# Importing necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.preprocessing import StandardScaler

from sklearn.impute import SimpleImputer

from sklearn.ensemble import RandomForestClassifier

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

# Load the dataset (replace 'your\_dataset.csv' with the actual file name)

data = pd.read\_csv('diabetes.csv')

# Display basic information about the dataset

print("Dataset Information:")

print(data.info())

# Display the first few rows of the dataset

print("\nSample Data:")

print(data.head())

# Handle missing values

imputer = SimpleImputer(strategy='mean')

# Specify the columns for imputation

selected\_columns = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']

# Check if the specified columns exist in the dataset

if all(column in data.columns for column in selected\_columns):

data[selected\_columns] = imputer.fit\_transform(data[selected\_columns])

else:

print("\nSpecified columns not found in the dataset.")

# Split the data into features (X) and target variable (y)

X = data.drop('Outcome', axis=1)

y = data['Outcome']

# 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)

# Standardize features

scaler = StandardScaler()

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

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

# Create a Random Forest Classifier

model = RandomForestClassifier(random\_state=42)

# Train the model

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

# Make predictions on the test set

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

# Evaluate the model

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

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

classification\_rep = classification\_report(y\_test, y\_pred)

# Display the evaluation results

print("\nModel Evaluation:")

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

print('Classification Report:')

print(classification\_rep)

# Interpretation

print("\nSummary:-")

print("The provided Python code uses a Random Forest classifier to build a machine learning model for classifying patients based on their risk of developing diabetes. The dataset includes features such as pregnancies, glucose levels, blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, and age. The model is trained, evaluated, and its performance is assessed using metrics such as accuracy, confusion matrix, and classification report. The goal is to assist healthcare professionals in identifying patients at elevated risk for targeted intervention and preventive measures, ultimately improving outcomes in diabetes management.")

# Prediction

print("\nForecast:")

print("For future patient data, the model can be utilized to predict the likelihood of diabetes, thereby aiding healthcare professionals in targeted intervention strategies and preventive measures.")