Spring 2018

Comparing Various Machine Learning Statistical Methods Using Variable Differentials to Predict College Basketball

Nicholas Bennett
nab85@zips.uakron.edu

Comparing Various Machine Learning Statistical Methods Using Variable Differentials to Predict College Basketball

Nick Bennett

Abstract

The purpose of this Senior Honors Project is to research, study, and demonstrate newfound knowledge of various machine learning statistical techniques that are not covered in the University of Akron's statistics major curriculum. This report will be an overview of three machine-learning methods that were used to predict NCAA Basketball results, specifically, the March Madness tournament. The variables used for these methods, models, and tests will include numerous variables kept throughout the season for each team, along with a couple variables that are used by the selection committee when tournament teams are being picked. The end goal is to find out which machine learning method populates the most successful bracket by using key differential statistics between teams and variables of past tournament winners using Neural Network, Boosted Decision Trees, and Naïve Bayes methodologies.

KEYWORDS: Machine-learning, NCAA Basketball

I. Introduction

In an increasingly technological world, more and more aspects of everyday life are data-driven. Companies rely on data-supported models to make decisions in a multitude of fields, such as consulting, investments, banking, and insurance. With the advancement of technology, data and statistics usages have only improved also. Of course, statistical software has existed for a couple decades now that can produce mid-to-high level statistical tests instantly, such as regression and ANOVA tests. However, in a more recent timeframe, statistical machine learning predictions have started to break through. From a general sense, machine learning is defined as "an application of artificial intelligence that provides systems the ability to automatically learn and improve from experience without being explicitly programmed." (Expert Systems, 2018) In a more specific standpoint, statistical machine learning is "the development of algorithms and techniques that learn from observed data by constructing models that can be used for making predictions and decisions" (Hutter, 2008). Essentially, in most statistical machine learning techniques, if you give the computer enough data, tell it which variables to pay attention to, and then give it a target variable, the computer will be able to predict future target variable values based on the given variables/data. The initial idea alone was incredible, but to have it be possible now by using a computer alone is next-level technology.

In this project, various methods of statistical machine learning will be applied to predict sports results; specifically for NCAA basketball. With SO much

data readily available to anyone with Internet access in today's world, it will be interesting to see how the various statistical techniques' results will compare to one another. To be specific, this project will use statistical machine learning techniques paired with data to observe and train algorithms/models that will then be used fill out brackets for the NCAA Basketball tournament. The goal is to determine which statistical machine learning method will perform the best in this data scenario. The three methods to be tested will be neural networks, decision trees, and Naïve Bayes methods. Each method will be discussed further in detail later; however, it is important to understand the background of the topic of the data first.

II. Topic Background

In a sporting event unlike any other in existence, the NCAA basketball Tournament is, to many, the crown jewel of all non-professional athletics in the world today. Every year in March sixty-eight teams meet for a nearly month-long collegiate basketball tournament to determine a National Champion in a tournament aptly-named: March Madness. Millions of hopeful fans nationwide tune in to watch their favorite teams play, and hundreds-of-millions fill out their own brackets with the team they think is going to win it all. In almost 80 years, no individual has EVER filled out a perfect bracket and guessed every winner correctly (SI Wire, 2017). With sixty-three total games to pick, the odds of picking a flawless bracket are 1 in 9,223,372,036,854,775,808, or 1 in 9.2 quintillion (Greenawalt, 2018); so it is no surprise that this feat has not been achieved yet.

The way that the tournament is set up is rather special in itself because any team collegiate team has a CHANCE to make it. At the end of the regular season,

each college team will play in their respective conference tournaments; and the winner of each conference tournament gains an automatic bid to the NCAA tournament. This is the easiest route to make the tournament also know as the "Big Dance"; however, the remaining teams that do not win their conference championships still have a chance to get selected for the tournament too. After the conference tournaments are over, "Selection Sunday" takes place and the tournament selection committee selects the remaining teams. Of the sixty-eight teams, thirty-two of the teams receive automatic bids through their conference championships, and the committee, based on several variables, picks the remaining thirty-six teams. These "at-large" bids are decided by a combination of specific statistics, such as strength of schedule, strength of record, etc. (Ellentuck, 2018). From there, teams are seeded from #1 to #16, in four geographical divisions, to create the final bracket. One might look at that breakdown of teams and think, "How are there only 64 seeds, but 68 teams in the bracket?" This is because there are four "play-in" games in which the winner of those games continues in the tournament. Overall, by the end of the play-in games, there are sixty-four teams split into four regions that battle it out for the national championship. These matchups are the ones that this project will attempt to predict; however, this is not possible until the data itself, methods of aggregation, and variables choices are understood.

III. The Data

a. Data Background

Overall, numerous months were spent attempting to fine-tune the data to a point that made the most logical sense. It was clear that the project goal was to use

past game data to train and predict future winners, by means of a categorical variable; however, the process of deciding this was not as simple. At the end of the research, the following decisions were made in regards to the data:

1. Using past tournament game data, compared to current year game data, made more logical sense for several reasons.
2. Games should not be entered and trained in the model by year; the aggregated data order used should be randomized.
3. Variables used will actually be the differential between the two teams playing one another.

Explanations of each of these decisions are important and go as follows.

i. The Decision to Use Past Data vs. Present Data

During the fall of 2017, an independent study was conducted to research various machine-learning methods that were to be used with this senior project. While doing so, some practice data was used in the same manner in which the actual NCAA basketball data would be used with the three machine-learning techniques to test for any formatting errors, issues with variables, etc. Once these test models started, almost immediately, the question arose of "Should you use game data from each team from this CURRENT year to train the models? Or should you train the models with games from the PREVIOUS years of the actual tournament games?" The answer to this question was not known off hand, so both sides of this question were researched deeper.

To begin, the method of using the present, regular season data was examined first. Without delving into the machine-learning model and methods that will be

later used, it became clear that using current year games and statistics would not work well for tournament games. The practice models used with present data took ten random, regular season games and their outcomes from each team in the tournament. From here, the statistical variables and categorical variable of who won each game "Home or Away" was used to train the practice machine-learning models. This categorical variable of "Who Won?" was the target variable that would be predicted once the tournament matchups were entered. After running the initial models with this current year's data, it was soon clear that this type of data had its issues. Most importantly, the model was used to looking for "home" and "away" teams and making its decisions likewise. The models tended to notice that home teams tend to win more often. However, in the tournament, there are no home and away teams; all games are at neutral sites. If anything, the only advantage that one team really has over another is the team with the higher ranking TENDS to have the shorter drive to the neutral site. Also, for these practice models, when they were run, they were defaulted to having the higher-ranking teams in the "home" slot, and vice versa. However, this method ran into issues in later rounds when seeds with the same ranking were playing one another: 1v1, 2v2, etc. Finally, it can also be argued that using outcomes from games during the year do not always accurately represent how the tournament will play out (Newton, 2018). It is not uncommon for lower ranked teams to win multiple games in a row, gain momentum, and make a strong run through the tournament. Overall, these are all reasons why using present, regular season game data would not be ideal for predicting tournament winners.

ii. Randomization of Game Data

This data standard is one that should be relatively standard; however, it should still be clarified in this situation using machine learning. As more and more practice models were conducted, it was clear that just entering the game data in year after year (Ex: All 65 tournament games were listed in order from 2007, then 2008, then 2009, etc.) was not the best manner to predict the most tournament winners. When the machine learning methods are being trained by the game data, it is almost as if the computer is looking for trends and patterns in the categorical target variable. For example, if for whatever reason the game data was entered in a manner in which the "Who Won?" target variable consistently had a lower seed winning every 5 lines, then the resulting model would have a lower seed winning every 5 lines, regardless of the actual variable data that the models were looking at. Thus, by randomizing the order of the games entered each time, it is less likely that the training data will not have the possibility of these winner outcome patterns that tend to happen as the tournament goes on. Randomizing the game data's order causes the model to rely solely on the values of the variable data that is being used to train the resulting "Who Won?" target variable. These reasons are why randomizing the game data is absolutely crucial.

iii. Using Differential Variable Data

Once again, this is a data decision that was made by a trial-and-error process. When the original practice data to generate some practice models to find how the final model data should be formatted was pulled, it was done in the following steps:

1. The outcome of every single tournament game from 2007-2008 to 2016-2017 was pulled. Thus, an outcome of "Who Won?" would be a "Higher" or "Lower" seed categorical variable.
2. From here, data from each team that made that tournament in those particular years was then pulled.
3. Finally, to train the models, the data from Team 1 and Team 2 were then listed in a row together, along with which seed (Higher or Lower) won that game.
4. At the end of the data gathering, we had roughly 500 different game outcomes from past tournament, along with the yearly team data for each team in that particular matchup. For example, one data observation might have looked like: "Team 1 Strength of Schedule, Team 1 Ranking, Team 1 Points per Game, etc…. Team 2 Strength of Schedule, Team 2 Ranking, Team 2 Points per Game, etc… Outcome of the Game".

At the end of the day, there were roughly 15 variables for both Team 1 and Team 2, and the outcome for each game. However, when it came time to decide which variables were statistically significant and should be used, the issues with the data setup were exposed. When a logistic regression was conducted on all of the variables and how they predict the "Who Won?" variable, it was quickly clear that having data variables for both Team 1 and Team 2 was not the way these models should be approached. For example, it makes NO sense for the chi-squared testing on each variable to say that Team 1's Strength of Schedule is statistically significant when predicting the categorical winner of each game, but Team 2's Strength of

Schedule is not needed. This is an issue that happened numerous times in the chi-squared testing used to determining which variables were important in predicting the winners; where a certain statistic for one team was important, but not from the opposing team. To get around this issue, it was clear that taking the differential between Team 1 and Team 2's values for each variable was the best way to make sure the statistically significant variables were not missed. These differential values for each variable will be used for each model to ensure the predicted models are as strong as possible, not leaving out any important variables.

b. How the Data were Aggregated

Another piece of information that is relevant to know in regards to the data that will be used for these models is how the data were aggregated. To be concise, there were different steps required to gather data needed for these predictive models.

1. To begin, it was necessary to gather the outcome of every tournament game for the past eight years, from the 2009-2010 season, up until last year, the 2016-2017 season. These sixty-three games per tournament, (or 504 total game outcomes) when paired with the differential statistical variables of the two teams that played one another, is what will be used to train each predictive model. These game outcomes, from the last eight years of tournament games, were pulled from https://www.sports-reference.com/cbb/seasons/. In an abbreviated sense, this website contains every game outcome, from every NCAA tournament ever. The game

outcomes (Who won?) were pulled into an Excel file and worked with from there.

2. Next, it was necessary to gather the desired statistical data for each team in the tournament, in their respective years. Once again, https://www.sports-reference.com/cbb/seasons/ was used to aggregate this data. The site splits up the data, year by year, for every team in the nation. So, EVERY possible variable was exported for each tournament team, year by year, to give as much depth to the model as possible. In the end, THIRTY-FIVE variables for each team in the tournament were exported to an excel file where each teams' statistics would be matched with the outcome of each tournament game. These team statistics were matched to individual tournament games by the use of "v-lookup" functions in excel. Essentially, a v-lookup function matches data from separate data tables based on what data points the user wants to use.
3. Although this step turned out to be unnecessary later on in the models, another separate variable that was aggregated purely by hand was the distance each team drove to each game; this was done by researching the distance each college travelled to each neutral site where games were played. The only reason this step was labeled as unnecessary is because, when the variables were later tested for statistical significance, mileage traveled to each game was inconsequential; however, it was still a statistic that would have been interesting IF it was statistically related to which team won.

4. Finally, it was necessary to take the difference of each team's data, per variable, in order to ensure the best variables would be chosen (explained previously). Although this is just an example, the layout of the before this step would look something like this:

Winner (By Ranking)	Team 1 Rank	Team 1 SRS	Team 1 SOS	Team 2 Rank	Team 2 SRS	Team 2 SOS
Higher	1	23.85	9.7	63	-3.46	-4.98
Higher	30	16.92	8.08	34	13.44	8.96
Higher	17	17.43	10.95	48	11.19	2.06
Higher	16	16.92	7.22	51	7.64	-0.55
Higher	22	13.28	7.15	46	9.38	-0.27
Higher	10	18.81	10.59	57	0.75	-4.41
Higher	27	14.56	4.35	39	14.45	10.93
Higher	8	20.97	3.15	60	3.06	-3.4
Higher	4	19.64	9.86	68	-7.92	-5.83
Higher	32	14.55	9.36	38	14.98	7.69
Higher	20	18.11	8.98	49	7.15	-0.58

Then, after taking the differential of each given variable, the data used in the later predictive models would look like the following:

Winner (By Ranking)	Differential in Rank	Differential in SRS	Differential in SOS
Higher	-62	27.31	14.68
Higher	-4	3.48	-0.88
Higher	-31	6.24	8.89
Higher	-35	9.28	7.77
Higher	-24	3.9	7.42
Higher	-47	18.06	15
Higher	-12	0.11	-6.58
Higher	-52	17.91	6.55
Higher	-64	27.56	15.69
Higher	-6	-0.43	1.67

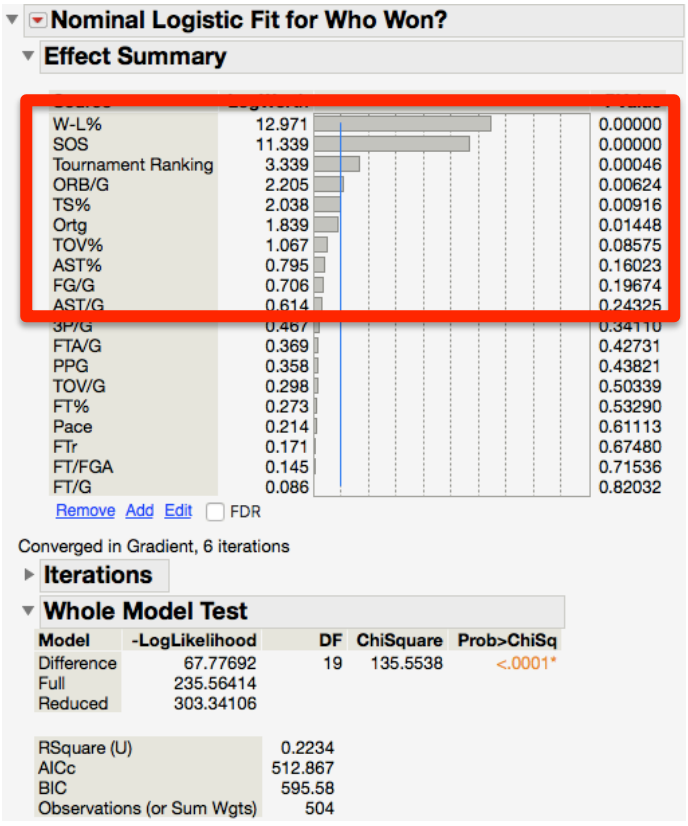
As is visible in the spreadsheets above, the variables changed from having a separate Team 1 and Team 2 variable for each, to have the variable be the differential of the two teams; that is:

(Team 1 Variable) – (Team 2 Variable) = (Differential Variable)

In the end, these steps taken to aggregate and organize the data so it was in a position to simply enter in the later predictive models was as important as the models themselves. When data are sloppy and inconsistent, results are sloppy and inconsistent.

c. Selecting Significant Variables

As stated in the previous passage, when the data were pulled for every team, there were THIRTY-SIX (!!) total variables that could be used to train the models and predict the final "Who Won?" categorical variable. Each of these was actually the differential of the values for each respective variable between the two teams, for reasons stated earlier. For example, "points per game" is really the Team 1's points per game minus Team 2's points per game. Of course, this amount of variables is an absurd amount and could lead to weaker coefficients for the variables that actually matter, so a process was conducted to narrow down the mass amounts of variables to the ones that really mattered. Using JMP Software, a logistic regression test was conducted on the 504 game observations from past years of tournament games and their respective outcomes, differential statistics between the two teams, etc.



The logistic regression test was testing the "Who Won?" (Higher of lower seed) variable to see which resulting variables would be statistically significant, by means of p-value testing. For the sake of the mass amount of variables that were used, any variable with a p-value lower than .4 was kept. This narrowed down the amount of variables from thirty-six to nineteen. Finally, to ensure even stronger parameters, another logistic regression test on the "Who Won?" variable was done with the remaining twenty variables. This time, any variable with a p-value lower than .2 was kept (These results are shown in the picture on the previous page). These are the final variables that will be used to train and test the actual models that will results from this project. In the end, it was found that the following nine variables were the most statistically significant in predicting which teams would win in the March Madness tournament:

- Win-Loss Percentage:
    - Total amounts of wins divided by total number of games.
- Strength of Schedule:
    - Refers to the difficulty or ease of a team's schedule in comparison to their opponents (Sports-Reference, 2018). Average score is zero. Positive scores denote a harder schedule, and vice versa.
- Tournament Ranking:
    - Each teams ranking in the tournament from 1 to 68; the lower the better.
- Offensive Rebounds per Game
- True Shooting Percentage

  - Refers to a measure of shooting efficiency that takes into account 2-point field goals, 3-point field goals, and free throws for each team (Sports-Reference, 2018). This exact formula would be:

$$\frac{Total\ Points}{2*Total\ Field\ Goals\ Attempted+0.95*Free\ Throws\ Attempted}$$

- Offensive Rating:
  - Refers to the amount of points scored per 100 possessions (Sports-Reference, 2018). This exact formula would be: $\frac{100*(Points)}{(\#\ of\ Possessions)}$

- Turnover Percentage
  - This is an estimation of the amount of turnovers a team will make per 100 plays (Sports-Reference, 2018). The exact formula would be:

$$\frac{100*(Total\ Amount\ of\ Turnovers)}{((Field\ Goal\ Attempts)+0.475*(Free\ Throw\ Attempts)+(Turnovers))}$$

- Assist Percentage
  - The percentage of assists a team makes per possession (Sports-Reference, 2018).
- Field Goals per Game

All in all, these statistically significant variables that will be used to train the resulting models really DO make sense. Teams that win the greatest percentage of the time tend to do better in the tournament, which supports the Win-Loss percentage variable. Next, strength of schedule has always been a solid indicator of later tournament success; teams that experience tougher teams during the regular

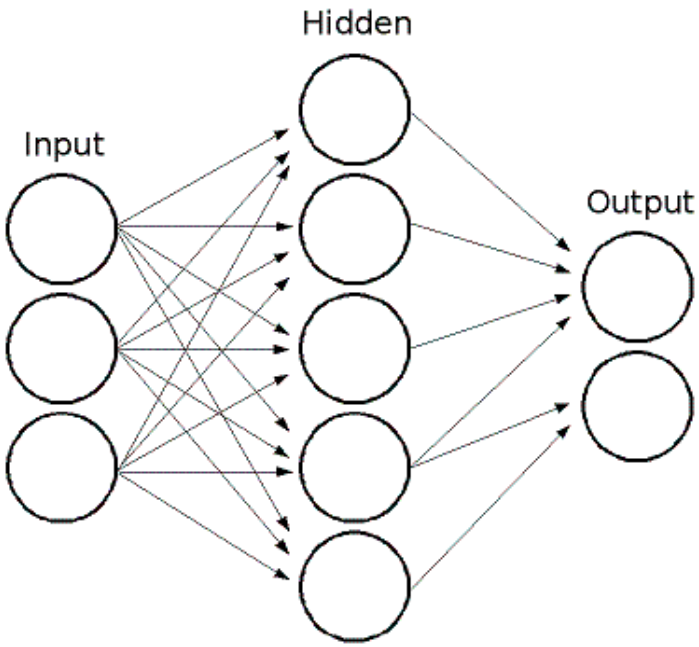
season are better prepared for the pressure of tournament games. In addition to these, variables such as offensive rating, field goals per game, and true shooting percentage also are completely understandable because once the tournament comes, teams that are the strongest offensively tend to do well. And finally, variables such as turnover percentage and assist percentage are essential because teams that can really pass while taking care of the ball tend to be successful also. In the end, these variables were deemed the best to use to train the machine-learning models. The only thing left to do before getting into the models and their results is to understand the statistical machine-learning methods themselves a little bit.

IV. Machine Learning Methods

Overall, there are two different types of machine learning using today's technology: Supervised and unsupervised learning. Supervised learning is when the software analyzes training data and produces an inferred function towards a specific variable, or variables, to predict future outputs/data (Polamuri, 2014). Using this function, it is possible to then predict outcomes afterwards using different data. On the other hand, unsupervised learning is a bit of an unknown. Unsupervised learning is when the software is given data and then attempts to find hidden structure and trends on its own without the user telling the system to focus on certain variables (Polamuri, 2014). In the context of this project, each of the machine learning methods that will be used to predict NCAA bracket outcomes will be supervised learning. Neural Networks, decision trees, and Naïve Bayes are all supervised learning because there is a specific variable that they target when trying to predict outcomes with future data.

a. Neural Networks

In regards to supervised machine learning, neural networks are popular because they can compute any function (Nielsen, 2017). The name "Neural Networks" is in relation to this method's similarities to the human body's own biological computational system, both aesthetically and functionally (Mannarswamy, 2017). Neural networks are, essentially, the brain's own method of classifying incoming data, feelings, etc. In relation to this, machine learning neural networks are similarly named for their ability to intake mass amounts of data and artificially adjust on their own to create the strongest predictive model. Essentially, neural networks generally focus on pattern recognition to make universal approximations, and still work well if the data you are using contains lots of differences and variation; much like the amount of difference in the statistics of a #1 seed college basketball team and a #16 seed. They tend to be more successful when the volume of data is great and there are many variables; but not too many variables (Beam, 2017). Also, NN's tend to work even better if variables are unrelated. With a dataset that has a vast amount of variables and topics, like the NCAA tournament, it will be interesting to see how neural networks perform.

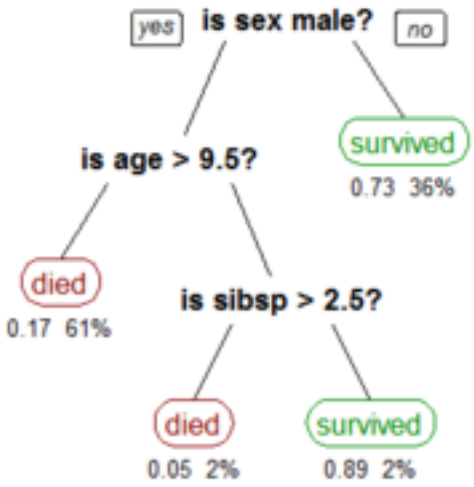


Neural networks are made up of connected layers of computational units called “nodes” (Shah, 2017). These nodes are given random weights that are used to eliminate any bias in the data. These layers of nodes then transform the data until they can make sense of it and classify it into a certain category. Each layer has a specific function that is chosen for it prior to the running of the data. These functions can be Tangential, Linear, or Gaussian, and are predetermined by the individual user. The data are then taken into each node, from layer to layer; at the end, the results are compared to the expected results and errors are put back through the system in a process called back propagation (Shah, 2017). The system then auto-adjusts the weights of each node to make the end results as close to the expected results as possible. In the end, the neural network will have created a model that will create the least amount of error with the greatest number of observations. In JMP, the neural network does not create an actual specific equation for the model algorithm; instead, the algorithm is the set of various functions that are used in the different nodes and layers, in relation to the given data to calculate the predicted values desired.

b. Decision Trees

Decision trees are another form of supervised machine learning that are becoming more and more well known due to their lesser difficulty, high accuracy, and ease of interpretation (Vidhya, 2016). Also, unlike other models designed to predict linear data, decision trees tend to work well with non-linear sets. They work with both categorical and numerical variables. Once the decision tree has the data set with variable names, the model works to identify the most important variable

(Gupta, 2017)



that predicts the greatest amount of the dataset as a whole. In this specific project, a "categorical variable" decision tree will be used because the target variable ("Why Won?") that the model is looking to predict is a binary answer (Higher or Lower). The decision tree starts with a "root node" of data that is, essentially, the entire data set; this gets split into two or more trees (Sanjeevi, 2017). When these additional trees are split into further branches, they are then called "decision nodes". Finally, these branches can be split into "leaves" to sort them categorically. Each node represents a certain feature of the data, each branch denotes a rule about the data, and each leaf represents an outcome (categorical: Higher or Lower). In this project, a "Boosted" decision tree will be used, meaning that each resulting tree learns by using the residuals from the tree before it. This is a measure used to ensure the variance is as small as possible (Petersen, 2018). In the end, the system creates a "tree" of the data to create an outcome for every possible scenario that can be entered in (Sanjeevi, 2017). Overall, similar to the neural networks, the decision tree trains itself to predict categorical outcomes based on groupings of data previously entered into it that fit a certain output category (Higher or Lower see) based on their related variable data (Points per game, strength of schedule, etc.).

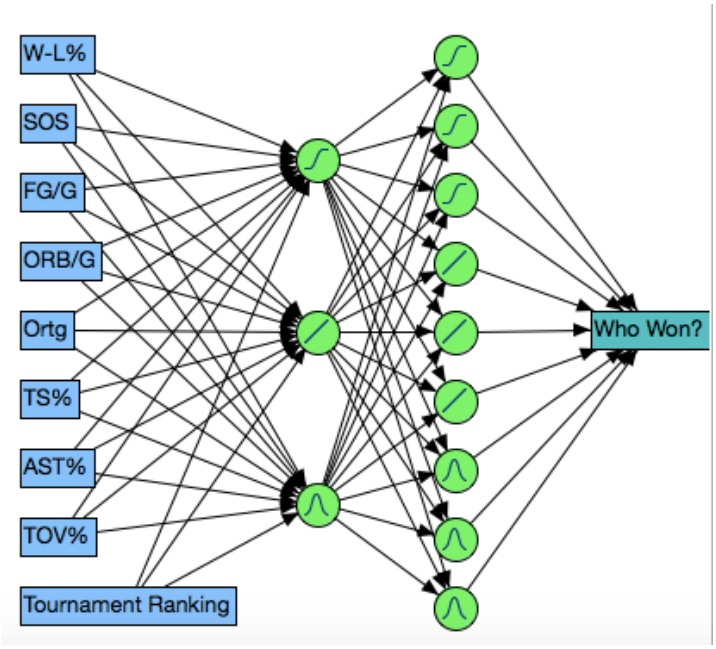
c. Naïve Bayes Model

The final statistical method of machine learning that will be used to predict NCAA tournament winners in this project is the Naïve Bayes classifier. It is generally based around Bayes Theorem; or the probability that one event will happen, given prior knowledge of conditions exist that might be related to that event (Merriam Webster, 2018). However, instead of using JUST Bayes Theorem, Naïve Bayes uses a group of algorithms that all acknowledge the notion that all variables in the data set being used are independent of one another (Aylien, 2015). This idea that no two variables can be related at all is very unlikely, which is where the name "Naïve" comes from. The idea of independence is rather clear and simple; however, Naïve Bayes is still successful in many applications where a simple binary classification is being sought after using vast multitudes of datasets, both linear and non-linear. Essentially, the model will predict the outcome, Higher or Lower seed, based on a given group of variable data using probability. Similar to decision trees, when the user loads mass amounts of data into the system, the model will split the data into sections based on learned parameters (Ex: Variables that are a certain category, or values above or below a distinguishing value). These sections each have a given probability. It will then group these "independent" sections/probabilities and train itself to make similar categorical predictions based on the previous categories of the dataset parameter groups. Except, in this situation, the system splits up each variable in the most common parameters and then gives a probability value to each of those section. Then, using Bayes Theorem, the model will group those sections and variables, and find the overall probability

that an observation of data is a certain category depending on what the highest parameter section's probability is. In the end, the Naïve Bayes classifier model classifies each observation of data based on which set group of parameters they are most likely to fit into. Using probabilities, the model is able to predict a given category for all new inputted data.

V. Actual Models Shown

Now that each statistical machine learning technique is more familiar and the data set and aggregation methods are understood also, it is time to acquire the actual models. Only the nine variables that were decided to be the most statistically significant will be used in any predictions. All three models will be generated using JMP software.



a. Neural Network Model

In the first model, a neural network predictive model was created using all three possible node functions: tangential, linear, and Gaussian. To be specific, this

Who Won?

Measures	Value
Generalized RSquare	0.4395718
Entropy RSquare	0.3055441
RMSE	0.3682637
Mean Abs Dev	0.2740119
Misclassification Rate	0.1942605
-LogLikelihood	189.19942
Sum Freq	453

Confusion Matrix

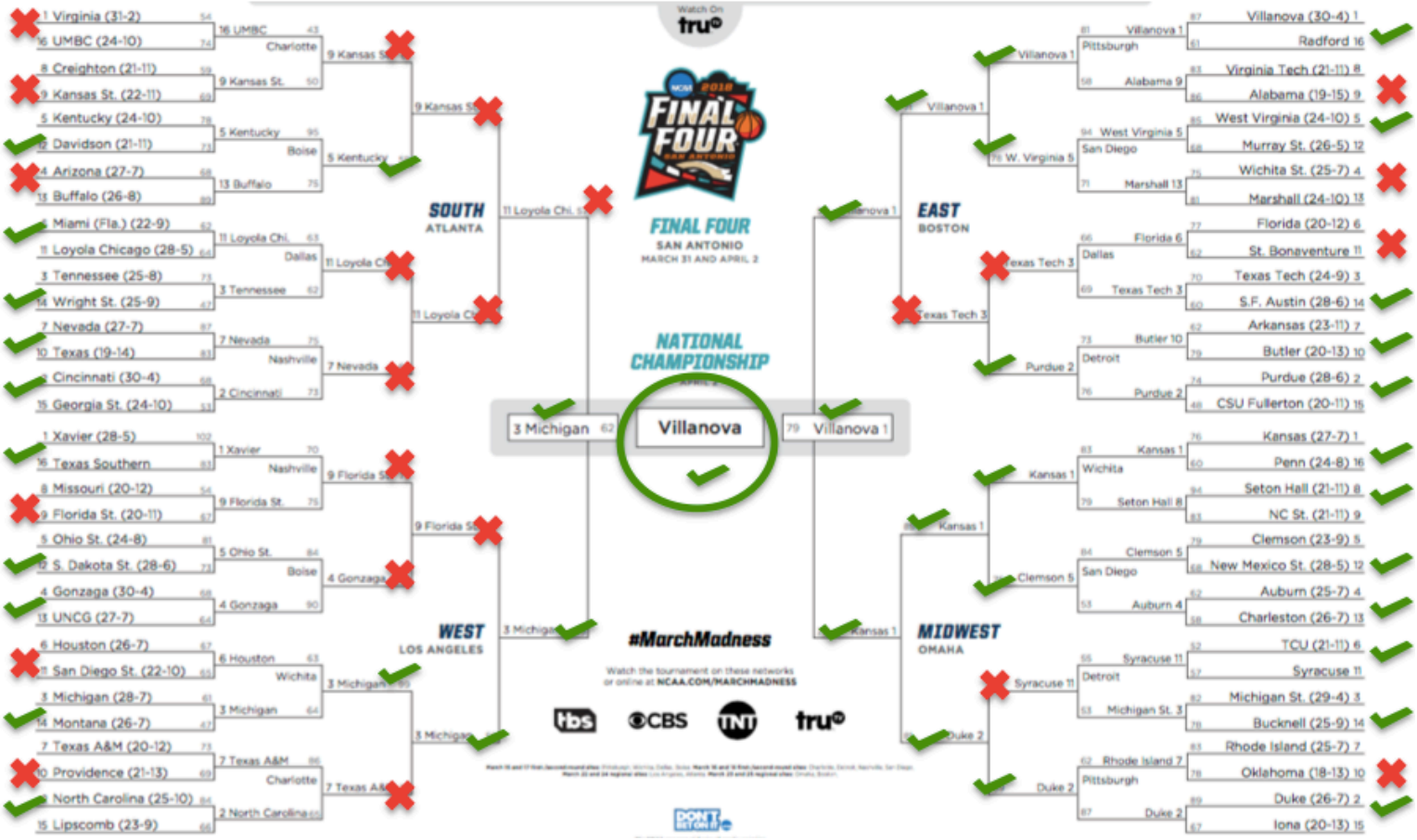
Actual	Predicted Count	
Who Won?	Higher	Lower
Higher	291	31
Lower	57	74

Confusion Rates

Actual	Predicted Rate	
Who Won?	Higher	Lower
Higher	0.904	0.096
Lower	0.435	0.565

model used one random function of each function type in its first layer, and then three of each function in the second layer. Training this layered model with the past eight years of data, using the nine variables decided upon caused the model to give the following output. Overall, the model was set to use 90% of the original 504 data observations to train the data, and then the final 10% were used to validate and test the data. For a model type with so much variance and different network possibilities, an r-squared value of 0.44 is not great, but for a neural network, it's not bad. After creating the given model, data for this year was entered into the model, game-by-game, round by round, to see how what the neural network methodology predicted. The resulting bracket looked as follows.

Overall, despite having an r-squared value that was not high, the neural network predictive model predicted 40 out of 63 total games, right around 64%. However, despite the *overall* selection percentage being only a little above average,



if the bracket were to have been entered in this year's "ESPN Tournament Challenge", it would have finished in the 99.8th percentile nationwide in scoring out of over 17 million bracket entries (1410 points out of 1920), according to ESPN.com. This incredibly high percentile finish was due to the neural networks' ability to predict the later round games well. This was very pleasing to see.

a. Boosted Decision Tree Model

In the second model, a decision tree predictive model was created using boosting. To be specific, this model has a maximum of 3 splits per tree, and a maximum of 50 total tree layers. The learning rate in this specific tree is 0.1. The learning rate can be anywhere between 0 and 1; the larger the learning rate, the longer the model will take to train and the more specific the model will be. In this situation, the learning rate will allow for the tree model to cover the majority of correct predictions without getting to specific in regards to the training data. Training this layered model with the past eight years of data, using the nine variables decided upon caused the model to give the following output.

Specifications

Target Column:	Who Won?	Num
Number of Layers:	12	Num
Splits per Tree:	3	
Learning Rate:	1	
Overfit Penalty:	0.0001	

Overall Statistics

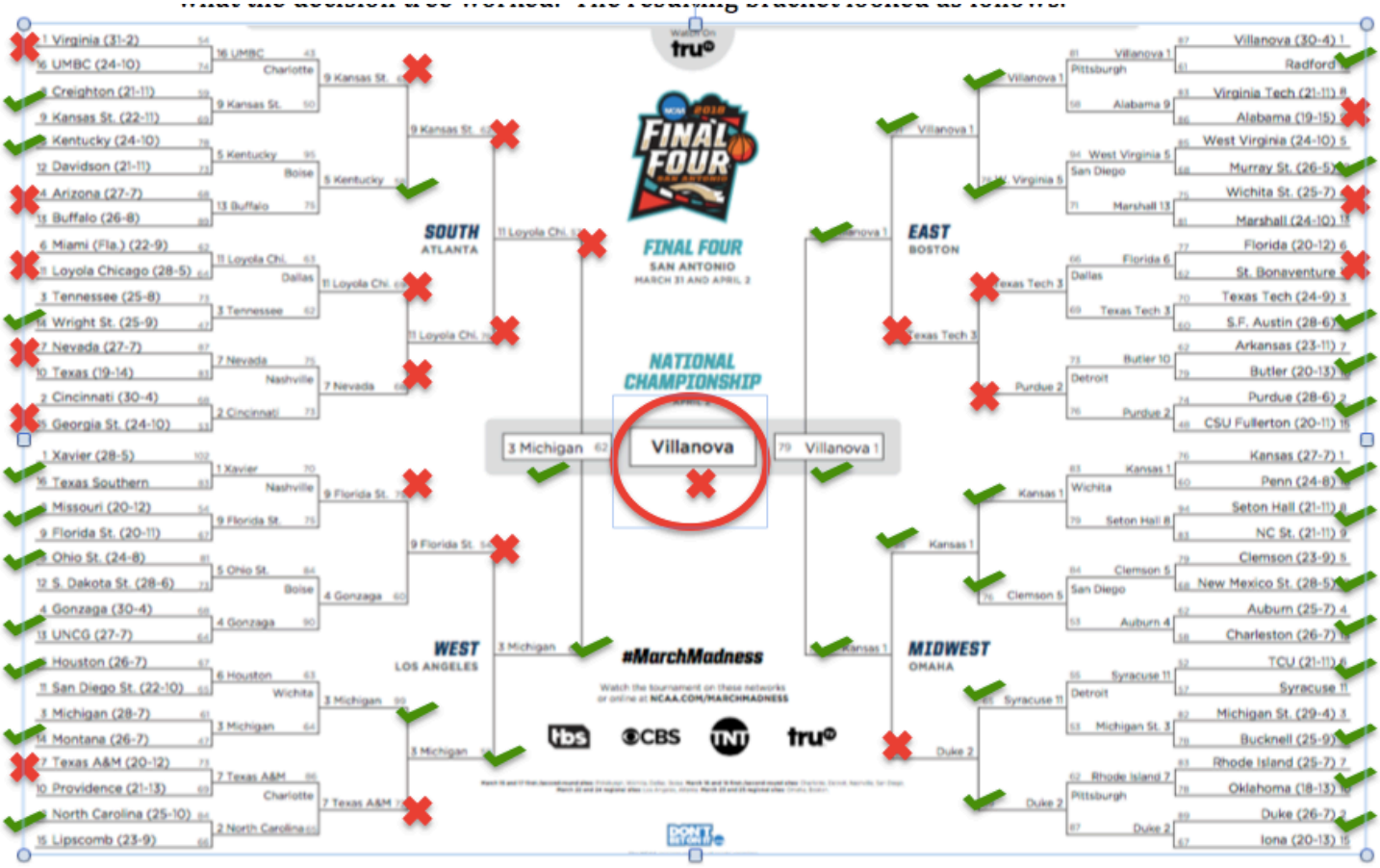
Measure	Training	Vali
Entropy RSquare	0.4860	
Generalized RSquare	0.6334	
Mean -Log p	0.3107	
RMSE	0.3130	
Mean Abs Dev	0.2199	
Misclassification Rate	0.1537	
N	410	

Confusion Matrix

Training

Actual	Predicted Count	
Who Won?	Higher	Lower
Higher	280	10
Lower	50	70

Overall, the model was set to use 81% of the original 504 data observations to train the data, and then the final 19% were used to validate and test the data. The model ended up only needing 12 layers, well short of the 50-layer max. One again, for a model type with so much variance and different network possibilities, an r-squared value of 0.49 is better, and for a decision tree, it's not bad. After creating the given model, data for this year was entered into the model, game-by-game, round by round, to see how what the decision tree worked. The resulting bracket looked as follows.



In this scenario, despite having an r-squared value that was higher, the decision tree predictive model predicted 39 out of 63 total games; or right around 62%. However, despite the *overall* selection percentage being similar to the

previous model, if the bracket were to have been entered in this year's "ESPN Tournament Challenge", it would have finished in the 92nd percentile nationwide in scoring out of over 17 million bracket entries (1070 points out of 1920), according to ESPN.com. This lower percentile finish was due to the decision tree's inability to predict the national championship winner correctly; the critical final game.

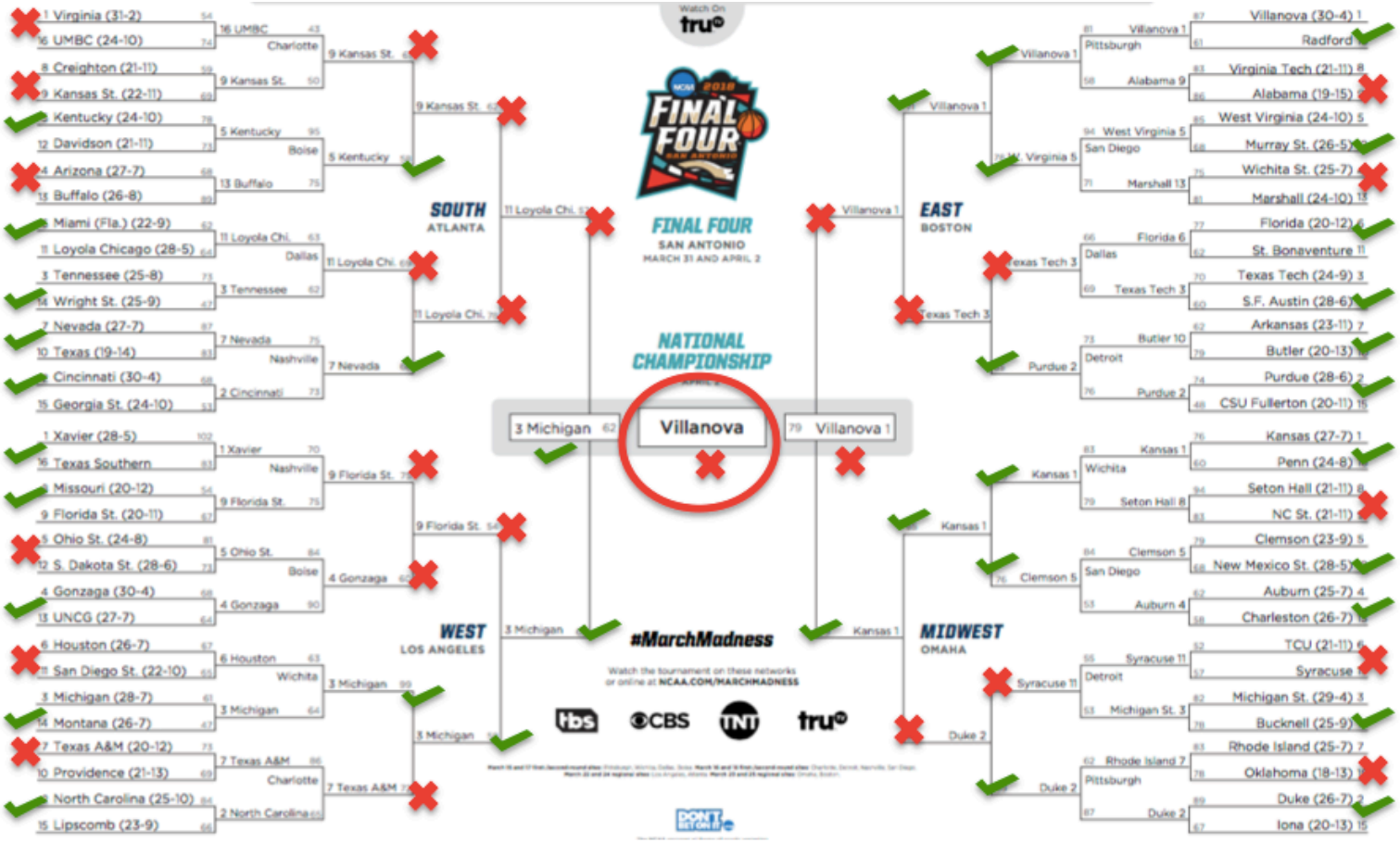
a. Naïve Bayes Model

In the final model, a Naïve Bayes predictive model was created. Similar to the other two models, the model uses a portion of the given data to train the model. Then, it uses the remaining portion to validate the model. To be specific, this model used 391 of the original rows of individual game differential data to train and produce the predictive model. In this type of model, instead of giving a relative "r-squared" value to represent the strength of the model, the Naïve Bayes model produces a "misclassification rate". The misclassification rate is how often the algorithm categorizes a game winner as the incorrect team; or, essentially, how often the model is wrong. This Naïve Bayes model split the data set into the most common parameters (which it does on its own), and continued to split the data from there into small sub-parameters, each with their own separate probabilities. In the end, the model finished with the resulting figures. Overall, a misclassification rate of 0.29, for a model that has so much variability, both in itself and its data, is good. However, in

Count	Misclassification Rate	Misclassifications
391	0.29412	115

Actual	Predicted Count	
Who Won?	Higher	Lower
Higher	205	69
Lower	46	71

this scenario, after data for this year was entered into the model, game-by-game, round by round, to see how well the Naïve Bayes model worked, the resulting bracket looked as follows, and this one was not as strong.



In this final model, despite having a misclassification value that was lower, the Naïve Bayes predictive model predicted only 36 out of 63 total games; or right around 57%. This percentage is hardly better than the overall random chance that you could get any game correctly, 50-50. Also, in this case, not only was the *overall* selection percentage lower than the other brackets, if the bracket were to have been entered in this year's "ESPN Tournament Challenge", it would have finished in the 80th percentile nationwide in scoring out of over 17 million bracket entries (830

points out of 1920), according to ESPN.com. Once again, this lower percentile finish was due to the decision tree's inability to predict the national championship winner correctly, as well as many other later round games.

VI. Comparison of Models/Conclusion

Of the three models created, the neural network performed the best in this years' NCAA tournament. With so many variables that can affect games and volatility in the tournament, it was exciting to see how well these models honestly did. It is clear that the neural network did the best, despite having the lowest r-squared value, because of its ability to predict the later round games correctly. This attribute in essential to have a strong bracket, and the neural network did a great job in this year's tournament with that. Overall, even without the ESPN percentile rankings, it would be clear that the neural network bested the other two models, followed by the boosted decision tree and Naïve Bayes models, respectively. The boosted decision tree would have been neck and neck with the neural network if it had picked the championship winner correctly; however, the Naïve Bayes model hardly had a chance because of its poor performance in the later rounds. This was not a complete surprise though; it is never completely reasonable to assume all

Method	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Percentage Correct	Percentile (ESPN)
Neural Network	22/32 = 69%	8/16 = 50%	4/8 = 50%	3/4 = 75%	2/2 = 100%	1/1 = 100%	64%	99.8th
Boosted Decision Tree	23/32 = 72%	8/16 = 50%	3/8 = 38%	3/4 = 75%	2/2 = 100%	0/1 = 0%	62%	92nd
Naïve Bayes	21/32 = 66%	9/16 = 56%	3/8 = 38%	2/4 = 50%	1/2 = 50%	0/1 = 0%	57%	80th

variables are independent, which is what the Naïve Bayes technique does. It makes sense for the Naïve Bayes model to have done the worst in this situation because of how related some variables may be in basketball. It is hard to not use certain statistics, when MANY statistical variables in the sport are somewhat related. In the end, it was very satisfying to see all three models doing well, especially the neural network. Even with the results of this project, there is no clear, right answer to "Which statistical machine learning technique is the best?" It really will always depend on the data set, variables, and parameters to see which one will perform the best. In this case, it was the neural network, but that may not be the story if another topic of data were to have been chosen.

Overall, studying these statistical machine learning techniques was very enjoyable, and even more so when applied to a topic as interesting as the NCAA Basketball Tournament. It is intriguing to research new techniques, start-to-finish, apply them, and get a finished product that performed as well as some of these did. I hope you enjoyed.

Sources:

- "How to Use Python to Select the Right Variables for Data Science." Dummies, www.dummies.com/programming/big-data/data-science/how-to-use-python-to-select-the-right-variables-for-data-science/.
- "What Is Machine Learning? A Definition." Expert System, 5 Oct. 2017, www.expertsystem.com/machine-learning-definition/.
- Seif, George. "Selecting the Best Machine Learning Algorithm for Your Regression Problem." Towards Data Science, Towards Data Science, 5 Mar. 2018, towardsdatascience.com/selecting-the-best-machine-learning-algorithm-for-your-regression-problem-20c330bad4ef.
- Ellentuck, Matt. "How Do Teams Get Picked for the NCAA Tournament?" SBNation.com, SBNation.com, 11 Mar. 2018, www.sbnation.com/college-basketball/2018/3/11/17098454/ncaa-tournament-selection-process-committee.
- Greenawalt, Tyler. "March Madness: Odds of a Perfect Bracket Are off-the-Charts Crazy; Here Are 7 Things More Likely." NCAA.com, 15 Mar. 2018, www.ncaa.com/news/basketball-men/bracket-beat/2016-03-14/march-madness-7-things-more-likely-happen-picking.
- "Has Anyone Ever Filled out a Perfect Bracket?" SI.com, www.si.com/college-basketball/2017/03/13/perfect-march-madness-bracket-possibility.
- Ellentuck, Matt. "How Do Teams Get Picked for the NCAA Tournament?" SBNation.com, SBNation.com, 11 Mar. 2018, www.sbnation.com/college-basketball/2018/3/11/17098454/ncaa-tournament-selection-process-committee.
- Cam_Newton. "The Argument for Ending Conference Tournaments." Mid-Major Madness, Mid-Major Madness, 27 Feb. 2018, www.midmajormadness.com/2018/2/27/17056636/get-rid-of-conference-tournaments-ncaa-basketball-march-madness-college-hoops-brackets.
- "Supervised and Unsupervised Learning." Dataaspirant, 9 Feb. 2017, dataaspirant.com/2014/09/19/supervised-and-unsupervised-learning/.
- "Glossary." College Basketball at Sports-Reference.com, www.sports-reference.com/cbb/about/glossary.html.
- Mannarswamy, Sandya. "Everything You Need to Know about Neural Networks." Open Source For You, 16 Mar. 2017, opensourceforu.com/2017/03/neural-networks-in-detail/.
- Shah, Jay. "Neural Networks for Beginners: Popular Types and Applications." Stats and Bots, Stats and Bots, 16 Nov. 2017, blog.statsbot.co/neural-networks-for-beginners-d99f2235efca.
- Sanjeevi, Madhu. "Chapter 4: Decision Trees Algorithms – Deep Math Machine Learning.ai – Medium." Medium, Deep Math Machine Learning.ai, 6 Oct. 2017, medium.com/deep-math-machine-learning-ai/chapter-4-decision-trees-algorithms-b93975f7a1f1.
- Dar, Pranav, et al. "A Complete Tutorial on Tree Based Modeling from Scratch (in R & Python)." Analytics Vidhya, 18 Apr. 2018,

www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/.

- Webster, Merriam. “Definition of Bayes Theorem.” Www.merriam-Webster.com/Dictionary/Bayes%27%20theorem, 2018, www.merriam-webster.com/dictionary/Bayes%27%20theorem.
- Waldron, Mike. “Naive Bayes for Dummies; A Simple Explanation.” AYLIEN, 20 Jan. 2017, blog.aylien.com/naive-bayes-for-dummies-a-simple-explanation/.
- Rastala. “Boosted Decision Tree Regression - Azure Machine Learning Studio.” Azure Machine Learning Studio | Microsoft Docs, docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/boosted-decision-tree-regression.
- Nielsen, and Michael A. “Neural Networks and Deep Learning.” Neural Networks and Deep Learning, Determination Press, 1 Jan. 1970, neuralnetworksanddeeplearning.com/chap4.html.
- Beam, Andrew. “You Can Probably Use Deep Learning Even If Your Data Isn't That
Big.” Https://Beamandrew.github.io/Deeplearning/2017/06/04/deep_learning_works.Html, 2017,

Appendix:

List of All Variables Used Before Eliminating Statistically Insignificant Ones:

- Win-Loss Percentage
- Simple Rated Score
- Strength of Schedule
- Points per Game
- Opponent Points per Game
- Field Goals per Game
- Field Goal Attempts per Game
- Field Goal Percentage
- 3-Pointers Made per Game
- 3-Pointer Attempts per Game
- 3-Point Percentage
- Free Throws Made per Game
- Free Throws Attempted per Game
- Free Throw Percentage
- Offensive Rebounds per Game
- Total Rebounds per Game
- Assists per Game
- Steals per Game
- Blocks per Game
- Turnovers per Game
- Fouls per Game
- Pace
- Offensive Rating
- Free Throw Rate
- 3-Point Attempt Rate
- True Shooting Percentage
- Total Rebound Percentage
- Assists Percentage
- Steal Percentage
- Block Percentage
- Efficient Field Goal Percentage
- Turnover Percentage
- Offensive Rebound Percentage
- Free Throws per Field Goal Attempts
- Tournament Ranking