import pandas as pd

import numpy as np

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

# Sample dataset

data = {

    "Food": ["Apple", "Banana", "Chicken Breast", "White Rice", "Broccoli", "Chocolate", "Salmon", "Cheese"],

    "Calories": [52, 89, 165, 130, 34, 546, 208, 402],

    "Protein (g)": [0.3, 1.1, 31.0, 2.7, 2.8, 4.9, 20.4, 25],

    "Fat (g)": [0.2, 0.3, 3.6, 0.3, 0.4, 31.0, 13.0, 33],

    "Carbohydrates (g)": [14, 23, 0, 28, 7, 61, 0, 1.3],

    "Suitable for Weight Loss": [1, 1, 1, 0, 1, 0, 1, 0]

}

# Train model

df = pd.DataFrame(data)

X = df[["Calories", "Protein (g)", "Fat (g)", "Carbohydrates (g)"]]

y = df["Suitable for Weight Loss"]

model = LogisticRegression().fit(X, y)

# Model performance metrics

y\_pred = model.predict(X)

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

precision = precision\_score(y, y\_pred)

recall = recall\_score(y, y\_pred)

f1 = f1\_score(y, y\_pred)

print("\n📊 Model Performance Metrics")

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

print(f"Precision: {precision:.2f}")

print(f"Recall: {recall:.2f}")

print(f"F1 Score: {f1:.2f}")

# Prediction and explanation function

def predict\_food\_suitability(calories, protein, fat, carbs):

    input\_data = pd.DataFrame([[calories, protein, fat, carbs]],

                              columns=["Calories", "Protein (g)", "Fat (g)", "Carbohydrates (g)"])

    prediction = model.predict(input\_data)[0]

    proba = model.predict\_proba(input\_data)[0][1]

    reasons = []

    if calories < 100:

        reasons.append("low in calories")

    if fat < 5:

        reasons.append("low in fat")

    if protein > 5:

        reasons.append("high in protein")

    if carbs < 15:

        reasons.append("controlled in carbs")

    if prediction == 1:

        result = "✅ This food is suitable for weight loss."

        explanation = f"It is {', '.join(reasons)}."

    else:

        result = "❌ This food is not suitable for weight loss."

        explanation = "It may be too high in calories, fat, or carbs for a typical weight loss plan."

    return result, f"Confidence: {proba\*100:.2f}%", explanation

# Batch predictions from CSV or list of foods

def batch\_predict\_foods(food\_list):

    print("\n🍱 Batch Prediction Results:")

    for food in food\_list:

        name = food["Food"]

        result, confidence, explanation = predict\_food\_suitability(

            food["Calories"], food["Protein (g)"], food["Fat (g)"], food["Carbohydrates (g)"])

        print(f"\nFood: {name}")

        print("Result:", result)

        print(confidence)

        print("Explanation:", explanation)

# Predefined examples (replaces interactive input for sandboxed environments)

if \_\_name\_\_ == "\_\_main\_\_":

    print("\n🍽️ Smartest AI Nutrition Assistant")

    print("Evaluating predefined food examples...\n")

    food\_examples = [

        {"Food": "Oatmeal", "Calories": 68, "Protein (g)": 2.4, "Fat (g)": 1.4, "Carbohydrates (g)": 12.0},

        {"Food": "French Fries", "Calories": 312, "Protein (g)": 3.4, "Fat (g)": 15.0, "Carbohydrates (g)": 41.4},

        {"Food": "Boiled Egg", "Calories": 77, "Protein (g)": 6.3, "Fat (g)": 5.3, "Carbohydrates (g)": 0.6}

    ]

    batch\_predict\_foods(food\_examples)