2 - Decision Tree - overview

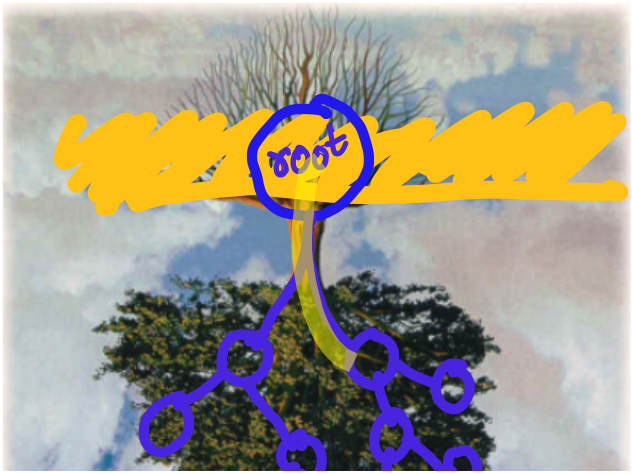
Tuesday, 17 November BE 2563 16:44

External Resources: https://towardsdatascience.com/machine-learning-basics-descision-tree-from-scratch-part-i-4251bfa1b45c

Author: Devesh Poojari

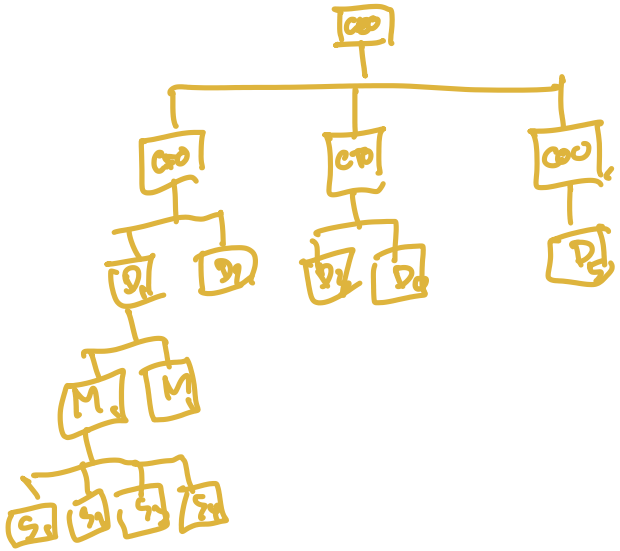
What is Decision Tree Algorithm

A decision tree is a flowchart-like tree structure where an internal node represents feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition based on the attribute value. It partitions the tree in a recursive manner called recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human-level thinking. That is why decision trees are easy to understand and interpret.

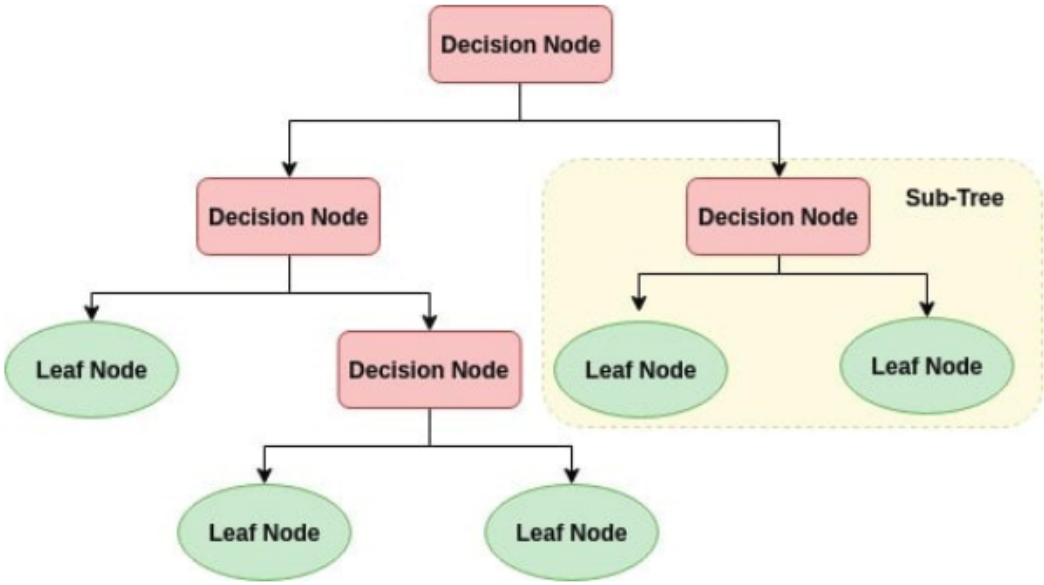


Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example, this approach is called a **Top-Down approach**. Each node in the tree acts as a test case for some attribute, and

each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive and is repeated for every



subtree rooted at the new nodes.



Decision Tree with its Terminologies

Before we dive deep, let's get familiar with some of the terminologies:

1. **Root Node (Top Decision Node):** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called a decision node.
4. **Leaf/ Terminal Node:** Nodes with no children (no further split) is called Leaf or Terminal node.

5. **Pruning:** When we reduce the size of decision trees by removing nodes (opposite of Splitting), the process is called pruning.
6. **Branch / Sub-Tree:** A sub section of the decision tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

How does the Decision Tree algorithm work?

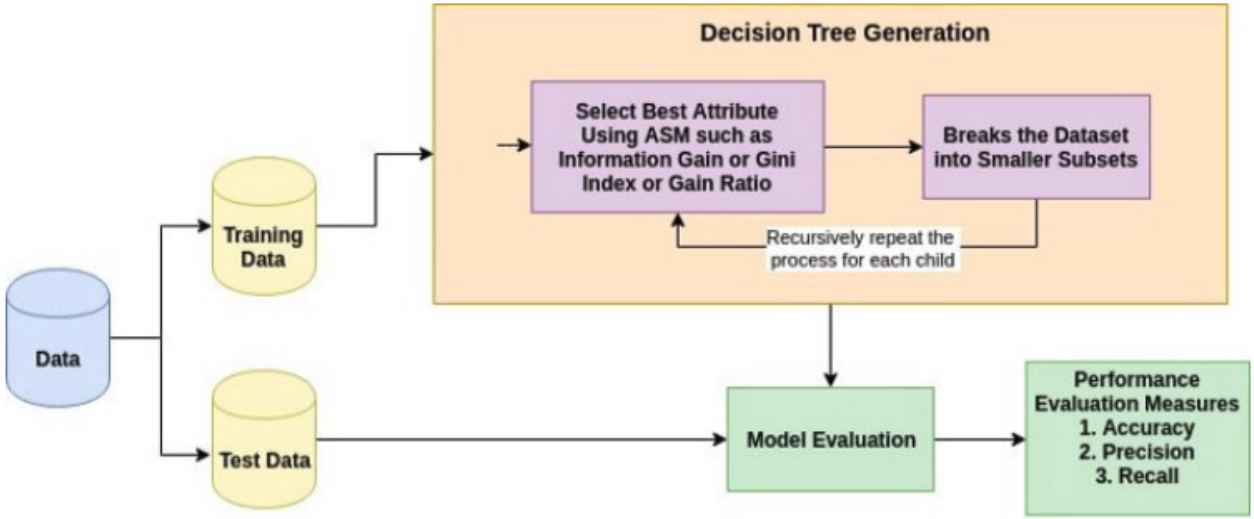
The basic idea behind any decision tree algorithm is as follows:

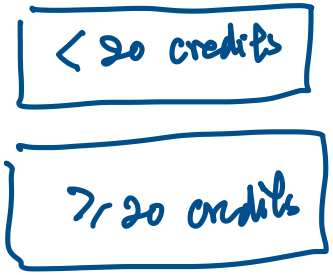
1. Select the best attribute using Attribute Selection Measures(ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the condition will match:

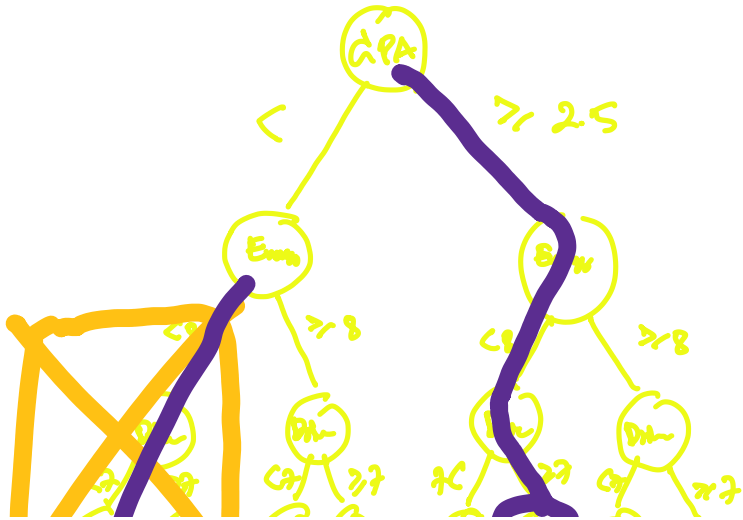
o All the tuples belong to the same attribute value.

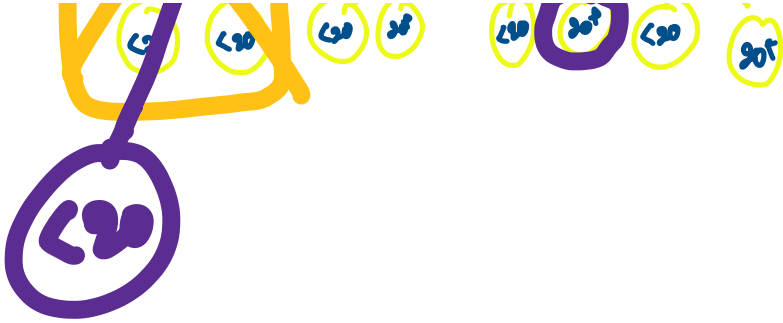
o There are no more remaining attributes.

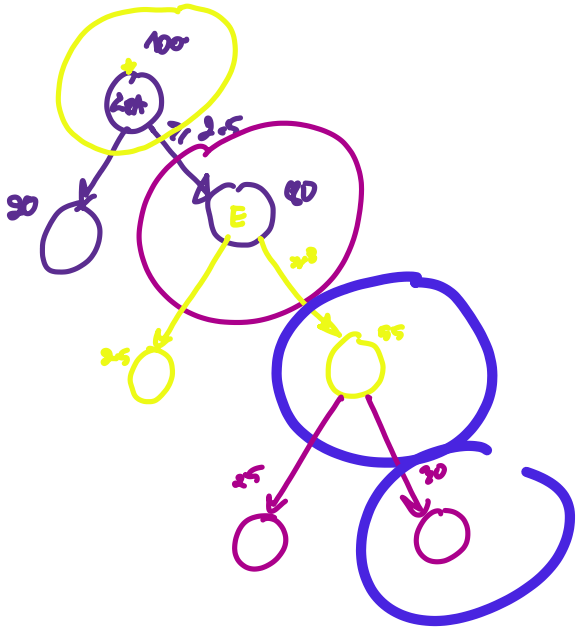
o There are no more instances.











The Math Behind Decision Trees:

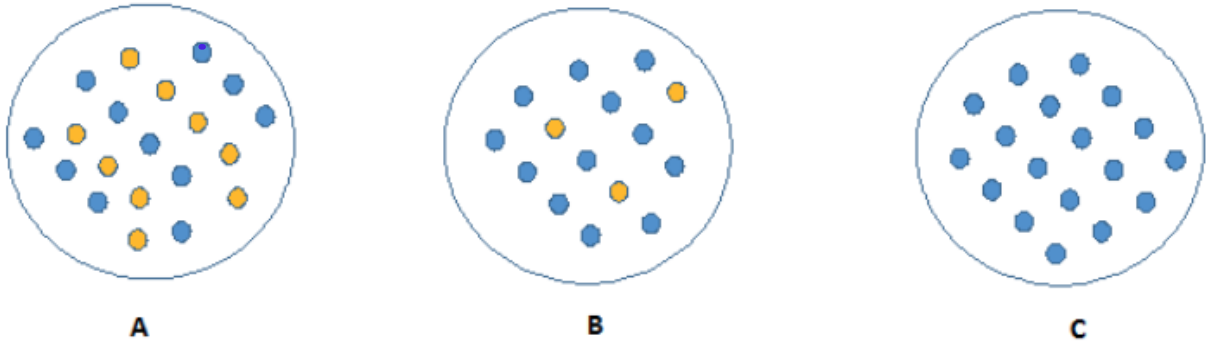
Attribute Selection Measures

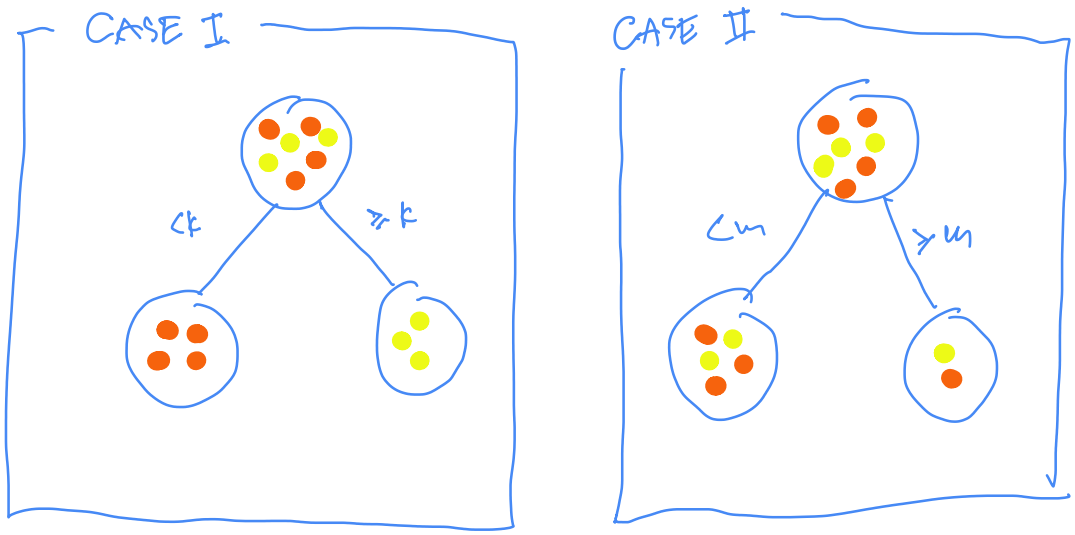
Attribute selection measure is a heuristic for selecting the splitting criterion that partition data into the best possible manner. It is also known as splitting rules because it helps us to determine breakpoints for tuples on a given node. ASM provides a rank to each feature(or attribute) by explaining the given dataset. Best score attribute will be selected as a splitting attribute (Source). In the case of a continuous-valued attribute, split points for branches also need to define. Most popular selection measures are Information Gain, Gain Ratio, and Gini Index.

1. **Information Gain:**

Information gain is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Look at the image below and think which node can be described easily. I am sure, your answer is C because it requires less information as all values are similar. On the other hand, B requires more information to describe it and A requires the maximum information. In

other words, we can say that C is a Pure node, B is less Impure and A is more impure.

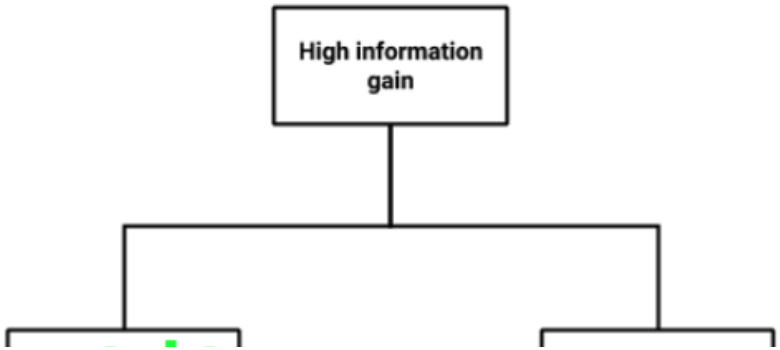


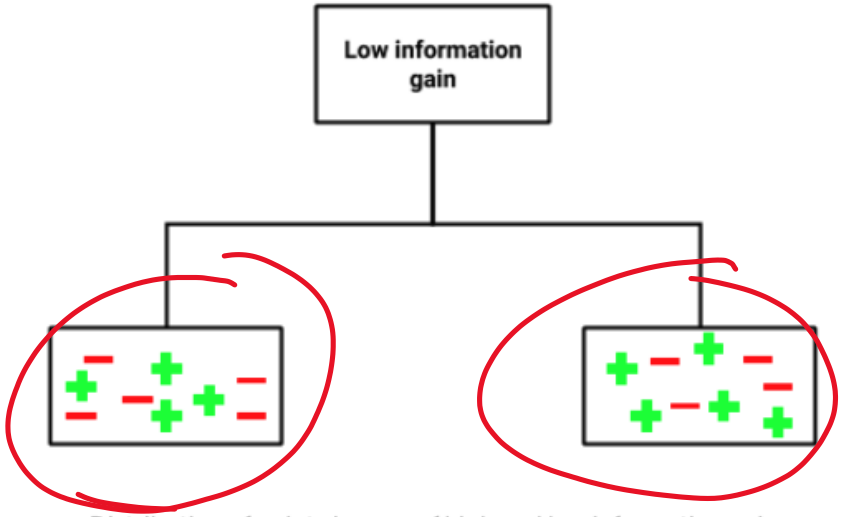


Prefer CASE I

Now, we can build a conclusion that less impure node requires less information to describe it. And, the more impure node requires more information.

In a parallel example considering Information Gain, In the figure below, we can see that an attribute with low information gain (bottom) splits the data relatively evenly and as a result doesn't bring us any closer to a decision. Whereas, an attribute with high information gain (top) splits the data into groups with an uneven number of positives and negatives and as a result, helps in separating the two from each other.





Distribution of points in case of high and low information gain

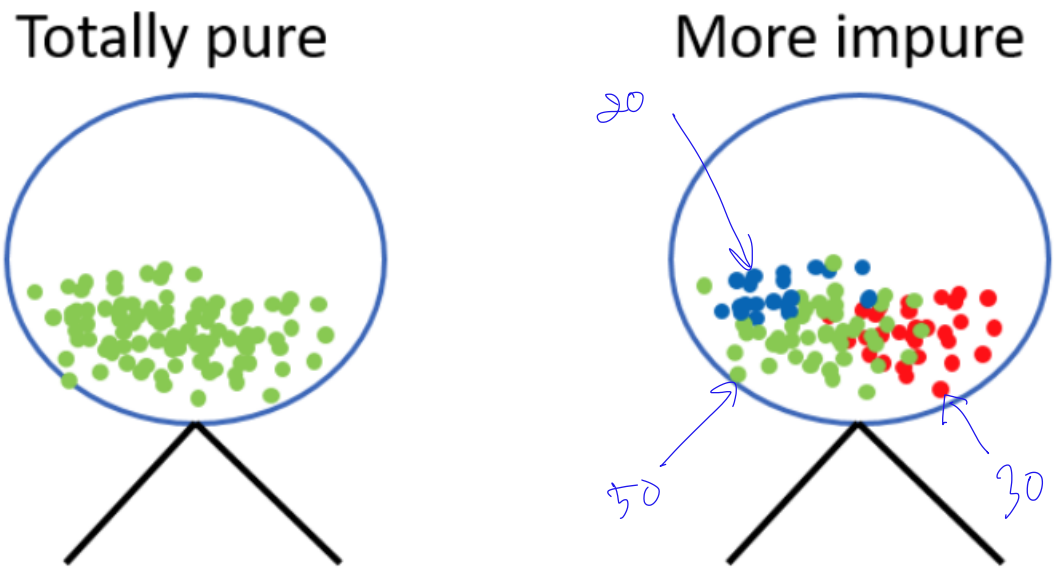
To be able to calculate the information gain, we have to first introduce the term *entropy* of a dataset.

Entropy

Shannon invented the concept of entropy, which measures the impurity of

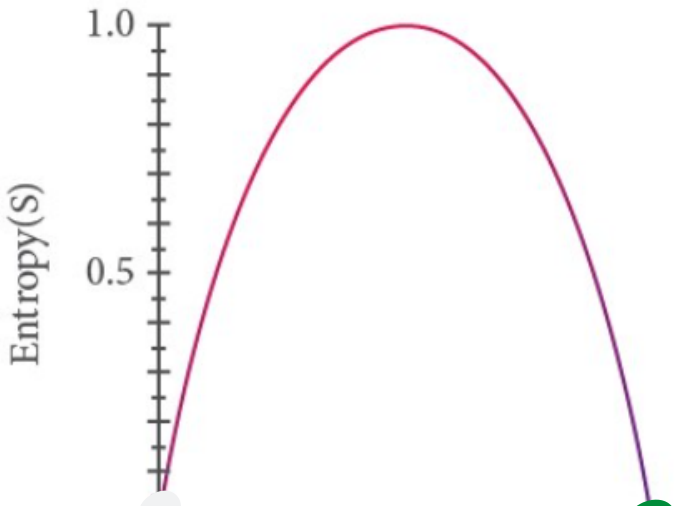
the input set. In physics and mathematics, entropy referred to as the randomness or the impurity in the system. In information theory, it refers to the impurity in a group of examples. Information gain is a decrease in entropy.

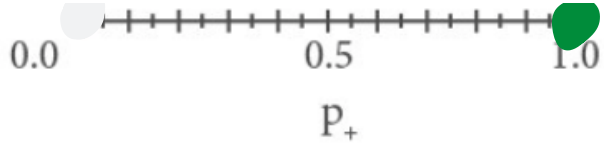
The idea behind the entropy is, in simplified terms, the following: Imagine you have a lottery wheel which includes 100 green balls. The set of balls within the lottery wheel can be said to be totally pure because only green balls are included. To express this in the terminology of entropy, this set of balls has an entropy of 0 (we can also say zero impurity). Consider now, 30 of these balls are replaced by red and 20 by blue balls.



If you now draw another ball from the lottery wheel, the probability of receiving a green ball has dropped from 1.0 to 0.5. Since the impurity increased, the purity decreased, hence also the entropy increased. Hence we can say, the more "impure" a dataset, the higher the entropy and the less "impure" a dataset, the lower the entropy

Note that entropy is 0 if all the members of S belong to the same class. For example, if all members are positive, $Entropy(S) = 0$. Entropy is 1 when the sample contains an equal number of positive and negative examples. If the sample contains an unequal number of positive and negative examples, entropy is between 0 and 1. The following figure shows the form of the entropy function relative to a boolean classification as entropy varies between 0 and 1.





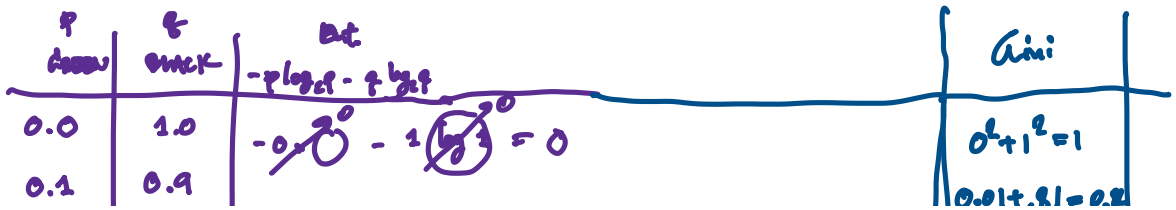
Entropy can be calculated using the formula:-

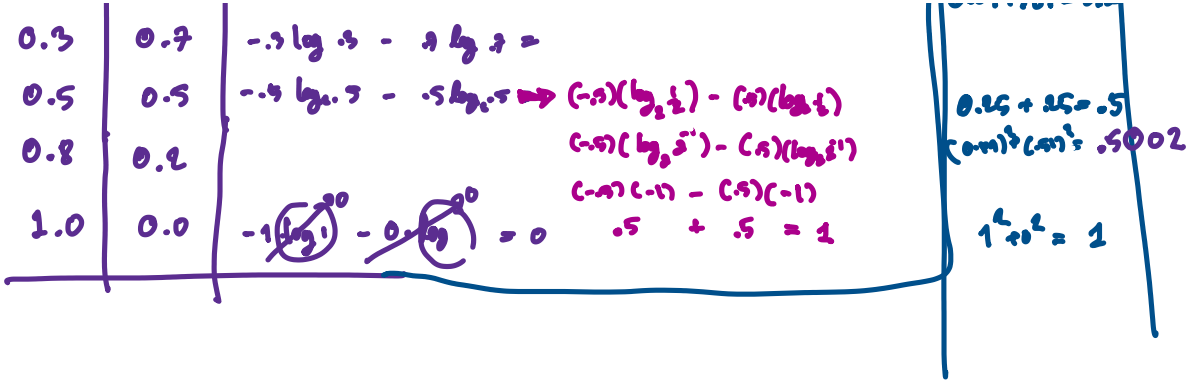
$$\text{Entropy} = -p \log_2 p - q \log_2 q$$

Here p and q is the probability of success and failure respectively in that node. Entropy is also used with the categorical target variable. It chooses the split which has the lowest entropy compared to the parent node and other splits. The lesser the entropy, the better it is.

Information gain computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values

> *Information Gain = Entropy(parent node) — [Avg Entropy(children)]*



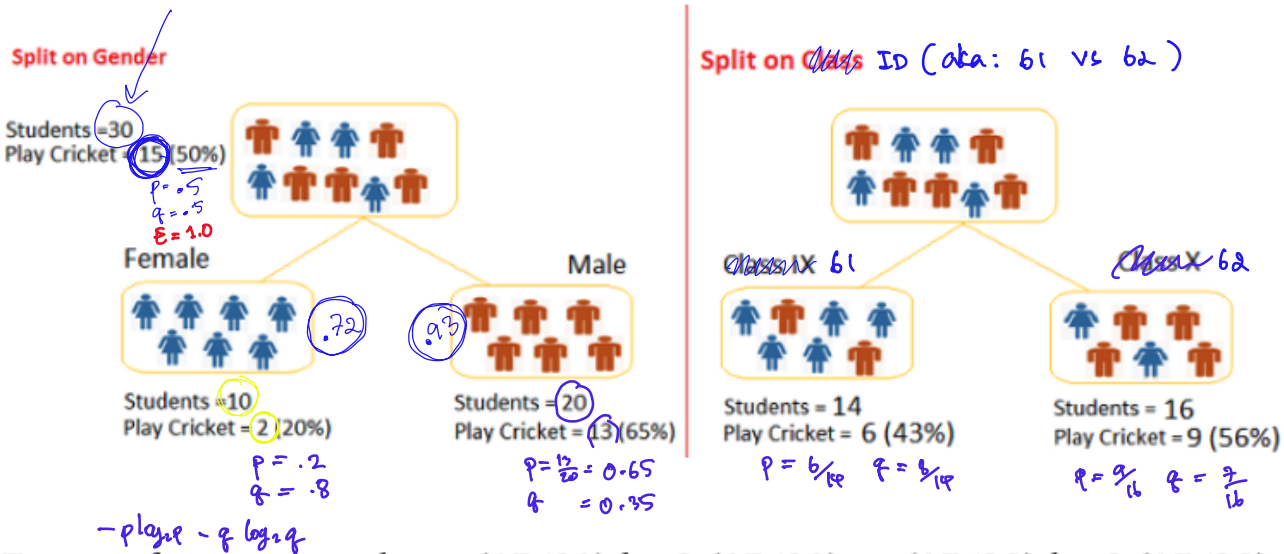


Steps to calculate entropy for a split:

1. Calculate the entropy of the parent node
2. Calculate entropy of each individual node of split and calculate the weighted average of all sub-nodes available in a split.

Example: Let's say we have a sample of 30 students with three variables Gender (Boy/ Girl), Class(IX/ X) and Height (5 to 6 ft). 15 out of these 30 play cricket in leisure time. Now, I want to create a model to predict who will play cricket during a leisure period? In this problem, we need to segregate students who play cricket in their leisure time based on highly

significant input variable among all three.



Entropy for parent node = -(15/30) log2 (15/30) — (15/30) log2 (15/30) = 1

Here 1 shows that it is a impure node.

For Split on gender:

Entropy for Female node = -(2/10) log2 (2/10) — (8/10) log2 (8/10) = 0.72 Entropy for Male node = -(13/20) log2 (13/20) — (7/20) log2 (7/20) = 0.93 Entropy for split Gender = (10/30)*0.72 + (20/30)*0.93 = **0.86**

*Information Gain for split on gender = 1–0.86 = **0.14***

For Split on Class:

Entropy for Class IX node = -(6/14) log2 (6/14) — (8/14) log2 (8/14) = 0.99 Entropy for Class X node = -(9/16) log2 (9/16) — (7/16) log2 (7/16) = 0.99 Entropy for split Class = (14/30)*0.99 + (16/30)*0.99 = **0.99**

*Information Gain for split on Class = 1– 0.99 = **0.01***

Above, you can see that Information Gain for *Split on Gender* is the Highest among all, so the tree will split on *Gender*.

2. Gini

Gini says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.

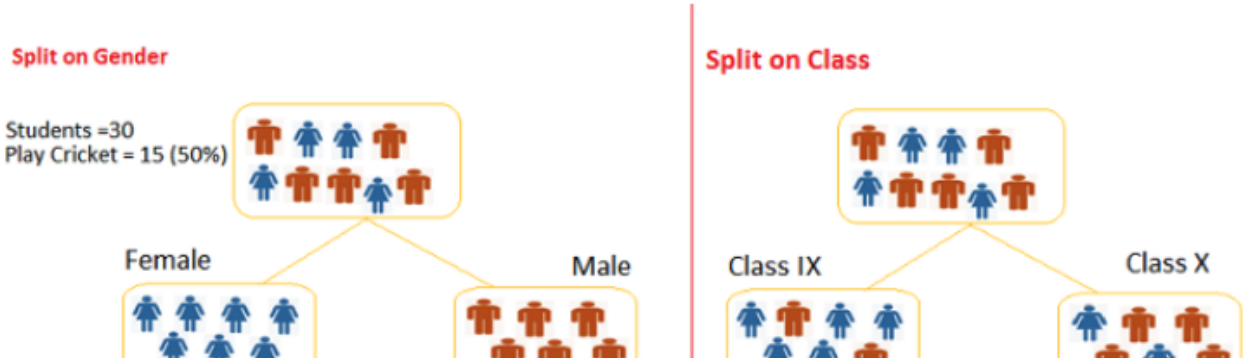
1. It works with categorical target variable “Success” or “Failure”.
2. It performs only Binary splits
3. Higher the value of Gini higher the homogeneity.

4. CART (Classification and Regression Tree) uses the Gini method to create binary splits.

Steps to Calculate Gini for a split

1. Calculate Gini for sub-nodes, using formula sum of the square of probability for success and failure (p^2+q^2).
2. Calculate Gini for split using weighted Gini score of each node of that split

Example: — Referring to the example used above, where we want to segregate the students based on the target variable (playing cricket or not). In the snapshot below, we split the population using two input variables Gender and Class. Now, I want to identify which split is producing more homogeneous sub-nodes using Gini.



Split on Gender:

1. Calculate, Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68
2. Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55
3. Calculate weighted Gini for Split Gender = (10/30)*0.68+ (20/30)*0.55 = 0.59

Similar for Split on Class:

1. Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51
2. Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51
3. Calculate weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = 0.51

Above, you can see that Gini score for *Split on Gender* is higher than *Split on Class,* hence, the node split will take place on Gender.

You might often come across the term 'Gini Impurity' which is determined by subtracting the Gini value from 1. So mathematically we can say,

Gini Impurity = 1-Gini

> *Gini is default parameter in Decision Tree Algorithm.*

There are other measures as well but mostly you'll use this two. I personally prefer doing Information Gain :)

To understand the implementation of the Decision Tree with scikit learn and how it works you can refer to this post, which will complete your information of Decision Tree Models.

. . .