Name : Yash Chikhale
Class : D15C
Roll no. : 50

DMBI Exp-2

Aim : To perform data preprocessing using python.

Theory :

Data Preprocessing

Data preprocessing refers to the process of converting raw, messy data into a clean and organized format suitable for analysis or machine learning tasks. In real-world scenarios, datasets often contain missing values, duplicate records, inconsistent formats, and outliers. If left unaddressed, these issues can result in misleading insights and poor model accuracy. Preprocessing ensures that the dataset is reliable, consistent, and ready for further exploration or modeling.

In this experiment, the following steps are carried out:

1. **Handling Missing Values**
   Missing data occurs when certain values are not recorded for specific observations. This may happen due to human error, technical issues, or incomplete data collection. For numerical attributes such as *Age*, missing entries can be replaced using the median of the available values. The median is often preferred over the mean since it is less sensitive to extreme values (outliers) and provides a more robust replacement.

2. **Removing Duplicates**
   Duplicate records are repeated entries that can distort statistical summaries and lead to biased results. Eliminating duplicates ensures that every observation in the dataset is unique, improving the accuracy of analysis and preventing redundant computation.

3. **Encoding Categorical Variables**
   Categorical features contain non-numeric labels (e.g., Male/Female, USA/India). Since most algorithms require numerical inputs, these categories must be converted into numbers.

   ○ *Label Encoding* assigns each category a unique integer.
   ○ *One-Hot Encoding* creates separate binary columns for each category, avoiding any false sense of ordering.

4. **Correcting Data Types**
   Each column should be stored in the appropriate data type. For instance, *Salary* should

be represented as a floating-point number instead of a string. Proper data typing avoids calculation errors and ensures smooth integration with analysis or ML tools.

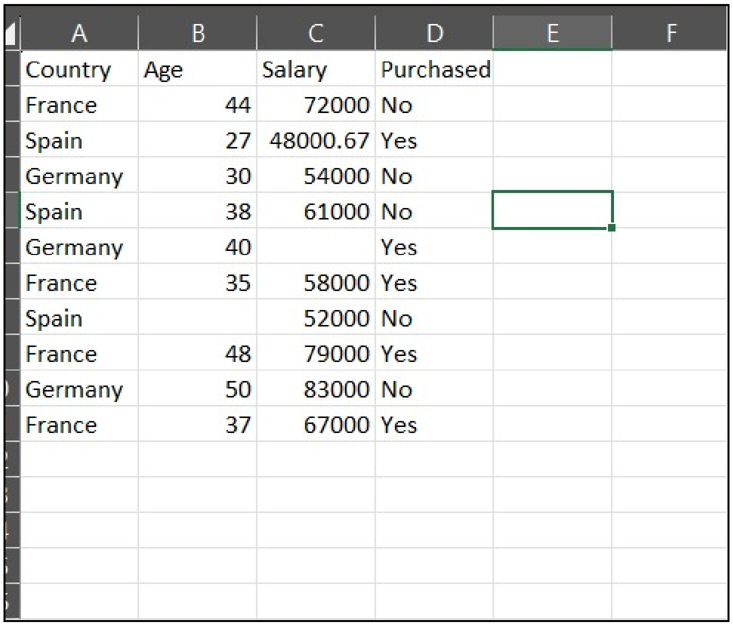
5. **Handling Outliers**
   Outliers are data points that deviate significantly from the majority, such as *Age > 100*. These values may result from data entry errors or rare but valid cases. Outliers can distort statistical measures and affect model training. Depending on context, they may be removed or treated using statistical methods like the Interquartile Range (IQR) method.

Order of Operations

To ensure efficiency and accuracy, preprocessing should follow a logical sequence:

- Handle missing values → avoids propagation of NaN values.
- Remove duplicates → prevents repeated processing of the same record.
- Encode categorical variables → prepares data for numerical operations.
- Fix data types → ensures accurate computations.
- Handle outliers → final step before applying models.

INPUT CSV DATASET :



Country	Age	Salary	Purchased
France	44	72000	No
Spain	27	48000.67	Yes
Germany	30	54000	No
Spain	38	61000	No
Germany	40		Yes
France	35	58000	Yes
Spain		52000	No
France	48	79000	Yes
Germany	50	83000	No
France	37	67000	Yes

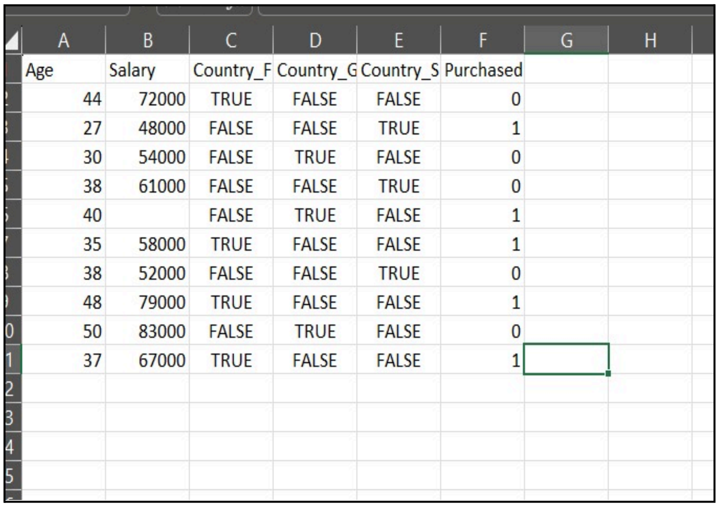
Code :

```
import pandas as pd
import tkinter as tk
from tkinter import filedialog
```

```python
# Hide the main tkinter window
root = tk.Tk()
root.withdraw()
# Dialog box to select the CSV file
input_path = filedialog.askopenfilename(title="Select the input CSV file",
filetypes=[("CSV Files", "*.csv")])
df = pd.read_csv(input_path)

# 1. Handle missing values (on age with median)
df['Age'] = df['Age'].fillna(df['Age'].median())

# 2. Remove duplicates
df = df.drop_duplicates()

# 3. Encode categorical variables
# One-hot encoding for 'Country'
country_encoded = pd.get_dummies(df['Country'], prefix='Country')
# Label encoding for 'Purchased'
purchased_encoded = df['Purchased'].map({'No': 0, 'Yes': 1}).to_frame()

# Print the encoded columns in the Terminal (as per Question 3)
print("Encoded 'Country' columns:")
print(country_encoded.head())
print("\nEncoded 'Purchased' column:")
print(purchased_encoded.head())

# 4. Fix datatypes (salary as float)
df['Salary'] = pd.to_numeric(df['Salary'], errors='coerce')

# 5. Handle outliers (eg age > 100)
df = df[df['Age'] <= 100]

# Remove the original categorical columns and add the encoded ones for the
output CSV
df = df.drop(['Country', 'Purchased'], axis=1)
df = pd.concat([df, country_encoded, purchased_encoded], axis=1)

# Dialog box to select destination for output CSV file
output_path = filedialog.asksaveasfilename(title="Save the cleaned CSV
as", defaultextension=".csv", filetypes=[("CSV Files", "*.csv")])
```

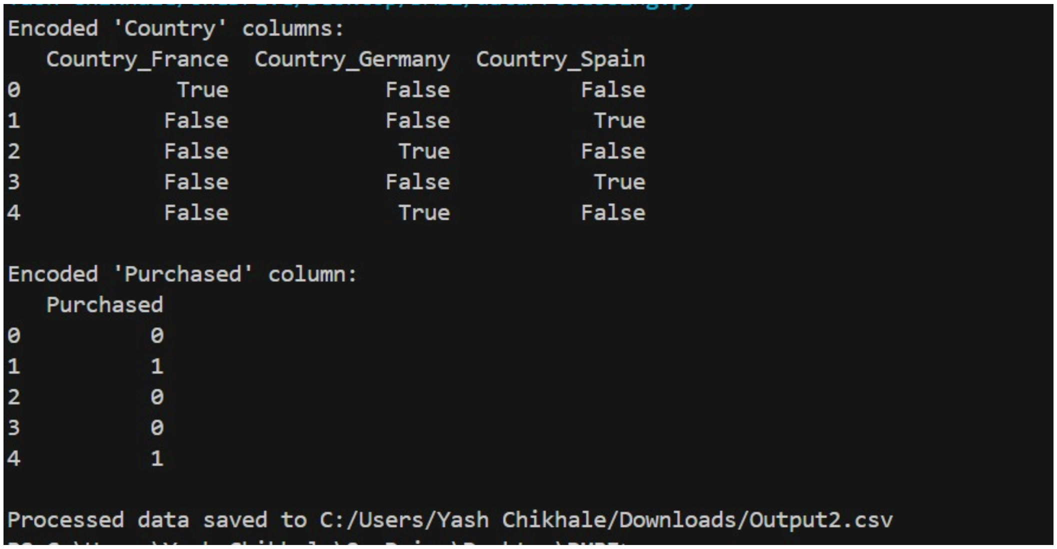
```
df.to_csv(output_path, index=False)
print(f"\nProcessed data saved to {output_path}")
```

OUTPUT CSV DATASET :



A	B	C	D	E	F	G	H
Age	Salary	Country_F	Country_G	Country_S	Purchased		
44	72000	TRUE	FALSE	FALSE	0		
27	48000	FALSE	FALSE	TRUE	1		
30	54000	FALSE	TRUE	FALSE	0		
38	61000	FALSE	FALSE	TRUE	0		
40		FALSE	TRUE	FALSE	1		
35	58000	TRUE	FALSE	FALSE	1		
38	52000	FALSE	FALSE	TRUE	0		
48	79000	TRUE	FALSE	FALSE	1		
50	83000	FALSE	TRUE	FALSE	0		
37	67000	TRUE	FALSE	FALSE	1		

3. encode categorical variables (separate)



```
Encoded 'Country' columns:
   Country_France  Country_Germany  Country_Spain
0            True            False          False
1           False            False           True
2           False             True          False
3           False            False           True
4           False             True          False

Encoded 'Purchased' column:
   Purchased
0          0
1          1
2          0
3          0
4          1

Processed data saved to C:/Users/Yash Chikhale/Downloads/Output2.csv
```