Mahatma Education Society's

Pillai College of Arts, Commerce & Science (Autonomous)

Affiliated to University of Mumbai

'NAAC Accredited 'A' grade (3 cycles)'
'Best College Award' by University of Mumbai
ISO 9001:2015 Certified

PROJECT REPORT ON

"Online Shopping Dataset"

IN PARTIAL FULFILLMENT OF

BACHELOR OF INFORMATION TECHNOLOGY

SEMESTER 2025-2026

SUBMITTED BY: -Dnyaneshwari Pandurang Javal

ROLL NO:-5526

SUBMITTED ON -02.02.2026

Mahatma Education Society's

Pillai College of Arts, Commerce & Science

(Autonomous)

Affiliated to University of Mumbai

NAAC Accredited 'A' grade (3 cycles)

Best College Award by University of Mumbai

ISO 9001:2015 Certified

pcacs

This is to verify that Miss.**Dnyaneshwari Pandurang Javal** Examination Seat no.**5526** has successfully completed **Project** of the subject **Operating System**.In partial fulfillment for the degree of B.Sc.(Information Technology) SEM IV, affiliated to the University of Mumbai for the academic year 2025-2026.

InternalExaminer

INDEX

1. INTRODUCTION

2. OBJECTIVES OF THE PROJECT

3. TOOLS AND TECHNOLOGIES USED

4. DATASET DESCRIPTION

5. METHODOLOGY

5.1 Data Loading
5.2 Data Cleaning Techniques
5.3 Data Transformation Techniques

6. IMPLEMENTATION/OUTPUT

6.1 Basic Functions (Load Data)
6.2 Handling Missing Values
6.3 Handling Duplicate Records
6.4 Detecting & Deleting Outliers
6.5 Handling Inconsistent Data
6.6 Data Type Conversion
6.7 Scaling Data (Standardization)
6.8 Normalization (Min-Max)
6.9 Binning
6.10 EDA (Exploratory Data Analysis)

7. FEATURE ENGINEERING

7.1 One Hot Encoding
7.2 Dummy Variable Creation
7.3 Label Encoding
7.4 Feature Extraction
7.5 Feature Scaling
7.6 Dimensionality Reduction (PCA)

8. TEXT PROCESSING

8.1 Lower Casing
8.2 Removing Punctuation
8.3 Tokenization

9. CONCLUSION

INTRODUCTION

In today's digital era, online shopping platforms generate a huge amount of data related to customer purchases, product details, transactions, and user behavior. Analyzing this data is very important for understanding customer patterns, improving business decisions, and enhancing user experience.

The **Online Shopping Data Science Project** focuses on analyzing an online shopping dataset using Python. This project demonstrates the practical implementation of **data cleaning, data transformation, exploratory data analysis (EDA), feature engineering, dimensionality reduction, and text processing** techniques as per the Data Science syllabus.

The project helps in converting raw data into meaningful insights using various data science tools and libraries.

OBJECTIVES OF THE PROJECT

The main objectives of this project are:

- To load and understand an online shopping dataset
- To clean the dataset by handling missing values, duplicates, and outliers
- To transform data using scaling, normalization, and binning
- To perform Exploratory Data Analysis (EDA)
- To apply feature engineering techniques
- To perform dimensionality reduction using PCA
- To process text data such as product descriptions
- To gain hands-on experience with real-world data science workflows

TOOLS AND TECHNOLOGIES USED

The following tools and libraries are used in this project:

- **Programming Language:** Python
- **Libraries Used:**
    - Pandas – Data manipulation and analysis
    - NumPy – Numerical operations
    - Matplotlib – Data visualization
    - Scikit-learn – Scaling, encoding, PCA
- **Platform:** GitHub (for project hosting and version control)

DATASET DESCRIPTION

The dataset used in this project is an **online shopping dataset** containing information such as:

- Customer-related data
- Transaction details
- Product information
- Numerical and categorical attributes
- Product descriptions (text data)

The dataset is stored in **CSV format** and loaded using the Pandas library.

Linke:-https://www.kaggle.com/datasets/jacksondivakarr/online-shopping-dataset

METHODOLOGY

Data Loading

The dataset is loaded using `pandas.read_csv()` and basic inspection is performed using:

- `head()`
- `info()`
- `describe()`
- `sample()`

Data Cleaning Techniques

The following data cleaning steps are applied:

a) Handling Missing Values

- Numerical columns: Filled using **mean**
- Alternative approaches shown: **median** and **dropna**

b) Removing Duplicate Records

- Duplicate rows are identified and removed using `drop_duplicates()`

c) Outlier Detection

- Outliers are detected and removed using the **Interquartile Range (IQR) method**

d) Handling Inconsistent Data

- Categorical data is converted to lowercase and trimmed to remove inconsistencies

Data Transformation Techniques

a) Data Type Conversion

- Date columns are converted to datetime format

b) Data Scaling

- **StandardScaler** is used for standardization

c) Normalization

- **MinMaxScaler** is used for normalization

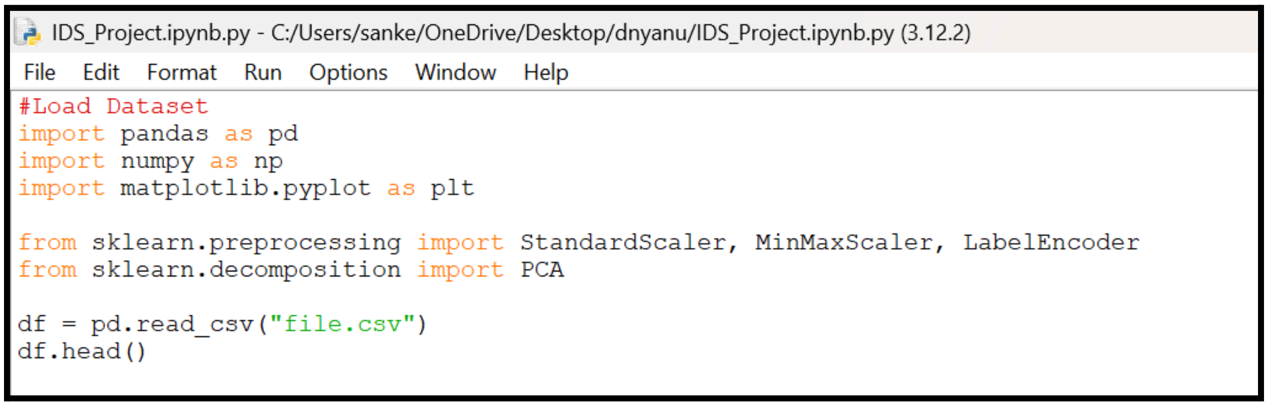
d) Binning

- Numerical data (e.g., tenure) is grouped into categories such as Low, Medium, and High

Implementation/OUTPUT

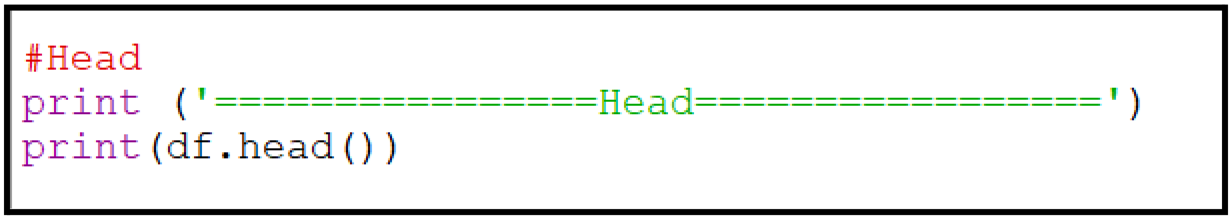
- **BASIC FUNCTIONS**

LOAD DATA

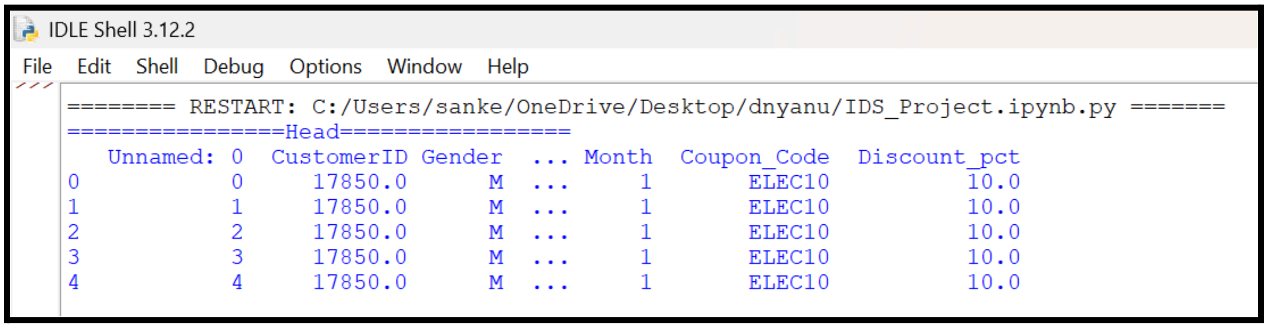

IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py (3.12.2)
File Edit Format Run Options Window Help

```
#Load Dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler, MinMaxScaler, LabelEncoder
from sklearn.decomposition import PCA

df = pd.read_csv("file.csv")
df.head()
```

Head

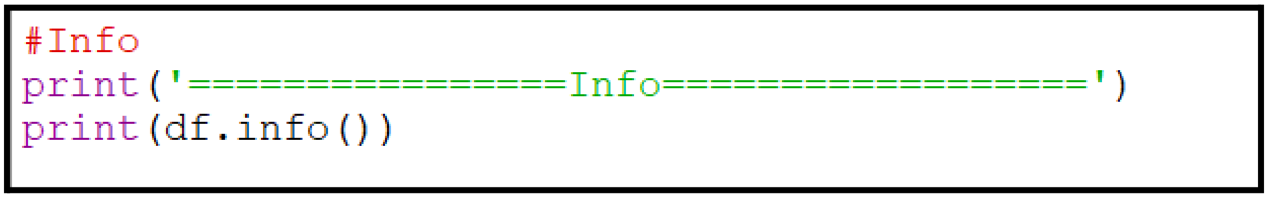


```
#Head
print ('================Head=================')
print(df.head())
```

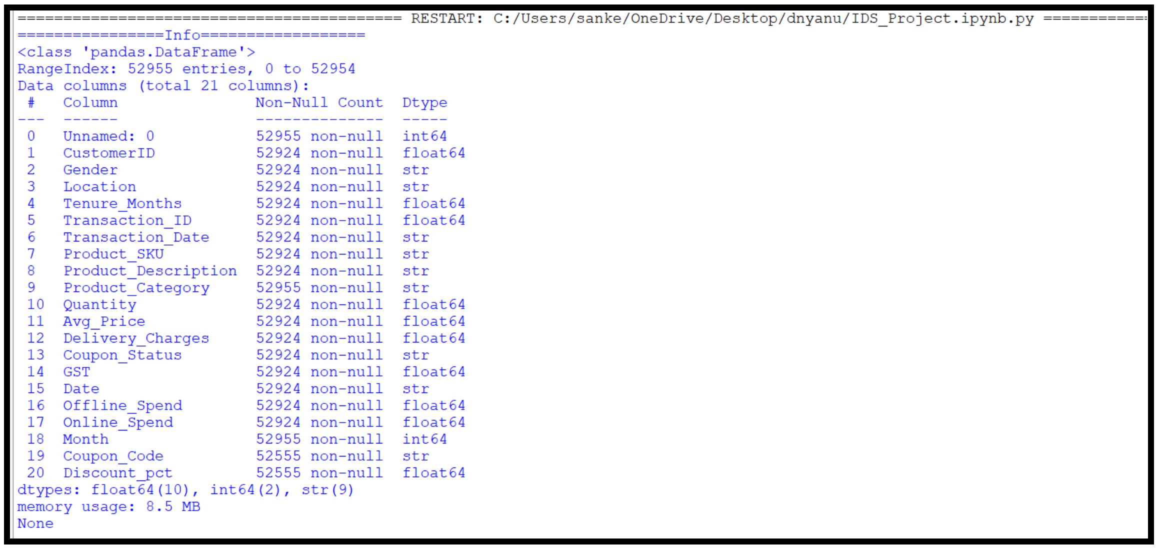


```
======== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =======
================Head=================
   Unnamed: 0  CustomerID Gender  ... Month  Coupon_Code  Discount_pct
0           0     17850.0      M  ...     1       ELEC10          10.0
1           1     17850.0      M  ...     1       ELEC10          10.0
2           2     17850.0      M  ...     1       ELEC10          10.0
3           3     17850.0      M  ...     1       ELEC10          10.0
4           4     17850.0      M  ...     1       ELEC10          10.0
```

Info

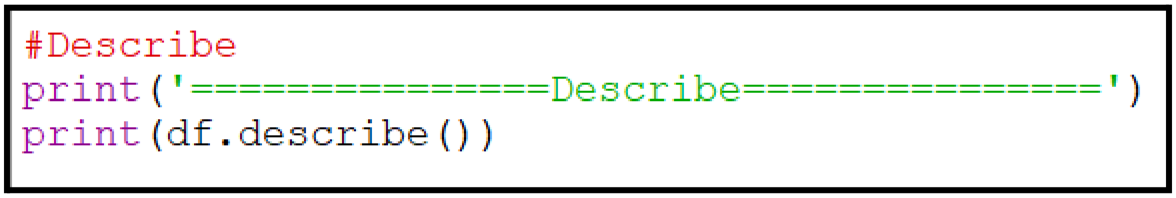


```
#Info
print('================Info==================')
print(df.info())
```

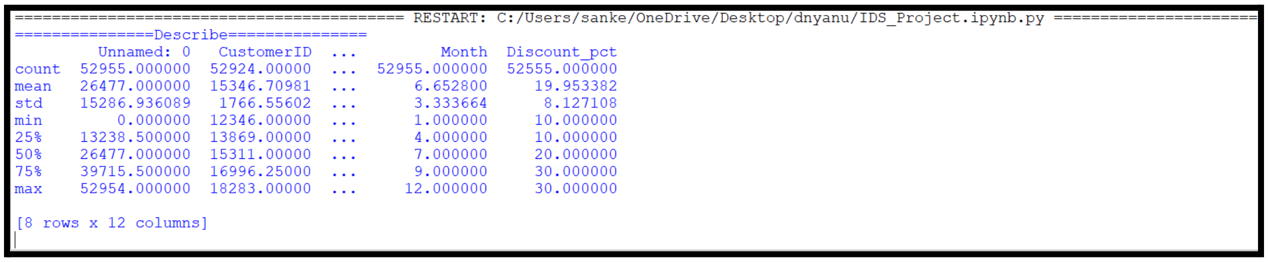


```
========================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
================Info==================
<class 'pandas.DataFrame'>
RangeIndex: 52955 entries, 0 to 52954
Data columns (total 21 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Unnamed: 0           52955 non-null  int64
 1   CustomerID           52924 non-null  float64
 2   Gender               52924 non-null  str
 3   Location             52924 non-null  str
 4   Tenure_Months        52924 non-null  float64
 5   Transaction_ID       52924 non-null  float64
 6   Transaction_Date     52924 non-null  str
 7   Product_SKU          52924 non-null  str
 8   Product_Description  52924 non-null  str
 9   Product_Category     52955 non-null  str
 10  Quantity             52924 non-null  float64
 11  Avg_Price            52924 non-null  float64
 12  Delivery_Charges     52924 non-null  float64
 13  Coupon_Status        52924 non-null  str
 14  GST                  52924 non-null  float64
 15  Date                 52924 non-null  str
 16  Offline_Spend        52924 non-null  float64
 17  Online_Spend         52924 non-null  float64
 18  Month                52955 non-null  int64
 19  Coupon_Code          52555 non-null  str
 20  Discount_pct         52555 non-null  float64
dtypes: float64(10), int64(2), str(9)
memory usage: 8.5 MB
None
```

Describe

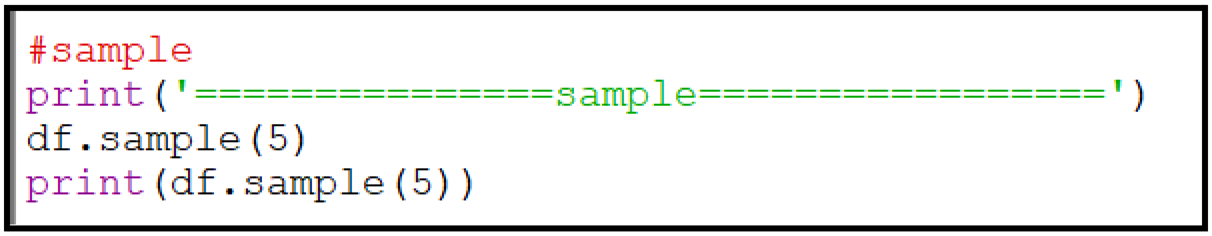


```
#Describe
print('================Describe================')
print(df.describe())
```

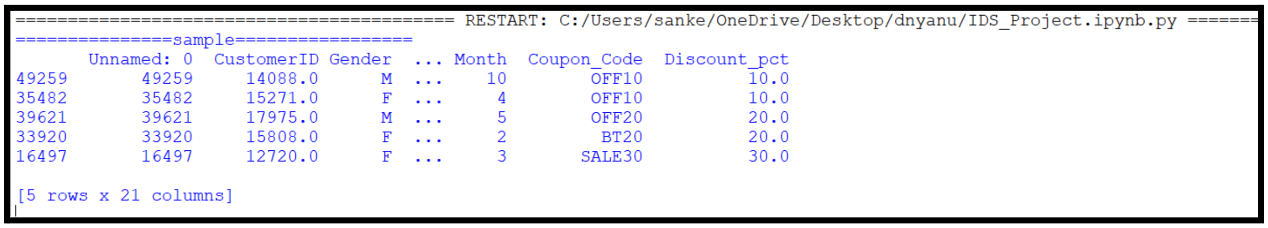


```
========================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====================
===============Describe===============
         Unnamed: 0   CustomerID  ...         Month  Discount_pct
count  52955.000000  52924.00000  ...  52955.000000  52555.000000
mean   26477.000000  15346.70981  ...      6.652800     19.953382
std    15286.936089   1766.55602  ...      3.333664      8.127108
min        0.000000  12346.00000  ...      1.000000     10.000000
25%    13238.500000  13869.00000  ...      4.000000     10.000000
50%    26477.000000  15311.00000  ...      7.000000     20.000000
75%    39715.500000  16996.25000  ...      9.000000     30.000000
max    52954.000000  18283.00000  ...     12.000000     30.000000

[8 rows x 12 columns]
```

sample



```
#sample
print('================sample==================')
df.sample(5)
print(df.sample(5))
```

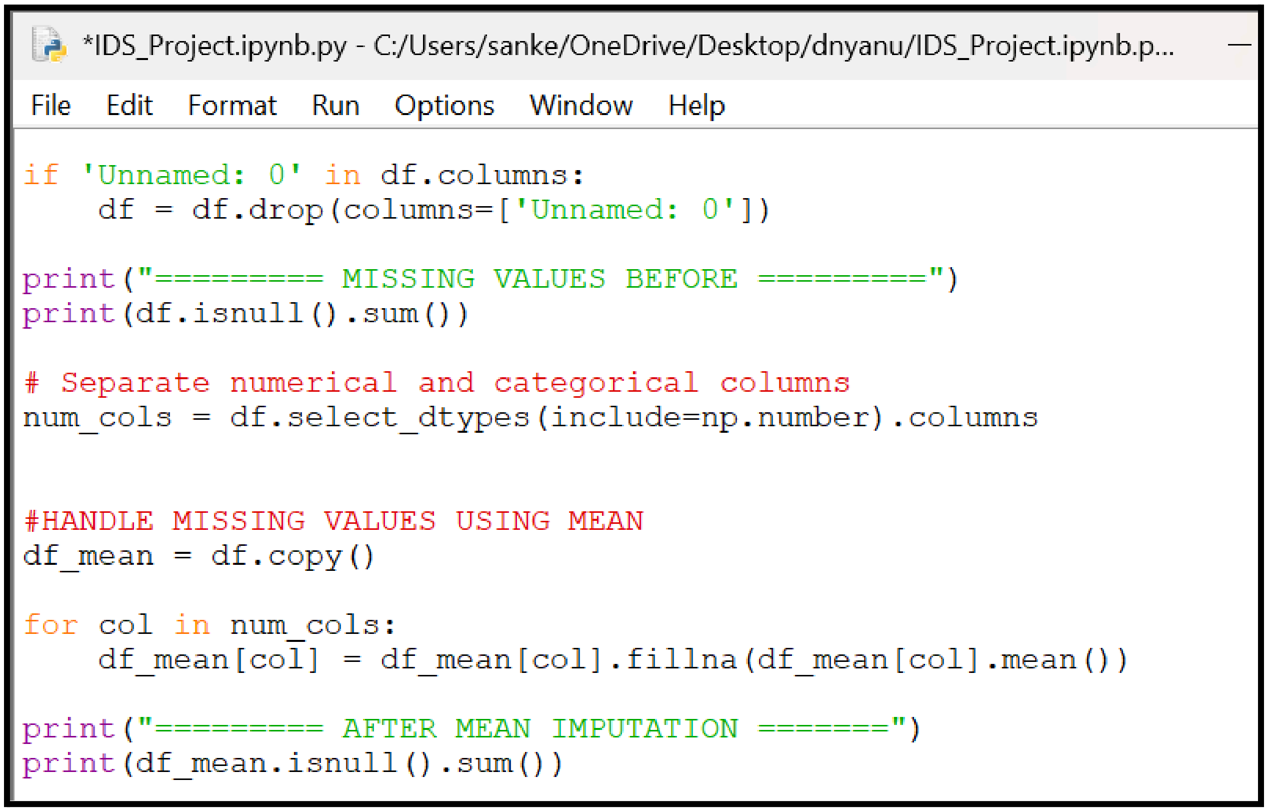


```
========================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =======
================sample=================
       Unnamed: 0  CustomerID Gender  ... Month  Coupon_Code  Discount_pct
49259       49259     14088.0      M  ...    10        OFF10          10.0
35482       35482     15271.0      F  ...     4        OFF10          10.0
39621       39621     17975.0      M  ...     5        OFF20          20.0
33920       33920     15808.0      F  ...     2         BT20          20.0
16497       16497     12720.0      F  ...     3       SALE30          30.0

[5 rows x 21 columns]
```

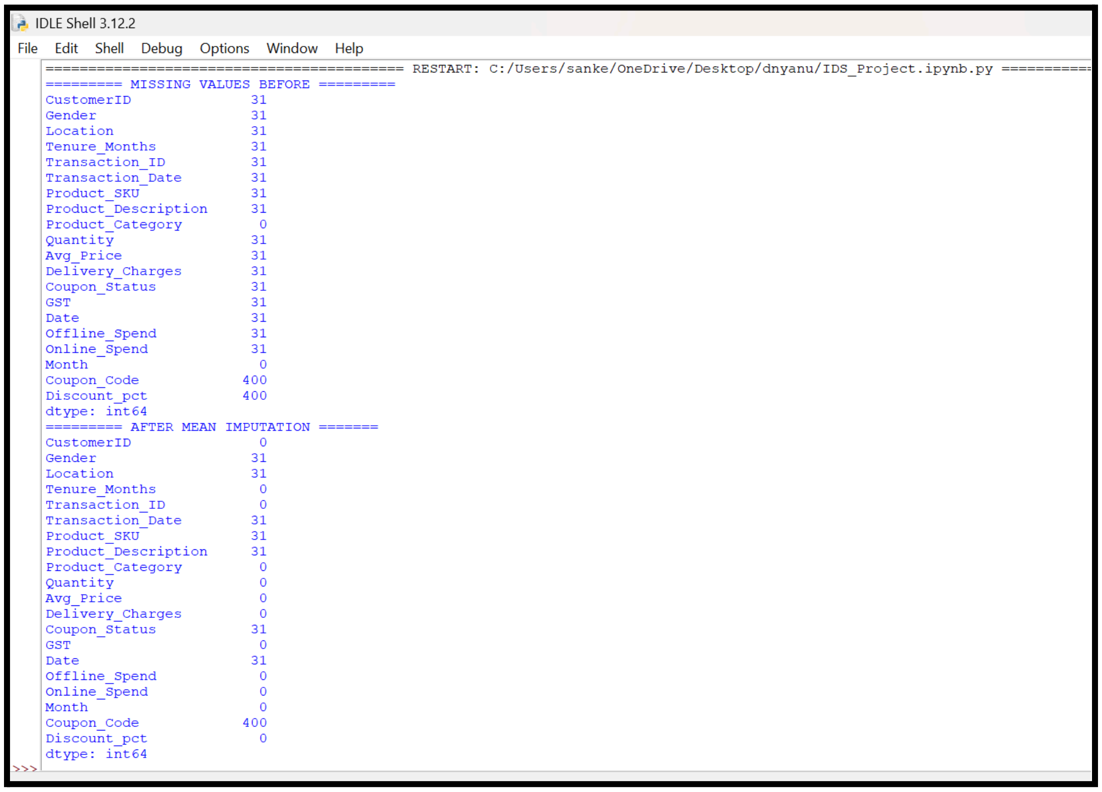
A)DATA CLEANING TECHNIQUES

1)HANDLING MISSING VALUES (MEAN, MEDIAN, DROP)

- **MEAN**

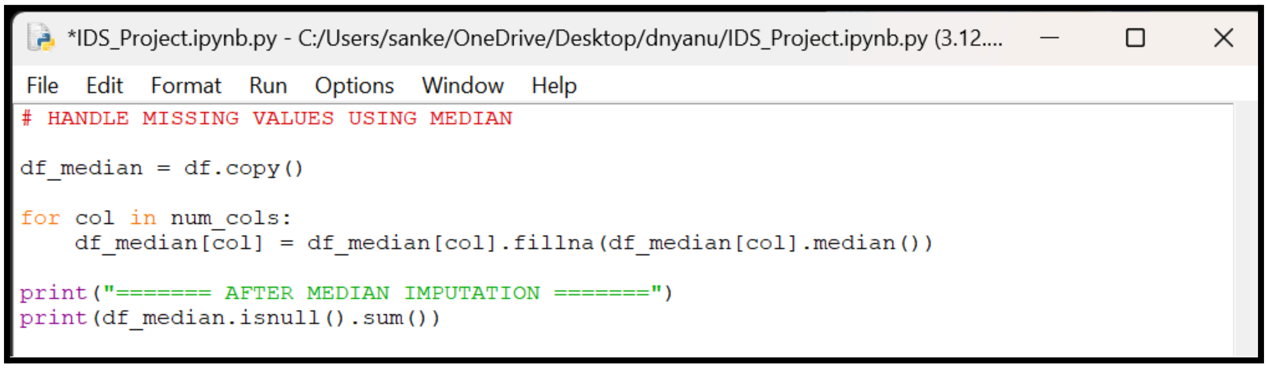

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
if 'Unnamed: 0' in df.columns:
    df = df.drop(columns=['Unnamed: 0'])

print("========= MISSING VALUES BEFORE =========")
print(df.isnull().sum())

# Separate numerical and categorical columns
num_cols = df.select_dtypes(include=np.number).columns


#HANDLE MISSING VALUES USING MEAN
df_mean = df.copy()

for col in num_cols:
    df_mean[col] = df_mean[col].fillna(df_mean[col].mean())

print("========= AFTER MEAN IMPUTATION =======")
print(df_mean.isnull().sum())
```



```
========================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ==========
========= MISSING VALUES BEFORE =========
CustomerID              31
Gender                  31
Location                31
Tenure_Months           31
Transaction_ID          31
Transaction_Date        31
Product_SKU             31
Product_Description     31
Product_Category         0
Quantity                31
Avg_Price               31
Delivery_Charges        31
Coupon_Status           31
GST                     31
Date                    31
Offline_Spend           31
Online_Spend            31
Month                    0
Coupon_Code            400
Discount_pct           400
dtype: int64
========= AFTER MEAN IMPUTATION =======
CustomerID               0
Gender                  31
Location                31
Tenure_Months            0
Transaction_ID           0
Transaction_Date        31
Product_SKU             31
Product_Description     31
Product_Category         0
Quantity                 0
Avg_Price                0
Delivery_Charges         0
Coupon_Status           31
GST                      0
Date                    31
Offline_Spend            0
Online_Spend             0
Month                    0
Coupon_Code            400
Discount_pct             0
dtype: int64
>>>
```

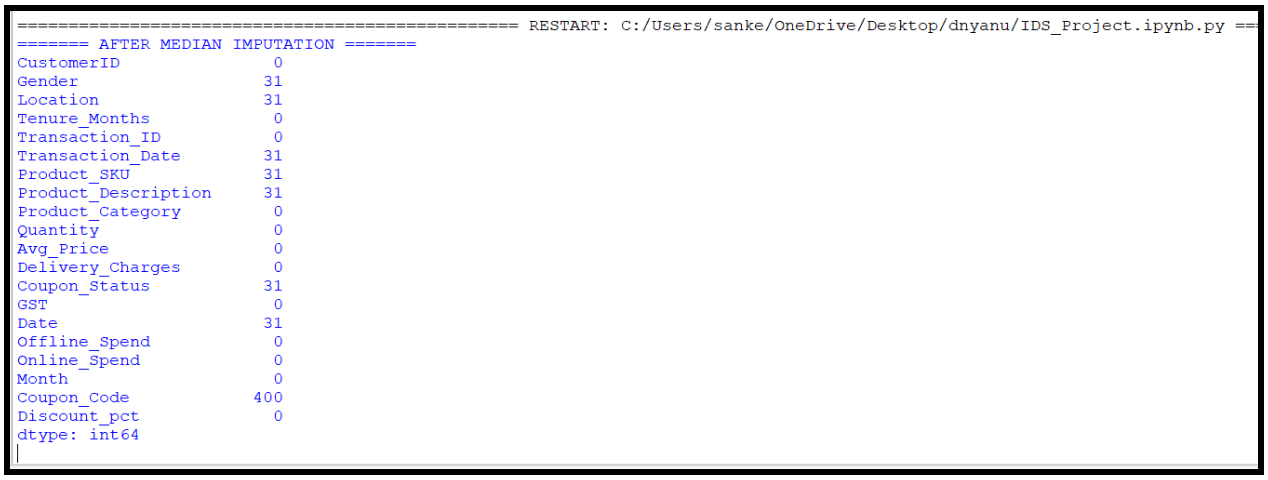
- **MEDIAN**



*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py (3.12....

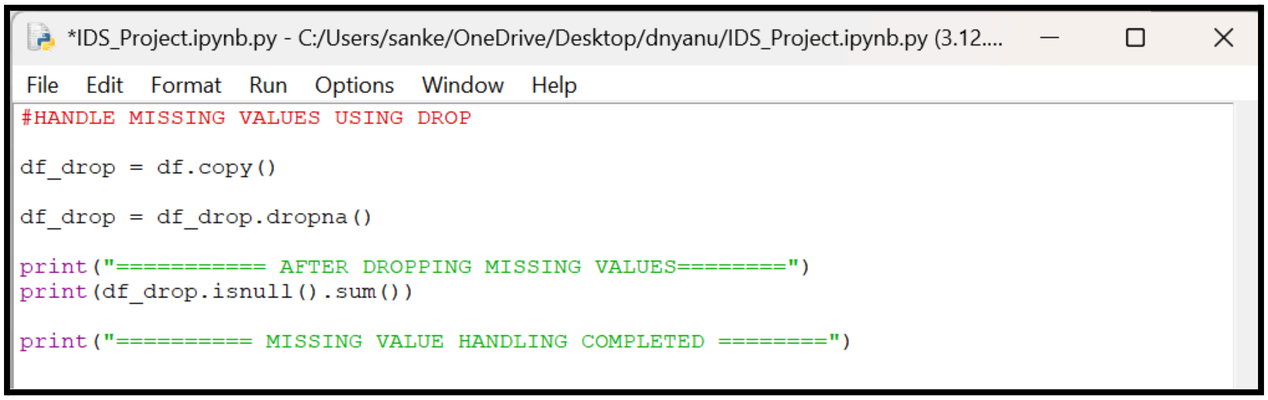
File Edit Format Run Options Window Help

```
# HANDLE MISSING VALUES USING MEDIAN

df_median = df.copy()

for col in num_cols:
    df_median[col] = df_median[col].fillna(df_median[col].median())

print("======= AFTER MEDIAN IMPUTATION =======")
print(df_median.isnull().sum())
```



```
================================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ==
======= AFTER MEDIAN IMPUTATION =======
CustomerID               0
Gender                  31
Location                31
Tenure_Months            0
Transaction_ID           0
Transaction_Date        31
Product_SKU             31
Product_Description     31
Product_Category         0
Quantity                 0
Avg_Price                0
Delivery_Charges         0
Coupon_Status           31
GST                      0
Date                    31
Offline_Spend            0
Online_Spend             0
Month                    0
Coupon_Code            400
Discount_pct             0
dtype: int64
```

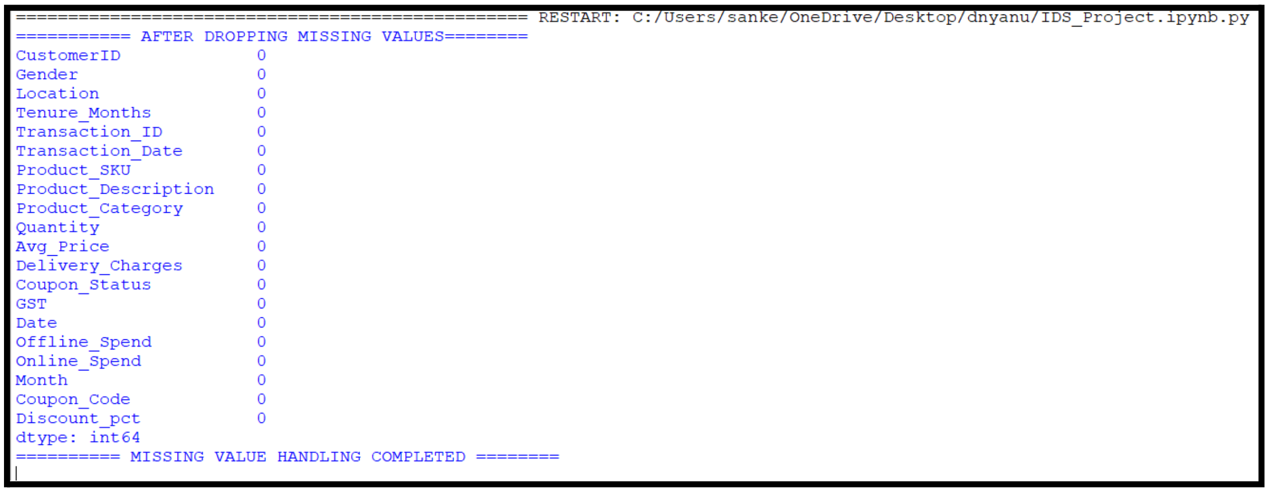
- **DROP**



*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py (3.12....

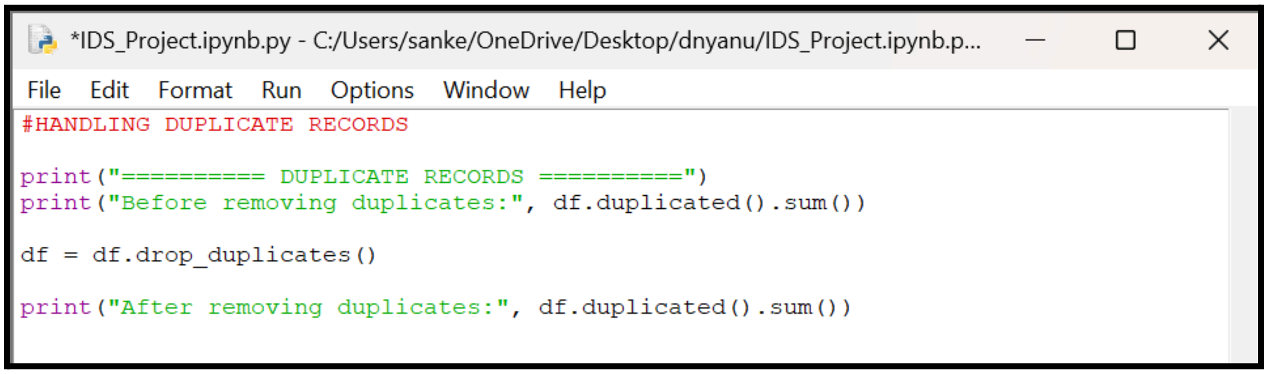
File Edit Format Run Options Window Help

```
#HANDLE MISSING VALUES USING DROP

df_drop = df.copy()

df_drop = df_drop.dropna()

print("=========== AFTER DROPPING MISSING VALUES========")
print(df_drop.isnull().sum())

print("========== MISSING VALUE HANDLING COMPLETED ========")
```

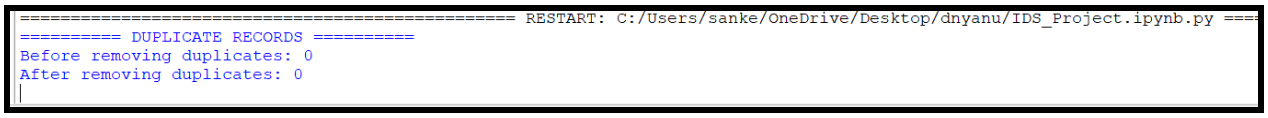


```
================================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py
=========== AFTER DROPPING MISSING VALUES========
CustomerID             0
Gender                 0
Location               0
Tenure_Months          0
Transaction_ID         0
Transaction_Date       0
Product_SKU            0
Product_Description    0
Product_Category       0
Quantity               0
Avg_Price              0
Delivery_Charges       0
Coupon_Status          0
GST                    0
Date                   0
Offline_Spend          0
Online_Spend           0
Month                  0
Coupon_Code            0
Discount_pct           0
dtype: int64
========== MISSING VALUE HANDLING COMPLETED ========
```

2)HANDLING DUPLICATE RECORDS

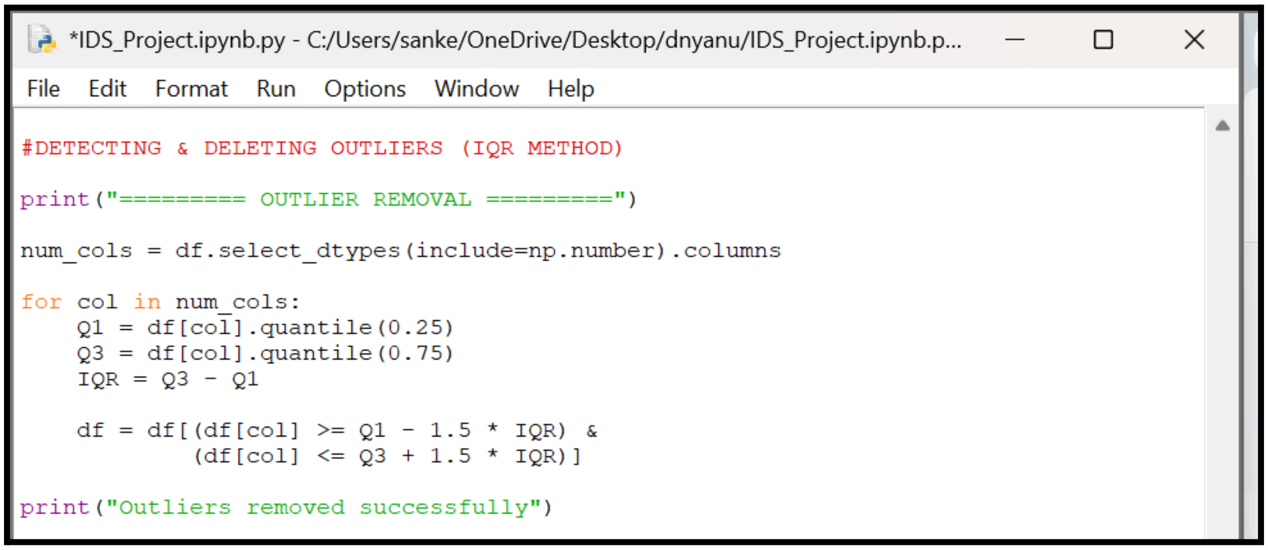

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
#HANDLING DUPLICATE RECORDS

print("========== DUPLICATE RECORDS ==========")
print("Before removing duplicates:", df.duplicated().sum())

df = df.drop_duplicates()

print("After removing duplicates:", df.duplicated().sum())
```

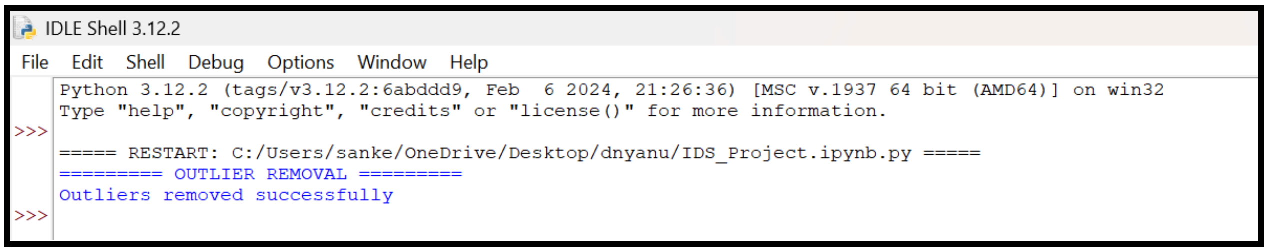


```
================================================= RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ====
========== DUPLICATE RECORDS ==========
Before removing duplicates: 0
After removing duplicates: 0
```

3)DETECTING & DELETING OUTLIERS (IQR METHOD)

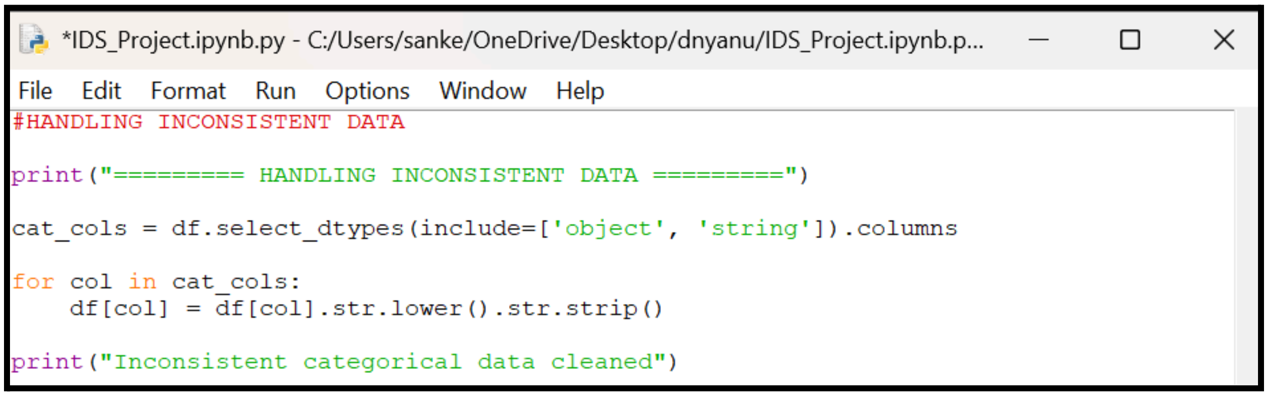

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
#DETECTING & DELETING OUTLIERS (IQR METHOD)

print("========= OUTLIER REMOVAL =========")

num_cols = df.select_dtypes(include=np.number).columns

for col in num_cols:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1

    df = df[(df[col] >= Q1 - 1.5 * IQR) &
            (df[col] <= Q3 + 1.5 * IQR)]

print("Outliers removed successfully")
```

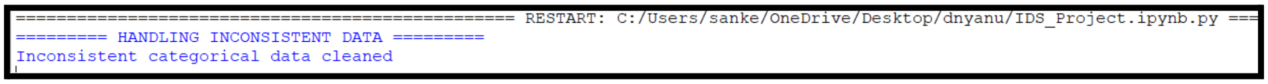


```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= OUTLIER REMOVAL =========
Outliers removed successfully
>>>
```

4)HANDLING INCONSISTENT DATA


*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

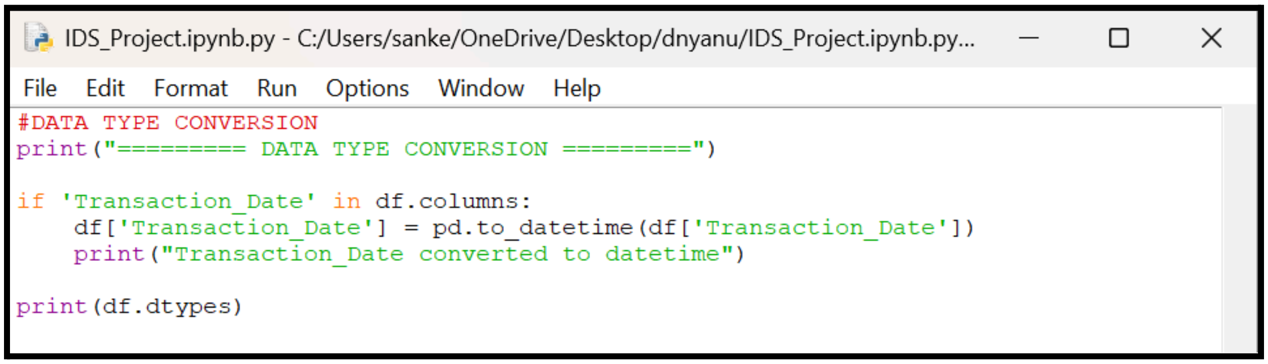
```
#HANDLING INCONSISTENT DATA

print("========= HANDLING INCONSISTENT DATA =========")

cat_cols = df.select_dtypes(include=['object', 'string']).columns

for col in cat_cols:
    df[col] = df[col].str.lower().str.strip()

print("Inconsistent categorical data cleaned")
```



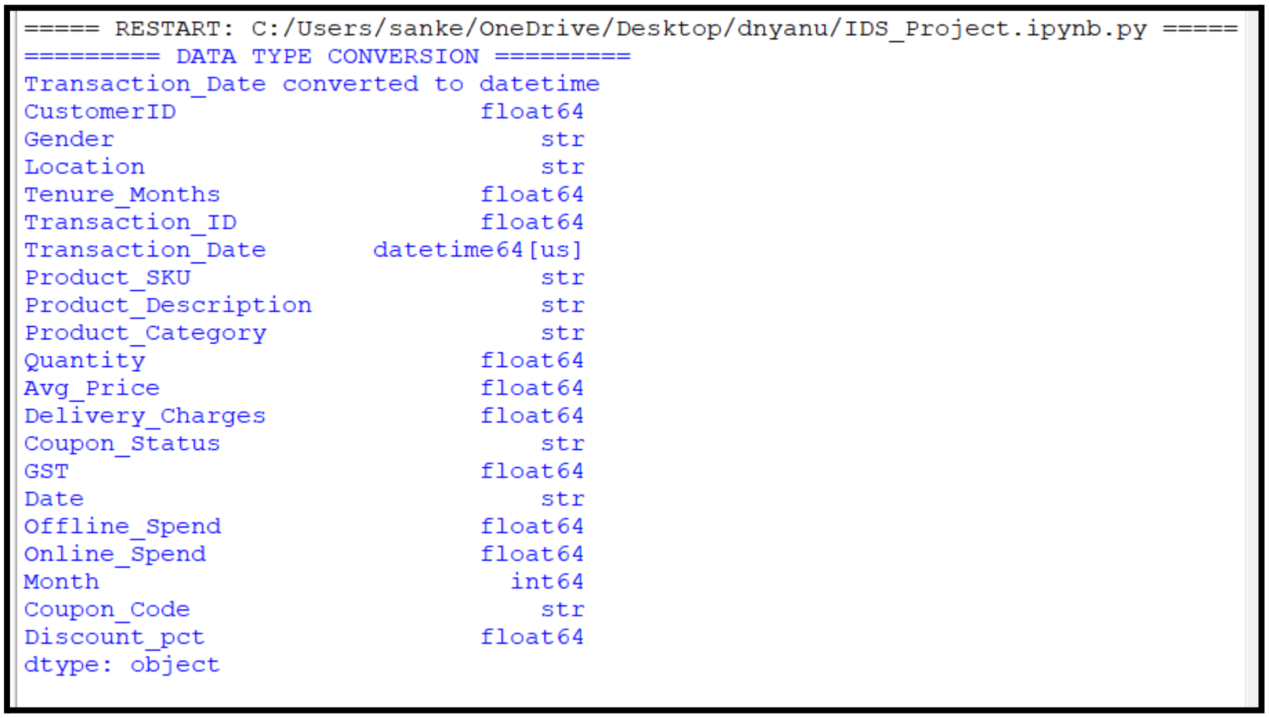
```
================================================== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===
========= HANDLING INCONSISTENT DATA =========
Inconsistent categorical data cleaned
```

B)DATA TRANSFORMATION TECHNIQUES

5)DATA TYPE CONVERSION

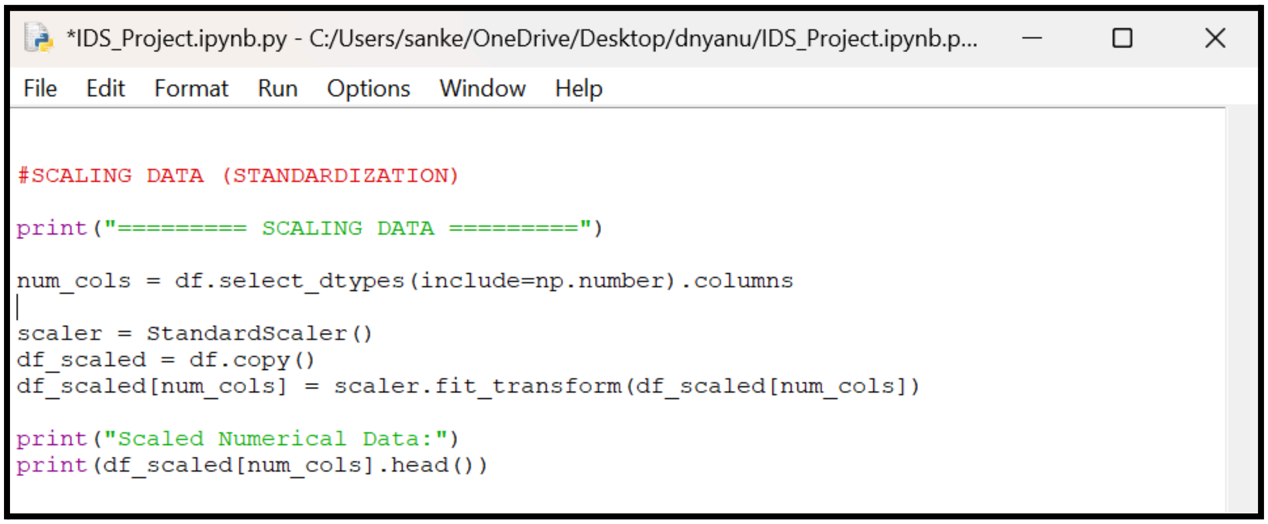

IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py...
File Edit Format Run Options Window Help

```
#DATA TYPE CONVERSION
print("========= DATA TYPE CONVERSION =========")

if 'Transaction_Date' in df.columns:
    df['Transaction_Date'] = pd.to_datetime(df['Transaction_Date'])
    print("Transaction_Date converted to datetime")

print(df.dtypes)
```

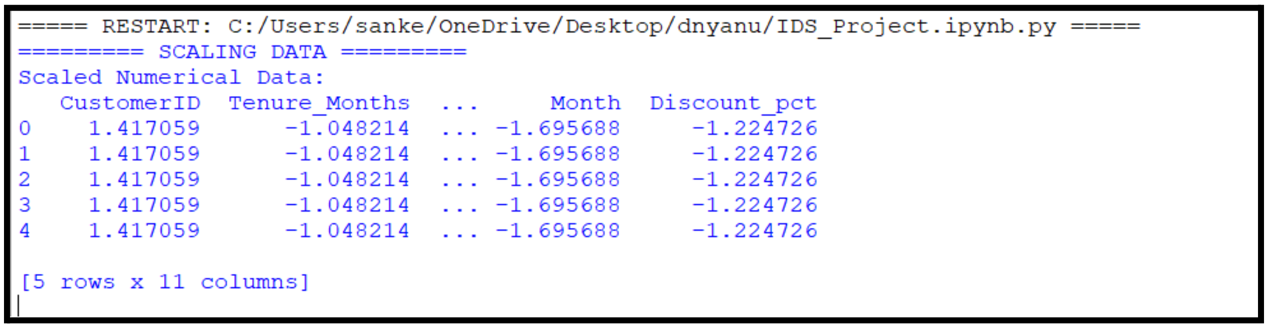


```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= DATA TYPE CONVERSION =========
Transaction_Date converted to datetime
CustomerID                   float64
Gender                           str
Location                         str
Tenure_Months                float64
Transaction_ID               float64
Transaction_Date      datetime64[us]
Product_SKU                      str
Product_Description              str
Product_Category                 str
Quantity                     float64
Avg_Price                    float64
Delivery_Charges             float64
Coupon_Status                    str
GST                          float64
Date                             str
Offline_Spend                float64
Online_Spend                 float64
Month                          int64
Coupon_Code                      str
Discount_pct                 float64
dtype: object
```

6)SCALING DATA (STANDARDIZATION)

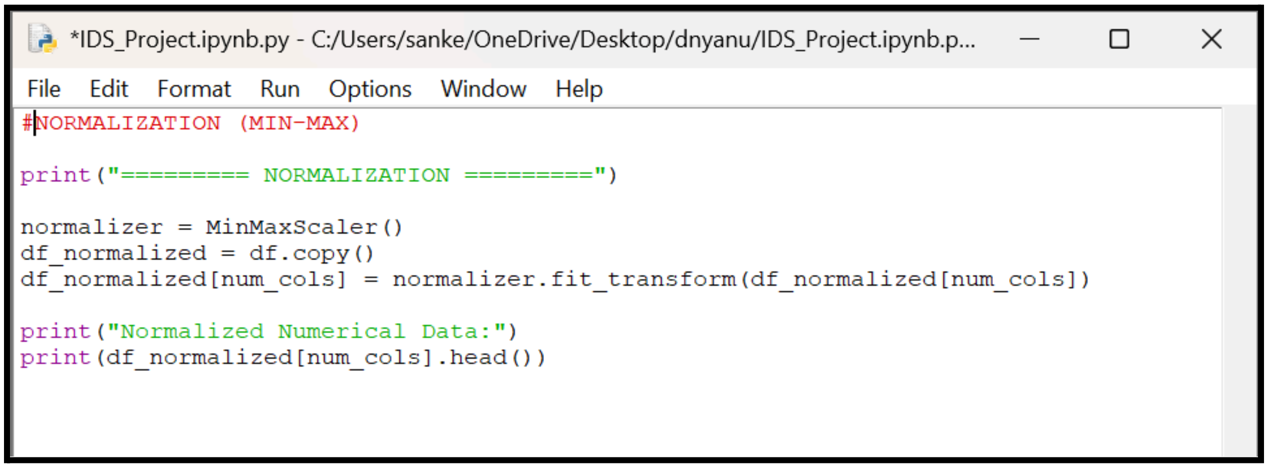

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
#SCALING DATA (STANDARDIZATION)

print("========= SCALING DATA =========")

num_cols = df.select_dtypes(include=np.number).columns

scaler = StandardScaler()
df_scaled = df.copy()
df_scaled[num_cols] = scaler.fit_transform(df_scaled[num_cols])

print("Scaled Numerical Data:")
print(df_scaled[num_cols].head())
```

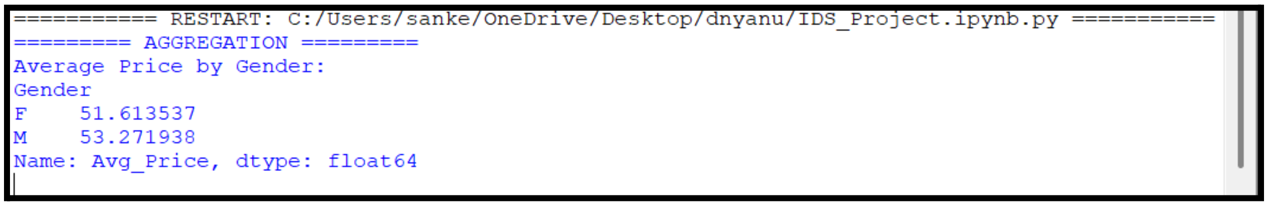


```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= SCALING DATA =========
Scaled Numerical Data:
   CustomerID  Tenure_Months  ...     Month  Discount_pct
0    1.417059      -1.048214  ... -1.695688     -1.224726
1    1.417059      -1.048214  ... -1.695688     -1.224726
2    1.417059      -1.048214  ... -1.695688     -1.224726
3    1.417059      -1.048214  ... -1.695688     -1.224726
4    1.417059      -1.048214  ... -1.695688     -1.224726

[5 rows x 11 columns]
```

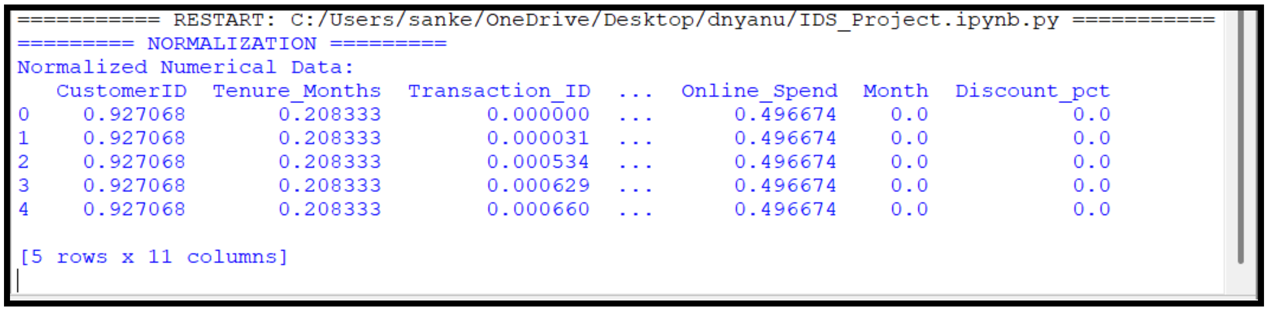
7)NORMALIZATION (MIN-MAX)


*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
#NORMALIZATION (MIN-MAX)

print("========= NORMALIZATION =========")

normalizer = MinMaxScaler()
df_normalized = df.copy()
df_normalized[num_cols] = normalizer.fit_transform(df_normalized[num_cols])

print("Normalized Numerical Data:")
print(df_normalized[num_cols].head())
```

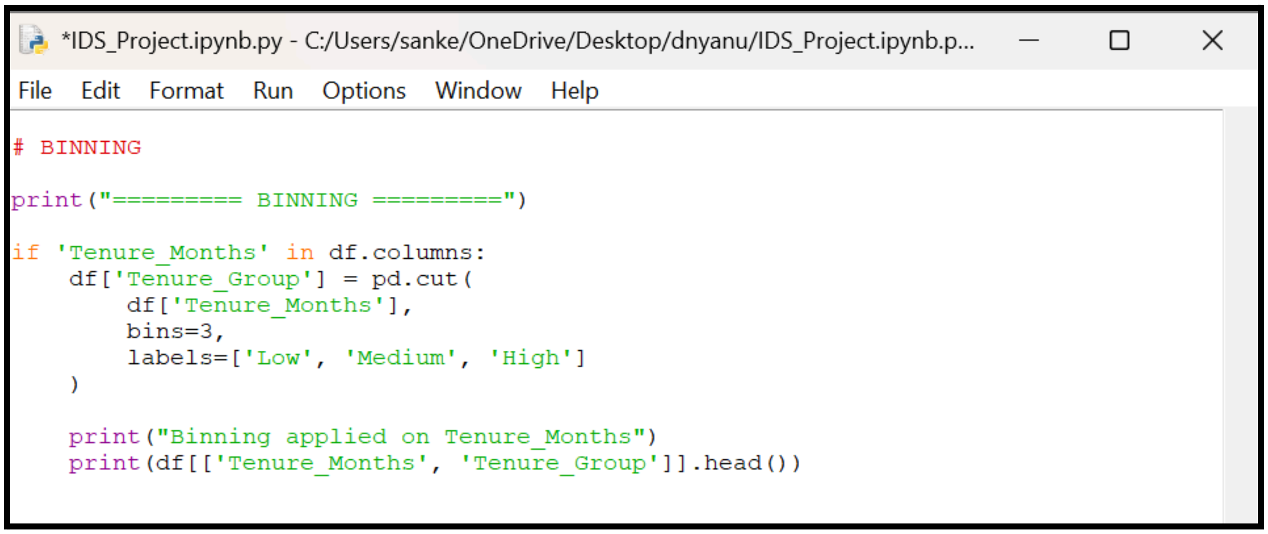


```
=========== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
========= AGGREGATION =========
Average Price by Gender:
Gender
F    51.613537
M    53.271938
Name: Avg_Price, dtype: float64
```



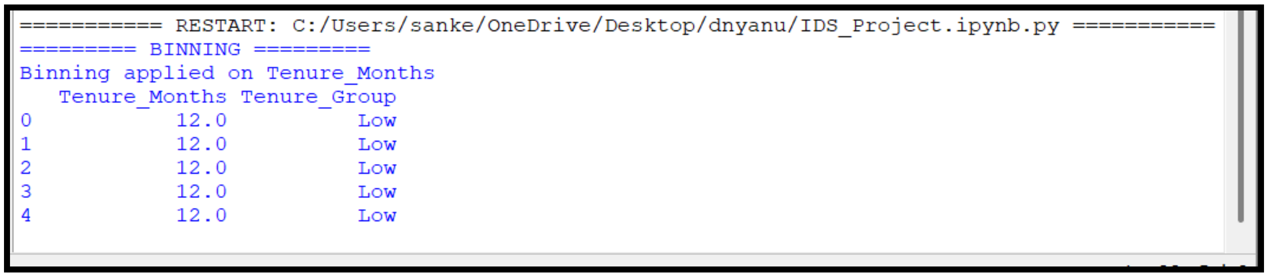
```
=========== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
========= NORMALIZATION =========
Normalized Numerical Data:
   CustomerID  Tenure_Months  Transaction_ID  ...  Online_Spend  Month  Discount_pct
0    0.927068       0.208333        0.000000  ...      0.496674    0.0           0.0
1    0.927068       0.208333        0.000031  ...      0.496674    0.0           0.0
2    0.927068       0.208333        0.000534  ...      0.496674    0.0           0.0
3    0.927068       0.208333        0.000629  ...      0.496674    0.0           0.0
4    0.927068       0.208333        0.000660  ...      0.496674    0.0           0.0

[5 rows x 11 columns]
```

8)BINNING



*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...

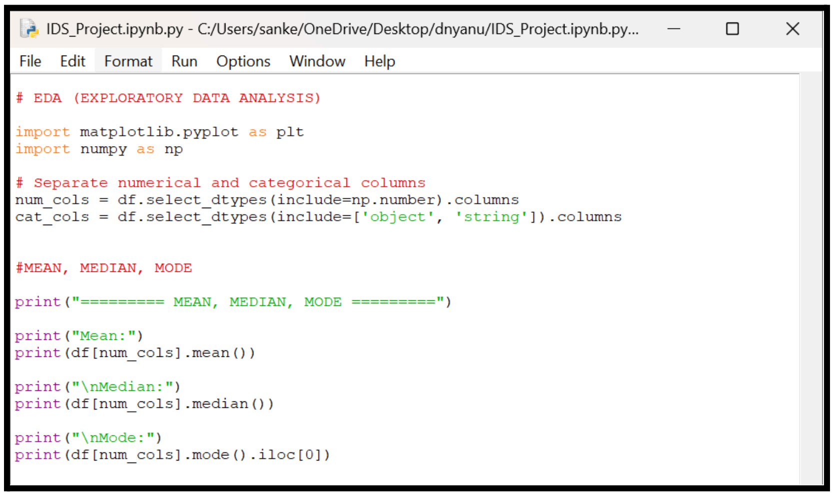
```python
# BINNING

print("========= BINNING =========")

if 'Tenure_Months' in df.columns:
    df['Tenure_Group'] = pd.cut(
        df['Tenure_Months'],
        bins=3,
        labels=['Low', 'Medium', 'High']
    )

    print("Binning applied on Tenure_Months")
    print(df[['Tenure_Months', 'Tenure_Group']].head())
```



```
=========== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
========= BINNING =========
Binning applied on Tenure_Months
   Tenure_Months Tenure_Group
0           12.0          Low
1           12.0          Low
2           12.0          Low
3           12.0          Low
4           12.0          Low
```

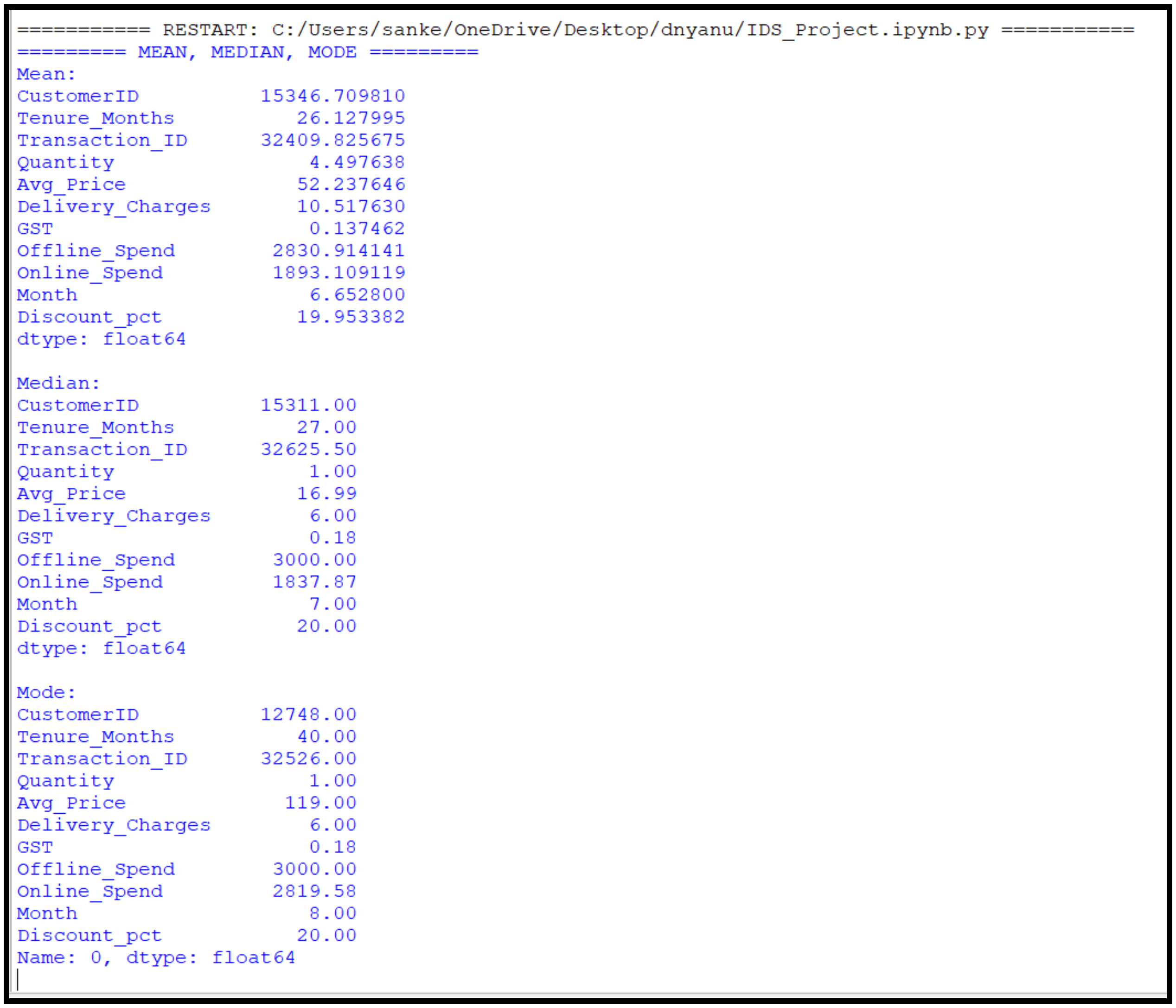
D)EDA (EXPLORATORY DATA ANALYSIS)

9)MEAN, MEDIAN, MODE



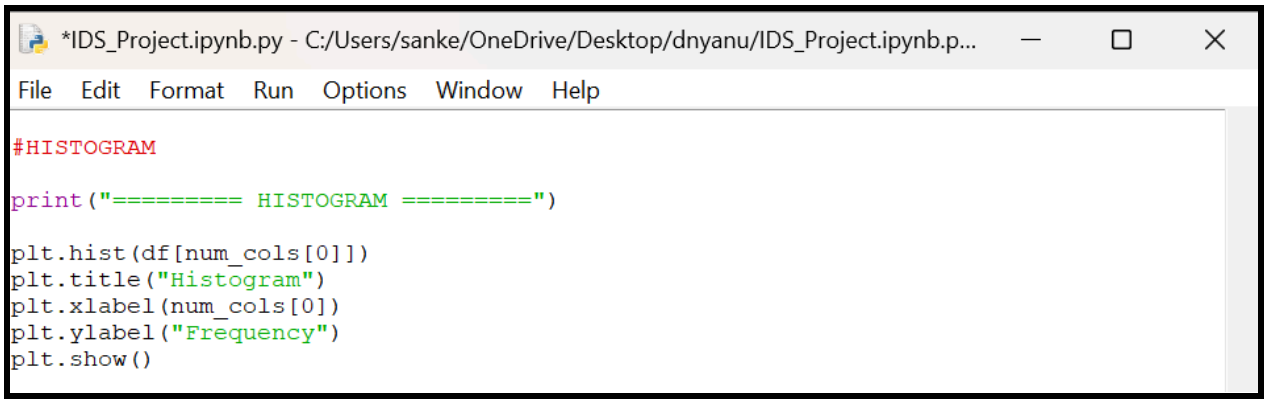
IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py...

```python
# EDA (EXPLORATORY DATA ANALYSIS)

import matplotlib.pyplot as plt
import numpy as np

# Separate numerical and categorical columns
num_cols = df.select_dtypes(include=np.number).columns
cat_cols = df.select_dtypes(include=['object', 'string']).columns


#MEAN, MEDIAN, MODE

print("========= MEAN, MEDIAN, MODE =========")

print("Mean:")
print(df[num_cols].mean())

print("\nMedian:")
print(df[num_cols].median())

print("\nMode:")
print(df[num_cols].mode().iloc[0])
```

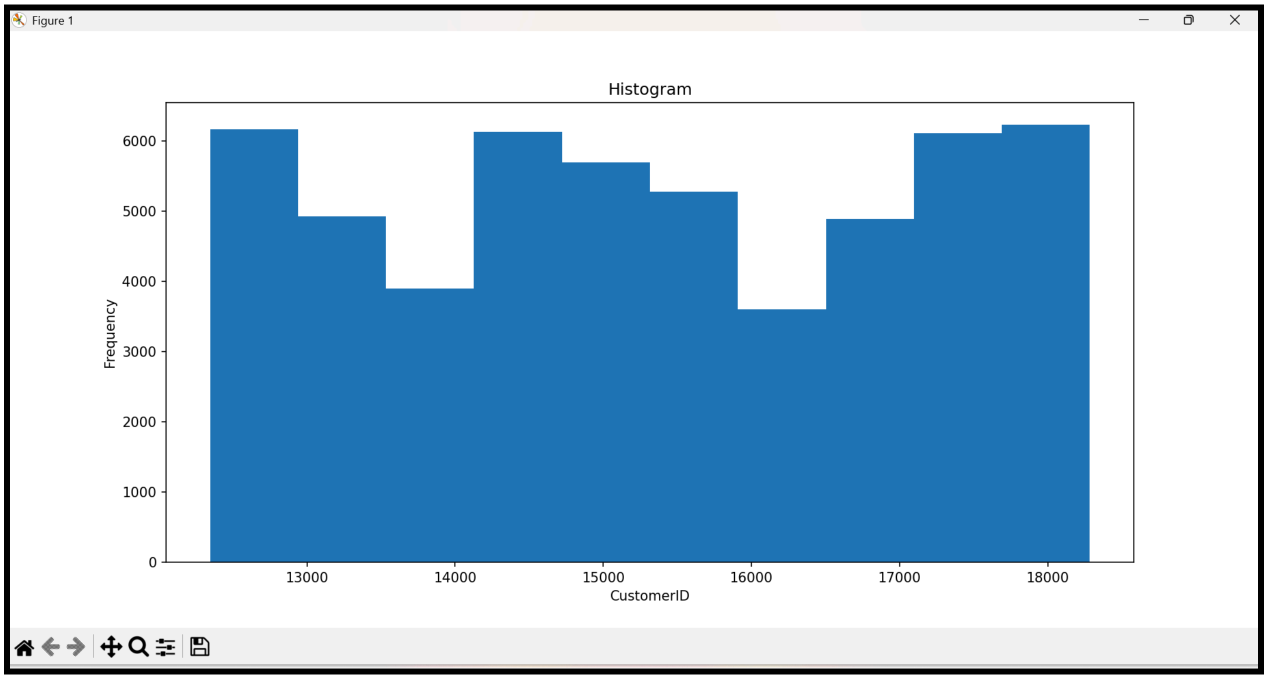


```
=========== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
========= MEAN, MEDIAN, MODE =========
Mean:
CustomerID           15346.709810
Tenure_Months           26.127995
Transaction_ID       32409.825675
Quantity                 4.497638
Avg_Price               52.237646
Delivery_Charges        10.517630
GST                      0.137462
Offline_Spend         2830.914141
Online_Spend          1893.109119
Month                    6.652800
Discount_pct            19.953382
dtype: float64

Median:
CustomerID           15311.00
Tenure_Months           27.00
Transaction_ID       32625.50
Quantity                 1.00
Avg_Price               16.99
Delivery_Charges         6.00
GST                      0.18
Offline_Spend         3000.00
Online_Spend          1837.87
Month                    7.00
Discount_pct            20.00
dtype: float64

Mode:
CustomerID           12748.00
Tenure_Months           40.00
Transaction_ID       32526.00
Quantity                 1.00
Avg_Price              119.00
Delivery_Charges         6.00
GST                      0.18
Offline_Spend         3000.00
Online_Spend          2819.58
Month                    8.00
Discount_pct            20.00
Name: 0, dtype: float64
```

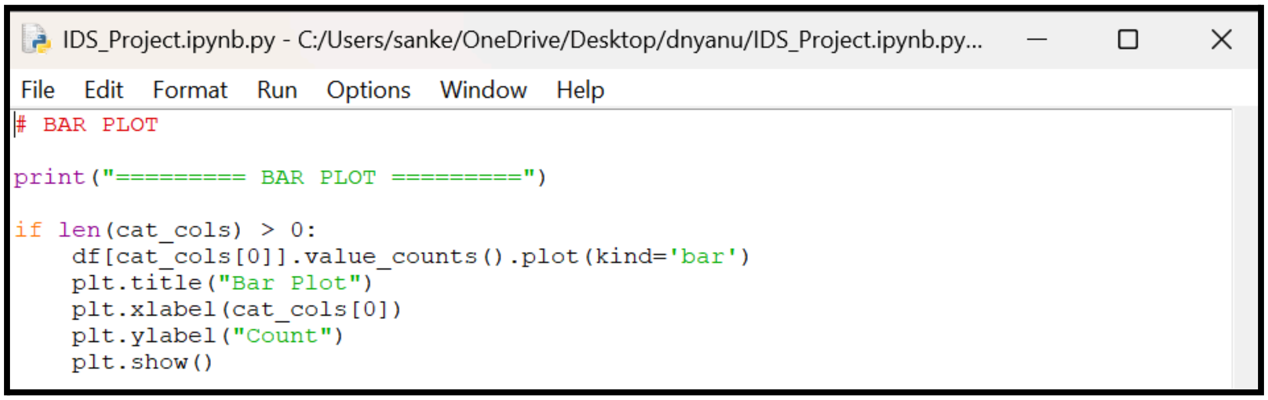
10)HISTOGRAM


*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

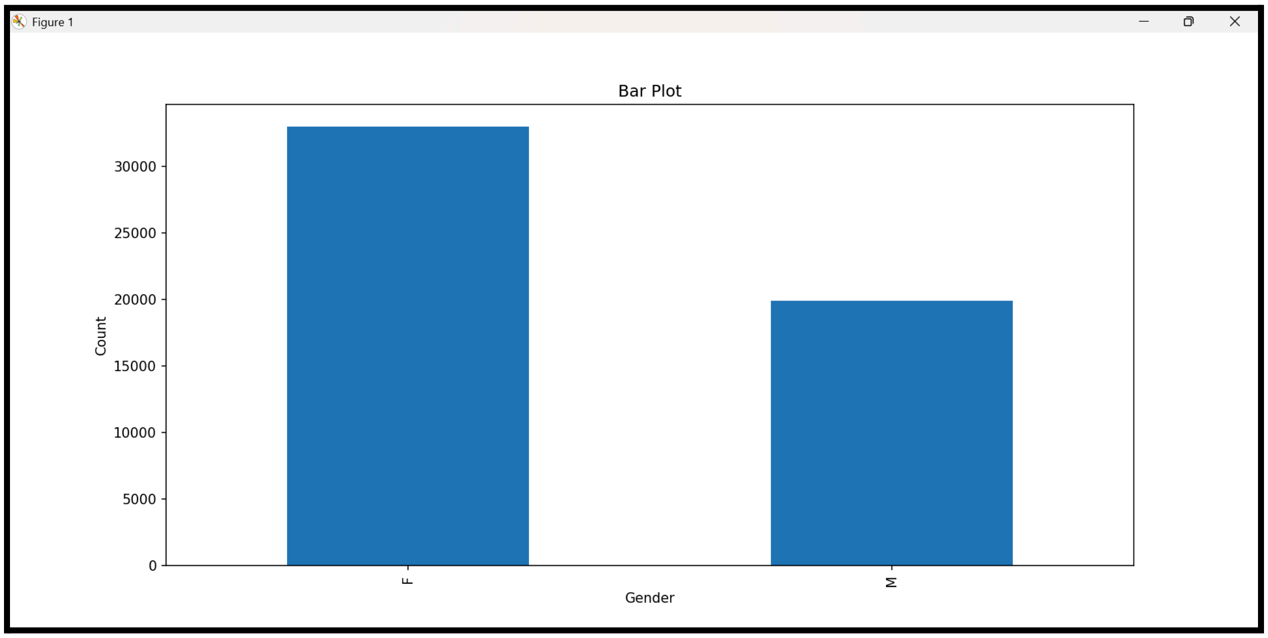
```
#HISTOGRAM

print("========= HISTOGRAM =========")

plt.hist(df[num_cols[0]])
plt.title("Histogram")
plt.xlabel(num_cols[0])
plt.ylabel("Frequency")
plt.show()
```



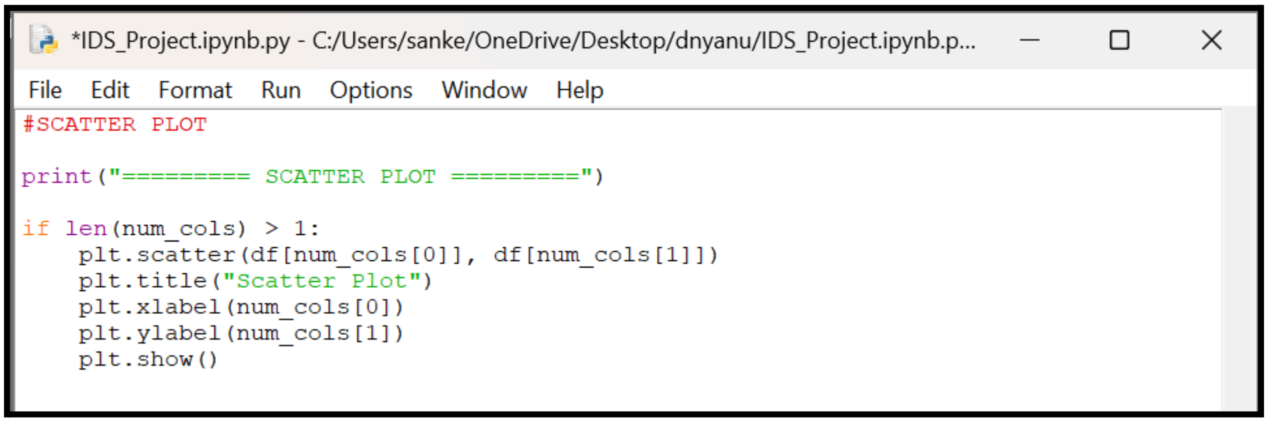
11)BAR PLOT


IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py...
File Edit Format Run Options Window Help

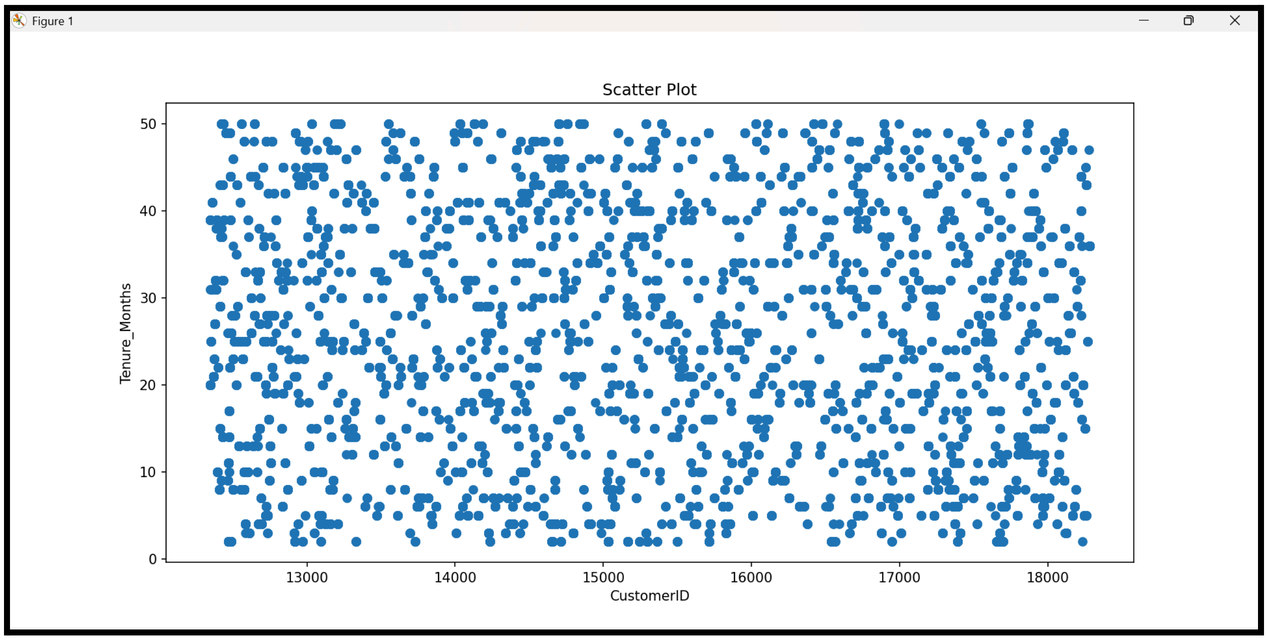
```
# BAR PLOT

print("========= BAR PLOT =========")

if len(cat_cols) > 0:
    df[cat_cols[0]].value_counts().plot(kind='bar')
    plt.title("Bar Plot")
    plt.xlabel(cat_cols[0])
    plt.ylabel("Count")
    plt.show()
```



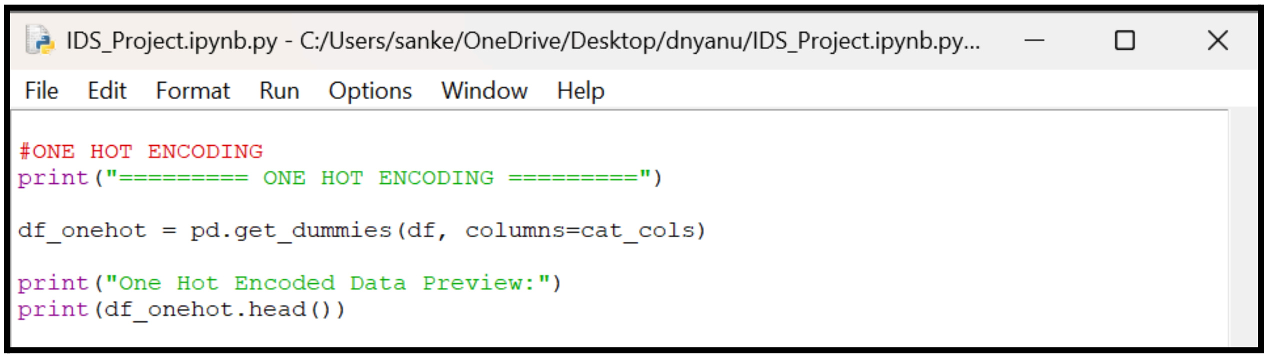
12)SCATTER PLOT


*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

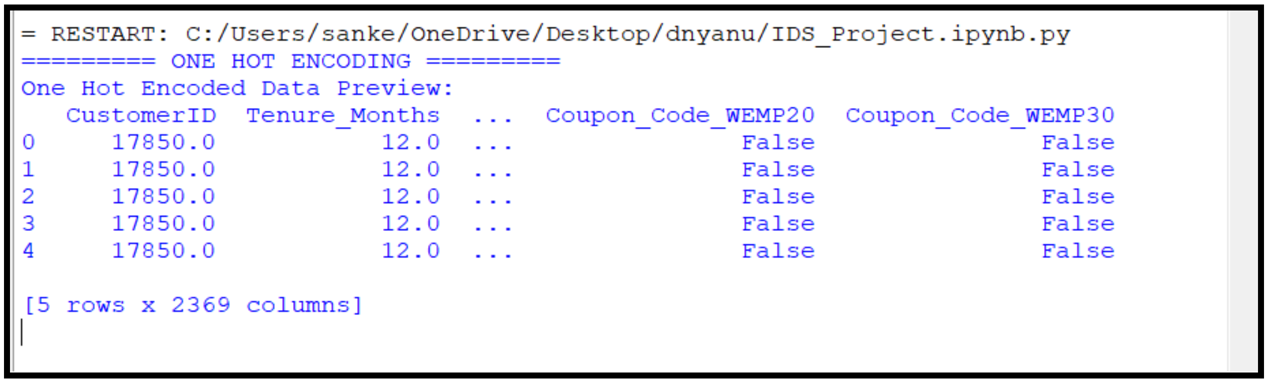
```
#SCATTER PLOT

print("========= SCATTER PLOT =========")

if len(num_cols) > 1:
    plt.scatter(df[num_cols[0]], df[num_cols[1]])
    plt.title("Scatter Plot")
    plt.xlabel(num_cols[0])
    plt.ylabel(num_cols[1])
    plt.show()
```



E)FEATURE ENGINEERING
13)ONE HOT ENCODING

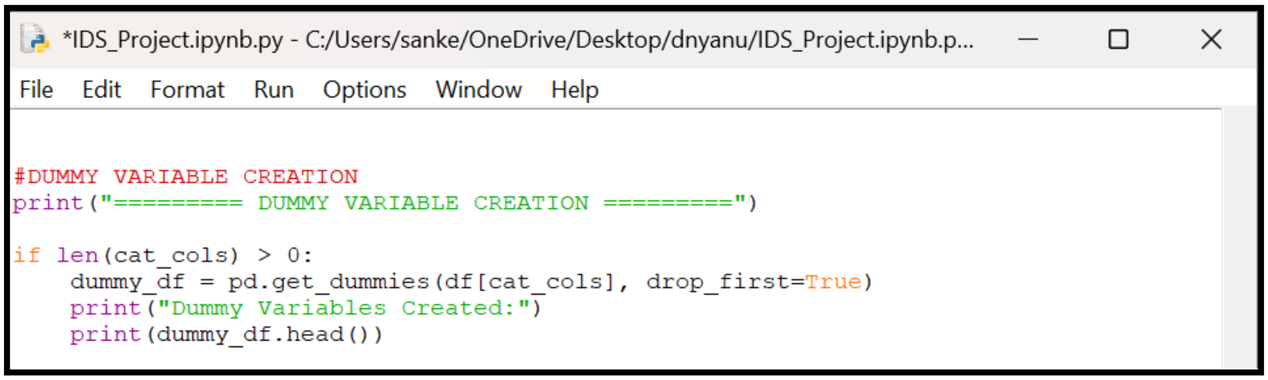

IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py...
File Edit Format Run Options Window Help

```
#ONE HOT ENCODING
print("========= ONE HOT ENCODING =========")

df_onehot = pd.get_dummies(df, columns=cat_cols)

print("One Hot Encoded Data Preview:")
print(df_onehot.head())
```



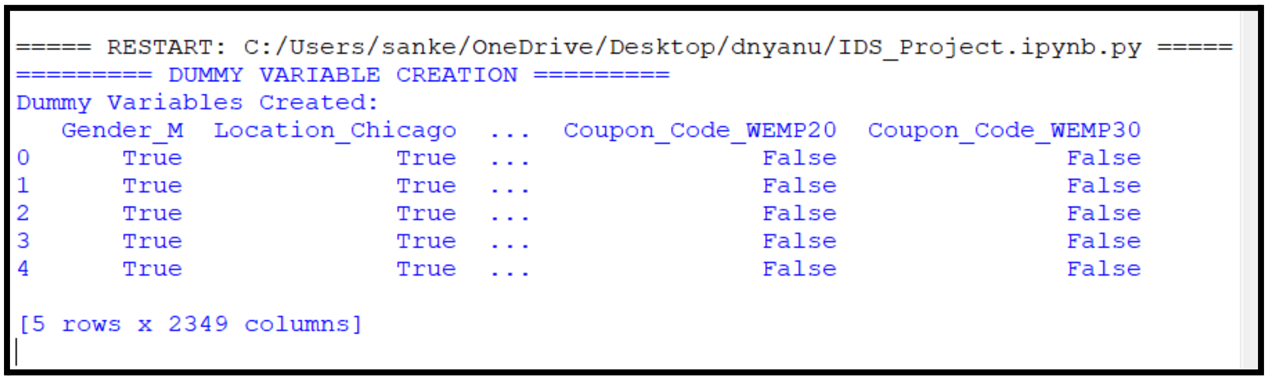
```
= RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py
========= ONE HOT ENCODING =========
One Hot Encoded Data Preview:
   CustomerID  Tenure_Months  ...  Coupon_Code_WEMP20  Coupon_Code_WEMP30
0     17850.0           12.0  ...               False               False
1     17850.0           12.0  ...               False               False
2     17850.0           12.0  ...               False               False
3     17850.0           12.0  ...               False               False
4     17850.0           12.0  ...               False               False

[5 rows x 2369 columns]
```

14)DUMMY VARIABLE CREATION



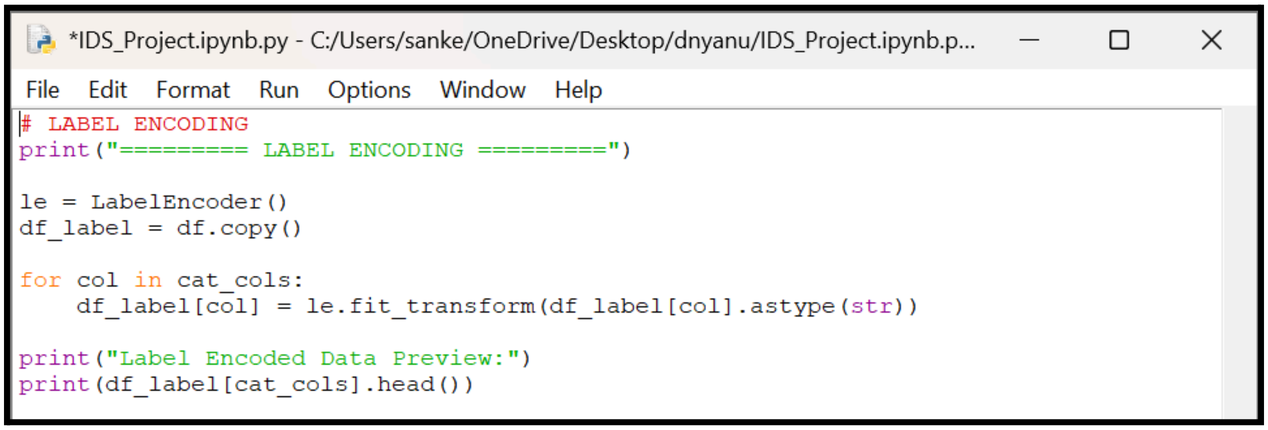
*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```python
#DUMMY VARIABLE CREATION
print("========= DUMMY VARIABLE CREATION =========")

if len(cat_cols) > 0:
    dummy_df = pd.get_dummies(df[cat_cols], drop_first=True)
    print("Dummy Variables Created:")
    print(dummy_df.head())
```



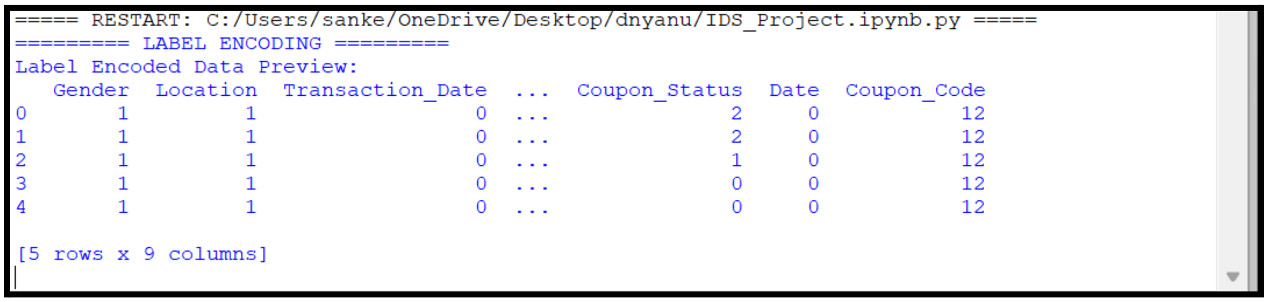
```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= DUMMY VARIABLE CREATION =========
Dummy Variables Created:
   Gender_M  Location_Chicago  ...  Coupon_Code_WEMP20  Coupon_Code_WEMP30
0      True              True  ...               False               False
1      True              True  ...               False               False
2      True              True  ...               False               False
3      True              True  ...               False               False
4      True              True  ...               False               False

[5 rows x 2349 columns]
```

14)LABEL ENCODING



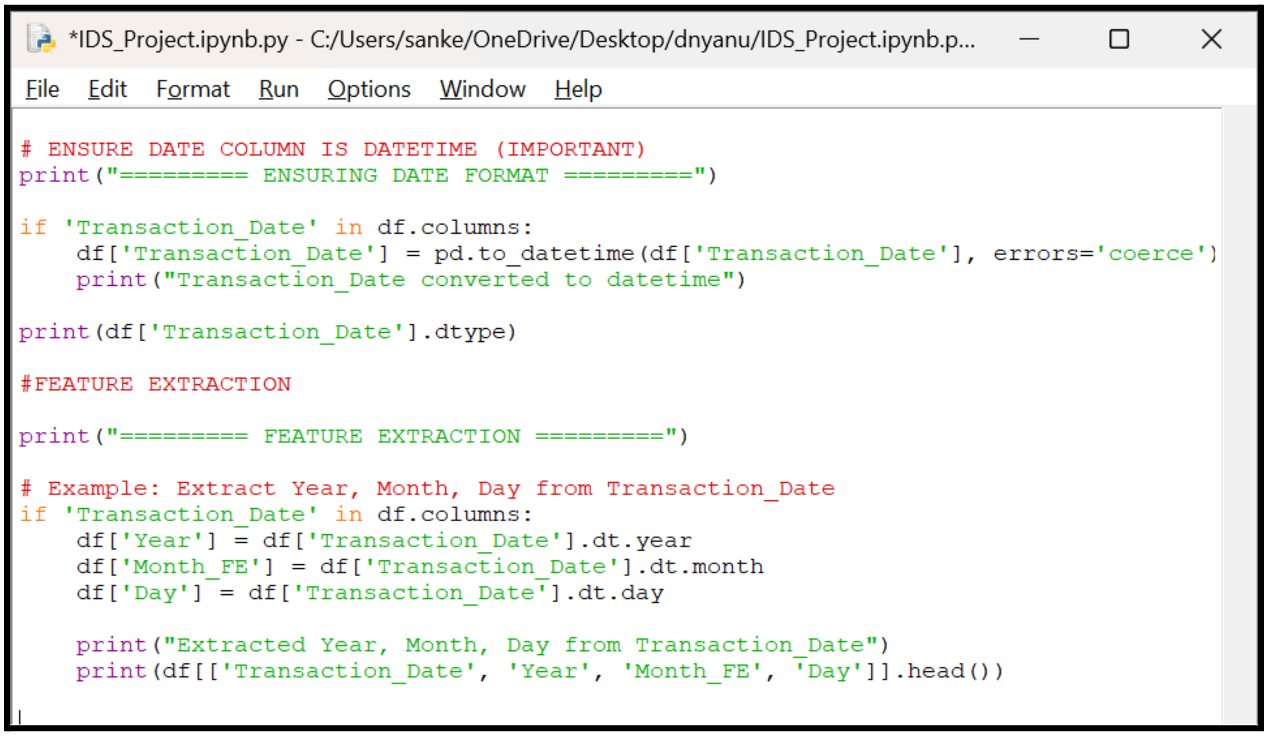
*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```python
# LABEL ENCODING
print("========= LABEL ENCODING =========")

le = LabelEncoder()
df_label = df.copy()

for col in cat_cols:
    df_label[col] = le.fit_transform(df_label[col].astype(str))

print("Label Encoded Data Preview:")
print(df_label[cat_cols].head())
```



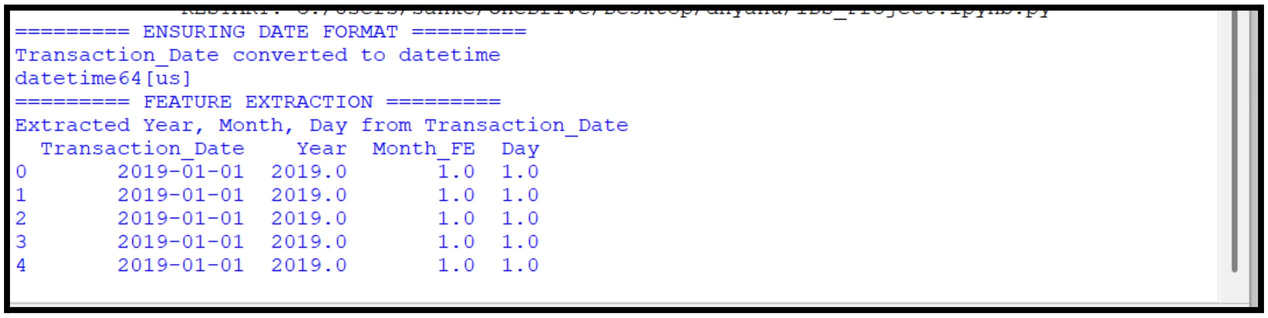
```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= LABEL ENCODING =========
Label Encoded Data Preview:
   Gender  Location  Transaction_Date  ...  Coupon_Status  Date  Coupon_Code
0       1         1                 0  ...              2     0           12
1       1         1                 0  ...              2     0           12
2       1         1                 0  ...              1     0           12
3       1         1                 0  ...              0     0           12
4       1         1                 0  ...              0     0           12

[5 rows x 9 columns]
```

15)FEATURE EXTRACTION



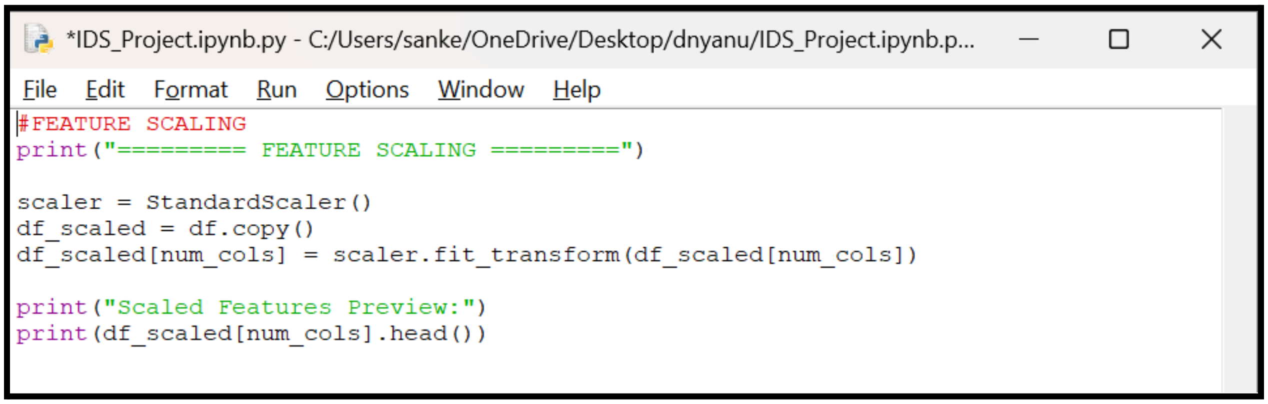
*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...

```python
# ENSURE DATE COLUMN IS DATETIME (IMPORTANT)
print("========= ENSURING DATE FORMAT =========")

if 'Transaction_Date' in df.columns:
    df['Transaction_Date'] = pd.to_datetime(df['Transaction_Date'], errors='coerce')
    print("Transaction_Date converted to datetime")

print(df['Transaction_Date'].dtype)

#FEATURE EXTRACTION

print("========= FEATURE EXTRACTION =========")

# Example: Extract Year, Month, Day from Transaction_Date
if 'Transaction_Date' in df.columns:
    df['Year'] = df['Transaction_Date'].dt.year
    df['Month_FE'] = df['Transaction_Date'].dt.month
    df['Day'] = df['Transaction_Date'].dt.day

    print("Extracted Year, Month, Day from Transaction_Date")
    print(df[['Transaction_Date', 'Year', 'Month_FE', 'Day']].head())
```

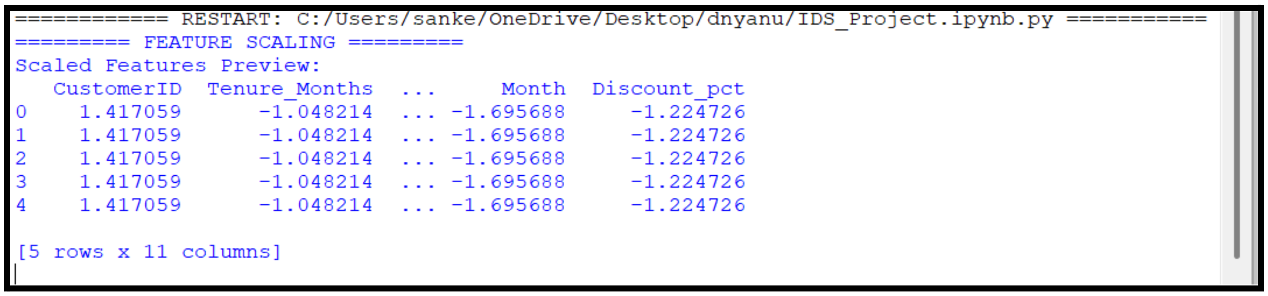


```
========= ENSURING DATE FORMAT =========
Transaction_Date converted to datetime
datetime64[us]
========= FEATURE EXTRACTION =========
Extracted Year, Month, Day from Transaction_Date
  Transaction_Date    Year  Month_FE  Day
0       2019-01-01  2019.0       1.0  1.0
1       2019-01-01  2019.0       1.0  1.0
2       2019-01-01  2019.0       1.0  1.0
3       2019-01-01  2019.0       1.0  1.0
4       2019-01-01  2019.0       1.0  1.0
```

16)FEATURE SCALING

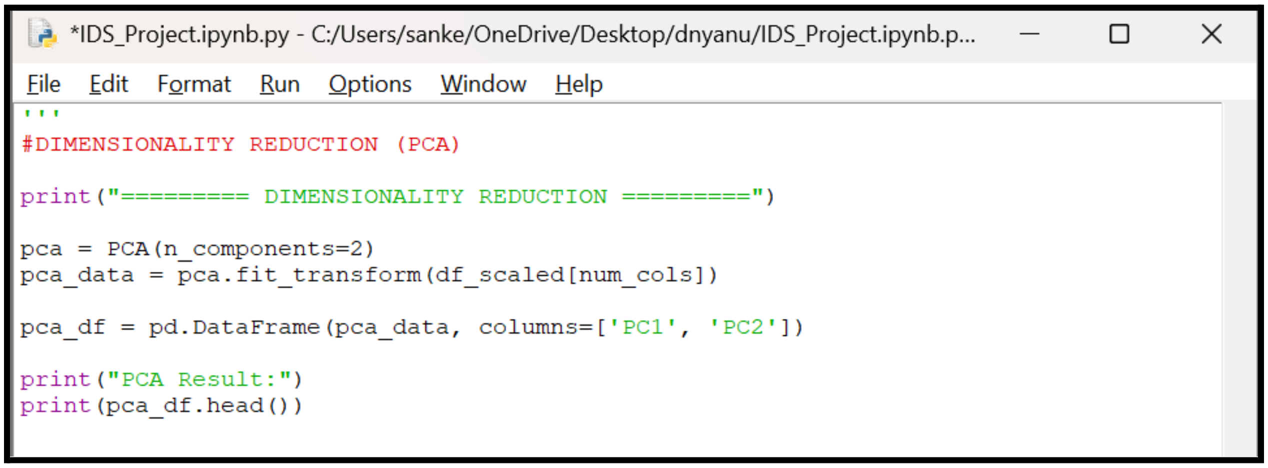

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
#FEATURE SCALING
print("========= FEATURE SCALING =========")

scaler = StandardScaler()
df_scaled = df.copy()
df_scaled[num_cols] = scaler.fit_transform(df_scaled[num_cols])

print("Scaled Features Preview:")
print(df_scaled[num_cols].head())
```

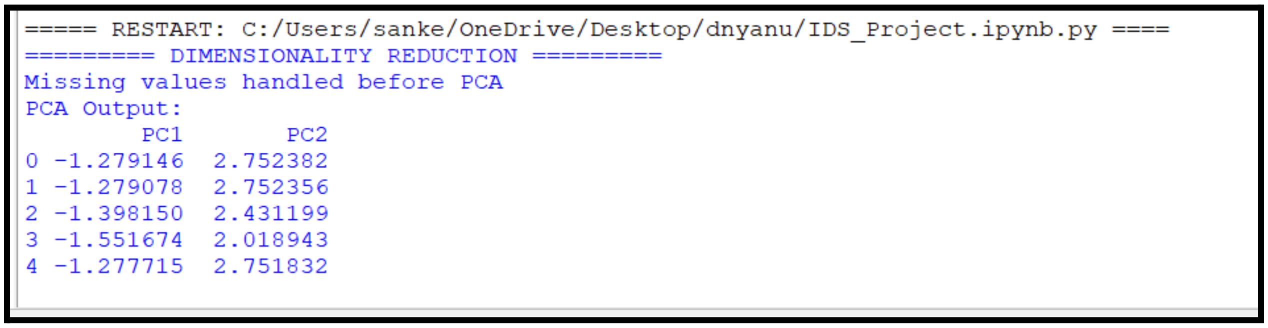


```
============ RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ===========
========= FEATURE SCALING =========
Scaled Features Preview:
   CustomerID  Tenure_Months  ...     Month  Discount_pct
0    1.417059      -1.048214  ... -1.695688     -1.224726
1    1.417059      -1.048214  ... -1.695688     -1.224726
2    1.417059      -1.048214  ... -1.695688     -1.224726
3    1.417059      -1.048214  ... -1.695688     -1.224726
4    1.417059      -1.048214  ... -1.695688     -1.224726

[5 rows x 11 columns]
```

17)DIMENSIONALITY REDUCTION (PCA)


*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

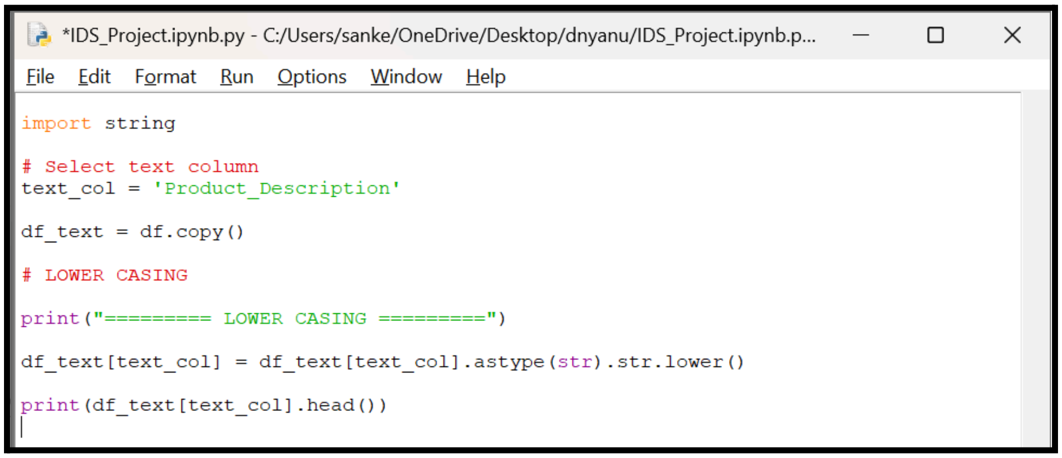
```
'''
#DIMENSIONALITY REDUCTION (PCA)

print("========= DIMENSIONALITY REDUCTION =========")

pca = PCA(n_components=2)
pca_data = pca.fit_transform(df_scaled[num_cols])

pca_df = pd.DataFrame(pca_data, columns=['PC1', 'PC2'])

print("PCA Result:")
print(pca_df.head())
```



```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py ====
========= DIMENSIONALITY REDUCTION =========
Missing values handled before PCA
PCA Output:
        PC1       PC2
0 -1.279146  2.752382
1 -1.279078  2.752356
2 -1.398150  2.431199
3 -1.551674  2.018943
4 -1.277715  2.751832
```

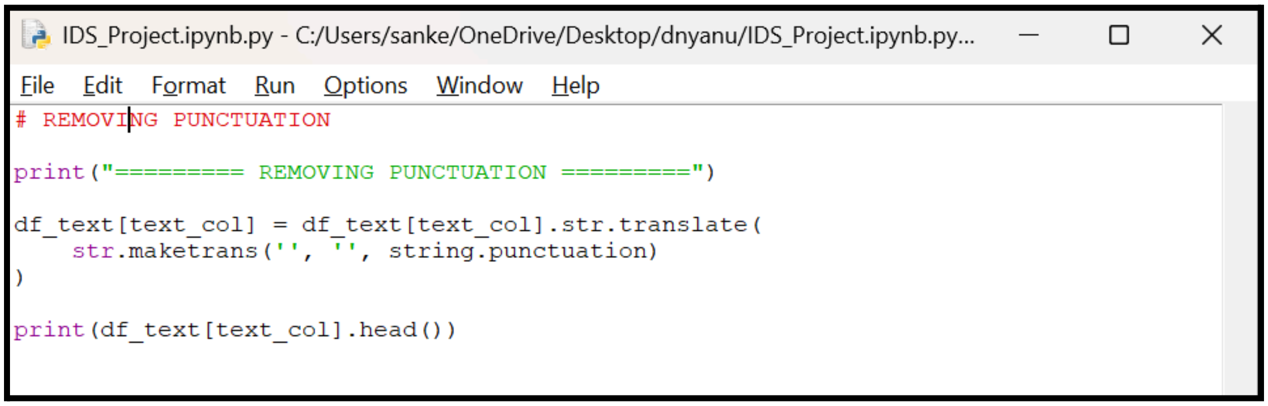
F)TEXT DATA PROCESSING TECHNIQUES

18) LOWER CASING

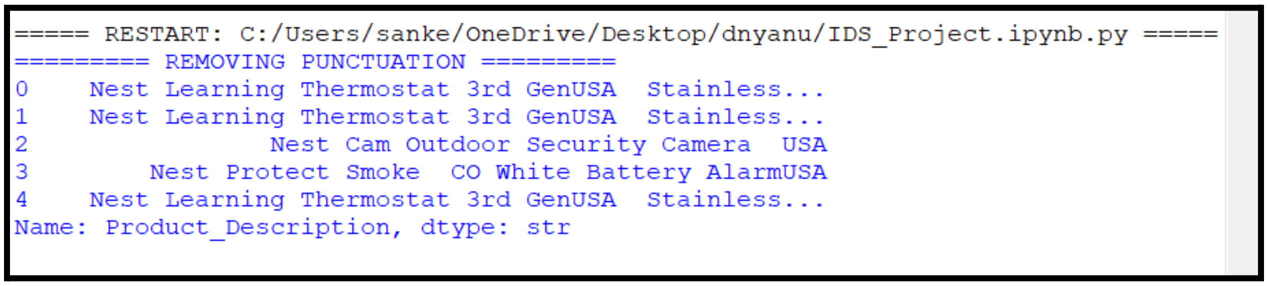

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
import string

# Select text column
text_col = 'Product_Description'

df_text = df.copy()

# LOWER CASING

print("========= LOWER CASING =========")

df_text[text_col] = df_text[text_col].astype(str).str.lower()

print(df_text[text_col].head())
```

19)REMOVING PUNCTUATION

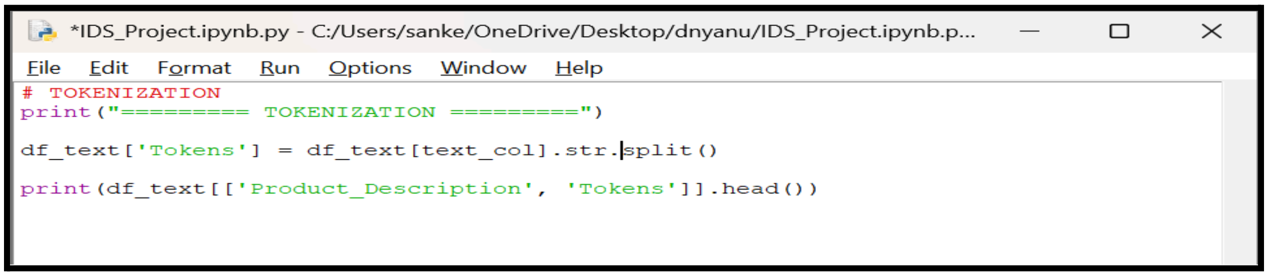

IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py...
File Edit Format Run Options Window Help

```
# REMOVING PUNCTUATION

print("========= REMOVING PUNCTUATION =========")

df_text[text_col] = df_text[text_col].str.translate(
    str.maketrans('', '', string.punctuation)
)

print(df_text[text_col].head())
```

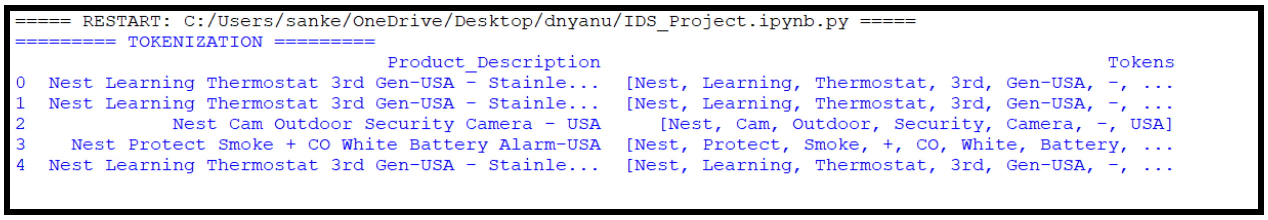


```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= REMOVING PUNCTUATION =========
0    Nest Learning Thermostat 3rd GenUSA  Stainless...
1    Nest Learning Thermostat 3rd GenUSA  Stainless...
2               Nest Cam Outdoor Security Camera  USA
3         Nest Protect Smoke  CO White Battery AlarmUSA
4    Nest Learning Thermostat 3rd GenUSA  Stainless...
Name: Product_Description, dtype: str
```

20) TOKENIZATION

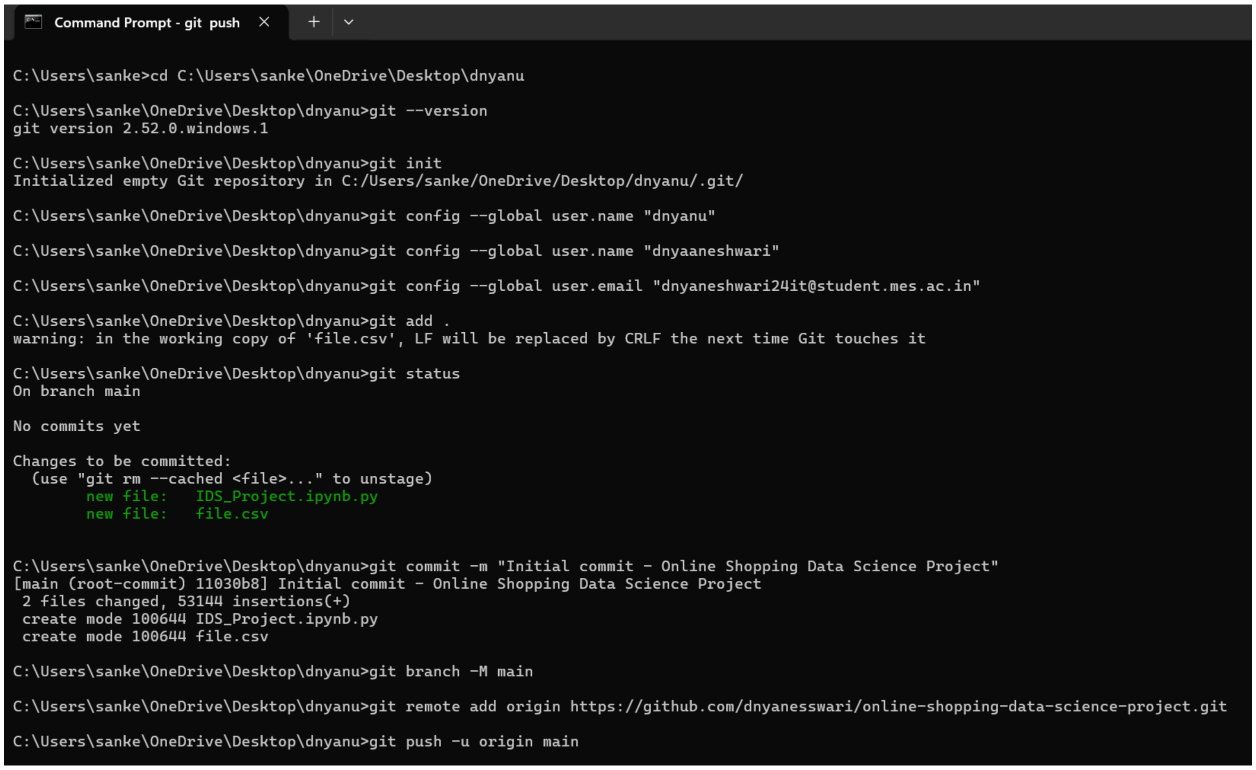

*IDS_Project.ipynb.py - C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.p...
File Edit Format Run Options Window Help

```
# TOKENIZATION
print("========= TOKENIZATION =========")

df_text['Tokens'] = df_text[text_col].str.split()

print(df_text[['Product_Description', 'Tokens']].head())
```



```
===== RESTART: C:/Users/sanke/OneDrive/Desktop/dnyanu/IDS_Project.ipynb.py =====
========= TOKENIZATION =========
                              Product_Description                                      Tokens
0  Nest Learning Thermostat 3rd Gen-USA - Stainle...  [Nest, Learning, Thermostat, 3rd, Gen-USA, -, ...
1  Nest Learning Thermostat 3rd Gen-USA - Stainle...  [Nest, Learning, Thermostat, 3rd, Gen-USA, -, ...
2             Nest Cam Outdoor Security Camera - USA     [Nest, Cam, Outdoor, Security, Camera, -, USA]
3    Nest Protect Smoke + CO White Battery Alarm-USA  [Nest, Protect, Smoke, +, CO, White, Battery, ...
4  Nest Learning Thermostat 3rd Gen-USA - Stainle...  [Nest, Learning, Thermostat, 3rd, Gen-USA, -, ...
```

Github Code:-



Command Prompt - git push

```
C:\Users\sanke>cd C:\Users\sanke\OneDrive\Desktop\dnyanu

C:\Users\sanke\OneDrive\Desktop\dnyanu>git --version
git version 2.52.0.windows.1

C:\Users\sanke\OneDrive\Desktop\dnyanu>git init
Initialized empty Git repository in C:/Users/sanke/OneDrive/Desktop/dnyanu/.git/

C:\Users\sanke\OneDrive\Desktop\dnyanu>git config --global user.name "dnyanu"

C:\Users\sanke\OneDrive\Desktop\dnyanu>git config --global user.name "dnyaaneshwari"

C:\Users\sanke\OneDrive\Desktop\dnyanu>git config --global user.email "dnyaneshwari24it@student.mes.ac.in"

C:\Users\sanke\OneDrive\Desktop\dnyanu>git add .
warning: in the working copy of 'file.csv', LF will be replaced by CRLF the next time Git touches it

C:\Users\sanke\OneDrive\Desktop\dnyanu>git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   IDS_Project.ipynb.py
        new file:   file.csv


C:\Users\sanke\OneDrive\Desktop\dnyanu>git commit -m "Initial commit - Online Shopping Data Science Project"
[main (root-commit) 11030b8] Initial commit - Online Shopping Data Science Project
 2 files changed, 53144 insertions(+)
 create mode 100644 IDS_Project.ipynb.py
 create mode 100644 file.csv

C:\Users\sanke\OneDrive\Desktop\dnyanu>git branch -M main

C:\Users\sanke\OneDrive\Desktop\dnyanu>git remote add origin https://github.com/dnyanesswari/online-shopping-data-science-project.git

C:\Users\sanke\OneDrive\Desktop\dnyanu>git push -u origin main
```

CONCLUSION

In this project, the complete data science pipeline was successfully implemented using Python. The dataset was first explored and cleaned by handling missing values, removing duplicates, detecting outliers, and correcting inconsistent data. These steps ensured that the data was accurate, reliable, and ready for analysis.

Further, various data transformation and exploratory data analysis (EDA) techniques were applied to understand the underlying patterns and distributions in the data. Visualizations such as histograms, box plots, bar charts, and scatter plots helped in gaining meaningful insights into customer behavior and transaction trends.

Feature engineering techniques including encoding, scaling, feature extraction, and dimensionality reduction using Principal Component Analysis (PCA) were performed to improve data quality and reduce complexity. Text data processing techniques such as lowercasing, removing punctuation, and tokenization were also applied to handle textual attributes effectively.

Overall, this project demonstrates how raw data can be transformed into useful information through systematic data preprocessing and analysis. The project provided hands-on experience with essential data science concepts and tools, making it a strong foundation for real-world data analysis and machine learning applications.