import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, confusion\_matrix

import pickle

# Step 1: Generate synthetic dataset

np.random.seed(42)

n\_samples = 300

# Features

age = np.random.randint(25, 80, n\_samples)

glucose = np.random.normal(130, 20, n\_samples).astype(int)

cholesterol = np.random.normal(200, 30, n\_samples).astype(int)

bmi = np.round(np.random.normal(26, 5, n\_samples), 1)

bp = np.random.randint(110, 160, n\_samples)

smoking = np.random.choice([0, 1], size=n\_samples, p=[0.7, 0.3])

family\_history = np.random.choice([0, 1], size=n\_samples, p=[0.6, 0.4])

# Disease label logic

risk = ((glucose > 140).astype(int) +(cholesterol > 220).astype(int) +

(bmi > 30).astype(int) +(bp > 140).astype(int) + smoking +family\_history)

disease = (risk >= 3).astype(int)

# Create DataFrame

df = pd.DataFrame({

'Age': age,

'Glucose': glucose,

'Cholesterol': cholesterol,

'BMI': bmi,

'BloodPressure': bp,

'Smoking': smoking,

'FamilyHistory': family\_history,

'Disease': disease

})

# Step 2: Save dataset

df.to\_csv("medical\_dataset.csv", index=False)

print("Dataset 'medical\_dataset.csv' saved.")

# Step 3: Model training

X = df.drop("Disease", axis=1)

y = df["Disease"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

model = LogisticRegression(max\_iter=1000)

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

# Step 4: Save model

pickle.dump(model, open("disease\_model.pkl", "wb"))

print("Model saved as 'disease\_model.pkl'.")

# Step 5: Evaluate model

y\_pred = model.predict(X\_test)

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

cm = confusion\_matrix(y\_test, y\_pred)

print("\n--- Model Evaluation ---")

print(f"Accuracy: {acc:.2f}")

print("Confusion Matrix:")

print(cm)

# Step 6: Predict on new input

print("\n--- Predictions ---")

test\_input = pd.DataFrame({

    'Age': [45, 35, 60],

    'Glucose': [160, 110, 145],

    'Cholesterol': [230, 180, 220],

    'BMI': [29.5, 22.0, 31.2],

    'BloodPressure': [145, 130, 150],

    'Smoking': [1, 0, 1],

    'FamilyHistory': [1, 0, 1] # Changed 'Family History' to 'FamilyHistory' to match training data

            })

predictions = model.predict(test\_input)

for i, pred in enumerate(predictions):

 result = "Disease Detected" if pred == 1 else "No Disease"

 print(f"Patient {i+1}: {result}")