Customer Purchase Prediction using Decision Tree Classifier

Name: Tarun Chincholia
Internship Domain: Data Science
Task Number: Task 3
Organization: Prodigy InfoTech
Date: 26-07-2025

Objective:

To build a machine learning model that predicts whether a customer will subscribe to a term deposit (purchase a financial product), based on their demographic and behavioral data.

Technologies Used:

Python
Pandas, NumPy – data preprocessing
Scikit-learn – model building and evaluation
Matplotlib, Seaborn – visualization

Data Used:

Source:UCI Machine Learning Repository

Download Link (Direct):https://github.com/Prodigy-InfoTech/data-science-datasets/tree/main/Task%203

In [ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder
```

In [ ]:
```
df = pd.read_csv('Customer-Churn-Records.csv')
```

In [ ]:
```
df.head(10)
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	T
0	1	15634602	Hargrave	619	France	Female	42	
1	2	15647311	Hill	608	Spain	Female	41	
2	3	15619304	Onio	502	France	Female	42	
3	4	15701354	Boni	699	France	Female	39	
4	5	15737888	Mitchell	850	Spain	Female	43	
5	6	15574012	Chu	645	Spain	Male	44	
6	7	15592531	Bartlett	822	France	Male	50	
7	8	15656148	Obinna	376	Germany	Female	29	
8	9	15792365	He	501	France	Male	44	
9	10	15592389	H?	684	France	Male	27	

In [ ]:
```
data = df.drop(['RowNumber', 'Surname', 'Geography', 'Gender', 'Card Type'], a
data.head()
```

Out[ ]:

	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCar
0	15634602	619	42	2	0.00	1	1
1	15647311	608	41	1	83807.86	1	0
2	15619304	502	42	8	159660.80	3	1
3	15701354	699	39	1	0.00	2	0
4	15737888	850	43	2	125510.82	1	1

In [ ]:
```
data.duplicated().sum()
```

Out[ ]:
```
0
```

In [ ]:
```
data.isnull().sum()
```

```
Out[ ]: CustomerId            0
        CreditScore           0
        Age                   0
        Tenure                0
        Balance               0
        NumOfProducts         0
        HasCrCard             0
        IsActiveMember        0
        EstimatedSalary       0
        Exited                0
        Complain              0
        Satisfaction Score    0
        Point Earned          0
        dtype: int64
```

```
In [ ]: data.shape
```

```
Out[ ]: (10000, 13)
```

```
In [ ]: threshold_balance = data["Balance"].mean()
        threshold_active = data["IsActiveMember"].mean()




        def create_purchase_label(row):
            if row['IsActiveMember'] > threshold_active and row['Balance'] > threshold
                return 1
            else:
                return 0
```

```
In [ ]: data['PurchaseLabel'] = data.apply(create_purchase_label, axis=1)
```

```
In [ ]: print(data[['Balance', 'IsActiveMember', 'PurchaseLabel']])
```

```
            Balance  IsActiveMember  PurchaseLabel
      0        0.00               1              0
      1    83807.86               1              1
      2   159660.80               0              0
      3        0.00               0              0
      4   125510.82               1              1
      ...       ...             ...            ...
      9995      0.00               0              0
      9996  57369.61               1              0
      9997      0.00               1              0
      9998  75075.31               0              0
      9999 130142.79               0              0

      [10000 rows x 3 columns]
```

```
In [ ]: (data['PurchaseLabel']).sum()
```

```
Out[ ]: 3016
```

```
In [ ]: y = data['PurchaseLabel']
        x = data.drop(['PurchaseLabel'] , axis=1 )
```

```
In [ ]: x.head()
```

Out[ ]:

	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCarc
0	15634602	619	42	2	0.00	1	1
1	15647311	608	41	1	83807.86	1	(
2	15619304	502	42	8	159660.80	3	1
3	15701354	699	39	1	0.00	2	(
4	15737888	850	43	2	125510.82	1	1

```
In [ ]: y.head()
```

```
Out[ ]: 0    0
        1    1
        2    0
        3    0
        4    1
        Name: PurchaseLabel, dtype: int64
```

```
In [ ]: X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.35, ranc
```

```
In [ ]: clf = DecisionTreeClassifier(random_state=42)
```

```
In [ ]: clf.fit(X_train, y_train)
```

```
Out[ ]: ▾         DecisionTreeClassifier
        DecisionTreeClassifier(random_state=42)
```

```
In [ ]: y_pred = clf.predict(X_test)
```

```
In [ ]: accuracy = accuracy_score(y_test, y_pred)
        print("Accuracy:", accuracy)
```

```
Accuracy: 0.9997142857142857
```

```
In [ ]: print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      2426
           1       1.00      1.00      1.00      1074

    accuracy                           1.00      3500
   macro avg       1.00      1.00      1.00      3500
weighted avg       1.00      1.00      1.00      3500
```

In [ ]:

```
from sklearn.tree import export_graphviz
import graphviz

dot_data = export_graphviz(clf, out_file=None, feature_names=x.columns, class_
graph = graphviz.Source(dot_data)
graph.render("decision_tree")
```

Out[ ]:

```
'decision_tree.pdf'
```