*Pruning of Decision Trees*

As Decision Tree ka sbhse bada disadvantages is that it has big problem of overfitting

So just to reduce this overfitting there are many settings.

How to Reduce Overfitting in Decision Trees?

We do pruning

1. Pre pruning
2. Post pruning

**Feature Importance**

The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as Gini Importance.

Decision Tree mai bydefault rehta hai

Feature\_importance\_

Attribute jo batadeta hai ki kaunsa features kitna important hai

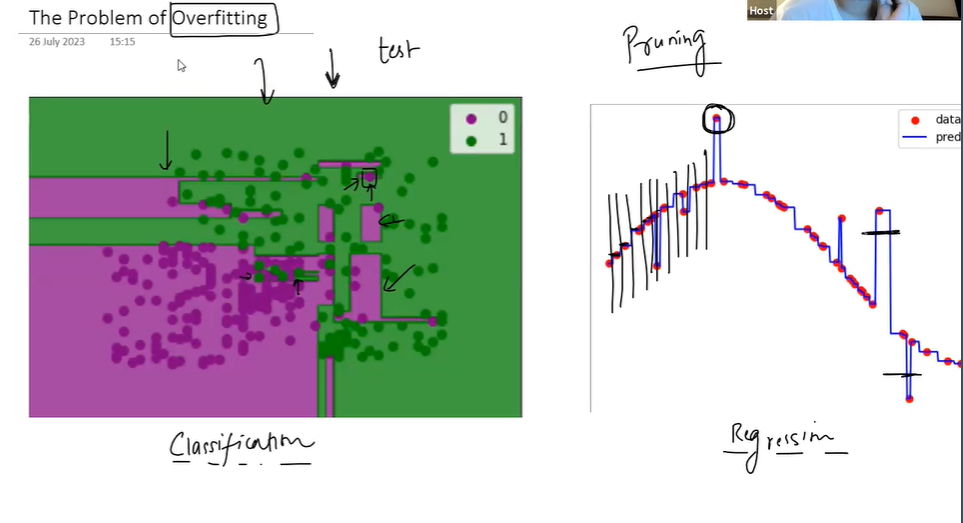
**Feature Importance sbhme same hai Random Forest, Boosting & Bagging**

**How to Calculate Node Importance ?**

1. **Saare node ka importance**
2. **Important Nodes are in which columns / Features**

*Why to Learn Prunning ? 🡪 Overfitting Problem ke vajese*

*The Problem of Overfitting*

**

**Why Overfitting Happens?**

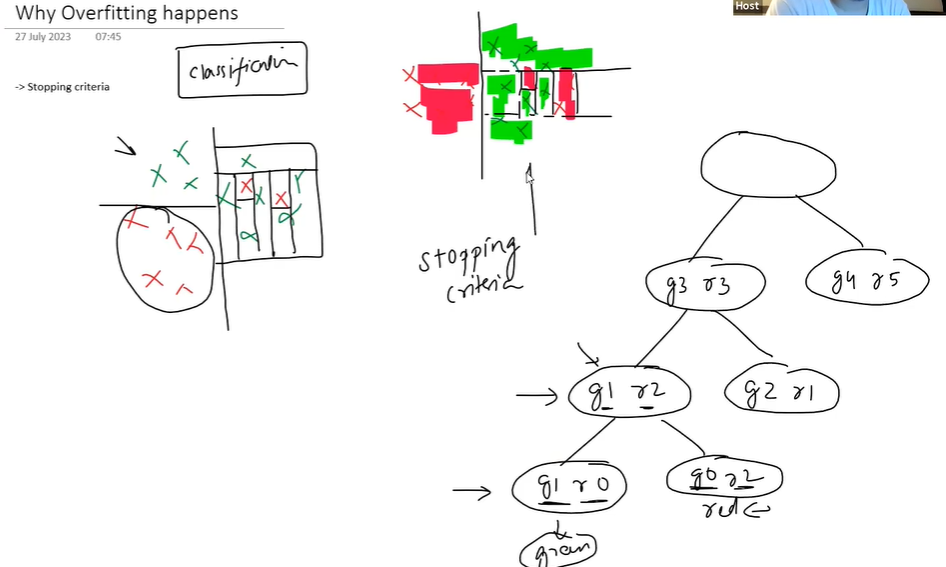
*Overfitting isliye horha hai because humare pass koi Stopping Criteria nahi hai*

*Ki yeah ese chalrha hai ki*

*Jbhtak usko 1 hie category ka data region main ahi milega tbhtak vo Aur Regions banate jaayega*

*So ese Model kaafi Complex hojayega*

***Classification Scenario:***



Regression Scenario :

Yaha bhi hum kaaatte hai ki Variance 0 hojaye

Yaha Humme tbh rukna hai jbh Y Axis ke Saare Points Same hojaaye

& Jbhtak Same nahi rahega tbhtak Kaatte jayega vo

So yeahi problem hai ki Stopping Criteria hie esi define hai

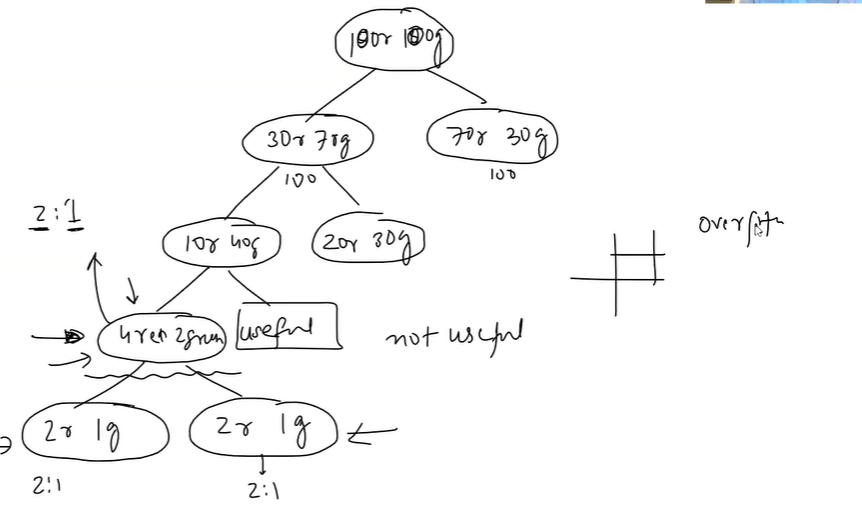
In case of Classification Jbhtak saari classes alag nahi hoti tbhtak kaatte jayega.

In case of Regression tbhtak kaatte raho jbhtak saare points ka Y value same nahi hota.

So Yeah Stopping Criteria kaam nahi karega,

Yeahi Problem hai Decision Tree ki iska Stopping Criteria ke vajese Overfitting issue aaraha hai

**Unnecessary Nodes**



Last child ka kuch mtlb nahi

Esehie Nodes ko Hum Remove karte hai , because yeahi esehie Nodes Cutt Create horahe hai

Overall Flow:

Decision Tree 🡪 Overfitting 🡪 Stopping Criteria 🡪 Deep Trees (with lot of Nodes ) 🡪 Lot of Nodes are Not Useful 🡪 Cut those Unnecessary Nodes 🡪 Tree Size Will Get Reduce 🡪 So Now Automatically Overfitting Reduce hoga.

**Pruning & it’s Types**

Pruning is a technique used in ML to Reduce the size of decision trees and to avoid Overfitting

Overfitting Happens when a model learns the training data too well ( ratta maarleta hai chutya )

Including its noise & outliers, which result in poor performance on test data or unseen data.

Decision Trees are Famous for Overfitting

Because they can potentially create very complex trees that perfectly classify training data but fail to generalize to new data. Pruning helps to reduce the complexity of Decision Tree.

There are 2 Main Types of Pruning:

1. Pre-pruning ( Early Stopping )

Tree banate waqt hie hum limited rkhte, faltu node nahi rkhte

1. Post-pruning ( Cost Complexity Pruning )

Pehle We Grow Tree , then We go backtrack & check ki kaunsa important hai & kaunsa nahi

Then we remove jo Important nahi hai

**Pre – pruning (Early Stopping )**

This method halts the tree construction early. It can be done in various ways. By setting a limit on the maximum depth of the tree, setting a limit on the minimum number of instances that must be in a node to allow a split, or stopping criteria when a split result in improvement of model’s accuracy below a certain threshold.

**Post-pruning (Cost Complexity Pruning):**

This method allows the tree to grow to its full size, then prunes it. Nodes are removed from the tree based on the error complexity trade-off. The basic idea is to replace a whole subtree by a leaf node, and assign the most common class in that subtree to the leaf node.

*Pre – pruning Session*

Pre-pruning, also known as early stopping, is a technique where the decision tree is pruned

during the learning process as soon as it's clear that further splits will not add significant value.

There are several strategies for pre-pruning:

**1. Maximum Depth:** One of the simplest forms of pre-pruning is to set a limit on the maximum

depth of the tree. Once the tree reaches the specified depth during training, no new nodes are

created. This strategy is simple to implement and can effectively prevent overfitting, but if the

maximum depth is set too low, the tree might be overly simplified and underfit the data.

(Hyper parameter tune krke (randomized Search & all ) vo krke hum Proper Maximum Depth nikalskte hai)

**2. Minimum Samples Split:** This is a condition where a node will only be split if the number of

samples in that node is above a certain threshold. If the number of samples is too small, then

the node is not split and becomes a leaf node instead. This can prevent overfitting by not

allowing the model to learn noise in the data.

(E.g Minimum split humne 10 rkha hai then Root Node ke 10 values pe hie vo Split hoga or else nahi hoga, ) Proper number iska bhi humme Hyper parameter tuning se millta hai

This controls on Parent Level

**3. Minimum Samples Leaf:** This condition requires that a split at a node must leave at least a

minimum number of training examples in each of the leaf nodes. Like the minimum samples

split, this strategy can prevent overfitting by not allowing the model to learn from noise in the

data.

( Suppose Minimum Sample leaf humne set kiya ha = 10 then jbhbhi Split hoo tou minimum 10 Samples hoo child mai )

This Controls on Child Level

**4. Maximum Leaf Nodes:** This strategy limits the total number of leaf nodes in the tree. The

tree stops growing when the number of leaf nodes equals the maximum number.

**5. Minimum Impurity Decrease:** This strategy allows a node to be split if the impurity decrease

of the split is above a certain threshold. Impurity measures how mixed the classes within a

node are. If the decrease is too small, the node becomes a leaf node.

( Sbhse Important hai yeah:

Splitting tbhi karo jbh Certain amount of Impurity decrease hoo For E.g

**6. Maximum Features:** This strategy considers only a subset of features for deciding a split at

each node. The number of features to consider can be defined and this helps in reducing

overfitting.

**Advantages of Pre-Pruning:**

1. Simplicity: Pre-pruning criteria such as maximum depth or minimum number of samples

per leaf are easy to understand and implement.

1. Computational Efficiency: By limiting the size of the tree, pre-pruning can substantially

reduce the computational cost of training and prediction.

**Disadvantages of Pre-Pruning:**

1. Risk of Underfitting: If the stopping criteria are too strict, pre-pruning can halt the growth

of the tree too early, leading to underfitting. The model may become overly simplified

and fail to capture important patterns in the data.

1. Requires Fine-Tuning: The pre-pruning parameters (like maximum depth or minimum

samples per leaf) often require careful tuning to find the right balance between

underfitting and overfitting.

1. Short Sightedness - Can prune good nodes if they come after a bad node.