模型表达：$\hat{y}_i^{(B)} = \sum_{b=1}^{B-1} f_b(x_i) + f_B(x_i)$

$$obj^{(b)} = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(b)}) + \sum_{j=1}^{b} \Omega(f_j)$$

$\Omega(f_j)$ is regularization term

$\Omega(f_j) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} C_j^2$ (γ, λ are hyperparams)

# of DT 节点

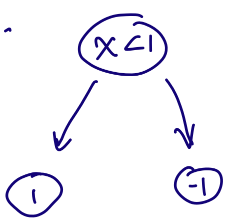
$\sum_{j=1}^{b-1} \Omega(f_j)$ is known by the time we train tree b.

$$\sum_{j=1}^{b} \Omega(f_j) \Rightarrow \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} C_j^2$$

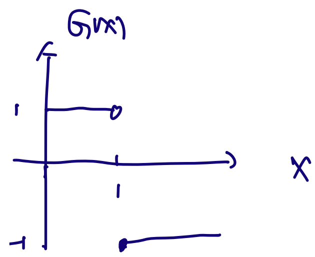
优化时，正则项只与当前 tree 的 T 和 Cj 有关

梯度下降不适合树模型

eg.



$$G(x) = \begin{cases} 1 & x<1 \\ -1 & x \geq 1 \end{cases}$$



不 continuous, can't take derivative

eg. $L(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$

$$obj^{(b)} = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(b)}) + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} c_j^2$$

$$\Rightarrow obj^{(b)} = \gamma T + \sum_{j=1}^{T}\Big[\sum_{x_i \in R_t} L(y_i, \hat{y}_i^{(b-1)} + c_j)\Big] + \frac{1}{2}\lambda \sum_{j=1}^{T} c_j^2$$

T: # of leaf nodes.

c_j: value assigned to jth leaf node

we don't know exact form of L, so how can we min L step by step! ⇒ approx L

GBDT: 一阶 taylor

XGBoost: 二阶 taylor

$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0)$$

For individual sample i:

$$\underset{\text{常量}}{L}(y_i, \underset{\text{equiv to 'x'}}{\hat{y}_i^{(t)}}) = L(y_i, \underbrace{\hat{y}_i^{(t-1)}}_{x_0} + \underbrace{c_j}_{x - x_0})$$

第t颗树对应当前样本i的叶节点取值

$$\approx \underbrace{L(y_i, \hat{y}_i^{(t-1)})}_{\text{常量}} + c_j \cdot L'(y_i, \hat{y}_i)\Big|_{\hat{y}_i = \hat{y}_i^{(t-1)}} + \frac{1}{2} c_j^2 L''(y_i, \hat{y}_i)\Big|_{\hat{y}_i^{(t-1)}}$$

$$\Rightarrow \min \; c_j \cdot \underbrace{L'(y_i, \hat{y}_i)\Big|_{\hat{y}_i = \hat{y}_i^{(t-1)}}}_{g_i:\ \text{一阶导}|_{\hat{y}^{(t-1)}}} + \frac{1}{2} c_j^2 \underbrace{L''(y_i, \hat{y}_i)\Big|_{\hat{y}_i^{(t-1)}}}_{h_i:\ \text{二阶导}|_{\hat{y}^{(t-1)}}}$$

$$= \min \; c_j g_i + \frac{1}{2} c_j^2 h_i$$

Aggregate: $\min \sum_{j=1}^{T} \sum_{x_i \in R_j} L(y_i - \hat{y}_i^{(t-1)} + c_j) + \gamma T + \frac{1}{2}\lambda \sum c_j^2$

$$\approx> \min \sum_{j=1}^{T} \left[c_j \sum_{x_i \in R_j} g_i + \frac{1}{2} c_j^2 \sum_{x_i \in R_j} h_i + \frac{1}{2}\lambda c_j^2 \right) + \gamma T$$

$$\Rightarrow obj^{(b)} = \sum_{j=1}^{T} \left[c_j \underbrace{\boxed{\sum_{x_i \in R_j} g_i}}_{G_j} + \frac{1}{2} c_j^2 \left(\underbrace{\boxed{\sum_{x_i \in R_j} h_i}}_{H_j} + \lambda \right) \right] + \gamma T$$

j: jth leaf node

g_i, h_i can be computed directly

$$(c_1^* \dots c_T^*) = \arg\min_{c_j} obj^{(b)}$$

* Each leaf node can be computed individually to min L, and then Σ results

$$\forall j \in [T] \qquad c_j^* = \arg\min \left(c_j G_j + \frac{1}{2} c_j^2 (H_j + \lambda) \right) + \gamma T$$

$(\because$ 当 tree grow 好后，每个 (x_i, y_i) 落入哪个 R is determined by x_i, deterministic$)$

$$c_j^* = \arg\min \left(c_j G_j + \frac{1}{2} c_j^2 (\underbrace{H_j + \lambda}_{>0}) \right) + \gamma T$$

H is Σ of 2nd derivative of L fn, which is convex.

$\Rightarrow H > 0$

So this 一元二次 to min is convex too

$$\begin{cases} c_j^* = -\frac{b}{2a} = -\frac{G_j}{H_j + \lambda} \\ obj^{*(b)} = \gamma T - \frac{1}{2} \frac{G_j}{H_j + \lambda} \end{cases}$$

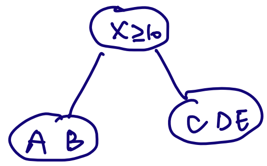
树的划分:

以上推导都建立在树已构建好的基础上，那树如何构建呢?

① Exact Greedy Algo

eg. ABCDE

G H



G_L H_L G_R H_R

$$w^* = -\frac{G}{H+\lambda}$$

$$obj^*_{before} = \gamma \cdot 1 - \frac{1}{2}\frac{G^2}{H+\lambda}$$

$G \to G_L + G_R$, $H \to H_L + H_R$

known & deterministic before split

$$obj^*_{after} = \gamma \cdot 2 - \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda}\right)$$

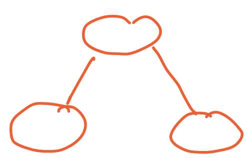
for diff split, will have diff obj^*_{after}

$gain = obj^*_{before} - obj^*_{after}$ (loss reduction after split)

$$gain = \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right) - \gamma$$

use split that max gain

Relation to CART



$c_L^* = \bar{y}_L$ $c_R^* = \bar{y}_R$

$$L_{before} = \left(\sum_i (y_i - \bar{y})^2\right) / N = Stdev_{total}$$ known & deterministic before split.

$$L_{after} = \left(\sum_{x_i \in R_L} (y_i - \bar{y}_L)^2 + \sum_{x_i \in R_R} (y_i - \bar{y}_R)^2\right) / (N_L + N_R)$$

$$= Stdev_L \cdot P_L + Stdev_R \cdot P_R$$

when to stop: ① max gain ≤ 10e-05

② 叶子节点包含样本个数 ≤ 1

③ depth | # of leaf nodes

Algo: Exact Greedy Search for Split Finding.

Algorithm 1: Exact Greedy Algorithm for Split Finding

Input: I, instance set of current node
Input: $\boxed{d,}$ feature dimension
$gain \leftarrow 0$ 怀疑是作者写错了
$G \leftarrow \sum_{i\in I} g_i$, $H \leftarrow \sum_{i \in I} h_i$
for $k = 1$ *to* $\boxed{d}$ **do** ← 遍历特征
  $G_L \leftarrow 0,\ H_L \leftarrow 0$
  for j *in sorted*(I, *by* $\mathbf{x}_{jk}$) **do** ← 遍历 k 特征的取值 (每个样本)
    $G_L \leftarrow G_L + g_j,\ H_L \leftarrow H_L + h_j$
    $G_R \leftarrow G - G_L,\ H_R \leftarrow H - H_L$
    $score \leftarrow \max(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$
  end
end
Output: Split with max score

pro: exact

con: computation cost

time complexity: $d \cdot n \log n$

优化: 对 feature / feature 取值进行筛选

{ compromise 精度
{ boost computation => 近似 Algo

② 近似 algo

a. feature selection

1、按树 rand. (split 前就 rand 出 subset of feature)

2、按层 rand (每次 split 都 rand 新的 subset of feature)

b. feature value selection

1、分桶

理想状态：assume all samples uniform distribute（出现 prob. 一样）

eg.

	X_1	Sample
bin 1	1	B D
	2	A E
bin 2	3	C
	4	H L
	5	I
bin 3	6	G
	7	M N
	8	T

样本：12

桶数：3

avg：4 samples / bin

result：split on $X_1=2$ and $X_1=5$

2、加权分位法

$$\sum_{i=1}^{N}\left(g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right) + \Omega(f_t)$$

$$= \sum_{i=1}^{N} \frac{1}{2} h_i \left(\boxed{\frac{g_i^2}{h_i^2}} + 2\frac{g_i}{h_i} f_t(x_i) + f_t^2(x_i) \right) + \Omega(f_t)$$

$$= \sum_{i=1}^{N} \frac{1}{2} h_i \left[f_t(x_i) - \left(-\frac{g_i}{h_i}\right)\right]^2 + \Omega(f_t)$$

常量，不影响目标优化

So 目标函数为真实值是 $-g_i/h_i$，权重为 h_i 的平方损失，因此，使用二阶梯度加权

eg.

feature x_1	sample	h_i	
1	B	3	bin1
	D	1	
2	A	2	
	E	2	
3	C	1	3
4	H	4	bin2
	L	1	
5	I	3	5
6	G	5	⋮
7	M	2	
	N	1	7
8	T	2	

$\Sigma h_i = 27$

eg. $\varepsilon = \frac{5}{27}$

hyperparam: $\varepsilon = \frac{n}{27}$ put as many samples in bin s.t. $\frac{\Sigma \text{weights}}{\Sigma h_i} \le \varepsilon$

加权分位法 分为
- 全局：对于每个 tree 的每个 feature，只进行一次分位
- 局部：每次 split 都对选中 feature 进行分位.

• 全局 above eg.

$x_1 \le 3$

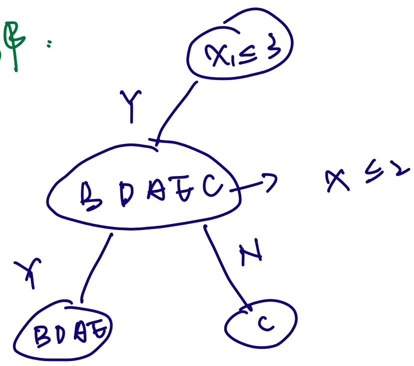
Y

B D A E C $\in$ bin 1

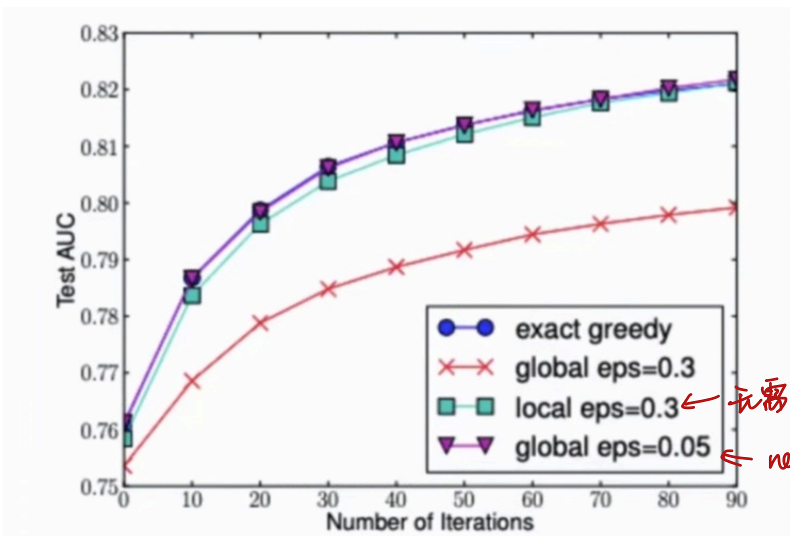
此时无法再对 x_1 进行分位，∵ 3, 5, 7 已固定

不够 flexible，往往需要 ε 更小（更多 bins） to perform well

· 局部：



more flexible，每次不需过多 bins



无需太多 bins

need ε small to perform well

·缺失值处理

eg.

feature x_1	sample
1	B D
2	A E
3	C
4	H L
5	J
6	G
7	M N
8	T
Nan	O P Q R S

可穷举，但 $O(N^2)$

论文采用方法：将 NaN 全体放到左 or 右，比较 $Gain_L$ and $Gain_R$，

if $Gain_L$ 大，则所有 NaN 放 left.

缺失值不参与排序和分位

· 学习率 Shrinkage

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i)$$

可防 overfitting，通常 $\eta = 0.1$