Identifying Top Tier Elite NBA Players using Unsupervised Machine Learning Algorithms

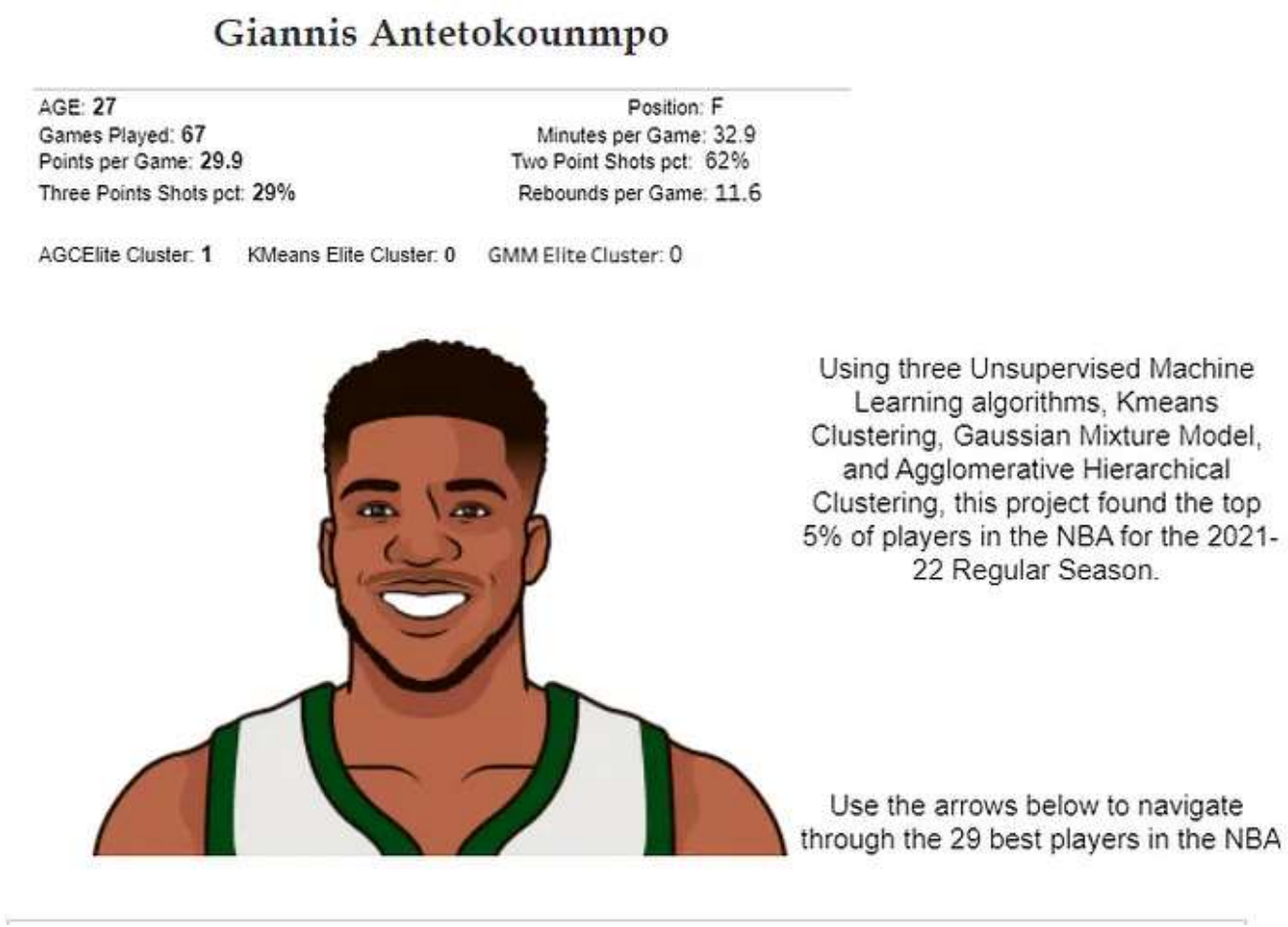
John K. Hancock

13 min read · Apr 18

KMeans Clustering, Gaussian Mixture Models, Agglomerative Clustering

by

John K. Hancock, MSDS (*jkhancock@gmail.com*)



Giannis Antetokounmpo

Every year, NBA journalists vote on the best 15 players in the NBA. The NBA elite. The vote is spread across First, Second, and Third All-NBA teams. Each team must have 2 guards and forwards and a center. These are the historical and prototypical positions on a NBA team even though in the modern game player positions are more fluid.

All-NBA team votes are critical because they can impact how much a player can be paid. Players who make the All-NBA team can get higher maximum value contracts. Thus, the possibility of subjectiveness and biases may come into play. A voter may vote for one player over another if it means that the player would get a better contract. This subjectiveness leads to contentious debates where we argue which player deserves to be voted on the team. What if there was a better method to decide which players are elite. A method purely based on the statistical performance of the players no matter their position.

Unsupervised machine learning, an algorithm that find patterns in unlabeled data, may provide us with sucg a method. Clustering data into similar groups is the most common usage. Using this method, players would be partitioned into clusters based on how similar they are to each other within a cluster and dissimilar to other clusters. Ostensibly, creating clusters of NBA players should put elite players into their own cluster providing an unbiased selection of the best players.

There are several different types of Unsupervised Machine learning clustering algorithms. They can be grouped based on their clustering methodology. For this project, I focused on three primary types:

1. Centroid based clustering: KMeans Clustering
2. Distribution based clustering: Gaussian Mixture Models
3. Hierarchical clustering: Agglomerative Hierarchial Clustering

Centroid based clustering (**"KMeans Cluster"**) groups data into clusters based on their closeness to central, representative data point called a centroid. The centroid is calculated as the average or mean of the data points within a cluster. One of the more prominent algorithms.

Distribution based clustering (**"GMM Cluster"**) groups data into clusters based on the underlying distribution of the data. Data points within the same cluster are expected to have similar probability distributions whereas data points from different clusters are expected to have different distributions. GMM models the data points as a mixture of Gaussian distributions, where each Gaussian component represents a cluster. The algorithm estimates the parameters of the Gaussian components, such as the mean, covariance, and mixture weights, to identify the clusters in the data.

Hierarchical clustering (**"AGC Cluster"**)groups data into clusters by building a hierarchy of clusters using a graphical diagram called a dendrogram. It recursively splits or merges clusters until a stopping criterion is met. Agglomerative hierarchical clustering uses a bottom-up clustering methodology. Each data point starts out as a separate cluster and iteratively merges clusters based on a similarity or dissimilarity measure until a stopping criterion is reached. The similarity or dissimilarity between clusters can be calculated using various distance metrics such as Euclidean distance, Manhattan distance, or cosine similarity.

Project Objective

The objective of this project is to see if the three unsupervised machine learning algorithms, KMeans Clustering, Gaussian Mixture Models, and Agglomerative Hierarchial clustering will be able to identify a cluster of NBA players whose statistics will indicate that they are the elite 5% of the NBA. Players in this elite cluster will be compared to players voted on the NBA All Pro Teams.

Data Collection

For this project, I will use NBA player statistics from the **2021–22 regular season**. The data was collected from the website www.nbastuffer.com. NBA Stuffer provides robust NBA player statistics that help more than 1 million people who want to analyze NBA. I excluded data from the NBA 2020–21 playoffs as that might bias the clusters to those players on those teams that made the playoffs. Also, players are voted to the All NBA teams prior to the playoffs.

Data Exploration and Cleaning

After importing the entirety of all NBA player regular season statistics for the 2021–22 season, there were 716 observations and 29 features with only three of the features being non-numeric. Descriptions of each feature are included in the name of the feature.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 716 entries, 0 to 715
Data columns (total 29 columns):
 #   Column
---  ------
 0   RANK
 1   FULL NAME
 2   TEAM
 3   POS
 4   AGE
 5   GP
 6   MPG
 7   MIN%Minutes PercentagePercentage of team minutes used by a player while he was on the floor
 8   USG%Usage RateUsage rate, a.k.a., usage percentage is an estimate of the percentage of team plays used by a player while he was on the floor
 9   TO%Turnover RateA metric that estimates the number of turnovers a player commits per 100 possessions
 10  FTA
 11  FT%
 12  2PA
 13  2P%
 14  3PA
 15  3P%
 16  eFG%Effective Shooting PercentageWith eFG%, three-point shots made are worth 50% more than two-point shots made. eFG% Formula=(FGM+ (0.5 x 3PM))/FGA
 17  TS%True Shooting PercentageTrue shooting percentage is a measure of shooting efficiency that takes into account field goals, 3-point field goals, and free throws.
 18  PPGPointsPoints per game.
 19  RPGReboundsRebounds per game.
 20  TRB%Total Rebound PercentageTotal rebound percentage is estimated percentage of available rebounds grabbed by the player while the player is on the court.
 21  APGAssistsAssists per game.
 22  AST%Assist PercentageAssist percentage is an estimated percentage of teammate field goals a player assisted while the player is on the court
 23  SPGStealsSteals per game.
 24  BPGBlocksBlocks per game.
 25  TOPGTurnoversTurnovers per game.
 26  VIVersatility IndexVersatility Index is a metric that measures a player's ability to produce in points, assists, and rebounds. The average player will score around a five on the index, whil
 27  ORTGOffensive RatingIndividual offensive rating is the number of points produced by a player per 100 total individual possessions.
 28  DRTGDefensive RatingIndividual defensive rating estimates how many points the player allowed per 100 possessions he individually faced while staying on the court.
dtypes: float64(21), int64(5), object(3)
memory usage: 162.3+ KB
```

Original Dataset

The RANK feature which is just an index was dropped, and the rest of the feature names were shortened. Below is a list of the shortened names and description of each

feature:

FULL NAME = Full name of the player
TEAM = Name of the player's team
POS = Position of the player
AGE = Age of the player
GP = No. of games the player played
MPG = No. of minutes a player played per game
MIN% = Minutes Percentage is the percentage of team minutes used by a player while he was on the floor
USG% = Usage rate percentage is an estimate of the percentage of team plays used by a player while he was on the floor
TO% = Turnover Rate estimates the number of turnovers a player commits per 100 possessions
FTA = Free throw attempts
FT% = Percentage of free throws made to free throws attempted
2PA = Two point shots attempted
2P% = Percentage of two points made to two point shots attempted
3PA = Three point shots attempted
3P% = Percentage of three points made to two point shots attempted
eFG% = Effective Shooting Percentage. Three-point shots made are worth 50% more than two-point shots made. eFG% Formula=(FGM+ (0.5 x 3PM))/FGA
TS% = True Shooting Percentage is a measure of shooting efficiency that takes into account field goals 3-point field goals and free throws.
PPG = No. of points per game
RPG = No. of rebounds per game
TRB% = Total Rebound Percentage is estimated percentage of available rebounds grabbed by the player while the player is on the court.
APG = No. of assists per game.
AST% = Assist Percentage is an estimated percentage of teammate field goals a player assisted while the player is on the court
SPG = No. of steals per game
BPG = No. of Blocks per game
TOPG = No. of Turnovers per game
VI = Versatility index is a metric that measures a player's ability to produce in points, assists, and rebounds. The average player will score around a five on the index while top players score above 10
ORTG = Offensive Rating Individual is the number of points produced by a player per 100 total individual possessions
DRTG = Defensive Rating Individual estimates how many points the player allowed per 100 possessions he individually faced while staying on the court

List of Features

Duplicate Players

Players who played for multiple teams during the season, either because of trades or just being released, are entered multiple times.

	FULL NAME	TEAM
140	Seth Curry	Phi
141	Seth Curry	Bro
251	James Harden	Bro
252	James Harden	Phi
328	Alize Johnson	Chi
329	Alize Johnson	Was
330	Alize Johnson	Nor
533	Kristaps Porzingis	Dal
534	Kristaps Porzingis	Was

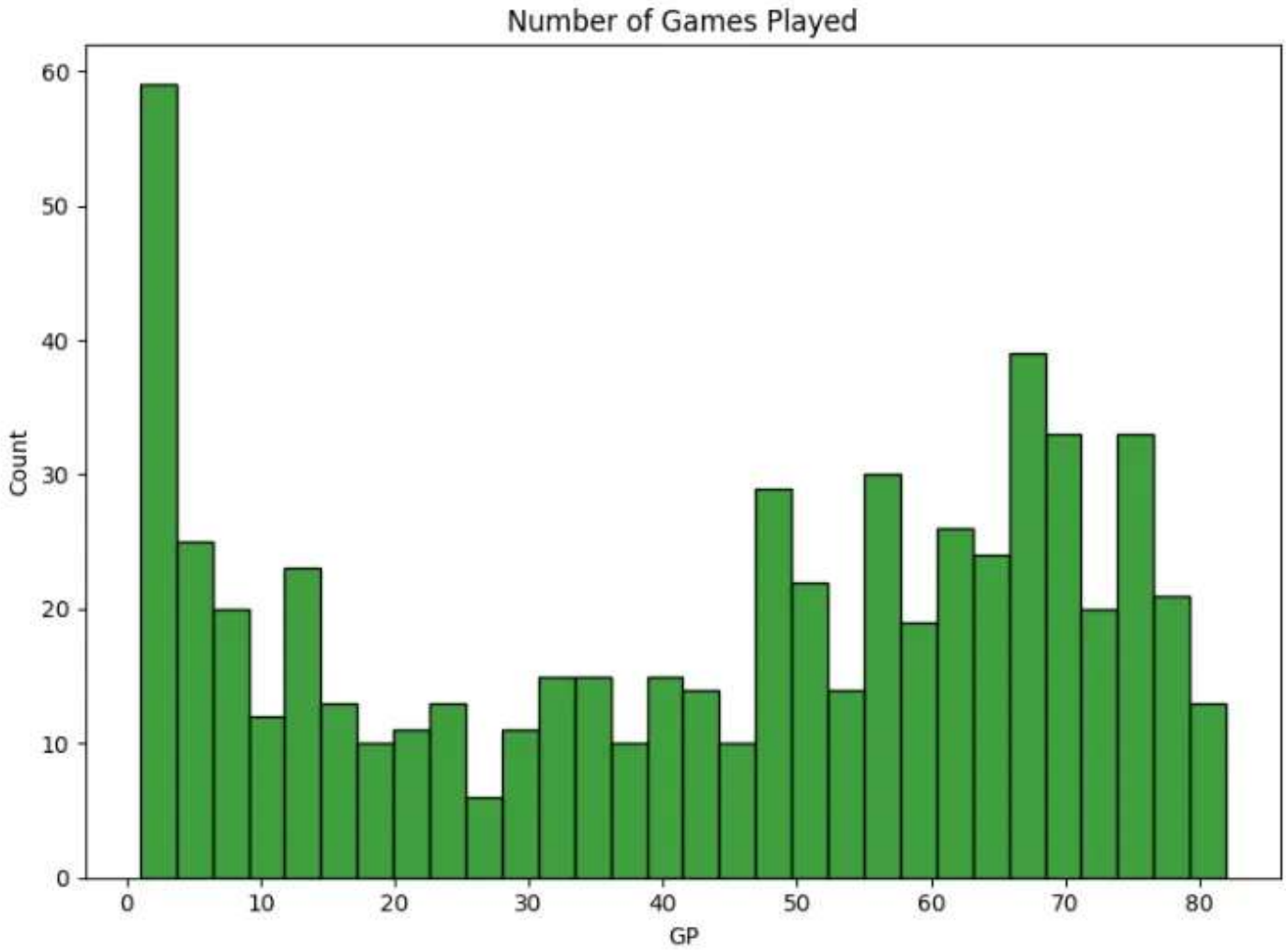
Multiple Players

These entries were de-duplicated by aggregating in-season stats for these duplicate players. This reduced the dataset to 605 players.

Data Exploration

	count	mean	std	min	25%	50%	75%	max
GP	605.0	43.042975	25.807966	1.0	17.0	48.0	66.0	82.0

A histogram for the number of Games Played (“GP”) reveals that a majority of players played less than 5 games:



Distribution of Games Played

Players who played less games than the bottom quartile (17 games) were removed from the dataset so as to not skew the results.

Minutes per Game (“MPG”)

	count	mean	std	min	25%	50%	75%	max
MPG	459.0	21.675454	8.438692	3.0	14.95	21.5	28.85	37.9

Minutes per Game

The average number of minutes played per game was 21.67 minutes. The bottom 25% played less than 15 minutes per game. For example, Kai Jones played only 3 minutes per the 21 games that he played which makes his offensive and defensive ratings NaN.

	FULL NAME	POS	AGE	GP	MPG
246	Kai Jones	C-F	21.23	21	3.0

Kai Jones 21 Games Played, 3 Minutes per Game

Players in the bottom quartile of Minutes per Game (less than 15 MPG) were removed from the dataset. After removal, there are 344 players in the dataset.

(*For the sake of brevity for this article, please see my full report on Google colab: for further discussion on Data Exploration including a discussion on Multicollinearity.)*

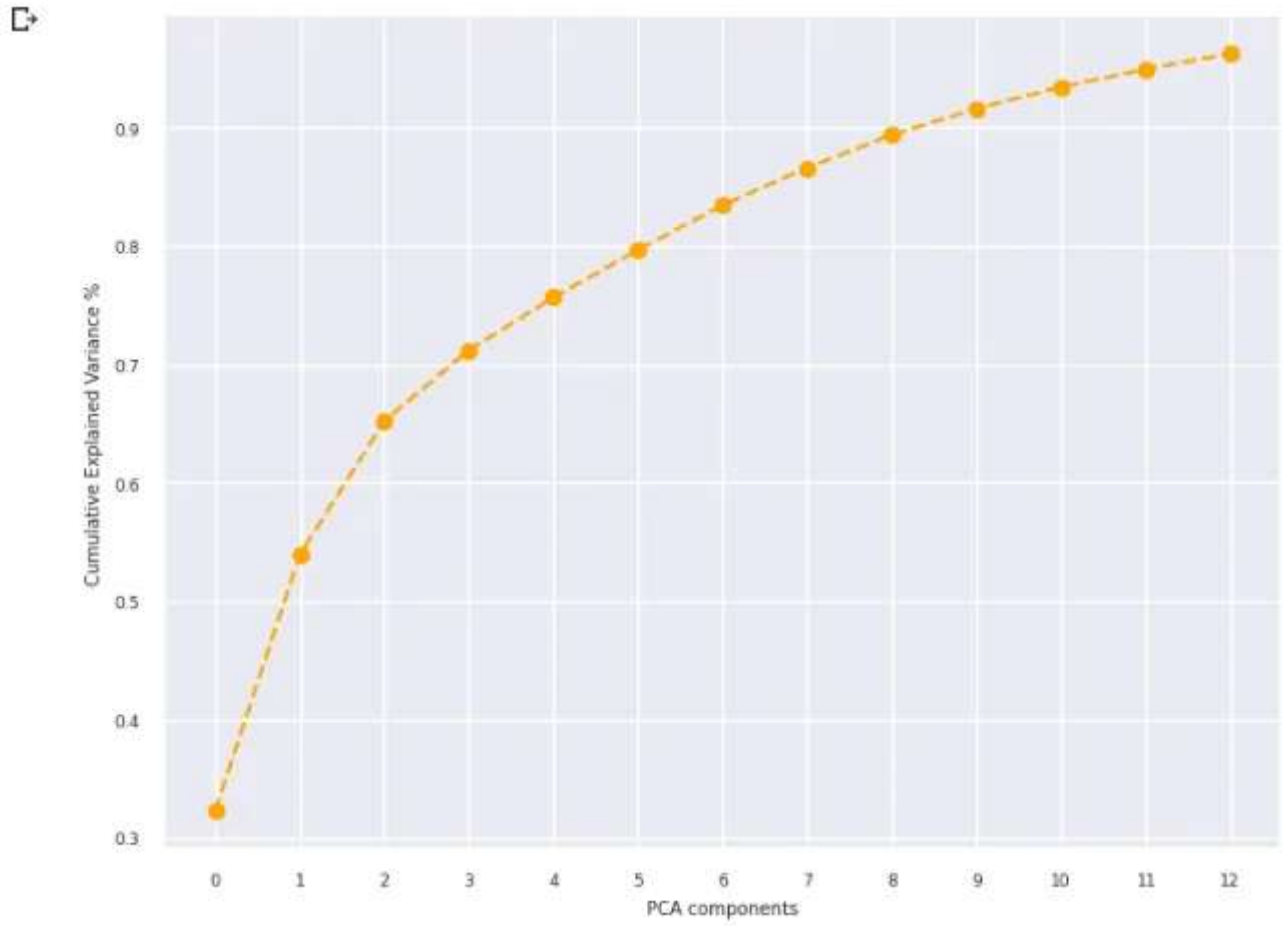
Dimensionality Reduction Using PCA and Manifold Learning

After removing the object features including the RANK feature and after scaling the features, the dataset now consists of 25 dimensions. Having a high number of dimensions can cause problems with clustering since clustering algorithms depend on measuring the distances between observations in order to identify the clusters. If the distances are all roughly equal, then all the observations appear equally alike (or equally different) and no meaningful clusters can be formed. High dimensionality means that space between each observation becomes equidistant. In this project, each player (or observation) represents a combination of the 25 dimensions. The statistical distances between each player become equally alike and no cluster with any meaningful information can be formed.

I tried two approaches to reduce dimensionality, Principal Component Analysis (“PCA”) and Manifold Learning. PCA provides a method to reduce the overall dimensionality of the features while still retaining the information of the larger feature set. Here, I reduce the number features need to retain 95% of the information in the dataset.

```
pca = PCA(n_components = .95)
# Fit the pca object to the standardized data
principalComponents = pca.fit_transform(df_scaled)
pca.explained_variance_ratio_
```

```
array([0.32270492, 0.21640281, 0.11323169, 0.05994224, 0.04453623,
       0.0402728 , 0.03781319, 0.03140261, 0.02790885, 0.02210657,
       0.01832836, 0.01497584, 0.01314284])
```



PCA has its limitations. "[I]t it often fails in that it assumes that the data can be modelled linearly. PCA expressed new features as linear combinations of existing ones by multiplying each by a coefficient…Whereas PCA attempts to create several linear hyperplanes to represent dimensions, much like multiple regression constructs as an estimation of the data, **Manifold learning** attempts to learn manifolds, which are smooth, curved surfaces within the multidimensional space."

Manifold Learning t-SNE, LLE, Isomap Made Easy, by Andre Ye (08/12/2020) Towards Data Science

Manifold learning generally refers to an unsupervised reduction, where the class is not presented to the algorithm (but may exist). Manifold learning attempts to learn manifolds, which are smooth, curved surfaces within the multidimensional space.

For this project, I used sklearn's manifold library, **Locally Linear Embedding** or "LLE". Seeks a lower-dimensional projection of the data which preserves distances within

local neighborhoods. *It can be thought of as a series of local Principal Component Analyses which are globally compared to find the best non-linear embedding.*

"In general, LLE is a more efficient algorithm as it eliminates the need to estimate pairwise distances between widely separated data points. Furthermore, it assumes that the manifold is linear when viewed locally. Thus it recovers the non-linear structure from locally linear fits", LLE: Locally Linear Embedding — A Nifty Way to Reduce Dimensionality in Python by Saul Dobilas (10/10/2021) TowardsDataScience.com

Using LLE, I reduced a 25 feature set down to just two features:

```
from sklearn.manifold import LocallyLinearEmbedding as LLE
lle = LLE(n_components=2)
df_lle = lle.fit_transform(df_scaled)
```

Manifold Learning

These two components were added to the main dataframe as well as saved off into their own numpy Series which will be used to fit the three clustering algorithsm.

```
final_df.head(2)
```

	AGE	GP	MPG	MIN%	USG%	TO%	FTA	FT%	2PA	2P%	...	APG	AST%	SPG	BPG	TOPG	VI	ORTG	DRTG	LLE_Component 1	LLE_Component 2
0	22.56	73	23.6	49.2	18.5	11.3	131.0	0.595	447.0	0.468	...	1.1	6.9	0.51	0.56	1.15	6.8	105.4	104.0	-0.039320	0.047856
1	28.73	76	26.3	54.8	12.0	19.6	199.0	0.543	383.0	0.548	...	3.4	16.1	0.87	0.79	1.51	9.4	124.7	103.9	-0.026388	-0.093104

Application of Unsupervised Learning Algorithms

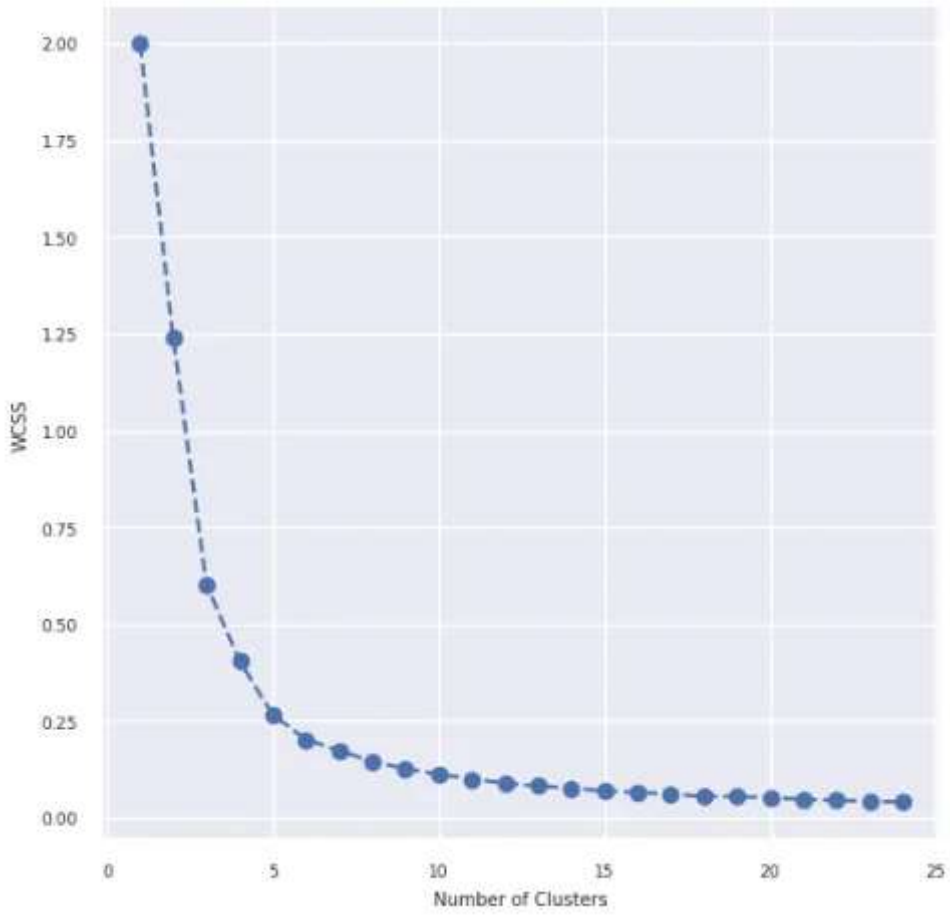
KMeans Clustering

The algorithm works by grouping data together into a fixed number of clusters. This number of clusters is the "K" in KMeans clustering. Data points are assigned to clusters based on their distance from the centroid of the cluster. It then calculates the means of each cluster. It iterates this process by taking the variation of the cluster. "The quality of the cluster assignments is determined by computing the sum of the squared error (SSE) after the centroids converge, or match the previous iteration's assignment. The SSE is defined as the sum of the squared Euclidean distances of each point to its closest centroid. Since this is a measure of error, the objective of k-means is to try to minimize

this value." Kevin Arvai, *K-Means Clustering in Python: A Practical Guide*, RealPython.com(2021)

Finding the correct number of cluster is an iterative process. I fitted the algorithm on the two Manifold components, df_lle, from 1 to 25 clusters. The number of clusters is determined by plotting each cluster against the within cluster sum of squares (WCSS) sum of the squared deviations from each observation and the cluster centroid. When the graph levels out after steep declines shows us the number of clusters.

The optimal number of KMeans clusters is 5. All 344 players are assigned one of these clusters



```
kmeans = KMeans(n_clusters=5, init='k-means++', n_init='auto', random_state=42,verbose=0)
kmeans.fit(df_lle)
```

```
KMeans
KMeans(n_clusters=5, n_init='auto', random_state=42)
```

```
kmeans.labels_
```

```
array([1, 4, 2, 3, 3, 4, 1, 1, 2, 0, 2, 0, 1, 3, 0, 2, 0, 3, 0, 0, 0, 2,
       0, 3, 3, 1, 2, 3, 0, 0, 3, 3, 1, 0, 0, 2, 3, 3, 2, 3, 1, 0, 0, 1,
       3, 2, 0, 3, 3, 0, 2, 0, 1, 4, 0, 1, 1, 4, 0, 4, 3, 0, 3, 0, 3, 3,
       2, 2, 0, 1, 3, 2, 1, 2, 0, 3, 1, 3, 1, 2, 2, 1, 1, 3, 2, 3, 3, 0,
       2, 1, 4, 0, 2, 3, 0, 3, 2, 2, 4, 0, 0, 0, 0, 3, 0, 3, 0, 3, 3, 3,
       1, 3, 3, 1, 0, 3, 0, 0, 3, 3, 1, 0, 3, 2, 3, 2, 3, 3, 1, 3, 0, 0,
       1, 1, 3, 2, 0, 0, 1, 3, 0, 2, 1, 0, 3, 0, 3, 2, 3, 3, 3, 1, 0, 1,
       0, 3, 3, 3, 1, 1, 3, 0, 2, 3, 1, 3, 0, 3, 3, 4, 3, 0, 2, 1, 0, 1,
       0, 3, 3, 3, 3, 3, 0, 1, 3, 3, 3, 4, 1, 1, 1, 2, 3, 1, 1, 2, 0, 0,
       2, 0, 1, 0, 2, 1, 3, 3, 3, 3, 0, 1, 3, 4, 3, 3, 3, 3, 0, 2, 1, 3,
       4, 0, 1, 2, 2, 3, 0, 3, 1, 3, 1, 2, 3, 3, 3, 3, 1, 4, 3, 1, 1, 0,
       1, 2, 3, 3, 1, 2, 0, 0, 1, 3, 3, 3, 1, 1, 0, 2, 3, 3, 1, 0, 3, 3,
       2, 0, 3, 3, 0, 3, 2, 0, 3, 0, 3, 0, 3, 1, 1, 1, 3, 1, 3, 2, 0, 4,
       3, 0, 4, 1, 3, 3, 3, 3, 0, 3, 2, 4, 1, 4, 1, 3, 0, 0, 1, 0, 3, 3,
       3, 1, 0, 0, 1, 3, 1, 1, 1, 3, 2, 1, 0, 3, 3, 1, 1, 0, 1, 3, 3, 4,
       1, 1, 0, 1, 3, 3, 0, 0, 3, 3, 3, 3, 3, 2], dtype=int32)
```

The KMeans cluster labels will be added to the dataframe.

```
[ ] final_df['KMeans Clusters'] = kmeans.labels_
```

KMeans Clusters

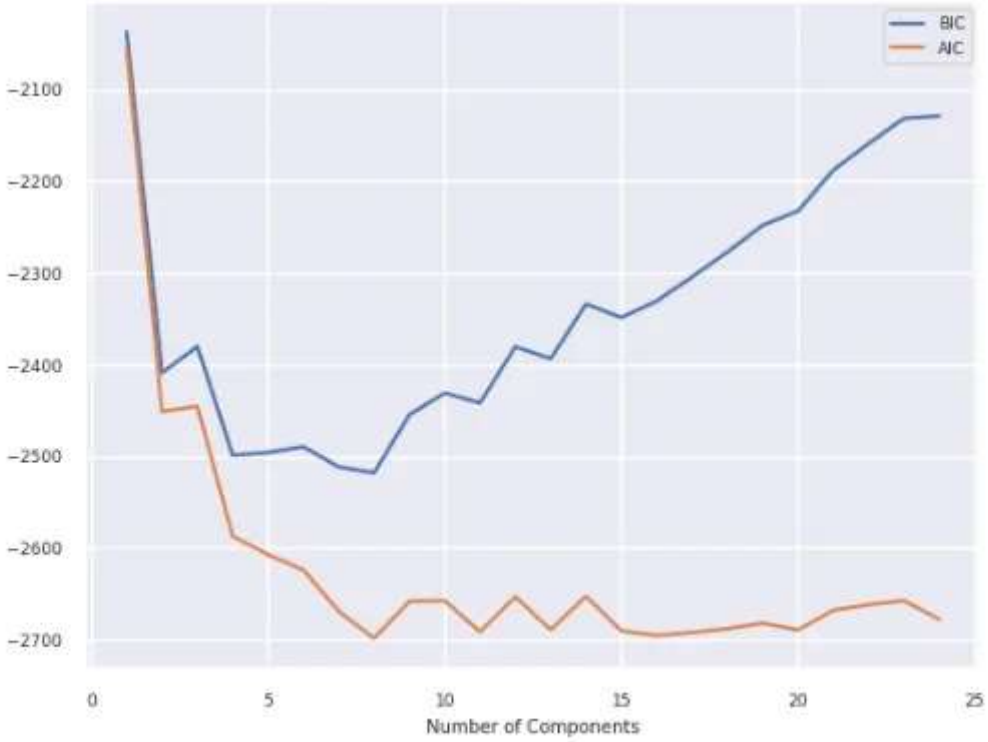
GMM Clusters

Similar to the KMeans approach, to discover the correct number of components for the algorithm is a trial and error process. I created a list comprehension which applied the Gaussian Mixture algorithm for 25 components. The **Akaike Information Criterion** (**“AIC”**) and **Bayesian Information Criteria** (**“BIC”**) are used to find the optimal number of components. Output from the AIC and BIC are plotted.The optimal number of components is determined by choosing the model with the lowest AIC or BIC value.In the model below, the best GMM model for the data is with 4 components.

```
#Find the number of optimal components

n_components = np.arange(1,25)

#Create a GMM model
model = [GaussianMixture(n_components=n,
                         random_state=42).fit(df_lle) for n in n_components ]
#Plot the model
plt.plot(n_components, [m.bic(df_lle) for m in model], label="BIC")
plt.plot(n_components, [m.aic(df_lle) for m in model], label="AIC")
plt.legend()
plt.xlabel('Number of Components');
```



Building the Model and Adding the GMM Segments

The GMM model is fitted and predicted which produces 4 clusters.

```
[ ]  model = GaussianMixture(n_components=4,
                             random_state=1502).fit(df_lle)
```

```
[ ]  #Predict for each cluster
     segments = pd.Series(model.predict(df_lle))
     final_df['GMM Clusters'] = segments
```

```
[ ]  final_df['GMM Clusters'].value_counts().sort_index()
```

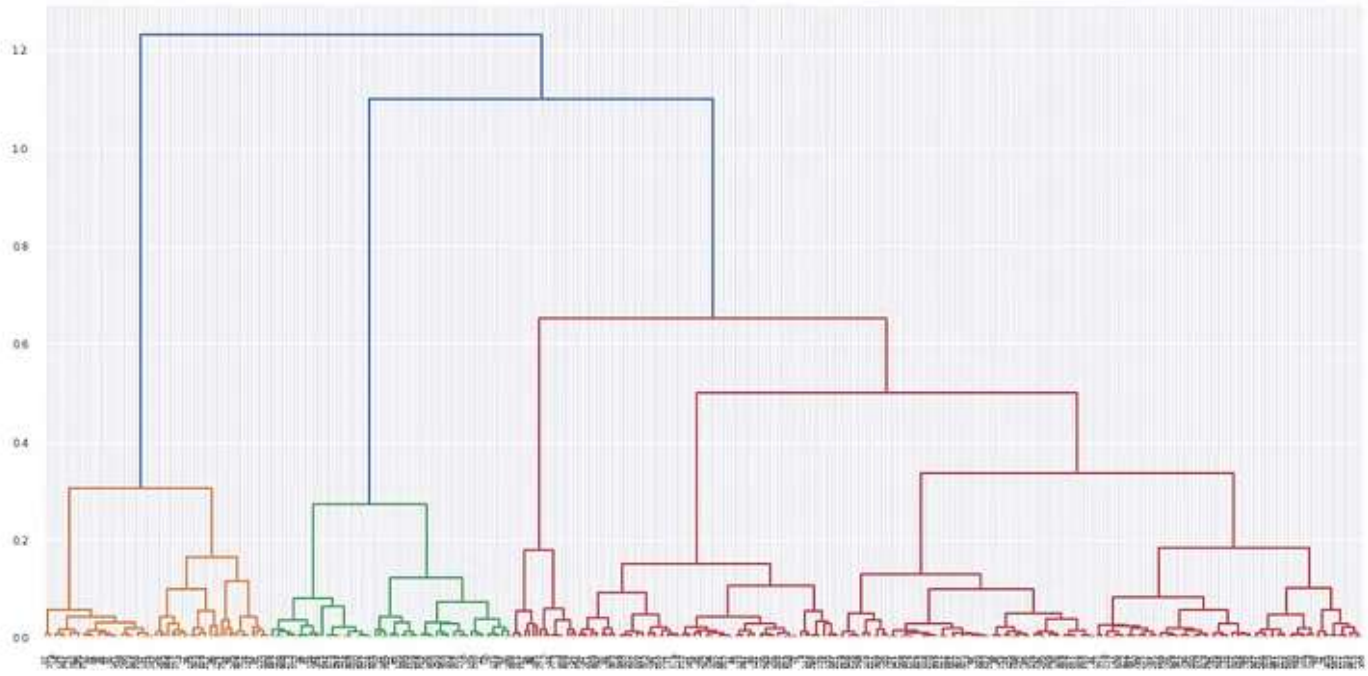
```
0    119
1     46
2     45
3    134
Name: GMM Clusters, dtype: int64
```

GMM Clusters

Agglomerative Hierarchy clustering

Agglomerative hierarchy clustering is bottom-up hierarchial clustering algorithm. It starts with each data point as a separate cluster and iteratively merge the closest pairs of clusters until a stopping criterion is met. This creates a hierarchy of clusters that can be visualized as a dendrogram. Common linkage methods for merging clusters include single linkage, complete linkage, and average linkage, among others.

The dendrogram shows that there are 4 AGC Clusters.



AGC Dendogram

```
[ ] final_df['AGC Clusters'].value_counts().sort_index()

0    205
1     59
2     63
3     17
Name: AGC Clusters, dtype: int64
```

NBA MVPs

Every year, the NBA names the Most Valuable Player (“MVP) of the league. The last 10 NBA MVPs are:

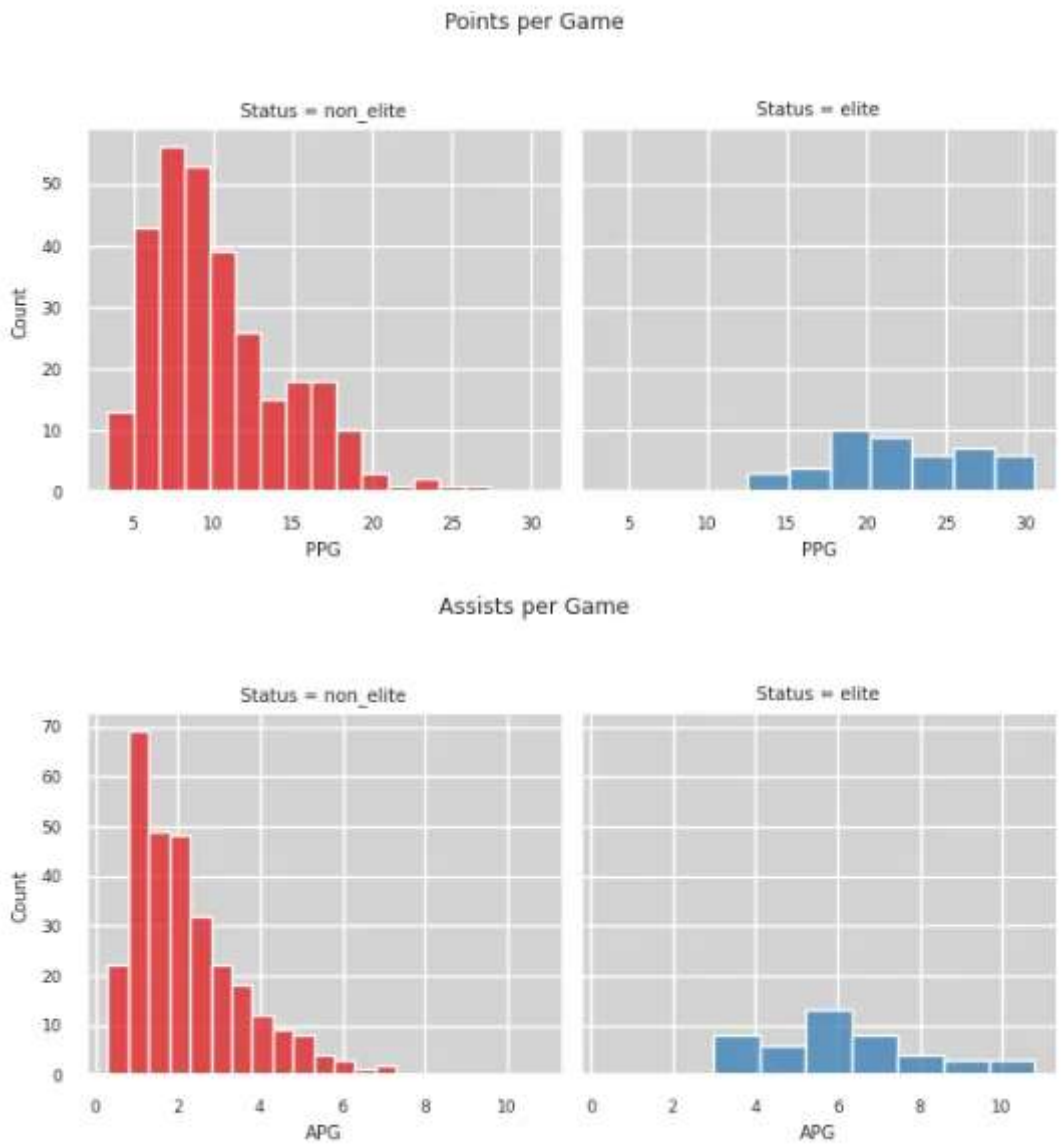
- LeBron James
- Kevin Durant
- Stephen Curry
- Russell Westbrook
- James Harden
- Giannis Antetokounmpo

- Nikola Jokic

Using the last 10 NBA MVPs (some were repeat), we can see that they all belong to the same clusters for each algorithm, KMeans and GMM which are cluster 2 and AGC which is cluster one.

	FULL NAME	KMeans Clusters	GMM Clusters	AGC Clusters
8	Giannis Antetokounmpo	2	2	1
318	James Harden	2	2	1
79	Kevin Durant	2	2	1
141	LeBron James	2	2	1
147	Nikola Jokic	2	2	1
283	Russell Westbrook	2	2	1
67	Stephen Curry	2	2	1

Out of 344 players in the dataset, 45 fit into these clusters and can be deemed “elite”. Visually, we can see the difference between elite players and other players the features. For example, see how skewed the distributions are over Points and Assists per Game.



Comparison of non-elite and elite players on PPG and APG

Finding the Elite of the Elite

At this point in the project, I have applied the three unsupervised machine learning algorithms which identified the initial elite clusters, KMeans Clusters = 2, GMM Clusters = 2, and AGC Clusters =1. There were 45 players out of 344 that were in all three clusters and could be called “Elite”. A sample of these players shown below includes recent MVP winners:

	FULL NAME	KMeans Clusters	GMM Clusters	AGC Clusters
8	Giannis Antetokounmpo	2	2	1
318	James Harden	2	2	1
79	Kevin Durant	2	2	1
141	LeBron James	2	2	1
147	Nikola Jokic	2	2	1
283	Russell Westbrook	2	2	1
67	Stephen Curry	2	2	1

Sample of Players Identified as Elite

To find the elite of the elite or top tier elite, the entire process was re-run but this time just against the elite cluster which were filtered into its own dataframe, elite_df, and its LLE components put into their own Series.

KMeans Elite Clusters

The KMeans algorithm found three clusters within the elite cluster:

```
random.seed(42)
kmeans = KMeans(n_clusters=3, init='k-means++', n_init='auto', random_state=42,verbose=0)
kmeans.fit(elite_lle)
```

```
KMeans(n_clusters=3, n_init='auto', random_state=42)
```

```
elite_df["KMeans Elite Clusters"] = kmeans.labels_
```

```
elite_df["KMeans Elite Clusters"].value_counts()
```

```
0    31
1     8
2     6
Name: KMeans Elite Clusters, dtype: int64
```

KMeans Elite Clusters

GMM Elite Clusters

There are only two GMM Elite Clusters:

```
[ ] model = GaussianMixture(n_components=2,
                            random_state=1502).fit(elite_lle)
```

```
[ ] #Predict for each cluster
    segments = pd.Series(model.predict(elite_lle))
    elite_df['GMM Elite Clusters'] = segments
```

```
[ ] elite_df['GMM Elite Clusters'].value_counts()
```

```
0    31
1    14
Name: GMM Elite Clusters, dtype: int64
```

AGC Elite Clusters

The Agglomerative Hierarchial cluster algorithm also only found two clusters.

```
[ ] random.seed(42)
    agc2 = AGC(n_clusters=2, linkage="ward")
    agc2.fit(elite_lle)
```

```
AgglomerativeClustering()
```

```
[ ] elite_df['AGC Elite Clusters'] = agc2.labels_
    elite_df['AGC Elite Clusters'].value_counts()
```

```
1    29
0    16
Name: AGC Elite Clusters, dtype: int64
```

The MVP Test

Using the previous and current MVPs as a test, the elite elite cluster is comprised of KMeans and GMM Elite Clusters equaling 0 and the AGC Elite Cluster equaling 1. Players who are in these three clusters are tagged as “top tier elite” while players not in these three clusters are tagged as “second tier elite”.

	FULL NAME	KMeans Elite Clusters	GMM Elite Clusters	AGC Elite Clusters
1	Giannis Antetokounmpo	0	0	1
11	Stephen Curry	0	0	1
14	Kevin Durant	0	0	1
16	Joel Embiid	0	0	1
24	LeBron James	0	0	1
25	Nikola Jokic	0	0	1
43	James Harden	2	1	0

Top Tier Elite Clusters

The Top 5% — Top Tier Elite

The algorithms have now identified the top 29 players (the top 5%)

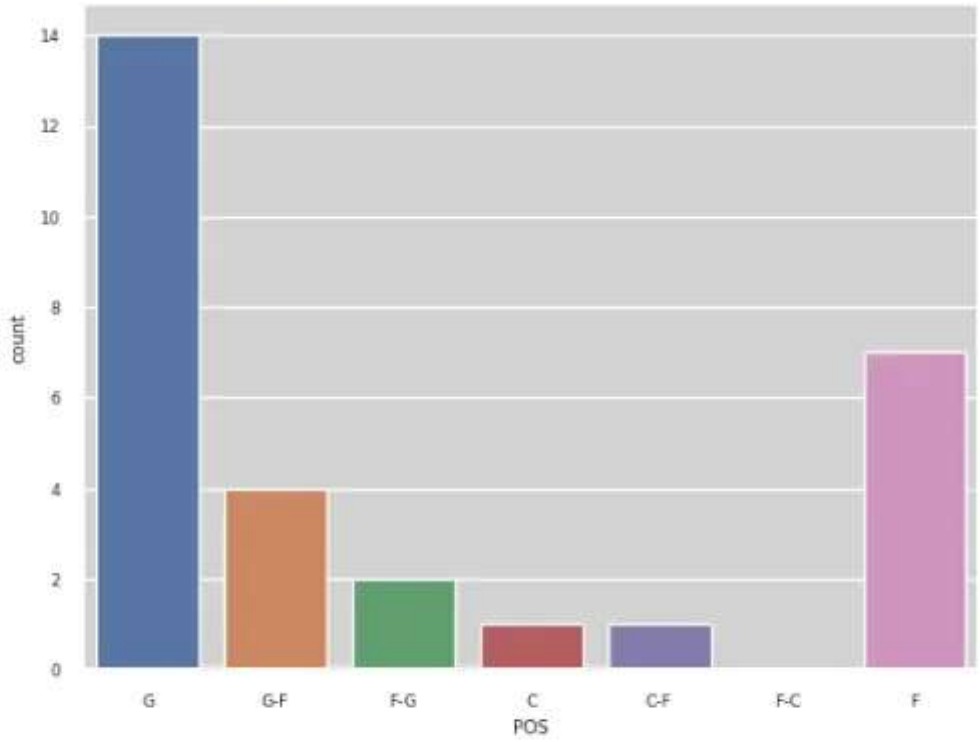
```
[ ] elite_df["Elite Status"].value_counts()

top tier elite       29
second tier elite    16
Name: Elite Status, dtype: int64
```

Characteristics of Top Tier Elite — Position

Of the 29 top tier elite players, 14 of them purely play the Guard position. There are no hybrid Forward-Centers, and there is only pure center, Nikola Jokic, and one hybrid Center-Forward, Joel Embiid.

```
<Axes: xlabel='POS', ylabel='count'>
```

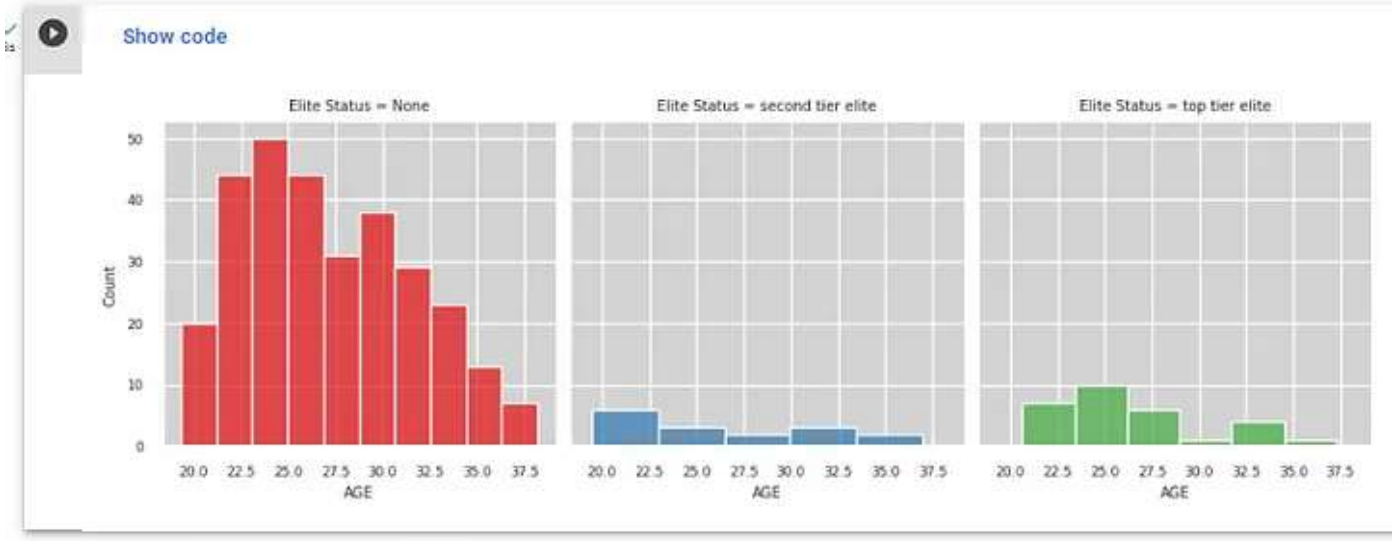


Show code

	FULL NAME	POS	Elite Status
16	Joel Embiid	C-F	top tier elite
25	Nikola Jokic	C	top tier elite

Characteristics of Top Tier Elite — Age

For the top tier elite, the age drop off is even more pronounced. There's a significant drop off after age 27.5 compared to the more gradual drop off of other players. The average age is 29 with only 9 players are aged 27.5 or higher.



▸ **Age Distribution of Top Tier Elite Players**

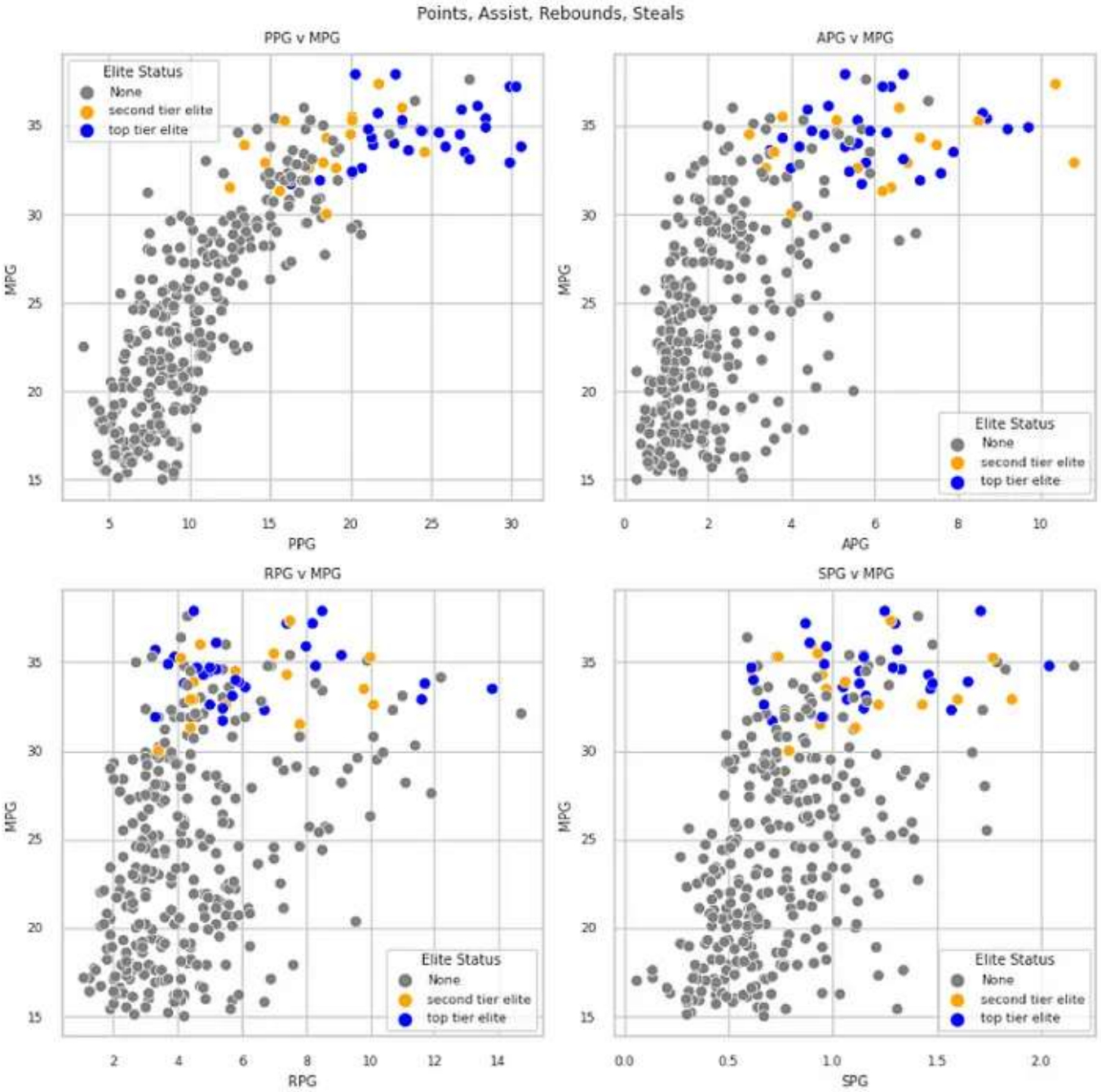
[335] Show code

	count	mean	std	min	25%	50%	75%	max
AGE	29.0	26.501724	4.270812	20.64	23.56	25.56	28.07	37.28

Age Distribution of the Top Tier Elite

Characteristics of Top Tier Elite Players — More Points, Rebounds, Assists, and Steals

Across the production categories of Points, Assists, Rebounds, and Steals, the top tier elite well out-perform the non-elite players and only slightly out-perform the second tier elite.



Characteristics of Top Tier Elite Players — Versatility Index

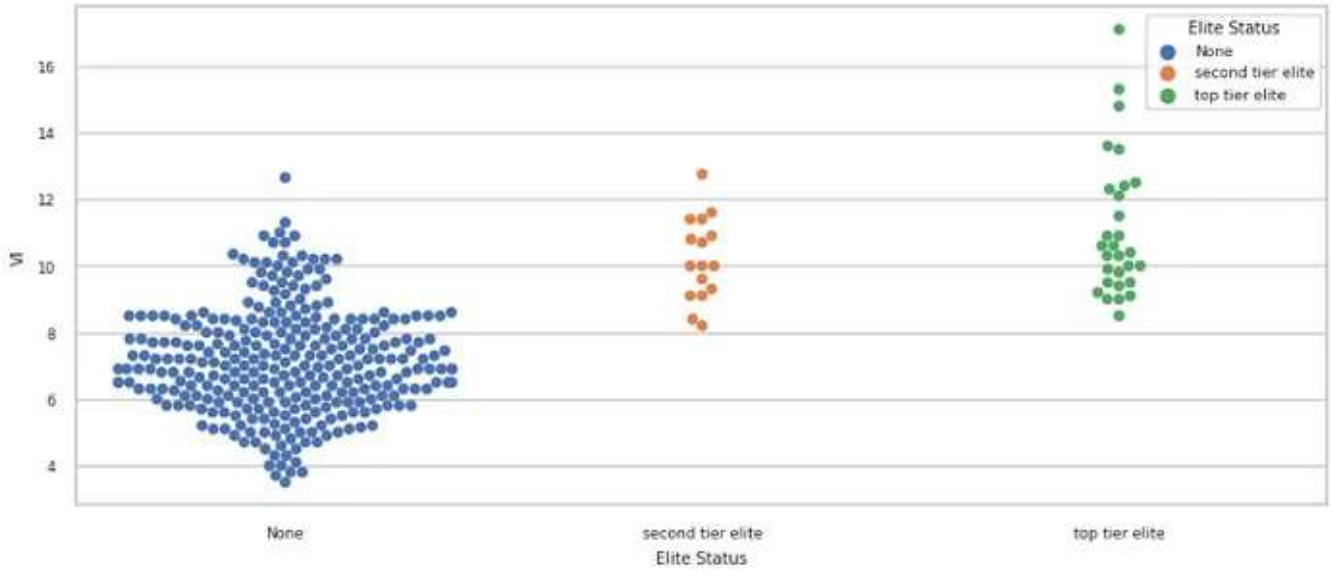
The Versatility Index (“VI”) is a metric that measures a player’s ability to produce in points, assists, and rebounds. The average player will score around a five on the index while top players score above 10.

▸ Versatility Index Mean

Show code

Elite Status	VI
None	7.134392
second tier elite	10.203125
top tier elite	11.103448

VI Mean



Top Tier Elite Players — The Top 5%

Out of 605 total players, these 29 have been identified as the top tier elite:

	FULL NAME	POS	NBA_ALL_Pro	Elite Status
1	Devin Booker	G	1st	top tier elite
2	Giannis Antetokounmpo	F	1st	top tier elite
3	Jayson Tatum	F-G	1st	top tier elite
4	Luka Doncic	F-G	1st	top tier elite
5	Nikola Jokic	C	1st	top tier elite
6	DeMar DeRozan	G-F	2nd	top tier elite
7	Ja Morant	G	2nd	top tier elite
8	Joel Embiid	C-F	2nd	top tier elite
9	Kevin Durant	F	2nd	top tier elite
10	Stephen Curry	G	2nd	top tier elite
11	LeBron James	F	3rd	top tier elite
12	Pascal Siakam	F	3rd	top tier elite
13	Trae Young	G	3rd	top tier elite
1	Anthony Edwards	G	Not Selected	top tier elite
2	Brandon Ingram	F	Not Selected	top tier elite
3	Cole Anthony	G	Not Selected	top tier elite
4	D'Angelo Russell	G	Not Selected	top tier elite
5	Darius Garland	G	Not Selected	top tier elite
6	De'Aaron Fox	G	Not Selected	top tier elite
7	Dejounte Murray	G	Not Selected	top tier elite
8	Donovan Mitchell	G	Not Selected	top tier elite
9	Fred VanVleet	G	Not Selected	top tier elite
10	Jaylen Brown	G-F	Not Selected	top tier elite
11	Jimmy Butler	F	Not Selected	top tier elite
12	Khris Middleton	F	Not Selected	top tier elite
13	LaMelo Ball	G	Not Selected	top tier elite
14	Shai Gilgeous-Alexander	G-F	Not Selected	top tier elite
15	Tyler Herro	G	Not Selected	top tier elite
16	Zach LaVine	G-F	Not Selected	top tier elite

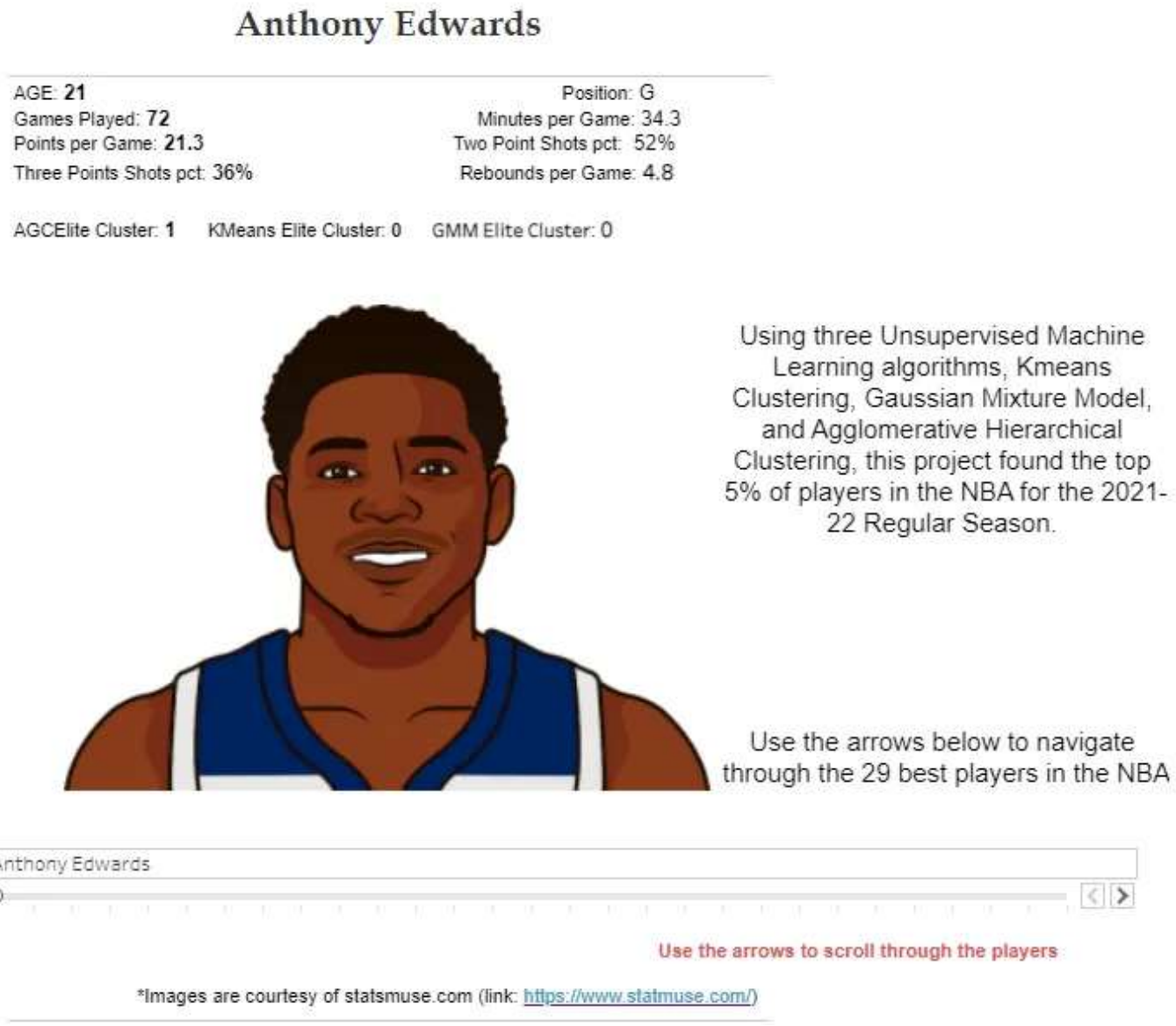
All of the players identified by the three algorithms also made a 2021–22 All NBA teams with two exceptions:

Karl Anthony Towns who is a Center. All NBA teams have to vote by position. Towns, who is in the Second Tier Elite, is one of two Centers in that cluster. There are only 4 Centers in total in the Elite Clusters.

The other exception is Chris Paul which is a curious selection. There are 10 Guards ahead of him in the top tier. Paul is in the second tier elite, but that was cleary a sentimental, biased choice.

Tableau Visualization

For a Tableau Visualization of these players along with in-depth analysis, please visit this link: Tableau Visualization



Google Colab Notebook

NBA Top Tier Segment Analysis — Google Colab Notebook

Contact information

John K. Hancock
Email: *jkhancock@gmail.com*

References

Richard Bellman, Wikipedia Article, https://en.wikipedia.org/wiki/Richard_E._Bellman (Last Updated April 9, 2023)

Andre Ye, Manifold Learning t-SNE, LLE, Isomap Made Easy, Towards Data Science, https://www.towardsdatascience.com (08/12/2020)

Saul Diblas, LLE: Locally Linear Embedding — A Nifty Way to Reduce Dimensionality in Python, Towards DataScience, https://towardsdatascience.com (10/10/2021)

Matt Brems, A One-Stop Shop for Principal Component Analysis, Towards Data Science, https://towardsdatascience.com (April 17, 2017)

Kevin Arvai, K-Means Clustering in Python: A Practical Guide 2021 English Real Python, https://realpython.com/k-means-clustering-python/

Chris Ding and Xiaofeng He, K-means Clustering via Principal Component Analysis, K-means Clustering via Principal Component Analysis, Proc. of Int'l Conf. Machine Learning, https://ranger.uta.edu/~chqding/papers/KmeansPCA1.pdf (2004)

Ajitesh Kumar, Gaussian Mixture Models: What are they & when to use?, Data Analytics, https://vitalflux.com/gaussian-mixture-models-what-are-they-when-to-use/ (April 14, 2022)

Images courtesy of the website www.statsmuse.com

Manifold Learning Kmeans Clustering Gaussian Mixture Model

Agglomerative Clustering Tableau

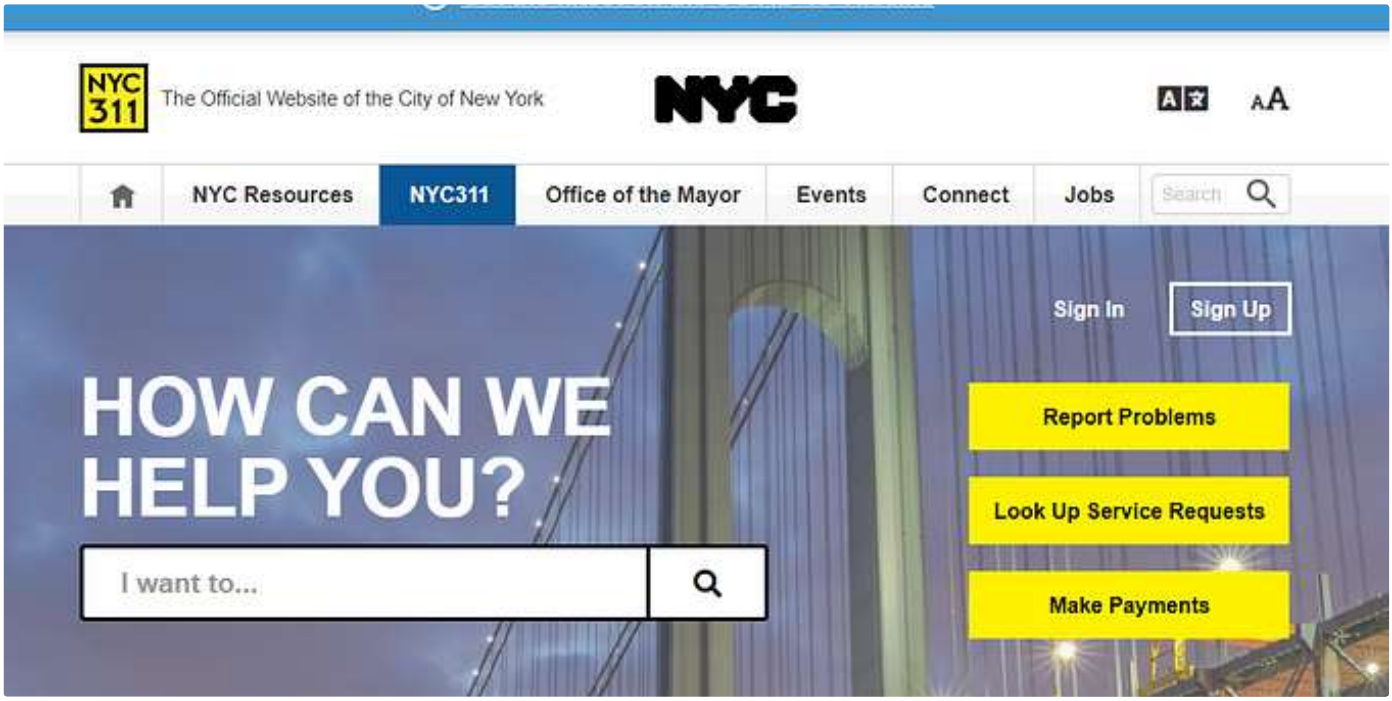
Edit profile

Written by John K. Hancock

14 Followers

Team Lead, MSDS, MBA, PMP

More from John K. Hancock



John K. Hancock

Visualizing NYC Noise Complaints with MS PowerBI

Data visualization is a powerful tool that provides fast and easily accessible insights into large data collections. Key decision makers at...

6 min read · Mar 3

See all from John K. Hancock

Recommended from Medium

Andrew Morris in INST414: Data Science Techniques

Finding Similar Companies with Cosine Similarity

Identifying businesses similar to a particular entity is a critical task that underpins various strategic decisions, enabling organizations...

3 min read · 4 days ago

39

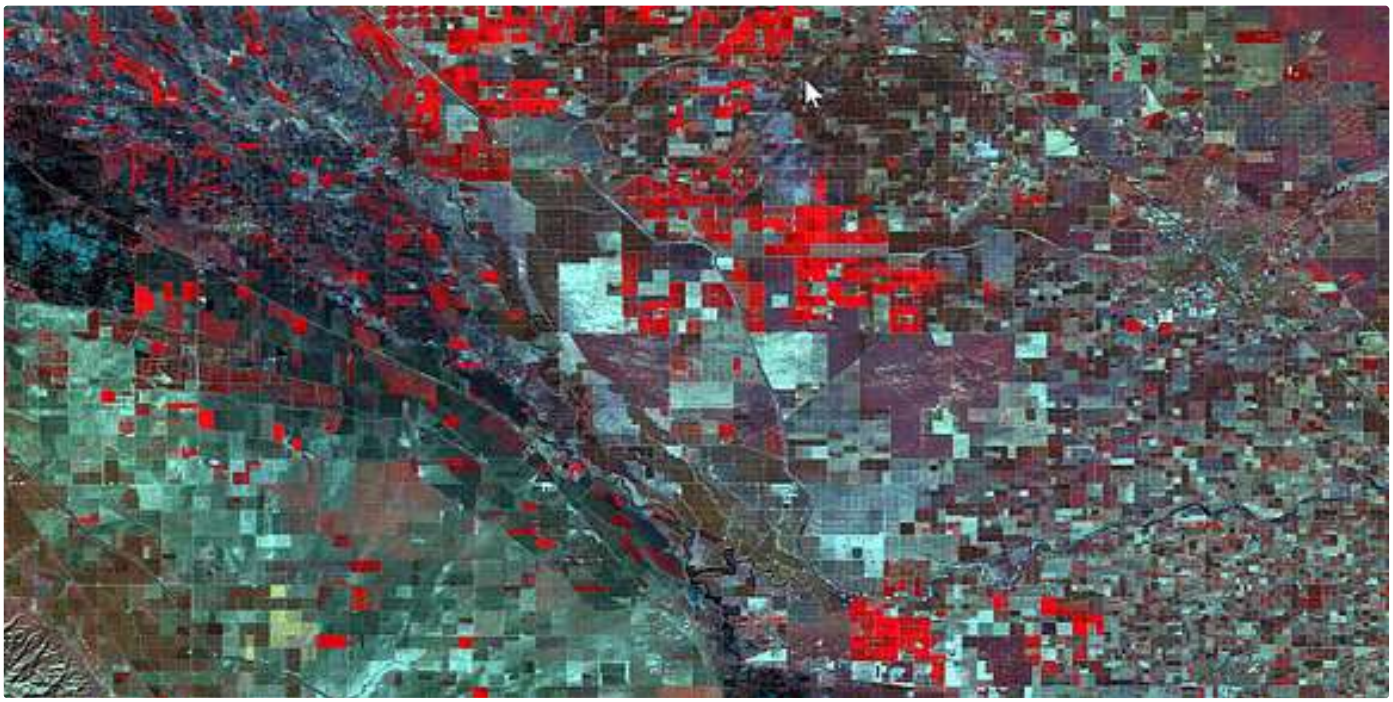
Emmanuel Ikogho

Classification—Predicting Customer Churn

1.0 Introduction (What is Customer Churn?)

12 min read · Mar 25

35 3



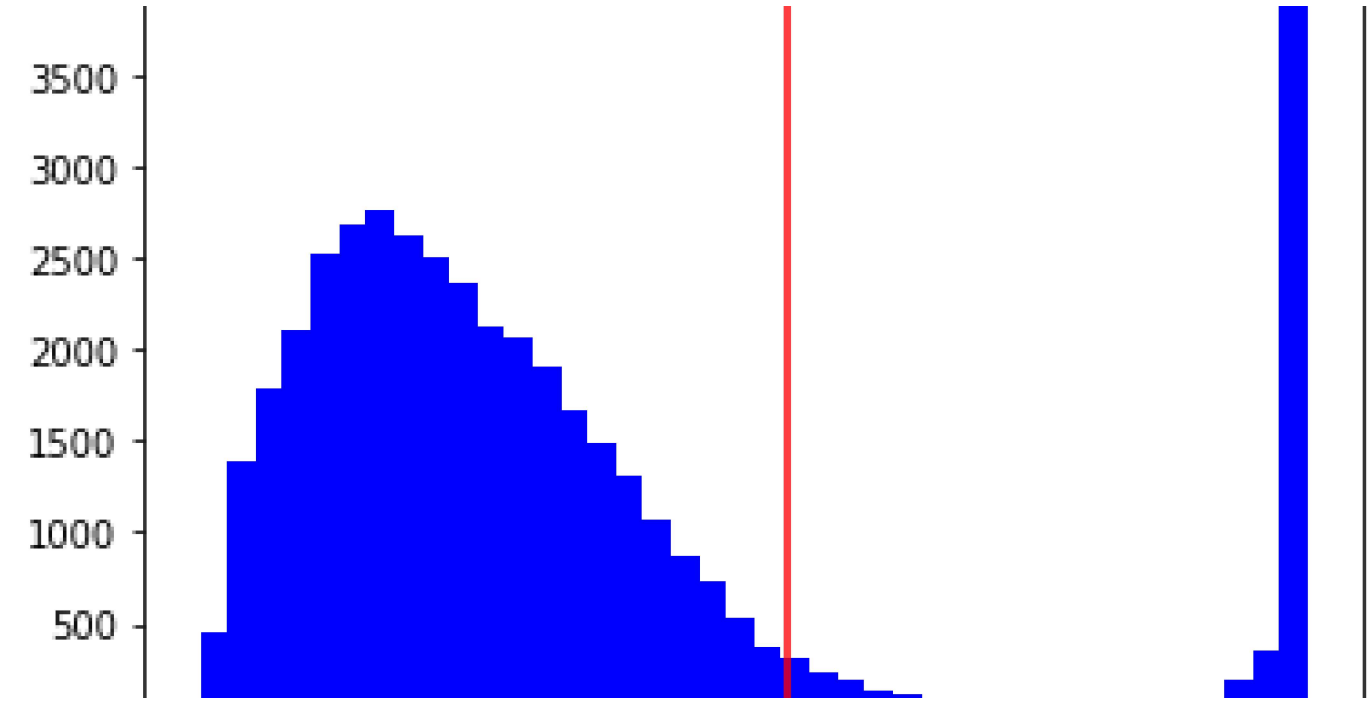
Soumya Kanta Dash in GeoAI

Crop classification via satellite image time-series and PSETAE deep learning model

In modern agriculture, crop classification plays a crucial role in identifying and mapping different crop types. However, this task poses a...

9 min read · May 9

6 1



Dana Fatadilla Rabba

Credit Risk Modelling using Loan Dataset

Nowadays, one can invest in other people's loans using online peer-to-peer lending platforms such as, for example, Lending Club. In the...

7 min read · Mar 20

3

Omolara Babatunde

ANALYSIS OF BIKE SALES DATA

This project covers data cleaning, data analysis and data visualization process using Microsoft excel, and this is the project's...

5 min read · Apr 16

35 3

See more recommendations