# Predicting where college basketball players will be drafted in the NBA draft and NBA performance

Team 13: Alex Chan, Mike Selesko, and Christin Adjei

## Introduction

The NBA is a professional basketball league in North America. The NBA draft occurs at the end of June every year in which players are drafted by teams in the NBA. There are two rounds, each consisting of 30 picks, with a total of 60 picks. The supervised learning portion of this project aims to predict the pick number of a player based on their college statistics. We also hope to gain insights into which features are important for determining where a player will be drafted. This could be a valuable resource for teams in their draft planning as knowing where a player will be drafted can help teams approach how they want to position themselves in the draft. The unsupervised learning portion of this project aims to cluster players already in the NBA based on their statistics. The goal is to learn about which features are important for different groups of players. We were motivated to do this project as we are all sports fans and wanted to learn more about gaining insights on player success through machine learning methods. The NBA draft prediction model ended up getting within 15 picks of the correct pick number. The clustering methods grouped players based on how well they produced.

## Related Work

*Predicting National Basketball Association Success: A Machine Learning Approach*

This project was done to predict the success of players based on their college statistics. This is similar to our project, but instead of predicting player success in the NBA we are predicting draft position (Costa et al. 2024).

*Applying Machine Learning to Predict NFL Draft*

This project looks at predicting which round a player will be chosen in the NFL draft, similar to how we are predicting the pick number for the NBA draft. The difference is that we want to specifically try to predict the pick number, not just the round (Grassy).

*Redefining NBA Player Classifications using Clustering*

This project clusters NBA players using hierarchical clustering. This project is similar to our NBA clustering, however we will not be using hierarchical clustering. We are using DBSCAN and K-means (Jyad, 2020).

## Data Source

The datasets were obtained from the Stathead API (https://stathead.com/basketball/). Datasets were downloaded as csv files. Datasets further described below in their appropriate sections.

## <u>Part A. Supervised Learning</u>

### Methods Description

The supervised learning part of our project aims to develop a machine learning model to predict the draft pick of college players. If our model can accurately predict when a specific player will be drafted based on their playing statistics while in college, both stakeholders and players can benefit. For stakeholders, such as the scouting department or general manager for a team, predicting when a player will get drafted will allow them to identify players they are most likely to draft based on their designated picks. This would benefit successful teams who end up receiving late picks. For players, they can look at strengths and weaknesses of players predicted to get drafted earlier and improve their own statistics to enhance their draft prospects. If players drafted in the top 10 tend to record more steals, a player can focus on improving their defensive skills to match this trend.

The *pick* (numerical) is the target variable. Each draft consists of 60 picks. For our first model development phase, we used all numerical variables provided in the data except for variables revolving around 3-point shooting (3P, 3PA, and 3P%) as features. We excluded these variables because many players didn't attempt any 3-point shots during their college career. This caused an error when we were building our model because NaN was present in the '3P%' column. Replacing the NaNs with zeros didn't seem like a good approach because it would have skewed the data. For the second model development phase, we conducted feature engineering and dropped features we didn't think carried significance. Description for features can be found in Appendix A.

**Methods and Evaluation**

We built 3 different supervised learning models, then selected the most accurate one for detailed analysis. The 3 algorithms include:

1. Linear Regression
    a. This model was chosen for its simplicity and interpretability, creating an ideal starting point for understanding the relationship between player data and draft pick. It provides a clear mathematical equation that represents how each feature affects the target variable. The ease of implementation and speed of computation make it an adequate baseline model for comparison of more complex models. Although the model assumes a linear relationship between the features and the target variable, it helps create a foundation for understanding our data's underlying trends.
2. Random Forest Regression
    a. This model was chosen for its ability to handle non-linear relationships between features and the target variable and provide robust predictions. It is an ensemble method that creates multiple decision trees during training, averaging their outputs to improve the accuracy of predictions and reduce overfitting. An important feature of this model that supports its use is its ability to deal with high-dimensional data and capture complex interactions between features. Additionally, it offers insights into feature importance, allowing for a deep understanding of which player stats affect their draft pick the most.
3. Gradient Boosting Regression
    a. This model was chosen for its performance and ability to model complex patterns in the data. By sequentially building trees that correct the errors of previous ones, this model incrementally boosts prediction accuracy. Its flexibility allows it to handle different data distributions. Additionally, its regularization techniques, involving learning rate and subsampling, can reduce overfitting.
4. K-Nearest Neighbors
    a. This model was chosen because of its non-parametric nature, allowing for assumptions about the underlying data distribution. The simplicity and flexibility allow it to effectively capture local patterns and relationships in the data, which can be useful for situations where draft picks depend on specific clusters of player statistics. By predicting outcomes based on the closest data points in the feature space, the model can provide a different perspective compared to the other models.

We used Root Mean Squared Error (RMSE) for model performance. RMSE provides a unit of measurement that is easy to interpret for this project. The value itself represents, on average, how far off the predicted pick was from the actual pick. Since our target variable is continuous, the use of RMSE is appropriate. It measures the accuracy of continuous predictions and provides a straightforward way to see how close the model's predictions are to the actual picks.

For model training, a train/test split of 80/20 was used. College data for players drafted between 2000 and 2018 was used for training and college data for players drafted between 2019 and 2023 was used for testing. Since the actual pick for players in the testing data was already known, it was easy to determine the accuracy of predictions. For the Gradient Boosting Regression and K-Nearest Neighbors models, we used GridSearchCV to find the best parameters for hyperparameters tuning.

*First Phase of Model Development*

| | Linear Regression | Random Forest Regression | Gradient Boosting Regression | K-Nearest Neighbors |
|---|---|---|---|---|
| RMSE | 8.377 ± 0.267 | 8.479 ± 0.234 | 8.472 ± 0.191 | 15.967 ± 0.520 |

In the first phase of model development, every numerical variable provided for a drafted college basketball player was used, with the exception of 3-point shooting (as described in *Methods Description*). The models all produced RMSEs around 8.5, except for K-Nearest Neighbors, which was almost double. Linear Regression had the lowest RMSE, indicating it is the best-performing model among the models we tested. This suggests that the relationship between the features and the draft picks might be approximately linear, or that for this specific dataset, linear regression is well-suited.

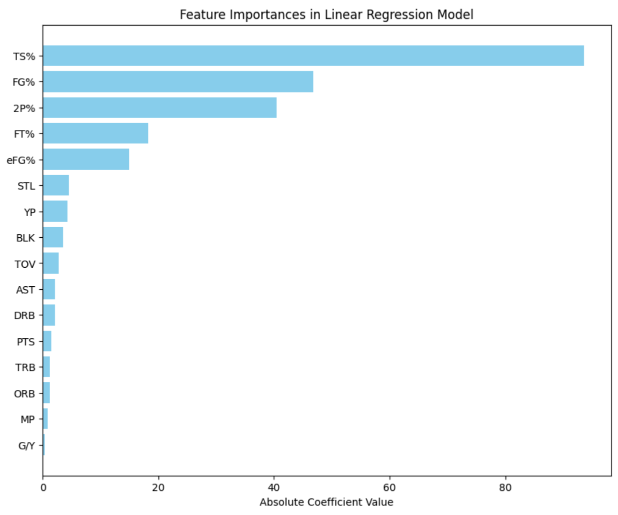
*Second Phase of Model Development*

| | Linear Regression | Random Forest Regression | Gradient Boosting Regression | K-Nearest Neighbors |
|---|---|---|---|---|
| RMSE | 15.182 ± 0.860 | 15.644 ± 0.675 | 15.656 ± 0.743 | 16.595 ± 0.323 |

The second phase of model development produced results that were very different from the first phase. For this phase, we created two new features: 'YP' and 'G/Y'. YP stands for *Years Played* and represents the number of seasons a player played college basketball. G/Y stands for *Games per Year* and represents how many games per season a player played in. We also removed features that didn't seem important such as *PF (personal fouls), FGA (Field Goals Attempted), and 2PA (2 Points Attempted).* We removed one feature that had significant importance in the first phase: *Round*. This feature represented which round a player got drafted in (first or second).

## Feature Analysis

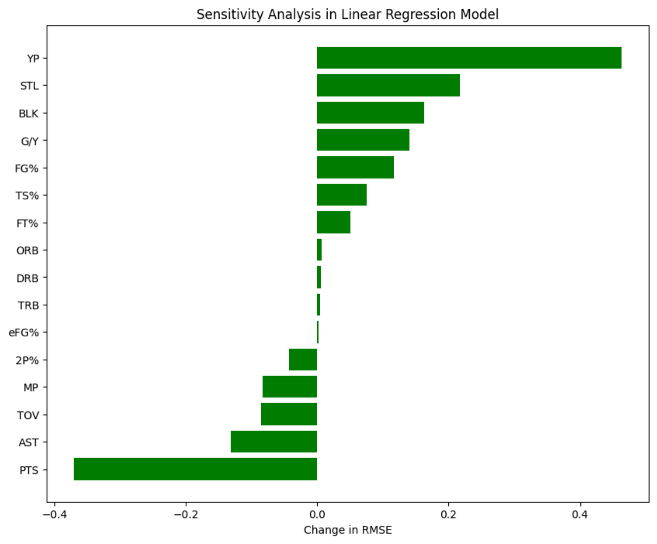
We performed feature analysis on our best model, which was the Linear Regression model. For tree-based models like Random Forest or Gradient Boosting, feature importance is straightforward. However, for Linear Regression models, the importance of a feature is typically associated with its coefficient in the regression equation. Each feature is assigned a coefficient,



which represents the expected change in the target variable for a one-unit change in the feature, while holding other variables constant. A bigger absolute value for the coefficient indicates a stronger influence of the feature on the target variable. As shown in the figure to the right, features revolving around shooting had the largest effect on draft pick, with *True Shooting %* having the absolute largest effect. Rebounding stats and playing time/games played had the smallest effect on draft pick.

### Sensitivity Analysis

Sensitivity analysis helps to understand how changes in model parameters or features affect the model's performance. For our Linear Regression model, since there are no hyperparameters to tune, we can focus on sensitivity to feature selection. We can do this by systematically removing each feature and observing the change in model performance. According to the figure on the right, *YP (Years Played)* and *PTS (Points per Game)* were the most sensitive features and caused the most change in RMSE when removed from the model.



### Trade Offs

1. *Simplicity vs Performance*
    a. The Linear Regression model was our best performing model in both phases of development. The model is simple and interpretable, with the lowest RMSE among the models we tried. However, it assumes a linear relationship between features and the target variable, which might not capture complex patterns in the data.
    b. More complex models like Random Forest and Gradient Boosting can capture non-linear relationships and interactions between features, potentially providing better performance on complex datasets. However, they are more complex and less interpretable than Linear Regression.
2. *Interpretability vs Accuracy*
    a. Linear Regression offers high interpretability, with coefficients that directly indicate the relationship between each feature and the target. This is useful for understanding and communicating the model
    b. Random Forest and Gradient Boosting are potentially more accurate but are less interceptable. SHAP values can help with interpretation, but the models' decisions are still less transparent compared to Linear Regression.

### Failure Analysis

- *Luka Garza* - Predicted: 17.72, Actual: 52
    - Had very high percentages across shooting metrics. As we observed in the feature importance for the Linear Regression model, these features had the highest coefficients. He played in a weak conference which might have inflated his statistics.
- *Cade Cunningham* - Predicted: 20.29, Actual: 1
    - Had very low percentages across shooting metrics. Even with low shooting percentages, he still averaged around 20 points a game and NBA teams most likely saw other attributes that they liked.
- *Zion Williamson* - Predicted: -10.96, Actual: 1

  - Had very high numbers for almost every statistic. He was considered one of the best college basketball prospects of all time before he even got to Duke. During the one season he played, he dominated every game and his statistics highlight this. He was so good that our model predicted a negative pick for him.

In order to improve these failures, there are adjustments we can make to the data. One adjustment would be to standardize player data based on what school the player attended. Every school has a strength of schedule score that isn't included in our data which represents the average 'difficulty' of schools they play against. Incorporating this to standardize player data can help make players with low statistics look 'better' and help make players with high statistics look 'worse'.

## Unsupervised Learning

### *Methods Description*

**Data:**

For this analysis, we utilized a comprehensive dataset of NBA player statistics from the 2023-2024 season, sourced from Basketball Reference. This choice ensures our analysis is based on the most recent and relevant information available in professional basketball.

**Dataset insights:**

Scope: Per-game statistics for the 2023-2024 NBA season

Size: 735 records across 31 fields

Player Coverage: 572 unique players

Teams: All 31 NBA teams represented

The dataset encompasses 12 distinct player positions.

This diverse positional classification allows for nuanced analysis of player roles and contributions.

**Data variables:**

The 31 fields in our dataset encompass a wide range of statistical categories, including but not limited to:

Basic stats: PTS(Points), Rebounds, Assists

Shooting percentages: Field Goal %, Three-Point %, Free Throw %

Advanced metrics: Player Efficiency Rating (PER), True Shooting %

Playing time: Minutes per game

Defensive stats: Steals, Blocks

This comprehensive set of statistics provides a holistic view of player performance, enabling our clustering algorithm to identify meaningful patterns and archetypes.

**Data cleaning:**

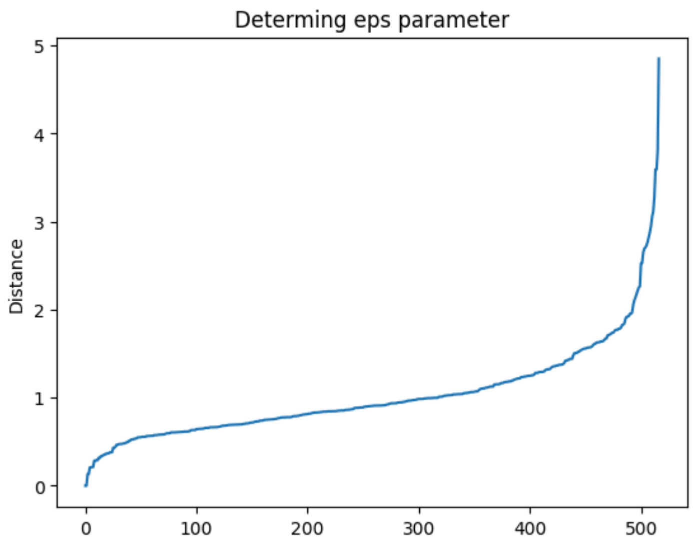
Before analysis, we conducted thorough data cleaning and preprocessing steps to ensure data integrity. This included handling missing values, removing duplicates, and standardizing statistical formats for consistency across all players and teams.

**DBSCAN:**

For the first unsupervised learning method we used Density-Based Clustering of Applications with Noise (DBSCAN). An advantage to the DBSCAN method is that you do not need to specify the number of clusters beforehand. The clusters are based on the density of the data points. In this case, we were looking to identify patterns in the NBA player data so we did not know how many clusters to expect beforehand. Another advantage to DBSCAN is that it is good for outlier detection, as some of the data points get labeled as outliers. This could prove helpful in looking at which types of NBA players are outliers. For feature representation in DBSCAN, we chose to use PCA to reduce the dimensionality. Because DBSCAN is a density based approach, we wanted to reduce the dimensionality.

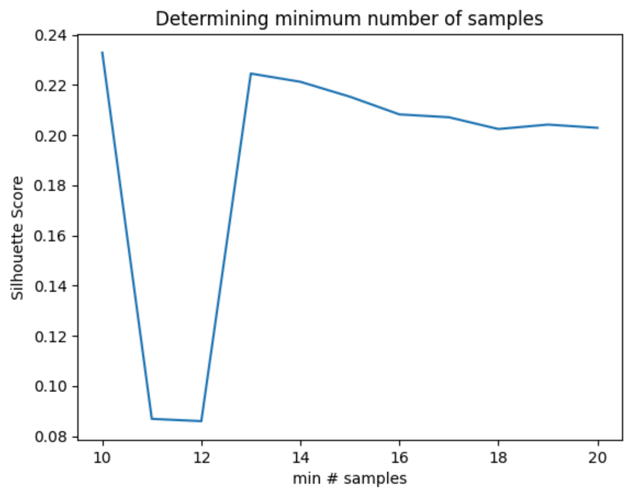
The first step was to determine how many principal components to use. This involves looking at the cumulative explained variance that each principal component adds. Five principal components explained about 90% of the variance, with diminishing returns adding more principal components. This reduced the number of features from 24 to 5.

The first hyperparameter to tune for DBSCAN is the eps. The eps is the distance between data points in order for them to be considered part of a cluster. In order to determine the eps, we want to plot the distances between points and look for where the elbow begins. This distance gives us the eps parameter. The distances are created using the nearest neighbors algorithm.



The figure to the right shows the plot for the distances between points. The elbow begins around a distance of 1.5, thus we will use 1.5 as our eps parameter.

The next parameter to determine is the minimum number of samples. This is the minimum number of data points for a group to be considered a cluster. A general rule of thumb is that the minimum samples should be about twice the number of dimensions in the dataset (Sander et al. 1998). We have 5 dimensions and so will start the tuning at 10 for the minimum number of samples. If the data are noisy, then the number of samples should be higher. In order to evaluate the minimum number of samples parameter we will start at 10 and increase it to 20. It will be evaluated by the silhouette score for each model. Each model will use an eps of 1.5 as determined previously.



**K-Means:**

For this analysis, we want to utilize the k-means clustering method. K-means clustering is a popular unsupervised machine learning algorithm used to partition a dataset into K distinct, non-overlapping subsets or clusters. The main goal is to group data points in such a way that points in the same cluster are more similar to each other than to those in other clusters. Using K-means clustering, we can analyze the 2023-24 NBA player statistics to discover natural groupings of players, which can then be used to inform decisions about team selection, player development, and game strategy.

**Feature Engineering and Dimensionality Reduction**

A critical challenge in this analysis is addressing the curse of dimensionality. With 31 initial features, it is essential to identify the most relevant ones to enhance model performance and interpretability.

First, we removed non-numeric features such as ranking (Rk), player name (Player), position (Pos), age (Age), and team (Tm). This left us with 26 numeric features, which still presents a high-dimensional space for clustering.

Domain expertise in basketball analytics will be crucial and most appropriate for guiding feature selection. This ensures that the selected features are not only statistically significant but also meaningful in the context of player evaluation.

**Principal Component Analysis (PCA)**

We applied PCA as a dimensionality reduction technique and explored various thresholds for variance retention:

**95% retention**: Resulted in 21 features (10 principal components).

**90% retention**: Yielded 19 features (7 principal components).

**70% retention**: Produced 10 features (2 principal components).

We opted for 70% variance retention to achieve an optimal balance between data representation and model simplicity.

Choosing the 70% threshold significantly reduces dimensionality from 26 to 10 features while retaining two principal components, which allows for easier visualization and interpretation. This approach captures the most influential aspects of player performance while filtering out potential noise.

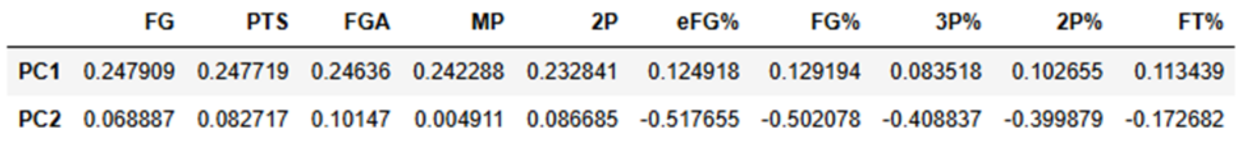
**Model Flexibility**

Our framework is flexible, allowing for adjustments in the variance retention threshold or the selection of specific features based on analytical needs. This flexibility enables fine-tuning the model for different analytical perspectives or team-specific requirements.

**Model Building**

To provide a focused and interpretable context for our analysis, we chose to build and test our model on a specific player position: Point Guards (PG). Focusing on a single position helps control position-specific statistical variations.

As discussed in our feature engineering section, we applied PCA to reduce dimensionality while retaining the most significant information. The results of this process are summarized below:



| | FG | PTS | FGA | MP | 2P | eFG% | FG% | 3P% | 2P% | FT% |
|---|---|---|---|---|---|---|---|---|---|---|
| **PC1** | 0.247909 | 0.247719 | 0.24636 | 0.242288 | 0.232841 | 0.124918 | 0.129194 | 0.083518 | 0.102655 | 0.113439 |
| **PC2** | 0.068887 | 0.082717 | 0.10147 | 0.004911 | 0.086685 | -0.517655 | -0.502078 | -0.408837 | -0.399879 | -0.172682 |

Figure: PCA summary

From the above table summary, we could see that PC1: Seems to represent overall offensive production and playing time and PC2: Seems  to contrast volume scoring against shooting efficiency.

With these principal components established, we're now positioned to apply our K-means clustering algorithm. We used these features captured as the most important features on PC1 and PC2 for the K-means clustering.

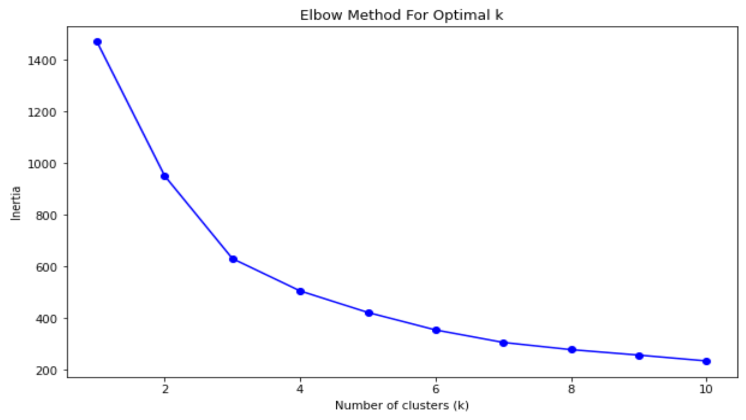
**Determining the Optimal Number of Clusters**

K-means clustering requires specifying the number of clusters at the outset, which can be determined using methods like the elbow method or silhouette analysis.

**Elbow Method**

The elbow method is a technique used to determine the optimal number of clusters in K-means clustering by identifying the "elbow" point in the plot of the within-cluster sum of squares (WCSS) against the number of clusters. The point where the rate of decrease in WCSS slows down represents a good trade-off between minimizing variance within clusters and avoiding overfitting.

The elbow plot to the right indicates a slight flattening at point 3, suggesting that 3 or 4 clusters could be suitable. However, to ensure clarity, we will conduct a silhouette analysis to determine the optimal number of clusters more accurately.



**The silhouette analysis**

The silhouette method evaluates the quality of clustering by measuring the similarity of data points within clusters compared to other clusters. It provides a score between -1 and 1, where higher values indicate better-defined clusters.

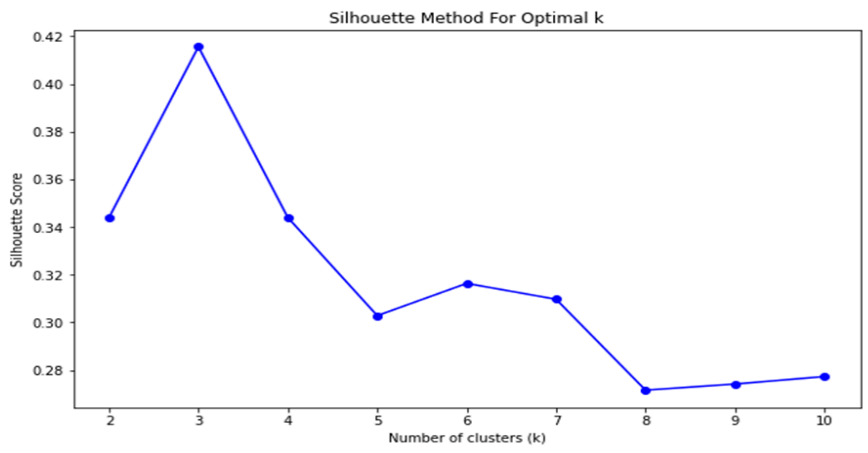


Figure: Silhouette Score Plot

From the silhouette plot above , the  silhouette score indicates that the highest score is achieved with 3 clusters, followed by 4 clusters.

We proceeded to build the k-means clustering model with the initial cluster selection of 3 and 4 clusters and compared the results.

**Method Initialization and Data Preprocessing**

Initialization Methods

To ensure robust clustering results, we employed and compared multiple initialization methods for the K-means algorithm:

K-means++: This method intelligently selects initial centroids, leading to faster convergence and potentially better results.

Random: Traditional random initialization, providing a baseline for comparison.

PCA-based: Utilizes principal components to guide initial centroid placement, potentially aligning with the underlying data structure.

Each method was evaluated based on the quality and stability of the resulting clusters, as well as computational efficiency.

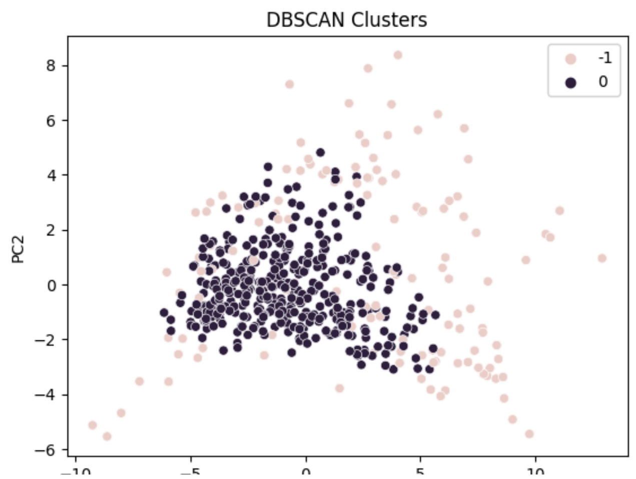
**Data Scaling**

Prior to clustering, we standardized the data to ensure all features contribute equally to the analysis, regardless of their original scale. This critical preprocessing step involves: Applying StandardScaler from the scikit-learn library. Transforming each feature to have a mean of 0 and a standard deviation of 1. Standardization is particularly important for K-means clustering, as the algorithm is sensitive to the scale of input features. This process prevents features with larger magnitudes from dominating the clustering outcomes, ensuring a fair comparison across all player statistics.

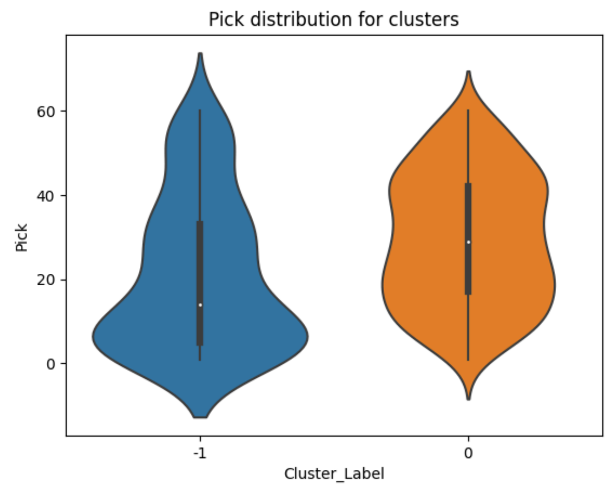
By systematically comparing different initialization methods and properly scaling our data, we aim to produce the most accurate and meaningful clusters of NBA player archetypes.

**Results**

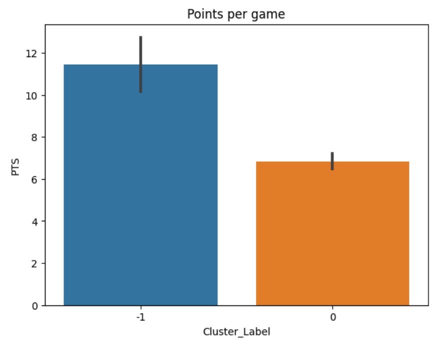
**DBSCAN:**

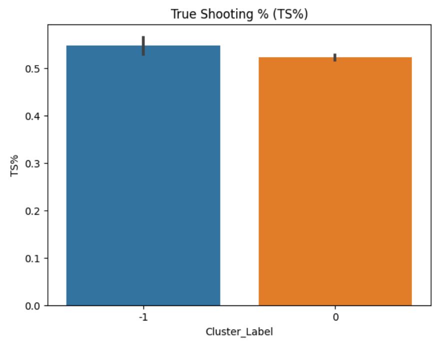


The figure to the right shows a scatter plot of the clusters in 2 dimensions, using the first and second principal components for the sake of visualization. The DBSCAN method resulted in one main cluster (labeled 0). The data points that are labeled as -1 are classified as noise or outliers. We can further investigate the attributes of these clusters.



We wanted to use this method to relate back to the college data analyzed previously. The figure to the right shows the pick distributions of the players in each cluster. For the outlier group (-1), we can see that they have a lower average pick number. This means that they were drafted higher in the draft, meaning they were thought to have a higher upside when entering the NBA. Next we can look at some key statistics of their playing time in the NBA.

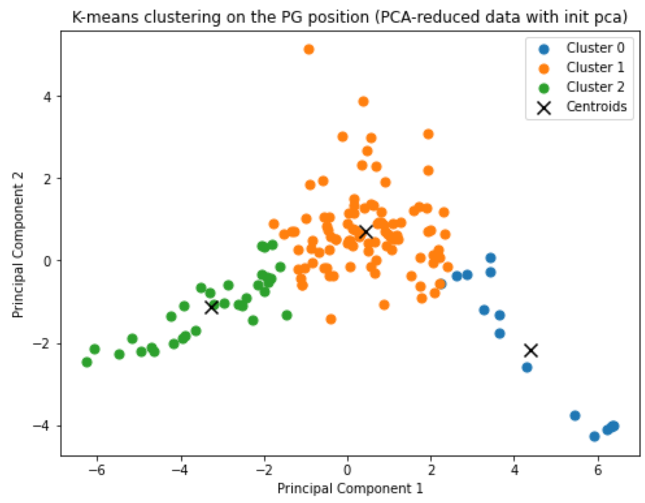




The figure above shows the average points per game (left) and the true shooting percentage (right). This statistic is used because it encompasses the three ways that a player can score in basketball: 2 point field goal, 3 point field goal, and free throws. For the points per game, we see that the outlier group scores almost double the amount of points and the main cluster. The true shooting percentage is also higher for the outlier group, although not to the same extent. It is only slightly higher.

The figure to the right shows that we get the highest silhouette score when using a minimum number of samples of 10.

The final model uses an eps of 1.5 and minimum number of samples of 10.



The K-means++ initialization method achieved a higher silhouette score of **0.4156** compared to **0.4070** with PCA initialization, indicating better cluster separation and cohesion.

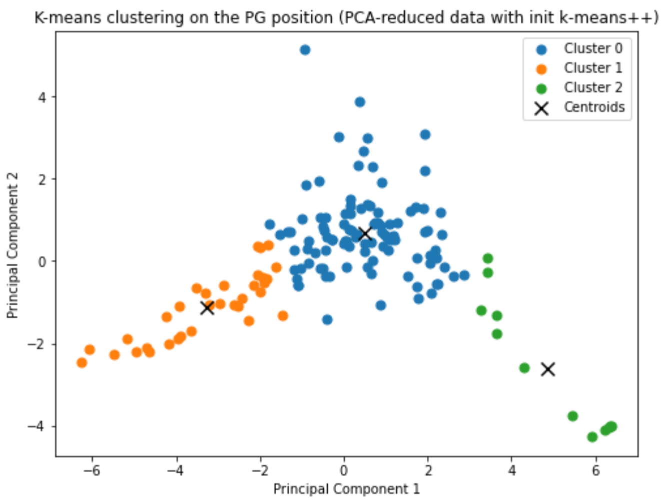
In the figure Cluster plot -pca, K-means clustering was applied to the PCA-reduced data with an initialization PCA. Here are the evaluation metrics for k=3 clusters: Silhouette Score: **0.4070**

The silhouette score of 0.4070 indicates a moderate level of cluster cohesion and separation. The clusters are somewhat distinct, but there is still some overlap or noise present. This initialization method provides a reasonable clustering outcome, but there may be room for improvement.

In the figure Cluster plot - k-means++, K-means clustering was applied with K-means++ initialization on the same PCA-reduced data. Here are the evaluation metrics for k=3 clusters:

Silhouette Score: **0.4156**

The silhouette score of 0.4156 is slightly higher than that of the PCA initialization method, suggesting that the clusters are slightly more cohesive and better separated. This indicates that K-means++ initialization yields a marginally better clustering outcome compared to the PCA initialization.



Both methods achieve reasonable clustering results, with silhouette scores indicating moderate cluster separation. However, the K-means++ initialization method shows a slight improvement in the silhouette score.
Visual inspection of the plots reveals that K-means++ initialization provides more distinct clusters, whereas the PCA initialization shows some overlap between clusters.

Cluster Summary

When we look at the clusters on the plot and their summaries, we can see some interesting patterns in player performance.

Cluster 1 is made up of high-performing scorers. On average, these players score 19.95 points, with a 45% field goal percentage and a 53% effective field goal percentage.

Cluster 0 includes moderate contributors. These players typically score around 5.91 points, with a 42% field goal percentage and a 50% effective field goal percentage.

Cluster 2 consists of rarely-used reserves. They have low averages: 0.75 points, 13% field goal percentage, and 14% effective field goal percentage.

**Recommendations**

Optimize Team Strategy: Cluster 1 players are your go-to scorers. They should be the focal point of your offensive strategy. Design plays that maximize their scoring opportunities and leverage their high shooting percentages. Cluster 0 players are reliable contributors who can support your star players. Use them to maintain balance on the court, providing steady scoring and shooting efficiency. Cluster 2 players are rarely-used reserves, so consider using them in specific situations where their limited offensive contribution can be offset by other team members' strengths.

Tailor Training Programs: High-performers (Cluster 1) might benefit from advanced training focused on maintaining their performance levels and expanding their skills. Moderate contributors (Cluster 0) could use targeted training to elevate their game to the next level, focusing on areas where they can improve their scoring and shooting efficiency. Reserves (Cluster 2) may need more foundational training and development to increase their overall contribution and readiness for more playing time.

Manage Player Rotations: Use Cluster 1 players in high-stakes situations where scoring is crucial. Rotate Cluster 0 players to maintain consistent performance throughout the game, ensuring they complement your star players. Deploy Cluster 2 players in low-pressure scenarios or when giving key players rest, to minimize the impact of their lower performance.

Inform Contract Decisions and Trades: Players in Cluster 1 are valuable assets. Retain them and consider building your team around their capabilities. Cluster 0 players provide solid value and could be included in trade discussions to strengthen other areas of the team. Evaluate the potential of Cluster 2 players and decide if further development is worthwhile or if their roster spot could be better utilized**.**

**Limitations**

Position Specificity and Data Limitations: It's important to note that some players in our dataset played multiple positions, and others might be misclassified. This could skew the results a bit. To make things simpler, we focused only on point guards (PG) for this analysis. While this helps in making the data more manageable, it does limit our understanding of player roles across all positions. Plus, we're mainly looking at offensive stats here, which means we're missing out on defensive contributions and other aspects of the game. This might not fully capture the value of defensive specialists or point guards known for their leadership.

Cluster(k) Number: We chose to use three clusters for this analysis, which makes sense but can also be a bit simplistic. Player types are diverse, and reducing them to just three groups might not capture all the nuances.

PCA Reduction: We used Principal Component Analysis (PCA) to reduce the data to two principal components. This helps with feature selection and visualization, but it also means we might lose some of the more detailed information.

Single-Season Data: The use of data from just one season doesn't account for player development, injuries, or changes from year to year. We're capturing current performance, but this doesn't reflect player potential or their trajectory over time.

**Model Comparison**

| **Model** | **Silhouette Score** |
|---|---|
| DBSCAN | 0.2328 |
| K-Means | 0.4156 |

The silhouette score was higher for the K-means method.  This means that the K-means clustering algorithm was better at assigning players to their correct cluster.

**Sensitivity Analysis**

Sensitivity analysis was done for each model during the model creation phase by looking at the silhouette score when tuning parameters (see above). For DBSCAN, the model was sensitive to the number of minimum samples. We saw a decrease in the silhouette score when it was increased from 10. For K-means the silhouette score was sensitive to the number of clusters specified.

## Discussion

**Part A**

A. What surprised us about our results was the range of predicted values the model produced. Besides two outliers that had negative predicted values. The rest of the predicted values were between 10 and 45, which isn't accurate considering the actual range is from 1 to 60. This tells us that the statistics of drafted players are very close and there aren't many players that are significantly better or worse than the rest in terms of statistics.

B. One challenge we encountered was data quality and preprocessing. Dealing with missing values, inconsistencies, and outliers in the data. We found that college data collected from before 2005 was not entirely accurate and was missing some values such as shooting percentages. We made sure missing values were filled and removed values that weren't needed such as 3-point %.

C. With more time/resources, we would try to standardize player data based on the school they attended and strength of the opponents they played against. Also, we could look at the draft order and determine exactly what kind of player each team needs. For example, if the Boston Celtics are drafting number 1 and they really need a center, we could look at the best college centers' statistics and determine who they would draft.

**Part B**

A. It surprised us that data did not form into nice, neat clusters. We thought that the data would separate players into their positions because, in theory, different positions have different roles on the team. For example, centers and forwards will tend to have more rebounds and guards more assists. However, players are much more versatile and can produce in all statistics from any position. The groups had more to do with a player's overall performance.

B. One challenge that we faced was getting clusters to form initially. We first approached this by reducing the data using PCA. This seemed to help with the algorithm. For example, with DBSCAN we were getting one large cluster of noise and then several smaller clusters. Hyperparameter tuning was also key in getting clusters to form and extracting information out of them.

C. If we had more time we could try different clustering algorithms, such as hierarchical clustering, in order to compare to the models that we created. It's possible that a different algorithm would have helped create more clearly defined clusters.

## Ethical Considerations

**Part A.**

A. Transparency and Accountability
    a. Lack of transparency in how the model makes predictions can lead to mistrust and accountability issues.
        i. Using explainable AI techniques to make the model's decisions understandable to stakeholders could help with this
        ii. Maintaining comprehensive documentation of the model development process, including data sources, feature selection, and model evaluation can also help

**Part B**

A. Resulting contracts due to model
    a. Players in the NBA make their living based off of how they play. An algorithm that groups players based on their performance could affect how a team views them and thus how they are paid. We should be completely transparent with both players and teams about how the algorithm works so that one group does not use it unfairly in negotiations.

## **Statement of Work**

Alex Chan led development of the supervised learning models in predicting NBA player draft position.

Mike Selesko led development of the NBA clustering using the DBSCAN method.

Christian Adjei led development of the NBA clustering using the K-means method.

All team members contributed to the writing of the report as well as the visualizations created.

**References**

Jyad, A. (2020, November 19). *Redefining NBA player classifications using clustering.* Medium. https://towardsdatascience.com/redefining-nba-player-classifications-using-clustering-36a348fa54a8

Costa, D., Fechine, J., Brito, J., Ferro, J., Costa, E., & Lopes, R. (2024). A machine learning approach using interpretable models for predicting success of NCAA basketball players to reach NBA. *Proceedings of the 16th International Conference on Agents and Artificial Intelligence*. https://doi.org/10.5220/0012390000003636

Grassy, B. (n.d.). Applying machine learning to predict the NFL draft. https://www.brockgrassy.com/blog/applying-machine-learning-to-predict-the-nfl-draft/

Sander, J., Ester, M., Kriegel, HP. *et al.* Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications. *Data Mining and Knowledge Discovery* 2, 169–194 (1998). https://doi.org/10.1023/A:1009745219419

**Appendix A - College Player Data**

| Column Name | Description | Data Type |
|---|---|---|
| Player | Player first and last name | Text |
| Draft Team | NBA team that drafted player | Text |
| Round | Round player was drafted in | Integer |
| Pick | Pick player was drafted | Integer |
| Draft Year | Year player was drafted | Integer |
| Draft College | College player played at | Text |
| From | First season player played | Text |
| To | Last season player played | Text |
| G | Number of games | Integer |
| GS | Number of games started | Integer |
| MP | Average minutes played | Float |
| FG | Average field goals scored | Float |
| FGA | Average field goals attempted | Float |
| 2P | Average 2-pointers scored | Float |
| 2PA | Average 2-pointers attempted | Float |
| 3P | Average 3-pointers scored | Float |
| 3PA | Average 3-pointers attempted | Float |
| FT | Average free throws scored | Float |
| FTA | Average free throws attempted | Float |
| ORB | Average offensive rebounds | Float |
| DRB | Average defensive rebounds | Float |
| TRB | Average total rebounds | Float |
| AST | Average assists | Float |
| STL | Average steals | Float |
| BLK | Average blocks | Float |
| TOV | Average turnovers | Float |

| | | |
|---|---|---|
| PF | Average personal fouls | Float |
| PTS | Average points scored | Float |
| FG% | Field goal percentage | Float |
| 2P% | 2-point percentage | Float |
| 3P% | 3-point percentage | Float |
| FT% | Free throw percentage | Float |
| TS% | True shooting percentage | Float |
| eFG% | Effective field goal percentage | Float |
| Pos | Position | Text |