

**KLUSTERISASI WINE DATASET MENGGUNAKAN
METODE *K-MEANS CLUSTERING***

Untuk Memenuhi Tugas Besar 2 Mata Kuliah Machine Learning



Disusun Oleh :

ALMANIK BALINGGA (1301184253)

MOCH. ALFIAN MISBACHUL MUNIR (1301183322)

IF-41-GAB01

S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2020

BAB I

PENDAHULUAN

A. LATAR BELAKANG

Tren Pembelajaran Mesin atau yang lebih dikenal dengan sebutan *Machine Learning* saat ini terus berkembang. Berbagai algoritma terus dikembangkan dan ditemukan. Salah satu jenis pembelajaran yang populer adalah klusterisasi.

Klusterisasi atau clustering adalah suatu jenis metode pembelajaran yang termasuk dalam unsupervised learning. Metode klusterisasi dilakukan jika kita ingin menemukan klaster/ kelompok dari sebuah data. Sementara itu klaster sendiri adalah sebuah kumpulan data atau objek yang memiliki kemiripan satu sama lain di dalam kumpulan atau kelompok tersebut, dan berbeda dengan objek di kelompok lain.

Metode ini biasa digunakan untuk memudahkan kita dalam mengelompokkan data yang berjumlah besar menjadi beberapa kelompok data yang lebih kecil sehingga selanjutnya akan mudah untuk dianalisis. Sehingga metode ini sangat penting untuk dipelajari.

Oleh karena itu, untuk memenuhi tugas besar mata kuliah pembelajaran mesin dan untuk mempelajari metode klusterisasi lebih jauh, kami memutuskan untuk melakukan observasi dengan membuat suatu model pembelajaran mesin yang akan diujikan ke dataset sungguhan.

B. RUMUSAN MASALAH

1. Bagaimana membangun model Clusterisasi menggunakan algoritma k-Means untuk dataset Wine?
2. Berapakah nilai k yang tepat untuk Clusterisasi menggunakan algoritma k-Means?

C. TUJUAN

1. Membangun model Clusterisasi menggunakan algoritma k-Means untuk dataset Wine.
2. Menemukan nilai k yang tepat untuk Clusterisasi menggunakan algoritma k-Means.

BAB II

LANDASAN TEORI

A. KLAUSTERISASI

Berkhin (2006) menyatakan bahwa Clustering / Klusterisasi adalah membagi data ke dalam grup-grup yang mempunyai obyek yang karakteristiknya sama. Garcia (2002) menyatakan clustering adalah mengelompokkan item data ke dalam sejumlah

kecil grup sedemikian sehingga masing-masing grup mempunyai sesuatu persamaan yang esensial.

B. K-MEANS CLUSTERING

K-means clustering merupakan salah satu metode cluster analysis non hirarki yang berusaha untuk mempartisi objek yang ada kedalam satu atau lebih cluster atau kelompok objek berdasarkan karakteristiknya, sehingga objek yang mempunyai karakteristik yang sama dikelompokkan dalam satu cluster yang sama dan objek yang mempunyai karakteristik yang berbeda dikelompokkan kedalam cluster yang lain.

C. EUCLIDEAN DISTANCE

Euclidean distance adalah perhitungan jarak dari dua buah titik dalam Euclidean space. Euclidean space diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi dapat juga diterapkan pada dimensi yang lebih tinggi.

D. WCSS DAN ELBOW METHOD

WCSS atau Within-Cluster Sums of Squares adalah sebuah perhitungan yang berupa penjumlahan dari nilai kuadrat dari jarak antara titik item data dari titik centroidnya. Nantinya semakin kecil nilai dari WCSS ini maka semakin bagus nilai k yang digunakan dalam klasterisasi.

$$WCSS = \sum_{P_i \text{ in cluster } 1} jarak(P_i, C_1)^2 + \sum_{P_i \text{ in cluster } 2} jarak(P_i, C_2)^2 + \sum_{P_i \text{ in cluster } 3} jarak(P_i, C_3)^2$$

Figure 1 - Rumus WCSS

Setelah menggunakan perhitungan WCSS untuk mendapatkan nilai penjumlahan kuadrat jarak, selanjutnya kita harus membuat plot/visualisasi dari nilai WCSS terhadap nilai k yang digunakan dalam klasterisasi.

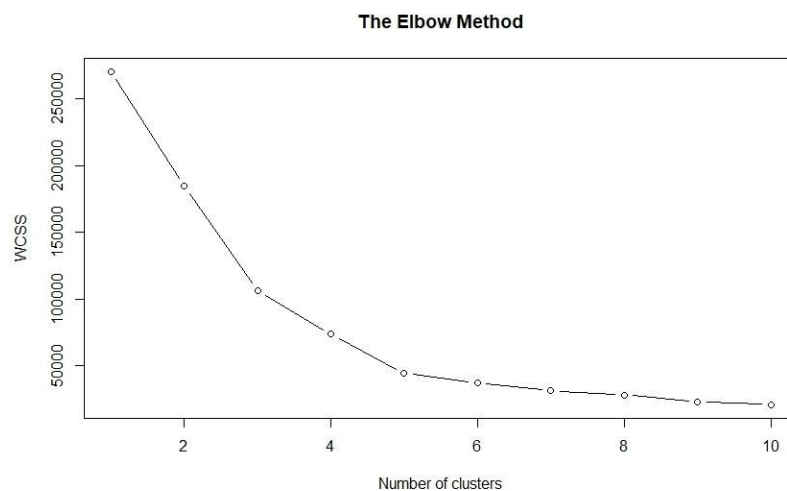


Figure 2- Elbow Method

Untuk memilih nilai k yang paling optimum, maka selanjutnya kita menggunakan elbow method, dimana kita akan memilih nilai k yang menjadi titik dimana grafik berubah menjadi landau / pengurangan jumlah kuadrat jarak tidak terlalu signifikan (biasanya titik yang membentuk siku dari grafik)

BAB III PEMBAHASAN

A. ANALISIS DAN PEMBAHASAN

Untuk melakukan klusterisasi ini, dataset yang kami gunakan adalah dataset Wine for Clustering (wine_clustering.csv) adalah sebuah dataset yang diadaptasi dari dataset yang ada di <https://archive.ics.uci.edu/ml/datasets/wine> dengan menghapus informasi kelas untuk digunakan dalam *Unsupervised Learning*.

Data ini adalah hasil dari analisis kimiawi pertumbuhan anggur di wilayah yang sama di Italia, tetapi tetapi berasal dari tiga kultivar yang berbeda. Analisis menentukan jumlah 13 unsur yang ditemukan di masing-masing dari tiga jenis anggur.

Adapun daftar atributnya adalah sebagai berikut : Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines dan Proline.

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

Figure 3 - Wine Dataset

Dengan rangkuman data sebagai berikut :

	count	mean	std	min	25%	50%	75%	max
Alcohol	178.0	13.000618	0.811827	11.03	12.3625	13.050	13.6775	14.83
Malic_Acid	178.0	2.336348	1.117146	0.74	1.6025	1.865	3.0825	5.80
Ash	178.0	2.366517	0.274344	1.36	2.2100	2.360	2.5575	3.23
Ash_Alcanity	178.0	19.494944	3.339564	10.60	17.2000	19.500	21.5000	30.00
Magnesium	178.0	99.741573	14.282484	70.00	88.0000	98.000	107.0000	162.00
Total_Phenols	178.0	2.295112	0.625851	0.98	1.7425	2.355	2.8000	3.88
Flavanoids	178.0	2.029270	0.998859	0.34	1.2050	2.135	2.8750	5.08
Nonflavanoid_Phenols	178.0	0.361854	0.124453	0.13	0.2700	0.340	0.4375	0.66
Proanthocyanins	178.0	1.590899	0.572359	0.41	1.2500	1.555	1.9500	3.58
Color_Intensity	178.0	5.058090	2.318286	1.28	3.2200	4.690	6.2000	13.00
Hue	178.0	0.957449	0.228572	0.48	0.7825	0.965	1.1200	1.71
OD280	178.0	2.611685	0.709990	1.27	1.9375	2.780	3.1700	4.00
Proline	178.0	746.893258	314.907474	278.00	500.5000	673.500	985.0000	1680.00

Figure 4 - Wine Dataset Summary

Kemudian kami menormalisasi dataset agar dengan menggunakan fungsi normalisasi agar data tahan terhadap pencilan data/*outliers*.

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Figure 5- Rumus Normalisasi

Setelah data di normalisasi, kemudian kami melakukan klusterisasi menggunakan 10 nilai k pada interval 1 sampai 10 dengan perhitungan jarak menggunakan rumus *Euclidean Distance*, sehingga didapat grafik nilai WCSS terhadap nilai k sebagai berikut :

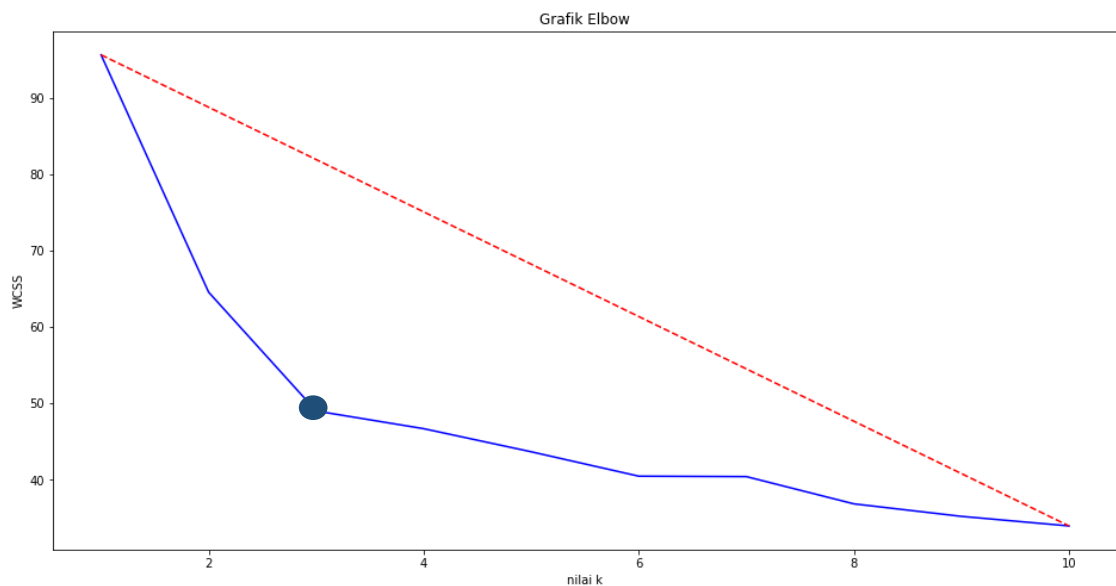


Figure 6 - Grafik WCSS

Berdasarkan grafik di atas, titik dimana nilai WCSS mulai berhenti berubah secara signifikan adalah pada saat nilai $k = 3$, sehingga berdasarkan elbow method, klusterisasi akan menghasilkan kluster yang terbaik pada saat nilai $k = 3$.

Dari hasil tersebut, kami lalu mengklusterisasi dataset dengan nilai $k = 3$ yang kemudian menghasilkan kluster sebagai berikut :

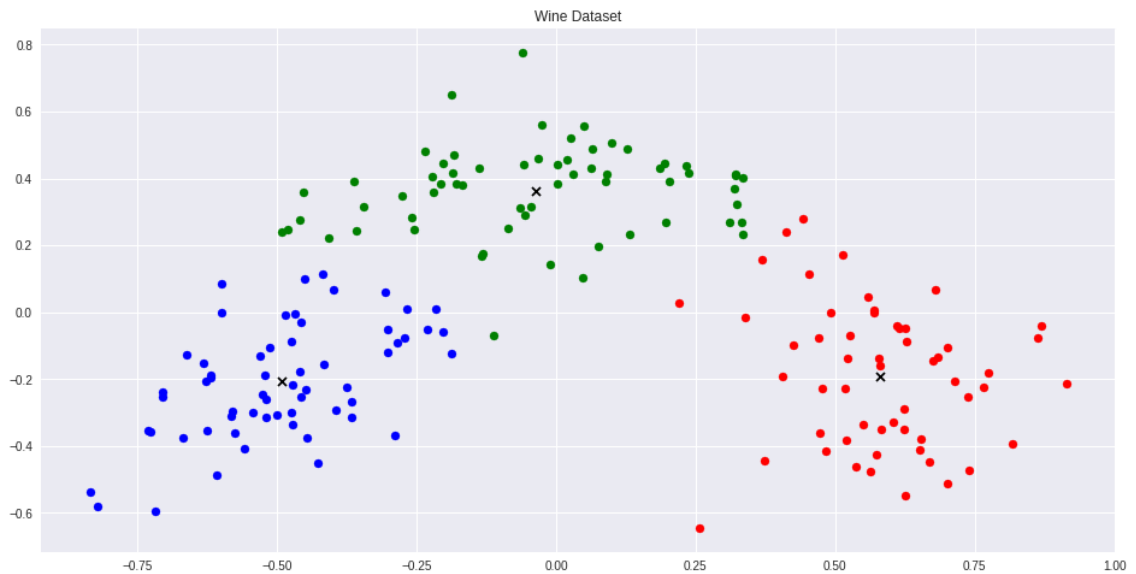


Figure 7 - Hasil Clustering

Dengan rincian klaster yang diberi label “0” untuk klaster pertama, “1” untuk klaster kedua dan “2” untuk klaster ketiga sebagai berikut.

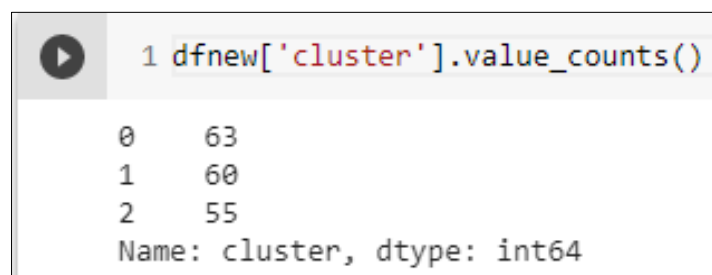


Figure 8 - Jumlah Tiap Cluster

BAB IV PENUTUP

A. KESIMPULAN

Berdasarkan percobaan yang kami lakukan, kami dapat membuat sebuah model Clustering menggunakan metode k-Means dengan menggunakan jarak *Euclidean*. Selain itu, kami juga mendapat nilai k yang paling optimum yaitu $k = 3$, dengan menggunakan metode *Elbow Method*.

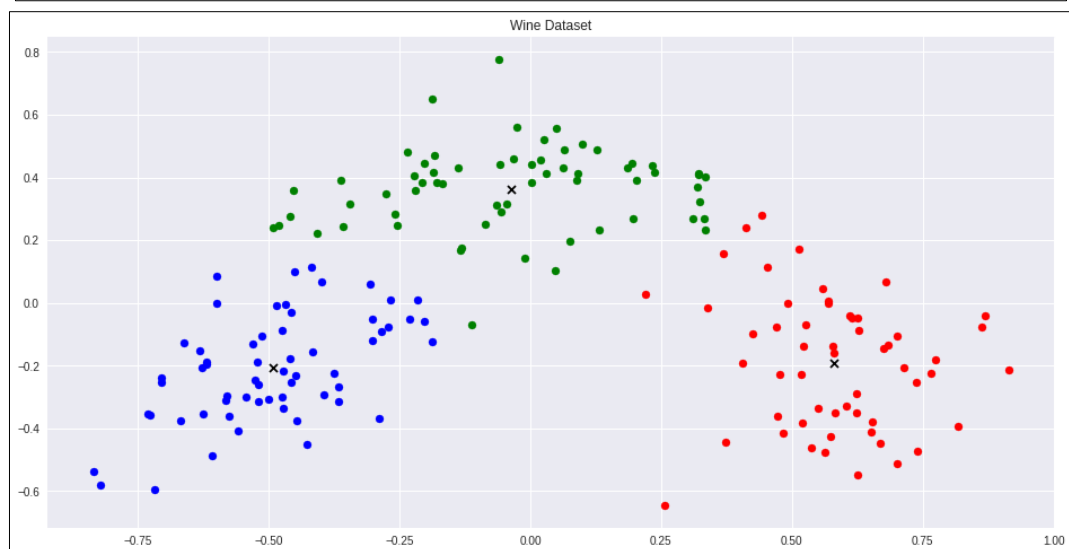
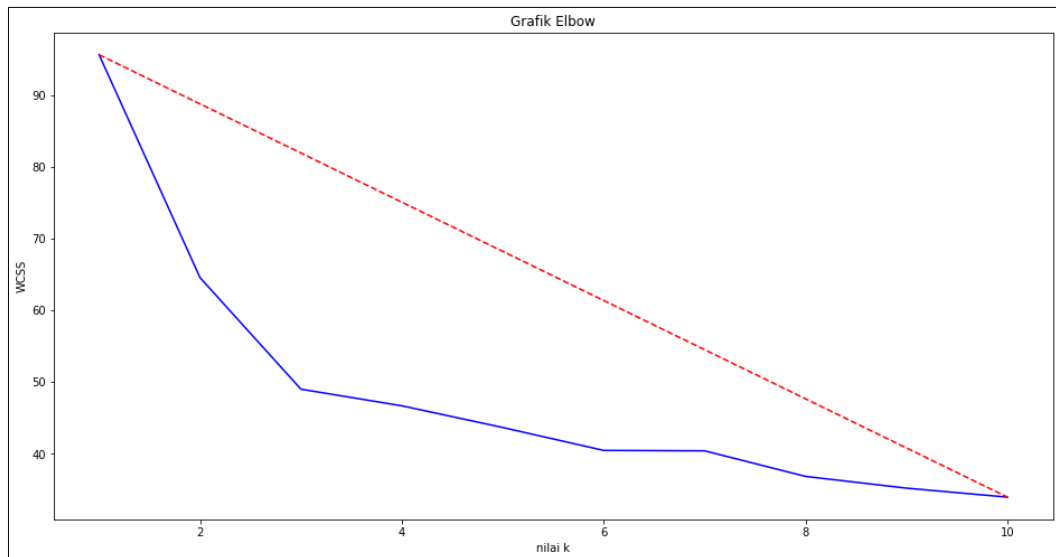
Dari hasil tersebut, dapat kami simpulkan bahwa metode k-Means dapat menghasilkan klasterisasi data yang bagus apabila didukung dengan nilai k yang tepat pula, sehingga proses klasterisasi menghasilkan klaster yang paling optimum.

LAMPIRAN

Link menuju file code : https://drive.google.com/file/d/1UVq2vMJ_LA-DgXF1LfIEJaOrLiZTF5SM/view?usp=sharing

Link dataset : <https://www.kaggle.com/harrywang/wine-dataset-for-clustering/>

Screen Shot Output




▼ Jumlah Klaster

```
[43] 1 dfnew['cluster'].value_counts()
```

```
0    63
1    60
2    55
Name: cluster, dtype: int64
```

Screen Shot Code :

 clusteringML.ipynb ☆

File Edit Lihat Sisipkan Runtime Fitur Bantuan Sem

+ Kode + Teks

Connecting to GDrive

```
[1] 1 from google.colab import drive
    2 drive.mount('/gdrive')
    3 %cd /gdrive/My Drive/tubes/ML_CLS
```

Mounted at /gdrive
/gdrive/My Drive/tubes/ML_CLS

Import Library

```
[2] 1 import pandas as pd
    2 import numpy as np
    3 import matplotlib.pyplot as plt
```

PREPROCESSING

Load Dataframe

```
[3] 1 df = pd.read_csv('wine-clustering.csv')
[4] 1 df.head()
```

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

```
1 df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Alcohol	178.0	13.00618	0.811827	11.03	12.3625	13.050	13.6775	14.83
Malic_Acid	178.0	2.336348	1.117146	0.74	1.6025	1.865	3.0825	5.80
Ash	178.0	2.366517	0.274344	1.36	2.2100	2.360	2.5575	3.23
Ash_Alcanity	178.0	19.494944	3.339564	10.60	17.2000	19.500	21.5000	30.00
Magnesium	178.0	99.741573	14.282484	70.00	88.0000	98.000	107.0000	162.00
Total_Phenols	178.0	2.295112	0.625851	0.98	1.7425	2.355	2.8000	3.88
Flavanoids	178.0	2.029270	0.998859	0.34	1.2050	2.135	2.8750	5.08
Nonflavanoid_Phenols	178.0	0.361854	0.124453	0.13	0.2700	0.340	0.4375	0.66
Proanthocyanins	178.0	1.590899	0.572359	0.41	1.2500	1.555	1.9500	3.58
Color_Intensity	178.0	5.058090	2.318286	1.28	3.2200	4.690	6.2000	13.00
Hue	178.0	0.957449	0.228572	0.48	0.7825	0.965	1.1200	1.71
OD280	178.0	2.611685	0.709990	1.27	1.9375	2.780	3.1700	4.00
Proline	178.0	746.893258	314.907474	278.00	500.5000	673.500	985.0000	1680.00


```
[ ] 1 df.isna().sum()

Alcohol      0
Malic_Acid    0
Ash           0
Ash_Alcanity  0
Magnesium     0
Total_Phenols 0
Flavanoids    0
Nonflavanoid_Phenols 0
Proanthocyanins 0
Color_Intensity 0
Hue           0
OD280         0
Proline       0
dtype: int64
```

▸ 'Float'ing Everything

```
1 df.astype('float64')
2 df.head()
```

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

▸ Fungsi Normalisasi

```
[8] 1 def normal(x):
    2 | return (x-x.min()) / (x.max()-x.min())
```

▾ Menormalisasi Data

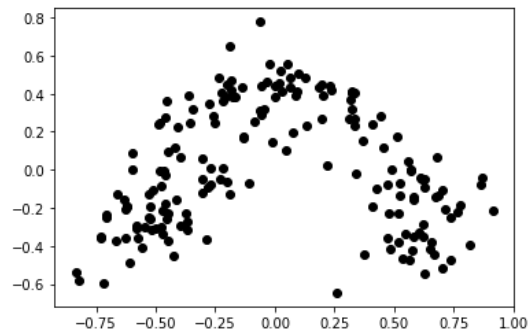
```
[9] 1 col = list(df.columns)
    2 for i in range(len(col)):
    3 | df[[col[i]]] = df[[col[i]]].apply(normal)

[10] 1 df.head()
```

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
0	0.842105	0.191700	0.572193	0.257732	0.619565	0.627586	0.573840	0.283019	0.593060	0.372014	0.455285	0.970696	0
1	0.571053	0.205534	0.417112	0.030928	0.326087	0.575862	0.510549	0.245283	0.274448	0.264505	0.463415	0.780220	0
2	0.560526	0.320158	0.700535	0.412371	0.336957	0.627586	0.611814	0.320755	0.757098	0.375427	0.447154	0.695971	0
3	0.878947	0.239130	0.609626	0.319588	0.467391	0.989655	0.664557	0.207547	0.558360	0.556314	0.308943	0.798535	0
4	0.581579	0.365613	0.807487	0.536082	0.521739	0.627586	0.495781	0.490566	0.444795	0.259386	0.455285	0.608059	0

▼ VISUALISASI DATA

```
[11] 1 from sklearn.decomposition import PCA
      2 pca = PCA(n_components=2).fit(df)
      3 pca_2d = pca.transform(df)
      4 plt.scatter(pca_2d[:,0],pca_2d[:,1],c='black')
      5 plt.show()
```



▼ FUNGSI CLUSTERISASI

▼ PILIH INDEX CENTROID

```
[12] 1 def centroidindexpick(k,dfnya):
      2 |     return np.random.choice(len(dfnya['Ash']),k, replace=False)
```

▼ AMBIL DATA CENTROID

```
[13] 1 def centroidpick(key,dfq):
      2 |     k = centroidindexpick(key,dfq)
      3 |     centroid = []
      4 |     for i in k:
      5 |         centroid.append([i,list(dfq.iloc[i])])
      6 |     return centroid
      7
```

▼ MENENTUKAN JARAK

```
[14] 1 def euclidean(kiri,kanan):
      2 |     sum = 0
      3 |     for i in range(len(kiri)):
      4 |         sum = sum + (kiri[i]-kanan[i])**2
      5 |     return np.sqrt(sum)
```

```
[15] 1 def jarak(dfww, centroid):
      2 |     pilihcluster = []
      3 |     for i in range(len(centroid)):
      4 |         temp =[]
      5 |         for j in range(len(list(dfww['Ash']))):
      6 |             p1 = centroid[i][1]
      7 |             p2 = list(dfww.iloc[j])
      8 |             temp.append([str(i),euclidean(p1,p2)])
      9 |         pilihcluster.append(temp)
     10 |     return pilihcluster
     11
```

▼ MEMILIH CLUSTER

```
[16] 1 def pilihcluster(jarak):
      2     clusternya=[]
      3     elbow = []
      4
      5     for i in range(len(jarak[0])):
      6         pilihjarak = []
      7         pilihclusterjarak = []
      8         for j in range(len(jarak)):
      9             pilihclusterjarak.append(jarak[j][i])
     10             #print(pilihclusterjarak)
     11             pilihjarak.append(jarak[j][i][1])
     12         a = min(pilihjarak)
     13         #print(a)
     14         for k in range(len(pilihclusterjarak)):
     15             if a in pilihclusterjarak[k]:
     16                 clusternya.append(int(pilihclusterjarak[k][0]))
     17                 elbow.append(pilihclusterjarak[k][1]**2)
     18                 #print(clusternya)
     19                 break
     20         elbownya = sum(elbow)
     21     return clusternya,elbownya
     22
```

▼ FUNGSI ITERASI CLUSTERING

```
[17] 1 def clusterit(dfg,k,centroid):
      2     clust = jarak(dfg,centroid)
      3     cluster,elbow = pilihcluster(clust)
      4     dfgx = dfg
      5     dfgx.insert(13,'cluster',cluster)
      6     df3 = dfgx.groupby('cluster').mean()
      7     df3.reset_index
      8     #display(df3)
      9     barucentroid = []
     10     for i in range(k):
     11         barucentroid.append([i,list(df3.iloc[i])])
     12     #display(barucentroid)
     13     # print(barucentroid)
     14     return barucentroid
```

▼ FUNGSI CLUSTERING K OPTIMUM

```
[18] 1 def clusterfix(df2,k,centroid):
      2     clust = jarak(df2,centroid)
      3     cluster,elbow = pilihcluster(clust)
      4     df21 = df2
      5     df21.insert(13,'cluster',cluster)
      6     df3 = df21.groupby('cluster').mean()
      7     df3.reset_index
      8     #display(df3)
      9     barucentroid = []
     10     for i in range(k):
     11         barucentroid.append([i,list(df3.iloc[i])])
     12     return df2, barucentroid,elbow
```

Memilih K yang paling optimal

```
[25] 1 a = int(input('Jumlah iterasi K = ')) + 1
      2 elbowgraph = []
      3 for i in range(1,a):
      4     #print(i)
      5     centx = centroidpick(i,df)
      6     newcent = clusterit(df,i,centx)
      7     while (centx != newcent):
      8         centx = newcent
      9         df = df.drop('cluster',axis=1)
      10        newcent = clusterit(df,i,centx)
      11    df = df.drop('cluster',axis=1)
      12    dfnew,fixcent,elbow = clusterfix(df,i,centx)
      13    elbowgraph.append(elbow)
      14    df = df.drop('cluster',axis=1)
      15
      16 fig, ax = plt.subplots(figsize=(16,8))
      17 ax.set_title('Grafik Elbow')
      18 graphX = np.arange(1,a)
      19 ax.plot(graphX,elbowgraph,color='b',label = 'nilai WCSS')
      20 b = [elbowgraph[0],elbowgraph[a-2]]
      21 c = [1,a-1]
      22 ax.plot(c,b, color='red', ls='dashed')
      23 ax.set_xlabel('nilai k')
      24 ax.set_ylabel('WCSS')
      25 plt.show()
```

CLUSTERING DENGAN K = 3

```
[32] 1 import seaborn as sns
      2 k= 3
      3 centx = centroidpick(k,df)
      4 newcent = clusterit(df,k,centx)
      5 while (centx != newcent):
      6     centx = newcent
      7     df = df.drop('cluster',axis=1)
      8     newcent = clusterit(df,k,centx)
      9     df = df.drop('cluster',axis=1)
     10 dfnew,fixcent,elbow = clusterfix(df,k,centx)
     11 df = df.drop('cluster',axis=1)
     12
     13 # PLOTTING
     14
     15 pca = PCA(n_components=2).fit(df)
     16 pca_2d = pca.transform(df)
     17 clusterx = list(dfnew['cluster'])
     18 aa,ab,ba,bb,ca,cb = [],[],[],[],[],[]
     19 for i in range(len(pca_2d[:,0])):
     20     if clusterx[i] == 0:
     21         aa.append(pca_2d[i,0])
     22         ab.append(pca_2d[i,1])
     23     elif clusterx[i] == 1:
     24         ba.append(pca_2d[i,0])
     25         bb.append(pca_2d[i,1])
     26     elif clusterx[i] == 2:
     27         ca.append(pca_2d[i,0])
     28         cb.append(pca_2d[i,1])
     29
     30
     31 d = {0: fixcent[0][1], 1: fixcent[1][1], 2:fixcent[2][1]}
     32 centnyaa = pd.DataFrame(d)
     33 centnyaa = centnyaa.T
     34
     35 pca2 = PCA(n_components=2).fit(centnyaa)
     36 pcacent = pca.transform(centnyaa)
     37 fig, ax = plt.subplots(figsize=(16,8))
     38 ax.set_title('Wine Dataset')
     39 plt.style.use('seaborn')
     40 ax.scatter(aa,ab,c='green')
     41 ax.scatter(ba,bb,c='blue')
     42 ax.scatter(ca,cb,c='red')
     43 ax.scatter(pcacent[:,0],pcacent[:,1], marker='x',color = 'black')
     44 plt.show()
```