Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

Laporan TUBES 1 Machine Learning

1. Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random as rd
```

Library yang saya gunakan untuk tubes ini adalah pandas untuk membuat tabel dan mengecek data dan sebagainya, lalu numpy untuk perhitungan, lalu matplotlib untuk membuat plot, lalu seaborn untuk membuat grafik, dan random untuk meng-generate angka random.

2. Dataset

```
salju_test = "/content/salju_test.csv"
salju_train = "/content/salju_train.csv"
```

Dataset yang saya gunakan adalah salju_train, karena memiliki jumlah row yang lebih banyak

```
df_train = pd.read_csv(salju_train)
df_train
```

	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAngin9am	ArahAngin3pm	KecepatanAngin9am	KecepatanAn
0	1	01/06/2014	C4	10.4	15.5	4.8	NaN	NaN	WSW	24.0	NaN	WSW	0.0	
1	2	15/07/2014	C10	9.0	17.0	8.0	2.6	7.4	NaN	NaN	SW	WNW	13.0	
2	3	16/02/2011	C46	18.2	32.0	0.0	NaN	NaN	ESE	44.0	SE	SE	15.0	
3	4	08/08/2012	C36	7.3	24.5	0.0	8.4	10.4	SSW	54.0	N	SW	13.0	
4	5	29/10/2016	C7	5.9	20.3	0.0	3.6	12.6	N	37.0	NNW	ESE	22.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
109090	109091	31/01/2009	C38	20.1	23.7	0.0	7.2	8.9	ESE	43.0	SE	ESE	24.0	
109091	109092	03/11/2010	C16	15.7	25.2	0.0	NaN	NaN	SSE	37.0	SSE	E	28.0	
109092	109093	11/11/2010	C17	7.5	20.4	1.6	NaN	NaN	NW	33.0	N	NW	4.0	
109093	109094	16/04/2012	C11	10.8	29.8	0.0	7.8	11.2	E	48.0	ESE	SE	13.0	
109094	109095	09/10/2011	C16	12.3	27.4	9.0	NaN	NaN	WNW	35.0	NNE	NE	11.0	

109095 rows × 24 columns

3. Preprocessing data

a. Mengecek nilai yang null

```
miss_data = df_train.isnull()
miss_data.head()
```

	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAngin9am	ArahAngin3pm	KecepatanAngin9am	KecepatanAngin3pm
0	False	False	False	False	False	False	True	True	False	False	True	False	False	False
1	False	False	False	False	False	False	False	False	True	True	False	False	False	False
2	False	False	False	False	False	False	True	True	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False

```
for column in miss_data.columns.values.tolist():
  print(column)
  print(miss_data[column].value_counts())
  print("")
```

Saya membuat dataframe dengan nama *miss_data* untuk mengetahui kolom yang memiliki nilai NaN. Kemudian saya menampilkan isi setiap kolom dari *miss_data* dan dihasilkan jumlah nilai NaN setiap kolom.

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

```
"""
"SuhuMin" : 1122 miss data
"SuhuMax" : 929 miss data
"Hujan" : 2431 miss data
"Penguapan" : 47024 miss data
"SinarMatahari" : 52379 miss data
"ArahAnginTerkencang" : 7744 miss data
"KecepatanAnginTerkecang" : 7696
"ArahAngin9am" : 7923 miss data
"ArahAngin3pm" : 3197 miss data
"KecepatanAngin9am" : 1353 miss data
"KecepetanAngin3pm" : 2303
"Kelembaban9am" : 2002
"Kelembaban3pm" : 3374
"Tekanan9am" : 11327
"Tekanan3pm" : 11308
"Awan9am" : 41844
"Awan3pm" : 44471
"Suhu9am" : 1340
"Suhu3pm" : 2698
"BersaljuHariIni" : 2431

"BersaljuBesok" : 2431 -> label

"""
```

Sehingga dihasilkan seperti gambar disamping untuk setiap kolom yang memiliki nilai NaN.

Untuk mengisi nilai NaN saya menggunakan 2 cara, yaitu mengisi setiap nilai NaN pada kolom numerik dengan rata-rata pada setiap kolom dan mengisi nilai NaN pada kolom non-numerik dengan modus dari kolom tersebut

```
avg_suhumin = df_train['SuhuMin'].astype('float').mean(axis=0)
df_train['SuhuMin'].replace(np.nan, avg_suhumin, inplace=True)

avg_suhumax = df_train['SuhuMax'].astype('float').mean(axis=0)
df_train['SuhuMax'].replace(np.nan, avg_suhumax, inplace=True)

avg_hujan = df_train['Hujan'].astype('float').mean(axis=0)
df_train['Hujan'].replace(np.nan, avg_hujan, inplace=True)

avg_penguapan = df_train['Penguapan'].astype('float').mean(axis=0)
df_train['Penguapan'].replace(np.nan, avg_penguapan, inplace=True)

avg_sinarmatahari = df_train['SinarMatahari'].astype('float').mean(axis=0)
df_train['SinarMatahari'].replace(np.nan, avg_sinarmatahari, inplace=True)

avg_kecepatananginterkencang = df_train['KecepatanAnginTerkencang'].astype('float').mean(axis=0)
df_train['KecepatanAnginTerkencang'].replace(np.nan, avg_kecepatananginterkencang, inplace=True)

avg_kecepatanangin9am = df_train['KecepatanAngin9am'].astype('float').mean(axis=0)
df_train['KecepatanAngin9am'].replace(np.nan, avg_kecepatanangin9am, inplace=True)

avg_kecepatanangin3pm = df_train['KecepatanAngin3pm'].astype('float').mean(axis=0)
df_train['KecepatanAngin3pm'].replace(np.nan, avg_kecepatanangin3pm, inplace=True)
```

```
df_train["ArahAnginTerkencang"].replace(np.nan, "W", inplace=True)
df_train['ArahAnginTerkencang'].describe()
```

```
count     109095
unique        16
top            W
freq       15235
Name: ArahAnginTerkencang, dtype: object
```

	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAngin9am	ArahAngin3pm	KecepatanAngin9am	KecepatanAngin3pm
0	1	01/06/2014	C4	10.4	15.5	4.8	5.46244	7.599527	WSW	24.000000	NaN	WSW	0.0	13.0
1	2	15/07/2014	C10	9.0	17.0	8.0	2.60000	7.400000	NaN	40.032002	SW	WNW	13.0	20.0
2	3	16/02/2011	C46	18.2	32.0	0.0	5.46244	7.599527	ESE	44.000000	SE	SE	15.0	26.0
3	4	08/08/2012	C36	7.3	24.5	0.0	8.40000	10.400000	SSW	54.000000	N	SW	13.0	19.0
4	5	29/10/2016	C7	5.9	20.3	0.0	3.60000	12.600000	N	37.000000	NNW	ESE	22.0	19.0

b. Mengubah kolom object numerik menjadi float

```
df_train[["SuhuMin", "SuhuMax", "Hujan", "Penguapan", "SinarMatahari"]] = df_train[["SuhuMin", "SuhuMax", "Hujan", "Penguapan", "SinarMatahari"]].astype("float")
df_train[["KecepatanAnginTerkencang"]] = df_train[["KecepatanAnginTerkencang"]].astype("float")
df_train[["KecepatanAngin9am", "KecepatanAngin3pm"]] = df_train[["KecepatanAngin9am", "KecepatanAngin3pm"]].astype("float")
df_train[["Kelembaban9am", "Kelembaban3pm"]] = df_train[["Kelembaban9am", "Kelembaban3pm"]].astype("float")
df_train[["Tekanan9am", "Tekanan3pm"]] = df_train[["Tekanan9am", "Tekanan3pm"]].astype("float")
df_train[["Awan9am", "Awan3pm"]] = df_train[["Awan9am", "Awan3pm"]].astype("float")
df_train[["Suhu9am", "Suhu3pm"]] = df_train[["Suhu9am", "Suhu3pm"]].astype("float")
```

Saya merubah kolom numerik dengan tipe data objek menjadi float agar nanti bisa di lakukan normalisasi

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

```
a columns (total 24 columns):
 Column                    Non-Null Count   Dtype
 ------                    --------------   -----
 id                        109095 non-null  int64
 Tanggal                   109095 non-null  object
 KodeLokasi                109095 non-null  object
 SuhuMin                   109095 non-null  float64
 SuhuMax                   109095 non-null  float64
 Hujan                     109095 non-null  float64
 Penguapan                 109095 non-null  float64
 SinarMatahari             109095 non-null  float64
 ArahAnginTerkencang       109095 non-null  object
 KecepatanAnginTerkencang  109095 non-null  float64
 ArahAngin9am              109095 non-null  object
 ArahAngin3pm              109095 non-null  object
 KecepatanAngin9am         109095 non-null  float64
 KecepatanAngin3pm         109095 non-null  float64
 Kelembaban9am             109095 non-null  float64
 Kelembaban3pm             109095 non-null  float64
 Tekanan9am                109095 non-null  float64
 Tekanan3pm                109095 non-null  float64
 Awan9am                   109095 non-null  float64
 Awan3pm                   109095 non-null  float64
 Suhu9am                   109095 non-null  float64
 Suhu3pm                   109095 non-null  float64
```

c. Normalisasi

```
df_train['Tekanan9am'] = df_train['Tekanan9am']/df_train['Tekanan9am'].max()
df_train['Tekanan3pm'] = df_train['Tekanan3pm']/df_train['Tekanan3pm'].max()
```

Saya hanya melakukan normalisasi pada kolom *Tekanan9am* dan *Tekanan3pm* saja, karena memiliki nilai ribuan.

Sebelum:

Tekanan9am	Tekanan3pm
1020.10000	1018.500000
1015.20000	1014.600000
1017.64708	1015.253117
1019.20000	1016.900000
1019.70000	1014.700000

Sesudah:

Tekanan9am	Tekanan3pm
0.979923	0.979704
0.975216	0.975952
0.977567	0.976581
0.979059	0.978165
0.979539	0.976048

d. Mengubah tipe data objek yang tersisa menjadi category

```
object_column = df_train.select_dtypes(['object']).columns

df_train[object_column]=df_train[object_column].apply(lambda x: x.astype('category'))
```

```
df_train['Tanggal'] = df_train['Tanggal'].cat.codes
df_train['KodeLokasi'] = df_train['KodeLokasi'].cat.codes
df_train['ArahAnginTerkencang'] = df_train['ArahAnginTerkencang'].cat.codes
df_train['ArahAngin9am'] = df_train['ArahAngin9am'].cat.codes
df_train['ArahAngin3pm'] = df_train['ArahAngin3pm'].cat.codes
df_train['BersaljuHariIni'] = df_train['BersaljuHariIni'].cat.codes
df_train['BersaljuBesok'] = df_train['BersaljuBesok'].cat.codes
```

Sisa kolom dengan tipe data objek saya ubah menjadi category kemudian diubah menjadi category berupa numerik agar bisa dilakukan scaling.

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 109095 entries, 0 to 109094
Data columns (total 24 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   id                        109095 non-null  int64
 1   Tanggal                   109095 non-null  category
 2   KodeLokasi                109095 non-null  category
 3   SuhuMin                   109095 non-null  float64
 4   SuhuMax                   109095 non-null  float64
 5   Hujan                     109095 non-null  float64
 6   Penguapan                 109095 non-null  float64
 7   SinarMatahari             109095 non-null  float64
 8   ArahAnginTerkencang       109095 non-null  category
 9   KecepatanAnginTerkencang  109095 non-null  float64
 10  ArahAngin9am              109095 non-null  category
 11  ArahAngin3pm              109095 non-null  category
 12  KecepatanAngin9am         109095 non-null  float64
 13  KecepatanAngin3pm         109095 non-null  float64
 14  Kelembaban9am             109095 non-null  float64
 15  Kelembaban3pm             109095 non-null  float64
 16  Tekanan9am                109095 non-null  float64
 17  Tekanan3pm                109095 non-null  float64
 18  Awan9am                   109095 non-null  float64
 19  Awan3pm                   109095 non-null  float64
 20  Suhu9am                   109095 non-null  float64
 21  Suhu3pm                   109095 non-null  float64
 22  BersaljuHariIni           109095 non-null  category
 23  BersaljuBesok             109095 non-null  category
```

	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAngin9am	ArahAngin3pm	KecepatanAngin9am	KecepatanAngin3pm
0	1	53	33	10.4	15.5	4.8	5.46244	7.599527	15	24.000000	3	15	0.0	13.0
1	2	1632	1	9.0	17.0	8.0	2.60000	7.400000	13	40.032002	12	14	13.0	20.0
2	3	1693	40	18.2	32.0	0.0	5.46244	7.599527	2	44.000000	9	9	15.0	26.0
3	4	857	29	7.3	24.5	0.0	8.40000	10.400000	11	54.000000	3	12	13.0	19.0
4	5	3226	46	5.9	20.3	0.0	3.60000	12.600000	3	37.000000	6	2	22.0	19.0

e. Scaling

```
from sklearn.preprocessing import MinMaxScaler
normalisasi = MinMaxScaler()
df_train[df_train.columns] = normalisasi.fit_transform(df_train[df_train.columns])
df train.head()
```

Disini saya menggunakan library preprocessing untuk melakukan min-max-scaler untuk mengubah nilai setiap kolom menjadi rentang 0 sampai 1

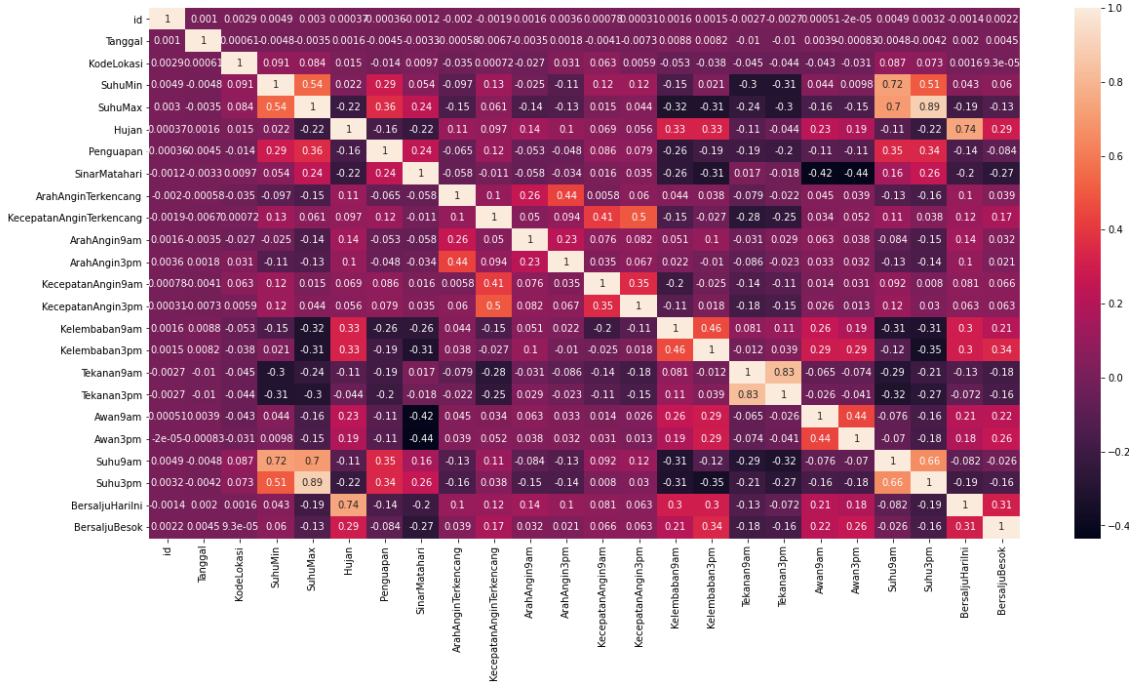
	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAngin9am	ArahAngin3pm	KecepatanAngin9am
0	0.000000	0.015529	0.687500	0.445755	0.389635	0.012938	0.037672	0.531435	1.000000	0.132812	0.2	1.000000	0.000000
1	0.000009	0.478172	0.020833	0.412736	0.418426	0.021563	0.017931	0.517483	0.866667	0.258063	0.8	0.933333	0.100000
2	0.000018	0.496045	0.833333	0.629717	0.706334	0.000000	0.037672	0.531435	0.133333	0.289062	0.6	0.600000	0.115385
3	0.000027	0.251099	0.604167	0.372642	0.562380	0.000000	0.057931	0.727273	0.733333	0.367188	0.2	0.800000	0.100000
4	0.000037	0.945209	0.958333	0.339623	0.481766	0.000000	0.024828	0.881119	0.200000	0.234375	0.4	0.133333	0.169231

4. Feature Selection

```
plt.figure(figsize=(20,10))
cor = df_train.corr(method="kendall")
sns.heatmap(cor, annot=True)
plt.show()
```

Pada feature selection saya membuat figure korelasi dengan heatmap menggunakan metode kendall.

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153



Dari figure diatas dapat didapati bahwa kolom ke-5 yaitu Hujan dan kolom ke-15 yaitu Kelembaban3pm adalah 2 kolom yang sangat berpengaruh pada label

Kemudian saya memasukkan 2 kolom tersebut kedalam dataframe baru dengan nama data.

```
data = df_train.iloc[:, [5,15]].values
```

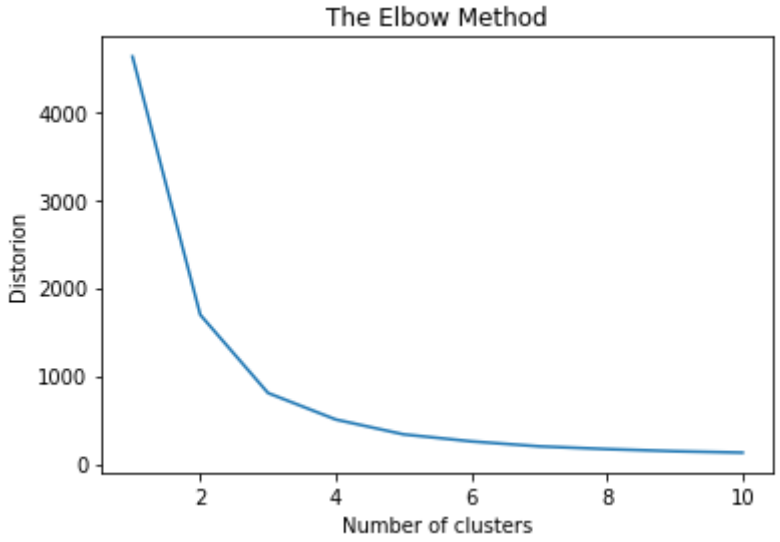
5. Penentuan nilai k

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 50)
    kmeans.fit(data)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('Distorion')
plt.show()
```

Untuk mendapatkan nilai k yang terbaik saya menggunakan elbow method.

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153



Dari hasil yang didapatkan diatas maka dapat disimpulkan bahwa nilai k yang terbaik adalah 3.

6. Clustering

a.

```
#Berdasarkan Elbow Method diatas maka nilai k=3
k = 3

#Jumlah iterasi
n_iterasi = 100

#Membuat array kosong untuk centroid
centroid=np.array([]).reshape(data.shape[1],0)

#Membuat random centroid sejumlah nilai k
for i in range(k):
  rand=rd.randint(0,data.shape[0]-1)
  centroid=np.c_[centroid,data[rand]]
```

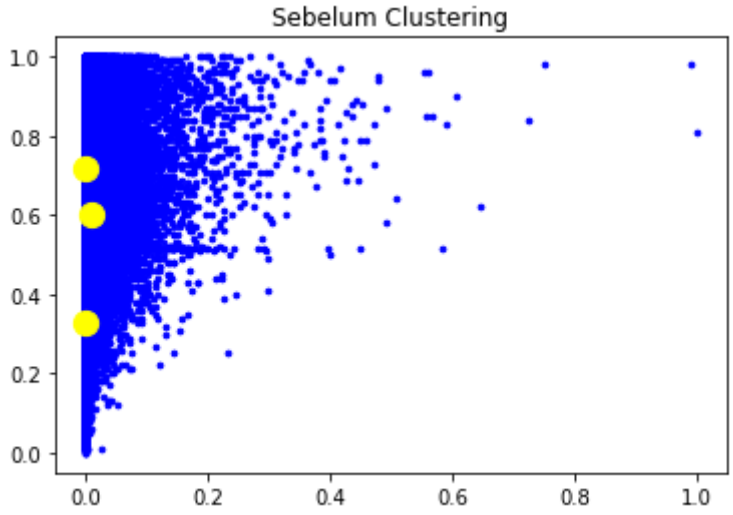
Pertama-taman saya memasukkan nilai k yaitu 3. Kemudian nilai iterasi yaitu 100, karena berdasarkan sumber yang saya pakai 100 sudah cukup untuk mewakilkan. Kemudian saya membuat centroid dengan posisi acak dengan jumlah centroid sama dengan k.

b.

```
plt.scatter(data[:,0],data[:,1], c='blue', s=7)
plt.scatter(centroid[0,:], centroid[1,:], c='yellow', label='Centroid', s=150)
plt.title('Sebelum Clustering')
plt.legend
plt.show()
```

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

Lalu saya membuat plot untuk setiap data dan centroid yang telah dibuat sebelumnya (plot belum dilakukan clustering)



c.

```
output = {}

#Melakukan perulangan sebanyak nilai n_iterasi
for i in range(n_iterasi):
  #Membuat array kosong untuk euclidian
  euclidian = np.array([]).reshape(data.shape[0],0)

  #Mencari jarak antar centroid
  for j in range(k):
    dist = np.sum((data-centroid[:,j])**2, axis=1)
    euclidian=np.c_[euclidian, dist]

  #Menyimpan jarak minimum dari hasil hitungan
  minimum = np.argmin(euclidian, axis=1)+1

  #Menghtung nilai mean untuk cluster yang terpisah
  cent = {}
  for j in range(k):
    cent[j+1]=np.array([]).reshape(2,0)

  #Menetapkan cluster ke poin tertentu
  for j in range(data.shape[0]):
    cent[minimum[j]]=np.c_[cent[minimum[j]],data[j]]

  for j in range(k):
    cent[j+1]=cent[j+1].T

  #Menghitung mean dan memperbaruinya
  for j in range(k):
    centroid[:,j]=np.mean(cent[j+1], axis=0)

  #Menyimpan hasil akhirnya pada output
  output=cent
```

Kemudian saya mencari jarak antara poin-poin dari setiap centroid dan disimpan pada variabel euclidian. Disini saya menggunakan Euclidian distance karena merupakan metode yang paling sering digunakan. Disini juga saya menyimpan jarak minimum pada variable minimum.

Kemudian setiap poin data dikelompokkan berdasarkan nilai minimum dan disimpan pada output. Kemudian juga dilakukan perhitungan mean untuk setiap cluster acak dan ditentukan sebagai centroid baru. Kemudia cent digunakan untuk menyimpan solusi iterasi tertentu.

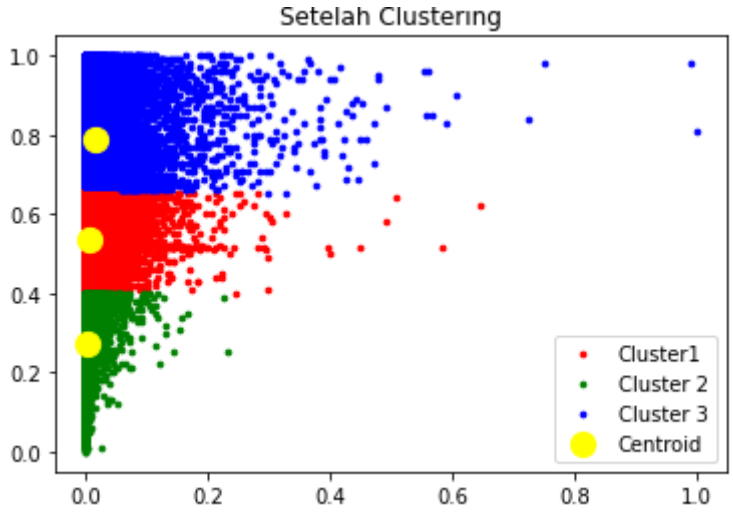
Tahapan tersebut dilakukan terus-menerus selama iterasi yang telah ditentukan

Nama : Andika Elang Dirgantara
Kelas : IF 42 07
NIM : 1301184153

d.

```
color=['red','green','blue']
labels=['Cluster1','Cluster 2','Cluster 3']
for i in range(k):
  plt.scatter(output[i+1][:,0], output[i+1][:,1], c = color[i], label = labels[i], s=7)
plt.scatter(centroid[0,:], centroid[1,:], c='yellow', label='Centroid', s=150)
plt.title('Setelah Clustering')
plt.legend()
plt.show()
```

Setelah dilakukan clustering maka tampilan nya akan menjadi seperti berikut.



Link youtube : https://youtu.be/6Ad2IsaQzc8