Project Overview

This project's primary goal was to harness the power of machine learning to predict outcomes of the Bracket for the NCAA March Madness competition for both Men and Women. This was achieved by analyzing past game results, team statistics, and other relevant metrics. We sought to develop models capable of accurately forecasting which team would emerge victorious in future contests. This endeavor is not just a statistical challenge but also a strategic tool that can be leveraged for planning, betting, analytics, and fan engagement.

Data Collection and Preparation

Datasets

There were several key datasets that were integral to our analysis.

- **Tourney Seeds**: This dataset provides the seeding information for teams in the tournament, an essential aspect of understanding the competitive dynamics and rankings within the tournament framework.
- **Season Data**: Detailed records of team performances across different seasons, providing a rich source of statistics for feature generation.
- **Matchup Data**: A crucial dataset constructed to encapsulate the specific pairings of teams in historical matchups. This data includes identifiers for each team involved in a match, the outcome of the match, and any relevant context such as game location or tournament stage. Creating this dataset involved aggregating match records, identifying team pairings, and encoding the results in a manner conducive to predictive modeling.
- **Team Stats Data**: Derived from the season data, the team stats dataset was meticulously engineered to provide a granular view of each team's capabilities and tendencies. This included aggregating season-long statistics to capture average performance metrics, such as points per game, shooting percentages, and defensive stops, offering a comprehensive profile of team strengths and weaknesses.

Preprocessing Steps

Feature Engineering

From the season data, we crafted a comprehensive set of features to capture the strengths, weaknesses, and overall performance trends of the teams. This included direct statistics and derived metrics designed to reflect aspects significant to game outcomes.

Data Splitting

A time-series approach was adopted for splitting the data, ensuring the chronological integrity of the training and validation sets. This method respected the temporal nature of the sports data, avoiding lookahead bias.

Differentiation Between Women's and Men's Competitions

The inherent differences in playing styles, team dynamics, and structures of competitions between women's and men's sports necessitated a tailored approach:

- **Specialized Models**: Separate models were developed for both women's and men's datasets to allow for the nuanced learning of patterns and key predictive elements relevant to each category.
- **Adapted Feature Engineering**: Recognizing the distinctions between the competitions, our feature engineering process was meticulously adapted to reflect the unique attributes of each dataset, ensuring the models were precisely attuned to the specificities of the data.

Data Variants

In pursuit of creating a predictive model that encapsulates the complex dynamics of team performances, we further divided our dataset into four unique variants.

Variant 1: Average Stats per Season

- **Description**: 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, providing metrics such as average points scored, average points allowed, shooting percentages, and rebounding numbers.
- **Purpose**: The average stats per season variant gives a broad overview of a team's performance, highlighting strengths and weaknesses over a season. This variant is crucial for understanding long-term team trends and season-to-season performance fluctuations.
- **Features**: TeamID, FGM, FGA, FGM2, FGA2, FGM3, FGA3, FTM, FTA, OR, DR, Ast, TO, Stl, Blk, PF, FGMA, FGAA, FGM2A, FGA2A, FGM3A, FGA3A, FTMA, FTAA, ORA, DRA, AstA, TOA, StlA, BlkA, PFA

Variant 2: Average Stats and Ranking per Season

- **Description**: Building upon the first variant, this dataset incorporates both the average statistics and ratings of teams per season. Ratings similar to ELO were derived from various sources, reflecting a team's standing in national polls and analytical rankings.
- **Purpose**: 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.
- **Features**: Variant 1. Plus: Average Rating per system. I.E: '7OT', 'ARG', 'ATP'

  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 unknown for that system, then that column was dropped.

Variant 3: Last 10 Games

- **Description**: This variant focuses on the most recent 10 games played by a team in a season. 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.
- **Purpose**: 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**: See Variant 1

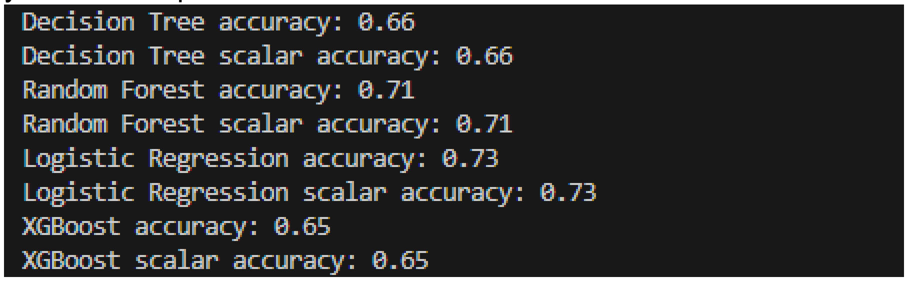
Variant 4: Last 10 Games with Current Ratings

- **Description**: 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.
- **Purpose**: 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**: See Variant 2

Model Development Journey

Initial Approach

Our exploratory phase began with a selection of models chosen for their distinct characteristics and potential suitability for our dataset. We started with *Logistic Regression, Decision Tree, and Random Forest*, appreciating Random Forest for its ensemble approach, which often yields robust performance.



```
Decision Tree accuracy: 0.66
Decision Tree scalar accuracy: 0.66
Random Forest accuracy: 0.71
Random Forest scalar accuracy: 0.71
Logistic Regression accuracy: 0.73
Logistic Regression scalar accuracy: 0.73
XGBoost accuracy: 0.65
XGBoost scalar accuracy: 0.65
```

Figure 1 Accuracies before Hyperparameter Tuning

As our understanding deepened, we expanded our arsenal to include *Multilayer Perceptron (MLP), XGBoost*, and *Gaussian Naive Bayes*, aiming to explore a broader spectrum of modeling techniques.

Evaluation Methods

To assess model performance, we employed a dual approach. Initially, we tested models against the *Average_stats* variant and conducted a k-fold validation on all season data, focusing on accuracy as our primary metric. This method allowed us to gauge each model's baseline performance and identify promising candidates for further refinement.

Custom Cross-Validation

Recognizing the unique structure of our data, we developed a custom cross-validation function tailored for our needs. This function iteratively trained models on data from previous seasons and tested them on the current season, progressing sequentially up to 2024. This season-by-season approach, as opposed to arbitrary data chunks, ensured that our validation process mirrored the real-world scenario of predictive modeling in sports. For intra-season analyses, we utilized Scikit-learn's built-in time series cross-validation, benefiting from its efficiency in smaller-scale tests.

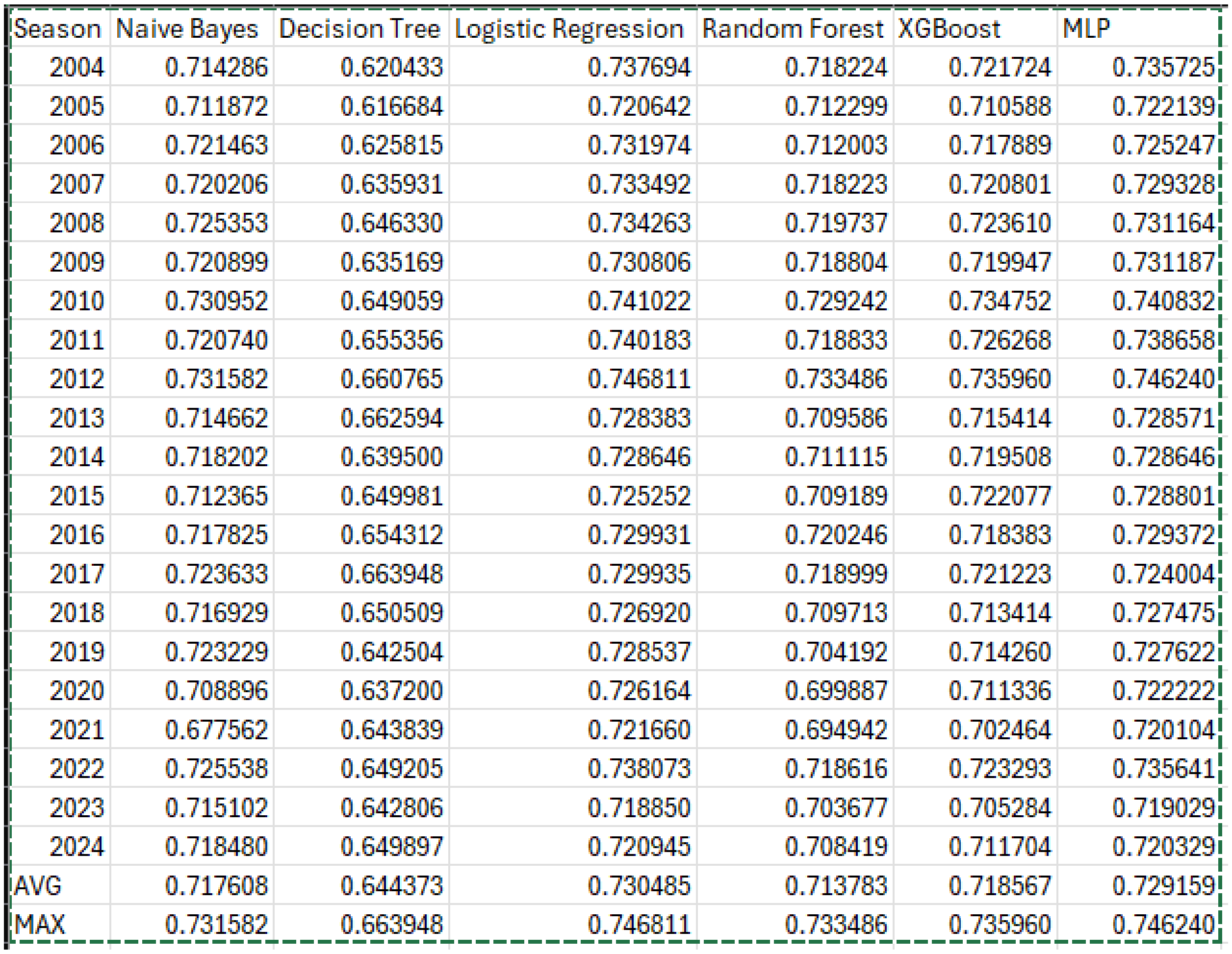
Hyperparameter Tuning

Through **GridSearchCV**, we conducted thorough hyperparameter optimization, seeking the ideal combination of parameters for optimal performance. This step was crucial in refining the models for better accuracy.

- **MLP**: Adjusting the activation rate to ***'logistic'*** and changing the learning rate strategy to ***'invscaling'*** cumulatively enhanced the MLP model's accuracy by 10%, bringing it to an impressive 75%.
- **XGBoost, Decision Tree, and Random Forest**: Reducing the **Max Depth** parameter from 100 to 10 significantly improved accuracy. This adjustment likely helped mitigate overfitting, allowing these models to generalize better to unseen data. Which increased it accuracy by 7% bringing it to 74%

These hyperparameter optimizations were instrumental in refining our models, with the MLP model showing substantial gains from these adjustments.

Sadly due to time and hardware constraints we could not do unique Hyperparameter tuning for each data Variant so the base line ‘Average_stats’ variant was used.



Season	Naive Bayes	Decision Tree	Logistic Regression	Random Forest	XGBoost	MLP
2004	0.714286	0.620433	0.737694	0.718224	0.721724	0.735725
2005	0.711872	0.616684	0.720642	0.712299	0.710588	0.722139
2006	0.721463	0.625815	0.731974	0.712003	0.717889	0.725247
2007	0.720206	0.635931	0.733492	0.718223	0.720801	0.729328
2008	0.725353	0.646330	0.734263	0.719737	0.723610	0.731164
2009	0.720899	0.635169	0.730806	0.718804	0.719947	0.731187
2010	0.730952	0.649059	0.741022	0.729242	0.734752	0.740832
2011	0.720740	0.655356	0.740183	0.718833	0.726268	0.738658
2012	0.731582	0.660765	0.746811	0.733486	0.735960	0.746240
2013	0.714662	0.662594	0.728383	0.709586	0.715414	0.728571
2014	0.718202	0.639500	0.728646	0.711115	0.719508	0.728646
2015	0.712365	0.649981	0.725252	0.709189	0.722077	0.728801
2016	0.717825	0.654312	0.729931	0.720246	0.718383	0.729372
2017	0.723633	0.663948	0.729935	0.718999	0.721223	0.724004
2018	0.716929	0.650509	0.726920	0.709713	0.713414	0.727475
2019	0.723229	0.642504	0.728537	0.704192	0.714260	0.727622
2020	0.708896	0.637200	0.726164	0.699887	0.711336	0.722222
2021	0.677562	0.643839	0.721660	0.694942	0.702464	0.720104
2022	0.725538	0.649205	0.738073	0.718616	0.723293	0.735641
2023	0.715102	0.642806	0.718850	0.703677	0.705284	0.719029
2024	0.718480	0.649897	0.720945	0.708419	0.711704	0.720329
AVG	0.717608	0.644373	0.730485	0.713783	0.718567	0.729159
MAX	0.731582	0.663948	0.746811	0.733486	0.735960	0.746240

Figure 2 Accuracies, after hyperparameter tuning, AVG_Stats Variant

MLP's Success

1. **Algorithm Flexibility**: MLP's ability to model complex, non-linear relationships was crucial, unlike simpler, linear models. It effectively used diverse features, capturing the nuanced dynamics of sports competitions.

2. **Strategic Feature Use**: Utilizing detailed season statistics and recent game performance, including current ratings, enriched the MLP's learning context, enhancing its predictive accuracy.

3. **Overfitting Control**: Adjustments in activation and learning rate (**logistic** and **invscaling**) helped prevent overfitting, ensuring the MLP learned general patterns without memorizing the training data.

4. **Balanced Complexity**: Fine-tuning the MLP's structure to moderate complexity improved its generalization, striking a balance that captured essential patterns without being overly sensitive to training data specifics.