Predicting Premier League Results & Comparison to Bookmakers' Odds

Patrick Dillon

Final Year Project – **2016**
B.Sc. Double Honours in Computer Science

Maynooth University
National University of Ireland Maynooth

Department of Computer Science
Maynooth University
Maynooth, Co. Kildare
Ireland

A thesis submitted in partial fulfilment of the requirements for the B.Sc Double Honours in Computer Science
Supervisor: Dr. Phil Maguire

Abstract

The ability to predict the outcomes of football matches accurately can be a very lucrative enterprise today. Sports betting is enormous industry today, with millions of Euro being wagered on the outcomes of sporting events every year. The aim of this project is to create a software system which produces the probabilities of the possible outcomes of Premier League matches and try to exceed the performance, by some metric, of the predictions derived from the bookmakers' and betting exchanges' odds. This report will outline the attempts at creating such a system through machine learning, Monte Carlo simulations and Poisson distributions and how the data required for the system was obtained through web scraping techniques. Also gathered using web scraping techniques is odds data from various bookmakers and betting exchanges. To evaluate the system's success, ideas from information theory are used to show that the system has "outpredicted" or outperformed the predictions derived from bookmakers and betting exchanges over 990 Premier League matches. The result of this project gives a powerful, generalizable predication tool for football matches.

Contents

1 Introduction

Sports betting is massive industry today. This can be exemplified by the recently agreed €9.4 billion merger between two of the largest betting companies Paddy Power and Betfair [1]. Every weekend from August to May, punters try to win money by betting on Premier League matches, the most watched football league in the world[2].

It would be of high value to both the bookmakers' and punters have a system which accurately predicts the outcome of matches. For the punters, the more accurately you can predict the outcomes of matches, the greater the number of bets you win, so the larger amount of money you can make. For the bookmakers, accurate predictions allow bookmakers to offer more accurate odds.

In football, coming up with an accurate prediction model can be quite difficult - a ball bounces the wrong way, a goalkeeper fumbles the ball out of his hands or a player misses a penalty. Generally, football can be quite a stochastic game due to the influence of such random variables.

The main goal of this project is to create a software system that outputs the probabilities of the different outcomes of a Premier League match, i.e. to produce the probabilities of a home team win, a draw or an away team win for any match. These probabilities have to be "accurate" in some way, so some metric has to used to measure how well a prediction method works in respect to the probabilities of various bookmakers and betting exchanges.

To build such a system, data was gathered on the football teams and matches from many years previously from internet sources, using web scraping tools, such as Selenium and Beautiful Soup libraries for the Python programming language.

Three approaches were made at making a prediction system. The first approach was machine learning classification. This involved selecting different aspects of the football team data and using it to train a variety of machine learning classifiers. These classifiers tried to learn the outcome of the football match with the data given. The second approach involved Monte Carlo simulation. This involved taking some statistic to be used as a goal rate and "simulating" each match many times, and figuring out how many times each team should score during a match. Adding up the home wins, away wins and draws for each simulated match gave the probabilities for each outcome of the match. The final approach used a Poisson distribution along with some measure of a goal rate, to determine the probability of the number of goals scored by each team in a match.

The probabilities that were outputted from these prediction methods were evaluated using a measure called the "Shannon Information Content", which is a measure of surprising an outcome is compared the probability you generated for that outcome and it is measured in bits. This measure is used to compare the results of the prediction against the probabilities derived from bookmakers' and betting exchanges' odds, which were gathered from internet sources.

I tested my prediction system over 990 matches from 2013-14 season to the 2015-16 season. The best method for calculating probabilities was a logistic regression machine learning classifier. The average Shannon Information Content for each match was 1.381 bits. This outperformed the bookmakers and betting exchanges significantly whose best average was 1.405 bits per match.

2 Technical Background

2.1 Topic Material

There has been already several papers written at trying to predict football (as well as other sports) matches through a variety of techniques.

One such paper [3] uses machine learning techniques to predict the results of the football club Tottenham Hotspur in the Premier League using data from the 1995-96 and 1996-97 seasons. The statistics used by Bayesian networks for each match were the presence of "key" players, the position of another "key" player, the quality of opposing team and whether the team played home or away. The best model this paper produced was a Bayesian network, which, when trained on data from the 1995-96 season, correctly predicted 65.79% of Tottenham Hotspur's matches for the 1996-1997 season.

Bayesian networks also plays a part in a paper which attempts to predict matches in the 2002 World Cup [4]. This machine learning method combined with rules for strategy making and result calculating are used to create a prediction of a time series of a football match.

Research is not limited to predicting football matches. One paper [5] uses a Naive Bayes classifier to predict the outcomes of basketball matches in the NBA. It used a variety of statistics, such as the teams' league standings, as input to the classifier and mad 67% correct predictions over the NBA 2009-10 season.

Another system for prediction [6] used the Elo rating system to predict and produce probabilities for the outcomes of Australian Rules matches. This rating system was originally designed to grade chess players and it was combined with machine learning techniques and modified to work for data on the Australian Football League.

2.2 Technical Material

2.2.1 Odds

There are several types of odds that a bookmaker can offer. The odds of an outcome correspond to the probability of that outcome. fractional odds are in the form x/y. This means if you wager €y and you win the bet, you will receive €x as winnings. To convert these odds to probabilities you calculate $\frac{y}{x+y}$. Decimal odds are a decimal number x. This means if you wager €1 and you win, you will receive €x as winnings To convert these odds to probabilities you calculate $\frac{1}{x}$[7]

2.2.2 Information Theory

Say we have event E, whose outcome is the value x. x can take on any of the values of the set $\{a_1, ..., a_t\}$ whose associated probabilities are $\{p_1, ..., p_t\}$ Then the "Shannon Information Content" is $h(x = a_i) = log_2(\frac{1}{p_i})$. This is a measure of how much information is gained through a statistical experiment. In other words, the larger h is, the smaller the probability you predicted of the outcome of that experiment and more surprising the outcome was to you. [8]

2.2.3 Selenium & Beautiful Soup libraries

Selenium is a library for the Python programming language which contains tools for interacting with web browsers, such as Firefox and PhantomJs (a non-graphical user interface browser). It allows you to automate all possible browser functions that would be available such retrieving web pages and clicking links or buttons on the page. [10] It also allows to extract elements within a HTML web page through css selectors and XPath (a language for finding nodes in a HTML document). Another library which is used for extracting elements from a HTML page is beautifulsoup. It is easier to use than selenium for finding elements but you cannot dynamically interact with the web page. [11]

2.2.4 scikit-learn & Machine Learning

The most important library I used for this project were scikit-learn. This contains a set of tools to perform machine learning. Classification problems are those where you want to classify data into a number of different into different collections. For example, you may want to classify patients into two classes, those you have cancer and those who don't, based on data about those patients' blood test results. The classifiers in the scikit-learn library can all be used in the same way: First, you have to train the classifier using the training data set. This set contains a lists of numeric data along with the corresponding class it belongs to (it is data which you already know which class it belongs to). The `fit` method is used to train the classifier with the training data. Then you test the classifier on other data to see how accurate it is using the `predict` method of the classifier. The prediction the classifier makes is the class which the classifier has determined to be most probable. Most classifiers in the scikit-learn libraries allow the user to see the probability of each class for a test sample by using the `predict_proba` method [12] [16]

2.2.5 Poisson Distributions

A Poisson distribution is a probability model which is used to find the probability of the success of an event happening k times, where k is an integer. The formula for Poisson distribution is a follows:

$$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

where λ is the average number of successful events, e is the Euler constant and $P(X = k)$ is the probability of k successful events. [9]

3 The Problem

The main aim of this project is to create a system which allows me to predict the results of Premier League football matches accurately and which "outpredict" odds on offer from bookmakers and betting exchanges. To create any prediction system I need information about previous results and matches. To do this, I need to gather data about previous and current Premier League teams and results (which is preferably online and free to use). Using this data I need to be able to create a program which outputs probabilities of each possible result(Win, Loss, Draw) of a each match. These predictions must be more accurate than bookmakers' predictions. I also need a way of gathering data from the bookmakers and ascertaining the probabilities from these odds.

For each match, it would be ideal that the outcome of the match would be the outcome associated with the largest probability of the probabilities produced. In other words, we are not surprised of the outcome of a match given the set of probabilities for the match. A precise mathematical definition of what this problem is follows:

Let T be a set of matches. For $m \in T$ produce a set of probabilities of the possible outcomes of the match $\{p_{m,h}, p_{m,d}, p_{m,a}\}$ where $p_{m,h}$ means the probability of the home team winning match m, $p_{m,d}$ means the probability of the match m being drawn and $p_{m,a}$ means the probability of the away team winning the match m. We want to minimize the following formula:

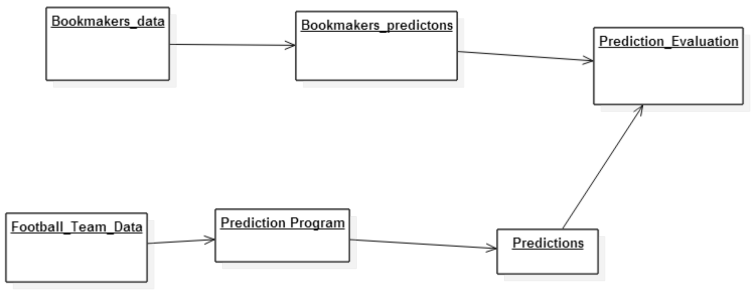
$$\sum_{m \in T} log_2(\frac{1}{p_{m,i}})$$

where i is the result of the match. i.e. minimise sum total of the Shannon information Content for all matches. Then find the sum total of the Shannon Information Content for the probabilities derived from bookmakers and betting exchanges and compare the results of the system to these results.

The problem can be divided up into several parts:

1. Find sources of data about premier league teams and results and collect this data.
2. Create a system using the data gathered to predict Premier League matches
3. Find a source of data for bookmakers' odds and convert these to probabilities
4. Compare my predictions versus the bookmakers' predictions

Figure 1: A simple model of the problem.



The functional requirements of this project are:

- Produce probabilities of Premier League football matches.
- Find and manipulate data from Premier League teams and results.
- Perform better than the bookmakers odds.

The non-functional requirements are:

- Use free data preferably from the internet.

- Make sure the system efficient.

4 The Solution

4.1 Choice of Tools

I decided to use the Python programming language for this project. The reasons for this choice was that it was the programming language I was most proficient in and it was a wide variety of data analysis and web scraping tools available for it.

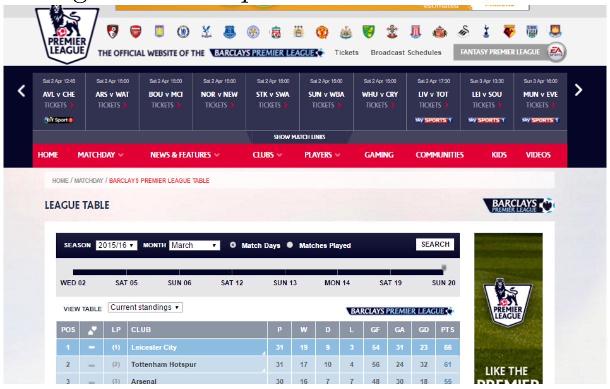
4.2 Football Data

There is a variety of websites which provide historical data on Premier League teams, matches and results. However, most did not provide the data in an easily downloadable and readily usable format. Rather, if you wanted to collect the information in a usable format they would have to use web scraping techniques. My two main sources of data were the football-data.co.uk [13] and the EA Sports Performance Index [14]

The most important source of data was the football-data.co.uk website. This website had CSV files which contained information for each match of various European Top level leagues, including the Premier League for many years. This was very useful as the data was easy to handle when creating my prediction method.

Another source I used was the EA Sports Performance Index on the official Premier League website. It contained a vast amount of data, including league tables, individual matches and player data from virtually all Premier League seasons. However, the data from this source was not readily available for use in a computer program; so web scraping techniques must be used to extract data from the pages.

Figure 2: EA Sports Performance Index.



LEAGUE TABLE

SEASON 2015/16 MONTH March ● Match Days ● Matches Played

WED 02 SAT 05 SUN 06 SAT 12 SUN 13 MON 14 SAT 19 SUN 20

VIEW TABLE Current standings

POS		LP	CLUB	P	W	D	L	GF	GA	GD	PTS
1	–	(1)	Leicester City	31	19	9	3	54	31	23	66
2	–	(2)	Tottenham Hotspur	31	17	10	4	56	24	32	61
3	–	(3)	Arsenal	30	16	7	7	48	30	18	55

To obtain the league table data (such as league table position, goals scored that season etc), I downloaded the web pages of interest and then parsed the web page by using the Beautiful Soup library. Through the use of Firefox's web inspection tools, I was able to figure which HTML elements contained the data I was after. Then by using Beautiful Soup, I was able to extract this data from the HTML element. For example, in the page the league position of each team was contained in HTML table cell which was in the CSS class "col-pos". After collecting all the season league table data I stored it in CSV files.
Associated Files: `season_data_parse.py`

4.3 Machine Learning methods

The problem of forecasting the outcome of a match can be characterised as a classification problem (whether the outcome is a Win, Loss or Draw), but for this project we also require that probabilities for each outcome are produced Fortunately, The scikit-learn library has a wide variety machine learning classifiers available to use for this task. For training data

I used data from the seasons 2001-02 up to 2012-13. For each match in the training set I had to form a "vector", or a list of numeric values were which associated with the teams involved in the match, along with the match result. These vectors had to represent the the strength of each team involved in the match. This would allow the classifier to learn how to predict the result of the match based on the strength of the team given by the associated values. For example, a vector might be $\mathbf{x} = [x_1, x_2]$ where x_1 is the league position of the home team last season and x_2 is the league position of the away team last season. To train the classifier, the program had to create such a vector for all matches in the training set and pass this data along with the corresponding result into the machine learning classifier. To get the probabilities, you create $\mathbf{x}$ for the vector you want and using the `predict_proba` method you can find out the probabilities of the outcomes of the match. Here are the different "vectors"(the different methods of data as input to the classifier) by the classifier for distinguishing the strength of the teams:

- Method 1: the league table position, the goals scored, the goals conceded and the number of matches won, lost and drawn by the team from the previous season to the season in which the match is being played.
- Method 2: The same information as Method 1 but for the previous two seasons.
- Method 3: the league positions for both teams from the previous season to the season in which the match is being played.
- Method 4: the league positions for both teams for the previous two seasons
- Method 5: the league table position, the goals scored, the goals conceded both home and away by the team from the season up to the point the match is being played as well as the previous season.
- Method 6: The same as Method 5 but for the previous two seasons along with the season data up to the point the match is being played.

If the team was not in the league the previous season the match was being played (they were promoted from the lower division that season) I used data from the teams in the bottom 3 league positions that season.

Here is a list of the different machine learning methods used in the classifiers I used from the scikit-learn library:

- Logistic Regression
- Stochastic Gradient Descent with log loss function
- Stochastic Gradient Descent with modified Huber loss function
- Support vector machine with a linear kernel
- Support vector machine with a quadratic kernel
- Support vector machine with a cubic kernel
- Support vector machine with a sigmoid kernel
- Support vector machine with a radial basis function kernel
- Decision tree
- K Nearest Neighbours
- Random forest of decision trees
- Linear Discriminant Analysis
- Quadratic Discriminant Analysis
- Bernoulli Naive Bayes

- Gaussian Naive Bayes
- Gradient Boost

Also used was a “dummy” classifier, which always predicted the class of an input was the class that most frequently appeared in the test set. This was used as a comparison against all the other methods.

Associated Files: `MLMachine.py`, `season_predictions.py`, `Match.py`

4.4 Goals Predictions Methods

Another way of predicting matches is to predict how many goals are scored by each team in each match. A method for doing this is to have a measure of some sort of goal rate i.e. how many goals are scored per minute by a team and “simulate” how many goals are scored during a match. To do this, information were taken on the goals scored and goals conceded data for both teams in a match. Then, a Monte Carlo simulation of a match was created. This involved looping over “90 minutes” of the match. In each “minute” I generated a random number between 0 and 1. If the goal rate was less than the random number, the team “scored” a goal. Each simulation resulted in an outcome: home win, away win or draw. I added up all the home wins, draws and aways wins and divided each one by the number of simulations to get the probabilities of each possible outcome. Here is a list of the different of statistics I used to create a goal rate I used in the Monte Carlo simulations.

- Method 1: goals scored by the team the previous season
- Method 2: the average of goals scored the previous season and goals conceded by the opposing team the previous season
- Method 3: goals scored by the team for the previous season and goals scored by the team in the season the match is being played up to the day of the match.
- Method 4: the average of goals scored by the team and goals conceded by the opposing team from the previous season as well as the current season the match is being played (up to the day of the match being played).

Associated Files: `season_predictions.py`, `Match.py`

4.5 Poisson distribution

A more analytic approach to predicting how many goals were scored in a match was to use a Poisson distribution. Given a goal rate λ for some team A, you can use a Poisson distribution to find the probability of team A scoring k goals in a match. A maximum number of goals scored by a team in the Premier League was 9 goals, so it was sufficient to calculate for each team in the match

$$P_A(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

for $0 \leq k \leq 9$. I assumed the goals scored by the home team to be an independent event from the goals scored by the away team (which, of course is not realistic, for instance a team will change it’s strategy when they have conceded a goal). Then this allowed me to figure out the probabilities of the final score of the match by multiplying the probabilities of what the individual teams scored. For example, say you wanted to find the probability of the match ending in 2-1 win of team A over team B then calculate the probability of team A scoring 2 goals by using a Poisson distribution and calculate the probability of team A scoring 2 goals. Then, the probability of the result of a match being a home win is given by:

$$\sum_{k>j, 0\leq k,j\leq 9} p_A(k)p_B(j)$$

To find the probability for away wins and draws change the expression $k > j$ in the equation above to $k < j$ and $k = j$ respectively.

To find a goal rate I took statistics from the previous and current season (i.e. the season the match is taking place in up to the day the match is being played). Then I used the following formulas to determine the goal rate:

$$GF = \alpha GFC + (1 - \alpha) GFP$$

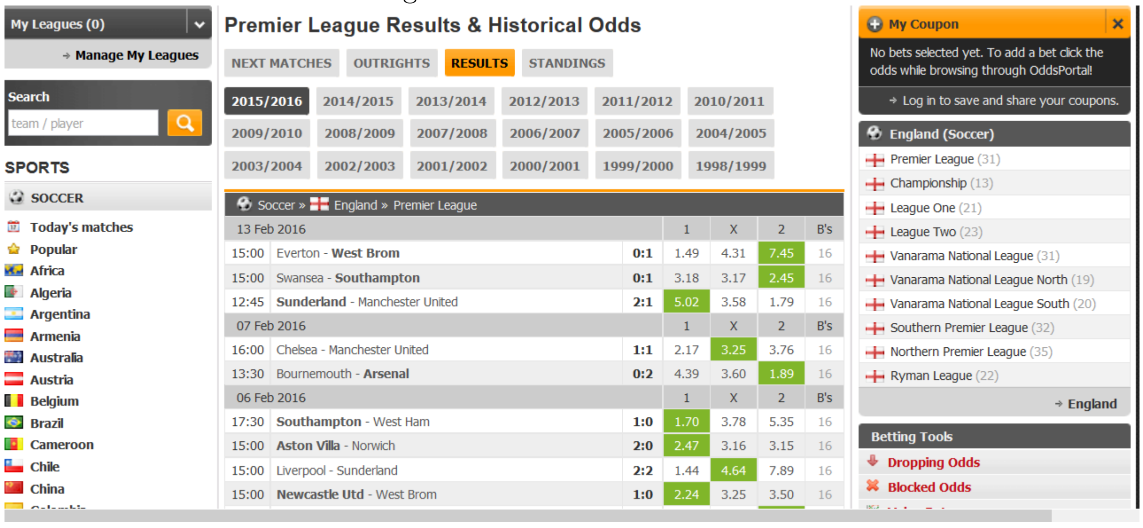
$$GA = \beta GAC + (1 - \beta) GAP$$

$$\lambda = \gamma GF + (1 - \gamma) GA$$

where GFC is the rate of goals scored by the team up to that point of the season the match is taking place in, GFP is the rate of goals scored by the team of the previous season, GAC is the rate of goals conceded by the opposing team up to that point the season, GAP is rate of goals conceded by the opposing team by the opposing team of the previous season and $0 \leq \gamma, \alpha, \beta \leq 1$. To find the best values of γ, α, β it iterated from 0 to 1 it steps of 0.1 to find the values that gave the best results over a set of training data.
Associated Files: `season_predictions.py`, `Match.py`

4.6 Odds Portal Data

Figure 3: Odds Portal Website



No bets selected yet. To add a bet click the odds while browsing through OddsPortal!

13 Feb 2016			1	X	2	B's
15:00	Everton - **West Brom**	**0:1**	1.49	4.31	7.45	16
15:00	Swansea - **Southampton**	**0:1**	3.18	3.17	2.45	16
12:45	**Sunderland** - Manchester United	**2:1**	5.02	3.58	1.79	16
07 Feb 2016			1	X	2	B's
16:00	Chelsea - Manchester United	**1:1**	2.17	3.25	3.76	16
13:30	Bournemouth - **Arsenal**	**0:2**	4.39	3.60	1.89	16
06 Feb 2016			1	X	2	B's
17:30	**Southampton** - West Ham	**1:0**	1.70	3.78	5.35	16
15:00	**Aston Villa** - Norwich	**2:0**	2.47	3.16	3.15	16
15:00	Liverpool - Sunderland	**2:2**	1.44	4.64	7.89	16
15:00	**Newcastle Utd** - West Brom	**1:0**	2.24	3.25	3.50	16

For information on the bookmakers and betting exchanges odds I took data from the website Odds Portal[15]. This website displays the odds that were offered on Premier League matches over a variety of seasons.

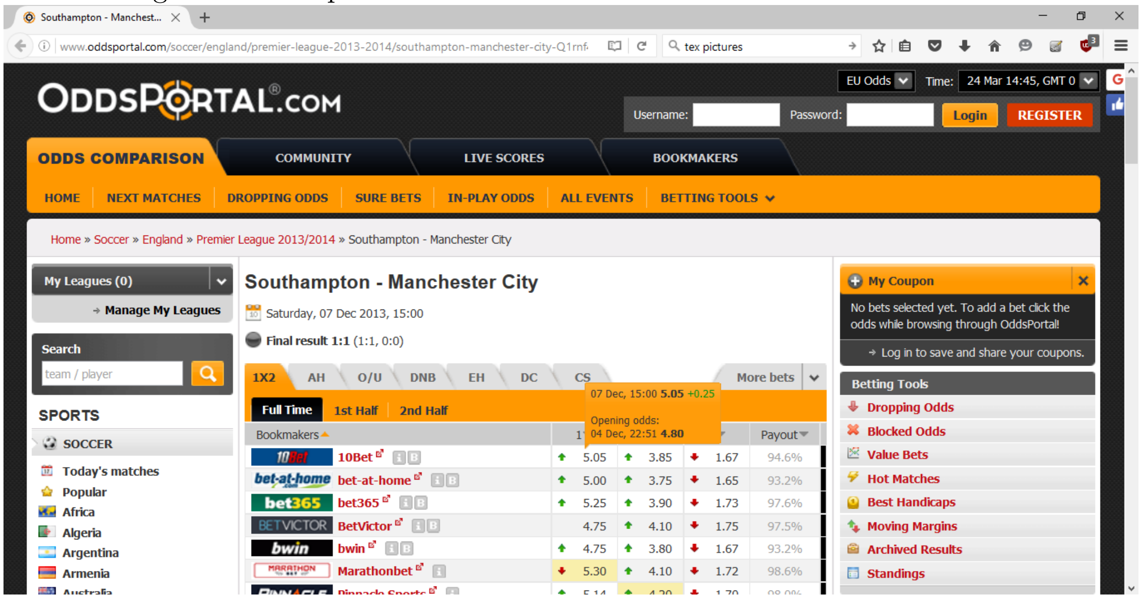
4.6.1 Aggregate Odds Portal Data

Odds portal takes an aggregate of the final odds from the bookmakers on website. These odds are displayed in a list of matches as seen in Figure 3. The odds on each page of Odds Portal are loaded after the rest of the page is loaded. Thus, the Selenium library was used to retrieve these web pages and wait until the page is fully loaded to download the HTML table of odds. Then Beautiful Soup was used to extract the important information from the HTML tables and the resulting information was stored in CSV files.
Associated Files: `odds_portal.py`

4.6.2 Opening and Closing Odds of individual bookmakers

When the betting begins, initially bookmakers will get set the odds by making a rough estimates of the probabilities of the matches. During the period when betting is ongoing, they modify the odds because of how much money is placed on each outcome. Therefore, the difference between the opening and closing odds could be significant in terms of the probabilities they produce and both the opening and closing odds would be of interest when comparing with the methods I have created for predictions. Taking the opening and closing odds of the bookmakers was a much more complex task than taking the aggregate data for the closing odds. There was no aggregate score for opening odds. To see them you

Figure 4: Example of the information Odds Portal has on each match



Home » Soccer » England » Premier League 2013/2014 » Southampton - Manchester City

Southampton - Manchester City

Saturday, 07 Dec 2013, 15:00

Final result 1:1 (1:1, 0:0)

07 Dec, 15:00 **5.05** +0.25

Opening odds:
04 Dec, 22:51 **4.80**

Bookmakers	1			Payout
10Bet	5.05	3.85	1.67	94.6%
bet-at-home	5.00	3.75	1.65	93.2%
bet365	5.25	3.90	1.73	97.6%
BetVictor	4.75	4.10	1.75	97.5%
bwin	4.75	3.80	1.67	93.2%
Marathonbet	5.30	4.10	1.72	98.6%

No bets selected yet. To add a bet click the odds while browsing through OddsPortal!

Log in to save and share your coupons.

have to click into each individual match page and then you have to hover over a particular bookmaker's closing odds to see the opening odds for that outcome. For example, if you wanted to find the opening odds of the bookmaker 10bet of a home win in the Southampton - Manchester City game, then you have hover over the cell in table associated with the closing odds for that element (See Figure 4). Again I used Selenium to scrap these odds. First, all the links to the match pages were scraped. Then for each link Selenium was used to open the corresponding web page and the hover action was mimicked for each of the closing odds in the table to produce the HTML element containing opening odds for that outcome. On the match pages, Odds Portal also had data on two betting exchanges, the Betfair exchange and the Matchbook exchange. However coverage for these was very patchy, as some matches had odds for just Matchbook, some had odds for both and some had odds for only Matchbook. I did manage to collect data for all matches, taking averages whenever both exchanges were present.

Associated Files: `odds_portal2.py`

Table 1: Best Machine Learning methods

ML method	Data Vector method	Total Bits	bits per match	% correct
Logistic Regression	Five	1366.85	1.381	53.33
Logistic Regression	Six	1366.92	1.381	54.55
Linear Discriminant Analysis	Six	1376.53	1.39	54.14
Linear Discriminant Analysis	Five	1376.7	1.391	53.54
SVC with linear kernels	Five	1390.42	1.404	54.04
Gradient Boosting	Six	1394.6	1.409	54.55
Gradient Boosting	Five	1395.55	1.41	52.63
SVC with rbf kernel	Five	1406.79	1.421	54.34
SVC with rbf kernel	Six	1410.44	1.425	53.03
Logistic Regression	One	1428.55	1.443	52.83

(SVC stands for support vector machine)

Table 2: Monte Carlo Methods

method	total bits	bits per match	% correct
Method 1	1448.48	1.463	50.1
Method 2	1446.1	1.461	50.5
Method 3	1447.82	1.462	50.7
Method 4	1444.51	1.459	50.8

5 Evaluation

My testing data for this project were the first 23 match days of 2015-16 season and all the matches of 2014-15 and 2013-14 seasons. There were 990 matches altogether in the testing set. To evaluate my methods and the bookmakers' odds I calculated the sum total of the Shannon Information Content (the total bits of information gained from a prediction) across all matches in the testing set. I also calculated percentage of correct predictions each method made (the outcome with the highest probability is considered to be the prediction) and also the average Shannon Information Content (bits per match) for each match in training set.

5.1 Machine Learning Methods

For full results on the machine learning methods see the ML_results.csv. Table 1 lists of the 10 best classifiers along with the type of data vector and machine learning method used to train the classifier. The data methods Five and Six were clearly the best data to use for the classifiers. This is unsurprising as these are the only two methods which contained up-to-date season information for the two teams. The other data methods used information from the seasons before the match was taking place and thus they did not have the most pertinent information relating to a team's current strength. The best machine learning method was the logistic regression classifier, followed closely by the Linear Discriminant Analysis classifier. Interestingly, the worse result out of all the machine learning classifiers was the Quadratic Discriminant Analysis classifier which used data Method Six. It averaged 71.01 bits of information per match. This was worse than the dummy classifier which averaged 55.5 bits per match by picking the most frequent outcome each time.

Table 3: Poisson method Results for the Test Data

γ	α	β	total bits	bits per match	% correct
0.6	0.9	0.9	1404.81	1.419	54
0.7	0.9	0.9	1405.09	1.419	53.3
0.5	0.9	0.9	1405.81	1.42	53.4
0.8	0.9	0.9	1406.64	1.421	53.1
0.6	0.8	0.9	1408.1	1.422	53.6
0.7	0.8	0.9	1407.66	1.422	53
0.8	0.8	0.9	1408.44	1.423	53
0.4	0.9	0.9	1408.08	1.422	52.5
0.5	0.8	0.9	1409.76	1.424	52.5
0.6	0.9	0.8	1409.7	1.424	54
0.9	0.9	0.9	1409.53	1.424	52.7
0.5	0.9	0.8	1410.2	1.424	53.3
0.7	0.7	0.9	1410.36	1.425	53.1
0.7	0.9	0.8	1410.34	1.425	54.1
0.8	0.7	0.9	1410.29	1.425	52.9
0.9	0.8	0.9	1410.47	1.425	52.7
0.6	0.7	0.9	1411.6	1.426	53.3
0.9	0.7	0.9	1411.43	1.426	52.5
0.3	0.9	0.9	1411.68	1.426	52

5.2 Monte Carlo methods

See table 2. These methods performed poorly compared to the machine learning techniques, with the best method, Method 4, achieving an average of 1.459 bits per match. This method uses a combination of goals scored and goals conceded from both the season the match is being played and the previous season.

5.3 Poisson Methods

I ran the Poisson against data from 2007-13 for many different values of α, β and γ to find the best settings of those parameters. See the file poisson_training.csv for the results this "training". Then I took the 10 best settings of the parameters and used them in the Poisson methods and ran them against the test data, as seen in table 3. It is interesting to note that the best settings for α and β were both 0.9 for the top four results of the Poisson method This means that goals rate in the current season matters significantly more than the goal rate in previous seasons.

5.4 Bookmakers and Betting Exchanges

From Odds Portal, I got the opening and closing odds from 6 bookmakers for all of the test data. I also got an aggregate of the bookmakers' closing odds on odds portal and a combination of the two exchanges, Betfair and Matchbook (This was not a complete combination as Betfair was missing a lot of odds and so was Matchbook, but together they covered all the test data). See table 4 for this data. Unsurprisingly, the opening odds were less accurate than the closing odds in terms of average number of bits gained, but the number of correct predictions made by the bookmakers were roughly the same. The best results were

Table 4: Bookmakers and Betting Exchanges Results

	Opening Odds			Closing Odds		
bookie	total bits	bits per match	% correct	total bits	bits per match	% correct
Exchanges	1431.42	1.446	53.8	1391.21	1.405	54
Pinnacle Sports	1396.66	1.411	53.8	1392.76	1.407	53.8
TonyBet	1399.27	1.413	54	1393.24	1.407	53.9
Unibet	1402.52	1.417	53.4	1394.37	1.408	53.8
bwin	1405.76	1.42	53.6	1392.09	1.406	53.9
Bet365	1402.37	1.417	53.9	1392.68	1.407	53.5
10Bet	1400.45	1.415	53.5	1391.94	1.406	53.9
Odds Portal Aggregate	N/A	N/A	N/A	1393.03	1.407	53.8

from the combination of exchanges' data, with an average of 1.405 bits per match. Pinnacle Sports had the best opening odds averaging 1.411 bits per match.

6 Conclusions

The best methods for this project offer a very powerful tool for the prediction of Premier League matches. The logistic regression classifier using the data Method Five was the best method for calculating probabilities with an average Shannon Information Content of 1.381 bits per match. This method outperformed the best result from the bookmakers and betting exchanges, which was the combination of odds from the two betting exchanges Betfair and Matchbook which achieved an average of 1.405 bits per match. In relative terms, this machine learning classifier performs only 1.7% times better then the exchange odds when we compare the average number of bits.

This project shows how powerful machine learning can be as a prediction tool. Using very general data I was able to construct a machine learning system which outperformed the Poisson and Monte Carlo methods of prediction as well as the predications derived from bookmakers' odds. However, it also showed how some methods of machine learning classification are unsuited to certain tasks. Algorithms such as K Nearest Neighbours and decision trees performed dismally in this task.

This system could have significant consequences in the area of sports prediction. It could be used as tool for bookmakers to generate accurate odds or be used by punters to win more money than they would have otherwise.

All methods I have created are very generalizable. All they use as data are the simple league table statistics such as league table position and goals scored. There are analogous pieces of data for any other sports leagues, such those in Rugby, Basketball or American football.

A risk to this system is the fact the bookmakers are continuously trying to improve their odds every day, and have large resources behind them to do so. Therefore, if there is no continuous development of the system, it may fare worse at predication in comparison to the bookmakers in the future.

There is a lot more that could be investigated in this topic. For the machine learning methods, other statistics could be used as data in the machine learning classifiers. For example, you could look at data for the individual players of the team. For the Monte Carlo simulations, you could try to make the simulation more realistic by adding in data such as how many shots on target a team has in a match or how much possession of the ball each team has during a match.

References

[1] B. O'Halloran. (2016, Jan. 16). "Paddy Power merger with Betfair clears final hurdle", *Irish Times* [Online]. Available: [http://www.irishtimes.com/business/retail-and-services/paddy-power-merger-with-betfair-clears-final-hurdle-1.2498353, accessed on (2016, Mar. 26)]

[2] S. Ebner. (2013, July 2)."History and time are key to power of football, says Premier League chief", *The Times*[Online]. Available: [http://www.thetimes.co.uk/tto/public/ceo-summit/article3804923.ece, accessed on (2016, Mar. 28)]

[3] A. Joseph, N.E. Fenton, M. Neil,(2005,April 21) "Predicting football results using Bayesian nets and other machine learning techniques",Computer Science Department, Queen Mary, University of London, UK

[4] Byungho Min, Jinhyuck Kim, Chongyoun Choe, Hyeonsang Eom, Robert Ian (Bob) McKay,"A Compound Framework for Sports Prediction:The Case Study of Football", School of Computer Science and Engineering, Seoul National University, Seoul, Korea

[5] Dragan Miljkovi, Ljubia Gaji, Aleksandar Kovaevi, Zora Konjovi,"The Use of Data Mining for Basketball Matches Outcomes Prediction",Faculty of Tehnical Sciences, Novi Sad, Republic of Serbia

[6] Ryan Baird, "Probalilisitc Sports Prediction Using Machine Learing", School of Computer Science and Software Engineering,Monash University

[7] http://www.bettingexpert.com/how-to/convert-odds

[8] David J.C. MacKay, "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, http://www.inference.phy.cam.ac.uk/itprnn/book.pdf, pp. 32

[9] Richard D. De Veaux, Paul F. Velleman, David E. Brock, "Stats: Data and Models", Third Edition, Pearson Education, Inc

[10] http://selenium-python.readthedocs.org/getting-started.html

[11] http://www.crummy.com/software/BeautifulSoup/bs4/doc/

[12] Tom M. Mitchell, "Machine Learning", McGraw-Hill,ISBN 0070428077

[13] http://www.football-data.co.uk/

[14] http://www.premierleague.com/en-gb/players/ea-sports-player-performance-index.html

[15] http://www.oddsportal.com/

[16] http://scikit-learn.org/stable/supervised_learning.html#supervised-learning