Predicting Soccer Match Results in the English Premier League

Ben Ulmer
School of Computer Science
Stanford University
Email: ulmerb@stanford.edu

Matthew Fernandez
School of Computer Science
Stanford University
Email: matthew0@stanford.edu

***Abstract*—In this report, we predict the results of soccer matches in the English Premier League (EPL) using artificial intelligence and machine learning algorithms. From historical data we created a feature set that includes gameday data and current team performance (form). Using our feature data we created five different classifiers: Linear from stochastic gradient descent, Naive Bayes, Hidden Markov Model, Support Vector Machine (SVM), and Random Forest. Our prediction is in one of three classes for each game: win, draw, or loss. Our best error rates were with our Linear classifier (.48), Random Forest (.50), and SVM (.50). Our error analysis focused on improving hyperparameters and class imbalance, which we approached using grid searches and ROC curve analysis respectively.**

I. Introduction

There were many displays of genius during the 2010 World Cup, ranging from Andrew Iniesta to Thomas Muller, but none were as unusual as that of Paul the Octopus. This sea dweller correctly chose the winner of a match all eight times that he was tested. This accuracy contrasts sharply with one of our team member's predictions for the World Cup, who was correct only about half the time. Due to love of the game, and partly from the shame of being outdone by an octopus, we have decided to attempt to predict the outcomes of soccer matches. This has real world applications for gambling, coaching improvements, and journalism. Out of the many leagues we could have chosen, we decided upon the English Premier League (EPL), which is the world's most watched league with a TV audience of 4.7 billion people [1]. In our project, we will discuss prior works before analyzing feature selection, discussing performance of various models, and analyzing our results.

A. *Related Literature*

Most of the work on this task has been done by gambling organizations for the benefit of oddsmakers. However, because our data source is public, several other groups have taken to predicting games as well. One example of the many that we examined comes from a CS229 Final Project from Autumn 2013 by Timmaraju et al. [2]. Their work focused on building a highly accurate system to be trained with one season and tested with one season of data. Because they were able to use features such as corner kicks and shots in previous games, as well as because their parameters were affected by such small datasets, their accuracies rose to 60% with an RBF-SVM. We chose to focus on a much larger training set with the focus on building a more broadly applicable classifier for the EPL.

Another attempt at predicting soccer works was done by Joseph et al. This group used Bayesian Nets to predict the results of Tottenham Hotspur over the period of 1995-1997. Their model falls short in several regards: self-admittedly, it relies upon trends from a specific time period and is not extendable to later seasons, and they report vast variations in accuracy, ranging between 38% and 59%. However, they do provide useful background on model comparison and feature selection [3]. While we took into account their insight on these areas, we remained true to our philosophy of generality and long-term relevance.

Finally, we were influenced by the work of Rue et al., who used a Bayesian linear model to predict soccer results. Notably, they used a time-dependent model that took into account the relative strength of attack and defense of each team [4]. While we didn't have data available to us regarding statistics for each individual player, we did take into account the notion of time-dependence in our model.

II. Dataset

Our training dataset contained the results of 10 seasons (from the 2002-03 season to the 2011-12 season) of the EPL, and our testing dataset included the results of 2 seasons (the 2012-13 season and the 2013-14 season) of the EPL. As each of the 20 teams plays all other teams twice per season, this translates to 3800 games in our training set and 760 games in our testing set. For each game, our dataset included the home team, the away team, the score, the winner, and the number of goals for each team. We obtained this data from an England based group that records historical data about soccer matches called Football-Data and preprocessed this data with python [6].

III. Methodology

A. *Initial Challenges*

We ran into several challenges over the course to the project. Two were the most significant: a lack of data, and the randomness of the data. Since our data spanned a gap of 12 years, there was always going to be a difference in the amount of information available for the 2002 season and the 2013 season. As such, we were limited to the features from the

2002 season (described above). In our review of literature and the current leading predictors however, we saw use of features such as availability of specific players and statistics such as shots on goal. The predictors used by gambling agencies had the resources to keep track of or research injuries; we did not. Another challenge we faced was the randomness of the data. For example, in the 2010 premier league season, 29% of the games were draws, 35.5% were wins, and 35.5% were losses [7]. If we calculate the entropy (a measure of randomness), of these statistics we see an Entropy value of .72:

$$Entropy = -\left(.29 * log_3(.29) + 2(.355 * log_3(.355))\right) \quad (1)$$
$$= .72 \quad (2)$$

This is close to an entropy value of 1, which corresponds to pure randomness. This spread makes the data hard to classify based on a MAP estimation, and makes it hard for classifiers to discount or ignore a class. Moreover, not only was the data distributed evenly, but in soccer, there are a large amount of upsets. It is not uncommon to see teams like Wigan triumph against teams like Manchester City. Famously, in 2012, an amateur side (Luton Town) beat EPL team Norwich. In a sport where non-league teams can overcome teams of the highest league, the amount of upsets at the top level will be relatively high. With many outliers such as these, the results become hard to predict.

B. Feature Selection

Many features make sense to include. Literature and our own intuition suggested using the features of whether a team is home or away, features for each team (similar to an Elo ranking for each team), and form (results in recent matches). We used form to be a measure of the "streakiness" of a team; for example, if a team is on a hot streak, it is likely that they will continue that hot streak. However, we ran into several issues with calculating form. First of all, what to do for the first x games of the season for which there are not enough prior games to calculate form. We toyed with two approaches: scaling the data and leaving out the games. To scale the data, we calculated the form based on the results of the matches that had already been played, and multiplied it by $7/x$. However, we found that the method of ignoring the first 7 games led to better results, and, more importantly, allowed us to use the feature of having time dependent weights for each game played (separate weights for the last game, the second to last game, the third to last game, and so on). We then had to decide how many games to aggregate form over. We analyzed using between 2 and 8 games and the results are shown in Figure 1.

After testing for various numbers of previous games to consider for our form feature, we found that the optimal accuracy came from calculating form over the past 4 games for the RBF-SVM model shown in Figure 1. For every other model we found the optimal number to be 7 games, so we used that consistently across all models. Finally, we occasionally used the feature of goal differential, but found it often led to overfitting. See model descriptions below to see where we used this feature.

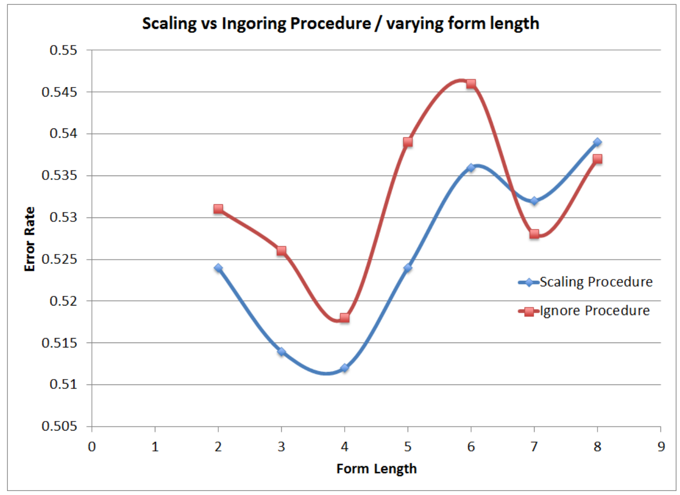

Figure 1. Error rate as a function of form length

C. Models

We tried a variety of methods to address our three-class classification problem:

1) Baseline: We began our project by reducing the problem from a 3-class classification problem to a 2-class classification problem, and predicted whether a team would win or not win by implementing and training our own stochastic gradient descent algorithm. For this preliminary test, we used as features whether a team is home or away and the form for that team. This method resulted in a test error of .38 and a train error of .34. We then extended this model by implementing a 3-class classification one-vs-all stochastic gradient descent algorithm on the same set of features. This model achieved a training error of .39 and a testing error of .60.

2) Naive Bayes: We also implemented a naive bayes classifier. We didn't expect this model to achieve much success, as its assumption of independence does not hold on the often "streaky" results that are prevalent in soccer matches. As features, we used the home team's form, the away team's form, whether a team is home or away, and ratings for each team. Our gaussian naive bayes had a training error of .52 and a testing error of .56, while our multinomial naive bayes had a training error of .51 and a testing error of .54. As expected, these models did not seem to be well suited for this dataset, although they did perform better than our baseline.

3) Hidden Markov Model: Due to the time dependence of this problem, we tried a hidden markov model. We treated the result of the match (win, loss, draw) as the unobserved states and past results as the evidence. This model had a training error of .52 and a test error of .56. The lack of success here can be explained by the assumption of the model that past states are hidden, while in reality, those states are known.

4) Support Vector Machines: Literature suggested that the most successful model for this task was an SVM, so we began by implementing one with an RBF (gaussian kernel):

$$K(x, z) = e^{||x-z||^2/2\sigma^2} \quad (3)$$

We used features of the home team's form, the away team's form, whether a team is home or away, and ratings for each team. We tuned the hyperparameters using a grid search with k-folds cross validation (we used a k-value of 5). Figure 2 shows the largest range and Figures 3 and 4 are zoomed regions. In Figure 2 our grid ranges are $C = [.01 - 3]$ and Gamma $= [0 - 1]$. In Figure 3, we focus on $C = [.25 - .75]$ and Gamma $= [0.05 - .02]$. Lastly in Figure 4, we find the optimal values of $C = .24$ and Gamma $= .15$

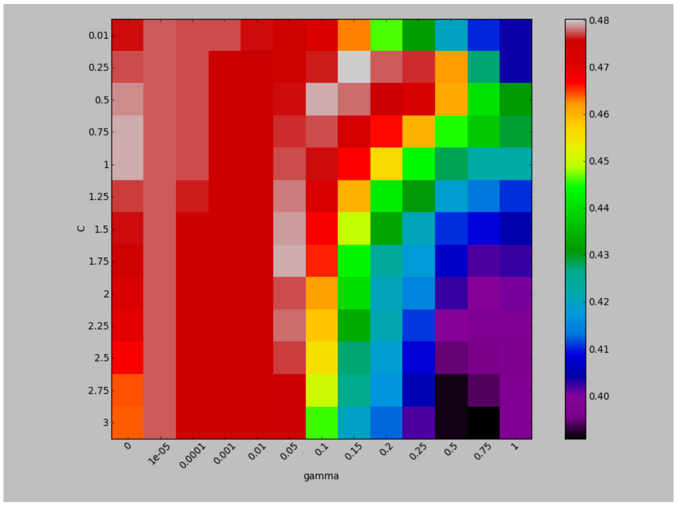

Figure 2. RBF-SVM Grid Search for hyperparameters of C and gamma

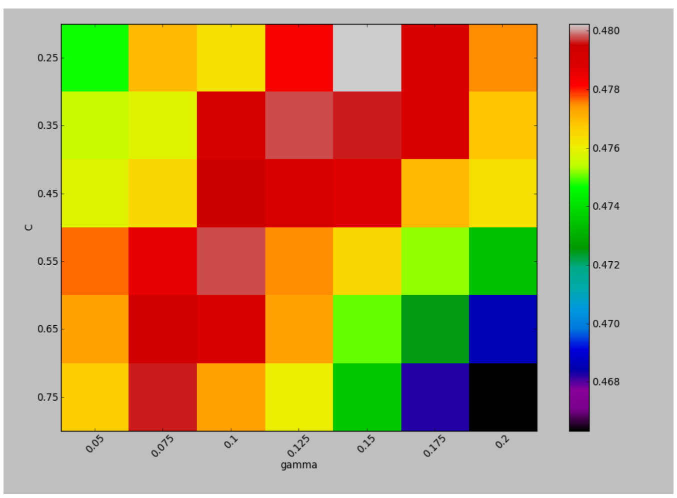

Figure 3. RBF-SVM Grid Search for hyperparameters of C and gamma

After optimizing our SVM, we achieved a training error of .21 and a test error of .5. We experimented with adding additional features such as goal differential and separate weights for each of the last seven games, but we found that this only exacerbated our overfitting issue (at one point, we had a difference of .5 between our training error and our testing error). To address our overfitting issue, we attempted a less complicated decision boundary, and used a gram kernel:

$$K(x, z) = x^\top z \tag{4}$$

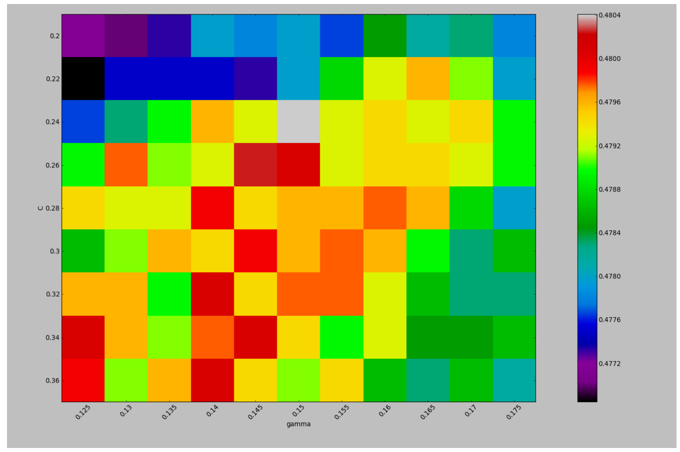

Figure 4. RBF-SVM Grid Search for hyperparameters of C and gamma

This kernel allowed us to use goal differential and more complicated form features in addition to what we had used previously, and we achieved test error of .49 and train error of .5.

5) Random Forest: We also tried an ensemble method, as the upsets in the data means that many results aren't entirely representative of the matches we are trying to predict. We used the same set of features as the SVM above, and tuned the hyperparameters with a grid search. This is shown in Figure 5 where we found the best hyperparameters were 110 for the number of estimators and minimum of 2 examples required to split an internal node.

Our Random Forest model achieved a test error rate that ranged between .49 and .51 (due to random nature of the aptly named random forest, we would get different error rates on each run), but averaged at .50. Similarly, our training error was between .04 and .05, but averaged at .045.

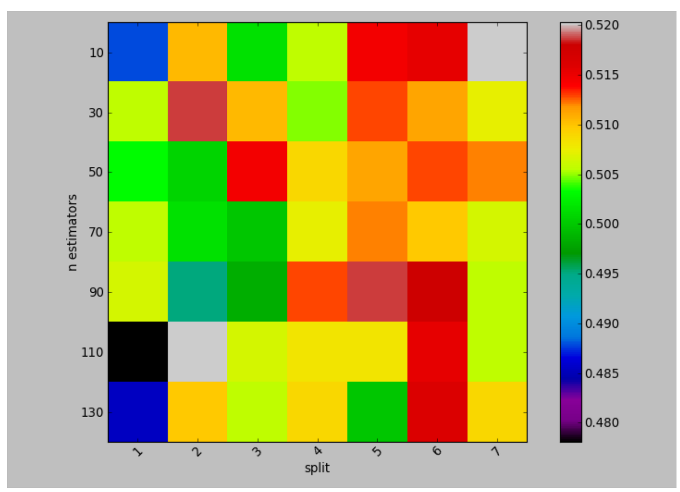

Figure 5. Random Forest error rate as a function of the number of estimators and the minimum number of examples required to split an internal node

6) One-vs-all Stochastic Gradient Descent: Surprisingly, after we updated our features, we found that a one-vs-all algorithm that used stochastic gradient descent to generate scores performed well with a test error of .48 and a training error of .5. Similar to the SVM with the gram kernel, the simplicity of the decision boundary stopped the overfitting which was a problem in models such as an SVM with an RBF kernel.

IV. Results

We implemented and tuned 8 different models. Those models and respective error rates are illustrated in Figure 6.

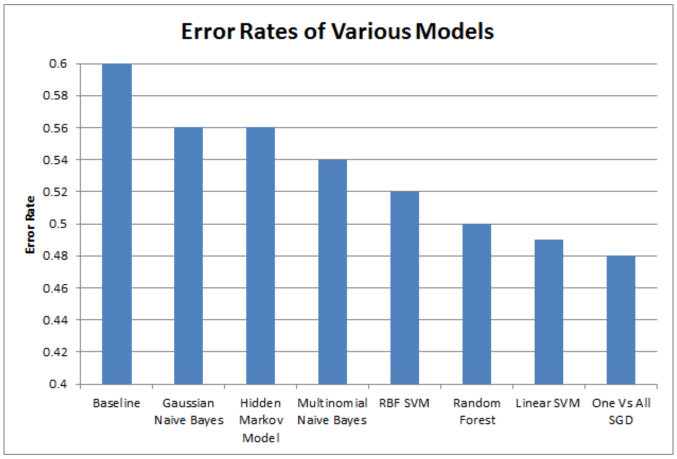

Figure 6. A comparison of error rates between models

We will focus on our best performing models, which were an SVM with an RBF kernel, an SVM with a linear kernel, a Random Forest model, and a one-vs-all SGD model. These models achieved error rates of between .48 and .52, which is comparable with the error rate of .48 for leading BBC soccer analyst Mark Lawrenson, but still worse than an "oracle" of .45 for the betting organization Pinnacle Sports [5].

Out of our models with linear decision boundaries, we found that the best model was a one-vs-all linear classifier created with our stochastic gradient descent algorithm. However, both of the linear models drastically under-predicted draws. The confusion matrices for these two models are shown in Tables 1 and 2. Our results are expected for this type of system. Their unweighted average recall values are hindered, but weighted accuracy is higher because win and loss classes have more examples than the draw class.

	Predict Draw	Predict Win	Predict Loss
Actual Draw	3	91	200
Actual Win	12	249	208
Actual Loss	12	68	389

Table I
Confusion Matrix for One-vs-all Stochastic Gradient Descent

Moreover, our one-vs-all SGD model in particular was highly dependent on training set size. When we reduced the number of seasons to 9, 7, 5, and 3 our model reported test error of .49, .51, .52, and .51 respectively. The poor performance on draws and dependence on training set size suggests that, although the one-vs-all SGD model reported the highest accuracy for this data set and training set, it would not be easily extendable for real world applications such as gambling. In contrast with the linear classifiers, our Random Forest Model and SVM with an RBF kernel had better accuracy on the "draw" class. The confusion matrices for these models are shown in Tables 3 and 4.

	Predict Draw	Predict Win	Predict Loss
Actual Draw	0	142	152
Actual Win	0	307	162
Actual Loss	0	152	317

Table II
Confusion Matrix for Linear SVM

	Predict Draw	Predict Win	Predict Loss
Actual Draw	12	142	140
Actual Win	13	302	154
Actual Loss	23	134	312

Table III
Confusion Matrix for Random Forest

	Predict Draw	Predict Win	Predict Loss
Actual Draw	42	126	126
Actual Win	34	275	160
Actual Loss	40	136	293

Table IV
Confusion Matrix for RBF SVM

To add to our understanding of the performance of an RBF-SVM and the Random Forest Model, we implemented a receiver operating characteristic curve (ROC curve) to compare the two. While ROC curves are normally used for binary classification problems, we adapted them to show pairwise comparison (one class vs. all other classes) in order to fit our 3-class classification problem. From the Figures 7 and 8 below, it is evident from the area underneath the draw curve (class 1) that the SVM with an RBF kernel outperforms the Random Forest model in terms of predicting draws, although it itself is outperformed in terms of predicting wins and draws as well as overall performance (the micro-average curve).

Overall, we found that every model under-predicted draws to some degree, which is in accordance with the models presented in literature on the subject. We attempted to weight this class more, but found that the trade off in accuracy for wins and losses was detrimental. Timmaraju et al. were only able to report a recall value of .23 [2]. The failure to predict draws stems from the fact that draws are the least likely result to occur (even though draws occur only 6% less than wins or losses, this is enough to affect our predictor) [7].The models that were more accurate in terms of predicting draw results sacrifice accuracy on wins and losses.

While our project met with success comparable to expert human analysts, it still lags behind leading industry methods. This is in part due to the availability of information; we were unable to use as features statistics for players and elements of

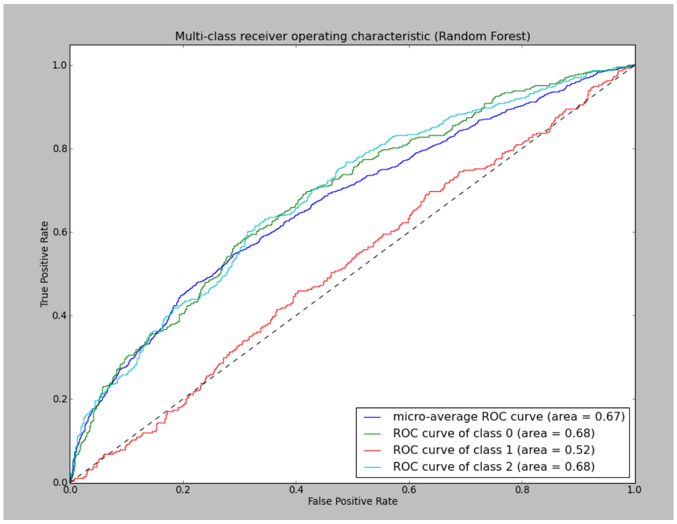

Figure 7. A receiver operating characteristic curve of classes 0 (win), 1, (draw), and 2(loss) from an Random Forest model

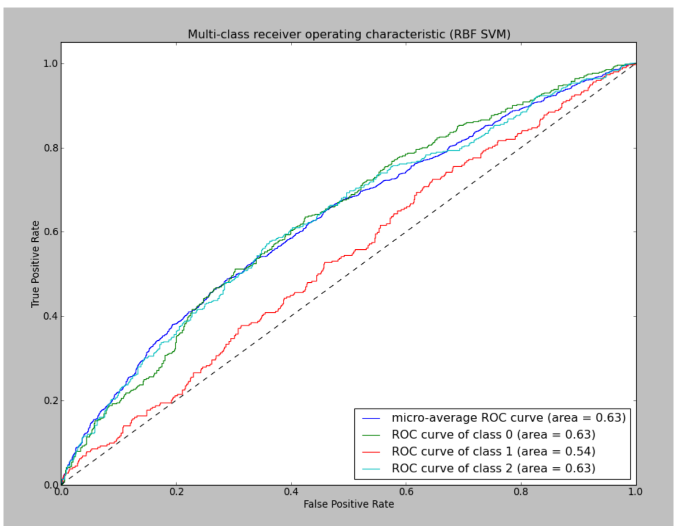

Figure 8. A receiver operating characteristic curve of classes 0 (win), 1, (draw), and 2(loss) from an RBF-SVM model

previous matches such as corner kicks. However, this project made strides in the evaluation of form as a feature, and reducing the overfitting problem that comes with the variability of soccer results.

V. Future Work

As mentioned above, one of our main challenges was the limited amount of data from the earlier seasons of our dataset. The obvious improvement to the accuracy of our system would come from finding sources of more statistics from that time period. Many models in literature and in industry use features such as player analysis, quality of attack and defense, statistics from previous matches such as possession and corner kicks, and one model went so far as to examine the effects of a team's mood before the game. Beyond researching data, we would also like to try implementing some of the probabilistic models referenced in Rue et al. and Joseph et al.; however, since these models met with limited success, we speculate that they would perform even worse on a more limited dataset [3], [4]. Finally, we are extending this system to predict league standings after a season. When we tried to predict the results of the 2012-2013 EPL season we correctly predicted Manchester United to win.

Acknowledgment

We would like to thank Professors Andrew Ng and Percy Liang. And the TAs of CS229 and CS221 for their contributions to our project. We would like to especially thank the following TAs for their guidance and insight: Will Harvey, Clement Ntwari Nshuti, and Celina Xueqian Jiang.

References

[1] "The World's Most Watched League." Web. 11 Dec. 2014. <http://www.premierleague.com/en-gb/about/the-worlds-most-watched-league.html>.

[2] A. S. Timmaraju, A. Palnitkar, & V. Khanna, *Game ON! Predicting English Premier League Match Outcomes*, 2013.

[3] A. Joseph, A. E. Fenton, & M. Neil, *Predicting football results using Bayesian nets and other machine learning techniques*. Knowledge-Based Systems 19.7 (2006): 544-553.

[4] H. Rue and O. Salvesen, *Prediction and retrospective analysis of soccer matches in a league*. Journal of the Royal Statistical Society: Series D (The Statistician) 49.3 (2000): 399-418

[5] "Mark Lawrenson vs. Pinnacle Sports." Web. 11 Dec. 2014. <http://www.pinnaclesports.com/en/betting-articles/soccer/mark-lawrenson-vs-pinnacle-sports>.

[6] "England Football Results Betting Odds — Premiership Results & Betting Odds." Web. 11 Dec. 2014. <http://football-data.co.uk/englandm.php>.

[7] "SoccerVista-Football Betting." Web. 11 Dec. 2014. <http://www.soccervista.com/soccer_leagues_ordered_by_number _of_draws.php>.

[8] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.