# Import library yang diperlukan

import os

import zipfile

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.cluster import KMeans

# Fungsi untuk mengunduh dan mengekstrak dataset dari Kaggle

def download\_and\_extract\_kaggle\_dataset(dataset\_name, output\_dir):

zip\_file = f"/content/{dataset\_name.split('/')[-1]}.zip"

if not os.path.exists(zip\_file):

os.system(f"kaggle datasets download -d {dataset\_name} -p /content")

if os.path.exists(zip\_file):

with zipfile.ZipFile(zip\_file, 'r') as zip\_ref:

zip\_ref.extractall(output\_dir)

os.remove(zip\_file)

print(f"Dataset berhasil diekstrak ke: {output\_dir}")

else:

print("File ZIP tidak ditemukan!")

# Unduh dataset

dataset\_name = "vjchoudhary7/customer-segmentation-tutorial-in-python"

output\_dir = "/content/customer-segmentation"

download\_and\_extract\_kaggle\_dataset(dataset\_name, output\_dir)

# Memuat dataset

data\_path = os.path.join(output\_dir, "Mall\_Customers.csv")

if os.path.exists(data\_path):

customer\_data = pd.read\_csv(data\_path)

else:

raise FileNotFoundError("Dataset tidak ditemukan. Periksa path dataset.")

# Eksplorasi Data Awal

print("\n=== Informasi Dataset ===")

print(customer\_data.info())

print("\n=== Statistik Deskriptif ===")

print(customer\_data.describe())

print("\n=== Nilai Hilang ===")

print(customer\_data.isnull().sum())

=== Informasi Dataset ===

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 200 entries, 0 to 199

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 CustomerID 200 non-null int64

1 Gender 200 non-null object

2 Age 200 non-null int64

3 Annual Income (k$) 200 non-null int64

4 Spending Score (1-100) 200 non-null int64

dtypes: int64(4), object(1)

memory usage: 7.9+ KB

None

=== Statistik Deskriptif ===

CustomerID Age Annual Income (k$) Spending Score (1-100)

count 200.000000 200.000000 200.000000 200.000000

mean 100.500000 38.850000 60.560000 50.200000

std 57.879185 13.969007 26.264721 25.823522

min 1.000000 18.000000 15.000000 1.000000

25% 50.750000 28.750000 41.500000 34.750000

50% 100.500000 36.000000 61.500000 50.000000

75% 150.250000 49.000000 78.000000 73.000000

max 200.000000 70.000000 137.000000 99.000000

=== Nilai Hilang ===

CustomerID 0

Gender 0

Age 0

Annual Income (k$) 0

Spending Score (1-100) 0

dtype: int64

# Fungsi untuk EDA

def perform\_eda(df):

# Distribusi kategori

print("\n=== Distribusi Gender ===")

if 'Gender' in df.columns:

print(df['Gender'].value\_counts())

# Histogram untuk fitur numerik

print("\n=== Visualisasi Distribusi ===")

numerical\_features = ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']

for feature in numerical\_features:

if feature in df.columns:

plt.figure(figsize=(6, 4))

sns.histplot(df[feature], kde=True, bins=20, color='blue')

plt.title(f'Distribusi {feature}')

plt.xlabel(feature)

plt.ylabel('Frekuensi')

plt.grid()

plt.show()

# Boxplot untuk outlier

print("\n=== Pemeriksaan Outlier ===")

for feature in numerical\_features:

if feature in df.columns:

plt.figure(figsize=(6, 4))

sns.boxplot(x=df[feature], color='orange')

plt.title(f'Outlier di {feature}')

plt.xlabel(feature)

plt.grid()

plt.show()

# Scatter plot hubungan antar fitur

print("\n=== Hubungan Antar Fitur ===")

if 'Annual Income (k$)' in df.columns and 'Spending Score (1-100)' in df.columns:

plt.figure(figsize=(8, 6))

sns.scatterplot(data=df, x='Annual Income (k$)', y='Spending Score (1-100)', hue='Gender', palette='Set1')

plt.title('Hubungan antara Annual Income dan Spending Score')

plt.xlabel('Annual Income (k$)')

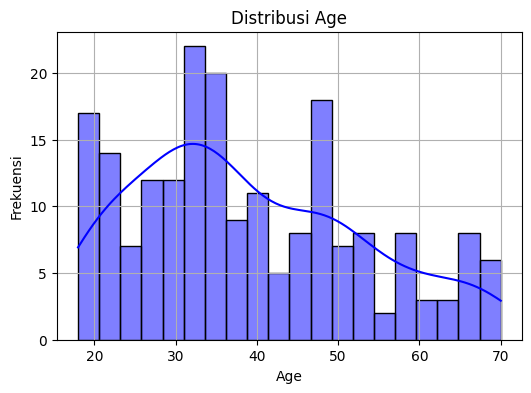
plt.ylabel('Spending Score (1-100)')

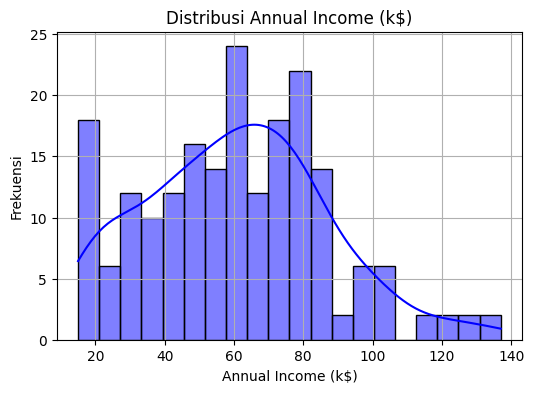
plt.legend(title='Gender')

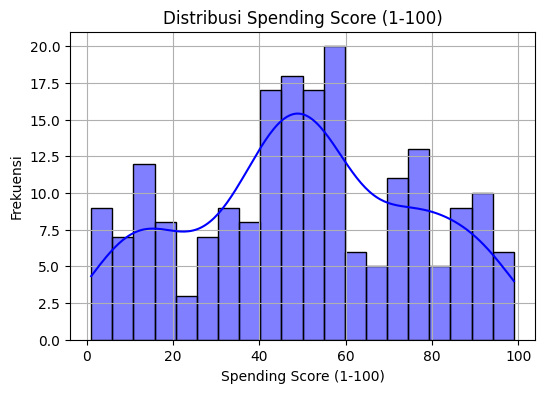
plt.grid()

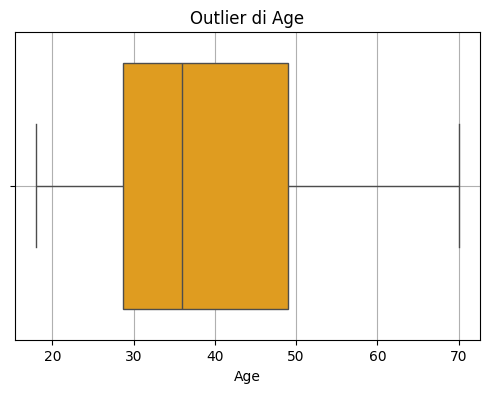
plt.show()

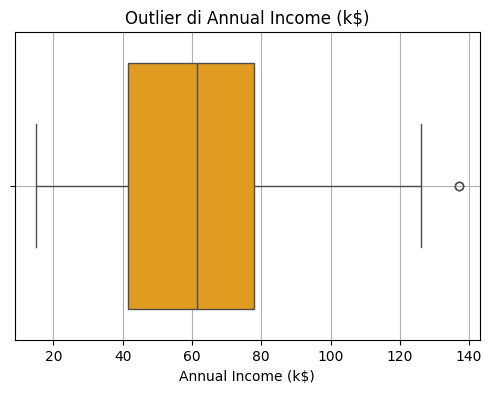
perform\_eda(customer\_data)

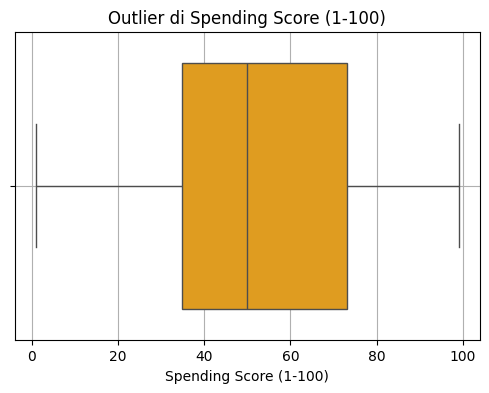


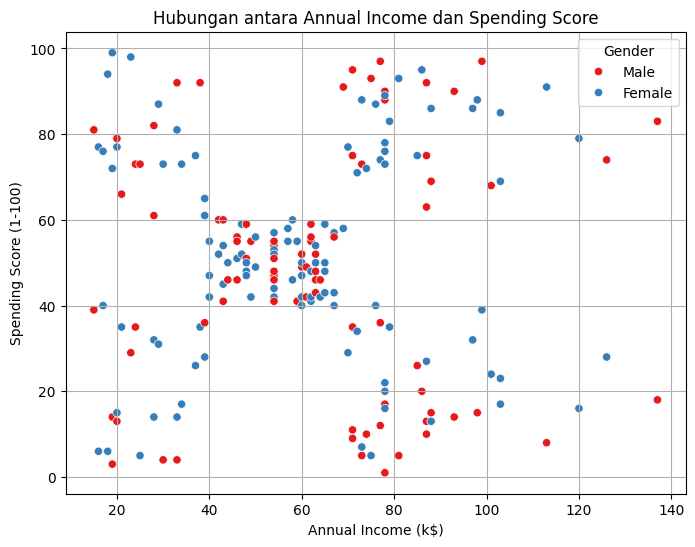












# Ekstraksi fitur

X = customer\_data.iloc[:, [3, 4]].values # 'Annual Income' dan 'Spending Score'

# Menentukan jumlah klaster dengan metode Elbow

def plot\_elbow\_graph(X):

wcss = []

for i in range(1, 11):

kmeans = KMeans(n\_clusters=i, init='k-means++', random\_state=42, n\_init=10)

kmeans.fit(X)

wcss.append(kmeans.inertia\_)

plt.figure(figsize=(8, 5))

plt.plot(range(1, 11), wcss, marker='o', linestyle='--', color='b')

plt.title("The Elbow Point Graph", fontsize=14)

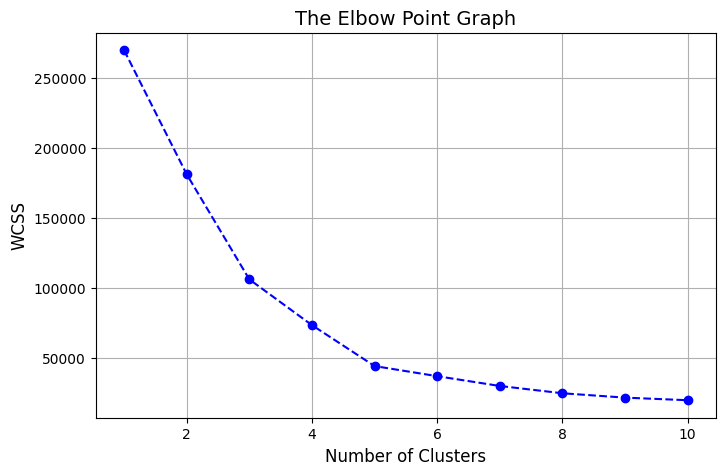
plt.xlabel("Number of Clusters", fontsize=12)

plt.ylabel("WCSS", fontsize=12)

plt.grid()

plt.show()

plot\_elbow\_graph(X)



# K-Means Clustering

optimal\_clusters = 5

kmeans = KMeans(n\_clusters=optimal\_clusters, init='k-means++', random\_state=42, n\_init=10)

y\_kmeans = kmeans.fit\_predict(X)

# Visualisasi Klaster

def plot\_clusters(X, y\_kmeans, kmeans):

plt.figure(figsize=(8, 8))

colors = ['green', 'red', 'yellow', 'violet', 'blue']

for i in range(optimal\_clusters):

plt.scatter(X[y\_kmeans == i, 0], X[y\_kmeans == i, 1], s=50, c=colors[i], label=f'Cluster {i+1}')

# Plotting centroid

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1],

s=200, c='black', marker='X', label="Centroids")

plt.title('Customer Segments', fontsize=16)

plt.xlabel('Annual Income (k$)', fontsize=12)

plt.ylabel('Spending Score (1-100)', fontsize=12)

plt.legend()

plt.grid()

plt.show()

plot\_clusters(X, y\_kmeans, kmeans)

