In [2]:
```
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, plot_tree
from sklearn.metrics import classification_report, accuracy_score
from sklearn.preprocessing import LabelEncoder
```

In [3]:
```
df = pd.read_csv('carsdata.csv')
```

In [4]:
```
print("Dataset Head:\n", df.head())
print("\nDataset Info:\n")
print(df.info())
```

```
Dataset Head:
   Model  Engine SC/Turbo   Weight Fuel Economy Fast
0    M1   Small       No  Average         Good   No
1    M2   Small       No    Light      Average   No
2    M3   Small      Yes  Average          Bad  Yes
3    M4  Medium       No    Heavy          Bad  Yes
4    M5   Large       No  Average          Bad  Yes

Dataset Info:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Model         15 non-null     object
 1   Engine        15 non-null     object
 2   SC/Turbo      15 non-null     object
 3   Weight        15 non-null     object
 4   Fuel Economy  15 non-null     object
 5   Fast          15 non-null     object
dtypes: object(6)
memory usage: 852.0+ bytes
None
```

In [5]:
```
label_encoders = {}
for column in df.select_dtypes(include=['object']).columns:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
```

In [6]:
```
X = df.drop('Car_Name', axis=1, errors='ignore')  # Drop Car_Name if it exists
y = df.iloc[:, -1]  # Assuming the last column is the target
```

In [7]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [8]:
```
# Create and train the Decision Tree Classifier
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)
```

Out[8]:
```
▾        DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

In [9]:
```
# Predict on test data
y_pred = model.predict(X_test)
```

In [10]:
```
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

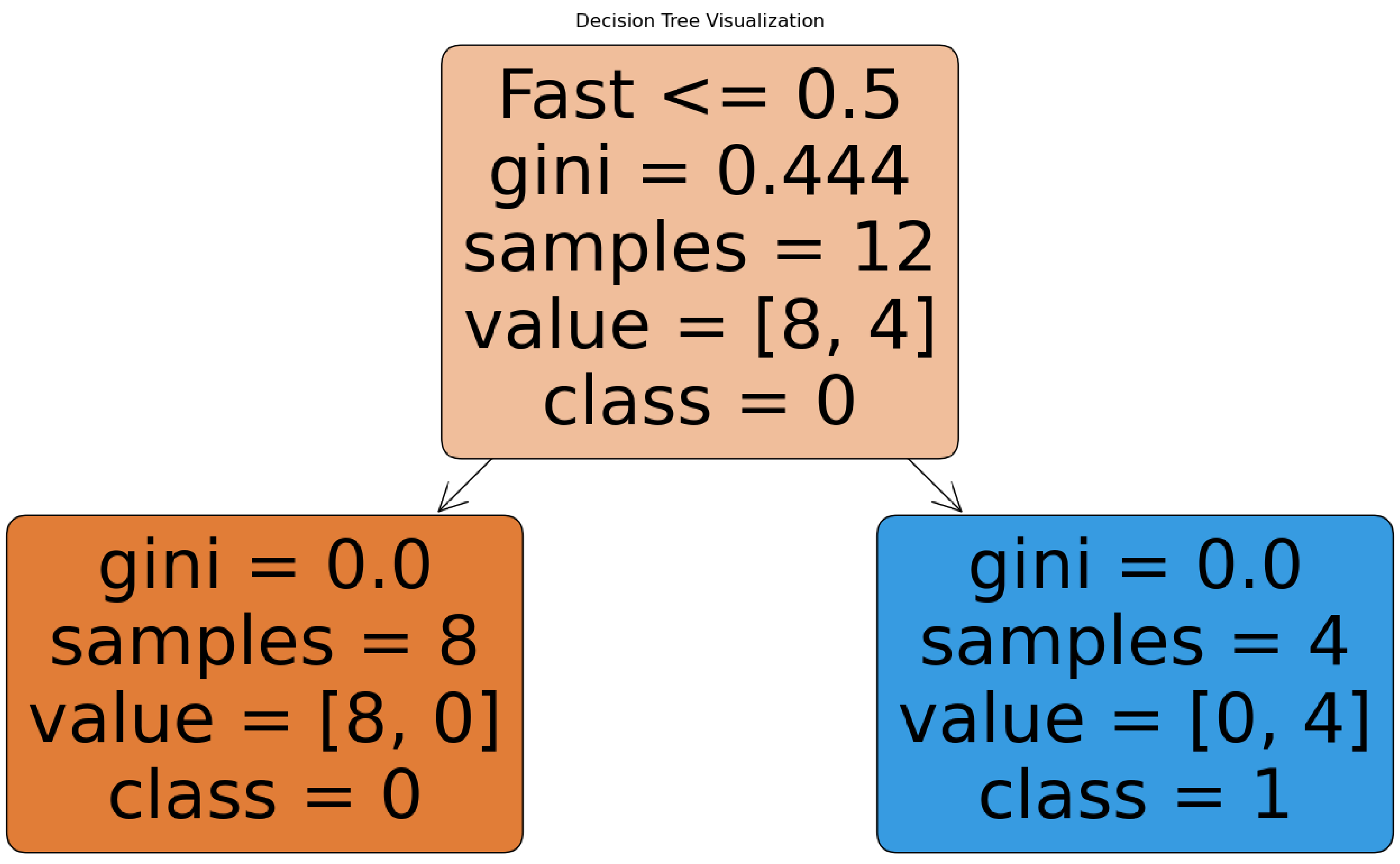
```
Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         1

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3

Accuracy Score: 1.0
```

In [13]:
```
# Visualize the Decision Tree
plt.figure(figsize=(20,10))
plot_tree(
    model,
    feature_names=X.columns.tolist(),
    class_names=[str(cls) for cls in np.unique(y)],
    filled=True,
    rounded=True
)
plt.title("Decision Tree Visualization")
plt.show()
```



In [ ]: