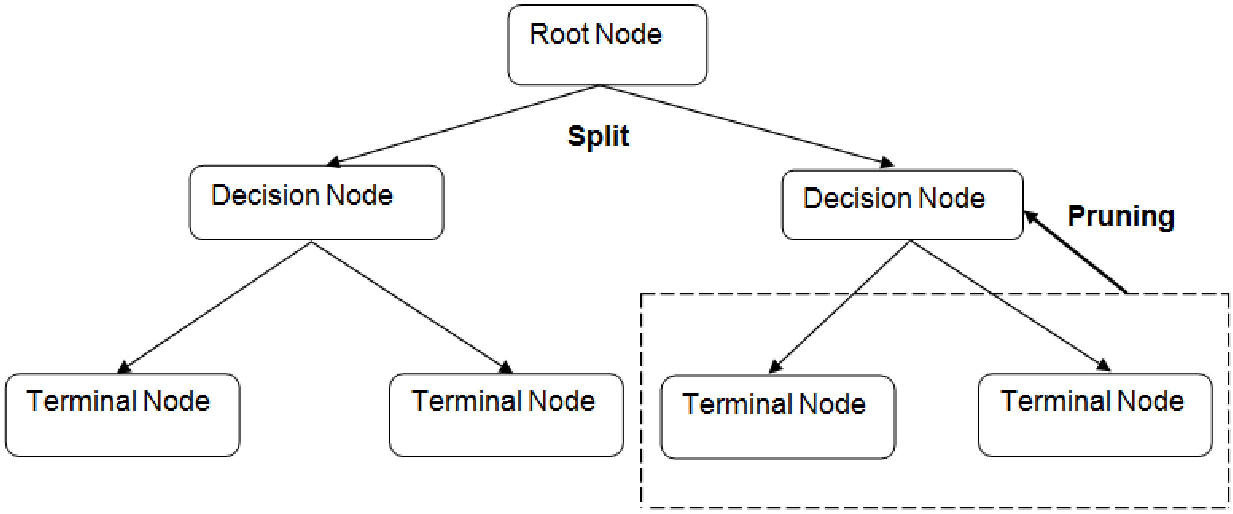
Decision Trees (DTs) are a non-parametric supervised learning method used for classification (Spam/not spam) and regression (pricing a car or a house).

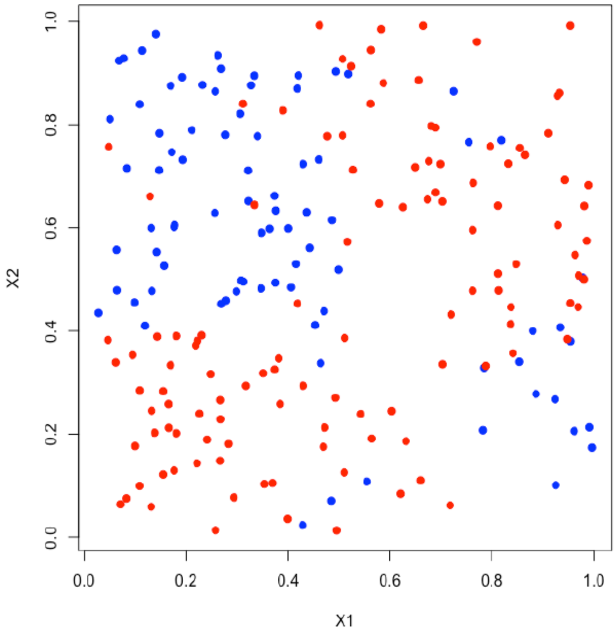
Common terminologies used in Decision trees:

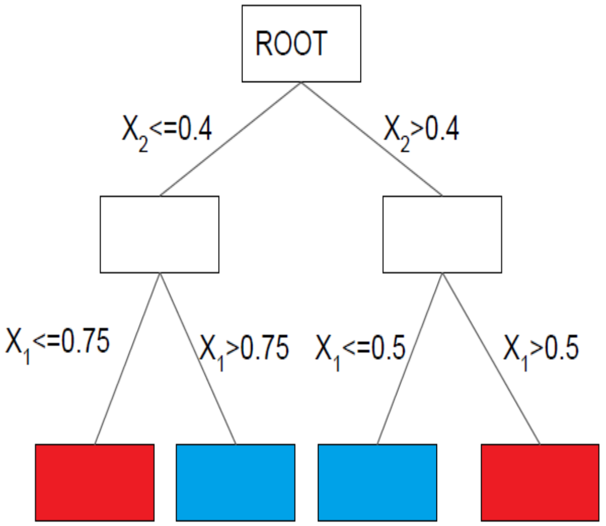
1. **Root Node**: It represents the entire population or sample which gets divided into two or more homogeneous sets (branches or sub-trees).
2. **Child and Parent Node**: A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of the parent node.
3. **Branch / Sub-Tree:** A subsection of the entire tree.
4. **Decision Node**: A sub-node that splits into further sub-nodes.
5. **Leaf / Terminal Node**: These nodes do not split further.
6. **Splitting:** It is the process of dividing a node into two or more sub-nodes. The most widely used splitting criteria are Gini-Index and Information Gain
7. **Pruning**: The process of removing the sub-nodes of a decision node. It is the opposite of the splitting process. This will help in tuning the tree to get a less biased decision.



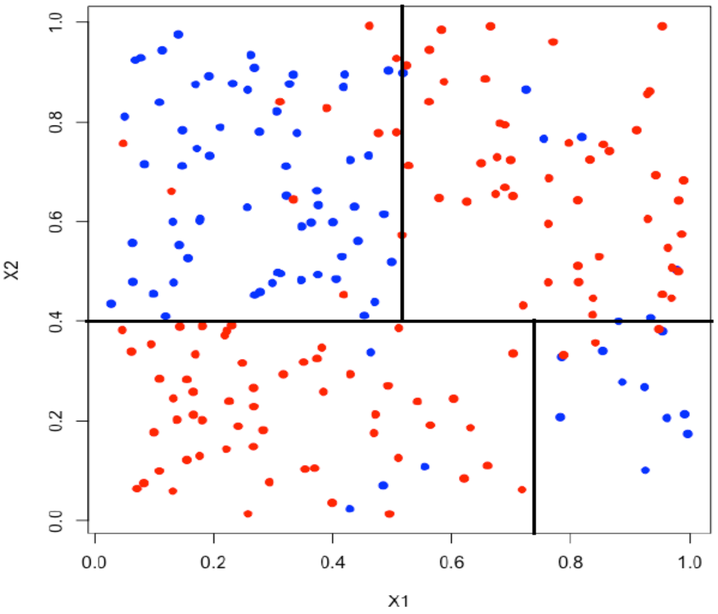
Decision trees usually work top-down, by choosing a variable at each step that **best splits** the set of items. The end nodes can have a category (classification) or a continuous number (regression).

Consider a scenario for classification. Let two continuous independent variables be x1 and x2 and the dependent variable be either Red or Blue. The decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogeneous). The data points, sample decision tree and homogeneous subsets are given in the below diagram.





The decision tree makes orthogonal splits in feature space based on rules applied to the features/variables that best split the data into homogenous subsets.



Metrics

To find the best split independent variable, metrics are used. Metrics measure how similar a region or a node is. The larger these impurity metrics, the larger the “dissimilarity” of a node/region.

Example metrics: Gini impurity, Entropy, Variance

CART (Classification And Regression Trees)

Classification Trees:

CART is one of the most popular decision tree algorithms. CART uses Gini impurity as its impurity measure. It is a measure of how often a randomly chosen element from the set would be incorrectly labeled.

Mathematically, the Gini impurity of a set can be expressed as:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

where C is the number of classes.

Consider a sample dataset which has 3 independent variables and one dependent variable (Target):

Cust_ID	Gender	Occupation	Age	Target
1	M	Sal	22	1
2	M	Sal	22	0
3	M	Self-Emp	23	1
4	M	Self-Emp	23	0
5	M	Self-Emp	24	1
6	M	Self-Emp	24	0
7	F	Sal	25	1
8	F	Sal	25	0
9	F	Sal	26	0
10	F	Self-Emp	26	0

P1(Probability of class 1) = 0.4

P2(Probability of class 0) = 0.6

Parent node GINI = 1 – $(0.4)^2$ – $(0.6)^2$ =0.48

- If the split is made on Gender:

1. GINI for Sub-node Male:

P1=0.5, P2=0.5

Gini = 1-$(0.5)^2$-$(0.5)^2$ = 0.5

2. GINI for Sub-node Female:

P1=0.25 P2=0.75

Gini = 1-$(0.25)^2$-$(0.75)^2$ = 0.375

3. Weighted GINI for Gender split:

6/10 (0.5) + 4/10(0.375) = 0.45

4. Gini Gain = GINI of Parent node - weighted GINI of nodes = 0.48-0.45=0.03

- If the split is made on Occupation:

1. GINI for Sub-node Self-employed:

P1=0.4, P2=0.6

Gini = 1-$(0.4)^2$-$(0.6)^2$ = 0.48

2. GINI for Sub-node Salary:

P1=0.4 P2=0.6

Gini = 1-$(0.4)^2$-$(0.6)^2$ = 0.48

3. Weighted GINI for Occupation split:

5/10 (0.48) + 5/10(0.48) = 0.48

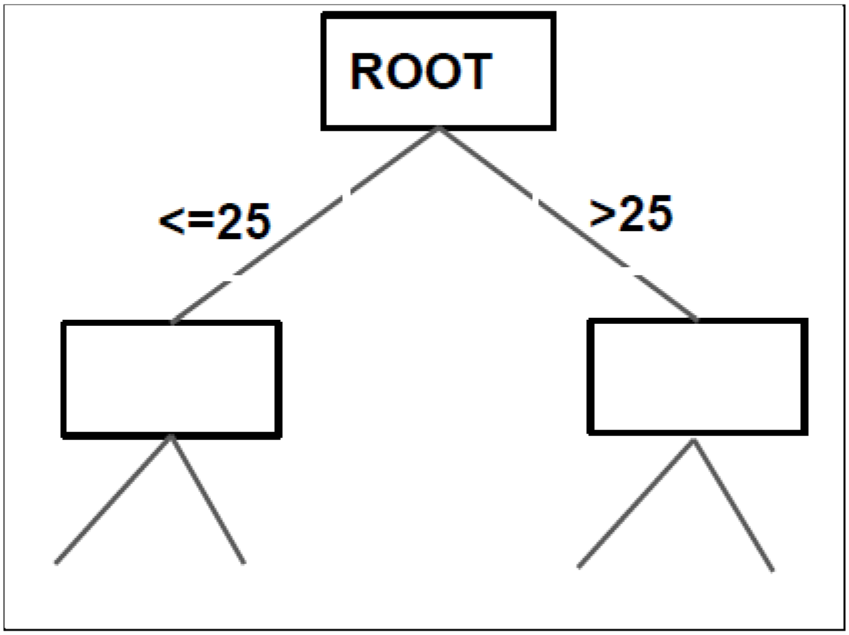
4. Gini Gain = GINI of Parent node - weighted GINI of nodes = 0.48-0.48=0

- If the split is made on Age:

Age is a continuous variable. Gini Gain is computed for all possible splits as shown below.

	Left	Right	Gini Split	Gini Gain
<=22,>22	0.5	0.47	0.48	0.48-0.48 = 0
<=23,>23	0.5	0.44	0.47	0.48-0.47 = 0.01
<=24,>24	0.5	0.38	0.45	0.48-0.45 = 0.03
<=25,>25	0.5	0	0.40	0.48-0.40 = 0.08

As we can see the GINI GAIN is maximum if split is made on Age <=25 and Age > 25. Hence the root node will split on Age as shown below.

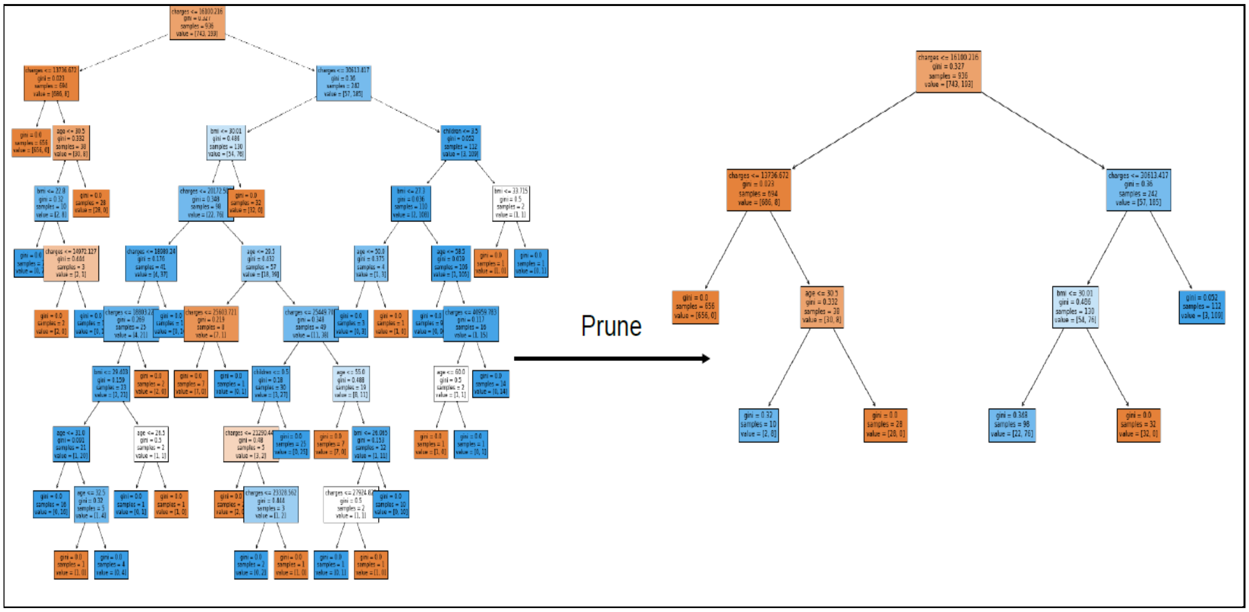


Overfitting

In decision trees, overfitting occurs when the tree is allowed to grow fully. The trees become too large and complex. So it fits perfectly with the training data set, but it effects the accuracy when predicting samples that are not part of the training set (test set)

- One of the methods used to avoid overfitting in decision trees is called **pruning.**
- A simple way to prune a decision tree is by limiting the depth of the tree to avoid overfitting.

For example the tree on the right below is generated with a max depth of 2 while the tree on the left has no depth restriction (and hence overfits the data)



How to prevent overfitting through Pruning:

Ideally we would like a tree that does not overfit the given data. There are several methods to avoid overfitting in building decision trees.

- **Pre-Pruning:** In pre-pruning, the growth of the decision tree is stopped before it fully grows and overfits the training data. It is also known as early stopping.
- **Post-Pruning:** In post-pruning, the decision tree is allowed to grow fully and then non-significant branches are removed.

Pre Pruning can be done by bounding the following parameters:

- **max_depth** - The maximum depth of the tree. If set to 'None', then nodes are expanded until all leaves are pure. Higher the value, more complex the tree
- **min_samples_split** - The minimum number of samples required to split a node. Doesn't split any node that is smaller than this number. Higher the values, less complex the tree
- **min_samples_leaf** - The minimum number of samples required at a leaf node. All leaf nodes have at least these many data points. Higher the value, less complex the tree

The process of trying different aspects of all these parameters can be referred to as "hyperparameter tuning". Grid Search is a process of searching the best combination of hyperparameters from a predefined set of values. In Python GridSearchCV( ) is an implementation of Grid Search with Cross Validation.

Cross validation is a mechanism that allows us to test a model repeatedly on data that was not used in training to build the model. A common approach is simply to choose the tree with minimum cross validation error.

Post Pruning can be done by bounding the following parameter:

- **Cost complexity-** It chooses a CP (Complexity Parameter) and requires each split to decrease relative error by at least that amount.

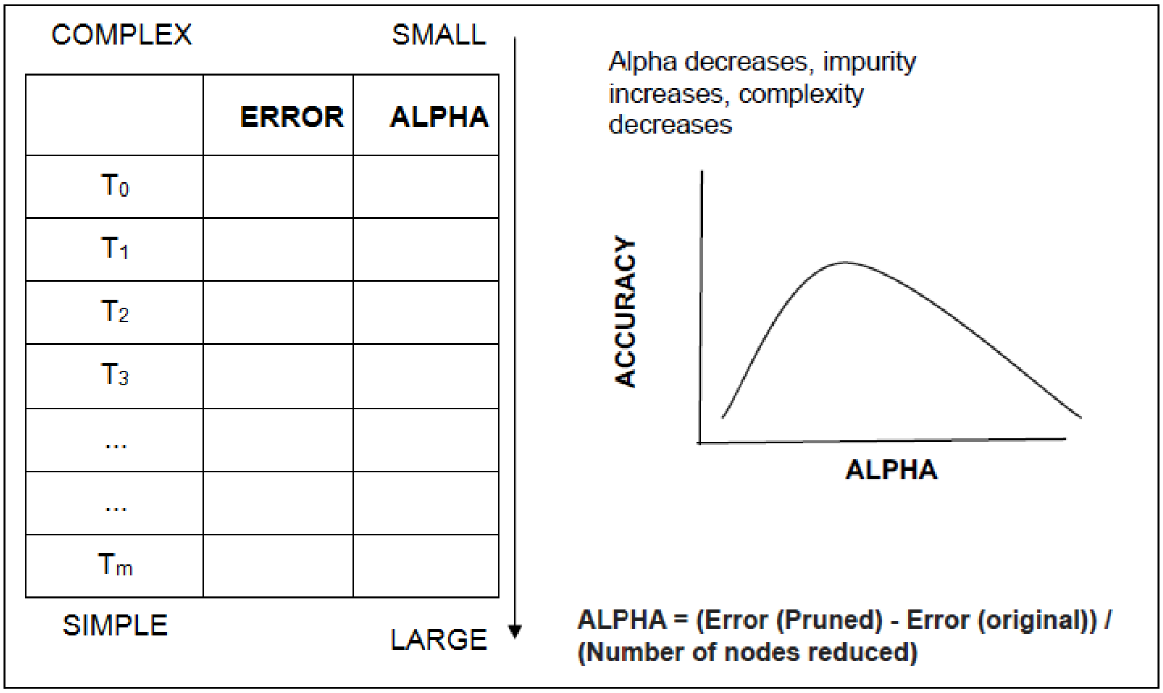
Starting from the Full tree, create a sequence of trees that are sequentially smaller (pruned)

At each step the algorithm

1. try removing each possible subtree
2. find the ‘relative error decrease per node’ for that subtree -Complexity parameter, α
3. And remove the subtree with the minimum

With the list of subtrees, one usually reverts back to using cross-validation errors to find the best final pruned tree.

The relationship among the size of the tree, CV error and α is shown below.



	ERROR	**ALPHA**
T_0		
T_1		
T_2		
T_3		
...		
...		
T_m		

ALPHA = (Error (Pruned) - Error (original)) / (Number of nodes reduced)

Impurity measures in Decision Trees:

	GINI INDEX	ENTROPY	INFORMATION GAIN	VARIANCE
When to use	Classification	Classification	Classification	Regression
Formula	$1 - \Sigma p_i^2$	$-\Sigma p_i \log(p_i)$	$E(Y) - E(Y\|X)$	$\Sigma(x - \bar{x})^2/N$
Range	0 to 0.5 0=most pure 0.5=most impure	0 to 1 0=most pure 1 = most impure	0 to 1 0=less gain 1 = more gain	>=0
Characteristics	Easy to compute, Non-additive	Computationally intensive, Additive	Computationally intensive	The most common measure of spread

Regression Trees:

- Regression trees work the same as classification trees, where you start building a tree by checking feature by feature and consider which feature to choose by exhausting all possible splits of that feature.
- And then choosing that feature as best split and iterating this and building it out exactly the same as classification trees.
- Instead of Gini Impurity, variance is used in regression trees. we will find the split that minimizes the variance amongst the two different children nodes.

Advantages of Decision Trees

- Easy to understand and interpret
- Useful in data exploration as it give the splitting base on the significance of variables
- Not influenced by the outlier/Null values and hence requires less data cleaning. Require less time and effort during data pr -processing than other algorithms.
- Can handle both continuous and categorical variables

- Does not require any underlying assumptions in data. Works with both linearly and nonlinearly related variables.

Disadvantages of Decision Trees

- A small change in the data-set can result in a large change in the structure of the decision tree causing instability in the model.
- Large trees can be difficult to interpret.
- Tends to overfit.