



Article

Towards Predicting Basketball Player Positions with Transformative Insights

Angelos Tsiannis and Christodoulos Efstathiades *

Department of Information Technologies, School of Technology and Innovation, University of Limassol,
Nicosia 2107, Cyprus

* Correspondence: cefstathiades@uol.ac.cy; Tel.: +357-22462223

Abstract: Basketball players are traditionally classified into five positions. This study examines the correlation between player performance, game statistics, and designated positions. It also explores how statistical contributions have evolved over time. Machine learning classifiers were used to identify key metrics that distinguish player positions and determine the most effective classification algorithms. Our findings confirm a correlation between game statistics and positions, reinforcing the relevance of traditional roles. However, results also show that modern players contribute across multiple positions, reflecting a shift toward versatility. Despite this flexibility, players maintain distinct roles and responsibilities. This classification approach highlights key performance metrics and lays the groundwork for future clustering and mapping analysis, offering deeper insights into player roles and team dynamics in contemporary basketball.

Keywords: machine learning; sports data science; data mining; data analytics; sports statistics

Academic Editors: Phivos Mylonas
and Elias Dritsas

Received: 3 January 2025

Revised: 30 January 2025

Accepted: 2 February 2025

Published: 5 February 2025

Citation: Tsiannis, A.; Efstathiades, C. Towards Predicting Basketball Player Positions with Transformative Insights. *Mach. Learn. Knowl. Extr.* **2025**, *7*, 11. <https://doi.org/10.3390/make7010011>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The use of data has become increasingly important across various industries, including healthcare, finance, and commerce. The sports industry has also adopted data-driven approaches to gain a competitive edge and improve performance. Data analytics is transforming the world of sports by helping analyze players and team performance, designing injury prevention models, and predicting game outcomes. By utilizing data, sports organizations can make more informed decisions, identify trends, and tailor strategies to achieve desired outcomes, changing the way we perceive and experience sports.

This paper focuses on the application of advanced analytics and machine learning (ML) methodologies related to NBA player statistics. It aims to utilize data analysis techniques and provides insights into players' performance relative to their playing positions. By deploying machine learning algorithms and models, this paper aims to create predictive frameworks capable of determining player positions based on performance metrics. Essentially, this study seeks to understand and evaluate player and team performance based on gameplay and positional player analysis.

The scope of this study is to investigate the correlation between player positions and game statistics. In basketball, players are assigned specific predefined positions. This paper seeks to address two main questions:

- a. Are the game statistics correlated with player positions?

- b. Which classification model yields the most accurate results?

Moreover, this discussion serves as a starting point for examining the impact of classification errors for players' positions on the game and their roles in modern basketball.

A motivational presentation by Muthu Alagappan [1] raised the question of whether basketball players are defined correctly in five positions. The presentation revealed that players should be classified into thirteen (13) positions instead of the traditional five (5) positions. The utilization of advanced clustering techniques and topological analytics [1] demonstrated that players playing in the same position can offer varying statistics and contributions to their teams. Ultimately, it showcased that understanding player-revised positions and roles enables coaches to strategically devise offensive and defensive team plays.

We perform a review of the related literature in Section 2, where we describe the relevant work on the topic by highlighting techniques, features, and metrics that have been used. This section serves as the foundational step that guides the methodology which follows. Section 3 presents the data analytics process that was used to provide relevant insights and understanding of the dataset. Finally, Section 4 demonstrates the implementation of different classifiers, showcasing hyperparameters and features that have been used. Moreover, we evaluate the efficiency, effectiveness, and quality of the results using different performance metrics. This enables discussion and additional exploration of whether the statistical aspect of the game is related to the players' positions.

2. Related Work

Research in sports analytics has caught the attention of data science researchers and has grown tremendously in recent years. The employment of machine learning techniques enhances the sports analytics process and opens the ground for rigorous analysis of the complex data generated by the sports industry [2]. Similarly to other sports, basketball caught the attention of data scientists due to the multi-dimensional parameters that can lead to different projections and predictions [3,4]. During a game, a basketball team consists of five players each assigned to a specific position. The positions are Point Guard (PG), Shooting Guard (SG), Small Forward (SF), Power Forward (PF), and Center (C). Traditionally, taller players are used as *Centers*, while shorter and quicker players are designated as *Point Guards*. This classical structure was determined through observation and consideration of various human characteristics that would aid each team in winning the game. In addition, the work described in [5] examines whether human characteristics, anthropometric measurements, and physiological features can play a key role in identifying a suitable position for a player.

Furthermore, the study conducted in [5] focused on implementing a model to predict player positions based on their physical characteristics. They gathered data on attributes such as height, weight, body fat, and muscle. Following the data modeling process, three different classifiers were utilized to predict positions, and the performance of each algorithm was evaluated. The algorithms employed in this study included a Multilayer Perceptron based on Neural Networks (MLP), a Linear Support Vector Machine Classifier (SVM), and a Decision Tree Classifier (DT). The evaluation identified DT as the most accurate classifier, followed by SVM and MLP. Finally, the authors suggested that gaining a deeper understanding of SVM and incorporating Extreme Gradient (XG) Boost models could potentially enhance the results.

In addition, the classification of players based on their game statistics accumulates significant attention from analysts. Several works aimed to predict players' position by selecting a limited number of statistics such as assists, rebounds, and shooting percentages. The methodology employed by [6] involved evaluating the performance of

a Random Forest Classifier (RF), Decision Tree Classifier (DT), and K-Nearest Neighbors (KNN) algorithm. Overall, the evaluation revealed that the Random Forest Classifier indicated the highest accuracy. However, the limited size of the dataset resulted in a relatively low prediction rate. Conversely, when approaching the same topic with a larger dataset and more features, the Random Forest algorithm demonstrated enhanced effectiveness [7]. The study compares Random Forest Classifiers (RF), Support Vector Machines (SVM), Decision Tree Classifiers (DT), and Gradient Boosted Trees. It is noteworthy that scaling functions and Principal Component Analysis (PCA) contributed to the enhanced performance of the models. Incorporating these techniques into the classifier design could prove beneficial in improving the efficiency of the algorithms used.

The selection of game statistics and player characteristics plays a crucial role in the predictive model. Enhancing the dataset with additional statistics like drives, touches, and deflections can improve the efficiency of the algorithm. In [8], the author utilized nineteen (19) features to classify the players into their respective positions, focusing on one season game statistics. The study applied Multinomial Logistic Regression (MNLR), a Linear Discriminant Analysis (LDA) model and a multi-class classification with an SVM to predict players' positions. The outcome of this analysis shows that the SVM was the most effective algorithm to use. It is worth noting that leveraging a larger dataset could further increase the model's performance [8].

In addition to basketball, several approaches have been made to predict players' position in other sports. When identifying positions in football, a methodology combining physical and performance characteristics was performed. The primary objective of the work in [9] was to determine the optimal combination of hidden layers and learning rates in a backpropagation (BP) neural network to attain the highest achievable accuracy. In this study, thirteen features were utilized to generate indicators and *p*-values. The model achieved a 77% accuracy rate through the utilization of six hidden layers and a learning rate of 0.0125 [9].

An additional study carried out by [10] used a method in football to predict player positions based on spatial and physical movement data. The research aimed to categorize the football players into five different positions: center-back, wing-back, wing player, midfielder, and forward. To accomplish this, the authors evaluated the performance of three machine learning models. They implemented a Convolutional Neural Network (CNN) and two Random Forest (RF) models. The RF models utilized a model from the sklearn [11] library and an additional model with a gradient boosting procedure from the XGBoost library [12]. The findings revealed that among the three models, the RF algorithm combined with XGBoost yielded the highest accuracy, reaching levels of 80% [10].

Throughout the paper, considering the literature, it is evident that specific classification algorithms prevail. Decision Trees (DT) are robust algorithms that can perform classification, regression, and multi-output problems in a complex dataset [13] and can make accurate predictions, and enable easier verification of calculations behind these predictions [11,13]. Random Forests (RF) are an ensemble of decision trees that are trained via bagging or a bootstrap aggregating method [13], where each tree builds on a different dataset, and is different from all other trees [14]. The Random Forest capability to reduce prediction variance and to prevent overfitting increases the algorithm's accuracy [11,15]. The Multinomial Logistic Regression classifier (MNLR) uses a statistical method to predict categorical targets which belong in more than two classes. MNLR is widely used due to its ability to model complex relationships among variables and create a multi-category classifier. MNLR is noted for its flexibility compared to the common logistic regression making it suitable for more applications [16]. The Support Vector Machine (SVM) is a powerful and versatile machine learning model. SVM usually performs well with linear and nonlinear boundaries depending on the kernel that would be used. SVMs,

when used in medium size datasets, are memory efficient making them a preferred choice compared to the wider range of machine learning applications [13,16,17].

Summarizing the literature, it is evident that certain guidelines emerge which prove beneficial for the objectives of this paper. The size of the dataset is identified as a critical factor influencing the analysis results. Therefore, it is important to collect as much game statistics data as possible. In this study, we will gather data spanning more than ten seasons. This extensive dataset facilitates the cleaning and selection of high-quality data for the model. Furthermore, the choice of classifier for the prediction is paramount to achieving positive and effective outcomes. In the subsequent Sections, we will delve deeper into several classification algorithms to gain a better understanding of their usage and suitability based on the desired outcomes.

3. Materials and Methods

3.1. Feature Selection

In this section, we explore the extensive body of literature to pinpoint the pivotal variables and features essential for developing a classification algorithm. This part of the paper is essential in delineating the crucial variables for our dataset. Features are defined as the attributes or independent variables describing the labels [13].

The selection of features related to the classification can vary across different domains. In sports analytics, attributes can be related to players' physical characteristics, youth development, team style of play, and game statistics. As previously discussed, whether in basketball or football, physical characteristics emerge as crucial features to consider [5,9]. Nevertheless, for the purpose of this paper, we will be focusing mainly on game statistics and metrics that can be calculated through game statistics.

In the previous section, we recognized that the number of features play an essential role in the efficiency of the model. At this stage, we examine the related research to identify the key features from game statistics that we need to incorporate into our dataset. Firstly, [6] attempted to limit the size of the variables to six. This selection, according to the author, was made because these statistics have been considered as being the most important. The selected features were as follows:

- a. three shooting percentages (Field Goal Percentage—FG%, Free throw percentage—FT%, and 3-point percentage—3P%);
- b. assists (AST);
- c. rebounds (REB);
- d. blocks (BLK).

As these variables are considered, the study described in [7] implemented the models using twenty (20) features. Based on this study and using a Decision Tree classifier, the most important features were the total rebounds (TREB), assists (AST), steals (STL), blocks (BLK), personal fouls (PF), FT%, and 2-point attempts (2PA). Surprisingly, compared to the previous study, the FG% and 3P% were of lower importance. In the same study, [7] showed that TREB, AST, STL, BLK, 2PA, and free throw attempts (FTA) were again the key features in an SVM classifier.

The literature includes studies that explored additional and more advanced game statistics. Throughout the implementation of a Random Forest classifier, the author of [18] had an initial dataset with forty-two (42) features. During the modeling stage, it was decided to proceed with features related to offensive play and to not use any defensive statistics. These features included shooting attempts and percentages from various distances. This involved dividing 2P and 3P shoots into several distance ratios of 3 feet to provide a more precise shooting area for the players. In addition to the shooting percentages, features such as FTA, FT%, AST, Turnovers (TOs), assist-to-turnover ratio

(A/TO), offensive rebounds, and offensive rebound percentages were used [18]. A larger dataset, consisting of one hundred (100) features, was used during the development of a classifier based on tracking statistics from Sport-VU cameras [8]. This work proceeded with a decreased size of nineteen features. Among those features were the average minutes played, blocks (BLKs), steals (STLs), assists (ASTs), assist-to-turnover ratio (A/TO), and shooting percentage. Similarly to the previous study, shooting attempts were structured differently by investigating the type of shooting attempts based on gameplay style, such as catch-and-shoot or pull-up attempts. Finally, the per-36 min metrics on several game statistics were used as a feature; for example, assist per 36 min [8].

The combination of variables and aggregated metrics during feature selection is also worth considering. The per 36 min ratio which has been used is a weighted ratio that illustrates the contribution of a player to a specific aspect of the game, if all players were to play the same number of minutes (36 min consisted of 75% of total playing time). For example, in assists, this is calculated by dividing the total number of assists by the minutes played and then multiplying by 36. These weighted metrics were also calculated and identified as key features during a cluster analysis [19]. The author also used standard shooting percentages, like 2PA, 2P% etc., defensive statistics (STLs and BLKS), and ASTs.

It is also important to examine the features that have been used in clustering studies. In an approach to redefine traditional positions, game statistics have been combined with other features like salaries, age, and physical characteristics. Looking into the game statistics, variables like rebound percentages, ASTs, STLs, BLKs and personal fouls (PF) were used [1]. Following the same concept, during a clustering analysis, standard statistics with aggregated metrics were utilized. The initial dataset consisted of eighteen (18) variables, all related to game statistics. Ratios regarding assists with turnovers and rebounding were also calculated. Additionally, the per 36 min ratio was applied to some critical data statistics such as assists and points [20].

The summary of features and their usage throughout our exploration is depicted in Table 1. Our focus will be on game statistics and metrics derived from them, to examine how they are correlated to the playing positions. Table 1 delineates grouped variables utilized in classification and clustering implementations.

Table 1. Summary of feature usage in the literature.

Game Statistics	[1]	[6]	[7]	[8]	[18]	[19]	[20]
Shooting Attempts. FGA, 2PA, 3PA	☐	☐	☒	☒	☒	☒	☒
Shooting Percentages. FG%, 2P%, 3P%	☐	☒	☒	☒	☒	☒	☒
Advance Shooting Percentage. Distance or style of play oriented	☐	☐	☐	☒	☒	☐	☐
Free throw shooting Attempts and Percentage	☐	☒	☒	☒	☒	☒	☒
Rebounds. Offensive and Defensive	☒	☒	☒	☒	☐	☒	☒
Rebounds ratio	☒	☐	☐	☒	☒	☐	☒
Assists	☒	☒	☒	☒	☒	☒	☒
Turnovers	☒	☐	☒	☒	☒	☒	☒
A/TOs	☐	☐	☐	☒	☒	☐	☒
Steals	☒	☐	☒	☒	☐	☒	☒
Blocks	☒	☒	☒	☒	☐	☒	☒
Per-36 min metrics	☐	☐	☐	☒	☐	☒	☒
Personal Fouls	☒	☐	☒	☐	☐	☐	☒

3.2. Algorithm Performance Metrics

After thoroughly examining the relevant literature, it is important to pinpoint the key performance metrics utilized in classification algorithms. These metrics serve as crucial indicators to assess the effectiveness and accuracy of various algorithms. We will examine the primary performance metrics employed in the field.

Across all research works, accuracy emerges as the paramount metric. Accuracy is defined by the ratio of correctly predicted class labels to the total labels. In addition to accuracy, studies in the literature employ a confusion matrix. This matrix serves as a tool for evaluating classifier's performance. Upon implementing the confusion matrix, we can define additional metrics for assessing the performance of classification algorithms. The first metric is precision, which signifies the ratio of true positive predictions to the total predicted positive values (true positives and false positives). Precision aids in identifying the accuracy of positive predictions, highlighting instances where incorrect predictions may lead to negative outcomes. A higher precision value correlates with a lower false positive rate. The second metric, recall, quantifies the ratio of true positive predictions to the total actual positive values (true positives and false negatives). Recall provides insight into the completeness of positive predictions made. Finally, the F1 score combines precision and recall, offering a comprehensive measure of performance that is independent of dataset size. F1 score serves as the weighted average between precision and recall, providing a balanced assessment of model effectiveness [14,21].

Among the plethora of metrics, accuracy and the confusion matrix emerge as the most prevalent in relevant research. These metrics serve as primary benchmarks to measure the efficiency of models. Furthermore, certain papers incorporate additional metrics, including recall, precision, and the F1 score, to facilitate a more comprehensive comparison of classification algorithms. Table 2 shows an overview of the key performance metrics extracted from the literature for evaluating classifier performance.

Table 2. Summary table of performance metrics used in classification models.

Performance Metrics	[5]	[6]	[7]	[8]	[9]	[10]	[18]	[21]
Accuracy	☒	☒	☒	☒	☒	☒	☒	☒
Confusion Matrix	☒	☐	☒	☒	☒	☒	☒	☒
Precision	☐	☐	☒	☐	☐	☐	☒	☒
Recall	☐	☐	☒	☐	☐	☐	☒	☒
F1 Score	☐	☐	☒	☐	☐	☐	☒	☒

3.3. Data Analysis

The first phase of our analysis consists of collecting and loading the data. The dataset was downloaded in CSV format from Basketball-Reference [22]. This dataset was selected as it was used from several reports in the literature to obtain the NBA game statistics. This was used by [6–8,18–20]. We acquired total player statistics for NBA players ranging from the 2004 season to the 2023 season. Each season is stored in a separate CSV file. The raw dataset consists of 12,407 rows and 32 columns. Table 3 outlines the attributes presented by each column [22].

Table 3. Dataset Glossary.

Column Name	Description
Rk	Rank integer used on website
Player	Full name
Pos	Position
Age	Age during the season (referenced date 1 February each season)

Tm	NBA Team
G	Games Played
GS	Games Started
MP	Minutes Played
FG	Field Goals Made
FGA	Field Goals Attempts
FG%	Field Goals Percentage. Field Goals Made divided by Attempts.
3P	Three-Point (3-point) Shots Made
3PA	3-point Shots Attempts
3P%	3-point Shots Percentage
2P	Two-Point (2-point) Shots Made
2PA	2-point Shots Attempts
2P%	2-point Shots Percentage
eFG%	Effective Field Goal Percentage. Weighted field goal percentage since 3-point worths 1 point more than 2-point.
FT	Free Throws Made
FTA	Free Throws Attempts
FT%	Free Throws Percentage
ORB	Offensive Rebounds
DRB	Defensive Rebounds
TRB	Total Rebounds
AST	Assists
STL	Steals
BLK	Blocks
TOV	Turnovers. Mistakes made by player.
PF	Personal Fouls
PTS	Points
Player-additional	Player's unique identification key [22]
Season	Season–Year

The next phase is to clean and model the data. We begin by removing the column 'RK'. This contains the index of the ranking available on the website from which we downloaded the raw data. However, this column does not provide any meaningful information to the dataset used.

While exploring the data, we noticed that in the column 'Tm', there is a value called 'TOT' that does not represent an acronym for any NBA team. Upon investigation, we identified that the dataset provider aggregates the total number of player statistics per season and assigns this value for the team. For instance, in the 2022 season, James Harden played some games for the Philadelphia 76ers before moving to the Brooklyn Nets. Therefore, in the dataset, we see three rows for James Harden in 2022 with PHI, BRK, and TOT. We have decided to remove the rows where the column 'Tm' is equal to 'TOT' and re-create our metrics. After dropping these rows, the dataset consists of 11,137 rows and 31 columns. After removing these rows, the 'Pos' column is verified. This is a crucial column for the analysis as it will be used as the label column for classification algorithms. It is important to have only five specific values in this column: PG, SG, SF, PF, and C, which represent the playing positions. By aggregating the data, we verified the existence of those five positions, and Table 4 shows the record count for each position allocation. It appears that the distribution follows an even pattern, with data almost equally allocated to each position.

Table 4. Distribution of playing positions.

Position	Records Count	Positions Allocation
PG	2160	19.39%
SG	2403	21.58%
SF	2064	18.53%
PF	2284	20.51%
C	2226	19.99%
Totals	11,137	100%

As a next step, we aggregate the data at the player, season, and position levels. This will provide one record per player per season with their game statistics related to the playing positions. This aggregation will alter the values of shooting percentages, so we will drop the following columns: 'FG%', 'FT%', '3P%', '2P%', and 'eFG%'. Additionally, columns 'Tm' and 'Age', which do not provide any meaningful information, have been removed. After dropping these columns, we aggregate the rows per player, season, and position, summing the total values to generate one row per player per season. Before proceeding with additional modeling, we check the dataset to ensure that a player has not played in two or more different positions in the same season. We have identified that there are 403 rows that do not pass this validation. As this is less than 0.5% of our dataset, we proceed by dropping these rows.

In the literature review, it was identified that shooting percentages are an important feature. Upon examination of the references in Table 1, standard shooting percentages as well as more advanced shooting statistics are used in the literature. Hence, the percentages for 'FG%', 'FT%', '3P%', '2P%', and 'eFG%' are calculated using the formulas (1) and (2) below. At this point, the True Shooting Percentage (TS%) is introduced (Formula (3)). The TS% is a weighted shooting percentage that highlights a player's shooting efficiency for every type of shot.

$$\text{Shooting Percentage} = (\text{Successful Attempts Made}) / (\text{Total Attempts}) * 100 \quad (1)$$

Formula (1) is applicable for FG%, FT%, 2P% and 3P%, i.e., $2P\% = 2P/2PA * 100$

$$eFG\% = (FG + 0.5 * 3P) / FGA * 100 \quad (2)$$

$$TS\% = PTS / (2 * (FGA + 0.44FTA)) * 100 \quad (3)$$

When the shooting percentages have been recalculated, we can define additional ratios. As noted in the literature review, analysts frequently utilize assist turnover ratio and rebounds ratios. Furthermore, defensive efficiency will be determined by dividing the total blocks by personal fouls. This would also be calculated for steals as well. Another calculated ratio is the minutes per game, which indicates the average playing time of each player and will prove useful in the final data cleaning process. Lastly, as indicated in Table 1, per-36 metrics are of significance. Thus, we will compute the weighted per-36 min metrics for five game statistics: AST, TRB, PTS, STLs, and BLKS.

$$AST/TO = AST/TOV \quad (4)$$

$$STL/PF = STL/PF \quad (5)$$

$$BLK/PF = BLK/PF \quad (6)$$

$$DRB/TRB = DRB/TRB \quad (7)$$

$$ORB/TRB = ORB/TRB \quad (8)$$

$$\text{Metric per 36 min} = (\text{Metrics Total}) / (\text{Minutes played}) * 36 \quad (9)$$

Formula (9) is applicable for AST36, TRB36, PTS36, STL36 and BLK36, i.e., $\text{AST36} = \text{AST}/\text{MP} * 36$

Finally, the last step involves removing players with low playing activity. To determine low activity, we calculate values at five different percentiles: 10%, 25%, 50%, 75%, and 90%. These values are calculated for the average minutes per game and the number of games played. Figure 1 and Table 5 display the percentiles for each measure. We observe that 10% of players have played fewer than ten games per season. This percentile was chosen as it captures the lower tail of the distribution, representing players with unusually low participation, whose contribution to game statistics was considered less significant due to limited playing activity. Figure 1 presents the distributions and percentiles for both metrics, highlighting in red the different percentiles which are calculated.

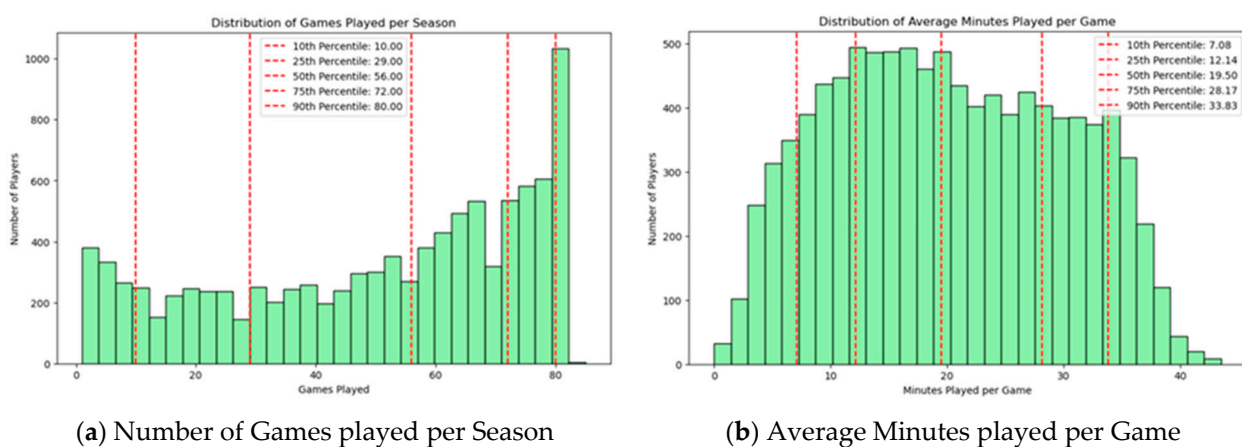


Figure 1. Histograms with the distribution and percentiles for (a) the number of games played per season and (b) the average minutes played per game, to facilitate outlier detection during data pre-processing.

Given that each season consists of 82 games, playing less than this threshold would not yield an acceptable level of accuracy for our model. Additionally, the total playing time per game is 48 min, and it is noted that 10% of players played less than 7 min per game. Therefore, we will remove every row where a player played less than ten games per season or had an average playing time of less than 7 min.

Table 5. Percentiles of games played and average playing time.

Percentile	Games Played ('G')	Average Minutes Played ('MIN/G')
10%	10	7.07
25%	20	12.14
50%	56	19.50
75%	72	28.17
90%	80	33.83

After carefully refining the dataset, we now have a solid base to explore various trends in player performance and team dynamics. The final dataset consists of 8518 rows and 42 columns.

We proceed in a data exploration phase to uncover potential trends and insights. Figure 2 shows a notable trend of a rapid increase in players adopting the shooting guard (SG) position over the years. It appears that more and more players are being deployed in

this role. This can be observed as an escalating number of players occupying these positions from 2018 onwards. Additionally, there is a noticeable drop in the number of point guards. This might indicate a shift in positional dynamics, where playmaking responsibilities are now distributed across all players.

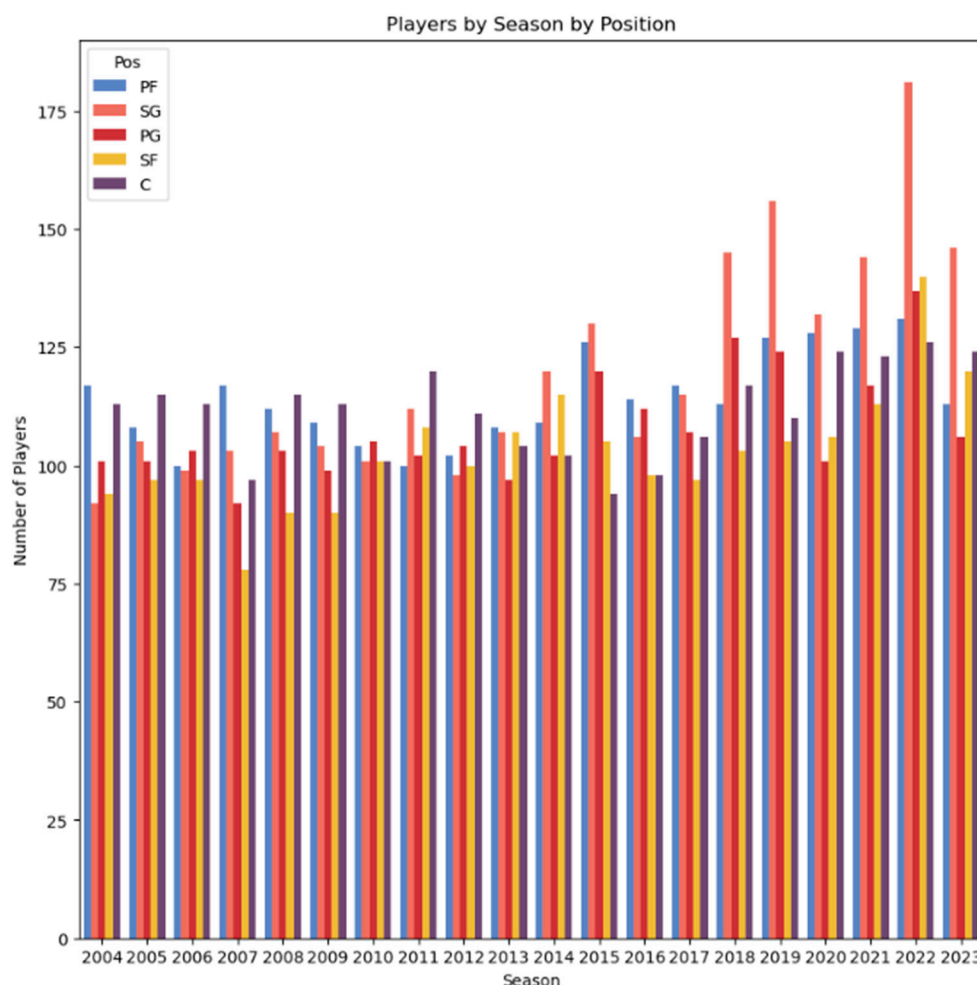


Figure 2. Position count over the years.

The box plots in Figure 3 illustrate the four main game statistics related to playing positions. Players in the PG position emerge as the primary playmakers, as expected. Interestingly, players labeled as Centers (C) indicate lower contributions in assists. However, upon closer examination of the box plots of Figure 4 representing (C) position over the years, a positive trend emerges: an increasing trend in assists from this position, suggesting evolving dynamics. Additionally, upon observing the TRB box plot in Figure 4, we see that the primary rebounding options for teams are Centers and Power Forwards (PF). Lastly, Small Forwards (SF) appear to demonstrate an all-around contribution, providing substantial amounts in both assists and rebounds, contrasting with other positions.

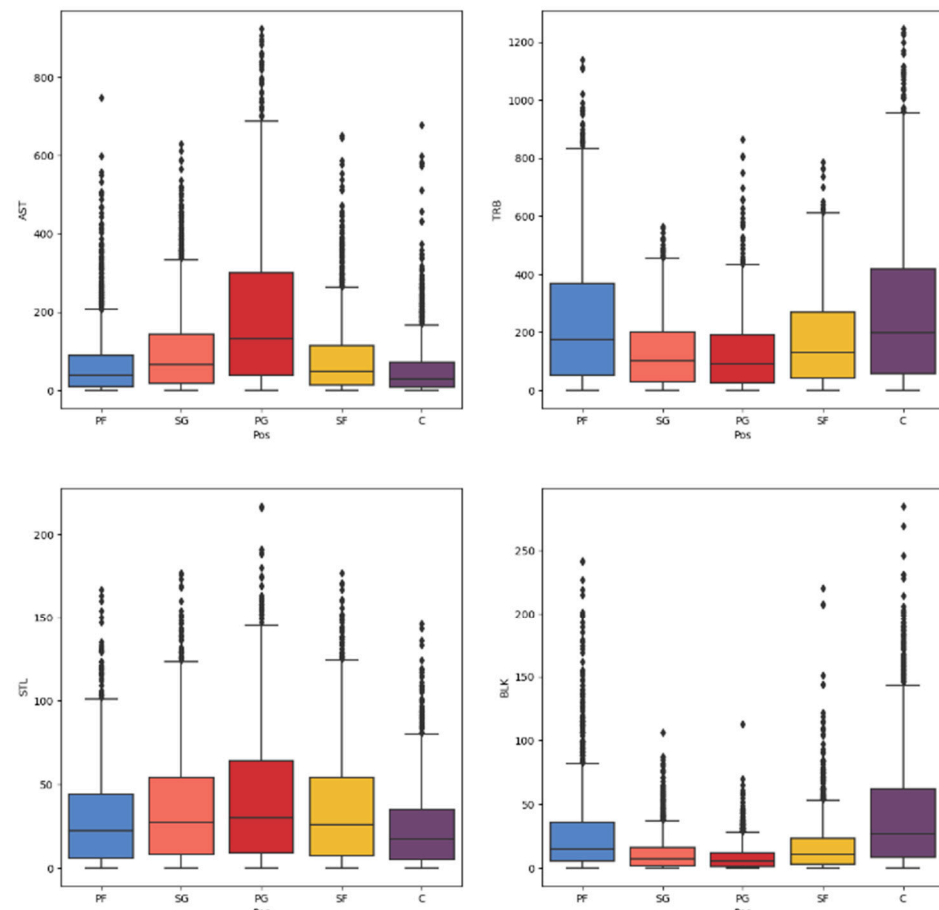


Figure 3. Game statistics boxplots per position (all time data).

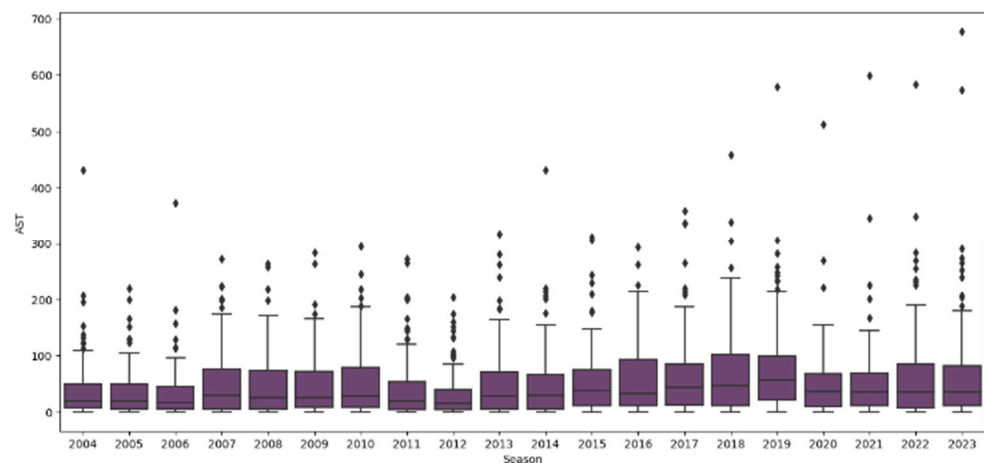


Figure 4. Boxplots assist over years analysis for Centers (C).

Moving on to defensive statistics, we observe that steals are generally distributed across every position, although higher totals can be seen among PGs. All other three positions—SG, SF, and PF—contribute to defensive plays, particularly in deflecting and stealing the ball. Alternatively, blocks are primarily contributed by players in the C position. Finally, Figure 5 pinpoints the distribution of players based on steals and blocks. It is notable that many players contribute to steals regardless of their positions, with a noticeable increase among SGs. The spike in block contributions from Cs is evident in this distribution, with players clustered at higher y -axis values and lower x -axis values.

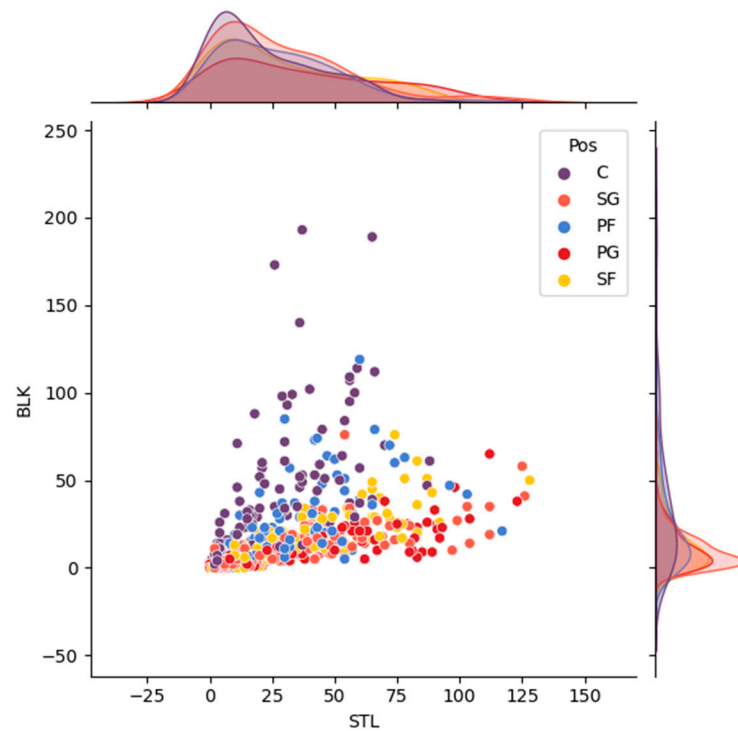


Figure 5. Scatter plot STLS and BLKS for 2023 statistics.

4. Experimental Results

4.1. Experimental Comparison of Literature

In the initial phase of implementing classifiers, a critical step involves delving into the several models outlined in the literature review. This section examines and tests each model in the journey of identifying the best performing algorithm for our investigation.

Before executing the different models, it is important to mention the columns that have been removed from the dataset. The initial stage involved removing the columns ‘Player-additional’, ‘Player’, ‘index’, ‘Season’, ‘MIN/G’, ‘G’, ‘GS’, and ‘MP’, as they provided player characteristics that are not relevant to the game statistics. For example, minutes or games played were not selected to be used for the position predictions. The column used as the label was ‘Pos’. Therefore, the final dataset consists of thirty-three (33) features.

We first implement a decision tree algorithm and thus the first step is to identify the parameters used in the literature to tune models. We therefore identify that the literature uses an 80/20 split of training and testing datasets. Moreover, in all cases, the literature uses the sklearn library [11] for the implementation. Finally, the parameters that the literature tunes in their implementation are only the random state, while the rest are left as the default options provided by the library. Table 6 summarizes the key findings and parameters.

Table 6. Summary of Decision Tree Parameters.

	[5]	[6]	[7]
Train–Test Split	80–20	75–25	80–20
Random_state	Not defined	5	1
Library–Model	Mentions SKLEARN	DecisionTreeClassifier	DecisionTreeClassifier
Reported Accuracy	90%	40%	61.7%

In executing the algorithms for the models, the results were not so encouraging. Overall, the results vary around an accuracy of 55% for each model, as only the random state is changed during each run. Firstly, compared to the model suggested in [5], we identified a significant variation in the results, with a calculated accuracy of 55% compared to the authors' reported 90%. It is important to note that in this paper, the researchers used data with a total of 250 features with anthropometric data (such as height), and since no pruning was applied, there is a possibility of overfitting. Compared to [6], the high variation in the results is due to the study using one season of clean data (around 500 rows) and only 6 features, whereas we used around 11,000 with 33 features. Lastly, the results compared to [7] seem promising as the difference is low, from 61.7% provided compared to the calculated 55%. This can be attributed to the difference in dataset size, 11,000 compared to 4000, and 33 features compared to 20.

Taking this into account and considering that level of performance, we decided to implement an additional Decision Tree classifier. To ensure comparability with the previous implementation, we set the criterion to 'Gini' impurity and did not modify it to a different quality split function. Upon testing, switching the criterion parameter to 'entropy' performance improved slightly over the default 'Gini': accuracy of 63.9% compared to 63.1%. Upon reviewing the literature, it is recommended to set the maximum depth to a level that would avoid tree pruning. An estimated number of tests suggest using an integer near the $\log_2$ of the number of features for maximum depth value. When the model was tested using a maximum depth of five and six, better performance was found with six leaves being optimal. The random state was set to 42 [13,14].

In Table 7, we illustrate the comparison between the four DT main models. It is important to highlight that setting the max depth parameter of the model to six improves the model's performance as it is likely to avoid overfitting. In all other cases, the training accuracy was 100%, which is an indication of overfitting. This implementation led to an increase in accuracy by 6–7%. Figure 6 shows the confusion matrix for the decision tree. It seems that PG and C predictions have better results, while the remaining three positions (SG, SF, and PF) varied in mid-levels.

Table 7. Comparison of Decision Tree Performance Metrics.

	DT Classifier	[5]	[6]	[7]
Accuracy Training	68%	100%	100%	100%
Accuracy Testing	63%	54%	56%	56%
Precision	63%	54%	56%	55%
Recall	63%	55%	56%	55%
F1 Score	63%	54%	56%	55%

Additionally, when plotting the feature importance (Figure 13), we notice that the weighted metrics of per-36 min played a crucial role in the decision-making process. It seems that the top five key features were TRB36, AST36, BLKS36, STL/PF, and PTS36. TRB36 is the main feature of the model, as it is used by the first node when the model tries to allocate players to their positions. This can be seen in Figure 7, where the pivotal role of TRB36 is clearly depicted in guiding the classification process.

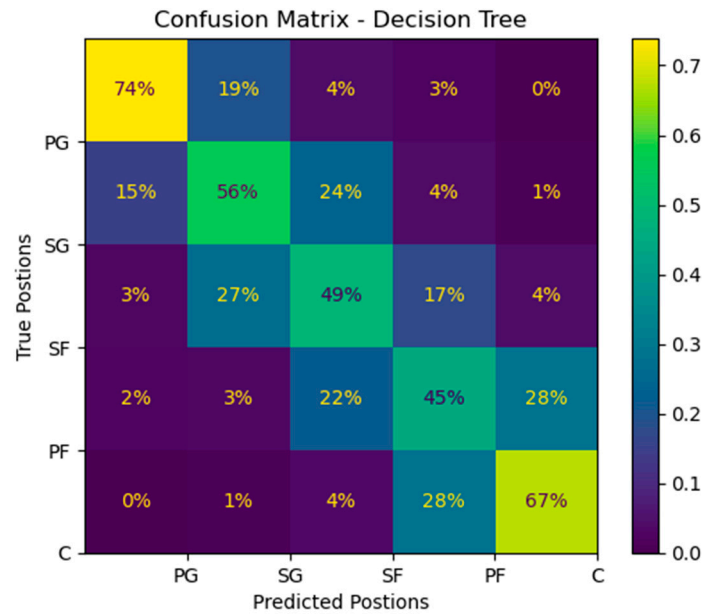


Figure 6. Decision Tree Confusion Matrix.

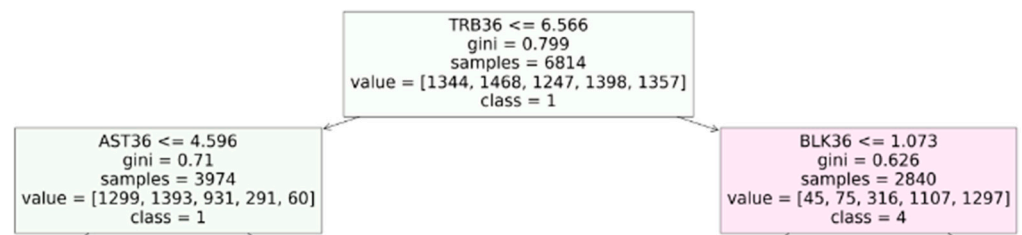


Figure 7. Decision Tree starting nodes.

Following the decision tree classification, our attention now shifts to implementing a model based on random forest trees, known for their robustness and ability to handle complex datasets. While variations exist in dataset split across references, maintaining a consistent 80/20 split for the training versus testing dataset ensures comparability. For the model described in [10], default parameters from the sklearn library [11] are employed, while parameters for the remaining models, shown in Table 8, align with those proposed in the literature, ensuring a comprehensive evaluation.

Table 8. Random Forest (RF) Summarization of parameters.

	[6]	[7]	[10]	[18]
Train–Test Split	80–20	80–20	60–40	75–25
Random_state	2	1	Not defined	default
N_estimators	default	500	Not defined	150
Max_depth	10	default	Not defined	4
Library-Model	SKLEARN Ensemble Random Forest Classifier	SKLEARN Ensemble Random Forest Classifier	Mentions SKLEARN	SKLEARN Ensemble Random Forest Classifier
Reported Accuracy	55%	71%	80%	63%

During the testing phase, we see that three of the suggested parameter models achieved an accuracy level of 69%. The number of estimators (n_estimators) and the maximum depth of leaves (max_depth) play a crucial role in avoiding overfitting. We observe that the model used in [10] with default parameters returned a training accuracy

of 100%, and the model used in [7] returned 100% even though the number of estimators was set to 500. In an effort to optimize the results and anticipate potential overfitting issues, a model was implemented with parameter adjustments. In [13], the number of estimators was set to 500, the max_leaf_nodes was set to 16, n_jobs to -1, and the random state to 42. These adjustments were made with the aim of improving model performance. As shown in Table 9, upon analyzing the performance outcomes, although the training accuracy decreased (suggesting reduced overfitting), the overall accuracy dropped by 5% compared to other models. Additionally, precision, recall, and F1 score each showed a decline of around 5%.

Table 9. RF Models Performance Summary.

	RF Model	[6]	[7]	[10]	[18]
Accuracy Training	65%	86%	100%	100%	64%
Accuracy Testing	64%	69%	69%	69%	63%
Precision	63%	69%	69%	69%	63%
Recall	64%	69%	69%	69%	63%
F1 Score	63%	69%	69%	69%	63%

Henceforth, for the Random Forest classifier, the parameters will be tuned based on the suggestions of the model [6], even though they were originally utilized for a dataset with a lower number of features and records. This model yields higher metrics across all accuracy metrics and does not tend to overfit on the training dataset like the other models. The confusion matrix (Figure 8) demonstrates that the model predicted well for three out of five positions: PG, SG, and C. This follows a similar trend to the DT algorithm. On the RF model, the top five features were TRB36, AST36, BLK36, AST/TO, and 3PA, as shown in Figure 13. The top three features are the same as the DT algorithm which outlines that per 36 min metrics could essentially boost the performance of the models.

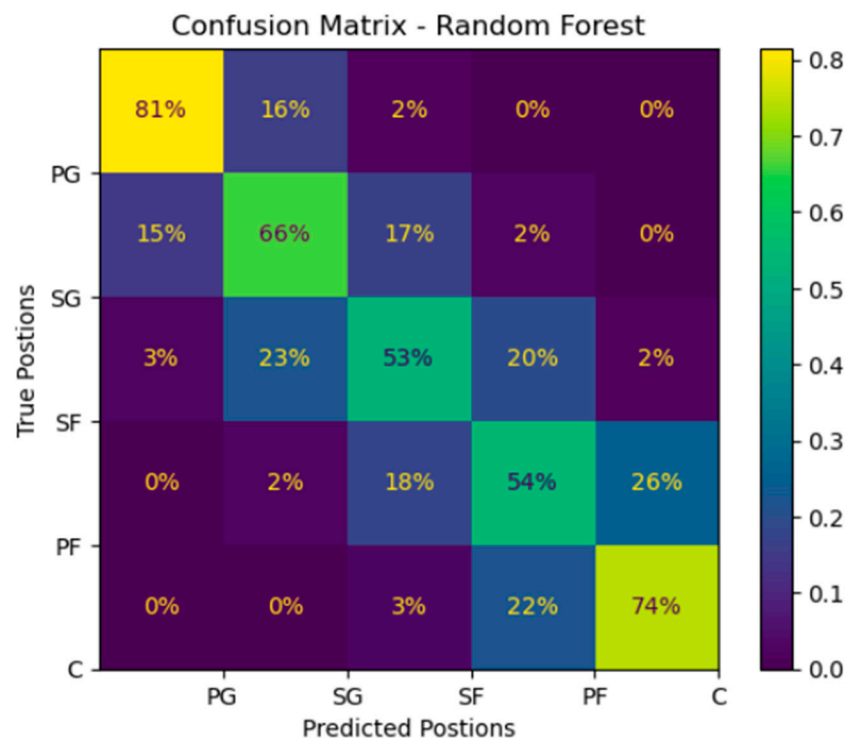


Figure 8. Confusion Matrix of RF Classifier.

The next option is to implement a multinomial linear regression model. We ran the model with the parameters provided by the study in [18], which provided an accuracy level of 73%. Table 10 presents the summary for the MNLR models parameters. The scaler that is used for these models is the standard scaler from scikit-learn (SKLearn). The dataset split ratio is 80/20 as with the previous models.

Table 10. MNLR Summary Table.

	[8]	[18]
Train–Test Split	70–30	75–25
Solver	liblinear	newton-cg
max_iter	100	1000
C ¹	Not defined	0.01
Library-Model	R nnet library	SKLearn Linear_Model
Scaler	Not defined	StandardScaler
Reported Accuracy	30%	73%

¹ Inverse value of regularization strength [11].

The execution of MNLR resulted in accuracy levels ranging from 65% to 69%. This is slightly lower than the reference, but it could be attributed to differences in dataset size in terms of features and records. The scaler used for this model was the standard scaler provided by sklearn. To identify an optimal model, we tested several solvers: newton-cg, saga, lbfgs, and liblinear. During the testing procedure, the best performing solver was lbfgs, which is suitable for medium-sized datasets. This solver is also the default option provided by the library. For the maximum number of iterations, we selected 1000 as in the reference model, and the C factor was set to 0.1 [14]. Table 11 summarizes the performance comparison among the tested models.

Table 11. MNLR Models Performance Evaluation.

	MNLR Model	Default	[8]	[18]
Accuracy Training	67%	67%	65%	66%
Accuracy Testing	68%	68%	65%	67%
Precision	68%	68%	64%	67%
Recall	68%	68%	63%	68%
F1 Score	68%	68%	64%	67%

The most effective MNLR utilized the lbfgs solver with the maximum iteration parameter set to 300. This model was chosen over the default option due to the most efficient execution. Upon examining the confusion matrix in Figure 9, this model demonstrated higher accuracy in predicting the PG and C positions. Moreover, in the MNLR model, the top five features are TRB36, AST36, AST, DRB, and TRB. This may explain why the model achieved higher accuracy in predicting PG and C positions, while being less effective with the remaining three positions.

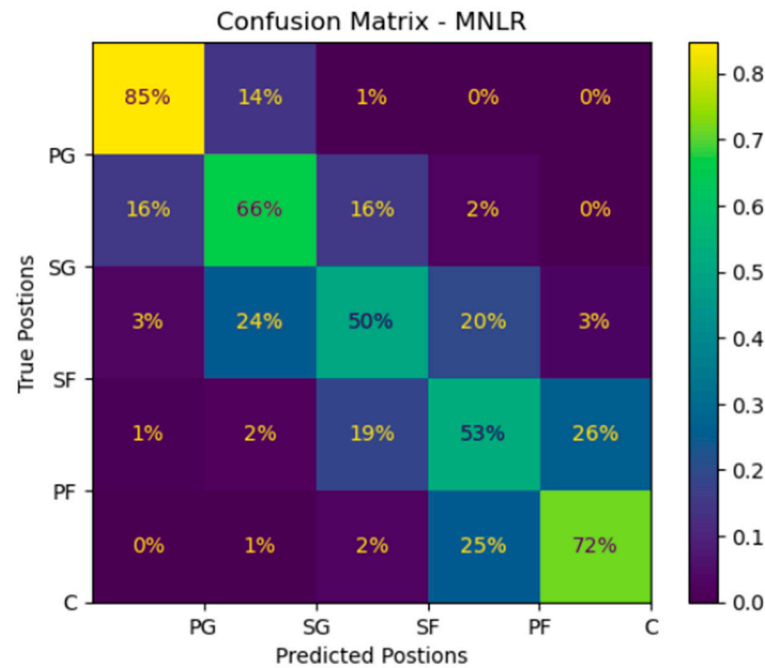


Figure 9. MNLR Confusion Matrix.

The next step involves creating a classifier based on the support vector machine (SVM) algorithm. Similarly with previous implementations, we maintained the 80/20 ratio for splitting the training and testing datasets. We set the kernel to linear, aligning with the approach of other researchers. As for scaling the dataset, we employed the standard scaler. Table 12 summarizes the parameters used in SVM implementations.

Table 12. SVM Models' Parameters Summary.

	[5]	[7]	[8]
Train-Test Split	80–20	80–20	70–30
random_state	Not defined	1	Not defined
Kernel	Linear	Linear	Linear
Library-Model	Mentions SKLEARN	SKLEARN.SVM SVC	Analysis using R.
Reported Accuracy	82%	71%	65%

Running the model using a linear kernel (Linear Kernel 1) and a random state equal to one resulted in an accuracy of 69%. This falls between the results of models used by [7] and [8] during their utilization. This difference could be due to variations in dataset size and the number of features used. On the other hand, as previously stated in the work by [6], the small size of the dataset and the different features (based on anthropometric measurements) again resulted in a higher difference. For further analysis and testing, different types of kernel and parameters have been used to utilize the SVM [13,17].

In addition to Linear Kernel 1, we have implemented an additional linear kernel with a regularization parameter (C) equal to ten, a random state equal to 42, and set the maximum iteration to 10,000. This Linear Kernel 2 underperformed compared to the others. Lastly, a polynomial kernel was introduced with the regularization parameters again set to ten. The degree hyperparameter was set to three to avoid overfitting. Values higher than three resulted in training accuracy checks of more than 90%, which is an indication of possible overfitting. The coef0 parameter, which controls the amount of influence that the model has on higher-degree terms, was set to one to allow for less

influence. The polynomial kernel SVM had similar performance metrics to Linear Kernel 1, but the running process took longer to execute. Table 13 demonstrates the comparison between the models.

Table 13. Performance Evaluation of SVM Classifiers.

	Linear Kernel 1	Linear Kernel 2	Polynomial Kernel
Accuracy Training	67%	50%	78%
Accuracy Testing	68%	47%	68%
Precision	68%	46%	68%
Recall	68%	46%	68%
F1 Score	68%	45%	68%

It is evident that the best performing model is Linear Kernel 1. Similarly, Figure 10 shows the confusion matrix with the percentage of proper player predictions and Figure 13 includes the corresponding Feature Importance plot. It seems that PG and C were easier to identify compared to other playing positions. Furthermore, the top five features for the SVM model were AST36, TRB36, FG%, BLK, and BLK36. This aligns with the importance of playmaking, rebounding, shooting accuracy, and defensive ability in determining player positions.

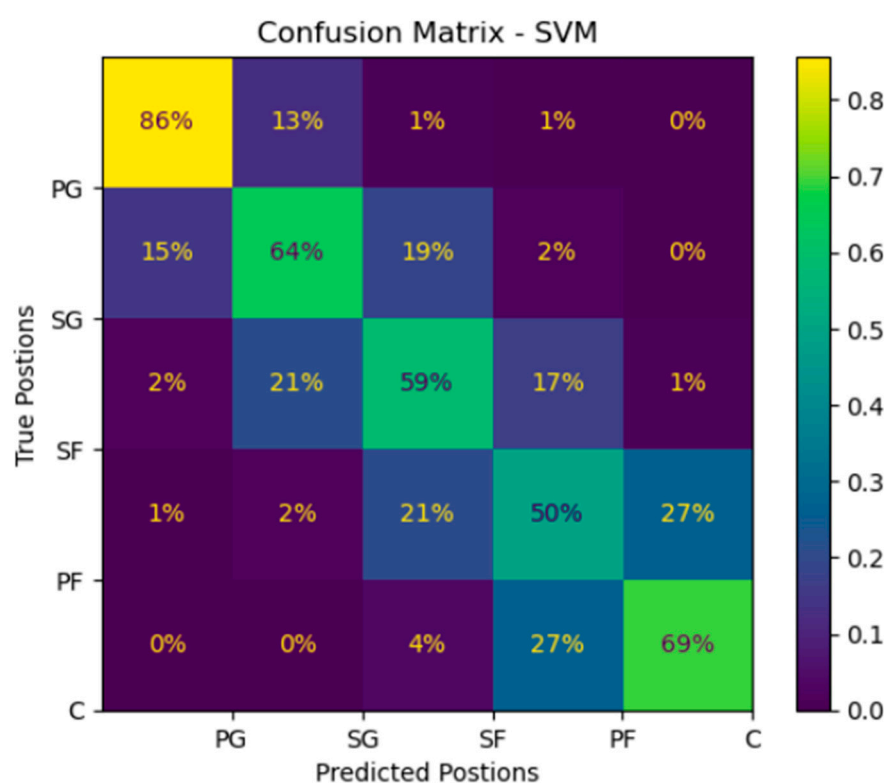


Figure 10. Confusion Matrix for SVM Classifier (Linear Kernel 1).

4.2. Proposed Models and Experimental Comparison

While the preceding subsection evaluates established classification models documented in the literature, this section ventures into models that have not been discussed before. In this phase, we aim to broaden the scope of our investigation by introducing two distinct classification methodologies, each offering unique insights into our research domain. By deviating from traditional models, we hope to discover alternative ways of creating a classification model.

4.2.1. Voting Classification Model

The first attempt is to create a voting classification model. This was found to be a very interesting approach as the voting classifier can be described as the wrapper of other algorithms. A Voting Classifier is an ensemble machine learning model that aggregates predictions from multiple other models. Instead of training separate models, it creates a single model that learns from the outputs of different models. There are two different voting strategies: hard voting and soft voting. In hard voting the class with the highest number of votes is selected. In soft voting, the average probability for each class is calculated, and the class with the highest average probability is predicted. Voting classifiers are typically employed to enhance model performance and achieve higher accuracy compared to individual models [23,24]. In our study, we implemented both soft voting and hard voting.

Examining the classifier performance, we observed that it achieved reasonable results. For the voting classifiers, we employed the four classification models implemented in Section 4.1. During the testing phase, both hard and soft voting options were executed. The soft voting option performed slightly faster and yielded slightly higher metrics as shown in Table 14. The confusion matrix in Figure 11 reveals a similar trend to previous classifiers, with the highest predicted position being PG followed by C. Analyzing feature importance (results shown in Figure 13), we found that TRB36, AST36, BLK36, STL/PF, and 3PA are the top five features in ascending order of importance.

Table 14. Voting Classifier Performance Summary.

	Hard Voting Model	Soft Voting Model
Accuracy Training	71%	73%
Accuracy Testing	68%	68%
Precision	68%	68%
Recall	68%	69%
F1 Score	68%	68%

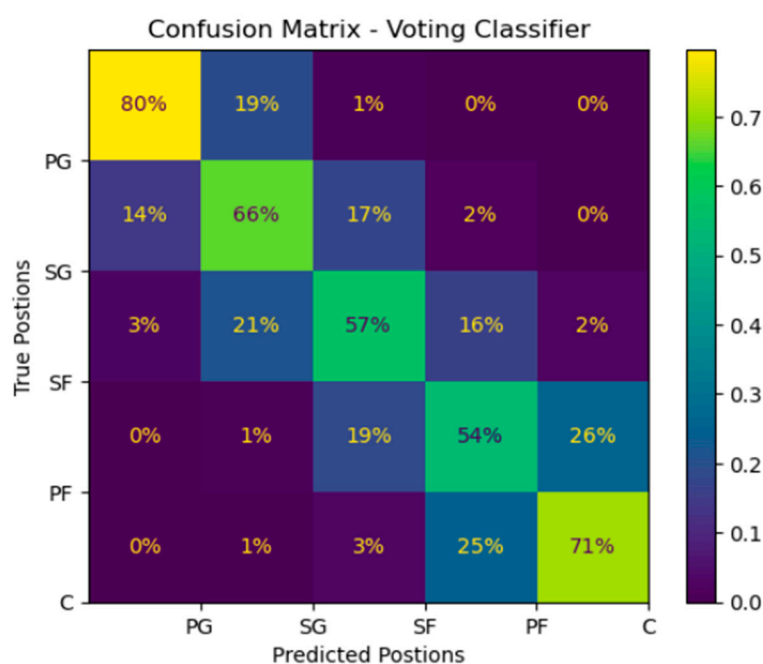


Figure 11. Soft Voting Classifier Confusion Matrix.

4.2.2. Gradient-Boosted Trees Algorithm

For this implementation, we use a different library, moving away from sklearn. Delving into the literature, we noticed a trend of utilizing the gradient-boosted trees algorithm for creating a classifier [10] and thus we use the XGBoost library [12]. To reduce overfitting, we fine-tuned parameters such as max_depth set to five and max_leaves to eight. The values were carefully chosen after numerous iterations, aiming to stabilize training accuracy below 85%. Furthermore, we set the number of estimators at 50. Table 15 provides a view of the performance metrics for our executed model.

Table 15. Performance Metrics of XGB Model.

	XGB Classifier
Accuracy Training	76%
Accuracy Testing	67%
Precision	67%
Recall	67%
F1 Score	67%

The XGB classifier achieved an accuracy level of 67%, aligning with the trends observed in previous implementations, but without experiencing a significant boost in results. According to the confusion matrix in Figure 12, PG and C emerged as the most accurately predicted groups by this model, with success rates of 82% for PG and 71% for C. Notably, the top five features identified by this model are AST36, TRB36, BLK36, 3PA, and STL/PF. Figure 13 shows the feature importance based on this implementation. For the identification of feature importance, we calculated the average value based on each individual classifier that was used in the voting process.

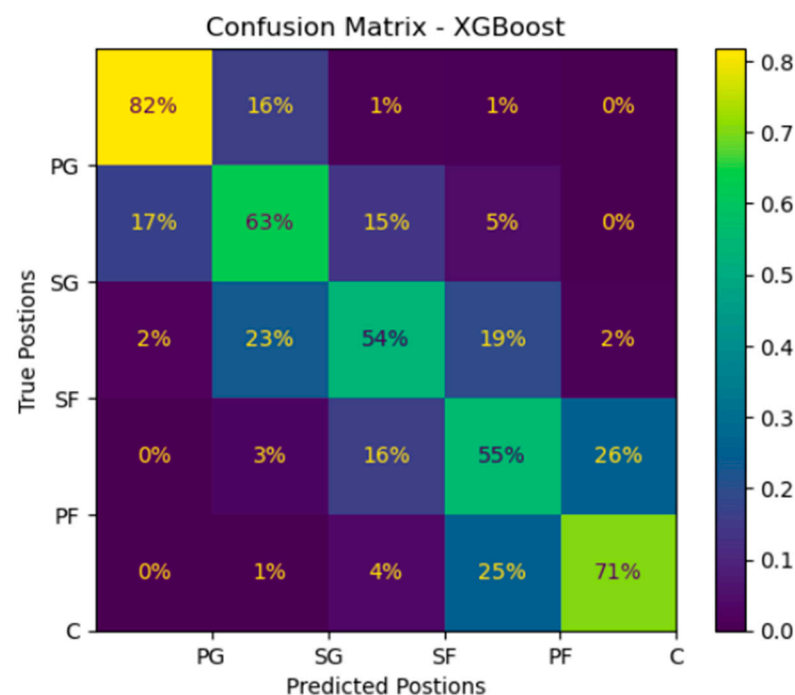


Figure 12. XGB Model Confusion Matrix.

4.3. Evaluation and Performance Comparison of Literature and Proposed Models

We performed a full evaluation of the performance of all models introduced in the previous sections, aiming to identify the optimal choice for our research objectives. Based

on the established classification models examined based on the literature and the methodologies explored and via testing and analysis, we examined each model's effectiveness in delivering accurate predictions within the scope of this paper.

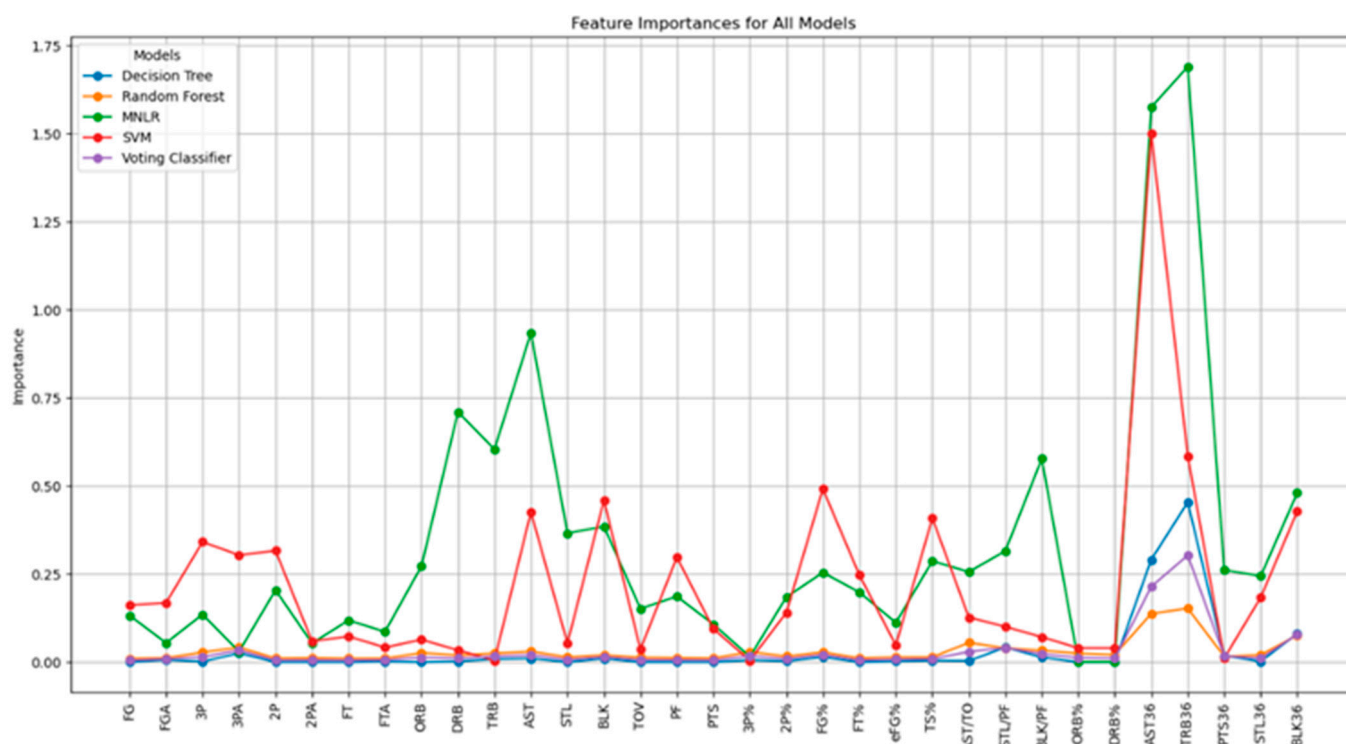


Figure 13. Feature Importance comparison for all classifiers.

Figure 13 presents a consolidated Feature Importance plot for all classifiers, in an attempt to illustrate the most important features for each model and determine which features are the most important overall. We noticed that TRB36, AST36, and BLKS36 are among the metrics that are important for all the models. We can identify metrics that are important for one model but not as significant for others. For example, AST is among the top three metrics for MNLR and SVM but not among the top features for the rest of the models. Figure 14 provides a comprehensive overview of performance metrics across all classifiers, encompassing the success rate of position prediction and the top five features identified for each model. The fact that accuracy testing for all models remains below 85%, suggests that overfitting has been avoided. Particularly, our evaluation reveals testing accuracy ranging from 60% to 68% among all classifiers. These results reveal interesting insights into the challenges of accurate labeling, potentially influenced by evolving gameplay dynamics and the different statistical contributions across different positions. Within our evaluation, we observed a consistent trend wherein predictions for PG and C positions showed higher accuracy compared to SG, SF, and PF positions. This discrepancy suggests player roles and positional flexibility, impacting prediction accuracy. Furthermore, Precision, Recall, and F1 Score metrics consistently range between 60% and 70% across all classifiers. This shows that our models are good at both identifying correct predictions and avoiding false predictions. Moreover, the integration of per-36 metrics emerges as an important feature in increasing position prediction. Remarkably, features such as total rebounds per-36 (TRB36), weighted assists per-36 (AST36), and block contributions per-36 (BLK36) stand out as key contributors to predictive accuracy, underscoring the significance of statistical metrics in refining classification models.

	DT	RF	MNLR	SVM	Soft Voting	XGB
Accuracy Training	68%	65%	67%	67%	73%	76%
Accuracy Testing	63%	64%	68%	68%	68%	67%
Precision	63%	63%	68%	68%	68%	67%
Recall	63%	64%	68%	68%	69%	67%
F1 Score	63%	63%	68%	68%	68%	67%
PG - Prediction	74%	81%	85%	86%	80%	82%
SG - Prediction	56%	66%	66%	64%	66%	63%
SF - Prediction	49%	53%	50%	59%	57%	54%
PF - Prediction	45%	54%	53%	50%	54%	55%
C - Prediction	67%	74%	72%	69%	71%	71%

Figure 14. Performance comparison of implemented models. Scaling color highlighted darker when metric value increases.

As shown in Figure 14, cells highlighted in a darker color indicate the best-performing model for each metric. Upon evaluation, it becomes evident that the soft voting classifier is the top performing model. This highlights the effectiveness of combining and integrating various conventional models into a single voting class, resulting in slight improvements over the others.

In order to effectively highlight the evolution of game statistics by position, Figure 15 shows the six shooting percentages for Centers (C) (Figure 15a) and Point Guards (PG) (Figure 15b) across four different seasons: 2004, 2010, 2016, and 2023. These six shooting percentages highlight various aspects of shooting and reveal how players have either improved or remained consistent in their respective positions over the years. Figure 15a shows that players in the C position have significantly improved their long-range shooting percentages (3-point) over time. This improvement reflects the evolution of team strategies and the increasing demand for better positional shooting. On the other hand, as shown in Figure 15b, PGs have shown relatively stable shooting percentages over the years. This consistency indicates that the role of PGs in terms of shooting percentages has not undergone significant changes.

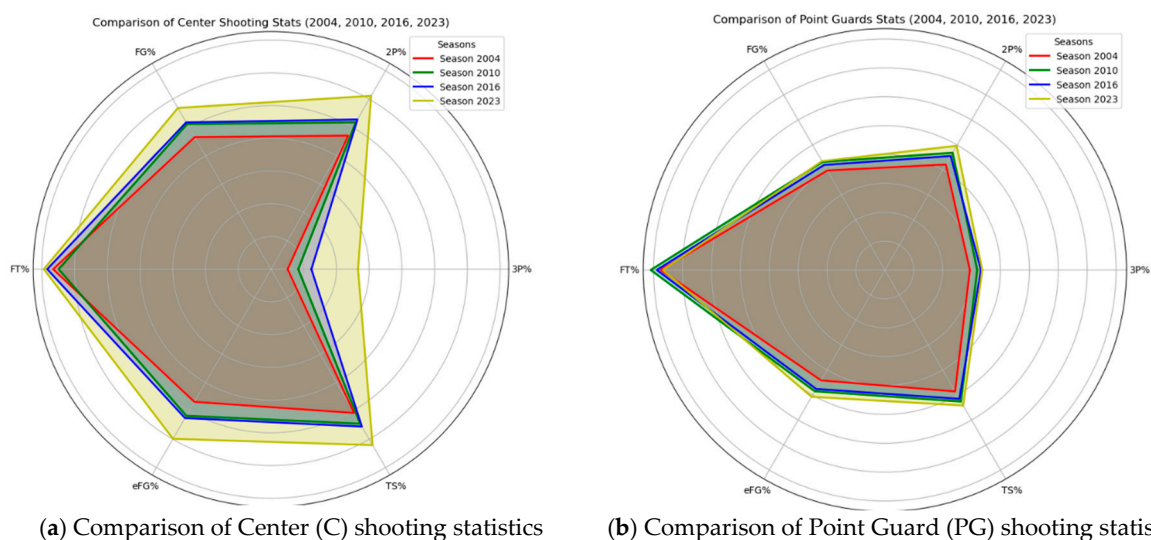


Figure 15. Shooting percentages for Centers (C) and Point Guards (PG) and comparison of the shooting statistics across four different seasons: 2004, 2010, 2016, and 2023.

5. Discussion and Future Directions

This study explores NBA player statistics using advanced analytics and machine learning methodologies. The primary objective is to investigate the correlation between player positions and game statistics, with a focus on creating a classification model

capable of determining player positions based on performance statistics. We first perform the process of data collection, cleaning, and modeling. During the preprocessing steps, the dataset was prepared for analysis, ensuring accuracy and reliability in the subsequent stages. Visualizations and insights derived from the dataset provide an understanding of players' statistical contribution relative to their positions. Several trends are identified in this phase, and the dataset sheds light on various game statistics. An interesting trend that we noticed is the increase in assist contributions by Centers.

The implementation and evaluation of classifiers are among the key contributions, where different algorithms were deployed and evaluated for their effectiveness. Decision Tree, Random Forest, Multinomial Linear Regression, Support Vector Machine, Voting Classifier, and Gradient Boosted Tree models were implemented and fine-tuned to achieve optimal performance. Through testing and evaluation, the Soft Voting Classifier became the top performer, demonstrating the effectiveness of ensemble learning techniques to improve classification accuracy.

Insights gained from the analysis outline the relationship between game statistics and player positions, highlighting the significance of key performance metrics in player classification. Weighted game statistics like per-36 statistics and key playing ratios such as assist-to-turnover have proven to be among the most important features for the classification process. The analysis shows that players contribute to several statistics despite their playing position. This can support the analysis of Muthu Alagappan [1], suggesting potential classification errors of players' positions. Utilizing more advanced statistics and clustering techniques can divide players into certain roles rather than defining a specific position.

The work in this paper represents a significant step towards leveraging data analytics and machine learning to gain insights into player performance in the NBA. By understanding the statistical aspect of the game and its relationship with player positions, coaches and teams can make more informed decisions, optimize team composition, and ultimately enhance on-court performance. Finally, this work can serve as the foundation for further advancements in basketball analytics. In this concluding section, we outline two potential expansions of this research and how these approaches can be further developed.

As previously discussed, this approach and modeled dataset can be used for a clustering model. This can be utilized to map the players into two different groups and provide a different approach to understanding the game and roles instead of positions similar to the works in [1,20] in order to relate players with what they are directly contributing to the game and their roles in both offensive and defensive plays. This could be beneficial, as specific players would be labeled, for example, as "*elite scorers*" and could play in different positions. Consider, for instance, current examples like Giannis Antetokounmpo, who is labeled as a PF, and Stephen Curry, who is labeled as a PG by position definition.

Lastly, a probabilistic model based on classification models can be designed. The primary challenge lies in creating a predictive framework capable of estimating the percentages of each position that a player is best suited for. Consider a player like LeBron James, where the model aims to predict the likelihood of him fitting into various positions based on his statistical profile. The model's output might resemble the following breakdown: PG: 60%, SG: 7%, SF: 20%, PF: 12%, and C: 1%. As mentioned in [18], this information could then be utilized to develop a metric quantifying a player's flexibility across positions. By weighing this metric, it can provide an indication of which team has more flexibility in switching players among positions.

Author Contributions: Conceptualization, A.T. and C.E.; methodology, C.E.; software, A.T.; validation, A.T.; formal analysis, A.T.; investigation, A.T.; resources, A.T.; data curation, A.T.; writing—original draft preparation, A.T.; writing—review and editing, C.E.; visualization, A.T.; supervision, C.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset used in this paper was downloaded in CSV format from Basketball-Reference <https://www.basketball-reference.com/> (accessed on 30 January 2025) [22].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Alagappan, M. From 5 to 13: Redefining the Positions in Basketball. In Proceedings of the 19th Annual MIT Sloan Sports Analytics Conference, Boston, MA, USA, 2–3 March 2012.
2. Morgulev, E.; Azar, O.H.; Lidor, R. Sports analytics and the big-data era. *Int. J. Data Sci. Anal.* **2018**, *5*, 213–222.
3. Sarlis, V.; Papageorgiou, G.; Tjortjis, C. Leveraging Sports Analytics and Association Rule Mining to Uncover Recovery and Economic Impacts in NBA Basketball. *Data* **2024**, *9*, 83.
4. Sarlis, V.; Tjortjis, C. Sports Analytics: Data Mining to Uncover NBA Player Position, Age, and Injury Impact on Performance and Economics. *Information* **2024**, *15*, 242.
5. Arpitha, T.; Sanjay, H.S.; Kumar, H.K.K.; Prithvi, B.S.; Rohan, A. Machine Learning Based Prediction Of The Best Suitable Playing Positions of The Players In The Game of Basketball. In Proceedings of the IEEE Mysore Sub Section International Conference (MysuruConj), Hassan, India, 24–25 October 2021.
6. Miao, C. Predicting NBA Positions Using Machine Learning Algorithms. *Nerd For Tech*, 24 May 2021.
7. Fischler, B. NBA Machine Learning Position Predictor. *Analytics Vidhya*, 3 February 2021.
8. Lefebure, L. Understanding Player Positions in the NBA. *Stanford University*, 12 December 2014.
9. Zixue, Z.; Bingyu, P. A Machine Learning Model to Predict Player's Positions based on Performance. In Proceedings of the icSPORTS 2021—9th International Conference on Sport Sciences Research and Technology Support, Valletta, Malta, 28–29 October 2021.
10. Frey, M.; Murina, E.; Rohrbach, J.; Walser, M.; Haas, P.; Dettling, M. Machine Learning for Position Detection in Football. In Proceedings of the 6th Swiss Conference on Data Science (SDS), Bern, Switzerland, 14 June 2019.
11. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res. JMLR* **2011**, *12*, 2825–2830.
12. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
13. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed.; O'Reilly Media Inc.: Sebastopol, CA, USA, 2022.
14. Grus, J. *Data Science from Scratch*, 2nd ed.; O'Reilly Media Inc.: Sebastopol, CA, USA, 2019.
15. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.
16. Zhao, X.; Yan, X.; Yu, A.; Van Hentenryck, P. Modeling Stated preference for mobility-on-demand transit: A comparison of Machine Learning and logit models. *arXiv* **2018**, arXiv:1811.01315.
17. Anguita, D.; Ghio, A.; Greco, N.; Oneto, L.; Ridella, S. Model selection for support vector machines: Advantages and disadvantages of the Machine Learning Theory. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010.
18. Welland, C. Predicting NBA Players Positions with Machine Learning. *PI.Exchange Sports Analytics*, 19 December 2023.
19. Bianchi, F. *Towards a New Meaning of Modern Basketball Players Positions*; University of Pavia—Faculty of Engineering, Pavia, Italy, 2016.
20. Patel, R. *Clustering Professional Basketball Players by Performance*; University of California, Los Angeles, Los Angeles, CA, USA, 2017.
21. Mazidi, A.; Golsorkhtabaramiri, M.; Etminan, N. Sport Result Prediction Using Classification Methods. *J. Appl. Dyn. Syst. Control* **2020**, *3*, 39–48.

22. Sports Reference LLC. Basketball Reference—NBA Player Statistics. Available online: <https://www.basketball-reference.com/> (accessed on 30 January 2025).
23. Bonaccorso, G. *Machine Learning Algorithms*, 2nd ed.; O'Reilly Media Inc.: Sebastopol, CA, USA, 2018.
24. Abdul Fattah, H.M.; Azharul Hasan, K.M.; Sunanda, D. A Voting Classifier for the Treatment of Employees' Mental Health Disorder. In Proceedings of the International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), Rajshahi, Bangladesh, 8–9 July 2021.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.