# Import necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import OneHotEncoder

from sklearn.metrics import accuracy\_score

# Load your dataset

data = pd.read\_csv('Data\_Gov\_Tamil\_Nadu.csv', encoding='ISO-8859-1')

# Handle missing values, duplicates, and outliers

data.dropna(inplace=True)

data.drop\_duplicates(inplace=True)

from scipy.stats import zscore

data = data[(np.abs(zscore(data.select\_dtypes(include=[np.number]))) < 3).all(axis=1)]

# Define the dependent and independent variables

X = data.drop('COMPANY\_STATUS', axis=1)

y = data['COMPANY\_STATUS']

# List of categorical columns for one-hot encoding

categorical\_columns = [

    'CORPORATE\_IDENTIFICATION\_NUMBER', 'COMPANY\_NAME', 'COMPANY\_CLASS',

    'COMPANY\_CATEGORY', 'COMPANY\_SUB\_CATEGORY', 'DATE\_OF\_REGISTRATION',

    'INDUSTRIAL\_CLASS', 'PRINCIPAL\_BUSINESS\_ACTIVITY\_AS\_PER\_CIN',

    'REGISTERED\_OFFICE\_ADDRESS', 'REGISTRAR\_OF\_COMPANIES', 'EMAIL\_ADDR',

    'LATEST\_YEAR\_ANNUAL\_RETURN', 'LATEST\_YEAR\_FINANCIAL\_STATEMENT'

]

# Create a one-hot encoder

encoder = OneHotEncoder(sparse=False, handle\_unknown='ignore')

# Split the data into training and testing sets

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

# Fit and transform the encoder on the categorical columns for both training and testing sets

X\_train[categorical\_columns] = X\_train[categorical\_columns].astype(str)

X\_test[categorical\_columns] = X\_test[categorical\_columns].astype(str)

# Fit and transform the encoder on the categorical columns for both training and testing sets

X\_train\_encoded = encoder.fit\_transform(X\_train[categorical\_columns])

X\_test\_encoded = encoder.transform(X\_test[categorical\_columns])

# Get the feature names after one-hot encoding

encoded\_feature\_names = encoder.get\_feature\_names\_out(input\_features=categorical\_columns)

# Create DataFrames for the encoded data

X\_train\_encoded\_df = pd.DataFrame(X\_train\_encoded, columns=encoded\_feature\_names)

X\_test\_encoded\_df = pd.DataFrame(X\_test\_encoded, columns=encoded\_feature\_names)

# Concatenate the encoded data with the numeric data

X\_train\_final = pd.concat([X\_train\_encoded\_df, X\_train.drop(categorical\_columns, axis=1)], axis=1)

X\_test\_final = pd.concat([X\_test\_encoded\_df, X\_test.drop(categorical\_columns, axis=1)], axis=1)

# Create and train a Random Forest Classifier

rf\_classifier = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_classifier.fit(X\_train\_final, y\_train)

# Make predictions on the test data

y\_pred = rf\_classifier.predict(X\_test\_final)

# Evaluate the model using accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy: {accuracy}")