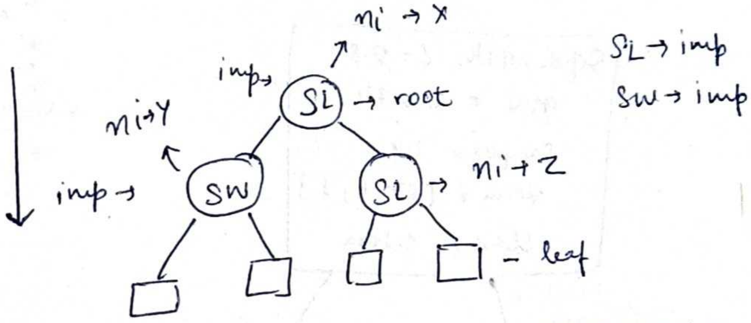
Feature Importance

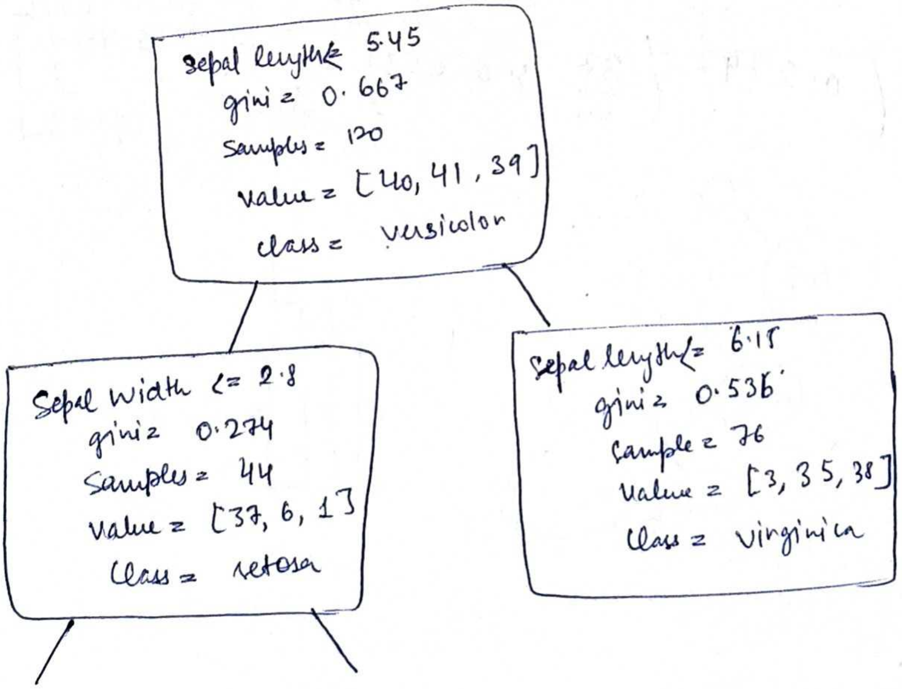
The important of a feature is computed as the (normalized) total reduction of the criterian bought by that feature. It is also known as the gini importance.



$$\boxed{fi_k = \frac{\sum_{j \in \text{node splitter feature } k} n_i}{\sum_{i = \text{all node}} n_i}}$$

Overall imp of SL → $\boxed{SL = \frac{X+Z}{X+Y+Z}}$

Overall imp of SW $\boxed{SW = \frac{Y}{X+Y+Z}}$



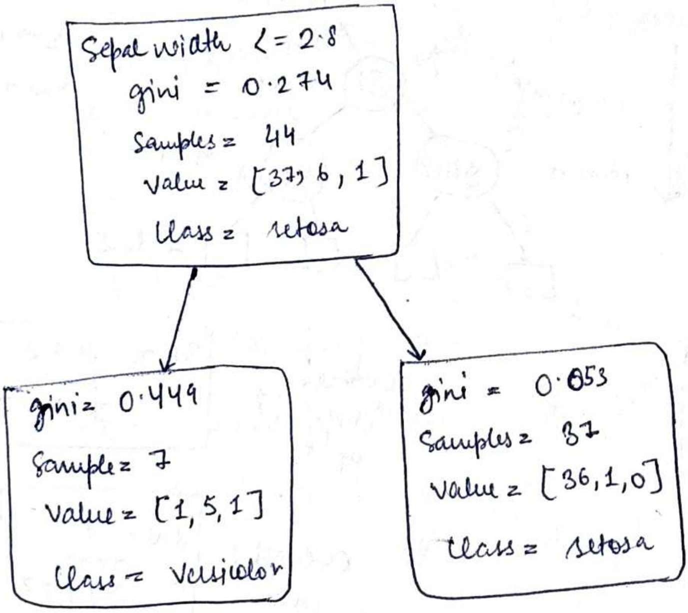
sepal length <= 5.45
gini = 0.667
samples = 120
value = [40, 41, 39]
class = versicolor

Sepal width <= 2.8
gini = 0.274
samples = 44
value = [37, 6, 1]
Class = setosa

sepal length <= 6.15
gini = 0.536
sample = 76
value = [3, 35, 38]
Class = virginica

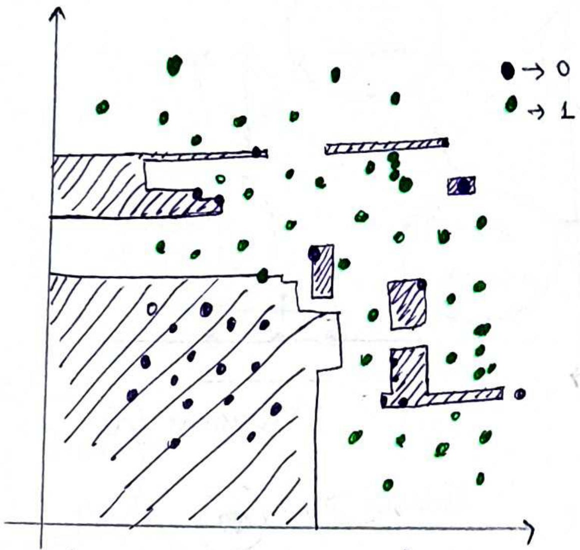
$$\boxed{n_i = \frac{N-t}{N}\left[\text{impurity} - \left(\frac{N-t-r}{N-t} \times \text{right -impurity}\right) - \left(\frac{N-t-l}{N-t} \times \text{leftimpurity}\right)\right]}$$

$$= \frac{120}{120}\left[0.667 - \left(\frac{76}{120} \times 0.536\right) - \left(\frac{44}{120} \times 0.27\right)\right]$$

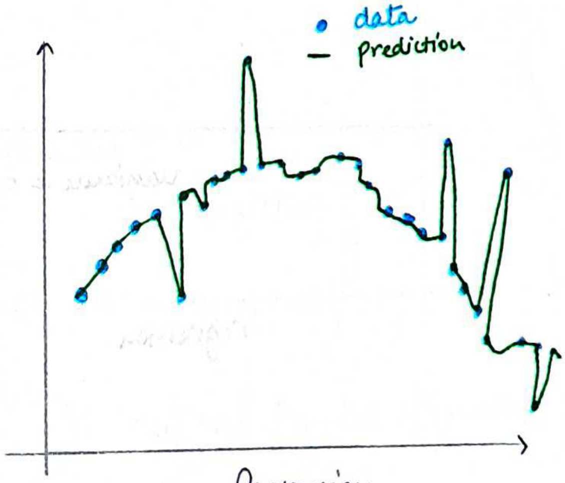


$$= \frac{44}{120}\left[0.274 - \left(\frac{37}{44} \times 0.053\right) - \left(\frac{7}{44} \times 0.449\right)\right]$$

The Problem of Overfitting



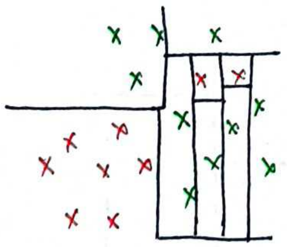
Classification

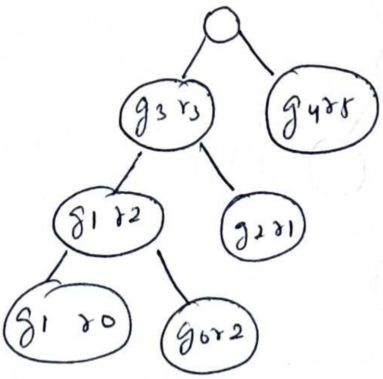


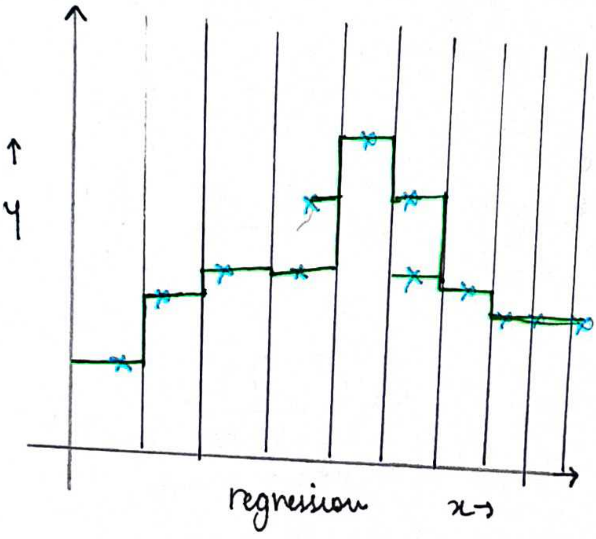
Regression

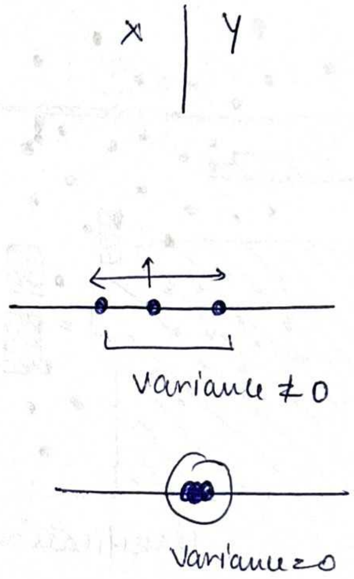
Why Overfitting happen?

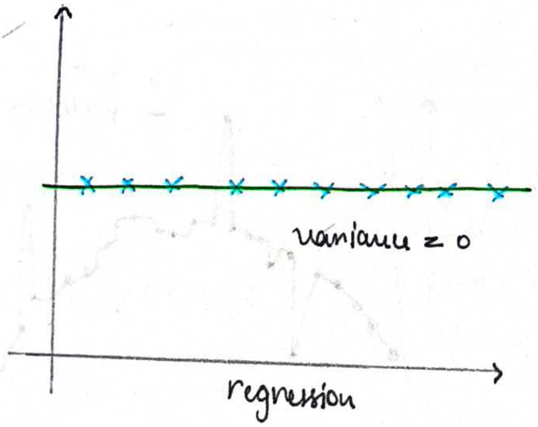
- Stopping criteria



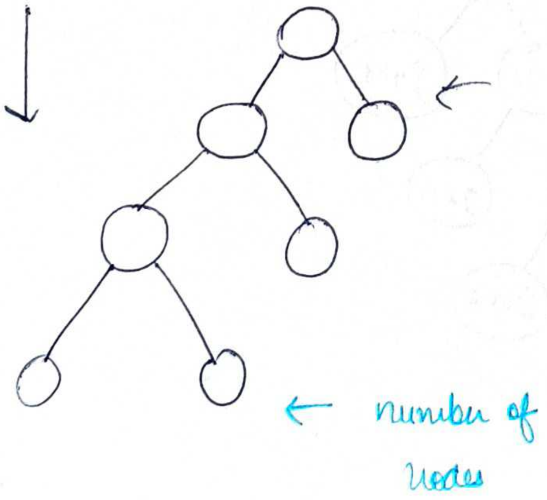


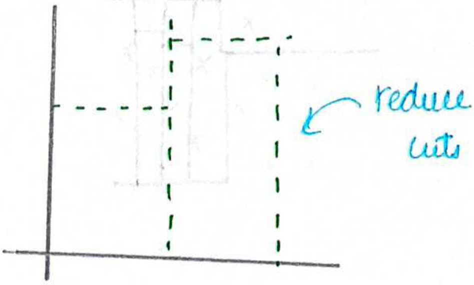


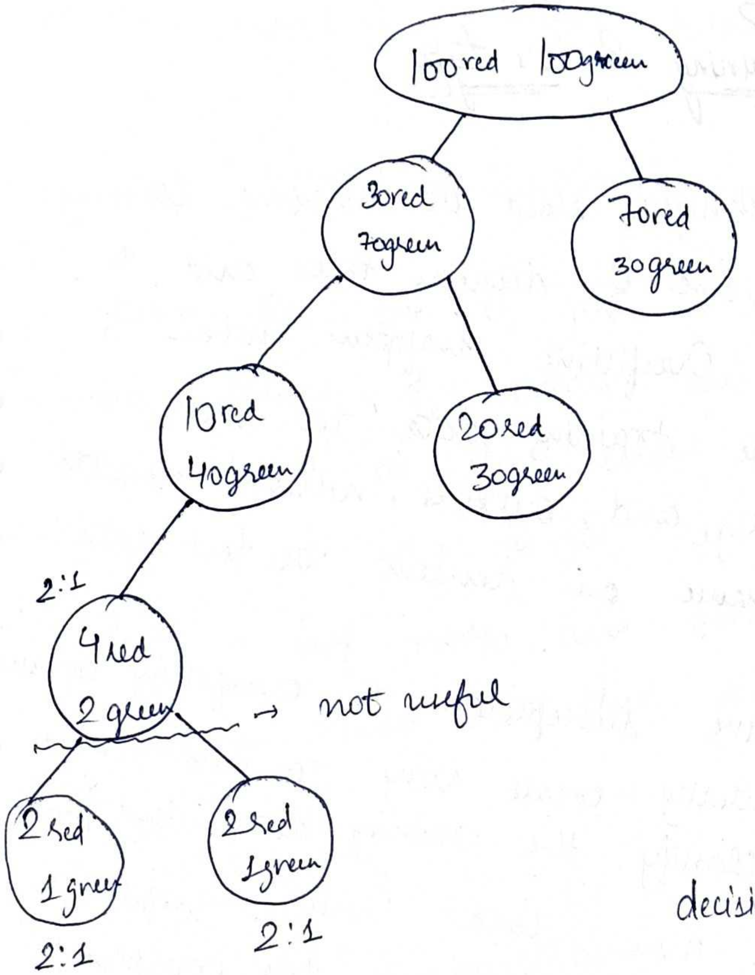




Unnecessary Nodes







decision tree
↳ overfitting
stopping
↓
deep tree (a lot of nodes)
(not useful) → cut those node
reduce overfitting ← tree size reduce

Pruning & it's types

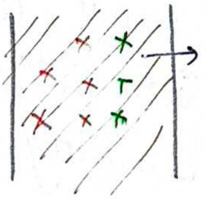
Pruning is a technique used in machine learning to reduce the size of decision trees and to avoid overfittiy. Overfitting happens when a model learns the training data too well, including its noise and outliers, which results in poor performance on unseen or test data.

Decision trees are susceptible to overfitting because they can potentially create very complex trees that perfectly classify the training data but fail to generalize to new data. Pruning helps to solve this issue by reducing the complexity of the decision tree, thereby improving its predictive power on unseen data.

There are two main types of pruning: pre-pruning and post pruning

1: **Pre-pruning (Early stopping)**: This methods 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 maximum number of instances that must be in a node to allow a split, or stopping when a split results in the improvement of the model's accuracy below a certain threshold.

2. **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.



some points are green but count in red area, so new point will be also red not green not cut line for green.

Pre-purning

Pre-pruning

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 stategie for pre-pruning:

1. **Maximum Depth**: One of the simplest forms of prepruning is 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.

2. **Minimum Samples Split**: This is a conditional where a node will only be split if the numbers 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.

3. **Minimum Samples leaf**: This condition requires that a split at a node must leave at least a minimum number of training example 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.

(4) **Maximum Leaf Nodes** : This strategy limits the total numbe of leaf nodes in the tree. The tree stops growing when the numbers of leaf nodes equals the maximum numbe.

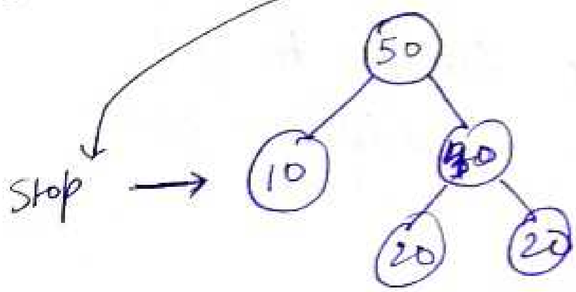
(8) **Maximum Impurity Decrease**: This strategy allows a node to be split if the impurity decrease of the split is above a certain threshold, impurity measure how mixed classes with a node are. If the decrease is too small, the node becomes a leaf node.

(6) **Maximum Features** : This strategy considers only a subset of feature for deciding a split at each node. The number of feature to consider can be defined and this help in reducing overfitting.

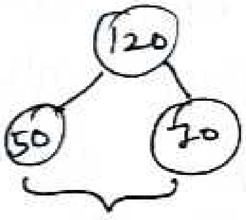
(1) → Max_depth = None → overfit

Max - depth = 4 → underfit

(2) Minimum - sample - split = 10



(3) Minimum sample leaf = 50



leaf node contain 50

Advantages of Pre-Pruning

1. **Simplicity**: Pre-purning criteria such as maximum depth or minimum number of samples per leaf are easy to understand and implement.

2. **Computational Efficiency**: By limiting the size of the tree, pre-prunig can substantially reduce the computational cost of trainig 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 pattern in the data.

2. **Requires Fine-Tuning**: The pre-pruning parameter (like maximum depth or minimum sample per leaf) Often require careful tuning to find the right balance between underfittig and Overfittig.

3. Short Sightedness: Can prune good nodes if they came after a bad node.