HIGH SCHOOL EDITION
Journal of Student Research

Volume 12 Issue 3 (2023)

Identifying the most important factors on player performance in the National Basketball Association using Machine Learning

Raghav Singh[1] and Michael Stanley[#]

[1]Inventure Academy
[#]Advisor

ABSTRACT

The field of data analytics in basketball has been on a meteoric rise in recent years. Improved optical tracking technology has allowed data collection on a scale that has never been seen before. As a result, a plethora of work related to how player performances get affected by various in-game factors have been published. The rush of incoming data has also allowed teams to get a better handle on how they can mitigate certain factors to increase a player's performance. However, no one has been able to capture the relationship that extrinsic factors, such as weather, precipitation, or attendance could affect player performance, and rank which factors have the most impact. In this paper, I will describe the process of creating a novel dataset that consists of game attendance, opponent information, weather information, date information, and player form. With these data points, I also identify which group of factors have the most impact on player performance. I will also discuss the process of creating a supervised model, which has the potential to predict how well a player will perform in a game based on the input external factors for a particular game.

Introduction

With the ever-increasing integration of technology in the world of sports, the need to handle data and interpret it in a meaningful way is paramount. Movies, such as Moneyball (1), provide an insight into how powerful harnessing sports data and using it to make decisions really is. Additionally, with the increase in the amount of data available in recent years, it has been easier to develop models which can predict player performances using on-court data. Models with these capabilities can prove to be of high importance to teams, who can determine how to change their game plan based on the model's predictions. Thus, in recent years, teams have started leveraging data analytics to help gain an edge in what's becoming a game of very fine margins.

Even though player performance is, for the most part, predictable, there are slight performance fluctuations for each player. A player may score 5-10 more or less points than what's expected from them during a game. These subtle changes, however, have mostly been deemed as "random" - a player can just simply not "feel" his usual self before a game. A more famous example is a "shooting slump" (2), which is when a player fails to score as much as he should during a stretch of games.

Each player can have a multitude of factors affecting his performance, like the weather, arena capacity, or the day of the week. I categorized these factors into 5 groups:

1. Weather Data - temperature, average precipitation, city played in.
2. Opponent Information - opponent division, opponent conference, opponent coach.
3. Arena Information - game attendance, stadium capacity, % of arena filled, home check.
4. Date Information - day of the week, month.
5. Player Information- height, weight, age, player form.

To test the importance of each of these external factors on player performance, a supervised model would need to be developed which has the ability to take all of these factors as inputs and output a player's predicted performance. This paper explores the process behind creating this supervised model and examines which features have the biggest impact on player performances.

Supervised models revolve around using an algorithm and training data to train these algorithms and make predictions on the testing data (3). The usage of supervised models, such as Linear regression, Random Forest, and support vector machines, have been used before to predict player performances in an abundance of papers in the past- The model described by Kevin Wheeler (4) uses linear regression with Naive Bayes and support vector machine (SVM) implementations to draw relationships between a player's performance with player-level and team-level statistics. Nguyen, Nguyen Hoang, et al (5) also make use of supervised regression models, like linear regression, support vector machines, and random forest, to determine a player's popularity, using the frequency of all star selection as a feature. Similar to the model by Kevin Wheeler (4), the inputted/predictor features only included in-game statistics, such as points, minutes, and shooting percentages.

Metrics, such as DARKO (6) (**D**aily **A**djusted and **R**egressed **K**alman **O**ptimized projections) and LEBRON (7) (**L**uck-adjusted player **E**stimate using a **B**ox prior **R**egularized **ON**-off), can predict how much a player will impact future games based off of their previous performances.

DARKO (6) uses box score data, such as points scored by a player in a specific game, assists, and other game-specific metrics. Similarly, LEBRON (7) uses box-score data to create a new feature, known as box-prior. This box-prior score allows for the model to have a better picture of a player's full performance rather than going off player averages. In the papers and models above, all of them have used supervised regression models; however, there is a lack of external factors as features in any of their papers.

Models

To create a model with the ability to predict player performances with external and in-game factors, I chose a variety of supervised models, as the aim of the study is to predict Game Score (8), a real value, from a set of variables. More specifically, I chose 3 supervised models: Random Forest, K-Nearest Neighbour and Linear Regression. I imported all of the regression models with the scikit-learn library (9) from python.

Random Forest

Random forest is a type of ensemble machine learning algorithm that's used for classification and regression tasks. It consists of many smaller decision trees coming together to make a 'forest.' A decision tree decides the best way to split the dataset to predict the targeted value. Each of these decision trees predict their own value, and the random forest will take the average of all decision trees (ensemble), and output the result as the predicted value, thus making it a more accurate model than a normal decision tree (10).

The most important aspect of the Random Forest algorithm is that the individual decision trees must be uncorrelated, so that the Random Forest will function better than any individual decision tree. The trees should be uncorrelated so that the random forest algorithm **will not make the same mistake twice** - it protects the other decision trees from an individual mistake.

The Random Forest algorithm can achieve this degree of uncorrelation in two ways:

1. Bagging / Bootstrap Aggregation - Decision trees randomly sample features from the dataframe with replacement, resulting in different kinds of decision trees that have no correlation with each other (11).
2. Feature Randomness - Picking a feature with the highest degree of separation between the two decision nodes in a decision tree (10).

For my Random Forest model, I set the max_leaf_nodes to 4000 and random_state to 4 to give the best predictions for the model. The max_leaf_nodes set the number of the terminal nodes in the model, and random_state will determine the shuffling of the data before it gets split into training and testing data.

K-Nearest Neighbor

The K-nearest neighbor algorithm is a supervised learning algorithm which uses the distance between data points to make predictions about a specific 'query' point (the point you are trying to predict). In K-Nearest Neighbour regression, the model will average 'k' data points that are closest to the query point, and that average is the value of the query point (12). In KNN classification, the most repeated class that is near the query point is the query point's new class. To determine the points which are closest to the query point, we need to calculate the distance between the points. Distance is commonly computed using the Euclidean metric (12). This formula will measure the distance of the straight line between the two defined points. The formula is given below.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} \quad (y_i - x_i)^2} \tag{1}$$

There are other distance metrics which can be used as well, such as the Minkowski distance, Manhattan distance, and the cosine distance (13). To define how many neighbors, we should take the average of (k), we can square root N (√N), where N is the total number of samples in the dataframe. In my KNN model, I set the k value to 120, approximately the value of. √15,844, which is the number of samples in the training dataset.

Linear Regression

Linear regression is a type of machine learning algorithm which is used to predict the value of a specified target variable based on other input features assuming a linear relationship between inputs and outputs (14). This is in contrast to the previous two models which do not posit any particular mathematical relationship. . There are two types of variables:

1. Independent variables - the variables that you change to get the result. In my model, the independent variables are the external factors.
2. Dependent variables - the variable whose value you want to measure. In my model, the dependent variable is Game Score.

A linear regressor will attempt to plot a straight line between the independent variable, which is on the X-axis, and the Dependent variable, which is on the Y-axis. The straight line will prove whether or not there is a linear relationship between these two variables. The equation of the line, in the form Y=M*X + C, defines the relationship between the two variables, and using this relationship, we can predict the values of Y for a given X value.

There are multiple types of linear regression, such as:

1. Simple Linear Regression - Is it a linear function, defined by $y= a_0+a_1x+ \varepsilon$, where a_0 and a_1 are the constants which represent the regression slope, and ε, is the error function. There is only one independent variable and one dependent variable for a simple linear regression model. For a good model, the value of ε should be close to 0.
2. Multiple Linear regression - Multiple linear regression is similar to simple linear regression; however, there are multiple independent variables for one dependent variable. It is now defined by $y= a_0+a_1x+ a_2x+...a_nx + \varepsilon$, where 'n' is the number of independent variables. Like linear regression, the value of ε should be minimized for a good prediction.

There are multiple independent variables which need to be examined in this study, so multiple linear regression would be the most appropriate linear regression type to use (14).

Metric

The metric that I used to evaluate the accuracy of the predictions of my model with the testing dataset is the mean absolute error (MAE) (15).

Mean absolute error (MAE)

The mean absolute error is the mean of all absolute values of the error calculated. The absolute error calculated can be given by the formula $x_i - x$ where x_i is the value predicted and x is the true value. The formula for mean absolute error is given below.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} ■ |x_i - x| \quad (2)$$

Where 'n' is the total number of errors (15). This metric can give us a good sense of the discrepancy between the model's prediction and what was the actual value present in the testing set. The lower the value of the MAE, the better, as we want to minimize the amount of difference between the predicted value and true value as possible.

Materials and Methods

I had to make use of secondary data sources and compile them into one primary table which would contain both in-game stats as well as the external factors. In order to do so, I utilized the Pandas library from Python.
The player box scores section of the NBA Stats website (16) contains the in-game performance of each individual player for each game over the regular season or postseason. In order to scrape the data off of the website, I used the request python module and the 'https://stats.nba.com/stats/leaguegamelog' API endpoint and put the scraped data for the 2022-2023 season into a .CSV file. The initial table contained season ID, team ID, player ID, GAME_DATE, GAME_ID, and in-game information, such as points scored, assists, rebounds, blocks, steals, field goal percentage, and turnovers. As the value that we are trying to predict, which is Game Score, is in this dataframe, I considered this to be the primary dataframe.

Using points scored in a game as a performance metric was, however, unsatisfactory, as purely the number of points scored in a game does not represent the entire contribution that a player gives to the team during a game. Therefore, I used John Hollinger's Game Score metric (8), which combines all of the stats in a player's game to give one all-around metric that sums up a player's impact over the course of the game. Game Score is an inherently better metric to use for player performance than points scored, as it takes into account every aspect of a player's game, such as defensive stats, field goal percentage, and player efficiency (turnovers). This allows players who are more defensive–minded having the same game score as more offensive-minded players, even though they may not score the same amount of points.

The formula for Game Score is:
Game Score = Points + 0.4 * Field Goals - 0.7 * Field Goals Attempted - 0.4 (Free throws attempted - Free throws made) + 0.7 * Offensive rebounds + 0.3 * Defensive rebounds + Steals +0.7* Assists + 0.7 * Blocks - 0.4 * Personal Fouls - Turnovers.

I entered this formula in a separate column in Excel and applied said formula for all of the rows of the dataset. Upon entering this formula and obtaining all of the values for Game Score I created a histogram for all of the Game Scores in the league for the 2022-2023 season. On examining the histogram for Game Score, I could infer that the modal values of Game Score in the league were between -5 and 0, with a mean of game score of 8.33. The histogram for Game Score is also positively skewed, indicating that a higher number of values are clustered at the left end of the distribution, while the right end of the distribution is longer. A positively skewed curve also means that the mean is greater than the median, and the median is greater than the mode.

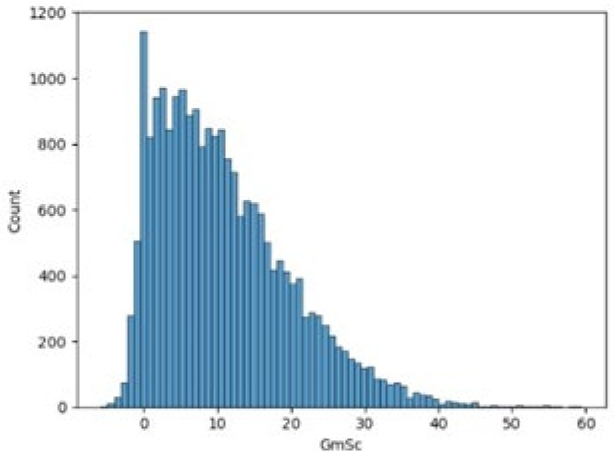

Figure 1. Histogram of Game Scores for every game played in the league for each player.

Each individual player has a specific PLAYER_ID, GAME_ID, and TEAM_ID for each game, which aided in mapping each external factor to a specific player and game. In order to obtain the opponent ID, I used the NBA Database by Wyatt Walsh (17), and more specifically, the game.csv file. In the game.csv file, each row contains GAME_ID, team_id_home, and team_id_away. I used an IF statement, to check if the player's TEAM_ID is equal to team_id_home for a specific GAME_ID. If the condition is true, then I would return team_id_away and vice-versa. I then used the pandas.apply function to call the returned value and apply it to each row in the database. With opponentID, I was able to add other opponent information, like opponent conference and opponent division with the pandas.merge function. With team_id_home, I was also able to add the city the game was played in, with the help of the team_info_common.csv file in the same NBA Database.

There was a lack of data available for the attendance of specific games, so to obtain attendance data, I used average attendance data for the 2022-2023 season from ESPN (18). I scraped all of the data off the ESPN website for the season and used the team_id_home column in the table to merge both datasets. The stadium capacity was available from the previously-used NBA Database in the team_details.csv file. The percentage of the arena was found by dividing the average attendance by the stadium capacity and multiplying it with 100.

The date information, such as day of the week and month the game was played in, were derived from the GAME_DATE column in the primary table. With the help of the pd.dt.month function in pandas, I was able to extract the month the game was played in. Day of the week was, however, extracted with the help of Excel. I used the formula WEEKDAY to take out the day of the week from the GAME_DATE column and converted the binary digits of the WEEKDAY column to categorical variables.

I used the Daily Average Temperatures of Major Cities database on Kaggle (19) to add the temperature information for each game with the pd.merge() function on pandas. I added the city, month, day and year to the primary table to match the daily average temperatures table and merged them together, getting the temperature for that day in the specific location. For precipitation, I used monthly averages from the National Centres for Environmental Information website (20) for precipitation as there was a lack of daily values in the 29 cities games are played in.

The categorical variables, like city played in, opponent abbreviation, day of the week. and month cannot be directly understood by the model. Therefore, I used the pd.get_dummies function to convert all of the categorical variables in the primary dataframe into 0's and 1's, which is something that the model can understand. The inclusion of PLAYER_ID as one of the features to allow the model to identify certain players also had to be converted into 0's and 1's with get.dummies, as simply inputting them as a feature will raise the problem of the model assuming that there is an order between two players based on PLAYER_ID.

To add a player's form as a feature, I approached the problem in two ways. One way was taking the rolling mean of a player's last 2 game scores. This was done with the help of the .groupby, .shift, .lambda, and .apply functions in pandas. The other approach was by adding a player's average over the course of the entire season as his form, which was achieved with the .average function in pandas.

Results

I randomly split the data frame into a training and testing set, with the training set consisting of 15,844 rows and the testing set consisting of 5,282 rows. The table below shows the results of each model when inputted with the testing dataset.

Table 1. Accuracy scores for each of the three supervised-learning models on the dataset. Input the testing data into the trained model and saw how accurate each prediction was against the actual prediction.

Model Used	MAE
Random Forest	4.86
Linear Regression	4.90
K-Nearest Neighbor	6.062

As shown in the table above, the Random Forest model had the lowest mean absolute error (MAE), showing that the model has the least error out of the 3 chosen supervised models and was the best at predicting the Game Score of the training data set.

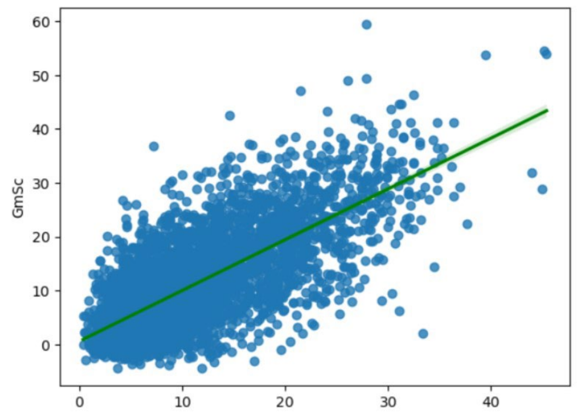

Figure 2. Scatterplot with predicted Game Score values on the X-axis and actual game Score values on the Y-Axis.

To obtain the most important features from the best predicting model, which is Random Forest, I removed certain features from the model and saw how much the model's MAE increased, which shows the performance drop of the model. The table below shows the MAE of the model upon removal of the group of features -

Table 2. The accuracy scores for the random forest model upon removing certain groups of features to check the importance of each feature.

Feature Group name	MAE after removing feature group
Weather Data	4.85
Opponent Information	4.89
Arena Information	4.87
Date Information	4.94
Player Information	5.01

Discussion

It is evident from the findings above that the features that have the highest impact on player performance are player height, player age, player weight, and a player’s form, as removing them from the model decreased the model’s accuracy by the greatest amount in the most accurate model, which is random forest.
Surprisingly, the model performed worse with the rolling mean than without the rolling mean, and the average game score proved to be a much better predictor of a player’s performance. This goes to show how volatile a player’s performance is on a game-to-game basis, which is why an average of all the game scores of a player’s entire season is a better indicator of performance than rolling mean.

This model can come into use in many scenarios in our day-to-day lives. One example is betting - usually betting is based on previous data and opponent information, although this information is not enough for a reliable bet. A bettor usually bets with his ‘gut’, and winning these bets usually comes down to luck.The bettors can get a sense of which factors to look out for while placing their bets, and the addition of the other external factors can help bettors place their bets with more certainty with the added reliability of the external factors.

As discussed above, team coaches can also use this model to modify their line-up before matches. After inputting the external factor data, coaches can get a sense of which players usually play better in these environments, which can help them win more games. In the same manner, Fantasy basketball players would also benefit from the increased reliability in how well a player will perform before a certain match to help choose their team.

By looking at the features which have the biggest impact on player performances, teams can attempt to mitigate the effect of certain conditions to improve player’s performances. This is already common practice around the league, but this could surely increase the amount of control the teams will have over their players’ surroundings.

Future Work

More work could be done in adding more key external features that I have not considered in this experiment. There may be something that is key to a player’s performance that I could have overlooked. Analyzing media coverage of a specific player before a game, or the publicity surrounding a game, is an example of another feature. Vectorizing news articles that have been released in close proximity to a game or containing key words about a player could be analyzed and input into the model to check how they can affect a player’s performance. An app which contains my model could also be developed to make the results accessible. The user could input certain external factors and player and get the desired game score for the player. The general public and team coaches would benefit greatly by this application.

Conclusion

The results from this experiment show that a player's form is the predominant factor when it comes to a player's performance. With readings of mean absolute error (MAE), Random Forest proved to be the best model to predict player performances, compared to Multiple regression and K-Nearest Neighbor regression.

My results prove that a player's performance can be affected by even the smallest changes in his environment, which shows why teams invest so heavily in ensuring that a player's condition and surroundings are satisfactory. The findings of this paper also show just how many inputs go into how well a player performs, not only in the NBA, but sports in general. This is why the lives of athletes are extremely controlled by teams - they monitor their sleep, food, and water religiously. This is also why teams are prepared to invest heavily into athlete conditioning, to make sure the players are in the best shape to give their all on the court. The experiment also gives teams more scope for controlling player conditions in an attempt to increase player performance.

Although the model is able to predict a player's performance fairly well, with a coefficient of determination of 0.35, the accuracy should be increased to give the user some added confidence in the model's predictions. The addition of more features, using different algorithms, or altering the hyperparameters are certain ways to achieve this goal.

Limitations

The main limitation that I faced in this experiment was the dearth of precise data. For instance, there was a lack of game-to-game attendance information, so I had to alternatively use average attendance information in my primary table. The use of average attendance does not give a full picture of how attendance can really affect player performances. The attendance for each home game of a player would also be repeated, which skewed the results.

The final primary table only contained regular season games, as I was only able to scrape that off the NBA stats website at the time. The inclusion of post-season matches can also be added in the future. Additionally, a reason why the rolling mean of a player's last 2 games was not a good feature could be due to the quality of data.

References

1. "Moneyball." *IMDb*, IMDb.com, 23 Sept. 2011, www.imdb.com/title/tt1210166/.
2. "What Is a Slump in Sports? Definition & Meaning on Sportslingo.com." *What Is A Slump In Sports? Definition & Meaning On SportsLingo.com*, 28 Jan. 2022, www.sportslingo.com/sports-glossary/s/slump/#:~:text=Slump%20In%20Basketball%3F-,1.,shooting%20technique%2C%20injury%20or%20fatigue.
3. "Supervised Machine Learning - Javatpoint." *Www.javatpoint.com*, www.javatpoint.com/supervised-machine-learning.
4. Wheeler, Kevin. "Stanford University." *Predicting NBA Player Performance*, cs229.stanford.edu/proj2012/Wheeler-PredictingNBAPlayerPerformance.pdf.
5. Nguyen, Nguyen Hoang, et al. "The Application of Machine Learning and Deep Learning in Sport: Predicting NBA Players' Performance and Popularity." *Journal of Information and Telecommunication*, vol. 6, no. 2, 2021, pp. 217–235., doi:10.1080/24751839.2021.1977066.
6. Medvedovsky , Kostya. "What Is Darko?" *DARKO Exploration*, apanalytics.shinyapps.io/DARKO//.
7. "Lebron Introduction." *Basketball Index*, 24 May 2022, www.bball-index.com/lebron-introduction/.

8. "Game Score in Basketball Explained." *NBAstuffer*, 13 June 2020, www.nbastuffer.com/analytics101/game-score/.
9. "Learn." *Scikit*, scikit-learn.org/stable/.
10. Yiu, Tony. "Understanding Random Forest." *Medium*, Towards Data Science, 29 Sept. 2021, towardsdatascience.com/understanding-random-forest-58381e0602d2.
11. "What Is Bagging?" *IBM*, www.ibm.com/in-en/topics/bagging#:~:text=Bagging%2C%20also%20known%20as%20bootstrap,be%20chosen%20more%20than%20once.
12. "What Is the K-Nearest Neighbors Algorithm?" *IBM*, www.ibm.com/in-en/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point.
13. "Most Popular Distance Metrics Used in KNN and When to Use Them." *KDnuggets*, www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html.
14. Kanade, Vijay. "What Is Linear Regression? Types, Equation, Examples, and Best Practices for 2022." *Spiceworks*, 3 Apr. 2023, www.spiceworks.com/tech/artificial-intelligence/articles/what-is-linear-regression/.
15. Glen, Stephanie. "Absolute Error & Mean Absolute Error (MAE)." *Statistics How To*, 28 Dec. 2020, www.statisticshowto.com/absolute-error/..
16. "Players Box Scores: Stats." *Players Box Scores | Stats | NBA.com*, www.nba.com/stats/players/boxscores.
17. Walsh, Wyatt. *NBA Database*, Kaggle, www.kaggle.com/datasets/wyattowalsh/basketball.
18. "NBA Attendance Report - 2021." *ESPN*, ESPN Internet Ventures, www.espn.com/nba/attendance.
19. Rajkumar , Sudalai. "Daily Temperature of Major Cities." *Kaggle*, 5 June 2020, www.kaggle.com/datasets/sudalairajkumar/daily-temperature-of-major-cities.
20. NCEI.Monitoring.Info@noaa.gov. "City Time Series: Climate at a Glance." *City Time Series | Climate at a Glance | National Centers for Environmental Information (NCEI)*, www.ncei.noaa.gov/access/monitoring/climate-at-a-glance/city/time-series/USW00023234/tavg/all/1/2021-2022.