Task 3

Build a decision tree classifier to predict whether a customer will purchase a product or service based on their demographic and behavioral data. Use a dataset such as the Bank Marketing dataset from the UCI Machine Learning Repository.

Step 1: Import Required Libraries

```
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
```

Step 2: Load the Dataset

```
data = pd.read_csv("/content/BMW car.csv")
data.head(5)
```

	Model	Year	Region	Color	Fuel_Type	Transmission	Engine_Size_L	Mileage_KM	Price_USD	Sales_Volume	Sales_Classification
0	5 Series	2016	Asia	Red	Petrol	Manual	3.5	151748	98740	8300	High
1	i8	2013	North America	Red	Hybrid	Automatic	1.6	121671	79219	3428	Low
2	5 Series	2022	North America	Blue	Petrol	Automatic	4.5	10991	113265	6994	Low
3	X3	2024	Middle East	Blue	Petrol	Automatic	1.7	27255	60971	4047	Low
4	7 Series	2020	South America	Black	Diesel	Manual	2.1	122131	49898	3080	Low

Step 3: Data Cleaning

```
data.dropna(inplace=True)

data.drop_duplicates(inplace=True)
```

Step 4: Encode Categorical Variables

```
le = LabelEncoder()

categorical_cols = ['Region', 'Fuel_Type', 'Transmission', 'Sales_Classification']

for col in categorical_cols:
    data[col] = le.fit_transform(data[col])
```

Step 5: Define Features & Target

```
X = data[['Year', 'Region', 'Fuel_Type', 'Transmission',
        'Engine_Size_L', 'Mileage_KM', 'Price_USD', 'Sales_Volume']]

y = data['Sales_Classification']
```

Step 6: Train-Test Split

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

Step 7: Build Decision Tree Model

```
model = DecisionTreeClassifier(criterion='gini', max_depth=5)
model.fit(X_train, y_train)
```

DecisionTreeClassifier

DecisionTreeClassifier(max_depth=5)

Step 8: Make Predictions

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

Step 9: Evaluate Model Performance

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

print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 1.0

Classification Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      3032
           1       1.00      1.00      1.00      6968

    accuracy                           1.00     10000
   macro avg       1.00      1.00      1.00     10000
weighted avg       1.00      1.00      1.00     10000
```