




MARCH MACHINE LEARNING MANIA 2024

FORECASTING THE 2024 MEN'S COLLEGE BASKETBALL TOURNAMENTS



THE PROBLEM

OUR GOAL IS TO PREDICT THE
RESULTS OF THE MEN'S 2024
COLLEGE BASKETBALL
TOURNAMENTS BY SUBMITTING A
COLLECTION OF BRACKETS
INFORMED BY HISTORICAL DATA,
WHICH WILL BE A CLASSIFICATION
PROBLEM.



TACKLING THE PROBLEM



The Problem was tackled in a roundabout fashion.



Instead of focusing on the bracket itself. We focused on predicting between two teams who would win, given their known stats, and ratings



Why?



Simplification: By breaking down the problem, we simplified the prediction task, making it more tractable for machine learning models. Predicting single games reduces complexity and allows for more precise modeling.



Data Utilization: This approach enabled us to leverage detailed historical performance data of teams, using statistics and rankings to inform our predictions on a game-by-game basis, rather than trying to account for the entire tournament's dynamics at once.

DATA PREPROCESSING

- The majority of time spent on the project was spent on preprocessing.
- Firstly, we split all data by seasons.
- Then we removed all seasons which did not include all data sets. (1985 – 2002).
- Next, we aggregated all stats from the MRegularSeasonDetailedResults to get for and against stats, for each team, for each one of our 4 data variants AVG Stats, AVG Stats with Rating, AVG 10, AVG 10 W Rating.
- Then for each game in that season, both team's stats we aggregated into our matchup's dataset.

MTeamSpellings.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	23 KB
MTeams.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	10 KB
MTeamConferences.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	220 KB
MTeamCoaches.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	384 KB
MSecondaryTourneyTeams.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	27 KB
MSecondaryTourneyCompactResults.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	59 KB
MSeasons.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	2 KB
MRegularSeasonDetailedResults.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	10,929 KB
MRegularSeasonCompactResults.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	5,341 KB
MNCAATourneySlots.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	48 KB
MNCAATourneySeeds.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	37 KB
MNCAATourneySeedRoundSlots.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	15 KB
MNCAATourneyDetailedResults.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	129 KB
MNCAATourneyCompactResults.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	72 KB
MMasseyOrdinals.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	111,897 KB
MGameCities.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	2,525 KB
MConferenceTourneyGames.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	160 KB
Conferences.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	2 KB
Cities.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	10 KB
2024_tourney_seeds.csv	✓	3/7/2024 10:55 PM	Microsoft Excel C...	2 KB

Season	DayNum	WTeamID	WScore	LTeamID	LScore	WLoc	NumOT	WFGM	WFGA	WFGM3	WFGA3	WFTM	WFTA	WOR	WDR	WAst	WTO	WStl	WBlk	WPF	LFGM	LFGA	LFGM3	LFGA3	LFTM	LFTA	LOR	LDR	LAst	LTO	LStl	LBlk	LPF
2013	4	1104	70	1355	67	H	0	25	52	6	21	14	14	8	19	14	12	8	4	16	25	50	8	21	9	12	8	19	8	13	3	2	14

Example row of MRegularSeasonDetailedResults

VARIANT 1: AVERAGE STATS PER SEASON

Our baseline data set captures the average statistical performance of each team for every season. It aggregates team-level data across all games in a season, such as average points scored, average points allowed, shooting attempts, fouls and rebounding numbers.

Features for Both Teams Per Matchup:

- TeamID, FGM, FGA, FGM2, FGA2, FGM3, FGA3, FTM, FTA, OR, DR, Ast, TO, Stl, Blk, PF, FGM_A, FGA_A, FGM2_A, FGA2_A, FGM3_A, FGA3_A, FTM_A, FTA_A, OR_A, DR_A, Ast_A, TO_A, Stl_A, Blk_A, PF_A

Range: Roughly 62 Features.



VARIANT 2: AVERAGE STATS AND RANKING PER SEASON

Building upon the first variant, this dataset incorporates both the average statistics and ratings of teams for that season.

Adding ratings to the average statistical performance provides context to the raw numbers, placing team performance within the broader competitive landscape. Ratings offer a comparative metric, indicating not just how a team performs but how its performance stacks up against others.

Note. There are some inaccuracies in the ratings as some ratings are not available until weeks into the season. To combat these unknown ratings values were backfilled where possible and if not interpolated. If ratings are still unknown for that system, then that column was dropped.

Features:

Similar to Variant 1 + Ratings

Range: 62 – 180(Depending on if a specific systems rated every team)



VARIANT 3: AVERAGE LAST 10 GAMES

This variant focuses on the stats of the most recent 10 games played by a team before a given matchup. It captures the same set of statistics as the first variant but limits the data to the last 10 games, providing a snapshot of a team's form heading into a game.

Benefits:

The recent form of a team can be a significant indicator of its current performance level, especially heading into key games. This variant allows the model to weigh recent performances more heavily, potentially capturing momentum or changes in team dynamics.

Features:

Same as Variant 1



LAST 10 GAMES WITH CURRENT RATINGS

Like Variant 3, this dataset focuses on the last 10 games but also includes current ratings and rankings at the time of each game. This addition offers a real-time perspective on how teams were viewed in the context of their recent performances

Benefits:

Including current ratings with recent game performances offers a dynamic view of a team's status, combining the immediate past performance with the contemporary perception of team strength. This variant is particularly useful for capturing the impact of late-season changes, injuries, or strategic adjustments.

Features:

Same as Variant 2



SIZE ON THE DISK:
712MB

NUMBER OF TRAINING INSTANCES:
20

NUMBER OF FEATURES:
60-180

RANGE:
2

INFORMATION ON THE DATASET



The background image shows a computer monitor with a data table and a sidebar menu. The table has columns labeled 'id', 'name', 'age', 'weight', 'height', 'gender', 'team', 'position', 'salary', and 'experience'. The sidebar menu includes items like 'Dashboard', 'Users', 'Teams', 'Positions', 'Salaries', 'Experiences', 'Gender', 'Height', 'Weight', 'Age', 'Name', and 'ID'. The foreground features several basketballs on a wooden floor.

id	name	age	weight	height	gender	team	position	salary	experience
1	John	25	180	190	M	Team A	Guard	10000	2
2	Jane	28	160	170	F	Team B	Forward	12000	3
3	Mike	30	200	200	M	Team C	Center	15000	4
4	Sarah	22	150	165	F	Team D	Guard	8000	1
5	David	32	220	210	M	Team E	Center	18000	5
6	Emily	26	170	180	F	Team F	Forward	11000	3
7	Chris	29	190	195	M	Team G	Guard	9000	2
8	Alex	31	210	205	M	Team H	Center	16000	4
9	Olivia	24	165	175	F	Team I	Forward	10000	2
10	Ben	27	185	190	M	Team J	Guard	11000	3
11	Mia	23	155	170	F	Team K	Forward	9000	1
12	Ethan	33	230	215	M	Team L	Center	19000	6
13	Sophia	25	175	185	F	Team M	Guard	12000	3
14	Lucas	28	195	200	M	Team N	Forward	13000	4
15	Isabella	21	150	165	F	Team O	Guard	7000	1
16	Noah	34	240	220	M	Team P	Center	20000	7
17	Ava	26	170	180	F	Team Q	Forward	11000	3
18	Liam	29	190	195	M	Team R	Guard	10000	2
19	Ella	23	155	170	F	Team S	Forward	9000	1
20	Oliver	35	250	225	M	Team T	Center	21000	8



ALGORITHMS USED

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost
- MLP Classifier (Neural network)
- Naïve Bayes

HOW THE ALGORITHMS PERFORMED



THE BASELINE

THE INITIAL AVERAGE PERFORMANCE –
LOGISTIC REGRESSION

- The initial assessment of the average performance of the Logistic Regression algorithm on our dataset was 0.73.
- The initial feature configuration was a test using the model as a general test.

ALGORITHM TRAINING & CONTINUOUS EVALUATION

ADDITIONAL ALGORITHMS USED

HYPERPARAMETER TUNING

```
param_grid_mlp = {
    'hidden_layer_sizes': [(60), (30,15), (60,15), (60,30,15)], # Sizes of the hidden layers
    'activation': ['tanh', 'relu', 'logistic'], # Activation function for the hidden layer
    'solver': ['sgd', 'adam'], # The solver for weight optimization
    'learning_rate': ['constant', 'adaptive', 'invscaling'], # Learning rate schedule for weight updates
    'max_iter': [25, 50, 100, 200, 300], # Maximum number of iterations
}
MLPClassifier(random_state = 3270, hidden_layer_sizes = (60,30,15), max_iter = 25, activation = 'logistic', learning_rate = 'invscaling')
```

```
param_grid_dt = {
    'max_depth': [None, 10, 20, 30, 40], # Maximum depth of the tree
    'min_samples_split': [2, 5, 10, 20], # Minimum number of samples required to split an internal node
    'criterion': ['gini', 'entropy'], # Function to measure the quality of a split
}
DecisionTreeClassifier(random_state=3270, max_depth=10, min_samples_split=10)
```

```
param_grid_rf = {
    'n_estimators': [10, 50, 100, 200], # Number of trees in the forest
    'max_depth': [None, 10, 20, 30], # Maximum depth of the tree
    'min_samples_split': [2, 5, 10], # Minimum number of samples required to split an internal node
}
RandomForestClassifier(random_state=3270, n_estimators=200, max_depth=10, min_samples_split=10, n_jobs=-1)
```

```
param_grid_lr = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100], # Regularization strength
    'penalty': ['l1', 'l2'], # Type of norm used in the penalization
    'solver': ['liblinear', 'saga'], # Algorithm to use for optimization
    'max_iter': [100, 50, 1000], # Maximum number of iterations
}
LogisticRegression(random_state=3270, max_iter=1000, penalty = None, solver = 'lbfgs')
```

```
param_grid_xgb = {
    'n_estimators': [100, 200, 300], # Number of gradient boosted trees
    'learning_rate': [0.01, 0.1, 0.2], # Boosting learning rate (xgb's "eta")
    'max_depth': [3, 6, 9], # Maximum depth of a tree
    'subsample': [0.8, 1], # Subsample ratio of the training instances
    'colsample_bytree': [0.8, 1], # Subsample ratio of columns when constructing each tree
}
XGBClassifier(random_state = 3270, n_estimators = 100, max_depth = 3, learning_rate = 0.1, gamma = 0, subsample = 0.8, colsample_bytree = 0.8)
```

GaussianNB()

ALGORITHMS USED - CLASSIFICATION

Top Performers

- Logistic Regression – 77.3%
- MLP Classifier – 77.2%
- XGBoost – 76.0%

Worst Performers

- Naïve Bayes – 75.3%
- Random Forest – 75.1%
- Decision Tree – 68.0%



INTERPRETING THE TOP PERFORMER

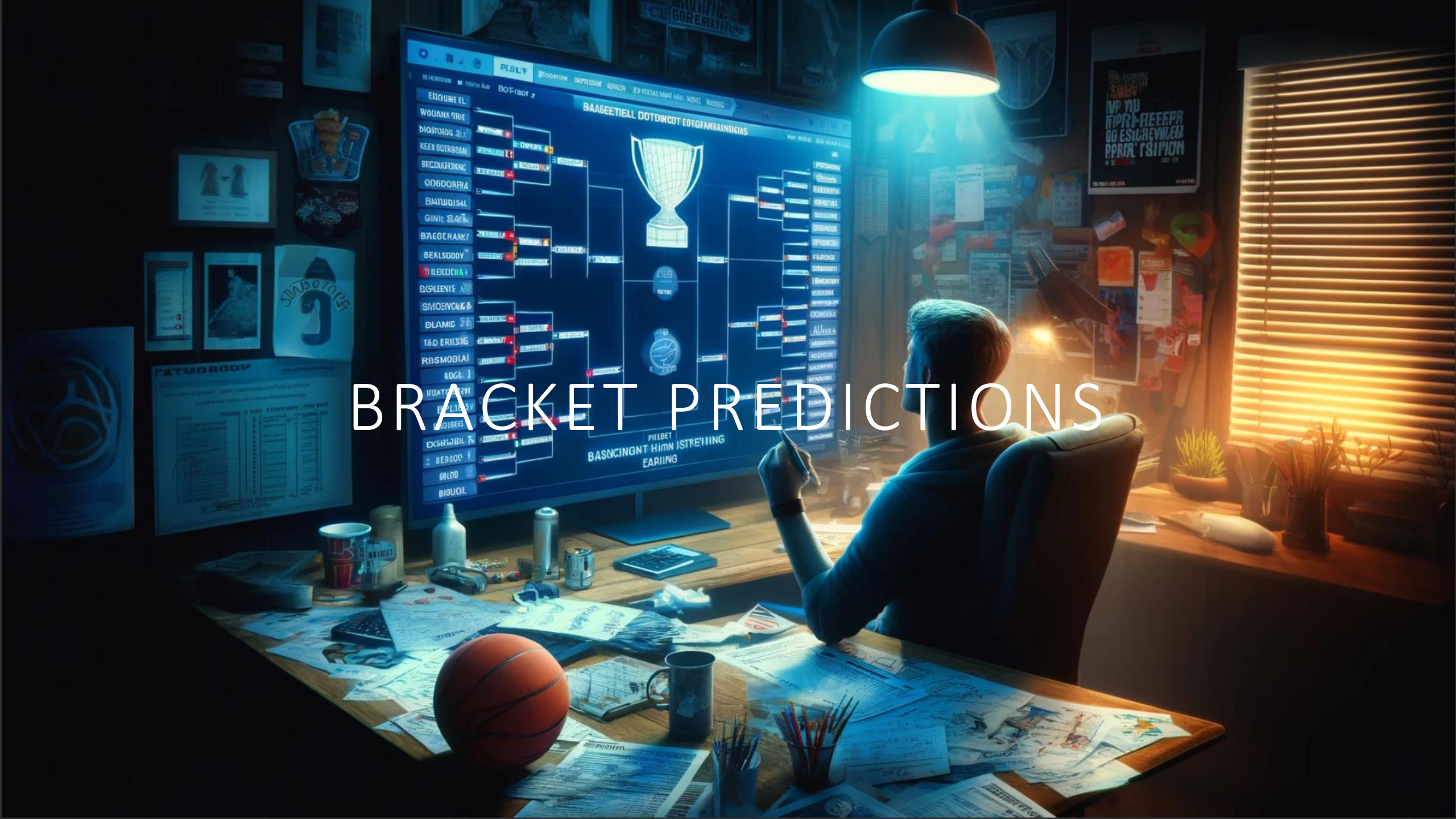
- OUR TOP PERFORMING ALGORITHM WAS LOGISTIC REGRESSION, WITH AN ACCURACY OF 77.3%.
- TOP PERFORMING CONFIGURATIONS
 - LOGISTIC REGRESSION WITHOUT RATINGS
 - MLP CLASSIFICATION WITH RATINGS
 - XGBOOST WITH RATINGS



AVERAGES FOR EACH ALGORITHM

Data Variants	Averages	Averages with Ratings	Average of Past 10 Games	Average of Past 10 Games W Rating
Naïve Bayes	0.694963	0.717608	0.743741	0.752905
Logistical Regression	0.735479	0.730485	0.776097	0.772723
Decision Tree	0.642981	0.644373	0.692437	0.680367
Random Forest	0.726145	0.713783	0.756219	0.751237
XGBoost	0.719841	0.718567	0.755495	0.760408
MLP	0.731234	0.729159	0.770152	0.772158

BRACKET PREDICTIONS





HOW?

Aside from the team's statistical data two other datasets were used in predicting of the tournament bracket.

MNCAATourneySeeds:

Teams are represented by **Seed** a Letter(W,X,Y,Z) followed by a number(1-16) and sometimes follow by a

Season	Seed	TeamID	2021 W11a	1277	or aR1 slot
2021	W01	1276	2021 W11b	1417	
2021	W02	1104	2021 W12	1207	
2021	W03	1400	2021 W13	1422	

MNCAATourneySlots:

This shows for each **Slot** in the tournament who would be going up against each other for that specific slot as StrongSeed and the WeakSeed.

Season	Slot	StrongSeed	WeakSeed	2021 R5WX	R4W1	R4X1	R2W3
2021	R1W1	W01	W16	2021 R5YZ	R4Y1	R4Z1	onship game which is R6CH,
2021	R1W2	W02	W15	2021 R6CH	R5WX	R5YZ	
2021	R1W3	W03	W14	2021 W11	W11a	W11b	
2021				2021 W16	W16a	W16b	

Given a base TourneySeeds the model would first predict the play-ins game like Slot W11(W11a vs W11b). The 'model' pulls in the team's most recent stats from its respective environment. Then makes a prediction on who wins the winners of the playins team now gain that slot as their '**Seed**'. A similar process happens for every slot up to R6CH the championship game.



BRACKET STRONG SEED ACCURACIES

Data Variants	Averages	Averages with Ratings	Average of Past 10 Games	Average of Past 10 Games W Rating
Naïve Bayes	.73	0.85	0.73	0.85
Logistical Regression	0.75	0.85	0.76	0.83
Decision Tree	0.2	0.86	0.8	0.76
Random Forest	0.75	0.85	0.75	0.85
XGBoost	0.78	0.85	0.73	0.87
MLP	0.8	0.85	0.76	0.8



BRACKET WEAK SEED ACCURACIES

Data Variants	Averages	Averages with Ratings	Average of Past 10 Games	Average of Past 10 Games W Rating
Naïve Bayes	.63	0.67	0.75	0.57
Logistical Regression	0.62	0.6	0.82	0.56
Decision Tree	0.65	0.72	0.71	0.43
Random Forest	0.63	0.7	0.76	0.58
XGBoost	0.63	0.72	0.8	0.6
MLP	0.63	0.62	0.78	0.53



BRACKET SLOT WINNER ACCURACIES

Data Variants	Averages	Averages with Ratings	Average of Past 10 Games	Average of Past 10 Games W Rating
Naïve Bayes	.37	0.46	0.35	0.57
Logistical Regression	0.4	0.43	0.43	0.56
Decision Tree	0.35	0.5	0.37	0.43
Random Forest	0.37	0.5	0.37	0.58
XGBoost	0.4	0.5	0.32	0.6
MLP	0.43	0.46	0.42	0.53





TRICKS

IMPROVEMENTS?



LOOKING TO THE FUTURE

```
Training Logistic Regression model...
Log Regression Season 2004 Top Half accuracy: 0.81444, Bottom Half accuracy: 0.75536
Overall accuracy: 0.78473
Log Regression Season 2005 Top Half accuracy: 0.80188, Bottom Half accuracy: 0.74240
Overall accuracy: 0.77198
Log Regression Season 2006 Top Half accuracy: 0.80572, Bottom Half accuracy: 0.73759
Overall accuracy: 0.77170
Log Regression Season 2007 Top Half accuracy: 0.80127, Bottom Half accuracy: 0.74494
Overall accuracy: 0.77315
Log Regression Season 2008 Top Half accuracy: 0.80821, Bottom Half accuracy: 0.73964
Overall accuracy: 0.77397
Log Regression Season 2009 Top Half accuracy: 0.80831, Bottom Half accuracy: 0.72752
Overall accuracy: 0.76796
Log Regression Season 2010 Top Half accuracy: 0.81072, Bottom Half accuracy: 0.74154
Overall accuracy: 0.77617
Log Regression Season 2011 Top Half accuracy: 0.81510, Bottom Half accuracy: 0.72665
Overall accuracy: 0.77087
Log Regression Season 2012 Top Half accuracy: 0.81721, Bottom Half accuracy: 0.75857
Overall accuracy: 0.78774
Log Regression Season 2013 Top Half accuracy: 0.81617, Bottom Half accuracy: 0.73383
Overall accuracy: 0.77500
Log Regression Season 2014 Top Half accuracy: 0.80492, Bottom Half accuracy: 0.75121
Overall accuracy: 0.77807
Log Regression Season 2015 Top Half accuracy: 0.80650, Bottom Half accuracy: 0.74636
Overall accuracy: 0.77643
Log Regression Season 2016 Top Half accuracy: 0.80551, Bottom Half accuracy: 0.73174
Overall accuracy: 0.76849
Log Regression Season 2017 Top Half accuracy: 0.79681, Bottom Half accuracy: 0.75158
Overall accuracy: 0.77424
Log Regression Season 2018 Top Half accuracy: 0.80903, Bottom Half accuracy: 0.73390
Overall accuracy: 0.77151
Log Regression Season 2019 Top Half accuracy: 0.80520, Bottom Half accuracy: 0.73563
Overall accuracy: 0.77027
Log Regression Season 2020 Top Half accuracy: 0.79955, Bottom Half accuracy: 0.73574
Overall accuracy: 0.76764
Log Regression Season 2021 Top Half accuracy: 0.79657, Bottom Half accuracy: 0.72600
Overall accuracy: 0.76135
Log Regression Season 2022 Top Half accuracy: 0.79416, Bottom Half accuracy: 0.76796
Overall accuracy: 0.78110
Log Regression Season 2023 Top Half accuracy: 0.77936, Bottom Half accuracy: 0.73188
Overall accuracy: 0.75562
Log Regression Season 2024 Top Half accuracy: 0.80493, Bottom Half accuracy: 0.73347
Overall accuracy: 0.76920
```

We noticed in our training that our models were very good at predicting earlier games in the season compared to later games in the season. We will be looking into why that is.

We would also like to our current models on actual previous tournament data to see if it can predict. Or even create and train new models on only tournament data, to in hops it can learn the nuances and unpredictability of tournament games.

Creating an inhouse ensemble which not only takes a model's prediction into account but also weight. Where weight would be its accuracy in past predictions. This we believe would allow for more accurate "upset" predictions.



THANK YOU

Project 1

Team: The Overfitting Overlords Taking this to Vegas.

Members: Anton Maynard, Larry Jones, & Kenneth Mitchell

Presentation Date: April 4th, 2024