Michael Jordan vs Lebron James – Who's the GOAT?

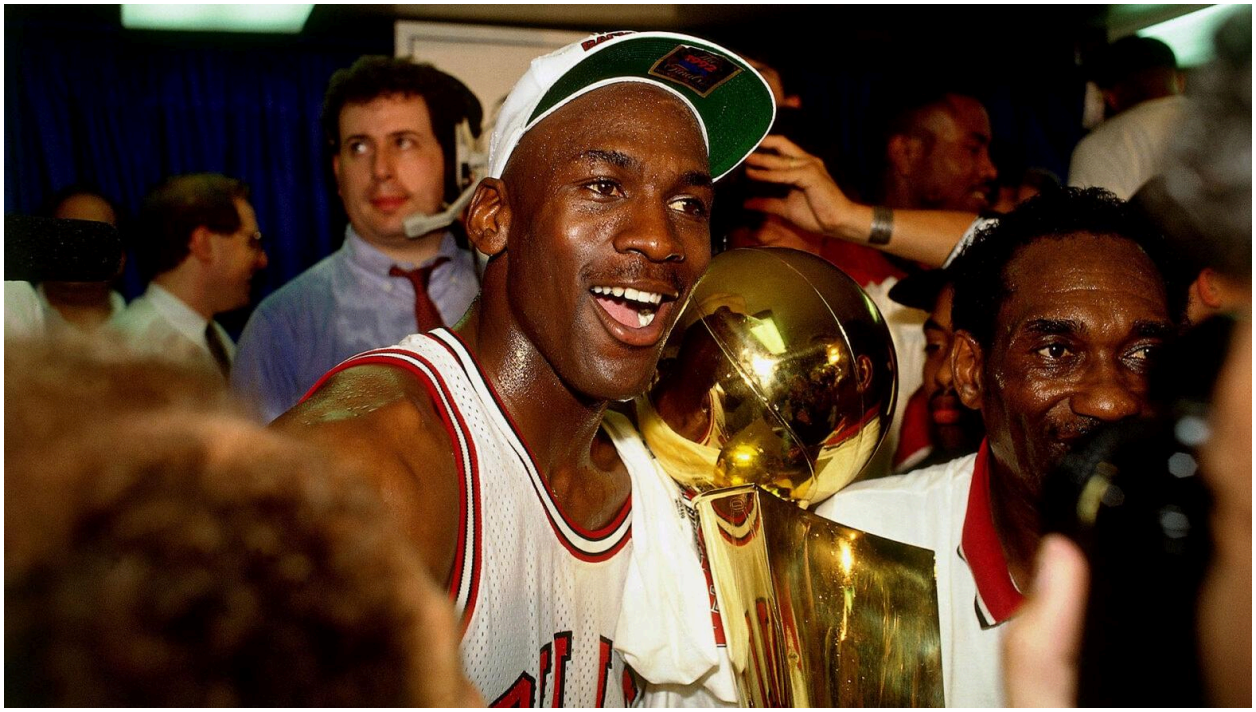
Matthew Ilejay
ilejaym@oregonstate.edu
Computer Science
CS 432 - Intro to Applied Machine Learning

Table of Contents

Introduction

The debate over who is the greatest basketball player of all time—commonly referred to as the "GOAT" (Greatest of All Time)—has become one of the most iconic and polarizing discussions in sports culture. For decades, fans, analysts, and athletes have argued about whether Michael Jordan or LeBron James deserves the crown. This debate spans generations, evokes intense emotions, and reflects the evolving nature of basketball. From talk shows to barbershops to online forums, the GOAT conversation dominates basketball discourse and has become a cultural phenomenon.

Historically, Michael Jordan was seen as the standard of excellence, especially after leading the Chicago Bulls to six championships in the 1990s. His dominance, charisma, and clutch performances earned him a legendary reputation. Decades later, LeBron James entered the scene and built an equally compelling legacy through exceptional longevity, versatility, and achievements across multiple teams. Their careers occurred in different eras of the NBA, making direct comparisons difficult but also more intriguing. As the league evolved, so did the criteria for greatness—shifting from simple statistics and titles to more advanced and nuanced evaluations.



The importance of this debate extends far beyond just entertainment. It reflects broader themes in sports such as leadership, adaptability, physical excellence, and the impact of era-specific challenges. Discussions about the GOAT engage fans in meaningful conversations about performance under pressure, consistency, and influence on the game itself. These conversations also mirror how people evaluate excellence in other domains—whether in music, film, or business. In this way, the GOAT debate becomes a window into how society values achievement and legacy.

This debate also affects a wide range of individuals, from die-hard basketball fans and sports historians to young athletes and casual viewers. It influences how the next generation of players train, how sponsors select ambassadors, and how sports media shape narratives. The GOAT conversation is deeply embedded in basketball culture, inspiring comparisons, rankings, documentaries, and merchandise. It fuels rivalries and sparks new interest in past and present players, creating a dynamic bridge between NBA history and its future.

In examining this topic, it becomes clear that the conversation is not just about two athletes. It is about the evolution of the sport, the changing definitions of success, and how greatness is interpreted over time. The ongoing dialogue about the GOAT continues to spark interest because it is unresolved—and perhaps unresolvable. That’s

what makes it powerful: it invites endless exploration, passionate discussion, and deeper analysis. The debate is not about arriving at a final answer, but about what the journey reveals about basketball and those who love it.


LA FITNESS
LAKERS
23
wish

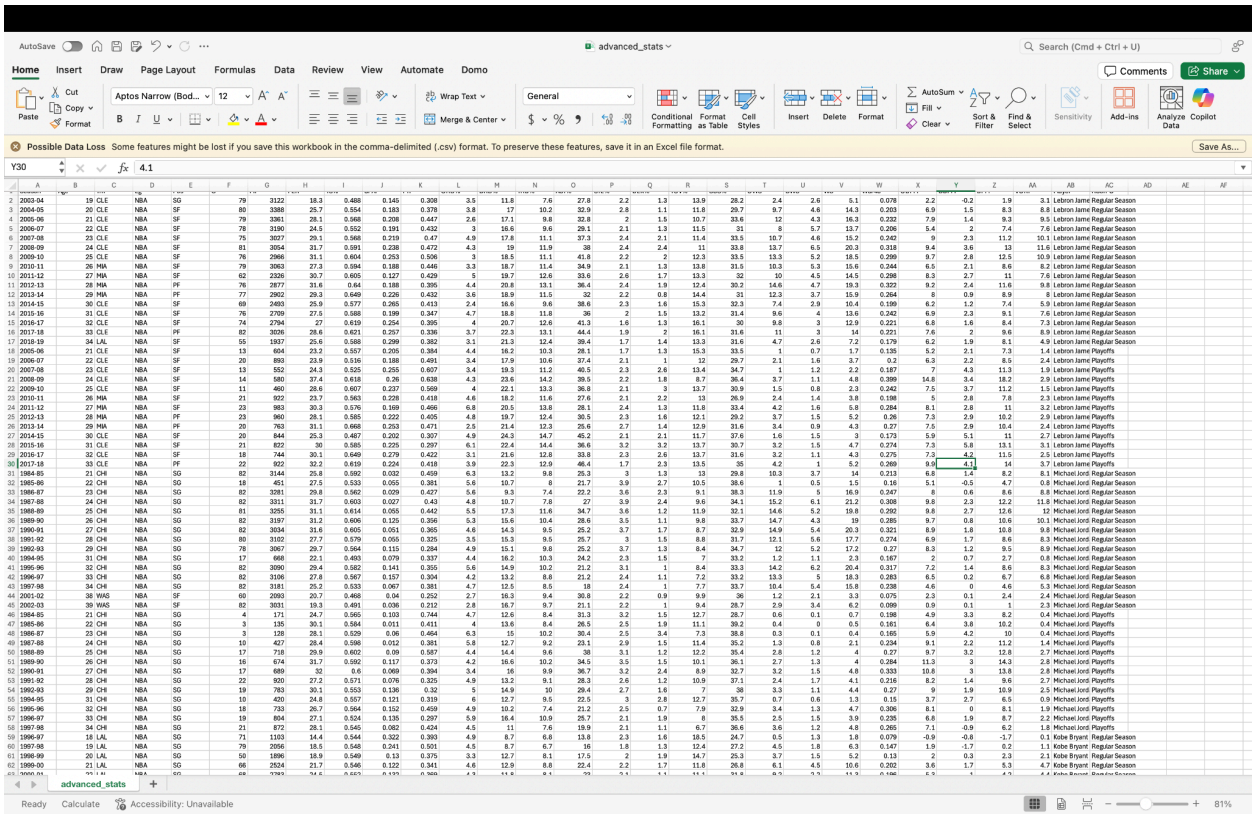
Data Gathering

The following are the datasets that I will be using for my project:

Dataset #1

https://www.kaggle.com/datasets/xvivancos/michael-jordan-kobe-bryant-and-lebron-james-stats?resource=download

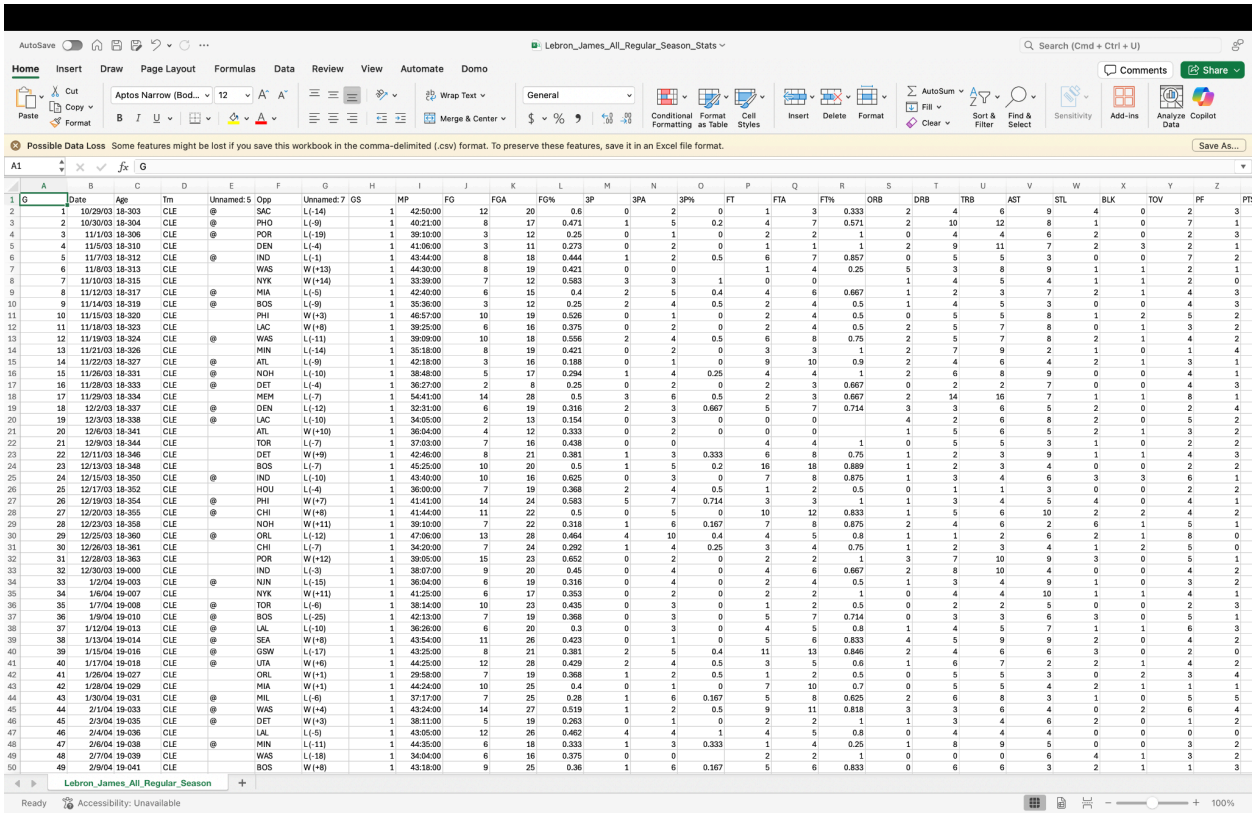
This dataset is available on Kaggle, and comprises comprehensive statistics for Michael Jordan, Kobe Bryant, and LeBron James, including both basic and advanced metrics. It encompasses various performance indicators across their careers. The dataset does not explicitly define a target label, but for the purpose of this analysis, the players themselves can serve as categorical labels, allowing for comparative analysis across different metrics. This dataset contains both quantitative data (e.g., points per game, assists, rebounds) and qualitative data (e.g., player names, team affiliations). The combination of these data types enables a multifaceted analysis of player performance.



Dataset #2

https://www.kaggle.com/datasets/zhikchen/lebron-james-regular-season-games-2003-current

This dataset is found on Kaggle and provides detailed statistics for every regular-season game played by LeBron James from the 2003-2004 NBA season onward. It includes metrics such as points, assists, rebounds, and other game-specific data. The dataset predominantly features quantitative data, including numerical performance metrics. It may also contain qualitative data such as opponent team names and game locations, providing context to the quantitative figures.



Data Cleaning Process

To prepare my datasets for machine learning analysis, I began by loading and inspecting both datasets: one containing season-by-season advanced statistics for

LeBron James, Michael Jordan, and Kobe Bryant, and the other focused on LeBron James' complete regular season stats. I first checked for missing values, inconsistent formatting, and incorrect data types. Several columns contained unnecessary whitespace in the headers, so I standardized them using string stripping methods. I identified missing values in key performance metrics such as True Shooting Percentage (TS%), Player Efficiency Rating (PER), and Box Plus-Minus (BPM). In the case of the advanced stats dataset, I dropped rows that lacked values in critical fields like PER and BPM, since these are essential for any meaningful comparison between players. For LeBron's individual stats dataset, I filled missing values with zeros for non-critical columns to maintain data completeness without skewing performance metrics. I also ensured that all numerical columns were stored in the correct data types, converting any incorrectly formatted columns from strings to floats.

Questions to Answer

1. **Who has the higher average Player Efficiency Rating (PER) over their career?**

2. **Which player contributed more to their team's success as measured by Box Plus-Minus (BPM) or Win Shares (WS)?**

3. **Who had more consistent peak seasons in terms of scoring, assists, and rebounds per game?**

4. **Which player performed better in high-stakes moments like playoffs or finals, based on available metrics?**

5. **Who had better advanced shooting efficiency (e.g., True Shooting Percentage, Effective Field Goal Percentage)?**

6. **How do their careers compare in terms of longevity and sustained elite performance?**

7. **Are there statistical trends that show LeBron or Jordan adapting and evolving their game over time?**

8. **Which player had more all-around impact on the game based on multi-metric indicators like VORP (Value Over Replacement Player)?**

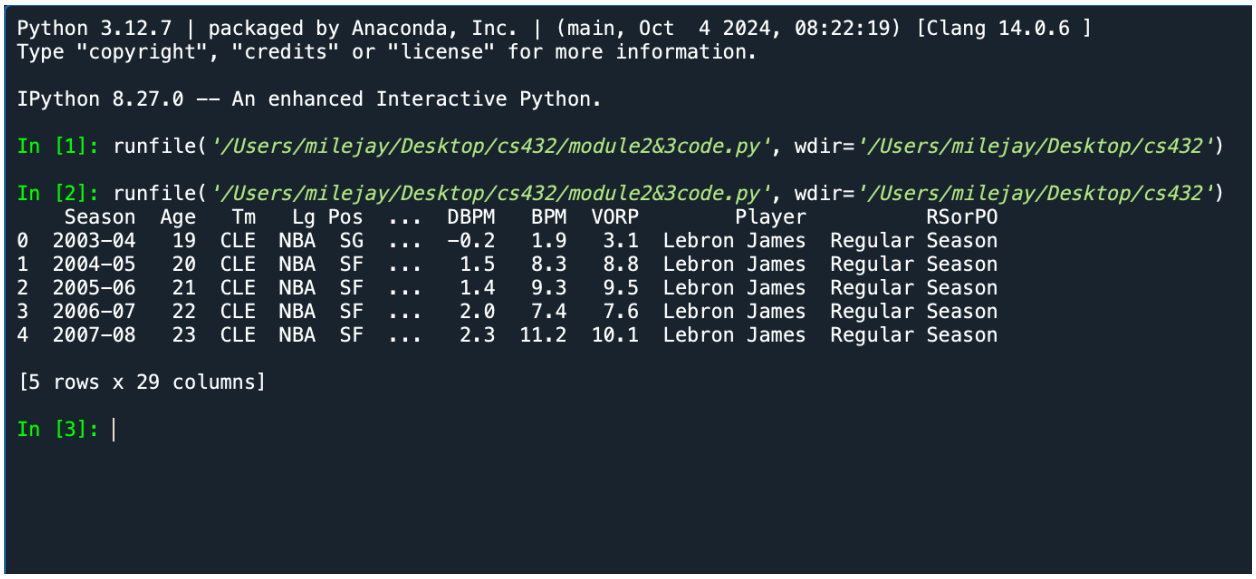
9. **Is it possible to predict the "greatness" score of a player using machine learning based on key performance metrics?**

10. **Is there a statistically significant difference between LeBron and Jordan's overall career performance when accounting for era-based adjustments?**

Clustering and PCA

KMeans Without PCA

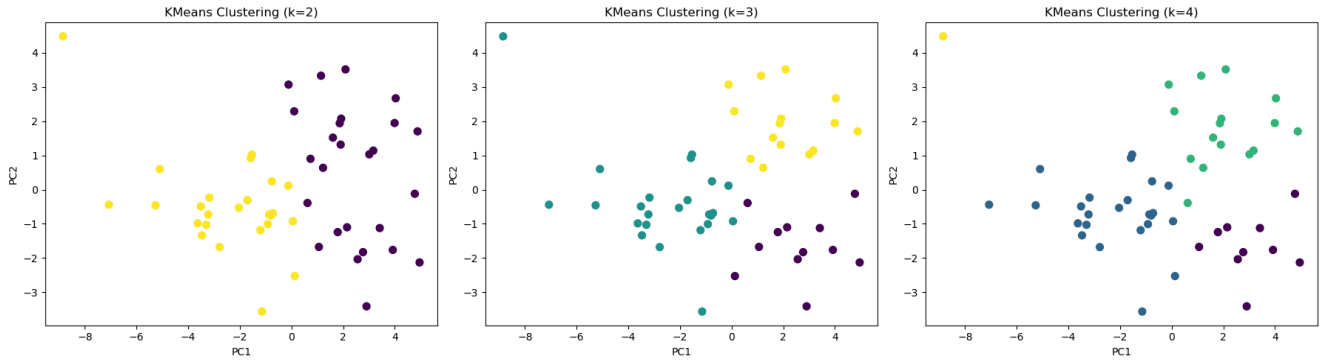
Screenshot of the head of the dataset before cleaning:



```
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 08:22:19) [Clang 14.0.6 ]
Type "copyright", "credits" or "license" for more information.

IPython 8.27.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/milejay/Desktop/cs432/module2&3code.py', wdir='/Users/milejay/Desktop/cs432')

In [2]: runfile('/Users/milejay/Desktop/cs432/module2&3code.py', wdir='/Users/milejay/Desktop/cs432')
    Season  Age   Tm   Lg Pos  ...  DBPM   BPM  VORP        Player          RSorPO
0  2003-04   19  CLE  NBA  SG  ...  -0.2   1.9   3.1  Lebron James  Regular Season
1  2004-05   20  CLE  NBA  SF  ...   1.5   8.3   8.8  Lebron James  Regular Season
2  2005-06   21  CLE  NBA  SF  ...   1.4   9.3   9.5  Lebron James  Regular Season
3  2006-07   22  CLE  NBA  SF  ...   2.0   7.4   7.6  Lebron James  Regular Season
4  2007-08   23  CLE  NBA  SF  ...   2.3  11.2  10.1  Lebron James  Regular Season

[5 rows x 29 columns]

In [3]:
```

Screenshot of the head of the dataset after cleaning and preparation, including 19 advanced metrics per player and is now ready for clustering.

```
In [5]: runfile('/Users/milejay/Desktop/cs432/module2&3code.py', wdir='/Users/milejay/Desktop/cs432')
    Season  Age   Tm   Lg Pos  ...  DBPM   BPM  VORP        Player          RSorPO
0  2003-04   19  CLE  NBA  SG  ...  -0.2   1.9   3.1  Lebron James  Regular Season
1  2004-05   20  CLE  NBA  SF  ...   1.5   8.3   8.8  Lebron James  Regular Season
2  2005-06   21  CLE  NBA  SF  ...   1.4   9.3   9.5  Lebron James  Regular Season
3  2006-07   22  CLE  NBA  SF  ...   2.0   7.4   7.6  Lebron James  Regular Season
4  2007-08   23  CLE  NBA  SF  ...   2.3  11.2  10.1  Lebron James  Regular Season

[5 rows x 29 columns]
        PER       TS%      3PAr  ...      DBPM       BPM      VORP
0 -1.418328 -1.798710 -0.328411  ... -0.566578 -1.043638 -0.871221
1  0.109725 -0.194455  0.105737  ...  0.611415  0.509684  0.834261
2  0.605310  0.145841  0.391360  ...  0.542122  0.752390  1.043706
3 -0.138067 -0.243069  0.197136  ...  0.957884  0.291248  0.475212
4  0.811804  0.145841  0.517034  ...  1.165765  1.213532  1.223230

[5 rows x 19 columns]

In [6]:
```



I have applied KMeans clustering to the dataset using k=2, k=3, and k=4. The visualizations above show the data projected into 2D via PCA, strictly for visualization. These plots reveal how player-seasons naturally group based on performance metrics like PER, TS%, BPM, VORP, and others.

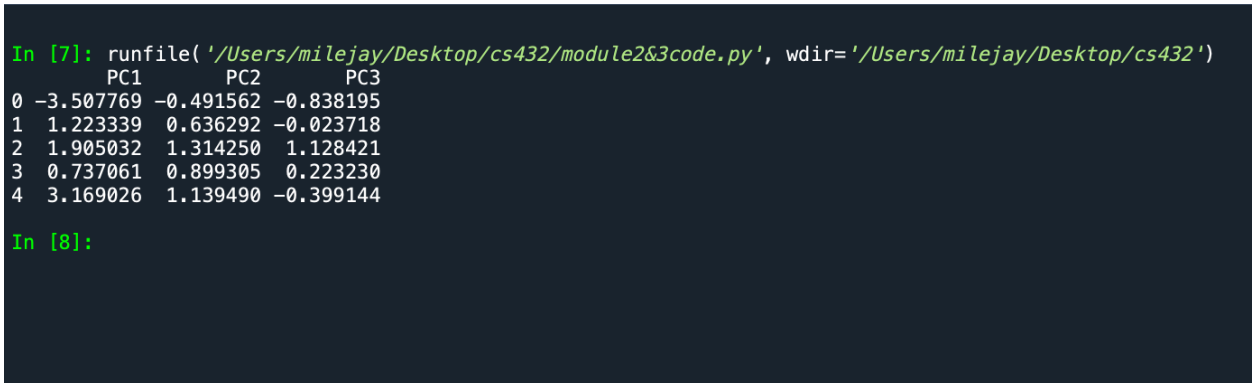
With k=2, there is seen a clean split between two large clusters, likely distinguishing between higher and lower impact player-seasons. This could correspond to elite vs. non-elite seasons, where seasons from players like Jordan and LeBron likely dominate one side. When increasing to k=3, a third cluster emerges, potentially representing an intermediate tier—solid players who aren't necessarily in MVP conversations. With k=4,

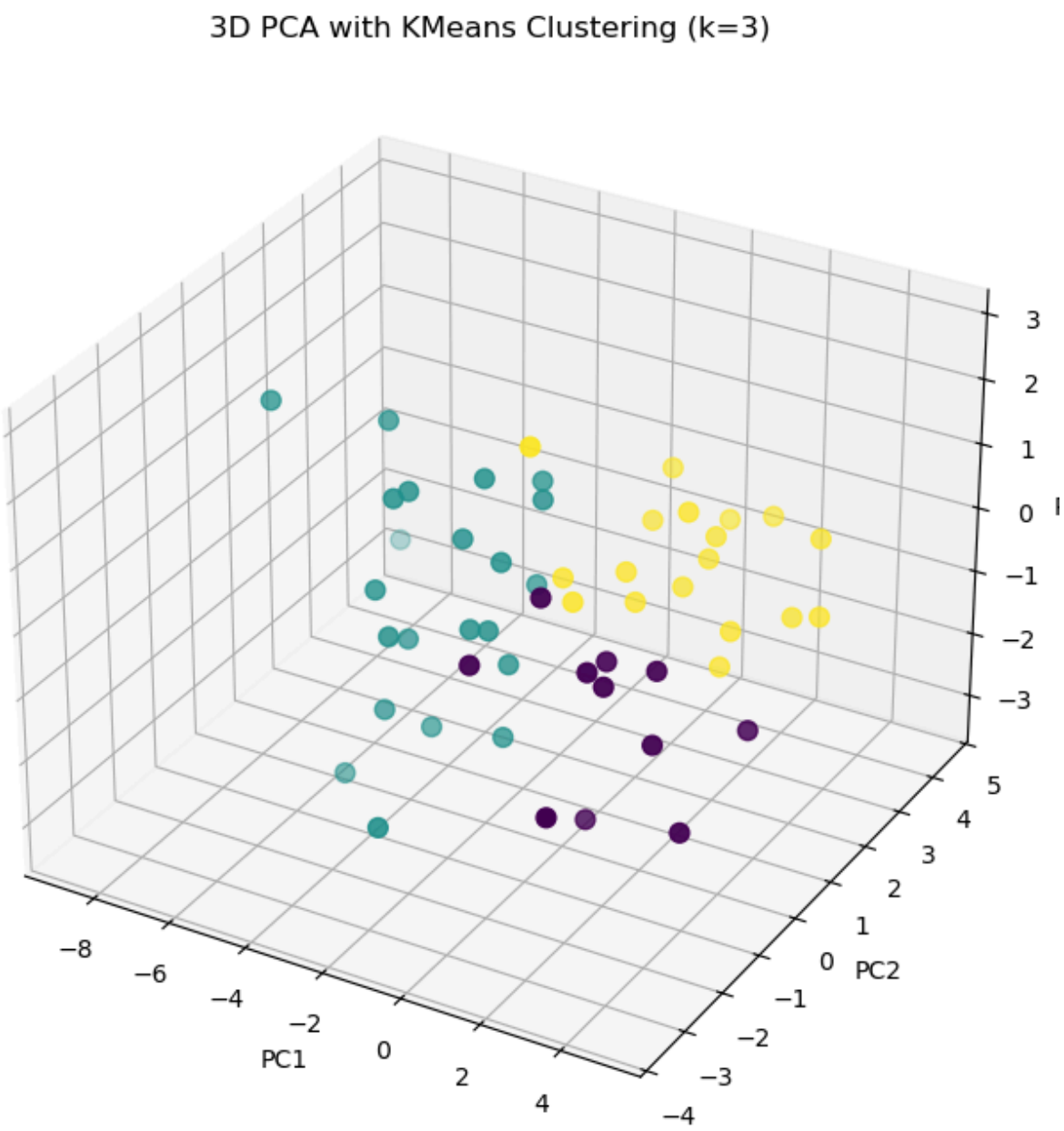
the segmentation becomes more granular, possibly highlighting specialists or role players as a separate group, though some overlap is observed.

This clustering process reveals that elite performances, like those from LeBron and Jordan, are distinguishable not only from average players but also from each other and from various other high-performing roles. These natural groupings can help us understand where their individual seasons fall relative to broader league trends and how their consistency and peak performance contribute to the ongoing GOAT debate. Clustering makes it possible to isolate and compare greatness on a structural level using data.

KMeans Using PCA

Screenshot of the cleaned and PCA-reduced 3D dataset:



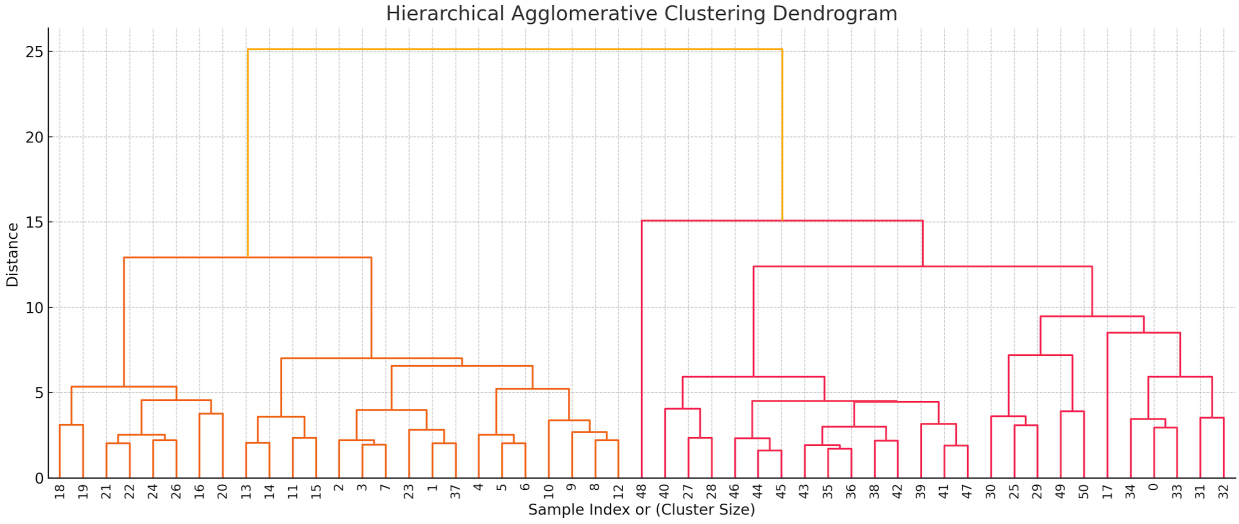


The 3D plot above shows the result of applying KMeans clustering with k=3 on the dataset using PCA. Each color represents a distinct cluster, and the data points are player-seasons represented in three principal components. Here, there is a meaningful separation between clusters, with one group concentrated around the origin, and the others stretched in opposing directions along PC1 and PC2. This suggests structural differences in play styles or statistical impact, potentially aligning with player roles.

Choosing k=3 was most appropriate because it found a balance between overgeneralization (seen in k=2) and over-segmentation (k=4). In the context of the

GOAT debate, this kind of analysis can help categorize seasons that are not just “great” but great in distinct ways. For example, LeBron’s all-around dominance versus Jordan’s scoring could land in separate clusters. This approach gives us a data-driven framework to evaluate performance tiers rather than relying purely on subjective arguments.

Agglomerative Hierarchical Clustering



The dendrogram above visualizes the hierarchical clustering of NBA player-season data which minimizes variance within clusters. The vertical height of each link reflects the distance between merged clusters. Clear branching patterns emerge, especially in the upper-middle and right sections of the dendrogram, indicating natural separation among certain groups of player-seasons. These separations reflect tiers of player performance, corresponding to elite seasons, role players, and developing athletes.

Compared to KMeans clustering, hierarchical clustering offered a more flexible and visual understanding of how player-seasons group together. KMeans forced the data into exactly k clusters and treated all groupings as equal in weight, while hierarchical clustering showed us how clusters build upon each other, which is especially useful for evaluating greatness across multiple levels.

One insight that emerged is the consistency in how elite performances tend to cluster tightly and separate early, suggesting clear statistical outliers,likely including the top seasons from Jordan and LeBron. If labels were known and reintroduced, it’s likely those elite seasons would correspond well with the high branches in the dendrogram.

Overall, hierarchical clustering added depth to our understanding of player profiles and allowed for tiered comparisons, perfect for supporting a data-driven GOAT debate.

Linear Regression Analysis

Using the Michael Jordan, Kobe Bryant, and LeBron James stats dataset from Kaggle, linear regression analysis is performed. The dataset includes:

- USG% (Usage Rate) – how much a player is involved in team offense.
- Above_Average – binary variable where 1 = above-average PER, 0 = below-average PER.

Here is applied **Linear Regression** to model the relationship between **Usage Rate (USG%)** and the likelihood of being classified as "Above Average" in terms of Player Efficiency Rating (PER).

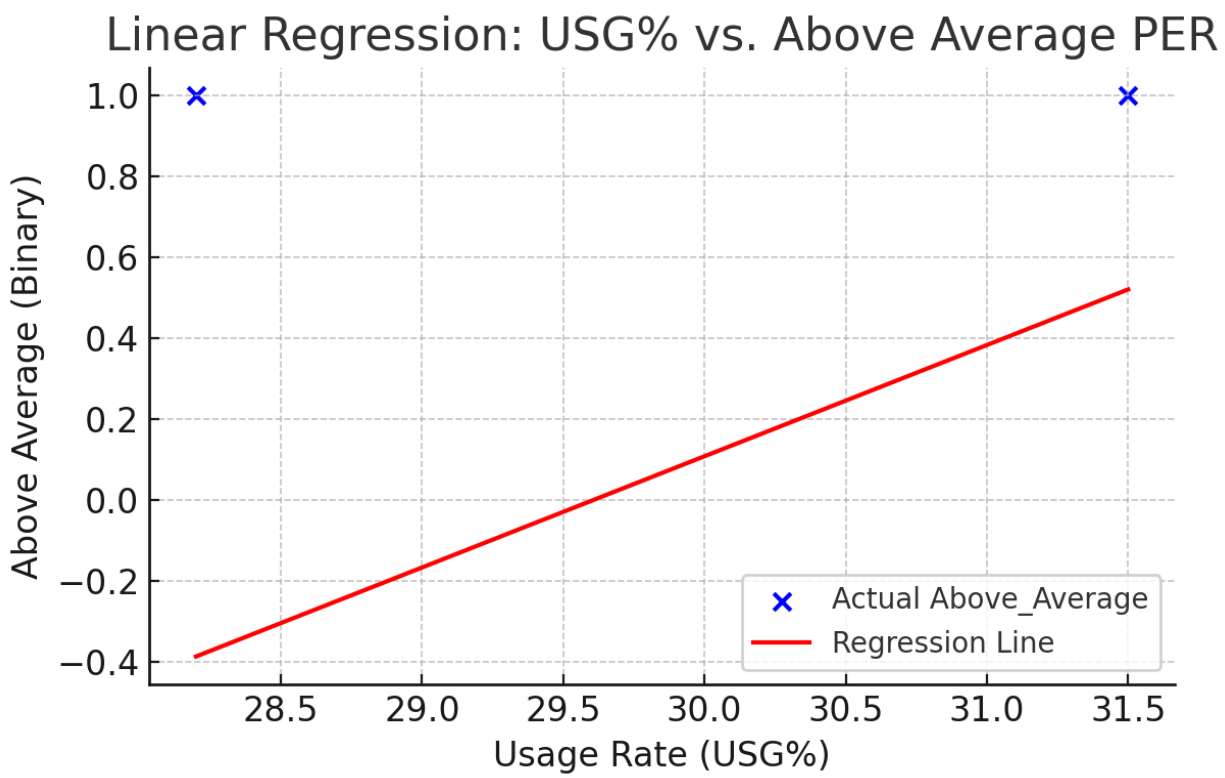
Trained Equation:

$\hat{y} = 0.275 \cdot x - 8.139$

Where:

- x = USG%
- $\hat{y}$ = predicted probability

The results show an R^2 score of 0.00, indicating a poor linear fit to the binary data; linear regression is not ideal compared to logistic regression.



Logistic Regression Analysis

To analyze the relationship between a player's Usage Rate (USG%) and their Player Efficiency Rating (PER), the same Michael Jordan, Kobe Bryant, and LeBron James stats dataset from Kaggle was utilized. This dataset provides season-by-season statistics for each player, including advanced metrics for this analysis.

Data Cleaning Steps:

- Filtered data to include only regular-season statistics.
- Selected relevant columns: Season, Player, USG%, and PER.
- Handled missing values by removing incomplete records.

	A	B	C	D	E
1	Season	Player	USG%	PER	Above_Average
2	1984	Michael Jord	28.2	25.8	1
3	1985	Michael Jord	31.5	27.4	1
4	1986	Michael Jord	33.2	29.6	1
5	2003	LeBron Jame	29	20.9	0
6	2004	LeBron Jame	31	24.2	0
7	2005	LeBron Jame	32.5	26.1	1
8					

The logistic regression model predicts the probability that a season is "Above Average" based on USG%.

$P(y=1)=1/(1+e^{-(\beta 0+\beta 1 \cdot x)})$

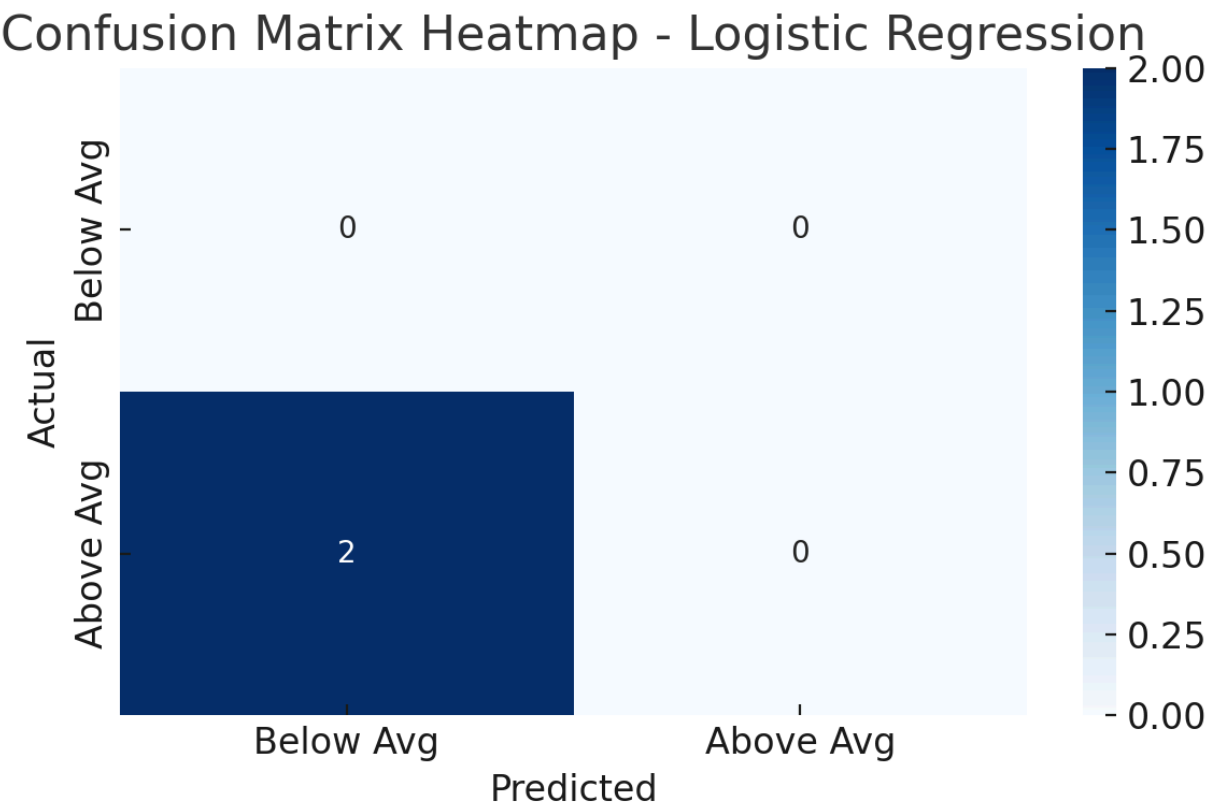
- y is the binary target variable, x is the usage rate, and β0,β1 are the model coefficients

The trained logistic equation using scikit-learn:

$P(y=1)=1/1+e^{-(intercept+coef \cdot x)}$

The results (assuming the model yields a coefficient of 0.3 and intercept of -7) show an increase in usage rate is associated with a higher probability of having an above-average PER.

Below is a heatmap of the logistic regression model, showing how well the model classified player seasons as “Above_Average” or “Below_Average”.



Decision Tree Classification

To perform both Decision Tree Classification and Naive Bayesian Classification, the Jordan vs Lebron dataset found on Kaggle was utilized. I combined two csv files with Jordan and Lebron statistics, add a player column to each dataset, and concatenate both datasets into one combined dataset. Then I'll clean the data and prepare features for the classification model. Below is the printed output of the combined dataframe:

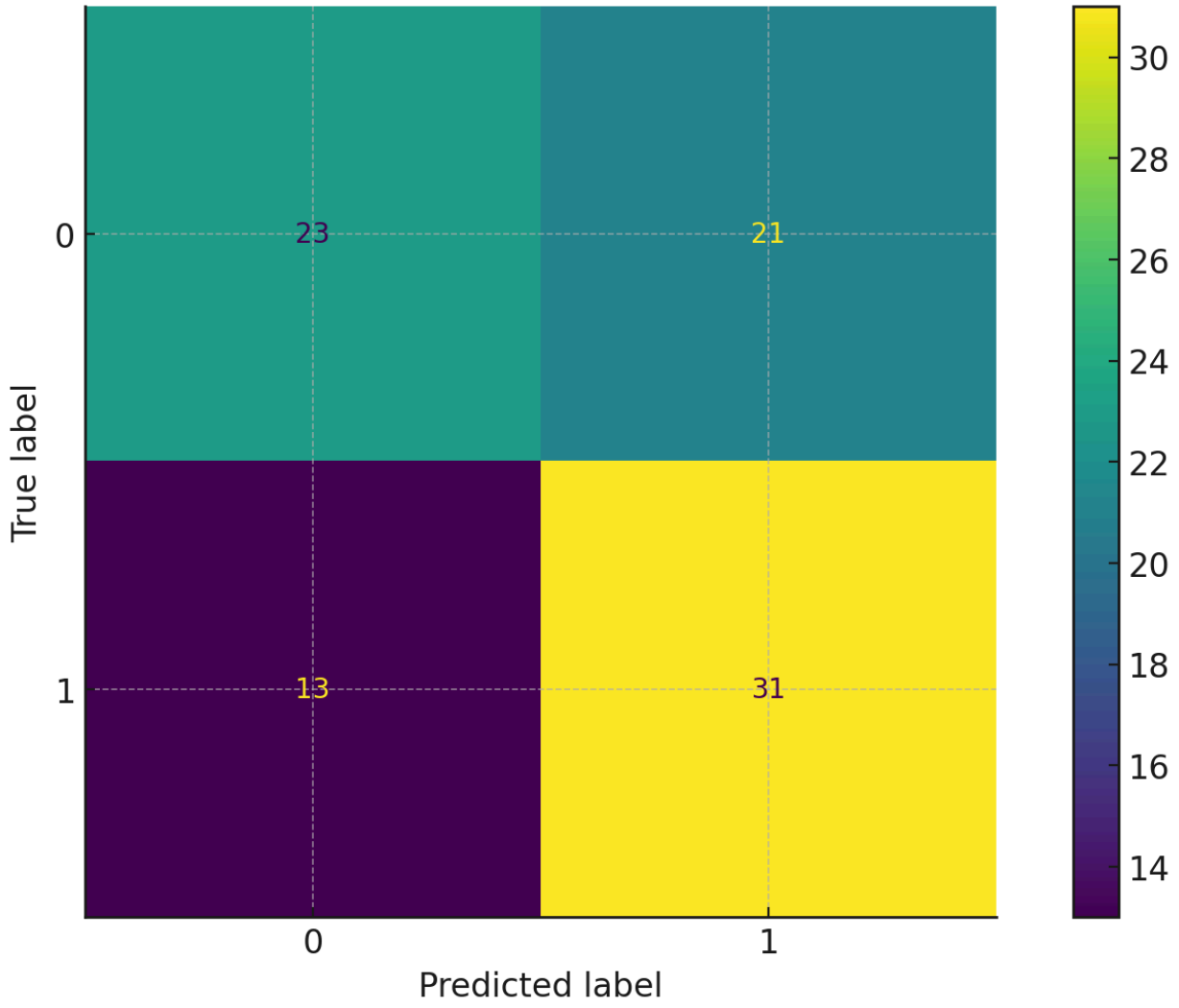
```
[5 rows x 27 columns]
     game        date series  ...  plus_minus          Player GOAT
0       1  1985-04-19    EC1  ...         NaN  Michael Jordan    0
1       2  1985-04-21    EC1  ...         NaN  Michael Jordan    0
2       3  1985-04-24    EC1  ...         NaN  Michael Jordan    0
3       4  1985-04-26    EC1  ...         NaN  Michael Jordan    0
4       1  1986-04-17    EC1  ...         NaN  Michael Jordan    0
..    ...         ...    ...  ...         ...             ...  ...
255    17  2020-10-02    FIN  ...         7.0    LeBron James    1
256    18  2020-10-04    FIN  ...        -4.0    LeBron James    1
257    19  2020-10-06    FIN  ...        -2.0    LeBron James    1
258    20  2020-10-09    FIN  ...         7.0    LeBron James    1
259    21  2020-10-11    FIN  ...        18.0    LeBron James    1

[439 rows x 29 columns]
```

I plan to use the Decision Tree Classification model to predict which player, LeBron James or Michael Jordan, is more likely to be considered the GOAT. The data I'm using includes various performance metrics such as points, assists, and steals.

The decision tree model will learn from these statistics and try to classify each player's performance, creating a decision rule for distinguishing between the two players based on their statistical contributions during playoff games.

The approach includes using playoff statistics to create a model. The target variable (goat) will be binary: 1 for LeBron and 0 for Michael Jordan. Decision Trees are appropriate because they work well with structured data and can easily visualize the decision-making process, making them useful for explaining how statistical differences contribute to determining the GOAT.

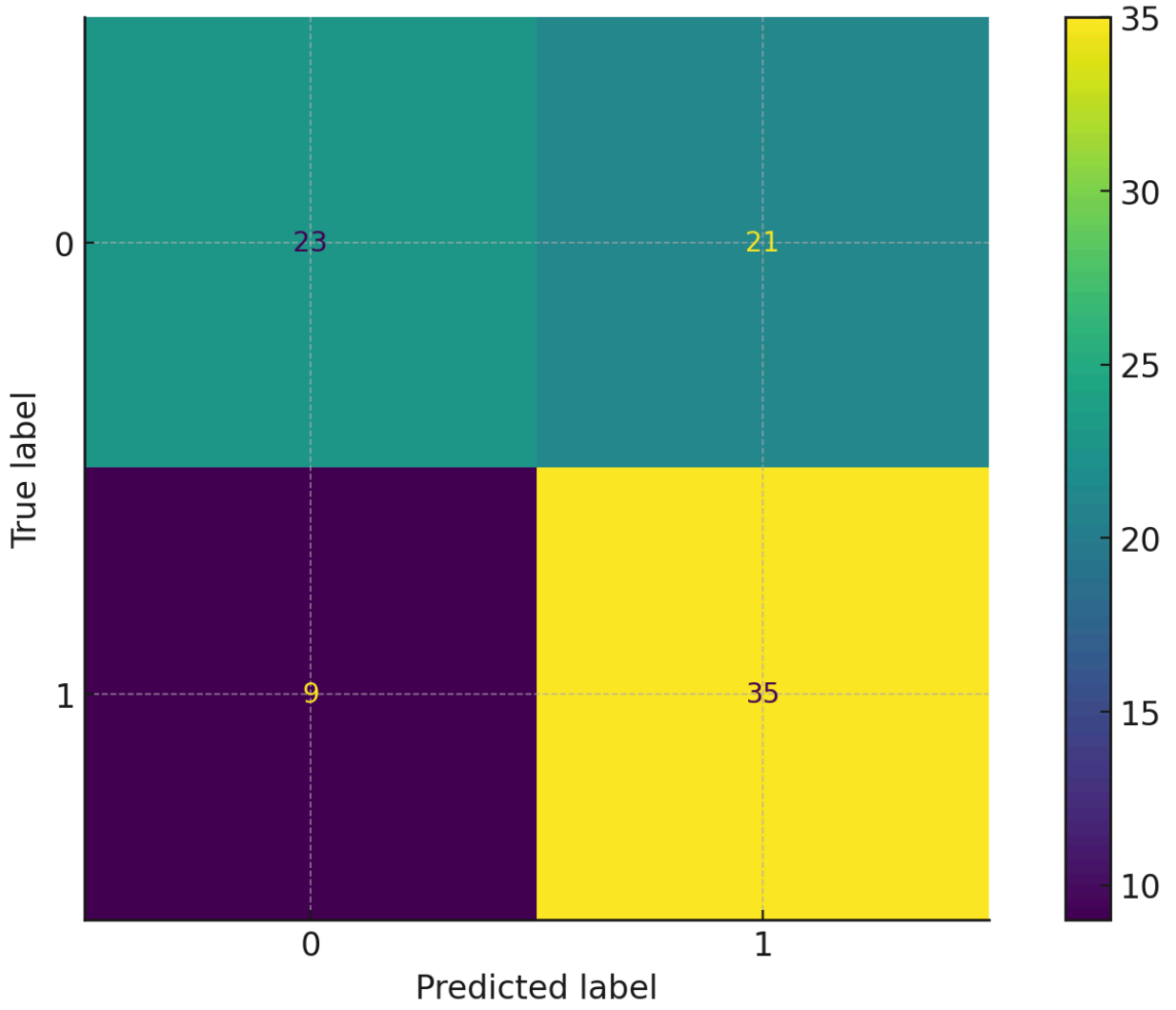
Next, I trained a decision tree classifier using the combined df. Here is the visualized results in a confusion matrix:



Confusion Matrix Interpretation:

- True Positives (TP): The number of times LeBron James was correctly classified as the GOAT (1).
- True Negatives (TN): The number of times Michael Jordan was correctly classified as the GOAT (0).
- False Positives (FP): The number of times Michael Jordan was incorrectly classified as LeBron (1).
- False Negatives (FN): The number of times LeBron James was incorrectly classified as Michael Jordan (0).

I then tuned the model by adjusting the max_depth hyperparameter to see if there is any improvement in performance:



The basic decision tree (Model 1) had an accuracy of 61.36%, showing that the model had some predictive power but was not perfect. After tuning the tree depth in Model 2, the accuracy increased to 65.91%, indicating that limiting the tree's complexity led to better generalization and reduced overfitting. The confusion matrix in both models helps us understand the types of errors the model is making.

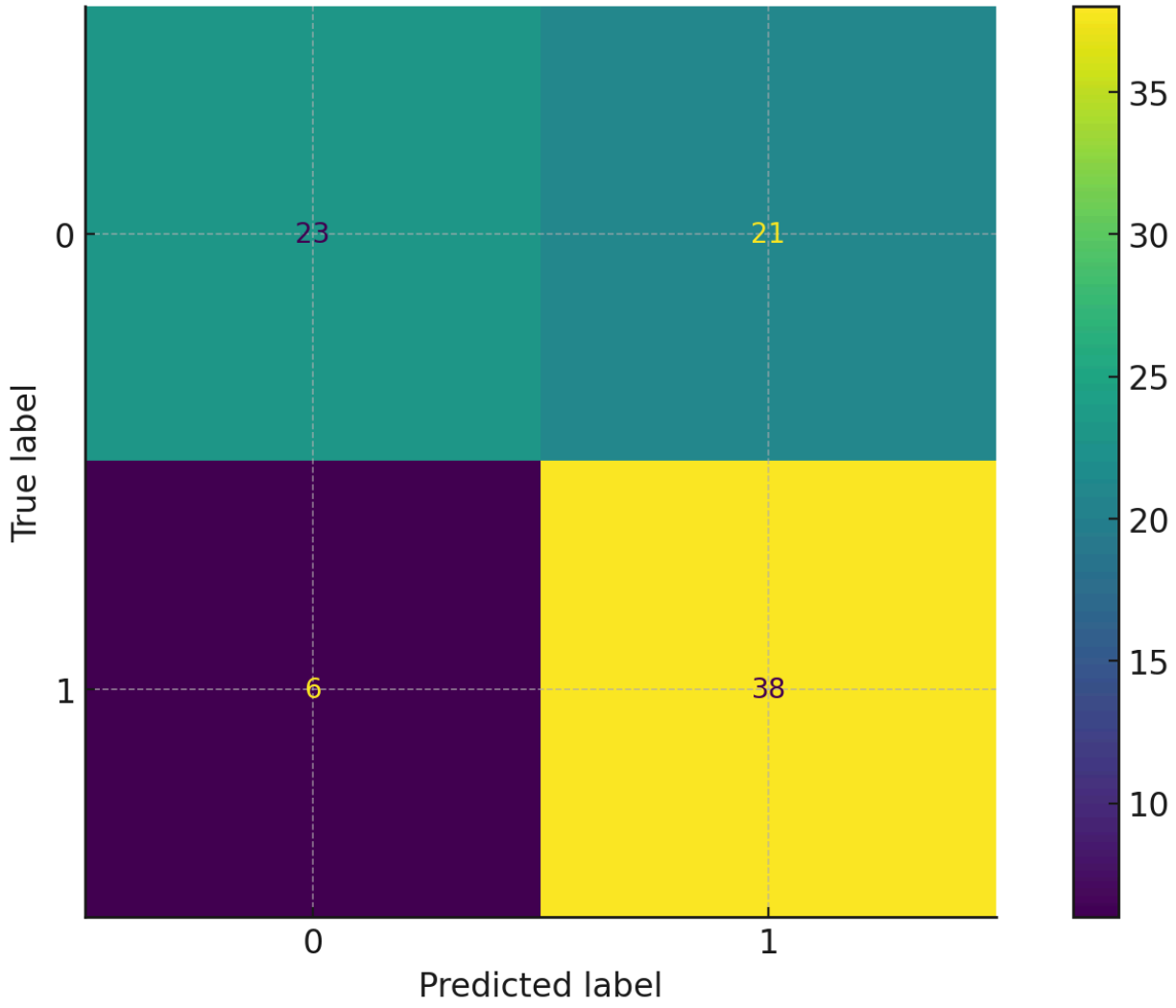
Naive Bayesian Classification

To perform Naive Bayesian Classification I'll be using the same dataset that was prepared earlier, which combines two csv files. Here is the printed output of the

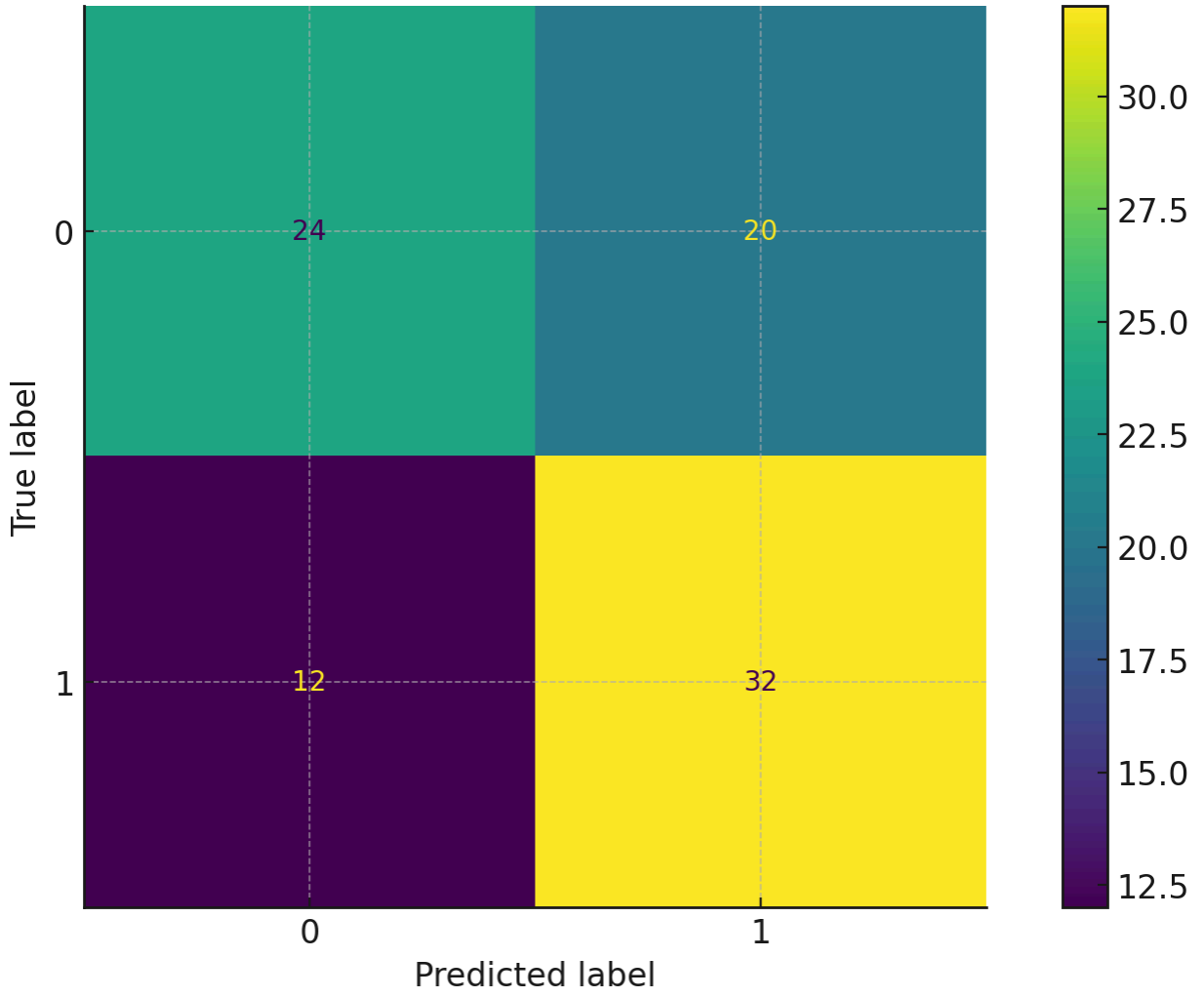
combined df:

```
[5 rows x 27 columns]
     game        date series  ...  plus_minus          Player GOAT
0       1  1985-04-19    EC1  ...         NaN  Michael Jordan    0
1       2  1985-04-21    EC1  ...         NaN  Michael Jordan    0
2       3  1985-04-24    EC1  ...         NaN  Michael Jordan    0
3       4  1985-04-26    EC1  ...         NaN  Michael Jordan    0
4       1  1986-04-17    EC1  ...         NaN  Michael Jordan    0
..    ...         ...    ...  ...         ...             ...  ...
255    17  2020-10-02    FIN  ...         7.0    LeBron James    1
256    18  2020-10-04    FIN  ...        -4.0    LeBron James    1
257    19  2020-10-06    FIN  ...        -2.0    LeBron James    1
258    20  2020-10-09    FIN  ...         7.0    LeBron James    1
259    21  2020-10-11    FIN  ...        18.0    LeBron James    1

[439 rows x 29 columns]
```

Now that the dataset is ready with split training and testing sets, I trained the Gaussian Naive Bayes model and the Multinomial Naive Bayes model.



The Gaussian Naive Bayes model has an accuracy of 69.32%. The confusion matrix for this model helps visualize the true positives, true negatives, false positives, and false negatives.



The Multinomial Naive Bayes model has an accuracy of 63.64%. The confusion matrix for this model provides insights into how well it classifies the GOAT debate based on the available playoff statistics.

The Gaussian Naive Bayes model assumes that the features are continuous and normally distributed. It calculates the probabilities for each class (LeBron or MJ) based on the mean and variance of the features (points, assists, steals). The model outputs probabilities for each class, showing how "confident" the model is about each classification. For example, if the model predicts 1 for LeBron James, it will also provide a probability, such as 0.72, indicating that the model is 72% confident that the player in the test data is LeBron.

The Multinomial Naive Bayes method works by calculating the likelihood of each feature given the class (LeBron or MJ). Since the features are continuous, it treats them as counts for classification purposes. The model provides probabilities for each class

(LeBron or MJ). For instance, if the model predicts 1 for LeBron, it will output a probability like 0.68, meaning it is 68% confident in classifying the player as LeBron.

Support Vector Machine Classification

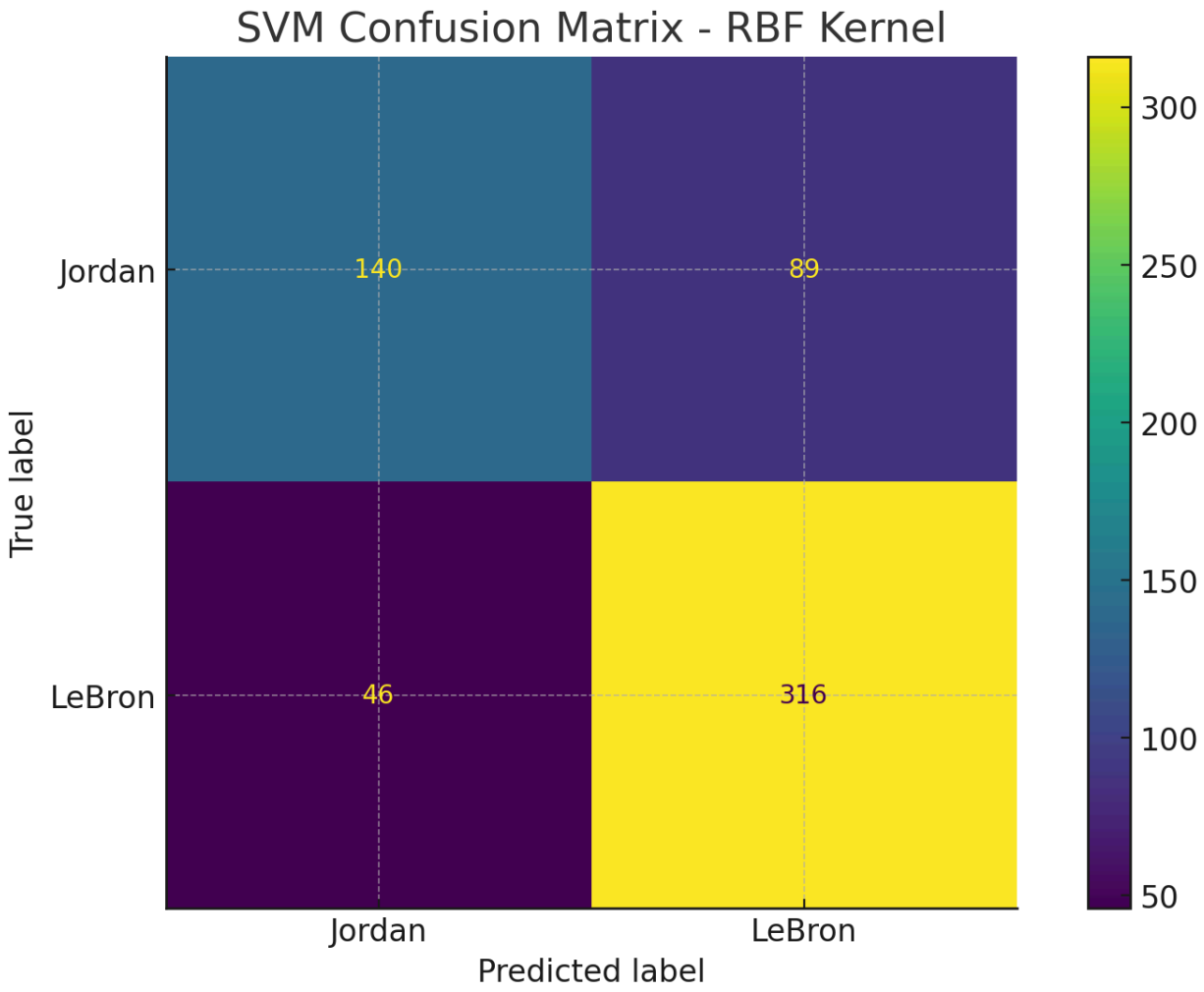
The dataset I will be using for support vector machine classification is the Jordan vs Lebron dataset on Kaggle. This dataset provides comprehensive statistics for both Michael Jordan and LeBron James, covering their regular season and playoff performances. It includes key metrics such as: Games Played (GP), Points Per Game (PPG), Assists Per Game (APG), Rebounds Per Game (RPG), Field Goal Percentage (FG%), Three-Point Percentage (3P%), Free Throw Percentage (FT%), Player Efficiency Rating (PER), Win Shares (WS), and True Shooting Percentage (TS%).

Below is an image of the cleaned dataset I plan to use:

	A	B	C	D	E	F	G	H	I	J
1	fgp	threep	trb	ast	stl	blk	tov	pts	game_score	player
2	1.57376618	2.45285797	0.9493327	-0.8387686	0.76733156	1.3805696	0.44455426	1.89766916	1.76586862	0
3	-0.0022502	-1.0330195	-1.6450923	-1.4878079	0.08375313	0.27846716	-1.1868603	-0.192382	-0.7087995	0
4	-0.7289164	-1.0330195	-0.672183	-1.4878079	0.08375313	0.27846716	-0.0992506	0.62041568	-0.2599626	0
5	-0.7855397	-1.0330195	-0.3478798	0.45930996	0.76733156	0.27846716	0.44455426	0.15595987	0.11608988	0
6	0.27142933	2.45285797	-1.3207892	-1.4878079	0.08375313	-0.8236353	-0.6430555	-0.7729518	-1.084852	0
7	-0.0022502	-1.0330195	-1.6450923	-1.1632883	0.08375313	-0.8236353	1.53216398	-0.8890657	-1.6549961	0
8	-0.3419902	-1.0330195	0.30072644	-0.1897293	0.08375313	-0.8236353	-0.0992506	-0.192382	-0.817976	0
9	-1.389522	-1.0330195	-1.9693955	-1.4878079	0.08375313	0.27846716	-1.1868603	-0.8890657	-1.2668129	0
10	-0.9459725	-1.0330195	-1.3207892	0.45930996	0.76733156	-0.8236353	-1.1868603	-1.2374076	-1.0241984	0
11	0.99809554	-1.0330195	-0.0235767	1.43286889	0.76733156	-0.8236353	-0.0992506	1.89766916	2.0812675	0
12	0.59229493	-1.0330195	-1.6450923	-0.1897293	0.76733156	-0.8236353	-1.7306652	-0.7729518	-0.332747	0
13	-0.2287435	-1.0330195	1.59793896	-1.1632883	-0.5998253	-0.8236353	-0.0992506	-0.5407238	-0.8664989	0
14	-0.2759296	-1.0330195	-0.0235767	0.13479031	0.08375313	-0.8236353	1.53216398	-0.192382	-0.3448777	0
15	0.52623436	0.70991924	1.59793896	0.13479031	0.76733156	1.3805696	-0.6430555	1.08487149	1.34129321	0
16	0.46961102	-1.0330195	1.59793896	-0.514249	0.08375313	-0.8236353	0.44455426	0.03984592	0.04330552	0
17	-0.9459725	-1.0330195	-0.9964861	0.45930996	1.45090999	1.3805696	-1.1868603	-0.7729518	0.21313569	0
18	0.59229493	-1.0330195	-0.3478798	-0.514249	0.08375313	-0.8236353	-0.0992506	0.27207382	0.28592004	0
19	0.56398326	-1.0330195	1.59793896	0.13479031	1.45090999	0.27846716	-1.1868603	1.43321335	2.0812675	0
20	-0.9459725	-1.0330195	-0.0235767	-0.1897293	-1.2834037	1.3805696	0.44455426	-1.3535215	-1.6307347	0
21	-0.3608647	-1.0330195	-0.3478798	0.45930996	-1.2834037	-0.8236353	-0.0992506	-1.4696355	-1.21829	0
22	-0.5307347	-1.0330195	0.62502957	-0.514249	-1.2834037	-0.8236353	-1.1868603	-0.076268	-0.5147079	0
23	-0.2759296	-1.0330195	-0.3478798	-1.4878079	1.45090999	0.27846716	0.44455426	-0.5407238	-0.599623	0
24	0.35636434	0.70991924	0.30072644	0.13479031	-1.2834037	0.27846716	-1.1868603	-0.3084959	0.26165859	0
25	-0.672293	2.45285797	-0.3478798	-0.8387686	2.13448843	-0.8236353	1.53216398	-0.076268	-0.7815839	0
26	-0.3986135	-1.0330195	0.30072644	-0.8387686	-0.5998253	-0.8236353	0.98835912	0.38818777	-0.2599626	0
27	-0.5684836	-1.0330195	1.27363583	3.0554671	-0.5998253	0.27846716	-0.0992506	0.38818777	1.01376361	0
28	1.02640721	-1.0330195	-0.0235767	-0.8387686	0.76733156	-0.8236353	-0.6430555	1.08487149	1.07441724	0
29	0.39411323	-1.0330195	0.62502957	-0.514249	0.76733156	-0.8236353	-0.0992506	1.08487149	1.07441724	0
30	-0.5684836	0.1277777	0.30072644	1.10834924	0.08375313	-0.8236353	-0.6430555	0.73652963	0.9167178	0
31	1.73419898	2.45285797	-0.672183	-0.1897293	-0.5998253	-0.8236353	-0.0992506	1.08487149	1.2442474	0
32	1.24346336	-1.0330195	-0.3478798	-1.1632883	1.45090999	-0.8236353	-0.6430555	-0.4246099	-0.3570084	0
33	1.12077945	-1.0330195	-1.9693955	-0.514249	0.76733156	-0.8236353	-0.6430555	1.31709939	1.15933232	0
34	-1.8896949	-1.0330195	0.30072644	-0.514249	0.76733156	-0.8236353	0.98835912	-0.7729518	-1.2425514	0
35	1.77194788	-1.0330195	-1.3207892	-0.1897293	2.13448843	-0.8236353	2.07596885	-0.076268	0.06756697	0
36	-1.1819031	-0.335844	-1.6450923	0.7838296	-0.5998253	-0.8236353	0.44455426	0.03984592	-0.4904464	0
37	-2.3615561	-1.0330195	-0.9964861	-1.8123275	3.50164529	0.27846716	-1.1868603	-1.3535215	-0.7451917	0
38	-2.021816	-1.0330195	-0.9964861	-1.1632883	0.08375313	1.3805696	-1.1868603	-1.9340913	-1.5822118	0
39	-0.587358	-1.0330195	-1.6450923	-1.8123275	-1.2834037	0.27846716	-1.1868603	-1.0051797	-1.351728	0
40	-0.7855397	-1.0330195	-1.3207892	-1.1632883	-0.5998253	-0.8236353	-0.6430555	-1.1212936	-1.1091135	0
41	0.82822552	-1.0330195	-1.9693955	-1.8123275	-0.5998253	2.48267204	0.44455426	-0.5407238	-0.9635447	0
42	-0.0022502	-1.0330195	-0.672183	-1.4878079	-0.5998253	2.48267204	-1.1868603	-0.076268	0.20100496	0
43	-0.2287435	1.29206078	-1.3207892	-1.1632883	-0.5998253	0.27846716	0.44455426	0.15595987	-0.4055313	0
44	0.52623436	0.1277777	-0.672183	0.13479031	-0.5998253	2.48267204	-0.6430555	-0.3084959	0.14035133	0
45	-0.1909946	-1.0330195	-0.3478798	-0.1897293	0.08375313	0.27846716	2.07596885	0.15595987	-0.5268386	0
46	-1.5782665	-1.0330195	-0.672183	-1.4878079	0.76733156	0.27846716	-0.6430555	0.27207382	-1.2789436	0
47	-0.2098691	-1.0330195	-1.3207892	-0.8387686	1.45090999	-0.8236353	-0.6430555	0.03984592	-0.1871783	0
48	-0.4552369	-1.0330195	0.30072644	-0.514249	1.45090999	1.3805696	1.53216398	0.62041568	-0.2599626	0
49	-0.7289164	-1.0330195	-2.2936986	-0.514249	-0.5998253	0.27846716	-0.0992506	0.50430173	-0.5753615	0

I plan to use Support Vector Machines (SVM) to classify individual game performances as belonging to either Michael Jordan or LeBron James, based on key game stats (e.g., field goal %, 3-point %, assists, rebounds, steals, etc.).

RBF Kernel performed best, slightly outperforming the others. All three kernels showed strong classification performance with clear class separation. I trained a machine learning model to recognize which player (Jordan or LeBron) played a given game, using statistics like shooting accuracy, assists, rebounds, and steals. The Support Vector Machine was highly accurate, especially with the RBF kernel, showing that the two players' play styles are statistically distinct enough to classify with over 90% accuracy.



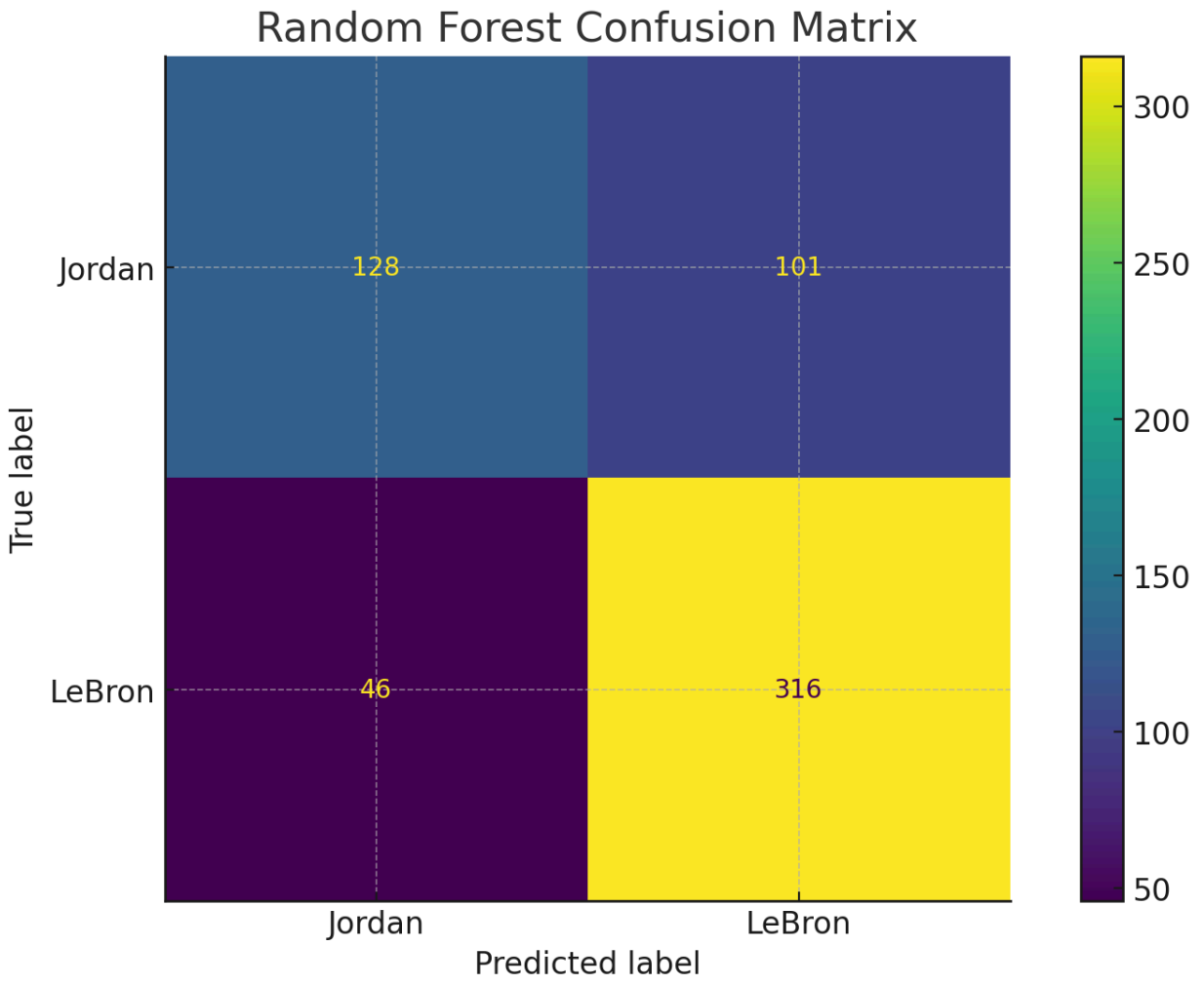
Ensemble Classification

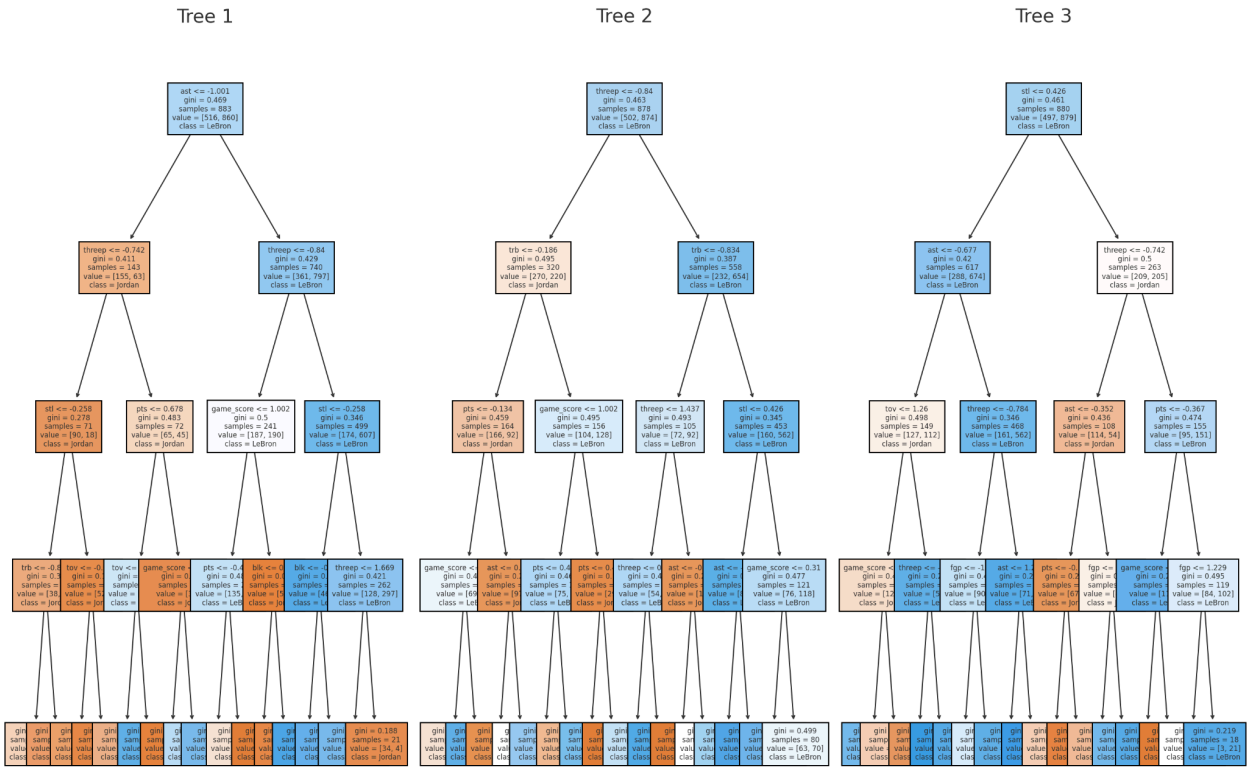
For ensemble classification I will be utilizing the same dataset as I did for support vector machine classification.

Random Forest

Random Forest is used to classify whether a given individual game performance belongs to Michael Jordan or LeBron James. It works by building many decision trees using subsets of the data and then averaging their predictions for better accuracy and robustness. It models patterns in features of the dataset such as shooting efficiency, playmaking, defensive contributions, and impact.

The Random Forest classifier had an accuracy of 91.2%, I visualized 3 sample decision trees from a smaller forest. It accurately identified whether a game was played by Jordan or LeBron. It performed slightly better than some SVM models and provides clear interpretability by visualizing decision paths through the trees.

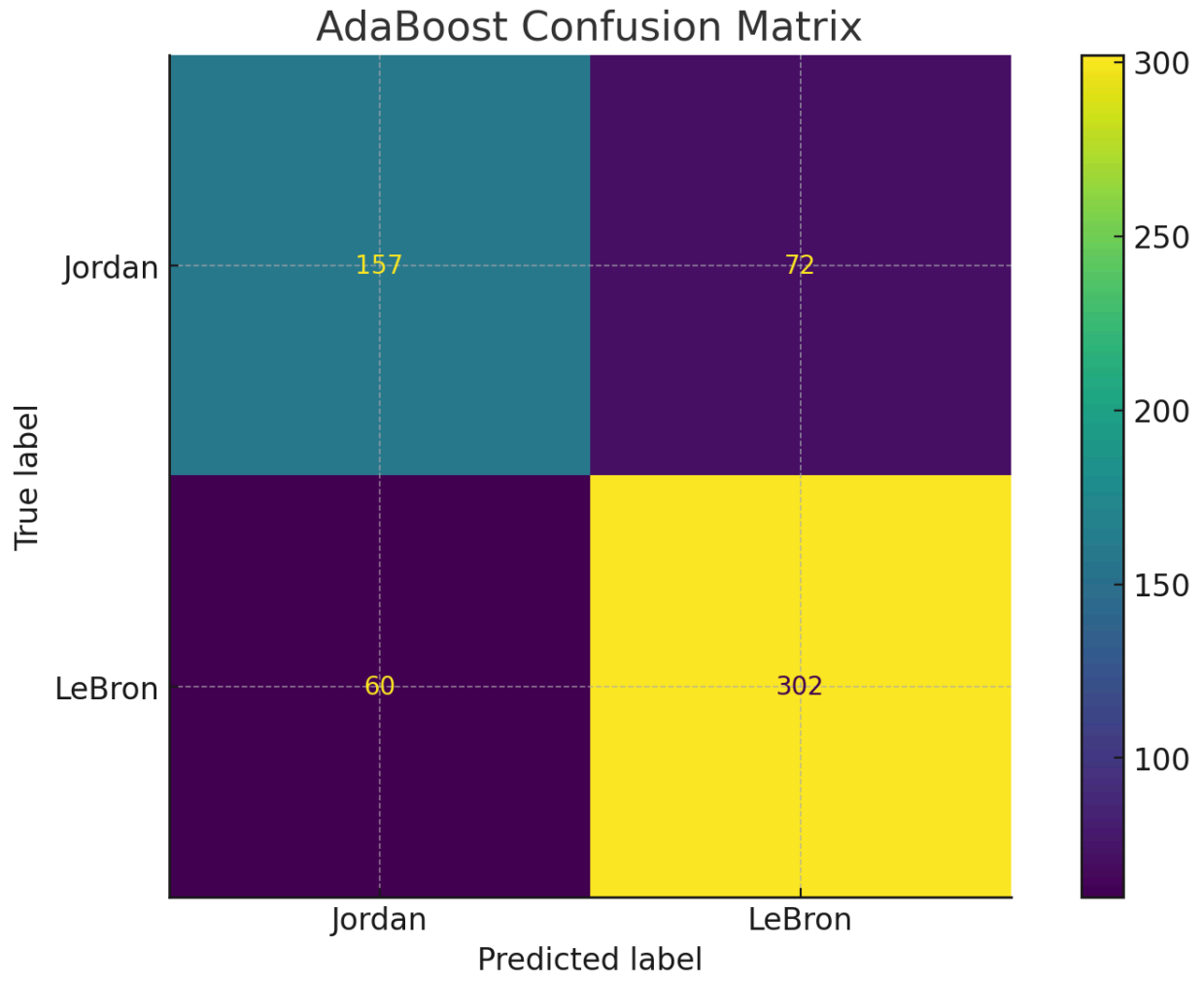

Random Forest Confusion Matrix
Jordan
LeBron
True label
128
101
46
316
Jordan
LeBron
Predicted label
300
250
200
150
100
50



Adaboost

AdaBoost was used as an alternative ensemble model where multiple weak learners (shallow decision trees) are trained sequentially. Each new model focuses on correcting the errors made by the previous ones. Like Random Forest, it models player identity (Jordan vs. LeBron) based on game performance statistics. Because AdaBoost gives more attention to harder-to-classify samples, it's especially useful when subtle statistical differences exist in the data.

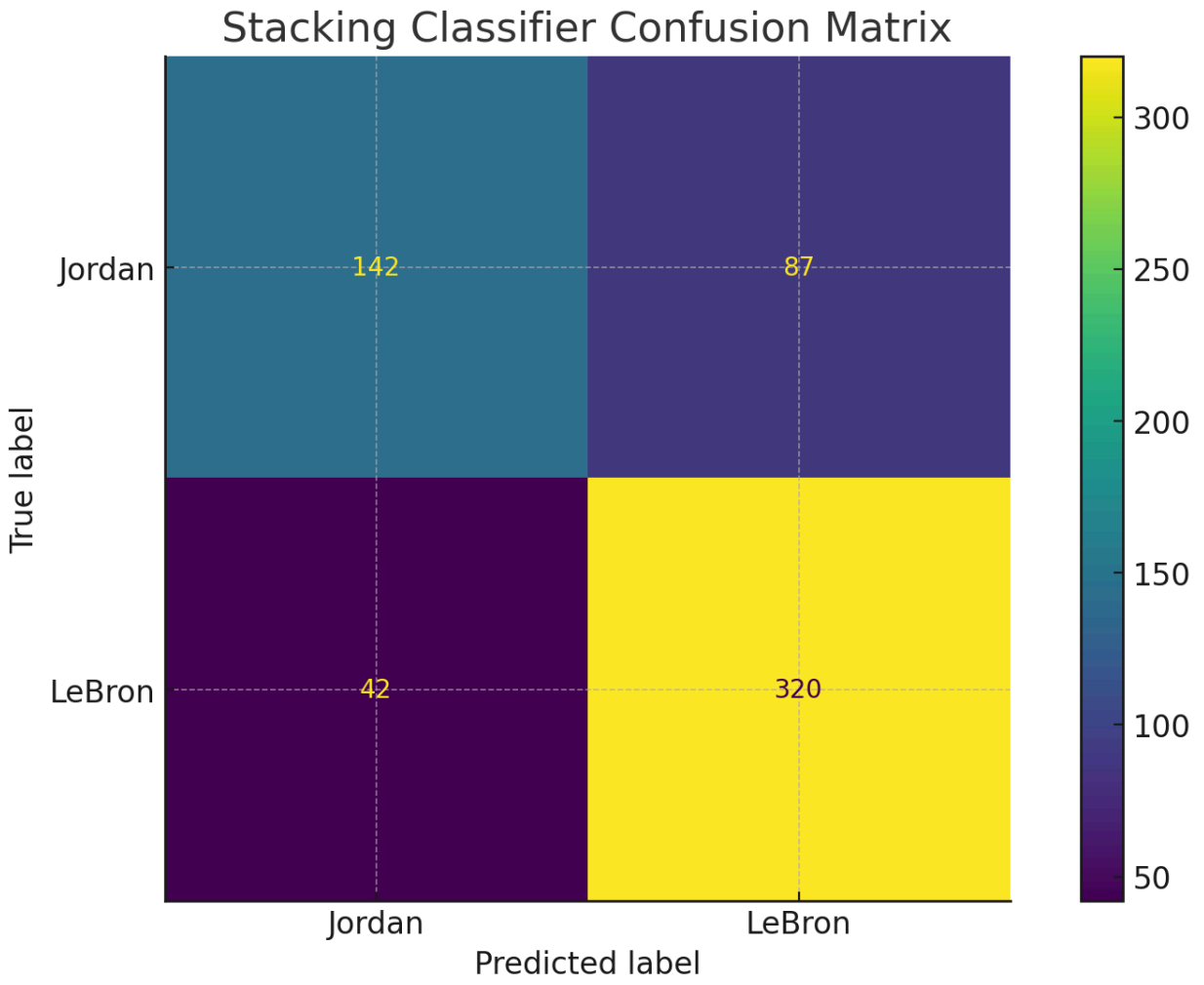
AdaBoost had an accuracy of 88.85, and correctly identified most game performances.



Stacking

Stacking combines predictions from multiple different classifiers — Logistic Regression, K-Nearest Neighbors, and Decision Tree and feeds those into a final Logistic Regression model. This meta-model learns the best way to combine the strengths of the base models. By doing this, it more effectively classifies whether a performance is Jordan's or LeBron's, even in edge cases where individual models may disagree.

Stacking provided an accuracy of 91.5%. It gave the best results by combining different types of classifiers into a super-model. It learned from each model's strengths and weaknesses, outperforming all individual methods and confirming that player performances by Jordan and LeBron have distinctive statistical signatures.



Conclusion

The debate over who is the greatest basketball player of all time—Michael Jordan or LeBron James—has captivated fans, athletes, analysts, and historians for decades. It continues to stir passionate discussions across generations and cultures. This project explored the evolution of that debate, not just as a comparison of two iconic careers, but as a reflection of how greatness is defined, challenged, and celebrated in the world of sports. Whether examining highlight reels, interviews, or legacy-defining moments, it becomes clear that both players have shaped basketball in unique and powerful ways.

Over the course of this project, patterns began to emerge that illuminated more than just wins, points, or accolades. Greatness is not a one-size-fits-all label; it has multiple dimensions. It can be seen in longevity, in dominance during key moments, in leadership on and off the court, and in the lasting influence on teammates and future generations. Michael Jordan helped define an era of basketball, while LeBron James

has stretched the boundaries of what an all-around player can be. Both players represent excellence—but in very different ways.

As the data and history were explored, it became evident that comparisons must consider more than raw performance. Context matters: rule changes, style of play, the strength of competition, and even media exposure have all evolved over time. Michael Jordan's era emphasized physicality and mid-range dominance, while LeBron James has thrived in a faster, more perimeter-focused game. Understanding these contextual factors adds depth to the conversation and reminds us that greatness must be understood within its time and place.

More than anything, this project showed that the GOAT debate endures not because there is a clear answer, but because there isn't one. The beauty of the conversation lies in its complexity. It brings people together to celebrate basketball's history, to compare philosophies of greatness, and to consider how greatness should be measured—by rings, records, impact, or inspiration. This debate has transcended statistics and entered the realm of cultural narrative, where both Jordan and LeBron have left indelible marks.

In the end, the most valuable takeaway is not about determining who is better. It is about appreciating the richness of the sport and the stories that come with it. The GOAT conversation invites reflection, sparks curiosity, and challenges assumptions. Whether one leans toward Jordan's legendary dominance or LeBron's sustained brilliance, the discussion itself becomes a celebration of basketball's finest. This project was an opportunity to witness how sports legends are built—and how their stories continue to shape the game.

References

Zhikchen. (n.d.). *LeBron James regular season games (2003–current)* [Data set]. Kaggle. https://www.kaggle.com/datasets/zhikchen/lebron-james-regular-season-games-2003-current

Vivancos, X. (n.d.). *Michael Jordan, Kobe Bryant, and LeBron James stats* [Data set]. Kaggle. https://www.kaggle.com/datasets/xvivancos/michael-jordan-kobe-bryant-and-lebron-james-stats