L-Università
ta' Malta

FACULTY/INSTITUTE/CENTRE/SCHOOL FACULTY OF ICT, DEPT OF AI

DECLARATIONS BY POSTGRADUATE STUDENTS

Student's I.D. /Code 20891(G)

Student's Name & Surname MATTHEW JOSEPH ZAMMIT

Course MASTER OF SCIENCE . ARTIFICIAL INTELLIGENCE

Title of Dissertation
PREDICTIVE ANALYSIS OF FOOTBALL
MATCHES USING IN-PLAY DATA

(a) Authenticity of Dissertation

I hereby declare that I am the legitimate author of this Dissertation and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education.

I hold the University of Malta harmless against any third party claims with regard to copyright violation, breach of confidentiality, defamation and any other third party right infringement.

(b) Research Code of Practice and Ethics Review Procedures

I declare that I have abided by the University's Research Ethics Review Procedures.

As a Master's student, as per Regulation 58 of the General Regulations for University Postgraduate Awards, I accept that should my dissertation be awarded a Grade A, it will be made publicly available on the University of Malta Institutional Repository.

Signature of Student

MATTHEW JOSEPH ZAMMIT
Name of Student (in Caps)

22/10/2018
Date

Predictive Analysis of Football Matches using In-Play Data

Matthew Joseph Zammit

Supervisor(s): Dr. Jean Paul Ebejer and Dr. George Azzopardi

Faculty of ICT

University of Malta

April 2018

Submitted in partial fulfillment of the requirements for the degree of M.Sc. Artificial Intelligence

Abstract:

Sports betting has emerged as a booming industry driven by the popularity of betting on different scenarios within sporting events. Football is one of the most popular sports that is followed by millions of fans around the world. Its dynamic nature, low-scoring matches and other complex variables that could influence the outcome of a game make it hard to predict the outcome of a match. In recent years, more in-game and detailed statistics have been collected and analysed by professionals of the game. The aim of this study is to investigate the application of machine learning techniques for predicting the full-time result (Home Win/Draw/Away Win) of football matches at the half-time interval by the use of in-play data. We collect and analyse a rich data set of temporal data from seven seasons of five major European leagues between 2009 and 2016. We focus our research on the application of random forest as the main machine learning technique for this problem. We build a genetic algorithm to perform feature selection and hyper-parameter tuning to investigate if the initial results could be further improved. Finally, we contextualise the data set with pre-match data and analyse how this changes the results and the predictors selected. We find that after feature selection and model tuning, the random forest has a mean accuracy 45.0% (±1.6) on unseen data across the different leagues. With the addition of pre-match data the mean accuracy increased to 46.0% (±2.1), but the results for each league remained similar. We evaluate different models on an unseen data set from the year 2016/17. The tuned random forest using both pre-match and in-game data achieves a mean accuracy of 44.8% across the leagues. The highest accuracy was that of 50.0% on the test sample of the English Premier League. The lowest was that of 40.0% on the French and Spanish leagues. We also converted the random forest classification to a probabilistic prediction based on the output of the underlying decision trees. We compare these probabilities to implied odds from a betting exchange (Betfair) on small sample of matches from the unseen data of the English and Italian leagues. We used the Brier Score function to calculate the accuracy of the predictions. Results show that the accuracy is similar for the English Premier League and Italian Serie A for both the Random Forest and Betfair. This comparable performance may indicate that the Machine Learning predictions are similar to those of the betting exchange markets.

Acknowledgements

I would like to thank my supervisors, Dr. Jean Paul Ebejer and Dr. George Azzopardi for guiding me and helping me form my thesis. They have supplied me with great knowledge and ideas on how I could tackle problems and improve on my results. They helped me formulate better my thoughts and always pushed me harder to do more; be it to collect more detailed data sets or to try more experiments. Their interest in my work and findings kept me motivated throughout the course of the project. Their attention to detail has driven me to produce the best work I could have done. I would like to thank Dr. Jean Paul Ebejer especially for the support and selfless help he gave to me when I needed it the most. I am also grateful to my family, close friends and colleagues for the support they have given me during this tough time of long nights and days of continuous hard work and isolation.

Contents

List of Figures

List of Tables

Predictive Analysis of Football Matches using In-Play Data

Matthew Joseph Zammit[*]

Supervised by: Dr. Jean Paul Ebejer and Dr. George Azzopardi

April 2018

[*] Submitted in partial fulfillment of the requirements for the degree of M.Sc. Artificial Intelligence.

1 Introduction

The ability to accurately predict the outcome of an event is considered to be essential in many real world applications. Examples of these predictions are commonly found in finance, where investors forecast or predict the growth of a public company in terms of its share price. Further examples could be seen in other domains such as in politics, where people try to predict the winner of an election campaign. Predictions are also cast for sporting events. For example, predicting the score of a football match or the winner of a tennis tournament. Insight into future events could help those making predictions gain from these scenarios if their predictions are proven to be correct. Punters who backed Leicester city to win the English Premier League in 2016 or Donald Trump to win the 45^{th} presidential election of the United States would have had great returns on their stakes. Investors buying in on public companies such as Google or Amazon when their share prices were low, benefited from the growth these companies have had throughout the years. Predictions about the possible outcomes of such events are cast by different entities. These can be experts in their respective fields, bookmakers and exchange markets [SS09].

Experts rely heavily on their expertise in the domain to come to a conclusion on the outcome of a particular event. In sports, these are usually former professionals of the game. These experts usually have large audiences and their opinions are heard and read by many people. Depending on the event of interest at the time, the experts are asked for their opinions on the possible outcomes of such events. For example, as the date of a football event becomes nearer, the experts will be asked for their predictions on final result of the game. Other predictions might be made with respect to which teams are going to be relegated to lower divisions or which team will win a league or a tournament. Sports experts, usually take into consideration the qualities and attributes of the players participating in the event, the form of the teams and of their key players. By using this information, combined with their experience and knowledge in the domain, they come up with a particular outcome for the event depending on how they interpret their available data. Tipsters and experts, put their reputation at stake with their predictions as they are usually made public and seen by the followers of the sport.

The same questions asked to the experts would also be found as markets of bookmakers with each outcome having its payout price. The price depends on the perceived probability of that outcome happening. The higher the price is, the less likely the outcome is to occur. In markets set up by bookmakers, the odds are composed by their odds compilers, experts and models. Apart from the actual outcome probabilities, the odds are influenced by factors such as the bookmakers' profit margin, the risk they are willing to take on certain events and the prices offered by their competitors. Bookmakers monitor closely the odds of their competitors, either to stay competitive in the market or not to be exposed by sudden drop (shortening) in prices. In prediction markets, agents with different beliefs about the outcome of the events form as a group of users interacting with an exchange system by matching their back bets with counter lay bets of each other. The price of the outcomes represents the aggregated knowledge and belief of all the contributors placing bets in the market on the likelihood of that outcome occurring [SS09, AGF13]. Betting exchanges are more similar to a prediction market than bookmakers. The participants in a betting exchange can play the role of both the backer (bet in favour of an outcome occurring) and the layer (bet against an outcome occurring). When betting against a bookmaker, one can only back outcomes. For example, a punter can only back Juventus to win the Serie A at the price given by the bookmaker. In a betting exchange, a punter can offer his odds against Juventus winning the league for others to back them at that price. This is known as laying the market. The prices of betting exchange markets can be interpreted as predictions for the outcomes of future events founded by the Hayek Hypothesis, that the best way for "aggregating asymmetrically dispersed information possessed by market participants" is through the price mechanism. This kind of betting is similar to that found in stock exchange and foreign exchange markets in finance [AGF13].

In recent years, prediction markets have emerged as one of the most popular markets where most money is exchanged on a daily basis. One factor leading to this boom is due to the high availability of electronic markets by the means of the Internet. This has allowed more users to participate with their contributions, and in turn this has led to more liquidity in the markets [AGF13]. Furthermore, most electronic prediction exchanges expose functionality on their system by the use of public APIs to allow for programmatic interfacing with their systems. Because of this, prediction markets are being studied more due to the fact that they have been shown to be as accurate as bookmakers and much better than tipsters [SS09]. This emergence of betting markets and the popularity of football has lead researchers to present their own models of the game with predictions on tournament winners, final league positions and individual match outcomes [CFN12]. In this thesis, we will be investigating the use of models based on machine learning techniques to cast predictions on football match events. We will evaluate their overall performance on the ability to correctly classify final match results by learning patterns from historical data and compare their probabilistic accuracy against that of a prediction market.

Most of the research that has been carried out on the game of football is mainly concerned with identifying and measuring key match performance indicators that are best adaptive at discriminating between winning and losing teams. Many of these studies considered game related statistics describing the main motives of a game of football. These being goal scoring, offensive, defensive and other contextual attributes [LBLP10, CCL12, LPGRY17]. Other recent research studied specific variables to investigate common held beliefs in the football community with respect to these variables' effect on the outcome of the game. One of the most studied indicator is the team possession of the ball [LPGRY17] and how this co-varies as the team is winning, drawing and losing in duration of a game. Other studies have been conducted on other variables, such as, the effect of playing at home [CT05], the effect of red cards [RCH94], the number of entrances a team has in the opposition's penalty area [RRFGZ13].

There have been some studies that investigated Machine Learning for the prediction of football matches. The research, has been mainly focused on predicting events on pre-match data. Many of the studies developed Bayesian Networks built with the help of domain experts to predict the full-time result of football matches [RMYA17, BB10, CFN12, JFN06]. Others have investigated the use of Artificial Neural Networks to predict the final score of Iranian League matches [ATASNG14] and the outcome of Australian Football League match results [OF97]. There have also been some studies on comparing different models such as Neural Networks, Bayesian Networks, Decision Trees and Random Forest trained on the same data to predict the match results of specific English Premier League teams [JFN06, HR11]. Other studies investigated the use fuzzy [RPR05] and rule-based logic [MKC+08] to predict the outcomes of tournament football matches. Although scarce, some research has been also been focused on in-play prediction of the over/under football prediction markets using Case Based Reasoning with logical rules [AGF13].

1.1 Motivation

Football has always been a very popular sport that is followed by millions of fans around the world [RPR05, HR11], especially in Europe and South America. Football is different in many ways to other popular sports. For example, football has a limited number of set plays, compared to other sports such as American football, baseball, cricket and others. This means that gameplay is not stopped and started too often and when it is, the game resumes quickly. Therefore, this dynamic and interactive nature and a low-scoring goal count per match (around 3 goals), make it harder for analysts to collect meaningful data and know which of the many and complex variables are more valuable to measure [LBLP10, HR11, JFN06]. There are many variables that may have an influence on how a team performs on a given day. A team's current form, tactics, available key players and their individual form, psychological impact of playing home or away and player fatigue are some of many factors that could impact the performance of a team in a game [HR11]. Capturing correct and accurate performance indicators is critical for analysts, coaches and players to make an objective performance and match analysis [CCL12, VNH14, JJM04]. These performance indicators are also suitable for match and outcome predictions [LBLP10, JJM04]. With time, more detailed and sophisticated in-play and players match statistics are being captured and made available for consumption [VvK16]. For example, outfits like Opta[1] are gathering detailed, temporal and positional in-play statistics of players for popular football competitions.

The popularity of betting on sporting events is generating more transactions than in any other market, making it one of the markets where the most money is exchanged daily by its large amount of users [AGF13, RPR05]. And as such, in recent years, using different techniques to predict the outcome of football match results have become more popular [RMYA17]. The results found in [AGF13] suggest that the market takes time to adjust the price since the participants are mainly human and humans are not that good at estimating the correct probability for the outcome of an event. Furthermore, the authors argue that humans can have their decisions affected by emotional factors and thus can perform poorly because of being afraid or indecisive.

Despite this emergence and popularity of sports betting markets, literature and research about this domain has not kept up with respect to in-play and odds movement prediction (evolutionary betting) as the event is progressing [AGF13, EU10]. Most of the research has been conducted on pre-match basis, meaning that many events such as scores, injuries, yellow/red cards that happen during the event itself are not considered. These studies make use of information available only before the game starts. The results of such studies are focused on the accuracy of the predictions of the match result and by the use of different betting strategies evaluate whether accumulative payouts would have registered a profit at the end of the betting season [CFN12]. When predicting the odds movement, the models' accuracy is not evaluated against the actual outcomes of the events but the number of times it was able to correctly predict the direction and magnitude of the price for the duration of an event. Both [AGF13, EU10] argue that in-play and evolutionary betting

[1] Opta - https://www.optasports.com/

are becoming more important research areas in this field. Because of this, further research should be carried out in this regard. The authors of [EU10] also argue that, some of the research on in-play sports betting is incomplete as the information from the betting market is not compared to a model of match outcomes for that particular sport.

Research that has been carried out on the application of machine learning and time series techniques for the prediction of price movements in financial markets have not yet found their way into sports betting markets [AGF13, SS09]. Literature seems to show a consensus on the fact that the sports betting market have certain characteristics that are desirable and that are difficult to evaluate or test in the financial markets. In these markets, researchers are not able to observe the arrival of new information to the participants of foreign and stock exchanges. For example, when news breaks out of a company involved in a scandal, this is usually reflected in its share price. However, it is difficult for researchers to know in what form and how long this information takes to arrive to those invested in the situation. This makes it harder for the researchers to measure how fast the new information is reflected into the prices of the market. However, in sports markets this information is present for all the participants in the market in a simultaneous fashion and can also be observed by the researcher in the form of goals, cards, points, or whatever the information might be [KM03].

These findings make it a well-suitable problem for applying Machine Learning techniques in the domain of in-game football match predictions and investigate the performance of such techniques in this field.

1.2 Aims and Objectives

Our work in this study builds on the importance of the studied match performance statistics at the half-time interval of football events and use them as training data for Machine Learning algorithms. The aim of this study is to investigate the applicability of Machine Learning techniques to predict the full-time result (home win, draw, away win) of major European league games that are in a drawn state at half-time. From this the following research questions are;

- How accurate are Machine Learning techniques at classifying correctly the full-time result of a game by using team performance statistics till the half-time interval?
  - Given that there was no publicly available data set that includes the required attributes by our study, the first objective is the building of a data set consisting

  of in-play data of the major European football league matches that are required for our study.

  - Once the data set is constructed, the next objective is the training of a number of selected classifiers on a sample of the data set such that the performances of each could be compared.

- Is it possible to achieve better performance when applying feature selection and model-tuning techniques to the model?

  - Build a genetic algorithm for the application of feature selection

  - Apply the genetic algorithm to model training process with randomised search for hyper-parameter tuning and evaluate the classifier's performance on left-out data set.

- Does the overall performance of the classifiers improve with the addition of pre-match data to the in-game statistics at half-time?

  - Repeat the training process of the classifier using the combined data set of pre-match and in-game statistics and evaluate its accuracy on external data sets.

- How does the probabilistic predictions of the classifier compare to the implied probabilities of a betting exchange market?

  - Use the Brier score function to evaluate the probabilistic performance of the market and the model against the actual outcomes of a sample from the test set.

To investigate our research questions we collected and built a data set from five major European leagues for the seasons ranging 2009-2016. We applied machine learning techniques on a sample of the data set and found that the random forest classifier was performing consistently better than the rest of the classifiers. We then applied, nested cross validation with a genetic algorithm for feature selection and random-search for parameter tuning. We repeated the tests with a newly formed data set containing both in-play and pre-match data to investigate whether this would improve the scores of the classifier.

1.3 Document Structure

The rest of the thesis is laid out as follows, in Chapter 2, we give a detailed literature review in the field of match statistic analysis and the use of Machine Learning techniques in the field of football and sports predictions. In this Chapter, we also give a brief background on the techniques used in this thesis and other terminology that needs defining. In Chapter 3, we give a detailed account of how the data was collected and transformed into the data sets required by our study. We detail all the steps taken in our experimentation of the subject. We present the results of our findings in Chapter 4, where we discuss and give in detail the results achieved from the multiple experiments that were carried out. In this chapter we also evaluate our best models from experimentation and test their performances on an unseen data set from the season 2016/2017. In Chapter 5, we present our thoughts on future works in the area and we outline our conclusions of this study.

2 Background and Literature Review

In this chapter, we describe in detail the background knowledge that is essential for understanding the body of work in this research. Furthermore, we describe in depth the key literature carried out in the subject of football performance indicators and match result predictions. In the background, we describe how a standard game of football for professional teams is played and how this may vary in different types of competitions. In Section 2.2, we show some examples of football betting markets and define important terms related to betting exchanges. In this section, we describe how odds and prices can be interpreted as implied probabilities of the match outcomes and present the profit and loss equations for backing and laying a selection. In the literature review section we discuss the main areas of research carried out on the sport of football. Most of the research done in this domain is with regards to finding and evaluating performance indicators that discriminate between successful and unsuccessful teams. From this literature we identified important team statistics to include as part of our feature vector in our research. Other research focused on using machine learning techniques to predict the full-time result of football matches. Most studied techniques were Bayesian networks built with the help of domain experts. Others considered techniques such as random forest, artificial neural networks, decision trees, naive Bayes and fuzzy and rule based logic. More recent studies have investigated how models can be used to predict the odds movement of events as they are being played. In these studies, the authors compare the odds predicted by the model with those from betting markets. We then outline the different machine learning techniques used for this study and how these are used together to build and evaluate the models. We describe the standardisation of the data set values as part of the data pre-processing procedure used for certain machine learning algorithms. We then detail the internal mechanisms of how the machine learning algorithms learn to classify samples into distinct classes. For each classifier we mention their most notable strong and weak points. We discuss resampling techniques used for model selection such as k-fold cross-validation and nested cross-validation. In Subsection 2.5.5 we describe the use of genetic algorithms and how they are used for the scope of feature selection in this study. We end the Literature and Background section by describing the evaluation metrics used for classification problems such as the one tackled in our research.

2.1 Association Football

Football is one of the most followed sport around the globe, especially in Europe and South America. A professional football match is played between two teams of 11 players in each, made up of one goal-keeper and ten outfield players. The match is decided over two halves of 45 minutes each, making a total of 90 minutes per game. Injury time is added at the end of each half to accommodate for lost time during that period for injuries, goal celebrations and substitutions. The team that scores most goals wins the match. If both teams score the same number of goals, the game can either end as a draw or else the match goes into extra-time. Extra-time is an additional period of 30-minutes play time, divided into two halves of 15 minutes each. If the score remains even after the extra-time period is played out, the game goes into a penalty shoot-out. Whether drawn games go into extra-time or end with an even score depends on what type of competition the match is of. For knock-out tournaments, drawn games have to be settled out if the score remains tied at end of the game (or two games in some competitions). This is so that the winner can progress to the next phase of the competition, whilst the loser is knocked out. Events in league tournaments can end in a draw. The most common format of league competitions consists of a defined number of teams, in which each team plays against every other team twice, once at home and once away. The total number of games played is defined by $(n)(n-1)$ where n is the number of teams in the league. For example, a league of 20 teams would have a total of $(20)(19) = 380$ games. Each team would play, $(n-1)$ games at home and $(n-1)$ games away, at the oppositions' stadiums. A particular team would thus play $2(n-1)$ games in total over a season. In our example, a team would play 19 home games and 19 away games, giving a total of 38 games per team. The winner of a match is awarded three points and the loser does not receive any. If the game ends in a draw, both teams receive one point each. The team which accumulates the most points over the season is declared the winner for that year. Bottom teams may be relegated to lower leagues. In this study we consider five major European leagues for the years from 2011 to 2016. The leagues are the English Premier League, Italian Serie A, Spanish La Liga, German Bundesliga and the French Ligue 1.

2.2 Sports Betting

Sports betting markets consist of speculative situations that can happen during a sporting event. For example, a goal is scored or a card being given to a player. The following is a list of some examples of betting markets and their outcomes for a football event.

- **Full-time Result** - What is the final result of the match going to be? *Home win, draw or away win*?
- **Correct Score** - What is the final score of the match going to be? *0-0, 0-1, 1-0, 1-1, 2-0, 2-1, 0-2, 1-2, 2-2,...*?
- **Over/Under 2.5 Goals** - Will there be over or under 2.5 goals in the match? *Over, Under*?
- **Total number of corners** - How many corners will there be in the match? *0, 1, 2, 3, 4,..., 10+*?

2.2.1 Odds and Implied Probability

As already stated in the previous chapter bookmakers show the odds against an outcome occurring. The equation for odds against is shown in Equation 1, where h represents the number of times an outcome occurs and $\bar{h}$ denotes the number of times the outcome does not occur. For example, the odds against getting a particular number from a dice such as getting number six, has odds against of five to one. Five of the times the number will not be six and one time it will. In terms of betting, this means that for a stake of one unit the payout will be five times that stake. In our example, if the bettor stakes ten units the payout will be 50 if the top facing number of the dice is six. This means that $\frac{\bar{h}}{h}$ represents the profit the bettor will make and adding one to this number results in the return on the bet.

$$OddsAgainst = \frac{\bar{h}}{h} \tag{1}$$

The probability for the same outcome can be computed as shown in Equation 2. The price of the outcome is the inverse of the probability defined in Equation 3. The price from this equation will result in the return on the bet and this will be equal to the odds against plus one as described in the example above. This means that for the dice example the return will be six, as shown by using both the odds against and inverse of the probability, methods, $\left(\frac{1}{(\frac{1}{6})}\right) = 1 + (\frac{5}{1})$, are equal to each other.

$$Probability(h) = \frac{h}{h + \bar{h}} \tag{2}$$

$$Price(h) = \frac{1}{Probability(h)} \tag{3}$$

2.2.2 Backing and Laying on outcomes

Participants of the market have different beliefs on the possibility of outcomes of these scenarios to happen. The participants that believe that the outcome is likely to happen can *back* the outcome. Meaning that they bet for that particular outcome to happen. Such as for example a particular team to win. The other participants who believe that the outcome is unlikely to happen can *lay* it, meaning to bet for the outcome not to happen. The betting exchange is in charge of matching the bets to their counter bets of the participants where they agree on the probability of that outcome, i.e the price [AGF13].

After two bets (one for, one against) are matched by the price, the participants must also declare the amount they would like to stake for the trade to happen. The bet might not be fully matched by the two corresponding bets such that one of the bets is partially matched and requires other opposite bets to become fully traded. Otherwise, if there are no more bets to be matched, the price and remaining balance to be matched remains on the exchange until the trader cancels the bet or the market closes. Whether or not the trader will make a profit or loss depends on the outcome as follows;

The backer will make a profit if the outcome bet on is released as True. The profit gained by the backer is the loss made by the layer and is equal the price of the outcome multiplied by the number of units staked by the backer. If the outcome does not occur the backer loses the number of units he staked to the layer [AGF13].

$$PL_b(s,p) = \begin{cases} s * (p-1) & \text{if outcome is realised} \\ -s & \text{if outcome is not realised} \end{cases} \tag{4}$$

$$PL_l(s,p) = \begin{cases} -(s * (p-1)) & \text{if outcome is realised} \\ s & \text{if outcome is not realised} \end{cases} \tag{5}$$

The Profit and Loss functions are described in Equation 4 for the backer and Equation 5 for the layer. Both Equations have two terms, the stake ($s \in \mathbb{R} > 0$), representing the amount of money put on the bet. And the price/odd ($p \in \mathbb{R} > 1$) of the outcome.

2.3 Literature Review

In this section we discuss in detail the literature conducted on football with regards to match result prediction using machine learning techniques and the identification of important match statistics. Most of the research carried out on the sport of football is with respect to finding and measuring the effect of key match performance indicators that discriminate between winning and losing teams [LPGRY17, LBLP10, CCL12, LPD10, JJM04, VNH14, CT05, RRFGZ13, RCH94, TRB10, Peñ14]. In these studies the authors do not use their findings to predict the outcome of the games. There are some contradictions in the results found but mostly agree that score and offensive related attributes discriminate best between the different classes. The majority of this research investigates how ball possession varies between teams and how this changes when the teams are winning, drawing or losing [JJM04, VNH14, LPD10, LPGRY17]. Some interpret their results as teams playing direct football have a better chance at winning games than those adopting a possession based strategy. Other researchers focused on the use of machine learning techniques to predict the full-time result of football matches [JFN06, CFN12, RMYA17, BB10, HR11, OF97, ATASNG14, RPR05]. The most popular technique used is that of Bayesian networks which are built with the help of domain experts. The results show that these have the best performance when compared with other techniques such as neural networks, decision trees and clustering techniques. More recent studies have investigated the use of models to predict odds for the length of the game [AGF13, Øvr08, EU10]. In these studies, the authors compare the predictions to those from the betting markets and show that after an important action happens during a game, the prices take long to settle. They indicate that this may be due to humans not being able to judge the impact of such action on the outcome of the game.

2.3.1 Identifying discriminatory football match indicators

Research carried out on the game of football is mostly with regards to the study of analysing and identifying discriminatory match performance indicators. These studies have been done with the intention of giving further insights into which of the match statistics are most effective at distinguishing between successful and unsuccessful teams. The measurements of these key performance indicators would be particularly useful for sport scientists, coaches, the media and fans of the sport [LPGRY17]. As a result, these studies are designed in such a way that their findings could be interpreted by football coaches and players to be able to prepare better against opposing teams [LPGRY17]. Other interpretations could lead to the design and implementation of practices and training sessions to achieve peak performance in the statistics found to be most discriminant [LPD10]. For the majority part of this body of research, the main aim of the studies is to measure the impact that the variables taken into consideration have on the outcome of the game and how the significance of the values of these differ between winning, drawing and losing teams [LPGRY17, LBLP10, CCL12, LPD10, JJM04, VNH14, CT05, RRFGZ13, RCH94, TRB10, Peñ14].

A number of these studies considered match statistics that try to encapsulate the different motives that occur in a game of football with respect to defensive, offensive and goal scoring abilities of the competing teams [LBLP10, CCL12, LPGRY17]. For example, in [LBLP10] the following statistics were considered for 380 matches in the Spanish La Liga for the season 2008/09. The goal scoring group consisted of the total shots, shots on goal and effectiveness variables calculated for each team per match. Effectiveness was calculated as a percentage of the shots on goal over the total shots. The group related to the offensive attributes of a team were, passes, successful passes, crosses, off-sides committed, fouls received and ball possession. The third group on the other hand related to defensive attributes. This considered variables with respect to crosses against, off-sides received, fouls committed, yellow cards and red cards received by a team. The last group contained the venue and the quality of the opposition. The quality of the opposition was calculated to be the difference in the initial ranking of the two opposing teams. In [CCL12] a very similar grouping was considered but the sample was retrieved from normal time matches of three different world cups of the years, 2002, 2006 and 2010 (n=177). In both studies, one way ANOVA was used to find the most discriminant variables, and both share similar results where it was found that top teams had significantly more shots and shots on target than middle and low tier teams. In [LBLP10], it was found that effectiveness was also significantly higher for the top teams than the rest, however, this attribute was not considered in [CCL12]. Losing teams in the world cup had significantly more fouls committed and red cards than the winning teams, whilst in the Spanish league, these (or any other defensive attribute) were not found to be significantly different between the tiers. Similar results for defensive attributes found in [CCL12] were also evident in [RCH94]. In their research, they studied the effect red cards have on the outcome of the game. They investigate the red card effect with respect to expected goals in the game and the extent of the advantage given to the opposing team. They conduct their research on 140 matches in which a red card was given from the Dutch professional football league between the years 1989-1992. They find that a red card raises the goal expectancy in a game. According to their results, a red card given to a team reduces its probability of winning the match and increases that of the opposition. They also find that the probability of the game ending in a draw is also relatively small given a red card is produced. In their study on the home-side effect in the English Premier League, [CT05] find that yellow and red cards effect the away team less than they do to the home since they found that away teams are more likely to be deployed in a defensive tactic. For the 1997/98 season they show that home teams won 48% of the total matches (380) and 57% of the total points. From their results, it is shown that home teams are more accurate than away teams in terms of shots on target over all the shots taken. They also conclude that, goals scored from the home team are more likely to come from sustained pressure on the away defence rather than counter-attacks. [RRFGZ13] continue to sustain the argument of the red card effect by finding that teams with a numerical disadvantage receive more entrances into their penalty box, increasing the likelihood of conceding a goal. In their study, [RRFGZ13] use one-way ANOVA on 64 matches of the 2006 World Cup tournament to find how entrances into the penalty box relate to team performance and game dynamics. They find that winning teams receive significantly less entrances per game (m=41.42) than drawing (m=50) and losing ones (m=47). It is also shown that teams losing by one (m=0.41), two (m=0.42) and two or more (m=0.34) goals receive less entries per minute into their penalty box than teams winning by one (m=0.54) or two goals (m=0.59). This suggests that when winning, teams alter their strategy to a counter-attacking one and invite pressure to their penalty area. However, they fail to stop the opposition from entering, thus risking to concede a goal [RRFGZ13].

2.3.2 Ball Possession

Other studies research the difference in ball possession between the successful and unsuccessful teams [JJM04, VNH14, LPD10]. [LPGRY17], investigate how top and bottom teams in the Spanish top division (2008/09) adopt different strategies depending on the evolution of a football match (winning, drawing, losing). In their study they find that successful teams had higher percentage of mean possession throughout the season than losing ones, suggesting that higher performing teams were able to maintain their style of play irrespective of the changing variables during the game. Possession was found to be greater when a team was losing or drawing. This might indicate that teams adopted a counter-attacking and direct style of play when in a winning situation. This result, further substantiates that whilst winning, teams tend to have higher entrances into their penalty area as found in [RRFGZ13]. In a similar study [JJM04], looked into 12 matches of successful teams and 12 matches of unsuccessful teams in English Premier League of the 2001/02 season. Similarly, they found that successful teams had significantly more possession than losing teams irrespective of the match state (winning, losing or drawing). They also found similar results for teams that were losing holding more possession of the ball during this period. [VNH14] took a different approach and looked at the defensive aspect of ball possession. They measure the difference in time to retain possession between the top, medium and bottom teams in the German Bundesliga. In their investigations they find that top teams recover the ball quickest. They also note that all groups had the lowest reaction time when they were losing. In other research, possession was found to be significant between top and medium tiers [LBLP10]. In [CCL12], when all world cup matches were considered together, no significantly different ball possession was found between the different groups. However, when taken separately, difference in possession was found to be significant in 2006 and 2010 world cups. [Peñ14] make use of a Markov process to describe the transition probabilities of a team's possession probabilities from historic data of the English Premier League 2012/2013 season. In their work, the authors model the team's game by a finite state automaton using Recovery, Possession, Ball lost and Shot taken as the states of the model. In their investigation they do not consider the goals scored. Their findings show a high correlation between teams' attractiveness in playing style and high values of possession in the model. They also find that aggressive teams are highly correlated with high values of shots taken from the model.

2.3.3 Predicting Football match results

Research was also focused on football match result predictions. The earliest research in this field looked into with techniques as the Poisson and bi-variate Poisson distributions for estimating the final score of football matches [Mah82]. Recently, researchers have also studied the application of Bayesian networks developed with the help of experts in the domain [JFN06, CFN12, RMYA17, BB10]. In general, the use of expert and subjective information in models have been found to make better predictions than using only objective data [CFN12, MKC^{+}08]. In their research, [CFN12] describe subjective data as being important information used for prediction but which is not captured in historical data (objective data). This type of subjective information usually includes the opinion of field experts. These experts indicate which kind of predictors might be used for prediction or give value estimates based on their knowledge. For example, the inclusion of a key player of a team for a particular match as a predictor [JFN06] or giving a estimate of the strength of a team as valued by the expert [CFN12]. Some research has also investigated the use of machine learning techniques [JFN06].

In [BB10], the authors developed a hierarchical Bayesian Network for the prediction of football matches by estimating the number of goals to be scored by each team in match. They used historical data of the Italian Serie A 1997/98 to train the model. The authors then tested the model on a recent season of the Serie A, 2007/08, by comparing the estimated table from the model to the actual final position league table. The results showed that the model estimated some teams better than others. The model under-estimated teams that performed well in the league, whilst it over-estimated those that placed bottom. [CFN12] investigated the application of an expert Bayesian Network for the prediction of English Premier League 2010/11 football match outcomes. The authors considered both subjective and objective factors to build the network. In total they used four factors, with one being objective (team strength) and three subjective (form, fatigue and psychology). The subjective components were developed in collaboration with an expert in the domain. The authors used data of the English Premier League for the seasons between 1994-2010 to formulate the team strength component. In their experiments they demonstrated that with the subjective nodes in the network, the models performed as good as the bookmakers. Using only the objective factor, the performance was significantly worse than both the bookmakers and the model using both objective and subjective data. [CFN12], report that this may indicate that bookmakers do really have information which is not available through public data. They also tested their model for profitability and found that it made a profit in the end of the season using a one-pound betting strategy against a mean price of bookmakers and a popular bookmaker. In a separate study [JFN06], it was shown that Bayesian Networks built with the help of expert domain knowledge out-performed the machine learning techniques. In their research, authors train a Decision Tree, Naive Bayes, KNN and a data-driven Bayesian network with data from the 1995-1997 English Premier League for the matches of the Tottenham Hotspurs Club. The expert Bayesian Network had an overall average accuracy of 59.21%. The Decision Tree was the closest to this score with an average accuracy of 38.65% for disjoint test sets. Interestingly, the expert Bayesian network only considered the inclusion of three players in the team, whether a particular player was playing in midfield, the quality of opposition (high, medium, low) and if Tottenham were playing at home or away. This strengthens the idea that presented in [CFN12], that Bookmakers hold expert information not available in public data. Bayesian Networks excel at complex and interacting factors and are also known to work well with few data [JFN06]. The machine learning techniques only had one team and two seasons to be trained on. There are also other machine learning techniques such Neural Networks and Random Forests that have been shown to perform well in many fields [ATASNG14]. Furthermore, the approaches developed in [CFN12, JFN06] require the services of an expert in the field. Also, given that the models are built subjectively, different experts might come to separate conclusions on what should be modeled. The models would also become irrelevant with time as new players join the club whilst others retire or leave to another one [JFN06]. The authors of [TJ15] investigate the same claim that bookmakers have insider information by using two data sets, one made of public data and the other a hybrid, containing both public and bookmakers data. In the study, they train several classifiers on 13 seasons from the Dutch Eredivise league (2000/01-2012/13) for both data sets.

In [HR11], the authors also compare a number of popular classifiers of different categories to predict the outcome of European Champions League football matches from a particular year. The classifiers investigated were Random Forest, Naive Bayes, Neural Networks, LogitBoost, Bayesian Networks and KNN. The models were trained using two types of feature sets, one basic feature set and another built by an expert. As a part of the feature vector they included the current form of the team, outcome of previous meeting between the teams, the current position of the team in the group table, injured players from the first team and average number of goals scored and conceded per game. The best prediction accuracy was achieved by the Neural network with accuracy up to 68%. Random forest (with basic feature set) was also reported to have best accuracy score of up to 65%. The data set used in their study is unbalanced with 54% of the instances end in home wins. The authors split the data set by the rounds of the group phase into total of three training sets and three validation sets. On the last split, the one containing most training instances and least testing instances the classifiers register a decrease in classification accuracy from the previous split. One reason for this happening might be that teams might have already qualified to the next phase of the competition and so field a lower quality side for the last round. Therefore, the match outcome would not be representative of the form and the other predictors given from the previous rounds [HR11]. The group phase of the European Champions league competition only has 96 matches (32 teams) in total, compared to 380 from a league of 20 teams. Because of these issues, this competition might not be suitable for machine learning use.

In one of the earliest research investigating neural networks for football match predictions [OF97], the authors compare the performance of neural networks with those of human experts and logistic regression to predict the outcome of Australian football league matches. They show that neural networks outperform both human experts (best=70%) and the logistic regression model (best=73.7%), achieving a best accuracy rate of 77% on their test set. To train the network, they use samples from the seasons 1992/93 to 1994/95 of the same league. As predictors, the authors include information such as whether the team is playing home or away, the league table positions of the teams, their head-to-head record, the teams' record in the competing grounds and the time of day (night/day). The test is then carried out on an unseen set from the 1995/96 season. In a more recent study [ATASNG14], the researchers train a neural network on 2068 samples from the Iran Professional League (seven seasons) to predict the correct score of a match. The model was tested on the last eight instances of the 2013/14 league repeated for 30 times. By deducing the match winner from the scores, the model correctly predicted 62.5% (5/8) of the matches, 37.5% (6/16) home/away scores and 12.5% (1/8) exact final match scores.

Other researchers investigated the use of logical rules to predict the outcome of football matches. In [RPR05], the authors use a fuzzy logic model to predict the score difference of football match by considering the result of the competing teams' latest encounter and results of their separate previous five games. The model was trained on 1065 matches from the Finnish first division championship for the years between 1994 and 2001. The model's output could be either of one of the following, big loss, small loss, draw, small win, big win. The fuzzy model was optimised using a genetic algorithm and a neural network. Out of 350 test samples, the overall accuracy for the genetically and neurally tuned fuzzy models were 79% and 87% respectively. The best results were for big loss and big win predictions (genetic tuning=86%, neural tuning=93%). Both had a lower accuracy rate for low scoring game results (genetic tuning=76%, neural tuning=84%).

2.3.4 In-play and Evolutionary Odds

There is little research carried out that combines both in-play data analysis and predictions with sports markets behaviour for the duration of live sporting events. This is particularly interesting because the outcome odds could be used to derive how the market participants are receiving and interpreting information about the game and be compared with the probabilities from the in-game model. The authors of one such study [EU10] make use of a tennis model that processes in-play information of a particular match and outputs the probability of the winner in the duration of the event. The study considers the sport of tennis, however, the same methodology could be applied to football. The probabilities are then compared to those derived from the betting market to see how efficient the market is. The authors tackle the issue of updating the predictions during an in-play tennis event by the use of a mathematical model described in [KM03]. The model updates the predictions of the outcomes of tennis events on a point-by-point basis. The model takes into consideration the scoring system of a tennis match, including points, games, sets, rules and server advantage. The model used in this research is based on the advantage the tennis player has when he/she is serving (player winning their service points) and the probability of the opposing players winning the match (player 1 or 2 win the match). For the first set of values i.e. the probabilities of each player winning the match, the authors argue that by the efficient market theory, the pre-match odds of the game should incorporate all the information such as head-to-head record, injury, form, particular surfaces and other information available before the game. Thus, they plug in the exact odds of each player before the game starts from the betting market. For the second value, (the sum of probabilities of each player winning their service) historical data from the Wimbledon Championships between 1992-1995 is used for both men and woman matches. Interestingly they found that for man the value was 1.29 and for woman 1.12. This indicates that the dominance of the server over the recipient is present in both gender of the sports, but it is greater in matches between men. The predictions derived from the model are then compared with the current price from the event market. From their analysis they find that even though the market is efficient, prices keep trading higher after a break of service occurs, even though it is known that a break gives a greater probability of that player winning the match. Trading decisions however, are not considered in this study. In another research [Øvr08] involving the sport of tennis, the author explores the use of neural networks making use of custom cost function designed specifically for learning trading signals in a betting exchange. The author argues that tennis markets are more volatile than those of other sports such as football and are well suited for trading because of this. Main reason behind his argument is that goals in a soccer game have dramatic effects on the markets and suggest that this may lead to gambling (hoping for an event to occur) rather than trading. However, in [AGF13] the authors explore the same idea but for football matches and markets. The model used in this study predicts whether a goal is to be scored in a particular minute of the game. The authors combine this model with the development of an agent capable of making decisions similar to what a human would make by capturing features from past events. They argue that humans might get emotionally involved in the trades they make, such as in scenarios of when and if they open and close trades. By the use of a Case Based

Reasoning agent, the decisions taken are solely made up from logical reasoning. Their study is not to predict correctly the result of the match but to buy/sell a sports asset for a profit. A profit can be achieved by backing or laying at the appropriate time.

In both of the studies [AGF13, Øvr08] the authors build a trade agent that interfaces with Betfair; A sports exchange system that is similar to how the stock exchange and foreign exchange markets work with Continuous Double Auction but for sports. The agent is able to ask for information such as back and lay odds prices, the current score of the game, the time elapsed in a game and can also send its orders to the interface. From this information the agent in [AGF13] is able to create its base case and then measure the similarity with past events so that the odds for different time periods 1,5,10,15 in the unseen game are generated. In this study they focus on the "under/over 2.5 goals" market. This shows the probability of whether there will be 0, 1, 2 goals scored in total (under) or 3 or more (over) in a particular football event. The case based reasoner agent's performance is evaluated on 60 different matches which are divided into three sets of 20. In each experiment the performance of the 11 human traders is tested with that of the agent. The test involves the prediction of the odds for the next time-steps such as one minute, five minutes, ten and 15 minutes from the current time. Three different testing scenarios are given; no goal is scored, a goal is scored(harder) and what trading decision (back, lay, no bet) to take. Through the tests it was found that the agent performed better at predicting the prices for the first two experiments showing low mean error rates even for the odds predicted at the furthest time-step (15 mins). For experiment on trading decisions, the agent was able to take decisions where humans were not willing and thus chose not to bet, whilst the agent made profit from those particular scenarios. However, experienced traders were able to achieve similar yields from their bets as the agent. This is similar to what was found in [EU10] for the tennis markets. The agent developed in [Øvr08], trades on the match odds of tennis markets (who will win the match). To make it adaptable such that it is able to work better with neural networks the back and lay prices of both players are converted into a binary value representing the probability of the tennis player to win the game which resemble the "asking price" and the "bidding price" in financial markets [Øvr08]. They then experiment with several cost functions including a specific one for the problem, as they argue that the general Sum of Means Error is not adequate for such a problem. For the neural networks to learn the time series data they make use of the sliding window approach. This enables the neural networks to learn a subset of the data that corresponds to a sequence of time just before the point to be predicted. The window is then shifted by one and the neural networks is trained on the following data until it goes through all of the data set. In their tests, the neural networks using sum of means error cost function made an overall negative yield on small tests. However, training the neural networks by using their custom cost function showed that it might be potentially profitable as a small overall positive result was achieved.

The research done on predicting odds movement during in-play football games motivated us to form the main aim of this study with respect to using half-time match statistics as predictors for the machine learning algorithms. From this research, we also saw that it is important to evaluate the models against the implied probabilities from the prediction markets to get a better indication of the models' performance. From the literature review we carried out we have identified the most studied performance indicators that distinguish between winning, drawing and losing teams. These attributes relate to scoring, offensive, defensive and possession motives. We focus the construction of our feature vectors to reflect those that have been shown to be the most successful at discriminating between the different groups. We also noted that the task of identifying and measuring meaningful attributes indicative of a team's performance is hard. We have also seen a number of machine learning algorithms that have been applied to the problem of predicting football match results. The most applied and successful techniques in the literature are neural networks, random forest and Bayesian networks. We use some of these techniques that have been shown to have good results for this problem. For this study, we decide to use random forest, neural networks, naive Bayes and decision trees as our main techniques to investigate the problem with.

2.4 Machine Learning Techniques

In this section, we discuss the machine learning techniques that produced the best results in previous research and which were subsequently used in this study. First, we discuss data pre-processing and why this is essential for certain classifiers. We then describe the chosen classifiers for this research, their internal mechanisms and their advantages and disadvantages. We describe the techniques used in relation to model and feature selection. We end this section by defining the evaluation metrics used for quantifying the performance of the models.

2.4.1 Data pre-processing

Data is usually recorded as it is observed in the form of a statistic by some sensory or manual input. The order of magnitude and the ranges of each attribute recorded may vary largely from the others. Such attributes might be given greater importance during the learning phase by some of the algorithms and so may affect their performance. This is dependent on the internal mechanisms used in the machine learning techniques to 'learn' patterns in the data. For example, the gradient descent algorithm converges faster by using a normalised data set. For other techniques such as decision trees, the data does not need to be pre-processed. One such technique used for data re-scaling is called standardisation. We use standardisation in this study when training Neural Networks.

2.4.2 Standardisation

When the data set is standardised, the values of the different predictors have zero mean and unit variance. The standardisation equation is described in Equation 6, where $\bar{x}$ is the mean of the values for a predictor, σ is the standard deviation of the same predictor and x_i is the actual value. The equation shown is repeated for each value of the feature vector.

$$x_i = \frac{x_i - \bar{x}}{\sigma} \tag{6}$$

2.4.3 Decision Trees

Decision trees can be used for both regression and classification type problems. When used for classification, the decision trees create **if-then-else** rules to split the predictor space using orthogonal lines within and between the different dimensions depending on which predictors are found most useful to split the samples into purer regions of particular classes [KJ13]. In their research, [JFN06] found that the decision tree had the best classification accuracy of all the machine learning algorithms tested with an average accuracy of 38.65%. Because of this we decided to include decision trees as part of our classifiers to experiment with in this study. In decision trees, the predictors used to split the data become nodes of the tree. The first predictor used in the tree is called the root node. The final nodes at the end of the tree are called the leaf/terminal nodes. The leaf nodes make the final prediction to which class an unknown sample belongs to. There are two main methods used to calculate the purity of a split, the Gini index described in Equation 7 and Entropy described in Equation 8. In this study both are used.

Gini Index

$$Gini = p_1(1 - p_1) + p_2(1 - p_2) \tag{7}$$

Cross Entropy

$$Entropy = -p_1(log_2 p_1) - p_2(log_2 p_2) \tag{8}$$

Classification trees, produce an interpretable model, meaning that one could analyse it and understand which decisions are being made to come up with the final classification. Because of the way classification trees work, data does not have to be pre-processed beforehand as the model is not affected by different ranges of data. However, they suffer from model instability. This means that a small change in data used to train the tree may produce a very different model. Classification trees might not be the best predictors if the sub-spaces cannot be split into orthogonal regions. They also suffer from over-fitting. To rectify this problem a method called pruning is used to reduce the depth of the tree by penalising long rules after the training phase of the algorithm [KJ13].

Some of the problems mentioned above can be tackled by using a ensemble technique that creates a number of independent classification trees. This particular type of ensemble technique is called a random forest.

2.4.4 Random Forest

Random forest is an ensemble machine learning technique that can be used for both classification and regression problems [KJ13]. Random forest constructs multiple and independent decision trees on a subset of samples, by using a bootstrapping technique. Bootstrapping takes n' random number of samples from a data set of n instances with replacement. Meaning that the same sample may be used twice or more in the training phase of a particular tree. The rest of the training samples are unique for that training set. This procedure is done several times to build a k number of estimators. This approach adds randomness to the construction of the tree. To classify samples, random forest takes the vote of each classifier built and the sample is then classified to the class predicted by the majority of the estimators. The prediction of the random forest can also be interpreted as a probability by taking the number of estimators that predicted a class over all the number of classifiers used in the ensemble. The main hyper-parameters for random forest are the number of trees to construct(k) and the maximum number of features(f') to be used [KJ13]. In their research, [HR11] found that one of the best classification algorithms was the random forest with full-time result classification accuracy of up to 65%. Decision trees are then trained on a sub-space of the sample and added to the random forest.

2.4.5 Neural Networks

Artificial neural networks are a machine learning technique inspired by how the biological and neurological brain works. The technique makes use of synapses, perceptrons and activation functions such as those found in a biological brain. A number of previous studies [OF97, HR11, ATASNG14], made use of neural network with positive results. In most cases this technique had the best classification accuracy and also outperformed human predictions. The Neural Network makes use of multiple logistic regression to produce a classification. Neural networks have a hidden layer with multiple units in between the input and the output layers where higher order terms are learnt by the network on their own as they are trained on the data [KJ13]. Equation 9 describes the Sigmoid function also known as a squashing function. The output of the Sigmoid function is between 0 and 1, for the input vector $\vec{x}$.

$$h_\theta(\vec{x}) = \frac{1}{1 + e^{-\sum_{i=1}^{n} \theta_i x_i}} \tag{9}$$

The neural networks are at the most fundamental a sum of weighted logistic functions that generate higher order terms able to create complex decision boundaries. For classification, the final output is passed through a final logistic function and output the probability of the sample being in a particular class. For a multi-class problem the softmax function is used such that the final outcome could be interpreted as a probability. Neural Networks make use of back propagation to update the weights of each layer that the target outcome depends upon. Neural Networks generally either use a stochastic method or gradient descent to update the weights of the network. With gradient descent the update is done in batches. Meaning after all the training examples have been seen it updates the new weights according the average sum error of all the training examples. With the stochastic approach, the weights will be adjusted after each sample. With stochastic methods, one could stop the learning phase at any point. Neural networks are also known to overfit the data and the model generated is not easily interpretable.

2.5 Model Selection

When deciding between models and their hyper-parameters re-sampling methods are used in order to have an empirical estimate of the models' performances. K-fold cross-validation is one such techniques that is most commonly used. This technique ensures that the estimate error of the model is representative of the true error. Care has to be taken when using both feature selection and parameter tuning as is in our case. If these techniques are applied sequentially using the same data set, the cross-validation error estimate might be optimistic. In this case nested cross-validation is used for a fairer estimate of the true error [KJ13]. Nested cross-validation first partitions that data set into a number of outer folds which are used as validation sets. Cross-validation is then applied again to the training sets within the outer folds to find the best feature sets and parameters. The best combination is then tested on the outer fold which is left out of the training process for that fold.

2.5.1 k-Fold Cross-Validation

k-Fold cross-validation is a re-sampling method that ensures that all the instances in the data set are considered both for training and testing. In k-Fold cross-validation, the data set is split into k sets. For each experiment a partition is used as the testing set and the rest is used for training. Meaning that, a data set of n number of examples and k partitions, for the first experiment the testing set would be $x_0..x_k$, while $x_{k+1}..x_n$ are used for training. For the second experiment the testing partition moves k examples to the right, such that $x_k..x_{2k}$ would be the new validation set. The training set would now be composed of the left partition $x_0..x_k$ and the right partition $x_{2k+1}..x_n$. This continues until all k partitions are considered for validation. This ensures that all the examples have been considered for both the training and validation of the model. The score from each validation set is retrieved and their mean is found. The mean score should give a good estimate of the performance of the model on unseen data. Figure 1 displays a schematic of the cross-validation process. Sometimes, the examples are shuffled before they are passed through this method. The number of folds used (i.e k) depends on the number of examples in the data set. For large data sets, as a rule of thumb $k = 3$ or $k = 10$ should be enough. For smaller data sets, it might be considered that k be larger, such that only one example is left for testing for each experiment. This method of choosing $k = n$ for k-Fold cross-validation is often called the Leave-One Out method. This technique requires further computation power than techniques using a lower number of partitions (k). When using a large number for k, such as that used for the Leave-One-Out method, further computation is required than

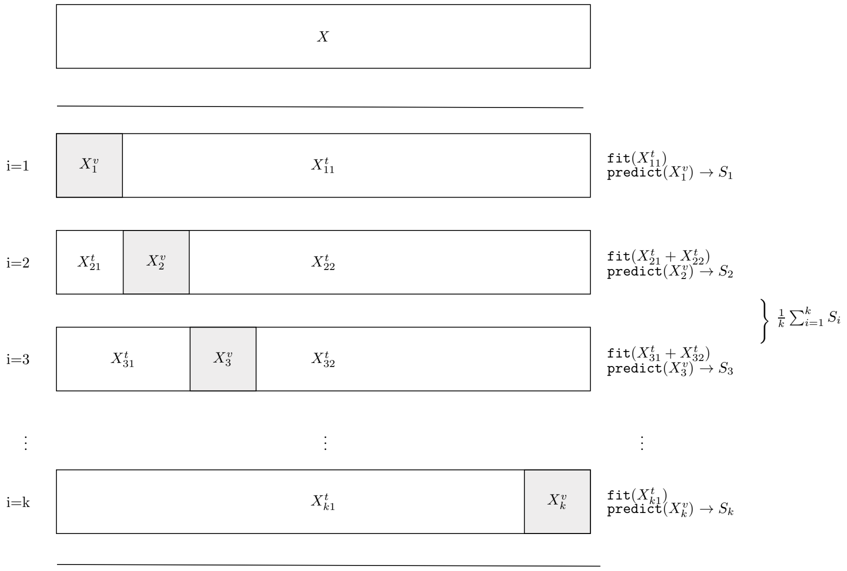

X
i=1
X_1^v
X_{11}^t
fit(X_{11}^t)
predict(X_1^v) $\to S_1$
i=2
X_{21}^t
X_2^v
X_{22}^t
fit($X_{21}^t + X_{22}^t$)
predict(X_2^v) $\to S_2$
$\frac{1}{k}\sum_{i=1}^{k} S_i$
i=3
X_{31}^t
X_3^v
X_{32}^t
fit($X_{31}^t + X_{32}^t$)
predict(X_3^v) $\to S_3$
i=k
X_{k1}^t
X_k^v
fit(X_{k1}^t)
predict(X_k^v) $\to S_k$

Figure 1: Schematic view of k-fold cross-validation.

the other techniques using a lower number of k. This is because further tests are required to be executed.

2.5.2 Evaluation Metrics

Several metrics are used to evaluate the performance of classifiers. With these metrics one could compare the performance of a number of classifiers with respect to different desired qualities such as accuracy, precision and recall. In this study we use the above mentioned metrics and also the f-score, which are all detailed in this section.

Accuracy is the ratio of correctly classified examples over all the examples in the test sample. In the research conducted, classification accuracy is the most utilised metric in this domain for evaluating the models [JFN06, LPGRY17, LBLP10, CCL12, LPD10, JJM04, VNH14, CT05, RRFGZ13, RCH94, TRB10, Peñ14]. With a binary target vector, where the classes can either be positive or negative, a classifier can either classify the instance correctly, such that a positive class is classified as a positive (True Positive) or a negative as a negative(True Negative). In either case the algorithm classifies the examples correctly. However, it can also classify a positive as a negative (False Negative) and a negative as a positive (False Positive). By using these terms, the performance of the classifier with respect to how accurately the model classifies the instances can be calculated as shown in Equation 10.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

The precision equation presented in Equation 11 measures the number of actual true positives over all the samples that were classified as positives by the classifier. The best model is the one that classifies all the samples as true positives and no false positives, giving a precision of 1. In the worst case, the model classifies all the actual negative classes as positives giving a precision of 0.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

Recall, shown in Equation 12, measures the number of correctly classified positive samples over all the actual positive samples.

$$Recall = \frac{TP}{TP + FN} \tag{12}$$

The f-score measure, shown in Equation 13 is the harmonic mean of the precision and recall metrics. This is used to represent the performance of the models in both regards in a single value.

$$F1 - Score = \frac{2rp}{r + p} \tag{13}$$

2.5.3 Brier Score

The Brier Score is a scoring function that measures the accuracy of probabilistic predictions. In previous studies [Con13], this metric was presented as a good measure for evaluating probabilistic predictions for even when more than two outcomes are possible. The score function for a multi-class problem is presented in Equation 14, where n represents the total number of observations, r denotes the possible number of classes, p and o denote the predicted probability and the actual outcome of the j^{th} class for the i^{th} instance, respectively. For this test, the output class is binarised such the observed outcome is represented as either 1 or 0. For example, a three class problem as is in our case, the vector would be represented as three elements, with each position representing the class of the vector. For example, in our case, 0 denotes a home win, 1 and 2 represent a draw and an away win, respectively. The vector $[0, 0, 1]$ would then represent an away win as the vector $[1, 0, 0]$ would represent a home win.

$$BS = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{r} (p_{ij} - o_{ij})^2 \tag{14}$$

2.5.4 Feature Selection

Feature selection is the process of eliminating non-informative or redundant predictors from the feature space of a model. As highly dimensional data is becoming more feasible due to technological advances, finding which predictors to include in a model is becoming more important [KJ13]. In recent years, data collection in sports has become more available and outlets such as Opta are gathering large amounts of player oriented in-game statistics. In this study, we build several data sets from a source that uses Opta as their provider.

Statistically, it is better to estimate a model with fewer parameters. Another desirable effect of having less predictors is that of having a more interpretable model. Non-informative predictors might negatively impact the performance of certain estimators, especially parametrically structured models, such as Logistic Regression and Neural Networks [KJ13]. Other models like Tree-based and Rule-bases models, have built-in feature selection mechanisms that help mitigate the affect of non-informative predictors. One example of such mechanism is bootstrapping.

Feature Selection can either be supervised or unsupervised. In supervised feature selection, the objective is to increase the accuracy of the model or reduce complexity of the predictor subset. In unsupervised techniques outcomes of the samples are ignored in the process of eliminating the predictors. For example, removing predictors that are highly co-related to other predictors. A kind of supervised techniques are called wrapper methods. The objective of these algorithms is to maximise the model's performance by finding the optimal predictor subset by adding/removing predictors to/from the model's feature space. Some of these methods are forward/backward search, simulated annealing and Genetic Algorithms [KJ13]. Wrapper methods are computationally intensive, especially when model tuning is also performed. They are also known to over-fit [KJ13].

2.5.5 Genetic Algorithm

Genetic Algorithm (GA) is an optimisation technique inspired by concepts of evolution principles of population biology [KJ13]. The algorithm mimics the evolutionary process by allowing members of the population to reproduce, survive and mutate to produce children for next generations. The children are then left to compete for survival and the fittest are allowed to reproduce another generation of children. This process continues until the fitness of generations converges to a plateau over time. As we have described in the preceding section, feature selection is a complex problem where the combination of features that produce the optimal prediction are searched for. The fundamental mechanics of Genetic Algorithms are the chromosomes, the making of a population. Chromosomes are made up of genes(each holding a particular value) and their performance is evaluated by their fitness. Genetic Algorithms can be used for the feature selection problem and have been shown to perform well in different fields [KJ13]. In the field of football result prediction they have been used for tuning the parameters of fuzzy logic models in [RPR05]. In this study we use Genetic Algorithms for feature selection. For the scope of feature selection, the chromosomes are encoded as binary vectors representing the predictors of the model. Each gene, denotes the presence(if 1) or absence(if 0) of a predictor in the data. One chromosome represents a solution in the search space in 2^n, where n is the number of predictors.

Genetic Algorithms start with an initial population of randomly selected chromosomes. Every chromosome's fitness is evaluated. This determines the likelihood that the chromosome will be selected for reproduction. Two chromosomes are then selected based on their fitness score to reproduce. The first phase of reproduction is the crossover. Here, the parent chromosomes are split at a random position and the head of one parent chromosome is attached to the other one's tail and vice verse. After the children have been constructed, their genes can be randomly selected for mutation. This means that the chosen genes for mutation get flipped ($0 \leftarrow 1$ OR $1 \leftarrow 0$). After mutation, the child created is added to the next generation. The cross-over-mutation procedure is repeated until the maximum number of the population is reached. The parents themselves can also be included in the next generation. The same process is repeated for each generation created. Figure 2 shows a diagrammatic view of the reproduction phase of the Genetic Algorithm.

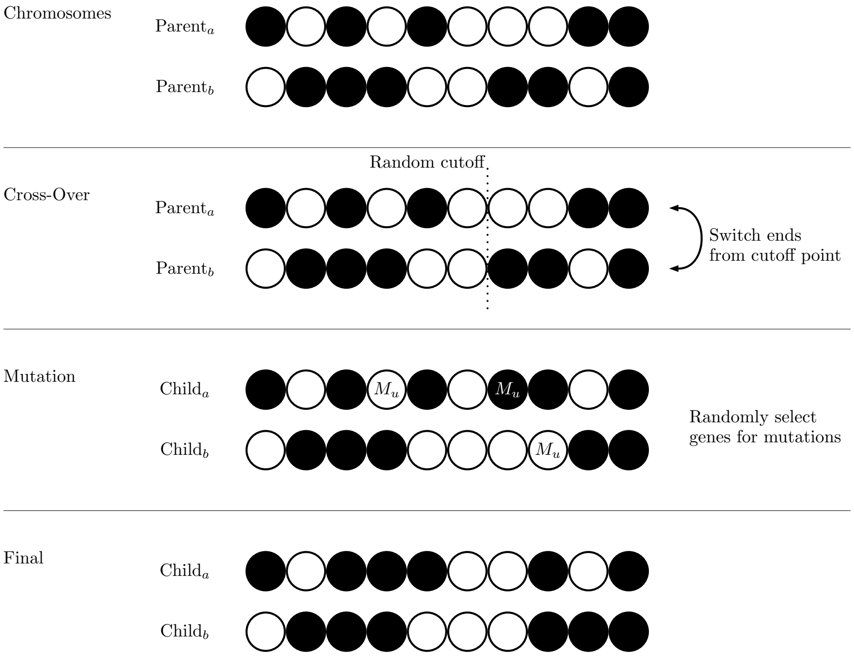


Figure 2: Diagrammatic view of the reproduction process of Genetic Algorithms

3 Methodology

In this chapter we discuss in detail the methods applied for the experiments carried out in relation to the research questions set out in this thesis. For each experiment conducted, we state what kind of data was required and from which data sources these were retrieved. We discuss why the particular data sources were chosen and how the data was represented and extracted from each source. We then describe any pre-processing that was required in order to construct our data sets into formats that could be analysed or input to machine learning models. For each experiment conducted we define the techniques used and outline all the details and parameters that were set for each test. The first section of this chapter relates to the experiments done on analysing the relationship between the half-time and full-time results of football matches. From the observations of the initial experiments, we present a base-rule used for predicting the full-time result of matches by taking into account the half-time result of the match. We then use the results achieved by the base-rule to compare them with those achieved by more advanced models and more detailed data sets, to see if the classification rate improves with such additions. In Section 3.2, we continue to build upon the initial experiments by looking into in-play team performance attributes to predict the full-time match results of drawn games at half-time by the use of a number of classifiers. To conduct these experiments, a more detailed and time-oriented data set was required from the one used in Section 3.1. Of the selected half-time attributes, we take the difference between the home and away team to create the half-time team attribute difference feature vector. The feature vector is then used for fitting of the selected classifiers. The classifiers chosen for this experiment were the Decision Tree, Random Forest, Naive Bayes and Neural Net. Initial experiments using manual feature selection showed that the Random Forest was the most performant classifier. We then describe our implementation of a genetic algorithm used in combination with the random forest classifier to reduce the dimensionality of the feature vector and find the most performing feature subset. In Section 3.3, we continue to look into ways on improving the scores achieved in the methods described in Section 3.2 by aggregating pre-match data on a week game basis to the half-time team performance feature vector. All the experiments conducted in this study were done using *Python 3.5.4*[2] and all the respective figures were plotted using the graphical plotting library, *Matplotlib 2.1.2*[3] library. For all the machine learning algorithms expect for the Genetic Algorithm, the

[2] *Python 3.5.4* - https://www.python.org/downloads/release/python-354/

[3] *Matplotlib 2.1.2* - https://matplotlib.org/

library *Scikit-Learn*[4] [PVG+11] was used. *Scikit-Learn* offers a simple and efficient library containing a multitude of machine learning algorithms. The library provides modules for all parts of the machine learning process, from data preprocessing to model fitting, selection and evaluation, for both supervised and unsupervised learning.

3.1 Predicting the Full-time Result of Matches by their Half-time Results

In this section, we describe the methodologies used to derive a base-rule to predict the full-time result of football matches by taking only into consideration the half-time result of the match. For the experiments carried out in this section we make us of the *Football-data*[5] data set, which is further described in Section 3.1.1. We point out some of the shortcomings of the data set for the purpose of our study, especially in relation to the granularity at which the data is recorded. Because of this limitation, we could not use this data-set for machine learning techniques for the purpose of this study and thus we had to find other data sources described in Section 3.2.1. However, because of the large amount of instances the *Football-data* data set contains and the number of different leagues it covers, we carried out experiments on it by making use of the available attributes. From the results of the experiments, described in Section 3.1.2, it was observed that football matches have a higher tendency of ending in the full-time result state as that of which they are in at the half-time interval. From this observation, we derive the base-rule described in Section 3.1.3, which states that the full-time result of a match will be the same as that of the half-time result. We then use the results achieved by the base-rule to compare them with the results achieved by machine learning techniques making use of data sets containing more in-play statistics.

3.1.1 The Pre-match Data set

Football-data has a large collection of match data dating back to 1994, spanning most prominent European leagues; including English, Italian, Spanish, French and German major and minor leagues, amongst others. The *Football-data* collection includes, for each match in a season, attributes such as the match date, half-time and full-time results, goals scored at each half for both home and away teams and other match statistics that are

[4] *Scikit-Learn* - http://scikit-learn.org/stable/

[5] *Football-data* data set - http://www.football-data.co.uk/downloadm.php

Table 1: Football-data attributes and their descriptions

Attribute	**Description**
`Div`	League Division
`Date`	Match Date
`HomeTeam`	Home Team Name
`AwayTeam`	Away Team Name
`FTHG`	Full-time Home Team Goals
`FTAG`	Full-time Away Team Goals
`FTR`	Full-time Result
`HTHG`	Half-time Home Team Goals
`HTAG`	Half-time Away Team Goals
`HTR`	Half-time Result

usually recorded for football matches including total shots, shots on target, fouls, cards and others. The *Football-data* data set also includes the closing prices (the odds of an outcome at the start of the event) of a number of popular markets from different bookmakers. Table 1 describes a subset of the attributes found in the *Football-data* data set, which are relevant to our study.

Seasons in the *Football-data* collection are represented as separate CSV files, with each row of a file representing a distinct match instance for that year. The columns denote the match details, match statistics and results for each match. Because of how the *Football-data* data set is constructed, it was already in a state for experiments to be carried upon it with little pre-processing required beforehand. Two major features of the *Football-data* data set that made it appealing for us to use in this research are the vast amount of instances contained within the collection and the number of different leagues considered, with respect to both the country of the league and also the number of divisions considered for each country. However, one major problem encountered with this data set for our studies was that all the match statistics, such as, total shots, shots on target, fouls committed, yellow cards, red cards, etc. per team, are recorded as a summation of the whole match. By utilising this data set alone, there is no way to know the attributes' statistic count at other interval periods during the game, such as for example the total number of home team shots at the half-time period. The only attributes of the data set stored in intervals are the match result and the match scores, which are recorded at each half in the following attributes; half-time result (`HTR`), half-time home goals (`HTHG`) and half-time away goals (`HTAG`). Because of this, the *Football-data* data set did not contain enough granularity for the purposes of our experiments. Nonetheless, we carried out experiments on this data set

Table 2: Confusion matrix for the half-time/full-time results. The rows represent the result at half-time and the columns denote that of the full-time result. The intersection of each cell represents the total count for that particular transition from the half-time state to the full-time result. The marginal probability of the full-time result is also denoted in each cell, given it is already in the half-time state.

Confusion Matrix		Full-time Result (FTR)		
		Home Win	Draw	Away Win
Half-time Result (HTR)	Home Win	|H/H| P(FTR=H|HTR=H)	|H/D| P(FTR=D|HTR=H)	|H/A| P(FTR=A|HTR=H)
	Draw	|D/H| P(FTR=H|HTR=D)	|D/D| P(FTR=D|HTR=D)	|D/A| P(FTR=A|HTR=D)
	Away Win	|A/H| P(FTR=H|HTR=A)	|A/D| P(FTR=D|HTR=A)	|A/A| P(FTR=A|HTR=A)

by making use of the attributes which were relevant to our studies described in Section 3.1.2. From the observations noted in the results of these experiments, a base-rule was created such that it could be used for comparisons with the results of future experiments.

3.1.2 Half-time/Full-time Result Confusion Matrix

The first experiment conducted on the *Football-data* data set was to find out how the full-time result of a match varied from its result at the half-time period. And thus see whether one could predict the full-time result of a match by looking at its half-time result. To conduct this experiment a confusion matrix was created for each season of each league as shown in Table 2. The rows of the confusion matrix represent the result at half-time, whilst the columns represent that of the full-time result. The cells represent a count for each transition of a distinct match being in a state at half-time and ending at another in full-time.

This experiment was run separately for each season of each league considered in the test. The same experiment was then also run using all the instances from the different seasons and leagues to come up with the final matrix. Table 3 shows a summary of the leagues considered for this experiment.

Another experiment was then run using also the scores of both teams at half-time. The difference of these attributes were taken such that a new attribute was created; `halftime_goal_diff`. All instances were then grouped together by this attribute and the same method was run again to see how the full-time result varies when also taking into account the difference in goals the winning team has over the losing one at half-time. Someone having some basic understanding of the game of football should have the following

Table 3: Football-data data set

League	**Teams/Division**	**Instances/Season**	**Instances/10 seasons**
England Premier League	20	380	3,800
England Championship	24	552	5,520
England League 1	24	552	5,520
England League 2	24	552	5,520
English Conference	24	552	5,520
Italy Serie A	20	380	3,800
Italy Serie B	22	462	4,620
Bundesliga 1	18	306	3,060
Bundesliga 2	18	306	3,060
Spain La Liga Primera	20	380	3,800
Spain La Liga Segunda	22	462	4,620
French Ligue 1	20	380	3,800
French Ligue 2	20	380	3,800
Scotland Premier League	12	228	2,280
Scotland Division 1	10	180	1,800
Scotland Division 2	10	180	1,800
Scotland Division 3	10	180	1,800
Belgium League	16	306	3,060
Greece Super League	16	240	2,400
Netherlands Eredivise	18	306	3,060
Portugal Liga 1(2006/07 - 2013/14)	16	240	1,920
Portugal Liga 1(2014/15 - 2015/16)	18	306	612
Turkey Futbol Ligi 1	18	306	3,060
Total			**77,755**

expectations from such experiments;

- Teams playing at home should be more likely to win at full-time.
- The higher the goal advantage is for a team at half-time (`goal_diff`), the more likely the winning team is favoured to win the match, i.e the harder it is for the losing team to score as much or more goals, without conceding, in order to change the result in its favour. For example, a team winning by a margin of two goals at half-time should be more likely to win the match than a team having only a one goal advantage.

3.1.3 Match Result Prediction Base Rule

When analysing the *Football-data* half-time/full-time matrices, we noticed that the full-time result tended to be similar to that at the half-time. When considering all the instances of the 220 seasons over the ten years, the home-team won 79.0% of the matches in which they were leading at half-time. Furthermore, when the away team was winning at half-time, the full-time result ended as an away win 68.0% of the times. With these results, we noticed the overwhelming tendency of the full-time result to be the same as that of the half-time when one of the teams is winning. For drawn games at half-time, 38.0% of them ended in the same result, whilst 37.% ended as a home win and 25.0% finished as an away win. Because of this, it was decided to focus exclusively on the matches that were drawn at half-time because the full-time result was less predictable when the match was in this state. The results of this experiment are discussed in further detail in Section 4.1. The high retention rate between results at half-time and full-time led us to define the base-line rule so that we could compare the results from the models with those from the rule. The rule is an identity function which returns the input it is given. It is presented in Equation 15, where R_{htr} is the half-time result.

$$BR_{ftr}(R_{htr}) = R_{htr} \tag{15}$$

3.2 Predicting Match Results using Team Performance Metrics at Half-time

In this Section we describe the methodologies used to investigate the major claim of this study. That being the ability to train machine learning algorithms to predict the full-time result (`H`, `D`, `A`) of matches by the use of in-play performance statistics of two competing teams as their input. We select a number of attributes, which through our research and experience in the game think that are good at discriminating between winning and losing teams and take their summation up till the half-time interval. These team statistics are then used to train a number of selected classifiers to predict the full-time result of the match. As we previously saw in the experiments of Section 3.1, the goal difference at the half-time interval plays a major role in deciding the full-time match result of the game. So for the experiments conducted in this Section we decided to make use only of instances which were in a draw at the half-time interval. Given that the goal difference at the half-time would always be 0 for drawn matches, the task to classify the instances correctly should be more challenging for the machine learning algorithms. This action should also lead to more informative results on the other discriminating attributes used given that the goal difference is not available for the classifiers. Because the data set used in Section 3.1 was not granular enough to test out this claim, we had to find an alternative data source. Through our research we had found that the best available data source for our experiments was the *Match Centre* data found for each match at the *WhoScored*[6] website, because it contained the desired features as described in Section 3.2.1. However, the data was not in a format ready to be fed into machine learning algorithms. Thus, after extracting the data from the site, several pre-processing steps discussed in detailed in Section 3.2.3 had to be implemented using conditional logic on the raw statistics in order to transform the raw data as described in Section 3.2.2 to the final feature and target vectors, detailed in Section 3.2.4. The final data set was uploaded and made public on *Gitlab*[7]. Once we had constructed our data set we were ready to start experimenting with the chosen classifiers by manually selecting the features we thought were most important at classifying the instances into their correct category, as discussed in Section 3.2.5. We carried out this initial experiment to get a better intuition at the sensitivity of the data and find the best performing classifier. We then continued experimentation by using the implemented genetic algorithm described in Section 3.2.6 to find the best performing subset of features. The experiments were done per league/season such that comparisons between the different subsets selected by the GA could be made for each league over the years and also to compare the selected features between the different leagues. Such experiments where done to identify both the similar and different features selected which were chosen by the stochastic process.

[6] WhoScored website - https://www.whoscored.com

[7] Public data set - https://bit.ly/2QdlCs6

3.2.1 The In-play Data Source

As mentioned in the Section 3.1, even though the *Football-data* data set contained both varied and substantial amount of instances, the intervals at which the statistics were recorded were not granular enough for the purpose of our studies. Because of this, other data sources that included in-play data had to be found. After going through several options, we had elected to use the data present on the *WhoScored* site. *WhoScored*'s data set had several characteristics which were suitable for our studies, of which, the most important were the following;

1. The data is recorded at a play-by-play rate. Meaning that every action taken in the duration of a match is accounted for, such that one could extract all or some actions taken at or till a particular minute and second of the match.

2. Each action is labelled with a type, the x and y coordinates of the ball at the time of play and other characteristics describing in more detail what kind of action it was. For example, a 'Pass' action by a player taken at the (28.5,49.9) coordinates of the pitch and the rest of the subtypes indicate that the pass was a long ball, a cross and chipped.

3. As stated by *WhoScored*, its data source for the in-play match statistics is Opta Sports. Opta Sports is the gold standard of in-game match data and are used by top punditry outfits and the like for detailed match statistics and analysis.

4. *WhoScored* is constantly being updated with the most recent play-by-play data for the latest seasons and competitions, whilst going back to the year 2009/10 for the most popular leagues in Europe, including the English Premier League, Italian Serie A, Spanish La Liga, French Ligue and the German Bundesliga. For other less popular competitions, such as the Major League Soccer, English Championship, German Bundesliga II and others, it features only more recent in-play data.

In the rest of this section we will describe first how *WhoScored*'s raw statistics are represented. We will describe how these were then transformed into attributes that better described the teams' performances during the match and why such attributes were given importance to others for describing the match state. In turn we will then describe how these were then transformed into feature and target vectors for the input of several machine learning classifiers. We will discuss the process which was used to manually select features

Table 4: *WhoScored* match information data

Attribute	**Description**	**Type**
`home.name`	Home Team Name	String
`away.name`	Away team name	String
`home.teamId`	Home team Id	$\{e \in \mathbb{N} \| e > 0\}$
`away.teamId`	Away team Id	$\{e \in \mathbb{N} \| e > 0\}$
`home.averageAge`	Home team average age	$\{e \in \mathbb{R} \| e > 0\}$
`away.averageAge`	Away team average age	$\{e \in \mathbb{R} \| e > 0\}$
`htScore`	Half-time score	String, denoted as “home:away”
`ftScore`	Full-time score	String, denoted as “home:away”

such that a better understanding of the data set could be garnered. Finally, we will discuss how a genetic algorithm was then used to automate the process for finding the best performing set of features in a stochastic manner.

3.2.2 Composition of the *WhoScored* Raw Data

The *WhoScored* data set is made up of a series of JSON(JavaScript Object Notation) files, in a way that each file represents a particular match. Each JSON file includes many interesting data on a match, such as individual player data and their ratings, however, not all of them were considered for our studies. The reason why these were not used was because of the scope of the project. We planned to construct the data set using team attributes that we deemed were most important at discriminating between winning and losing teams using both our intuition of the game and also from previous literature. However, we do think that it would be important to make use of this available data in future research. For our experiments we were more interested in the play-by-play actions of each match, such that the teams' overall performance until a particular point in time of the event could be summed up into in-game statistics for each team. Each file was traversed and the relevant data was extracted from each one so that for every JSON file, a new CSV file was created. The CSV file represents in chronological order, all the actions taken in a whole match, with each row defining an action that happened during the event at a particular minute and second of the match. The columns describe what type of action it was, by which team and player it was taken, the position on the pitch it happened at (if applicable), the minute and second of game time, and other attributes that give more detail on how the action was executed. In the rest of the section we describe the most important data captured from the *WhoScored* data source.

The match information data described in Table 4 defines the type of information found specific to the match and the participants partaking in the event. These fields are repeated for each row/action recorded in the CSV file. Each action that happened during the match is then described in detail. Information such as who was the player and of which team he is playing for performed or received the specific action, at which coordinates of the pitch was the player when the action was executed, at which position on the field did the ball end at (for `Pass` and `Shot` types), was the action a successful one or not. This information is available for every action taken during the event and all of the attributes and their types are fully described in Table 5.

Every action in turn has other attributes describing it in further detail depending on the action's type. This means that an action of the type `Pass` might have a number of different sub-attributes than those actions of types `SavedShot` or `Foul`. For example, a `Pass` action can have sub-attributes such as `Chipped`, `HeadPass` and `Longball` whilst a `Card` can be either `Yellow` or `Red`. Table 6 describes all the sub-attributes used in our study because they were required to create the team performance attributes chosen. These are just a few of the many other attributes present in the *WhoScored* data set. Other attributes which are of great interest but were not required because of the scope of the project and thus not considered in our study include action details for different types such as `RegularPlay`, `Fastbreak` and `Throughball`, amongst others.

3.2.3 Constructing the In-Play Data set

The raw data set is presented in a time series format and each individual row is an independent action performed in the game. Because of this fact, further processing to the data set had to be carried out such that the individual actions performed by the players be aggregated together to represent a team's contributed effort in the match till a certain point in time. Because of the base-rule derived from previous experiments using the football-data data set we had decided that we should consider the aggregated team metrics at half-time such that we could compare the results achieved by using these team statistics to the base rule. We then decided upon which aggregated team metrics to consider from the *WhoScored* data set by using both our intuition of the game and from previous literature carried out, in particular the ones which studied the effect of certain attributes that differentiated successful team from the others and also those that studied the impact that certain attributes could have on the outcome of a game [LPGRY17, LBLP10, CCL12, LPD10, JJM04, VNH14, CT05, RRFGZ13, RCH94, TRB10, Peñ14]. Using both these references and taking into consideration the available data at hand for the selection of the

Table 5: *WhoScored* action data

Attribute	**Description**	**Type**
TeamId	Identifies the team of which one of its players performed the action.	$\{e \in \mathbb{N}\}$
playerId	Identifies the player which performed the action.	$\{e \in \mathbb{N}\}$
minute	Indicates the minute of the match time the action occurred at.	$\{e \in \mathbb{N}\}$
second	Indicates the second of the match time the action occurred at.	$\{e \in \mathbb{N} \mid e < 60\}$
period	Indicates in which period of the game (1st half or 2nd half) the action happened in.	$e \in \{\texttt{1}, \texttt{2}\}$
outcomeType	Describes whether or not the outcome of the action was a successful one. In some case (depending on the type) it describes to which team that action applies to. For example, value of *0* indicates that a corner was awarded to the away team.	$e \in \{\texttt{0}, \texttt{1}\}$
isTouch	Defines whether the action should be considered as a possession or not.	$e \in \{\texttt{0}, \texttt{1}\}$
type	Describes what action was performed at this point in time during the game.	$e \in \{$ `TakeOn`, `SavedShot`, `MissedShot`, `Goal`, `Pass`, `CornerAwarded`, `Foul`, `OffsideGiven`, `Interception`, `Card`, `Challenge`, `Tackle`, `TakeOn` $\}$
x	Describes the x coordinate of the pitch the action happened at.	$\{e \in \mathbb{R} \mid 0 \leq e \leq 100\}$
y	Describes the y coordinate of the pitch the action happened at.	$\{e \in \mathbb{R} \mid 0 \leq e \leq 100\}$
endX	Describes the x coordinate of the pitch the action ended at, depending on the type of action that was performed.	$\{e \in \mathbb{R} \mid 0 \leq e \leq 100\}$
endY	Describes the y coordinate of the pitch the action ended at, depending on the type of action that was performed.	$\{e \in \mathbb{R} \mid 0 \leq e \leq 100\}$

Table 6: *WhoScored* extended action data for `Pass` and `Card` types

Attribute	**Description**	**Type**
isKeeperThrow	Describes whether a `Pass` type was a throw by the keeper (`True`) or not (‘“’), thus being a regular pass.	$e \in \{$`True`,‘“’$\}$
isCross	Describes whether a `Pass` type was a medium to long range effort from a wide area on the pitch towards the center of the field close to the opponent's goal (`True`).	$e \in \{$`True`,‘“’$\}$
isLongball	Describes whether a `Pass` action type was an attempt to move the ball a long distance from a deep position in the field to an attacking area in the opposition half (`True`) or else not (‘“’).	$e \in \{$`True`,‘“’$\}$
isIntentionalShotAssist	Describes whether a `Pass` action type leading to a shot towards the opponent's goal was set up intentionally by the player performing the action (`True`) or else not (‘“’).	$e \in \{$`True`,‘“’$\}$
isIntentionalGoalAssist	Describes whether a `Pass` action type leading to a shot and goal against the opponent was set up intentionally by the player performing the action (`True`) or else not (‘“’).	$e \in \{$`True`,‘“’$\}$
isYellow	Denotes that a `Card` action received by a player was a yellow card.	$e \in \{$`True`,‘“’$\}$
isRed	Denotes that a `Card` action received by a player was a red card.	$e \in \{$`True`,‘“’$\}$

Table 7: Selected In-play team attributes

Team Attribute	Description
SHOT_TOTAL	Total shots performed by a team
SHOT_ON_GOAL	Total shots on target
GOAL	Number of goals scored
ASSIST_SHOT	Total assisted shots
ASSIST_INTENTIONAL	Number of intentionally assisted shot.
ASSIST_INTENTIONAL_GOAL	Number of intentionally assisted shot that resulted in a goal.
PASS_TOTAL	Total number of passes carried out by a team.
PASS_SUCCESS	Number of successful passes carried out by a team.
PASS_LONG	Number of long passes carried out by a team.
PASS_FORWARD	Number of forward passes (towards the opponent's goals) carried out by a team.
PASS_BACKWARD	Number of backward (away from opponent's goal) passes carried out by a team.
PASS_TARGET_FINAL_THIRD	Number of passes directed towards the third half of the opposing side that was carried out by a team.
PASS_TARGET_MIDDLE_THIRD	Number of passes directed towards the middle of field that was carried out by a team.
PASS_TARGET_DEFENSIVE_THIRD	Number of passes directed towards the team's own side of the field that was carried out by a team.
CORNER_FAVOUR	Number of corners awarded to a team.
FOUL_RECEIVED	Number of fouls awarded to a team.
CROSS_FAVOUR_TOTAL	Number of attempted crosses carried out by a team.
CROSS_FAVOUR_SUCCESS	Number of successful crosses carried out by a team.
OFFSIDE_COMMITTED	Number of times the team was caught offside.
POSSESSION_ATTACK	Number of times the team had possession of the ball inside the opposing team's half.
POSSESSION_TOTAL	Total number of possessions the team had of the ball.
POSSESSION_DEFENCE	Number of times the team had possession of the ball inside their own half.
INTERCEPTION	Number of interceptions made by a team to the opponent's passes.
CARD_YELLOW	Total number of yellow cards received.
CARD_RED	Total number of red cards received.
TACKLE_TOTAL	Total number of attempted challenges performed by a team.
TACKLE_SUCCESS	Total number of successful challenges performed by a team.
DRIBBLE_TOTAL	Number of times the team attempted to dribble past (move with the ball forward) a member of the opposing team.
DRIBBLE_SUCCESS	Number of times the team successfully dribbled past (move with the ball forward) a member of the opposing team.

team performance attributes, we finally decided upon using the list described in Table 7.

After it was decided that these would be the attributes to be considered for measuring a team's performance until the half-time period, conditional logic rules had to be used on the present data in the *WhoScored* raw statistics such that the team attributes could be constructed. Table 8 gives a detailed description of how each attribute was constructed. For all the matches present in the data set each action was checked against the conditions shown in Table 8 and every time a condition was met, the count for that respective attribute was incremented by one for the particular team. As one could notice in Table 8, the period of the match is always required to be of value one. This condition was used to serve the purpose of limiting the counts to the first half of the game. The rest of the conditions for the specific attributes were put together in a way such that the final amount of the counts would be equal to those shown on the *WhoScored* website. This measure was taken in order to be able to reference the final amount computed back to the original source. This means that the total count of an attribute for a team of a specific match computed by the construction rules would equal the amount shown on the *WhoScored* website for that same attribute. For example, actions of the type `Pass` and also being a keeper throw (by hand), a throw-in (isTouch = 0) or a cross are not considered to be passes in the summation for

Table 8: Transforming raw statistics into team performance attributes

Team Attribute	Action Type	period	Construction Conditional Logic
SHOT_TOTAL	SavedShot|MissedShot|Goal	1	
SHOT_ON_GOAL	SavedShot	1	
GOAL	Goal	1	
ASSIST_SHOT		1	(shotAssist)
ASSIST_INTENTIONAL		1	(shotAssist) & (_IntentionalShotAssist)
ASSIST_INTENTIONAL_GOAL		1	(shotAssist) & (_IntentionalGoalAssist)
PASS_TOTAL	Pass	1	(isTouch = 1) & (¬ _cross) & (¬ _KeeperThrow)
PASS_SUCCESS	Pass	1	(isTouch = 1) & (¬ _cross) & (¬ _KeeperThrow) & (outcome = 1)
PASS_LONG	Pass	1	(isTouch = 1) & (¬ _cross) & (¬ _KeeperThrow) & (_Longball)
PASS_FORWARD	Pass	1	(endX > x)
PASS_BACKWARD	Pass	1	(endX < x)
PASS_TARGET_FINAL_THIRD	Pass	1	(endX > 66.66)
PASS_TARGET_MIDDLE_THIRD	Pass	1	(33.32 < endX < 66.67)
PASS_TARGET_DEFENSIVE_THIRD	Pass	1	
CORNER_FAVOUR	CornerAwarded	1	(outcome = 1)
FOUL_RECEIVED	Foul	1	(outcome = 1)
CROSS_FAVOUR_TOTAL	Pass	1	(_Cross)
CROSS_FAVOUR_SUCCESS	Pass	1	(_Cross) & (outcome = 1)
OFFSIDE_COMMITTED	OffsideGiven	1	
POSSESSION_TOTAL		1	(isTouch = 1)
POSSESSION_ATTACK		1	(isTouch = 1) & (x > 49)
POSSESSION_DEFENCE		1	(isTouch = 1) & (x < 50)
INTERCEPTION	Interception	1	
CARD_YELLOW	Card	1	(_Yellow)
CARD_RED	Card	1	(_Red)
TACKLE_TOTAL	Challenge|Tackle	1	
TACKLE_SUCCESS	Tackle	1	
DRIBBLE_TOTAL	TakeOn	1	
DRIBBLE_SUCCESS	TakeOn	1	(outcome = 1)

the total passes of a team on the website. Thus, when making the conditional rules for construction of the data set's attributes this was taken into consideration such that the computed amount for the attribute would equal that of how it would have been presented on *WhoScored*.

3.2.4 The Feature and Target Vectors

The final step before being able to use the data for training a classifier was to turn the amassed attributes into feature and target vectors. A feature vector for each event was created by taking the half-time attributes of both teams and subtracting the away team's values from those of the home team, such that the difference of that attribute with respect to the home team is recorded. This means that for each attribute, if the value from the subtraction resulted to a positive value it means that the home team had accumulated more of that attribute than the away team. On the other hand, if the result was negative, it means that it was the away team that had more counts of that attribute. A result of 0 means that both had exactly the same counts. Along with the differences of the attributes at half-time, the difference in average age between the two teams was also added to the feature vector as it was thought that this could be an important factor for deciding the output of a match. It was decided that the difference of the attributes be taken instead of using the

Table 9: Example of a feature vector with corresponding targets showing the Arsenal match instances of the 2013/14 season of the English Premier League. The targets represent the full-time result of the match with, 0 denoting a home win, 1 representing a draw and 2 an away win. The attributes represent the differences in in-play stats at half-time between the home team and the away team. The attributes shown in the table represent the following; STD - ShotTotalDiff, SOGTD - ShotsOnGoalTotalDiff, PTD - PassesTotalDiff, PLD - PassesLongDiff, PSD - Passes Success Diff, PassesBackDiff, PTFTD - PassesTargetFinalThirdDiff, TSD - TackleSuccessDiff, DTD - DribbleTotalDiff.

match	target (FTR)	STD	SOGTD	PTD	PLD	PSD	PBD	PTFTD	...	TSD	DTD
Arsenal v Aston Villa	2	2	2	115	-10	123	44	22	...	-3	8
Arsenal v Cardiff	0	5	1	132	-4	138	49	64	...	-5	8
Arsenal v Chelsea	1	-4	-3	89	4	86	46	12	...	-5	1
Arsenal v Crystal Palace	0	4	2	297	-2	302	120	139	...	-3	-6
Arsenal v Everton	1	-2	1	-123	-20	-131	-69	-2	...	5	-4
Arsenal v Fulham	0	7	3	42	-19	52	12	95	...	-1	3
Arsenal v Hull	0	11	5	232	-20	243	86	123	...	-4	8
Arsenal v Liverpool	0	2	0	118	1	102	58	76	...	10	-7
Arsenal v Man City	1	-6	-3	-80	0	-75	-37	-25	...	-1	7
Arsenal v Man Utd	1	6	4	33	1	27	1	52	...	-4	8
Arsenal v Newcastle	0	8	5	130	-1	133	48	50	...	-3	-1
Arsenal v Norwich	0	3	0	22	-8	27	16	23	...	4	-3
Arsenal v Southampton	0	-1	-2	7	-3	8	-4	19	...	-4	6
Arsenal v Stoke	0	7	4	179	-9	184	94	45	...	-2	13
Arsenal v Sunderland	0	9	5	280	0	279	131	131	...	-4	9
Arsenal v Swansea	1	7	4	105	-4	110	52	143	...	-7	7
Arsenal v Tottenham	0	3	-2	-27	-14	-40	-18	-1	...	5	-8
Arsenal v West Brom	0	4	4	182	15	176	87	55	...	-1	-3
Arsenal v West Ham	0	1	2	147	2	146	55	50	...	-4	2

actual counts themselves for the following main reasons; first, to reduce the dimensionality of the feature space by half and second, since the teams are competing against each other it is more intuitive that the difference between their performance attributes be considered rather than the total amount of the statistic. In the experiments conducted with this data we wanted to test the claim that the full-time result of a match can be predicted by a classifier by considering the differences in the team performances at the half-time period and correctly classify the end game result. Thus the target vector for each instance had to include only one element; the full-time result of that match. The target vector included to the data set is thus represented as one element which must be one of the values $\{0, 1, 2\}$, where 0 denotes a home win, 1 denotes a draw and 2 denotes an away win. Table 9 shows examples of the instance feature vectors with their corresponding target vector. A distinct and separate data set was created per season league of the data collected from whoscored, such that in the end we had a total of 35 data sets after all the raw statistics were converted into the feature vectors shown in Table 9.

Table 10: In-play data set summary. The cells marked in bold represent the instances that were used for the experiments in this study; meaning the matches that were drawn at half-time.

League	Total Instances	Half-time Result			Full-time Result		
		Home Win	Draw	Away Win	Home Win	Draw	Away Win
English Premier League	2,660	927	**1,095**	638	1,217	686	757
Italian Serie A	2,279	792	**1,001**	486	1,037	609	633
Spanish La Liga	2,660	987	**1,072**	601	1,301	621	738
Germany Bundesliga	2,142	733	**855**	554	960	523	659
France Ligue 1	2,280	781	**1,006**	493	1,014	651	615
Total	12,021	4,225	**5,029**	2,772	5,529	3,090	3,402

In total 12,021 instances of in-play match data were collected. Of these instances, 5,029 were used in the study. This was because we decided to keep only the instances which were in a draw at half-time as can be seen in Table 10. We focused only on draw matches because the full-time result was less predictable when events were in this state. This makes it a more suitable problem for machine learning. In Table 10, the instances are further categorised by their league. As can be noticed for each league the most common result at half-time is the draw. However, by the full-time result, a home win is the most probable result.

3.2.5 Machine Learning Classifiers with Manual Feature Selection

Once the feature and target vectors were created, we were ready to start carrying out experiments by fitting the data sets into different classification models. We had decided to carry out our initial experiments by using a number of different types of classifiers so that we could understand better which were those that performed better with the data available. The classifiers that were used for these experiments were the following; Decision Tree classifier, Random Forest classifier, Neural Networks and Gaussian Naive Bayes classifier. For each classifier the default settings were used and no tweaking of the models was carried out to improve any of the scores during this phase of experimentation.

We had also decided that for this experiment the features will not be fed all into the classifiers at once, but rather start with a small set of features and incrementally and manually add new ones after each test and reevaluate the performance of the models with the added feature. After the tests we could see whether the feature added had increased or decrease the performance of the different classifiers. We would then decide to drop or keep the feature depending on the score achieved in the tests. We used the mean accuracy score of the prediction results achieved from 3-fold cross-validation during the training of the classifier as the metric to compare the classifiers' performance. Each classifier was given the same set of feature vectors and for each classifier the experiment was run twice, one with raw and the other with scaled data. To compare the results of each classifier, a Classifier by Feature Set matrix was built up for each experiment. The columns of the matrix specify a particular classifier and whether scaling of the data is used or not, and each row specifies the feature set used. The intersection of each is the score achieved by that classifier for that feature set. The initial test carried out in this experiment was to see which of the attributes held most information on their own to be able to predict the outcome of the game. After each test, then we would choose the best performing set and then use it as the base set for the next test where other attributes are added to it. Sometimes we also branched out to see how sets with lower scores than the best ones but which are deemed important in football would perform with additional attributes. This process was repeated several times until we reached a point when the scores of the classifiers where not varying much after with the additional attributes being added to the set. These tests were conducted on the English Premier League 2015/16 Season data.

3.2.6 Automated Feature Selection using Genetic Algorithm with Random Forest Classifier

The experiments using manual feature selection across several classifiers helped us to identify the most performant classifier on a sample of the data set, which was the random forest classifier. By using this fact learned from the previous experiments, we decided to apply a stochastic approach to find the most informative feature subset across the whole data set in an automated manner specifically for the random forest classifier. We decided to use a genetic algorithm as our approach which is described in Algorithm 5. In the algorithm, C denotes the type of classifier used for fitting the data. In our experiments this was the Random Forest. The other input parameters of the genetic algorithm are the following; E is the maximum number of epochs that could be run, N is the chromosome population size, X_t is the training input vector, Y_t is the training target vector, X_v is the validation input vector and Y_v is the validation target vector, G is the parameter grid, th is the stopping threshold, r is the percentage of chromosomes to keep between generations and k is the number of internal folds to be performed for each parameter in the grid.

Using a genetic algorithm as a feature selection mechanism requires that the feature set is represented as a chromosome. A chromosome is a string of bits {0,1}, with its length being equal to the length of the object it represents, in our case the feature set.

The information that a specific chromosome holds translates to the features that are to be considered for the training of the classifier, represented by (`1`) if present or (`0`) if not. To measure the performance of a particular chromosome (feature subset), we implemented a fitness function specifically for this case. The score returned by the fitness function, reflects how accurately the feature subset performed classification on the data set with the least number of features used. K-fold cross-validation with 3 folds was used to get an empirical score of the classifier and the mean of the scores returned by each fold was used as the final accuracy score produced by the feature subset. The fitness function rewards chromosomes with fewer features, such that of two subsets resulting in the same mean accuracy score from the inner cross-validation, the one with the least amount of features would score higher than the one containing a larger set of features. This was done to promote feature sets of a lower dimensionality. The fitness function implemented for this experiment is described in Equation 16, where $\overline{s}$ denotes the mean score achieved from the 3-fold cross-validation test, c_0 denotes the count of 0s making up the chromosome , 0.4 is a scalar value used to adjust the additional score given to smaller subsets and f is the fitness score. However, for all the experiments carried out with genetic algorithms, this scalar was always left at that value.

$$f = (10^4 \times \overline{s}) + (0.4 \times c_0)) \tag{16}$$

The fitness function is further described in Algorithm 1, where G is the parameter grid, C is the classifier type, s is the encoded chromosome in binary and k is the number of cross-validation folds to be performed. The function returns a tuple containing the highest achieving parameter set and its fitness score.

Algorithm 1 Fitness Function with GridSearch

```
$G$: parameter grid
$C$: classifier class
$s$: chromosome
$k$: number of cross-validation folds to perform
function GRIDSEARCHFITNESSFUNCTION($G$,$C$,$s$,$k$)
    $X \leftarrow X[, \vec{s}]$                          ▷ filter out features not in $\vec{s}$
    $m,\ \vec{p} \leftarrow GridSearch(C, G, X, Y, k)$
    $nzeros \leftarrow |\vec{s}| - sum(\vec{s})$       ▷ Cardinality of $\vec{s}$ minus of sum its elements
    return $(m * 10^4) + nzeros,\ \vec{p}$
end function
```

The chromosomes are then sorted by their scores in a descending order such that the highest performing chromosome are at the top. A predefined percentage of the chromosome population is selected to be the parents of the new off-springs in the next generation. A pair of chromosomes are selected randomly from the parent pool and a random line at which both parent chromosomes are to be split at is found such that the new children chromosomes could be created from the union of inverse partitions from both parents. This procedure is called a cross-over and is described in Algorithm 2, where p_1 and p_2 are the parent chromosomes as parameters of the function, whilst c_1 and c_2 are the children chromosomes returned by the function.

Algorithm 2 CrossOver Algorithm

```
$p_1$: chromosome
$p_2$: chromosome
function CrossOver($p_1, p_2$)
    $r \leftarrow random() * |p_1|$                    ▷ randomly choose a split point
    $c_1 \leftarrow p_1[: r] + p_2[r :]$
    $c_2 \leftarrow p_1[r :] + p_2[: r]$
    return $c_1, c_2$
end function
```

After the cross-over of the two parent chromosomes, the resulting chromosomes pass through another function referred to as the mutation function. In this function, a random number between 0 and 1 is generated for each location of the genes in the chromosome. If the generated random number for a location in the chromosome is greater than a defined threshold, the value of that gene is flipped, such that $if1 \leftarrow 0$. In our version of the mutation function described in Algorithm 3, the threshold is dependant on the epoch number such that, at the start, when the generation number is low, the threshold is close to 0. The mutation function accepts as parameters; p as the encoded chromosome in binary, i as the current epoch number and n as the maximum number of possible epochs. The algorithm then returns p as the mutated chromosome.

As the generation number increases, the threshold also increases. This means that mutations become less common as the epoch counter reaches the maximum value. We decided to implement a mutation function so that the early generations would experience higher mutation rates such that more random search space is covered in the beginning. We also wanted that the later generations' population would have more of the best chromosomes with less mutations to try and maximise the scoring function. The mutation function is described in Equation 17, where i is the current epoch number, n is the maximum epoch and mt is the mutation threshold.

Algorithm 3 Mutation Function

```
p: chromosome
i: current epoch
n: total number of epochs
function MUTATE(p,i,n)
    $t \leftarrow \tanh(\frac{2i}{n})$            ▷ threshold value depending on current epoch (0..1)
    $r \leftarrow random(|p|)$                    ▷ list of random numbers between 0 and 1.
    for $j \leftarrow 0; j < r; j++$ do
        if $r[j] > t$ then
            $p[j] = 1 - p[j]$                     ▷ Switch value if random value higher than threshold
        end if
    end for
    return p
end function
```

$$mt = \tanh\left(\frac{2i}{n}\right) \tag{17}$$

The whole procedure of creating a new population of chromosomes from a select parents is described in detail in Algorithm 4, where cs is the list of chromosomes, n is the number of parent chromosomes to keep in the next generation, e is the current epoch and max_e is the maximum epoch that could be reached. The `for` loop in Algorithm 4, continues until all the previous chromosomes (except the parents) are replaced by the newer ones. Algorithm 4 then returns the new population of chromosomes cs.

We conducted this experiment on each league separately, such that a total of five distinct training sets were considered. This was done so that we could compare the feature subsets produced by the different leagues to examine whether there were any attributes that were persistent across the leagues. The Genetic Algorithm was run ten times for every league, each time having an exclusive validation set for each run. On average the size of the validation set was between 80 and 100. This could be seen from Table 10, where when noticing the range of the instance set size is between 855 and 1095 for the draw matches at half-time. The training set for each run was the other remaining instances which were not included in the validation set. For each GA run in a league, a feature could either be present (1) or absent (0) in the final best feature subset produced. Then for every league, the mean number of times an attribute was chosen from the ten runs was recorded and compared between each other in a matrix similar to Table 11. In this table, the rows represent the whole list of features and the columns describe the league used in

Algorithm 4 NewGeneration

```
$cs$: chromosomes
$n$: number of parents to keep
$e$: current epoch
$max\_e$: maximum epoch
function NEWGENERATION($cs, n, e, max\_e$)
    $n\_children \leftarrow |cs| - n$
    $pr \leftarrow random.permutation(keep, size = n\_children)$        ▷ random permutation
    $j = 0$
    for $i \leftarrow n; i < n; i+ = 2$ do
        $id_1 \leftarrow pr[j]$
        $id_2 \leftarrow pr[j + 1]$
        $c_1, c_2 \leftarrow CrossOver(cs[id_1], cs[id_2])$
        $cs[n] \leftarrow Mutate(c_1, e, max\_e)$        ▷ mutation function on new child
        $cs[n + 1] \leftarrow Mutate(c_2, e, max\_e)$        ▷ mutation function on new child
        $j+ = 2$
    end for
    return $cs$
end function
```

the experiment. The cells' values could be between $\{0, 1\}$ indicating the percentage the features were included in the final best scoring feature subset produced by the GA of each run for that league. By using this matrix we were able to analyse the importance of certain features that were repeatedly being chosen by the genetic algorithm across the different leagues. We could also analyse the attributes that were present for specific leagues only, giving some insight that for different leagues, different features might be more indicative than others for predicting the end result of a game. The total number of times a feature appeared in the feature subset and the highest mean score are also recorded in the result matrix.

Like the previous experiments only the instances which were in a draw state (`goalDiff` = 0) at half-time were kept and the rest of the instances were filtered out. For each experiment conducted on the data set we used the same parameter settings for the genetic algorithm. These were a maximum epoch number of 200 and a total population of 50 chromosomes with a 20% parent retention ratio. For the mutation threshold parameter we used a function such that the mutation threshold increases with the epoch/generation number.

Algorithm 5 Genetic Algorithm

```
$C$: classifier Class to perform GA on
$E$: number of total generations to run algorithm for
$N$: population size
$X_t$: set of training input vectors
$Y_t$: set of training target vectors
$X_v$: set of validation input vectors
$Y_v$: set of validation target vectors
$G$: parameter grid
$th$: stopping threshold
$r$: parent retention ratio
$k$: number of internal cross-validation folds to perform
procedure GeneticAlgorithm($C$,$E$,$N$,$X_t$,$Y_t$,$X_v$,$Y_v$,$G$,$th$,$r$,$k$)
    $keep \leftarrow$ floor $|N| * r$
    $pop \leftarrow initialPopulation(N)$
    $best \leftarrow []$
    for $e \leftarrow 0; e< E; e{+}{+}$ do
        $res \leftarrow []$
        for $i,\ el \in pop$ do
            $fitness\_score,\ param \leftarrow GridSearchFitnessFunction(C, G, el, X_t, Y_t, k)$
            $res[i] \leftarrow (fitness\_score,\ param,\ el)$
        end for
        $res.sort(by = fitness\_score, ascending = False)$
        $fitness\_score,\ param,\ el \leftarrow res[0]$
        $c \leftarrow C(param)$
        $c.fit(X_t[:, el], Y_t)$
        $ext\_score \leftarrow c.score(X_v[:, el], Y_v)$
        $best[e] \leftarrow (\ ext\_score, fitness\_score,\ param,\ el)$
        if $e > 0$ & $(best[e].score - best[e-1].score) < th$ then
            break
        end if
        $pop \leftarrow NewGeneration(res.keys(), keep)$
    end for
    return $best[-1]$                    ▷ return last best chromosome and parameter
end procedure
```

Table 11: Genetic Algorithm results matrix. The columns of the matrix represent the league and the rows represent the features. The cells denote the percentages a feature was selected in the most performing subset promoted by the GA for that league, represented by the range 0..1.

	England	....	**Italy**	**Spain**	**France**	**Mean**
averageAgeDiff	0.70	....	0.80	0.10	0.90	**0.79**
shotTotalDiff	0.40	....	0.60	0.20	0.90	**0.45**
shotOnGoalDif	0.30	....	0.50	0.10	0.80	**0.32**
....	....	....	....	....	....	....
passTotalDiff	0.20	....	0.40	0.10	0.30	**0.21**
dribbleTotalDiff	0.10	....	0.30	0.10	0.10	**0.10**

3.3 Predicting Match Results Using Both Pre-match and Half-time Team Statistics

The data sets used in the previous section using half-time in-play statistics does not take into account any information about the teams aside from that match instance. In this section, we test whether related information about the teams in a particular season might be considered for providing better classification of the end results. For example, a team that is currently in good form and that has scored more goals than its opponent might be able to change the result from a draw to a win more often than a team with fewer goals scored and a bad form, even though both teams may have similar in-game statistics for a particular instance. In the in-game match data set, there is no way of differentiating between these two teams of similar in-play statistic vectors and both instances would be classified the same way even though one might have a better chance of winning the match due to its form and number of goals scored. Because of this, we decided to contextualise the half-time team statistics data set with aggregate pre-match data on a match-game basis for that season of the two opposing teams. We considered two types of attributes for pre-match statistics, the first three were the raw counts accumulated over the season. These were, accumulated points, total goals scored and total goals conceded. The other attributes we used were computed statistics to measure a team's overall ability in scoring and conceding goals compared with the rest of the teams. We called these attributes attack strength, described in Subsection 3.3.1 and defence strength, presented in Subsection 3.3.2. The final computed pre-match statistic considered was the form. We developed the form function, described in Subsection 3.3.3 to take into account the teams' results of the last six games and is designed to give more importance to the results of the most recent matches played. Finally, we show an example of both the collected and computed data of Barcelona for the 2015/16 season.

3.3.1 Attack Strength

The attack strength of a team is the ratio between the mean goals scored per game by that team and the mean goals scored per game by all the teams in the league. We present the attack strength function in Equation 18, where S is the matrix containing the goals scored per team for each match game, the columns represent the teams and the rows represent the matches. In the equation, t denotes the team which the attack strength is being calculated for, n denotes the maximum match game that will be considered and m represents the number of teams.

$$AttackStr(t, n, m) = \frac{\frac{1}{n}\sum\limits_{i=1}^{n} S_{it}}{\frac{1}{mn}\sum\limits_{j=1}^{m}\sum\limits_{i=1}^{n} S_{ij}} \tag{18}$$

3.3.2 Defence Strength

The defence strength of a team is calculated similarly to the one presented for the attacking strength in the previous section. Instead of goals scored, the defence strength considers the goals conceded. It is the ratio between the mean goals conceded per game by that team and the mean goals conceded per game by all the teams in the league. We present the defence strength function in Equation 19, where C is the matrix containing the goals conceded per team for each match game, the columns represent the teams and the rows represent the matches. In the equation, t denotes the team which the defence strength is being calculated for, n denotes the maximum match game that will be considered and m represents the number of teams.

$$Defence(t, n, m) = \frac{\frac{1}{n}\sum\limits_{i=1}^{n} C_{i}}{\frac{1}{mn}\sum\limits_{j=1}^{m}\sum\limits_{i=1}^{n} C_{ij}} \tag{19}$$

3.3.3 Team Form

The form equation measures the performance of the teams' last six matches in terms of their full-time result. The team is awarded one point for matches that it wins, 0.5 points if it draws and 0 points when it loses. We developed the form function such that the most recently played games are given the most importance. To achieve this, we use a decay function that assigns the score depending on how far the matches took place from the current one. The decay function used is shown in Figure 3.

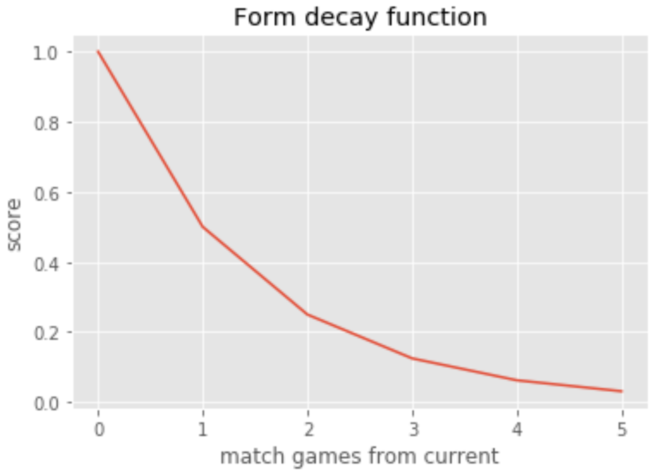


Figure 3: Decay function used for the form equation to give importance to the most recently played matches. The score given to a match game depends on how far it happened from the current one.

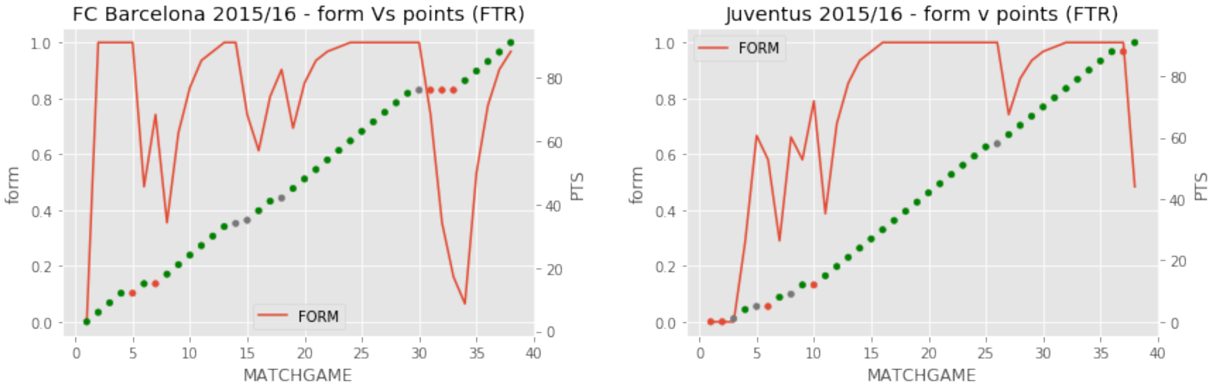
The full-time result score (L=0, D=0.5, W=1) given to each match are then multiplied by the score from the decay function for the same match game to produce the final score for each match. This means that a team losing (FTR score = 0) the most recent game (Decay score = 1) would be awarded 0 (0x1) form points for that match. The final scores of each match are then added together and divided by the best possible score (The sum of the decay function). This is done so that the form is scaled between 0 and 1. By using this method, two teams having the same full-time results for the last six matches but in a different order would produce different form values for each team. This is demonstrated in the example shown in Table 12. In this table, we show the computation of the form for team$_A$ and team$_B$ having the same number of wins, draws and losses, but in a different order. For team$_A$ the full-time result of the last six matches were L,D,W,W,D,W and for team$_B$ these were W,W,W,D,D,L. From the example we can see that the final scores for team$_A$ and team$_B$ were 0.69 and 1.84, respectively. To produce the final form measure, these are divided by the sum of the decay function, resulting in $\frac{0.69}{1.97} = 0.349$ for team$_A$ and $\frac{1.84}{1.97} = 0.937$ for team$_B$. This indicates that team$_B$ is going into the upcoming match in a better form than team$_A$. The Form function is presented in Equation 20, where t is the

Table 12: Form computation example for two separate teams, where i represents how long ago the match had happened in terms of match games. $(\frac{1}{2})^i$ represents the decay function, $FTR_{A/B}$ denotes the full-time results of the matches and their respective form points given as $y_{A/B}$. The computation shows the multiplication of the designated score for that game with the form points achieved by the team. We can note that because team$_A$ lost its most recent game, it received 0 of the designated points for that game.

i	$(\frac{1}{2})^i$	FTR$_A$	y$_A$	$(\frac{1}{2})^i$y$_A$	FTR$_B$	y$_B$	$(\frac{1}{2})^i$y$_B$
0	1.00	L	0.0	0.00	W	1.0	1.00
1	0.50	D	0.5	0.25	W	1.0	0.50
2	0.25	W	1.0	0.25	W	1.0	0.25
3	0.13	W	1.0	0.13	D	0.5	0.06
4	0.06	D	0.5	0.03	D	0.5	0.03
5	0.03	W	1.0	0.03	L	0.0	0.00
$\sum$	1.97			0.69			1.84

team which the form is being calculated for, j is the match game and i denotes the previous matches. y_{j-i} represents the form points (1, 0.5, 0) achieved in those games depending on their full-time result (W, D, L).

In Figure 4, we show the form over the 2015/16 season for the champions of the Spanish and Italian leagues in our data sets. In Sub-figure 4(a) we show the form of Barcelona over the match games in the season and compare it with their full-time results. One can notice that Barcelona had three dips in form over the whole season. The first happened when they lost the 5^{th} and 7^{th} games of the season resulting in a form of just over 0.3. Barcelona had another dip in form when they ended three of five matches in a draw, between the 14^{th} and 18^{th} match game. The worst drop in form they suffered was from the 29^{th} to 33^{rd} match game with one draw and three consecutive losses, resulting in a form just under 0.1. We can clearly see from the sub-figure that they recovered their form to 0.9, winning all the remaining games of the season. In Sub-figure 4(b) we show the form of Juventus over the 2015/16 season. From the sub-figure we can see that Juventus did not have a good start to the season with four loses and three draws in the first ten matches. However they recovered and won the rest of the matches except for a draw and a loss on the 26^{th} and 37^th match games respectively. We can also note the for the last game the form for Juventus was 0.5, because they had lost the previous game. The decay function can be adjusted such that the impact of the most recent full-time result would be reduced. This can be achieved by using a higher value as 0.8 instead of 0.5 for the decay function. For all our experiments the decay function used was set to 0.5.



(a) Barcelona 2015/16 Form

(b) Juventus 2015/16 Form

Figure 4: Form compared with points of the Spanish and Italian league champions for the 2015/16 season. The colour of the scatter points represents a win when coloured green, a draw when coloured grey and a loss when shown in red.

$$Form(t,j) = \frac{\sum\limits_{i=0}^{5} (\frac{1}{2})^i (y_{j-i,t})}{\sum\limits_{i=0}^{5} (\frac{1}{2})^i} \tag{20}$$

Finally, in Table 13 we show the collected and computed pre-match attributes described in the preceding sections of Barcelona FC, for each match game in the 2015/16 La Liga season.

3.3.4 Modified Nested Cross-Validation

For this experiment, we used the same model-tuning and genetic algorithm procedure described in the previous section. However we had to modify the nested cross-validation process used to estimate the performance of hyper-parameters and feature subsets by modifying the inner loop to account for seasonality. To accomplish this, we partitioned the data sets by the season and always trained the model on instances that happened before those used for validation. For each partition, the validation used in the previous fold would be added to the training set. The validation set would then consist of the instances from the following season. The process continues until no more sets remain. For each partition, the model is validated on the instances of the next season and the accuracy score for that partition would be recorded. For this experiment, we used five seasons between 2011 to 2016. By using the data partitioning method described above,

Table 13: Example of the pre-match data with the calculated form, attack strength and defence strength and the other attributed per match for the Barcelona team for the season 2015/16. The Team column represents the team, MG, DATE and OPP denote the match game number, the date of the match and the opposition, respectively. The HOME field represents whether the team was at home (1) or away (0). HTR, HTHO and HTAW represent the half-time results and goals scored for the home and away teams, respectively. FTR, FTHO and FTAW denote the full-time result and the scores of the home and away teams, similar to that of the half-time. GS, GC and PTS represent the accumulated goals scored, goals conceded and points over the season. ATT, DEF and FORM denote the attack strength, defence strength and form per match game. Y represents the points given per match result such that the form could be computed.

Team	MG	DATE	OPP	HOME	HTHO	HTAW	HTR	FTHO	FTAW	FTR	GS	GC	PTS	ATT	DEF	Y	FORM
FC Barcelona	1	23 Aug.	Athletic Bilbao	0	0	0	D	0	1	A	1	0	3	0.833	0.000	1	0.000
FC Barcelona	2	29 Aug.	Malaga	1	0	0	D	1	0	H	2	0	6	0.571	0.000	1	1.000
FC Barcelona	3	12 Sep.	Atletico Madrid	0	0	0	D	1	2	A	4	1	9	0.606	0.152	1	1.000
FC Barcelona	4	20 Sep.	Levante	1	0	0	D	4	1	H	8	2	12	0.860	0.215	1	1.000
FC Barcelona	5	23 Sep.	Celta Vigo	0	2	0	H	4	1	H	9	6	12	0.750	0.500	0	1.000
FC Barcelona	6	26 Sep.	Las Palmas	1	1	0	H	2	1	H	11	7	15	0.780	0.496	1	0.484
FC Barcelona	7	3 Oct.	FC Sevilla	0	0	0	D	2	1	H	12	9	15	0.732	0.549	0	0.742
FC Barcelona	8	17 Oct.	Rayo Vallecano	1	2	1	H	5	2	H	17	11	18	0.842	0.545	1	0.355
FC Barcelona	9	25 Oct.	Eibar	1	1	1	D	3	1	H	20	12	21	0.862	0.517	1	0.677
FC Barcelona	10	31 Oct.	Getafe	0	0	1	A	0	2	A	22	12	24	0.849	0.463	1	0.839
FC Barcelona	11	8 Nov.	Villarreal	1	0	0	D	3	0	H	25	12	27	0.865	0.415	1	0.935
FC Barcelona	12	21 Nov.	Real Madrid	0	0	2	A	0	4	A	29	12	30	0.932	0.386	1	0.968
FC Barcelona	13	28 Nov.	Real Sociedad	1	2	0	H	4	0	H	33	12	33	0.988	0.359	1	1.000
FC Barcelona	14	5 Dec.	Valencia	0	0	0	D	1	1	D	34	13	34	0.947	0.362	0.5	1.000
FC Barcelona	15	12 Dec.	Deportivo	1	1	0	H	2	2	D	36	15	35	0.945	0.394	0.5	0.742
FC Barcelona	16	17 Feb.	Sporting Gijon	0	1	2	A	1	3	A	39	16	38	0.949	0.389	1	0.613
FC Barcelona	17	30 Dec.	Real Betis	1	2	0	H	4	0	H	43	16	41	0.993	0.370	1	0.806
FC Barcelona	18	2 Jan.	Espanyol	0	0	0	D	0	0	D	43	16	42	0.933	0.347	0.5	0.903
FC Barcelona	19	9 Jan.	Granada	1	2	0	H	4	0	H	47	16	45	0.965	0.329	1	0.694
FC Barcelona	20	17 Jan.	Athletic Bilbao	1	2	0	H	6	0	H	53	16	48	1.004	0.303	1	0.855
FC Barcelona	21	23 Jan.	Malaga	0	1	1	D	1	2	A	55	17	51	0.973	0.301	1	0.935
FC Barcelona	22	30 Jan.	Atletico Madrid	1	2	1	H	2	1	H	57	18	54	0.960	0.303	1	0.968
FC Barcelona	23	7 Feb.	Levante	0	0	1	A	0	2	A	59	18	57	0.955	0.291	1	0.984
FC Barcelona	24	14 Feb.	Celta Vigo	1	1	1	D	6	1	H	65	19	60	0.998	0.292	1	1.000
FC Barcelona	25	20 Feb.	Las Palmas	0	1	2	A	1	2	A	67	20	63	0.993	0.296	1	1.000
FC Barcelona	26	28 Feb.	FC Sevilla	1	1	1	D	2	1	H	69	21	66	0.986	0.300	1	1.000
FC Barcelona	27	3 Mar.	Rayo Vallecano	0	0	2	A	1	5	A	74	22	69	1.012	0.301	1	1.000
FC Barcelona	28	6 Mar.	Eibar	0	0	2	A	0	4	A	78	22	72	1.020	0.288	1	1.000
FC Barcelona	29	12 Mar.	Getafe	1	4	0	H	6	0	H	84	22	75	1.058	0.277	1	1.000
FC Barcelona	30	20 Mar.	Villarreal	0	0	2	A	2	2	D	86	24	76	1.048	0.292	0.5	1.000
FC Barcelona	31	2 Apr.	Real Madrid	1	0	0	D	1	2	A	87	26	76	1.027	0.307	0	0.742
FC Barcelona	32	9 Apr.	Real Sociedad	0	1	0	H	1	0	H	87	27	76	1.002	0.311	0	0.355
FC Barcelona	33	17 Apr.	Valencia	1	0	2	A	1	2	A	88	29	76	0.982	0.324	0	0.161
FC Barcelona	34	20 Apr.	Deportivo	0	0	2	A	0	8	A	96	29	79	1.033	0.312	1	0.065
FC Barcelona	35	23 Apr.	Sporting Gijon	1	1	0	H	6	0	H	102	29	82	1.063	0.302	1	0.532
FC Barcelona	36	30 Apr.	Real Betis	0	0	0	D	0	2	A	104	29	85	1.058	0.295	1	0.774
FC Barcelona	37	8 May.	Espanyol	1	1	0	H	5	0	H	109	29	88	1.078	0.287	1	0.903
FC Barcelona	38	14 May.	Granada	0	0	2	A	0	3	A	112	29	91	1.074	0.278	1	0.968

	Split				
Data Set	1	2	3	4	
2011/12	t_1	t_2	t_3	t_4	
2012/13	v_1				
2013/14		v_2			
2014/15			v_3		
2015/16				v_4	
Score	S_1	S_2	S_3	S_4	$\bar{S}$

Figure 5: Schematic view of the inner loop partitioning and scoring mechanisim used for the data sets including the pre-match data, $t_{n \in N}$ denote the instances used for training and $v_{n \in N}$ described those used for validation. $S_{n \in N}$ denotes the score achieved for each partition and $\bar{S}$ is the mean of the scores.

we end up with five total train-validate folds. For the first partition, the instances of the 2010/11 season would be considered as training instances and those of the 2011/12 would be considered for validation. For the second partition the instances of the season 2011/12 would be added to the training set and the validation set would become that with the instances from the season 2012/13. This process continues until no data sets remain. The accuracy from each partition is recorded and the mean is taken such that the feature subset and the parameters used could be compared with the rest chromosomes and the other parameters. A schematic view of the process is described in Figure 5. By using this methodology the instances of the season 2010/11 are never considered for validation and the instances of the 2015/16 season are not considered for training.

3.4 Betfair Market Data

Market data at the half-time interval of the unseen samples was required such that we could compare the probabilities of the final model with the implied probability from the prices of the match winner market. We decided to get the samples from Betfair for two main reasons. The first being that Betfair is a prediction market where its participants are able to both back and lay outcomes of events. Therefore, Betfair is not interested in altering the market prices for profits and so the prices are purely moved by market forces.

The odds/prices were retrieved from Betfair[8] and stored in a mongodb database. The data set contains the price changes of the exchange markets on a minute interval. In order to find the half-time price of the match winner market, we ran the following steps;

- Retrieve the time the half-time winner market was settled for each match in the data set. By doing so we would know when the half-time interval of the event started.
- Find the full-time winner market of each event by using the match id.
- Get all the price changes for the full-time winner market of each match and order them by their published time.
- For each event, take the half-time winner market settlement time and use it to find the latest traded price of the outcomes (Home, Draw, Away) of the full-time winner market. This was done by taking the traded price of the outcomes which were at or just after the settlement time.

[8] http://historicdata.betfair.com

4 Results and Discussion

In this chapter we discuss the results that were achieved from the experiments run in this study. First, we look into the initial tests carried out on the *Football-data* data set and the results which led us to define the base-rule. We identify that there is a strong relationship between the half-time and full-time results. When taking into account all the instances from the different leagues, 38.0% of drawn matches at half-time end as a draw. We then carry out experiments with the chosen classifiers on a sample of the in-game data set from *Whoscored* and find that the random forest classifier had a consistent performance with the different predictors being given. The best accuracy of 52.0% on was achieved when the predictors, { `shotTotalDiff`, `cardRedDiff`, `cornerFavourDiff`, `assistShotIntentionalGoalDiff`, `crossFavourSuccessDiff`, `cornerFavourDiff`} were used. We then train the random forest with default parameters on all the samples of each league separately and compare the results achieved to the base-rule. The random forest had a slightly higher mean accuracy across the leagues (38.9%) than that of the base-rule. Because of the drop in accuracy of the random forest from previous experiment, we split the data set we carry out a time series test. In this experiment, the data sets are sorted in ascending order with respect to time (season) and the random forest is then trained on incrementing partitions of the data set and validated on the next one until no further seasons are left. The base-rule was also applied to the test and the results compared. With a mean accuracy of 38.4% (±4.0), the time series split did not show an improvement on the initial result. The use of a genetic algorithm is then applied to remove predictors from the feature set in a stochastic manner. From these experiments it resulted that on ten validation sets, the classification rate increased to a mean of 43.8% (±1.1) across the leagues. We then apply hyper-parameter tuning to the genetic algorithm, such that we find the best combination of feature and parameter subsets. From this experiment it resulted that the mean accuracy across the leagues increased to 45.0% (±1.6). We then introduce the pre-match data to the half-time in-play statistics and repeat the experiments. The mean accuracy was that of 46.0% (±2.1). This means that the addition of team data before the start of the match was able to improve on the classification rate but the change is not significant. Also, given that the accuracy did not increase with the experiments conducted may indicate that more research is needed in finding and collecting new statistics or making better use of the available ones. We evaluate the last model on an unseen data set and compare it with the default random forest and base-rule model using accuracy, and weighted measures of precision, recall and f-score to assess the overall performance.

Finally, we use a small random sample from Betfair market data to evaluate the predictive performance of the model in terms of probability against the odds of a prediction market. We use Brier score to measure the probabilistic accuracy of the market and the model and find that for the English league, the results were similar with results of 0.665 for the model and 0.622 for the market. A lower Brier score shows better predictability. This continues to show that predicting the final result at half-time for drawn matches is truly a hard problem which might be driven by the fact that football is a low scoring sport and a single goal can change the dynamic of the game in a dramatic manner. One mistake or brilliant action by an individual leading to a goal diminishes a team's ability to win the game with respect to its performance.

4.1 Half-time/Full-time Match Results

The first experiment that we carried out was on the football-data data set. From the available instances, we retrieved the match result at half-time and full-time of each event such that we could analyse how often the final result varied from that at half-time. We conducted this experiment for each league/season individually, so that we could see whether there are any specific trends relating only to particular leagues. The results were aggregated in confusion matrices such that the rows represent the state of the instances at half-time, and the columns represent that of the full-time result.

By analysing the confusion matrices it is noticeable that the majority of matches across all the leagues considered end in a home win. This can be noticed by seeing that for each matrix, the full-time result (FTR) home column has higher frequencies than the draw and away ones. As an example, we analyse the English Premier League half-time/full-time result confusion matrix for 2006/2007 season shown in Figure 6. From the matrix in Figure 6, we can see that without considering the state at half-time, 182 matches ended as a home win, accounting to 47.9% of the total matches of the season; by far exceeding the other end results, where 98 (25.8%) matches ended as a draw and 100 (26.3%) matches ended as an away win. We summarise the findings for all different leagues considered for the experiment in Table 14, showing the mean distribution of full-time results per season across ten years between 2006/07 and 2015/16. From Table 14, we can see that almost half of the matches for a given season of all the leagues end up in a home win.

The most interesting result from these experiments was to see that football matches have a tendency to finish at the full-time result in the state that they were in at the half-time interval. Meaning that when the home team is winning at half-time, for the majority

Table 14: Mean full-time result distribution per league over ten years. The highest occurring result is marked in bold. Standard deviation is shown in parenthesis.

League	**Home(%)**	**Draw(%)**	**Away(%)**
Belgium League	**0.471 (0.026)**	0.251 (0.015)	0.278 (0.032)
Bundesliga 1	**0.451 (0.023)**	0.246 (0.024)	0.302 (0.024)
Bundesliga 2	**0.447 (0.029)**	0.282 (0.028)	0.272 (0.024)
England Premier League	**0.460 (0.026)**	0.257 (0.024)	0.282 (0.029)
England Championship	**0.434 (0.023)**	0.279 (0.026)	0.285 (0.017)
England League 1	**0.436 (0.025)**	0.262 (0.017)	0.301 (0.026)
England League 2	**0.417 (0.026)**	0.268 (0.027)	0.316 (0.028)
English Conference	**0.438 (0.020)**	0.257 (0.014)	0.305 (0.019)
French Ligue 1	**0.449 (0.025)**	0.286 (0.030)	0.262 (0.023)
French Ligue 2	**0.453 (0.026)**	0.316 (0.015)	0.231 (0.019)
Greece Super League	**0.476 (0.039)**	0.244 (0.027)	0.255 (0.029)
Italy Serie A	**0.465 (0.028)**	0.271 (0.025)	0.262 (0.022)
Italy Serie B	**0.435 (0.020)**	0.312 (0.018)	0.249 (0.027)
Netherlands Eredivise	**0.479 (0.028)**	0.235 (0.020)	0.284 (0.023)
Portugal Liga 1	**0.442 (0.018)**	0.267 (0.025)	0.291 (0.024)
Scotland Premier League	**0.427 (0.034)**	0.241 (0.038)	0.331 (0.030)
Scotland Division 1	**0.427 (0.034)**	0.255 (0.032)	0.319 (0.035)
Scotland Division 2	**0.436 (0.025)**	0.216 (0.029)	0.347 (0.020)
Scotland Division 3	**0.450 (0.019)**	0.208 (0.032)	0.342 (0.022)
Spain La Liga Primera	**0.483 (0.023)**	0.235 (0.016)	0.282 (0.021)
Spain La Liga Segunda	**0.458 (0.020)**	0.288 (0.030)	0.254 (0.016)
Turkey Futbol Ligi 1	**0.461 (0.020**)	0.261 (0.021)	0.273 (0.019)

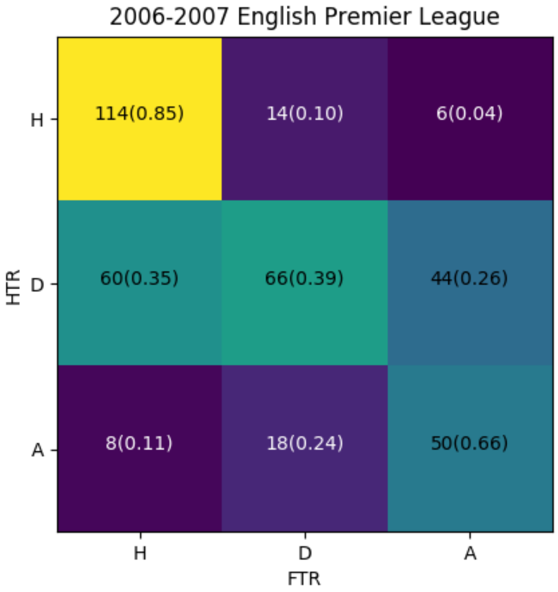


Figure 6: English Premier League 2006/2007 Half-time/Full-time Result Confusion Matrix. The values in each cell denote the total number of times the transition occurred in the given season. The numbers in parenthesis denote the probability of the match ending in that state at full-time given the present state at half-time.

of times the match ends in a home win. On the other hand, when the away team is winning at half-time, the game has a higher chance of ending as an away win. In Figure 7, we show the half-time/full-time result matrices of the top tier European leagues for the 2015/2016 season. The main diagonal line (from left to right) comprising of the cells `H/H`, `D/D` and `A/A` represent the states for when the full-time result ends the same as that of the half-time. From the result matrices it could be seen that these cells are brighter, reflecting the higher frequency this transition occurred during the season. For the instances where the half-time result is draw (`D`), this tendency is less evident and in this state, the full-time results seem more evenly distributed across the different end states. This behaviour is not specific only to the top tier leagues but it is also evident in lower tier ones.

Figure 8 shows the same matrices of the 2015/2016 season for six lower tier European leagues. By analysing these examples from Figure 8 it is also evident that for each state at the half-time interval, the most common transition from that particular state to the full-time result is to the one the match is already in at the half-time period. This can also be seen graphically from the matrices in Figure 8 where the main diagonal line of the matrices are also the brightest cells of the matrix, denoting a higher frequency. Table 15 shows the mean distribution of the half-time/full-time transitions per league over ten seasons. From the table, it can be seen that the highest occurring transition for all of the leagues is HTR=`H`/FTR=`H`. This observation is summarized in Figure 9, where the mean distribution of the full-time results of all of the leagues, differentiated by the half-time result are displayed. The transition with the highest mean distribution (26.7%) is that of HTR=`H`/FTR=`H`. The second transition is HTR=`A`/FTR=`A` (16.2%), followed closely by the HTR=`D`/FTR=`D` transition with mean of 16.15%. This result is consistent with the previous observations, displaying the tendency for matches to end in the same state as that of the half-time, irrelevant of the leagues and its deemed quality. Furthermore to these results, we carried out a final experiment by aggregating the instances from the different leagues into one set to see the marginal distribution of the half-time result when considering all the leagues together. Figure 10 shows the result of the marginal distribution of the half-time result. We can see that of all the instances, for when the home team is winning at half-time, 79% of the matches end in home team wins, 15% draw, 6% away wins. Alternatively, when the away team is winning at half-time, 68% of the matches end in away wins, 21% draw and 11% home wins. Matches in a draw at half-time also have a higher probability of ending in the same state, however, the results are more evenly distributed than the other two half-time states; with 38% of these matches ending in a draw, 37% in home wins and 25% in away wins.

From these observations we derived a base rule such that we could compare results of future experiments with and also made a decision that we would only consider matches in a draw state at half-time for the future experiments. This is because when considering the goal difference between the teams at half-time for non-drawn matches, this attribute would be enough to classify the full-time result on its own without the need to consider other attributes. So we decided to focus the study on finding discriminating attributes of teams on equal goals at the half-time interval. The base rule derived from the football-data data set and the half-time/full-time experiments is that of using the same half-time result to predict that of the full-time. By only considering drawn states at half-time, the base-rule prediction would have a 38.0% accuracy rate for predicting the full-time result as a draw when the half-time result is also a draw.

4.2 Manual Feature Selection Experiments

After having derived the base-rule and having constructed the half-time team performance statistics data set, we continued to carry out further experiments to try and find important

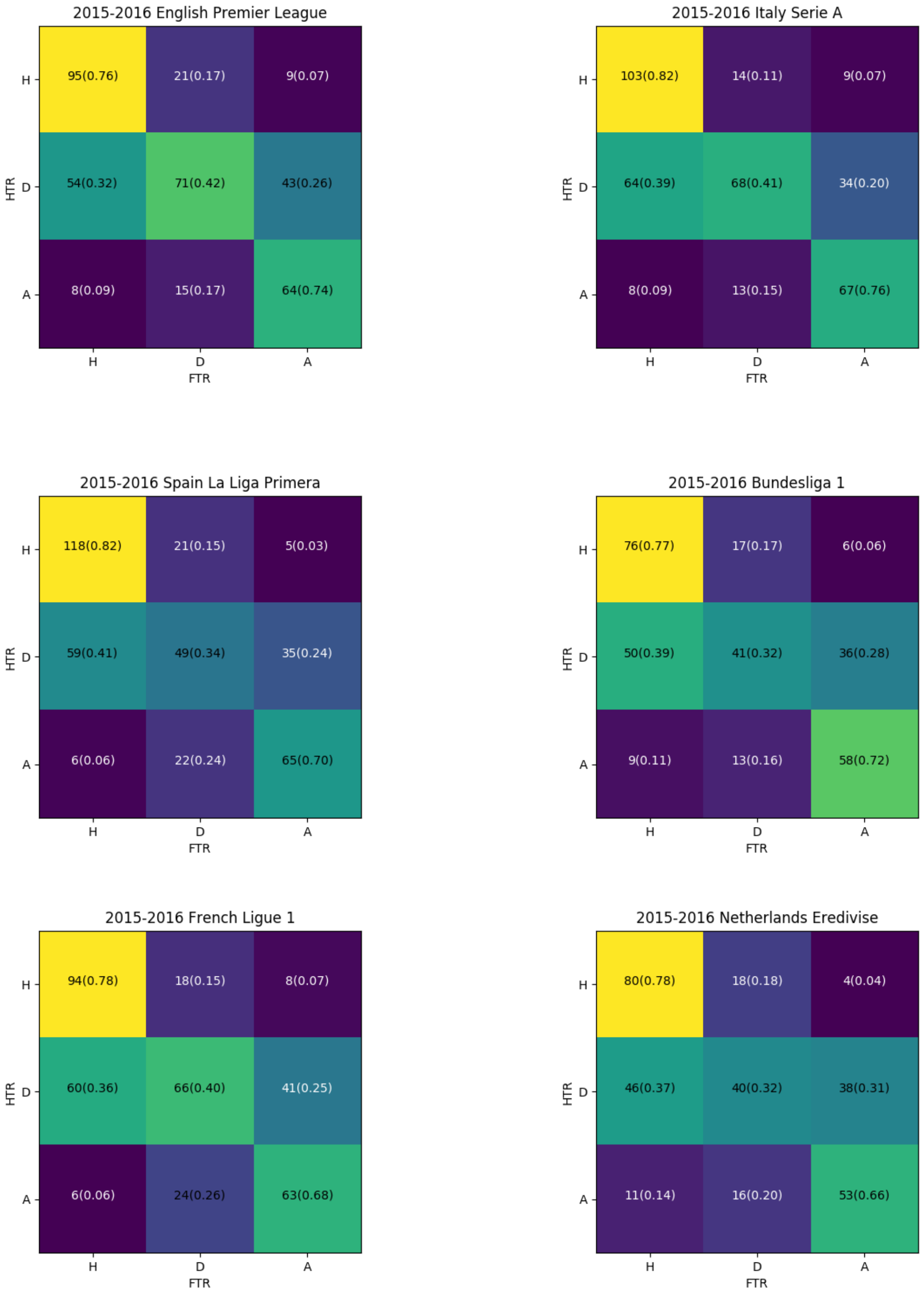


Figure 7: Top tier leagues half-time/full-time transition matrices for the 2015/16 Season

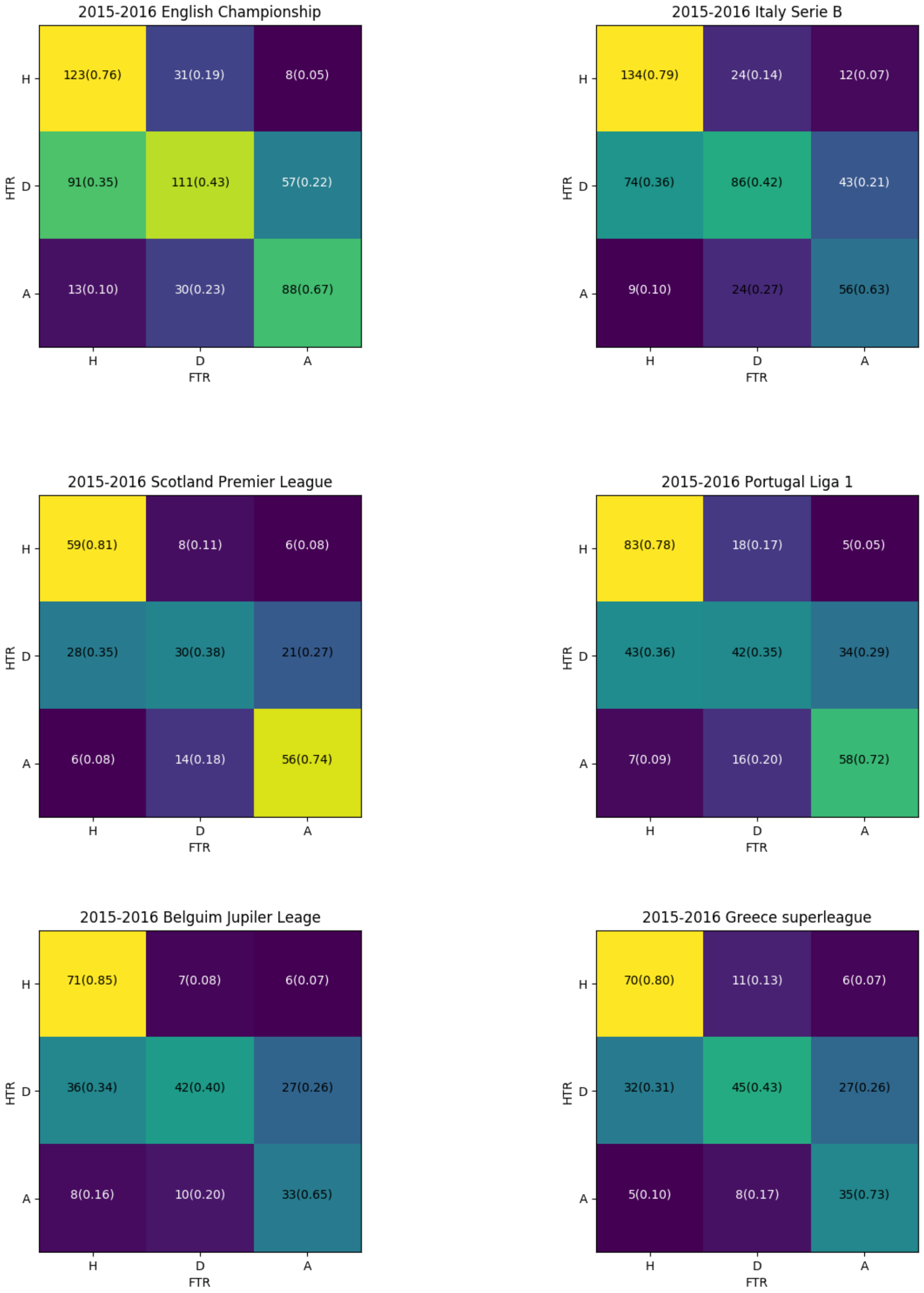


Figure 8: Low tier European leagues half-time/full-time transition matrices for the 2015/16 Season

Table 15: Mean half-time/full-time result transitions distribution per league over a ten year period (2006/07 - 2015/16). The highest occurring transition is marked in bold.

League	**HH**	**HD**	**HA**	**DH**	**DD**	**DA**	**AH**	**AD**	**AA**
Belgium League	**0.278**	0.039	0.023	0.169	0.162	0.102	0.026	0.047	0.151
Bundesliga 1	**0.277**	0.050	0.024	0.152	0.146	0.105	0.025	0.052	0.172
Bundesliga 2	**0.260**	0.054	0.017	0.158	0.171	0.099	0.029	0.057	0.157
England Premier League	**0.285**	0.048	0.017	0.153	0.162	0.104	0.021	0.048	0.159
England Championship	**0.253**	0.056	0.019	0.154	0.168	0.106	0.028	0.054	0.159
England League 1	**0.260**	0.052	0.019	0.147	0.151	0.115	0.03	0.058	0.169
England League 2	**0.244**	0.053	0.023	0.147	0.166	0.116	0.024	0.048	0.176
English Conference	**0.257**	0.057	0.019	0.150	0.15	0.115	0.028	0.048	0.172
French Ligue 1	**0.269**	0.056	0.019	0.161	0.184	0.105	0.022	0.047	0.141
French Ligue 2	**0.259**	0.053	0.015	0.165	0.202	0.093	0.027	0.061	0.124
Greece Super League	**0.275**	0.038	0.013	0.177	0.173	0.098	0.023	0.037	0.140
Italy Serie A	**0.276**	0.052	0.018	0.162	0.168	0.108	0.025	0.052	0.139
Italy Serie B	**0.256**	0.054	0.017	0.155	0.202	0.101	0.025	0.055	0.136
Netherlands Eredivise	**0.298**	0.053	0.023	0.150	0.133	0.105	0.031	0.049	0.154
Portugal Liga 1	**0.261**	0.045	0.019	0.159	0.171	0.104	0.023	0.051	0.168
Scotland Premier League	**0.267**	0.051	0.022	0.142	0.148	0.122	0.019	0.042	0.188
Scotland Division 1	**0.252**	0.050	0.022	0.149	0.157	0.102	0.025	0.046	0.199
Scotland Division 2	**0.254**	0.039	0.026	0.155	0.128	0.104	0.029	0.047	0.216
Scotland Division 3	**0.272**	0.043	0.028	0.150	0.115	0.111	0.028	0.049	0.206
Spain La Liga Primera	**0.295**	0.046	0.017	0.164	0.148	0.103	0.026	0.040	0.160
Spain La Liga Segunda	**0.265**	0.058	0.018	0.170	0.186	0.100	0.025	0.044	0.138
Turkey Futbol Ligi 1	**0.259**	0.058	0.019	0.173	0.163	0.109	0.031	0.042	0.142

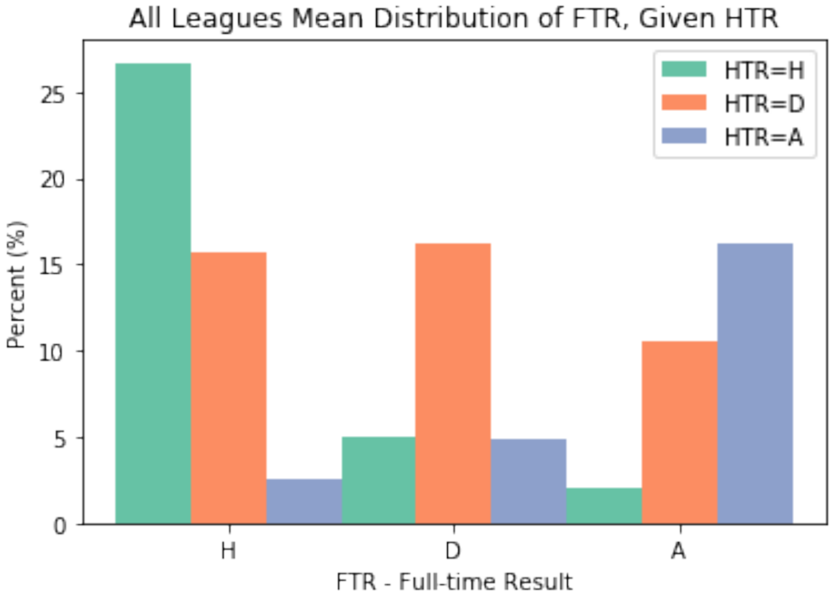


Figure 9: Mean distribution of full-time results differentiated by the half-time result, considering all instances of the different leagues.

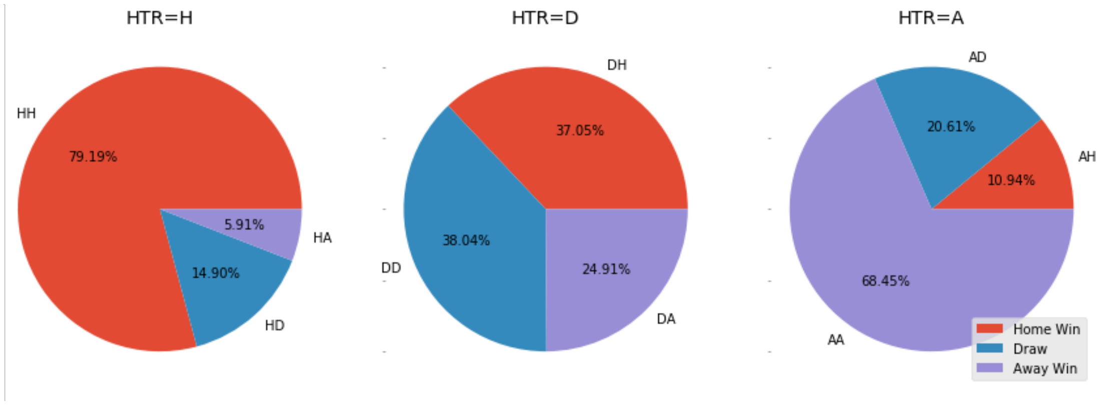
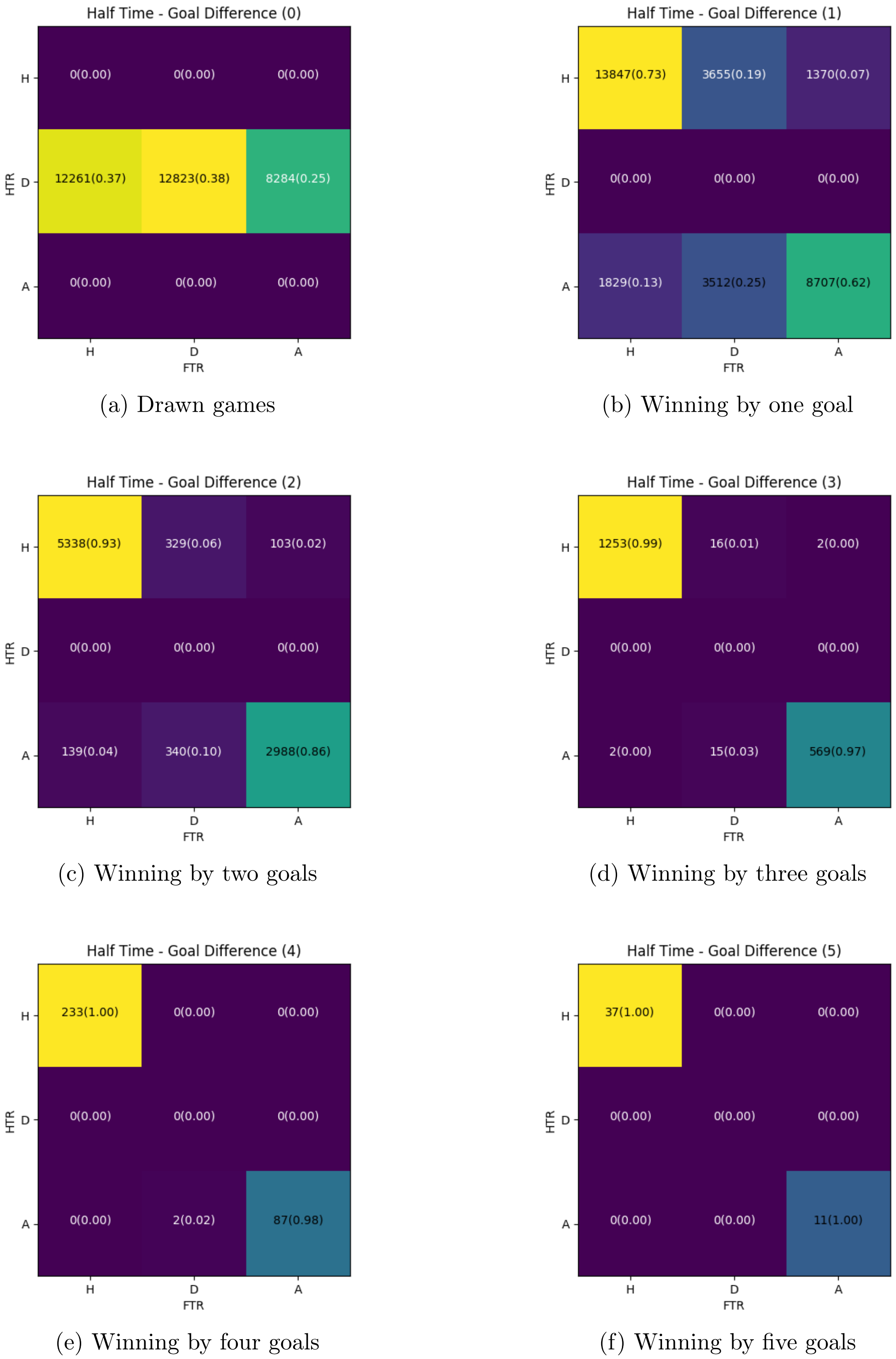


Figure 10: Probability of full-time result given the half-time result.



(a) Drawn games

(b) Winning by one goal

(c) Winning by two goals

(d) Winning by three goals

(e) Winning by four goals

(f) Winning by five goals

Figure 11: Half-time/Full-time transition matrices grouped by goal difference at halftime

in-play attributes that are informative at discriminating between the different full-time results. The first experiment we carried on this data set was that of manually selecting the features by using our knowledge in the game of football and also by using results from previous literature to try to find a good performing subset of features that are good at accurately predicting the full-time result. We first started out by using only one attribute from the feature set and used that alone as input to the classifiers to predict the outcome of the game. For each classifier used, tests were carried out with and without scaling of data. The types of classifiers used for the tests were Random Forest (RF), Decision Tree (DT), Naive Bayes (NB) and Neural Networks (NN). For these experiments the 2015/16 English Premier League data set was used.

From the experiment it resulted that the `shotTotalDiff` was the best performing attribute having achieved an classification accuracy of 47.6%. Table 16 shows the results achieved by each individual attribute and the classifier and data scaling used. With having seen that this was a good informative attribute which is given much importance in the footballing community, we decided to keep it such that for each of the subsets made up of one attribute used in the experiment we added the `shotTotalDiff` attribute to each subset. We re-ran the experiment with the new feature sets and the best classification accuracy was achieved by the {`shotTotalDiff`,`cardRedDiff`} subset scoring an accuracy of 48.0% this time with the neural net classifier and using scaled data. We continued this process until we arrived at a point where the addition of new features to the subset did not produce better results from the previous one with the fewer features. The final subset created from the manual selection includes the attributes { `shotTotalDiff`, `cardRedDiff`, `cornerFavourDiff`, `assistShotIntentionalGoalDiff`, `crossFavourSuccessDiff`, `cornerFavourDiff`} with classification accuracy of 52.0% with raw data and using the random forest classifier. We tested out other attributes that we thought were good at indicating whether a team was having a good performance in a game and so we started off the subsets with these features even though they achieved a lower score than the first tested attribute (`shotTotalDiff`) on their own. Our thought was that with the combination of other attributes, these features could achieve a higher classification rate. The same process of creating larger feature sets by adding the selected feature to the rest of the features was used as in the first experiment. The attributes tested were `possessionAttackDiff` and `passTargetFinalDiff`. The bigger the difference in these attributes for two teams imply how a team dominated more of the ball possession in the other team's half and close to the opposition's goal. The final subset constructed from when the `passTargetFinalThirdDiff` attribute was considered was {

passTargetFinalThirdDiff, possessionAttackDiff, tackleSuccessDiff } with an accuracy score of 49.4% and using the Naive Bayes classifier with raw (unscaled) data. When considering first the possessionAttackDiff as the first attribute, the final feature set constituted of the following attributes; possessionAttackDiff, possessionDefenceDiff, passForwardDiff with accuracy score of 51.2% and using the Naive Bayes classifier with raw (unscaled) data. For all the tests done in this experiment, the process was stopped as soon as the new features added to the subset did not improve the accuracy score of the current set. All the scores achieved from the three tests are in the same range and none of the final feature sets achieved a far greater result than any other.

Table 16: Results of manually selected feature set containing only one attribute. Test carried out on the 2015/2016 English Premier League data. Best performing attribute was the `shotTotalDiff` with the Random Forest Classifier. `S` denotes data was scaled and `R` denotes data was left in raw form (without scaling).

Feature	RF(R)	RF(S)	DT(R)	DT(S)	NB(R)	NB(S)	NN(R)	NN(S)
averageAgeDiff	0.292	0.292	0.274	0.280	0.411	0.411	0.339	0.357
shotTotalDiff	**0.476**	**0.476**	0.470	0.470	0.440	0.440	0.381	0.423
shotOnGoalDiff	0.364	0.370	0.328	0.328	0.387	0.387	0.363	0.387
passTotalDiff	0.375	0.370	0.381	0.375	0.446	0.446	0.315	0.405
passLongDiff	0.352	0.352	0.346	0.346	0.423	0.423	0.399	0.423
passSuccessDiff	0.345	0.356	0.351	0.363	0.447	0.447	0.358	0.411
passForwardDiff	0.411	0.423	0.417	0.423	0.446	0.446	0.351	0.423
passBackwardDiff	0.386	0.374	0.404	0.392	0.411	0.411	0.357	0.387
passTargetFinalThirdDiff	0.400	0.394	0.340	0.346	0.453	0.453	0.304	0.458
passTargetMiddleThirdDiff	0.310	0.304	0.321	0.321	0.422	0.422	0.382	0.387
passTargetDefensiveThirdDiff	0.328	0.328	0.327	0.327	0.369	0.369	0.381	0.410
cornerFavourDiff	0.441	0.441	0.423	0.423	0.423	0.423	0.423	0.435
foulReceivedDiff	0.429	0.429	0.429	0.429	0.411	0.411	0.417	0.417
assistShotDiff	0.405	0.405	0.429	0.429	0.417	0.417	0.376	0.411
assistShotIntentionalDiff	0.429	0.429	0.441	0.441	0.387	0.387	0.447	0.417
assistShotIntentionalGoalDiff	0.411	0.411	0.399	0.399	0.351	0.351	0.411	0.411
crossFavourTotalDiff	0.322	0.322	0.381	0.381	0.364	0.364	0.364	0.358
crossFavourSuccessDiff	0.375	0.375	0.399	0.399	0.386	0.386	0.381	0.363
offsideCommittedDiff	0.422	0.422	0.375	0.369	0.405	0.405	0.405	0.405
possessionAttackDiff	0.321	0.321	0.357	0.357	0.447	0.447	0.327	0.458
possessionTotalDiff	0.309	0.309	0.298	0.298	0.453	0.453	0.286	0.429
possessionDefenceDiff	0.386	0.386	0.416	0.416	0.423	0.423	0.363	0.392
interceptionDiff	0.327	0.327	0.333	0.333	0.429	0.429	0.333	0.363
cardYellowDiff	0.375	0.375	0.375	0.375	0.399	0.399	0.405	0.405
cardRedDiff	0.447	0.447	0.447	0.447	0.435	0.435	0.447	0.447
tackleTotalDiff	0.334	0.334	0.351	0.352	0.405	0.405	0.399	0.399
tackleSuccessDiff	0.435	0.435	0.446	0.446	0.429	0.429	0.351	0.423
dribbleTotalDiff	0.321	0.321	0.328	0.328	0.387	0.387	0.423	0.411
dribbleSuccessDiff	0.351	0.351	0.387	0.387	0.410	0.410	0.386	0.410

4.3 Initial Tests with Random Forest Default Parameters Compared with Base-Rule Predictions

After initial experimentation with manual feature selection, we tested the performance of the random forest on each of the leagues with the default hyper-parameters provided by *Scikit-Learn*. The default parameters for the random forest were the following; ten estimators, gini index as purity criterion, two minimum samples required for splitting a node and one sample required for creating a node leaf. The data sets were composed of all the drawn matches at half-time of the English, Italian, Spanish, German and French major leagues for the seasons between 2009/10 - 2015/16. The difference between the home team's and away team's in-game match statistics were the only predictors considered as part of the feature set. The construction of these data sets are detailed in Section 3.2.2. We re-sampled the data sets using stratified ten-fold cross validation to have an accurate estimation of the models' performance. We computed the performance of the base-rule for the same instances used to train the random forest. In Figure 12, we display the base-rule predictions as confusion matrices for the same leagues and seasons used by the random forest. These matrices are used to derive the accuracy of the predictions made by the base-rule. In the figure, we show all the possible predictions of the base-rule and not only those which are drawn at half-time. This shows that the base-rule is also valid for the period being considered in this experiment. The base-rule has a very high prediction rate for when the home team or away team are winning at half-time. When considering all the leagues together, the base-rule has an accuracy rate of more than 77.0% for home team wins and above 68.0% for away team wins. When the game is drawn at half-time, the performance of the base-rule drops as the full-time results are more evenly distributed across the end states. By following the base-rule for the drawn games at half-time, the match outcome will be predicted as draw. The accuracy of these predictions can also be retrieved from the matrices in Figure 12. The results of the base-rule per league are as follows; 39.5% for the French Ligue 1, 39.1% for both the English Premier League and Italian Serie A, 36.7% for the Spanish La Liga and 36.2% for the German Bundesliga. The following are the accuracy scores in descending order, that were achieved from the 10-fold cross validation of the random forest with default parameters. The highest score was recorded on the Spanish data set with an accuracy of 41.8%, followed by the French, where a mean of 39.4% of the instances were classified correctly. The model trained on the Premier League had an accuracy rate of 39.0%. The random forest achieved the same score (37.2%) for the Italian and German data sets. The results of the base-rule and random forest are summarised in Table 17. On average the random forest performed slightly better than the base-rule, however the difference is marginal. The random forest did not perform as well as expected with the mean score of all the leagues just 5.9% (33%) above that of random guessing and 0.009% above the performance of a very simple base-rule. The assumption made was that a team's current state in a game, depicted by their in-play statistics could be enough to discriminate between the tied competitors. More contextual data about the competitors might be needed in order to improve the discriminatory ability of the model. Apart from this, there are other factors that could be negatively affecting the performance of the random forest classifier. These factors could be the parameters being used or some non-informative predictors in the feature space. These issues are tackled in the experiments carried out in the following sections.

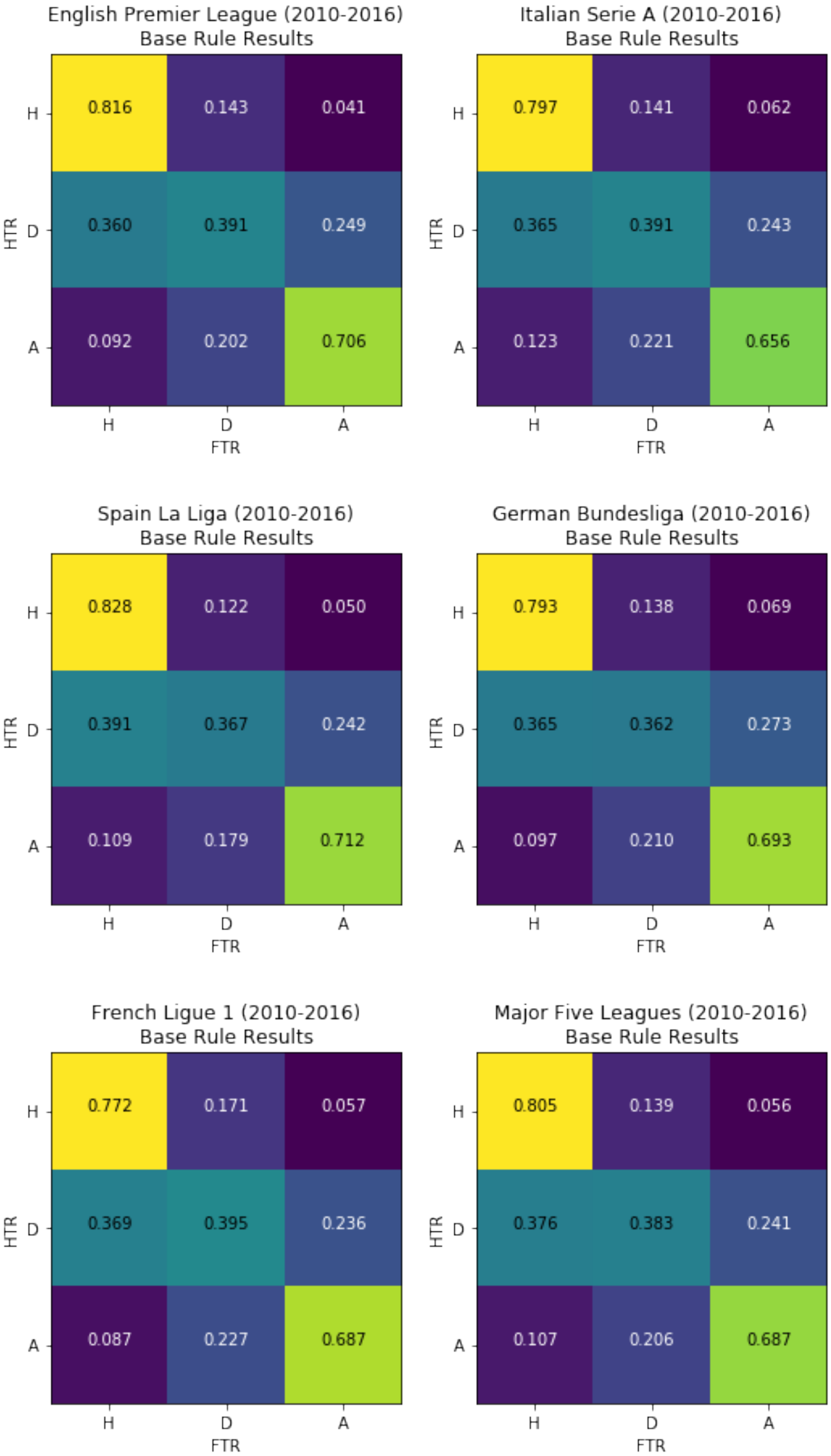
Table 17: 10-fold cross-validation on Random Forest with default parameters compared to Base Rule.

League	**Random Forest**	**Base Rule**
English Premier League	0.390	**0.391**
Italian Serie A	0.372	**0.391**
Spanish La Liga	**0.418**	0.366
German Bundesliga	**0.372**	0.362
French Ligue 1	0.392	**0.394**
Mean	**0.389**	0.380

4.4 Time Series Split Experiments

Another issue which we looked into was that for the first experiment we did not consider the temporal characteristic of the instances for the ten-fold cross validation. And since we are using seven seasons for each league (six for Italy), it might be the case that predicting instances that happened closer (with respect to time) to the training data could increase the classification rate of the model. It is important to note the feature set does not contain in-between game data, however, it might be the case that there would be latent relationships between the independent samples.

As with the previous experiment, this test was conducted separately for each league. The samples were sorted in ascending order by their season. Meaning that the instances of the 2010/11 season were situated at the top and those of the 2015/16 season were placed



English Premier League (2010-2016)
Base Rule Results

Italian Serie A (2010-2016)
Base Rule Results

Spain La Liga (2010-2016)
Base Rule Results

German Bundesliga (2010-2016)
Base Rule Results

French Ligue 1 (2010-2016)
Base Rule Results

Major Five Leagues (2010-2016)
Base Rule Results

Figure 12: Half-time/Full-time base-rule predictions for the five major European leagues for the seasons 2009/10 - 2015/16

last. We then split the data set to have an equal number of partitions to the number of seasons present in the data set. This means that the model is trained on the instances of a number seasons and is then validated on the upcoming one in a retrospective procedure. The set used for validation is then added to the training set and the model is trained again on the joined set. The next partition in the data set becomes the validation set. This process repeats until all partitions have been used for validation. The first partition is never used for validation and the last partition is not used for training. In our case, each partition contains the instances of a particular season. For this experiment, the default parameters of the random forest were kept the same as the initial experiments. Apart from the accuracy of the random forest we also wanted to measure the performance of the base-rule on each validation set for comparisons. The results of the experiments for each league can be seen in Figure 13. In the figure, the top part of the subplots shows the accuracy achieved by the random forest and the base-rule for the same validation sets. The bottom half of the subplots displays the ratio of the training samples to validation samples used in each split. One would expect to see that the accuracy of the random forest would increase as more training samples are added to the model. However, by analysing the graphs of each season, one could see that the scores seem to oscillate around 40.0%, for both the random forest and the base-rule.

In total, across all the leagues, 34 validation sets were used. As a result, a mean accuracy of 0.384 (±0.040) was achieved by the random forest and 0.371 (±0.041) by the base-rule. We used a one-tailed paired t-test to check whether the random forest had performed significantly better than the base-rule. The scores were checked for normality using skew test, kurtosis test and finally a normal test. This was done since, the t-test requires that the samples used come from a normal distribution. The results indicated that scores come from a normally distributed sample and so we proceeded to draw our null and alternative hypothesis. The results from the normality tests for the random forest and base-rule scores can be seen in Table 18. Given the p-values from the tests were greater than 0.05, this demonstrated that the scores are much more likely to be coming from a normal distribution than not, and thus we fail to reject that the samples are normally distributed.

We use one-tailed t-test because for the value to be significant for our test we want it to be in a specific direction. That being to the right of the mean differences between the scores achieved on the same validation set by the random forest and the base rule. Otherwise, the accuracy of the random forest might be worse and it would still be accepted as significant,

Table 18: P-values from normality tests conducted on the Random Forest and base-rule scores

Tests	**Random Forest p-values**	**Base Rule p-values**
Skew test	0.610	0.580
Kurtosis test	0.130	0.616
Normal test	0.285	0.75

which is not the test we want to conduct. Also, we use a paired t-test because the same samples are being used by the separate models. Because of this, we test on the difference of the scores. Considering these issues, the null hypothesis was set as the following; the difference in scores of the base-rule from the random forest are less than or equal to 0. Which is described in Equation 21, where RF_{acc} is the accuracy of the random forest and BR_{acc} is the accuracy of the base-rule. The alternate hypothesis is that the difference in scores of the base-rule from the random forest is greater than 0. Equation 22 describes the alternate hypothesis.

$$h_0 = RF_{acc} - BR_{acc} <= 0 \tag{21}$$

$$h_a = RF_{acc} - BR_{acc} > 0 \tag{22}$$

The results from the paired t-test showed that with a positive t-statistic greater than 0, of 1.33, the random forest did perform slightly better than the base-rule but with a p-value of 0.19, it was not significant. Thus, we fail to reject the null hypothesis that the difference between the random forest and base-rule scores is less than or equal to 0.

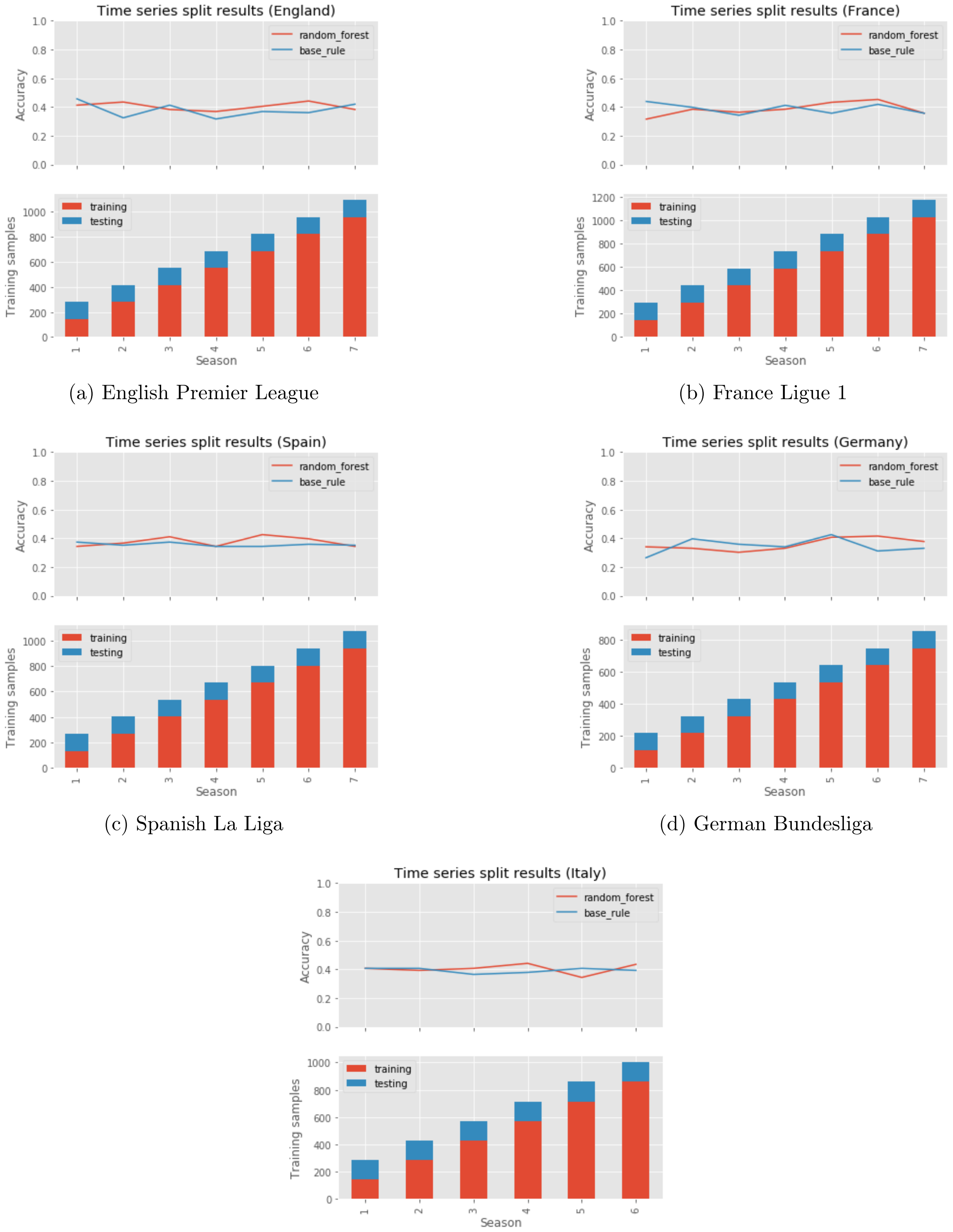
We then compare the results of the time series splits with those achieved by the random forest trained by cross validation. The random forest trained in time series registered a higher accuracy for the Premier League (40.3%) and the Serie A leagues (40.4%). However, it did not perform as well on the rest of the leagues, showing a lower accuracy for each one when compared to the cross validated one.

Table 19: Time series split results for Random Forest and base-rule models compared with previous results of the 10-fold CV Random Forest. $_{TS}$ denotes the scores from the time series test. $_{10CV}$ denotes the results from 10 fold Cross Validation. Numbers in parenthesis represent the standard deviation. The highest accuracy across the leagues is marked in bold.

League	**Random Forest**$_{10CV}$	**Random Forest**$_{TS}$	**Base Rule**$_{TS}$
English Premier League	0.390	**0.403(±.028)**	0.379(±.052)
Italian Serie A	0.372	**0.404(±.035)**	0.392(±.018)
Spanish La Liga	**0.418**	0.375(±.035)	0.356(±.013)
German Bundesliga	**0.372**	0.357(±.043)	0.346(±.053)
French Ligue 1	**0.392**	0.384(±.046)	0.388(±.037)
Mean	**0.389**	0.384(±.040)	0.371(±.041)

4.5 Feature Selection using Genetic Algorithm on Random Forest with Default Parameters

As we described before, one problem that may affect a model's performance could be caused by non-informative predictors in the feature space. The random forest classifier performs its own feature selection by the use of bootstrapping as described in Section 2.4.4. When using bootstrapping, each classification tree in the random forest is built independently from the others by replacing the original training samples with ones randomly selected from the training set. Each training set would end up having a number of unique samples and repeated samples. This results in each tree having its own training set and thus each ending up as different models. Despite this, we use a genetic algorithm to reduce the initial feature space allowed for the random forest. The mechanisms and inner workings of the genetic algorithm are described in Section 2.5.5. We run this experiment to see if the classification accuracy of the random forest could be improved by omitting certain attributes from the feature set. The genetic algorithm removes the predictors in a stochastic manner and keeps those that maximise the objective score. Because of this, the random forest would now be able to consider predictors that would otherwise have not been picked because of the presence of the other predictors. We use nested cross-validation of ten outer folds, with the genetic algorithm run on each fold. Every chromosome within the outer fold, is then evaluated using an inner cross validation of ten folds on the training set created by the outer fold. The use of nested cross-validation helps us understand if over-fitting



(a) English Premier League

(b) France Ligue 1

(c) Spanish La Liga

(d) German Bundesliga

(e) Italian Serie A

Figure 13: Time series split results per league showing the accuracy of the base-rule and the Random Forest for each split (top) along with the train/validate data sets in terms of number of samples (bottom).

is occurring. The genetic algorithm overfits to the objective it is trying to maximise, in our case this is the inner cross-validation score. By testing the best chromosome for a generation on the partition left out by the outer fold we are able to evaluate if the model built with the chromosome is able to still generalise for unseen data. By using this procedure, the performance of the model is not over estimated. For this experiment we keep the default parameters of the random forest as done with the previous experiments so far. Figure 14 shows the mean internal and external scores of the ten folds per generation for each league. In all the charts we can see that mean internal score is monotonically increasing with every generation. This behaviour is expected, as the genetic algorithm is selecting the chromosomes that maximise the fitness function (internal score). However, we can see that the external mean score is not always correlated with the internal score. Subplots 14(a) and 14(d), for the English and German league respectively show that the mean external score was achieved in the first generations. Even though the internal scores were being optimised with each generation, the models were not generalising as well for the unseen data. For the Spanish league, shown in Subplot 14(c), the mean external score dropped after the second generation, but continued increasing with each generation until it surpassed the initial best performing external score. The external mean score for the Italian league shown in Subplot 14(b) and the French league Subplot 14(e), were the highest at the 50^{th} and the 84^{th} generation, respectively. For this experiment we also joined the instances of the different leagues together and performed the same procedure on it. The mean external score for this data set was the highest at the 79^{th} generation. The results of the experiment are displayed in Subplot 14(f). The generations at which the models achieved the highest mean external score are used as a stopping criteria for the final genetic algorithm to be used for the training of the model before evaluation. When training the final model, we can not know when the genetic algorithm starts to overfit (since we will be using all the samples available). So this experiment serves us as a guide to how long we should keep the genetic algorithm running.

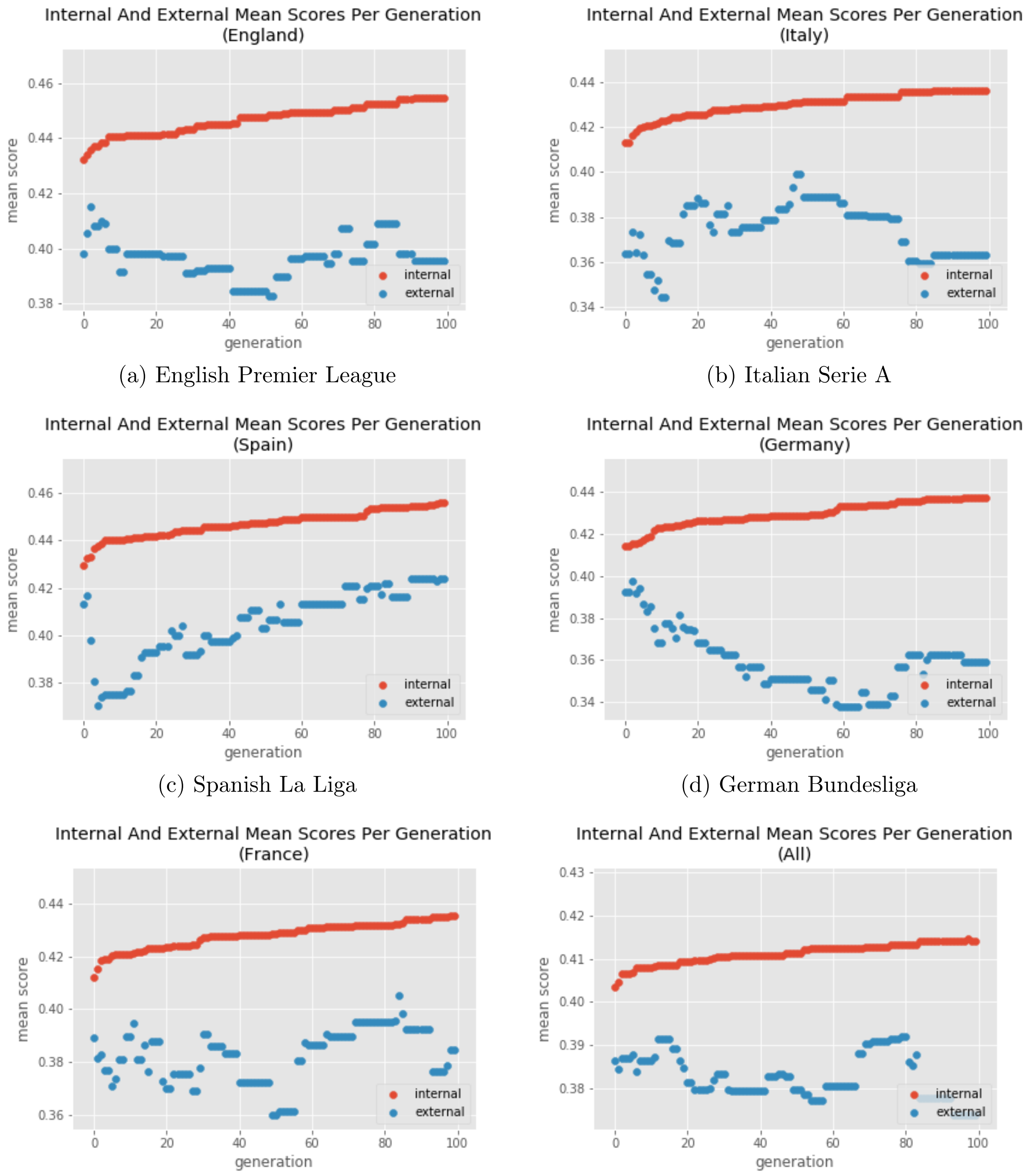
The chromosomes with the highest external scores were recorded for each fold and their mean calculated. We compare the mean external accuracy achieved on each league with that of the previous results from the ten fold cross validation. The results are presented in Table 20. The random forest with feature selection achieved slightly higher scores for all of the leagues with an average of 4.9% higher scores. The standard deviation of the internal scores of the leagues are low, meaning that all the internal scores of the folds were roughly the same. The external scores show a higher variation, implying that for some

Table 20: Mean accuracy of ten outer folds per league using genetic algorithms for feature selection compared with the results of the Random Forest$_{10CV}$ from previous experiment. Highest accuracy achieved per league is marked in bold. Standard Deviation is shown in parenthesis.

Leagues	Mean internal score	Mean external score	Random Forest$_{10CV}$
English Premier League	0.439 (±.006)	**0.440** (±.034)	0.390
Italian Serie A	0.422 (±.007)	**0.426** (±.039)	0.372
Spanish La Liga	0.447 (±.017)	**0.455** (±.031)	0.418
German Bundesliga	0.420 (±.012)	**0.433** (±.031)	0.372
French Ligue 1	0.424 (±.011)	**0.435** (±.040)	0.392
Mean		**0.438** (±.011)	0.389
All leagues	0.408 (±.003)	**0.407** (±.015)	-

folds the selected features were better than others at generalising to the unseen samples. The external scores for each league are similar to their internal scores, mostly lower but some achieve a higher score. In the context of feature selection, this means that predictors that were chosen by the genetic algorithm, were able to discriminate the classes of the training samples. However, given that these scores range about 3.5% within one standard deviation, it is more likely that these would be closer to the internal scores. All mean internal scores are also higher than the those of the random forest$_{10CV}$.

Finally, we measure the percentage that a predictor was present in the feature space of the highest achieving model on unseen data for the different outer folds. The result is displayed in Figure 15, where the rows represent the predictors and the columns denote the different leagues. The value of the cell is the percentage a predictor was included in each fold. The value signifies the importance of that predictor for classifying unseen data. The rows are ordered in descending order by the mean of the attribute across the leagues. For example, `passLongDiff` has a value of 0.9 for the English league. Meaning that when considering the external scores of the chromosomes, this predictor was present in nine feature subsets out of the ten folds the Genetic Algorithm was run on. This indicates that `passLongDiff` was found to be effective in predicting unseen data. Furthermore, the `passTargetDefensiveThirdDiff` and `passbackwardDiff` predictors for the English league were in the best subset for 20% of the folds. Indicating that these might not be good predictors. The predictor might not be present in any of the final chromosomes of the ten folds. We do not report on the optimised chromosomes because we are more concerned with the external scores to gauge how the final model will perform.
`passLongDiff` also has a high percentage in the other leagues, except for the French one



(a) English Premier League

(b) Italian Serie A

(c) Spanish La Liga

(d) German Bundesliga

(e) France Ligue 1

(f) All leagues

Figure 14: Genetic algorithm results for external and internal scores per generation for all league data sets

where it was considered in only four folds. For the Italian, Spanish and German leagues, passLongDiff was considered in 70% of the folds for each league. When combining the samples of all the leagues, this predictor was found in 80% of the folds. On average, passLongDiff was found in 70% of the folds, being the most chosen of all the predictors. The second most considered predictor on average was the assistShotDiff (62%), with passForwardDiff (60%) being the third and passTargetFinalThirdDiff (60%) in fourth. The rest of the predictors were on average chosen less than 60%. Interestingly, the top 11 predictors are all offensive attributes, meaning they relate to match statistics of an attacking team with the intention of scoring a goal. Apart from the aforementioned, the rest are as follows, passSuccessDiff (58%), assistShotIntentionalGoalDiff (57%), possessionAttackDiff (57%), dribbleSuccessDiff (57%), crossFavourTotalDiff (55%), offsideCommittedDiff (55%), shotTotalDiff (53%), interceptionDiff (52%), assistShotIntentionalDiff (50%).
The least used predictors across the leagues are either of a defensive nature or are general attributes that do not signify much motive towards attacking or defending statistics, such as passTotalDiff (50%), averageAgeDiff (50%) and possessionDefenceDiff (50%). The following are the defensive attributes and their respective inclusion mean rate; cardRedDiff (50%), tackleTotalDiff(50%), cardYellowDiff (47%), passTargetMiddleThirdDiff (47%), passTargetDefensiveThirdDiff (0.43%), possessionTotalDiff (40%), tackleSuccessDiff (38%), foulReceivedDiff (38%), passBackwardDiff (37%). It is interesting to note that the following goal scoring related attributes had a lower mean inclusion rate across the leagues compared with the other offensive attributes detailed above; shotOnGoalDiff (47%), dribbleTotalDiff (45%), cornerFavourDiff (43%), crossFavourSuccessDiff (37%).

These results are in line with some of the previous research. In [LBLP10], differences in team attacking attributes were found to be discriminant whilst the defensive ones were not. It was also shown that different leagues have different attributes. Also, general possession was not found to be a good indicator of success in several studies, [JJM04, CCL12]. Several also concluded that direct style of play was a better strategy and more indicative of success [RRFGZ13]. In this study, the attributes that represent direct play are the passLongDiff, passForwardDiff, dribbleTotalDiff and passTargetFinalThirdDiff. Whilst the ones that are more representative of a possession based are possessionAttackDiff, possessionDefenseDiff and passBackwardDiff. The direct play attributes across the leagues are on average selected more often than those of possession based attributes. However, unless looking further into how the Decision Trees in the random forest are generating the rules, we will not be able to know how these predictors relate to strategy and how that strategy relates to a team winning, drawing or losing. For example, although unlikely, there might be a rule in the Decsision Trees which states; possessionAttackDiff > 100 **Then** Away win. Meaning that if the home team has 100 more possessions in the opposition half than the away team has in theirs, the away team will be the predicted winner. This example illustrates that by simply knowing that these attributes are being picked we have to look further into the rules of the trees in order to know how these attributes interplay with one another.

It is also important to view the these results with respect to the individual leagues as this might indicate certain characteristics of the playing styles in the respective leagues. We highlight the features that were most used and least included in the final best chromosomes for every league. For the English Premier League, the top predictors were passLongDiff (90%), assistShotDiff (70%), dribbleTotalDiff(90%), averageAge (70%), redCardDiff (70%), passForwardDiff (80%), passSuccessDiff (70%) and assistShotIntentional (70%). The least predictors included passBackwardDiff (30%), tackleSuccessDiff (20%) and passTargetDefensiveThirdDiff (20%). The top predictors of the Italian league were passTargetFinalThirdDiff (80%), passTargetMiddleThirdDiff (80%), passLongDiff (70%), tackleTotalDiff (70%) and cornerFavourDiff (70%). The least included were passBackwardDiff (30%), possessionAtttackDiff (30%) and crossFavourSuccessDiff (20%). For the Spanish league the most included predictors were passLongDiff (70%), offsideCommitted (70%), shotOnGoal (70%), foulRecieved (70%) . The least used were, crossFavourSuccessDiff (20%), interceptionDiff (30%), crossFavourSuccessDiff (20%) and tackleTotalDiff (20%).

4.6 Feature Selection with Genetic Algorithms and Model Tuning with Random Search

As discussed previously, the hyper-parameters of the model could be tuned such that the model's performance is improved. A common way this is done is by using a technique called grid search. In grid search, a product of all the parameters to be tested is mapped out and the model is evaluated with each combination. A re-sampling technique is carried out for each test so that an accurate estimate of the parameters' performance is taken. Since we are also using feature selection, these two procedures have to be carried out together.

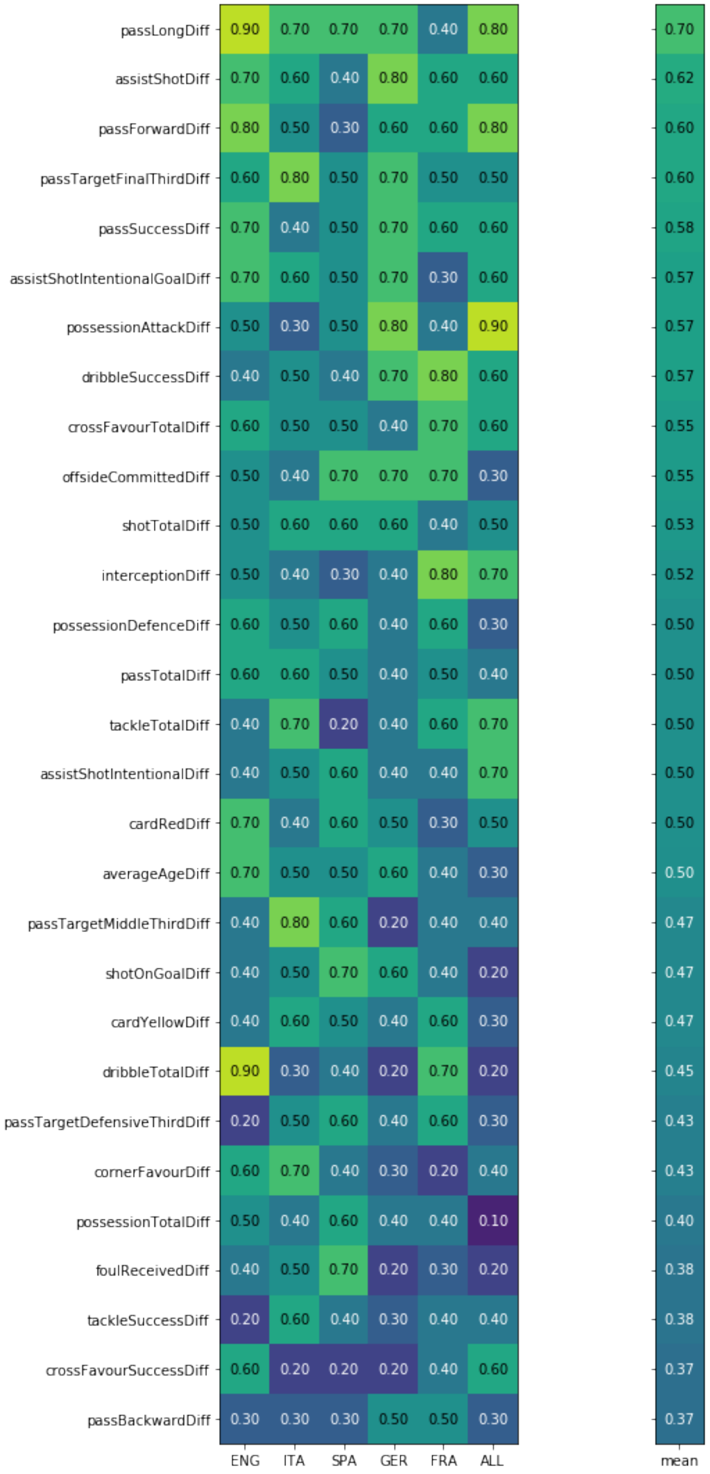


Figure 15: Percentage of the times the predictors were chosen by the GA in each fold per league. For example, the GA chose the predictor `dribbleTotalDiff`, 9/10 folds for the English Premier League. Light cells indicate higher chosen percentage, dark cells denote a lower chosen rate.

Table 21: Hyper-parameter search space for model tuning used in the inner cross validation loop of the genetic algorithm.

Parameter	Description	Search space
n_estimators	Number of estimators	20, 30, 50, 90, 100
max_depth	Maximum tree depth	1-30
min_sample_split	Minimum samples required to split node (percentage of the samples)	0-1
min_sample_leaf	Minimum samples required to make a leaf node (percentage of the samples)	0-1
bootstrap	Whether bootstrapping is used or not	True, False
criterion	Criterion & Measure used to evaluate purity	gini or entropy

If feature selection and model tuning are done sequentially using the same data set, the results from the model tuning procedure will be over-estimated. In our experiment, we add model tuning to the genetic algorithm procedure in the step when the chromosomes are being tested for their performance. The chromosome is tested with all the different parameters and their scores recorded. The parameter yielding the best accuracy is saved with the chromosome. This is the internal score of the chromosome. After the chromosomes are sorted by their fitness scores, the best one is chosen along with its parameter set and is trained once again on all the samples of the training set created by the outer fold. The chromosome is then tested on the validation set left out by the same outer fold. This is the external score of the chromosome and its parameter. This process is run in parallel for all the outer folds. In our experiment, the outer folds splits the data set into ten subsets of training and validation sets. The parameters search space we used in this experiment is listed in Table 21. Because of the computational time for the genetic algorithm to terminate, we altered the genetic algorithm's stopping criteria. Instead of a maximum number of generations, the genetic algorithm terminates if the current generation's best chromosome score does not improve by more than one percent of the last ten generations.

When running the tests on the English samples, the most common values yielding the best external accuracy were the following; gini index as the criterion, max_depth of 5, min_sample_split of 0.44 and min_sample_leaf of 0.139. The number of estimators was split between 50% of the folds having 100 and 20. The same was for bootstrapping, were for half of the folds it was true and half was false. The full results per fold are presented in Table 22. Fold six took the longest to terminate with a total of 23 generations. The mean internal score of all the folds was 0.461(0.004) and the mean external score was 0.441(0.043).

Table 22: The parameters yielding the best external score for each fold when trained on the English league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	1	0.461	4612.737	0.482	5	0.044	0.139	gini	False	20
1	19	0.468	4685.003	0.509	20	0.010	0.072	entropy	True	20
2	7	0.463	4635.442	0.482	5	0.044	0.139	gini	True	100
3	2	0.457	4576.128	0.445	5	0.044	0.139	gini	False	20
4	5	0.462	4624.489	0.418	5	0.044	0.139	gini	True	100
5	2	0.462	4626.889	0.500	5	0.044	0.139	gini	True	100
6	23	0.463	4635.442	0.518	5	0.044	0.139	gini	False	20
7	18	0.464	4647.194	0.455	5	0.044	0.139	gini	True	100
8	0	0.452	4524.344	0.398	20	0.010	0.072	entropy	True	20
9	0	0.457	4581.299	0.505	5	0.044	0.139	gini	True	100

For the Italian leagues, the most common number of estimators was 20, used in half of the folds. Tree max depth had two common values, ten and five, chosen in three folds each. For six of the folds, not using bootstrapping yielded the best results. As for the purity measure, the gini index occurred in eight of the ten folds. The full results are displayed in Table 23. The best parameter combination had an accuracy of 0.510. The parameter set values were, max_depth of 20, min_samples_leaf and min_samples_split of 0.010 and 0.072, respectively. A total of 20 estimators were used and bootstrapping was allowed. The purity measure used was entropy . The mean internal score of all the folds was 0.438 (±0.006) and the external score of 0.471 (±0.041). For most of the folds, the chromosome with the best external score was found in the first generation and the latest at the 19th.

Table 23: The parameters yielding the best external score for each fold when trained on the Italian league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	0	0.433	4332.230	0.490	25	0.010	1.000	gini	False	50
1	19	0.434	4344.554	0.510	20	0.010	0.072	entropy	True	20
2	11	0.447	4474.267	0.436	10	0.016	0.268	gini	False	20
3	0	0.434	4349.244	0.505	5	0.044	0.139	gini	True	100
4	14	0.446	4468.109	0.440	10	0.016	0.268	gini	False	20
5	1	0.445	4452.076	0.354	25	0.010	1.000	gini	False	50
6	0	0.428	4285.779	0.414	5	0.044	0.139	gini	False	20
7	0	0.435	4352.298	0.465	5	0.044	0.139	gini	True	100
8	9	0.435	4352.698	0.414	15	0.118	0.010	entropy	True	90
9	0	0.441	4418.417	0.394	10	0.016	0.268	gini	False	20

For the experiment carried out on the Spanish samples, the gini index was also the most common used purity measure. For six of the folds the number of estimators was 100. For the parameters, bootstrapping, min_samples_leaf and min_samples_split were found in eight of the ten folds with the values True, 0.044 and 0.139, respectively. The best combination of parameters had an external accuracy of 0.538 and had the following as parameters, max_depth of 5, min_samples_leaf of 0.044, min_samples_split of 0.139, gini index as the purity measure, using bootstrapping and a total of 100 estimators. The mean internal score of the ten folds was 0.468 (±0.007) and that of the external accuracy equal to 0.462 (±0.038).

Table 24: The parameters yielding the best external score for each fold when trained on the Spanish league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	14	0.462	4628.176	0.459	15	0.193	0.268	gini	True	20
1	16	0.472	4732.418	0.459	5	0.044	0.139	gini	True	100
2	21	0.476	4766.611	0.500	5	0.044	0.139	gini	False	20
3	3	0.472	4724.317	0.444	5	0.044	0.139	gini	True	100
4	9	0.459	4596.674	0.467	5	0.044	0.139	gini	True	100
5	15	0.478	4782.802	0.393	5	0.044	0.139	gini	True	100
6	0	0.463	4632.929	0.472	5	0.044	0.139	gini	True	100
7	5	0.477	4777.457	0.434	20	0.010	0.072	entropy	True	20
8	3	0.458	4582.369	0.538	5	0.044	0.139	gini	True	100
9	0	0.466	4662.785	0.453	10	0.016	0.268	gini	False	20

In the experiments carried out on the German league samples, the best parameter set yielded an external accuracy of 0.488 on fold number two. The values of the parameter were the following; 20 estimators, max_depth of 15, using bootstrapping and min_samples_leaf and min_samples_split of 0.193 and 0.268, respectively. All the results per fold are presented in Table 25. The gini index was in the parameter vector of the highest scoring parameter in eight folds. The most common number of estimators across the folds was 20 used in six of the ten folds and the use of bootstrapping was also common in six folds. The majority of the folds had different values for the min_samples_leaf and min_samples_split. The mean internal score of all the folds was 0.431 (±0.010) and the external score of 0.415 (±0.037).

Table 25: The parameters yielding the best external score for each fold when trained on the German league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	0	0.451	4512.408	0.368	5	0.044	0.139	gini	False	20
1	6	0.426	4265.013	0.437	5	0.044	0.139	gini	False	20
2	14	0.421	4220.864	0.488	15	0.193	0.268	gini	True	20
3	1	0.438	4388.715	0.395	25	0.016	0.518	gini	False	50
4	8	0.429	4292.914	0.424	20	0.010	0.072	entropy	True	20
5	0	0.419	4202.005	0.400	5	0.044	0.139	gini	True	100
6	0	0.436	4369.236	0.376	15	0.118	0.010	entropy	True	90
7	1	0.434	4344.062	0.400	20	0.193	0.072	gini	False	30
8	1	0.421	4214.992	0.459	15	0.193	0.268	gini	True	20
9	13	0.431	4310.096	0.405	15	0.193	0.268	gini	True	20

The experiments run on the French league samples also had the gini index being selected in the parameter vector that yielded the best external scores for nine of the folds.

Most common number of estimators selected was similar to that of the results using the Spanish samples with a total of 100 trees used in the models for five of the folds. The min_samples_leaf and min_samples_split had the same amount for six of the folds, with the values, 0.044 and 0.139, respectively. Bootstrapping was also used in six of the total folds. The best parameter vector yielded an external accuracy of 0.490 and constituted of the following values for its parameters; gini index as the purity measure, 100 estimators, max_depth of 5, using bootstrapping and min_samples_leaf and min_samples_split of 0.444 and 0.139, respectively. The mean internal and external scores of the folds were 0.450 (±0.006) and 0.438 (±0.036). The results are displayed in full for every fold in Table 26.

Table 26: The parameters yielding the best external score for each fold when trained on the French league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	1	0.449	4495.950	0.441	20	0.193	0.072	gini	False	30
1	1	0.450	4506.612	0.422	10	0.016	0.268	gini	False	20
2	1	0.456	4563.522	0.412	5	0.044	0.139	gini	False	20
3	10	0.456	4568.336	0.465	5	0.044	0.139	gini	True	100
4	1	0.453	4535.987	0.436	20	0.010	0.072	entropy	True	20
5	15	0.440	4411.174	0.480	5	0.044	0.139	gini	True	100
6	4	0.439	4398.536	0.490	5	0.044	0.139	gini	True	100
7	0	0.446	4464.761	0.430	5	0.044	0.139	gini	True	100
8	1	0.452	4525.197	0.363	20	0.193	0.072	gini	False	30
9	1	0.454	4548.448	0.444	5	0.044	0.139	gini	True	100

When considering all the samples of the different leagues together, entropy is found to be the best choice for measuring purity to produce the best external accuracy scores. It is used in seven of ten folds. This is the first time that this measure is more common than the gini index, when compared to the previous experiments. For the number of estimators, the use of 20 trees was found in nine of the folds and the use of bootstrapping in eight. The max_depth, min_samples_leaf and min_samples_split parameters were the same in seven folds, with values of 20, 0.010 and 0.072, respectively. All the results for each fold are presented in Table 27. The parameter vector with the best external score of 0.480 was found to be with the same values as those mentioned above, which most common through all of the ten folds. The mean internal score of all the folds was 0.431 (±0.003) and that of the external score was 0.434 (±0.027).

Table 28: Results summary: Mean accuracy of ten outer folds per league using genetic algorithms for feature selection with model tuning compared with the results of the Random Forest$_{GA}$ and Random Forest$_{10CV}$ from previous experiments. Highest external accuracy achieved per league is marked in bold. Standard Deviation is shown in parenthesis.

Leagues	Mean internal score	Mean external score	Random Forest$_{GA}$	Random Forest$_{10CV}$
English Premier League	0.461 (±.004)	**0.471** (±.041)	0.440 (±.034)	0.390
Italian Serie A	0.438 (±.006)	**0.442** (±.051)	0.426 (±.039)	0.372
Spanish La Liga	0.468 (±.007)	**0.462** (±.038)	0.455 (±.031)	0.418
German Bundesliga	0.443 (±.010)	0.415 (±.037)	**0.433** (±.047)	0.372
French Ligue 1	0.450 (±.006)	**0.438** (±.036)	0.435 (±.040)	0.392
Mean		**0.450** (±.016)	0.438 (±.011)	.389
All leagues	0.431 (±.007)	**0.434** (±.027)	0.407 (±.015)	-

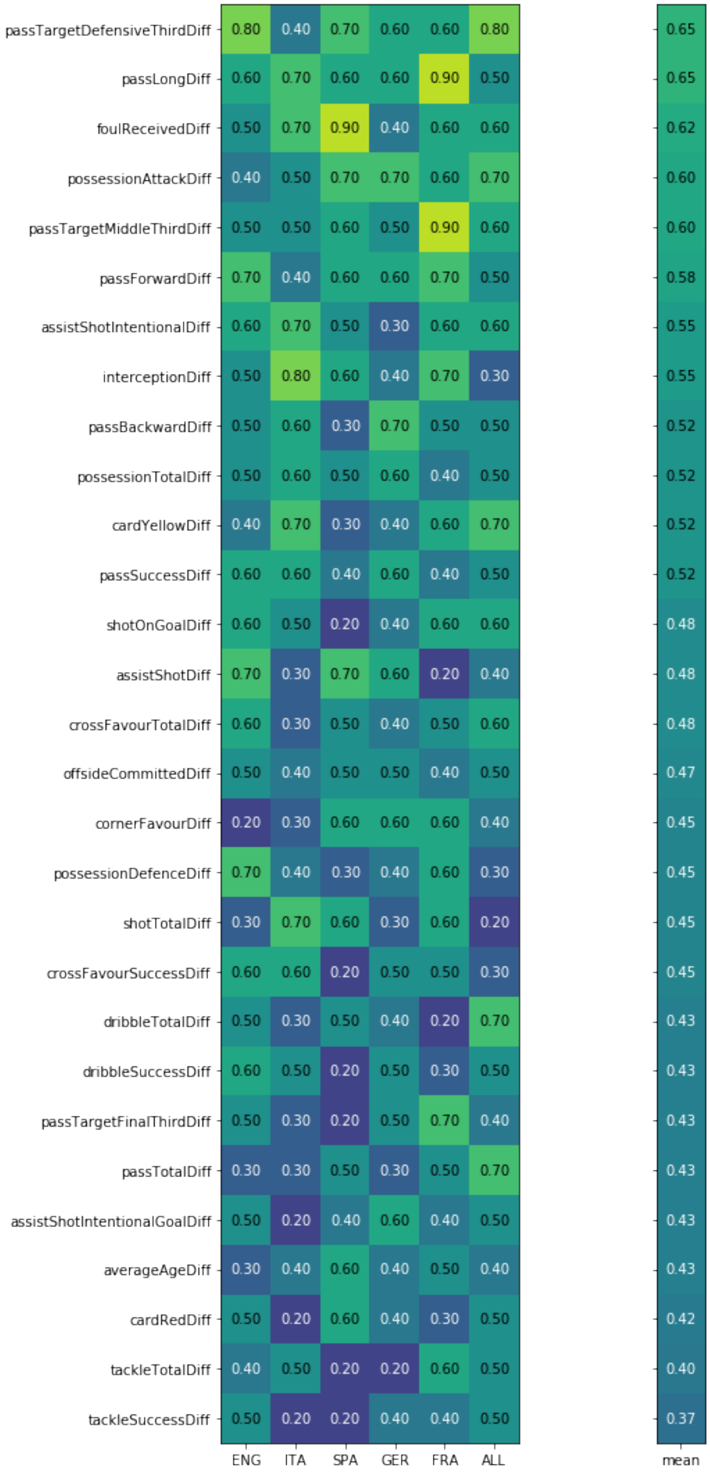
Table 27: The parameters yielding the best external score for each fold when trained on all the league samples

fold	epoch	internal_score	fitness_score	external_score	max_depth	min_samples_leaf	min_samples_split	criterion	bootstrap	n_estimators
0	1	0.430	4303.333	0.435	20	0.010	0.072	entropy	True	20
1	14	0.431	4313.182	0.425	5	0.044	0.139	gini	False	20
2	0	0.427	4280.833	0.466	10	0.016	0.268	gini	False	20
3	1	0.428	4289.336	0.447	20	0.010	0.072	entropy	True	20
4	16	0.433	4334.725	0.435	20	0.010	0.072	entropy	True	20
5	15	0.432	4326.287	0.421	20	0.010	0.072	entropy	True	20
6	0	0.436	4366.104	0.384	20	0.010	0.072	entropy	True	20
7	9	0.435	4355.459	0.412	20	0.010	0.072	entropy	True	20
8	3	0.430	4304.861	0.434	20	0.010	0.072	entropy	True	20
9	16	0.429	4296.217	0.480	5	0.044	0.139	gini	True	100

When looking across the folds of the different leagues it is important to see common values in the best performing parameters. This shows that the model is stable. Meaning that by changing some of the training samples, the model parameters do not change drastically. For most of the parameters, their chosen values were common amongst the other folds. It is interesting to see that for the all leagues model, the criterion changes to entropy when it was mostly the gini index that was being used when the samples grouped by their league. Also, for most of the leagues the number of estimators chosen is low. This is not expected as higher trees should have more bias and less variance in accuracy. However, keeping the rest of the other parameters constant and adding more trees should move the accuracy towards the true error of the model. In Table 28, we present a summary of the results from this experiment, compared with those from previous ones. We can see that the random forest with model tuning out performs both the random forest with feature selection and default parameters and the default ten fold cross validated random forest. Although greater, the external scores do not improve significantly from those of the random forest with feature selection.

In this experiment, feature selection and model tuning were done during the same process using nested cross-validation. Meaning that along with the parameters found to be best suited to predict unseen samples, for each fold a number of predictors were also chosen with those parameters that resulted in the best external scores. The results are shown in Figure 16. Similar to the Figure 15, the rows represent the predictors and columns denote the leagues. Each cell includes the percentage the predictor was chosen in the folds for that league. The predictors are sorted in descending order by their mean inclusion across all the leagues. We can notice some differences from the results found in the first experiment were genetic algorithm did not include parameter optimisation. The predictors with the highest inclusion rate are `passTargetDefensiveThird` (65%), `passLongDiff` (65%), `foulReceivedDiff` (62%), `posessionAttackDiff` (60%), `passTargetMiddleThirdDiff` (60%), and `passForwardDiff` (58%). The least used predictors were `passTotalDiff` (43%), `assisShotIntentionalGoal` (43%), `averageAgeDiff` (43%), `cardRedDiff` (42%), `tackleTotalDiff` (40%), and `tackleSuccessDiff` (37%). The `passTargetDefensiveThirdDiff` is surprising at first, given that this is not a particular statistic which is considered in the football community. However, this may be interpreted as part of the strategy of "playing from the back". In this playing strategy, when the keeper restarts play, he passes the ball short to his defenders and in turn, build up play by passing short balls up the field instead of shooting long balls into the opponents half. This interpretation is further noteicable, when this predictor is taken into with the rest of the other top attributes, such as `passTargetMiddleThirdDiff`, `passForwardDiff` and `possessionAttackDiff`. The attribute `passLongDiff` was on average the second most included. This is an attribute related more to direct play but may show that a combination of direct and possession play are important factors at analysing teams for predictive analysis. More so, when seeing that the most common analysed statistics in the football community by punters and fans alike, the `possessionDiff` (52%), `shotTotalDiff` (45%) and `shotOnGoalDiff` (45%) placed much lower in the table. This may mean that when analysing football matches for predictive purposes it is better to look into attributes that imply some motive and not general statistics like possession or end of action statistics like shots on target.

Finally, we look into detail at how internal and external scores changed with each generation for every fold in the experiments mentioned above. We do this to check if there is any over-fitting occurring in the separate folds. Over-fitting is characterised by the result of a decrease in the validation score as a consequence of an increase on the training score. Meaning that the model 'learned' new rules with which is not able to generalise to unseen



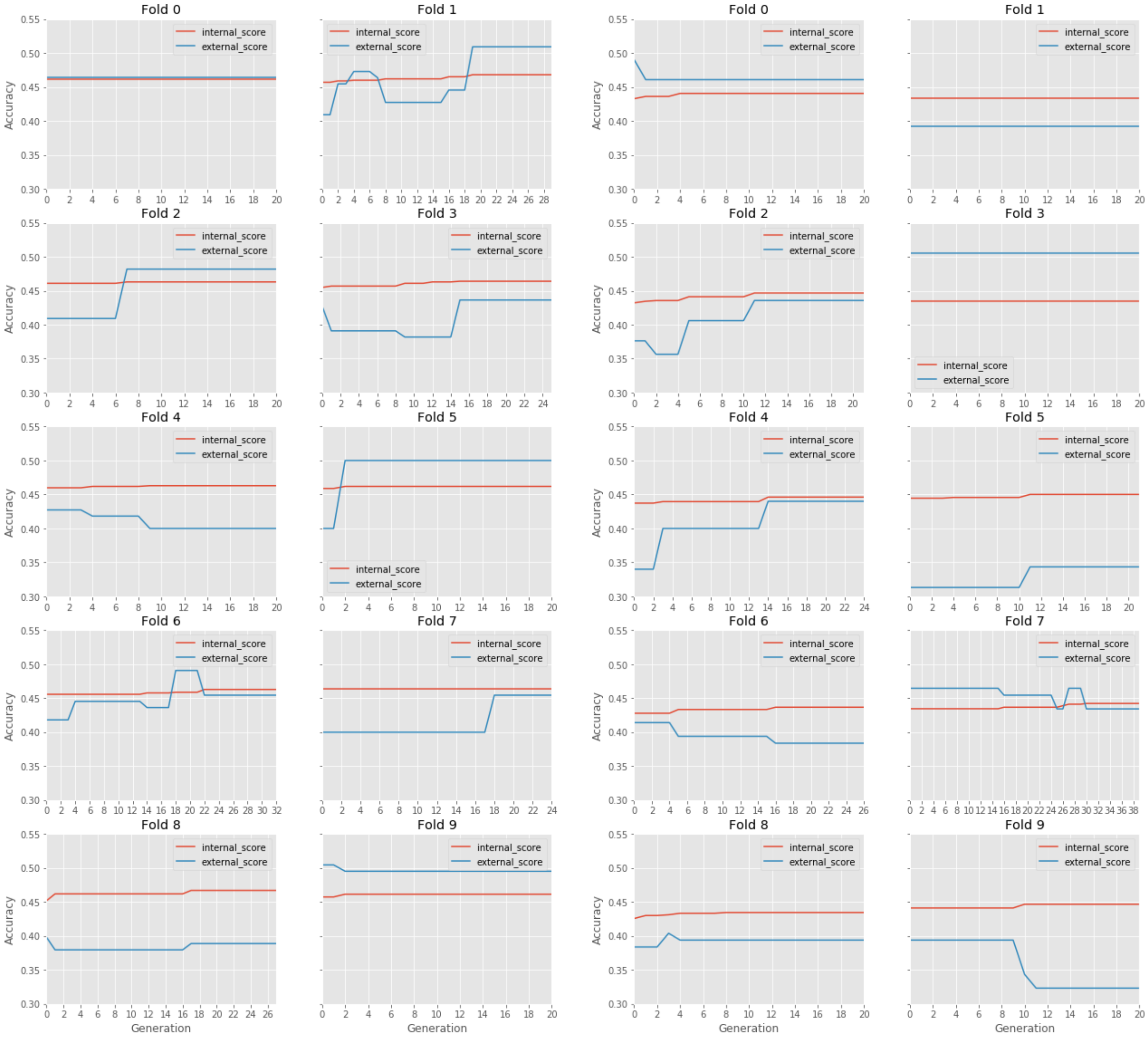
	ENG	ITA	SPA	GER	FRA	ALL	mean
passTargetDefensiveThirdDiff	0.80	0.40	0.70	0.60	0.60	0.80	0.65
passLongDiff	0.60	0.70	0.60	0.60	0.90	0.50	0.65
foulReceivedDiff	0.50	0.70	0.90	0.40	0.60	0.60	0.62
possessionAttackDiff	0.40	0.50	0.70	0.70	0.60	0.70	0.60
passTargetMiddleThirdDiff	0.50	0.50	0.60	0.50	0.90	0.60	0.60
passForwardDiff	0.70	0.40	0.60	0.60	0.70	0.50	0.58
assistShotIntentionalDiff	0.60	0.70	0.50	0.30	0.60	0.60	0.55
interceptionDiff	0.50	0.80	0.60	0.40	0.70	0.30	0.55
passBackwardDiff	0.50	0.60	0.30	0.70	0.50	0.50	0.52
possessionTotalDiff	0.50	0.60	0.50	0.60	0.40	0.50	0.52
cardYellowDiff	0.40	0.70	0.30	0.40	0.60	0.70	0.52
passSuccessDiff	0.60	0.60	0.40	0.60	0.40	0.50	0.52
shotOnGoalDiff	0.60	0.50	0.20	0.40	0.60	0.60	0.48
assistShotDiff	0.70	0.30	0.70	0.60	0.20	0.40	0.48
crossFavourTotalDiff	0.60	0.30	0.50	0.40	0.50	0.60	0.48
offsideCommittedDiff	0.50	0.40	0.50	0.50	0.40	0.50	0.47
cornerFavourDiff	0.20	0.30	0.60	0.60	0.60	0.40	0.45
possessionDefenceDiff	0.70	0.40	0.30	0.40	0.60	0.30	0.45
shotTotalDiff	0.30	0.70	0.60	0.30	0.60	0.20	0.45
crossFavourSuccessDiff	0.60	0.60	0.20	0.50	0.50	0.30	0.45
dribbleTotalDiff	0.50	0.30	0.50	0.40	0.20	0.70	0.43
dribbleSuccessDiff	0.60	0.50	0.20	0.50	0.30	0.50	0.43
passTargetFinalThirdDiff	0.50	0.30	0.20	0.50	0.70	0.40	0.43
passTotalDiff	0.30	0.30	0.50	0.30	0.50	0.70	0.43
assistShotIntentionalGoalDiff	0.50	0.20	0.40	0.60	0.40	0.50	0.43
averageAgeDiff	0.30	0.40	0.60	0.40	0.50	0.40	0.43
cardRedDiff	0.50	0.20	0.60	0.40	0.30	0.50	0.42
tackleTotalDiff	0.40	0.50	0.20	0.20	0.60	0.50	0.40
tackleSuccessDiff	0.50	0.20	0.20	0.40	0.40	0.50	0.37

Figure 16: Percentage of the times the predictors were chosen by the GA with model tuning in each fold per league. Light cells indicate higher chosen percentage, dark cells denote a lower chosen rate.

data. In Figure 17, we show the internal and external scores of each generation for every fold of the model tuning and feature selection process for the English and Italian league samples. In the Subplots, we can visually detect when a new parameter and chromosome combination has been chosen by the genetic algorithm which increases the internal score but consequently decreases the external score. For example, in Subplot 17(a) fold 4, every time a new chromosome, seen on the graph by an increase in internal score, the external external score decreases. The internal score for each fold is around 0.450. This is a good indicator that the model is stable. For the Italian folds in 17(b), we can see overfitting happening in fold six, nine and later stages of fold seven. On the other hand, if we stop the genetic algorithm to early, we run the risk of under-fitting the model. For example, if we stop the genetic algorithm on the English Premier League for fold number four at generation five, then the chromosome and parameter combination that increases the external score at the next generation would not have been found. We display the results of the rest of the leagues in Figure 18.

4.7 Feature Selection and Model Tuning with the addition of Pre-match data to the feature space

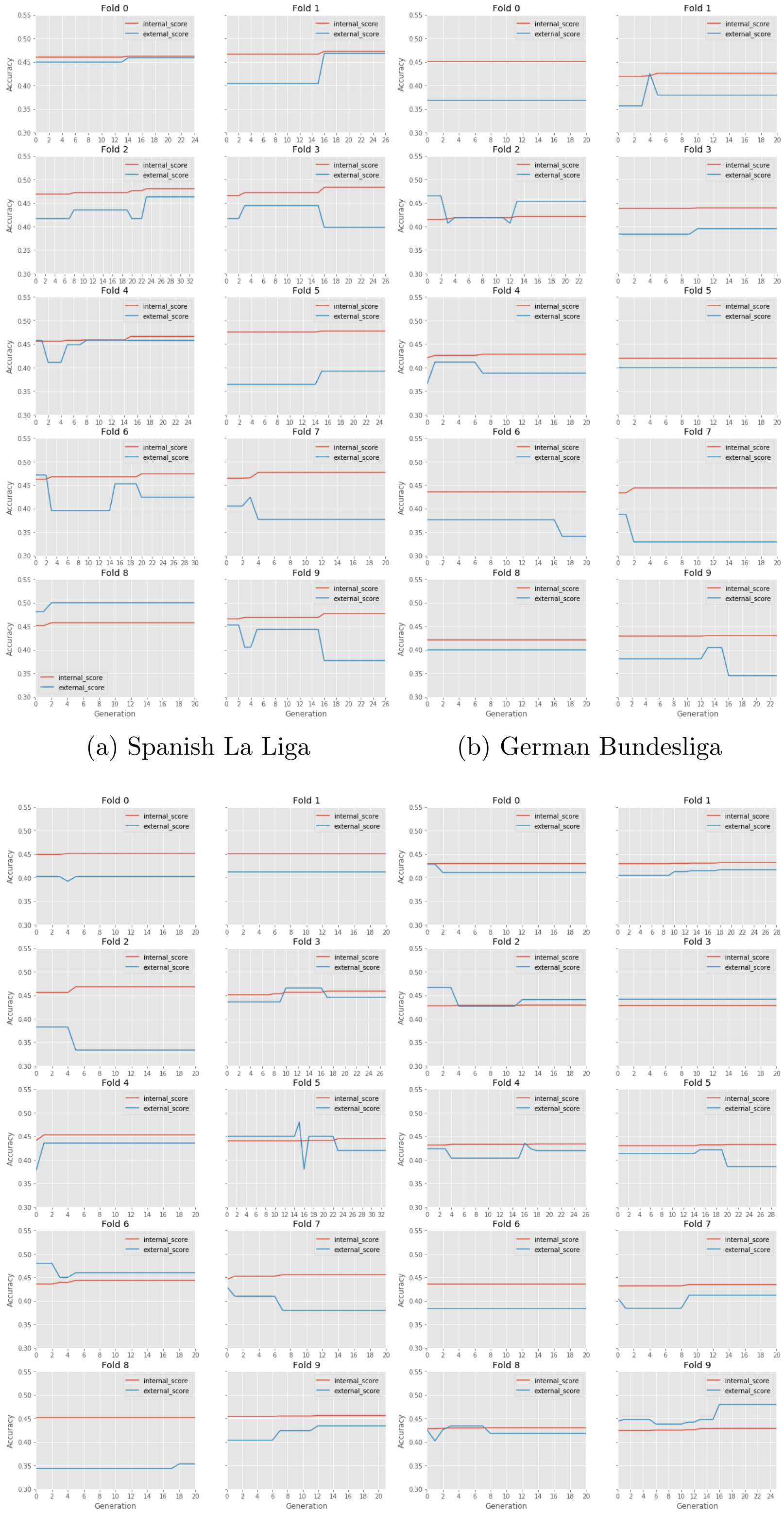
The final test was to aggregate pre-match data with the half-time interval in-play statistics and evaluate whether the accuracy improves with the additional information. The pre-match data included statistics about the team's performance in a season. This included the following predictors, difference in form of the last 6 games between the competing teams, difference in goal scored and goals conceded, difference in attack and defence strengths, reflecting the team's goal scoring and defensive abilities compared with the league average. All these statistics are calculated up to match-game intervals. For example, for match-game one, all the pre-match differences will be 0. These statistics build up as the season progresses to the end. The difference in the pre-match statistics depend on how the opposing teams were performing in that season. For example, a match sample of the opposing teams being the first and the last placed teams would have a large difference between the variables. The closer the teams are in the table should have similar statistics, indicating that they are of similar strengths. We analyse this data by using the same process as the above experiments. However, because of the time dependent variables we do not apply a standard cross-validation for inner cross-validation but use a custom one designed to tackle this problem. Each parameter and chromosome combination in the genetic algorithm are used to train the model season by season and every iteration of the inner loop the pre-



(a) English Premier League

(b) Italian Serie A

Figure 17: Genetic algorithm results for external and internal scores per generation for each fold for the English and Italian league data sets



(a) Spanish La Liga

(b) German Bundesliga

(c) France Ligue 1

(d) All leagues

Figure 18: Genetic algorithm results for external and internal scores per generation for each fold for all league data sets

Table 29: Parameters selected by the genetic algorithm resulting in the highest external score for each league.

league	bootstrap	max_depth	n_estimators	criterion	min_samples_split	min_samples_leaf
English Premier League	False	15	90	entropy	0.010	0.016
Italian Serie A	True	20	100	gini	0.072	0.044
Spanish La Liga	True	25	20	entropy	0.072	0.044
German Bundesliga	True	25	20	entropy	0.072	0.044
French Ligue 1	True	25	20	entropy	0.072	0.044

vious test set is added to the train set until no more test sets are left. Each season acts as a test set except for the first season and each acts as a train set accept for the last season. In this experiment only one outer fold was used because of the time dependency between the data sets. Meaning that the left out samples were of the 2015/16 season and the inner loop to drive the objective function of the genetic algorithm consisted of the following (train/validation) sets; (2010-11/ 2011-12), (2010-11, 2011-12/ 2012-2013), (2010-11, 2011-12, 2012-2013/ 2013-14), (2010-11, 2011-12, 2012-2013, 2013-14/ 2014-15). The chromosome and parameter sets were ranked by the mean accuracy achieved from the mentioned tests and the most performant was then tested on the outer validation set and its score recorded.

In Table 29, we present the parameters chosen by the genetic algorithm. There are similarities in the parameters that yielded the highest accuracy on the external data sets. Bootstrapping is almost always used except for the data set containing the English samples. The max depth of trees allowed is in the range of 15 to 25, with three of the leagues having 25 max depth. For the English Premier League samples 90 estimators were used and 100 for the Italian Serie A. For, the rest of the leagues 20 estimators yielded the highest scores. Entropy was chosen as the impurity measure for all the leagues but the Italian Serie A. The min_samples_split and min_samples_leaf measures were also similar for most of the leagues having values of 0.072 and 0.044, respectively. For English samples, these were 0.010 for min_samples_split and 0.016 for min_samples_leaf.

The highest external score for the English Premier League was 48.2% and the chosen predictors were `preFormDiff`, `preGoalScoreDiff`, `prePointsDiff`, `averageAgeDiff`, `shotOnGoalDiff`, `passBackwardDiff`, `passTargetFinalThirdDiff`, `passTargetMiddleThirdDiff`, `assistShotDiff`, `assistShotIntentionalDiff`, `offsideCommittedDiff`, `possessionAttackDiff`, `possessionDefenceDiff`, `interceptionDiff`, `dribbleTotalDiff` and `dribbleSuccessDiff`. The form, goals scored and points differences were the predictors chosen from the pre-match statistics. The rest of the predictors were from the half-time team attributes. Most are possession and attack related attributes. Number of interceptions was the only defensive attribute chosen.

For the Italian league, the highest external accuracy on the 2015/16 data set was 48.5%. The predictors associated with this score were the following, `preAttackStrength`, `shotTotalDiff`, `passForwardDiff`, `passTargetMiddleThirdDiff`, `passTargetDefensiveThirdDiff`, `cornerFavourDiff`, `foulReceivedDiff`, `assistShotDiff`, `assistShotIntentionalDiff`, `assistShotIntentionalGoalDiff`, `crossFavourTotalDiff`, `possessionTotalDiff`, `possessionDefenceDiff`, `tackleSuccessDiff`. The only pre-match statistic used was the team attacking strength. The half-time interval statistics are mostly related to possession and attacking attributes. Differences in successful tackles was the only defensive statistic included.

The tests on the Spanish samples yielded an accuracy of 43.8%. The predictors chosen by the genetic algorithm were `preGoalsConceded`, `preAttackStrengthDiff`, `preGoalsScoredDiff`, `foulReceivedDiff`, `assistShotDiff`, `passTotalDiff`, `possessionDefenceDiff`, `possessionTotalDiff`, `possessionAttackDiff`, `assistShotIntentionalGoalDiff`, `passForwardDiff`, `passTargetMiddleThirdDiff`, `passTargetDefensiveThirdDiff`, `crossFavourTotalDiff`, `crossFavourSuccessDiff`, `shotOnGoalDiff`, `passLongDiff`, `interceptionDiff`, `cardRedDiff`, `dribbleSuccessDiff`. The pre-match statistics used were differences in goals conceded, goal scored and attacking strength. Similar to both the English and Italian leagues, the attributes are mostly related to attacking and possession based features. Differences in red card and interceptions made were the only two defensive attributes included in the feature subset.

For the German league samples the accuracy was that of 44.9%. The predictors included where `preFormDiff`, `preDiffGoalsScored`, `preDefenceStrengthDiff`, `shotTotalDiff`, `passLongDiff`, `passBackwardDiff`, `assistShotDiff`, `assistShotIntentionalDiff`, `offsideCommittedDiff`, `possessionAttackDiff`, `possessionDefenceDiff`, `tackleSuccessDiff`, `dribbleTotalDiff`. The pre-match statistics, form, goals scored and defence strength were considered in the final subset with no defensive attributes included in the set. The model on the French samples achieved an accuracy of 45.8% on the 2015/16 season samples. The feature subset included the following, predictors, `preGoalsScoredDiff`, `preGoalsConcededDiff`, `preAttackStrengthDiff`, `preDefenceStrength`, `prePointsDiff` were the pre-match data included in the feature subset. The rest were `passLongDiff`, `passBackwardDiff`,

Table 30: Most selected pre-match statistics by the genetic algorithm across the different leagues.

Attribute	Percentage
preAttackStrengthDiff	0.6
preGoalsScoredDiff	0.6
preFormDiff	0.4
prePointsDiff	0.4
preDefenceStrengthDiff	0.4
preDiffGoalsScored	0.2
preGoalsConcededDiff	0.2

`passTargetDefensiveThirdDiff`, `cornerFavourDiff`, `foulReceivedDiff`, `assistShotIntentionalDiff`, `assistShotIntentionalGoalDiff`, `crossFavourTotalDiff`, `crossFavourSuccessDiff`, `interceptionDiff`, `cardYellowDiff`, `cardRedDiff`, `tackleSuccessDiff`, `dribbleTotalDiff`, `dribbleSuccessDiff`. Difference in yellow and red cards were also included as was seen for the model trained on the Spanish samples.

In Table 30, we present the pre-match attributes that were selected the most from the genetic algorithm process used on the different leagues. From the tables, we can see that the most selected attributes were the form, points, attacking strength and goals scored.

From the feature subsets selected, we notice that possession in defence and passing to middle and the defensive thirds are included in many of the sets. As already discussed previously, we interpret this as either being strategy related, for example, teams playing from the back are classified as winners. Or else, some of the chosen statistics might apply to one team and whilst others apply to the opposing team. For example, if possession in defence is found great for the away team and possession in attack is high for the home, this might be indicate a home win. A further look into the classifiers could give more insight on how the decisions are made. In Figure 19, we show an example of a tree estimator used in a random forest. Following the first few rules we can see how the tree is making the classification. For example, if `possessionTotalDiff` $>$ 26.5 & `passForwardDiff` $>$ 68.5 & `shotOnGoalDiff` $>$ 1.5 is classified as a home win. If `possessionTotalDiff` $<=$ 26.5 & `passTargetMiddleThirdDiff` $<=$ -105.5, then away win. Else if, `possessionTotalDiff` $<=$ 26.5 & `passTargetMiddleThirdDiff` $>$ -105.5 & `passSuccessDiff` $<=$ -121.5 is classified as a Draw. More analysis could be carried out on how strategies could be drawn up from such decision rules, however, this is not the scope of the study and could be tackled in future research.

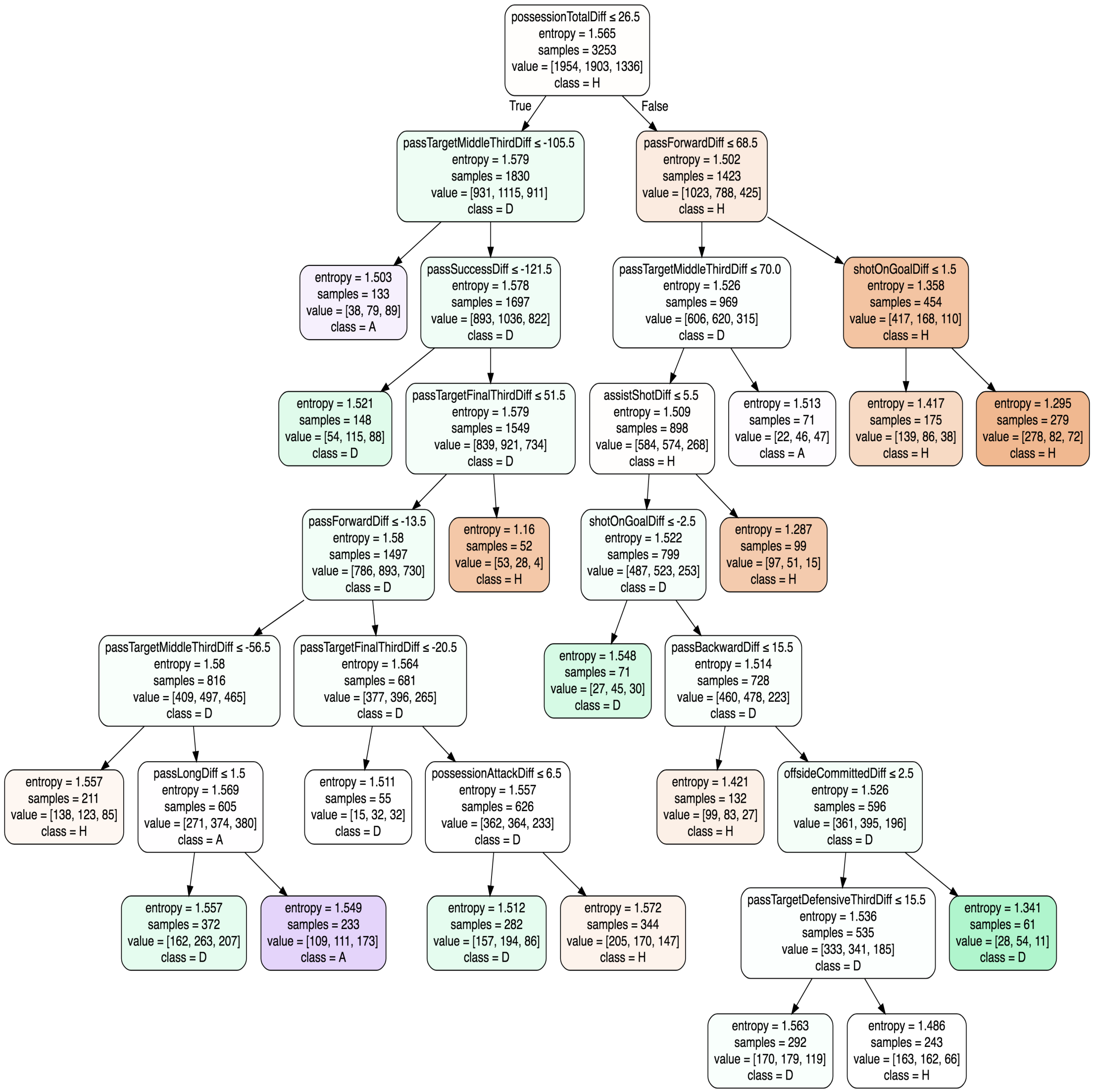


Figure 19: Example of one estimator and its classification rules. Orange nodes indicate a higher probability of a home win, whilst purple ones represent a higher chance for a draw outcome. Green nodes represent away wins. A higher probability of an outcome occurring is represented by a denser gradient of the colour.

4.8 Evaluation on unseen data set

In this section, we evaluate the performance of the models on an unseen data set containing instances from the 2016/2017 season of the leagues used in the experimentation. We run the training/testing phase for 50 times to account for the variance in the results generated by the randomness in the models. We take the mean and standard deviation for each metric calculated to get an estimate of the model stability over the 50 runs. The models are evaluated on their accuracy, precision and recall. Because of some small imbalances in the target classes of the test set, we use a weighted average of the metrics to have a better understanding of how well the models are performing. We then compare the best performing model with the predictions derived from a sports betting exchange market (Betfair) on a small subset of the samples. The sample contains instances from the month of January 2017. This month was chosen on the availability of the data from the Betfair markets.

In Table 31, we present the evaluation metrics of the models on the unseen data set. The base-rule is expected not to have a good overall performance because it is only retrieving the drawn instances. It has 100% recall for the drawn matches but the rest of the classes are mis-classified. The random forest trained with pre-match and in-game data has an accuracy of 50%. It has the same values for the weighted precision and recall measures classification. This indicates that the random forest is incorrectly classifying half of the samples that it predicts as a particular class and not classifying the other half that it should be predicting as that class. The accuracy is within the expected range from the external tests.

Table 31: Evaluation metrics of the English Premier League 2016/17 season test set. BR denotes the base-rule, RF_{10} denotes the default Random Forest and $\mathrm{RF}_{GA(PM+HT)}$ represents the genetically tuned Random Forest with pre-match and in-game data.

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score
BR	0.32	0.10	0.32	0.15
RF_{10}	0.44 (±0.04)	0.44 (±0.04)	0.44 (±0.04)	0.43 (±0.04)
$\mathrm{RF}_{GA(PM+HT)}$	0.50 (±0.02)	0.51 (±0.02)	0.50 (±0.02)	0.49 (±0.02)

The genetically tuned Random Forest is also the best peformant classifier for the Italian, Spanish, German and French leagues. For the Italian league samples the results are shown in Table 32. The accuracy achieved by the random forest was that of 0.48 (±0.02), precision of 0.56 (±0.02) and recall of 0.48 (±0.02). The result achieved for the Spanish samples, shown in Table 33, was an accuracy of 0.40 (±0.02), precision of 0.39 (±0.06) and recall

Table 32: Evaluation metrics of the Italian Serie A 2016/17 season test set. BR denotes the base-rule, RF$_{10}$ denotes the default Random Forest and RF$_{GA(PM+HT)}$ represents the genetically tuned Random Forest with pre-match and in-game data.

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score
BR	0.26	0.10	0.26	0.11
RF$_{10}$	0.42 (±0.03)	0.44 (±0.04)	0.42 (±0.03)	0.42 (±0.03)
RF$_{GA(PM+HT)}$	0.48 (±0.02)	0.56 (±0.02)	0.48 (±0.02)	0.47 (±0.02)

Table 33: Evaluation metrics of the Spanish La Liga 2016/17 season test set. BR denotes the base-rule, RF$_{10}$ denotes the default Random Forest and RF$_{GA(PM+HT)}$ represents the genetically tuned Random Forest with pre-match and in-game data.

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score
BR	0.36	0.13	0.36	0.19
RF$_{10}$	0.37 (±0.03)	0.36 (±0.03)	0.37 (±0.03)	0.35 (±0.03)
RF$_{GA(PM+HT)}$	0.40 (±0.02)	0.39 (±0.06)	0.40 (±0.02)	0.37 (±0.03)

of 0.40 (±0.02). For the German league samples the results are presented in Table 34. The classification accuracy of the random forest was 0.46 (±0.02), with precision of 0.39 (±0.06) and recall of 0.40 (±0.02). The results of the French league are shown in Table 35. The accuracy of the random forest for this league was 0.40 (±0.02). The precision and recall were 0.39 (±0.06) and 0.40 (±0.02), respectively. We can see that the results are similar to some extent between the different leagues. Most importantly is the fact the results resemble the ones achieved from the nested cross-validation procedure. This shows that the results from our experiments were close to the true error of model and these were not over-estimated.

Table 34: Evaluation metrics of the German Bundesliga 2016/17 season test set. BR denotes the base-rule, RF$_{10}$ denotes the default Random Forest and RF$_{GA(PM+HT)}$ represents the genetically tuned Random Forest with pre-match and in-game data.

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score
BR	0.36	0.13	0.36	0.19
RF$_{10}$	0.38 (±0.04)	0.38 (±0.04)	0.38 (±0.04)	0.38 (±0.04)
RF$_{GA(PM+HT)}$	0.46 (±0.02)	0.39 (±0.06)	0.40 (±0.02)	0.37 (±0.03)

Table 35: Evaluation metrics of the French Ligue 1 2016/17 season test set. BR denotes the base-rule, RF_{10} denotes the default Random Forest and $\text{RF}_{GA(PM+HT)}$ represents the genetically tuned Random Forest with pre-match and in-game data.

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score
BR	0.36	0.13	0.36	0.19
RF_{10}	0.38 (±0.04)	0.41 (±0.04)	0.38 (±0.04)	0.38 (±0.04)
$\text{RF}_{GA(PM+HT)}$	0.40 (±0.02)	0.39 (±0.06)	0.40 (±0.02)	0.37 (±0.03)

4.9 Comparison with Betting Markets

We retrieved a small sample of BetFair data to analyse and compare the implied odds at half-time for the match winner martket such that we compare the probabilities of the betting market with that of the model. The results of the English sample is shown in Table 36. We can note that for some matches the probabilities of both the model and the market are similar. Using the Brier score function, the score for the random forest was 0.655 and for the betting market it was 0.622. The closer the Brier scores are to 0 the more accurate the predictions are. For these samples the random forest and the betting market show very similar scores. This might be interpreted as both having comparable performances for predicting full-time results at half-time.

The same experiment was run on a small sample of the Italian data set. For these samples, the betting market performed better than the random forest with a Brier score function of 0.544 and 0.623, respectively. Even so, the results achieved by the random forest is still comparable to that achieved for the English samples. This result means that the betting market performed better for these samples but the random forest had the same performance. The results of this test is presented in Table 37. We can also see that for some results the model has very similar scores to the prediction market. For some instances, the random forest has a better prediction when comapred with the actual observed outcome.

In the evaluation on unseen data set we have seen that the random forest trained with pre-match and in-game data had the best overall performance when considering the accuracy, precision and recall metrics.

Table 36: Comparison of English Premier League match predictions between Random Forest and implied odds from Betfair Exchange

Match	Prediction	Home	Draw	Away
Arsenal v Burnley	Actual	**1**	0	0
	Random Forest	**0.70**	0.17	0.13
	BetFair	**0.70**	0.22	0.07
Burnley v Southampton	Actual	**1**	0	0
	Random Forest	0.14	0.38	**0.48**
	BetFair	0.16	0.41	**0.44**
Everton v Southampton	Actual	**1**	0	0
	Random Forest	**0.59**	0.28	0.13
	BetFair	0.35	**0.40**	0.25
Hull v Bournemouth	Actual	**1**	0	0
	Random Forest	0.35	0.26	**0.39**
	BetFair	0.29	**0.38**	0.32
Liverpool v Swansea	Actual	0	0	**1**
	Random Forest	**0.71**	0.14	0.15
	BetFair	**0.66**	0.28	0.07
Man City v Burnley	Actual	**1**	0	0
	Random Forest	**0.55**	0.31	0.14
	BetFair	**0.53**	0.33	0.14
Man City v Tottenham	Actual	0	**1**	0
	Random Forest	**0.52**	0.22	0.26
	BetFair	**0.46**	0.34	0.20
Watford v Middlesbrough	Actual	0	**1**	0
	Random Forest	**0.40**	0.32	0.27
	BetFair	0.31	**0.46**	0.23
West Ham v Man Utd	Actual	0	0	**1**
	Random Forest	0.29	**0.36**	0.35
	BetFair	0.06	0.23	**0.71**

Table 37: Comparison of Italian Serie A match predictions between Random Forest and implied odds from Betfair Exchange

Match	Prediction	Home	Draw	Away
AC Milan v Cagliari	Actual	**1**	0	0
	Random Forest	**0.45**	0.38	0.17
	BetFair	**0.61**	0.29	0.09
Crotone v Empoli	Actual	**1**	0	0
	Random Forest	0.39	**0.42**	0.19
	BetFair	0.34	**0.41**	0.24
Empoli v Udinese	Actual	**1**	0	0
	Random Forest	0.26	**0.43**	0.30
	BetFair	0.28	**0.43**	0.29
Lazio v Chievo	Actual	0	0	**1**
	Random Forest	**0.54**	0.30	0.16
	BetFair	**0.62**	0.30	0.08
Lazio v Crotone	Actual	**1**	0	0
	Random Forest	**0.60**	0.24	0.16
	BetFair	**0.68**	0.25	0.06
Napoli v Pescara	Actual	**1**	0	0
	Random Forest	**0.56**	0.28	0.16
	BetFair	**0.75**	0.21	0.04
Palermo v Inter	Actual	0	0	**1**
	Random Forest	0.19	**0.41**	0.40
	BetFair	0.12	0.33	**0.56**
Roma v Cagliari	Actual	**1**	0	0
	Random Forest	**0.48**	0.36	0.16
	BetFair	**0.72**	0.21	0.06
Sampdoria v Empoli	Actual	0	**1**	0
	Random Forest	0.38	**0.40**	0.22
	BetFair	**0.46**	0.36	0.18
Sassuolo v Torino	Actual	0	**1**	0
	Random Forest	0.26	**0.39**	0.34
	BetFair	0.23	0.36	**0.40**
Udinese v AC Milan	Actual	**1**	0	0
	Random Forest	0.32	**0.40**	0.27
	BetFair	0.27	**0.40**	0.33

5 Conclusion

Football is a popular sport that is watched and followed by millions of fans around the world. In recent years, sports betting has emerged as one of the most lucrative markets where millions of transactions are processed on a daily basis by its large and active user base. Football is a dynamic, continuous and interactive sport with a low number of goals scored per game. Apart from this, there are many variables that could affect a team's performance. These factors make it harder for analysts to collect and measure meaningful data indicative of a team's success. This variability and uncertainty of what statistics one should follow in order to evaluate team performance makes it hard to predict the outcomes of a football match. When trading, humans make decisions based on their emotions and thus are not able to make rational actions, such as pulling out of a trade. In turn, they fail to maximise their profits or minimise their losses. Also, they are not adept at making or interpreting probabilistic predictions. The application of machine learning techniques into such a problem demonstrable of these factors and difficulties makes it an important investigation to evaluate the predictive performance of such techniques in this field.

In this study, we investigate the application of machine learning algorithms for predicting football match results by the use of in-game and pre-match team statistics. Our main aim of the study was that of predicting the final match result by using the competing teams' performance data till the half-time interval. We capture the most studied and valued team performance indicators by researchers and the football community. The first question we investigated was that of assessing the performance of machine learning techniques when predicting the full-time result of a match by learning differences in in-play match statistics between the opposing teams. In order to carry out this research question, our first objective was to collect a data set containing temporal in-game statistics of football matches. We decided to collect the data of several seasons for the five major European leagues. These being the English Premier League, the Italian Serie A, Spanish La Liga, German Bundesliga and the French Ligue 1. The reason why we used these leagues was that they are the most followed leagues globally and because of the quality of the teams and their players taking part in them. In total, 34 seasons ranging from the year 2009 to 2016 of the leagues were collected. Data sets were also collected for other European leagues for initial experiments. The achievement of this objective involved the collection and parsing of data from known data sources such as *football-data* and *Whoscored*. *Whoscored* make use of Opta as their main data provider, which is a leading outfit in recording temporal and positional in-game match data of football matches. Opta data sets have been shown to be reliable and are used by other researches for their studies. The data collected was not in a format ready to be used for machine learning. The final data sets had to be constructed by taking each action recorded for an event in the original set and by the use of conditional logic rules decide on how it should be presented as a metric in final data set. We carried out some initial experiments on a sample of the data using several machine learning techniques. These were decision trees, random forest, neural networks and naive Bayes. The most performant and consistent technique was found to be the random forest algorithm with an accuracy of 50.2% when applied on a sample of the English Premier League. Because of this, we decided to carry out the rest of the experiments using this technique. We then used ten-fold cross-validation to train and test the random forest with default parameters separately for all the instances of each league. The mean accuracy of the separate results was 38.9%. The highest accuracy from the experiment was 41.8% (±1.2) for the Spanish league and the lowest was of 37.2% (±1.4) for both the Italian and German leagues. When applying a time series approach to train the random forest, the results were seen to improve for the English league (40.3%±2.8) and Italian league (40.4%±3.5) by a degree of 1.3% and 3.2%, respectively. However, the mean across all the leagues (38.4%) was less than that achieved by the cross-validated score. This showed that training the random forest with the samples ordered by time did not improve its predictive performance. This feature set did not include in-between game dependant variables and so this result indicates that there are no latent variables in the data that was being learnt by the order of the instances. One tailed paired t-test showed that the results from the time series random forest with a t-statistic of 1.33 but p-value greater than 0.19, the accuracy was not significantly different than that of the base-rule. The second question was to evaluate if the model's performance could be improved with hyper-parameter tuning and feature selection by the use of genetic algorithms. We tackled this question by building a genetic algorithm and using nested cross-validation in order to get an accurate estimation of the model's performance. Random forests have a built-in mechanism to accommodate for feature selection, however, this is done stochastically. Therefore, by further reducing the search space for the random forest, we are able to get a better judgement on which predictors are able to discriminate better between the classes. We use ten outer folds, with a genetic algorithm run on each to produce the best feature subset and model parameters of that fold. Inner cross-validation was used to evaluate the accuracy of the combination of the chromosome representing the feature space and the model hyper-parameters. This score was used to drive the objective of the genetic algorithm to find an optimal solution. For each generation in the separate folds, the best combination was tested on the left out samples to get the external performance. This was used both as a measure for over-fitting and an empirical estimate for the models' performance. For the first experiment, we use the genetic algorithm for feature selection only. From this test, we find that the mean accuracy (43.8%±1.1) on unseen data improves by an average of 4.9% across the leagues on the results of the default random forest. For each of the leagues, the random forest with feature selection registers a better accuracy. We analyse the features that were selected per league and find that the most selected features by the genetic algorithm are related to offensive attributes. The most included attributes were the following; the number in parenthesis represents the mean percentage the attributes were selected for the different leagues. `passLongDiff` (70%), `assistShotDiff` (62%), `passForwardDiff` (60%), `passTargetFinalThirdDiff` (60%), `passSuccessDiff` (58%), `assistShotIntentionalGoalDiff` (57%), `possessionAttackDiff` (57%), `dribbleSuccessDiff` (57%), `crossFavourTotalDiff` (55%), `offsideCommittedDiff` (55%), `shotTotalDiff` (53%), `interceptionDiff` (52%).

The importance of attacking attributes as performance indicators has been shown to be significant in other literature. The predictors chosen are associated with a direct playing strategy. In this study, we do not look into whether it is better for one team to play one style over another, but other researchers have concluded that teams should adapt a direct playing style as they found that this is more effective in their research. In our study, we do not investigate if a team having less or more of these attributes from their competitor at half-time are contributing to that team winning. Thus further investigation could be done in this area to look into how classes are clustered by the signs of these attributes' values and examining in more detail the underlying rules of the decision trees of the random forest. Hyper-parameter tuning was then applied to the genetic algorithm, for both feature and parameter vector selection. The parameters tuned were the number of trees included in the random forest (20,30,50,10), the criterion for purity measure (Gini index or Entropy), whether bootstrapping was allowed or not, the maximum depth of the trees (1-30), minimum samples required to split a node and minimum samples required to create a leaf node both given as a percentage of the sample size. Randomised Search was used to explore the search space of the parameters. This was preferred to Grid Search because of the less time it required to complete. From the experiment using the same nested cross-validated procedure, it resulted that with hyper-parameter tuning the model achieved a mean external accuracy of 45% (±1.6), a marginal improvement of 1.2% on the accuracy of the random forest with feature selection only. All, expect the German league registered an increase in external accuracy. The external mean scores of the leagues have an average standard deviation of 4.3%. The most stable results come from the model trained with all the instances of the leagues combined with one standard deviation of 2.7% on the accuracy. This is intuitive since more samples were used for training the machine learning model. This stability is also shown by the parameters that were chosen in each fold. By choosing this model we might sacrifice some of the accuracy because of the difference in characteristics in the samples. However, we should be more confident on what that accuracy will be for unseen datasets. The predictors with the highest inclusion rate are `passTargetDefensiveThird` (65%), `passLongDiff` (65%), `foulReceivedDiff` (62%), `posessionAttackDiff` (60%), `passTargetMiddleThirdDiff` (60%), and `passForwardDiff` (58%). The least used predictors were `passTotalDiff` (43%), `assisShotIntentionalGoal` (43%), `averageAgeDiff` (43%), `cardRedDiff` (42%), `tackleTotalDiff` (40%).

For the final experiment we included pre-match data accumulated over a season to the half-time team statistics of each match instance. For the pre-match statistics we considered the overall points, goals scored, goals conceded, attack and defence strength and the form of the opposing teams' prior to the beginning of each match. The statistics were calculated up to every match game for the duration of a season. These were added to the half-time statistics of the respective matches. Hyper-parameter tuning and feature selection was carried out using the same genetic algorithm, however, we made use of a different approach to drive the objective function because of the dependency of in-between match statistics. We trained the models season by season, adding the validated set to the training set after each iteration until no further sets remained. We left out the 2015/16 data set as the testing set to measure the performance on unseen data. With the use of pre-match statistics, the mean accuracy increased to 46.1% (±2.0). The highest performance was for the Italian Serie A and English Premier League with highest external accuracy of 48.5% and 48.2%, respectively. The lowest was for the French Ligue 1, with an accuracy of 41.6%. The most selected pre-match statistics from the different leagues were the differences in goals scored (80.0%), attacking strength (60.0%), form (40.0%), points (40.0%) and defensive strength (40.0%).

We evaluated the final models on an unseen test set containing the instances of the 2016/17 season for the considered leagues. The random forest had an accuracy of 50.0%

on the English Premier League, 48.5% Italian Serie A and 46.0% the German Bundesliga. For the La Liga and French Ligue 1, the random forest had an accuracy of 40.0% on each. We also evaluate the accuracy of the probabilistic predictions of the random forest on a small sample of matches and compare them to the implied probabilities from a betting exchange using the Brier score function. From the results we found that the random forest had similar performance to the betting market, which is mostly made of human traders. This leads us to conclude that since we have similar Brier scores, the models from the random forest have comparable predictions to the betting market. Predicting the final result of drawn games at the half-time interval by using pre-match and in-game statistics is a difficult problem. Because of the low-scoring nature of the game, one goal can change the dynamic of the game significantly and thus undermine a team's performance attributes in relation to them winning the match.

Following on these results, we believe that more research needs to be done with respect to informative predictors. From the results of the experiments that we carried out, we find that there exists a degree of variability in the predictive accuracy of the models, which highly depends on the data that is included in the training set. We interpret this result as requiring further research and analysis to be done on the construction of more informative feature vectors and performance indicators. With regards to in-game statistics, these could be analysed at interval rates, for example, the number of times a team enters the opposition penalty box per minute. This could be applied to other statistics such as successful or key passes made per minute. Grouping these statistics into temporal frames could give further insight into the sustained pressure by a team on the competitor. By using a growth function to give more importance to the latest actions, apart from full-time result predictions, these could also be applied to in-play markets such as next team to score or goal scored in the next 10 minutes. The inclusion of player data, especially key players have been shown to be informative with regards to classification of final match results. Statistics such as the inclusion of such players in the team and their scoring ability, shooting range, ratings and current form should be considered as predictors for training the Machine Learning techniques. Furthermore, other predictors that might help discriminate further between the teams could be looked at by adding more statistics of the opposing teams against whom the relative performances were achieved. For example, certain teams might be more likely to concede or score late goals or score/concede goals from counter attacks or set pieces. All these statistics could be derived and computed from the kind of rich data set that we constructed for this research. Domain experts should also be consulted for their opinions, as models built with their help have been shown to perform better. We have applied Machine Learning techniques to predict the final result of football matches at the half-time interval. From the experiments carried out, we have seen that by using the selected predictors, the best model achieved an accuracy of 50.0% on test data of English Premier League. The results have been shown to be comparable with a betting exchange market. This strengthens the argument that the sports predictions are difficult to predict. The models in our experiments had some variability in the predictions they made on the different sets and partitions they were trained on. We interpret this result as more research needs to be carried out on finding and analysing other informative predictors that may be better representative of a team's ability in outscoring its opponent at the end of the game.

From this study, we have seen that sports prediction is a hard problem to predict. However, from the results we could note that the models had comparable predictive accuracy to that of the betting market. Prediction markets have been shown to be as accurate as bookmaker markets. For some instances, we have seen that the random forest model had probabilities closer to the actual observed result than the market. This means that for those cases the random forest would beat the prices offered by the market. This result could be used by an agent for betting decisions. In the future, with more analysis and other types of data that could be captured and shown to be indicative of successful teams, the results achieved in this study could be improved.

References

[AGF13] Juan M. Alberola and Ana Garcia-Fornes. Using a case-based reasoning approach for trading in sports betting markets. *Applied Intelligence*, 38(3):465–477, Apr 2013.

[ATASNG14] S. Mohammad Arabzad, M.E. Tayebi Araghi, S. Sadi-Nezhad, and Nooshin Ghofrani. Football match results prediction using artificial neural networks; the case of iran pro league. *Journal of Applied Research on Industrial Engineering*, 1(3):159–179, 2014.

[BB10] Gianluca Baio and Marta Blangiardo. Bayesian hierarchical model for the prediction of football results. *Journal of Applied Statistics*, 37(2):253–264, 2010.

[CCL12] Julen Castellano, David Casamichana, and Carlos Lago. The use of match statistics that discriminate between successful and unsuccessful soccer teams. *Journal of Human Kinetics*, 31:137–147, Mar 2012.

[CFN12] Anthony C. Constantinou, Norman E. Fenton, and Martin Neil. pi-football: A bayesian network model for forecasting association football match outcomes. *Knowledge-Based Systems*, 36:322–339, December 2012.

[Con13] Anthony Costa Constantinou. *Bayesian networks for prediction, risk assessment and decision making in an inefficient Association Football gambling market.* PhD thesis, Queen Mary University of London, UK, 2013.

[CT05] Fiona Carmichael and Dennis Thomas. Home-Field Effect and Team Performance. *Journal of Sports Economics*, 6(3):264–281, August 2005.

[EU10] Stephen Easton and Katherine Uylangco. Forecasting outcomes in tennis matches using within-match betting markets. *International Journal of Forecasting*, 26(3):564 – 575, 2010. Sports Forecasting.

[HR11] J. Hucaljuk and A. Rakipović. Predicting football scores using machine learning techniques. In *2011 Proceedings of the 34th International Convention MIPRO*, pages 1623–1627, May 2011.

[JFN06] A. Joseph, N.E. Fenton, and M. Neil. Predicting football results using bayesian nets and other machine learning techniques. *Knowledge-Based Systems*, 19(7):544 – 553, 2006. Creative Systems.

[JJM04] P. D. Jones, N. James, and S. D. Mellalieu. Possession as a performance indicator in soccer. *International Journal of Performance Analysis in Sport*, 4(1):98–102, 2004.

[KJ13] Max Kuhn and Kjell Johnson. *Applied Predictive Modeling.* Springer, New York, Heidelberg, Dordrecht, London, 2013.

[KM03] Franc Klaassen and Jan Magnus. Forecasting the winner of a tennis match. *European Journal of Operational Research*, 148(2):257–267, 2003.

[LBLP10] Joaquin Lago-Ballesteros and Carlos Lago-Peñas. Performance in team sports: Identifying the keys to success in soccer. *Journal of Human Kinetics*, 25:85–91, Sep 2010.

[LPD10] Carlos Lago-Penas and Alexandre Dellal. Ball possession strategies in elite soccer according to the evolution of the match-score: the influence of situational variables. *Journal of Human Kinetics*, 25:93–100, Sep 2010.

[LPGRY17] Carlos Lago-Penas, Miguel Gomez-Ruano, and Gai Yang. Styles of play in professional soccer: an approach of the chinese soccer super league. *International Journal of Performance Analysis in Sport*, 17(6):1073–1084, 2017.

[Mah82] M. J. Maher. Modelling association football scores. *Statistica Neerlandica*, 36(3):109–118, 1982.

[MKC$^+$08] Byungho Min, Jinhyuck Kim, Chongyoun Choe, Hyeonsang Eom, and R.I. (Bob) McKay. A compound framework for sports results prediction: A football case study. *Knowledge-Based Systems*, 21(7):551 – 562, 2008.

[OF97] E. Ong and A. Flitman. Using neural networks to predict binary outcomes. In *1997 IEEE International Conference on Intelligent Processing Systems (Cat. No.97TH8335)*, volume 1, pages 427–431 vol.1, Oct 1997.

[Øvr08] Øyvind Norstein Øvregård. Trading "in-play" betting exchange markets with artificial neural networks. Master's thesis, Norges teknisk-naturvitenskapelige universitet, 2008.

[Peñ14] Javier López Peña. A markovian model for association football possession and its outcomes. 2014. arXiv:1403.7993.

[PVG⁺11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[RCH94] G. Ridder, J. S. Cramer, and P. Hopstaken. Down to ten: Estimating the effect of a red card in soccer. *Journal of the American Statistical Association*, 89(427):1124–1127, 1994.

[RMYA17] Nazim Razali, Aida Mustapha, Faiz Ahmad Yatim, and Ruhaya Ab Aziz. Predicting football matches results using bayesian networks for english premier league (epl). *IOP Conference Series: Materials Science and Engineering*, 226(1):012099, 2017.

[RPR05] A. P. Rotshtein, M. Posner, and A. B. Rakityanskaya. Football predictions based on a fuzzy model with genetic and neural tuning. *Cybernetics and Systems Analysis*, 41(4):619–630, Jul 2005.

[RRFGZ13] Carlos Ruiz-Ruiz, Luis Fradua, Ángel Fernández-GarcÍa, and Asier Zubillaga. Analysis of entries into the penalty area as a performance indicator in soccer. *European Journal of Sport Science*, 13(3):241–248, 2013. PMID: 23679140.

[SS09] Martin Spann and Skiera. Sports forecasting: a comparison of the forecast accuracy of prediction markets, betting odds and tipsters. *Journal of Forecasting*, 28(1):55–72, 2009.

[TJ15] Niek Tax and Yme Joustra. Predicting the dutch football competition using public data: A machine learning approach. *Transactions on Knowledge and Data Engineering*, 10(10):1–13, 2015.

[TRB10] Albin Tenga, Lars T. Ronglan, and Roald Bahr. Measuring the effectiveness of offensive match-play in professional soccer. *European Journal of Sport Science*, 10(4):269–277, 2010.

[VNH14] Martin Vogelbein, Stephan Nopp, and Anita Hökelmann. Defensive transition in soccer – are prompt possession regains a measure of success? a quantitative analysis of german fußball-bundesliga 2010/2011. *Journal of Sports Sciences*, 32(11):1076–1083, 2014. PMID: 24506111.

[VvK16] Petar Vračar, Erik Štrumbelj, and Igor Kononenko. Modeling basketball play-by-play data. *Expert Syst. Appl.*, 44(C):58–66, February 2016.