Beating the Bookies: Predicting the Outcome of Soccer Games

Steffen Smolka SMOLKA@STANFORD.EDU
Stanford University, USA

1. Introduction

Soccer is the most popular sport in the world. With an estimated 3.5 billion fans around the world, it is enjoyed by nearly half the world's population. My project asks the following question: **Is it possible to predict the outcome of soccer games with high accuracy automatically?** This questions is not merely of interest to fans who must satisfy their curiosity, but it is also of significant economical relevance. A BBC article from 2013 (BBC) estimates that the soccer betting market is worth between 500 and 700 billion USD a year.

History has shown again and again that soccer prediction can be hard. Leicester City stunned the world in 2016 when winning the English Premier league season 2015/16 after just getting promoted from the second league the year before. Figure 1 shows that this came as a complete surprise not only to fans, but also the major betting companies: Ladbrokes estimated the chance of this event to 1 in 5000 before the season. In the very same year, Portugal surprisingly won the Euro Cup.

This project considers soccer game prediction as a ternary classification problem. The goal is to predict the outcome $\mathbf{y}^{(i)} \in \{+1, 0, -1\}$ of a game i given some feature vector $\mathbf{x}^{(i)}$. Here the values $+1$, 0 and -1 encode the three possible outcomes *Home Win*, *Draw*, and *Away Win*, respectively. I focus on games of the English Premier League (EPL)—the most popular soccer league in the world—and take a minimalistic approach: for a game i between home team $\mathbf{h}^{(i)}$ and away team $\mathbf{a}^{(i)}$, I derive features $\mathbf{x}^{(i)}$ using nothing but the final scores of previous games involving either of the teams. I do *not* include the identities of $\mathbf{h}^{(i)}$ and $\mathbf{a}^{(i)}$ as features, deciding instead to base forecasts purely on performance.

2. Related Work

Previous work in this area can be divided into goal-based and result-based approaches. While the former try to predict the goals scored and conceded by each team, the latter predict the win-draw-loose outcome directly. My own work falls into the latter category. Goddard (Goddard, 2005) compares the two approaches and concludes they are of similar predictive power, but suggest that hybrid approaches may perform best.

The works of Rue and Salvesen (Rue & Salvesen, 2000) and Karlis et. al (Karlis & Ntzoufras, 2003) are goal-based, using Poisson distributions to model the number of goals scored by a team. Similar to my work, they use attack and defense parameters; but in contrast to my work, their models are team-dependent. Unfortunately, neither of the papers report how many outcomes they can predict correctly. In particular, the focus in (Karlis & Ntzoufras, 2003) is mainly on how well the model can fit existing data.

The PhD thesis of Constantinou (Constantinou et al., 2012) is result-based and uses a Bayesian network based on team strength, form, psychologial impact, and fatigue. The thesis describes a model based purely on objective data, and a model that incorporates subjective estimates from a human expert. It concludes that the accuracy of the purely objective forecasts is significantly inferior to bookmakers forecasts, while the subjective model is on par.

(Joseph et al., 2006) is also result-based but take a very different approach: it compares the performance of an expert-constructed Bayesian network to the performance of several models trained by machine learning algorithms. They conclude that the expert BN is generally superior to the automatic techniques.

3. Data Set and Computed Features

Although there seems to be an abundance of data on soccer games available online at first look, obtaining a data set amenable to machine learning proved challenging. The public data sets I identified in my project proposal (EUS; His) are either incomplete and erroneous, or contain only the most basic information: a list of games with their final scores. While very detailed databases of soccer statistics exist, they are maintained by commercial providers such as opta[1], and access is granted only at high charges.

After spending a significant amount of time looking for data sets with sophisticated features, I had to give up on my initial idea to use information such as the freshness of a

[1] `http://www.optasports.com/`

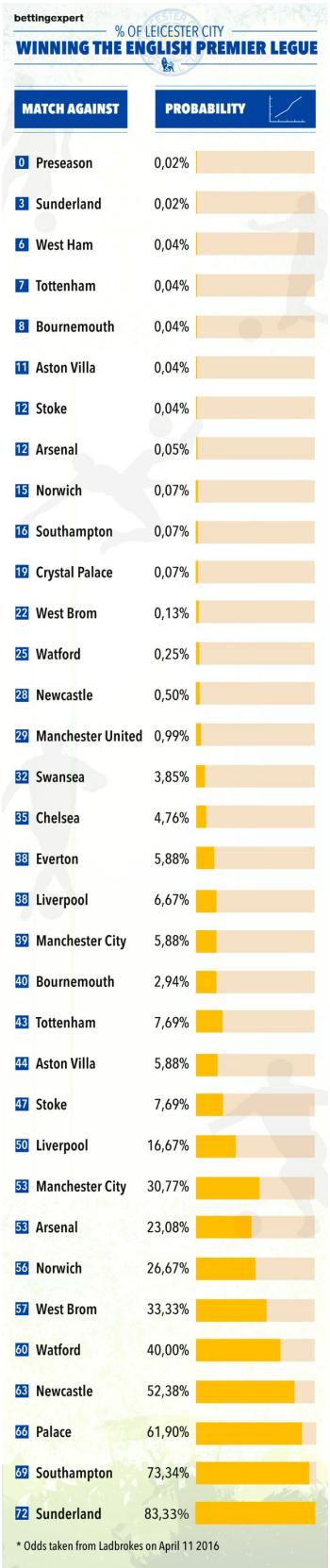

Figure 1. Leicester's chance of winning the Premier League season 2015/16, as predicted by major betting company Ladbrokes.

team (measured by the time since the last game), the number of injured players, or the value of a team's squad. Instead I use only features derived from the final scores of a team's previous games. I extracted this raw data from csv files provided by (His) that contain the final results of over 6500 Premier League games between 1993 and 2016.

A first attempt. As a first attempt, I computed the following features, both for the home team and the away team, for each game:

- #goals scored in each of the last w home games
- #goals scored in each of the last w away games
- #goals conceived in each of the last w home games
- #goals conceived in each of the last w away games

Using a windows size of $w = 3$, this yields $8w$ features for each game (discarding the initial games for which no data is available). Training a SVM using these features, I initially obtained a model with extremely high accuracy. Unfortunately, this turned out to be due to a poor validation approach: I trained the model on the first 85% of the games of a single season, and then tested it against the last 15% of the season. I think this may suggest that the last games of a season are much more predictable than other games.

After moving to 10-fold cross validation and using the games of all 23 seasons, I saw very poor predictive performance using these features. I thus moved to a different set of features, which I will describe next.

3.1. Form coefficients

The idea behind my final set of features was to compile the previous results of a team into performance coefficients that are meant to measure *general team form*, *offensive form*, and *defensive form*. All coefficient are initialized to 1 at the beginning of the season, and then get updated based on performance, so that a higher coefficient indicates better form. When updating the coefficients, we take the current form of the opponent team into account: beating a strong opponent gives more points than beating a weak opponent.

For illustration, let us define the *general form* coefficient formally. Associated with each team t and time τ is a score s_t^τ. The time τ indicates the number of games played by team t in the current season. Initially we set

$$s_t^0 = 1 \quad (\forall t) \tag{1}$$

Now suppose that at time $\tau > 0$, team t beats team u. Then the coefficients of the teams get updated as follows:

$$s_t^\tau = s_t^{\tau-1} + \gamma s_u^{\tau-1} \tag{2}$$

$$s_u^\tau = s_u^{\tau-1} - \gamma s_u^{\tau-1} \tag{3}$$

Intuitively, team t "steals" a fraction $0 < \gamma < 1$ of team u's score. Since this fraction is proportional to u's score, beating a good team (with a high score) gives more points than beating a weak team (with a low score). The parameter γ can be understood as a discount factor. It controls how quickly the fitness score reacts to changes in performance: if τ is close to 1, then the fitness score captures mostly the performance in very recent games (and a single lost game will lead to a fitness score close to 0); if on the other hand γ is close to 0, then the fitness score changes only slowly between games and it captures the long term performance of a team.

If team t and team u tie at time τ, we update the coefficients as follows:

$$s_t^\tau = s_t^{\tau-1} - \gamma(s_t^{\tau-1} - s_u^{\tau-1}) \quad (4)$$

$$s_u^\tau = s_u^{\tau-1} - \gamma(s_t^{\tau-1} - s_u^{\tau-1}) \quad (5)$$

That is, the coefficient of the stronger team will decrease, and the coefficient of the weaker team will increase. The rate of change is proportional to the difference in strength between the teams, so that tying against a much stronger team gives more points than tying against a team of similar strength. Importantly, the two coefficients approach equal values in the limit. This makes intuitive sense, since teams that repeatedly tie against each other are likely to be of similar strength.

Note that the coefficients are always *non-negative* and that they are always *normalized* in the following sense:

$$\sum_t s_t^\tau = T \quad (\forall \tau) \quad (6)$$

where T denotes the number of teams. This follows directly from equations (1)-(5) and also implies that

$$0 \leq s_t^\tau \leq T \quad (7)$$

Empirically, I observed that invariants (6) and (7) are crucial for the performance of my models. In reflection, this is obvious: without the invariants, the score s_t^τ would measure only *relative fitness*, i.e. its value would be completely meaningless without comparing it to a second score s_u^τ of a different team at the same time. But with the invariant, s_t^τ turns into an *absolute* fitness measure that is meaningful in itself:

Fact 1 *If a team has fitness score $n = s_t^\tau$, then it is in the top $\lfloor \frac{T}{n} \rfloor$-quantile at time τ.*

Similar to the *general fitness* coefficient just discussed, I also defined *offensive fitness* and *defensive fitness* coefficients that capture how likely a team is to score or conceit a goal. They are calculated not based on the outcome of a game, but rather on the number of goals scored by each side in a game: scoring against a defensively-strong team is interpreted as a sign of a strong attack, and conversely keeping a clean-sheet against an offensively-strong team is a sign of a strong defense. Interestingly, invariants (6) and (7) have to be slightly updated: now the offensive *and* defensive coefficients together sum up to $2T$, accounting for the possibility that there may be no goals (or no clean-sheets) at all during a season.

Feature Vector. My final feature vector was established through experimentation. It includes two instantiations of the *general form* coefficient with discount factors $\gamma = \frac{1}{7}$ and $\gamma = \frac{1}{3}$, respectively, capturing both high frequency events and low frequency events, i.e. long term form and recent form. Additionally I include the *offensive* and *defensive* form coefficients with $\gamma = \frac{1}{7}$.

4. Methods.

Models. I used three models: a self-implemented SVM, using stochastic gradient descent for training; a libsvm(Chang & Lin, 2011) implementation of SVM; and a scikit-learn implementation of a neural network. For the SVMs, experimentation established the radial basis kernel as best-suited. The parameters for the latter two models were optimized using grid search and 10-fold cross validation; the parameters of the first model were hand-tuned.

Binary Classification. Besides the ternary classification problem, I also considered a simplified classification problem by discarding all games that ended in a tie. Formally, this can be understood as finding a model for the binary conditional random variable

$$y|x, y \neq 0$$

instead of the ternary random variable

$$y|x$$

Obviously, the binary classification problem is much easier than the ternary classification problem: random guessing already gives 50% accuracy in the binary case, but only 33% accuracy in the binary case.

Multinomial Classification. Both the neural network and the SVM libraries supported multinomial classification out of the box. The neural network uses soft-max function to achieve this. libsvm uses what is called the *one-vs-one* reduction: it trains one binary classifier for each pair of class labels. At prediction time, each of the classifiers is querried and the class with the highest number of "votes" wins.

For my custom SVM implementation, I used an adhoc technique based on the idea that ties occur in games in which

samples	% win	% tie	% loss	% predicted
34	73.5	0.0	26.5	82.0

Table 1. test set with wins and losses only – ties filtered out

samples	% win	% tie	% loss	% predicted
48	52.1	33.3	14.6	72.0

Table 2. test set with home wins, ties, and home losses

neither a win nor a loss is very likely. Although the SVM model does not output probabilities, we can interpret the distance of a sample from the separating hyperplane as the confidence of the model: games far on one side are likely wins, games far on the the other side are likely losses. Games somewhat in the middle, i.e. within some threshold δ of the hyperplane, can then be interpreted as ties. Some experimentation established $\delta = 0.25$ as a reasonable threshold.

5. Experiments and Results

Initial Results. As mentioned in Section 2, I initially achieved very high accuracies using the features described under "A first attempt." I trained a custom SVM with RBF kernel using stochastic gradient descent. I used the data from season 2016-17 only, containing 320 games (32 for each of the 20 teams) after discarding the first games as described in the Section 2. I divided the samples into training and test data using a 85%/15% split. Since the models were fitted using stochastic gradient descent, Tables 1 and 2 report the *average* prediction rate over 20 iterations.

Note that I did not split the data set randomly into training and test set, but instead used the first 85% games of the season for training and then the last 15% games for testing. The same model achieved quite poor performance when moving to larger data sets and 10-fold cross validation. In fact the performance was so bad that I started investigating better features, eventually designing the fitness coefficients described in Section 3.1.

Nonetheless these results are interesting: they suggest that maybe the last games of a season are significantly easier to predict than other games.

Rigorous Results. To establish higher confidence in my results, I moved to 10-folded cross validation (where the 10 sets are chosen uniformly at random). Unfortunately, my initial features performed very poorly in this setting, sometimes predicting games less accurately than the naive strategy that always predicts a home win (the most likely event). I was able to improve performance dramatically by using the fitness coefficients described in Section 3.1 as features. To my surprise, all three models were quite sensitive to adding "bad" features.

	training acc.	10-fold cv acc.
home wins†	62%	62%
libsvm	67%	65%
custom SVM	90%	67%
neural net	85%	**73%**

Table 3. Average accuracy for binary classification.

	training acc.	10-fold cv acc.
home wins†	46%	46%
libsvm	49%	48%
custom SVM	87%	**53%**
neural net	58%	50%

Table 4. Average accuracy for ternary classification.

The accuracy (i.e., average prediction rate) of the three models for the binary and the ternary classification problems are shown in Tables 3 and 4. The neural network performed best in the binary case, predicting an average 72% of games in the test sets correctly. This is 9% better than naively always predicting a win, but also %10 worse than the result from my initial experiment (Tabel 1). Libsvm was only able to achieve a 3% edge over the naive strategy, my custom SVM achieved a 5% edge.

In the ternary case, the custom SVM performed best, predicting an average of 53% of games (in the tests sets) correctly. This is 7% better than always predicting a home win. On the training set, the custom SVM achieved 87% accuracy, suggesting some overfitting.

Remarkably, my custom SVM implementation performed better than libsvm in both cases. This might be due to the fact that my implementation used a randomized algorithm, namely stochastic gradient descent, while libsvm uses a deterministic algorithm.

6. Conclusion

My project demonstrates that it is possible to predict the outcome of soccer games—win, tie, or loss—with over 50% accuracy automatically. There is lots of room for improvement. My project used nothing but the outcome of previous games for prediction, and I expect more sophisticated features such as the fatigue of a team can enable predictions with an accuracy beyond 60%. It would be interesting to combine my result-based model with a goal-based model (using Poisson distributions, for example) to obtain an even more accurate hybrid model, as suggested by (Goddard, 2005).

Another direction for future work is to investigate algorithms

that can place bets based on the models' predictions and some measure of confidence. My data set contains historical betting odds that can be used for evaluating if such a system is able to generate profits.

I also discovered that it seems to be much easier to predict the last games of a season. In particular, I achieved over 72% accuracy for predicting the final games of the 2015/16 EPL season. This deserves further investigation, and my have important implications for placing bets.

References

Football betting - the global gambling industry worth billions. http://www.bbc.com/sport/football/24354124. Accessed: 2016-12-15. Published: 2016-10-3.

European soccer database. https://www.kaggle.com/hugomathien/soccer. Accessed: 2016-10-21.

Historical football results and betting odds data. http://www.football-data.co.uk/data.php. Accessed: 2016-10-22.

Chang, Chih-Chung and Lin, Chih-Jen. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

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

Goddard, John. Regression models for forecasting goals and match results in association football. *International Journal of forecasting*, 21(2):331–340, 2005.

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

Karlis, Dimitris and Ntzoufras, Ioannis. Analysis of sports data by using bivariate poisson models. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 52 (3):381–393, 2003.

Rue, Havard and Salvesen, Oyvind. Prediction and retrospective analysis of soccer matches in a league. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 49(3):399–418, 2000.