-May**	**0.135835**	**0**	**0.20497888**	**0**	TRUE	TRUE
Wolves	12-May	0.088613	-2	-0.05133533	0	FALSE	FALSE
Hudderfield	**12-May**	**0.177262**	**0**	**0.44300818**	**0**	TRUE	TRUE
West Ham	12-May	0.211245	3	0.6270208	1	TRUE	FALSE
Watford	**12-May**	**0.125253**	**-3**	**0.12242955**	**0**	FALSE	FALSE
Cardiff	12-May	0.054627	2	-0.26903716	0	FALSE	FALSE
Burnley	**12-May**	**0.203146**	**-2**	**0.5535342**	**1**	FALSE	FALSE
Brighton	05-May	0.138016	0	0.17035866	0	TRUE	TRUE
Arsenal	**05-May**	**0.235009**	**0**	**0.7829143**	**1**	FALSE	FALSE
Man Utd	05-May	0.062633	0	-0.25343007	0	TRUE	TRUE
Chelsea	**05-May**	**0.111381**	**3**	**0.03181309**	**0**	TRUE	TRUE
Hudderfield	05-May	0.283254	0	1.0316195	1	FALSE	FALSE
Watford	**05-May**	**0.075699**	**-3**	**-0.21812448**	**0**	FALSE	FALSE
Liverpool	04-May	0.078987	1	-0.13351414	0	FALSE	FALSE
Newcastle	**04-May**	**0.144367**	**-1**	**0.25393784**	**0**	FALSE	FALSE
Crystal Palace	04-May	0.133595	1	0.18547016	0	FALSE	FALSE
Cardiff	**04-May**	**0.124679**	**-1**	**0.12164044**	**0**	FALSE	FALSE
Southampton	04-May	0.117299	-3	0.1458506	0	FALSE	FALSE
West Ham	**04-May**	**0.207005**	**3**	**0.61554986**	**1**	TRUE	FALSE
Wolves	04-May	0.143253	1	0.23943281	0	FALSE	FALSE
Burnley	**09-Feb**	**0.141024**	**2**	**0.2526787**	**0**	FALSE	FALSE
Brighton	09-Feb	-0.034181	-2	-0.8069314	-1	TRUE	FALSE
Spurs	**24-Jan**	**0.034709**	**0**	**-0.43779588**	**0**	TRUE	TRUE