




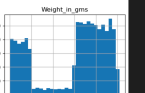



## Data Collection and Preprocessing Phase

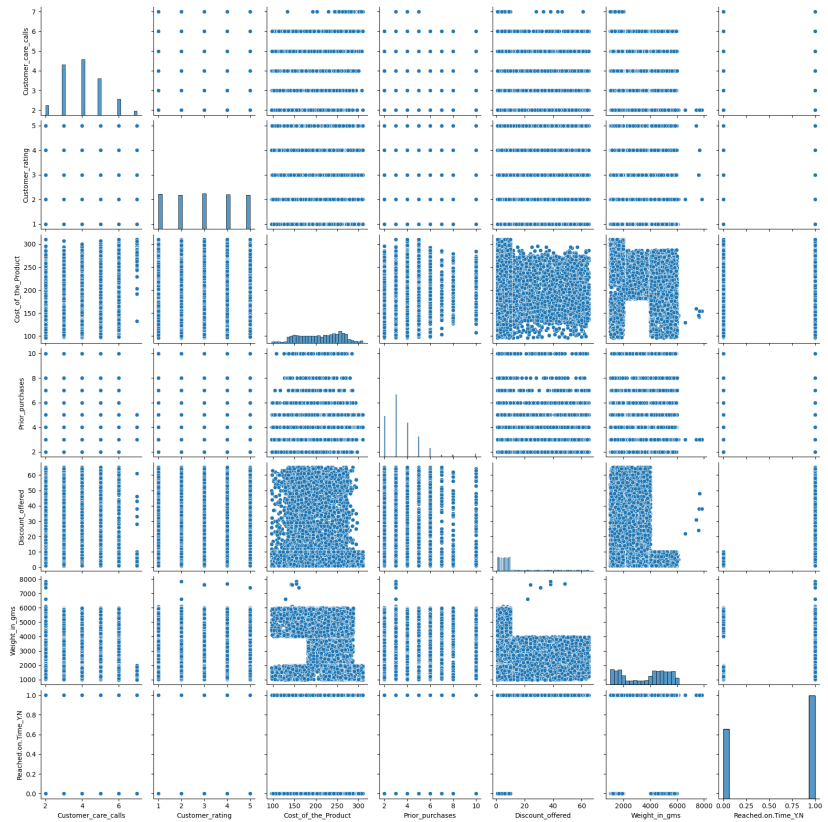
Date	15 March 2024
Team ID	SWTID1720161281
Project Title	Ecommerce Shipping Prediction Using ML
Maximum Marks	6 Marks

## Data Exploration and Preprocessing Template

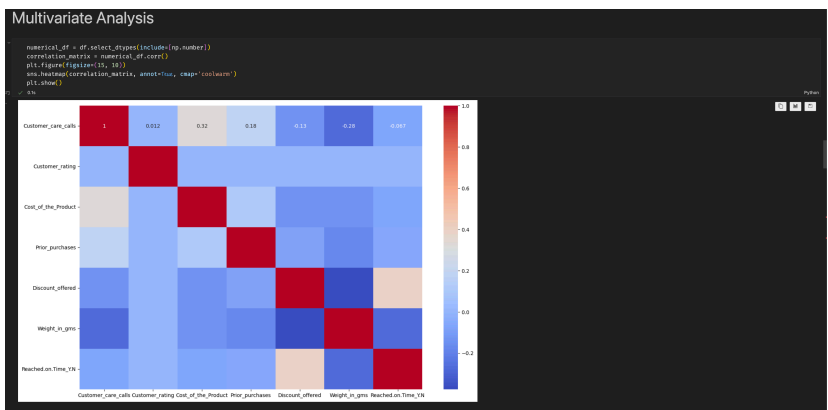
Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

Section	Description
Data Overview	<div><div>DATA COLLECTION AND OVERVIEW</div><pre>df=pd.read_csv('Train.csv') df.head()  # In [ ]:  ID Warehouse_block Mode_of_Shipment Customer_care_calls Customer_rating Cost_of_the_Product Prior_purchases Product_Importance Gender Discount_offered Weight_in_gms Reached_on_Time_Y/N 0 1 D Flight 4 2 177 5 low F 44 1233 1 1 2 F Flight 4 5 276 2 low M 59 3088 1 2 3 A Flight 2 2 183 4 low M 48 3374 1 3 4 B Flight 3 3 176 4 medium M 50 1177 1 4 5 C Flight 2 2 184 3 medium F 46 2484 1  print(df.info())  # In [ ]:  &lt;class 'pandas.core.frame.DataFrame'&gt; Int64Index: 8737 entries, 3 to 10888 Data columns (total 12 columns): # Column Non-Null Count Dtype --- 0 Warehouse_block 8737 non-null object 1 Mode_of_Shipment 8737 non-null object 2 Customer_care_calls 8737 non-null int64 3 Customer_rating 8737 non-null int64 4 Cost_of_the_Product 8737 non-null int64 5 Prior_purchases 8737 non-null int64 6 Product_Importance 8737 non-null object 7 Discount_offered 8737 non-null int64 8 Weight_in_gms 8737 non-null int64 9 Reached_on_Time_Y/N 8737 non-null int64 dtypes: int64(7), object(5) memory usage: 1068.5+ KB  None  print(df.describe())  # In [ ]:  Customer_care_calls Customer_rating Cost_of_the_Product \ count 8737.000000 8737.000000 8737.000000 mean 4.232221 2.292574 212.611353 std 1.364629 1.489653 45.098477 min 2.000000 2.000000 76.000000 25% 3.000000 2.000000 175.000000 50% 4.000000 2.000000 221.000000 75% 5.000000 4.000000 254.000000 max 7.000000 5.000000 315.000000</pre></div>
Univariate Analysis	<div><div>Univariate Analysis</div><pre>df.select_dtypes(include=[np.number]).hist(bins=25, figsize=(15, 10)) fig.show()</pre><div><div># In [ ]:</div><div><div>Customer_care_calls</div><div>Customer_rating</div><div>Cost_of_the_Product</div><div>Prior_purchases</div><div>Discount_offered</div><div>Weight_in_gms</div><div>Reached_on_Time_Y/N</div></div></div></div>

## Bivariate Analysis

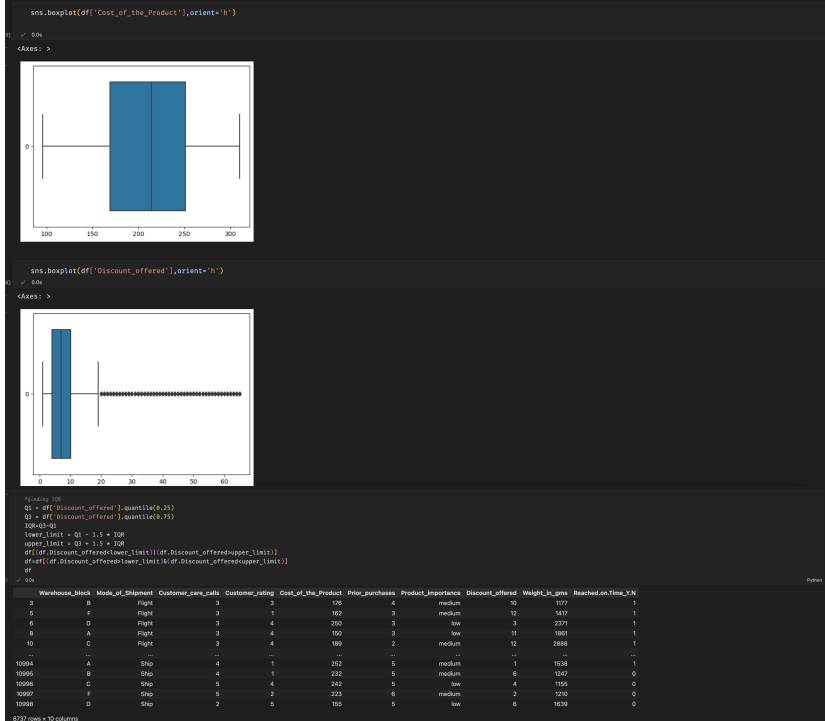


## Multivariate Analysis



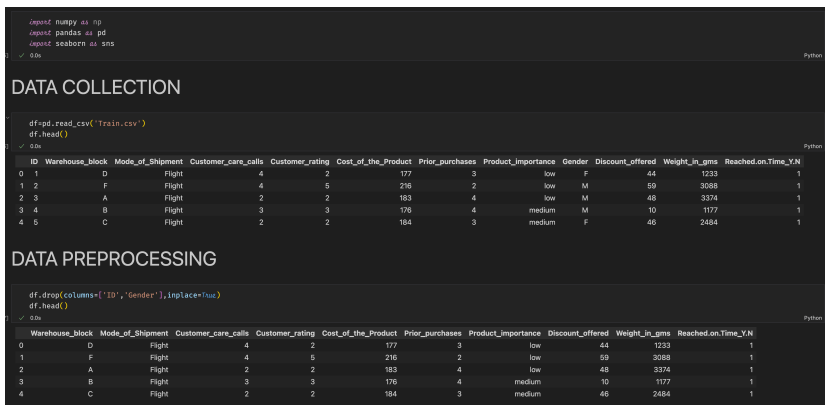
## Outliers and Anomalies

### CHECKING AND REMOVING OUTLIERS

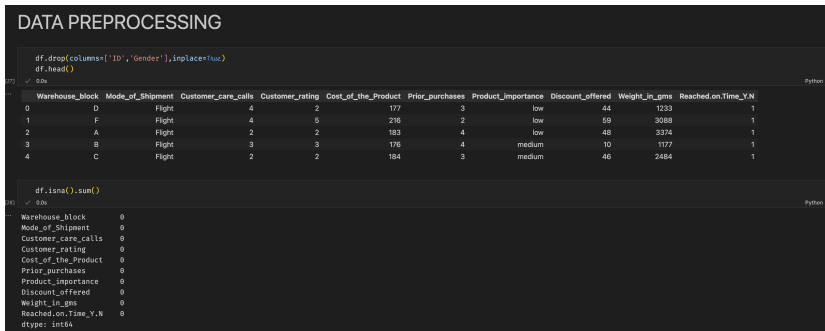


## Data Preprocessing Code Screenshots

### Loading Data



### Handling Missing Data



## Data Transformation

### DATA ENCODING

```
from sklearn.preprocessing import LabelEncoder  
le=LabelEncoder()  
X['Warehouse_Units']=le.fit_transform(X['Warehouse_Units'])  
X['Mode_of_Shipment']=le.fit_transform(X['Mode_of_Shipment'])  
X['Product_Importance']=le.fit_transform(X['Product_Importance'])  
X.head()
```

Warehouse_Units	Mode_of_Shipment	Product_Importance
3	1	0
5	4	0
6	3	0
8	0	0
10	2	0

### SMOTE FOR DATA HANDLING

```
from imblearn.over_sampling import SMOTE  
smote=SMOTE()  
X_resampled=fit_resample(X,y)  
X_resampled
```

Warehouse_Units	Mode_of_Shipment	Product_Importance
3	1	0
5	4	0
6	3	0
8	0	0
10	2	0

### DATA SCALING

```
from sklearn.preprocessing import StandardScaler  
sc=StandardScaler()  
X_rescaled=fit_transform(X,y)  
X_rescaled
```

Warehouse_Units	Mode_of_Shipment	Product_Importance
3	1	0
5	4	0
6	3	0
8	0	0
10	2	0

## Feature Engineering

-NA-

## Save Processed Data

### Save Processed Data

```
num_features = df.select_dtypes(include=[np.number]).columns  
cat_features = df.select_dtypes(include=[object]).columns  
  
num_transformer = Pipeline(steps=[  
    ('imputer', SimpleImputer(strategy='mean')),  
    ('scaler', StandardScaler())  
)  
  
cat_transformer = Pipeline(steps=[  
    ('imputer', SimpleImputer(strategy='most_frequent')),  
    ('onehot', OneHotEncoder(handle_unknown='ignore'))  
)  
  
preprocessor = ColumnTransformer(  
    transformers=[  
        ('num', num_transformer, num_features),  
        ('cat', cat_transformer, cat_features)  
    ]  
)  
  
df_processed = preprocessor.fit_transform(df)  
processed_df = pd.DataFrame(df_processed)  
processed_df.to_csv('processed_dataset.csv', index=False)  
print("Processed data saved as 'processed_dataset.csv'")  
  
new_data = pd.read_csv('processed_dataset.csv')  
new_data.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	-0.047771	-0.700785	-0.800722	-0.372736	1.088883	-1.048440	0.822138	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
1	-0.047771	1.427176	0.102186	-0.026424	2.819836	-0.338893	0.822138	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
2	-1.798887	-0.700785	-0.368881	0.281854	1.158824	-0.160022	0.822138	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	-0.822736	0.046668	-0.779139	-0.283854	-0.305892	-1.020486	0.822138	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	-1.798887	-0.700785	-0.368881	-0.372736	2.013404	-0.703444	0.822138	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0