

# Tugas 1: Praktikum 2 dan Praktikum Mandiri 2

Fatih Dzakwan Susilo - 0110224235


<sup>1</sup> Teknik Informatika, STT Terpadu Nurul Fikri, Depok

\*E-mail: [fatihdzakwansusilo@gmail.com](mailto:fatihdzakwansusilo@gmail.com)

## 1. Praktikum 2

Analisis data merupakan langkah penting dalam memahami karakteristik, pola, dan hubungan antar variabel dalam sebuah dataset. Melalui proses seperti pembagian data, perhitungan statistik deskriptif, visualisasi data, hingga analisis korelasi, kita dapat memperoleh wawasan yang lebih dalam tentang data yang dimiliki. Proses ini juga menjadi dasar penting dalam machine learning dan pengambilan keputusan berbasis data.

### 1.1 Menghubungkan Collab dengan Google Drive



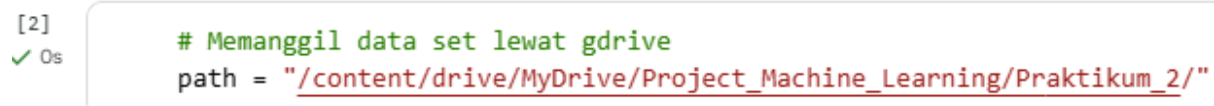
```
[1]
✓ 12s
# Menghubungkan colab dengan google drive
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

**Gambar 1.** Menghubungkan Collab dengan Google Drive.

Kode tersebut digunakan untuk menghubungkan Google Colab dengan Google Drive sehingga file atau dataset yang tersimpan di Drive dapat diakses dan digunakan langsung dalam lingkungan kerja Colab.

### 1.2 Memanggil data set lewat gdrive

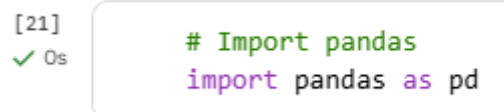


```
[2]
✓ 0s
# Memanggil data set lewat gdrive
path = "/content/drive/MyDrive/Project_Machine_Learning/Praktikum_2/"
```

**Gambar 2.** Memanggil data set lewat gdrive.

Kode tersebut berfungsi untuk menentukan lokasi folder dataset di Google Drive sehingga program dapat mengakses dan memanggil file data yang tersimpan di jalur tersebut.

### 1.3 Import pandas



```
[21]
✓ 0s
# Import pandas
import pandas as pd
```

**Gambar 3.** Import pandas.

Kode tersebut digunakan untuk mengimpor library pandas dengan alias pd, yang berfungsi untuk mengolah, menganalisis, dan memanipulasi data dalam bentuk tabel (DataFrame) di Python.

### 1.4 Membaca file csv menggunakan pandas

[22]  
✓ 0s

```
# Membaca file csv menggunakan pandas
df = pd.read_csv(path + 'Data/500_Person_Gender_Height_Weight_Index.csv')
df
```



	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.** Membaca file csv menggunakan pandas.

Kode tersebut berfungsi untuk membaca file dataset berformat CSV dari lokasi yang ditentukan dan menyimpannya ke dalam variabel df sebagai DataFrame pandas, sehingga data dapat diolah dan dianalisis lebih lanjut.

#### 1.5 Mencari info data pada file (tipe datanya, non nul count data, nama kolom)

[23]  
✓ 0s

```
# 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.** Mencari info data pada file (tipe datanya, non nul count data, nama kolom).

Dataset memiliki jumlah baris dan kolom tertentu (sesuai hasil `df.info()`), setiap kolom sudah memiliki tipe data yang sesuai (misalnya `int64` atau `object`), dan tidak ada nilai yang kosong (`non-null`) di seluruh kolom. Artinya, data siap digunakan untuk proses analisis selanjutnya tanpa perlu pembersihan awal.

#### 1.6 Menghitung mean semua kolom numerik

```
[24]
✓ Os # Menghitung mean semua kolom numerik
      df['Height'].mean()

      np.float64(169.944)
```

**Gambar 6.** Menghitung mean semua kolom numerik.

Rata-rata tinggi badan dari seluruh data dalam dataset tersebut adalah sebesar nilai mean yang dihasilkan. Nilai ini menunjukkan tinggi badan rata-rata populasi dalam data.

#### 1.7 Menghitung median semua kolom numerik

```
[25]
✓ Os # Menghitung median semua kolom numerik
      df['Height'].median()

      170.5
```

**Gambar 7.** Menghitung median semua kolom numerik.

Median tinggi badan dari seluruh data menunjukkan nilai tengah dari data tinggi badan setelah diurutkan, sehingga mencerminkan tinggi badan yang paling representatif tanpa terpengaruh oleh nilai ekstrem.

#### 1.8 Menghitung modus (hati-hati karena bisa lebih dari satu)

```
[24]
✓ Os # Menghitung mean semua kolom numerik
      df['Height'].mean()

      np.float64(169.944)
```

**Gambar 8.** Menghitung modus (hati-hati karena bisa lebih dari satu).

Modus tinggi badan menunjukkan nilai yang paling sering muncul dalam data tinggi badan. Jika hasilnya lebih dari satu, berarti terdapat lebih dari satu tinggi badan yang memiliki frekuensi kemunculan tertinggi.

#### 1.9 Menghitung variansi

```
[27] # Menghitung variansi
      df.var(numeric_only=True)
```

0

Height	268.149162
Weight	1048.633267
Index	1.836168

dtype: float64

**Gambar 9.** Menghitung variansi.

Variansi menunjukkan tingkat penyebaran atau keragaman data pada setiap kolom numerik. Semakin besar nilai variansinya, semakin bervariasi atau tersebar data tersebut dari nilai rata-ratanya.

#### 1.10 Menghitung standar deviasi

```
[28] # Menghitung standar deviasi
      df.std(numeric_only=True)
```

0s

0

Height	16.375261
Weight	32.382607
Index	1.355053

dtype: float64

**Gambar 10.** Menghitung standar deviasi.

Standar deviasi menunjukkan seberapa jauh data menyebar dari nilai rata-ratanya. Semakin besar nilai standar deviasi, semakin tinggi tingkat variasi atau penyebaran data pada kolom numerik tersebut.

#### 1.11 Hitung kuartil pertama (Q1)

```
[47] # Hitung kuartil pertama (Q1)
      q1 = df['Height'].quantile(0.25)
      print("Q1\t: ", q1)
```

0s

Q1 : 156.0

**Gambar 11.** Hitung kuartil pertama (Q1).

Kuartil pertama (Q1) menunjukkan 25% data terendah dari kolom *Height*. Artinya, seperempat dari seluruh data tinggi badan berada di bawah nilai Q1 tersebut.

#### 1.12 Hitung kuartil ketiga (Q3)

```
[48] ✓ Os
# Hitung kuartil ketiga (Q3)
q3 = df['Height'].quantile(0.75)
print("Q3\t: ", q3)
```

Q3 : 184.0

**Gambar 12.** Hitung kuartil ketiga (Q3).

Kuartil ketiga (Q3) menunjukkan 75% data terendah dari kolom *Height*. Artinya, tiga perempat data tinggi badan berada di bawah nilai Q3, dan 25% sisanya berada di atasnya.

#### 1.13 Hitung IQR (Interquatile Range)

```
[49] ✓ Os
# Hitung IQR (Interquatile Range)
iqr = q3 - q1
print("IQR\t: ", iqr)
```

IQR : 28.0

**Gambar 13.** Hitung IQR (Interquatile Range).

IQR (Interquartile Range) menunjukkan rentang sebaran data tengah (50% data) antara kuartil pertama (Q1) dan kuartil ketiga (Q3). Nilai ini membantu memahami variasi data utama dan mendeteksi kemungkinan adanya outlier di luar rentang tersebut.

#### 1.14 Untuk membuat statistika deskripsi pada type data int

```
[50]
✓ 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 14.** Untuk membuat statistika deskripsi pada type data int.

Perintah `df.describe()` memberikan statistik deskriptif dari kolom bertipe numerik pada dataset, seperti jumlah data (count), rata-rata (mean), standar deviasi (std), nilai minimum (min), kuartil (25%, 50%, 75%), dan nilai maksimum (max). Hasil ini berguna untuk memahami gambaran umum distribusi data secara cepat.

### 1.15 Menghitung matriks korelasi untuk semua kolom numerik

```
[51]
✓ Os # Menghitung matriks korelasi untuk semua kolom numerik
correlation_matrix = df.corr(numeric_only=True)

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

	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 15.** Menghitung matriks korelasi untuk semua kolom numerik.

Perintah `df.corr(numeric_only=True)` digunakan untuk menghitung matriks korelasi antar semua kolom numerik pada dataset. Hasilnya menunjukkan tingkat hubungan linear antara setiap pasangan variabel, dengan nilai berkisar dari -1 hingga 1.

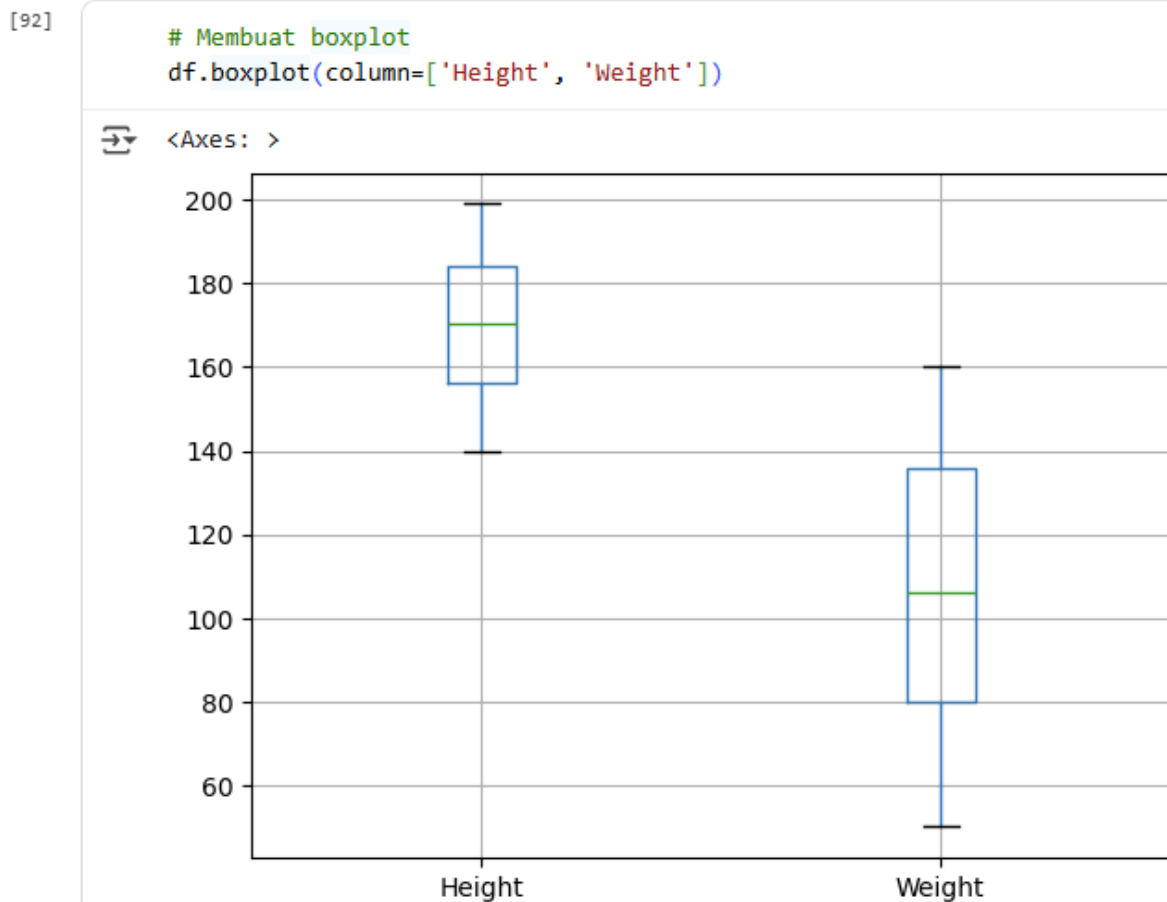
### 1.16 Import numpy

```
[91] ✓ Os # Import numpy
import numpy as np
```

**Gambar 16.** Import numpy.

Perintah `import numpy as np` digunakan untuk mengimpor library NumPy dan memberi alias `np`, sehingga kita bisa menggunakan berbagai fungsi matematika, operasi array, dan perhitungan numerik secara efisien dalam analisis data.

### 1.17 Membuat boxplot



**Gambar 17.** Membuat boxplot.

Digunakan untuk membuat boxplot dari kolom *Height* dan *Weight*. Visualisasi ini membantu melihat sebaran data, median, kuartil, serta outlier secara cepat. Dengan boxplot, kita dapat menganalisis apakah data terdistribusi simetris atau condong, serta mengidentifikasi nilai ekstrem yang perlu diperhatikan.

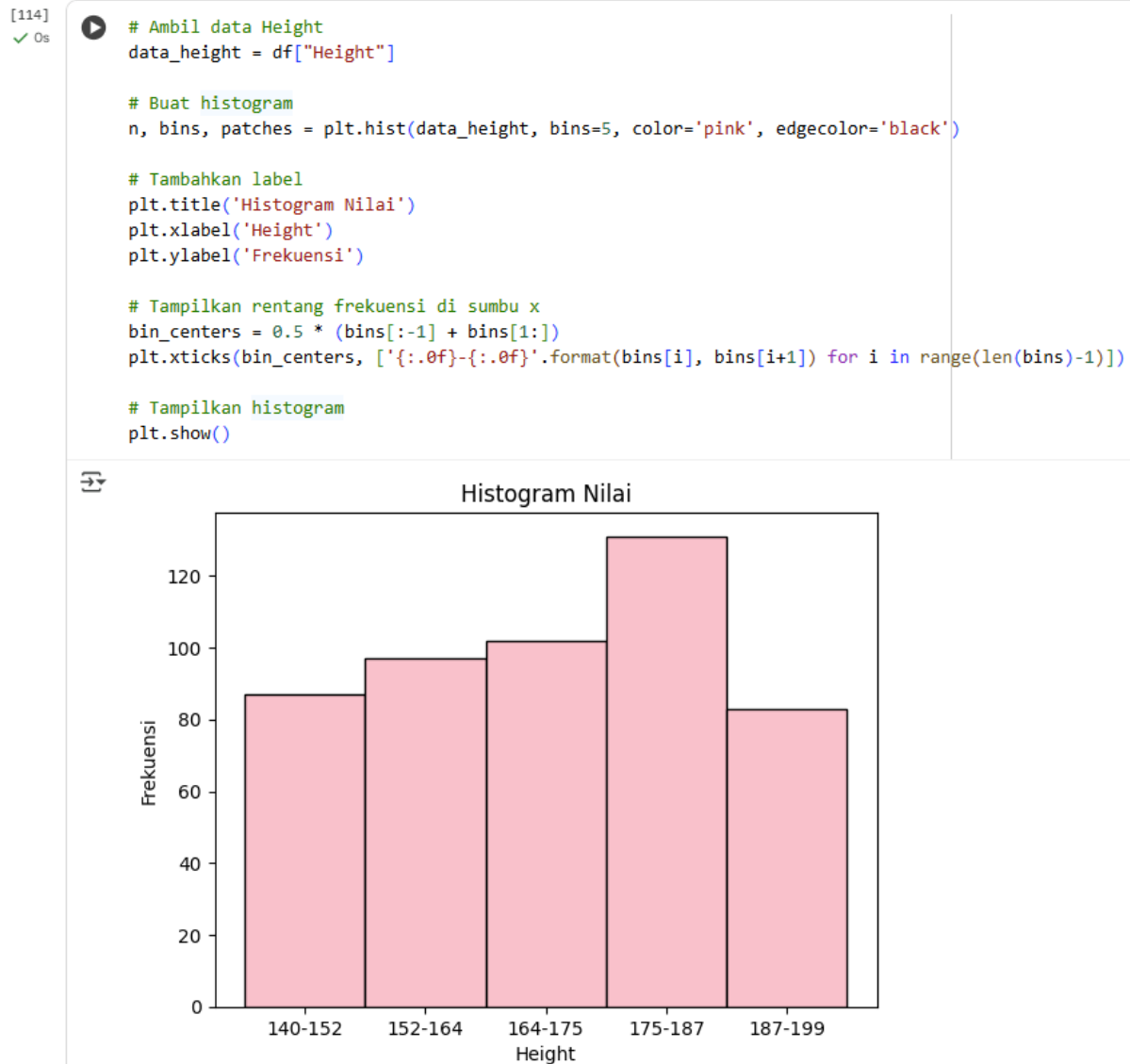
### 1.18 Import matplotlib

```
[113] ✓ Os # Import matplotlib
import matplotlib.pyplot as plt
```

**Gambar 18.** Import matplotlib.

Mengimpor library Matplotlib bagian pyplot dengan alias plt, yang berfungsi untuk membuat berbagai jenis visualisasi data seperti grafik garis, histogram, scatter plot, dan lainnya guna membantu analisis data secara visual.

### 1.19 Membuat histogram

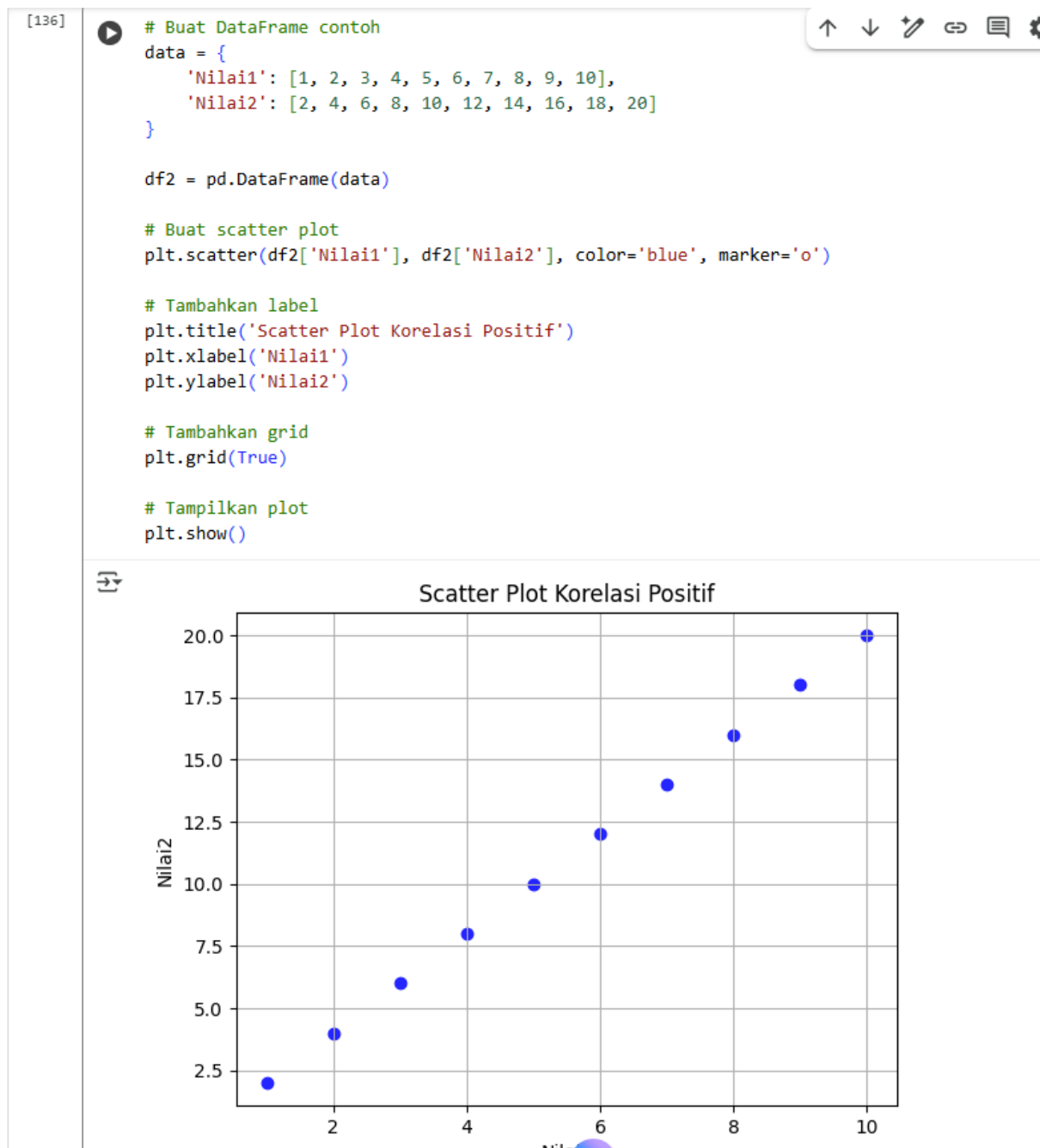


**Gambar 19.** Membuat histogram.

Histogram tersebut menampilkan distribusi frekuensi tinggi badan (*Height*) dari data, sehingga kita dapat melihat penyebaran data dalam rentang tertentu serta kelompok nilai yang paling sering muncul. Visualisasi ini memudahkan dalam memahami pola distribusi data, apakah cenderung merata, terpusat pada nilai tertentu, atau menyebar luas.

### 1.20 Membuat scatter plot korelasi positif





**Gambar 20.** Membuat scatter plot korelasi positif.

Scatter plot tersebut menunjukkan adanya korelasi positif kuat antara kolom *Nilai1* dan *Nilai2*, di mana kenaikan pada *Nilai1* selalu diikuti oleh kenaikan pada *Nilai2*. Visualisasi ini membantu memahami hubungan linear antar variabel dan memverifikasi pola keterkaitan data secara visual.

### 1.21 Membuat scatter plot korelasi negatif

```
[138] # Buat DataFrame contoh
✓ 0s data = {
    'Nilai1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Nilai2': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
}

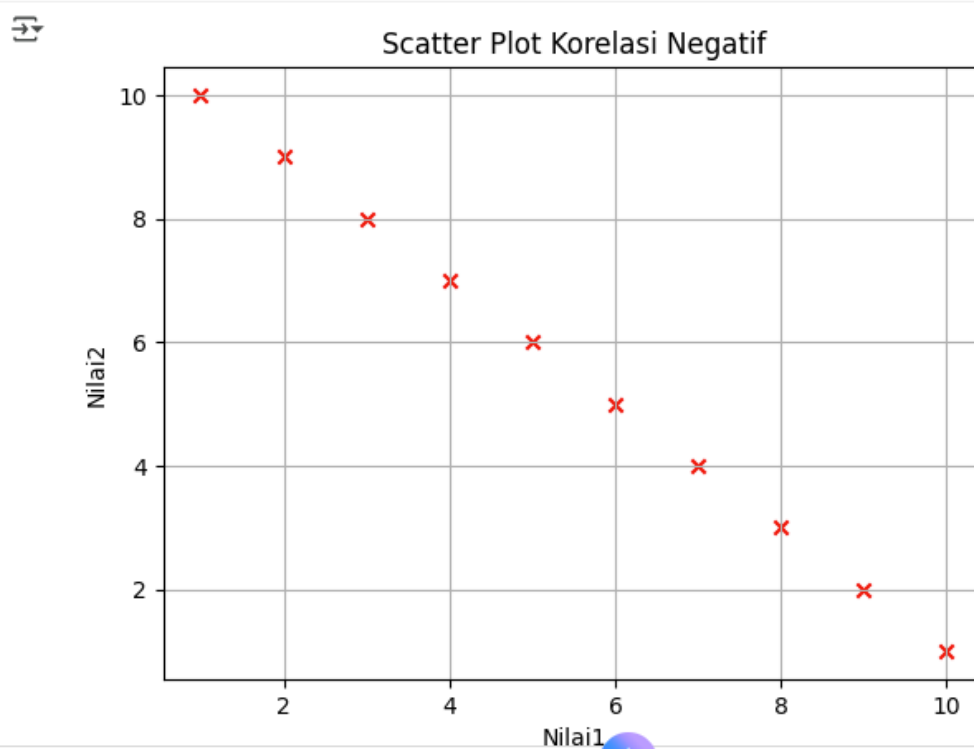
df3 = pd.DataFrame(data)

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

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

# Tambahkan grid
plt.grid(True)

# Tampilkan plot
plt.show()
```



**Gambar 21.** Membuat scatter plot korelasi negatif.

Scatter plot tersebut menunjukkan adanya korelasi negatif kuat antara *Nilai1* dan *Nilai2*, di mana kenaikan pada *Nilai1* diikuti oleh penurunan pada *Nilai2*. Visualisasi ini menggambarkan hubungan terbalik antar variabel dan membantu memahami pola keterkaitan data secara jelas.

## 2. Praktikum mandiri 2

## 2.1 Menghubungkan Collab dengan Google Drive

```
[1]
✓ 12s
# Menghubungkan colab dengan google drive
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

**Gambar 1.** Menghubungkan Collab dengan Google Drive.

Kode tersebut digunakan untuk menghubungkan Google Colab dengan Google Drive sehingga file atau dataset yang tersimpan di Drive dapat diakses dan digunakan langsung dalam lingkungan kerja Colab.

## 2.2 Memanggil data set lewat gdrive

```
[39]
✓ 0s
# Memanggil data set lewat gdrive
path = "/content/drive/MyDrive/Project_Machine_Learning/Praktikum_Mandiri_2/"
```

**Gambar 2.** Memanggil data set lewat gdrive.

Kode tersebut berfungsi untuk menentukan lokasi folder dataset di Google Drive sehingga program dapat mengakses dan memanggil file data yang tersimpan di jalur tersebut.

## 2.3 Import pandas

```
[21]
✓ 0s
# Import pandas
import pandas as pd
```

**Gambar 3.** Import pandas.

Kode tersebut digunakan untuk mengimpor library pandas dengan alias pd, yang berfungsi untuk mengolah, menganalisis, dan memanipulasi data dalam bentuk tabel (DataFrame) di Python.

## 2.4 Membaca file csv menggunakan pandas

```
[41]
✓ 0s
# Membaca file csv menggunakan pandas
df = pd.read_csv(path + "Data/day.csv")
df
```

	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 4.** Membaca file csv menggunakan pandas.

Kode tersebut berfungsi untuk membaca file dataset berformat CSV dari lokasi yang ditentukan dan menyimpannya ke dalam variabel df sebagai DataFrame pandas, sehingga data dapat diolah dan dianalisis lebih lanjut.

## 2.5 Import scikit learn

```
[42]
✓ Os
# Import scikit learn
from sklearn.model_selection import train_test_split
```

**Gambar 5.** Import scikit learn.

Perintah `from sklearn.model_selection import train_test_split` digunakan untuk mengimpor fungsi `train_test_split` dari library *scikit-learn*, yang berfungsi membagi dataset menjadi data training, data validation, dan data testing. Pembagian ini penting agar model machine learning dapat belajar dari sebagian data dan dieuji akurasi pada data lain secara objektif.

## 2.6 Membagi dataset

```
[43]
✓ Os
# Membagi dataset menjadi training (80%) dan testing (20%)
training_df, testing_df = train_test_split(df, test_size=0.2, random_state=42)

# Membagi dataset training menjadi training utama (90%) dan validation (10%)
training_df, validation_df = train_test_split(training_df, test_size=0.1, random_state=42)
```

**Gambar 6.** Membagi dataset.

Pembagian ini penting agar model machine learning tidak overfitting dan hasil prediksinya lebih akurat serta dapat digeneralisasi.

## 2.7 Menampilkan jumlah data pada masing-masing set

```
[44]
✓ Os
# Menampilkan jumlah data pada masing-masing set
print("Jumlah data:")
print(f"Training Set    : {len(training_df)} baris")
print(f"Validation Set    : {len(validation_df)} baris")
print(f"Testing Set       : {len(testing_df)} baris\n")

➡ Jumlah data:
Training Set    : 525 baris
Validation Set   : 59 baris
Testing Set     : 147 baris
```

**Gambar 7.** Menampilkan jumlah data pada masing-masing set.

Kode tersebut menampilkan jumlah baris data pada masing-masing set (*training*, *validation*, dan *testing*). Dengan informasi ini, kita dapat memastikan bahwa pembagian data telah dilakukan dengan benar sesuai proporsi yang diinginkan, sehingga proses pelatihan dan evaluasi model dapat berjalan secara optimal.

## 2.8 Menampilkan 5 baris data teratas

[45]  
✓ Os

```
# Menampilkan 5 baris data teratas
print("==== TRAINING DATA (5 baris pertama) =====")
print(training_df.head())

print("\n\n==== VALIDATION DATA (5 baris pertama) =====")
print(validation_df.head())

print("\n\n==== TESTING DATA (5 baris pertama) =====")
print(testing_df.head())
```

```
===== TRAINING DATA (5 baris pertama) =====
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
```

```

===== VALIDATION DATA (5 baris pertama) =====
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

===== TESTING DATA (5 baris pertama) =====
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 8.** Menampilkan 5 baris data teratas.

Kode tersebut menampilkan 5 baris pertama dari masing-masing set data (*training*, *validation*, dan *testing*). Tujuannya adalah untuk memeriksa isi dan struktur data setelah proses pembagian, memastikan bahwa data tetap konsisten, tidak berubah format, dan siap digunakan untuk proses pelatihan, validasi, maupun pengujian model.