TASK 1 – 26/05/2025 – ELEVATE LABS – ANGELINA NAYAK – nayakangelina000@gmail.com

I am cleaning data using Python in a Jupyter Notebook, and the dataset I am working with is Customer - Sheet1.csv.

Here's a summary of the steps I have taken so far:

1. **Dataset Upload & Kernel Setup:**
    - I uploaded the Customer - Sheet1.csv file to my Jupyter Notebook environment.
    - I opened a new Python kernel within my Jupyter Notebook and named it Task 1 Elevate Labs Customer Data.
2. **Library Imports:**

I ran the following code block to import essential Python libraries:

Python

```
#Libraries
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns


#Ignore the warnings
import warnings
warnings.filterwarnings("ignore")
```

3. **Dataset Loading:**

I successfully loaded Customer - Sheet1.csv into a Pandas DataFrame named df and displayed its contents.

Python

```
df = pd.read_csv('Customer - Sheet1.csv')
df
```

This showed my DataFrame contained 2005 rows and 11 columns.

4. **Initial Data Information (df.info()):**

I ran the following code to get a summary of my DataFrame, including data types and non-null counts:

Python

```
df.info()
```

The output indicated that the Profession column had 1970 non-null entries and the Season column had 1976 non-null entries, suggesting missing values in these columns.

5. **Column Header Verification:**

I executed the following code to retrieve and display a list of all column headers to check for errors:

Python

```
# We want to check all the column headers to avoid errors
df = pd.DataFrame(df)
```

```
df.head()

column_list = list(df.columns)

column_list
```

The result confirmed the existing column names.

6. **Drop 'Profession' Column:**

I dropped the Profession column from the DataFrame, as it was not needed for data analysis, using this code:

Python

```
#Drop the Profession column - not needed in data analysis

df.drop(columns="Profession", inplace=True)

df.head()
```

7. **Add 'Age Group' Column:**

I created a new column named Age Group by categorizing the Age column into predefined bins using the following code:

Python

```
df['Age Group'] = pd.cut(df['Age'], bins=[0, 18, 25, 35, 45, 55, 65, float('inf')],

        labels=['Under 18', '18-24', '25-34', '35-44', '45-54', '55-64', '65+'],

        include_lowest=True)
```

8. **Check for Missing Values (df.isna().sum()):**

I ran the following code to check for N/A or null values in the DataFrame:

Python

```
#check for N/A values

df.isna().sum()
```

9. **Drop Nulls in 'Season' Column:**

I identified that the Season column had 29 null values and subsequently dropped the rows containing these nulls using this code:

Python

```
#Drop N/A values in Season column

df = df.dropna(subset = ["Season"])
```

10. **Check and Drop Duplicate Customer IDs:**

I checked for duplicate CustomerID values using the code below, which reported 5 duplicate entries:

Python

```
#Check the column for duplicates

    df.duplicated('CustomerID').sum()
```

Following this, I dropped these duplicate rows, keeping only the first occurrence for each CustomerID, with this code:

Python

```
df.drop_duplicates(subset = "CustomerID", inplace = True)
```

```
df.duplicated('CustomerID').sum()
```

11. **Data Description (df.describe()):**

I ran the following code to view descriptive statistics for the numerical columns (CustomerID, Age, Purchase Amount) in my cleaned dataset:

```
Python

#Description of the data

df.describe()
```

The output of this command provided the count, mean, std, min, quartiles, and max for these columns.

The data got successfully cleaned and uploaded to output path using this code :

```
import os


output_folder_path = r'C:\Users\nayak\Downloads\ELEVATE LABS\TASK 1'


# Ensure the directory exists.
if not os.path.exists(output_folder_path):
  os.makedirs(output_folder_path)
  print(f"Created directory: {output_folder_path}")


# Define the full path for your cleaned CSV file
cleaned_file_name = 'Cleaned Customer Data Task 1 - Elevate Labs.ipynb'
full_save_path = os.path.join(output_folder_path, cleaned_file_name)


# Save the DataFrame to CSV
df.to_csv(full_save_path, index=False)


print(f"\nYour cleaned dataset has been successfully saved to: {full_save_path}")
```