Oracle Arena: Predicting the Winner and Total Score of NBA games using Machine Learning

Ryan Peruski
Department of EECS
University of Tennessee, Knoxville
yhg461@vols.utk.edu

Triton Eden
Department of EECS
University of Tennessee, Knoxville
teden@vols.utk.edu

***Abstract*—We use machine learning models trained on team-level NBA statistics to predict the winner and total score of games. We use deep learning for classification, and tree-based and linear models for regression, we achieve benchmark-level performance that may aid sports analytics and betting strategies.**

I. Introduction

Sports betting is a rapidly growing industry, with millions of fans placing wagers based on team performance, trends, and intuition. However, betting odds often reflect complex and uneven factors such as home-court advantage, recent form, and season-long trends. In this project, we leverage historical NBA team statistics to build machine learning models capable of predicting both the game winner and total combined score. By incorporating deep learning and regression techniques, our goal is to offer insights that could support more informed betting decisions and improve predictive accuracy in sports analytics.

II. Methodology

A. Overview

First, in order to start work on the ML techniques, we must obtain the data. We did so by using functions from nba_api like boxscore (for the current day) and boxscoreV2 (for previous days) [1]. We ended up getting data for players, teams, games, and stats tables from the 2018-19 season to the current season. Since we are using statistics that are constantly being changed, we are using a database rather than a static data file. This database contains the following ERD.

ERD Tables

Games

Attribute	Description
Game ID	Primary key
Season Year	Season of the game (e.g., 2024–25)
Date	Date of the game
Home Team ID	Foreign key to Teams
Away Team ID	Foreign key to Teams
Winner	Winner Prediction
Total Score	Total Score Prediction

Teams

Attribute	Description
Team ID, Season Year	Composite primary key
Team Name	Full name of the team
Team Abbr	Abbreviation of the team name
Team Location	City or location of the team

Players

Attribute	Description
Player ID	Primary key
Player First Name	First name of the player
Player Last Name	Last name of the player

Player Game Stats

Attribute	Description
Player ID, Game ID	Composite primary key
Team ID	Foreign key to Teams
Player Stats JSONB	JSON-formatted player stats

B. The Data Collection Process

Because the data requires regular updates, we use a PostgreSQL database that is automatically updated daily with fresh API data pulled at noon. We obtain the results by using Python libraries such as SQLAlchemy [2], Pandas [3], and the nba stats api [1].

III. Machine Learning Techniques

A. Feature Engineering

The features used in our models are derived from game-level statistics normalized per 100 possessions to account for pace differences between teams. The following core stats were selected for each team: FGM, FGA, FG3M, FG3A, FTM, FTA, OREB, DREB, AST, STL, BLK, TO, PTS, POSS, wins, and losses. For both the home and away teams, we computed:

- Season averages
- Rolling 5-game averages
- Opponent's season averages
- Opponent's 5-game averages

Prior to training, all feature values were scaled to the [0,1] range using a Min-Max scaler, ensuring that each feature contributes equally to the learning process. We

split our dataset into 80% training and 20% testing sets, maintaining chronological order to avoid data leakage.

B. Win Prediction Model

We used a deep feedforward neural network implemented with the Keras Sequential API. The architecture includes:

- 4 hidden layers with 256, 128, 64, and 32 neurons respectively
- ReLU activation functions
- Batch normalization after each layer
- Dropout layers with increasing rates (30% to 60%) to prevent overfitting
- A final sigmoid-activated output neuron for binary classification

The model was compiled using the Nadam optimizer (a variant of Adam with Nesterov momentum) and binary cross-entropy loss. Accuracy was used as the primary evaluation metric. For playoff games, a reduced dropout rate was used due to the limited number of games and the lower variance in performance.

C. Total Score Prediction Models

We approached total score prediction as a regression problem. After experimenting with several models, the following were selected:

- **Regular Season**: XGBoost, chosen for its strong performance after hyperparameter tuning
- **Playoffs**: Ridge Regression, selected due to its simplicity and generalization performance on smaller datasets

XGBoost helped capture non-linear relationships in the data, while ridge regression performed better under data constraints by reducing model variance.

IV. Results

A. Regular Season Win Prediction

The regular season model achieved an accuracy between 60% and 70%, which aligns with top-performing benchmarks in the sports analytics field. The model produced a confusion matrix that showed a balanced distribution of true positives and true negatives, with some over-predictions in high-variance games.

B. Playoff Win Prediction

Although the playoff dataset was significantly smaller, the playoff model outperformed the regular season model in terms of accuracy. This is likely due to the lower variance in playoff performance and fewer unexpected outcomes. The confusion matrix for the playoff model showed a tighter grouping around correct predictions.

C. Score Prediction Results

Scatter plots and histograms were generated to evaluate model performance for total score prediction. For the regular season, XGBoost performed well with tightly clustered scatter plots around the line of best fit and a narrow error distribution. During the playoffs, ridge regression's simplicity proved beneficial, producing smoother, less erratic predictions with lower overfitting risk.

Win Prediction			Total Score Prediction		
	Regular Season	Playoffs		Regular Season	Playoffs
	Deep Feedforward Neural Network	Deep Feedforward Neural Network		Model Tuned XGBoost Regression	Ridge Regression
Accuracy	0.66	0.67	RMSE	18.60	16.67
F1 Score	0.71	0.73	MSE	346.33	277.83
Loss	0.62	0.66	R^2 Score	0.11	0.08

Fig. 1. Table displaying the results of regular season and playoff win and total score prediction models.

V. Conclusion

Our experiments demonstrated the feasibility of using historical team-level NBA statistics to predict game outcomes and total scores. Deep learning models such as feedforward neural networks performed reliably for binary classification tasks like win prediction, especially when paired with batch normalization and dropout. For regression tasks, XGBoost delivered strong performance in high-variance environments, while ridge regression proved more robust in low-data, low-variance playoff scenarios.

These results suggest that properly engineered features and well-matched models can provide sports bettors and analysts with tools for higher-accuracy predictions. Future work could incorporate player-level data, in-game betting odds, and more granular temporal features to further boost prediction accuracy.

VI. Acknowledgment

Thank you to the University of Tennessee Department of EECS for providing the opportunity to learn and grow in the field of computer science, as well as our professor, Dr. Jack Marquez, and our TA, Gomathi Lakshmanan.

References

[1] National Basketball Association. (n.d.). NBA Stats API. Retrieved April 15, 2025, from https://www.nba.com/stats

[2] Bayer, M. (n.d.). SQLAlchemy [Computer software]. SQLAlchemy. Retrieved April 15, 2025, from https://www.sqlalchemy.org/

[3] The pandas development team. (2023). pandas [Computer software]. https://pandas.pydata.org/