A Study on Prediction of Outcome in Soccer Games of Premier League

Simou Nikonas
Computer Science Department
Heraklion, Greece
simou@csd.uoc.gr

Abstract

The prediction of soccer games is a task of high interest and challenge. Here, we try to present an approach to this problem using Historical data/statistics. The main aspects of this project are Feature Creation, Accuracy evaluation using Time Series cross-validation,Feature Selection and hyperparameter tuning. For that purpose data from English Premier league (years 2007-2017) where used. Results using Support Vector Machines with a linear kernel show potential given the fact that no data of team structure,player injuries,weather e.t.c where used.

1 Introduction

1.1 Soccer/Football Prediction

Soccer/Football is probably the most famous sport worldwide while the attempt to accurately predict the outcome of a soccer match is intriguing to soccer enthusiasts, sports managers/analysts and researchers alike. In soccer many things can be a subject of betting(number of fouls,off-sides,red cards,goals e.t.c) however, here, we are trying to predict only the winner (Home,Away,Draw). The decision to select Premier League as our target is because of the high acclaim of that league which gives the potential for future work using data/statistics which would not be available for "smaller" leagues. Also it is widely used in the bibliography and therefore can be used as a benchmark.

1.2 Motivation & Previous work

Soccer is estimated to be responsible for the 70% of the betting industry's net worth. However, apart from profit making, prediction of sports(and especially soccer) are known for their high complexity and unpredictability which makes the task a rather interesting one from a Machine Learning standpoint. Various approaches considering the subject have been considered. The first one which represents a statistical approach, was speculated in 1982 by Moroney who used Poisson distributions and negative binomial distributions to predict the number of goals scored by both teams in a match and later confirmed with actual data by Mike J. Maher. [1]. Another popular approach is the use of Bayesian Networks first used in [2] proposing a Bayesian model which presents new parametrisation and ideas in goal-modelling. The last famous approach, is that of Machine Learning algorithms like Logistic Regression, Random Forests, S.V.M and others. Lately some work has been done with other data sources like Twitter[3] and video games [4] which show great potential compared to historical data.

2 Problem Definition - Data Description

The goal of the project is to try various classification algorithms on historical football data in order to predict whether a match results in Home/Away win or Draw (denoted H/A/D from now on). Model

Submitted to Course CS-577 of Computer Science Department - University of Crete 2018

selection using nested cross validation was used to tune the hyperparameters and Feature Selection in order to boost algorithm performance.
The dataset used, consists of data from `http://www.football-data.co.uk/data.php` gathered for seasons 2007-2017. The data provided by the website consist of Home Team Name,Goals scored,Fouls,Yellow-Red cards,number of shoots,number of shoots on target e.t.c for both Home-Away teams. Also betting odds for H/A/D. The dataset is partitioned in 11 sub-datasets (one for each season) each of which is stored in a `.csv` file. A more detailed description of the dataset is presented in the file `Dataset_explained.txt` which is taken directly from the website mentioned before. The full dataset contains 3650 matches each one represented by a row.

3 Methods Used

3.1 Final Dataset Creation / Feature Extraction

In order to build a classifier that can be used in a real life situations we need to use data that are available prior to the match that needs to be classified. For that reason we calculate the average of in-game performance statistics(Fouls,Shoots on Target e.t.c) for each team and each match (prior to the current one) in the same season. To be more specific, given a team name we have to select all previous matches of that season in which that team plays in and calculate the average statistics of the k previous matches for that team and it's opponent. The procedure is repeated for every match in the season (except the first of each season because no previous matches are there to consider) and for each season (unless the desired team does not appear in that season). Additionally, the feature `WinStreak` was calculated and represents the consecutive wins of a team up to that match (if the team has no consecutive wins, the feature will be set to zero).After repeating this for each season, we combine the data of each season and create the final dataset. It should be noted that the previous procedure (called "K-Team Statistics" from now on) results in a unique dataset for every unique team where every row represents the performance of the team in it's k previous games, the opponent team's performance in k previous games and the betting odds provided by various sources.

3.2 Algorithms/Methods used

3.2.1 Preprocessing

In the initial dataset,various features like Referee,Match Date, league e.t.c should be omitted preserving only numerical data. Furthermore, in order to use classifiers more robustly we need to scale our data so that every feature is between $[0, 1]$. Scaling in this project was applied using this formula[5]:

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)}$$

3.2.2 Cross Validation

Even though k-fold cross validation is a useful tool, it does not operate properly on data that have time dependencies. For example a model that trains on future matches but tests on current ones will result in inaccurate estimations of accuracy. In order to fix that problem we use Roll forward cross validation where every train example is always preceding all the test examples.The process is called Time Series Cross Validation (Figure 1).

3.2.3 Feature Selection

By following the "K-Team statistics" method as described and using all of the betting odds provided leads to a feature space of over 50 dimensions. That calls for the need of dimensionality reduction / feature selection. In this project feature selection was applied using Recursive Feature Elimination (R.F.E) using Random Forests. The goal of R.F.E is to select variables by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each one is measured. Then, the least important features are pruned from current set of variables. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached.

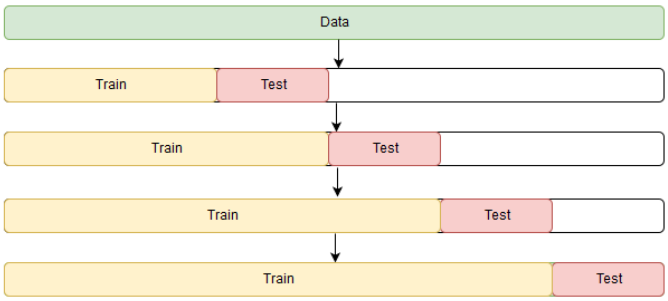

Figure 1: Example of Roll forward cross validation

3.2.4 Classification Methods

- **Random Forests:** An algorithm developed by Leo Breiman[6] that tries to fix the problem of overfitting when using Decision Trees by creating many bootstrap aggregated decision trees and simultaneously splitting on a subset of the feature space each time.
- **Support Vector Machines:** One of the most successful algorithms in the field of M.L especially when considering kerneled S.V.M where we can map our data to high (even infinite length) feature spaces. The training time of the model is small because of the rather few training examples.
- **K-Nearest Neighbors:**K.N.N is a memory based learning method where the classification of one training point depends on the label of it's k neighbors that have the smallest distance from it. Works efficiently for small (≤ 25) dimensions but not for large dimensions because of the problem known as 'Curse of dimensionality'. The training time is dependent of the dataset size (in this occasion small).

However using these methods requires hyperparameter tuning, in order to create a model that fits our data in an optimal (best accuracy) way. In this project, the various hyperparameters that where "tuned" are:

1. **Random Forests:** Number of estimators (decision trees) to be used. Note that increasing the number of estimators reduces variance but greatly affects training time.
2. **S.V.M:** Selection of kernel (Polynomial,R.B.F,linear) and also selection of a penalization constant (C) for samples that fall under the wrong side of the margin.
3. **K-N.N:** Number of neighbors(k) to consider every time,which also affects training time.

3.2.5 Nested Cross Validation

In order to tune our parameters and perform R.F.E we need to have a hold-out set upon which we estimate the error for various configurations of the algorithm's hyper parameters. Note that the same separation is done on the initial data, creating a hold-out set for performance which results in a nested cross validation setting.[1]

3.3 Description of the Implementations and Software used

The implementation was executed in Python 3.6.3 [Anaconda, Inc] using mainly the libraries: Sickit-learn,Pandas,Matplotlib. The code written consists of two files `Functions.py` and `Test.py`. Because of the fact that running the tests for all possible configurations takes a reasonable amount of time we have saved accuracy results in the files with extension `.spydata` which can be imported using the Spyder I.D.E.
Because of the various splittings of the dataset required for nested cross validation Teams which do

[1]It is highly important that the tuning, feature selection and scaling procedures are performed according to training data only, otherwise the classifier "peeks into the future" and produces optimistic estimations [3]. In this project the scaler created using training/validation data only is saved in a file to be used afterwards on the test data

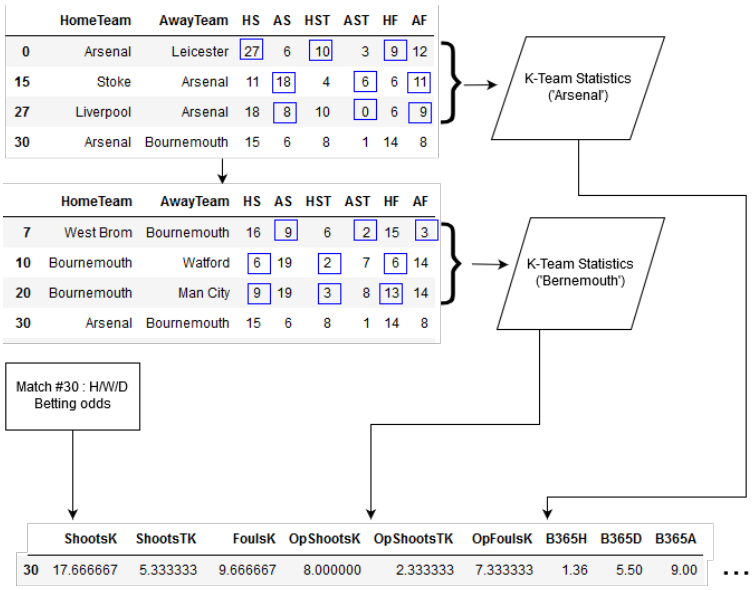

Figure 2: Example of "K-Team Statistics" for Match "Arsenal VS Bournemouth. In the blue boxes are the statistics that correspond to the team in which we are interested in. The final results are the average of each statistic(HS,Fouls e.t.c) respectively combined with the betting odds.

not have at least 200 games on record are rejected because they result in folds of too few samples and therefore we may have folds with one or two classes only.
Hyperparameter tuning was performed using the grid search algorithm which is actually an exhaustive search through the specified hyperparameters.

4 Experiments & Discussion

The experiments presented below, derive using a 5-fold Time Series cross validation.

Algorithm	k=5	k=10	k=15
S.V.M	51.28	50.85	50.57
R.F	49.73	48.44	49.34
K.N.N	46.36	46.39	45.08

Table 1: Average accuracy(over all teams) for different values of k(kTeam Statistics)

As it can be seen, S.V.M (with linear kernel) and R.F achieve the best results in most cases. Furthermore, k=5 seems to be a better decision than larger values. That can be explained through intuition because using too large values of k means that we rely on matches that happened a long time ago and have little impact on the current ones. Thus, the dataset becomes noisy.
By looking at the dataset we can easily see that the most common class (result) is `H` which is because of the phenomenon called "Home Advantage". Thus, our trivial classifier will be the one that always predicts `H`. In average (and in most cases) the trivial classifier is outperformed by S.V.M, R.F and K.N.N. However, there are some teams for which a trivial classifier does better than the previous algorithms (Figure 3).
In order to see how well the algorithms perform on each class individually we use confusion matrices (Figure 4). We can clearly see that both precision and recall are very low for the `D` class. Also, S.V.M seems to have a rather good accuracy for `H` class but lower for `A`,`D`. This indicates that there is high bias towards `H` class due to it's high frequency of occurrence in the dataset compared to the 2 others.

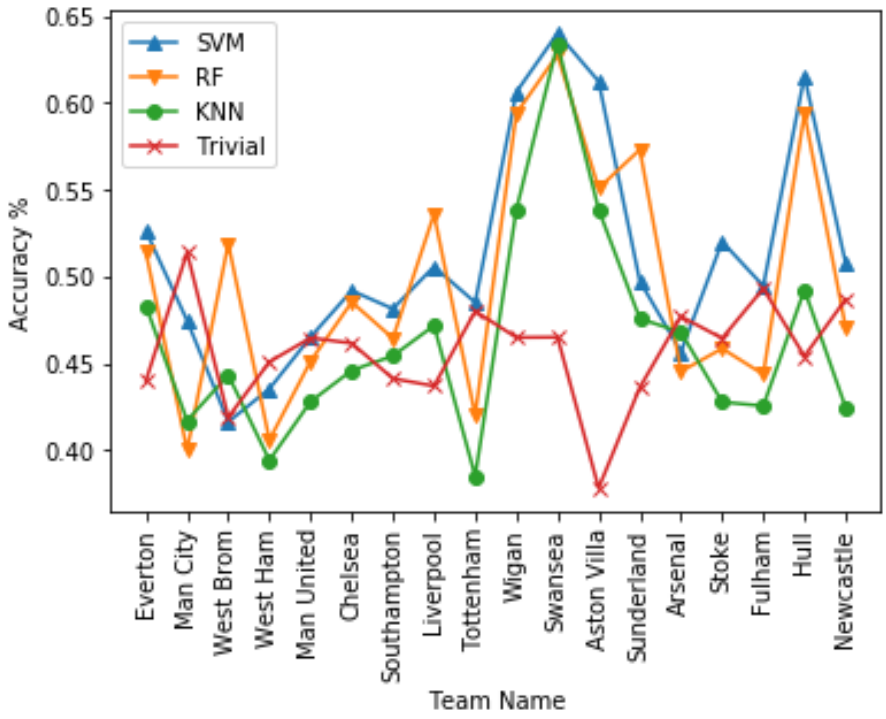

Figure 3: Accuracy of each model when trained/tested on data of one team using k=5 Team Statistics.

5 Conclusion

Predicting the outcome of a football match (especially for three class classification) appears to be a rather complicated endeavor. However, even by using public data, one can achieve a non trivial result. It should be stated that the highest average score achieved ($\approx 51\%$ using linear S.V.M) is probably conservative because in the first steps of k-fold Time Series c.v the training data are too few and the model probably does not reach it's full potential.
Moreover, because of the high unpredictability of the sport, one could also extract some confidence levels for the result if interested in betting for that match. In the case of S.V.M's one can measure the distance of the testing example to the margin, while in the case of R.F's we can measured confidence of classification as the number of individual trees that voted for the same class. That way some matches with high risk can be avoided.

5.1 Future Work

There are various directions for future work on this project. Firstly, one can experiment with relying less on H/W/D odds provided as they mostly represent the trends of the betting industry and not the actual probabilities of the result.Also, using text mining techniques on text provided by football analysts or social media one can extract more sophisticated information about a soccer match like top scorer injuries, financial struggles of team, weather predictions,social life of team players e.t.c.[8].
Another interesting direction would be the use of causal models in order to find which features matter for prediction and which are redundant/irrelevant and may deteriorate our learning algorithms.
Last but not least, if enough data are gathered, deep learning architectures should be considered for more accurate predictions.

References

[1] J.Maher(1982) Modelling association football scores. Statistica Neerlandica.

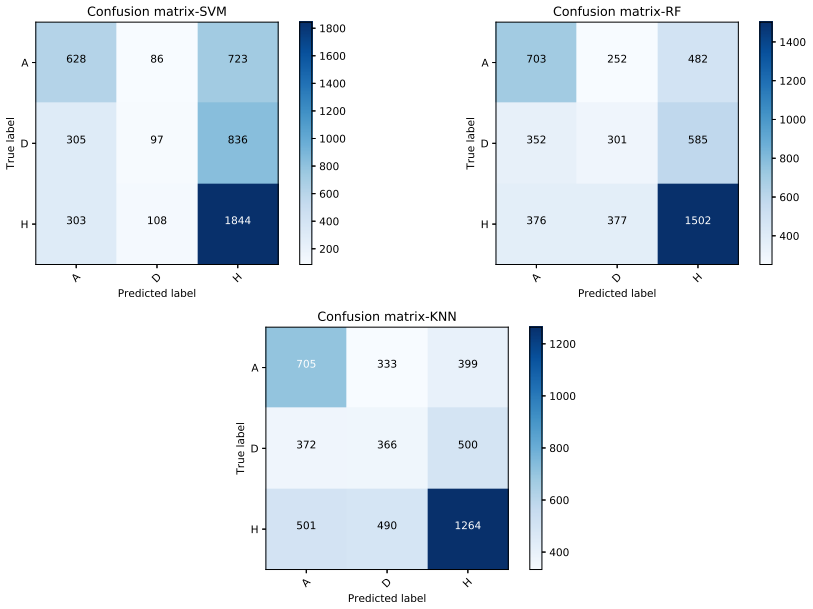

Figure 4: Confusion Matrix.

[2] Håvard Rue & Øyvind Salvesen(1997) Prediction and Retrospective Analysis of Soccer Matches in a League. Norwegian University of Science and Technology.

[3] Shiladitya Sinha, Chris Dyer, Kevin Gimpel & Noah A. Smith Predicting the NFL Using Twitter.

[4] Jongho Shin & Robert Gasparyan (2014) A novel way to Soccer Match Prediction.

[5] `http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html`

[6] Leo Breiman Statistics Department Berkeley (2001) Random Forests

[7] Performance-Estimation Properties of Cross-Validation-Based Protocols with Simultaneous Hyper-Parameter Optimization Ioannis Tsamardinos,Amin Rakhshani,Vincenzo Lagan

[8] `http://www.bbc.com/sport/football/premier-league`