Predicting NBA Game Outcomes with Machine Learning

Michael Murray
Cal Poly, San Luis Obispo
March 4, 2025

Introduction

In the National Basketball Association (NBA), predicting the winners of games is a subject of interest for everyone from the casual fan to professional analysts. This is a difficult statistic to predict due to the large number of factors that contribute to game outcomes. Individual player performance, injuries, and even referee decisions can affect which team comes out on top. All this variability causes traditional statistical prediction methods to fall short.

In this project, I used machine learning (ML) methods to predict NBA game outcomes. By analyzing patterns within historical game data, ML models can learn what factors contribute to a win, and by how much. I experimented with various ML models to predict game outcomes with as high of an accuracy as possible.

Literature Review

Predicting NBA game outcomes using ML methods is a well-researched area. This section will summarize the methodology and results from this work.

A project at Bryant University tested six models – Logistic Regression, Random Forest, K-Nearest Neighbors, Support Vector Classifier, Gaussian Naïve Bayes, and XGBoost – on a dataset with statistics for both the home and away teams. The dataset contained two complex statistics: a 10-game rolling win percentage and an Elo rating. Naïve Bayes was the best performing model, achieving an accuracy of 65.1% after hyperparameter tuning.

Another approach to this problem was taken by Weiner et al. which aimed to use Elo ratings and team statistics as features for training models. They employed Logistic Regression and Random Forest Regression models, finding that the optimized Random Forest model performed best, achieving a testing accuracy of 67.15%. They also experimented with incorporating aggregated player performance into their dataset but found the models did not improve from these statistics.

A different strategy that has been taken relies on player stereotyping. In “Predicting the winning team in basketball: A novel approach,” player types were classified through clustering, and those types were then used to train ML models. This study achieved a prediction accuracy of ~71% on a season not used for training. This approach performed better than most other models which rely on historical game data. The paper also noted the difference in per-team model performance, where teams with abnormal winning percentages such as the Golden State Warriors (83% home and 71% away winning percentage) were much easier to predict than other teams with average win percentages.

An interesting project from UC Santa Barbara used Logistic Regression, SVM, Deep Neural Networks, and Random Forest models to analyze NBA game outcomes. The Random Forest model had the highest accuracy at 83.78%. The high performance of these models compared to the typical range of around 60-70% led me to delve into this paper more. There is a key difference in this dataset, as the models are trained on statistics about the outcome of the games which are being predicted. This means the model is unable to predict future games – to do so it would need data about those games which doesn't exist yet. This project emphasizes the importance of crafting an effective dataset and understanding which data the models should be trained on.

Data Collection and Preprocessing

1. Data Sources

Player performance metrics were sourced from the "NBA player stats since 1950" dataset available on Kaggle. This dataset contains one entry per player per season, providing various performance-related statistics that help assess player contributions to their team. Historical game data was collected using the nba_api library from GitHub, which scrapes game data from NBA.com. This dataset includes detailed game outcomes, team matchups, and other in-game statistics.

2. Datasets

Two datasets were constructed for this project, the Game Dataset and the Game and Player Dataset.

Game Dataset

Contains all NBA games from 2017 to 2023, including various rolling statistics for each team. This dataset was created using the "NBA Win-Loss Modeling Data Prep" example from the nba_api GitHub repository as a reference.

For each season from 2017 to 2023, the dataset was populated by iterating through all NBA teams and collecting every game played. This resulted in an initial dataset where each game had two records—one for the home team and one for the away team. To avoid redundancy, these duplicate entries were merged into a single row per game.

Several rolling statistics were computed to capture recent team performance trends:

- Offensive Efficiency (OE): Measures a team's scoring efficiency per possession, based on the number of possessions the team scored on vs. their total possessions
- Home Win Percentage: Tracks a team's win rate when playing at home
- Away Win Percentage: Tracks a team's win rate when playing in away games
- Total Win Percentage: Represents overall team strength by considering wins across all games played.

- Rolling Scoring Margin: Measures the point differential over the last 3 games to reflect momentum and performance consistency.

- Rolling Offensive Efficiency: Tracks changes in a team's offensive output over the last 3 games, helping to detect recent trends in scoring performance.

The final dataset included these advanced statistics for each team per game while excluding any statistics directly derived from the current game to prevent data leakage.

Game and Player Dataset

With the goal of enhancing predictive capabilities, player statistics were incorporated into the Game Dataset to create the Game and Player Dataset. This allowed for a more detailed analysis of how individual players contribute to overall team performance.

First, I imported the NBA player statistics dataset from Kaggle. Then, for each team in the 2017 season, the following statistics were aggregated:

- Player Efficiency Rating (PER) A composite metric evaluating a player's overall statistical performance.

- Win Shares (WS): Estimates the number of wins a player contributes to their team.

- Box Plus-Minus (BPM): Measures a player's impact per 100 possessions compared to an average player.

- True Shooting Percentage (TS%): Adjusts for three-pointers and free throws to provide a better efficiency measure.

- Field Goals (FG): The total number of shots made, indicating scoring volume.

- Three-Point Percentage (3P%): Reflects a team's three-point shooting efficiency, crucial in modern NBA strategies.

- Free Throws (FT): Measures accuracy from the free-throw line, often a key factor in close games.

- Usage Rate (USG%): Represents the percentage of team plays used by a player while on the floor, useful for assessing ball dominance.

- Steal Percentage (STL%): Estimates how often a player generates steals, indicating defensive effectiveness.

- Turnover Percentage (TOV%): Measures a player's tendency to commit turnovers, which can impact team efficiency.

For each statistic, the mean, sum, and standard deviation were computed at the team level to provide a well-rounded representation of each team’s strengths and weaknesses.

Data Cleaning

After both datasets were created, they were cleaned by removing records containing null values.

Approach

Model Selection

I tested various ML models that are typically good for binary classification problems:

- Naïve Bayes: A probabilistic model that assumes feature independence, useful as a baseline classifier.
- K-Nearest Neighbors (KNN): A distance-based model that classifies based on the majority label of the nearest neighbors. For binary classification, I chose k=2.
- Logistic Regression: A simple yet effective model for binary classification that predicts probabilities.
- Support Vector Machine (SVM): A robust classifier that finds the optimal separating hyperplane.
- Decision Tree: A rule-based model that learns simple if-else conditions to classify data.
- Random Forest: An ensemble of decision trees that improves generalization and reduces overfitting.
- Neural Network: A deep learning model with multiple layers to capture complex feature interactions.

Data Preprocessing

The dataset was split into training (67%) and testing (33%) sets using train_test_split() from scikit learn with stratification to maintain class balance. Features were standardized using StandardScaler() from scikit learn to ensure consistency across models.

Implementation

Most of the models – Naïve Bayes, KNN, Logistic Regression, SVM, Decision Tree, and Random Forest – were implemented in python using Scikit-learn's fit() method. Hyperparameter tuning was performed using GridSearchCV() where applicable.

The neural network is a custom class implemented using the PyTorch library. It consists of three fully connected layers with ReLU activations, dropout layers (p=0.4) to prevent overfitting, and a final sigmoid activation to output probabilities. The Adam optimizer was used with a learning rate of 0.0001 and weight decay of 1e-5. The loss function chosen was BCELoss.

Model Evaluation

To assess model performance, the following metrics were used:

- Accuracy: Measures overall correctness of predictions.
- Precision: Evaluates how many predicted wins/losses were actual correct.
- Recall: Measures how many actual wins/losses were correctly predicted.
- F1-Score: Balances precision and recall providing an overall effectiveness measure.

Results

Table 1. Model results when trained on Game Dataset

Model	**Accuracy**	**Precision (Loss)**	**Precision (Win)**	**Recall (Loss)**	**Recall (Win)**	**F1-score (Loss)**	**F1-score (Win)**
Naïve Bayes	0.63	0.58	0.67	0.55	0.69	0.57	0.68
K-Nearest Neighbors (KNN)	0.56	0.50	0.61	0.51	0.60	0.51	0.61
Logistic Regression	0.63	0.60	0.65	0.48	0.75	0.53	0.69
Support Vector Machine (SVM)	0.63	0.60	0.65	0.46	0.76	0.52	0.70
Decision Tree	0.61	0.56	0.66	0.56	0.65	0.56	0.65
Random Forest	0.63	0.60	0.64	0.42	0.78	0.50	0.70
Neural Network	0.62	0.61	0.63	0.45	0.77	0.52	0.69

The models demonstrated comparable overall accuracy, with most models achieving approximately 63% accuracy. K-Nearest Neighbors (KNN) exhibited the lowest accuracy at 56%. The Neural Network achieved an accuracy of 0.62. There is a noticeably higher recall for win predictions than losses. This indicates the models are better at predicting wins. Random Forest had the highest recall for wins at 0.78 and lowest recall for losses at 0.42.

Table 2. Model results when trained with Game and Player Dataset

Model	Accuracy	Precision (Loss)	Precision (Win)	Recall (Loss)	Recall (Win)	F1-score (Loss)	F1-score (Win)
Naïve Bayes	0.63	0.55	0.69	0.56	0.68	0.56	0.68
K-Nearest Neighbors (KNN)	0.53	0.43	0.60	0.42	0.61	0.42	0.61
Logistic Regression	0.64	0.59	0.67	0.44	0.78	0.50	0.72
Support Vector Machine (SVM)	0.64	0.57	0.67	0.47	0.75	0.51	0.71
Decision Tree	0.58	0.49	0.67	0.60	0.57	0.54	0.62
Random Forest	0.64	0.62	0.64	0.31	0.87	0.41	0.74
Neural Network (NN)	0.70	0.65	0.72	0.52	0.82	0.57	0.77

The Neural Network (NN) exhibited the highest accuracy at 0.70. Logistic Regression, Support Vector Machine (SVM), and Random Forest all achieved an accuracy of 0.64. Naïve Bayes had an accuracy of 0.63, while Decision Tree reached 0.58. K-Nearest Neighbors (KNN) demonstrated the lowest accuracy at 0.53. Similarly to the first dataset, the recall for win predictions was consistently higher than recall for loss predictions. Random Forest again had the largest disparity with a recall of 0.87 for predicting win and 0.31 for predicting loss.

Discussion

The models performed similarly for both datasets. KNN had the worst performance with 53-56% accuracy. For all other models, the only result outside of the range of 58-64% accuracy was the neural network, achieved over 70% accuracy for the game and player dataset.

This result led me to question the validity of the player and game dataset. The player data is aggregated over the entire season, which means that the model has access to data that is influenced by future events. The model may learn information about the strength of a team, even if their record prior to the game being predicted is poor.

Another potential issue is the size of the dataset. The game and player dataset only spans the 2017-18 season. With a total of only 991 games, the neural network may be overfitting the dataset, which would reflect poorly for games not in the dataset.

Additionally, I noticed that all the models exhibit a strong bias toward predicting wins, as recall for wins is typically higher than for losses. This suggests that the models have learned to predict home team victories a large majority of the time. In the NBA, the home team wins on average 61.45% of the time (López-García). Given that my best-performing model from the game dataset achieves 63% accuracy, it only improves upon this baseline by around 1.5%. This may suggest that the models use the home team as the primary statistic in their predictions, not relying on other factors very much.

Future Work

Data

A key task for future improvements is expanding and refining the dataset, which could improve model accuracy drastically. The quality of the dataset is one of the most important factors in creating a useful model. Increasing the dataset size to cover a broader range of seasons will help mitigate overfitting. Additionally, incorporating a broader set of statistics and advanced metrics into the dataset could help the models better understand some of the hidden complexities of NBA games. These could include player and team stereotyping, starting lineups, referee assignments, and contextual factors such as the stage of the season and recent trades. By making these changes to the dataset, the models may be able to find deeper trends and predict game outcomes with higher accuracy.

Neural Network

Further experimentation with neural network architectures could lead to improvements. Although many architectures with 5 or fewer layers were tried for this project, more experimentation could be done with deeper networks. Additionally, exploring alternative architectures such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks could help capture temporal dependencies within the game data. This could remove the need for the rolling metrics calculated in the game dataset, instead relying on the models to uncover these metrics naturally.

Conclusion

This project successfully trained and evaluated machine learning models for predicting NBA game outcomes, achieving reasonable accuracy for most of the models. While K-Nearest Neighbors (KNN) consistently underperformed, the remaining models demonstrated comparable predictive capabilities, suggesting that fundamental patterns within the data were effectively captured. Multiple models achieved around 63% accuracy, including Naïve Bayes, Logistic

Regression, SVM, and Random Forest. The Neural Network did not outperform the other models, implying that the current dataset might not fully leverage its capabilities.

There is significant potential for improved prediction accuracy with a more refined dataset. Concerns regarding the validity and size of the game and player dataset – specifically, the aggregation of seasonal player data and the limited scope of the 2017-18 season – raise potential issues of overfitting and information leakage. Addressing these concerns through dataset expansion, refined feature engineering, and a more rigorous validation process could improve model performance. Future work should prioritize these enhancements to develop more robust, accurate, and unbiased predictive models for NBA game outcomes.

Works Cited

Goldstein, Omri. "NBA Players Stats Dataset." *Kaggle*, https://www.kaggle.com/datasets/drgilermo/nba-players-stats?select=Seasons_Stats.csv. Accessed 24 Feb. 2025.

Houde, Matthew. "Predicting the Outcome of NBA Games." *Bryant University Digital Commons*, Bryant University, https://digitalcommons.bryant.edu/cgi/viewcontent.cgi?article=1000&context=honors_data_science. Accessed 24 Feb. 2025.

López-García, Adrián, et al. "Home-Court Advantage and Home Win Percentage in the NBA: An In-Depth Investigation by Conference and Team Ability." *Applied Sciences*, 1 Nov. 2024, https://www.mdpi.com/2076-3417/14/21/9989. Accessed 24 Feb. 2025.

Osken, Cem, and Ceylan Onay. "Predicting the Winning Team in Basketball: A Novel Approach." *PubMed Central (PMC)*, U.S. National Library of Medicine, https://pmc.ncbi.nlm.nih.gov/articles/PMC9764182/. Accessed 24 Feb. 2025.

Swar. *nba_api*. GitHub, https://github.com/swar/nba_api. Accessed 24 Feb. 2025.

Swar. "Home Team Win-Loss Modeling Data Prep." *nba_api*, GitHub, https://github.com/swar/nba_api/blob/master/docs/examples/Home%20Team%20Win-Loss%20Modeling/Home%20Team%20Win-Loss%20Data%20Prep.ipynb. Accessed 24 Feb. 2025.

Wang, Junwen. "Predictive Analysis of NBA Game Outcomes through Machine Learning." *Association for Computing Machinery (ACM)*, https://dl.acm.org/doi/pdf/10.1145/3635638.3635646. Accessed 24 Feb. 2025.

Weiner, Josh. "Predicting the Outcome of NBA Games with Machine Learning." *Towards Data Science*, https://towardsdatascience.com/predicting-the-outcome-of-nba-games-with-machine-learning-a810bb768f20/. Accessed 24 Feb. 2025.