BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi Sistem

4.1.1 Langkah-Langkah menggunakan *Google Colab*

Pada bab ini akan memaparkan implementasi algoritma *K-Means Clustering* menggunakan *Machine Learning* dengan alat bantu dan *tools* pendukung yaitu *Python* dan *Google Colab*. Sebagai pendukung terhadap hasil perhitungan manual yang telah dijelaskan sebelumnya, data tersebut kemudian diuji menggunakan *Machine Learning*.

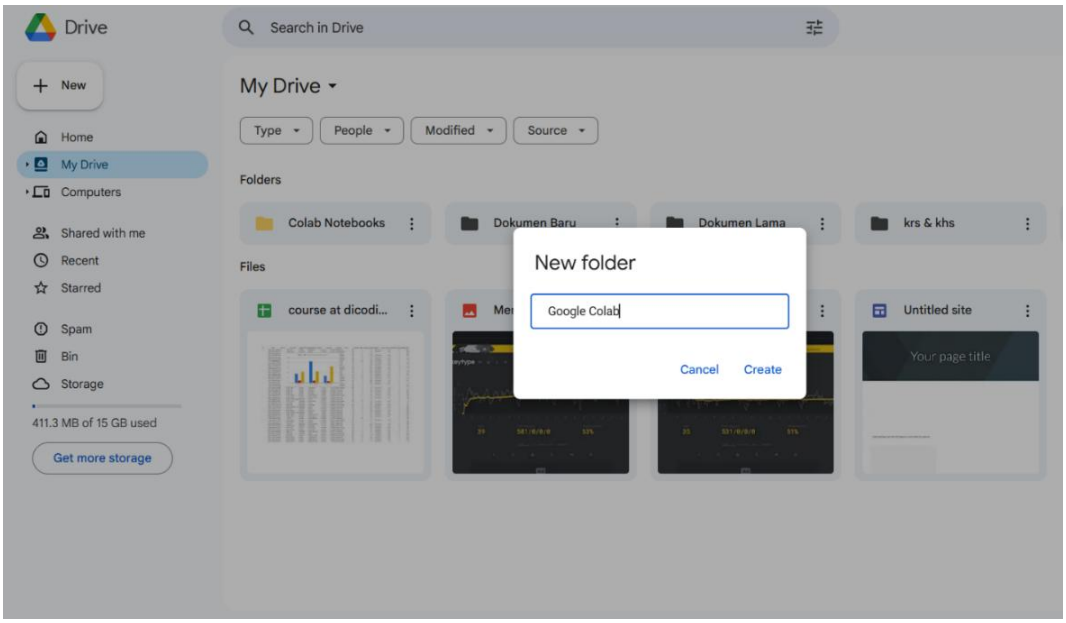
Adapun langkah-langkah menggunakan *Google Colab* adalah sebagai berikut [27]:

1. Membuat folder di *Google Drive*

   Langkah awal untuk menggunakan *Google Cola*b adalah dengan masuk ke akun *Google* dan mengakses *Google Drive*. Agar lebih mudah menemukan file, sebaiknya buat folder terlebih dahulu.

   a. Tekan ikon tambah '+' > New Folder

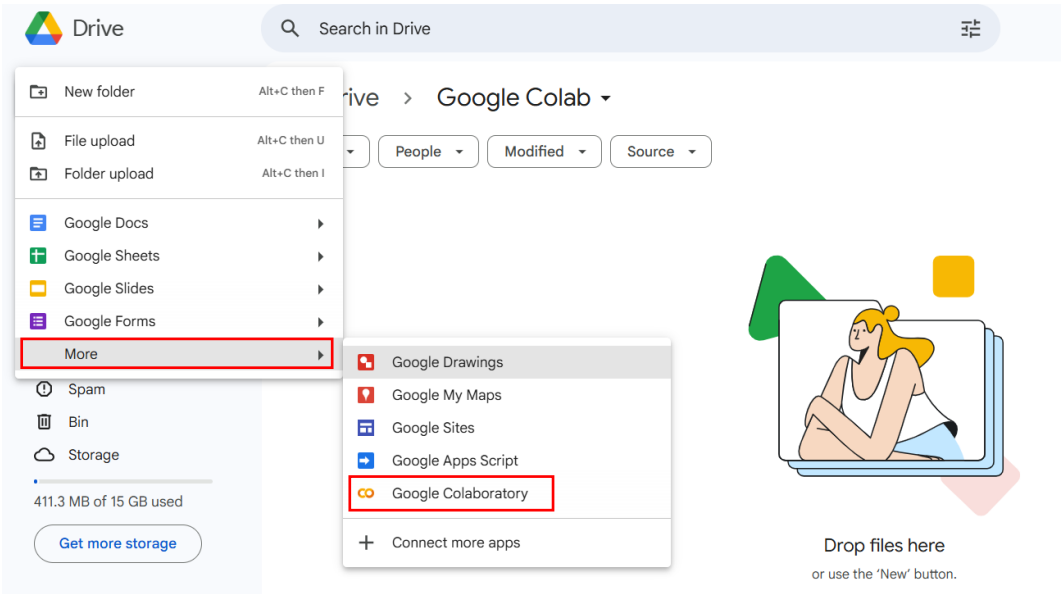
   b. Setelah itu, beri nama folder dan pilih 'Create'



Gambar 4.1 Membuat Folder Baru *Google Drive*

2. Membuat notebook di *Google Colab*

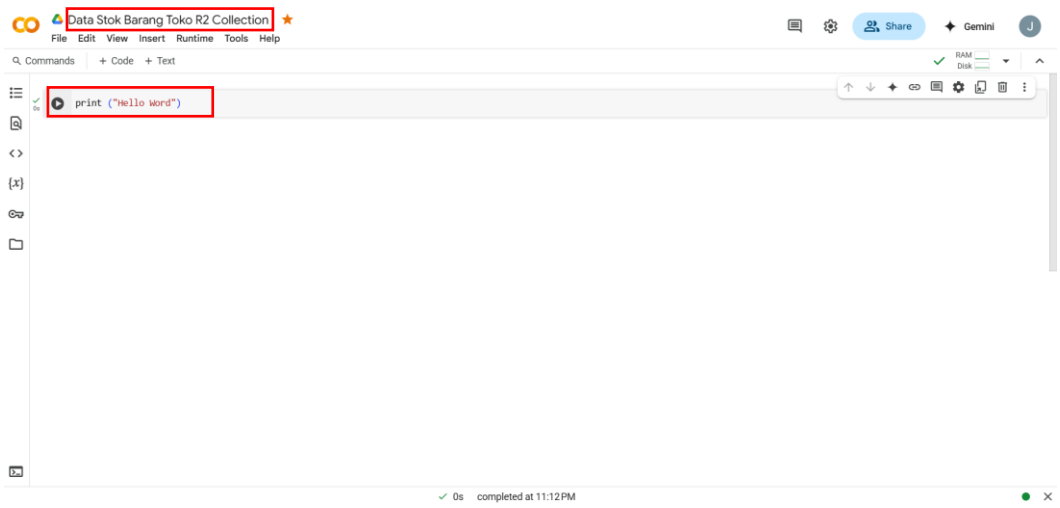
   Setelah folder sudah dibuat, buka folder itu dan kita akan segera membuat notebook di *Google Colab*. Untuk melakukannya, klik tanda tambah '+' > More > *Google Colaboratory*.



Gambar 4.2 Membuat Notebook *Google Colab*

3. Menulis kode program

   Jika notebook telah berhasil dibuat, akan muncul halaman editor seperti berikut.

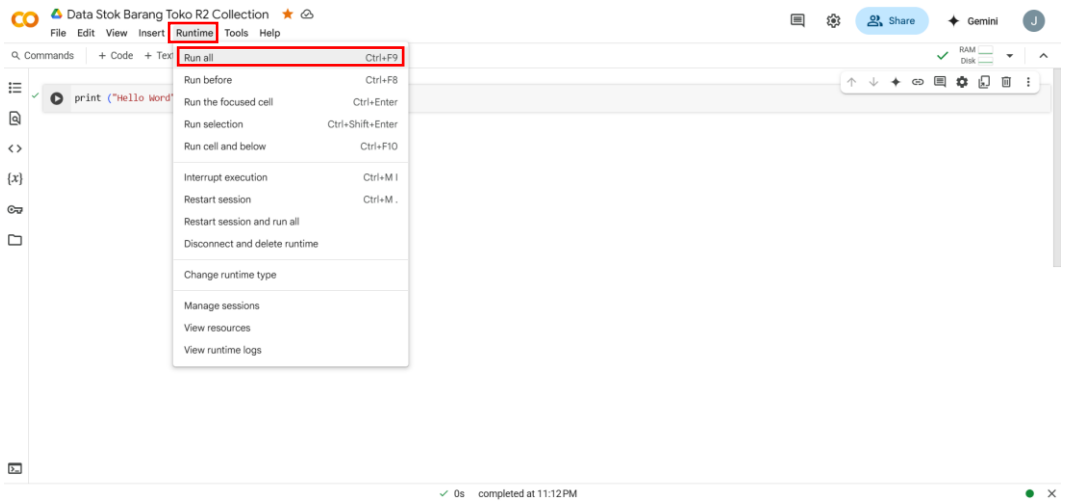


Gambar 4.3 Tampilan Notebook

Di sini, kita bisa langsung menulis kode program. Contohnya, seperti pada gambar di atas, kita akan menampilkan teks “Hello World”. Jangan lupa juga untuk mengganti nama file pada pojok kiri atas “Data Stok Barang Toko R2 Collection”.

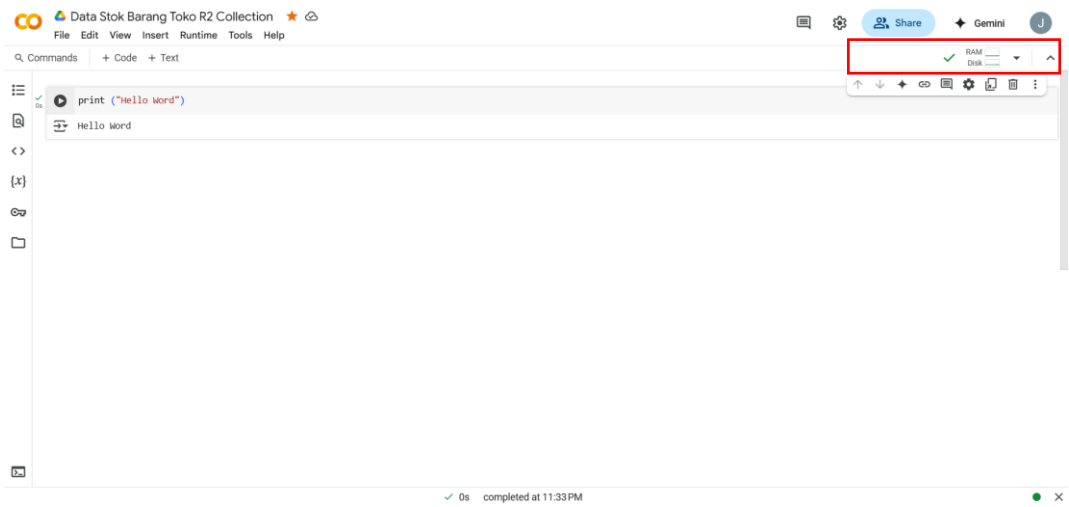
4. Menjalankan kode program

   Apabila sudah selesai menuliskan program, selanjutnya kita akan coba untuk menjalankan atau run kode tersebut. Caranya klik Runtime > Run All, seperti pada gambar 4.4 berikut ini.



Gambar 4.4 Tampilan Menjalankan Program

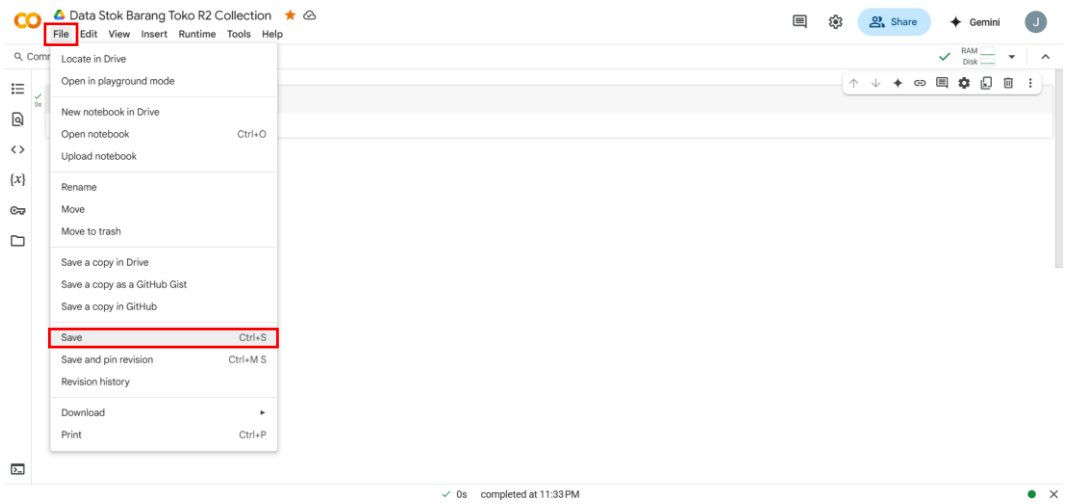
Jika kode yang tulis tidak ada yang error, maka *output* akan muncul di bawahnya, lalu di bagian kanan atas terdapat indikator *resource*, jika kode berhasil dijalankan maka tandanya akan bercentang hijau.



Gambar 4.5 Tampilan Program Berhasil

5. Menyimpan file notebook

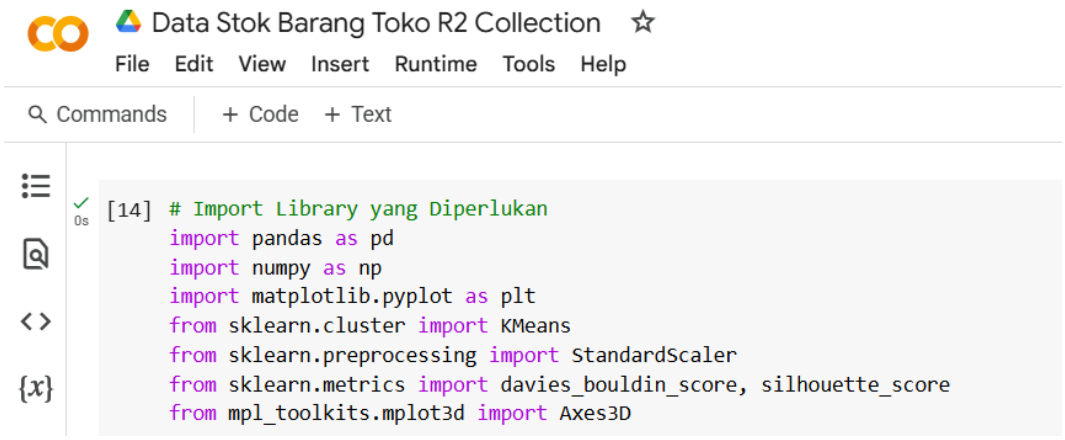
   Setelah program berhasil dijalankan dan tidak ada yang eror, file notebook bisa di simpan dengan cara klik file > save, maka file notebook sudah otomatis tersimpan di *Google Drive*.



Gambar 4.6 Tampilan Menyimpan File Notebook

4.1.2 Implementasi Algoritma *K-Means* di *Google Colab*

Langkah 1: *Import Library* yang Diperlukan

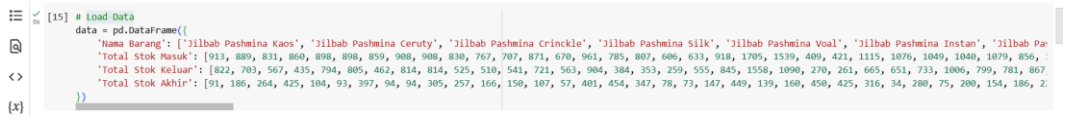


Gambar 4.7 *Import Library* yang Diperlukan

Pada tahap awal, dilakukan pemanggilan sejumlah *library* yang berfungsi untuk mendukung proses analisis data. pandas digunakan untuk mengelola dan memproses data dalam bentuk tabel. numpy membantu dalam perhitungan numerik. matplotlib.pyplot dipakai untuk membuat visualisasi grafik. KMeans dari sklearn.cluster digunakan untuk melakukan proses *clustering* dengan algoritma *K-*

Means. StandardScaler digunakan untuk menstandarisasi skala data agar hasil *clustering* lebih akurat. Sementara davies_bouldin_score dan silhouette_score digunakan untuk mengevaluasi kualitas *cluster*. Terakhir, Axes3D dari mpl_toolkits.mplot3d digunakan untuk menampilkan visualisasi *clustering* dalam bentuk tiga dimensi.

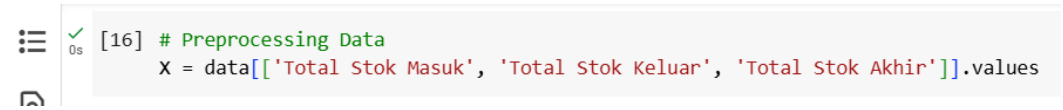
Langkah 2: *Load* Data



Gambar 4.8 Load Data

Pada tahap ini, data dimasukkan secara manual ke dalam program menggunakan fungsi pd.DataFrame() dari pustaka *pandas*. Data ini terdiri dari daftar nama barang beserta tiga atribut numerik utama, yaitu Total Stok Masuk, Total Stok Keluar, dan Total Stok Akhir. Seluruh informasi tersebut disimpan dalam sebuah struktur data bertipe *dataframe*, yang nantinya akan digunakan sebagai objek utama dalam proses analisis. Tujuan dari langkah ini adalah untuk menyiapkan data mentah yang akan dianalisis lebih lanjut melalui proses *clustering* menggunakan algoritma *K-Means*.

Langkah 3: *Preprocessing* Data

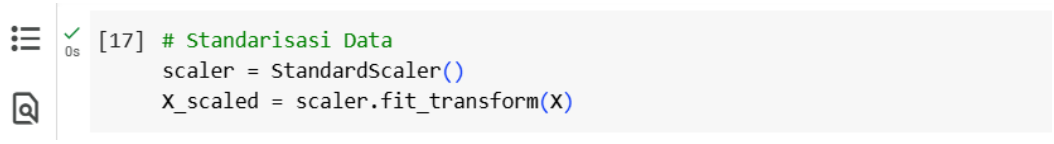


Gambar 4.9 *Preprocessing* Data

Pada tahap ini, dilakukan proses seleksi terhadap data yang akan dianalisis, yaitu hanya mengambil tiga atribut numerik utama: Total Stok Masuk, Total Stok

Keluar, dan Total Stok Akhir. Ketiga kolom ini dipisahkan dari data utama dan disimpan dalam sebuah variabel bernama X. Tujuannya adalah untuk mempersiapkan data numerik yang akan digunakan dalam proses klasterisasi. Dengan hanya menyimpan nilai-nilai dalam bentuk array (tanpa nama kolom), maka data siap untuk diproses lebih lanjut, seperti standarisasi dan pelatihan model.

Langkah 4: Standarisasi Data

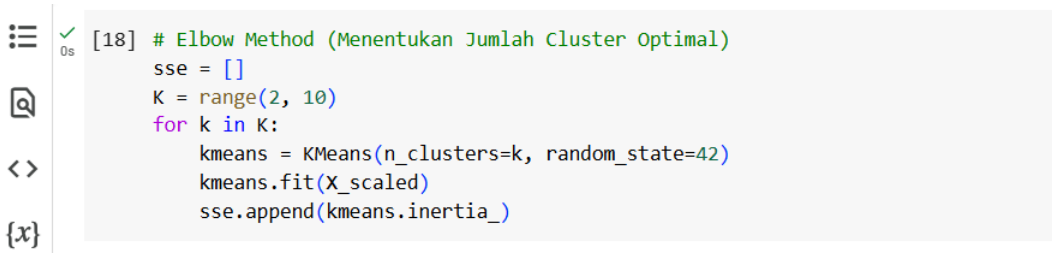


```
[17] # Standarisasi Data
     scaler = StandardScaler()
     X_scaled = scaler.fit_transform(X)
```

Gambar 4.10 Standarisasi Data

Langkah ini bertujuan untuk menyamakan skala dari setiap fitur numerik agar tidak ada satu variabel pun yang mendominasi proses klasterisasi hanya karena perbedaan skala. Proses ini dilakukan dengan bantuan fungsi StandardScaler() dari library *scikit-learn*. Metode ini akan mengubah setiap nilai dalam data X menjadi nilai yang memiliki rata-rata (mean) 0 dan standar deviasi 1. Hasil transformasi ini disimpan dalam variabel X_scaled dan akan digunakan pada tahapan klasterisasi berikutnya.

Langkah 5: *Elbow Method* (Menentukan Jumlah Cluster Optimal)



```
[18] # Elbow Method (Menentukan Jumlah Cluster Optimal)
     sse = []
     K = range(2, 10)
     for k in K:
         kmeans = KMeans(n_clusters=k, random_state=42)
         kmeans.fit(X_scaled)
         sse.append(kmeans.inertia_)
```

Gambar 4.11 *Elbow Method* (Menentukan Jumlah *Cluster* Optimal)

Pada tahap ini digunakan metode *Elbow* untuk membantu menentukan jumlah *cluster* yang paling sesuai dalam proses klasterisasi. Metode ini dilakukan dengan menghitung nilai SSE (*Sum of Squared Errors*) untuk berbagai jumlah *cluster*. Baris kode sse = [] digunakan untuk membuat list kosong sebagai tempat menyimpan nilai SSE dari masing-masing percobaan jumlah cluster.

Kemudian, dilakukan perulangan untuk jumlah *cluster* k dari 2 hingga 9 menggunakan range(2, 10). Di dalam perulangan tersebut, objek KMeans dibentuk dengan jumlah *cluster* k, lalu model dilatih menggunakan data yang telah distandarisasi (X_scaled). Nilai *inertia_* dari model, yaitu total jarak kuadrat dari setiap titik ke pusat clusternya (yang juga merupakan nilai SSE), disimpan dalam list sse. Nilai-nilai SSE ini nantinya akan digunakan untuk menggambar grafik *Elbow* pada langkah berikutnya, guna mengidentifikasi titik siku (*elbow point*) sebagai indikator jumlah *cluster* yang optimal.

Langkah 6: Plot *Elbow*

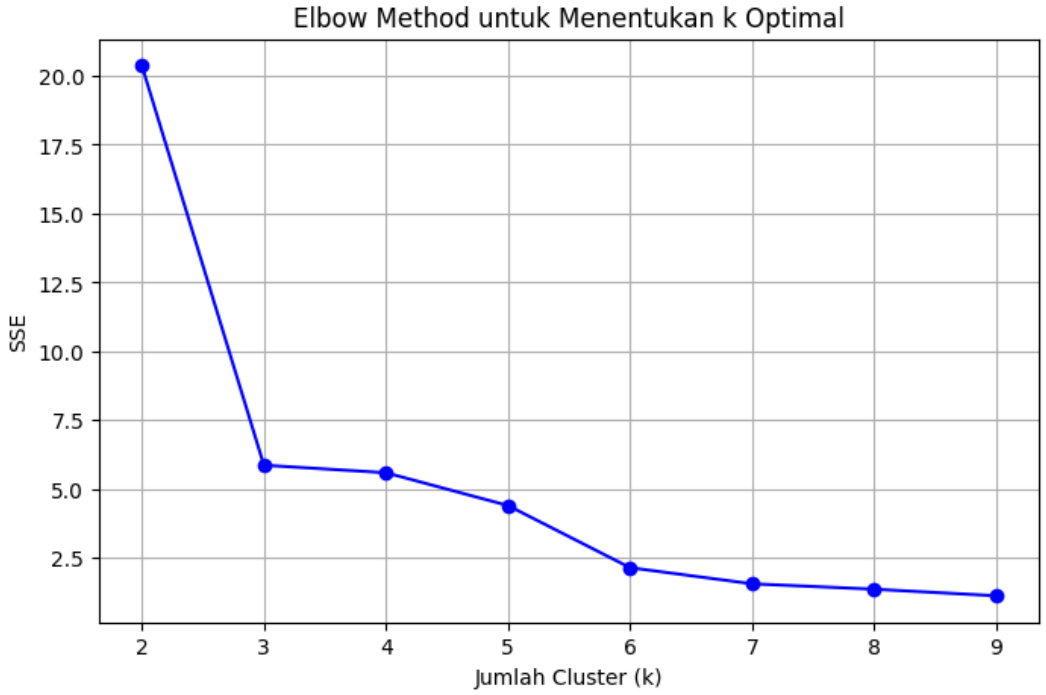
```
# Plot Elbow
plt.figure(figsize=(8,5))
plt.plot(K, sse, 'bo-')
plt.xlabel('Jumlah Cluster (k)')
plt.ylabel('SSE')
plt.title('Elbow Method untuk Menentukan k Optimal')
plt.grid(True)
plt.show()
```

Gambar 4.12 Plot *Elbow*

Setelah nilai SSE diperoleh dari berbagai jumlah *cluster* pada langkah sebelumnya, langkah ini bertujuan untuk memvisualisasikan nilai-nilai tersebut dalam bentuk grafik *Elbow*. Visualisasi ini membantu mengidentifikasi jumlah *cluster* yang paling sesuai secara visual. Sintaks plt.figure(figsize=(8,5)) digunakan untuk mengatur ukuran grafik agar tampil lebih jelas. Kemudian, plt.plot(K, sse,

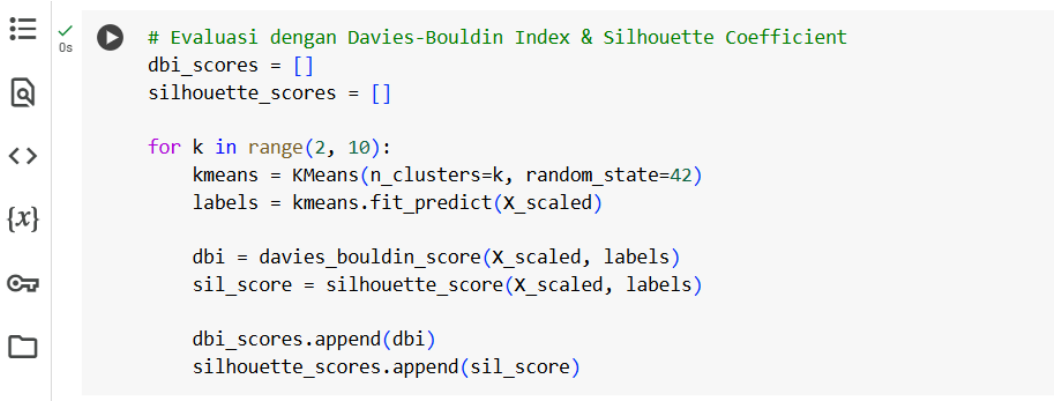
'bo-') menggambar grafik dengan sumbu horizontal sebagai jumlah *cluster* (K) dan sumbu vertikal sebagai nilai SSE. Huruf 'bo-' menunjukkan titik-titik berwarna biru yang dihubungkan oleh garis.

Label sumbu X dan Y ditambahkan melalui plt.xlabel() dan plt.ylabel(), sedangkan judul grafik ditentukan oleh plt.title(). plt.grid(True) menampilkan garis bantu (grid) agar grafik lebih mudah dibaca. Terakhir, plt.show() digunakan untuk menampilkan grafik secara utuh. Dari grafik ini, titik "siku" (*elbow point*) yang terlihat menjadi acuan dalam menentukan jumlah *cluster* yang optimal karena setelah titik tersebut, penurunan nilai SSE tidak lagi signifikan.



Gambar 4.13 Grafik *Elbow Method*

Langkah 7: Evaluasi dengan *Davies-Bouldin Index* & *Silhouette Coefficient*



```
# Evaluasi dengan Davies-Bouldin Index & Silhouette Coefficient
dbi_scores = []
silhouette_scores = []

for k in range(2, 10):
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(X_scaled)

    dbi = davies_bouldin_score(X_scaled, labels)
    sil_score = silhouette_score(X_scaled, labels)

    dbi_scores.append(dbi)
    silhouette_scores.append(sil_score)
```

Gambar 4.14 Evaluasi dengan *DBI* & *SC*

Pada tahap ini, dilakukan evaluasi kualitas hasil *clustering* untuk masing-masing jumlah *cluster* menggunakan dua metrik yang umum dipakai, yaitu *Davies-Bouldin Index (DBI)* dan *Silhouette Coefficient (SC)*. Tujuan dari langkah ini adalah untuk membantu memilih jumlah *cluster* yang memberikan hasil pemisahan data terbaik. dbi_scores = [] dan silhouette_scores = [] berfungsi untuk membuat dua list kosong yang akan digunakan menyimpan hasil evaluasi untuk setiap jumlah *cluster* dari 2 hingga 9.

Dalam perulangan for k in range(2, 10):, algoritma K-Means dijalankan untuk setiap nilai k (jumlah cluster) dari 2 sampai 9. kmeans = KMeans(n_clusters=k, random_state=42) membuat objek KMeans untuk masing-masing k. labels = kmeans.fit_predict(X_scaled) melakukan proses pelatihan (*training*) dan menghasilkan label *cluster* untuk setiap data setelah distandarisasi. davies_bouldin_score(X_scaled, labels) menghitung nilai DBI, di mana semakin kecil nilainya, maka hasil clustering dianggap semakin baik karena cluster lebih terpisah dan kompak. silhouette_score(X_scaled, labels) menghitung nilai Silhouette Coefficient, di mana semakin tinggi nilainya (mendekati 1), maka

kualitas cluster semakin baik. Nilai-nilai hasil evaluasi tersebut kemudian disimpan ke dalam list masing-masing melalui dbi_scores.append(dbi) dan silhouette_scores.append(sil_score).

Langkah ini sangat penting karena digunakan untuk mendukung hasil dari *Elbow Method* secara kuantitatif dan memberi gambaran seberapa baik data dipisahkan oleh masing-masing jumlah *cluster*.

Langkah 8: Plot *DBI* & *SC*

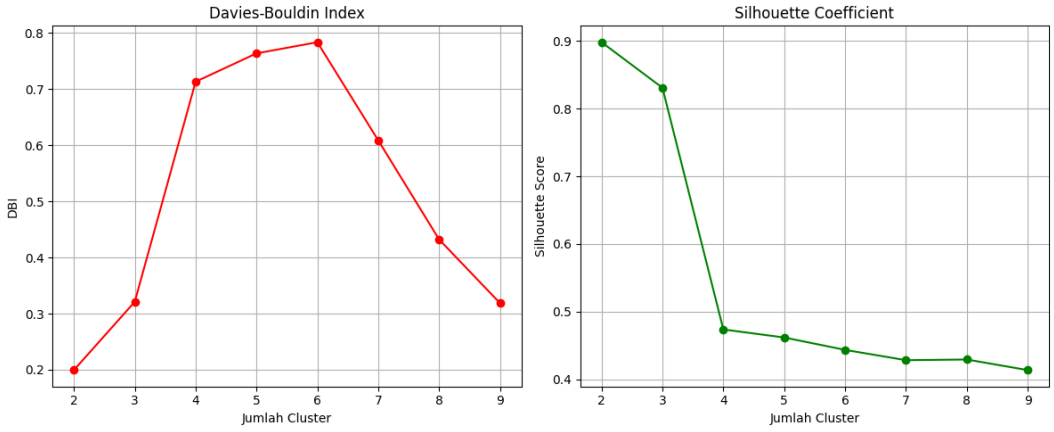
```
# Plot DBI dan SC
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(range(2, 10), dbi_scores, 'ro-')
plt.title('Davies-Bouldin Index')
plt.xlabel('Jumlah Cluster')
plt.ylabel('DBI')
plt.grid(True)

plt.subplot(1, 2, 2)
plt.plot(range(2, 10), silhouette_scores, 'go-')
plt.title('Silhouette Coefficient')
plt.xlabel('Jumlah Cluster')
plt.ylabel('Silhouette Score')
plt.grid(True)

plt.tight_layout()
plt.show()
```

Gambar 4.15 Plot DBI & SC

Setelah dilakukan evaluasi kualitas *clustering* pada langkah sebelumnya, hasilnya kemudian divisualisasikan dalam bentuk grafik agar lebih mudah dianalisis. Visualisasi ini membantu dalam memilih jumlah *cluster* yang memberikan hasil pemisahan data terbaik. plt.figure(figsize=(12, 5)) digunakan untuk mengatur ukuran keseluruhan gambar menjadi lebar 12 dan tinggi 5 agar grafik tampak proporsional dan mudah dibaca. plt.subplot(1, 2, 1) membagi area gambar menjadi 1 baris dan 2 kolom, lalu memilih bagian pertama (kiri) untuk menampilkan grafik *DBI*. plt.plot(range(2, 10), dbi_scores, 'ro-') menggambarkan

garis grafik *DBI* menggunakan titik-titik berwarna merah (‘ro-’ artinya red circle with line) berdasarkan jumlah *cluster* dari 2 hingga 9. plt.title('Davies-Bouldin Index'), plt.xlabel(), dan plt.ylabel() digunakan untuk memberi judul grafik serta label pada sumbu x dan y. plt.grid(True) menambahkan garis bantu pada latar belakang untuk memudahkan pembacaan nilai grafik. plt.subplot(1, 2, 2) memilih bagian kedua (kanan) dari area gambar untuk menampilkan grafik *Silhouette Coefficient*. plt.plot(range(2, 10), silhouette_scores, 'go-') menggambarkan grafik *Silhouette Score* menggunakan titik-titik hijau. plt.tight_layout() secara otomatis menyesuaikan tata letak agar tidak terjadi tumpang tindih antar elemen visual. plt.show() menampilkan grafik hasil visualisasi.

Melalui visualisasi ini, pengguna dapat melihat pola perubahan nilai *DBI* dan S*ilhouette Score* terhadap jumlah *cluster*, sehingga bisa lebih meyakinkan dalam menentukan jumlah *cluster* yang paling optimal untuk digunakan dalam analisis *clustering*.



Gambar 4.16 Grafik DBI & SC

Langkah 9: *Training* Model dengan *K-Means* (k = 3)

```
[22] # Training Model dengan K-Means (k = 3)
     kmeans = KMeans(n_clusters=3, init='k-means++', random_state=42)
     y_kmeans = kmeans.fit_predict(X_scaled)
```

Gambar 4.17 *Training* Model dengan *K-Means* (k = 3)

Setelah ditentukan bahwa jumlah *cluster* optimal adalah 3 berdasarkan hasil dari metode *Elbow* dan evaluasi sebelumnya, maka pada langkah ini dilakukan pelatihan (*training*) model *clustering* menggunakan algoritma *K-Means*. kmeans = KMeans(n_clusters=3, init='k-means++', random_state=42). Baris ini digunakan untuk membuat objek model *KMeans* dari library scikit-learn. n_clusters=3 menyatakan bahwa jumlah kelompok (cluster) yang diinginkan adalah tiga. init='k-means++' digunakan untuk mengatur metode awal pemilihan titik pusat (centroid), yang bertujuan mempercepat proses konvergensi dan meningkatkan hasil cluster. random_state=42 ditetapkan agar hasil clustering tetap konsisten ketika program dijalankan ulang (*reproducibility*). y_kmeans = kmeans.fit_predict(X_scaled), sintaks ini melakukan dua hal sekaligus: *fit* → Model dilatih menggunakan data yang telah dinormalisasi (X_scaled). *predict* → Data secara otomatis diklasifikasikan ke dalam salah satu dari tiga cluster berdasarkan kesamaan karakteristiknya. Hasil prediksi tersebut disimpan dalam variabel y_kmeans, yang berisi label cluster untuk masing-masing data.

Dengan langkah ini, setiap data barang dalam dataset telah dimasukkan ke dalam salah satu dari tiga kelompok berdasarkan kesamaan pola dari tiga variabel: Total Stok Masuk, Total Stok Keluar, dan Total Stok Akhir.

Langkah 10: Visualisasi *Clustering* dalam 3D

```
# Visualisasi Clustering dalam 3D
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')

ax.scatter(X_scaled[y_kmeans == 0, 0], X_scaled[y_kmeans == 0, 1], X_scaled[y_kmeans == 0, 2], s=100, c='red', label='Cluster 1')
ax.scatter(X_scaled[y_kmeans == 1, 0], X_scaled[y_kmeans == 1, 1], X_scaled[y_kmeans == 1, 2], s=100, c='blue', label='Cluster 2')
ax.scatter(X_scaled[y_kmeans == 2, 0], X_scaled[y_kmeans == 2, 1], X_scaled[y_kmeans == 2, 2], s=100, c='green', label='Cluster 3')

ax.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], kmeans.cluster_centers_[:, 2],
           s=300, c='yellow', marker='X', label='Centroids')

ax.set_xlabel('Total Stok Masuk')
ax.set_ylabel('Total Stok Keluar')
ax.set_zlabel('Total Stok Akhir')
ax.set_title('Visualisasi 3D Clustering K-Means')
ax.legend()
plt.show()
```

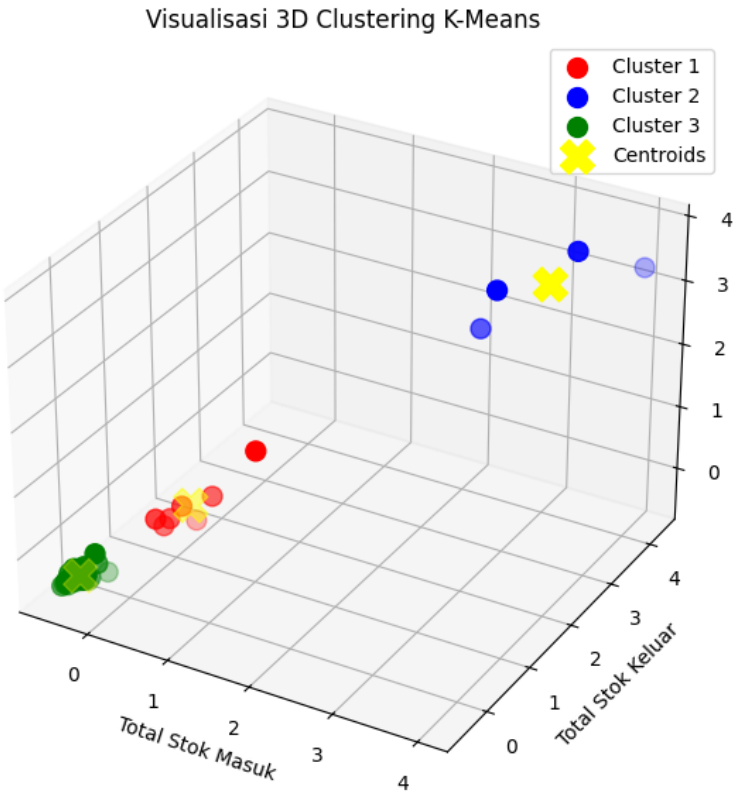
Gambar 4.18 Visualisasi *Clustering* dalam 3D

Setelah proses *clustering* selesai dilakukan menggunakan algoritma *K-Means*, pada langkah ini dilakukan visualisasi hasil pengelompokan data ke dalam bentuk grafik 3 dimensi. Visualisasi ini bertujuan untuk memudahkan dalam memahami sebaran data berdasarkan tiga variabel utama, yaitu Total Stok Masuk, Total Stok Keluar, dan Total Stok Akhir. Pada sintaks fig = plt.figure(figsize=(10, 7)) untuk membuat sebuah *figure* atau kanvas kosong dengan ukuran 10x7 inci yang akan digunakan untuk menampilkan grafik. ax = fig.add_subplot(111, projection='3d') untuk menambahkan subplot ke dalam figure tersebut dengan proyeksi tiga dimensi (3D), sehingga memungkinkan visualisasi dalam tiga sumbu (x, y, dan z). ax.scatter(...) digunakan untuk menggambarkan titik-titik data dalam ruang 3D berdasarkan hasil clustering: data dengan label cluster 0 digambarkan menggunakan warna merah, cluster 1 menggunakan warna biru, dan cluster 2 menggunakan warna hijau.

Masing-masing cluster berisi data yang memiliki pola stok yang mirip, dan setiap titik mewakili satu jenis barang. ax.scatter(..., marker='X', c='yellow', label='Centroids') untuk menambahkan tanda khusus berwarna kuning dengan bentuk huruf X yang menunjukkan posisi pusat cluster (centroid), yaitu titik rata-

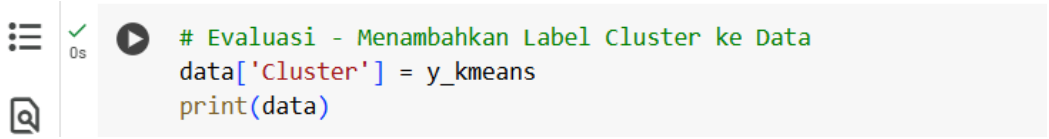
rata dari masing-masing kelompok data. ax.set_xlabel, ax.set_ylabel, ax.set_zlabel, ketiga baris ini memberikan label pada sumbu x, y, dan z untuk menunjukkan bahwa masing-masing sumbu mewakili Total Stok Masuk, Total Stok Keluar, dan Total Stok Akhir. ax.set_title dan ax.legend() untuk memberikan judul pada grafik dan menampilkan legenda agar pembaca dapat memahami makna dari warna dan simbol yang digunakan dalam visualisasi. plt.show() untuk menampilkan seluruh grafik 3D tersebut ke dalam *output Google Colab*.

Dengan visualisasi ini, hasil segmentasi barang berdasarkan pola stoknya dapat dilihat secara langsung. Setiap kelompok tampak terpisah berdasarkan kedekatan karakteristik stok, sehingga dapat membantu dalam pengambilan keputusan manajemen inventori.



Gambar 4.19 Grafik 3D *Clustering K-Means*

Langkah 11: Evaluasi - Menambahkan Label *Cluster* ke Data



Gambar 4.20 Evaluasi - Menambahkan Label *Cluster* ke Data

Setelah proses *clustering* dilakukan dengan algoritma *K-Means* dan setiap data telah dikelompokkan ke dalam tiga *cluster*, maka pada tahap ini hasil *cluster* yang diperoleh akan ditambahkan ke dalam data awal. Tujuannya adalah agar kita bisa melihat setiap barang masuk ke dalam kelompok (*cluster*) yang mana. Pada sintaks data['Cluster'] = y_kmeans, baris ini berfungsi untuk menambahkan kolom baru bernama Cluster ke dalam *dataframe* data. Nilai-nilai pada kolom ini berasal dari variabel y_kmeans, yaitu hasil keluaran dari model K-Means yang berisi label cluster (misalnya 0, 1, dan 2) untuk masing-masing baris data. Dengan begitu, setiap baris data akan teridentifikasi termasuk dalam *cluster* mana berdasarkan pola stoknya. Pada sintaks print(data) untuk menampilkan isi keseluruhan tabel data setelah kolom *cluster* ditambahkan. Hasil ini akan memperlihatkan nama barang, jumlah stok masuk, jumlah stok keluar, jumlah stok akhir, dan label *cluster* dari setiap barang.

Langkah ini sangat penting sebagai bagian dari evaluasi awal hasil *clustering*, karena kita dapat mengamati pola pengelompokan yang terbentuk dan memeriksa apakah hasilnya sudah sesuai dengan karakteristik data yang kita miliki. Output dari langkah ini bisa dilihat seperti pada Gambar 4.21 berikut ini.

	Nama Barang	Total Stok Masuk	Total Stok Keluar	\	Total Stok Akhir	Cluster
0	Jilbab Pashmina Kaos	913	822	0	91	2
1	Jilbab Pashmina Ceruty	889	703	1	186	2
2	Jilbab Pashmina Crinckle	831	567	2	264	2
3	Jilbab Pashmina Silk	860	435	3	425	2
4	Jilbab Pashmina Voal	898	794	4	104	2
5	Jilbab Pashmina Instan	898	805	5	93	2
6	Jilbab Pashmina Plisket	859	462	6	397	2
7	Jilbab Nurcheck	908	814	7	94	2
8	Jilbab Umama	908	814	8	94	2
9	Jilbab Azara Motif Syari	830	525	9	305	2
10	Jilbab Bordir Jambul	767	510	10	257	2
11	Jilbab Umama Syari	707	541	11	166	2
12	Jilbab Umama Motif Syari	871	721	12	150	2
13	Jilbab Luxuri Syari	670	563	13	107	2
14	Jilbab Bella Square	961	904	14	57	2
15	Jilbab Oskara Premium	785	384	15	401	2
16	Jilbab Umama Osaka	807	353	16	454	2
17	Jilbab Umama Eyelash	606	259	17	347	2
18	Jilbab Umama Voal Tryspan	633	555	18	78	2
19	Jilbab Paris Premium	918	845	19	73	2
20	Jilbab Bergo Pet	1705	1558	20	147	2
21	Jilbab Bergo Non Pet	1539	1090	21	449	2
22	Ciput Kaos	409	270	22	139	2
23	Ciput Rajut	421	261	23	160	2
24	Mukena Katun Bali	1115	665	24	450	2
25	Mukena Sutra Ori	1076	651	25	425	2
26	Mukena Parasut	1049	733	26	316	2
27	Mukena Silk	1040	1006	27	34	2
28	Mukena Jaguar	1079	799	28	280	2
29	Abaya	856	781	29	75	2
30	Gamis Ceruty	1067	867	30	200	2
31	Gamis Crinckle	902	748	31	154	2
32	Gamis Silk	889	703	32	186	2
33	Gamis Shimmer	1027	800	33	227	2
34	Gamis Katun Bordir	1038	772	34	266	2
35	Gamis Katun Twill	993	820	35	173	2
36	Gamis Polo Linen	1152	1045	36	107	2
37	Gamis Katun Dobi	1077	868	37	209	2
38	Gamis Renda	1603	870	38	733	2
39	Gamis Tile	1008	695	39	313	2
40	Baju Batik Wanita	4915	3874	40	1041	0
41	Baju Kaos Wanita	19104	13046	41	6058	1
42	Baju Kemeja Wanita	20974	15697	42	5277	1
43	Baju Tidur Wanita	3919	2833	43	1086	0
44	Rok Wanita	15435	10870	44	4565	1
45	Celana Wanita	16314	10834	45	5480	1
46	Baju Batik Pria	3778	2460	46	1318	0
47	Baju Kaos Pria	7496	4921	47	2575	0
48	Baju Koko Pria	4688	3173	48	1515	0
49	Baju Kemeja Pria	5682	4078	49	1604	0
50	Celana Pria	4181	2934	50	1247	0
51	Sajadah	390	292	51	98	2
52	Kain Panjang	411	374	52	37	2
53	Sarung	224	177	53	47	2
54	Peci	323	238	54	85	2

Gambar 4.21 Output Evaluasi - Menambahkan Label *Cluster* ke Data

Langkah 12: Menampilkan Hasil *Clustering*

```
# Menampilkan Hasil Clustering
print('Hasil Clustering:')
print(data[['Total Stok Masuk', 'Total Stok Keluar', 'Total Stok Akhir', 'Cluster']])
```

Gambar 4.22 Menampilkan Hasil *Clustering*

Pada langkah ini, program menampilkan hasil akhir dari proses *clustering* dalam bentuk tabel yang terdiri dari kolom Total Stok Masuk, Total Stok Keluar, Total Stok Akhir, dan *Cluster*. Tujuannya adalah untuk melihat ke *cluster* mana

setiap barang dikelompokkan berdasarkan karakteristik stoknya. Hal ini memudahkan dalam menganalisis pola distribusi stok pada masing-masing *cluster*.

Hasil Clustering:

	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
0	913	822	91	2
1	889	703	186	2
2	831	567	264	2
3	860	435	425	2
4	898	794	104	2
5	898	805	93	2
6	859	462	397	2
7	908	814	94	2
8	908	814	94	2
9	830	525	305	2
10	767	510	257	2
11	707	541	166	2
12	871	721	150	2
13	670	563	107	2
14	961	904	57	2
15	785	384	401	2
16	807	353	454	2
17	606	259	347	2
18	633	555	78	2
19	918	845	73	2
20	1705	1558	147	2
21	1539	1090	449	2
22	409	270	139	2
23	421	261	160	2
24	1115	665	450	2
25	1076	651	425	2
26	1049	733	316	2
27	1040	1006	34	2
28	1079	799	280	2
29	856	781	75	2
30	1067	867	200	2
31	902	748	154	2
32	889	703	186	2
33	1027	800	227	2
34	1038	772	266	2
35	993	820	173	2
36	1152	1045	107	2
37	1077	868	209	2
38	1603	870	733	2
39	1008	695	313	2
40	4915	3874	1041	0
41	19104	13046	6058	1
42	20974	15697	5277	1
43	3919	2833	1086	0
44	15435	10870	4565	1
45	16314	10834	5480	1
46	3778	2460	1318	0
47	7496	4921	2575	0
48	4688	3173	1515	0
49	5682	4078	1604	0
50	4181	2934	1247	0
51	390	292	98	2
52	411	374	37	2
53	224	177	47	2
54	323	238	85	2

Gambar 4.23 Hasil *Clustering*

Langkah 13: Interpretasi Hasil *Clustering*

```
# Interpretasi Hasil Clustering
for cluster in sorted(data['Cluster'].unique()):
    print(f'\nInterpretasi Cluster {cluster+1}:')
    print(data[data['Cluster'] == cluster].describe())
```

Gambar 4.24 Interpretasi Hasil *Clustering*

Pada tahap ini, dilakukan interpretasi terhadap masing-masing *cluster* yang telah terbentuk. Sintaks di atas akan menampilkan ringkasan statistik (seperti nilai rata-rata, minimum, maksimum, dll.) dari data yang termasuk dalam setiap *cluster*. Dengan demikian, kita dapat memahami karakteristik umum dari setiap kelompok barang, dengan tingkat permintaan rendah, sedang, dan tinggi. Hal ini membantu dalam pengambilan keputusan strategis seperti pengelolaan stok atau perencanaan pembelian.

Interpretasi Cluster 1:

	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
count	7.000000	7.000000	7.000000	7.0
mean	4951.285714	3467.571429	1483.714286	0.0
std	1298.441593	860.195105	523.453186	0.0
min	3778.000000	2460.000000	1041.000000	0.0
25%	4050.000000	2883.500000	1166.500000	0.0
50%	4688.000000	3173.000000	1318.000000	0.0
75%	5298.500000	3976.000000	1559.500000	0.0
max	7496.000000	4921.000000	2575.000000	0.0

Interpretasi Cluster 2:

	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
count	4.000000	4.000000	4.000000	4.0
mean	17956.750000	12611.750000	5345.000000	1.0
std	2548.060095	2302.276319	616.338111	0.0
min	15435.000000	10834.000000	4565.000000	1.0
25%	16094.250000	10861.000000	5099.000000	1.0
50%	17709.000000	11958.000000	5378.500000	1.0
75%	19571.500000	13708.750000	5624.500000	1.0
max	20974.000000	15697.000000	6058.000000	1.0

Interpretasi Cluster 3:

	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
count	44.000000	44.000000	44.000000	44.0
mean	884.363636	669.522727	214.840909	2.0
std	302.564549	270.672418	150.752676	0.0
min	224.000000	177.000000	34.000000	2.0
25%	780.500000	498.000000	94.000000	2.0
50%	898.000000	712.000000	169.500000	2.0
75%	1038.500000	815.500000	307.000000	2.0
max	1705.000000	1558.000000	733.000000	2.0

Gambar 4.25 Interpretasi Hasil *Cluster* 1, *Cluster* 2, dan *Cluster* 3

4.2 Pembahasan

4.2.1 Evaluasi Model *Clustering* dan Penentuan *Cluster* Optimal

Dalam proses analisis *clustering*, pemilihan jumlah *cluster* yang tepat merupakan langkah penting untuk memastikan bahwa hasil segmentasi yang diperoleh memiliki makna dan kualitas yang baik. Dengan melakukan evaluasi model pada sistem didapatkan hasil grafik dari *Elbow Method* pada Gambar 4.13, *Davies-Bouldin Index*, dan *Silhouette Coefficient* pada Gambar 4.16, berikut adalah tabel komparasi nilai SSE, DBI, dan *Silhouette Coefficient* untuk masing-masing jumlah *cluster* (k) dari 2 hingga 9:

Tabel 4.1 Komparasi Evaluasi Model *K-Means*

Jumlah Cluster (k)	SSE (Elbow Method)	Davies-Bouldin Index	Silhouette Coefficient
2	20.4	0.20	0.90
3	5.9	0.33	0.83
4	5.6	0.72	0.48
5	4.4	0.76	0.46
6	2.1	0.79	0.44
7	1.6	0.61	0.43
8	1.4	0.43	0.43
9	1.2	0.32	0.42

Berikut penjelasan mengenai evaluasi model menggunakan tiga metode pengukuran, yaitu *Elbow Method*, *Davies-Bouldin Index (DBI)*, dan *Silhouette Coefficient (SC)*:

a. *Elbow Method*

Elbow Method digunakan untuk mengukur seberapa besar penurunan nilai *Sum of Squared Error* (SSE) terhadap jumlah *cluster* yang dibentuk [18]. Berdasarkan grafik yang dihasilkan pada Gambar 4.13, nilai SSE

mengalami penurunan yang signifikan hingga jumlah cluster ke-3, kemudian penurunan berikutnya tidak terlalu tajam. Pola ini membentuk sudut menyerupai siku (*elbow*) pada titik k = 3, yang menunjukkan bahwa pemilihan jumlah *cluster* sebanyak tiga merupakan titik optimal, karena penambahan jumlah *cluster* setelah titik ini tidak memberikan penurunan error yang berarti.

b. *Davies-Bouldin Index* (*DBI*)

Davies-Bouldin Index merupakan metrik evaluasi yang menilai seberapa baik suatu *cluster* dipisahkan dari *cluster* lainnya, dengan nilai yang lebih rendah menunjukkan pemisahan yang lebih baik [19]. Dari hasil pengujian pada Gambar 4.16, nilai DBI paling rendah terlihat pada k = 2 (0,20) dan k = 3 (0,33). Meskipun k = 2 memberikan nilai DBI yang lebih kecil, namun nilai ini harus dibandingkan pula dengan metrik lainnya untuk menentukan jumlah *cluster* yang lebih representatif terhadap pola data.

c. *Silhouette Coefficient* (*SC*)

Silhouette Coefficient digunakan untuk menilai seberapa baik data dikelompokkan. Nilai ini menunjukkan seberapa dekat suatu data dengan kelompoknya dibandingkan dengan kelompok lain. Semakin mendekati angka 1, berarti pengelompokan tersebut semakin baik [20]. Berdasarkan grafik yang dihasilkan pada Gambar 4.16, nilai tertinggi terlihat saat jumlah *cluster* adalah 2 (dengan skor 0,90), lalu sedikit menurun pada jumlah *cluster* 3 (dengan skor 0,83). Walaupun skor terbaik ada di angka 2, namun memilih 3 *cluster* dianggap lebih tepat karena dapat memberikan

pengelompokan yang tetap baik dan lebih beragam, sehingga lebih mudah dianalisis dan digunakan dalam pengambilan keputusan.

Berdasarkan hasil ketiga metode evaluasi tersebut, jumlah *cluster* yang paling optimal adalah sebanyak tiga (k = 3). Hal ini didukung oleh penurunan SSE yang signifikan pada *Elbow Method*, nilai *DBI* yang masih tergolong rendah, serta nilai *Silhouette Coefficient* yang cukup tinggi. Dengan memilih tiga *cluster*, model dapat memberikan segmentasi yang efisien sekaligus tetap mempertahankan kualitas pemisahan antar kelompok data.

4.2.2 Perbandingan Hasil Perhitungan Manual dengan di *Google Colab*

Hasil Perhitungan Manual:

Perhitungan secara manual menggunakan *Excel* dilakukan dengan menghitung jarak data ke setiap pusat *cluster* (centroid), lalu data dikelompokkan ke dalam *cluster* yang jaraknya paling dekat. Proses ini dilakukan secara berulang hingga posisi centroid tidak berubah. Meskipun metode ini dapat dilakukan tanpa pemrograman, namun prosesnya cukup memakan waktu dan rentan terhadap kesalahan perhitungan, terutama jika jumlah data besar. Hasil perhitungan manual dapat dilihat pada Tabel dibawah ini.

Tabel 4.2 *Cluster* 1 Perhitungan Manual

No.	Nama Barang	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
1	Baju Batik Wanita	4915	3874	1041	**C1**
2	Baju Tidur Wanita	3919	2833	1086	**C1**
3	Baju Batik Pria	3778	2460	1318	**C1**
4	Baju Kaos Pria	7496	4921	2575	**C1**
5	Baju Koko Pria	4688	3173	1515	**C1**

6	Baju Kemeja Pria	5682	4078	1604	**C1**
7	Celana Pria	4181	2934	1247	**C1**
Jumlah					**7**

Tabel 4.3 *Cluster* 2 Perhitungan Manual

No.	Nama Barang	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
1	Baju Kaos Wanita	19104	13046	6058	**C2**
2	Baju Kemeja Wanita	20974	15697	5277	**C2**
3	Rok Wanita	15435	10870	4565	**C2**
4	Celana Wanita	16314	10834	5480	**C2**
Jumlah					**4**

Tabel 4.4 *Cluster* 3 Perhitungan Manual

No.	Nama Barang	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
1	Jilbab Pashmina Kaos	913	822	91	**C3**
2	Jilbab Pashmina Ceruty	889	703	186	**C3**
3	Jilbab Pashmina Crinckle	831	567	264	**C3**
4	Jilbab Pashmina Silk	860	435	425	**C3**
5	Jilbab Pashmina Voal	898	794	104	**C3**
6	Jilbab Pashmina Instan	898	805	93	**C3**
7	Jilbab Pashmina Plisket	859	462	397	**C3**
8	Jilbab Nurcheck	908	814	94	**C3**
9	Jilbab Umama	908	814	94	**C3**
10	Jilbab Azara Motif Syar'i	830	525	305	**C3**
11	Jilbab Bordir Jambul	767	510	257	**C3**

12	Jilbab Umama Syar'i	707	541	166	**C3**
13	Jilbab Umama Motif Syar'i	871	721	150	**C3**
14	Jilbab Luxuri Syar'i	670	563	107	**C3**
15	Jilbab Bella Square	961	904	57	**C3**
16	Jilbab Oskara Premium	785	384	401	**C3**
17	Jilbab Umama Osaka	807	353	454	**C3**
18	Jilbab Umama Eyelash	606	259	347	**C3**
19	Jilbab Umama Voal Tryspan	633	555	78	**C3**
20	Jilbab Paris Premium	918	845	73	**C3**
21	Jilbab Bergo Pet	1705	1558	147	**C3**
22	Jilbab Bergo Non Pet	1539	1090	449	**C3**
23	Ciput Kaos	409	270	139	**C3**
24	Ciput Rajut	421	261	160	**C3**
25	Mukena Katun Bali	1115	665	450	**C3**
26	Mukena Sutra Ori	1076	651	425	**C3**
27	Mukena Parasut	1049	733	316	**C3**
28	Mukena Silk	1040	1006	34	**C3**
29	Mukena Jaguar	1079	799	280	**C3**
30	Abaya	856	781	75	**C3**
31	Gamis Ceruty	1067	867	200	**C3**
32	Gamis Crinckle	902	748	154	**C3**
33	Gamis Silk	889	703	186	**C3**
34	Gamis Shimmer	1027	800	227	**C3**
35	Gamis Katun Bordir	1038	772	266	**C3**
36	Gamis Katun Twill	993	820	173	**C3**
37	Gamis Polo Linen	1152	1045	107	**C3**
38	Gamis Katun Dobi	1077	868	209	**C3**
39	Gamis Renda	1603	870	733	**C3**
40	Gamis Tile	1008	695	313	**C3**
41	Sajadah	390	292	98	**C3**

42	Kain Panjang	411	374	37	**C3**
43	Sarung	224	177	47	**C3**
44	Peci	323	238	85	**C3**
Jumlah					**44**

Hasil Perhitungan di *Google Colab*:

Pada pendekatan berbasis *Machine Learning*, algoritma *K-Means* digunakan untuk melakukan *clustering* secara otomatis. Proses ini dijalankan menggunakan *Python*, yang secara efisien menghitung jarak, memperbarui posisi centroid, dan mengelompokkan data secara cepat dan konsisten. Dalam proses ini, digunakan tiga model evaluasi untuk menentukan kualitas hasil *clustering*, yaitu *Elbow Method*, *Davies-Bouldin Index (DBI)*, dan *Silhouette Coefficient (SC)*. Dari ketiga model evaluasi tersebut, jumlah *cluster* yang paling optimal adalah sebanyak tiga (k = 3). Oleh karena itu, pendekatan *Machine Learning* terbukti lebih unggul karena prosesnya lebih cepat, dapat dievaluasi secara objektif, dan cocok digunakan untuk data dalam jumlah besar. Hasil perhitungan di *Google Colab* dapat dilihat pada Tabel dibawah ini.

Tabel 4.5 Hasil Perhitungan di *Google Colab*

Hasil Clustering:				
	Total Stok Masuk	Total Stok Keluar	Total Stok Akhir	Cluster
0	913	822	91	2
1	889	703	186	2
2	831	567	264	2
3	860	435	425	2
4	898	794	104	2
5	898	805	93	2
6	859	462	397	2
7	908	814	94	2
8	908	814	94	2
9	830	525	305	2

10	767	510	257	2
11	707	541	166	2
12	871	721	150	2
13	670	563	107	2
14	961	904	57	2
15	785	384	401	2
16	807	353	454	2
17	606	259	347	2
18	633	555	78	2
19	918	845	73	2
20	1705	1558	147	2
21	1539	1090	449	2
22	409	270	139	2
23	421	261	160	2
24	1115	665	450	2
25	1076	651	425	2
26	1049	733	316	2
27	1040	1006	34	2
28	1079	799	280	2
29	856	781	75	2
30	1067	867	200	2
31	902	748	154	2
32	889	703	186	2
33	1027	800	227	2
34	1038	772	266	2
35	993	820	173	2
36	1152	1045	107	2
37	1077	868	209	2
38	1603	870	733	2
39	1008	695	313	2
40	4915	3874	1041	0
41	19104	13046	6058	1
42	20974	15697	5277	1
43	3919	2833	1086	0
44	15435	10870	4565	1
45	16314	10834	5480	1
46	3778	2460	1318	0
47	7496	4921	2575	0
48	4688	3173	1515	0
49	5682	4078	1604	0

50	4181	2934	1247	0
51	390	292	98	2
52	411	374	37	2
53	224	177	47	2
54	323	238	85	2

4.2.3 Interpretasi Hasil Perhitungan

Dengan melakukan perhitungan manual dan implementasi menggunakan *Machine Learning* di *Google Colab*, hasil segmentasi data menunjukkan hasil yang konsisten. Produk-produk yang tersedia di Toko R2 Collection terbagi menjadi tiga kelompok berdasarkan pola permintaannya, yaitu:

a. *Cluster* 1 dengan permintaan produk rendah berjumlah 7 produk.

   Pada *cluster* ini barang-barang yang termasuk dengan permintaan rendah, antara lain: Baju Batik Wanita, Baju Tidur Wanita, Baju Batik Pria, Baju Kaos Pria, Baju Koko Pria, Baju Kemeja Pria, dan Celana Pria. Produk dalam cluster ini mengalami perputaran stok yang lambat, yang berarti produk tidak banyak diminati atau dibeli oleh pelanggan. Produk dalam *cluster* ini berisiko menumpuk di gudang, yang dapat menyebabkan pemborosan ruang penyimpanan dan potensi kerugian akibat produk kadaluarsa atau tidak laku. Sehingga, Perlu dilakukan evaluasi ulang terhadap strategi pemasaran, desain produk, atau relevansi produk dengan kebutuhan pelanggan.

b. *Cluster* 2 dengan permintaan produk sedang berjumlah 4 produk.

   Pada *cluster* ini barang-barang yang termasuk dengan permintaan sedang, antara lain: Baju Kaos Wanita, Baju Kemeja Wanita, Rok Wanita, dan

Celana Wanita. Terdapat keseimbangan antara stok masuk dan stok akhir, yang menunjukkan bahwa produk memiliki permintaan yang relatif stabil. Perputaran barang tidak terlalu cepat, tetapi juga tidak terhenti. Produk dalam *cluster* ini masih memiliki potensi untuk ditingkatkan penjualannya. Sehingga, Perlu dilakukan monitoring berkala dan strategi optimalisasi seperti penempatan yang lebih strategis di toko, diskon ringan, atau bundling dengan produk laris.

c. *Cluster* 3 dengan permintaan produk tinggi berjumlah 44 produk.

Pada *cluster* ini barang-barang yang termasuk dengan permintaan tinggi yaitu: Jilbab Pashmina Kaos, Jilbab Pashmina Ceruty, Jilbab Pashmina Crinckle, Jilbab Pashmina Silk, Jilbab Pashmina Voal, Jilbab Pashmina Instan, Jilbab Pashmina Plisket, Jilbab Nurcheck, Jilbab Umama, Jilbab Azara Motif Syar'I, Jilbab Bordir Jambul, Jilbab Umama Syar'I, Jilbab Umama Motif Syar'I, Jilbab Luxuri Syar'I, Jilbab Bella Square, Jilbab Oskara Premium, Jilbab Umama Osaka, Jilbab Umama Eyelash, Jilbab Umama Voal Tryspan, Jilbab Paris Premium, Jilbab Bergo Pet, Jilbab Bergo Non Pet, Ciput Kaos, Ciput Rajut, Mukena Katun Bali, Mukena Sutra Ori, Mukena Parasut, Mukena Silk, Mukena Jaguar, Abaya, Gamis Ceruty, Gamis Crinckle, Gamis Silk, Gamis Shimmer, Gamis Katun Bordir, Gamis Katun Twill, Gamis Polo Linen, Gamis Katun Dobi, Gamis Renda, Gamis Tile, Sajadah, Kain Panjang, Sarung, dan Peci.

Produk pada *cluster* ini menunjukkan stok keluar yang tinggi, menandakan bahwa produk banyak dicari dan dibeli oleh pelanggan. Perputaran stok

berlangsung cepat, sehingga stok akhir cenderung rendah dan perlu segera diisi kembali untuk menghindari kehabisan. Produk dalam *cluster* ini adalah produk utama atau andalan toko, sehingga pengelolaan stoknya harus sangat diperhatikan. Toko sebaiknya menjaga ketersediaan produk dalam *cluster* ini agar tidak terjadi kekosongan barang yang dapat menurunkan kepuasan pelanggan.