Question29:

Question: Evaluation Metrics for Model Performance

You have trained a machine learning model on a dataset, and now you want to evaluate its

performance using various metrics.

Write a Python program that loads a dataset and trained model from scikit-learn. The program

should ask the user to input the names of the features and the target variable they want to use for

evaluation. The program should then calculate and display common evaluation metrics such as

accuracy, precision, recall, and F1-score for the model's predictions on the test data.

Answer:

import pandas as pd

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

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

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

# Load dataset

df = pd.read\_csv(r"C:\Users\jampa\Downloads\iris\_sample.csv")

# Show available columns

print("Available columns in the dataset:")

print(df.columns.tolist())

# Get user input

features = input("Enter feature column names (comma-separated): ").split(',')

target = input("Enter the target column name: ")

# Strip extra spaces

features = [col.strip() for col in features]

target = target.strip()

# Encode target if categorical

if df[target].dtype == 'object':

    le = LabelEncoder()

    df[target] = le.fit\_transform(df[target])

# Split into features (X) and target (y)

X = df[features]

y = df[target]

# Split into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Train model

model = RandomForestClassifier(random\_state=42)

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

# Make predictions

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

# Calculate evaluation metrics

print("\nEvaluation Metrics:")

print(f"Accuracy:  {accuracy\_score(y\_test, y\_pred):.2f}")

print(f"Precision: {precision\_score(y\_test, y\_pred, average='weighted'):.2f}")

print(f"Recall:    {recall\_score(y\_test, y\_pred, average='weighted'):.2f}")

print(f"F1 Score:  {f1\_score(y\_test, y\_pred, average='weighted'):.2f}")

Output:

