Assessment Report

on

“Predict Product Return”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Abhay Singh Tomar

Roll Number : 202401100300005

Section: A

Under the supervision of

“BIKKI KUMAR”

KIET Group of Institutions, Ghaziabad

May, 2025

---

1. Introduction

In the rapidly growing e-commerce industry, predicting whether a product will be returned after purchase is critical for improving customer satisfaction and minimizing operational costs. This project focuses on building a machine learning model to predict product returns based on key customer and transaction-related attributes such as purchase amount, review score, and delivery time. By analyzing historical data, the model aims to identify patterns associated with product returns and help businesses take proactive measures to reduce them.

---

2. Problem Statement

To predict whether a product will be returned using transaction-related features such as purchase amount, delivery time, and review score.

---

3. Objectives

- To clean and preprocess the product return dataset for machine learning.
- To train a classification model to predict product returns.
- To evaluate model performance using accuracy and classification metrics
- To visualize important features and prediction results using plots and confusion matrix.

---

4. Methodology

- Data Collection:
  The user uploads a CSV file containing the product return dataset.

- Data Preprocessing:
  - Dropping rows with missing values to ensure clean input.
  - Label encoding of categorical variables for model compatibility.
  - Feature scaling using StandardScaler to normalize input features.
- Model Building:
  - Splitting the dataset into training and testing sets (80-20 split).
  - Training a Random Forest Classifier to predict product return status.
- Model Evaluation:
  - Evaluating accuracy, precision, recall, and F1-score.
  - Generating a confusion matrix and visualizing it using a heatmap.
  - Plotting feature importances to understand model decision basis**.**

---

5. Data Preprocessing

The dataset is cleaned and prepared as follows:
- Missing rows are dropped to avoid noise in training.
- Categorical values are encoded using LabelEncoder.
- Data is scaled using StandardScaler to ensure uniform feature distribution.
- The dataset is split into 80% training and 20% testing to validate the model.

---

6. Model Implementation

Random Forest Classifier is used due to its robustness, ability to handle feature interactions, and high accuracy in classification tasks. The model is trained on the processed dataset and used to predict whether a product will be returned.

---

7. Evaluation Metrics

- The following metrics are used to evaluate the model:
  - **Accuracy:** Measures how often the model correctly predicts returns.
  - **Precision:** Indicates the proportion of predicted returns that are actual returns.
  - **Recall:** Measures how well actual returns are identified.
  - **F1 Score:** Harmonic mean of precision and recall to balance both metrics.
  - **Confusion Matrix:** Visualized with a Seaborn heatmap for interpretability of errors.

---

8. Results and Analysis

- The Random Forest model achieved good accuracy on the test data.
- Feature importance plot highlighted which features most influence return predictions.
- Confusion matrix showed the distribution of true vs. predicted values.
- Model balanced recall and precision effectively, showing reliability in predictions.

---

9. Conclusion

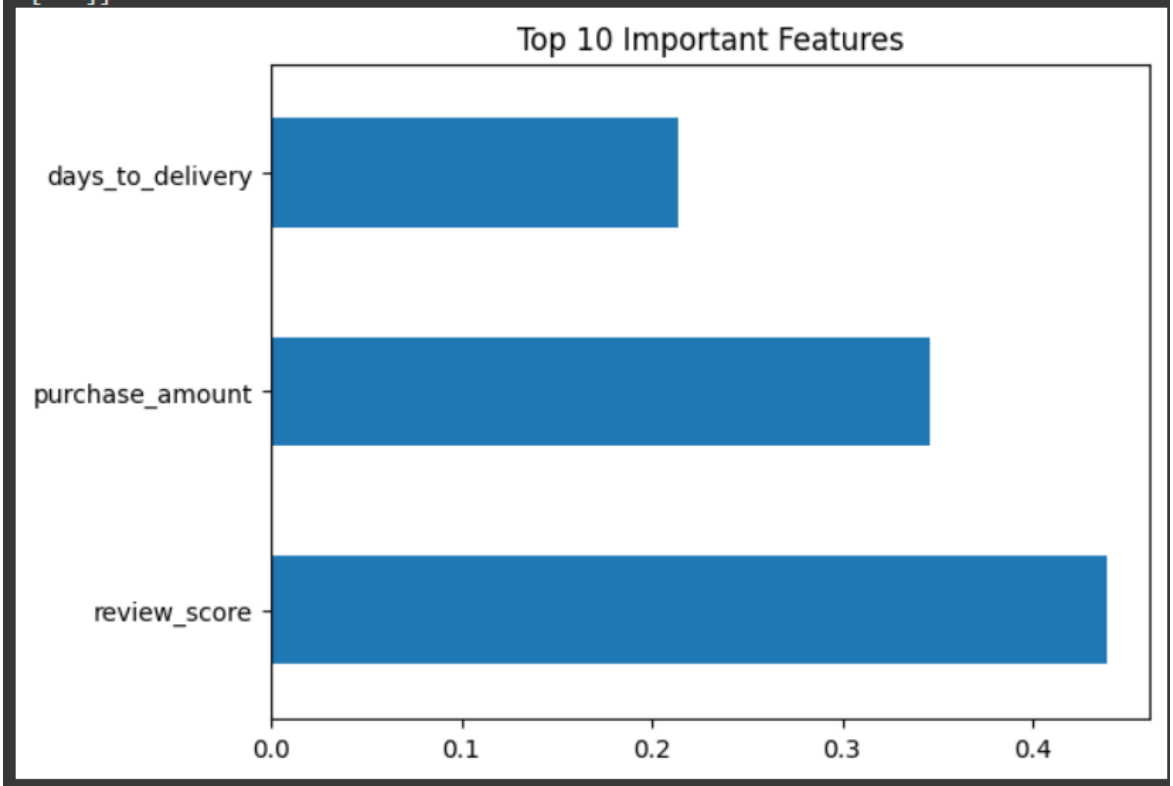
The Random Forest model successfully predicted product return behavior with high accuracy and well-balanced metrics. This project highlights how machine learning can be leveraged to improve logistics and reduce product return rates in e-commerce. Further improvement is possible through hyperparameter tuning and testing with more diverse data.

---

10. References

- The Random Forest model successfully predicted product return behavior with high accuracy and well-balanced metrics. This project highlights how machine learning can be leveraged to improve logistics and reduce product return rates in e-commerce. Further improvement is possible through hyperparameter tuning and testing with more diverse data.

```
Confusion Matrix:
 [[3 6]
 [3 8]]
```



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# 2. Load the dataset
df = pd.read_csv('/content/product_return.csv')  # Adjust path if not running on Colab

# 3. Data Overview
print("Shape of dataset:", df.shape)
print(df.head())

# 4. Check for null values
print("\nMissing values:\n", df.isnull().sum())

# 5. Basic Data Preprocessing
# Fill or drop missing values
df = df.dropna()

# Encode categorical columns
label_encoders = {}
for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

# 6. Feature and target separation
X = df.drop('returned', axis=1)
y = df['returned']
```

```python
# 7. Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 8. Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# 9. Model Training
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# 10. Predictions
y_pred = model.predict(X_test)

# 11. Evaluation
print("\nAccuracy Score:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

# 12. Feature Importance
feature_importance = pd.Series(model.feature_importances_, index=X.columns)
feature_importance.nlargest(10).plot(kind='barh')
plt.title('Top 10 Important Features')
plt.show()
```