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

Step 1: Import Required Libraries

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

Step 2: Load the Pizza Dataset

```
data=pd.read_csv("/content/pizza - pizza.csv")
data.head(5)
```

	order_details_id	order_id	pizza_id	quantity	order_date	order_time	unit_price	total_price	pizza_size	pizza_category	pizza_ingredients	pizza_name
0	1	1	hawaiian_m	1	1/1/2015	11:38:36	13.25	13.25	M	Classic	Sliced Ham, Pineapple, Mozzarella Cheese	The Hawaiian Pizza
1	2	2	classic_dlx_m	1	1/1/2015	11:57:40	16.00	16.00	M	Classic	Pepperoni, Mushrooms, Red Onions, Red Peppers,...	The Classic Deluxe Pizza
2	3	2	five_cheese_l	1	1/1/2015	11:57:40	18.50	18.50	L	Veggie	Mozzarella Cheese, Provolone Cheese, Smoked Go...	The Five Cheese Pizza
3	4	2	ital_supr_l	1	1/1/2015	11:57:40	20.75	20.75	L	Supreme	Calabrese Salami, Capocollo, Tomatoes, Red Oni...	The Italian Supreme Pizza
4	5	2	mexicana_m	1	1/1/2015	11:57:40	16.00	16.00	M	Veggie	Tomatoes, Red Peppers, Jalapeno Peppers, Red O...	The Mexicana Pizza

Step 2: Create Target Variable (Purchase or Not)

```
# Create binary target column
average_price = data['total_price'].mean()

data['purchase'] = data['total_price'].apply(
    lambda x: 1 if x >= average_price else 0
)
```

Step 3: Select Features (Behavioral Data)

```
X = data[['quantity', 'unit_price', 'pizza_size', 'pizza_category']]
y = data['purchase']
```

Step 4: Encode Categorical Variables

```
le = LabelEncoder()
X['pizza_size'] = le.fit_transform(X['pizza_size'])
X['pizza_category'] = le.fit_transform(X['pizza_category'])
```

```
/tmp/ipython-input-1833330895.py:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipython-input-1833330895.py:3: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Step 5: Train-Test Split

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

Step 6: Build Decision Tree Classifier

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

DecisionTreeClassifier

```
DecisionTreeClassifier(max_depth=5)
```

Step 7: Make Predictions

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

Step 8: Evaluate the Model

```
print("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))
```

```
Accuracy: 1.0

Confusion Matrix:
 [[6063    0]
 [   0 3661]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6063
           1       1.00      1.00      1.00      3661

    accuracy                           1.00      9724
   macro avg       1.00      1.00      1.00      9724
weighted avg       1.00      1.00      1.00      9724
```