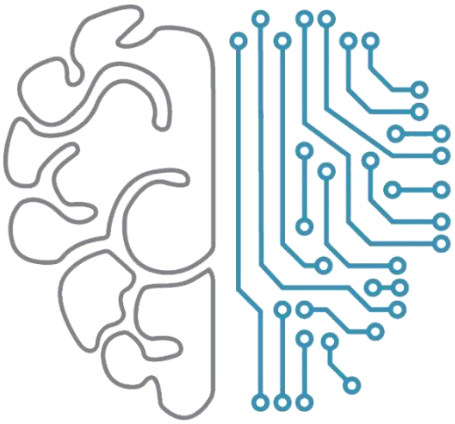
LAPORAN PRAKTIKUM

PEMBELAJARAN MESIN



NAMA : Ichsan Budiman Putra

NIM : 202131153

KELAS : B

DOSEN : Dr. Dra. DWINA KUSWARDANI, M.Kom.

No.PC : 02

ASISTEN : 1. ARIFINA RESYUANTI

2. HAYA Q. LUTHFIYANINGSIH

3. VIANA SALSABILA FAIRUZ SYAHLA

4. MUHAMMAD HANIEF FEBRIANSYAH

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2023

Penjelasan Praktikum

Library

```
In [28]: #Ichsan Budiman Putra
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.cluster.hierarchy as sch
```

import pandas as pd: Mengimpor pustaka Pandas untuk manipulasi dan analisis data tabular.

import numpy as np: Mengimpor pustaka NumPy untuk operasi matematika dan manipulasi array multidimensional.

import matplotlib.pyplot as plt: Mengimpor modul pyplot dari Matplotlib untuk membuat visualisasi grafis, seperti plot.

import scipy.cluster.hierarchy as sch: Mengimpor modul hierarchy dari Scipy, yang digunakan untuk analisis hierarki klaster.

Read Dataset

```
In [29]: data = pd.read_csv("sales.csv")
data
```

Out[29]:

	Product_Code	W0	W1	W2	W3	W4	W5	W6	W7	W8	...	Normalized 42	Normalized 43	Normalized 44	Normalized 45	Normalized 46	Normalized 47	Normalized 48	Norm
0	P1	11	12	10	8	13	12	14	21	6	...	0.06	0.22	0.28	0.39	0.50	0.00	0.22	
1	P2	7	6	3	2	7	1	6	3	3	...	0.20	0.40	0.50	0.10	0.10	0.40	0.50	
2	P3	7	11	8	9	10	8	7	13	12	...	0.27	1.00	0.18	0.18	0.36	0.45	1.00	
3	P4	12	8	13	5	9	6	9	13	13	...	0.41	0.47	0.06	0.12	0.24	0.35	0.71	
4	P5	8	5	13	11	6	7	9	14	9	...	0.27	0.53	0.27	0.60	0.20	0.20	0.13	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
806	P815	0	0	1	0	0	2	1	0	0	...	0.00	0.33	0.33	0.00	0.00	0.33	0.00	
807	P816	0	1	0	0	1	2	2	6	0	...	0.43	0.43	0.57	0.29	0.57	0.71	0.71	
808	P817	1	0	0	0	1	1	2	1	1	...	0.50	0.00	0.00	0.50	0.50	0.00	0.00	
809	P818	0	0	0	1	0	0	0	0	1	...	0.00	0.00	0.00	0.50	0.50	0.00	0.00	
810	P819	0	1	0	0	0	0	0	0	0	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	

811 rows × 107 columns

Data dari file CSV dengan nama "sales.csv" dibaca dan dimuat ke dalam sebuah objek DataFrame menggunakan pustaka pandas. Setelah data dimuat, perintah "data" digunakan untuk menampilkan isi dari DataFrame tersebut, memberikan gambaran ringkas mengenai struktur dan konten data penjualan yang mungkin terdapat dalam file CSV tersebut.

memilih data dari atribut dataset untuk digunakan sebagai atribut x

In [30]:
```
data = data[["W1", "W2"]] #balance x1; purchases= 2x
data.head(10)
```

Out[30]:

	W1	W2
0	12	10
1	6	3
2	11	8
3	8	13
4	5	13
5	3	2
6	8	3
7	6	10
8	9	10
9	19	19

Kemudian lakukan pemilihan kolom pada DataFrame yang telah disimpan dalam variabel "data". Kolom yang dipilih untuk dianalisis hanya "W1" dan "W2", dan DataFrame yang telah dimodifikasi disimpan kembali dalam variabel "data". Operasi ini mungkin dilakukan untuk fokus pada dua kolom tertentu yang berkaitan dengan analisis tertentu, seperti balance (keseimbangan) di "W1" dan "W2". Selanjutnya, ditampilkan sepuluh baris pertama dari DataFrame yang telah dimodifikasi dengan menggunakan perintah "data.head(10)".

Melihat Ringkasan Statistik

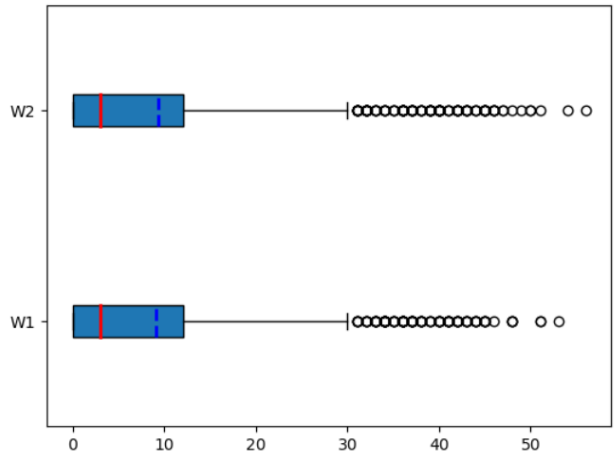
In [31]:
```
data.describe()
```

Out[31]:

	W1	W2
count	811.000000	811.000000
mean	9.129470	9.389642
std	12.564766	13.045073
min	0.000000	0.000000
25%	0.000000	0.000000
50%	3.000000	3.000000
75%	12.000000	12.000000
max	53.000000	56.000000

Kode di atas digunakan untuk menghasilkan ringkasan statistik deskriptif dari DataFrame yang telah disimpan dalam variabel "data". Hasilnya mencakup berbagai metrik statistik seperti rata-rata (mean), standar deviasi (std), nilai minimum (min), kuartil (25%, 50%, 75%), dan nilai maksimum (max) untuk setiap kolom numerik dalam DataFrame.

In [32]:
```
fig, ax = plt.subplots()
ax.boxplot(data,
           vert = False,
           showmeans = True,
           meanline = True,
           labels = ("W1", "W2"),
           patch_artist = True,
           medianprops = {"linewidth" : 2, "color" : "red"},
           meanprops = {"linewidth" : 2, "color" : "blue"})
plt.show()
```



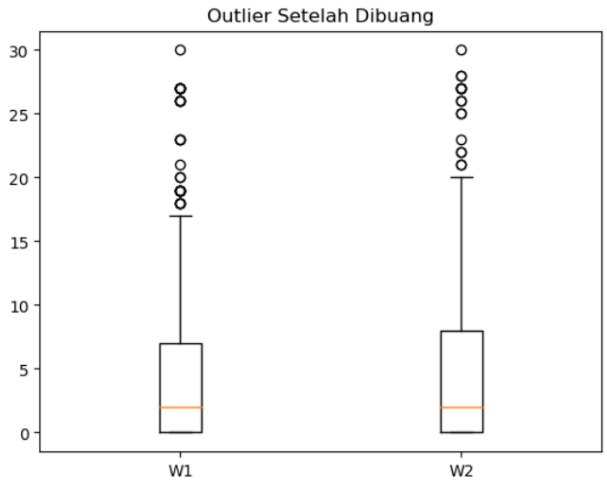
Kode di atas menggunakan Matplotlib untuk membuat diagram boxplot dari data yang tersimpan dalam DataFrame "data". Dua kotak (box) horizontal mewakili distribusi data untuk kolom "W1" dan "W2". Diagram ini mencakup garis median yang ditandai dengan warna merah, serta garis rata-rata yang ditandai dengan warna biru. Pengaturan vert = False membuat boxplot horizontal, showmeans = True menampilkan titik-titik yang menunjukkan nilai rata-rata, dan labels = ("W1", "W2") memberikan label pada sumbu y.

Membuang Outlier

In [33]:
```
#q1, q3, dan IQR
kolom = ["W1", "W2"]

Q1 = data[kolom].quantile(0.25)
Q3 = data[kolom].quantile(0.75)
IQR = Q3-Q1
data = data[~((data[kolom]<(Q1 - 1.5 * IQR)) |
              (data[kolom]>(Q3 + 1.5 * IQR))).any(axis = 1)]

plt.boxplot(data[kolom])
plt.xticks([1,2], kolom)
plt.title("Outlier Setelah Dibuang")
plt.show()
```



Dilakukan perhitungan kuartil pertama (Q1), kuartil ketiga (Q3), dan rentang antar kuartil (IQR) untuk kolom "W1" dan "W2" dalam DataFrame. Selanjutnya, data yang berada di luar batas bawah (Q1 - 1.5 IQR) dan batas atas (Q3 + 1.5 IQR) dihapus (outlier removal) dari DataFrame menggunakan operasi boolean masking. Diagram boxplot kemudian digambarkan ulang untuk menunjukkan distribusi data yang telah dimodifikasi setelah penghapusan outlier.

```
In [35]: data.info()

<class 'pandas.core.frame.DataFrame'>
Index: 692 entries, 0 to 810
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   W1      692 non-null    int64
 1   W2      692 non-null    int64
dtypes: int64(2)
memory usage: 16.2 KB
```

Perintah `data.info()` digunakan untuk memberikan informasi terperinci tentang DataFrame, seperti tipe data, jumlah nilai non-null, dan penggunaan memori. DataFrame ini memiliki 692 entri (baris) dari indeks 0 hingga 810, dengan dua kolom: "W1" dan "W2". Setiap kolom memiliki 692 nilai non-null, menandakan bahwa tidak ada nilai null atau missing values dalam DataFrame ini. Kedua kolom ini memiliki tipe data integer (int64). Keseluruhan DataFrame menggunakan 16.2 kilobyte (KB) dari memori komputer.

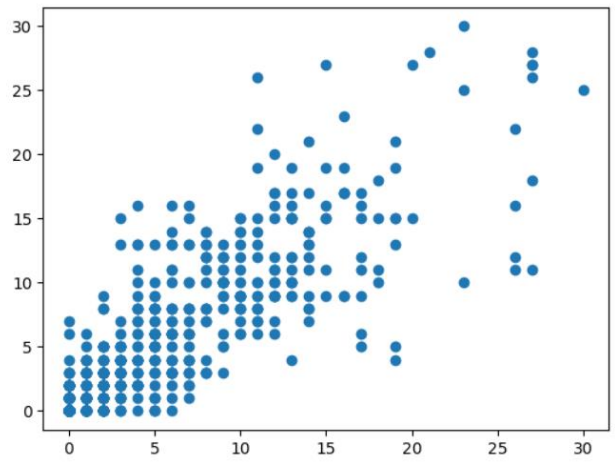
Membuat nilai x

```
In [36]: x_array = np.array (data)
```

Code digunakan untuk mengonversi DataFrame data menjadi sebuah array NumPy, dan hasilnya disimpan dalam variabel x_array.

Visualisassi persebaran data

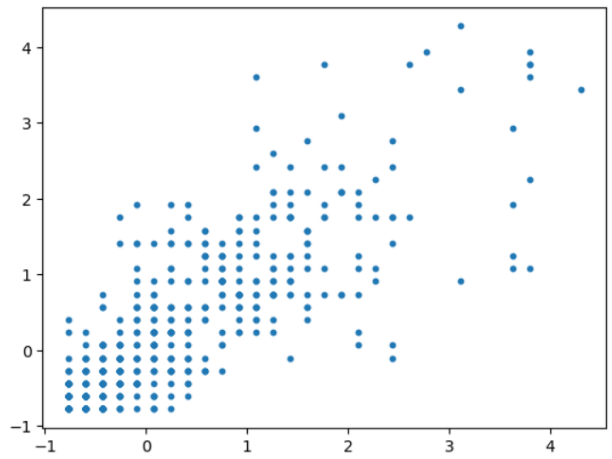
```
In [37]: plt.scatter(data.W1, data.W2)
         plt.show()
```



Untuk membuat scatter plot, dilakukan tampilan titik-titik data pada bidang dua dimensi. Pada scatter plot ini, sumbu x mewakili kolom "W1" dari DataFrame, sementara sumbu y mewakili kolom "W2". Setiap titik pada plot merepresentasikan satu baris dari data, dengan posisi titik menunjukkan nilai "W1" dan "W2" dari baris tersebut. Scatter plot berguna untuk memvisualisasikan hubungan atau pola antara dua variabel numerik.

Satandarisasi

```
In [38]: from sklearn.preprocessing import StandardScaler
         scaler = StandardScaler()
         x_scaled = scaler.fit_transform(x_array)
         plt.scatter(x_scaled[:,0], x_scaled[:,1], s=10)
         plt.show()
```



Kode ini digunakan untuk menampilkan scatter plot dari data yang telah di-scaling menggunakan StandardScaler. Dengan menggunakan StandardScaler, fitur-fiturnya diubah skala sehingga memiliki rata-rata nol dan standar deviasi satu. Scatter plot ini memberikan visualisasi dari distribusi data yang telah mengalami proses scaling.

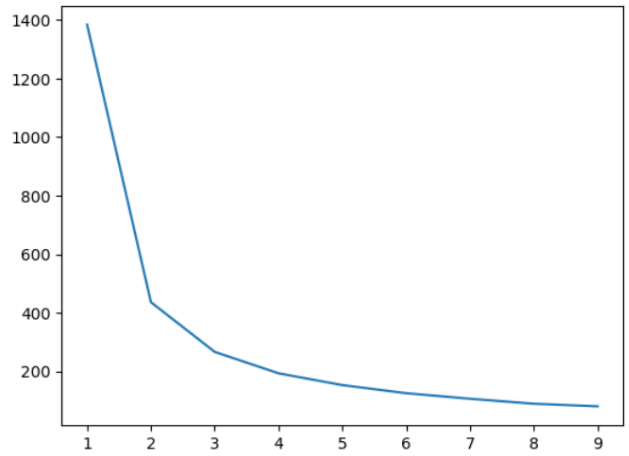
Cek Sum of Squared Errors (SSE) dari KMeans clustering

In [12]:
```
from sklearn.cluster import KMeans
sse = []
index = range(1,10)
for i in index :
    kmeans = KMeans(n_clusters=i, random_state=0, n_init=10)
    kmeans.fit(x_scaled)
    sse_ = kmeans.inertia_
    sse.append(sse_)
    print (i,sse_)
```

```
1 1384.0000000000014
2 436.10809145136136
3 266.26957213944615
4 193.13149929757355
5 152.79814460052006
6 125.17349262904683
7 106.54787525949338
8 89.2644991272542
9 79.42769146228859
```

List `sse` digunakan untuk menyimpan nilai SSE (Sum of Squared Errors) dan index dari 1 hingga 9 dalam konteks mengevaluasi kinerja algoritma KMeans untuk berbagai jumlah klaster. Parameter `n_init=10` menentukan jumlah kali inisialisasi yang berbeda akan dilakukan untuk memperbaiki konvergensi ke hasil terbaik. Nilai SSE dari model KMeans dihitung menggunakan atribut `inertia_`. Oleh karena itu, kode ini bertujuan untuk mengevaluasi nilai SSE untuk berbagai jumlah klaster dalam algoritma KMeans.

In [40]:
```
plt.plot(index, sse)
plt.show()
```



Kode untuk membuat grafik garis yang menampilkan nilai Sum of Squared Errors (SSE) terhadap jumlah klaster yang berbeda dari 1 hingga 9.

Pemodelan

```
In [14]: kmeans = KMeans (n_clusters = 3, random_state = 0, n_init=10)
         kmeans.fit(x_scaled)
```

Out[14]:

```
KMeans(n_clusters=3, n_init=10, random_state=0)
```

Dibentuk sebuah model KMeans dengan parameter yang telah ditentukan. Model KMeans ini siap digunakan untuk melakukan prediksi klaster pada data.

Melihat Cluster Pusat

```
In [42]: kmeans.cluster_centers_
```

```
Out[42]: array([[ 0.66561553,  0.70309988],
                [ 2.22042053,  2.21995313],
                [-0.59523924, -0.61091112]])
```

Digunakan untuk mengakses koordinat pusat (centroid) dari masing-masing klaster setelah model KMeans melakukan klasterisasi pada data.

Visualisasi Perseberan Data Setelah Clustering

```
In [43]: output = plt.scatter(x_scaled[:,0],x_scaled[:,1], s=10, c = kmeans.labels_) #Datanya

         centers = kmeans.cluster_centers_
         plt.scatter(centers[:,0], centers[:,1], c="red", s=10) #centroid
         plt.title("KMeans Clustering sales")
         plt.xlabel("W1")
         plt.xlabel("W2")
         plt.colorbar(output)
         plt.show()
```

Dilakukan visualisasi hasil klasterisasi yang telah dilakukan menggunakan KMeans. Tujuannya adalah untuk melihat distribusi data dalam ruang fitur yang terklaster, serta posisi pusat-pusat (centroid) dari masing-masing klaster. Visualisasi ini membantu dalam pemahaman pola kedekatan (proximity) antar datapoint berdasarkan hasil klasterisasi.

Evaluasi Model

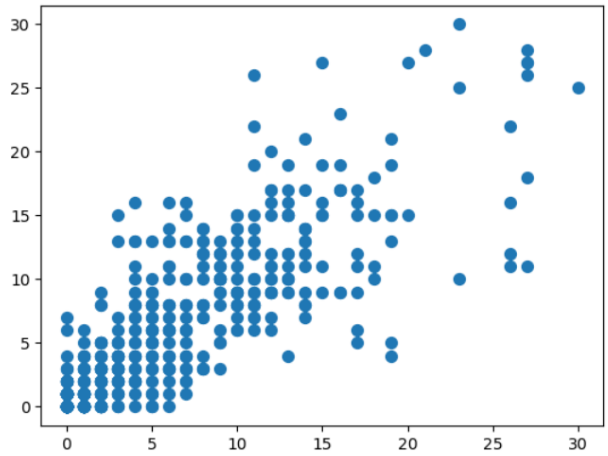
In [44]:
```
from sklearn.metrics import davies_bouldin_score
labels = kmeans.labels_
davies_bouldin_score(x_scaled, labels)
```

Out[44]: 0.7319476108924287

Dilakukan evaluasi terhadap kualitas klasterisasi yang telah dilakukan. Davies-Bouldin Score digunakan untuk mengukur seberapa baik klasterisasi telah dilakukan dengan mempertimbangkan kedekatan antara klaster yang terbentuk. Hasil dari Davies-Bouldin Score memberikan indikasi seberapa baik klasterisasi tersebut, di mana semakin rendahnya skor menandakan semakin baiknya klasterisasi.

Agglomerative Hierarchical Clustering

```
In [45]: plt.scatter(data['W1'], data['W2'], s=50)
         plt.show()
```



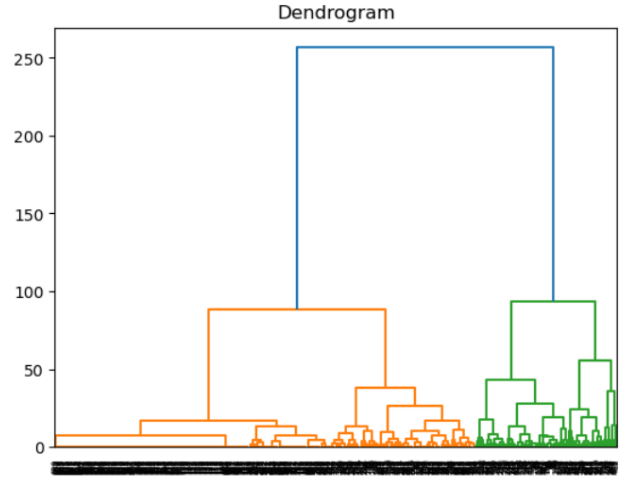
Scatter plot digunakan untuk memvisualisasikan hubungan atau pola antara dua variabel. plot menunjukkan sebaran data dari fitur 'W1' dan 'W2'.

```
In [46]: data = np.asarray(data[['W1', 'W2']])
         print(data)
```

```
[[12 10]
 [ 6  3]
 [11  8]
 ...
 [ 0  0]
 [ 0  0]
 [ 1  0]]
```

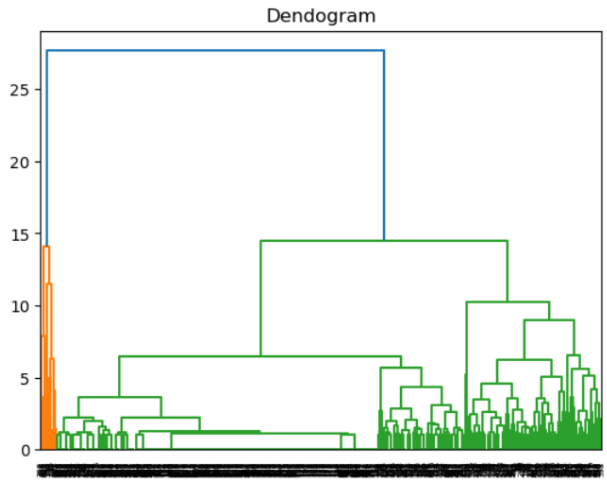
Untuk mengonversi kolom 'W1' dan 'W2' dari dataset menjadi sebuah array NumPy dua dimensi yang disimpan dalam variabel data. Hasil dari operasi ini adalah array dua dimensi NumPy yang berisi data dari kolom 'W1' dan 'W2' dari dataset. Setiap baris dalam array ini merepresentasikan satu entri data dari kolom 'W1' dan 'W2'.

```
In [47]: plt.title('Dendrogram')
         dendogram = sch.dendrogram(sch.linkage(data, method='ward'))
```



Untuk membuat dendrogram menggunakan metode hierarkis klastering (hierarchical clustering). Tujuan dari dendrogram adalah memberikan pemahaman visual tentang bagaimana datapoint dalam dataset tersebut terelasi satu sama lain dan bagaimana klaster-klaster terbentuk berdasarkan jarak antar datapoint.

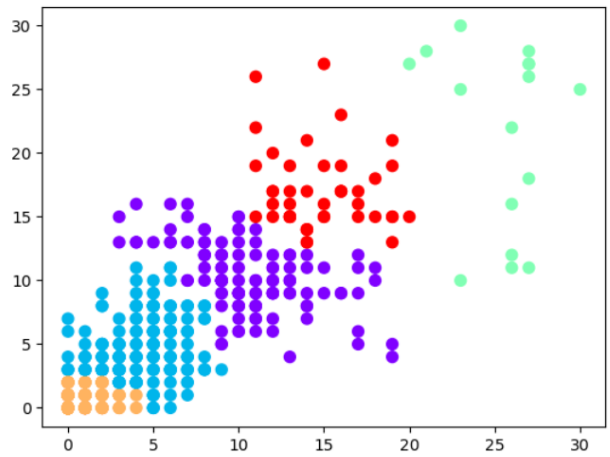
```
In [48]: plt.title('Dendogram')
         dendrogram = sch.dendrogram(sch.linkage(data, method='average'))
```



Menggunakan dendrogram untuk visualisasi dari metode hierarkis klastering. metode penggabungan klaster (linkage method).

```
In [49]: from sklearn.cluster import AgglomerativeClustering
         ac = AgglomerativeClustering(n_clusters=5)
         output_euclidean = ac.fit_predict(data)

         plt.scatter(data[:,0], data[:,1], c=output_euclidean, s=50, cmap='rainbow')
         plt.show()
```



Menggunakan algoritma Agglomerative Clustering dari sklearn.cluster untuk melakukan klasterisasi hierarkis terhadap data yang diberikan, dengan menunjukkan titik-titik data yang terkelompok ke dalam klaster yang berbeda dalam plot scatter.

```
In [51]: from sklearn.metrics import davies_bouldin_score
         print(davies_bouldin_score(data, output_euclidean))
         print(davies_bouldin_score(data, output_manhattan))

         0.8268624396485178
         0.8459686906234065
```

Menggunakan metrik Davies-Bouldin untuk mengevaluasi kualitas klasterisasi yang telah dilakukan dengan dua konfigurasi yang berbeda dari algoritma Agglomerative Clustering. Pertama, membandingkan hasil klasterisasi dengan metode pengukuran jarak Euclidean (output_euclidean) dan kedua, dengan metode pengukuran jarak Manhattan (output_manhattan).

Kesimpulan

Dalam evaluasi menggunakan metrik Davies-Bouldin Score, semakin rendah skornya menunjukkan kualitas klasterisasi yang lebih baik. Nilai Davies-Bouldin Score yang lebih rendah ditemukan pada metode Euclidean (0.8268624396485178) dibandingkan dengan metode Manhattan (0.8459686906234065). Oleh karena itu, berdasarkan perbandingan Davies-Bouldin Score, dapat disimpulkan bahwa metode Euclidean memberikan hasil yang sedikit lebih baik dalam konteks kualitas klasterisasi untuk data yang digunakan.