Data Collection and Preprocessing Phase

Date	09 July 2024
Team ID	SWTID1720084775
Project Title	Ecommerce Shipping Prediction Using Machine Learning
Maximum Marks	6 Marks

Data Exploration and Preprocessing

Identifying data sources, assessing quality issues like missing values and duplicates, and implementing resolution plans to ensure accurate and reliable analysis.

Section	Description

Data Overview	`data.info()`
	`<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)`
	`memory usage: 1.0+ MB`

```
data.isnull().sum()

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
```

Univariate Analysis

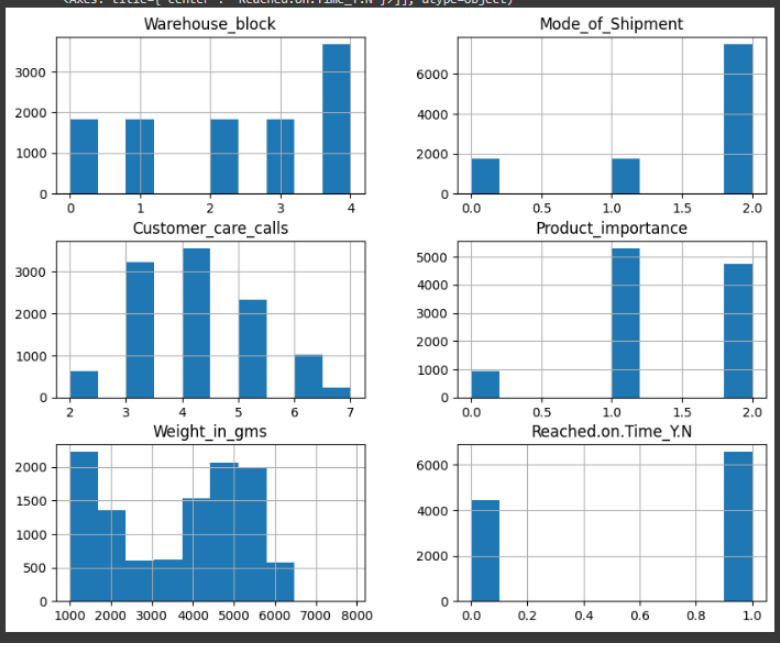
```
data.describe()
```

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

<table>
<tr>
<td></td>
<td>

```
[33] data_updated.hist(figsize=(10,8))
```

```
array([[<Axes: title={'center': 'Warehouse_block'}>,
        <Axes: title={'center': 'Mode_of_Shipment'}>],
       [<Axes: title={'center': 'Customer_care_calls'}>,
        <Axes: title={'center': 'Product_importance'}>],
       [<Axes: title={'center': 'Weight_in_gms'}>,
        <Axes: title={'center': 'Reached.on.Time_Y.N'}>]], dtype=object)
```



</td>
</tr>
<tr>
<td>Bivariate Analysis</td>
<td>

```
data.corr()
```

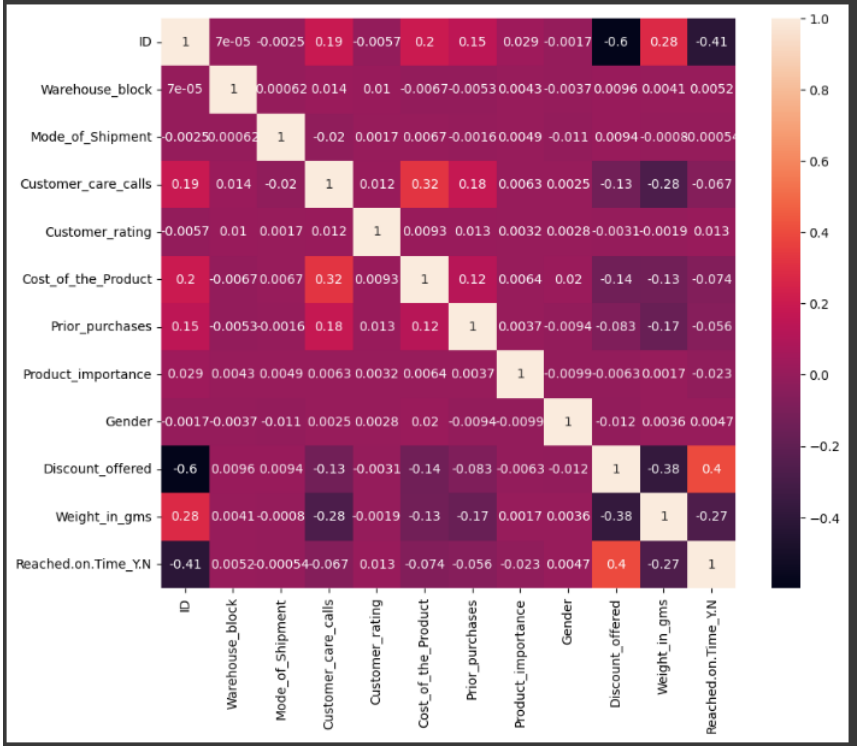
	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discount_offered	Weight_in_gms	Reached.on.Time_Y.N
ID	1.000000	0.000070	-0.002459	0.188998	-0.005722	0.196791	0.145369	0.029081	-0.001695	-0.598278	0.278312	-0.411822
Warehouse_block	0.000070	1.000000	0.000617	0.014496	0.010169	-0.006679	-0.005262	0.004260	-0.003700	0.009569	0.004086	0.005214
Mode_of_Shipment	-0.002459	0.000617	1.000000	-0.020164	0.001679	0.006681	-0.001640	0.004911	-0.011288	0.009364	-0.000797	-0.000535
Customer_care_calls	0.188998	0.014496	-0.020164	1.000000	0.012209	0.323182	0.180771	0.006273	0.002545	-0.130750	-0.276615	-0.067126
Customer_rating	-0.005722	0.010169	0.001679	0.012209	1.000000	0.009270	0.013179	0.003157	0.002775	-0.003124	-0.001897	0.013119
Cost_of_the_Product	0.196791	-0.006679	0.006681	0.323182	0.009270	1.000000	0.123676	0.006366	0.019759	-0.138312	-0.132604	-0.073587
Prior_purchases	0.145369	-0.005262	-0.001640	0.180771	0.013179	0.123676	1.000000	0.003662	-0.009395	-0.082769	-0.168213	-0.055515
Product_importance	0.029081	0.004260	0.004911	0.006273	0.003157	0.006366	0.003662	1.000000	-0.009865	-0.006251	0.001652	-0.023483
Gender	-0.001695	-0.003700	-0.011288	0.002545	0.002775	0.019759	-0.009395	-0.009865	1.000000	-0.011777	0.003573	0.004689
Discount_offered	-0.598278	0.009569	0.009364	-0.130750	-0.003124	-0.138312	-0.082769	-0.006251	-0.011777	1.000000	-0.376067	0.397108
Weight_in_gms	0.278312	0.004086	-0.000797	-0.276615	-0.001897	-0.132604	-0.168213	0.001652	0.003573	-0.376067	1.000000	-0.268793
Reached.on.Time_Y.N	-0.411822	0.005214	-0.000535	-0.067126	0.013119	-0.073587	-0.055515	-0.023483	0.004689	0.397108	-0.268793	1.000000

</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>

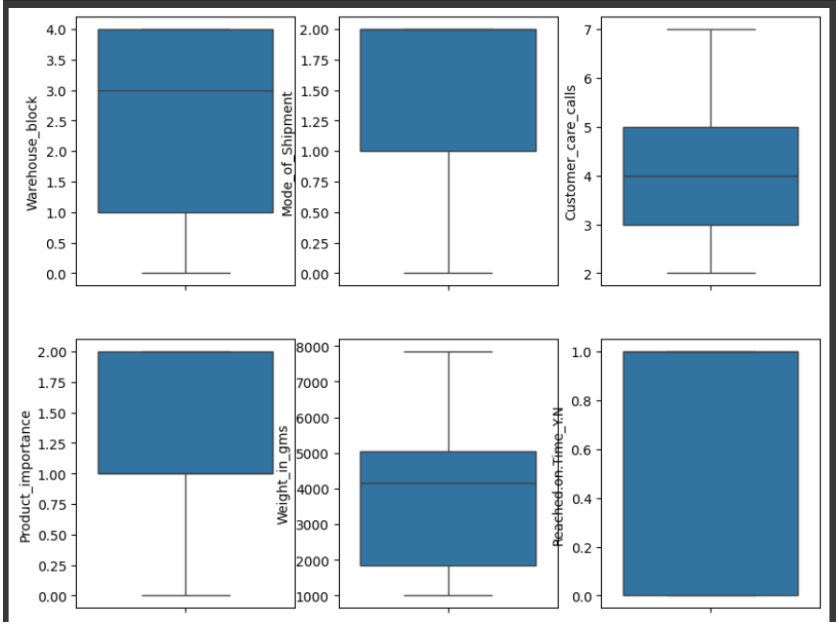
```
plt.figure(figsize=(10,8))
sns.heatmap(data.corr(),annot=True)
```

```
<Axes: >
```



</td>
</tr>
<tr>
<td>Multivariate Analysis</td>
<td>Patterns and relationships involving multiple variables.</td>
</tr>
<tr>
<td>Outliers and Anomalies</td>
<td>

```
plt.figure(figsize=(10,8))
for i in data_updated.columns:
  plt.subplot(2,3,data_updated.columns.get_loc(i)+1)
  sns.boxplot(data_updated[i])
```



</td>
</tr>
</table>

Data Preprocessing Code Screenshots

Loading Data

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle
```

```
cp: cannot stat 'kaggle.json': No such file or directory
```

```
!kaggle datasets download -d prachi13/customer-analytics
```

```
Dataset URL: https://www.kaggle.com/datasets/prachi13/customer-analytics
License(s): other
Downloading customer-analytics.zip to /content
100% 121k/121k [00:00<00:00, 308kB/s]
100% 121k/121k [00:00<00:00, 308kB/s]
```

```
!unzip '/content/customer-analytics.zip'
```

```
Archive:  /content/customer-analytics.zip
  inflating: Train.csv
```

```
data = pd.read_csv(r"/content/Train.csv")
```

Handling Missing Data

There was no missing data hence no handling.

Data Transformation

Scaling:

```
[49] scaler = StandardScaler()
     X_scaled = pd.DataFrame(scaler.fit_transform(x), columns=x.columns, index=x.index)
```

```
[50] X_scaled
```

	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Product_importance	Weight_in_gms
0	0.447189	-2.004158	-0.047711	-0.548034	-1.468240
1	1.118034	-2.004158	-0.047711	-0.548034	-0.333893
2	-1.565345	-2.004158	-1.799887	-0.548034	-0.159002
3	-0.894500	-2.004158	-0.923799	1.035735	-1.502484
4	-0.223656	-2.004158	-1.799887	1.035735	-0.703244
...	...	...	...	...	...
10994	-1.565345	0.638342	-0.047711	1.035735	-1.281730
10995	-0.894500	0.638342	-0.047711	1.035735	-1.459679
10996	-0.223656	0.638342	0.828377	-0.548034	-1.515937
10997	1.118034	0.638342	0.828377	1.035735	-1.482304
10998	0.447189	0.638342	-1.799887	-0.548034	-1.219968

10999 rows × 5 columns

Normalising:

```
[51] pca = PCA()
     pca_result = pca.fit_transform(X_scaled)
```

```
[52] pca_result
```

```
array([[ 1.10475734,  1.19348539,  1.57132482,  0.39776473,  1.18439716],
       [ 0.32348709,  0.85091912,  2.15927308,  0.12300214,  0.41767556],
       [-1.11073268,  2.2181348 ,  0.19559807,  1.3642948 ,  1.40668505],
       ...,
       [ 1.61045377,  0.16284354, -0.7027347 , -0.66871241,  0.42547053],
       [ 1.64046499, -1.61189307,  0.2361085 , -0.14675122,  0.49704411],
       [-0.43944655, -0.14981796, -0.2207732 , -0.96779987,  2.10589153]])
```

Balancing:

```
[36] us = RandomUnderSampler(random_state=0)
```

```
[37] x = data_updated.drop(['Reached.on.Time_Y.N'], axis=1)
```

```
[38] y = data_updated['Reached.on.Time_Y.N']
```

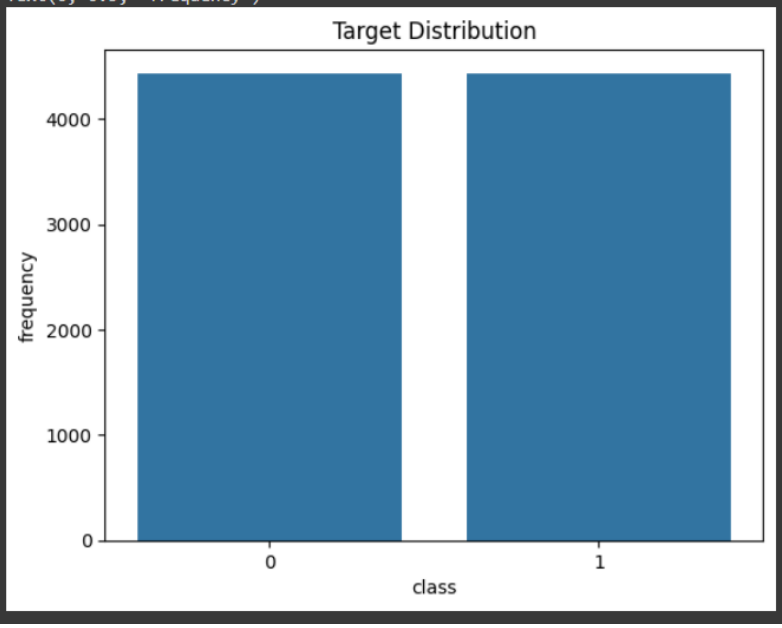
```
[39] x_res, y_res=us.fit_resample(x, y)
```

```
[40] x_res.head()
```

	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Product_importance	Weight_in_gms
0	1	1	6	2	5031
1	2	1	3	1	5956
2	4	1	4	0	4245
3	3	1	4	2	4622
4	0	1	3	2	4732

```
[43] sns.countplot(x=y_res,data=data_updated)
     plt.title('Target Distribution')
     plt.xlabel('class')
     plt.ylabel('frequency')
```

```
Text(0, 0.5, 'frequency')
```



<table>
<tr>
<td>Feature Engineering</td>
<td>Encoding:

```
[19] le = LabelEncoder()

[46] data['Warehouse_block']=le.fit_transform(data['Warehouse_block'])
     data['Mode_of_Shipment']=le.fit_transform(data['Mode_of_Shipment'])
     data['Product_importance']=le.fit_transform(data['Product_importance'])
     data['Gender']=le.fit_transform(data['Gender'])

[47] data.head()
```

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender
0	1	3	0	4	2	177	3	1	0
1	2	4	0	4	5	216	2	1	1
2	3	0	0	2	2	183	4	1	1
3	4	1	0	3	3	176	4	2	1
4	5	2	0	2	2	184	3	2	0

Dropping Unnecessary Features:

```
[30] data_updated = data.drop(['ID','Gender','Discount_offered','Cost_of_the_Product','Prior_purchases','Customer_rating'],axis=1)

[31] data_updated.head()
```

	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Product_importance	Weight_in_gms	Reached.on.Time_Y.N
0	3	0	4	1	1233	1
1	4	0	4	1	3088	1
2	0	0	2	1	3374	1
3	1	0	3	2	1177	1
4	2	0	2	2	2484	1
</td>
</tr>
<tr>
<td>Save Processed Data</td>
<td>

```
[6]  data = pd.read_csv(r"/content/Train.csv")
     data.head()

[30] data_updated = data.drop(['ID','Gender','Discount_offered','Cost_of_the_Product','Prior_purchases','Customer_rating'],axis=1)

[37] x = data_updated.drop(['Reached.on.Time_Y.N'], axis=1)

[38] y = data_updated['Reached.on.Time_Y.N']

[39] x_res, y_res=us.fit_resample(x, y)
```
</td>
</tr>
</table>