AI ASSISSTED CODING

LAB TEST-4

SET-C

QUESTION-1: AI-Assisted Data Cleaning

Task:

You are given a raw dataset named health_data.csv containing missing values, inconsistent text entries, and duplicate rows. Use an AI-assisted coding tool (e.g., ChatGPT, GitHub Copilot, or Google Gemini) to generate a Python script that:

1. Detects and removes duplicate rows.

2. Handles missing values using appropriate techniques (mean, median, or mode).

3. Standardizes inconsistent categorical data entries (e.g., “Male”, “male”, “M”).

Instructions:

• Verify that the dataset is clean by displaying the first 5 rows before and after cleaning.

• Document which AI suggestions you used and how they improved your code.

Given data set:

	A	B	C	D	E	F
1	Patient_ID	Age	Gender	Blood_Pressure	Cholesterol	Diabetic
2	1	25	Male	120	200	Yes
3	2	30	Female		180	No
4	3		M	130	190	No
5	4	45	F	140		
6	5	50	male		210	Yes
7	6		female	110	220	No
8	7	35		125	195	yes
9	8	60	M			No
10	9	40	F	135	185	
11	10	55	Male	145	205	Yes
12						
13						

CODE:

```
# health_data_cleaning.py

import pandas as pd
import numpy as np

# 1. Load dataset
df = pd.read_csv("health_data.csv")

print("---- Before Cleaning ----")
print(df.head())

# 2. Remove duplicate rows
df = df.drop_duplicates()

# 3. Handle missing values
# Identify numeric and categorical columns
numeric_cols = df.select_dtypes(include=[np.number]).columns
categorical_cols = df.select_dtypes(exclude=[np.number]).columns

# Fill missing numeric values with mean
for col in numeric_cols:
    df[col].fillna(df[col].mean(), inplace=True)

# Fill missing categorical values with mode
for col in categorical_cols:
    df[col].fillna(df[col].mode()[0], inplace=True)

# 4. Standardize inconsistent categorical data entries
# Example: Gender column with values like "Male", "male", "M"
if 'Gender' in df.columns:
    df['Gender'] = df['Gender'].str.strip().str.lower()
    df['Gender'] = df['Gender'].replace({
        'm': 'male',
        'male': 'male',
        'f': 'female',
        'female': 'female'
    })
    df['Gender'] = df['Gender'].str.title()  # Capitalize first letter
(Male, Female)

# 5. Verify cleaning results
print("\n---- After Cleaning ----")
print(df.head())
```

```python
# 6. Save cleaned dataset
df.to_csv("health_data_cleaned.csv", index=False)
print("\n✅Cleaned dataset saved as 'health_data_cleaned.csv'")
```

OUTPUT:

```
---- Before Cleaning ----
   Patient_ID   Age  Gender  Blood_Pressure  Cholesterol Diabetic
0           1  25.0    Male           120.0        200.0      Yes
1           2  30.0  Female             NaN        180.0       No
2           3   NaN       M           130.0        190.0       No
3           4  45.0       F           140.0          NaN      NaN
4           5  50.0    male             NaN        210.0      Yes

---- After Cleaning ----
   Patient_ID   Age  Gender  Blood_Pressure  Cholesterol Diabetic
0           1  25.0    Male      120.000000      200.000      Yes
1           2  30.0  Female      129.285714      180.000       No
2           3  42.5    Male      130.000000      190.000       No
3           4  45.0  Female      140.000000      198.125       No
4           5  50.0    Male      129.285714      210.000      Yes

✅ Cleaned dataset saved as 'health_data_cleaned.csv'
```

EXPLANATION:

- **1. Load dataset**: Reads the `health_data.csv` file into a pandas DataFrame named `df`.
- **2. Remove duplicate rows**: Identifies and removes any exact duplicate rows from the DataFrame.
- **3. Handle missing values**:
  - It first separates columns into numeric and categorical types.
  - Then, it fills missing values (NaN) in numeric columns with the mean of that column.
  - For categorical columns, it fills missing values with the mode (most frequent value) of that column.
- **4. Standardize inconsistent categorical data entries**:
  - Specifically targets the 'Gender' column (if it exists).
  - It cleans up entries by stripping whitespace, converting to lowercase, and then replacing common variations ('m', 'f') with standardized terms ('male', 'female'). Finally, it capitalizes the first letter for consistent display (e.g., 'Male', 'Female').
- **5. Verify cleaning results**: Prints the first few rows of the DataFrame before and after cleaning, allowing for a quick visual check of the changes.
- **6. Save cleaned dataset**: Saves the processed DataFrame to a new CSV file named `health_data_cleaned.csv` without including the DataFrame index.

QUESTION-2: AI-Assisted Data Preprocessing .

Task: You are provided with a dataset named **customer_data.csv** containing both numerical and categorical features. Use AI tools to generate a **Python preprocessing script** that performs:

1. **Label Encoding or One-Hot Encoding** for categorical variables.
2. **Normalization or Standardization** for numerical features.
3. Splitting the dataset into **training and testing sets (80:20)**.

Instructions:

- Display statistical summaries (.describe() output) before and after preprocessing.
- Explain how the preprocessing steps can improve the performance of a machine learning model.
- Highlight any optimizations or corrections made based on AI tool recommendations

GIVEN DATASET:

A	B	C	D	E	F
Customer	Age	Gender	Annual_Income	Spending_Score	Membership_Level
1	22	Male	40000	30	Silver
2	25	Female	50000	40	Gold
3		male	60000		Platinum
4	35	F		70	
5	40		80000	60	Gold
6	45	M	75000		Silver
7		Female	65000	50	
8	28	male		80	Platinum
9	30	F	70000	55	Gold
10	33	Female	72000	65	Silver

CODE:

```
# customer_data_preprocessing.py

import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder,
StandardScaler, MinMaxScaler
from sklearn.model_selection import train_test_split

# 1. Load dataset
df = pd.read_csv("customer_data.csv")

print("---- Before Preprocessing ----")
print(df.head())
print("\nStatistical Summary (Before):")
print(df.describe(include='all'))

# Separate features by data type
numeric_cols = df.select_dtypes(include=[np.number]).columns
categorical_cols = df.select_dtypes(exclude=[np.number]).columns

# 2. Handle categorical data (Encoding)
# Option 1: One-Hot Encoding for nominal variables
df_encoded = pd.get_dummies(df, columns=categorical_cols, drop_first=True)

# 3. Normalize or Standardize numerical features
# Option: Standardization (mean=0, std=1)
scaler = StandardScaler()
df_encoded[numeric_cols] = scaler.fit_transform(df_encoded[numeric_cols])

# Alternative (Normalization: MinMaxScaler)
# scaler = MinMaxScaler()
# df_encoded[numeric_cols] =
scaler.fit_transform(df_encoded[numeric_cols])

# 4. Split dataset into training and testing sets (80:20)
train_df, test_df = train_test_split(df_encoded, test_size=0.2,
random_state=42)

print("\n---- After Preprocessing ----")
print(train_df.head())
print("\nStatistical Summary (After):")
print(train_df.describe(include='all'))
```

```python
# 5. Save preprocessed datasets
train_df.to_csv("customer_data_train.csv", index=False)
test_df.to_csv("customer_data_test.csv", index=False)
print("\n✅ Preprocessed data saved as 'customer_data_train.csv' and
'customer_data_test.csv'")
```

OUTPUT:

```
---- Before Preprocessing ----
   Customer_ID   Age  Gender  Annual_Income  Spending_Score Membership_Level
0            1  22.0    Male        40000.0            30.0           Silver
1            2  25.0  Female        50000.0            40.0             Gold
2            3   NaN    male        60000.0             NaN         Platinum
3            4  35.0       F            NaN            70.0              NaN
4            5  40.0     NaN        80000.0            60.0             Gold

Statistical Summary (Before):
        Customer_ID        Age  Gender  Annual_Income  Spending_Score  \
count      10.00000   8.000000       9       8.000000        8.000000
unique          NaN        NaN       5            NaN             NaN
top             NaN        NaN  Female            NaN             NaN
freq            NaN        NaN       3            NaN             NaN
mean        5.50000  32.250000     NaN   64000.000000       56.250000
std         3.02765   7.667184     NaN   13448.313755       16.201852
min         1.00000  22.000000     NaN   40000.000000       30.000000
25%         3.25000  27.250000     NaN   57500.000000       47.500000
50%         5.50000  31.500000     NaN   67500.000000       57.500000
75%         7.75000  36.250000     NaN   72750.000000       66.250000
max        10.00000  45.000000     NaN   80000.000000       80.000000

       Membership_Level
count                 8
unique                3
top              Silver
freq                  3
mean                NaN
std                 NaN
min                 NaN
25%                 NaN
50%                 NaN
75%                 NaN
max                 NaN
```

```
---- After Preprocessing ----
   Customer_ID       Age  Annual_Income  Spending_Score  Gender_Female  \
5     0.174078  1.777748       0.874421             NaN          False
0    -1.566699 -1.429170      -1.907829       -1.732051          False
7     0.870388 -0.592583            NaN        1.567094          False
2    -0.870388       NaN      -0.317971             NaN          False
9     1.566699  0.104573       0.635943        0.577350           True

   Gender_M  Gender_Male  Gender_male  Membership_Level_Platinum  \
5      True        False        False                      False
0     False         True        False                      False
7     False        False         True                       True
2     False        False         True                       True
9     False        False        False                      False

   Membership_Level_Silver
5                     True
0                     True
7                    False
2                    False
9                     True
```

```
Statistical Summary (After):
        Customer_ID       Age  Annual_Income  Spending_Score Gender_Female  \
count      8.000000  6.000000       6.000000        6.000000             8
unique          NaN       NaN            NaN             NaN             2
top             NaN       NaN            NaN             NaN         False
freq            NaN       NaN            NaN             NaN             6
mean       0.000000  0.220766       0.105990        0.192450           NaN
std        1.002162  1.147805       1.137239        1.150766           NaN
min       -1.566699 -1.429170      -1.907829       -1.732051           NaN
25%       -0.609272 -0.418294      -0.218605       -0.247436           NaN
50%        0.000000  0.244005       0.357718        0.412393           NaN
75%        0.609272  0.906303       0.814802        0.824786           NaN
max        1.566699  1.777748       1.271886        1.567094           NaN

       Gender_M Gender_Male Gender_male Membership_Level_Platinum  \
count         8           8           8                         8
unique        2           2           2                         2
top       False       False       False                     False
freq          7           7           6                         6
mean        NaN         NaN         NaN                       NaN
std         NaN         NaN         NaN                       NaN
min         NaN         NaN         NaN                       NaN
25%         NaN         NaN         NaN                       NaN
50%         NaN         NaN         NaN                       NaN
75%         NaN         NaN         NaN                       NaN
max         NaN         NaN         NaN                       NaN

       Membership_Level_Silver
count                        8
unique                       2
top                      False
freq                         5
mean                       NaN
std                        NaN
min                        NaN
25%                        NaN
50%                        NaN
75%                        NaN
max                        NaN

✅ Preprocessed data saved as 'customer_data_train.csv' and 'customer_data_test.csv'
```