

K-Means Clustering

Langkah – Langkah Algoritma K-Means

Langkah-langkah algoritma K-means adalah sebagai berikut:

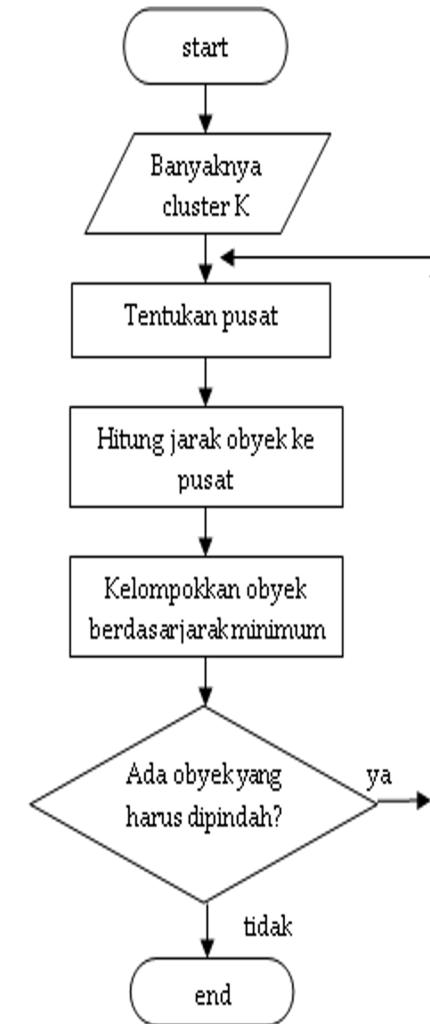
1. Tentukan nilai k sebagai jumlah klaster yang ingin dibentuk.
2. Bangkitkan k centroid (titik pusat klaster) awal secara random.
3. Hitung jarak setiap data ke masing-masing centroid menggunakan rumus korelasi antar dua objek yaitu Euclidean Distance dan kesamaan Cosine.
4. Kelompokkan setiap data berdasarkan jarak terdekat antara data dengan centroidnya.
5. Tentukan posisi centroid baru ($k C$) dengan cara menghitung nilai rata-rata dari data-data yang ada pada centroid yang sama.

$$C_k = \left(\frac{1}{n_k} \right) \sum d_i$$

Dimana k n adalah jumlah dokumen dalam cluster k dan i d adalah dokumen dalam cluster k .

6. Kembali ke langkah 3 jika posisi centroid baru dengan centroid lama tidak sama.

$$\text{Sim}(dx,dy) = \frac{\sum_{k=1}^n x_k \times y_k}{\sqrt{\sum_{k=1}^n x_k^2} \times \sqrt{\sum_{k=1}^n y_k^2}}$$



Gambar 1. Flowchart algoritma K-Means

Euclidean Distance

- **Cara Menghitung Euclidean Distance**
- **Euclidean distance** adalah perhitungan jarak dari 2 buah titik dalam Euclidean space. Euclidean space diperkenalkan oleh seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini biasanya diterapkan pada 2 dimensi dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.
 - **1 dimensi**
Semisal ingin menghitung jarak Euclidean 1 dimensi. Titik pertama adalah 4, titik kedua adalah -10. Caranya adalah kurangkan -10 dengan 4. sehingga menghasilkan -14. Cari nilai absolut dari nilai -14 dengan cara mempangkatkannya sehingga mendapat nilai 196. Kemudian diakarkan sehingga mendapatkan nilai 14. Sehingga jarak euclidean dari 2 titik tersebut adalah 14.
 - **2 dimensi**
Caranya hampir sama. Misalkan titik pertama mempunyai kordinat (3,5). Titik kedua ada di kordinat (5,-3). Caranya adalah kurangkan setiap kordinat titik kedua dengan titik yang pertama. Yaitu, (5-3, -3-5) sehingga menjadi (2,-8). Kemudian pangkatnya sehingga memperoleh (4,64). Kemudian tambahkan semuanya sehingga memperoleh nilai $64+4 = 68$. Hasil ini kemudian diakarkan menjadi 8.25. Sehingga jarak euclideanya menjadi 8.25.

Euclidean Distance [2]

- **Euclidean Distance** adalah metrika yang paling sering digunakan untuk menghitung kesamaan 2 vektor.
- Euclidean distance menghitung akar dari kuadrat perbedaan 2 vektor (root of square differences between 2 vectors).

Rumus dari Euclidian Distance:

$$d_{ij} = \sqrt{\sum_{k=1}^n (x_{ik} - x_{jk})^2}$$

- Contoh :

Terdapat 2 vektor sebagai beriku

$$\begin{aligned}A &= [0, 3, 4, 5] \\B &= [7, 6, 3, -1]\end{aligned}$$

Euclidean Distance dari vektor A dan B adalah

$$\begin{aligned}d_{AB} &= \sqrt{(0-7)^2 + (3-6)^2 + (4-3)^2 + (5-(-1))^2} \\&= \sqrt{49+9+1+36} = 9.747\end{aligned}$$

Contoh Kasus 1

Menggunakan algoritma K-Means temukan pengelompokan dan mean terbaik dari dua cluster data 2D di bawah ini.

$$M_1 = (2, 5.0),$$

$$M_2 = (2, 5.5),$$

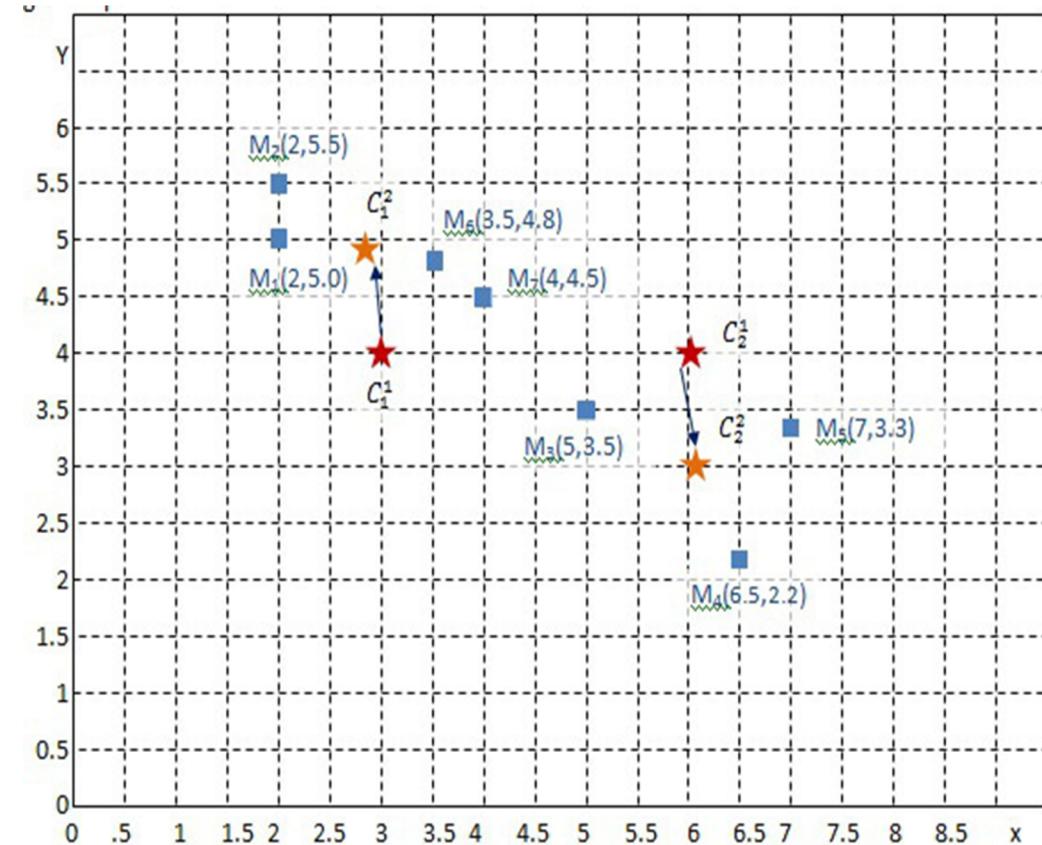
$$M_3 = (5, 3.5),$$

$$M_4 = (6.5, 2.2),$$

$$M_5 = (7, 3.3),$$

$$M_6 = (3.5, 4.8),$$

$$M_7 = (4, 4.5)$$



Asumsi:

- Semua data akan dikelompokkan ke dalam dua kelas
- Titik pusat kedua cluster adalah C₁(3,4), C₂(6,4)

Contoh Kasus – Iterasi 1

Iterasi 1

a. Menghitung *Euclidean distance* dari semua data ke tiap titik pusat pertama.

Sehingga didapatkan:

$$D_{11}=1.41,$$

$$D_{12}=1.80,$$

$$D_{13}=2.06,$$

$$D_{14}=3.94,$$

$$D_{15}=4.06,$$

$$D_{16}=0.94,$$

$$D_{17}=1.12,$$

$$D_{11} = \sqrt{(M_{1x} - C_{1x})^2 + (M_{1y} - C_{1y})^2} = \sqrt{(2 - 3)^2 + (5 - 4)^2} = \sqrt{2} = 1.41$$

$$D_{12} = \sqrt{(M_{2x} - C_{1x})^2 + (M_{2y} - C_{1y})^2} = \sqrt{(2 - 3)^2 + (5.5 - 4)^2} = \sqrt{3.25} = 1.80$$

$$D_{13} = \sqrt{(M_{3x} - C_{1x})^2 + (M_{3y} - C_{1y})^2} = \sqrt{(5 - 3)^2 + (3.5 - 4)^2} = \sqrt{4.25} = 2.06$$

$$D_{14} = \sqrt{(M_{4x} - C_{1x})^2 + (M_{4y} - C_{1y})^2} = \sqrt{(6.5 - 3)^2 + (2.2 - 4)^2} = \sqrt{2} = 3.94$$

$$D_{15} = \sqrt{(M_{5x} - C_{1x})^2 + (M_{5y} - C_{1y})^2} = \sqrt{(7 - 3)^2 + (3.3 - 4)^2} = \sqrt{2} = 4.06$$

$$D_{16} = \sqrt{(M_{6x} - C_{1x})^2 + (M_{6y} - C_{1y})^2} = \sqrt{(3.5 - 3)^2 + (4.8 - 4)^2} = \sqrt{2} = 0.94$$

$$D_{17} = \sqrt{(M_{7x} - C_{1x})^2 + (M_{7y} - C_{1y})^2} = \sqrt{(4 - 3)^2 + (4.5 - 4)^2} = \sqrt{2} = 1.12$$

- Dengan cara yang sama hitung jarak tiap titik ke titik pusat kedua, dan kita akan mendapatkan :

$$D_{21} = 4.12, D_{22} = 4.27, D_{23} = 1.18, D_{24} = 1.86,$$

$$D_{25} = 1.22, D_{26} = 2.62, D_{27} = 2.06.$$

Contoh Kasus – Iterasi 1 [2]

b. Dari penghitungan *Euclidean distance*, kita dapat membandingkan:

	M1	M2	M3	M4	M5	M6	M7
Jarak ke C1	1.41	1.80	2.06	3.94	4.06	0.94	1.12
Jarak ke C2	4.12	4.27	1.18	1.86	1.22	2.62	2.06

{M1, M2, M6, M7} anggota C1 and {M3, M4, M5} anggota C2

c. Hitung titik pusat baru

$M1 = (2, 5.0)$, $M2 = (2, 5.5)$, $M3 = (5, 3.5)$, $M4 = (6.5, 2.2)$, $M5 = (7, 3.3)$, $M6 = (3.5, 4.8)$,
 $M7 = (4, 4.5)$

$$C1 = \left(\frac{2+2+3.5+4}{4}, \frac{5+5.5+4.8+4.5}{4} \right) = (2.85, 4.95)$$

$$C2 = \left(\frac{5+6.5+7}{3}, \frac{3.5+2.2+3.3}{3} \right) = (6.17, 3)$$

Contoh Kasus – Iterasi 1 [3]

ITERASI 2

- a) Hitung Euclidean distance dari tiap data ke titik pusat yang baru Dengan cara yang sama dengan iterasi pertama kita akan mendapatkan perbandingan sebagai berikut:

	M ₁	M ₂	M ₃	M ₄	M ₅	M ₆	M ₇
Jarak ke C ₁	0.76	0.96	2.65	4.62	4.54	0.76	1.31
Jarak ke C ₂	4.62	4.86	1.27	0.86	0.88	3.22	2.63

- b) Dari perbandingan tersebut kita tahu bahwa {M₁, M₂, M₆, M₇} anggota C₁ dan {M₃, M₄, M₅} anggota C₂
c) Karena anggota kelompok tidak ada yang berubah maka titik pusat pun tidak akan berubah.

KESIMPULAN

{M₁, M₂, M₆, M₇} anggota C₁ dan {M₃, M₄, M₅} anggota C₂

Contoh Kasus 2

- Diketahui Dataset Panen Tanaman Padi sebagai berikut :

No	Luas Panen (ha)	Produksi
1	1081	6142
2	2172	12354
3	213	1202
4	828	4708
5	135	766
6	176	993
7	15	85
8	40	224
9	1014	5758
10	889	5046

- Tentukan anggota untuk masing-masing klaster jika dibentuk 3 klaster?

Langkah 1

- Tentukan nilai k.
- Diketahui jumlah klaster yang akan dibentuk sebanyak 3 sehingga $k = 3$.

Langkah 2

- Bangkitkan Centroid Awal secara random.
- Misal dibangkitkan centroid (titik pusat klaster) sebagai berikut :
- | | X | Y |
|----|------|-------|
| C1 | 1000 | 10000 |
| C2 | 500 | 5000 |
| C3 | 50 | 100 |

Langkah 3

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat pertama*** :

- $D(1,1) = \sqrt{(M_{1x} - C_{1x})^2 + (M_{1y} - C_{1y})^2} = \sqrt{(1081 - 1000)^2 + (6142 - 10000)^2} = 3858,85$

$$D(1,2) = \sqrt{(M_{2x} - C_{1x})^2 + (M_{2y} - C_{1y})^2} = \sqrt{(2172 - 1000)^2 + (12354 - 10000)^2} = 2629,62$$

$$D(1,3) = \sqrt{(M_{3x} - C_{1x})^2 + (M_{3y} - C_{1y})^2} = \sqrt{(213 - 1000)^2 + (1202 - 10000)^2} = 8833,13$$

$$D(1,4) = \sqrt{(M_{4x} - C_{1x})^2 + (M_{4y} - C_{1y})^2} = \sqrt{(828 - 1000)^2 + (4708 - 10000)^2} = 5294,79$$

$$D(1,5) = \sqrt{(M_{5x} - C_{1x})^2 + (M_{5y} - C_{1y})^2} = \sqrt{(135 - 1000)^2 + (766 - 10000)^2} = 9274,43$$

Langkah 3 [2]

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat pertama*** :

- $D(1,6) = \sqrt{(M_{6x} - C_{1x})^2 + (M_{6y} - C_{1y})^2} = \sqrt{(176 - 1000)^2 + (993 - 10000)^2} = 9044,61$

$$D(1,7) = \sqrt{(M_{7x} - C_{1x})^2 + (M_{7y} - C_{1y})^2} = \sqrt{(15 - 1000)^2 + (85 - 10000)^2} = 9963,81$$

$$D(1,8) = \sqrt{(M_{8x} - C_{1x})^2 + (M_{8y} - C_{1y})^2} = \sqrt{(40 - 1000)^2 + (224 - 10000)^2} = 9823,02$$

$$D(1,9) = \sqrt{(M_{9x} - C_{1x})^2 + (M_{9y} - C_{1y})^2} = \sqrt{(1014 - 1000)^2 + (5758 - 10000)^2} = 4242,02$$

$$D(1,10) = \sqrt{(M_{10x} - C_{1x})^2 + (M_{10y} - C_{1y})^2} = \sqrt{(889 - 1000)^2 + (5048 - 10000)^2} = 4955,24$$

Langkah 3 [3]

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat kedua*** :

- $D(2,1) = \sqrt{(M_{1x} - C_{2x})^2 + (M_{1y} - C_{2y})^2} = \sqrt{(1081 - 500)^2 + (6142 - 5000)^2} = 1281,3$

$$D(2,2) = \sqrt{(M_{2x} - C_{2x})^2 + (M_{2y} - C_{2y})^2} = \sqrt{(2172 - 500)^2 + (12354 - 5000)^2} = 7541,68$$

$$D(2,3) = \sqrt{(M_{3x} - C_{2x})^2 + (M_{3y} - C_{2y})^2} = \sqrt{(213 - 500)^2 + (1202 - 5000)^2} = 3808,83$$

$$D(2,4) = \sqrt{(M_{4x} - C_{2x})^2 + (M_{4y} - C_{2y})^2} = \sqrt{(828 - 500)^2 + (4708 - 5000)^2} = 439,14$$

$$D(2,5) = \sqrt{(M_{5x} - C_{2x})^2 + (M_{5y} - C_{2y})^2} = \sqrt{(135 - 500)^2 + (766 - 5000)^2} = 4249,7$$

Langkah 3 [4]

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat kedua*** :

- $$D(2,6) = \sqrt{(M_{6x} - C_{2x})^2 + (M_{6y} - C_{26})^2} = \sqrt{(176 - 500)^2 + (993 - 5000)^2} = 4020,08$$

$$D(2,7) = \sqrt{(M_{7x} - C_{2x})^2 + (M_{7y} - C_{2y})^2} = \sqrt{(15 - 500)^2 + (85 - 5000)^2} = 4938,87$$

$$D(2,8) = \sqrt{(M_{8x} - C_{2x})^2 + (M_{8y} - C_{2y})^2} = \sqrt{(40 - 500)^2 + (224 - 5000)^2} = 4798,1$$

$$D(2,9) = \sqrt{(M_{9x} - C_{2x})^2 + (M_{9y} - C_{2y})^2} = \sqrt{(1014 - 500)^2 + (5758 - 5000)^2} = 915,84$$

$$D(2,10) = \sqrt{(M_{10x} - C_{2x})^2 + (M_{10y} - C_{2y})^2} = \sqrt{(889 - 500)^2 + (5048 - 5000)^2} = 391,71$$

Langkah 3 [5]

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat ketiga*** :

- $D(3,1) = \sqrt{(M_{1x} - C_{3x})^2 + (M_{1y} - C_{3y})^2} = \sqrt{(1081 - 50)^2 + (6142 - 100)^2} = 6129,33$

$$D(3,2) = \sqrt{(M_{2x} - C_{3x})^2 + (M_{2y} - C_{3y})^2} = \sqrt{(2172 - 50)^2 + (12354 - 100)^2} = 12436,37$$

$$D(3,3) = \sqrt{(M_{3x} - C_{3x})^2 + (M_{3y} - C_{3y})^2} = \sqrt{(213 - 50)^2 + (1202 - 100)^2} = 1113,99$$

$$D(3,4) = \sqrt{(M_{4x} - C_{3x})^2 + (M_{4y} - C_{3y})^2} = \sqrt{(828 - 50)^2 + (4708 - 100)^2} = 4673,22$$

$$D(3,5) = \sqrt{(M_{5x} - C_{3x})^2 + (M_{5y} - C_{3y})^2} = \sqrt{(135 - 50)^2 + (766 - 100)^2} = 671,4$$

Langkah 3 [6]

- Hitung jarak setiap data ke masing-masing centroid.
- Menghitung Euclidean distance setiap data dengan ***titik pusat ketiga*** :

- $D(3,6) = \sqrt{(M_{6x} - C_{3x})^2 + (M_{6y} - C_{3y})^2} = \sqrt{(176 - 50)^2 + (993 - 100)^2} = 901,85$

$$D(3,7) = \sqrt{(M_{7x} - C_{3x})^2 + (M_{7y} - C_{3y})^2} = \sqrt{(15 - 50)^2 + (85 - 100)^2} = 38,08$$

$$D(3,8) = \sqrt{(M_{8x} - C_{3x})^2 + (M_{8y} - C_{3y})^2} = \sqrt{(40 - 50)^2 + (224 - 100)^2} = 124,4$$

$$D(3,9) = \sqrt{(M_{9x} - C_{3x})^2 + (M_{9y} - C_{3y})^2} = \sqrt{(1014 - 50)^2 + (5758 - 100)^2} = 5739,53$$

$$D(3,10) = \sqrt{(M_{10x} - C_{3x})^2 + (M_{10y} - C_{3y})^2} = \sqrt{(889 - 50)^2 + (5048 - 100)^2} = 5016,66$$

Langkah 4

- Kelompokkan setiap data

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
C1	3858.85	2629.62	8833.13	5294.79	9274.43	9044.61	9963.81	9823.02	4242.02	4955.24
C2	1281.3	7541.68	3808.83	439.14	4249.7	4020.08	4938.87	4798.1	915.84	391.71
C3	6129.33	12436.37	1113.99	4673.22	671.4	901.85	38.08	124.4	5739.53	5016.66

- {M2} anggota C1
- {M1, M4, M9, M10} anggota C2
- {M3, M5, M6, M7, M8} anggota C3

Langkah 5

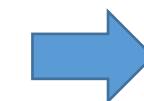
- Tentukan posisi centroid baru

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
C1	3858.85	2629.62	8833.13	5294.79	9274.43	9044.61	9963.81	9823.02	4242.02	4955.24
C2	1281.3	7541.68	3808.83	439.14	4249.7	4020.08	4938.87	4798.1	915.84	391.71
C3	6129.33	12436.37	1113.99	4673.22	671.4	901.85	38.08	124.4	5739.53	5016.66

$$C1 = \left(\frac{2172}{1}, \frac{12354}{1} \right) = (2172, 12354)$$

$$C2 = \left(\frac{1081+828+1014+889}{4}, \frac{6142+4708+5758+5046}{4} \right) = (953, 5413.5)$$

$$C2 = \left(\frac{213+135+176+40}{5}, \frac{1202+766+933+224}{5} \right) = (115.8, 654)$$



	X	Y
C1	2172	12354
C2	953	5413.5
C3	115.8	654

Langkah 6

- Kelompokkan setiap data.
- Hasil Iterasi 2:

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
C1	6307.08	0	11322.76	7763.22	11765.68	11535	12457.17	12315.94	6696.88	7419.77
C2	739.66	7046.74	4276.02	716.49	4718.94	4488.27	5410.43	5269.2	349.86	373.03
C3	5572.23	11879.31	556.55	4116.08	113.63	344.3	577.86	436.63	5182.43	4459.54

- Hasil Iterasi 1:

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
C1	3858.85	2629.62	8833.13	5294.79	9274.43	9044.61	9963.81	9823.02	4242.02	4955.24
C2	1281.3	7541.68	3808.83	439.14	4249.7	4020.08	4938.87	4798.1	915.84	391.71
C3	6129.33	12436.37	1113.99	4673.22	671.4	901.85	38.08	124.4	5739.53	5016.66

Karena anggota kelompok tidak ada yang berubah maka kesimpulannya :

- {M2} anggota C1.
- {M1, M4, M9, M10} anggota C2.
- {M3, M5, M6, M7, M8} anggota C3.

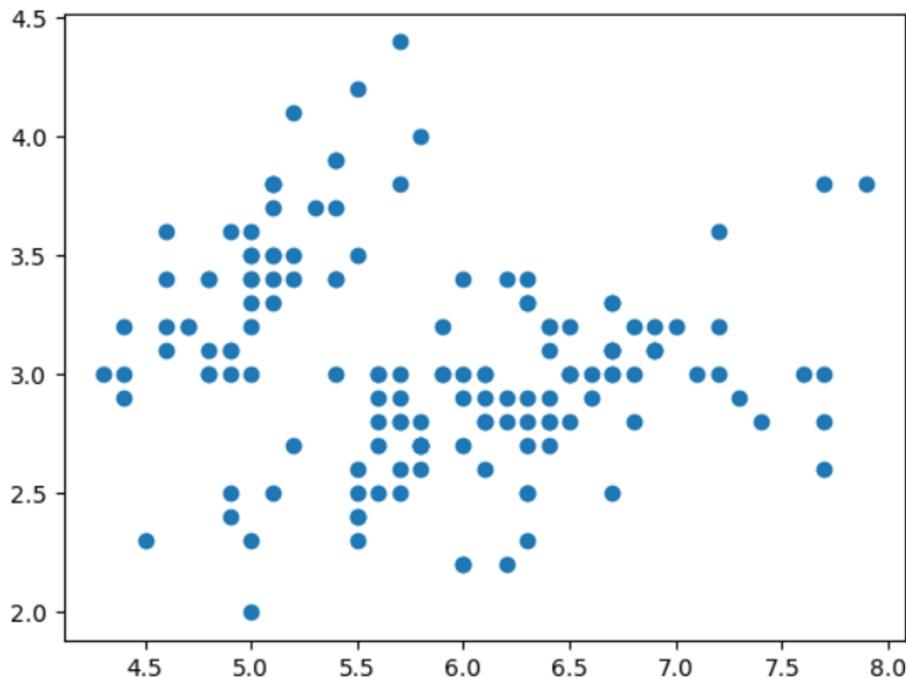
Implementasi Python

- Mengimpor library python yang diperlukan:

```
from sklearn import datasets
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from scipy.spatial.distance import cdist
import matplotlib.pyplot as plt
```

Implementasi Python [2]

```
iris = datasets.load_iris()  
features = iris.data  
plt.scatter(features[:,0], features[:,1])  
plt.show()
```



- Membaca data input.
- Sebagai ilustrasi, dataset yang digunakan di dalam program ini adalah data iris yang bersumber kepada data yang dipublikasikan oleh Fisher (Fisher, 1950).
- Data tersebut dapat didownload dari website UCI Machine Learning Repository.
- Hasil visualisasi scatter data iris bisa dilihat pada gambar disamping.

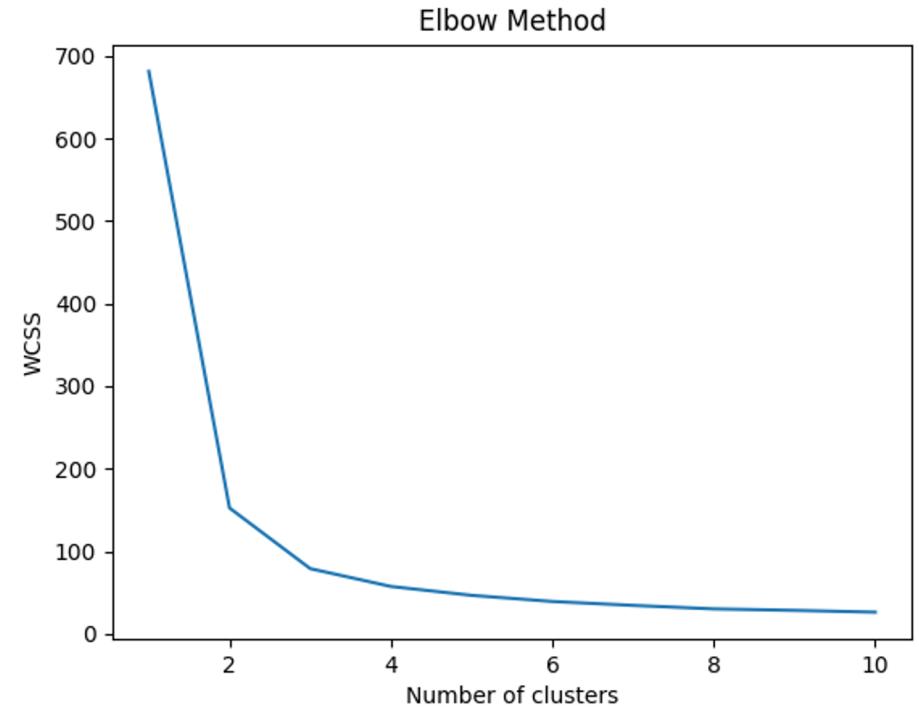
Implementasi Python [3]

- Menstandarisasi fitur.

```
scaler = StandardScaler()  
features_standardized = scaler.fit_transform(features)
```

Implementasi Python [4]

```
from sklearn.metrics import silhouette_samples, silhouette_score
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++',
                     max_iter=300, n_init=10, random_state=0)
    kmeans.fit(features)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



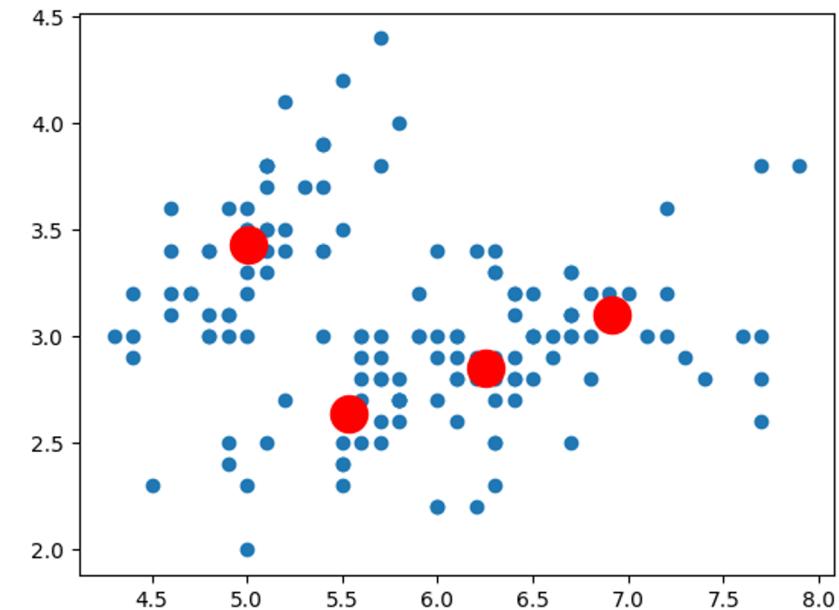
- Membuat klastering dan mencari nilai k yang paling optimal, dengan menggunakan elbow method.
- Tampilan hasil elbow method seperti terlihat pada gambar diatas.

Implementasi Python [5]

- Melakukan klastering.

```
kmeans = KMeans(n_clusters=4, init='k-means++',
                  max_iter=300, n_init=10, random_state=0)
pred_y = kmeans.fit_predict(features)
plt.scatter(features[:,0], features[:,1])
plt.scatter(kmeans.cluster_centers_[:,0],
            kmeans.cluster_centers_[:,1], s=300, c='red')
plt.show()
```

- Hasil visualisasi klaster seperti gambar disamping.



Implementasi Python dengan Dataset txt

```
#import libary yang dibutuhkan
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.pyplot import cm
import time
import itertools

#baca dataset dari file
datasetPath = "D:\dataku.txt"
dataset = np.loadtxt(datasetPath, delimiter=" ")

#mendefinisikan parameter k-means klustering
k = 2 #jumlah klaster yg diinginkan
iterationCounter = 0 #counter untuk iterasi
input = dataset #input data

#fungsi untuk inisialisasi titik pusat klaster (random)
def initCentroid(dataIn, k):
    result = dataIn[np.random.choice(dataIn.shape[0], k, replace=False)]
    return result
```

-9.6961	6.0031
11.9126	0.2844
-3.9586	0.0571
-3.6131	-1.3299
-1.6446	-3.0697
-9.0433	-8.0838
-2.9757	-1.5796
13.4308	-6.4078
1.9012	0.9898
3.8303	13.9523
3.0094	2.1778

- Pertama kali kita harus import library yang dibutuhkan seperti numpy, matplotlib, dll.
- Kemudian baca dataset dari file (dataku.txt => berisi 400 data dalam bentuk teks).
- Setelah itu mendefinisikan parameter K-Means.
- Buat fungsi untuk inisialisasi titik pusat kluster secara random.

Implementasi Python dengan Dataset txt [2]

```
#fungsi untuk plot hasil klaster per iterasi
def plotClusterResult(listClusterMembers, centroid, iteration, converged):
    n = listClusterMembers.__len__()
    color = iter(cm.rainbow(np.linspace(0, 1, n)))
    plt.figure("result")
    plt.clf()
    plt.title("iteration-" + iteration)
    marker = itertools.cycle(( '.', '*', '^', 'x', '+'))
    for i in range(n):
        col = next(color)
        memberCluster = np.asmatrix(listClusterMembers[i])
        plt.scatter(np.ravel(memberCluster[:, 0]), np.ravel(memberCluster[:, 1]),
                   marker=marker.__next__(), s=100, c=col, label="klaster-"+str(i+1))
    for i in range(n):
        plt.scatter((centroid[i, 0]), (centroid[i, 1]), marker=marker.__next__(),
                   , c=col, label="centroid-" + str(i + 1))
    if(converged == 0):
        plt.legend()
        plt.ion()
        plt.show()
        plt.pause(0.1)
    if (converged == 1):
        plt.legend()
        plt.show(block=True)
```

- Selanjutnya buat fungsi untuk plot hasil klister per iterasi.

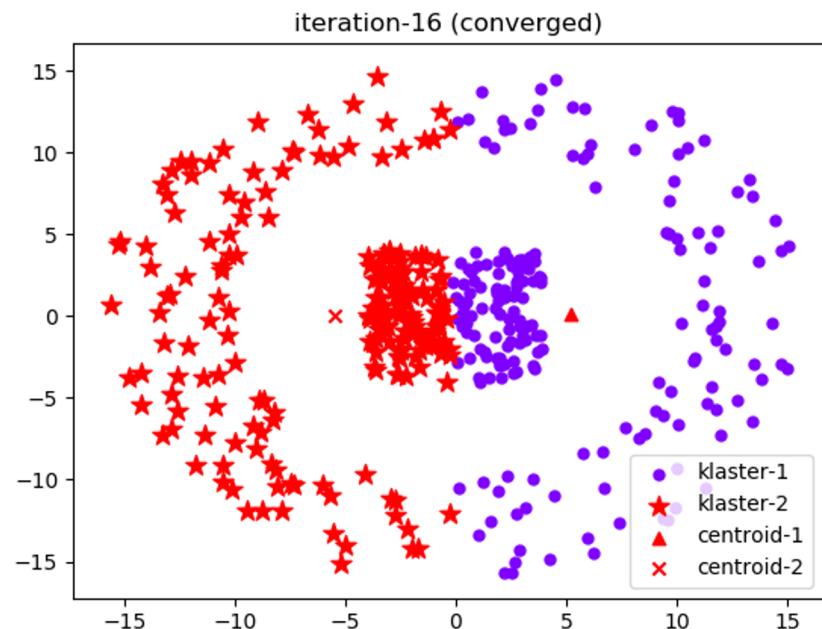
Implementasi Python dengan Dataset txt [3]

```
#fungsi utama algoritma k-means
def kMeans(data, centroidInit):
    nCluster = k #banyaknya klaster
    global iterationCounter
    centroidInit = np.matrix(centroidInit)
    # looping hingga konvergen
    while(True):
        iterationCounter +=1
        euclideanMatrixAllCluster = np.ndarray(shape=(data.shape[0], 0))
        #ulangi proses untuk semua klaster
        for i in range(0, nCluster):
            centroidRepeated = np.repeat(centroidInit[i,:], data.shape[0], axis=0)
            deltaMatrix = abs(np.subtract(data,centroidRepeated))
            #hitung jarak Euclidean
            euclideanMatrix = np.sqrt(np.square(deltaMatrix).sum(axis=1))
            euclideanMatrixAllCluster = \
                np.concatenate((euclideanMatrixAllCluster, euclideanMatrix), axis=1)
            #tempatkan data ke klaster yang jarak Euclideananya plg dekat
            clusterMatrix = np.ravel(np.argmin(np.matrix(euclideanMatrixAllCluster), axis=1))
            listClusterMember = [[] for i in range(k)]
            for i in range(0, data.shape[0]):#assign data to cluster regarding cluster matrix
                listClusterMember[np.asscalar(clusterMatrix[i])].append(data[i,:])
            #hitung titik pusat klaster terbaru
            newCentroid = np.ndarray(shape=(0, centroidInit.shape[1]))
            for i in range(0,nCluster):
                memberCluster = np.asmatrix(listClusterMember[i])
                centroidCluster = memberCluster.mean(axis=0)
                newCentroid = np.concatenate((newCentroid, centroidCluster), axis=0)
            print("iter: ", iterationCounter)
            print("centroid: ", newCentroid)
            #break dari loop jika sudah konvergen
            if((centroidInit == newCentroid).all()):
                break
            # update titik pusat klaster dengan nilai yg baru
            centroidInit = newCentroid
            #plot hasil klaster per iterasi
            plotClusterResult(listClusterMember, centroidInit, str(iterationCounter), 0)
            time.sleep(1) #diberi jeda 1 detik agak hasil plot klaster nyaman dilihat
    return listClusterMember, centroidInit
```

- Selanjutnya buat fungsi utama untuk algoritma K-Means.

Implementasi Python dengan Dataset txt [4]

```
#panggil fungsi inisialisasi klaster
centroidInit = initCentroid(input, k)
#panggil fungsi k-means
clusterResults, centroid = kMeans(input, centroidInit)
#plot hasil final klaster setelah konvergen
plotClusterResult(clusterResults, centroid, str(iterationCounter) + " (converged)", 1)
```



- Kemudian panggil fungsi inisialisasi kluster dan panggil fungsi utama K-Means.
- Selanjutnya plot hasil final klaster setelah konvergen.
- Hasil visualisasi Klaster seperti gambar disamping.

Implementasi Python dengan Dataset CSV

A	B	C
No	Gaji	Pengeluaran
1	2500	1750
2	3800	4200
3	3900	3800
4	4350	5500
5	4400	3200
6	5500	5450
7	5600	5950
8	5750	4100
9	6850	6050
10	6900	8500
11	7250	9500
12	7350	6050
13	7500	8500
14	7800	9500
15	8200	8300
16	8500	6500
17	8550	8400
18	8750	6000
19	9100	10500
20	9100	8500

- Siapkan 20 record data, kemudian simpan dengan nama konsumen.csv
- Kemudian, memanggil library yang dibutuhkan.

```
#Import Library yang akan digunakan
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as numpy
import pandas as pd
from sklearn.cluster import KMeans
```

- Ket:
 - Matplotlib => membuat grafik plot
 - Numpy => untuk kebutuhan scientific.
 - Pandas => untuk manipulasi data, spt membuat tabel, mengubah dimensi data, mengecek data dsb.
 - Sklearn => untuk metode dan algoritma Machine Learning.

Implementasi Python dengan Dataset CSV [2]

- Memanggil dataset konsumen.csv

```
#Menyiapkan data dan memanggil dataset
dataset = pd.read_csv('D:\konsumen.csv')
dataset.keys()
```

- Untuk menampilkan 5 baris data pertama dari dataset tersebut:

```
dataku = pd.DataFrame(dataset)
dataku.head()
```

Implementasi Python dengan Dataset CSV [3]

- Konversi Dataset ke Data Array:

```
#Konversi ke data Array  
X = np.asarray(dataset)  
print(X)
```

- Hasil konversinya seperti tampil pada gambar disamping.

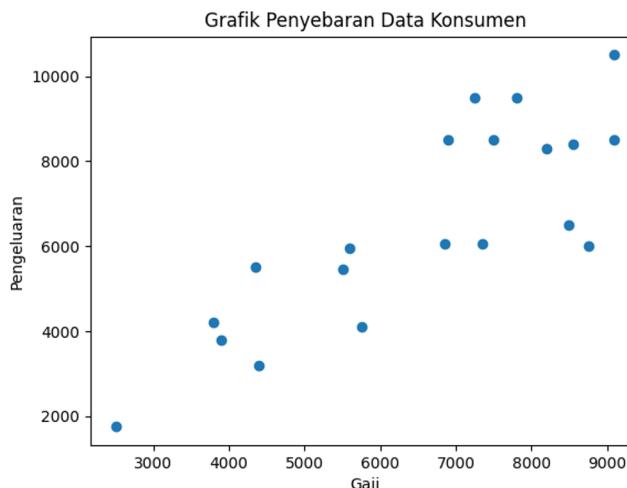
```
[[ 2500 1750]  
[ 3800 4200]  
[ 3900 3800]  
[ 4350 5500]  
[ 4400 3200]  
[ 5500 5450]  
[ 5600 5950]  
[ 5750 4100]  
[ 6850 6050]  
[ 6900 8500]  
[ 7250 9500]  
[ 7350 6050]  
[ 7500 8500]  
[ 7800 9500]  
[ 8200 8300]  
[ 8500 6500]  
[ 8550 8400]  
[ 8750 6000]  
[ 9100 10500]  
[ 9100 8500]]
```

Implementasi Python dengan Dataset CSV [4]

- Menampilkan data Array ke dalam Scatter Plot.
- Untuk visualisasi data, berikut adalah perintah untuk menampilkan data array.

```
#Menampilkan data dalam bentuk scatter plot
plt.scatter(X[:,0], X[:,1], label='True Position')
plt.xlabel("Gaji")
plt.ylabel("Pengeluaran")
plt.title("Grafik Penyebaran Data Konsumen")
plt.show()
```

- Hasil visualisasi penyebaran data konsumen seperti terlihat pada gambar dibawah ini.



Implementasi Python dengan Dataset CSV [5]

- Mengaktifkan K-Means dari Sklearn.
- Untuk langkah selanjutnya, perlu dilakukan konversi dataset ke dalam tipe data array, dan kemudian melakukan fitting data dengan kode program sebagai berikut.

```
#Mengaktifkan K-Means dengan jumlah K=2
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
```

Implementasi Python dengan Dataset CSV [6]

- Menampilkan nilai Centroid.
- Untuk menampilkan nilai centroid atau titik pusat klaster yang degenerate oleh algoritma dapat menggunakan kode sebagai berikut:

```
#Menampilkan nilai Centroid yang digenerate oleh algoritma
print(kmeans.cluster_centers_)
```

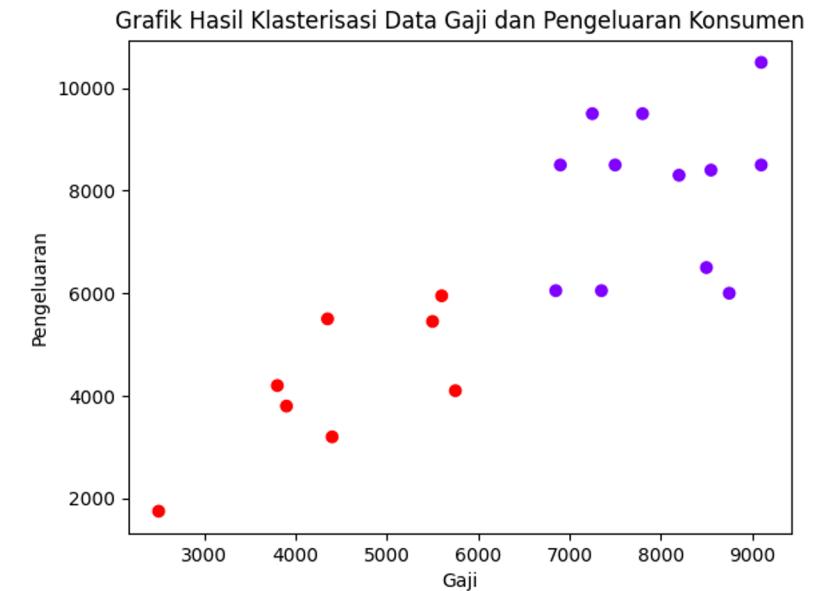
- Hasil Nilai Centroid seperti gambar dibawah ini:

```
[[4475.    4243.75]
 [7987.5   8025.  ]]
PS C:\Users\Junz> 
```

Implementasi Python dengan Dataset CSV [7]

- Visualisasi Hasil.
- Untuk menampilkan scatter plot dari data-data setelah dilakukan klasterisasi oleh algoritma K-Means sebagai berikut:

```
#Plot Data Point
#Memvisualisasikan Hasil Klasterisasi Data Konsumen
plt.scatter(X[:,0], X[:,1], c=kmeans.labels_, cmap='rainbow')
plt.xlabel("Gaji")
plt.ylabel("Pengeluaran")
plt.title("Grafik Hasil Klasterisasi Data Gaji dan Pengeluaran Konsumen")
plt.show()
```

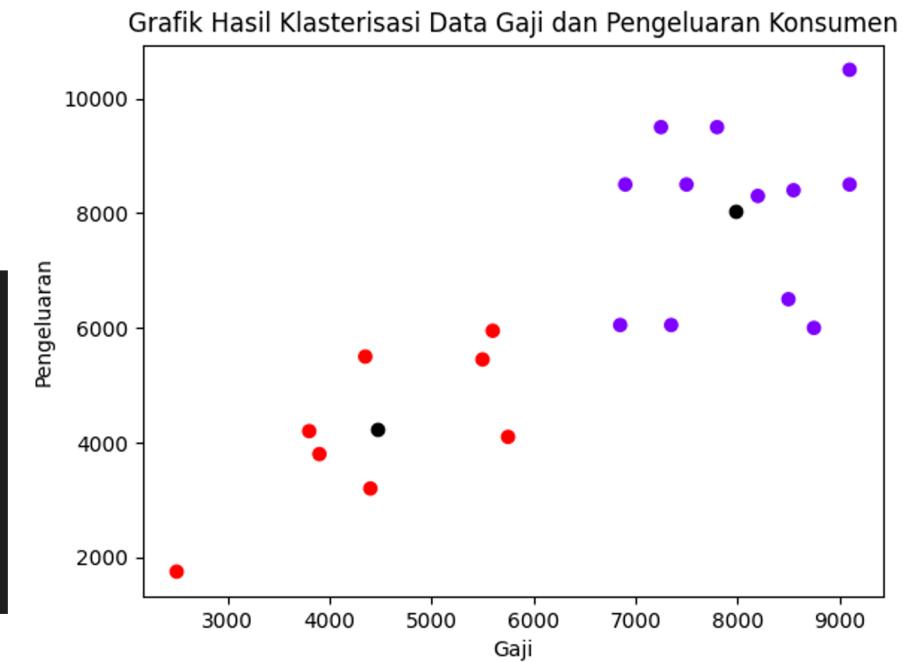


- Dan visualisasi hasil klaster dapat dilihat pada gambar grafik disamping.

Implementasi Python dengan Dataset CSV [8]

- Kita juga bisa menampilkan centroid dari masing-masing klaster dengan kode sebagai berikut:

```
#Plot Data Point
#Memvisualisasikan hasil klasterisasi dengan centroid dr masing2 klaster
plt.scatter(X[:,0], X[:,1], c=kmeans.labels_, cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], color='black')
plt.xlabel("Gaji")
plt.ylabel("Pengeluaran")
plt.title("Grafik Hasil Klasterisasi Data Gaji dan Pengeluaran Konsumen")
plt.show()
```



- Dan visualisasi hasil klaster dapat dilihat pada gambar grafik disamping.