JOBSHEET 3

PENGENALAN PYTHON DALAM STATISTIKA, DISTRIBUSI NORMAL DAN IMPUTASI DATA

TUJUAN

1. Mahasiswa mampu melakukan operasi statistika dasar menggunakan python
2. Mahasiswa dapat melakukan imputasi data jika ada data yang kosong

PENJELASAN UMUM

- Pada ujicoba ini, menggunakan data ecommerce_consumer_behaviour yang dapat diunduh di sini: https://drive.google.com/file/d/1gNBfaLOm-u17kSQew-SLBmBiQ1EgBijd/view?usp=drive_link
- Untuk menjalankan kode python, dapat menggunakan Google Collaboratory atau Visual Studio Code, dengan menginstal python sebelumnya

Bagian 1: Perintah Dasar

Sebelum melakukan analisis statistik, kita perlu memahami beberapa perintah dasar dalam Python.

```python
# Menampilkan teks
print("Hello, Statistik!")

# Variabel dan tipe data
angka = 10
teks = "Data Statistik"
boolean = True

# Struktur data dasar
list_data = [1, 2, 3, 4, 5]
tuple_data = (10, 20, 30)
dict_data = {"mean": 25, "median": 30}

print(list_data, tuple_data, dict_data)
```

Silakan dicoba, bagaimana hasilnya?

Bagian 2: Membuka Dataset

Setelah mengetahui perintah-perintah dasar dalam python, selanjutnya yang perlu dipelajari adalah cara membuka dataset.

```python
# Import necessary library
import pandas as pd

# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

# Define the file path in Google Drive
file_path = '/content/drive/MyDrive/Lab Statistika 2025/Job Sheet 3/Ecommerce_Consumer_Behavior_Analysis_Data.csv'

# Open the CSV file using pandas
try:
  # Read the CSV file into a pandas DataFrame
  data = pd.read_csv(file_path)

  # Print the first few rows of the DataFrame to verify
  print(data.head())
```

Catatan:
Pada contoh di atas, data diambil dari Google Drive dengan lokasi file yang sama dengan file python. Untuk cara membuka data yang lain, silakan dicari.

Silakan dicoba dan bagaimana hasilnya?

Bagian 3: Perhitungan Pemusatan Data

Pada bagian ini akan ditunjukkan bagaimana cara menghitung pemusatan data, yakni Mean, Median dan Modus, berdasarkan data yang ada.

```python
import pandas as pd
from google.colab import drive
import statistics

# Mount Google Drive
drive.mount('/content/drive')

# Define the file path in Google Drive
file_path = '/content/drive/MyDrive/Lab Statistika 2025/Job Sheet 3/Ecommerce_Consumer_Behavior_Analysis_Data.csv'

# Open the CSV file using pandas
try:
  # Read the CSV file into a pandas DataFrame
  data = pd.read_csv(file_path)

  # Calculate Mean, Median, and Mode for relevant numerical columns
  for column in data.select_dtypes(include=['number']):  # Only calculate for numerical columns
    if data[column].notnull().any(): #check for empty columns
      mean_val = data[column].mean()
      median_val = data[column].median()
      try:
        mode_val = statistics.mode(data[column]) # Use statistics.mode for single mode
      except statistics.StatisticsError: # Handle cases where there might be multiple modes or no unique mode
          mode_val = "No unique mode"

      print(f"--- Analysis for column '{column}' ---")
      print(f"Mean: {mean_val} (The average value of the data)")
      print(f"Median: {median_val} (The middle value when the data is sorted)")
      print(f"Mode: {mode_val} (The most frequent value in the data)")
      print()
    else:
      print(f"--- Analysis for column '{column}' ---")
      print("Column contains only null values. Cannot calculate statistics.")
      print()
```

Silakan dicoba, bagaimana hasilnya?

Bagian 4: Perhitungan Variablitas Data

Pada bagian ini akan dilakukan penghitungan variabilitas data, seperti variance dan standart deviasi.

```
# Assuming 'data' DataFrame is already loaded as in the previous code

# Calculate Variance and Standard Deviation for relevant numerical columns
for column in data.select_dtypes(include=['number']):
    if data[column].notnull().any():  # Check for empty columns
        variance_val = data[column].var()
        std_dev_val = data[column].std()

        print(f"--- Analysis for column '{column}' ---")
        print(f"Variance: {variance_val} (Measures the spread of the data)")
        print(f"Standard Deviation: {std_dev_val} (Square root of the variance)")
        print()
    else:
        print(f"--- Analysis for column '{column}' ---")
        print("Column contains only null values. Cannot calculate statistics.")
        print()
```

Silakan dicoba, bagaimana hasilnya?

Bagian 5: Penyajian Data

Pada bagian ini akan dicontohkan beberapa cara menyajikan data, menggunakan data ecommerce_consumer_behaviour

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import drive

# Mount Google Drive
drive.mount('/content/drive')

# Define the file path in Google Drive
file_path = '/content/drive/MyDrive/Lab Statistika 2025/Job Sheet 3/Ecommerce_Consumer_Behavior_Analysis_Data.csv'

try:
    # Read the CSV file into a pandas DataFrame
    data = pd.read_csv(file_path)

    # --- Histograms ---
    for col in data.select_dtypes(include=['number']):
        if data[col].notnull().any():
            plt.figure(figsize=(8, 6))  # Adjust figure size as needed
            sns.histplot(data[col], kde=True)  # KDE adds a kernel density estimate
            plt.title(f'Distribution of {col}')
            plt.xlabel(col)
            plt.ylabel('Frequency')
            plt.show()
```

```python
# --- Box plots ---
for col in data.select_dtypes(include=['number']):
    if data[col].notnull().any():
      plt.figure(figsize=(8, 6))
      sns.boxplot(y=data[col])
      plt.title(f'Box Plot of {col}')
      plt.ylabel(col)
      plt.show()

# --- Scatter plots (example: if you have two numerical columns to compare)---
# Replace 'col1' and 'col2' with actual column names
if 'Purchase Amount (USD)' in data.columns and 'Age' in data.columns :
  if data['Purchase Amount (USD)'].notnull().any() and data['Age'].notnull().any():
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='Age', y='Purchase Amount (USD)', data=data)
    plt.title('Scatter Plot of Purchase Amount vs. Age')
    plt.xlabel('Age')
    plt.ylabel('Purchase Amount (USD)')
    plt.show()

# --- Bar plots (example: for categorical data) ---
# Replace 'Category' with an actual categorical column name
if 'Gender' in data.columns:
  if data['Gender'].notnull().any():
      plt.figure(figsize=(8, 6))
      sns.countplot(x='Gender', data=data)
      plt.title('Distribution of Gender')
      plt.xlabel('Gender')
      plt.ylabel('Count')
      plt.show()
```

Silakan dicoba, bagaimana hasilnya?

Bagian 6: Mengecek Data Kosong

Pada bagian ini akan dicontohkan bagaimana cara mengecek apakah ada data kosong atau tidak.

```
# Check for missing values in the DataFrame
missing_values = data.isnull().sum()

# Print the number of missing values for each column
print("--- Missing Values per Column ---")
print(missing_values)

# Check if there are any missing values in the entire DataFrame
if missing_values.sum() > 0:
    print("\nThe DataFrame contains missing values.")
else:
    print("\nThe DataFrame does not contain any missing values.")
```

Silakan dicoba, bagaimana hasilnya?

Bagian 7: Treatment Data Kosong

Jika ada data kosong, maka dilakukan imputasi data, berikut ini cara melakukannya.

```python
import pandas as pd
# Check for missing values in the DataFrame
missing_values = data.isnull().sum()

# Print the number of missing values for each column
print("--- Missing Values per Column ---")
print(missing_values)

# Iterate through columns with missing values and perform imputation
for column in data.columns:
    if data[column].isnull().any():
        if pd.api.types.is_numeric_dtype(data[column]):
            # For numerical columns, use the mean to fill missing values
            #This is just an example. You can use median, or other imputation methods.
            data[column].fillna(data[column].mean(), inplace=True)
            print(f"Imputed missing values in '{column}' with the mean.")
        elif pd.api.types.is_object_dtype(data[column]):
            # For categorical columns (objects), use the mode to fill missing values
            data[column].fillna(data[column].mode()[0], inplace=True) #mode()[0] is used because mode() returns a series.
            print(f"Imputed missing values in '{column}' with the mode.")
        else:
            print(f"Column '{column}' has a data type that is not handled by this code.")

# Verify if missing values have been imputed
missing_values_after_imputation = data.isnull().sum()
print("\n--- Missing Values After Imputation ---")
missing_values_after_imputation
```

Silakan dicoba, bagaimana hasilnya?

Tugas

Tugas 1

- Download dataset Titanic pada https://www.kaggle.com/datasets/yasserh/titanic-dataset/code
- Cek, apakah variabel “Age” dan “Fare” memiliki data yang nilainya hilang. Jika ya, berapa jumlahnya?
- Cek pola distribusi data pada “Age” dan “Fare”.
- Gunakan teknik *simple data imputation* yang tepat untuk mensubtitusi data yang hilang.

Tugas 2

- Imputasi data yang hilang pada “Age” dengan menggunakan mean
- Hitung z-score dari “Age” untuk semua data
- Tentukan jumlah outlier dari “Age” → $|Z| > 3$
- Diasumsuikan ”Age” terdistribusi secara normal, berapa peluang "$Age" < 20$?