LAPORAN TUGAS 2.1

MACHINE LEARNING

K-*Means Clustering*

Disusun Oleh :

Nadine Azhalia P. (1301154519)

Kelas : IF 39-01

PRODI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2018

I. Analisis Masalah

Dalam tugas 2.1 ini diberi dua jenis data pada *file* TrainsetTugas2.txt dan TestsetTugas2.txt yang tidak memiliki label, dimana data tersebut harus dikelompokan berdasarkan kesamaan karteristik yang dimiliki oleh setiap data. Untuk dapat mengelompokan data tidak berlabel digunakan pendekatan *Unsupervised Learning* dimana pembelajaran dilakukan secara tidak terbimbing.

Untuk memecahkan masalah tersebut digunakan algoritma K-*Means*. K-*Means* merupakan salah satu algoritma *Partional Clustering* yang menerapkan pendekatan *unsupervised learning*. Pada algoritma ini masukan yang diterima berupa data dan nilai *k* yang merepresentasikan jumlah *cluster* yang digunakan untuk mengelompokan data. Keluaran yang akan didapat berupa nilai centroid awal, centroid akhir, dan hasil *clustering*. Pada setiap *cluster* terdapat titik pusat atau ***centroid*** yang merepresentasikan *cluster* tersebut. Data yang dimasukan akan dikelompokan dalam *k* buah kelompok berdasarkan jarak minimum antar data ke salah satu *centroid*.

Data yang ada di TrainsetTugas2.txt digunakan untuk mendapatkan nilai *k* yang optimum, nilai yang optimum dapat dilihat dari hasil visualisasi, nilai error, dan *elbow method*. Setelah mendapat nilai *k* yang optimum, maka akan digunakan untuk *clustering* data di TestsetTugas2.txt.

II. Desain dan Implementasi

Algoritma K-*Means*

Pseudocode algotima K-*Means*:

Algorithm 8.1 Basic K-means algorithm.

1: Select K points as initial centroids.
2: **repeat**
3: Form K clusters by assigning each point to its closest centroid.
4: Recompute the centroid of each cluster.
5: **until** Centroids do not change.

Langkah-langkah yang dilakukan untuk membangun program K-*Means*:

1. Menentukan jumlah *cluster* (nilai *k*) dan membangkitkan bilangan acak sebagai inisialisasi centroid awal

   Berdasarkan hasil visualisasi dan nilai error yang didapat, maka nilai *k* optimum yang didapat sebesar k=5

```
# menentukan jumlah K dan me-random nilai centroid
k = 5
centroid = np.random.rand(k,2) * 36
print ("Nilai Centroid Awal :")
print(centroid)
```

2. Mengelompokan data menggunakan jarak dari data ke centroid yang terdekat. Untuk menghitung jarak menggunakan rumus euclide

```
# fungsi menghitung nilai euclide
def distc(a, b, ax=1):
    return np.linalg.norm(a - b, axis=ax)
```

```
cent_awal = np.zeros_like(centroid)
clusters = np.zeros(len(data))
error = 1

# perulangan akan dilakukan selama jarak centroid ke cent_awal
tidak sama dengan 0
while (error > 0):
    # perulangan untuk menentukan jarak euclid dari centroid awal
    ke setiap data
    for i in range(len(data)):
        euclide = distc(data[i,:], centroid, ax=1)
        cluster = np.argmin(euclide)
        clusters[i] = cluster
    cent_awal[:,:] = centroid[:,:]
```

3. Memperbarui nilai centroid awal

   Untuk mendapat nilai centroid baru didapat dengan cara mencari nilai rata-rata antara centroid awal ke setiap data.

```
# perulangan untuk menentukan centroid baru (didapat dari rata-rata
data per cluster)
    for i in range(k):
        points = []
        for j in range(len(data)):
            if clusters[j] == i:
                points.append(data[j])
        centroid[i] = np.mean(points, axis=0)
    error = distc(centroid, cent_awal, None)

print("Nilai Centroid Akhir :")
print(centroid)
```

4. Menentukan nilai Sum Squared Error (SSE)

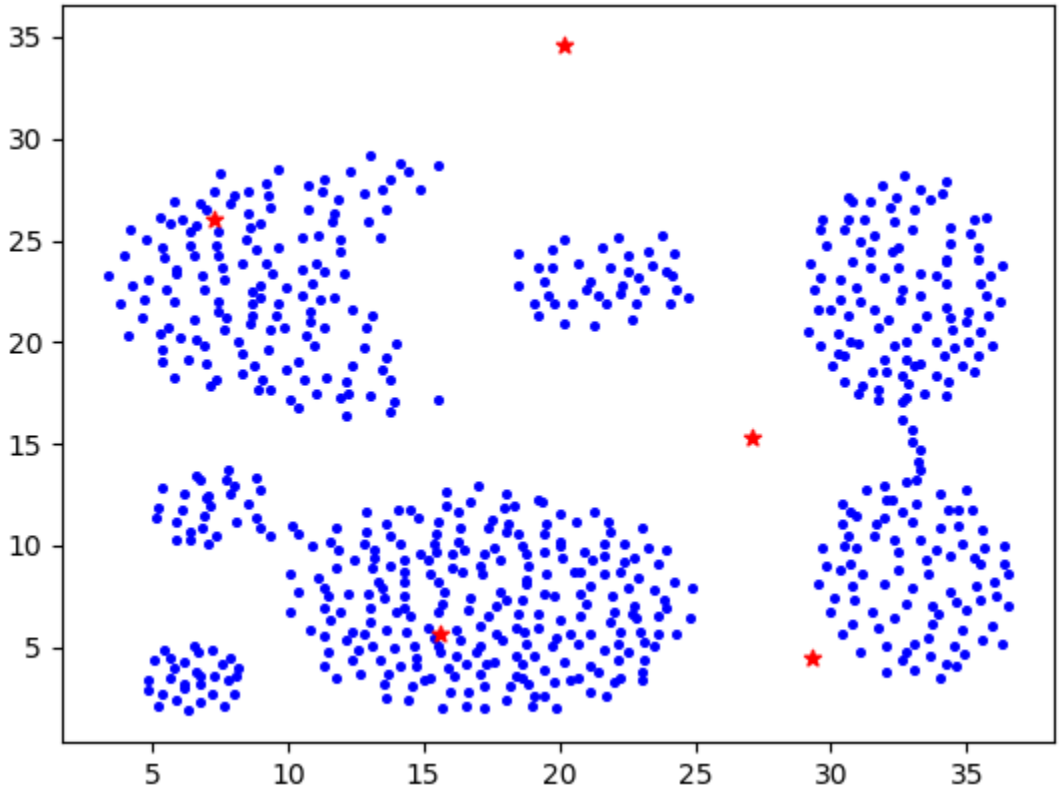
```
# fungsi menentukan nilai dari SSE
def sse (k, data, centroid):
    sse = 0
    for i in range(k):
        points = np.array([data[j] for j in range(len(data)) if
clusters[j] == i])
        sse += np.sum((centroid[i] - points) ** 2)
    return sse

print("Nilai SSE :")
print(sse(5,data,centroid))
```

III. Hasil Eksperimen

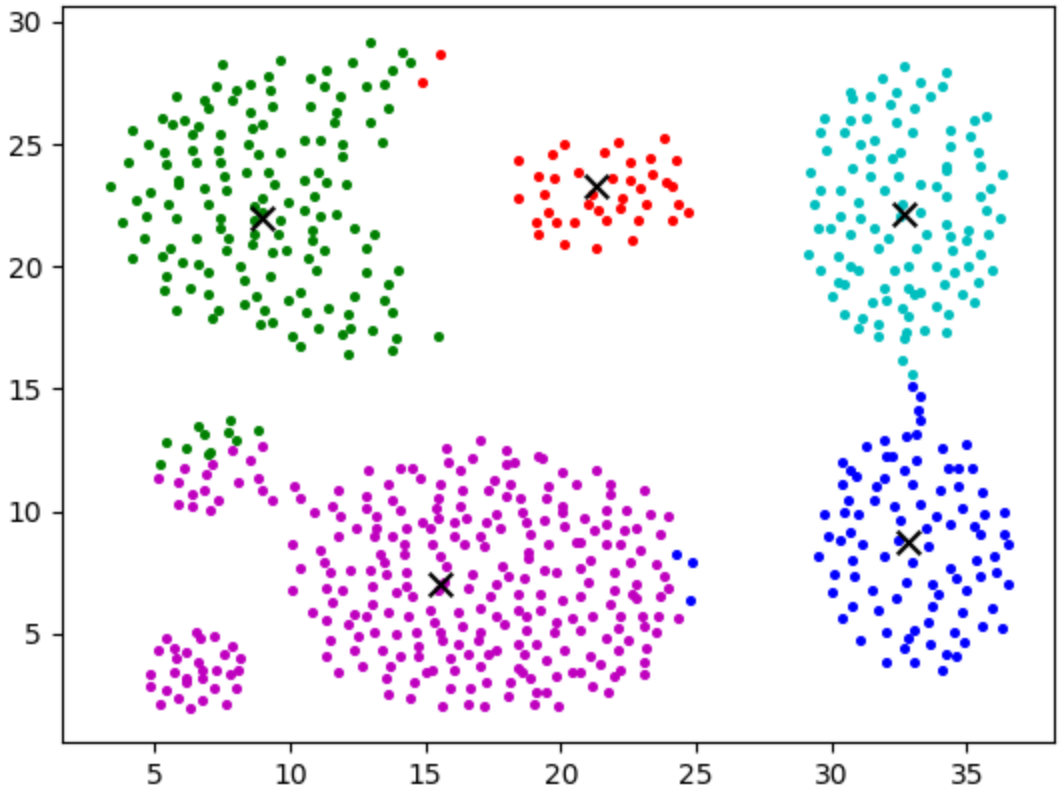
Train Set

- Visualisasi persebaran data untuk *file* TrainsetTugas2.txt dan letak dari *centroid* awal:



Gambar 1. Data TrainsetTugas2.txt dan letak Centroid Awal

- Hasil pengelompokan untuk data TrainsetTugas2.txt yang didapat dengan menggunakan inisialisasi k=5 adalah sebagai berikut:



Gambar 2. Hasil Pengelompokan Data TrainsetTugas2.txt

- Dengan nilai Centroid Awal, Centroid Akhir, dan Nilai SSE sebagai berikut:

Nilai Centroid Awal :

[[29.2668111 4.50248198]

[15.60967284 5.68241827]

[20.20221131 34.64150729]

[27.1087962 15.33388065]

[7.31313511 26.02826217]]

Nilai Centroid Akhir :

[[32.84479167 8.74322917]

[15.56631579 7.0222807]

[21.31097561 23.27560976]

[32.65272727 22.11454545]

[9.01185897 21.97179487]]

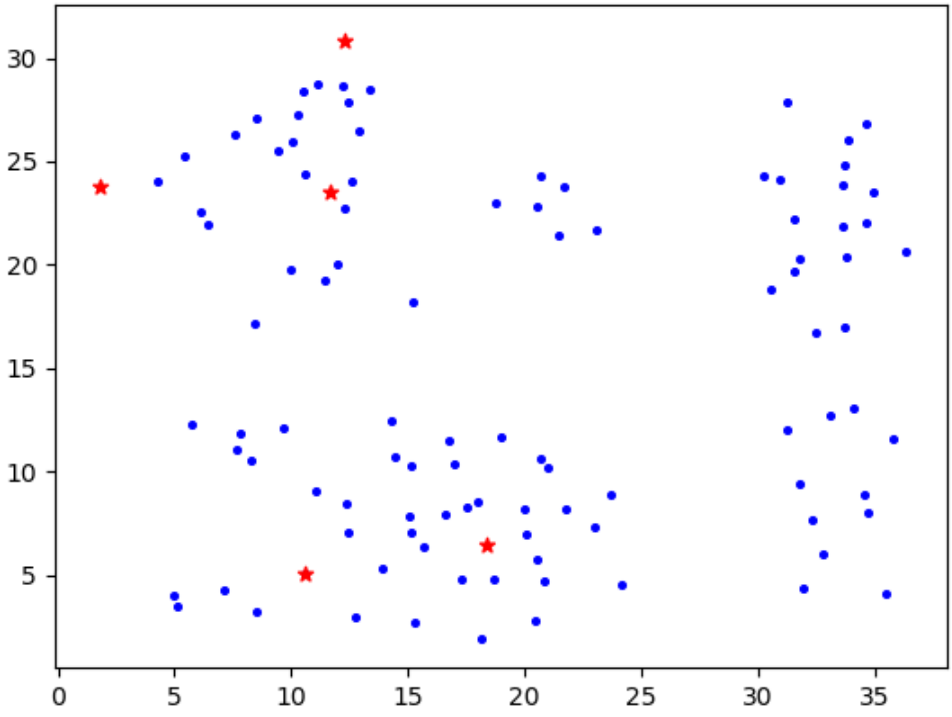
Nilai SSE :

17167.4986821

Test Set

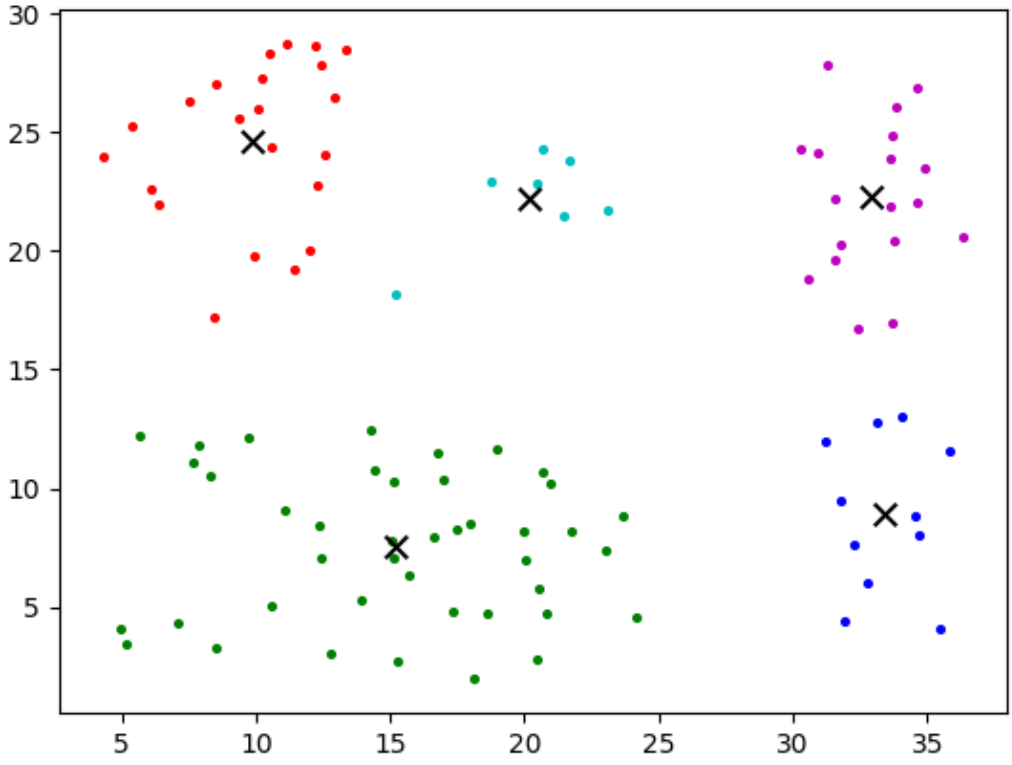
Setelah mendapat nilai *k* yang optimal berdasarkan hasil SSE, maka nilai *k* tersebut digunakan untuk mengelompokan data pada *file* TestsetTugas2.txt. Hasil dari pengelompokan pada data test:

- Visualisasi persebaran data untuk *file* TestsetTugas2.txt dan letak dari *centroid* awal:



Gambar 3. Data TestsetTugas2.txt dan letak Centroid Awal

- Hasil pengelompokan untuk data TrainsetTugas2.txt yang didapat dengan menggunakan inisialisasi k=5 adalah sebagai berikut:



Gambar 4. Hasil Pengelompokan Data TestsetTugas2.txt

- Dengan nilai Centroid Awal dan Centroid Akhir sebagai berikut:

Nilai Centroid Awal :

[[10.5622165 5.09668415]
 [12.30843297 30.84399141]
 [1.81708199 23.78625326]
 [18.33577922 6.47697002]
 [11.65665304 23.54323333]]

Nilai Centroid Akhir :

[[14.54102564 7.57692308]
 [32.95 22.28055556]
 [9.89545455 24.62727273]
 [31.32142857 8.46785714]
 [20.19285714 22.17857143]]

- Hasil Prediksi cluster untuk data test

Hasil Prediksi:

[3. 3. 3. 3. 3. 3. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.
2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 3. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4.
4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4.
4. 4. 4. 4. 4. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 4. 4. 4. 4. 4.]

Ket. :

Hasil prediksi juga dapat dilihat pada file HasilCluster.csv pada folder yang sama dengan laporan ini.