

**Assessment Report**

**on**

**“Product Return Prediction Analysis”**

**submitted as partial fulfillment for the award of**

**BACHELOR OF TECHNOLOGY**

**DEGREE**

**SESSION 2024-25**

**in**

**CSE(AI)**

**By**

**Name : Om prakash kannaujiya**

**Roll Number : 202401100300165**

**Section: C**

**Under the supervision of**

**“MAYANK SIR”**

KIET Group of Institutions, Ghaziabad

**INTRODUCTION:**

**The problem focuses on predicting whether a product will be returned or not using features like purchase amount, review score, and delivery time. Predicting returns helps improve logistics and customer satisfaction.**

**METHODOLOGY:**

**The dataset was preprocessed by renaming columns and converting the 'returned' column into binary format. We used Logistic Regression to classify whether a product would be returned based on features like purchase amount, review score, and days to delivery. The model was evaluated using accuracy, precision, recall, and a confusion matrix.**

**CODE:**

**import pandas as pd**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.linear\_model import LogisticRegression**

**from sklearn.metrics import confusion\_matrix, accuracy\_score, precision\_score, recall\_score, classification\_report**

**# Load dataset (adjusted to reflect your uploaded structure)**

**data = pd.read\_csv("/content/product\_return.csv")**

**# Rename columns if needed (e.g., if they contain typos or extra underscores)**

**data.columns = ["purchaseAmount", "reviewScore", "daysToDelivery", "returned"]**

**# Convert target to numeric**

**data["returned"] = data["returned"].map({"yes": 1, "no": 0})**

**# Features and target**

**features = data[["purchaseAmount", "reviewScore", "daysToDelivery"]]**

**target = data["returned"]**

**# Train-test split**

**trainX, testX, trainY, testY = train\_test\_split(features, target, test\_size=0.2, random\_state=42)**

**# Train model**

**model = LogisticRegression()**

**model.fit(trainX, trainY)**

**# Predictions**

**predictions = model.predict(testX)**

**# Metrics**

**accuracy = accuracy\_score(testY, predictions)**

**precision = precision\_score(testY, predictions)**

**recall = recall\_score(testY, predictions)**

**print(f"Accuracy: {accuracy:.2f}")**

**print(f"Precision: {precision:.2f}")**

**print(f"Recall: {recall:.2f}")**

**print("\nClassification Report:\n", classification\_report(testY, predictions))**

**# Confusion matrix**

**matrix = confusion\_matrix(testY, predictions)**

**sns.heatmap(matrix, annot=True, fmt="d", cmap="coolwarm", xticklabels=["No", "Yes"], yticklabels=["No", "Yes"])**

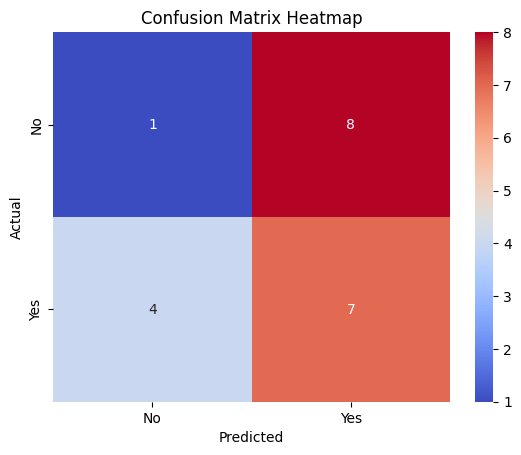
**plt.title("Confusion Matrix Heatmap")**

**plt.xlabel("Predicted")**

**plt.ylabel("Actual")**

**plt.show()**

**IMAGE:**



### **References / Credits:**

1. **Scikit-learn Documentation** https://scikit-learn.org/stable/user\_guide.html  
    (For Logistic Regression, model evaluation metrics, and data preprocessing techniques.)
2. **Pandas Documentation** https://pandas.pydata.org/docs/  
    (For data loading, cleaning, and transformation.)
3. **Matplotlib and Seaborn**
   * <https://matplotlib.org/>
   * https://seaborn.pydata.org/  
      (For data visualization and plotting the confusion matrix.)
4. **Dataset Source** Internal dataset: product\_return.csv uploaded and used during the project.
5. **Google Colab** https://colab.research.google.com/  
    (Used as the development environment for running and testing Python code.)
6. **Python Programming Language**<https://www.python.org/> (Main language used for implementing machine learning models.)