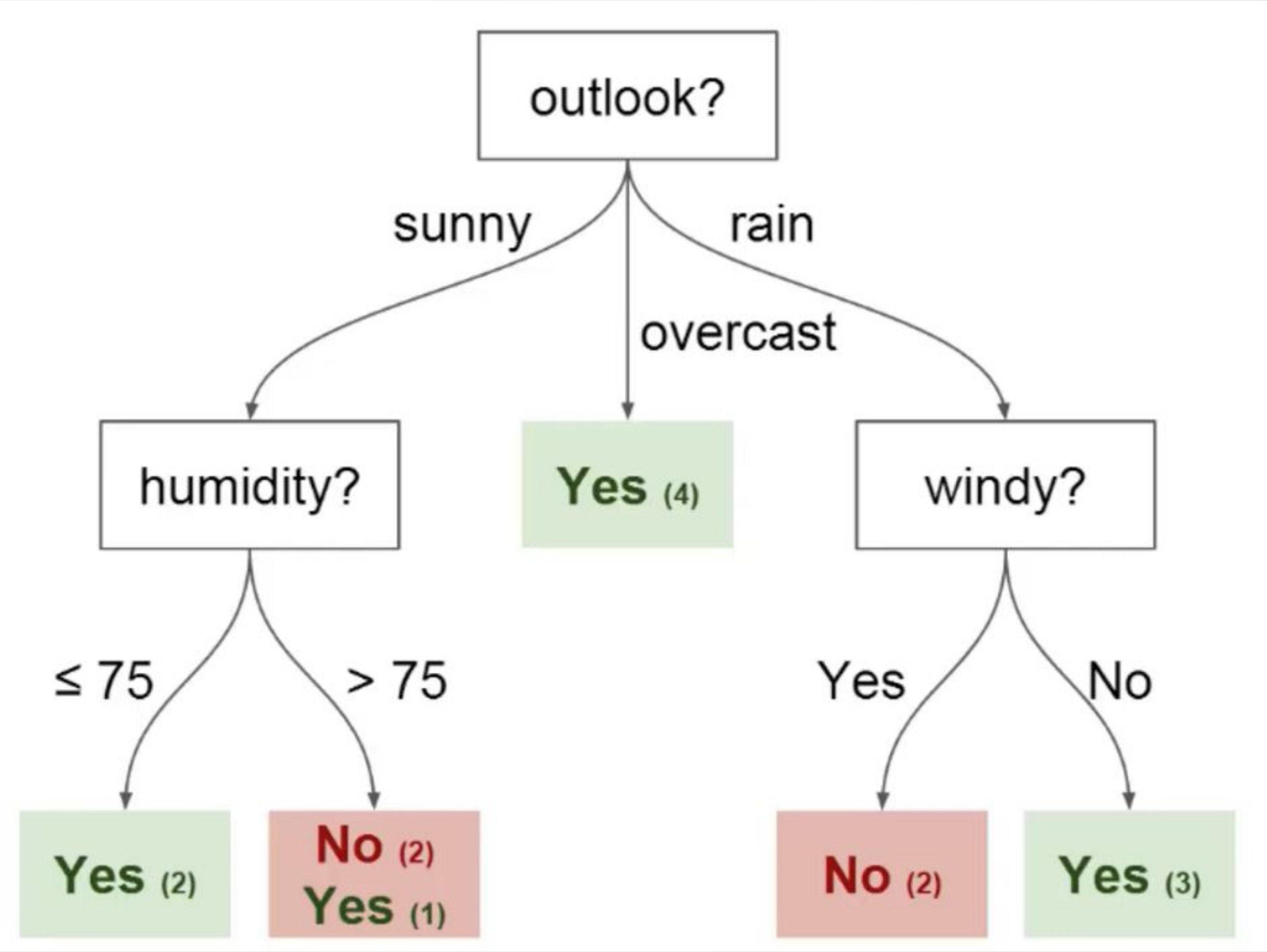
Decision Tree

Decision Tree

- A Decision Tree is a Supervised Machine Learning algorithm
- The decision tree is like a tree with nodes
- It splits data into branches like these till it achieves a threshold value
- A decision tree consists of the root nodes, children nodes, and leaf nodes.



*Imagine that you play Cricket every Sunday and you always invite your friend to come to play with you. Sometimes your friend actually comes and sometimes he doesn’t.

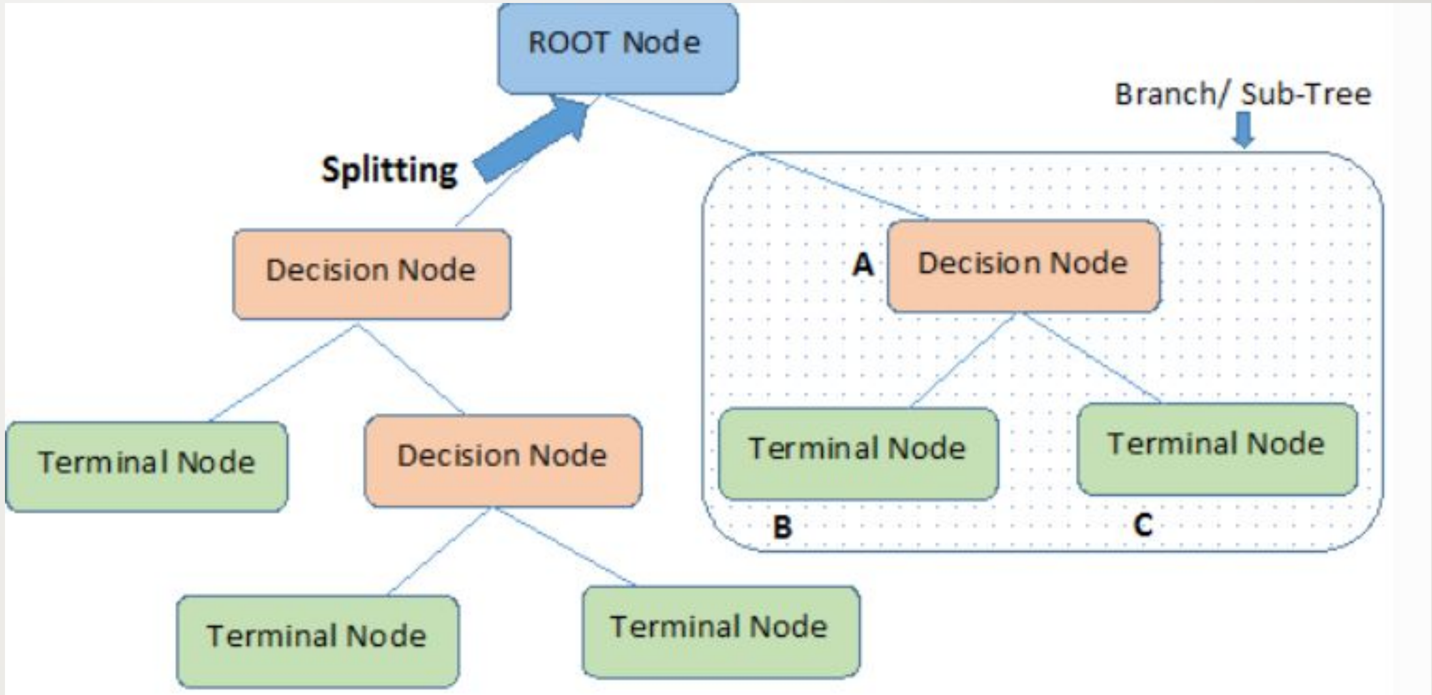
Decision Tree Terminologies

Root Node: It represents the entire population or sample. This further gets divided into two or more homogeneous sets.

Decision Node: When a sub-node splits into further sub-nodes, then it is called a decision node.

Leaf/Terminal Node: Nodes that do not split are called Leaf or Terminal nodes.

Parent and Child Node: A node, which is divided into sub-nodes is called the parent node of sub-nodes where sub-nodes are the children of a parent node



How to Build Decision Trees from Scratch

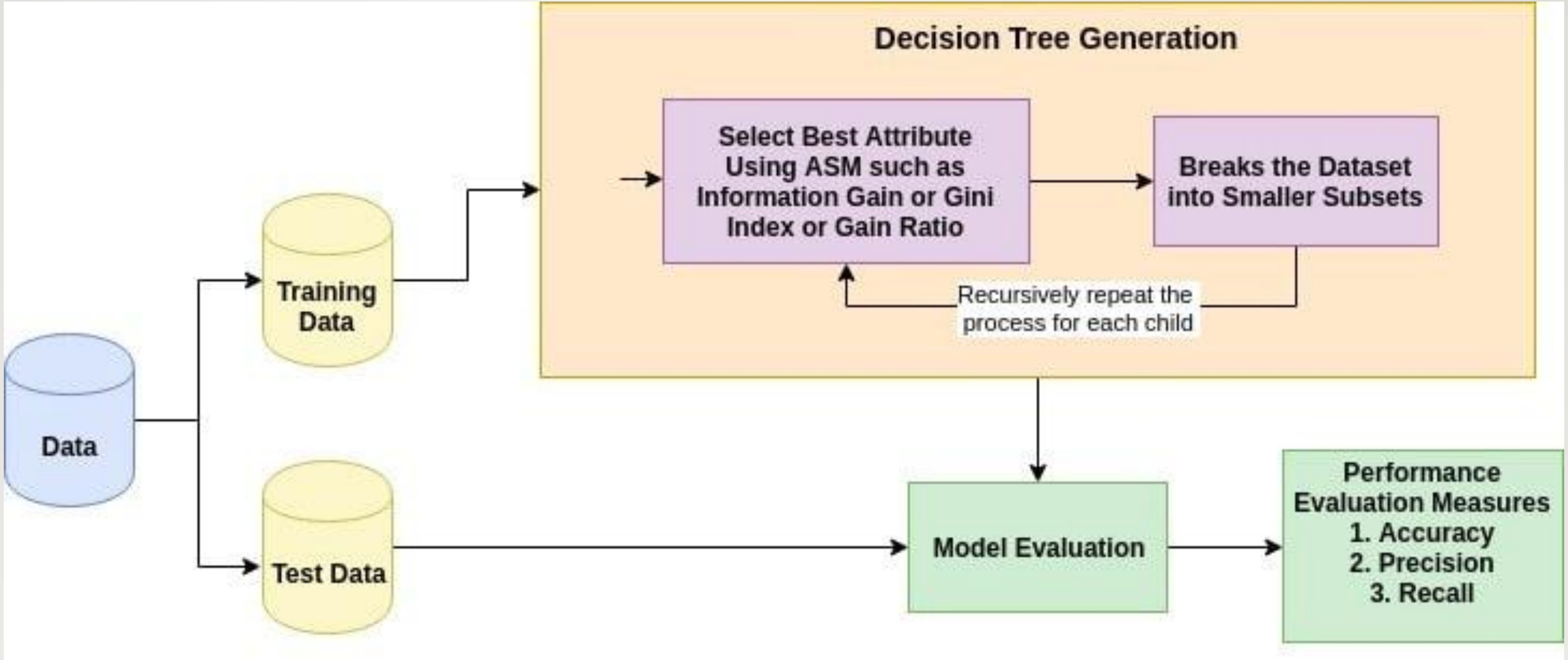
The most important thing to remember when creating a Decision tree is to choose the best attribute from the dataset's entire features list for the root node and sub-nodes.

The selection of best attributes is being achieved with the help of a technique known as the Attribute selection measure (ASM).

There are two techniques for ASM:

- **Information Gain : (Entropy)** maximum information gain attribute should be selected as compared to lower information gain valued attribute
- **Gini Index : (measure of impurity and purity) :** lower the Gini value higher is attribute importance

Decision Tree Algorithm Working

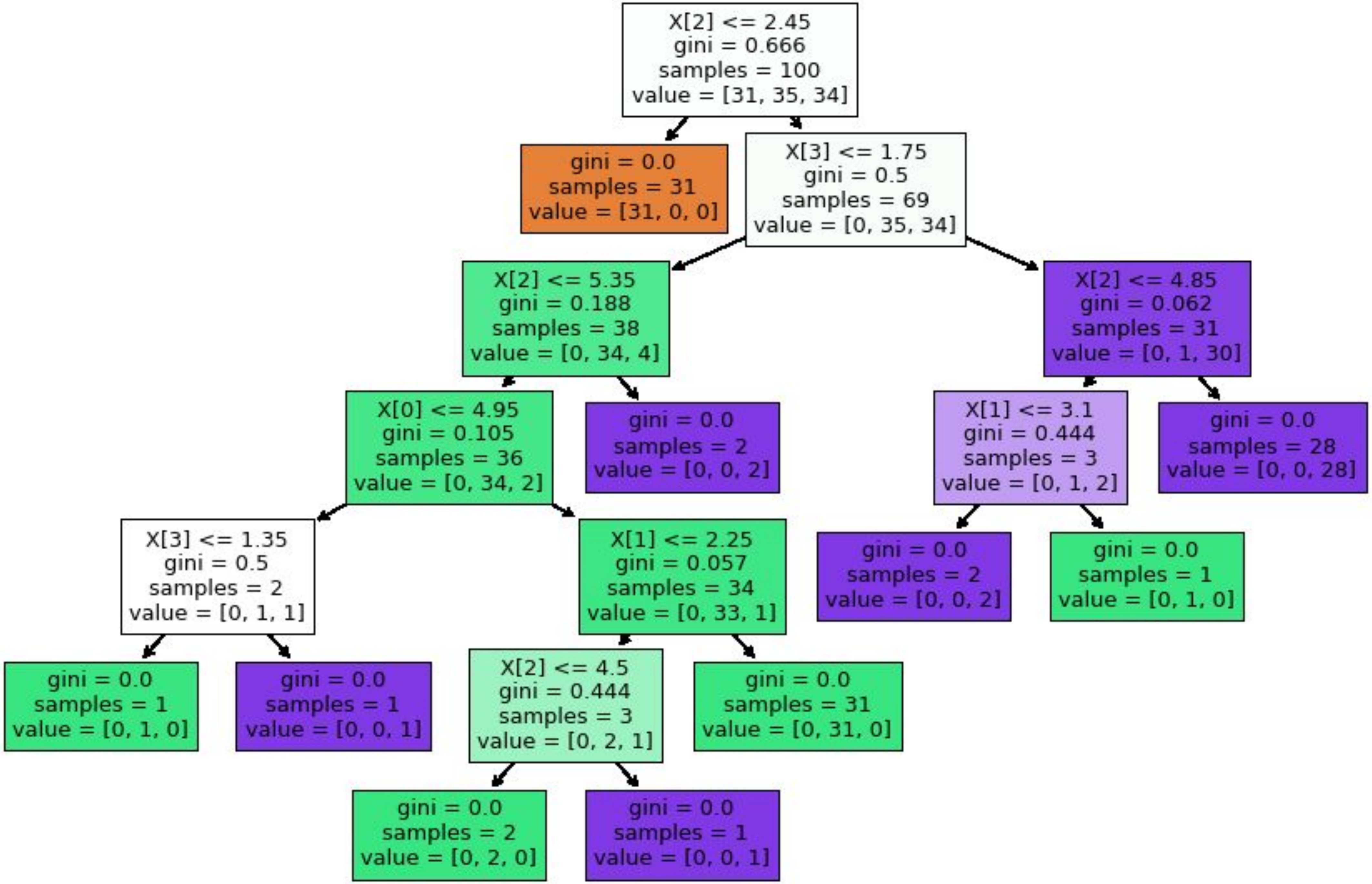

Decision Tree Generation
Select Best Attribute Using ASM such as Information Gain or Gini Index or Gain Ratio
Breaks the Dataset into Smaller Subsets
Recursively repeat the process for each child
Training Data
Data
Test Data
Model Evaluation
Performance Evaluation Measures
1. Accuracy
2. Precision
3. Recall

Advantages of Decision Tree Algorithm

- It is simple to implement and it follows a flow chart type structure that resembles human-like decision making.
- It proves to be very useful for decision-related problems.
- It helps to find all of the possible outcomes for a given problem.
- There is very little need for data cleaning in decision trees compared to other Machine Learning algorithms.

Disadvantages

- Too many layers of decision tree make it extremely complex sometimes.
- It may result in overfitting (which can be resolved using the **Random Forest algorithm)**
- For the more number of the class labels, the computational complexity of the decision tree increases.


X[2] <= 2.45
gini = 0.666
samples = 100
value = [31, 35, 34]
gini = 0.0
samples = 31
value = [31, 0, 0]
X[3] <= 1.75
gini = 0.5
samples = 69
value = [0, 35, 34]
X[2] <= 5.35
gini = 0.188
samples = 38
value = [0, 34, 4]
X[2] <= 4.85
gini = 0.062
samples = 31
value = [0, 1, 30]
X[0] <= 4.95
gini = 0.105
samples = 36
value = [0, 34, 2]
gini = 0.0
samples = 2
value = [0, 0, 2]
X[1] <= 3.1
gini = 0.444
samples = 3
value = [0, 1, 2]
gini = 0.0
samples = 28
value = [0, 0, 28]
X[3] <= 1.35
gini = 0.5
samples = 2
value = [0, 1, 1]
X[1] <= 2.25
gini = 0.057
samples = 34
value = [0, 33, 1]
gini = 0.0
samples = 2
value = [0, 0, 2]
gini = 0.0
samples = 1
value = [0, 1, 0]
gini = 0.0
samples = 1
value = [0, 1, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]
X[2] <= 4.5
gini = 0.444
samples = 3
value = [0, 2, 1]
gini = 0.0
samples = 31
value = [0, 31, 0]
gini = 0.0
samples = 2
value = [0, 2, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]