Kimberly Liu & Isaac Tabor

DS 3001 Final Project Paper

5/6/2025

Can predictive models trained on regular-season performance differentials improve forecasting accuracy in NCAA tournament matchups?

## Abstract

This project explores the use of machine learning to predict NCAA March Madness basketball outcomes using a comprehensive dataset of historical team performance and game results. Leveraging data from the 2003 to 2024 seasons, we built a supervised binary classification model using XGBoost, with the goal of estimating the probability that a given team will win a specific tournament matchup. We also used an XGBRegressor, which is typically used for regression tasks; however, we used the log loss metric, which aligned with the classification nature of our study, and our outputs predicted probabilities of one class (Team A winning), and we used a threshold of .5 to classify each game into the appropriate binary class (Win vs. Lose).

Each observation in our dataset represents a single game, enriched with detailed team and opponent statistics aggregated at the season level. To capture matchup dynamics more effectively, we engineered differential features, such as the difference in win percentage, shooting efficiency, and rebounds, between competing teams.

To evaluate our model, we split the data temporally, training on seasons prior to 2025 and validating on more recent years. We used accuracy as our primary evaluation metric, supported by logloss, ROC AUC and confusion matrices. We trained the model using a low learning rate

and a fixed maximum of 1,000 trees to balance learning capacity and overfitting risk. We also used a GroupKFold cross-validation strategy to ensure robust seasonal generalization. Our final model achieved an average CV accuracy of .6568 and on average was outperformed by our "chalk" benchmark that had a baseline accuracy of .69744. The "chalk" benchmark predicts winners based solely on seed ranking.

Our findings show that even with just box score statistics, our machine learning model was unable to outperform a purely seed ranking tournament prediction strategy. Future improvements could include exploring including a rankings differential to the features we used, removing unhelpful features we used, adding possession-based metrics, advanced efficiency ratings, or combining models through ensembling. Overall, this work explores the value of structured feature engineering and thoughtful validation in sports analytics.

## Introduction

In this paper, we will first provide an overview of the data used to train XGBoost models to predict NCAA tournament outcomes based on regular season data. Additionally we will discuss how we carefully merged and formatted the data because this was crucial to ensure that we were predicting the right thing. For example, it was key to ensure we were predicting NCAA Tournament games and not regular season games, and doing so without any data leakage. That is, we did not want to include any NCAA tournament data accidentally in our predictions for NCAA tournament outcomes because the model would then be "seeing" the outcome before making the prediction. We will also provide sample tables of what our final data looked like in order to be ready to train the model and make predictions.

Then, we will discuss the methods we used to build our predictive model. For example, we will discuss features used, hyperparameters chosen, and how we evaluated our model's performance against seed-based predictions.

Lastly, we will discuss the results of the 2025 predictions, how well our model predicted 2025 March Madness games, notably compared to mere seed-based predictions, and discuss future areas of potential to improve and advance our work.

## Data

Our dataset contains historical NCAA men's basketball game results from the 2003 season through early 2025, compiled from a comprehensive Kaggle repository. These data originate from multiple sources, including Kenneth Massey and Jeff Sonas, and span both regular season and postseason games. We focused on the men's dataset (denoted by M prefixes), using key files such as MTeams, MSeasons, MRegularSeasonDetailedResults, and MNCAATourneyDetailedResults. After filtering to include only teams active since 2003, we merged relevant files to form a unified game-level dataset with over 119,000 observations and 38 variables. Each row represents a single NCAA game and includes team identifiers, scores, box score statistics, game type (e.g., regular season or NCAA tournament), and contextual features such as location and conference.

To prepare our data for modeling, we first focused on regular season data at the game-level unit of observation. Then, we aggregated the data to the season-level for each team. In order to aggregate the data to the regular season-level for each team unit of observation, we had to format the game-level data to a new unit of observation such that there was one row for each team's game win or loss. For example, before UVA versus Duke Feb 9, 2019 had one row.

After reformatting, it had 2 rows, one for UVA documenting the loss and one for Duke documenting the win. Then, we could aggregate that data for season-level data for each team such as having statistics like average point difference, win percentage, average field goals made per game, for the entirety of each regular season from 2003 to 2024.

Then, we had to prepare the NCAA Tourney Data by adding the seeds for NCAA tournament games, keeping this data at the game-level unit of observation. We merged this data with the aggregated regular season data to achieve a dataset that has each NCAA Tournament matchup since 2003 for each team along with their regular season statistics and their opponents' regular season statistics. It also contained the game result for each tournament matchup. Below is a table generally showing what this data looks like, though note, it only includes average score differential as a key regular season-level statistic for brevity.

**Sample Table of Final Data Structure (Before Matchup Differentials)**

| Season | TeamID | Opp TeamID | Team Score | Opp Score | Win | Avg Score Diff | Opp Avg Score Diff |
|---|---|---|---|---|---|---|---|
| 2024 | 1181 | 1301 | 85 | 78 | 1 | 12.406 | 9.680 |
| 2024 | 1397 | 1345 | 64 | 76 | 0 | 11.593 | 13.242 |

The final dataset we constructed reflects a rich set of engineered features useful for modeling March Madness outcomes. While the original repository contained over 36 CSV files, we focused only on those most relevant to team performance. We now have a streamlined yet

information-dense view of NCAA basketball performance over two decades, which will enable us to train a machine learning model to predict tournament results based on historical patterns of success, scoring efficiency, and other performance indicators.

## Methods

As aforementioned, to build a predictive model for March Madness outcomes, we began by compiling and merging over a dozen NCAA datasets, including game-level tournament results, team statistics, tournament seeds, and team metadata. Our unit of observation was each individual matchup, represented twice: once from the perspective of the winning team and once from the losing team. We engineered features to capture each team's average season-long performance, such as shooting efficiency, rebound and turnover rates, and scoring margins.

Our core modeling strategy used XGBoost, a gradient boosting algorithm known for its strong performance on structured data. We framed the problem as supervised binary classification: for each matchup, the model predicts the probability of the team winning. Rather than relying on raw statistics, we computed matchup-specific differentials, the difference between a team and their opponent's season-long averages, so the model learns relative strengths rather than absolute values.

**Sample Table of Final Data Structure (After Matchup Differentials for Predicting)**

| Season | TeamID | Opp TeamID | Avg Score Diff Diff | Win Percent Diff | Avg FGM Diff | Avg FGA Diff | AvgRebounds_diff |
|---|---|---|---|---|---|---|---|
| 2024 | 1181 | 1301 | 2.726 | -0.151 | 2.279 | 5.086 | 2.518 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2024 | 1397 | 1345 | −1.649 | 0.237 | 4.806 | 9.264 | 2.336 |

To evaluate performance and prevent overfitting, we used a GroupKFold cross-validation strategy grouped by season. This allowed us to hold out entire seasons during training and better simulate out-of-sample generalization to new tournaments. We trained the model using a low learning rate and a fixed maximum of 1,000 trees to balance learning capacity and overfitting risk. Beyond accuracy, we assessed our model using log loss and ROC AUC.

In comparison to a “chalk” baseline, which assumes the better seed always wins, our model on average underperformed. The final tuned XGBoost model achieved an average cross-validation accuracy of 64.67% compared to a chalk baseline average accuracy of 0.6974. This suggests our feature set captures meaningful patterns in team performance. Still, future improvements could involve incorporating tempo-adjusted or possession-based metrics, and using model ensembling to better account for uncertainty in close matchups.

## Results

As aforementioned, we evaluated our XGBoost models on NCAA tournament outcomes using a rich dataset of team-level regular season statistics from historical game performance. Our XGRegressor models were trained with a low learning rate and a fixed maximum of 1,000 trees to balance learning capacity and overfitting risk achieved an average cross-validation accuracy of 64.67% compared to a chalk baseline average accuracy of 0.6974.. This indicates decent predictive power almost on par with a simple "chalk" model, which assumes the better-seeded team always wins. Some years, XGBoost did outperform the chalk model, for example, in 2018, our model had an accuracy of 75.78% compared to baseline accuracy of 67.16%, other notable

years were 2003, 2006, and 2024. However, the chalk baseline generally outperformed our model. This is a great example of how more complex modeling is not always better than simple models.

For 2025 predictions, we first imported a dataframe for the seeds of each team in the 2025 NCAA Tournament including which region they were in (e.g. “W01 for West Region 1 seed). Since we would not know which teams would face which teams in each round, we created a dataframe of every single matchup possible my merging the original dataframe on itself and taking out games where a team would play itself since that would be impossible. We had then has to recreate our dataframe to have regular-season statistics for each team, and merge the data frames to create our final dataframes with our feature columns.

Then, we had each model that held out one season each time, predicted the probability of each team winning each 2025 possible matchup game, and took the average of predictions.

Then, we simulated the bracket adapting some of the code used from this reference using our aggregated predictions. Below are the teams predicted by our XGBoost for each round.

**Round of 32:**

| **East** | **Midwest** | **South** | **West** |
|---|---|---|---|
| 1 — Duke | 1 — Houston | 1 — Auburn | 1 — Florida |
| 2 — Alabama | 2 — Tennessee | 2 — Michigan St | 2 — St John's |
| 3 — Wisconsin | 14 — Troy | 3 — Iowa St | 3 — Texas Tech |
| 4 — Arizona | 4 — Purdue | 13 — Yale | 4 — Maryland |

| | | | |
|---|---|---|---|
| 12 — Liberty | 5 — Clemson | 12 — UC San Diego | 5 — Memphis |
| 6 — BYU | 6 — Illinois | 6 — Mississippi | 6 — Missouri |
| 7 — St Mary's CA | 7 — UCLA | 7 — Marquette | 7 — Kansas |
| 8 — Mississippi St | 8 — Gonzaga | 8 — Louisville | 8 — Connecticut |

**Sweet 16:**

| **East** | **Midwest** | **South** | **West** |
|---|---|---|---|
| 1 — Duke | 1 — Houston | 1 — Auburn | 1 — Florida |
| 7 — St Mary's CA | 2 — Tennessee | 7 — Marquette | 7 — Kansas |
| 6 — BYU | 6 — Illinois | 3 — Iowa St | 3 — Texas Tech |
| 4 — Arizona | 4 — Purdue | 13 — Yale | 4 — Maryland |

**Elite 8:**

| **East** | **Midwest** | **South** | **West** |
|---|---|---|---|
| 1 — Duke | 1 — Houston | 1 — Auburn | 4 — Maryland |
| 6 — BYU | 2 — Tennessee | 3 — Iowa St | 3 — Texas Tech |

**Final Four:**

| **East** | **Midwest** | **South** | **West** |
|---|---|---|---|
| 1 — Duke | 1 — Houston | 1 — Auburn | 4 — Maryland |

**Championship:** 1 — Houston **versus** 4 — Maryland

**Champion:** 4 — Maryland

Unfortunately, these predictions were only 63.49% accurate compared to an accuracy of 76.19% if you just picked the higher seed each time. March Madness was particularly chalky this season. We could play around more with features for example to try to improve our model in the future.

Also, our model predicted 1 of 2 championship teams, while chalk predicted 0 of 2 (the top overall seeds were Auburn and Duke respectively). In the ESPN Men's Tournament Challenge bracket scoring, a correct pick in the Round of 64 is worth 10 point, in the Round of 32 it's 20 points, Sweet 16 is 40 points, Elite Eight is 80 points, Final Four is 160 points, and the Championship is 320 points, so there is no guarantee that our predictions would have lost to chalk in a tournament bracket competition.

## Conclusion

Future work that can be done with these results includes further investigation into what features may be best for predicting tournament outcomes. For example, one could include rankings that were done throughout the season into the features. It would also be interesting to do analysis into what features XGBoost is currently weighing the heaviest right now. This could potentially improve accuracy against the chalk.

In addition, one could play with hyperparameters to try to achieve the best outcome with respect to underfitting and overfitting. By this, I also mean experimenting with how many

features are included in the features to predict game outcomes. We currently included a lot of differencing features, and in this case less could be more.

If there were some way to incorporate key injuries, I would be very interested in that.

One could also experiment with other models to use. For example, one could experiment with logistic regression, decision trees, and/or random forests.

Lastly, it could be interesting to evaluate models not only on their accuracy in terms of most correct picks, but also based on Tournament Challenge bracket scoring such as the ESPN one described earlier. Brackets are typically scored on these types of scales where predicting future rounds correctly results in higher scoring, so merely choosing the most correct may not be the best way to evaluate performance, especially since you can usually win money based on these challenges.

In sum, a lot of work can be done to try to improve predicting March Madness outcomes and beating chalk. It is a difficult task as March Madness is known for its unpredictability.

# References

Haupts, Lennart. Simulate n Brackets. Kaggle, 2024, https://www.kaggle.com/code/lennarthaupts/simulate-n-brackets.

Kaggle. March Machine Learning Mania 2025. 2025. Kaggle, https://www.kaggle.com/competitions/march-machine-learning-mania-2025/data.

Mulla, Rob. March Madness Predictions with Data Science. YouTube, 12 Mar. 2022, https://www.youtube.com/watch?v=cHtAEWkvSMU.