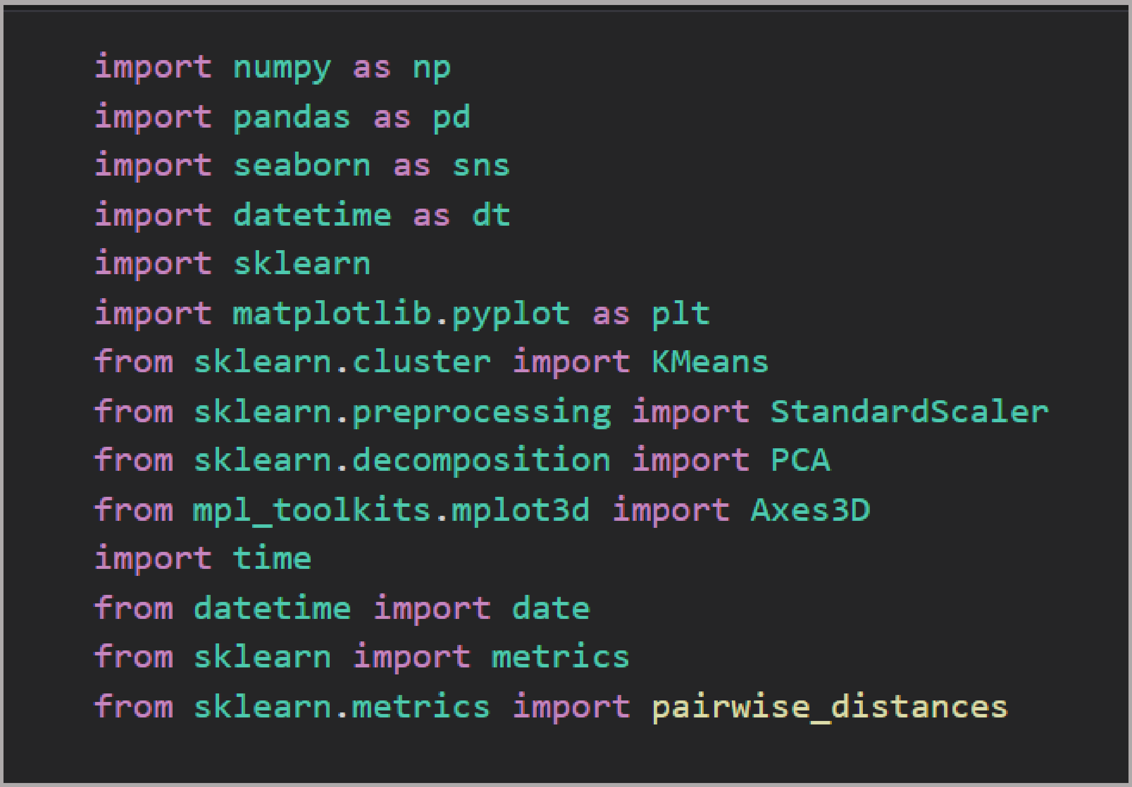
Computational Intelligence

Red Wine-Quality

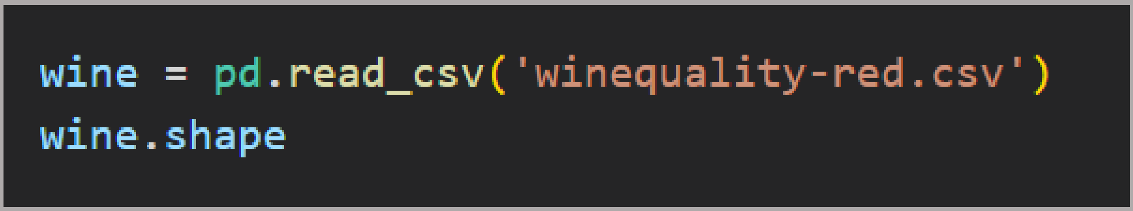
CHRISTIAN J H PAKPAHAN
5025201153

Source Code



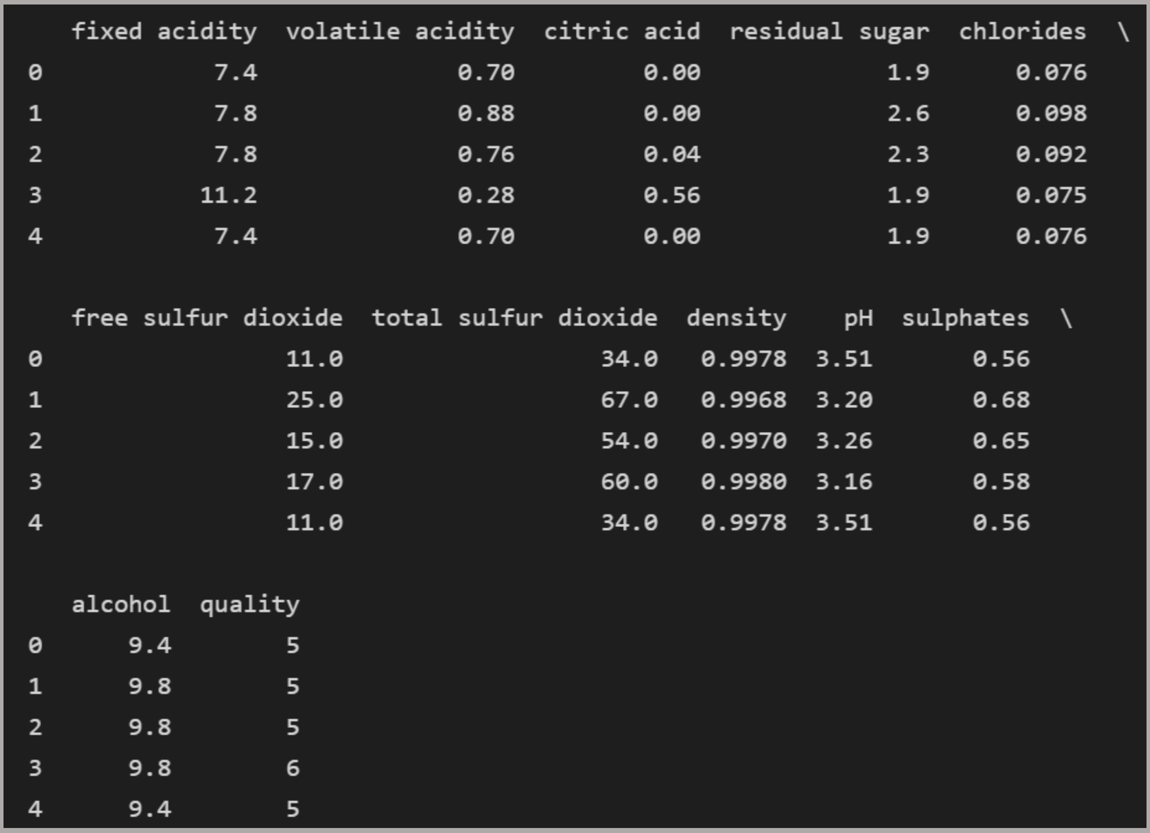
```
import numpy as np
import pandas as pd
import seaborn as sns
import datetime as dt
import sklearn
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from mpl_toolkits.mplot3d import Axes3D
import time
from datetime import date
from sklearn import metrics
from sklearn.metrics import pairwise_distances
```

Step 1 add the library and import the file (csv) into the code. After rhat, use function :



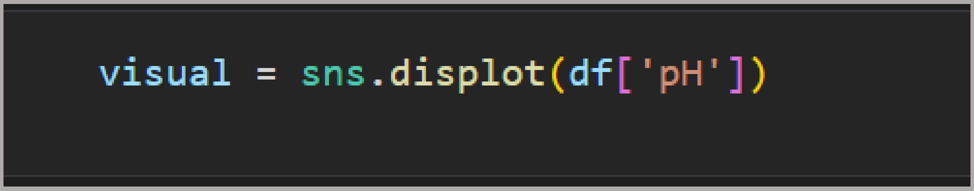
```
wine = pd.read_csv('winequality-red.csv')
wine.shape
```

And print the data with function 'print'. It will show the data that we imported from file csv

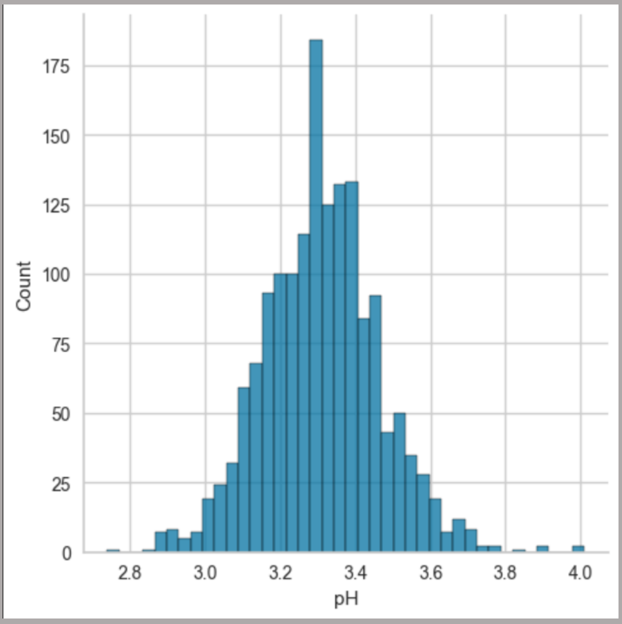


```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

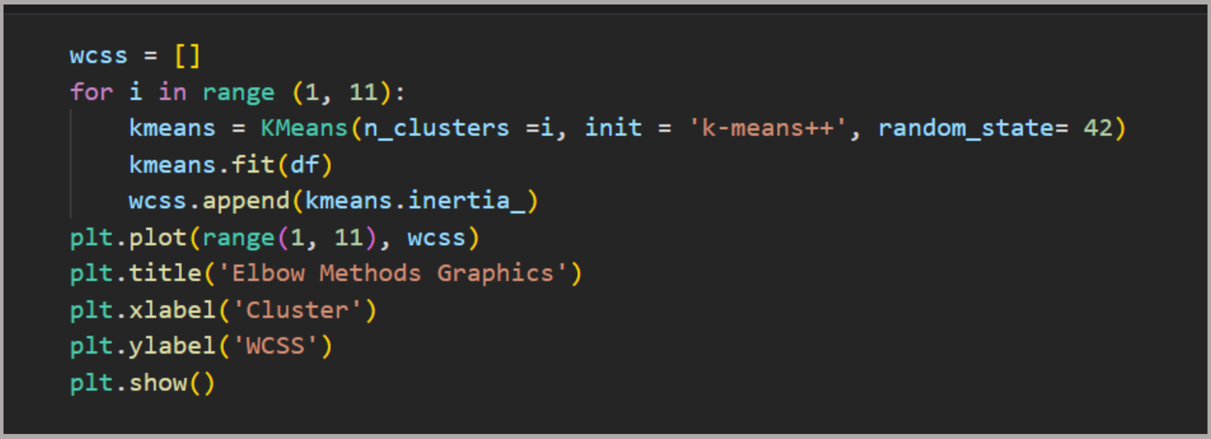
Now , in this code we make chart pH and will be show the chart below.



```
visual = sns.displot(df['pH'])
```

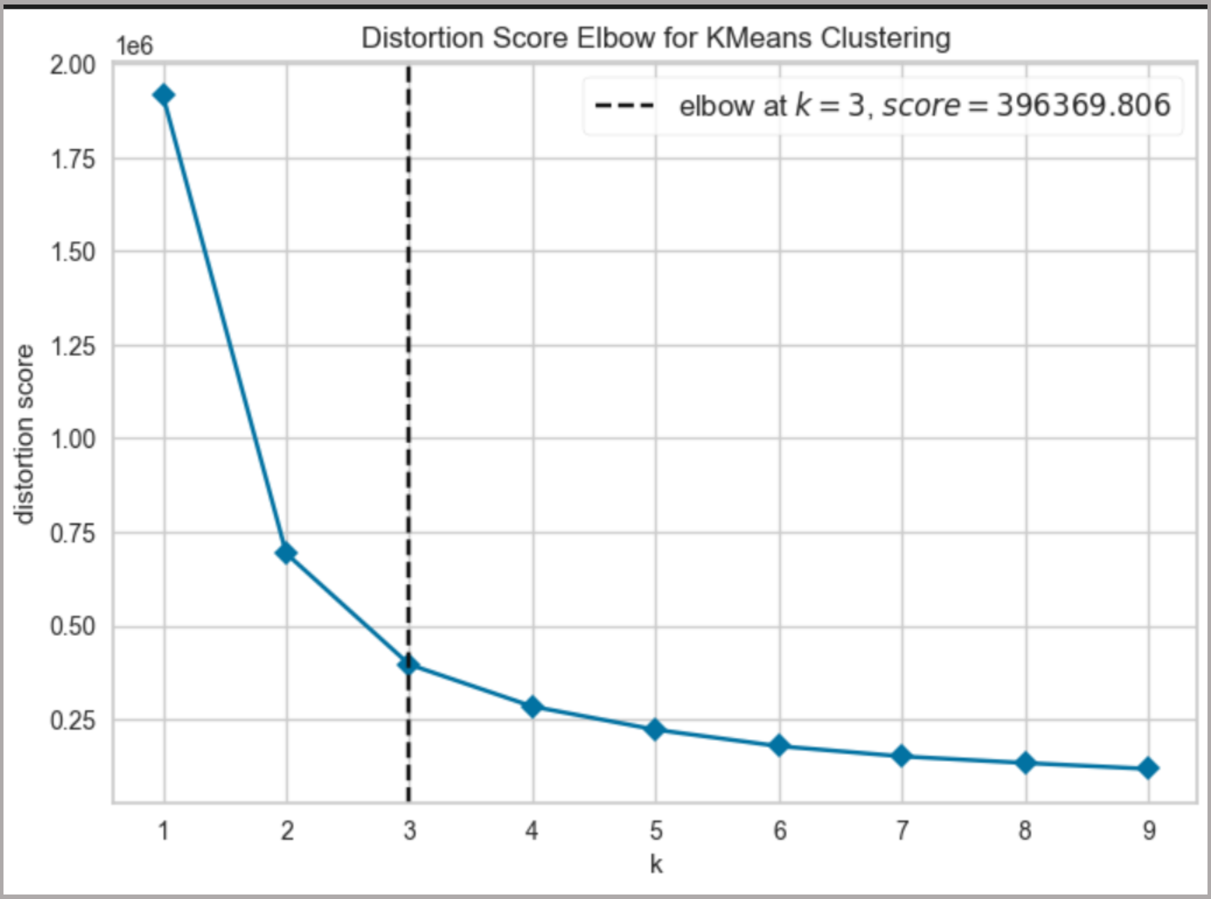


Make the Elbow methods and have inertia value with 'wcss.append(kmeans.inertia_)'



```
wcss = []
for i in range (1, 11):
    kmeans = KMeans(n_clusters =i, init = 'k-means++', random_state= 42)
    kmeans.fit(df)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('Elbow Methods Graphics')
plt.xlabel('Cluster')
plt.ylabel('WCSS')
plt.show()
```

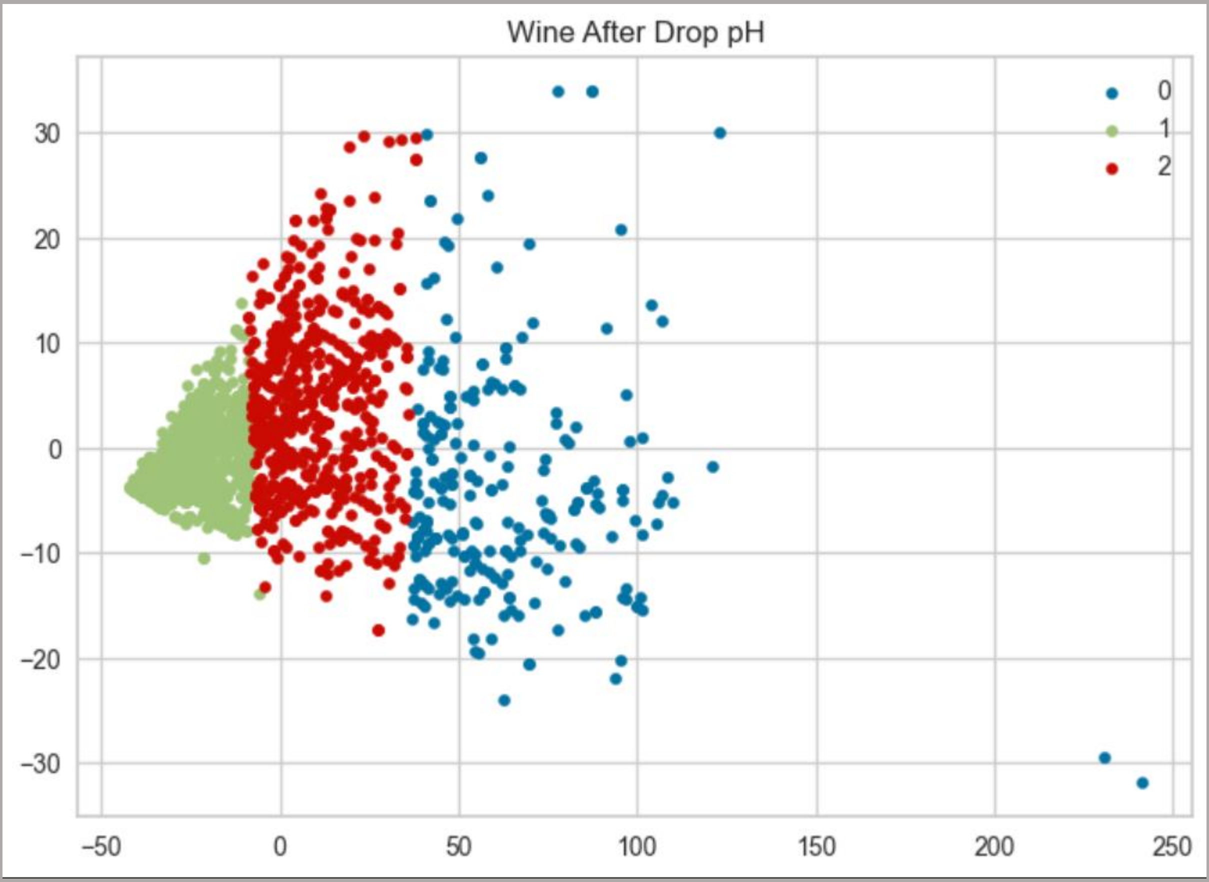
With the result :



Take the Graphic to clustering dot value in label with size 20

```
for i in graphics:
    plt.scatter(X[label==i,0], X[label==i,1], label=i, s=20)

plt.legend()
plt.title('Wine After Drop pH')
plt.show()
```

The result



And make this code to normalize data

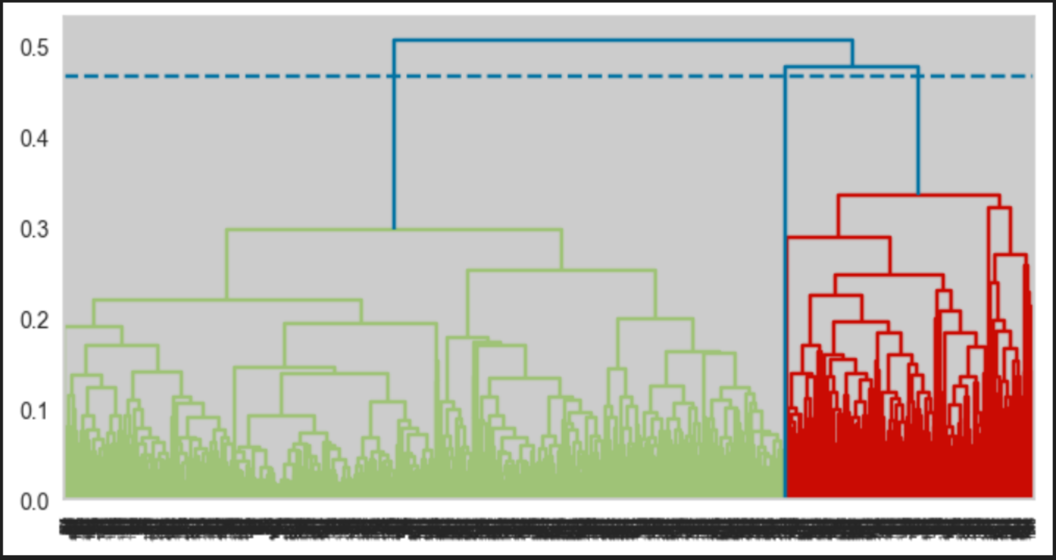
```
scl= normalize(df)
scl= pd.DataFrame(scl, columns=df.columns)
scl.head()
print(scl.head())
✓ 0.6s
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0       0.193478          0.018302     0.000000        0.049677   0.001987   
1       0.106989          0.012071     0.000000        0.035663   0.001344   
2       0.134949          0.013149     0.000692        0.039793   0.001592   
3       0.173611          0.004340     0.008681        0.029452   0.001163   
4       0.193478          0.018302     0.000000        0.049677   0.001987   

   free sulfur dioxide  total sulfur dioxide   density        pH  sulphates  \
0             0.287602              0.888952  0.026088  0.091771   0.014642   
1             0.342913              0.919006  0.013673  0.043893   0.009327   
2             0.259517              0.934261  0.017249  0.056402   0.011246   
3             0.263517              0.930059  0.015470  0.048983   0.008991   
4             0.287602              0.888952  0.026088  0.091771   0.014642   

    alcohol   quality  
0  0.245769  0.130728  
1  0.134422  0.068583  
2  0.169551  0.086506  
3  0.151910  0.093006  
4  0.245769  0.130728  
```

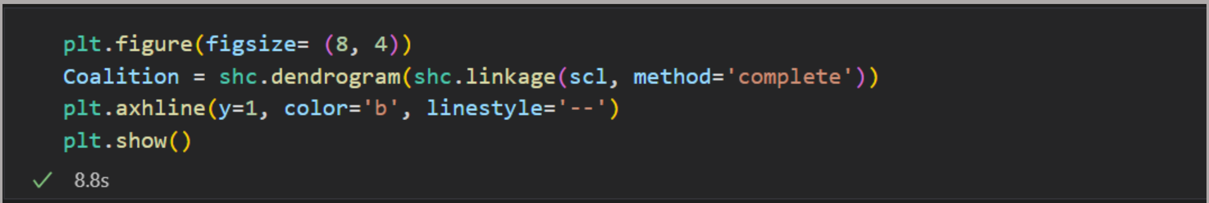
To cluster distance using average linkage

```
plt.figure(figsize= (8, 4))
Coalition = shc.dendrogram(shc.linkage(scl, method='average'))
plt.axhline(y=0.467, color='b', linestyle='--')
plt.show()
```

✓ 9.2s

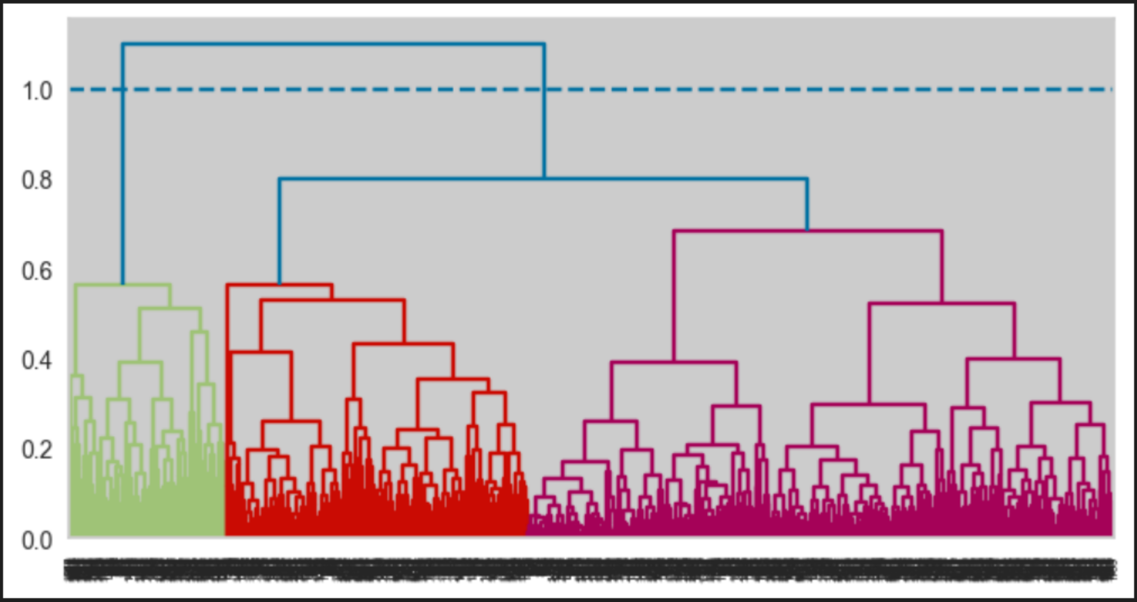


Cluster distance using this complete methode :

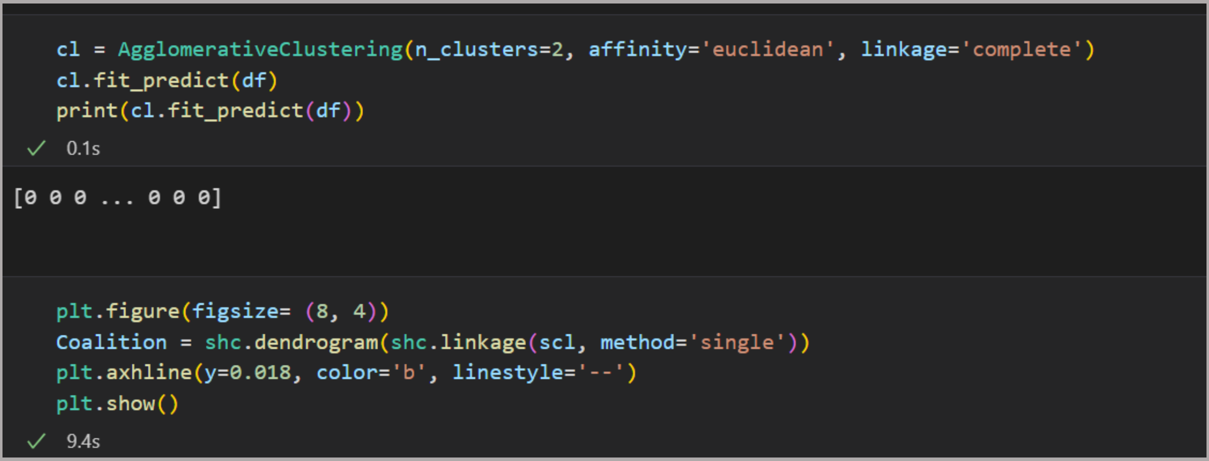


```
plt.figure(figsize= (8, 4))
Coalition = shc.dendrogram(shc.linkage(scl, method='complete'))
plt.axhline(y=1, color='b', linestyle='--')
plt.show()
```

✓ 8.8s



Cluster distance using this single methode :



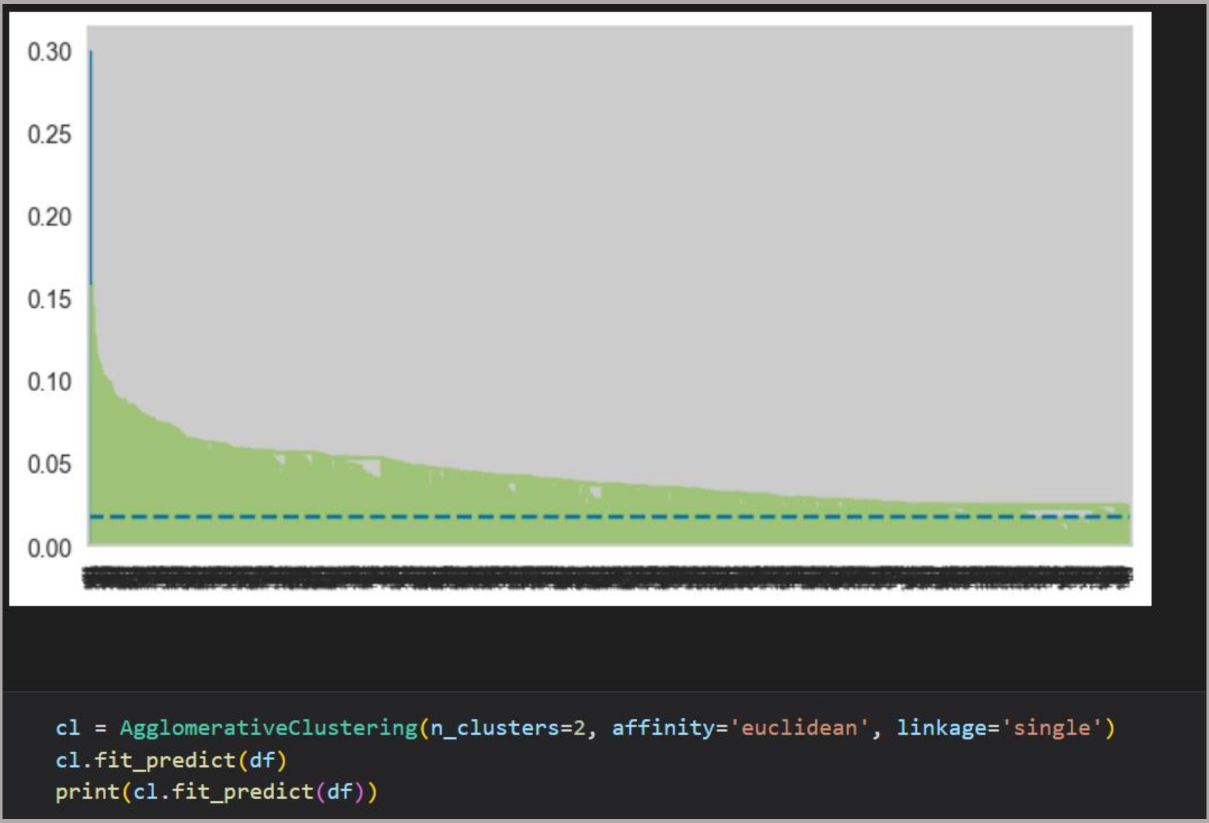
```
cl = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='complete')
cl.fit_predict(df)
print(cl.fit_predict(df))
```

✓ 0.1s

```
[0 0 0 ... 0 0 0]
```

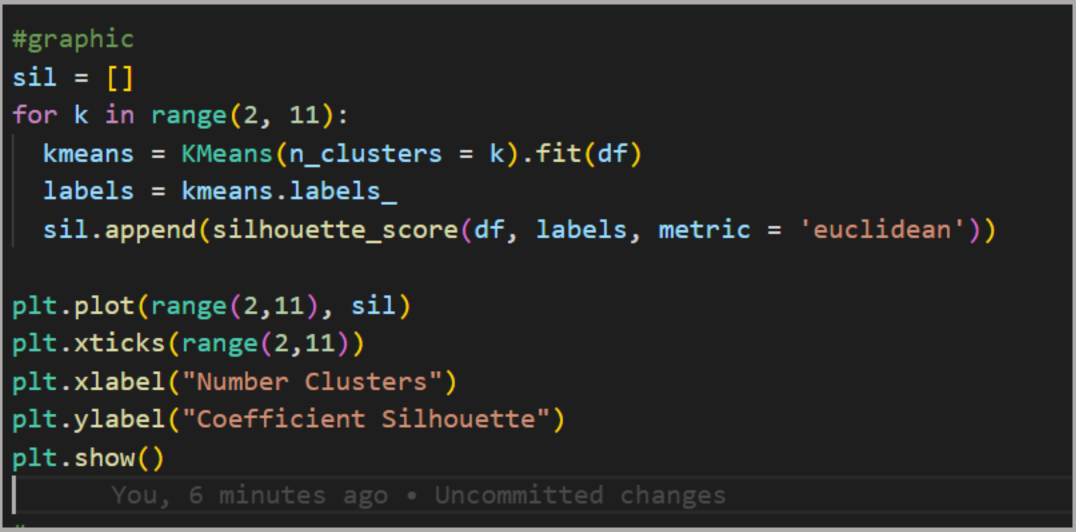
```
plt.figure(figsize= (8, 4))
Coalition = shc.dendrogram(shc.linkage(scl, method='single'))
plt.axhline(y=0.018, color='b', linestyle='--')
plt.show()
```

✓ 9.4s



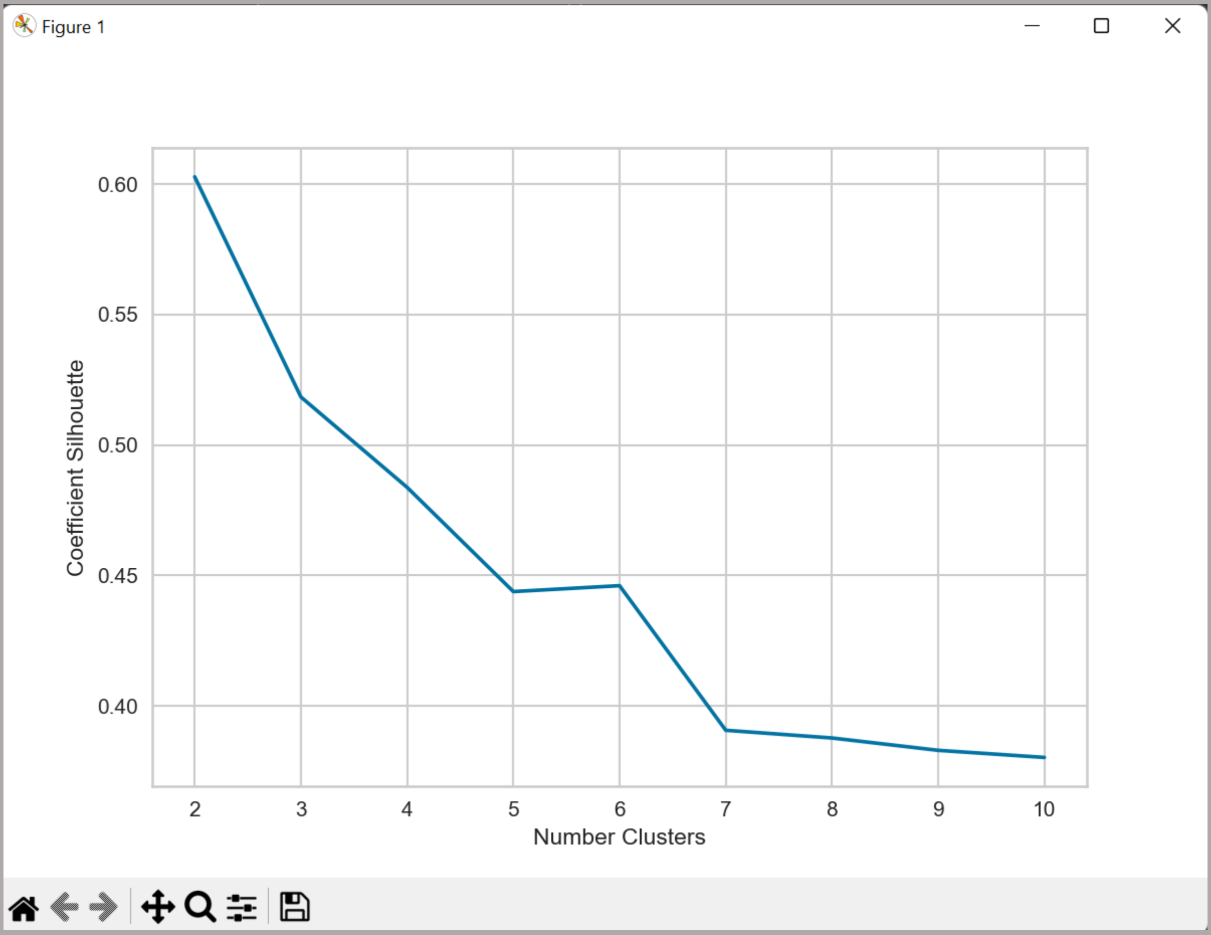
```
cl = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='single')
cl.fit_predict(df)
print(cl.fit_predict(df))
```

Added Evaluation on k-Means ClusteringAssignment with silhouette score and graph methods. First, add the Silhouette graphic code on

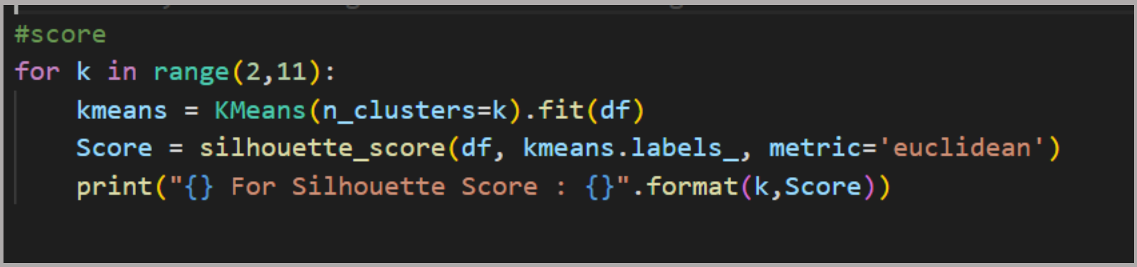


```
#graphic
sil = []
for k in range(2, 11):
  kmeans = KMeans(n_clusters = k).fit(df)
  labels = kmeans.labels_
  sil.append(silhouette_score(df, labels, metric = 'euclidean'))

plt.plot(range(2,11), sil)
plt.xticks(range(2,11))
plt.xlabel("Number Clusters")
plt.ylabel("Coefficient Silhouette")
plt.show()
        You, 6 minutes ago • Uncommitted changes
```

And this is for result :

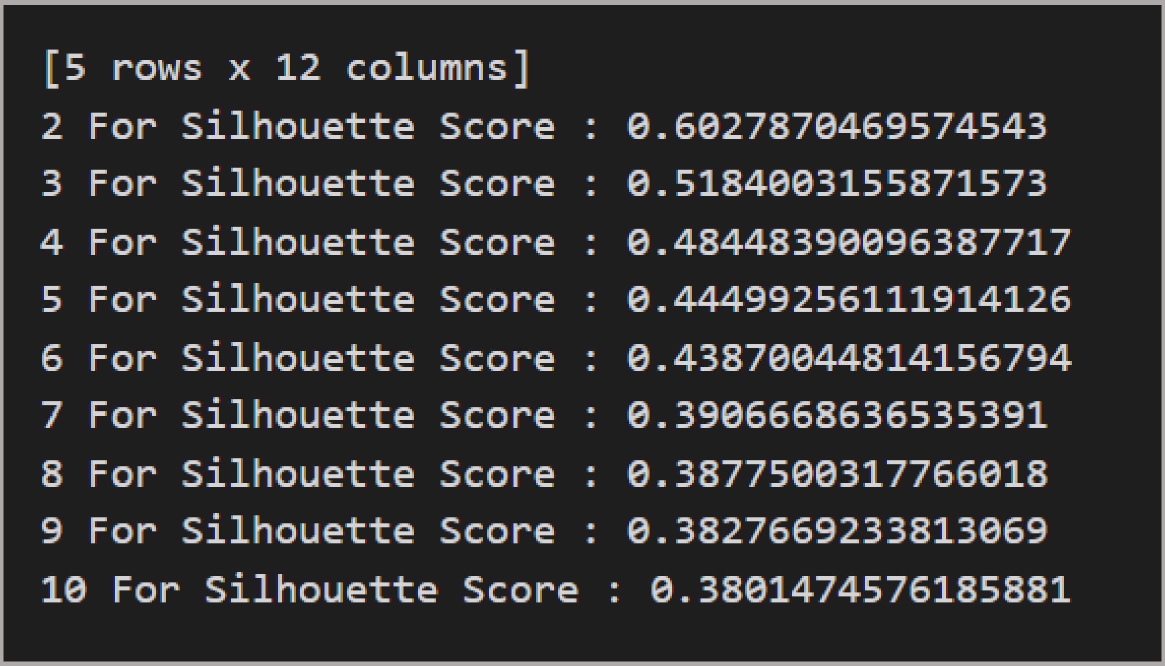


And second added score silhouette code , for calculate silhouette coefficient and easily find exact number of k. :



```
#score
for k in range(2,11):
    kmeans = KMeans(n_clusters=k).fit(df)
    Score = silhouette_score(df, kmeans.labels_, metric='euclidean')
    print("{} For Silhouette Score : {}".format(k,Score))
```

This is the result :



```
[5 rows x 12 columns]
2 For Silhouette Score : 0.6027870469574543
3 For Silhouette Score : 0.5184003155871573
4 For Silhouette Score : 0.48448390096387717
5 For Silhouette Score : 0.44499256111914126
6 For Silhouette Score : 0.43870044814156794
7 For Silhouette Score : 0.3906668636535391
8 For Silhouette Score : 0.3877500317766018
9 For Silhouette Score : 0.3827669233813069
10 For Silhouette Score : 0.3801474576185881
```

Elbow is very simple method that it gives us plot like elbow shape . And we can easily guess optimal number of k from the plot . Maybe we become ambiguity to take decision when we get complex plot because i feel that sometime plot is vague . However , by silhouette method we can calculate silhouette coefficient and easily find exact number of k.