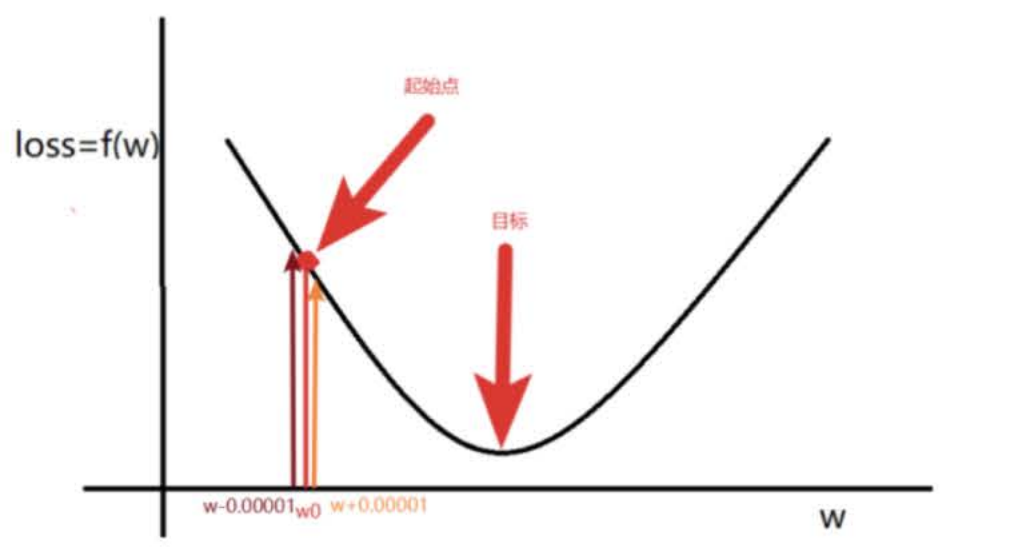
2 GBDT(了解)

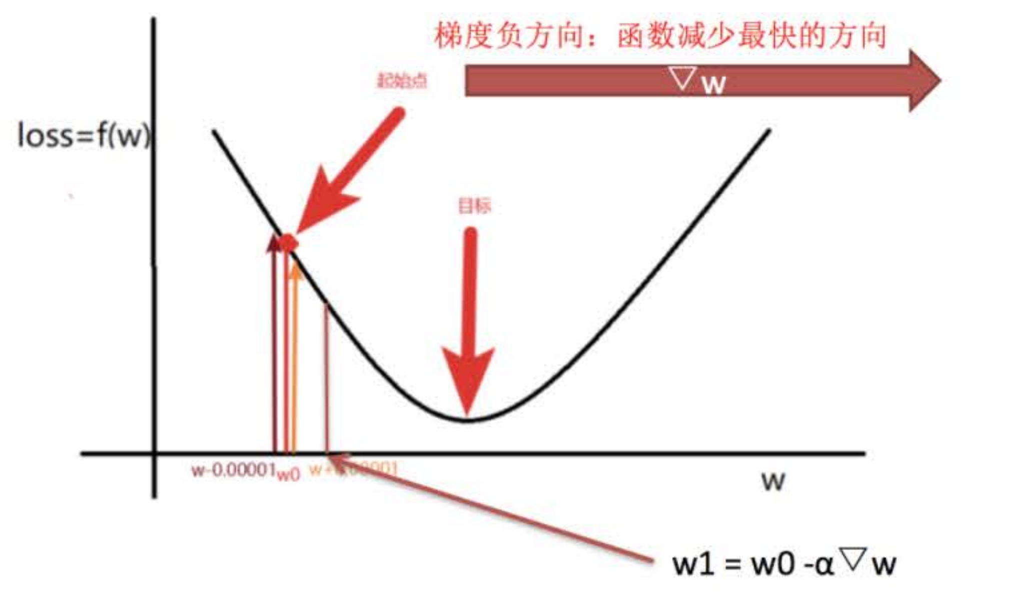
梯度提升决策树(GBDT Gradient Boosting Decision Tree) **是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。**它在被提出之初就被认为是泛化能力（generalization)较强的算法。近些年更因为被用于搜索排序的机器学习模型而引起大家关注。

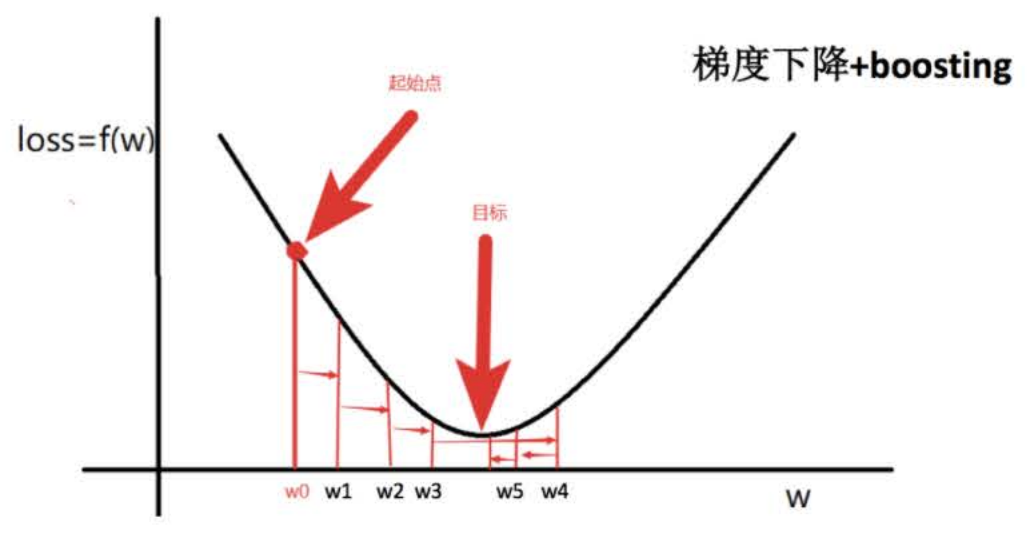
GBDT = 梯度下降 + Boosting + 决策树

2.1 梯度的概念(复习)



$$\nabla w=\frac{f(w+0.000001)-f(w-0.000001)}{2*0.000001}$$





w1 = w0 -α∇f(w0;x)   w2 = w1 -α∇f(w1;x)   w3 = w2 -α∇f(w2;x)

w4 = w3 -α∇f(w3;x)   w5 = w4 -α∇f(w4;x)

w5 = w0 -α∇f(w0;x) -α∇f(w1;x) -α∇f(w2;x) -α∇f(w3;x) -α∇f(w4;x)

2.2 GBDT执行流程

w5 = w0 -α∇f(w0;x) -α∇f(w1;x) -α∇f(w2;x) -α∇f(w3;x) -α∇f(w4;x)

α=1 $h_i(x)$ = -∇f(w_i;x)   H(x) – boosting集成表达方式

$$H(x)=h_0(x)+h_1(x)+h_2(x)+h_3(x)+h_4(x)+......$$

如果上式中的$h_i(x)$=决策树模型,则上式就变为:

GBDT = 梯度下降 + Boosting + 决策树

2.3 案例

预测编号5的身高:

编号	年龄(岁)	体重(KG)	身高(M)
1	5	20	1.1
2	7	30	1.3
3	21	70	1.7
4	30	60	1.8
5	25	65	?

第一步:计算损失函数,并求出第一个预测值:

$$\text{loss}=\frac{1}{2\text{m}}\sum_{i=1}^{m}(y-y')^2 \quad\Rightarrow\quad \frac{\partial\text{loss}}{\partial y'}=-\frac{1}{m}\sum_{i=1}^{m}(y-y')$$

$$-\frac{\partial\text{loss}}{\partial y'}=\frac{(1.1-y')+(1.3-y')+(1.7-y')+(1.8-y')}{4}=0$$

$$\frac{1.1+1.3+1.7+1.8}{4}-y'=0 \qquad y'=1.475$$

$h_0(x)$=1.475

$$H(x)=h_0(x)+h_1(x)+h_2(x)+h_3(x)+h_4(x)+......$$

第二步:求解划分点

真实身高	预测身高	误差值
1.1	1.475	-0.375
1.3	1.475	-0.175
1.7	1.475	0.225
1.8	1.475	0.325

重构目标值

编号	年龄	体重	目标值
1	5	20	-0.375
2	7	30	-0.175
3	21	70	0.225
4	30	60	0.325
5	25	65	?

划分点	小于划分点	大于划分点	纯度（使用方差）
年龄5	无	1，2，3，4	0.082
年龄7	1	2，3，4	0.047
年龄21	1，2	3，4	0.0125
年龄30	1，2，3	4	0.062
体重20	无	1，2，3，4	0.082
体重30	1	2，3，4	0.047
体重70	1，2	3，4	0.0125
体重60	1，2，4	3	0.0867

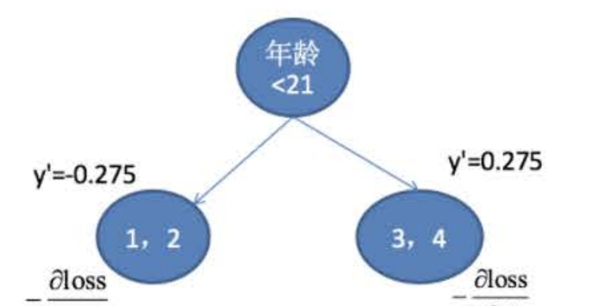
以年龄21为例

$$\frac{(-0.375+0.275)^2+(-0.175+0.275)^2}{2}=0.01$$

$$\frac{(0.225-0.275)^2+(0.325-0.275)^2}{2}=0.0025$$

得出:年龄21为划分点的方差=0.01+0.0025=0.0125

第三步:通过调整后目标值,求解得出$h_1(x)$



$$-\frac{\partial\text{loss}}{\partial y'}=\frac{(-0.375-y')+(-0.175-y')}{2}=0 \qquad -\frac{\partial\text{loss}}{\partial y'}=\frac{(0.225-y')+(0.325-y')}{2}=0$$

编号	目标值
1	-0.375
2	-0.175
3	0.225
4	0.325
5	?

编号	预测值
1	-0.275
2	-0.275
3	0.275
4	0.275
5	?

$h_0(x)$=1.475

$$h_1(x)=\begin{cases}-0.275, & 年龄<21\\ 0.275, & 年龄=>21\end{cases}$$

$$H(x)=h_0(x)+h_1(x)+h_2(x)+h_3(x)+h_4(x)+......$$

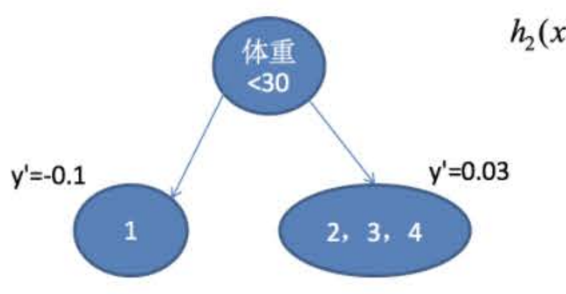
第四步:求解$h_2(x)$

第一轮目标值	预测值	误差值
-0.375	-0.275	-0.1
-0.175	-0.275	0.1
0.225	0.275	-0.05
0.325	0.275	0.05

重构目标值

编号	年龄	体重	目标值
1	5	20	-0.1
2	7	30	0.1
3	21	70	-0.05
4	30	60	0.05
5	25	65	?

体重30为划分点



$$h_2(x)=\begin{cases}-0.1, & 体重<30\\ 0.03, & 体重=>30\end{cases}$$

编号	预测值	误差值
1	-0.1	0
2	0.03	0.07
3	0.03	-0.02
4	0.03	0.02

... ...

得出结果:

$$H(x)=1.475+\begin{cases}-0.1, & 体重<30\\ 0.03, & 体重=>30\end{cases}+\begin{cases}-0.275, & 年龄<21\\ 0.275, & 年龄=>21\end{cases}$$

编号5身高 = 1.475 + 0.03 + 0.275 = 1.78

2.4 GBDT主要执行思想

1.使用梯度下降法优化代价函数;

2.使用一层决策树作为弱学习器，负梯度作为目标值;

3.利用boosting思想进行集成。

3.XGBoost【了解】

XGBoost= 二阶泰勒展开+boosting+决策树+正则化

- **面试题：了解XGBoost么，请详细说说它的原理**

回答要点：二阶泰勒展开，boosting，决策树，正则化

Boosting：XGBoost使用Boosting提升思想对多个弱学习器进行迭代式学习

二阶泰勒展开：每一轮学习中，XGBoost对损失函数进行二阶泰勒展开，使用一阶和二阶梯度进行优化。

决策树：在每一轮学习中，XGBoost使用决策树算法作为弱学习进行优化。

正则化：在优化过程中XGBoost为防止过拟合，在损失函数中加入惩罚项，限制决策树的叶子节点个数以及决策树叶子节点的值。

4 什么是泰勒展开式【拓展】

泰勒展开式

$$f(x+\Delta x)=f(x)+f'(x)\cdot\Delta x+\frac{1}{2}f''(x)\cdot\Delta x^2+.....+\frac{1}{n!}f^{(n)}(x)\cdot\Delta x^n$$

一阶泰勒展开

$$f(x+\Delta x)\approx f(x)+f'(x)\cdot\Delta x$$

二阶泰勒展开

$$f(x+\Delta x)\approx f(x)+f'(x)\cdot\Delta x+\frac{1}{2}f''(x)\cdot\Delta x^2$$

泰勒展开越多，计算结果越精确