Default title text

```
# @title Default title text
# 1. Imports
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

# 2. A tiny dataset implementing logical OR:
#    f1, f2 → f1 OR f2
X = [
    [0, 0],  # False OR False → False
    [0, 1],  # False OR  True → True
    [1, 0],  #  True OR False → True
    [1, 1],  #  True OR  True → True
]
y = [0, 1, 1, 1]  # 0=False, 1=True

# 3. Train the tree
clf = DecisionTreeClassifier(random_state=0)
clf.fit(X, y)

# 4. Make some predictions
print("Predict [0,0]:", clf.predict([[0, 0]])[0])  # → 0
print("Predict [1,0]:", clf.predict([[1, 0]])[0])  # → 1
print("Predict [0,1]:", clf.predict([[0, 1]])[0])  # → 1
print("Predict [1,1]:", clf.predict([[1, 1]])[0])  # → 1

# 5. Visualize the learned tree
plt.figure(figsize=(4,3))
plot_tree(
    clf,
    feature_names=["f1", "f2"],
    class_names=["False", "True"],
    filled=True,
    rounded=True
)
plt.title("Decision Tree for OR Gate")
plt.show()
```

```
Predict [0,0]: 0
Predict [1,0]: 1
Predict [0,1]: 1
Predict [1,1]: 1
```


```
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

# 1. Our tiny dataset: [temperature] → label (0=cold, 1=hot)
X = [
    [10],  # cold
    [15],  # cold
    [20],  # hot
    [25],  # hot
    [30],  # hot
]
y = [0, 0, 1, 1, 1]

# 2. Train the Decision Tree
clf = DecisionTreeClassifier(random_state=0)
clf.fit(X, y)

# 3. Make predictions
print("Predict 18°C →", "hot" if clf.predict([[18]])[0] else "cold")
print("Predict  5°C →", "hot" if clf.predict([[5]])[0]  else "cold")
print("Predict 28°C →", "hot" if clf.predict([[28]])[0] else "cold")
```

```
# 4. Visualize the tree
plt.figure(figsize=(4, 3))
plot_tree(
    clf,
    feature_names=["temperature"],
    class_names=["cold", "hot"],
    filled=True,
    rounded=True
)
plt.title("Tree: cold vs. hot")
plt.show()
```

```
Predict 18°C → hot
Predict  5°C → cold
Predict 28°C → hot
```