National Basketball Association Player Statistics Forecasting

Stephen Feddes

Lewis University

Data Science Capstone Project

**Abstract**

This study will investigate designing a machine learning model to predict the statistics a National Basketball Association (NBA) player will get in an upcoming game. It will focus on using a player’s weighted average and the opposing team’s defense, derived from the dataset of player game logs, in making its predictions. The weighted average considers the player's average as a starter or bench player depending on if they are starting. The average also only considers the games the player has played with their current NBA team. Older games are weighted exponentially less. For each statistic, the opposing team’s strength is measured as the statistic per minute the team forfeits to players that play the same position and have the same starting or bench status as the player of interest.

In contrast to prevailing models, this paper offers a dynamic model given new data every day. The predictions are made with multivariate linear regressions trained with machine learning techniques. Linear relationships and model performance are visualized.

Predicting player statistics is a major concern for fantasy basketball users, who rely on player statistics to win fantasy matchups. Fantasy basketball is a popular game and only growing in popularity. By leveraging machine learning and multivariate linear regressions, this paper proposes a time-series forecasting model capable of making bold, nuanced, accurate, and interpretable predictions.

The findings of this study provide valuable, detailed, accurate, and interesting insights for fantasy basketball users to pick and play the players that will help them win. By understanding the underlying model, users can make informed decisions about which players to acquire and games to play them in.

In conclusion, this research demonstrates the effectiveness of machine learning, multivariate regression, weighted player averages, and defensive ratings in forecasting the statistics an NBA player will get in upcoming games. By utilizing a weighted average where older games are weighted exponentially less than recent games, the effectiveness of such an approach could prove valuable for other time series forecasting models. With the model dynamically integrated with APIs and deployed in the cloud, it provides immediate, practical insights for fantasy users and developers in need of NBA player statistical predictions.

**Acknowledgements**

I thank the software engineers at the NBA for providing API endpoints for fetching game log data. I thank the software engineers at Sleeper for providing an API endpoint for fetching player starting and injury information. I also thank the professors at Lewis University for instilling in me the values and knowledge to make this project possible.

**Table of Contents**

Introduction.................................................................................................................................6-9

Background and Content.................................................................................................6

Fantasy Basketball...........................................................................................................6-7

Relevance and Importance...............................................................................................7-8

Dataset.............................................................................................................................8-9

Related Work.............................................................................................................................9-10

Feature Engineering.................................................................................................................11-13

Player Averages...........................................................................................................11-12

Opposing Team Defensive Ratings.............................................................................12-13

Methodology and Results........................................................................................................14-21

Model...........................................................................................................................14-16

Results.........................................................................................................................16-18

Interpretation...............................................................................................................19-21

Deployment.............................................................................................................................21-22

Conclusions.............................................................................................................................22

References...............................................................................................................................23-27

Appendices..............................................................................................................................28-32

**List of Figures**

Fig 1. Scatter plot of average field goals attempted and actual field goals attempted..................12

Fig. 2 Scatter plot of field goal attempt defensive rating and actual field goals attempted..........14

Fig. 3 Time series of predicted game scores and actual game scores for Precious Achiuwa.......21

**List of Tables**

Table 1. R2 and RMSE for the training and test sets....................................................................17

Table 2. Linear regressions for the predicted statistics.................................................................17

Table 3. Correlations between each variable in each regression..................................................18

**Introduction**

**Background and Content**

Predicting the statistics a player will achieve in an upcoming game is crucial for fantasy sports enthusiasts. Fantasy sports involve assembling a team of sport players to compete in a virtual league. Participants aim to construct the most statistically formidable fantasy teams to compete against others. *Fantasy basketball* is very popular and only getting even more popular, with over 2 million active users on ESPN’s fantasy basketball app two years in a row [1].

In the *National Basketball Association* *(NBA)*, players play positions such as center (C), power forward (PF), small forward (SF), shooting guard (SG), and point guard (PG). Each position corresponds to a distinct role that a basketball player fulfills within their team. For instance, point guards are primarily responsible for scoring and assisting scoring opportunities for teammates, while centers focus on blocking opponents' scoring attempts and securing rebounds. In basketball, a team can only have 5 players on the court at a time. Players who are on the court when the game starts are called *starters*. Players that start the game on the bench and come into the game to relieve the starters are called *bench players*. The proposed model is designed in Jupyter Notebooks [2] and is available at this GitHub repository: [https://github.com/DraftBash/draftbash-projection-model.](http://projections-model/)

The application the model is deployed to is available at this GitHub repository: <https://github.com/DraftBash/draftbash-nba-players-api>

**Fantasy Basketball**

Fantasy basketball typically employs two primary scoring systems: fantasy points and categories [3]. In fantasy point leagues, managers aim to assemble rosters with the highest total fantasy points. These fantasy points are determined by a weighted sum of statistics accumulated by the user's players, including field goals made, field goals missed, points scored, rebounds, assists, steals, blocks, and turnovers. In category leagues, users aim to have the majority statistics in different statistical categories, such as points scored, rebounds, assists, steals, blocks, free-throw percentage, field-goal percentage, turnovers, and three-pointers made. Victory in a category is achieved by surpassing other users in the quantity of that particular statistic. Users accumulate categories by starting players that accumulate them.

**Relevance and Importance**

National Basketball Association player statistics forecasting aids fantasy basketball users in selecting the best players for their team and starting them accordingly. Throughout a fantasy basketball season, users can add NBA players to their fantasy team if they are not already rostered by another user. However, a user’s team is constrained by a limited number of available roster spots, meaning a user sometimes must cut a player from their team [4].

In fantasy basketball, users are only able to utilize a subset of players for their starting lineup. Only the players in that starting lineup accrue points for a given day or week [5]. Choosing the right player to start on your fantasy team is critical for success.

Moreover, in certain fantasy leagues, users may be restricted to utilizing a player for just one game per week [6]. This prompts users to deploy players against defenses perceived as weak. Thus, the proposed model factors in the defensive ratings of opposing teams.

Furthermore, players can get traded in real life to different teams. Because players may perform differently on different teams, the proposed model only considers the average the player has when playing for their current team [7].

As an experienced fantasy user who has won in a few leagues, I know that one of the key strategies in fantasy sports is to acquire undervalued players who subsequently earn starting roles on a real team or have just earned them. Unlike other models, the proposed model adjusts for instances where a player has recently been promoted to starting status on an NBA team or demoted to the bench.

While most models focus solely on predicting player performance within a single metric, the proposed model forecasts all statistics relevant to fantasy basketball. These statistics include points, rebounds, assists, steals, blocks, turnovers, field goal attempts, field goals made, free throw attempts, and free throw misses. Predicting individual statistics is particularly crucial for fantasy category leagues, where the objective is to accumulate the highest number of statistics across various categories.

**Dataset**

The dataset used in this study comprises data from the 2022-23 and 2023-24 NBA seasons. Although the model is trained on games from the 2023-24 NBA season, the values utilized to compute averages leading up to each game in that season included data from the 2022-23 season. The dataset was sourced from the NBA API. It consists of 63,017 player game logs in total. To focus solely on predicting statistics for active players (inactive, injured players would presumably have zero statistics), only game logs where players were eligible to participate were considered. 31,873 player game logs are from the 2023-24 NBA season.

In 161 instances across both seasons, player game logs were missing player position information. Given the all-purpose role of small forwards in basketball, all missing positions were substituted with the small forward position. The dataset comprises 24 attributes, including the date of the game, player ID, season, player’s team ID, opposing team’s ID, position, minutes played, assists, rebounds, and points. Further details regarding these attributes can be found in the data dictionary (See Appendix A).

**Related Work**

To predict player statistics, much of the relevant literature focuses on a player’s recent individual performances, recent team performances, and the opposing team’s recent defensive performances. David Menn’s model considers both the player and their team [8], Chan-Hu Shivakumar's model focuses on the player’s recent performance and the opposing team’s recent performance [9], and Benjamin Miller’s model underscores the importance of projecting the minutes a player is expected to play [10]. Menn’s and Miller’s models utilize the same performance metric called *game score,* which was created by basketball analyst John Hollinger to provide a measure of a player’s productivity in a game [11]. Menn utilized an XGBoost machine learning model, which works by creating new models that predict the residuals of the prior models and then compounds them to make a prediction [8].

(1)

Menn’s model employs k-means clustering to classify player archetypes, achieving success by considering how a player performs with different player archetypes. Menn’s model achieved a *root mean square error (RMSE)* of 3.127 game scores, while Miller’s model had an RMSE of 6.7015 game scores. RMSE is a goodness-of-fit metric that uses the squared root of the mean for the squared difference between each predicted value y and outcome value ŷ. In other words, it measures the standard deviation of the residuals [12]. A residual is the difference between the predicted and actual value. The smaller the RMSE is, the better the model fits.

(2) Comparing the effectiveness of Shivakumar’s model is more challenging because they used different weights for their performance calculation, along with a different performance metric, ranked difference error. Rank difference error is the absolute difference between the actual rank and the predicted rank of a player in a game, playing for a specific team [9].

The authors of Menn’s model acknowledge, “We noticed that our model is always scared to predict more extreme games, resulting in less powerful prediction” [8]. Although Menn’s model predicts consistent players well, its issues in predicting extreme performances are a significant drawback.

While other models contribute to understanding the relative accuracies of player performance predictive models, Kostya Medvedovsky’s Daily Adjusted and Regressed Kalman Optimized Projections (DARKO) model [13] directly influenced the proposed model in this paper. His technique of using a weighted average, where older games are weighted exponentially less than recent games, influenced the *feature-engineering* of the averages used in the proposed model.

Unlike the aforementioned models, the proposed model considers whether the player has become a new starter or has recently been benched. Furthermore, instead of focusing on just one metric for player performance, my model’s ability to predict individual statistics that a player will achieve, such as assists or rebounds, offers more valuable, interpretable, detailed insights than most others.

**Feature Engineering**

For feature engineering, I considered two main types of features: player averages and opposing team’s recent performance. Feature engineering is the extraction and transformation of raw variables into features suitable for machine learning [14]. In this project, Python’s Pandas [15] and NumPy [16] libraries were used to manipulate raw player game log data into features suitable for machine learning. The Matplotlib [17] and Seaborn Python libraries [18] are used for the data visualizations.

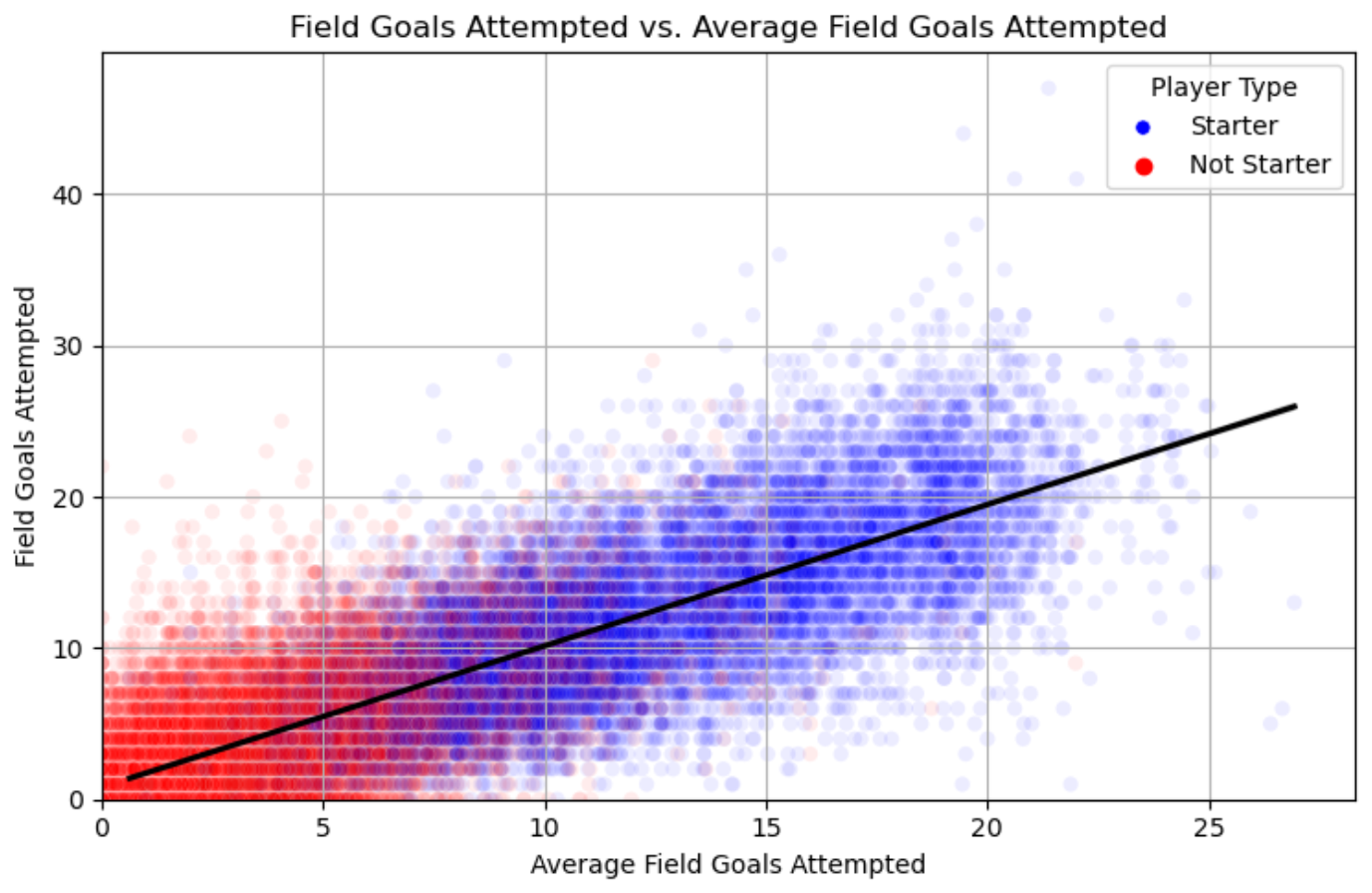
**Player Averages**

For each statistic of interest (See Appendix B.1), I engineered a sophisticated weighted average of previous performances for each player by weighing older games exponentially less than more recent games (See Appendix B.2 and Appendix B.3). The base of the exponent, known as the decay rate, is the constant 0.98. The exponent corresponds to the number of days (t) since the game was played. In Medvedovsky’s work, he wrote, “In practice, with few exceptions, values of β [decay rate] tend to be 0.98 or higher” [13]. If the exponent is set to 1, all previous games for a player are weighted equally. Conversely, a low exponent implies that only the most recent games are effectively considered.

If a player transitions from being a bench player to a starter, the player's average is estimated by calculating the weighted average of the player’s games as a starter. If the player has no games played as a starter, the average of starters who play the same position as the player is utilized (See Appendix B.2). Similarly, if a player transitions from being a starter to a bench player, the weighted average of their games as a bench player is employed and is similarly approximated if the player has no previous games played as a bench player (See Appendix B.3).

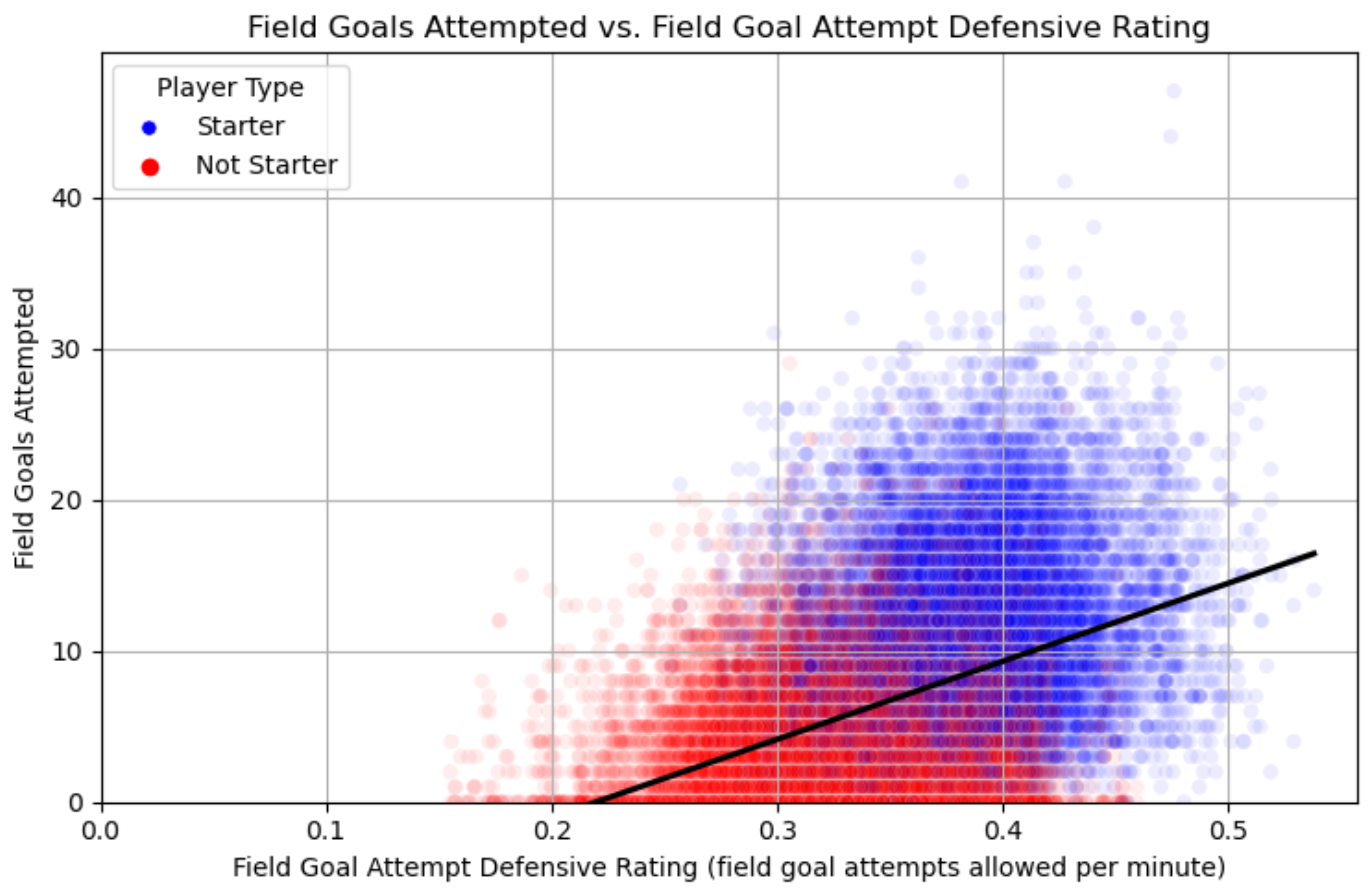
Furthermore, only the games the player has played with their current team are considered. This is important because a player's performance can change quickly if they do not fit well with their new real-life team.

Fig. 1 and Fig. 2 illustrate some of the engineered features. In Fig. 1, a strong, positive linear relationship is evident between a player’s average field goal attempts per game and the actual number of field goals they attempt in a game. Notice that starters average and attempt significantly more field goals than non-starters (bench players).

 Fig 1. Scatter plot of average field goals attempted and actual field goals attempted

**Opposing Team Defensive Ratings**

For each statistic of interest (See Appendix C.1), I quantified the opponent’s tendency to surrender that statistic to players who have the same position and starting status as the player being considered. These measures are evaluated on a per-minute basis (See Appendix C.2). For instance, if there is a starting player at the center position, a function would calculate the points per minute that starting centers scored against the opponent. These measures consider all the games the opposing team played in the last 50 days. If the sample size is too small, as often occurs when a new NBA season starts, the measure is replaced with the team’s average in the previous season. In Fig. 2, there is a slight, positive linear relationship between a player’s opponent's field goal attempts defensive rating and the actual number of field goals the player attempts in the game. Notice that starters attempt more field goal attempts, and opposing defenses allow more field goal attempts per minute to starters. Also, notice that predicting field goal attempts with just the opposing defense has significantly more variance than using the player’s weighted average.

Fig. 2 Scatter plot of field goal attempt defensive rating and actual field goals attempted

**Methodology and Results**

**Model**

For each target statistic, except steals, a *multivariate linear regression* model was trained and tested using Scikit-Learn's [19] *supervised machine learning* modelfor linear regression. Supervised machine learning uses labeled datasets to train algorithms to predict outcomes and recognize patterns [19]. Multiple regression is a type of regression where the dependent variable shows a linear relationship with two or more independent variables [20]. It assumes that the independent variables are not highly correlated with each other [20].

For each multivariate regression, only two independent variables will be considered for predicting a dependent variable. Each equation will look like the equation below, where x1 and x2 are the independent variables, ŷ is the dependent variable, a is a constant, and bi is the coefficient of the ith independent variable.

(3)

The statistics to be predicted are field goals attempted, field goals made, free throws attempted, free throws made, points scored, rebounds grabbed, assists, steals, blocks, turnovers, and threes made. Each equation is presented in Table 3. The linear regression models' *R2* and RMSE for the training and testing sets are provided in Table 1. R2 measures the goodness of fit for a model. It has values between 0 and 1, indicating the proportion of variance in the dependent variable that the model explains [21]. R2 is equal to 1 – sum of squared residuals (SSR) / total sum of squares (TSS). A squared residual is the squared difference between a predicted value ŷ and actual value y. The total sum of squares is the sum of the squared differences between the mean of the independent variable ȳ and each predicted value, which is by the definition the variance of the independent variable. If the variance with the predictions is no different than the variance of the independent variable, then R2 becomes 1 – (σ2/ σ2) = 0, meaning that the model explains 0% of the variance in the dependent variable. If the variance of the squared residuals is 0, then our model fits the data in a perfect line and R2 becomes 1 – (0/ σ2) = 1, meaning 100% of the variance is explained by the model. So, R2 is the proportion of variance in the dependent variable explained by the model. A higher R2 indicates a better fit. A very high R2, however, may indicate overfitting [21], which means the model predicts outcomes from the training set well, but performs poorly on new data.

(3)

For predicting steals, only the player’s average is used as a predictor because the *correlation coefficient* *(R)* between the steals a player gets in a game and the opposing team’s tendency to have players steal the ball from them was very low, below 0.05. The correlation coefficient is a measure of how linearly related two variables are. It ranges between –1 and 1. -1 indicates a perfect negative linear relationship, 0 indicates no linear relationship, and 1 indicates a perfect linear relationship [22]. The preprocessed dataset for machine learning, with features derived from the 2023-24 NBA season player game logs, was split 80% for training and 20% for testing. The training estimates the coefficients of the regressions. The test partition is for evaluating the performance of the model at predicting values from unseen data. Each target statistic, besides steals, was considered with two predictors: the player's average and the opposing team’s rating (statistics given up per minute) against that statistic. Although this paper is not concerned with fouls, offensive rebounds, or defensive rebounds because they are usually not used in fantasy basketball, those statistics were also predicted with linear regression models so that the game score measure for each game in the testing and training sets could be calculated. The reason for calculating the game score measure is to compare my model’s performance to Menn’s and Miller’s models.

**Results**

On the training set, the model had a root mean squared error (RMSE) value of 5.52 game scores. On the testing set, the model had an RMSE of 5.58 game scores. These measures are higher than Menn’s game score RMSE of 3.127 but lower than Miller’s game score RMSE of 6.7015.

The R2 and RMSE for each statistic of interest are in Table 1. Both of those measures were used on the training and test sets. Because the RMSE and R2 values for the training and testing sets are very close, it indicates that the model is not overfitting to the training data.

Although steals have a comparatively low R2 compared to the other predicted statistics, the rarity of steals makes that level of explained variation valuable because steals are typically more valuable than other statistics due to their relative scarcity and unpredictability [23].  
 Using a linear regression for each target statistic enables the model to make efficient predictions because each target statistic can be calculated with a simple linear equation. Table 2 shows the linear equations the model uses to make its predictions. Table 3 shows the correlation coefficients between each variable in each regression.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Statistic | R2 (Training) | RMSE (Training) | R2 (Test) | RMSE (Test) |
| fieldGoalsAttempted | 0.719 | 3.412 | 0.723 | 3.374 |
| fieldGoalsMade | 0.600 | 2.136 | 0.596 | 2.155 |
| threesMade | 0.381 | 1.122 | 0.389 | 1.119 |
| points | 0.629 | 5.521 | 0.627 | 5.613 |
| steals | 0.154 | 0.818 | 0.167 | 0.814 |
| blocks | 0.233 | 0.698 | 0.262 | 0.687 |
| assists | 0.575 | 1.695 | 0.587 | 1.656 |
| reboundsTotal | 0.557 | 2.331 | 0.562 | 2.333 |
| turnovers | 0.354 | 1.057 | 0.370 | 1.078 |
| freeThrowsAttempted | 0.423 | 1.975 | 0.421 | 2.021 |
| freeThrowsMade | 0.404 | 1.666 | 0.407 | 1.715 |

Table 1. R2 and RMSE for the training and test sets.

|  |
| --- |
| **Linear regressions for the predicted statistics** |
| **fieldGoalsAttempted = -0.604 + 0.949 \* fieldGoalsAttemptedAvg + 2.956 \* fieldGoalsAttemptedOppDRating** |
| **fieldGoalsMade = -0.367 + 0.925 \* fieldGoalsMadeAvg + 3.902 \* fieldGoalsMadeOppDRating** |
| **threesMade = -0.02 + 0.856 \* threesMadeAvg + 3.39 \* threesMadeOppDRating** |
| **freeThrowsAttempted = -0.08 + 0.874 \* freeThrowsAttemptedAvg + 3.212 \* freeThrowsAttemptedOppDRating** |
| **freeThrowsMade = -0.046 + 0.864 \* freeThrowsMadeAvg + 3.219 \* freeThrowsMadeOppDRating** |
| **points = -0.778 + 0.928 \* pointsAvg + 3.307 \* pointsOppDRating** |
| **assists = -0.004 + 0.931 \* assistsAvg + 1.789 \* assistsOppDRating** |
| **reboundsTotal = 0.059 + 0.931 \* reboundsTotalAvg + 1.392 \* reboundsTotalOppDRating** |
| **turnovers = -0.018 + 0.846 \* turnoversAvg + 3.354 \* turnoversOppDRating** |
| **steals = 0.171 + 0.714 \* stealsAvg** |
| **blocks = 0.027 + 0.747 \* blocksAvg + 3.439 \* blocksOppDRating** |

Table 2. Linear regressions for the predicted statistics.

|  |  |
| --- | --- |
| Dependent and Independent Variables | Correlation Coefficient (R) |
| fieldGoalsAttempted, fieldGoalsAttemptedAvg | 0.848 |
| fieldGoalsAttempted, fieldGoalsAttemptedOppDRating | 0.418 |
| fieldGoalsAttemptedAvg, fieldGoalsAttemptedOppDRating | 0.472 |
| fieldGoalsMade, fieldGoalsMadeAvg | 0.773 |
| fieldGoalsMade, fieldGoalsMadeOppDRating | 0.370 |
| fieldGoalsMadeAvg, fieldGoalsMadeOppDRating | 0.443 |
| threesMade, threesMadeAvg | 0.617 |
| threesMade, threesMadeOppDRating | 0.265 |
| threesMadeAvg, threesMadeOppDRating | 0.360 |
| freeThrowsAttempted, freeThrowsAttemptedAvg | 0.649 |
| freeThrowsAttempted, freeThrowsAttemptedOppDRating | 0.211 |
| freeThrowsAttemptedAvg, freeThrowsAttemptedOppDRating | 0.277 |
| freeThrowsMade, freeThrowsAttemptedAvg | 0.635 |
| freeThrowsMade, freeThrowsAttemptedOppDRating | 0.219 |
| freeThrowsMadeAvg, freeThrowsAttemptedOppDRating | 0.298 |
| points, pointsAvg | 0.793 |
| points, pointsOppDRating | 0.394 |
| pointsAvg, pointsOppDRating | 0.470 |
| assists, assistsAvg | 0.759 |
| assists, assistsOppDRating | 0.355 |
| assistsAvg, assistsOppDRating | 0.441 |
| reboundsTotal, reboundsTotalAvg | 0.747 |
| reboundsTotal, reboundsTotalOppDRating | 0.268 |
| reboundsTotalAvg, reboundsTotalOppDRating | 0.328 |
| turnovers, turnoversAvg | 0.597 |
| turnovers, turnoversOppDRating | 0.168 |
| turnoversAvg, turnoversOppDRating | 0.234 |
| steals, stealsAvg | 0.394 |
| steals, stealsOppDRating | 0.042 |
| stealsAvg, stealsOppDRating | 0.048 |
| blocks, blocksAvg | 0.484 |
| blocks, blocksOppDRating | 0.211 |
| blocksAvg, blocksOppDRating | 0.326 |

Table 3. Correlations between each variable in each regression.

**Interpretation**

Another benefit of using linear regressions is their interpretability. Users can consider the player’s average and the opposing team’s strength when deciding to use that player. Although these multivariate regressions are valuable, there is some collinearity between the predictors in each regression. Collinearity means that two explanatory variables are correlated with each other [24]. In each regression, the correlation coefficient between the predictors is no higher than 0.472, which is not high, but could make the interpretability of the model’s independent variables difficult. The collinearity is likely because a player’s average can be influenced not just by their skills but also by the defenses the player has played against, and because both the defensive rating and player average are adjusted for whether the player is a starter or not. Although there is some concern for the interpretability of the predictors' effects in each regression, this approach at least offers more insights than a black-box approach with models like a neural network.

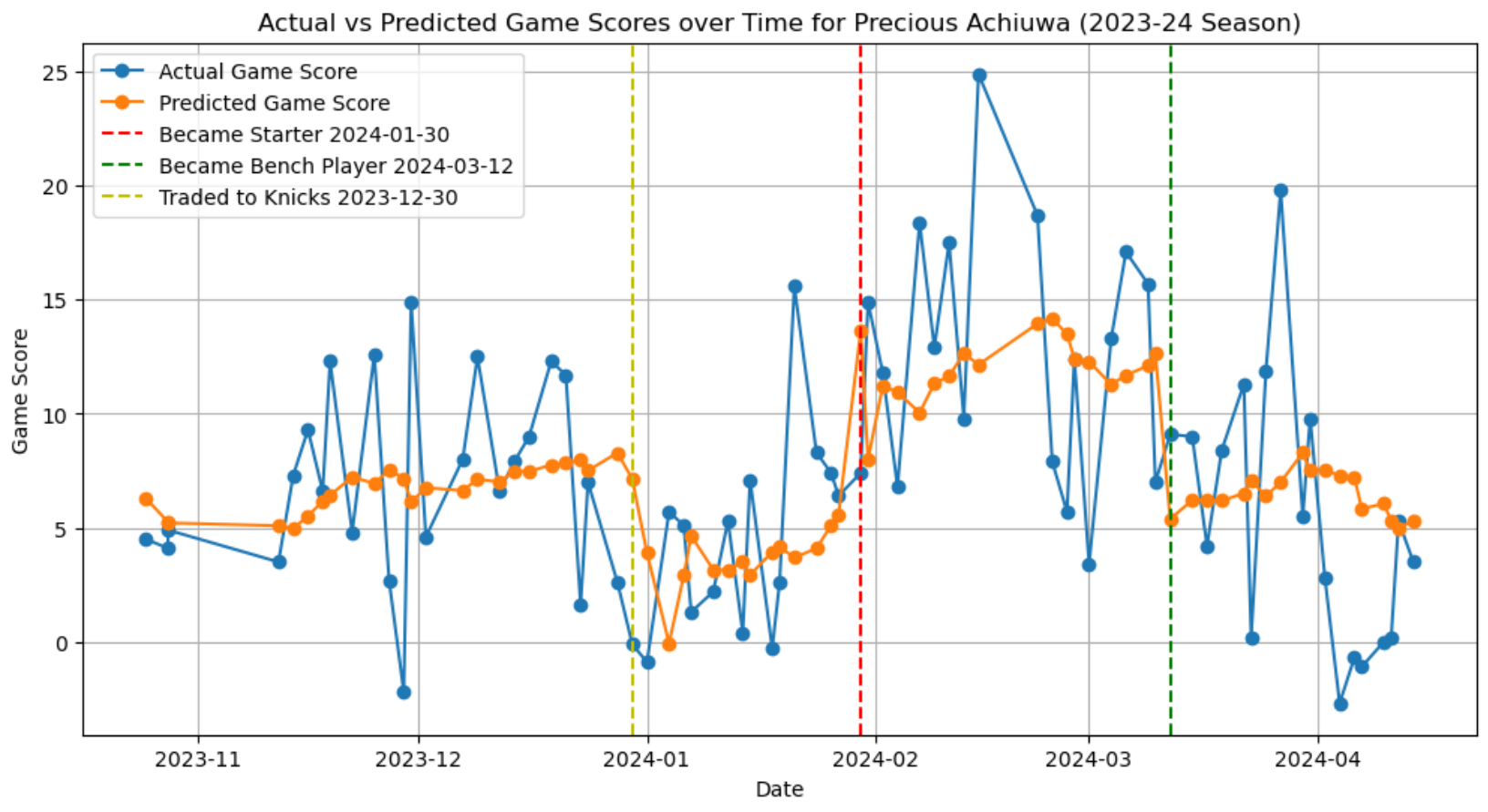
A player’s average provides a strong baseline estimate for a predicted statistic, and the opposing defense's strength can indicate if the player is expected to perform above or below that average. If a player faces a weak defense against assists, they should be expected to get more assists than their average. The greater the opposing defense's tendency to surrender assists to players with the same position as the player and same starting or bench status as the player, the higher the predicted assists should be. That increase should be linearly related to the defense’s rating, where a higher rating corresponds to a higher predicted value. Because fantasy users sometimes can only choose one game in a week for their player to participate in, they cannot rely solely on the player’s average to help them choose the game because their predicted statistic for each game would be the same if the opposing team for each game is not considered.

For example, if a starting point guard had a weighted average of 5 assists and was playing against a team that allows 0.2 assists per minute to starting point guards, then they would be expected to get

If the player was playing against a team that allows 0.15 assists per minute to starting point guards, the player would be expected to get

Therefore, if a fantasy user needs assists for their team and can only play each player in one game per week, they should pick the games with the defenses that allow the most assists per minute to players with the same roles as the user’s players.

A good model should make bold projections when a major change happens to a player, such as becoming a starter for an NBA team. To illustrate the model’s ability to make bold predictions, consider the time series in Fig. 3. Fig. 3 shows the predicted and actual game scores for Precious Achiuwa from the entire 2023-24 NBA season. Precious Achiuwa was a bench power forward for the Toronto Raptors, but got traded to December 30th, 2023, to the New York Knicks [25]. He remained a bench player on Knicks, but his game scores went considerably down. Then, when the starting power forward for the Knicks, Julius Randle, got injured on January 27, 2024, [26] Precious Achiuwa became the starting power forward. Recognizing that Precious Achiuwa became a starter, the model boldly increases the predicted game score for Precious Achiuwa right away. When another power forward got healthy, Precious Achiuwa became a bench player again on March 12, 2024. Recognizing that he became a bench player again, the model boldly decreases its predicted game scores for Achiuwa.

Fig. 3 Time series of predicted game scores and actual game scores for Precious Achiuwa

**Deployment**

After the linear regression for each statistic of interest was calculated through machine learning, the linear equations were added to the forecaster application. The application is built with Python’s FastAPI library [29], a web development API framework. Through Github Actions [30], that application was deployed to Azure [31], which hosts the application in the cloud. Using Azure Webjobs [32], scripts were scheduled to fetch daily game log and player data from the NBA and Sleeper APIs [33] and store it in MongoDB [34], a NoSQL database. Each day, the game logs from the database would be extracted into a Pandas dataframe. That dataframe is then transformed to have the features needed to feed the linear regressions. The predicted statistics from the regressions are then stored in the database. MongoDB stores data in JSON format and can store large amounts of data with a flexible schema. To see an example of the stored projections in the database, view Appendix D.1. Notice that the projections of Nikola Jokic for the example game are quite accurate when compared to the actual results of the game for Nikola Jokic in Appendix D.2.

**Conclusions**

In this work, I introduced a dynamic model integrated with APIs to forecast fantasy-relevant statistics for NBA players in the current week.

I engineered features for machine learning by calculating the player’s weighted averages across different statistics as well as the opposing defense’s strength across different statistics. The averages were adjusted if a player became a bench player or a starter, or if they changed teams. With the success of using weighted averages where older games are weighed exponentially less than recent games, other time series forecasting models may benefit from such an approach.

To make the predictions, multivariate linear regressions, which consider the player's average and opposing defense, were calculated through machine learning. Because the statistics can be predicted with simple linear equations, the model can predict each statistic efficiently. The linear regressions are interpretable, though the presence of some collinearity may confound the predictors.

Although fantasy basketball involves a lot of chance, this model, when combined with human insight, can help fantasy managers make better-informed decisions when crafting their fantasy basketball teams.

**References**

[1] Kevin Ota. *ESPN Men’s Fantasy Basketball: 2 Million Players, 2 Years in a Row*. ESPN Fantasy. Accessed April 28, 2024. Available: <https://espnpressroom.com/us/press-releases/2024/01/espn-mens-fantasy-basketball-2-million-players-2-years-in-a-row>

[2] Mike Driscoll. *Jupyter Notebook: An Introduction*. RealPython. Accessed April 28, 2024.

Available: <https://realpython.com/jupyter-notebook-introduction>

[3] *How to Play Fantasy Basketball*. Rotowire. Accessed April 28, 2024.

Available: <https://www.rotowire.com/basketball/advice>

[4] Yahoo Fantasy Basketball: A 101 Guide on How to Play. Yahoo Sports. Accessed May 2, 2024.

Available: <https://sports.yahoo.com/yahoo-fantasy-basketball-a-101-guide-on-how-to-play-225349765.html>

[5] *Can bench players score points in fantasy basketball?* Rotowire. Accessed May 2, 2024.

Available: <https://www.rotowire.com/faq/can-bench-players-score-points-in-fantasy-basketball-b3475f5b>

[6] *Game Pick Details*. Sleeper. Accessed April 28, 2024. Available: <https://support.sleeper.com/en/articles/4701537-game-pick-details>

[7] Dashiell Nusbaum. *Nylon Calculus: How long does it take traded players to adjust to their new team?* Accessed on May 4th, 2024. Available: <https://fansided.com/2020/01/14/nylon-calculus-traded-players-adjust-new-team>

[8] David Menn. NBA Player Performance Prediction Based on XGBoost and Synergies. University of Washington. Accessed April 28, 2024. Available: <https://courses.cs.washington.edu/courses/cse547/23wi/old_projects/23wi/NBA_Performance.pdf>

[9] Chan-Hu Shivakumar. *Learning to Turn Fantasy Basketball Into Real Money Introduction to Machine Learning*. University of Pennsylvania. Accessed April 28, 2024.

Available: <https://shreyasskandan.github.io/Old_Website/files/report-ChanHuShivakumar.pdf>.

[10] Benjamin Miller. *Using Automated Machine Learning to Predict NBA Player Performance*. Data Robot. Accessed April 28, 2024. Available: <https://www.datarobot.com/blog/using-datarobot-to-predict-nba-player-performance>

[11] *Basketball Game Score Calculator*. iCalculator. Accessed April 28, 2024.

Available: <https://sports.icalculator.com/basketball-game-score-calculator.html>

[12] Jim Frost. Root Mean Square Error (RMSE). Statistics by Jim. Accessed May 2, 2024.

Available: <https://statisticsbyjim.com/regression/root-mean-square-error-rmse>

[13] Kostya Medvedovsky. *Daily Adjusted and Regressed Kalman Optimized Projection*. Accessed April 28, 2024. Available: <https://sports.icalculator.com/basketball-game-score-calculator.html>

[14] *What is Feature Engineering?* AWS. Accessed May 2, 2024. Available: <https://aws.amazon.com/what-is/feature-engineering>

[15] Vidhi Chugh. *Python Pandas Tutorial: The Ultimate Guide for Beginners*. Accessed April 28, 2024. Available: <https://www.datacamp.com/tutorial/pandas>.

[16] Karlijn Willems. *Python Numpy Array Tutorial*. Datacamp. Accessed on April 28, 2024.

Available: <https://www.datacamp.com/tutorial/pandas>.

[17] Kevin Babitz. *Introduction to Plotting with Matplotlib in Python*. Datacamp. Accessed on April 28, 2024. Available: <https://www.datacamp.com/tutorial/matplotlib-tutorial-python>

[18] Moez All. *Python Seaborn Tutorial for Beginners: Start Visualizing Data*. Datacamp. Accessed on April 28, 2024. Available: <https://www.datacamp.com/tutorial/seaborn-python-tutorial>

[19] *Linear Regression using sklearn*. Geeks for Geeks. Accessed April 28, 2024.

Available: <https://www.geeksforgeeks.org/python-linear-regression-using-sklearn>

[20] *What is Supervised Learning?* Google Cloud. Accessed May 2, 2024.

Available: <https://cloud.google.com/discover/what-is-supervised-learning>

[21] Callum Ballard. An Ode to R-Squared. Towards Data Science. Accessed May 2, 2024.

Available: <https://towardsdatascience.com/an-ode-to-r-squared-804d8d0ed22c>

# [22] *Correlation Coefficient*. JMP. Accessed May 2, 2024.

Available: <https://www.jmp.com/en_us/statistics-knowledge-portal/what-is-correlation/correlation-coefficient.html>

[23] *Benjamin Morris. Steals Are Predictive, But Are They That Important?* FiveThirtyEight. Accessed May 2, 2024. Available: <https://fivethirtyeight.com/features/steals-are-predictive-but-are-they-that-important>

# [24] Nathan Rosidi. *A Beginners Guide to Collinearity: What it is and How it affects our regression model*. Stratascratch. Accessed on April 28, 2024. Available: <https://www.stratascratch.com/blog/a-beginner-s-guide-to-collinearity-what-it-is-and-how-it-affects-our-regression-model>

# [25] Jorge Sierra. *Precious Achiuwa headed to New York as part of Knicks trade.* USA Today. Accessed on May 5th, 2024. Available: <https://www.usatoday.com/story/sports/nba/2023/12/30/precious-achiuwa-headed-to-new-york-as-part-of-knicks-trade/72066897007>

# [26] Mark Sanchez. *Knicks’ feel-good win over Heat dampened by Julius Randle’s dislocated shoulder.* New York Post. Accessed on April 28, 2024. Available: https://nypost.com/2024/01/27/sports/knicks-win-over-heat-dampened-by-julius-randle-injury

[27] Moez All. FastAPI Tutorial: An Introduction to Using FastAPI. Datacamp. Accessed April 28, 2024. Available: <https://www.datacamp.com/tutorial/introduction-fastapi-tutorial>

[28] Nathan Cheng. *Deploy FastAPI on Azure with Github Actions*. Towards Data Science. Accessed April 28, 2024. Available: <https://towardsdatascience.com/deploy-fastapi-on-azure-with-github-actions-32c5ab248ce3>

[29] Stephen Bigelow. *What is Microsoft Azure and How Does it Work?* TechTarget. Accessed April 28, 2024. Available: <https://www.techtarget.com/searchcloudcomputing/definition/Windows-Azure>

[30] *Marco Venturi. How to Automate Tasks with Azure Webjobs*. FreeCodeCamp. Accessed April 28, 2024. Available: <https://www.freecodecamp.org/news/how-to-automate-tasks-with-azure-webjobs>

[31] Sleeper API. Accessed April 28, 2024. Available: <https://docs.sleeper.com/#fetch-all-players>

# [32] Bex Tuychlev. MongoDB Tutorial: How to Set Up and Query MongoDB Databases. Accessed April 28, 2024. Available: <https://www.datacamp.com/tutorial/mongodb-tutorial-how-to-set-up-and-query-mongodb-databases>

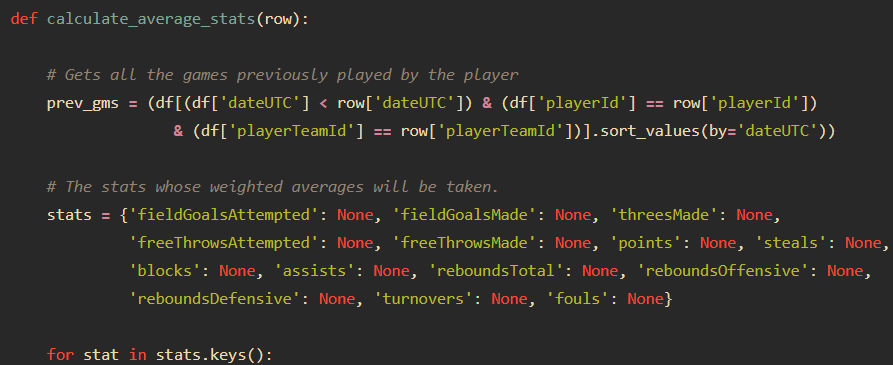
**Appendices**

**Appendix A. Data Dictionary**

|  |  |  |
| --- | --- | --- |
| Attribute | Data Type | Description |
| season | Integer | NBA season number, i.e. 2023 would be the 2023-24 season |
| dateUTC | Date | Date of a NBA game (UTC time) |
| playerTeamId | Integer | Id of the player’s team in a game |
| opposingTeamId | Integer | Id of the opposing team in a game |
| playerId | Integer | Id of a player (from the NBA API) |
| position | String | Position a player plays. C - center, PF - power forward,  SF - small forward, SG-shooting guard, PG - point guard |
| isStarter | Boolean | Indicates if a player started the game, True or False |
| minutes | Float | Total minutes a player played in a game |
| points | Integer | Total points a player scored in a game |
| assists | Integer | Total field goals a player assisted in a game |
| reboundsDefensive | Integer | Total rebounds a player got from opponent field goal misses |
| reboundsOffensive | Integer | Total rebounds a player got from their team’s field goal misses |
| reboundsTotal | Integer | Total offensive and defensive rebounds a player got in a game. |
| steals | Integer | Total steals a player got in a game |
| blocks | Integer | Total blocks a player got in a game |
| turnovers | Integer | Total turnovers a player got in a game |
| fieldGoalsAttempted | Integer | Total field goals a player attempted in a game |
| fieldGoalsMade | Integer | Total field goals a player made in a game |
| freeThrowsAttempted | Integer | Total free throws a player attempted in a game |
| freeThrowsMade | Integer | Total free throws a player made in a game |
| fouls | Integer | Total fouls a player received in a game |

**Appendix B. Player Averages Calculator**

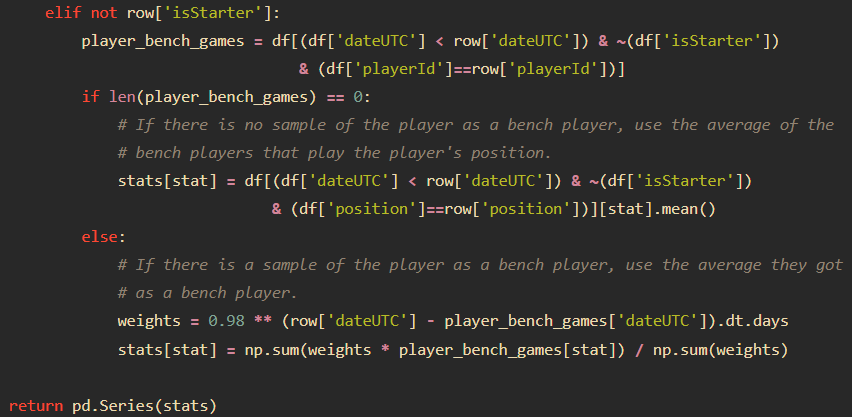
**B.1 Statistics whose averages will be calculated for each player**



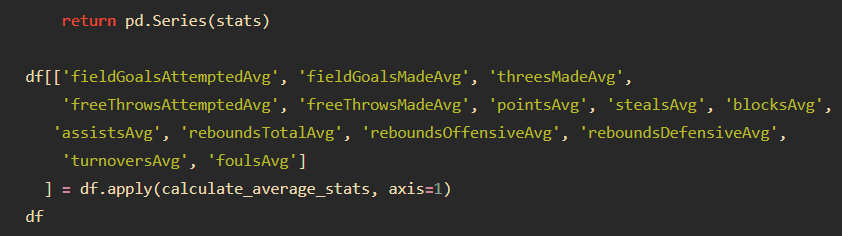
**B.2 Weighted average of games where a player is a starter**



**B.3. weighted average of games where a player is a bench player**

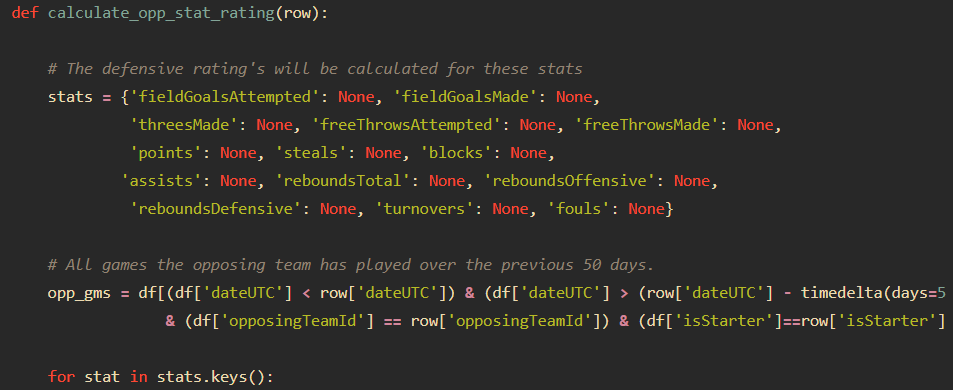


**B.4 Add the averages to the dataframe**

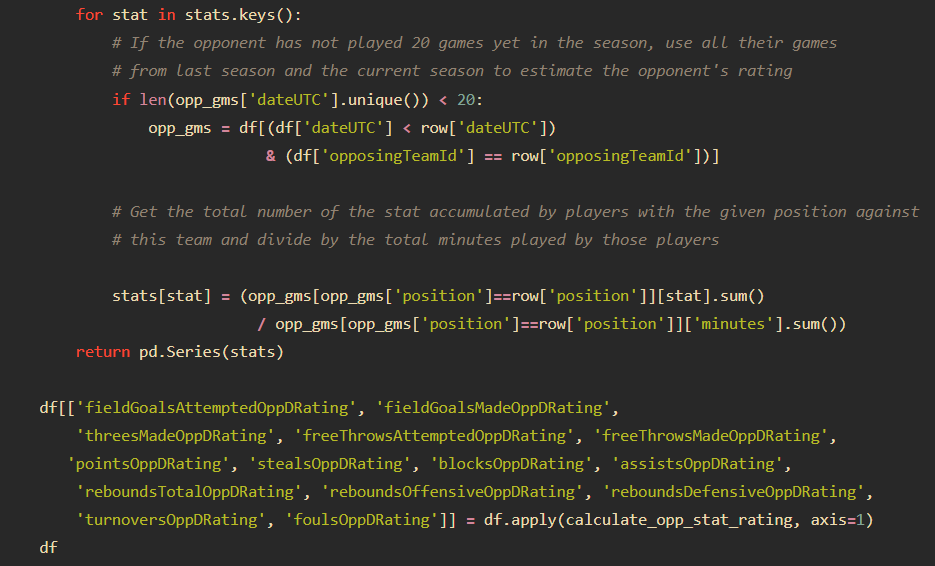


**Appendix C. Opposing Defensive Ratings Calculator**

**C.1 The stats for which the opponent’s defensive ratings will be calculated for.**



**C.2 Adjusting for the player’s role and the number of games the opponent has played, calculate the per-minute statistics the opposing team gives up to players that play the player’s role.**



**Appendix D. Projection Storage**

**D.1 Projected statistics for Nikola Jokic**



**D.2 Actual statistics by Nikola Jokic in the game**

