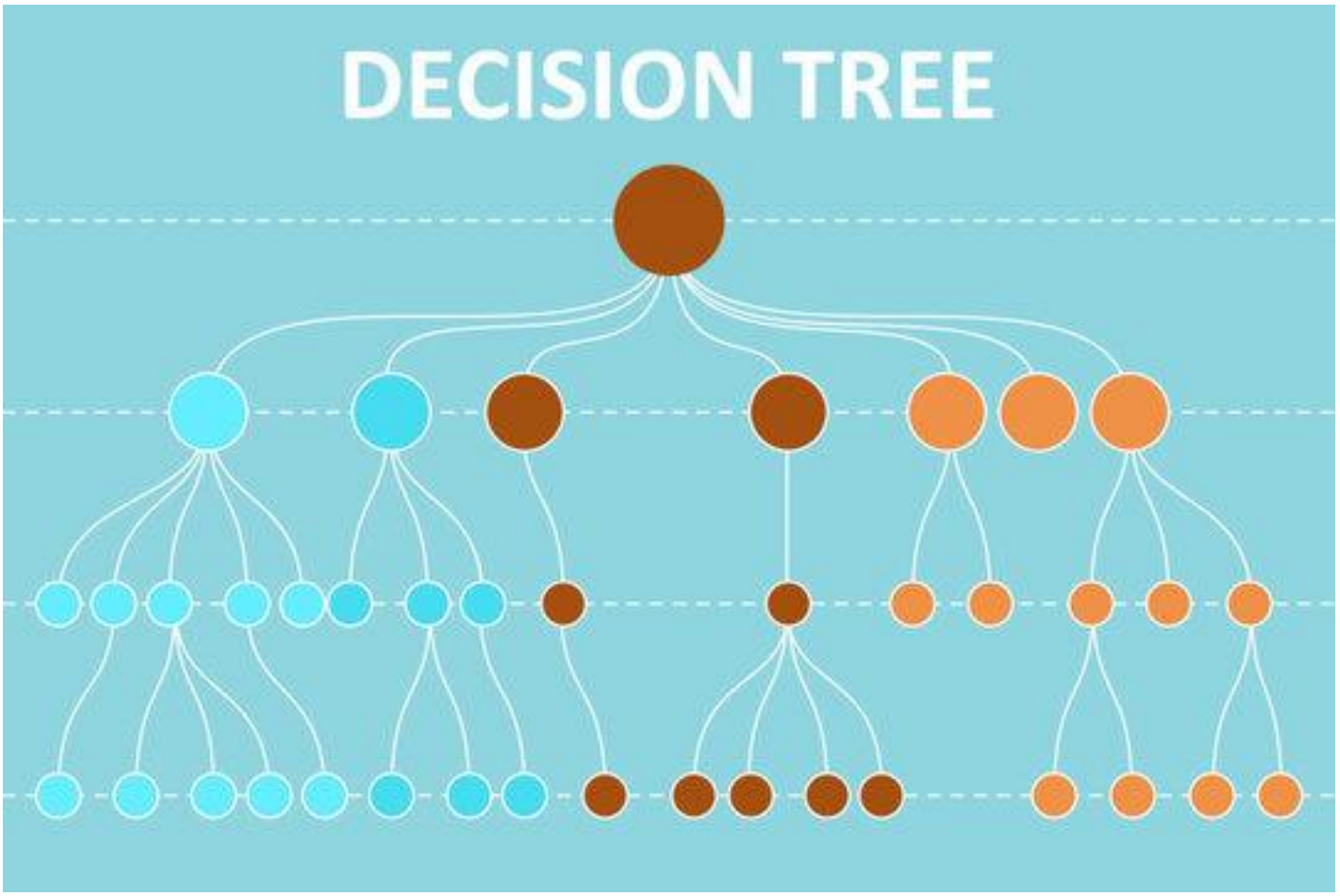


Table Of Contents

scaling is not needed for DT

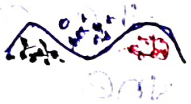
(3) Decision Tree

for both Regression & classification (supervised learning)

→ As of now we have Regression models, which makes a best fit (line) or a best fit (polynomial)

→ But in case of classification models. we have this logistic regression which can only classify by using line.

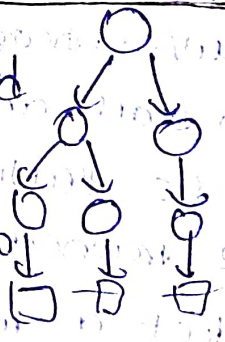
* what if our data is classified in polynomial fig



} Logistic Regression can't be used here drawback

So, this is where "decision tree" came to picture

→ As the name suggests, this algorithm works by dividing the whole dataset into a "tree-like structure" based on some rules & conditions & then gives predictions based on those conditions. Let's see how it works.



1 → First it select a 'root node' based on given cond.

2 → then the root node will split into child nodes based on cond.

3 → If any node doesn't fulfill its cond, then it again split into new cond.

4 → continues till all the cond. are met or if you have pre-defined the depth of your tree

But, how do we select a root node?

~ this is where mathematical intuition comes in

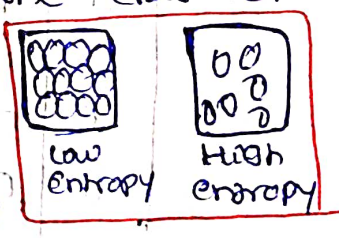
→ usually Regression trees are used for quantitative data. Incase of qualitative/categorical data we use classification trees.

→ In Regression tree, we split the nodes based on RSS (Residual Sum of Squares) criteria.

In classification, it is done using 'classification error rate', 'Gini Impurity' & 'entropy'.

we need a pure element (root node) to build tree. so to find that we can use entropy.

(*) **Entropy** is the measure of randomness of data. It gives the impurity present in the dataset.



→ when we split our node into two regions and put different observations in both the regions, the main goal is to reduce the randomness in the region & divide our data cleanly than it was in the previous node. If splitting the node doesn't lead to entropy reduction, we try to split based on different cond, or we stop.

→ A region is **clean (low entropy)** when it contains data with the same labels & **random** (high entropy) if there is a mixture of labels present.

we need to calculate entropy of each column.

$E = -P \cdot \log(P)$ P-Probability,

$E = -P \cdot \log P - q \cdot \log q$

④ **Information gain (Gn)** calculates the decrease in entropy after splitting a node. It is the difference b/w entropies before & after the split. The more the information gain, the more entropy is removed, & more clean the node.

$$G(A) = 1 - \sum \left(\frac{S_v}{S} \cdot E_i \right)$$

$$\text{Gain}(T,X) = \text{Entropy}(T) - \text{Entropy}(T,X)$$

Q) Let's take the data,

x	y	z	class
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

If we introduce new data,

x y z
1 0 1 , what is class?

Soln First we need individual column entropy.

(0x, 0y, 0z)

In each column we have (1,0) & total 2 classes (I, II)

$$G_x(1) = \left(\frac{-2}{3} \cdot \log \frac{2}{3} \right) + \left(\frac{-1}{3} \cdot \log \frac{1}{3} \right) = -0.276$$

(I) (II)

↓

we have '3' 1's in x column, out of which '2' are I class.

'1' 1 out of 3 is II class.

I

$$E_x(0) = \left(\frac{0}{-1}\cdot\log\left(\frac{0}{-1}\right)\right) + \left(\frac{-1}{1}\cdot\log 1/1\right) = 0$$

$$E_y(1) = \left(\frac{-2}{2}\cdot\log 2/2\right) + \left(\frac{0}{-2}\cdot\log\frac{0}{-2}\right) = 0$$

$$E_y(0) = \left(\frac{0}{-2}\cdot\log\frac{0}{-2}\right) + \left(\frac{-2}{2}\cdot\log 2/2\right) = 0$$

$$E_z(1) = \left(\frac{-1}{2}\cdot\log 1/2\right) + \left(-1/2\cdot\log 1/2\right) = 1$$

$$E_z(0) = \left(\frac{-1}{2}\cdot\log 1/2\right) + \left(-1/2\cdot\log 1/2\right) = 1.$$

Information Gain

$$G_x = 1 - \left[\frac{S(1)}{S}\cdot E(1) + \frac{S(0)}{S}\cdot E(0)\right]$$

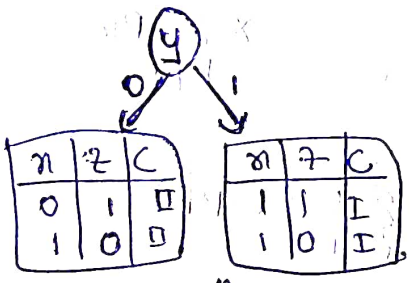
S(1): no. of 1's; S(0): no. of 0's; S: total no. of values in x

$$G_x = 1 - \left[\frac{3}{4}\times 0.276 + \frac{1}{4}\times 0\right] \cong 0.7$$

$$G_y = 1 - \left[\frac{2}{4}\times 0 + \frac{2}{4}\times 0\right] = 1$$

$$G_z = 1 - \left[\frac{2}{4}\times 1 + \frac{2}{4}\times 1\right] = 0$$

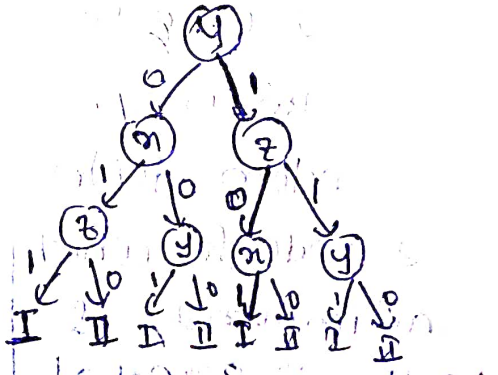
$G_y = 1$, is the highest info gain so lowest entropy. Hence it is "Root Node: (y)"



Again we have to find E_x, E_z, G_x, G_z. And largest is parent node. Then leaf node gives a class.

Find E_x, E_z, G_x, G_z.

Finally it can be like this



So, 1-0-1 → I class

Gini Impurity

Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset.

→ It is calculated by multiplying the probability that a given observation is classified into the correct class & sum of all the probabilities when that particular observ. is classified into the wrong class.

→ Let's suppose there are k no. of classes and an observation belongs to the class 'i', & it's probability is given by 'p_i'.
then probability that observation belongs to any other class other than 'i' is,

$$\sum_{k \neq i} p_k = 1 - p_i$$

then gini impurity $= \sum_{i=1}^{J} p_i \times \sum_{k \neq i} p_k$

Gini impurity value lies b/w 0 & 1,
0 - no impurity
1 - Random distr.
The node for which Gini impurity is least is selected as Root node to split.

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

$$Gini = \sum_{i=1}^{J} p_i - p_i^2$$

$$Gini = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} p_i^2$$

$$Gini = 1 - \sum_{i=1}^{J} p_i^2$$

Q) Given data,

class	gender	stay in hostel
9	M	Yes
10	F	No
8	F	Yes
8	F	No
9	M	Yes
10	M	No
11	F	Yes
11	M	Yes
8	F	Yes
9	M	No
11	M	No
11	M	Yes
10	F	No
10	M	Yes

Here understand how root node is selected by calc gini impurity.

We have 2 features which can use for node class & gender we will calculate gini for each of the feature & then select that feature which has least gini impurity

Sol:

(1)

class	stay	Total	P(Yes)	P(No)
8	Yes=2, No=1	3	2/3	1/3
9	Yes=2, No=1	3	2/3	1/3
10	Yes=1, No=3	4	1/4	3/4
11	Yes=3, No=1	4	3/4	1/4
		14		

Gini for class feature

$$G(class=8) = 1-(P(Yes))^2-(P(No))^2 = 1-(2/3)^2-(1/3)^2 = 4/9$$

$$G(class=9) = 1-(2/3)^2-(1/3)^2 = 4/9$$

$$G(class=10) = 1-(1/4)^2-(3/4)^2 = 6/16$$

$$G(class=11) = 1-(1/4)^2-(3/4)^2 = 6/16$$

Weighted sum of gini for class

$$G(class) = \frac{\text{no of class 8}}{\text{total}} * G(8) + \frac{\text{no of class 9}}{\text{total}} G(9) + \cdots$$

$$G(class) = \frac{3}{14} \times \frac{4}{9} + \frac{3}{14} \times \frac{4}{9} + \frac{4}{14} \times \frac{6}{16} + \frac{4}{14} \times \frac{6}{16}$$

$$G(class) = 0.404$$

(ii) Gini for gender

Gender	stay	n	P(Y)	P(N)
M	Yes=5, No=3	8	5/8	3/8
F	Yes=3, No=3	6	1/2	1/2
		14		

$$G(M) = 1 - (5/8)^2 - (3/8)^2 = 0.469$$

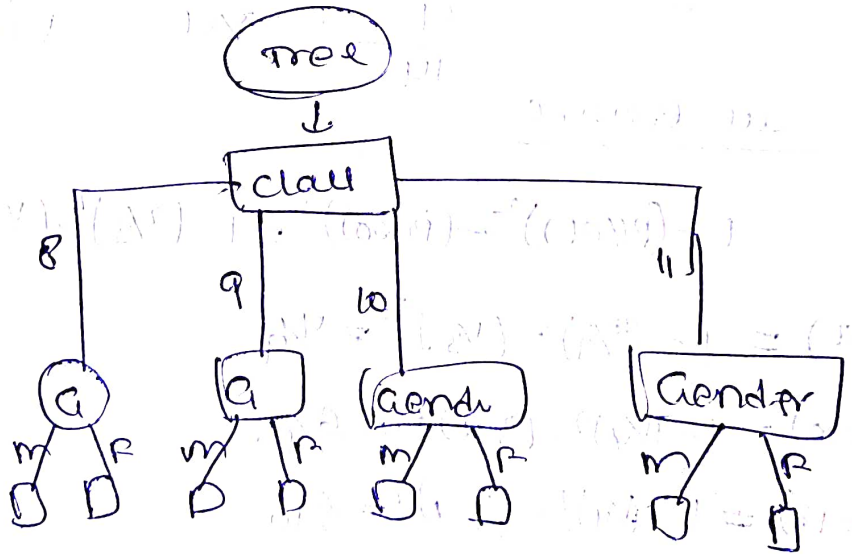
$$G(F) = 1 - (1/2)^2 - (1/2)^2 = 0.5$$

$$G(\text{Gender}) = \frac{8}{14} \times 0.469 + \frac{6}{14} \times 0.5$$

$$\boxed{G(\text{Gender}) = 0.4822}$$

we can see

$$\boxed{G(\text{class}) < G(\text{Gender})}$$ thus, root node is class



this is how decision tree node is selected by calculating gini impurity for each node individually. If two or more features, then we need to repeat same steps after selecting each nodes

Note:-

① we should control growth of tree by 'min_sample_split' ~~In which way we will not~~, because a infinitely grown tree with depth (>100) also. there is a high chance at time of Prediction. it may go "overfitting"

Decision Tree for Regression:-

→ when performing regression with a decision tree, we try to divide the given values of x into distinct & non-overlapping regions.

Eg:- For a set of possible values $x_1, x_2, \ldots x_p$ we will try to divide them into 'J' distinct & non-overlapping regions $R_1, R_2, \ldots R_J$. For a given observation falling into the region "R_j", the Prediction is equal to the mean of the response value for each training observation in the Region.

(*) Regions are selected in a way to reduce residual sum.

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

$\hat{y}_{R_j}$ - mean of all the response vars. in the region 'j'.

Disadv of DT:-

① Small change in data can cause instability in the model because of greedy approach.	② Proba of overfitting is very high for DT	③ Takes more time to train a DT model.

Recursive Binary splitting (Greedy Approach) :-

→ As mentioned, we try to divide the x values into J regions, but is very expensive in terms of computational time to try to fit every set of x values into J regions.

→ Thus decision tree opts for a greedy approach in which nodes are divided into regions based on the given condition is called **greedy**, because it does the best split at a given step at that point of time **rather than** looking for splitting a step for a better tree in upcoming steps.

→ It decided threshold to divide others into different regions.

$$\sum_{x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

this process have a high chance of overfitting the training data as it is very complex

Tree Pruning :-

→ It is the method of trimming down a full tree to reduce the complexity & variance in the data. Just as we regularized linear regression, we can also regularize the decision tree model by adding a new term.

$$\sum_{m=1}^{|T|} \sum_{i: x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

where,

'T' is the subtree which is a subset of full tree T_0.

'α' is non-negative tuning parameter, which penalizes the MSE with an increase in tree len.

Post-Pruning :- (Backward pruning).

→ It is the process when DT is generated first & then the non-significant branches are removed. cross-validation set of data is used to check the effect pruning & tells whether expanding a node will make an improvement or not. If node is having improvement we continue else convert to leaf node.

Pre-Pruning :- (Forward Pruning)

→ It stops the non-significant branches from generating. It uses a condition to decide when should it terminate splitting of some of the branches prematurely as tree is generated by

* Different algorithms for Decision trees

① ID3 (Iterative Dichotomiser) :-

It is one of the algorithms used to construct decision tree for classification. It uses information gain as the criteria for finding the root node and splitting them.

It only accepts categorical attributes

② C4.5 :-

It is an extension of ID3 algorithm, & better than ID3 as it deals with both

Continuous & discrete values

Also used for classification purposes.

③ Classification & Regression Algo (CART) :- (Baucalli, Decision ma

It is the most popular Algo used for constructing decision trees. It uses 'gini' impurity as the default calculation for selecting root node, however one can use "entropy" for criteria as well. Both for regression & classification problem.

Gini is easier to calculate, which is why CART uses 'Gini' entropy complicated.