TUGAS APPLIED MACHINE LEARNING

DISUSUN OLEH :

NAMA : RAMADHINI ANJANI HAMID

NIM : 105841119823

KELAS : 5AI-A

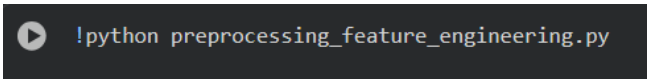
FAKULTAS TEKNIK

PROGRAM STUDI INFORMATIKA

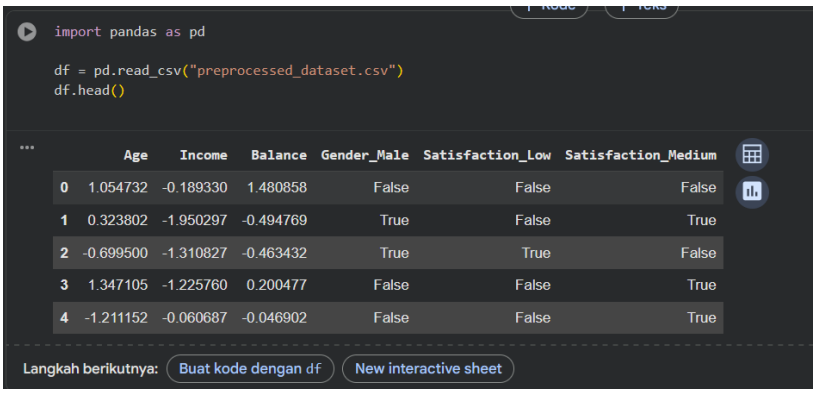
UNIVERSITAS MUHAMMADIYAH MAKASSAR

2025/2026

Penjelasan Tahap – Tahap Preprocessing



Perintah **!python preprocessing_feature_engineering.py** yang terlihat pada gambar merupakan instruksi untuk menjalankan sebuah file Python bernama *preprocessing_feature_engineering.py* di dalam lingkungan notebook seperti Jupyter Notebook atau Google Colab. Tanda seru (!) digunakan untuk mengeksekusi perintah *command line* langsung dari sel notebook. Ketika perintah ini dijalankan, seluruh kode yang ada di dalam file tersebut akan dieksekusi, termasuk proses *data preprocessing* dan *feature engineering* yang sudah disusun sebelumnya—misalnya pembersihan data, normalisasi, encoding, pembuatan fitur baru, atau transformasi lainnya. Dengan mengeksekusi file ini, user dapat menjalankan seluruh rangkaian tahapan preprocessing secara otomatis tanpa harus menuliskan ulang kode di dalam notebook. Ini memastikan workflow menjadi lebih rapi, terstruktur, dan mudah direproduksi.



```
import pandas as pd

df = pd.read_csv("preprocessed_dataset.csv")
df.head()
```

	Age	Income	Balance	Gender_Male	Satisfaction_Low	Satisfaction_Medium
0	1.054732	-0.189330	1.480858	False	False	False
1	0.323802	-1.950297	-0.494769	True	False	True
2	-0.699500	-1.310827	-0.463432	True	True	False
3	1.347105	-1.225760	0.200477	False	False	True
4	-1.211152	-0.060687	-0.046902	False	False	True

Kode pada gambar menunjukkan proses pemuatan dataset hasil preprocessing menggunakan **pandas**. File *preprocessed_dataset.csv* dibaca ke dalam DataFrame dengan pd.read_csv(), lalu df.head() menampilkan lima baris pertama untuk memastikan datanya sudah benar. Dari output terlihat bahwa data telah melalui tahap normalisasi pada fitur numerik dan *one-hot encoding* pada fitur kategorikal, sehingga muncul kolom seperti Gender_Male, Satisfaction_Low, dan Satisfaction_Medium. Tampilan ini memastikan bahwa hasil preprocessing dan feature engineering sudah berhasil diterapkan dan siap digunakan untuk analisis atau pemodelan berikutnya.

```
df.info()
```

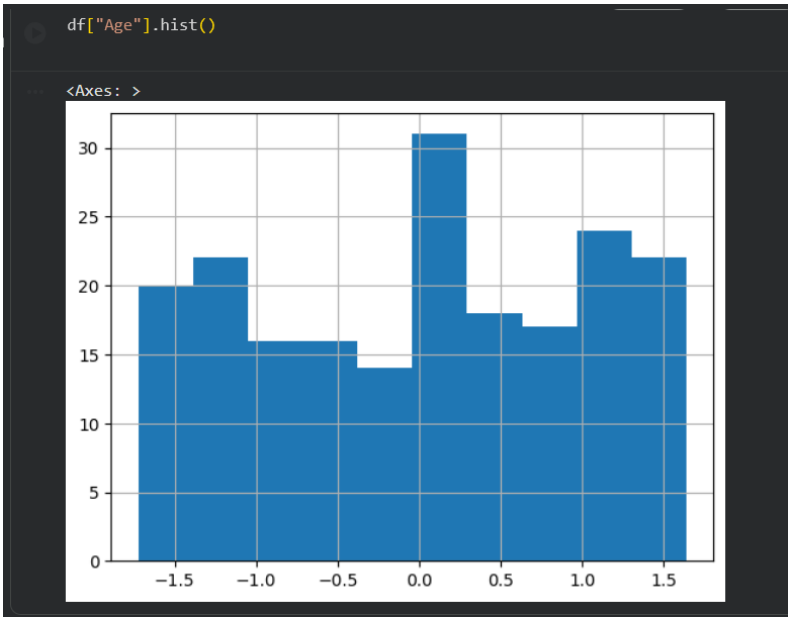
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Age                 200 non-null    float64
 1   Income              200 non-null    float64
 2   Balance             200 non-null    float64
 3   Gender_Male         200 non-null    bool
 4   Satisfaction_Low    200 non-null    bool
 5   Satisfaction_Medium  200 non-null    bool
dtypes: bool(3), float64(3)
memory usage: 5.4 KB
```

Output df.info() menunjukkan ringkasan struktur dataset setelah preprocessing. Dataset terdiri dari **200 baris** dan **6 kolom**, tanpa nilai yang hilang (semua kolom memiliki 200 non-null). Tiga kolom numerik—**Age, Income, Balance**—bertipe *float64*, menandakan bahwa ketiganya sudah melalui proses normalisasi atau standardisasi. Sementara tiga kolom lainnya—**Gender_Male, Satisfaction_Low, Satisfaction_Medium**—bertipe *bool*, hasil dari *one-hot encoding* pada fitur kategorikal. Informasi ini menegaskan bahwa dataset sudah bersih, terstruktur, dan siap digunakan untuk analisis lanjutan atau pemodelan machine learning.

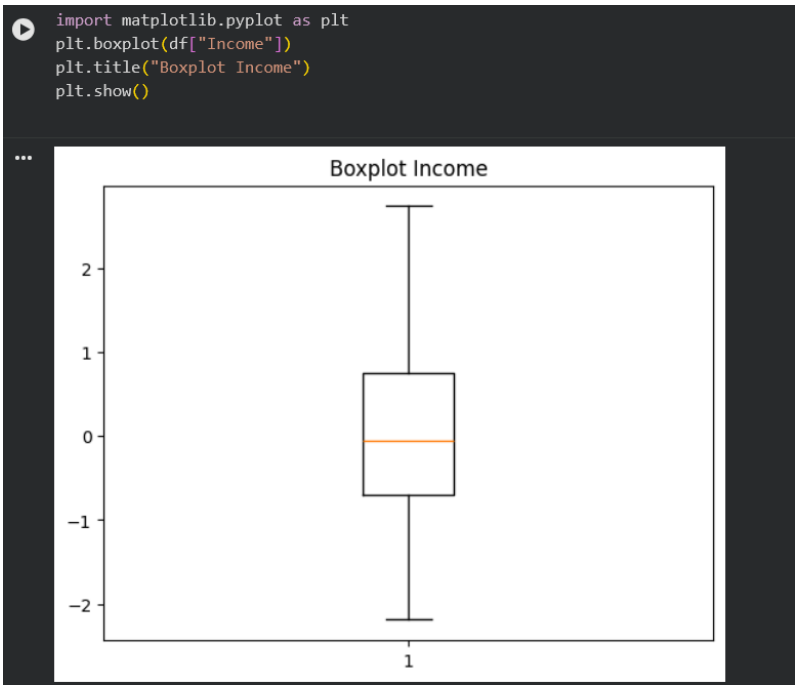
```
df.describe()
```

	Age	Income	Balance
count	2.000000e+02	2.000000e+02	2.000000e+02
mean	-2.220446e-17	1.243450e-16	-5.107026e-16
std	1.002509e+00	1.002509e+00	1.002509e+00
min	-1.722803e+00	-2.188510e+00	-2.495038e+00
25%	-8.456864e-01	-6.998278e-01	-5.506103e-01
50%	3.143000e-02	-6.068735e-02	-5.455799e-02
75%	9.085464e-01	7.583174e-01	7.852624e-01
max	1.639477e+00	2.739992e+00	2.295421e+00

Output df.describe() menampilkan ringkasan statistik untuk tiga fitur numerik: **Age, Income, dan Balance**. Nilai mean yang mendekati nol serta standar deviasi sekitar satu menunjukkan bahwa ketiga fitur tersebut telah melalui proses **standardisasi** sehingga berada pada skala yang sama. Rentang nilai minimum dan maksimum juga konsisten dengan hasil transformasi z-score, yaitu berada di sekitar -2 hingga 2. Selain itu, nilai kuartil (25%, 50%, 75%) menunjukkan distribusi data yang sudah teratur dan tidak memiliki anomali besar. Secara keseluruhan, statistik ini mengonfirmasi bahwa proses normalisasi atau standardisasi telah diterapkan dengan benar dan dataset siap digunakan untuk tahap pemodelan.



Kode df["Age"].hist() digunakan untuk menampilkan histogram dari kolom *Age* pada DataFrame. Grafik yang dihasilkan menunjukkan distribusi nilai usia yang telah distandarisasi, terlihat dari rentang nilai yang berada di sekitar -1.5 hingga 1.5. Histogram ini memperlihatkan bagaimana frekuensi nilai usia tersebar pada berbagai interval, di mana sebagian besar data berada di sekitar nilai 0 yang mewakili nilai rata-rata setelah proses standardisasi. Area dengan nilai ekstrem terlihat memiliki frekuensi yang lebih rendah, sehingga grafik ini membantu memberikan gambaran mengenai pola penyebaran data usia dalam bentuk visual yang mudah dipahami.

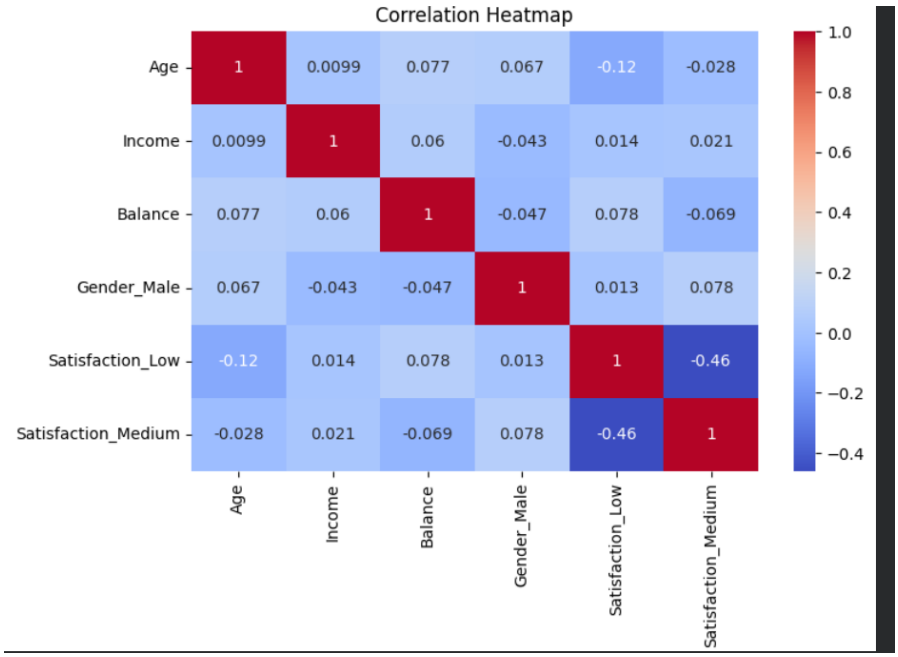


Kode pada gambar menggunakan matplotlib.pyplot untuk menampilkan boxplot dari kolom **Income** pada DataFrame. Baris plt.boxplot(df["Income"]) menghasilkan grafik boxplot yang menggambarkan penyebaran data Income, sementara plt.title("Boxplot Income")

memberi judul pada grafik, dan plt.show() menampilkannya. Output berupa boxplot menunjukkan nilai tengah (median), rentang kuartil (Q1–Q3), serta batas bawah dan atas data. Karena nilai Income tampak berada pada rentang sekitar -2 hingga 2, dapat disimpulkan bahwa data telah distandarisasi. Boxplot ini membantu melihat persebaran data, mendeteksi pencilan, serta mengetahui apakah distribusi Income cenderung simetris atau miring.

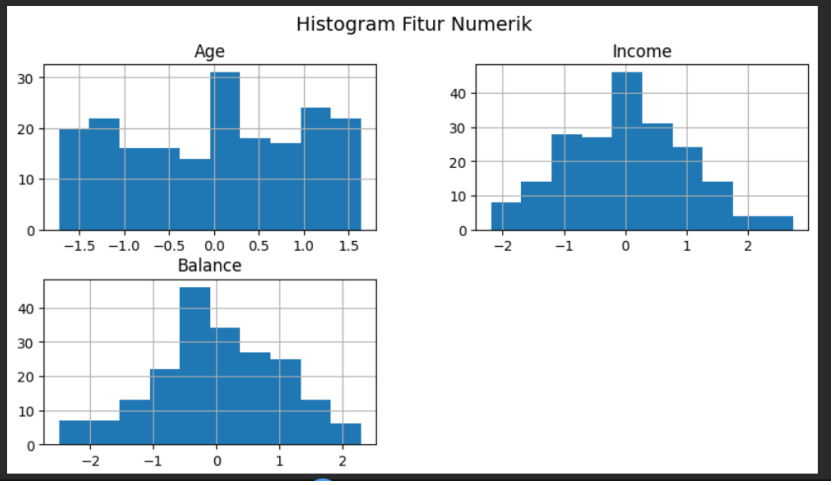
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```



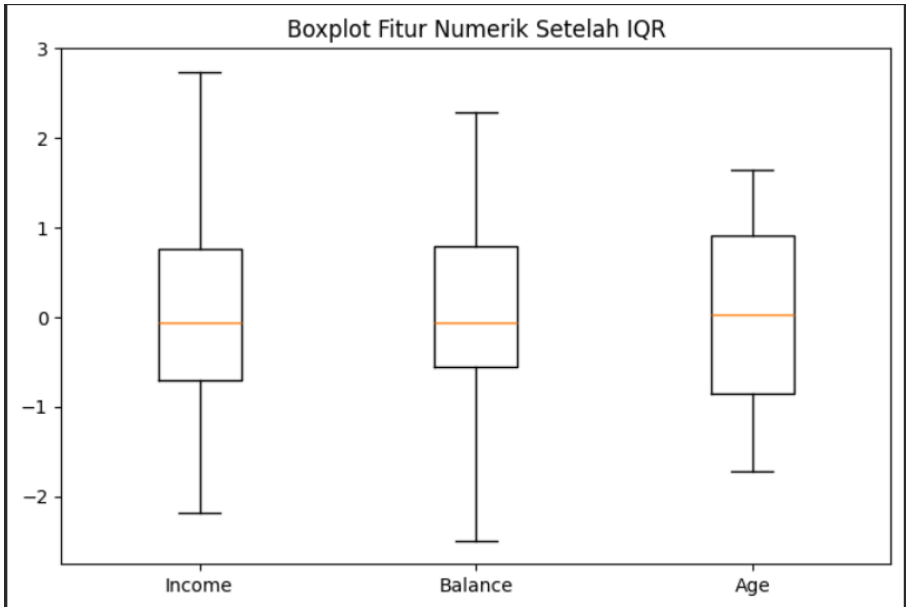
Kode tersebut digunakan untuk menampilkan **heatmap korelasi** pada dataset. Baris plt.figure(figsize=(8,5)) mengatur ukuran gambar, sementara sns.heatmap(df.corr(), annot=True, cmap="coolwarm") menghasilkan heatmap berdasarkan nilai korelasi antar variabel numerik dalam DataFrame. Parameter annot=True membuat setiap sel menampilkan angka korelasinya, dan cmap="coolwarm" memberi warna gradasi dari merah (korelasi tinggi) hingga biru (korelasi negatif). Output grafik menunjukkan hubungan antar variabel, misalnya *Age* memiliki korelasi negatif kecil dengan *Satisfaction_Low*, dan variabel *Satisfaction_Low* serta *Satisfaction_Medium* memiliki korelasi negatif kuat sekitar -0.46. Heatmap ini memudahkan untuk melihat pola hubungan antar fitur secara visual dan cepat.

```
df[["Age", "Income", "Balance"]].hist(figsize=(10, 5))
plt.suptitle("Histogram Fitur Numerik", fontsize=14)
plt.show()
```



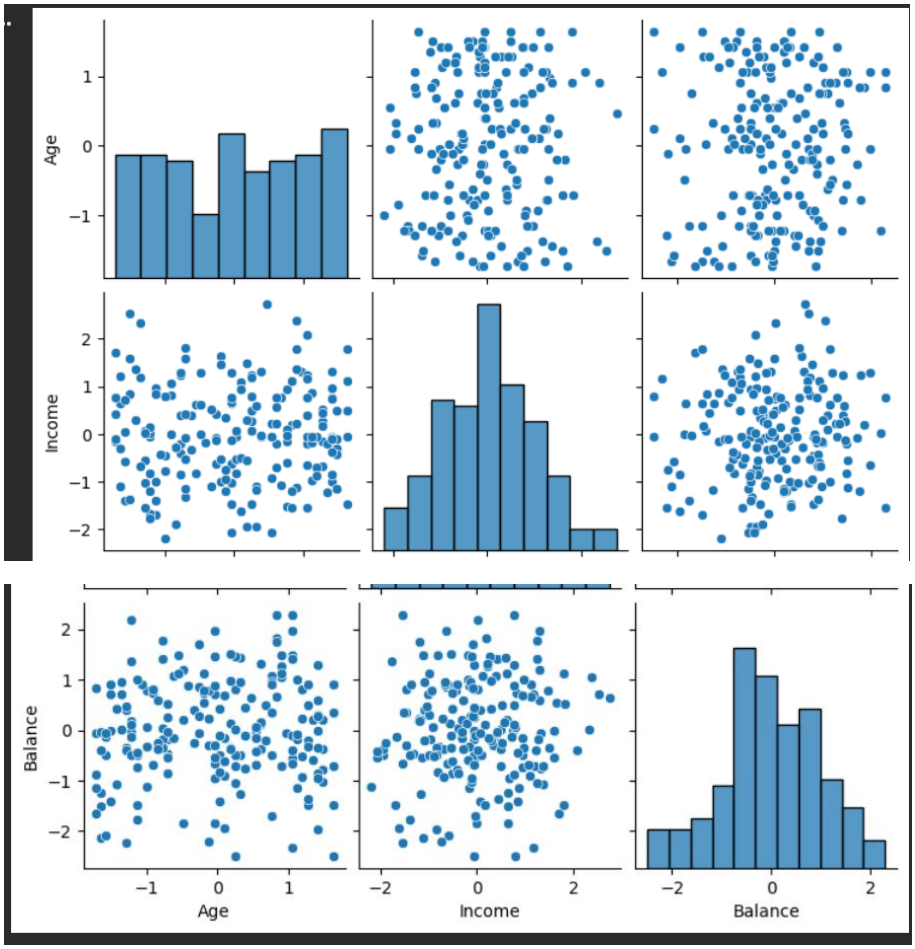
Kode tersebut digunakan untuk menampilkan **histogram beberapa fitur numerik sekaligus**, yaitu *Age*, *Income*, dan *Balance*. Perintah df[["Age", "Income", "Balance"]].hist(figsize=(10, 5)) membuat tiga histogram dalam satu tampilan dengan ukuran gambar 10×5. Selanjutnya, plt.suptitle("Histogram Fitur Numerik", fontsize=14) menambahkan judul utama untuk seluruh subplot, dan plt.show() menampilkan grafiknya. Output yang dihasilkan berupa tiga histogram terpisah yang menunjukkan distribusi masing-masing fitur numerik. Ketiganya menampilkan pola penyebaran data yang tampak sudah melalui proses standardisasi, karena nilai berada di sekitar rentang –2 hingga 2. Histogram ini membantu memahami bentuk distribusi setiap variabel, apakah datanya simetris, miring, atau tersebar secara merata.

```
plt.figure(figsize=(8,5))
plt.boxplot([df["Income"], df["Balance"], df["Age"]], labels=["Income", "Balance", "Age"])
plt.title("Boxplot Fitur Numerik Setelah IQR")
plt.show()
```



Kode tersebut digunakan untuk menampilkan **boxplot** dari tiga fitur numerik dalam dataframe, yaitu *Income*, *Balance*, dan *Age*, setelah proses pembersihan outlier menggunakan metode **IQR**. Baris plt.boxplot([df["Income"], df["Balance"], df["Age"]], labels=["Income", "Balance", "Age"]) membuat tiga boxplot berdampingan, sementara plt.figure(figsize=(8,5)) mengatur ukuran gambar dan plt.title() memberi judul grafik. Output yang dihasilkan berupa visualisasi persebaran data tiap fitur, menunjukkan nilai median, rentang kuartil (IQR), dan titik ekstrim. Grafik ini membantu melihat apakah outlier sudah berkurang dan bagaimana distribusi masing-masing variabel setelah preprocessing dilakukan.

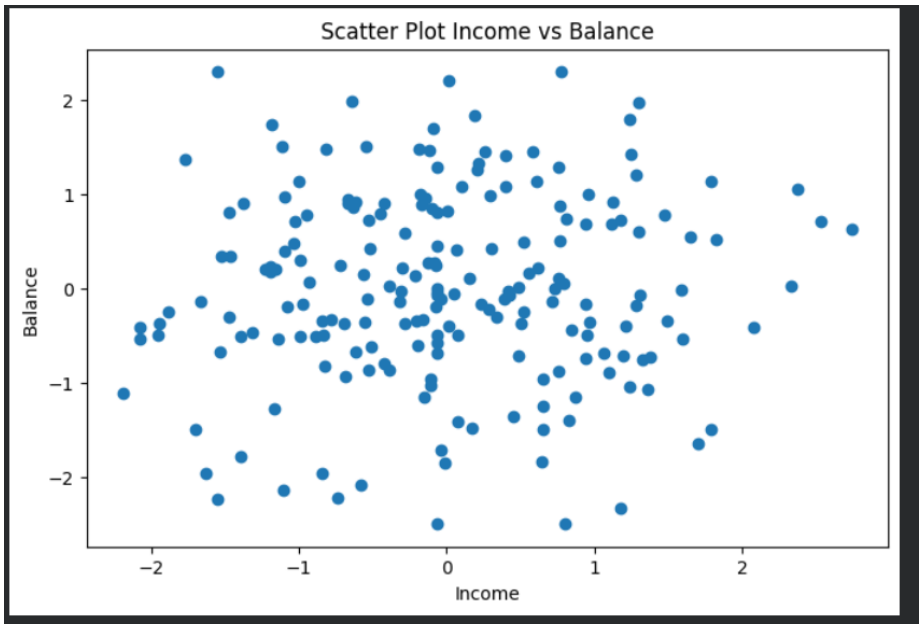
```
sns.pairplot(df[["Age", "Income", "Balance"]])
plt.show()
```



Kode tersebut menggunakan sns.pairplot() untuk menampilkan hubungan antar tiga fitur numerik—**Age**, **Income**, dan **Balance**—dalam bentuk scatterplot dan histogram. Pairplot secara otomatis membuat *scatterplot* untuk setiap pasangan variabel, serta *histogram* pada diagonal untuk menunjukkan distribusi masing-masing fitur. Output yang muncul menunjukkan bahwa ketiga variabel memiliki sebaran data yang cukup acak tanpa pola hubungan linear yang kuat, sehingga tidak tampak korelasi yang signifikan antar fitur. Histogram pada diagonal menggambarkan persebaran masing-masing variabel setelah proses standardisasi, sehingga bentuk distribusi terlihat lebih teratur dan berada dalam skala yang

sama. Visualisasi ini membantu memahami struktur data serta potensi hubungan antar variabel sebelum dilakukan analisis lebih lanjut.

```
plt.figure(figsize=(8,5))
plt.scatter(df["Income"], df["Balance"])
plt.xlabel("Income")
plt.ylabel("Balance")
plt.title("Scatter Plot Income vs Balance")
plt.show()
```

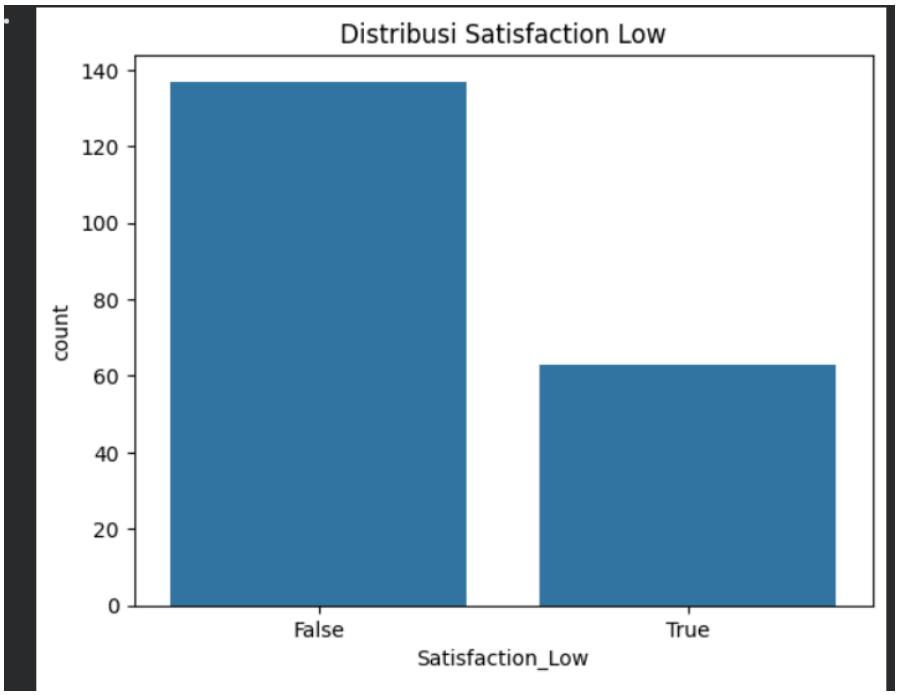


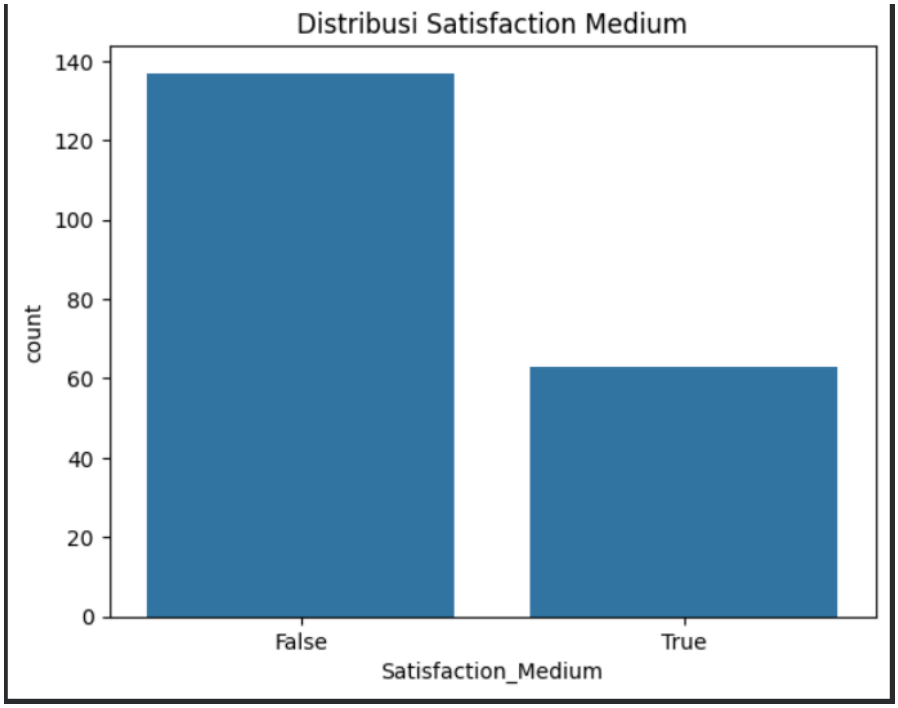
Kode tersebut membuat visualisasi *scatter plot* menggunakan Matplotlib untuk melihat hubungan antara variabel **Income** dan **Balance** dalam sebuah *dataframe*. Baris plt.figure(figsize=(8,5)) menentukan ukuran gambar, lalu plt.scatter(df["Income"], df["Balance"]) menampilkan titik-titik yang mewakili pasangan nilai Income dan Balance. Selanjutnya, plt.xlabel("Income") dan plt.ylabel("Balance") memberi label pada sumbu X dan Y, sementara plt.title("Scatter Plot Income vs Balance") menambahkan judul pada grafik. Terakhir, plt.show() digunakan untuk menampilkan plot tersebut.

Output berupa grafik *scatter plot* yang menunjukkan sebaran titik data antara income dan balance. Dari visualisasi terlihat bahwa titik-titik tersebar acak tanpa pola tertentu, menandakan bahwa tidak ada hubungan atau korelasi yang kuat antara pendapatan (*Income*) dan saldo (*Balance*) pada dataset tersebut.

```
sns.countplot(x="Satisfaction_Low", data=df)
plt.title("Distribusi Satisfaction Low")
plt.show()

sns.countplot(x="Satisfaction_Medium", data=df)
plt.title("Distribusi Satisfaction Medium")
plt.show()
```





Kode tersebut menggunakan sns.countplot() untuk menampilkan distribusi dua variabel kategorikal biner, yaitu **Satisfaction_Low** dan **Satisfaction_Medium**. Masing-masing plot

menghitung jumlah data dengan nilai **True** dan **False** untuk setiap kategori, lalu menampilkannya dalam bentuk grafik batang. Fungsi plt.title() memberikan judul pada setiap grafik, dan plt.show() digunakan untuk menampilkannya.

Output pertama menunjukkan distribusi **Satisfaction_Low**, di mana jumlah nilai **False** jauh lebih banyak dibanding **True**, menandakan bahwa sebagian besar pelanggan *tidak* termasuk kategori kepuasan rendah. Output kedua menampilkan distribusi **Satisfaction_Medium**, dengan pola serupa: nilai **False** lebih dominan dibanding **True**, sehingga mayoritas pelanggan *tidak* berada pada kategori kepuasan sedang. Grafik-grafik ini membantu memahami persebaran tingkat kepuasan dalam dataset secara visual.