

TUGAS 2: PRAKTIKUM

Rika Rahma - 0110222134¹

¹ Teknik Informatika, STT Terpadu Nurul Fikri, Depok

*E-mail: rika22134ti@student.nurulfikri.ac.id

Abstract. Pada praktikum ini dipelajari penggunaan Google Colab sebagai lingkungan pengembangan Machine Learning, serta penerapan konsep statistik deskriptif dan probabilitas dasar untuk analisis data. Dataset yang digunakan adalah *500_Person_Gender_Height_Weight_Index.csv* yang berisi data jenis kelamin, tinggi badan, berat badan, dan indeks kesehatan. Analisis meliputi perhitungan nilai sentral (mean, median, modus), ukuran penyebaran (variansi, standar deviasi, kuartil, IQR), serta korelasi antar variabel. Selain itu dilakukan visualisasi data dengan boxplot, histogram, dan scatter plot. Praktikum juga mencakup pembagian dataset *day.csv* menjadi data training, validation, dan testing. Hasil menunjukkan bahwa Google Colab memudahkan proses analisis data dengan Python, serta visualisasi dapat membantu memahami distribusi dan hubungan antar variabel.

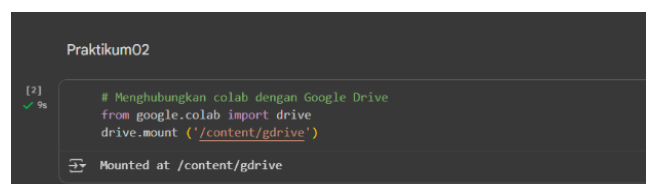
1. Pendahuluan

Sebelum membangun model Machine Learning, langkah awal yang penting adalah memahami data melalui statistik deskriptif dan probabilitas. Statistik deskriptif membantu memahami nilai sentral dan penyebaran data, sementara probabilitas digunakan untuk memprediksi kemungkinan suatu kejadian. Praktikum ini bertujuan melatih mahasiswa menggunakan Google Colab, Pandas, Matplotlib, dan Seaborn untuk menganalisis data serta menyiapkan dataset untuk Machine Learning.

2. Praktikum

2.1 Menghubungkan Google Colab dengan Google Drive

- Menghubungkan Google Colab dengan Google Drive
Kode:



```
Praktikum02

[?]
✓ %
# Menghubungkan colab dengan Google Drive
from google.colab import drive
drive.mount('/content/gdrive')

Mounted at /content/gdrive
```

Gambar 1. Menghubungkan Colab dengan Google Drive

- Kode `from google.colab import drive` → mengimpor modul drive dari package google.colab yang menyediakan fungsi mount untuk mengakses Drive.

- Perintah `drive.mount('/content/gdrive')` → menghubungkan Google Colab dengan akun Google Drive kita, sehingga file yang ada di Drive dapat diakses langsung melalui Colab.
 - Setelah dijalankan, Colab menampilkan link otorisasi dan/atau pesan Mounted at `/content/gdrive`.
- Membaca Dataset dari Google Drive
Input:

```
[3] # Memanggil dataset lewat Gdrive
path = '/content/gdrive/MyDrive/Praktikum_ML/praktikum02/data/'

[4] # Membaca file csv menggunakan pandas
import pandas as pd

df = pd.read_csv(path + '500_Person_Gender_Height_Weight_Index.csv')
df
```

Gambar 2. Memanggil Dataset

Output:

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3
4	Male	149	61	3
...
495	Female	150	153	5
496	Female	184	121	4
497	Female	141	136	5
498	Male	150	95	5
499	Male	173	131	5

500 rows x 4 columns

Next steps: [Generate code with df](#) [New interactive sheet](#)

Gambar 3. Output Gender, Height, Weight and Index

Penjelasan Kode:

- `path = '...'` → menyimpan lokasi folder di Google Drive agar path mudah diubah dan dipakai berulang.
- `import pandas as pd` → mengimpor library Pandas, standar untuk manipulasi data tabular di Python.
- `pd.read_csv(path + '...csv')` → membaca file CSV menjadi DataFrame bernama `df`.
- Fungsi `pd.read_csv()` membaca file CSV bernama `500_Person_Gender_Height_Weight_Index.csv` dan menyimpannya ke dalam DataFrame `df`.
- Perintah `df` atau `df.head()` menampilkan data, yang pada output terlihat terdiri dari 4 kolom yaitu: *Gender*, *Height*, *Weight*, dan *Index*.
 - Kolom *Gender*: jenis kelamin (Male/Female).
 - Kolom *Height*: tinggi badan (cm).
 - Kolom *Weight*: berat badan (kg).
 - Kolom *Index*: indeks kesehatan.

2.2 Analisis Statistik Deskriptif

1) Melihat Informasi Umum Data

Kode:

```
Analisis Statistik Deskriptif

1. Melihat Informasi Umum Data

[5] ✓ Os # Mencari info data pada file (tipe datanya, not nul count data, nama kolom)
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
#   Column  Non-Null Count  Dtype  
---  -
0   Gender   500 non-null     object  
1   Height   500 non-null     int64   
2   Weight   500 non-null     int64   
3   Index    500 non-null     int64   
dtypes: int64(3), object(1)
memory usage: 15.8+ KB
```

Gambar 4. Melihat Informasi Umum Data

Penjelasan:

- o Jumlah baris data: 500 entri
- o Jumlah kolom: 4 (Gender, Height, Weight, Index)
- o Semua kolom non-null (tidak ada data kosong)
- o Tipe data: Gender (object), Height (int64), Weight (int64), Index (int64)
- o Kode df.info() digunakan untuk melihat ringkasan dataset.
- o Sesuai komentarnya, fungsi ini memberikan informasi tentang tipe data, jumlah data yang tidak kosong, serta nama-nama kolom.
- o Dari hasil output, dataset ini lengkap (tanpa missing value), sehingga bisa langsung digunakan untuk analisis statistik.

2) Menghitung Nilai – Nilai Sentral (Mean, Median, Modus)

Kode:

```
2. Menghitung Nilai-Nilai Sentral (Mean, Median, Modus):

[6] ✓ Os # Menghitung mean semua kolom numerik
df['Height'].mean()

np.float64(169.944)

[7] ✓ Os # Menghitung median semua kolom numerik
df['Height'].median()

170.5

[8] ✓ Os # Mencari modus hati-hati karena bisa lebih dari satu
df['Height'].mode()

Height
0      188
dtype: int64
```

Gambar 5. Menghitung Nilai-Nilai Sentral

Penjelasan:

- o Mean (Rata-rata):
 - o df['Height'].mean() menghitung rata-rata dari kolom *Height*.
 - o Nilainya adalah 169.944 cm, artinya rata-rata tinggi badan dari seluruh data sekitar 170 cm.
- o Median (Nilai Tengah):
 - o df['Height'].median() mencari nilai yang tepat di tengah jika data diurutkan.
 - o Nilainya 170.5 cm, menunjukkan bahwa separuh data memiliki tinggi ≤ 170.5 cm, dan separuh lainnya ≥ 170.5 cm.
- o Mode (Modus):
 - o df['Height'].mode() mencari nilai yang paling sering muncul.

- Hasilnya 188 cm, artinya tinggi badan yang paling sering muncul dalam dataset adalah 188 cm.
- 3) Menghitung Ukuran Persebaran (Variansi & Standar Deviasi)

Kode:

```

3. Menghitung Ukuran Persebaran (Variansi & Standar Deviasi):

[9] ✓ Os
# Menghitung Variansi & Standard Deviasi
df.var(numeric_only=True)

Height    268.149162
Weight    1048.633267
Index      1.836168
dtype: float64

[10] ✓ Os
# Menghitung Standar Deviasi
df.std(numeric_only=True)

Height     16.375261
Weight     32.382607
Index       1.355053
dtype: float64

```

Gambar 6. Menghitung Ukuran Persebaran

Penjelasan :

- `df.var()` menghitung variansi, yaitu ukuran penyebaran data terhadap rata-rata. Semakin besar nilainya, semakin besar pula sebaran datanya.
- `df.std()` menghitung standar deviasi, yaitu akar dari variansi, dengan satuan sama seperti data aslinya.
- Dari output terlihat bahwa:
 - Height memiliki standar deviasi sekitar 16.37 cm, artinya tinggi badan menyebar sekitar ± 16 cm dari rata-rata.
 - Weight memiliki standar deviasi sekitar 32.38 kg, menunjukkan penyebaran berat badan cukup besar.
 - Index hanya memiliki standar deviasi 1.35, menandakan penyebarannya relatif kecil.

4) Menghitung Kuartil

Kode:

```

4. Menghitung Kuartil

[11] ✓ Os
# Menghitung kuartil pertama (Q1)
q1 = df['Height'].quantile(0.25)
print("Q1 : ", q1)

# Menghitung kuartil ketiga (Q3)
q3 = df['Height'].quantile(0.75)
print("Q3 : ", q3)

# Menghitung IQR (Interquartile Range)
iqr = q3 - q1
print("IQR : ", iqr)

Q1 : 156.0
Q3 : 184.0
IQR : 28.0

```

Gambar 7. Menghitung Kuartil

Penjelasan :

- `df['Height']` → Mengambil kolom Height dari DataFrame `df`.
- `.quantile(0.25)` → Menghitung nilai kuartil ke-1 (Q1), yaitu nilai pada 25% data terbawah setelah data diurutkan.
- `q1 = ...` → Menyimpan hasilnya ke variabel `q1`.
- `print("Q1 : ", q1)` → Menampilkan nilai Q1 di layar.
- `.quantile(0.75)` → Menghitung kuartil ke-3 (Q3), yaitu nilai pada 75% data terbawah atau 25% data teratas setelah data diurutkan.
- `q3 = ...` → Disimpan dalam variabel `q3`.
- `print("Q3 : ", q3)` → Menampilkan nilai Q3.
- `iqr = q3 - q1` → IQR dihitung dengan selisih kuartil ketiga dan kuartil pertama.
- `print("IQR : ", iqr)` → Menampilkan nilai IQR.
- Q1 (Kuartil 1) menunjukkan nilai pada posisi 25% data terbawah, yaitu 156 cm. Artinya 25% data tinggi badan berada di bawah 156 cm.
- Q3 (Kuartil 3) menunjukkan nilai pada posisi 75% data terbawah, yaitu 184 cm. Artinya 75% data berada di bawah 184 cm.
- IQR (Interquartile Range) dihitung dengan rumus $Q3 - Q1 = 28$ cm. IQR ini menggambarkan rentang data di bagian tengah (50% data), yaitu antara 156 cm sampai 184 cm.

5) Menghitung Statistik Deskriptif Otomatis

Kode:

5. Menghitung Statistik Deskriptif Otomatis:

```
[12]
✓ Os # Untuk membuat statistika deskripsi pada type data int
df.describe()
```

	Height	Weight	Index
count	500.000000	500.000000	500.000000
mean	169.944000	106.000000	3.748000
std	16.375261	32.382607	1.355053
min	140.000000	50.000000	0.000000
25%	156.000000	80.000000	3.000000
50%	170.500000	106.000000	4.000000
75%	184.000000	136.000000	5.000000
max	199.000000	160.000000	5.000000

Gambar 8. Menghitung Statistik Deskriptif

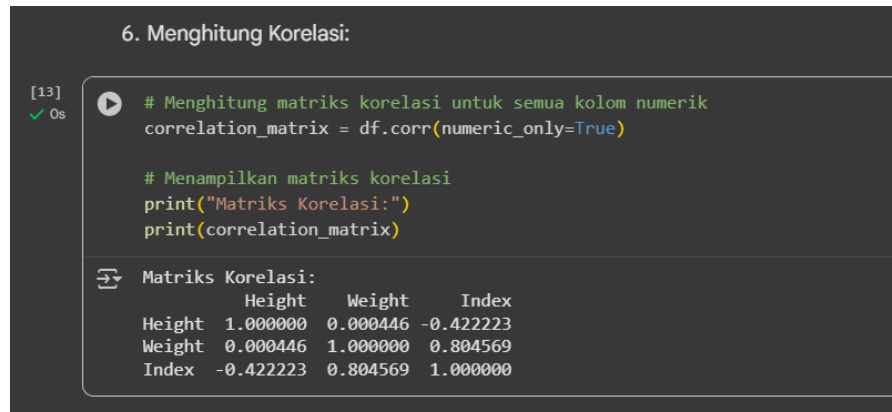
Penjelasan:

- Fungsi `df.describe()` memberikan ringkasan statistik otomatis untuk semua kolom numerik.
- Informasi yang ditampilkan mencakup jumlah data (`count`), nilai rata-rata (`mean`), standar deviasi (`std`), nilai minimum (`min`), kuartil 25%, 50% (median), 75%, dan nilai maksimum (`max`).
- Dari hasil:

- Tinggi badan (Height) berkisar antara 140 cm – 199 cm, dengan rata-rata sekitar 170 cm.
- Berat badan (Weight) antara 50 kg – 160 kg, dengan rata-rata sekitar 106 kg.
- Index berkisar 0 – 5, dengan median 4.

6) Menghitung Korelasi

Kode:



```

6. Menghitung Korelasi:

[13] ✓ 0s # Menghitung matriks korelasi untuk semua kolom numerik
correlation_matrix = df.corr(numeric_only=True)

# Menampilkan matriks korelasi
print("Matriks Korelasi:")
print(correlation_matrix)

Matriks Korelasi:
      Height  Weight  Index
Height  1.000000  0.000446 -0.422223
Weight  0.000446  1.000000  0.804569
Index   -0.422223  0.804569  1.000000

```

Gambar 9. Menghitung Korelasi

Penjelasan:

- `df.corr(numeric_only=True)` Menghitung koefisien korelasi Pearson antar semua kolom numerik di `df`.
- `numeric_only=True` memastikan hanya kolom bertipe numerik yang dihitung (menghindari error jika ada kolom teks).
- `correlation_matrix` = Menyimpan hasil matriks korelasi ke variabel `correlation_matrix`.
- `print(...)` Menampilkan teks penjelasan dan matriks korelasi di output notebook agar bisa di-screenshot sebagai bukti.

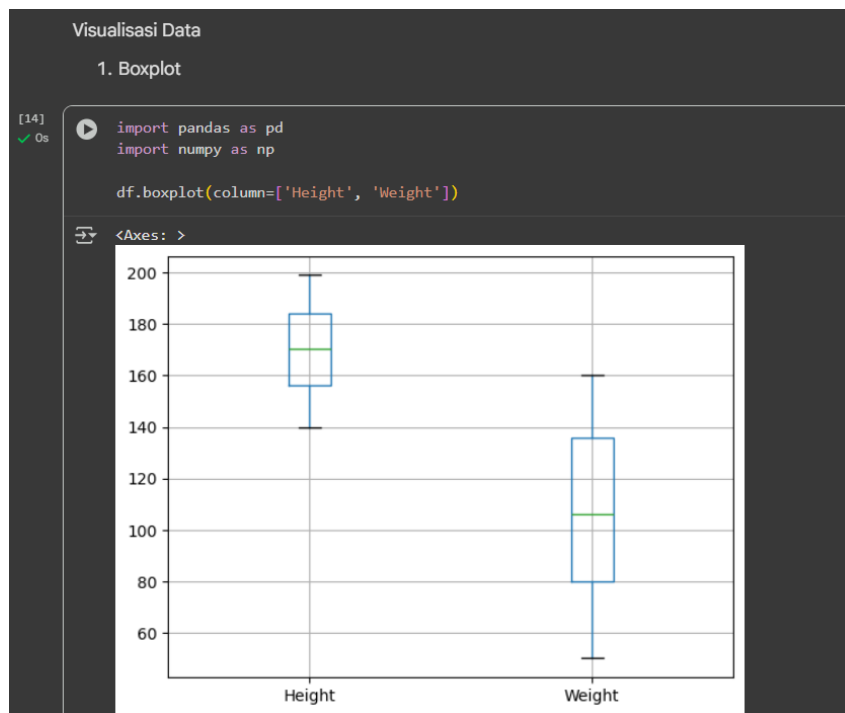
Output:

- Height vs Weight = 0.000446
Nilai ~ 0 → tidak ada korelasi linier antara tinggi dan berat pada dataset ini (secara linear tidak saling berkaitan).
- Height vs Index = -0.422223
Nilai negatif (sekitar -0.42) → ada korelasi negatif sedang: ketika Height meningkat, Index cenderung turun (hubungan linier berarah berlawanan).
- Weight vs Index = 0.804569
Nilai ~ 0.80 → korelasi positif kuat: ketika Weight meningkat, Index juga cenderung meningkat secara linier.

2.3 Visualisasi Data

1) Boxplot

Kode:



Gambar 10. Boxplot

Penjelasan:

- `df.boxplot(column=['Height', 'Weight'])`
- Membuat boxplot untuk kolom Height (tinggi badan) dan Weight (berat badan). Boxplot digunakan untuk melihat distribusi data, nilai tengah (median), serta mendeteksi outlier (jika ada titik di luar “whisker”).
- Hasil boxplot menampilkan dua grafik:
 - Sumbu vertikal = nilai tinggi/berat
 - Sumbu horizontal = nama kolom (Height dan Weight)
- Height (Tinggi Badan):
 - Median (garis hijau di dalam box) sekitar 170 cm.
 - Sebagian besar data (Q1 – Q3) berada dalam rentang 156 cm – 184 cm.
 - Whisker bawah sekitar 140 cm dan whisker atas sekitar 200 cm.
 - Tidak terlihat outlier yang ekstrem (titik di luar whisker).
- Weight (Berat Badan):
 - Median sekitar 106 kg.
 - Data tengah (Q1 – Q3) berada di kisaran 80 kg – 136 kg.
 - Whisker bawah sekitar 50 kg dan whisker atas sekitar 160 kg.
 - Distribusi data terlihat lebih lebar dibanding Height → menunjukkan variasi berat badan lebih besar.

2) Histogram

```
2. Histogram

[16] ✓ 0s ▶ import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Mengambil data Height
data_height = df["Height"]

# Membuat Histogram
n, bins, patches = plt.hist(data_height, bins=5, color='pink', edgecolor='black')

# Menambahkan Label
plt.title('Histogram Nilai')
plt.xlabel('Height')
plt.ylabel('Frekuensi')

# Menampilkan rentang frekuensi di sumbu x
bin_centers = 0.5 * (bins[:-1] + bins[1:])
plt.xticks(bin_centers, ['{:0.0f}-{:0.0f}'.format(bins[i], bins[i+1]) for i in range(len(bins)-1)])

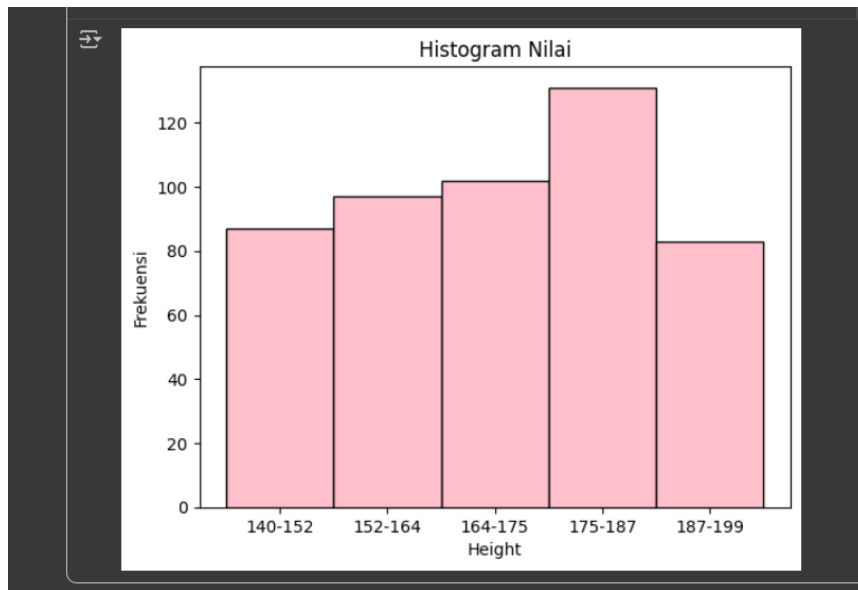
# Menampilkan Histogram
plt.show()
```

Gambar 11. Kode Histogram

Penjelasan:

- Import library
 - numpy, matplotlib.pyplot, dan pandas digunakan untuk mengolah data serta membuat visualisasi histogram.
- Ambil data kolom Height
 - `data_height = df["Height"]` → hanya mengambil kolom tinggi badan dari dataset.
- Membuat histogram
 - `plt.hist(data_height, bins=5, color='pink', edgecolor='black')`
 - `bins=5` artinya data dibagi menjadi 5 kelompok interval.
 - Warna batang histogram pink dengan garis tepi hitam.
 - Nilai `n`, `bins`, `patches` menyimpan informasi jumlah data per interval, batas interval, dan objek batang histogram.
- Menambahkan label grafik
 - `plt.title('Histogram Nilai')` → judul grafik.
 - `plt.xlabel('Height')` → label sumbu X (tinggi badan).
 - `plt.ylabel('Frekuensi')` → label sumbu Y (jumlah data per interval).
- Menampilkan rentang interval di sumbu X
 - Menghitung titik tengah tiap bin: `bin_centers = 0.5 * (bins[:-1] + bins[1:])`.
 - `plt.xticks(...)` mengganti angka di sumbu X dengan format rentang, misalnya 140–156, 156–170, dst.
- Menampilkan grafik
 - `plt.show()` → menampilkan histogram di output notebook.

Output:



Gambar 12. Output Histogram Nilai

Penjelasan:

- "Histogram Nilai" → menunjukkan bahwa grafik ini menampilkan distribusi nilai (dalam hal ini tinggi/Height).
- Sumbu X (Horizontal):
Labelnya berupa rentang tinggi badan (Height) yang dibagi dalam interval:
 - 140–152
 - 152–164
 - 164–175
 - 175–187
 - 187–199
- Jadi data tinggi badan dikelompokkan ke dalam 5 kelas/rentang.
- Sumbu Y (Vertikal):
Berlabel "Frekuensi" → menunjukkan jumlah data (berapa banyak orang) yang masuk ke tiap rentang tinggi badan.
- Batang Histogram:
 - Rentang 140–152: frekuensinya sekitar 87 orang
 - Rentang 152–164: frekuensinya sekitar 97 orang
 - Rentang 164–175: frekuensinya sekitar 102 orang
 - Rentang 175–187: frekuensinya paling tinggi, sekitar 130 orang
 - Rentang 187–199: frekuensinya sekitar 83 orang
- Interpretasi:
 - Sebagian besar orang berada pada rentang tinggi 175–187 cm (paling banyak).
 - Paling sedikit ada di rentang 187–199 cm.
 - Distribusi data terlihat cukup merata, tapi cenderung menumpuk di tengah (164–187 cm).

3) Scatter Plot (Hubungan Antar Variabel)

Kode:

```
3. Scatter Plot (Hubungan Antar Variabel):

[17] ✓ 0s ▶ import pandas as pd
import matplotlib.pyplot as plt

# Membuat Dataframe contoh
data = {
    'Nilai1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Nilai2': [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
}

df2 = pd.DataFrame(data)

# Membuat scatter plot
plt.scatter(df2['Nilai1'], df2['Nilai2'], color='blue', marker='o')

# Menambahkan Label
plt.title('Scatter Plot Korelasi Positif')
plt.xlabel('Nilai1')
plt.ylabel('Nilai2')

# Menambahkan Grid
plt.grid(True)

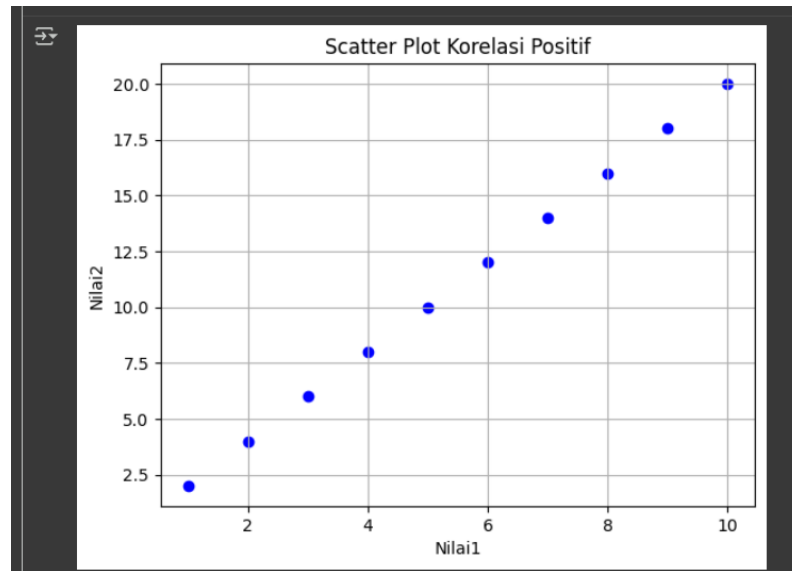
# Menampilkan plot
plt.show()
```

Gambar 13. Kode Scatter Plot Korelasi Positif

Penjelasan:

- "Scatter Plot Korelasi Positif" → menunjukkan grafik ini digunakan untuk melihat hubungan antar dua variabel.
- Import library
 - pandas untuk membuat DataFrame.
 - matplotlib.pyplot untuk membuat grafik scatter plot.
- Membuat dataset contoh
 - Nilai1 berisi angka 1–10.
 - Nilai2 berisi kelipatan 2 dari Nilai1 → sehingga ada hubungan linier sempurna.
 - Data ini disimpan dalam DataFrame df2.
- Membuat scatter plot
 - `plt.scatter(df2['Nilai1'], df2['Nilai2'], color='blue', marker='o')`
 - Membuat grafik sebaran antara Nilai1 (sumbu X) dan Nilai2 (sumbu Y).
 - Titik ditampilkan dengan warna biru (blue) dan bentuk lingkaran (marker='o').
- Memberikan label & grid
 - `plt.title(...)`, `plt.xlabel(...)`, `plt.ylabel(...)` menambahkan judul grafik serta label sumbu.
 - `plt.grid(True)` menambahkan garis bantu (grid).
- Menampilkan grafik
 - `plt.show()` → menampilkan scatter plot ke output.

Output:



Gambar 14. Output Scatter Plot Korelasi Positif

Penjelasan:

- Sumbu X (horizontal):
Berlabel Nilai1 dengan rentang dari 1 sampai 10.
- Sumbu Y (vertikal):
Berlabel Nilai2 dengan rentang dari 2 sampai 20.
- Titik-titik data (warna biru):
Masing-masing titik adalah pasangan (Nilai1, Nilai2), yaitu:
(1,2), (2,4), (3,6), (4,8), (5,10), (6,12), (7,14), (8,16), (9,18), (10,20).
Semua titik membentuk pola garis lurus naik dari kiri bawah ke kanan atas.
- Grid:
Garis bantu (grid) ditambahkan untuk memudahkan membaca posisi titik.
- Terlihat bahwa ketika Nilai1 naik, Nilai2 juga ikut naik secara teratur.
- Hubungan ini disebut korelasi positif sempurna (karena $\text{Nilai2} = 2 \times \text{Nilai1}$).
- Jadi semakin besar nilai pada variabel Nilai1, maka semakin besar juga nilai pada variabel Nilai2.

Scatter plot dibuat untuk memperlihatkan hubungan antar variabel. Pada contoh ini, Nilai1 berbanding lurus dengan Nilai2. Titik-titik data membentuk pola garis lurus dari kiri bawah ke kanan atas, yang menunjukkan korelasi positif sempurna. Artinya, semakin tinggi Nilai1, maka semakin tinggi pula Nilai2.

Kode:

```
[18]
✓ Os
import pandas as pd
import matplotlib.pyplot as plt

# Membuat Dataframe contoh
data = {
    'Nilai1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Nilai2': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
}

df3 = pd.DataFrame(data)

# Membuat scatter plot
plt.scatter(df3['Nilai1'], df3['Nilai2'], color='red', marker='x')

# Menambahkan Label
plt.title('Scatter Plot Korelasi Negatif')
plt.xlabel('Nilai1')
plt.ylabel('Nilai2')

# Menambahkan Grid
plt.grid(True)

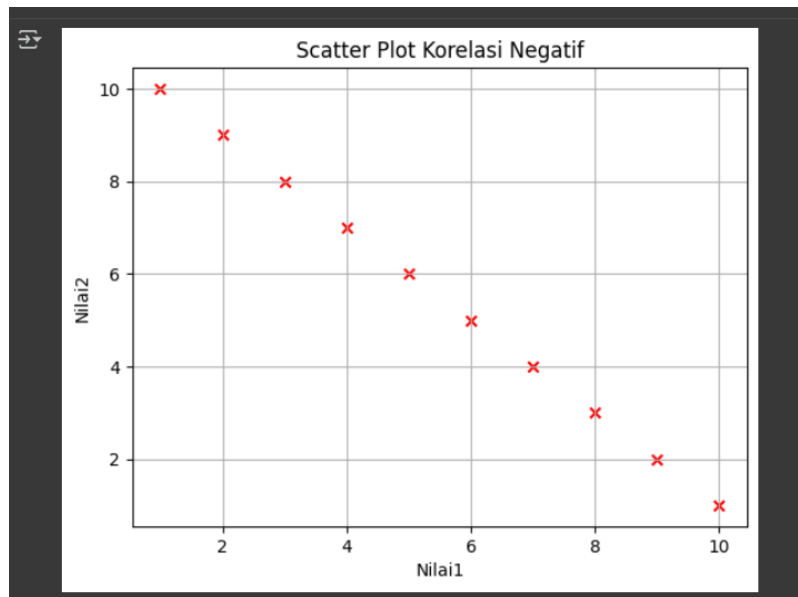
# Menampilkan plot
plt.show()
```

Gambar 15. Kode Scatter Plot Korelasi Negatif

Penjelasan:

- Import library
 - pandas untuk membuat DataFrame.
 - matplotlib.pyplot untuk visualisasi grafik scatter plot.
- Membuat dataset contoh
 - Nilai1 berisi angka 1–10 (naik).
 - Nilai2 berisi angka 10–1 (turun).
 - Data ini menunjukkan hubungan berlawanan: ketika Nilai1 naik, Nilai2 turun.
- Membuat scatter plot
 - plt.scatter(...) menggambar titik-titik pada grafik:
 - color='red' → titik berwarna merah.
 - marker='x' → bentuk marker silang.
- Memberikan judul dan label
 - plt.title() → judul grafik.
 - plt.xlabel() dan plt.ylabel() → label sumbu X dan Y.
- Menambahkan grid
 - plt.grid(True) → menambahkan garis bantu grid agar pola hubungan lebih mudah terlihat.
- Menampilkan grafik
 - plt.show() → menampilkan scatter plot ke layar.

Output:



Gambar 16. Output Scatter Plot Korelasi Negatif

Penjelasan:

- "Scatter Plot Korelasi Negatif" → menunjukkan bahwa grafik ini digunakan untuk melihat hubungan antar variabel yang sifatnya negatif.
- Sumbu X (horizontal):
Label Nilai1 dengan nilai dari 1 sampai 10.
- Sumbu Y (vertikal):
Label Nilai2 dengan nilai dari 10 sampai 1.
- Titik Data (warna merah, marker "x"):
- Titik-titik ini merepresentasikan pasangan nilai:
- (1,10), (2,9), (3,8), (4,7), (5,6), (6,5), (7,4), (8,3), (9,2), (10,1).
- Pola titik terlihat turun dari kiri atas ke kanan bawah.
- Saat Nilai1 bertambah (bergerak ke kanan), Nilai2 justru berkurang (bergerak ke bawah).
- Pola ini menunjukkan adanya korelasi negatif sempurna antara kedua variabel.
- Artinya hubungan kedua variabel bersifat berlawanan: jika Nilai1 naik, maka Nilai2 turun.

Scatter plot memperlihatkan hubungan negatif antara Nilai1 dan Nilai2. Titik-titik data membentuk pola menurun dari kiri atas ke kanan bawah. Hal ini menunjukkan bahwa semakin besar nilai Nilai1, maka nilai Nilai2 semakin kecil. Hubungan ini disebut korelasi negatif sempurna (nilai korelasi mendekati -1).

Praktikum Mandiri

Kode:

```
Praktikum Mandiri

[19] ✓ 1s
import pandas as pd
from sklearn.model_selection import train_test_split

# Membaca dataset day.csv
path = '/content/gdrive/MyDrive/Praktikum_ML/praktikum02/data/'
df = pd.read_csv(path + 'day.csv')

[20] ✓ 0s
# Membagi data menjadi training (80%) dan testing (20%)
train_df, test_df = train_test_split(df, test_size=0.2, random_state=42)

[21] ✓ 0s
# Membagi data training menjadi training (90%) dan validation (10%)
train_df, val_df = train_test_split(train_df, test_size=0.1, random_state=42)

[22] ✓ 0s
# Menampilkan jumlah data tiap set
print("Jumlah data Training : ", len(train_df))
print("Jumlah data Validation : ", len(val_df))
print("Jumlah data Testing : ", len(test_df))

Jumlah data Training : 525
Jumlah data Validation : 59
Jumlah data Testing : 147
```

Gambar 17. Kode Praktikum Mandiri

Penjelasan:

- Import pandas untuk manipulasi data (membaca CSV, DataFrame).
- `train_test_split` dari scikit-learn untuk membagi data menjadi subset.
- `path` adalah lokasi file (di contoh: path Google Colab / Google Drive).
- `pd.read_csv(...)` membaca file CSV menjadi DataFrame `df`.
- Membagi `df` menjadi dua bagian: `train_df` (80% data) dan `test_df` (20% data).
- `test_size=0.2` → 20% menjadi test.
- `random_state=42` → memastikan pembagian yang sama saat dijalankan ulang (reproducible).
- Default `shuffle=True` (data akan diacak sebelum pembagian).
- Membagi ulang `train_df` sebelumnya: 10% dari `train_df` dipakai sebagai `val_df`.
- `test_size=0.1` di sini artinya 10% dari data training sebelumnya, bukan 10% dari total data.
- Setelah dua langkah ini, proporsi akhir terhadap total N adalah:
- $\text{train_final} = 0.8 \times 0.9 = 0.72$ (72% dari total)
- $\text{val} = 0.8 \times 0.1 = 0.08$ (8% dari total)
- $\text{test} = 0.2$ (20% dari total)

Kode tersebut membaca file CSV (`day.csv`) ke DataFrame `df`, lalu membagi data menjadi train / validation / test dengan rasio 72% train, 8% validation, 20% test (hasil dari pemecahan 80/20 lalu 90/10 pada bagian training). Setelah dibagi, kode mencetak jumlah baris tiap set: Training = 525, Validation = 59, Testing = 147.

Kode:

```
[23] # Menampilkan 5 baris pertama tiap set
✓ Os print("\nData Training (5 baris teratas):")
      print(train_df.head())

      print("\nData Validation (5 baris teratas):")
      print(val_df.head())

      print("\nData Testing (5 baris teratas):")
      print(test_df.head())

      weathersit    temp    atemp    hum    windspeed    casual    registered \
657      2    0.563333    0.537896    0.815000    0.134954    753    4671
163      1    0.635000    0.601654    0.494583    0.305350    863    4157
305      1    0.377500    0.390133    0.718750    0.082092    370    3816
111      2    0.336667    0.321954    0.729583    0.219521    177    1506
538      1    0.777500    0.724121    0.573750    0.182842    964    4859

      cnt
657    5424
163    5020
305    4186
111    1683
538    5823
```

Gambar 18. Menampilkan 5 baris pertama tiap set

Penjelasan:

- o `print("Data Training (5 baris teratas):\n")`
- o Digunakan hanya untuk menampilkan teks keterangan di output agar lebih jelas data mana yang sedang ditampilkan.
- o Tanda `\n` artinya newline atau membuat baris baru setelah teks.
- o `x_train.head()`
- o `head()` adalah fungsi bawaan dari Pandas untuk menampilkan beberapa baris awal dari DataFrame.
- o Secara default, `head()` menampilkan 5 baris pertama.
- o Tujuannya untuk melihat isi data training di awal, sehingga kita bisa memastikan data berhasil dipanggil dan strukturnya sesuai.

Ouput:

Data Training (5 baris teratas):									
	instant	dteday	season	yr	mnth	holiday	weekday	workingday	\
657	658	2012-10-19	4	1	10	0	5	1	
163	164	2011-06-13	2	0	6	0	1	1	
305	306	2011-11-02	4	0	11	0	3	1	
111	112	2011-04-22	2	0	4	0	5	1	
538	539	2012-06-22	3	1	6	0	5	1	
	weathersit	temp	atemp	hum	windspeed	casual	registered	\	
657	2	0.563333	0.537896	0.815000	0.134954	753	4671		
163	1	0.635000	0.601654	0.494583	0.305350	863	4157		
305	1	0.377500	0.390133	0.718750	0.082092	370	3816		
111	2	0.336667	0.321954	0.729583	0.219521	177	1506		
538	1	0.777500	0.724121	0.573750	0.182842	964	4859		
	cnt								
657	5424								
163	5020								
305	4186								
111	1683								
538	5823								

Gambar 19. Output Data Training

Penjelasan:

- o Menampilkan 5 baris awal dari dataset training (`x_train`).
- o Kolom yang muncul antara lain: `instant`, `dteday`, `season`, `yr`, `mnth`, `holiday`, `weekday`, `workingday`, `weathersit`, `temp`, `atemp`, `hum`, `windspeed`, `casual`, `registered`, `cnt`.

- Data ini digunakan untuk melatih model machine learning karena memuat sebagian besar informasi dari dataset.

Interpretasi:

- Data training berfungsi sebagai data utama untuk membuat model belajar pola.
- Contoh: baris pertama menunjukkan data pada tanggal 2011-12-04 dengan kondisi cuaca (weathersit), suhu (temp), kelembaban (hum), dan jumlah peminjaman sepeda (cnt = 6606).

Data Testing (5 baris teratas):									
	instant	dteday	season	yr	mnth	holiday	weekday	workingday	\
703	704	2012-12-04	4	1	12	0	2	1	
33	34	2011-02-03	1	0	2	0	4	1	
300	301	2011-10-28	4	0	10	0	5	1	
456	457	2012-04-01	2	1	4	0	0	0	
633	634	2012-09-25	4	1	9	0	2	1	
	weathersit	temp	atemp	hum	windspeed	casual	registered	\	
703	1	0.475833	0.469054	0.733750	0.174129	551	6055		
33	1	0.186957	0.177878	0.437826	0.277752	61	1489		
300	2	0.330833	0.318812	0.585833	0.229479	456	3291		
456	2	0.425833	0.417287	0.676250	0.172267	2347	3694		
633	1	0.550000	0.544179	0.570000	0.236321	845	6693		
	cnt								
703	6606								
33	1550								
300	3747								
456	6041								
633	7538								

Gambar 20. Output Data Testing

Penjelasan:

- Menampilkan 5 baris awal dari dataset testing (x_test).
- Struktur kolom sama seperti data training.
- Contoh: pada tanggal 2012-12-04 (instant 704), jumlah peminjaman sepeda (cnt) adalah 6606.

Interpretasi:

- Data testing digunakan untuk mengukur performa model setelah dilatih.
- Data ini tidak dilihat model saat training, sehingga hasil evaluasi akan lebih objektif.
- Dengan data testing, kita bisa tahu apakah model bekerja dengan baik pada data baru.

Data Validation (5 baris teratas):									
	instant	dteday	season	yr	mnth	holiday	weekday	workingday	\
325	326	2011-11-22	4	0	11	0	2	1	
410	411	2012-02-15	1	1	2	0	3	1	
92	93	2011-04-03	2	0	4	0	0	0	
47	48	2011-02-17	1	0	2	0	4	1	
508	509	2012-05-23	2	1	5	0	3	1	
	weathersit	temp	atemp	hum	windspeed	casual	registered	\	
325	3	0.416667	0.421696	0.962500	0.118792	69	1538		
410	1	0.348333	0.351629	0.531250	0.181600	141	4028		
92	1	0.378333	0.378767	0.480000	0.182213	1651	1598		
47	1	0.435833	0.428658	0.505000	0.230104	259	2216		
508	2	0.621667	0.584612	0.774583	0.102000	766	4494		
	cnt								
325	1607								
410	4169								
92	3249								
47	2475								
508	5260								

Gambar 21. Output Data Validation

Penjelasan:

- Menampilkan 5 baris awal dari dataset validation (x_val).
- Contoh: pada tanggal 2011-11-22 (instant 326), cnt atau jumlah peminjaman sepeda = 1607.

Interpretasi:

- Data validation dipakai untuk menyetel parameter (hyperparameter tuning) dan mengevaluasi model saat proses pelatihan.
- Dengan data ini, kita bisa tahu apakah model overfitting atau tidak sebelum dites dengan data testing.

Link Github:

<https://github.com/rikaarahma/MachineLearning/blob/main/praktikum02/notebook/praktikum02.ipynb>