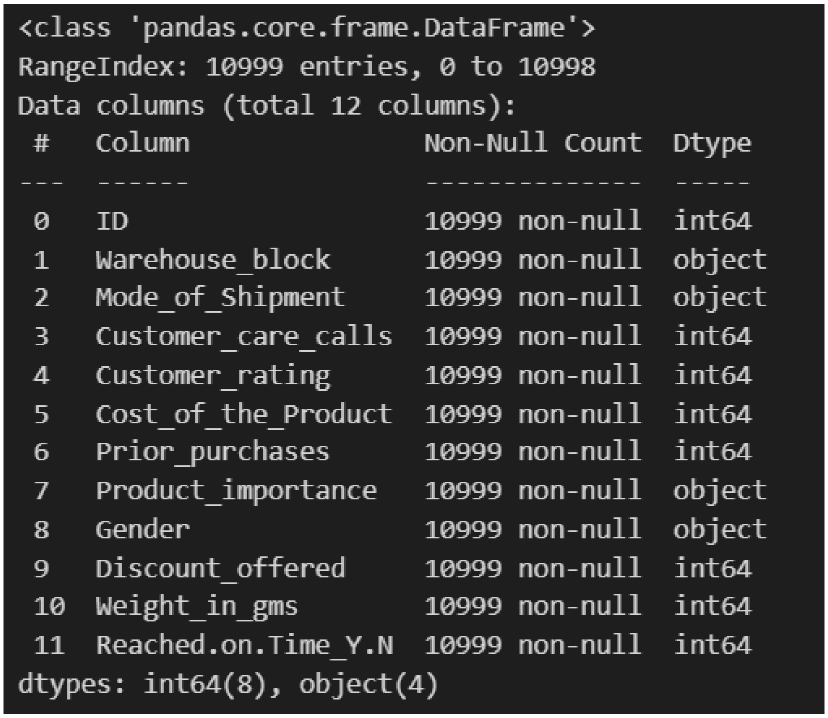
Data Collection and Preprocessing Phase

Date	09 July 2024
Team ID	SWTID1720499933
Project Title	Ecommerce Shipping Prediction Using Machine Learning
Maximum Marks	6 Marks

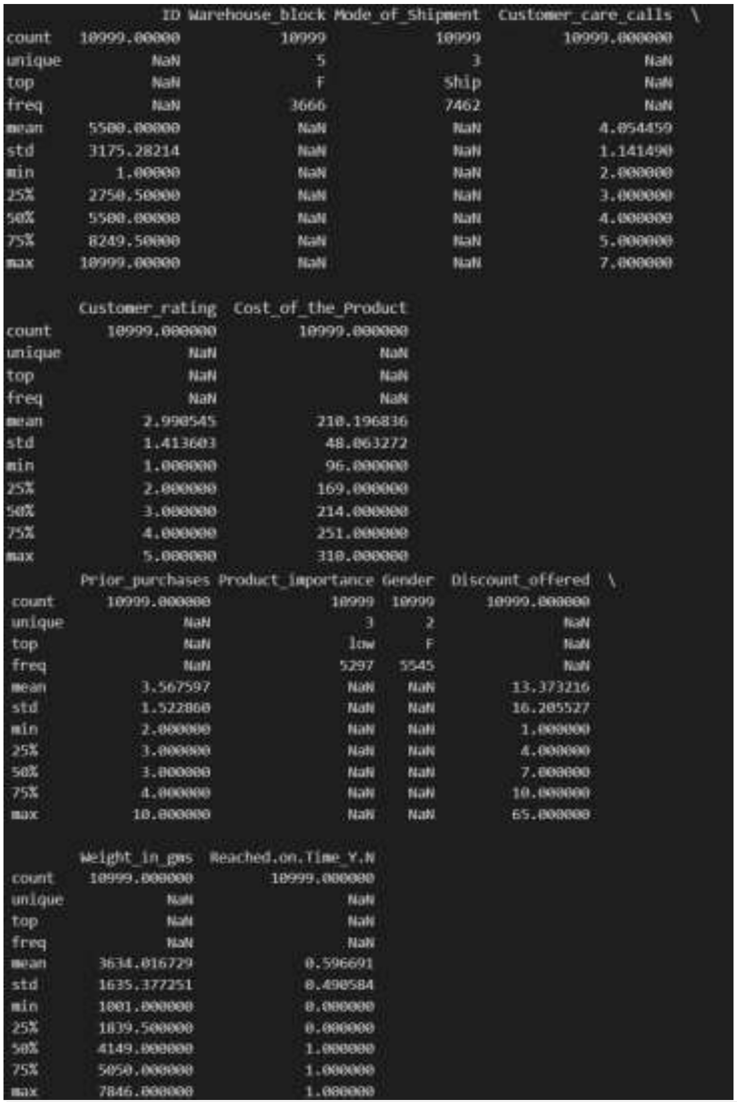
Data Exploration and Preprocessing Template

Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

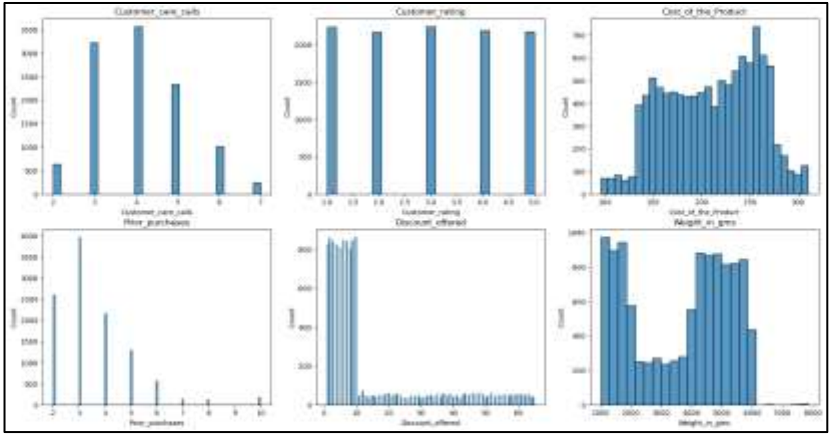
Section	**Description**
Data Overview	Dimensions: 10999 rows x 12 columns 

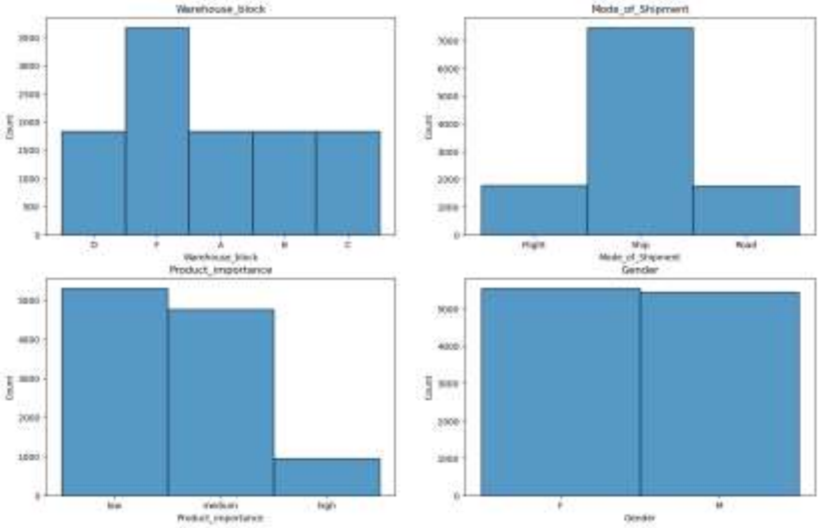
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10999 entries, 0 to 10998
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   ID                   10999 non-null  int64 
 1   Warehouse_block      10999 non-null  object
 2   Mode_of_Shipment     10999 non-null  object
 3   Customer_care_calls  10999 non-null  int64 
 4   Customer_rating      10999 non-null  int64 
 5   Cost_of_the_Product  10999 non-null  int64 
 6   Prior_purchases      10999 non-null  int64 
 7   Product_importance   10999 non-null  object
 8   Gender               10999 non-null  object
 9   Discount_offered     10999 non-null  int64 
 10  Weight_in_gms        10999 non-null  int64 
 11  Reached.on.Time_Y.N  10999 non-null  int64 
dtypes: int64(8), object(4)
```

Univariate Analysis

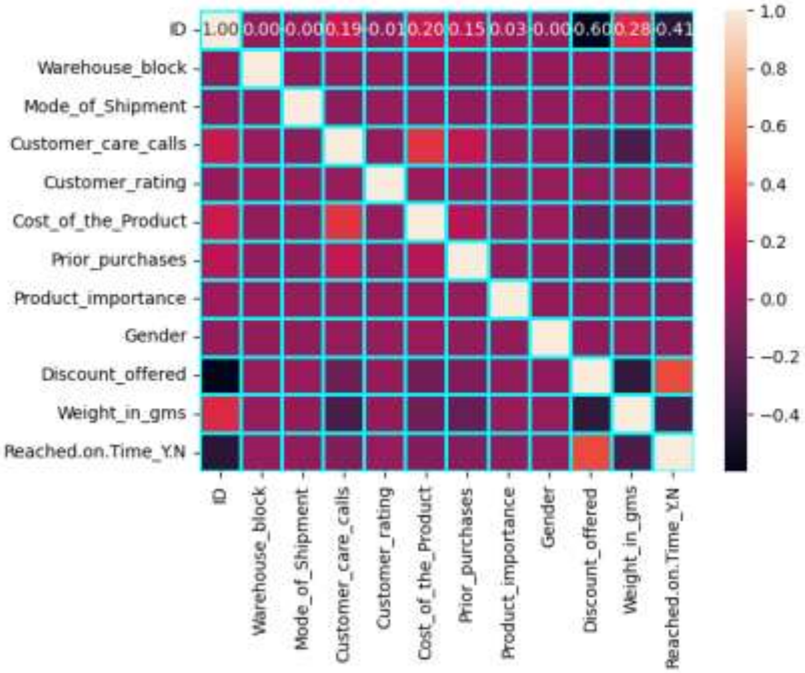
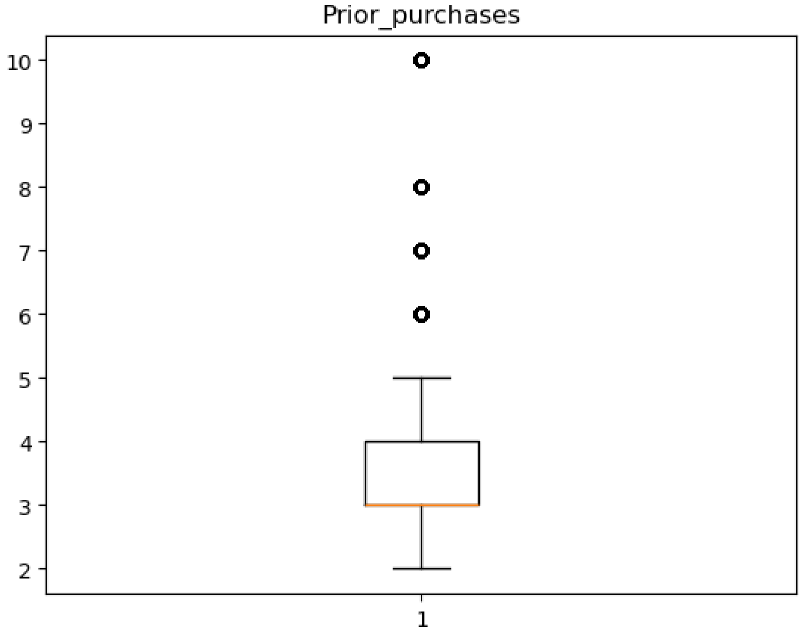


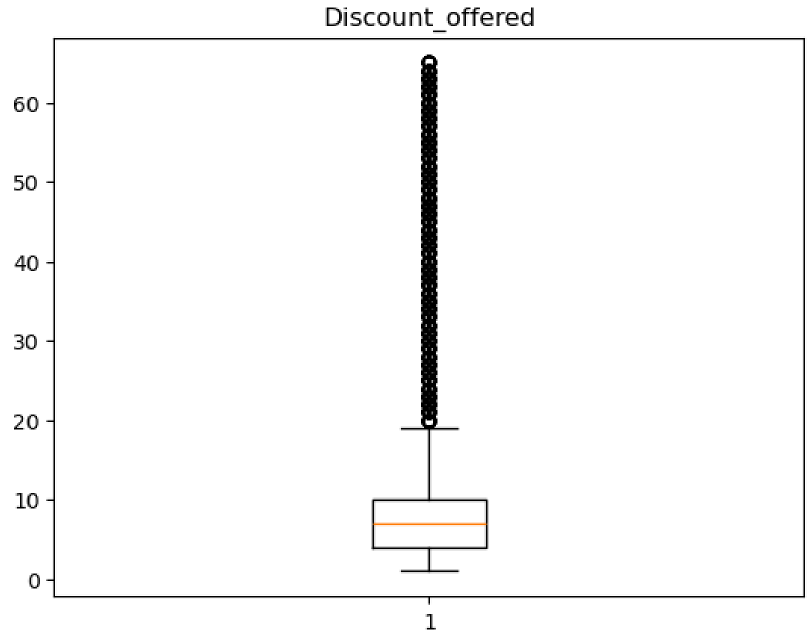
```
                 ID Warehouse_block Mode_of_Shipment  Customer_care_calls  \
count   10999.00000           10999            10999         10999.000000
unique          NaN               5                3                  NaN
top             NaN               F             Ship                  NaN
freq            NaN            3666             7462                  NaN
mean     5500.00000             NaN              NaN             4.054459
std      3175.28214             NaN              NaN             1.141490
min         1.00000             NaN              NaN             2.000000
25%      2750.50000             NaN              NaN             3.000000
50%      5500.00000             NaN              NaN             4.000000
75%      8249.50000             NaN              NaN             5.000000
max     10999.00000             NaN              NaN             7.000000

        Customer_rating  Cost_of_the_Product
count      10999.000000         10999.000000
unique              NaN                  NaN
top                 NaN                  NaN
freq                NaN                  NaN
mean           2.990545           210.196836
std            1.413603            48.063272
min            1.000000            96.000000
25%            2.000000           169.000000
50%            3.000000           214.000000
75%            4.000000           251.000000
max            5.000000           310.000000
        Prior_purchases Product_importance Gender  Discount_offered  \
count      10999.000000              10999  10999      10999.000000
unique              NaN                  3      2               NaN
top                 NaN                low      F               NaN
freq                NaN               5297   5545               NaN
mean           3.567597                NaN    NaN         13.373216
std            1.522860                NaN    NaN         16.205527
min            2.000000                NaN    NaN          1.000000
25%            3.000000                NaN    NaN          4.000000
50%            3.000000                NaN    NaN          7.000000
75%            4.000000                NaN    NaN         10.000000
max           10.000000                NaN    NaN         65.000000

        Weight_in_gms  Reached.on.Time_Y.N
count    10999.000000         10999.000000
unique            NaN                  NaN
top               NaN                  NaN
freq              NaN                  NaN
mean      3634.016729             0.596691
std       1635.377251             0.490584
min       1001.000000             0.000000
25%       1839.500000             0.000000
50%       4149.000000             1.000000
75%       5050.000000             1.000000
max       7846.000000             1.000000
```

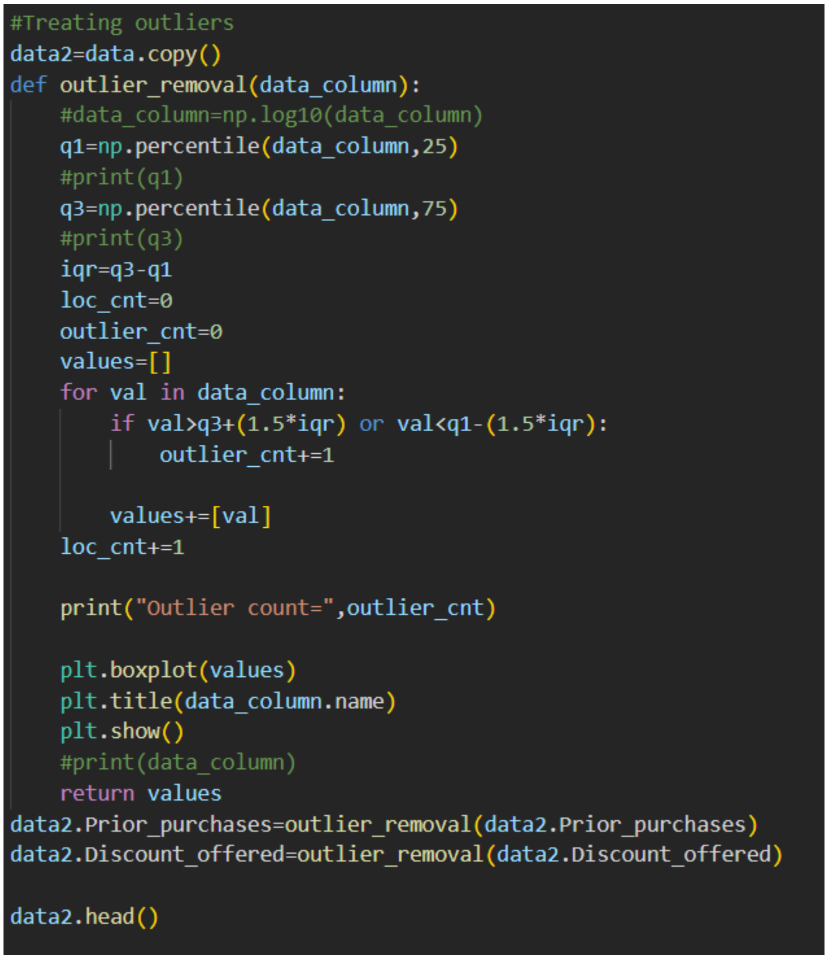


Warehouse_block
Count
0
500
1000
1500
2000
2500
3000
3500
D
F
A
B
C
Warehouse_block
Mode_of_Shipment
Count
0
1000
2000
3000
4000
5000
6000
7000
Flight
Ship
Road
Mode_of_Shipment
Product_importance
Count
0
1000
2000
3000
4000
5000
low
medium
high
Product_importance
Gender
Count
0
1000
2000
3000
4000
5000
F
M
Gender

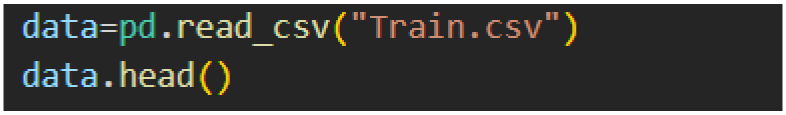
Multivariate Analysis	
Outliers and Anomalies	

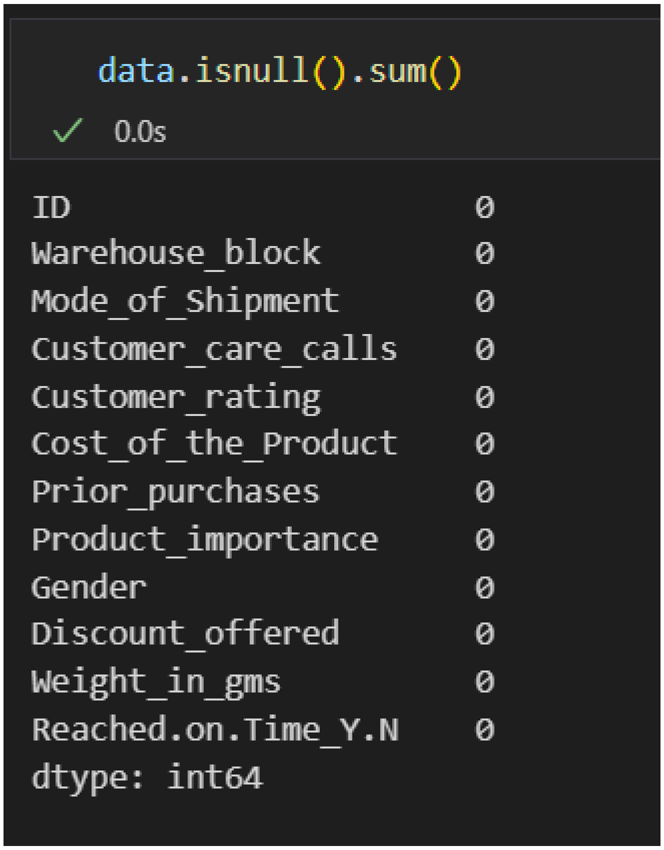


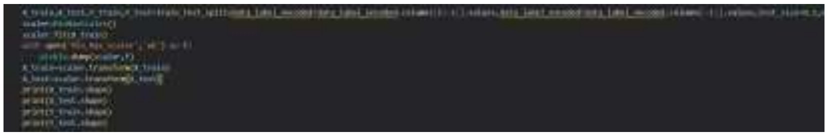


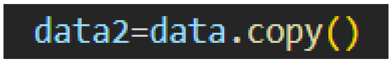
```
#Treating outliers
data2=data.copy()
def outlier_removal(data_column):
    #data_column=np.log10(data_column)
    q1=np.percentile(data_column,25)
    #print(q1)
    q3=np.percentile(data_column,75)
    #print(q3)
    iqr=q3-q1
    loc_cnt=0
    outlier_cnt=0
    values=[]
    for val in data_column:
        if val>q3+(1.5*iqr) or val<q1-(1.5*iqr):
            outlier_cnt+=1

        values+=[val]
    loc_cnt+=1

    print("Outlier count=",outlier_cnt)

    plt.boxplot(values)
    plt.title(data_column.name)
    plt.show()
    #print(data_column)
    return values
data2.Prior_purchases=outlier_removal(data2.Prior_purchases)
data2.Discount_offered=outlier_removal(data2.Discount_offered)

data2.head()
```

Data Preprocessing Code Screenshots	

Loading Data	
	`data=pd.read_csv("Train.csv")`
	`data.head()`

Handling Missing Data	
	`data.isnull().sum()`
	✓ 0.0s
	`ID                     0`
	`Warehouse_block        0`
	`Mode_of_Shipment       0`
	`Customer_care_calls    0`
	`Customer_rating        0`
	`Cost_of_the_Product    0`
	`Prior_purchases        0`
	`Product_importance     0`
	`Gender                 0`
	`Discount_offered       0`
	`Weight_in_gms          0`
	`Reached.on.Time_Y.N    0`
	`dtype: int64`

Data Transformation	
	`X_train,X_test,y_train,y_test=train_test_split(` [illegible]
	`scaler=StandardScaler()`
	`scaler.fit(X_train)`
	`with open(` [illegible] `,'wb') as f:`
	`    pickle.dump(scaler,f)`
	`X_train=scaler.transform(X_train)`
	`X_test=scaler.transform(X_test)`
	`print(X_train.shape)`
	`print(X_test.shape)`
	`print(y_train.shape)`
	`print(y_test.shape)`

Save Processed Data	 `data2=data.copy()`