CMPT491 Data Mining and Advanced Database Topics 2019

Sports Betting in Men's Basketball

Graham Cooper, Richard Mills, Wamaitha Gitaka

MacEwan University

December 9, 2019

Abstract

This paper aims to forecast the outcome of the March Madness during the 2019 NCAA's Division 1 Men's Basketball Championships. March Madness is a college basketball tournament hosted by the National Collegiate Athletic Association generally during the month of March. March Madness is comprised of an original 32 teams that gain automatic entry through winning their conference championship. A remaining 36 teams for men that rely on the selection committee to award them a bid into the tournament which brings the total number of teams to 68. The term "March Madness" was coined during Illinois' statewide high-school basketball tournament in 1939 by Henry V. Porter, an official with the Illinois High School Association for the organization's in-house magazine. [2]

The March Madness datasets were collected from the Kaggle competition "Google Cloud NCAA ML Competition 2019-Men's". The goal of this competition is to apply machine learning to NCAA March Madness in order to forecast the outcome of all possible matchups through score predictions. Predicting the probability of one team beating the other team. These datasets included results of past tournaments including teams, seasons, locations, conferences, public rankings, play-by-plays, and supplements including 80+ years of historical and play-by-play data, from 90 championships and 24 sports. The size of the primary games data set is 34 columns and 87505 rows.

The analysis is performed using classification algorithms in machine learning including K-Nearest Neighbor, Support Vector Machines, Decision Trees, Ada Boosted, Gradient Boosting, Random Forest, Neural Networks, and Naive Bayes Classifier. The application software that were used included Microsoft Business Intelligence (MSBI), Microsoft SQL Server, TensorFlow, SciKit-Learn (SKLearn) and Python. Test data was built using Principal Component Analysis (PCA) and feature clustering. This test data was binned using clusters generated by MSBI. This data was then used to predict the winning team between any two team match ups, regardless if they had played against each other before or not.

Keywords
basketball, men's basketball, sports, march madness, betting

1 Introduction

An industry has developed based on statistical analysis services for any given sport, or even for betting behavior analysis on these sports. Data mining, the process of extracting hidden patterns from data, has become an intrinsic part of all professional sports throughout the world.

The record of sports events and statistics has been growing exponentially for the greater part of the last century. Basketball statistics used to be a luxury, available only to big professional teams. Statistics was a nightmare for the average coach because collecting and managing the data required a great deal of time and effort due to its manual nature at the time [3]. Today, basketball statistics has undergone a revolution. With the advent of the internet, data has quickly risen to the height of accessibility where sport-related data can be found easily and quickly, oftentimes in searchable form [5].

This data can be used for a variety of things and its popularity has trickled down to the fans who are now consuming more analytical content than ever. There are entire websites dedicated to the research and analysis of sports statistics and how they relate to a prediction in performance [6]. The website FiveThirtyEight.com is dedicated to crunching game analytics using a sophisticated system. NBA teams are now using a form of technology called "Player Tracking," which evaluates the efficiency of a team by an analysis of player movement [6]. Nike has created an entire line of products around taking previously unquantifiable athletic data and disseminating it wirelessly into easy-to-comprehend Apple-supported apps using sensors in cell phones, running shoes, bracelets and more to track movements and quantify data [4].

Today, sport prediction is very popular among fans around the world which has contributed to the expansion of sports betting. This makes the problem of predicting the results of sporting events, a new and interesting challenge. This paper attempted to tackle that challenge by predicting the winning team within the 2019 March Madness competition using data mining techniques. It correctly predicted the winners of about 65% of matches in the validation data set.

2 Materials & Methods

The data used for this analysis is from a Kaggle competition to predict the winner of the men's NCAA March Madness basketball bracket: https://www.kaggle.com/c/mens-machine-learning-competition-2019. It consists of a number of tables containing individual game statistics, team information, location information, player information, etc. Our analysis focused on game statistics. The game statistics are laid out with a winning team, a losing team, and the game statistics for each team including features such as: score, free throws, free throw attempts, etc. There are a total of 34 features. The games data set includes seasonal data and covering the years 2003–2019.

For our analysis we wanted to be able to predict the winner between a matchup between any two of the 357 teams. In order to achieve this we needed a model that could accept only two team ID's and then make a prediction. Our first step was to take the data and split the game statistics for each team and use those statistics to create aggregate data for each team for each season. For each feature in the original dataset, a new feature was created representing the average of that feature over each season. With the aggregate season table, an additional aggregate table was created to include aggregate data for each team across all

seasons. With the data in this format and given a team ID, the team's average game statistics could be looked up for an individual season or across all seasons. Next we constructed a training dataset for out models.

2.1 Data Cleaning

The training data set was created by first going through the original game statistics data set, consisting of 87k games, randomly splitting the winning and losing team into "Team 1" and "Team 2"; this resulted in Team 1 being selected as the winning team approximately 50% of the time. With the selected Team 1 and Team 2, each of their team aggregate stats were included in the data set and a label: "Win" was added to indicate whether or not Team 1 won the game: 1 for a win and 0 for a loss. Effectively the data was rearranged from the winning team versus the losing team with games statistics for each team to Team 1 versus Team 2 with average game statistics for each team plus an additional label to signify whether Team 1 won the game or lost the game. With the data in this format, we could then begin training a model that only required two team IDs to make a prediction. With a team ID, we could look up the team's statistics and attach them as features for the model to make a prediction. One additional dataset was created that, instead of having each team's statistics, instead had one set of statistics which were the delta of the two teams: we took Team 2's statistics and subtracted them for Team 1's and kept the difference as features.

2.2 Binning the Data Set

To bin the dataset we ran Microsoft Business Intelligence clustering algorithm to determine how the dataset should be binned. We went through each attribute

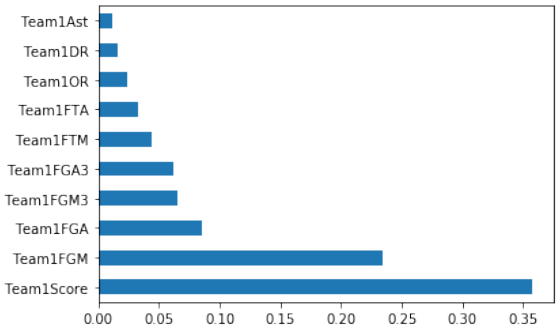


Figure 1: Top 10 attributes of variance explained for unbinned data set

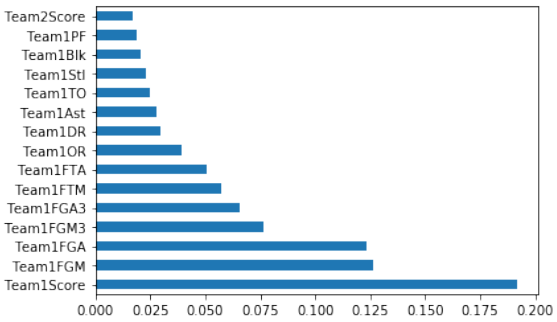


Figure 2: Top 15 attributes of variance explained for binned data set

and binned it according to the clusters that the algorithm discovered. The main idea in how we binned our data was to look at each distinct cluster and bin according to its mean +/- the standard deviation. This varied a little bit because some of the bins were left with few values; we attempted to keep the bins relatively equal in size.

2.3 Principal Component Analysis

We used Principal Component Analysis (PCA) to select the best features. We applied PCA to both our unbinned data and also our binned data. We had to drop the Team 1 and Team 2 attributes before running PCA because those two attributes explained 100% of the variance. After removing those two attributes, PCA found that about 5 to 10 attributes explained the majority of the

variance for the unbinned dataset, figure 1, and 10-15 for the binned datasets, figure 2.

It's important to note that PCA would find that Team 1's attributes were more important that Team 2's attributes of the same statistic even though the team that won the game was randomly selected to be either Team 1 or Team 2. We believe the reason for this is because the label "Win" indicates whether Team 1 wins or loses the match. For PCA's binned data, we selected the top 10 attributes in the bar chart with their team ID along with their mirror stats for Team 2. For PCA's unbinned data, we selected the top 9 attributes plus the team ID and Team 2 mirror stats.

2.4 TensorFlow Artificial Neural Networks

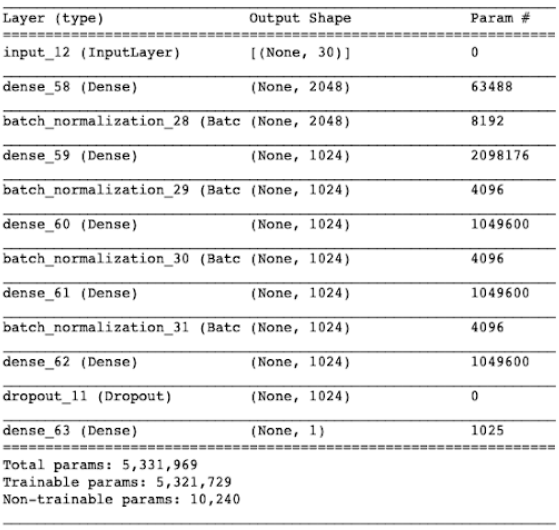
We originally were using Microsoft Business Intelligence, but because of the limited control we had over fine-tuning the artificial neural networks (ANN) we switch to using TensorFlow in Python that gave us more control to optimize and fine-tune the ANN.

We made numerous neural networks for this project experimenting with a variety of different structures, hyper parameters with different attributes from the dataset. We fine-tuned each of the ANN's to produce the best results for each of the models. Early stopping was used for each model and training was stopped after the next three epochs produced a higher validation loss than the one before it. The relu activation function performed the best on all models. Using a dropout layer of 20% right before the output layer gave the best performance for all models. Binary cross entropy was used as the loss function and the Adam optimizer was used on all models. Batch normalization improved the performance of almost all of the unbinned models, which provides regularization of the data passing between the layers by normalizing the batch distribution around the mean. Batch normalization was not used on any of the binned datasets because it reduced performance. The output layer of all the models used a Sigmoid activation function with only one output neuron to determine if Team1 won or lost the match. The same random seed was fixed so we could reproduce the results. The training size 65628 out of 87504 for 75% of the dataset with a validation split of 20% and the testing size was 21876 of 87504 for 25% of the dataset. The number of layers and the number of neurons in each layer varied for every model.

Because we made so many different ANN's and to keep the length of the paper shorter, we will only discuss the best ANN structure. This ANN uses three kinds of regularization: batch normalization, lasso regression set at 1e-6 and dropout of 20%. Lasso regression adds absolute value magnitude of the coefficient as the penalty to the loss function. The lasso regression is applied to every dense layer. In figure 2.4, the neural network's first layer is at the top and its last layer is at the bottom. The layer column is the type of layer in the ANN. The shape column is the number of neurons in that layer and the *param* column is the number of neuron connections made at that layer. This ANN is six layers deep, a width of either 2048 or 1024 neurons per layer and has batch normalization between each layer and a dropout layer right before the output layer of one neuron.

2.5 TensorFlow Convolutional Neural Network

We ran out of time making our convolutional neural networks (CNN); therefore, they were not optimized in any meaning-



Layer (type)	Output Shape	Param #
input_12 (InputLayer)	[(None, 30)]	0
dense_58 (Dense)	(None, 2048)	63488
batch_normalization_28 (Batc	(None, 2048)	8192
dense_59 (Dense)	(None, 1024)	2098176
batch_normalization_29 (Batc	(None, 1024)	4096
dense_60 (Dense)	(None, 1024)	1049600
batch_normalization_30 (Batc	(None, 1024)	4096
dense_61 (Dense)	(None, 1024)	1049600
batch_normalization_31 (Batc	(None, 1024)	4096
dense_62 (Dense)	(None, 1024)	1049600
dropout_11 (Dropout)	(None, 1024)	0
dense_63 (Dense)	(None, 1)	1025

Total params: 5,331,969
Trainable params: 5,321,729
Non-trainable params: 10,240

Figure 3: Artificial neural network architecture showing layers and parameter count.

ful way and are not a time series analysis. Our CNN uses a ReLu activation function, batch normalization between every layer, ridge regression on every layer, a Sigmoid activation function on the output layer, and a binary cross entropy loss function. The CNN is a ResNet model with 11 layers, 256 filters per layer and a kernel size of 3.

2.6 Microsoft Business Intelligence

In Microsoft Business Intelligence (MSBI) we ran all of the learning algorithms available, this includes Naive Bayes, decision trees, clustering, association rules, logistic regression, and a neural network. All the algorithms in MSBI used a training set of 61253 for 70% of the dataset and a testing set of 26251 for 30% of the dataset. All MSBI classification algorithms and association rules ran on binned data.

2.7 Machine Learning Classification Algorithms

Using Jupyter Notebook with a Python kernel and the Pandas, Numpy, and SKLearn libraries we conducted our analysis with the following classifiers: support vector machine (using a linear kernel), k-nearest neighbors, random forest, Ada boost, gradient boosting, and Gaussian Naive Bayes. Analysis were run on the data set of features for both teams and on the data set with team deltas. Both binned and unbinned data was used however unbinned data performed better than binned on all models. A test/train split was used with 70% of the data used for training and 30% used for testing. For comparison purposes, the same test/train data was used for all of the previously mentioned models. Analysis was also conducted using solely the team score, as identified by PCA as the most important feature, however this performed worse than using all features but did have the benefit of speeding up learning. An analysis was also conducted using an aggregate data set of only the latest season for each individual team, with the idea being that the latest season would be most representative of the team's current performance. However, this data set performed worse than the aggregate set of all teams.

3 Results

We ran a model with only team ID's and no other attributes and the model performed no better than chance. We also ran a model with team ID's and their scores and the best model accuracy was slightly under 60%. Adding all the features increased our best model's accuracy to slightly over 65.5%. The best models selected by PCA was 63.5%.

The most promising results can be seen in figure 3. These results were obtained

by running a number of different models on both binned and unbinned data as well as with all features, a subset of the features; as selected by PCA, and with the delta data set; the data set using the difference in team aggregation statistics rather than individual team aggregates. Accuracy was chosen as the validation score because it most closely represents what we are trying to achieve. The goal is to predict the winning team when given Team 1 and Team 2. The order of teams holds no significance except that the classified label applies to whether Team 1 won, that is to say, inverting would mean then that it applies to Team 2. Given this is binary classification and that the goal to rank machine learning as an effective means of predicting the winning team; accuracy best represents our goal.

The models and datasets that performed the best are shown in Table 3. As can be seen by the sorted results, the CNN was our best model and it was not optimized in any meaningful way. However, its accuracy score is only marginally better than the other accuracy scores and no model and data set combination achieved 0.66 accuracy. Also of note, in the best performing models and datasets, Table 3, all of the best performing data sets were unbinned; i.e. besides being rearranged and aggregated, no other data cleaning was performed.

Association rules didn't really find anything very interesting. Bins that were made with higher statistics in them were more likely to be associated with wins and teams with with lower statistics in their bins were more likely to be associated with loses.

4 Discussion

ANN, CNN, RNN, Ada Boost, gradient boosting, and support vector machines performed the best. This is not surprising because these learning algorithms are some of the most powerful algorithms available. In the vast majority of cases binning our data reduced our algorithms performance. However, binning our data set had the advantage of reducing the time needed to run some of the algorithms; perhaps unintuitively, it increased the run time of the support vector machine algorithm.

MSBI's learning algorithms had limitations that we did not encounter in TensorFlow and SKLearn. MSBI's learning algorithms couldn't run on continuous attributes, with the exception of the clustering algorithm, and all of our attributes were continuous data except the label. This required that we had to bin our data if we were to use MSBI's learning algorithms, but our algorithms performed better with TensorFlow and SKLearn unbinned. We had greater ability to fine-tune our learning algorithms in TensorFlow and SKLearn than we did with MSBI.

It is interesting to note that the best performing datasets always included using all the features rather than using only a subset of the features as chosen via clustering or PCA. This is most likely because even though those features best represent the variance in the data, the other features still contribute and the algorithms that we employed were powerful enough to utilize both make the best predictions. Adding more features to the model may improve the performance.

4.1 Improvements

There are some areas that we would have liked to improve on, but were unable because of time constraints. It could be beneficial to integrate the players' statistics into our data set. We believe adding more attributes to our dataset would have increased our algorithms performance. Another area would be to do a time series analysis of the matches with

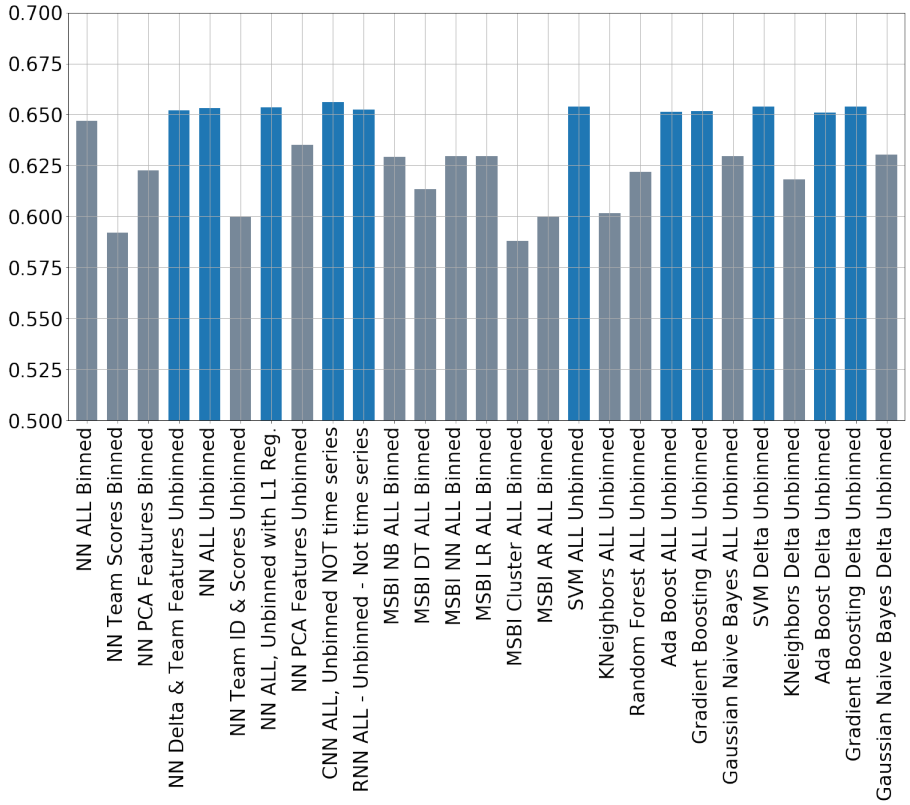


Figure 4: Model accuracy scores; scores greater than 0.65 are highlighted. Models ran with all features or a subset of features chosen by cluster or by PCA. Includes both the team aggregate data set and the delta data set.

deep learning. We would have used total average for each team that we currently are using then add the last x number of recently played games or seasons from that point in time. This would have allowed us to incorporate more deep neural networks and optimize our CNN and RNN models that have the potential to learn much better than artificial neural networks. Deep learning has significant advantages over other learning algorithms. One of these benefits is that very large dataset won't suffer from overfitting like other learning algorithms suffer from. Another benefit is that they both can be used in time series analysis. The simple fact that our CNN performed best and we didn't have time to optimize it and it doesn't currently perform time series provides significant evidence that these improvements mentioned above would be valuable to use in future models.

Another area of improvement could be to further fine-tune the bins for the data. We found that our binned data, with bins selected from feature clustering in MSBI, performed worse than the unbinned data. This signifies that, with binning, we are losing some granularity that our algorithms are learning as significant. Ideally, we would have constructed bins that contained the granularity that was necessary for high accuracy scores while smoothing out outliers and noise that negatively affected accuracy. While this would be a nice improvement, achieving it is a time consuming process as we can only see the effect of ours binned data after binning and running the machine learning algorithms to see the results.

Table 1: Best performing models and datasets with their accuracy score in descending order

Model and dataset	Accuracy score
CNN ALL, Unbinned NOT time series	0.6559
SVM Delta Unbinned	0.6540
Gradient Boosting Delta Unbinned	0.6537
SVM ALL Unbinned	0.6537
NN ALL, Unbinned with L1 Reg.	0.6533
NN ALL Unbinned	0.6532
RNN ALL - Unbinned - Not time series	0.6523
NN Delta & Team Features Unbinned	0.6521
Gradient Boosting ALL Unbinned	0.6517
Ada Boost ALL Unbinned	0.6514
Ada Boost Delta Unbinned	0.6508

Conclusions

We conducted an analysis of predicting men's NCAA match outcome using a variety of data mining techniques. We cleaned our data transforming it from 87k games of winner versus loser to 87k games of Team1 versus Team 2 with accompanying team statistics, as well as a second set that contained the difference in statistics between the two teams. These data sets included a new label to classify whether or not Team 1 won the game. With these data sets we tried binning using clustering from MSBI and feature reduction using PCA. With our data sets ready, we used neural networks, MSBI analysis, and classification algorithms from the SKLearn library to construct models to predict the winner of the match. We found that the neural network, Ada boost, gradient boosting, and support vector machine performed the strongest; with only marginal differences in their predictive accuracy. Overall, we achieved 65% predictive accuracy in determining the winner of a match given any two team ID's.

For sports betting, we would not recommend utilizing our models as we do not believe that the success rate would be high. The outcome of a game relies on many, many more factors than we were able to include in our models and data sets. And while we achieved a better than random guessing accuracy; randomly guessing should result in approximately 50% accuracy, we achieved 65% accuracy. This accuracy is not likely high enough to overcome the odds to be successful financially in the sports gambling world.

We do believe it is possible to successfully use data mining to analyse and predict matches. Our model shows potential and if our recommendations for improvements are incorporated into a model then it may have significant improvements in predictive accuracy.

References

[1] Bunker, R., & Thabtah, F. (2019). A machine learning framework for sport result prediction. *Applied Computing And Informatics, 15*(1), 27–33. doi: 10.1016/j.aci.2017.09.005

[2] Koerner, B. (2019). Why is it called "March Madness"?. Retrieved from https://slate.com/news-and-politics/2010/03/why-is-it-called-march-madness.html

[3] Manovska, Lj Stamatoski, A. Lacmanovic, Izabela Pecev, Predrag Srecković, M. (2012). Potentials of using data mining in basketball coaching.

[4] Newcomb, T. (2019). 10 Companies On The Cutting Edge Of Sports Data. retrived from https://www.fastcompany.com/1671570/10-companies-on-the-cutting-edge-of-sports-data

[5] Schumaker, R., Solieman O., & Chen H. (2010). *Sports data mining*. New York: Springer.

[6] Steinberg, L. (2019). CHANGING THE GAME: The Rise of Sports Analytics. Retrieved from https://www.forbes.com/sites/leighsteinberg/2015/08/18/changing-the-game-the-rise-of-sports-analytics/#5ffab20c4c1f