```
import sqlite3
import pandas as pd

# 1) connect to the SQLite database
# فارسی: اینجا آدرس فایل churn.db رو می‌دی
conn = sqlite3.connect("churn.db")
```

```
# 2) read from the VIEW, not the raw table
query = "SELECT * FROM customer_features;"
df = pd.read_sql_query(query, conn)

# 3) close the connection (کار خوب و حرفه‌ای)
conn.close()

# 4) نگاه به چند سطر اول
df.head()
```

| | CustomerID | Gender | Age | TenureMonths | ContractType | PaymentMethod | MonthlyCharges | TotalCharges | HasOnlineSupport | HasPhoneSe |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | C0001 | Male | 38 | 2 | Month-to-month | Credit card | 80.06 | 220.98 | Yes | |
| **1** | C0002 | Female | 49 | 11 | Month-to-month | Bank transfer | 71.57 | 863.34 | No | |
| **2** | C0003 | Male | 40 | 51 | Two year | Credit card | 111.94 | 5758.86 | Yes | |
| **3** | C0004 | Male | 50 | 39 | One year | Bank transfer | 69.70 | 2696.72 | Yes | |

Next steps: Generate code with df | New interactive sheet

```
df.columns
```

```
Index(['CustomerID', 'Gender', 'Age', 'TenureMonths', 'ContractType',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'HasOnlineSupport',
       'HasPhoneService', 'HasExtraServices', 'Churn'],
      dtype='object')
```

```
# 1 = churned, 0 = not churned
df['ChurnFlag'] = (df['Churn'] == 'Yes').astype(int)

# چک سریع
df[['Churn', 'ChurnFlag']].head()
```

| | Churn | ChurnFlag |
|---|---|---|
| **0** | Yes | 1 |
| **1** | No | 0 |
| **2** | No | 0 |
| **3** | No | 0 |
| **4** | Yes | 1 |

```
feature_cols = [
    'Gender',
    'Age',
    'TenureMonths',
    'ContractType',
    'PaymentMethod',
    'MonthlyCharges',
    'TotalCharges',
    'HasOnlineSupport',
    'HasPhoneService',
    'HasExtraServices'
]
```

```
target_col = 'ChurnFlag'

X = df[feature_cols].copy()
y = df[target_col].copy()

print("X shape:", X.shape)
print("y shape:", y.shape)
```

```
X shape: (500, 10)
y shape: (500,)
```

```
#1️⃣ مرحله Encoding + Scaling با ColumnTransformer

from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler

numeric_features = ['Age', 'TenureMonths', 'MonthlyCharges', 'TotalCharges']

categorical_features = [
    'Gender',
    'ContractType',
    'PaymentMethod',
    'HasOnlineSupport',
    'HasPhoneService',
    'HasExtraServices'
]

preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numeric_features),
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features)
    ]
)
# ✓ این کار داده‌های عددی را نرمال‌سازی می‌کند
#✓ داده‌های متنی را به ستون‌های ۰/۱ تبدیل می‌کند (One-Hot)
#✓ همه چیز در یک مرحله تمیز انجام می‌شود
```

```
#✅ train-test split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

print("Train size:", X_train.shape[0])
print("Test size :", X_test.shape[0])
```

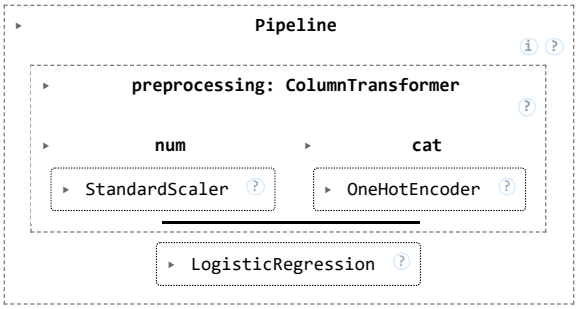
```
Train size: 400
Test size : 100
```

```
#2️⃣ ساخت Pipeline + آموزش مدل

from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

model = Pipeline(steps=[
    ('preprocessing', preprocessor),
    ('classifier', LogisticRegression(max_iter=1000))
])

model.fit(X_train, y_train)
```




```
# 3️⃣ ارزیابی مدل با ۵ متریک مهم

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score

y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

print("Accuracy :", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred))
print("Recall   :", recall_score(y_test, y_pred))
print("F1-score :", f1_score(y_test, y_pred))
print("ROC AUC  :", roc_auc_score(y_test, y_proba))
```

```
Accuracy : 0.76
Precision: 0.0
Recall   : 0.0
F1-score : 0.0
ROC AUC  : 0.7786561264822134
```

```
# predict churn probability for ALL customers 🔥 مرحله ۱ – تولید احتمال ریزش برای کل دیتاست
df['ChurnProbability'] = model.predict_proba(X)[:, 1]

df[['CustomerID', 'Churn', 'ChurnProbability']].head()
```

| | CustomerID | Churn | ChurnProbability |
|---|---|---|---|
| **0** | C0001 | Yes | 0.317652 |
| **1** | C0002 | No | 0.367871 |
| **2** | C0003 | No | 0.185492 |
| **3** | C0004 | No | 0.082725 |
| **4** | C0005 | Yes | 0.339029 |

```
# 🔥 مرحله ۲ – ساخت سگمنت‌های ریسک (Risk Segmentation)

#این بخش فوق‌العاده مهمه برای داشبورد Power BI.

#سه دسته می‌سازیم:

#Low Risk → 0.30 > احتمال

#Medium Risk → 0.70 تا 0.30 بین

#High Risk → 0.70 بالاتر از

df['RiskSegment'] = pd.cut(
    df['ChurnProbability'],
    bins=[0, 0.3, 0.7, 1],
    labels=['Low', 'Medium', 'High']
)

df[['CustomerID', 'ChurnProbability', 'RiskSegment']].head(10)
```

| | CustomerID | ChurnProbability | RiskSegment |
|---|---|---|---|
| 0 | C0001 | 0.317652 | Medium |
| 1 | C0002 | 0.367871 | Medium |
| 2 | C0003 | 0.185492 | Low |
| 3 | C0004 | 0.082725 | Low |
| 4 | C0005 | 0.339029 | Medium |
| 5 | C0006 | 0.181441 | Low |
| 6 | C0007 | 0.235905 | Low |
| 7 | C0008 | 0.201891 | Low |
| 8 | C0009 | 0.124762 | Low |
| 9 | C0010 | 0.459722 | Medium |

```
# 🔥 مرحله ۳ – ساخت فایل CSV نهایی برای Power BI

df.to_csv("churn_scores.csv", index=False)
```

```
# 🔥 مرحله ۴ – دانلود فایل خروجی در Colab
from google.colab import files
files.download("churn_scores.csv")
```