In [1]:
```
#IMPORT ALL LIBRARIES I WILL NEED.
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
```

In [2]:
```
#LOAD THE DATASET.
dataset=pd.read_excel('cleaned_dataset.xlsx')
```

In [3]:
```
#DISPLAY DATA TYPES OF COLUMNS.
dataset.dtypes
```

Out[3]:
```
Flight ID                          int64
Airline                           object
Flight_Distance                    int64
Origin_Airport                    object
Destination_Airport               object
Scheduled_Departure_Time           int64
Day_of_Week                        int64
Month                              int64
Airplane_Type                     object
Weather_Score                    float64
Previous_Flight_Delay_Minutes    float64
Airline_Rating                   float64
Passenger_Load                   float64
Flight_Cancelled                   int64
dtype: object
```

In [4]:
```
#CONVERT ALL CATEGORICAL VARIABLES TO NUMERIC VARIABLES.
from sklearn.preprocessing import LabelEncoder
label_encoder=LabelEncoder()
```

In [5]:
```
dataset['Airline']=label_encoder.fit_transform(dataset['Airline'])
dataset['Origin_Airport']=label_encoder.fit_transform(dataset['Origin_Airport'])
dataset['Destination_Airport']=label_encoder.fit_transform(dataset['Destination_Air
dataset['Airplane_Type']=label_encoder.fit_transform(dataset['Airplane_Type'])
```

In [6]:
```
#DISPLAY DATA TYPES AGAIN TO ENSURE ALL VARIABLES ARE NOW NUMERIC.
dataset.dtypes
```

Out[6]:
```
Flight ID                          int64
Airline                            int32
Flight_Distance                    int64
Origin_Airport                     int32
Destination_Airport                int32
Scheduled_Departure_Time           int64
Day_of_Week                        int64
Month                              int64
Airplane_Type                      int32
Weather_Score                    float64
Previous_Flight_Delay_Minutes    float64
Airline_Rating                   float64
Passenger_Load                   float64
Flight_Cancelled                   int64
dtype: object
```

In [7]:
```
#HANDLE MISSING VALUES.
dataset=dataset.fillna(dataset.mean())
```

In [8]:
```
#DEFINE FEATURES(X) AND TARGET(Y).
x=dataset.drop('Weather_Score', axis=1)
y=dataset['Flight_Cancelled']
```

In [9]:
```
#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
```

In [10]:
```
#INITIALIZING THE DECISION TREE REGRESSOR.
model=DecisionTreeRegressor(random_state=42)
```

In [11]:
```
#TRAIN THE MODEL.
model.fit(x_train, y_train)
```

Out[11]:
```
DecisionTreeRegressor
DecisionTreeRegressor(random_state=42)
```

In [12]:
```
#MAKE PREDICTIONS ON THE TEST SET.
y_pred=model.predict(x_test)
```

In [13]:
```
#CALCULATE THE MEAN SQUARED ERROR.
mse=mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
```

```
Mean Squared Error: 0.0
```

In [14]:
```
#CALCULATE THE R-SQUARED.
r2=r2_score(y_test, y_pred)
print(f'R-squared: {r2}')
```

```
R-squared: 1.0
```

In [17]:
```
#IMPORT LIBRARIES.
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
```

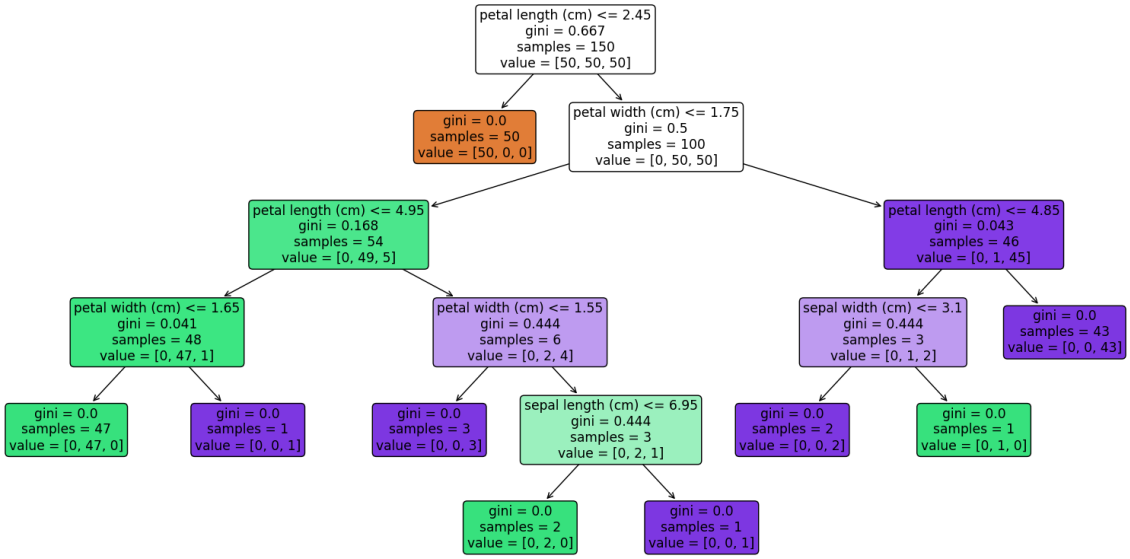
In [18]:
```
#LOAD THE DATASET.
iris=load_iris()
x, y=iris.data, iris.target
```

In [19]:
```
#TRAIN A DECISION TREE CLASSIFIER.
clf=DecisionTreeClassifier()
clf.fit(x, y)
```

Out[19]:
```
DecisionTreeClassifier
DecisionTreeClassifier()
```

In [20]:
```
#DEFINE FEATURE NAMES( MAKE SURE ITS A LIST OF STRINGS).
feature_names=iris.feature_names
```

In [21]:
```
#PLOT THE TREE.
plt.figure(figsize=(20, 10))
plot_tree(clf, feature_names=feature_names, filled=True, rounded=True)
plt.show()
```


petal length (cm) <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
gini = 0.0
samples = 50
value = [50, 0, 0]
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
petal length (cm) <= 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
petal length (cm) <= 4.85
gini = 0.043
samples = 46
value = [0, 1, 45]
petal width (cm) <= 1.65
gini = 0.041
samples = 48
value = [0, 47, 1]
petal width (cm) <= 1.55
gini = 0.444
samples = 6
value = [0, 2, 4]
sepal width (cm) <= 3.1
gini = 0.444
samples = 3
value = [0, 1, 2]
gini = 0.0
samples = 43
value = [0, 0, 43]
gini = 0.0
samples = 47
value = [0, 47, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]
gini = 0.0
samples = 3
value = [0, 0, 3]
sepal length (cm) <= 6.95
gini = 0.444
samples = 3
value = [0, 2, 1]
gini = 0.0
samples = 2
value = [0, 0, 2]
gini = 0.0
samples = 1
value = [0, 1, 0]
gini = 0.0
samples = 2
value = [0, 2, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]

In [ ]: