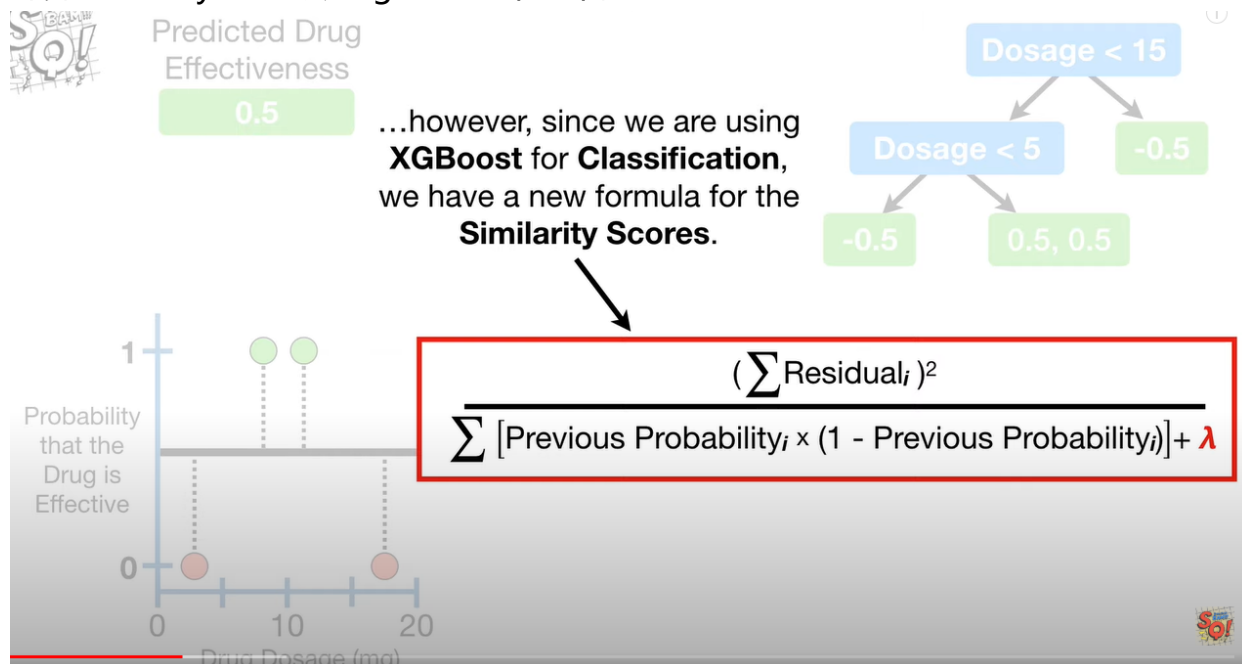


和解决Regression问题一样，要下给出一个预估的值，本例子中假设是0.5（4个样本中2个为正例，即effective）。可以代表概率。比如在视频中，0.5代表有50%的概率药物是有效的。

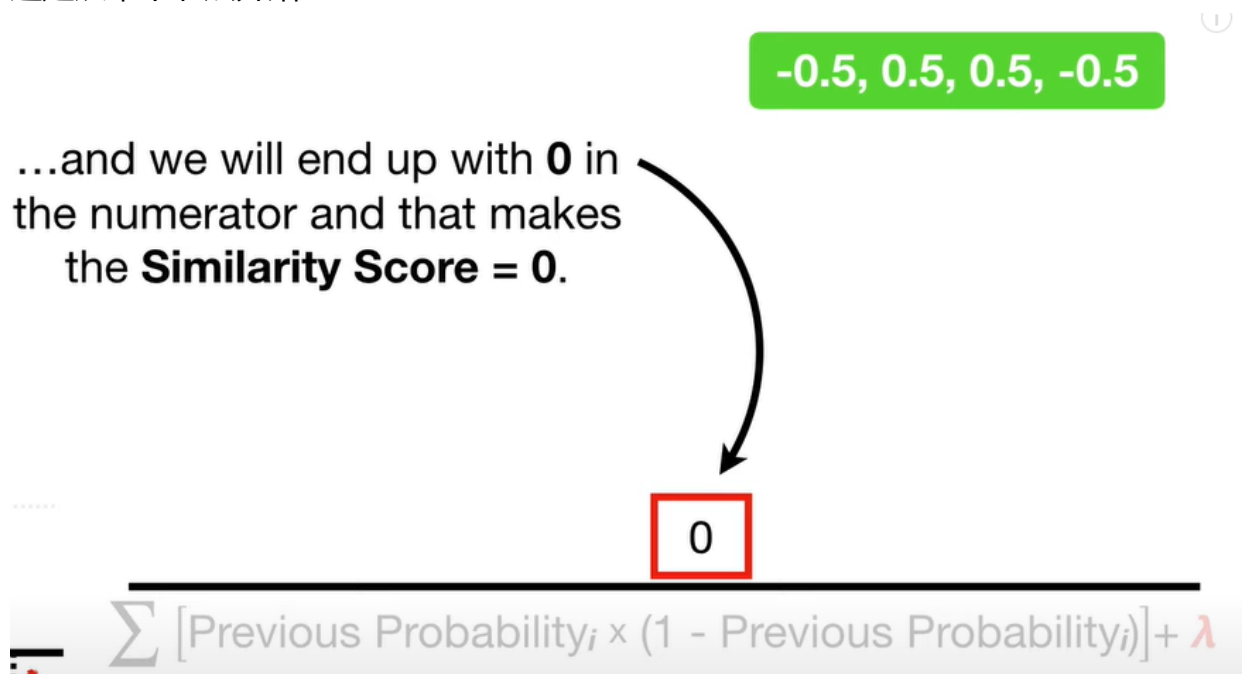
计算similarity的公式跟regression中的不同：



正类样本的概率是1，负类样本的概率是0。

如何构建树：

还是从单个节点开始

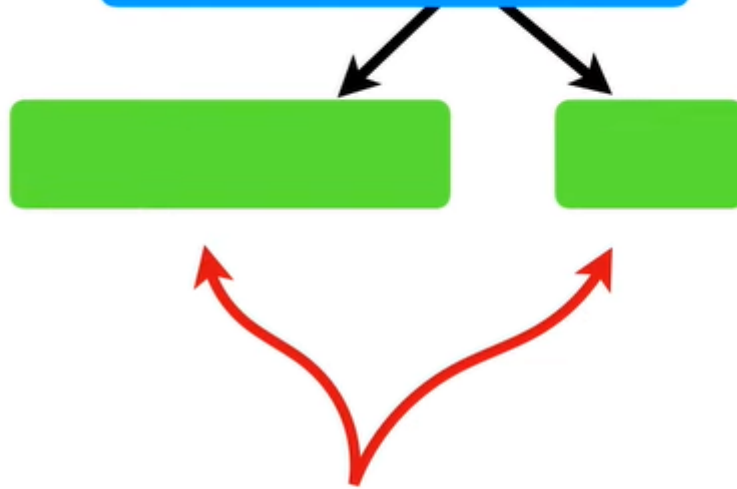


有趣的是，在这个例子中，residuals刚好相互抵消，因此分母不用计算了，最初的similarity是0。

接着又是对residuals进行聚类

Similarity = 0

-0.5, 0.5, 0.5, -0.5



Now we need to decide if we can do a better job clustering similar **Residuals** if we split them into two groups.

尝试用dosage < 15 (即最后两个样本的均值) 作为threshold



Predicted Drug Effectiveness  
0.5

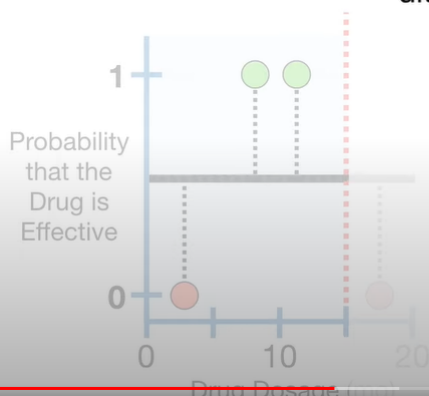
Now, just to keep things simple, we'll let  $\lambda = 0$ . However, you know from **Part 1** that  $\lambda$  (**lambda**) reduces the **Similarity Score**, which ultimately makes leaves easier to prune.

Similarity = 0

Dosage < 15

-0.5, 0.5, 0.5

-0.5



$$(-0.5 + 0.5 + 0.5)^2$$

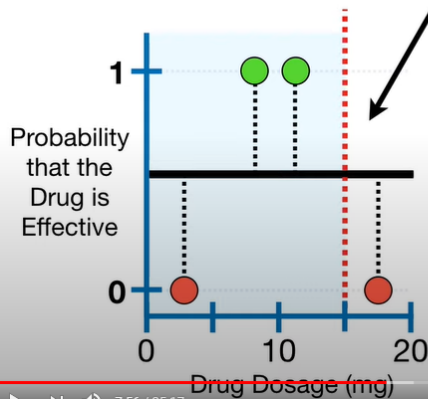
$$(0.5 \times (1-0.5)) + (0.5 \times (1-0.5)) + (0.5 \times (1-0.5)) + 0$$

previous probability都是0.5, 即最初的预测。

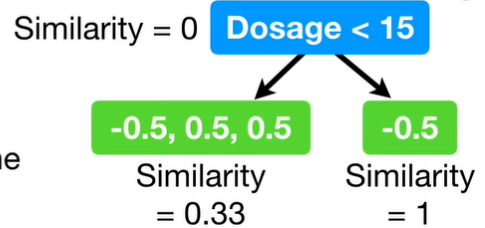


Predicted Drug Effectiveness  
0.5

So when we split the **Observations** based on the threshold **Dosage < 15**, **Gain = 1.33**.



$$\text{Gain} = 0.33 + 1 - 0 = 1.33$$

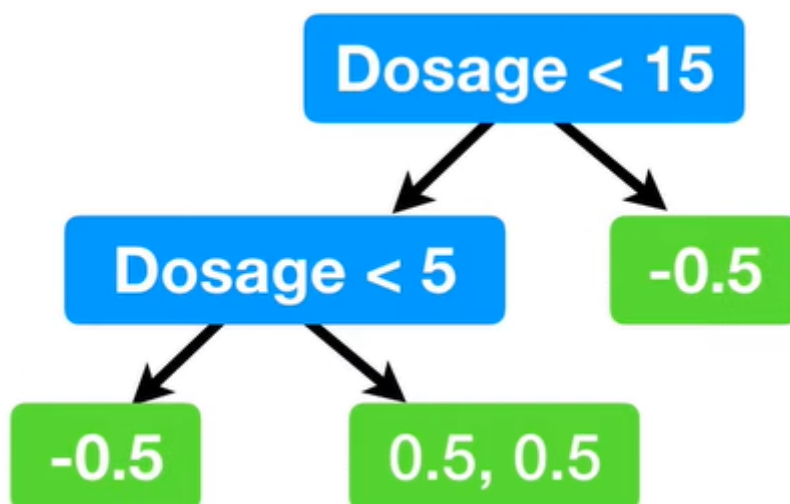


$$\text{Gain} = \text{Left}_{\text{Similarity}} + \text{Right}_{\text{Similarity}} - \text{Root}_{\text{Similarity}}$$

用相同方法计算右子节点的similarity后，得到dosage<15作为threshold的Gain值。（假设dosage<15的Gain值就是最大的，因此用dosage<15作为root节点的分裂条件）

继续对左子节点进行分裂

尝试使用dosage<5和dosage<10作为分裂判断条件，dosage<5的Gain较大，因此使用其作为分裂条件



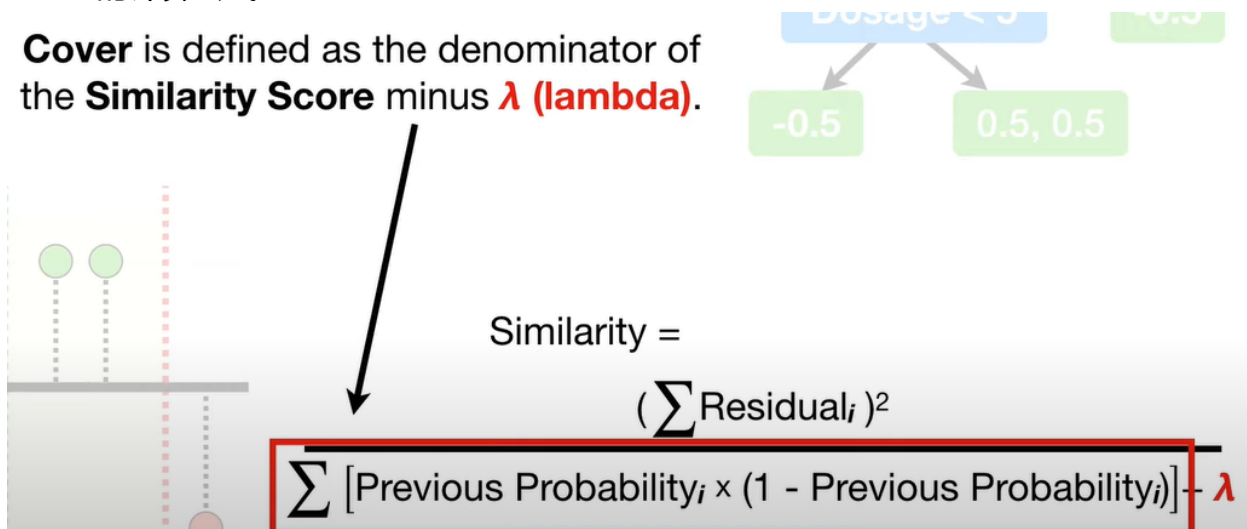
这个例子中限制XGB的树最大深度为2，就不继续分裂了。

除此之外，还可以对每个叶节点的residuals的最少数量进行限制，称为Cover。

# The minimum number of **Residuals** in each leaf is determined by calculating something called **Cover**.

Cover的计算公式:

**Cover** is defined as the denominator of the **Similarity Score** minus  $\lambda$  (**lambda**).



Similarity =

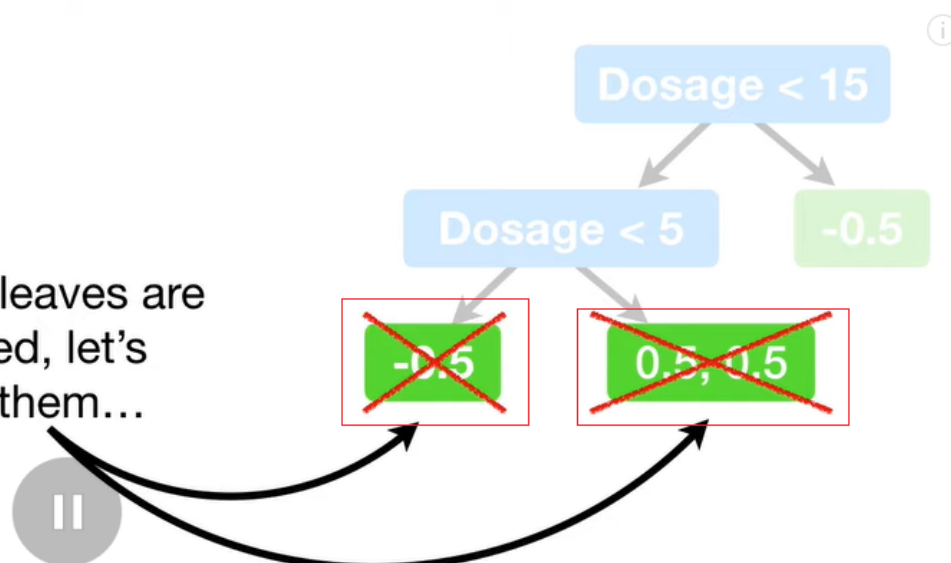
$$\frac{(\sum \text{Residual}_i)^2}{\sum [\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)] - \lambda}$$

即classification问题中, similarity分母部分的前半部分, 即红框中部分 (求和是该叶节点的所有residuals, 即分配到该叶节点的所有样本)

注意, 无论在regression还是classification中, cover的最小值都为1。

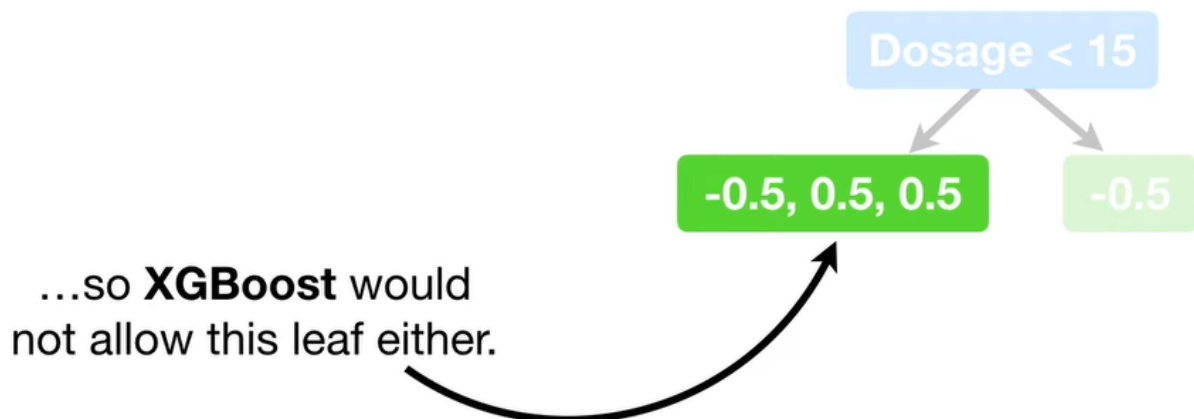
然而, 在下面这个树中, 通过计算可以得知, 红框中的叶节点的Cover值都小于1, 因此这两个节点是不允许出现的。两个叶节点合并回到原来的父节点。

Since these leaves are not allowed, let's removed them...





但是， $(-0.5, 0.5, 0.5)$ 的节点的Cover值仍然小于1，又被剪掉。同样的，右子节点也被剪掉。



$$\text{Cover} = 3 \times [0.5 \times (1 - 0.5)] = 0.75$$

这样就只剩下根节点。

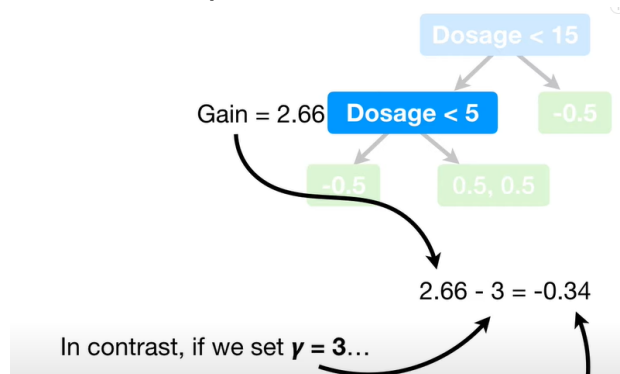
但是XGB中规定树不能只有一个根节点。因此，我们可以设定Cover的最小值为0（即所有节点的Cover值必须大于0）。

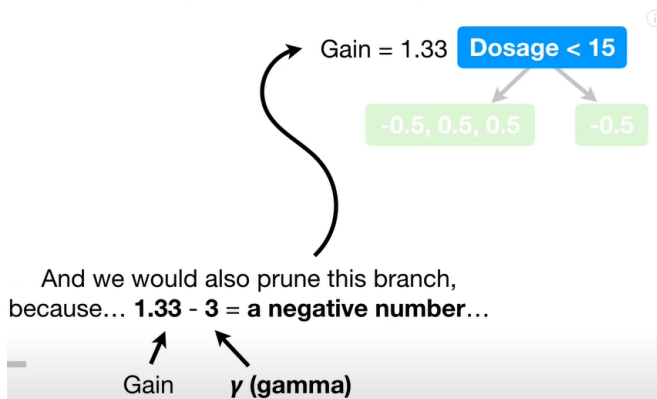
这样就可以继续讨论剪枝的问题。

如何进行剪枝？

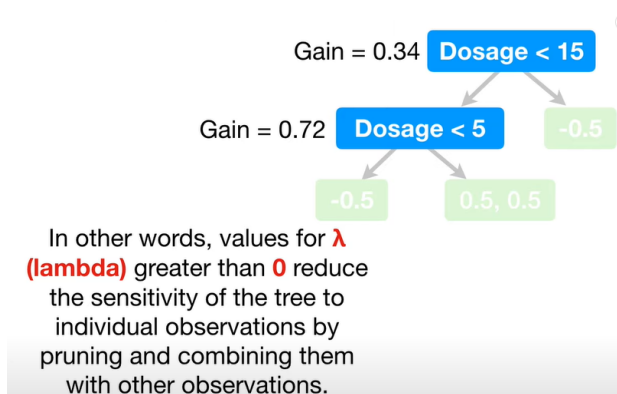
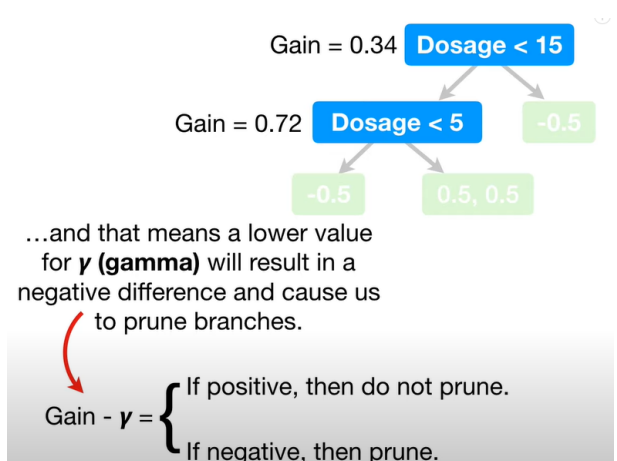
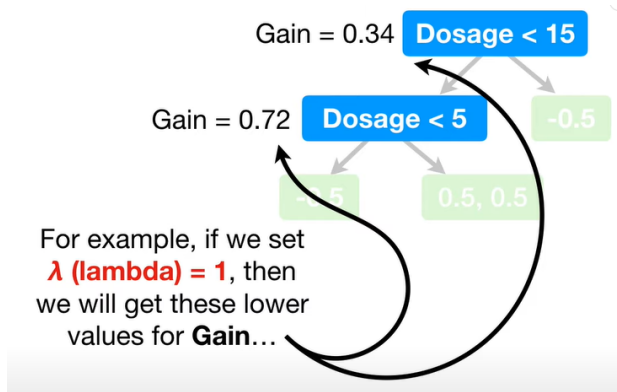
通过similarity计算Gain，再计算 $\text{Gain} - \gamma$ ，如果结果为正，则不需要剪枝，反之剪枝。

如果现在假设 $\gamma = 3$ ，则整个树都被剪掉，就剩下原来的预测0.5。



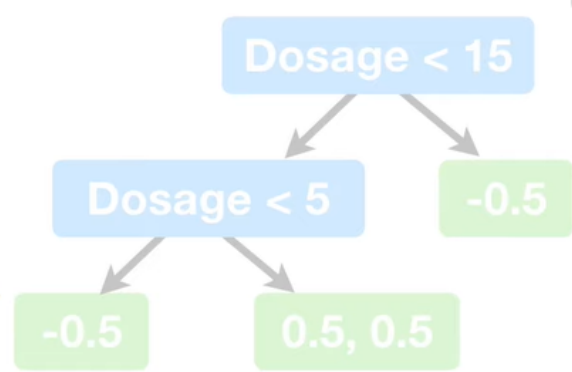


注意之前假设的是 $\lambda=0$ .现在可以假设 $\lambda=1$ . $\lambda$ 越大, similarity越小, 则Gain也越小。



计算每个节点的输出值

计算公式跟Gradient Boost的大体相同, 分母部分多加正则化项



**NOTE:** With the exception of  $\lambda$  (lambda), the **Regularization Parameter**, this is the same formula we used for unextreme **Gradient Boost**.

$$\frac{(\sum \text{Residual}_i)}{\sum [\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)] + \lambda}$$

当 $\lambda=1$ 时，输出值为-0.4，当 $\lambda=0$ 时，输出值为-2.

...then the

**Output Value = -0.4**, which is closer to zero than **-2**, when  $\lambda = 0$ .



$$\frac{-0.5}{0.5 \times (1 - 0.5) + 1} = -0.4$$

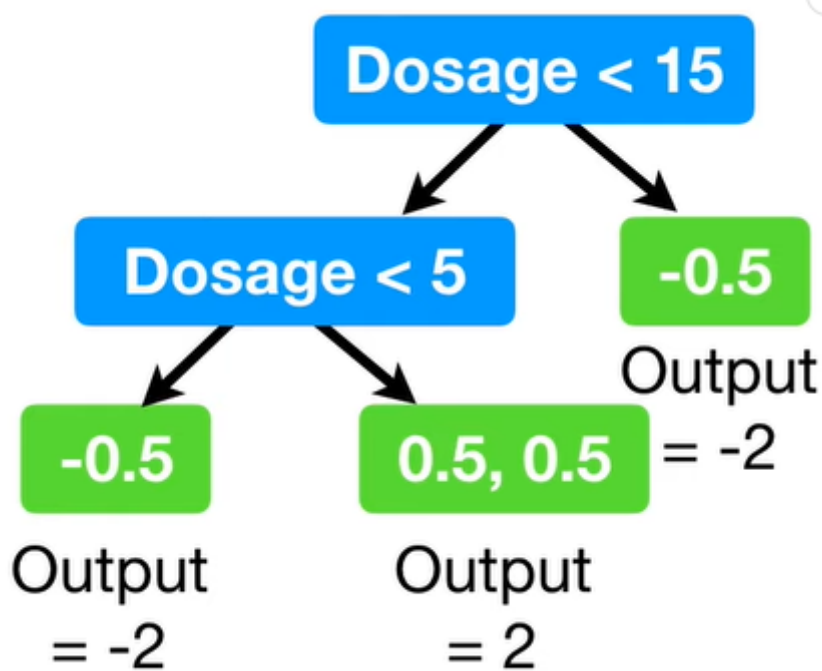
当 $\lambda$ 越大时，节点的输出值会越小。当做预测时，节点的输出值变小，预测值也会变小，从而达到解决过拟合的问题。

In other words, when  $\lambda > 0$ , then it reduces the amount that this single observation adds to the new prediction.



Thus,  **$\lambda$  (lambda)**, the **Regularization Parameter**, reduces the prediction's sensitivity to isolated observations.

最后计算出第一棵树的所有叶节点的输出值



如何做预测（更新）

和其他boosting方法一样，probability要转换为log(odd)

$$\frac{p}{1-p} = \text{odds}$$

$$\log\left(\frac{p}{1-p}\right) = \log(\text{odds})$$



Predicted Drug Effectiveness

0.5

$$\log\left(\frac{0.5}{1 - 0.5}\right) = \log(\text{odds})$$

$$0 = \log(\text{odds})$$



Predicted Drug Effectiveness

0.5

+

$$\text{Output} = \log(\text{odds}) = 0$$

learning rate

0.3 X

Dosage < 15

Dosage < 5

-0.5

Output = -2

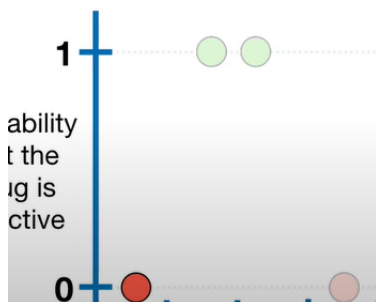
-0.5

Output = -2

0.5, 0.5

Output = 2

...and that gives us a **log(odds)** value = -0.6.



$$\log(\text{odds}) \text{ Prediction} = 0 + (0.3 \times -2) = -0.6$$

将log(odds)转回成probability

To convert a **log(odds)** value into a probability, we plug it into the **Logistic Function**.

$$\text{Probability} = \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}$$

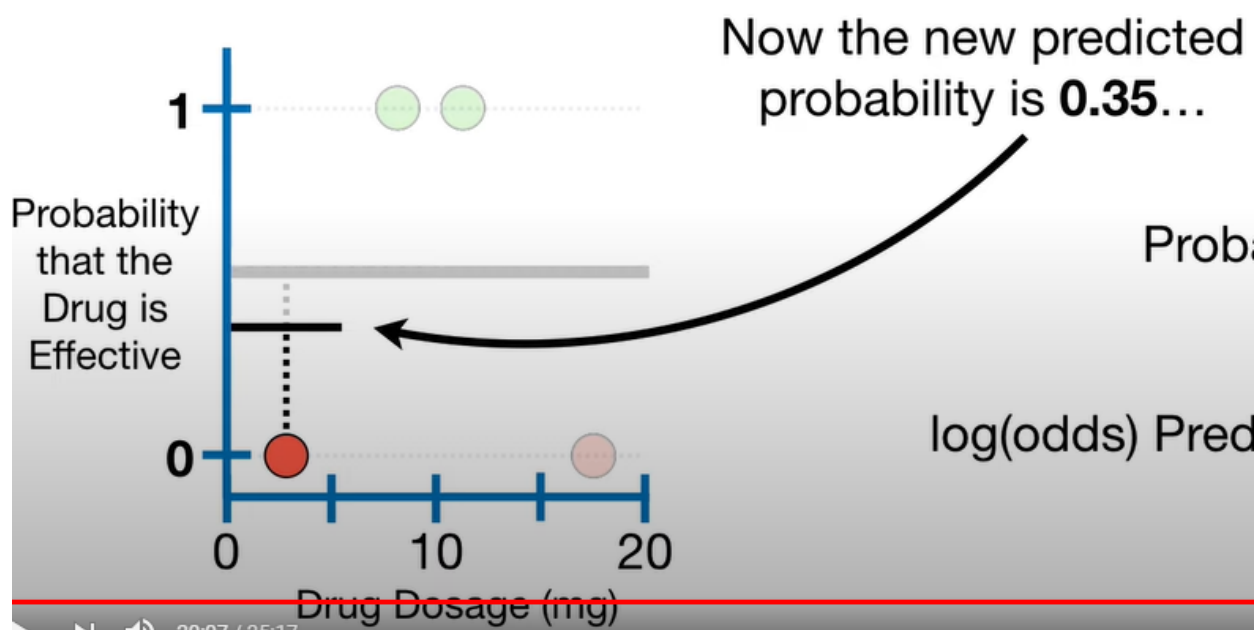
$$\log(\text{odds}) \text{ Prediction} = 0 + (0.3 \times -2) = -0.6$$

...and the the new predicted probability is **0.35**.

$$\text{Probability} = \frac{e^{-0.6}}{1 + e^{-0.6}} = 0.35$$

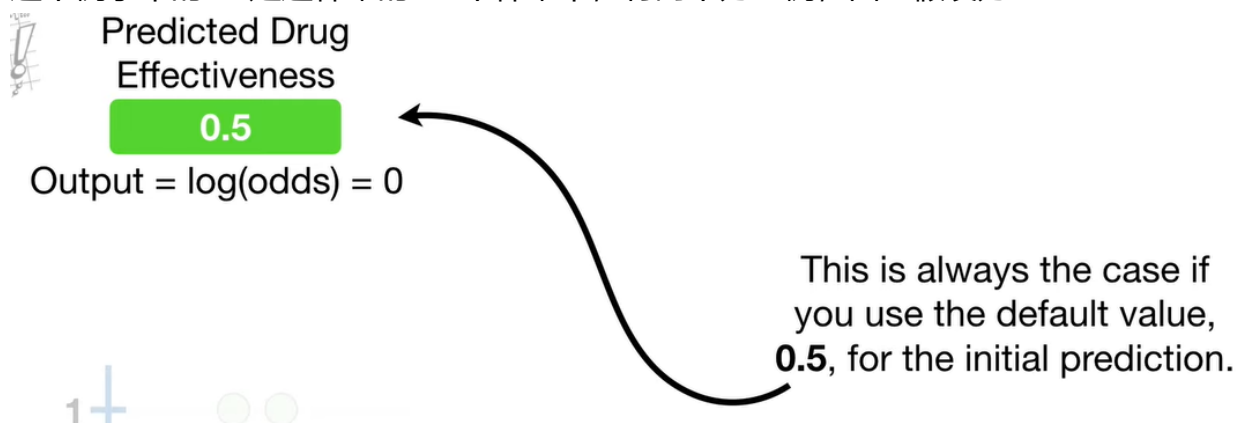
$$\log(\text{odds}) \text{ Prediction} = 0 + (0.3 \times -2) = -0.6$$

可以看到，和原来的0.5对比，样本和0.35之间的residual比和0.5之间的要小。证明已经得到了优化。



当最初预测值假设为0.5时，他的output一定为0.  $\log[p/(1-p)]$ 。

这个例子中的0.5是这样来的：4个样本中，有两个为正例，因此假设为0.5.



For example, if **75%** of the observations in the **Training Data** said that the drug was effective, we might set the initial prediction to **0.75** and now the initial  **$\log(\text{odds}) = 1.1...$**

...so we would plug **1.1** into the equation instead of **0**.



$$\log(\text{odds}) \text{ Prediction} = 0 + (0.3 \times -2) = -0.6$$

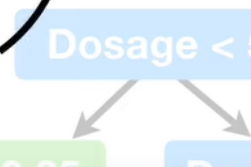
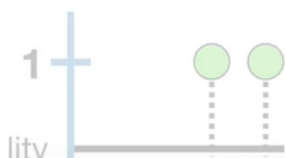
构造下一个树的注意事项

previous probability已经不是0.5了，是经过第一棵树计算后得到的结果。

因此计算similarity和output value时也会根据经过第一棵树计算后，样本更新的结果进行计算。

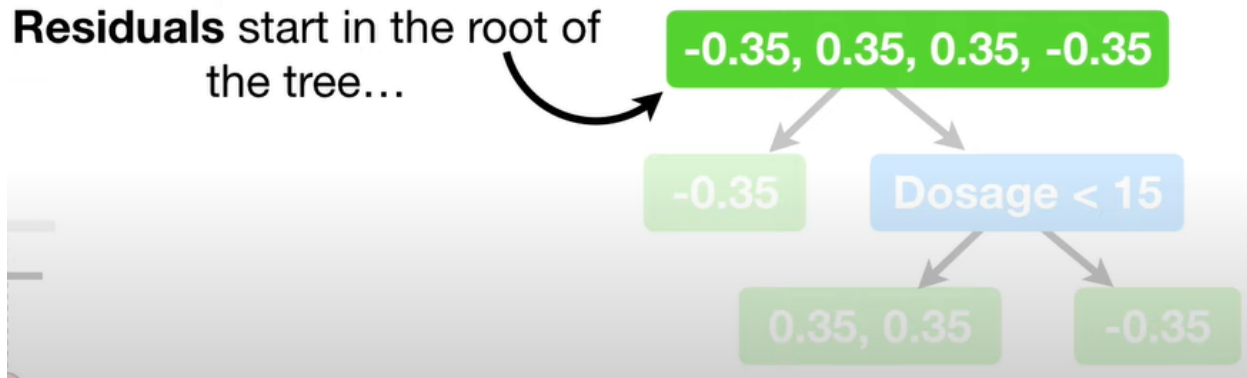
$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\sum [\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)] + \lambda}$$

**NOTE:** When we build the second tree, calculating the **Similarity Scores** is a little more interesting because the **Previous Probabilities** are no longer the same for all of the observations.



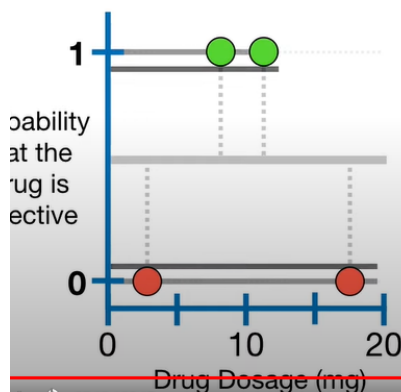
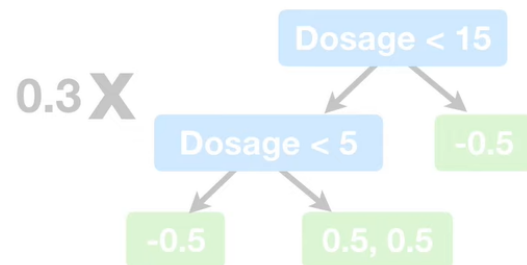
而且第二棵树的Root中的residual也会不同（是先把residuals放到节点中，再决定该节点的分裂条件是哪个）

For example, since all of the **Residuals** start in the root of the tree...



构建达到预设数量的树（或误差小于一个值时）停止构件树。

将样本在所有树的predicted value相加（包括initial prediction），得到最后的结果。



...and we keep building trees until the **Residuals** are super small, or we have reached the maximum number of trees.

