```
import pandas as pd
df=pd.read_csv("CC GENERAL.csv")
df.head()
```

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENT
0	C10001	40.900749	0.818182	95.40	0.00	
1	C10002	3202.467416	0.909091	0.00	0.00	
2	C10003	2495.148862	1.000000	773.17	773.17	
3	C10004	1666.670542	0.636364	1499.00	1499.00	
4	C10005	817.714335	1.000000	16.00	16.00	

```
df.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
```

```
df['MINIMUM_PAYMENTS'].value_counts()
```

```
299.351881     2
3.197940       1
111.691332     1
129.682608     1
872.760983     1
              ..
1227.773229    1
127.210691     1
6422.472544    1
471.940554     1
189.459157     1
Name: MINIMUM_PAYMENTS, Length: 8636, dtype: int64
```

```
df['MINIMUM_PAYMENTS'].fillna(df['MINIMUM_PAYMENTS'].mean(),inplace=True)
df['MINIMUM_PAYMENTS'].value_counts()
```

```
      864.206542     313
      299.351881       2
      140.596138       1
      1078.106633      1
      111.691332       1
                    ...
      825.485459       1
      1227.773229      1
      127.210691       1
      6422.472544      1
      189.459157       1
      Name: MINIMUM_PAYMENTS, Length: 8637, dtype: int64
```

```
df.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                    0
      PRC_FULL_PAYMENT                    0
      TENURE                              0
      dtype: int64
```

```
df['CREDIT_LIMIT'].value_counts()
```

```
      3000.0    784
      1500.0    722
      1200.0    621
      1000.0    614
      2500.0    612
               ...
      50.0        1
      9700.0      1
      6850.0      1
      5450.0      1
      3650.0      1
      Name: CREDIT_LIMIT, Length: 205, dtype: int64
```

```
df['CREDIT_LIMIT'].fillna(df['CREDIT_LIMIT'].mean(),inplace=True)
df.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                        0
PAYMENTS                            0
MINIMUM_PAYMENTS                    0
PRC_FULL_PAYMENT                    0
TENURE                              0
dtype: int64
```

```
del df['CUST_ID']
df.head()
```

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHAS
0	40.900749	0.818182	95.40	0.00	9
1	3202.467416	0.909091	0.00	0.00	
2	2495.148862	1.000000	773.17	773.17	
3	1666.670542	0.636364	1499.00	1499.00	
4	817.714335	1.000000	16.00	16.00	

```
from sklearn.cluster import AgglomerativeClustering
model=AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='complete')
clust_labels=model.fit_predict(df)
agglomerative=pd.DataFrame(clust_labels)
agglomerative
```

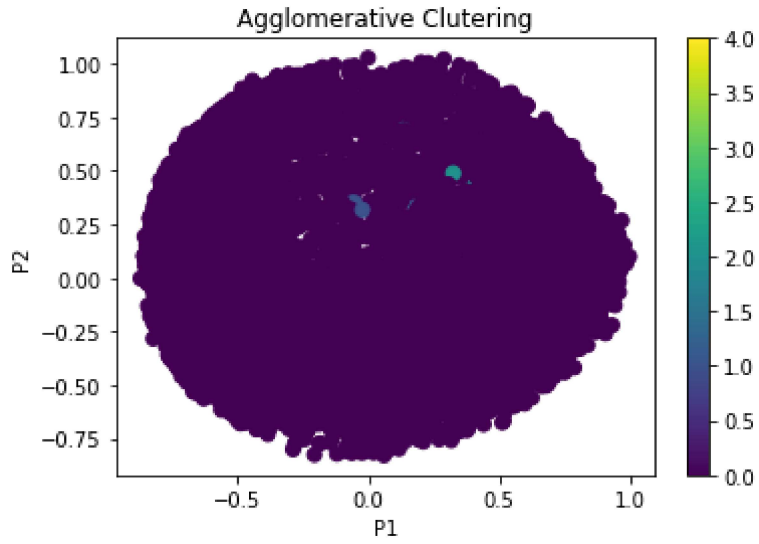
	0
0	0

```
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import StandardScaler, normalize
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
import scipy.cluster.hierarchy as shc
scaler = StandardScaler()
scaled_df = scaler.fit_transform(df)
normalized_df = normalize(scaled_df)
normalized_df = pd.DataFrame(normalized_df)
pca = PCA(n_components = 2)
X_principal = pca.fit_transform(normalized_df)
X_principal = pd.DataFrame(X_principal)
X_principal.columns = ['P1', 'P2']
X_principal.head(2)
```

	P1	P2
0	-0.489825	-0.679679
1	-0.518791	0.545010

```
import matplotlib.pyplot as plt
fig =plt.figure()
ax = fig.add_subplot(111)
scatter = ax.scatter(X_principal['P1'],X_principal['P2'], c= agglomerative[0], s=50)
ax.set_title("Agglomerative Clutering")
ax.set_xlabel("P1")
ax.set_ylabel("P2")
plt.colorbar(scatter)
```

```
<matplotlib.colorbar.Colorbar at 0x7f23c9362f10>
```




```
import matplotlib.pyplot as plt
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10,7))
```

```
plt.title("Dendrograms")
dend=shc.dendrogram(shc.linkage(X_principal, method="complete"))
```

```
Error in callback <function flush_figures at 0x7f23fafaa440> (for post_execute):
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/ipykernel/pylab/backend_inline.py in
flush_figures()
    115         # ignore the tracking, just draw and close all figures
    116         try:
--> 117             return show(True)
    118         except Exception as e:
    119             # safely show traceback if in IPython, else raise

                                21 frames
<decorator-gen-2> in __call__(self, obj)

/usr/local/lib/python3.7/dist-packages/matplotlib/font_manager.py in get_font(filena
hinting_factor)
   1326     if hinting_factor is None:
   1327         hinting_factor = rcParams['text.hinting_factor']
-> 1328     return _get_font(os.fspath(filename), hinting_factor,
   1329                      _kerning_factor=rcParams['text.kerning_factor'])
   1330

KeyboardInterrupt:
```

```
from sklearn.cluster import KMeans  #Importing our clustering algorithm: KMeans
kmeans=KMeans(n_clusters=5, random_state=0)  #Cluster our data by choosing 5 as number of
kmeans.fit(df)
labels=pd.DataFrame(kmeans.labels_)
labels
```

	0
0	0
1	4
2	4
3	4
4	0
...	...
8945	0
8946	0
8947	0
8948	0
8949	0

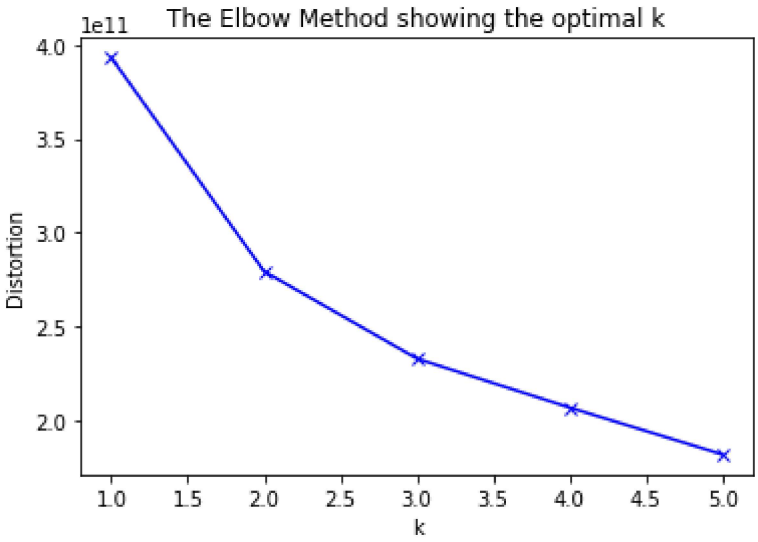
8950 rows × 1 columns

```
kmeans.predict(df)
```

```
print(kmeans.cluster_centers_)
```

```
[[7.87675442e+02 8.49448288e-01 5.09775413e+02 2.49350298e+02
  2.60721138e+02 4.83221366e+02 4.51368043e-01 1.30908566e-01
  3.46901779e-01 1.09266117e-01 2.28184627e+00 9.50227149e+00
  2.22908081e+03 9.03805320e+02 5.26784706e+02 1.48572171e-01
  1.13796111e+01]
 [5.48720526e+03 9.54910300e-01 2.40789541e+03 1.55343184e+03
  8.54576730e+02 4.13952954e+03 5.31917954e-01 3.25465421e-01
  4.01272510e-01 3.11313742e-01 9.66793893e+00 3.05114504e+01
  1.13739880e+04 5.66199017e+03 1.97499024e+03 1.07193598e-01
  1.17302799e+01]
 [4.05814769e+03 9.88636375e-01 1.02737875e+03 1.18389464e+02
  9.08989286e+02 9.22757849e+02 4.71320321e-01 3.92315536e-02
  4.41152625e-01 1.05654714e-01 3.01785714e+00 1.86250000e+01
  4.26785714e+03 1.62493914e+03 2.27600316e+04 1.48808929e-03
  1.19107143e+01]
 [5.14958549e+03 9.04434585e-01 1.95473678e+04 1.43043888e+04
  5.24297902e+03 4.86981405e+03 8.28861756e-01 6.90243927e-01
  6.98373976e-01 1.40243878e-01 7.80487805e+00 1.10658537e+02
  1.48926829e+04 2.62560508e+04 2.87567961e+03 4.76472902e-01
  1.19512195e+01]
 [1.91737096e+03 9.10317516e-01 1.33455853e+03 8.25285725e+02
  5.09650605e+02 1.01276218e+03 5.56277022e-01 3.14077082e-01
  3.83765853e-01 1.37242546e-01 3.28861154e+00 1.94227769e+01
  7.08624663e+03 1.91894612e+03 7.37498189e+02 1.77176504e-01
  1.17320593e+01]]
```

```
from sklearn.cluster import KMeans
import sklearn.cluster as cluster
import time
import matplotlib.pyplot as plt
distortions = []
K = range(1,6)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(df)
    distortions.append(kmeanModel.inertia_)
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```



0 s terminée à 14:50