Pruning : Kyon chahiye

1. Height Restriction of Tree
2. Ignoring Unnecessary branches
3. Decrease Computation
4. Less Time
5. Avoid Overfitting

Pruning of Trees in XGBoost is purely done on gain

Let us Assume a Threshold gain = γ [gamma].

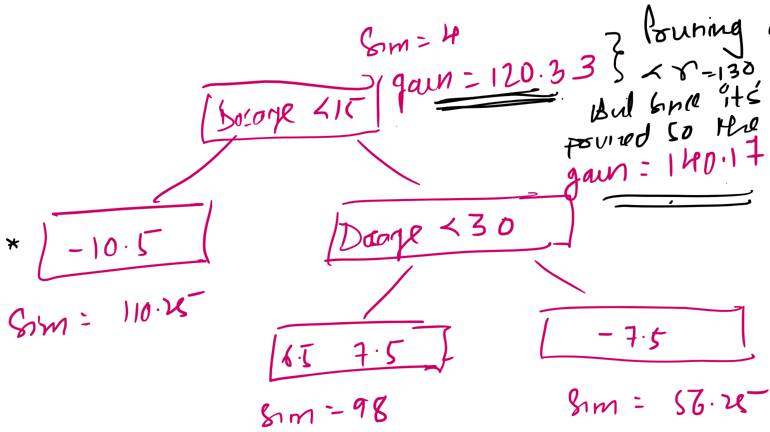
[Rule for Pruning :

* If at branch $\text{gain} - \gamma < 0$ then Prune and go above

* If $\text{gain} - \gamma \geq 0$ then do not Prune
* If child is not pruned then parent also will not be pruned]

$\therefore$ Tree $\lambda = 0$

$\boxed{\text{Let } \gamma = 130}$



This is candidate for Pruning as gain = 120.33 < γ = 130

But since its child is not pruned so the parent is also not pruned

not a candidate for pruning $\because 140.17 > \gamma = 130$

* Note

If γ is sufficiently large then it may result into complete Tree pruning till root → Extreme Pruning

* Regardless of value of λ and γ let us assume the Tree as :-

* Regardless of value of λ and γ let us assume the Tree as →

* O/p of Each leaf $= \dfrac{\Sigma \text{ Residual}}{\text{No of Residual} + \lambda}$

$\lambda = 0$

Sim = 4
gain = 120.33

$\boxed{\text{Dosage} < 15}$

$\boxed{-10.5}$
Sim = 110.25

o/p = $\frac{-10.5}{1}$ = -10.5

$\boxed{\text{Dosage} < 30}$ gain = 140.17

$\boxed{6.5, 7.5}$
Sim = 98

o/p = $\frac{6.5 + 7.5}{2 + 0}$ =

$\boxed{-7.5}$
Sim = 56.25

o/p = $\frac{-7.5}{1 + 0} = -7.5$

New Prediction = 0.5 + 0.3 *

(In XGBoost the learning rate is known as Eta (ϵ) and default value = 0.3)

Sim = 4
gain = 120.33

$\boxed{\text{Dosage} < 15}$

$\boxed{-10.5}$
Sim = 110.25
o/p = -10.5

$\boxed{\text{Dosage} < 30}$ gain = 140.17

$\boxed{6.5, 7.5}$
Sim = 98
o/p = 7

$\boxed{-7.5}$
Sim = 56.25
o/p = -7.5

Now Prediction for Dosage = 10

= 0.5 + 0.3 * (-10.5) = -2.65

for Dosage = 10   Initial prediction = 0.5
New Prediction = -2.65

* Find New Prediction for all the other 3 Dosages.
* Using the New Residual Build a New Prediction that will give Smaller Residual
* Keep Building the Tree until the max No of Trees Constructed or residual is Significantly Small.

(T_1 T_2 T_3 T_4)

residual is Significantly Small.

Let Consider for above Case there are 4 Tree (T_1, T_2, T_3, T_4)

* $$\boxed{\text{final Prediction} = 0.5 + 0.3 * T_1 + 0.3 * T_2 + 0.3 * T_3 + 0.3 * T_4}$$

XGBoost (Xtreme Gradient Boost)

* Implementation of Gradient Boost Decision Tree.
* Decision Tree is constructed in Sequential form.
* Weights plays Important Role in XGBoost
* Weights are assigned to all independent variables which are fed into Decision Tree that predict result.
* Weights of variables predicted wrong by previous Tree is increased and that variable are fed in Next Decision Tree.
* These Individual predictors/classifiers can ensemble to give Strong and precise model.
* XGBoost is faster than Gradient Boost
* There is stop criterion for Tree splitting in XGBoost
  - → XGBoost uses Max Depth parameter that it starts pruning the Tree backward.
  - → This pruning improves computational performance and helps to overcome problem of "overfitting"

XG Boost for Classification →

1. Convert the observed value into Probabilities (1 & 0).
2. Decide on initial prediction (Ex 0.5)
3. Find Residual (observed − Initial Prediction).
4. Create Leaf with All Residuals.
5. Calculate Similarity = $\dfrac{\Sigma (\text{Residual})^2}{\Sigma \text{Prev Prob} \times (1 - \text{Previous Prob}) + \lambda}$. [$\lambda = 0$ or $\lambda = 1$]

   ↑ (Initial predi)

6. Now to decide where to Split

   [Take number of Averages cases and Calculate gain for each.
   Select the split with gain]

7. [XGB have threshold for Minimum No of Residuals in the Leaf

* To find Minimum No of Residuals in each Leaf Cover is calculated.

$$\text{Cover} = \Sigma (\text{Previous Prob}) \times (1 - \text{Prev Prob})$$

[In Regression ⇒ Cover ⇒ 1 (default)]]

8. Construct a Tree.
9. Calculate o/p value at Each Leaf = $\dfrac{\Sigma (\text{Residual})}{\text{No of Residual} + \lambda}$.
10. Calculate New Predicted Value = Initial Pred + Learning Rate × Tree1

(10) Calculate New Predicted Value = Initial Prob + Learning Rate × Tree1

(11) Repeat the process untill max No of Trees are reached - or Residual becomes Insignificant.

(12) [Note > New Prediction will be log(odd).

To Convert into Probability = $\frac{1}{1+e^{-\log(odd)}}$.

=> This gives New Probability]

VImp   Bagging & Boosting

(1) Bagging (Boostrapping Aggregate)

- ↳ Weak learners organized in Parallel.
- ↳ Independent weak learners
- ↳ Used to reduce Variance

Steps:
(1) Multiple Subsets are created from Original Dataset Selecting observations with replacement.
(2) A base model (weak) is created for each of subset
(3) The weak model runs in parallel and are Independent to each other.
(4) The final predictions are determined by combining the predictions from all the models.

Ex: Random Forest.

Boosting > Basic >
If a data point is incorrectly predicted by first model then the next model will correct the prediction thus giving better result for immediate next model.

Steps
(1) A Subset is Created from Original Dataset
(2) Initially all the data points are given equal weights.
(3) A base model (Initial prediction) is created on dataset
(4) This base model is used to make prediction (Initial) on whole dataset
(5) [illegible] Calculated for each Sample.

whole dataset

(5) Residual (Initial) are Calculated for each Sample.

(6) Now the observation that are Incorrectly classified are given <u>higher</u> weight

(7) Another model is created based on predictions of previous model.

[This model tries correct the errors from the previous model]

(8) Similarly we will have multiple models (each based on previous model)

(9) The final model (Strong Learner) is weighted mean of all the model

<u>Ex</u>

ADABOOST < Regression, Classification

XGBOOST < Regression, Classification

Q) Which one is Better?

Depends on
(1) Data
(2) Simulation
(3) Circumstances

* If Individual Single model has High Bias.

then Bagging will not improve Bias

However Boosting will improve the Bias.

* If Single Model Overfits then Bagging is the Best option

as Boosting will not help in Overfitting.

[Note

↑ Bias → Boosting is helpful

↑ Variance → Bagging is helpful.]

Q) Similarities of Bagging & Boosting

1. Both are Ensemble methods (we need weak learners)
2. Both generate Several Training data sets by random Sampling.
3. Both make final Decisions by averaging the N learners (Regression) or Majority Voting (classification).
4. Both are good at Reducing Variance and providing Higher Stability.

Q) Difference of Bagging & Boosting

Bagging	Boosting
1. Bagging is Simplest way of Combining predictions that belongs to Same type	1. Boosting is way of Combining predictions that belongs to diff types.
2. Bagging primarily helps in reducing Variance	2. Boosting primarily helps in reducing Bias
3. Each model (Tree) recieves equal weighte	3. Boosting models are weighted according to their performance (Amount of say).
4. Each model is built Independently.	4. New model is built influenced by performance of previously built model.
5. Different Training Dataset are randomly built with replacement from	5. Every new subset contains Element that were Misclassified

are randomly built with replacement from Entire Training Dataset

(6) It is better for Overfitting.

(7) Ex: Random Forest

Element that were Misclassified by previous model.

(6) It is better for Underfitting

(7) Ex. Gradient Boost
ADA Boost
XG Boost