# XGBoost (classification)

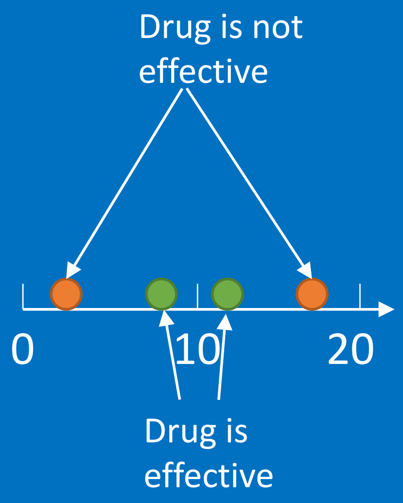
| Drug dosage | Drug usefulness |
|---|---|
| 5 | No |
| 8 | Yes |
| 11 | Yes |
| 18 | No |

Assuming that we have this very simple dataset

## Step 1: make the initial prediction

| Drug dosage | Drug usefulness |
|---|---|
| 5 | No |
| 8 | Yes |
| 11 | Yes |
| 18 | No |

Plot it out, we have

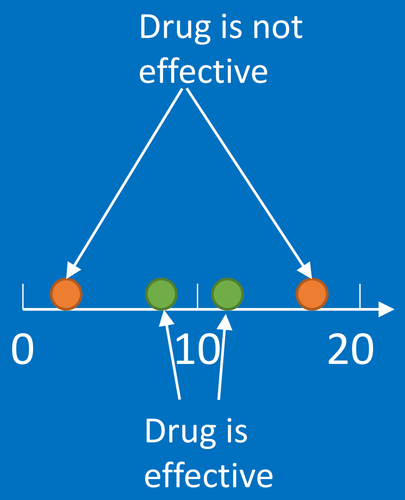


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

| **Drug dosage** | **Drug usefulness** |
|---|---|
| 5 | No |
| 8 | Yes |
| 11 | Yes |
| 18 | No |

Plot it out, we have



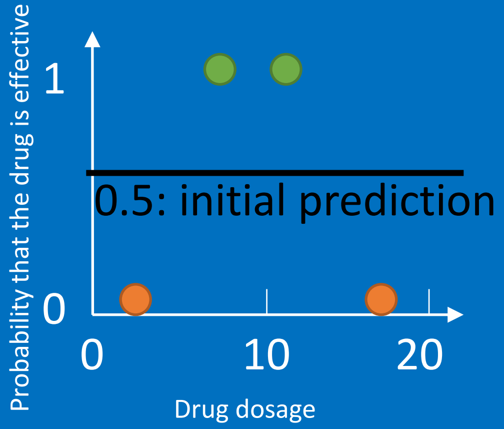
Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

In other words, does not matter how many dosages we take, the initial prediction for the usefulness for the drug is 50%

| Drug dosage | Drug usefulness |
|---|---|
| 5 | No |
| 8 | Yes |
| 11 | Yes |
| 18 | No |

Plot it out, we have



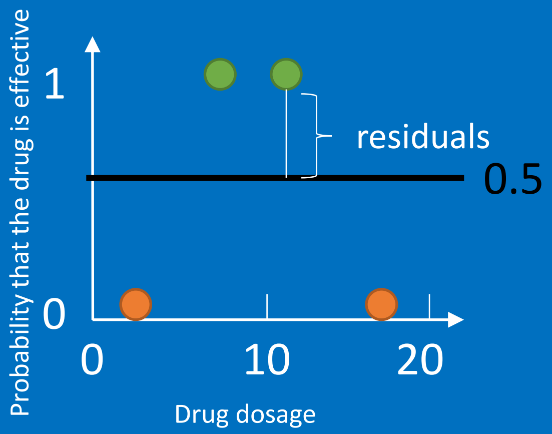
Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

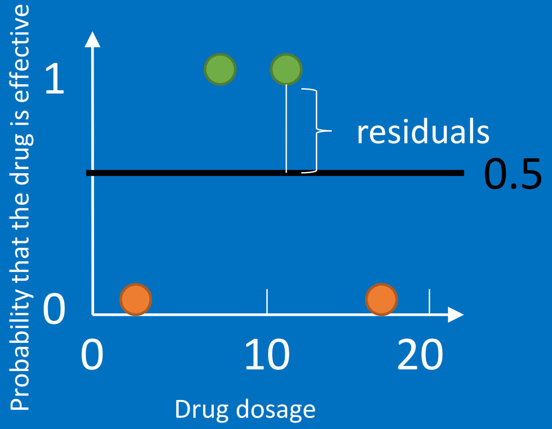
| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

3.1 All the tree starts from a leaf, with all the residuals

-0.5, 0.5, 0.5, -0.5

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

3.1 All the tree starts from a leaf, with all the residuals

-0.5, 0.5, 0.5, -0.5

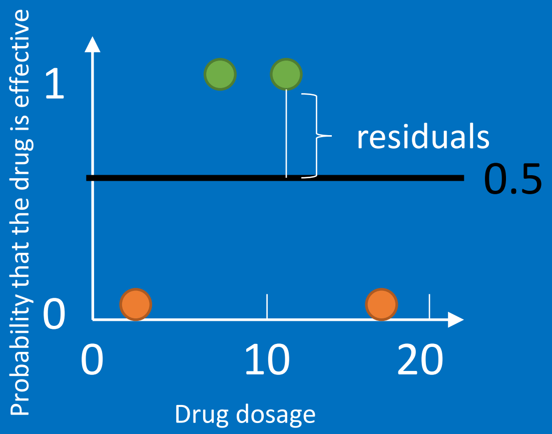
Note that for regression:

$$Similarity\ score = \frac{(\sum residuals)^2}{number\ of\ residuals + \lambda}$$

3.2 Calculate the Similarity score using

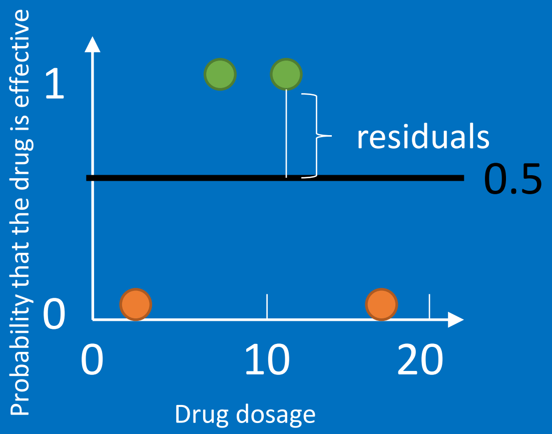
$$Similarity\ score = \frac{(\sum residuals)^2}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda}$$

Plot it out, we have



| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

3.1 All the tree starts from a leaf, with all the residuals

-0.5, 0.5, 0.5, -0.5

Note that for regression:

$$Similarity\ score = \frac{(\sum residuals)^2}{number\ of\ residuals + \lambda}$$

3.2 Calculate the Similarity score using

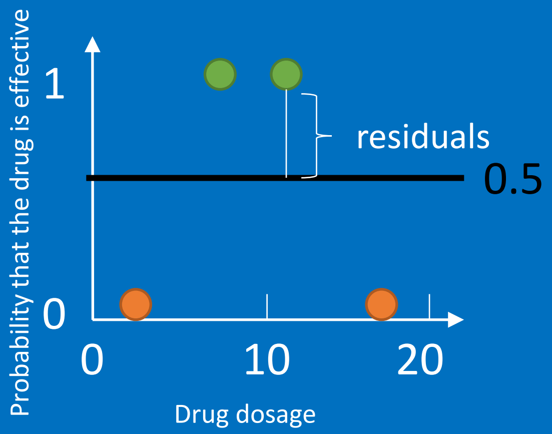
$$Similarity\ score = \frac{(\sum residuals)^2}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda}$$

For this case we have

$$Similarity\ score = \frac{(-0.5 + 0.5 + 0.5\ - 0.5)^2}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda} = 0$$

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

3.1 All the tree starts from a leaf, with all the residuals

-0.5, 0.5, 0.5, -0.5 0 *Similarity score*

3.2 Calculate the Similarity score

3.3 Further split the tree

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

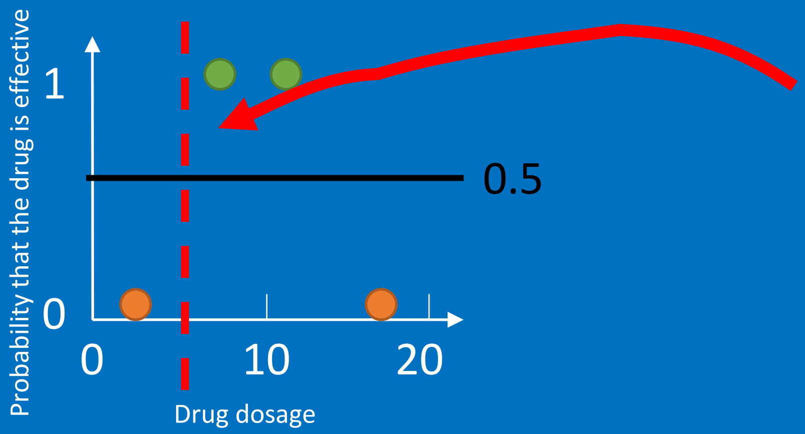
3.1 All the tree starts from a leaf, with all the residuals

-0.5, 0.5, 0.5, -0.5 0 *Similarity score*

3.2 Calculate the Similarity score

3.3 Further split the tree

Plot it out, we have



Similar to regression xgboost tree, we can start from the first data point, calculate the similarity score, and the gain. Then move to the next two points

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

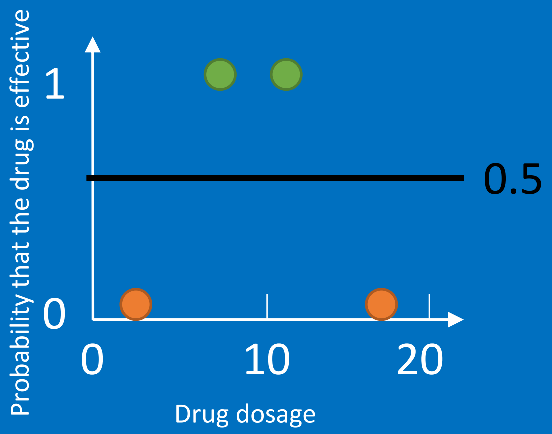
3.1 All the tree starts from a leaf, with all the residuals

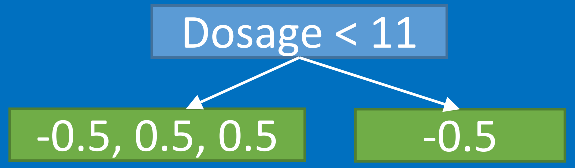
-0.5, 0.5, 0.5, -0.5 0 *Similarity score*

3.2 Calculate the Similarity score

3.3 Further split the tree

Plot it out, we have

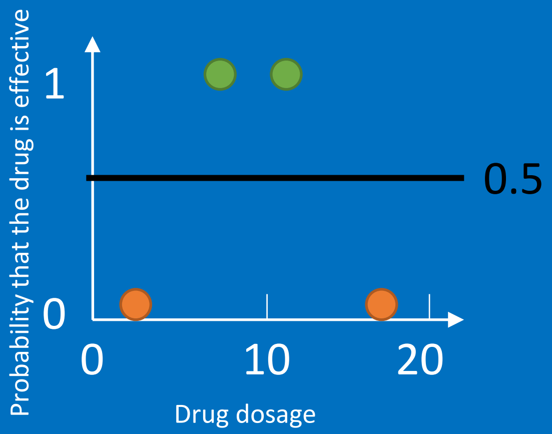




After gone through all the thresholds, we can tell that this tree gives the highest “Gain”

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

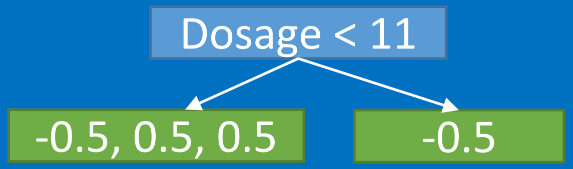


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)


Dosage < 11

-0.5, 0.5, 0.5

-0.5

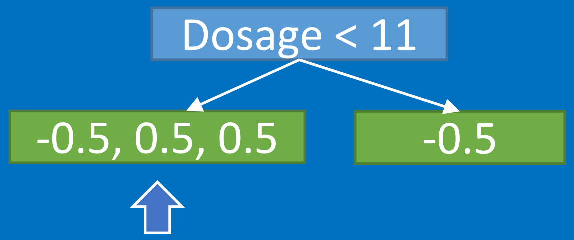
| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

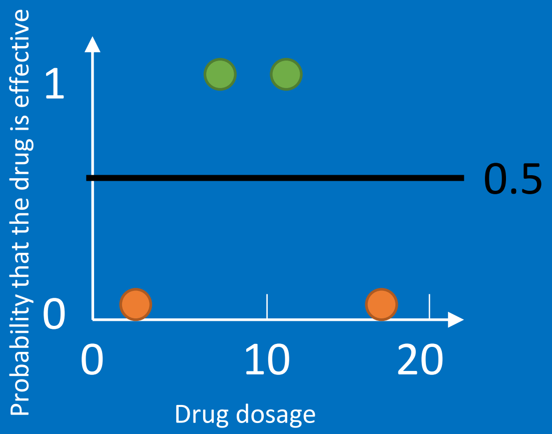
Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



We can continue split these residuals (following the same method)…

Plot it out, we have

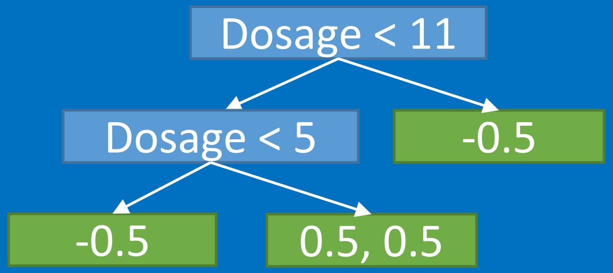


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

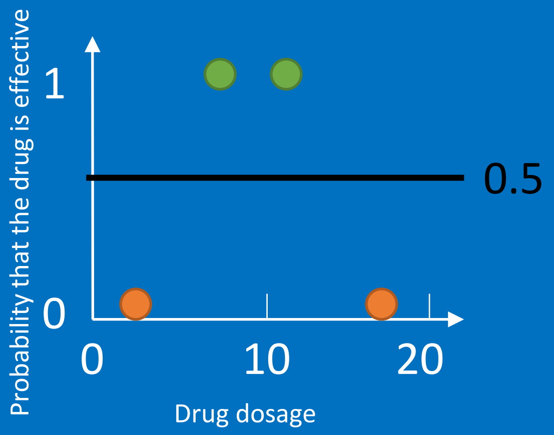
Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



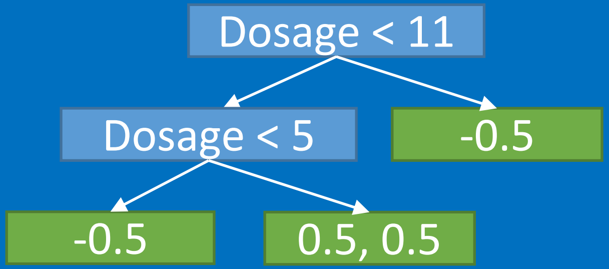
Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Dosage < 11

Dosage < 5

-0.5

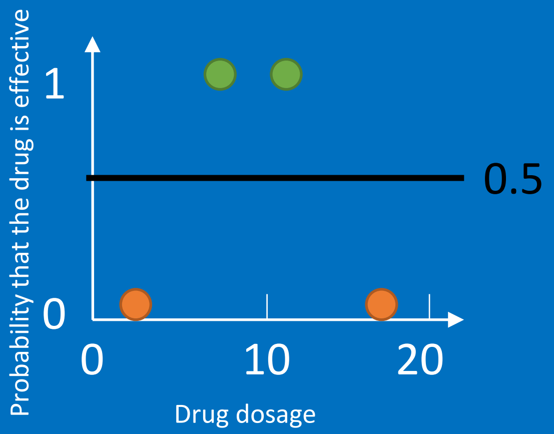
-0.5

0.5, 0.5

Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

Plot it out, we have



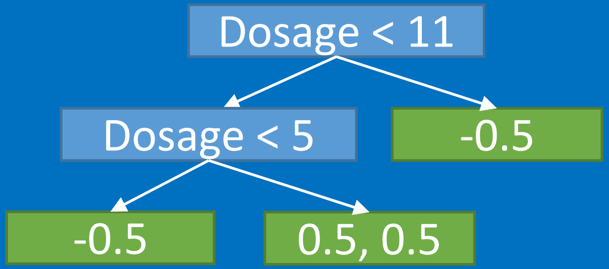
Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



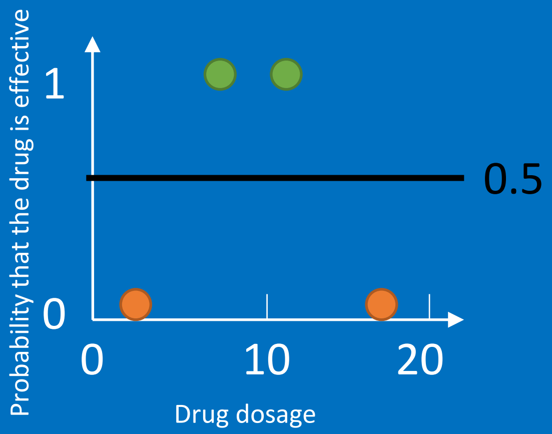
Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum [previous\ prob_i \times (1 - previous\ prob_i)]$

Plot it out, we have



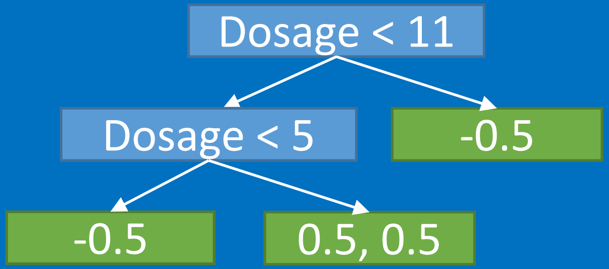
Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)


Dosage < 11

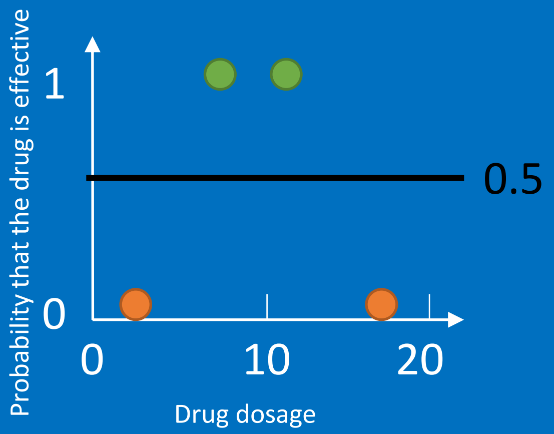
Dosage < 5

-0.5

-0.5

0.5, 0.5

Plot it out, we have



Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum [previous\ prob_i \times (1 - previous\ prob_i)]$

So for regression, if we don't set cover by ourself, we always can just have one residual in one leaf

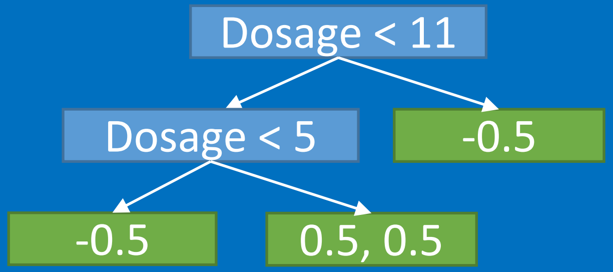
| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

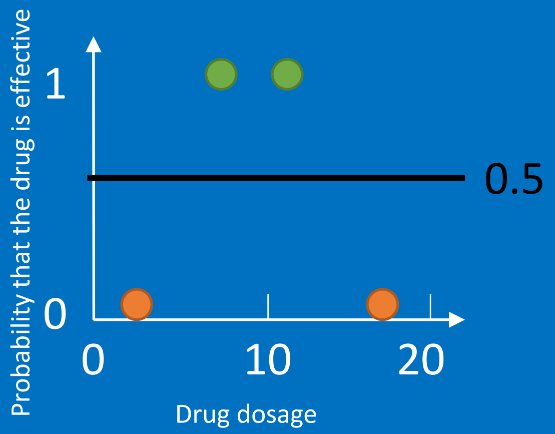
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Plot it out, we have



Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum [previous\ prob_i \times (1 - previous\ prob_i)]$

So for regression, if we don’t set cover by ourself, we always can just have one residual in one leaf

For classification, it’s more complicated

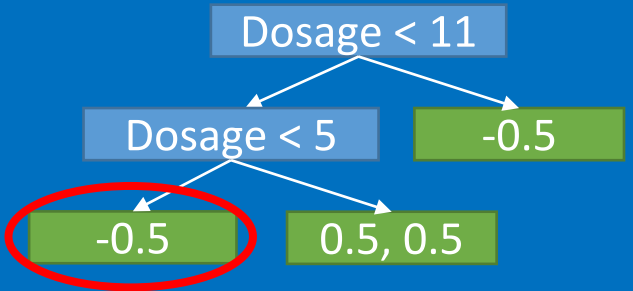
| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

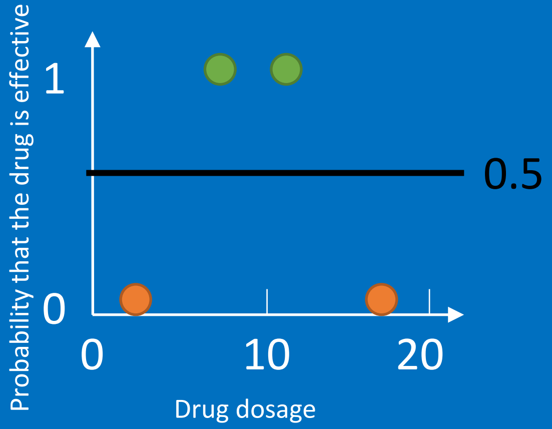
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Plot it out, we have



Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum [previous\ prob_i \times (1 - previous\ prob_i)]$

So for regression, if we don’t set cover by ourself, we always can just have one residual in one leaf

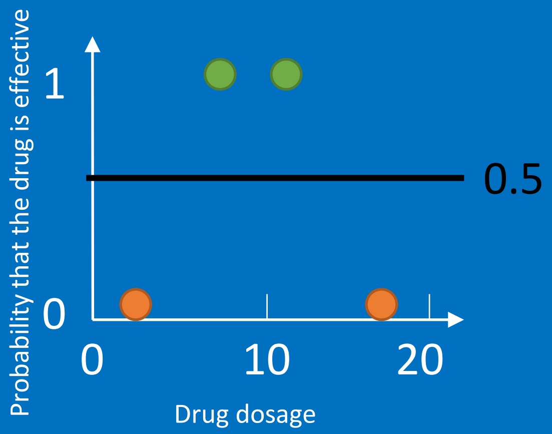
For classification, it’s more complicated

For example, the cover for this leaf is

$$cover = 0.5 \times (1 - 0.5) = 0.25$$

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

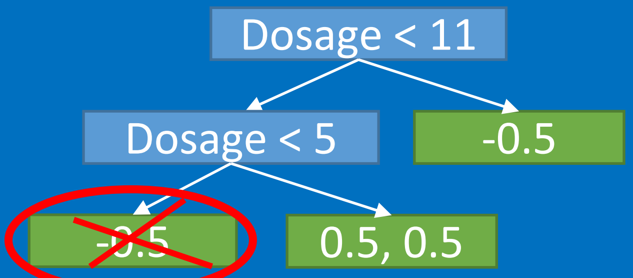


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum[previous\ prob_i \times (1 - previous\ prob_i)]$

So for regression, if we don't set cover by ourself, we always can just have one residual in one leaf

For classification, it's more complicated

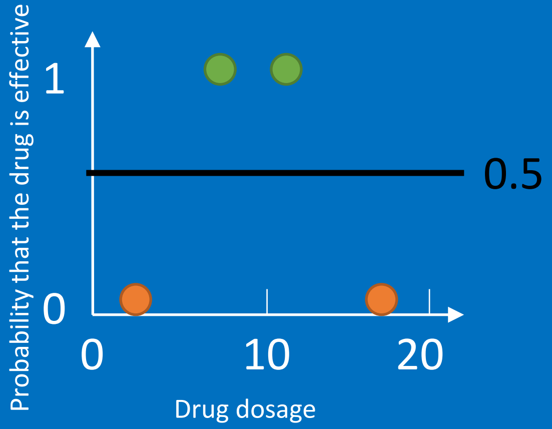
For example, the cover for this leaf is

$$cover = 0.5 \times (1 - 0.5) = 0.25$$

So this leaf will be removed

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

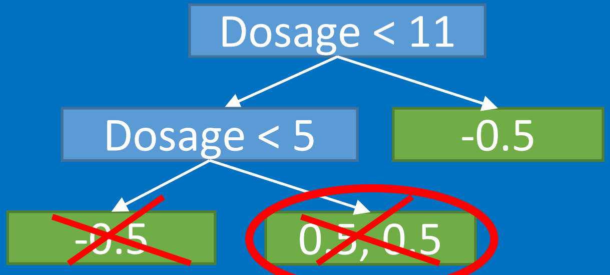


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Dosage < 11

Dosage < 5

-0.5

~~-0.5~~

~~0.5, 0.5~~

Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum[previous\ prob_i \times (1 - previous\ prob_i)]$

So for regression, if we don't set cover by ourself, we always can just have one residual in one leaf

For classification, it's more complicated

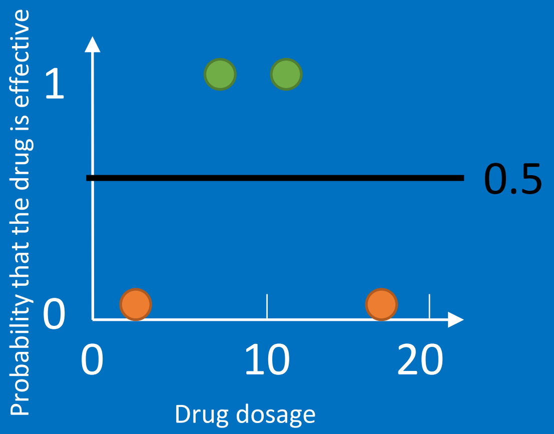
For example, the cover for this leaf is

$$cover = 0.5 \times (1 - 0.5) + 0.5 \times (1 - 0.5) = 0.5$$

So this leaf will be removed, too

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

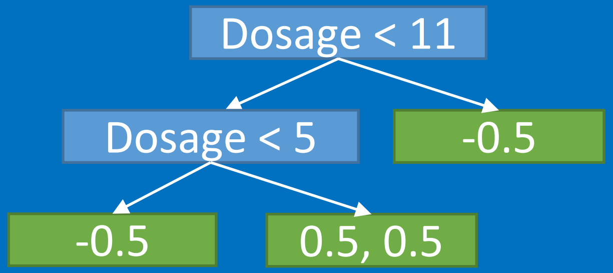


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Note that for a xgboost tree, there are two limitations we can implement:

- How many levels of the tree can grow (like any other tree methods)
- How many residuals in each leaf: this is defined as cover

cover, by default, is defined as

- For regression $cover = number\ of\ residuals$
- For classification $cover = \sum [previous\ prob_i \times (1 - previous\ prob_i)]$

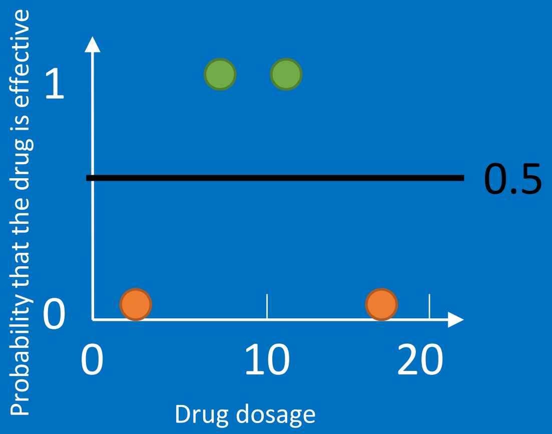
So for regression, if we don't set cover by ourself, we always can just have one residual in one leaf

So as you can see, by default, too many leaf will be removed in this example, if we use the default cover. In order to make this example working, let's just set cover=0

In some packages, cover is called min_child_weight

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

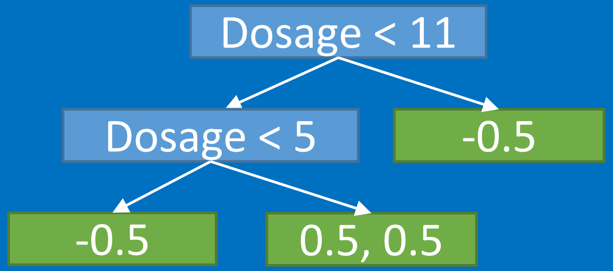


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

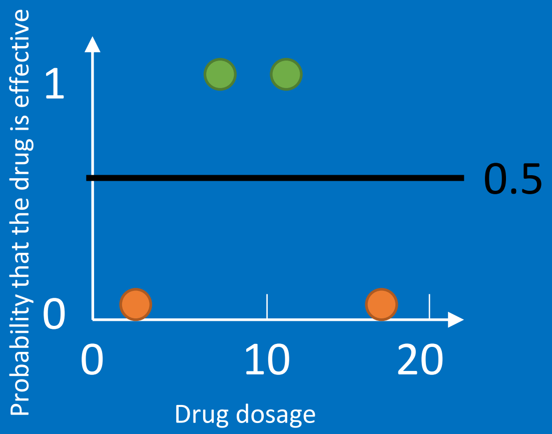
Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Step 4: Prune the tree (like the xgboost for regression)

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

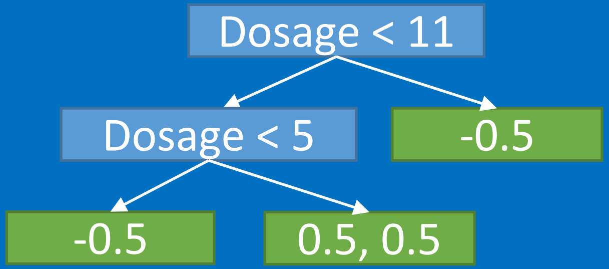


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

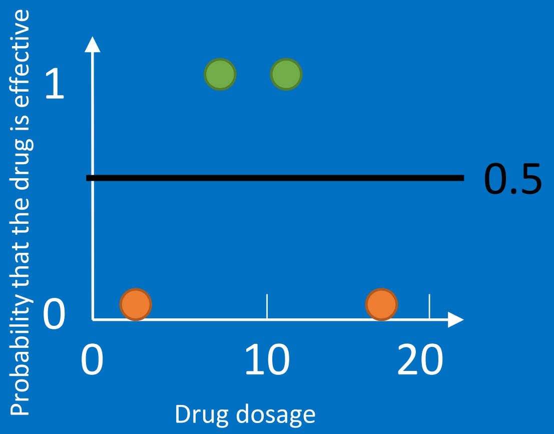


Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

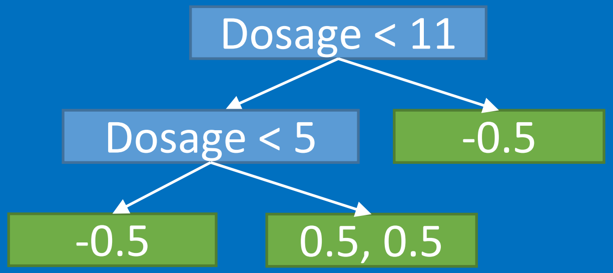


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

Note that for regression xgboost:

$$output = \frac{sum\ of\ residuals}{number\ of\ residuals + \lambda}$$

$$output = \frac{\sum residuals}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda}$$

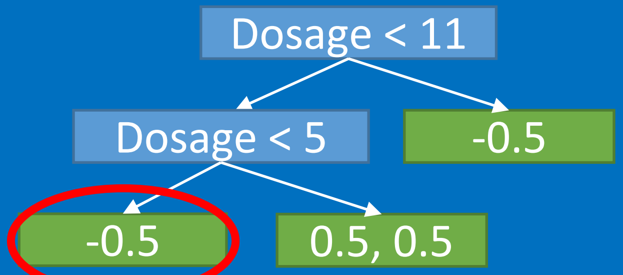
| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

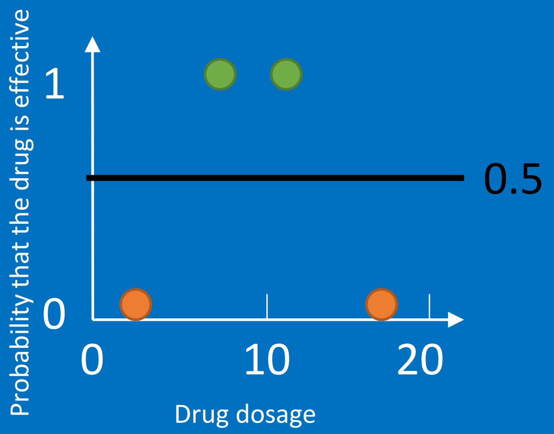
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



Plot it out, we have



Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

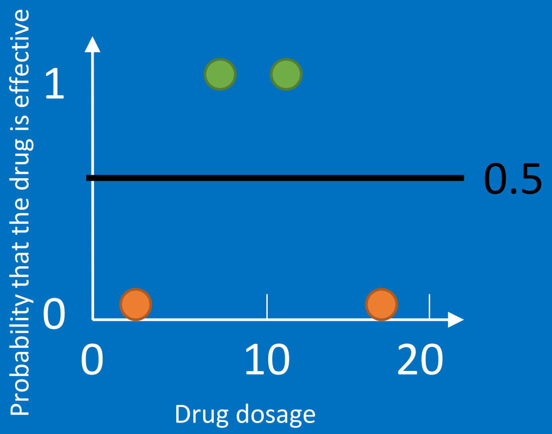
$$output = \frac{\sum residuals}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda}$$

So for this leaf, we have (assuming lambda=0):

$$output = \frac{-0.5}{0.5 \times (1 - 0.5) + 0} = -2$$

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have

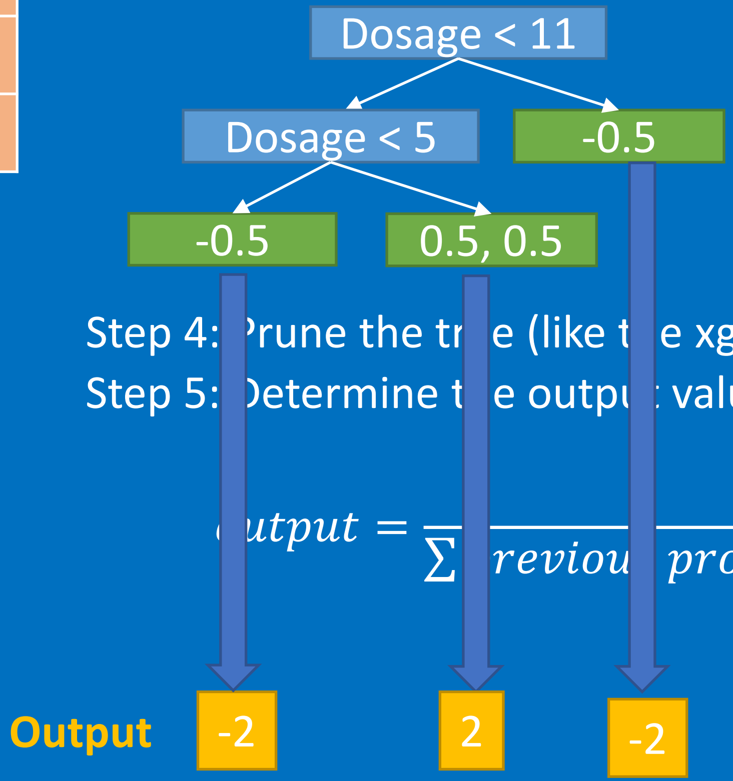


Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)



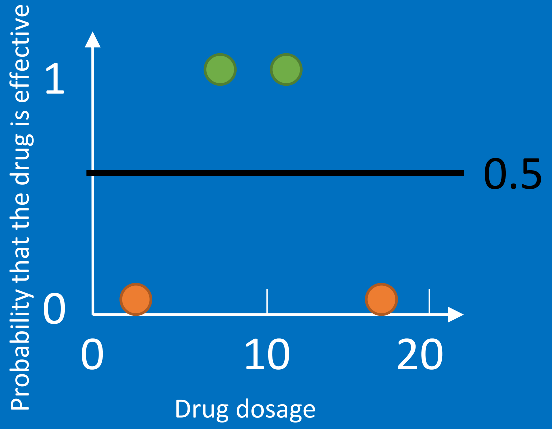
Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

$$output = \frac{\sum residuals}{\sum previous\ prob_i \times (1 - previous\ prob_i) + \lambda}$$

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

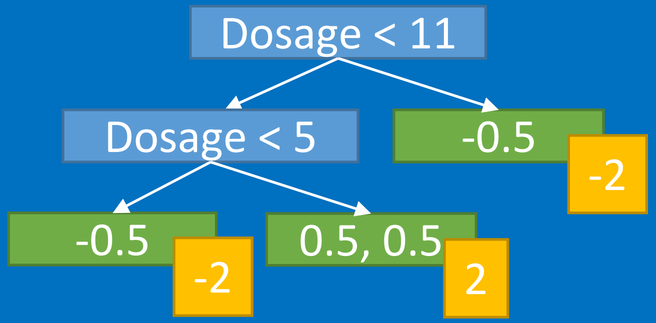
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

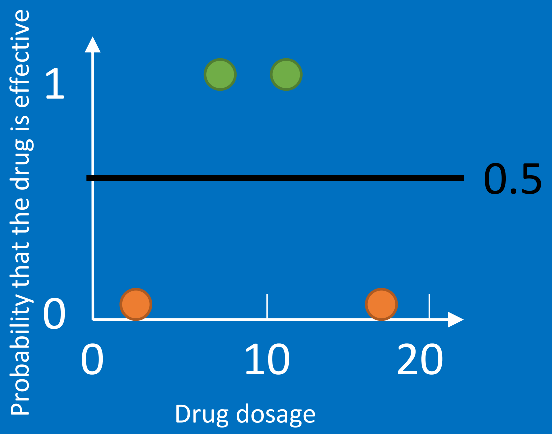
Step 5: Determine the output value



So the first tree is completed

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

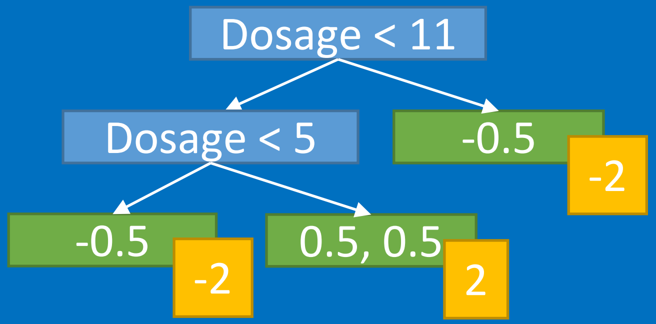
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

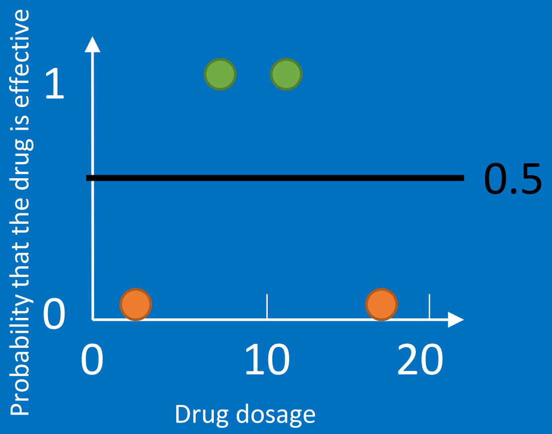


So the first tree is completed

Step 6: Start making predictions

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

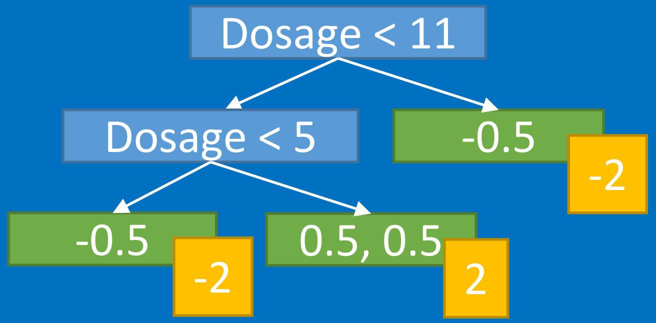
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

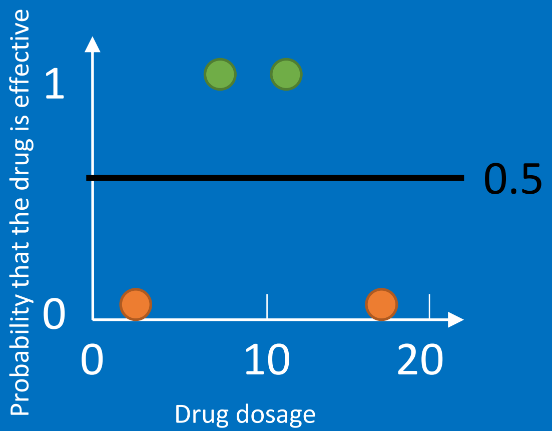
Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)

The equation is $\log(odds) = \log(\frac{p}{1-p})$

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

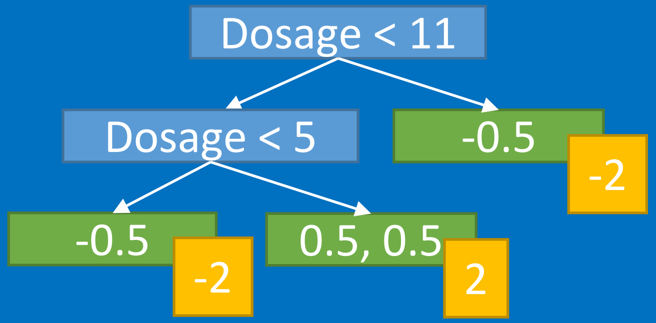
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)

The equation is $\log(odds) = \log(\frac{p}{1-p})$

The previous probability is 0.5, so log(odds) = 0

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

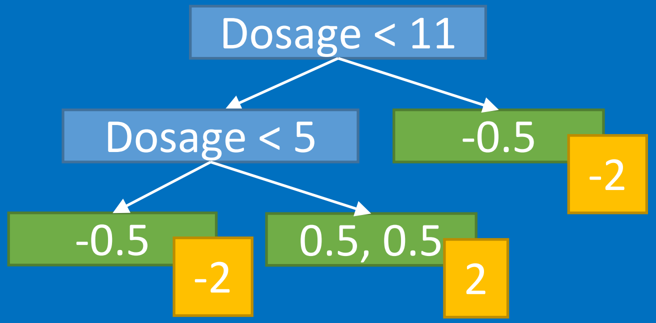
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

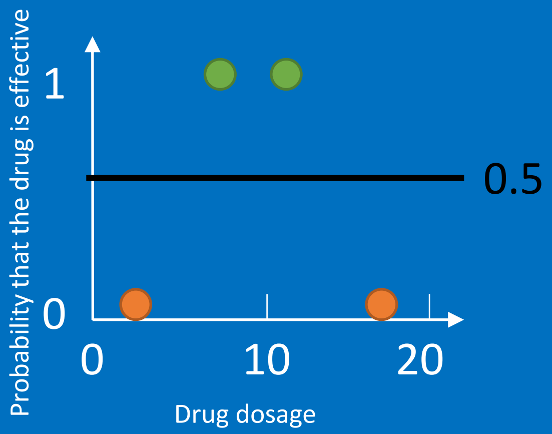
Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

Plot it out, we have

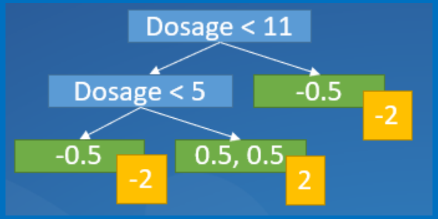


Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with

0.5 log(odds)=0 + 0.3 X



| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

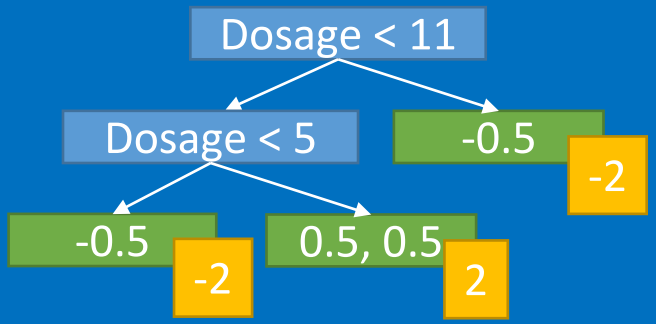
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

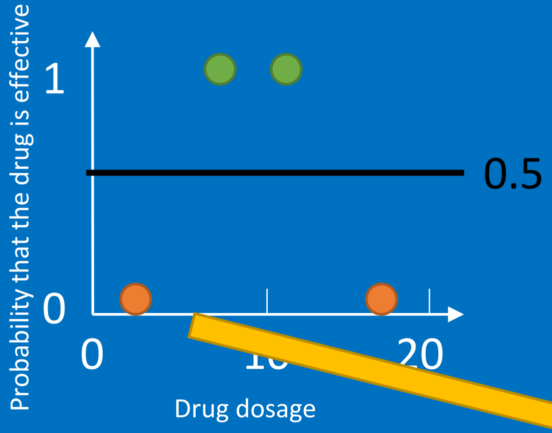
Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

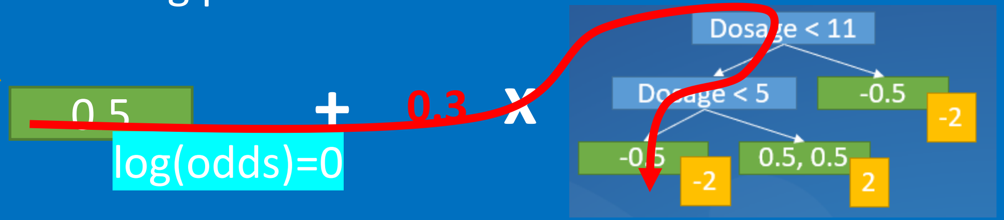
Plot it out, we have



Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with



For example, for the first sample, we can have the log(odds) prediction as 0+0.3x(-2)=-0.6

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

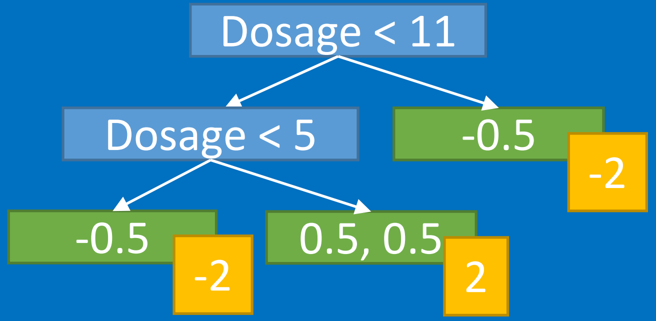
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

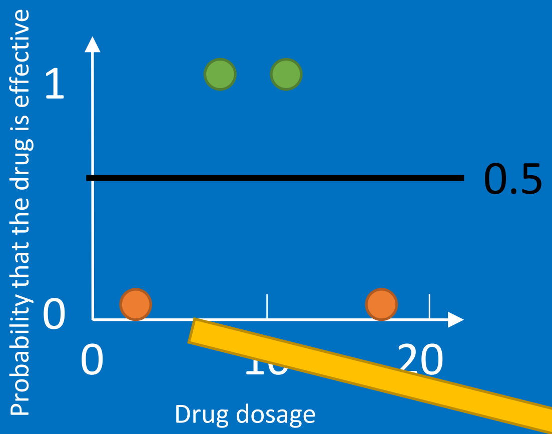
Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

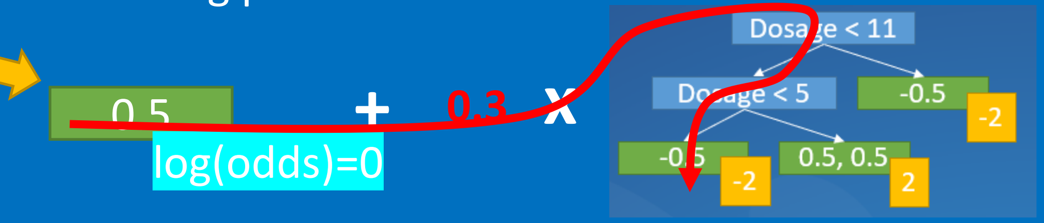
Plot it out, we have



Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with



Then we need to convert log(odds) back to probability as:

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

For example, for the first sample, we can have the log(odds) prediction as 0+0.3x(-2)=-0.6

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

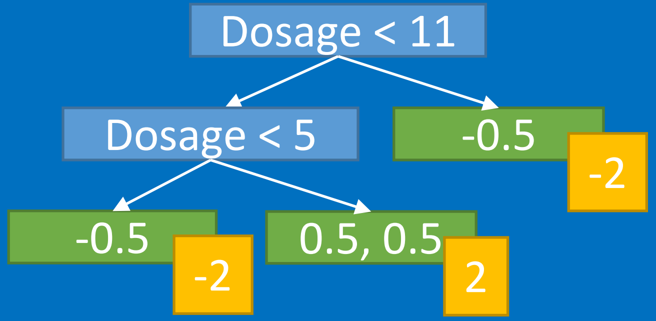
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value

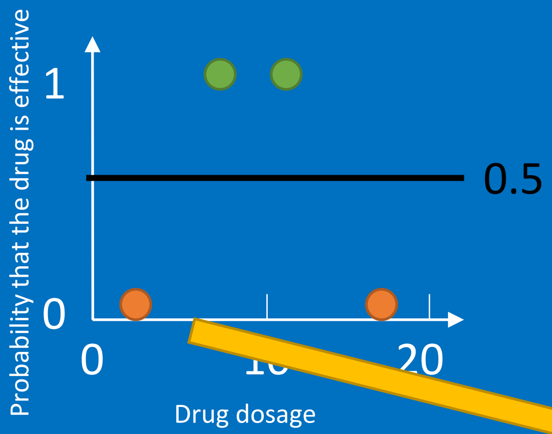


So the first tree is completed

So for the first sample, the prediction of probability is

$$p = \frac{e^{\log(odds)}}{1+e^{\log(odds)}} = \frac{e^{-0.6}}{1+e^{-0.6}} = 0.35$$

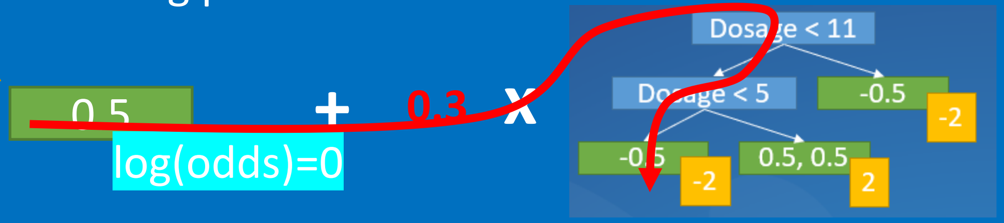
Plot it out, we have



Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with



Then we need to convert log(odds) back to probability as:

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

For example, for the first sample, we can have the log(odds) prediction as 0+0.3x(-2)=-0.6

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Step 1: make the initial prediction

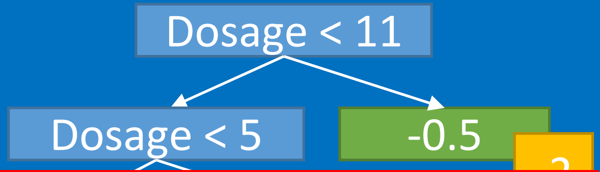
For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

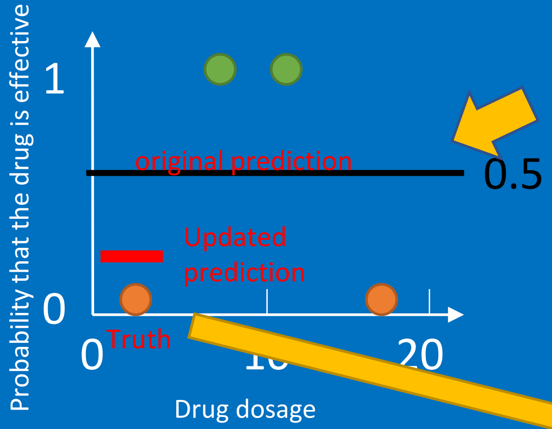
Step 4: Prune the tree (like the xgboost for regression)

Step 5: Determine the output value



So the first tree is completed

Plot it out, we have



The original prediction is 0.5, and the new prediction is 0.35 (get reduced from the original), and it is getting closer to the truth

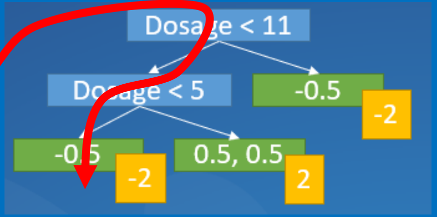
So for the first sample, the prediction of probability is

$$p = \frac{e^{\log(odds)}}{1 + e^{\log(odds)}} = \frac{e^{-0.6}}{1 + e^{-0.6}} = 0.35$$

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with

0.5 log(odds)=0 + 0.3 x



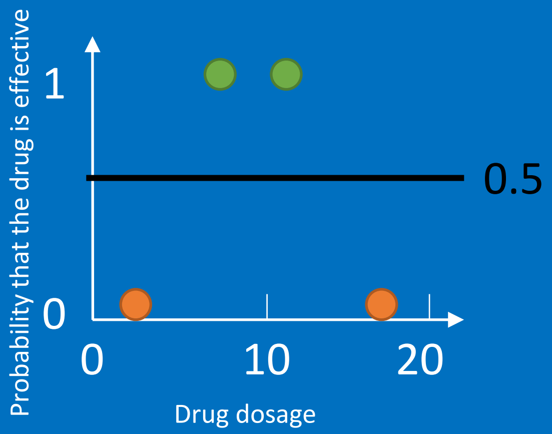
Then we need to convert log(odds) back to probability as:

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

For example, for the first sample, we can have the log(odds) prediction as 0+0.3x(-2)=-0.6

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

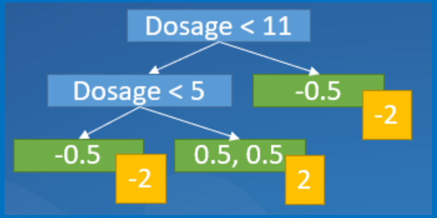
Step 5: Determine the output value

Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

6.2: making prediction with

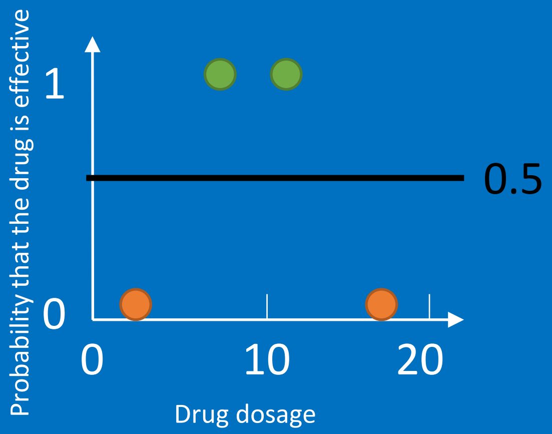
0.5 log(odds) + 0.3 X



We repeat the process from Step 1 to Step 6 but with updated prediction, every time we get reduced residuals, until the residuals do not change much anymore …

| Drug dosage | Drug usefulness | Residuals |
|---|---|---|
| 5 | No | -0.5 |
| 8 | Yes | 0.5 |
| 11 | Yes | 0.5 |
| 18 | No | -0.5 |

Plot it out, we have



Step 1: make the initial prediction

For example, the probability of the dosage is effective: 0.5

Step 2: we get the residuals for all the samples

Step 3: grow a xgboost tree to the residuals (Similar method used for regression)

Step 4: Prune the tree (like the xgboost for regression)

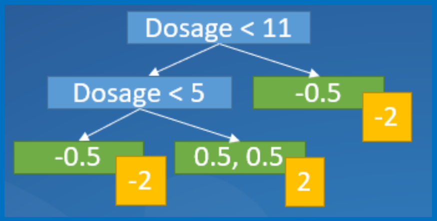
Step 5: Determine the output value

Step 6: Start making predictions

6.1: first we need to convert the previous prob to log(odds)=0

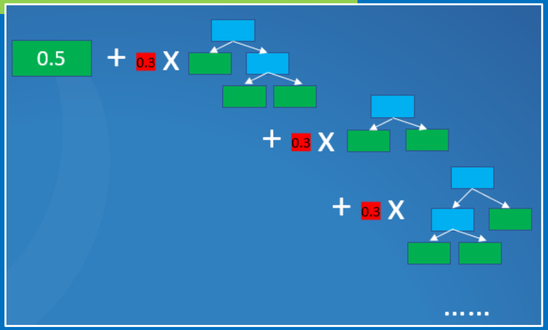
6.2: making prediction with

0.5 log(odds) + 0.3 X



We repeat the process from Step 1 to Step 6 but with updated prediction, every time we get reduced residuals, until the residuals do not change much anymore …

So the final xgboost tree for classification will be something like this:



0.5
+ 0.3 X
+ 0.3 X
+ 0.3 X
……