**Black Friday Sales Prediction**

1. **Introduction**

A retail company ABC Private Limited wants to understand the customer purchase behavior (specifically, purchase amount) against various products of different categories. They have shared purchase summary of various customers for selected high-volume products from last month. The data set also contains customer demographics (age, gender, marital status, city type, stay in current city), product details (product id and product category) and Total purchase amount from last month.

Now, they want to build a model to predict the purchase amount of customer against various products which will help them to create personalized offer for customers against different products.

1. **Problem Statement**

Setting the optimum price of a product is often a problem for retailers, especially during a sale like the ‘Black Friday’. The challenge here is to predict purchase prices of various products purchased by customers based on historical purchase patterns. The data contains features like age, gender, marital status, categories of products purchased, city demographics etc. My solution looks into estimating the sale price. Given the dataset, I could estimate the price a customer would pay for an item with known Product Identification and Category as well as having customer Information.

1. **About the Dataset**

The data was found from the “Black Friday” dataset provided by Kaggle’s website. <https://www.kaggle.com/mehdidag/black-friday> and was saved into the local as ‘**BlackFriday.csv**’.

This is the current data they have available:

|  |  |
| --- | --- |
| **Variable** | **Definition** |
| User\_ID | User ID |
| Product\_ID | Product ID |
| Gender | Sex of User |
| Age | Age in bins |
| Occupation | Occupation (Masked) |
| City\_Category | Category of the City (A, B, C) |
| Stay\_In\_Current\_City\_Years | Number of years stay in current city |
| Marital\_Status | Marital Status |
| Product\_Category\_1 | Product Category (Masked) |
| Product\_Category\_2 | Product may belong to another category (Also Masked) |
| Product\_Category\_3 | Product may belong to another category (Also Masked) |
| Purchase | Purchase Amount |

1. **Steps on Data-Wrangling**

**Taking a quick look at the Data Structure:**

Let’s start by importing some libraries and our data.

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

%matplotlib inline

**Acquiring the data and getting some insights of the dataset:**

df = pd.read\_csv('BlackFriday.csv')

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 537577 entries, 0 to 537576

Data columns (total 12 columns):

User\_ID 537577 non-null int64

Product\_ID 537577 non-null object

Gender 537577 non-null object

Age 537577 non-null object

Occupation 537577 non-null int64

City\_Category 537577 non-null object

Stay\_In\_Current\_City\_Years 537577 non-null object

Marital\_Status 537577 non-null int64

Product\_Category\_1 537577 non-null int64

Product\_Category\_2 370591 non-null float64

Product\_Category\_3 164278 non-null float64

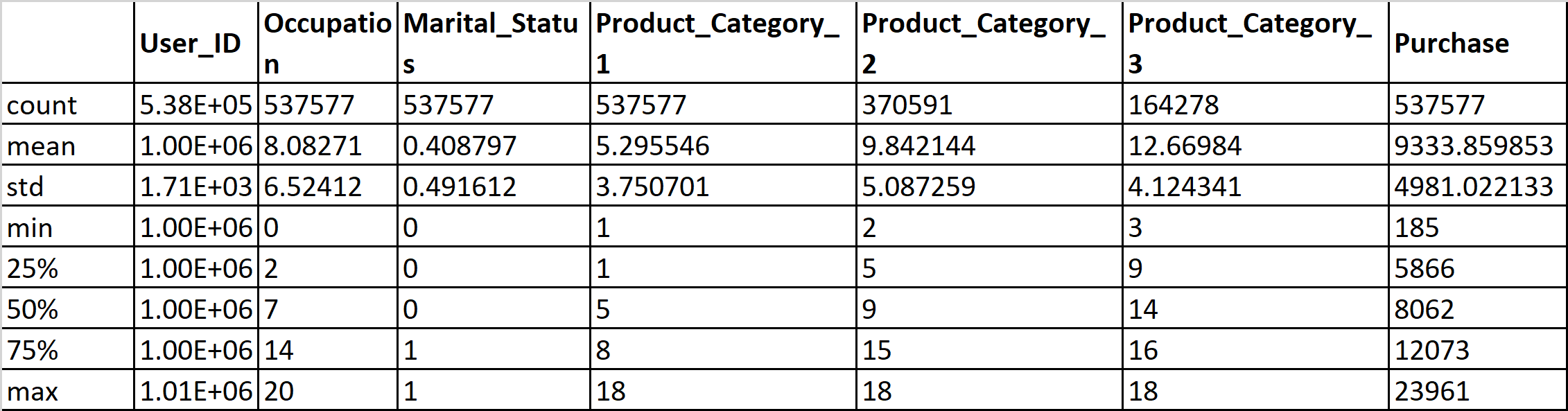
Purchase 537577 non-null int64

dtypes: float64(2), int64(5), object(5)

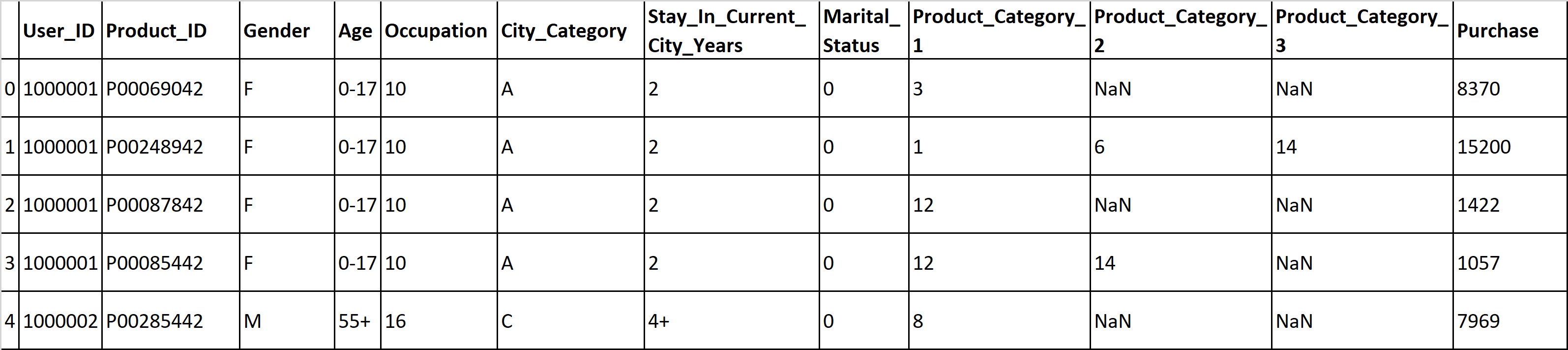
memory usage: 49.2+ MB

We see from the above data that there are a total of 12 columns with different formats of data. Also, we see lot many missing values in the Product\_Category\_2 and Product\_Category\_3.

df.describe()



df.head()



### Looks like we have some null/NaN values in the product categories 2 and 3. Next step would be to find the distinct values followed by cleaning the null/NaN values.

# Finding the count of distinct values in the dataset:

df.nunique()

User\_ID 5891

Product\_ID 3623

Gender 2

Age 7

Occupation 21

City\_Category 3

Stay\_In\_Current\_City\_Years 5

Marital\_Status 2

Product\_Category\_1 18

Product\_Category\_2 17

Product\_Category\_3 15

Purchase 17959

dtype: int64

### We can see from the above data that all the columns except 'Purchase' column are categorical values and the 'Purchase' column would be considered as non-categorical.

### Replacing the NaN's with 0. Also, we can change the type of Product Category 2 and 3 from float to int type.

df = df.fillna(0)

df["Product\_Category\_2"] = df["Product\_Category\_2"].astype(int)

df["Product\_Category\_3"] = df["Product\_Category\_3"].astype(int)

df.head(10)

### 

### We see from the above table that all the NaN/Null values have been replaced with '0'.

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 537577 entries, 0 to 537576

Data columns (total 12 columns):

User\_ID 537577 non-null int64

Product\_ID 537577 non-null object

Gender 537577 non-null object

Age 537577 non-null object

Occupation 537577 non-null int64

City\_Category 537577 non-null object

Stay\_In\_Current\_City\_Years 537577 non-null object

Marital\_Status 537577 non-null int64

Product\_Category\_1 537577 non-null int64

Product\_Category\_2 537577 non-null int32

Product\_Category\_3 537577 non-null int32

Purchase 537577 non-null int64

dtypes: int32(2), int64(5), object(5)

memory usage: 45.1+ MB

### The dataset is now cleaned and ready for further exploration.