An Analysis of NCAA DI Men's Basketball Team Success

Kelvin Sun, Kylan Lim-Hanna, Peter Emanuel

1 Abstract

We develop a series of models for predicting NCAA DI Men's Basketball Tournament performance, given an array of numerical and categorical features. We first use XGBoost to predict NCAA Tournament seeding, as if we are standing before seeds are announced. We then use K-Nearest-Neighbors to predict NCAA Tournament expected round placement, given that the seeding has already been determined but the tournament has not yet started. We define expected round placement as the final round a team will make it to in the tournament before a loss. Finally, we use K-Means Clustering to group the entire field into long-term tiers.

2 Introduction

2.1 Motivation

It has become increasingly difficult to predict NCAA DI Men's Basketball Tournament placement in recent years. Mid-major teams have outperformed, especially in the earlier rounds, and the level of play has become more even across conferences. Furthermore, institutional mathematical model-driven predictions are opaque, preventing the public from knowing exactly how to attribute teams' success to specific performance metrics. Our analysis will allow viewers to access and understand the relationship between these performance metrics, such as offensive efficiency, pace of play, and turnover rate, and team success. We define team suc-

cess using predicted seeding, expected round placement, and long-term program tier. Having a predetermined seed and round placement in mind can help viewers make more informed predictions and bets. Coaching staff can also use these findings to learn what to improve on. Finally, prospective recruits can see a clear, quantitative picture of their potential future team's style of play and where that program stands compared to other programs, helping them decide where they fit best.

2.2 Dataset, Feature Space, and Preprocessing

Our dataset, pulled from Kaggle, consists of NCAA Division I Men's Basketball team season statistics, containing a total of 58,920 data points, with 24 variable columns and 2,455 rows. The dataset contains results from years 2013-2021; however, we have decided to omit the 2020 year due to lack of postseason data given the COVID-19 Pandemic. The numeric feature space contains statistics such as number of games played, number of games won, adjusted offensive and defensive efficiency, turnover rate, and various field goal percentages. We convert conference into a categorical variable via one-hot encoding. However, we condense the conference categories, categorizing any team not in the Big 10, Big 12, PAC-10, ACC, SEC, and (former) Big East conferences as "MIDMAJOR". We normalize all numeric features for K-Nearest-Neighbors and K-Means. We do not randomize data at all during this study. Each season produces an equal number of 1 through 16 seeds and an equal number of teams exit in a given round across years, so randomizing data would disrupt the natural bounds on the number of teams in each category. We remove the Power Rating ("BARTHAG") feature as we believe it to be derived from the other features via an external model. For seed prediction using XGBoost, we remove the Wins Above Bubble ("WAB") feature as, in a given season, this metric is not known prior to the tournament seeds being published and would induce lookahead bias. Finally, for XGBoost and K-Nearest-Neighbors, we drop the Number

of Games Played ("G") feature and replace it with Win Percentage ("W/G").

3 Predictive Models

3.1 Predicted Seeding - XGBoost

We first use XGBoost to predict tournament seeding, the ordinal response variable. Though tournament seeding is not always directly associated with tournament wins, having a model for seeding is an important first step in explaining the nuances of the dataset and, if successful, may be input in later models. XGBoost is a decision tree algorithm that iteratively trains trees to fit the residuals of the prior tree, but with a regularization term in the objective function that penalizes the complexity of each tree. We first train an initial model on data from the 2013 through 2018 season. Teams that did not make the tournament at all are given 17 as their seed. Next, we validate by tuning three hyperparameters of the model, maximum tree depth (maxdepth), number of trees (nrounds), and shrinkage rate (eta), on data from the 2019 season. We then recombine the training and validation sets to re-train the model chosen in validation. Finally, we test the performance of the model on data from the 2021 season. The two error metrics used to quantify model performance are out-of-sample classification rate (the number of teams correctly classified), and average misseeding (to punish larger misclassifications more). Figure 1 compares the results of XGBoost with that of a trivial classifier that predicts every team to earn seed 17 each year.

Optimal Hyperparameters	
maxdepth	3
nrounds	30
eta	0.5

Model	Classification Rate	Average Misseeding
Training	99.4%	0.02
Testing - Trivial	80.4%	1.6
Testing - XGBoost	80.7%	1.1

Figure 1: Predicted seeding model summary

The top three most important features in the XGBoost model are Wins, Adjusted Offensive

Efficiency, and Adjusted Defensive Efficiency, which explain over half the variability in the features. Winning games efficiently is the key to scoring a better seed in the tournament. Though XGBoost only slightly outperforms the trivial classifier in terms of number of mistakes made, it shines in reducing the average misseeding by half of a seed, which can make a significant difference in first round match-up or side of the bracket predicted.

3.2 Expected Round Placement - K-Nearest-Neighbors

We then use K-Nearest-Neighbors to predict expected round placement in the tournament, given we now know seeding and can use it as a feature. K-Nearest-Neighbors is an algorithm that takes the Euclidean distance from a data point to determine its closest K neighbors in the training set, creating a decision boundary for different classifications. The testing set points are then classified according to this decision boundary. The rounds possible for teams to reach are remaining 68, remaining 64, remaining 32, Sweet 16, Elite 8, Final 4, 2nd place, and Champions. Our goal is to predict the last round a team will reach before a loss as accurately as possible with the data provided. We again use 2013 through 2019 as the recombined training set and 2021 as the testing set. Figure 2 illustrates the testing errors for each value of K from 1 to 30, using 2019 as the validation set. Our resulting best test error occurs when $K = 9$, with a testing misclassification error of 47%. Both the testing misclassification error and model chosen using XGBoost's predicted seedings versus the actual seedings are similar.

It is not possible to constrain a certain number of teams to each round with K-Nearest-Neighbors, so a large number of teams are misclassified in the remaining 32 and 64 rounds due to the proximity and abundance of lower-performing teams. Despite this, the model does make a robust qualitative prediction about the teams it places in the remaining 32 and Sweet 16. Many correctly placed teams won either their conference tournament or their conference

regular season. For example, the model correctly predicts Gonzaga, Houston, Texas, Oregon, and Alabama to outperform in 2021.

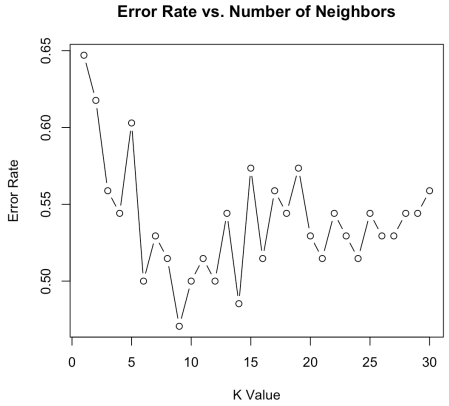

Figure 2: Picking the best k-value

3.3 Long-Term Program Tier - K-Means

Finally, we aim to use unsupervised learning to analyze the factors that determine long-term program success. We apply the K-Means algorithm to separate the teams over the years into a specified number of clusters, until we can clearly differentiate clusters into distinct tiers. We then group the features within each cluster and compare across clusters to see if the characteristics of each cluster are aligned with the performance metrics in the data and if there are any that stand out in the cluster differentiating process. We use the distribution of tournament round placement of each cluster to determine the appropriate number of clusters. The specific K-Means algorithm we apply here is with 3 clusters. This is a deliberate choice because increasing the number of clusters does not create meaningful new clusters that can be differentiated in terms of level of success. Figure 3 provides a detailed example.

We find a distinct top-tier cluster that includes all the historical Champions and 2nd place teams, and most of the Final 4 and Elite 8 teams. The second cluster is the mid-tier with a

	1	2	3
2ND	8	0	0
Champions	8	0	0
E8	31	1	0
F4	15	1	0
R32	114	14	0
R64	137	114	5
R68	9	20	3
S16	61	3	0
Unplaced	6	1127	1125

	1	2	3	4
2ND	0	8	0	0
Champions	0	8	0	0
E8	0	31	1	0
F4	0	16	0	0
R32	1	111	16	0
R64	13	125	114	4
R68	6	8	16	2
S16	1	62	1	0
Unplaced	708	4	817	729

	1	2	3	4	5	6	7	8
2ND	0	0	8	0	0	0	0	0
Champions	0	0	8	0	0	0	0	0
E8	0	0	30	0	0	1	0	1
F4	0	0	14	0	0	2	0	0
R32	0	0	94	3	0	18	0	13
R64	3	4	88	38	7	66	0	50
R68	2	2	2	2	5	8	0	11
S16	0	0	56	2	1	3	0	2
Unplaced	283	471	0	311	433	276	245	239

Figure 3: Clustering with 3, 4 and 8 clusters

large number of historical remaining 32 and remaining 64 teams. We define the third cluster as the bottom-tier because it has the rest of the historical unplaced teams and teams that are very rarely successful when they do make the tournament. Increasing the number of clusters does not change the characteristics of the clusters. In both the 4 and 8-cluster case, there is still a distinct top-tier with all the high-placing teams, while the mid-tier and bottom-tier are split up. Therefore, we conclude that increasing the number of clusters above 3 does not provide more information. Grouping the variables within clusters, we can see in Figure 4 that the top-tier, cluster 1, includes all the historical top-seeded teams. Many of the highest-ranked teams within the top-tier cluster are household names with multiple Final 4 appearances, national championships, conference tournament championships, and conference regular season championships, so we conclude that the clusters are indeed aligned with public perception.

	1	2	3
1	32	0	0
2	32	0	0
3	33	0	0
4	31	0	0
5	32	0	0
6	32	0	0
7	32	0	0
8	32	0	0
9	31	0	0
10	30	2	0
11	39	8	0
12	21	12	0
13	6	27	0
14	0	32	0
15	0	31	1
16	0	41	7
17	6	1127	1125

	1	2	3
Gonzaga	1.00	0.00	0.00
Kansas	1.00	0.00	0.00
North Carolina	1.00	0.00	0.00
Villanova	1.00	0.00	0.00
Cincinnati	0.88	0.12	0.00
Duke	0.88	0.12	0.00
Michigan	0.88	0.12	0.00
Michigan St.	0.88	0.12	0.00
Oklahoma	0.88	0.12	0.00
Oregon	0.88	0.12	0.00

Figure 4: Seeding distribution by cluster and our top-ranked programs

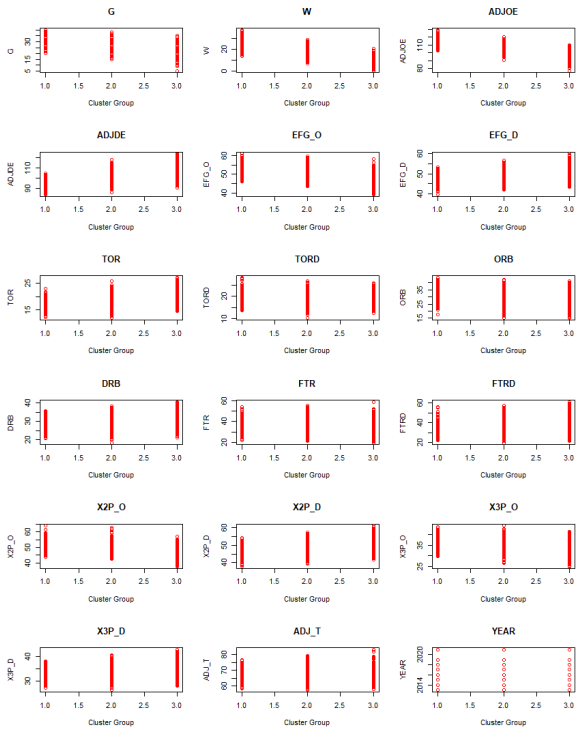

Figure 5: Feature distribution by cluster

We plot each feature against cluster to investigate which features are most indicative of team success in Figure 5. We see that, for each feature, its mean in each cluster is distinctly different. However, it is also clear that there is no feature where the distribution of values of each cluster is significantly different from the rest. Therefore, we cannot conclude that any single feature is most prominent in separating the clusters. In other words, clusters are formed with a holistic view, and the program tiers reflect that.

4 Conclusion

We successfully formulate a meaningful tournament seeding and round placement prediction, as well as categorize the overall long-term capabilities of different NCAA DI Men's Basketball programs. The seeding and placement predictions may yield a betting edge. When in doubt, bet on a conference champion, a regular season champion, or an efficient, offensively/defensively balanced team to win at least their first round game. Coaches should aim for their teams to achieve these feats too. The clustering analysis gives insight into the quality of basketball programs across the nation for recruits. However, we fail to robustly differentiate on styles of play, since no feature was most prominent in cluster separation. Perhaps analyzing the results of multiple rounds of clustering, using random subsets of features at each round, would better reveal this information.

References

https://www.kaggle.com/datasets/andrewsundberg/college-basketball-dataset

Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016