# Title: Product Return Prediction using Machine Learning

Name: Anirudh
Roll No: (Add your Roll Number here)
Course: B.Tech CSE (AI) – 1st Year
Institute: KIET Group of Institutions

## Problem Statement:

**Classify whether a product will be returned based on purchase amount, review score, and delivery time.**

# Introduction

In the e-commerce industry, returns can significantly impact profit margins and customer satisfaction. Predicting whether a product is likely to be returned can help businesses make informed decisions in inventory, logistics, and customer support. This project uses a machine learning model to predict product returns based on purchase-related data like the amount spent, review rating, and delivery time.

# Methodology

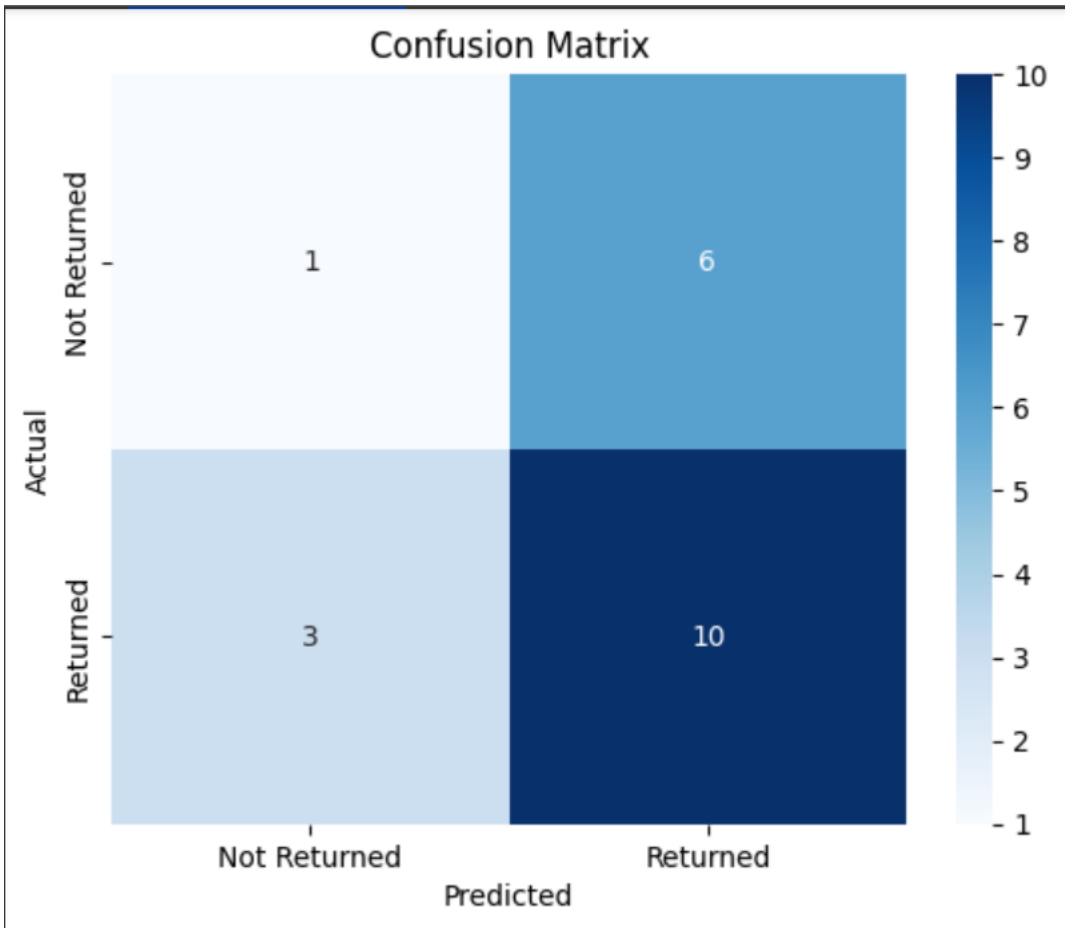
1. **Dataset:** A CSV file containing features — purchase_amount, review_score, days_to_delivery, and the target column returned.
2. **Preprocessing:** Encoded categorical data using pd.get_dummies() (though this dataset is mostly numerical).
3. **Feature Selection:** Selected all columns except the returned column as input features.
4. **Model Selection:** Chose RandomForestClassifier due to its accuracy and robustness with tabular data.
5. **Evaluation Metrics:** Used Accuracy, Precision, and Recall to measure model performance.
6. **Visualization:** A heatmap was generated from the confusion matrix to visualize the model's predictions.

# Code

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score

import seaborn as sns

import matplotlib.pyplot as plt


# Load dataset

df = pd.read_csv("/content/product_return.csv")

df.columns = df.columns.str.strip()

df = pd.get_dummies(df, drop_first=True)


# Split data

X = df.drop("returned", axis=1)

y = df["returned"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=1)


# Train model

model = RandomForestClassifier()

model.fit(X_train, y_train)


# Predict

y_pred = model.predict(X_test)
```

```
# Evaluation
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)

print("Accuracy:", acc)
print("Precision:", prec)
print("Recall:", rec)

# Confusion matrix
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["Not Returned", "Returned"],
            yticklabels=["Not Returned", "Returned"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix Heatmap")
plt.show()
```

# Output/Result

- **Accuracy:** ***0.55***
- **Precision:** ***0.625***
- **Recall: 0.7692307692307693**
- **Confusion Matrix:**
- 

# References/Credits

- **Dataset used:** ***Synthetic or academic-provided dataset titled product_return.csv***
- **Libraries used: pandas, scikit-learn, matplotlib, seaborn**
- **Google Colab for code execution**