NBA 2021 APRIL 13-30

BEST MODELS TO PREDICT SPREAD, TOTAL AND OFFENSIVE REBOUNDS OF GAMES

PLAYOFFS 2 - GROUP 8

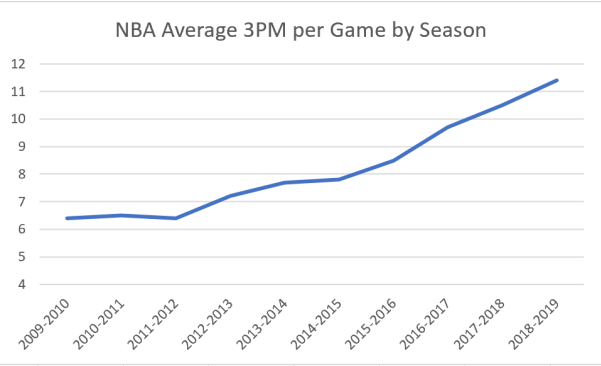
Izzy Hastings
Abby Kadlec
Adelynn Morris
Alexa Penwell
Nikhil Vytla

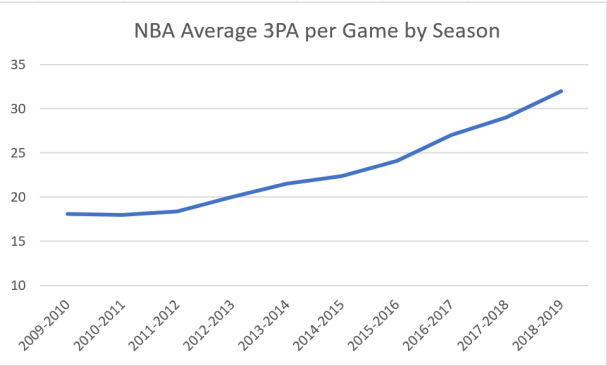
Data Information

When cleaning up and combining the GAMES_DETAILS and GAMES data sets, our group first cleaned up issues within each data set, then merged them. To start, with GAMES_DETAILS, we divided the data into two separate datasets, one for players who did play (DidPlay_rows) and one for those who did not (DidNotPlay_rows). This made it easier to address N/A values in the larger set because many of those values were due to a particular team member not playing in a game. All values in the COMMENT column for DidPlay_rows were N/A values, so we removed that column. Then, we changed N/A values in “START_POSITION” to “bench” to indicate that while the player did not start in the game, they did come off the bench to play. In the DidNotPlay_rows dataset, we deleted all columns that had N/A values for every player, leaving only GAME_ID, TEAM_ID, TEAM_ABBREVIATION, TEAM_CITY, PLAYER_ID, PLAYER_NAME, and COMMENT, which explains why that player was not in that particular game.

Next, we began to clean up the GAMES data set, starting by removing rows with N/A values because they would not give enough data to compute any necessary variables. This decision ended up eliminating 100 rows from our data. Following the code given to us, we then created the OREB variable and added that variable to the dataset. Next, we aggregated player level data for turnovers (TO), defensive rebounds (DREB), and field goal attempts (FGA) from GAMES_Details and merged them into the GAMES dataset to aid in developing our engineered variable (described below). We also created a separate variable to indicate whether each team was the home team or away team in the given game, then merged team name into the original data.

To create our final dataset, we merged the manipulated version of GAMES with the DidPlay data. Because this was a large amount of data, we ended up creating a subset so we would only use games from the 2014-2015 season through the current season. We chose to cut off our data at the 2014-2015 season because later seasons will have the most similar rosters to the current season, so going back to 2004 would not have been extremely useful. In addition, the primary reason we chose 2014-2015 as the cut-off is that, in recent years, the NBA has seen a significant uptick in the number of 3-point attempts and shots made, as shown in the graphs here. Alongside Steph Curry





and the Golden State Warriors' dominating championship run that season, we decided that data from 2014 onwards would most accurately reflect the impact of what is referred to in many professional circles as the "3-point revolution".

Next we split our data into train (80%) and test (20%) sets so that later we could build a model from the train set, then test our predictions using the test set. For our engineered variable, we decided to look at offensive inefficiency of teams, which we think could be used as a way to potentially predict all three of the variables we are trying to predict. The offensive inefficiency variables (OI_{home} and OI_{away}) are created by combining turnovers and defensive rebounds, and dividing by the number of scoring

opportunities the team had. To calculate the home team's offensive inefficiency we added home turnovers, since losing the ball is a missed scoring opportunity, and away defensive rebounds, which represents a missed offensive rebounding opportunity that could have led to a field goal. Ideally the number of possessions would be a better representation of "scoring opportunities" than field goal attempts, especially since if a team turns the ball over, they likely never attempted a field goal, but we could not find data on the number of possessions. The formula for home and away offensive inefficiency are as follows:

$$(1)\ OI_{home} = (TO_{home} + DREB_{away})/FGA_{home} \text{ and}$$

$$(2)\ OI_{away} = (TO_{away} + DREB_{home})/FGA_{away}\ .$$

After creating our variable and cleaning up the data, we decided that it would be beneficial for our predictions if we brought in outside data from the current season, including games played until March 12th, 2021. This data would have the most accurate representation of how teams are currently performing and the most updated rosters. The data was taken from the same Kaggle collection (Nathan Lauga's NBA games data) as the original data we were provided with.

Methodology for Spread

Basic Modeling for Spread

In order to find the ideal model to predict Spread, we first conducted basic modeling, including linear regression and transformations. We used forward regression, backward regression and stepwise regression on our train dataset to see what variables

were most useful in predicting the spread variable. Within the three basic models, the predictors were constricted to game data, and we took out all player data. Game data encompassed the player data, so restricting predictors to game data mitigated any possible multicollinearity. Additionally, within the game data used, all totals of home+away variables were not included. For example, OREB_total was not included since OREB_home and OREB_away were included in the model.

Advanced Modeling for Spread

After finding a basic model to start with, we moved to more advanced modeling. Before we began building models, we modified our initial GAMES dataset to also include the respective home and away totals for the following stats provided on the player-level in GAMES_Details: FG_PCT, FT_PCT, FG3_PCT, AST, REB, and PTS. The first three variables were retrieved from the original GAMES dataset, and the latter three were the remaining numerical variables from GAMES_Details based on results from an initial correlation matrix analysis. Our first approach involved using recursive partitioning to formulate regression trees via the rpart library. Using k-fold cross-validation with k = 10, we attempted to preprune our model by adjusting our initial maxdepth parameter, and ultimately decided that the default value was the most optimal. This approach allowed us to attain a model with the following statistics:

$$RMSE = 6.1206499\,,\ R^2 = 0.8099611\,,\ \text{and}\ \ MAE = 4.6876823\,.$$

Our second approach involved computing a random forest model via the randomForest library in R. When creating this model, we looked at the same numerical variables as we did in our regression tree model, only this time, we used k-fold

cross-validation with k = 10 to check the accuracy of the model, then made adjustments regarding the number of trees (nTrees) and the number of variables randomly selected at each split (mtry). The primary challenge with this method was attempting to combat overfitting during our tuning. In our initial tuning attempts, we realized that random forest models are similar to recursive partitioning, only instead of splitting all the variables at each point, the RF model randomly selects mtry variables from the set of predictors available. Our analysis showed that the ideal value for mtry was 24, the same number as the number of total variables inputted into the model, but this attempt ran into severe issues with overfitting on our training set. We thus opted to grow a larger forest with a reduced mtry value of 8 in order to govern the number of features randomly chosen to grow each tree from the bootstrapped data and to improve predictive accuracy, which attained the following statistics:

$$RMSE = 3.567746\,,\ R^2 = 0.972674\,,\text{ and } MAE = 2.605515\,.$$

This tuned Random Forest model concluded our advanced modeling. A summary of the information gathered during the modeling phase can be seen here:

Type of Model	RMSE
Forward Regression	7.851554
Backward Regression	7.851554
Stepwise Regression	7.851554
Decision Tree Regression - (Random Partition)	6.1206499
Random Forest Regression - Tuned Parameters: `mtry`=8, `nTrees`=500	3.567746

We determined that the model with the lowest $RMSE$ and highest R^2 would best predict Spread, and thus we chose the tuned Random Forest model for our final predictions. To make our final predictions we first downloaded the Predictions file provided to us. We then subset our GAMES dataset to only include games from the current season (beginning December 22nd, 2020). We grouped this smaller dataset first by the home team and averaged each variable applicable to our model, with the output of this operation being a dataset that had a row for each team with their corresponding variable averages. We then repeated this method for the variables corresponding to the away team. Next we left-joined the Predictions dataset to our new Home dataset then left-joined that dataset to the Away dataset. After these manipulations we were able to run our model to obtain our predictions.

Methodology for Total

Basic Modeling for Total

To get a basic idea of what a model could look like for total, we first attempted to run stepwise, forward, and backward regression, but did not find a model that worked for the variables. Next, we created a correlation matrix, comparing Total to all the other variables, including the one we created and the data we brought in. Based on the correlation matrix, the variables that had the strongest positive or negative relationship with Total were FGA_home, FGA_away, OI_home, and OI_away. The last step of looking at basic modeling was to plot the relationship between Total and each of the four strongly correlated variables mentioned above.

Advanced Modeling for Total

After looking at the correlation matrix obtained during basic modeling, the variables deemed as important to include in our models, through them either having a medium to strong positive or negative correlation with Total, are as follows: FGA_home, FGA_away, OI_home, and OI_away. The Home Team and Away Team variables were also included in order to be able to make our final predictions for Total. We then looked at generating advanced models using our previously created train data that included the above mentioned variables. Once we had a model equation for each type of advanced model, we generated predictions for Total using the test data. We then calculated the RMSE of each model. These values can be seen here:

Type of Model	RMSE
Linear Regression	17.35717
Decision Tree Regressor	18.9193
Decision Tree Regressor - Bootstrapped	18.2451
Random Forest Regressor	17.89661
Support Vector Machine Regressor	23.44046

Looking at the RMSE values obtained during modeling, we decided to work on tweaking our linear regression model since this model had the lowest starting RMSE value. We first looked at interaction terms to see if these manipulations would help us reduce our RMSE and we found that interacting FGA_home with FGA_away and OI_home with OI_away reduced our RMSE to 17.30443. We then tried to further reduce this value by looking at log and square root transformations. After completing these transformations we obtained a final RMSE value of 17.26066 and final MAE value of

13.71518. Although these values appeared high to us, we determined that they were satisfactory to make our predictions. After selecting our best model, we then followed the same methodology outlined in the Spread section to obtain our final predictions.

Methodology for OREB

Basic Modeling for OREB

In order to find an ideal model to predict OREB, we first started with looking at basic models, such as linear regression and Poisson. Three basic models were built: a stepwise linear regression, a stepwise Poisson regression, and a Poisson regression using variables from a correlation matrix with OREB_total. These three models had correlations of 0.5509, 0.5499, and 0.4634, and RMSEs of 4.307, 18.178, and 18.181, respectively. Three variables were significant in at least two models: FGA_home and FGA_away in all three models, our engineered variables OI_home and OI_away in the linear model and correlation matrix based Poisson model, and DREB_home and DREB_away in the linear and Poisson stepwise models. These three models served as starting points for more advanced and effective models for OREB.

Advanced Modeling for OREB

After more fully exploring the correlation matrix that was developed during basic modeling, we narrowed down the variables to use in advanced modeling to be TO_home, TO_away, DREB_home, DREB_away, FGA_home, FGA_away, OI_home, and OI_away. The Home Team and Away Team variables were also included in order to be able to create our final predictions once the best model was determined. Although the

Total variable calculations and OREB variable calculations are both count variables and we followed the same methodology to obtain the variables needed for our best model, there are a few notable reasons as to why the same variables aren't included in these two models. The reason we see defensive rebounding variables included in OREB but not in Total is because if a team is good at all around rebounding, then both their metrics for OREB and DREB will probably be higher, and vice versa for a team that is not good at rebounding. Defensive rebounding wouldn't matter in regards to Total because these plays are not shown to directly impact how many points a team scores given the fact that the players are not near their scoring basket when they receive a defensive rebound. Essentially, a defensive rebound does not mean that the team will be efficient when on offense after getting the rebound, and it would not necessarily impact the Total. Similar reasoning can be applied to the inclusion of turnover metrics in OREB and not in Total. Turnovers don't guarantee a scored point for the team who has the possession after a turnover, but if a team has a lower amount of turnovers and defensive rebounds then you can assume they are keeping more control of the ball, which could be shown in part by their number of offensive rebounds.

Moving on to the specific advanced models that we created, we looked at four new models as well as the linear model that we created during the basic modeling stage. Once we had a model equation for each type of model, we trained and tested our data, using the same methodology as the train and test data for Total, in order to generate predictions for OREB. We evaluated the accuracy of these models based on their RMSE and the results can be seen here:

Type of Model	RMSE
Linear Regression	4.307437
Decision Tree Regressor	4.866668
Decision Tree Regressor - Bootstrapped	4.738443
Random Forest Regressor	4.616722
Support Vector Machine Regressor	4.710741

In order to obtain the best predictions for OREB, we chose to focus on manipulating our basic linear regression model because it had the lowest starting RMSE value. We decided to look at the effects of interaction terms and log and square root transformations with the goal of tweaking our model in a way that would reduce our RMSE. This process was conducted mostly in a “trial and error” fashion where we looked at graphs of our model and then based our changes on ways we thought would make our line of best fit more linear. Once we made a change in model equation, we would then re-test our model using the same training and testing methodology used in the above models. After these manipulations were complete we ended with a model that had an RMSE of 4.283692 and an MAE of 3.423975.

Although we weren’t able to decrease our RMSE value as much as we had hoped, we think that this model is very easy to understand while still providing us with an RMSE value that we believe to be reasonable, which is why we chose this model as our best model. After selecting our best model, we then followed the same methodology outlined in the Spread section to obtain our final predictions.