

dataaspirant.com

How Decision Tree Algorithm works

Rahul Saxena

15-18 minutos



Decision Tree Algorithm

Decision Tree algorithm belongs to the family of [supervised learning algorithms](#). Unlike other supervised learning algorithms, decision tree algorithm can be used for solving [regression and classification problems](#) too.

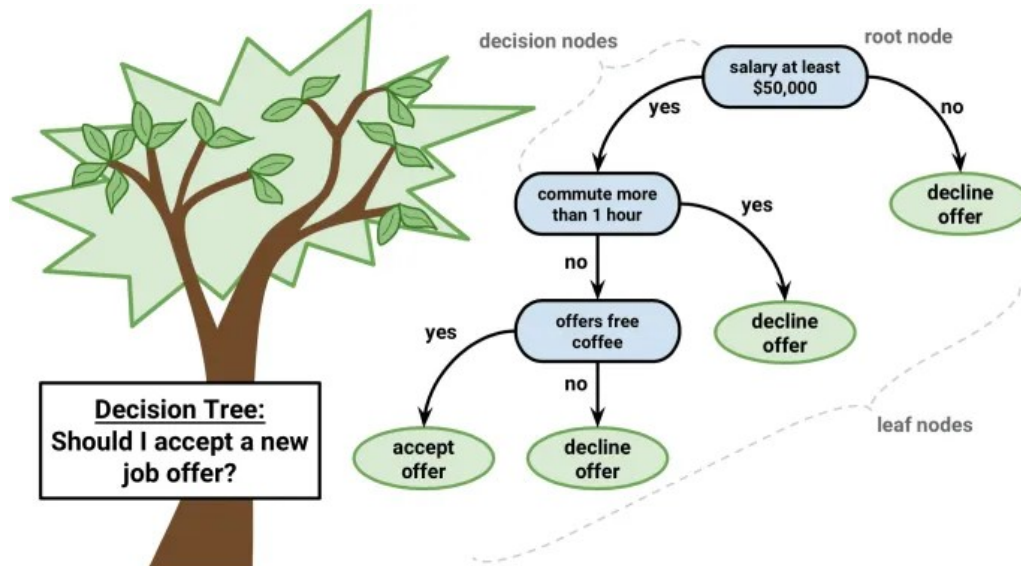
The general motive of using Decision Tree is to create a training

model which can use to predict class or value of target variables by **learning decision rules** inferred from prior data(training data).

The understanding level of Decision Trees algorithm is so easy compared with other classification algorithms. The decision tree algorithm tries to solve the problem, by using tree representation. Each **internal node** of the tree corresponds to an attribute, and each **leaf node** corresponds to a class label.

Decision Tree Algorithm Pseudocode

1. Place the best attribute of the dataset at the **root** of the tree.
2. Split the training set into **subsets**. Subsets should be made in such a way that each subset contains data with the same value for an attribute.
3. Repeat step 1 and step 2 on each subset until you find **leaf nodes** in all the branches of the tree.



Decision Tree classifier, **Image credit:** www.packtpub.com

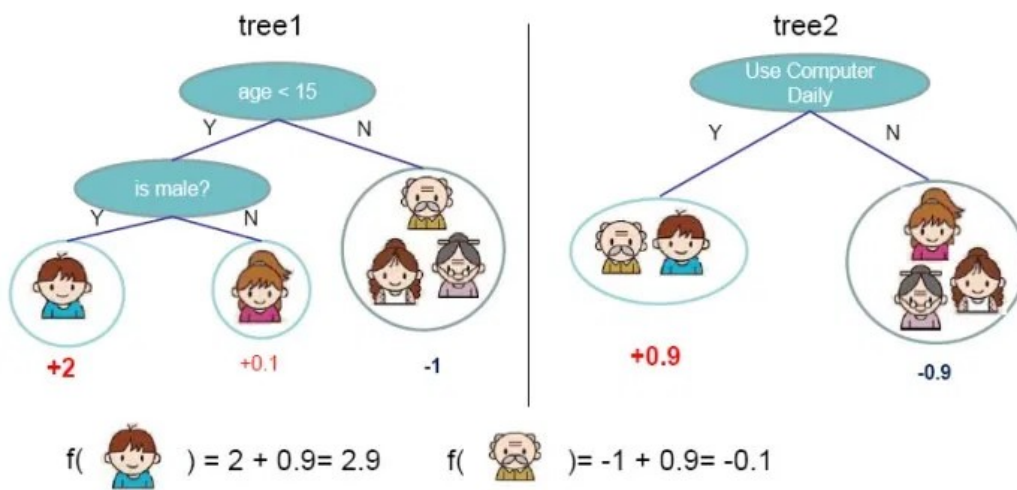
In decision trees, for predicting a class label for a record we start from the **root** of the tree. We compare the values of the root attribute with record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

We continue comparing our record's attribute values with other **internal nodes** of the tree until we reach a **leaf node** with predicted class value. As we know how the modeled decision tree can be used to predict the target class or the value. Now let's understanding how we can create the decision tree model.

Assumptions while creating Decision Tree

The below are the some of the assumptions we make while using Decision tree:

- At the beginning, the whole training set is considered as the **root**.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.



Decision tree model example Image

Credit: <http://zhanpengfang.github.io/>

Decision Trees follow **Sum of Product (SOP)** representation. For the above images, you can see how we can predict **can we accept the new job offer?** and **Use computer daily?** from traversing for the root node to the leaf node.

It's a sum of product representation. The Sum of product(SOP) is also known as **Disjunctive Normal Form**. For a class, every branch from the root of the tree to a leaf node having the same class is a conjunction(product) of values, different branches ending in that class form a disjunction(sum).

The primary challenge in the decision tree implementation is to identify which attributes do we need to consider as the root node and each level. Handling this is know the attributes selection. We have different attributes selection measure to identify the attribute which can be considered as the root note at each level.

The popular attribute selection measures:

- Information gain
- Gini index

Attributes Selection

If dataset consists of “n” attributes then deciding which attribute to place at the root or at different levels of the tree as internal nodes is a complicated step. By just randomly selecting any node to be the root can't solve the issue. If we follow a random approach, it may give us bad results with low accuracy.

For solving this attribute selection problem, researchers worked and devised some solutions. They suggested using some *criterion* like **information gain**, **gini index**, etc. These criterions will calculate values for every attribute. The values are sorted, and attributes are placed in the tree by following the order i.e, the attribute with a high value(in case of information gain) is placed at the root.

While using information Gain as a criterion, we assume attributes to be categorical, and for gini index, attributes are assumed to be continuous.

Information Gain

By using information gain as a criterion, we try to estimate the information contained by each attribute. We are going to use some points deducted from [information theory](#).

To measure the randomness or uncertainty of a random variable X is defined by **Entropy**.

For a binary classification problem with only two classes, positive and negative class.

- If all examples are positive or all are negative then entropy will be zero i.e, low.
- If half of the records are of positive class and half are of negative class then entropy is one i.e, high.

$$H(X) = \mathbb{E}_X[I(x)] = - \sum_{x \in X} p(x) \log p(x).$$

By calculating **entropy measure** of each attribute we can calculate their **information gain**. Information Gain calculates the expected reduction in entropy due to sorting on the attribute. Information gain can be calculated.

To get a clear understanding of calculating **information gain & entropy**, we will try to implement it on a sample data.

Example: Construct a Decision Tree by using “information gain” as a criterion

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

We are going to use this data sample. Let's try to use information

gain as a criterion. Here, we have 5 columns out of which 4 columns have continuous data and 5th column consists of class labels.

A, B, C, D attributes can be considered as predictors and E column class labels can be considered as a target variable. For constructing a decision tree from this data, we have to convert continuous data into categorical data.

We have chosen some random values to categorize each attribute:

A	B	C	D
≥ 5	≥ 3.0	≥ 4.2	≥ 1.4
< 5	< 3.0	< 4.2	< 1.4

There are **2 steps for calculating information gain** for each attribute:

1. Calculate entropy of Target.
2. Entropy for every attribute A, B, C, D needs to be calculated. Using information gain formula we will subtract this entropy from the entropy of target. The result is Information Gain.

The entropy of Target: We have 8 records with negative class and 8 records with positive class. So, we can directly estimate the entropy of target as 1.

Variable E	
Positive	Negative
8	8

Calculating entropy using formula:

$$\begin{aligned}
 E(8,8) &= -1 * ((p(+ve) * \log_2(p(+ve))) + (p(-ve) * \log_2(p(-ve)))) \\
 &= -1 * ((8/16) * \log_2(8/16)) + (8/16) * \log_2(8/16)) \\
 &= 1
 \end{aligned}$$

Information gain for Var A

Var A has value ≥ 5 for 12 records out of 16 and 4 records with value < 5 value.

- For Var A ≥ 5 & class == positive: 5/12
- For Var A ≥ 5 & class == negative: 7/12
- Entropy(5,7) = $-1 * ((5/12) * \log_2(5/12) + (7/12) * \log_2(7/12)) = 0.9799$
- For Var A < 5 & class == positive: 3/4
- For Var A < 5 & class == negative: 1/4
- Entropy(3,1) = $-1 * ((3/4) * \log_2(3/4) + (1/4) * \log_2(1/4)) = 0.81128$

$$\begin{aligned} \text{Entropy}(\text{Target}, A) &= P(\geq 5) * E(5,7) + P(< 5) * E(3,1) \\ &= (12/16) * 0.9799 + (4/16) * 0.81128 = 0.937745 \end{aligned}$$

$$\text{Information Gain(IG)} = E(\text{Target}) - E(\text{Target}, A) = 1 - 0.937745 = 0.062255$$

Information gain for Var B

Var B has value ≥ 3 for 12 records out of 16 and 4 records with value < 3 value.

- For Var B ≥ 3 & class == positive: 8/12
- For Var B ≥ 3 & class == negative: 4/12
- Entropy(8,4) = $-1 * ((8/12) * \log_2(8/12) + (4/12) * \log_2(4/12)) = 0.39054$
- For Var B < 3 & class == positive: 0/4
- For Var B < 3 & class == negative: 4/4
- Entropy(0,4) = $-1 * ((0/4) * \log_2(0/4) + (4/4) * \log_2(4/4)) = 0$

$$\begin{aligned} \text{Entropy}(\text{Target}, B) &= P(\geq 3) * E(8,4) + P(< 3) * E(0,4) \\ &= (12/16) * 0.39054 + (4/16) * 0 = 0.292905 \end{aligned}$$

$$\text{Information Gain(IG)} = E(\text{Target}) - E(\text{Target}, B) = 1 - 0.292905 = 0.707095$$

Information gain for Var C

Var C has value ≥ 4.2 for 6 records out of 16 and 10 records with value < 4.2 value.

- For Var C ≥ 4.2 & class == positive: 0/6

- For Var C ≥ 4.2 & class == negative: 6/6
- Entropy(0,6) = 0
- For Var C < 4.2 & class == positive: 8/10
- For Var C < 4.2 & class == negative: 2/10
- Entropy(8,2) = 0.72193

$$\text{Entropy}(\text{Target}, C) = P(\geq 4.2) * E(0,6) + P(< 4.2) * E(8,2)$$

$$= (6/16) * 0 + (10/16) * 0.72193 = 0.4512$$

$$\text{Information Gain(IG)} = E(\text{Target}) - E(\text{Target}, C) = 1 - 0.4512 = 0.5488$$

Information gain for Var D

Var D has value ≥ 1.4 for 5 records out of 16 and 11 records with value < 1.4 value.

- For Var D ≥ 1.4 & class == positive: 0/5
- For Var D ≥ 1.4 & class == negative: 5/5
- Entropy(0,5) = 0
- For Var D < 1.4 & class == positive: 8/11
- For Var D < 1.4 & class == negative: 3/11
- Entropy(8,3) = $-1 * ((8/11) * \log_2(8/11) + (3/11) * \log_2(3/11))$
= 0.84532

$$\text{Entropy}(\text{Target}, D) = P(\geq 1.4) * E(0,5) + P(< 1.4) * E(8,3)$$

$$= 5/16 * 0 + (11/16) * 0.84532 = 0.5811575$$

$$\text{Information Gain(IG)} = E(\text{Target}) - E(\text{Target}, D) = 1 - 0.5811575 = 0.4189$$

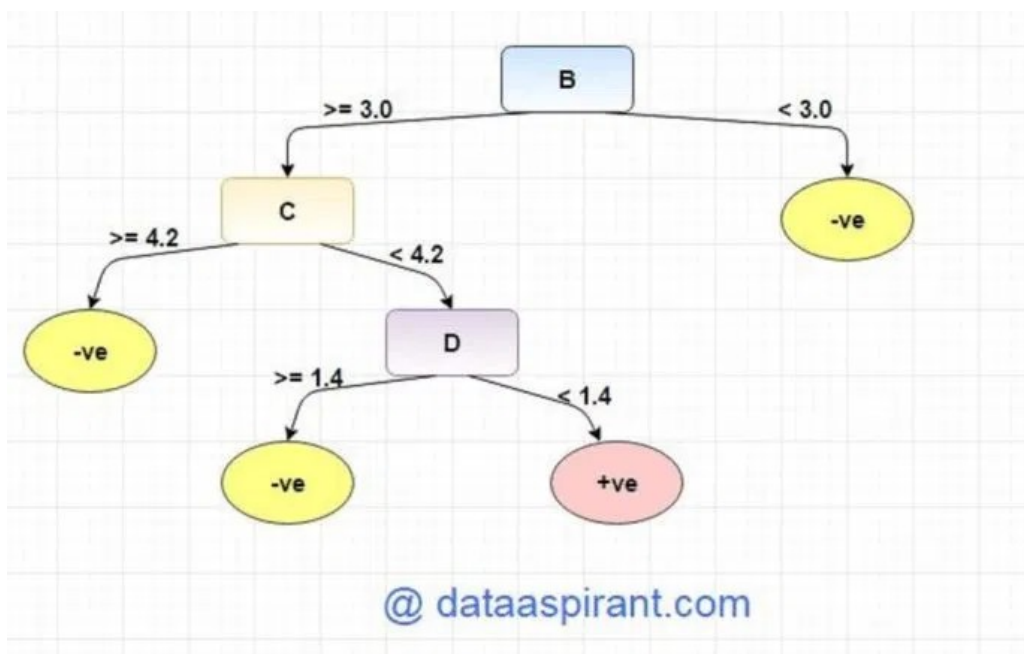
		Target	
		Positive	Negative
A	≥ 5.0	5	7
	< 5	3	1
Information Gain of A = 0.062255			

		Target	
		Positive	Negative
B	≥ 3.0	8	4
	< 3.0	0	4

		Information Gain of B= 0.7070795																				
		<table> <tr> <th colspan="2"></th><th colspan="2">Target</th></tr> <tr> <th colspan="2"></th><th>Positive</th><th>Negative</th></tr> <tr> <td rowspan="2">C</td><td>≥ 4.2</td><td>0</td><td>6</td></tr> <tr> <td>< 4.2</td><td>8</td><td>2</td></tr> <tr> <td colspan="4">Information Gain of C= 0.5488</td></tr> </table>				Target				Positive	Negative	C	≥ 4.2	0	6	< 4.2	8	2	Information Gain of C= 0.5488			
		Target																				
		Positive	Negative																			
C	≥ 4.2	0	6																			
	< 4.2	8	2																			
Information Gain of C= 0.5488																						
		<table> <tr> <th colspan="2"></th><th colspan="2">Target</th></tr> <tr> <th colspan="2"></th><th>Positive</th><th>Negative</th></tr> <tr> <td rowspan="2">D</td><td>≥ 1.4</td><td>0</td><td>5</td></tr> <tr> <td>< 1.4</td><td>8</td><td>3</td></tr> <tr> <td colspan="4">Information Gain of D= 0.41189</td></tr> </table>				Target				Positive	Negative	D	≥ 1.4	0	5	< 1.4	8	3	Information Gain of D= 0.41189			
		Target																				
		Positive	Negative																			
D	≥ 1.4	0	5																			
	< 1.4	8	3																			
Information Gain of D= 0.41189																						

From the above Information Gain calculations, we can build a decision tree. We should place the attributes on the tree according to their values.

An Attribute with better value than other should position as root and A branch with entropy 0 should be converted to a leaf node. A branch with entropy more than 0 needs further splitting.



Gini Index

Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified. It means an attribute with lower gini index should be preferred.

Example: Construct a Decision Tree by using “gini index” as a criterion

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.5	0.2	positive
3	5	3.4	1.5	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.5	0.2	positive
7	4.8	3.1	1.5	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	5.4	3.2	4.5	1.5	negative
11	5.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	5.5	2.8	4.5	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	5.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

We are going to use same data sample that we used for information gain example. Let's try to use gini index as a criterion. Here, we have 5 columns out of which 4 columns have continuous data and 5th column consists of class labels.

A, B, C, D attributes can be considered as predictors and E column class labels can be considered as a target variable. For constructing a decision tree from this data, we have to convert continuous data into categorical data.

We have chosen some random values to categorize each attribute:

A	B	C	D
≥ 5	≥ 3.0	≥ 4.2	≥ 1.4
< 5	< 3.0	< 4.2	< 1.4

Gini Index

Gini Index for Var A

Var A has value ≥ 5 for 12 records out of 16 and 4 records with

value <5 value.

- For Var A ≥ 5 & class == positive: 5/12
- For Var A ≥ 5 & class == negative: 7/12
- $\text{gini}(5,7) = 1 - ((5/12)^2 + (7/12)^2) = 0.4860$
- For Var A <5 & class == positive: 3/4
- For Var A <5 & class == negative: 1/4
- $\text{gini}(3,1) = 1 - ((3/4)^2 + (1/4)^2) = 0.375$

By adding weight and sum each of the gini indices:

$$\text{gini}(\text{Target}, A) = (12/16) * (0.486) + (4/16) * (0.375) = 0.45825$$

Gini Index for Var B

Var B has value ≥ 3 for 12 records out of 16 and 4 records with value <3 value.

- For Var B ≥ 3 & class == positive: 8/12
- For Var B ≥ 3 & class == negative: 4/12
- $\text{gini}(8,4) = 1 - ((8/12)^2 + (4/12)^2) = 0.446$
- For Var B <3 & class == positive: 0/4
- For Var B <3 & class == negative: 4/4
- $\text{gini}(0,4) = 1 - ((0/4)^2 + (4/4)^2) = 0$

$$\text{gini}(\text{Target}, B) = (12/16) * 0.446 + (4/16) * 0 = 0.3345$$

Gini Index for Var C

Var C has value ≥ 4.2 for 6 records out of 16 and 10 records with value <4.2 value.

- For Var C ≥ 4.2 & class == positive: 0/6
- For Var C ≥ 4.2 & class == negative: 6/6
- $\text{gini}(0,6) = 1 - ((0/6)^2 + (6/6)^2) = 0$
- For Var C < 4.2 & class == positive: 8/10
- For Var C < 4.2 & class == negative: 2/10

- $\text{gin}(8,2) = 1 - \left(\left(\frac{8}{10} \right)^2 + \left(\frac{2}{10} \right)^2 \right) = 0.32$

$$\text{gini}(\text{Target}, C) = \left(\frac{6}{16} \right) * 0 + \left(\frac{10}{16} \right) * 0.32 = 0.2$$

Gini Index for Var D

Var D has value ≥ 1.4 for 5 records out of 16 and 11 records with value < 1.4 value.

- For Var D ≥ 1.4 & class == positive: 0/5
- For Var D ≥ 1.4 & class == negative: 5/5
- $\text{gini}(0,5) = 1 - \left(\left(\frac{0}{5} \right)^2 + \left(\frac{5}{5} \right)^2 \right) = 0$
- For Var D < 1.4 & class == positive: 8/11
- For Var D < 1.4 & class == negative: 3/11
- $\text{gin}(8,3) = 1 - \left(\left(\frac{8}{11} \right)^2 + \left(\frac{3}{11} \right)^2 \right) = 0.397$

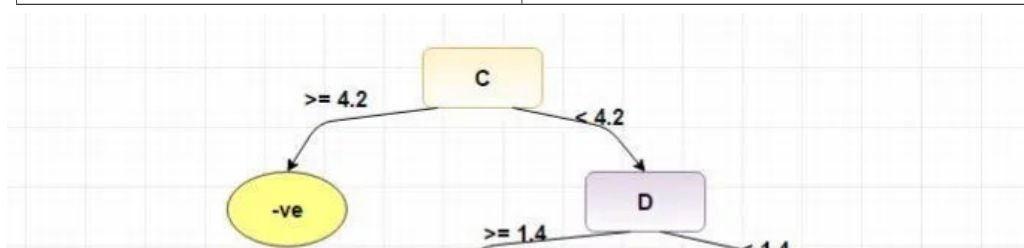
$$\text{gini}(\text{Target}, D) = \left(\frac{5}{16} \right) * 0 + \left(\frac{11}{16} \right) * 0.397 = 0.273$$

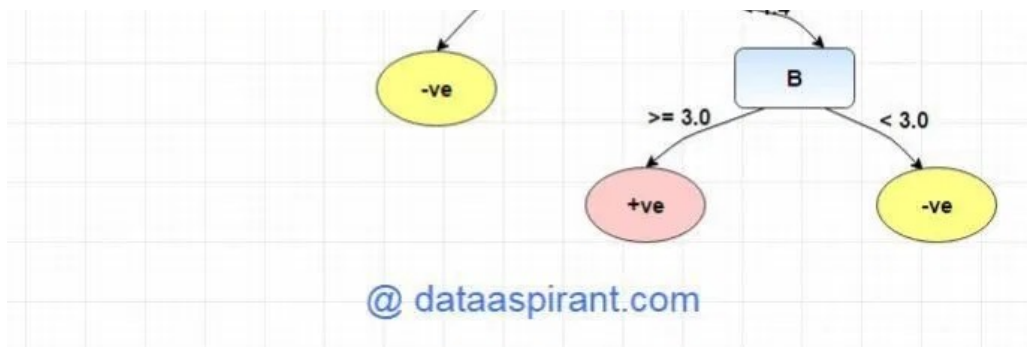
		wTarget	
		Positive	Negative
A	≥ 5.0	5	7
	< 5	3	1
Ginin Index of A = 0.45825			

		Target	
		Positive	Negative
B	≥ 3.0	8	4
	< 3.0	0	4
Gini Index of B= 0.3345			

		Target	
		Positive	Negative
C	≥ 4.2	0	6
	< 4.2	8	2
Gini Index of C= 0.2			

		Target	
		Positive	Negative
D	≥ 1.4	0	5
	< 1.4	8	3
Gini Index of D= 0.273			





Overfitting

Overfitting is a practical problem while building a decision tree model. The model is having an issue of overfitting is considered when the algorithm continues to go deeper and deeper in the to reduce the training set error but results with an increased test set error i.e, Accuracy of prediction for our model goes down. It generally happens when it builds many branches due to outliers and irregularities in data.

Two approaches which we can use to avoid overfitting are:

- Pre-Pruning
- Post-Pruning

Pre-Pruning

In pre-pruning, it stops the tree construction bit early. It is preferred not to split a node if its goodness measure is below a threshold value. But it's difficult to choose an appropriate stopping point.

Post-Pruning

In post-pruning first, it goes deeper and deeper in the tree to build a complete tree. If the tree shows the overfitting problem then pruning is done as a post-pruning step. We use a cross-validation data to check the effect of our pruning. Using cross-validation data, it tests whether expanding a node will make an improvement or not.

If it shows an improvement, then we can continue by expanding

that node. But if it shows a reduction in accuracy then it should not be expanded i.e, the node should be converted to a leaf node.

Decision Tree Algorithm Advantages and Disadvantages

Advantages:

1. Decision Trees are easy to explain. It results in a set of rules.
2. It follows the same approach as humans generally follow while making decisions.
3. Interpretation of a complex Decision Tree model can be simplified by its visualizations. Even a naive person can understand logic.
4. The Number of hyper-parameters to be tuned is almost null.

Disadvantages:

1. There is a high probability of overfitting in Decision Tree.
2. Generally, it gives low prediction accuracy for a dataset as compared to other machine learning algorithms.
3. Information gain in a decision tree with categorical variables gives a biased response for attributes with greater no. of categories.
4. Calculations can become complex when there are many class labels.


Follow us:

[FACEBOOK](#) | [QUORA](#) | [TWITTER](#) | [GOOGLE+](#) | [LINKEDIN](#) | [REDDIT](#) | [FLIPBOARD](#) | [MEDIUM](#) | [G](#)

I hope you like this post. If you have any questions, then feel free to comment below. If you want me to write on one particular topic, then do tell it to me in the comments below.

Related Courses:

Do check out [unlimited data science courses](#)

Title & links	Details	What You Will Learn
Machine Learning A-Z: Hands-On Python & R In Data Science 	Students Enrolled:: 19,359 Course Overall Rating:: 4.6	<ul style="list-style-type: none">• Master Machine Learning on Python & R• Make robust Machine Learning models.• Handle specific topics like Reinforcement Learning, NLP and Deep Learning.• Build an army of powerful Machine Learning models and know how to combine them to solve any problem.
Machine Learning: Classification	Course Overall Rating:: 4.7	<ul style="list-style-type: none">• Describe the input and output of a classification model.• Build a classification model to predict sentiment in a product review dataset.• Analyze financial data to predict loan defaults.• Evaluate your models using precision-recall metrics.• Implement these techniques in Python (or

		in the language of your choice, though Python is highly recommended).
Practical Machine Learning	Course Overall Rating:: 4.4	<ul style="list-style-type: none">• This course will cover the basic components of building and applying prediction functions with an emphasis on practical applications.• Will provide the basic grounding in concepts such as training and tests sets, overfitting, and error rates and introduce a range of model-based and algorithmic machine learning methods• These methods including regression, classification trees, Naive Bayes, and random forests.• The course will cover the complete process of building prediction functions including data collection, feature creation, algorithms, and evaluation.