XGBoost

各类赛事排名第一的算法;

XGBoost: A Scalable Tree Boosting System

Tianqi Chen
University of Washington
tqchen@cs.washington.edu

Carlos Guestrin
University of Washington
guestrin@cs.washington.edu

ABSTRACT

Tree boosting is a highly effective and widely used machine learning method. In this paper, we describe a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. We propose a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning. More importantly, we provide insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, XGBoost scales beyond billions of examples using far fewer resources than existing systems.

problems. Besides being used as a stand-alone predictor, it is also incorporated into real-world production pipelines for ad click through rate prediction [15]. Finally, it is the de-facto choice of ensemble method and is used in challenges such as the Netflix prize [3].

In this paper, we describe XGBoost, a scalable machine learning system for tree boosting. The system is available as an open source package[2]. The impact of the system has been widely recognized in a number of machine learning and data mining challenges. Take the challenges hosted by the machine learning competition site Kaggle for example. Among the 29 challenge winning solutions [3] published at Kaggle's blog during 2015, 17 solutions used XGBoost. Among these solutions, eight solely used XGBoost to train the mod-

贪心科技 | 让每个人享受个性化教育服务

Bagging vs Boosting

Ensemble

PCAC Learn

Bagging ⇒ list of experts

Leverages unstable base learners that are weak because of overfitting

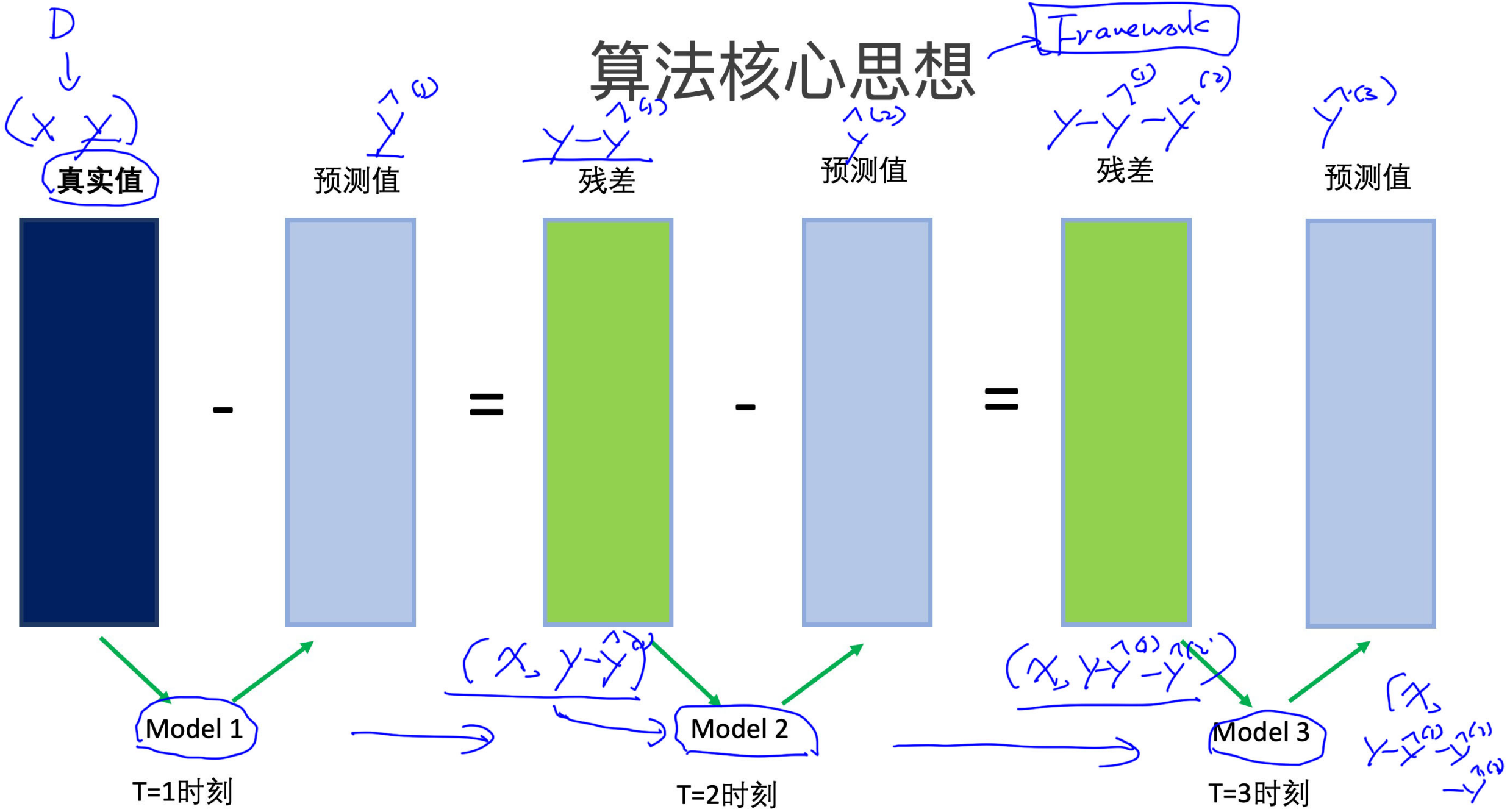
Boosting ⇒ list of weak learner ⇒ predict > random guess

Boosting: Leverage stable base learners that are weak because of underfitting

Linear Regression

Gradient Boosting

算法核心思想

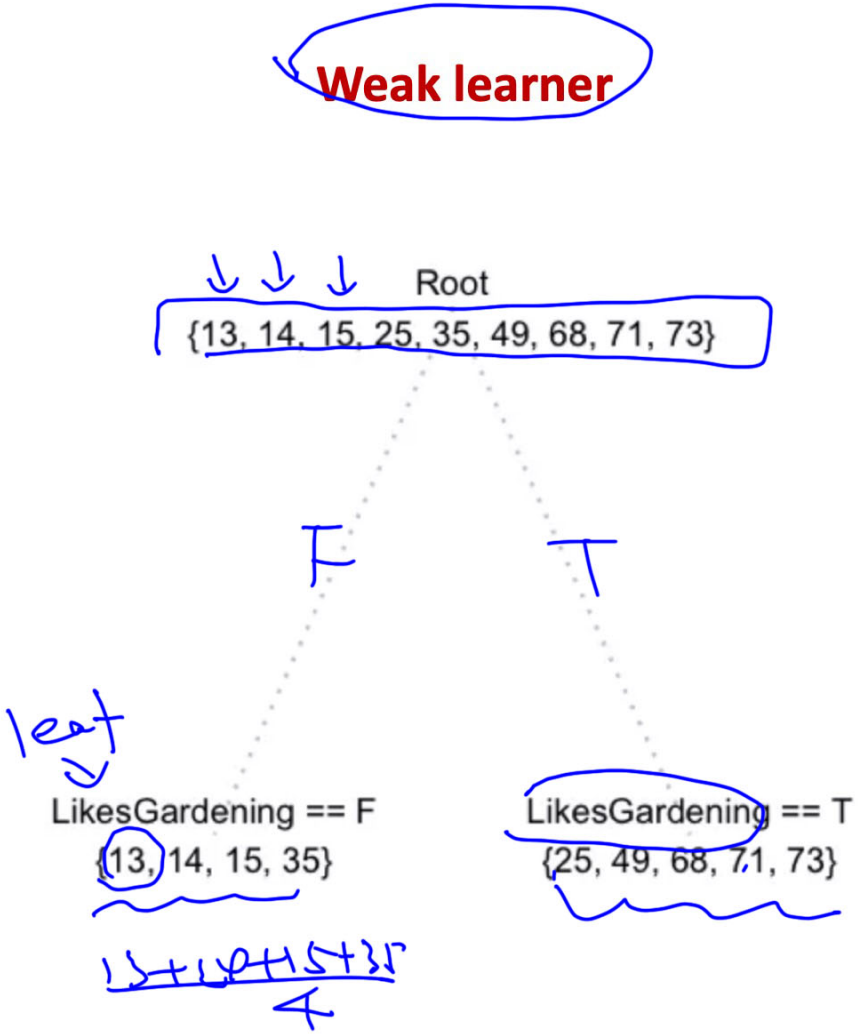

Framework
D
(x y)
真实值
预测值
残差
预测值
残差
预测值
-
=
-
=
Model 1
Model 2
Model 3
T=1时刻
T=2时刻
T=3时刻

例子：预测年龄

PersonID	Age	LikesGardening	PlaysVideoGames	LikesHats
1	13	FALSE	TRUE	TRUE
2	14	FALSE	TRUE	FALSE
3	15	FALSE	TRUE	FALSE
4	25	TRUE	TRUE	TRUE
5	35	FALSE	TRUE	TRUE
6	49	TRUE	FALSE	FALSE
7	68	TRUE	TRUE	TRUE
8	71	TRUE	FALSE	FALSE
9	73	TRUE	FALSE	TRUE

例子：预测年龄（续）

下一颗决策树需要拟合这个数



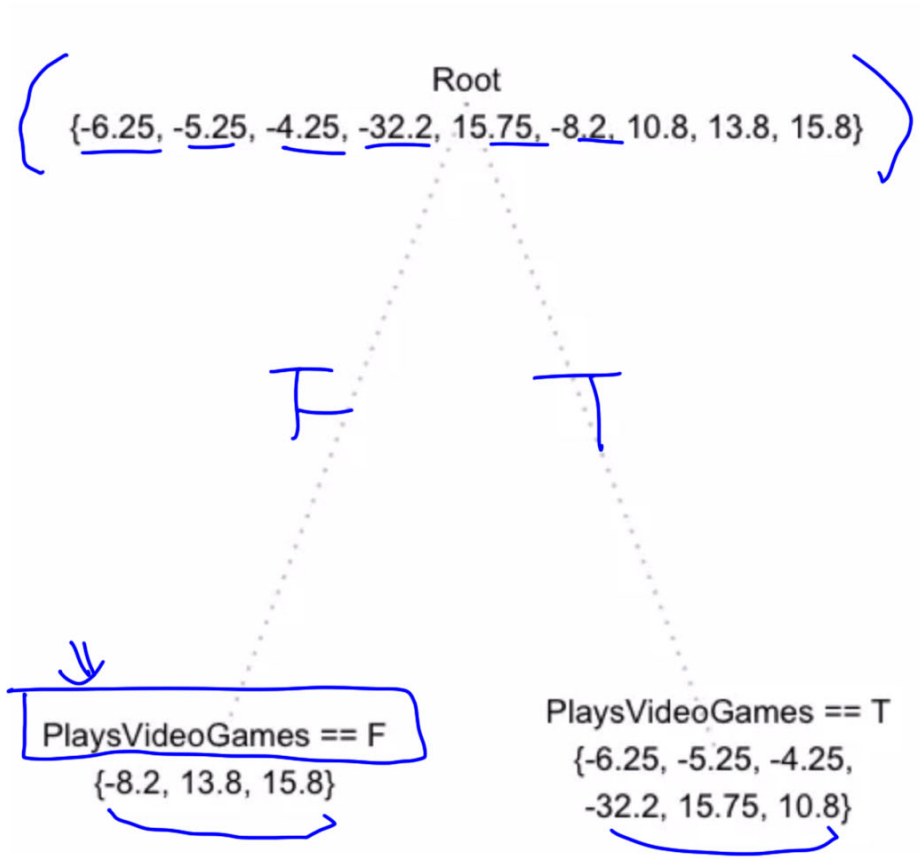
PersonID	Age	Tree1 Prediction	Tree1 Residual
1	13	19.25	-6.25
2	14	19.25	-5.25
3	15	19.25	-4.25
4	25	57.2	-32.2
5	35	19.25	15.75
6	49	57.2	-8.2
7	68	57.2	10.8
8	71	57.2	13.8
9	73	57.2	15.8

例子：预测年龄（续）

上一次的残差

第二颗决策树

PersonID	Age	Tree1 Prediction	Tree1 Residual
1	13	19.25	-6.25
2	14	19.25	-5.25
3	15	19.25	-4.25
4	25	57.2	-32.2
5	35	19.25	15.75
6	49	57.2	-8.2
7	68	57.2	10.8
8	71	57.2	13.8
9	73	57.2	15.8



例子：预测年龄（续）

PersonID	Age	Tree1 Prediction	Tree1 Residual	Tree2 Prediction	Combined Prediction	Final Residual
1	13	19.25	-6.25	-3.567	15.68	2.683
2	14	19.25	-5.25	-3.567	15.68	1.683
3	15	19.25	-4.25	-3.567	15.68	0.6833
4	25	57.2	-32.2	-3.567	53.63	28.63
5	35	19.25	15.75	-3.567	15.68	-19.32
6	49	57.2	-8.2	7.133	64.33	15.33
7	68	57.2	10.8	-3.567	53.63	-14.37
8	71	57.2	13.8	7.133	64.33	-6.667
9	73	57.2	15.8	7.133	64.33	-8.667

Tree1 SSE	Combined SSE
1994	1765

Looking into details of XGBoost ...

本节很多的PPT来自于Tianqi: https://homes.cs.washington.edu/~tqchen/pdf/BoostedTree.pdf

Why XGBoost is so popular?

Speed and Performance

Core algorithm is parallelizable

Consistently outperforms other algorithm methods

Wide variety of tuning parameters (Hyperparameters)

学习路径...

$(Xw-y)^2$

Taylor Expansion

① 如何构造目标函数 ⇒ Objective function

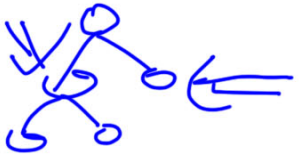
② 如何近似目标函数 Objective function'

优化（求最小值 大）

如何引入树到目标函数，改造目标函数

拉手的函数

如何使用贪心算法构造树

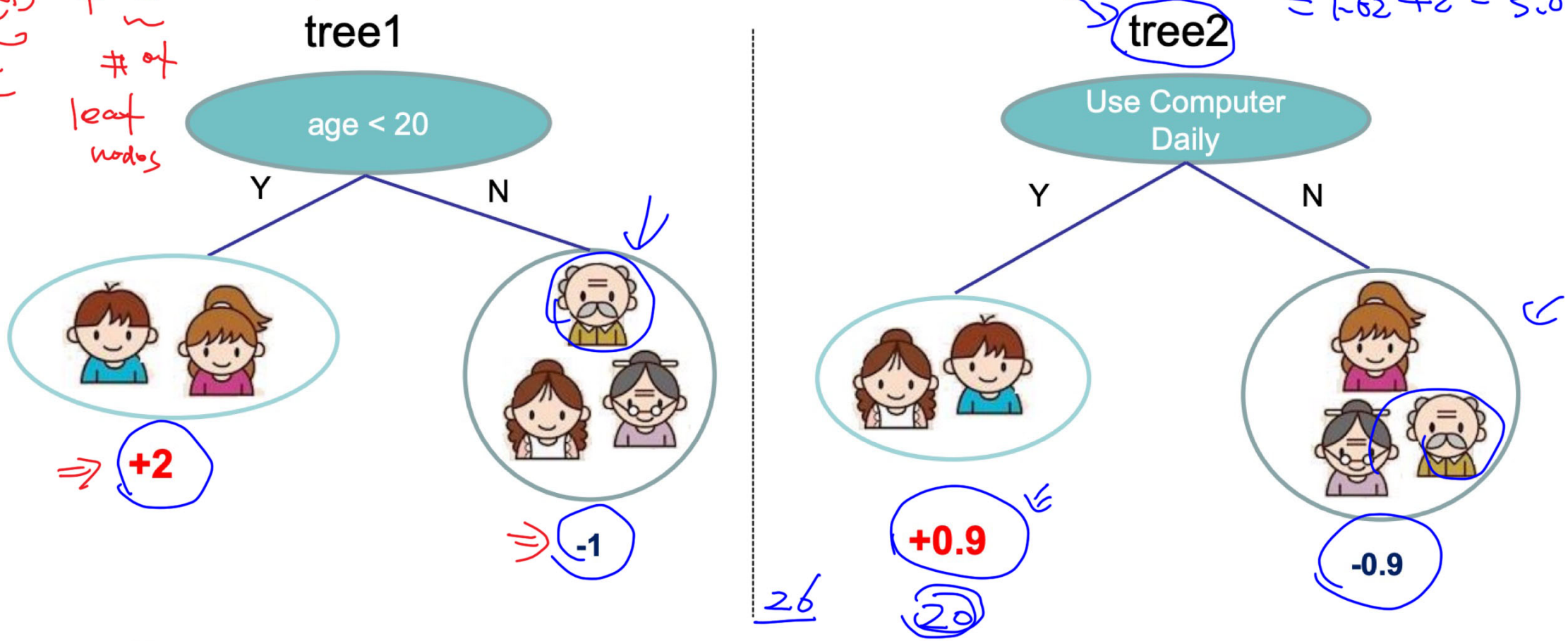


structural data

Ensemble of Trees

$\Omega(\text{tree}_1) = \underbrace{2^2 + (-1)^2}_{\text{权重}} + \underbrace{2}_{\#\text{ of leaf nodes}} = 7$

$0.8^2 + 0.8^2 + 2 = 1.62 + 2 = 3.62$



f( ) = 2 + 0.9= 2.9

f( )= -1 - 0.9= -1.9

使用多棵树预测

x_i

$f_1(x_i) =$

$f_2(x_i) =$

$f_3(x_i) =$

假设已经训练了K颗树，对于第i个样本的预测值为：

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F}$$

Space of functions containing all Regression trees

目标函数

假设有K颗树：预测过程

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F}$$

- Depth ↙
- leaf-node ⇒ w ↓
- 人工 # of leaf nodes

回归：$(y_i - \hat{y}_i)^2$

二分类：cross-entropy

目标函数：

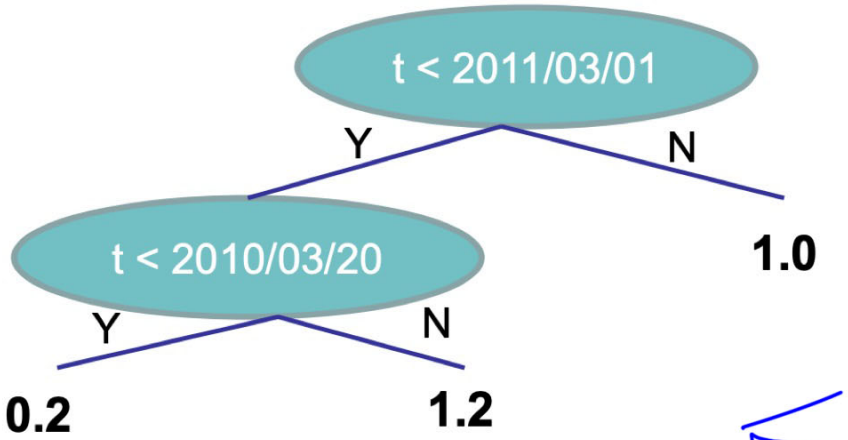
$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Training loss　　Complexity of the Trees

真　预

MSE　RMSE　　Regularization

The model is regression tree that splits on time

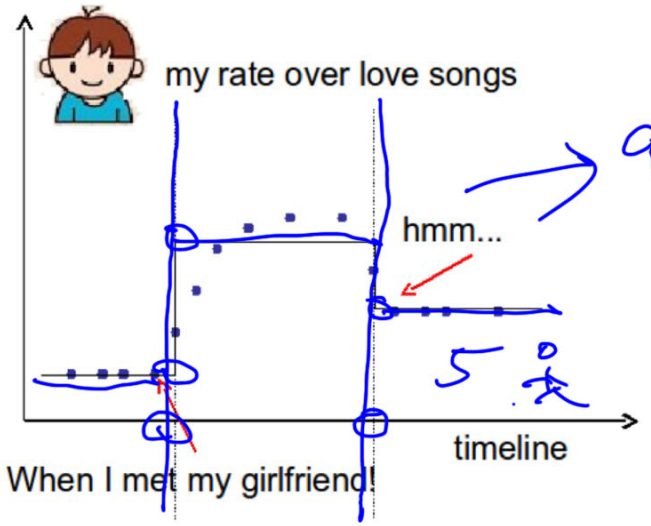


Tree

Equivalently

Parameters

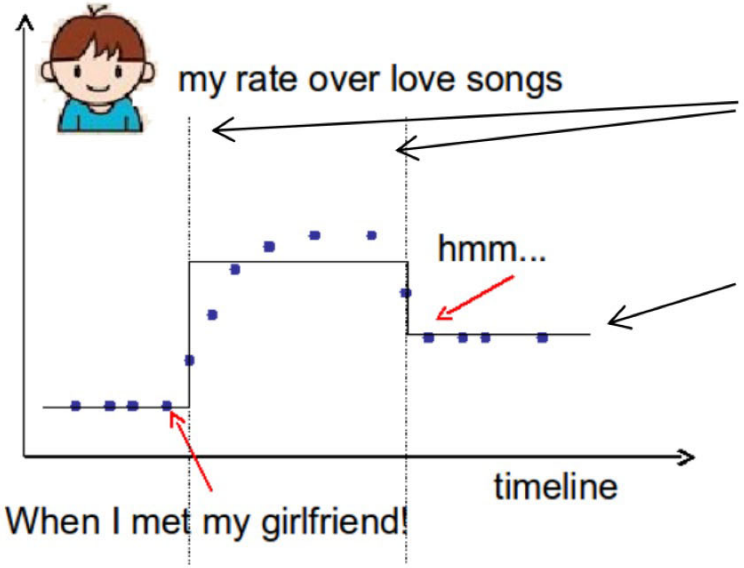
Piecewise step function over time



$q(x)$

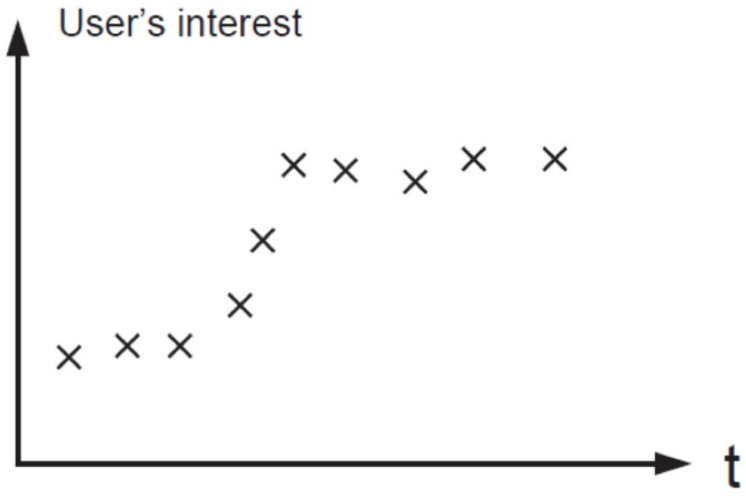
$q(\text{[illegible]})=5$

5 [illegible]

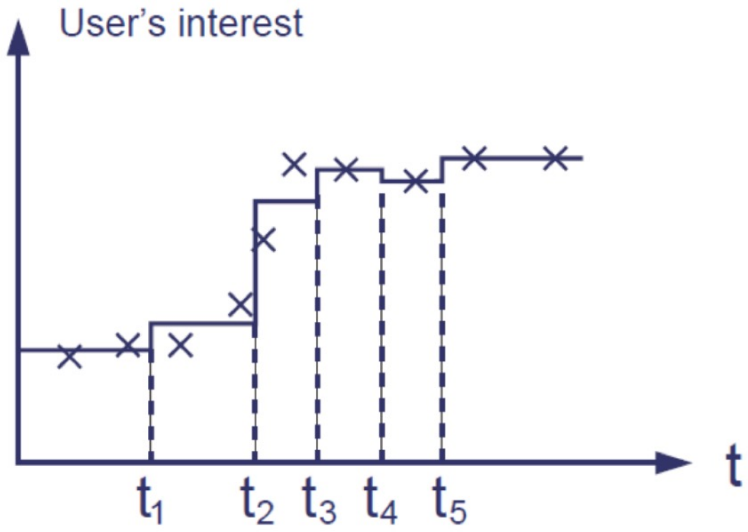


Splitting Positions

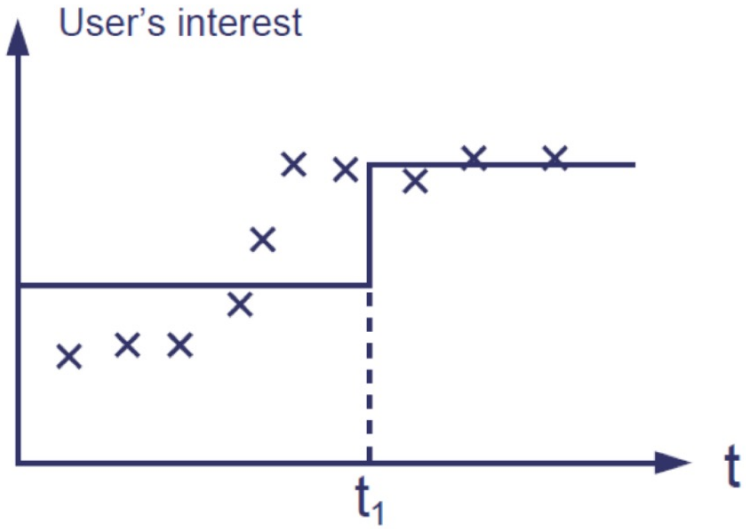
The Height in each segment



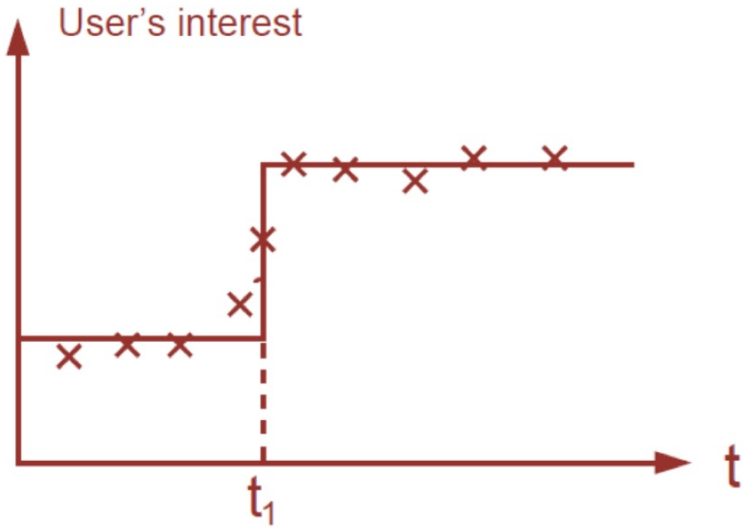
Observed user's interest on topic k against time t



✗ Too many splits, $\Omega(f)$ is high



✗ Wrong split point, $L(f)$ is high



✓ Good balance of $\Omega(f)$ and $L(f)$

目标函数

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Training loss　　Complexity of the Trees

Using Square loss $l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$

- Will results in common gradient boosted machine

Using Logistic loss $l(y_i, \hat{y}_i) = y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i) \ln(1 + e^{\hat{y}_i})$

- Will results in LogitBoost

如何去训练模型

Additive Training

combined predictor

$$
\begin{aligned}
1 \quad \hat{y}_i^{(0)} &= 0 \quad \text{---> Base case} \\
2 \quad \hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + \boxed{f_1(x_i)} \\
3 \quad \hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\dots \\
t \quad \hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}
$$

y_i

y_i

New function

Model at training round t

$(1 \dots t-1)$

Keep functions added in previous round

Additive Training

- How do we decide which f to add?
  - Optimize the objective!!
- The prediction at round t is $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$

This is what we need to decide in round t

$$Obj^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i)$$
$$= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant$$

Goal: find f_t to minimize this

- Consider square loss

$$Obj^{(t)} = \sum_{i=1}^{n} \left(y_i - (\hat{y}_i^{(t-1)} + f_t(x_i))\right)^2 + \Omega(f_t) + \cancel{const}$$
$$= \sum_{i=1}^{n} \left[2(\hat{y}_i^{(t-1)} - y_i) f_t(x_i) + f_t(x_i)^2\right] + \Omega(f_t) + \cancel{const}$$

This is usually called residual from previous round

使用泰勒级数近似目标函数

1到t-1 预测之和　当前模型预测　complexity　正则

- Goal $Obj^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant$
  - Seems still complicated except for the case of square loss
- Take Taylor expansion of the objective
  - Recall $f(x+\Delta x) \simeq f(x) + f'(x)\Delta x + \frac{1}{2} f''(x)\Delta x^2$　只保留二阶
  - Define $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$, $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$

$$Obj^{(t)} \simeq \sum_{i=1}^{n} \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) + constant$$

$f(x) = l(y_i, \hat{y}_i^{(t-1)})$　$f'(x) = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$

$f(x+\Delta x) = l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i))$

得到的新的目标函数

minimize

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

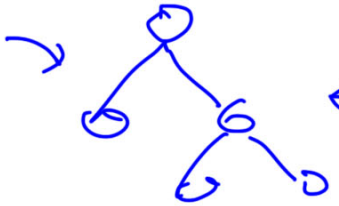
where $g_i=\partial_{\hat{y}^{(t-1)}} l(y_i,\hat{y}^{(t-1)}),\quad h_i=\partial^2_{\hat{y}^{(t-1)}} l(y_i,\hat{y}^{(t-1)})$

真实

前t-1之和

已知

g, h 是很容易计算的，但f(x)如何去表示呢？ 因为它是一棵树，如果表示成函数的形式呢?



$\longleftrightarrow f_t(x)$

重新定义一棵树

Tree → Parameterize

w

- We define tree by a vector of scores in leafs, and a leaf index mapping function that maps an instance to a leaf

T: # of leaf nodes

$$\Rightarrow \boxed{f_t(x) = w_{q(x)}}, \quad w \in \mathbf{R}^T, q: \mathbf{R}^d \to \{1, 2, \cdots T\}$$

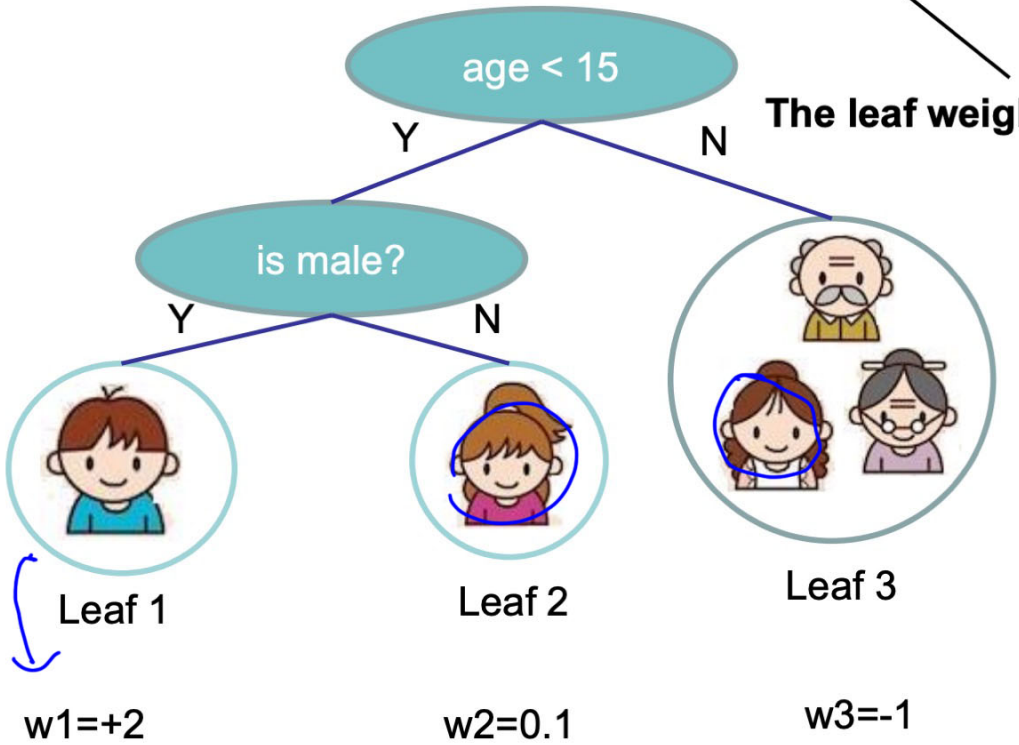
参数: T, $w \in R^T$, q

The structure of the tree

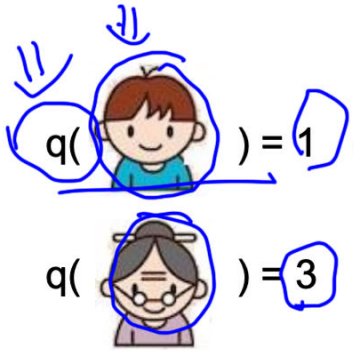
The leaf weight of the tree

if x = boy

$w_{q(x)} = 1$



q T=3



拟合点,
拟合值

树的复杂度

PAC learning

- Define complexity as (this is not the only possible definition)

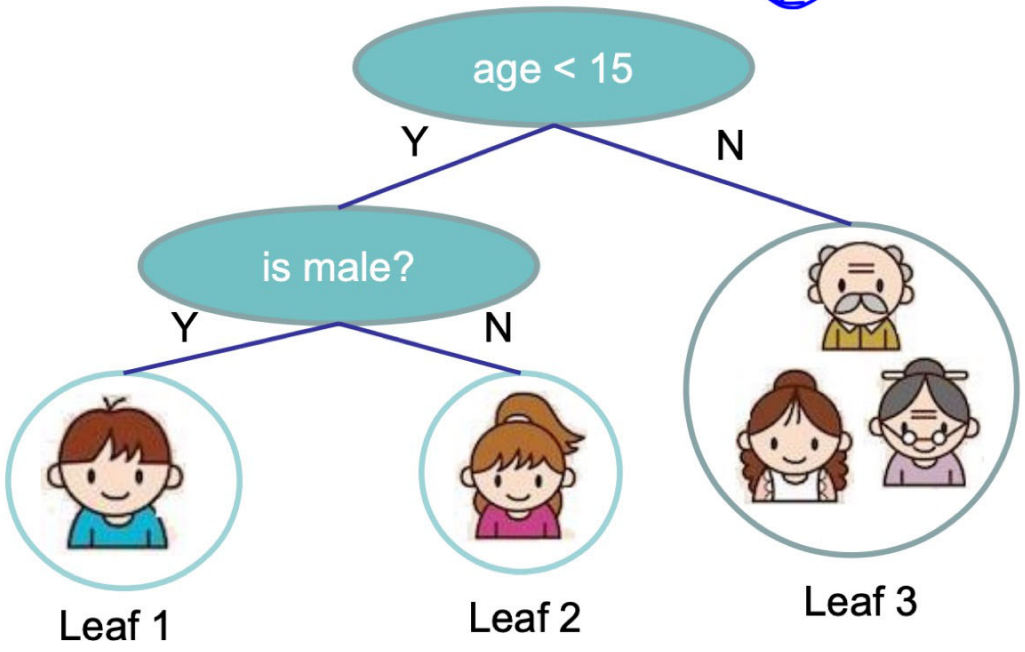
$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

L2范数 VC Dimension

Number of leaves ①

L2 norm of leaf scores ②

SVM



w1=+2 w2=0.1 w3=-1

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

2^2 0.1^2 $(-1)^2$

新的目标函数

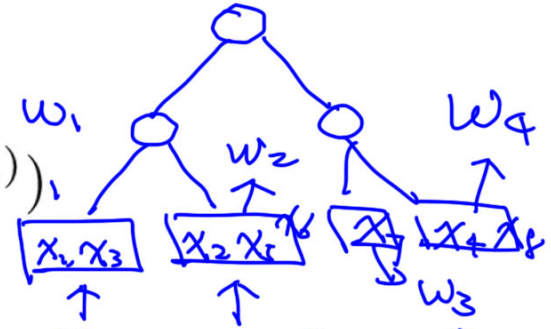
$f_t(x_i) = w_{q(x_i)}$

- Objective, with constants removed

OLD $\Rightarrow$

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

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}), \quad h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$$



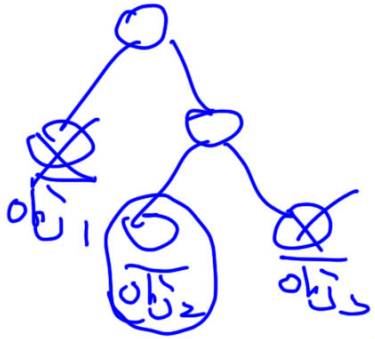
$I_2 = \{2, 5, 6\}$

$I_3 = \{7\}$

- Define the instance set in leaf j as $I_j = \{i | q(x_i) = j\}$

$I_1 \; I_2$

  - Regroup the objective by leaf



$$Obj^{(t)} \simeq \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

$$= \sum_{i=1}^{n} \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2 \right] + \gamma T + \lambda \frac{1}{2} \sum_{j=1}^{T} w_j^2$$

$$\Leftrightarrow = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T$$

① ② $x_1 \; x_2 \cdots x_8$

③ $[x_1, x_3]$ $[x_2 x_5 x_6]$ $[x_7]$ $[x_4 x_8]$

$\sum_{j=1}^{4}$

$obj_t = obj_1 + obj_2 + obj_3$

- This is sum of T independent quadratic function

新的目标函数（续）

$f(x)=ax^2+bx+c \Leftarrow$ 最小值

$x=-\frac{b}{2a}$ $f(x)=$

- Two facts about single variable quadratic function

$x=-\frac{G}{H}$

$$\Rightarrow \quad argmin_x \; Gx + \frac{1}{2}Hx^2 = -\frac{G}{H}, \quad H > 0 \qquad \min_x \; Gx + \frac{1}{2}Hx^2 = -\frac{1}{2}\frac{G^2}{H}$$

- Let us define $\boxed{G_j = \sum_{i \in I_j} g_i}$ $\boxed{H_j = \sum_{i \in I_j} h_i}$

$$\begin{aligned} Obj^{(t)} &= \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i\right) w_j + \frac{1}{2}\left(\sum_{i \in I_j} h_i + \lambda\right) w_j^2 \right] + \gamma T \\ &= \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2 \right] + \gamma T \end{aligned}$$

$\Leftrightarrow$ 新的目标函数

$\frac{G_j^2}{H_j+\lambda}$

- Assume the structure of tree (q(x)) is fixed, the optimal weight in each leaf, and the resulting objective value are

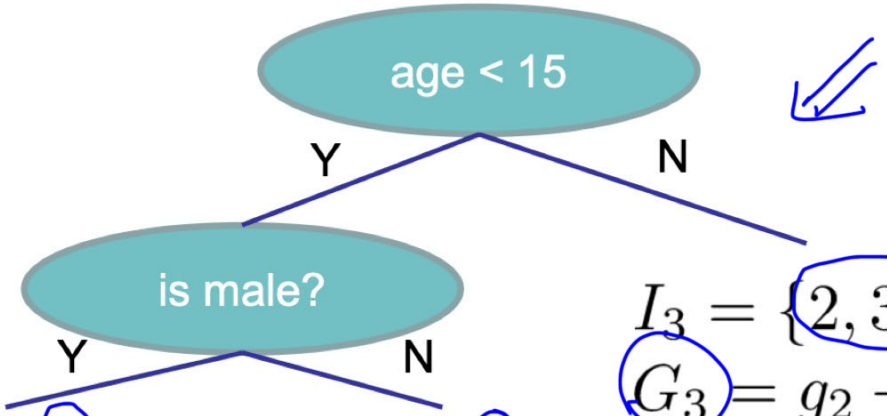
$$w_j^* = -\frac{G_j}{H_j+\lambda} \qquad Obj = -\frac{1}{2}\sum_{j=1}^{T} \frac{G_j^2}{H_j+\lambda} + \gamma T$$

$\Leftarrow$ 树的形状已知情况下的最小值

This measures how good a tree structure is!

计算Score

Instance index	gradient statistics
1	g1, h1
2	g2, h2
3	g3, h3
4	g4, h4
5	g5, h5



$I_1 = \{1\}$
$G_1 = g_1$
$H_1 = h_1$

$I_2 = \{4\}$
$G_2 = g_4$
$H_4 = h_4$

$I_3 = \{2, 3, 5\}$
$G_3 = g_2 + g_3 + g_5$
$H_3 = h_2 + h_3 + h_5$

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

The smaller the score is, the better the structure is

如何寻找树的形状?

Brute Force Solution： ×

Greedy Solution

A ① 特征, ⇒ Information gain

B C (H(A) → H(B) + H(C))

- Enumerate the possible tree structures q
- Calculate the structure score for the q, using the scoring eq.

$$Obj = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda} + \gamma T$$

Obj(A) → Obj(B) + Obj(C)

- Find the best tree structure, and use the optimal leaf weight

$$w_j^* = -\frac{G_j}{H_j+\lambda}$$

如何寻找树的形状?

贪心算法 (DT 算法) → Information Gain

- In practice, we grow the tree greedily
  - Start from tree with depth 0
  - For each leaf node of the tree, try to add a split. The change of objective after adding the split is

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

The complexity cost by introducing additional leaf

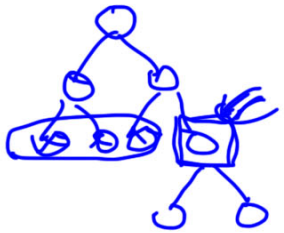
the score of left child

the score of right child

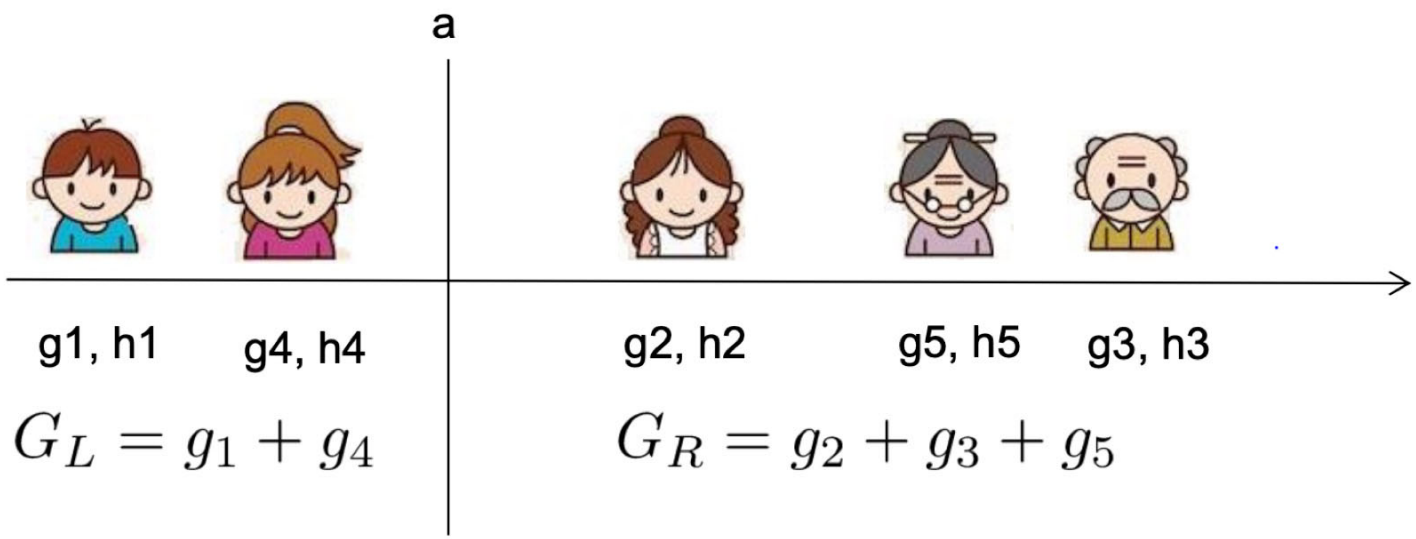
the score of if we do not split

  - Remaining question: how do we find the best split?

寻找最好的Split



- What is the gain of a split rule $x_j < a$? Say x_j is age



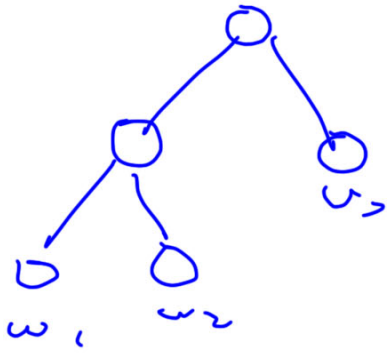
- All we need is sum of g and h in each side, and calculate

$$Gain = \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda} - \gamma$$

- Left to right linear scan over sorted instance is enough to decide the best split along the feature

Note

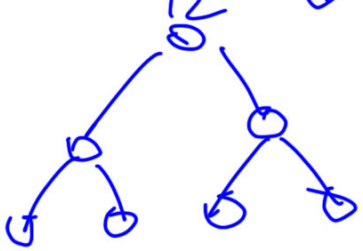
T_1



$\Downarrow$

$obj_1 = 10$

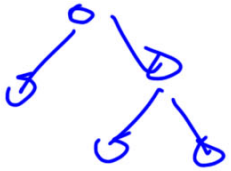
T_2



$\Downarrow$

$obj_2 = 9$

T_3



$\Downarrow$

$obj_3 = 11$

列出来所有可能的树

$T_1, T_2 - T_{56}$ T_{99999}

$\downarrow$ $\downarrow$ $\downarrow$

$(obj_1$ obj_2 $obj_{99999})$

$\downarrow$

minimum (56)

Question: 树的形状?

minimum objective

Note

真实值

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

x_2 $\quad$ x_i

$$\begin{array}{ccccccc} f_1 & & f_2 & & \cdots & & f_{t-1} \\ \hat{y}_1^{(1)} & + & \hat{y}_1^{(2)} & + & \cdots & + & \hat{y}_1^{(t-1)} \\ \hat{y}_2^{(1)} & + & \hat{y}_2^{(2)} & + & \cdots & + & \hat{y}_2^{(t-1)} \\ \vdots & & \vdots & & & & \vdots \\ \hat{y}_N^{(1)} & & \hat{y}_N^{(2)} & & & & \hat{y}_N^{(t-1)} \end{array}$$

已经学出来的

$$\begin{bmatrix} y_1^{t-1} \\ y_2^{t-1} \\ y_3^{t-1} \\ \vdots \\ y_N^{t-1} \end{bmatrix}$$

之前知

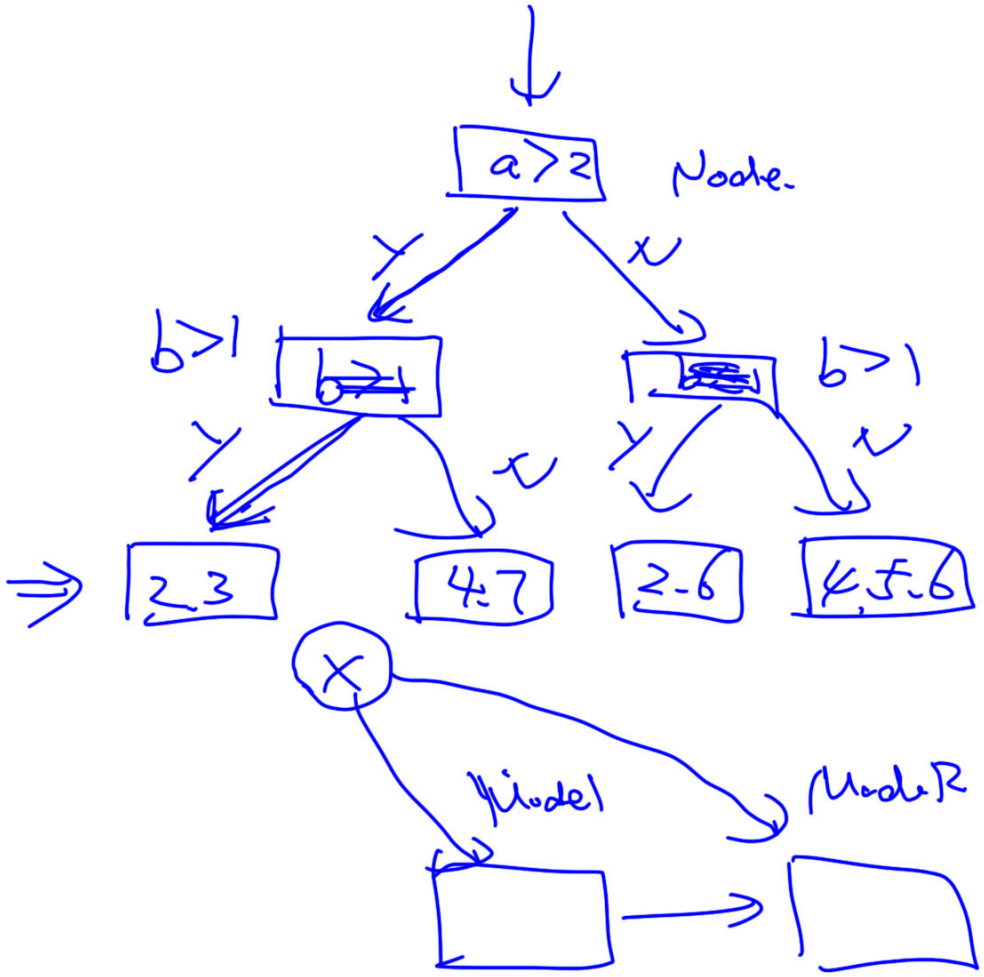
上个

f_t

$$\text{objective} = \underbrace{\sum_{i=1}^{N} \left(y_i - \left(\boxed{y_i^{t-1}} + \boxed{f_t(x_i)} \right) \right)}_{\text{Training Loss}} + \sum_{i=1}^{t} \Omega(f_i)$$

$1, 2, \ldots, t$

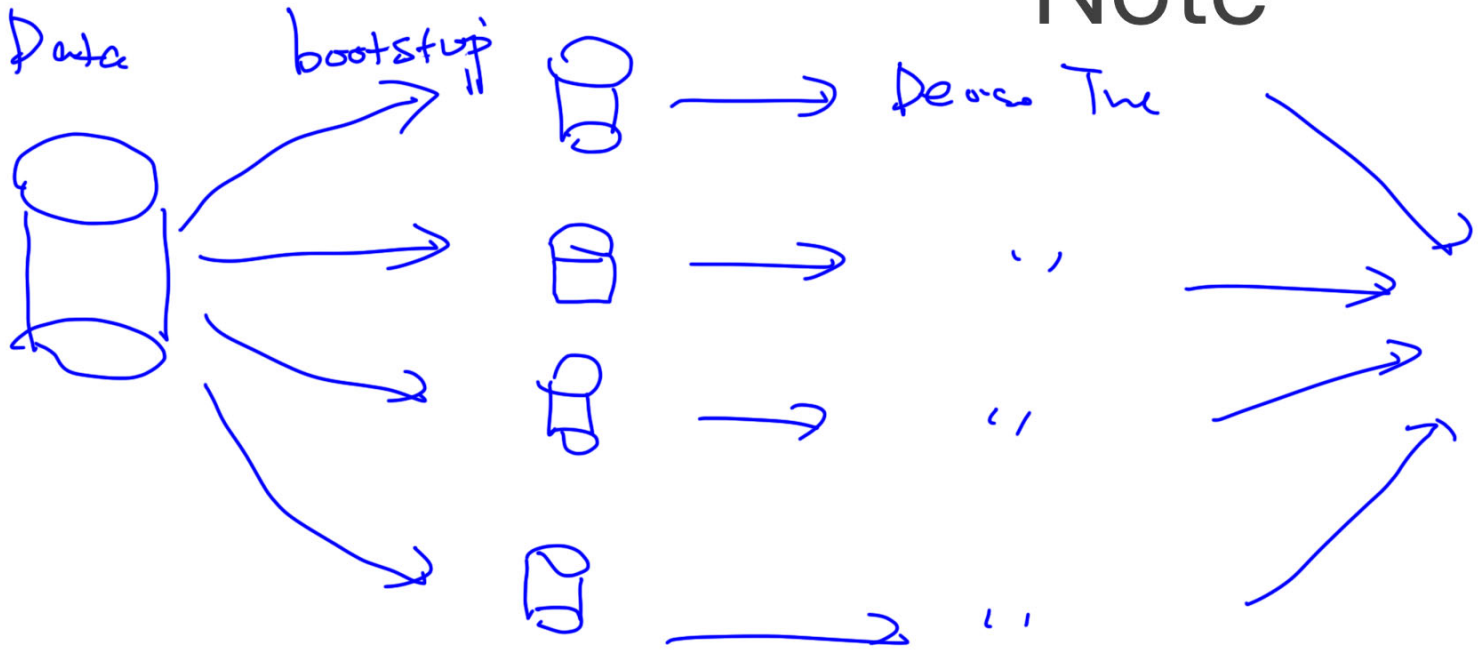
Note



$\hat{x} = (a=3, b=2)$

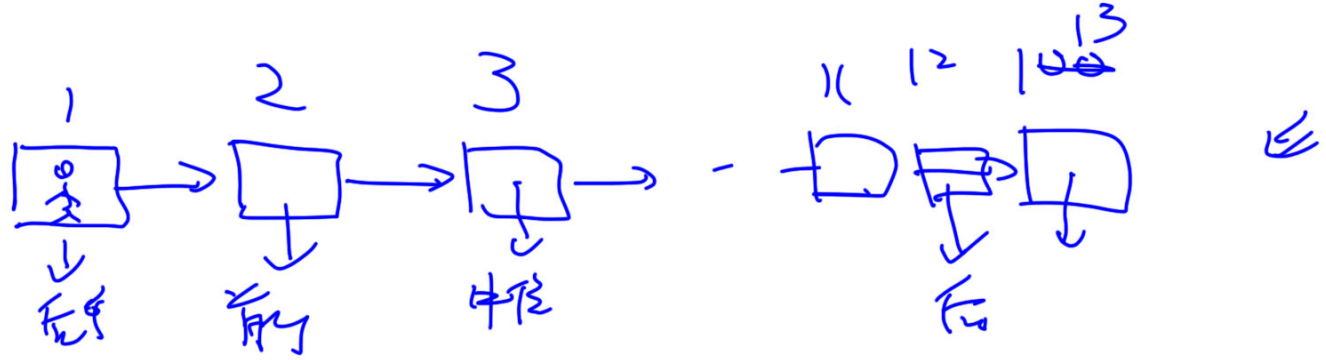
$\hat{y} = \frac{2+3}{2} = 2.5$

Note



随机森林

Boosting:



Note

$X_1 \sim N(\mu_1, \sigma_1^2)$

$X_2 \sim N(\mu_2, \sigma_2^2)$

$X_1 + X_2 \sim N(\mu_1 + \mu_2, \bigcirc)$