LAPORAN CASE BASED-2 MACHINE LEARNING

Disusun oleh :

Muhammad Daffa' Ibrahim (1301204051)

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

TAHUN 2022

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Algoritma Unsupervised Learning merupakan suatu algoritma yang digunakan pada data yang belum memiliki label. Algoritma ini hanya memiliki input dan tidak memiliki output yang dikeluarkan. Melakukan analisis terhadap data yang belum memiliki label tersebut dengan mencari pola tersembunyi dan korelasi dari tiap datanya. Bertujuan untuk melakukan pengelompokkan data (clustering), mendeteksi anomali data, dan lain-lain.

1.2 Spesifikasi Tugas

Diberikan file water treatment plan.xlsx yang berisi dataset mengenai pengolahan air limbah dalam perkotaan. Dataset tersebut menggambarkan record pengolahan air limbah selama ratusan hari. Terdapat 38 atribut yang dimiliki setiap record pada dataset tersebut yang mempengaruhi setiap datanya. Tujuannya yaitu melakukan klasifikasi atau pengelompokkan terhadap dataset tersebut.

	Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	PH-P	...	COND-S	RD-DBO-P	RD-SS-P	RD-SED-P
count	249.000000	249.000000	249.000000	249.000000	249.000000	249.000000	249.000000	249.000000	249.000000	249.000000	...	249.000000	249.000000	249.000000	249.000000
mean	37186.460553	1.802610	7.819277	184.235227	392.959022	192.190542	64.682622	4.143552	1465.028112	7.839759	...	1466.461847	38.153022	57.807455	92.473663
std	5963.901015	1.402932	0.240861	46.212034	97.407315	41.844533	8.655866	1.313396	340.877750	0.218096	...	307.114615	12.475064	9.642151	4.344313
min	26348.000000	0.100000	7.200000	69.000000	105.000000	98.000000	37.600000	1.400000	810.000000	7.400000	...	858.000000	7.300000	29.600000	80.000000
25%	32845.000000	0.800000	7.600000	150.000000	318.000000	166.000000	59.300000	3.000000	1207.000000	7.700000	...	1233.000000	31.200000	51.900000	90.400000
50%	35636.000000	1.300000	7.800000	184.000000	389.000000	186.000000	66.100000	4.000000	1433.000000	7.800000	...	1454.000000	39.085806	58.800000	93.300000
75%	40933.000000	2.500000	8.000000	214.000000	457.000000	216.000000	70.700000	5.000000	1672.000000	8.000000	...	1690.000000	45.800000	64.500000	95.700000
max	53210.000000	6.000000	8.400000	327.000000	688.000000	324.000000	85.000000	7.500000	2350.000000	8.300000	...	2290.000000	70.200000	82.000000	100.000000

8 rows × 38 columns

1.3 Ikhtisar Kumpulan Data yang Dipilih

1.3.1 Missing value

Terdapat banyak missing value pada dataset water treatment ini. Karena terlalu banyak, akan kurang baik jika melakukan drop terhadap data, oleh karenanya saya mengganti nilai missing value dengan nilai rata2 dari atribut tersebut.

Days	0
Q-E	18
ZN-E	3
PH-E	0
DBO-E	23
DQO-E	6
SS-E	1
SSV-E	11
SED-E	25
COND-E	0
PH-P	0
DBO-P	40
SS-P	0
SSV-P	11
SED-P	24
COND-P	0
PH-D	0
DBO-D	28
DQO-D	9
SS-D	2
SSV-D	13
SED-D	25
COND-D	0
PH-S	1
DBO-S	23
DQO-S	18
SS-S	5
SSV-S	17
SED-S	28

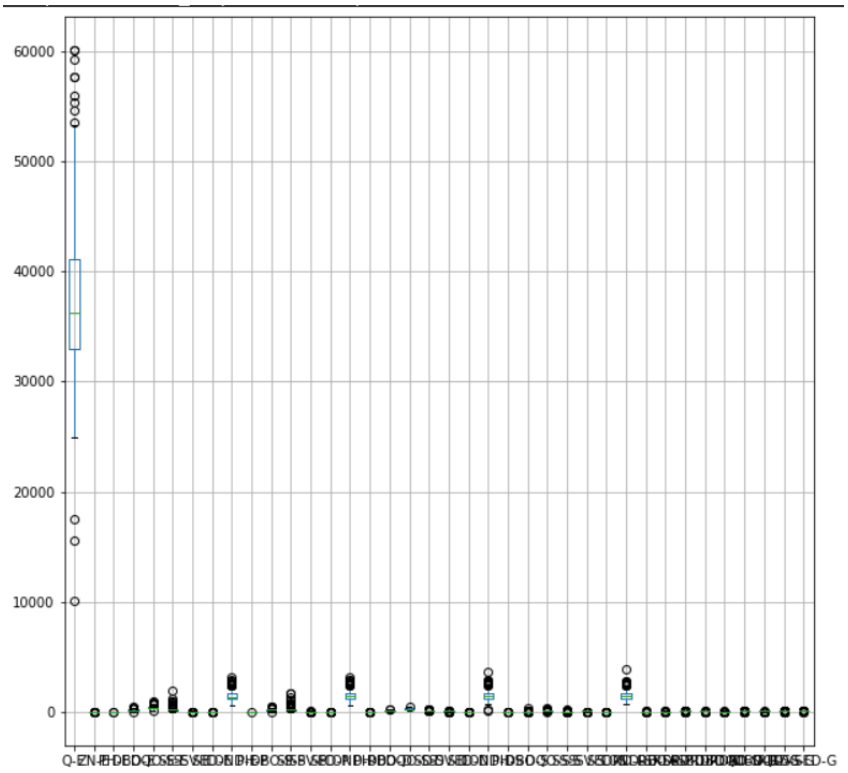
1.3.2 Tipe data yang berbeda

Terdapat beberapa atribut dengan tipe data yang berbeda, agar mempermudah dan membuat proses semakin akurat, maka data diubah menjadi tipe yang seragam.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 527 entries, 0 to 526
Data columns (total 39 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Days      527 non-null    object
 1   Q-E       509 non-null    float64
 2   ZN-E      524 non-null    float64
 3   PH-E      527 non-null    float64
 4   DBO-E     504 non-null    float64
 5   DQO-E     521 non-null    float64
 6   SS-E      526 non-null    float64
 7   SSV-E     516 non-null    float64
 8   SED-E     502 non-null    float64
 9   COND-E    527 non-null    int64
 10  PH-P      527 non-null    float64
 11  DBO-P     487 non-null    float64
 12  SS-P      527 non-null    int64
 13  SSV-P     516 non-null    float64
 14  SED-P     503 non-null    float64
 15  COND-P    527 non-null    int64
 16  PH-D      527 non-null    float64
 17  DBO-D     499 non-null    float64
```

1.3.3 Menghilangkan pencilan data

Terdapat beberapa pencilan data yang dapat mengurangi kualitas data set. Perlu adanya handling terhadap pencilan data sehingga data menjadi bersih dan optimal untuk diproses.



1.3.4 Normalisasi Data

Dataset dengan jumlah yang banyak memiliki nilai data dengan skala beragam. Perlu adanya normalisasi data untuk mempermudah proses pemodelan serta meringankan proses komputasi pada saat pemodelan.

Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	\
0.660896	0.237288	0.500000	0.464009	0.518010	0.300885	0.605485	
0.218934	0.830508	0.333333	0.464009	0.725557	0.389381	0.681435	
0.393716	0.237288	0.666667	0.670543	0.670669	0.345133	0.573840	
0.455067	0.491525	0.500000	0.515504	0.457976	0.389381	0.658228	
0.549736	1.000000	0.500000	0.464009	0.766724	0.725664	0.559072	
...	...	...	...	...	...	...	
0.055469	0.005085	0.333333	0.426357	0.274443	0.132743	0.723629	
0.267553	0.037288	0.500000	0.476744	0.413379	0.327434	0.654008	
0.244732	0.033898	0.166667	0.271318	0.449400	0.362832	0.565401	
0.217482	0.033898	0.083333	0.507752	0.754717	0.707965	0.580169	
0.154121	0.018644	0.250000	0.321705	0.334477	0.150442	0.677215	

SED-E	COND-E	PH-P	...	COND-S	RD-DBO-P	RD-SS-P	\
0.508197	0.844156	0.555556	...	0.797486	0.505339	0.557252	
0.327869	0.555844	0.333333	...	0.719274	0.505339	0.545802	
0.426230	0.844156	0.555556	...	0.881285	0.505339	0.631679	
0.508197	0.541558	0.444444	...	0.632682	0.505339	0.574427	
0.590164	0.514935	0.444444	...	0.590084	0.505339	0.616412	
...	...	...	...	...	...	...	
0.065574	0.118182	0.222222	...	0.129888	0.101749	0.293893	
0.426230	0.115584	0.444444	...	0.064246	0.505339	0.547710	
0.262295	0.162338	0.111111	...	0.194134	0.505339	0.675573	
0.426230	0.292208	0.000000	...	0.326816	0.516693	0.692748	
0.523578	0.170779	0.000000	...	0.255587	0.505339	0.761450	

1.3.5 Reduksi Dimensi

Dataset Water Treatment memiliki cukup banyak atribut yaitu sebanyak 38 atribut. Perlu adanya reduksi dimensi atau pemangkasan atribut untuk memudahkan proses pemodelan.

PCA_1	PCA_2
-0.419269	-0.244035
-0.839763	-0.181101
-0.282144	-0.169161
-0.799293	-0.129454
-0.649815	-0.144493
...	...
0.978394	-0.190840
0.533344	-0.149866
0.480775	-0.097264
0.154976	0.191929
0.630016	-0.141099

BAB 2
PRA-PEMROSESAN DATA

2.1 Missing Value

Handling missing value dengan mengganti nilai tersebut dengan nilai rata-rata pada atributnya.

```
df['Q-E'].fillna(df['Q-E'].mean(), inplace = True)
df['ZN-E'].fillna(df['ZN-E'].mean(), inplace = True)
df['DBO-E'].fillna(df['DBO-E'].mean(), inplace = True)
df['DQO-E'].fillna(df['DQO-E'].mean(), inplace = True)
df['SS-E'].fillna(df['SS-E'].mean(), inplace = True)
df['SSV-E'].fillna(df['SSV-E'].mean(), inplace = True)
df['SED-E'].fillna(df['SED-E'].mean(), inplace = True)
df['DBO-P'].fillna(df['DBO-P'].mean(), inplace = True)
df['SSV-P '].fillna(df['SSV-P '].mean(), inplace = True)
df['SED-P'].fillna(df['SED-P'].mean(), inplace = True)
df['DBO-D'].fillna(df['DBO-D'].mean(), inplace = True)
df['DQO-D'].fillna(df['DQO-D'].mean(), inplace = True)
df['SS-D'].fillna(df['SS-D'].mean(), inplace = True)
df['SSV-D'].fillna(df['SSV-D'].mean(), inplace = True)
df['SED-D'].fillna(df['SED-D'].mean(), inplace = True)
df['PH-S'].fillna(df['PH-S'].mean(), inplace = True)
df['DBO-S'].fillna(df['DBO-S'].mean(), inplace = True)
df['DQO-S'].fillna(df['DQO-S'].mean(), inplace = True)
df['SS-S'].fillna(df['SS-S'].mean(), inplace = True)
df['SSV-S'].fillna(df['SSV-S'].mean(), inplace = True)
df['SED-S'].fillna(df['SED-S'].mean(), inplace = True)
df['COND-S'].fillna(df['COND-S'].mean(), inplace = True)
df['RD-DBO-P'].fillna(df['RD-DBO-P'].mean(), inplace = True)
df['RD-SS-P'].fillna(df['RD-SS-P'].mean(), inplace = True)
df['RD-SED-P'].fillna(df['RD-SED-P'].mean(), inplace = True)
df['RD-DBO-S'].fillna(df['RD-DBO-S'].mean(), inplace = True)
df['RD-DQO-S'].fillna(df['RD-DQO-S'].mean(), inplace = True)
df['RD-DBO-G'].fillna(df['RD-DBO-G'].mean(), inplace = True)
df['RD-DQO-G'].fillna(df['RD-DQO-G'].mean(), inplace = True)
df[' RD-SS-G'].fillna(df[' RD-SS-G'].mean(), inplace = True)
df['RD-SED-G'].fillna(df['RD-SED-G'].mean(), inplace = True)
```

Menghasilkan tidak terdapat lagi missing value pada dataset water treatment

```
Days      0
Q-E       0
ZN-E      0
PH-E      0
DBO-E     0
DQO-E     0
SS-E      0
SSV-E     0
SED-E     0
COND-E    0
PH-P      0
DBO-P     0
SS-P      0
SSV-P     0
SED-P     0
COND-P    0
PH-D      0
DBO-D     0
DQO-D     0
SS-D      0
SSV-D     0
SED-D     0
COND-D    0
PH-S      0
DBO-S     0
DQO-S     0
SS-S      0
SSV-S     0
SED-S     0
```

1.3.2 Tipe Data yang Berbeda

Mengganti tipe data yang masih berbeda dengan tipe data float, sehingga setiap data pada dataset bertipe float.

```
convert_dict = {'COND-E': float,
                'SS-P': float,
                'COND-P': float,
                'COND-D': float,
                }
df = df.astype(convert_dict)
df.info()
```

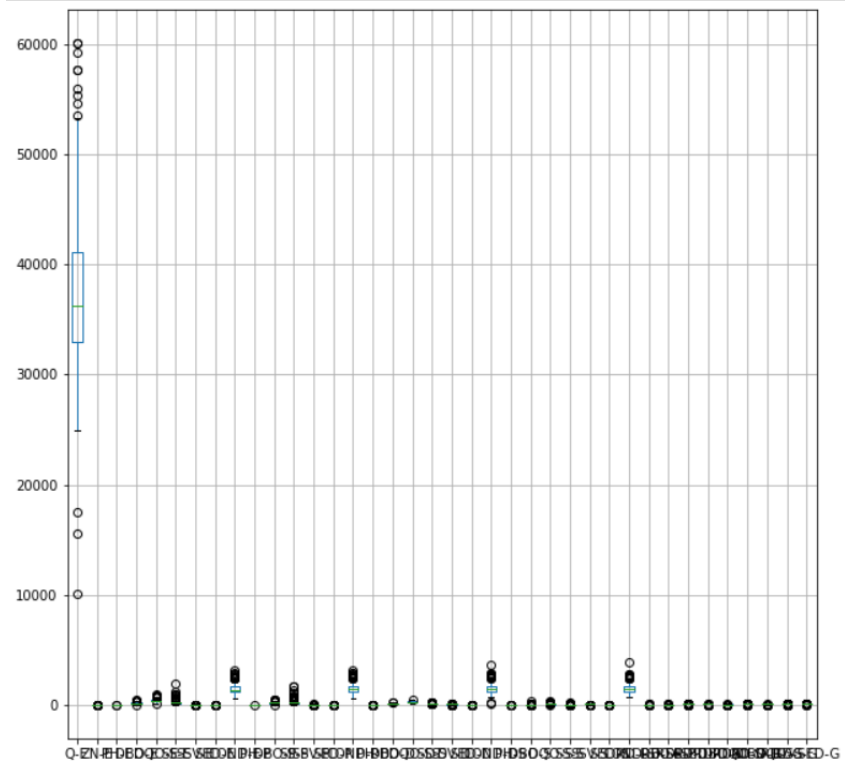
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 527 entries, 0 to 526
Data columns (total 39 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Days      527 non-null    object
 1   Q-E       527 non-null    float64
 2   ZN-E      527 non-null    float64
 3   PH-E      527 non-null    float64
 4   DBO-E     527 non-null    float64
 5   DQO-E     527 non-null    float64
 6   SS-E      527 non-null    float64
 7   SSV-E     527 non-null    float64
 8   SED-E     527 non-null    float64
 9   COND-E    527 non-null    float64
 10  PH-P      527 non-null    float64
 11  DBO-P     527 non-null    float64
 12  SS-P      527 non-null    float64
 13  SSV-P     527 non-null    float64
 14  SED-P     527 non-null    float64
```

1.3.3 Menghilangkan pencilan data

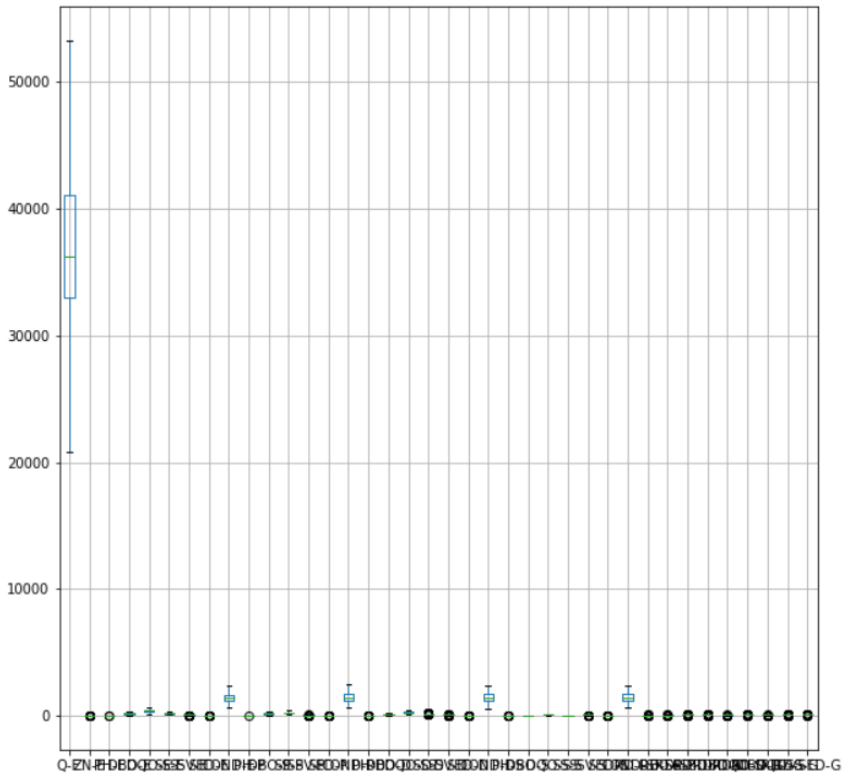
Terdapat beberapa outlier data yang dapat menyebabkan dataset kurang bagus untuk dilakukan pemodelan. Outlier tersebut ditangani dengan memberi batasan terhadap outlier. Batas nilai maksimum yang digunakan adalah nilai (q3 + (interquartile * 1.5)) sedangkan batas nilai minimum yang digunakan adalah nilai (q1 - (interquartile * 1.5)). Jika terdapat nilai yang melebihi batas maksimum, maka nilai tersebut akan di assign dengan nilai pada batas maksimum, Jika terdapat nilai yang lebih kecil batas minimum, maka nilai tersebut akan di assign dengan nilai pada batas minimum.

```
# menghitung jarak interquartile
def interquartile(data,x):
  q1 = (data[x]).quantile(0.25)
  q3 = (data[x]).quantile(0.75)
  iqr = q3 - q1
  maximum = q3 + (1.5 *iqr)
  minimum = q1 - (1.5 *iqr)
  return maximum,minimum

# menggantikan value outliners dengan hasil dari perhitungan jarak interquartile
def sub_outliners(data,x,maximum,minimum):
  more_than = (data[x] > maximum)
  less_than = (data[x] < minimum)
  data[x] = data[x].mask(more_than, maximum,axis=0)
  data[x] = data[x].mask(less_than, minimum,axis=0)
  return data

maximum,minimum = interquartile(df,'Q-E')
df = sub_outliners(df,'Q-E',maximum,minimum)

maximum,minimum = interquartile(df,'DBO-E')
df = sub_outliners(df,'DBO-E',maximum,minimum)

maximum,minimum = interquartile(df,'DQO-E')
df = sub_outliners(df,'DQO-E',maximum,minimum)

maximum,minimum = interquartile(df,'SS-E')
df = sub_outliners(df,'SS-E',maximum,minimum)
```

Data sebelum handling outlier



Data setelah handling outlier



1.3.4 Normalisasi Data

Dataset yang digunakan memiliki nilai data yang beragam, terdapat data dengan nilai data yang berjumlah besar, terdapat juga data dengan nilai data yang kecil. Oleh Karena itu, untuk memudahkan proses eksekusi pemodelan, perlu adanya normalisasi data untuk mengubah nilai dari data menjadi data berskala 0 - 1.

```
dt = (dt-np.min(dt))/(np.max(dt)-np.min(dt))
print(dt)
```

Sehingga setiap nilai data berskala 0-1.

```
     Q-E      ZN-E      PH-E     DBO-E     DQO-E      SS-E      SSV-E  \
0.717466  0.041916  0.500000  0.512548  0.510980  0.269841  0.739554
0.561347  0.086826  0.444444  0.512548  0.571791  0.460317  0.779944
0.352399  0.146707  0.388889  0.512548  0.715372  0.349206  0.789694
0.438315  0.101796  0.555556  0.567568  0.816723  0.373016  0.729805
0.496771  0.041916  0.611111  0.692568  0.661318  0.309524  0.718663
     ...       ...       ...       ...       ...       ...       ...
0.367589  0.001796  0.444444  0.189189  0.249155  0.309524  0.607242
0.392558  0.006587  0.500000  0.523649  0.407939  0.293651  0.771588
0.373708  0.005988  0.277778  0.344595  0.443412  0.325397  0.713092
0.351199  0.005988  0.222222  0.550676  0.744088  0.634921  0.722841
0.298862  0.003293  0.333333  0.388514  0.330236  0.134921  0.786908
```

1.3.5 Reduksi Dimensi

Dataset Trial.xlsx memiliki cukup banyak atribut, yaitu sebanyak 38 atribut yang berbeda. Jumlah atribut yang banyak akan menyulitkan proses pemodelan dan memperburuk kualitas dataset. oleh karena itu dibutuhkan proses reduksi dimensi untuk membuat atribut menjadi lebih sedikit. Metode proses reduksi dimensi yang digunakan adalah metode Principal Component Analysis (PCA). Principal Component Analysis (PCA) adalah salah satu metode reduksi dimensi pada machine learning. PCA akan memilih variabel-variabel yang mampu menjelaskan sebagian besar variabilitas data. PCA mengurangi dimensi dengan membentuk variabel-variabel baru yang disebut Principal Components.

```
pca = PCA(n_components=2)
fit_pca = pca.fit_transform(dt)
data_fit = pd.DataFrame(data = fit_pca, columns = ['PCA_1', 'PCA_2'])
```

```
    PCA_1     PCA_2
-0.419269 -0.244035
-0.839763 -0.181101
-0.282144 -0.169161
-0.799293 -0.129454
-0.649815 -0.144493
      ...       ...
 0.978394 -0.190840
 0.533344 -0.149866
 0.480775 -0.097264
 0.154976  0.191929
 0.630016 -0.141099
```

BAB 3
ALGORITMA

Algoritma K-MEANS merupakan algoritma yang membutuhkan parameter input sebanyak k dan membagi sekumpulan n objek kedalam k cluster sehingga tingkat kemiripan antar anggota dalam satu cluster tinggi sedangkan tingkat kemiripan dengan anggota pada cluster lain sangat rendah. Kemiripan anggota terhadap cluster diukur dengan kedekatan objek terhadap nilai mean pada cluster atau dapat disebut sebagai centroid cluster. Algoritma K-MEANS digunakan karena sederhana dan relatif mudah dan cepat.

3.1 Menentukan Centroid Awal

Centroid awal didapat dari data ke-1 sebagai centroid 1 dan data ke-2 sebagai centroid 2.

```
c1x1 = data_fit['PCA_1'][0]
c1x2 = data_fit['PCA_2'][0]

c2x1 = data_fit['PCA_1'][1]
c2x2 = data_fit['PCA_2'][1]
```

3.2 Perhitungan Jarak

Perhitungan jarak antara centroid dengan data menggunakan metode euclidean distance dengan rumus sebagai berikut:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

implementasi pada kode:

```
def hitungJarak(x1,x2,c1,c2):
  jarak = math.sqrt((x1-c1)**2 +(c2-x2)**2)
  return jarak
```

3.3 Clustering berdasarkan jarak terdekat

Melakukan clustering sesuai dengan jarak terdekat. Jika data berjarak lebih dekat dengan centroid 1, maka akan di assign pada cluster 1, sebaliknya, jika data berjarak lebih dekat dengan centroid 2, maka akan di assign pada cluster 2.

```
for i in range(len(data_fit)):
  j1 = hitungJarak(data_fit['PCA_1'][i],data_fit['PCA_2'][i],c1x1,c1x2)
  j2 = hitungJarak(data_fit['PCA_1'][i],data_fit['PCA_2'][i],c2x1,c2x2)
  if j1 < j2 :
    p1c1.append(data_fit['PCA_1'][i])
    p2c1.append(data_fit['PCA_2'][i])
  else:
    p1c2.append(data_fit['PCA_1'][i])
    p2c2.append(data_fit['PCA_2'][i])
```

3.4 Menghitung nilai rata-rata dari tiap cluster

Melakukan perhitungan terhadap nilai rata-rata pada tiap cluster yang nantinya akan menjadi nilai centroid baru.

```
mc1x1 = statistics.mean(p1c1)
mc1x2 = statistics.mean(p2c1)
```

3.5 Mengganti centroid lama dengan centroid baru

Hasil perhitungan nilai rata-rata pada tiap cluster akan menjadi nilai centroid baru untuk perulangan berikutnya. Jika hasil perhitungan rata-rata sama dengan nilai centroid sebelumnya, maka proses perulangan dihentikan dan berhasil mendapatkan hasil klasifikasi.

```
if(mc1x1 == c1x1 and mc1x2 == c1x2 and mc2x1 == c2x1 and mc2x2 == c2x2):
  break
else:
  c1x1 = mc1x1
  c1x2 = mc1x2

  c2x1 = mc2x1
  c2x2 = mc2x2
```

3.6 Implementasi kode

```
c1x1 = data_fit['PCA_1'][0]
c1x2 = data_fit['PCA_2'][0]

c2x1 = data_fit['PCA_1'][1]
c2x2 = data_fit['PCA_2'][1]

j = 1
while(True):
  print("Iterasi Ke- ",j)
  p1c1 = []
  p2c1 = []
  c1 = {
      "PCA_1":p1c1,
      "PCA_2":p2c1,
  }
  pc1 = pd.DataFrame(c1)

  p1c2 = []
  p2c2 = []
  c2 = {
      "PCA_1":p1c2,
      "PCA_2":p2c2,
  }
  pc2 = pd.DataFrame(c2)

  print("Centroid 1: (",c1x1,",",c1x2,")")
  print("Centroid 2: (",c2x1,",",c2x2,")")
```

```
for i in range(len(data_fit)):
  j1 = hitungJarak(data_fit['PCA_1'][i],data_fit['PCA_2'][i],c1x1,c1x2)
  j2 = hitungJarak(data_fit['PCA_1'][i],data_fit['PCA_2'][i],c2x1,c2x2)
  if j1 < j2 :
    p1c1.append(data_fit['PCA_1'][i])
    p2c1.append(data_fit['PCA_2'][i])
  else:
    p1c2.append(data_fit['PCA_1'][i])
    p2c2.append(data_fit['PCA_2'][i])

mc1x1 = statistics.mean(p1c1)
mc1x2 = statistics.mean(p2c1)

mc2x1 = statistics.mean(p1c2)
mc2x2 = statistics.mean(p2c2)

plt.scatter(c1['PCA_1'], c1['PCA_2'], color = 'green')
plt.scatter(c2['PCA_1'], c2['PCA_2'], color = 'pink')
plt.scatter(c1x1, c1x2, color = 'Red')
plt.scatter(c2x1, c2x2, color = 'Red')
plt.title("Grafik Hasil Klasterisasi Cluster1 dan Cluster 2 pada Data PC1_1 dan PCA_2")
plt.show()
```

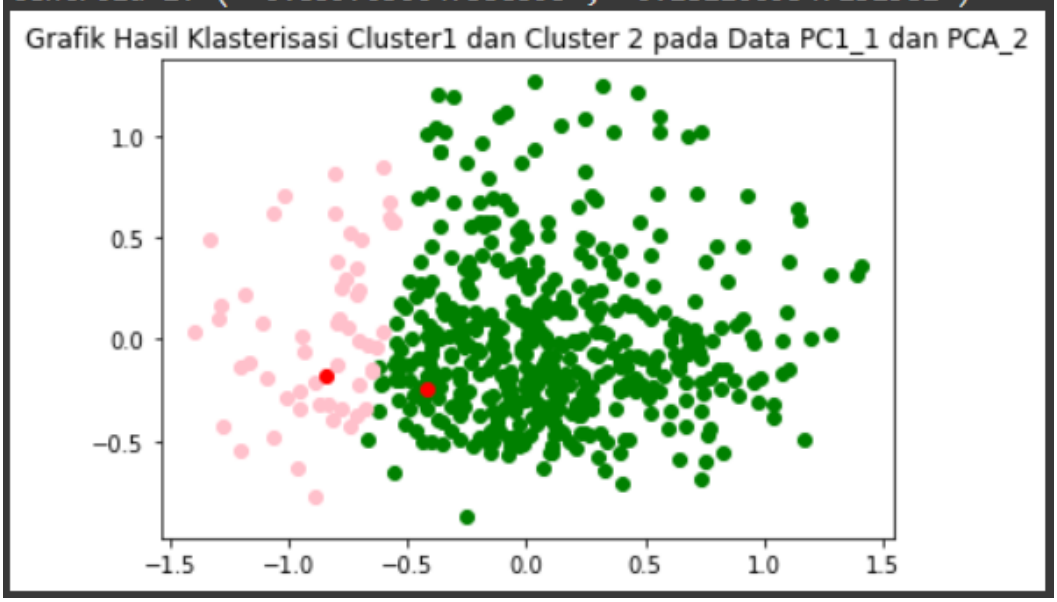
```
if(mc1x1 == c1x1 and mc1x2 == c1x2 and mc2x1 == c2x1 and mc2x2 == c2x2):
  break
else:
  c1x1 = mc1x1
  c1x2 = mc1x2

  c2x1 = mc2x1
  c2x2 = mc2x2



j = j + 1

pc1 = pd.DataFrame(c1)
print(pc1)
```

BAB 4
HASIL DAN ANALISIS

Hasil dari clusterisasi menggunakan metode K-Means:

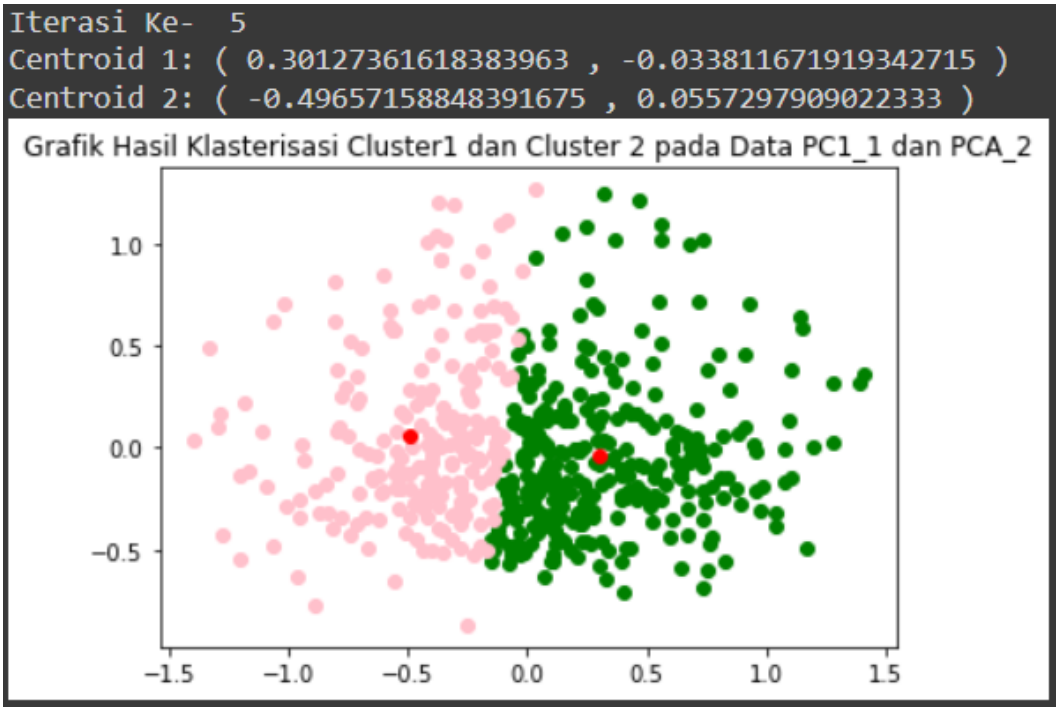
Iterasi ke-1:

```
Iterasi Ke-  1
Centroid 1: ( -0.4192691697987761 , -0.2440350795947824 )
Centroid 2: ( -0.8397630647338595 , -0.1811009347131582 )
```



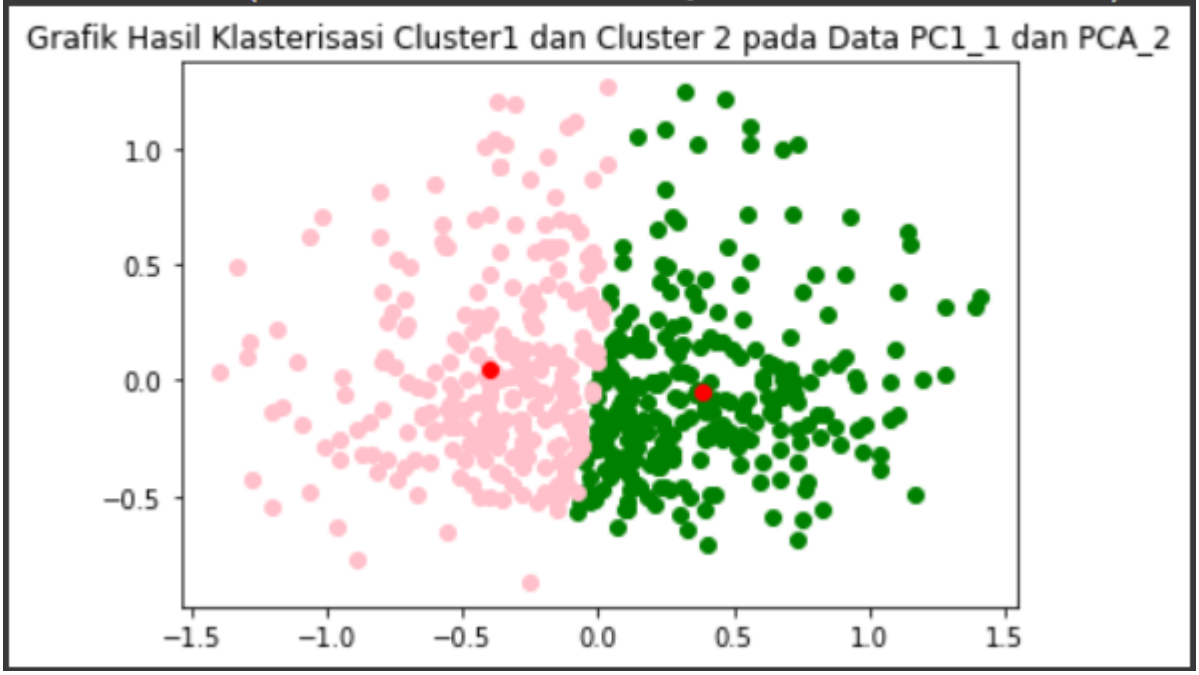
Iterasi Ke-5:

```
Iterasi Ke-  5
Centroid 1: ( 0.30127361618383963 , -0.033811671919342715 )
Centroid 2: ( -0.49657158848391675 , 0.0557297909022333 )
```



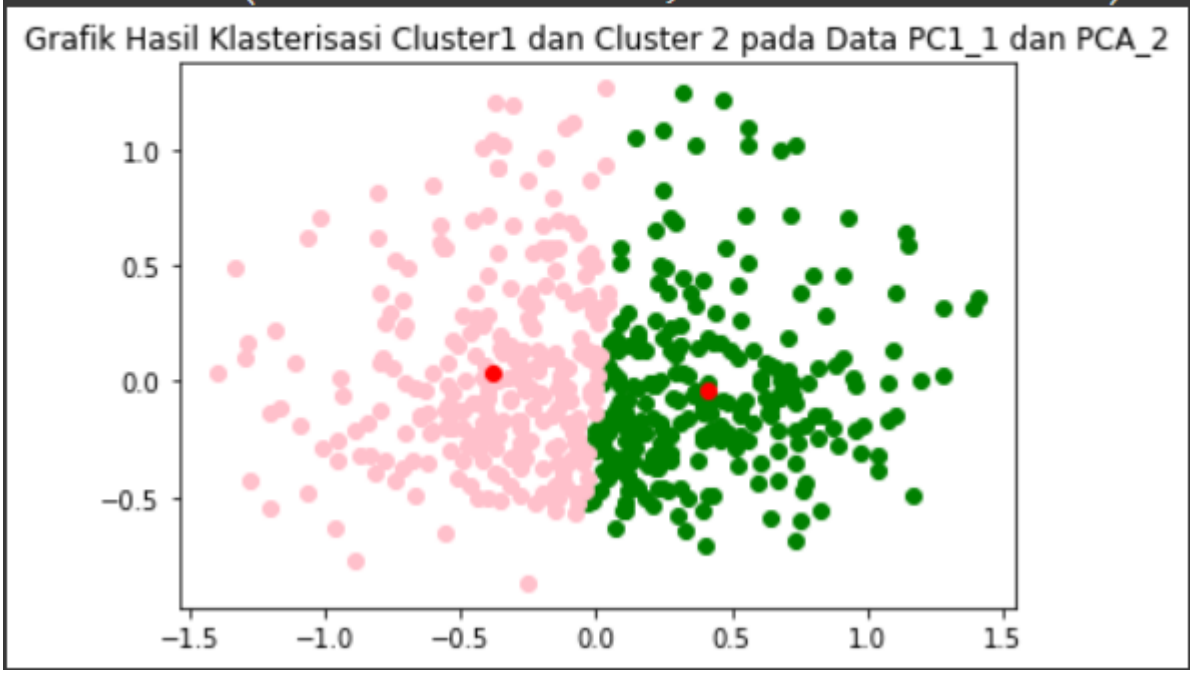
Iterasi Ke-10:

```
Iterasi Ke-  10
Centroid 1: ( 0.38142047128806766 , -0.047608132265516384 )
Centroid 2: ( -0.40376932702760293 , 0.050397671265449 05 )
```



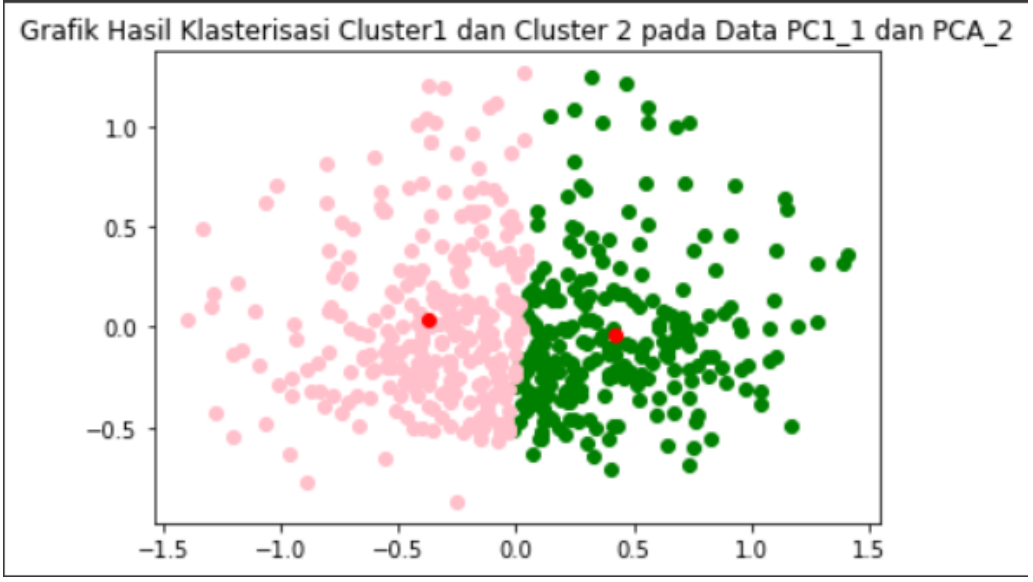
Iterasi Ke-15:

```
Iterasi Ke-  15
Centroid 1: ( 0.40807292952872243 , -0.043900671288055 33 )
Centroid 2: ( -0.3796722494516319 , 0.0408453132130625 34 )
```



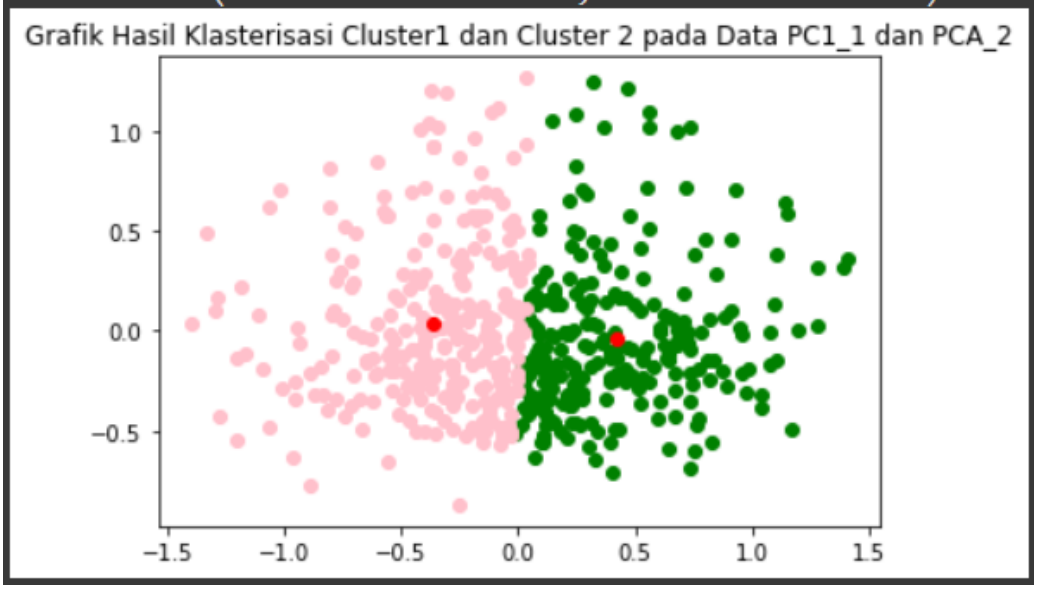
Iterasi Ke-20:

```
Iterasi Ke-  20
Centroid 1: ( 0.41837797434732804 , -0.03673467120519381 )
Centroid 2: ( -0.37189153275318054 , 0.03265304107128345 )
```



Iterasi Ke-22:

```
Iterasi Ke-  22
Centroid 1: ( 0.42328131338302755 , -0.03639504830473193 )
Centroid 2: ( -0.3677444034710701 , 0.0316198114704232 )
```



Proses berhenti pada iterasi ke-22 data terbagi menjadi 2 cluster, diindikasikan berwarna pink dan hijau, pada awal iterasi terlihat cluster berwarna hijau lebih mendominasi, kemudian perlahan-lahan berkurang dan hasil akhir memperlihatkan data terbagi hampir rata antara cluster pink dengan cluster hijau, membuktikan bahwa proses clustering berjalan dan berhasil.

Lampiran:

Link Google Colab:
https://colab.research.google.com/drive/1GQ3_EunUzp9T_JVB-VMQYNbWTBnAvSMt?usp=sharing

Link Video Presentasi:

https://drive.google.com/drive/folders/1hPVnuTuv0FDS_65f_wEFaNUjlPkDJD9P?usp=sharing