Customer Segmentation by Credit Card Data
Unsupervised Learning (Kmeans and GMM)

MET CS 777 Term Project

- Sushant Khot

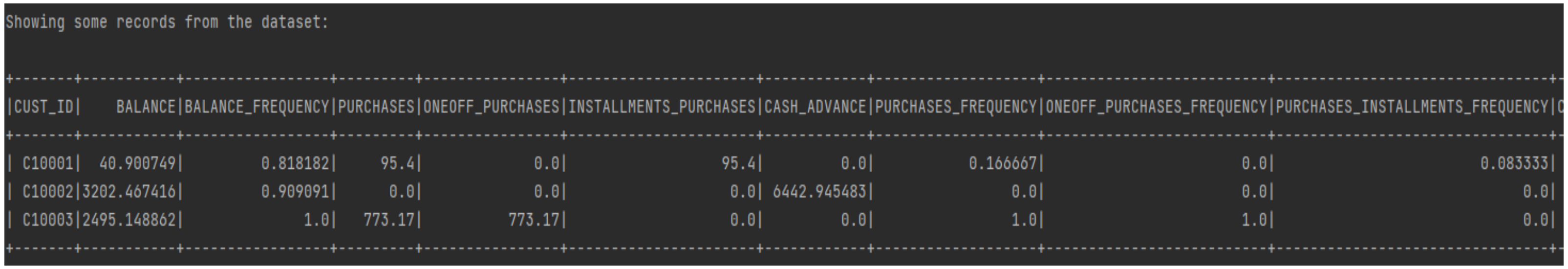
Agenda

- Introduction and Research Questions
- Data Pre-Processing Steps
- KMeans Clustering Model
  - Silhouette Score
  - Elbow Method
- GMM (Gaussian Mixture Model)
- Training and Testing the Models (70:30 Split)
- Accuracy and Confusion Matrix

Introduction

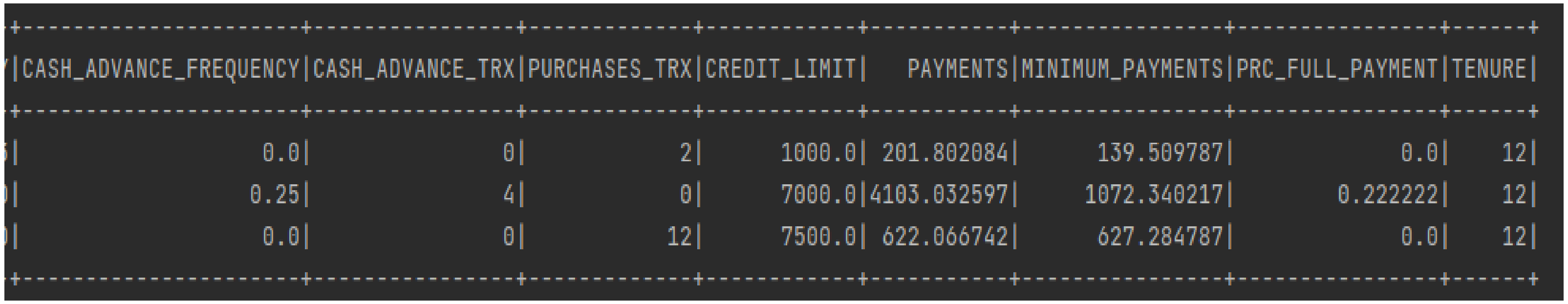
- This Term project is to analyze and segment Customers based on available Credit Card data.
- The sample Dataset summarizes the usage behavior of about 9000 active credit card holders during the last 6 months.
- The file consists of customer credit card data with 18 behavioral variables of the customer.
- The description of features / columns of Credit Card dataset :-
    - **CUST_ID** : Identification of Credit Card holder (Categorical)
    - **BALANCE** : Balance amount left in their account to make purchases
    - **BALANCE_FREQUENCY** : How frequently the Balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated)
    - **PURCHASES** : Amount of purchases made from account
    - **ONEOFF_PURCHASES** : Maximum purchase amount done in one-go
    - **INSTALLMENTS_PURCHASES** : Amount of purchase done in installment
    - **CASH_ADVANCE** : Cash in advance given by the user
    - **PURCHASES_FREQUENCY** : How frequently the Purchases are being made, score between 0 and 1
    - (1 = frequently purchased, 0 = not frequently purchased)
    - **ONEOFF_PURCHASES_FREQUENCY** : How frequently Purchases are happening in one-go (1 = frequently purchased, 0 = not frequently purchased)
    - **PURCHASES_INSTALLMENTS_FREQUENCY** : How frequently purchases in installments are being done (1 = frequently done, 0 = not frequently done)
    - **CASH_ADVANCE_FREQUENCY** : How frequently the cash in advance being paid
    - **CASH_ADVANCE_TRX** : Number of Transactions made with "Cash in Advanced"
    - **PURCHASES_TRX** : Number of purchase transactions made
    - **CREDIT_LIMIT** : Limit of Credit Card for user
    - **PAYMENTS** : Amount of Payment done by user
    - **MINIMUM_PAYMENTS** : Minimum amount of payments made by user
    - **PRC_FULL_PAYMENT** : Percent of full payment paid by user
    - **TENURE** : Tenure of credit card service for user

Introduction Cont'd



```
Showing some records from the dataset:

+-------+-----------+-----------------+---------+----------------+----------------------+------------+-------------------+--------------------------+--------------------------------+-
|CUST_ID|    BALANCE|BALANCE_FREQUENCY|PURCHASES|ONEOFF_PURCHASES|INSTALLMENTS_PURCHASES|CASH_ADVANCE|PURCHASES_FREQUENCY|ONEOFF_PURCHASES_FREQUENCY|PURCHASES_INSTALLMENTS_FREQUENCY|C
+-------+-----------+-----------------+---------+----------------+----------------------+------------+-------------------+--------------------------+--------------------------------+-
| C10001|  40.900749|         0.818182|     95.4|             0.0|                  95.4|         0.0|           0.166667|                       0.0|                        0.083333|
| C10002|3202.467416|         0.909091|      0.0|             0.0|                   0.0| 6442.945483|                0.0|                       0.0|                             0.0|
| C10003|2495.148862|              1.0|   773.17|          773.17|                   0.0|         0.0|                1.0|                       1.0|                             0.0|
+-------+-----------+-----------------+---------+----------------+----------------------+------------+-------------------+--------------------------+--------------------------------+-
```



```
-+----------------------+----------------+-------------+------------+-----------+----------------+----------------+------+
Y|CASH_ADVANCE_FREQUENCY|CASH_ADVANCE_TRX|PURCHASES_TRX|CREDIT_LIMIT|   PAYMENTS|MINIMUM_PAYMENTS|PRC_FULL_PAYMENT|TENURE|
-+----------------------+----------------+-------------+------------+-----------+----------------+----------------+------+
3|                   0.0|               0|            2|      1000.0| 201.802084|      139.509787|             0.0|    12|
0|                  0.25|               4|            0|      7000.0|4103.032597|     1072.340217|        0.222222|    12|
0|                   0.0|               0|           12|      7500.0| 622.066742|      627.284787|             0.0|    12|
-+----------------------+----------------+-------------+------------+-----------+----------------+----------------+------+
```

Research Questions:

1. Can we try to segment Customers into Clusters to identify which group is spending high Amount on Purchases using their Credit Cards?

2. Are there any Customers that have High Credit Limit but are not spending high on Purchases using their credit Cards?

3. Are their any Customers that have High Balance available and are NOT spending much on Purchases using their Credit Cards?

4. How Accurately are our Clustering Models identifying these Customer groups?

Data Pre-Processing Steps

- We have removed all rows that had missing values bringing down the number of rows count to 8636.

- We have combined the input columns from the dataset that will be used for analysis using VectorAssembler.
- **VectorAssembler** is a transformer that combines a given list of columns into a single vector column.
- It is useful for combining raw features and features generated by different feature transformers into a single feature vector.

- We have also scaled our features using Standard Scalar.
- **StandardScaler** transforms a dataset of Vector rows, normalizing each feature to have unit standard deviation and/or zero mean.

- Additionally, we have done **Dimensionality reduction** of our features using PCA.
- It is the process of reducing the number of variables under consideration.
- It can be used to extract latent features from raw and noisy features or compress data while maintaining the structure.

- We will apply tis to our dataframe on the "standardized" column to reduce it to a vector of 2 elements from 17 using PCA.
- **Principal component analysis (PCA)** is a statistical method to find a rotation such that the first coordinate has the largest variance possible, and each succeeding coordinate, in turn, has the largest variance possible.

Implementation of KMeans – Unsupervised Learning

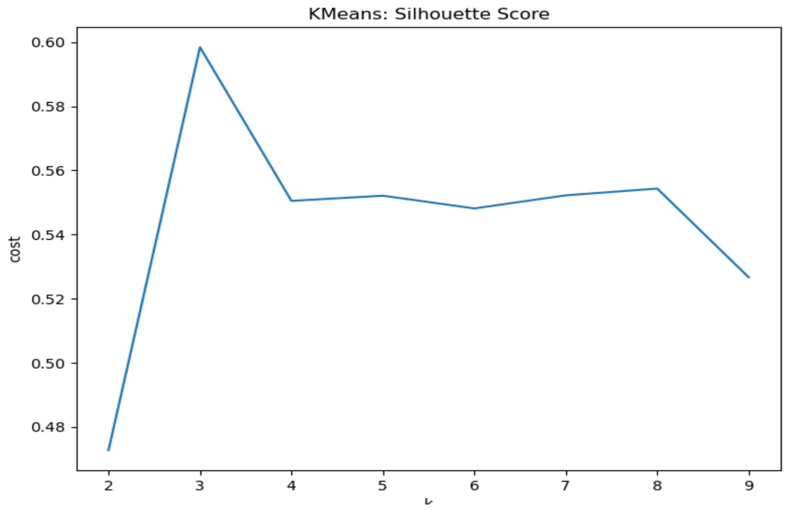
- K-Means is an unsupervised machine learning algorithm that groups data into k number of clusters.
- The number of clusters (k) is user-defined and the algorithm will try to group the data even if this number is not optimal for the specific case.

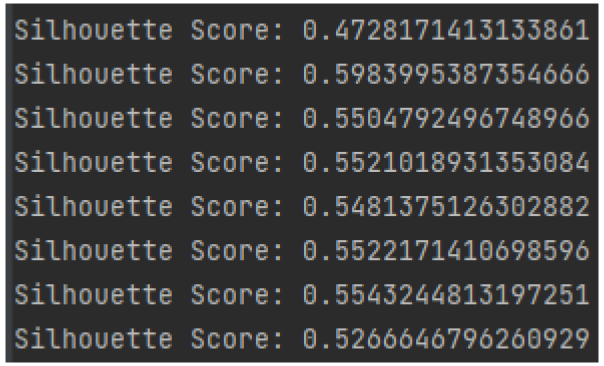
- We need to find the Optimal number of clusters (k) in order to get good clustering results.
- We will approach this problem in 2 different ways:
- 1) Calculating Silhouette Score
- 2) Elbow method.

Silhouette score:
- Silhouette score is used to evaluate the quality of clusters created using clustering algorithms such as K-Means in terms of how well samples are clustered with other samples that are similar to each other.
- The Silhouette score is calculated for each sample of different clusters.

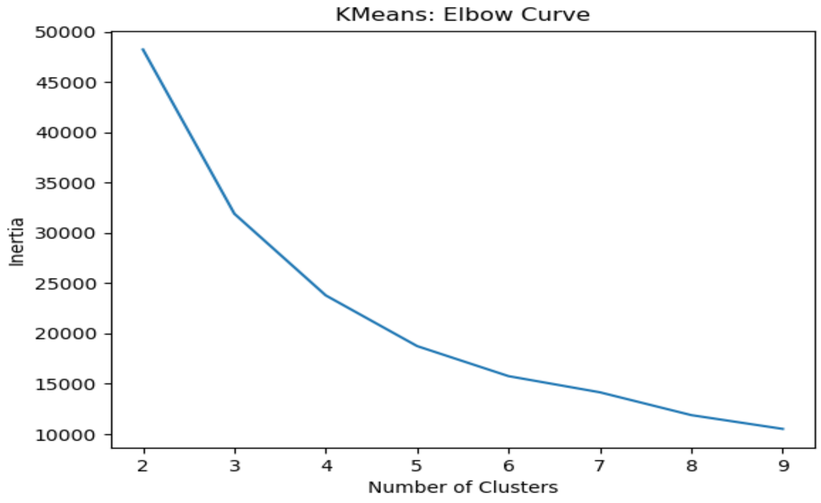
The Elbow Method:
- The Elbow method is a very popular technique, and the idea is to run k-means clustering for a range of clusters k (let's say from 1 to 10) and for each value,
- we are calculating the sum of squared distances from each point to its assigned center(distortions).
- When the distortions are plotted, and the plot looks like an arm then the "elbow"(the point of inflection on the curve) is the best value of k.

Silhouette Score and Elbow Method Plots





```
Silhouette Score: 0.4728171413133861
Silhouette Score: 0.5983995387354666
Silhouette Score: 0.5504792496748966
Silhouette Score: 0.5521018931353084
Silhouette Score: 0.5481375126302882
Silhouette Score: 0.5522171410698596
Silhouette Score: 0.5543244813197251
Silhouette Score: 0.5266646796260929
```

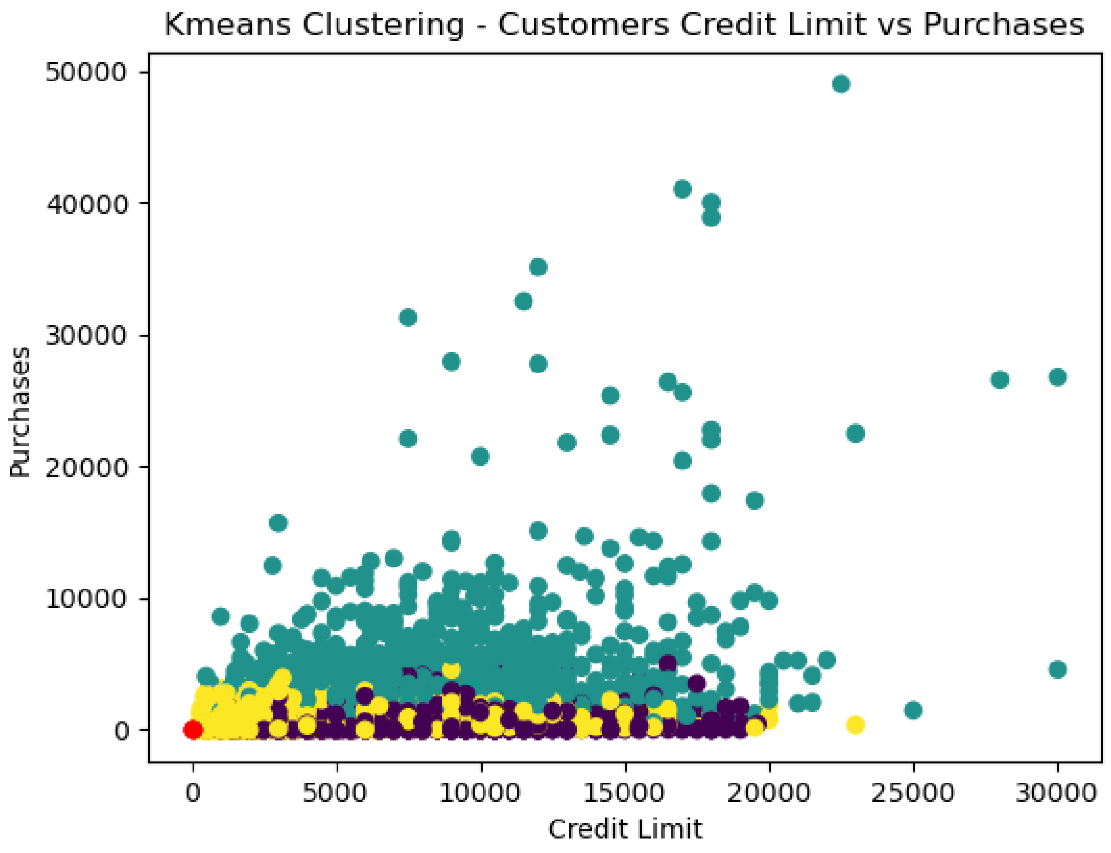


K = 3 is our Optimal value

Kmeans Cont'd

- After Training the Kmeans Model with K = 3, we can calculate the Label / Class Column: [0, 1 , 2] assigned to the dataset.
- The K = 3 Centroids from the KMeans mode: [array([-2.23076312, 4.18438519]), array([-7.29058669, 1.60678818]), array([-2.72764428, 0.72389284])]

- We will now plot the clusters for Customer Credit Card's Credit Limit vs Purchases made using Kmeans.



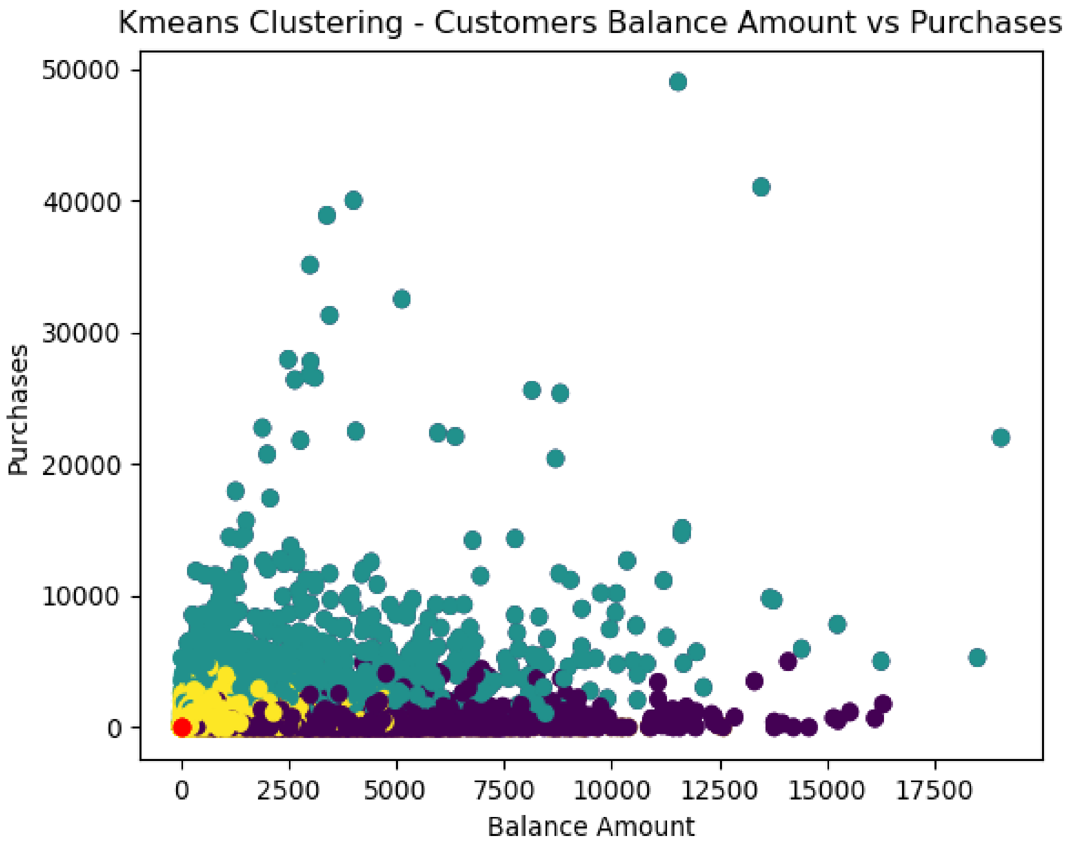
- **cluster 0: yellow :** Customers with Low to High Credit Limit and Low Amount of Purchases on their credit cards.

- **cluster 1: purple:** Medium to High Credit Limit but Low Amount of Purchases on their credit cards.

- **cluster 2: green:** Low to High Credit Limit and High Amount of Purchases on their credit cards.

Kmeans Cont'd

- Similarly, We can now plot the clusters for Customer Credit Card's Balance vs Purchases made using KMeans.

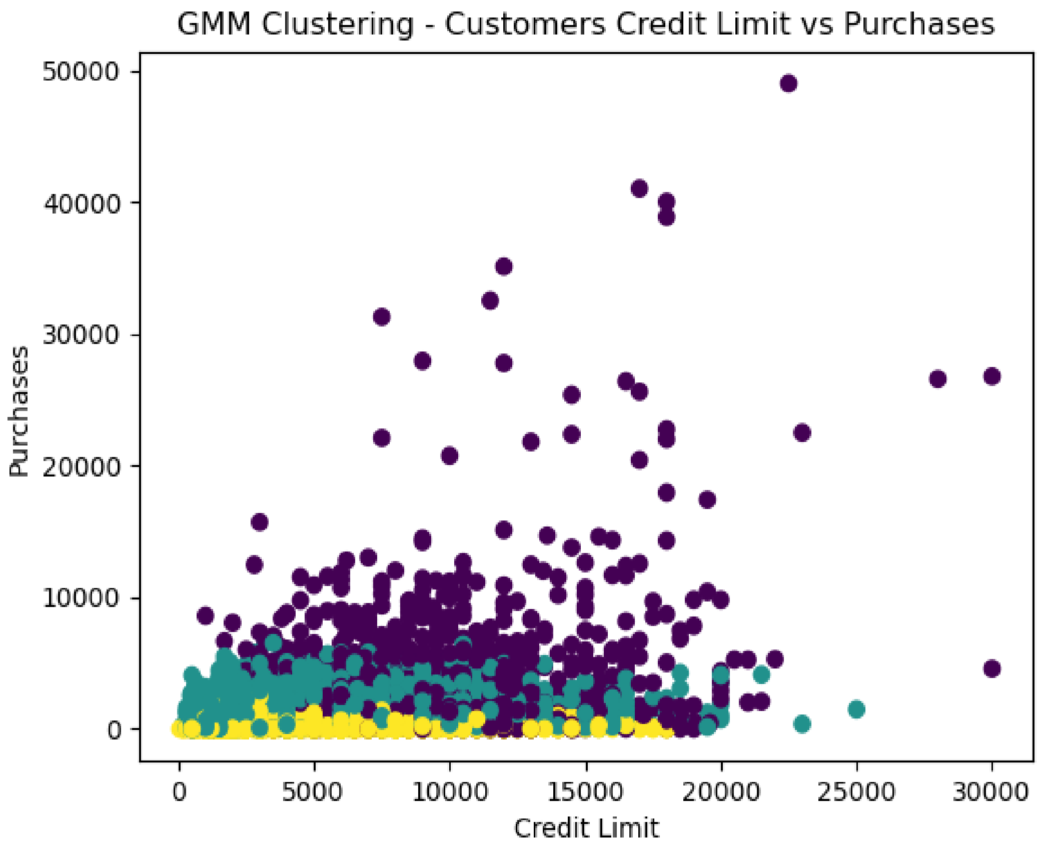


- **cluster 0: yellow :** Customers with Low Balance and Low Amount of Purchases on their credit cards.
- **cluster 1: purple:** Customers with Low to High Balance but Low Amount of Purchases on their credit cards.
- **cluster 2: green:** Customers with Low to High Balance but High Amount of Purchases on their credit cards.

Implementation of Gaussian Mixture Model (GMM) – Unsupervised Learning

- A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.
- After Training the model, we can see "probability" column added to the dataframe.

- We will now plot the clusters for Customer Credit Card's Credit Limit vs Purchases made using GMM.



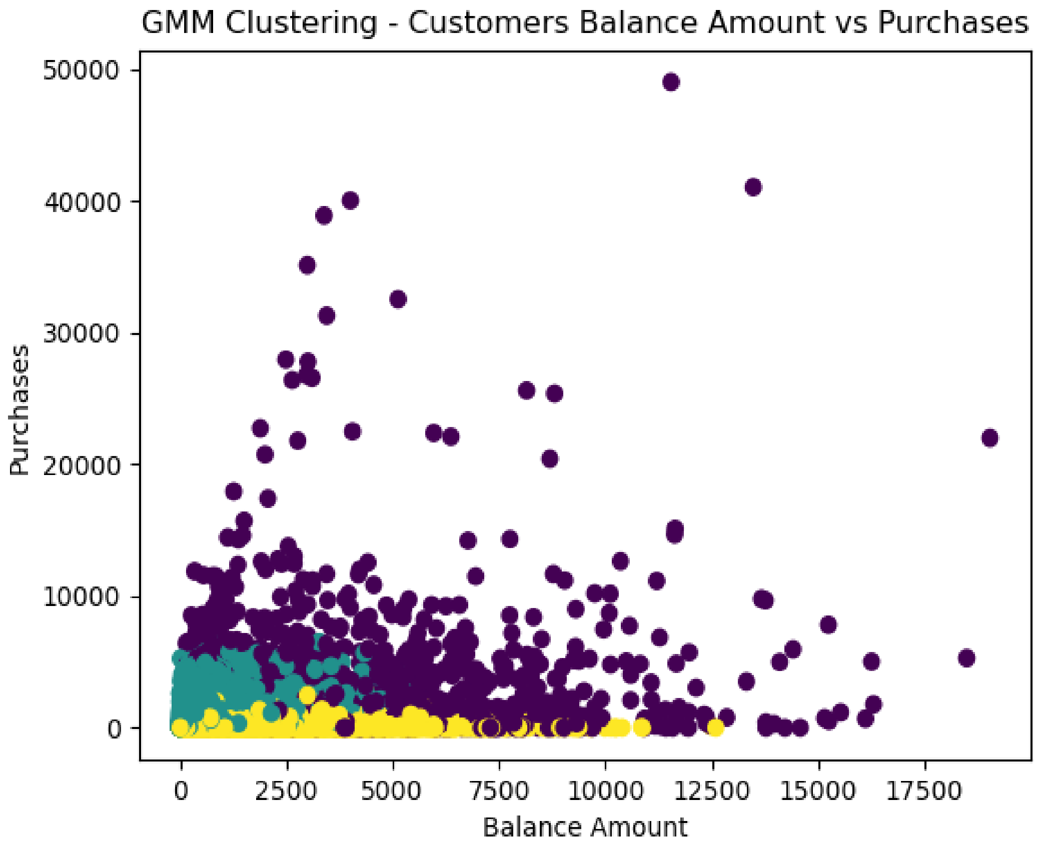
- **cluster 0: yellow:** Low Amount of purchases done by Customers having low to high Credit Limit.

- **cluster 1: green:** Low to Medium amount of purchases done by Customers having low to high Credit Limit.

- **cluster 2: purple:** Medium to high Amount of Purchases done by Customers having low to high Credit Limit.

GMM Cont'd

- We will now plot the clusters for Customer Credit Card's Balance vs Purchases made using GMM.



- **cluster 0: yellow:** Low Amount of Purchases done by Customers having low to high Balance on their credit cards.
- **cluster 1: green:** Low to Medium amount of Purchases done by Customers having low Balance on their credit cards.
- **cluster 2: purple:** Medium to high Amount of Purchases done by Customers having low to high Balance on their credit cards.

Training and Testing of the Models (70:30) split. (Confusion Matrix Scores)

- We split our dataset into Training (70%) and Testing (30%). We have also combined the cluster labels to 2 major classes:
- 0 = Customers with Low Credit / Balance and Low Purchases. (Negative Event)
- 1 = Customers with Med-High Credit / Balance and Med-High Purchases. (Positive Event)

- After Training the “Train” dataset, we use the model on our “Test” dataset and find out the Class / Labels and Probabilities for the KMeans and GMM Models.

- We Calculated the Confusion Matrix and the different Scores for both the Models based on out Test dataset outcomes:

KMeans Model:

- **Accuracy:** 0.971
- **Precision:** 0.998
- **Recall:** 0.965
- **F1_Score:** 0.981

GMM:

- **Accuracy:** 0.274
- **Precision:** 0.746
- **Recall:** 0.302
- **F1_Score:** 0.43

Observations:

- **Accuracy**: In terms of Accuracy, KMeans model is a lot Accurate in clustering our Customers based on their Credit Card data compared to GMM. In our case, the Accuracy of GMM is very low compared to KMeans.

- **Precision**: It is a score that tells us: Out of all the positive predicted, what percentage is truly positive (Med-High Purchase Customers). In this case we see that again KMeans has performed better than compared to GMM.

- **Recall**: It is a score that tells us: Out of the total positive, what percentage are predicted positive (Med-High Purchase Customers). In this case we see that KMeans has performed much better than compared to GMM.

- **F1 Score**: It is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively, it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. In this case we see that KMeans has performed much better than compared to GMM.

In our case, KMeans has outperformed GMM.

Research Questions – Answered below:

1) Can we try to segment Customers into Clusters to identify which group is spending high Amount on Purchases using their Credit Cards?
Yes, we were able to cluster the customers based on their credit card data to find who are spending High Amount on Purchases.

2) Are there any Customers that have High Credit Limit but are not spending high on Purchases using their credit Cards?
Yes, we were able to cluster the customers based on their credit card data to find Customers with high Credit Limit and **not** spending much on Amount on Purchases.

3) Are their any Customers that have High Balance available and are NOT spending much on Purchases using their Credit Cards?
Yes, we have found a cluster of Customers who have high Balance available on the Credit Card and not spending much on Purchases.

4) How Accurately are our Clustering Models identifying these Customer groups?
KMeans model has more accurately segmented our customers than GMM based on their credit card data. We have calculated their Accuracies and other scores.

Thank You!

<u>My Details:</u>

Sushant Mohan Khot

Email: sushantk@bu.edu
BU ID: U73866118