Customer segmentation of credit card holders using unsupervised learning

--PCA, K-means and Hierarchical clustering

Chengming He

https://github.com/Chengminghe/ds5510.git

1. Problem description

Customer segmentation (also known as market segmentation) is a marketing technique that groups customers into groups. Grouping can be based on geography, demographics, consumer psychology, behavior, and other characteristics. The key to effectively segmenting customers is to group them based on a prediction of their value. Then use a different strategy for each group of customers to get the most value from high- and low-margin customers.

In this project, I'll use a credit card usage data about 9,000 active card holders during 6 months to cluster these customers using **unsupervised learning approaches**, including **PCA, K-means, and Hierarchical clustering**. The results will uncover the underlying relations among customers, which will enable the bank to promote customized products to card users, hence increasing revenues of the company as well as customer experiences.

2. Exploratory data analysis

2.1 Data description

Data source: https://www.kaggle.com/datasets/arjunbhasin2013/ccdata?datasetId=14701.

The dataset summarizes the usage behavior of about 9000 active credit card holders during the last 6 months. The file is at a customer level with 18 behavioral variables.

`CUSTID` : Identification of Credit Card holder (Categorical)
`BALANCE` : Balance amount left in their account to make purchases
`BALANCEFREQUENCY` : 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
`ONEOFFPURCHASES` : Maximum purchase amount done in one-go
`INSTALLMENTSPURCHASES` : Amount of purchase done in installment
`CASHADVANCE` : Cash in advance given by the user
`PURCHASESFREQUENCY` : How frequently the Purchases are being made, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)
`ONEOFFPURCHASESFREQUENCY` : How frequently Purchases are happening in one-go (1 = frequently purchased, 0 = not frequently purchased)
`PURCHASESINSTALLMENTSFREQUENCY` : How frequently purchases in installments are being done (1 = frequently done, 0 = not frequently done)
`CASHADVANCEFREQUENCY` : How frequently the cash in advance being paid
`CASHADVANCETRX` : Number of Transactions made with "Cash in Advanced"
`PURCHASESTRX` : Numbe of purchase transactions made
`CREDITLIMIT` : Limit of Credit Card for user
`PAYMENTS` : Amount of Payment done by user
`MINIMUM_PAYMENTS` : Minimum amount of payments made by user
`PRCFULLPAYMENT` : Percent of full payment paid by user
`TENURE` : Tenure of credit card service for user

2.2 Data cleaning

- **Check NA**

```
#check NA
data.isnull().sum()
```

```
CUST_ID                             0
BALANCE                             0
BALANCE_FREQUENCY                   0
PURCHASES                           0
ONEOFF_PURCHASES                    0
INSTALLMENTS_PURCHASES              0
CASH_ADVANCE                        0
PURCHASES_FREQUENCY                 0
ONEOFF_PURCHASES_FREQUENCY          0
PURCHASES_INSTALLMENTS_FREQUENCY    0
CASH_ADVANCE_FREQUENCY              0
CASH_ADVANCE_TRX                    0
PURCHASES_TRX                       0
CREDIT_LIMIT                        1
PAYMENTS                            0
MINIMUM_PAYMENTS                  313
PRC_FULL_PAYMENT                    0
TENURE                              0
dtype: int64
```

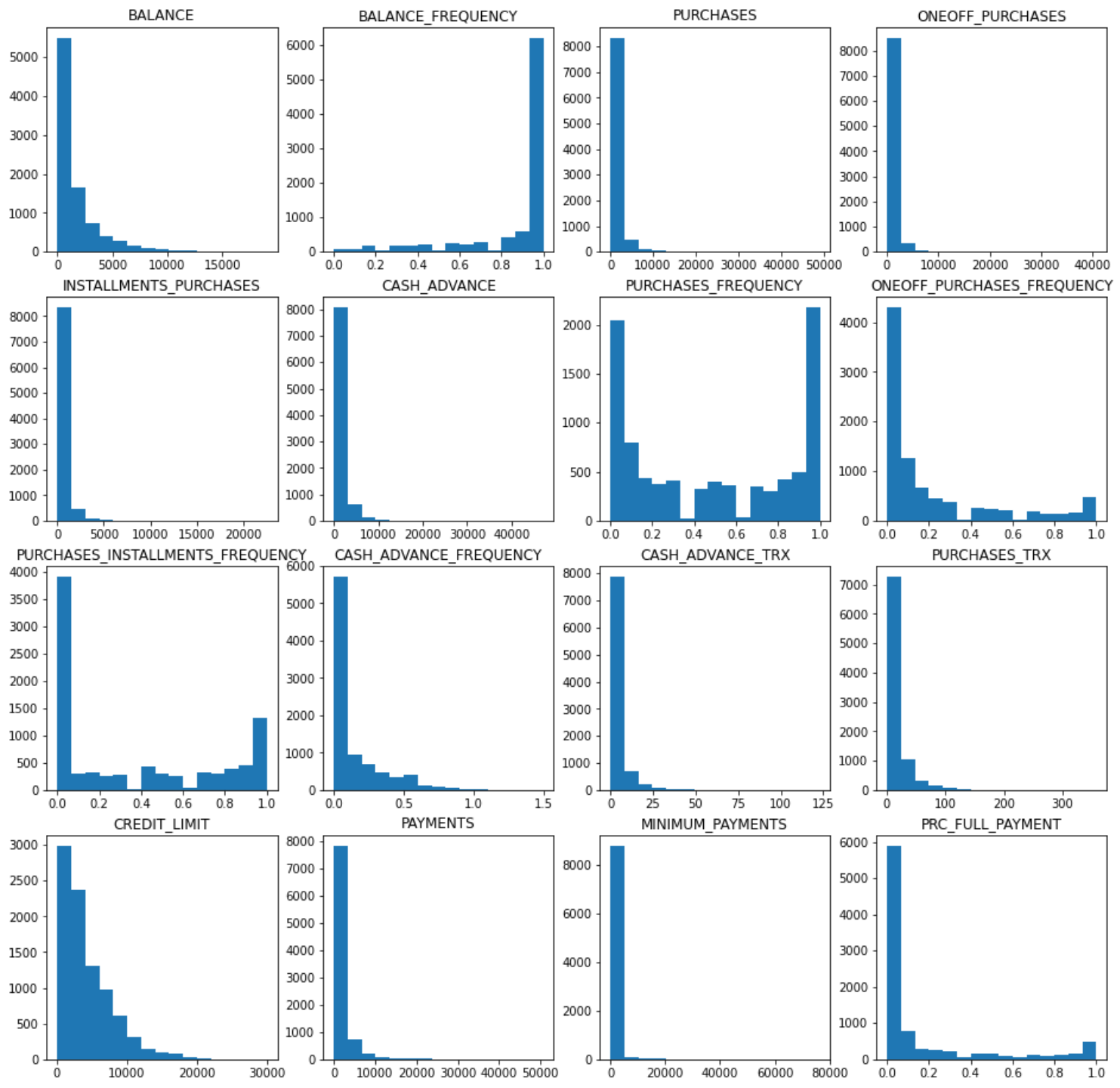
- **Imputing by median and add another attribute indicating whether imputed**

```
##imputing by median
data['IMPUTED'] = 0
data.loc[data['MINIMUM_PAYMENTS'].isnull(),'IMPUTED'] = 1
data.loc[data['MINIMUM_PAYMENTS'].isnull(),'MINIMUM_PAYMENTS'] = data['MINIMUM_PAYMENTS'].median()
data.loc[data['CREDIT_LIMIT'].isnull(),'CREDIT_LIMIT'] = data['CREDIT_LIMIT'].median()
```

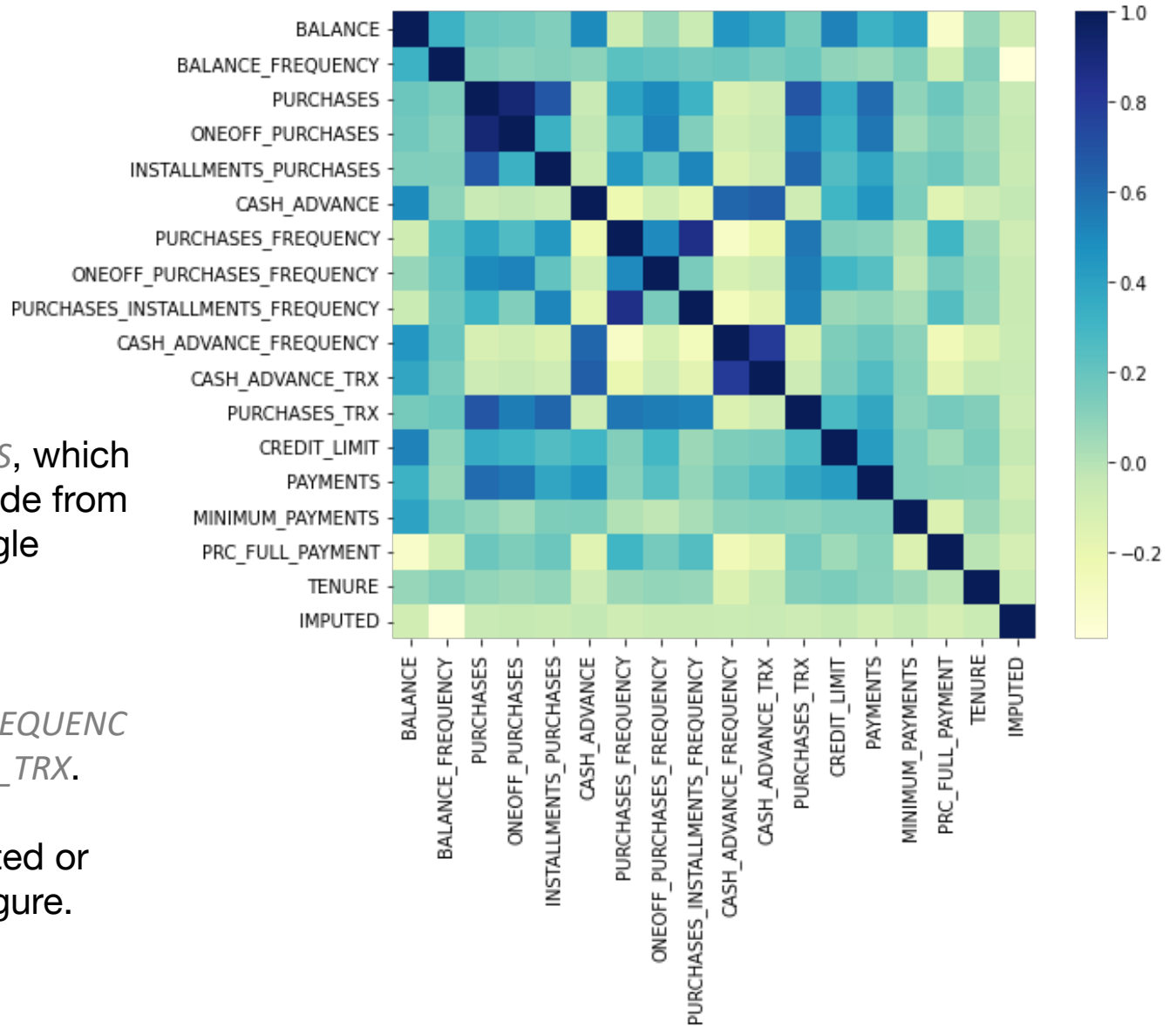
2.3 Exploratory data visualization

Distributions of attributes

- Most are heavily skewed towards one side.
- Two exceptions, *PURCHASES_INSTALLMENTS_FREQUENCY* and *PURCHASES_FREQUENCY*, they have large densities on both ends.



Correlation among attributes



- some highly correlated attributes near the diagonal.
- *PURCHASES* --ONEOFF_PURCHASES / *INSTALLMENTS_PURCHASES*, which is not surprising because the total amount of purchases made from account should be related to the maximum amount in a single purchase and the amount of purchase done in installment.
- Similar correlations can also be observed in *PURCHASES_FREQUENCY* and *PURCHASES_INSTALLMENTS_FREQUENCY* as well as *CASH_ADVANCE_FREQUENCY* and *CASH_ADVANCE_TRX*.
- Meanwhile, we can see that whether the data point is imputed or not is not correlated with other attributes as shown in the figure.

3. Modeling

3.1 Normalization

```
## X is the original data
X = data[data.columns[1:-1]]
scaler = StandardScaler()
X = scale.fit_transform(X)
X.shape
```

```
(8950, 17)
```

```
## X_imp is the original data with one extra attribute `IMPUTED`
imputed_points = data['IMPUTED'].to_numpy().reshape(X.shape[0],1)
X_imp = np.hstack((X,imputed_points))
X_imp.shape
```

```
(8950, 18)
```

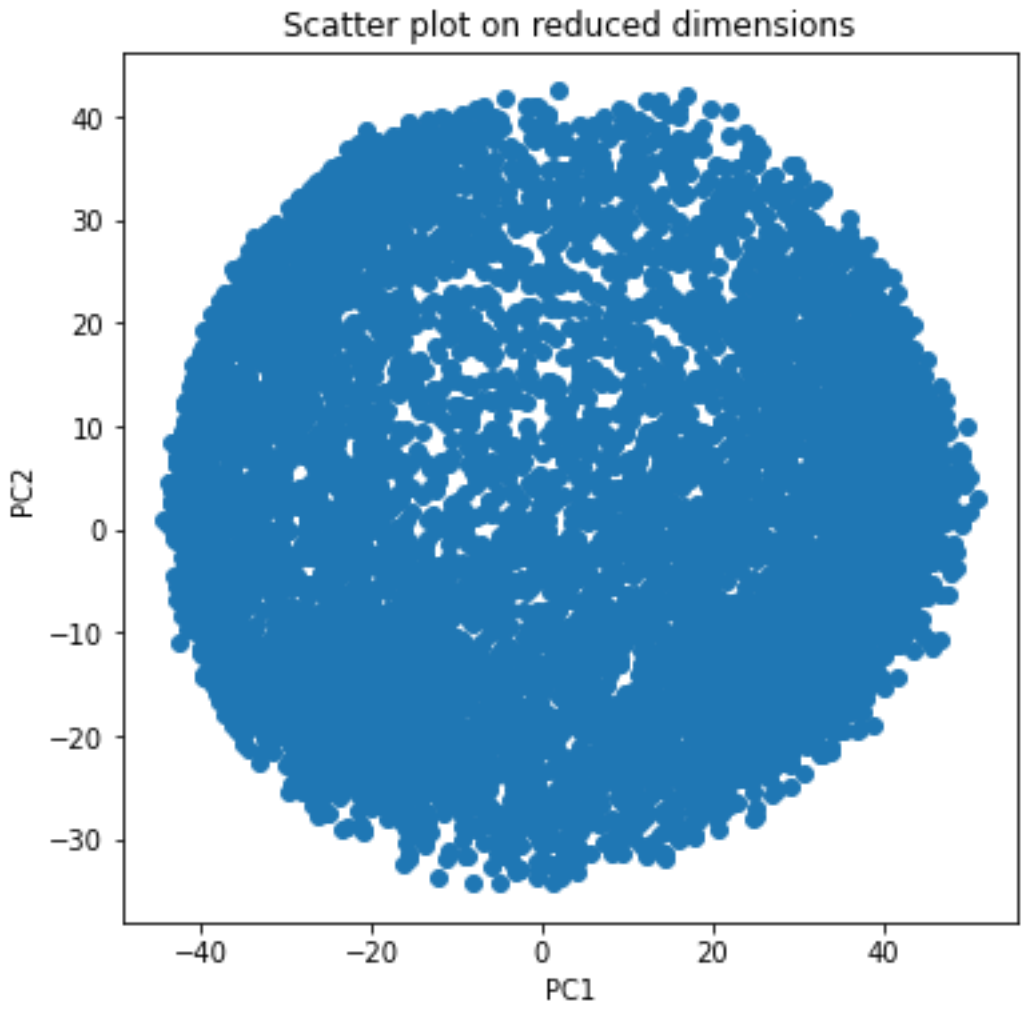
3.2 PCA on a similarity matrix for visualization

- **PCA on similarity matrix**

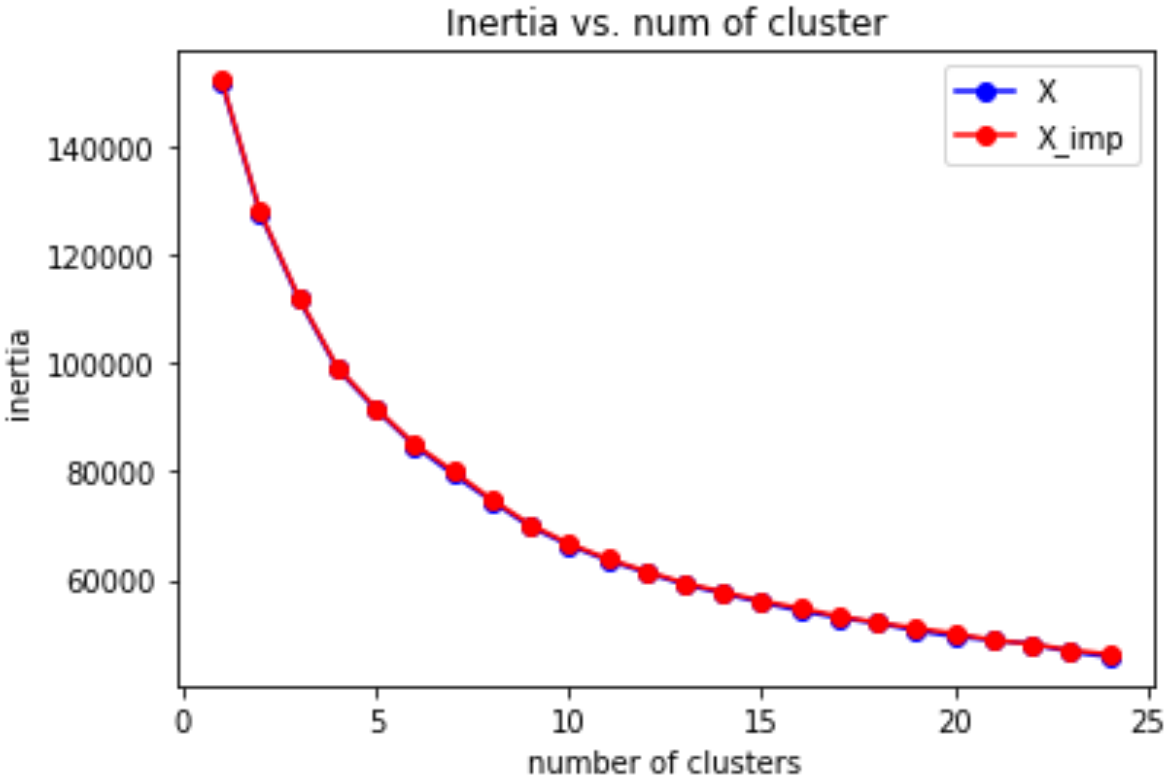
```
X_sim = cosine_similarity(X)
pca = PCA(n_components=2)
X_sim_reduced = pca.fit_transform(X_sim)
print(X_sim.shape)
print(X_sim_reduced.shape)
```

```
(8950, 8950)
(8950, 2)
```

- **Visualization on reduced dimension**



3.3 K-means: optimizing the number of clusters using inertia



- Two set of experiments are done with X and X_imp respectively to check whether the extra attribute influences the performance.
- These two curves are almost identical and the 'elbow' feature is not very typical, instead the transition is relatively smooth.
- So a range of possible numbers of clusters are proposed, including 3,4,5,6,7,8.

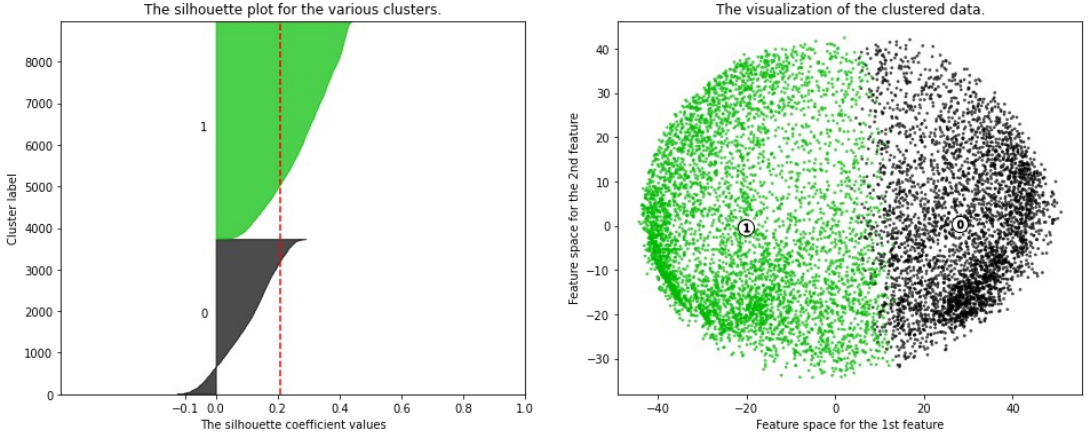
3.4 Selecting the number of clusters with silhouette analysis on K-Means clustering

- Since we can't select the most appropriate number of clusters from the elbow curve, a different method is used.

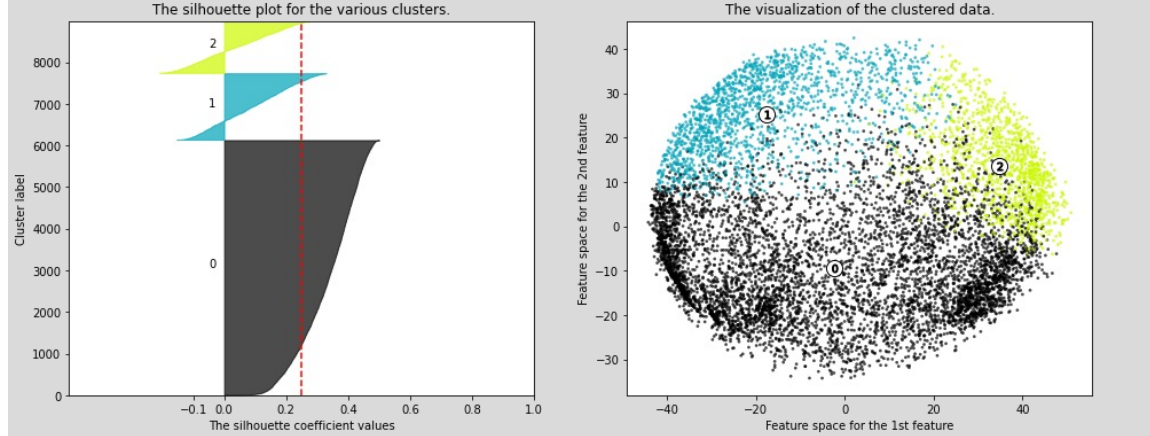
- **Silhouette analysis** is a method of interpretation and validation of consistency within clusters of data. The silhouette value is a measure of how similar an object is to its own cluster compared to other clusters. The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

https://en.wikipedia.org/wiki/Silhouette_(clustering)

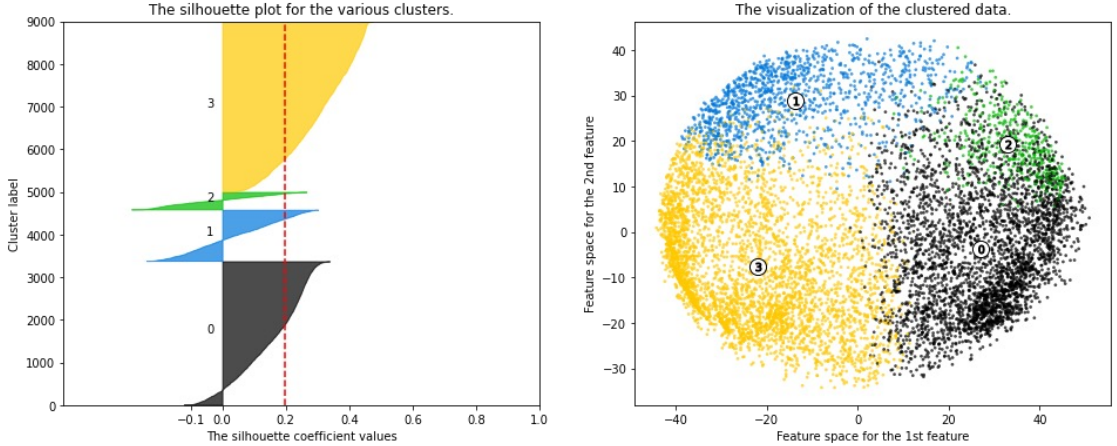
Silhouette analysis for KMeans clustering on sample data with n_clusters = 2



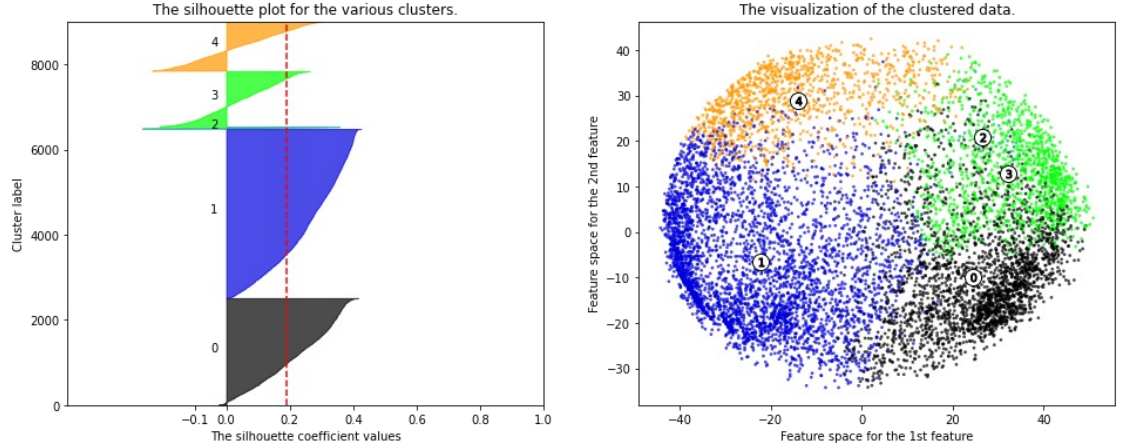
Silhouette analysis for KMeans clustering on sample data with n_clusters = 3



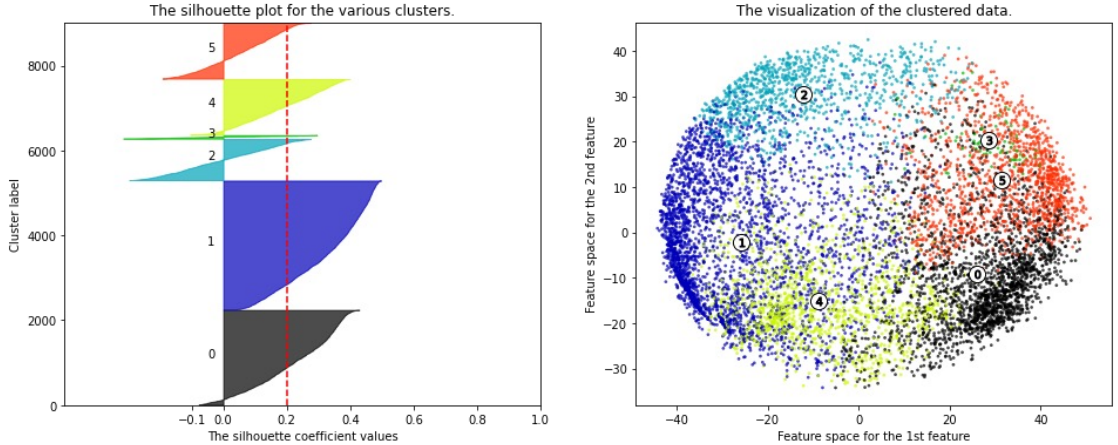
Silhouette analysis for KMeans clustering on sample data with n_clusters = 4



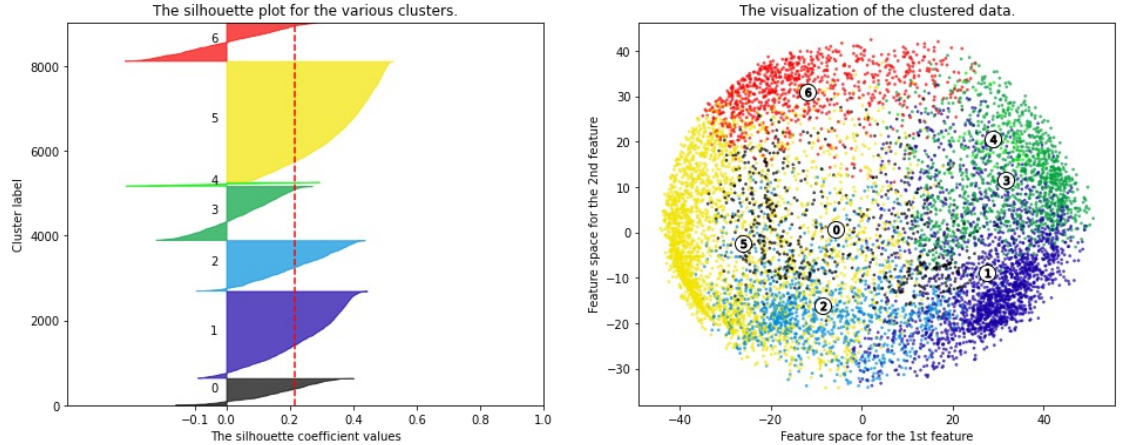
Silhouette analysis for KMeans clustering on sample data with n_clusters = 5



Silhouette analysis for KMeans clustering on sample data with n_clusters = 6



Silhouette analysis for KMeans clustering on sample data with n_clusters = 7

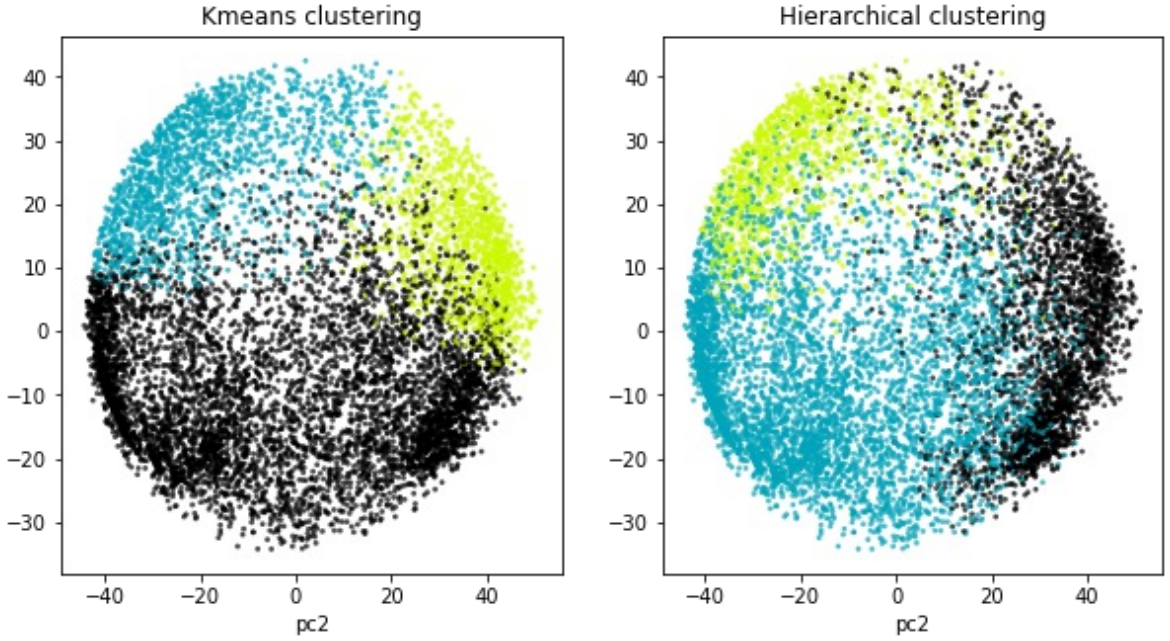


3.4 Comparison with Hierarchical clustering

```
n_clusters = 3
kmeans_labels = KMeans(n_clusters=n_clusters, random_state=0).fit_predict(X)
hier_labels = AgglomerativeClustering(n_clusters=n_clusters,linkage='ward',affinity='euclidean').fit_predict(X)
silhouette_avg = silhouette_score(X, hier_labels)
rand = rand_score(labels,kmeans_labels)

print('The average silhouette score of hierarchical clustering is {}.'.format(silhouette_avg))
print('The rand score between two methods is: {}.'.format(rand))
```

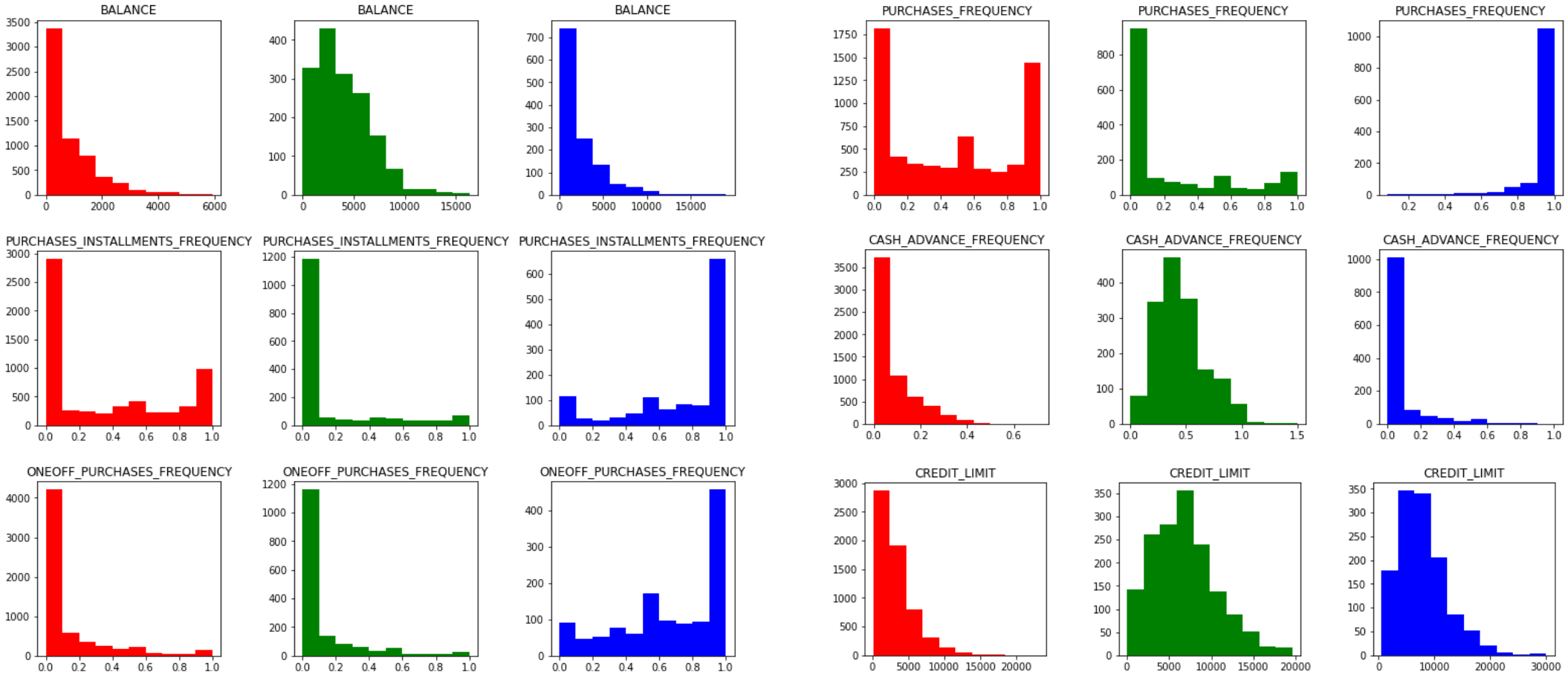
```
The average silhouette score of hierarchical clustering is 0.1674469835551831.
The rand score between two methods is: 0.6756981304986481.
```



The linkage and affinity for hierarchical clustering are *'ward*' and *'euclidean*' (other combinations also tried, this gives the best performance. The silhouette score is lower than the k-means clustering. The rand score is around 67%, suggesting that nearly 2/3 of the data are clustered into the same group in two methods, indicating the result is pretty robust.

4. Results and discussion

Distributions in different groups



In this project, I used K-means to cluster credit users to three groups. The resulting groups show distinct distributions of many attributes. This suggests that some meaningful relationships among customers are uncovered and the clustering is pretty robust.

Thank you!