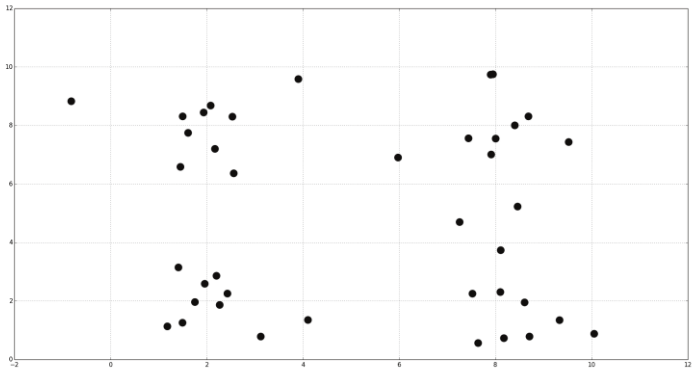
BAB 12
K-means

K-means merupakan salah satu algoritma un-supervised learning yang digunakan untuk permasalahan clustering/pengelompokan. Pada unsupervised learning belum diketahui label/class dari suatu dataset. Teknik unsupervised learning akan mencari pola-pola data yang tersembuyi dari dataset yang tidak berlabel tersebut. Tujuan dari unsupervised learning adalah mencari pola-pola data berdasarkan kemiripan data.

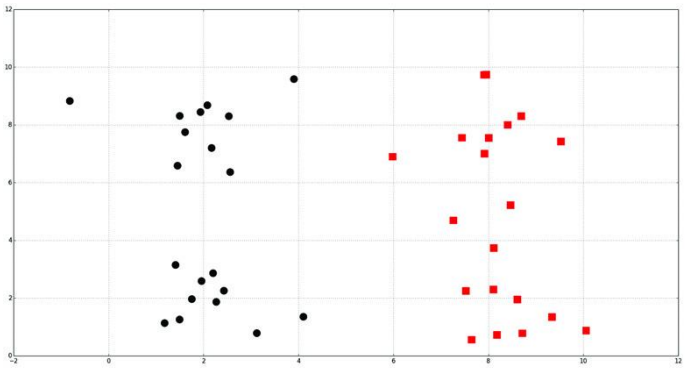
1. Clustering

Clustering atau yang sering disebut sebagai cluster analysis adalah salah satu algoritma machine learning yang bertujuan untuk Menemukan sebuah kumpulan data atau objek yang memiliki kemiripan satu sama lain di dalam kumpulan atau kelompok tersebut, dan berbeda dengan objek di kelompok lain. Berdasarkan pengelompokan data menjadi cluster, metode clustering dibedakan menjadi dua jenis yaitu partitioning dan hierarchical. Partitioning (flat clustering) merupakan metode yang paling mendasar dan sederhana dimana data dikelompokkan menjadi cluster tanpa struktur yang eksplisit. Metode patitioning yang populer dan sering digunakan adalah k-means. Sedangkan Hierarchical Clustering menghasilkan output sebuah struktur hierarki dari cluster. Algortima agglomerative hierarchical clustering merupakan salah satu metode Hierarchical clustering yang sering digunakan pada data teks. (Han, Kamber, & Pei, 2012)

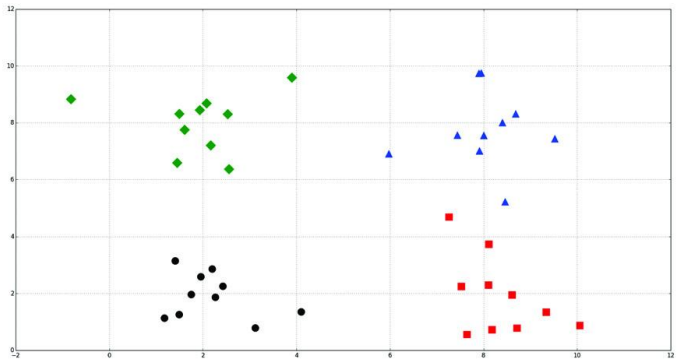
Ilustrasi Clustering dapat dilihat pada gambar 12.1



12.1 (a) Plot Dataset awal



12.1 (b) Plot dataset Ketika dikelompokkan menjadi 2 cluster



12.1 (c) Plot dataset Ketika dikelompokkan menjadi 4 cluster

Gambar 12. 1 Ilustrasi Clustering

Contoh implementasi clustering :

a. Menemukan segmentasi pelanggan berdasarkan transaksi yang dilakukan
b. Sistem rekomendasi terkadang menggunakan pengelompokan untuk mengidentifikasi produk atau media lain yang mungkin menarik bagi pengguna.
c. Dalam biologi, pengelompokan digunakan untuk menemukan kelompok gen dengan pola ekspresi yang serupa.
d. Pengelompokan orang di jejaring sosial berdasarkan kesamaan tekstual atau komunitas.

2. K-means

K-means diperkenalkan oleh (MacQueen, 1967) merupakan salah satu teknik partitioning berbasis centroid dimana teknik ini menggunakan centroid dari cluster untuk mewakili cluster tersebut. K-means clustering merupakan algoritma clustering yang popular karena kecepatan dan juga skalabilitasnya. Pada k-means penentuan centroid awal dilakukan secara random dengan menentukan sejumlah k centroid. Parameter k adalah hyperparameter yang menentukan jumlah cluster yang harus dibuat dan harus berupa bilangan bulat positif yang lebih kecil dari jumlah instance (jumlah data) yang ada pada dataset. Terkadang jumlah cluster ditentukan oleh konteks masalah clustering.

Centroid selanjutnya dicari dengan cara menghitung rata-rata dari titik data yang ditetapkan sebagai anggota cluster. Langkah pertama adalah untuk memilih jumlah k centroid sesuai cluster yang akan dibentuk. Selanjutnya data-data selain centroid akan dihitung kemiripannya terhadap centroid. Algoritma k-means kemudian menentukan anggota cluster dengan cara memilih data - data yang memiliki kemiripan tertinggi antara centroid dan non-centroid dokumen. Selanjutnya untuk setiap cluster dihitung rata-rata dari anggota cluster untuk menentukan centroid baru. iterasi berlanjut sampai tidak ada perubahan anggota cluster.

Tahapan Algoritma k-means

Input : Data = { d1, d2, d3 ,d4.... dn }
K = jumlah dari cluster
Output : himpunan k cluster C= (c1,c2,...ck)
Metode :

a) Pilih k data secara random pada sekumpulan n data sebagai centroid
b) Hitung jarak antara centroid dengan setiap data non-centroid
c) Dapatkan anggota cluster dengan memilih data yang mempunyai jarak yang paling kecil antara antara data non centroid dengan centroid
d) Update centroid dengan cara menghitung rata-rata dari anggota cluster
e) Ulangi langkah b sampai d sampai tidak ada perubahan dari centroid.

Salah satu metode untuk melakukan perhitungan jarak pada algoritma k-means adalah Euclidean distance (persamaan 12.1) . Perhitungan jarak yang digunakan dapat disesuaikan dengan konteks permasalahan. Jika diterapkan pada document clustering maka perhitungan kemiripan yang digunakan dapat berupa cosine similarity.

Euclidean distance

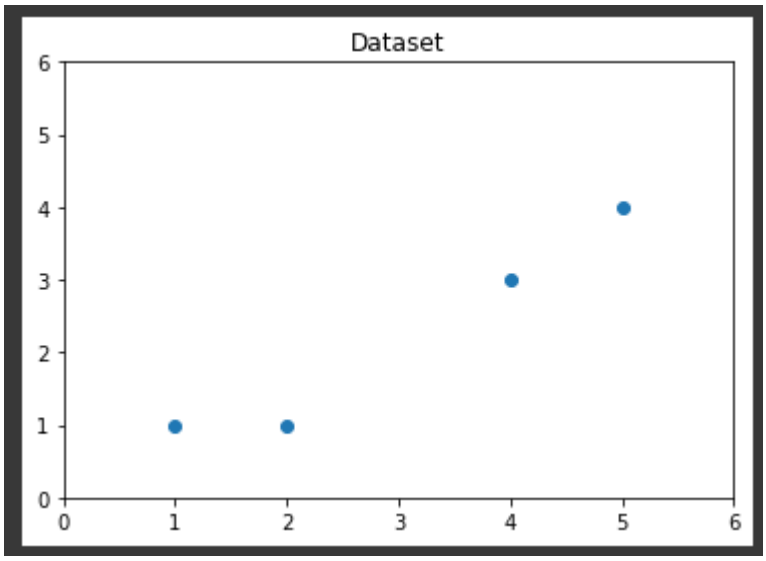
$$d(xi, yi) = \sqrt{\sum_{i=1}^{n}(xi - yi)^2} \quad (12.1)$$

Contoh Perhitungan Manual k-means

Contoh dataset

No	x	y
1	1	1
2	2	1
3	4	3
4	5	4

Ploting data tersebut adalah sebagai berikut :



a. Pilih k data secara random pada sekumpulan n data sebagai centroid

Missal dipilih k = 2, dan dipilih data (1,1) sebagai centroid1 (C1) dan (1,2) sebagai centroid2 (C2)

Iterasi 1

b. Hitung jarak antara centroid dengan setiap data non-centroid

Jarak data ke-3 terhadap C1 dan C2

$d(C1,3) = \sqrt{(1-4)^2 + (1-3)^2}$ = 3.605551275

$d(C2,3) = \sqrt{(2-4)^2 + (1-3)^2}$ = **2,828427125**

Jarak data ke 3 dengan C2 lebih kecil sehingga data ke 3 menjadi anggota cluster 2

Jarak data ke-4 terhadap C1 dan C2

$d(C1,4) = \sqrt{(1-5)^2 + (1-4)^2}$ = 5

$d(C2,4) = \sqrt{(2-5)^2 + (1-4)^2}$ = **4.242640687**

Jarak data ke 4 dengan C2 lebih kecil sehingga data ke 4 menjadi anggota cluster 2

c. Dapatkan anggota cluster dengan memilih data yang mempunyai jarak yang paling kecil antara data non centroid dengan centroid

Anggota cluster 1 adalah C1 / data ke 1 (1,1)

Anggota cluster 2 adalah C2 (1,2)/data ke 2 (2,1) , data ke 3 (4,3) dan data ke 4 (5,4)

d. Update centroid dengan cara menghitung rata-rata dari anggota cluster

Anggota cluster 1 hanya C1 / data ke 1 sehingga C1 / data ke 1 menjadi centroid

Anggota cluster 2 terdapat data ke 2 (1,2) , data ke 3 (4,3) dan data ke 4 (5,4)

Sehingga centroid yang baru adalah rata-rata dari data ke 2, ke 3, dan ke 4

$$C_x' = \frac{(2+4+5)}{3} = 3.666666667$$

$$C_y' = \frac{(2+3+4)}{3} = 2.666666667$$

C2 baru = (3.67 , 2.67)

Iterasi 2

b. Hitung jarak antara centroid dengan setiap data non-centroid

Jarak data ke – 2 dengan C1 dan C2

$d(C1,2) = \sqrt{(1-2)^2 + (1-1)^2}$ = **1**

$d(C2,2) = \sqrt{(3.67 - 2)^2 + (2.67 - 1)^2}$ = 1,799388785

Jarak data ke 1 dengan C1 lebih kecil sehingga data ke 1 yang awalnya berada dalam cluster 2 pada iterasi kedua data ke-2 berpindah menjadi anggota cluster 1

Jarak data ke-3 terhadap C1 dan C2

$d(C1,3) = \sqrt{(1-4)^2 + (1-3)^2}$ = 3,605551275

$d(C2,3) = \sqrt{(3.67 - 4)^2 + (2.67 - 3)^2}$ = **1,370328428**

Jarak data ke 3 dengan C2 lebih kecil sehingga data ke 3 tetap menjadi anggota cluster 2

Jarak data ke-4 terhadap C1 dan C2

$d(C1,4) = \sqrt{(1-5)^2 + (1-4)^2}$ = 5

$d(C2,4) = \sqrt{(3.67-5)^2 + (2.67-4)^2}$ = **1,880904038**

Jarak data ke 4 dengan C2 lebih kecil sehingga data ke 4 tetap menjadi anggota cluster 2

c. Dapatkan anggota cluster dengan memilih data yang mempunyai jarak yang paling kecil antara data non centroid dengan centroid

Anggota cluster 1 adalah data ke 1 (1,1) dan data ke 2 (2,1)

Anggota cluster 2 adalah data ke 3 (4,3) dan data ke 4 (5,4)

d. Update centroid dengan cara menghitung rata-rata dari anggota cluster

Anggota cluster 1 terdapat data ke 1 (1,1) dan data ke 2 (2,1)

Sehingga centroid yang baru adalah

$$C'_x = \frac{(1+2)}{2} = 1.5$$

$$C'_y = \frac{(1+1)}{2} = 1$$

C1 baru = (1.5, 1)

Anggota cluster 2 terdapat data ke 3 (4,3) dan data ke 4 (5,4)

Sehingga centroid yang baru adalah

$$C'_x = \frac{(4+5)}{2} = 4.5$$

$$C'_y = \frac{(3+4)}{2} = 3.5$$

C2 baru = (4.5 , 3.5)

Iterasi 2

Ulangi langkah b sampai d sampai tidak ada perubahan dari centroid

3. Elbow Method

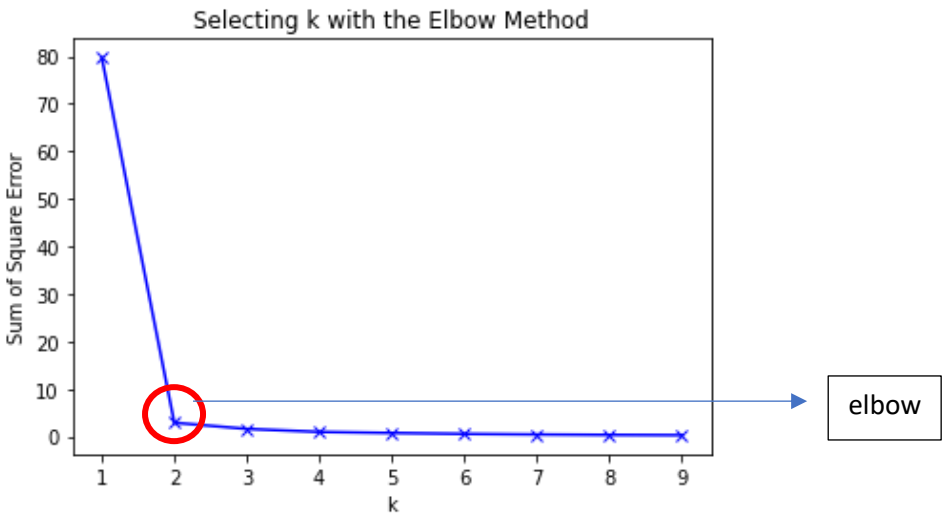
Langkah mendasar pada k-means adalah menentukan nilai "k" atau jumlah cluster yang optimal dimana data tapat dikelompokkan dengan baik. Elbow Method adalah salah satu metode yang popular untuk menentukan nilai "k" yang optimal. Pada elbow method dilakukan ploting nilai *cost function* yang dihasilkan oleh nilai k yang berbeda. Salah satu *cost function* yang digunakan dapat *Sum of Square Error (SSE)*

Formula Sum of Square Error (SSE)

$$J = \sum_{k=1}^{K} \sum_{i \in C_k} |x_i - \mu_k|^2$$

μ_k adalah centroid dari cluster k

Pada elbow method dilakukan ploting nilai *cost function* yang dihasilkan oleh nilai k yang berbeda sehingga akan terbentuk grafik antara nilai k dengan cost fungsion. Pada grafik elbow method akan terlihat bahwa maka semakin besar nilai k, nilai cost function akan semakin menurun dan setiap instance akan lebih dekat ke centroid masing-masing. Perhatikan gambar 12.2 pada nilai k dimana distorsi atau SSE menurun paling banyak disebut "elbow", yang artinya pada gambar 12.2 didapatkan k yang paling optimal adalah k=2.



Gambar 12. 2 Grafik elbow method

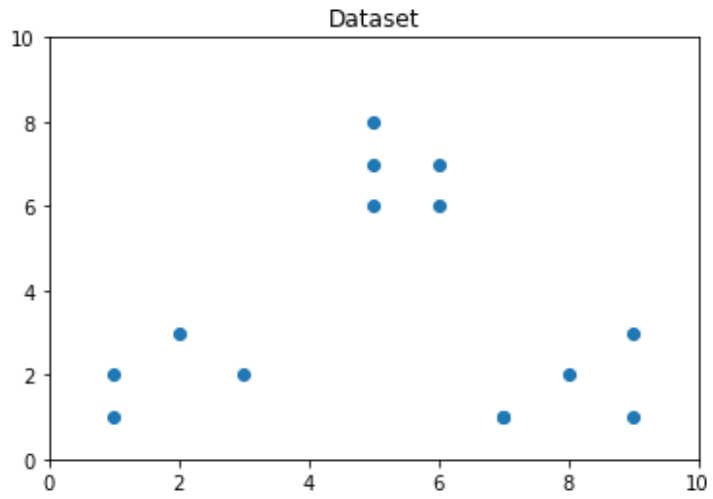
Kode program

```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt

# Creating the data
x1 = np.array([1, 2, 3, 1, 5, 6, 5, 5, 6, 7, 8, 9, 7, 9])
x2 = np.array([1, 3, 2, 2, 8, 6, 7, 6, 7, 1, 2, 1, 1, 3])
X = np.array(list(zip(x1, x2))).reshape(len(x1), 2)

# Visualizing the data
plt.plot()
plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title('Dataset')
plt.scatter(x1, x2)
plt.show()
```

Output dari ploting data



Dari visualisasi di atas, dapat dilihat bahwa jumlah cluster yang optimal seharusnya ada pada k=3. Namun memvisualisasikan data saja tidak selalu dapat memberikan k yang optimal karena data pada dunia nyata belum tentu dapat telihat kelompok data / cluster data.

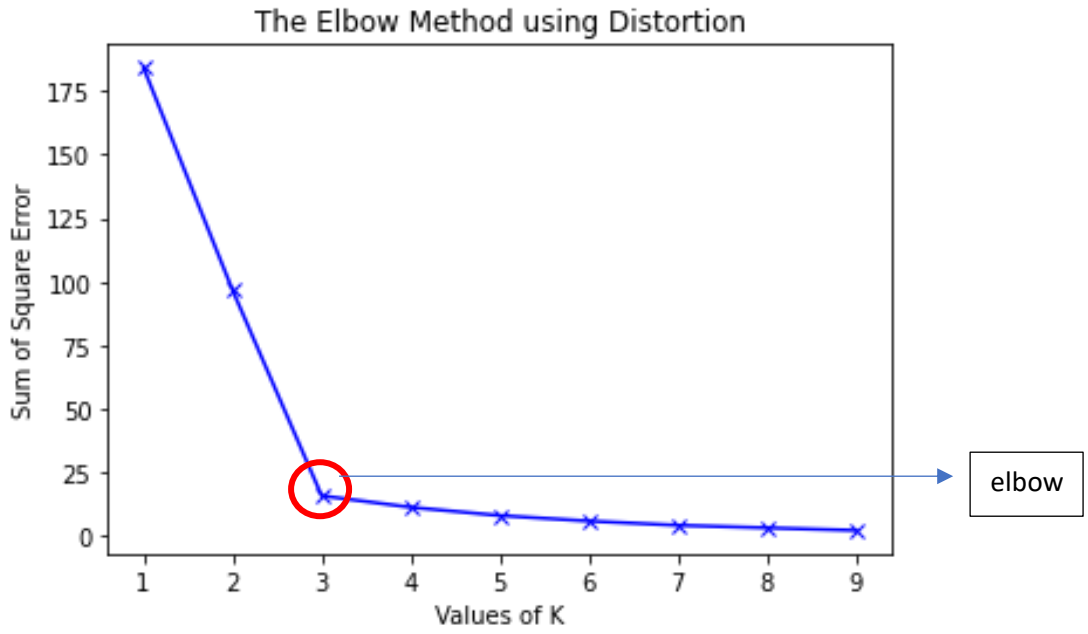
```
distortions = []
K = range(1, 10)
print(X.shape[0])
for k in K:
    # Building and fitting the model
    kmeanModel = KMeans(n_clusters=k).fit(X)
    kmeanModel.fit(X)

    distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_)**2, axis=1)))

plt.plot(K, distortions, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Sum of Square Error')
plt.title('The Elbow Method using Distortion')
plt.show()
```

Pada kode program diatas distorsi yang digunakan adalah *Sum of Square Error (SSE)* dan dipatkan hasil bahwa pada k=3 disebut sebagai elbow dimana nilai SSE menurun paling banyak dan menuju datar.

Output :



4. Evaluasi Cluster dengan silhouette coefficient

Silhouette Coefficient yang dikenalkan oleh (Rousseeuw, 1987) merupakan salah satu cara untuk mengevaluasi kualitas cluster yang dihasilkan. Selain itu silhouette coefficient juga mengindikasikan derajat kepemilikan setiap objek yang berada di dalam cluster. Data Oj yang berada pada cluster memiliki rentang nilai Silhouette antara -1 sampai 1. Semakin dekat nilai

silhouette ke 1 maka semakin tinggi derajat Oj di dalam cluster. Pada persamaan 12.2 dan 12.3 merupakan perhitungan nilai Silhouette (s(i)) untuk setiap data.

$$s = \frac{b(i) - a(i)}{max(a(i), b(i))}$$

$$b = min_{cj \neq a}\, d(i, C_j)$$

Dimana a(i) adalah jarak data (instance) i terhadap seluruh data yang ada di cluster tempat i berada disebut juga cluster internal. Sedangkan b adalah jarak data (instance) i terhadap seluruh dokumen yang ada cluster selain cluster internal (cluster external). Selanjutnya setiap cluster yang telah dihitung nilai s akan dihitung nilai rata-rata dari s. Perhitungan ini lebih dikenaldengan nama Average Silhouette Width (ASW). Range nilai ASW dapat dibagi menjadi empat kriteria yaitu:

1. Sangat baik : range (0,71 ≤ ASW < 1)
2. Baik : range (0,51 ≤ ASW < 0,71)
3. Cukup baik : range (0,26 ≤ ASW < 0,51)
4. Kurang baik : range (ASW < 0,26)

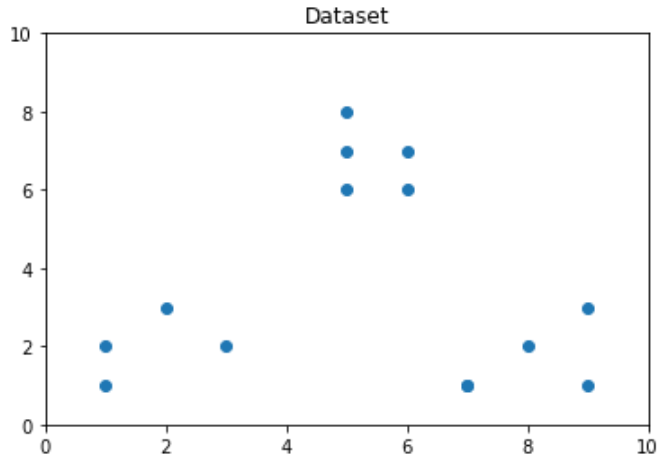
Kode program

```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt

# Creating the data
x1 = np.array([1, 2, 3, 1, 5, 6, 5, 5, 6, 7, 8, 9, 7, 9])
x2 = np.array([1, 3, 2, 2, 8, 6, 7, 6, 7, 1, 2, 1, 1, 3])
X = np.array(list(zip(x1, x2))).reshape(len(x1), 2)

# Visualizing the data
plt.plot()
plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title('Dataset')
plt.scatter(x1, x2)
plt.show()
```

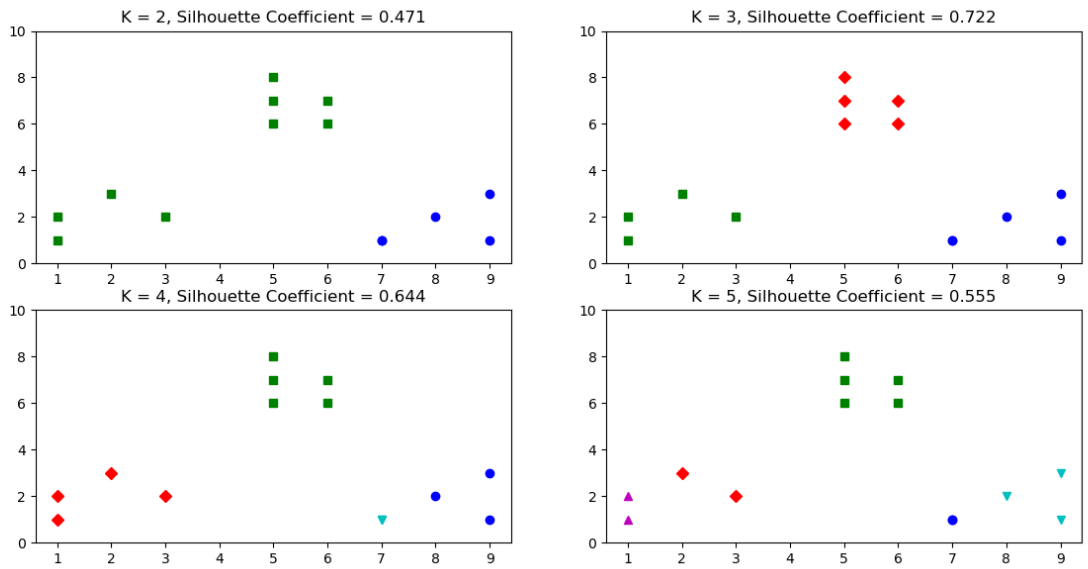
Output dari ploting data




```
plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title('Instances')
plt.scatter(x1, x2)

plt.subplot(3, 2, 1)
colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'b']
markers = ['o', 's', 'D', 'v', '^', 'p', '*', '+']
tests = [2, 3, 4, 5, 8]
subplot_counter = 1
for t in tests:
    subplot_counter += 1
    plt.subplot(3, 2, subplot_counter)
    kmeans_model = KMeans(n_clusters=t).fit(X)
    for i, l in enumerate(kmeans_model.labels_):
        plt.plot(x1[i], x2[i], color=colors[l], marker=markers[l], ls='None')
    plt.xlim([0, 10])
    plt.ylim([0, 10])
    plt.title('K = %s, Silhouette Coefficient = %.03f' % (
        t, metrics.silhouette_score(X, kmeans_model.labels_, metric='euclidean')))
plt.show()
```

Output :



Dari dataset yang digunakan pada kode program dapat dilihat bahwa jumlah cluster yang optimal ada pada k=3 yang artinya dari dataset akan terbentuk tiga cluster. Silhouette Coefficient yang dihasilkan adalah

K=2, Silhouette Coefficient = 0.471

K=3, Silhouette Coefficient = 0.722

K=4, Silhouette Coefficient = 0.644

K=5, Silhouette Coefficient = 0.555

Pada k=3 didapatkan nilai Silhouette Coefficient tertinggi sehingga dapat dikatakan bahwa k=3 adalah k yang paling optimal dengan hasil cluster yang paling baik dibandingkan k yang lainnya.