# Final Report - Identifying March Madness Cinderellas

Andrew Moss, Zac Hatch, and Andrew Knox

December 2019

## 1 Abstract

The NCAA March Madness Tournament is among the hallmarks of annual American sporting events. A major reason for the massive public appeal of the tournament is the inherent unpredictability of its format; despite the best efforts of the selection committee, the tournament seeds are never as predictive of actual success as the casual observer would assume. Every year, so called "Blue Blood" programs which occupy the top of the Power Five NCAA conferences are toppled by smaller schools which win the hearts of many and impact the wallets of many others. Although it is often claimed that these dramatic upsets are nearly impossible to see coming, we set out to investigate whether these so-called "upsets" actually consisted of consistent, repeatable patterns such that one could successfully predict computationally what so many experts had missed through the use of the eye test. In order to do so, we scraped five years worth of tournament data from a combination of Sports-Reference.com, which provided the matchup history of the tournaments, and Kenpom.com, which provided a series of advanced statistics which were used to create a statistical profile of each team's offensive and defensive efficiencies, pace of play, and strength of schedule. When beginning the experiment, we established the definition of an upset as a victory by a team which was placed four seeds or higher below the team which was defeated. We treated the prediction as a binary classification problem, with the positive class indicating an upset, and the negative class a win for the higher seeded team. The data we compiled was then used to train four different binary classification algorithms: KNN, Gaussian Naieve Bayes, Decision Tree, and a Random Forest Model. The models were evaluated with precision as the primary metric for success, and the results indicated that KNN was by far the best classification method for the problem. In fact, out of 53 existing upsets in the data set, the optimized KNN algorithm successfully identified 50 of them, with a precision of 100%. Despite the success of KNN, we believe the model can be further refined in the future by training it on an even larger data set.

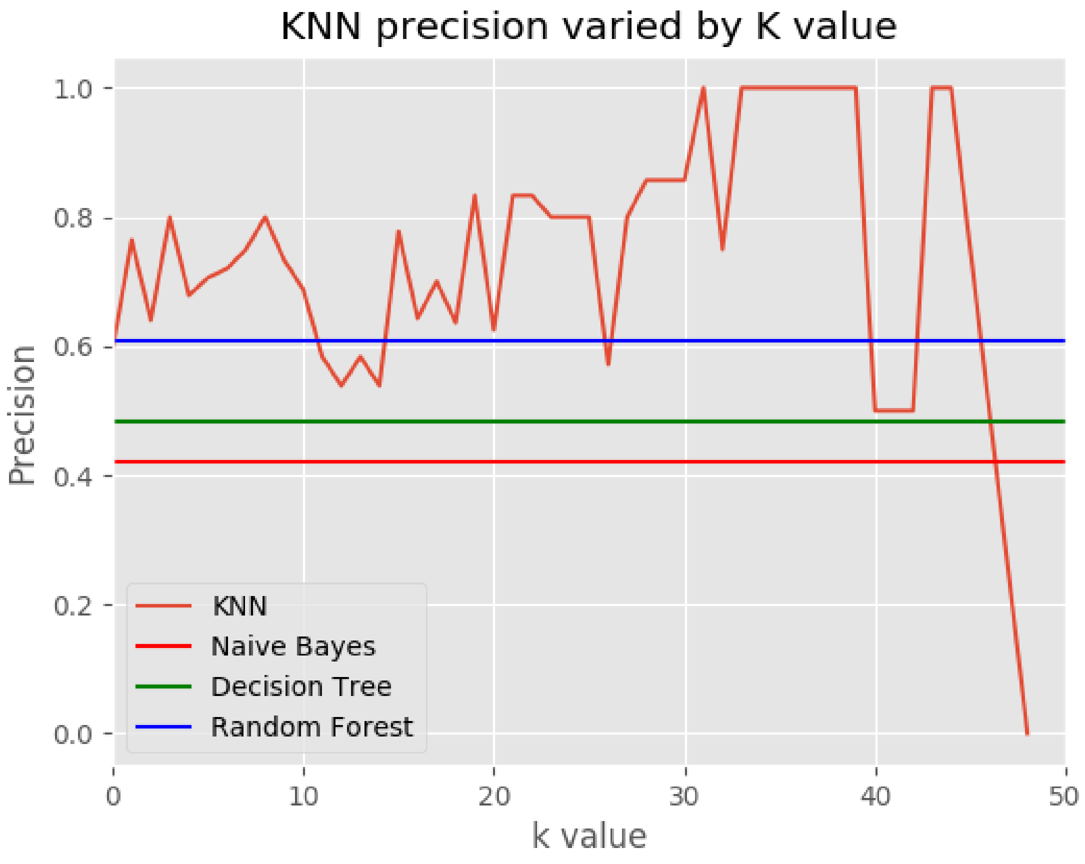
## 2 Background and Prior Work

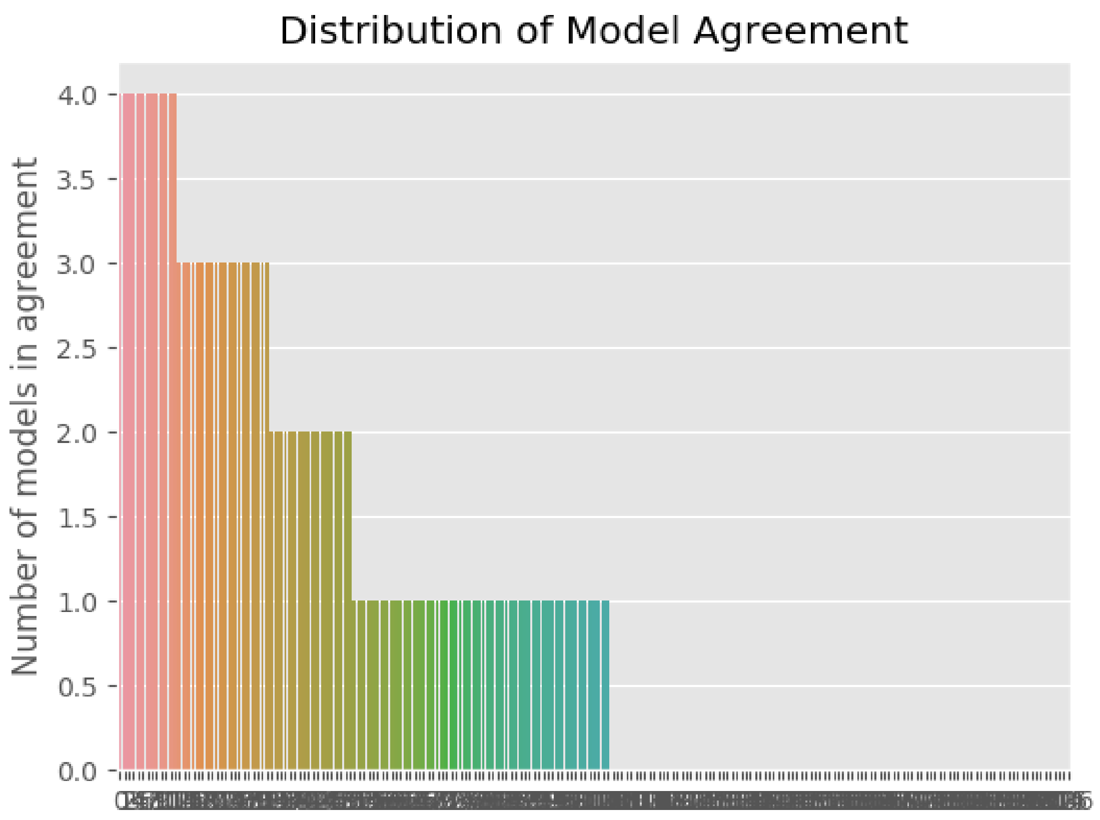
What makes this project different than a regular basketball spread calculator is that it only looks at the NCAA tournament games, and within the tournament, it only looks at games where the seed difference is greater than 4. However, ESPN has tried to identify small schools that stand a chance to win a game or two with their Giant Killer metric. In the metric, they describe using advanced stats and a distance-based metric that describes how "close" the two teams are to each other. This sounds similar to K-nearest neighbors, which is one of the approaches that we try with our data.
Another study done by Brian Mull, an analyst with NCAA.com, was done to identify the traits of likely Cinderella stories, and he found the following statistics the most important: Offensive efficiency, Defensive efficiency, turnover rate, and experience of the three top leading scorers. We use similar per 100 possession stats to the ones that Mull used, but we didn't include a variable for experience.

## 3 Data and Methods

The data on which the classifiers were trained consisted of the archived matchup data found on sports-reference.com, as well as the individual team advanced stats scraped from kenpom.com. The stats from kenpom were AdjEM, AdjO, AdjD, AdjT, Luck, OppAdjEM, OppO, and OppD. Each game was represented as a grouping of each of these stats for both the higher and lower seeded teams. AdjO is the number of points the team can be expected to score over 100 possessions against an opponent of average defensive strength as a measure of offensive efficiency. AdjD is AdjO's defensive counterpart, a measure of how many points the team can be expected to give up over 100 possessions to an average offensive team. The two of them are combined to form the adjusted efficiency margin (AdjEM), the difference between AdjO and AdjD, or the margin by which the team would be expected to defeat a team of average offensive and defensive strength. AdjT is the number of possessions a team plays per 40 minutes, the time of play in a college basketball game without overtime. The luck metric is the difference between a team's predicted winning percentage (based on their statistical profile) and its actual winning percentage in an effort to capture unexplained and/or intangible reasons for victory, especially in highly random scenarios arising at the ends of very close games. OppO, OppD, and OppAdjEM are the equivalents of AdjO, AdjD, and AdjEM respectively, representing the average values of these statistics of the opponents which that team faced throughout the year, intended to capture the strength of the schedule faced by the team. After each of these features was taken from kenpom, the data points were created using the matchup data from sports-reference, where the stats from kenpom were taken for each team and combined into a single matchup point with 2n+3 attributes, with n being the number of attributes associated with each individual team, and the three additional attributes associated with the matchup being the values of the high seed and the low seed, and the outcome of the game (all taken from sports-reference) in order to determine whether the matchup would be labeled an upset, which was defined prior to beginning the experiment as a victory by a team which was four seeds or higher below the team which they defeated.

Once the scrape was complete and the individual team data was combined into matchup data points in a single csv file, the csv was used to train the scikit learn implementations of the three binary classification algorithms which were covered in class (KNN, Naieve Bayes, and Decision Tree), in addition to a Random Forest, which was added after seeing that Decision Trees consistently out-performed Naieve Bayes, and yet always fell short of KNN. The performance of the classifiers was evaluated using precision, which was the ideal metric for the task, as upsets are inherently rare, and thus the proportion of correctly identified positives was what the models were built to maximize. KNN was implemented using cosine distance in order to standardize the scales of the attributes. The optimum value for K was searched for by running the algorithm with k = 1-50, with the results visualized on the elbow graph below. This range was selected based on the fact that numerous different K values within it resulted in perfect precision, with a dramatic drop-off right as k approached the total number of upsets. Due to the continuous nature of the data Gaussian Naieve Bayes was selected. When selecting the number of trees with which to build the Random Forest, it was determined that 100 trees resulted in the ideal trade-off between precision and computational complexity. In addition to reporting the precision of every model, the number of upsets which the model predicted in the data set was reported in order to compare to the actual value of 53. At the end of the experiment, a bar graph was constructed in order to visualize the level of agreement between the predictions of each model for each specific game, with a maximum value of four indicating that all four models agreed that the game appeared as though it would be an upset, and the precision of the combined agreeing models was calculated at all degrees of agreement. The width of the bars in the graph show the distribution of games at each level of agreement


KNN precision varied by K value
Precision
k value
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
KNN
Naive Bayes
Decision Tree
Random Forest



## 4 Results

Using the four classification techniques discussed in the methodology yielded valuable results for precision. The most promising classifier was the KNN algorithm, which reached 100% precision in numerous iterations of the script. Following KNN was random forest, which usually scored a precision percentage in the 60's. Decision tree averaged in the low 50's for precision percentage, and Naive-Bayes sat around 40% precision for most tests.

Initially, our implementation of KNN used Euclidean distance, and was averaging precision values in the 80's. However, after we realized that the model was likely being biased by the fact that our attributes were on different scales - some with a mean of zero and others with a mean of 100 - we modified the algorithm to use the cosine distance metric in order to standardize the individual effect of each attribute on classification. Afterwards, we discovered that KNN was becoming increasingly precise as the K-value was raised, and ended up with an optimal and consistent K-value of 32. This produced a consistent precision of 100%.

After obtaining the predicted class labels from each classifier for the data set, we made a visualization of classification confidence by counting how many times each instance was classified as an upset by distinct classification algorithms. The range was 0 through 4, symbolizing the amount of classifiers that agreed on a particular instance being an upset.

Since we used leave-one-out cross validation, we received very consistent results for the amount of predicted upsets generated by each classifier. Naive-Bayes predicted the most upsets at 88, followed by decision tree which predicted 54-55 upsets, then KNN with 50, and Random Forest with 36-38 predicted upsets. Interestingly, all 50 predictions made by KNN were upsets, and the algorithm only failed to predict 3 games as upsets.

## 5 Conclusion and Future Work

Historically, predictions of sports games tend to plateau at around 70 percent accuracy. However, with our KNN model, we were able to achieve 50/53 correct upset predictions, resulting in 100 percent precision and 98.5 percent accuracy. We believe these high results are due to two reasons:

Firstly, our goal was only to predict games that could be considered upsets. Games played between teams with very similar seeds, such as 1 and 2, are incredibly hard to predict because the teams are so close in skill level. We believe that our high numbers are a result of disregarding these types of closely-matched games, and believe this was how we were able to break the 70 percent accuracy threshold.

Secondly, our dataset was relatively small. We used data from the past 5 years, and only ended up with 207 data points. With a more robust data set, it's possible that our precision results would not be as high.

In the future, we wish to implement three changes:

First, we are going to scrape the current NCAA teams' data, to get accurate predictions for the upcoming March Madness 2020 tournament. Once the seeding results and team names have been released, we will scrape that data and perform unsupervised classification with these new data points.

Second, we are going to import more data from the past in order to have a larger dataset. This will lead to a more robust training set for each classifier, and we will be able to see if our high-precision results are consistent for larger quantities of data.

Third, we wish to implement another class label for classification, and analyze games that are not upsets. Currently, a value of 0 indicates a non-upset, and 1 indicates an upset. This is only for games that meet the criteria we defined, where the difference in seeds between each team was at least 4. We wish to implement -1, a game with criteria to be an upset, but is not; 0, a game that cannot be predicted because the teams are too close in skill; and 1, a game with criteria to be an upset, and is an upset.

## 6 References

https://www.espn.com/blog/giant-killers/insider/post/$_{/}id/945$
$https : //www.ncaa.com/news/basketball - men/2019 - 03 - 06/march - madness - three - traits - cinderella - must - have$
$https : //kenpom.com/index.php?y = 2015$
$https : //www.sports - reference.com/cbb/$