Tugas 2: Statistik Deskriptif dan Probabilitas

Oryza Ayunda Putri - 0110224030

Teknik Informatika, STT Terpadu Nurul Fikri, Depok
E-mail: nasi.tektekmangudin@gmail.com

1. Praktikum

1.1 Membaca File CSV

```
Membaca File CSV:
```

```
import pandas as pd
import numpy as np

df = pd.read_csv('../data/praktikum1.csv')
df
```

Penulis ditugaskan untuk membaca file CSV dengan menggunakan python, dengan rumus **''import pandas as pd"** mengganti kata pandas dengan sebutan pd **"import numpy as np"** mengganti kata numpy menjadi np. Terdapat **df = pd.read_csv('../data/praktikum1.csv')** perintah ini merupakan hal utama dalam membaca data CSV. df= artinya hasil bacaan disimpan pada DataFrame (tabel), pd.read_csv() ini merupakan fungsi dari pandas dalam membaca CSV, '../ praktikum1.csv' lokasi file csv dengan posisi naik satu folder dari file notebook lalu menuju ke lokasi folder data dan mengambil file "praktikum1.csv", **df** digunakan python dalam menampilkan data.

1.2 Melihat Informasi

```
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
```

DataFrame ini berisi **500 data** tentang gender, tinggi badan, berat badan, dan index, dengan semua kolom **lengkap (tanpa missing value)**.

1.3 Menghitung Nilai Mean, Median, dan Modus

```
df['Height'].mean()
```

```
np.float64(169.944)
```

```
df['Height'].median()
```

```
np.float64(170.5)
```

```
df['Height'].mode()
```

```
0    188
Name: Height, dtype: int64
```

1. Hasil: **169.944**
   → Artinya, rata-rata tinggi badan dalam data adalah sekitar **169.94 cm**.
2. Hasil: **170.5**
   → Artinya, nilai tengah dari distribusi data tinggi badan adalah **170.5 cm**. Separuh data memiliki tinggi ≤ 170.5, dan separuh lainnya ≥ 170.5.
3. Hasil: **188**
   → Artinya, nilai tinggi badan yang paling sering muncul dalam data adalah **188 cm**.

1.4 Menghitung Ukuran Persebaran (Variansi & Standar Deviasi):

```
df.var(numeric_only=True)
```

```
Height     268.149162
Weight    1048.633267
Index        1.836168
dtype: float64
```

```
df.std(numeric_only=True)
```

```
Height    16.375261
Weight    32.382607
Index      1.355053
dtype: float64
```

Variance menunjukkan seberapa jauh data menyebar dari rata-ratanya.
Semakin besar nilai variance, semakin besar penyebaran datanya.

- Tinggi badan (Height) punya variasi sekitar **268**.
- Berat badan (Weight) lebih bervariasi, dengan nilai **1048**, artinya sebarannya jauh lebih lebar.
- Kolom Index hanya **1.83**, artinya sangat kecil penyebarannya.

Standard deviation adalah akar kuadrat dari variance, dan punya satuan yang sama dengan datanya, sehingga lebih mudah diinterpretasi:

- Tinggi badan memiliki simpangan baku sekitar **16.38 cm**.
- Berat badan memiliki simpangan baku sekitar **32.38 kg**.
- Kolom Index hanya **1.36**, hampir tidak menyebar.

1.5 Menghitung Kuartil

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

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

iqr = q3 - q1
print('IQR : ', iqr)
```

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

Distribusi tinggi badan memiliki sebaran 50% data di antara **156 cm sampai 184 cm**. Dengan IQR sebesar **28 cm**, artinya variasi data di tengah distribusi relatif sedang (tidak terlalu rapat, tapi juga tidak terlalu lebar).

1.6 Menghitung Statistik Deskriptif Otomatis

```
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

Data berisi 500 observasi dengan tiga variabel numerik. Tinggi badan relatif stabil di sekitar 170 cm, berat badan lebih bervariasi dengan rata-rata 106 kg, sementara index memiliki rentang kecil (0–5) dan cenderung terkonsentrasi pada nilai tinggi (3–5).

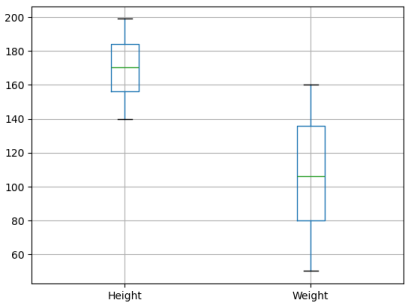
1.7 Menghitung Korelasi

```
correlation_matrix = df.corr(numeric_only=True)

print("Matriks Korelasi:")
print(correlation_matrix)

df.describe()
```

```
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
```

- **Index** memiliki keterkaitan kuat dengan **Weight** (positif) dan sedang dengan **Height** (negatif).
- **Height** hampir tidak punya hubungan dengan **Weight**.
- Jadi, **Index lebih dipengaruhi oleh berat badan dibanding tinggi badan**.

1.8 Visualisasi Data ()




```python
import matplotlib.pyplot as plt

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

plt.show()
```

- **import matplotlib.pyplot as plt**
  → Mengimpor library matplotlib.pyplot untuk membuat visualisasi data.
- **df.boxplot(column=['Height', 'Weight'])**
  → Membuat boxplot untuk kolom Height dan Weight dari DataFrame df.
- **plt.show()**
  → Menampilkan grafik boxplot.

1.9 Vsiualisasi Data (Histogram)

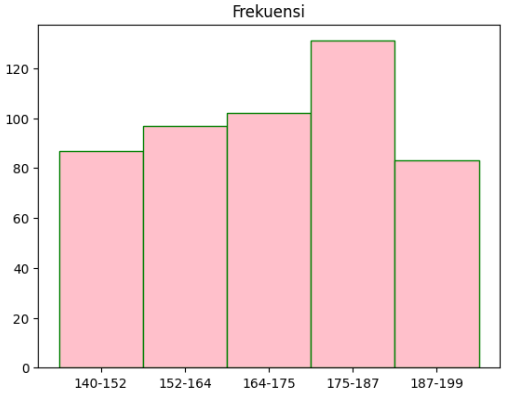
```python
import numpy as pd
import matplotlib.pyplot as plt
import pandas as pd

data_height = df["Height"]
n, bins, patches = plt.hist(data_height, bins=5, color='pink', edgecolor= 'green')

plt.title('Histogram NIlai')
plt.title('Height')
plt.title('Frekuensi')

bin_centers = 0.5 * (bins[:-1] + bins[1:])
plt.xticks(bin_centers, ['{:.0f}-{:.0f}'.format(bins[i], bins[i+1]) for i in range(len(bins) -1)])

plt.show()
```

Di sini diasumsikan kamu sudah punya df (DataFrame pandas) yang punya kolom **"Height"**. Baris ini mengambil semua nilai tinggi dan menyimpannya ke variabel data_height.

- plt.hist() = fungsi untuk membuat **histogram** (diagram batang untuk data numerik).
- data_height = data yang mau ditampilkan.
- bins=5 = membagi data jadi **5 kelompok (interval)**.
- color='pink' = warna isi batang.
- edgecolor='green' = warna tepi batang.

Fungsi plt.hist() menghasilkan 3 nilai:

- n = jumlah data dalam tiap bin (frekuensi).
- bins = batas bawah dan atas dari tiap kelompok.
- patches = objek visual batang histogram.

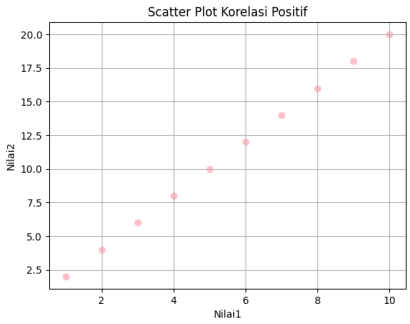


Kesimpulan:

- Mengambil data tinggi dari DataFrame.
- Membuat histogram dengan 5 kelompok data.
- Memberi warna, label, dan judul.
- Menampilkan grafik histogram dengan label rentang di sumbu X.

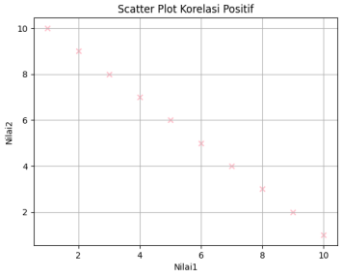
1.10 Scatter Plot (Hubungan Antar Variabel)

```python
import pandas as pd
import matplotlib.pyplot as plt

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)

plt.scatter(df2['Nilai1'], df2['Nilai2'], color='pink', marker='o')

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

plt.grid(True)

plt.show()
```



Kode ini membuat scatter plot yang menunjukkan hubungan antara dua variabel (Nilai1 dan Nilai2). Karena setiap nilai Nilai2 = 2 × Nilai1, maka hasil grafiknya akan berupa garis diagonal naik — menandakan korelasi positif sempurna (r = +1).

```python
import pandas as pd
import matplotlib.pyplot as plt

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

df2 = pd.DataFrame(data)

plt.scatter(df2['Nilai1'], df2['Nilai2'], color='pink', marker='x')

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

plt.grid(True)

plt.show()
```



Kode ini menampilkan **scatter plot korelasi negatif sempurna**, karena:

- Saat **Nilai1 naik**, **Nilai2 turun** secara teratur.
- Hubungannya berbanding terbalik (r = -1).
- Visualnya akan seperti garis menurun dari kiri atas ke kanan bawah.

2. Tugas

```python
import pandas as pd
from sklearn.model_selection import train_test_split

# 1. Baca dataset
df = pd.read_csv("../data/day.csv")

# 2. Split data -> Training (80%), Testing (20%)
train_df, test_df = train_test_split(df, test_size=0.2, random_state=42)

# 3. Dari Training, ambil Validation (10% dari Training)
train_df, val_df = train_test_split(train_df, test_size=0.1, random_state=42)

# 4. Tampilkan jumlah data
print("Jumlah total data:", len(df))
print("Jumlah data Training:", (train_df.shape))
print("Jumlah data Validation:", (val_df.shape))
print("Jumlah data Testing:", (test_df.shape))

# 5. Tampilkan 5 baris pertama tiap set
print("\nData Training (5 baris):")
print(train_df.head())

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

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

Kode ini melakukan langkah-langkah umum dalam **data preprocessing untuk machine learning**:

1. Membaca dataset dari file CSV.
2. Membagi dataset menjadi:
   - **Training set** → untuk melatih model.
   - **Validation set** → untuk mengecek performa model selama pelatihan.
   - **Testing set** → untuk menguji model di data yang belum pernah dilihat.
3. Menampilkan ukuran dan contoh data dari tiap bagian.

Kesimpulan:

```
Jumlah total data: 731
Jumlah data Training: (584, 16)
Jumlah data Validation: (59, 16)
Jumlah data Testing: (147, 16)

Data Training (5 baris):
     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
...
33   1550
300  3747
456  6041
633  7538
```

- **Total data keseluruhan:**
  Jumlah seluruh baris dalam dataset adalah **731 data**.
- **Pembagian data:**
  - **Training set:** 584 data (16 kolom)
    → digunakan untuk melatih model.
  - **Validation set:** 59 data (16 kolom)
    → digunakan untuk mengevaluasi performa model selama pelatihan.
  - **Testing set:** 147 data (16 kolom)
    → digunakan untuk menguji performa akhir model pada data yang belum pernah dilihat.