Effects of NBA Team and Player Statistics on Predicting Wins/Losses & Standings

Drew Suranjan, Nancy Ji, Nilay Agarwalla
DS 100: Principles & Techniques of Data Science
University of California, Berkeley
May 2020

Abstract:

The National Basketball Association (NBA) was founded in New York City in the summer of 1946. Since then, there have been many different models and theories to predict a team's overall success. Provided with data frames consisting of NBA players and seasonal team statistics from 2012-2018, this paper discusses the use of a metric called PER (Player Efficiency Ratio) alongside several other offensive and defensive features such as turnovers, rebounds and defensive ratings in order to predict the win/loss ratio of an NBA team with a root mean squared error of approximately 8.47% for the seasons between 2012-2017 using a Ridge Regression model. In fact, the paper looks into more non-traditional metrics such as starter minutes played and BMI categories to find hidden correlations with a team's win/loss ratio. Using these win/loss ratios, this paper also attempts to predict the standings for the 2017-2018, using the data trained from previous seasons.

Introduction:

After initially exploring the different data frames provided within the basketball csv, our group decided upon a couple of different questions that we wanted to explore throughout the course of this project:

1) What are the best features from NBA statistics that can predict a team's overall success in a league, specifically its win/loss ratio?
2) In addition, can we use this win/loss ratio regression model to determine the NBA rankings (post-season) for future seasons based on regular seasons statistics?

We decided upon using the follow csv files which provide us with relevant statistics that we felt are important to use to predict wins/losses and standings in the NBA:

- 2012-18_standings.csv - provides us with data such as a team's ranking and game date during a certain game in a season
- 2012-18_teamBoxScore.csv - provides us with team/opponent statistics for each game such as the result of the game, game date, FIC score, total assists, points, steals, etc.
- 2012-18_playerBoxScore.csv - provides us with player statistics for each game such as game date, points, assists, 3 point percentage, etc. and attributes such as height or weight

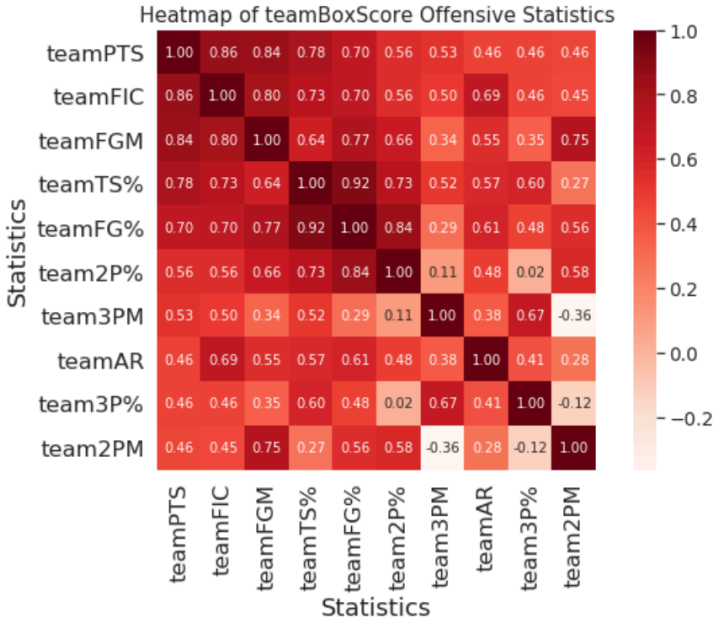
Description of EDA, Data & Methods:

Prior to testing any model, our group needed to discover the features we wanted to use. As mentioned above, there were three csv files from which we pulled statistics. Initially, we wanted to check for any null values within the three data frames we used. Using the isnull().sum() method, we were able to discover 4 games with null values in the teamBoxScore.csv and 41 games with null values in the playerBoxScore.csv all resulting from a missing referee for that game. In order to fill these null values, we performed the .fillna(" ", inplace = True) function which filled all the empty referee name data points as empty strings and made sure the data frames remained in that form. This was an ethical choice since we wanted to treat the missing referee names as having no meaning in the dataframe and using " " allowed the values to represent a computationally valid yet meaningless datapoint.

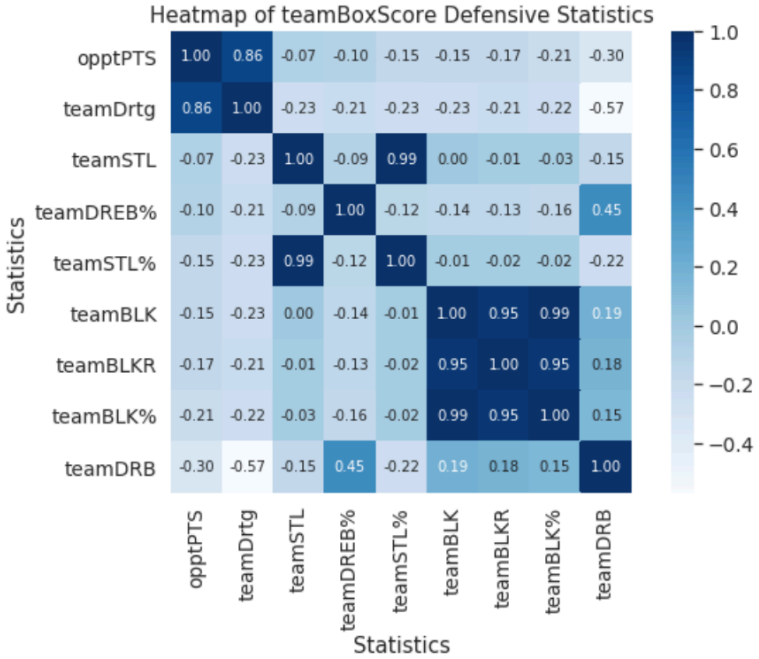
Next, we realized that since all the game dates were strings and not labeled by specific seasons that it would be difficult to group games by the seasons they were played in. In order to fix this, we needed to convert the string time date into a datetime object. This was done by importing datetime from the datetime library and creating a function date_fixer() which takes in a string of dates, parses through each date and returns its year, month and day as a datetime object. Once we applied this function to replace the date columns of all three data frames, we created a function called nba_season_finder_standings() which takes in a dataframe and column name. In this function, we create an empty list and iterate through each row of the specific column selected. In these iterations, we classify which season the date provided falls into based on the opening and

closing months of the NBA regular season and add them to the empty list which is eventually returned. After applying this function to the three data frames, we now have the season in which each game occurred in a list.

Now, we wanted to explore the different offensive and defensive features provided and figure out which ones would be most useful in predicting how well a team would do in a season. For the offensive features, we decided to create a heatmap, pictured below, and to find the statistics that had the highest positive correlation with team points as we believe those metrics would be a good indicator of offensive success in a game. However, we will not use team points as a feature at any point in our model because it is a biased feature and is too easy to create a model around as the team with the higher number points will obviously win. Based on the heatmap, we decided to use teamFGM (field goals made), teamFG% (field goal percentage), teamAR (assist rate) and team3P% (3-point percentage). We excluded teamFIC (Floor Impact Counter) despite its high correlation with team points because its formula includes team points. We decided to combine these features into one offensive score by creating a function called offensive_teamscore_calculator which takes in a dataframe and returns an output based on this formula: (3 * df["teamFGM"] + 100 * df["teamFG%"]+ 2 * df["teamAR"]+ 100 * df["team3P%"]). Different weights were placed on the features when combining them based on our basketball knowledge as well as to balance out the impact of each feature on the offensive score. For example, we placed weights of 100 on teamFG% and team3P%, which are represented as decimals between 0 and 1, so that they would have similar impacts to the score as teamFGM and teamAR would have as positive numbers much greater than 1. With this formula applied to the teamBoxScore dataframe, we created a new offensive_score column.

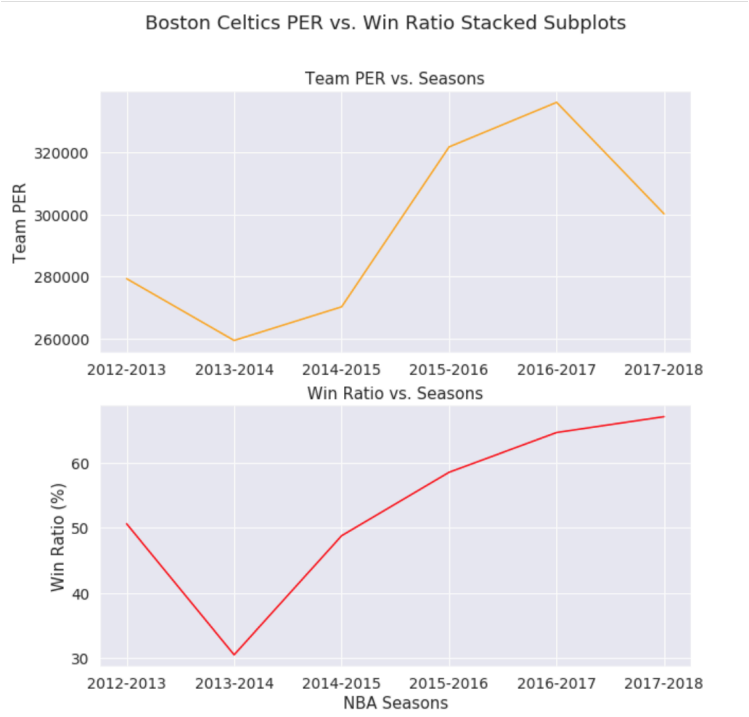


The aforementioned process was done for defensive features as well except this time, we were looking for statistics with large negative correlations with opponent points as if your defense is doing well, you expect the other team to score less. The only exception to this is teamDrtg (defensive rating) because it already takes into account opponent points. Based on the heatmap, pictured to the right, we decided to use teamDrtg, teamBLK% (block percentage), teamSTL% (steal percentage) and teamDREB% (defensive rebound percentage). We decided to combine these features into one defensive score by creating a function called defensive_teamscore_calculator which takes in a dataframe and returns an output based on this formula: (3*x["teamDrtg"] + 11 * x["teamBLK%"] + 7 * x["teamSTL%"]+ (1/2) * x["teamDREB%"]). Different weights were placed on the features when combining them based on our basketball knowledge. We felt like teamBLK% and teamSTL% were most impactful in a game in terms of judging a team's defensive performance which is why they were given higher weights than teamDrtg and teamDREB%. With this formula applied to the teamBoxScore dataframe, we created a new defensive_score column.



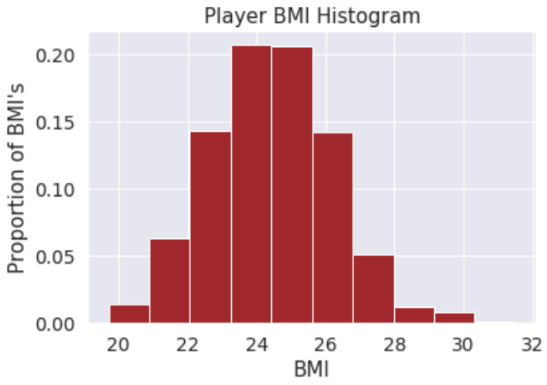
Now that we have the offensive and defensive score for every game in the data frame, we want to get a team's total offensive and defensive score for the entire season. We do this by grouping teamBoxScore by each season we had created earlier and teamAbbr (team abbreviation), calling both score columns and then summing them. This will create a dataframe which consists of each team's offensive and defensive score for each season between 2012-2018.

For our next feature, we drew inspiration from an online article about player efficiency ratio (PER)[1]. This metric was invented by John Hollinger and measures a player's efficiency per minute by assigning different weights to each player's statistic and combining them all into one efficiency score. Using the PER_formula function, we created a new column for PER for each player by grouping the original player_box_score dataframe by player first name, last name, team, and season, and applying the function to all of the player features. We then accounted for the minutes played, as PER is a per minute metric, and put this in the PER_real_affected column. Then, we used our team_PER_calculator function to group the PER values for each team each season and used the PER's of the top 12 players from each team in terms of minutes played. Using the top 12 players like this helped us make sure we did our calculations based on the players that were most present during the season, and most likely were playing for the same team the entire season. We used this overall team PER as one of our features and mentioned later that it had one of the greatest impacts on reducing or Cross Validation error.



We graphed PER versus win ratio for each season in the subplot seen above. This graphic represents the data for the Boston Celtics (BOS) and it shows that there is a clear correlation between Team PER and Win Ratio over the seasons. This trend appeared for the majority of the other teams which is why we decided that Team PER would be a good feature to use.

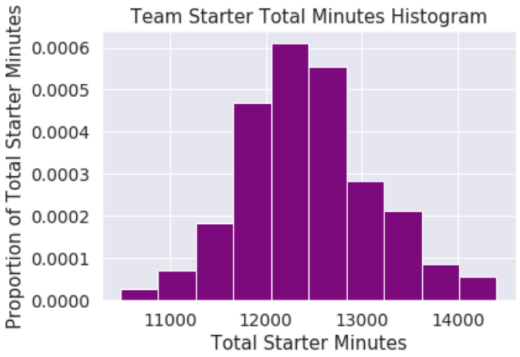
One method we thought about exploring was a player's Body Mass Index (BMI) because athletes have certain ranges that they consider ideal shape, underweight, overweight, etc. To do this, we created a histogram, pictured below, of player BMI's by using their weight divided by height[2], and noticed a normal distribution, suggesting that there might be an ideal height to weight ratio. We then used their BMI's and our bmi_containers() function to place athletes into categories — underweight, slightly underweight, ideal, slightly overweight, overweight — based on this ratio. We decided to make the category bins a bit higher than what they normally are (underweight is usually <18 as opposed to our bin of <21) because we would expect an athlete to be heavier than an average person. This is due to athletes having more muscle mass, with muscle weighing more than fat. Then, we used one-hot encoding on this categorical variable to convert the categories to columns of 0's and 1's, the numbers indicating what category each player belongs to. We then grouped the resulting dataframe by player first, then team and season, and took the aggregate sum to get the total count of each BMI category for each team, each season. Later, we will discuss whether or not we decided to use BMI as a feature for our model.



We knew we wanted to use some more features but weren't sure which ones to decide upon. After doing some research, we came across an article titled "Game Indicators Determining Sports Performance in the NBA" by Kazimierz

Mikolajec, Adam Maszczyk and Tomasz Zajac[2]. In this article, it is proven that some of the most critical indicators of game effectiveness in the NBA are fouls and steals. For this reason, we create two lists in which we group the playerBoxScore by teamAbbr and season and then sum playPF (personal fouls) and playSTL (steals) so we have a total number of steals and fouls per team per season.

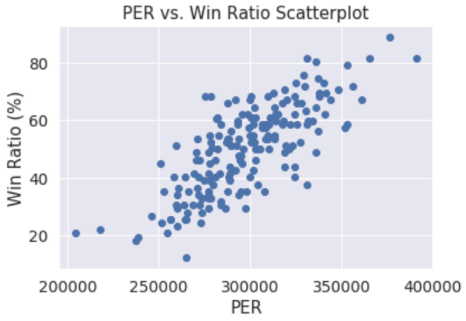
Next, we thought that maybe a team's division would affect their win ratio, as some divisions can be much harder and better than others, so we used one-hot encoding on the categorical variable of a team's division. The 1's represented the division that each team was in and the 0's represented every other division. We used the division category as a feature at first, but then later removed it from our model, which we will go more into detail about later in this report.

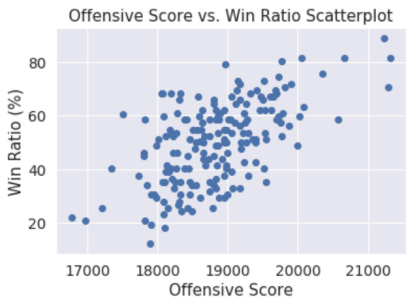


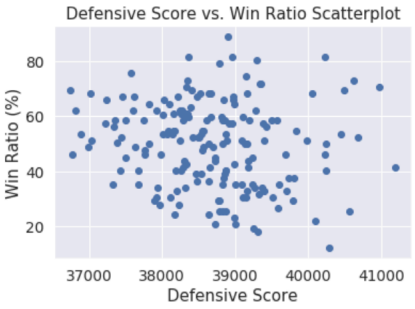
We also wanted to use the amount of playing time of the starters of each team as a feature. Our reasoning was that starters are usually the best players on the team, so them playing more would increase the team's performance. We also created a histogram of total minutes played by starters for each team just to see what the distribution was, and we saw that some teams had starters playing a lot more minutes than others, so we then decided to use the total starter minutes as a potential feature as well.

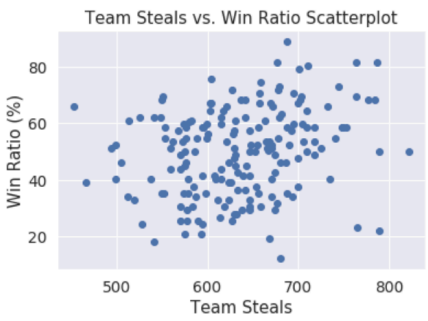
At this point, we have created all the features we would potentially use for our model. Now, we must calculate the value we are trying to predict which is the win ratio for every team. We do this by grouping the standings dataframe by teamAbbr and season and then finding the max amount of games won for each team during that season by using .agg(max) into a date frame called win_ratio_data. To calculate the win ratio, we use the formula: win_ratio_data['gameWon']/(win_ratio_data['gameWon']+win_ratio_data['gameLost'])*100. This formula is applied and is added to the win_ratio_data dataframe as a new column called win_ratio.

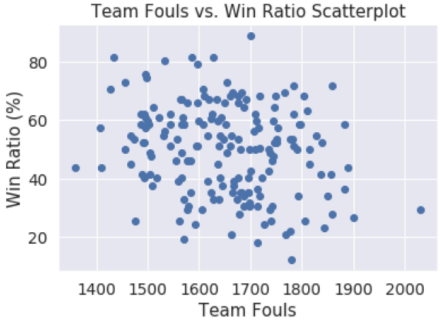
Next, we wanted to explore the correlations between some of the features we chose and the win ratios to make sure our features were acting the way we intended. As expected, PER, Offensive Score, Defensive Score and Team Steals showed a positive correlation with win ratio. Also, as expected, team fouls showed a negative correlation with win ratio. The Ideal BMI Count scatterplot shows us something unexpected when you see that the more ideal players you have, the less likely you are to win. We will continue to explore BMI and its usage as a feature later on. Shown below are the scatterplots we created:

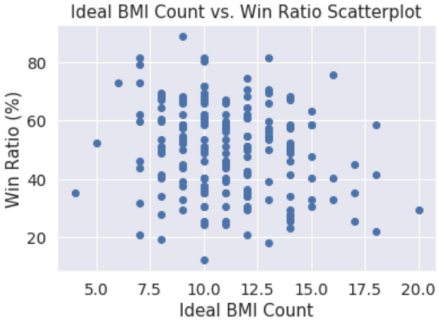








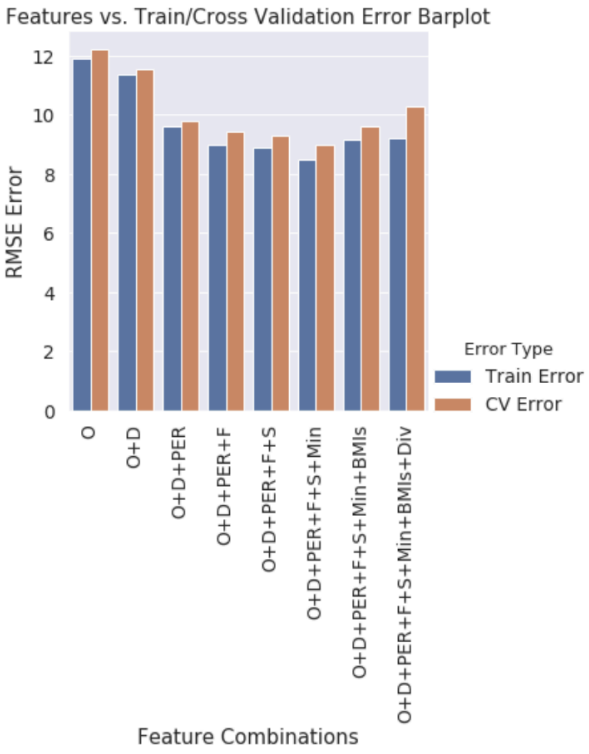




After deciding on all of our features, we merged all of them into one dataframe, full_seasonal_data, and standardized all the quantitative ones, which was necessary because a lot of the features are measured on different scales. We did this using StandardScaler, which we imported from sklearn.preprocessing. We also in this step classified our data into the Eastern and Western conferences so that we could rank each team in their respective conference. We did this because when we were performing EDA on the data we found that for each season there were two of the same rankings, making us come to the realization that each conference in the standings data has its own respective rank. We found the ranking by getting each team's last game of each season using an aggregate max function over 4 columns in the original standings table and set this to season_rank_holder. Then we dropped the "rank" column from season_rank_holder (because our max function would've taken each team's highest and worst rank) and replaced it with the original rank column from standings by merging the two dataframes together, calling this new dataframe season_rank. Then we merged both the conference data with season_rank into full_seasonal_data.

After discussing, we figured that the best way to test how accurately we could predict the win ratio would be through a Ridge Regression model testing root mean squared error (rmse). We used a Ridge model instead of a basic Linear Regression model because we wanted to take into account overfitting through the regularization of the data. In order to do this, we first created a rmse function that would calculate rmse from actual and predicted values. Next, we split the training data -- seasons 2012-2017 --- into a feature matrix (X) and what we are predicting in win ratio (Y). Once we did this, we imported a linear_model from sklearn and created a Ridge Regression Model. We fit the model to the X feature matrix and the Y win ratios for the 2012-2017 seasons data points. Then used the model to predict fitted Y values on the X features set. Once we have the fitted y values, we can find the rmse of that and the real Y win ratio set. We did not import train_test_split from sklearn.model_selection because the way our research question is posed is that our training and test are split by seasons. We used the first five seasons of the given data to predict the outcomes win ratios for the last season --- 2017-2018 season --- using the model we fit.

We also realized we would want to calculate the cross validation errors of the model so we could achieve a more accurate estimate of the test error and future season predictions. In order to do this, we had to import KFold from sklearn.model_selection as KFold decides how the data gets split up for cross validation testing. Next, we created the compute_CV_error function which results in the cross validation error for the model. We tested these errors on combinations of features (in order of us discovering them) and graphed the results to the right. Here is a legend for the graph: {O: Offensive Score, D: Defensive Score, PER: PER Score, F: Total Fouls, S: Total Steals, BMIs: counts of BMI categories, Div: Division Count, Min: Total Starter Minutes}. This graph displays the various training errors and Cross Validation errors based on different combinations of features. We used the CV error as an estimate of the test error, so that we could determine the best features for this model without actually seeing how well our model performs on the test data.

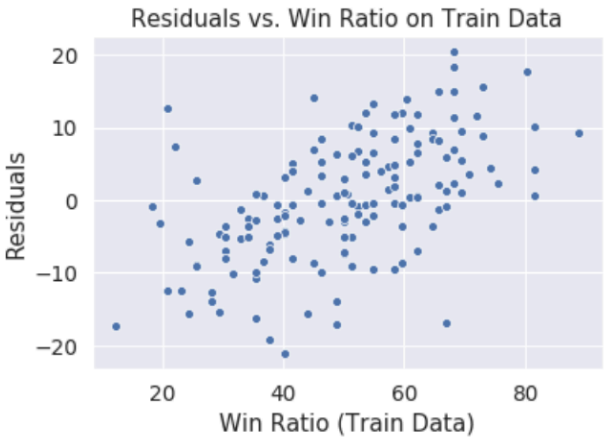


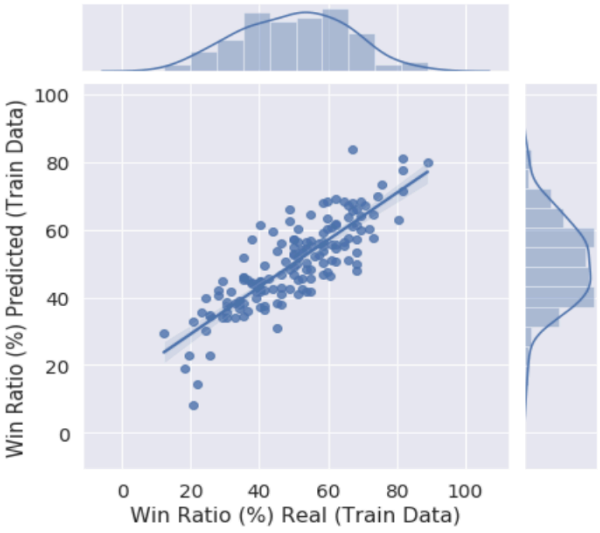
Features CV vs RMSE Error Barplot Highlights:

After noticing the strange correlation between win ratio and ideal BMI count, we noticed another strange event with the addition of BMIs to the combination of features. The CV error increased from approximately 8.98 to 9.6 when BMI was added to the features. In fact, then adding the DIV feature to the model it further increased the CV from 9.6 to 10.3. On the other hand, we noticed that when we added the MIN feature, the CV error decreased from 9.3 to 8.98, which influenced us to keep the feature. Since the BMI and DIV features increased the CV error, we decided to remove them as features we would use for our model. However, since the MIN feature decreased the CV error, we decided to keep it. In fact, the PER

feature had the greatest impact on our model, reducing the CV error substantially from 11.5 to 9.8. This showcases how powerful this metric is at determining how well a team did in a season.

Summary of Results:





Once our features were finalized to Offensive score, Defensive score, PER, total fouls, total steals and starter minutes, based on the CV error values, we ran our model on our training set again. The resulting outputs were as follows: RMSE training error of approximately 8.47 and Cross Validation error of approximately 8.98. The significance of the RMSE error is that on average, our predictions were about 8.47% away from the actual win loss ratio of the team. Furthermore, we plotted the residuals of our training model to see if there were any patterns. The residuals do show a slight positive correlation. This trend could be because there are not many teams that have super low win ratios in a season, which provides justification for the gap of lacking values in the upper left of the residual plot that makes the graphic look a bit linear. However, we believe that our model demonstrates how close our predictions were to the win loss ratio based on the features chosen. This can be seen in the jointplot to the left where the predicted and real win ratios showcase a linear relationship close to the identity line (y=x).

Therefore, we felt as though this training error was sufficient for us to warrant running our model on the test data. The resulting test RMSE error was 9.06 which meant that our predictions for the test set --- seasons 2017-2018-- were about 9.06% away from the actual win loss ratio of the team, which in our opinion demonstrates a good model since predicting the outcome of a new NBA season can have many other non-statistics impacts that could majorly impact the teams outcomes. It is also interesting and informative to point out that the CV error we calculated is very similar to the test error. Since the test error is so low and the training cross validation error is minimized, this demonstrates that the best features from NBA statistics that can predict a team's overall success in a league, specifically its win/loss ratio are Offensive Score, Defensive Score, PER Score, Total Fouls, Total Steals and Total Starter Minutes.

Further Interest:

If you recall our second question of interest, we wanted to see if our predicted win ratio could help us determine the standings of future seasons. To test this, we decided we would need to use the Ridge Regression model we just fitted with the 2012-2017 to once again predict the win ratios for the 2017-2018 season and then use these predictions to create a mock ranking lineup. To do this we created the function rank_predictor_conference() which takes in the grouped data from a respective NBA season and the conference of the teams and outputs a dataframe showing the predicted rank versus the real rank by taking our Ridge model to predict win ratios and rank the teams by win ratio.

Our rank predictor function did well in predicting the ranks of NBA teams in the seasons 2017-2018. Below are some of the examples of the top 8 predicted teams in each conference:

2017-2018 Western Conference Rankings

	Predicted Win Ratio	Real Win Ratio	Team	Real Rank	Predicted Rank
65	80.469851	79.268293	HOU	1	1
113	72.830775	58.536585	NO	4	2
107	67.964244	57.317073	MIN	7	3
125	60.767197	58.536585	OKC	4	4
47	60.144196	56.097561	DEN	9	5
155	59.296030	57.317073	SA	7	6
149	56.554112	59.756098	POR	3	7
59	55.674502	70.731707	GS	2	8

2017-2018 Eastern Conference Rankings

	Predicted Win Ratio	Real Win Ratio	Team	Real Rank	Predicted Rank
167	69.810263	71.951220	TOR	1	1
137	64.127361	63.414634	PHI	3	2
179	62.337290	52.439024	WAS	8	3
71	61.031669	58.536585	IND	5	4
23	58.922606	43.902439	CHA	10	5
17	56.531006	67.073171	BOS	2	6
95	52.850312	53.658537	MIA	6	7
53	52.547867	47.560976	DET	9	8

As you can see from the rank predictions tables above, our predictions are quite accurate, as we are usually able to predict the top actual performing teams in the league within +/- 3 ranks. In fact, we are able to predict pretty well the teams that would make the postseason tournaments as our model predicts most of the top 8 real ranked teams in each conference to actually be in the top 8 predicted ranks. However, there are some anomalies such as the Charlotte Hornets in the Eastern conference, as despite being one of the worst teams in the conference they are predicted to do really well. This could be due to other factors our model can't predict, such as team chemistry or coaching. Overall, we believe that the model we created using the features we chose did a good job at reducing the rmse test error and predicting relatively good rankings for an unseen season.

Discussion:

Below we have answered the seven questions required:

i) One of the most interesting features we encountered was John Hollinger's widely respected basketball metric, Player Efficiency Rating (PER). Upon finding PER, we were immediately intrigued by the complexity of his formula and the amount of industry wide respect/usage of it, so we decided to utilize it. While we did use a minimized version of the PER formula, it is clear that PER holds quite a significance in relation to the win ratio of a team, as noted by the very linear relationship in the scatterplot we included above and the tremendous reduction in our model's CV error. It is interesting how simple statistics placed in a certain way (with certain weights) can actually have such an impact on our model. Another interesting feature we came across was our Defensive Score. Since PER is heavily offensive, we wanted to make sure we would be able to take into account good defensive teams such as Milwaukee and Toronto in our model and finally realized that a combined score of different defensive statistics would be our best method. In essence, by adding in these defensive scores, it allowed for our model to account for changes in team win ratios that solely offensive statistics couldn't explain. Furthermore, we thought that the total starter minutes feature we created was really interesting as it was something that we thought was very indicative of a team success, but not talked about at all in external research articles or blogs.

ii) BMI was the feature we thought would most benefit our model but turned out not to work at all. We believed categorizing athletes into ideal body shape categories would help us gauge the amount of each weight type on each team which would represent the overall conditioning of that team. You would expect a team that is very physically fit to outperform a team that consists of a bunch of underweight and overweight players. However, this feature increased both our cross validation and RMSE error so it was not used. In addition, we also thought our Division (DIV) feature would benefit our model because some divisions in the NBA have higher ranked players (e.g. all-stars) which would lead to those teams winning more games. However, we once again decided not to use it because it increased both errors substantially. Both of these features seem to have had logical reasoning to be correlated to a team's win ratio; however, they were computationally proven to not be as effective in our model.

iii) There were many challenges that we encountered while exploring the data. The biggest challenge was finding effective features that weren't based on team points. Since, besides statistics that included team points, most team statistics didn't have a high correlation with a team scoring a higher number of points. An example of this is the FIC (Floor Impact Counter) score, as we initially wanted to use it because of its high correlation with team points, but we eventually realized that it included team points in its formula. Moreover, statistics vary so greatly with each game that it was difficult to find such

features, which is where we got stuck. To find a solution to this problem, we did some research, which was when we found articles that helped us discover statistics such as PER. Another challenge we faced with the data was the creation of these multi-statistic features (i.e Offensive score) as using just the given statistics themselves didn't reduce the RMSE or CV error. In fact, we had to get creative, which resulted in the BMI and team starters minutes features.

iv) There were some limitations of our analysis that need to be addressed. One of the primary limitations is that our model is more reliant on offense than defense. Although we have a defensive score, there are not as many statistics that accurately represent defense as there are for offense, so we recognize that our model isn't accounting for defense as much as it should be. Another one of the limitations was that teams with one really good player that carries the entire team are ranked worse in our model than they actually are, because all of our features are based on the sum of player stats over the entire team. For example, Damian Lillard is the star of the Portland Trailblazers and has usually been considered to have little help alongside him besides for CJ McCollum but is able to lead Portland to yearly success. Our model in the 2017-2018 season predicts Portland as the 8th best team when in reality they were 3rd. One final over-arching limitation is that the model doesn't account for non-statistical and non-quantitative elements such as teamwork, synergy, and coaching, which are really important to a team and are big factors in determining how well they do in a season.

v) An ethical dilemma we faced with this data was with the weights we assigned each feature in our offensive and defensive scores. We based these weights on our own basketball knowledge rather than an accredited source, which can lead to some forms of bias. This ethical dilemma is similar in the BMI categories we created to account for athletes having a naturally higher BMI value, as it was based on our own assumptions. Another dilemma we faced was in the inclusion of PER. We didn't use John Hollinger's full complex model but rather a simplified version that accounted for the statistics we had. The entire formula requires intensive statistics and calculations that aren't readily available in our given datasets; therefore, we used a linear model created by a famous sports analyzation center, Bleacher Report, to accurately calculate a team players PER based on optimized weighting and certain statistics that we had access to. However, this could potentially negatively affect the true potential of our model as our PER values aren't the exact calculations that John Hollinger invented.

vi) Some examples of additional data that would strengthen our analysis are more defensive data, player substitution patterns, and NBA combine scores. Our model currently focuses heavily on the offensive side of the game but we believe it would benefit from the inclusion of defensive statistics such as Defensive Win Shares which is a measure of a player's defensive value, 3 point defending percentage and average points allowed in the last 2 minutes of a quarter. Additionally, our data would be better off with the inclusion of information regarding player substitutions so we could gain a sense of which players are on the court at the same time and if there is an ideal lineup for each team. NBA Combine statistics such as a player's vertical leap, wingspan and three quarter sprint could be ideal features to determine a player's ability to impact a game, as they are partially drafted to a NBA team based on the scores given in the Combine.

vii) Some ethical concerns we might encounter in studying this problem include not understanding the atmosphere around each team during the season and how this may impact them. For example, in the 2012-2013 season, the city of Boston was devastated by the Boston Marathon which definitely impacted the team in the long run and we are unable to account for this. As mentioned before, team chemistry plays a huge part in a team's success, and it is a concern that we are also unable to account for the team's chemistry in this dataset. For example, Kevin Durant's impending free agent decision in 2015-2016 could have affected the overall success of the Oklahoma City Thunder as certain players may not have trusted him. In essence, our model emphasizes the bottom lines results of a player or team, which doesn't provide context details that are just as important in a team's win ratio in a season. We could address this problem by creating another feature in our model that places a cumulative score each season on the external issues in the area or region and internal issues within the team. The issue with this feature is that it is nearly impossible to place a value to such a metric (could possibly use categorical variables) and it would require intensive research into the markets and social environments for each team every season.

Some future applications of this research and model could be predicting the outcome of future seasons down the line such as 2021 and beyond, as using our model we can potentially predict the postseason results based on regular season data. Also, we could consider including playoff data into our training and test set that could potentially help us predict the outcome of the playoffs for each season as well.

Works Cited

[1] Fein, Zach. “Cracking the Code: How to Calculate Hollinger's PER Without All the Mess.” Bleacher Report, Bleacher Report, 3 Oct. 2017, bleacherreport.com/articles/113144-cracking-the-code-how-to-calculate-hollingers-per-without-all-the-mess.

[2] Mikołajec, Kazimierz et al. “Game Indicators Determining Sports Performance in the NBA.” Journal of human kinetics vol. 37 145-51. 5 Jul. 2013, doi:10.2478/hukin-2013-0035