Title:
Predicting Product Returns Using Machine Learning

Submitted By:
Name: Samarth Shukla
Roll No.: 202401100300212
Course: Intro to AI
Instructor: Shivansh Sir
Date: 13/05/2025

Problem Statement:
The goal of this project is to predict whether a product will be returned based on purchase-related features such as amount spent, review score, and delivery time using machine learning.

In e-commerce, product returns lead to financial losses and logistical challenges. Being able to **predict the likelihood of a return** helps companies:

- Reduce unnecessary shipment costs.
- Improve inventory planning.
- Identify potentially unsatisfied customers early.

This project uses a dataset containing features like:

- Purchase amount
- Customer review score
- Days taken for delivery
- Return status (Yes/No)

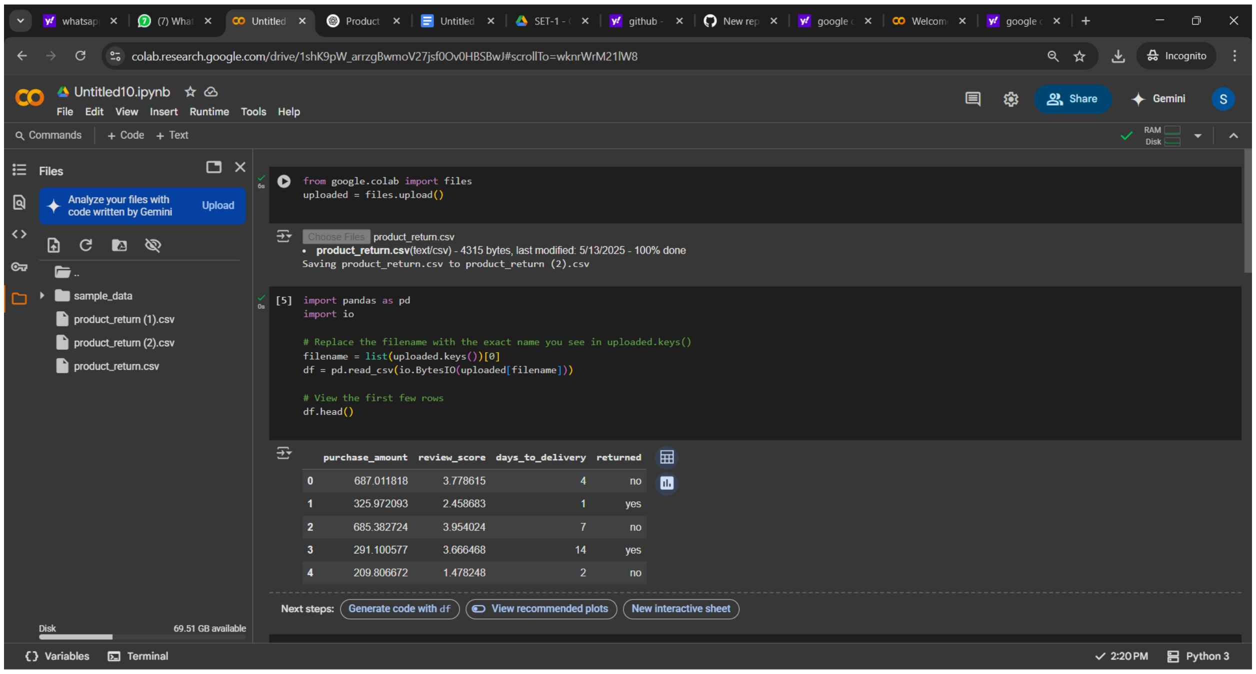
The objective is to train a machine learning classifier that can predict the return status using these numeric inputs.

---

c. Methodology

1. **Data Upload & Exploration:**

   - The dataset (`product_return.csv`) was uploaded in Google Colab.
   - It was found to have no missing values.
   - All features were numerical except the target column (`returned`), which was categorical.
2. **Preprocessing:**
   - The target variable was label-encoded (Yes = 1, No = 0).
   - No additional text or categorical features were present.
3. **Model Selection:**
   - A **Random Forest Classifier** was chosen for its high accuracy and ability to handle numeric data well.
4. **Training and Evaluation:**
   - Data was split into 80% training and 20% testing.
   - Evaluation metrics included **confusion matrix** and **classification report**.
5. **Prediction:**
   - A custom function was implemented to predict returns based on new input data.


colab.research.google.com/drive/1shK9pW_arrzgBwmoV27jsf0Ov0HBSBwJ#scrollTo=wknrWrM21lW8
Untitled10.ipynb
File Edit View Insert Runtime Tools Help
Files
sample_data
product_return (1).csv
product_return (2).csv
product_return.csv
Disk 69.51 GB available

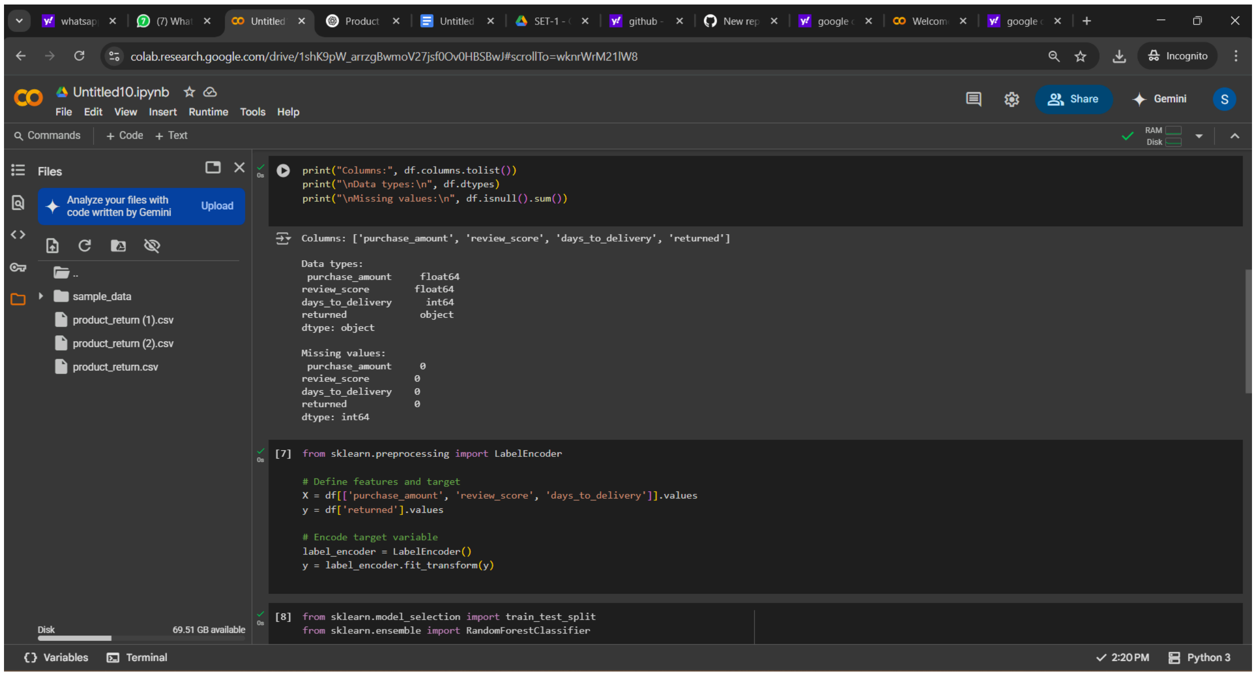
```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files product_return.csv
• product_return.csv(text/csv) - 4315 bytes, last modified: 5/13/2025 - 100% done
Saving product_return.csv to product_return (2).csv
```

```
[5] import pandas as pd
    import io

    # Replace the filename with the exact name you see in uploaded.keys()
    filename = list(uploaded.keys())[0]
    df = pd.read_csv(io.BytesIO(uploaded[filename]))

    # View the first few rows
    df.head()
```

	purchase_amount	review_score	days_to_delivery	returned
0	687.011818	3.778615	4	no
1	325.972093	2.458683	1	yes
2	685.382724	3.954024	7	no
3	291.100577	3.666468	14	yes
4	209.806672	1.478248	2	no

Next steps: Generate code with df
View recommended plots
New interactive sheet
Variables Terminal 2:20 PM Python 3

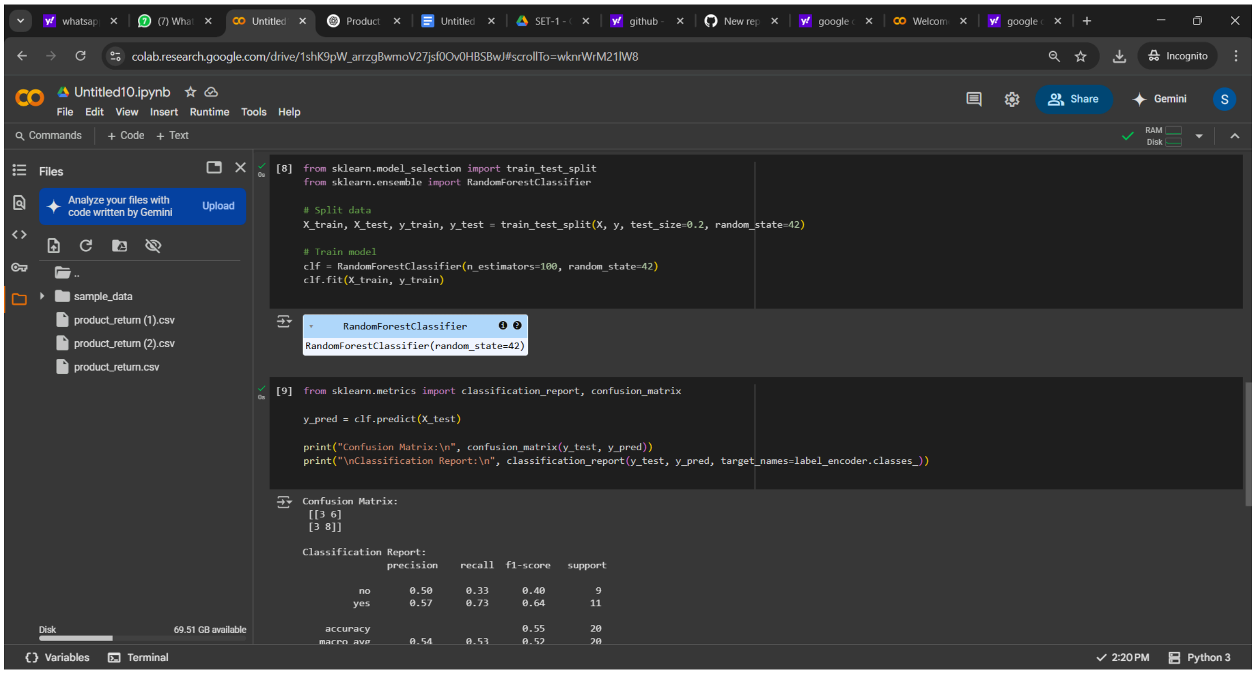

colab.research.google.com/drive/1shK9pW_arrzgBwmoV27jsf0Ov0HBSBwJ#scrollTo=wknrWrM21lW8
Untitled10.ipynb
File Edit View Insert Runtime Tools Help
Files
sample_data
product_return (1).csv
product_return (2).csv
product_return.csv
Disk 69.51 GB available

```
print("Columns:", df.columns.tolist())
print("\nData types:\n", df.dtypes)
print("\nMissing values:\n", df.isnull().sum())
```

```
Columns: ['purchase_amount', 'review_score', 'days_to_delivery', 'returned']

Data types:
 purchase_amount     float64
review_score        float64
days_to_delivery      int64
returned             object
dtype: object

Missing values:
 purchase_amount     0
review_score        0
days_to_delivery    0
returned            0
dtype: int64
```

```
[7] from sklearn.preprocessing import LabelEncoder

    # Define features and target
    X = df[['purchase_amount', 'review_score', 'days_to_delivery']].values
    y = df['returned'].values

    # Encode target variable
    label_encoder = LabelEncoder()
    y = label_encoder.fit_transform(y)
```

```
[8] from sklearn.model_selection import train_test_split
    from sklearn.ensemble import RandomForestClassifier
```

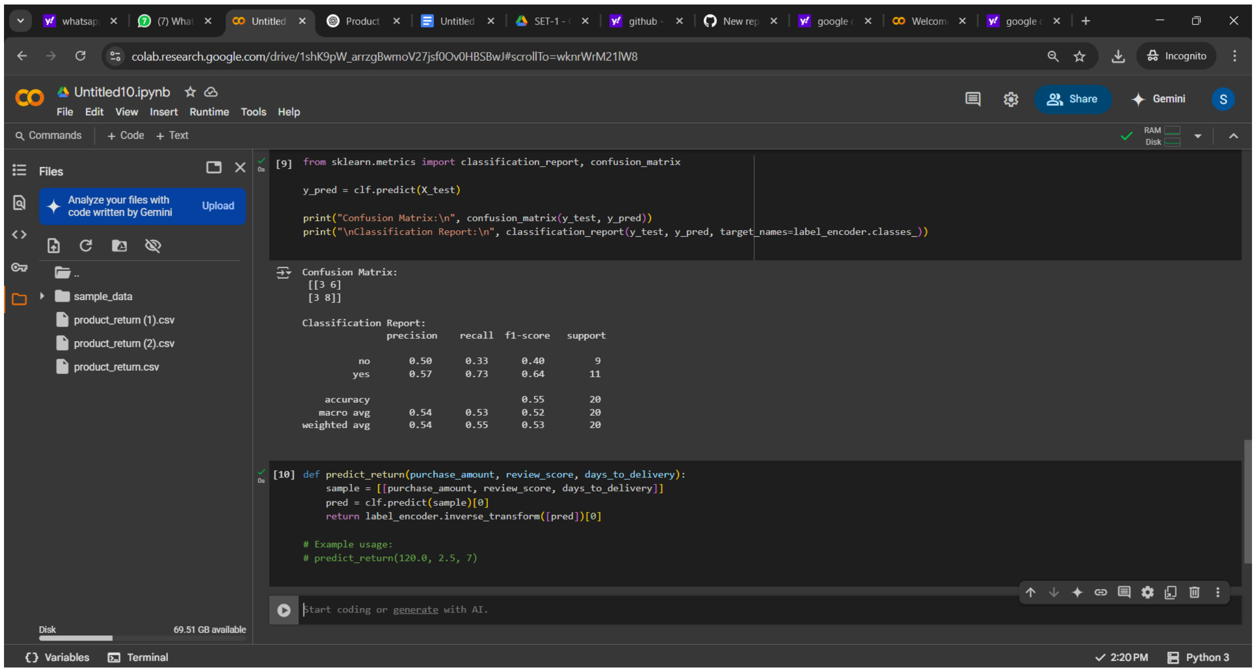


```
[8] from sklearn.model_selection import train_test_split
    from sklearn.ensemble import RandomForestClassifier

    # Split data
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Train model
    clf = RandomForestClassifier(n_estimators=100, random_state=42)
    clf.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
[9] from sklearn.metrics import classification_report, confusion_matrix

    y_pred = clf.predict(X_test)

    print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
    print("\nClassification Report:\n", classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

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

Classification Report:
               precision    recall  f1-score   support

          no       0.50      0.33      0.40         9
         yes       0.57      0.73      0.64        11

    accuracy                           0.55        20
   macro avg       0.54      0.53      0.52        20
```



```
[9] from sklearn.metrics import classification_report, confusion_matrix

    y_pred = clf.predict(X_test)

    print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
    print("\nClassification Report:\n", classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

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

Classification Report:
               precision    recall  f1-score   support

          no       0.50      0.33      0.40         9
         yes       0.57      0.73      0.64        11

    accuracy                           0.55        20
   macro avg       0.54      0.53      0.52        20
weighted avg       0.54      0.55      0.53        20
```

```
[10] def predict_return(purchase_amount, review_score, days_to_delivery):
         sample = [[purchase_amount, review_score, days_to_delivery]]
         pred = clf.predict(sample)[0]
         return label_encoder.inverse_transform([pred])[0]

     # Example usage:
     # predict_return(120.0, 2.5, 7)
```