NBA Player Classification

Eric Sun, Kevin Andrews, Arnav Joshi

Abstract

Instead of conventionally categorizing NBA players by position, we aim to categorize and cluster NBA players by their style of play and the role they serve on their team using NBA statistics that measure these contributions. We implement several classification methods in an attempt to classify players as belonging to a certain cluster.

1 Introduction

The NBA is entering its 75th season this year, a remarkable milestone for any sport. The game has seen legend after legend dominate their respective era, and underdogs come out of nowhere to win it all. The way basketball is played undeniably changed throughout the years — from valuing midrange and inside shots and playing within one's position to playing fast-paced positionless basketball while shooting threes at a high volume.

1.1 The Problem

In today's basketball, we can see players playing outside their position's traditional role. Draymond Green, who plays at the power forward position, plays a more facilitating and playmaking role, with the Golden State Warriors. The same can be said for the reigning MVP, Nikola Jokic, who plays the center position for the Denver Nuggets. Players can't be grouped according to their position anymore, hence the usefulness of classifying them by their individual stats and the role they play on their respective teams.

1.2 Relevance and Application

This could be a useful tool for scouting for future games. Teams can look up a player of an opposing team and quickly determine what they're up against and do whatever is necessary to prepare. Another aspect of scouting this could be really useful for is scouting young talent coming into the league. By measuring the young players' stats in college (or another league they may have played in), they could combine our classification model with a hypothetical prediction algorithm to determine what *kind* of player they could develop into. This could be extremely useful as teams could then more confidently stake their future on a young prospect.

2 Technical Approach

The entirety of the project is written in python and uses primarily the Scikit-Learn library. Using the Scikit-Learn library, we utilized the K-Means clustering algorithm to determine the player clusters. After this

clustering, three different classification/regression techniques were implemented in order to determine player classes: support vector machine classification, random forest classification, and logistic regression. Then, Scikit-Learn is used to tune optimal hyperparameters for these classifiers.

2.1 K-Means Clustering

K-means clustering is an algorithm that looks for a fixed amount, k, of clusters with equal variance in a set of data. For the Scikit-Learn algorithm, it does so by minimizing a criterion called inertia which refers to the within-cluster-sum-of-squares [2]. The means of the samples in each cluster are commonly called the centroids, and the algorithm attempts to choose centroids that minimize the criterion with the following equation:

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2)$$

[2].

2.2 Support Vector Machine Classification

For support vector machine classification, the implementation used was the default provided by the SVC section of Scikit-Learn's SVM library. This section accounts for the "C-Support Vector Classification" algorithms and runs an implementation based on an older library, LIBSVM. By "C-Support", this library means that this model is running soft-margin SVM where C is the hyperparameter controlling the influence of the misclassification of points into their classes (allowance for misclassification of points so that a margin can be reached if points are misclassified in training data) [8]. A standard formula for the cost function that this model is likely using is as follows:

$$J = \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) > 1 - \xi_i \quad .$$

2.3 Random Forest Classification

Scikit-Learn's random forest classification is an averaging algorithm based on randomized decision trees [5]. It is an estimator that fits multiple decision tree classifiers on sub-samples of the data and then averages them [5]. This improves accuracy and attempts to prevent over-fitting.

2.4 Logistic Regression

Scikit-Learn implements a relatively standard algorithm for logistic regression for classification. In this classifier, a logistic function is implemented to separate classes. Scikit-Learn allows for multinomial logistic regression which is what makes separating all the classes possible [3]. A basic cost function used by Scikit-Learn to classify is as follows:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

[3].

2.5 Hyperparameter Tuning

Scikit-Learn's GridSearchCV and KFold libraries are used to tune the hyperparameters for the project. KFold is used to separate the dataset into *k* folds, and the folds are used as validation sets while tuning [7]. GridSearchCV is used to search over parameter values with an estimator. The parameters of the estimator are optimized by cross-validation during the grid search in this library [6]. GridSearchCV exhaustively searches all parameter combinations, meaning longer runtime but better results [6].

3 Experimental Results

Everything for the project was done in Python. We specifically used Scikit-Learn, NumPy, and panda to implement the K-Means clustering, logistic regression, support vector machine, and random forest classification.

3.1 Dataset

The data we used was from *NBAStuffer*, who has compiled season stats for all NBA players dating back to the 2008-2009 season. For this project, we used data from the 2020-2021 regular season. We didn't include last season's playoffs as not every player played, and we didn't use the current season's dataset as it is still ongoing. This dataset contains 484 players and the following stats for each player: Games Played (GP), Minutes Per Game (MPG), Minutes Percentage (MIN%), Usage Rate (USG%), Turnover Rate (TO%), Free Throws Attempted (FTA), Free Throw Percentage (FT%), 2-Pointers Attempted (2PA), 2-Pointer Percentage (2P%), 3-Pointers Attempted (3PA), 3-Pointers Percentage (3P%), Effective Field Goal Percentage (eFG%), True Shooting Percentage (TS%), Points Per Game (PPG), Rebounds Per Game (RPG), Total Rebound Percentage (TRB%), Assists Per Game (APG), Assist Percentage (AST%), Steals Per Game (SPG), Blocks Per Game (BPG), Turnovers Per Game (TOPG), Versatility Index (VI), Offensive Rating (ORTG), and Defensive Rating (DRTG) [4].

3.2 Researching on Number of Clusters to Use

One problem we had earlier on was figuring out how many clusters to use to group the players. As we weren't required to do clustering in good detail, we looked at what others have done on the clustering of the NBA players [1]. With this information and through trial and error on our own, we settled on doing 8

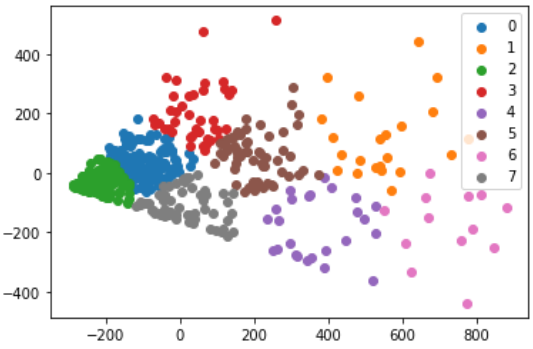
clusters. We felt this amount produced the most accurate clusters based on the player's style of play and the role they play on their teams.

3.3 Implementation - Clustering

The clustering via KNN model is done on python (python notebook) using Scikit-learn library. The dataset is in the format of csv and is imported via dataframe (modules in panda). After importing the dataset, Nan values are dropped, and only numeric values are kept. Using this dataset, KNN clustering is done using random_state as 40. After clustering, PCA is done to reduce the dimension of data samples from 24 to 2, in order to plot and analyze.

3.4 Result - Clustering

We ended up with 8 total clusters — we felt this number produced the most accurate clusters based on the type of player. Here were the clusters we ended up with:



CLUSTER (Description)	NOTABLE PLAYERS	AVERAGE STATS
0: Versatile Big Men - These players score mostly on the inside and get a lot of rebounds (also has most blocks per game). The high VI and ORTG shows they're more involved in the offensive than people think	Andre Drummond, Draymond Green, Serge Ibaka, Steven Adams	{'GP': 53.775862068965516, 'MPG': 22.334482758620684, 'MIN%': 46.524137931034474, 'USG%': 17.839655172413792, 'TO%': 14.377586206896547, 'FTA': 102.43103448275862, 'FT%': 0.7055862068965518, '2PA': 284.7931034482759, '2P%': 0.5912586206896552, '3PA': 49.74137931034826, '3P%': 0.24315517241379309, 'eFG%': 0.5840172413793102, 'TS%': 0.6082931034482759, 'PPG': 8.882758620689659, 'RPG': 5.894827586206898, 'TRB%': 14.993103448275864, 'APG': 2.003448275862069, 'AST%': 12.787931034482757, 'SPG': 0.7244827586206894, 'BPG': 0.79448275862069, 'TOPG': 1.2317241379310344, 'VI': 8.017241379310347, 'ORTG': 120.6603448275862, 'DRTG': 107.53275862068966}
1: High Usage, Offensive Players - this cluster features one of the highest offensive ratings, versatility index (which measures a player's ability to produce in more than one statistic), and AST%. They can score and aid the offense in a variety of ways.	Anthony Edwards, Donovan Mitchel, Damian Lillard, Luka Doncic, Stephen Curry	{'GP': 63.40909090909091, 'MPG': 33.53181818181818, 'MIN%': 69.85000000000001, 'USG%': 28.472727272727273, 'TO%': 11.090909090909092, 'FTA': 278.86363636363636, 'FT%': 0.8405454545454547, '2PA': 652.2272727272727, '2P%': 0.5138181818181818, '3PA': 465.31818181818183, '3P%': 0.3848181818181818, 'eFG%': 0.5394545454545455, 'TS%': 0.5803636363636365, 'PPG': 22.999999999999996, 'RPG': 5.35, 'TRB%': 8.631818181818183, 'APG': 4.363636363636363, 'AST%': 21.086363636363636, 'SPG': 0.9877272727272726, 'BPG': 0.4727272727272728, 'TOPG': 2.496363636363636, 'VI': 9.477272727272727, 'ORTG': 112.5227272727273, 'DRTG': 108.6090909090909}

2: Defensive Players - this cluster features the highest defensive ratings — although they don't have the highest BPG or SPG, their defensive presence is felt. These players don't score all that much.	Alex Caruso, Al Horford, Andre Iguodala, Myles Turner, Patrick Beverley	{'GP': 45.99047619047619, 'MPG': 21.476190476190474, 'MIN%': 44.740952380952386, 'USG%': 17.657142857142862, 'TO%': 12.300000000000006, 'FTA': 55.36190476190476, 'FT%': 0.7844571428571425, '2PA': 141.23809523809524, '2P%': 0.5197047619047618, '3PA': 146.22857142857143, '3P%': 0.3572571428571428, 'eFG%': 0.5301142857142856, 'TS%': 0.5587428571428571, 'PPG': 8.257142857142862, 'RPG': 3.3752380952380956, 'TRB%': 8.700952380952382, 'APG': 1.9685714285714286, 'AST%': 13.200000000000003, 'SPG': 0.6744761904761905, 'BPG': 0.3925714285714285, 'TOPG': 1.044190476190476, 'VI': 6.854285714285715, 'ORTG': 110.68095238095238, 'DRTG': 110.93047619047621}
3: Pure Scorers - These are players whose sole role is to score; what mostly differentiates this cluster from cluster 1 is the lower VI and AST%, meaning their role is limited to just scoring	Caris Levert, James Harden, Kevin Durant, Michael Porter Jr., Shai Gilgeous-Alexander, LeBron James	{'GP': 57.05172413793103, 'MPG': 29.193103448275863, 'MIN%': 60.82241379310345, 'USG%': 22.262068965517244, 'TO%': 11.25344827586207, 'FTA': 149.27586206896552, 'FT%': 0.8059999999999997, '2PA': 383.3448275862069, '2P%': 0.5148793103448276, '3PA': 282.5344827586207, '3P%': 0.3729655172413793, 'eFG%': 0.5349310344827587, 'TS%': 0.5691724137931033, 'PPG': 15.234482758620699, 'RPG': 4.562068965517242, 'TRB%': 8.575862068965517, 'APG': 3.4051724137931028, 'AST%': 17.410344827586204, 'SPG': 0.8974137931034485, 'BPG': 0.5270689655172414, 'TOPG': 1.7286206896551724, 'VI': 7.989655172413792, 'ORTG': 111.70172413793107, 'DRTG': 109.28448275862068}
4:Midrange Scorers, Facilitators - This features players who score mostly in the midrange/in the paint - high 2PA and percentage and low 3PA and 3P% and the highest BPG of any cluster. These players have the same usage rate but a higher VI, which means they play a larger role than scoring.	Anthony Davis, Bam Adebayo, Chris Paul, Kawhi Leonard, Pascal Siakam	{'GP': 61.07692307692308, 'MPG': 30.473076923076917, 'MIN%': 63.480769230769223, 'USG%': 22.780769230769224, 'TO%': 11.557692307692308, 'FTA': 224.84615384615384, 'FT%': 0.7668076923076925, '2PA': 616.7692307692307, '2P%': 0.5607692307692308, '3PA': 127.46153846153847, '3P%': 0.2761153846153846, 'eFG%': 0.5564230769230769, 'TS%': 0.5916923076923077, 'PPG': 16.788461538461546, 'RPG': 7.919230769230769, 'TRB%': 14.253846153846155, 'APG': 3.7153846153846153, 'AST%': 18.711538461538464, 'SPG': 0.9838461538461539, 'BPG': 0.7634615384615386, 'TOPG': 1.8753846153846148, 'VI': 9.542307692307691, 'ORTG': 117.51923076923079, 'DRTG': 105.05769230769229}
5: Low Usage Players - these players don't play too many minutes, usually a role player (someone who plays a supporting role for the team off the bench in the few minutes they play)	Austin Rivers, Dennis Smith Jr., Iman Shumpert, Meyers Leonard	{'GP': 25.165562913907284, 'MPG': 14.495364238410582, 'MIN%': 30.189403973509936, 'USG%': 17.501324503311253, 'TO%': 13.524503311125828, 'FTA': 24.172185430463575, 'FT%': 0.7583576158940397, '2PA': 57.788079470198674, '2P%': 0.5328079470198673, '3PA': 37.90728476821192, '3P%': 0.29833774834437093, 'eFG%': 0.5350596026490064, 'TS%': 0.5677947019867546, 'PPG': 5.273509933774832, 'RPG': 2.706622516556291, 'TRB%': 10.796688741721857, 'APG': 1.1529801324503332, 'AST%': 11.340397350993376, 'SPG': 0.4535761589403976, 'BPG': 0.3451655629139074, 'TOPG': 0.690662251655629, 'VI': 6.731125827814572, 'ORTG': 113.90596026490068, 'DRTG': 112.04701986754964}
6: Sharpshooters, 3-and-D players - this cluster features the highest 3PA and 3 point percentage; the low versatility index also indicates that the role of these players is to just shoot.	Bogdan Bogdanovic, Buddy Hield, Jae Crowder, Marcus Smart, Joe Harris	{'GP': 59.136363636363636, 'MPG': 27.015909090909076, 'MIN%': 56.27954545454547, 'USG%': 17.675000000000004, 'TO%': 9.806818181818183, 'FTA': 83.18181818181819, 'FT%': 0.8304545454545454, '2PA': 187.93181818181818, '2P%': 0.4981818181818181, '3PA': 345.04545454545456, '3P%': 0.3980681818181818, 'eFG%': 0.5615227272727273, 'TS%': 0.5867954545454547, 'PPG': 11.622727272727275, 'RPG': 3.6772727272727277, 'TRB%': 7.365909090909091, 'APG': 2.3295454545454455, 'AST%': 11.947727272727276, 'SPG': 0.8075000000000002, 'BPG': 0.31704545454545446, 'TOPG': 1.0911363636363633, 'VI': 6.515909090909091, 'ORTG': 114.99545454545455, 'DRTG': 108.76363636363635}

7: Engine of the Team - highest usage rate, high offensive rating,just to shoot and versatility index, meaning they do a bit of everything for the team (scoring, rebounding, and assisting). You usually won't find two players on the same team in this cluster.	Bradley Beal, Trae Young, Giannis Antetokoumpo, Ja Morant, Nikola Jokic, Russell Westbrook	{'GP': 62.53846153846154, 'MPG': 34.30769230769231, 'MIN%': 71.44615384615385, 'USG%': 30.815384615384623, 'TO%': 13.038461538461537, 'FTA': 454.0769230769231, 'FT%': 0.797, '2PA': 878.0, '2P%': 0.5372307692307692, '3PA': 257.53846153846155, '3P%': 0.33638461538461534, 'eFG%': 0.5346923076923077, 'TS%': 0.5904615384615386, 'PPG': 25.284615384615385, 'RPG': 6.830769230769231, 'TRB%': 10.861538461538462, 'APG': 6.338461538461539, 'AST%': 29.930769230769233, 'SPG': 1.0623076923076924, 'BPG': 0.4884615384615384, 'TOPG': 3.216153846153846, 'VI': 11.484615384615385, 'ORTG': 115.58461538461539, 'DRTG': 107.66153846153846}

3.5 Implementation - Classifying

For the classification section, more things would be done as this is a main section of our report. We will first determine which model is the best to use to train the data. After that, the best model will be done hyperparameter tuning to find the best parameter to fit the model. In the end, the model will be evaluated using the test set.

3.5.1 Choosing the Correct Model

Before all procedures, the resulting labels from clustering will be separated into the training sets and testing tests through sklearn with a random_state = 4400. In choosing the correct model, each training model will fit the training set using default hyperparameters and the performance will be evaluated using the testing set, using metric "R-squared value for training/testing set" and "MSE value for training/testing set" R-square indicate the percentage of data explained through this model and the MSE value means the error during the prediction of the samples. The model with the best metric will be chosen.

3.5.2 Choosing the Best Hyperparameter

After the correct model is chosen, the hyperparameter will be found via grid-search. This is done using the KFold of sklearn module, and the number of folds will be five; through grid-search, the best parameters will be found and displayed for final evaluation.

3.6 Result - Classifying

3.6.1 Choosing the correct model

```
SVM classifier:
        R-squared value for training set:  0.9229171523237018
        R-squared value for testing set:  0.920073266172675
        MSE value for training set:  0.3165266106442577
        MSE value for testing set:  0.26666666666666666

Random Forest Classifier:
        R-squared value for training set:  1.0
        R-squared value for testing set:  0.6752976438264924
        MSE value for training set:  0.0
        MSE value for testing set:  1.0833333333333333

Logistic Regression:
        R-squared value for training set:  1.0
        R-squared value for testing set:  0.9075847140121556
        MSE value for training set:  0.0
        MSE value for testing set:  0.30833333333333335
```

From the above result, we see that Logistic regression did the best in classifying the data. This is because although currently, SVM did have the R-squared value 0.92 for both the training set and testing set. SVM

has reached the best but Logistic regression still has room for improvement. The R-squared value for the training set is 1 and the R-squared value for the testing set is 0.90. This number means that around 90% of the testing data is explained via this model while 100% of the training data is explained. The low MSE of the testing set means fewer errors in the prediction process. This means the Logistic regression model did perfectly on the training set and can still predict the labels of the testing set with decent accuracy. Through hyperparameter tunning, we are able to reach better results.

This result is probably because as we use k-means for the clustering, the cluster will have clear geometric shapes for each cluster. This gives a clean linear separator between one class and another class, and such a simple linear separator can be best captured through a Logistic regression model.

3.6.2 Choosing the best hyperparameter
As logistic regression will be chosen, C, regularization parameter, that is the hyperparameter of the Logistic Regression model, will be used. Three intervals, 0.01, 0.1, and 1, will be tested via cross-validation.

```
Best cross-validation score:  0.9804381846635367
Best parameters:  {'C': 0.1}
Training set score with best parameters:  1.0
Test set score with best parameters:  0.9583333333333334
```

From the result, we see that the best hyperparameter chosen is C with 0.1. In this situation, the training result stayed the same while the classification results got better by 5% exceeding the previous SVM result.

In conclusion, we are able to predict a given player by using the stats generated on the field. This is done by first clustering the data into different types and training a classifier using the data labeled by the KNN model.

4 Participant's Contribution

Arnav Joshi: Data Cleaning, Technical Approach, Average Methods, Formatting
Kevin Andrews: Data Research, Data Cleaning, Data Analysis, Clustering, Output Analysis
Eric Sun: Data Analysis, Clustering, Classification, Model Selection, Hyperparameter Tuning

References

[1] Cheng, A. (2017, March 9). *Using machine learning to find the 8 types of players in the NBA*. Medium. Retrieved December 18, 2021, from https://medium.com/fastbreak-data/classifying-the-modern-nba-player-with-machine-learning-539da03bb824

[2] *Clustering*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/clustering.html#k-means

[3] *Linear Models (Logistic Regression)*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

[4] *NBA stats 2021/22: All player statistics in one page*. NBAstuffer. (2021, December 12). Retrieved December 18, 2021, from https://www.nbastuffer.com/2021-2022-nba-player-stats/#

[5] *Sklearn.ensemble.randomforestclassifier*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[6] *Sklearn.model_selection.GRIDSEARCHCV*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[7] *Sklearn.model_selection.Kfold*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

[8] *Sklearn.svm.SVC*. scikit. (n.d.). Retrieved December 18, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html