Predicting NBA Game Outcomes Using 5-Game Rolling Player Statistics and Team Metrics

Kaden Hodgeman[1], Kyle Asmundson[1], and Kyle Jones[1]

[1]Arizona State University, Tempe, AZ 85281, USA

November 6, 2024

Abstract

Sports in the modern age have become highly data driven in nature, with an ever increasing focus on the collection of performance-based statistics that aim to quantify the effectiveness of athletes in their respective sporting realms. However, despite this concentration on data collection there are few existing methods that are able to consistently and accurately predict the outcomes of sports matches. Harnessing these statistics to successfully create a highly accurate game prediction model would have massive ramifications on advancing sports analytics and improving decision-making for professional sports organizations, not to mention the potential implications regarding the now-legal enterprise of sports gambling.

This study explores a machine learning-based approach for predicting NBA game outcomes using a combination of rolling statistics, individual player metrics, and advanced team metrics. By analyzing data from the 2013-14 through 2023-24 NBA seasons, we integrate both traditional and advanced metrics to explore the key factors influencing game outcomes. Our methodology includes logistic regression, XGBoost, neural networks, and a blended model, achieving a peak validation accuracy of 64.23% with XGBoost.

A key innovation in this study is the use of 5-game rolling averages for both team and player statistics, which smooths out the issue of game-to-game volatility and helps to harness recent performance trends for the purpose of improving model accuracy. Additionally, we apply K-Means clustering for the purpose of grouping players according to their respective archetype. These clusters are then incorporated into our prediction models to provide a more nuanced understanding of individual player contributions to game outcomes. Further analysis making use of SHAP (Shapley Additive Explanations) reveals that the win percentage differential, pace, and efficiency metrics are among the most important predictors of overall game outcomes.

Challenges remain in capturing intangible factors that impact game outcomes such as team chemistry and other in-game dynamics that are difficult to quantify with statistics alone. However, the models explored in this paper prove to be of use in the context of broader sports analytics. With proper tuning, there are clear and present potential applications for models using the techniques explored in this paper in the realms of real-time analytics and sports strategy.

1 Introduction and Background

1.1 Introduction

Basketball, like many sports, generates a wealth of data, from traditional box scores covering visible stats like points, rebounds, and assists, to more advanced metrics that have emerged in recent years. The sheer volume of this information presents a significant opportunity for deep analysis and predictive modeling. With the increased availability of detailed basketball statistics, the potential for applying data science techniques to better understand the game has grown substantially. In response to this opportunity, our project focuses

on predicting NBA game outcomes by leveraging a combination of individual player statistics, overall team performance metrics, and advanced statistics.

1.2 Background

Predicting game outcomes has been a subject of significant interest in sports analytics. Traditionally, statistical models such as logistic regression and Elo ratings were used to forecast game results. These models, while effective in capturing basic relationships, have struggled to accommodate the nuances and dynamic aspects of the game [3]. These limitations have motivated researchers to explore more sophisticated machine learning models capable of handling the complexities of basketball game data [2]. Recent efforts to incorporate ensemble learning methods, such as XGBoost, have demonstrated promising results due to their ability to handle high-dimensional and structured data [6].

The predictive modeling of basketball game outcomes is a growing area of interest in sports analytics. Early models, such as logistic regression and Elo ratings, provided some insight into game outcomes, but their predictive accuracy was often limited by the dynamic nature of the sport. More recently, machine learning models such as XGBoost and neural networks have been applied to NBA game data, offering a more nuanced analysis of player and team performance. These advanced models can handle high-dimensional and sparse data, improving the precision of predictions.

Additionally, the integration of advanced metrics such as effective field goal percentage (eFG%) and true shooting percentage (TS%) has contributed to more accurate models by providing a deeper understanding of player and team efficiency. However, challenges remain, particularly in capturing contextual factors like player injuries or in-game dynamics, which are not easily quantifiable. These challenges highlight the need for continued development in predictive modeling, particularly the incorporation of real-time data.

This project is guided by a need to integrate both traditional and advanced basketball metrics to provide a holistic view of game performance. Advanced metrics, such as those explored by Osken and Onay [5], have introduced additional insight into player performance, helping to predict game outcomes with greater accuracy. By applying machine learning models and focusing on the integration of these advanced metrics, our project aims to provide new insights into the critical factors that influence NBA game outcomes.

1.3 Methodologies in Literature

There exists a plethora of approaches for the prediction of game outcomes, the complexity of which has grown over the years. Different methods use different sets of input parameters and data refinement techniques to train their models. The diversity in data coming from different seasons of the NBA, NCAA, and Euro-league basketball further muddy the waters when trying to compare approaches. That being said, there are still noticeable trends and interesting approaches to note.

Ouyang et al. [6] used linear regression, and a range of ensemble models to predict NBA game outcomes using data from basketball-reference at different points during a game using a standard array of team box score statistics. Making predictions after the first two quarters, again after the third quarter, and finally after play has completed. Notable even after a full game model accuracy topped out at 93% with XGBoost leading in all instances in terms of accuracy and precision. Using SHAP to interpret their XGBoost models, they found interesting results in the change of significance in different features throughout different periods of play. For example, assists were an important indicator in the first half, but lost significance in the second. While offensive rebounds and 3-point shooting percentage were much more significant later in the game.

An introduction to the clustering of players can be seen in Lutz [4] cluster analysis where he finds success in clustering players by their box stats and shooting location choices. Laying the groundworks for moving beyond the traditional player labels. Finding correlation between clusters as well as groups of clusters have to game point differentials.

Osken and Onay [5] build on this idea of redefining traditional player positions with much closer focus on game predictions. Using a similar set of player box stats and shooting information for clustering. Then feeding the play time of each cluster and a number of other factors including rest time, winning percentage, etc. into an feed forward neural network that predicted winners. Seeing a 76.5% prediction rate in their test games. This approach is notable for not using the team level statistics normally used.

Cheng et al. [2] made use of a novel approach using a Maximum Entropy Model for predicting NBA playoff outcomes using data from the NBA stats API . The model was built around the standard set of team box score statistics using K-Means Clustering to discretize input variables to work with their model. Their resulting model performed well over a number of seasons compared to several other traditional models. They also show the high degree of variability between seasons, with season to season variance being as high as $\approx 10\%$.

1.4 Our Contribution

Our approach uses a novel combination of features and their preparation, by combining elements of team and individual stats in a rolling approach to a basketball game outcome predictor. Exploring how various factors contribute to the success or failure of teams, in specific match-ups. This approach allows us to not only predict game outcomes more effectively, but also offers insights into which player contributions or strategic decisions impact a team's overall performance. To address the variability in seasons seen in [2] we use a large set of training seasons to attempt minimize the effect of outliers.

Building on top of existing literature our approach aims to find a balance between studies focused more solely on team level stats as in [2] and using individual stats through clustering as in [5]. Hypothesising that the information gained from individual statistics will supplement the team level statistics providing information on the contributions of each player type. While resisting the loss of precision that is inherent in clustering.

2 Methodology

2.1 Overview

We use data from a large range of seasons to build a dataset that describes a diverse group of statistics at both the player and team level. The individual players are transformed into clusters based on their stats, and are aggregated by teams usage of each cluster. The cluster usage and each teams stats are then averaged over a 5 game window to predict the game outcomes.

2.2 Gathering Data

This project used a dataset containing team-level statistics across NBA games from the 2013-14 season to the 2023-24 season. It included both conventional statistics (such as points, rebounds, assists) and advanced metrics (such as offensive rating, effective field goal percentage, pace). Individual statistics included the conventional and advanced stats for each player, their shooting percentage broken down by contested and uncontested, as well as their shooting location. After importing the dataset from a CSV file, the data was thoroughly prepared and cleaned for modeling.

We sourced data primarily from the NBA API[7]. The NBA API offered extensive access to comprehensive basketball statistics. It included game logs, advanced statistics, and play-by-play events for each NBA game. We collected and processed relevant data using Python scripts. The data encompassed both team and player performance statistics from the 2013-14 season to the 2023-24 season.

To gather advanced statistics, the `BoxScoreAdvancedV2` endpoint of the NBA API was employed with the purpose of retrieving advanced team metrics from each individual game. The following is an example of how these advanced team statistics were obtained for each preseason game:

```
from nba_api.stats.endpoints import BoxScoreAdvancedV2

def fetch_preseason_adv_stats_by_game(game_ids):
    all_advanced_stats = []

    for game_id in game_ids:
        print(f"Fetching adv stats for {game_id}...")
        try:
            boxscore_adv = BoxScoreAdvancedV2(game_id=game_id)
            team_stats = boxscore_adv.get_data_frames()[1]
            all_advanced_stats.append(team_stats)
            time.sleep(1)
        except Exception as e:
            print(f"Error on {game_id}: {e}")
            continue

    if all_advanced_stats:
        df_adv_stats = pd.concat(all_advanced_stats, ignore_index=True)
        return df_adv_stats
    else:
        print("None")
        return pd.DataFrame()

game_ids_list = preseason_game_ids['GAME_ID'].unique()
df_preseason_adv_stats = fetch_preseason_adv_stats_by_game(game_ids_list)
```

Several relevant NBA API endpoints were used to collect data for building an accurate prediction model. The following section will list in further detail the function of each endpoint that is used in this project:

- `BoxScoreAdvancedV2` - Provided advanced game statistics, offering metrics beyond traditional box scores. These included: offensive and defensive ratings (points scored or allowed per 100 possessions), assist-to-turnover ratios (efficiency metrics for passing and ball handling), rebound percentages (percentage of available rebounds secured by players or teams), player impact estimate (evaluating a player's overall contribution to game outcomes), and true shooting percentage (adjusted for free throws and 3-point shots; evaluates scoring efficiency).

- `teamgamelogs` - Offered game logs for individual NBA teams across specified date ranges. These logs contained: points, rebounds, assists, steals, blocks, turnovers, field goal percentages, free throw percentages, and 3-point percentages. Additionally, each log included date and outcome data, such as the opponent and game result (win or loss). Statistics could be filtered by season, distinguishing between regular season and playoff data.

- `ShotChartDetail` - Supplied detailed spatial shot data for each game, including: shot location (exact spatial coordinates), shot type (such as layup, mid-range, 3-pointer), shot result (make or miss), and player-specific data (player attempting the shot, defender, and assists). Filter options included date range, team, player, and shot type.

- `BoxScorePlayerTrackingV3` - Dispensed detailed player tracking data, derived from optical tracking cameras. Metrics included: speed, distance, touches, time of possession, passing, defensive metrics, rebounding, and shot details (e.g., contested or uncontested shots). This endpoint enabled analysis of player performance beyond traditional statistics, providing a more nuanced view of player activity.

- `BoxScoreTraditionalV3` - Included traditional box score statistics essential for basic game analysis, such as points, rebounds, and assists, at both player and team levels.

The collected data was split into two subgroups for further analysis: full team performance data and individual player performance data.

2.2.1 Team Data Processing

The team data analysis aimed to calculate a rolling 5-game average for both standard and advanced statistics. This was achieved by merging the team game logs with the advanced statistics dataset, using 'GAME_ID' and 'TEAM_ID' as primary keys to align the data across sources.

Missing or zero values were carefully addressed to ensure data consistency. For example, discrepancies in rebounding statistics were resolved by applying established formulas for calculating offensive, defensive, and total rebounding percentages.

Before calculating rolling averages, redundant columns (e.g., rank columns) were removed, and the existing win/loss indicators were converted into binary values ("1" for wins and "0" for losses). Missing rebounding percentages ('OREB_PCT', 'DREB_PCT', 'REB_PCT') were imputed using the following formulas:

$$\text{OREB\_PCT} = \frac{\text{OREB}}{\text{OREB} + \text{OppDREB}}$$

$$\text{DREB\_PCT} = \frac{\text{DREB}}{\text{DREB} + \text{OppOREB}}$$

$$\text{REB\_PCT} = \frac{\text{REB}}{\text{REB} + \text{OppREB}}$$

The operations undertaken to accomplish these goals are displayed below:

```
# Merge advanced and game logs data
df_team_stats = team_game_stats.merge(team_game_adv_stats, on=['GAME_ID', 'TEAM_ID'], how='inner')

# Handle missing rebounding percentages using opponent statistics
df_team_stats['HOME_AWAY'] = df_team_stats['MATCHUP'].apply(lambda x: 1 if 'vs' in x else 0)
df_team_stats['OppOREB'] = df_team_stats.groupby('GAME_ID')['OREB'].shift(-1)
df_team_stats['OppDREB'] = df_team_stats.groupby('GAME_ID')['DREB'].shift(-1)
df_team_stats['OppREB'] = df_team_stats.groupby('GAME_ID')['REB'].shift(-1)
```

2.2.2 Rolling Average Calculation

Rolling 5-game averages were then computed for each team's statistics, covering both traditional and advanced metrics. A key consideration was to ensure that rolling averages did not span multiple seasons, especially for the first five games of each new season. Data from previous seasons was excluded to maintain temporal consistency. The rolling average calculations were performed as follows:

```
def apply_rolling_average_within_season(df, columns_to_roll, window=5):
    df = df.sort_values(by=['TEAM_ID', 'SEASON_YEAR', 'GAME_DATE'])
    for col in columns_to_roll:
        df[col + '_rolling'] = df.groupby(['TEAM_ID', 'SEASON_YEAR'])[col]\
            .rolling(window=window, min_periods=1)\
            .apply(lambda x: x[:-1].mean() if len(x) > 1 else np.nan)\
            .reset_index(level=[0,1], drop=True)
    return df
```

The approach pictured above allowed for the generation of rolling averages that accurately reflected recent team performance while maintaining the integrity of seasonality boundaries.

SEASON_YEAR	DREB_rolling	E_NET_RATING_rolling
TEAM_ID	REB_rolling	NET_RATING_rolling
GAME_ID	AST_rolling	AST_PCT_rolling
WL	TOV_rolling	AST_TOV_rolling
HOME_AWAY	STL_rolling	AST_RATIO_rolling
FGM_rolling	BLK_rolling	E_TM_TOV_PCT_rolling
FGA_rolling	BLKA_rolling	TM_TOV_PCT_rolling
FG_PCT_rolling	PF_rolling	EFG_PCT_rolling
FG3M_rolling	PFD_rolling	TS_PCT_rolling
FG3A_rolling	PTS_rolling	E_USG_PCT_rolling
FG3_PCT_rolling	PLUS_MINUS_rolling	E_PACE_rolling
FTM_rolling	E_OFF_RATING_rolling	PACE_rolling
FTA_rolling	OFF_RATING_rolling	PACE_PER40_rolling
FT_PCT_rolling	E_DEF_RATING_rolling	POSS_rolling
OREB_rolling	DEF_RATING_rolling	PIE_rolling

Table 1: Pictured above are the features of the fully cleaned versions of the NBA team game logs, including both classic and advanced stats, following the rolling statistics calculations

2.2.3 Player Data Processing

We obtained individual player data from four endpoints: Box Score Traditional, Box Score Advanced, Box Score Player Tracking, and Shot Chart Detail. The Shot Chart Detail data was updated seasonally and thus used for season-level clustering. We merged the remaining data using game and player IDs. Several columns contained missing values, including player position, comments, jersey number, and minutes played. We assigned a placeholder value of 'n' to non-starters to maintain a list of starters for each game. Comments, which were limited to instances where players did not play due to coach decisions, injuries, or suspensions, were replaced with 'none'. As no jersey numbers were recorded, the corresponding column was dropped. Players who did not participate in a game had null values for their minutes, which were replaced with zeros. This preprocessing yielded a dataset with 11 descriptive columns (such as game, team, and player details) and 45 player statistics.

Shot Chart Detail data provided details of each shot attempt, including the shooting player, game time, shot outcome (make or miss), and shot location as (x, y) coordinates and designated court zones. Zone-specific data was used to build the model, as outlined in Table 2. Shots were grouped by player, with shooting frequency and shooting percentage calculated for each zone. This information was combined with an aggregated version of the remaining box score data by player. The data was scaled using Sci-kit Learn's Robust Scaler before being processed through the k-means clustering model.

Shot Zone Basic	**Shot Zone Area**	**Shot Zone Range**
Restricted Area	Center (C)	Less Than 8 ft.
Mid-Range	Left Side Center (LC)	8-16 ft.
Above the Break 3	Right Side (R)	16-24 ft.
Right Corner 3	Right Side Center (RC)	24+ ft.
In The Paint (Non-RA)	Left Side (L)	Back Court Shot
Left Corner 3	Back Court (BC)	

Table 2: Shot Zones, Areas, and Ranges

2.2.4 Feature Engineering

We enhanced the dataset by calculating derived features. A key addition was a column representing each team's win percentage before each game. This win percentage acted as a cumulative measure of a team's prior performance during the season up to each game date, calculated using the formula:

$$\text{win\_percentage} = \frac{\text{wins\_to\_date}}{\text{total\_games\_played}}$$

We also calculated differences between the rolling averages of various game metrics (e.g., shooting percentages, turnovers, rebounds) for each team and their opponent. These "differential" features (such as `diff_FG_PCT_rolling`) compared the competing teams, serving as indicators of relative strength before the game. The differential columns were computed using:

$$\text{diff\_stat} = \text{team\_stat} - \text{opponent\_stat}$$

For each game, the difference between a team's rolling average and its opponent's was calculated for all rolling statistics, creating a dataset that captured how much stronger or weaker a team was relative to its opponent based on recent performance.

Table 3 below shows the features for the dataset used to run our models.

SEASON_YEAR	TEAM_ID	GAME_ID
WL	HOME_AWAY	win_percentage
opp_win_percentage	diff_FGM_rolling	diff_FGA_rolling
diff_FG_PCT_rolling	diff_FG3M_rolling	diff_FG3A_rolling
diff_FG3_PCT_rolling	diff_FTM_rolling	diff_FTA_rolling
diff_FT_PCT_rolling	diff_OREB_rolling	diff_DREB_rolling
diff_REB_rolling	diff_AST_rolling	diff_TOV_rolling
diff_STL_rolling	diff_BLK_rolling	diff_BLKA_rolling
diff_PF_rolling	diff_PFD_rolling	diff_PTS_rolling
diff_PLUS_MINUS_rolling	diff_OFF_RATING_rolling	diff_DEF_RATING_rolling
diff_NET_RATING_rolling	diff_AST_PCT_rolling	diff_AST_TOV_rolling
diff_AST_RATIO_rolling	diff_TM_TOV_PCT_rolling	diff_EFG_PCT_rolling
diff_TS_PCT_rolling	diff_PACE_rolling	diff_PACE_PER40_rolling
diff_POSS_rolling	diff_PIE_rolling	diff_cluster_0
diff_cluster_1	diff_cluster_2	diff_cluster_3
diff_cluster_4	diff_cluster_5	diff_cluster_6
diff_cluster_7	diff_cluster_8	diff_cluster_9
diff_cluster_10		

Table 3: Final Features for Modeling

2.2.5 Data Splitting and Scaling

The dataset was split into training, validation, and test sets for model building and evaluation. The test set included data from the 2022-23 and 2023-24 seasons, reserved for final evaluation, while the remaining seasons (2014-15 to 2021-22) were used for training and validation.

The splits were as follows:

- The test set included all games from the 2022-23 and 2023-24 seasons, used for evaluating model performance on recent data.

- The remaining seasons served as validation sets for Logistic Regression and XGBoost during hyperparameter tuning.
- An 80:20 split was applied to these seasons to create training and validation sets for the Neural Network.

To ensure comparability across features with different scales, the data were normalized using a `Robust -Scaler`. This method was chosen for its resistance to outliers, minimizing the influence of extreme values. All numerical features were scaled, while the target variable (`WL`, indicating whether a team won or lost) remained unchanged.

2.3 Machine Learning Methodology

The primary goal of the modeling process was to predict the outcome of NBA games based on the available rolling statistics and differential features. The modeling pipeline began with logistic regression and evolved to incorporate more complex models such as XGBoost and a neural networks. Each of these models were tuned to optimize performance on the validation set.

2.3.1 Unsupervised Machine Learning (K-Means Clustering)

Individual player statistics present a unique challenge due to the sheer volume and variability in player performance. To address these challenges, a clustering approach was used, grouping players together based on the features from the combined box score statistics, their shooting profile built from the shot chart data and their effectiveness in different plays using the percentile rank for each play type in the play type data all of which can be seen in table 4. In order to help remove the overlap between the large number of individual features we used principle component analysis for dimensional reduction on the input features 4 aiming for 90% of variance explained between the players, yielded 29 components to use for clustering.

offensive Rating	defensive Rating	assist %	assist To Turnover
assist Ratio	rebound %	effective Field Goal %	usage %
pace Per 40	speed	distance	rebound Chances Offensive
rebound Chances Defensive	rebound Chances Total	touches	secondary Assists
free Throw Assists	passes	assists	contested Field Goals Made
contested Field Goals Attempted	contested Field Goal %	uncontested Field Goals Made	uncontested Field Goals Attempted
uncontested Field Goals %	field Goal %	defended At Rim Field Goals Made	defended At Rim Field Goals Attempted
defended At Rim Field Goal %	field Goals Made	field Goals Attempted	field Goals %
three Pointers Made	three Pointers Attempted	three Pointers %	free Throws Made
free Throws Attempted	free Throws %	rebounds Offensive	rebounds Defensive
rebounds Total	steals	blocks	turnovers
fouls Personal	points	plus Minus Points	avg minutes
games started	making % Above the Break 3	making % In The Paint (Non-RA)	making % Left Corner 3
making % Mid-Range	making % Restricted Area	making % Right Corner 3	usage % Above the Break 3
usage % In The Paint (Non-RA)	usage % Left Corner 3	usage % Mid-Range	usage % Restricted Area
usage % Right Corner 3	making % Backcourt	usage % Backcourt	Defensive Handoff
Defensive Isolation	Defensive Off Screen	Defensive PR Roll Man	Defensive Postup
Defensive Spotup	Offensive Cut	Offensive Handoff	Offensive Isolation
Offensive Misc	Offensive Off Rebound	Offensive Off Screen	Offensive PR Ball Handler
Offensive PR Roll Man	Offensive Postup	Offensive Spotup	Offensive Transition

Table 4: Variable Names for Player Metrics

The number of clusters used was informed using the average silhouette score at different cluster counts for models trained for each season. This approach helped find a cluster count that balanced performance across all seasons. The output of which is shown in figure 1. Deciding on a value for cluster count was a balance between a low number of clusters that provided the best distinction between players but regressed to identifying starters, bench players and third string backup players and a cluster count with no distinction between players to inform a model. After testing a selection of cluster counts predictive ability, we found a count of 10 to be a good balance between these trade offs. Once we had a value for the number of clusters to use the model was formed using data from our training seasons.

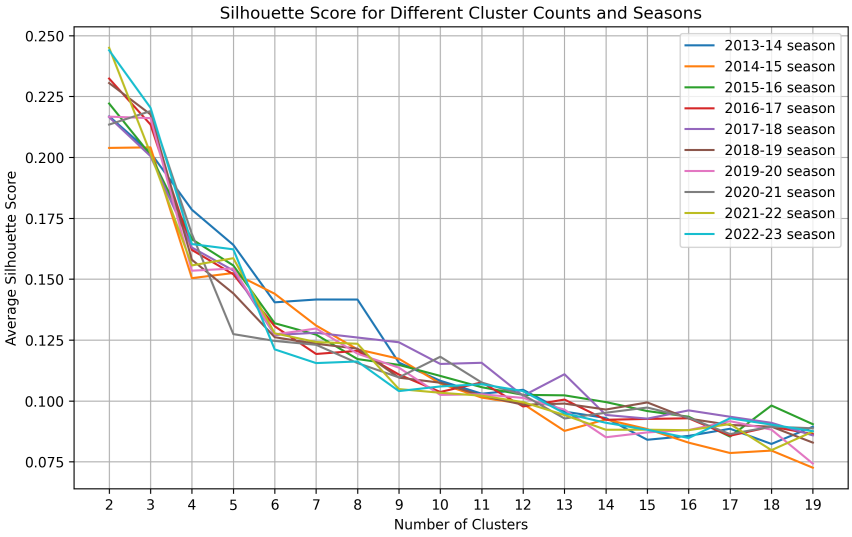

Figure 1: Plot of the silhouette score at different cluster counts for models fit to each season in the test data

To make the clustering information useful in the context of a game predictor the usage percentage of players in the rolling game window is grouped by cluster ID and then summed to create a usage percentage for each cluster. This can then be used along side the rest of the team data. To address the the concern of players performances latter in a season effecting their cluster value earlier in a season cluster values were determined on the previous seasons data and new players to the league were added to their only cluster for there inaugural year (cluster 10). This approach had the downfall of not adapting to players development between and during seasons which could be addressed in the future by updating the frequency of cluster updates at the expense of computation time.

2.3.2 Logistic Regression

Logistic Regression was selected as a baseline model for binary classification in this analysis. This model was trained on a scaled dataset using player and team statistics. The key advantage of Logistic Regression lies in its simplicity and interpretability. By modeling the probability that a team will win or lose as a function of various input features, Logistic Regression provides a straightforward benchmark for comparison with more advanced techniques. In this context, Logistic Regression assumed a linear relationship between the predictor variables and the log-odds of the outcome. To reduce overfitting, regularization methods were applied by adjusting the penalty term and testing various solvers.

2.3.3 XGBoost Design

XGBoost was utilized as a gradient-boosting algorithm, known for its capacity to handle structured data with numerous features effectively. To enhance the model's predictive performance, a comprehensive grid search was conducted, testing combinations of hyperparameters such as max depth, learning rate, number of estimators, subsample ratio, and column sampling ratio. The search examined tree depths of 3, 4, 5, and 7, learning rates ranging from 0.009 to 0.2, and boosting rounds between 100 and 800. The optimal configuration, which minimized overfitting and captured intricate patterns in the dataset, is presented in Table 5. This setup was subsequently used to train the final model, leveraging early stopping to avoid excessive iterations.

Parameter	Value
colsample_bytree	0.7
learning_rate	0.009
max_depth	4
n_estimators	500
subsample	0.7

Table 5: Best Parameters for XGBoost model

Post-training, we applied SHAP (SHapley Additive exPlanations) to interpret the contribution of individual features to the model's predictions. SHAP analysis helps identify how different features impact the prediction probability, offering transparency in model behavior. By calculating SHAP values for each feature in the validation set, we gained insight into which factors were most influential in determining game outcomes. This analysis highlighted the importance of metrics such as rolling averages of shooting percentages, defensive ratings, and home-court advantage.

To ensure clarity, the SHAP analysis excluded less interpretable features like cluster-based statistics, focusing instead on conventional team and game-level metrics. The summary plot (see Figure 3) visually represents the influence of each feature, showcasing how they contributed positively or negatively to the model's predictions. The combination of XGBoost's gradient-boosting capabilities with SHAP analysis not only provided high predictive accuracy but also improved our understanding of key factors driving the model's decisions.

2.3.4 Neural Network Design

The Neural Network model design evolved through multiple stages. Initially, the network struggled with overfitting and variability in test performance. To counteract these challenges, adjustments were made to the learning rate, dropout rates, batch size, and the number of epochs. Schedulers such as StepLR and ReduceLROnPlateau were tested to manage the learning rate dynamically with the final model using reduce on plateau for its smarter approach to slowing. The final neural network architecture consisted of six fully connected layers, with LeakyReLU activation's using the default negative slope. The Tree-structured Parzen Estimator approach[1] was utilized for hyperparameter optimization, as it efficiently searched the large parameter space over grid search methods. The model was then fine-tuned based on those results, refining both dropout rates and layer sizes. This process ensured a model that could handle the high dimensionality and complexity of the dataset.

Hyperparameter	Value
Batch Size	256
Dropout Rate	6.685×10^{-1}
Epochs	20
Factor	4.043×10^{-1}
Layer Sizes	[[640, 320, 160, 80, 64]
Learning Rate	6.887×10^{-3}
Patience	8
Weight Decay	6.749×10^{-3}

Table 6: The hyperparameters used in the final neural network

2.3.5 Blended Model

The blending technique employed in this analysis used soft voting, where the predicted probabilities from the Logistic Regression, XGBoost, and Neural Network models were averaged to produce a final prediction. By averaging probabilities rather than discrete predictions (hard voting), soft voting enabled the blend to capture

different aspects of the data learned by each model. Each of the three models contributed equally to the final prediction, as no explicit weights were assigned during the combination process. This blended approach aimed to leverage the strengths of all three models: Logistic Regression for its interpretability, XGBoost for its ability to handle structured data, and the Neural Network for capturing complex relationships. The blended model was particularly useful for balancing the predictions of wins and losses, as it mitigated the weaknesses of any individual model, ultimately improving the robustness of the predictions.

3 Results

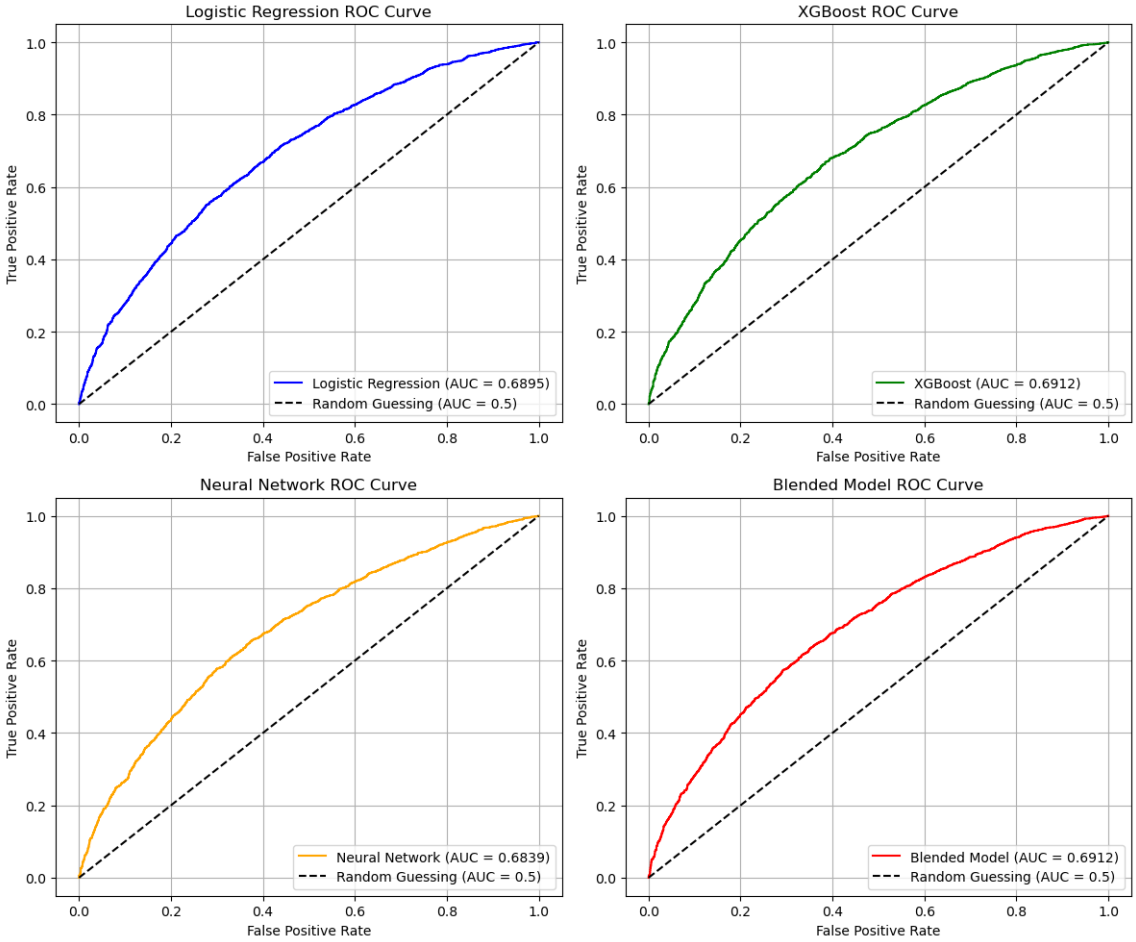

Figure 2: The ROC curves for the different models

The predictive models developed - Logistic Regression, XGBoost, Neural Network, and a blended model - exhibited varying degrees of success in predicting NBA game outcomes using both team and individual player statistics. The Logistic Regression model achieved a validation accuracy of 63.40%, providing a strong baseline. The XGBoost model outperformed Logistic Regression with a validation accuracy of 64.23%, indicating its superior ability to capture complex relationships within the data. The Neural Network, after extensive hyperparameter tuning, performed comparably but did not significantly surpass the other models. In contrast, the blended model, which combined the predictions of all three, yielded a validation accuracy of 61.08%. While the blended model did not achieve the highest accuracy, it offered a more balanced approach

by integrating insights from multiple models, highlighting the benefits and limitations of model ensembling in this context.

Exploring team performance revealed that metrics such as effective field goal percentage (eFG%) and net rating were significant contributors to the model's ability to distinguish between wins and losses. Furthermore, integrating rolling game statistics enhanced the overall predictive accuracy.

At the individual level, clustering players based on advanced statistics yielded valuable insights. Usage percentage by cluster was integrated into the models, with players grouped into archetypal clusters that contributed to their team's overall performance in distinct ways. This approach allowed for a more nuanced understanding of player contributions beyond traditional box scores.

3.1 Exploring Team Performance

The SHAP analysis provided valuable insights into how differentials in team performance metrics, calculated over a 5-game rolling average, contribute to predicting NBA game outcomes. These features represent differences between a team's and their opponent's performance over the most recent five games, capturing recent momentum and comparative advantages or disadvantages against recent opponents. Instead of raw performance metrics, these differentials provide context-sensitive insights into how teams perform relative to their competition.

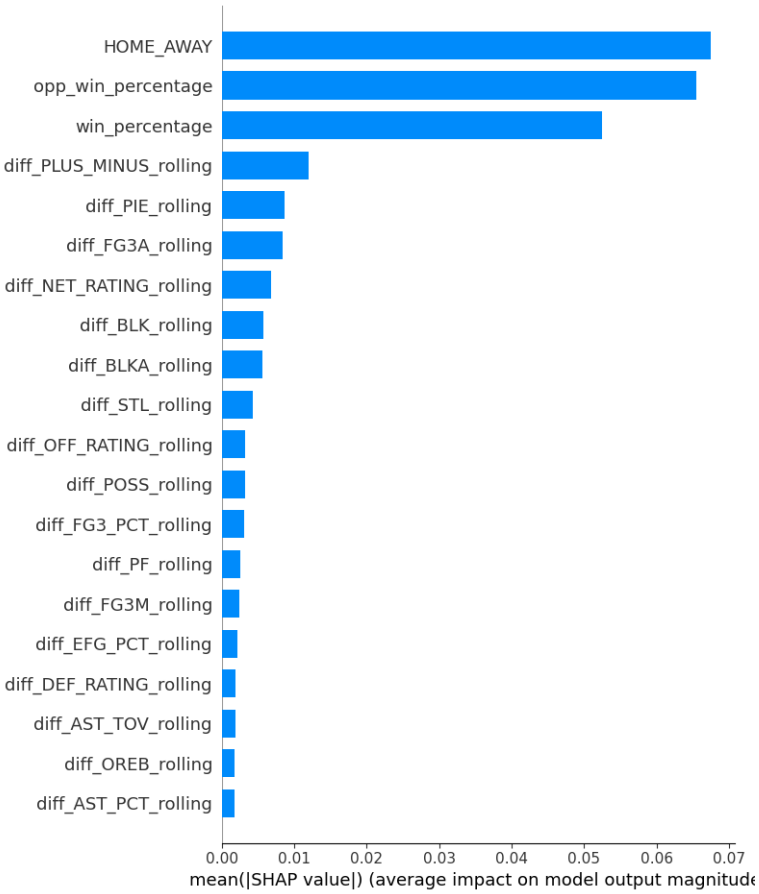

Figure 3: SHAP Bar Plot for Team Performance Features

3.1.1 Team SHAP Values

- **Home/Away Status**: The HOME_AWAY feature holds the highest absolute SHAP value, indicating that playing at home or away significantly impacts game outcomes. A negative SHAP value suggests that being the home team tends to increase the likelihood of a win when all other factors are held constant, potentially reflecting the adjustments teams must make when playing in their opponents' home venue.
- **Win Percentage**: The differential in win percentage (win percentage and opponent win percentage) retains importance in determining game outcomes. A team's recent performance compared to its opponent remains a crucial factor, with a higher win percentage positively influencing predictions. However, the negative SHAP value for the opponent's win percentage emphasizes the advantage a team holds over an opponent with a lower recent winning rate.
- **Scoring and Plus-Minus Differentials**: The differential in plus-minus over recent games shows a notable positive SHAP value, indicating that teams with a better average point differential over the last five games are more likely to secure wins. Additionally, other scoring metrics, such as points differential, suggest that teams outscoring their opponents have an increased chance of success, though their impact is more modest.
- **Efficiency and Turnover Metrics**: The assist-to-turnover ratio and assist percentage features indicate that ball movement efficiency remains a relevant predictor, with better ratios contributing positively to game outcomes. Conversely, turnover percentage reflects the negative impact of giving up possession, although it is less critical than other metrics.
- **Pace and Game Control**: Although less prominent, the pace feature still suggests that maintaining a higher tempo can be advantageous. A positive SHAP value for pace indicates that teams with a faster playing speed over the last five games are more likely to win, potentially reflecting their control over the game flow.
- **Defense and Rebounding**: While features like defensive rating and defensive rebounds have positive SHAP values, their impact is relatively minor compared to offensive metrics. This suggests that while defensive play is important, it does not drive game outcomes as strongly as offensive performance or recent winning trends.
- **Shooting Metrics**: Differentials in effective field goal percentage and true shooting percentage appear with small negative SHAP values, suggesting a limited but still notable influence. These metrics, while significant, did not dominate the prediction space as much as win percentage or home/away factors.

3.1.2 Implications of Team Rolling Differentials

These rolling differentials provide a dynamic, context-sensitive view of a team's recent form, emphasizing trends in performance rather than focusing on one-off anomalies. By smoothing out performance variability over the last five games, we capture a more accurate picture of a team's momentum A team that has outpaced opponents or displayed superior shooting efficiency over recent games is likely experiencing a winning streak, and these rolling metrics help highlight those trends.

The SHAP analysis of these rolling differentials aligns well with expectations: teams that outperform their recent opponents in key areas, like pace, shooting efficiency, defensive strength, and ball control, are more likely to win.

3.2 Exploring Individual Performance

We used the clustering results to explore the composition of different teams across seasons and the league. The different combinations of players used by teams for the 2023-24 season, as shown in Figure 4, demonstrated

the range of compositions that exist. For example, the Bulls and Knicks used few or no rookies compared to the Spurs and Pistons, whose rookies saw a substantial amount of playtime. The skills represented in each cluster were identified by analyzing the top-performing clusters across various player metrics (see Table 7).

Clusters 7 and 3 stood out, dominating the majority of metrics and including many all-star players, while clusters 0, 1, 2, 4, and 10 were not represented among the top metrics. Among the prominent clusters, cluster 3 included many dynamic bigs, such as Dwight Howard and Nikola Jokić. Cluster 7 featured elite shooters like Stephen Curry and Kobe Bryant. Cluster 9 contained playmakers and strong defensive players, cluster 8 had many skilled shooters, and cluster 5 included traditional guards and shooters like Steve Nash.

Table 7: standout clusters for individual player metrics

Cluster	**Statistics**
9	defensive Rating, assist Percentage, assist To Turnover, assist Ratio
8	effective Field Goal Percentage, field Goal Percentage, making Percentage Restricted Area, Offensive PR Roll Man, Offensive Transition
7	offensive Rating, usage Percentage, touches, secondary Assists, free Throw Assists, passes, assists, uncontested Field Goals Made, uncontested Field Goals Attempted, three Pointers Made, three Pointers Attempted, free Throws Made, free Throws Attempted, free Throws Percentage, steals, turnovers, points, plus Minus Points, avg Minutes, games Started, making Percentage Left Corner 3, making Percentage Mid-Range, making Percentage Right Corner 3, usage Percentage Mid-Range, usage Percentage Restricted Area, Offensive Cut, Offensive Handoff, Offensive Isolation, Offensive Misc, Offensive Off Screen, Offensive PR Ball Handler, Offensive Spotup
6	speed, making Percentage Above the Break 3, making Percentage Backcourt, Defensive Handoff
5	distance, usage Percentage Backcourt
3	rebound Percentage, rebound Chances Offensive, rebound Chances Defensive, rebound Chances Total, contested Field Goals Made, contested Field Goals Attempted, contested Field Goal Percentage, uncontested Field Goals Percentage, defended At Rim Field Goals Made, defended At Rim Field Goals Attempted, defended At Rim Field Goal Percentage, rebounds Offensive, rebounds Defensive, rebounds Total, blocks, fouls Personal, usage Percentage Above the Break 3, usage Percentage In The Paint (Non-RA), usage Percentage Left Corner 3, usage Percentage Right Corner 3, Defensive Isolation, Defensive Off Screen, Defensive PR Roll Man, Defensive Postup, Defensive Spotup, Offensive Off Rebound

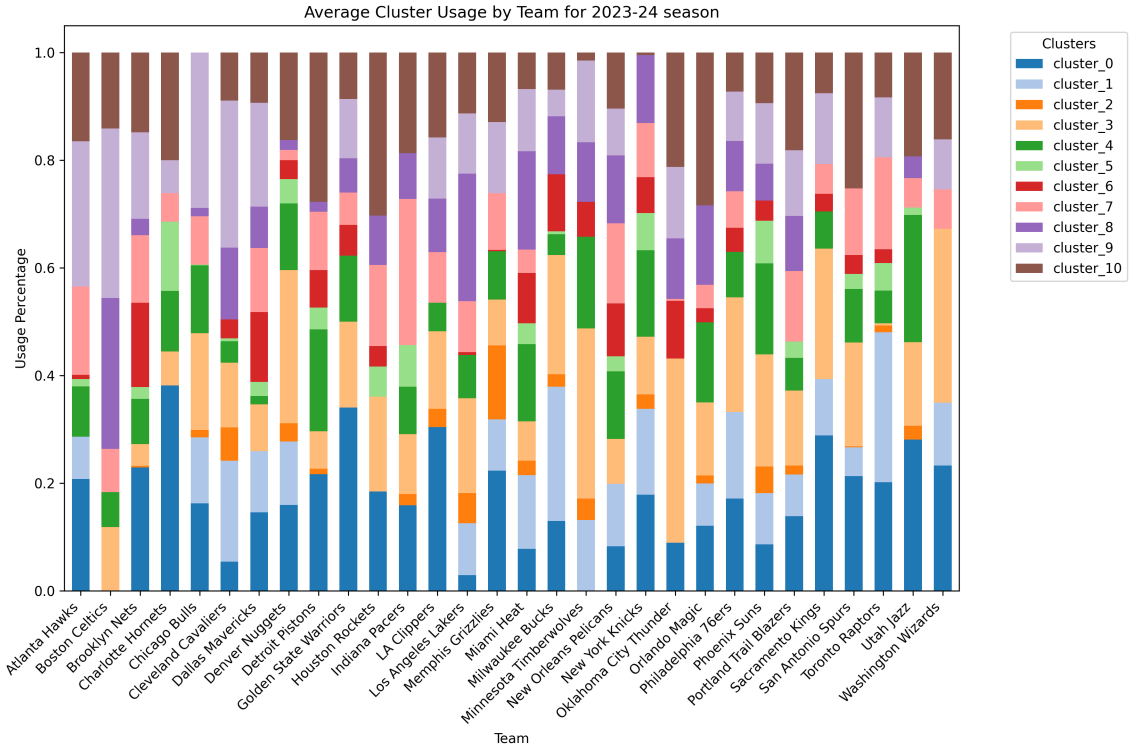

Figure 4: The different player types used by teams for the 2023-24 season.

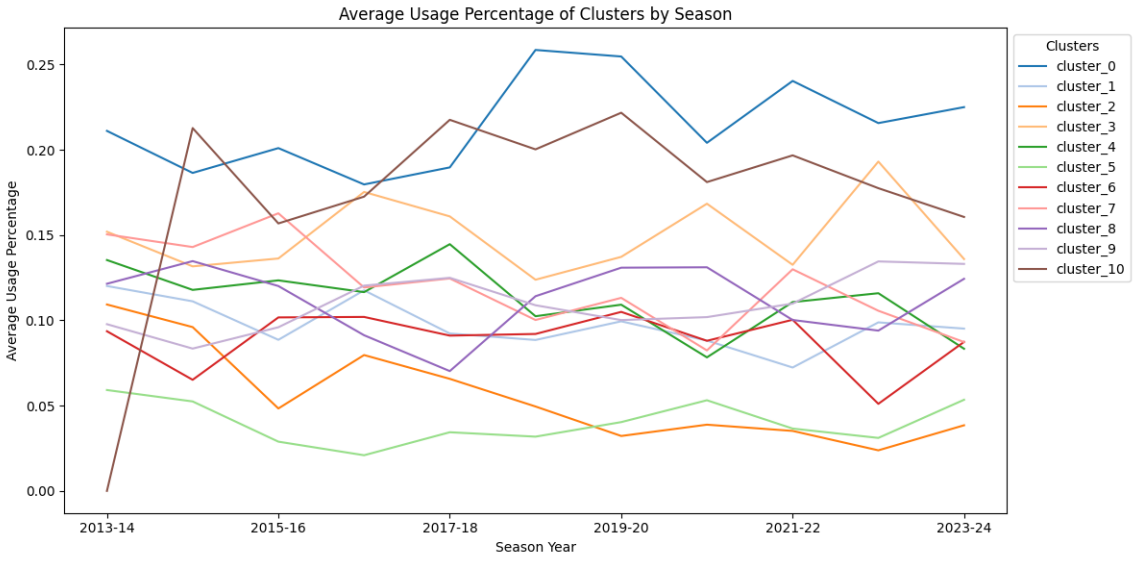

Figure 5: The Evolution of player type usage through the seasons, note the lack of cluster 10 values for the 2013-14 season is due to there not being any new players for that year.

In figure 5 there are a couple features of note, the players in cluster 0 are not included in our list of

standout clusters for any metric but receive the most use across any player type. Players are a mix of guards and forwards that play supportive rolls for teams with consistent if not stand out performances. The usage of the superstar shooters of cluster 7 falls through the seasons in comparison to the other clusters. Rookies and transitioning players also contribute significantly to teams as their rolls are cemented. The product of player clustering shows some of the differences between teams in the league, how the game has evolved and help define key contributors to a teams overall performance.

3.3 Merged Individual and Team Rolling Statistics

The integration of both rolling individual player metrics and team statistics offers a more comprehensive approach to predicting NBA game outcomes. By leveraging features that capture player-specific performance clusters and team-level statistics, we build a model that accounts for the interdependencies between players' contributions and overall team dynamics. The rolling average statistics over a 5-game window provide an effective means to smooth out game-to-game volatility and reveal underlying performance trends.

3.3.1 Logistic Regression

The Logistic Regression model serves as a baseline for our analysis, utilizing the merged dataset of individual and team rolling statistics. The model achieved a validation accuracy of 63.40%, which suggests a balanced, though limited, ability to predict game outcomes. Table 8 displays the confusion matrix, while Table 9 summarizes the precision, recall, and F1-scores for each class (win/loss).

Table 8: Confusion Matrix for Logistic Regression Model

	Predicted Win	**Predicted Loss**
Actual Win	1528	882
Actual Loss	882	1528

The confusion matrix illustrates that the model's predictions are relatively balanced, with similar counts for true positives and true negatives . However, the false positives and false negatives indicate some limitations in the model's discriminative ability.

Table 9: Classification Report for Logistic Regression Model

Class	**Precision**	**Recall**	**F1-score**	**Support**
Win	0.63	0.63	0.63	2410
Loss	0.63	0.63	0.63	2410
Accuracy	0.63 (4820 samples)			
Macro Avg	0.63	0.63	0.63	4820
Weighted Avg	0.63	0.63	0.63	4820

The classification report in Table 9 provides further insights into the model's balanced performance across win and loss predictions. Both precision and recall hover around 0.63 for each class, indicating that the model is equally likely to identify true positives and negatives. The F1-score, which balances precision and recall, remains consistent at 0.63 for both classes, underscoring the uniform nature of the model's performance across different outcomes. Despite its simplicity, the Logistic Regression model establishes a benchmark for comparing more complex models like XGBoost and neural networks.

3.3.2 XGBoost

The XGBoost model, after optimization through grid search, demonstrated a slight improvement over the Logistic Regression model, with a validation accuracy of 64.23%. Table 5 lists the best parameters from

the grid search process. The confusion matrix (Table 10) and classification report (Table 11) highlight the stronger performance of XGBoost, particularly in capturing more complex relationships within the data.

Table 10: Confusion Matrix for XGBoost Model

	Predicted Win	**Predicted Loss**
Actual Win	1545	865
Actual Loss	859	1551

The confusion matrix indicates a balanced number of correct predictions for wins and losses, with a slight edge in identifying true wins. This suggests that XGBoost effectively captures subtle interactions between team-level and individual performance metrics that Logistic Regression may miss.

Table 11: Classification Report for XGBoost Model

Class	**Precision**	**Recall**	**F1-score**	**Support**
Win	0.64	0.64	0.64	2410
Loss	0.64	0.64	0.64	2410
Accuracy	0.64 (4820 samples)			
Macro Avg	0.64	0.64	0.64	4820
Weighted Avg	0.64	0.64	0.64	4820

The classification report shows that XGBoost achieves consistent precision and recall values across both win and loss classes. This balanced performance suggests that the model can effectively differentiate between competitive games and more predictable outcomes, making it a valuable tool for game predictions.

3.3.3 Neural Network

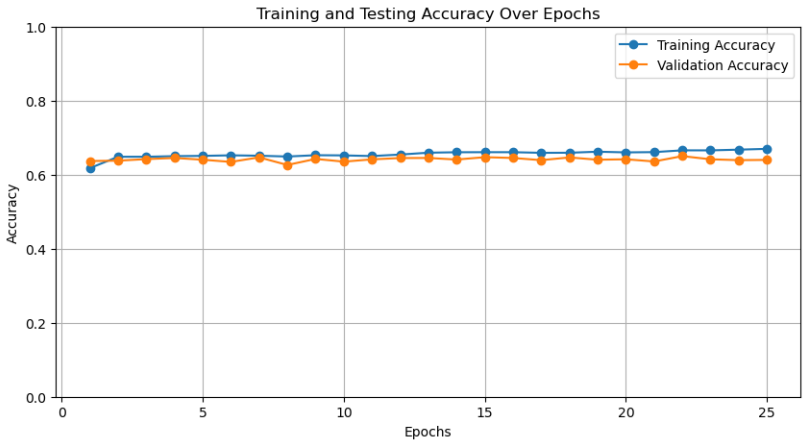

Figure 6: Plot of the training and validation accuracy through the model's training epochs

The Neural Network was designed to capture nonlinear relationships between features, and the final model was optimized using a set of hyperparameters listed in Table 6. The model achieved a validation accuracy of 63.53%, outperforming Logistic Regression but still falling short of XGBoost. The results of the hyperparameter search showed a preference for a larger network size compared to models such as in[5]. The small variation between the performance on the train, validation and test data success, shows we were successful in tampering the models desire to over fit but still struggled to perform well.

Table 12: Confusion Matrix for Neural Network Model

	Predicted 0	**Predicted 1**
Actual 0	1422	988
Actual 1	768	1642

Table 13: Classification Report for Neural Network Model

Class	**Precision**	**Recall**	**F1-score**	**Support**
0	0.65	0.59	0.62	2410
1	0.62	0.68	0.65	2410
Accuracy	0.64 (4820 samples)			
Macro Avg	0.64	0.64	0.63	4820
Weighted Avg	0.64	0.64	0.63	4820

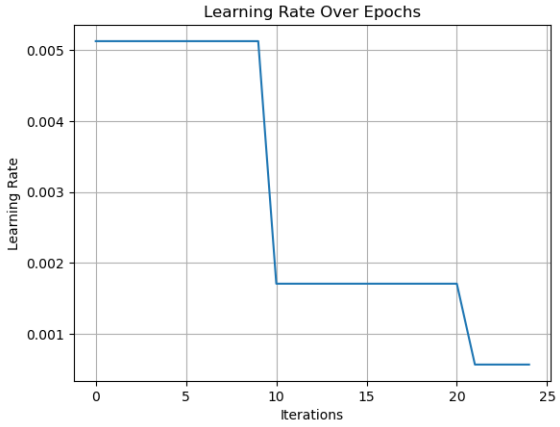

Figure 7: Plot of the changes in the learning rate introduced by the plateau recognizing scheduler

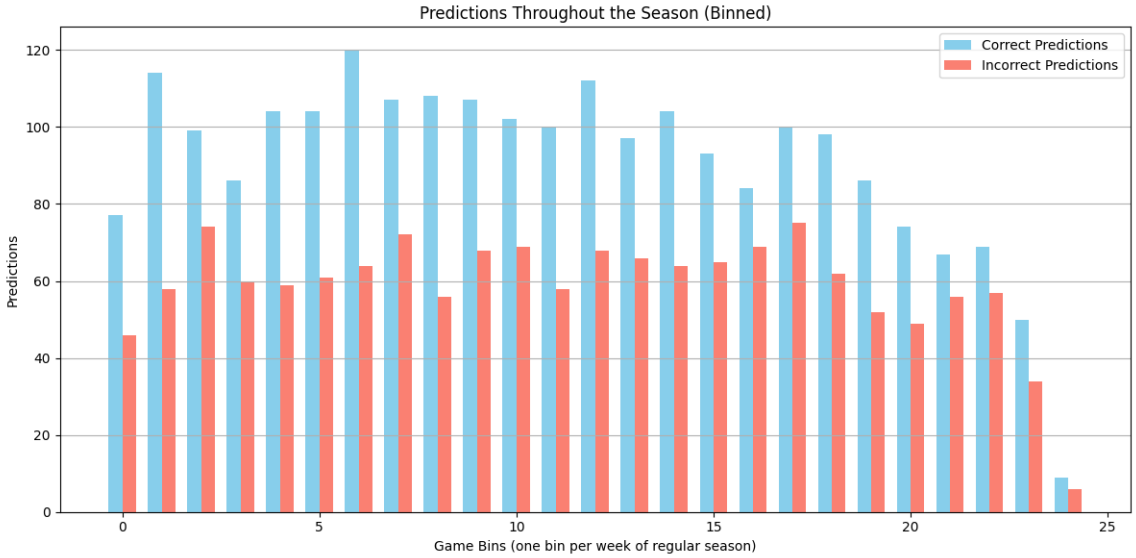

Figure 8: The sum of correct and incorrect scores throughout the season for the Neural network on the test dataset

3.3.4 Blended Model

The Blended Model combines the predictions of the Logistic Regression, XGBoost, and Neural Network models using a soft voting approach, with the aim of leveraging the strengths of each individual model. This ensemble method achieved a validation accuracy of 61.08%, which, while lower than the XGBoost model's performance of 64.23%, indicates a different balance in prediction strengths. Table 14 displays the confusion matrix, and Table 15 summarizes the precision, recall, and F1-scores for each class (win/loss).

Table 14: Confusion Matrix for Blended Model

	Predicted Win	**Predicted Loss**
Actual Win	1334	657
Actual Loss	990	1251

The confusion matrix highlights that the Blended Model tends to be conservative in its win predictions, as evidenced by a higher number of false positives compared to false negatives. This conservative tendency results in a relatively balanced distribution of predictions but suggests that the model may struggle to generalize as effectively as the individual XGBoost model.

Table 15: Classification Report for Blended Model

Class	**Precision**	**Recall**	**F1-score**	**Support**
Win	0.57	0.67	0.62	1991
Loss	0.66	0.56	0.60	2241
Accuracy	0.61 (4232 samples)			
Macro Avg	0.61	0.61	0.61	4232
Weighted Avg	0.62	0.61	0.61	4232

The classification report in Table 15 provides further insights into the model's prediction balance. The precision for predicting losses (0.66) is higher than that for predicting wins (0.57), indicating that the model is

more reliable when identifying teams that are likely to lose. Conversely, the recall for predicting wins (0.67) is higher, suggesting that the model is better at identifying wins when they actually occur, though it is prone to misclassifying some losses as wins. The F1-score, which balances precision and recall, hovers around 0.60-0.62 for each class, underscoring the model's ability to handle both classes with relative uniformity.

Overall, while the Blended Model does not outperform XGBoost in terms of raw accuracy, it offers a broader perspective by integrating insights from multiple models, potentially providing a more holistic understanding of game outcomes.

4 Discussion

4.1 Summary Analysis

The results of this analysis point to several key implications regarding the effectiveness of different machine learning models in predicting NBA game outcomes based on rolling statistics. These implications can be grouped into three main themes: model complexity, feature representation, and practical applications.

In trying to profile how our effective our models were, we examined how the odds of a correct prediction changed throughout the season. The results of this can be seen in figure 8. Games were binned by their approximate week in the season by removing the season identifier ordering by game number and binning by the number of weeks in each season. Surprisingly, there is not a significant trend in model performance throughout the season nor a significant loss in performance for the first week of the season in which teams do not have a full history of games being used for prediction.

4.2 Model Complexity

The performance of logistic regression, XGBoost, and the neural network highlights the trade-offs between model complexity and predictive power. While XGBoost outperformed logistic regression, the improvement was marginal, suggesting that more complex models don't always offer substantial improvements without the right tuning or data. This finding echoes similar studies in the sports analytics literature, where tree-based models like XGBoost often perform well but require carefully curated feature sets to reach their full potential.

The neural network, while capable of capturing non-linear and complex relationships, under-performed compared to XGBoost. This could be due to a number of factors. Such as more complex structures being necessary to find relations between our inputs. Or further refinement of our clustering approach may be needed to realise the gains available from the underlying data.

4.3 Feature Representation

The consistent performance across all models points to the strength of the rolling statistics and engineered features used in this analysis. These rolling averages encapsulate the team's form leading up to the game, offering a relevant snapshot of performance trends. However, the relatively narrow improvement margin from logistic regression to XGBoost suggests that these features, while useful, may still lack certain dimensions of game dynamics, such as further player-level metrics or situational factors such ass the home-court advantage is likely not the same for unpopular under performing teams as it is for popular teams.

The integration of individual player stats into a rolling team stats prediction is still in need of some refinement. While the approach did not produce the high accuracy results of other studies. When examined on their own the results were valid and produced interesting insight into teams makeup, changes in the league through the years as well as provided additional model performance over.

These findings are consistent with the broader literature, where feature engineering is often cited as a critical determinant of model success in sports predictions [8]. Incorporating player-specific data or more

granular in-game statistics (such as quarter, possession or minute-by-minute performance changes) could potentially boost model accuracy [6]. Additionally, while the blended model offered slight improvements, it likely reflected the redundancy in feature space across the individual models, meaning the same statistical patterns were being captured by each model in a similar way.

4.4 Practical Applications and Ethical Considerations

From a practical standpoint, the performance of these models - particularly the XGBoost model - suggests potential applications in real-time sports analytics, such as live betting odds generation or team performance monitoring. However, caution is warranted in how these predictions are interpreted and applied, especially when considering the ethical implications of predictive analytics in competitive sports.

The use of predictive models for sports betting introduces ethical concerns around fairness and transparency. Models that can accurately predict game outcomes could potentially skew betting markets or be used to manipulate odds, leading to ethical dilemmas about their role in gambling. Furthermore, reliance on these models without understanding their limitations could foster overconfidence, both in betting markets and in managerial decision-making within teams.

In the broader context of sports analytics, our findings add to the growing body of work showing that machine learning models can offer value in predictive sports applications. However, the results also emphasize the importance of responsible model use, especially considering the uncertainty inherent in predicting human behavior and performance. These models, while statistically powerful, should be seen as tools to assist decision-making, not replace it entirely.

5 Conclusion

5.1 Challenges

The prediction of NBA basketball games turned out to be far more complex than initially anticipated. Rather than finding a single, dominant indicator of game outcomes, our progress came from a combination of many smaller, incremental improvements. Identifying these refinements proved to be a major challenge. The time required to explore more advanced variations — such as clustering players, stacking models, and employing complex model types — was substantial, demanding considerable computational resources to execute and validate. This complexity was compounded by the constraints of a limited timeline and the computing resources available to us.

5.2 Next Steps

Overall, our approach still struggles to fully account for the intangible factors that influence basketball game outcomes. While our model attempts to capture some of these less quantifiable metrics, such as team chemistry, through rolling statistics, there remain significant elements, for example, psychological momentum shifts during games, that are difficult to incorporate. Accurately reflecting these factors without overfitting remains a major challenge. The key to improving our model lies in carefully selecting and continuously refining features until a design emerges that better captures these nuances.

References

[1] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. *NIPS: Neural Information Processing Systems*, 2011.

[2] G. Cheng, Z. Zhang, M. Kyebambe, and N. Kimbugwe. Predicting the outcome of NBA playoffs based on the maximum entropy principle. 18(12):450.

[3] N. Dobson. Fivethirtyeight's elo ratings and logistic regression.

[4] D. Lutz and D. Lutz. A CLUSTER ANALYSIS OF NBA PLAYERS. 2012.

[5] C. L. Osken and C. Onay. Predicting the winning team in basketball: A novel approach. *Heliyon*, 8(12):e12189, Dec. 2022.

[6] Y. Ouyang, X. Li, W. Zhou, H. Wei, W. X. Zheng, Q. Feng, and L. Peng. Integration of machine learning xgboost and shap models for nba game outcome prediction and quantitative analysis methodology. *PLOS ONE*, 19(7):e0307478, July 2024.

[7] Swar. Nba api. https://github.com/swar/nba_api, 2024. GitHub repository.

[8] A. Zimmermann, S. Moorthy, and Z. Shi. Predicting college basketball match outcomes using machine learning techniques: some results and lessons learned.

A Appendix A: Code Listing

Project code: https://github.com/khodgema/Capstone