NBA Prediction Experiments

Kecheng Chen , Ruinan Xu, Xidong Wu

[a]*Department of Civil and Environmental Engineering, University of California, Berkeley*

Abstract

This project examines the feasibility of predictions for players' offensive performance based on college statistics, last game's result for teams from the same division, and team winning percentage in the whole season. In this project, we employed tools such as Train-test split and Cross-validation, prediction models such as Linear Regression, Ridge Regression, LASSO Regression, Logistic Regression, Decision Trees, Random Forests, K-neighbors, Gradient Boosting, Clustering to help us better understand those data. This paper concludes processes of forming models, related discussion and future work.

Keywords: NBA, EDA, Data Cleaning, Prediction, Modeling

Presentation Video:
https://drive.google.com/file/d/1sbqDS4dYGCqIDVGVJRBwgBUMSMX8i1aU/view?usp=sharing
Project Narrative excluding figures and tables (6 Pages):
https://docs.google.com/document/d/17LJRnNvSh-_ff2Ebq6EPWlZuJ73lspaxv7_6eeqGxBk/edit?usp=sharing

1. Introduction

As NBA teams have begun to embrace data analytics, more emphasis has been placed on performance prediction relying on existing sources. In this project, we have explored some interesting topics while using tools we have learned during class, and get some pretty good results. The first topic we are interested in is college and college players. We want to know the differences between universities in the development of players and the prediction of the future development of college players. The second topic is to predict the outcome based on previous games. Since we have the results of the first three games for teams in the same division in one season, we want to use this result to predict the result of the fourth game. The third topic is to predict team winning percentage based on team statistics. We want to know which statistics has the greatest impact on the winning percentage. Furthermore, we also explored the trend of three-point shooting in the league and its relationship with the team's offensive efficiency.

2. Experiments and Results

2.1 EDA and Data Visualization

2.1.1 Which college's players have the best offensive performance

College.csv is a csv file with a shape of 4274 rows and 33 columns. Players information contains their personal information, Playing career time, position, college as well as their performance in NBA and in NCAA.

Many players start their NBA career after college and college plays a key role in the development of players. Firstly, we wonder which college has the best performance in training

players. It is helpful for the high school players to choose their ideal college and regulate their career path.

Data Cleaning: The row with the null data in ‘college’ columns is dropped. And the columns 'NBA_g_played' and ‘NBA__3ptpg’ are chosen to evaluate players’ performance. To quantify the performance of college, we explore it from these respects:

1. Which college has the highest total score?
2. Which college has the highest average score?
3. Which college trains the most NBA players?
4. Which school's players are the best at making three-pointers?

The data is grouped according to the question above and sorted. Then the results are merged.

college	Total Points	Total Points Rank	Avg Players	Ave Players Rank	Avg Points	Ave Points Rank	Sum 3pt	Total 3pt Rank	Avg 3pt	Ave 3pt Rank
University of North Carolina	34253	1	87	2	393.72	111	25.40	3	0.2920	147
University of California, Los Angeles	33460	2	87	3	384.60	118	25.80	2	0.2966	145
University of Kentucky	27566	3	96	1	287.15	217	32.60	1	0.3396	107
Duke University	22801	4	67	4	340.31	151	23.7	5	0.3537	98
University of Kansas	22028	5	67	5	328.77	167	17.1	7	0.2552	164
All College Ave	1631		5.96		219.47		1.18		0.1764	

Table-1 Merged results.

Conclusion: Table 1 shows that University of North Carolina, University of California, Los Angeles, University of Kentucky, Duke University and University of Kansas ranks top in all rankings and their scores largely outweigh the average level in different aspects.

2.1.2 Three Pointer and Offensive Rating

There has been a lot of talk about increasing the number of 3-pointers you take to improve your team's offensive efficiency. In this part, we are interested in the three-pointers trend in the NBA

and whether increasing team three-point attempts increase team offensive rating by using the dataset *2012-18_teamBoxScore.csv*.

Data Cleaning: Firstly, we can go through a data cleaning process to get a single dataframe with each row representing a team's final statistics in one season. To help us select group data by seasons, we change the type of game date column to datetime type. Since data given is a record of six seasons per game, we need to group data by teams. We create a list with 6 elements where each element is a dataframe representing one season final statistics of 30 teams. Finally, we can stack those dataframes and get a single dataframe with each row representing a team final statistics in one season. Group by different seasons, we can get all team average three-pointers performance(three-pointer attempts, three-pointer shot made and three-pointer percentage).

Data Visualization: By drawing a bar plot of statistics of three-pointer performance, we can see that teams in the NBA tend to attach more importance to three-pointer shooting, as shown in fig-1. Also, all teams' offensive ratings are increasing.

season	team3PA	team3PM	team3P%	teamOrtg
2012-2013	19.961748	7.160976	0.355262	105.911280
2013-2014	21.534146	7.745528	0.357221	106.696656
2014-2015	22.414634	7.845528	0.347712	105.719675
2015-2016	24.073171	8.517480	0.352089	106.484495
2016-2017	26.997154	9.654065	0.356613	108.892624
2017-2018	29.000407	10.491057	0.361020	108.648174

Table-2 NBA all team three-point performance

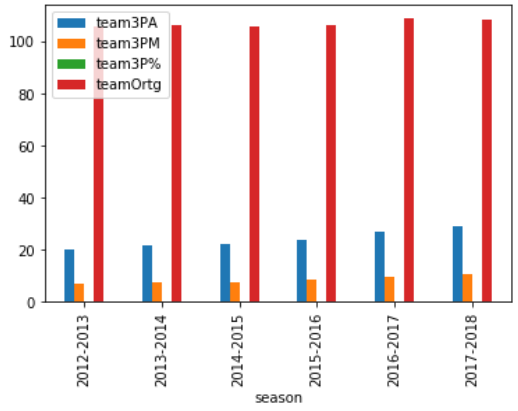


Fig-1 Bar plot of three-pointer performance

Secondly, we draw a scatter plot of 30 teams three-pointer performance vs. team Offensive rating in 6 seasons, where x-axis is Team three-pointer attempts, y-axis is Team offensive rating, color represents Team three-pointer shots made. It turns out that teams who shoot more three pointers or have higher three-pointer percentages have higher offensive ratings.

Conclusion: Finally, we look at the correlation coefficient of team two-point attempts and team three-point attempts with team offensive rating. Result shows three-pointer attempts are positively correlated with team offensive rating , while two-point attempts is negatively correlated with team offensive rating. It turns out that if a team wants to get a higher offensive rating, they should increase their three-pointer attempts and reduce their two-pointer attempts. This result is counterintuitive, since generally speaking, two-pointer percentage is relatively higher compared with three-pointer percentage.

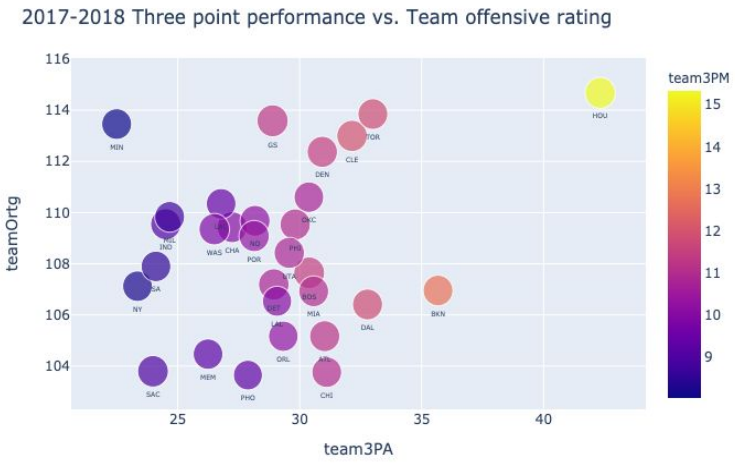


Fig-2 2017-2018 three-pointer performance vs. team offensive rating

	teamOrtg
team2PA	-0.451923
team3PA	0.469426

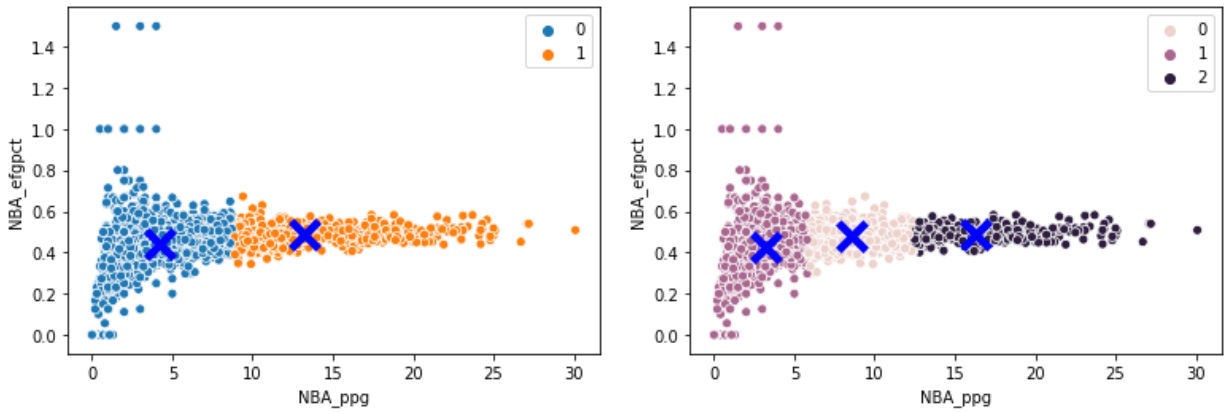
Table-3 Correlation of team offensive rating vs. team 2-poiner and 3-pointer attempts

2.2 Prediction Models

2.2.1 Prediction of players' performance in NBA from their data in NCAA

After entering colleges, players' scores in NCAA have a significant effect on decision by the NBA teams. The team managers also want to know how to choose potential outstanding players. Therefore, it is important to predict players' career prospects based on their performance in NCAA. For convenience, it is regarded as a classification problem.

Data Cleaning: The First challenge we met was how to transfer it to be a classification problem.We choose some features as classification criteria and clustering method is used to help group data. Because the total scores are affected by the length of career, we prefer the features associated with scores per game to quantify the performance of players. "NBA_ppg" ("Points per game") is used as the metric where outstanding players "Wilt Chamberlain" and "Michael Jordan" rank the top two. In addition, part 2.4 denotes the importance of features "NBA_efgpct". Then, the clustering method is used to help us group data because we need to transfer players' numerical scores to levels. Initially, we thought the clustering method would separate data according to these two features. Results (Fig. 3) shows that we could only consider "NBA_ppg" features to separate data whether data are divided into three groups or two groups. "NBA_efgpct" does not work in the clustering. Before fitting the data, we check whether the data contain null value.



(a) (b)

Fig-3 Results from clustering with : (a) 2 groups; (b) 3 groups.

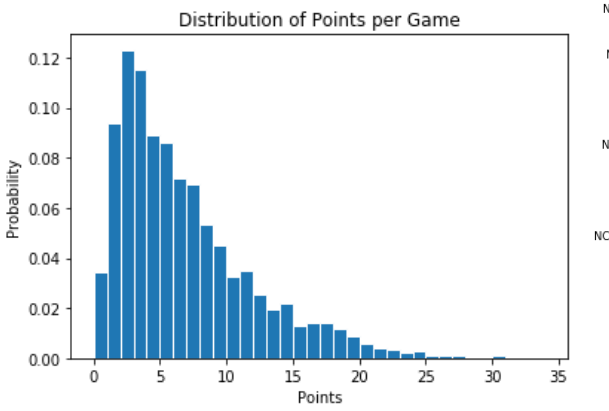


Fig-4 Distribution of points per game

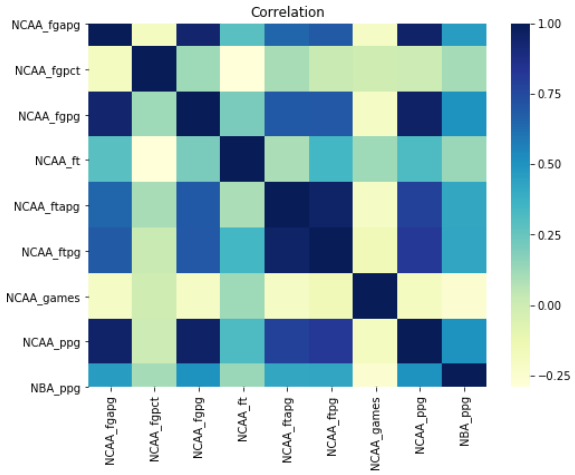


Fig-5 Correlation between NCAA features and pgg

From Fig. 4, we could see although the range is from 0 to 35, the scores are mainly situated in the lower points interval. Fig. 3 shows that the boundary lies in roughly 10. Therefore, the player with points higher than 10 points is a good player. Otherwise, the player is a bad player.

Later, we consider multi-classification. From fig. 4 distribution reaches the peak at 3, so the players with scores not larger than 3 are regarded as C class. Players with scores larger than 20 are outstanding and regarded as A class. and so another threshold is 20.

Feature Engineering: Another challenge is to choose features. From Fig. 5, we could see that the correlation between features of NCAA and the NBA_ppg are lower. It is a big challenge because we cannot directly select the features. So we use a forward selection algorithm to select features.Forward selection is a type of stepwise regression which begins with an empty model and adds in variables one by one. In each forward step, you add the one variable that gives the single best improvement to your model. The detail code is presented in the jupyter notebook.

Model fitting: At this part, Logistic Regression, Random Forest and Decision Tree are applied. Logistic Regression is a linear classification method while Random Forest and Decision Tree are nonlinear methods. We use different kinds of methods to understand their property.

Application and conclusion: Table 3 shows in binary classification Logistic regression has the best performance among three methods. After the feature selection, the results increase slightly for all methods. From the selected features, surprisingly, only one feature 'NCAA_fgpg' is selected in Random Forest and Decision Tree, while 7 / 8 features are selected in logistic regression. The selection of 'NCAA_fgpg' satisfies our common sense. In multi-classification, the feature selection plays a more obvious role for random forest and decision trees and Random Forest has the best performance. The selected results are the same. Compared with Random Forest and Decision Tree, Logistic Regression has a good performance in binary classification. But for multi-classification, the Random Forest outweighs it. It should be mentioned that Random Forest could reduce the error caused by overfitting compared with other two methods. So it always has the best performance in the test accuracy.
In addition, initially I think 'NCAA_games' and 'NCAA_ppg' should be selected because they contain the information about the total scores players get but they are not selected.

Table-3 Prediction results under two group

	Train Set and Valid Set		Test Set
	Without Feature Selection	[Selected Feature Index[1]] With Feature Selection	
Logistic Regression	0.804	[0, 1, 2, 3, 4, 6, 7] 0.807	0.846
Random Forest	0.796	[2] 0.803	0.846
Decision Tree	0.706	[2] 0.803	0.842

Tabel 4. Prediction results under three group

	Train Set and Valid Set		Test Set
	Without Feature Selection	[Selected Feature Index[1]] With Feature Selection	
Logistic Regression	0.742	[1, 2, 4, 5, 6, 7] 0.743	0.676
Random Forest	0.712	[2] 0.729	0.683
Decision Tree	0.602	[2] 0.729	0.676

Note: 1. Feature Index: 0 - NCAA_fgapg; 1 - NCAA_fgpct; 2 - NCAA_fgpg; 3 - NCAA_ft; 4 - NCAA_ftapg;5 - NCAA_ftpg; 6 - NCAA_games; 7 - NCAA_ppg

2.2.2 Prediction of game results based on previous games for teams from the same division

During the regular season, a team needs to face opponents in its own division four times. Then a question comes to our mind, can we predict the last game's result based on the previous three games? If a certain relationship exists, the final result of two teams may be determined before the last game starts. It also provides a reference for the team's manager to do some improvements.

Data cleaning: In this part, data in the file *2012-18_teamBoxScore.csv* are utilized. The shape of the data is (14758, 123). There are NaN values in columns offLNm3 and offFNm3. Types and descriptive statistics of data are checked.

EDA shows that each game is described twice in the perspectives of two teams, so a specific sequence of names is employed to filter the data. Only two teams from the same division have remained. Some useless columns with type as an object, like seasTyp, and those with NaN values are deleted. Since we wish that data from 2012 to 2016 are selected as training data and those in 2017 are selected as test data, this condition is used to filter data to get corresponding datasets. For training and test datasets, X are final features, for each of which is the weighted combination of corresponding features in the previous three games for teams from the same division, and Y is the last game's result.

Feature Engineering: LightGBM, a gradient boosting framework that uses tree-based learning algorithms, is employed to choose features, because it focuses on the accuracy of results and takes lower memory to run. Common and effective classifiers like k-neighbors, random forest, and gradient boosting are compared with respect to the validation result.

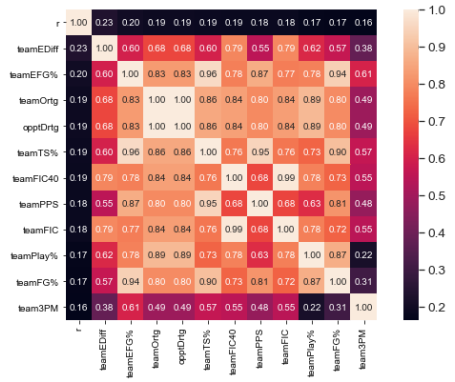


Fig-6 Correlation matrix focused on team result

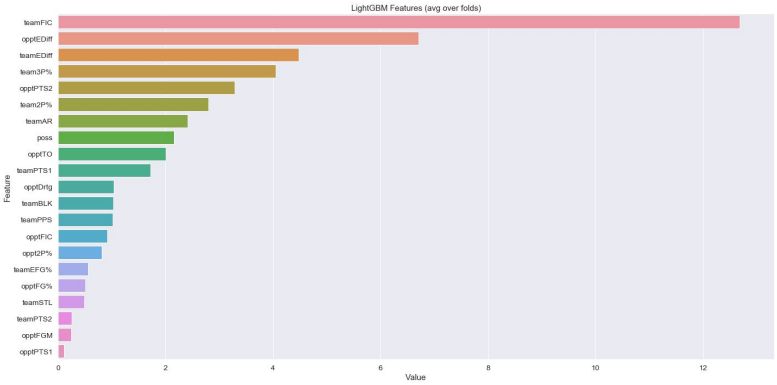


Fig-7 LightGBM features

Characteristics of methods: Gradient boosting builds one tree at a time, where each new tree helps to correct errors made by previously trained trees. It performs better than random tree if parameters are tuned carefully. K nearest neighbors stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions).

Model fitting and visualization: We assumed the last game's result is only related to the previous three games. After feature selection, final training and test datasets are formed based on these features. Also these features are standardized. Since k-neighbors get the best validation result, it is applied to do the final fitting and prediction work. **The final test accuracy is 0.65.** It is not good but not that bad. Based on our model, it is easy to know if a certain feature has a positive or negative effect on the final result.

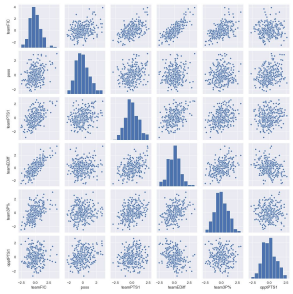
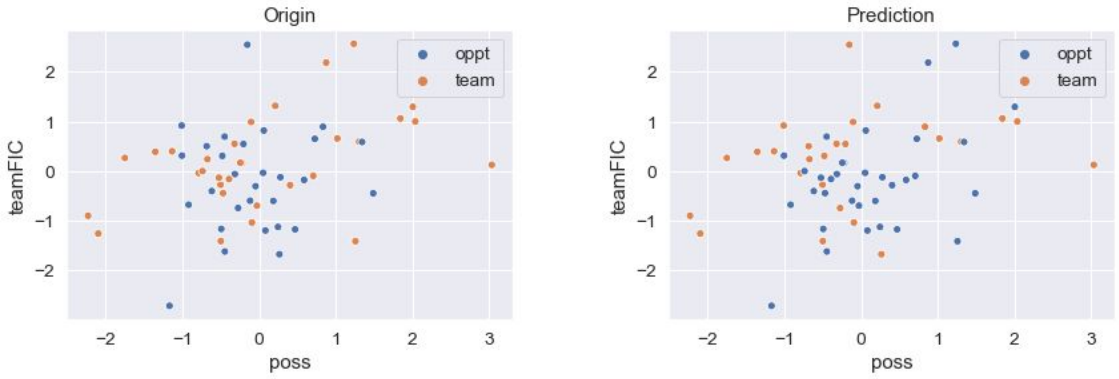


Fig-8 Pairwise relationships

Classifier	K-neighbors	Random Forest	Gradient Boosting
Training Accuracy	0.538	0.542	0.545
Validation Accuracy	0.683	0.466	0.567
Test Accuracy	0.65		

Table-5 Accuracy comparison

Conclusion: The efficiency differential for the team (teamEDiff) of the previous three games has the largest correlation coefficient with the last game's result, which is about 0.3. However, based on the feature selection, the floor impact counter for team (teamFIC) of the previous three games has the largest feature importance value. Points scored by team (teamPTS) of the previous three games are thought to be useful, but its feature importance value is 0. As mentioned above, features for the previous three games all have small correlation coefficients with the last game's result. It disproves our assumption that we cannot predict which team could win in the fourth game very well only based on what we got from the previous three games. So audiences still can hope their favorite teams to reverse the situation.



(a) (b)

Fig-9 Comparison of original test data (a) and prediction result (b)

2.2.3 Winning Percentage Prediction

Predicting the winning percentage of NBA teams has always been a fascinating topic. In this part, we are trying to predict team season winning percentage based on team statistics by using *2012-18_teamBoxScore.csv*. Team statistics such as team average offensive rating and defensive rating may be useful to this prediction.

Data Cleaning: We can convert Loss-Win to 0-1, which will make it easier for us to compute the winning percentage. To help us select data by seasons, we change the type of game date column to datetime type. Since data given is a record of six seasons per game, we need to group data by teams and using mean aggregation function, thus team results column turns to team winning percentage. We create a list with 6 elements where each element is a data frame representing one season final statistics of 30 teams. Finally, we can stack those dataframes and get a single dataframe with each row representing a team final statistics in one season.

Feature Engineering: After data cleaning, we need to select features that are most useful to predict team winning percentages. By looking at the correlation coefficients with respect to the winning percentage column, we found the most useful feature: team efficiency differential, which has a correlation coefficient 0.97 with team winning percentage. which means these two are highly linearly correlated. So we can fit them into a line and try to predict one use the other.

Model fitting and visualization: To fit these data in a line, we need first split the data into training data and testing data. We choose Linear Regression, Ridge Regression and LASSO Regression to fit our model. Difference between these three models is that both Ridge Regression and LASSO Regression add regularization terms, which will penalize large values of coefficients. Advantage of LASSO is that it can set low values of coefficients to zero, which makes us focus on features with higher correlation. After using cross validation, we can test the model performance (R-squared and root mean squared error(RMSE)) on the testing data. Best alpha in Ridge Regression is 1 while best alpha in LASSO Regression is 0.001. Fig-10 shows all data points(including training and testing data) and fitted model. We can see all three models perform well in this special case. Finally, we get R-squared and RMSE shown in Table 6.

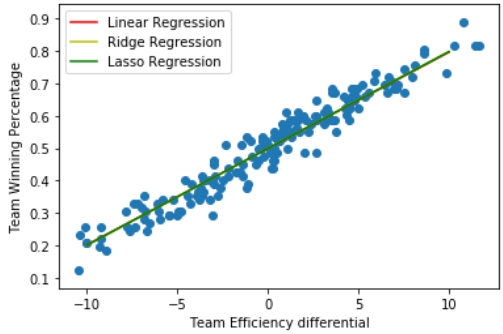


Fig-10 Fitted Regression Model

	Linear Regression	Ridge Regression (alpha=1)	LASSO Regression (alpha=0.001)
R-squared	0.9636	0.9646	0.9644
RMSE	0.00116	0.00113	0.00113

Table-6 Accuracy comparison on test data between models

Application and conclusion: In short, we can use Team Efficiency Differential to predict Team Winning Percentage. For example, if we get the Team Efficiency Differential of one team at half season, by assuming this team will keep such statistics throughout the whole season, we can predict Team Winning Percentage of the whole season. Or if one team at the beginning of the season wants to set a Team Winning Percentage goal, by using our model, they can know what kind of Team Efficiency Differential they must get to finally achieve their goal.

In this particular problem, the most useful feature is Team Efficiency Differential, which has a 0.97 correlation coefficient with Team Winning Percentage. At first, we supposed features such as Team Offensive Rating and Team Defensive Rating maybe useful, but since these features are highly correlated with Team Efficiency Differential(Team Efficiency Differential = Team Offensive Rating - Team Defensive Rating), it turns out that if we use those additional features the model won't perform well due to collinearity.

3. Discussion

3.1 Limitations and Future Work

In section 2.2.1, binary classification and multi-classification are conducted and Logistic Regression, Random Forest, Decision Tree are applied. After feature selection, the results increase slightly for all methods in binary classification and the results increase dramatically in mul-classification. The limitation in this question is that we just use the raw feature and do not extract new features to improve the model. In addition, all data is offensive statistics. If we have data including defensive statistics, we could evaluate players more comprehensively.

In section 2.2.2, a shortage exists for the LightGBM used which is sensitive to overfitting and can easily overfit small data. So for improvement, other methods for feature selection should be tried. The performance of teams is pretty unstable in the previous three games, so the prediction accuracy is not that high. In the future, our model should take preseason and postseason games' data into consideration to select more suitable features.

In section 2.2.3, the major limitations of our model is that our model reflects the immediate relationship between Team Efficiency Differential and Team Winning Percentage, which means we need to use the Team Efficiency Differential for a whole season to predict the Team Winning Percentage of one season. If we want to predict the Team Winning Percentage at the end of the season while the season is not over, we have to use the Team Efficiency Differential of the team at this point and assume this team will keep such statistics throughout the whole season. This may not be a valid assumption, because a team can go through up-and-downs during one season. So Team Efficiency Differential may not remain the same. If we have more season data instead of just 6 seasons final statistics, we can better test our model accuracy.

3.2 Ethical Problem

In section 2.2.1, we have encountered an ethical problem that it may not be appropriate to use only how many points they have scored to judge players. Such judgement criteria may lead to players practice shooting all day long while ignoring the fundamental essence of the basketball games. It is even possible for players with higher scores to discriminate against players with lower scores. There are some players who may not have decent points per game, but play a significant role to this team. To solve this problem, we recommend that teams can only use our model as part of their assessment of their players and do not use our model as whole judgement criteria.