Men's March Madness Tournament 2022 ML Project

Spencer Duncan and Geoff Johnson

EECE 397B - March 16, 2022

Introduction

Not all problems that are trying to be solved have the weight of the future balancing on them. While yes, the advances of Machine Learning (ML) in areas like image recognition, speech translation, and medical advances are important, solving a problem that has yet to be solved has merits of its own. This is the case for our project in trying to correctly predict all the winners of the largest basketball tournament in the US. Each March, 64 college basketball teams face off and compete for the title. Odds makers, sports personalities, academics, and even billionaire businessmen have taken interest in predicting who the winner will be. With odds of 9.2 QUINTILLION to one [1] of correctly predicting all the games, this is a problem worth solving just due to its incredibly small odds, not to mention no one has ever picked every game in a single tournament correctly. Our goal for this project was to use the historical data provided by Kaggle.com to train a ML algorithm capable of outputting the odds of each team winning against any other team in 2021 and comparing our model against the known winners of last year's games to check our accuracy.

Dataset and Features

Out of the many datafiles that are associated with the competition, we chose two that contained enough variables to work with but not so many to be overwhelmed when developing our training data. One of our datafiles contains ID numbers for all the basketball teams and their winning/losing game scores for the regular season. The other contains the seed round information on each team. Teams play more than one game in the season, so we needed to aggregate all the scoring information

for each team. We were then able to start calculating some of our own stats for each team over every season. Examples include a team's win ratio, average score margin, and points ratio. The datasets available to us contained information dating as far back as 1985 but with player turnover being about four years, much of that data is irrelevant to current team metrics. We decided to only calculate our statistical values with data from the 2015 season and forward.

Much of our work involved creating data frames that contained all our data in coherent and meaningful sets. We ended up creating team statistics for each team twice. Once for the team being classified as a "winning" team and the other as the team being classified as a "losing" team. This allowed us to double the amount of potentially meaningful predictors for each team ID and make sure we did not accidentally leave out any teams. Once the calculations were done on the winning and losing statistics for each team, they were renamed from 'W' and 'L' to more ambiguous classifications of 'A' and 'B'. From there, we calculated more statistics such as an average score margin difference between team IDs, point ratio differences, and seed value differences. This was done using some of the statistics we had already calculated. The most important metric is the score-margin which is used as a binary variable to represent the output variable of the training model. If the score margin was positive, team A would receive a '1' for winning. If the score margin was negative, team A lost against team B and receives '0'.

Once we developed and completed the data processing procedure for the training set, we repeated the same process with the test data. Here, we chose to test the tournament data against the regular season data since the information was already split into two sets and contained identical forms of information for us to manipulate. We used the team-win binary variable (stated previously) from all the years except the 2021 season as the training output since that is the year, we want our model to predict.

Methodology

There are many ways to tackle this classification problem but based on our studies, we know a variant of decision trees could be a viable candidate for generating a model. We elected to stay away from a single decision tree as there are better options out there such as random forest implementations. Using our knowledge of boosting mechanisms, we wanted to add this mechanic

to our random forest implementation. Having some basic knowledge of the way that Sklearn's boosted gradient descent algorithm works we began looking at trying to implement this with our data. In our research of boosted gradient, we came across XGBoost. XGBoost is fundamentally the same in its methodology of performing boosted gradient decent but where XGBoost shines is in the implementation. XGBoost was developed for optimization improvements and is becoming popular with many users on Kaggle. It will yield the same results as Sklearn's boosted gradient decent but in a shorter amount of time. Due to the large number of data points and of parameters we used, this is the algorithm and library host we chose to use for this project. We were happy at how fast it worked through our input data and provided results for all the possible team matchup combinations and their win probabilities.

Experiment/Results

The results from our model are a CSV file which contains all the possible tournament matchups between every team and the probability prediction of the first team beating the second. The display format is as follows, 'year_teamID1_teamID2 predicted_win_of_teamID1'. We then went through and checked all the probabilities for the teams that participated in the 2021 basketball tournament and compared our model's prediction to the actual results. We did this by hand and now realize it would have been a good application for additional code to search the CSV and pull out the relevant information. However, in the end, our model predicted 41 out of the 64 games correctly, giving our model an accuracy of 64%.

One aspect we observed from our model's results is the number of abnormal seed-based upsets in this year's tournaments. As we included a team's seed as a component of our training data, our model did not predict most of the surprising upsets where a large seed value wins over a small seed value. However, our model did correctly predict two upsets where a 14 seed beat a 3 seed and an 11 seed beat a 2 seed. While this looks good on the surface, it would be misleading if we did not point out that it incorrectly predicted a 10 over 2, 15 over 3, and 11 over 1. These results are why some researchers are suggesting that using the seed data to train your algorithm is actually hurting the results more than helping [1]. They attribute this to other non-qualitative factors at play such as a teams player experience, the inherent pressure on players to perform better during

the tournament, and how well a teams fan base travels.

While our model did predict two big upsets correctly, we feel this would be a good candidate for further investigation moving forward; we do not feel that the seed data is completely without merit and is why we included it in our data. As far as how well our results performed, we believe this was not a good year to test on. According to the NCAA there was not a single bracket still in contention after the first round of games, the fastest disqualification in the search for a perfect bracket ever [2], due to the abnormally high number of high seeds losing in the first round.

Conclusion

We are very pleased with our exploration and met our goal of producing an output in a form that could be submitted to the Kaggle competition. Due to the large amount of data, that is spread across a variety of files, we had to explore and manipulate into something we could work with, it left us with little opportunity to explore different machine learning models outside of our boosted gradient decent. We do not know how our model's results compare against others and it is likely that we would need to restructure our data to implement some of the models we are more familiar with from lab assignments. Our model currently achieved an accuracy of 64% and further work can be done to try to improve that accuracy. Calculating additional statistics, using different host libraries, and experimenting with different models are all next steps that could be taken to try to improve model accuracy.

References

[1] J. Gumm, A. Barrett and G. Hu, "A machine learning strategy for predicting march madness winners," 2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015, pp. 1-6, doi: 10.1109/SNPD.2015.7176206.

[2] Ncaa.com, 2022. [Online]. Available: https://www.ncaa.com/live-updates/basketball-men/d1/last-perfect-ncaa-brackets-bust-maryland-beats-uconn. [Accessed: 13- Mar- 2022]