LAPORAN PRAKTIKUM KECERDASAN BUATAN
"AI – Clustering"

AGUNG ALDI PRASETYA (3122552803)

PJJ D3 TEKNIK INFORMATIKA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
2023

Assignment

1. dataset ← heart.csv, dan tampilkan
2. data ← normalisasi dengan min-max(0-1)
3. cluster ← lakukan clustering pada data menggunakan K-means (k=2)
4. cluster ← lakukan clustering pada data dengan Single, Average, Complete Linkage, dengan k=2
5. Lakukan untuk setiap jumlah atribut di langkah ke-4 : cluster_i[1-10], cluster_val[1-10] ← lakukan clustering pada data dengan atribut yang paling berpengaruh dengan K-Means, dengan k=3, sebanyak 10 kali. Setiap kali selesai clustering, lakukan cluster analysis dengan SSE.
6. cluster ← ambil cluster_i yang mempunyai cluster_val terkecil

Jawaban:

1. dataset -> heart.csv, dan tampilkan
   a. import

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans, AgglomerativeClustering
from sklearn.metrics import pairwise_distances
import numpy as np
from scipy.cluster.hierarchy import linkage
from scipy.spatial.distance import cdist
import warnings
```

   b. code

```
# 1. Baca dataset heart.csv
dataset = pd.read_csv('heart.csv')
print("Dataset:")
print(dataset.head())
```

   c. output

```
Dataset:
   age  sex  cp  trestbps  chol  fbs  ...  exang  oldpeak  slope  ca  thal  target
0   63    1   3       145   233    1  ...      0      2.3      0   0     1       1
1   37    1   2       130   250    0  ...      0      3.5      0   0     2       1
2   41    0   1       130   204    0  ...      0      1.4      2   0     2       1
3   56    1   1       120   236    0  ...      0      0.8      2   0     2       1
4   57    0   0       120   354    0  ...      1      0.6      2   0     2       1
```

2. data -> normalisasi dengan min-max(0-1)
    a. code

```python
# 2. Normalisasi data dengan min-max (0-1)
scaler = MinMaxScaler()
normalized_data = scaler.fit_transform(dataset)
normalized_df = pd.DataFrame(normalized_data, columns=dataset.columns)
print("\nData setelah dinormalisasi:")
print(normalized_df.head())
```

    b. output

```
Data setelah dinormalisasi:
        age  sex        cp  trestbps  ...  slope   ca      thal  target
0  0.708333  1.0  1.000000  0.481132  ...    0.0  0.0  0.333333     1.0
1  0.166667  1.0  0.666667  0.339623  ...    0.0  0.0  0.666667     1.0
2  0.250000  0.0  0.333333  0.339623  ...    1.0  0.0  0.666667     1.0
3  0.562500  1.0  0.333333  0.245283  ...    1.0  0.0  0.666667     1.0
4  0.583333  0.0  0.000000  0.245283  ...    1.0  0.0  0.666667     1.0
```

3. cluster -> lakukan clustering pada data menggunakan K-means (k=2)
    a. code

```python
# 3. Melakukan clustering menggunakan K-means (k=2)
kmeans = KMeans(n_clusters=2, random_state=0 , n_init=10)
kmeans.fit(normalized_df)
cluster_labels = kmeans.labels_
cluster_centers = kmeans.cluster_centers_
print("\nHasil clustering dengan K-means (k=2):")
print(cluster_labels)
```

    b. output

```
Hasil clustering dengan K-means (k=2):
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0]
```

4. cluster -> lakukan clustering pada data dengan Single, Average,Complete Linkage, dengan k=2
    a. code

```python
# 4. Melakukan clustering dengan Single, Average, dan Complete Linkage (k=2)
linkage_methods = ['single', 'average', 'complete']
for method in linkage_methods:
    linkage_clusters = AgglomerativeClustering(n_clusters=2, linkage=method)
    linkage_labels = linkage_clusters.fit_predict(normalized_df)
    print(f"\nHasil clustering dengan {method.capitalize()} Linkage (k=2):")
    print(linkage_labels)
```

b. output

```
Hasil clustering dengan Single Linkage (k=2):
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 0 0]

Hasil clustering dengan Average Linkage (k=2):
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0]

Hasil clustering dengan Complete Linkage (k=2):
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 0]
```

5. Lakukan untuk setiap jumlah atribut di langkah ke-4 : cluster_i[1-10],cluster_val[1-10] -> lakukan clustering pada data dengan atribut yang paling berpengaruh dengan K-Means, dengan k=3, sebanyak 10 kali. Setiap kali selesai clustering, lakukan cluster analysis dengan SSE.

   a. Code

```python
# 5. Melakukan clustering dengan atribut yang paling berpengaruh menggunakan K-Means (k=3) sebanyak 10 kali
num_attributes = 10
cluster_i = []
cluster_val = []

for i in range(1, num_attributes + 1):
    attributes = normalized_df.nlargest(i, 'target').columns
    selected_data = normalized_df[attributes]

    sse_values = []
    for _ in range(10):
        kmeans = KMeans(n_clusters=3, random_state=0 , n_init=10)
        kmeans.fit(selected_data)
        sse = np.sum(np.min(cdist(selected_data, kmeans.cluster_centers_, 'euclidean'), axis=1))
        sse_values.append(sse)

    best_sse_index = np.argmin(sse_values)
    cluster_i.append(best_sse_index + 1)
    cluster_val.append(sse_values[best_sse_index])

print("\nHasil clustering dengan atribut yang paling berpengaruh menggunakan K-Means (k=3):")
print("cluster_i:", cluster_i)
print("cluster_val:", cluster_val)
```

b. Output

```
Hasil clustering dengan atribut yang paling berpengaruh menggunakan K-Means (k=3):
cluster_i: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
cluster_val: [263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046, 263.98096959213046]
```

6. cluster -> ambil cluster_i yang mempunyai cluster_val terkecil

a. code

```python
# 6. Mengambil cluster_i dengan cluster_val terkecil
smallest_val_index = np.argmin(cluster_val)
smallest_cluster_i = cluster_i[smallest_val_index]
print("\nCluster_i dengan cluster_val terkecil:", smallest_cluster_i)
```

b. output

```
Cluster_i dengan cluster_val terkecil: 1
```