Market Segmentation with Cluster Analysis

Introduction:

This report delves into the analysis of data obtained from a retail shop, encompassing information from 30 clients, each associated with their respective satisfaction and loyalty scores.

Satisfaction: This metric is self-reported, with customers asked to rate their shopping experience on a scale from 1 to 10. A rating of 10 indicates extreme satisfaction, while a score of 1 reflects no satisfaction at all. Consequently, satisfaction is a discrete variable ranging from 1 to 10.

Brand Loyalty: Assessing brand loyalty involves employing various metrics, and while there is no universally accepted technique, proxies such as churn rate, retention rate, or Customer Lifetime Value (CLV) are commonly used. In this study, Brand Loyalty is determined through the formula:

Brand Loyalty = Number of purchases from that shop per year +Other significant Factors

The resulting Brand Loyalty is a continuous variable, standardized within a range of -2.5 to 2.5.

It's crucial to note that the standardization of this variable has already been performed. The understanding of loyalty, in this context, extends beyond mere purchase frequency to incorporate additional influential factors.

Let’s start with the project:

1. Begin by importing the necessary Python libraries.

Import the relevant libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# Set the styles to Seaborn
sns.set()
# Import the KMeans module so we can perform k-means clustering with sklearn
from sklearn.cluster import KMeans
```

2. Next, load and examine the dataset.

Load the data

```
In [5]: # Load the data
        data = pd.read_csv ('3.12. Example.csv')

In [6]: # Check what's inside
        data
```

Out[6]:

	Satisfaction	Loyalty
0	4	-1.33
1	6	-0.28
2	5	-0.99
3	7	-0.29
4	4	1.06
5	1	-1.66
6	10	-0.97
7	8	-0.32
8	8	1.02

3. To enhance comprehension, visualize the data by creating plots.

Plot the data

Create a preliminary plot to see if you can spot something

```
[7]: # We are creating a scatter plot of the two variables
     plt.scatter(data['Satisfaction'],data['Loyalty'])
     # Name your axes
     plt.xlabel('Satisfaction')
     plt.ylabel('Loyalty')
```

Out[7]: Text(0, 0.5, 'Loyalty')

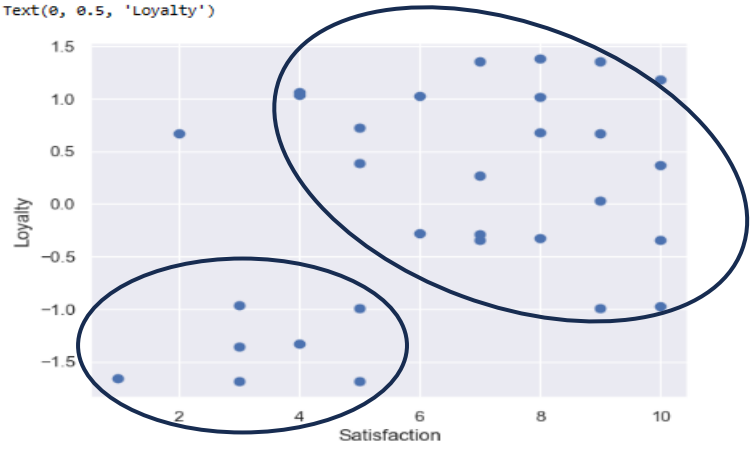


Figure 1

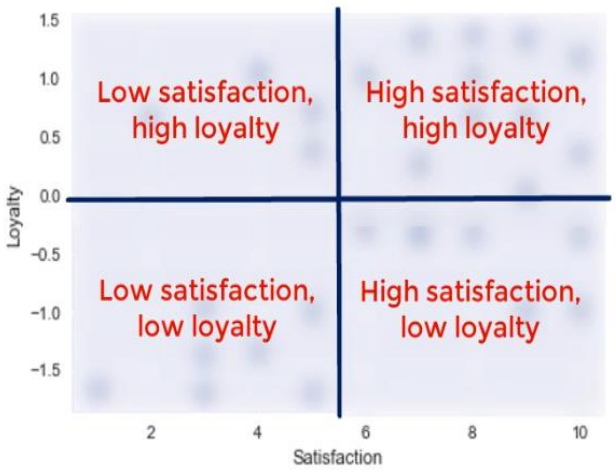


Figure 2

Examining Figure 1, we observe the formation of an elliptical shape comprised of two segments. Before delving further, let's contemplate dividing the graph into four squares, as illustrated in Figure 2. Upon combining these graphs in Figure 3, the interpretation of our initial segment becomes nonsensical. While one ellipse represents low satisfaction and low loyalty, the other appears

Figure 3

scattered without a clear pattern. Consequently, the two-cluster solution lacks reliability.

Clustering

Let's leverage the new knowledge we possess and build something new for in depth understanding to divide into a meaningful cluster/segment.

Select the features

```
# Select both features by creating a copy of the data variable
x = data.copy()
```

Clustering

```
# Create an object (which we would call kmeans)
# The number in the brackets is K, or the number of clusters we are aiming for
kmeans = KMeans(2)
# Fit the data
kmeans.fit(x)
```

Clustering results

```
# Create a copy of the input data
clusters = x.copy()
# Take note of the predicted clusters
clusters['cluster_pred']=kmeans.fit_predict(x)
clusters
```

Out[11]:

	Satisfaction	Loyalty	cluster_pred
0	4	-1.33	1
1	6	-0.28	1
2	5	-0.99	1
3	7	-0.29	0
4	4	1.06	1
5	1	-1.66	1
6	10	-0.97	0
7	8	-0.32	0
8	8	1.02	0
9	8	0.68	0
10	10	-0.34	0

```
# Plot the data using the Satisfaction on X-Axis and Loyalty on Y-Axis
# c (color) is an argument which could be coded with a variable, to visualize the segment clearly.
# The variable in this case has values 0,1, indicating to plt.scatter, that there are two colors (0,1)
# All points in cluster 0 will be the same colour, all points in cluster 1 - another one, etc.
# cmap is the color map.
plt.scatter(clusters['Satisfaction'],clusters['Loyalty'],c=clusters['cluster_pred'],cmap='rainbow')
plt.xlabel('Satisfaction')
plt.ylabel('Loyalty')
```

```
4]: Text(0, 0.5, 'Loyalty')
```

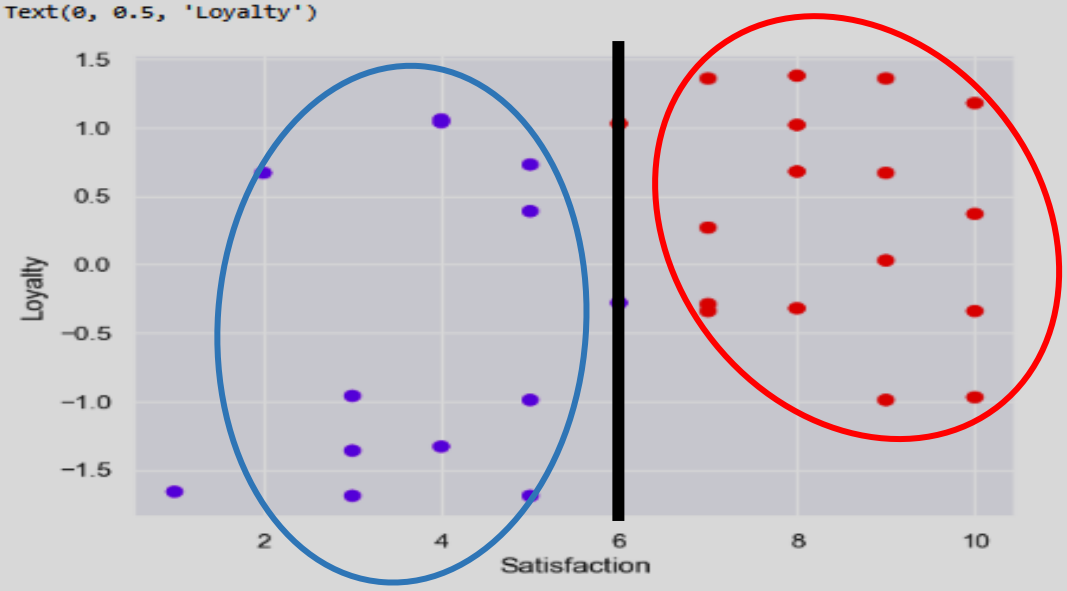


Figure 4

The KMeans predictive algorithm divide the data into two segments, right and left, at a satisfaction value of 6, as shown in figure 4. However, it is highly likely that the algorithm predominantly factored in satisfaction as a feature, given that we did not standardize the variable. Because the satisfaction values surpass those of loyalty, KMeans likely neglected loyalty as a significant feature in its clustering process.

Including both Satisfaction and Loyalty as variables

Since Loyalty and Satisfaction share equal importance/weight in Market Segmentation, but we see above the algorithm only take satisfaction as a feature, we can do this by standardizing the variables. There are many ways to do this in SK learn but I use the simplest one which is pre-processing model and scale method (it will scale every variable separately or it will make each column standardize). Result is shown in figure below:

Standardize the variables

Let's standardize and check the new result

```
]:  # Import a library which can do that easily
    from sklearn import preprocessing
    # Scale the inputs
    # preprocessing.scale scales each variable (column in x) with respect to itself
    # The new result is an array
    x_scaled = preprocessing.scale(x)
    x_scaled
```

```
:[16]: array([[-0.93138063, -1.3318111 ],
       [-0.15523011, -0.28117124],
       [-0.54330537, -0.99160391],
       [ 0.23284516, -0.29117733],
       [-0.93138063,  1.05964534],
       [-2.09560642, -1.6620122 ],
       [-1.39707805, -0.97159172]])
```

The array of x_scaled contains the standardized value for Satisfaction and Loyalty.

Now we need number of clusters “K” and distance between points in a cluster (within-cluster sum of squares,WCSS), we can find this by using Inertia method of SK Learrn.

Take advantage of the Elbow method

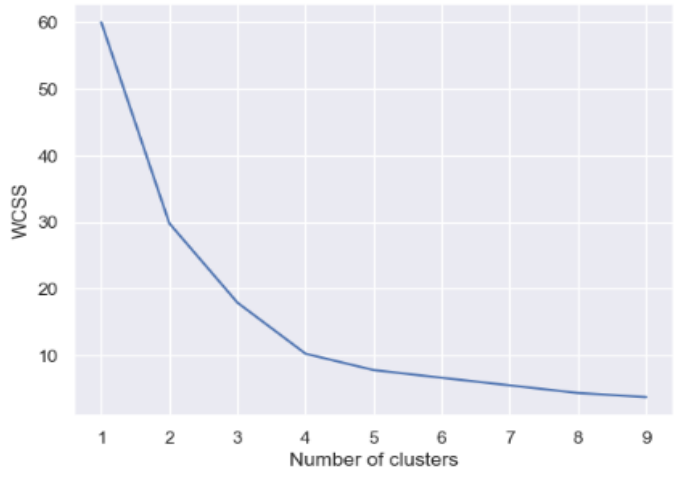
```
]:  # Createa an empty list
    wcss =[]

    # Create all possible cluster solutions with a loop
    # We have chosen to get solutions from 1 to 9 clusters; you can ammend that if you wish
    for i in range(1,10):
        # Clsuter solution with i clusters
        kmeans = KMeans(i)
        # Fit the STANDARDIZED data
        kmeans.fit(x_scaled)
        # Append the WCSS for the iteration
        wcss.append(kmeans.inertia_)

    # Check the result
    wcss
7]: [59.999999999999986,
     29.818973034723143,
     17.913349527387968,
     10.247181805928422,
     7.792695153937187,
     6.586212092192188,
     5.34807941029098,
     4.337110750237059,
     3.7972682187482842]
```

The array above represents within cluster sum of squares of distance.

Optimum number of K and WCSS

To find the optimal number of K and WCSS let’s plot the result we already have.

```
    # Plot the number of clusters vs WCSS
    plt.plot(range(1,10),wcss)
    # Name your axes
    plt.xlabel('Number of clusters')
    plt.ylabel('WCSS')
10]: Text(0, 0.5, 'WCSS')
```



Looking at the graph we cannot decide directly the number of clusters as there may be many possibilities like 3,4,5 and so on but we don’t know the best solution., this is basically the limitation of the Elbow Method. We can see the change in WCSS with the increase in the number of clusters but still we don’t know the best solution.

To find the best solution it is worth inspecting the difference with standardize variables, to do this I made a new data frame(cluster_new).

Explore clustering solutions and select the number of clusters

```
# Fiddle with K (the number of clusters)
kmeans_new = KMeans(2)
# Fit the data
kmeans_new.fit(x_scaled)
# Create a new data frame with the predicted clusters
clusters_new = x.copy()
clusters_new['cluster_pred'] = kmeans_new.fit_predict(x_scaled)
```

```
# Check if everything seems right
clusters_new.head()
```

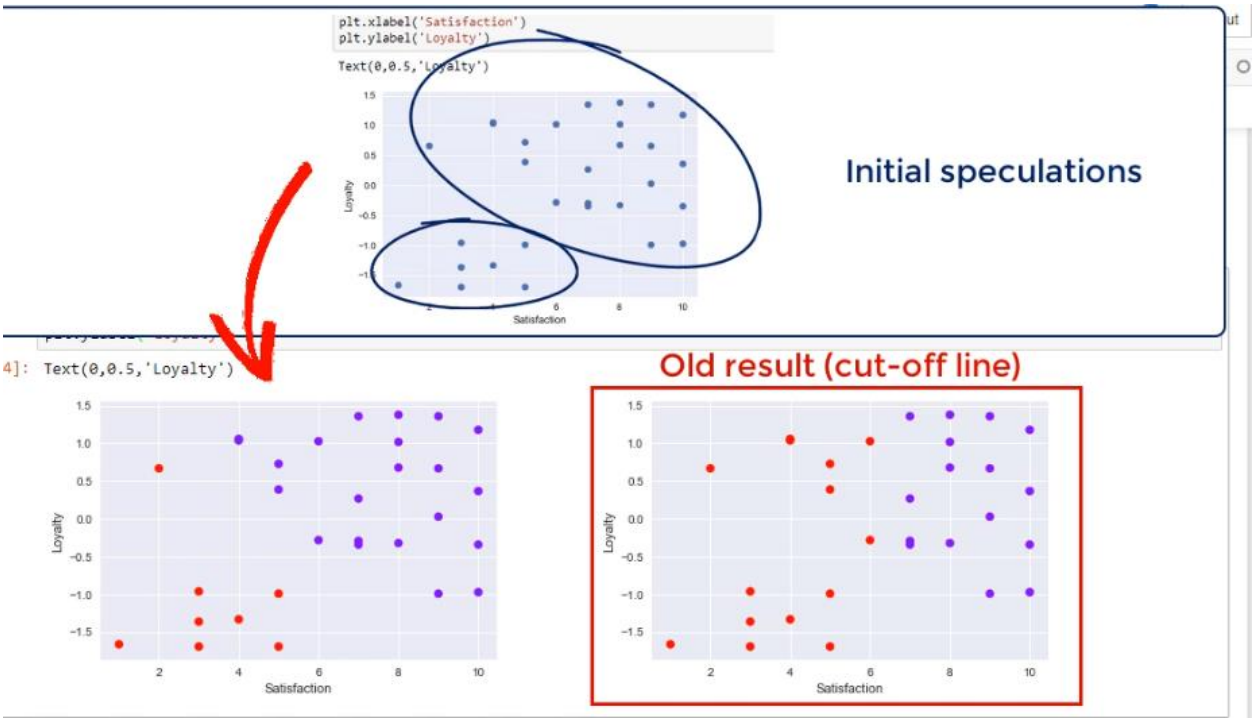
49]:

	Satisfaction	Loyalty	cluster_pred
0	4	-1.33	1
1	6	-0.28	0
2	5	-0.99	1
3	7	-0.29	0
4	4	1.06	0

Note: The "cluster_new" data frame includes the original values for Satisfaction and Loyalty, whereas the Predicted cluster is determined using the standardized values of Satisfaction and Loyalty.

In order to visualize the data, we have the option to plot it without standardizing the axes. However, it's essential to note that the ultimate solution is based on the standardized values. The following graph serves as a clearer illustration: when observing the graph on the left, where the original x-axis provides insights into the level of customer satisfaction, we gain an understanding of the satisfaction distribution. However, if we choose to represent this data with standardized values, a potential misconception arises. The central region of the two graphs differs; while the left graph indicates a midpoint of 5.5, the standardized graph places the zero midpoint, which corresponds to the mean of the variable in the left graph, at 6.4. This discrepancy underscores the importance of considering the standardized values for a more accurate interpretation

Upon comparing the recent findings with the previous results, a discernible pattern emerges, affirming that both dimensions—loyalty and satisfaction—were meticulously considered. This alignment corresponds closely with our initial speculations. Consequently, our confidence in the efficacy of standardization has grown, solidifying its merit as a superior approach.



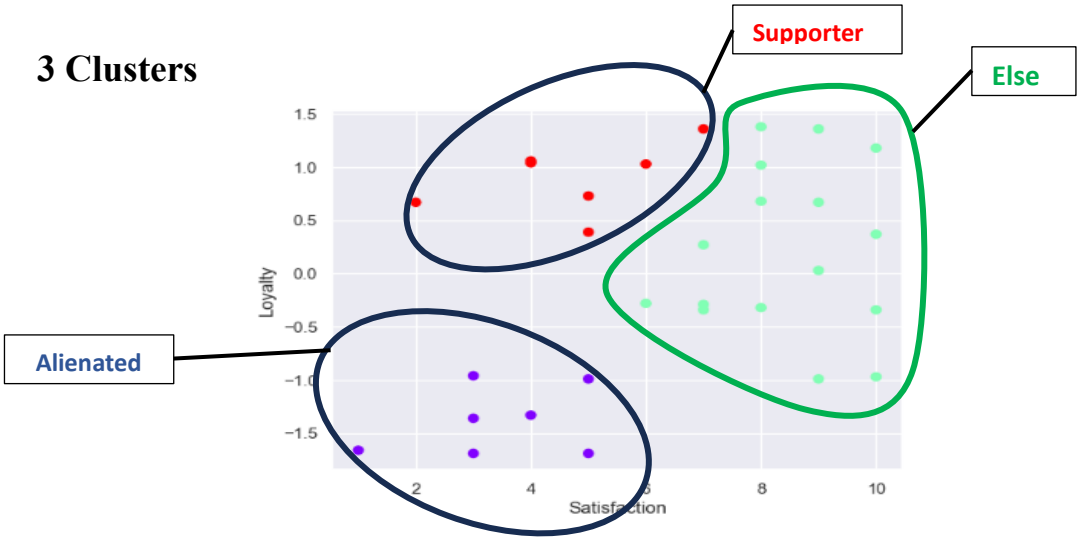
But the problem is not still solved as this two cluster solutions does not make complete sense as we find at the start, but at least it is a good starting point to reach out goal.

Optimizations of Clusters

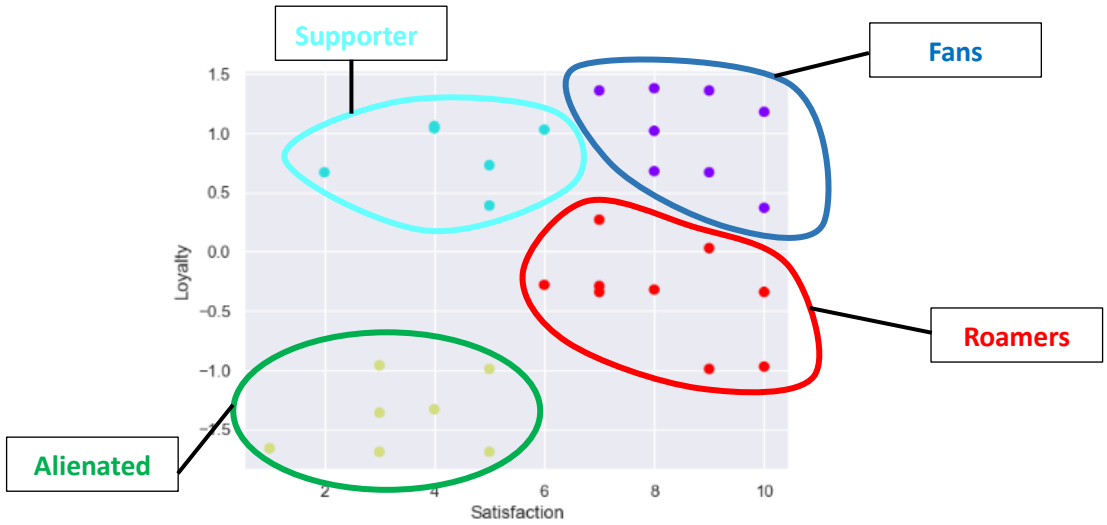
2 Clusters:

The figure on the right show the graph with 2 clusters, Randomly assigned name to each cluster for better understanding of the whole optimization.

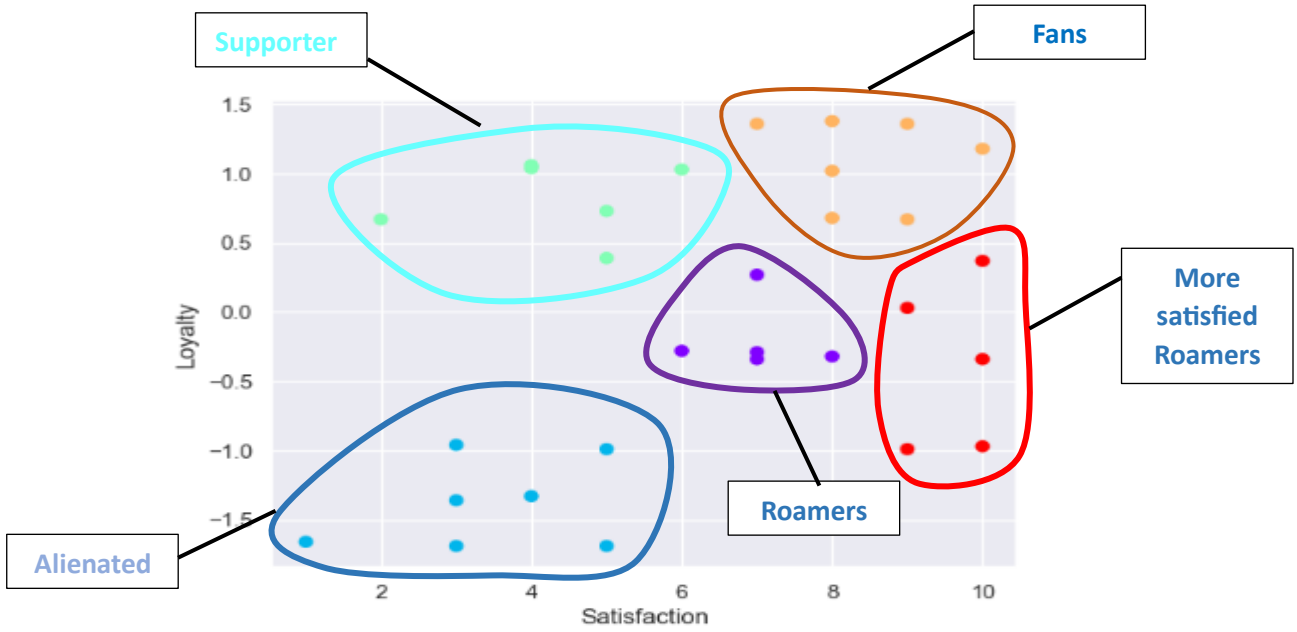
3 Clusters



4 Clusters

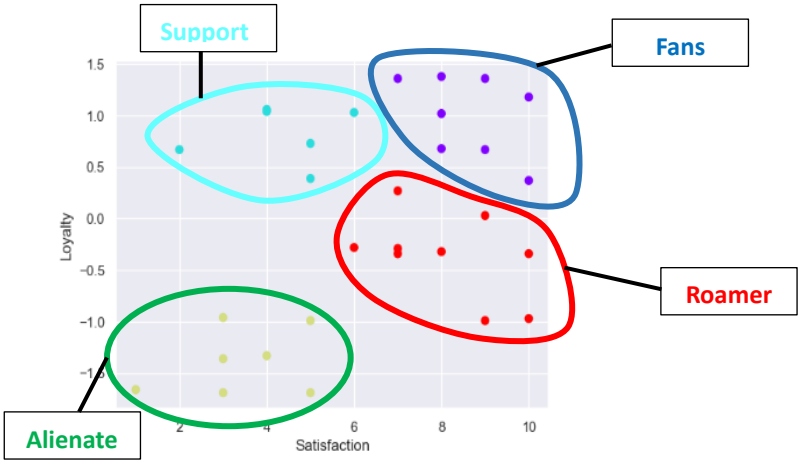


5 Clusters



Conclusion

We can make as many clusters as we want but clustering further will make the result more complex or sometime unreliable like naming and drawing conclusions. Depending on the problem at hand we can carry out further analysis but now for this specific problem 4 and 5 clusters solution is the best.



In formulating the company's strategy, the objective is to attract a broad customer base. Referring to the graph on the right, the company aims to cultivate a large and dedicated fan, ensuring both satisfaction and loyalty among customers.

The clustering method has been employed to discern distinct customer behaviors. This enables the development of tailored strategies for each cluster, such as Loyalty Programs or discounted coupons, with the aim of transitioning supporters, roamers, and alienated customers into fans.