import tkinter as tk

from tkinter import messagebox

import pandas as pd

import numpy as np

import pickle

from sklearn.ensemble import RandomForestClassifier

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

from sklearn.preprocessing import StandardScaler

# Load the dataset

data\_path = 'heart-disease.csv' # Path to the dataset

data = pd.read\_csv(data\_path)

# Data preprocessing

X = data.drop(columns=['target']) # Assuming 'target' is the target column

y = data['target']

# Split the dataset

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

# Standardize the features

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Train a Random Forest model

model = RandomForestClassifier(random\_state=42)

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

# Save the model and scaler to a file

with open('model.pkl', 'wb') as model\_file:

pickle.dump((model, scaler), model\_file)

# Load the trained model and scaler

with open('model.pkl', 'rb') as model\_file:

loaded\_model, loaded\_scaler = pickle.load(model\_file)

# Create the GUI application

def predict():

try:

user\_input = {feature: float(entry\_fields[feature].get()) for feature in feature\_names}

input\_df = pd.DataFrame([user\_input])

input\_scaled = loaded\_scaler.transform(input\_df)

prediction = loaded\_model.predict(input\_scaled)

prediction\_proba = loaded\_model.predict\_proba(input\_scaled)

if prediction[0] == 1:

result\_text = "The model predicts that the patient is at HIGH RISK of heart disease."

else:

result\_text = "The model predicts that the patient is at LOW RISK of heart disease."

result\_text += f"\n\nPrediction Probability:\nLow Risk: {prediction\_proba[0][0]\*100:.2f}%\nHigh Risk: {prediction\_proba[0][1]\*100:.2f}%"

messagebox.showinfo("Prediction Result", result\_text)

except ValueError as e:

messagebox.showerror("Input Error", "Please enter valid numeric values for all fields.")

# Initialize the GUI

root = tk.Tk()

root.title("Heart Disease Risk Prediction")

feature\_names = X.columns

entry\_fields = {}

# Create input fields

frame = tk.Frame(root)

frame.pack(pady=10)

for feature in feature\_names:

label = tk.Label(frame, text=f"{feature}:")

label.grid(row=feature\_names.get\_loc(feature), column=0, padx=5, pady=5, sticky='w')

entry = tk.Entry(frame)

entry.grid(row=feature\_names.get\_loc(feature), column=1, padx=5, pady=5)

entry.insert(0, f"{X[feature].mean():.2f}")

entry\_fields[feature] = entry

# Add predict button

predict\_button = tk.Button(root, text="Predict", command=predict)

predict\_button.pack(pady=10)

# Run the GUI

root.mainloop()