

Introduction to Xgboost

璧羽 2018/07/16

簡介

- ❖ Xgboost=eXtreme Gradient boost
- ❖ 發表者：陳天奇、Carlos Guestrin
- ❖ 核心概念：Decision Tree
- ❖ 目標是優化訓練速度、效率、可運算大樣本、處理稀疏數據
- ❖ Gradient Boosting Decision Tree (GBDT)延伸優化
- ❖ 以C++撰寫，再用A P I 提供給各軟體環境使用
- ❖ Xgboost 已經廣泛運用在工業領域

增量訓練 Additive Training

$$\hat{y}_i^{(0)} = 0$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \quad \text{新增加一顆樹}$$

...

保留原模型

$$\hat{y}_i^{(t)} = \sum_{k=1}^t f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i) \quad \text{新增加第t棵樹}$$

保留前t-1棵樹的模型

目標函式

$$\mathcal{L}(\phi) = \sum_i \underbrace{l(\hat{y}_i, y_i)}_{\text{loss}} + \sum_k \underbrace{\Omega(f_k)}_{\text{懲罰項(模型複雜度)}} \quad (2)$$

$$\text{where } \Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2$$

$$Obj(\Theta) = L(\Theta) + \Omega(\Theta)$$

Training Loss measures how well model fit on training data


Regularization, measures complexity of model

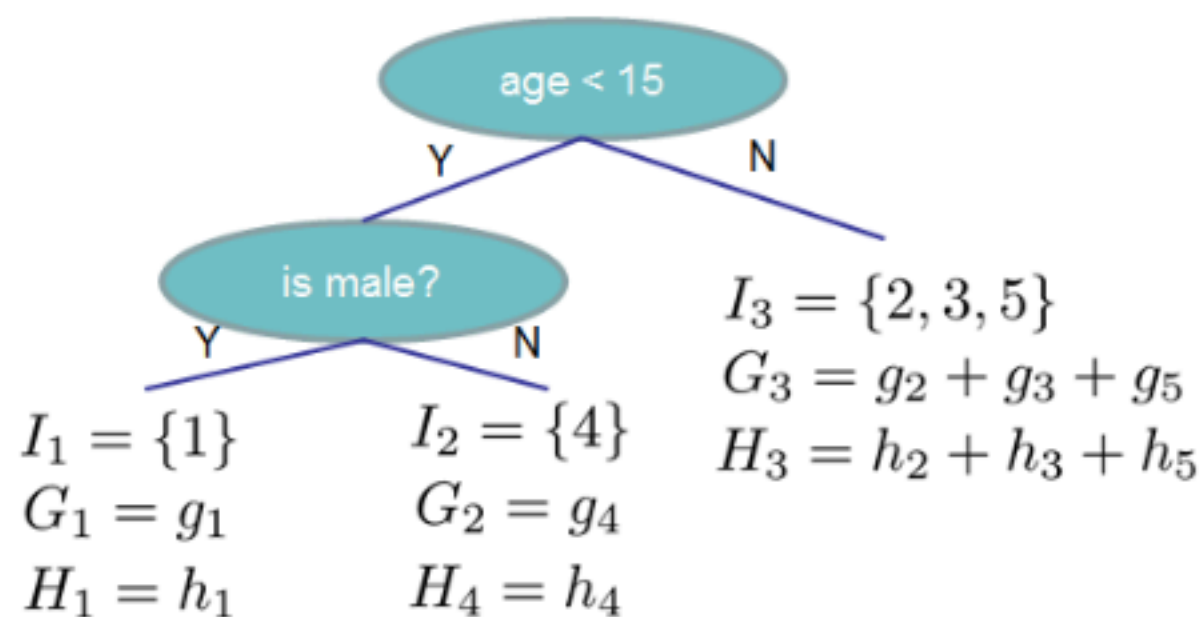
- ❖ 從錯誤中學習
- ❖ 準確度與樹的複雜度取得平衡
- ❖ 建立 N 個樹.....

评分

vi. 打分函数计算举例

Obj代表了当我们指定一个树的结构的时候，我们在目标上面最多减少多少。我们可以把它叫做结构分数 (structure score)。你可以认为这个就是类似吉尼系数一样更加一般的对于树结构进行打分的函数。下面是一个具体的打分函数计算的例子

样本号	梯度数据
1 	g_1, h_1
2 	g_2, h_2
3 	g_3, h_3
4 	g_4, h_4
5 	g_5, h_5



$$Obj = -\frac{1}{2} \sum_j \frac{G_j^2}{H_j + \lambda} + 3\gamma$$

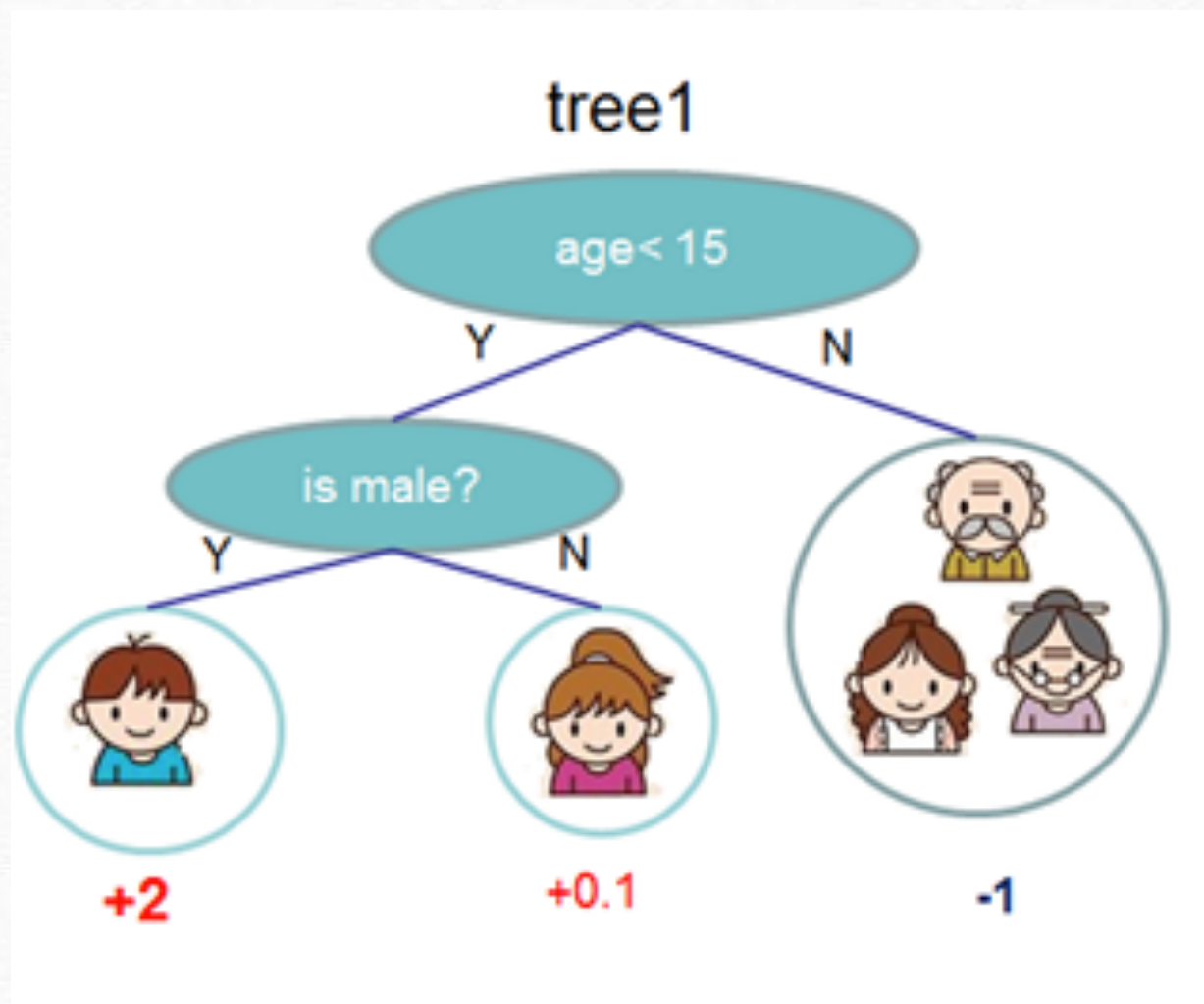
这个分数越小，代表这个树的结构越好

判斷是否要繼續切下去

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

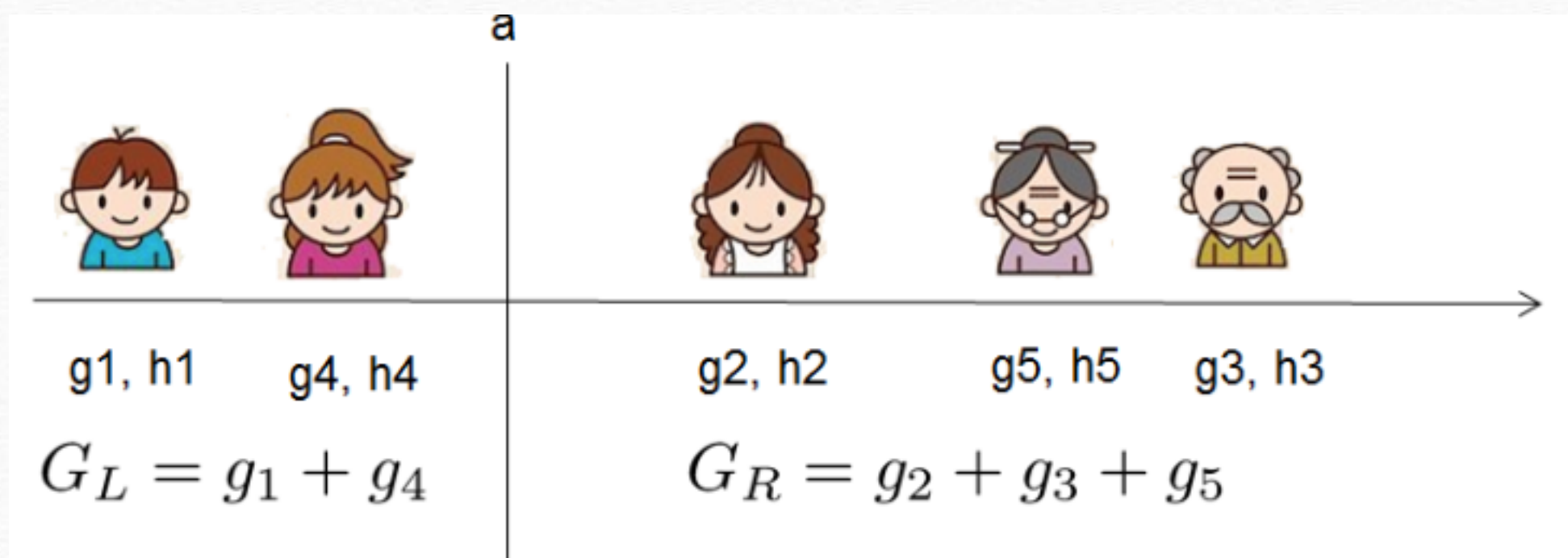
左子树分数 右子树分数 不分割我们可以拿到的分数 加入新叶子节点引入的复杂度代价

www.52cs.org



- exact greedy algorithm
- （貪心算法獲取最優切分點）列舉所有可能的情形

判斷最佳切點



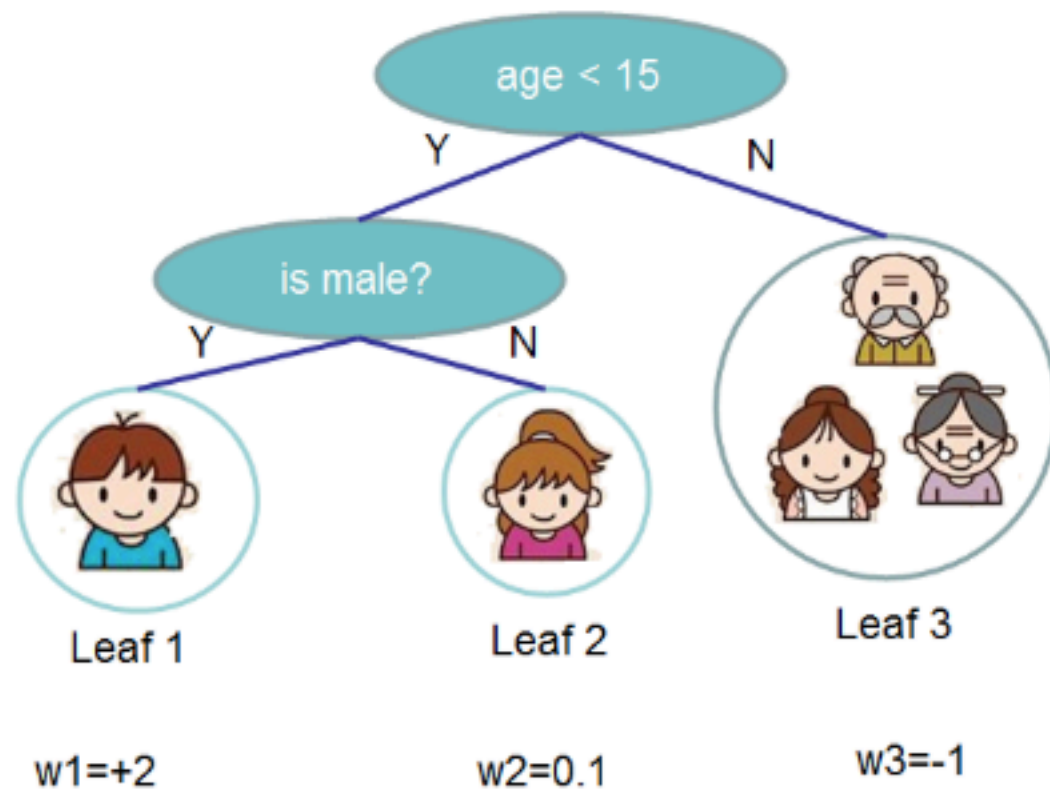
GBDT vs Xgboost

- ❖ GBDT 只有一階泰勒式、XGBoost用到二階泰勒式
- ❖ 正則項的運用 (Regularization)
- ❖ Feature預排序、Feature平行運算

複雜度公式

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

叶子的个数 w 的L2模平方



$$\Omega = \gamma 3 + \frac{1}{2} \lambda (4 + 0.01 + 1)$$

其他小補充

- 避免 overfitting 的兩個小設計
 - (1) shrinkage：在每一步的加強訓練之後，增加一點 weighted，有點像是 learning rate 的意味。
 - (2) column (feature) subsampling：這個技巧在 RF 當中也有使用到，同時也有助於加快訓練速度。

建議閱讀

- ❖ 前半段介紹 G B
- ❖ <https://blog.csdn.net/u011094454/article/details/78948989>
- ❖ 先看完這三篇
- ❖ <https://hk.saowen.com/a/e997166f37dc6022138607838ec7c83ba6f89b2d5d11fe248e0925968b410f33>
- ❖ <https://hk.saowen.com/a/7214d5cc99d98d81736f766d77cd568dae07aadf85f027a1e5acdd57839e7f91>
- ❖ <http://www.52cs.org/?p=429>
- ❖ 最後再看這篇
- ❖ <https://medium.com/@cyeninesky3/xgboost-a-scalable-tree-boosting-system-%E8%AB%96%E6%96%87%E7%AD%86%E8%A8%98%E8%88%87%E5%AF%A6%E4%BD%9C-2b3291e0d1fe>
- ❖ 論文原文：<https://arxiv.org/pdf/1603.02754v1.pdf>
- ❖ 作者 P P T：<https://homes.cs.washington.edu/~tqchen/pdf/BoostedTree.pdf>
- ❖