模型集群方法

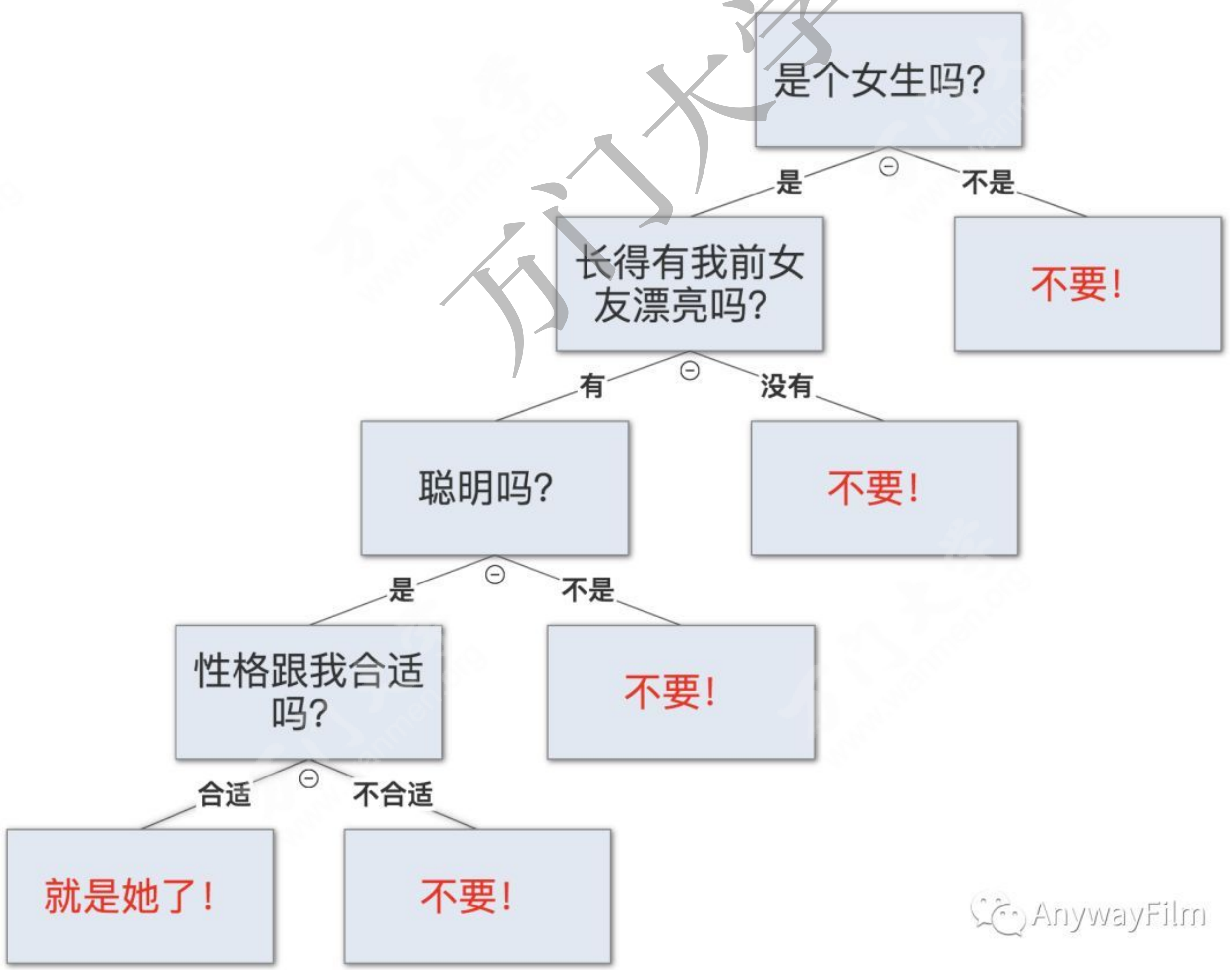
Bagging 方法

Boosting方法

Stacking方法

CRUISER

复习决策树

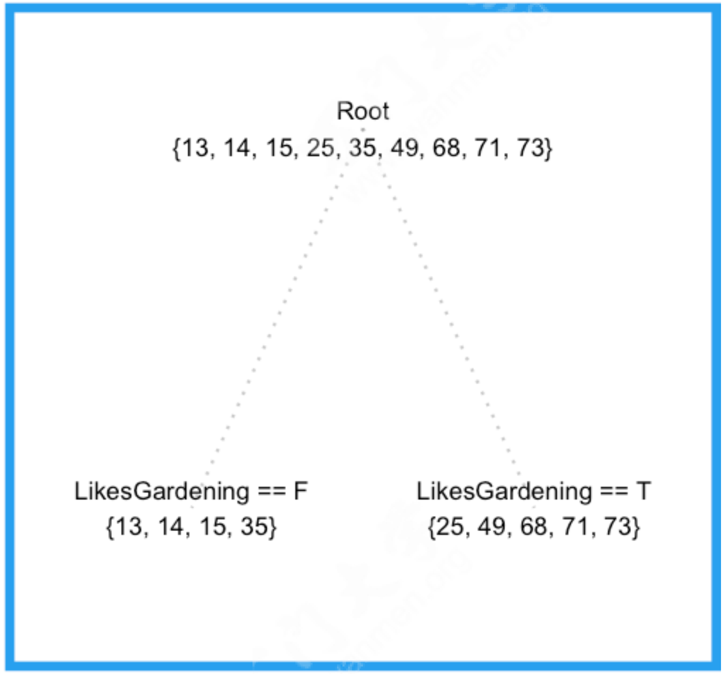


PersonID	Age	LikesGardening	PlaysVideoGames	LikesHats
1	13	FALSE	TRUE	TRUE
2	14	FALSE	TRUE	FALSE
3	15	FALSE	TRUE	FALSE
4	25	TRUE	TRUE	TRUE
5	35	FALSE	TRUE	TRUE
6	49	TRUE	FALSE	FALSE
7	68	TRUE	TRUE	TRUE
8	71	TRUE	FALSE	FALSE
9	73	TRUE	FALSE	TRUE

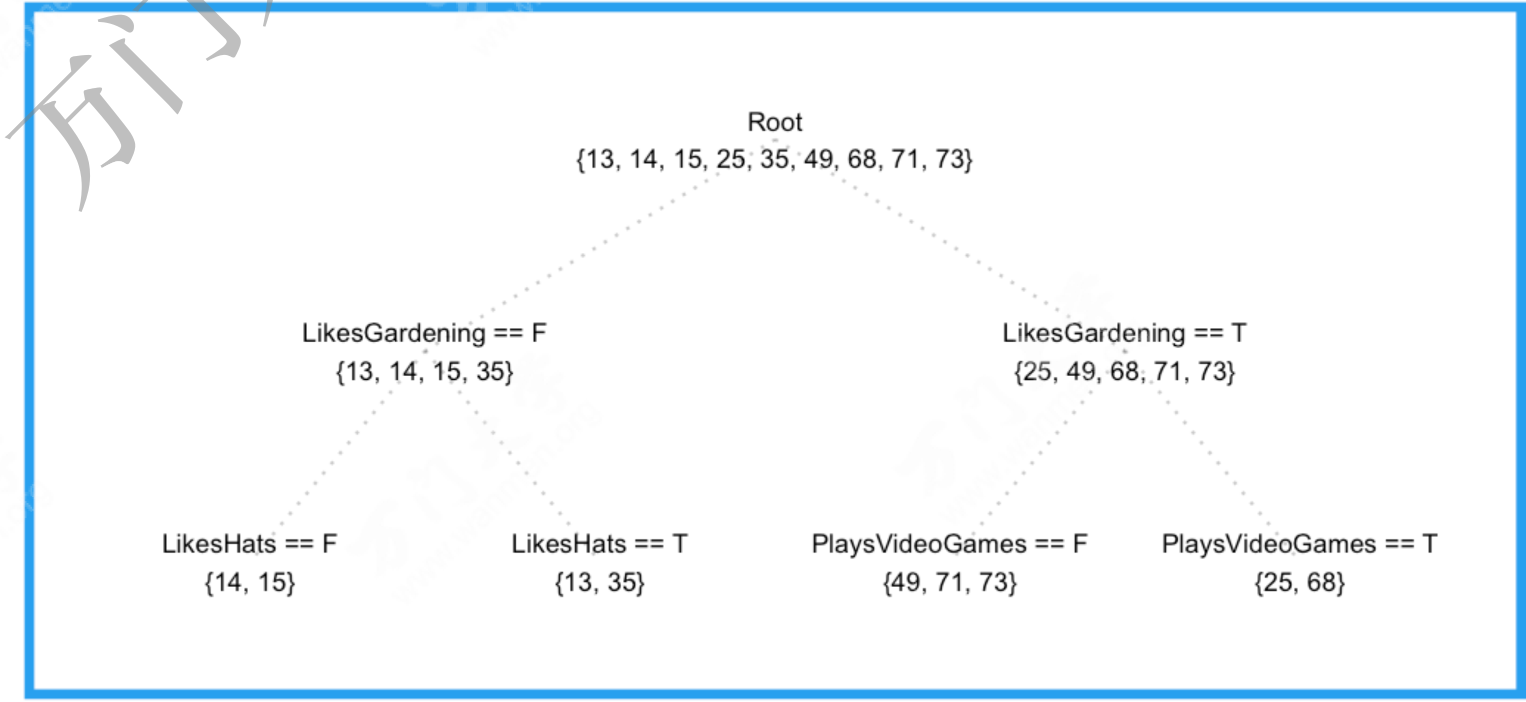
CRUISER

让一颗树无穷强大还是很多颗树

Tree 1



Overfit Tree

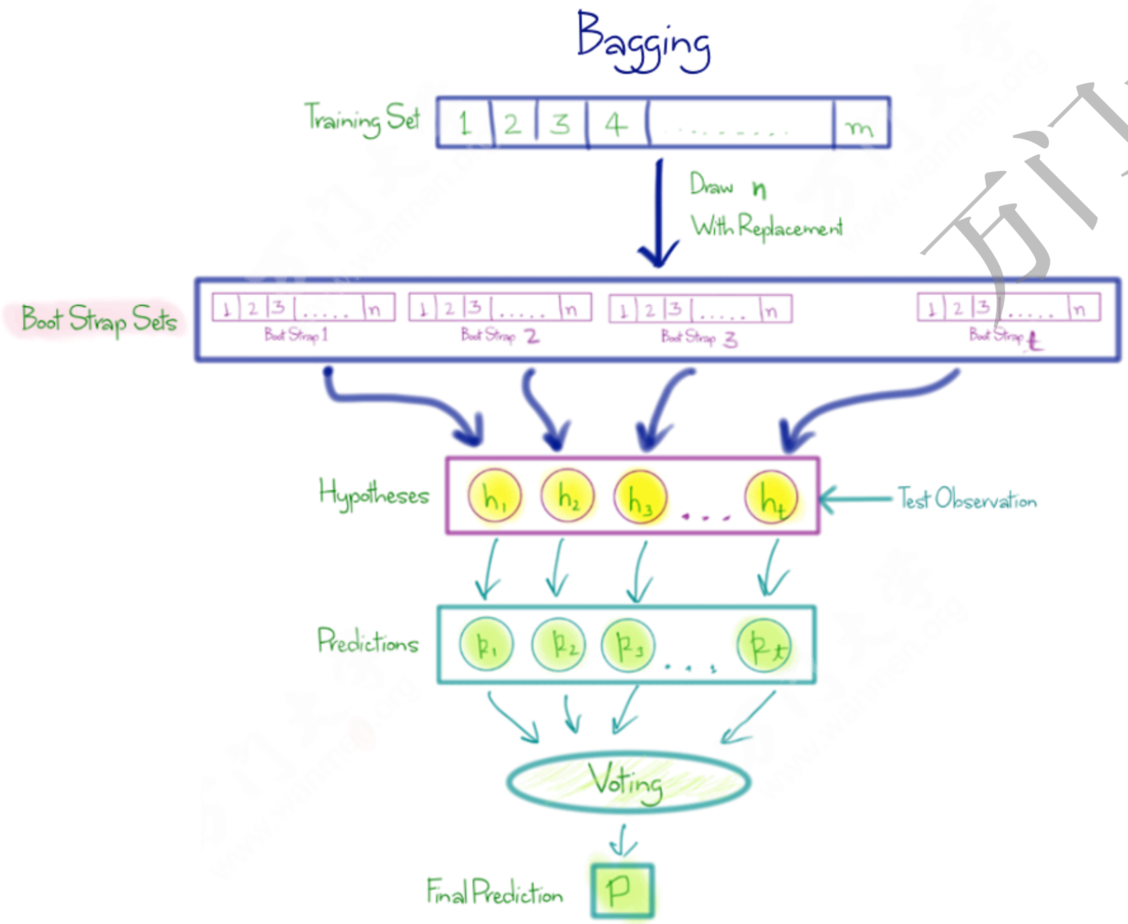


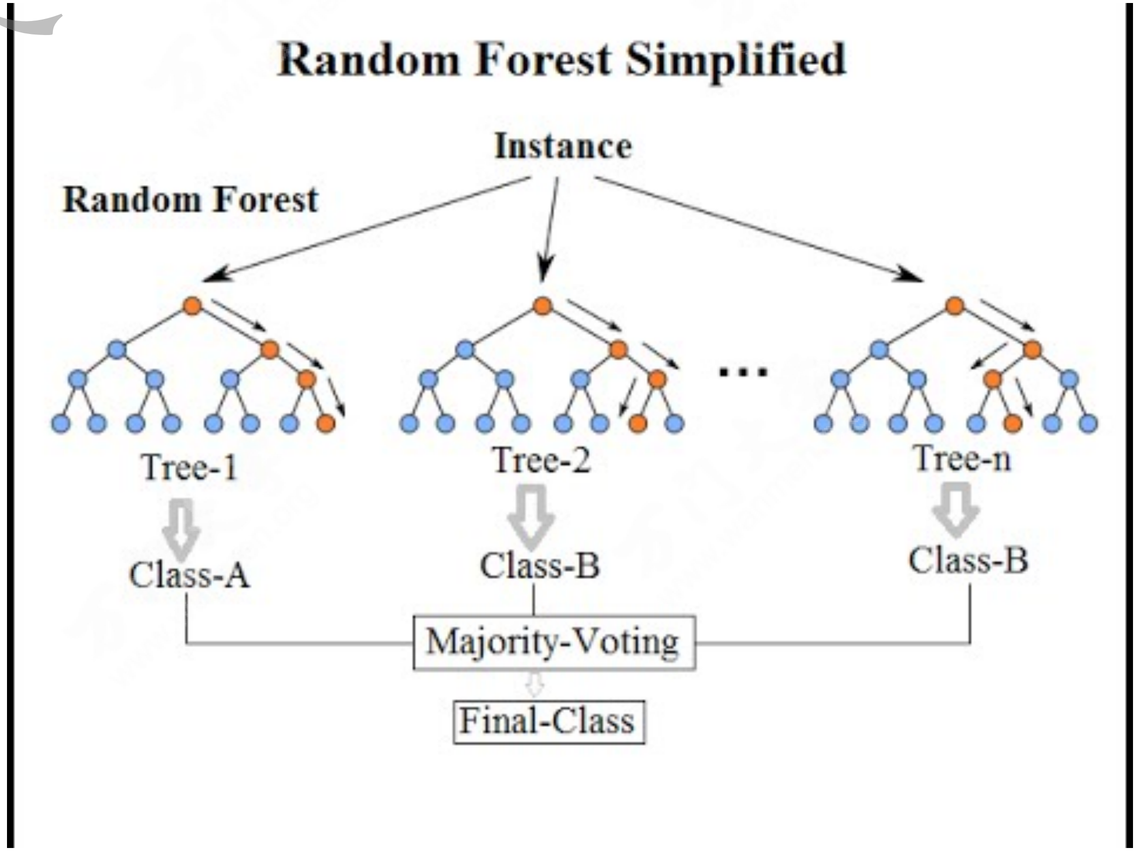
为什么一颗树会过拟合？

一颗树还是一片森林

解决方法I:Bagging Method- 随机森林

和而不同


Bagging
Training Set
1 2 3 4 .... m
Draw n
With Replacement
Boot Strap Sets
Boot Strap 1
Boot Strap 2
Boot Strap 3
Boot Strap t
Hypotheses
Test Observation
Predictions
Voting
Final Prediction
P


Random Forest Simplified
Instance
Random Forest
Tree-1
Tree-2
Tree-n
Class-A
Class-B
Class-B
Majority-Voting
Final-Class

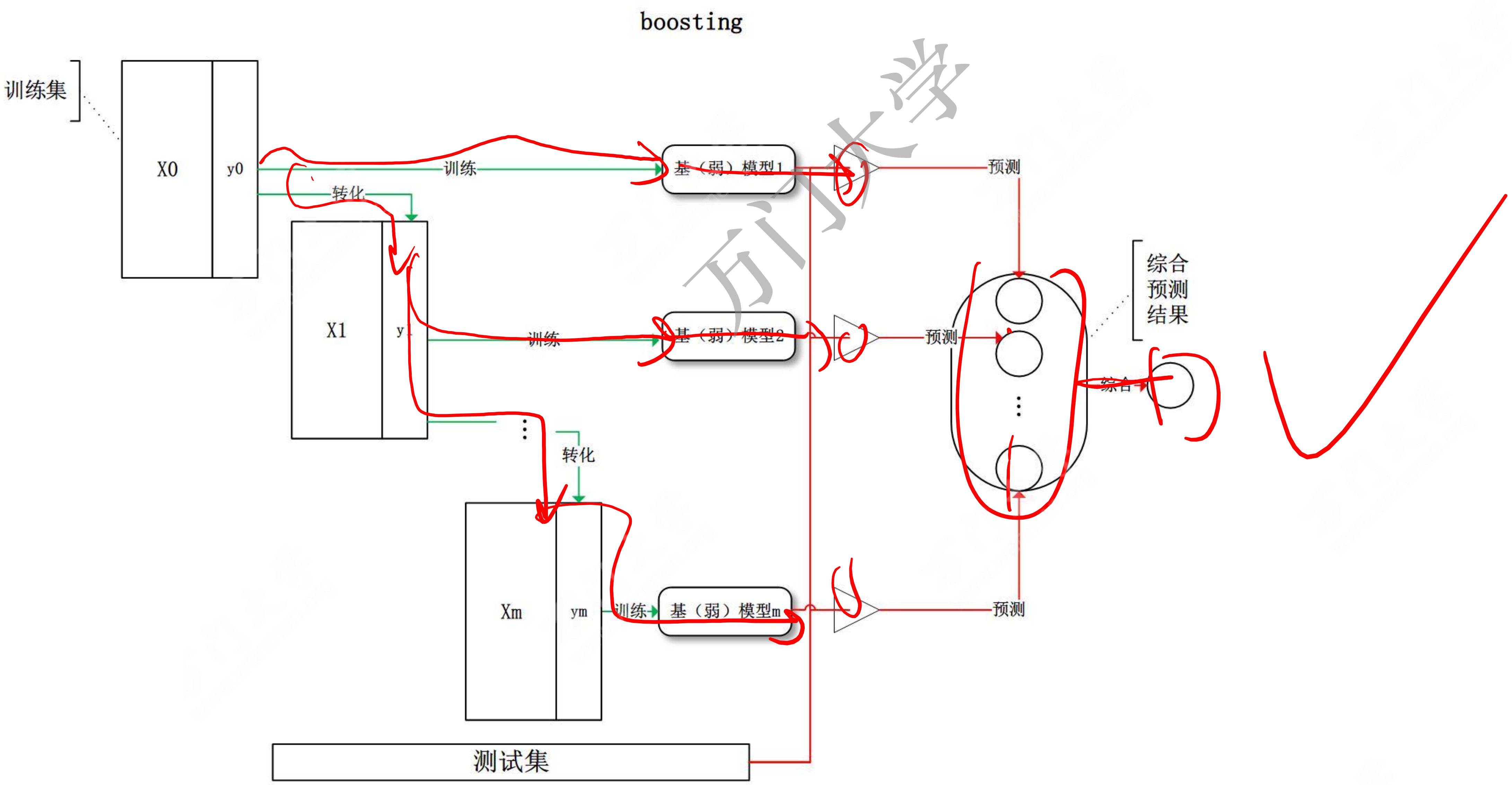
CRUISER

问题： 随机森林的显著效果是?

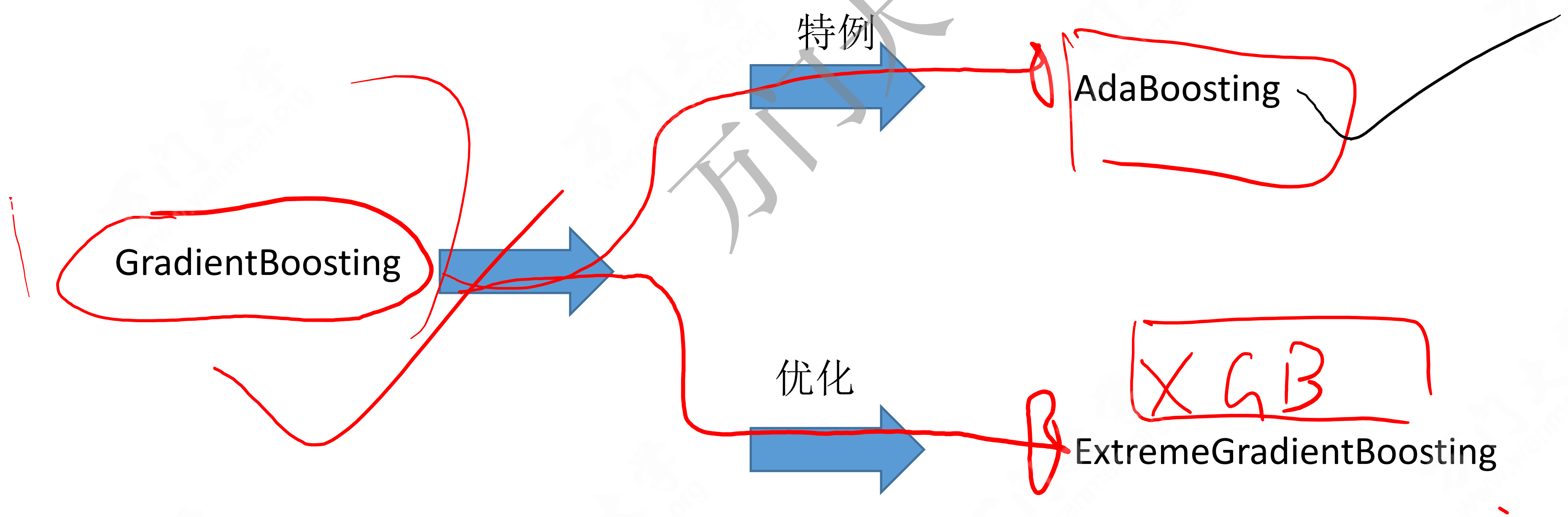
A， 减少了模型方差， 可以有效防止过拟合

B， 增大模型拟合力，减少偏差

Boosting示意图



解决方法2： Boosting方法 - 在跌倒的地方爬起来



CRUISER

回到刚刚的问题

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

CRUISER

回到刚刚的问题

残差　　　　　　　　　　　　　　　　残差

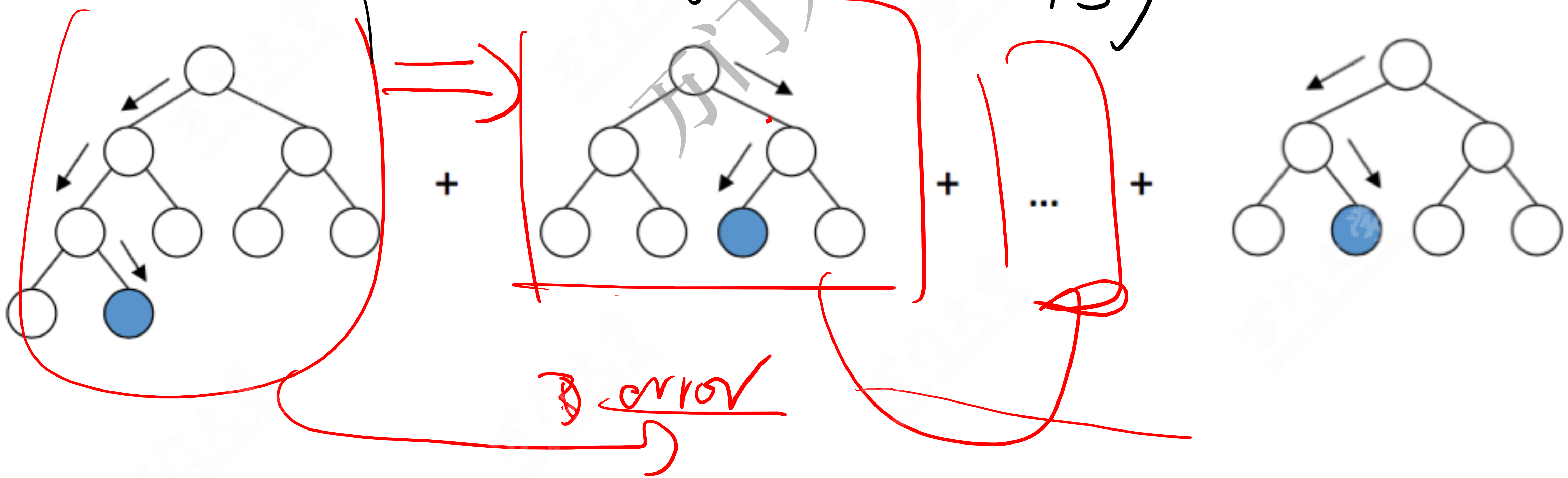
PersonID	Age	Tree1 Prediction	Tree1 Residual	Tree2 Prediction	Combined Prediction	Final Residual
1	13	19.25	-6.25	-3.567	15.68	2.683
2	14	19.25	-5.25	-3.567	15.68	1.683
3	15	19.25	-4.25	-3.567	15.68	0.6833
4	25	57.2	-32.2	-3.567	53.63	28.63
5	35	19.25	15.75	-3.567	15.68	-19.32
6	49	57.2	-8.2	7.133	64.33	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

Gradient Boosting

$F = (f_1 + f_2 + f_3)$

① 拟合残差



③ error

https://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html

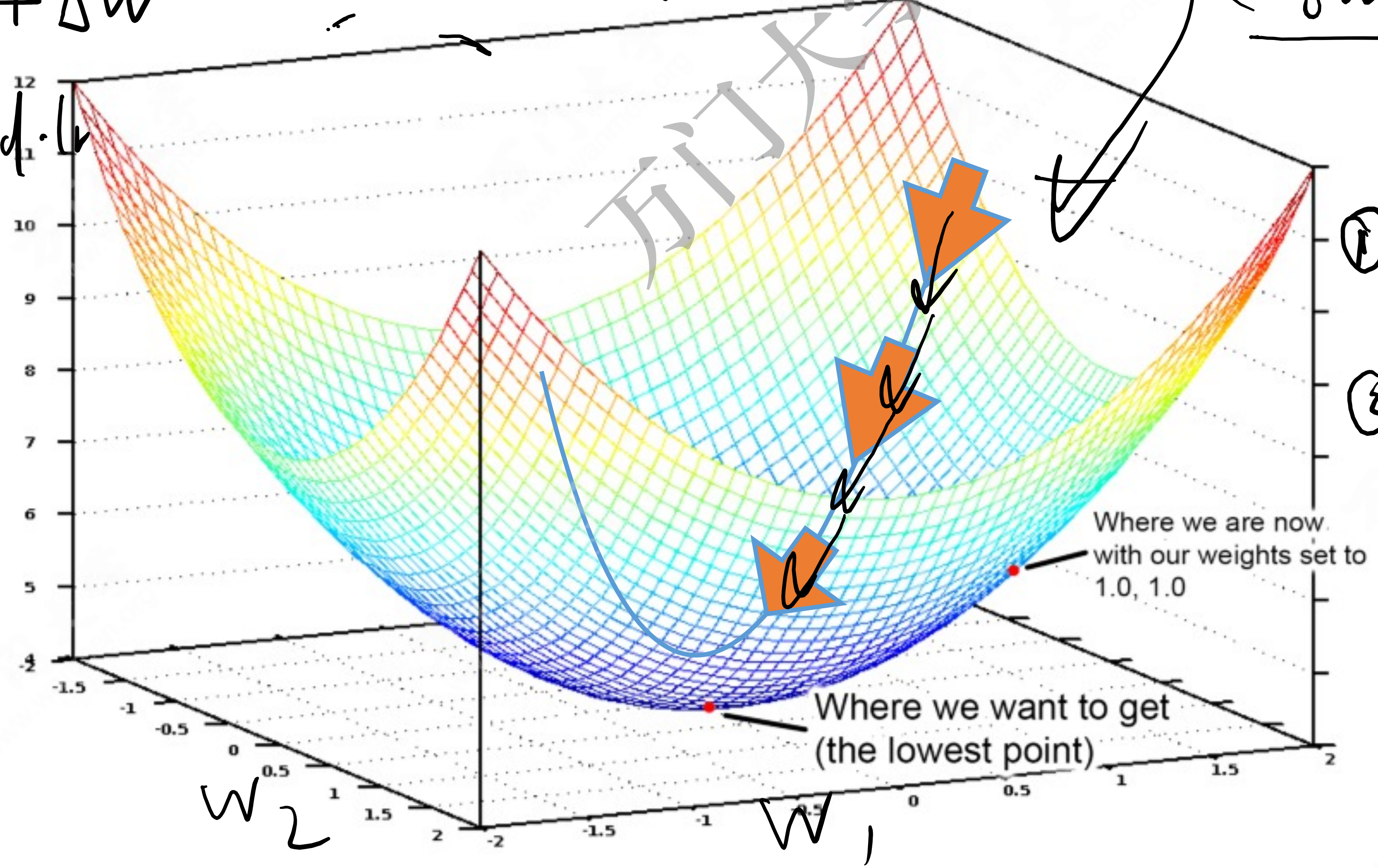
为什么是梯度？ 从损失函数讲起

$W = W + \Delta W$

$Cost = f(w)$

$\Delta W = -grad \cdot lr$

$\underbrace{\left(\frac{\partial f(w)}{\partial w_1}, \frac{\partial f(w)}{\partial w_2}\right)}_{grad}$



①每个都态、

② 做1个最好的改变

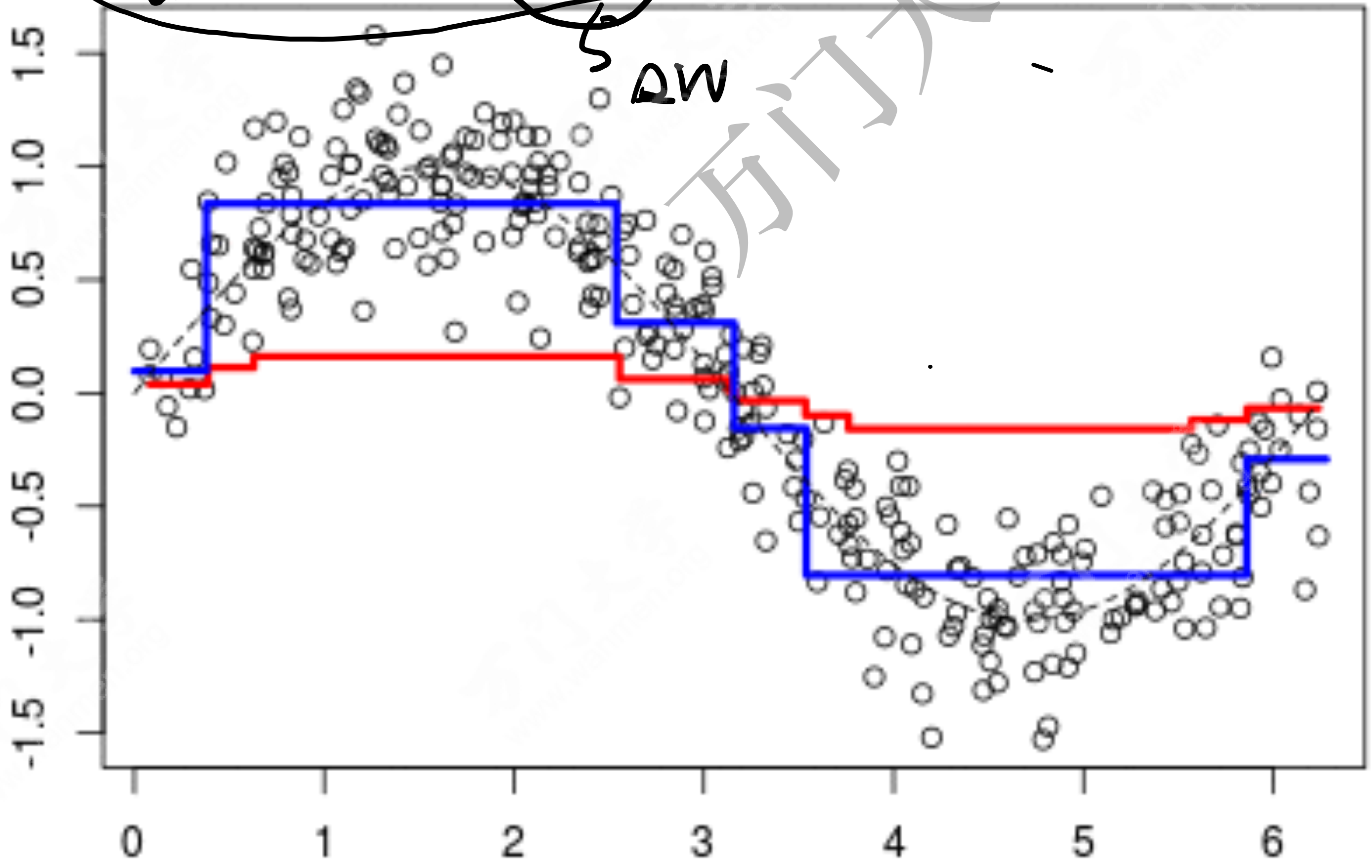
③ 得到更好的自己

梯度： 寻找当下最优的解决路径

$$F_{t+1} = F_t + f_i$$

梯度

ΔW



CRUISER

“函数”作为梯度前进的方向

Gradient Boosting [J. Friedman, 1999]

$\partial \, Cost(w) / \partial w$

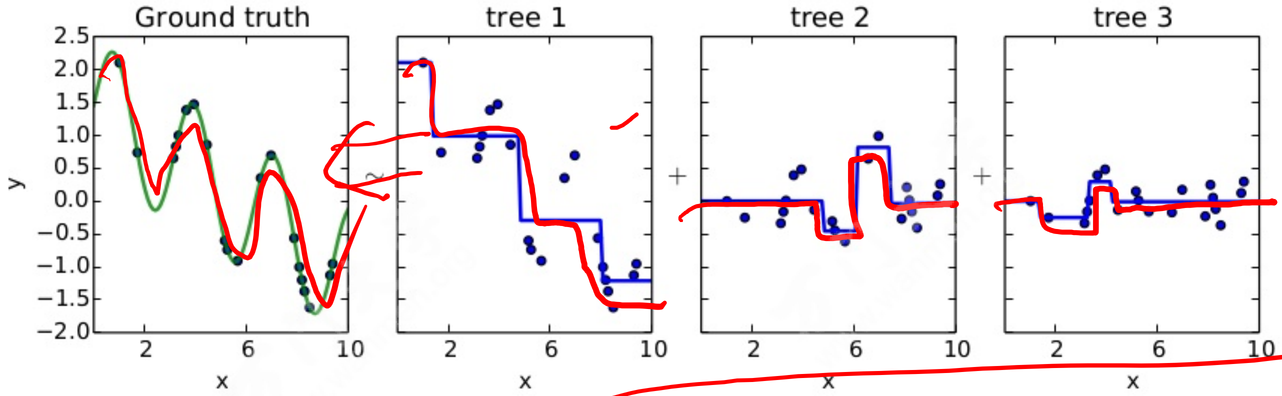
$$Cost(F) = \sum_{i=1}^{N} (F(x_i) - Y_i)^2$$

Statistical view on boosting

- $\Rightarrow$ Generalization of boosting to arbitrary loss functions

$$\partial \, Cost(F) / \partial F(x_i) = (F(x_i) - Y_i) \cdot 2$$

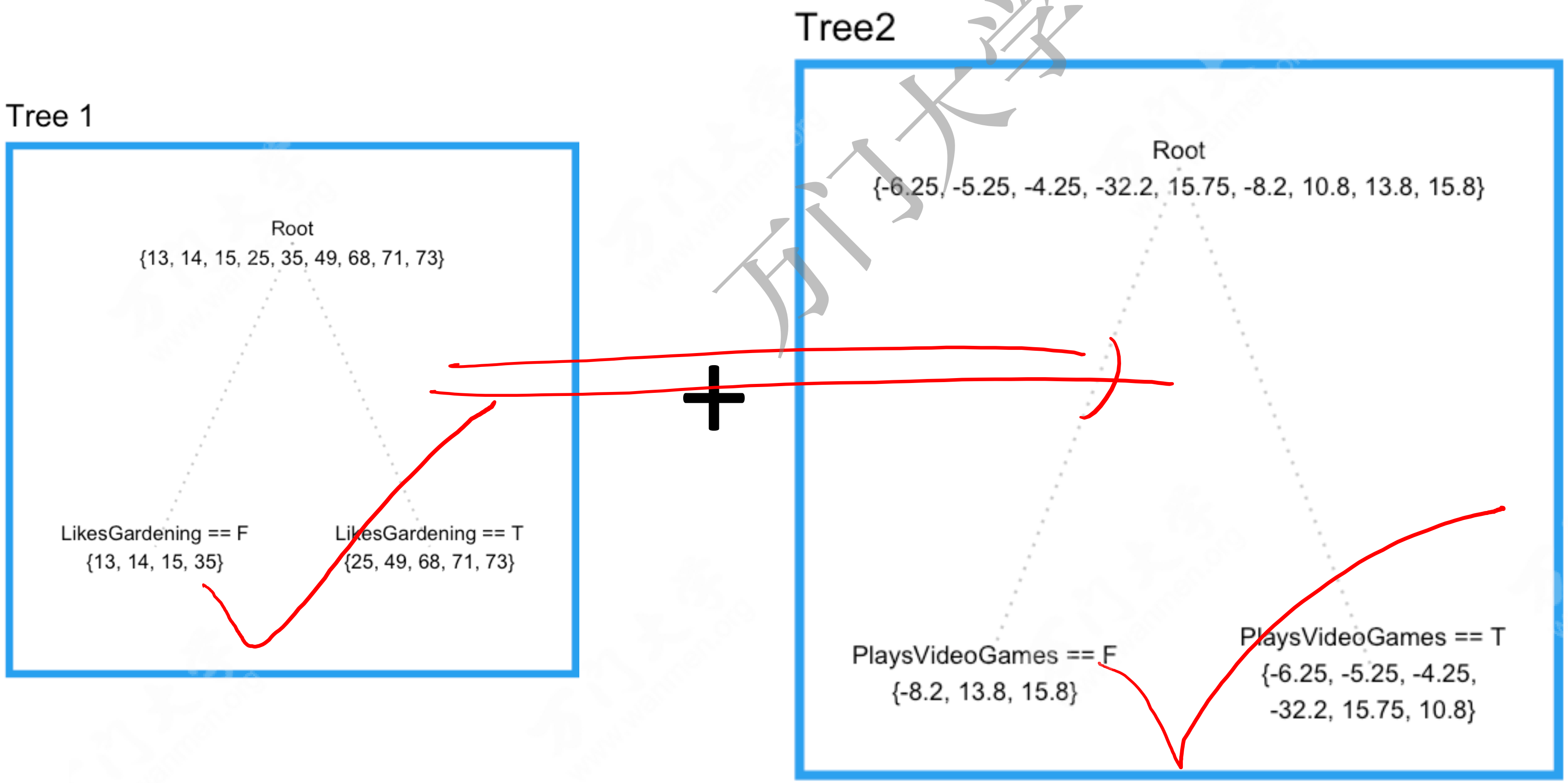
Residual fitting



$$\frac{\partial \, Cost(F)}{\partial F(x_1)} \cdots \frac{\partial \, F(cost)}{\partial F(x_N)} = (F(x_1) - Y_1, \ldots, F(x_N) - Y_N)$$

sklearn.ensemble.GradientBoostingClassifier|Regressor

分类树


Tree 1
Root
{13, 14, 15, 25, 35, 49, 68, 71, 73}
LikesGardening == F
{13, 14, 15, 35}
LikesGardening == T
{25, 49, 68, 71, 73}
+
Tree2
Root
{-6.25, -5.25, -4.25, -32.2, 15.75, -8.2, 10.8, 13.8, 15.8}
PlaysVideoGames == F
{-8.2, 13.8, 15.8}
PlaysVideoGames == T
{-6.25, -5.25, -4.25,
-32.2, 15.75, 10.8}

尝试不同损失函数

1. $\sum_{i=1}^{n} (Y - f(x))^2$ | 回归

2. $cost = |Y - f(x)|$ $\frac{\partial\, cost}{T\ominus \partial F}$

3. $cost = \sum e^{-Y_i f(x_i)}$ $\begin{cases} Y_i = [-1, +1] \\ f(x_i) = [-1, +1] \end{cases}$

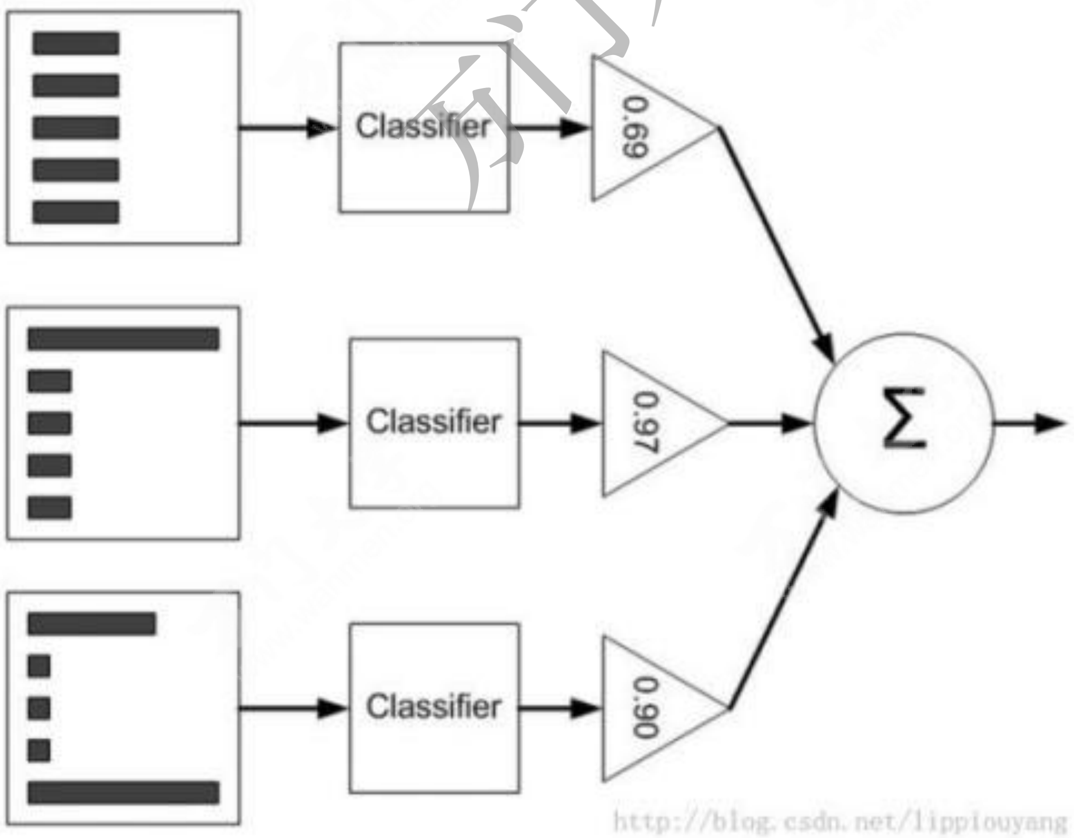
$Y_i = f(x)$ $Y_i f(x_i) = 1$ $loss = e^{-1}$

$Y_i \neq f(x)$ $Y_i f(x_i) = -1$ $loss = e^{+1}$

Adaboosting

$$\alpha = \frac{1}{2} \ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$$

The AdaBoost algorithm can be seen schematically in figure 7.1.



Algorithm AdaBoost.M2

-1

+1

Input: sequence of m examples $\langle(x_1, y_1), \ldots, (x_m, y_m)\rangle$ with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Let $B = \{(i, y) : i \in \{1, \ldots, m\}, y \neq y_i\}$
Initialize $D_1(i, y) = 1/|B|$ for $(i, y) \in B$.
Do for $t = 1, 2, \ldots, T$

1. Call **WeakLearn**, providing it with mislabel distribution D_t.
2. Get back a hypothesis $h_t : X \times Y \to [0, 1]$.
3. Calculate the pseudo-loss of h_t: $\epsilon_t = \frac{1}{2} \sum_{(i,y)\in B} D_t(i, y)(1 - h_t(x_i, y_i) + h_t(x_i, y))$.
4. Set $\beta_t = \epsilon_t/(1 - \epsilon_t)$.
5. Update D_t: $D_{t+1}(i, y) = \frac{D_t(i, y)}{Z_t} \cdot \beta_t^{(1/2)(1+h_t(x_i,y_i)-h_t(x_i,y))}$
where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the hypothesis: $h_{fin}(x) = \arg\max_{y\in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x, y)$.

Figure 2: The algorithm **AdaBoost.M2**.

如何建立损失函数

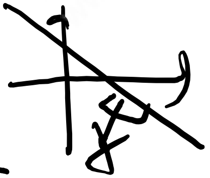
1， 衡量真实和预测值得距离

$Loss \sim y \log p + (1-p) \log p$

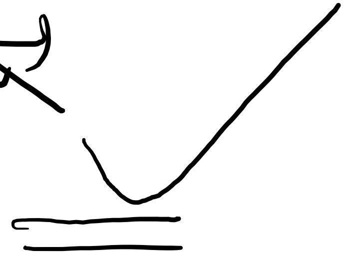
$\Sigma e^{-f(x) y}$

2， 连续可微可导

$x \quad \Sigma\, y_i (wx+b)\, x$



$\sim y \log p + (1-y) \log p$



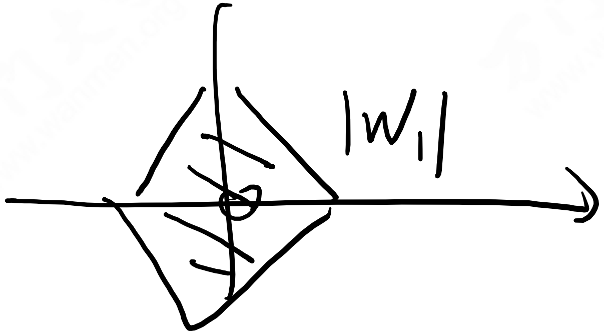
3， 根据数据分布设定

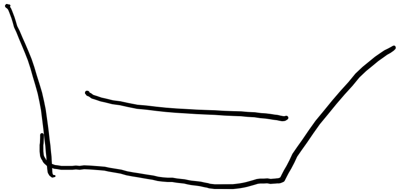
$|y - f(x)| \sim$

$\Sigma (y - f(x))^2 \sim$

4， 具有对噪声的鲁棒性

5， 合理加入正则项





$P(w) \sim e^{-\lambda w^2}$: ridge L2

$P(w) \sim e^{-|w|}$

升级版本： GradientBoosting