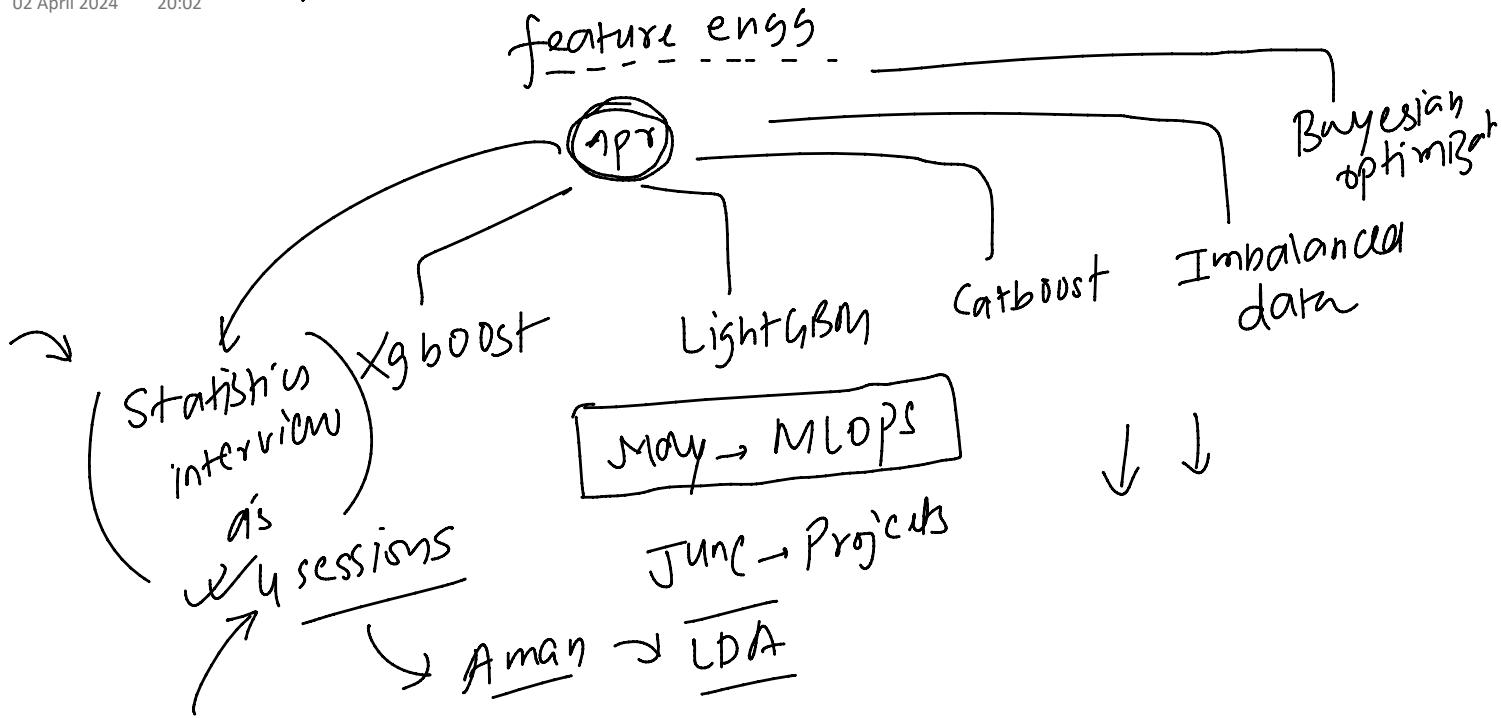


# Plan of Attack

02 April 2024 20:02



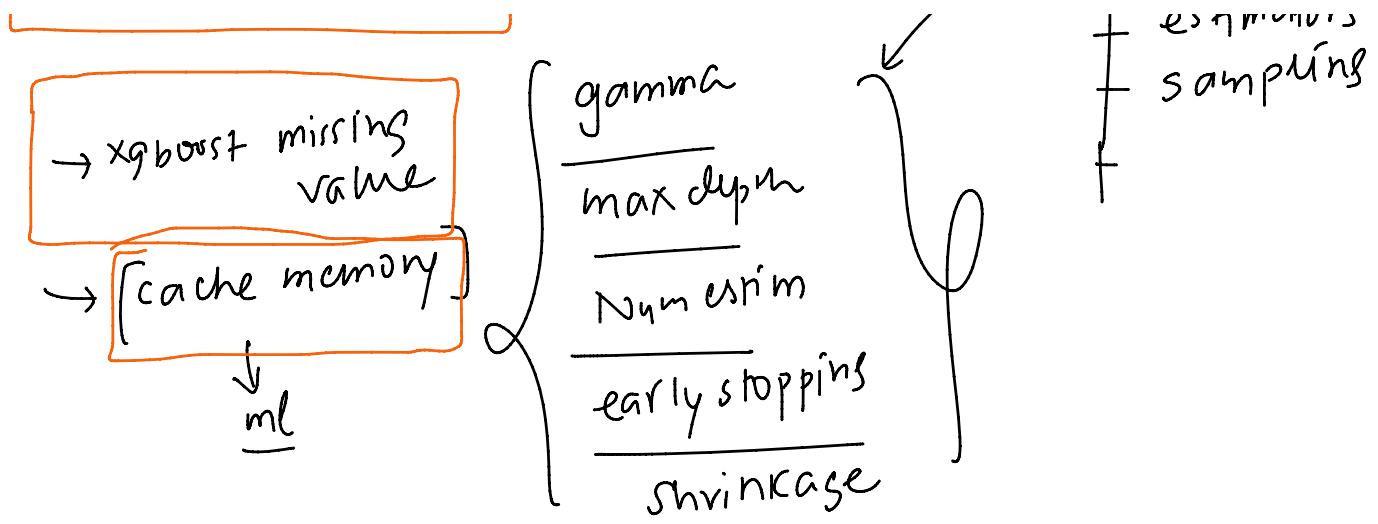
- Lambda
- Subsample
- Col subsample
- Min child weight

Gamma

Viz tools

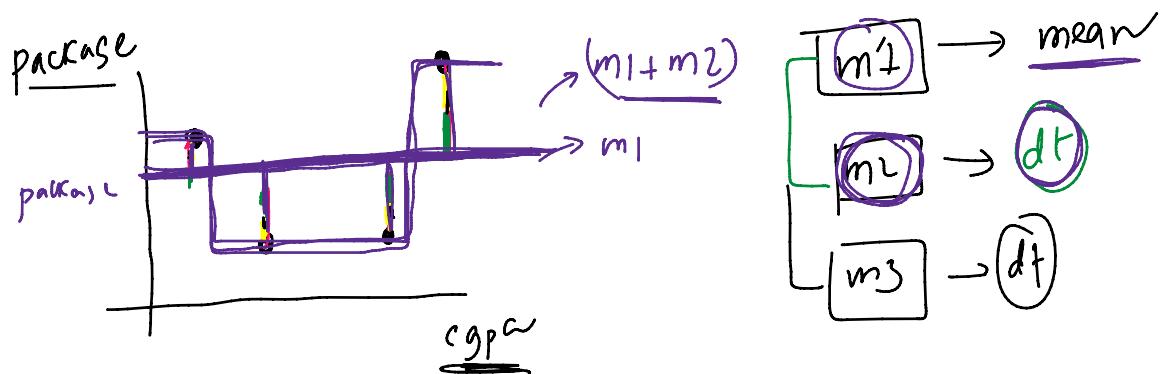
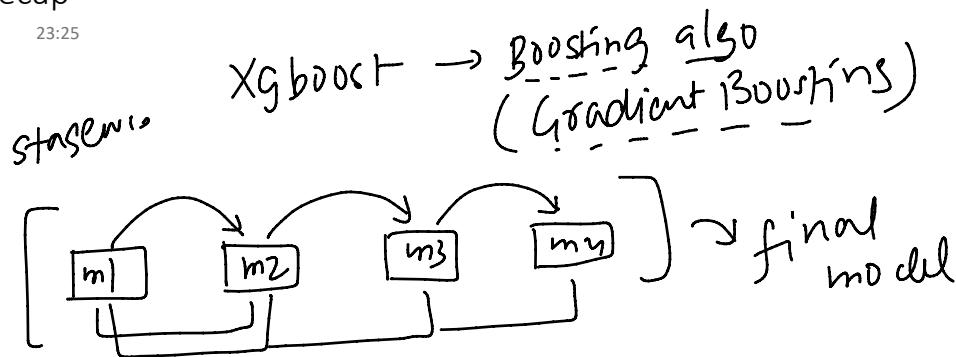
code

- + Regularization
  - + simpler trees
  - + estimators
  - + sampling



## Quick Recap

30 March 2024 23:25



loss func

$L(y_i, \hat{y}_i) \leftarrow$  minimizable

$[f_t(x_i)]$  functions

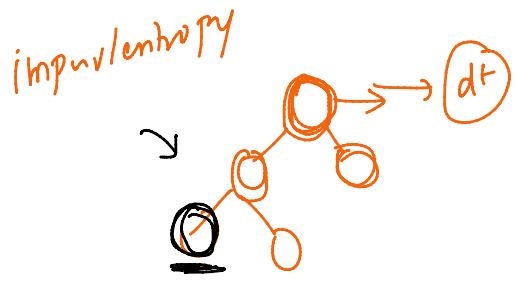
$f_t(x_i)$  function parameters

$$L = L(y_i, \hat{y}_i) + \sum_{t=1}^T f_t(x_i)$$

regularization

incompleteness

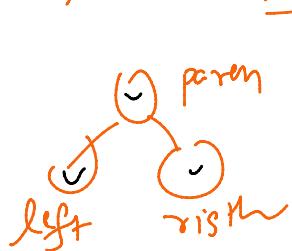
IL



input entropy

$$\text{Similarity score} = \frac{(\text{sum of residuals})^2}{\text{num of residuals} + \lambda}$$

gain



$$\text{gain} = (ss_{\text{left}} + ss_{\text{right}} - ss_{\text{parent}}) - \gamma$$

regression

$$\text{output} = \frac{\text{sum of residuals}}{\#\text{ of residuals} + \lambda}$$

$$[\square + \square + \square + \square + \square]$$

# Ways to reduce overfitting in XGBoost

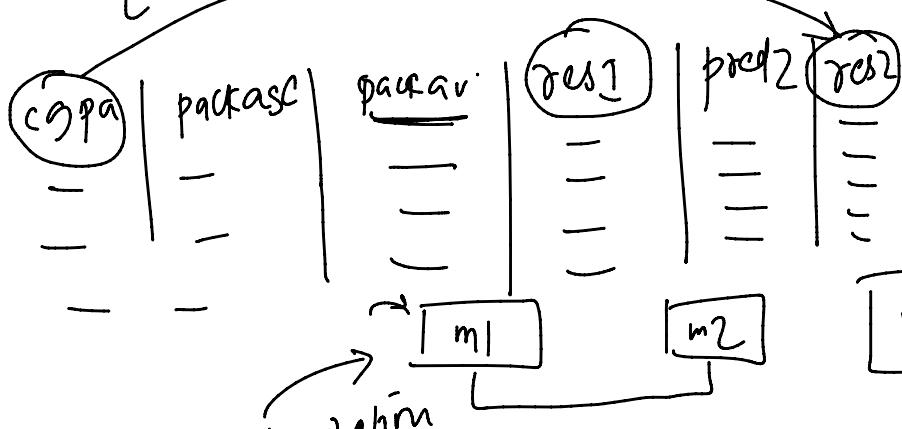
30 March 2024 23:57

## XGBoost Regularization

$$L^t = L(y_i, q_i) + \text{Reg} + \frac{1}{2} \sum_{i=1}^t w_i^2$$

## Regularization

way to prevent overfitting by simplifying the model



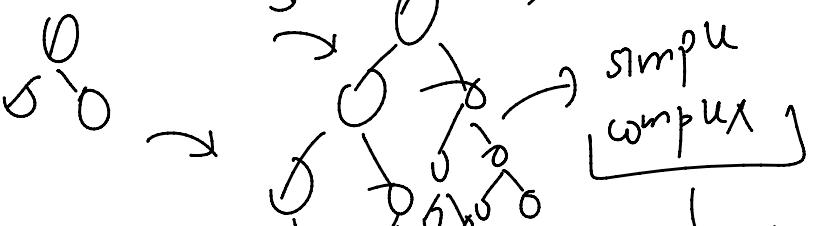
+  $m_1 + \dots + m_t$

↳ number of dts

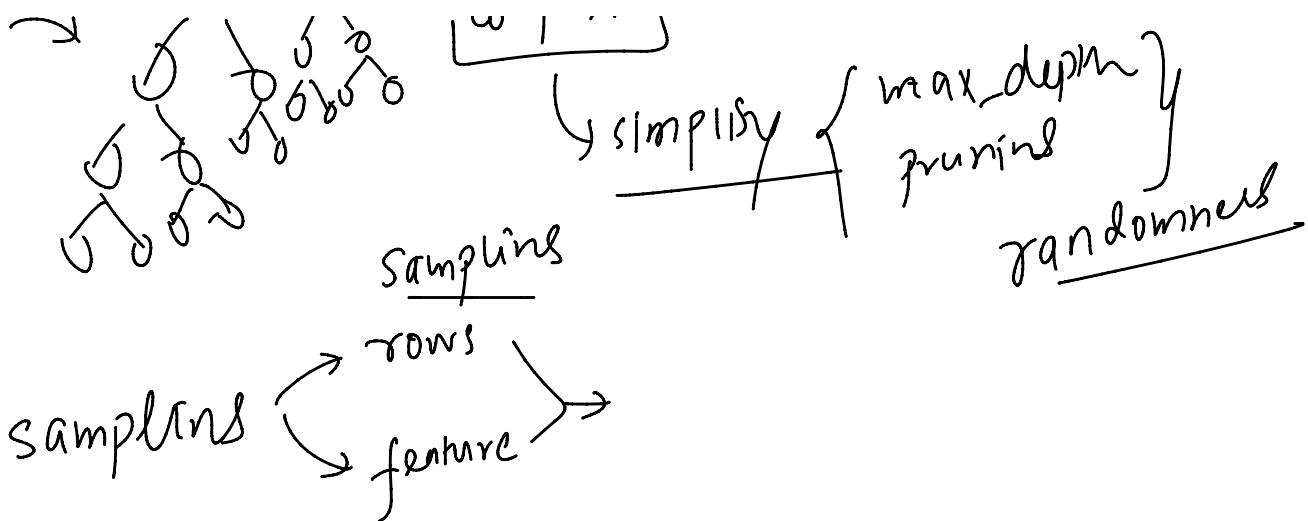
simpler trees

1) reduce the # estimators

2) simple trees

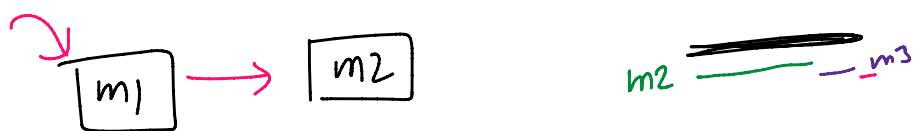
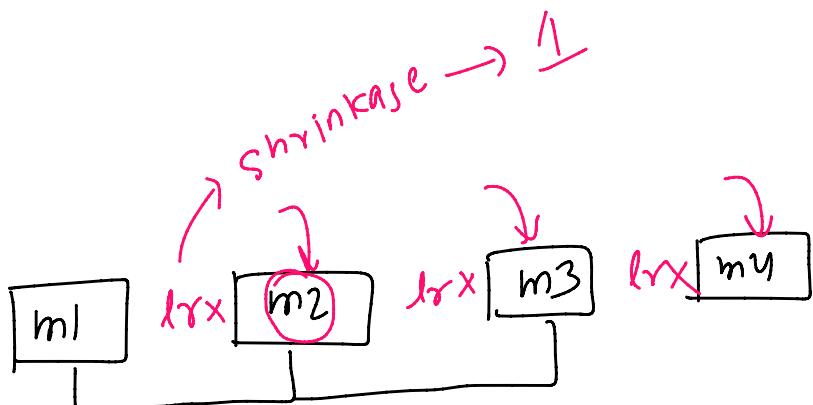


{ max\_depth }



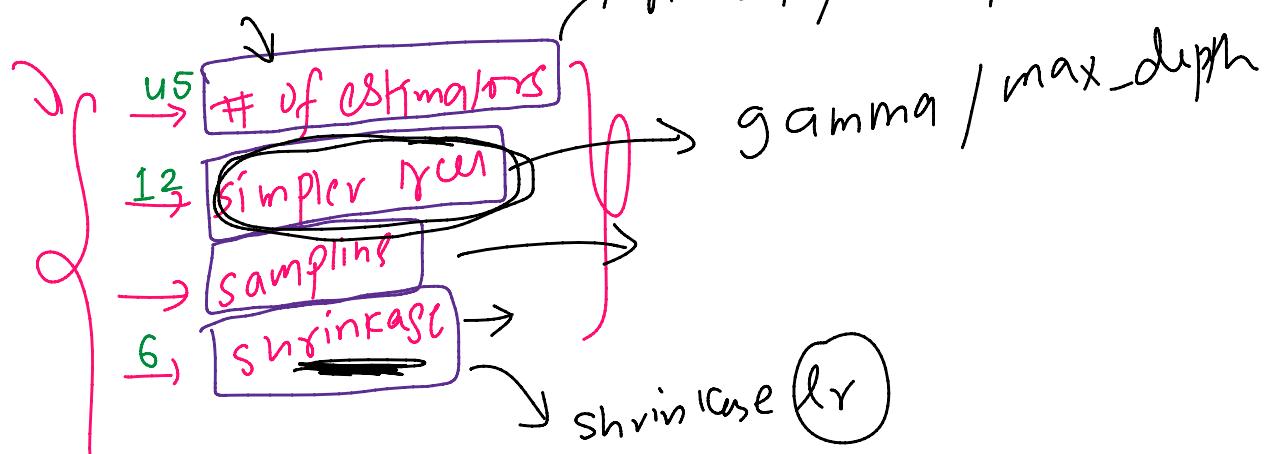
4)

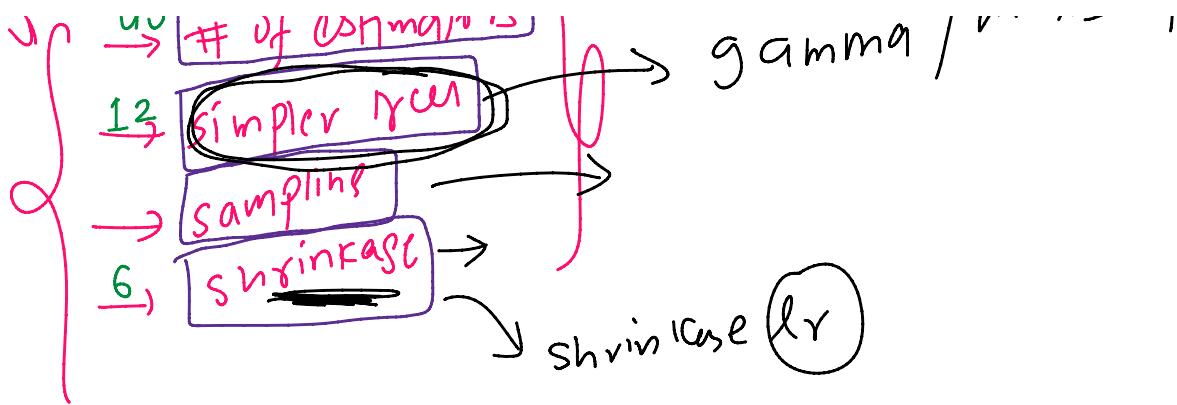
0.5



m1      m2      m3      m4      gradual nature

→ n-est / early stopping





gamma →

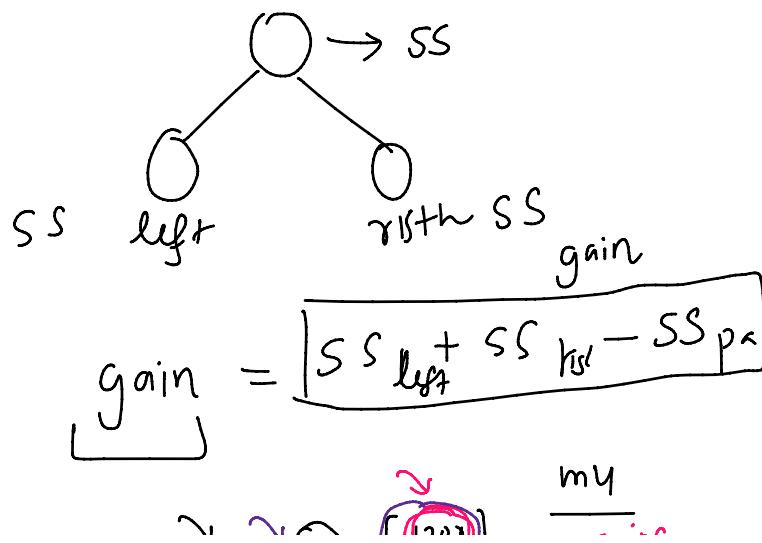
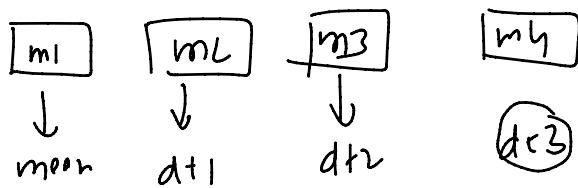
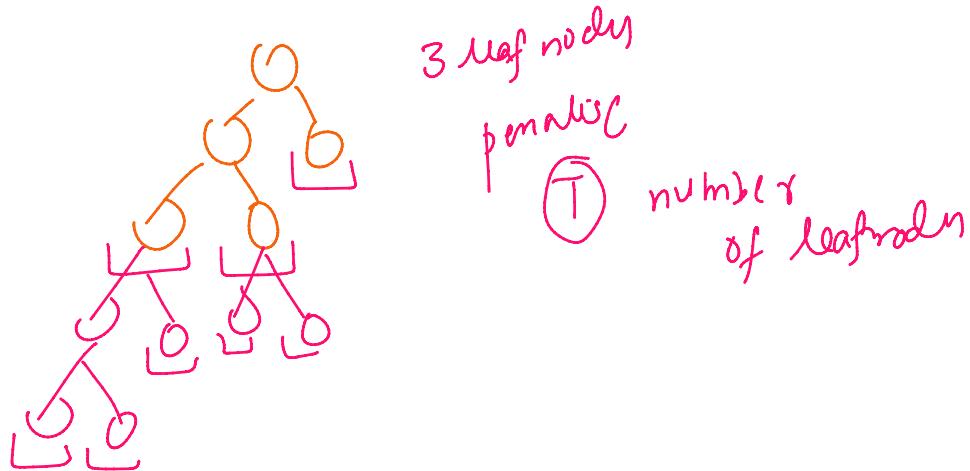
$\lambda \rightarrow \underline{\text{simpler ren}}$

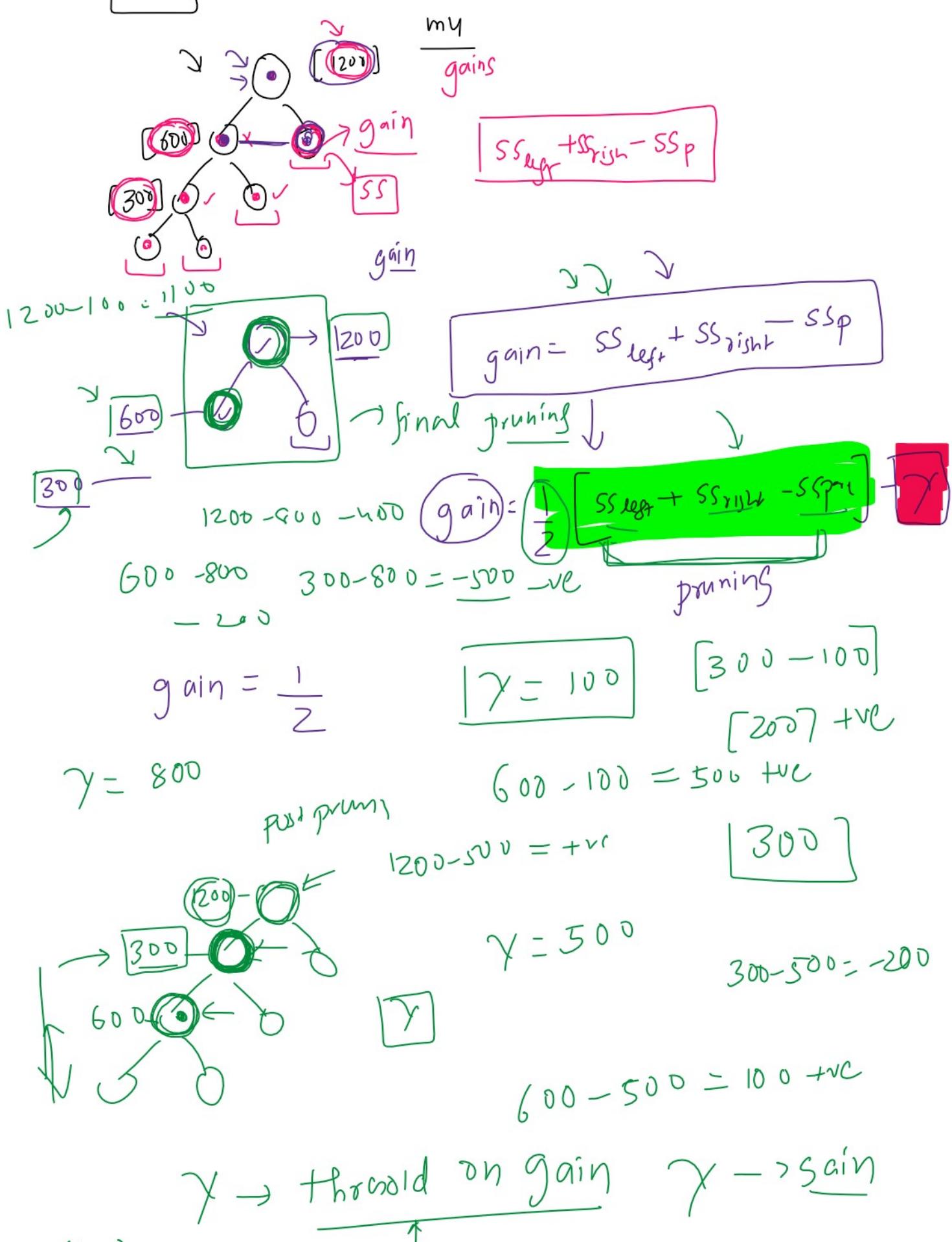
# 1. Gamma ( $\gamma$ )

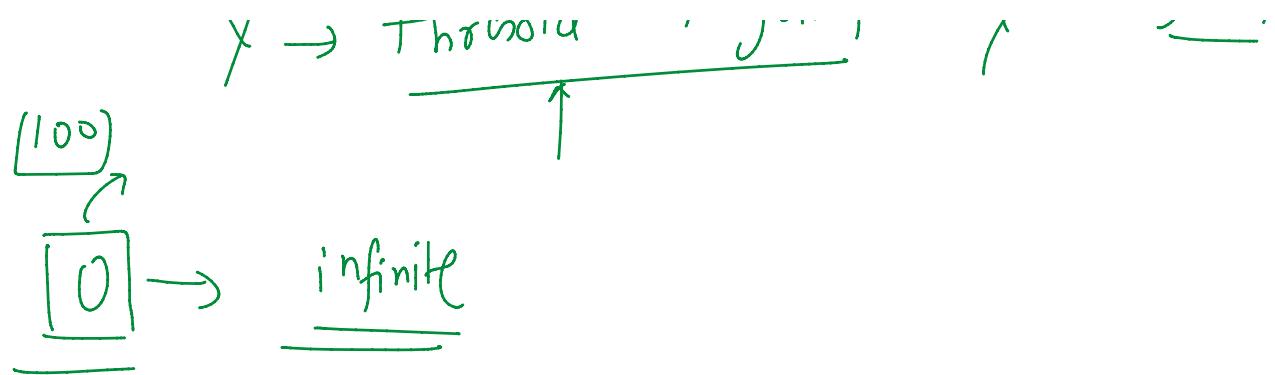
30 March 2024 23:26

$$L = L(y_i, \hat{y}_i) + \boxed{\gamma T} + \frac{1}{2} \lambda \sum_{i=1}^T w_i^2$$

post pruning  
iter  $t$   $f_t(x_i)$



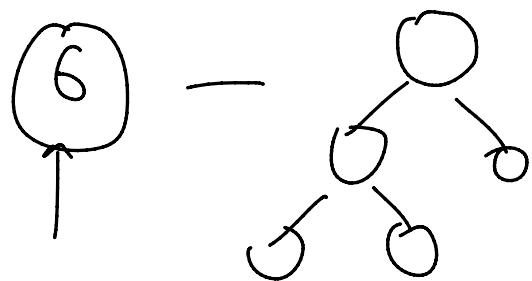




$\begin{cases} \text{high } \gamma \rightarrow \text{underfitting} \\ \text{low } \gamma \rightarrow \text{overfitting} \end{cases}$

## 2. Max Depth

30 March 2024 23:27



computation

$$\text{max\_depth} = 6$$

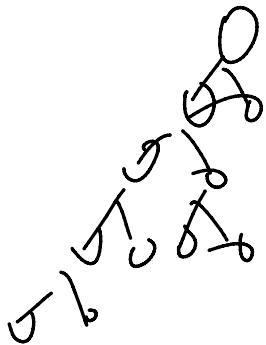
$$\boxed{\text{max\_depth} = 2}$$

$\left. \begin{array}{l} \text{pre} \\ \text{prunings} \end{array} \right\}$

→ why do need max-  
[gamma]

$$\rightarrow \text{max\_depth} = \underline{\text{None}}$$

$$\boxed{1000} \text{ - est'mal}^L$$

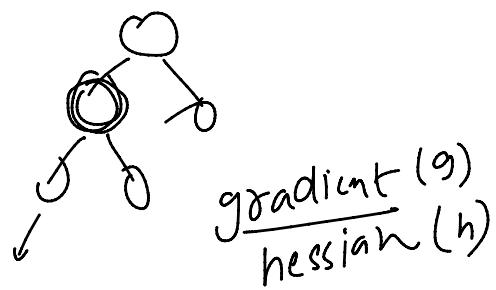


3. Min Child Weight  $\rightarrow$  pruning  $\rightarrow$  simpler tree

30 March 2024 23:27

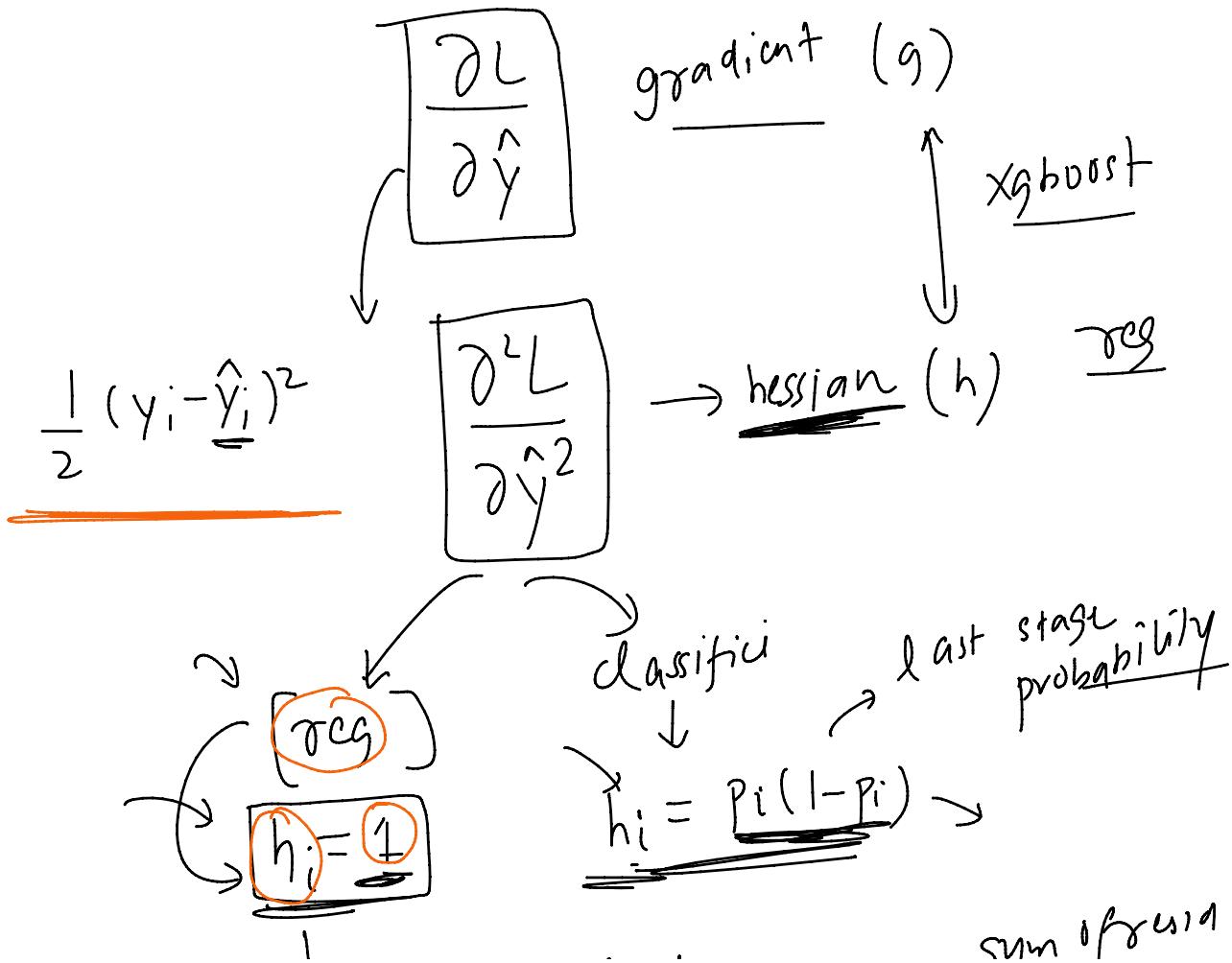
tricky  $\rightarrow$  complex

Cover  $\rightarrow$  boosting



Cover define  $\downarrow$   
sum of hessians

gradient  $\rightarrow$  derivative  
 $\int \rightarrow \hat{y}$



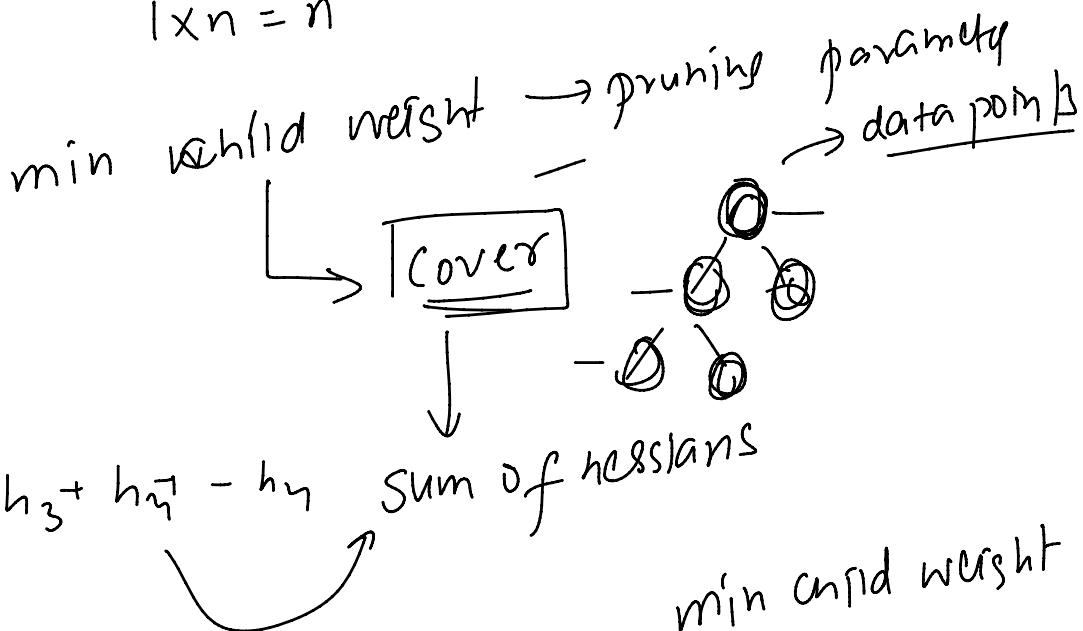


$$W = \frac{\text{sum of residual}}{\boxed{\# \text{ of residual}} + \lambda}$$

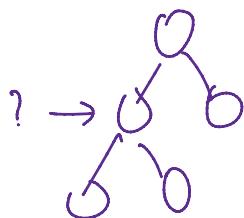
$$W = \frac{\text{sum of residual}}{p(1-p) + \lambda}$$

sum of hessians

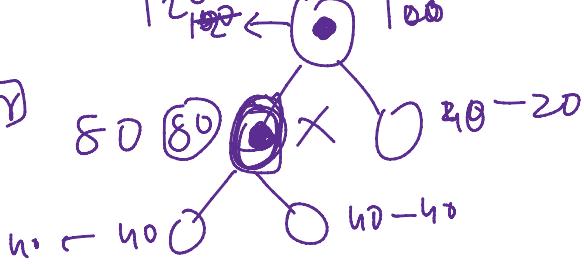
$$1 \times n = n$$



Reg →  $h_i = 1$



[Cover]



$\gamma_{\text{reg}}$  → cover is nothing but number of datapoints

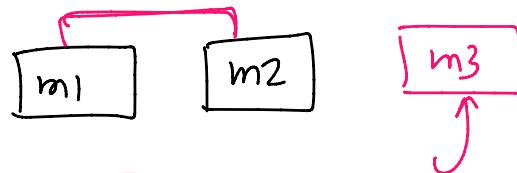
$\min_{\text{child}} \text{weight} = 100 \rightarrow$   
 ~~$\min_{\text{child}} \text{weight}$~~  → cover > min child weight

$\gamma_{\text{reg}} \text{ calc}$        $\gamma_{\text{cls}}$   
 ~~$\min_{\text{child}} \text{weight}$~~  →  $\min_{\text{sample-split}}$

$\min_{\text{child wt}} \uparrow$  overfitting ↓

$\gamma_{\text{cls}}$

$$h_i = 1$$



### Classification

$$h_i = p_i(1-p_i)$$

↓ prev stage prob

cga	placement	pred	$\gamma_{\text{cls}}$	$\gamma_{\text{pred}}$	$\gamma_{\text{cls}}$
			0.5		
8	1	0.5	0.5	0.7	0.3
7	0	0.5	-0.5	0.3	-0.3
4	0	0.5	-0.5	0.2	-0.2
5	1	0.5	0.5	0.8	0.2

$$h_i = p_i(1-p_i) = 0.7(1-0.7) = 0.21$$

$$0.3(1-0.3) = 0.21$$

$$0.2(1-0.2) = 0.16$$

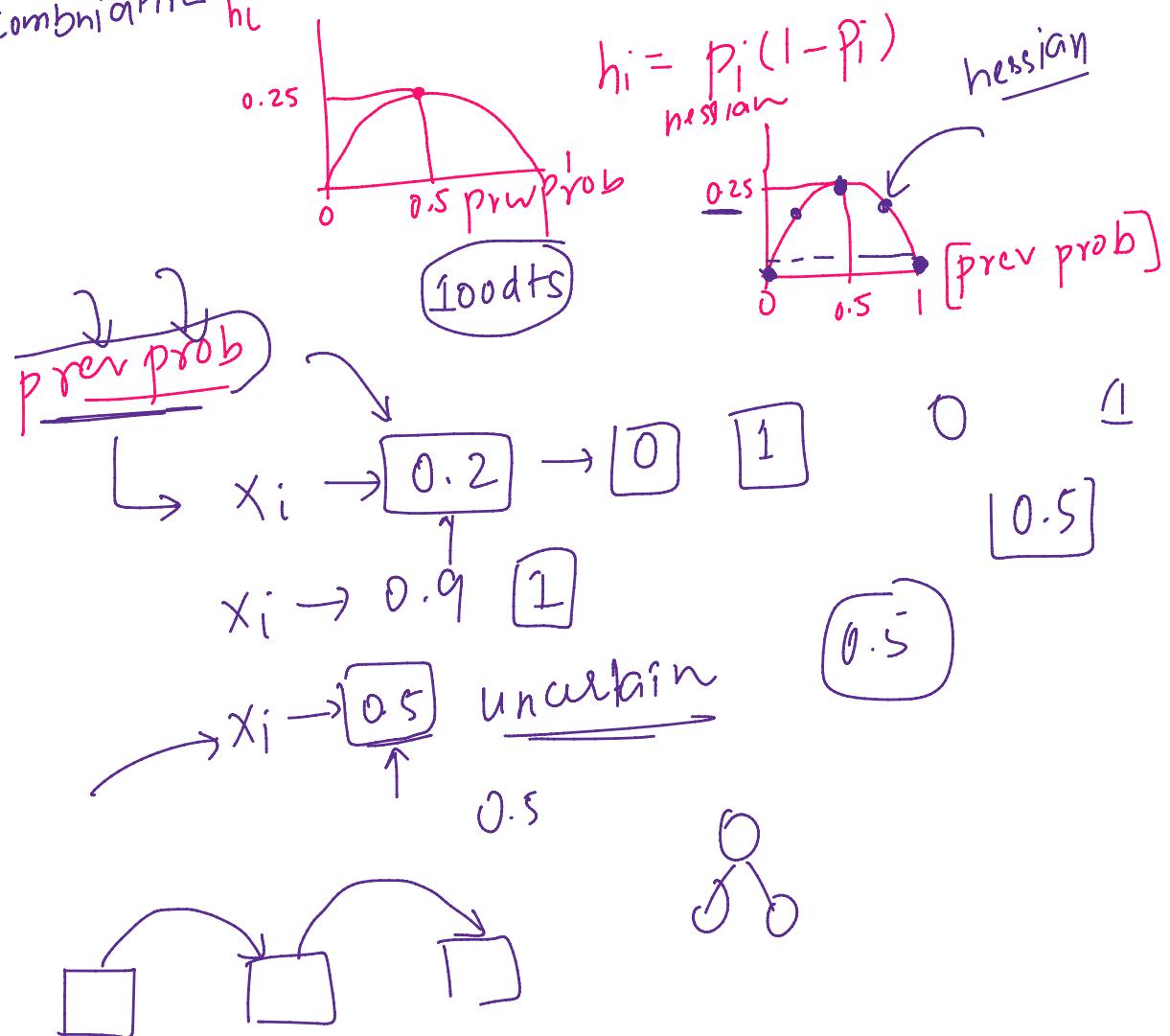
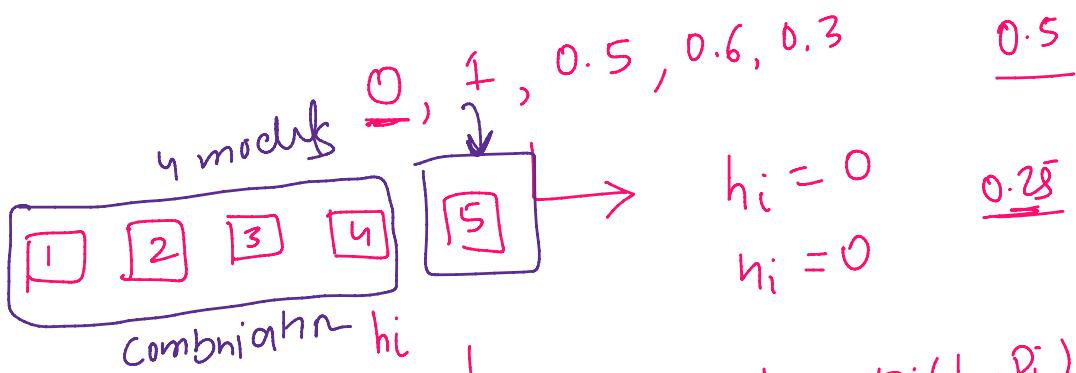
4

$$0.2(1-0.2) = 0.16$$

J  
0.16

$h_i \rightarrow \max (0.25)$

$1(1-1)$



hessian gives a quantitative value  
in each data point

hessian "j"  
to each data point

↓  
explain its importance

$0.25 \uparrow \rightarrow$  next dev

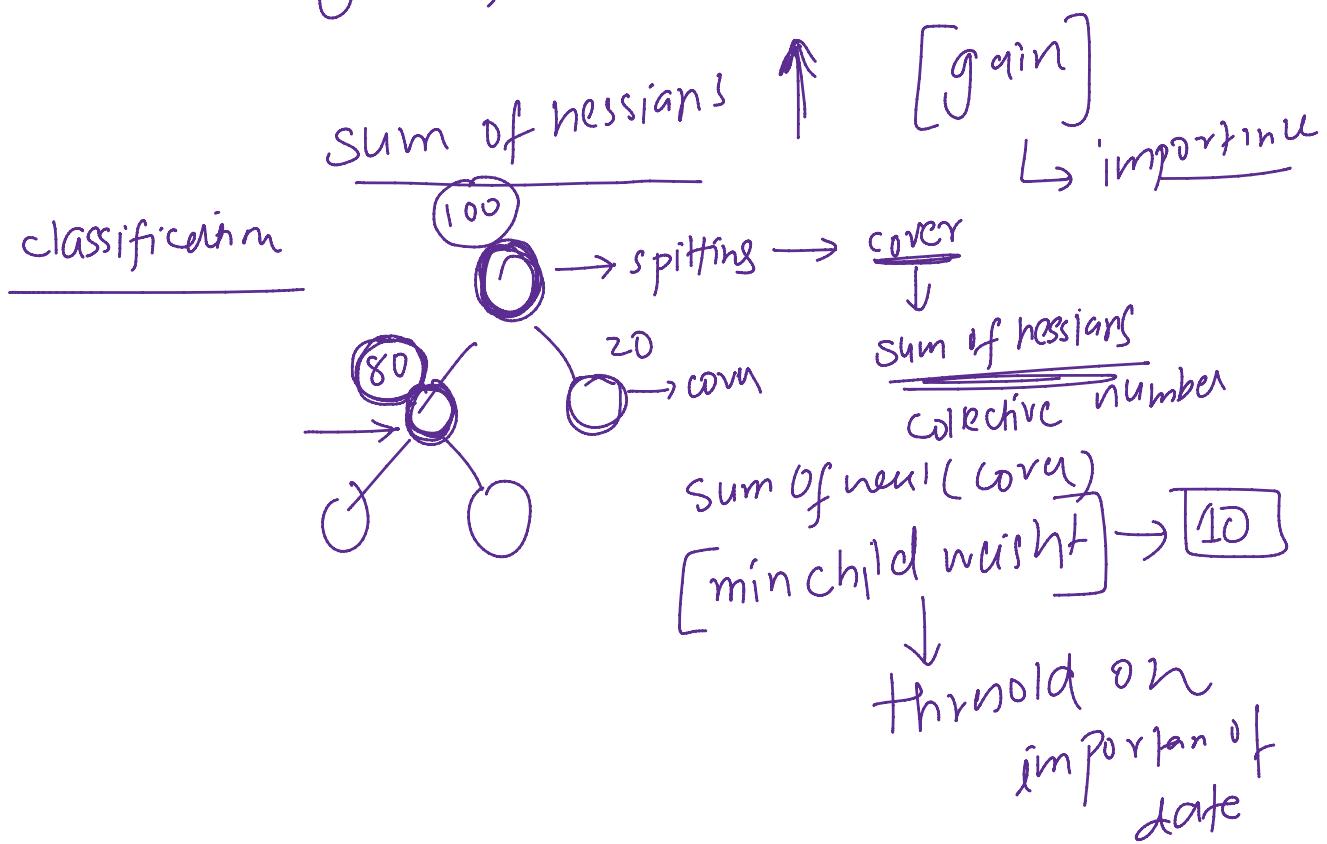
$0 \rightarrow$  not so imp

$$\underline{t_1} \rightarrow 0.25$$

$$\underline{t_2} \rightarrow 0.1$$

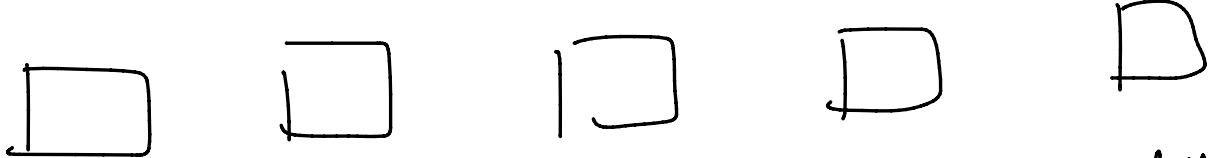
$$\underline{t_3} \rightarrow 0$$

$$\underline{t_4} \rightarrow 0$$



## 4. Num Estimators

02 April 2024 17:57



$n_{\text{estimator}} \uparrow (100) \rightarrow \text{overfitting}$   
 $n_{\text{estimator}} \downarrow (1) \rightarrow \text{underfitting}$

## 5. Early Stopping

02 April 2024 08:28

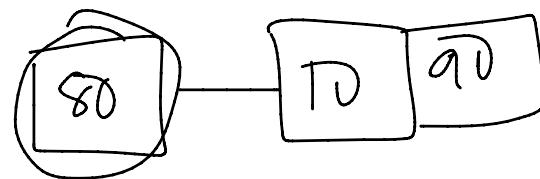
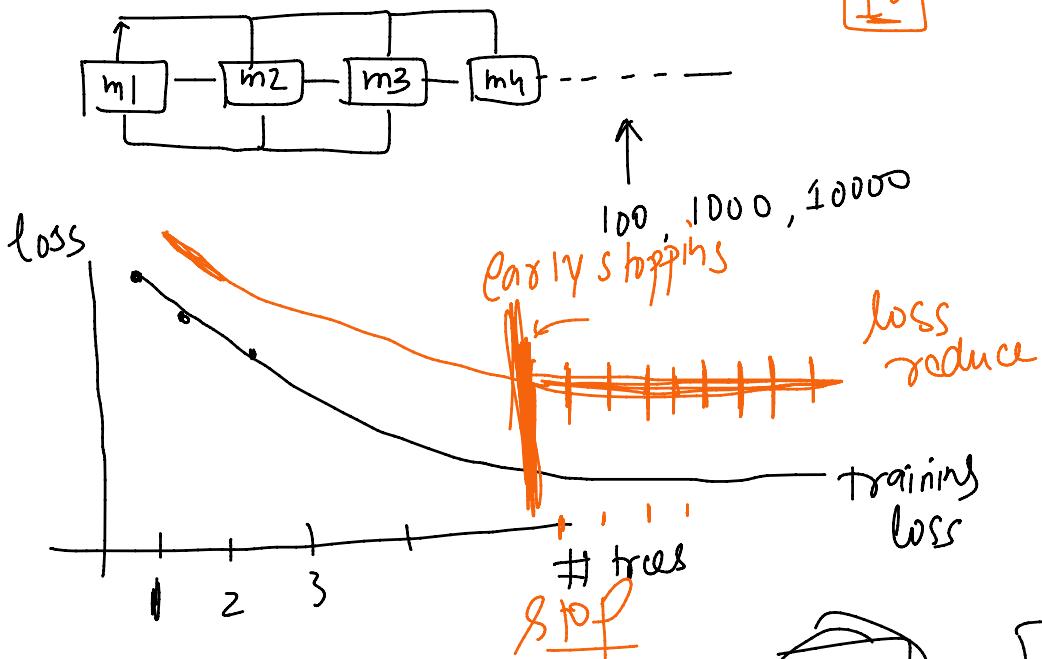
XG boost

early stoppings

Train loss

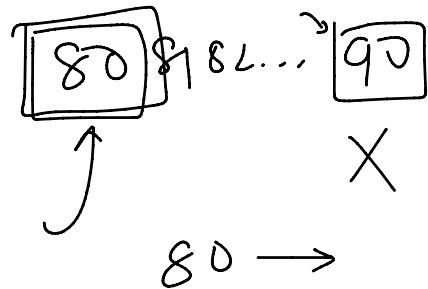
test loss

10 →



$$80 - 10 = 70 \text{ model}$$

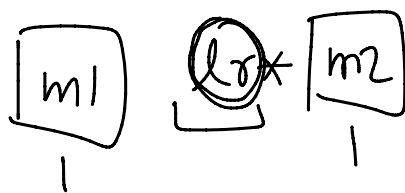
patience  $\boxed{80}$  model  $\times$



## 6. Shrinkage

30 March 2024 23:26

$[5.23]$



best possible  
number

1 2 3 4 5 6  $\underline{[5.23]}$

$\underline{0.1}$  1. 1.1 1.2 1.3 ... 2 2. 1. 2.7

$\underline{0.01}$  5.1 5.2 5.3

1 1.01 1.02 ... 1.23 ... 2. 2.01, ..

3 .. 5.01 ...  $\underline{[5.23]}$

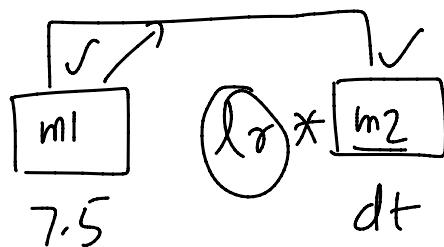
$l_r \rightarrow \uparrow$  ①  $5.23$   
 $l_r \rightarrow \downarrow$   $0.00001$  5 6

$0.3$  ← 0.0  
0.3

$$7.5 + 5 = 12.5$$

$$7.5 + 0.3 \times 5 = 9$$

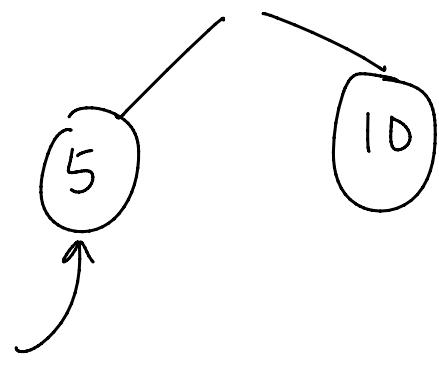
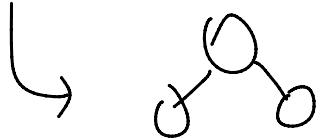
$$7.5 + 0.5$$



capa	package	$p_{red}^1$	$\sigma_{es(1)}$	$p_{red}$
	-	$\sim$	0.5	

if  $csp_n < 7.5$

c9pa	package	pre <sup>++</sup>	v <sup>-</sup>	l <sup>+</sup>
7 8	7 8	7.5 7.5	0.5 -0.5	1



## 7. Lambda (and Alpha)

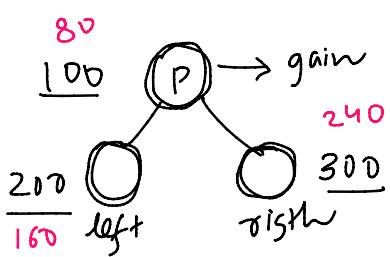
30 March 2024 23:26

$$L = L(y_i, \hat{y}) + \sqrt{f(x_i)} \quad \downarrow$$

$$\text{L} = \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

$$\lambda = 0 \quad \boxed{\lambda = 10}$$

similarity score



$$\text{similarity score} = \frac{(\text{sum of residuals})^2}{\# \text{residuals} + \lambda} \rightarrow \text{gain}$$

$$\boxed{SS_{\text{left}} + SS_{\text{right}} - SS_P}$$

$\lambda \uparrow \underline{SS} \downarrow$

gain

$\lambda \uparrow \downarrow$   
underfitting overfitting

$$\boxed{\text{output } w^* = \frac{\text{sum of residual}}{\# \text{residuals} + \lambda}}$$

$$200 + 300 - 100 = \boxed{400} \quad \lambda = 0$$

$\boxed{\lambda \uparrow}$  pruning ( $\uparrow$ )

$$160 + 240 - 80 = 320 \quad \lambda = 10$$

$\downarrow$  pruning

$$\boxed{Y} = \boxed{100} - \text{fixed}$$

$$\boxed{\text{gain}} - \gamma \xrightarrow{100} \text{node prune}$$

$\gamma \uparrow \rightarrow \boxed{\lambda} \uparrow$   $\lambda \uparrow \text{gain} (\downarrow) \gamma \text{ constant}$

$\text{prune} \rightarrow \text{node} \rightarrow \text{gain } \underline{\text{re}}$

$$\lambda = 0 \quad \lambda = 10$$

output  $\downarrow$





output:  $\frac{\text{sum of residual}}{\# \text{residual} + \lambda}$

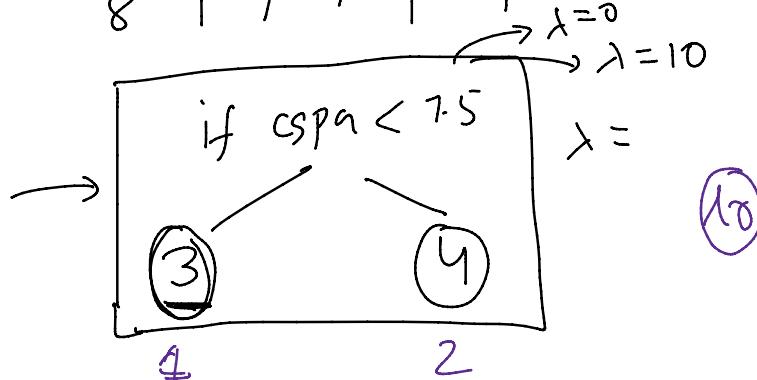
cspn	packag	pred1	res1	10
6	6	7	-1	
7	7	7	6	
8	8	7	1	

$$m_1 + m_2 \\ 7 + 8$$

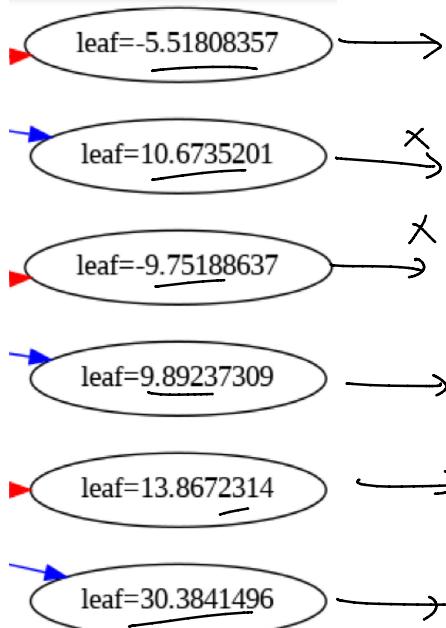
8

$\lambda \rightarrow \text{shrinkage}$

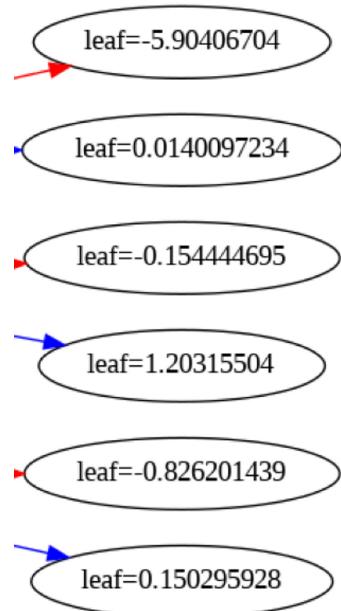
(3)  $\rightarrow$  2



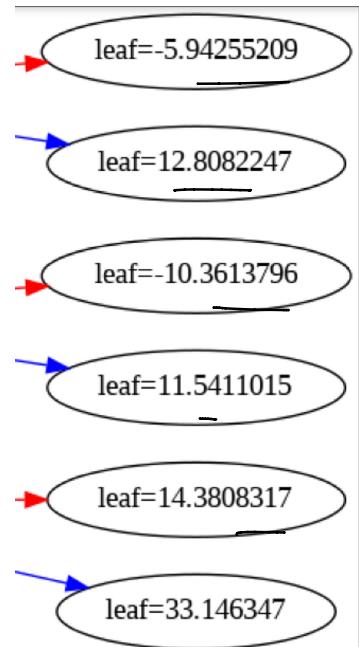
$\lambda = 1$



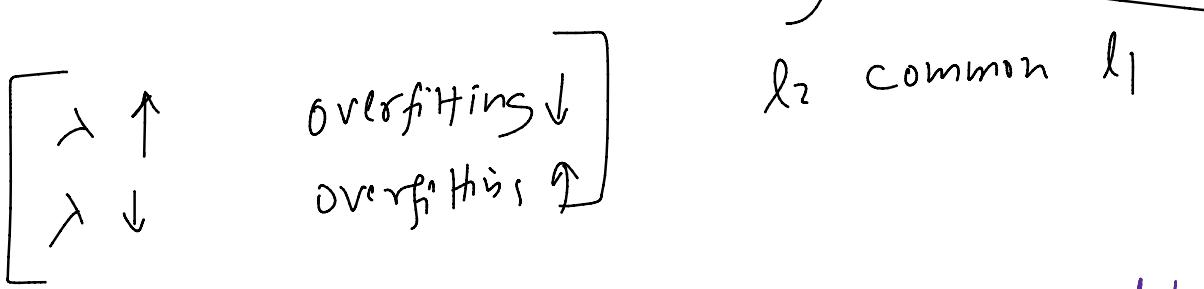
$\lambda = 100$



yes / shrinkage  
 $\lambda = 0$

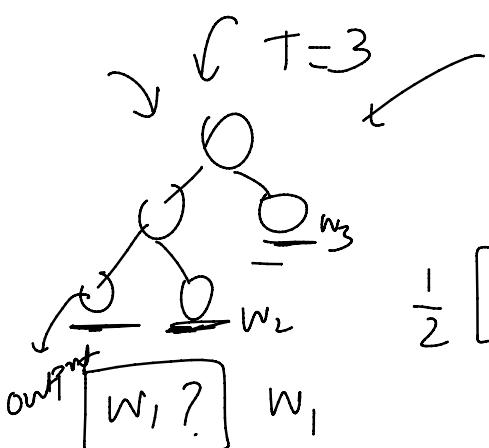


l<sub>2</sub> common l<sub>1</sub>



$$L^{(t)} = L(y_i, \hat{y}_i) + \mathcal{J}(f_t(x_i)) \quad \lambda \uparrow \text{ smaller}$$

$$\frac{1}{2} \sum_{i=1}^T \left[ \underline{y}_i^T + \frac{1}{2} \sum_{j=1}^d \underline{w_j}^2 \right] \quad \lambda \uparrow \text{ weight}$$



$$\frac{1}{2} \left[ \underline{w_1}^2 + \underline{w_2}^2 + \underline{w_3}^2 \right]$$

reg-lambda

reg-alpha

$$\gamma T + \frac{1}{2} \sum_{i=1}^T \|w_i\| \quad \rightarrow \ell_1$$

$$\frac{1}{2} \alpha [ |w_1| + |w_2| + |w_3| ] \quad \ell_1 \text{ reg}$$

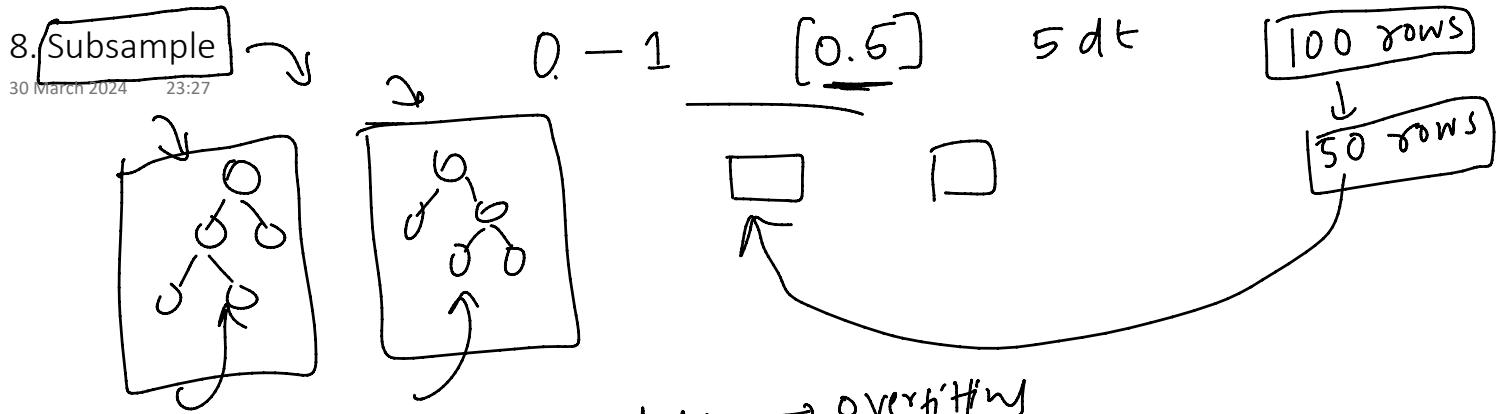
$\lambda, \alpha \rightarrow \text{loss}$

elastic net

$$T \rightarrow \frac{T}{\ell_1 \cdot \ell_2}$$

$\ell_2 \text{ reg} \gg \ell_1 \text{ reg}$

$$\gamma T + \frac{1}{2} \lambda \underbrace{\sum_{i=1}^T w_i^2}_{\text{L}} + \frac{1}{2} \alpha \overline{\sum_{i=1}^T |w_i|}$$

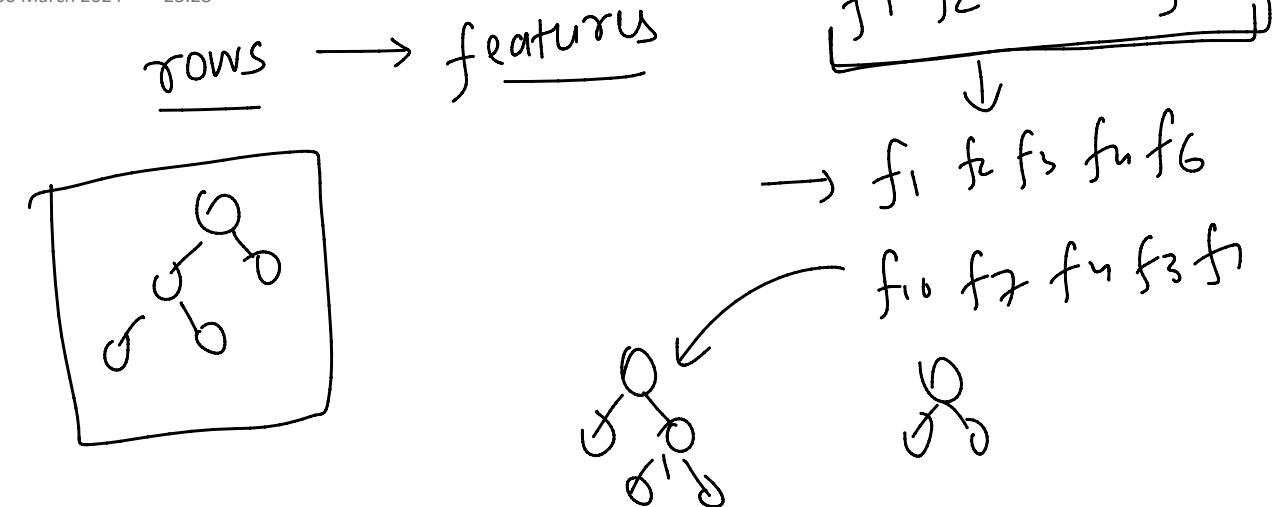


subset of the data → overfitting  
reduce

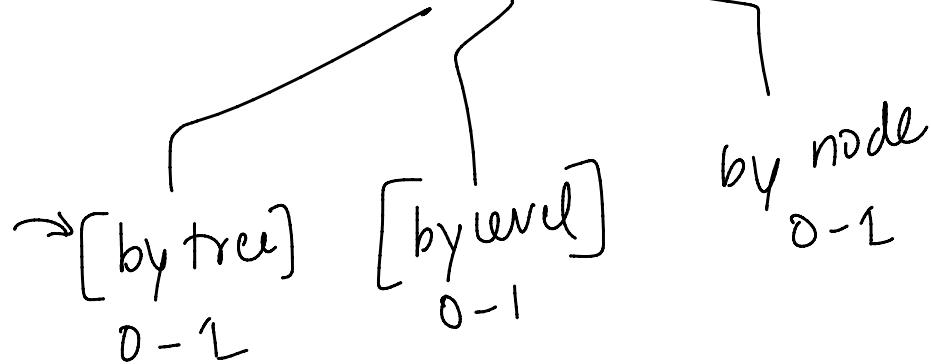
$0.11 - 11\%$      $1 \rightarrow \underline{100\%}$      $0.5$     Subsample  $\textcircled{1} \rightarrow \underline{100\%}$   
 $0.1$   
 $\underline{[0.5 - 0.8]}$

## 9. [Col Subsample]

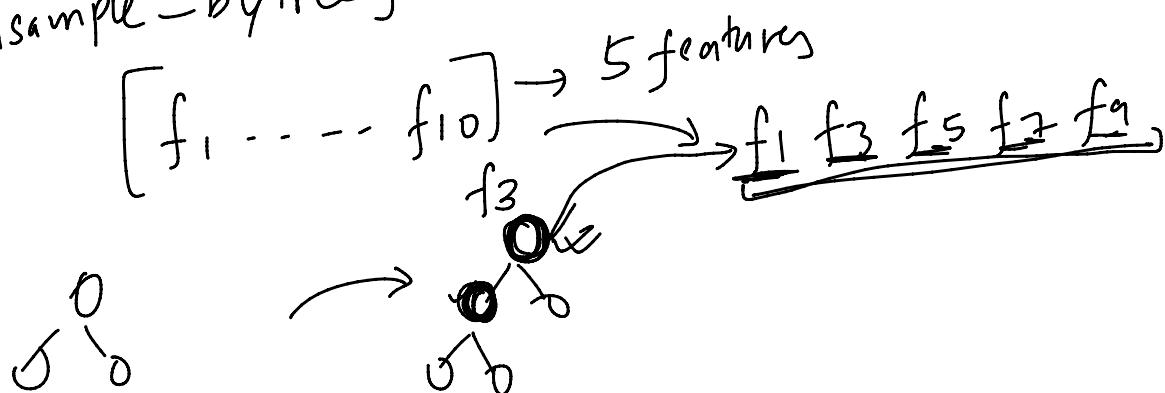
30 March 2024 23:28



col subsample



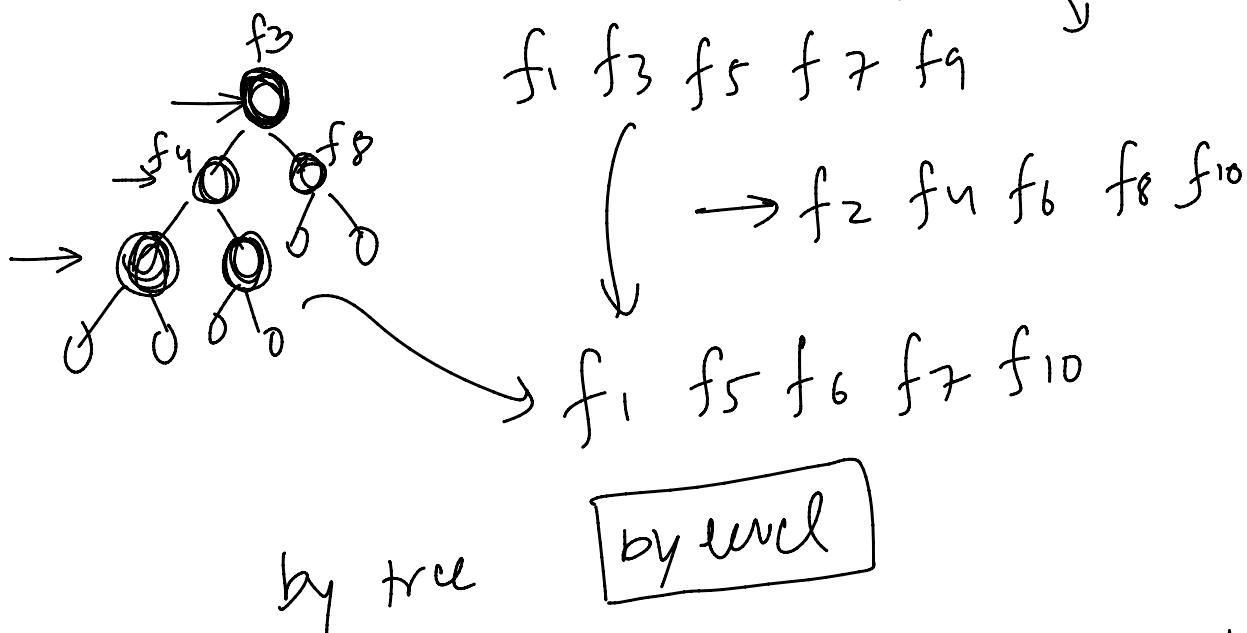
$$[\text{colsample\_bytree}] = 0.5$$



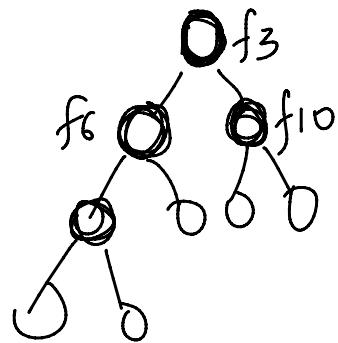
colsample-bylevel [0-1]

$$\downarrow 0.5$$

$$f_1 \ f_3 \ f_5 \ f_7 \ f_9$$



colsample -by node [0 - 1]      random  
 $\downarrow$       forest  
 $\underline{0.5}$



$f_1 \ f_2 \ f_3 \ f_6 \ f_5$

$f_1 \ f_5 \ f_6 \ f_7 \ f_8$

$f_6 \ f_7 \ f_8 \ f_9 \ f_{10}$

by tree

by level      by node  $\times$

# How XGBoost Handles Missing Values?

04 April 2024 17:40

# Cache Aware Access

04 April 2024 17:40

# XGBoost Hyperparameters

04 April 2024 09:21

Hyperparameter Group	Hyperparameter	Description	Possible Values	General Advice
Tree Complexity and Structure	max_depth ✓	Maximum depth of each tree.	Any positive integer	Controls overfitting; deeper trees capture more complex patterns.
	max_leaves ✓	Maximum number of terminal nodes in a tree.	Any positive integer or 0 (no limit)	More leaves increase model complexity and flexibility.
	max_bin	Maximum number of bins for histogram-based methods.	Any positive integer	More bins allow finer feature splits but increase computation.
	[max_delta_step] X	Maximum step for each tree's weight estimation.	Any non-negative float	Helps in logistic regression with imbalanced data. Rarely needed otherwise.
	grow_policy	Strategy to grow trees.	'depthwise', 'lossguide'	Determines how the tree nodes are expanded.
	tree_method	Algorithm for tree construction.	'auto', 'exact', 'approx', 'hist', 'gpu_hist'	Choose based on dataset size and computational resources.
	num_parallel_tree	Number of trees grown per round (for forests).	Any positive integer	More trees increase ensemble's robustness but

		per round (for forests).		ensemble's robustness but also computational load.
	interaction_constraints	Constraints on feature interactions.	List of pairs of feature indices	Limits interactions between features, based on domain knowledge.
Regularization	gamma ✓	Minimum loss reduction required to	Any non-negative float	Controls complexity; higher values
	min_child_weight ✓	Minimum sum of instance weight in a child.	Any non-negative float	Higher values prevent overfitting by avoiding creating nodes with few samples.
	reg_alpha ~	L1 regularization term on weights.	Any non-negative float	Encourages sparsity in the leaf weights, helping to regularize.
	reg_lambda ✓	L2 regularization term on weights.	Any non-negative float	Smoothens the leaf weights to prevent overfitting.
	monotone_constraints	Monotonicity constraints on features.	Dictionary or string	Enforces the model to learn a monotonic relationship with the target variable.

<b>Sampling and Column Subsampling</b>	subsample ✓	Subsample ratio of the training instances.	Float between 0 and 1	Lower values can help prevent overfitting but too low can cause underfitting.
	colsample_bytree ✓	Subsample ratio of columns when constructing each tree.	Float between 0 and 1	Controls the number of features considered by each tree, influencing overfitting.
	colsample_bylevel ✓	Subsample ratio of columns for each depth level.	Float between 0 and 1	Adjusts feature sampling as trees grow deeper, impacting model complexity.
	colsample_bynode ✓	Subsample ratio of columns for each split.	Float between 0 and 1	Fine-tunes the amount of feature consideration at each node split.

	sampling_method	Method to sample training instances.	'uniform', 'gradient_based'	'gradient_based' can focus on harder to learn samples, affecting model focus.
Boosting and Learning	n_estimators ✓	Number of boosting rounds or trees.	Any positive integer	More rounds increase complexity; should be tuned with learning rate.
	learning_rate ✓	Step size shrinkage used in update to prevent overfitting.	Any positive float	Lower rate requires more trees but can yield better generalization.
	booster	Type of model to run at each iteration.	'gbtree', 'gblinear', 'dart'	Selects the type of model to use at each step of boosting.
Objective and Evaluation	objective	The loss function to be minimized.	String or custom function	Should align with your task (regression, classification).
	eval_metric	Evaluation metric for validation data.	String, list of strings, or custom function	Should reflect the objective of the model; used for tuning and early stopping.

Miscellaneous Controls	scale_pos_weight	Balancing of positive and negative weights.	Any positive float	Important for imbalanced datasets to balance the influence of classes.
	base_score	The initial prediction score.	Any float	Generally, the default is sufficient; it's the initial prediction for all instances.
	random_state ✓	Random number seed.	Any integer or None	Ensures reproducibility of your model's results.
	missing	Value to represent missing data.	Typically `np.nan`	Ensures the model correctly handles missing values in your data.

	n_jobs	Number of parallel threads.	Any non-negative integer	Optimize to leverage hardware capabilities for faster training.
	verbosity 	Degree of verbosity.	Integer from 0 (silent) to 3 (debug)	Useful for monitoring the training process.
	validate_parameters	Whether to validate input parameters.	True or False	Helps in identifying potentially incorrect parameter settings.
	predictor	Type of predictor algorithm to use.	'cpu_predictor', 'gpu_predictor'	Useful when explicitly specifying to use CPU or GPU resources.
Early Stopping and Callbacks	early_stopping_rounds 	Rounds without improvement to stop training.	Any positive integer	Prevents overfitting by stopping training when validation metric stops improving.
	callbacks	Custom callback functions for training.	List of callback functions	Allows custom operations to be applied at each training iteration.

<b>Categorical Features and GPU Support</b>	enable_categorical	Support for categorical features.	True or False	Utilize to handle categorical features directly without preprocessing.
	gpu_id	ID of the GPU to use.	Any valid GPU ID	Specifies the GPU to use for training, if applicable.
	max_cat_to_onehot	Threshold for one-hot encoding for categorical features.	Any positive integer	Decides when to use one-hot encoding based on the number of categories.
	max_cat_threshold	Maximum categories considered for each split in partition-based trees.	Any positive integer	Helps to prevent overfitting with high cardinality categorical features.