Contents

[Decision Tree Algorithm Pseudocode 1](#_Toc517100348)

[Recursive Binary Splitting 1](#_Toc517100349)

[Assumptions while creating Decision Tree 2](#_Toc517100350)

[Split Criteria 2](#_Toc517100351)

[1- Information Gain 3](#_Toc517100352)

[2 - Gini Index 7](#_Toc517100353)

[Regression Tree Split Criteria 9](#_Toc517100354)

[Stopping Criteria and Pruning 10](#_Toc517100355)

[Pre-Pruning 11](#_Toc517100356)

[Post-Pruning 11](#_Toc517100357)

[Advantages and disadvantages 12](#_Toc517100358)

[Interview Questions 12](#_Toc517100359)

[What are the key parameters of tree modeling and how can we avoid overfitting in decision trees? 12](#_Toc517100360)

# ****Decision Tree Algorithm Pseudocode****

Based on split criteria split the best attribute to be placed on the top.Criteria is based on information gain or gini index.

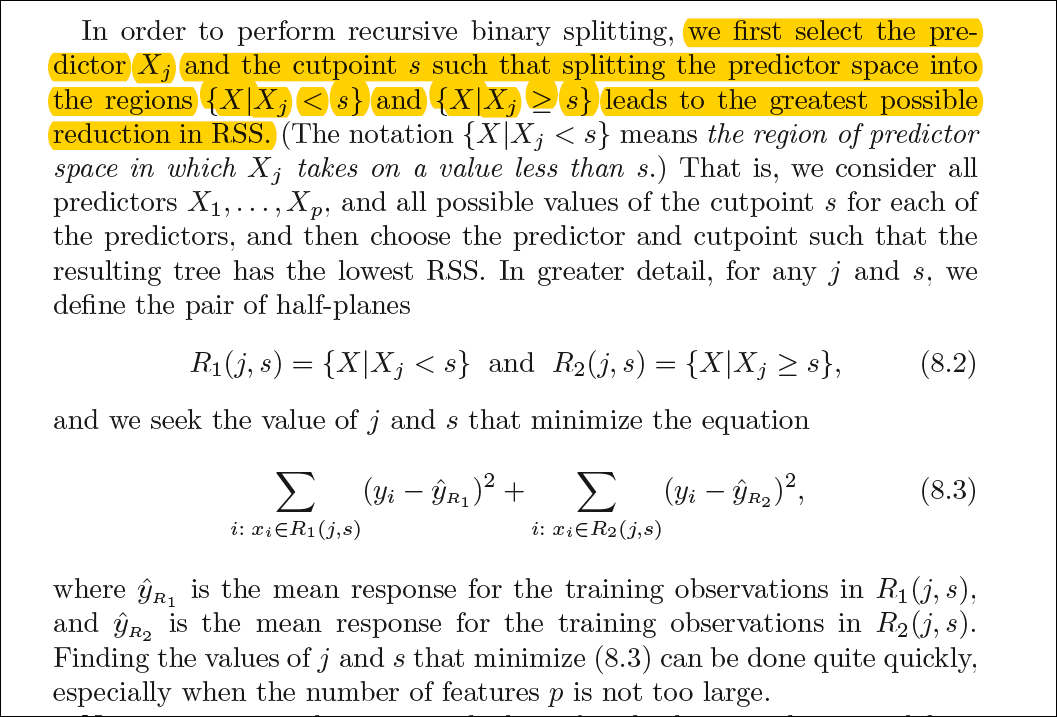
* CART (Classification and Regression Trees) → uses Gini Index(Classification) as metric.
* ID3 (Iterative Dichotomiser 3) → uses Entropy function and Information gain as metrics.

# ****Recursive Binary Splitting****

In this procedure all the features are considered and different split points are tried and tested using a cost function. The split with the best cost (or lowest cost) is selected.

Consider the earlier example of tree learned from titanic dataset. In the first split or the root, all attributes/features are considered and the training data is divided into groups based on this split. We have 3 features, so will have 3 candidate splits. Now we will calculate how much [accuracy](https://medium.com/towards-data-science/balancing-bias-and-variance-to-control-errors-in-machine-learning-16ced95724db)each split will cost us, using a function. The split that costs least is chosen, which in our example is sex of the passenger. This algorithm is recursive in nature as the groups formed can be sub-divided using same strategy. b Due to this procedure, this algorithm is also known as the greedy algorithm, as we have an excessive desire of lowering the cost. This makes the root node as best predictor/classifier.

The method which is used for this recursive binary splitting.



# 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.

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.

# Split Criteria

For classification, it is typically either [Gini impurity](http://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity) or [information gain/entropy](http://en.wikipedia.org/wiki/Information_gain_in_decision_trees) and for regression trees it is [variance](http://en.wikipedia.org/wiki/Variance).

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

|  |
| --- |
| A decision tree is built top-down from a root node and involve partitioning of data into homogenious subsets. **ID3** uses entropy to check the homogeneity of a sample. If the sample is completely homogenious then entropy is zero and if the sample is an equally divided it has entropy of one. |

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](https://en.wikipedia.org/wiki/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.

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.

**Entropy**

Information theory is a measure to define this **degree** **of disorganization** in a system known as Entropy. If the sample is completely homogeneous, then the entropy is zero and if the sample is an equally divided (50% – 50%), it has entropy of one.

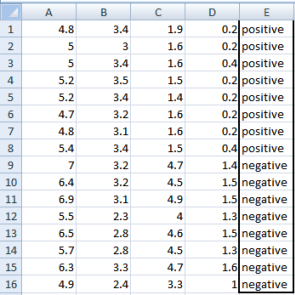
Entropy can be calculated using formula:-Entropy, Decision Tree



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

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



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:

* Calculate entropy of Target.
* 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. |

**Calculating entropy of the target variable using formula:**

|  |
| --- |
| Entropy(var A) = probability (>= critical Value) \* Entropy(+ve,-ve)  + probability (< critical Value) \* Entropy(+ve,-ve)  Entropy(+ve,-ve) = -p log p -q log q  p = probability of class positive  q probability of class negative  Information gain = Entropy(Target) – Entropy( var A) = 1 - |

E(8,8) = -1\*( (p(+ve)\*log( p(+ve)) + (p(-ve)\*log( p(-ve)) )  
= -1\*( (8/16)\*log2(8/16)) + (8/16) \* log2(8/16) )  
= 1

P = probability of success

**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)\*log2(5/12) + (7/12)\*log2(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)\*log2(3/4) + (1/4)\*log2(1/4)) = 0.81128

Entropy(Target, A) = P(>=5) \* E(5,7) + P(<5) \* E(3,1)  
= (12/16) \* 0.9799 + (4/16) \* 0.81128 = 0.937745

\textrm{Information Gain(IG) = E(Target) - E(Target,A) = 1- 0.9337745 = 0.062255}  

**Information gain for Var B**

Var B has value >=3 for 12 records out of 16 and 4 records with value <5 value.

For Var B >= 3 & class == positive: 8/12

For Var B >= 3 & class == negative: 4/12

Entropy(8,4) = -1 \* ( (8/12)\*log2(8/12) + (4/12)\*log2(4/12)) = 0.39054

For VarB <3 & class == positive: 0/4

For Var B <3 & class == negative: 4/4

Entropy(0,4) =  -1 \* ( (0/4)\*log2(0/4) + (4/4)\*log2(4/4)) = 0

Entropy(Target, B) = P(>=3) \* E(8,4) + P(<3) \* E(0,4)  
= (12/16) \* 0.39054 + (4/16) \* 0 = 0.292905

\textrm{Information Gain(IG) = E(Target) - E(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 VarC < 4.2 & class == positive: 8/10

For Var C < 4.2 & class == negative: 2/10

Entropy(8,2) = 0.72193

Entropy(Target, C) = P(>=4.2) \* E(0,6) + P(< 4.2) \* E(8,2)  
= (6/16) \* 0 + (10/16) \* 0.72193 = 0.4512

\textrm{Information Gain(IG) = E(Target) - E(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 <5 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 < 14 & class == negative: 3/11

Entropy(8,3) =  -1 \* ( (8/11)\*log2(8/11) + (3/11)\*log2(3/11)) = 0.84532

Entropy(Target, D) = P(>=1.4) \* E(0,5) + P(< 1.4) \* E(8,3)  
= 5/16 \* 0 + (11/16) \* 0.84532 = 0.5811575

|  |  |
| --- | --- |
| \textrm{Information Gain(IG) = E(Target) - E(Target,D) = 1- 0.5811575 = 0.41189} |  |
|  |  |

Information Gain of A = 0.062255

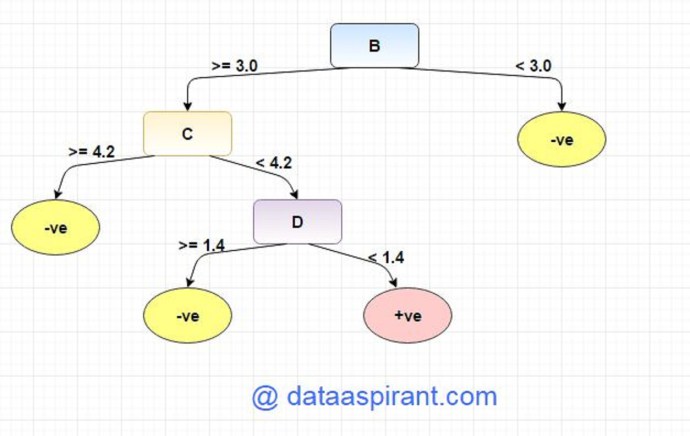
Information Gain of B= 0.7070795

Information Gain of C= 0.5488

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.



## 2 - Gini Index

## **Definition**

The Gini index is the name of the cost function used to evaluate splits in the dataset.It is used by CART

A Gini score gives an idea of how good a split is by how mixed the classes are in the two groups created by the split. A perfect separation results in a Gini score of 0, whereas the worst case split that results in 50/50 classes in each group result in a **Gini score of 0.5** (for a 2 class problem).

**Steps:**

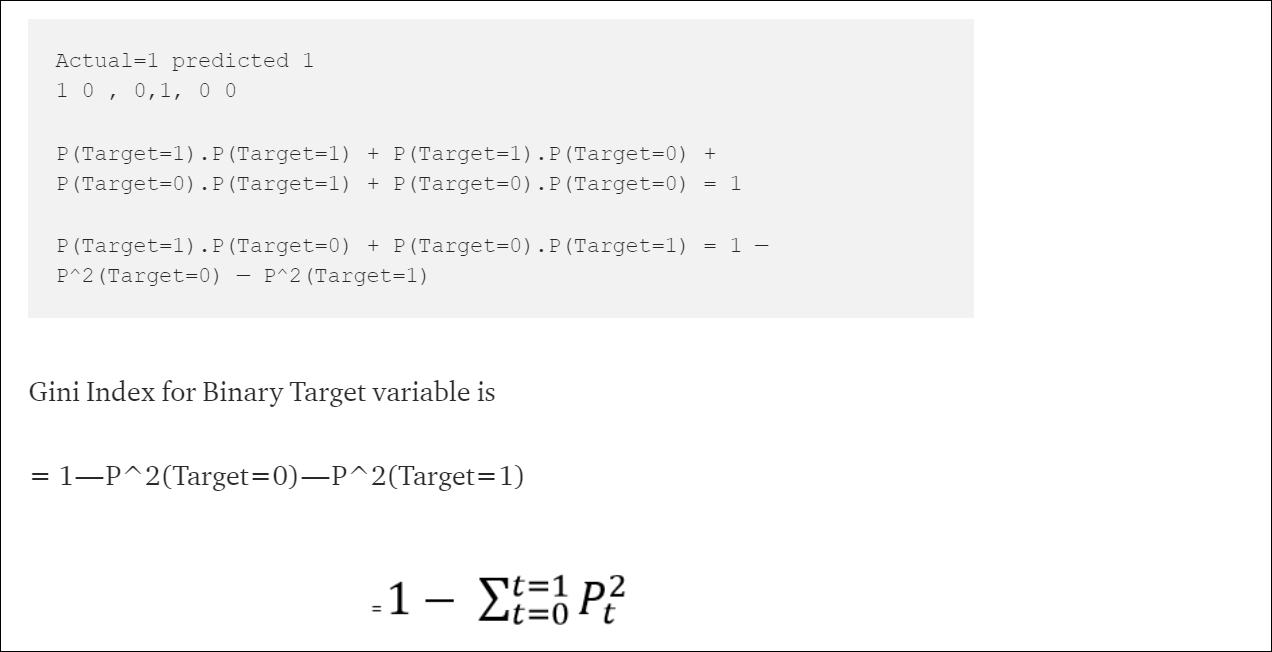
1.compute the gini index of the target variable (dependent variable)

2.for every attribute/feature:  
 1.calculate gini for all split value of the variable   
 2.calculate the gini of all the variables with all the splits   
 3.calculate the gini gain

3. pick the best gini gain attribute.  
4. Repeat until we get the tree we desired.

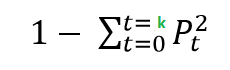
**Example**

Calculating Gini is best demonstrated with an example.

We have two groups of data with 2 rows in each group. 

pt : Proportion of observations with target variable value t. In Binary t takes value 0 and 1.

Similarly if Target Variable is categorical variable with multiple levels, the Gini Index will be still similar. If Target variable takes k different values, the Gini Index will be

[](http://dni-institute.in/blogs/wp-content/uploads/2015/07/Gini-Index-Target-k.png)

**Maximum value of Gini Index** could be when all target values are equally distributed.

For Binary Target variable, Max Gini Index value

= 1 - (1/2)2 - (1/2)2  
= 1 - 2\*(1/2)2  
= 1- 2\*(1/4)  
= 1-0.5  
= 0.5

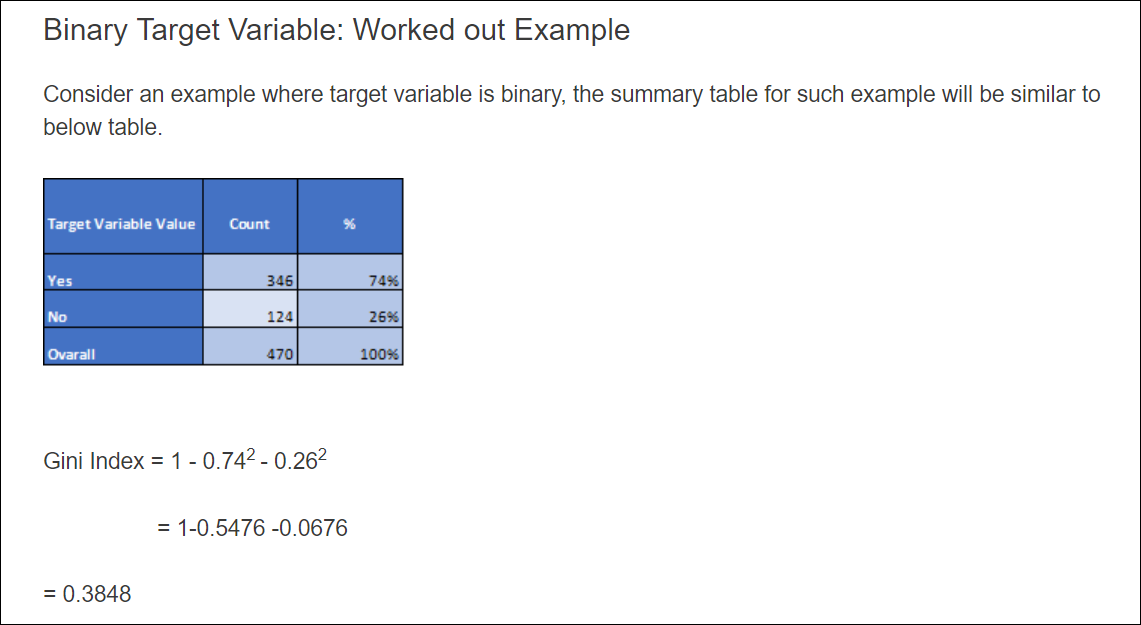
Similarly for Nominal variable with k level, the maximum value Gini Index is

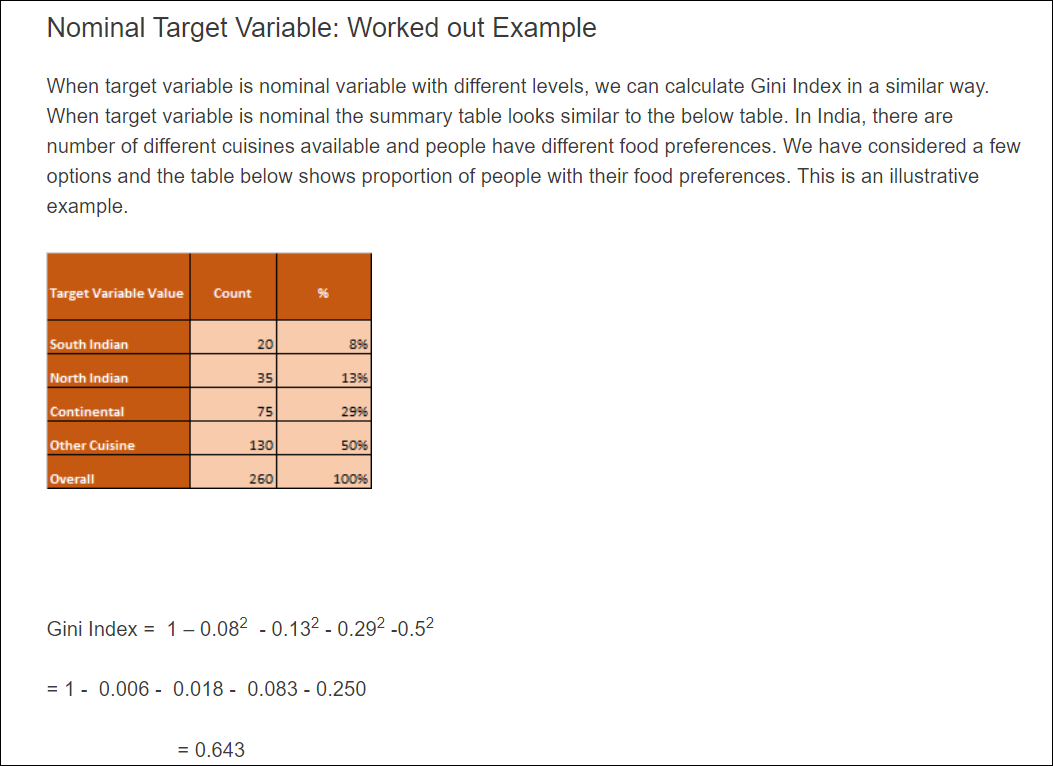
= 1 - 1/k

**Minimum value of Gini Index** will be 0 when all observations belong to one label.

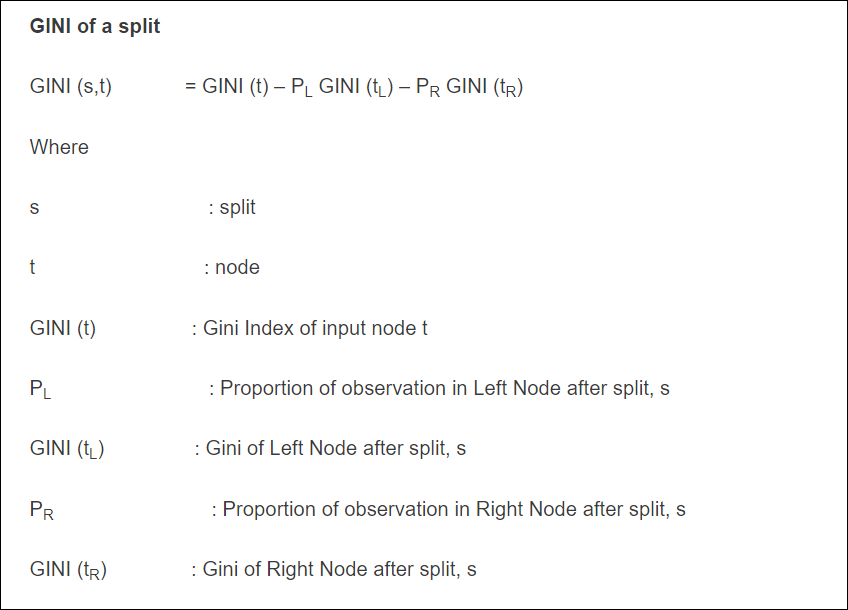
**Example**

Step 1 - Gini index of target

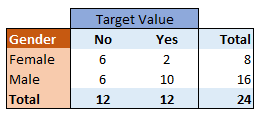


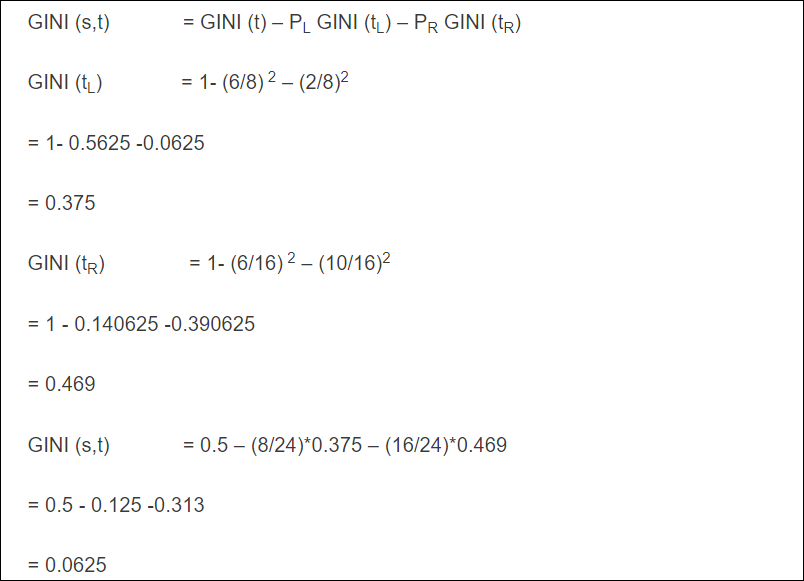


Step 2 - Gini index of split



Suppose we want to split on gender





Similarly, we need to find GINI index value for all the split points and select the best split for a variable. Also the best split points are calculated for all the variables. The best variable and the split is selected to split the input node.

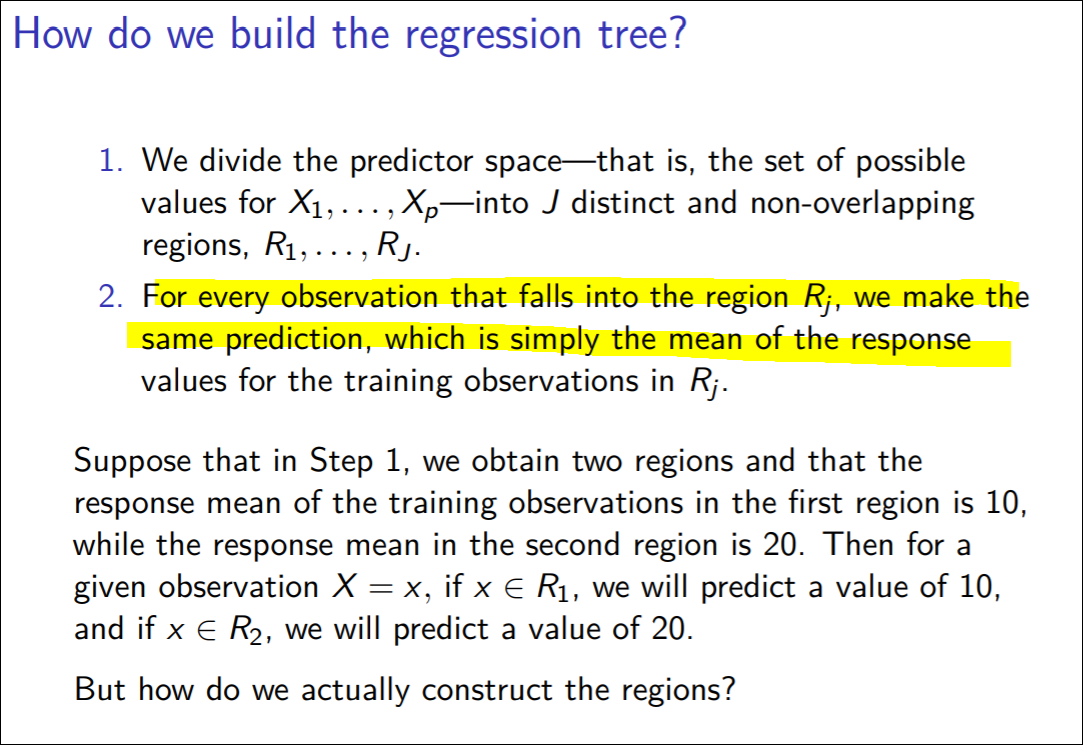
Continuous variable in classification decision tree

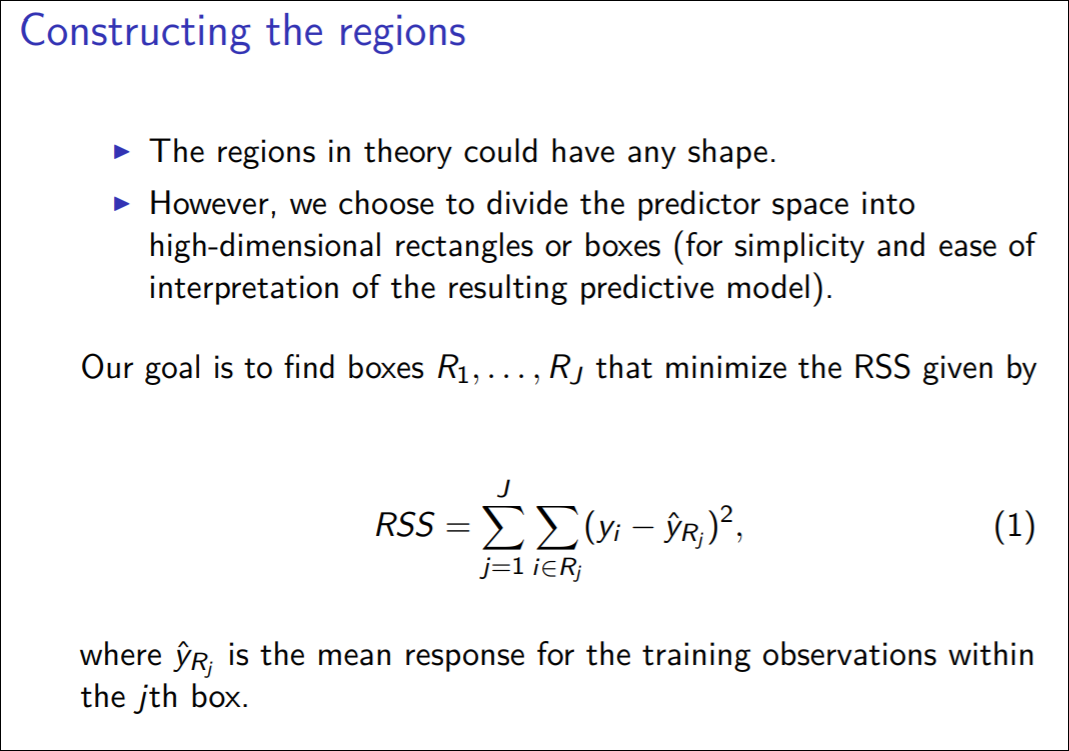
We use the threshold split for the continuous variables

To choose the threshold

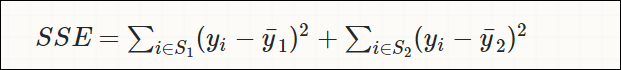
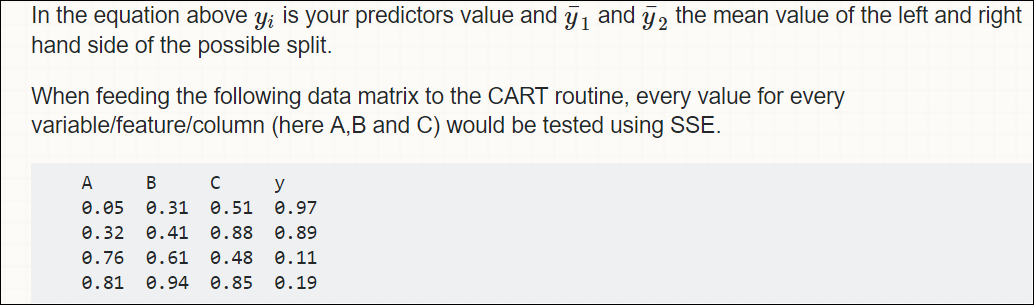
A simple idea is to find the observed value which when treated as a theshold gives the best split. Mostly 1 threshold is used in the tree( C4.5 uses single threshold tree)

## Regression Tree Split Criteria

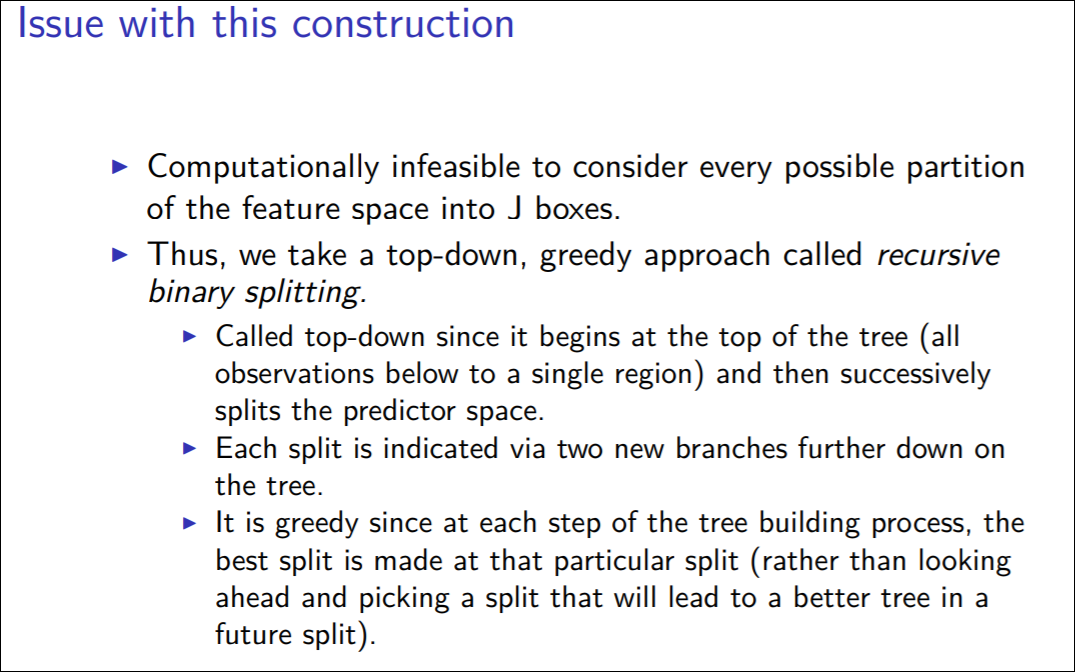




For example if there are 2 regions - one region has mean y1bar and other one has mean y2 bar.The value used for splitting is determined by testing every value for every variable, that the one which minimizes the sum of squares error (SSE) best is chosen:

The one minimizing SSE best, would be chosen for split. CART would test all possible splits using all values for variable A (0.05, 0.32, 0.76 and 0.81) and then using variable B, then C.



Other method for regression tree - Reduction in Variance

Till now, we have discussed the algorithms for categorical target variable. Reduction in variance is an algorithm used for continuous target variables (regression problems). This algorithm uses the standard formula of variance to choose the best split. The split with lower variance is selected as the criteria to split the population:

# Stopping Criteria and Pruning

You might ask when to stop growing a tree? As a problem usually has a large set of features, it results in large number of split, which in turn gives a huge tree. Such trees are complex and can lead to overfitting. So, we need to know when to stop?

One way of doing this is to **set a minimum number of training inputs to use on each leaf.** For example we can use a minimum of 10 passengers to reach a decision(died or survived), and ignore any leaf that takes less than 10 passengers.

Another way is to set **maximum depth** of your model. **Maximum depth refers to the the length of the longest path from a root to a leaf.**

**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.

**How to implement pruning it in decision tree?**

1. We first make the decision tree to a large depth.

2. Then we start at the bottom and start removing leaves which are giving us negative

returns when compared from the top.

3. Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on

that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we

will see that the overall gain is +10 and keep both leaves.

# Advantages and disadvantages

1. Brainstorming Outcomes : Decision trees help you think of all possible outcomes for an upcoming choice. The consequences of each outcome must be fully explored, so no details are missed. Taking the time to brainstorm prevents overreactions to any one variable. The graphical depiction of various alternatives makes them easier to compare with each other. The decision tree also adds transparency to the process. An independent party can see exactly how a particular decision was made.
2. Decision Tree Versatility : Decision trees can be customized for a variety for situations. The logical form is good for programmers and engineers. Technicians can also use decision trees to diagnose mechanical failures in equipment or troubleshoot auto repairs. Decision trees are also helpful for evaluating business or investment alternatives. Managers can recreate the math used in a particular decision tree to analyzing the company’s decision-making process.
3. Less data cleaning required: It requires less data cleaning compared to some other modeling techniques. It is not influenced by outliers and missing values to a fair degree.
4. Data type is not a constraint: It can handle both numerical and categorical variables.
5. Non Parametric Method: Decision tree is considered to be a non-parametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure.

# Interview Questions

### What are the key parameters of tree modeling and how can we avoid overfitting in decision trees?

Minimum samples for a node split

Minimum samples for a terminal node (leaf)

Maximum depth of tree (vertical depth)

Maximum number of terminal nodes: Can be defined in place of max\_depth. Since

binary trees are created, a depth of ‘n’ would produce a maximum of 2^n leaves.

Maximum features to consider for split

This is exactly the difference between normal decision tree & pruning. A decision tree

with constraints wont see the truck ahead and adopt a greedy approach by taking a left.

On the other hand if we use pruning, we in effect look at a few steps ahead and make a

choice.

So we know pruning is better. But how to implement it in decision tree? The idea is

simple.

1. We first make the decision tree to a large depth.

2. Then we start at the bottom and start removing leaves which are giving us negative

returns when compared from the top.

3. Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on

that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we

will see that the overall gain is +10 and keep both leaves.