<u>PROGRAMMING FOR BUSINESS ANALYTICS</u>

<u>SPRING 2022- BCIS 5110 SECTION 001</u>

<u>PROJECT REPORT ON</u>

<u>E COMMERCE PRODUCT SHIPPING DATAFILE USING MACHINE LEARNING ALGORITHMS</u>

Sainath Ravirala-11526529
Aasrita Sriparti -11544670
Rythwik Reddy Devireddy-11544548
Meher Jahnavi Viguturi-11544192

Table of Contents

E-commerce Product Shipping-

Prediction of on-time Delivery Products using Machine Learning Algorithm

Executive Summary

The main goal of the paper is to predict the E-commerce product shipping, that is to determine the on-time delivery of the product. Now a days ecommerce companies are facing a lot of critical time in delivering a product on time as this affects both the customers and suppliers. To solve this problem a model is developed using the full datafile (downloaded from Kaggle website), from the prediction result the companies can take steps to solve the problem and improve the satisfaction. In this project, prediction is developed using a machine learning classification algorithm: Logical Regression, Decision tree algorithm and Support Vector Machine algorithm, then the accuracy of the prediction is determined using these three algorithms. Finding which algorithm displays a better result and how that is useful in solving problems. The model is developed using a datafile with more than 10,000 delivery information's of the delivery company. The model shows that on-time delivery rates are not that much high because of mode of shipment, warehouse location and so on.

Project Motivation

In the fast-growing technology, online shopping has become a part of our life, as many customers are attracted towards online commerce because of trust and getting their needs from the single touch. Therefore, e-commerce companies also have their responsibilities in fulfilling all the customers needs, this can be possible only by delivering their product on-time and make the payment process hassle free. Generally, when a customer places an order, it goes into some process of work before reaching it to the customers. Due to this process, the orders are delayed, to find the reason of the delay and to reduce the problem, the project is proposed.

Key Questions

1. What is average number of customers who get satisfied with the delivery option?

2. Which mode of shipment is the better option in delivering on-time product?

3. What is the significant cause in customer satisfaction?

4. How the warehouse location affects the product delivery?

Above questions are answered in this report using data visualization techniques and machine learning Algorithms.

Data Source

The datafile is downloaded from the Kaggle website where the datafile contains more than 10,000 data observations with 12 different columns of an e-commerce company details.

Data Description

There are 12 different columns: ID (it shows how many transactions are in datafiles), Warehouse_block (it is divided into 5 blocks A, B, C, D, E), Mode_of_Shipment (Ship, Flight, Road), Customer_care_calls (1 to 5), Customer_ratings (1 to 5), Cost_of_the_Product (product cost), Prior_purchases (1 to 5), Product_importance (Low, Medium, High), Gender(Male, Female), Discount_offered (1 to 10), Weight_in_gms (weight of the products in grams), Reached on Time (Yes, No). Data types are integers and objects, Range Indexes: 0 to 1099 entries.

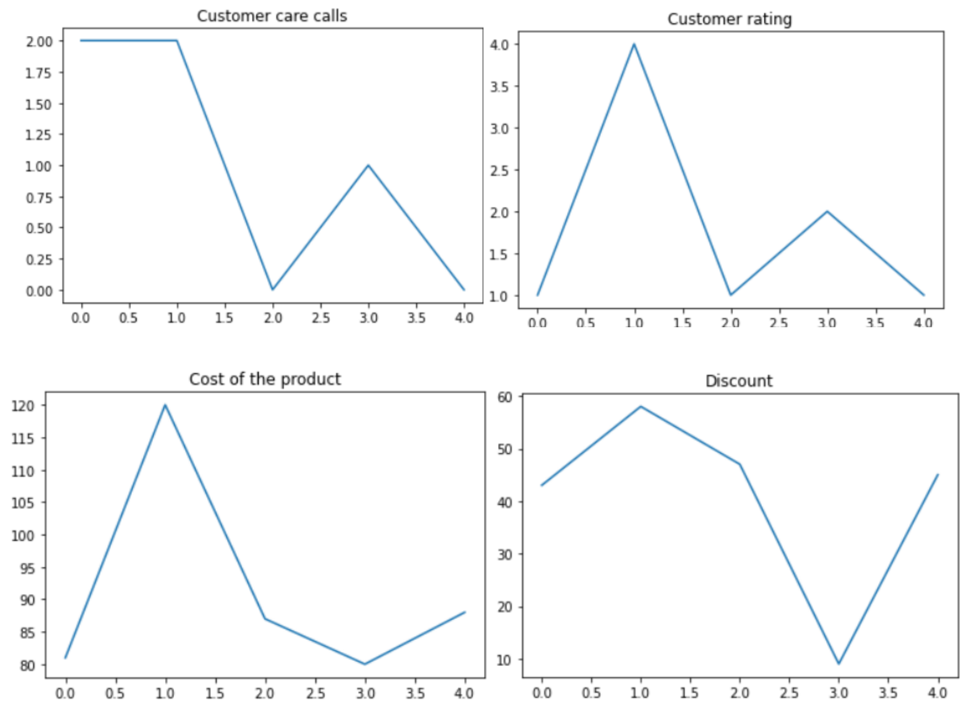
To work on the datafiles all the required python libraries are imported in the beginning, the datafile can be read using a read.csv command. First datafile is downloaded and stored in drive, after that it is added into the workspace by using the path. DataFile.head() command is used to display first five rows in the dataFile. Datafile.Shape command is used to display how many columns and entities are in datafile. Datafile.info() command is used to display all the information of the datafile columns. Then the null value is checked at this point to know about a missing value in datafile: datafile.isnull().sum() command is used, after entering the command, the null value will get displayed, in this datafile there is no missing values.

Exploratory Data Analysis

The main aim of the project is to find why the product is delayed, to find the solution first step is analysing the datafile. From the datafile it is found that the 40% of the product are not reached on-time, the reason behind is unknown. And also, it is well observed the delayed deliveries of product based on mode of shipment: around 15.9 % is on flight, around 16.34 % is on road, around 67.7 % is on ship. Therefore, the maximum delayed deliveries are happened when the ship is used as a mode of transport compared to the shipment through flight and road service.

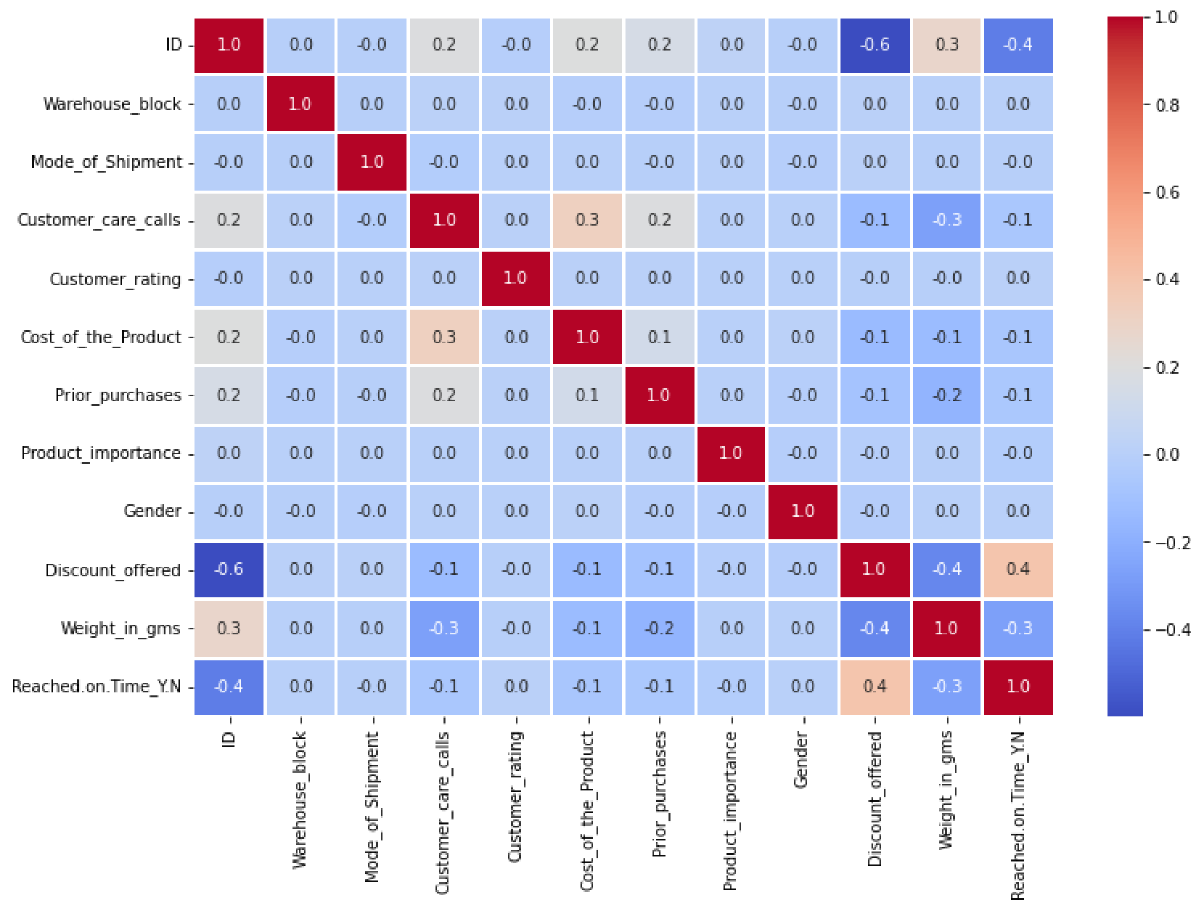
Delayed service in means of warehouse block is analysed to know the reason further, highest percentage of delayed deliveries are happened at the warehouse block F compared to other ware house blocks. Product importance deliveries is observed at this rate, where the low importance product falls under high number of deliveries compared to the highly important products, this shows that the low important products are ordered by high number of customers through the e-commerce companies.

Furtherly, analysis is done for the customer care calls, customer ratings, cost of the product and discounts to know better about the data as shown in the below screenshot.



Screenshot 1: Data Analysis

Heatmap: A heat map is a two-dimensional representation of information with the help of colours. Heat maps can help the user visualize simple or complex information. Heatmap is analysed to know about the correlation. A heatmap is a data visualization technique that shows magnitude of a phenomenon as colour in two dimensions. The variation in colour may be by hue or intensity, giving obvious visual clues to the reader about how the phenomenon is clustered or varies over space. The correlation varies from blue to red which relates directly the intensity of relation between two variables from low to high.



Screenshot 2: Heatmap

There is a slight positive correlation between the customer care calls as well as product cost as shown in the above screenshot 2.

Most common and popular machine learning algorithms:

- **Naive Bayes Classifier Algorithm (Supervised Learning - Classification)**

The Naive Bayes classifier is based on Bayes' theorem and classifies every value as independent of any other value. It allows us to predict a class/category, based on a given set of features, using probability.

- **K Means Clustering Algorithm (Unsupervised Learning - Clustering)**

The K Means Clustering algorithm is a type of unsupervised learning, which is used to categorise unlabelled data, i.e. data without defined categories or groups. The algorithm works by finding groups within the data, with the number of groups represented by the variable K. It then works iteratively to assign each data point to one of K groups based on the features provided.

- **Support Vector Machine Algorithm (Supervised Learning - Classification)**

Support Vector Machine algorithms are supervised learning models that analyse data used for classification and regression analysis. They essentially filter data into categories, which is achieved by providing a set of training examples, each set marked as belonging to one or the other of the two categories. The algorithm then works to build a model that assigns new values to one category or the other.

- **Linear Regression (Supervised Learning/Regression)**

Linear regression is the most basic type of regression. Simple linear regression allows us to understand the relationships between two continuous variables.

- **Logistic Regression (Supervised learning – Classification)**

Logistic regression focuses on estimating the probability of an event occurring based on the previous data provided. It is used to cover a binary dependent variable, that is where only two values, 0 and 1, represent outcomes.

- **Decision Trees (Supervised Learning – Classification/Regression)**

A decision tree is a flow-chart-like tree structure that uses a branching method to illustrate every possible outcome of a decision. Each node within the tree represents a test on a specific variable – and each branch is the outcome of that test.

- **Random Forests (Supervised Learning – Classification/Regression)**

Random forests or 'random decision forests' is an ensemble learning method, combining multiple algorithms to generate better results for classification, regression and other tasks. Each individual classifier is weak, but when combined with others, can produce excellent results. The algorithm starts with a 'decision tree' (a tree-like graph or model of decisions) and an input is entered at the top. It then travels down the tree, with data being segmented into smaller and smaller sets, based on specific variables.

- **Nearest Neighbours (Supervised Learning)**

The K-Nearest-Neighbour algorithm estimates how likely a data point is to be a member of one group or another. It essentially looks at the data points around a single data point to determine what group it is actually in. For example, if one point is on a grid and the algorithm is trying to determine what group that data point is in (Group A or Group B, for example) it would look at the data points near it to see what group the majority of the points are in.

Models and Analysis

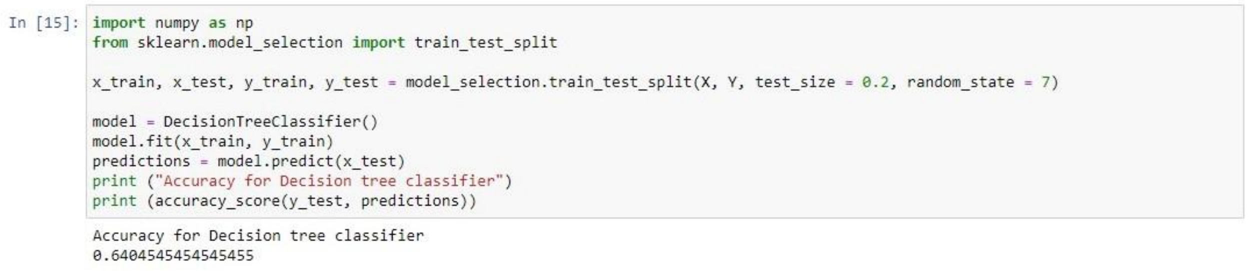
To build the model, the data need to be prepared well for application and research from the above analysis it is seen that the data is ready for next step. To make the project work well, datafile need to be divided into train and test data. Machine Learning Algorithm works well in two stages, usually datafile is split into 20% to 80% between testing and training. After dividing the datafile, 3 models are build using machine learning algorithms (Decision tree, Logistic Regression and Simple Vector Machine) and the comparison is made to find the best outcome.

Decision tree Algorithm

To make prediction, the algorithm starts from root node, then it compares the root node with the attributes, after comparison it will jump into next mode. In this project we implemented a decision tree using python. Following steps are considered for prediction: Data pre-processing, fitting a decision tree algorithm to the training set and predict the test result.

To fit the model to the training set, Decision tree classifier class is imported from sklearn. tree library, and then the below output is displayed after entering the code.

Got an accuracy rate of 64% (0.64045) for decision tree classification algorithm.



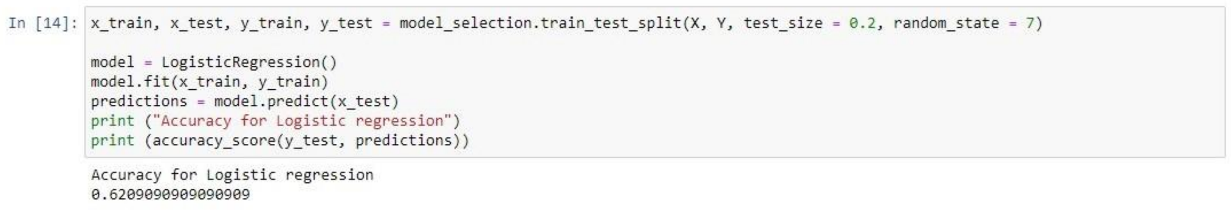
```
In [15]: import numpy as np
         from sklearn.model_selection import train_test_split

         x_train, x_test, y_train, y_test = model_selection.train_test_split(X, Y, test_size = 0.2, random_state = 7)

         model = DecisionTreeClassifier()
         model.fit(x_train, y_train)
         predictions = model.predict(x_test)
         print ("Accuracy for Decision tree classifier")
         print (accuracy_score(y_test, predictions))

         Accuracy for Decision tree classifier
         0.6404545454545455
```

Logistic Regression Classification

Logistic functions are used to model the binary dependent variable. It is generally a statistical model.

Implementing the logistic regression using python, got an accuracy rate of 62%. (0.62090)

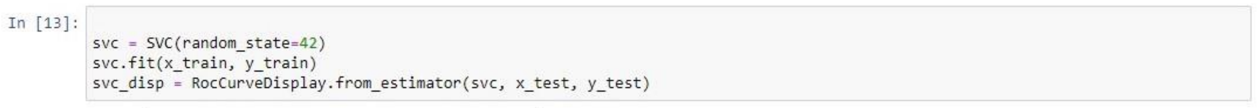
False Positive Rate (Positive label: 1)



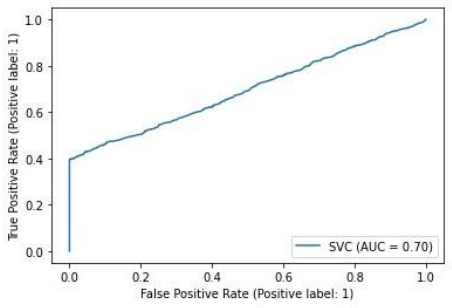
```
In [14]: x_train, x_test, y_train, y_test = model_selection.train_test_split(X, Y, test_size = 0.2, random_state = 7)

         model = LogisticRegression()
         model.fit(x_train, y_train)
         predictions = model.predict(x_test)
         print ("Accuracy for Logistic regression")
         print (accuracy_score(y_test, predictions))

         Accuracy for Logistic regression
         0.6209090909090909
```

Support Vector Machine

Support Vector Machine (SVM) algorithm is a supervised machine learning algorithm used for both classification and regression. Using this model we got accuracy rate of 70%.

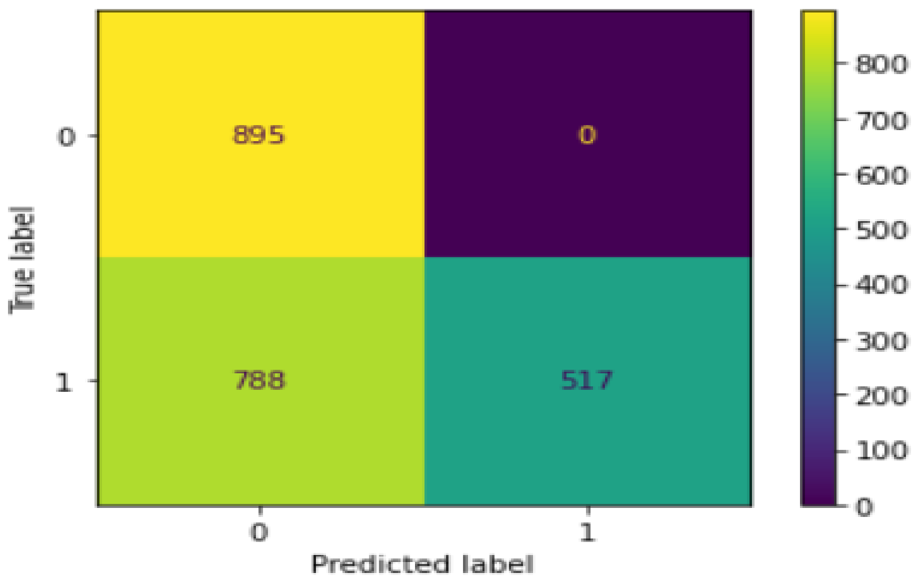


```
In [13]:
         svc = SVC(random_state=42)
         svc.fit(x_train, y_train)
         svc_disp = RocCurveDisplay.from_estimator(svc, x_test, y_test)
```



Findings and Results

Confusion matrix is built to know the performance of the classification algorithm. The below screenshot shows and summarize the performance of the confusion algorithm. The below result shows (screenshot 3) the list of expected values and the list of prediction from the machine learning model. Overall, it gives a positive result.



Screenshot 3: Confusion Matrix

It is seen that the Simple vector machine algorithm gives a better accurate result compare to the Logistic Regression classification algorithm and Decision tree classifier algorithm and the classification result is around 70%.

Conclusion

To find the cause of on-time delivery delay, the model is developed in this project work. the developed model gave a measure and different reason related to delivering the on- time product by analysing a large set of datafiles. By using this prediction, the e-commerce company can able to improve their result and rectify the errors in the future.

Appendix

```
import pandas as pd
from sklearn import model_selection
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt


from sklearn.metrics import RocCurveDisplay
from sklearn.svm import SVC

dataFile = pd.read_csv('C:\\Users\\LaptopCheckout\\Downloads\\
Train.csv')
dataFile.head()

   ID Warehouse_block Mode_of_Shipment  Customer_care_calls
Customer_rating  \
0   1               D           Flight                    4
2
1   2               F           Flight                    4
5
2   3               A           Flight                    2
2
3   4               B           Flight                    3
```

```
3
4   5               C           Flight                      2
2

   Cost_of_the_Product  Prior_purchases Product_importance Gender  \
0                  177                3                low      F
1                  216                2                low      M
2                  183                4                low      M
3                  176                4             medium      M
4                  184                3             medium      F

   Discount_offered  Weight_in_gms  Reached.on.Time_Y.N
0                44           1233                    1
1                59           3088                    1
2                48           3374                    1
3                10           1177                    1
4                46           2484                    1
```

```
dataFile.shape
```

```
(10999, 12)
```

```
dataFile.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  int32
 2   Mode_of_Shipment     10999 non-null  int32
 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  int32
 8   Gender               10999 non-null  int32
 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: int32(4), int64(8)
memory usage: 859.4 KB
```

```
dataFile.describe()
```

```
                ID  Warehouse_block  Mode_of_Shipment
Customer_care_calls  \
count  10999.00000     10999.000000      10999.000000
10999.000000
mean    5499.00000         2.333394          1.516865
2.054459
std     3175.28214         1.490726          0.756894
1.141490
min        0.00000         0.000000          0.000000
0.000000
25%     2749.50000         1.000000          1.000000
1.000000
50%     5499.00000         3.000000          2.000000
2.000000
75%     8248.50000         4.000000          2.000000
3.000000
max    10998.00000         4.000000          2.000000
5.000000

       Customer_rating  Cost_of_the_Product  Prior_purchases  \
count     10999.000000         10999.000000     10999.000000
mean          1.990545           114.196836         1.551414
std           1.413603            48.063272         1.458359
min           0.000000             0.000000         0.000000
25%           1.000000            73.000000         1.000000
50%           2.000000           118.000000         1.000000
75%           3.000000           155.000000         2.000000
max           4.000000           214.000000         7.000000

       Product_importance        Gender  Discount_offered
```

```
Weight_in_gms  \
count       10999.000000  10999.000000      10999.000000
10999.000000
mean            1.346031      0.495863         12.373216
2054.449586
std             0.631434      0.500006         16.205527
1239.074244
min             0.000000      0.000000          0.000000
0.000000
25%             1.000000      0.000000          3.000000
811.500000
50%             1.000000      0.000000          6.000000
2265.000000
75%             2.000000      1.000000          9.000000
3130.000000
max             2.000000      1.000000         64.000000
4033.000000

       Reached.on.Time_Y.N
count         10999.000000
mean              0.596691
std               0.490584
min               0.000000
25%               0.000000
50%               1.000000
75%               1.000000
max               1.000000
```

```
rows, columns = dataFile.shape
cell_count = rows * columns
number_of_nulls = dataFile.isnull().sum().sum()
percentage_of_missing = (number_of_nulls / cell_count) * 100
print(f'Percentage of missing values: {percentage_of_missing}%')
```

```
Percentage of missing values: 0.0%
```

```
pd.set_option("display.max_rows", None, "display.max_columns", None)
```

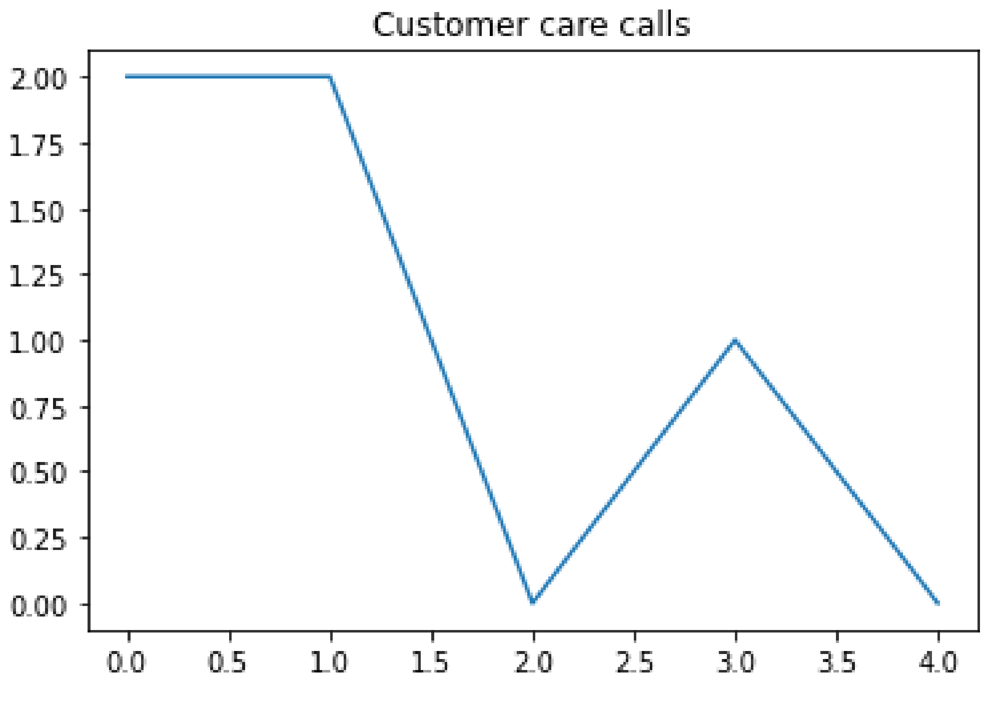
#dataFile = "Train.csv"

```
dataFile = pd.read_csv('C:\\Users\\LaptopCheckout\\Downloads\\
Train.csv')
```

columnNames = ['ID','Warehouse_block','Mode_of_Shipment','Customer_care_calls','Cust
omer_rating','Cost_of_the_Product','Prior_purchases','Product_importan
ce','Gender','Discount_offered','Weight_in_gms','Reachedon.Time_Y.N'\ *#dataset =
pd.read_csv(dataFile, names = columnNames, skiprows = 1)*

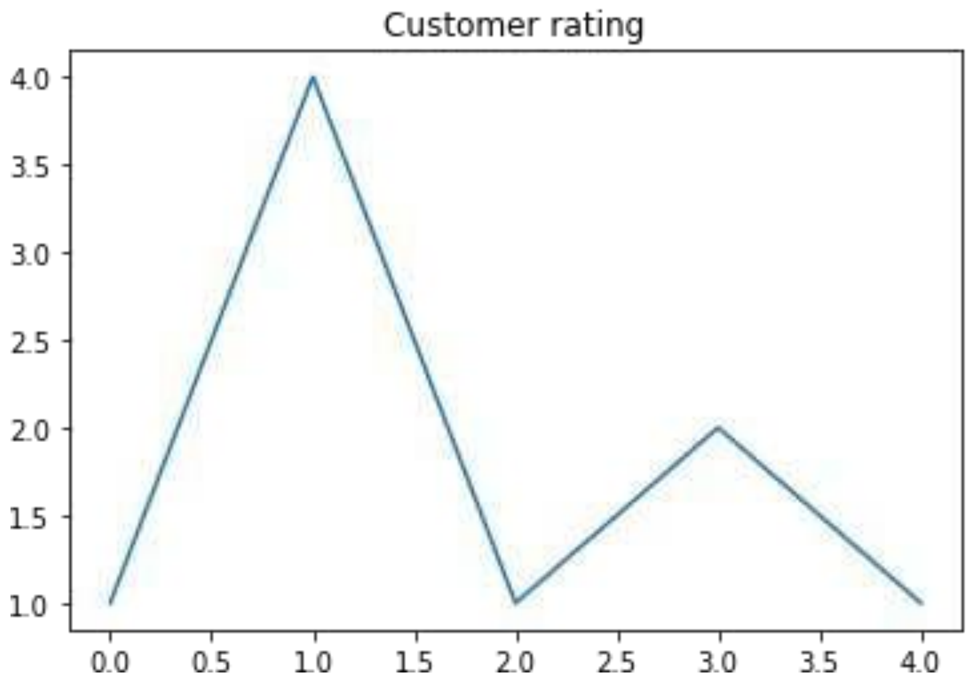
```
getColumn =
['ID','Warehouse_block','Mode_of_Shipment','Customer_care_calls','Cust
omer_rating','Cost_of_the_Product','Prior_purchases','Product_importan
```

```
ce','Gender','Discount_offered','Weight_in_gms'\

le = LabelEncoder()
for labelData in getColumn:

    dataFile[labelData\ = le.fit_transform(dataFile[labelData\)

array = dataFile.values
X = array[:,6:10\
Y = array[:,11\
Y = Y.astype('int')

print (array)

[[     0      3      0 ...     43   228      1\
 [     1      4      0 ...     58  1595      1\
 [     2      0      0 ...     47  1754      1\
 ...
 [10996      2      2 ...      3   151      0\
 [10997      4      2 ...      1   206      0\
 [10998      3      2 ...      5   619      0\\

id = dataFile['ID'\.head()
wareHouse = dataFile['Customer_care_calls'\.head()
plt.plot(id,wareHouse)
plt.title("Customer care calls")
plt.show()
```

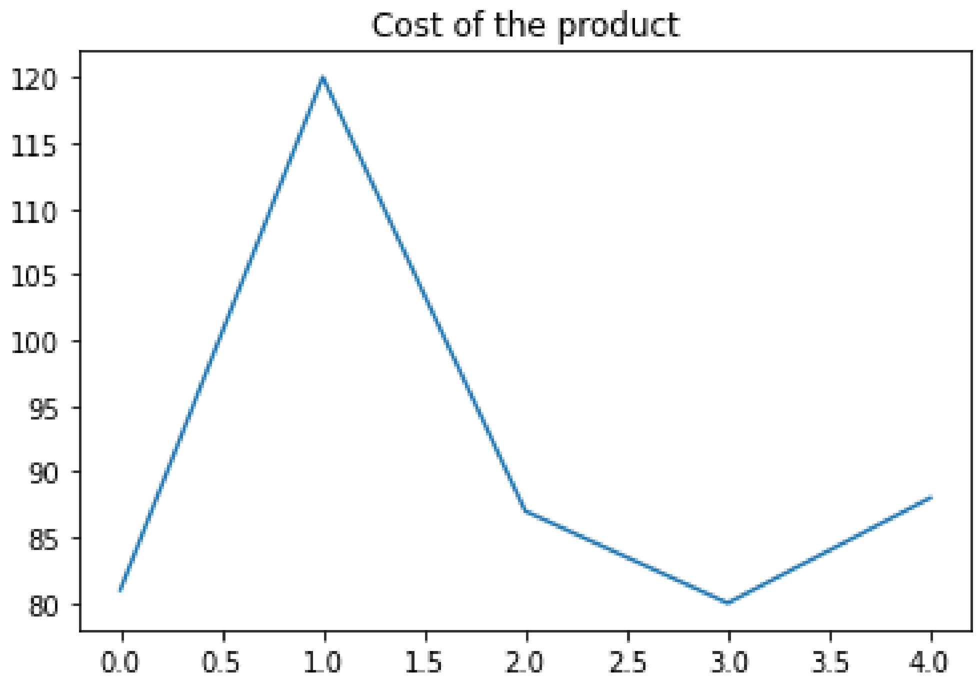


```
id = dataFile['ID'\.head()
customerRating = dataFile['Customer_rating'\.head()
```

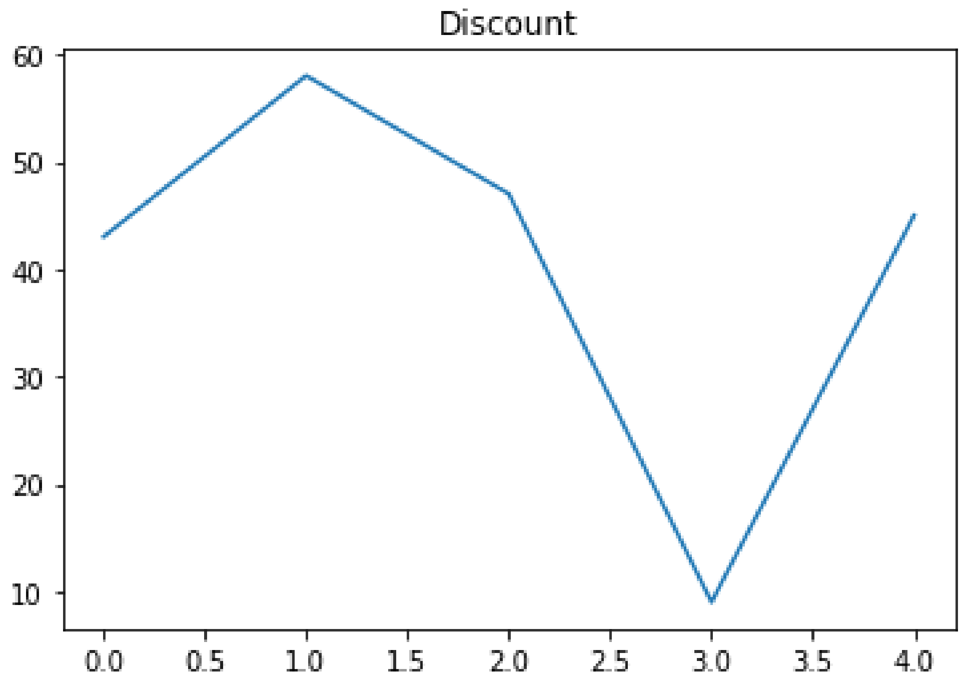
```
plt.plot(id,customerRating)
plt.title("Customer rating")
plt.show()
```



```
id = dataFile['ID'\.head()
productCost = dataFile['Cost_of_the_Product'\.head()
plt.plot(id,productCost)
plt.title("Cost of the product")
plt.show()
```

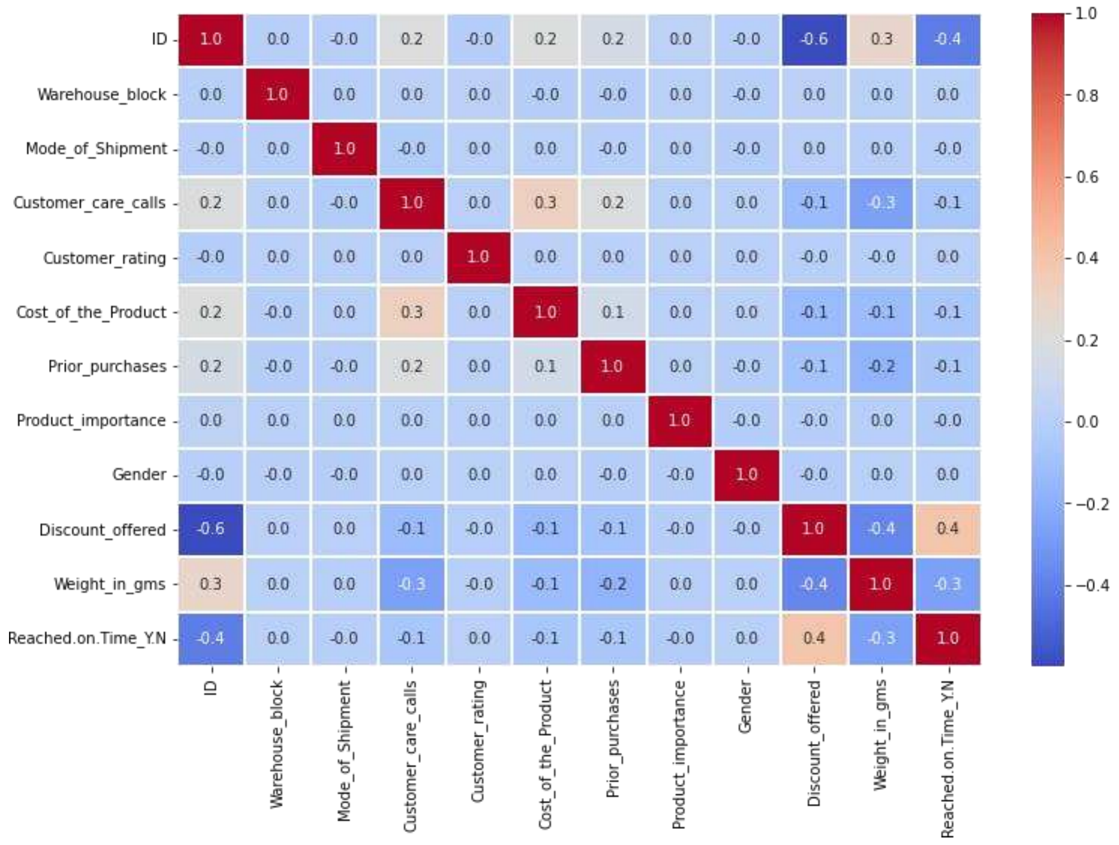


```
id = dataFile['ID'\.head()
discontOffered = dataFile['Discount_offered'\.head()
plt.plot(id,discontOffered)
plt.title("Discount")
plt.show()
```



```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
```

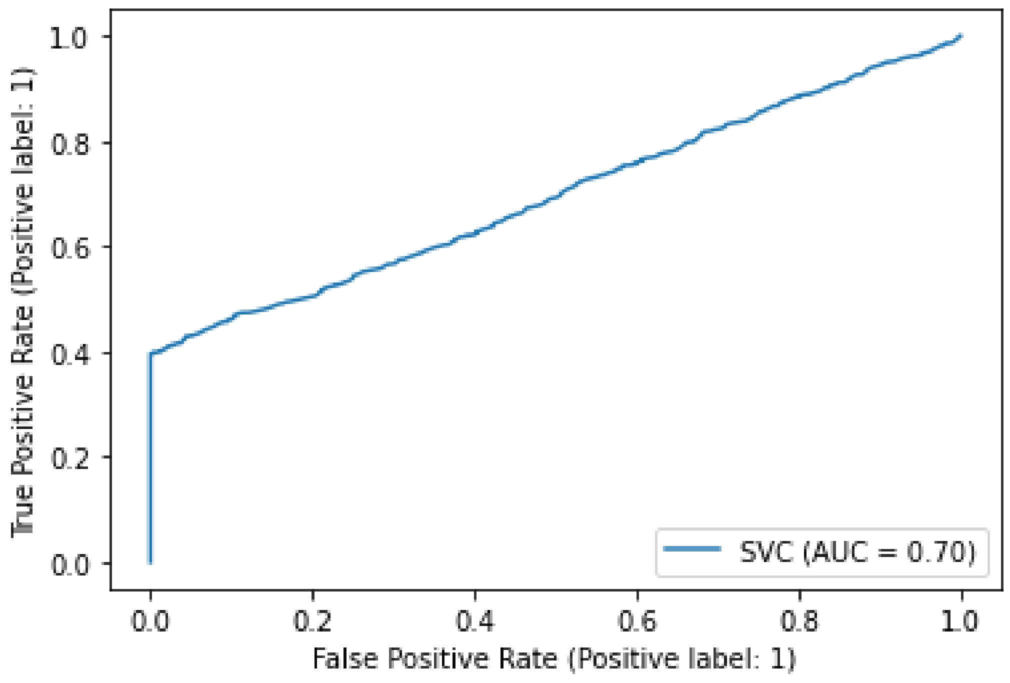
```
plt.figure(figsize = (12,8))
sns.heatmap(dataFile.corr(), cmap = 'coolwarm', annot = True, fmt =
'.1f', linewidth = .1)
plt.show()
```



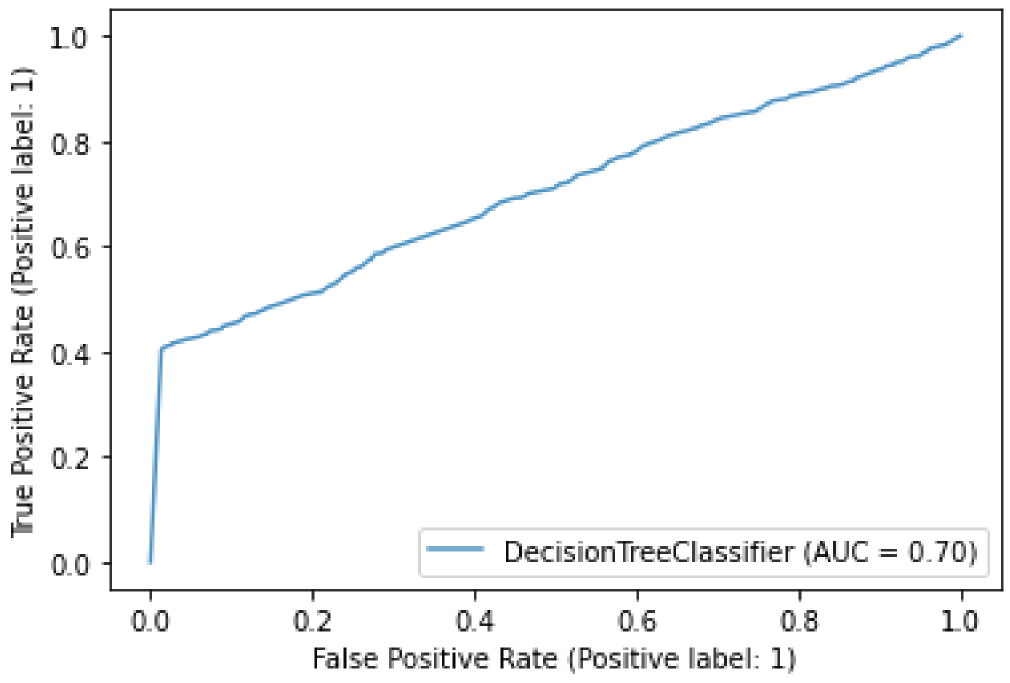
```
import numpy as np
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = model_selection.train_test_split(X,
Y, test_size = 0.2, random_state = 7)

model = DecisionTreeClassifier()
model.fit(x_train, y_train)
predictions = model.predict(x_test)
print ("Accuracy for Decision tree classifier")
print (accuracy_score(y_test, predictions))
```

```
Accuracy for Decision tree classifier
0.6404545454545455
```

```
svc = SVC(random_state=42)
svc.fit(x_train, y_train)
svc_disp = RocCurveDisplay.from_estimator(svc, x_test, y_test)
```



```
ax = plt.gca()
rfc_disp = RocCurveDisplay.from_estimator(model, x_test, y_test,
ax=ax, alpha=0.8)
```

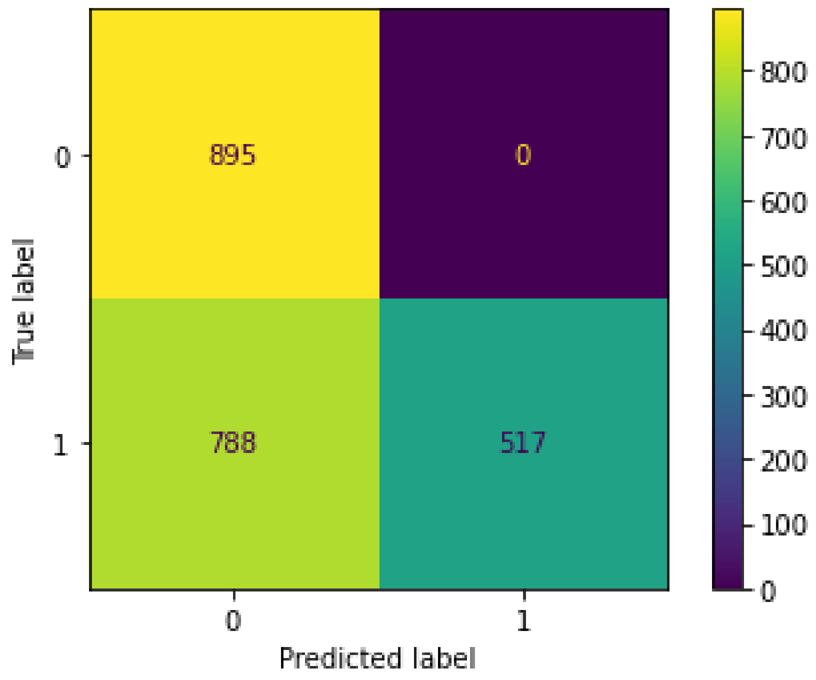


```
x_train, x_test, y_train, y_test =
model_selection.train_test_split(X,Y, test_size = 0.2, random_state
= 7)

model = LogisticRegression()
model.fit(x_train, y_train)
predictions =
model.predict(x_test)
print ("Accuracy for Logistic regression")
print (accuracy_score(y_test,
predictions))
```

```
Accuracy for Logistic
regression0.6209090909090909
```

```
from sklearn.metrics import plot_confusion_matrix

clf  =  SVC(random_state=0)
clf.fit(x_train, y_train)
SVC(random_state=0)
plot_confusion_matrix(clf, x_test,
y_test)plt.show()
```

```
C:\Users\LaptopCheckout\anaconda3\lib\site-packages\sklearn\utils\
deprecation.py:87: FutureWarning: Function plot_confusion_matrix is
deprecated; Function `plot_confusion_matrix` is deprecated in 1.0
andwill be removed in 1.2. Use one of the class methods:
ConfusionMatrixDisplay.from_predictions or
ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
```


True label
Predicted label