Soccer Analytics: Match result prediction in English Premiere League

Akhilesh Soni
Department of ISyE, UW-Madison
soni6@wisc.edu

Abstract

We predict the match results of soccer matches in English Premier League using Machine Learning algorithms. We create features using the historic data that incorporates overall strength and current form of the team. We use one vs rest strategy to handle multiple classes (Win, Draw, Lose). We train three classifiers: k nearest neighbors (KNN), Support vector Machines (SVM), and Neural Network (NN). We obtain the best error rate of 58.4% both with SVM and NN when using a subset of most important features identified through Random Forest.

1 Introduction

Soccer is a game of coordination. Each side consists of 11 players with 3 allowed substitutions. A game is played in two intervals of 45 minutes each with a 15 minutes break in between. The team which scores more goals is declared as the winner. Each goal, more often than not, involves a highly complex interaction among players amidst uncertainty of the situation. Sometimes goal results from individual brilliance and sometimes from the well coordinated passes among players of the attacking team. Of course, each good attack does not result in a goal and often there are goals scored against the run of play. However the game stats do give a sense of which team might be dominating the game and is more likely to win. We aim to study if we can predict result of the match from the historic game stats of the two teams against themselves and against other teams. In this project, we keep our focus on English Premier League or mostly refereed to as EPL. In a typical match of EPL, one team hosts another team. This means that each match involves one team playing “Home" and the other team play “Away" game. Each season of EPL runs from August to May of following year. There are 20 teams and each team plays 38 games: 19 home and 19 away.

Dataset EPL dataset for all matches over the last 20 seasons is available here [1] . The data along with final result (Win/Loss/Draw) consists of statistics like shots taken by home and away team, yellow or red cards conceded, score at half time,possession, number of off-sides, number of free kicks conceded, etc. These stats are a good representation of the match summary and hence can be used to predict the likelihood of home team winning, drawing, or losing. We also have standings of each team over the last 20 seasons in EPL. This is a good representative of the overall strength of the team over the last few seasons. There are roughly 30 features which consists of goals, off-sides, fouls, free kicks, cards, shots, corners etc. Each season of EPL consists of 380 games. We use 20 seasons of data which is equivalent to around 6000 data points roughly. Target label in our dataset is whether home team won, drew or lost.

Project Scope Our goal is to predict the match outcome using the historic statistics of the two teams playing. In particular, we consider the recent performance of the home and away teams in last 3 games at home and away venues against other teams in the league. We also consider the history of encounter of the playing teams against each other and their match stats in those encounters. To

Report submitted for CSE/ECE 532 project requirment, Fall 2020

account for the overall strength of the team, we also include position on which the team finished the previous season.

This model can then be used in real time for getting the odds of home team winning the game before the game has started. As there are 3 possible outcomes, a random classifier will have an accuracy of 33 %.

2 Data Engineering

We extracted following features from the data:

- Strength of a team: We measure strength of the team by it's position (standing) in the last year's league. Every season, top 17 teams are qualified for next year's league and bottom three teams in the table are relegated, thus giving a chance to three new teams to enter the league. Hence, sometimes it happens that we do not have information of team's performance in the last season's league. In such cases, we assign a position of 30 to the teams which did not play the league last season i.e. they didn't qualify.

- Current form: Current form of the home and away teams: Each season of EPL goes for almost 8-9 months. Hence, players go through different phases of performance and injuries which also affects the team's performance as a whole. To account for this behavior, we measure current form of the team by calculating exponential decay function of number of shots on target and goals (both scored and conceded) in the last 5 games with the most recent game getting the highest weight.

$$Form = \sum_{i=1}^{5} (STD + 2 * GD) e^{-\alpha i}$$

  Here STD stands for shots on target difference between the home and away team and GD stands for goal difference between home team and the away team. We give twice the weight to goal difference compared to shots on target difference as goal scored have a bigger impact on the game outcome team morale. We used $\alpha = 0.6$ in the Form expression. We experimented with different games to look back in future i.e. 3, 5, 8, 12. We observed the best performance for linear SVM using 5 games to to calculate form. Hence, we use 5 games through out the other models as well to calculate form feature.

- Previous encounters: If Team A (Home) is facing Team B, we account for all previous encounters of those teams. In particular, we give a point of +1 to Team A for a win in the past, 0 for a draw, and -1 for a loss. We then take an average of these scores. Recent performance: To account for team's recent performance, we also consider an average of the following stats over the last three games.
  - 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
  - HS = Home Team Shots
  - AS = Away Team Shots
  - HST = Home Team Shots on Target
  - AST = Away Team Shots on Target
  - HHW = Home Team Hit Woodwork
  - AHW = Away Team Hit Woodwork
  - HC = Home Team Corners
  - AC = Away Team Corners
  - HF = Home Team Fouls Committed
  - AF = Away Team Fouls Committed
  - HY = Home Team Yellow Cards
  - AY = Away Team Yellow Cards
  - HR = Home Team Red Cards
  - AR = Away Team Red Cards

  Note that we calculate this separately for a home or an away game. This implies that if a team is playing a home game, we only consider last 3 home games to measure the recent performance of team in home games. Similarly for an away game, we only consider last three away games of the team. We converted the targets (Win, Loss, Draw) to numerical features which are equivalent to +1,0,-1 respectively.

3 Methods

We use the following algorithms for prediction and tune the necessary hyper parameters to get best in sample accuracy.

1. k-nearest neighbors: Parameters: Number of neighbors (k) and distance norm (p)
2. Kernel SVM: Parameters: Kernel type (linear, polynomial, RBF) and penalty parameter
3. Neural Networks: Parameters: Number of hidden layer, nodes in hidden layer, activation function

Note that we make use of python libraries to implement the above mentioned models. In particular, we use sklearn for implementing KNN and SVM and keras to implement Neural Network.

We use GridSearchCV method in sklearn to perform hyper parameter tuning for KNN and SVM models. For neural network, we manually experiment with different network structure. In the hyper parameter tuning, we select best parameters using cross validation in our training data set to avoid overfitting and select the hyper parameters which give best accuracy on validation sets. Accuracy is the ratio of correctly labeled observations to the total number of observations.

As we are dealing with multi class problem, we use a OneVsRest methodology to handle multiple classes (Win,Draw,Loss). The multi-class dataset is split into multiple binary classification problems. A binary classifier is then trained on each binary classification problem and predictions are made using the model that is the most confident. We make use of *sklearn.multiclass* class to implement this. We consider a split of 70:30 for training-testing in the chronological order i.e. the most recent data is reserved for model testing purpose.

4 Results

4.1 KNN

In k-nearest neighbor, we consider the following parameters choice:

- Number of neighbors, k: {20,50,200,500}
- Norm, p: {1,2}

It took 54 seconds for hyper parameter tuning with $k = 200$ and $p = 1$ being the best choice parameters. With these choice of parameters, we got a testing accuracy of 54.8%

4.2 SVM

For support vector machine, we consider the following choice of paramters:

- Kernel, K: {linear, polynomial, radial basis function}
- Penalty, C: {0.1,1,10}

It took 547.07 to train and find the best set of parameters. We got the best results using a linear kernel and penalty parameter value to be 10. Out of sample accuracy with this choice of parameters was found to be 58.2% as shown in Table 1. We also observe that for high penalty and non linear kernel functions, SVM can overfit.

4.3 Neural Network

We consider the following network choices for out network:

- Number of hidden layers: {5, 10, 20, 40}
- Number of nodes in each hidden layer: {3, 5, 10}

For our dataset, many of the points (e.g. a top team vs a bottom team) would be easier to predict. Hence, we use hinge loss. However, the conventional hinge loss is applicable to binary class problems

Table 1: SVM results

Parameters	Testing accuracy %
C =0.1, K = *linear*	52.9
C =0.1, K = *poly*	55.0
C =0.1, K = *rbf*	52.1
C =1, K = *linear*	55.5
C =1, K = *poly*	51.4
C =1, K = *rbf*	54.1
C =10, K = *linear*	**55.8**
C =10, K = *poly*	44.8
C =10, K = *rbf*	48.5

Table 2: Neural Network results

Parameters	Testing accuracy %
# Layers= 3 ,# Nodes=20, ReLU	52.1
# Layers= 5 ,# Nodes=20, ReLU	**56.4**
# Layers= 10 ,# Nodes=20, ReLU	27.2
# Layers= 3 ,# Nodes=20, SoftMax	48.1

while our data is a multi-class. Thus, we make use of categorical or multi-class hinge loss [2] which lets us exploit the power of hinge loss in multi-class data. We kept number of epochs (passes through the data) to be 100. With a very wide network, e.g. 20 layers, we over fit the data and get poor testing accuracy. 5 layered network with 20 nodes in each hidden layer gave the best accuracy of 56.4%.

Reduced features using PCA and Random Forest

We first do PCA and consider the top 20 dimensions. We then test performance of KNN, SVM and Neural Network on this reduced dataset and got testing accuracy of 51.4%, 50.7%, and 44.1%.

Next, we also identify the top 10 features using Random Forest classifier. Random Forest had a testing accuracy of 55.1%. Our goal is to use random forest forest just to identify the most important features in the data. We consider the following 10 top features:

- Average home team shots
- Average away team shots
- Average shots conceded by away team
- Average shots conceded by home team
- Average full time result for away team
- Home Form
- Away Form
- Home Previous Position
- Away Previous Position
- Win probability based on previous encounters

The average number of shots here are based on last three home or away matches.

We then test the performance of SVM and Neural networks on these features. We got an accuracy of **58.4%** with neural Networks with these 10 features and **58.4%** with linear SVM. In both cases, 5 layered neural network with 5 nodes in each hidden layer gave the best accuracy. For SVM, we got the best results using a linear SVM with a penalty of 10 for each misclassified point.

5 Discussion and Future Directions

We obtain the best model performance (58.4%) when using linear SVM and Neural network and using top 10 features obtained from Random Forests. KNN on the same shrink features data gave an accuracy of 56.3%. Surprisingly, we did not obtain significant improvement over KNN. One of the reasons might be the fact that KNN uses 200 neighbors for the optimal choice of number of neighbors. Neural network and linear SVM perform almost equivalently. It might be the case that our features are unable to capture the underlying relationship of the data and hence we are unable to improve to accuracy anymore. Nonetheless, 58.4% accuracy is still significant and at par with the existing state of the art methods [3].

Although our model succeeds when compared to a random choice model (33%), we still believe there is room for improvement. In particular, more sophisticated features which incorporate team strength based on the funds spent each season to buy new players, player statistics (or form), etc should be elemental in improving model performance. We did not have direct access to such information but it would be interesting to explore this direction. As conventional PCA for dimensional reduction did not help in improving the model performance, one can also consider performing supervised PCA [4] to identify most important features in the data and then test model performance.

Our code can be found on the project website:

`https://github.com/soniakhilesh/soccer-analytics`

Acknowledgment

We would like to thank course instructor Prof. Rob Nowak for insightful discussion in the early phase of this project. We also thanks TAs Rahul Parhi and Jianwei Ke for their guidance during the course which was quite helpful in the success of this project.

References

[1] English Premier League dataset, https://www.kaggle.com/saife245/english-premier-league

[2] J. Weston, C. Watkins Support vector machines for multiclass pattern recognition The Seventh European Symposium on Artificial Neural Networks (1999), pp. 219-224.

[3] S. Srinivas, Soccer Analytics and its future (2019). Creative Components. 254.

[4] Supervised PCA: A. Ritchie, L. Balzano, C. Scott, Supervised PCA: A Multiobjective Approach (2020). https://arxiv.org/abs/2011.05309