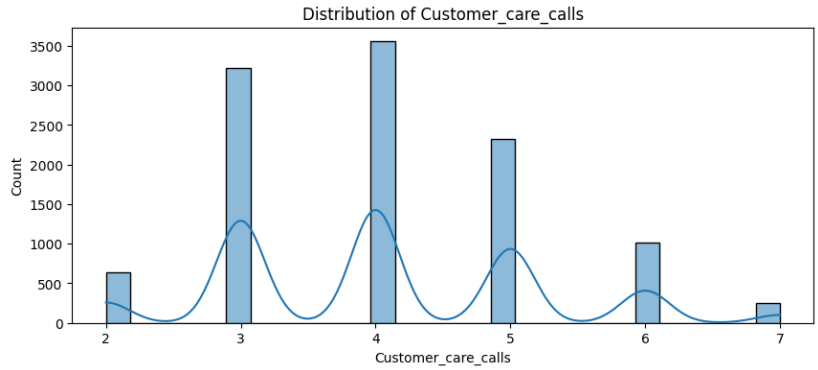
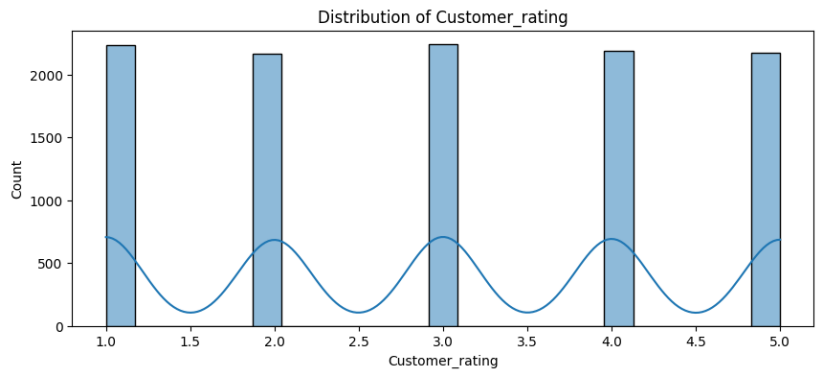
Data Collection and Preprocessing Phase

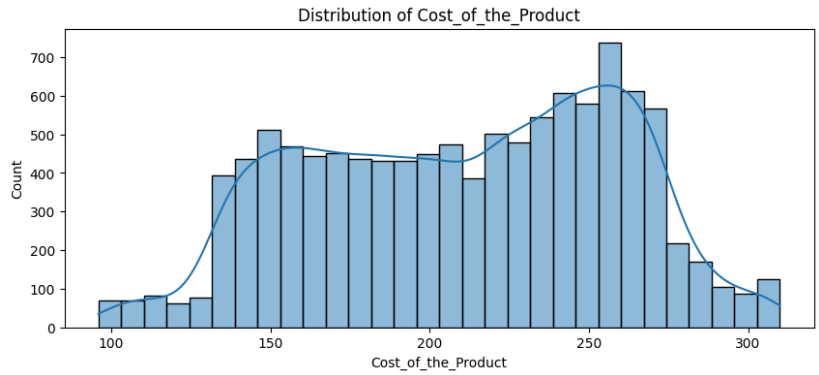
Date	20 July 2024
Team ID	SWTID1720110595
Project Title	Ecommerce Shipping Prediction Using Machine Learning
Maximum Marks	6 Marks

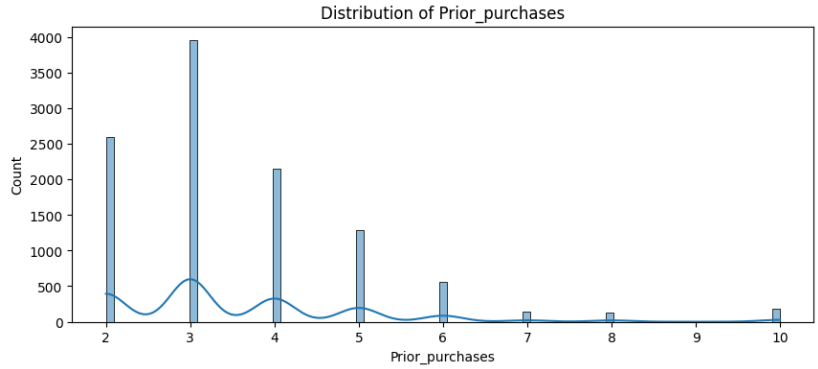
Data Exploration and Preprocessing Template

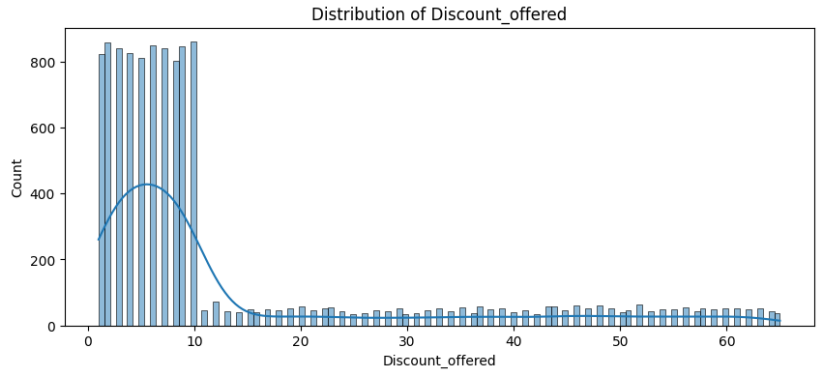
Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

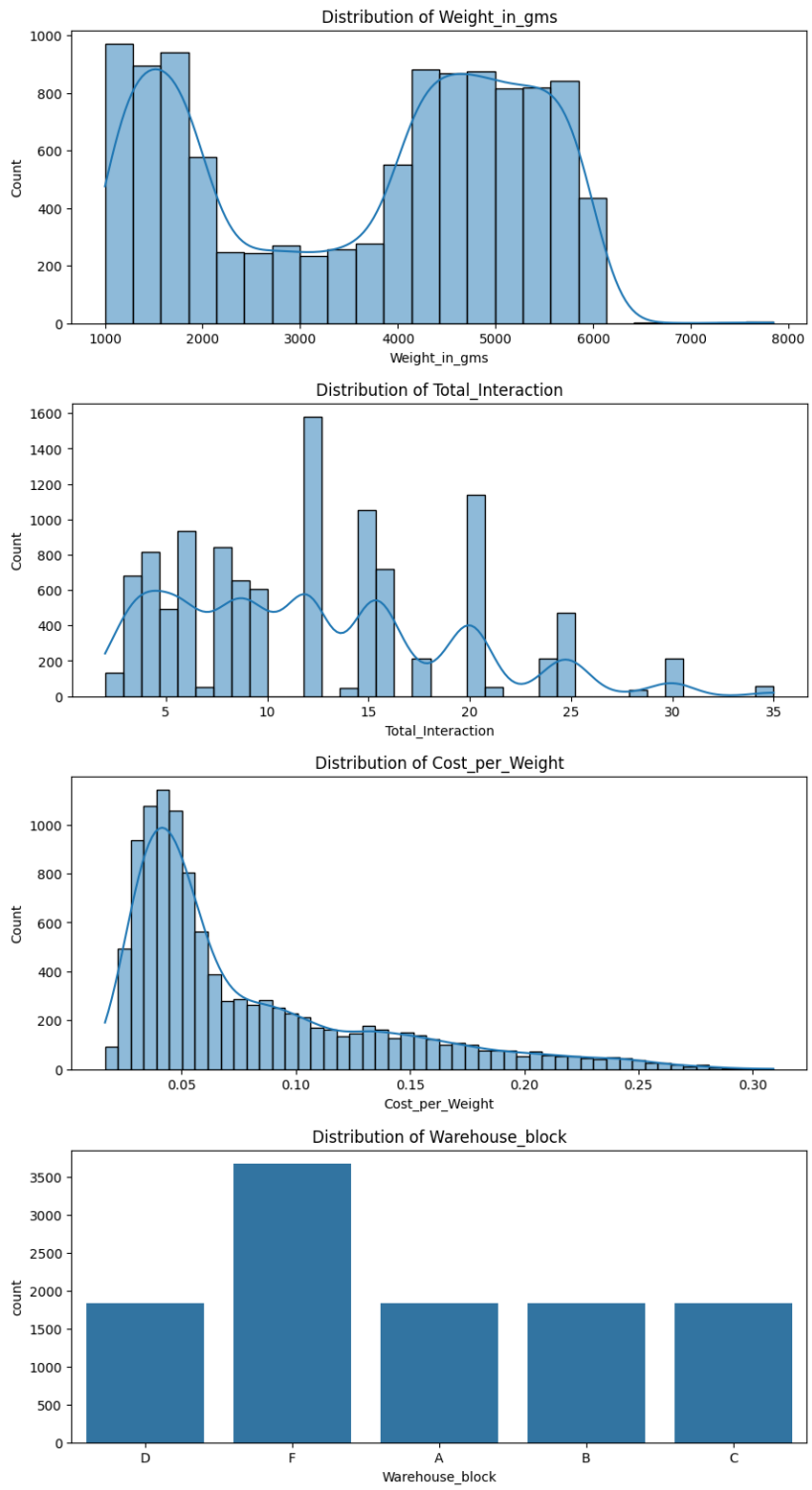
Section	Description
Data Overview	... ID Customer_care_calls Customer_rating Cost_of_the_Product \ count 10999.00000 10999.000000 10999.000000 10999.000000 mean 5500.00000 4.054459 2.990545 210.196836 std 3175.28214 1.141490 1.413603 48.063272 min 1.00000 2.000000 1.000000 96.000000 25% 2750.50000 3.000000 2.000000 169.000000 50% 5500.00000 4.000000 3.000000 214.000000 75% 8249.50000 5.000000 4.000000 251.000000 max 10999.00000 7.000000 5.000000 310.000000 Prior_purchases Discount_offered Weight_in_gms Reached.on.Time_Y.N count 10999.000000 10999.000000 10999.000000 10999.000000 mean 3.567597 13.373216 3634.016729 0.596691 std 1.522860 16.205527 1635.377251 0.490584 min 2.000000 1.000000 1001.000000 0.000000 25% 3.000000 4.000000 1839.500000 0.000000 50% 3.000000 7.000000 4149.000000 1.000000 75% 4.000000 10.000000 5050.000000 1.000000 max 10.000000 65.000000 7846.000000 1.000000
Univariate Analysis	

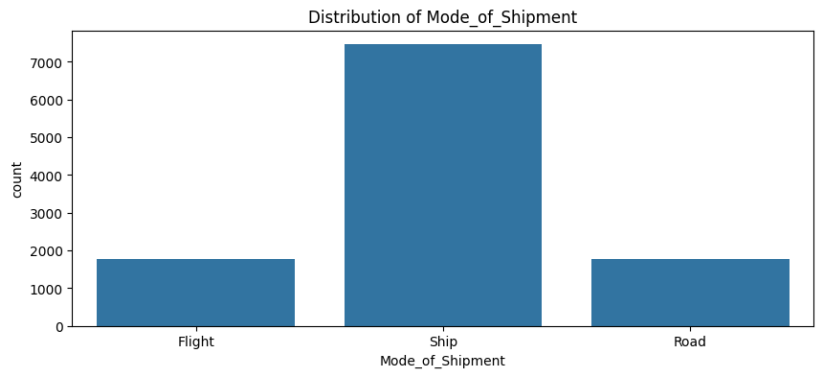

Distribution of Customer_rating
Count
Customer_rating

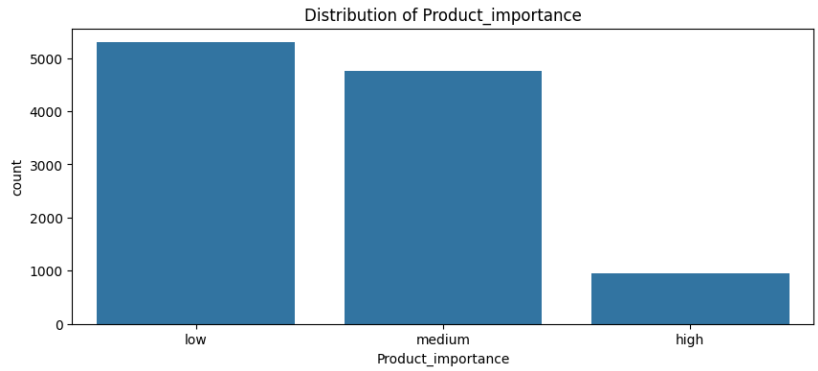

Distribution of Cost_of_the_Product
Count
Cost_of_the_Product

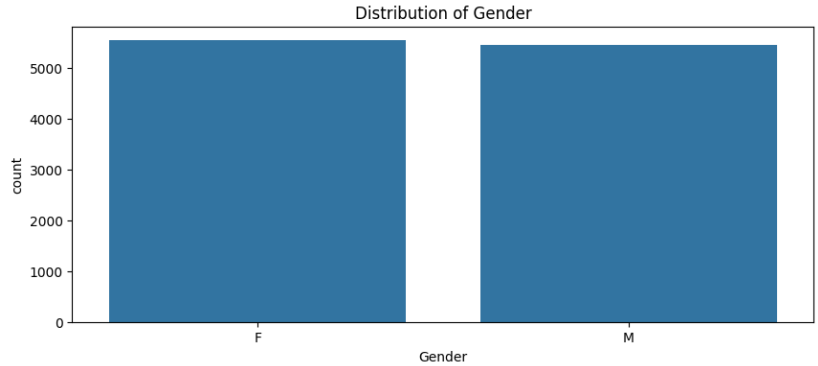

Distribution of Prior_purchases
Count
Prior_purchases

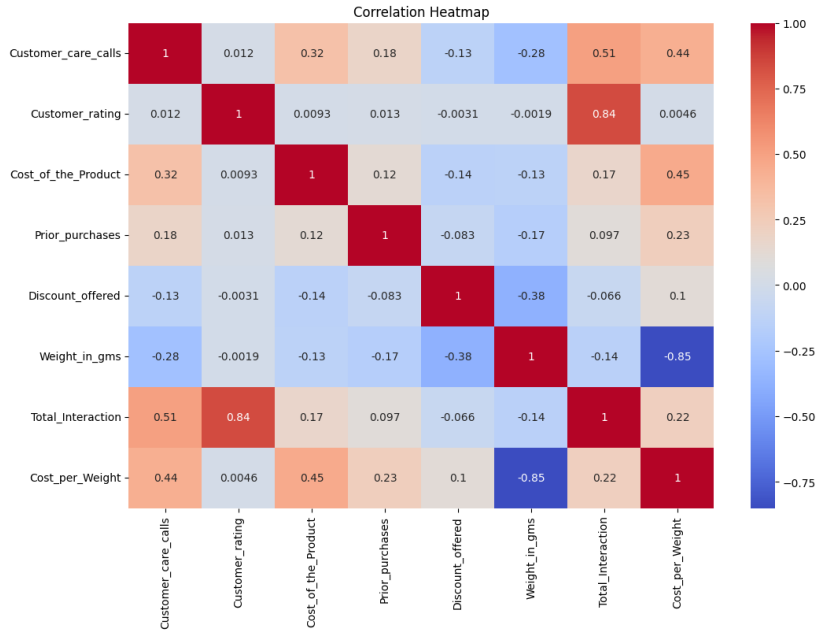
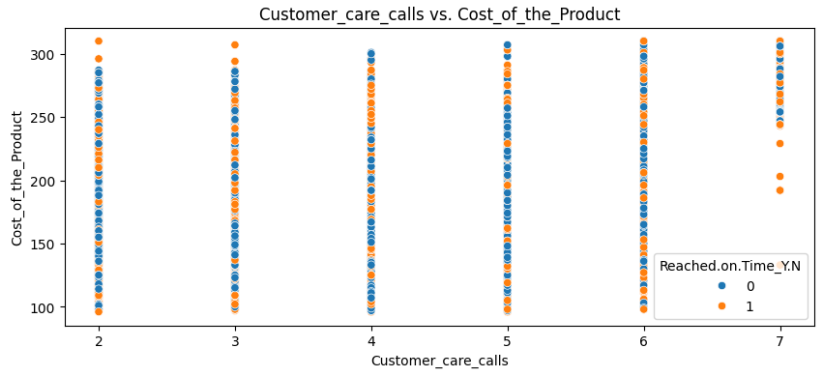
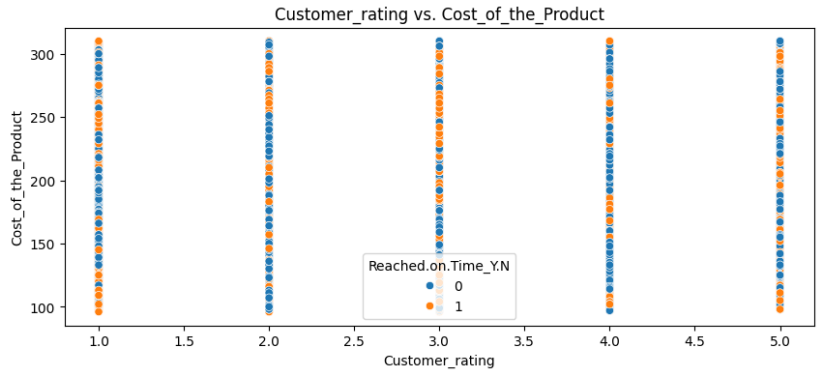

Distribution of Discount_offered
Count
Discount_offered

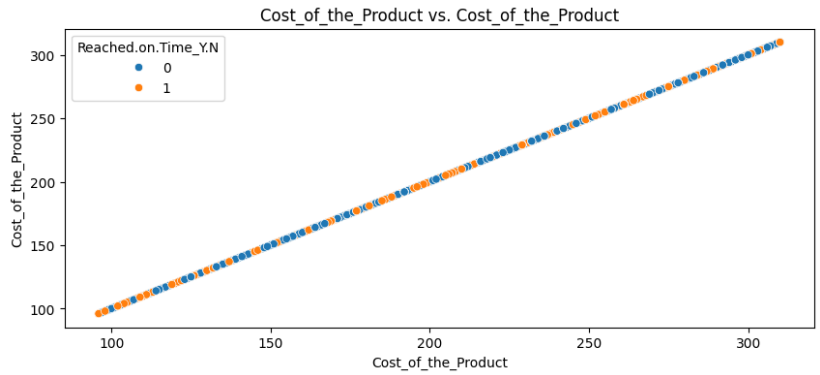

Distribution of Weight_in_gms
Count
Weight_in_gms
Distribution of Total_Interaction
Count
Total_Interaction
Distribution of Cost_per_Weight
Count
Cost_per_Weight
Distribution of Warehouse_block
count
D
F
A
B
C
Warehouse_block

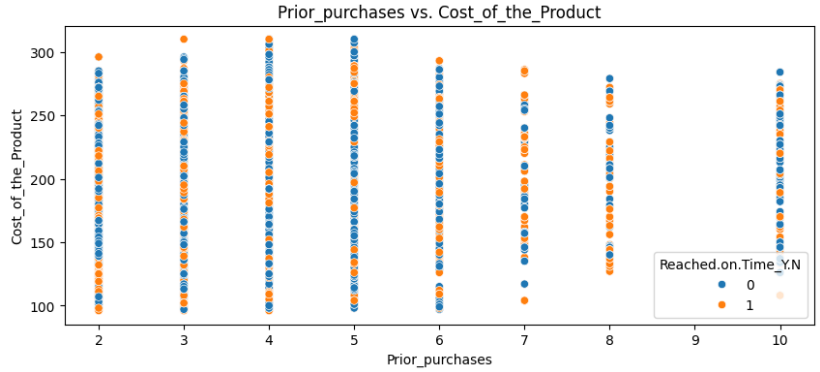

Distribution of Mode_of_Shipment
count
0
1000
2000
3000
4000
5000
6000
7000
Flight
Ship
Road
Mode_of_Shipment


Distribution of Product_importance
count
0
1000
2000
3000
4000
5000
low
medium
high
Product_importance

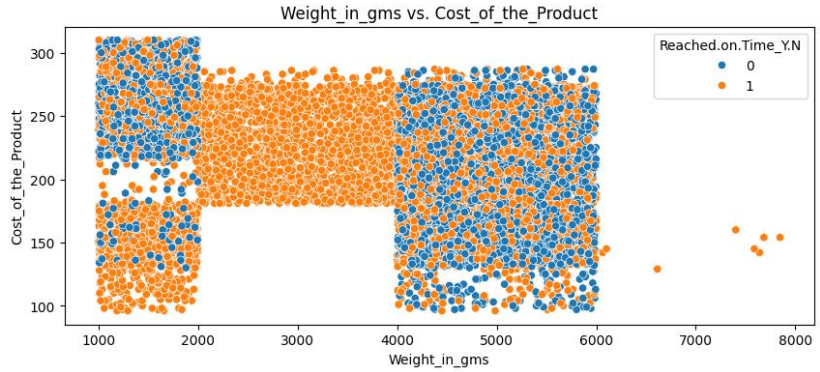

Distribution of Gender
count
0
1000
2000
3000
4000
5000
F
M
Gender

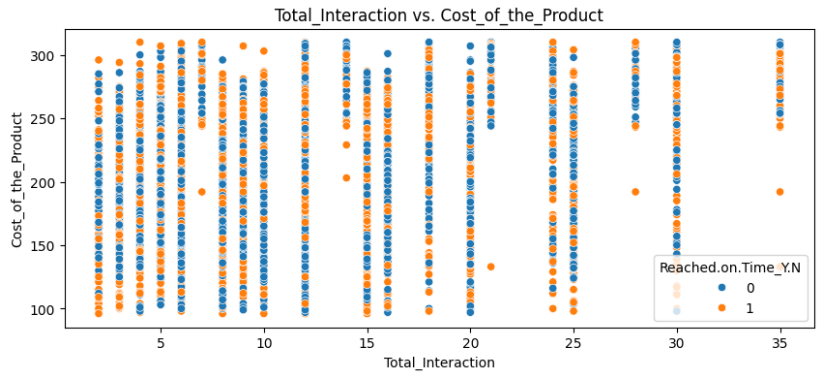
Bivariate Analysis	  

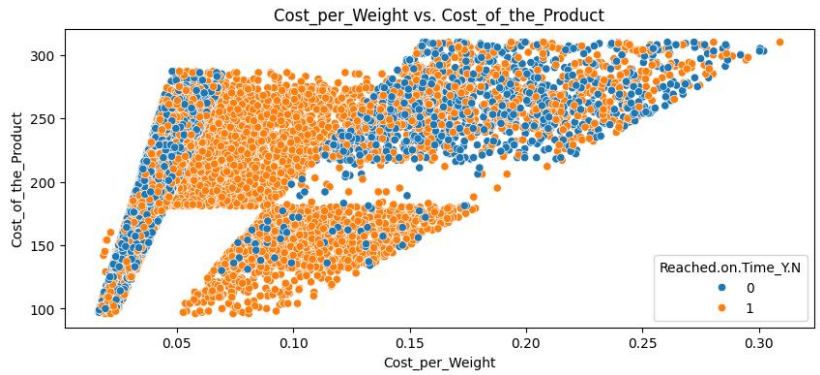

Cost_of_the_Product vs. Cost_of_the_Product
Reached.on.Time_Y.N
0
1
Cost_of_the_Product
Cost_of_the_Product

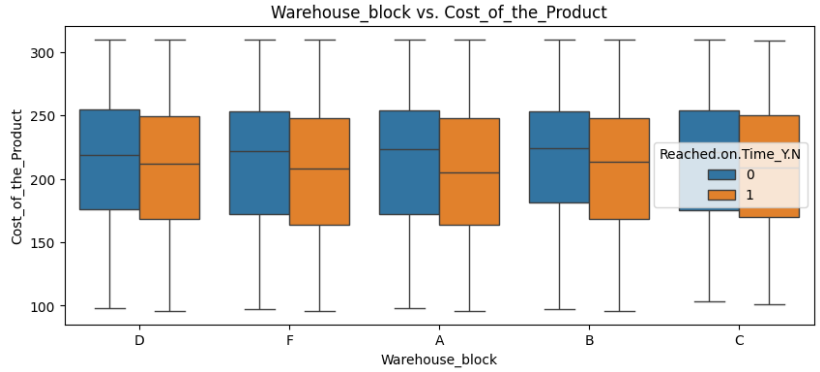

Prior_purchases vs. Cost_of_the_Product
Reached.on.Time_Y.N
0
1
Cost_of_the_Product
Prior_purchases

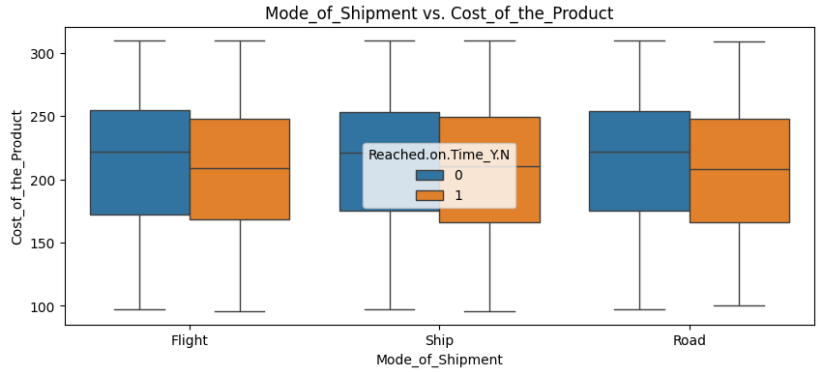

Discount_offered vs. Cost_of_the_Product
Reached.on.Time_Y.N
0
1
Cost_of_the_Product
Discount_offered

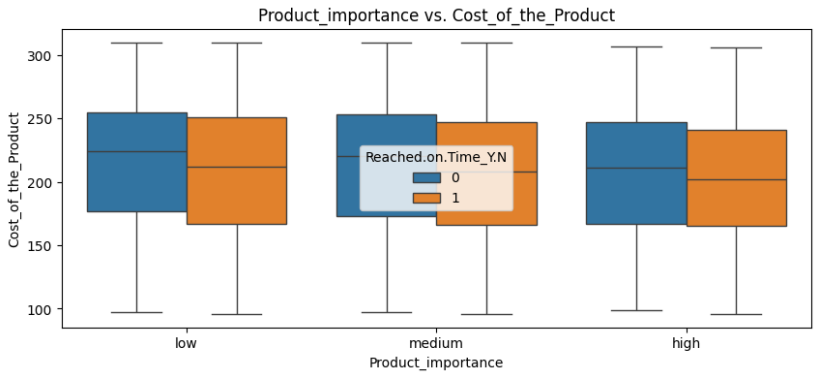
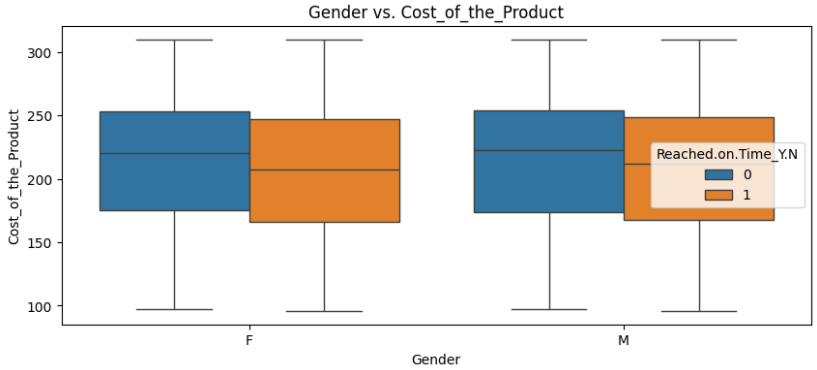
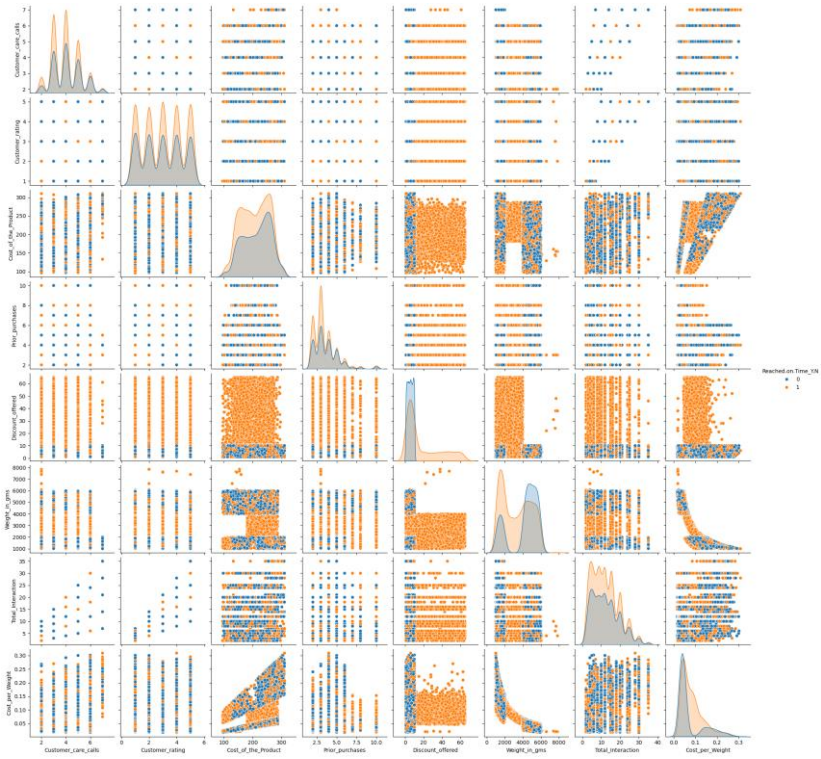

Weight_in_gms vs. Cost_of_the_Product
Reached.on.Time_Y.N
0
1
Cost_of_the_Product
Weight_in_gms


Total_Interaction vs. Cost_of_the_Product
Cost_of_the_Product
Total_Interaction
Reached.on.Time_Y.N
0
1


Cost_per_Weight vs. Cost_of_the_Product
Cost_of_the_Product
Cost_per_Weight
Reached.on.Time_Y.N
0
1


Warehouse_block vs. Cost_of_the_Product
Cost_of_the_Product
Warehouse_block
D
F
A
B
C
Reached.on.Time_Y.N
0
1


Mode_of_Shipment vs. Cost_of_the_Product
Cost_of_the_Product
Mode_of_Shipment
Flight
Ship
Road
Reached.on.Time_Y.N
0
1

	 
Multivariate Analysis	

Outliers and Anomalies

```python
# Outliers and Anomalies
z_scores = np.abs(stats.zscore(data[numerical_features]))
outliers = np.where(z_scores > 3)
print("Outliers detected:", outliers)

for feature in numerical_features:
    plt.figure(figsize=(10, 4))
    sns.boxplot(data[feature])
    plt.title(f'Boxplot of {feature}')
    plt.show()

data_cleaned = data[(z_scores < 3).all(axis=1)]
```

Data Preprocessing Code Screenshots

Loading Data

```python
# Load dataset
data = pd.read_csv('Train.csv')
data
```

[2] ✓ 0.0s Python

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discount
0	1	D	Flight	4	2	177	3	low	F	
1	2	F	Flight	4	5	216	2	low	M	
2	3	A	Flight	2	2	183	4	low	M	
3	4	B	Flight	3	3	176	4	medium	M	
4	5	C	Flight	2	2	184	3	medium	F	
...	...	...	...	...	...	...	...	...	...	
10994	10995	A	Ship	4	1	252	5	medium	F	
10995	10996	B	Ship	4	1	232	5	medium	F	
10996	10997	C	Ship	5	4	242	5	low	F	
10997	10998	F	Ship	5	2	223	6	medium	M	
10998	10999	D	Ship	2	5	155	5	low	F	

10999 rows × 12 columns

Handling Missing Data

```python
# Data preprocessing
data = data.fillna(method='ffill')
```

Data Transformation

```python
# Define numerical and categorical features
numerical_features = ['Customer_care_calls', 'Customer_rating', 'Cost_of_the_Product', 'Prior_purchases', 'Discount_offered', 'Weight_in_gms', 'Total_In
categorical_features = ['Warehouse_block', 'Mode_of_Shipment', 'Product_importance', 'Gender']

numerical_transformer = StandardScaler()
categorical_transformer = OneHotEncoder(handle_unknown='ignore', drop='first')

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Preprocess data before applying SMOTE
X_preprocessed = preprocessor.fit_transform(X)
```

Feature Engineering

```python
# Feature engineering
data['Total_Interaction'] = data['Customer_care_calls'] * data['Customer_rating']
data['Cost_per_Weight'] = data['Cost_of_the_Product'] / data['Weight_in_gms']
```

Save Processed Data	```python
	# Serialize the preprocessor
	with open('preprocessor.pkl', 'wb') as f:
	    pickle.dump(preprocessor, f)
	
	# Serialize the trained stacking pipeline
	with open('model.pkl', 'wb') as f:
	    pickle.dump(stacking_pipeline, f)
	``