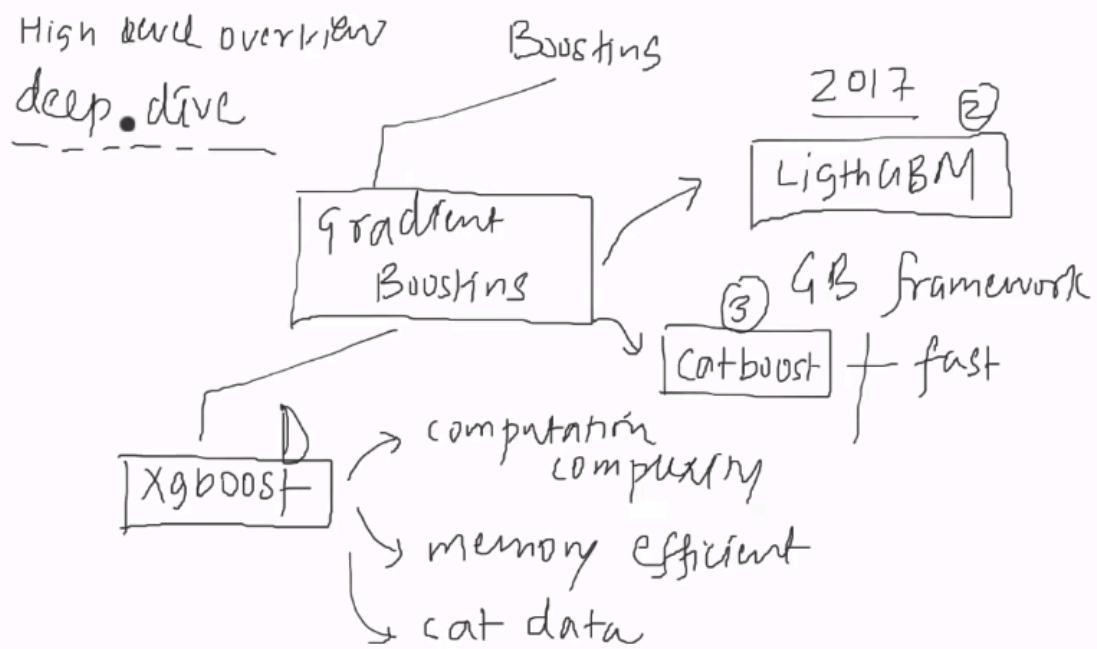


Plan of Attack

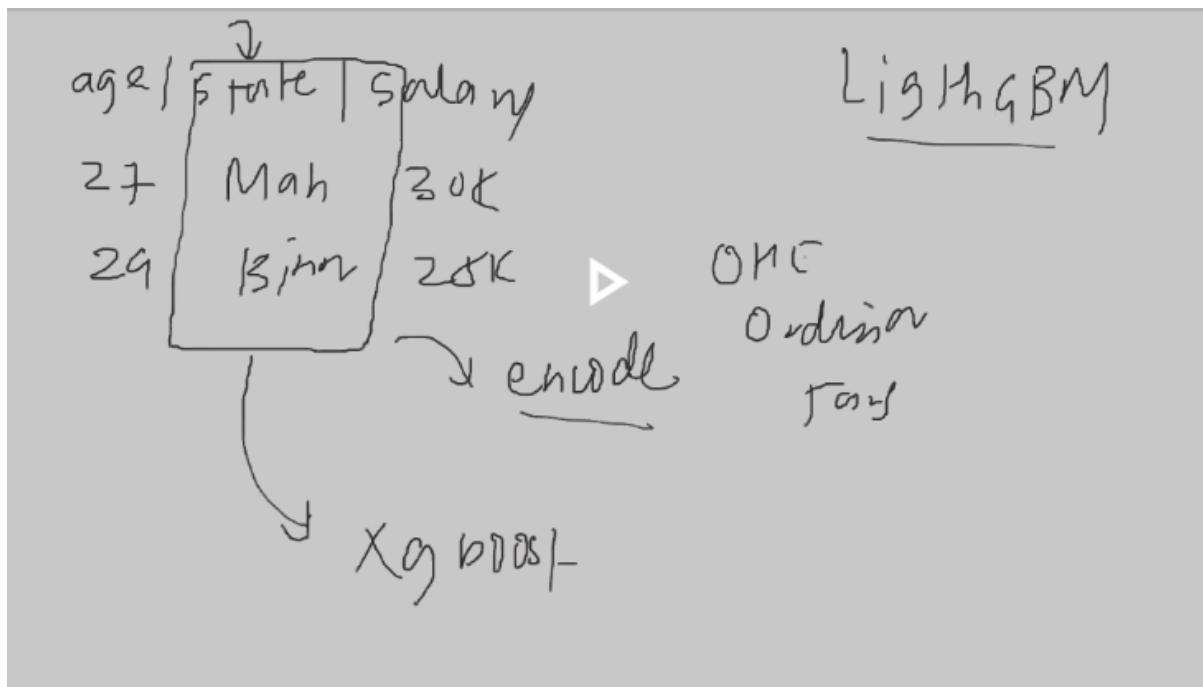


Introduction

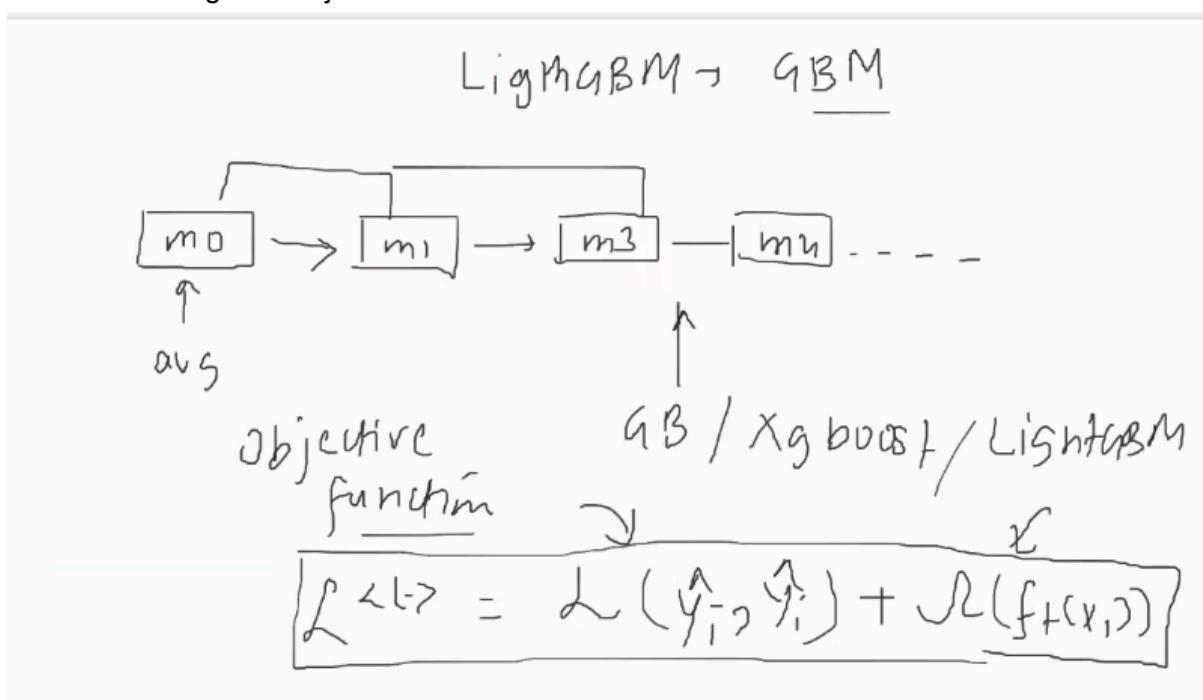
13 April 2024 08:29

NS

- LightGBM stands for Light Gradient Boosting Machine
- It is an open source Gradient Boosting Framework
- Developed by Microsoft DMTK in 2016
- Goal was to develop high performance Gradient Boosting framework optimized for speed.
- Core Features
 - o Speed and memory efficient
 - o Accuracy
 - o Works with categorical data
 - o Multiple Language support
 - o Supports Distributed Computing
 - o Supports GPU
 - o Can handle missing values and sparse data
 - o Can work with custom loss functions



1. Boosting and Objective Function



XGBoost / LightGBM

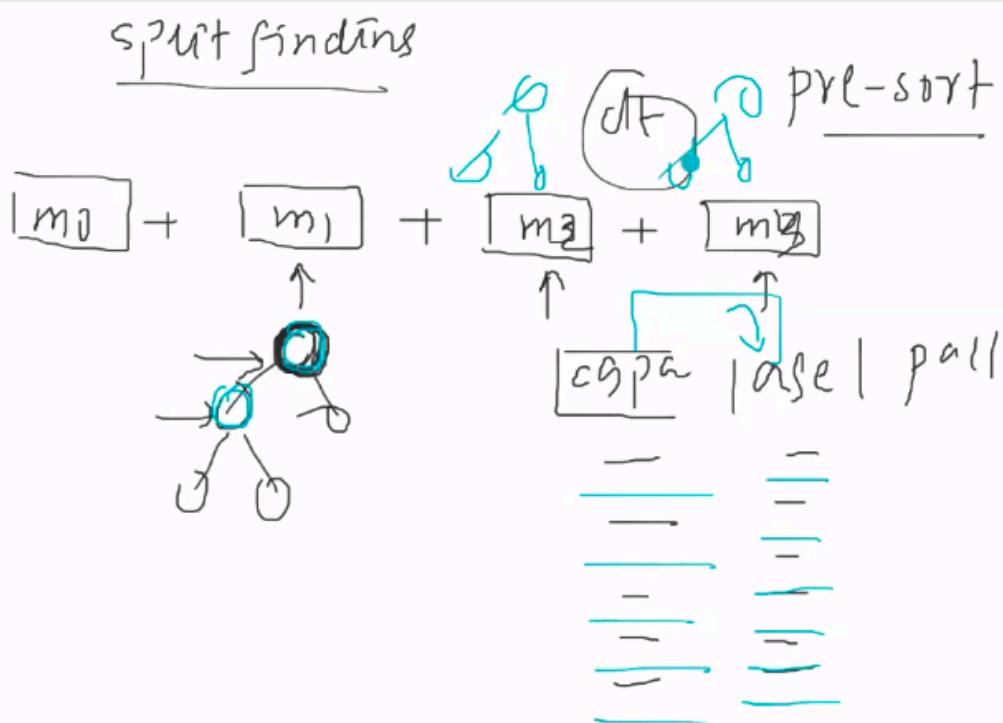
Objective same

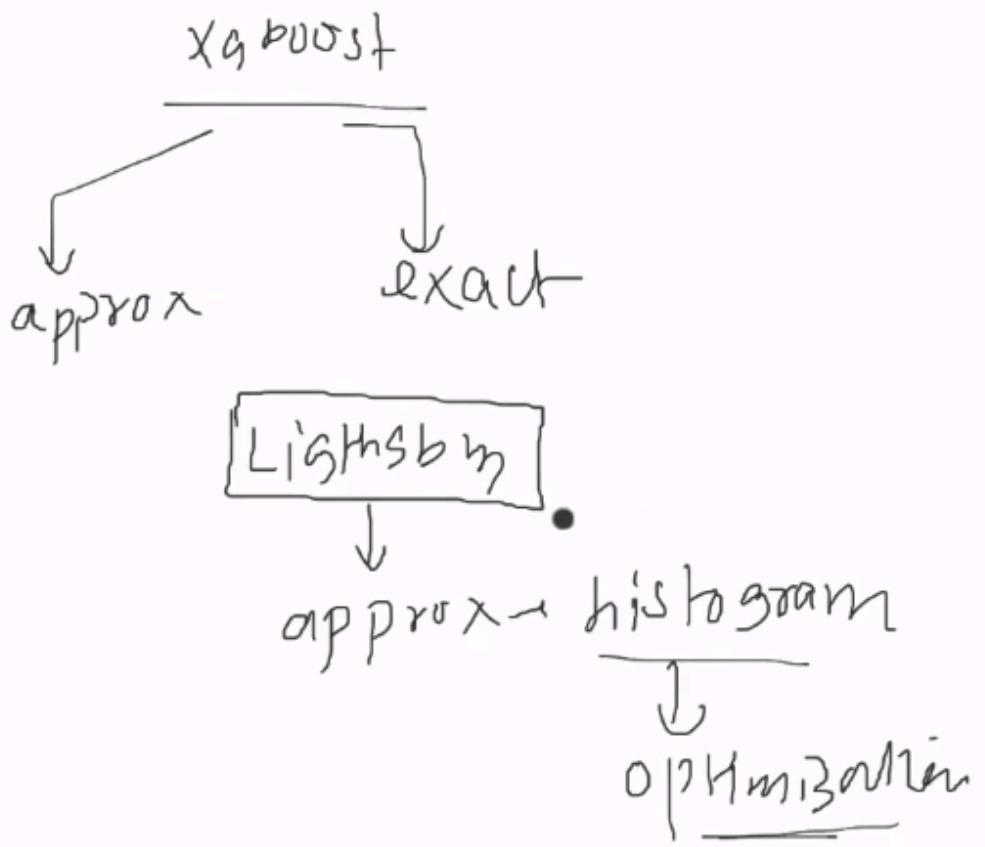
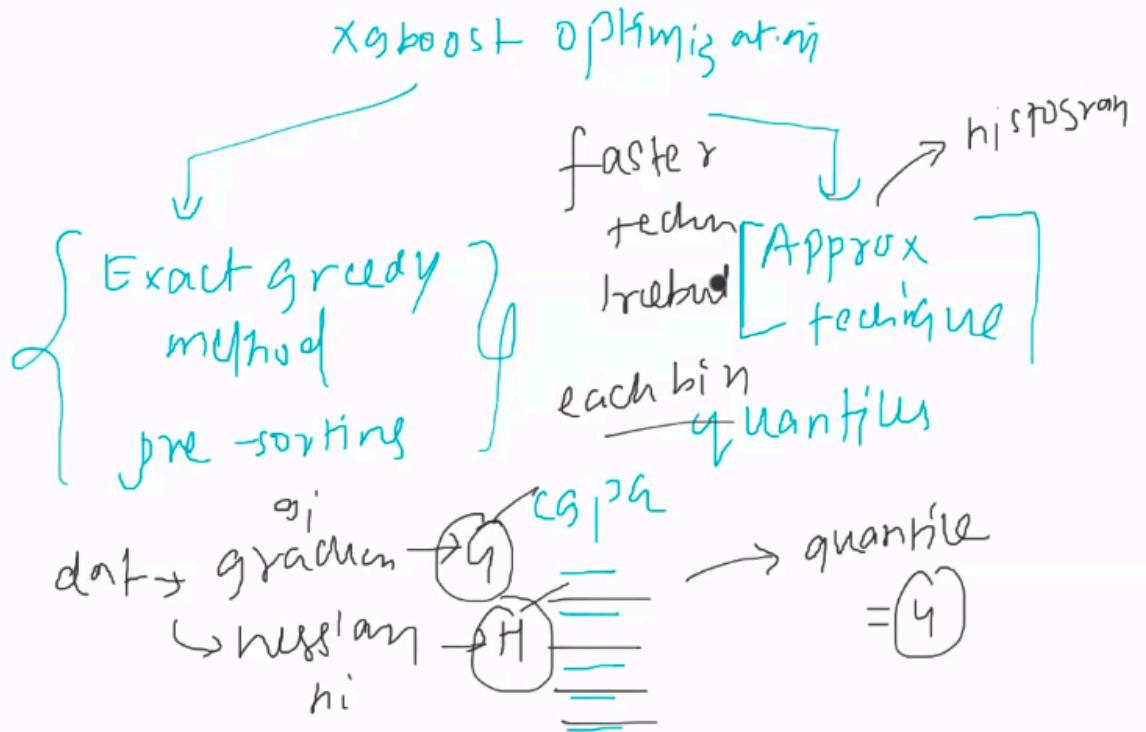
$$L(y_i, \hat{y}_i) + \nu_1 (f_t(x_i))$$



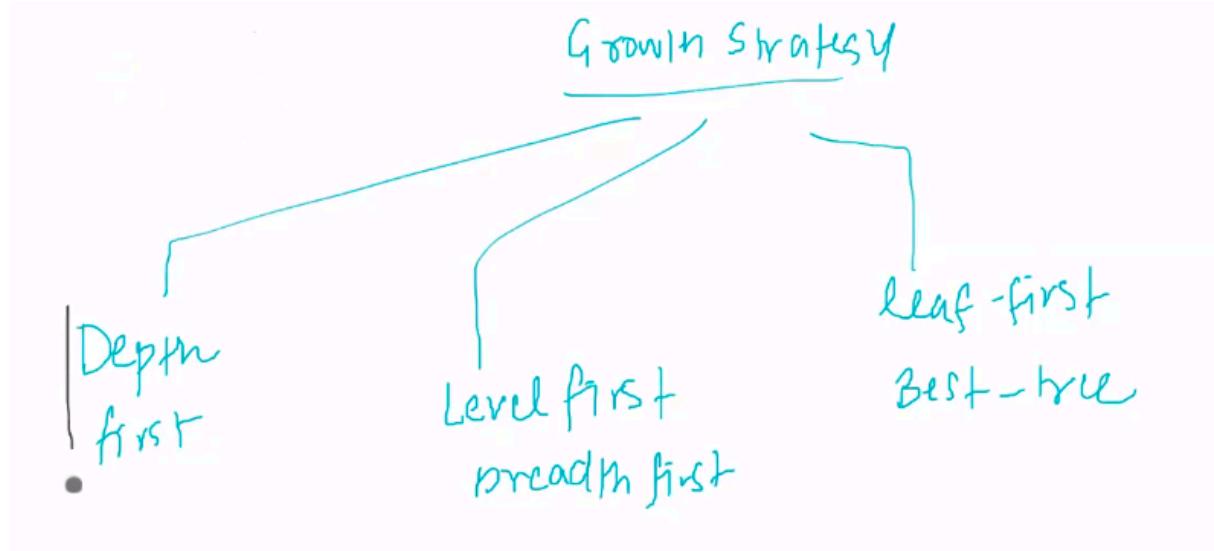
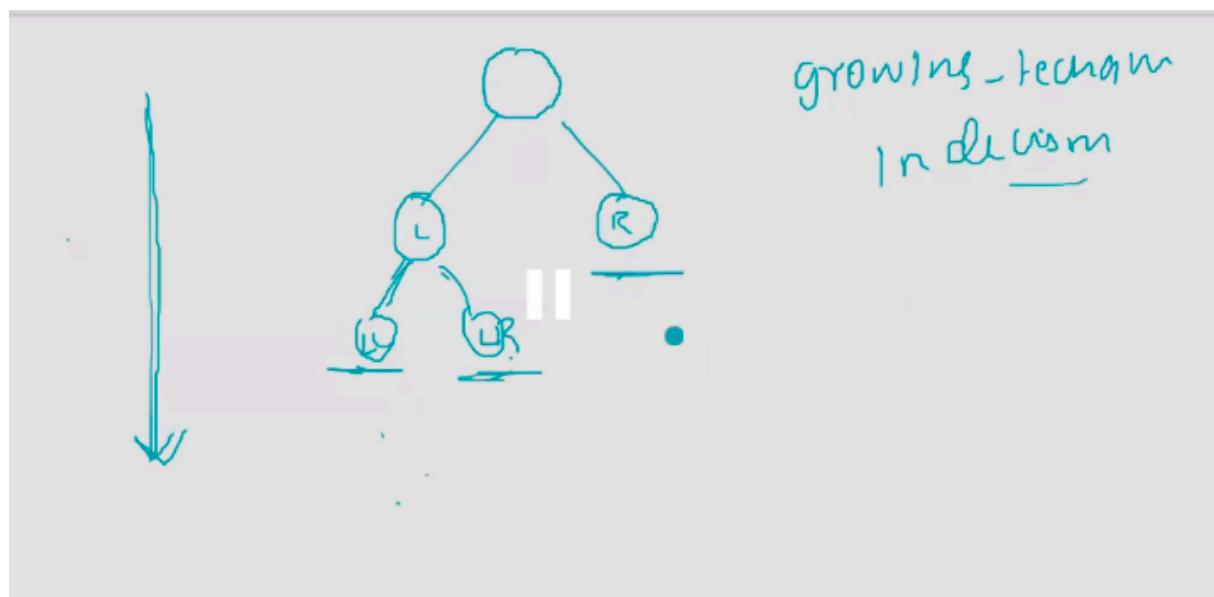
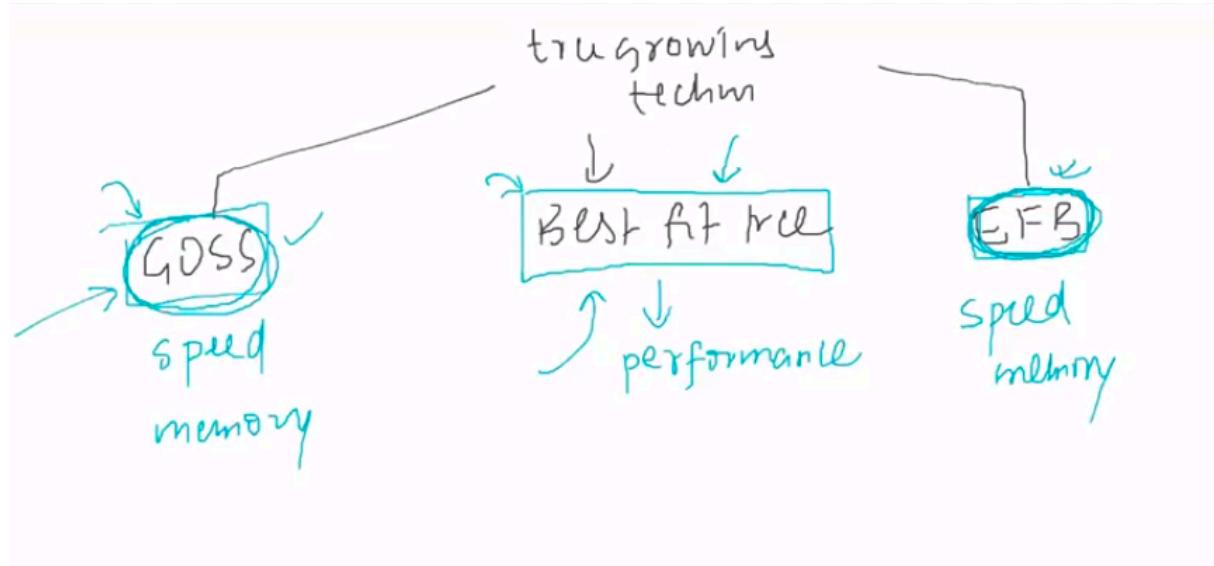
$$\left\{ \begin{array}{l} \text{gamma} \\ \text{reg-lambda} \\ \text{reg-alpha} \end{array} \right\}$$

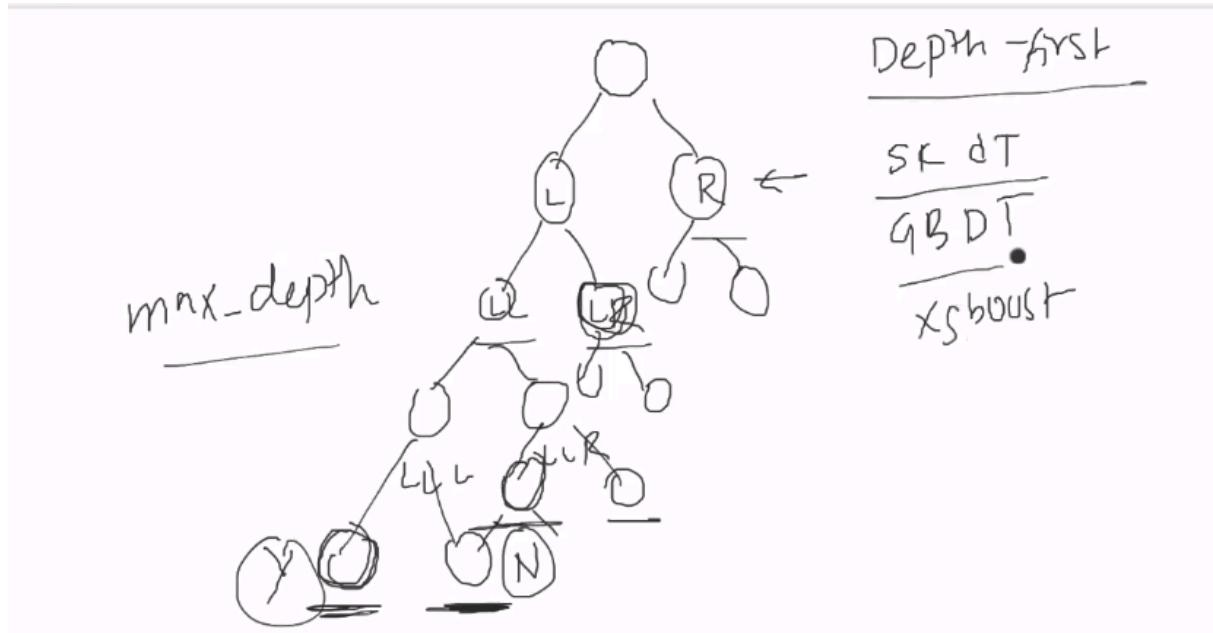
2. Histogram-based Split Finding



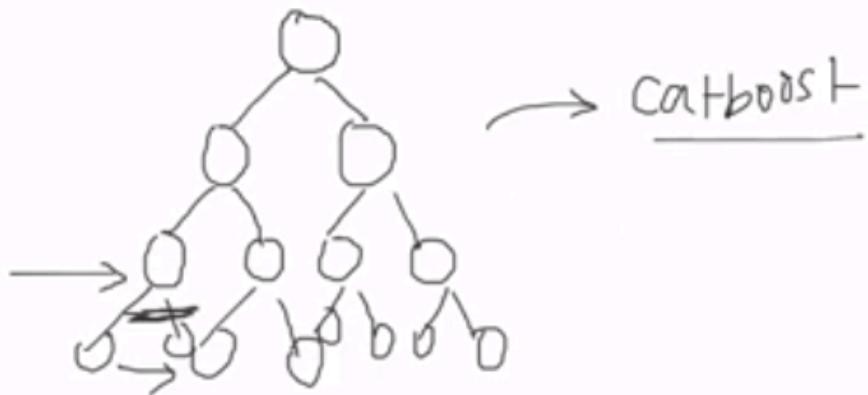


3. Best-fit Tree (Leaf-wise growth strategy)

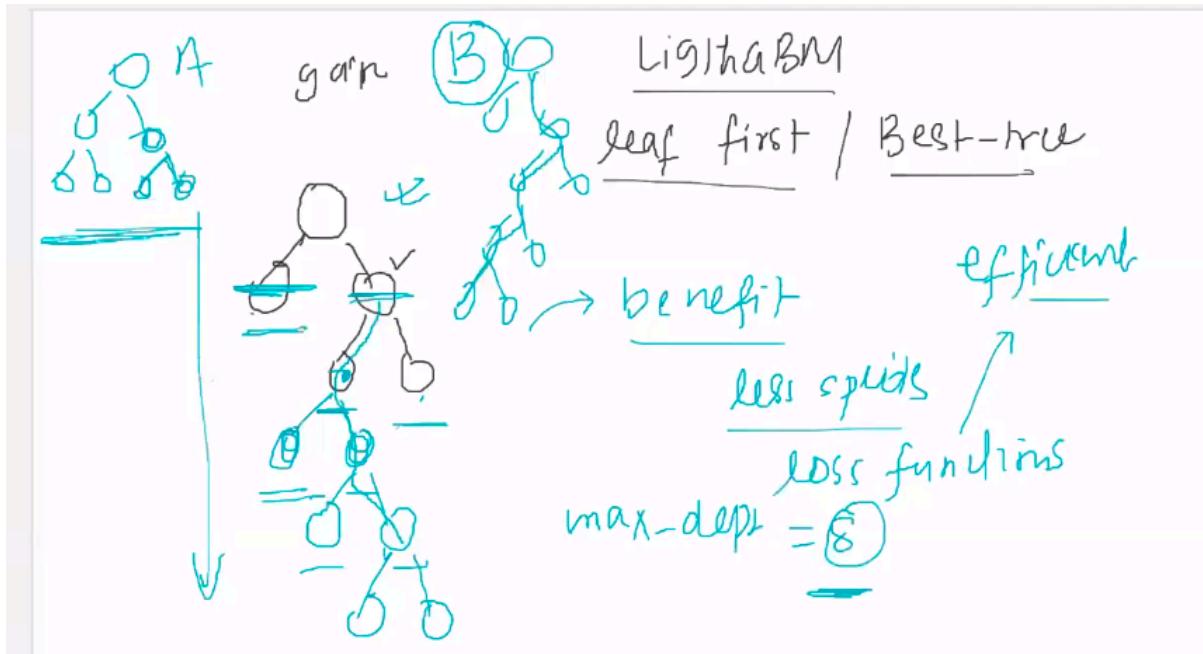




level-first / Breadthwise



Catboost

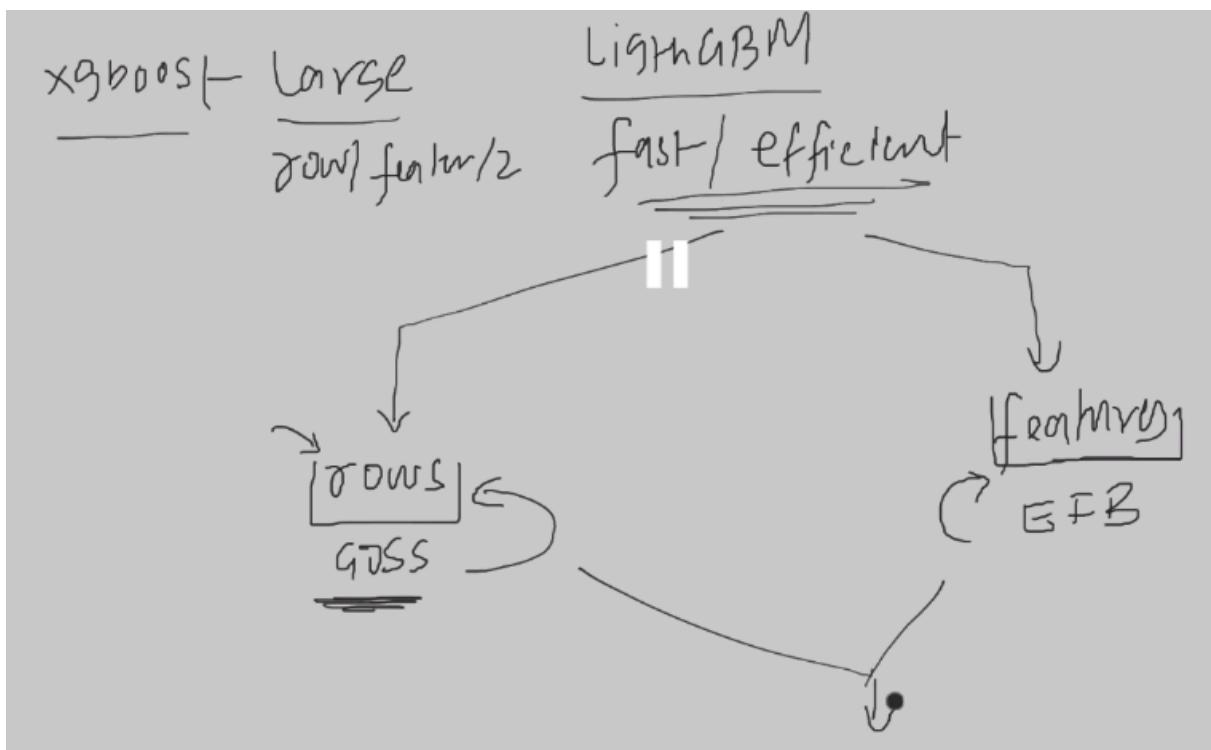


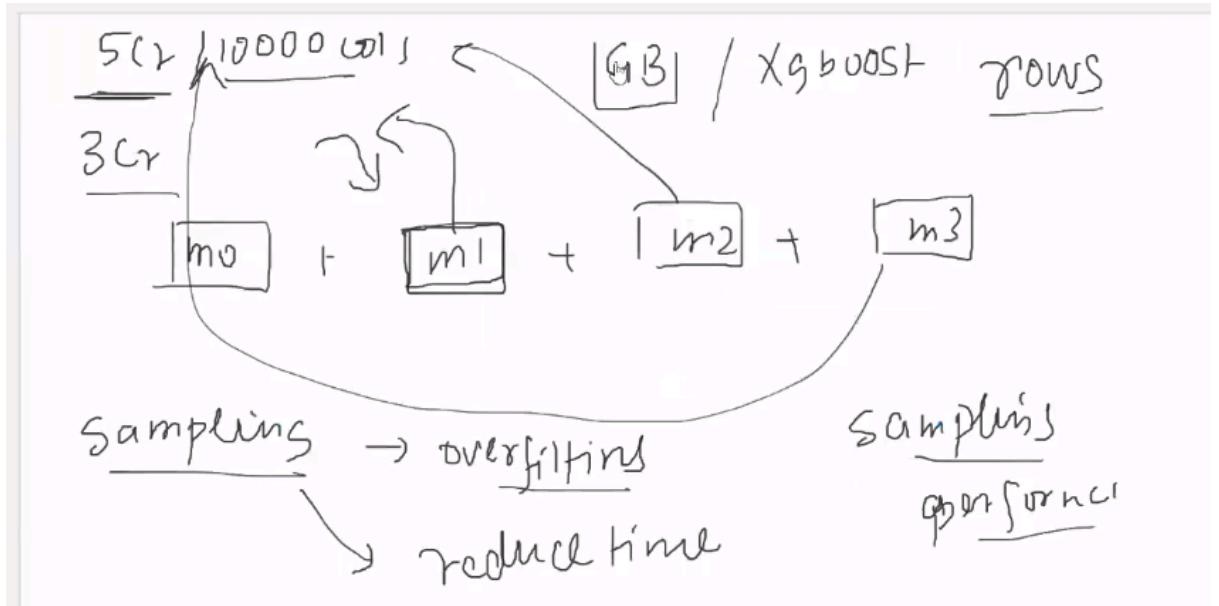
4. GOSS

GOSS is an innovative data sampling technique that addresses a common problem in the training of gradient boosting models: the balancing act between speeding up training by using a subset of the data (data sampling), and maintaining the accuracy of the learned model.

x1	x2	y	g
-0.4	-1.3	0	1.3
1.8	1.7	1	0.2
0.8	0.3	1	2.8
-1.2	-0.7	0	1.0
1.5	2.8	0	1.5
-0.4	0.3	1	0.8

x1	x2	y	g
0.8	0.3	1	2.8
1.5	2.8	0	1.5
-0.4	-1.3	0	1.3
-1.2	-0.7	0	1.0
-0.4	0.3	1	0.8
1.8	1.7	1	0.2





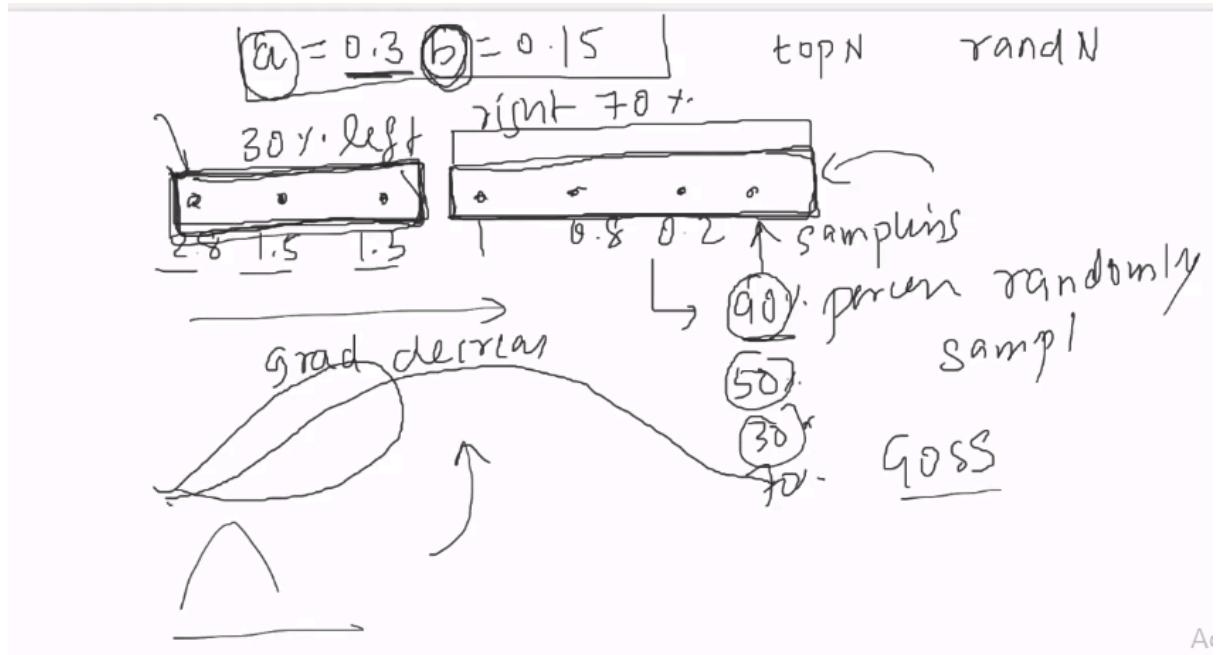
y
 gradient
 sorted
 l-g | 20 data points
 grad highest
 weights ↑ imp
 → gradient lowest

x1	x2	y	g
-0.4	-1.3	0	1.3
1.8	1.7	1	0.2
0.8	0.3	1	2.8
-1.2	-0.7	0	1.0
1.5	2.8	0	1.5
-0.4	0.3	1	0.8

x1	x2	y	g
0.8	0.3	1	2.8
1.5	2.8	0	1.5
-0.4	-1.3	0	1.3
-1.2	-0.7	0	1.0
-0.4	0.3	1	0.8
1.8	1.7	1	0.2

top 20
 sampling → random
 next delin.

cspan plan = $\frac{PC}{S_{\text{rac}}}$

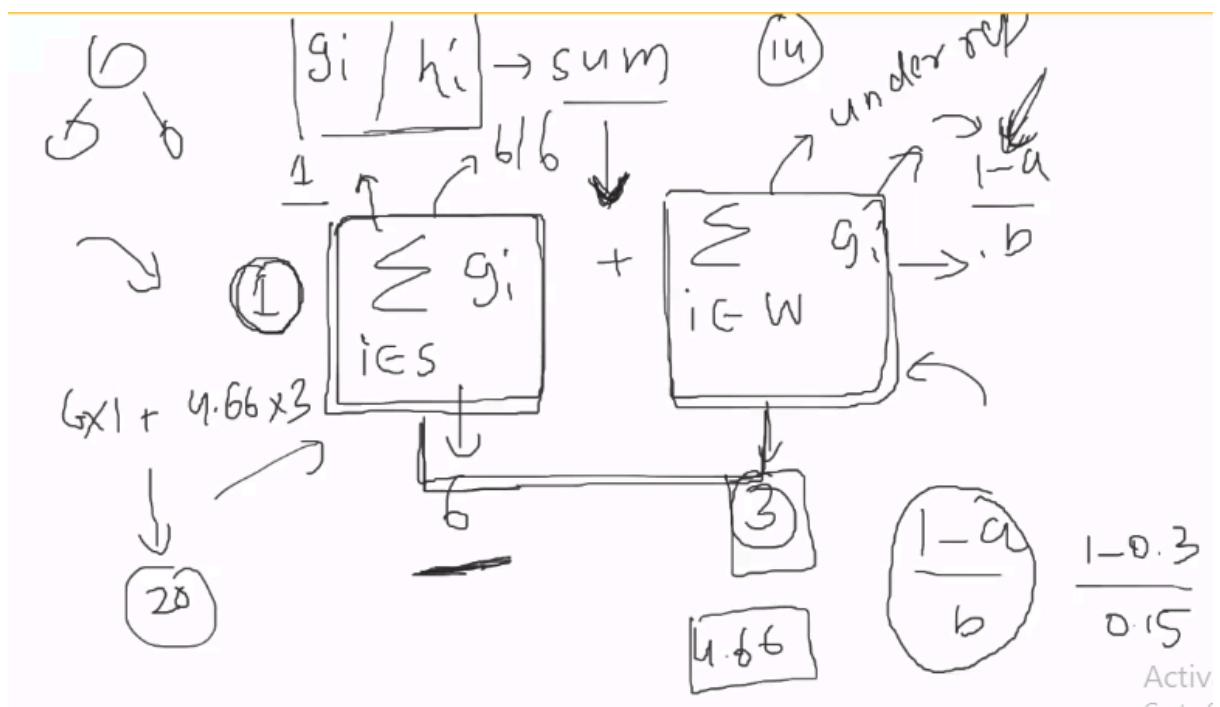


A

x_1	x_2	y	$ g $
-0.4	-1.3	0	1.3
1.8	1.7	1	0.2
0.8	0.3	1	2.8
-1.2	-0.7	0	1.0
1.5	2.8	0	1.5
-0.4	0.3	1	0.8
-0.4	0.3	1	0.2

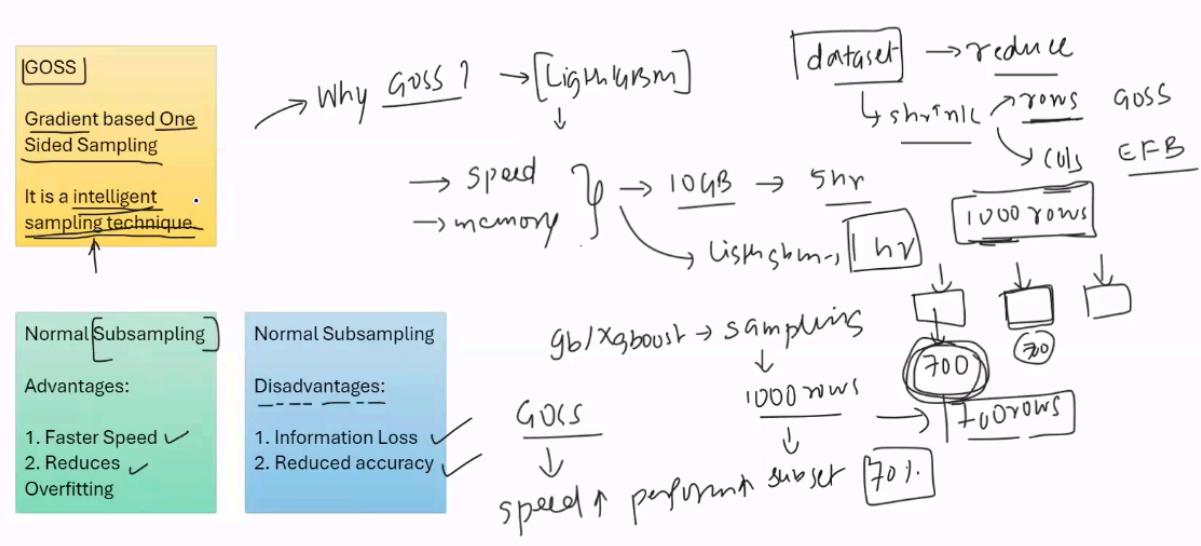
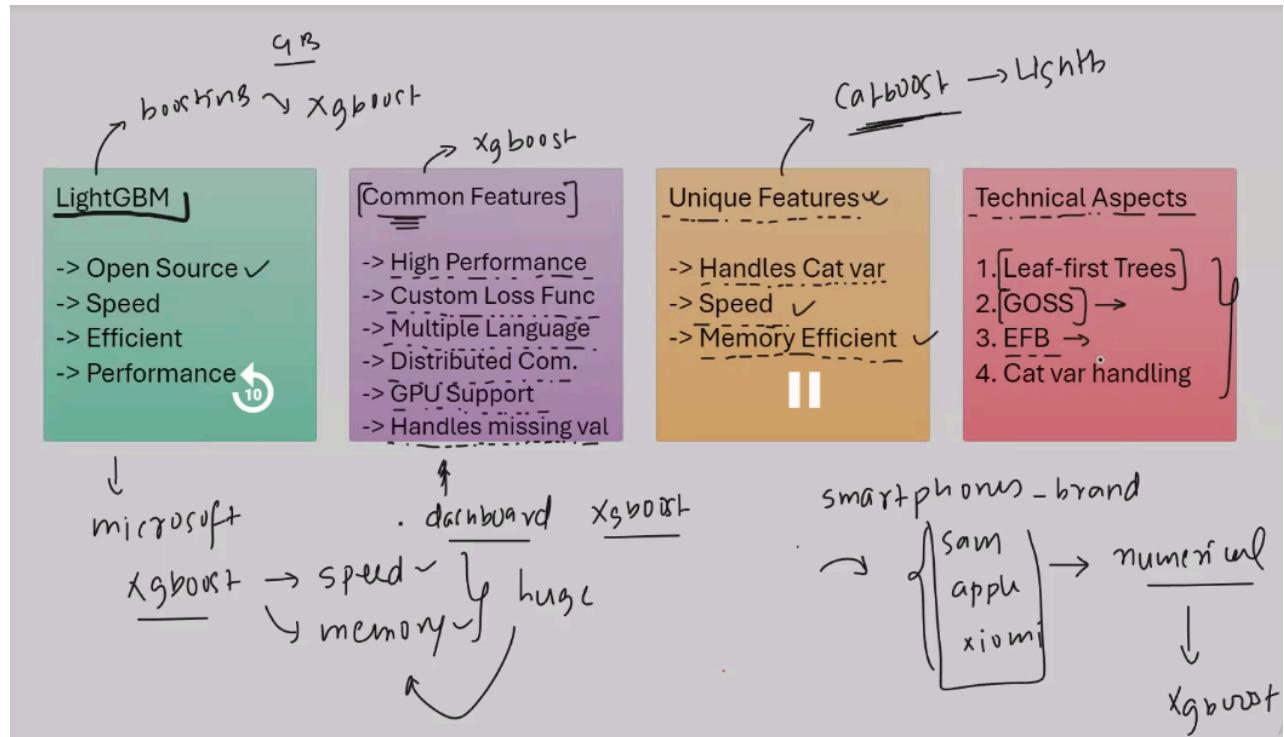
$a = 0.3$ $b = 0.15$
 $\text{topN} = a \times 20$ $(6) + (3)$
 $9 \text{ points } \frac{0.3 \times 20}{= 6}$
 $\text{randN} = b \times 20$ 0.15×20
 3
 highgrad
 $6+3$
 imp
 17

Activation



Session - 2 LightGBM

1. Recap

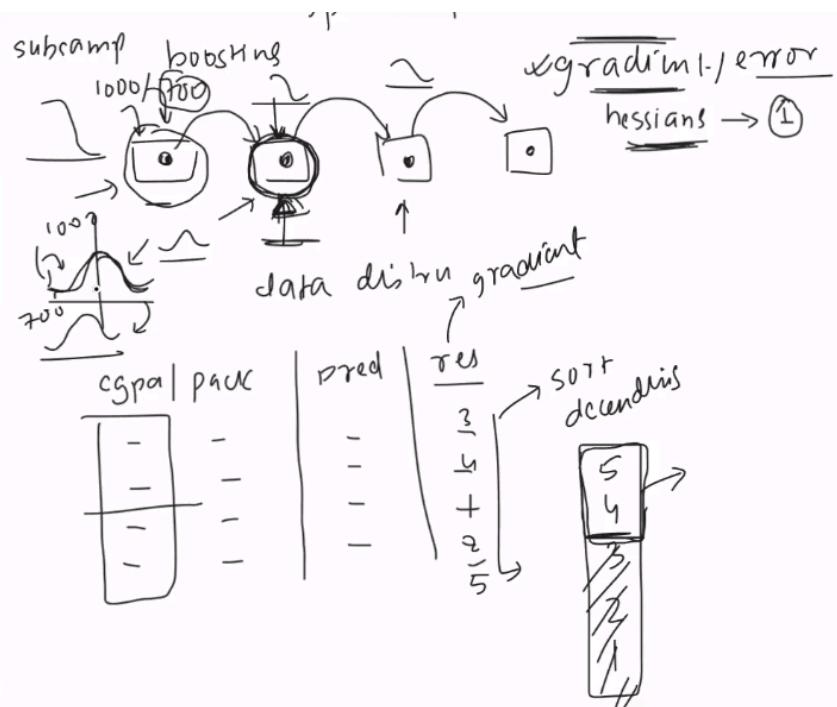


GOSS Main Idea

Some points are more important than other points!

Simplified Solution

Send high gradient points to the next boosting round.



Problem with this

Refined Approach

Best of both worlds!

How Exactly?

egpa	pack	pred	res/grad	abs
8	8	6	-2	2
6	6	3	-3	3
5	5	8	3	3
4	13	25	12	12
1	:	:	:	:
5	6	:	:	:
1	:	:	:	:
1	:	:	:	:
.
$\alpha = \text{top rate}$		$b = \text{other rate}$		
$\boxed{12 \ 3 \ 3 \ 2}$		$\boxed{\dots \dots}$		
$\alpha = 0.3$				
			$b = 0.15$	

$\text{top pts} = \underline{\alpha} \times 20$
 $0.3 \times 20 = 6$

$b \times 20 = 0.15 \times 20$
 $= 3$

gbdt

Hyperparameters

1. Boosting Type
2. Top_rate
3. Other_rate

Gauss

gbdt

data

ccpa

	ccpa	pack	pred1	grad	pred2	Grad
8	16	10	-6	6	-10	-10
7	12	10	-2	8	-4	-4
6	10	10	0	3	-7	-7
5	3	10	2	5	2	2
5	5	10	5	8	3	3

$a = \frac{1}{b} = \frac{6+3}{6+15} = \frac{1-0.3}{0.15}$

3. EFB

3. EFB

Purpose of EFB:

$$f_1 f_2 f_3 f_4 f_5 - \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}$$

LightGBM
+ speed
+ memory

Exclusive Feature Bundling is designed to reduce the dimensionality of the dataset by combining sparse and exclusive features into a single feature. This helps in managing datasets with a large number of features, where many of the features do not co-occur frequently.

Mechanism of Action:

EFB works by identifying features that are "exclusive," meaning their non-zero values rarely overlap with non-zero values of other features. These features are then combined into bundles where each original feature occupies a unique range of values, achieved by adding offsets.

Benefits for Machine Learning Models:

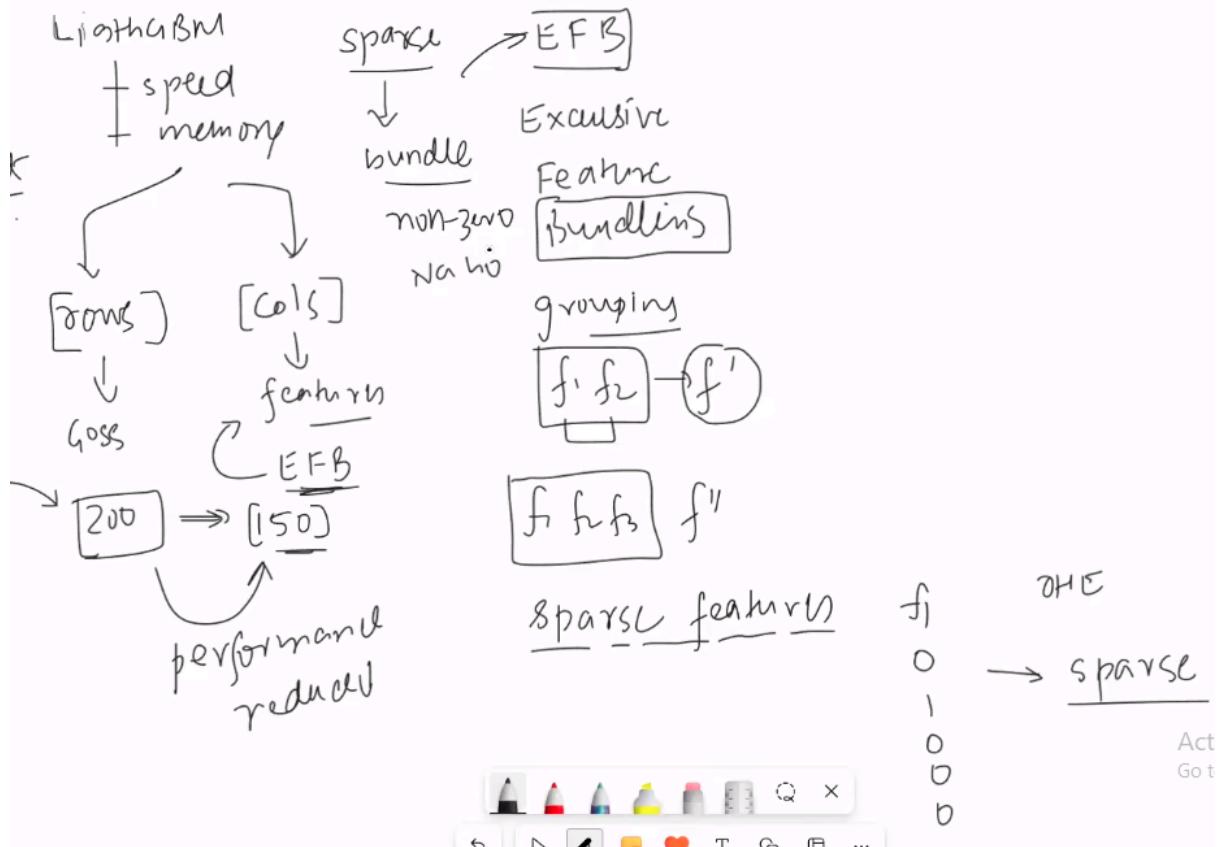
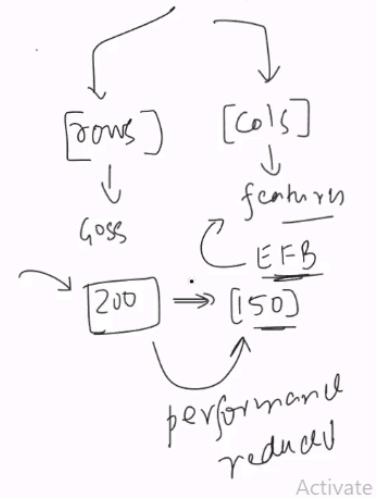
By reducing the number of features, EFB decreases the computational complexity and memory usage of the model. This leads to faster training times and can also help in reducing overfitting by simplifying the model.

Impact on Performance:

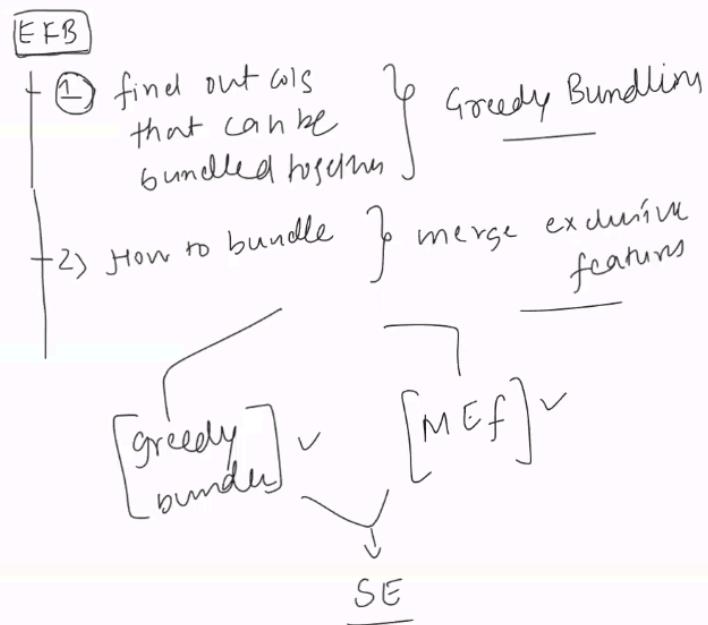
Bundling maintains the integrity of the original data by ensuring that the machine learning model can still differentiate between the contributions of the originally separate features. This allows the model to perform well even with the reduced number of features.

Application Context:

EFB is particularly useful in scenarios involving large-scale and high-dimensional data, especially where many features are categorical or have many zeros (sparse). It makes models like LightGBM highly effective and efficient in handling such datasets, which are common in areas like click prediction and other web-scale applications.



	F0	F3	(3)	x	5 bits
R1	F4	F2	Features		
i0	1	1	0	0	1
i1	0	0	1	1	1
i2	1	2	0	0	2
i3	0	0	2	3	1
i4	2	1	0	0	3
i5	3	3	0	0	1
i6	0	0	3	0	2
i7	1	2	3	4	3
i8	1	0	1	0	0
i9	2	3	0	0	2



	F0	F3	(3)	x	5 bits
R1	F4	F2	Features		
i0	1	1	0	0	1
i1	0	0	1	1	1
i2	1	2	0	0	2
i3	0	0	2	3	1
i4	2	1	0	0	3
i5	3	3	0	0	1
i6	0	0	3	0	2
i7	1	2	3	4	3
i8	1	0	1	0	0
i9	2	3	0	0	2

conflict count matrix

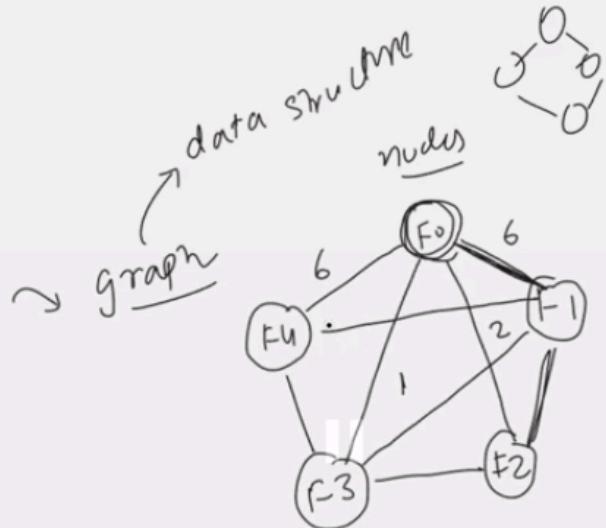
	F0	F1	F2	F3	F4
F0		6	2	1	
F1	6				
F2					
F3					
F4					

0
□
□
□

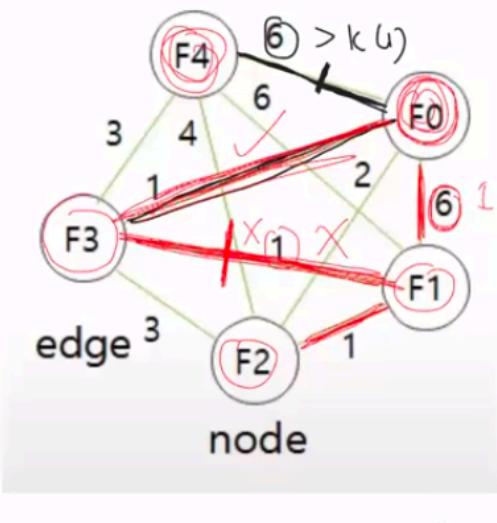
↓

conflict count matrix

	F0	F1	F2	F3	F4
F0		(6)	(2)	(1)	6
F1	6		1	1	6
F2	2	1		3	4
F3	1	1	3		3
F4	6	6	4	3	
Σ	15	14	10	8	19

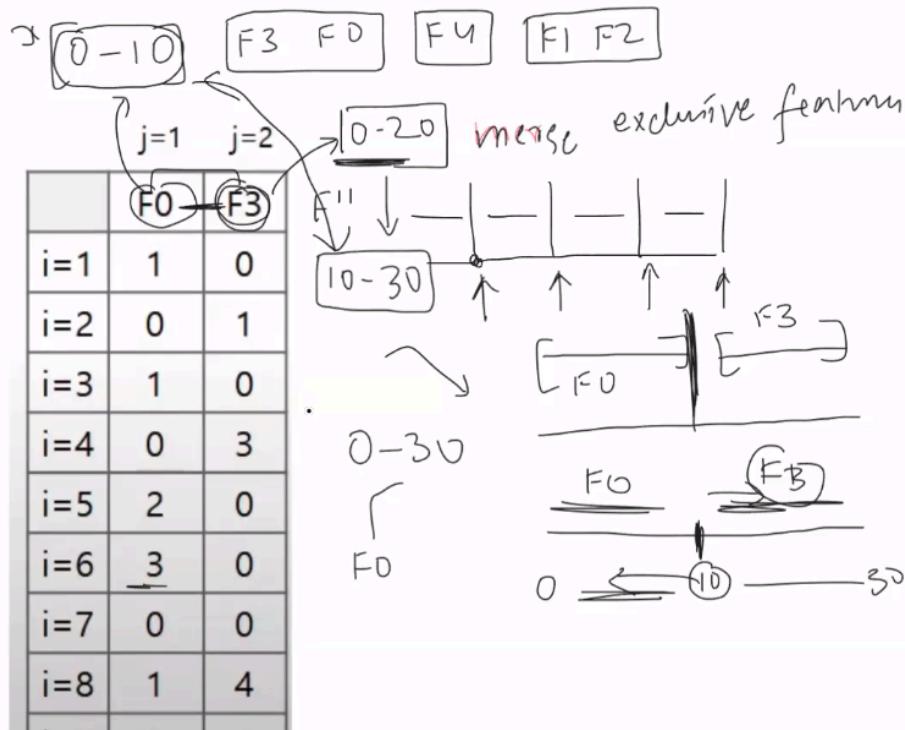


graph G



$$\max \underline{\text{conflict}} \quad \underline{(k)} = 1$$

Diagram illustrating a beam with supports and reaction forces. The beam is shown with two supports and a central reaction force. The supports exert reaction forces F_3 and R_1 . The central support also exerts a reaction force F_4 . The beam is subjected to a horizontal force F_1 at its center. The resulting deflection curve is shown as a parabola with segments labeled f'' and f''' . A curved arrow indicates the direction of deflection.



Algorithm 4: Merge Exclusive Features

```

Input: numData: number of data
Input: F: One bundle of exclusive features
binRanges ← {0}, totalBin ← 0
for f in F do
    totalBin += f.numBin
    binRanges.append(totalBin)
newBin ← new Bin(numData)
for i = 1 to numData do
    newBin[i] ← 0
    for j = 1 to len(F) do
        if F[j].bin[i] ≠ 0 then
            newBin[i] ← F[j].bin[i] + binRanges[j]

```

Output: newBin,

3. EFB

	F_0	F_3
$i=1$	1	0
$i=2$	0	1
$i=3$	1	0
$i=4$	0	3
$i=5$	2	0
$i=6$	3	0
$i=7$	0	0
$i=8$	1	4
$i=9$	1	0
$i=10$	2	0

$$binRanges = [0, 10, 20, 30] \text{ totalBins} = 0$$

$|+0|$

$\begin{matrix} i=1 & j=1 \\ i=1 & j=2 \\ i=2 & j=1 \\ i=2 & j=2 \end{matrix}$

Algorithm 4: Merge Exclusive Features

```

Input: numData: number of data
Input: F: One bundle of exclusive features
binRanges ← {0}, totalBin ← 0
for f in F do
    totalBin += f.numBin
    binRanges.append(totalBin)
newBin ← new Bin(numData)
for i = 1 to numData do
    newBin[i] ← 0
    for j = 1 to len(F) do
        if F[j].bin[i] ≠ 0 then
            newBin[i] ← F[j].bin[i] + binRanges[j]

```



Activate Windows
Go to Settings to activate Windows.

$F = \{F_0, F_3\}$ $\text{binRanges} = [0, 3, 7]$ $\text{totalBins} = 0$ $i=3 \quad j=1$
 $j=2 \quad i=4 \quad j=1$

Algorithm 4: Merge Exclusive Features
Input: numData : number of data
Input: F : One bundle of exclusive features
 $\text{binRanges} \leftarrow [0]$, $\text{totalBin} \leftarrow 0$
for f **in** F **do**
 $\text{totalBin} += f.\text{numBin}$
 $\text{binRanges.append(totalBin)}$
 $\text{newBin} \leftarrow \text{new_Bin}(\text{numData})$
for $i = 1$ **to** numData **do**
 $\text{newBin}[i] \leftarrow 0$
for $j = 1$ **to** $\text{len}(F)$ **do**
 if $F[j].\text{bin}[i] \neq 0$ **then**
 $\text{newBin}[i] \leftarrow F[j].\text{bin}[i] + \text{binRanges}[j]$
Output: $\text{newBin}, \text{binRanges}$

Activate Windows
Go to Settings to activate Windows.