In [1]:

```
"""
# Predictive Analysis with Decision Trees
-----------------------------------------------
## Problem Statement
Decision Trees are simple yet powerful supervised learning algorithms used for
both classification and regression tasks. However, they are prone to overfitti
This project implements Decision Tree models and applies pruning techniques
to improve generalization and predictive performance.
-------------------------------------------------
## Objectives
- Implement Decision Trees for classification and regression
- Understand and apply Entropy and Gini Index for splitting
- Apply pre-pruning and post-pruning to avoid overfitting
- Visualize the decision tree and interpret feature importance
------------------------------------------------------------
## Decision Tree Construction
A Decision Tree splits data recursively based on feature values.
Each split aims to maximize purity of the target variable.
----------------------------------------------------------
### Entropy
Entropy measures the randomness or impurity in data.
Entropy(S) = − Σ pᵢ log₂(pᵢ)
Lower entropy indicates purer data.
----------------------------------------------------------
### Gini Index
Gini Index measures the probability of incorrect classification.
Gini(S) = 1 − Σ pᵢ²
Lower Gini value indicates better split quality.
-----------------------------------------------------------
### Overfitting
Decision Trees may memorize training data.
Pruning techniques help reduce overfitting and improve generalization.
"""
```

Out[1]:

```
'\n# Predictive Analysis with Decision Trees\
n----------------------------------------------\n## Problem Statement\nDecis
ion Trees are simple yet powerful supervised learning algorithms used for\nbo
th classification and regression tasks. However, they are prone to overfittin
g.\nThis project implements Decision Tree models and applies pruning techniqu
es\nto improve generalization and predictive performanc
e.\n---------------------------------------------------\n## Objectives\n- Imple
ment Decision Trees for classification and regression\n- Understand and apply
Entropy and Gini Index for splitting\n- Apply pre-pruning and post-pruning to
avoid overfitting\n- Visualize the decision tree and interpret feature import
ance\n----------------------------------------------------------\n## Decisi
on Tree Construction\nA Decision Tree splits data recursively based on featur
e values.\nEach split aims to maximize purity of the target variabl
e.\n-----------------------------------------------------------\n### Entropy\n
Entropy measures the randomness or impurity in data.\nEntropy(S) = − Σ pᵢ log
₂(pᵢ)\nLower entropy indicates purer dat
a.\n-----------------------------------------------------------\n### Gini Inde
x\nGini Index measures the probability of incorrect classification.\nGini(S)
= 1 − Σ pᵢ²\nLower Gini value indicates better split qualit
y.\n------------------------------------------------------------\n### Overfitt
ing\nDecision Trees may memorize training data.\nPruning techniques help redu
ce overfitting and improve generalization.\n'
```

In [2]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, plot_t
from sklearn.metrics import accuracy_score, mean_squared_error
```

In [4]:

```python
# Load dataset
data = pd.read_csv(r"C:\Users\karpa\Downloads\customer_purchase_decision.csv")
data
```

Out[4]:

	Age	Income	Student	Credit_Rating	Purchase
0	22	Low	Yes	Fair	Yes
1	25	Low	Yes	Excellent	Yes
2	28	Medium	Yes	Fair	Yes
3	32	Medium	No	Fair	Yes
4	35	High	No	Fair	No
5	40	High	No	Excellent	No
6	45	Medium	No	Excellent	No
7	23	Low	No	Fair	No
8	27	Medium	Yes	Excellent	Yes
9	30	High	Yes	Fair	Yes
10	34	High	No	Fair	No
11	38	Medium	No	Fair	Yes
12	42	Low	Yes	Excellent	Yes
13	48	High	No	Excellent	No
14	50	Medium	No	Fair	No
15	29	Low	Yes	Fair	Yes
16	31	Medium	No	Excellent	No
17	36	High	Yes	Excellent	Yes
18	41	Low	No	Fair	No
19	46	Medium	Yes	Fair	Yes

In [5]:

```
# Convert categorical variables into numerical format
data_encoded = pd.get_dummies(data)
# Define features and target
X = data_encoded.drop("Purchase_Yes", axis=1)
y = data_encoded["Purchase_Yes"]
# Split data
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.2, random_state=42
)
X_train.shape, X_test.shape
```

Out[5]: ((16, 9), (4, 9))

In [6]:

```
dt_gini = DecisionTreeClassifier(criterion="gini", random_state=42)
dt_gini.fit(X_train, y_train)
y_pred_gini = dt_gini.predict(X_test)
print("Classification Accuracy (Gini Index):",
```

```
accuracy_score(y_test, y_pred_gini))
```

```
Classification Accuracy (Gini Index): 1.0
```

In [7]:
```
dt_entropy = DecisionTreeClassifier(criterion="entropy", random_state=42)
dt_entropy.fit(X_train, y_train)
y_pred_entropy = dt_entropy.predict(X_test)
print("Classification Accuracy (Entropy):",
accuracy_score(y_test, y_pred_entropy))
```

```
Classification Accuracy (Entropy): 1.0
```

In [8]:
```
dt_regressor = DecisionTreeRegressor(random_state=42)
dt_regressor.fit(X_train, y_train)
y_pred_reg = dt_regressor.predict(X_test)
print("Mean Squared Error (Regression):",
mean_squared_error(y_test, y_pred_reg))
```

```
Mean Squared Error (Regression): 0.0
```

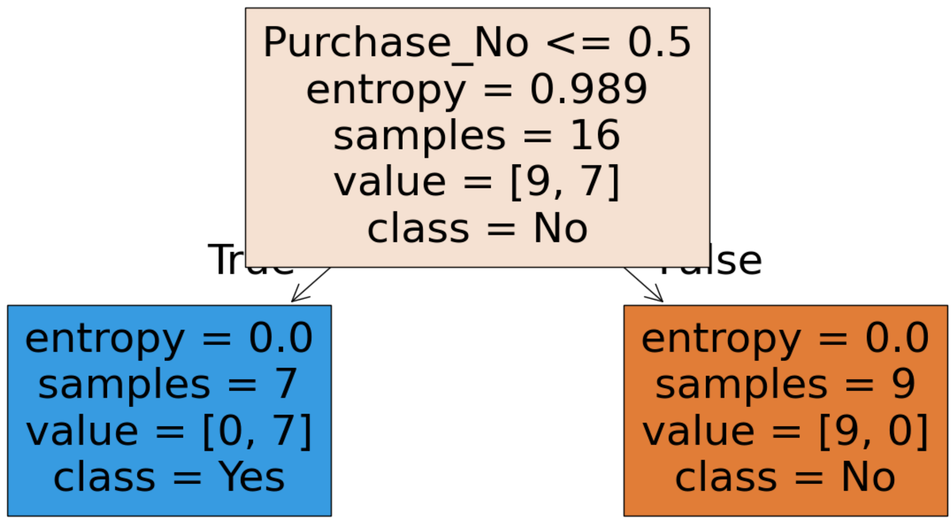
In [9]:
```
dt_prepruned = DecisionTreeClassifier(
criterion="entropy",
max_depth=3,
min_samples_split=2,
random_state=42
)
dt_prepruned.fit(X_train, y_train)
y_pred_prepruned = dt_prepruned.predict(X_test)
print("Accuracy after Pre-Pruning:",
accuracy_score(y_test, y_pred_prepruned))
```

```
Accuracy after Pre-Pruning: 1.0
```

In [10]:
```
path = dt_gini.cost_complexity_pruning_path(X_train, y_train)
ccp_alpha = path.ccp_alphas[1]
dt_postpruned = DecisionTreeClassifier(
random_state=42,
ccp_alpha=ccp_alpha
)
dt_postpruned.fit(X_train, y_train)
y_pred_postpruned = dt_postpruned.predict(X_test)
print("Accuracy after Post-Pruning:",
accuracy_score(y_test, y_pred_postpruned))
```

```
Accuracy after Post-Pruning: 0.0
```

In [11]:
```
plt.figure(figsize=(18, 8))
plot_tree(
dt_prepruned,
feature_names=X.columns,
class_names=["No", "Yes"],
filled=True
)
plt.show()
```



In [12]:
```
feature_importance = pd.Series(
dt_prepruned.feature_importances_,
index=X.columns
).sort_values(ascending=False)
feature_importance
```

Out[12]:
```
Purchase_No                 1.0
Age                         0.0
Income_High                 0.0
Income_Medium               0.0
Income_Low                  0.0
Student_No                  0.0
Student_Yes                 0.0
Credit_Rating_Excellent     0.0
Credit_Rating_Fair          0.0
dtype: float64
```

In [13]:
```
"""
Conclusion
- Decision Trees were successfully implemented for classification and regressi
- Entropy and Gini Index were used to evaluate split quality.
- Pre-pruning and post-pruning effectively reduced overfitting.
- Tree visualization improved interpretability.
- Feature importance helped identify key predictors.
"""
```

Out[13]:
```
'\nConclusion\n- Decision Trees were successfully implemented for classificat
ion and regression.\n- Entropy and Gini Index were used to evaluate split qua
lity.\n- Pre-pruning and post-pruning effectively reduced overfitting.\n- Tre
e visualization improved interpretability.\n- Feature importance helped ident
ify key predictors.\n'
```

In [ ]:
```
```