NBA Player Awards and Team Performance Prediction

GROUP 19

Zhan Shu
UNI: zs2584

Yuan Dou
UNI:yd2676

Yingqi Ma
UNI:ym2926

***Abstract*—In this paper, given the performance and award data of players and teams, we analyze the dataframe to get the feature and stats. With the feature correlation for candidate players and teams, we build 5 machine learning models to predict the awards of players and playoff teams, and make comparisons between different ML models for different tasks.**

I. INTRODUCTION

The basketball competition is one of the most watched sports activities in the world. With the advancement of technology, basketball games and player statistics can be more easily stored and therefore can be used for prediction using artificial intelligence techniques such as machine learning. The National Basketball Association (NBA), as the most prestigious basketball league, has the best basketball players and largest fan base in the world. As of 2022 season, the NBA ended the regular season with over 66 million followers on Instagram, making it the most-followed professional sports account. [1] Every fan has a favorite player and team. So prediction about the game brings a lot of fun to fans and improves their viewing experience.

Using the players and teams performance data from *basketball-reference.com*, we are able to predict the player awards and whether the team can enter the playoffs. In this project, we process the data and find the feature correlation for candidate players and teams. Then use different machine learning methods(logistic regression, gaussian naive bayes, random forest, decision tree and k-nearest neighbors) to predict the all-star selections, awards voting results (Rookie of the Year, Sixth Man of the Year, Most Valuable Player, etc), and teams that can enter the playoffs of current year. After that, we make comparisons between different ML models for different tasks and get the best result.

II. RELATED WORKS

In *Predicting NBA Games Using Neural Networks*[2], this paper applies a machine learning approach, Support Vector Machine (SVM), to predict the playoff results of the NBA, using the features composed of the historical statistics of regular seasons. Compared to our project, this paper only uses one machine learning model (SVM) to train the training data with cross validation to build up the classifier. This may lead to a higher error rate because of unpredictable factors.

In *A Novel Approach to Predicting the Results of NBA Matches*[3], this paper's approach uses both the team's own stats, but also the data known about the opposing team to make that prediction. And the findings show that utilizing the information about the opponent improves the error rate observed in the predictions. As opposed to our project that merely takes into account the player's or team's own performance and statistics in predicting the result, this paper gets a more accurate prediction in specific matches.

III. CURRENT WORKS

A. Exploratory Data Analysis

The purpose of implementing Exploratory Data Analysis (EDA) before we start building the learning model is we need a clear layout of our data. Imagine when facing a large amount of data, one would not directly select out the features that are important and may be confused by those statistics. That is when we need the EDA to clear things out and get the clean data before furthermore complex analysis. Exploratory Data Analysis is a vital process that does initial investigation on data to show features performance relationship, hypothesis correctness and assumption accuracy.

To start the process, we first need to decide what datasets we want to use. We choose the datasets from Kaggle [4] and the datasets we used for EDA are player_totals.csv, player_award_shares.csv, and team_stats.csv. In the player total dataset, we have every player's total statistical data (including points, field goal percentage, assistant, rebounds, etc.) for each season from 1980 till 2022. Player award shows each year's awards candidate voting result and basic performance data. Team statistics shows the team statistical performance data for each year and indicate whether this is a playoff team.

1) Player award prediction data

When visualizing the datasets, we first need to preprocess the data into a preferable format. Therefore, we use the player_award_shares.csv file as the template and stack those candidates' statistical performance data to the template.

We want to plot out each year's comparison between candidates' statistical performance using a bar chart. As shown below is an example of the 2022 and 2021 DPOY award. The top figure 1 is the total blocks data of each DPOY candidate in 2022 season showing as y-axis. Figure 2 is the total blocks data of each DPOY candidate in 2021 season. Both figures have the leftmost candidate on the x-axis as the winner of DPOY. From the plot we can see that the winner of 2021 has especially higher blocks than other candidates, whereas the winner of 2022 has the least. But if we give more data from other years, we can see that block number is a high-related data with DPOY award.

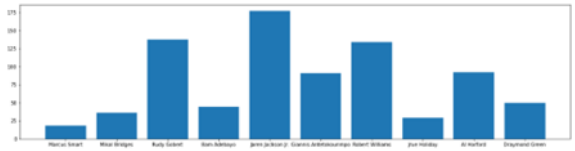

Figure 1. DPOY candidates block stats. 2022

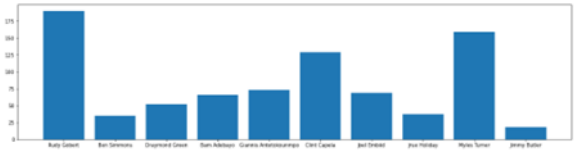

Figure 2. DPOY candidates block stats. 2021

2). Team statistics visualization

We then visualize data of all the playoff teams and non-playoff team performance data. Below are two examples that are typical. Figure 3 shows the points per game distribution of pf team and non-pf team. It is clear that the playoff team (orange, right) has higher values. Therefore, this is the kind of feature that we want to keep. Figure 4 shows the opposite way. Two teams have the same rebound per game. Therefore, we can consider dropping this feature for reducing complexity of data.

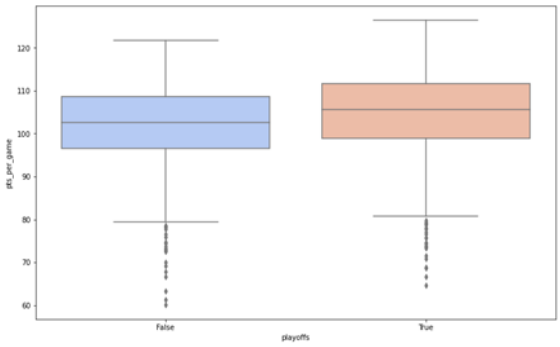

Figure 3. Playoff teams vs. non-pf team points per game

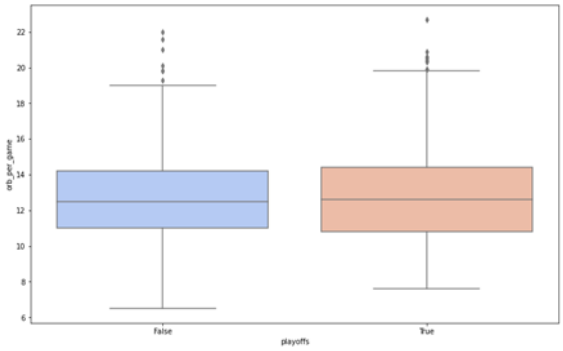

Figure 4. Playoff teams vs. non-pf team rebound per game

B. Data Preprocessing

The purpose of data preprocessing is to simplify data density and increase accessibility for our data shape. For instance, we want some input output of the same shape for stacking data and modeling them. Data preprocessing has four important steps: data quality check, data cleaning, data transformation and data reduction. After these processes, we would have the desired data that could enter the building model and training system.

We first decide the datasets we want to use. Same folder in the Kaggle dataset we choose the team_summaries.csv, team_stats_per_game.csv files. Team summaries file is the more depth analyzed data of teams for each season. Team stats per game file

shows the data of per game data of all NBA teams for each season.

First, we stack those two csv files together and check their features. We can see some feature columns have high correlation. Therefore, we drop them and keep the else feature performance data and get the desired dataset. We also observe the data have a row called League Average which shows the average statistical data of the whole season. We don't need this row showing each year, so we remove those rows as well for simplicity.

The following figure 5 and Figure 6 are the heatmaps we plotted with python matplot. It shows the correlation of each feature of teams and players and helps to make the dropout decision.

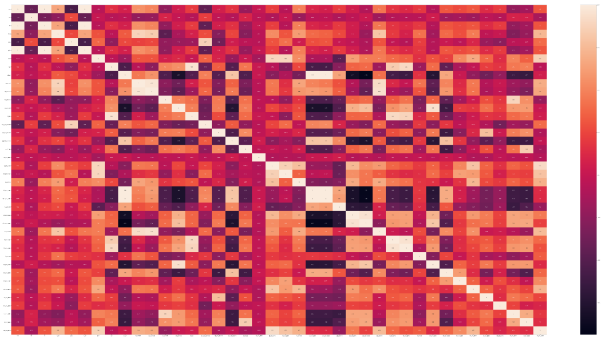

Figure 5. Heatmap of team statistic features

By observing the above heatmap, the following conclusions can be made:
"mov", "srs" and "n_rtg" have a high positive correlation with each other.
"pace" has a high positive correlation with "fg_per_game" and "fga_per_game".
"f_tr" has a high positive correlation with "fta_per_game" and "ft_fga".
"x3p_ar" has a high negative correlation with "x2pa_per_game" and "x2p_per_game", and a high positive correlation with "x3pa_per_game" and "x3p_per_game""x3p_per_game".
"ts_percent" has a high positive correlation with "x2p_percent" and "efg_percent".
"x3p_percent" has a high positive correlation with "x3pa_percent".
"ft_per_game" , "fta_per_game", "ft_fga" have a high positive correlation with each other.
"fg_per_game" has a high positive correlation with "fga_per_game".

Therefore, we decided to drop "mov", "srs", "pace", "f_tr", "x3p_ar", "ts_percent", "x3pa_percent", "fga_per_game", "ft_fga", "fga_per_game".

Definition of features of team statistics:
sos- Strength Of Schedule, which represents a team's average schedule difficulty faced by each team in the games that it's played so far or for all season. [5]
o_rtg - Offensive Rating, representing points scored per 100 possessions. [5]
d_rtgDefensive Rating, representing points allowed per 100 possessions. [5]
n_rtg - Net Rating, measuring a team's point differential per 100 possessions. [6]
e_fg_percent - Effective Field Goal Percentage, measuring field goal percentage adjusting for made 3-point field goals being 1.5 times more valuable than made 2-point field goals. [6]
tov_percent - Turnover
orb_percent - Offensive Rebounding Percentage, representing the percentage of available offensive rebounds a player or team obtains while on the floor. [6]
opp_e_fg_percent - Opponent Effective Field Goal Percentage, representing an opponent's field goal percentage that is adjusted for made 3 pointers being 1.5 times more valuable than a 2 point shot. [6]
opp_tov_percent - Opponent Turnover Percentage, representing the number of turnovers an opponent averages per 100 of their own possessions. [6]
opp_drb_percent - Opponent Defensive Rebounds, representing the percentage of opponent's number of defensive rebounds collected. [6]
opp_ft_fga - Opponent free throws per field-goal attempt
fg_per_game - Field Goal per Game
fg_percent - Field Goal Percentage
x3p_per_game - 3 Point Field Goals per Game
x3p_percent - Percent of 3 Point Field Goals Attempted. Representing the percentage of a team's 3 point field goals attempted. [6]
x2p_per_game - 2 Point Field Goals per Game
x2p_percent - Percent of Field Goals Attempted (2 Pointers). Representing the percentage of field goals attempted by a team that are 2 pointers. [6]
ft_per_game - Free throw per Game
ft_percent - Percent of Team's Free Throws
orb_per_game - Offensive Rebounding per Game
drb_per_game - Defensive Rebounding per Game

trb_per_game - Total Rebounding per Game
ast_per_game - Assists per Game
stl_per_game - Steal per Game
blk_per_game - Block per Game
tov_per_game - Turnover per Game
pf_per_game - Personal Fouls per Game
pts_per_game - Points per Game

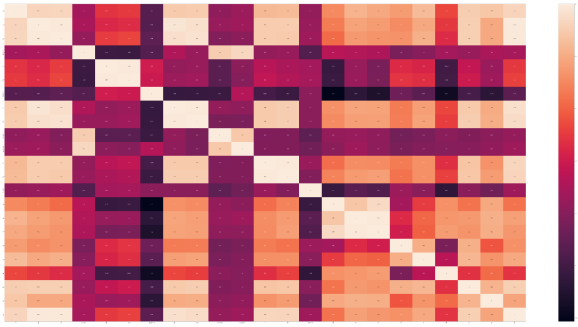

Figure 6. Heatmap of player statistic features

By observing the above heatmap, the following conclusions can be made:
“mp” is highly correlated with “fg”, “fga”, “tov”, “pf”, “pts”.
“fg” and “fga” are highly correlated.
“x2p” and “x2pa” are highly correlated.
“x3p” and “x3pa” are highly correlated.
“ft” and “fta” are highly correlated.
“drb” and “trb” are highly correlated.
“pts” is highly correlated with “mp”, “fg”, “fga”, “x2p”, “x2pa”, “ft”, “fta”, “tov”.
Therefore, we decided to drop “mp”, “fga”, “x2pa”, “x3pa”, “fta”, “trb”, “pts”.

Definition of features of player statistics:
fg - Field Goal.
fg_percent - Field Goal Percentage. Representing the percentage of field goal attempts that a player makes.[6]
x3p - 3 Point Field Goals
x3p_percent - Percent of 3 Point Field Goals Attempted. Representing the percentage of a team's 3 point field goals attempted. [6]
x2p - 2 Point Field Goals
x2p_percent - Percent of Field Goals Attempted (2 Pointers). Representing the percentage of field goals attempted by a team that are 2 pointers
e_fg_percent - Effective Field Goal Percentage, measuring field goal percentage adjusting for made 3-point field goals being 1.5 times more valuable than made 2-point field goals. [6]
ft - Free throw
ft_percent - Percent of Team's Free Throws Attempted. Representing the percentage of a team's free throws attempted. [6]
orb - Offensive Rebounding
drb - Defensive Rebounding
ast - Assists
stl - Steal
blk - Block
tov - Turnover
pf - personal fouls

C. Team Performance Prediction

It was not until June 1979 when the NBA adopted the three-point line for the 1979-80 season. [] Since then, NBA games have become more similar to modern games today despite some minor rule changes each year. Therefore, team data from 1980-2022 is used for training, while team data of season 2022-2023 is used for prediction. Since the features are not on the same scale, we first scale each feature by its standard deviation, so each scaled feature has a mean of zero, using the equation:

$$X_{scaled} = \frac{x - \mu(x)}{\sigma(x)}$$

In order to determine which machine learning algorithm performs better on the dataset, the training data is split into 70% training data and 30% validation data. Then the data is used to train 5 classification models, logistic regression, gaussian naive bayes, random forest, decision tree and k-nearest neighbors. After training, the models are evaluated using the validation data by number of correct predictions / size of the validation data.

```
lr = LogisticRegression(random_state=0).fit(X_train, y_train)
print(lr.score(X_test, y_test))
```

```
0.9211956521739131
```

```
nb = GaussianNB().fit(X_train, y_train)
print(nb.score(X_test, y_test))
```

```
0.9157608695652174
```

```
rf = RandomForestClassifier(max_depth=10, random_state=0).fit(X_train, y_train)
print(rf.score(X_test, y_test))
```

```
0.9184782608695652
```

```
dt = DecisionTreeClassifier(random_state=0).fit(X_train, y_train)
print(dt.score(X_test, y_test))
```

```
0.8586956521739131
```

```
knn = KNeighborsClassifier().fit(X_train, y_train)
print(knn.score(X_test, y_test))
```

```
0.8722826086956522
```

Figure 7. Accuracy score of 5 classification models

From the above result, we can see that logistic regression, gaussian naive bayes and random forest performs better on the task than decision tree and k-nearest neighbor. Since the logistic regression model achieved the highest accuracy score on the validation dataset, the model is trained again using all of the data from 1980-2022. The model is then used to make predictions with the team statistics of the current season. Instead of a binary output, we utilized predict_proba of the scikit_learn library to output the probability of each team getting into the playoffs. As the season goes on, the probability of each team getting into the playoffs will fluctuate as team statistics change. Below is the predicted probability of the logistic regression model with input of the latest team statistics.

team	playoffs
Atlanta Hawks	0.918265
Boston Celtics	0.992032
Brooklyn Nets	0.871761
Chicago Bulls	0.534361
Charlotte Hornets	0.591622
Cleveland Cavaliers	0.999653
Dallas Mavericks	0.773474
Denver Nuggets	0.911193
Detroit Pistons	0.119427
Golden State Warriors	0.734514
Houston Rockets	0.21314
Indiana Pacers	0.503872
Los Angeles Clippers	0.708626
Los Angeles Lakers	0.851453
Memphis Grizzlies	0.72601
Miami Heat	0.325904
Milwaukee Bucks	0.997425
Minnesota Timberwolves	0.540314
New Orleans Pelicans	0.999323
New York Knicks	0.498536
Oklahoma City Thunder	0.499137
Orlando Magic	0.245366
Philadelphia 76ers	0.964326
Phoenix Suns	0.999703
Portland Trail Blazers	0.149002
Sacramento Kings	0.88145
San Antonio Spurs	0.166523
Toronto Raptors	0.883631
Utah Jazz	0.549811
Washington Wizards	0.581965

Figure 8. Predicted probability of each team entering the playoffs

D. Player Awards Prediction

1) MVP Prediction

To predict MVP for the current season, player statistics from season 1980-2022 is used for training, while player statistics of season 2022-2023 is used for prediction. Each feature is scaled to have a mean of zero. The training data is split into 70% training data and 30% validation data. Then the data is used to train 5 classification models, logistic regression, gaussian naive bayes, random forest, decision tree and k-nearest neighbors. Following are the evaluation results of the 5 models.

```
lr = LogisticRegression(random_state=0).fit(X_train, y_train)
print(lr.score(X_test, y_test))
```

```
0.9973736047275115
```

```
nb = GaussianNB().fit(X_train, y_train)
print(nb.score(X_test, y_test))
```

```
0.9094987962355001
```

```
rf = RandomForestClassifier(max_depth=10, random_state=0).fit(X_train, y_train)
print(rf.score(X_test, y_test))
```

```
0.9970453053184505
```

```
dt = DecisionTreeClassifier(random_state=0).fit(X_train, y_train)
print(dt.score(X_test, y_test))
```

```
0.9954038082731451
```

```
knn = KNeighborsClassifier().fit(X_train, y_train)
print(knn.score(X_test, y_test))
```

```
0.9973736047275115
```

Figure 9. Accuracy score of 5 classification models

From the above figure, we can see that logistic regression and K-nearest neighbor achieved the highest accuracy score, while Gaussian Naive Bayes has the lowest accuracy score.

IV. PLANNED EXPERIMENTS

A. Predicted results of MVP

Screenshot of a csv file which contains the probability of each player winning the MVP award, ordered by probability descending.

B. DPOY prediction

Filter out irrelevant features and keep only features which help rating defensive performance of a player, for example, steals, rebounds, blocks, etc. Train 5 ML models on the data, make comparisons and make predictions.

C. ROY prediction

Filter the data, which contains only players with experience <= 2, ie. Only players with less than 2 years of experience in the league are qualified for the

ROY award. Train 5 ML models on the data, make comparisons and make predictions.

D. SMOY prediction

Select only players which plays most of the games as a bench player, ie. games as starters < threshold value. Train 5 ML models on the data, make comparisons and make predictions.

E. DAG

Build an airflow DAG to fetch the data from basketball-reference.com and set it scheduled to run daily at 7pm, so that it is able to fetch the data of player and team and update the prediction on a daily basis.

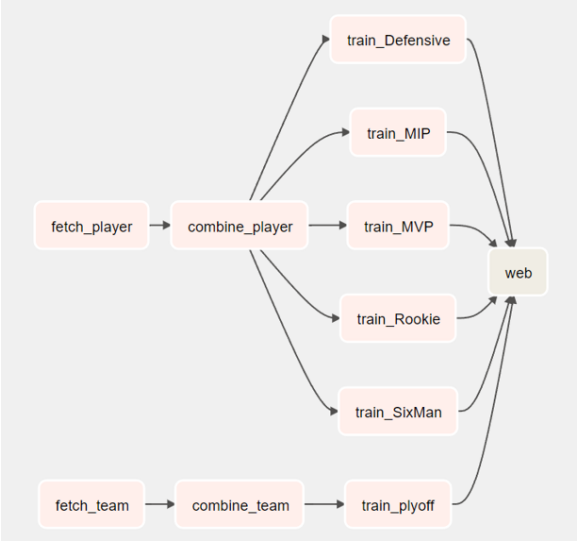

Figure 10. Graph of DAG

F. Develop Web Application

Visualize the data of the team/player stats of the current season on a web server to help fans find the data they are interested in.

Visualize the predicted results based on the selection of player and team using *HTML* and *JavaScript*. Then use *css* to format and beautify the web appearance to make it more attractive.

V. REFERENCE

[1]https://theathletic.com/3500551/2022/04/13/nba-average-viewership-up-19-for-2021-22-vs-last-season-across-abc-tnt-espn/
[2]Loeffelholz, B., Bednar, E. & Bauer, K. (2009). Predicting NBA Games Using Neural Networks. *Journal of Quantitative Analysis in Sports*, *5*(1). https://doi.org/10.2202/1559-0410.1156
[3]Jackie B. Yang and Ching-Heng Lu .(2012). PREDICTING NBA CHAMPIONSHIP BY LEARNING FROM HISTORY DATA.
[4]Kaggle.com https://www.kaggle.com/datasets/sumitrodatta/nba-aba-baa-stats?select=Team+Summaries.csv
[5]Basketball Reference https://www.basketball-reference.com/about/glossary.html
[6]NBA Glossary https://www.nba.com/stats/help/glossary