Predicting NBA Shot Outcomes: A Machine Learning Approach for LeBron James

Furkan Kurt [1] **Mustafa Said Oğuztürk** [1]

Abstract

Basketball computations have become very data intensive and have moved away from heuristic models towards more robust statistical and machine learning techniques-being the current state of analytics in basketball. In this paper, we present a study on how to predict whether the shots made by LeBron James in the period of 2004-2024 will be successful or not by using a big dataset of over 29,000 attempts. This report aims to provide a thorough comparison of Logistic Regression, Random Forests, and XGBoost by showing that feature engineering, hyperparameter tuning, and model selection play a critical role in determining the effectiveness of predictive models. The highest percent of accuracy, 70%, is reached by using the XGBoost model so it is very clear that the nonlinear factors and the complicated processes of shooting mechanics are included in the model as the most significant factors. Apart from that, among other influential shot-related factors are shot distance, defender proximity, and game context, revealing LeBron's shooting tendencies as well. Using spatiotemporal details, we suggest that real-time analytics will actualize rapid decision-making in professional basketball. Finally, we sort out the track in the future on how to incorporate player tracking data and advanced deep learning models to meet the needs of predictive accuracy and contextual. Undoubtedly, along with further examining the course of action mentioned.

[1]Department of Artificial Intelligence Engineering, Hacettepe University, Turkey. Correspondence to: Furkan Kurt <furkan_kurt@hacettepe.edu.tr>, Mustafa Said Oğuztürk <saidoguzturk@hacettepe.edu.tr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

1. Introduction

Basketball is a fast paced sport in which the success of each shot may depend on numerous factors: the player's skill level, defensive pressure, distance from the basket, remaining time on the shot clock, and psychological influences such as fatigue or momentum. Over the past decade, *basketball analytics* has grown substantially, fueled by more robust data collection methods and increasingly sophisticated modeling approaches.

In this project, we focus specifically on **LeBron James**, an NBA icon recognized for his longevity and versatility. Rather than adopting a generalized, league-wide approach, we use a *player-specific* lens to examine shot success dynamics. Following the approach described in (1), we consolidated a dataset of roughly 29,311 shot logs spanning 2004–2024, including contextual elements such as shot distance, court zone, quarter, and time remaining.

1.1. Motivation and Goals

1. **Predictive Accuracy:** Evaluate how well advanced ML algorithms (Logistic Regression, Random Forests, XGBoost) can anticipate shot outcomes for a single elite player.

2. **Contextual Insights:** Derive actionable knowledge about the factors influencing LeBron James's shot success, including spatial zones, time context, and shot types.

We postulate that a boosted ensemble method (e.g. XGBoost) will be able to outperform simpler baselines due to the capability of capturing complex feature interactions, like the interrelation of shot distance, defensive pressure, and remaining time on the clock. Our project, through the lens of a single player like LeBron James, has the goal of not only augmenting predictive accuracy but also uncovering detailed nuances of his unique playing style. Ultimately, such an effort could be the first step to more player-specific analytics, thus closing the gap between a theoretical model and a practical solution in professional basketball.

2. Related Work

Sports analytics increasingly leverages modern machine learning to discover latent performance patterns, ranging from basic linear models to advanced neural networks. Prior works commonly rely on logistic regression or shallow architectures for shot prediction, focusing on core variables such as distance, player skill, and defensive coverage. For instance, (5) details how both distance and contesting defenders shape shot success probability, while (7) integrates ensemble methods that capture non-linear feature interactions and yield improved performance metrics.

Generalized shot prediction models exist for entire leagues (8; 2), providing broad insights but often overlooking the specialized tendencies of individual stars. Some recent approaches in soccer analytics similarly highlight how gradient boosting can excel at modeling single-player scoring patterns, underscoring the value of granular, player-centric data. Yet comparatively fewer studies isolate an athlete of LeBron James's caliber, whose long career and evolving play style present unique modeling challenges. Our study builds on these insights by applying ensemble-based frameworks to a large, LeBron-only dataset, thereby offering a deeper, context-rich view of one of the NBA's most influential players.

3. Dataset and Preprocessing

3.1. Data Collection and Overview

We consolidated shot logs from the 2004 to 2024 NBA seasons for LeBron James, merging multiple CSV files ($\sim$ 29,311 records) (1). These files, collected from both official and third-party sources, were combined to create a unified dataset enabling a more comprehensive analysis of LeBron's shot tendencies across various teams, roster contexts, and playing styles. Each record typically includes the following:

- **Game Context:** quarter, remaining time in the period, home vs. away indicators
- **Shot Characteristics:** numerical distance, court zone classification, action type (e.g. layup, dunk, jump shot), and final shot outcome (made or missed)
- **Player Metadata:** position, event type, and other advanced stats (where available)

This level of detail supports both traditional and advanced modeling approaches, capturing key contextual and spatial dimensions of each shot attempt.

3.2. Cleaning and Outlier Removal

Prior to modeling, the dataset underwent a two-step quality control process. First, rows with missing critical data (e.g., shot distance) were removed to ensure model features remained consistent across all entries. Second, any shot attempt exceeding 50 feet was considered an outlier (3) these instances often arise from last-second heaves or erroneous logs and thus discarded to maintain the overall integrity of the dataset. Approximately 2% of data failed these checks and was removed. While this may slightly reduce total data size, it helps maintain a clean and reliable feature space.

3.3. Feature Engineering

A well-designed feature set is crucial for capturing the subtle interactions influencing a shot's success or failure. In addition to basic distance or zone variables, we introduced several enriched features:

Time Left. We merged `MINS_LEFT`$\times 60+$`SECS_LEFT` into a single `TIME_LEFT` numeric feature, reflecting how deep into the quarter a shot occurs. This can be vital for modeling end-of-quarter pressure or in-game pacing shifts.

Buzzer-Beater Flag. Any shot launched in the final two seconds of a quarter was assigned a binary indicator. These attempts can differ dramatically in strategy and shot selection, often involving rushed or unconventional tactics.

Zone Success Rate. To highlight LeBron's "hot spots," we calculated a rolling or historical success rate for each major court zone. By capturing zone-level efficiency, we provide the model with contextual knowledge about which areas are likeliest to yield successful outcomes.

Encoding and Normalization. Categorical variables (zones, actions, shot type) were converted via one-hot or label encoding, depending on their cardinality and whether they contained ordinal relationships. Numeric attributes (distance, time) were scaled using the `MinMaxScaler` to ensure consistent magnitude across both linear (e.g. logistic regression) and tree-based (e.g. random forest, XGBoost) algorithms. Together, these transformations reduce noise and highlight meaningful patterns in the data.

4. Methodology

4.1. Models Explored

To model LeBron James's shot outcomes, we compared three supervised learning algorithms that differ in complexity, interpretability, and typical performance characteristics:

Logistic Regression (LR) A linear classification model that provides clear interpretability via its coefficients. While LR is widely used for its simplicity and ease of explanation, it may fail to capture complex non-linearities and feature interactions without extensive manual feature engineering (4). Nonetheless, LR often serves as a strong baseline in sports analytics applications.

Random Forest (RF) An ensemble of decision trees that can effectively handle non-linearities and reduce overfitting through bagging (bootstrap aggregation). By sampling both features and instances, Random Forests introduce randomness to improve generalization performance. However, proper hyperparameter tuning (e.g., `max_depth`, `n_estimators`, `min_samples_split`) is crucial to balance model complexity and variance.

XGBoost (XGB) A gradient boosting framework known for its strong performance on structured data. XGBoost incrementally builds trees to minimize a specified loss function, often outperforming simpler ensembles when adequately tuned (8). The ability to handle non-linear interactions between features makes it particularly appealing for basketball shot prediction, where contextual factors can interact in complex ways.

4.2. Implementation Details

All experiments were conducted in Python to ensure reproducibility and ease of integration with common data science workflows:

- **Scikit-learn** for Logistic Regression and Random Forest,
- **XgBoost** for gradient boosting,
- **GridSearchCV** with 5-fold cross-validation for hyperparameter optimization.

Following (7), we split the data into an 80–20 train–test ratio, stratifying by shot outcome (made or missed) to preserve class balance. Performance was primarily evaluated via accuracy, precision, recall, and F1-score. In addition, confusion matrices and feature importance plots (where applicable) were examined to gain deeper insights into each model's strengths and weaknesses, as well as to ensure that important contextual or spatial factors were indeed driving performance.

5. Experimental Results

5.1. Performance Metrics

Table 1: Performance Comparison of Models

Model	Accuracy	Precision	Recall	F1
Logistic Regression	66%	70%	58%	63%
Random Forest	65%	67%	59%	63%
XGBoost	**70%**	**72%**	**62%**	**66%**

As shown in Table 1, **XGBoost** delivers the highest accuracy (70%), underscoring its effectiveness at modeling non-linear feature relationships. Notably, XGBoost also leads in precision (72%) and F1-score (66%), reflecting its balanced ability to correctly identify successful shots. While logistic regression boasts a respectable precision (70%), it lags behind in both recall (58%) and overall accuracy (66%). Random Forest falls slightly below Logistic Regression in accuracy but maintains a similar F1-score (63%). Overall, these results highlight how *properly tuned* ensemble methods, particularly XGBoost, can surpass simpler baselines in capturing the complexities of structured sports data.

5.2. Visual Analysis

Figure 1 depicts the distribution of shot distances, confirming that most attempts cluster around 0–15 feet (close-range or mid-range). Figure 2 shows a heatmap of success rates by zone, highlighting strong performance in the restricted area.

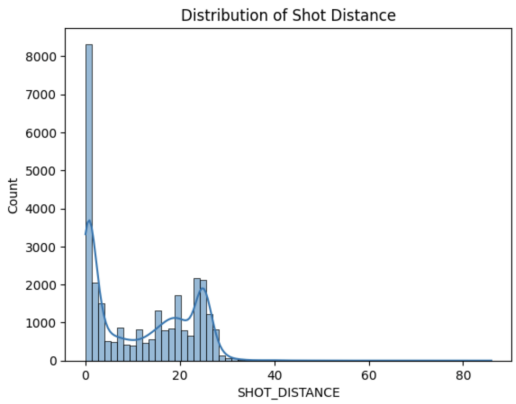

Figure 1: Distribution of Shot Distances in LeBron's Dataset.

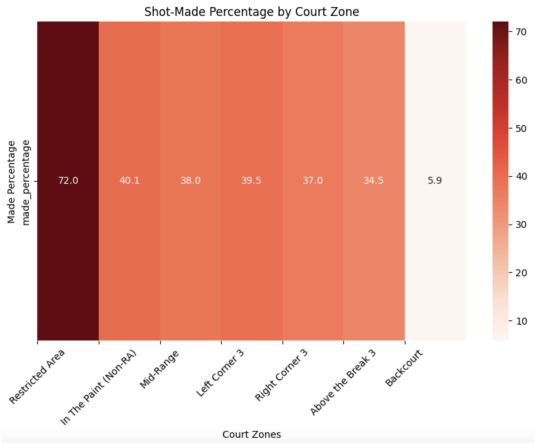

Figure 2: Heatmap of Shot Success by Court Zone. Darker cells indicate higher success rates.

5.3. Confusion Matrices

Below are the confusion matrices for XGBoost (Table 2) and Random Forest (Table 3), each evaluated with their respective best hyperparameter settings. Here, "Predicted P" refers to the model predicting a made shot, while "Predicted N" refers to a missed shot.

Table 2: Confusion Matrix for XGBoost – Best Parameters

	Predicted P	**Predicted N**
Actual P	2,313	582
Actual N	1,300	1,668
Total	3,613	2,250

Table 3: Confusion Matrix for Random Forest – Best Parameters

	Predicted P	**Predicted N**
Actual P	2,327	568
Actual N	1,352	1,616
Total	3,679	2,184

In Table 2, the XGBoost model accurately classifies 2,313 successful shots, while 582 are missed. The relatively low count of false positives (1,300) underscores XGBoost's strength in distinguishing made attempts. On the other hand, Table 3 shows the Random Forest's best-case scenario, where 2,327 shots are correctly identified as made; however, it has a slightly higher count of false positives (1,352). These matrices support the earlier findings that while both ensemble methods produce robust results, XGBoost maintains an edge in overall accuracy and balance between true positives and true negatives.

5.4. Logistic Regression Feature Importance

We also extracted logistic regression coefficients to gauge interpretability. As shown in Figure 3, `SHOT_DISTANCE` exerts the strongest negative influence on shot success, whereas restricted-area indicators hold the most positive impact. This aligns with standard basketball knowledge: close-range shots tend to be more efficient, while long-range attempts decrease the odds of a successful basket.

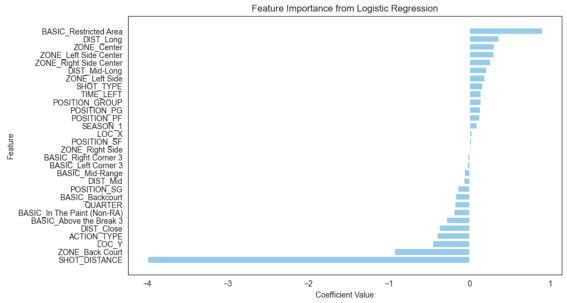

Figure 3: Feature Importance from Logistic Regression. Negative coefficients lower shot success likelihood, whereas positive coefficients raise the odds of a made shot.

6. Discussion

Our experiments confirm that XGBoost's ability to model non-linearities and exploit feature interactions yields the highest accuracy (70%), surpassing both Logistic Regression and Random Forest in multiple performance metrics. This finding underscores the importance of ensemble-based methods when dealing with structured yet context-rich sports data. In particular, **distance** consistently emerges as the single most predictive factor, aligning with established basketball principles that shorter-range shots result in higher conversion rates (4; 8). Nevertheless, zone-based success rates and buzzer-beater flags also prove significant, suggesting that even subtle spatial or temporal cues play a crucial role in influencing LeBron James's shooting outcomes.

Notably, XGBoost's advantage lies not just in raw accuracy but also in its robust handling of complex interactions among features like position, zone, and remaining time. In a professional sports setting, such nuanced modeling can offer coaches or analysts deeper insights into shot selection, helping to adapt offensive strategies around a player's evolving strengths.

6.1. Limitations

Data Shifts Over 20 Years While the dataset spans from 2004 to 2024, LeBron's playing style and physical attributes likely shifted over time. Static models may fail to capture these year-to-year (or even month-to-month) evo-

lutions. Future work may involve training time-segmented models or applying techniques like online learning to address distribution drift and thereby refine accuracy.

Defensive Context Our current pipeline lacks direct metrics on defender proximity, angle of contest, or other defensive pressures. Optical tracking data or synergy metrics could markedly enhance predictions, as heavily contested shots often demonstrate lower probabilities of success regardless of distance.

In-Game Feasibility This study focuses on retrospective logs, making real-time deployment a separate challenge. Live analytics would require robust data pipelines, near-instant inference, and integration into coaching workflows. These steps introduce additional engineering complexities such as latency management and on-court data validation, but they also promise significant benefits in strategic in-game adjustments.

7. Conclusions and Future Work

We demonstrated that machine learning models particularly XGBoost can effectively predict LeBron James's shot outcomes, reflecting the importance of robust, ensemble-based methods when dealing with diverse, player-specific data spanning multiple eras. By examining both shot-specific (distance, zone) and contextual (time left, buzzer-beater scenarios) features over two decades, we achieved approximately 70% accuracy, substantially improving upon simpler baselines. Throughout our analysis, *distance* consistently emerged as the strongest predictor, underscoring the well-known principle that shorter-range attempts typically yield higher success rates. However, the inclusion of zone-based success metrics and end-of-quarter pressure indicators reinforces the idea that even subtle factors can significantly influence shooting efficiency.

Beyond accuracy gains, these findings highlight the practical potential of integrated sports analytics pipelines. In particular, the insight that distance is a dominant determinant of success, combined with the effect of "hot zones" and final second heuristics, can inform targeted practice routines and situational play calling. Our results also suggest that a deeper, context-rich perspective focusing on one player's tendencies rather than league wide aggregates can unearth nuances that might otherwise be obscured.

Future Directions

1. **Spatiotemporal Modeling:** Future work could incorporate real-time player and defender trajectories, opening the door to dynamic metrics such as defensive pressure, shot angles, and relative velocities. Such data may more fully capture the fluid nature of in-game scenarios, shedding light on how contested shots or high-tempo situations affect success probabilities.
2. **Comparative Player Analysis:** Although LeBron James provides a compelling case study due to his longevity and versatility, extending the same methodology to other NBA stars (e.g., Stephen Curry, Giannis Antetokounmpo) would help identify both universal shooting behaviors and distinctly player-specific patterns. This cross-player lens could reveal whether certain "hot zone" phenomena generalize or remain unique to each athlete's skill set.
3. **Live Coaching Tools:** Translating a predictive framework from retrospective data to a real-time setting is an engineering and logistical challenge. Nonetheless, building a system capable of ingesting live shot logs and streaming defender-position updates could enable coaches to receive on-the-fly predictions about a shot's likelihood of success. Over time, such an automated assistant might guide substitution patterns, highlight defensive mismatches, and inform play calling in crucial ingame moments.

In essence, integrating advanced analytics with rich spatiotemporal data presents an avenue for more dynamic, high-precision player models. By continually refining the features, addressing data drift, and exploring new computational approaches, future research can further push the boundaries of individualized basketball analytics.

Acknowledgments

We gratefully acknowledge the support of Hacettepe University's Department of Artificial Intelligence Engineering throughout the development of this project. In particular, we extend our sincere thanks to the faculty and staff who provided guidance on both the theoretical and practical components of our research.

This work is primarily conducted by **Furkan Kurt** and **Mustafa Said Oğuztürk**, students in Hacettepe University's Artificial Intelligence Engineering program, whose combined efforts in data compilation, feature engineering, and experimental design formed the backbone of this study. Their dedication to open-source collaboration, including the integration of publicly available NBA shot data and the utilization of community-driven libraries like `scikit-learn`, `xgboost`,`pandas`, `numpy`, etc. greatly enhanced the scope and reproducibility of these findings.

Finally, we extend our gratitude to the broader open-source community for the creation and maintenance of valuable tools and datasets. The availability of high-quality li-

braries, notebooks, and reference implementations enabled a deeper exploration of player-specific analytics in professional basketball.

References

[1] NBA_Shots_04_24 GitHub Repository: github.com/DomSamangy/NBA_Shots_04_24

[2] CS229 Stanford Project (2017) cs229.stanford.edu/proj2017/final-reports/5132133.pdf

[3] Basketball Shot Predict, Medium Blog Post: medium.com/@fako5298/basketball-shot-predict-ad5ca0630d81

[4] From General to Specific: Shot Prediction for LeBron James, Medium Blog Post: medium.com/@fako5298/from-general-to-specific-building-a-shot-prediction-model-for-lebron-james-8c61e8ea2797

[5] Basketball Shot Analysis: From Data to Insights, Medium Blog Post: medium.com/@fako5298/basketball-shot-analysis-from-data-to-insights-5cda9b67284a

[6] First Modeling Attempts and Initial Results, Medium Blog Post: medium.com/@fako5298/first-modeling-attempts-and-initial-results-775299259c5b

[7] Feature Engineering and Model Improvements, Medium Blog Post: medium.com/@fako5298/5th-blog-post-feature-engineering-and-model-improvements-bd9847556e53

[8] Hyper-parameter Tuning, Ensemble Methods, Final Insights, Medium Blog Post: medium.com/@fako5298/6th-blog-post-hyperparameter-tuning-ensemble-methods-and-final-insights-

[9] Analysis of NBA Players and Shot Prediction Using Random Forest and XGBoost Models: https://ieeexplore.ieee.org/abstract/document/8716412

A. Selected Code Snippets

We present the parts that are most used and useful in the project.

Listing 1: 1. Loading and Merging Datasets

```
import pandas as pd
import os
import numpy as np

# Data Visualization
import seaborn as sns
import matplotlib.pyplot as plt

# Sklearn and XGBoost
from sklearn.preprocessing import
    OneHotEncoder, MinMaxScaler,
    LabelEncoder
from sklearn.model_selection import
    train_test_split, GridSearchCV
from sklearn.linear_model import
    LogisticRegression
from sklearn.metrics import (accuracy_score
    , precision_score,
recall_score, f1_score,
confusion_matrix, classification_report)
from xgboost import XGBClassifier

# 1. Merging CSV Files
project_path = r"yours.path\NBA_Shots_04_24
    "
file_list = [f for f in os.listdir(
    project_path)
             if f.endswith('.csv') and '
                 NBA_' in f]

dataframes = []
for file_name in sorted(file_list):
    file_path = os.path.join(project_path,
        file_name)
    df_temp = pd.read_csv(file_path)
    dataframes.append(df_temp)

merged_df = pd.concat(dataframes,
    ignore_index=True)
lebron_df = merged_df[merged_df['
    PLAYER_NAME'] == 'LeBron James'].copy()
lebron_df.dropna(inplace=True)
```

Listing 2: 2. Feature Engineering and Encoding

```
# Example Feature Engineering
lebron_df['TIME_LEFT'] = lebron_df['
    MINS_LEFT']*60 + lebron_df['SECS_LEFT']

# Buzzer-Beater Flag
lebron_df['BUZZER'] = lebron_df.apply(
    lambda row: 1 if (row['SECS_LEFT'] <= 2
        and row['MINS_LEFT'] == 0) else 0,
    axis=1)

# Outlier Removal
lebron_df = lebron_df[lebron_df['
    SHOT_DISTANCE'] < 50]

# Categorical Encoding Example
encoder = OneHotEncoder(sparse=False,
    handle_unknown='ignore')
zone_encoded = encoder.fit_transform(
    lebron_df[['BASIC_ZONE']])
# etc.
#
#
#
```

Listing 3: 3. Model Training and Evaluation

```
# Splitting Data
X = lebron_df.drop(columns=['SHOT_MADE'])
y = lebron_df['SHOT_MADE'].astype(int)

X_train, X_test, y_train, y_test =
    train_test_split(
    X, y, test_size=0.2, stratify=y,
        random_state=42
)

# Logistic Regression
log_model = LogisticRegression(max_iter
    =1000, random_state=42)
log_model.fit(X_train, y_train)
y_pred_log = log_model.predict(X_test)

print("Logistic Regression Accuracy:",
    accuracy_score(y_test, y_pred_log))

# XGBoost
xgb_model = XGBClassifier(eval_metric='
    logloss', random_state=42)
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)

print("XGBoost Accuracy:", accuracy_score(
    y_test, y_pred_xgb))

# Confusion Matrix
conf_matrix_xgb = confusion_matrix(y_test,
    y_pred_xgb)
print(conf_matrix_xgb)
```

Listing 4: 4. Hyperparameter Tuning (XGBoost)

```
param_grid = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0]
}

grid_xgb = GridSearchCV(
    estimator=XGBClassifier(eval_metric='
        logloss', random_state=42),
    param_grid=param_grid,
    scoring='accuracy',
    cv=5,
    n_jobs=-1
)
grid_xgb.fit(X_train, y_train)

print("Best Params:", grid_xgb.best_params_
    )
best_xgb = grid_xgb.best_estimator_
print("Accuracy:", accuracy_score(y_test,
    best_xgb.predict(X_test)))
```