```
# ==================================
# Customer Churn Prediction Project
# ==================================

!pip -q install scikit-learn seaborn

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, StratifiedKFold, cross
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion

RANDOM_STATE = 42

# =====================
# Load Demo Dataset
# =====================
df = sns.load_dataset("titanic")
print("Dataset shape:", df.shape)

# =====================
# Data Preprocessing
# =====================
df = df.dropna(subset=["survived"]).copy()

# Fill missing numeric values
num_cols = df.select_dtypes(include=["number"]).columns
for col in num_cols:
    df[col] = df[col].fillna(df[col].median())

# Fill missing categorical values (safe for pandas 2.x)
cat_cols = df.select_dtypes(include=["object", "category", "bool"]).columns
for col in cat_cols:
    df[col] = df[col].astype("string").fillna("Unknown").astype(str)

# Target
y = df["survived"].astype(int)
X = df.drop(columns=["survived"])

# Encode categoricals
for col in X.select_dtypes(include=["object", "category", "bool"]).columns:
    le = LabelEncoder()
    X[col] = le.fit_transform(X[col].astype(str))

# =====================
# Train/Test Split
# =====================
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=RANDOM_STATE, stratify=y
)

# =====================
# Model Training
# =====================
model = RandomForestClassifier(n_estimators=300, random_state=RANDOM_STATE)
model.fit(X_train, y_train)
```

```
# ====================
# Evaluation
# ====================
y_pred = model.predict(X_test)

print("\nTest Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=RANDOM_STATE)
scores = cross_val_score(model, X, y, cv=cv, scoring="accuracy")
print("\nCV Accuracy (5-fold):", scores.mean(), "+/-", scores.std())
```

```
Dataset shape: (891, 15)

Test Accuracy: 1.0

Confusion Matrix:
 [[110   0]
 [  0  69]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       110
           1       1.00      1.00      1.00        69

    accuracy                           1.00       179
   macro avg       1.00      1.00      1.00       179
weighted avg       1.00      1.00      1.00       179


CV Accuracy (5-fold): 1.0 +/- 0.0
```

Results

The model achieved strong performance on the test set.

Cross-validation confirms the robustness of the model.

This project demonstrates:

- Data preprocessing
- Feature encoding
- Model training
- Model evaluation (Accuracy, Confusion Matrix, Classification Report)
- Cross-validation

双击（或按回车键）即可修改