

Nội dung

| | | |
|-----|---|----|
| 1. | Giới thiệu | 3 |
| 1.1 | Mục đích của bài project | 3 |
| 1.2 | Tìm hiểu tổng quan về data | 3 |
| 2. | Cơ sở lý thuyết..... | 4 |
| 2.1 | Chuẩn bị data..... | 4 |
| 2.2 | Các phương pháp chuẩn hóa data..... | 5 |
| 2.3 | Tạo các biến để phân tích dữ liệu..... | 5 |
| 2.4 | Chuyển hóa 4 nhóm khách hàng | 6 |
| 3. | Phương pháp phân tích | 7 |
| 3.1 | Ma trận correlation | 7 |
| 3.2 | Chọn số lượng component trong PCA | 8 |
| 3.3 | Mô hình phân loại k_Means..... | 9 |
| 4. | Kết luận và khuyến nghị | 10 |
| | PHỤ LỤC | 12 |

1. Giới thiệu

1.1 Mục đích của bài project

Dataset là 1 file dữ liệu về thông tin giao dịch của các tài khoản sử dụng thẻ credit, gồm có 8,950 dòng và 18 cột. Dữ liệu này lưu trữ các thông tin về lịch sử giao dịch liên quan của các khách hàng như: số dư tài khoản, số tiền mua trong tháng, số tiền ứng trước trong tháng, mua hàng trả góp hay không trả góp. Bài tiểu luận này sử dụng k-Means để phân loại thành bốn nhóm khách hàng dựa vào hành vi tiêu dùng của khách hàng, từ đó đưa ra các khuyến nghị dựa trên từng nhóm khách hàng này.

1.2 Tìm hiểu tổng quan về data

Ý nghĩa các biến trong data.


- CUST_ID: số ID của từng khách hàng.
- BALANCE: số dư tài khoản.
- BALANCE_FREQUENCY: tỷ lệ số dư tài khoản.
- PURCHASES: lượng tiền khách hàng đã mua.
- ONEOFF_PURCHASES: giao dịch thanh toán một lần.
- INSTALLMENT_PURCHASES: giao dịch thanh toán trả góp
- CASH_ADVANCE: lượng tiền rút/ứng bằng tiền mặt.
- PURCHASES_FREQUENCY: tỷ lệ mua hàng
- ONEOFF_PURCHASES_FREQUENCY: tỷ lệ giao dịch thanh toán một lần.
- PURCHASES_INSTALLMENTS_FREQUENCY: tỷ lệ giao dịch thanh toán trả góp.
- CASH_ADVANCE_FREQUENCY: tỷ lệ thực hiện rút/ứng bằng tiền mặt.
- CASH_ADVANCE_TRX: số lần thực hiện rút/ứng bằng tiền mặt.
- PURCHASES_TRX: số lần thực hiện giao dịch mua hàng.
- CREDIT_LIMIT: hạn mức.
- PAYMENTS: tổng số tiền phải thanh toán.
- MINIMUM_PAYMENTS: số tiền thanh toán tối thiểu.
- PRC_FULL_PAYMENT: Tỷ lệ thanh toán đủ trên tổng dư nợ. Khách hàng không để dư nợ.

- TENURE: kỳ hạn (ngân hàng hay dùng TENOR).

2. Cơ sở lý thuyết

2.1 Chuẩn bị data

Do data là một bộ dữ liệu gồm có 8,950 dòng và 18 cột, trong đó có tổng cộng 314 giá trị null ở 2 trường. CREDIT_LIMIT có 1 giá trị null và MINIMUM_PAYMENTS có 313 giá trị null.



| |
|----------------------------------|
| BALANCE |
| BALANCE_FREQUENCY |
| PURCHASES |
| ONEOFF_PURCHASES |
| INSTALLMENTS_PURCHASES |
| CASH_ADVANCE |
| PURCHASES_FREQUENCY |
| ONEOFF_PURCHASES_FREQUENCY |
| PURCHASES_INSTALLMENTS_FREQUENCY |
| CASH_ADVANCE_FREQUENCY |
| CASH_ADVANCE_TRX |
| PURCHASES_TRX |
| CREDIT_LIMIT |
| PAYMENTS |
| MINIMUM_PAYMENTS |
| PRC_FULL_PAYMENT |
| TENURE |

```
CREDIT_LIMIT
False      8949
True        1
Name: CREDIT_LIMIT, dtype: int64

MINIMUM_PAYMENTS
False      8637
True       313
Name: MINIMUM_PAYMENTS, dtype: int64
```

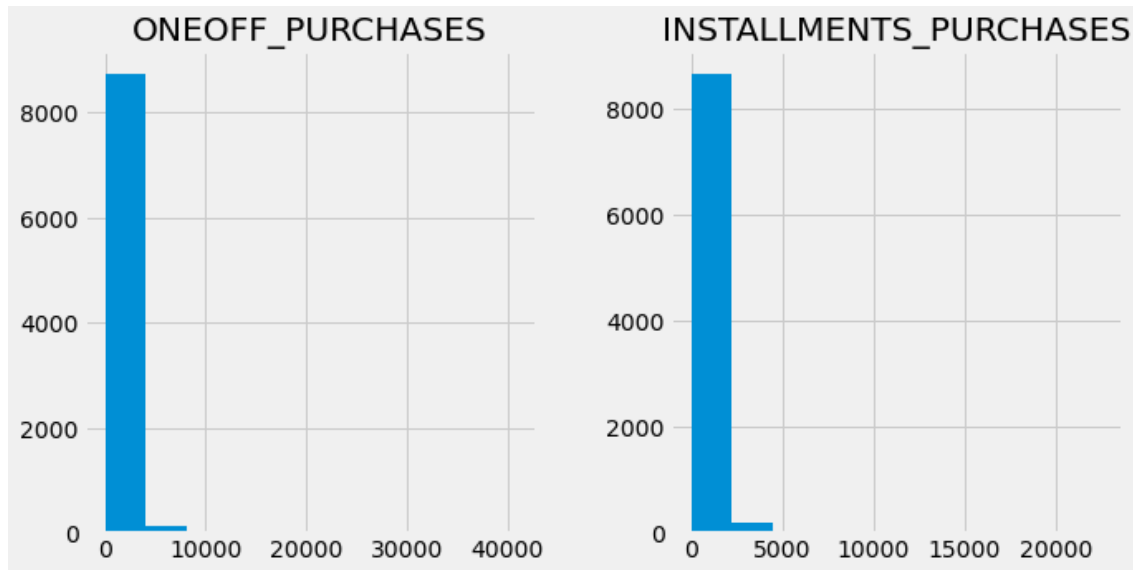
2.2 Các phương pháp chuẩn hóa data

- Xử lý giá trị null: do giá trị null ở các trường CREDIT_LIMIT và MINIMUM_PAYMENTS không thể bằng 0, vì tất cả các dòng đều có số dư tài khoản lẫn có các giao dịch. Để không tạo ra các giá trị ngoại biên ngoài mong muốn và phù hợp với data, nên phương pháp xử lý giá trị null trong bài này là dùng trung vị (median).
- Xử lý sự khoảng cách về giá trị ở các biến: do các giá trị trong data này là khác nhau và có khoảng cách rất lớn. Nếu lấy các giá trị này để phân tích thì sẽ tạo ra các giá trị ngoại biên lẫn hiểu sai về thông tin các biến. Do đó, bài này đưa các giá trị về giá trị logarit để tránh các nhược điểm vừa nêu.

2.3 Tạo các biến để phân tích dữ liệu

Từ data này, ta có thể tạo thêm các trường để hiểu rõ hơn về hành vi của khách hàng sử dụng thẻ. Cụ thể như sau:

- Giá trị mua hàng hàng tháng dựa vào kỳ hạn. Giá trị này được tính bởi: $PURCHASES / TENURE$.
- Số tiền khách hàng thực hiện rút/ứng bằng tiền mặt. Giá trị này được tính bởi: $CASH_ADVANCE / TENURE$.
- Tỷ lệ hạn mức còn lại được sử dụng. Giá trị này được tính bởi: $BALANCE / CREDIT_LIMIT$.
- Tỷ lệ thanh toán so với thanh toán tối thiểu. Giá trị này được tính bởi: $PAYMENTS / MINIMUM_PAYMENTS$.



Trong data này cần chú ý về 2 trường ONEOFF_PURCHASES và INSTALLMENT_PURCHASES, là trường thể hiện các giao dịch được khách hàng mua hàng được thanh toán 1 lần và giao dịch được khách hàng mua thông qua trả góp. Từ đây ta có thể phân loại khách hàng thành 4 nhóm như sau:

- Nhóm khách hàng thanh toán không mua hàng thông qua trả góp hay thanh toán 1 lần. Tức là khách hàng chỉ có thực hiện rút/ứng tiền mặt.
- Nhóm khách hàng vừa thực hiện giao dịch mua hàng trả góp và thanh toán 1 lần.
- Nhóm khách hàng chỉ thực hiện giao dịch mua hàng trả góp.
- Nhóm khách hàng chỉ thực hiện giao dịch mua hàng thanh toán 1 lần.

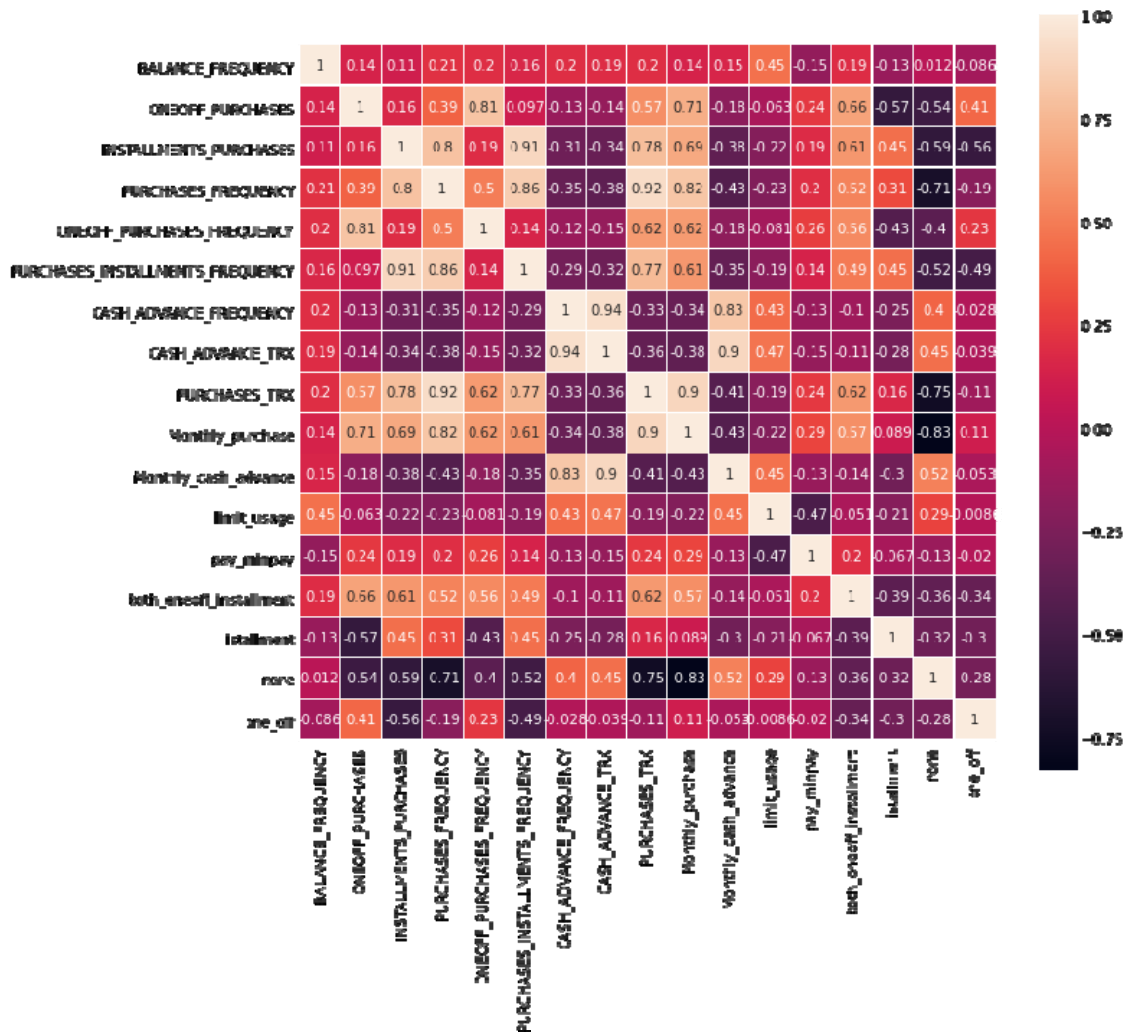
2.4 Chuyển hóa 4 nhóm khách hàng

Từ 4 nhóm khách hàng trên, để thuận tiện cho việc phân tích thì cần đưa 4 nhóm khách hàng này thành 4 trường khác nhau, là biến rời rạc và có giá trị 0 hoặc 1.

| Monthly_purchase | Monthly_cash_advance | limit_usage | pay_minpay | purchase_kind | both_oneoff_installment | installment | none | one_off |
|------------------|----------------------|-------------|------------|---------------|-------------------------|-------------|------|---------|
| 2.191654 | 0.000000 | 0.040086 | 0.894662 | installment | 0 | 1 | 0 | 0 |
| 0.000000 | 6.287695 | 0.376719 | 1.574068 | none | 0 | 0 | 1 | 0 |
| 4.180994 | 0.000000 | 0.287197 | 0.688979 | one_off | 0 | 0 | 0 | 1 |
| 4.835620 | 2.898616 | 0.200671 | 0.000000 | one_off | 0 | 0 | 0 | 1 |
| 0.847298 | 0.000000 | 0.519644 | 1.327360 | one_off | 0 | 0 | 0 | 1 |

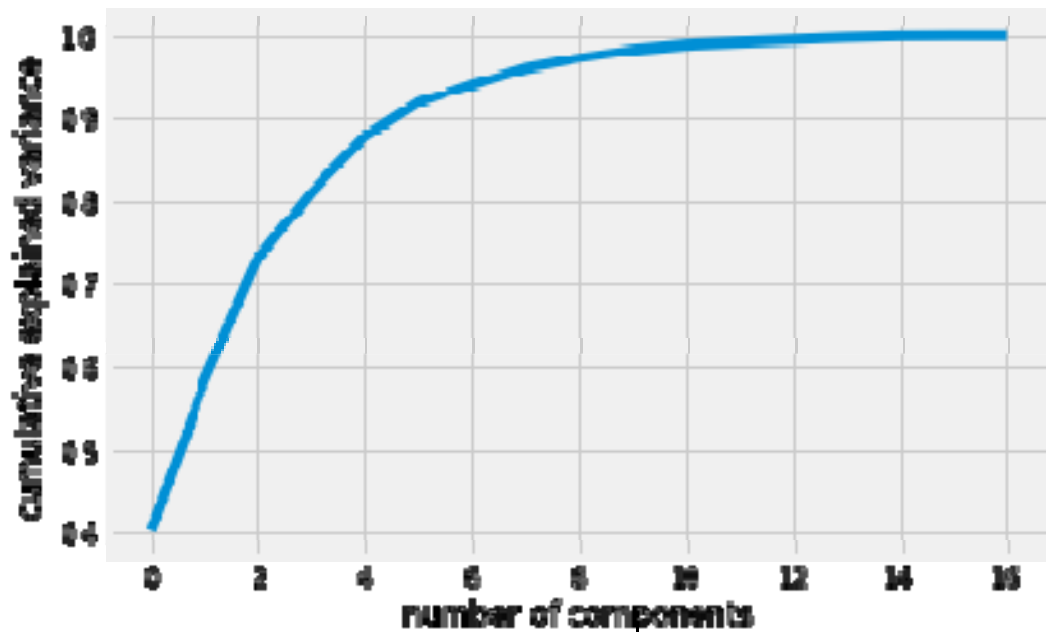
3. Phương pháp phân tích

3.1 Ma trận correlation

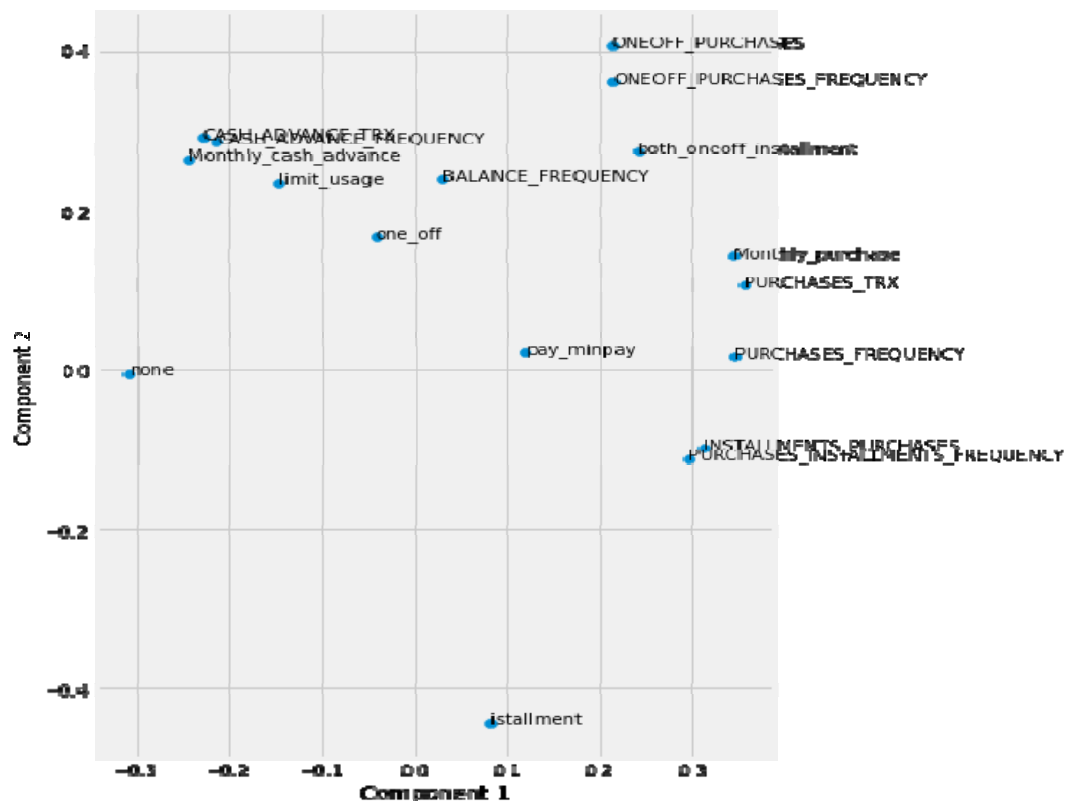


Vì tất cả biến trong data đều là giá trị liên tục, do đó ta chỉ có thể kiểm tra được độ tương quan giữa các biến với nhau. Các biến có giá trị tuyệt đối lớn hơn 0.7 thể hiện có sự tương quan mạnh với nhau. Biểu đồ trên cho thấy có rất nhiều biến có tương quan mạnh với nhau, vì vậy trong bài này tôi sử dụng phương pháp phân tích PCA để đơn giản hóa bộ dữ liệu mà không bị mất thông tin gốc từ data.

3.2 Chọn số lượng component trong PCA

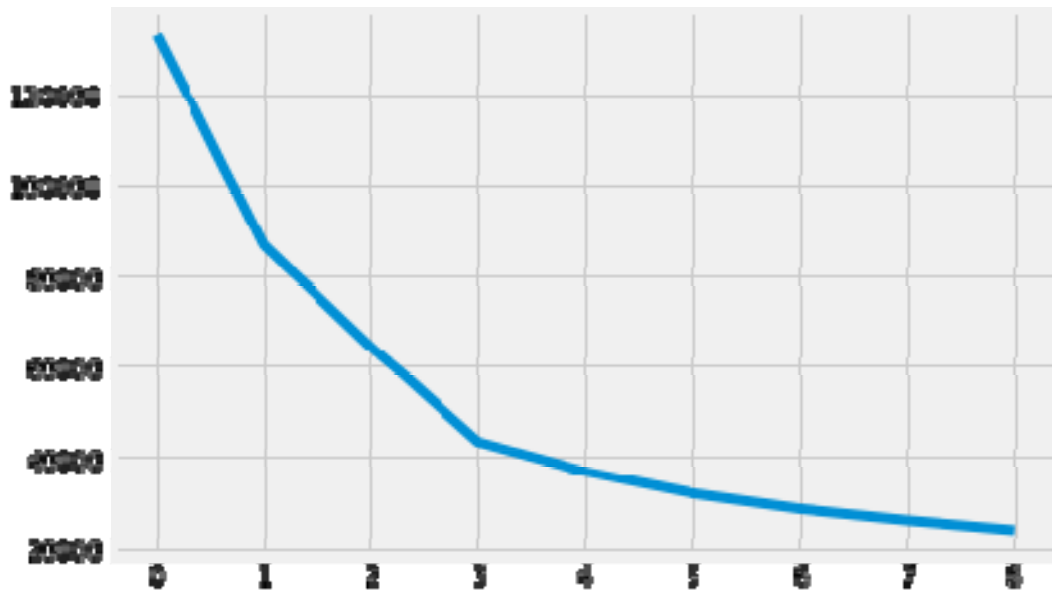


Từ biểu đồ trên, ta có thể thấy ở giá trị component là 5 giải thích được 92% giá trị phương sai. Ta chọn giá trị 5 vì độ dốc từ sau giá trị 5 không thay đổi nhiều. Ta có thể nhìn thấy các trường trong data được hiển thị qua biểu đồ sau:

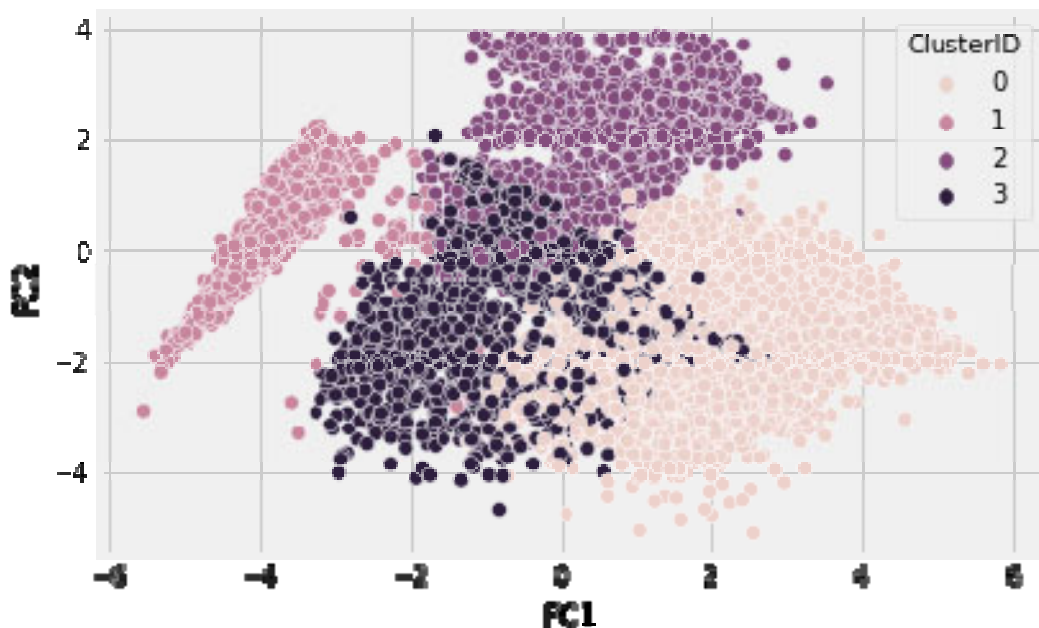


3.3 Mô hình phân loại k_Means

Đầu tiên, sử dụng phương pháp StandardScaler trong thư viện sklearn để chuẩn hóa data.



Trong biểu đồ này, ta có thể thấy giá trị $k=4$ (theo thứ tự 0,1,2,3) là đạt đỉnh độ dốc xuống. Từ sau giá trị $k=4$ đó, độ dốc ít dần đều và cũng cho tương ứng với số nhóm khách hàng cần phân loại trong bài này. Đồ thị scatter plot sau thể hiện sự phân cụm nhóm khách hàng thành 4 nhóm.



4. Kết luận và khuyến nghị

| ClusterID | 0 | 1 | 2 | 3 |
|-------------------------|---------|---------|---------|---------|
| BALANCE | 1811.24 | 2177.57 | 798.27 | 1429.02 |
| PURCHASES | 2280.61 | 1.86 | 543.58 | 787.35 |
| PURCHASES FREQUENCY | 0.80 | 0.00 | 0.70 | 0.32 |
| PURCHASES TRX | 33.14 | 0.05 | 12.05 | 7.12 |
| CASH ADVANCE TRX | 2.81 | 6.55 | 1.02 | 2.86 |
| CREDIT LIMIT | 5748.84 | 4055.58 | 3338.24 | 4512.91 |
| Monthly_purchase | 193.76 | 0.16 | 47.56 | 69.76 |
| Monthly_cash_advance | 67.64 | 186.30 | 33.47 | 77.84 |
| limit_usage | 0.35 | 0.58 | 0.26 | 0.38 |
| pay_minpay | 7.26 | 9.93 | 13.40 | 5.56 |
| both_oneoff_installment | 1.00 | 0.00 | 0.00 | 0.00 |
| installment | 0.00 | 0.02 | 1.00 | 0.00 |
| none | 0.00 | 0.98 | 0.00 | 0.00 |
| one_off | 0.00 | 0.00 | 0.00 | 1.00 |

Với nhóm phân cụm khách hàng 0, đây là nhóm khách hàng đồng thời thực hiện giao dịch mua hàng trả góp lẫn mua hàng thanh toán 1 lần, và cũng là nhóm có tần suất mua sản phẩm cao nhất trong 4 khách hàng. Điều đó chứng minh tổng mua hàng, sức mua hàng tháng của nhóm này cũng cao nhất. Tuy nhiên, nhóm khách hàng này chỉ mới sử dụng 35% hạn mức. Điều đó cho thấy khách hàng sử dụng tiền mặt nhiều hơn là phương thức ứng tiền trước, thực tế trong bài cho thấy số tiền mua hàng tháng của khách hàng là 193.76 trong khi giao dịch ứng/rút tiền của khách hàng chỉ có 67.64. Vì vậy đứng trên góc độ ngân hàng, cần khuyến khích nhóm khách hàng này sử dụng dịch vụ của thẻ credit nhiều hơn vì các lợi ích ngân hàng có thể mang lại ưu đãi nhiều hơn, qua đó ngân hàng có thể thu phí từ các đối tác liên kết nhiều hơn trong khi nhóm khách hàng này vẫn có quyền lợi thông qua tích điểm đổi quà.

Với nhóm phân cụm khách hàng 1, đây là nhóm có tần suất mua sắm kém nhất nhưng lại chọn phương án ứng trước nhiều nhất đồng thời lại có hạn mức sử dụng cao nhất trong nhóm 4 khách hàng. Điều này có nghĩa nhóm khách hàng này không có nhu cầu mua sắm nhưng khi mua sắm lại thích sử dụng phương thức được ứng trước. Do đó giải pháp kích thích nhóm khách hàng này là cần tìm hiểu nhu cầu mua sắm của họ để có thể cung ứng đúng sản phẩm dịch vụ nhằm tăng tần suất mua sắm nhiều hơn.

Với nhóm phân cụm khách hàng 2, đây là nhóm chỉ thực hiện mua sắm trả góp đồng thời cũng là nhóm có tỷ lệ thanh toán tốt nhất trong 4 nhóm. Do đó đây là nhóm khách hàng tiềm năng đang cần được tăng hạn mức tín dụng hoặc cung cấp các dịch vụ sử dụng thẻ tốt hơn.

Nhóm phân cụm khách hàng 3 là nhóm có hoạt động giao dịch tốt nhất, có độ rủi ro thấp nhất trong 4 nhóm khách hàng.

PHỤ LỤC

- Cài đặt thư viện và style vẽ đồ thị

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.decomposition import IncrementalPCA
from sklearn.metrics import silhouette_score
plt.style.use('fivethirtyeight')
```

- Đọc data

```
data = pd.read_csv('Credit Card.csv')
df= data.drop('CUST_ID', axis=1)
print(data.shape)
data.head(10)
```

- Kiểm tra giá trị null.

```
missing_data = df.isnull()
sns.heatmap(missing_data, cbar=False, yticklabels=False, cmap='viridis')
```

- Đếm giá trị null của các trường

```
missing_data = df.isnull()
for column in missing_data.columns.values.tolist():
    print(column)
    print (missing_data[column].value_counts())
    print("")
```

- Xử lý giá trị null bằng median

```
df['CREDIT_LIMIT'].fillna(df['CREDIT_LIMIT'].median(),inplace=True)
df['MINIMUM_PAYMENTS'].fillna(df['MINIMUM_PAYMENTS'].median(),inplace=True)
df.isnull().sum()
```

- Tạo các biến để phân tích dữ liệu (Mục 2.3)

```
df['Monthly_purchase']=df['PURCHASES']/df['TENURE']
df['Monthly_cash_advance']=df['CASH_ADVANCE']/df['TENURE']
```

```
df['limit_usage']=df.apply(lambda x: x['BALANCE']/x['CREDIT_LIMIT'], axis=1)
```

```
df['pay_minpay']=df.apply(lambda  
x:x['PAYMENTS']/x['MINIMUM_PAYMENTS'],axis=1)
```

- Tìm hiểu để phân nhóm khách hàng

```
df[['ONEOFF_PURCHASES','INSTALLMENTS_PURCHASES']].hist(bins = 10,  
figsize=(10,5))
```

```
plt.show()
```

```
df.loc[:,['ONEOFF_PURCHASES','INSTALLMENTS_PURCHASES']].tail(10)
```

- Phân nhóm khách hàng

```
def purchase(df):
```

```
    if (df['ONEOFF_PURCHASES']==0) & (df['INSTALLMENTS_PURCHASES']==0):
```

```
        return 'none'
```

```
    if (df['ONEOFF_PURCHASES']>0) & (df['INSTALLMENTS_PURCHASES']>0):
```

```
        return 'both_oneoff_installment'
```

```
    if (df['ONEOFF_PURCHASES']>0) & (df['INSTALLMENTS_PURCHASES']==0):
```

```
        return 'one_off'
```

```
    if (df['ONEOFF_PURCHASES']==0) & (df['INSTALLMENTS_PURCHASES']>0):
```

```
        return 'installment'
```

```
df['purchase_kind']=df.apply(purchase,axis=1)
```

```
df.head()
```

- Đưa giá trị về giá trị logarit

```
df_log=df.drop(['purchase_kind'],axis=1).applymap(lambda x: np.log(x+1))
```

```
df_log.describe()
```

```
col=['BALANCE','PURCHASES','CASH_ADVANCE','TENURE','PAYMENTS','MINI  
MUM_PAYMENTS','PRC_FULL_PAYMENT','CREDIT_LIMIT']
```

```
df_log=df_log[[x for x in df_log.columns if x not in col ]]
```

```
df_log.head()
```

- Đưa các giá trị ở cột purchase_kind về giá trị rời rạc

```
df_log['purchase_kind']=df_log[:, 'purchase_kind']
```

```
df_dummy=pd.concat([df_log,pd.get_dummies(df_log['purchase_kind'])],axis=1)
```

```
df_dummy.head()
```

- Vẽ biểu đồ correlation matrix

```
%matplotlib inline
```

```
fig, ax = plt.subplots(figsize=(10,10))
```

```
ax = sns.heatmap(df_dummy.corr(), linewidths=0.1, square=True, annot=True)
```

- Scale dữ liệu theo phương pháp Standard Scaler

```
for i in num_cols:
```

```
    # fit on training data column
```

```
    scale = Scaler().fit(X[[i]])
```

```
    X[i] = scale.transform(X[[i]])
```

```
X.head()
```

- Sử dụng PCA model

```
pca = PCA(svd_solver='randomized', random_state=42)
```

```
pca.fit(X)
```

```
pca.components_
```

- Tìm số lượng components bằng biểu đồ Line

```
%matplotlib inline
```

```
fig = plt.figure()
```

```
plt.plot(np.cumsum(pca.explained_variance_ratio_))
```

```
plt.xlabel('number of components')
```

```
plt.ylabel('cumulative explained variance')
```

```
plt.show()
```

- Lưu các giá trị components vào dataframe

```
pca_df = pd.DataFrame({ 'Variable':num_cols,'PC1':pca.components_[0],  
                        'PC2':pca.components_[1],'PC3':pca.components_[2],  
                        'PC4':pca.components_[3],'PC5':pca.components_[4]})
```

```
pca_df
```

- Thể hiện các giá trị component từng biến

```
%matplotlib inline
```

```
fig = plt.figure(figsize = (8,8))
```

```
plt.scatter(pca_df.PC1, pca_df.PC2)
```

```
plt.xlabel('Component 1')
```

```
plt.ylabel('Component 2')
```

```
for i, txt in enumerate(pca_df.Variable):
```

```
    plt.annotate(txt, (pca_df.PC1[i],pca_df.PC2[i]))
```

```
plt.tight_layout()
```

```
plt.show()
```

- Sử dụng PCA với số component là 5

```
pca_model = IncrementalPCA(n_components=5)
```

```
X_pca = pca_model.fit_transform(X)
```

- Pivot cột thành hàng

```
X_pca = np.transpose(X_pca)
```

```
pca_df1 =  
pd.DataFrame({'PC1':X_pca[0],'PC2':X_pca[1],'PC3':X_pca[2],'PC4':X_pca[3],'PC5':X_pca[4]})
```

```
print(pca_df1.shape)
```

```
pca_df1.head()
```

- Tìm số phân cụm trong model k-Means thông qua line chart

```
ssd = []
```

```
for num_clusters in list(range(1,10)):
```

```
    model_clus = KMeans(n_clusters = num_clusters, max_iter=50)
```

```
    model_clus.fit(pca_df1)
```

```
    ssd.append(model_clus.inertia_)
```

```
plt.plot(ssd)
```

- Sử dụng số phân cụm là 4 trong k-Means

```
df_cluster = KMeans(n_clusters = 4, max_iter=50,random_state = 50)
```

```
df_cluster.fit(pca_df1)
```

```
df1 = pca_df1
```

```
df1.index = pd.RangeIndex(len(df1.index))
```

```
df1_k_means = pd.concat([df1, pd.Series(df_cluster.labels_)], axis=1)
```

```
df1_k_means.columns = ['PC1', 'PC2','PC3','PC4','PC5','ClusterID']
```

```
df1_k_means
```

- Biểu diễn phân cụm bằng Scatter plot

```
sns.scatterplot(x='PC1',y='PC2',hue='ClusterID',legend='full',data=df1_k_means)
```

- Merge giá trị ở data 2 model PCA và phân cụm lại. Thêm giá trị cột ID khách hàng và giá trị phân loại nhóm

```
data_full_variable = pd.concat([df,pd.get_dummies(df['purchase_kind']),axis=1)
```

```
data_full_variable['CUST_ID'] = data['CUST_ID']
```

```
big_data=pd.merge(data_full_variable, df2, on='CUST_ID', how='inner')
big_data=big_data.drop(['PC1','PC2','PC3','PC4','PC5'],axis=1)

cols      =      ['BALANCE',      'PURCHASES',      'PURCHASES_FREQUENCY',
'PURCHASES_TRX',
      'CASH_ADVANCE_TRX',      'CREDIT_LIMIT',      'Monthly_purchase',
'Monthly_cash_advance',
      'limit_usage', 'pay_minpay', 'both_oneoff_installment', 'istallment', 'none', 'one_off',]
df_final=[]
for i in cols:
    cluster = big_data.groupby(["ClusterID"])[i].mean()
    df_final.append(cluster)
df_final = pd.concat(df_final, axis=1)
df_final


- Lưu dataframe phân cụm khách hàng


df_final.to_csv('kmeans.csv',index=True)
```