Experiment No. 5
Apply appropriate Unsupervised Learning Technique on the Wholesale Customers Dataset
Date of Performance:
Date of Submission:

Aim: Apply appropriate Unsupervised Learning Technique on the Wholesale Customers Dataset.

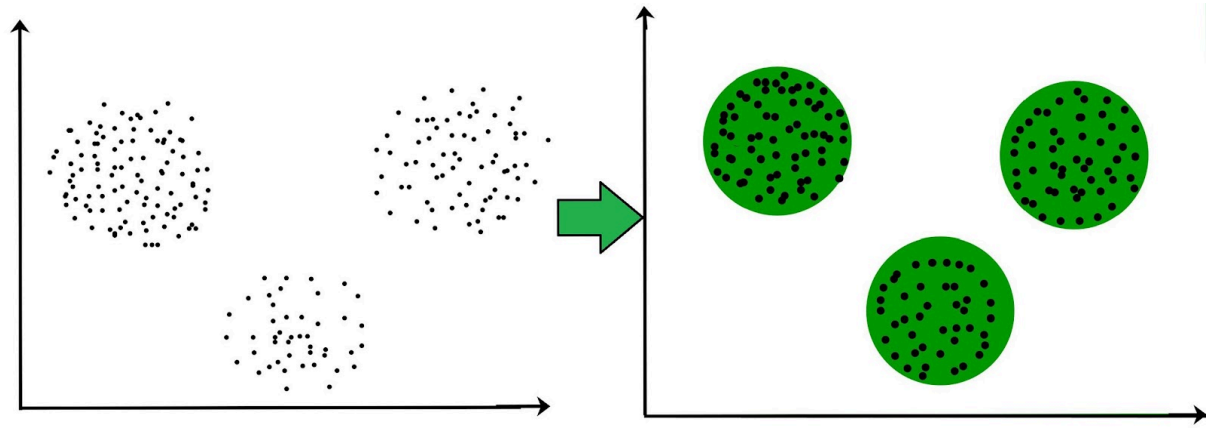
Objective: Able to perform various feature engineering tasks, apply Clustering Algorithm on the given dataset.

Theory:

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

For ex– The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.



Dataset:

This data set refers to clients of a wholesale distributor. It includes the annual spending in monetary units (m.u.) on diverse product categories. The wholesale distributor operating in different regions of Portugal has information on annual spending of several items in their stores across different regions and channels. The dataset consist of 440 large retailers annual spending on 6 different varieties of product in 3 different regions (lisbon , oporto, other) and across different sales channel (Hotel, channel)

Detailed overview of dataset

Records in the dataset = 440 ROWS

Columns in the dataset = 8 COLUMNS

FRESH: annual spending (m.u.) on fresh products (Continuous)

MILK:- annual spending (m.u.) on milk products (Continuous)

GROCERY:- annual spending (m.u.) on grocery products (Continuous)

FROZEN:- annual spending (m.u.) on frozen products (Continuous)

DETERGENTS_PAPER :- annual spending (m.u.) on detergents and paper products (Continuous)

DELICATESSEN:- annual spending (m.u.)on and delicatessen products (Continuous);

CHANNEL: - sales channel Hotel and Retailer

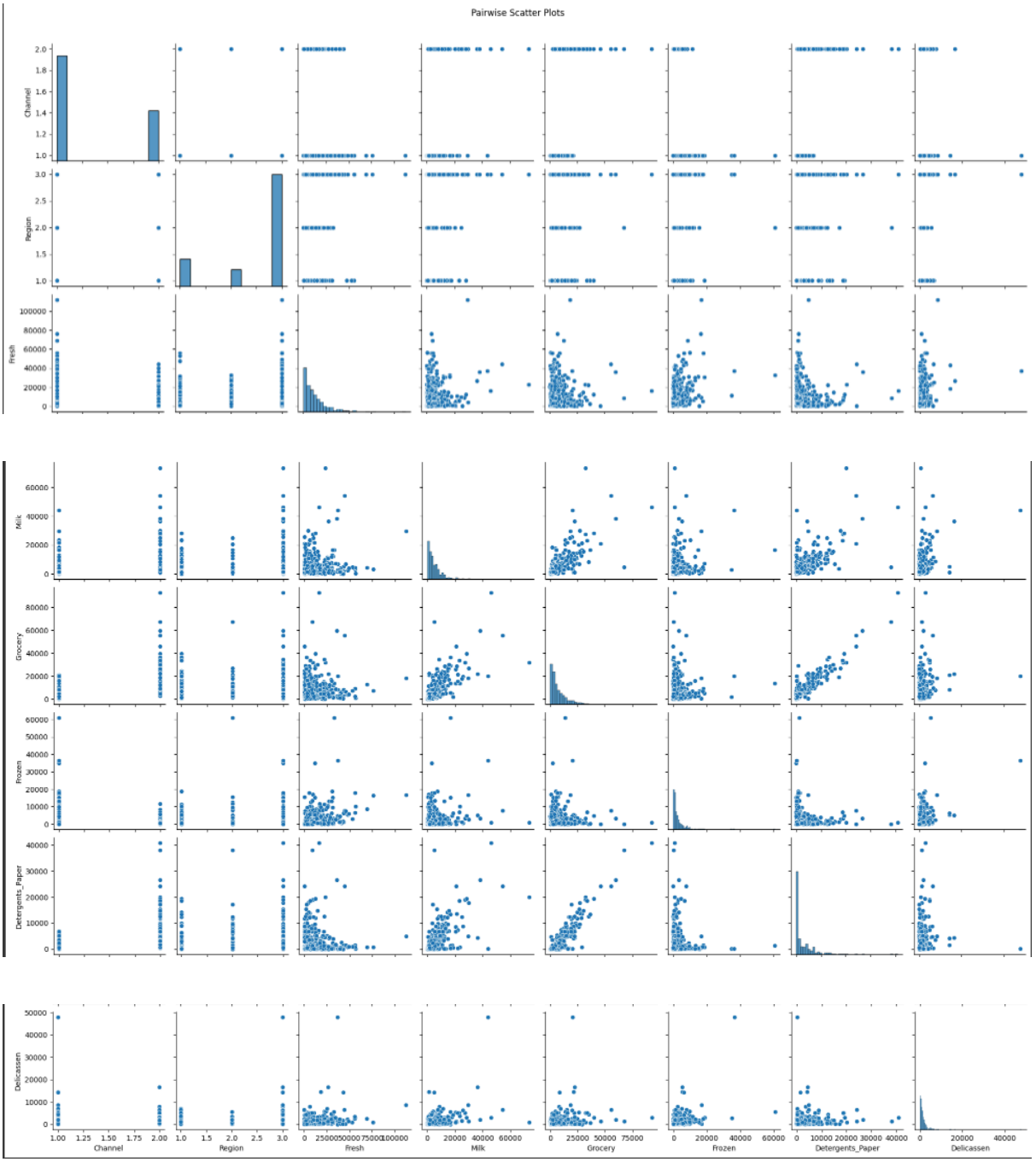
REGION:- three regions (Lisbon, Oporto, Other)

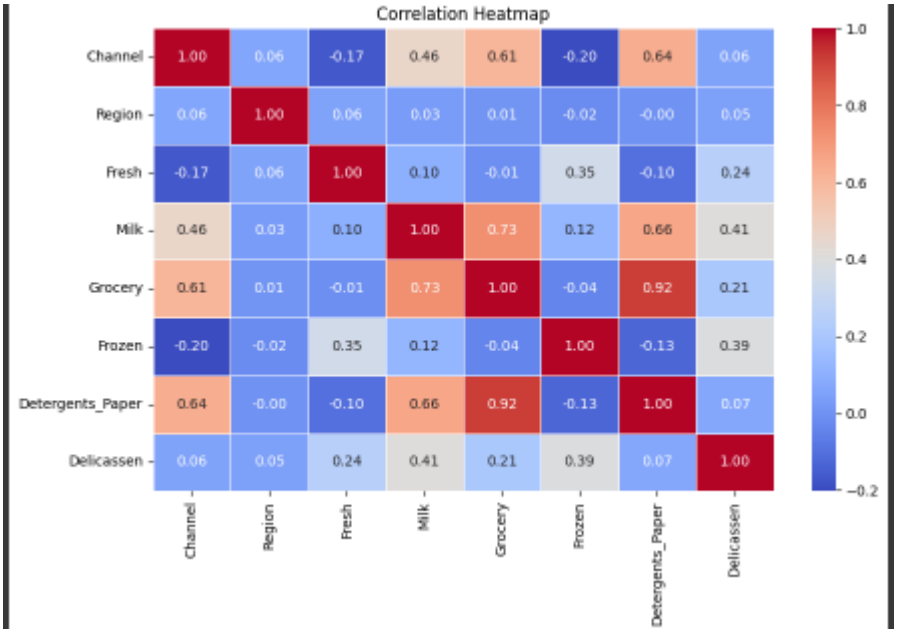
Code:

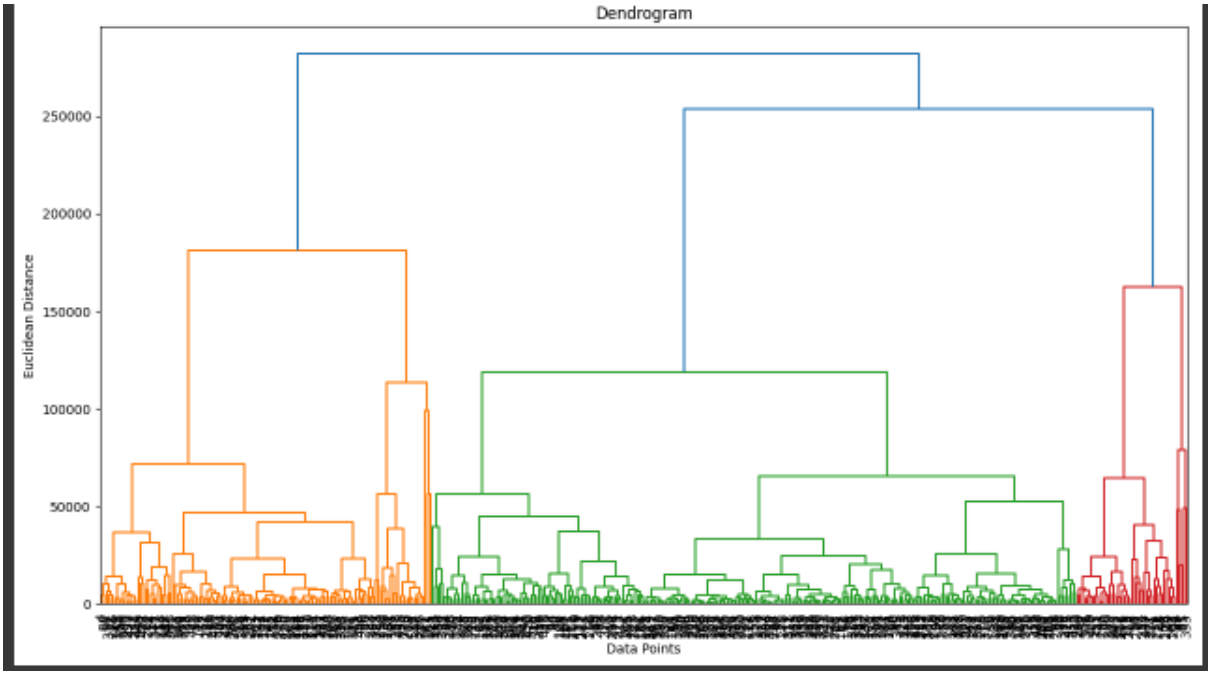
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv('/content/Wholesale customers data.csv')
print(data.isnull().sum())

data.describe()

plt.figure(figsize=(15, 10))
sns.pairplot(data)
plt.suptitle("Pairwise Scatter Plots", y=1.02)  # Adjust the title position
plt.show()

correlation_matrix = data.corr()
print(correlation_matrix)
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap")
plt.show()

from scipy.cluster.hierarchy import dendrogram, linkage
numerical_df = data.select_dtypes(include=['int64', 'float64'])
linkage_matrix = linkage(numerical_df, method='ward')

plt.figure(figsize=(15, 8))
dendrogram(linkage_matrix, leaf_rotation=90, leaf_font_size=10)
plt.title("Dendrogram")
plt.xlabel("Data Points")
plt.ylabel("Euclidean Distance")
plt.show()

from sklearn.cluster import KMeans

# Select numerical columns for clustering
numerical_df = data[['Fresh', 'Milk', 'Grocery', 'Frozen', 'Detergents_Paper', 'Delicassen']]

# Initialize an empty list to store the inertia values
inertia = []

# Calculate the inertia for different values of k (number of clusters)
```

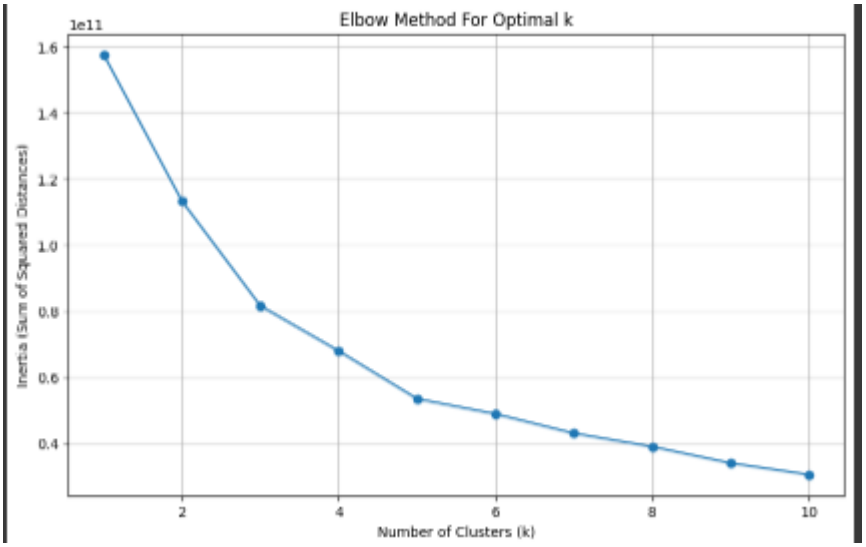
```
K_range = range(1, 11)  # You can change the range to explore more values
for k in K_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(numerical_df)
    inertia.append(kmeans.inertia_)

# Plot the elbow graph
plt.figure(figsize=(10, 6))
plt.plot(K_range, inertia, marker='o')
plt.title('Elbow Method For Optimal k')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Inertia (Sum of Squared Distances)')
plt.grid(True)
plt.show()

from sklearn.cluster import KMeans

numerical_df = data[['Fresh', 'Milk', 'Grocery', 'Frozen', 'Detergents_Paper', 'Delicassen']]

# Apply KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=42)  # You can change 'n_clusters' as needed
data['Cluster'] = kmeans.fit_predict(numerical_df)
print(data)

columns = numerical_df.columns

for col1, col2 in combinations(columns, 2):
    plt.figure(figsize=(10, 7))
    plt.scatter(numerical_df[col1], numerical_df[col2], c=data['Cluster'], cmap='viridis',
marker='o')
    plt.title(f'Scatter Plot: {col1} vs {col2}')
    plt.xlabel(col1)
    plt.ylabel(col2)
    plt.colorbar(label='Cluster Label')
    plt.grid(True)
    plt.show()
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
true_labels = data['Channel'].values  # Change to the appropriate true label column
predicted_clusters = data['Cluster'].values
accuracy = accuracy_score(true_labels, predicted_clusters)
precision = precision_score(true_labels, predicted_clusters, average='weighted', zero_division=0)
recall = recall_score(true_labels, predicted_clusters, average='weighted', zero_division=0)
f1 = f1_score(true_labels, predicted_clusters, average='weighted', zero_division=0)
print(f'Accuracy: {accuracy:.2f}')
print(f'Precision: {precision:.2f}')
print(f'Recall: {recall:.2f}')
print(f'F1 Score: {f1:.2f}')
```

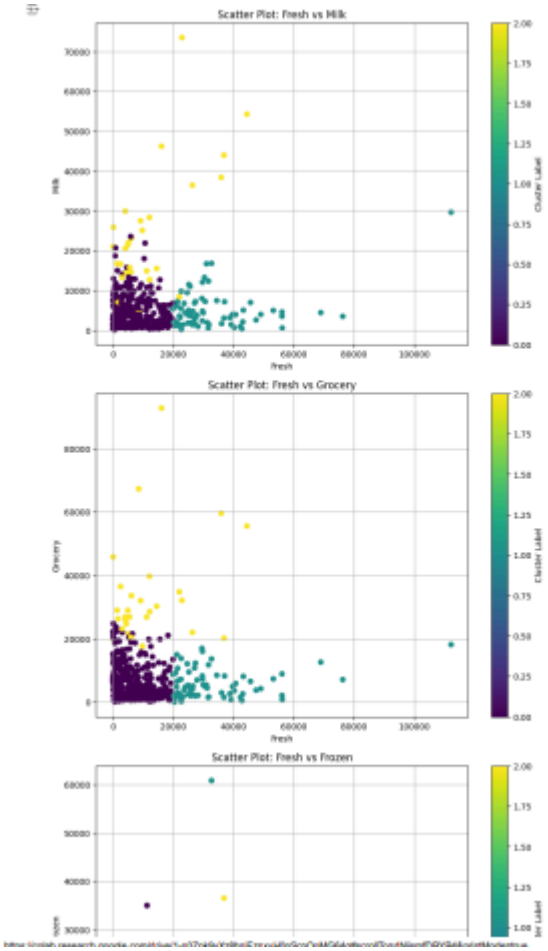
Output:

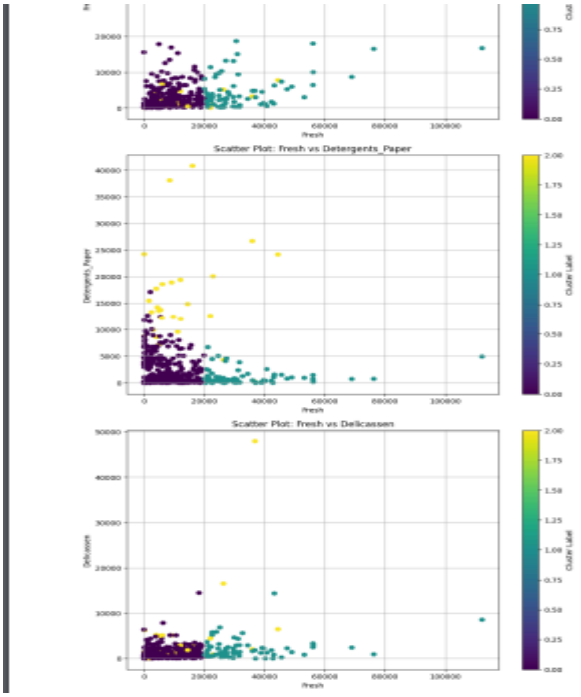

Pairwise Scatter Plots
Channel
Region
Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicassen

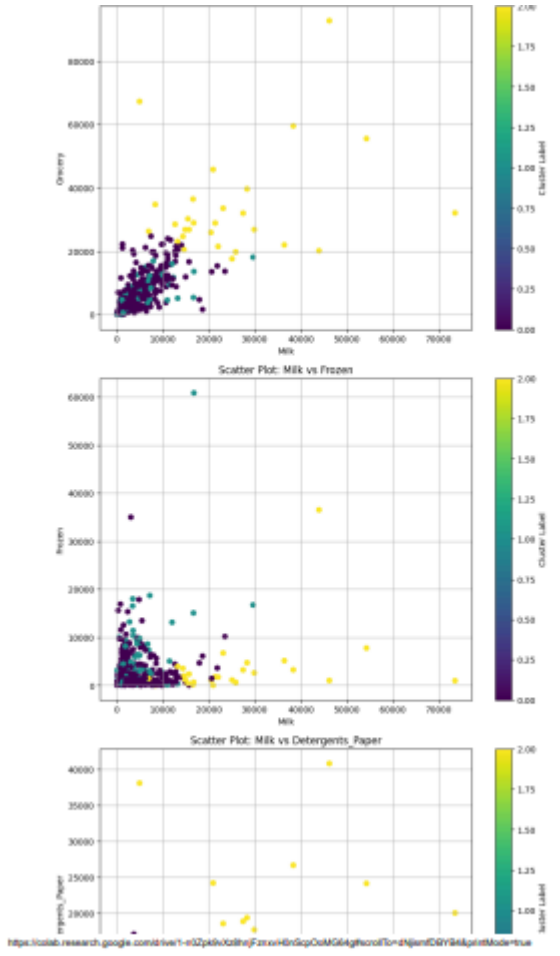

Correlation Heatmap
Channel
Region
Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicassen
1.00 0.06 -0.17 0.46 0.61 -0.20 0.64 0.06
0.06 1.00 0.06 0.03 0.01 -0.02 -0.00 0.05
-0.17 0.06 1.00 0.10 -0.01 0.35 -0.10 0.24
0.46 0.03 0.10 1.00 0.73 0.12 0.66 0.41
0.61 0.01 -0.01 0.73 1.00 -0.04 0.92 0.21
-0.20 -0.02 0.35 0.12 -0.04 1.00 -0.13 0.39
0.64 -0.00 -0.10 0.66 0.92 -0.13 1.00 0.07
0.06 0.05 0.24 0.41 0.21 0.39 0.07 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


Dendrogram
Euclidean Distance
250000
200000
150000
100000
50000
0
Data Points


Elbow Method For Optimal k
1e11
Inertia (Sum of Squared Distances)
Number of Clusters (k)


Scatter Plot: Fresh vs Milk
Scatter Plot: Fresh vs Grocery
Scatter Plot: Fresh vs Frozen


Scatter Plot: Fresh vs Detergents_Paper
Scatter Plot: Fresh vs Delicassen


Scatter Plot: Milk vs Frozen
Scatter Plot: Milk vs Detergents_Paper

Conclusion:

The obtained performance metrics indicate a mixed outcome regarding the clustering results:

Accuracy: 0.20

This low accuracy suggests that only 20% of the predicted clusters match the true labels. In clustering scenarios, especially with KMeans, this can often happen when the clusters formed do not align well with the actual classes. This indicates that the model is not effective at grouping the data correctly.

Precision: 0.88

Precision measures the ratio of true positive predictions to the total predicted positives. A precision of 0.88 indicates that when the model predicts a sample to be in a certain cluster, it is correct 88% of the time. This suggests that the clusters formed are relatively pure, even if they are not well-aligned with the true labels.

Recall: 0.20

Recall measures the ratio of true positive predictions to the total actual positives. A recall of 0.20 indicates that the model only identifies 20% of the actual positive samples. This shows that many true instances are being missed, which can be problematic if the goal is to capture as many relevant cases as possible.

F1 Score: 0.33

The F1 score is the harmonic mean of precision and recall, providing a balance between the two. An F1 score of 0.33 indicates that while the model is precise when it predicts a positive case, it struggles to find a large portion of the positive cases (as reflected in the low recall). This score reinforces the idea that the model's clustering is not effective overall.