A Capstone Project Report on

NBA Player Positions

PREDICTION

Submitted by

B.MARIA MANOJE

Abstract

To better understand the meaning of positions in the NBA, I train several models to predict a player's position from his stats, such as average speed on the court and number of three point attempts per game. The NBA and ESPN use different position labels, so I compare the performance of my models in predicting these labels separately. Ultimately, I hope that this analysis can determine how well position labels describe NBA player roles and identify the defining characteristics of each position. NBA player positions according to their game statistics in regular season 2019-2020. In the experiment, there were totally 519 players used to be examples. Five phases were processed during the experiment based on five different desired positions. In the first phase, five positions were classified. In the second phase, five positions were classified. We used confusion matrices to calculate the clustering accuracies. The experimental results show that less numbers of the desired clusters get better classifier results.

Acknowledgements

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of this group project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

Further, we were fortunate to have Mr.Anbu Joel as my mentor. He has readily shared his immense knowledge in data analytics and guide me in a manner that the outcome resulted in enhancing our data skills.

I certify that the work done by me for conceptualizing and completing this project is original and authentic.

Date: August 02, 2022 Name:MariaManoje.B

Certificate of Completion

I hereby certify that the project titled “NBA Player Position Prediction” was undertaken and completed (July 2022)

Mentor: Mr. Anbu Joel

Date: August 2, 2022

Table Of Contents

CHAPTER 1: INTRODUCTION

Basketball is a unique sport in that there are no position specific rules or statistics. All players are traditionally measured by the number of rebounds, blocks, and steals they get on defense and the number of points they score, passes they make, and assists they get on offense. Therefore, position labels can be somewhat ambiguous: the NBA classifies players as either a Guard, Guard-Forward, Forward, Forward-Center, or Center, and ESPN classifies players as either a Point Guard, Shooting Guard, Small Forward, Power Forward, or Center. This has led some to do cluster analysis on NBA players to discover more and ”new” positions other than the traditional five [1]. The goal of this analysis is to determine how well existing position labels actually describe a player’s game. This is done by building models to predict position label from player stats. I consider two different position labeling schemes, namely those given by the NBA and ESPN, and compare model performance in predicting these labels separately. In addition, by analyzing these models, I can identify the features that are most predictive of position

1.1 Title & Objective of the study:

1.1.1 Primary objectives

This analysis looks at 519 players from the 2019/2020 NBA season. I use Position features for player performance previous seasons to predict

1.2 Position

1.2.1 Center

The center is usually the tallest player on the team. He should be able to post up offensively, receiving the the ball with his back to the basket and use pivot moves to hit a variety of short jumpers, hook shots, and dunks. He also must know how to find the open player in the paint and grab rebounds and blocks.

Average Height: 6’11.25″

Average Weight:257lbs

1.2.2 **Point Guard**

The point guard is usually the shortest player on the team. Should be the team's best passer and ball handler; not primarily a shooter. Traditional role is to push the ball up-court and start the offensive wheels turning. Should either take the ball to the basket or remain near the top of the key, ready to retreat on defense. Point guards are known to get the most assists and steals.

Average Height: 6'2″

Average Weight: 189 lbs

1.2.3. **Shooting Guard**

The shooting guard is usually a little bit taller than the point guard but shorter than a small forward. The shooting guard is considered as the teams best shooter generally. A good shooting guard comes off screens set by taller teammates prepared to shoot, pass, or drive to the basket. In addition, the shooting guard must also be versatile enough to handle some of the point guard's ball-handling duties.

Average Height: 6'5.25

Average Weight: 209lbs

1.2.4**. Small Forward**

The small forward is considered as the all-purpose player on offense. An strong and aggressive who is tall enough to mix it up inside but agile enough to handle the ball and shoot well. Must be able to score both from the perimeter and from inside.

Average Height: 6'7.75″

Average Weight: 225lbs

1.2.5. **Power Forward**

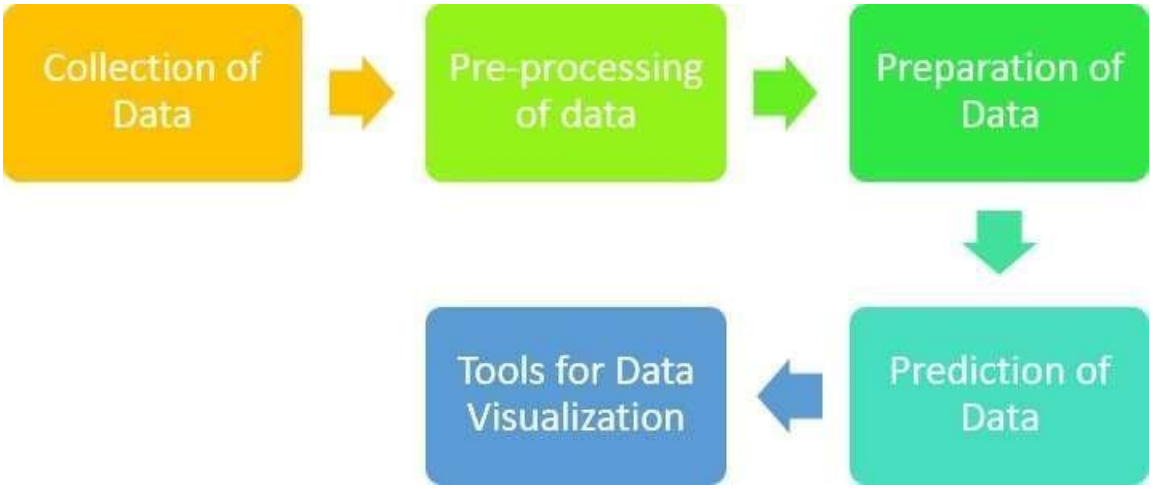
Finally, the power forward is a strong, tough re-bounder, but also athletic enough to move with some quickness around the lane on offense and defense. Expected to score when given the opportunity on the baseline, much like a center, but usually has a range of up to 15 feet all around the basket.

Average Height: 6’9.5″

Average Weight: 246lbs

CHAPTER 2: DATA PREPARATION AND UNDERSTANDING

2.1 Phase I – Data Extraction and Cleaning:



From the above figure 1 the steps used for the proposed system are collection of data, Pre Processing of data, Preparation of data, Prediction of data and Tools for data visualization. The steps are explained below briefly:

2.1.1 Collection of Data

The data that is feasible for analysis in NBA data set has been used and the prediction has been carried out for the same.

2.1.2 Pre processing of data

The pre processing of data involves 3 steps namely data cleaning, feature selection and data transformation. Each step is explained below: Data transformation comprises of two explanatory variables which can be transformed from binomial form into binary form to be much applicable for the chosen models.

The data cleaning step involves missing data imputation or handling. Some of the chosen algorithms cannot manage missing data that is why missing value can be transformed by median, mean or zero. However, the replacement of missing data by computed value statistically is a better choice. The used set of data involves missing values in certain numerical variables and two categorical variables. Before training of model, feature selection is one of the most essential factors that can influence the model's performance.

2.1.3 Preparation of data

The positions normally employed by NBA teams are the point guard, shooting guard, small forward, power forward, and the center. Despite the fact that the NBA has tried to move toward more position-less basketball, there exists clear distinctions between position based on player statistics. In this project I will attempt to classify players into one of the five main basketball positions based players statistics scraped from statscrunch. I make use of decision trees and random forest to do the prediction.. I calculated team PER the way as I described above because: 1) PER is a per-minute statistics, thus multiplying a player's PER by his minutes played can serve as an approximation of the total contribution he made towards his team over the entire season; 2) Some teams have made changes in their roster over the season through trade, due to injuries, or at coaching staff's discretion. However, the first 12 players on each team's roster are relatively stable, which indicates that those players are likely to be in the usual rotation of their teams' lineup. One huge benefit of calculating team PER the way I did is that the formula automatically puts more weight on players with higher PER because high-PER individuals are, by the way the metric is designed, more capable and are more likely to play a lot more than the players with lower PER. Even when an all-star player (usually with very high PER value) gets injured, the formula is not affected because he will play much fewer minutes and thus have much less weight when calculating team PER. Derrick Rose of Chicago Bulls, for instance, had multiple surgeries over last season and thus played very limited minutes. Therefore, even though Rose is a very high caliber player his contribution to Bull's team PER is rather small because of the limited time he played.

2.1.4 Prediction of data

Traditionally, the 5 basketball positions normally employed by NBA teams are the point guard, shooting guard, small forward, power forward, and the center. Despite the fact that the NBA has tried to move toward more position-less basketball, there exists clear distinctions between position based on player statistics. In this project I will attempt to classify players into one of the five main basketball positions based players statistics scraped from statscrunch. I make use of decision trees and random forest to do the prediction.

2.2 Phase II - Feature Engineering

The data was processed to convert it from its raw status into features to be used in machine learning algorithms. This process took the longest time due to the huge numbers of columns. The first idea was to aggregate values of columns per month (average, count, sum, max, min ...) for each numerical column per customer, and the count of distinct values for categorical columns.

Based on the data types and the values, following actions are defined to pre process/engineer the features for machine readability and further analysis:

Columns removed: RK and Playername (not relevent)

2.3 Exploratory Data Analysis

This study uses Kaggle website for data set in predicting and analyzing churn. Kaggle is a sit and community for hosting ML competitions. Rivalry ML can be a best way to practice and develop their skills as well as explain their abilities. Kaggle permits users to publish and find sets of data and describe models in a web-based data science surroundings, perform with other scientists of data and ML engineers and enter competition to resolve the barriers of data science. The pre processing steps used for data set are: 1) first the spaces are replaced with values of null in the column of total charges; 2) the values of null are reduced from the column of total charges which comprises 15 percent missing data; 3) then the data is converted to the type of float; 4) after than no internet service is replaced to no for the following columns:Device Protection, Streaming TV, Online Security, Tech Support, Streaming Movies and Online Backup; 5) the values for SenioCitizen is replaced with 0 as No and 1 as Yes; 6) Then the categorical column is made into Tenure; 7) After than the churn and non - churn customers are separated; 8) Finally the numerical and categorical columns are separated

CHAPTER 3: FITTING MODELS TO DATA

3.1 K-Nearest Neighbor

According to Keramatia et al (2014) K-Nearest Neighbor is one of the most useful and applicable non parametric algorithms of learning. K-Nearest Neighbor is also referred as lazy algorithm that is entire data of training is used at the phase of testing. There is no phase of training and entire points of data are used directly in the testing phase so these entire points required to be employed when it must be tested. K-Nearest Neighbor utilizes the distance between records so as to utilize it for classification. In order to estimate the distance between points K-Nearest Neighbor considers that these points are multidimensional or scalar vectors in feature space. All points of data are vectors of feature space and the label will refer their classes. The easiest case is when the class labels are binary but still it is useful on arbitrary class numbers. In K-Nearest Neighbor one parameter requires to be tuned. K is the number of neighbors/instances that are regarded for instance labeling to some class. The cross validations were carried out using different values of k. K-Nearest Neighbor does not attempt to build an internal structure and computations are not carried out until the time of classification. K-Nearest Neighbor stores only examples of the training information in feature space and the class of an example is decided based on most of the votes from its neighbors. Instance is labelled with class which is much similar among its neighbors. K-Nearest Neighbor decides neighbors based on hamming for categorical variables and distance using Manhattan, Murkowski and Euclidian measures of distance for continuous variables. Estimated distances are employed to recognize training instances set that are nearest to the new point and allot label from these. Despite its simplicity K-nearest neighbor have been used to different kinds of application. For churn K-nearest neighbor is used to examine if a customer churns or not based on features proximity to consumers in every classes .

In [31]:
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()
knn.fit(X,y)
predict4= cross_val_predict(estimator = knn, X = X, y = y, cv = 10)
print("Classification Report: \n",classification_report(y, predict4))
```

```
Classification Report:
              precision    recall  f1-score   support

           C       0.39      0.37      0.38       104
          PF       0.26      0.33      0.29       109
          PG       0.30      0.24      0.26       110
          SF       0.21      0.23      0.22       122
          SG       0.32      0.30      0.31       145

    accuracy                           0.29       590
   macro avg       0.30      0.29      0.29       590
weighted avg       0.29      0.29      0.29       590
```

3.2 Logistic Regression

Logistic regression is the proper model of regression analysis to utilize when the dependent variable is binary. Logistic regression is a predictive examination used to describe the relation between an independent variable set and dependent binary variable. For churn of customer logistic regression has been used to estimate the probability of churn as a function of customers characters or variables set (Sahu et al, 2018). According to Hassouna et al (2016) Logistic regression is also used to find the customer churn occurrence probability. Logistic regression is based on a mathematically oriented method to examine the impact of variables on others. Prediction is made by comprising a group of equations linking values of input with the output field

LogisticRegression

In [28]:
```
model= LogisticRegression(random_state=0).fit(X,y)
predict1 = cross_val_predict(estimator = model, X = X, y =y, cv = 10)
print("Classification Report: \n",classification_report(y, predict1))
```

```
Classification Report:
              precision    recall  f1-score   support

           C       0.60      0.68      0.64       104
          PF       0.40      0.31      0.35       109
          PG       0.67      0.63      0.65       110
          SF       0.38      0.34      0.36       122
          SG       0.50      0.61      0.55       145

    accuracy                           0.52       590
   macro avg       0.51      0.51      0.51       590
weighted avg       0.51      0.52      0.51       590
```

3.3 Random Forest

We applied Random Forest on the Training data set to validate if any further improvement of the model can be performed post the linear regression. Below were the parameters which were applied for Random Forest

For the Random Forest model Randomized Search CV is used to optimize for several hyper parameters including n_estimators, max_features, max_depth, criterion and bootstrap .Random Forest algorithm was also trained, we optimized the number of trees hyper parameter. We experimented with building the model by changing the values of this parameter every time in 100, 200, 300, 400 and 500 trees. Te best results show that the best number of trees was 200 trees. Increasing the number of trees after 200 will not give a signifcant increase in the performance. GBM algorithm was trained and tested on the same data, we optimized the number of trees hyper-parameter with values up to 500 trees. Te best value after the experiment was also 200 trees. GBM gave better results than RF and DT.

RandomForestClassifier

In [30]:
```
from sklearn.ensemble import RandomForestClassifier

randomforest = RandomForestClassifier(random_state = 0)
randomforest.fit(X, y)
predict3= cross_val_predict(estimator = randomforest, X = X, y = y, cv = 10)
print("Classification Report: \n",classification_report(y, predict3))
```

```
Classification Report:
              precision    recall  f1-score   support

           C       0.66      0.61      0.63       104
          PF       0.37      0.33      0.35       109
          PG       0.62      0.56      0.59       110
          SF       0.37      0.36      0.36       122
          SG       0.42      0.51      0.46       145

    accuracy                           0.47       590
   macro avg       0.49      0.47      0.48       590
weighted avg       0.48      0.47      0.47       590
```

3.4 GradientBoostingClassifier

Machine learning algorithms require more than just fitting models and making predictions to improve accuracy. Most winning models in the industry or in competitions have been using Ensemble Techniques or Feature Engineering to perform better.Ensemble techniques in particular have gained popularity because of their ease of use compared to Feature Engineering. There are multiple ensemble methods that have proven to increase accuracy when used with advanced machine learning algorithms. One such method is **Gradient Boosting**. While Gradient Boosting is often discussed as if it were a black box, in this article we'll unravel the secrets of Gradient Boosting step by step, intuitively and extensively, so you can really understand how it works.

GradientBoostingClassifier

```
In [34]: gb = GradientBoostingClassifier()
         gb.fit(X,y)
         predict7 = cross_val_predict(estimator = gb, X = X, y = y, cv = 10)
         print("Classification Report: \n",classification_report(y, predict7))
```

```
Classification Report:
              precision    recall  f1-score   support

           C       0.62      0.60      0.61       104
          PF       0.36      0.34      0.35       109
          PG       0.64      0.65      0.65       110
          SF       0.30      0.28      0.29       122
          SG       0.43      0.48      0.45       145

    accuracy                           0.46       590
   macro avg       0.47      0.47      0.47       590
weighted avg       0.46      0.46      0.46       590
```

3.5 Decision Tree – Classification

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

BEST MODEL

In [37]:
```
from sklearn.tree import DecisionTreeClassifier as dtc

decisiontree = dtc(random_state=0)
decisiontree.fit(X,y)

predicted = cross_val_predict(estimator = decisiontree, X = X, y = y, cv = 10)
print("Classification Report: \n",classification_report(y, predicted))
```

```
Classification Report:
              precision    recall  f1-score   support

           C       0.50      0.48      0.49       104
          PF       0.31      0.36      0.33       109
          PG       0.53      0.47      0.50       110
          SF       0.28      0.29      0.29       122
          SG       0.38      0.38      0.38       145

    accuracy                           0.39       590
   macro avg       0.40      0.40      0.40       590
weighted avg       0.40      0.39      0.39       590
```

CHAPTER 4: CONCLUSION

4.1 CONLUSION

I expected that I would not be able to predict position with very high accuracy. In theory, basketball positions are well-defined — Point Guards possess and pass the ball while Shooting Guards take shots — but in reality, the lines between positions are blurred. The fact that I was able to achieve similar accuracy with two different position labels (albeit with substantial overlap) reinforces this notion. These results suggest that doing a cluster analysis to ”redefine” positions in basketball is justified, such as that done by Lutz [1]. However, this analysis does provide a lot of insight into what features are characteristic of existing positions. By interpreting the coefficients of the Multinomial Logistic Regression model for example, we were able to see which features distinguished different positions. This paper details only a small portion of the information about positions that can be inferred from this analysis.

4.2Future

Because the Player Tracking data exists for only one full season, I was limited by a small sample size. As more data becomes available, I will be able to extend my analysis further. I also need to address problems of collinearity in the dataset. Some features, such as number of possessions and time of possession, show some correlation, and this causes these features to have unstable coefficient estimates in Multinomial Logistic Regression. Similarly, I would like to use a more elaborate feature selection strategy. Finally, I would like to try Quadratic Discriminant Analysis because this model does not have the potentially problematic assumption of LDA described previously

4.3 **Discussion**

This analysis can also be used to better understand the existing position labels. In Table 4, we see that there are several Power Forwards classified as Centers, two of whom are Pau Gasol and Tiago Splitter. What separates these players from other Power Forwards? By analyzing the Multinomial Logistic Regression model coefficients, we can answer this question. Both Gasol and Splitter are distinguished by their above average rebounding ability, and as a result, they both have large positive values for rebounding.OREB CHANCE and rebounding.DREB CHANCE. Conversely, the Power Forward coefficients for these two features are very negative in the model that uses Center as the referent group (the group for which the coefficients are defined to be zero). Referring to the equations for Multinomial Logistic Regression detailed previously, this means that positive rebounding values (above average), boost a player's probability of being a Center relative to being a Power Forward, and vice versa. This partly explains why both Gasol and Splitter were classified as Centers.

CHAPTER 6: REFERENCES

[1] References D. Lutz, ”A Cluster Analysis of NBA Players,” in MIT Sloan Sports Analytics Conf., Boston, MA, 2012

[2] NBA. Web. 7 Nov. 2014. http://stats.nba.com/

[3] ESPN NBA. Web. 25 Nov. 2014. http://espn.go.com/nba/players/

[4] C. Chang and C. Lin. ”LIBSVM: A Library for Support Vector Machines,” National Taiwan University, Taipei, Taiwan, 2013.

[5] Jackson, B., Dimmock, J. A., Gucciardi, D. F., and Grove, J. R. 2011.

[6] “Personality Traits and Relationship Perceptions in Coach–athlete Dyads: Do Opposites Really Attract?,” Psychology of Sport and Exercise (12:3), pp. 222–230. Morgan, W. P. 1980.

[7] “The Trait Psychology Controversy,” Research Quarterly for Exercise and Sport (51:1), pp. 50–76. Pennebaker, J. W. 201