import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv('consumer\_complaints.csv')

# Check structure

print(df.head())

print(df['Product'].value\_counts())

# Filter for selected categories

categories = [

'Credit reporting, credit repair services, or other personal consumer reports',

'Debt collection',

'Consumer Loan',

'Mortgage'

]

df = df[df['Product'].isin(categories)]

# Map to target labels

label\_map = {

'Credit reporting, credit repair services, or other personal consumer reports': 0,

'Debt collection': 1,

'Consumer Loan': 2,

'Mortgage': 3

}

df['label'] = df['Product'].map(label\_map)

# Plot distribution

sns.countplot(x='label', data=df)

plt.title('Target Distribution')

plt.show()

import nltk

from nltk.corpus import stopwords

from nltk.stem import WordNetLemmatizer

import re

nltk.download('stopwords')

nltk.download('wordnet')

stop\_words = set(stopwords.words('english'))

lemmatizer = WordNetLemmatizer()

def clean\_text(text):

text = str(text).lower()

text = re.sub(r'[^a-zA-Z\s]', '', text)

tokens = text.split()

tokens = [lemmatizer.lemmatize(word) for word in tokens if word not in stop\_words]

return ' '.join(tokens)

df['clean\_complaint'] = df['Consumer complaint narrative'].apply(clean\_text)

df = df.dropna(subset=['clean\_complaint'])

from sklearn.feature\_extraction.text import TfidfVectorizer

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

tfidf = TfidfVectorizer(max\_features=5000)

X = tfidf.fit\_transform(df['clean\_complaint']).toarray()

y = df['label']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

from sklearn.linear\_model import LogisticRegression

from sklearn.naive\_bayes import MultinomialNB

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

models = {

'Logistic Regression': LogisticRegression(max\_iter=1000),

'Naive Bayes': MultinomialNB(),

'Random Forest': RandomForestClassifier(),

'Support Vector Machine': SVC()

}

for name, model in models.items():

model.fit(X\_train, y\_train)

accuracy = model.score(X\_test, y\_test)

print(f'{name} Accuracy: {accuracy:.4f}')

from sklearn.metrics import classification\_report, confusion\_matrix

best\_model = LogisticRegression(max\_iter=1000)

best\_model.fit(X\_train, y\_train)

y\_pred = best\_model.predict(X\_test)

print("Classification Report:\n", classification\_report(y\_test, y\_pred))

sns.heatmap(confusion\_matrix(y\_test, y\_pred), annot=True, fmt='d')

plt.title('Confusion Matrix')

plt.show()

sample = ["I have issues with my mortgage payment"]

sample\_clean = [clean\_text(sample[0])]

sample\_vector = tfidf.transform(sample\_clean)

prediction = best\_model.predict(sample\_vector)

print(f'Predicted Category: {prediction[0]}')