Report On:

**Programming Assignment II: Improving Accuracy in NBA Player Position Classification**

**Fall 2023**



Report By:

Nikhil Das Karavatt

1002085391

Contents

[Introduction 2](#_Toc150623498)

[Data Preprocessing 2](#_Toc150623499)

[Data Source 2](#_Toc150623500)

[Filtering Data 2](#_Toc150623501)

[Feature Engineering 2](#_Toc150623502)

[Irrelevant Column Removal 2](#_Toc150623503)

[Feature Selection 2](#_Toc150623504)

[Target Variable Mapping 2](#_Toc150623505)

[Machine Learning Model 2](#_Toc150623506)

[Hyperparameter Tuning 3](#_Toc150623507)

[Convergence Warnings 3](#_Toc150623508)

[Other Methods Explored 3](#_Toc150623509)

[Model Evaluation 3](#_Toc150623510)

[Holdout Validation 3](#_Toc150623511)

[Confusion Matrix 3](#_Toc150623512)

[Cross-Validation 3](#_Toc150623513)

[Results 3](#_Toc150623514)

[Conclusion 4](#_Toc150623515)

# Introduction

In this document, I will outline the method I employed to achieve better accuracy in classifying NBA player positions. The primary objective was to predict player positions (Point Guard, Shooting Guard, Small Forward, Power Forward, and Centre) based on relevant player statistics. I will discuss the data preprocessing steps, feature selection, the choice of machine learning model that led to improved accuracy, and briefly touch on the other methods I explored.

# Data Preprocessing

## Data Source

I obtained the dataset from a CSV file containing NBA player statistics for the 2021 season. The data source provided essential information such as player names, positions, minutes played (MP), and various performance metrics.

## Filtering Data

I initiated the process by filtering the data to include only those NBA players who played substantial minutes per game (MP). Specifically, I considered players who played at least 8 minutes per game, ensuring that I focus on relevant and active players. Rows containing players with less than 8 minutes played would unfairly skew the data. For instance, if a player plays only 2 minutes for a team and scores 2 points, his Points Per game would be 48 which is clearly not a fair representation.

## Feature Engineering

Created a new column 'BLK/TRB' representing the ratio of blocks (BLK) to total rebounds (TRB) for each player, which could be a meaningful statistic in basketball analysis.

## Irrelevant Column Removal

To streamline the dataset and remove noise, I performed a step of column removal. Columns that were considered irrelevant for our classification task were dropped. These columns included players' names ("Player"), team names ("Tm"), games (“S”), and games started (“GS”).

## Feature Selection

A crucial aspect of our approach was featuring selection. I aimed to discover the optimal combination of columns for classification. The objective was to identify the most informative set of features that would lead to the best classification accuracy. The following statistical columns were dropped: Effective Field Goal Percentage (eFG%), 3-Point Field Goal Percentage(3P%), Field Goal Attempts (FGA), Field Goal Percentage (FG%), Free Throw Percentage (FT%)

## Target Variable Mapping

An essential step in our project was mapping the target variable, which is the player's position ("Pos"), to numerical values. I used a predefined mapping to assign numerical labels to each player position. This mapping provided the model with the necessary labels for accurate classification.  
Manually encode y-labels made them more meaningful for the confusion matrix during evaluation.

# Machine Learning Model

I selected a Support Vector Classifier (SVC) as the machine learning model for this classification task. Several hyperparameters were tuned to optimize the model's performance. I chose a linear kernel function and set the regularization parameter (C) to 3. This choice was informed by empirical results and the characteristics of the dataset.

## Hyperparameter Tuning

The SVM is known for its adaptability and ability to handle complex datasets. To optimize the performance of the SVM model, I conducted hyperparameter tuning. Specifically, I focused on fine-tuning the "C" parameter, which controls the trade-off between maximizing the margin and minimizing the classification error. Different kernel functions were explored. I used grid search to find best combination of hyperparameters and that led me to choose a linear kernel function and set the regularization parameter (C) to 3. This choice was informed by empirical results and the characteristics of the dataset. Also kept max iteration parameter to -1 which means there isn’t any limit to it.

## Convergence Warnings

During the tuning process, I encountered convergence warnings. To address this issue, I ensured that the data was correctly scaled using a StandardScaler. Scaling the data played a crucial role in mitigating convergence problems and allowed the SVM model to perform optimally. Also didn’t keep any limit on max iteration.

## Other Methods Explored

In the quest for better accuracy, I experimented with alternative machine learning models, such as Random Forest, Decision Tree, Gradient Boosting, XGBoost k-Nearest Neighbors (k-NN), Adaboost, Naïve bayes and Logistic Regression. Each of these models was trained with their hyperparameters tuned using grid search and was evaluated on the same dataset using similar preprocessing steps. However, they yielded lower accuracy compared to the final Support Vector Classifier.

# Model Evaluation

## Holdout Validation

I split the data into training and testing sets using a 75-25% ratio. This approach ensured that the model was trained on a representative portion of the data while leaving unseen data for evaluation.

## Confusion Matrix

I used a confusion matrix to assess the model's performance on the test data. The matrix showed how well the model predicted each player's actual position. The results indicated a balanced classification performance across different positions.

## Cross-Validation

To further validate the model's performance and assess its generalization capability, I employed k-fold cross-validation (k=10). The cross-validation results helped provide a more robust estimate of the model's accuracy and confirmed its effectiveness.

# Results

The method outlined in this document, which primarily focused on the Support Vector Classifier, led to an improved accuracy in predicting NBA player positions. The classifier achieved an average accuracy of 57.19% across the 10 folds of cross-validation. The test accuracy reached 66.04%, and the training accuracy was 67.19%. These results demonstrate the model's capability to accurately classify players into their respective positions. Below is the full result of my model on NBA dataset:

Classification Method Used: SVC (C=3, kernel='linear')

Train Accuracy of SVM Model: 67.19%

Test Accuracy of SVM Model: 66.04%

Confusion matrix:

Predicted C PF PG SF SG All

True

C 18 0 0 2 0 20

PF 4 11 1 4 1 21

PG 0 0 17 0 4 21

SF 0 3 1 5 9 18

SG 0 3 1 3 19 26

All 22 17 20 14 33 106

Fold 1 Accuracy: 62.79%

Fold 2 Accuracy: 55.81%

Fold 3 Accuracy: 65.12%

Fold 4 Accuracy: 57.14%

Fold 5 Accuracy: 59.52%

Fold 6 Accuracy: 57.14%

Fold 7 Accuracy: 57.14%

Fold 8 Accuracy: 57.14%

Fold 9 Accuracy: 47.62%

Fold 10 Accuracy: 52.38%

Average accuracy across all folds: 57.18%

# Conclusion

In conclusion, by carefully selecting relevant features, transforming the target variable, scaling the data, and employing a Support Vector Classifier with optimized hyperparameters, I was able to achieve improved accuracy in predicting NBA player positions. The approach outlined in this document serves as a robust method for position classification in basketball analytics.

This document provides a comprehensive overview of the steps and considerations that contributed to the successful classification of NBA player positions. While other methods were explored, the accuracy achieved with the Support Vector Classifier indicates its potential utility for practical applications in basketball analytics and player assessment.