Forecasting Outcomes of 2018 Women's March Madness

W207 Final Project

Julia Buffinton, Charlene Chen, Arvindh Ganesan, Prashant Kumar Sahay

Due: 4/17/18

In this project, we use historical data about NCAA Division I women's basketball games and team performance to predict game outcome probabilities for every possible matchup of teams of the 2018 NCAA Division I Women's Basketball Tournament.

Business Understanding

The purpose of this project is to build a model to predict the outcome probabilities for every possible matchup of teams of the 2018 NCAA Division I Women's Basketball Tournament. The NCAA tournament is also commonly known as March Madness, not only because it occurs annually in March, but also because of the fast-paced single elimination mechanism and the surprising results that often arise when pairing teams with no history playing each other. This year's men's tournament saw the first-ever upset of a 1-seed (UVA) by a 16-seed (UMBC).

Therefore, building a model to predict the results of the tournament is not only interesting but quite difficult - no one has ever made a perfect bracket. The rewards for doing so can be quite high. Predicting the outcomes is a fun pastime for millions of Americans who participate in pools among friends, family, and coworkers. In many cases, bragging rights are the prize, but in many others, money is at stake for those who buy into a pool or bet on individual game outcomes.

Data Understanding

Our dataset was obtained through a Kaggle competition sponsored by Google Clound and NCAA. This includes historical data from both tournament and regular season games back to 1998. However, they key information that we used for analysis is 'detailed' results from games play from 2010 (prior to that, it is unavailable in our dataset) which includes: season (year); day number in the season the game was played; teams participating; final score; location; number of overtimes; and performance statistics such as 2 pointers attempted/made, 3 pointers attempted/made, free throws attempted/made, offensive/defensive rebounds, assists, time outs used, steals, blocks, and personal fouls for each participating team.

While the bulk of the data is contained in these game-level results, our original dataset also contained seeds of all tournament teams from 1998 to present and cities in which the games occurred.

EDA

Distribution of performance measures for the winning and the losing teams

We analyzed the following performance measures : Field Goal Percentage, 2Pt Field Goal Percentage, 3Pt Field Goal Percentage, Free throw percentage, Offensive and defensive rebounding efficiencies, Turn over per game, Assists per game, etc.

On an average, Field Goal Percentage, 2Pt Field Goal Percentage, 3Pt Field Goal Percentage and Assists per game made the telling difference between winning and losing.

Impact of relative team performance on the game outcome

Based on the analysis of difference of the key metrics such as Field goal performance, assists, etc, these metrics have a positive correlation with score margin.

Impact of regular game performance on the tournament games

Our analysis also showed that the regular season performance has a positive correlation with the tournament game wins. It is evident from the distribution of the key performance metrics for the winning and losing teams as well.

League Performance

We analyzed the performance in the tournament of the teams within a league and aggregated that collective performance as a substitute for league strength. Using this as a league performance metric, we were able to compare the relative performance of leagues. It turned out that teams that belong to leagues with higher tournament win percentage tends to do better in the tournaments overall.

Distribution of each feature

There are 13 field performance features for each team in each game of the tournament or regular season (FGM, FGA, FGM3, FGA3, FTM, FTA, OR, DR, Ast, TO, Stl, Blk, PF). These indicate 2 pointers made/attempted, 3 pointers made/attempted, free throws made/attempted, offensive/defensive rebounds, assists, time outs, steals, blocks, and personal fouls, respectively.

For the relevant field statistics, we did the Kernel density estimation (KDE) plot to compare the winning team and losing team. Almost all the fields metrics have similar distribution between winning team and losing team. However, winning teams show an advantage in Field Goal Percentage (FGP), 2 point field goal shooting (FGP2), 3 point field goal shooting (FGP3),and Assists (Ast), indicating those metrics may be important features to examine in detail and be used in the modeling.

The winningest teams are relatively consistent across both tournament and regular season, reflecting that the better performing teams in the regular season qualify for and perform well in tournament play. indicating a consistent performance of each team from regular season to tournament play.

Regular Season Games Won since 1998	Tournament Games Won since 1998
UConn (632)	UConn (93)
Tennessee (554)	Tennessee (69)
Duke (544)	Duke (52)
Stanford (543)	Stanford (52)
WI Green Bay (531)	Notre Dame (51)

Using past tournament performance as a feature would have been difficult, because of the 364 teams, only 252 of them have participated in the NCAA March Madness tournament at least once since 1998. We are missing this history for many teams we may be predicting on, and are concerned about bias. However, we hypothesized that the lack of appearance may be noteworthy itself and considered it in feature selection.

As mentioned above, our dataset also includes seeds for all teams in the tournament since 1998. Seeds are between 1 and 16 within each region (4 regions, therefore 4 instances of each seed number per year).

Data Quality

We initially hypothesized that regular season game performance may not be a key indicator of tournament game outcomes because it reflects the relative strength of each team participating. However, teams play primarily within their leagues in regular season play, but frequently play across leagues in the tournament. We added league assignments, obtained from NCAA, to our dataset. This is the only place we encounter missing values (teams that no longer participate in NCAA basketball), so we excluded these (n=8).

Additionally, we noticed that the ‘compact’ results file from the regular season include more games than the ‘detailed’ results, which we needed for our analysis, but this is a difference of ~300 out of over 46,000 total games played between 2010-2018, so we determined that it was okay to continue our analysis with games from the ‘detailed’ results file.

Because our dataset is largely based on tournament games, it contains much more information for certain teams that perform at a high level and thus regularly participate in tournament games. Many teams that participate in regular season play will not have corresponding tournament performance data.

Other Noteworthy Observations

One thing that was very apparent in EDA is the overall dominance of the University of Connecticut team and several other teams (such as Notre Dame or Tennessee) in regular season and tournament games. Since UConn and these other teams have performed highly for many years in a row, this suggests that some other factor besides player-level skills (since turnover in college basketball is relatively high) contributes to the success (such as a coach, etc.). We did not include features such as coach, but consider it for future examination.

The ultimate output data is the win or lose label for the first team participating in each game. Since each team will either win or lose the game, the output data is uniformly distributed with equally probability of value 1 and 0.

Data Preparation

After our EDA, we concluded that a decent amount of feature engineering needed to be accomplished to achieve actionable results. Thus, we built our final training and test sets from several aggregate features present throughout the datasets received from Kaggle.

Generate negative examples

While for our final prediction, we will predict outcomes for combinations of two teams (order doesn’t matter), our datasets contain permutations of two teams, always with the winning team listed first and a result of 1. To remove bias in training, we generate examples with the reversal of pairs and a result of 0. Then we can complete feature engineering for all training combinations.

Feature Engineering

Features we used

In our final model (see **Modeling**, below), we use only three features: Seed Difference Percent, League Bin Difference, and Win Probability (WinProb).

Win Probability (WinProb)

Win probability reflects our attempt to create a ‘performance’ feature. This is a probability between [0,1] that a team would win a matchup if it occurred during the regular season. We augment this information with the features listed below to reflect that we are predicting games that occur during the tournament, where performance may not be as consistent with regular season.

Our EDA identified correlations between a team’s performance during the regular season and its wins during the tournament. Based on the findings of the EDA, we utilized regular season games for feature development. The following regular season data was used for feature development:

- Field goals attempted
- Field goals made
- Free throws attempted
- Free throws made
- Blocks
- Rebounds
- Assists
- Steals
- 3 point goals

We calculated intrinsic percentage-based features as follows: - Field goal percentage = Field goals made / Field goals attempted - Free throw percentage = Free throws made / Field goals attempted

In addition, we synthesized the concepts of Point Opportunity Developed (POD) and Opportunity Conversion Rate (OCR). The concept of Point Opportunity Developed (POD) represents the effectiveness with which a team develops new opportunities to score. It is defined as:

$$POD = three - pointgoalsattempted * 3 + two - pointgoalsattempted * 2 + freethrowsattempted * 1$$

The concept of Opportunity Conversion Rate (OCR) represents the effectiveness with which a team successfully executes on its PODs. It is defined as:

$$OCR = scoreinthegame/pointopportunitydeveloped$$

To calculate win probability, we first aggregated field statistics listed above (two pointers attempted, assists, etc.) for each team in each season.

Feature Scaling and Standardization:

All features were calculated the performance of a team during a season and within a league. This is because competitiveness varies across leagues. As a result, within-league performance is not directly comparable across leagues. We evaluated a couple of options to address this: 1) to normalize performance across leagues, and 2) to operationalize a feature that reflects the relative performance of leagues during model training. We decided to opt for the latter option as it allowed the model training process to pick up the latent signals. To enable this approach, we measured the performance of leagues by year, rescaled and normalized the league performance, and binned them as ordinal data.

We calculated the features in following steps: 1. Aggregation: Calculate the performance of a team within season and league 2. Centering: De-mean the data by subtracting the corresponding league-level mean 3. Normalization: Rescale by dividing by the corresponding league level standard deviation 4. Binning: To dampen the effect of small variations (noise)

To determine the WinProb, we utilize these binned features as ordinal measures.

With this, we were able to build "matchups" of all regular season games (2010-2017) that occurred, supplemented with each team's binned performance measures for that year, and the outcome of the matchup. We used this dataset to train a Gaussian Naive Bayes model. We used the predicted probabilities for the positive (winning) class generated for test 2018 data to reflect what the predicted winning probability would be for the 2018 tournament matchups of interest.

Seed Difference Percent

As suggested on Kaggle, we used seed difference as a baseline for model performance. However, we hypothesized that this was an imperfect measure, as in practice, the seed difference between higher seeds seems to carry more weight than that between lower seeds (for example, the difference between the #1 seed and the #2 seed is greater than the difference between a #15 seed and a #15 seed). To reflect this, we calculated a seed difference percentage, which is the seed difference divided by the sum of the seeds.

$$(team2seed - team1seed)/(team2seed + team1seed)$$

League Bin Difference

Furthermore, competitiveness varies across leagues. For example Big10 and Pac12 leagues are significantly more competitive than smaller leagues. As a result, within-league performance is not directly comparable across leagues. We evaluated a couple of options to address this: 1) to normalize performance across leagues, and 2) to operationalize a feature that reflects the relative performance of leagues during model training. We decided to opt for the latter option as it allowed the model training process to pick up the latent signals. To enable this approach, we measured the performance of leagues by year, rescaled and normalized the league performance, and binned them as ordinal data.

While conducting EDA, we again noticed that the regular season statistics are not necessarily indicative of tournament performance due to the difference in competitiveness between leagues. Some leagues contain teams that perform at a higher level than other leagues. We used aggregate statistics for historical win percentage in the tournament as an indicator of a league's competitiveness. This reflects the idea that leagues with teams that qualify and win more games in the tournament are more competitive. Our proxy for league competitiveness seems to track with general knowledge of the more competitive leagues, particularly for women's basketball.

From there, we binned the league win percentages and calculated the difference between the leagues to which the teams in each matchup belonged to reflect the difference in league strength. #### Feature engineering attempted but not selected for modeling Throughout our EDA, we explored numerous potential indicators of game outcomes. In addition to the above-mentioned features ultimately selected, we also considered: PCA on regular season field performance measures Previous year's tournament seeds Average tournament seed over past 4-5 years Current year seed Average games won in a tournament over past 4-5 years Win percentage in neutral sites Regular season performance over past 4-5 years Head to head history for each matchup Elo rating

Features related to tournament performance were calculated cross sectionally. For example, we compared league performance across time periods to operationalize the notion of a league's relative strength. The relative strength of the a league is intended to be used to normalize teams' performance during the regular season.

Modeling

We used an ensemble approach to modeling by combining Naïve Bayes, Random Forest, and Logistic Regression. We used a cross-validation training dataset to generate first stage classification using Naïve Bayes and Random Forest models. In the second stage, we used the first stage classifiers as features to generate the final classification log probabilities.

In the game of basketball, a team's strengths can be used to exploit the other team's weaknesses. For example, superior defensive rebounding and steals by one team can lean against the field goal attempts by the other team, or superior blocking by one team can lean against the field goal percentage for the other team. We have tried to operationalize the interplay of the strengths and weaknesses of teams by binning the performance under categories such as blocking, rebounding, etc. In order to dampen the effect of small variations (noise), we centered and normalized each team's performance before assigning it to a bin. We used a fixed number of bins under each category but it could make sense to treat the number of bins under each category as a hyper parameter and tune it during training.

The composition of NCAA teams changes over time due to high level of churn caused by (approximately) one fourth of the students graduating college every year. The performance of teams, therefore, varies under each category from year to year. In addition, we learned during EDA that the relative importance of performance under different categories has changed over time. To capture the time-dependent effects, all performance measurement and categorization was performed by team within each season.

We used a seed difference-based simple model as a starting baseline. The baseline model produced a log loss of about 0.46 with the cross validation training set. However, because seeding data is only available for tournament participants, only 16 teams from each region have associated seeds. During each season, we have 64 teams with seeds. This includes 32 teams the 32 top teams from each league and 32 teams selected by a committee for each season. This posed a problem, we could not use regular season data and seed difference together. To address this, we decided to develop a distinct set of models based on regular season data and use an ensemble approach to combine regular season based models with tournament data based models such as the one based on seed difference.

Evaluation

We finally choose our optimal model as the logistic regression model with aggregated features 'SeedDiffPct', 'LeaguebinDiff', 'WinProb'.

In the evaluation session, we first compared various performances of optimal model with that of our original baseline (just using one feature, 'SeedDiff'), with cross validation of training data. Then, we can use our optimal model to predict the winning prob of 2018 tournament and got the log loss score to compared with Kaggle leaderboard ranking.

- Log Loss
- Log Loss is the most important classification metric based on probabilities. It is also a key way of evaluation of our prediction from Kaggle NCAA page.
- It is also called the Log Likelihood Function. The log of the likelihood that the 2018 tournament bracket actually happens based on our prediction of winning probability of each team matchups.

$$LogLoss = \sum_{i=1}^{n}[y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})]$$

- The result will be a negative number that's in a range computers keep track of. However, Scikit-learn has a convention in its metrics that lower scores are better.
  - So, scikit-learn report

$$-1 * \frac{LogLoss}{NumObservations}$$

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}[y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})]$$

- The division by the number of datapoints is used so the range of values doesn't systematically vary with the dataset size. - The rage of that is 0 to infinity. - The lower the metrics is, the better model performance is.

- Log loss penalizes both types of errors, but especially those predictions that are confident and wrong.
- The Log Loss value of optimal model is 0.44228, which is a very good outcome, better than the baseline model by 0.015.
- Accuracy
- Accuracy measures a fraction of the classifier's predictions that are correct.
- The accuracy score of optimal model is 0.78178, higher than baseline model by 0.001.

- F1 measure with Cross Validation
- **Precision** is the fraction of positive predictions that are correct, which means the team who we predicts as winner that are actually winner.
  - (TP = True Positive, FP = False Positive)

$$P = \frac{TP}{TP + FP}$$

- **Recall**, the true positive rate, is the fraction of the truly positive instances that the classifier recognizes. A recall score of one indicates that the classifier did not make any false negative predictions. In our case, recall is the fraction of winner team that were truly classified as winner, according to our prediction of winning probability.
  - (TP = True Positive, FN = False Negative)

$$P = \frac{TP}{TP + FN}$$

- In our case, the precis- The **F1 measure** is the harmonic mean, or weighted average, of the precision and recall scores. Also called the f-measure or the f-score.
  - (P = precision, R = recall)

$$F1 = 2\frac{PR}{P + R}$$

- The F1 measure penalizes classifiers with imbalanced precision and recall scores, like the trivial classifier that always predicts the positive class.
- A model with perfect precision and recall scores will achieve an F1 score of 1. A model with a perfect precision score and a recall score of zero will achieve an F1 score of 0.
- Models are sometimes evaluated using the F0.5 and F2 scores, which favor precision over recall and recall over precision, respectively.ion and recalls are. . .
-
  - The **F1 measure** is the harmonic mean, or weighted average, of the precision and recall scores. Also called the f-measure or the f-score.
  - (P = precision, R = recall)

$$F1 = 2\frac{PR}{P + R}$$

- The F1 measure penalizes classifiers with imbalanced precision and recall scores, like the trivial classifier that always predicts the positive class.
- A model with perfect precision and recall scores will achieve an F1 score of 1. A model with a perfect precision score and a recall score of zero will achieve an F1 score of 0.
- Models are sometimes evaluated using the F0.5 and F2 scores, which favor precision over recall and recall over precision, respectively. In our case, we just use F1 score.
- For the optimal model, the arithmetic mean of our classifier's precision and recall scores is 0.7821, slightly higher than that of baseline model, 0.7812. For both model, the F1 scores are high, indicating that the F1 measure's penalty is small.

Error Analysis

During training, we examined classification errors by looking for clues as to why specific games were misclassified. However, it was difficult to identify any patterns or reasons for misclassification. Instead, we primarily relied on model accuracy scores and log losses on the development/training data to guide us in error estimation and feature selection.

As we proceeded with including various combinations of features in model development, some patterns stood out:

Including free throw attempts and free throw percentages in the model did not meaningfully change model accuracy. The log loss remained very close to the baseline log loss of around 0.46. It could be the case that different teams develop point opportunities differently in ways that align with their specific strengths. For example, some teams may emphasize drawing personal fouls from opponents to score points using free throws. Other teams may have better field goal percentages and emphasize scoring points using field goals instead. In that sense, emphasis on free throws represented just one of many possible strategies with no direct implication for game outcomes.

We noticed a very small effect, in the order of a hundredth of a log loss point, of two and three point field goal attempts. However, we observed a meaningful log loss improvement of between 0.1 and 0.2 when we used a derived measure of performance, the field goal percentage. It appeared that field goal percentage captured the quality of the team whereas the number of two-point and three-point field goals represented a strategy that the team chose. Most models worsened when using features based on steals and turnovers. The log losses got larger than the baseline log loss by 0.1 to 0.4. It appeared that there was not much systematic about steals and turnovers that the models could capture. Adding them to the model just led to more noisy data and resulted in more variance without increasing model accuracy.

It appeared that more intrinsic measures such as field goal percentage, point opportunities developed, and point opportunities converted better captured the qualitative elements of a team's strengths and generally led to lower log losses and higher model accuracy.

Both SVM and Random Forest classifiers tended to suffer from overfitting. This was surprising as we had a reasonably large amount of data about regular season games. It is likely that the large number of features led to overfitting. We experimented with using PCA to reduce dimensionality of data but lost considerable accuracy.

Based on the insights we have gained from the error analysis, we think that the models are not picking up important patterns in how the styles of play and strengths or weaknesses of opposing teams interact. We may be able to model such patterns by including interactions of features and by using custom kernels with logistic regression and SVM models to better deal with nonlinearities.

We have tried to recover the "edge" that one team has over another in a matchup by including features that potentially interact. For example, the blocking prowess of one team would tend to reduce the field goal percentage of the other team. Or, the rebounding prowess of one team would reduce the field goal and free throw attempts by the other team. But we are primarily relying on the model to capture these interactions. We can probably model a classifier with better RMSE in the first stage by using a neural network model that can better sense such bidirectional interactions.

- ROC Curve
- Receiver Operating Characteristic, or ROC curve, visualizes a classifier's performance.
- ROC curve illustrates the classifier's performance for all values of the discrimination threshold.
  - The true positive rate (Sensitivity, Recall) is plotted in function of the false positive rate (100-Specificity, Fall-out) for different cut-off points.
  - Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold.
- AUC is the area under the ROC curve; it reduces the ROC curve to a single value, which represents the expected performance of the classifier.
- The dashed line in the following figure is for a classifier that predicts classes randomly; it has an AUC of 0.5. The solid curve is for a classifier that outperforms random guessing.
- A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC curve

is to the upper left corner, the higher the overall accuracy of the test.

- Both of the ROC curves are on the upper left corner, indicating a good discrimination. The average AUC value from cross validation of optimal model is 0.87, higher than that of baseline model, which is 0.86. Hence, the expected performance of the optimal model is better than baseline model.
- The ROC curve of the optimal model is slightly deeper at the beginning than that of baseline model. Kaggle punishes the prediction that something is true when it is actually false, the false positive prediction, which means we are looking for less false positive rate. So, the optimal model performs better, under the rule of Kaggle (https://www.kaggle.com/c/womens-machine-learning-competition-2018#evaluation)
- Summary

After checking Log Loss, Accuracy, Precision and recall, F1 measure, and ROC Curve with cross validation of training data, the optimal model performs better in all measurements: - Higher accuracy and F1 score; - less false positive rate and higher AUC in ROC curve; - and most importantly, smaller log loss score.

Deployment

Log Loss on 2018 Tournament Result

The Kaggle competition required a submission of predictions for the win likelihood of every potential matchup in the 2018 NCAA Division I Women's Basketball Tournament. Since team1 vs. team2 is the same as team2 vs. team1, we only included the game pairs where team1 has the lower team id (total of 64*63 / 2 = 2,016 matchups). Predictions were submitted as a .csv file with a list of every possible matchup between the tournament teams and the prediction of the probability that team1 in the matchup would be team2.

However, evaluation (as described above) could only be completed for games actually played (63 matchups). The evaluation technique used in this competition is log loss. The winning Kaggle competition submission achieved a log-loss of 0.406819.

The log loss score for our optimal model is 0.43159, which is a great outcome. The Log Loss score follows the evaluation criteria of Kaggle competition, we would be placed in the rank of 16 from 505 teams. Accuracy is 0.7619, which is also relatively high.

Summary and Conclusions

We were able to achieve a log-loss of 0.43159 on 2018 NCAA Division I Women's Basketball Tournament game matchups using an 'ensemble' model reflecting regular season performance and tournament-level information for each team in the matchup.

There appears to be room for improvement, although we recognize that there may be a ceiling in performance of our models on tournament outcomes. Especially since we perform reasonably competitively against other teams in the Kaggle competition (and we certainly are not experts, yet!), we hypothesize that we may be quickly approaching this performance ceiling.

Our final predictions were not submitted to Kaggle, as we finalized this model after the competition had completed. However, we are encouraged by our performance and plan to test this model in future years as well. We look forward to predicting the 2019 NCAA Division I Women's Basketball Tournament.