

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv('/content/drive/MyDrive/online_shoppers_intention.csv')
```

```
print(df.head()) # Show first few rows
print(df.info()) # Overview: data types, nulls
print(df['Revenue'].value_counts()) # Target class distribution
```

```
↩
```

	Administrative	Administrative_Duration	Informational	\
0	0	0.0	0	
1	0	0.0	0	
2	0	0.0	0	
3	0	0.0	0	
4	0	0.0	0	

  

	Informational_Duration	ProductRelated	ProductRelated_Duration	\
0	0.0	1	0.000000	
1	0.0	2	64.000000	
2	0.0	1	0.000000	
3	0.0	2	2.666667	
4	0.0	10	627.500000	

  

	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	\
0	0.20	0.20	0.0	0.0	Feb		1
1	0.00	0.10	0.0	0.0	Feb		2
2	0.20	0.20	0.0	0.0	Feb		4
3	0.05	0.14	0.0	0.0	Feb		3
4	0.02	0.05	0.0	0.0	Feb		3

  

	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
0	1	1	1	Returning_Visitor	False	False
1	2	1	2	Returning_Visitor	False	False
2	1	9	3	Returning_Visitor	False	False
3	2	2	4	Returning_Visitor	False	False
4	3	1	4	Returning_Visitor	True	False

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Administrative                        12330 non-null  int64
1   Administrative_Duration               12330 non-null  float64
2   Informational                         12330 non-null  int64
3   Informational_Duration                12330 non-null  float64
4   ProductRelated                       12330 non-null  int64
5   ProductRelated_Duration              12330 non-null  float64
6   BounceRates                          12330 non-null  float64
7   ExitRates                           12330 non-null  float64
8   PageValues                           12330 non-null  float64
9   SpecialDay                           12330 non-null  float64
10  Month                                12330 non-null  object
11  OperatingSystems                     12330 non-null  int64
12  Browser                              12330 non-null  int64
13  Region                              12330 non-null  int64
14  TrafficType                          12330 non-null  int64
15  VisitorType                          12330 non-null  object
16  Weekend                              12330 non-null  bool
17  Revenue                              12330 non-null  bool
dtypes: bool(2), float64(7), int64(7), object(2)
memory usage: 1.5+ MB
None
Revenue
False    10422
True      1908
Name: count, dtype: int64
```

```
# Step 4: Store target before encoding
y = df['Revenue'].astype(int) # Convert True/False to 1/0
```

```
# Step 5: Encode the rest of the dataset
X = pd.get_dummies(df.drop('Revenue', axis=1), drop_first=True)
```

```
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.3, random_state=42, stratify=y)
```

```
dt_model = DecisionTreeClassifier(max_depth=5, random_state=42)
dt_model.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=5, random_state=42)
```

```
y_pred = dt_model.predict(X_test)
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.895647472289808
```

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

    0       0.92      0.96      0.94      3127
    1       0.72      0.53      0.61       572

 accuracy      0.90      3699
 macro avg     0.82      0.75      0.78      3699
 weighted avg  0.89      0.90      0.89      3699
```

```
Confusion Matrix:
[[3007 120]
 [ 266 306]]
```

```
plt.figure(figsize=(20,10))
plot_tree(dt_model, feature_names=X.columns, class_names=["No", "Yes"], filled=True, fontsize=8)
plt.title("Decision Tree for Online Shoppers' Purchase Prediction")
plt.show()
```



