Case Study 2 - project report

March Machine Learning Mania 2023 - Kaggle Competition

Jakub Grunas Aleksander Malinowski Damian Skowroński

1 Introduction

In the project, we focused on the Kaggle competition that can be found at this link[1]. The competition revolved around the March Madness tournament, specifically predicting the probabilities of match outcomes in this year's (2023) tournament. The tournament concluded in early April, and along with it, the competition ended. We joined in to develop a solution after the competition had already concluded.

2 About the tournament

In order to understand the decisions we made in the solution, it is important to grasp the nature of the March Madness tournament and its format. March Madness is a basketball tournament played between teams from the NCAA Division 1, the collegiate basketball league in the USA. It has been organized for both men and women since 1939 during March/April and typically lasts for about three weeks. The tournament is one of the most significant sporting events in the USA.

2.1 Current tournament format

The tournament includes 68 teams. They are divided into 4 conferences and assigned different seeds. The seeds are allocated by a committee based on various factors - regular season data, data during the selection week, and discussions among committee members. The process of assigning seeds is complex, but it is not necessary to understand it for our project. All the information contained in this section applies to both the men's and women's tournaments.

There are seven stages of the tournament (Table 1). The games are played in a bracket format.

Round	Teams Remaining	Teams Competing	Games
First Four (play-in round)	68	8	4
First round	64	64	32
Second round	32	32	16
Sweet Sixteen (regional semifinals)	16	16	8
Elite Eight (regional finals)	8	8	4
Final Four (national semifinals)	4	4	2
Championship game	2	2	1

Table 1: Summary of tournament rounds[2]

3 About the Kaggle Competition

The competition related to March Madness has been organized in various formats on the Kaggle website every year since 2014 except for 2020, when the tournament didn't take place due to Covid. Usually, the top results are rewarded with significant cash prizes. This year, it was necessary to focus on both the men's and women's tournaments. There were 1033 submissions and the prize money was $50,000.

3.1 Provided data

The organizers provided 31 CSV files containing historical data related to the tournament, with a total size of approximately 144 MB. They also encouraged the use of additional data beyond what was provided. The file naming scheme follows the convention where data related to the men's league has the prefix "M", while data related to the women's league has the prefix "W". In our solution, we used the following tables:

- *MRegularSeasonDetailedResults.csv*, *WRegularSeasonDetailedResults.csv* - These files provide team-level box scores for many regular seasons of historical data, starting with the 2003 season (men) or starting with the 2010 season (women). The box scores columns are ("W" and "L" prefixes refer to the winning or losing team):
  - **WFGM**, **LFGM** - Field Goals Made
  - **WFGA**, **LFGA** - Field Goals Attempted
  - **WFGM3**, **LFGM3** - Three Pointers Made
  - **WFGA3**, **LFGA3** - Three Pointers Attempted
  - **WFTM**, **LFTM** - Free Throws Made
  - **WFTA**, **LFTA** - Free Throws Attempted
  - **WOR**, **LOR** - Offensive Rebounds
  - **WDR**, **LDR** - Defensive Rebounds
  - **WAst**, **LAst** - Assists
  - **WTO**, **LTO** - Turnovers Committed
  - **WStl**, **LStl** - Steals
  - **WBlk**, **LBlk** - Blocks
  - **WPF**, **LPF** - Personal Fouls Committed

  There are also columns that are not part of the box score information:

  - **Season** - Season year. For example, during the 2016 season, there were regular season games played between November 2015 and March 2016, and all of those games will show up with a Season of 2016.
  - **DayNum** - Day the game was played on. Offset from "DayZero".
  - **WTeamID**, **LTeamID** - Team IDs for identifying between different files.
  - **WScore**, **LScore** - Number of points scored.
  - **WLoc** - Location of the winning team (Home/Away/Neutral court)
  - **NumOT** - Number of overtime periods.
- *MNCAATrouneyDetailedResults.csv*, *WNCAATrouneyDetailedResults.csv* - These files provide team-level box scores for many NCAA tournaments, starting with the 2003 season (men) or starting with the 2010 season (women). The data format is the same as in *MRegularSeasonDetailedResults.csv*, *WRegularSeasonDetailedResults.csv* files above.
- *MNCAATrouneySeeds.csv*, *WNCAATrouneySeeds.csv* - These files identify the seeds for all teams in each NCAA tournament, for all seasons of historical data. Thus, there are between 64-68 rows for each year, depending on whether there were any play-in games and how many there were. The columns are:
  - **Season** - Season. Same as above.
  - **TeamID** - Team IDs. Same as above.
  - **Seed** - This is a 3/4-character identifier of the seed, where the first character is either W, X, Y, or Z (identifying the region the team was in) and the next two digits (either 01, 02, ..., 15, or 16)
- *MTeamConferences.csv*, *WTeamConferences.csv* - These files indicate the conference affiliations for each team during each season. Some conferences have changed their names from year to year, and/or changed which teams are part of the conference.
  - **Season** - Season. Same as above.
  - **TeamID** - Team IDs. Same as above.
  - **ConfAbbrev** - 3-character identifier of the conference.
- *MMasseyOrdinals.csv* - This file lists out rankings (e.g. 1, 2, 3, ..., N) of men's teams going back to the 2002-2003 season, under a large number of different ranking system methodologies. The ranking systems are presented on a weekly basis. This table is only available for men's teams. The columns are:
  - **Season** - Season. Same as above.
  - **TeamID** - Team IDs. Same as above.
  - **RankingDayNum** - Expressed in the same terms as a game's DayNum. Indicates the first day that it is appropriate to use the rankings for predicting games.
  - **SystemName** - The abbreviation for each distinct ranking system.
  - **OrdinalRank** - The overall ranking of the team in the underlying system.

3.2 Additional data

In the project we also used data from different sources:

- *WRatings.csv* - From NCAA Women 538 team ratings[3]. Women's teams ranking from seasons between 2016 and 2023. Last updated before the 2023 tourney. This dataset should be somewhat equivalent to *MMasseyOrdinals.csv*. The columns are:
  - **Season** - Season. Same as above.
  - **TeamID** - Team IDs. Same as above. Actually consistent with the provided data Team IDs.
  - **TeamName** - The name of the team.
  - **538rating** - Team rating at the end of the season.
- *Mtournament_team_data.csv* - From March Madness Data[4]. This dataset features the teams and their stats from all tournaments since 2008 up to 2022 (only for men's teams, for women's teams we coulnd find anything like that). The dataset has many columns but those used by us are:
  - **YEAR** - Same as "Season" columns in the datasets above.
  - **TEAM** - The name of the collage team.
  - **ROUND** - The last round the team played in a tournament.

  This dataset is used specifically to get historic appearances of teams in the tournament. It doesn't have a dedicated "TeamID" column consistent with TeamIDs in other tables, therefore for joining we used *MTeamSpellings.csv* - dataset which was provided by the comptetition organiser. This dataset is a supplement specifically for merging external data. It's like a dictionary for different team spellings (column **TeamNameSpelling**) and the corresponding **TeamID**.

4 Feature engineering

We prepared the data and performed feature engineering in the file *feature_engineering.ipynb*. The subsequent steps of data transformation are saved as functions. In this section, we will explain what we did with the data to eventually obtain the data that we used for modeling. This process was rather complicated, therefore the explanation may be a little convoluted. During this process, we drew partial inspiration from the solutions presented on the competition website, namely the first and second place solutions.

4.1 Preparing regular season and tourney tables

Let's say that the tables we're working with are called **M_regular**, **W_regular**, **M_tourney**, **W_tourney** for regular and tourney tables for men's and women's league respectively. The main features in our model will be the statistics from the regular season and the tournament. To obtain this information, we start by transforming these tables using the `duplicate_and_switch_sides` function. We need to change the prefixes "W" and "L" to "T1_" and "T2_" in a way that makes sense. Here's roughly how the function achieves this:

1. The functions accepts one of the previously mentioned tables. Let's call such table `df`.
2. `df` is being copied. Let's call the copy `df_copy`. The copy will be used to duplicate the rows and swap positions of "WTeam" and "LTeam". In `df` the games are from the winner's point of view ("W" = "T1_"), in `df_copy` the games are from the loser's point of view ("L" = "T1_")
3. In both `df` and `df_copy` we make a new column **location**. In `df` this column has values:
   - 1 if WLoc is home court
   - 0 if WLoc is neutral court
   - -1 if WLoc is away court

   In `df_copy` this column has values -1 and 1 swapped.
4. In `df` we replace every "W" prefix with "T1_" and "L" with "T2_". In `df_copy` we do this the other way around - we replace every "L" prefix with "T1_" and "W" with "T2_".
5. We concatenate both `df` and `df_copy`, resulting in all games being duplicated from the perspective of both teams. It's important to note that there are no duplicate rows though.
6. We add a new column **point_diff** = T1_Score - T2_Score

We use this function on all of the four tables: **M_regular**, **W_regular**, **M_tourney**, **W_tourney**.

4.2 Regular season statistics

The function `prepare_regular_season_team_statistics` aggregates the data from the whole season per team and returns the average statistics. Here are the steps:

1. Create new columns:
   - **T1_FG%**, **T2_FG%** - Percentage of shots from game made.
   - **T1_3P%**, **T2_3P%** - Percentage of 3-point shots made.
   - **win** - 1 if T1 won, 0 otherwise.
   - **win_ratio_last_month** - win percentage for games after day 102 of the season.
2. Create new aggregated columns:

   T1_FG%_home_mean, **T1_3P%_home_mean**, **point_diff_home_mean**, **win_home_mean** and

   T1_FG%_away_mean, **T1_3P%_away_mean**, **point_diff_away_mean**, **win_away_mean**.

   These columns are about the average performance of a team during the season while playing on home court or away/neutral court.
3. Create new aggregated columns from the teams performance from the whole season:

   T1_Score_mean, **T1_FGM_mean**, **T1_FGA_mean**, **T1_FGM3_mean**, **T1_FGA3_mean**, **T1_FTM_mean**, **T1_FTA_mean**, **T1_OR_mean**, **T1_DR_mean**, **T1_Ast_mean**, **T1_TO_mean**, **T1_Stl_mean**, **T1_Blk_mean**, **T1_PF_mean** and the same for their opponent's average performance (those columns for now stay with prefix "T2_" but later the prefix will change to "T1_opponent").

We use this function on both **M_regular** and **W_regular** tables.

4.3 Last tourney performance

The function `prepare_previous_tourney_data` accepts both regular season and tourney tables. It creates two new columns in the regular season table:

- **played_prev_tourney** - whether a team played in the last year's tourney. 1 if True, 0 otherwise.
- **prev_win_ratio** - win ratio in the last tourney. If the team hasn't played then the win ratio is 0.

Using this function we transform **M_regular** and **M_tourney** into **M_teams_working** table. We do the same for women's league data and we get **W_teams_working** table.

4.4 Additional information

In this section we're adding tourney seeding, and team's rating for both men's and women's tables. Here are the steps taken:

1. Using `correct_seeds` function we transform **M_seeds** and **W_seeds** tables to a correct format and merge them with their corresponding **M_teams_working** and **W_teams_working** tables by Season and T1_TeamID.
2. For women's league, to get the team's rating we just merge **W_ratings** table with **W_teams_working** by Season and T1_TeamID. The new column is called **Ranking**.
3. For men's league we use **M_Massey** table to get the average rating from several rating systems for the last week of the regular season. Then we merge this table to **M_teams_working**. The new column is called **OrdinalRank**.

4.5 Male's tourneys past appearances

Firstly we prepare the data for past tourney appearances using the function `prepare_Mtourney_appearances`. It creates a dataframe, with the following columns:

- **Season** - As above.
- **TeamID** - As above.
- **total_tourney_appearances** - How many times the team has played in a tourney before this season.
- **top68_apperances** - How many times the team has ended the tourney in top 68.
- **top64_apperances** - How many times the team has ended the tourney in top 64.
- **top32_apperances** - How many times the team has ended the tourney in top 32.

- **top16_apperances** - How many times the team has ended the tourney in top 16.
- **top8_apperances** - How many times the team has ended the tourney in top 8.
- **top4_apperances** - How many times the team has ended the tourney in top 4.
- **top2_apperances** - How many times the team has ended the tourney in top 2.
- **top1_apperances** - How many times the team has won the tourney.

Such table is merged with **M_teams_working** by Season and T1_TeamID.

4.6 Conference margin

The reason we consider conference information is that teams' conferences are not equal in terms of playing level, which influences match statistics and win ratios. We aim to incorporate information about the disparity in playing level between conferences. Here are the steps taken:

1. Using the function `prepare_conferences` we create new tables for male's and female's leagues by accepting **M_regular** or **W_regular** tables and merging them with the corresponding **M_conferences** or **W_conferences** table with conferences abbreviations.
2. The resulting dataframe has the following columns:
   - **Season** - As above.
   - **T1_TeamID**, **T2_TeamID** - As above.
   - **T1_ConfAbbrev**, **T2_ConfAbbrev** - The abbreviations of T1's and T2's conference names.
   - **location** - -1 if T1 away, 0 if neutral, 1 if T1 home.
   - **point_diff** - T1_Score - T2_Score.
3. Let's call the resulting tables as **M_conferences_working** and **W_conferences_working**
4. We use the function `prepare_conference_margin` on **M_teams_working** and **M_conferences_working** for men's data and **M_teams_working** and **W_conferences_working** for woman's data.
5. The function creates two new columns in **M_teams_working** and **W_teams_working** tables:
   - **conf_point_margin** - average point difference when a team plays against a team from other conferences
   - **conf_win_margin** - win ratio when a team plays against a team from other conferences

4.7 Final preperation

At this point in **M_teams_working** and **W_teams_working** we have all the data we needed. This last section is about getting the final dataframes for training - past tourneys data with team's features, and this year's tourneys. Here are the steps:

1. We put **M_teams_working** and **M_tourneys** into the `final_preparation` function (for men's league, same for women's).
2. This function does a couple of things:
   - Replaces "T2_" prefix with "T1_opponent"
   - Copies the **M_teams_working** table (for men's league) into a new dataframe - `df_T2`
   - In `df_T2` - replaces every "T1" prefix with "T2". Now in the **M_teams_working** there are only "T1" columns and in `df_T2` there are only "T2" columns though the rows are duplicated.
   - Performs outer merge on those two tables by Season. This way there is every possible game between two teams in a given season.
   - Splits the merged dataframe into two subsets. The first one is called **M_solution** (or **W_solution**) - in this table there are only possible matches from this season. The second one is called **M_train** (or **W_train**), It has all the remaining seasons, we can drop all the rows where a seed from either T1 or T2 is missing, which means that the match couldn't have happened.
   - Merges the second table with the past tourneys data, so that there are the match outcomes. Drops every row that misses the match outcome - the match did not happen.
   - For the solution table drops every row where T1_TeamID >= T2_TeamID. For the solution table creates a new column called **ID** it is a string which follows the pattern "2023_<T1_TeamID>_<T2_TeamID>". This step makes sure that our solution dataset is consistent with the Kaggle competition requirement.

- For both solution and train dataframes adds a column **seed_diff** = T1_Seed - T2_Seed.

3. We try to get rid of the remaining missing values. After all the preprocessing we discovered that a small percentage of observations in women tourney training dataset contains missing values in several columns. The identified missing values were observed only for a few specific teams therefore we decided to remove these observations from our data since there was no satisfying way to impute them.

4. We save the resulting dataframes to CSV files for modelling.

4.8 Resulting datasets

After completing all these steps, we obtain two datasets. The first one is the training dataset, which includes all the game data from previous tournaments with the newly created features. The second dataset is the solution dataset, which contains possible games from this year's tournament. Our goal is to predict the probabilities for each game in the solution dataset and submit them for the competition. The attributes that ended up in the final datsets, spit into categories are:

1. Identification columns
   - **T1_TeamID**, **T2_TeamID**
   - **Season**
   - **ID** - (ONLY FOR THE SOLUTION DATASET) unique game ID.
2. Team's performance during the regular season split by game location:
   - **T1_FG%_home**, **T1_FG%_away** | **T2_FG%_home**, **T2_FG%_away**
   - **T1_3P%_home**, **T1_3P%_away** | **T2_3P%_home**, **T2_3P%_away**
   - **T1_FG%_home**, **T1_FG%_away** | **T2_FG%_home**, **T2_FG%_away**
   - **T1_point_diff_home**, **T1_point_diff_away** | **T2_point_diff_home**, **T2_point_diff_away**
   - **T1_win_home**, **T1_win_away** | **T2_win_home**, **T2_win_away**
3. Team's performance from the regular season:
   - **T1_Score_mean**, **T2_Score_mean**
   - **T1_FGM_mean**, **T2_FGM_mean**
   - **T1_FGM3_mean**, **T2_FGM3_mean**
   - **T1_FGA_mean**, **T2_FGA_mean**
   - **T1_FGA3_mean**, **T2_FGA3_mean**
   - **T1_OR_mean**, **T2_OR_mean**
   - **T1_DR_mean**, **T2_DR_mean**
   - **T1_AST_mean**, **T2_AST_mean**
   - **T1_TO_mean**, **T2_TO_mean**
   - **T1_Stl_mean**, **T2_Stl_mean**
   - **T1_Blk_mean**, **T2_Blk_mean**
   - **T1_PF_mean**, **T2_PF_mean**
   - **T1_win_mean**, **T2_win_mean**
   - **T1_win_ratio_last_month**, **T2_win_ratio_last_month**
   - **T1_OrdinalRank**, **T2_OrdinalRank** - (FOR MEN ONLY)
   - **T1_Ranking**, **T2_Ranking** - (FOR WOMEN ONLY)
4. Average performance of the teams that the given team faced during the regular season:
   - **T1_opponent_Score_mean**, **T2_opponent_Score_mean**
   - **T1_opponent_FGM_mean**, **T2_opponent_FGM_mean**
   - **T1_opponent_FGM3_mean**, **T2_opponent_FGM3_mean**
   - **T1_opponent_FGA_mean**, **T2_opponent_FGA_mean**
   - **T1_opponent_FGA3_mean**, **T2_opponent_FGA3_mean**
   - **T1_opponent_OR_mean**, **T2_opponent_OR_mean**

- **T1_opponent_DR_mean**, **T2_opponent_DR_mean**
- **T1_opponent_AST_mean**, **T2_opponent_AST_mean**
- **T1_opponent_TO_mean**, **T2_opponent_TO_mean**
- **T1_opponent_Stl_mean**, **T2_opponent_Stl_mean**
- **T1_opponent_Blk_mean**, **T2_opponent_Blk_mean**
- **T1_opponent_PF_mean**, **T2_opponent_PF_mean**

5. Conference influence:
   - **T1_conference_win_margin**, **T2_conference_win_margin**
   - **T1_conference_point_margin**, **T2_conference_point_margin**
6. Past tourney data:
   - **T1_played_prev_tourney**, **T2_played_prev_tourney**,
   - **T1_prev_win_ratio**, **T2_prev_win_ratio**,
   - (FOR MEN ONLY):
     - **T1_top1**, **T2_top1**
     - **T1_top2**, **T2_top2**
     - **T1_top4**, **T2_top4**
     - **T1_top8**, **T2_top8**
     - **T1_top16**, **T2_top16**
     - **T1_top32**, **T2_top32**
     - **T1_top64**, **T2_top64**
     - **T1_top68**, **T2_top68**
     - **T1_total**, **T2_total**
7. Other:
   - **T1_Seed**, **T2_Seed**
   - **seed_difference**
   - **win** - target attribute

5 Modelling

Having prepared all necessary datasets we entered the modelling phase of our project. A little research regarding best scores from last years' competitions as well as our personal preferences made us believe that XGBoost classifier would be the best pick for the model. However, we tried a couple different approaches, which we also describe below.
Before we trained our models it was essential to understand the competition's evaluation rules. Although the predicted variable is binary, the evaluation is based upon probabilities not predictions, ie. we submit the probability of Team_1's win and not predicted result. The final evaluation of submission follows the Brier score formula, which essentialy is mean squared error metric between the vector of probabilities and binary vector of games scores, which we want to minimize. We extract probabilities from our predictions using `predict_proba` method of XGBoost classifier.

5.1 XGBoost approach

The first task was to decide how to divide our past data into training and testing subsets. We considered two approaches: dividing randomly according to predetermined ratio and treating n last seasons as test set and all previous as training set. Hoping to maybe catch some time-dependent trends in our data we chose the second approach. Since the data for men tourney has a few more seasons the test set contains last four seasons (2018, 2019, 2021, 2022), whereas for women tourney - last three (2019, 2021, 2022).

5.1.1 Men tourney model

The initially trained model was an XGBoost Classifier with a couple preset hyperparameters: `learning_rate`=0.01 and `n_estimators`=800. This model gave us the following results on the test set:

- Accuracy score of 0.6686
- Brier score of 0.2189

We later tuned selected hyperparameters using grid search method along with crossvalidation. The crossvalidation was performed on 5 folds and the grid consisted of the following parameters and potential values:

- `n_estimators`: [200, 500, 800]
- `learning_rate`: [0.1, 0.01, 0.001]
- `max_depth`: [3, 4, 5, 6]

Interestingly enough, it turned out that our initial model is the best one out of 48 possible. Final results are therefore the ones presented above. Apart from training and tuning the model we were also interested, which features contributed the most to the predictions. To do that we plotted feature importance of our model (specifically `gain` statistic), which looks as follows:

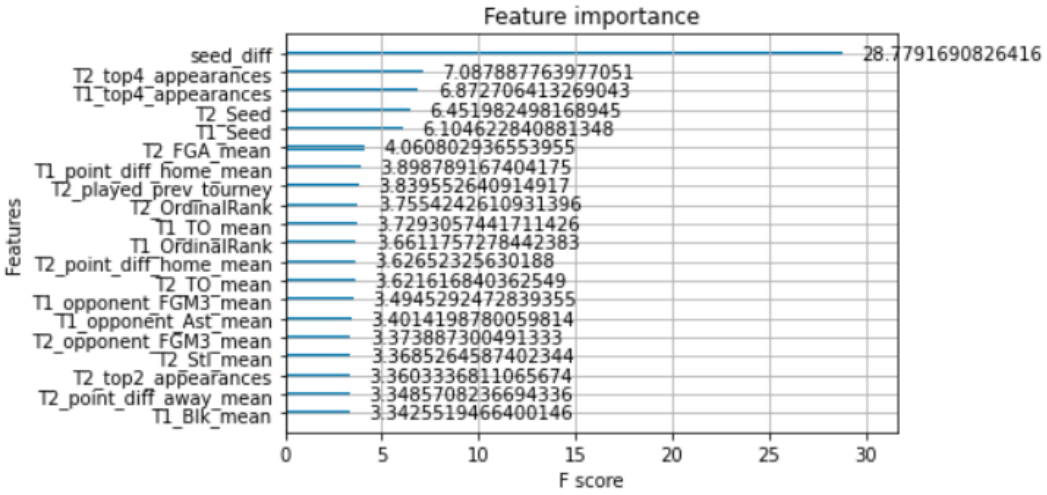

Figure 1: Feature importance - men tourney model

As we can see, the seed_diff feature is by far the most important one, leaving all other far behind. This is an interesting insight, telling us that most of the detailed statistics don't matter that much and the favourite team usually wins.

5.1.2 Women tourney model

The training and tuning model looked exactly the same as for men tourney model. Initial scpres on test set were as follows:

- Accuracy score of 0.7539
- Brier score of 0.1648

The hyperparameter tuning slightly improved the results this time. Selected parameters were:

- `n_estimators`=200
- `learning_rate`=0.01
- `max_depth`=6

And obtained results:

- Accuracy score of 0.7539
- Brier score of 0.1602

It is worth noting that the results are significantly better for women tourney than they are for men tourney. This is consistent with most of the submissions on Kaggle and probably related to lower unpredictability and less number of surprises in women's league. Feature importance plot for this model is as follows:

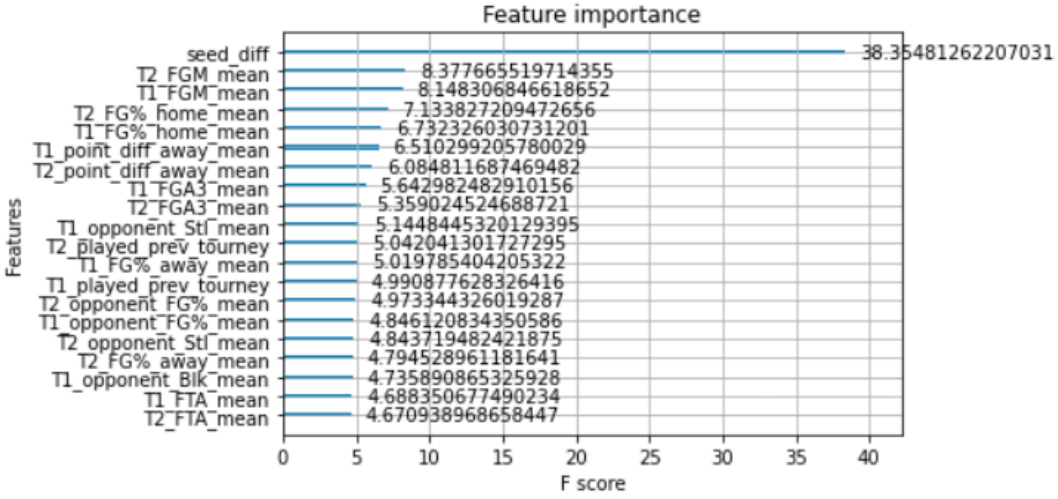

Figure 2: Feature importance - women tourney model

The same conlusions as before can be drawn.

5.1.3 Submission results

Having tuned the models, we trained them on this year's data. Thanks to the fact, that we entered the competition after the games had already concluded, we were able to instantly check our results:

- Accuracy score of 0.7143 (men: 0.6825, women: 0.7460)
- Brier score of 0.1936 (men: 0.2176, women: 1697)

The scores are close to the ones obtained on the test set, which means our model is properly tuned and does well on unseen data. We also compared our score to other Kaggle competitors. The median Brier score of official submissions is equal to 0.1984, while the best - 0.1737. Our submission would have given us 437th place out of 1033 teams total, which is above average.

5.2 Prediction based on seed_diff feature

Using the insights from XGBoost model, we wanted to see how a simple model, predicting a win for the better seeded team would perform. The model did a pretty good job, getting following results for men tourney:

- Accuracy score of 0.6666
- Brier score of 0.3333

And for women tourney:

- Accuracy score of 0.7797
- Brier score of 0.2202

As we can see, accuracy scores are comparable with the ones obtained with XGBoost (for women tourney they are even slightly better). However, in the Brier score metric that interests us most, the model did not perform so well, which is not a surprise given the fact that in this approach we treated predictions as probabilities, which inflates the error for wrong predictions.

5.3 Summary and Conclusions

In this project, we took on the challenge of participating in the Kaggle March Madness competition. After understanding the problem and selecting relevant data, we constructed a feature engineering pipeline. Our objective was to predict the probabilities of match outcomes in this year's tournament. We employed various machine learning models to generate these predictions, and our best-performing model was XGBoost. Notably, our results surpassed those obtained through a simple prediction method based solely on the seed differences of teams. In conclusion, we would have placed 437th out of 1033 teams in the Kaggle competition.

References

[1] Kaggle. March Machine Learning Mania 2023. https://www.kaggle.com/competitions/march-machine-learning-mania-2023/, 2023.

[2] Wikipedia. NCAA Division I men's basketball tournament. https://en.wikipedia.org/wiki/NCAA_Division_I_men%27s_basketball_tournament, 2023.

[3] Kaggle. NCAA Women 538 team ratings. https://www.kaggle.com/datasets/raddar/ncaa-women-538-team-ratings/, 2023.

[4] Kaggle. March Madness Data. https://www.kaggle.com/datasets/nishaanamin/march-madness-data, 2023.