PYTHON

CODE:

```
# return_prediction.py
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score

# Loading the data
df = pd.read_excel("superstore_return.xlsx", sheet_name="Sheet1")

# Clean and preprocess the data
df['ORDER_DATE'] = pd.to_datetime(df['ORDER_DATE'], errors='coerce',
dayfirst=True)
df['SHIP_DATE'] = pd.to_datetime(df['SHIP_DATE'], errors='coerce',
dayfirst=True)
df.dropna(subset=['RETURN_STATUS', 'PRODUCT_ID', 'CATEGORY',
'SUB_CATEGORY', 'SALES'], inplace=True)

# One-hot encode categorical variables
cat_cols = ['SHIP_MODE', 'SEGMENT', 'REGION', 'CATEGORY', 'SUB_CATEGORY']
df_encoded = pd.get_dummies(df, columns=cat_cols, drop_first=True)
df_encoded['RETURN_STATUS'] = df_encoded['RETURN_STATUS'].astype(int)

# Model training
X = df_encoded.drop(columns=['RETURN_STATUS', 'ROW_ID', 'ORDER_ID',
'ORDER_DATE', 'SHIP_DATE',
                             'CUSTOMER_ID', 'CUSTOMER_NAME', 'COUNTRY',
'CITY', 'STATE',
                             'POSTAL_CODE', 'PRODUCT_ID', 'PRODUCT_NAME'])

y = df_encoded['RETURN_STATUS']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Prediction and Evaluation
y_prob = model.predict_proba(X_test)[:, 1]
y_pred = (y_prob >= 0.5).astype(int)
```

```
print("Classification Report:\n", classification_report(y_test, y_pred))
print("ROC AUC Score:", roc_auc_score(y_test, y_prob))

# Predict on full dataset
df_encoded['High_Risk_Probability'] = model.predict_proba(X)[:, 1]

# Identify High-Risk Products
# Use top 5% as high-risk
threshold = df_encoded['High_Risk_Probability'].quantile(0.95)
df_encoded['High_Risk_Flag'] = df_encoded['High_Risk_Probability'] >=
threshold

# Visualize distribution
plt.hist(df_encoded['High_Risk_Probability'], bins=20, edgecolor='black')
plt.title('Distribution of Predicted Return Probabilities')
plt.xlabel('Probability')
plt.ylabel('Count')
plt.grid(True)
plt.show()

# Export high-risk products
high_risk = df_encoded[df_encoded['High_Risk_Flag']]
high_risk[['PRODUCT_ID', 'PRODUCT_NAME',
'High_Risk_Probability']].to_csv("high_risk_products.csv", index=False)

print("Exported high-risk products to 'high_risk_products.csv'")
```

OUTPUT:

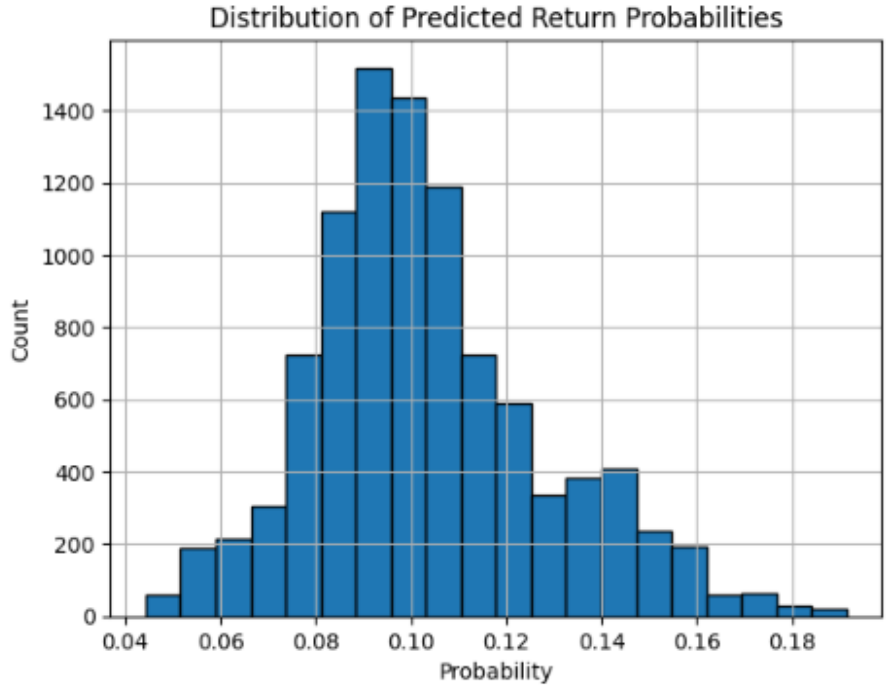
```
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with n
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with n
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with n
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
Classification Report:
               precision    recall  f1-score   support

           0       0.90      1.00      0.95      1763
           1       0.00      0.00      0.00       197

    accuracy                           0.90      1960
   macro avg       0.45      0.50      0.47      1960
weighted avg       0.81      0.90      0.85      1960

ROC AUC Score: 0.5459098617665435
```

```
ROC AUC Score: 0.5459098617665435
```




```
Exported high-risk products to 'high_risk_products.csv'
```

OUTPUT FILES:

high_risk_products.csv

superstore_return.xlsx