Math 189 - Final Project

NCAA March Madness

March 25th, 2019

Name	PID
Erin Werner	A12612584
Emma Choi	A12635909
Talal Alqadi	A13816618
Ella Lucas	A13557332
Samantha De La Torre	A13300273

1.) Introduction

Google Cloud and NCAA have collaborated to present a Kaggle competition with the objective of utilizing computational methods to predict the outcome of the incoming 2019 March Madness Men's and Women's Basketball Championships. This challenge is attempted by aspiring computer scientists, statisticians, and data scientists every year, and has been historically hosted by Kaggle for the past six. The data displays the results of past March Madness tournaments and is used to train and test models. The NCAA has migrated 80+ years of historical and play-by-play data from 90 championships and 24 sports to Google Cloud Platform. Decades of basketball data has been extracted using BigQuery, Cloud Spanner, Datalab, and Cloud Machine Learning. This empowers analysis of team and player performance metrics. According to the Kaggle competition's guidelines, individuals are welcome to bring in external data if it helps with modeling. The second part of the competition is based on predicting the outcomes of the games. Once the basketball tournament starts, competitors will be able to determine how effective their models were at predicting the competitions outcome by considering the results in real time as the games are played.

As stated by Google Cloud, the objective of hosting the computing competition on Kaggle is to challenge participants to strengthen their knowledge of basketball, statistics, data modeling, and cloud technology.

Submissions to the competition will be scored based on the log loss, which punishes model outputs that are confidently wrong.

The following report contains an attempt at the Kaggle competition. For our purposes we will be utilizing the data resources on the women's tournament. We will use several statistical methods to test and train a model on the NCAA historical tournament results dataset. We will then predict the results of the incoming 2019 March Madness tournament and measure the accuracy of our findings. All of the investigations contained in this report will be explained in terms of the rationale behind their implementation as well as the theory that drives the computation.

2.) Data

Each season there are thousands of NCAA® basketball games played between Division I women's teams, culminating in March Madness, the 64-team national championship that starts in the middle of March. The Kaggle competition provided extensive historical data to jump-start the modeling process, and this data is self-consistent (for instance, dates and team ID's are always treated the same way). All the data files provided are represented in Table 2.1, along with a description of the data in each file. Collectively, this data will allow us to build our predictive model. Our goal is to explore the data and develop our own distinctive ways of predicting March Madness® game outcomes.

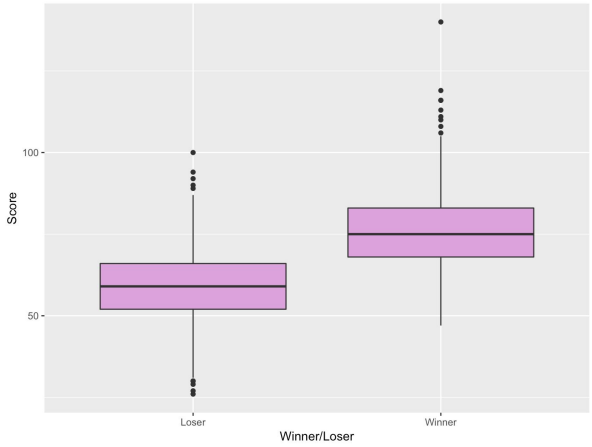
Table 2.1. Table with all the data file descriptions.

File (*.csv)	Description
WCities	This file provides a master list of cities that have been locations for games played.
WGameCities	This file identifies all games, starting with the 2010 season, along with the city that the game was played in.
WNCAATourneyCompactResults	This file identifies the game-by-game NCAA® tournament results for all seasons of historical data.
WNCAATourneyDetailedResults	This file provides team-level box scores for many NCAA® tournaments, starting with the 2010 season.
WNCAATourneySeeds	This file identifies the seeds for all teams in each NCAA® tournament, for all seasons of historical data.
WNCAATourneySlots	This file identifies the mechanism by which teams are paired against each other, depending upon their seeds, as the tournament proceeds.
WRegularSeasonCompactResults	This file identifies the game-by-game results for many seasons of historical data, starting with the 1998 season.
WRegularSeasonDetailedResults	This file provides team-level box scores for many regular seasons of historical data, starting with the 2010 season.
WSeasons	This file identifies the different seasons included in the historical data, along with certain season-level properties.
WTeams	This file identifies the different college teams present in the dataset.
WTeamSpellings	This file indicates alternative spellings of many team names.

The files that we will primarily focused on are WNCAATourneyCompactResults.csv, WNCAATourneySeeds.csv, and WRegularSeasonCompactResults.csv. These files have important information in regards to the teams scores, seeds, and status of winning or losing.

An important aspect in building a model to predict future winning and losing teams will be their past scores and status of winning and losing. This information is kept in WNCAATourneyCompactResults.csv and WRegularSeasonCompactResults.csv. These two files are essentially the same, but the data is in regards to stats from the past March Madness tournaments compared to the past regular seasons. The history of a teams performance is indicative of their future performance. If a team has performed well consistently in the past, it is fair to assume they will continue to do so. Of course, upsets can occur, but historical performance will be an important factor for building our model. So, it is important to have an understanding of the data in relation to features, such as score. Figure 2.1, below, shows the distribution of winning and losing scores during the tournament.

Figure 2.1. Box plot distribution of tournament scores.



Obviously, the winning scores are higher than the losing scores. But, both boxplots have similar distributions, just around their respective means. The losing scores have outliers outside of the IQR range, both above and below, where the winning scores only have outliers above the IQR range. This box plot demonstrates the general distribution of the score data that we can use to build our model, based on previous team performance in the tournament. Furthermore, we also have the score data for the regular season as well.

Figure 2.2. Density plot of winning scores.

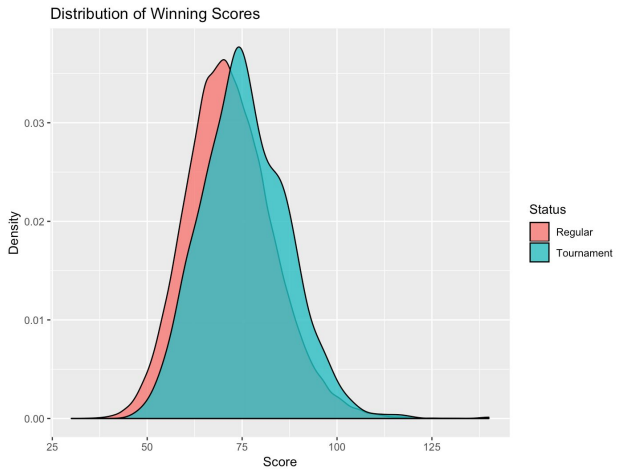


Figure 2.3. Density plot of loosing scores.

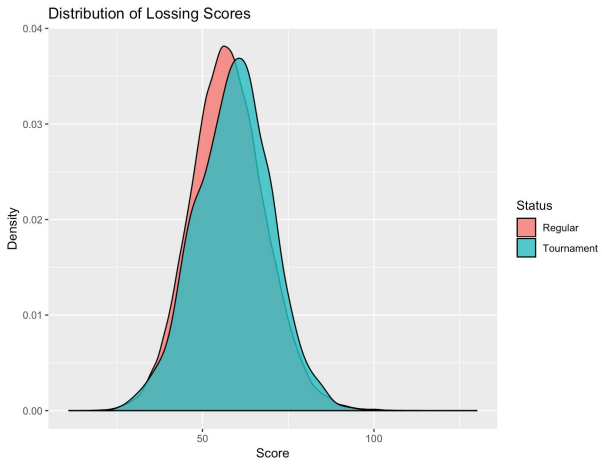


Figure 2.2 and 2.3 demonstrate that the teams perform roughly the same, but slightly better, during the tournament compared to the regular season. Both density plots are unimodal, with no major skew or tail. The distributions are centered around their respective means, where the regular season is close, but a little bit lower compared to the tournament. These plots reveal that there isn't any major difference in score performance in the tournament and the regular season. This is an important aspect in considering what investigations to conduct as well as how

to handle all of our data. Thus, these plots help to demonstrate the general score distribution that we can expect to be working with and building our model around.

Figure 2.4. Scatter plot of scores per team, with respect to the winning and losing team.

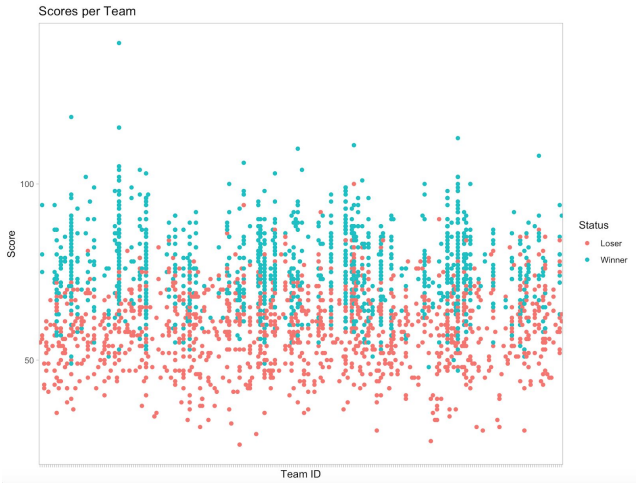


Figure 2.4 reveals the distribution of scores per team and differentiates whether that team/score won or lost. Although it is difficult to find patterns as there are so many teams in our data set, certain teams have a large amount of blue dots that almost make up a solid vertical line, indicating that the team won a significant amount of games. There are about ten teams with this behavior. As a result, it is likely that those teams with a lot of success in the past will have more success in the future. However, there doesn't seem to be a similar behavior with teams and consistent losing scores.

Figure 2.5. Histogram of seed differences (winning seed - losing seed).

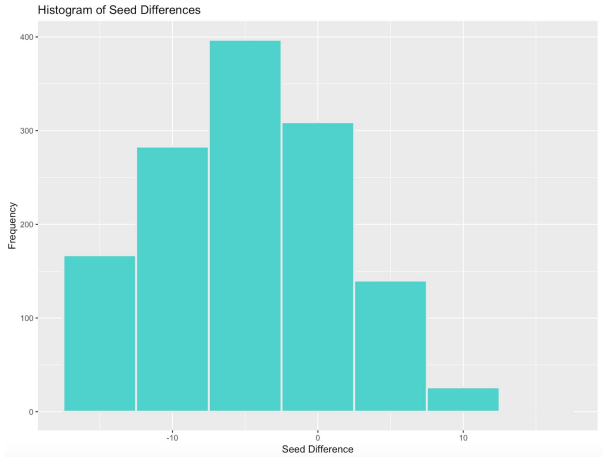


Figure 2.5 shows the distribution of seed differences between the winning and losing teams in the tournament. The distribution is unimodal with a peak around the mean, -4.5. This

means that the winning teams are generally 4.5 seeds higher than the losing team. The distribution is not quite symmetric as there are more negative seed differences than positive ones. This means that it is more common for winning teams to have lower seeds (i.e. seeds 1, 2, or 3) where their opponents have higher seeds (i.e. seeds 14, 15, or 16). Upsets can obviously occur, so these are influential factors to consider in our investigations.

There are many features included in all these files, but these plots provide a good, general understanding of the data that we will be working with in order to build a predictive model for the March Madness tournament.

3.) Background

The first ever NCAA March Madness tournament was held in 1939, with only eight Men's teams participating. NCAA welcomed the women's teams to the tournament after adopting the Women's Division I National Championships from the Association for Intercollegiate Athletics for Women in 1982. In its early years, the championship was dominated by smaller teams, such as Immaculata College. However, due to its increased popularity and the expansion of women's sports, March Madness is now frequently won by larger schools, such as the University of Tennessee and the University of Connecticut. While the first tournament had only 32 teams compete, the tournament has expanded to its present day format with 64 teams entering the battle.

The tournament is composed of four regions, each of which contains 16 teams that compete with each other to win their region and advance to the final four. The final four is a title given to the four regional champions participating in the semi-finals of the tournament. Upon entering the competition each team is given a seed, 1-16, in their region, based on their performance during the regular season. Seeds closer to one will be referred to as high seeds and seeds closer to 16 will be referred to as lower seeds. The bracket in each region is set up, as shown in Figure 3.1, with a one seed playing a 16 seed, a two seed playing a 15 seed, and so on, in the first round.

Figure 3.1. March Madness bracket.

The seed number of the team serves as a predictor for which place they should take in the region's portion of the bracket. For example, if a team comes in as a three seed in the West Region, they are predicted to take third place in the West Region. In this format, teams that come in with higher seeds (i.e. seeds 1, 2, or 3) are supposed to place well because they start out playing teams seeded below them (i.e. seeds 14, 15, or 16). However, upsets, where a high seeded team loses to a lower seeded team, occur every season, which is why March Madness has become a cult phenomenon in today's culture.

The national championship has become a major social event, with people copiously studying the NCAA basketball season and preparing, what they hope to be an accurate bracket to enter in office pools. Office pool is a term used to describe a group of March Madness brackets, submitted by different people, competing for the most accurate prediction of the tournament outcomes. Due to the popularity of the tournament, office pools have become increasingly common. While some people bet the traditional stakes on their office pool, money, many people are involved just for the social interaction and do not bet anything. A perfect bracket, that is, a bracket that correctly predicts the winners and matchups at every stage of the March Madness tournament, has never been created. The prevalence of the tournament along with the lack of a perfect bracket has led many to attempt to create a way to predict the game outcomes and matchups, but no one has succeeded yet. In this project we will use data from previous Women's NCAA seasons and March Madness tournaments to attempt to predict the outcomes of the 2019 Women's March Madness Tournament.

4.) Investigations

a.) Logistic Regression

Logistic regression is useful as it results in a binary classification output. In this case, our predictions will be either win or lose, making logistic regression an ideal model. We can treat winning and losing as success and failures arising from a binomial distribution where the probability of a success is given by a transformation of a linear model of the predictors.

i.) Seed Based Logistic Regression

By using just the seeding to predict the winner and confidence, this logistic regression model will act as our baseline. Logistic regression is a GLM used to model a binary categorical variable using numerical and categorical predictors. We assume a binomial distribution produced the outcome variable and we therefore want to model p the probability of success for a given set of predictors. For this seed based logistic regression, we will merge the two files, WNCAATourneyCompactResults.csv and WNCAATourneySeeds.csv, in order to form a new data frame that includes all the data in regards to a winning and losing teams and their respective

seeds. Once, the data is combined, we can calculate the seed differences. Seeds, in relation to basketball, often indicate a team's rank. So, higher seeds are assigned to teams that perform well and lower seeds are assigned to teams that do not perform as well. These seeds then form the bracket and it is typically intended so that the best teams do not meet until later in the competition. As a result, there are many seed pairings with large differences. Thus, seed differences can serve as a good predictor for winning and losing outcomes.

Logistic regression can serve as a model to predict these outcomes based off of team seed differences. The results are shown in Figure 4.1.1. Simple interpretation is only possible in terms of log odds and log odds ratios for intercept and slope terms. The intercept represents the log odds of winning for a team with a seed difference of -15. From this we can calculate the odds or probability, but additional calculations are necessary. The slope of the line demonstrates that, for a unit increase in seed difference (being 1 seed closer together), how much the log odds ratio will change, which is not particularly intuitive. More often than not, we care only about sign and relative magnitude. So, we check the residuals of our results, seen in Figure 4.1.2.

Figure 4.1.1. Logistic regression plot.

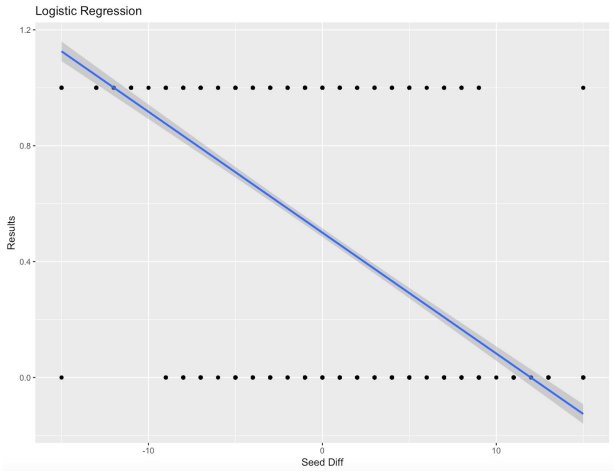
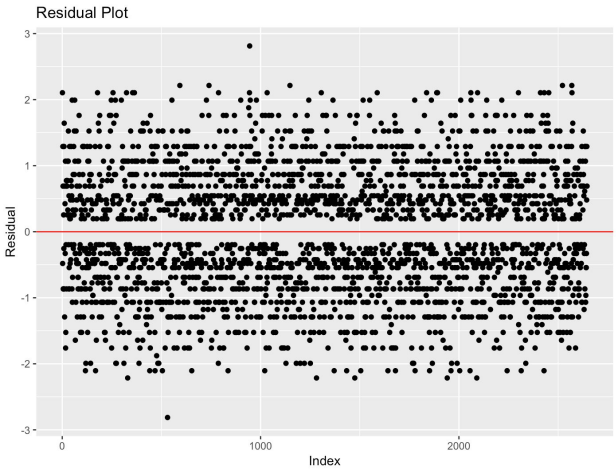


Figure 4.1.2. Residual plot.



Residual plots are useful in checking the assumptions of a regression model. Residual plots display the residual values on the y-axis and fitted values, or another variable, on the x-axis. After you fit a regression model, it is crucial to check the residual plots. There are two fundamental parts to regression models, the deterministic components and the random components. The deterministic component is the portion of the variation in the dependent variable that the independent variables explain. Stochastic just means unpredictable. In statistics, the error is the difference between the expected value and the observed value. Ultimately, we need to determine whether the residuals are consistent with random error. Although there is a gap around the x-axis in Figure 4.1.2, the data is still random and centered around the zero slope line. The residuals also appear to follow a normal distribution. As a result, the data does not appear to be heteroskedastic and there is variability of points around the least square line.

Figure 4.1.3. Histogram of residuals.

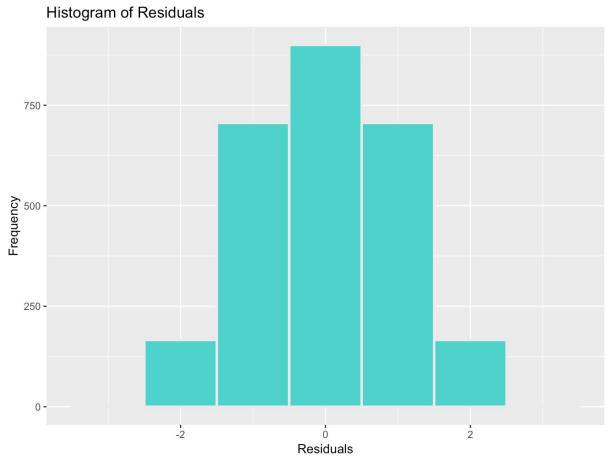
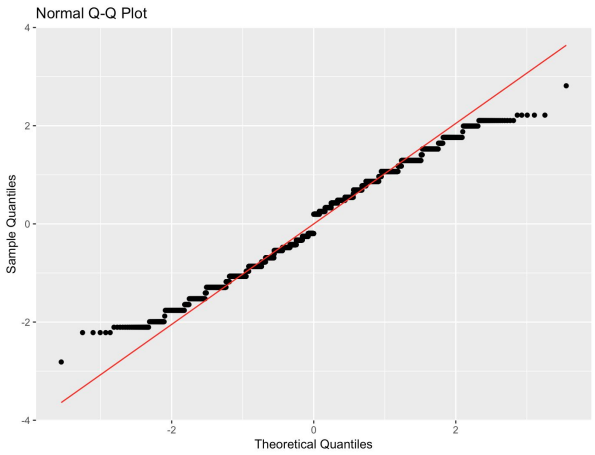


Figure 4.1.4. Q-Q plot of residuals.



Furthermore, Figure 4.1.3 reveals a histogram of the residuals that is symmetric with no major skew. This reinforces the notion that the residual data from our logistic regression does, in fact, follow a normal distribution. Both Figure 4.1.2 and Figure 4.1.4 reveal that the data does actually fit the normal line (represented in red) pretty well. This holds the condition of normality of residuals of the regression.

By checking for the linearity between the response and explanatory variables, for the normality of residuals of the regression, and the variability of points around the least square line, we are checking the fit of the logistic regression. All of these conditions have been met, as demonstrated by the plots above, so we can trust the assumptions of our logistic regression model and use in to make predictions for the winners and losers of the tournament.

Figure 4.1.5. Prediction probability given seed differences.

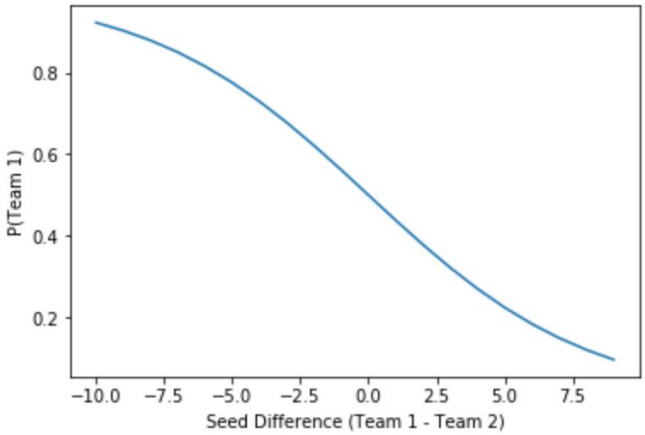


Figure 4.1.5. reveals the probability curve in response to team seed differences. So, given a seed difference of Team 1 - Team 2, it is more likely that Team 1 will win if the seed

difference has a larger negative value. This means that Team 1 has a high seed, indicating that they perform well, and Team 2 has a low seed. As a result, it would be more likely for Team 1 to win in this situation. Additionally, the roles are reversed is the seed difference is a larger positive value. If the two teams are in the same seed, the plot reveals that chance of either team winning is about 50%. This all makes sense with how seeding is supposed to work in forming the bracket for the March Madness tournament.

ii.) Team Performance Based Logistic Regression

In order to try and find another way to predict the outcomes of the March Madness tournament, the winning and losing scores of previous games were also fit into a logistic regression model. The results are plotted as a scatterplot in Figure 4.1.6. It represents the entire set of score explanatory variables with the status of winning or losing, as well as its resulting fitted least squares line. The values at y = 0 represent scores that lost the game and values at y = 1 represent scores that won the game. These past scores can serve to predict how future scores will result in winning or losing. Once again, Figures 4.1.7, 4.1.8, and 4.1.9 verify the fit of the logistic regression. This is done by checking for the linearity between the response and explanatory variables, the normality of residuals of the regression, and the variability of points around the least squares line. If all of these qualities are met by the results of the data, it is fair to assume that the regression model is to be trusted. Then, there is greater confidence in the predictions that are generated as a result.

Figure 4.1.6. Logistic regression plot.

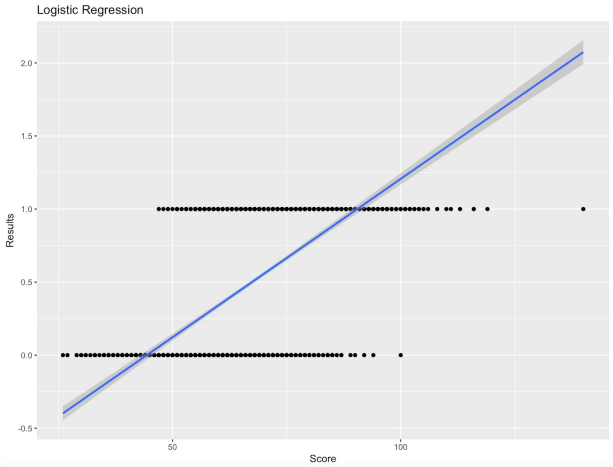
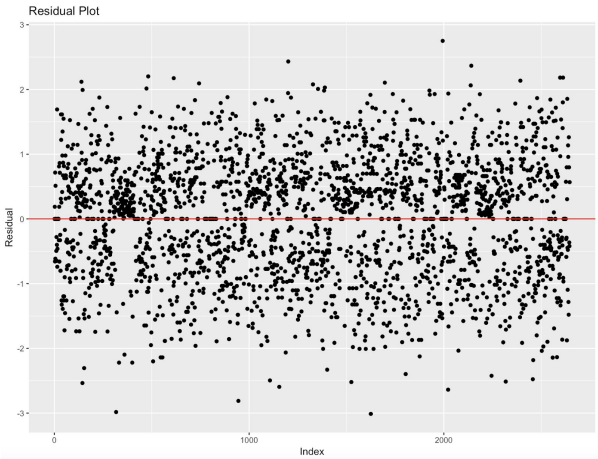


Figure 4.1.7. Residual plot.



First, Figure 4.1.6 shows that the first condition is satisfied, as the score data fits a logistic model. Figure 4.1.8 reveals that the residuals follow the normal distribution because the histogram is mostly symmetric with no major skew. Also, Figure 4.1.9 shows that the Q-Q Plot almost completely follows the normal line. So, it is fair to assume that the second condition is

followed. As there is no strong relationship between the explanatory variables on the x-axis and the residuals seen in Figure 4.1.7, condition three, concerning homoscedasticity, is satisfied.

Figure 4.1.8. Histogram of residuals.

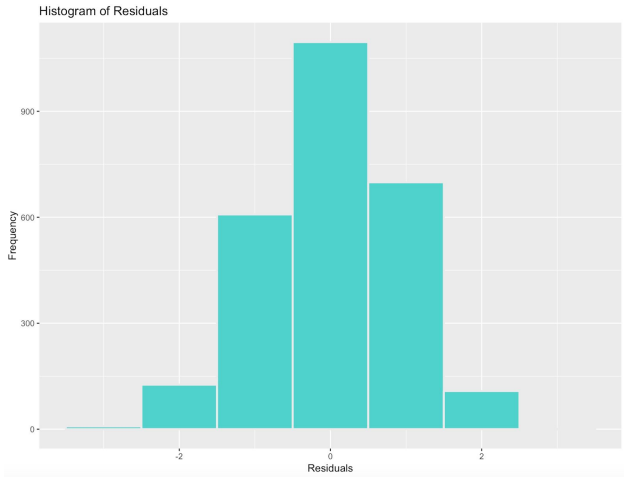
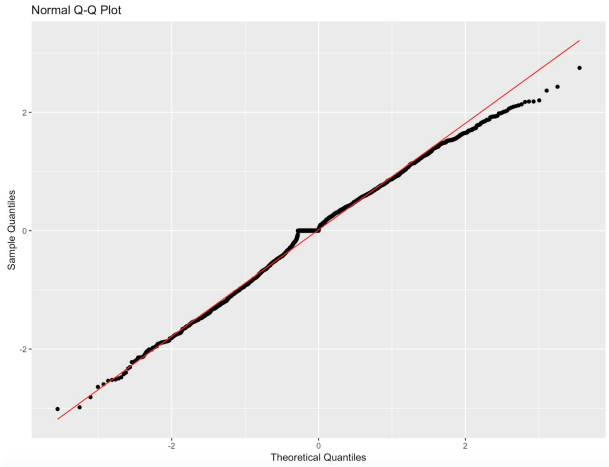


Figure 4.1.9. Q-Q plot of residuals.



Therefore, a fitted logistic regression model can also be applied to the relationship between team's scores and the status of winning and losing during the tournament.

Figure 4.1.10. Prediction probability given team history of winning and losing.

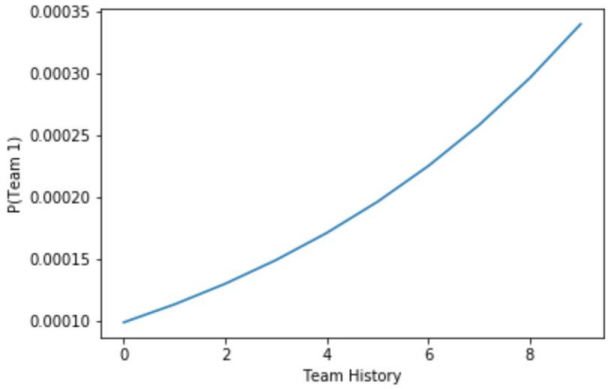


Figure 4.1.10 reveals the probability curve in response to previous team scores during games. So, given a team's history, it is more likely that they will win more games in the future if they have won more games in the past. This means that a team that has won 0 games has a low probability of winning future games. But, the probability of winning increases, almost exponentially, if the team has won more games. As a result, it will be more likely for teams that have won consistently in the past to win in the future. Additionally, the roles are reversed for teams that regularly lose. This makes sense as past performance in the tournament is generally a good indicator for future performance.

b.) Decision Tree

Decision trees build regression or classification models by breaking down the dataset into incrementally smaller subsets. It is well-suited for both categorical and numerical data, making it a great application to our data which contains mixed predictors. Furthermore, decision trees effectively classify subjects into known groups. The resulting formation both conceptually and physically resembles that of an actual March Madness bracket, and so it logically follows to run decision tree on the data. Computationally, decision trees can handle huge datasets with missing data and redundant variables, which means we can make use of the data provided without concern of restraints.

i.) Classification Tree (Classifying a Winner)

To predict the winner of the NCAA women's tournament, we will run a decision classifier on the data. This method will be implemented in Python using the standard packages sklearn, NumPy, and Pandas. In Python, sklearn is a machine learning package that contains modules that allow us to run a decision tree (train_test_split, DecisionTreeClassifier, accuracy_score) capable of making predictions and analyzing predictive accuracy.

The data used for the classification tree was 'WNCAATourneySeeds' and 'WNCAATourneyCompactResults'. The goal is to join the seeds and results, and have the trained model predict the outcome of a game based on the pairing of 'TeamID' and 'seed'. Data cleaning and preprocessing included removing the regional abbreviation above the seed, dropping columns that would distort the classification, namely 'DayNum', 'WScore', 'LScore', 'WLoc', 'NumOT'. Next, we merged seeds with their corresponding team ID broken into win and loss seeds. So that the decision classifier could understand wins and losses comparatively, a column was added to the dataframe that computed the difference between winning and losing seeds, see Figure 2.5 for a visual representation. The decision classifier makes predictions based on wins and losses for a given team.

In the implementation of the model, we divided our predictive data into training and test sets. The breakdown was 70:30, meaning we trained the model on 70% of the data and tested its accuracy with the remaining 30%. Next, we created a decision tree classifier object, trained it, and predicted the response for test data. The resulting matrix returned '0' for a loss and '1' for a win for the games it was tested on, based on the seed for that TeamID.

We used the 'accuracy_score' package to compute the accuracy of our model results. Our decision classifier has a classification rate of 75.94%, which is not bad considering that we haven't fine tuned our model. This rate could be improved by tuning parameters in the Decision Tree Algorithm.

ii.) Regression Tree (# Wins)

We pulled data from Team Box Scores displaying the average field goals made as a quantifying factor influencing the number of wins. This regression classifier was implemented in R using the provided packages for RPART to output a quantifiable judgement of how many wins will result from the tournament based on average values of all the attributes. Field goal terminology refers to any occurrence of a scoring point, non-distinguished between two or three point shots. It is the start of the regression tree, and the most occuring variable throughout the tree. More field goals is equivalent to more wins, and these variables are easily modeled in the regression tree below due to their ability to be weighed against one another in a bracket-like structure. The regression tree is essentially ordered from least predicted wins to most predicted wins. With n = 130 and wins = 26.86, the tree follows Opp_FGM >= 23.16 (rather than >=23.23), then FGM < 26.63, and finally, Opp_Astmean < 10.87. To interpret this, with stats equivalent to the previous conditions stated, 130 teams had an average of wins = 26.86. With max wins for a single team being 34, different from the max wins in decision tree (26.86), it is understandable because of the aggregated effect of all the other variables. The regression tree shows that with 4 variables being assessed we can reach an average of 26.86 wins.

We chose to use a regression model for the same fundamental reasons that decision trees are a good fit, but the method of regression compared to that of classification has its own rationale. The regression model is easily visualized and its structure follows very intuitively that of a March Madness bracket. Prediction trees are nonlinear predictive models which makes them a sensible fit to our nonlinear data. Since the computational process involved comes down to making fast predictions done by looking up constants in the tree, it is a quickly generated model that avoids the time consuming process of complex calculations. Since we were able to quickly distinguish the variables we want to incorporate to make the prediction of number of wins (field goals), the regression tree is a nice choice to visually model their effects.

The regression tree algorithm predicts the number of wins for n = 3118 with the value FGMmean < 23.56. We ran a cross-validation test on the model to understand its performance. The results are depicted below in Figure 4.2.2 showing r-squared and x relative error improving as the number of splits on the data increased.

Figure 4.2.1. Regression tree model of factors influencing the number of wins.

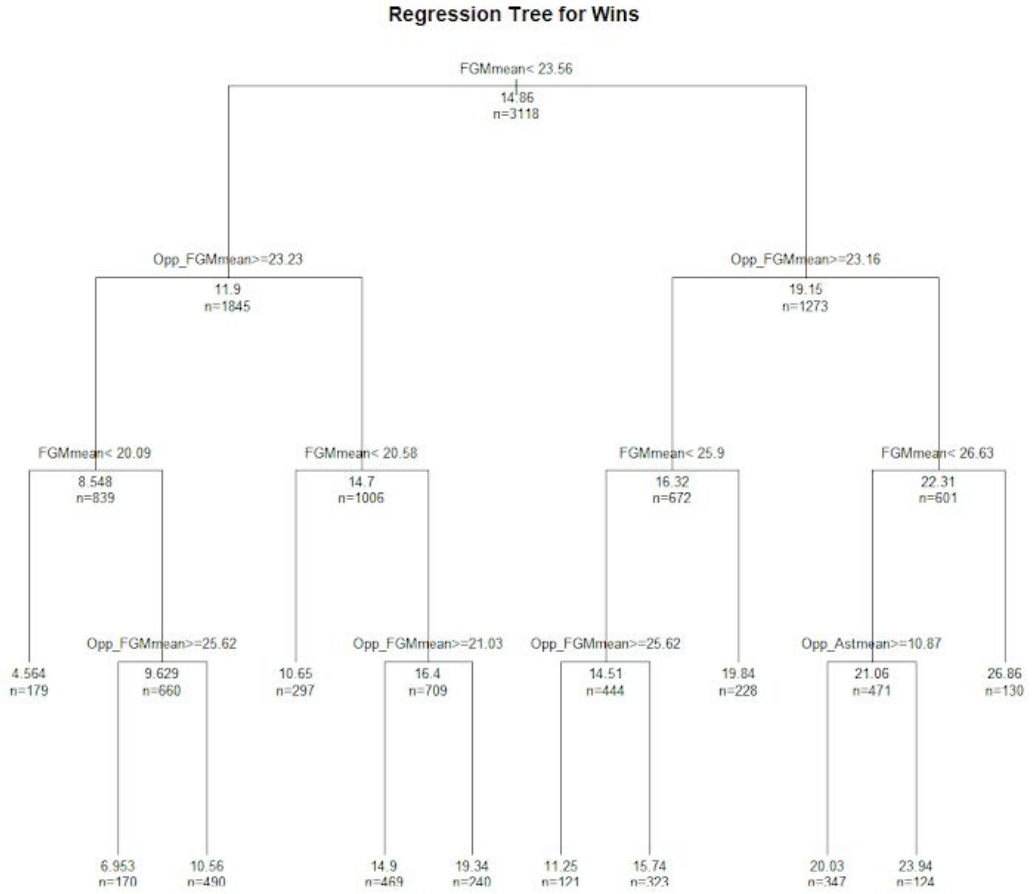
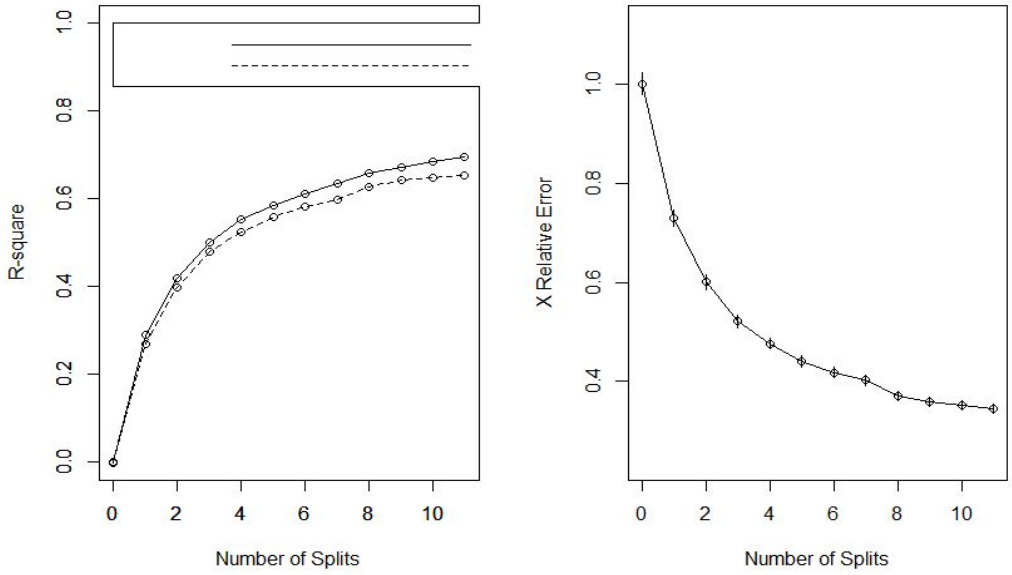


Figure 4.2.2. Cross validation results: displays R-squared and X relative error as splits increase.



c.) Random Forest

Random Forests are extremely helpful for non-linear multivariate classification. Since there are many variables and effects in this case, it is possible to apply a random forest training model with a target on wins versus all other applicable variables and effects. First, the dataset was created to contain the mean of all the stats for each team across all 3118 teams and all matches. The Random Forest model with tree size = 1000 was applied to this dataset that included all variables except for Wins (the target variable), Season, Games Played, and Win Percentage.

Figure 4.3.1. Plotted error rate.

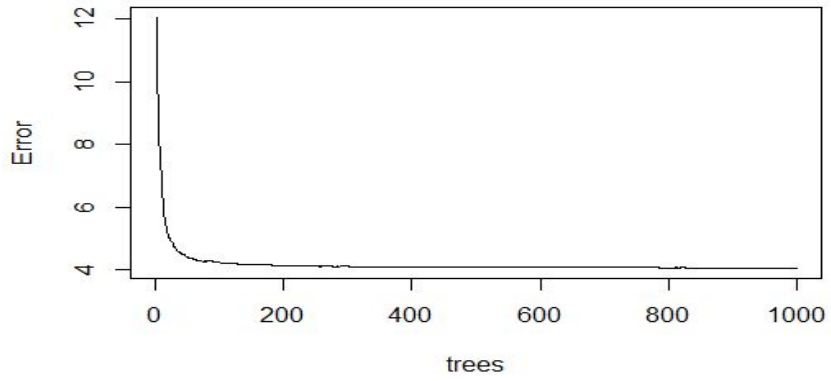


Figure 4.3.1 shows the error rate throughout the trees. It levels out around 100 trees, and continues slowly descending up until 1000 trees.

Averaging across 1000 trees, the Mean of Squared residuals = 5.44192. Looking at the quantiles of the MSE, we can see that most mean squared errors are around the 0% quantile. Thus, most trees provided an average around the lowest MSE.

0%	25%	50%	75%	100%
4.057400	4.069115	4.077407	4.108971	12.044026

Figure 4.3.2. Importance of variables.

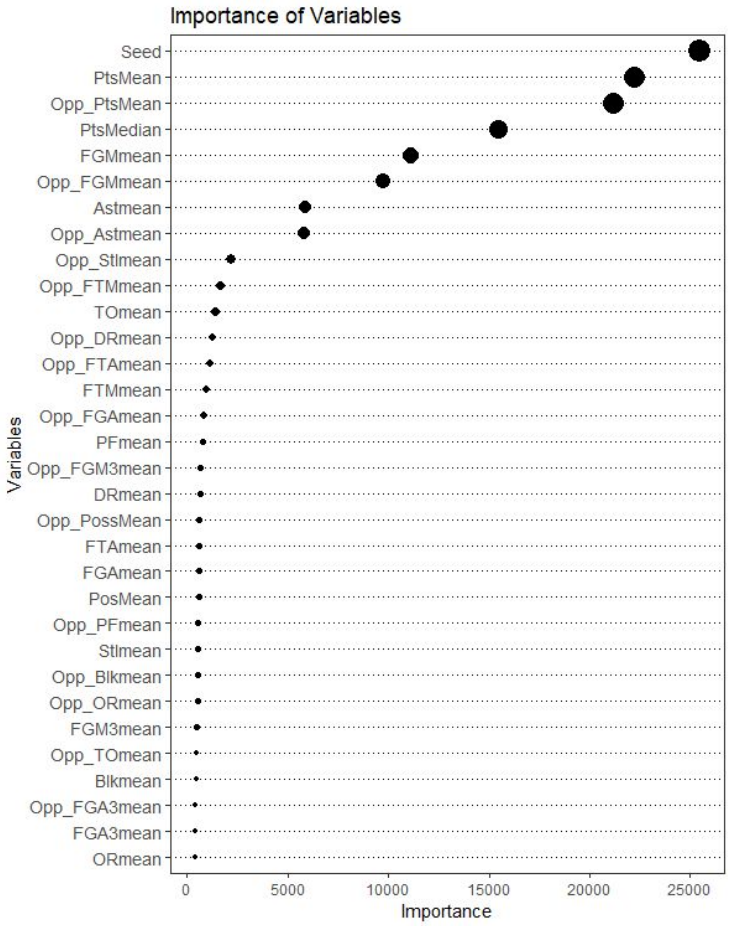


Figure 4.3.2 shows the importance of each variable as the ones with the greatest impact in reducing MSE across our 1000 trees. Top variables important to us in sequence:
Seed → Field Goals Made→ Opponent Field Goals made → Assists, all have the highest importance or node impurity. Seed is the highest due to most teams not having a seed (only 16 seeds); so, having an extremely good seed skews the wins. In addition to that, it is expected that the following 3 important variables are Point stats, as that's what decides wins in a game. Following those, both opposition and team Assists and Field Goals Made have good importance values. It is essential to note that most other variables also have a good node impurity value too, just not as strong as the ones mentioned. Furthermore, we continue this random forest model and apply it to a sample size of 600, and tree size of 1000, below.

Figure 4.3.3. Visualized mapping of RF model (target = wins, sample size = 600, trees = 1000)

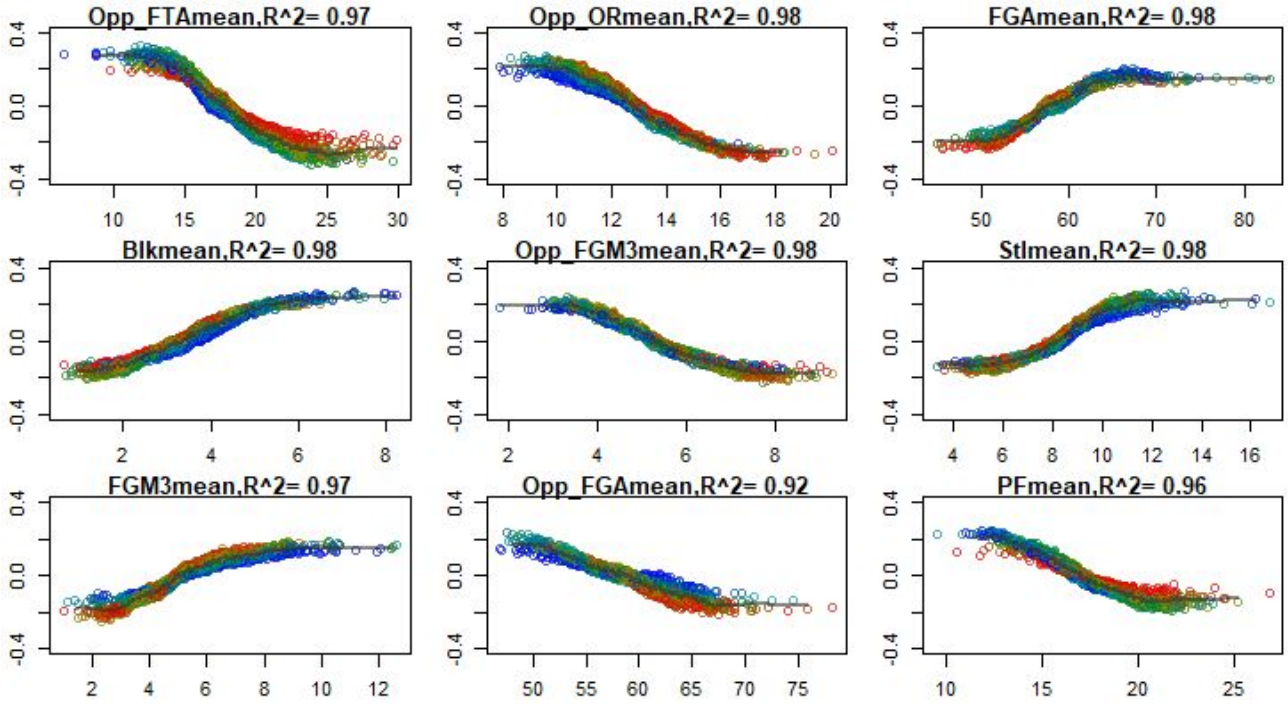


Figure 4.3.3 is a visualized mapping of the trained Random Forest model with target on wins. The top 9 variables sorted by importance are visualized individually. Each figure allows us to explore the curvature of the random forest model-fit. The Y-axis represents the predicted change of probability due to the variable value, while the X-axis is the variable value. R^2 quantifies the goodness-of-fit when visualizing the variable effect. The Color gradient shows the effect of all other variables on each individual variable. Red colored outliers near higher probability areas (that are colored in blue) will have a lower/higher predicted probability change due to an effect from other variables. For example, looking at the Personal Fouls, the red outlier with 11 personal fouls per game has a lower predicted change of probability, due to the added effect of all other variables. Thus, the more spread the points are spread out vertically, the more the other variables are subject to change the predicted probability change value at that certain x value. Looking at Opponent Free Throws attempted, decreasing the value consistently increases the predicted probability change. However, when looking at higher values of Opponent FTA, it is less consistent to a negative probability change, as other variables have a bigger effect on Opponent FTA, allowing other stats to make up for a high Opponent FTA. By looking at the slope of each figure, we can rank the effect of variables as follows:

Opp_FTA (Least consistent at lower values) > Opp_OR > Steals > Opp_FGM3 (Most consistent)

Figure 4.3.4 Correlations of Variable vs. Wins (with winner in orange) for each season

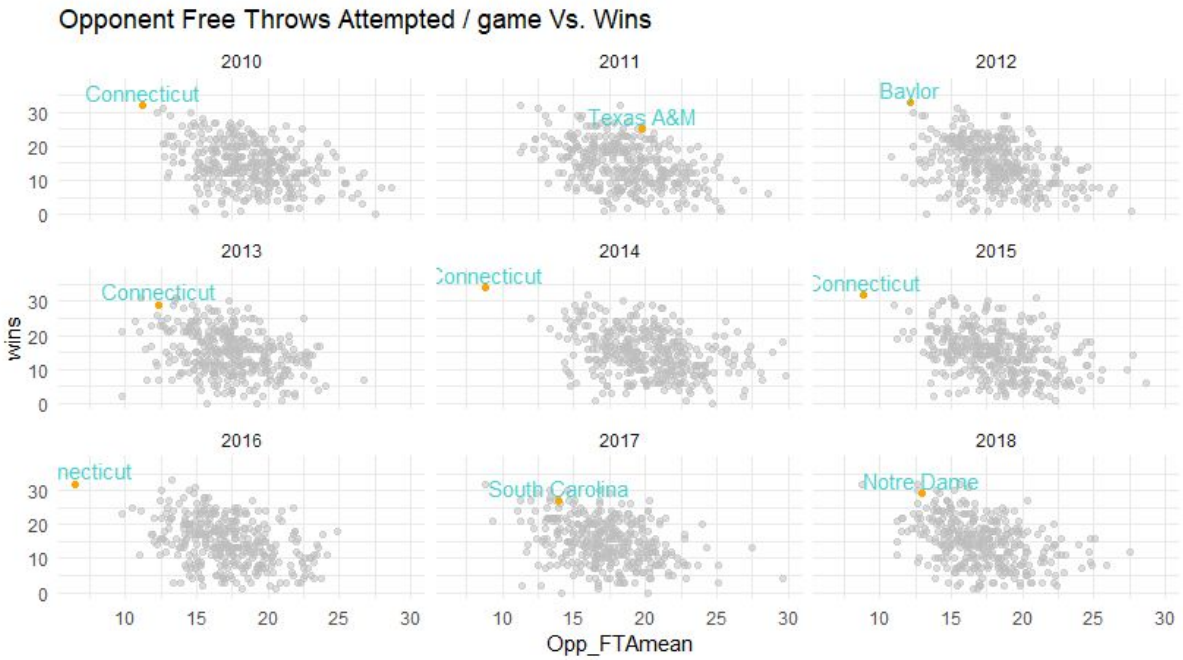


Figure 4.3.5. Correlations of opponent field goals.

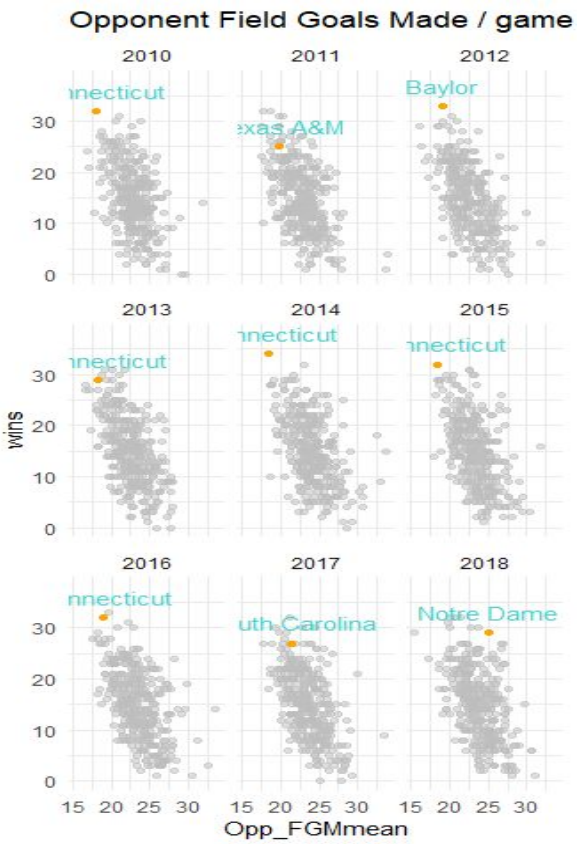
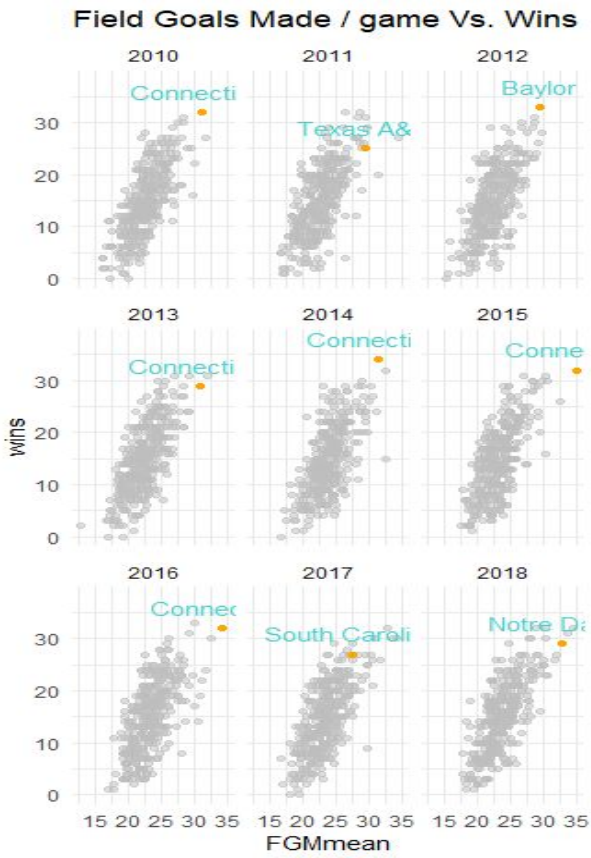


Figure 4.3.6. Correlations of field goals.



All figures allow us to observe the correlation of the three different variables versus wins of all of the seasons for each team, and compare them to the winner (the orange point). This is useful because it allows us to visualize the correlations, and understand the results of the random forest model a little better. Prior to this, the average stats for each team was calculated. Figure 4.3.4 allows us to understand why the Opponent Free Throws Attempted were rated so highly in the Random Forest model. Four of the nine season winners all had the lowest Opponent Free Throws attempted. However, the data seems a little scattered and not consistent. Figure 4.3.5 contains a strong negative correlation. However, better opponent Field goals made do not always indicate the championship or higher wins (see 2016 and 2017 Seasons). Figure 4.3.6 shows the field goals made, and all winners are always in the top 5% of field goals made from all teams. So FGM has a strong positive correlation with wins.

d.) Cross Tabulations

In order to compare final four teams from the 2010 to 2018 seasons to teams that were eliminated earlier in the tournament the data was broken into two categories: seasons with the regional championships on day 146 and 147 and seasons with the regional championships on day 147 and 148. Information pulled from the two datasets were summed to provide analysis spanning over nine seasons, from 2010 to 2018. The final four teams in each season were determined by if they won the game played on the day of the regional championships. These teams were then cross-checked with other variables to see which factors may correspond to advancing to the final four. In Table 4.4.1 the relationship between entering the tournament as a one or two seed and finishing in the final four was explored.

Table 4.4.1. Cross tabulation for seeds.

Cross Tabulation	Final Four	Not Final Four	Total
1 Seed	23	13	36
2 Seed	7	29	36
3 Seed	1	35	36
4 Seed	3	33	36
5 Seed	1	35	36
6 Seed	0	36	36
7 Seed	1	35	36
8 Seed or Lower	0	324	324
Total	36	540	576

The analysis in Table 4.4.1 shows that over the course of nine years, 83.3% of the Final Four Teams entered the tournament as a one or two seed. Of the 30 one and two seeds in the Final Four, 76.7% of them were one seeds. Additionally, 63.9% of one seeds finished in the Final Four. Of the teams that entered the tournament as a 3-16 seed 98.8% were eliminated before reaching the Final Four. Furthermore teams entering the tournament as an eight seed or lower never reached the Final Four. While one and two seeds have a higher propensity to appear in the Final Four, Figure 4.4.1 shows that one seeds win the tournament more often than two seeds.

Figure 4.4.1. Distribution of seed ranks that won the tournament.

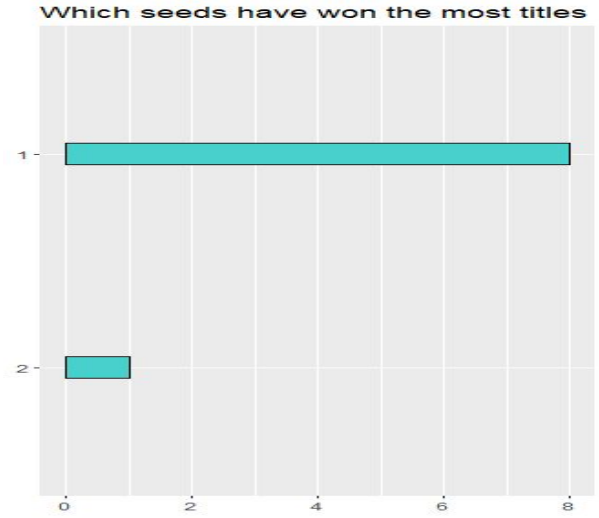


Table 4.4.2. Cross Tabulation for teams that won more than 28 games in their regular season.

Cross-Tabulation	Final Four	Not Final Four	Total
Won Over 30 games	24	45	69
Won Under 30 games	12	495	507
Total	36	540	576

In their regular season games the Women's NCAA Basketball Teams play roughly 30 games. The max amount of games a team played in a season from 2010 to 2019 was 34 games. In order to determine if there was a difference between teams that win more games in the season and their tendency to make it to the Final Four a cross-tabulation was made. As shown in Table 4.4.2, of the teams that won less than 30 games in their regular season only 2.37% of them advanced to the Final Four. Additionally, 97.6% of teams that won under 30 games were eliminated from March Madness before reaching the Final Four.

e.) Monte Carlo Simulations (Kurtosis/Skewness)

We can find out more about the distribution of our data by finding the kurtosis and skewness value of the seed differences of winners versus losers. The kurtosis value is an indicator of the tail-heaviness of the distribution. A normal distribution is known to follow a kurtosis of 3, while distributions with heavy tails will be above 3. The skewness of a distribution illustrates its symmetry or lack thereof. As explored in section 2 (Figure 2.5), the data of the seed differences follows a normal distribution; therefore, we can run a Monte Carlo simulation for skewness and kurtosis on seed differences and plot histograms to visualize where the values are centered.

Figure 4.5.1. Kurtosis on seed differences.

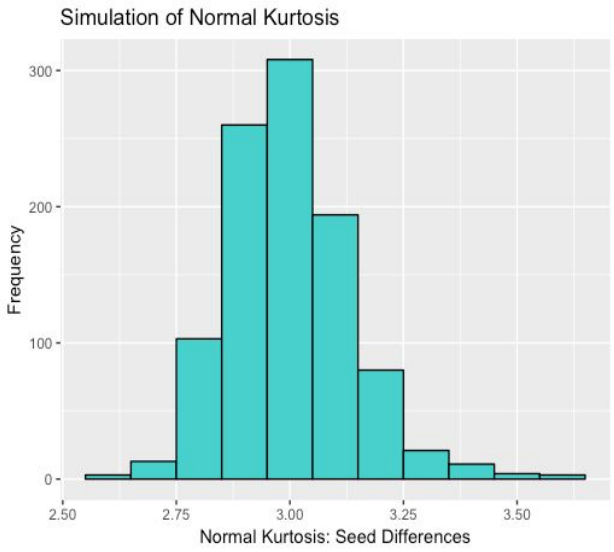


Figure 4.5.2. Skewness on seed differences.

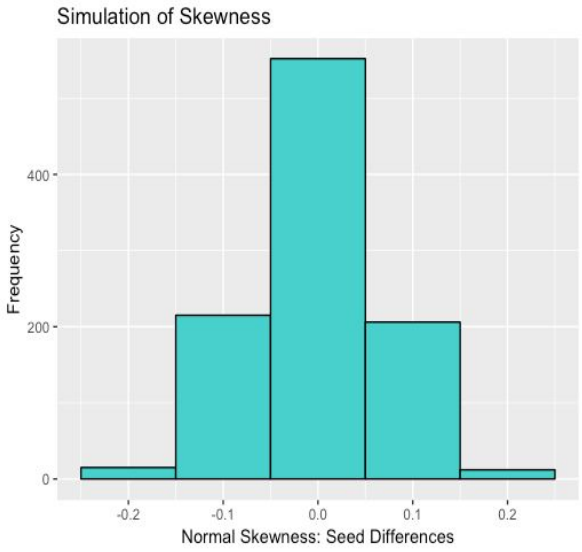


Figure 4.5.1 illustrates how a Monte Carlo simulation of the kurtosis of seed differences is centered around three, indicating that our seed differences follow a normal distribution and that we are correct in using logistic regression, which requires normality of its variables. In Figure 4.5.2, the skewness is centered perfectly at zero, meaning that the seed differences are symmetrical. Given these values, it makes sense to use them to build our models in order to predict the team to win the tournament.

We can also run these simulations on the scores of the games to ensure these distributions are normal and symmetric. The figures below display the spread of kurtosis and skewness of winning versus losing scores.

Figure 4.5.3. Kurtosis on winning scores.

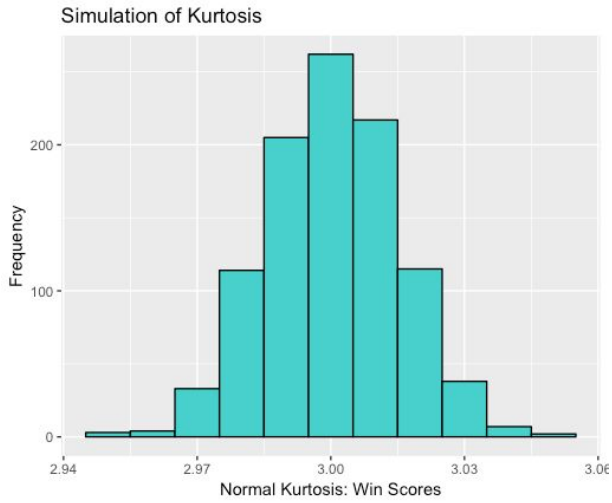


Figure 4.5.4. Skewness on winning scores.

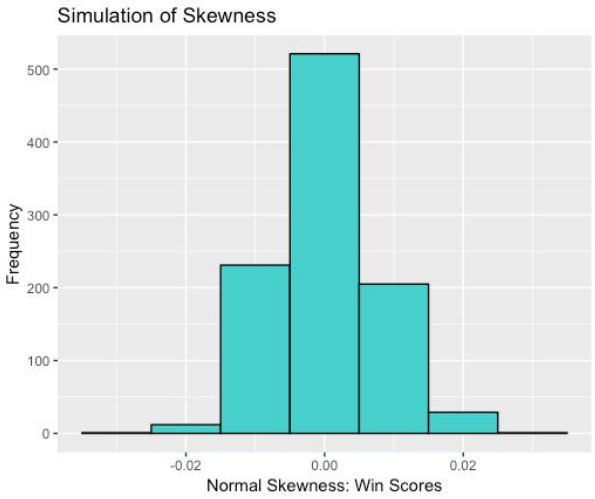


Figure 4.5.5. Kurtosis on losing scores.

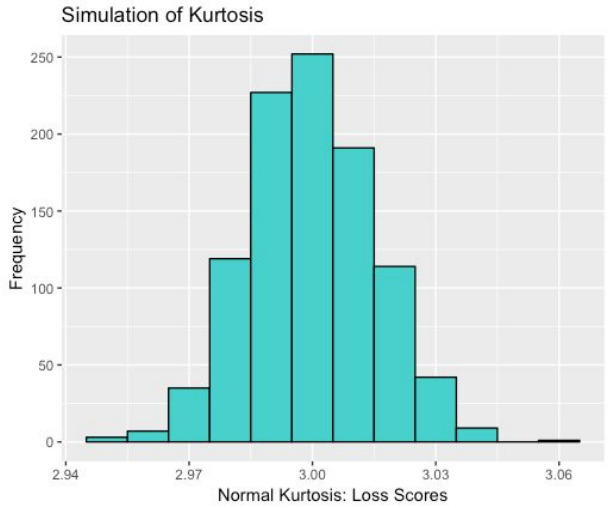
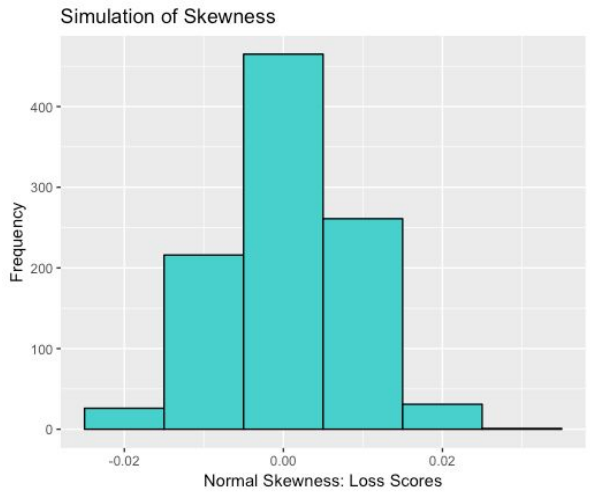


Figure 4.5.6. Skewness on losing scores.



Both distributions of winning and losing scores show the same results as our seed difference variable. With centers around three for kurtosis and zero for skewness, it specifies that they are both normally distributed with very symmetrical shapes. Due to these findings, we are able to further our search for a model by knowing the distribution of our variables. This helps us to check assumptions for certain GLM's since we can definitively state that the seed differences and scores are normally distributed.

5.) Theory

a.) Logistic Regression

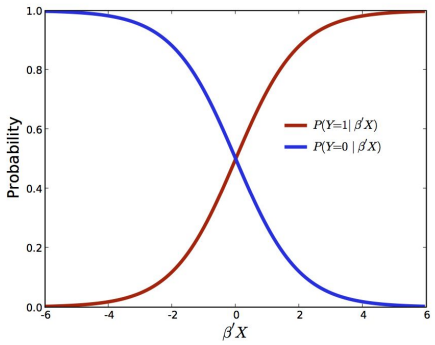
Generalized linear models (GLM's) are a class of nonlinear regression models that can be used in certain cases where linear models are not appropriate. Logistic regression is a specific type of GLM. Logistic regression is used for binary outcome data, where Y = 0 or Y = 1. Yet, these 0 and 1 outputs can also represent other results, such as win/lose or alive/dead. Logistic regression is defined by the following probability mass function:

$$P(Y = 1|X = x) = \frac{\exp(\beta' x)}{1 + \exp(\beta' x)} = \frac{1}{1 + \exp(-\beta' x)}$$

which implies that:

$$P(Y = 0|X = x) = 1 - P(Y = 1|X = x) = \frac{1}{1 + \exp(\beta' x)}$$

where $x_0 \equiv 1$ so β_0 is the intercept. This plot demonstrates the behavior of P(Y=1|X) and P(Y=0|X), plotted as functions of $\beta' X$.



The logit function {logit(x) = log(x/(1-x))} maps the unit interval onto the real line. The inverse logit function {expit(x) = exp(x)/(1+exp(x))} maps the real line onto the unit interval. In logistic regression, the logit function is used to map the linear predictor $\beta' X$ to a probability. The linear predictor in logistic regression is the conditional log odds:

$$\log\left[\frac{P(Y = 1|X)}{P(Y = 0|X)}\right] = \beta' X$$

Thus, one way to interpret a logistic regression model is that a one unit increase in X_j results in a change of β_j in the conditional log odds. Or, a one unit increase in X_j results in a multiplicative change of exp(β_j) in the conditional odds.

b.) Decision Tree

A decision tree model outputs a graphical representation of possible solutions, along with their probabilities, based on specified conditions. Computationally, the algorithm is checking if an attribute or set of attributes satisfy a particular condition and based on the outcome of that check, subsequent checks are performed. The tree is then splitting data into different parts based on the sequencing of these checks. Decision trees typically include chance event outcomes, resource costs, and utility. There are three components of decision trees:

- Root Node: Indication of the best predictor. The Root Node is located at the top of the tree and represents the independent variable.
- Decision Node: Part of decision tree where predictors are tested, each branch represents an outcome.
- Leaf Node: Part of the decision tree that holds a class label.

In decision analysis, a decision tree and its closely related influence diagram is used to visually and analytically represent the expected values or expected utility of competing alternatives. The process of running a decision tree algorithm on a dataset goes as follows: the data is split into independent (x) and dependent (y) variables, split into training and testing data, independent variables are scaled, and the decision tree is built. The criterion for the workings of the decision tree is entropy. The models parameters determine what function will measure the quality of a split. The functionality of this structure measures is to reduce randomness in the data, so that displayed results are more accurate. If given results are not promising, you can always manually tweak the parameters.

To make a prediction on the test data, you can use a classification or regression matrix, which will summarize the performance. This will give a better idea of what the model is doing right and wrong, and when wrong it indicates what types of errors are made.

Decision trees and other influence diagrams are considered advantageous because of their simple design and ease of interpretation. Furthermore, decision trees address the machine learning problem of overfitting when trained on small datasets as they have value even with limited hard data. Decision trees output the worst, best, and expected values for a scenario. They are also very useful in statistical analysis because decision trees can be combined with other decision-based techniques.

There are also aspects of decision trees that make them problematic in statistical analysis. For example, a small change in the dataset being fed into the model can lead to a large change in the structure of the decision tree, making its outcome unstable. When data is tested with a decision tree in combination with other predictors, other predictors will often outperform the decision tree in terms of accuracy. This observation is handled by replacing a decision tree with a

random forest of decision trees. Random forests are more difficult to interpret than decision trees, but have a higher accuracy rate. Calculations can become complex if there is a high degree of uncertainty among the values or if many values are linked.

c.) Random Forest

Random forests function as a way to average together individual trees, as stated above. The functionality of this mechanism is that it provides a more accurate, although more complex model of the data. Random forests are a popular refinement of bagged trees.

Since the process of bagging is relevant to both decision trees and random forests, we will describe its technicalities so that its associated data processing mechanisms are understood. Bagging is a term that essentially means bootstrap aggregation. This technique averages the procedure over many samples in order to reduce its variance.

$$\hat{C}_{bag}(x) = \text{Majority Vote } \{C(\mathcal{S}^{*b}, x)\}_{b=1}^{B}.$$

Computation involved in bagging a classifier.

The variables associated are as follows:

- C(S,x) is a classifier (tree's are an example of a classifier)
    - S: training data
    - X: predicted class label at input point x
    - S^*1,...S^*B: bootstrap samples drawn to bag C

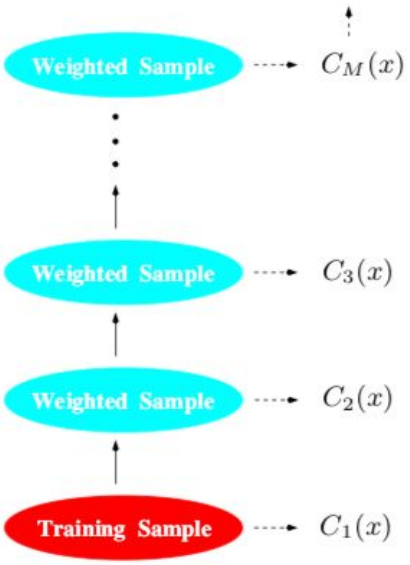
Reducing variance by bagging stabilizes procedures and leads to improved predictions. Unfortunately, the tradeoff for accuracy is simplicity. By averaging many trees, smoother decision boundaries are produced with a lower error rate.

Now back to the mechanics of random forests - as was stated, random forests utilize the same bagging techniques of decision trees yet performance is improved and complexity increases. At each tree split, a random sample of m features is drawn and only those m features are considered for splitting. Mathematically, m = sqrt(p) or log2(p), with p being the number of features. For each tree grown on a bootstrap sample, the error rate for observations left out of the bootstrap sample is monitored. This is referred to as the "out-of-bag" error rate. Random forests distinguish themselves from decision trees by attempting to improve on bagging by "de-correlating" the trees. Each tree is left with the same expectation.

Random forest theoretically dominates SVM and decision trees when ROC curves are calculated for all three. Another mechanism for computing random forests is boosting, which entails averaging many trees, each grown to re-weighted versions of the training data. The final classifier appears as follows:

$$C(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m C_m(x)\right]$$

Computation for computing the final classifier for a random forest.



Representation of classifier levels relative to the model data.

d.) Cross Tabulations

Cross Tabulations are used to quantitatively compare two variables by counting the frequency that each occurs. For example, to compare gender and their enjoyment of video games, the frequency of males liking and disliking video games would be counted, as well as the frequency of females liking and disliking video games. A supplementary Chi-Square test is used to determine if the two variables are independent of each other because if they are independent, then the test will be non-significant and the variables have no relationship. The Chi-Square value is computed by the equation shown below:

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

In this equation O denotes the observed value and E denotes the expected value.

e.) Generalized Linear Model

The modeling assumptions for a Generalized Linear Model (GLM) are: the Y_i are conditionally independent given X and the probabilty mass function/density can be written:

$$\log p(Y_i|\theta_i, \phi, X_i) = w_i(Y_i\theta_i - \gamma(\theta_i))/\phi + \tau(Y_i, \phi/w_i)$$

where w_i is a known weight, $\theta_i = g(\beta' X_i)$ for an unknown vector of regression slopes β, $g(\cdot)$ and $\gamma(\cdot)$ are smooth functions, φ is the "scale parameter" (known or unknown), and $\tau(\cdot)$ is a known function. The log-likelihood function is:

$$L(\beta, \phi|Y, X) = \sum_i w_i(Y_i\theta_i - \gamma(\theta_i))/\phi + \tau(Y_i, \phi/w_i)$$

The score function with respect to θ_i is:

$$w_i(Y_i - \gamma'(\theta_i))/\phi$$

If we let f_θ (Y) be a density in Y with parameter θ, the score function becomes:

$$\frac{\partial}{\partial\theta}\log f_\theta(Y) = f_\theta(Y)^{-1}\frac{\partial}{\partial\theta}f_\theta(Y)$$

When calculating the expected value of the score function, the value 0 is found when θ is at its true value. Since the expected value of the score function is 0, it can be concluded that

$$E(w_i(Y_i - \gamma'(\theta_i))/\phi|X) = 0,$$

so

$$E(Y_i|X) = \gamma'(\theta_i) = \gamma'(g(\beta' X_i))$$

GLMs were created to unify various statistical methods such as a Gaussian linear model, logistic regression, Poisson regression, and negative binomial regression. The Gaussian linear model GLM example is shown below. The density of Y | X can be written as:

$$\begin{aligned}\log p(Y_i|X_i) &= -\log(2\pi\sigma^2)/2 - \frac{1}{2\sigma^2}(Y_i - \beta' X_i)^2 \\ &= -\log(2\pi\sigma^2)/2 - Y_i^2/2\sigma^2 + (Y_i\beta' X_i - (\beta' X_i)^2/2)/\sigma^2\end{aligned}$$

This can be put into GLM form by setting $g(x) = x$, $\gamma(x) = x^2/2$, $w_i = 1$, $\varphi = \sigma^2$ and $\tau(Y_i, \varphi) = -log(2\pi\varphi)/2 - Y_i^2/2\varphi$.

f.) AdaBoost

Adaboost can help create a strong classifier, and could be applied in this case. In general, Adaboost is useful because it could be less susceptible to overfitting, and more protected against outliers and missing data. In our case, there are not that many rows with missing data. But, there may be overfitting for some matches. Adaboost uses observation weights. The larger the weight, the more accurate the corresponding classifier is. The weights are then iterated in a function loop to calculate each classifier function for each weight. Essentially, Adaboost builds an additive logistic regression model *f(x)* by stagewise fitting using the loss function *L(y,f(x))*:

$$f(x) = \log \frac{\Pr(Y=1|x)}{\Pr(Y=-1|x)} = \sum_{m=1}^{M} \alpha_m G_m(x)$$

&

$$L(y, f(x)) = \exp(-y\, f(x)).$$

Given each f(x), the solution for the Adaboost algorithm is:

$$\arg\min_{\beta, G} \sum_{i=1}^{N} w_i^{(m)} \exp(-\beta\, y_i\, G(x_i))$$

Where G is a tree classifier, and β is a coefficient

Step by step process for Adaboost algorithm:

(1) The AdaBoost classifier first initializes the observations weights:

$$w_i = 1/N,\ i = 1, 2, \ldots, N.$$

(2) For m = 1 to M:

A classifier is then fit into the training data using weights w_i, and weighter error is computed for newest tree through:

$$\mathrm{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq C_m(x_i))}{\sum_{i=1}^{N} w_i}.$$

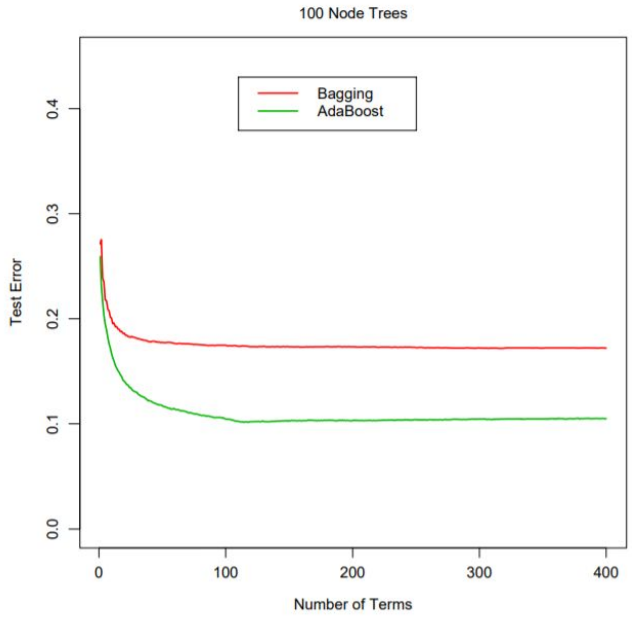
Compute $\alpha_m = \log[(1 - \mathrm{err}_m)/\mathrm{err}_m]$.
Update weights for $i = 1, \ldots, N$:
$w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq C_m(x_i))]$
and renormalize to w_i to sum to 1.

With $C_m(x)$ being the classifier for the corresponding weight w_i

(3) Output and final equation of AdaBoost:

$$C(x) = \mathrm{sign}[\, \Sigma \alpha_i C_i(x)\,]$$

We can also see the comparison of the AdaBoost versus Bagging with 100 node trees, and 2000 points from nested spheres. The error rate reduces drastically faster than the bagging tree and approaches a 0.1 test error.



By inputting all the variables into an AdaBoostClassifier, we can check the prediction of the games and the winner in the march madness tournament bracket.

6.) Conclusion

Throughout this report, we attempt to build different statistical models that help in predicting the outcome of the 2019 WNCAA March Madness tournament. Using records of game outcomes over the years, we can try to guess future results. A team's chances of winning the tournament are found by logistic regression of a team's seed and team's performance, a decision tree, a random forest, cross tabulations, and Monte Carlo simulations for analysing kurtosis and skewness calculations. In addition, the theory behind each of these models was shown, with an additional explanation of generalized linear models. Since our models are based off past events, we can not be entirely sure of our predictions because upsets always occur within sports, where a team with a lower seed or worse win history beats the predicted team. In summation, these statistical models are our efforts to complete the Kaggle competition.

Works Cited

Bowen, F. (2019, March 13). Women's college basketball wasn't always dominated by big. schools. First came the Mighty Macs. Retrieved March 15, 2019, from https://www.msn.com/en-us/kids/sports/womens-college-basketball-wasnt-always-dominated-by-big-schools-first-came-the-mighty-macs/ar-BBUJSKw

Editors, H. (2009, November 16). 'March Madness' crowns its first men's NCAA Champion. Retrieved March 10, 2019, from https://www.history.com/this-day-in-history/march-madness-is-born

Prabhakaran, Selva (2016-17). 'Logistic Regression with R'. Retrieved March 15, 2019, from http://r-statistics.co/Logistic-Regression-With-R.html