Use of algorithm model for predicting the outcome of English Premier league

Group 17

Go Jehwan, Lee Jae Yeon, Jeon Minhyek

Introduction to Brain and Machine Learning

Abstract

This paper introduces regression model that predict the goals made in the each given match. Based on analyzed player's data induced from English Premier League, competitiveness of specific team for prearranged matches was calculated. Using the output extracted from each match, the possible ranking for each team in English Premier league for 2020/2021 season was predicted. Associated to this prediction, top 4 teams that are going to participate UEFA champions league was decided.

With this algorithm, team with most systematically high of standard can be decided. Our team specifically focused on building a model for English Premier league, but the algorithm model proposed in this paper can be utilized to any kinds of league matches, regardless of its types (e.g. MBA, E-sports league) Our regression model was trained based on player's features from season 10~11 to season 19~20 and match result for season 20~21 was predicted.

1. Introduction

English Premier League became one of the most popular sporting leagues in the world and in 2020, it became highest valued league with a brand value of $6.4 billion. This refers to more investment and viewers. While preparing for our project, we wanted to make an algorithm model not just for our personal contentment but one that actually will be pragmatic and can be utilize by many people. Since English Premier League is most prominent league, there are more possibility that this model can be helpful to other people. As all three of our team members are ardent English Premier League supporters, our desire to make a complete model without flaws or deficiencies expanded. As we searched for information, we were able to find many modeling used for predicting the match result. Most of the model basically used the same algorism, computing the percentage of victories. Model normally used in many predicting websites basically estimated each team's competitiveness only using the output of past match results. However actual factor that decides the output of the match is the player's ability. They are the principal agent who actually compete in the match and fight against opponent team players.

To make the output result more precise and reflect reality, our team decided to train our model based on player's ability like pacing ability, shooting ability, defending ability and so on. With our trained, model, we selected best 11 players for the future matches and was able to predicted the result. By analyzing the matches for remaining 20~21 season, we were able to predict the outcome ranking of the English Premier league.

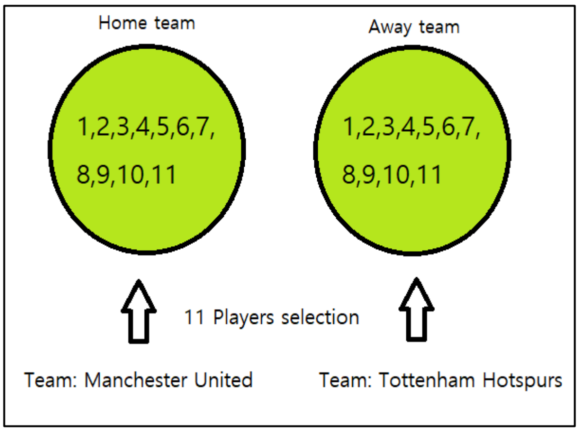


Figure 1: Description of our model

As it will be thoroughly explained later in this paper, Since we are using player's ability as our main feature, while computing the result of the match scheduled, our model was programmed to not make prediction on comparison of team by team, but took the form of comparing two group with selected 11 players.

Therefore, the prediction can be more adaptable to reality.

Another reason why selecting the player is more suitable for our project is because a team meets a team only twice in a season. Problems arose with only 20 matches data in a total of 10 seasons. If a model that predicts a match between two teams does not set up two teams as a specific team, we can use all matches data in 10 seasons as training data. (For example, the model doesn't predict 'Chelsea' versus 'Manchester United' but predict 'team A' versus 'team B' with information of each team). Using these models, a total of 3,800 data will be available as training data which have about 190 times more training data than the previously designed model. Therefore, the new model can be used to predict the results and scores of each team using the information of each team.

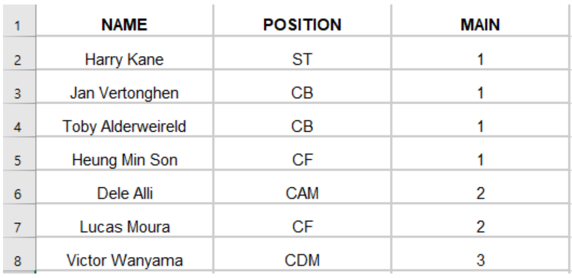
The project will be proceeded with 'Colab' which is offered by Google. "Colab" has the advantage of being able to execute heavy learning models without good equipment, convenient to share code among team members, and providing various Python libraries (e.g. Numpy, Pandas, Sklearn etc.). We will use the 3.6.9 version of Python provided by 'Colab', and use the Pandas, Numpy libraries for data analysis, and use the Sklearn library for implementing the learning model.

2. Modification of the raw data

This project can be distributed into 4 parts. First is to getting the data and modification of data so that it will be suitable and preferable in training. Second part is preprocessing of the data. Third part is the modeling and training process and last part is making prediction.

For our team, process of modifying the data took most of the time. The raw data we got was player's ability data for all the soccer players a in Europe. Thus, the players from La Liga(Spain league), Serie A(Italy league), etc. were all mixed up. From 10~11 season to 19~20 season. Nonetheless, process of extracting only players for each team for English Premier League was to be preceded. There are 20 teams for English Premier League and data from 10 seasons were used Therefore 20 files for 10 seasons were made, ready for preprocessing. Features extracted from raw data were position, tier, rating, passing, shooting, pace, dribbling, defending and attacking.

Since there was no label or data for the lineup of the match, we had to match each player into three groups. The first group was players selected for the starting lineup. Second group was sub players be that can be changed with starting members in case of injury or change in tactics, etc. The last group was formed with players that are in most cases not used. Additionally, same football tactic applied on every team that is 4-4-2(4 defenders, 4 middle fielders, 2 forwards). Positions in football distinguished in detail, so we set a standard. First CB, LB, RB, LWB, RWB as defender second CAM, CM, CDM, LW, RW as middle fielder last CF, LF, RF, ST as forward. Each group was labeled 1, 2, 3. 1 for starting members, 2 for sub members and 3 for others. This labeling was made in a separate column for use in selecting players for each match. This process was done one by one by our hands and it took a whole month for labeling the data. The first step for choosing the label was gathering information about each team and their lineups. Some players were never used in English Premier League but conjugated for other unimportant matches like UEFA Europa league. Some players were crucial and was utilized in every game. Based on this information, we had to label about 10,000 players starting from 2010~2011 seasons to 2019~2020 seasons.



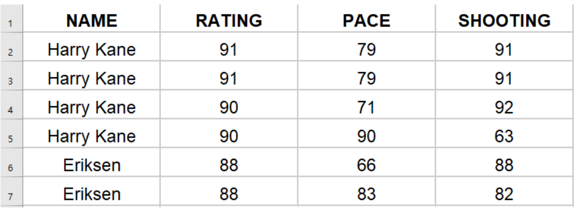
1	NAME	POSITION	MAIN
2	Harry Kane	ST	1
3	Jan Vertonghen	CB	1
4	Toby Alderweireld	CB	1
5	Heung Min Son	CF	1
6	Dele Alli	CAM	2
7	Lucas Moura	CF	2
8	Victor Wanyama	CDM	3

Figure 2: example of players data

This is an example of players file data file for 2019~2020 season for Tottenham Hotspurs. As it can be viewed in the column named MAIN, players were divided into three groups. Players like Harry Kane and Jan Vertonghen were key players during the season and was used in every match in English Premier League, excluding the time when they were injured. Players like Lucas Moura were used mostly as sub players and in most time, changed during the match with starting member Heung Min Son (Labeled 1). Therefore, Lucas Moura was labeled with 2.

There might be doubt in this process. Labeling best 11 based on the ranking would be much easier. However, there are two reasons why this method is not useable. Firstly, by only considering the ranking, there might be a team made up of only goalkeeper. For example, if there are two goalkeepers with highest top 2 ranking in the team, both of them will be picked as best 11 even though the team needs only one goal keeper. So best 11 might be picked without consideration of the position.

There can be a suggestion of making a limit made for number of each players for position and pick players based on their ranking. However different formation is used by each soccer coaches and some coaches preferred some players prior to players that had higher rating scores. Making a prediction needs data that are most precise to actual ones. Therefore, we decided to use the actual information for picking the best ll for the game.



1	NAME	RATING	PACE	SHOOTING
2	Harry Kane	91	79	91
3	Harry Kane	91	79	91
4	Harry Kane	90	71	92
5	Harry Kane	90	90	63
6	Eriksen	88	66	88
7	Eriksen	88	83	82

Figure 3: multiple data for same player

After labeling process, we need to erase or manipulate some data. In our raw data, there were different arrays of player's ability for same player, just like the table above.

As it can be seen from this example, in this data there are 4 possible ability score for Harry Kane. This is due to differentiation in the ability overall of the season. Every player has undulation in their performance. In some match, they might get high rating and in some other games, they can perform harmful to the team. Therefore, multiple data represent the different ability of player during the season. However, those data cannot be used as input for our model, since each of the array will be treated as different players. In this example, there are 4 players named Harry Kane, and 2 players named Eriksen.

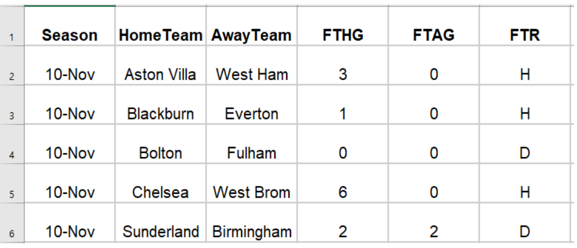
Therefore, we had to unify multiple data into one representative data. The best is using the performance data that match the dates for each match. However, this was not possible since the number of multiple data for each player were different. Some players had up to 6 data while some players only had one. For this reason, only uniting the data was the best solution.

Two steps were taken in the process of uniting. First step was erasing the outliers. Some data had absurd rating score. For example, in some season, in one of multiple data, Harry Kane had 100 rating score. Considering the fact that Lionel Messi, known as the best soccer player has 99 rating score, this data has no reliability. Therefore, those outliers were erased. Second step for this process was calculation of mean value for rest of remaining multiple data. With these two processes, multiple data were united into one data.

3. Getting the data for test set

After training the model, we need a test set to compare the result that our model has predicted. Thus, starting from 10~11 season to 19~20 season of the English premier league, we collected all the results of every match, goals made home team and goals made by away team. We first thought about dividing the result into 3 parts. In our first model, each match was labeled as 0 if away team won the match, 1 for draw and 2 if the home team won the game. Therefore, every match game has a label which varies from 0~2. Our model makes prediction of the winner of the match based on player's ability. Therefore, the outcome will also be 0~2. The output of the model will then be compared to test set.

However, as our team programmed our model, we realized that output cannot reflect players ability. Even though if an away team contains best rating 11 players, the output will be only 0 no matter how good they are. Therefore, a solution was needed in order to reflect player's ability in proportion. To solve this problem, model was modified into predicting the number of goals made in each match. By predicting the goals, not only will it be proportioned to team's ability, but it also represents players defending and attacking ability. For example, if away team made 5 goals and home team made 1 goal, it could be inferred that away team has good striker, while home team has inferior defender. By changing the output into number of goals, the model could be trained more suitably so that the prediction can reflect reality better.



1	Season	HomeTeam	AwayTeam	FTHG	FTAG	FTR
2	10-Nov	Aston Villa	West Ham	3	0	H
3	10-Nov	Blackburn	Everton	1	0	H
4	10-Nov	Bolton	Fulham	0	0	D
5	10-Nov	Chelsea	West Brom	6	0	H
6	10-Nov	Sunderland	Birmingham	2	2	D

Figure 4: example data for test set

Figure 4 is an example of output set we used to train our model. After inputting 11 best players for Aston villa and West Ham, each players ability will be associated with the output 3 : 0. Same with Blackburn and Everton. In the array 5, Chelsea won the West

Brom by 6 : 0. This infers the fact that Chelsea not only had good striker but also good defender, considering the fact that West Brom scored no goals. With every match result, the model will be trained to make a relationship between the number of goals and ability of each players.

4. Methods

To predict the results of Premier League matches this season, we design a model which predicts all results for the remaining match schedule after December 11th. The ranking requires not only 'points' but also 'goal difference', so a regression model was used as a predictive model to predict how many goals each team scored, rather than simply a classification model that predicted victory or loss in the game.

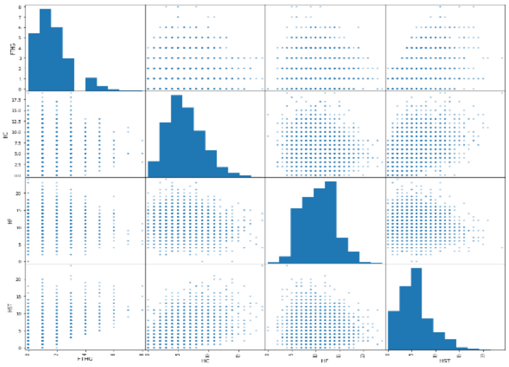


Figure 5: Correlation of each feature.

To find out which features affect the number or goals, the correlation between each feature and the number of goals was plotted. (Figure 5) And it was confirmed that the number of shootings, on target, and corner kicks had some relevance, and they were used as features needed to predict the number of goals along with player stat data. However, since we don't know the number of shootings or corner kicks of matches which not yet played, we need a new model to predict these features.

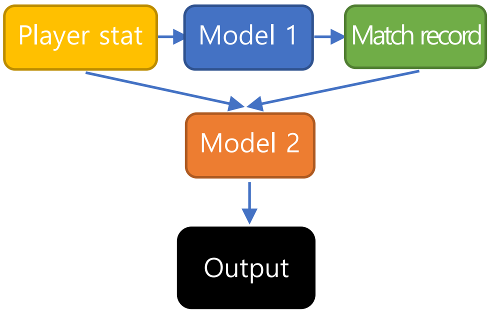


Figure 6: Description of two models

Like Figure 6, based on the player's stats, the first model predicts major records except for the score of the game. And then, based on the records which obtained by the first model, the second model predict the score of the game.

In a soccer game, which team is home or away has a very big impact on the outcome of the game. Therefore, each feature of the two models is divided into home and away in order to obtain the weight of each feature independently by home and away.

Player stat data, which play an important role in both models, could not be applied to the model as they were because all player has their own stats as each team. To apply the player stat data to both models, it was necessary to switch to dataset, which has representative features for the team, which is implemented by the following 'player_stat _standardization' function.

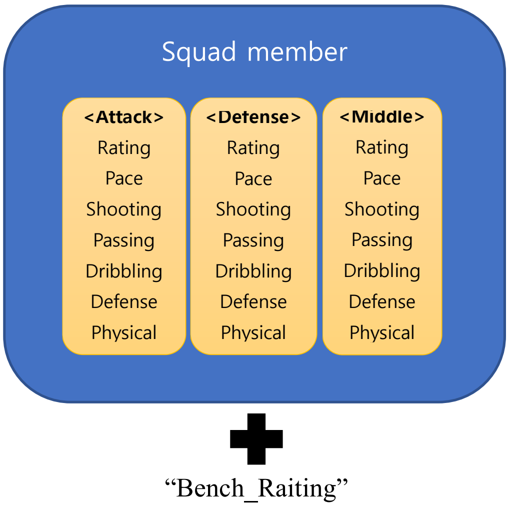


Figure 7: Schematic of 'player_stat_standardization'

In this function, the players are divided into three parts: squad, reserve, and candidate, and each player is also divided by position. There are many different position in soccer, but to prevent too many features, position were divided into three parts: Attack, Middle, Defense. Each player's stat consists of a total of seven elements: 'Rating', 'Pace', 'Shooting', 'Passing', 'Dribbling', 'Defending', and 'Physical', and all of their abilities were used as features because they were of the greatest importance to the game. Therefore, there are 21 features because there are seven elements(‘Rating’, ‘Pace’, ‘Shooting’, ‘Passing’, ‘Dribbling’, ‘Defending’, and ‘Physical’) per three positions(‘Attack’, ‘Middle’, Defense’) and the stat of the squad member was represented as the average value of 11 players for a total of 21 features. Also, since each team has a different position configuration for the reserve member, the average ‘Rating’ of seven members of the reserve member is set as a new feature called 'Bench_Rating' without considering the position. Finally, the player stat is standardized into 1*22 matrix with a total of 22 features per team.

Using these two trained models, we predict the score for the remaining 270 games. The last thing we need to do is to predict the final ranking as of December 11th by adding the points and goal difference based on the predicted results for each team.

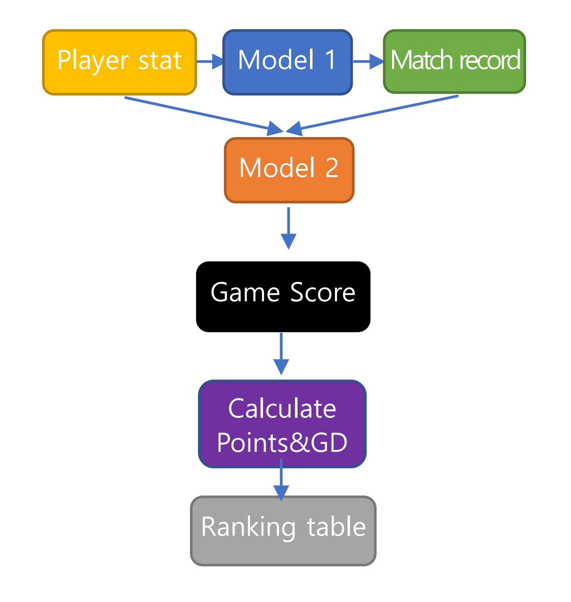


Figure 8: Schematic of whole process

5. Model

This project requires a total of two models, the first model predicts the number of shooting, on target, corner kicks based on player stat and second model predicts the score of matches based on player stat and results of model 1.

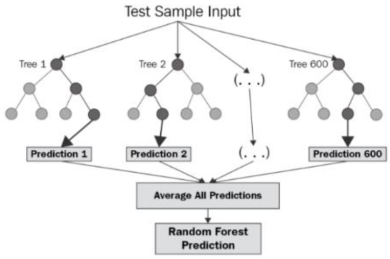


Figure 9: Diagram of Random Forest Regression

In the model 1, the Random Forest Regression model was used. It is a supervised learning algorithm that uses ensemble **learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. The diagram above shows the structure of a Random Forest. The trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees. A Random Forest Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationships. Disadvantages, however, include the following: there is no interpretability, overfitting may easily occur. So we need to tuning hyperparameter of Random Forest Regressor, using grid search, and could get proper hyperparameter to prevent overfitting.

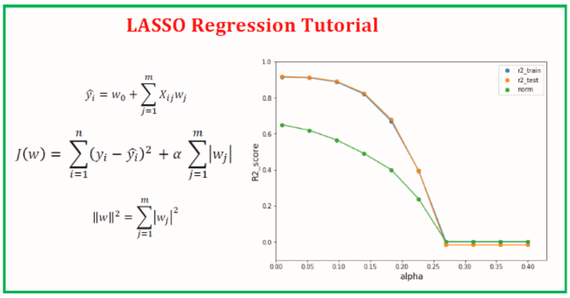


Figure 10: Schematic of Lasso Regression

In the model 2, the Lasso Regression model was used. Lasso regression is an example of regularized regression. Regularization is one approach to tackle the problem of overfitting by adding additional information, and thereby shrinking the parameter values of the model to induce a penalty against complexity. The 3 most popular approaches to regularized linear regression are the so-called Ridge Regression, Least Absolute Shrinkage and Selection Operator (LASSO) and Elastic Net method. Lasso regression is an L1 penalized model where we simply add the L1 norm of the weights to our least-squares cost function:

$$J(w) = \sum_{i=1}^{n}(y_i - \hat{y_i})^2 + \alpha \sum_{j=1}^{m}|w_j|$$

Figure 11: Cost function of Lasso regression

By increasing the value of the hyperparameter alpha, we increase the regularization strength and shrink the weights of our model. If alpha =0, it corresponds to standard regression analysis. Depending on the regularization strength, certain weights can become zero, which makes the Lasso method a very powerful technique for dimensionality reduction and it is proper model 2 which has large size of input data's feature. (It includes player stats and number of shootings, on target, coner kicks) Again, we tune hyperparameter 'alpha' using grid search.

Using grid search we could get 'n_estimators' = 2000, 'max_features' = 4 at model 1, 'alpha' = 0.001 at model 2 and could reduce overfitting problem.

6. Training

To train the models we need two datasets, player_stat and match_history. Because each team's roster changes every season, the player stat dataset of 20 teams has been separated into 10 seasons, creating a total of 200 datasets. (For example, Arsenal_2020 means dataset of Arsenal players at 19-20 season). The match history dataset includes the results of all 3,800 games over the last 10 seasons, we separated a total of 6 features as label: HS(Home shot), HST(Home Shot on Target), HC(Home Corner kick), AS(Away shot), AST(Away Shot on Target), AC(Away Corner kick) for implement model 1 and to generate input data for model 1, the Home/Away team's standardized player stat data was added as features for each 3,800 games. (It has a total of 22*2 = 44 features).

In model 2, the score was separated into a label in match history dataset this time. (FTHG means Full Time Home Goals, and FTAG means Full Time Away Goals). To generate input data suitable for model 2, we combined the results obtained from model 1 and the player stat input data used in model 1 to create new input dataset. (It has a total 44+6 features).

Because the training data used in model 1 and model 2 have no missing values, we didn't perform impute process. However, the distribution of law training data was not tidy, so the scaling was carried out. We used 'StandardScaler' in sklearn which change the mean of each feature to zero and variance to 1 to give the same scale.

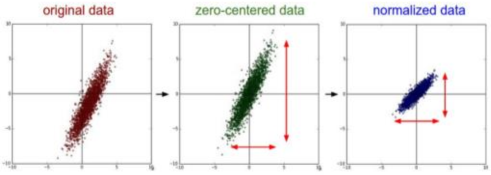


Figure 12: Plot of Standard Scale

For model 2, the results predicted in player stat data and model 1 are input, and the scale of the data varies by features, resulting in different weights between the two features, which may cause problems in predicting the result values. To prevent this problem, the standard scaler made the value into zero-centered and equivalent to all feature's scale.

If the model without scaling, the loss is very sensitive to changes in weight, and it is hard to optimize. However, after scaling include normalization, it is less sensitive to small change in weights, and easy to optimize. As a result, we were able to train more accurate models.

In order to train and evaluate model 1 and model 2, test sets should be split from the dataset in advance, but only 10% were used for the test sets considering that the amounts of datasets was not enough. For evaluate we use r2 score. The r2 score varies 0 and 100%. It is closely related to the **MSE**, but not the same. The definition r2 score is the proportion of the variance in the dependent variable that is predictable from the independent variables. So if it is 100%, the two variables are perfectly correlated, i.e., with no variance at all. The below equation shows how to r2 score calculated.

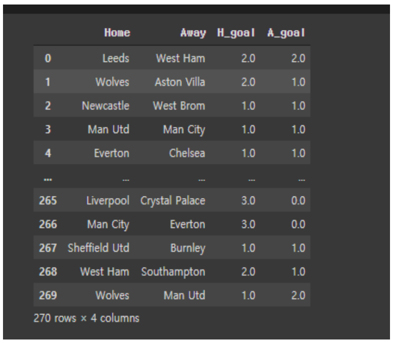
$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Figure 13: R-Squared equation

SSres means the residual sum of squared error of our regression model, and SStot means square the difference between the all the actual y values and the mean and add them together.

7. Experiments and Results

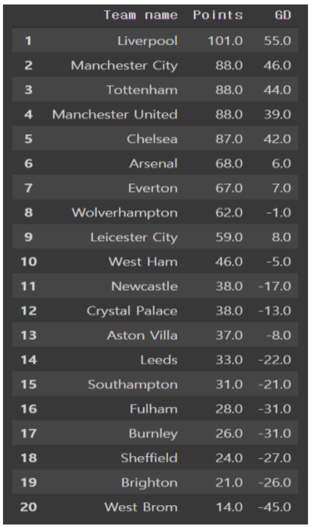
In this section, we will present experimental results of our premier league prediction model and calculated result of winning points. Experiments are trained on Google COLAB. First use random forest regressor to train model1 which predict shootings, corner kicks etc, second use lasso regressor to predict number of goals.(match prediction) Last, calculate win-points based on current win-points and previous result. (predicted win-points).



	Home	Away	H_goal	A_goal
0	Leeds	West Ham	2.0	2.0
1	Wolves	Aston Villa	2.0	1.0
2	Newcastle	West Brom	1.0	1.0
3	Man Utd	Man City	1.0	1.0
4	Everton	Chelsea	1.0	1.0
...	...	...	...	...
265	Liverpool	Crystal Palace	3.0	0.0
266	Man City	Everton	3.0	0.0
267	Sheffield Utd	Burnley	1.0	1.0
268	West Ham	Southampton	2.0	1.0
269	Wolves	Man Utd	1.0	2.0

270 rows × 4 columns

Figure 13-1 : match prediction



	Team name	Points	GD
1	Liverpool	101.0	55.0
2	Manchester City	88.0	46.0
3	Tottenham	88.0	44.0
4	Manchester United	88.0	39.0
5	Chelsea	87.0	42.0
6	Arsenal	68.0	6.0
7	Everton	67.0	7.0
8	Wolverhampton	62.0	-1.0
9	Leicester City	59.0	8.0
10	West Ham	46.0	-5.0
11	Newcastle	38.0	-17.0
12	Crystal Palace	38.0	-13.0
13	Aston Villa	37.0	-8.0
14	Leeds	33.0	-22.0
15	Southampton	31.0	-21.0
16	Fulham	28.0	-31.0
17	Burnley	26.0	-31.0
18	Sheffield	24.0	-27.0
19	Brighton	21.0	-26.0
20	West Brom	14.0	-45.0

Figure 13-2 : predicted win-points

According to Figure 13-1 270 matches will be predicted, Figure 13-2 Liverpool, Manchester City, Tottenham, Manchester United, Chelsea, Arsenal will be the Top Big 6 club. Result of prediction model's accuracy was low, but reflect current ranking as well.

8. Conclusion

We have demonstrated prediction of English Premier League using random forest, Lasso regression. Compare proposed model's result with other project, experimental results for the proposed model show low performance but reflect current ranking as well. Other projects such as convolution models, ensemble models are not as effective on the premier league prediction task. This may be due to automatic team composition. Whereas labeling main players, sub players, spare players enable correlations to be made straight forward views, ultimately contributing to better diagnoses. Predict number of shootings, corner kicks using random forest regressor to train model 1, predict number of goals using Lasso regressor to train model2. Combining predict number of goal data with residual schedule(win-points). We have demonstrated prediction of Premier League's match win-lose and win-point. Low performance could be improved by future work. we hope this work can be applied towards sports prediction algorithm.

9. Future Work

9.1 Data Augmentation

One major challenge for using the Premier League dataset is that Premier League contains only a very small amount of data(each team's relative record). As a solution, we applied models as Home team, Away team not as Each team. On the other hand, other work on this exact dataset had shown that augmenting the dataset helps with the network performance. If a new method is devised, result will be more convincing.

9.2 Data cleansing method

Labeling main players contributed to better performance. However, it could be the weakness for our project too. While modifying raw-data, 4-4-2 football tactics reflected to all teams.

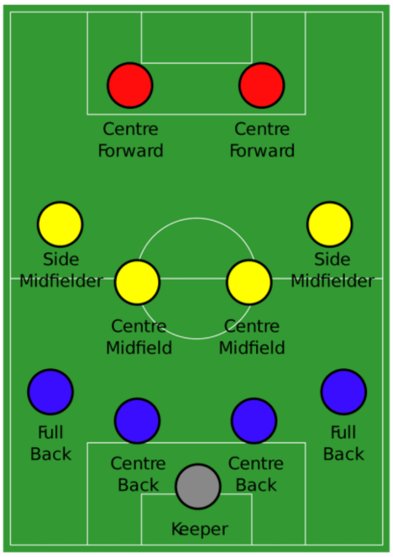


Figure 14: Formation used for this project

This process makes model has a limit that cannot capture each premier team's tactical strength. Improved process which considers each football team's player resources and tactics will accurate higher accuracy.

9.3 Select new features

Player's stats(shootings, dribble, passing, etc) and relative record is essential to predict match's win-lose. If models added more features then, it would have been more convincing. For instance, weather condition on match's date, Premier team's financial strength, player's condition etc. But our project can't add these features or more for realistic reasons: Ambiguous standardization of data, shortage of data(or even there's non-data) or new method of applying data must be devised. If new model improves these problems, model will show higher performance.

Figure 15: New Features that can be used (Starting from the top, director, each team's foundation, weather, sponsor)

9.4 Prediction of Future Match

For future game prediction, like 21-22 season, it is impossible to have the statistics and features for the game that has not happened. Therefore, creating some new variable should be necessary so that features needed for prediction are available for the future game. However new variables should have similar numerical value to the previous data. To solve this problem, the mean values should be calculated for all the variables. Using these data, matches on 21-22 season can be predicted with our model.

Our algorithm model should be able to decide which is home team and which team is away team. English premier league has 2 turns. If first game was home game, then the second game should be away. However, there was no need in making another variable for deciding the place the match was going to performed due to the fact that our model compares A team and B team. A team is home team and B team is the away team. Therefore, by imputing the match data like [Manchester united, Manchester city], the machine will perceive Manchester united as home team. Our model is not implemented yet but we would like to make away team penalty function. This function will be implemented by subtracting variable results of away team. This will show more reality that home team is superior to visitant team.

9.4. Fully Differentiable Model

Another area of future work lies in devising end-to-end trainable network. Although the model is end-to-end runnable, as of now, Premier league prediction model cannot be end-to-end trained. The reason is that model needs pre-data cleansing process and operation needs to be performed with residual schedule. The motivation for an end-to-end trainable network is that we want match's prediction without any additional process. Data inserting process(differentiate main player, sub player, other player), applying residual schedule processes are 2 big parts that disturbs end-to-end model. There are a few ways to combat this problem: first, instead of applying. An alternative to this approach is use reinforcement learning to perform region proposals.

References

[1] Yi Jae Hyun, Lee Soo Won – Prediction of English Premier League Game Using an Ensemble Technique.

[2]] Swung Hwan Gu, Hyun Soo Kim, and Seong Yong Jang," A Comparison Study On the Prediction Models For the Professional Basketball Game,"

[3] Andreas Groll, Thomas Kneib, Andreas Mayr, and Gunther Schauberger, "On the Dependency of Soccer Scores – A Sparse Bivariate Poisson Model for the UEFA European Football Championship 2016," Journal of Quantitative Analysis In Sports, Vol.14, No.2, pp.65-79, 2018.

[4] Raschka, Sebastian, and Vahid Mirjalili. Python Machine Learning, 2nd Ed. Packt Publishing, 2017.

Benjamin O. Tayo, Machine Learning Model for Predicting a Ships Crew Size, https://github.com/bot13956/ML_Model_for_Predicting_Ships_Crew_Size.

[5] Paul Johnson, Extending Extending R-squared beyond ordinary least-squares linear regression, https://www.slideshare.net/pcdjohnson/extending-rsquared-beyond-ordinary-leastsquares-linear

[6] The use of machine learning in sport outcome prediction: A review

Tomislav Horvat Josip Job

Wiley Interdisciplinary Reviews: Data Mining and Knowledge DiscoveryVolume 10, Issue 5

30 June 2020

[7] Machine Learning in Sports: Identifying Potential Archers

RM Musa, Z Taha, APPA Majeed, MR Abdullah - 2019 – Springer

[8] Sports data mining technology used in basketball outcome prediction

C Cao - 2012 - arrow.tudublin.ie

regression-95949488