BAB 3
METODE PENELITIAN

Penelitian ini menggunakan pendekatan analisis faktor-faktor untuk menentukan penentuan pola penjualan di PT Indonesia Plafon Semesta. Data akan dikumpulkan melalui survei lapangan pada perusahaan yang berfokus pada penjualan ataupun nilai stok barang. Selanjutnya, algoritma K-Means akan diterapkan untuk mengidentifikasi faktor-faktor dan melakukan penentuan pola penjualan pada PT Indonesia Plafon Semesta.

3.1 BAHAN DAN ALAT PENELITIAN

Bahan penelitian yang diperlukan untuk penelitian ini adalah data primer serta data sekunder. Data primer adalah data historis penjualan produk PT Indonesia Plafon Semesta selama 2,5 tahun terakhir, data informasi produk, seperti kode produk, nama produk, jumlah transaksi dan data faktor eksternal yang dapat mempengaruhi permintaan produk, seperti data musim, tren pasar, dan promosi.

Data sekunder berupa studi terdahulu mengenai Klastering Data dan penentuan pola penjualan untuk meningkatkan efisiensi manajemen persediaan serta literatur ilmiah tentang algoritma K-Means perlu dipelajari untuk memahami metodologi analisis yang digunakan dalam penelitian.

Alat yang dipakai di penelitian ini yaitu komputer yang memiliki spesifikasi cukup untuk menjalankan sistem operasi dan perangkat lunak pengembangan serta koneksi internet yang stabil.

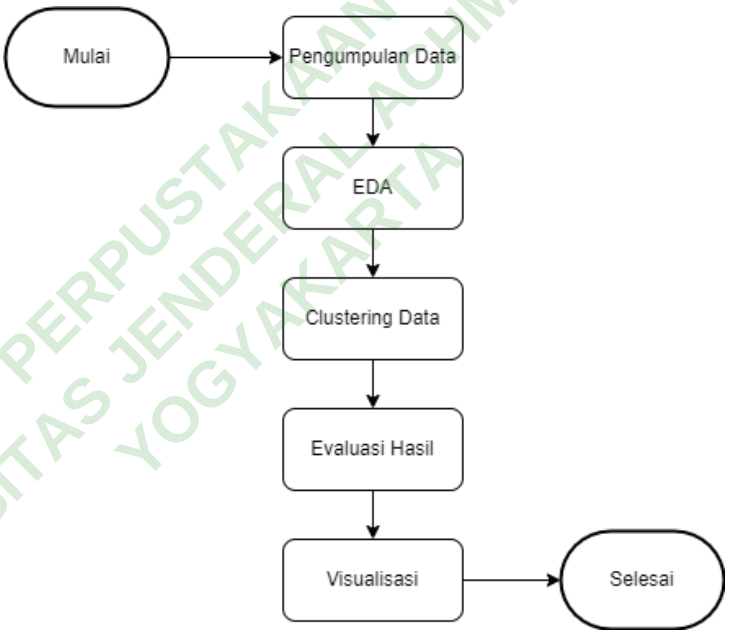
Sistem operasi dan aplikasi yang digunakan dalam pengembangan aplikasi ini adalah :

1. Perangkat Keras
    - Processor : 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz
    - RAM : 4 GB

2. Perangkat Lunak
    - Sistem Operasi : Windows 11
    - Bahasa Pemrograman : Python 3.9.0
    - Aplikasi Tambahan : Visual Studio Code 1.88, Google Collab

3.2 JALAN PENELITIAN

Pada tahapan ini akan dilakukan secara berurutan satu persatu dan berjalan berurutan. Berikut ini merupakan jalan penelitian serta tahapan K-Means yang digunakan pada penelitian tugas akhir, ditunjukan pada Gambar 3.1



Gambar 3.1 Alur Penelitian

3.2.1 Pengumpulan Data

Penelitian ini dilakukan di PT Indonesia Plafon Semesta, adapun data penjualan yang akan diproses merupakan data penjualan dari bulan Januari 2022-Juni 2024. Pengumpulan data ini dipakai untuk merekap semua data dan semua informasi yang dibutuhkan dalam proses pembuatan sistem. Pada tahap ini dilakukan pengumpulan data penjualan historis dari PT Indonesia Plafon Semesta. Data mencakup informasi seperti produk yang dibeli, waktu pembelian, dan jumlah

transaksi maupun jumlah pembelian. Contoh data penjualan bisa dilihat pada tabel dibawah

Tabel 3.1 Data Penjualan Bulan Juni 2024

No	Kode	Jumlah Transaksi	Jumlah Terjual
1	PF 01	11	173
2	PF 02	8	96
3	PF 03	7	174
4	PF 04	13	149
5	PF 05	35	550
6	PF 06	8	95
7	PF 07	3	29
8	PF 08	8	72
9	PF 09	25	321
10	PF 10	5	82
11	PF 11	2	10
12	PF 12	15	105

3.2.2 EDA (*Exploratory Data Analysis)*

Pada tahap ini dilakukan eksplorasi data penjualan dan memahami pola pembelian yang ada. Pada tahap ini dilakukan langkah langkah yaitu menampilkan data, menampilkan distribusi penjualan, menampilkan grafik produk terbaik dan terburuk dalam penjualan, pertumbuhan penjualan perbulan (Januari 2022 – Juni 24), cek data info, mengkodekan kolom objek menjadi numerik, memilih kolom yang digunakan untuk klastering dan mengubah variabel dari data frame menjadi *array*.

1. Menampilkan Data

   Langkah pertama adalah menampilkan data untuk melihat atribut atribut yang akan digunakan, seperti pada kode dibawah ini.

```
#menampilkan data
df.head()Klastering Data
```

Output yang diperoleh adalah tampilan data yang terdiri dari nomor, bulan, tahun, kode, transaksi dan terjual. Seperti pada Gambar 3.2 dibawah ini

	No	Bulan	Tahun	Kode	Transaksi	Terjual
0	1	Januari	2022	PF 01	12	72
1	2	Januari	2022	PF 02	14	68
2	3	Januari	2022	PF 03	28	184
3	4	Januari	2022	PF 04	26	183
4	5	Januari	2022	PF 05	39	345

Gambar 3.2 Menampilkan Data

2. Menampilkan Grafik Produk Terbaik dan Terburuk dalam Penjualan

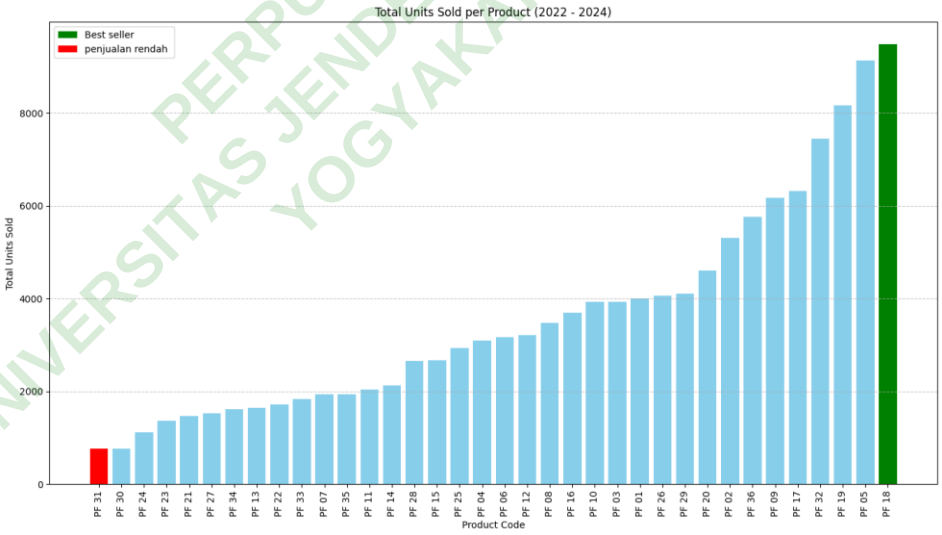
Menampilkan grafik produk terbaik dan terburuk dalam penjualan adalah salah satu cara untuk mengidentifikasi produk yang memiliki frekuensi penjualan tinggi dan yang memiliki frekuensi penjualan rendah. Hal ini bisa membantu dalam pengambilan keputusan bisnis seperti strategi pemasaran, pengelolaan inventaris, dan pengembangan produk. Berikut untuk kodenya

```
#grafik produk terbaik dan terburuk dalam penjualan
product_sales = df.groupby('Kode')['Terjual'].sum()
product_sales_sorted = product_sales.sort_values()

plt.figure(figsize=(14, 8))
plt.bar(product_sales_sorted.index,
product_sales_sorted.values, color='skyblue')
plt.xlabel('Product Code')
plt.ylabel('Total Units Sold')
plt.title('Total Units Sold per Product (2022 - 2024)')
plt.xticks(rotation=90)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
```

```
# Highlight produk dengan kinerja terbaik dan terburuk
plt.bar('PF 18', 9488, color='green', label='Best seller ')
plt.bar('PF 31', 762, color='red', label='penjualan rendah
')
plt.legend()
plt.show()
```

Hasil dari kode diatas maka output yang ditampilkan pada Gambar 3.3 adalah grafik penjualan dari yang terendah hingga tertinggi. Produk dengan kode "PF 18" adalah produk dengan penjualan tertinggi, mencapai lebih dari 9000 unit. Ini ditandai dengan batang berwarna hijau, yang menunjukkan bahwa produk ini merupakan *best seller*. Produk dengan kode "PF 31" memiliki penjualan terendah, di bawah 500 unit. Ini ditandai dengan batang berwarna merah, yang menunjukkan bahwa produk ini memiliki penjualan yang rendah.

.



Gambar 3.3 Grafik Produk Terbaik dan Terburuk dalam Penjualan

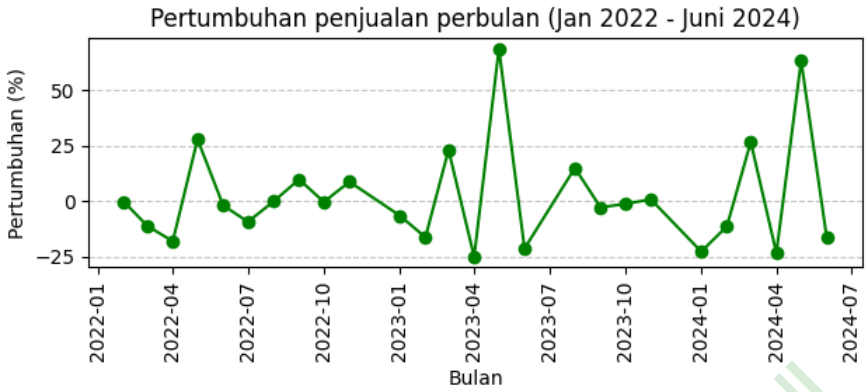
3. Pertumbuhan Penjualan Perbulan (Januari 2022 – Jun 2024)

Pertumbuhan penjualan perbulan mengacu pada peningkatan atau penurunan penjualan produk dari satu bulan ke bulan berikutnya. Ini adalah indikator penting yang dipakai oleh bisnis agar bisa menilai kinerja penjualan dalam jangka pendek. Berikut adalah kode untuk pertumbuhan penjualan perbulan

```
#Nama Bulan diubah menjadi inggris
month_mapping = {
    'Januari': 'January', 'Februari': 'February', 'Maret':
'March', 'April': 'April',
    'Mei': 'May', 'Juni': 'June', 'Juli': 'July', 'Agustus':
'August',
    'September':   'September',   'Oktober':   'October',
'November': 'November', 'Desember': 'December'
}

df['Bulan'] = df['Bulan'].map(month_mapping)

# membuat kolom tanggal
df['Tanggal'] = pd.to_datetime(df['Tahun'].astype(str) + '-
' + df['Bulan'], format='%Y-%B')

# Filter data dari Januari 2022 ke Juni 2024
filter_data     =   df[(df['Tanggal']   >=   '2022-01')   &
(df['Tanggal'] <= '2023-12')]

Penjualan_perbulan                                        =
filter_data.groupby(['Tanggal'])['Terjual'].sum()
Pertumbuhan_perbulan = Penjualan_perbulan.pct_change() * 100

sales_growth_perbulan = pd.DataFrame({
    'Total unit terjual': Penjualan_perbulan,
    'Pertumbuhan(%)': Pertumbuhan_perbulan
}).reset_index()
```

```
# Plot
plt.subplot(2, 1, 2)
plt.plot(sales_growth_perbulan['Tanggal'],
sales_growth_perbulan['Pertumbuhan(%)'],        marker='o',
color='green')
plt.xlabel('Bulan')
plt.ylabel('Pertumbuhan (%)')
plt.title('Pertumbuhan penjualan perbulan (Jan 2022 - Juni
2024)')
plt.xticks(rotation=90)
plt.grid(axis='y', linestyle='--', alpha=0.7)

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

Hasil dari *output* kode diatas yaitu menampilkan grafik pertumbuhan penjualan perbulan selama 2,5 tahun. Gambar tersebut menunjukkan grafik pertumbuhan penjualan perbulan untuk periode Januari 2022 hingga Juni 2024, grafik menunjukkan fluktuasi yang signifikan. Ada beberapa periode di mana pertumbuhan meningkat tajam, diikuti oleh penurunan yang cukup besar. Terdapat puncak pertumbuhan yang sangat tinggi pada sekitar April 2023, di mana pertumbuhan mencapai lebih dari 50%. Ini menunjukkan bahwa ada peningkatan penjualan yang sangat besar pada bulan tersebut. penurunan tajam juga terlihat beberapa kali, terutama setelah puncak pertumbuhan pada April 2023, yang mungkin menunjukkan bahwa peningkatan besar tersebut tidak berkelanjutan atau diikuti oleh periode penurunan yang signifikan. Secara keseluruhan, tidak ada tren pertumbuhan yang jelas terlihat sepanjang periode tersebut. Penjualan tampaknya mengalami variasi besar tanpa arah tren yang konsisten. Bisa di lihat pada Gambar 3.4



Gambar 3.4 Grafik Pertumbuhan Penjualan Perbulan

4. Cek Info pada Data

Digunakan untuk memahami karakteristik dasar dataset yang sedang dianalisis. Tujuan utamanya adalah untuk mendapatkan gambaran umum mengenai struktur dan kualitas data sebelum melakukan analisis lebih mendalam. Untuk cek jenis tipe data bisa dilihat pada kode berikut

```
#cek tipe data
df.info()
```

Hasil dari kode cek tipe data diatas adalah menampilkan jenis tipe data seperti pada Gambar 3.5 dibawah

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1080 entries, 0 to 1079
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   No         1080 non-null   int64
 1   Bulan      972 non-null    object
 2   Tahun      1080 non-null   int64
 3   Kode       1080 non-null   object
 4   Transaksi  1080 non-null   int64
 5   Terjual    1080 non-null   int64
 6   Tanggal    972 non-null    datetime64[ns]
dtypes: datetime64[ns](1), int64(4), object(2)
memory usage: 59.2+ KB
```

Gambar 3.5 Cek Tipe Data

5. Mengkodekan Kolom Objek Menjadi Numerik

Mengkodekan kolom objek menjadi numerik adalah proses mengubah data kategorikal atau teks dalam suatu kolom dataframe menjadi nilai numerik yang dapat digunakan dalam analisis statistik. Bisa dilihat pada kode berikut

```
#Mengkodekan kolom objek menjadi numerik
label_encoder = LabelEncoder()
for column in df.columns:
    df[column] = label_encoder.fit_transform(df[column])
df
```

Hasil dari kode diatas yaitu mengubah kolom objek menjadi numerik bisa dilihat pada Gambar 3.6

	No	Bulan	Tahun	Kode	Transaksi	Terjual	Tanggal
0	0	3	0	0	12	67	0
1	1	3	0	1	14	63	0
2	2	3	0	2	28	171	0
3	3	3	0	3	26	170	0
4	4	3	0	4	39	283	0
...	...	...	...	...	...	...	...
1075	1075	5	2	31	6	113	26
1076	1076	5	2	32	1	1	26
1077	1077	5	2	33	0	0	26
1078	1078	5	2	34	5	30	26
1079	1079	5	2	35	6	108	26

1080 rows × 7 columns

Gambar 3.6 Hasil Ubah Kolom Objek Menjadi Numerik

6. Memilih Kolom yang Digunakan Untuk Klastering

Pada saat menjalankan algoritma K Means tidak semua atribut di pakai, maka pilih kolom yang akan digunakan. Untuk kodenya bias dilihat seperti pada kode berikut

```
kolom_digunakan = [
    'Bulan',
    'Tahun',
    'Kode',
    'Transaksi',
    'Terjual'
    ]
data = df[kolom_digunakan]
data
```

Hasil dari kode diatas untuk memilih kolom bisa dilihat pada Gambar 3.7

	Bulan	Tahun	Kode	Transaksi	Terjual
0	3	0	0	12	67
1	3	0	1	14	63
2	3	0	2	28	171
3	3	0	3	26	170
4	3	0	4	39	283
...	...	...	...	...	...
1075	5	2	31	6	113
1076	5	2	32	1	1
1077	5	2	33	0	0
1078	5	2	34	5	30
1079	5	2	35	6	108

1080 rows × 5 columns

Gambar 3.7 Kolom yang Digunakan Untuk Klastering

7. Mengubah Variabel dari Bentuk Data Frame Menjadi Array

Untuk mengubah variabel yang awalnya memiliki bentuk data frame menjadi bentuk array yaitu dengan kode berikut

```
# mengubah variabel yang sebelumnya berbentuk data frame
menjadi array
x_array = np.array(data)
x_array
```

Hasil dari kode diatas bisa dilihat pada Gambar 3.8

```
array([[  3,   0,   0,  12,  67],
       [  3,   0,   1,  14,  63],
       [  3,   0,   2,  28, 171],
       ...,
       [  5,   2,  33,   0,   0],
       [  5,   2,  34,   5,  30],
       [  5,   2,  35,   6, 108]])
```

Gambar 3.8 Hasil Output Array

8. Scaling Data

Pada proses ini dilakukan normalisasi data, dipastikan bahwa semua variabel memiliki skala yang sama. Pada penelitian ini digunakan min-max scaling untuk menormalisasi data, seperti kode dibawah ini

```
scaler = MinMaxScaler()
data_scaled = scaler.fit_transform(data_scaled)
data_scaled
```

Hasil dari data yang telah di scaling bisa dilihat pada Gambar 3.9

```
array([[0.27272727, 0.        , 0.        , 0.3       , 0.2133758 ],
       [0.27272727, 0.        , 0.02857143, 0.35      , 0.20063694],
       [0.27272727, 0.        , 0.05714286, 0.7       , 0.54458599],
       ...,
       [0.45454545, 1.        , 0.94285714, 0.        , 0.        ],
       [0.45454545, 1.        , 0.97142857, 0.125     , 0.0955414 ],
       [0.45454545, 1.        , 1.        , 0.15      , 0.34394904]])
```

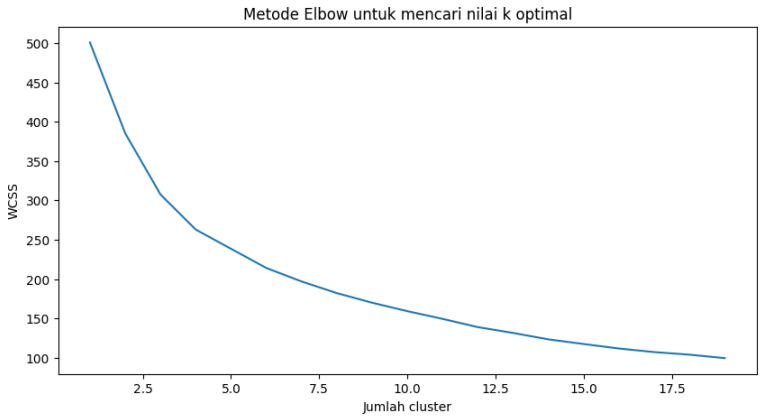
Gambar 3.9 Output Data Scaling

3.2.3 Elbow Method

Elbow Method merupakan salah satu cara yang dipakai dalam analisis klaster dengan tujuan menentukan jumlah klister yang optimal dalam kumpulan data. Teknik ini membantu dalam memilih jumlah klaster yang tepat dengan memvisualisasikan variasi total dalam data sebagai fungsi dari jumlah klaster yang digunakan. Prosesnya bisa dilihat pada kode dibawah ini

```
# Menentukan jumlah klaster yang optimal menggunakan metode elbow
wcss = []
for i in range(1, 20):
    kmeans = KMeans(n_klasters=i, init='k-means++', random_state=42)
    kmeans.fit(data_scaled)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(10, 5))
plt.plot(range(1, 20), wcss)
plt.title('Metode Elbow untuk mencari nilai k optimal')
plt.xlabel('Jumlah klaster')
plt.ylabel('WCSS')
plt.show()
```

Berdasarkan Gambar 3.10 dapat disimpulkan bahwa jumlah klaster optimal adalah sebanyak 3 klaster.



Gambar 3.10 Hasil Elbow Method

3.2.4 Konfigurasi K-Means

Setelah ditentukan jumlah klaster maka bisa dijalankan kode konfigurasi kmeans, bisa dilihat pada kode dibawah

```
# menentukan dan mengkonfigurasi fungsi kmeans
optimal_k = 3
kmeans = KMeans(n_klasters=optimal_k, init='k-means++', random_state=42)
# menentukan klaster dari data
kmeans.fit(data_scaled)
```

1. Nilai pusat dari masing masing klaster

   Untuk melihat nilai pusat dari masing masing klaster dapat menggunakan kode berikut ini

```
# mencari nilai pusat dari masing masing klaster
print(kmeans.klaster_centers_)
```

Hasilnya bisa dilihat pada Gambar 3.12 dibawah ini

```
[[0.59600313 0.16271552 0.55197044 0.18873922 0.20296371]
 [0.27272727 0.82191781 0.52113503 0.18390411 0.19251156]
 [0.53703704 0.3595679  0.40652557 0.60987654 0.66858339]]
```

Gambar 3.11 Nilai Pusat dari Masing Masing Klaster

2. Menambahkan Kolom Hasil klaster

   Untuk menambahkan kolom dan menampilkan hasil klaster bisa dilihat pada kode dibawah

```
# menampilkan hasil klaster
print(kmeans.labels_)
# menambahkan kolom klaster dalam dataframe data
data['klaster'] = kmeans.labels_
data.head(10)
```

Hasilnya yaitu setiap data sudah memiliki label klaster, bisa dilihat pada Gambar 3.13

	Bulan	Tahun	Kode	Transaksi	Terjual	cluster
0	3	0	0	12	67	0
1	3	0	1	14	63	0
2	3	0	2	28	171	2
3	3	0	3	26	170	2
4	3	0	4	39	283	2
5	3	0	5	17	111	0
6	3	0	6	3	10	0
7	3	0	7	11	96	0
8	3	0	8	37	242	2
9	3	0	9	30	120	2

Gambar 3.12 Output Hasil Klaster pada Tabel

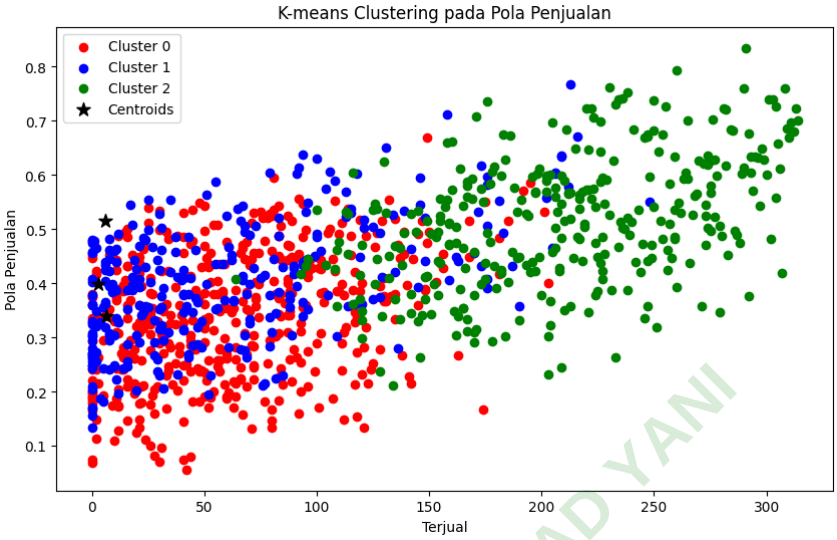
3. Menampilkan centroid pada k means

Nilai centroid dari masing-masing klaster sangat penting untuk dianalisis karena centroid tersebut merepresentasikan titik tengah atau pusat dari setiap klaster yang terbentuk. Bisa dilihat pada kode berikut

```
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler

# Centroids
centroids = kmeans.klaster_centers_

# Visualisasi
plt.figure(figsize=(10, 6))

colors = ['red', 'blue', 'green']
for i in range(3):
    klaster_data = data[data['klaster'] == i]
```

```
    # Calculate the mean of scaled features for each student
in the klaster
    mean_scaled_features = data_scaled[data['klaster'] ==
i].mean(axis=1)
    plt.scatter(klaster_data['Terjual'],
mean_scaled_features,  color=colors[i],  label=f'Klaster
{i}')

# Plotting centroids
centroid_terjual = scaler.inverse_transform(centroids)[:,
0]  # Pastikan scaler sudah terlatih dengan data yang sesuai
centroid_mean_features  =  centroids.mean(axis=1)     #
Memastikan rata-rata fitur centroid
plt.scatter(centroid_terjual,     centroid_mean_features,
color='black', marker='*', s=100, label='Centroids')
plt.xlabel('Terjual')
plt.ylabel('Pola Penjualan')
plt.title('K-means Klastering pada Pola Penjualan')
plt.legend()
plt.show()
```

Gambar 3.13 dibawah adalah hasil klasterisasi menggunakan algoritma K-Means pada data pola penjualan. Centroid dari masing-masing klaster ditandai dengan simbol bintang hitam. Data pola penjualan terbagi menjadi tiga klaster utama yang berbeda berdasarkan jumlah produk yang terjual dan pola penjualannya. Klaster 0 memiliki penjualan yang lebih rendah. Klaster 1 memiliki penjualan yang bervariasi namun tidak setinggi Klaster 2, mencerminkan produk dengan permintaan menengah. Sedangakan Klaster 2 memiliki penjualan yang tinggi, mencerminkan produk dengan permintaan tinggi.



Gambar 3.13 Hasil Centroid pada K-Means

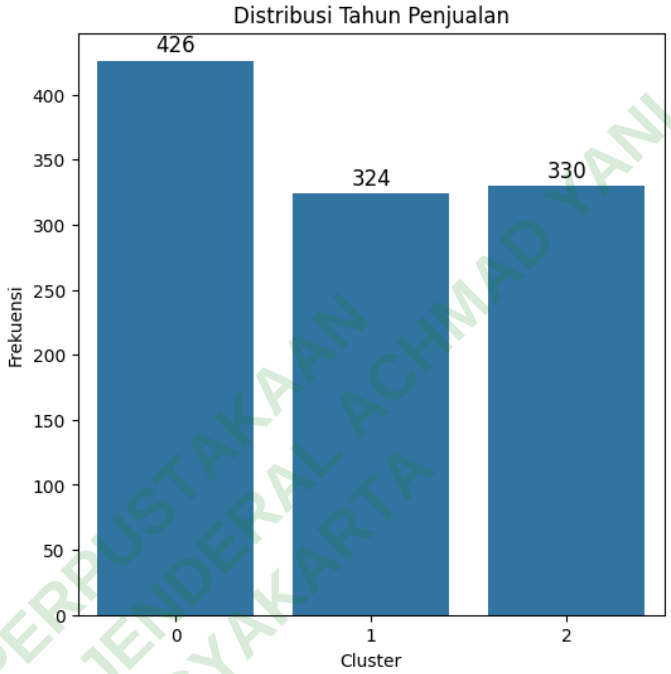
4. Menghitung Jumlah Cluster dan Visualisasinya

Untuk menghitung jumlah klaster pada data yang telah diolah serta memvisualisasikannya bisa dilihat pada kode dibawah

```
# menghitung jumlah cluster
for i in range(optimal_k):
    print(f"Jumlah di cluster {i}: {(data['cluster'] ==
i).sum()}")
# menampilkan visualisasi data distribusi tahun penjualan
setelah di clustering
plt.figure(figsize=(6, 6))
ax = sns.countplot(x='cluster', data=data)
plt.title('Distribusi Tahun Penjualan')
plt.xlabel('Cluster')
plt.ylabel('Frekuensi')
# Menambahkan jumlah data di atas setiap batang
for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() +
p.get_width() / 2., p.get_height()),
                ha='center', va='baseline', fontsize=12,
color='black', xytext=(0, 5),
                textcoords='offset points')
```

```
plt.show()
```

Hasil dari *output* kode diatas bisa dilihat pada Gambar dibawah. Dapat disimpulkan bahwa terdapat 3 klaster, klaster 0 memiliki frekuensi penjualan tertinggi 426 penjualan, klaster 1 memiliki frekuensi 324 penjualan dan klaster 2 memiliki frekuensi 330 penjualan.



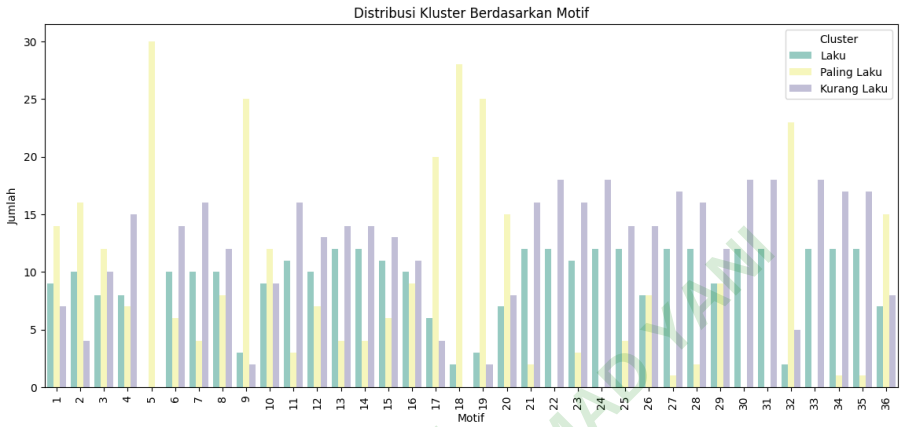
Gambar 3.14 Jumlah Distribusi Klaster

5. Distribusi Klaster Berdasarkan Motif

Merupakan analisis yang menunjukkan bagaimana data dalam klaster tertentu terbagi berdasarkan motif yang diidentifikasi. Ini membantu dalam memahami karakteristik unik dari setiap klaster dan bagaimana motif-motif tersebut muncul di dalam data. Berikut merupakan kode untuk melihat distribusi klaster berdasarkan motif

```
plt.figure(figsize=(14, 6))
sns.countplot(data=data,     x='Kode',     hue='klaster',
palette='Set3')
plt.title('Distribusi Klaster Berdasarkan Motif')
plt.xlabel('Motif')
plt.ylabel('Jumlah')
```

```
plt.legend(title='Klaster')
plt.xticks(rotation=90)
plt.show()
```



Gambar 3.15 Distribusi Klaster Berdasarkan Motif

Hasil outputnya bisa dilihat pada Gambar 3.15. Pada sumbu X merupakan jumlah kode motif dan pada sumbu Y merupakan jumlah setiap motif dalam klaster. Pada diagaram ini terdiri dari 3 kalster yaitu laku (hijau), paling laku (kuning) dan kurang laku (ungu). Hasil dari pengamatan pada diagaram diatas yaitu :

- Motif dengan jumlah paling banyak dalam kluster "Paling Laku/Warna Kuning" yaitu motif 3, 5, 18, 32, dan 36 memiliki jumlah yang secara signifikan lebih tinggi dibandingkan dengan klaster lainnya.
- Motif dengan jumlah paling banyak dalam klaster "Kurang Laku/Warna Ungu" yaitu motif 2, 7, 8, 15, 26, 31, dan 34 relatif tinggi dalam kluster ini.
- Sedangkan motif dengan jumlah paling banyak dalam klaster "Laku/Warna Hijau" yaitu motif 10, 11, 12, 14, 23, dan 27 menunjukkan jumlah yang lebih tinggi dibandingkan dengan kluster lainnya.

Diagram ini secara efektif menggambarkan bagaimana berbagai motif didistribusikan di tiga kluster, memberikan wawasan tentang popularitas atau frekuensi motif dalam setiap kluster.