BLACK FRIDAY SALES PREDICTION

Using Spark Pipeline

Aparna Arvind Rangari
A2306

IMPORTANT LINKS

DATA :

https://github.com/Aparna9096/bdns/blob/main/train.csv

COLAB NOTEBOOK :

https://colab.research.google.com/drive/1I14g-DUtDHpWMMmWaxVHJ0ssizJy3fMS#scrollTo=b3tGD1Gs4BhH

PROBLEM STATEMENT

Retail is the sale of goods and services from individuals or businesses to the end- user. The retail industry provides consumers with goods and services for their everyday needs. In retail one of crucial part is to understand the consumer behavior and make various arrangements for the sales of the company. 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.

ABOUT THE DATASET

This dataset comprises of sales transactions captured at a retail store. This is a regression problem. The dataset has 550,069 rows and 12 columns. Problem: Predict purchase amount.

<u>Data Overview Dataset has 550068 rows (transactions) and 12 columns (features) as described below:</u>

➢User_ID: Unique ID of the user.

➢Product_ID: Unique ID of the product.

➢Gender: indicates the gender of the person making the transaction.

➢Age: indicates the age group of the person making the transaction.

➢City_Category: User's living city category. Cities are categorized into 3 different categories 'A', 'B' and 'C'.

➢Stay_In_Current_City_Years: Indicates how long the users has lived in this city.

➢Marital_Status: is 0 if the user is not married and 1 otherwise.

➢Product_Category_1 to _3: Category of the product. All 3 are already labeled with numbers.

➢Purchase: Purchase amount.

➢Occupation: shows the occupation of the user, already labeled with numbers 0 to 20.

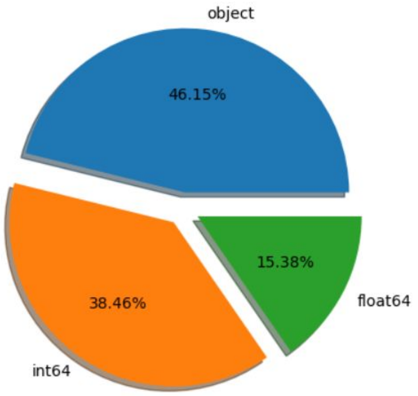
EDA on MongoDB

While handling null values found that aot all functions can be done on pyMongo so I have to create a dataframe from pandas and this can help me in Describing the dataset and finding the null values and many more.
As i have got that product category 1 and 2 has null value and this are the irrelevant features so i have dropped it.

	missing_values	percent_missing
User_ID	0	0.000000
Product_ID	0	0.000000
Gender	0	0.000000
Age	0	0.000000
Occupation	0	0.000000
City_Category	0	0.000000
Stay_In_Current_City_Years	0	0.000000
Marital_Status	0	0.000000
Product_Category_1	0	0.000000
Product_Category_2	173638	31.566643
Product_Category_3	383247	69.672659
Purchase	0	0.000000

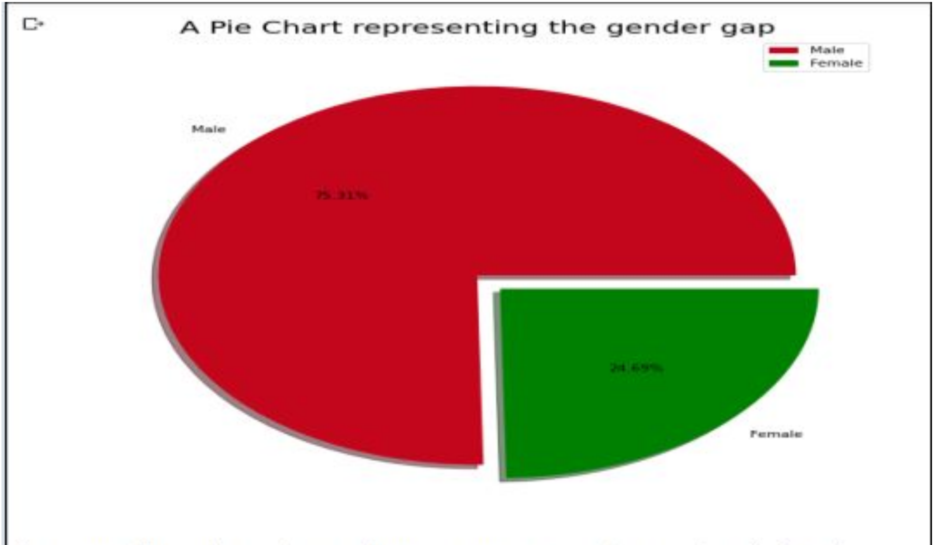
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0                               550068 non-null  object
 1   User_ID                     550068 non-null  int64
 2   Product_ID                  550068 non-null  object
 3   Gender                      550068 non-null  object
 4   Age                         550068 non-null  object
 5   Occupation                  550068 non-null  int64
 6   City_Category               550068 non-null  object
 7   Stay_In_Current_City_Years  550068 non-null  object
 8   Marital_Status              550068 non-null  int64
 9   Product_Category_1          550068 non-null  int64
 10  Product_Category_2          550068 non-null  float64
 11  Purchase                    550068 non-null  int64
dtypes: float64(1), int64(5), object(6)
memory usage: 50.4+ MB
```

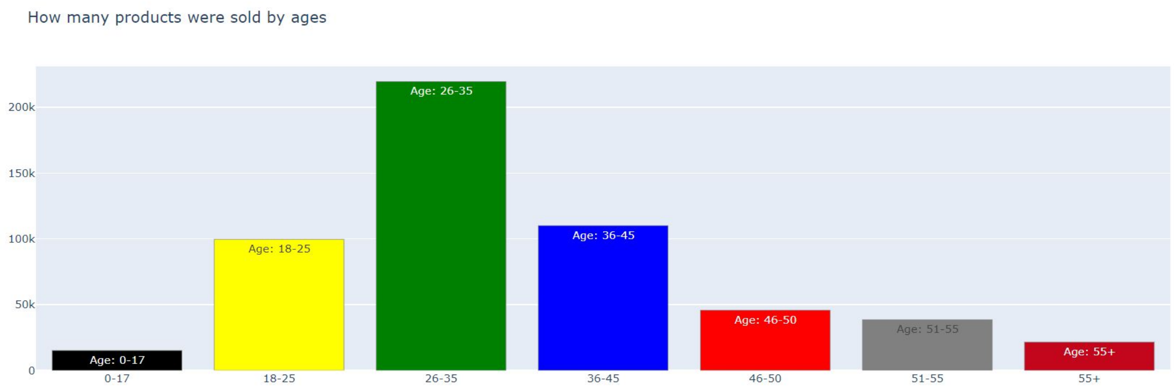


Type of Data

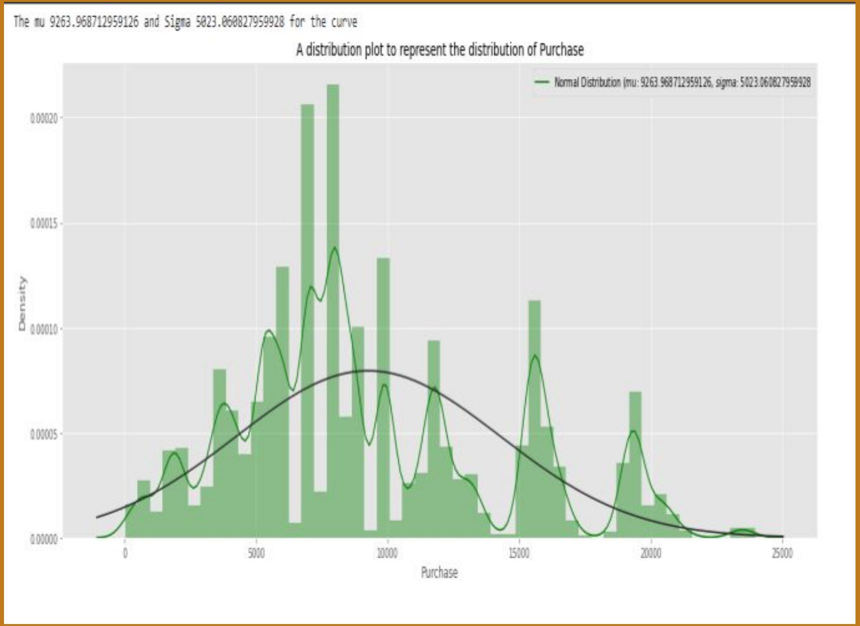
Categorical Variables

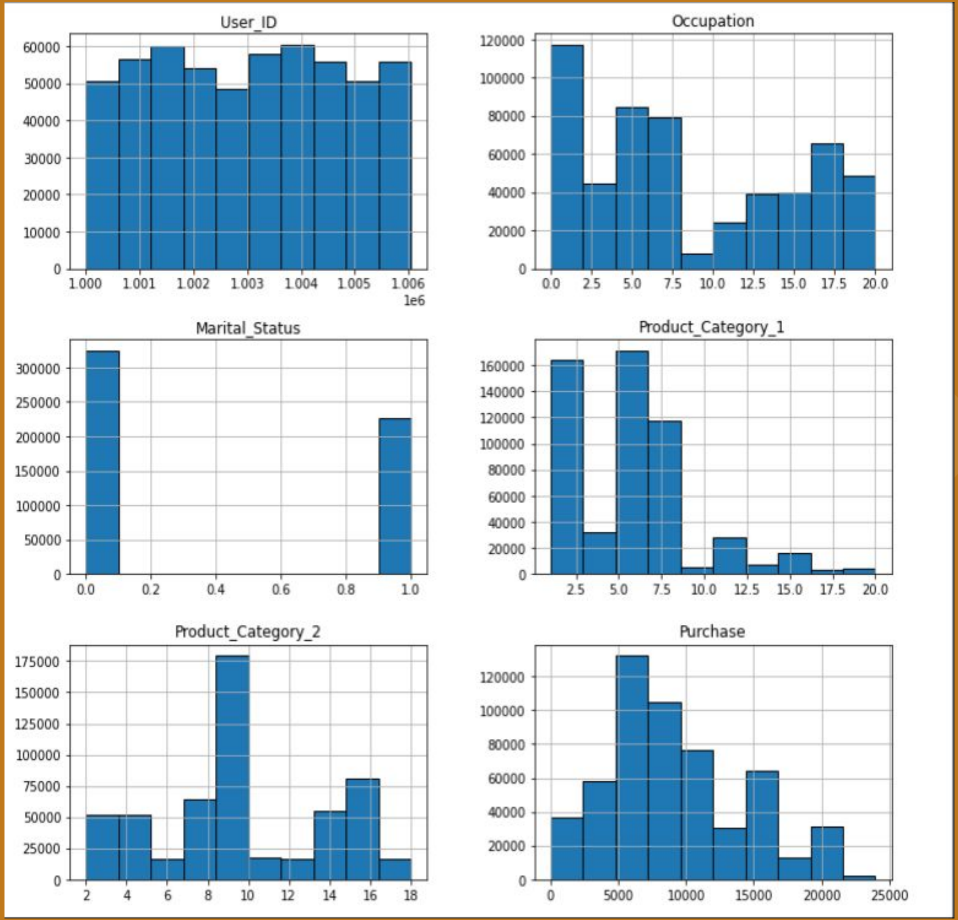


On average the male gender spends more money on purchase contrary to female.



Numerical Variable





Insights from Variable

High End Product

Product ID : P00052842 has the Expensive Product with **Amount 23961**.

The Whale Customer

User_ID : 1004277 ~ Purchase_Amount : 10536909 has the maximum Purchase.

The Loyal Customer

User ID : 1001680 ~ Purchase Amount : 8699596 has max frequency.

Most Demanded Product

Product ID : P00265242 is the favourite Product and has frequency of 1880.

Cheapest Product

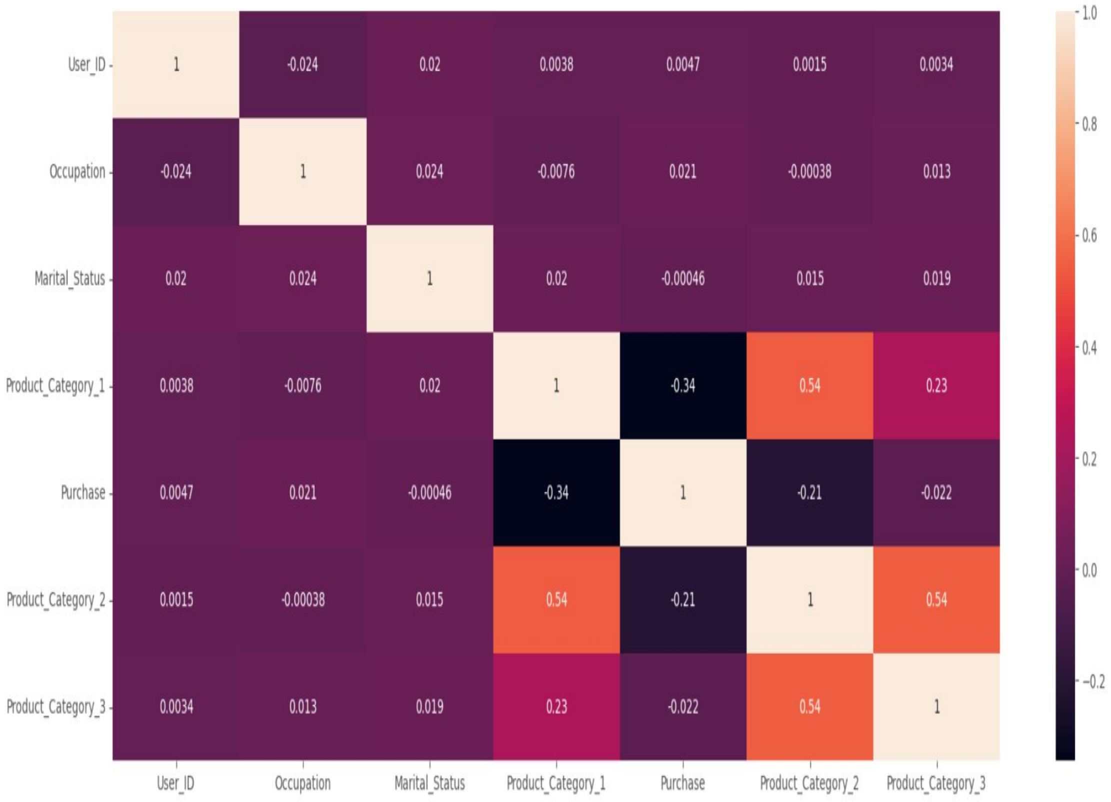
Product ID : P00370293 has the Expensive Product with **Amount 12**.

A basic observation is that:

1. Product P00265242 is the most popular product.
2. Most of the transactions were made by men.
3. Age group with most transactions was 26-35.
4. City Category with most transactions was B

Checking for Multicollinearity

From the correlation heatmap we can see that the linear association/ correlation between our variables is not more than 0.4 in all the cases which can be considered as weak correlation. So we can conclude that there are minimal chances of multicollinearity in our dataset.



Data Pre-Processing

➢Dropping Irrelevant Variables

➢For the purpose of data preprocessing we have used the following tools :

- String Indexer
- Assembler
- Standard Scaler

```
303] Data_StringIndexer2.show(5)
```

Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Purchase	GenderIndex	AgeIndex	City_CategoryIndex	Stay_In_Current_City_YearsIndex
F	0-17	10	A	2	0	3	8370	1.0	6.0	2.0	1.0
F	0-17	10	A	2	0	1	15200	1.0	6.0	2.0	1.0
F	0-17	10	A	2	0	12	1422	1.0	6.0	2.0	1.0
F	0-17	10	A	2	0	12	1057	1.0	6.0	2.0	1.0
M	55+	16	C	4+	0	8	7969	0.0	5.0	1.0	3.0

only showing top 5 rows

```
[306] Data_assembler.show(5)
```

pation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Purchase	GenderIndex	AgeIndex	City_CategoryIndex	Stay_In_Current_City_YearsIndex	features
10	A	2	0	3	8370	1.0	6.0	2.0	1.0	[1.0,6.0,10.0,2.0...
10	A	2	0	1	15200	1.0	6.0	2.0	1.0	[1.0,6.0,10.0,2.0...
10	A	2	0	12	1422	1.0	6.0	2.0	1.0	[1.0,6.0,10.0,2.0...
10	A	2	0	12	1057	1.0	6.0	2.0	1.0	[1.0,6.0,10.0,2.0...
16	C	4+	0	8	7969	0.0	5.0	1.0	3.0	[0.0,5.0,16.0,1.0...

5 rows

Train Test Split

Our dataset is split into training and testing in the ratio of 80 percent, 20 percent respectively.

```
# Split the data into train and test sets
train_data, test_data = scaled_df.randomSplit([.8,.2],seed=1234)
```

```
train_data.show(5)
```

```
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
|Gender| Age|Occupation|City_Category|Stay_In_Current_City_Years|Marital_Status|Product_Category_1|Purchase|GenderIndex|AgeIndex|City_CategoryIndex|Stay_In_Current_City_YearsIndex|
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
|     F|0-17|         0|            A|                         2|             0|                 1|   12113|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            A|                         2|             0|                 3|   10962|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            A|                         2|             0|                 5|    5210|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            A|                         2|             0|                 5|    7029|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            A|                         2|             0|                 5|    7180|        1.0|     6.0|              2.0|                            1.0|[1.
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
only showing top 5 rows
```

```
test_data.show(5)
```

```
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
|Gender| Age|Occupation|City_Category|Stay_In_Current_City_Years|Marital_Status|Product_Category_1|Purchase|GenderIndex|AgeIndex|City_CategoryIndex|Stay_In_Current_City_YearsIndex|
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
|     F|0-17|         0|            A|                         2|             0|                 3|   10807|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            A|                         2|             0|                 5|    5341|        1.0|     6.0|              2.0|                            1.0|[1.
|     F|0-17|         0|            B|                         1|             0|                 1|   15276|        1.0|     6.0|              0.0|                            0.0|(7,|
|     F|0-17|         0|            B|                         1|             0|                 1|   15647|        1.0|     6.0|              0.0|                            0.0|(7,|
|     F|0-17|         0|            B|                         1|             0|                 1|   15778|        1.0|     6.0|              0.0|                            0.0|(7,|
+------+----+----------+-------------+--------------------------+--------------+------------------+--------+-----------+--------+-----------------+-------------------------------+----
only showing top 5 rows
```

MODEL TRAINING USING PYSPARK

1. Linear Regression
2. Random Forest
3. Gradient Boost Regressor

3 different models used for training the data and the outputs were evaluated

```
[341] lr_rmse=regEval.evaluate(lr1model_predictions)
      print(round(lr_rmse,3), 'is the RMSE of the LR pipeline')

      4716.947 is the RMSE of the LR pipeline
```

```
rf_rmse=regEval.evaluate(rf1model_predictions)
print(round(rf_rmse,3), 'is the RMSE of the RF pipeline')

3890.067 is the RMSE of the RF pipeline
```

```
print(round(gbr_rmse,3), 'is the RMSE of the GBR pipeline')

2931.354 is the RMSE of the GBR pipeline
```

MODEL EVALUATION

- Gredient Boot Regressor: The RMSE measures the average deviation of predicted purchase amounts from the actual values. In our GBR model, the RMSE of 2931.35 indicates the typical difference between predicted and actual purchase amounts.

- Random Forest : The RMSE is a measure of the average difference between predicted and actual purchase amounts. In our RF model, the RMSE of 3890.07 indicates the typical deviation of predictions from the true values.

- Linear Regression : The RMSE measures the average deviation of predicted purchase amounts from the actual values. In the linear regression model, the RMSE of 4709.14 indicates the typical difference between predicted and actual purchase amounts.

- R-squared represents the proportion of the variance in the dependent variable (Purchase) that is predictable from the independent variables. An R^2 of 0.1205 suggests that approximately 12.05% of the variability in purchase amounts can be explained by the linear regression model.

CONCLUSION

Gredient Boot Regressor is giving the best result in our case when compared with other models because GBR starts with building a primary model from available training data sets then it identifies the errors present in the base model. After identifying the error, a secondary model is built, and further, a third model is introduced in this process. In this way, this process of introducing more models is continued until we get a complete training data set by which model predicts correctly.