

Praktikum 2: Laporan Praktikum di Kelas dan Praktikum Mandiri

Sintia Sari - 0110222135 ^{1*}

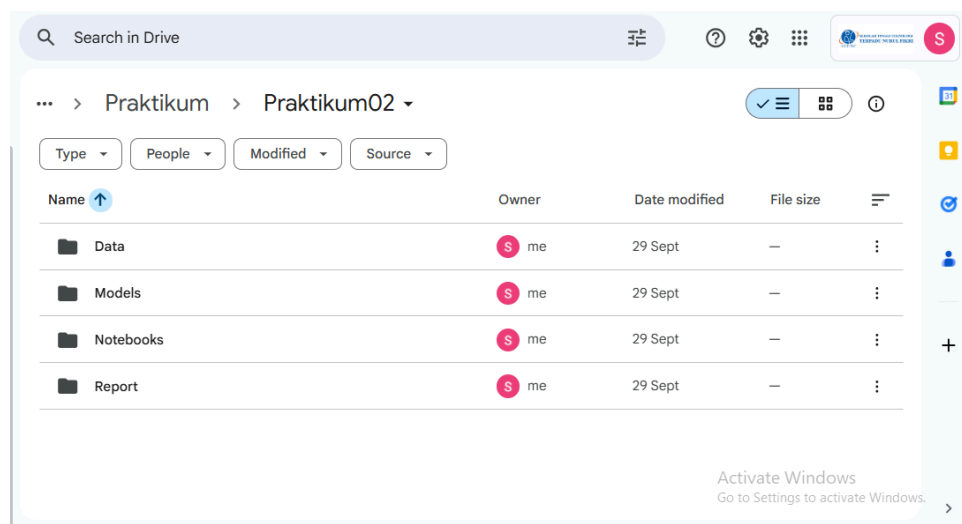
¹ Teknik Informatika, STT Terpadu Nurul Fikri, Depok

*E-mail: sint22135ti@student.nurulfikri.ac.id

Abstract. Praktikum ini berfokus pada analisis dan persiapan data menggunakan Python. Dilakukan eksplorasi dataset '500_Person_Gender_Height_Weight_Index.csv' analisis ini meliputi perhitungan statistik deskriptif (mean, median, dll.) dan korelasi pada data tinggi dan berat badan, yang menunjukkan hubungan kuat antara berat dan indeks tubuh. Data juga divisualisasikan menggunakan Boxplot, Histogram, dan Scatter Plot. Langkah kunci dalam praktikum mandiri adalah membagi dataset 'day.csv' menjadi tiga bagian penting untuk Machine Learning yaitu Training (80%), Testing (20%), dan Validation (10%). Kesimpulannya, praktikum ini berhasil menerapkan teknik dasar untuk memahami dan menyusun data sebelum pembangunan model.

Praktikum Pertemuan 2

1. Direktori Program



Gambar 1. Direktori Program

Dalam pengerjaan tugas maupun praktikum, semua coding dikerjakan dalam folder notebooks dengan menggunakan file Python bernama praktikum02.ipynb. Dataset mentah ditempatkan pada folder data, sedangkan model hasil training disimpan pada

folder models dalam format .pkl atau .joblib. Adapun laporan maupun hasil visualisasi disimpan pada folder reports.

2. Menghubungkan dengan Google Drive

```
# Menghubungkan colab dengan google drive
from google.colab import drive
drive.mount('/content/gdrive')
import os

Mounted at /content/gdrive

# Memanggil dataset melalui gdrive
path = "gdrive/MyDrive/Machine Learning/Praktikum/Praktikum02/Data/"
```

Gambar 2. Menghubungkan colab dengan Gdrive

Penjelasan Kode :

- `from google.colab import drive & drive.mount('/content/gdrive')`
Menghubungkan Google Colab dengan Google Drive supaya file dataset bisa diakses langsung dari Drive.
- `import os`
Digunakan untuk mengatur direktori/file di sistem (opsional, untuk navigasi folder).
- `path = "gdrive/MyDrive/Machine Learning/Praktikum/Praktikum02/Data/"`
Menyimpan alamat folder tempat file dataset berada.

Setelah dijalankan, Colab akan menampilkan pesan "Mounted at /content/gdrive", artinya Google Drive berhasil terhubung ke Colab dan bisa digunakan untuk membaca atau menyimpan file. Selain itu, variabel path menyimpan lokasi dataset sehingga bisa digunakan nanti.

3. Membaca File CSV

```
# Membaca file csv menggunakan pandas
import pandas as pd

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

Gambar 3. Membaca dan Menampilkan Dataset

Penjelasan Kode :

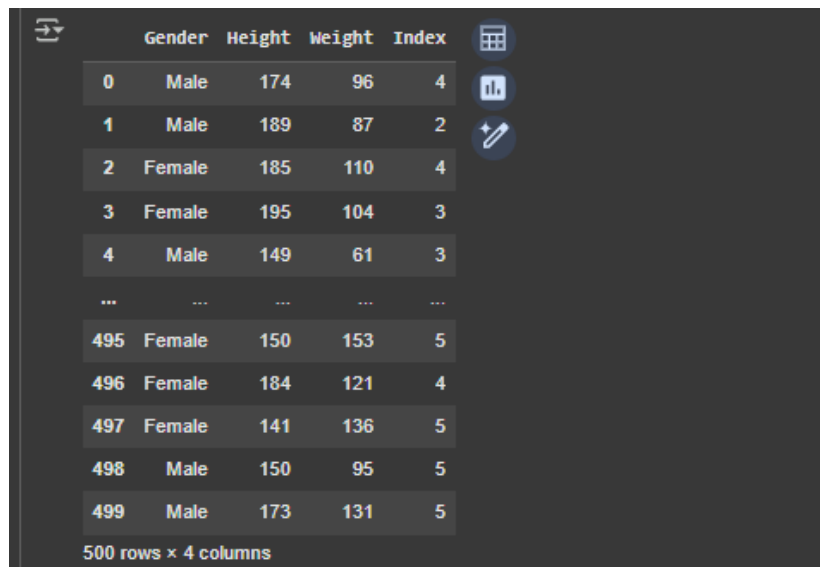
- `import pandas as pd`
Mengimpor library pandas dengan alias pd, digunakan untuk membaca dan mengolah data dalam bentuk tabel (DataFrame).

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

Membaca file CSV bernama `500_Person_Gender_Height_Weight_Index.csv` yang ada di folder `path`, lalu menyimpannya ke dalam DataFrame dengan nama variabel `df`.

- `df`

Menampilkan isi DataFrame `df`. Pandas hanya menampilkan sebagian baris awal dan baris akhir jika data panjang.



	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

Gambar 4. Isi dari Dataset `500_Person_Gender_Height_Weight_Index.csv`

Dataset berhasil dimuat dengan ukuran 500 baris \times 4 kolom, yaitu:

Tabel 1. Nama dan Deskripsi Dataset

Kolom	Deskripsi
Gender	Jenis kelamin (Male/Female)
Height	Tinggi badan (cm)
Weight	Berat badan (kg)
Index	Kategori indeks berat badan

4. Analisis Statistik Deskriptif

1) Melihat Informasi Umum Data

```
# Mencari info data pada file (tipe datanya, non 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 5. Melihat Informasi Umum Data

Penjelasan kode :

- `df.info()`

Menampilkan informasi ringkas tentang DataFrame, termasuk: jumlah baris, jumlah kolom, nama kolom, jumlah data non-null, tipe data tiap kolom, serta estimasi penggunaan memori.

Dataset memiliki total data sebanyak 500 baris dan kolom. Tipe data dari kolom Gender bertipe string/object dan kolom Height, Weight, Index bertipe integer.

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

```
# Menghitung mean semua kolom numerik
df['Height'].mean()

np.float64(169.944)
```

Gambar 6. Menghitung Mean

`df['Height'].mean()`, berfungsi untuk menghitung nilai rata-rata (mean) dari kolom Height. Hasil perhitungan mean adalah 169.944, artinya rata-rata tinggi badan dalam dataset sekitar 169,94 cm.

```
# Menghitung median semua kolom numerik
df['Height'].median()

170.5
```

Gambar 7. Menghitung Median

`df['Height'].median()`, berfungsi untuk menghitung nilai tengah (median) dari kolom Height. Hasil perhitungan median adalah 170.5, artinya nilai tengah tinggi badan adalah 170,5 cm.

```
# Menghitung modus semua kolom numerik
df['Height'].mode()

Height
0      188
dtype: int64
```

Gambar 8. Menghitung Modus

`df['Height'].mode()`, berfungsi untuk menghitung nilai yang paling sering muncul (modus) pada kolom Height. Hasil perhitungan modus adalah 188, artinya tinggi badan yang paling sering muncul dalam dataset adalah 188 cm.

3) Menghitung Kuartil

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

# Menghitung kuartil kedua (Q2)
q3 = df['Height'].quantile(0.75)
print("Q3:", q3)

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

Gambar 9. Menghitung Kuartil

Penjelasan kode :

- `q1 = df['Height'].quantile(0.25)`
Menghitung kuartil pertama (Q1) dari kolom Height pada DataFrame.
- `print("Q1:", q1)`
Untuk menampilkan hasilnya.
- `q3 = df['Height'].quantile(0.75)`
Menghitung kuartil ketiga (Q3) dari kolom Height.
- `print("Q3:", q3)`
Untuk menampilkan hasilnya.
- `iqr = q3 - q1`
Menghitung IQR (Interquartile Range) yaitu rentang antar kuartil.
- `print("IQR:", iqr)`
Untuk menampilkan hasilnya.

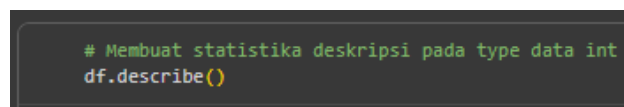


```
Q1: 156.0
Q3: 184.0
IQR: 28.0
```

Gambar 10. Hasil Perhitungan Kuartil

Nilai Q1 sebesar 156.0 menunjukkan 25% data berada di bawah angka tersebut, sedangkan Q3 sebesar 184.0 menunjukkan 75% data berada di bawah angka itu. Dengan demikian, 50% data berada dalam rentang 156.0 hingga 184.0. Selisihnya, yaitu IQR sebesar 28.0, menggambarkan sebaran data pada bagian tengah distribusi.

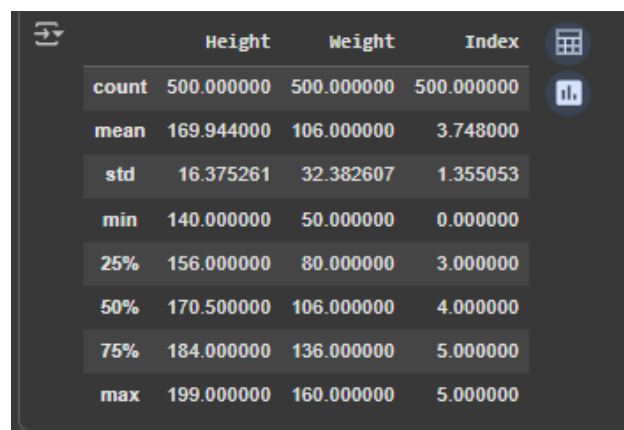
4) Menghitung Statistik Deskriptif Otomatis



```
# Membuat statistika deskripsi pada type data int
df.describe()
```

Gambar 11. Menghitung Statistik Deskriptif

`df.describe()` digunakan untuk menampilkan statistika deskriptif dari data numerik pada DataFrame `df`.



	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 12. Hasil Statistik Deskriptif

Data memiliki 500 sampel dengan tinggi badan rata-rata sekitar 169,9 cm (rentang 140–199 cm) dan berat badan rata-rata 106 kg (rentang 50–160 kg). Nilai median untuk tinggi adalah 170,5 cm dan untuk berat 106 kg, yang menunjukkan distribusi data relatif seimbang. Sebaran data terlihat cukup lebar terutama pada berat badan dengan simpangan baku lebih besar, sehingga variasi antarindividu lebih tinggi dibanding tinggi badan.

5) Menghitung Korelasi

```
# Menghitung matriks korelasi untuk semua kolom numerik
correlation_matrix = df.corr(numeric_only=True)

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

Gambar 13. Menghitung Korelasi

Penjelasan kode :

- `correlation_matrix = df.corr(numeric_only=True)`
Menghitung matriks korelasi antar kolom numerik dalam DataFrame `df`. Parameter `numeric_only=True` memastikan hanya kolom numerik yang dihitung. Korelasi menunjukkan hubungan antar variabel dengan nilai dari -1 sampai 1.
- `print("Matriks Korelasi")`
Menampilkan judul teks "Matriks Korelasi".
- `print(correlation_matrix)`
Menampilkan tabel matriks korelasi hasil perhitungan.

```
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 14. Hasil Matriks Korelasi

Hasil matriks korelasi menunjukkan bahwa tinggi badan tidak berkorelasi dengan berat badan. Indeks memiliki korelasi negatif sedang dengan tinggi badan dan korelasi positif kuat dengan berat badan, sehingga semakin tinggi seseorang indeksinya cenderung lebih kecil, sedangkan semakin berat seseorang indeksinya cenderung lebih besar.

5. Visualisasi Data

1) Boxplot

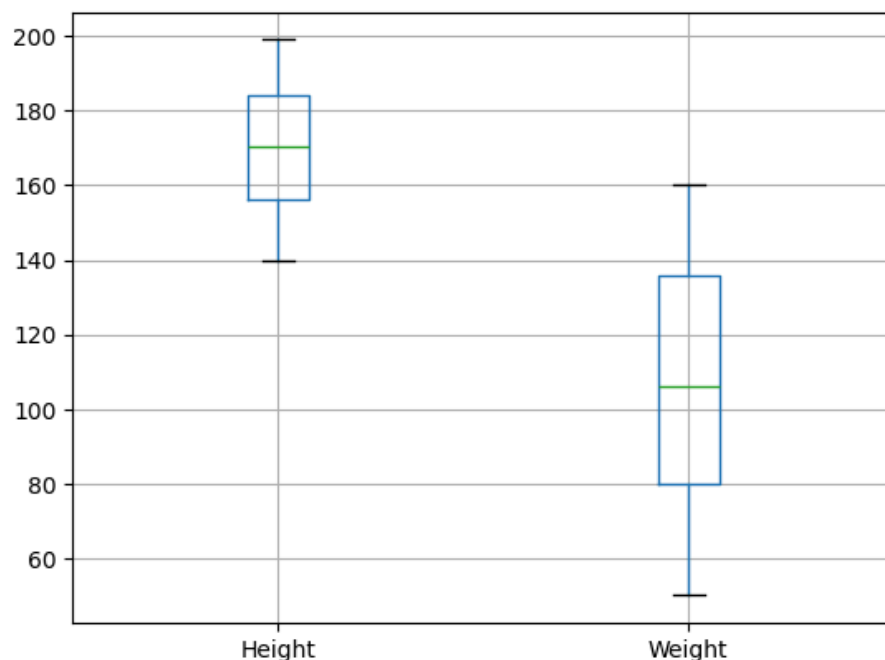
```
import numpy as np

df.boxplot(column=['Height', 'Weight'])
```

Gambar 15. Boxplot

Penjelasan kode :

- `import numpy as np`
Mengimpor library NumPy dengan alias `np`. Digunakan untuk perhitungan numerik, meskipun pada kode ini belum dipakai langsung.
- `df.boxplot(column=['Height', 'Weight'])`
Membuat boxplot (diagram kotak) untuk kolom `Height` dan `Weight` dari DataFrame `df`. Boxplot digunakan untuk melihat distribusi data, median, kuartil, serta outlier.



Gambar 16. Visualisasi Boxplot

Berdasarkan hasil visualisasi diatas, didapatkan informasi sebagai berikut.

a) Boxplot Height

- Median (garis hijau dalam kotak) berada sekitar 170.
- Sebagian besar data berada pada rentang 156 (Q1) hingga 184 (Q3).
- Rentang nilai penuh sekitar 140–199 tanpa terlihat outlier ekstrem.

b) Boxplot Weight

- Median berada sekitar 106.
- Sebagian besar data berada pada rentang 80 (Q1) hingga 136 (Q3).

- Rentang nilai penuh sekitar 50–160, dengan kemungkinan ada outlier di bawah 60.

2) Histogram

```
import matplotlib.pyplot as plt

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

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

# Menambahkan label
plt.title('Histogram Nilai')
plt.xlabel('Height')
plt.ylabel('Frequency')

# 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)])

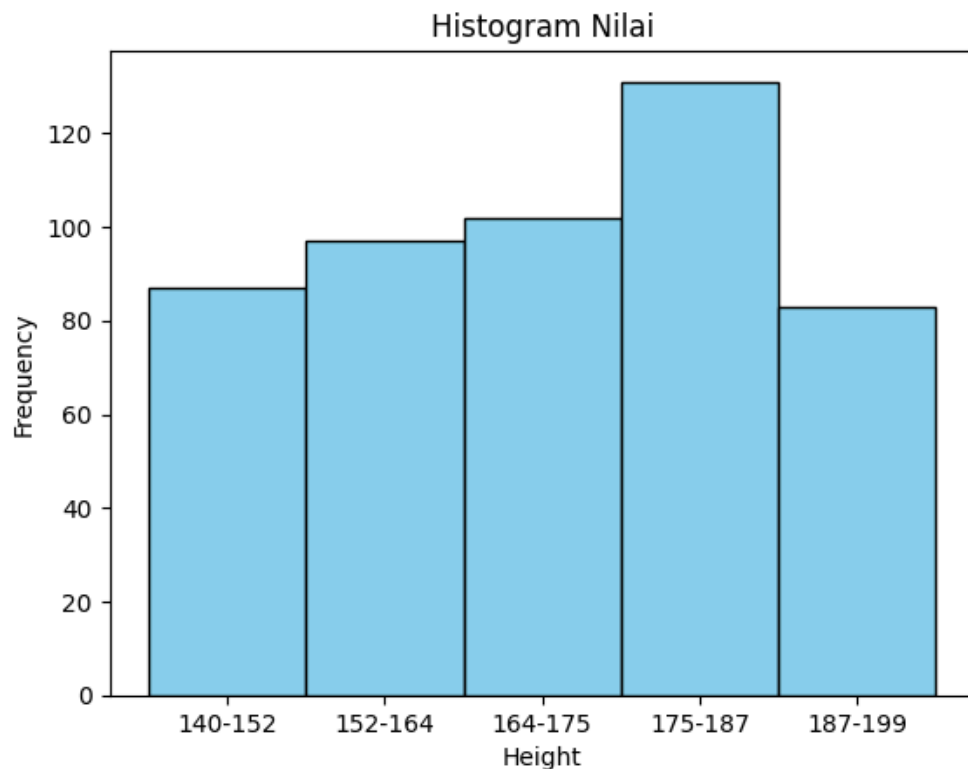
# Menampilkan Grafik
plt.show()
```

Gambar 17. Histogram

Penjelasan kode :

- `import matplotlib.pyplot as plt`
Mengimpor library Matplotlib untuk visualisasi data dengan alias plt.
- `data_height = df['Height']`
Mengambil kolom Height dari DataFrame dan menyimpannya ke variabel data_height.
- `n, bins, patches = plt.hist(data_height, bins=5, color='skyblue', edgecolor='black')`
 - `bins=5`
Membagi data ke dalam 5 interval (kelas).
 - `color='skyblue'`
Memberi warna biru muda pada batang histogram.
 - `edgecolor='black'`
Memberi garis tepi hitam pada batang.
- `n, bins, patches`
Menyimpan nilai frekuensi, batas interval, dan objek grafik.
- `plt.title('Histogram Nilai')`
Memberi judul grafik.

- `plt.xlabel('Height')` dan `plt.ylabel('Frequency')`
Memberi label pada sumbu X (Height) dan sumbu Y (Frequency).
- `bin_centers = 0.5 * (bins[1:] + bins[:-1])`
Menghitung titik tengah tiap interval bin.
- `plt.xticks(bin_centers, ['{:0f}-{:0f}'.format(bins[i], bins[i+1]) for i in range(len(bins)-1)])`
Menampilkan label interval bin di sumbu X dengan format rentang (contoh: 140-152, 152-164, dll).
- `plt.show()`
Menampilkan grafik histogram.



Gambar 18. Visualisasi Histogram

Histogram menunjukkan distribusi tinggi badan (Height) sebanyak 500 data yang dibagi ke dalam 5 interval.

- Interval 140–152 memiliki sekitar 90 data.
- Interval 152–164 memiliki sekitar 100 data.
- Interval 164–175 memiliki sekitar 110 data.
- Interval 175–187 memiliki frekuensi paling tinggi, sekitar 130 data.

- Interval 187–199 memiliki sekitar 80 data.
- 3) Scatter Plot (Hubungan Antar Variabel)
- a) Korelasi positif

```
# 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)
```

Gambar 19. Membuat Dataframe

- Membuat dataset dengan dua kolom (Nilai1 dan Nilai2).
- Nilai2 adalah kelipatan 2 dari Nilai1, sehingga memiliki korelasi positif sempurna.

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

Gambar 20. Membuat Scatter Plot

- Membuat grafik scatter plot antara Nilai1 (sumbu X) dan Nilai2 (sumbu Y).
- Titik ditampilkan dengan warna biru dan bentuk lingkaran (o).

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

Gambar 21. Menambahkan Label

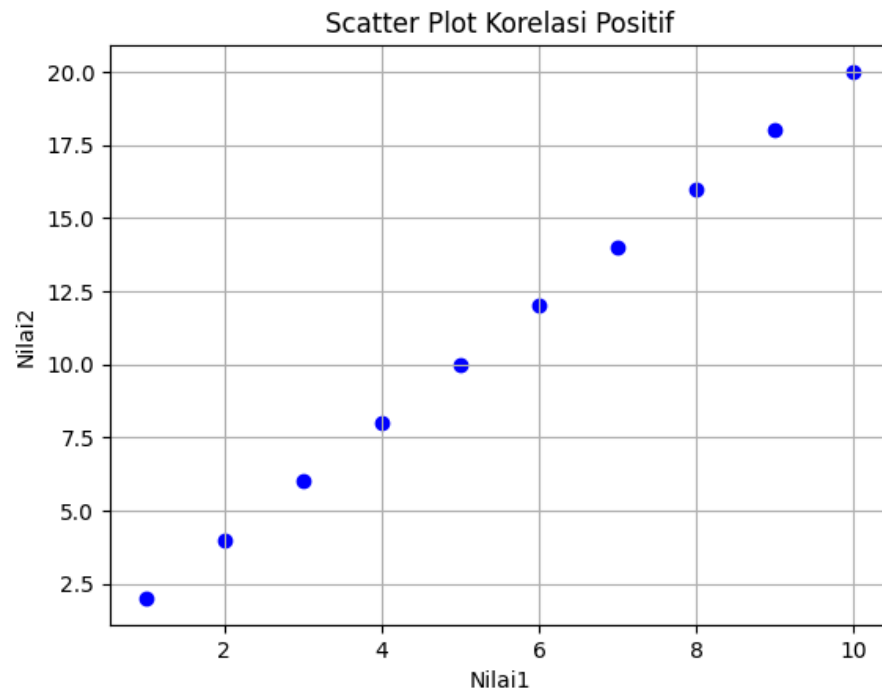
Memberikan judul grafik dan label pada masing-masing sumbu.

```
# Menambahkan grid
plt.grid(True)

# Menampilkan plot
plt.show()
```

Gambar 22. Menampilkan Grid dan Plot

Menambahkan garis grid dan menampilkan grafik scatter plot.



Gambar 23. Hasil Korelasi Positif

Scatter plot menunjukkan hubungan antara Nilai1 dan Nilai2. Terlihat bahwa semua titik berada pada garis lurus dengan arah positif (semakin besar Nilai1, semakin besar juga Nilai2). Hal ini menunjukkan korelasi positif sempurna (nilai korelasi = 1) antara kedua variabel.

b) Korelasi negatif

```
# 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')

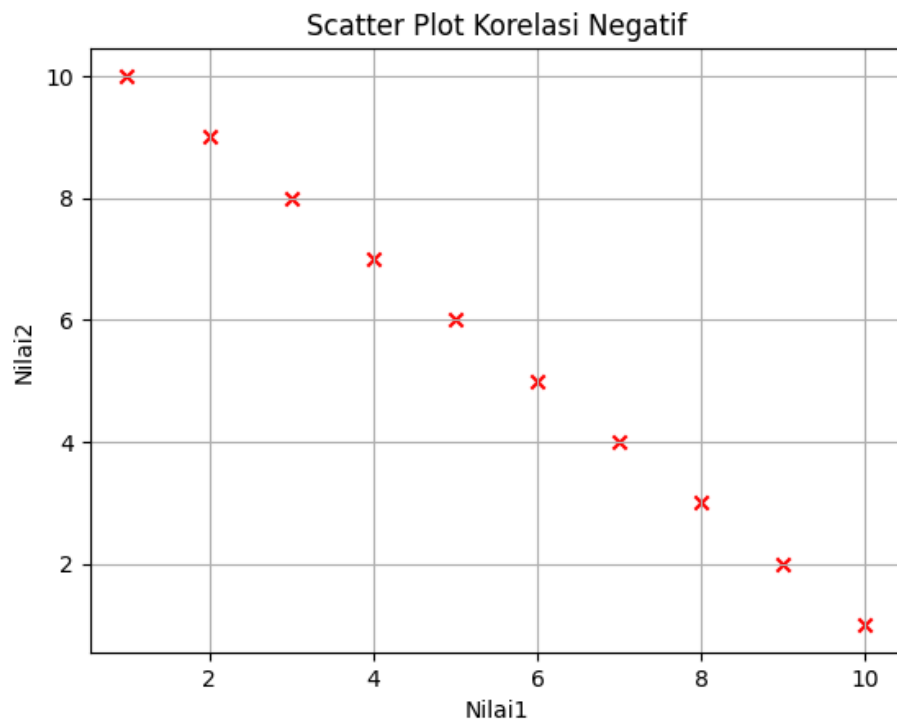
# Menambahkan label
plt.title('Scatter Plot Korelasi Negatif')
plt.xlabel('Nilai1')
plt.ylabel('Nilai2')

# Menambahkan grid
plt.grid(True)

# Menampilkan plot
plt.show()
```

Gambar 24. Korelasi Negatif

Secara keseluruhan, kodingan scatter plot yang menunjukkan korelasi negatif ini hampir sama dengan scatter plot yang menunjukkan korelasi positif, hanya berbeda pada isi data dan beberapa detail visual. Perbedaan utama terletak pada isi data Nilai2, yang menentukan arah hubungan (naik untuk positif, turun untuk negatif). Perbedaan kecil lain ada pada warna, marker, dan judul plot, sedangkan strukturnya tetap sama.



Gambar 25. Hasil Korelasi Negatif

Scatter plot pada gambar menunjukkan hubungan negatif sempurna antara Nilai1 dan Nilai2. Saat Nilai1 meningkat dari 1 ke 10, Nilai2 justru menurun dari 10 ke 1. Semua titik mengikuti pola garis lurus menurun, sehingga korelasinya mendekati -1.

(Praktikum mandiri di page selanjutnya)

Praktikum Mandiri

1. Memuat, membaca dan menampilkan data dalam dataset

```
# Menampilkan data dalam dataset day.csv
df_day = pd.read_csv(path + 'day.csv')
df_day
```

Gambar 26. Menampilkan Data dalam Dataset

Penjelasan kode :

- `pd.read_csv(path + 'day.csv')`

Digunakan untuk membaca file CSV bernama day.csv. Hasilnya akan dimuat ke dalam sebuah DataFrame Pandas bernama df_day.

- `df_day`

Menampilkan isi DataFrame df_day.

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.590435	0.160296	108	1454	1562
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.229270	0.436957	0.186900	82	1518	1600
...
726	727	2012-12-27	1	1	12	0	4	1	2	0.254167	0.226642	0.652917	0.350133	247	1867	2114
727	728	2012-12-28	1	1	12	0	5	1	2	0.253333	0.255046	0.590000	0.155471	644	2451	3095
728	729	2012-12-29	1	1	12	0	6	0	2	0.253333	0.242400	0.752917	0.124383	159	1182	1341
729	730	2012-12-30	1	1	12	0	0	0	1	0.255833	0.231700	0.483333	0.350754	364	1432	1796
730	731	2012-12-31	1	1	12	0	1	1	2	0.215833	0.223487	0.577500	0.154846	439	2290	2729

731 rows x 16 columns

Gambar 27. Isi dari Dataset

Hasil yang muncul adalah tabel dataset day.csv dengan 731 baris dan 16 kolom. Kolom-kolom tersebut berisi informasi mengenai data peminjaman sepeda harian.

2. Membagi dataset day.csv menjadi tiga bagian, yaitu Data Training, Validation, Testing dan menampilkan jumlah datanya

```

from sklearn.model_selection import train_test_split

# Membagi dataset day.csv menjadi data training (80%) dan data testing (20%)
train_df, test_df = train_test_split(df_day, test_size=0.2, random_state=42)

# Membagi data training menjadi data training (90% dari training awal, yaitu 72% dari total)
# dan data validation (10% dari training awal, yaitu 8% dari total)
train_df, val_df = train_test_split(train_df, test_size=0.1, random_state=42)

# Menampilkan jumlah data untuk setiap set
print("Jumlah data Training:", len(train_df))
print("Jumlah data Validation:", len(val_df))
print("Jumlah data Testing:", len(test_df))

```

Gambar 28. Membagi Dataset day.csv

Penjelasan kode :

- `from sklearn.model_selection import train_test_split`
Mengimpor fungsi `train_test_split` yang digunakan untuk membagi data secara acak.
- `train_df, test_df = train_test_split(df_day, test_size=0.2, random_state=42)`
Membagi dataset awal (`df_day`) menjadi dua set: data training (`train_df`) sebesar 80% dan data testing (`test_df`) sebesar 20% (karena `test_size=0.2`).
- `train_df, val_df = train_test_split(train_df, test_size=0.1, random_state=42)`
Membagi data training awal (`train_df`) menjadi dua set baru: data training (baru) dan data validation (`val_df`). Pembagian ini mengambil 10% dari set training yang sekarang sebagai data validation (yaitu $0.1 \times 80\% = 8\%$ dari total data awal). Sisanya 90% dari set training yang sekarang menjadi data training (yaitu $0.9 \times 80\% = 72\%$ dari total data awal).
- `print("Jumlah data Training:", len(train_df))`
Menampilkan jumlah baris (ukuran) dari set data training.
- `print("Jumlah data Validation:", len(val_df))`
Menampilkan jumlah baris (ukuran) dari set data validation.
- `print("Jumlah data Testing:", len(test_df))`
Menampilkan jumlah baris (ukuran) dari set data testing.

```

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

```

Gambar 29. Hasil dari Pembagian Dataset

Dari proses pembagian data, didapatkan total 731 baris data. Data ini kemudian dibagi menjadi tiga set :

- Data Training berjumlah 525 baris. Ini adalah 72% dari total data dan akan digunakan untuk melatih model.
- Data Testing berjumlah 147 baris. Ini adalah 20% dari total data dan akan digunakan untuk menguji performa akhir model setelah selesai dilatih.
- Data Validation berjumlah 59 baris. Ini adalah 8% dari total data dan akan digunakan untuk penyesuaian atau penyetelan hyperparameter model selama proses pengembangan.

3. Menampilkan 5 baris data teratas untuk Data Training

```
# Menampilkan 5 baris data teratas untuk Data Training
print("\nData Training (5 baris teratas):")
display(train_df.head())
```

Gambar 30. Menampilkan Data Training

Kode ini bertujuan untuk menampilkan 5 baris data teratas dari DataFrame yang telah ditetapkan sebagai Data Training (train_df) untuk memberikan gambaran cepat tentang isi dan format datanya. Fungsi display() untuk menampilkan hasilnya dalam format tabel yang mudah dibaca.

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

Gambar 31. Isi dari Data Training

Hasilnya adalah tampilan tabel yang menunjukkan 5 baris data dari set Data Training (train_df), yang merupakan data acak dari dataset asli (ditunjukkan dari indeks baris yang tidak berurutan, seperti 657, 163, dll.).

4. Menampilkan 5 baris data teratas untuk Data Validation

```
# Menampilkan 5 baris data teratas untuk Data Validation
print("\nData Validation (5 baris teratas):")
display(val_df.head())
```

Gambar 32. Menampilkan Data Validation

Kode ini berfungsi untuk menampilkan 5 baris data teratas dari DataFrame yang telah dialokasikan sebagai Data Validation (val_df). Fungsi display() untuk menampilkan hasilnya dalam format tabel yang mudah dibaca.

Data Validation (5 baris teratas):

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

Gambar 33. Isi dari Data Validation

Hasilnya adalah tampilan tabel yang menunjukkan 5 baris data (secara acak, ditunjukkan dari indeks baris yang tidak berurutan, seperti 325, 410, 92, dll.) dari set Data Validation (val_df).

5. Menampilkan 5 baris data teratas untuk Data Testing

```
# Menampilkan 5 baris data teratas untuk Data Testing
print("\nData Testing (5 baris teratas):")
display(test_df.head())
```

Gambar 34. Menampilkan Data Testing

Kode ini bertujuan untuk menampilkan 5 baris data teratas dari DataFrame yang telah dialokasikan sebagai Data Testing (test_df). Fungsi display() untuk menampilkan hasilnya dalam format tabel yang mudah dibaca.

Data Testing (5 baris teratas):

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

Gambar 35. Isi dari Data Testing

Tampilan hasil menunjukkan 5 baris data teratas dari set Data Testing (test_df), yang terdiri dari total 147 baris (20% dari data awal). Set data ini memiliki struktur kolom yang sama dengan set lainnya. Data Testing ini akan digunakan hanya sekali di tahap akhir untuk memberikan evaluasi performa model yang tidak bias terhadap prediksi variabel target (cnt).

References

Link Github :

https://github.com/ssintyaaa/MachineLearning/blob/main/praktikum%2002/notebooks/Praktikum2_Sintia_Sari_0110222135_ML_Pagi.ipynb

Link Gdrive :

<https://drive.google.com/drive/folders/1L9jFGp4q0XIyN7CxoUKYpPpblW0RnJlR?usp=sharing>