Decision Trees

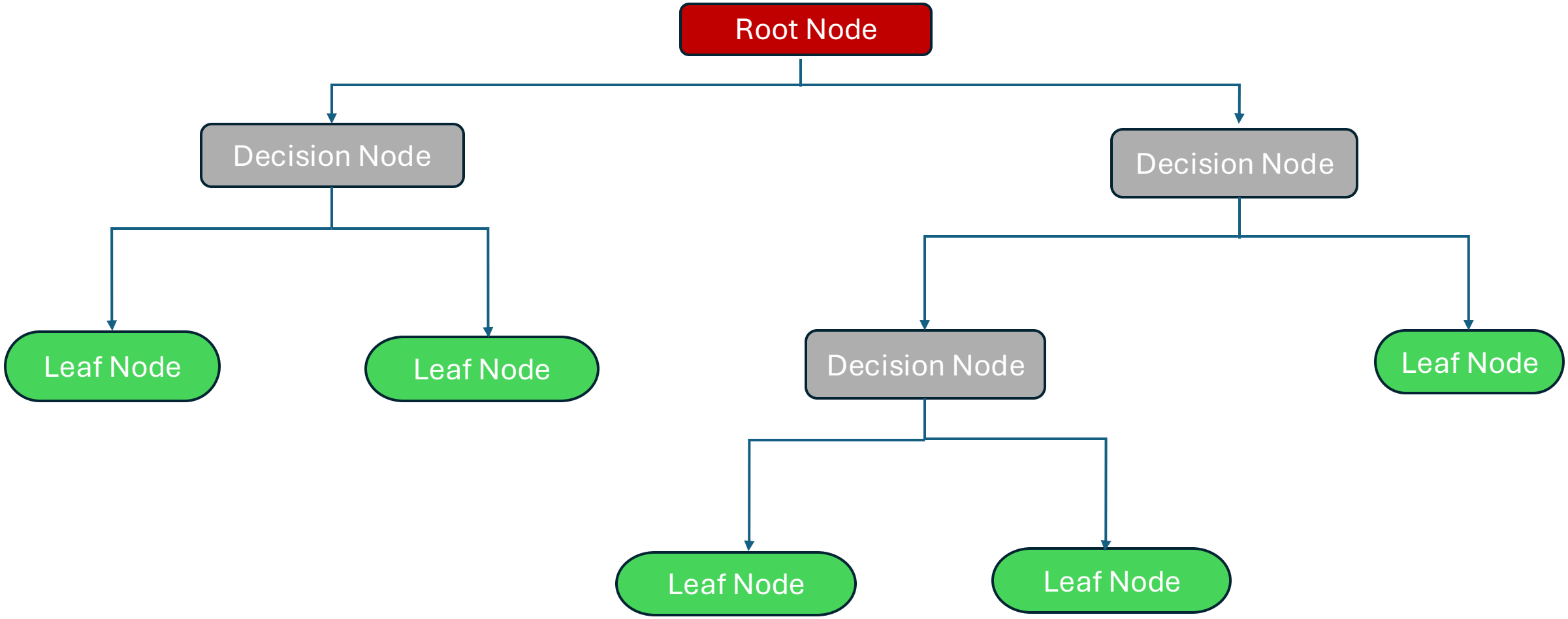
Bharat Kumar

Decision Tree

- A **Decision Tree** is a supervised machine learning model
- used for classification and regression tasks.
- It splits a dataset into subsets based on the value of input features.

Decision Tree

A **Decision Tree** is a supervised machine learning model used for classification and regression tasks. It splits a dataset into subsets based on the value of input features.



The topmost node that represents the entire dataset and initiates the first split.

An internal node that splits the data based on a feature condition.

The final node that gives the output label and does not split further.

Branch

A decision rule leading from one node to another.

Has Feathers ?

True

False

Can Fly ?

Has Fins ?

True

False

True

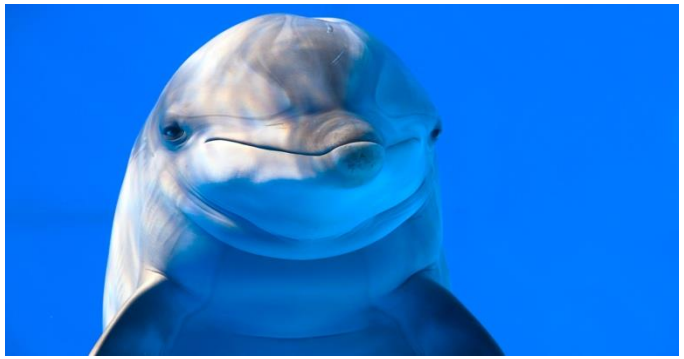
False

Hawk

Penguin

Dolphin

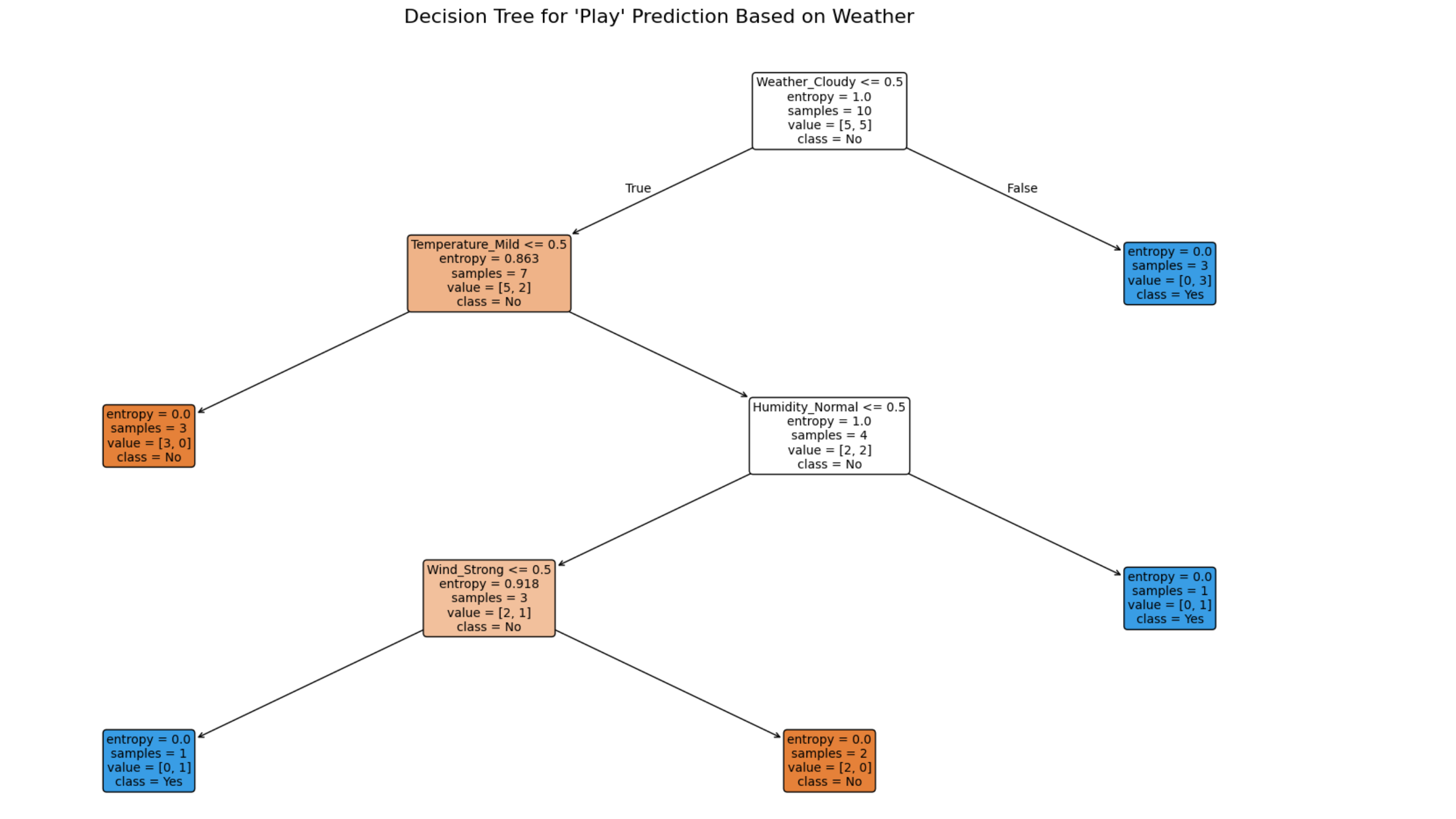
Bear



Animal	Has_Feathers	Can_Fly	Has_Fins	Class (Label)
Hawk	Yes	Yes	No	Hawk
Penguin	Yes	No	No	Penguin
Dolphin	No	No	Yes	Dolphin
Bear	No	No	No	Bear

Is the Child going to play ?

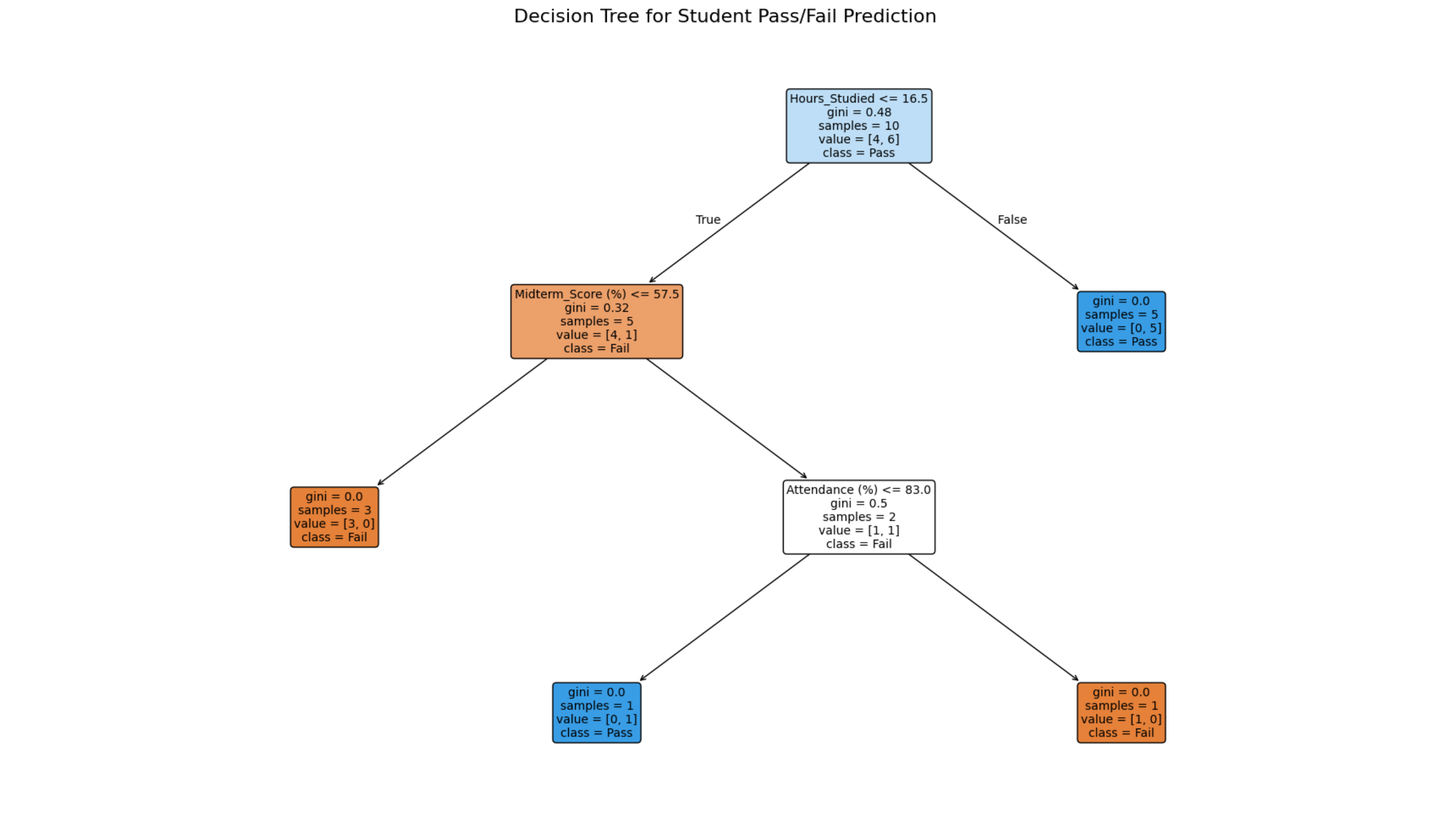
Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak[1]	Yes[2]
3	Sunny[4]	Mild[5]	Normal[6]	Strong[7]	Yes[8]
4	Cloudy[10]	Mild[11]	High[12]	Strong[13]	Yes[14]
5	Rainy[16]	Mild[17]	High[18]	Strong[19]	No[20]
6	Rainy[22]	Cool[23]	Normal[24]	Strong[25]	No[26]
7	Rainy[28]	Mild[29]	High[30]	Weak[31]	Yes[32]
8	Sunny[34]	Hot[35]	High[36]	Strong[37]	No[38]
9	Cloudy[40]	Hot[41]	Normal[42]	Weak[43]	Yes[44]
10	Rainy[46]	Mild[47]	High[48]	Strong[49]	N[50]o


Decision Tree for 'Play' Prediction Based on Weather
Weather_Cloudy <= 0.5
entropy = 1.0
samples = 10
value = [5, 5]
class = No
True
False
Temperature_Mild <= 0.5
entropy = 0.863
samples = 7
value = [5, 2]
class = No
entropy = 0.0
samples = 3
value = [0, 3]
class = Yes
entropy = 0.0
samples = 3
value = [3, 0]
class = No
Humidity_Normal <= 0.5
entropy = 1.0
samples = 4
value = [2, 2]
class = No
Wind_Strong <= 0.5
entropy = 0.918
samples = 3
value = [2, 1]
class = No
entropy = 0.0
samples = 1
value = [0, 1]
class = Yes
entropy = 0.0
samples = 1
value = [0, 1]
class = Yes
entropy = 0.0
samples = 2
value = [2, 0]
class = No

What is a Decision Tree?

- A supervised machine learning model used for making decisions.
- Works for both
- **classification** (e.g., Yes/No) and **regression** (e.g., price) tasks.
- It splits a large dataset into smaller, more manageable subsets based on its features

Hours_Studied	Attendance (%)	Midterm_Score (%)	Final_Exam_Status
25	95	85	Pass
10	60	45	Fail
18	80	55	Pass
5	90	40	Fail
30	98	92	Pass
12	75	65	Pass
15	50	50	Fail
22	85	78	Pass
19	70	51	Pass
9	91	75	Fail


Decision Tree for Student Pass/Fail Prediction
Hours_Studied <= 16.5
gini = 0.48
samples = 10
value = [4, 6]
class = Pass
True
False
Midterm_Score (%) <= 57.5
gini = 0.32
samples = 5
value = [4, 1]
class = Fail
gini = 0.0
samples = 5
value = [0, 5]
class = Pass
gini = 0.0
samples = 3
value = [3, 0]
class = Fail
Attendance (%) <= 83.0
gini = 0.5
samples = 2
value = [1, 1]
class = Fail
gini = 0.0
samples = 1
value = [0, 1]
class = Pass
gini = 0.0
samples = 1
value = [1, 0]
class = Fail

```python
# Import necessary libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

# 1. Create a more complex dataset
# We've added two more students to make the relationships less simple.
# Now, Midterm_Score alone isn't a perfect predictor.
data = {
    'Hours_Studied': [25, 10, 18, 5, 30, 12, 15, 22, 19, 9],
    'Attendance (%)': [95, 60, 80, 90, 98, 75, 50, 85, 70, 91],
    'Midterm_Score (%)': [85, 45, 55, 40, 92, 65, 50, 78, 51, 75],
    'Final_Exam_Status': ['Pass', 'Fail', 'Pass', 'Fail', 'Pass', 'Pass', 'Fail', 'Pass', 'Pass', 'Fail']
}
df = pd.DataFrame(data)

# 2. Prepare the data for the model
# X contains the features (the factors we use to predict).
# y contains the target (what we want to predict).
features = ['Hours_Studied', 'Attendance (%)', 'Midterm_Score (%)']
X = df[features]
y = df['Final_Exam_Status']

# 3. Create and train the Decision Tree model
# The model will learn the rules from the data.
# `random_state` is set for reproducibility, so the tree looks the same every time.
model = DecisionTreeClassifier(random_state=42)
model.fit(X, y)
```

```
# 4. Visualize the Decision Tree
# This plot shows the more complex rules the model has now learned.
print("Decision Tree Rules:")
plt.figure(figsize=(20, 12))
plot_tree(model,
          feature_names=features,
          class_names=model.classes_, # Shows 'Pass' and 'Fail'
          filled=True, # Colors the nodes for clarity
          rounded=True, # Makes the nodes have rounded corners
          fontsize=10)

plt.title("Decision Tree for Student Pass/Fail Prediction ", fontsize=16)
plt.show()

# 5. Make a prediction for a new student (Example)
# Let's test a student with mixed results.
new_student_data = [[10, 70, 68]] # [Hours_Studied, Attendance, Midterm_Score]
prediction = model.predict(new_student_data)

print("\n--- New Student Prediction ---")
print(f"Data: Hours Studied=10, Attendance=70%, Midterm Score=68%")
print(f"Predicted Exam Status: {prediction[0]}")
```