internship studio

Welcome to

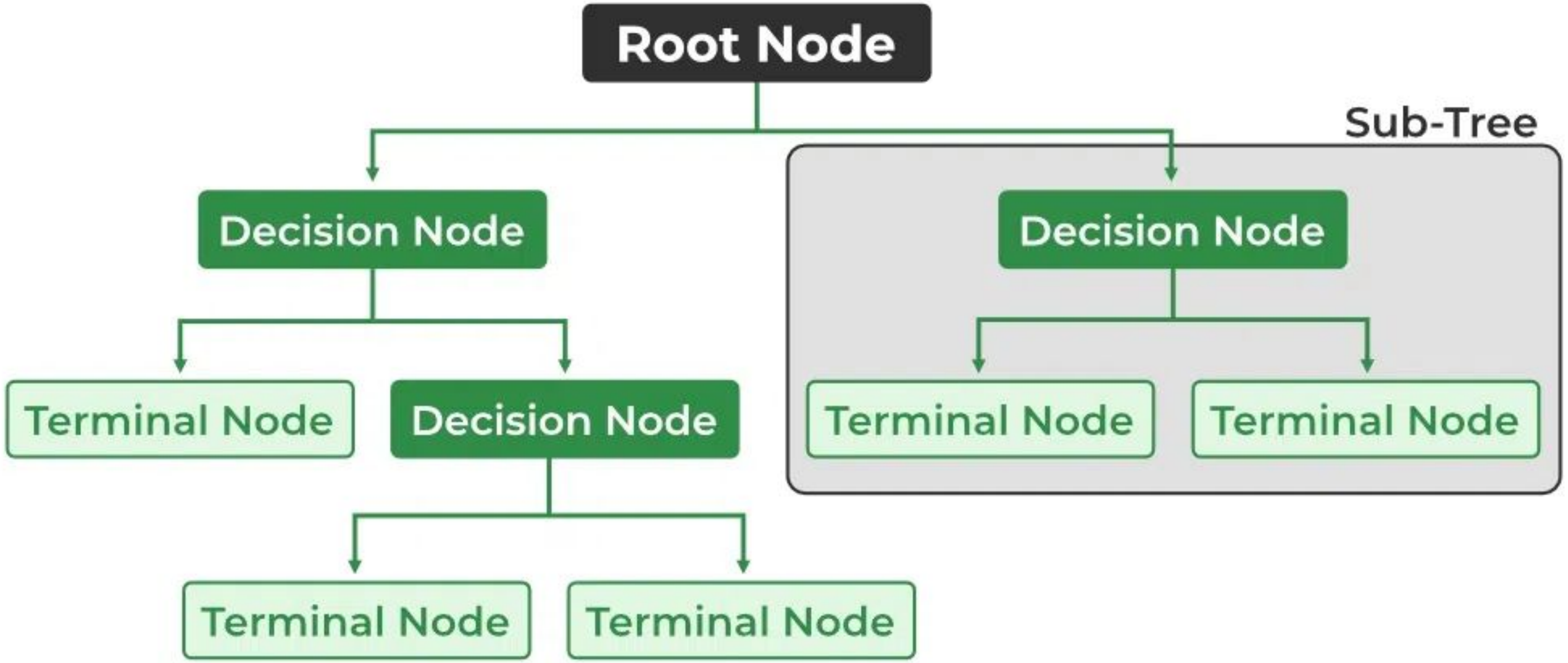
INTERNSHIP STUDIO

Module 04 | Lesson 04

Decision Tree

Decision Tree

- A decision tree is a flowchart-like tree structure where each internal node denotes the feature, branches denote the rules and the leaf nodes denote the result of the algorithm.
- It is supervised machine-learning algorithm.
- It is used for both classification and regression problems.



Decision Tree Terminologies

Some of the common Terminologies used in Decision Trees are as follows:

1. Root Node: It is the topmost node in the tree, which represents the complete dataset. It is the starting point of the decision-making process.

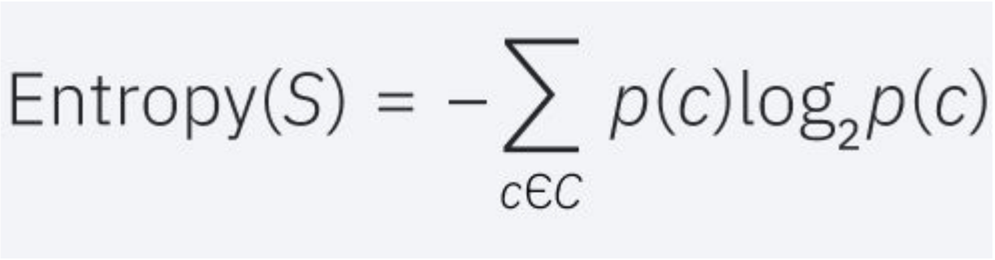
Decision/Internal Node: A node that symbolizes a choice regarding an input feature. Branching off internal nodes connects them to leaf nodes or other internal nodes.

2. Leaf/Terminal Node: A node without any child nodes that indicates a class label or a numerical value.
3. Splitting: The process of splitting a node into two or more sub-nodes using a split criterion and a selected feature.
4. Branch/Sub-Tree: A subsection of the decision tree starts at an internal node and ends at the leaf nodes.
5. Parent Node: The node that divides into one or more child nodes.
6. Child Node: The nodes that emerge when a parent node is split.
7. Impurity: A measurement of the target variable’s homogeneity in a subset of data. It refers to the degree of randomness or uncertainty in a set of examples. The Gini index and entropy are two commonly used impurity measurements in decision trees for classifications task
8. Variance: Variance measures how much the predicted and the target variables vary in different samples of a dataset. It is used for regression problems in decision trees. Mean squared error, Mean Absolute Error, friedman_mse, or Half Poisson deviance are used to measure the variance for the regression tasks in the decision tree.
9. Information Gain: Information gain is a measure of the reduction in impurity achieved by splitting a dataset on a particular feature in a decision tree. The splitting criterion is determined by the feature that offers the greatest information gain, It is used to determine the most informative feature to split on at each node of the tree, with the goal of creating pure subsets
10. Pruning: The process of removing branches from the tree that do not provide any additional information or lead to overfitting.

How to choose the best attribute at each node

There are two methods, information gain and Gini impurity, act as popular splitting criterion for decision tree models. They help to evaluate the quality of each test condition and how well it will be able to classify samples into a class.

Entropy and Information Gain



$$\text{Entropy}(S) = -\sum_{c \in C} p(c)\log_2 p(c)$$

- S represents the data set that entropy is calculated
- c represents the classes in set, S
- p(c) represents the proportion of data points that belong to class c to the number of total data points in set, S

How to choose the best attribute at each node

$$Information\ Gain\ (S,a) = Entropy(S) - \sum_{v \in vclaues(a)} \frac{|S_v|}{|S|}\ Entropy(S_v)$$

- *a* represents a specific attribute or class label
- *Entropy(S)* is the entropy of dataset, S
- |Sv|/ |S| represents the proportion of the values in S_v to the number of values in dataset, S
- *Entropy(S_v)* is the entropy of dataset, S_v

Gini Impurity Index: Gini Impurity is a score that evaluates how accurate a split is among the classified groups. The Gini Impurity evaluates a score in the range between 0 and 1, where 0 is when all observations belong to one class, and 1 is a random distribution of the elements within classes. In this case, we want to have a Gini index score as low as possible. Gini Index is the evaluation metric we shall use to evaluate our Decision Tree Model.

$$Gini\ impurity = 1 - \sum P_i^2$$

pi is the proportion of elements in the set that belongs to the ith category.

Advantages

- **Easy to interpret:** The Boolean logic and visual representations of decision trees make them easier to understand and consume. The hierarchical nature of a decision tree also makes it easy to see which attributes are most important, which isn’t always clear with other algorithms, like neural networks.

- **Little to no data preparation required:** Decision trees have a number of characteristics, which make it more flexible than other classifiers. It can handle various data types—i.e. discrete or continuous values, and continuous values can be converted into categorical values through the use of thresholds. Additionally, it can also handle values with missing values, which can be problematic for other classifiers, like Naïve Bayes.

- **More flexible:** Decision trees can be leveraged for both classification and regression tasks, making it more flexible than some other algorithms. It’s also insensitive to underlying relationships between attributes; this means that if two variables are highly correlated, the algorithm will only choose one of the features to split on.

Disadvantages

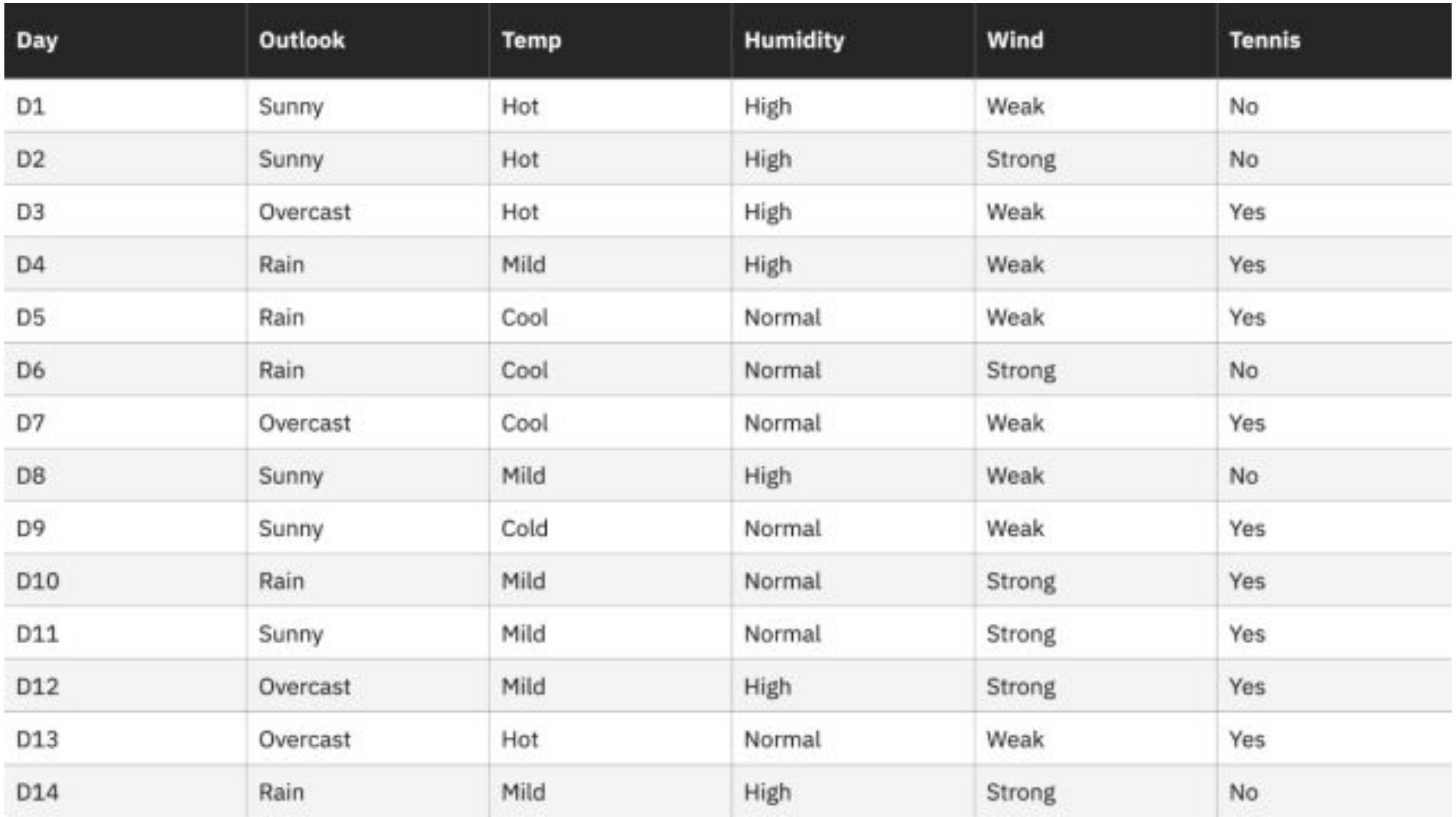
- **Prone to overfitting:** Complex decision trees tend to overfit and do not generalize well to new data. This scenario can be avoided through the processes of pre-pruning or post-pruning. Pre-pruning halts tree growth when there is insufficient data while post-pruning removes subtrees with inadequate data after tree construction.

- **High variance estimators:** Small variations within data can produce a very different decision tree. Bagging, or the averaging of estimates, can be a method of reducing variance of decision trees. However, this approach is limited as it can lead to highly correlated predictors.

- More costly: Given that decision trees take a greedy search approach during construction, they can be more expensive to train compared to other algorithms.

- Not fully supported in scikit-learn: Scikit-learn is a popular machine learning library based in Python. While this library does have a Decision Tree module (DecisionTreeClassifier, link resides outside of ibm.com), the current implementation does not support categorical variables.

Given Example



Day	Outlook	Temp	Humidity	Wind	Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Given Example

For this dataset, the entropy is 0.94. This can be calculated by finding the proportion of days where “Play Tennis” is “Yes”, which is 9/14, and the proportion of days where “Play Tennis” is “No”, which is 5/14. Then, these values can be plugged into the entropy formula above.

Entropy (Tennis) = -(9/14) log2(9/14) – (5/14) log2 (5/14) = 0.94

Gain (Tennis, Humidity) = (0.94)-(7/14)*(0.985) – (7/14)*(0.592) = 0.151

- 7/14 represents the proportion of values where humidity equals “high” to the total number of humidity values. In this case, the number of values where humidity equals “high” is the same as the number of values where humidity equals “normal”.

- 0.985 is the entropy when Humidity = “high”

- 0.59 is the entropy when Humidity = “normal”

Then, repeat the calculation for information gain for each attribute in the table above, and select the attribute with the highest information gain to be the first split point in the decision tree. In this case, outlook produces the highest information gain. From there, the process is repeated for each subtree.