Building Prediction Models and Visualizing Players Shooting Position for NBA

Author: Chun-Cheng Chang, Anthony Chu, Phyllis Chen

I. Summary of Research Questions:

A. How to efficiently visualize players' shooting data:

Typically, it is difficult to directly visualize and understand players' shooting data because plotting an x-y coordinate plot abstractly in your head is very hard to formulate. Therefore, we found it beneficial to build a user-friendly program that can efficiently plot various individual player shooting data by requiring user to input player's name, team working for, and year of season to plot.

B. Can we predict players' salary?:

We believe that salary is a crucial indicator of players' performance, so we want to build up a model to see how players' data affect their salaries. We accomplish this by building a model that predicts salary based on several players' data like season average points and assists.

C. Can we predict the winning rate of each team?:

It is common for people to judge a team's success based on their winning rate; this factor influences both the season and play-off game trend. By using players data such as points scored, rebounds, assists, we can build up a model to predict the winning rate of each team.

II. Motivation:

My friends and I are fascinated by the NBA, on top of our long-lasting interest in playing the sport. We often debate over which team will take home the championship, which of us has the best bracket, and which of our top teams or players are most dependable. While many online reports and news sources display their own selective statistics, they typically prioritize publishing eye-catching news of the most talented all-star players. Thus, we transformed our passion towards basketball into constructing a program that can easily analyze and visualize player data to predict their future performances.

III. Dataset:

We used the following three datasets in our project:

A. Nba-api | An API Client package to access the APIs for NBA.com:
https://github.com/swar/nba_api

B. NBA Player Salaries | HoopsHype: https://hoopshype.com/salaries/players/

C. **bttmly/nba: Node.js client for nba.com API endpoints (github.com):**

json file for teams:

https://raw.githubusercontent.com/bttmly/nba/master/data/teams.json

Column 'teamName' refer to the teams' names.

json file for players

https://raw.githubusercontent.com/bttmly/nba/master/data/players.json

Column 'firstName', 'lastName' refer to the players' names.

IV. Method:

A. My environment:

Using Anaconda to manage the package.

1. Including NumPy, BeautifulSoup, pandas, matplotlib, requests, nba-api, SciPy, TensorFlow, and keras.
2. Using git to manage version with teammates.

B. Data Preparation:

1. For each team's past winning rate:
   a. Using endpoint LeagueDashTeamStats in nba_api to access all teams' record of winning percentage and other information.
   b. Using json and pandas modules to filter data until we get each team's id, name, game played in a season, and winning rate.
   c. Save the filtered data into csv and category them by season.

2. For each player's data:
   a. Using endpoint LeagueDashPlayerStats in nba_api to access each player's data.
   b. Using json and pandas modules to clean the raw data.
   c. Save the prepared data into csv and category them by season.

3. For each player's salary:

a. Using requests and BeautifulSoup modules to parse the HTML information on NBA Player Salaries | HoopsHype.
b. Using pandas modules to transform dictionary-like data into csv file and category them by season.

C. Methods for Question A:

1. Use requests and json modules to obtain the data of each player and each team.
2. Define a class of methods to quickly access data of player's name, id and team's name, id.
3. Using endpoint ShotChartDetail in nba_api to access target player's shot location and the field goal made percentage.
4. Generate easy-visualized map of players using matplotlib.

D. Methods for Question B:

1. Using all the data above and turn them to feature df and label df by pandas.
2. Use keras.Sequential to build up the DNN model.
3. Save the best model based on the accuracy score of evaluating df.

E. Methods for Question C:

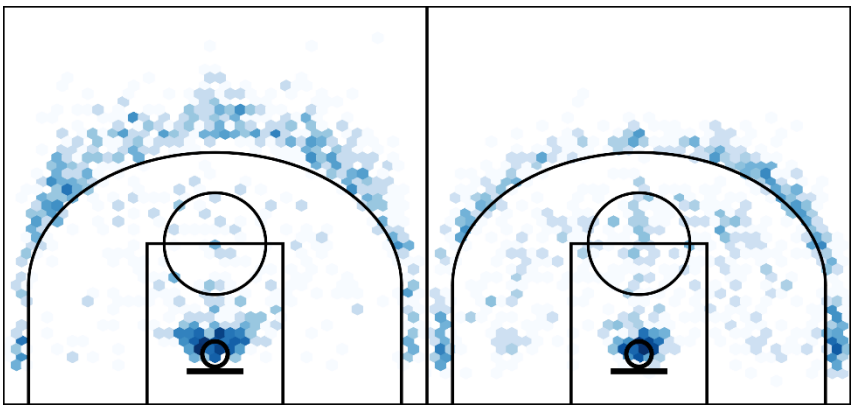
1. Using all the data above and turn them to feature df and label df by pandas and numpy.
2. Use keras.Sequential to build up the DNN model.
3. Save the best model based on the accuracy score of evaluating df.

V. Result

A. Question A:

1. By running the plotting program, we can now easily recognize the precise position and location where the target player makes a shot.
2. We shorten the process of accessing target player's shot chart by only required user to input player's name, target season, and team name.
3. The plot on the left side is the field goals made chart for Stephen Curry, Golden State Warriors, 2021-22 season; the plot on right side is the field goals made chart for Klay Thompson, Golden State Warriors, 2016-17 season. The darker color the mark is, the higher amount of field goals the

player made.



B. Question B:

1. After training by all players' data since 1996, the DNN model we built has 59 percent of accuracy on predicting the salary level of each player, and most wrong predictions are the neighbor salary level (For example, correct level is 3, and the prediction is 2). For this part of question, we divide each player into different salary level to make the prediction easier, and we only consider player's self-data to train the model; we haven't found a good way to rate the team chemistry in numeric way.
2. The result of this question does not exceed what we expect. Although players' salaries can be well affected by their numerical data, the chemistry of the entire team and the relationship between coach and players may be great influences when signing the contract. Thus, we think 59 percent of correctness will be an appropriate result that we got.
3. Feature we used for training:
   season, player age, team winning rate, total games played in season, real winning rate, minutes per game, points per game, rebounds per game, assists per game, steals per game, blocks per game, turnover per game, field goal percentage, 3-point field goal percentage, total plus minus
4. Salary level:
   level 0 – annual salary below 1500000
   level 1 – annual salary below 2500000
   level 2 – annual salary below 6000000
   level 3 – annual salary below 20000000
   level 4 – annual salary below 30000000

level 5 – annual salary above 30000000

C. Question C:

1. After training by all teams' data since 1996, the DNN model we built has 46 percent of accuracy on predicting the winning rate of each team, and most wrong predictions are the neighbor winning rate label (For example, correct label is 3, and the prediction is 2). For this part of question, we use players data mean value and standard deviation in the same team to make the prediction easier, and we only consider player's self-data to train the model; we haven't found a good way to rate the team chemistry in numeric way.
2. Before training, our team only think we can get about 20 to 30 percent of correctness on predicting the winning rate, while there are too many possible features that can not be digitizing have ability to influence a team's winning rate, such as the coach team and the completeness of teamwork. However, from the model we trained, we find out that it is possible to predict winning rate by only using players' data. Although the correctness still not exceed 50 percent, but we would say that the result from this question offers a possibility to pay more effort on numerical prediction for NBA.
3. Feature we used for training (for 1 team in each year, std -> standard deviation, mean -> mean):
   players' age std and mean, each players' total games played in season std and mean, each players' minutes per game std and mean, each players' points per game std and mean, each players' rebounds per game std and mean, each players' assists per game std and mean, each players' steals per game std and mean, each players' blocks per game std and mean, each players' turnover per game std and mean, each players' field goal percentage std and mean, each players' 3-point field goal percentage std and mean, each players' total plus minus std and mean, each players' salary std and mean
4. Meaning of winning rate label:
   0 – winning rate between 0 ~ 0.2
   1 – winning rate between 0.2 ~ 0.4

2 – winning rate between 0.4 ~ 0.6

3 – winning rate between 0.6 ~ 0.8

4 – winning rate between 0.8 ~ 1

VI. Impact and Limitation

The target audience of our project is people who are interested in using players' data to predict the game trajectory of entire season. People can use the shot chart visualizer we built to acquire a better idea of how each player makes shots on the real court; moreover, the models we trained can help people get a rough idea about the trend of a team performance and the salary level a player might get.

However, there are still lots of limitations about the results we got. First, for shot chart visualizing, we are only able to make a 2D plot that show the position of each shot, but we are not able to find data that show the real shooting heights, which is another influential factor of this dataset. Basketball is a sport that requires a good analysis of 3D space, while we are only eligible to visualize 2D plane.

Additionally, there are too many factors that can influence the players' salary and teams' winning rate, which makes it difficult to choose the correct feature for training models for prediction. We cannot prove that salary rate has direct correlation on performance. Also, there are some factors that we will never be able to include in our models, such as players' injury. Although we can still get 50 percent accuracy on salary prediction, we are not sure of how much each feature weighs in depicting the accuracy of our model and evaluating a player's ability. For winning rate prediction, the tactics of the coach and the team culture adaptability of each player may have great influences on team performance, but these data can't be valued and predicted only by evaluating each player's data.

As for the practicality of the results we obtained, I would only recommend people to use our result as a rough trend of future predictions. Nevertheless, our data should not be used as a prediction machine for any commercial activity or gambling, while our data is not accurate enough to put in real application, but it may be a great factor to consider during prediction.

VII. Challenge Goal

A. Multiple Datasets and Messy Data:

1. Using datasets from different sources to compile a more detailed result; we use the api offered by the official website of NBA and some other datasets that cover the limits of nba_api such as salary.
2. Filtering out some irrelavant data and combine different dataset's information into appropriate form for faster and easier used in future.

B. Machine Learning:

1. Training models that allow people to predict each player's salary by given information.
2. Use players data to predict each teams' winning rate.

C. New Library:

1. Using requests and BeautifulSoup module to get data from each website.
2. Using TensorFlow and keras to train models for salary and winning rate prediction.

D. Web Scraper:

1. Learning the structure of HTML to obtain the correct data from website.
2. Learning methods to obtain data from website, like BeautifulSoup module.

VIII. Work Plan Evaluation

We were fairly accurate in our proposed work plan estimates. They were close to reality because our group individually understood our levels of comfort in different topics and were able to properly adhere to the plan to come up with the final results that we aimed to receive while formulating the work plan estimate. We divided different tasks and served different purposes in making this project whole.

IX. Testing

A. Question A:

For testing, we randomly pick 50 players' data and plot their shot position manually, and the testing plots do match our results.

B. Question B:

While training, we randomly split the data into 80% for training and 20% for evaluating the accuracy. Also, we used the best model we got to evaluate the

entire dataset we own, and the accuracy of testing is exactly same as what we got while training (acc: 0.59).

C. **Question C:**

While training, we randomly split the data into 80% for training and 20% for evaluating the accuracy. Also, we used the best model we got to evaluate the entire dataset we own, and the accuracy of testing is slightly lower than what we got while training (acc_training: 0.52, acc_testing: 0.46).

X. Collaboration

nba_api:

https://github.com/swar/nba_api,

https://github.com/swar/nba_api/tree/master/docs/nba_api/stats/endpoints

NBA Player Salaries | HoopsHype

Beautiful Soup Documentation — Beautiful Soup 4.9.0 documentation (crummy.com)

tf.keras.Sequential | TensorFlow Core v2.9.1