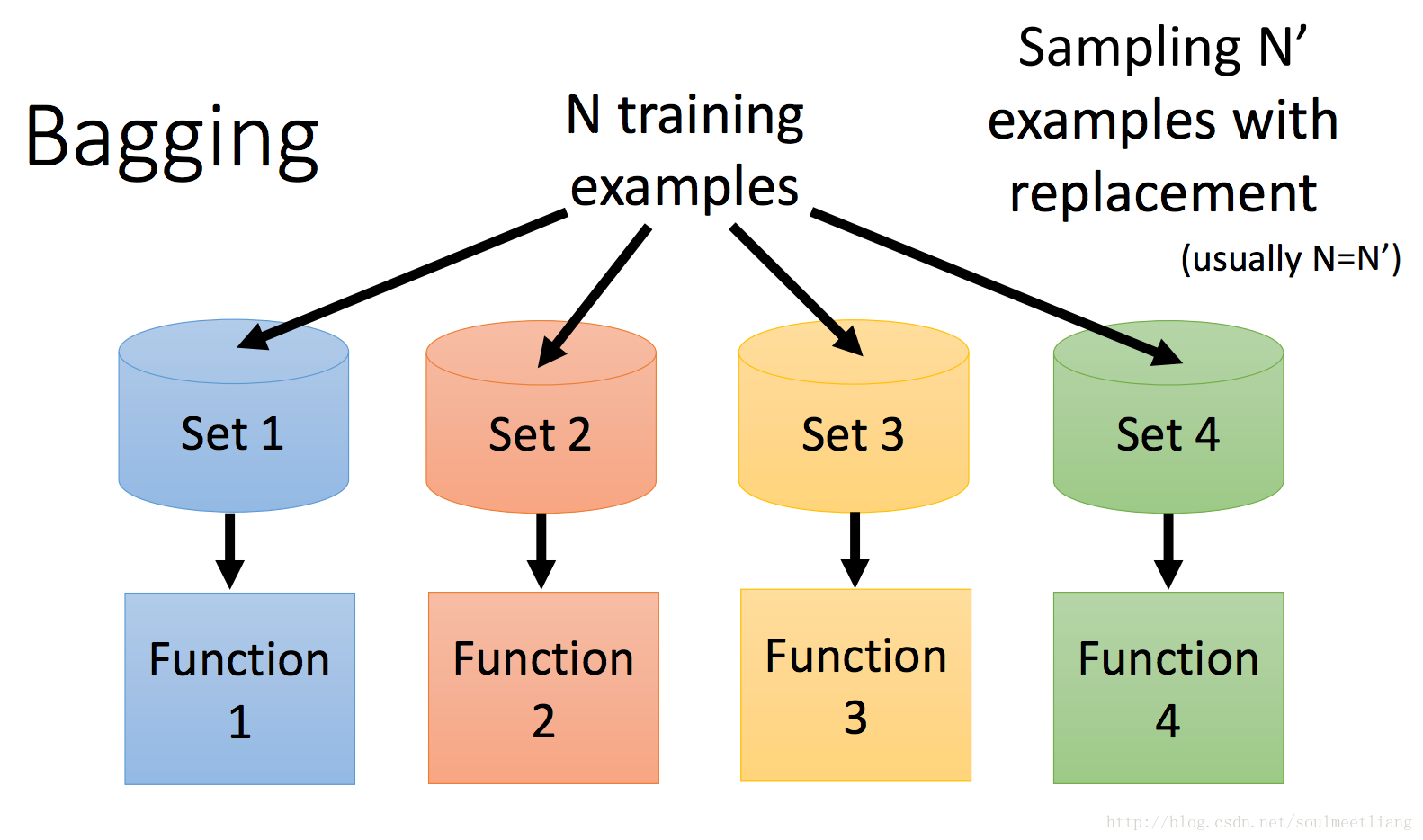
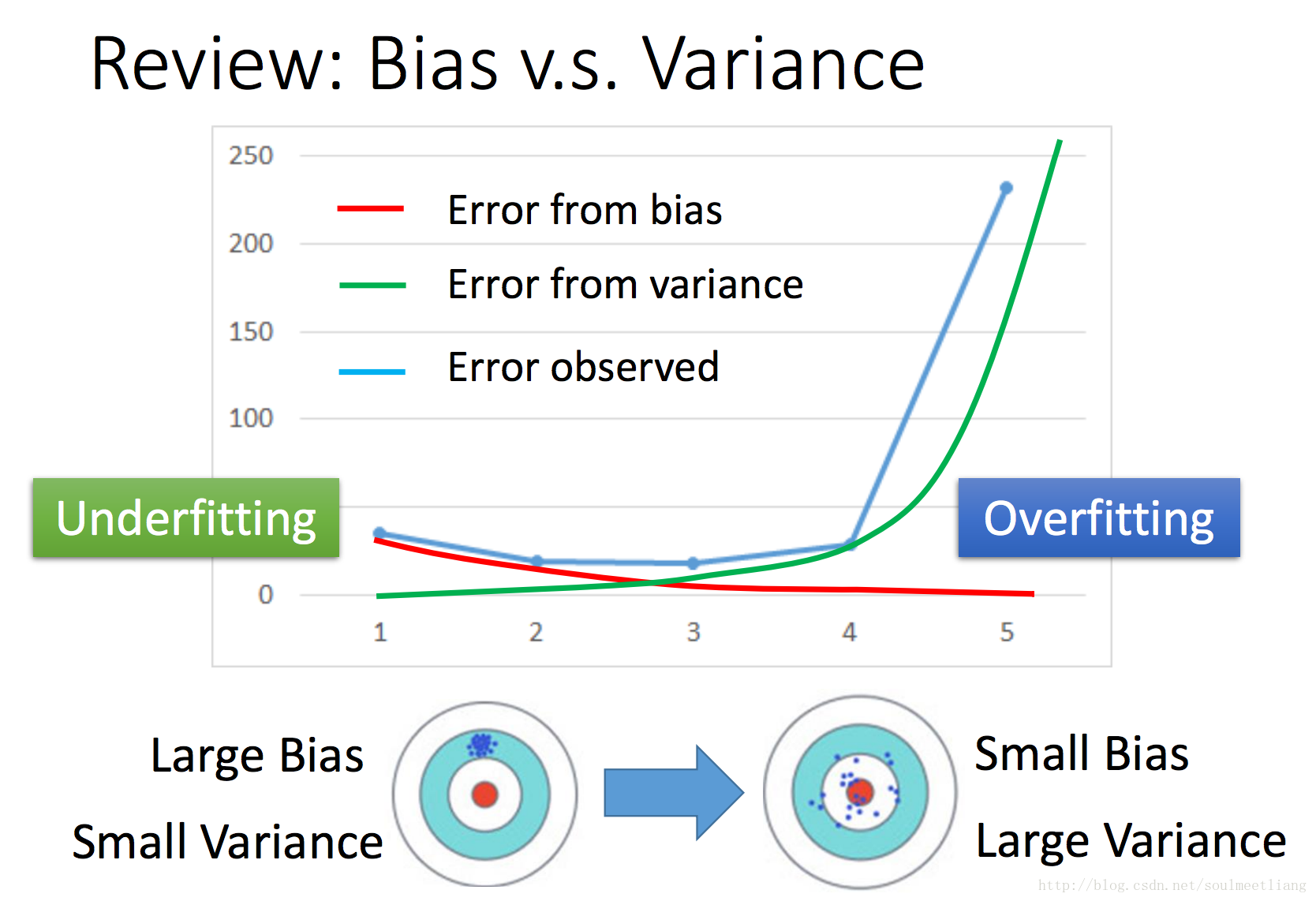
Ensemble: Bagging

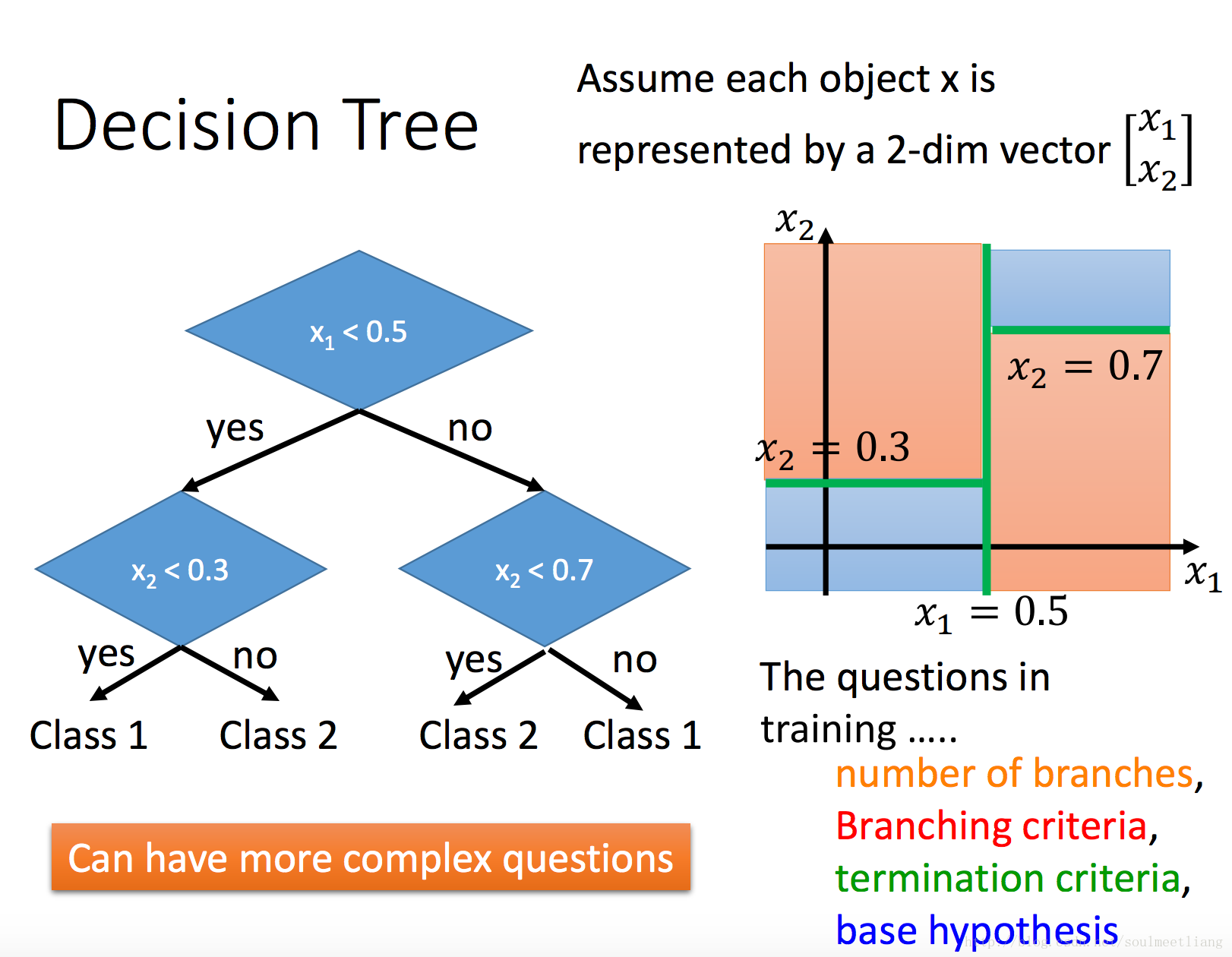
A complex model will have large variance.   
We can average complex models to reduce variance.   
If we average all the f\*, is it close to f^.   
所以，如果想让一个variance很大的model的error变小，可以训练好多个variance很大的model，然后把它们平均起来，这个方法就叫做Bagging。

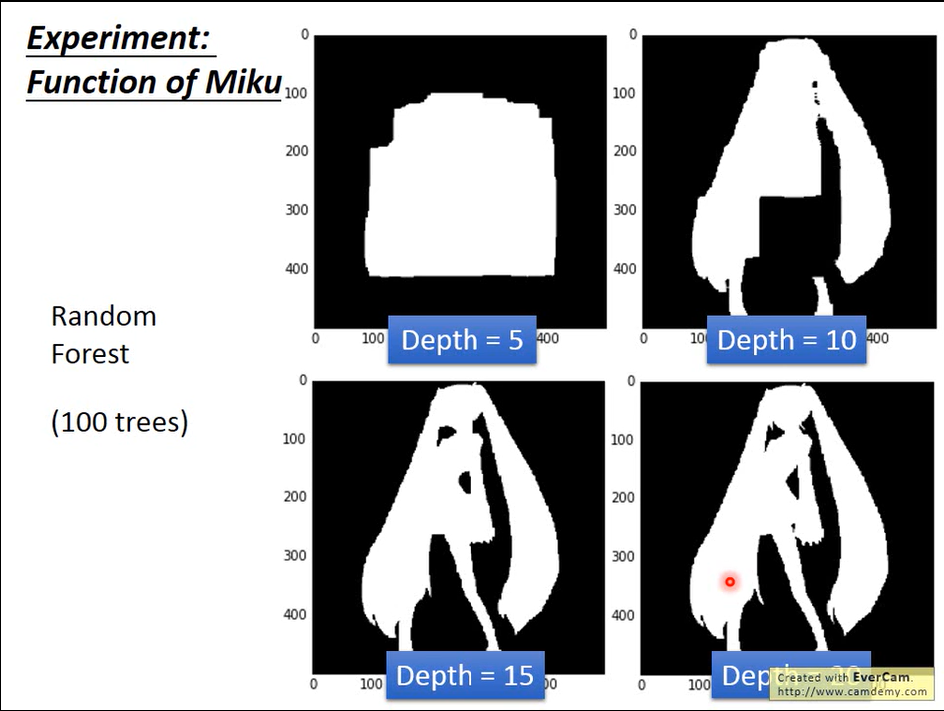


This approach would be helpful when your model is complex, easy to overfit.

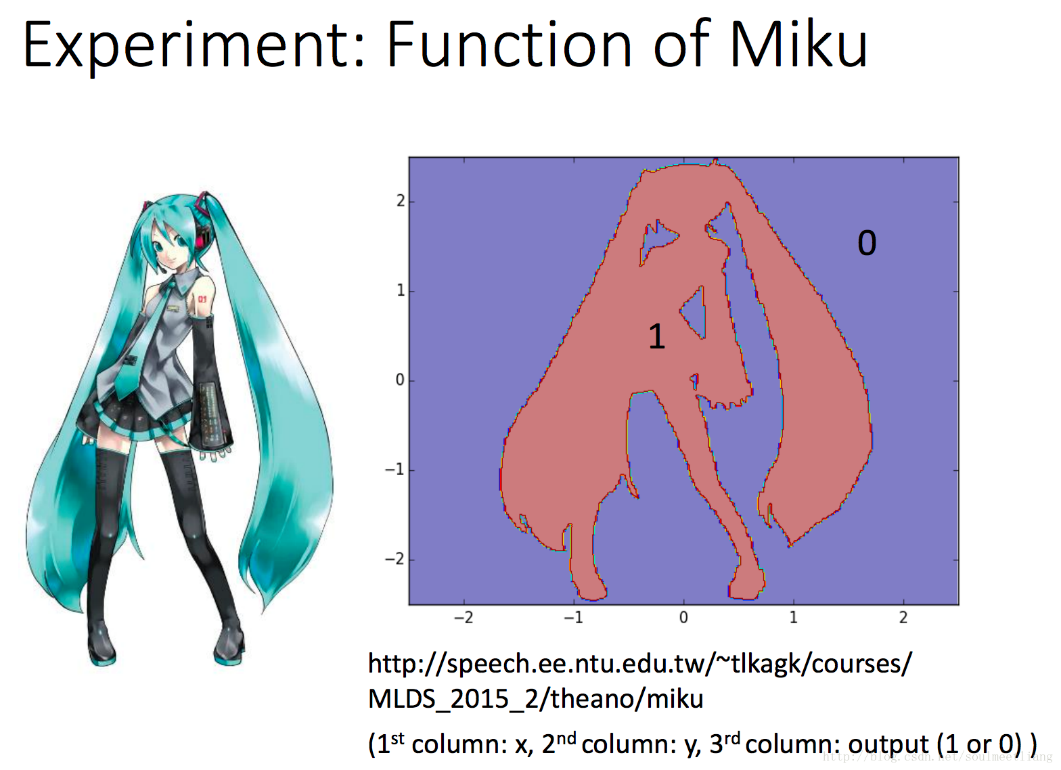
## Decision Tree

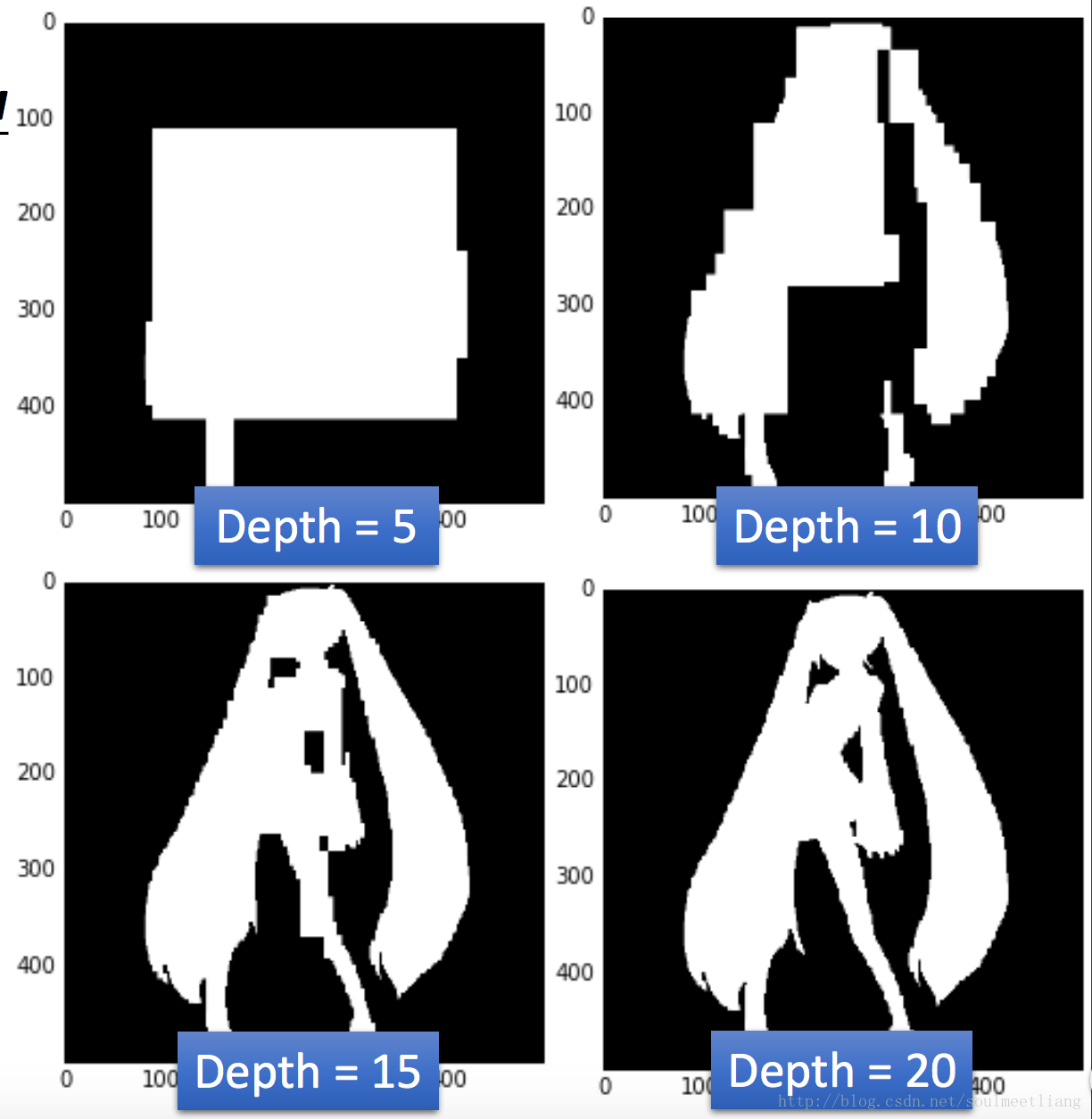
决策树很容易过拟合，模型很easy。只要想，随时可以在training data上达到100%的正确率（过拟合）。这时候就可以用embedding。

决策树不仅仅能判断上图的简单问题，还能解决很多复杂问题。   
比如……下图这个美少女，把初音的身体代表类别1，其他叫类别0，在这个二维平面上的值，就是input，来output判断是否在初音的身体上。



**注意：**Bagging并不会使model更fit data，但整体的function 会更平滑（variance更小）。

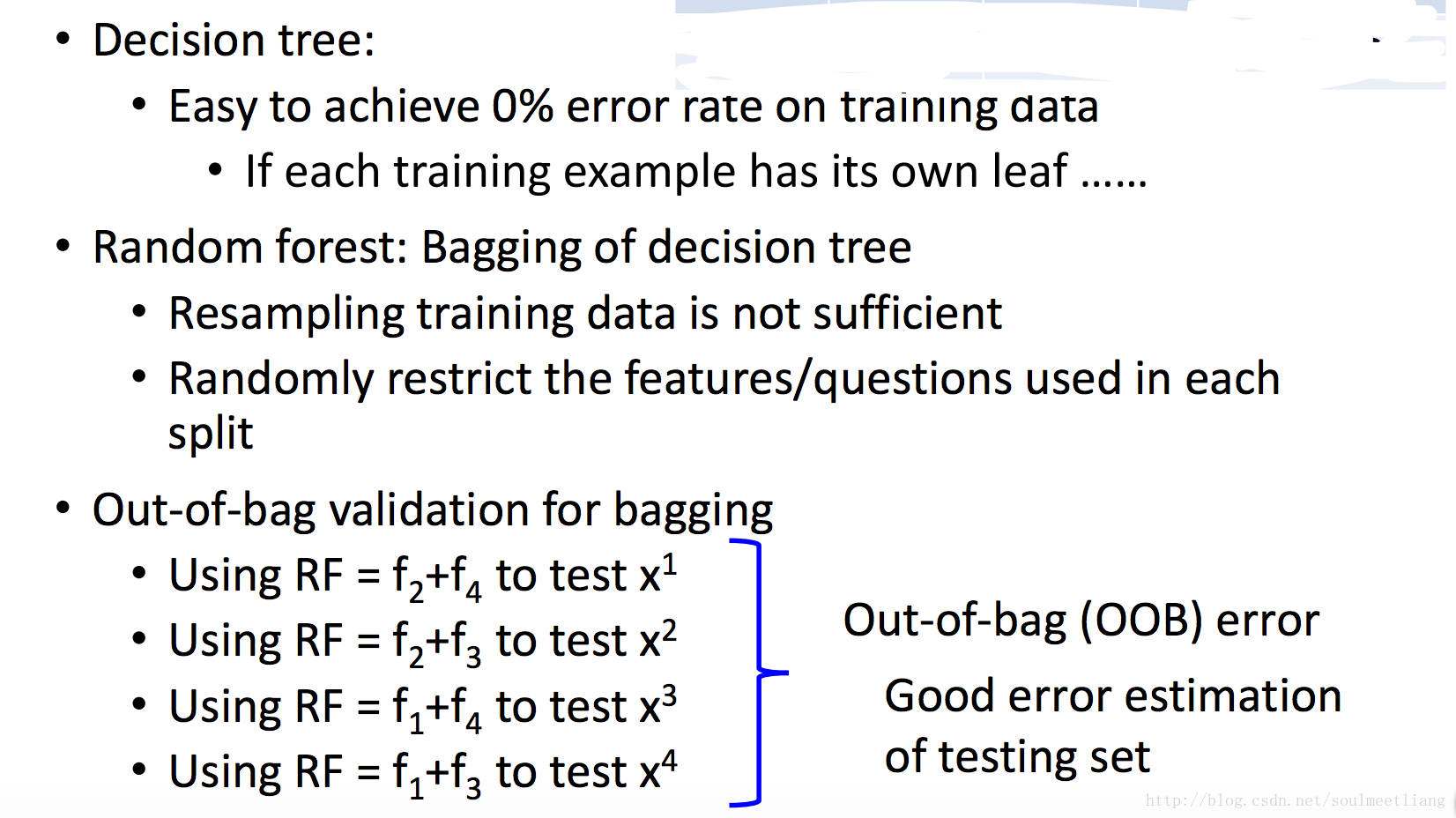




到这里很容易理解，决策树继续深下去完全可以做到training data上的完美，因为大不了它可以给每一个data分一个类，显而易见，这样很容易过拟合。

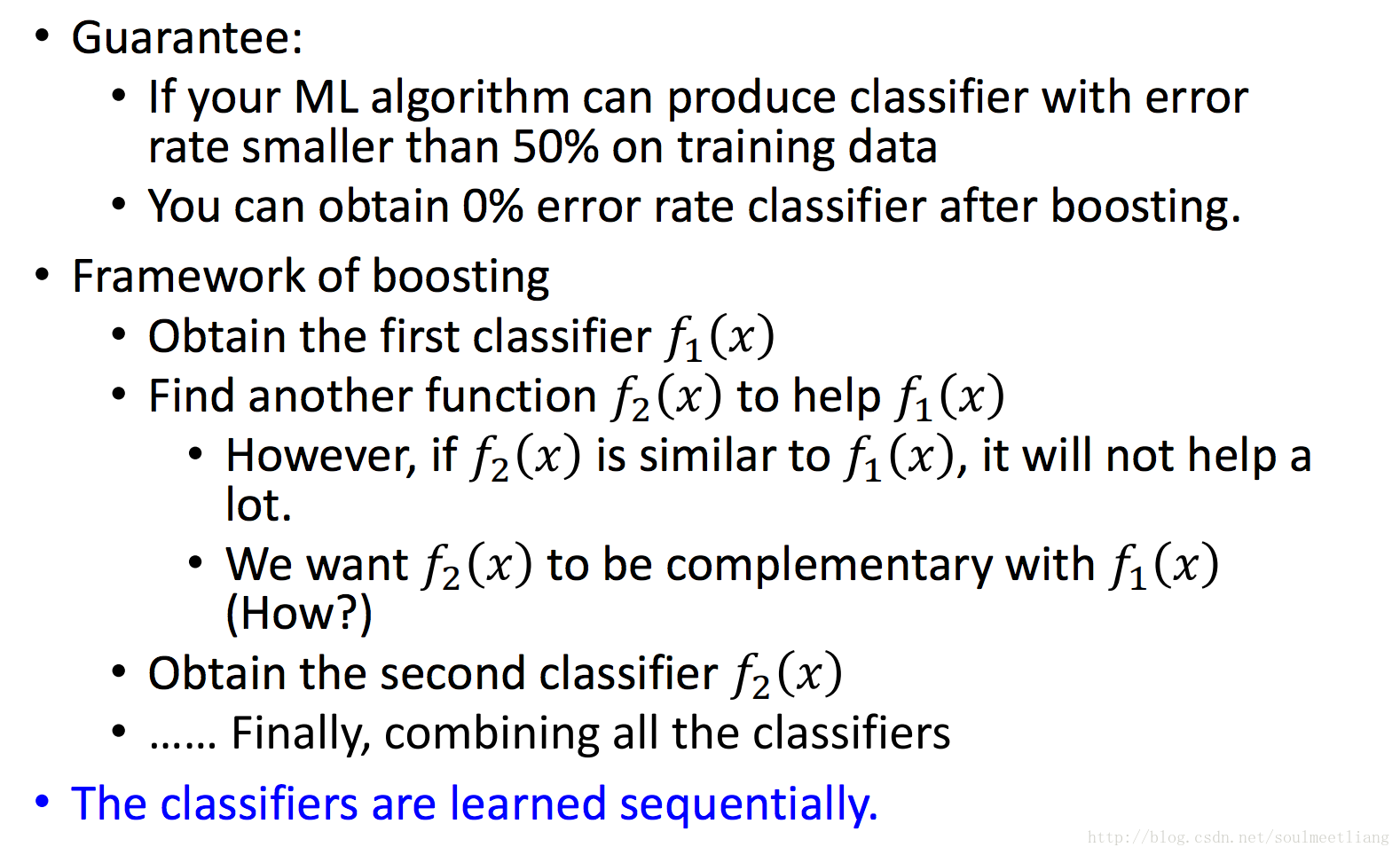
Decision Tree做 bagging 就是 Random Forest。

## Random Forest



# Ensemble: Boosting

Boosting 的目标和 Bagging 是相反的，Bagging是减弱过拟合，而 Boosting 是即使不能 fit training data 的 model，也要想办法使 其performance 更好，Boosting 通过把很多弱的 Classifiers结合起来，帮助得到强的 Classifiers。

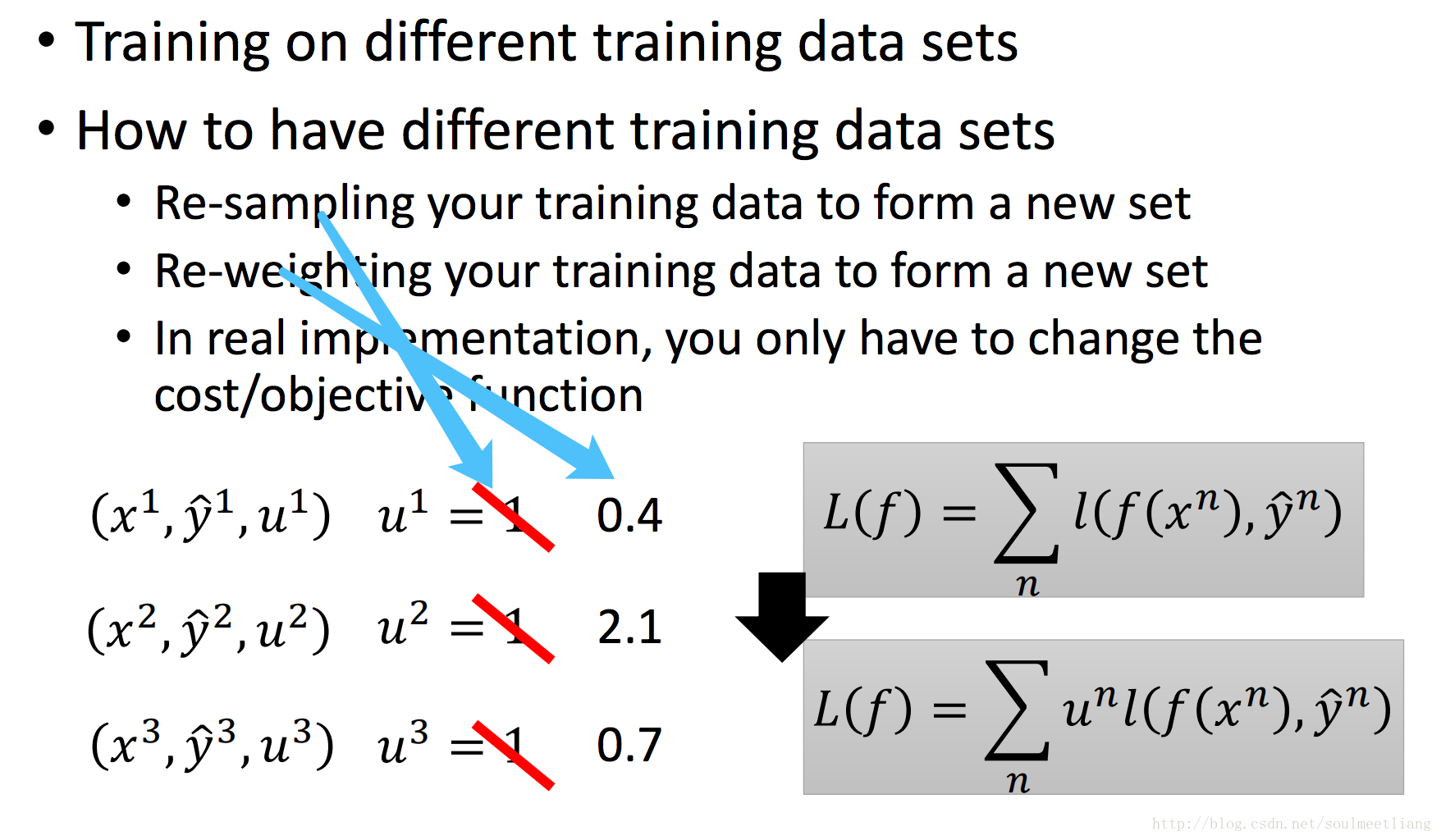


不同并且互补

必须按顺序boosting而不能并行。

## How to obtain different classifiers?

前面提到，classifier需要不同并且互补，则要不同的classifier通过在不同的 training data上做训练，不同的 training data 怎么得到呢？



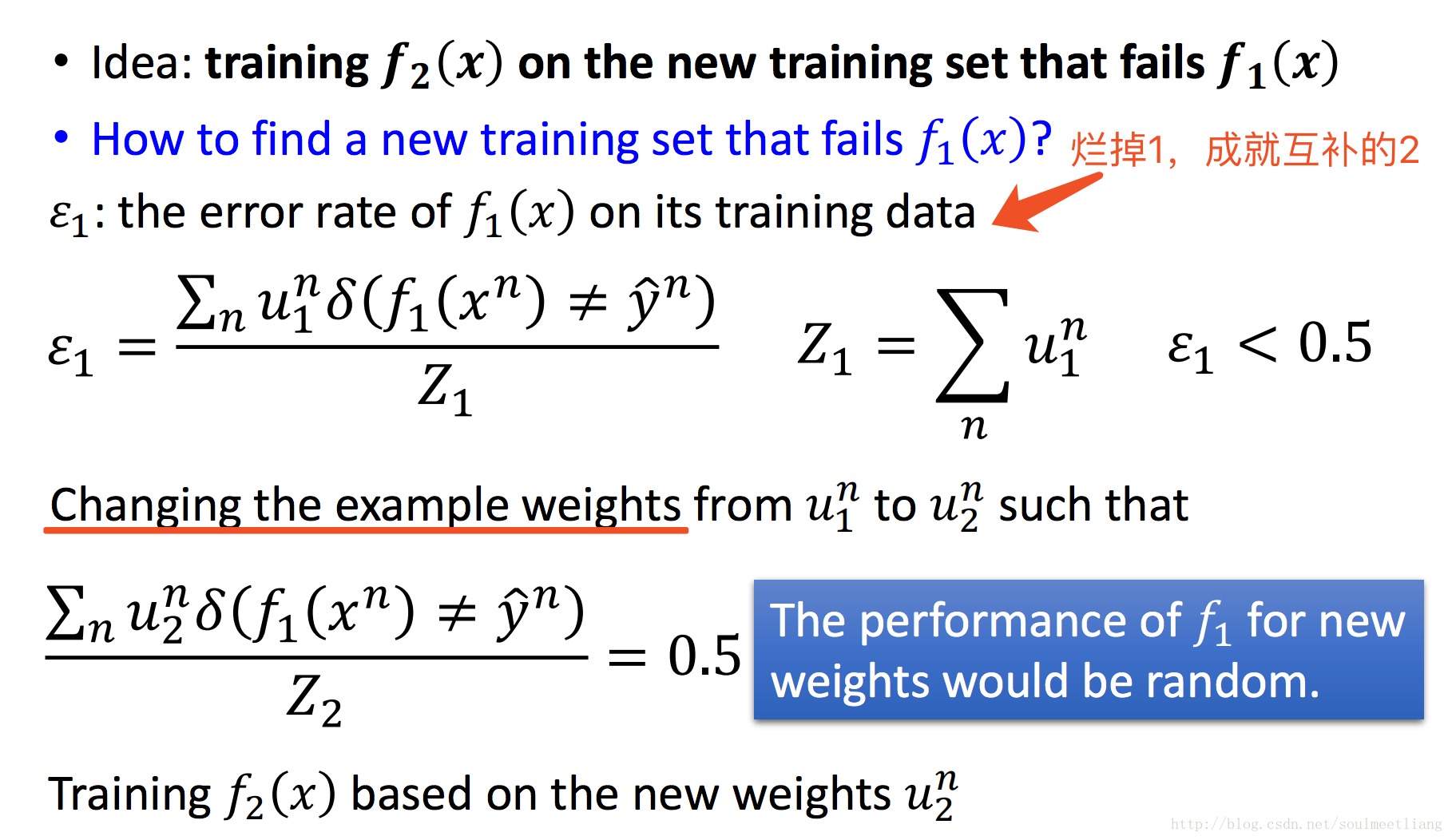
Re-sampling就相当于re-weighting，只不过re-sampling得到的weight只能是整数

Re-weighting 能使sample的次数不是整数而是小数。

## Adaboost

The performance of f1 for new weights would be random.

如何实现互补？



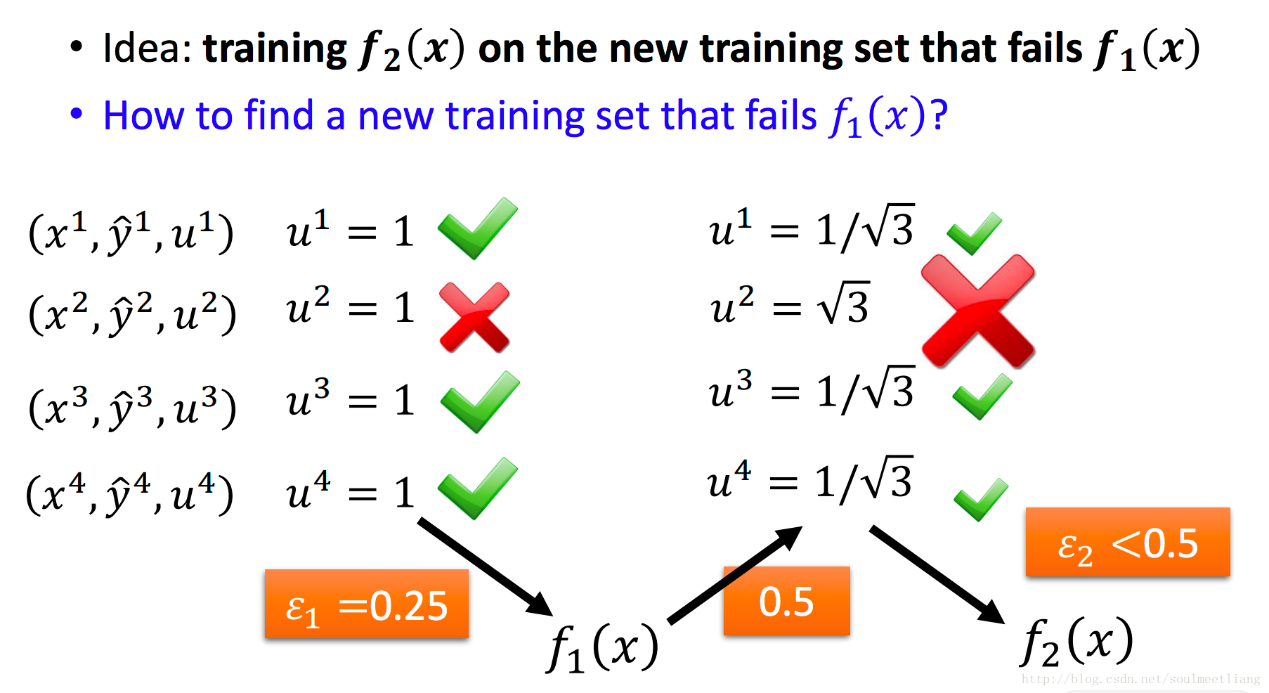
summation每个training set的weight（u）, n为example总数，

对的值为0，错的值为1

δ（delta）狄拉克函数，返回1或0，前面有提到过。

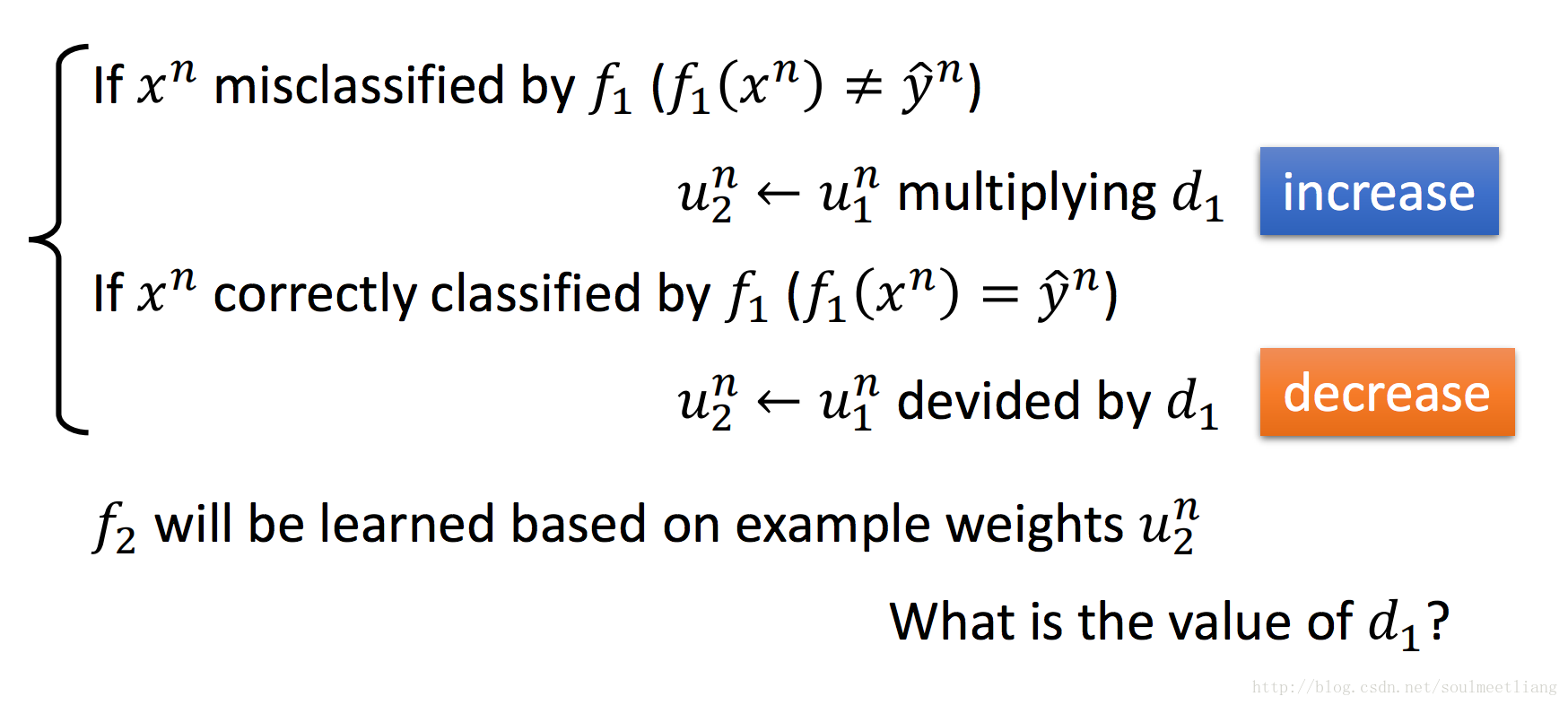
Normalization，将δ（epsilon）转化为小数

#### Re-weighting Training Data

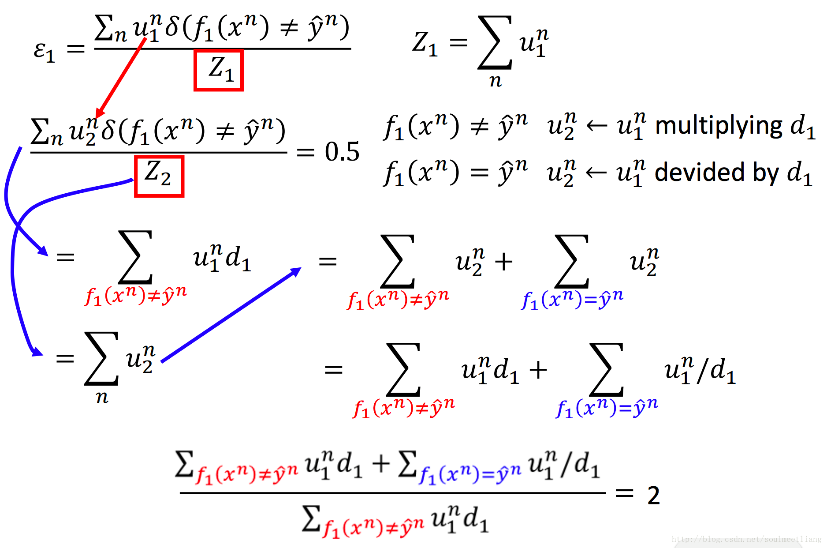


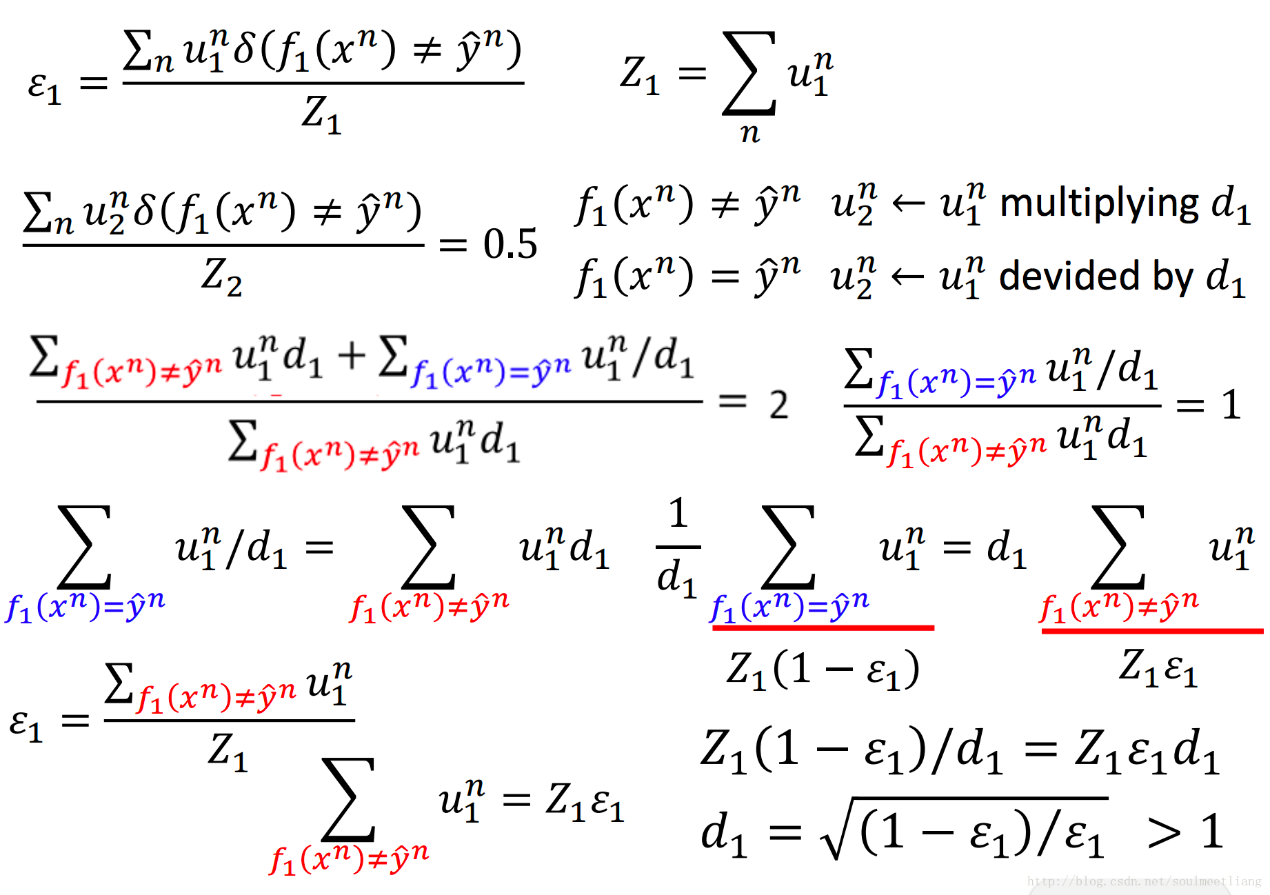
就比如，错的题分值变大，对的题分值变小，75分硬生生整成不及格。

实际操作（d1大于1）



那d1是多少呢？

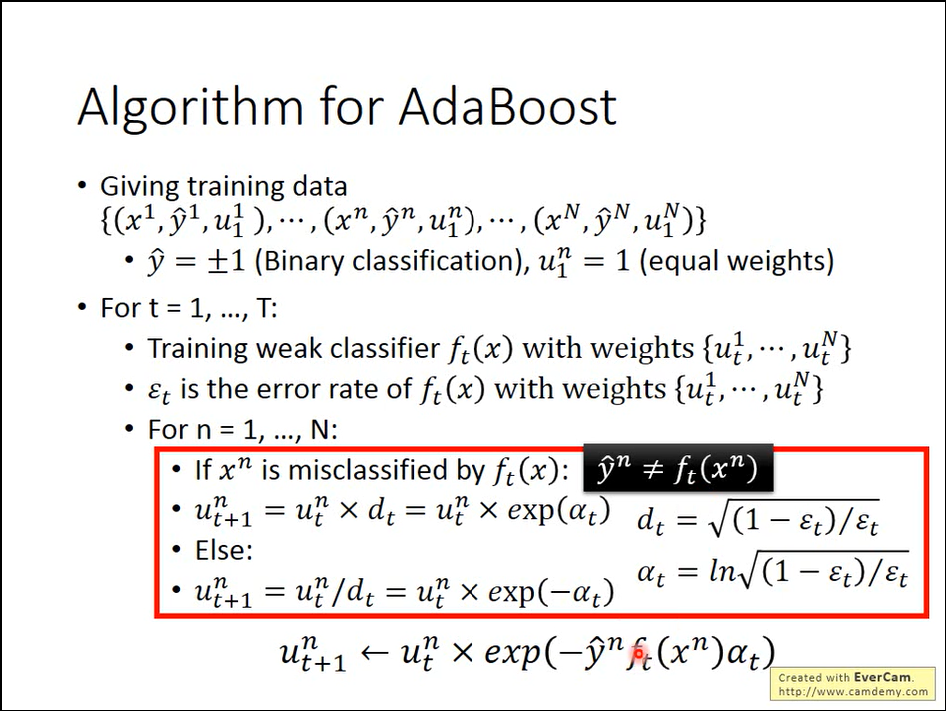




ε大于0.5

Error rating，已知

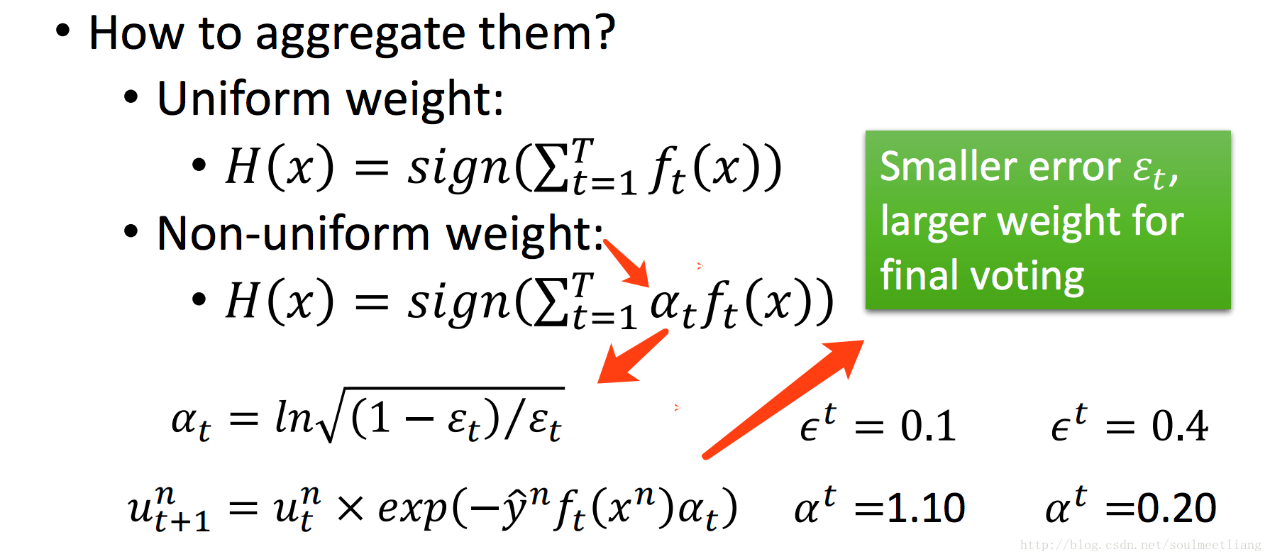
Algorithm for AdaBoost



Weight初始化为 1

如何用一个式子表示一正一负

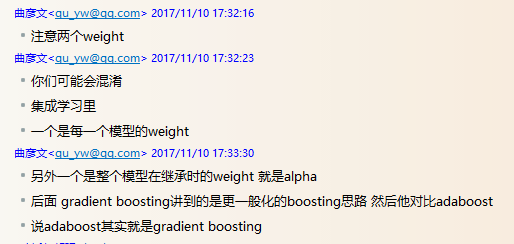
我们要把这些classifier通通aggregate集合起来，怎么集合呢？

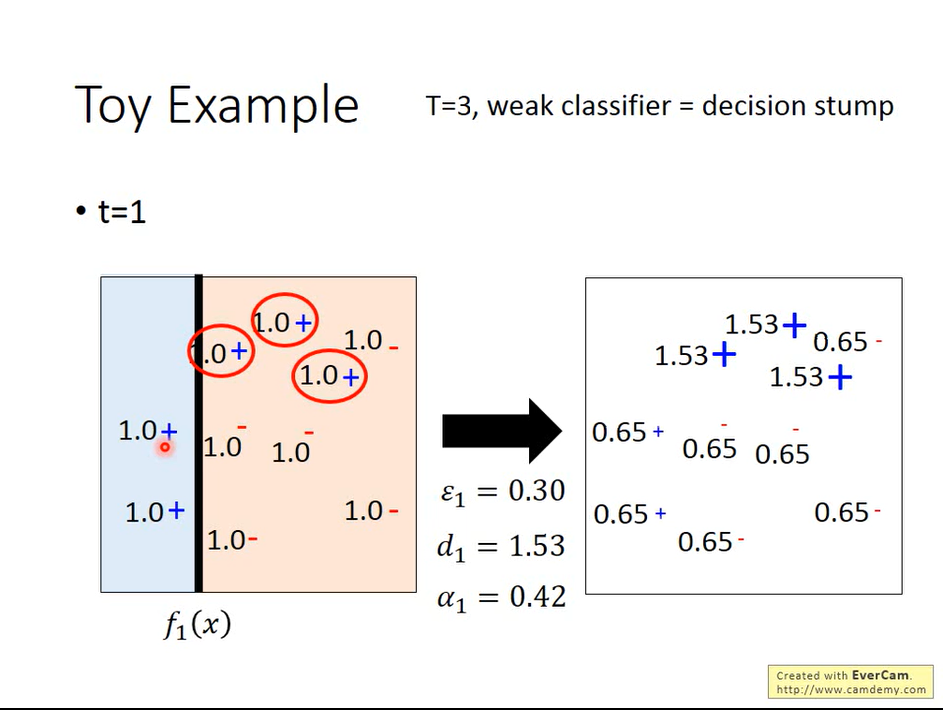


这里的weight是指进行下一次boosting（继承——最后加在一起）时要乘的weight（正比于f（分类器）判断的正确率），之前的weight是指每一次boosting，f1(x)->f2(x)时f1(x)每个x需要乘的weight（u1->u2）。

Smaller error，larger weight？

通俗理解就是，如果这个 f 本来就比较准确，那么最后算集合时投票权自然就大一点。

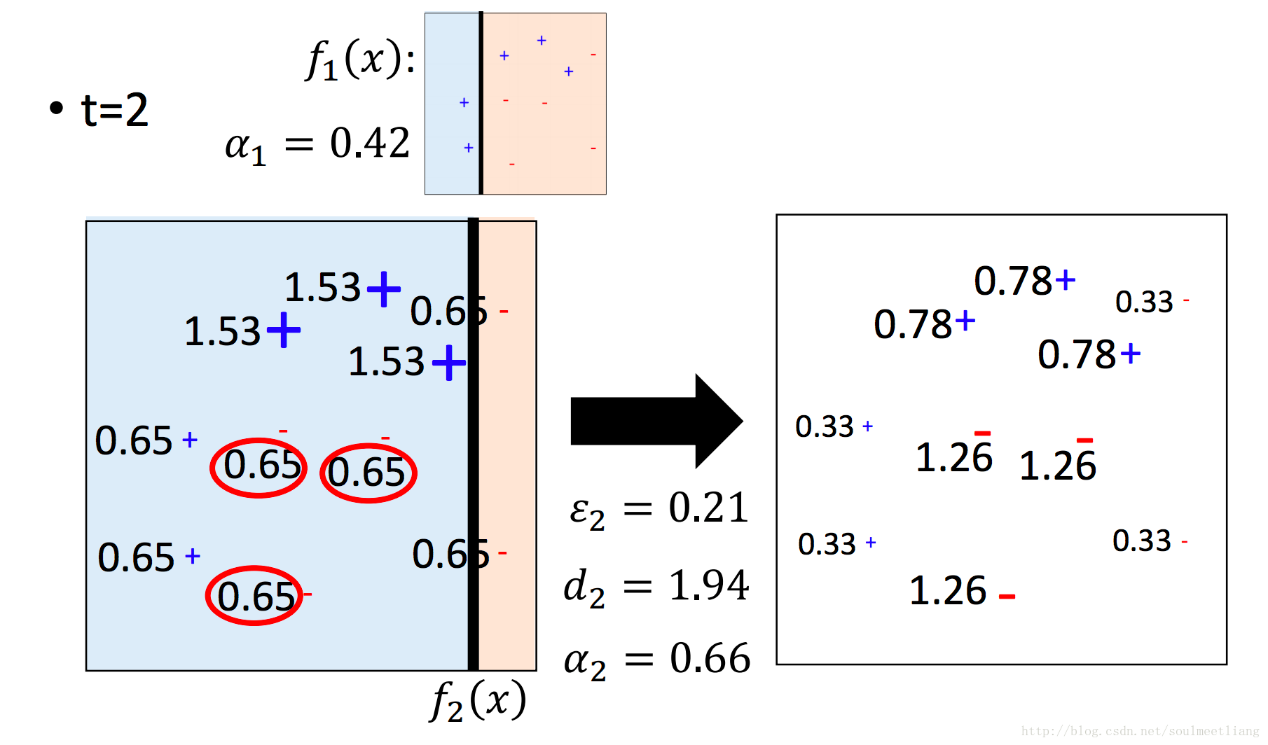


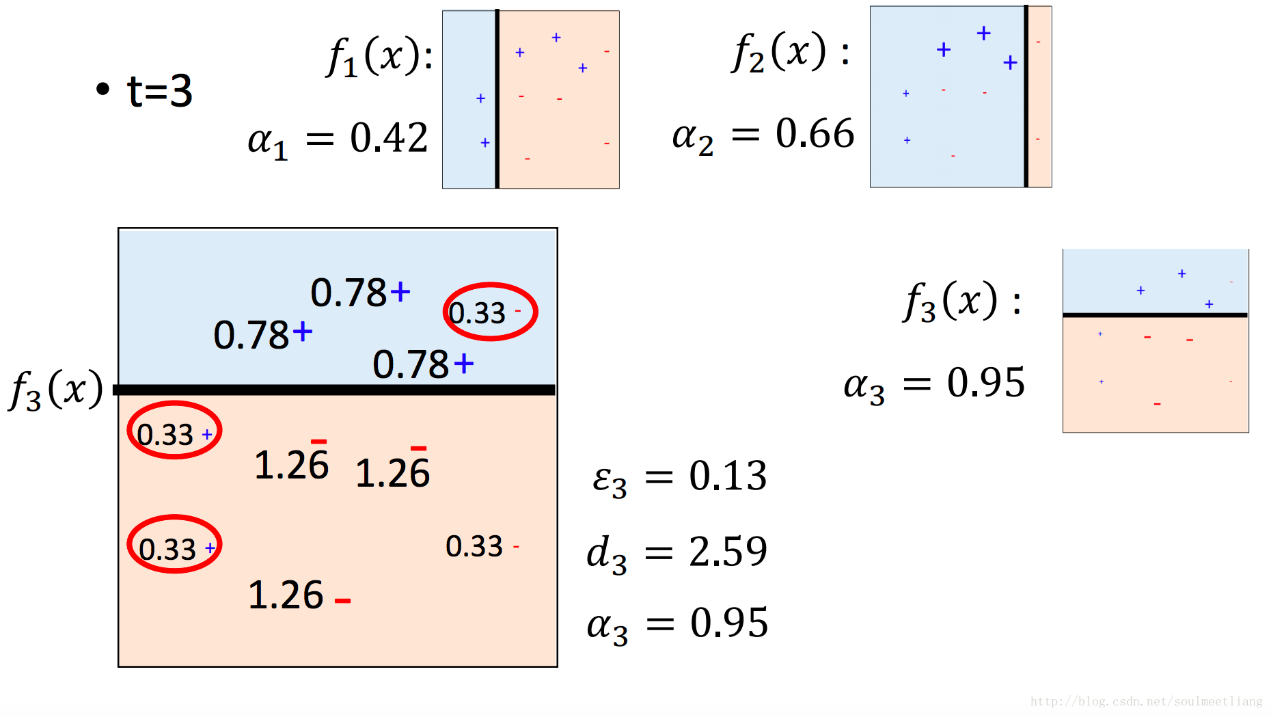


Decision stump比较weak and easy，所以拿来用在这里。

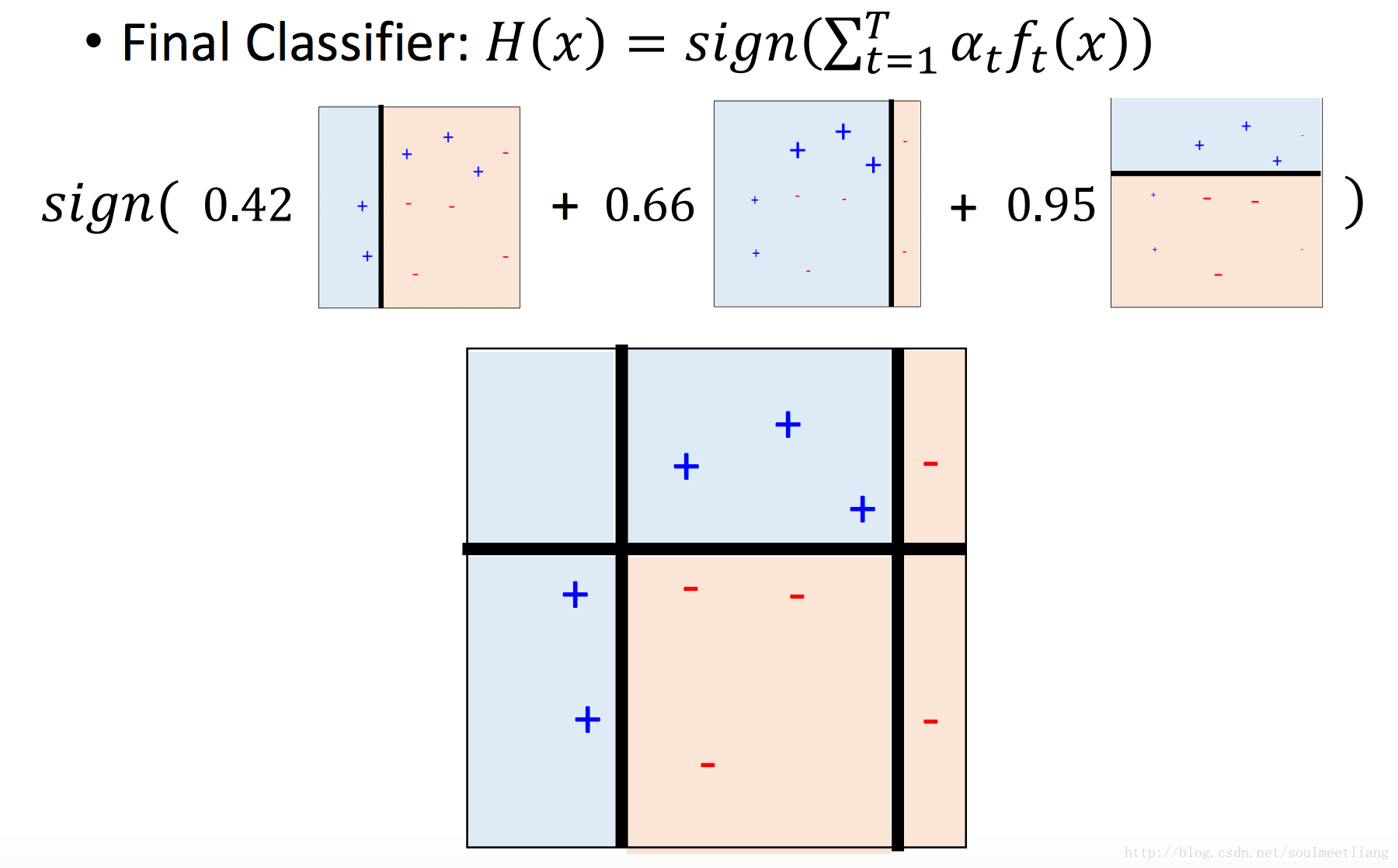
决策树桩（一刀切），也称**单层**决策树（a one level decision tree），**单层**也就意味着尽可对**每一列属性**进行一次判断（只对一列属性做一次判断）。

每个x的weight，f1(x)->f2(x)，\*|/d1





最后把3个classifier合起来：



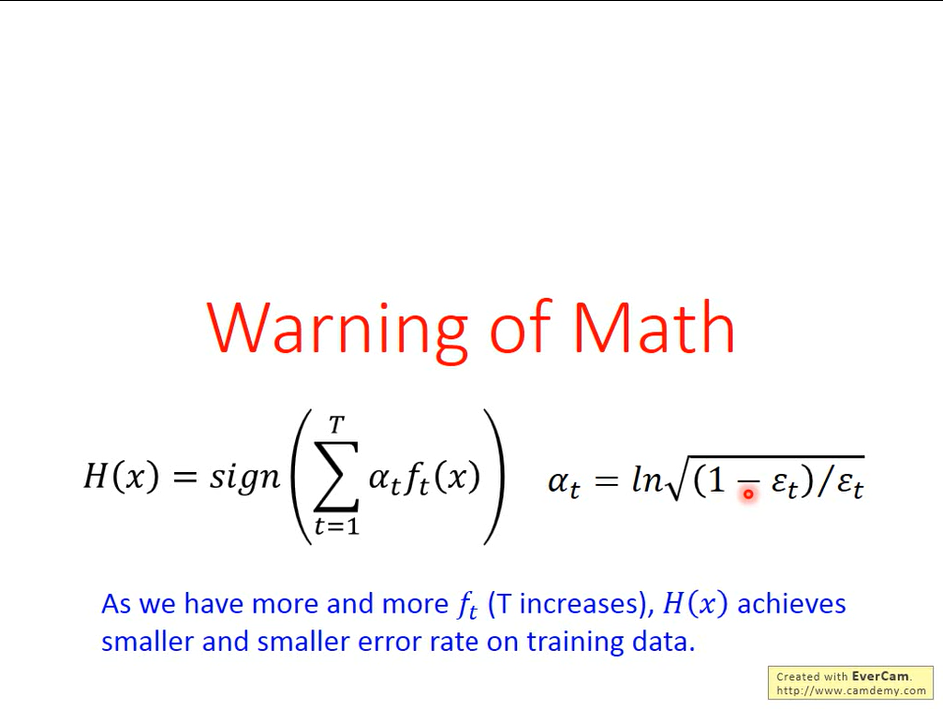
Weak classifier

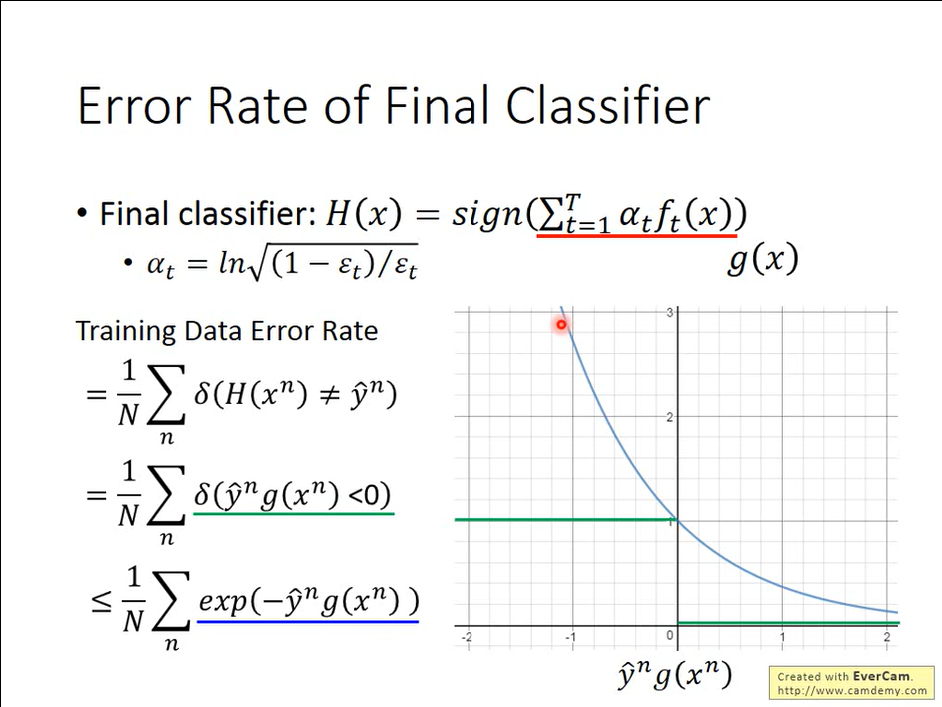
f(x)为1或-1，

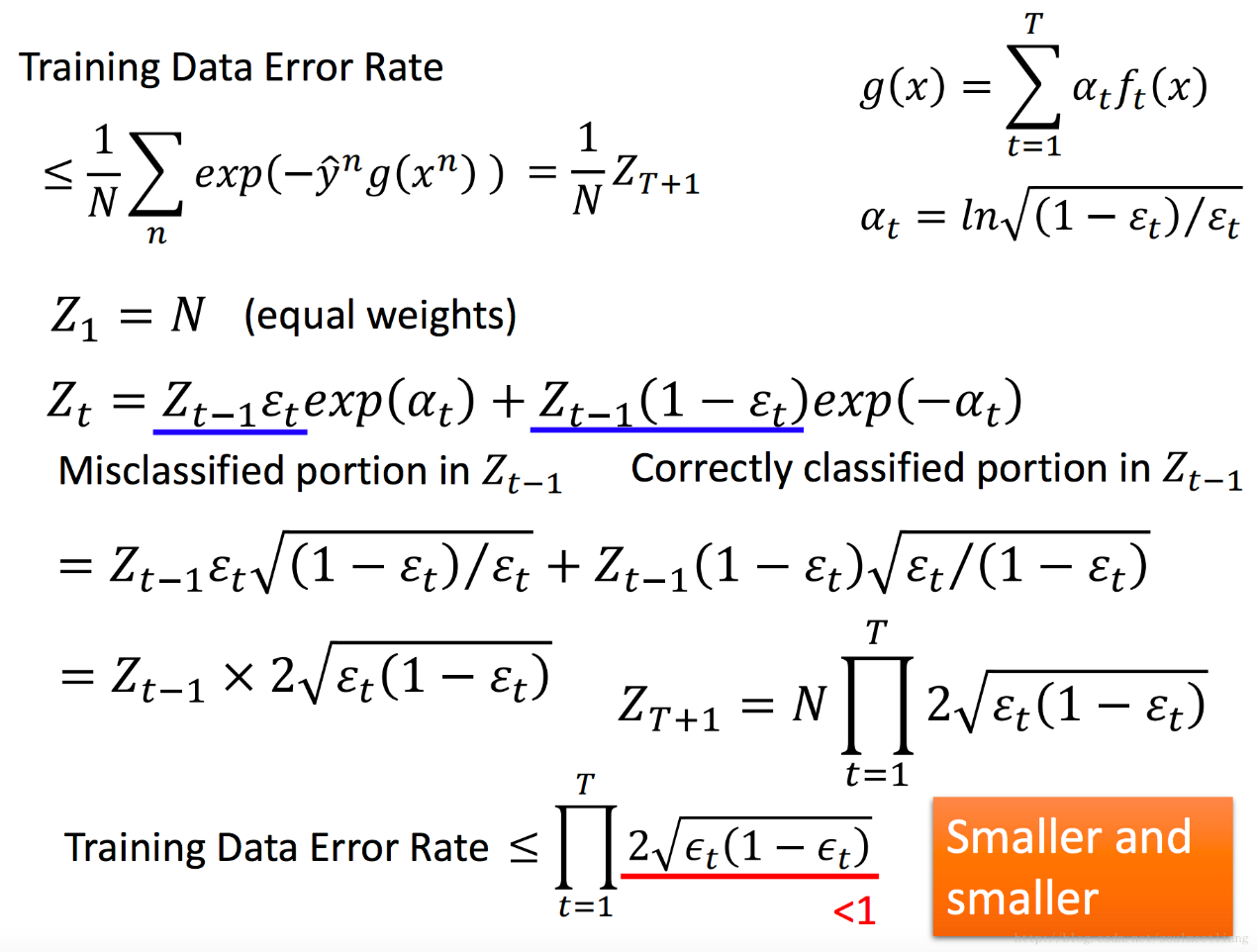
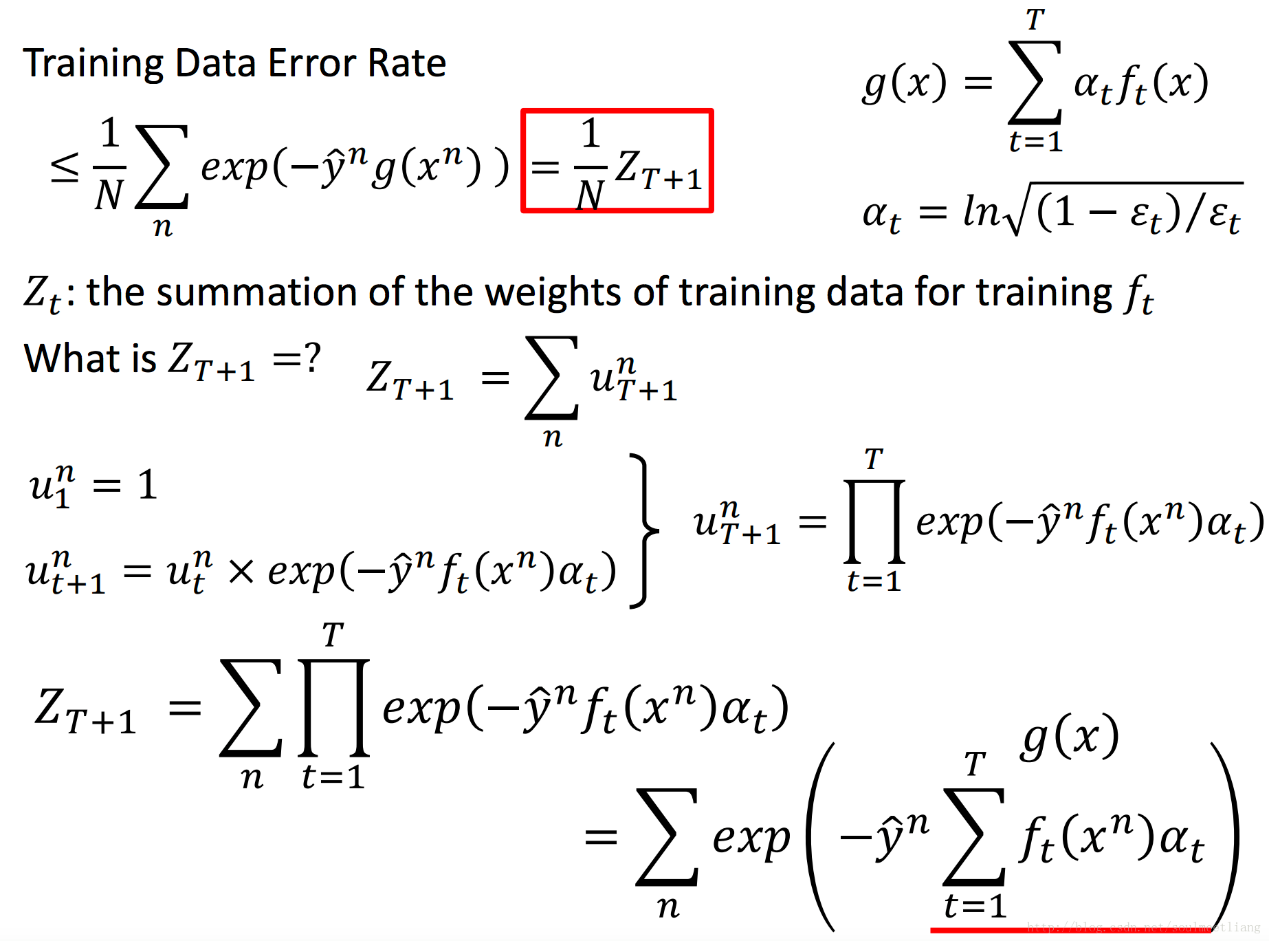
sum(αtft(x))最后的结果就是1或-1，举例：若分类结果为1的weight之和大于-1的，则结果是1

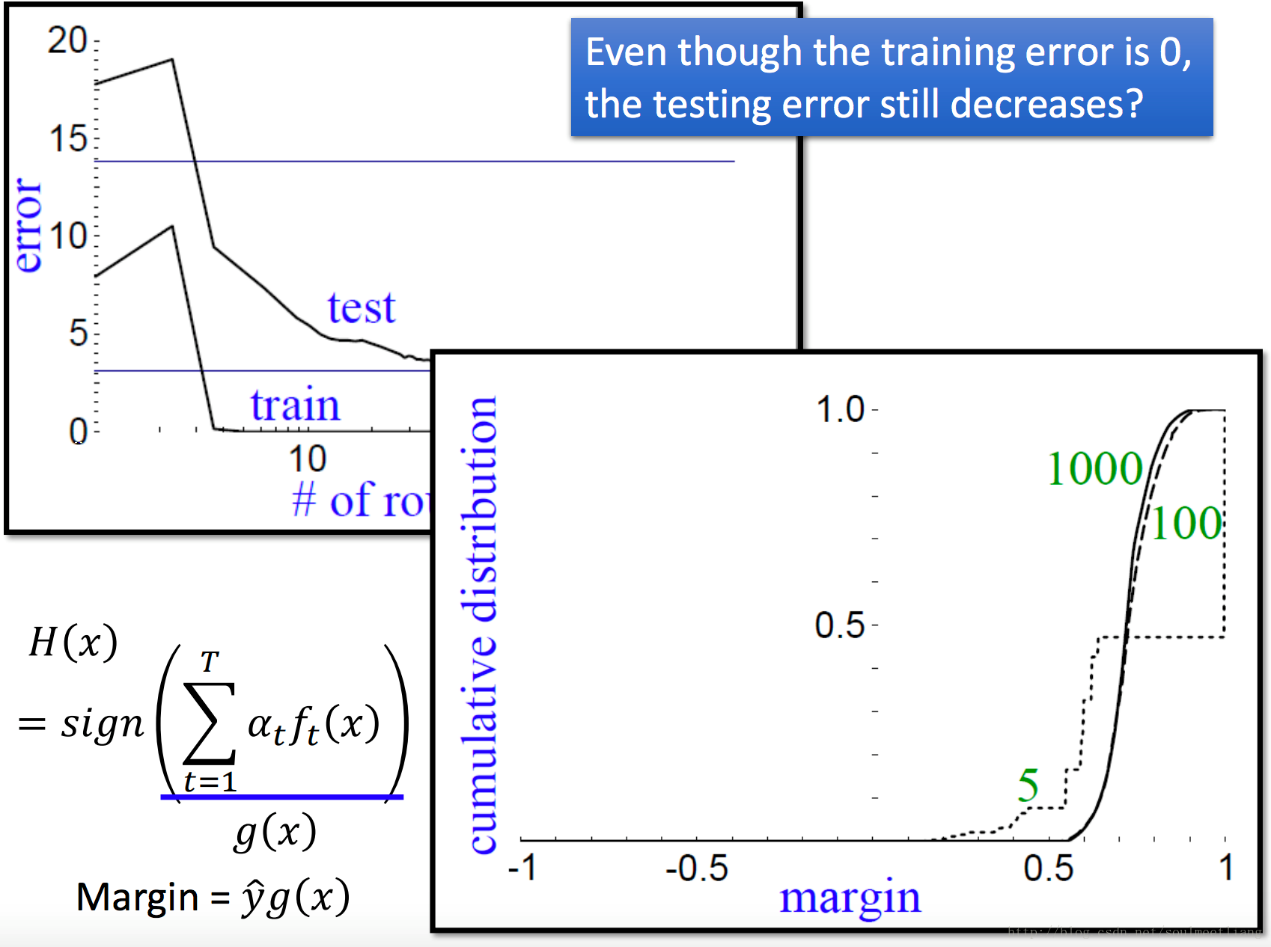
为什么是αt？

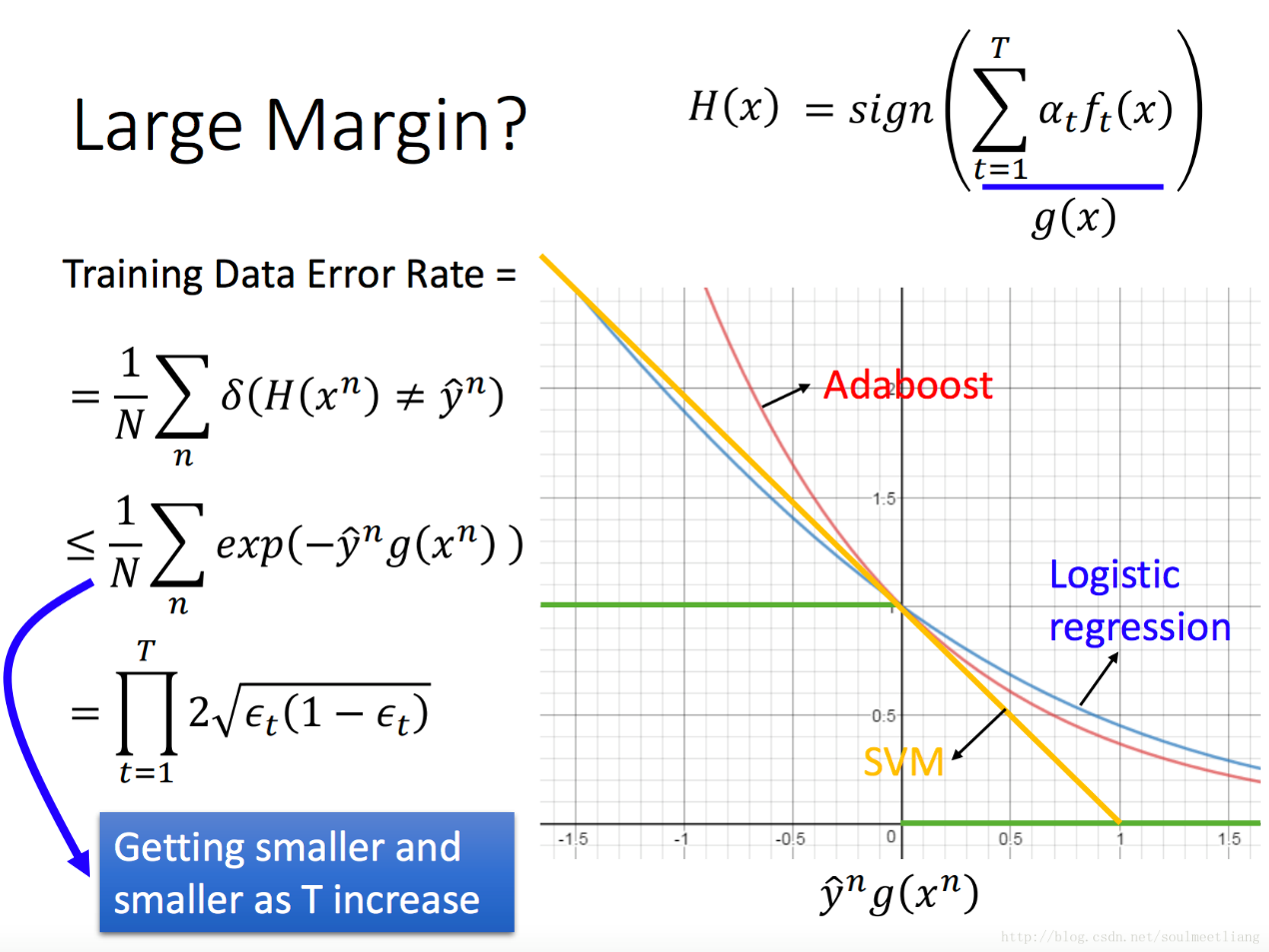
因为，αt=lnsqrt((1-ε)/ ε，与error rating ε成反比，即表示正确率





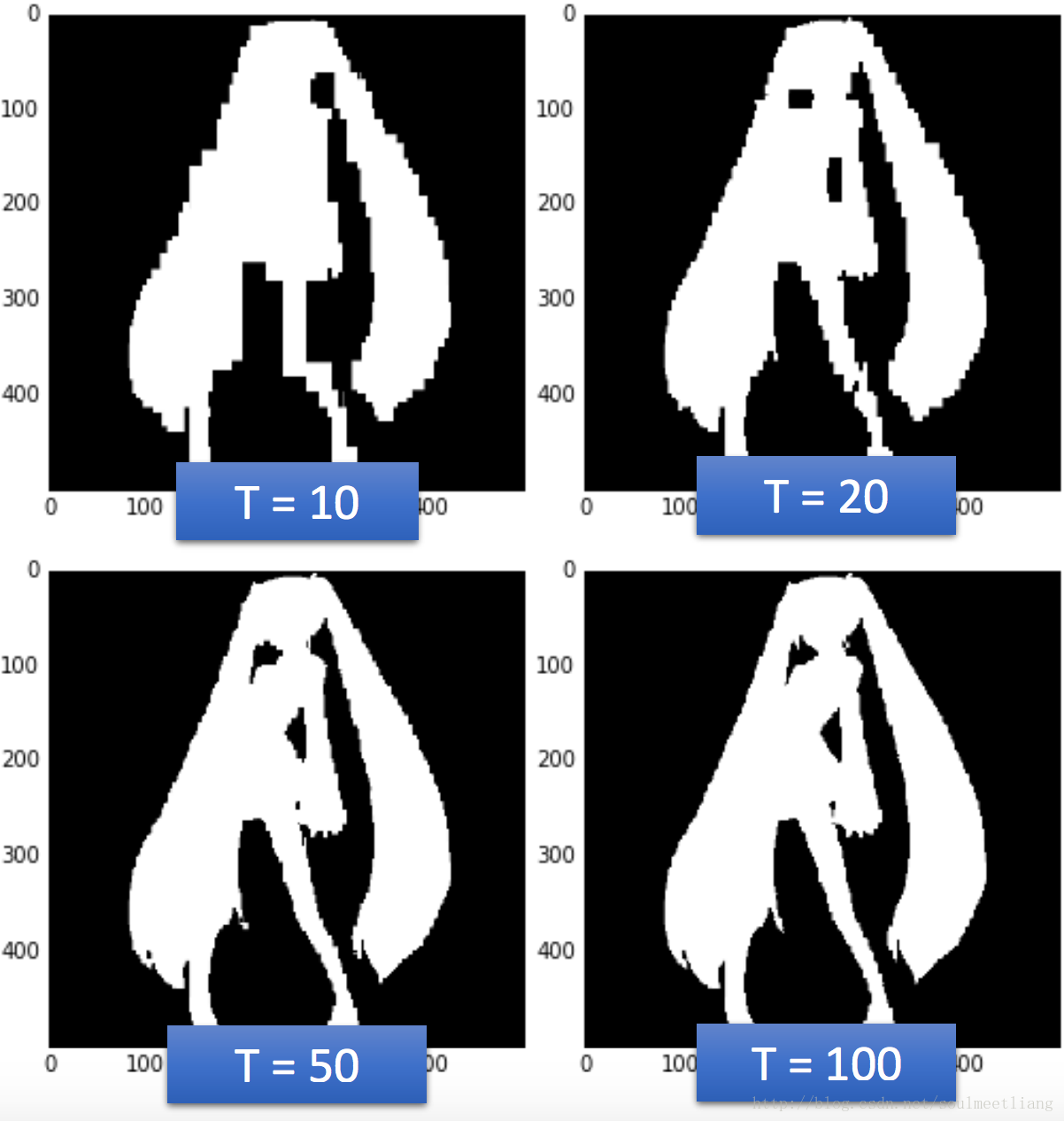
-AdaBoost 有一个神奇的现象，下图，当classifer越来越多，training data 的error rate 很快变成0，但奇怪的是 test data 的error rate 依然会下降。右图，后来有人分析margin，classifer的增多会把margin往右推。





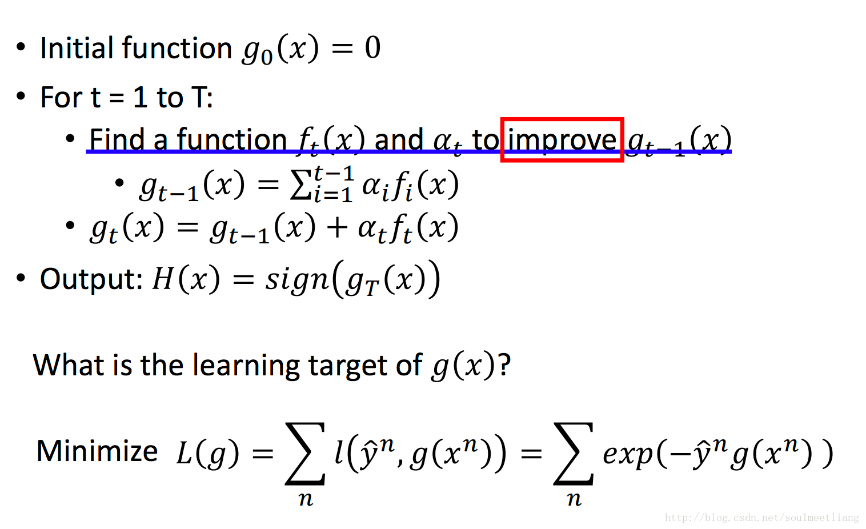
For example：

Adaboost + Decision Tree (depth = 5)

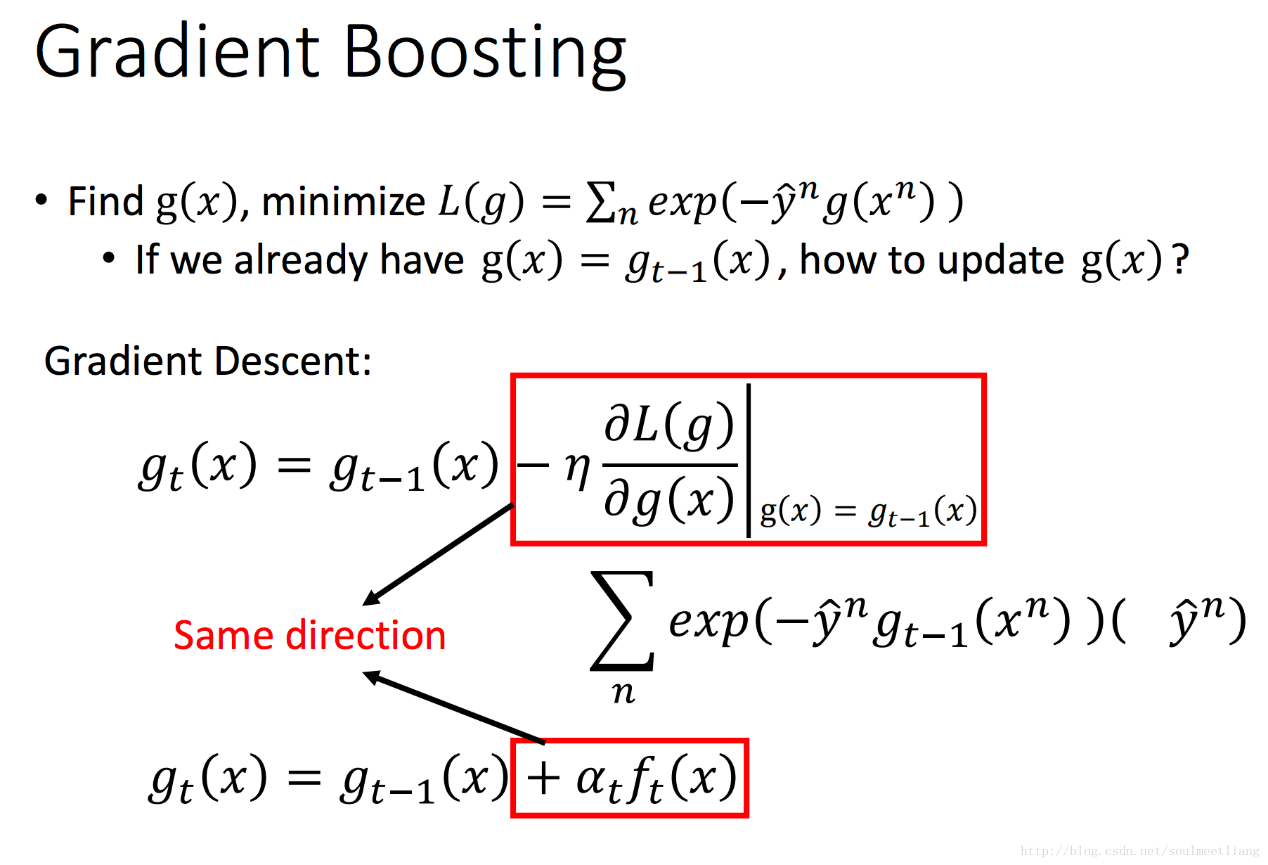


### General Formulation of Boosting

Adaboost是一个特例。Boosting的一种。

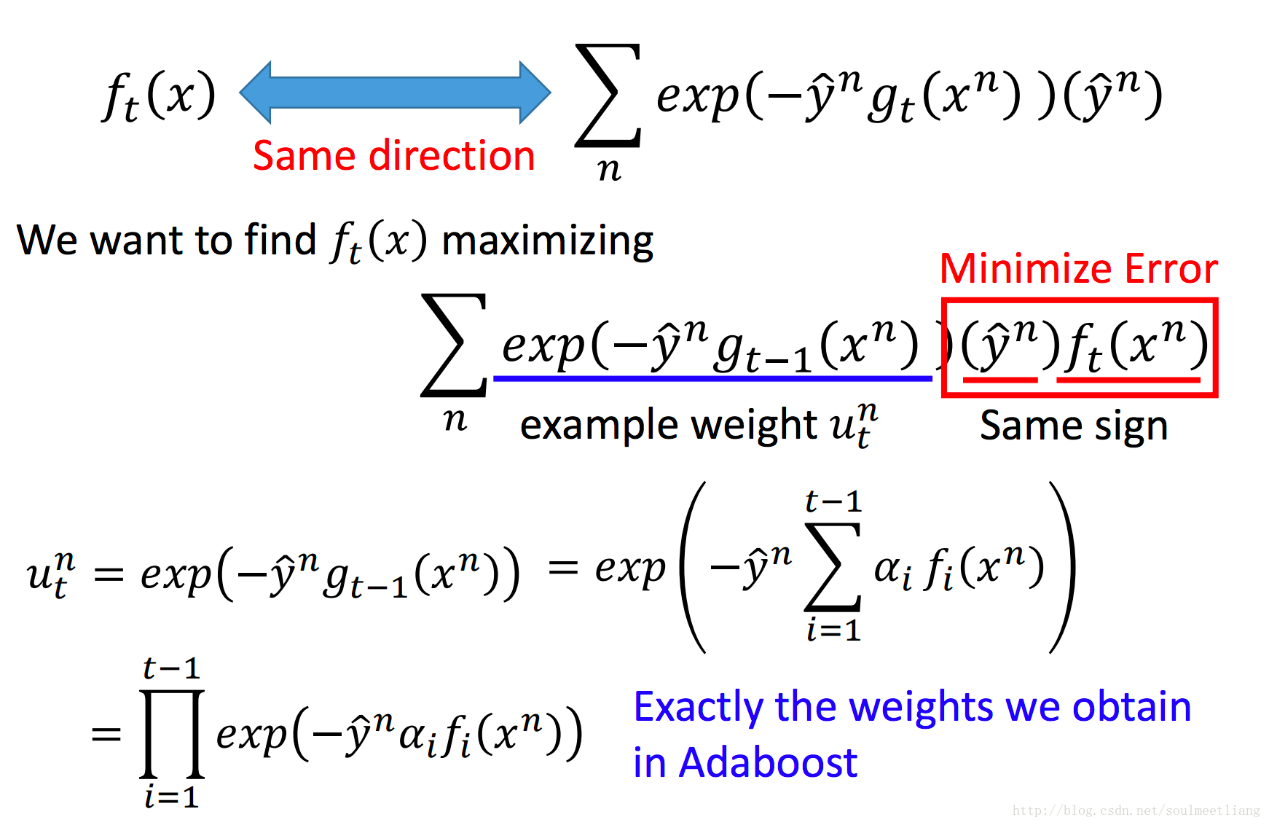


同号，则只要使g(xn)最大，就可以minimize L(g)



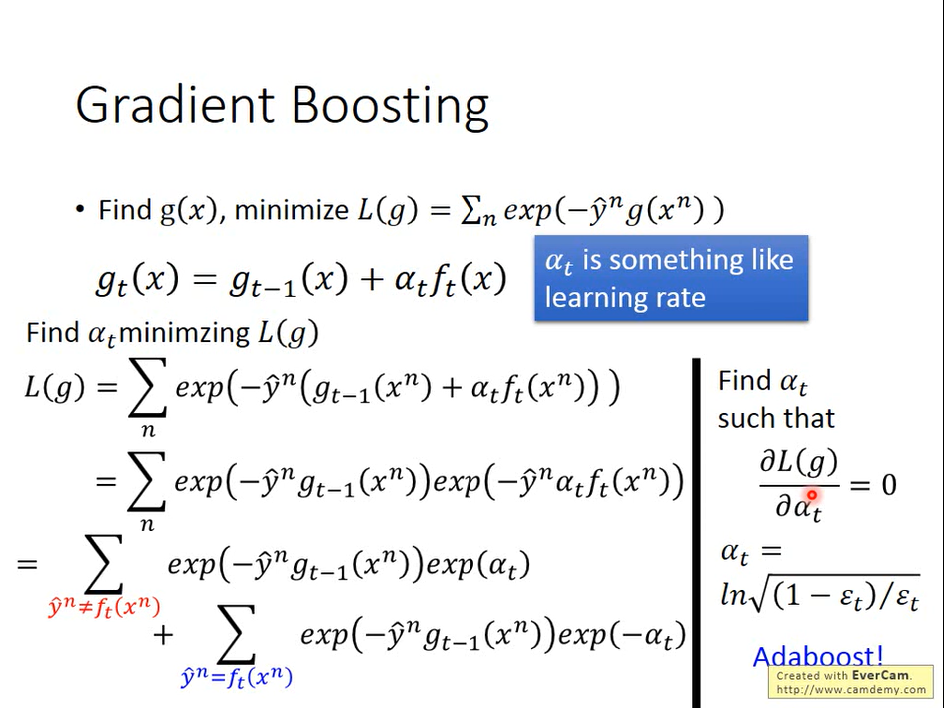
求导后的结果

这里的两者相等，就表明了adaboosting和gradient boosting本质上是一样的



同号，值就为1，只要左边的exp最大，就可以使g(x) update的效果最好（最大）

最后就变成了update α来minimize L(g)



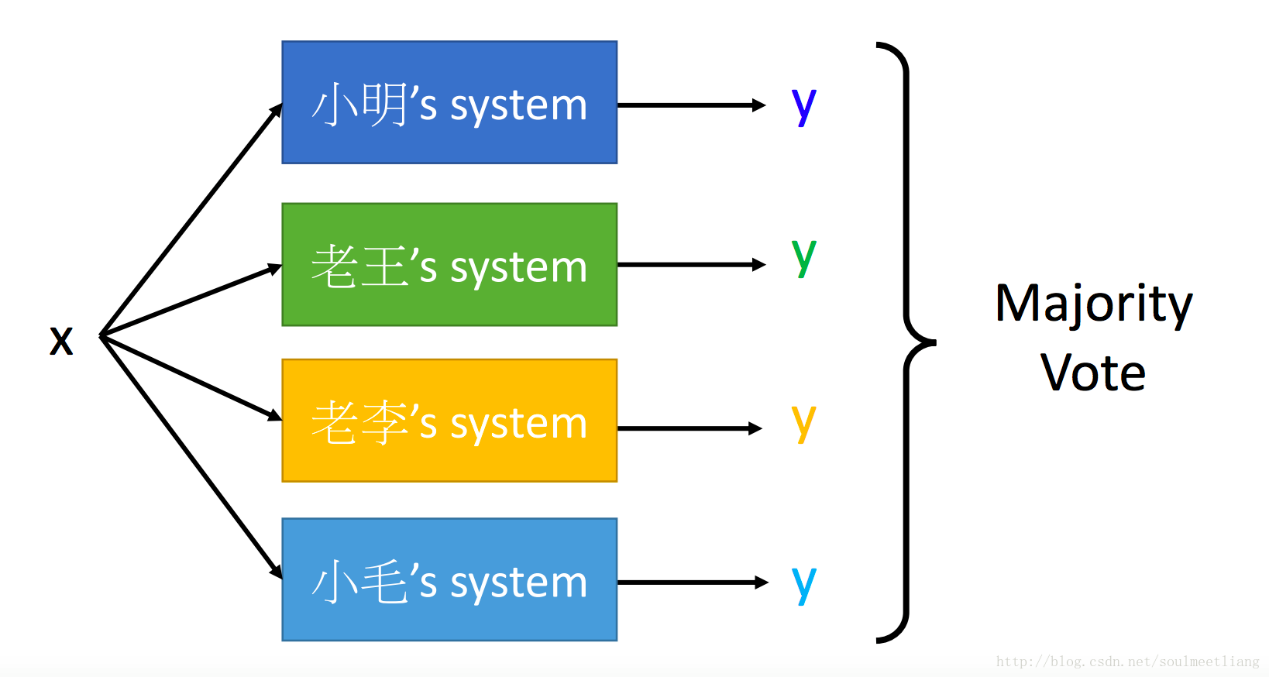
αt是ft(x)的weight，也是这里的learning rate，为最大利用ft(x),就去计算α的optimization

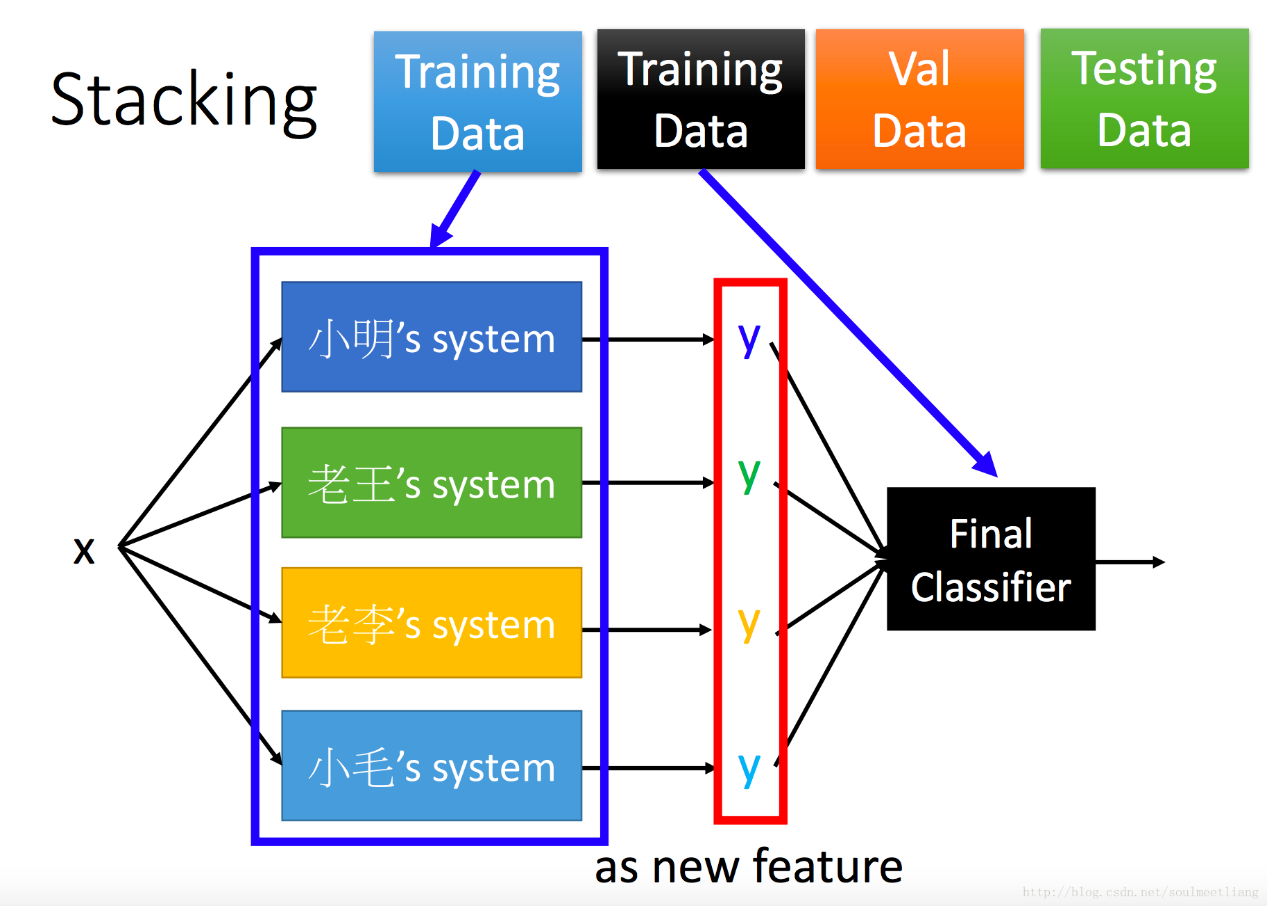
Gradient Boosting可以任意更改object（loss） function，永远可以定其他的object function，就可以创建出不一样的方法来实现boosting，并且Gradient boosting的运算速度更快。

**Gradient boosting**（Gradient boosting Decision tree又叫MART, Treenet)：Boosting是一种思想，Gradient Boosting是一种实现Boosting的方法（也可以说是模型），它主要的思想是，每一次建立模型是在之前建立模型损失函数的梯度下降方向。损失函数(loss function)描述的是模型的不靠谱程度，损失函数越大，则说明模型越容易出错。如果我们的模型能够让损失函数持续的下降，则说明我们的模型在不停的改进，而最好的方式就是让损失函数在其梯度（Gradient)的方向上下降。

# Ensemble: Stacking

## Voting





原理和boosting是一样的