PENERAPAN ALGORITMA K-MEANS CLUSTERING UNTUK MENGELOMPOKKAN DATA PELANGGAN DI SEBUAH DEALER KENDARAAN

Oleh :

Deny Ahmad Sofyan

1301194274

IF-43-10

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

2021

Kata Pengantar

Pertama-tama penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa atas kasih karunia dan rahmat-Nya sehingga penulis dapat menyelesaikan laporan *K-Means* dengan tepat waktu dalam memenuhi penilaian Mata Kuliah Pembelajaran Mesin. Meskipun banyak kesulitan dalam pembuatan laporan ini, namun berkat usaha dan berkah-Nya saya bisa menyelesaikannya tepat waktu.

Selanjutnya, saya ucapkan terima kasih kepada Dr. Gamma Kosala, S.Si selaku Dosen Pembelajaran Mesin di kelas IF-43-10 atas kesempatan dan ilmu yang diberikan sehingga saya dapat menyelesaikan Tugas 1 Programming *K-Means* , serta tidak lupa atas bantuan para asisten dosen juga sehingga saya dapat menyelesaikan tugas Pembelajaran Mesin ini ini dengan semaksimal mungkin dan tepat waktu.

Laporan ini masih jauh dari kesempurnaan, maka dari itu saya mengharapkan adanya kritik dan feedback membangun terhadap apa yang saya kerjakan.

BAB I

Pendahuluan

A. Latar Belakang

Pada sebuah kanal penjualan kendaraan atau biasa kita sebut dealer ingin membuat suatu system yang bisa melihat pelanggan tersebut tertarik untuk membeli kendaraan baru atau tidak dengan mengabaikan data ketertarikan pelanggan. Dealer pun sudah mempunyai data yang berisi umur, Jenis kelamin, Sim, kode daerah, sudah asuransi, umur kendaraan, kendaraan rusak, premi, kanal penjualan, lama berlangganan dan tertarik.

Pada permasalahan dataset yang diberikan oleh dealer ini, kita diharapkan bisa mengelompokkan data pembeli apakah pembeli tertarik untuk membeli kendaraan baru atau tidak, tanpa memperhatikan sebuah label ketertarikannya. Sehingga, output yang diharapkan kita dapat memunculkan sebuah *clustering* terhadap data-data tersebut mengenai apakah customer tersebut tertarik atau tidaknya membeli kendaraan baru.

B. Manfaat dan Tujuan

Adapun manfaat dalam pembuatan program ini untuk dapat meninjau atau meng*cluster* data dari kendaraan pada dealer, sehingga pada data tersebut kita dapat melihat kelompok dari data pada dealer untuk dilihat sebagai acuan dalam ketertarikan pelanggan

BAB II

Landasan Teori

Pada bab ini akan dipaparkan mengenai landasan teori yang mendukung pembuatan tugas ini. Dalam bab ini akan dijelaskan beberapa teori-teori yang digunakan baik dalam pembuatan program atau laporan ini.

A. K-Means Algorithm

K-Means merupakan sebuah algoritma pemrograman untuk melakukan cluster *n objek* berdasarkan sebuah attribute k partisi, dimana k pasti di bawah n. Algoritma ini memiliki langkah langkah sebagai berikut :

1. Menentukan Jumlah Cluster.
2. Menentukan Nilai Centroidnya.
3. Menghitung jarak antara titik centroid, biasanya menggunakan euclidean distance.
4. Lalu melakukan pengelompokkan object.
5. Dan terakhir kembali ke tahap 2, lalu melakukan perulangan sehingga nilai centroid yang dihasilkan tetap.

B. Clustering

Clustering merupakan pengelompokkan. Dalam hal ini, clustering di K-Means berfungsi untuk mengelompokkan beberapa objek data yang memiliki kemiripan satu sama lainnya. Apabila sebuah objek memiliki kemiripan, maka akan dikelompokkan dalam satu jenis. Algoritma clustering akan mencoba untuk mengelompokkan atau membagikan seluruh data yang ada menjadi kelompok-kelompok yang memiliki kemiripan tersendiri. Dalam algoritma K-means, akan memanfaatkan perhitungan jarak, salah satunya dengan memanfaatkan *euclidean distance* dengan menghitung jarak. Jika jarak nya dianggap dekat, maka akan dikelompokkan menjadi satu jenis.

BAB III

Implementasi

A. LIBRARY

Pada program K-Means kali ini saya menggunakan beberapa library untuk membatu menyelesaikan beberapa permasalahan pada program kali ini agar lebih simple dan juga mudah untuk di implementasikan pada sebuah kodingan, disini saya menggunakan library K-Means namun bukan untuk pada Permodelan melainkan untuk Evaluasi, library yang saya gunakan adalah sebagai berikut :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random
from math import ceil
import random as rd
from copy import deepcopy
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler , StandardScaler
from sklearn.decomposition import PCA
```

B. IMPORT DATASET

Persiapan data sebagai bahan olahan proses Clustering dengan menggunakan dataset kendaraan_train bertipe CSV

IMPORT DATASET

Persiapan data sebagai bahan olahan proses Clustering dengan menggunakan dataset kendaraan_train bertipe CSV

```
[ ] url = 'https://raw.githubusercontent.com/Denyrcrc/Tubes-1-ML/main/kendaraan_train.csv'
    dataset_train = pd.read_csv(url)
```

```
[ ] dataset_train
```

	id	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Pen
0	1	Wanita	30.0	1.0	33.0	1.0	< 1 Tahun	Tidak	28029.0	
1	2	Pria	48.0	1.0	39.0	0.0	> 2 Tahun	Pernah	25800.0	
2	3	NaN	21.0	1.0	46.0	1.0	< 1 Tahun	Tidak	32733.0	
3	4	Wanita	58.0	1.0	48.0	0.0	1-2 Tahun	Tidak	2630.0	
4	5	Pria	50.0	1.0	35.0	0.0	> 2 Tahun	NaN	34857.0	
...	...	...	...	...	...	...	...	...	...	...

C. PREPROCESSING

Persiapan dataset untuk langkah selanjutnya, persiapan seperti menghapus kolom yang tidak penting mencari missing value (null) dan mengatasi missing value

a) Penggunaan kolom penting

Disini di drop beberapa kolom yang dirasa kurang penting seperti ID karena sama dengan row CSV, SIM dan Sudah_Asuransi karena kurang variatif dan juga Tertarik dikarenakan itu adalah hasil dari sebuah proses

Penggunaan kolom penting

Menghapus kolom yang tidak penting terhadap proses Clustering. Disini kolom yang di hapus adalah:

- id, dikarenakan sama dengan row CSV
- SIM dan Sudah_Asuransi, dikarenakan mempunyai value yang kurang variatif dan kurang berkorelasi
- tertarik, dikarenakan kolom tersebut adalah hasil dari sebuah proses

```
[ ] dataset = dataset_train.drop(['id', 'SIM', 'Sudah_Asuransi','Tertarik'], axis=1)
```

```
[ ] dataset
```

	Jenis_Kelamin	Umur	Kode_Daerah	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Penjualan	Lama_Berlangganan
0	Wanita	30.0	33.0	< 1 Tahun	Tidak	28029.0	152.0	97.0
1	Pria	48.0	39.0	> 2 Tahun	Pernah	25800.0	29.0	158.0
2	NaN	21.0	46.0	< 1 Tahun	Tidak	32733.0	160.0	119.0
3	Wanita	58.0	48.0	1-2 Tahun	Tidak	2630.0	124.0	63.0
4	Pria	50.0	35.0	> 2 Tahun	NaN	34857.0	88.0	194.0

b) Cek missing value

Cek di semua column ada berapa row yang berisikan null, bisa dilihat pada masing-masing column masi terdapat beberapa missing value yang banyak yang dapat membuat noise data yang tinggi

+ Kode + Teks

Cek missing value

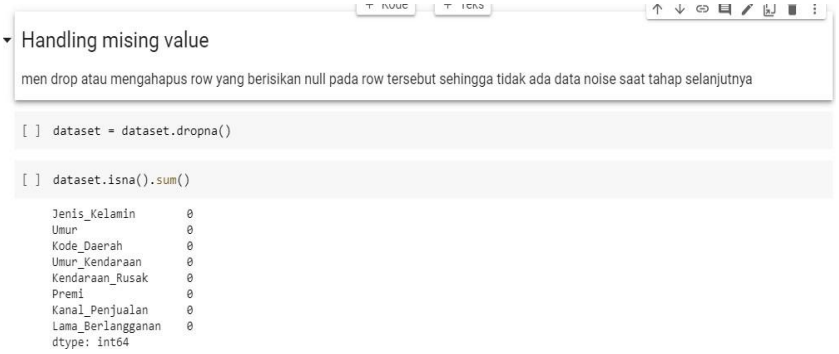
cek dan menampilkan value null dari semua kolom pada file CSV

```
[ ] dataset.isna().sum()
```

```
Jenis_Kelamin       14440
Umur                14214
Kode_Daerah         14306
Umur_Kendaraan      14275
Kendaraan_Rusak     14188
Premi               14569
Kanal_Penjualan     14299
Lama_Berlangganan   13992
dtype: int64
```

c) Handling missing value

Mengatasi value yang kosong setelah di cek sebelumnya dengan cara mendrop row yang ada pada colum tersebut sehingga dapat mengurangi noise pada data dan setelah di handling dapat dilihat bahwa pada dataset sudah tidak ada lagi missing value



Handling mising value

men drop atau mengahapus row yang berisikan null pada row tersebut sehingga tidak ada data noise saat tahap selanjutnya

```
[ ] dataset = dataset.dropna()
```

```
[ ] dataset.isna().sum()
```

```
Jenis_Kelamin       0
Umur                0
Kode_Daerah         0
Umur_Kendaraan      0
Kendaraan_Rusak     0
Premi               0
Kanal_Penjualan     0
Lama_Berlangganan   0
dtype: int64
```

d) Handling value non-numerical

menghapus value dari kolom yang berisi objek agar tidak adanya noise dan juga mengatasi kurang nya variasi data karena berisi value 0 sampai 1/2 membuat korelasi kurang maksimal dalam menentukan scalling

Handling value non-numerical

menghapus value dari kolom yang berisi objek agar tidak adanya noise dan juga mengatasi kurang nya variasi data karena berisi value 0 sampai 1/2 membuat korelasi kurang maksimal dalam menentukan scalling

```
[ ] filteredColumns = dataset.dtypes[dataset.dtypes == np.object]
    listOfColumnNames = list(filteredColumns.index)
    print(listOfColumnNames)
```

```
['Jenis_Kelamin', 'Umur_Kendaraan', 'Kendaraan_Rusak']
```

```
[ ] for column in dataset:
      if dataset.dtypes[column] == np.object:
        newdataset = dataset.drop(listOfColumnNames, axis=1)
        print('Data type of column', column, 'is dropped')
```

```
Data type of column Jenis_Kelamin is dropped
Data type of column Umur_Kendaraan is dropped
Data type of column Kendaraan_Rusak is dropped
```

e) Cek Outliner

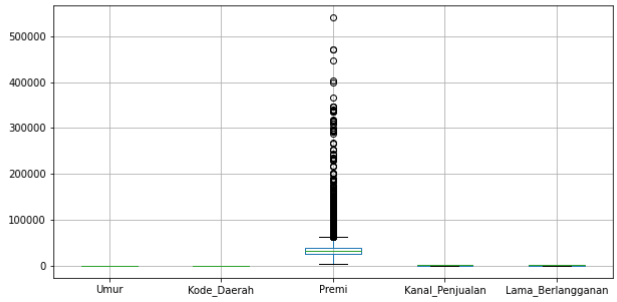
Meninjau data yang tidak wajar untuk di atasi agar noise tidak begitu banyak dan juga mengganggu korelasi antar data

Check outliner

men check apakah ada data yang tidak wajar (tidak umum)

```
newdataset.boxplot(figsize=[10,5])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd395d3450>
```



f) Handling Outliner

Mengatasi outliner data yang tidak wajar yang sudah di cek sebelumnya kebanyakan ada pada premi dan kita handling dengan mendekatkan data pada data terdekatnya

handling outliner

mengatasi data yang tidak wajar dengan menggunakan metode interquartile data

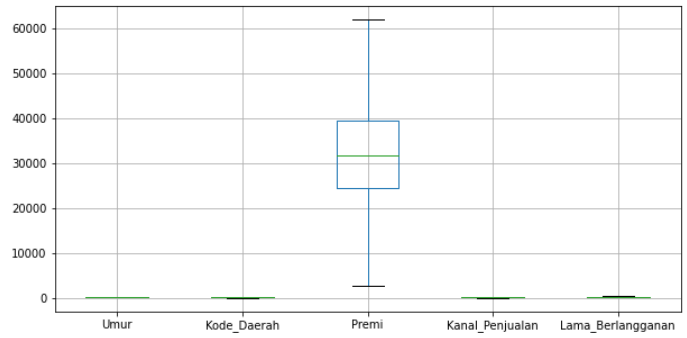
```
[40] def interquartile(newdataset,x):
       q1 = (newdataset[x]).quantile(0.25)
       q3 = (newdataset[x]).quantile(0.75)
       iqr = q3 - q1 #range q3 - q1
       maximum = q3 + (1.5 *iqr)
       minimum = q1 - (1.5 *iqr)
       return maximum,minimum

     max,min = interquartile(newdataset,'Premi')
     print('max: ',max,' | min:',min)
```

```
max:  61777.5  | min: 1925.5
```

```
[41] def HandlingOutliner(newdata,x,max,min):
       above = (newdataset[x] > max)
       below = (newdataset[x] < min)
       print('above: ',above,' | below: ',below)
       newdataset[x] = newdataset[x].mask(above, max,axis=0)
       newdataset[x] = newdataset[x].mask(below, min,axis=0)
       return newdataset

     max,min = interquartile(newdataset,'Premi')
     newdataset = HandlingOutliner(newdataset,'Premi',max,min)
     boxplot = newdataset.boxplot(column=['Premi'])
```

```
[42] newdataset.boxplot(figsize=[10,5])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd355ef090>
```



g) Scalling

Scaling dengan menggunakan StandardScaler, agar persebaran data setiap kolomnya tidak terlalu jauh dan angkanya masih bervariasi

Scalling

Scaling dengan menggunakan StandardScaler, agar persebaran data setiap kolomnya tidak terlalu jauh dan angkanya masih bervariasi

```
[43] scale = StandardScaler()
     scale.fit(newdataset)
     temp =scale.transform(newdataset)
     newdatasetnew = pd.DataFrame(temp, index=newdataset.index, columns=newdataset.columns)
     newdatasetnew
```

	Umur	Kode_Daerah	Premi	Kanal_Penjualan	Lama_Berlangganan
0	-0.567639	0.495812	-0.134144	0.735019	-0.682865
1	0.591373	0.948087	-0.278290	-1.537184	0.044955
3	1.235269	1.626500	-1.776665	0.217770	-1.088534
5	-1.147145	0.646570	-0.476500	0.735019	0.200064
7	-1.018366	0.118916	-0.209159	0.735019	0.808569

h) Splitting

menggunakan PCA untuk mereduksi data menjadi 2 kolom untuk selanjutnya digunakan dalam clustering

Splitting

menggunakan PCA untuk mereduksi data menjadi 2 kolom untuk selanjutnya digunakan dalam clustering

```
[44] temp = MinMaxScaler().fit_transform(newdatasetnew)
     newdatasetnew1 = pd.DataFrame(temp, index=newdatasetnew.index, columns=newdatasetnew.columns)
```

```
pca = PCA(n_components=2)
datasetbaru = pca.fit_transform(newdatasetnew1)
datasetbaru = pd.DataFrame(data = datasetbaru, columns = ['x', 'y'])
datasetbaru
```

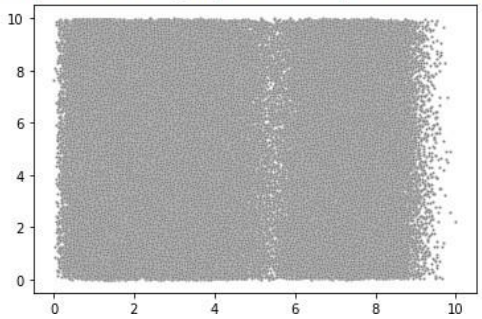
	x	y
0	-0.274482	-0.200329
1	0.513402	0.010798
2	0.037245	-0.320901
3	-0.353133	0.055236
4	-0.335913	0.233518

```
[46] arr = MinMaxScaler(feature_range=(0,10)).fit_transform(datasetbaru)
     print(arr)
```

```
[[1.55789241 3.01175782]
 [6.93978333 5.08874955]
 [3.68724061 1.82561801]
 ...
 [1.70122935 7.35034366]
 [4.3529984  8.97558765]
 [6.99449621 1.23580055]]
```

```
[47] x=arr[:,0]
     y=arr[:,1]
     sns.scatterplot(x, y, s=5, color='gray')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd355ef250>
```



D. PERMODELAN / CLUSTERING

Pemrosesan data utama dalam Clustering setelah sebelumnya sudah melewati tahap Preprosesing data

a) Cari jarak

Mencari jarak untuk centroid menggunakan metode Euclidian Distance

Find jarak

Mencari jarak untuk centroid menggunakan metode Euclidian Distance

```
[48] def Euclidian(a, b, ax=1):
         sub = a-b
         hasil = np.linalg.norm(sub, axis=ax)
         return hasil
```

b) Cari centroid

Find centroid

Mencari centroid untuk menampung data kendaraan dengan merandom nilai minimum dan maksimum yang terdapat pada array

```
[49] k = 5
     min = np.min(arr)
     max = np.max(arr)

     centroid1 = np.random.randint(min, max, size=k)
     centroid2 = np.random.randint(min, max, size=k)
     centroid = np.array(list(zip(centroid1, centroid2)))
     print(centroid)
```

```
[[8 1]
 [3 7]
 [3 3]
 [7 2]
 [8 5]]
```

c) K-Means

Melakukan centroid dengan menggunakan metode K-Means, membuat temparrcl untuk menampung hasil cluster dan temparrcen untuk menampung nilai centroid sebelumnya lalu array titik untuk menapung pengelompokan

K-Means

Melakukan centroid dengan menggunakan metode K-Means, membuat temparrcl untuk menampung hasil cluster dan temparrcen untuk menampung nilai centroid sebelumnya lalu array titik untuk menapung pengelompokan

```
[25] temparrcl = np.zeros(len(arr))
     temparrcen = np.zeros(centroid.shape)
     titik = []
     temp = []

     stop = Euclidian(centroid, temparrcen, None)
     print(stop)

     while stop != 0:
         for i in range(len(arr)):
             jarak = Euclidian(arr[i], centroid)
             cluster = np.argmin(jarak)
             temparrcl[i] = cluster
         temparrcen = deepcopy(centroid)

         for i in range(k):
             titik = [arr[j] for j in range(len(arr)) if temparrcl[j] == i]
             centroid[i] = np.mean(titik, axis=0)
             temp.append(temparrcl)

         stop = Euclidian(centroid, temparrcen, None)

         print(stop)
```

```
20.223748416156685
4.69041575982343
2.0
```

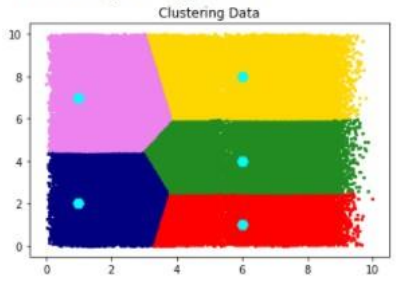
d) K-Means visualisasi

Menampilkan plotting dan centroid dengan asumsi awal k = 3 namun setelah melewati tahap evaluasi ternyata yang paing akurat k = 5

K-Means visualisasi

```
fig, ax = plt.subplots()
color = ['red', 'forestgreen', 'navy', 'gold', 'violet', 'turquoise', 'grey']

for i in range(k):
        titik = np.array([arr[j] for j in range(len(arr)) if temparrcl[j] == i])
        x = titik[:, 0]
        y = titik[:, 1]
        ax.scatter(x, y, s=5, c=color[i])

x_clus = centroid[:, 0]
y_clus = centroid[:, 1]
ax.scatter(x_clus, y_clus, marker='H', s=100, color='cyan')
plt.title("Clustering Data")
plt.show
```

```
<function matplotlib.pyplot.show>
```



E. EVALUASI

Evaluasi menggunakan metode Elbow Menthod untung meninjau atau mengomparasi dari clustering sebelummnya sehingga mendapatkan hasil yang maksimum dimungkinkan

a) Elbow menthod

metode untuk meninjau nilai k yang optimal dari hasil sebelumnnya, menggunakan library k-means untuk membantu jalannya proses pengecekan

Elbow menthod

metode untuk meninjau nilai k yang optimal dari hasil sebelumnnya

```
[53] datasetbaru
```

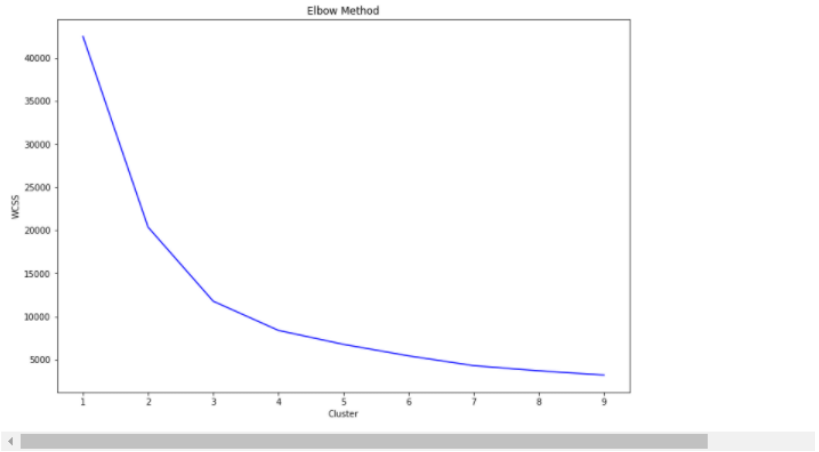
	x	y
0	-0.274482	-0.200329
1	0.513402	0.010798
2	0.037245	-0.320901
3	-0.353133	0.055236
4	-0.335913	0.233518
...	...	...
189694	-0.365450	0.222852
189695	-0.285205	-0.367017
189696	-0.253498	0.240691
189697	0.134709	0.405898
189698	0.521411	-0.380856

189699 rows × 2 columns

```
wcss=[]

k = range(1,10)

for i in k:
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=50)
    kmeans.fit(datasetbaru)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(12,8))
sns.lineplot(range(1,10), wcss, markers='o', color='blue')
plt.title('Elbow Method')
plt.xlabel('Cluster')
plt.ylabel('WCSS')
plt.show()
```

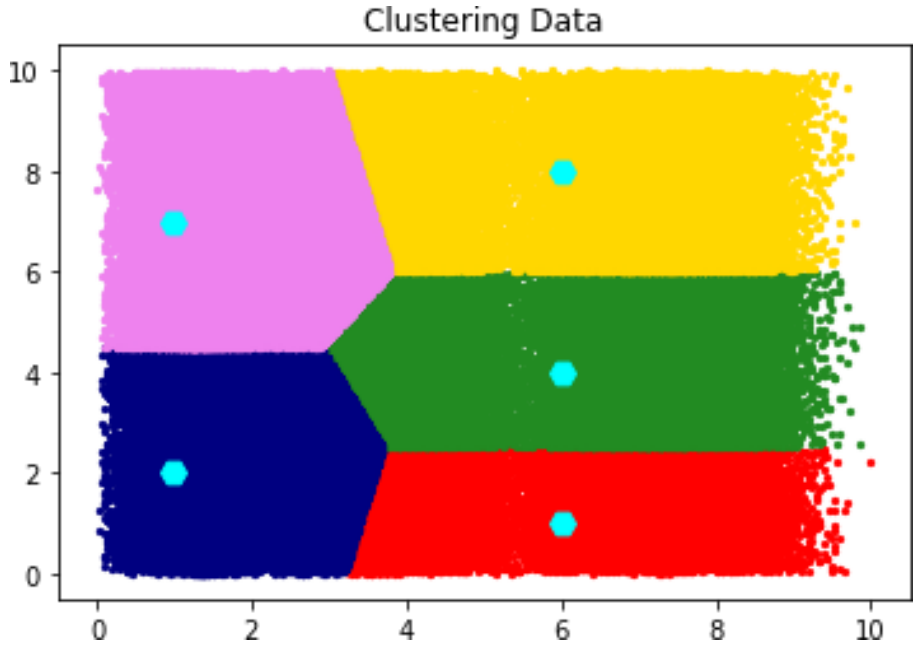
Terlihat bahwa pada cluster ke 5 merupakan sebuah kurva yang tidak begitu tajam dan juga stabil (4 – 5 - 6) sehingga sangat mungkin dilakukan k = 5 dikarenakan itu adalah nilai optimal dari k

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword ar
  FutureWarning
```



b) Hasil Clustering dan Evaluasi

Hasil dari Program adalah pot Clustering dengan nilai K = 5 yang disebar secara random, dan berikut hasilnya



BAB IV

Penutup

Kesimpulan

Berdasarkan hasil pengamatan dan hasil code yang dibuat diatas, penulis dapat menarik kesimpulan yang didapatkan sebagai berikut :

1. Pada saat melakukan Clustering kita harus melakukan Preprocessing data terlebih dahulu agar tidak adanya data yang tidak wajar dan juga noise pada data sehingga menggangu hasil dari Clustering

2. Nilai K-Means di dapat dari hasil korelasi, Scalling, dan Splitting untuk mencari kurva yang dirasa sangat baik untuk di terapkan pada Clustering K-Means

A. Penutup

Penulis mengucapkan banyak terimakasih kepada para pembaca yang sudah membaca laporan ini. Laporan ini juga masih jauh dari kata sempurna, dan tentu masih banyak hal yang kurang didalamnya. Penulis mengharapkan masukkan yang membangun demi pemahaman dalam memahami algoritma ini kedepannya. Sekian dari penulis, terimakasih.

Lampiran

Link Google Collab Code :
https://colab.research.google.com/drive/1Bm7qJ2usQCPOOm5BoUa2tQ6hFSraFZpt?usp=sharing#scrollTo=2NMY4v60qHvi

Link Video Presentasi :

https://www.youtube.com/watch?v=Flor_YvcGic

Daftar Pustaka

https://raharja.ac.id/2020/04/19/k-means-clustering/