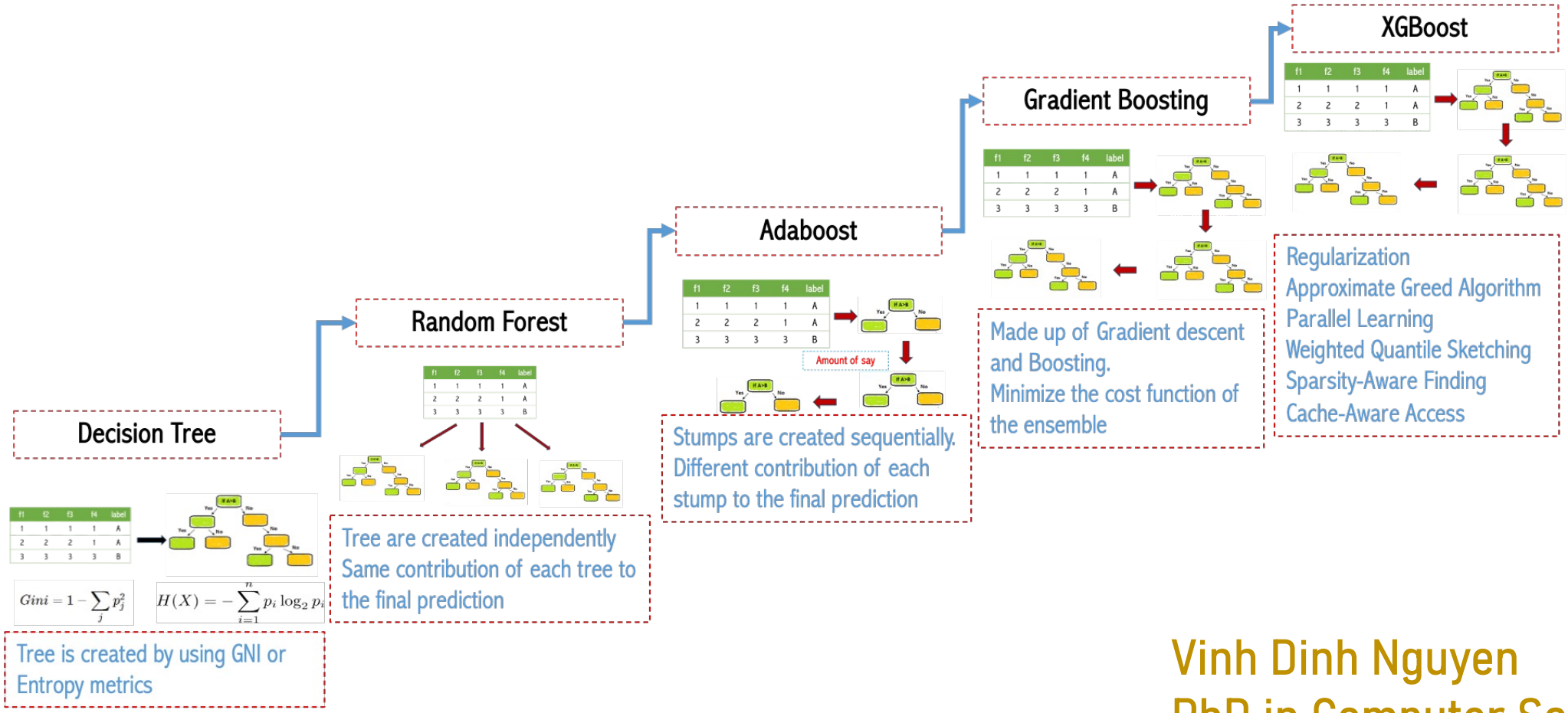
AI VIETNAM
All-in-One Course

Tree and Its Variants

(Review & Discussion)



Vinh Dinh Nguyen
PhD in Computer Science

Year 2023

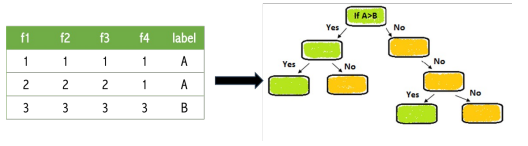
Outline

- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

AI VIETNAM
All-in-One Course

Evolution of Tree and Its Variant

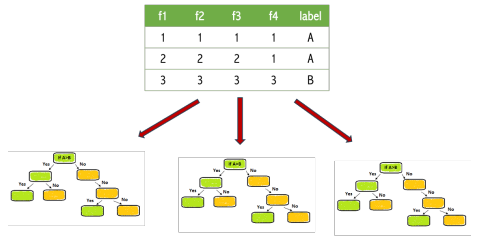
Decision Tree



$$Gini = 1 - \sum_j p_j^2 \qquad H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

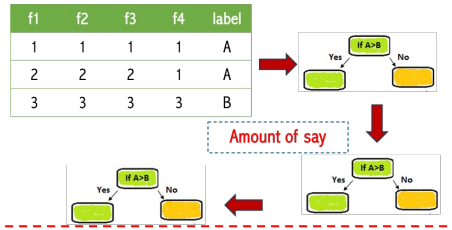
Tree is created by using GNI or Entropy metrics

Random Forest



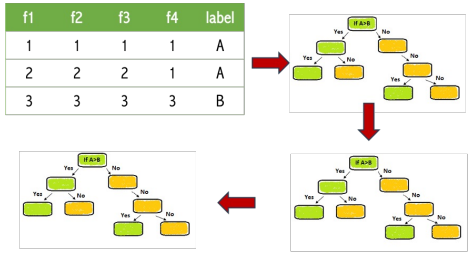
Tree are created independently
Same contribution of each tree to the final prediction

Adaboost



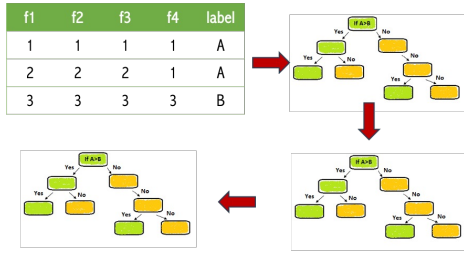
Stumps are created sequentially.
Different contribution of each stump to the final prediction

Gradient Boosting



Made up of Gradient descent and Boosting.
Minimize the cost function of the ensemble

XGBoost



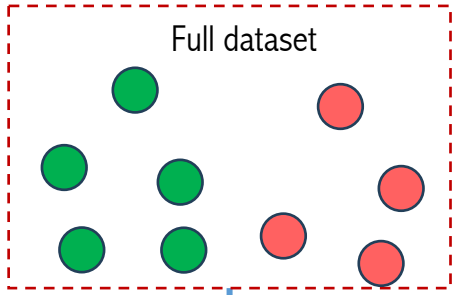
Regularization
Approximate Greed Algorithm
Parallel Learning
Weighted Quantile Sketching
Sparsity-Aware Finding
Cache-Aware Access

Outline

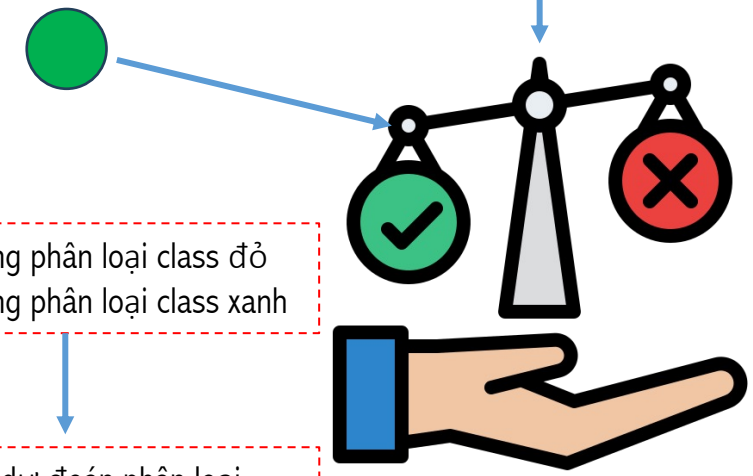
- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

Decision Tree Motivation

Case 1:



Dữ liệu Test

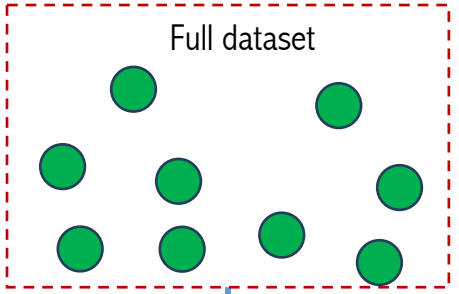


40% tin tưởng phân loại class đỏ
60% tin tưởng phân loại class xanh

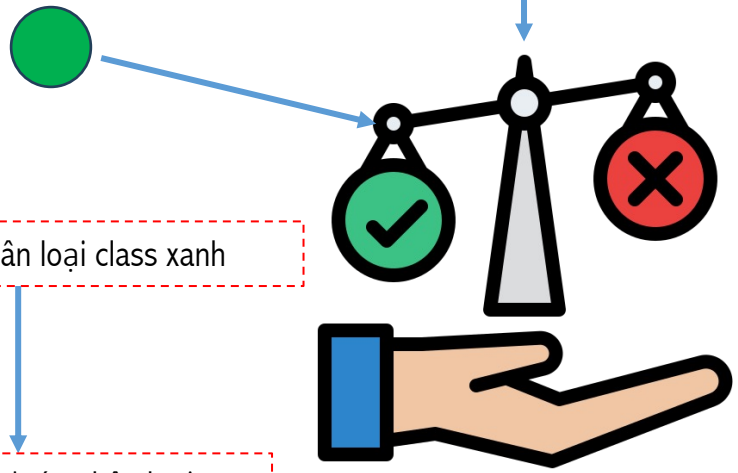
Kết quả dự đoán phân loại class xanh

Không tin tưởng vào kết quả dự đoán. Tại sao?

Case 2:



Dữ liệu Test

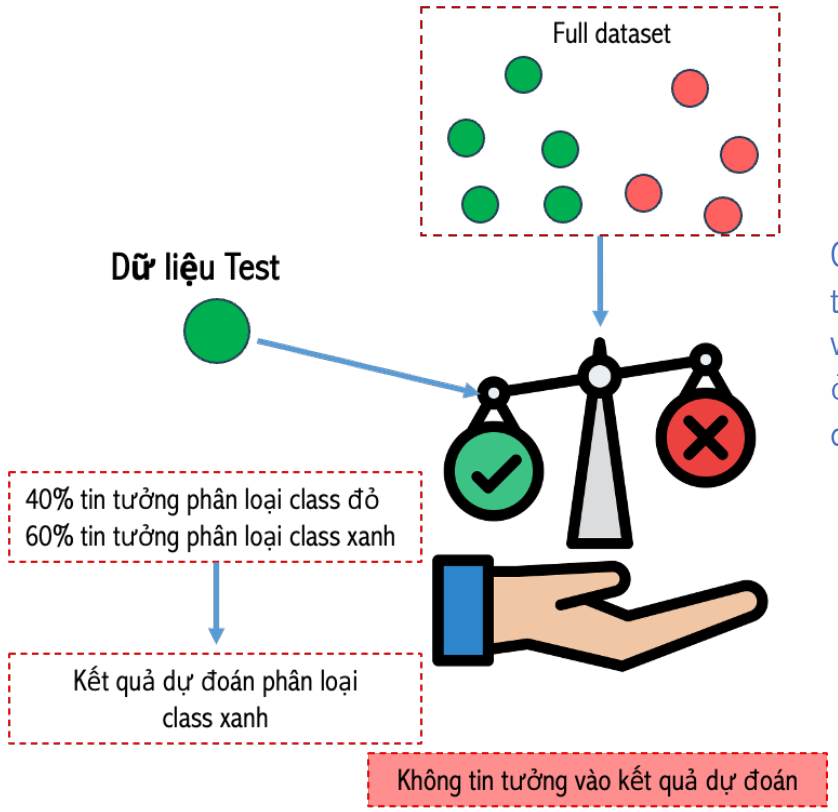


100% tin tưởng phân loại class xanh

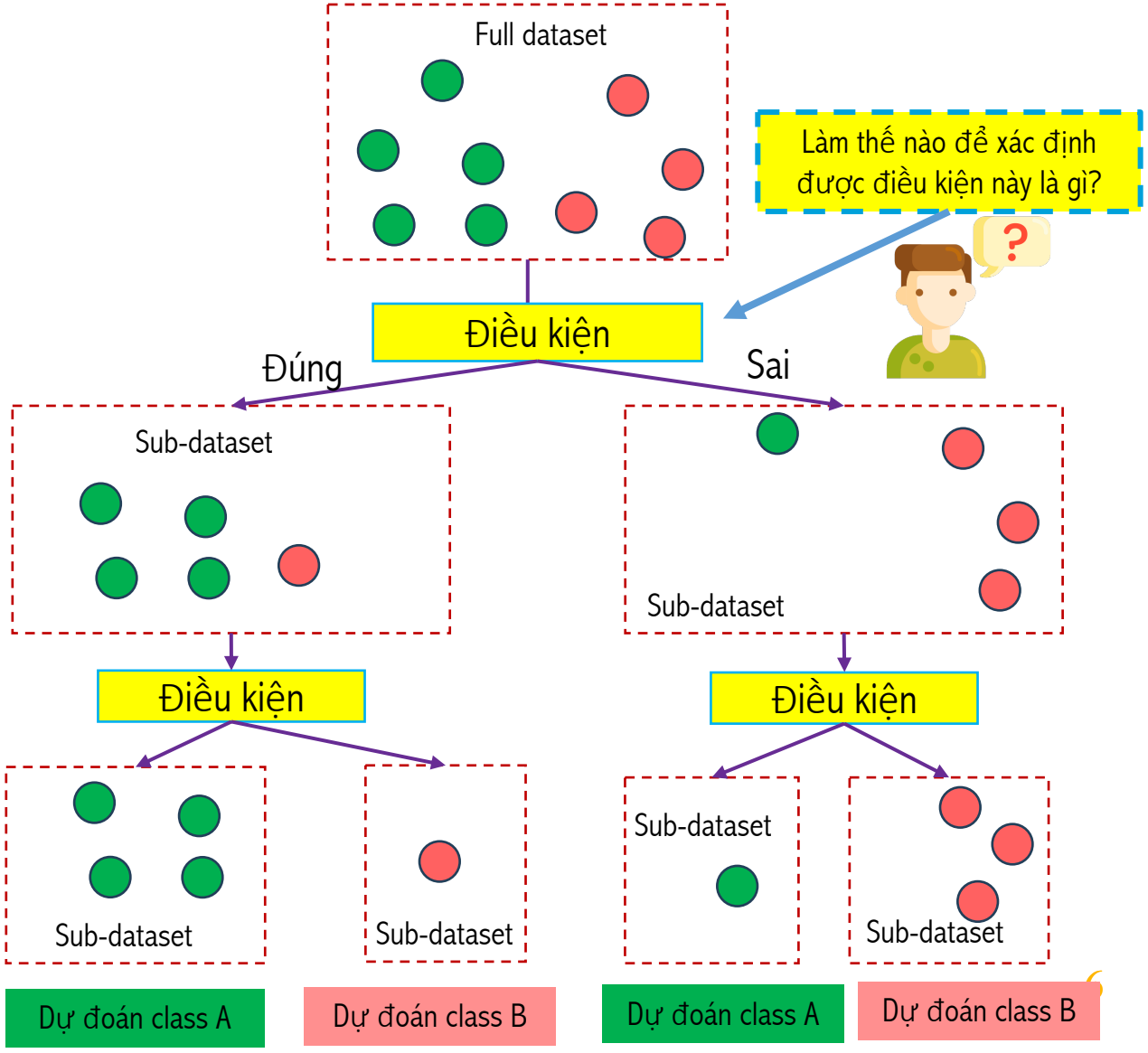
Kết quả dự đoán phân loại class xanh

Rất tin tưởng vào kết quả dự đoán. Tại sao?

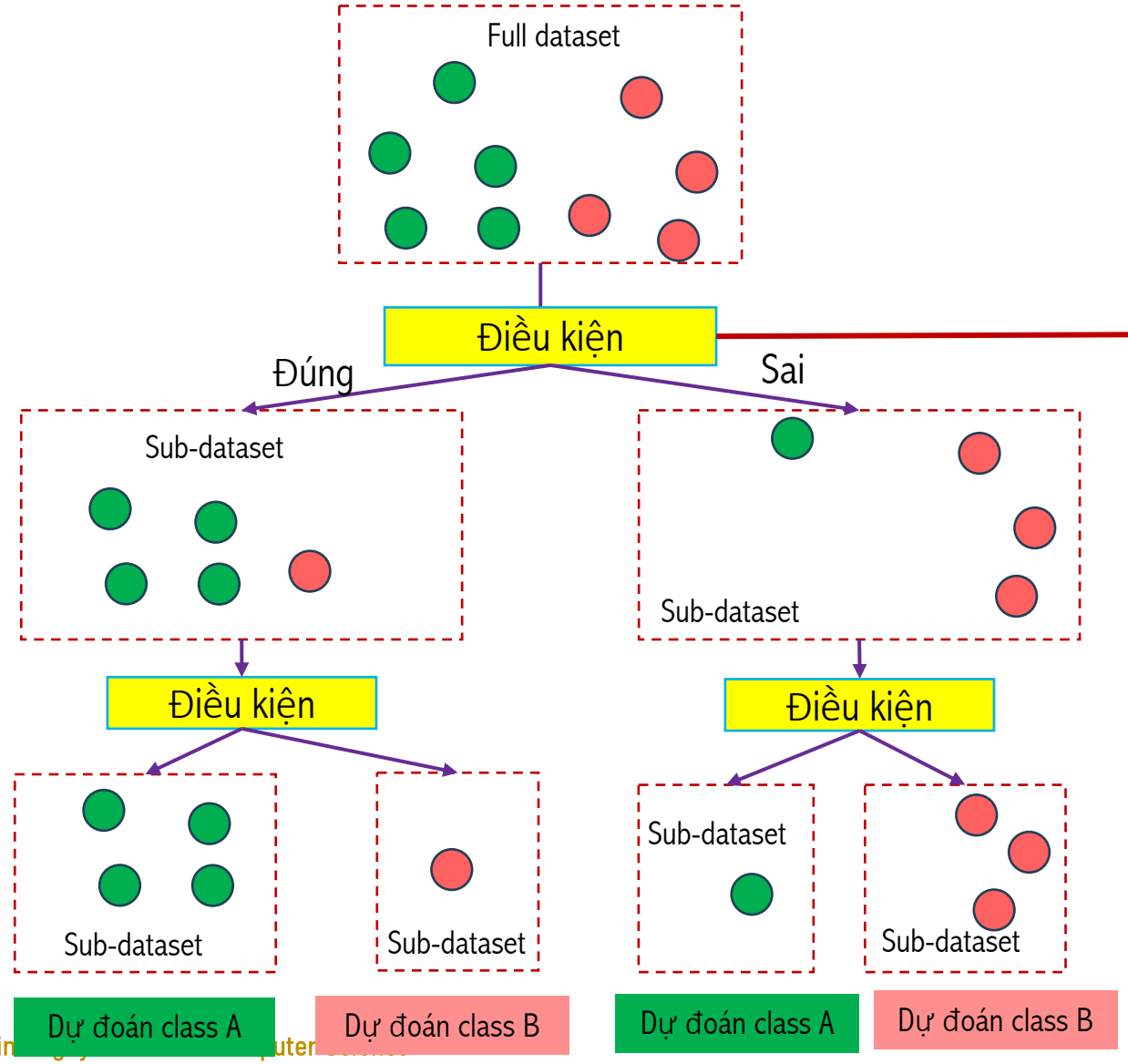
Decision Tree Motivation



Chia Full dataset đầu vào thành các tập con nhỏ hơn với điều kiện cho trước, mà ở đó chương trình tin tưởng dự đoán đúng



Decision Tree Motivation



Utility Tool for Building Conditions

Gini Impurity

Entropy

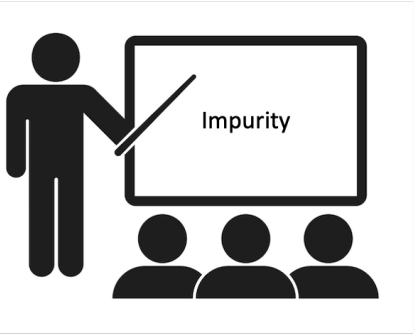
$$Gini = 1 - \sum_j p_j^2$$

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Question: Please show me! Minimum and Maximum Values of Gini and Entropy?

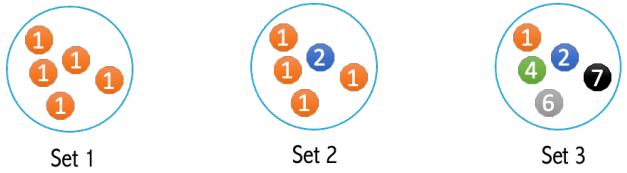
Gini and Entropy

Gini Impurity



Set 1	1	1	1	1	1
Set 2	1	1	2	1	1
Set 3	1	2	4	6	7

	Impurity Score
Set 1	Thấp
Set 2	Trung Bình
Set 3	Cao



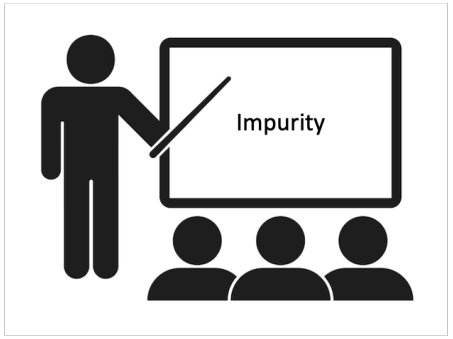
Qualitative measurement:

Quantitative measurement:

$$Gini = 1 - \sum_j p_j^2$$

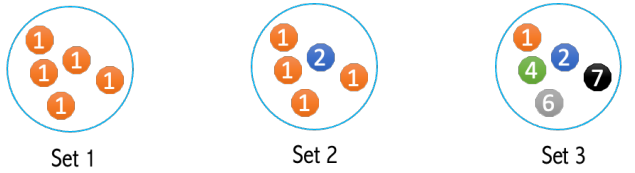
Gini càng nhỏ thì dataset …

Entropy



Set 1	1	1	1	1	1
Set 2	1	1	2	1	1
Set 3	1	2	4	6	7

	Impurity Score
Set 1	Thấp
Set 2	Trung Bình
Set 3	Cao



Qualitative measurement:

Quantitative measurement:

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Entropy càng lớn thì dataset …

Decision Tree for Classification

Why does Gini of "Love Math" equal to 0.214 ?

GINI = 0.214

Love Art

Thích | Không thích

Love AI | Love AI

Yes	No
3	1

$$Gini = 1 - \sum_j p_j^2$$

Yes	No
0	3

GINI = 0.375

GINI = 0

Why does entropy of "Love Math" equal to 0.463 ?

Entropy = 0.463

Love Math

Thích | Không thích

Love AI | Love AI

Yes	No
3	1

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Yes	No
0	3

Entropy = 0.81

Entropy = 0

GINI Example

No.	Love Math	Love Art	Age	Love AI
1	Yes	Yes	7	No
2	Yes	No	12	No
3	No	Yes	18	Yes
4	No	Yes	35	Yes
5	Yes	Yes	38	Yes
6	Yes	No	50	No
7	No	No	83	No

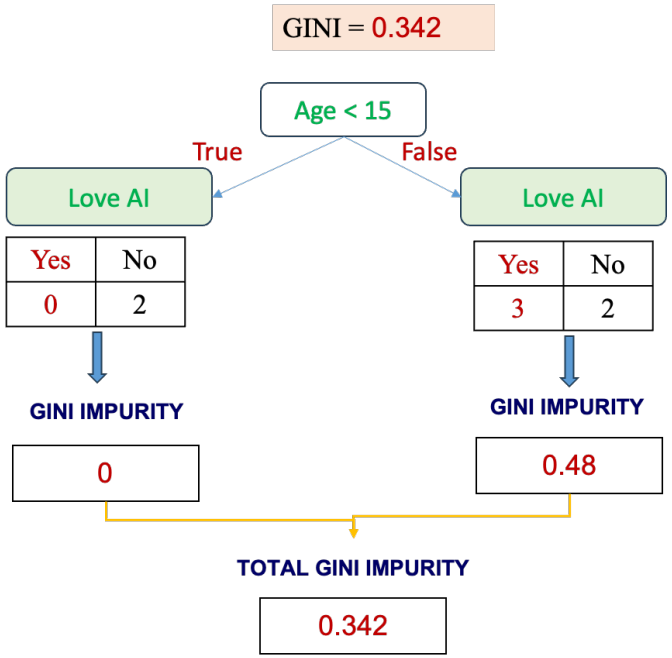
Love Math

Love Art

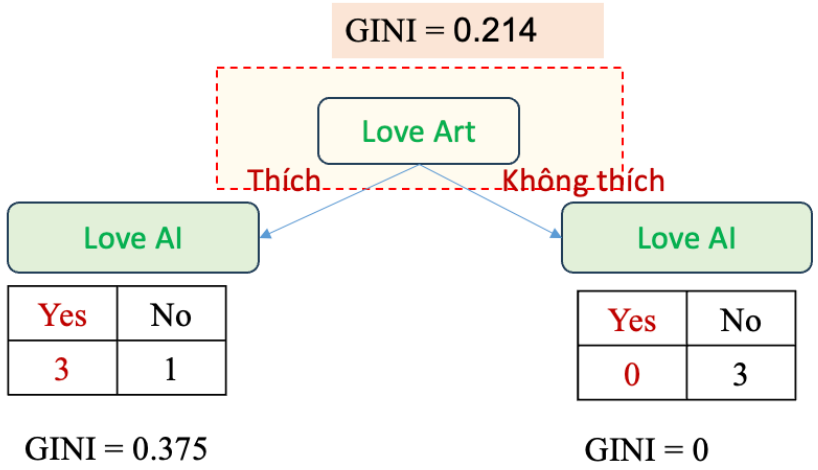
Age

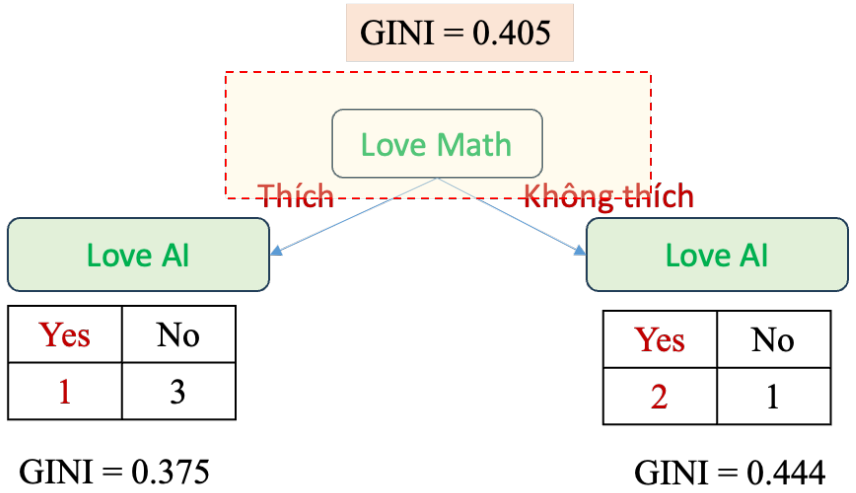
Which one is the root node?

GINI Example



Which attribute is in the first node?



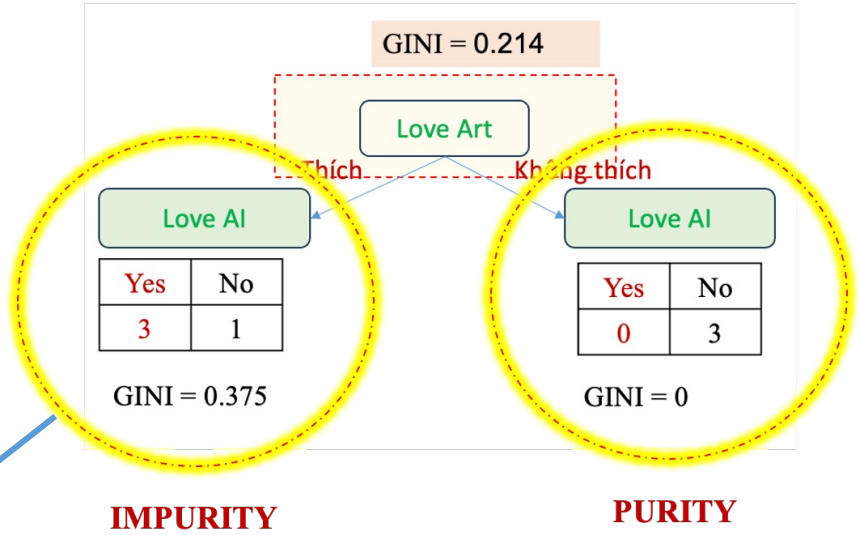


Love Art is the best one

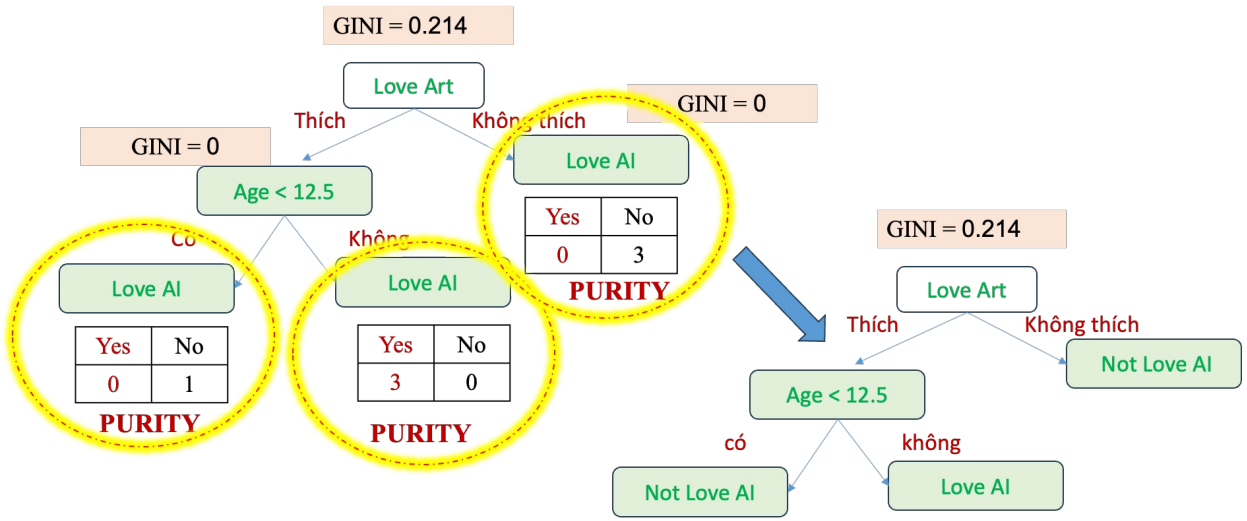
Cho biết nguyên tắc xây dựng cây bằng Gini?

GINI Solution

No.	Love Math	Love Art	Age	Love AI
1	Yes	Yes	7	No
2	Yes	No	12	No
3	No	Yes	18	Yes
4	No	Yes	35	Yes
5	Yes	Yes	38	Yes
6	Yes	No	50	No
7	No	No	83	No



No.	Love Math	Love Art	Age	Love AI
1	Yes	Yes	7	No
3	No	Yes	18	Yes
4	No	Yes	35	Yes
5	Yes	Yes	38	Yes

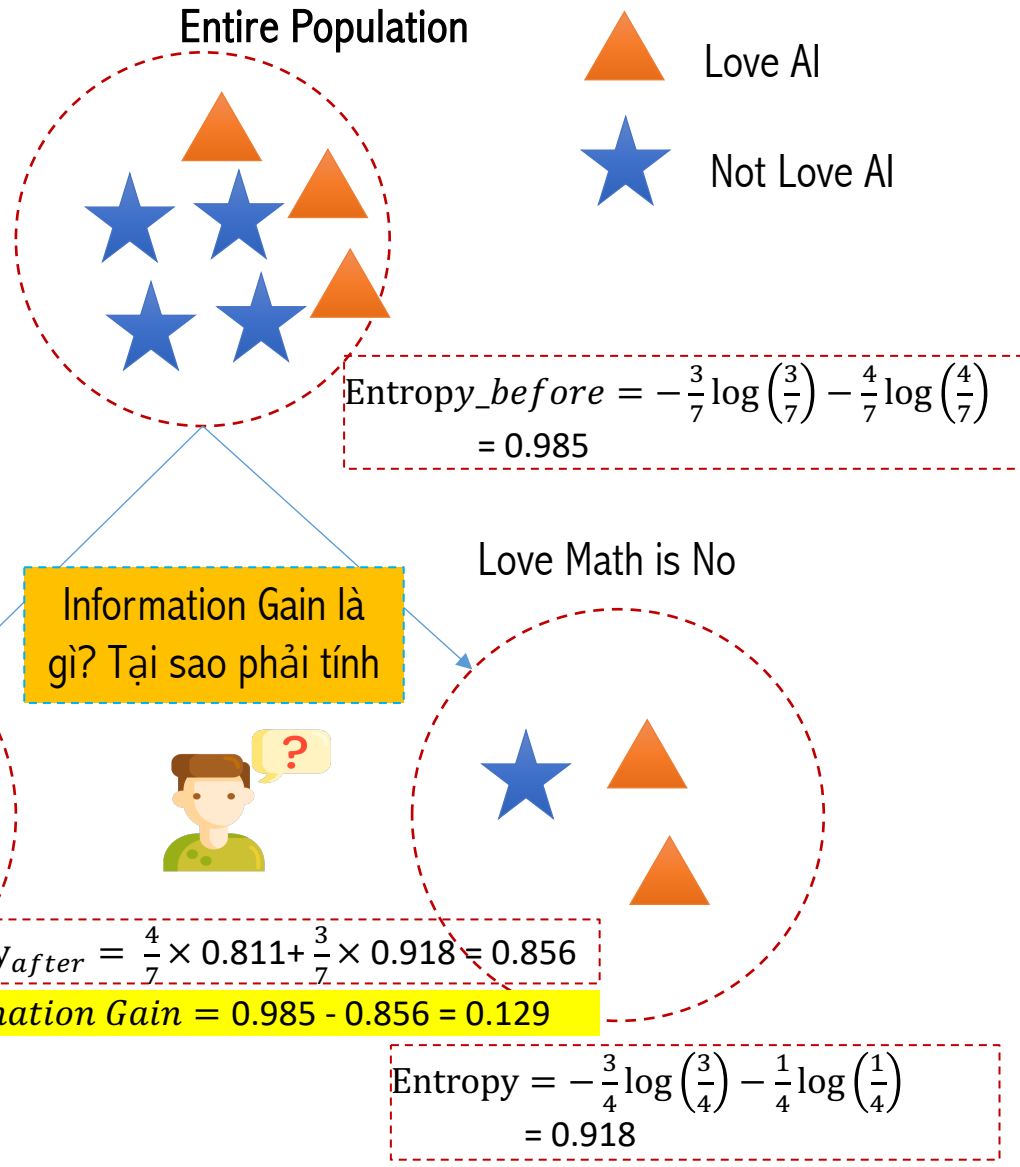


Entropy Example

No.	Love Math	Love Art	Love AI
1	Yes	Yes	No
2	Yes	No	No
3	No	Yes	Yes
4	No	Yes	Yes
5	Yes	Yes	Yes
6	Yes	No	No
7	No	No	No

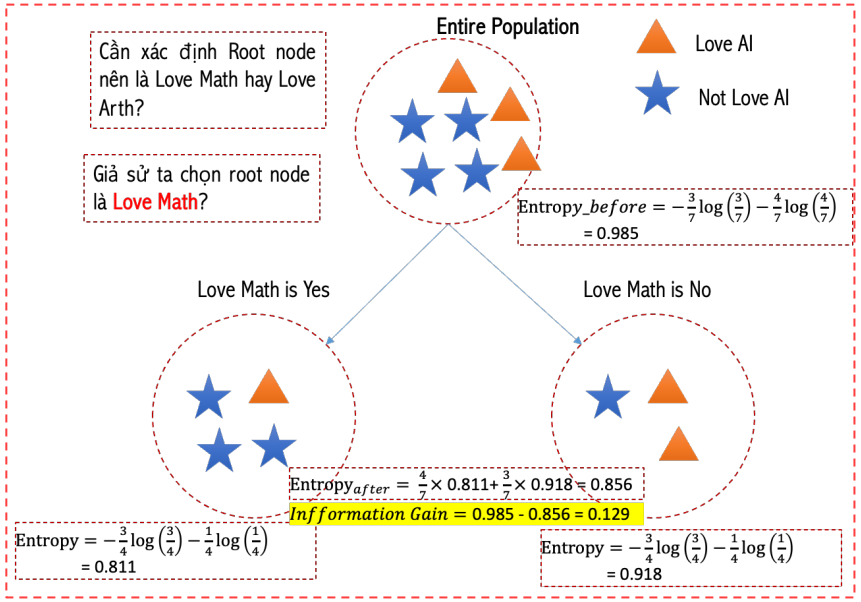
Cần xác định Root node nên là Love Math hay Love Art?

Giả sử ta chọn root node là Love Math?



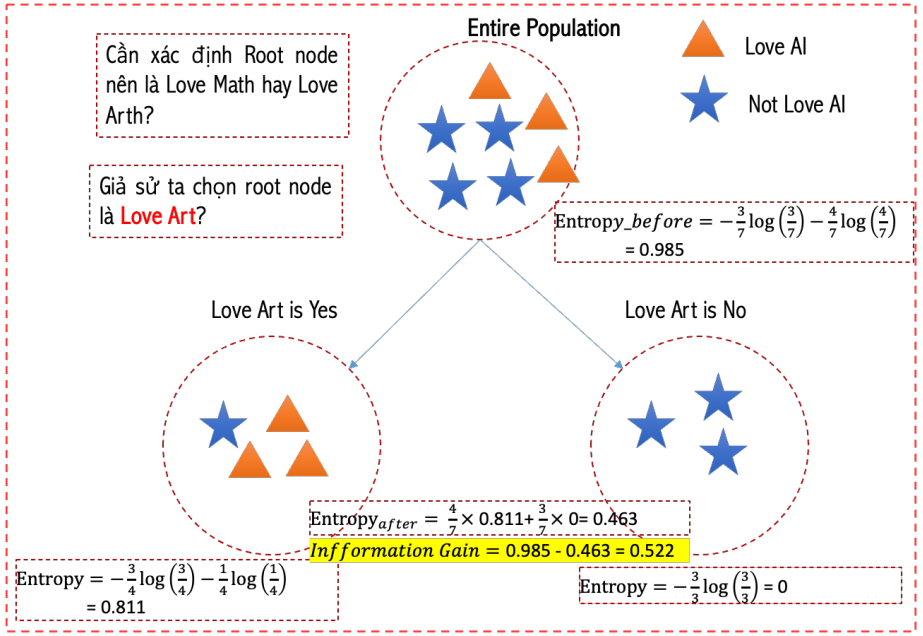
Entropy Example

Love Math is a Root Node



No.	Love Math	Love Art	Love AI
1	Yes	Yes	No
2	Yes	No	No
3	No	Yes	Yes
4	No	Yes	Yes
5	Yes	Yes	Yes
6	Yes	No	No
7	No	No	No

Love Art is a Root Node



$Information\ Gain$ = 0.129

Infformation Gain = 0.522

Which attribute is in the first node?

Cho biết nguyên tắc xây dựng cây bằng Entropy?

Decision Tree for Regression

Unit	Age	Sex	Effect (%)
10	25	Female	98
20	73	Male	0
35	54	Female	100
5	12	Male	44
...	...	...	...

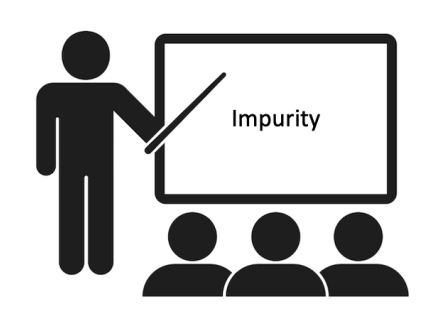
Can we still use Gini and Entropy for building a tree-based Regression?

Gini Index: $$Gini = 1 - \sum_j p_j^2$$

Entropy: $$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Decision Tree for Regression

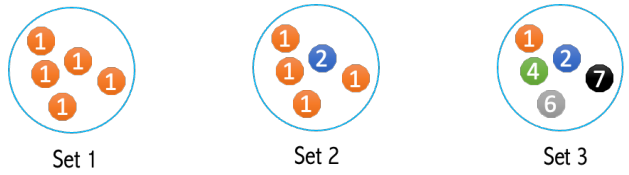
Classification Problem



Gini Impurity

Set 1	1	1	1	1	1
Set 2	1	1	2	1	1
Set 3	1	2	4	6	7

	Impurity Score
Set 1	Thấp
Set 2	Trung Bình
Set 3	Cao



Qualitative measurement:

Quantitative measurement:

$$Gini = 1 - \sum_j p_j^2$$

Regression Problem

Sum of Square Error (SSR)

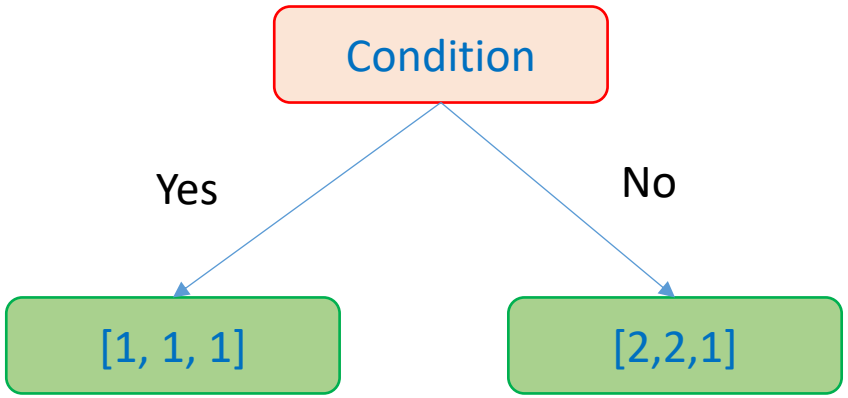
Set 1	78.5	90.5	55.6	60.5	70.5
Set 2	88.5	80.5	45.6	60.5	80.6
Set 2	98.5	70.5	65.6	60.5	79.5

$$\underbrace{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}_{\text{sum of the errors of all samples}}$$

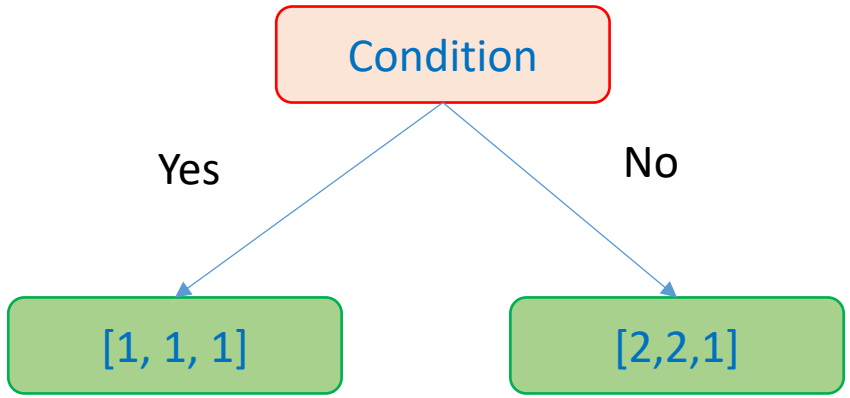
number of samples: n; real value: Y_i; predicted value: $\hat{Y}_i$

Output Value

Classification Problem



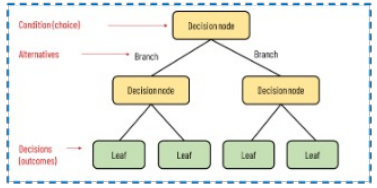
Regression Problem



1. What is the output value of each leaf node?
2. When shoud we stop building the tree?
3. What are disadvantages of Decision Tree?
4. Can DT obtain high performance in a regression problem?

Multiple Trees

5 Search: Cucci

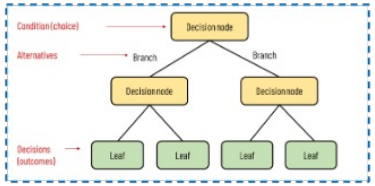


How to make the prediction based on multiple tree?

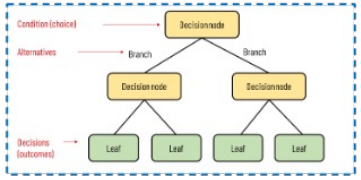
You want to buy a perfume for your girlfriend(s)?

What would you do?

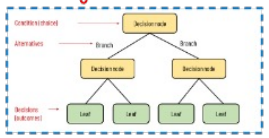
1 Channel đi con!



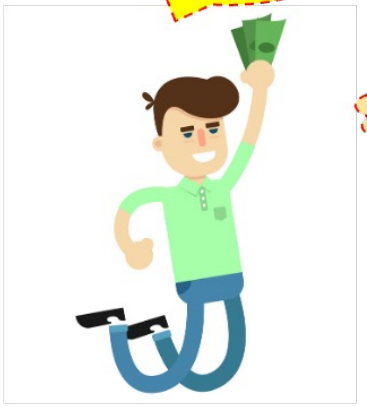
4 Cucci ạ!!



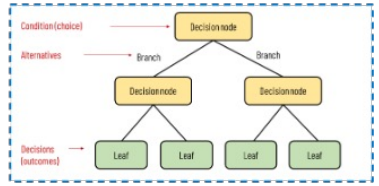
2 Channel đi bạn!!



Mua Gucci Thôi!!!



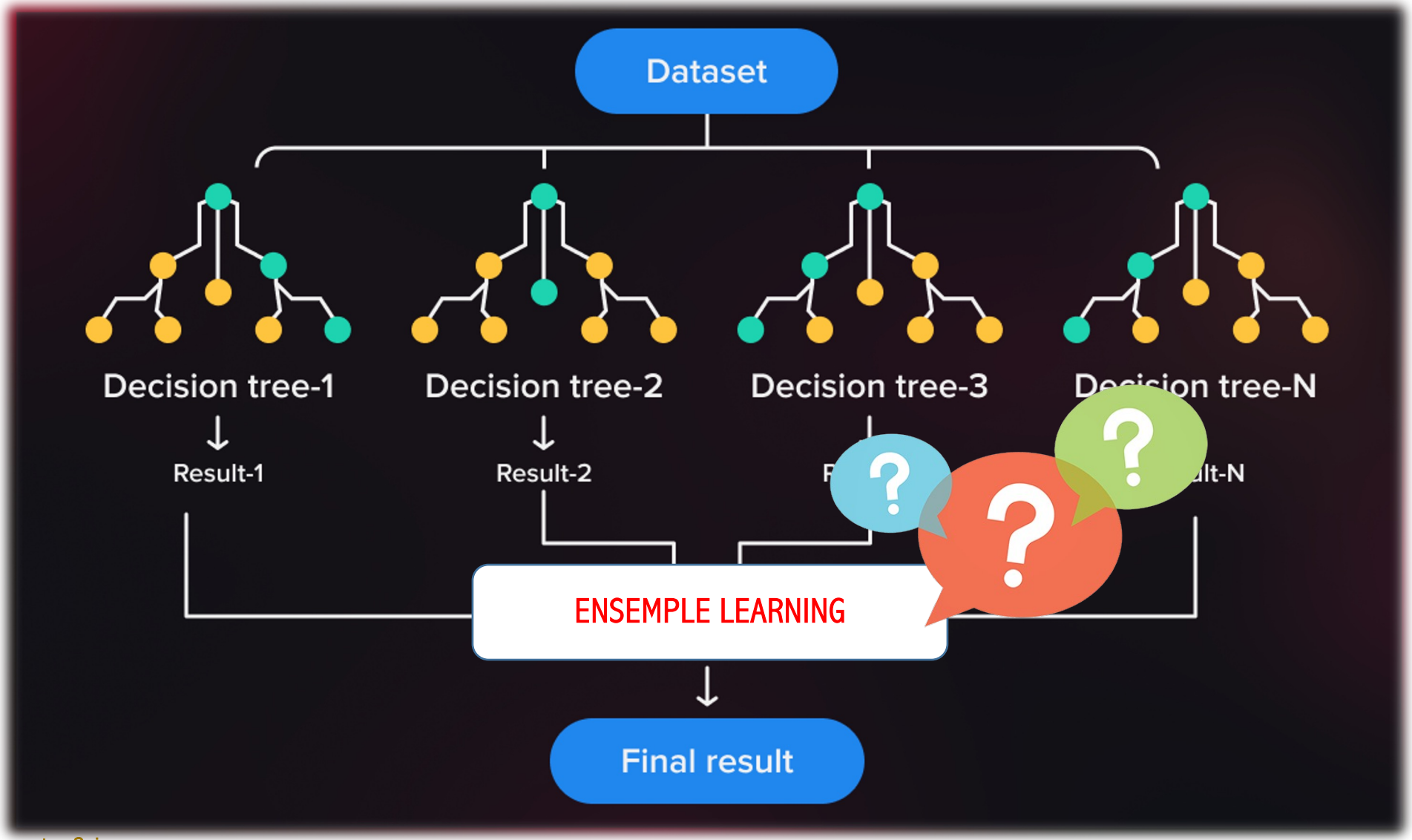
3 Cucci đi!!



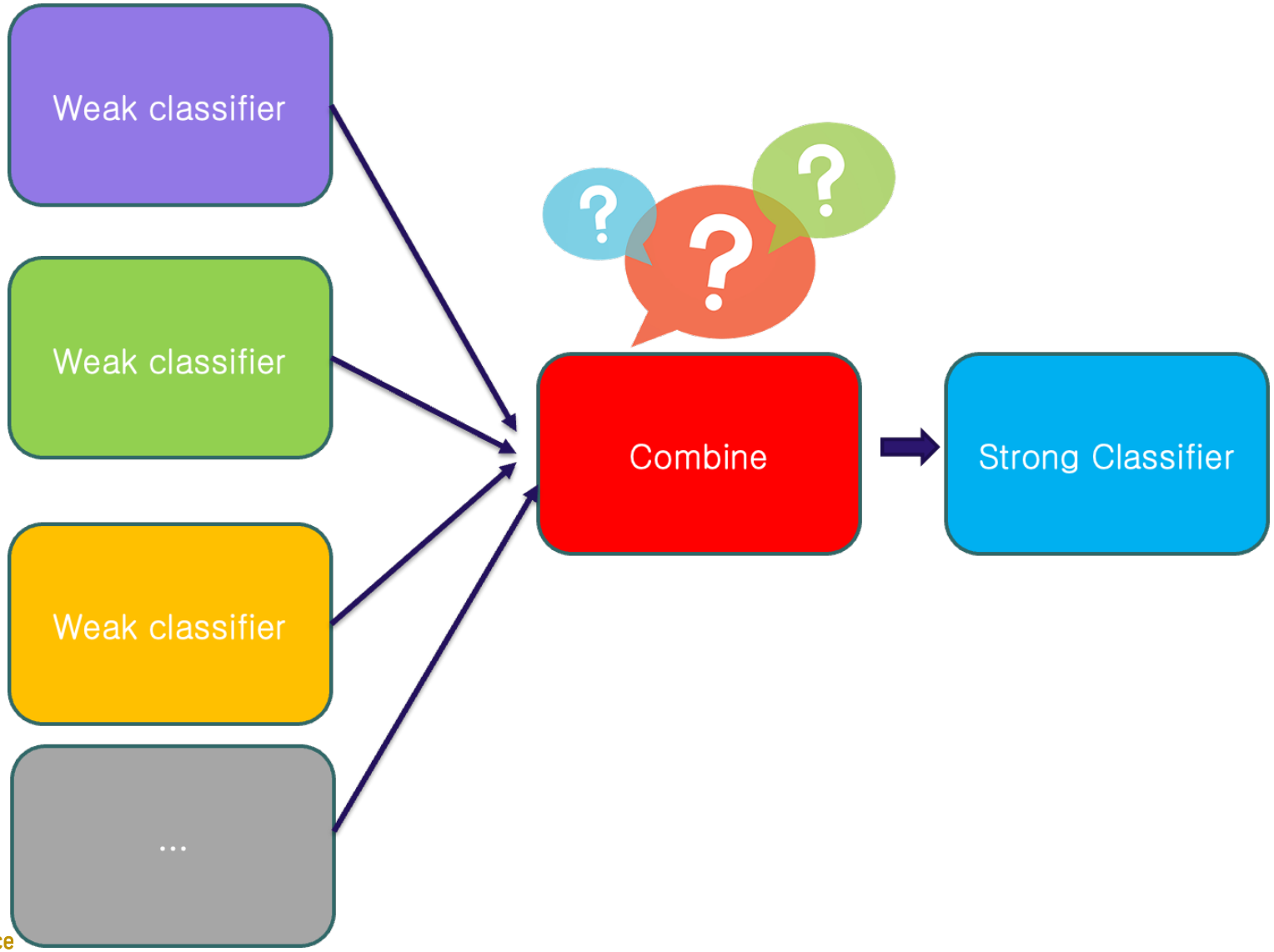
Outline

- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

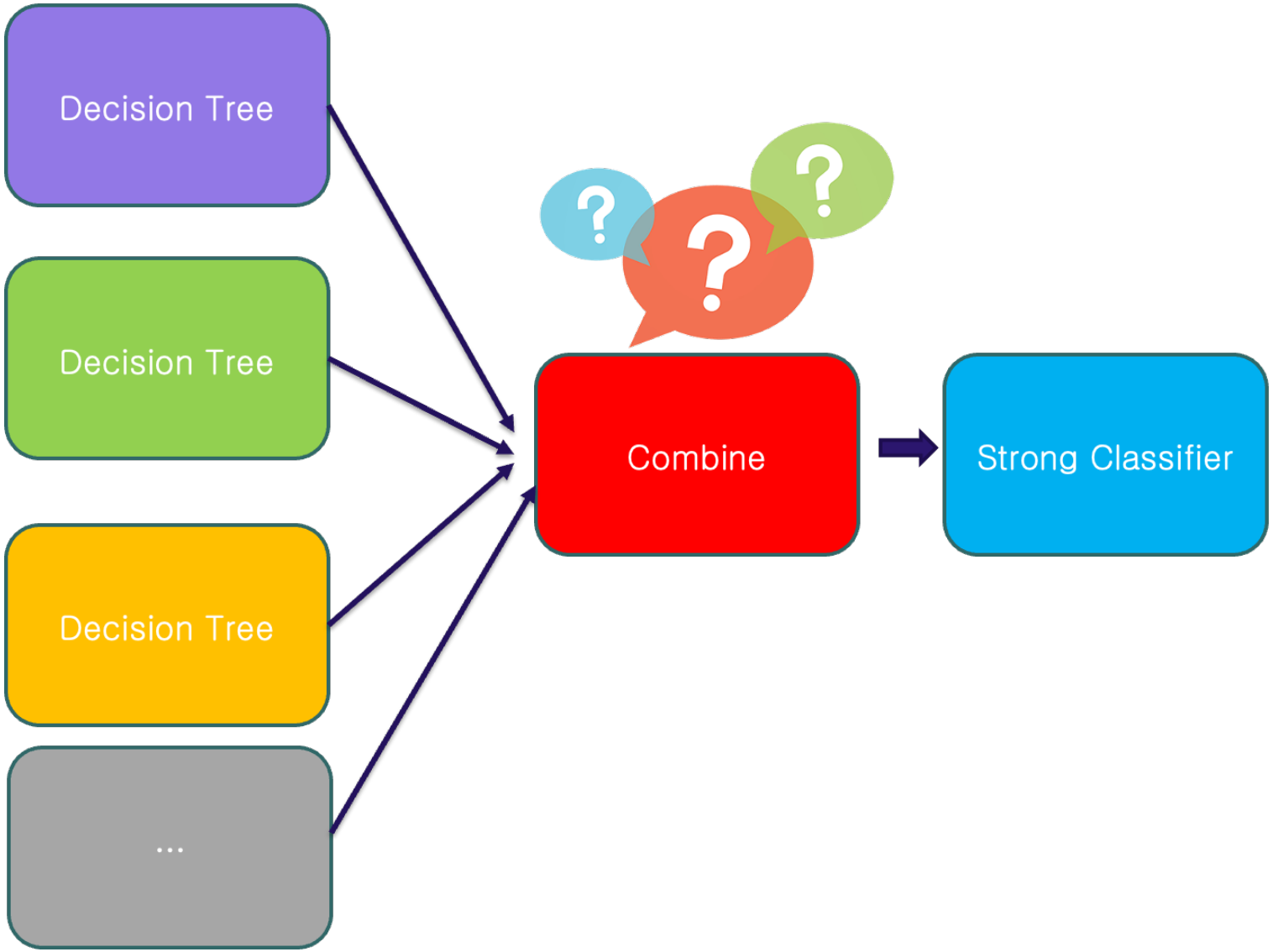
Ensemple Learning



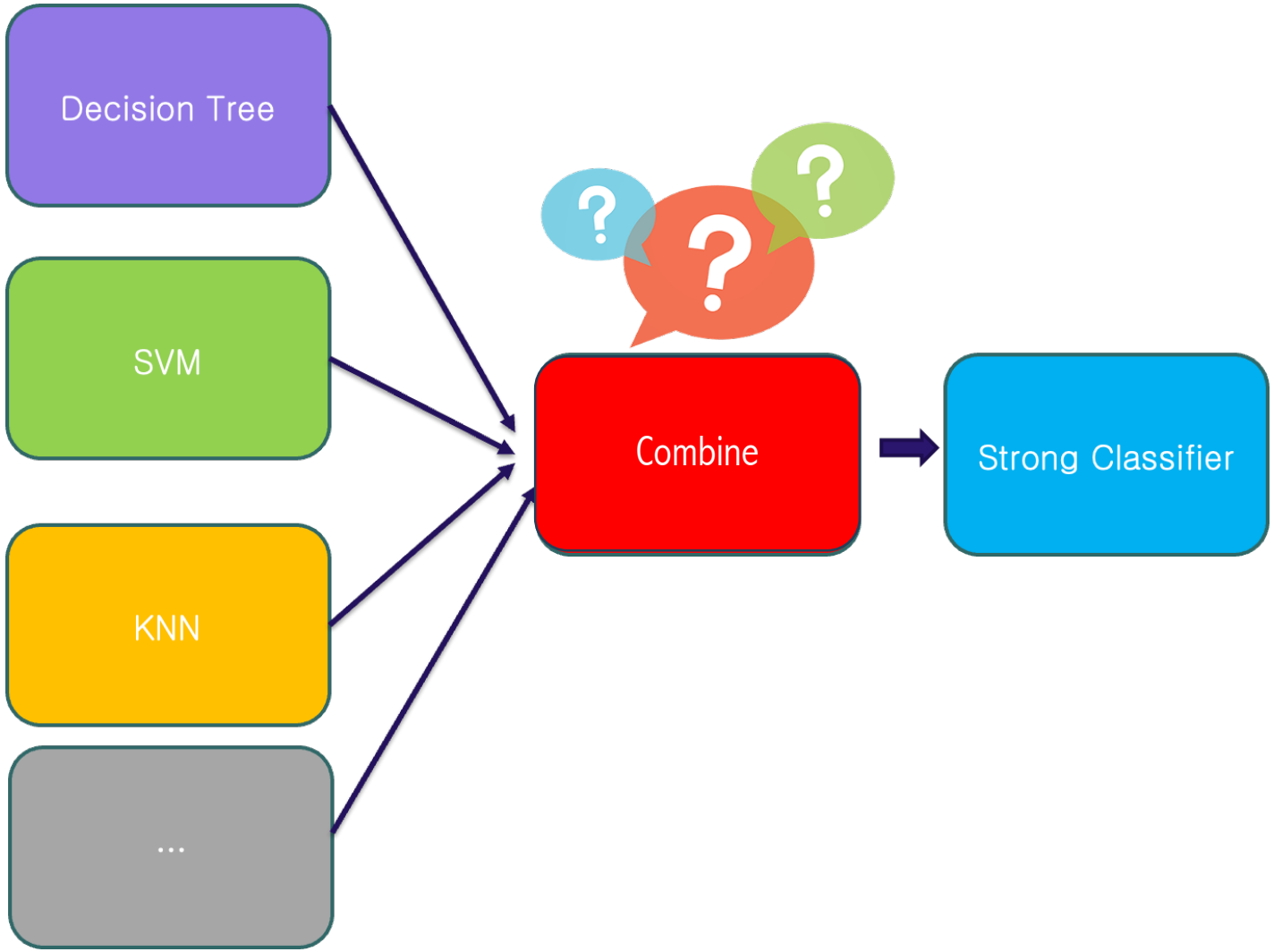
Ensemple Learning



Homogeneous Approach



Heterogeneous Approach

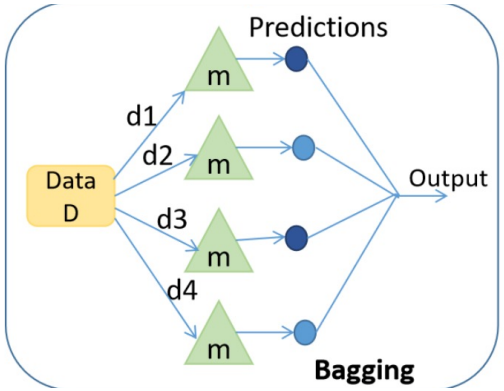


Ensemple Learning Techniques

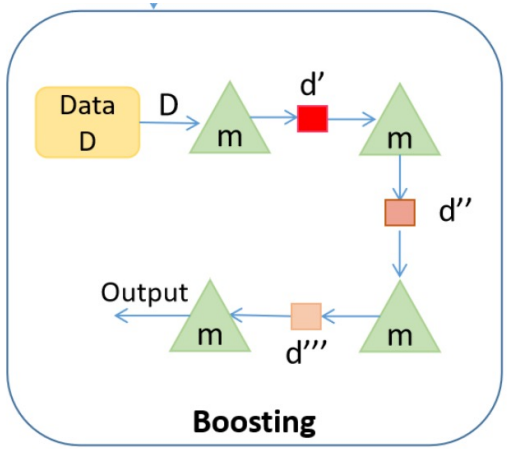
Ensemple Learning

Thông dụng ở các cuộc thi về AI

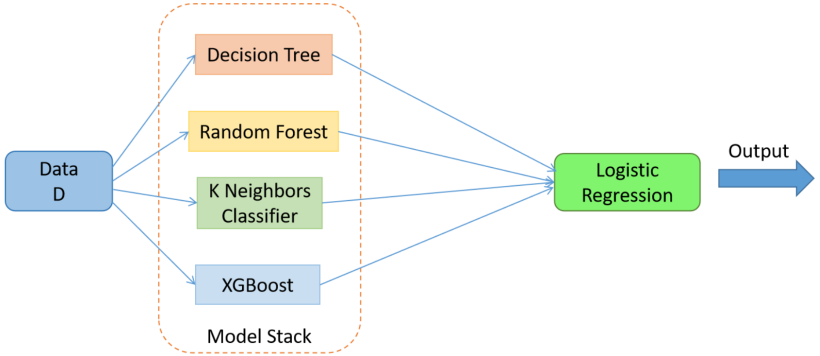
Bagging
homogeneous weak learners



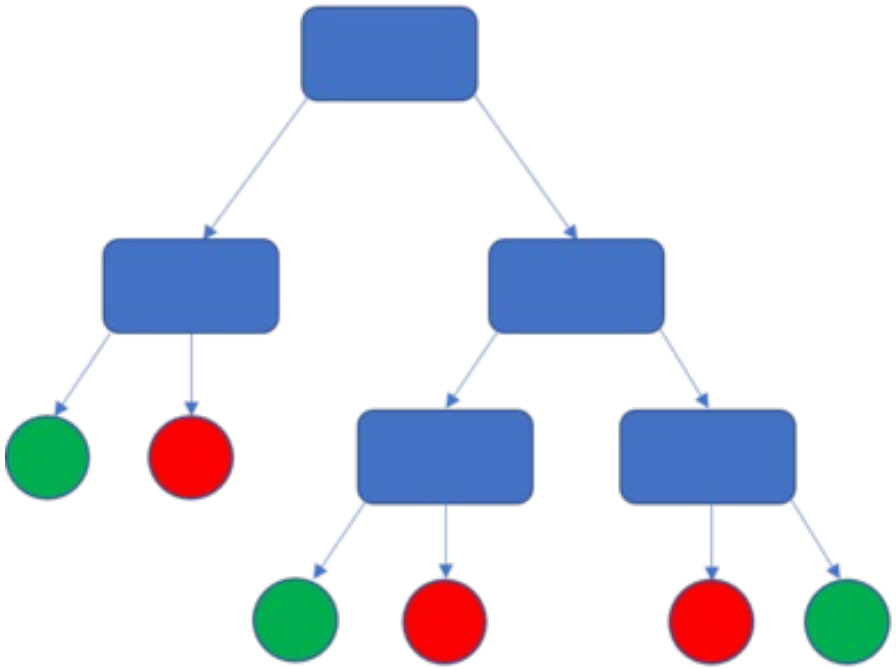
Boosting
homogeneous weak learners



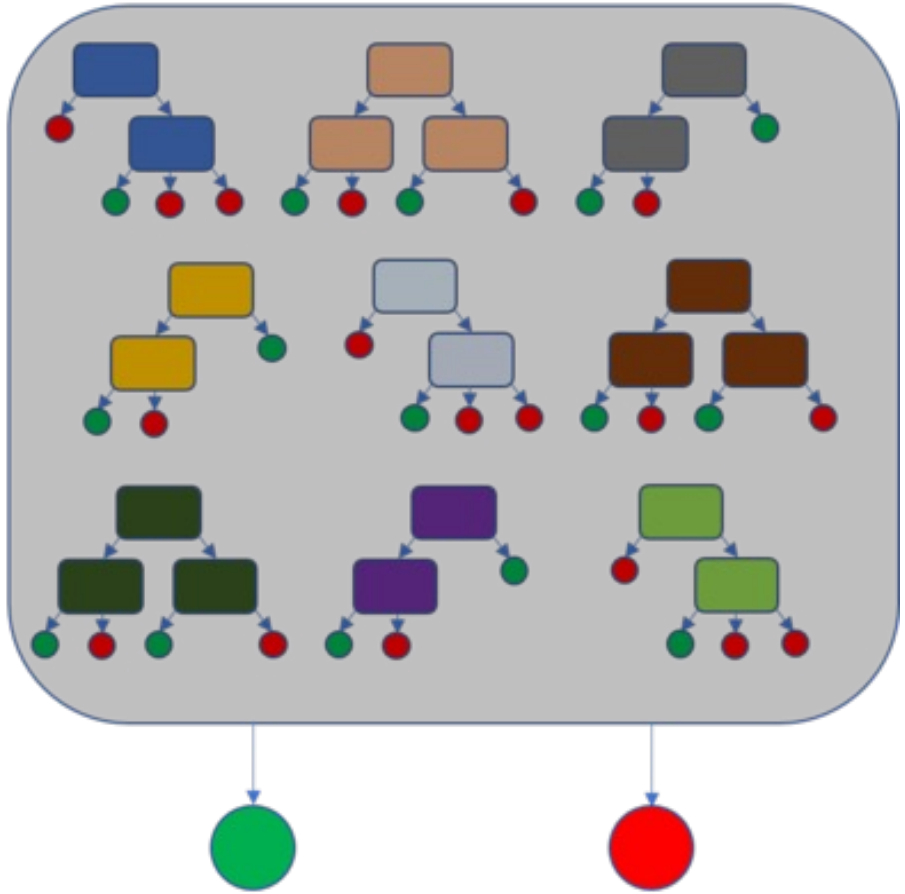
Stacking
Heterogeneous weak learners



Bagging-based Method



Decision Tree



Random Forest

Step to Random Forest

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

1st Step: Create a New Dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Original DATA

Chọn lựa ngẫu nhiên từ dataset ban đầu

New DATA

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

1st Step: Create a New Dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Original DATA

Chọn lựa ngẫu nhiên từ dataset ban đầu

Bootrapped Dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

2nd Step: Decision Tree from Boostrapped Dataset

GENERATE DECISION TREES FROM THE BOOTSTRAPPED DATASET USING **PREDEFINED CONDITIONS**

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

A RANDOM SUBSET OF 2 ATTRIBUTES (OR 2 COLUMNS).

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

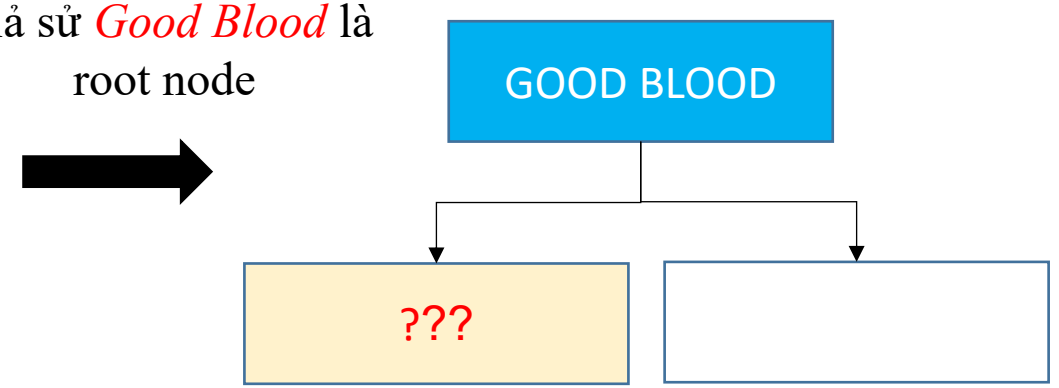
Traditional Tree

Tree with Predefined Conditions

Chọn lựa ngẫu nhiên 2 features (columns)

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

Giả sử *Good Blood* là root node



Loại bỏ Good Blood ra khỏi dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

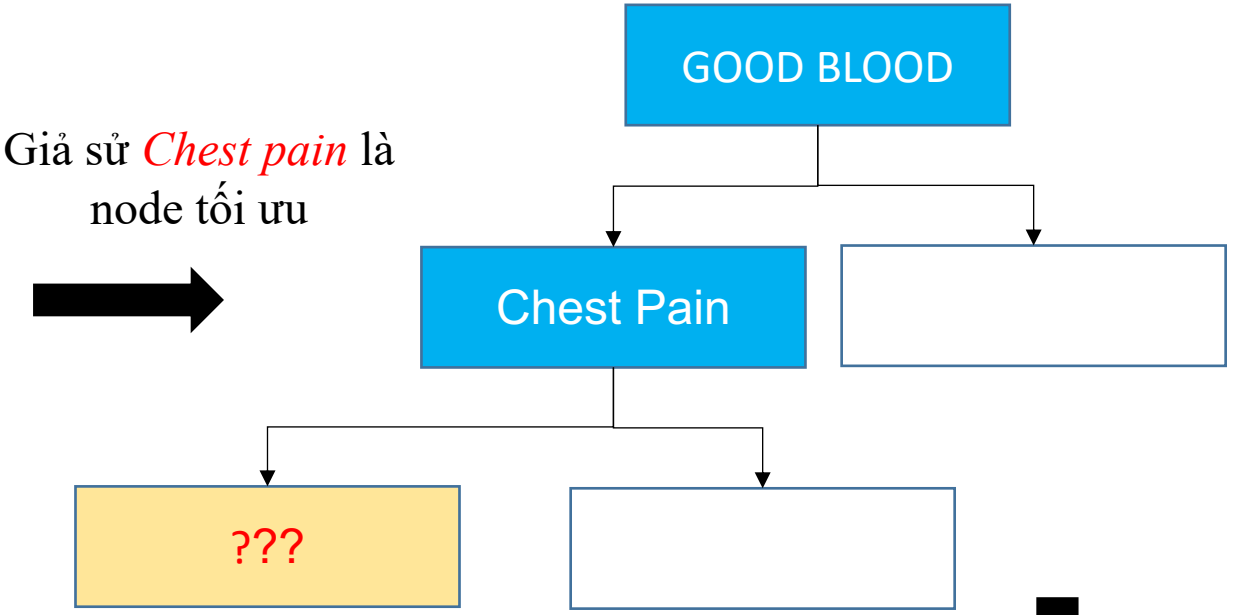
Chọn lựa ngẫu nhiên 2 features (columns)

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

Chọn lựa ngẫu nhiên 2 features (columns)

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

Giả sử *Chest pain* là node tối ưu



Chọn lựa ngẫu nhiên 2 features (columns)

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

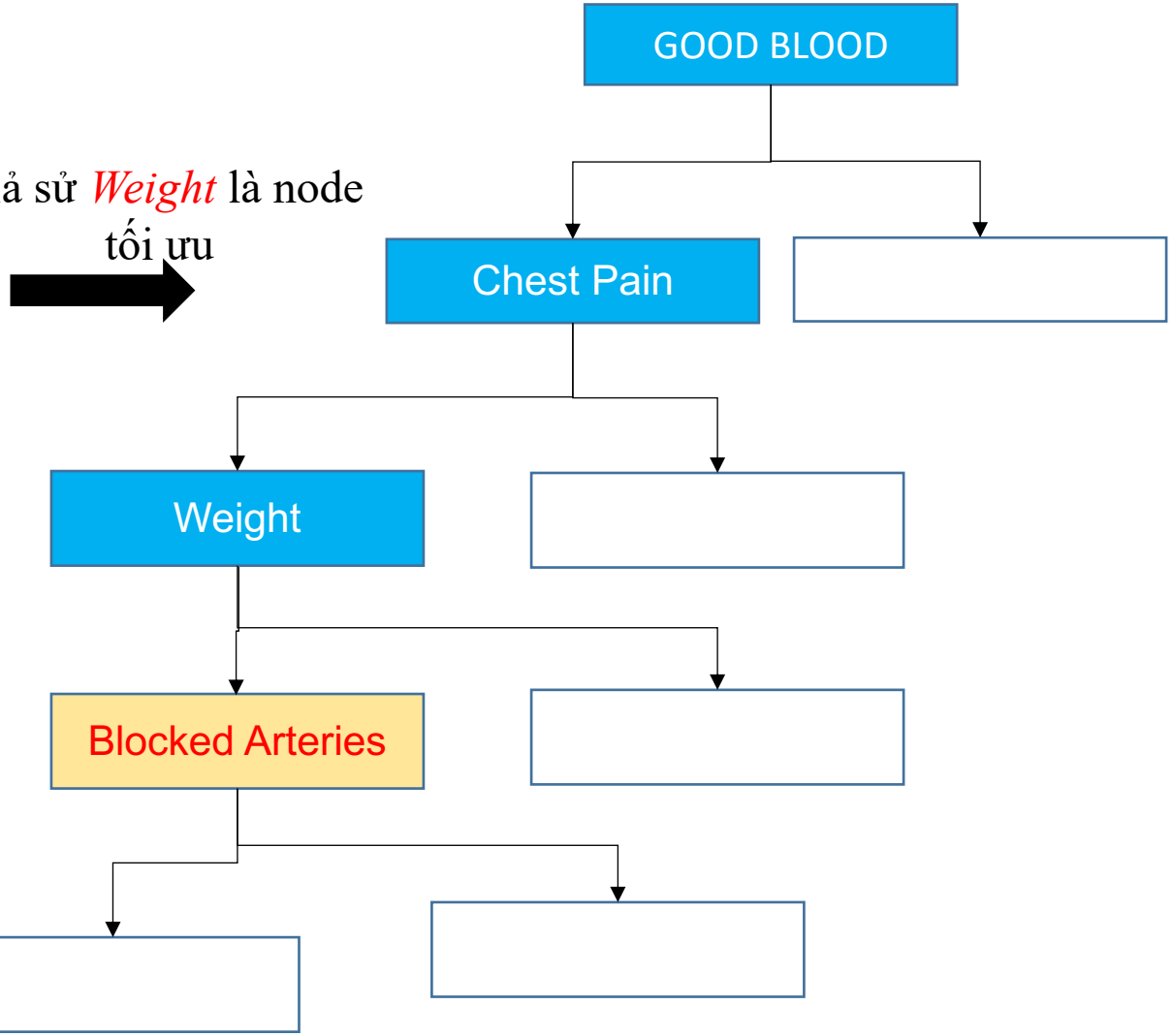
Loại bỏ *chest pain* ra khỏi dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

Chọn lựa ngẫu nhiên 2 features (columns)

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
		NO	125	NO
		YES	167	YES
YES	NO	YES	167	YES

Giả sử *Weight* là node tối ưu

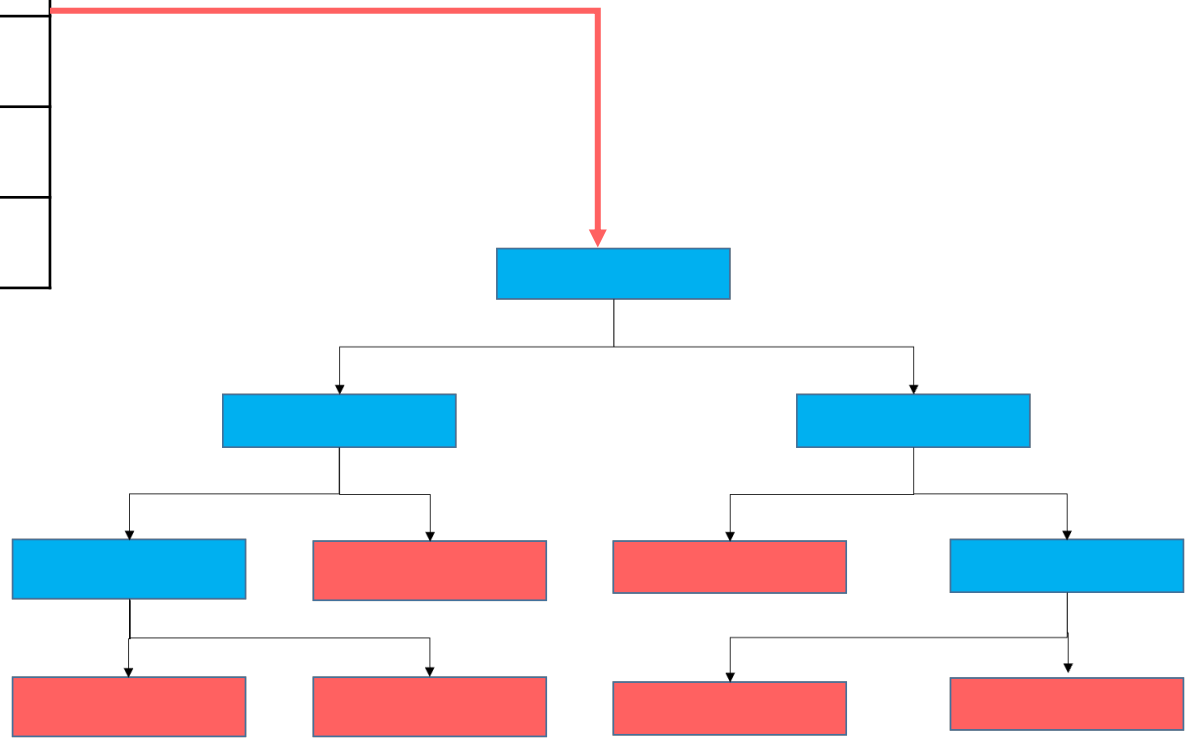


Loại bỏ *Weight* ra khỏi dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
		NO		NO
		YES		YES
YES	NO	YES	167	YES

1st Decision Tree

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES



Create N Tree

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Generate

1st bootstrapped dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

2nd bootstrapped dataset

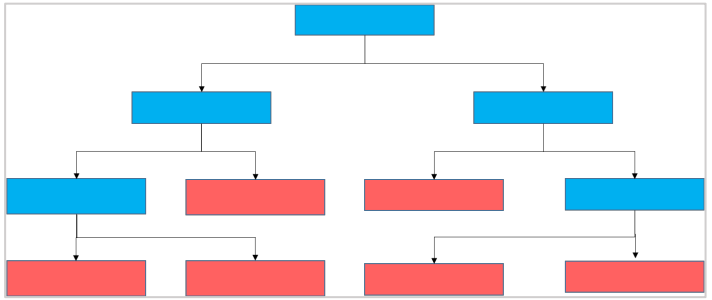
CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	YES	NO	210	NO

...

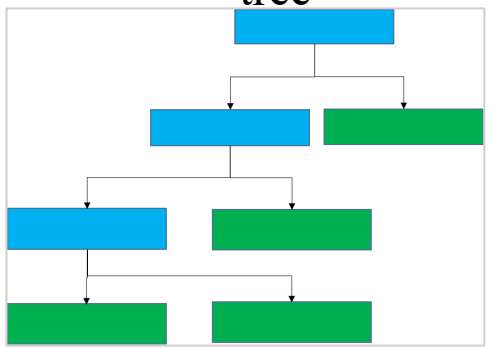
nth bootstrapped dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

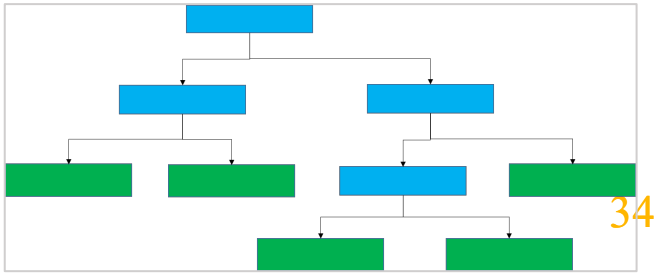
1st tree



2nd tree



nth tree



Create N Tree

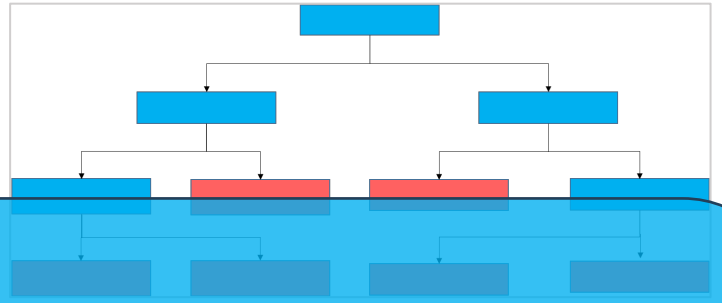
CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Generate

1st bootstrapped dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

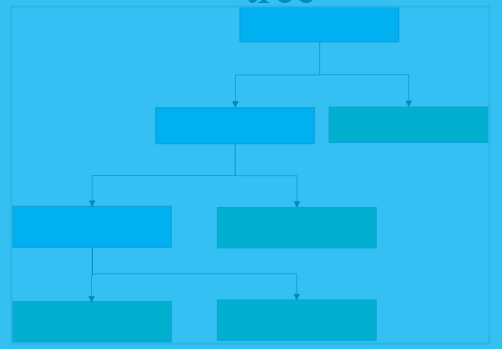
1st tree



2nd bootstrapped dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	YES	NO	210	NO

2nd tree

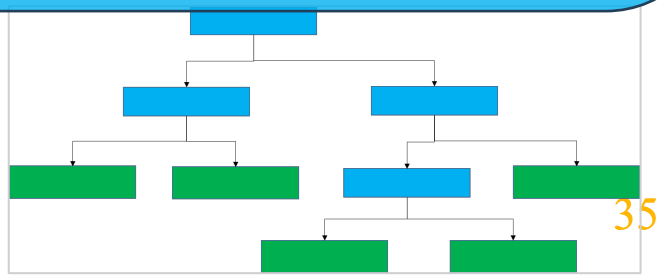


...

nth bootstrapped dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

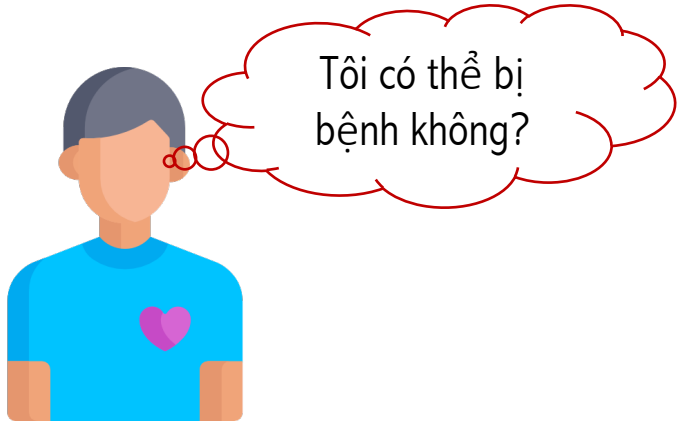
nth tree

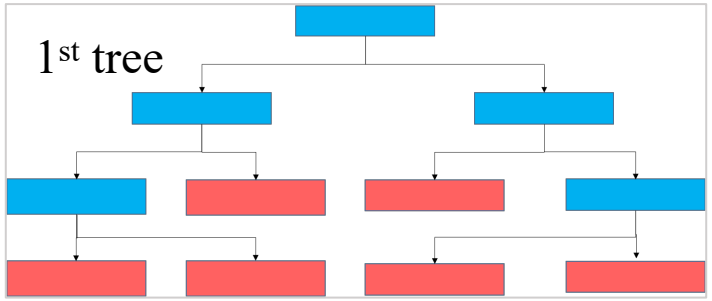


Random Forest

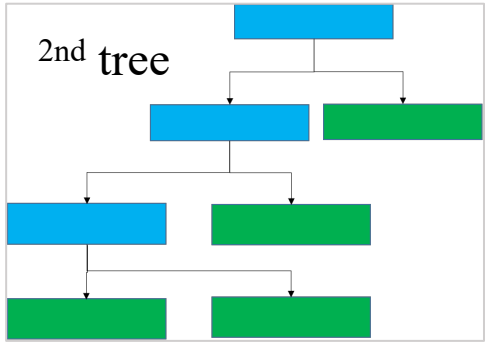
How to Predict New Sample

Chest Pain	No
GOOD BLOOD CIRCULATION	No
BLOCKED ARTERIES	No
Weight	125

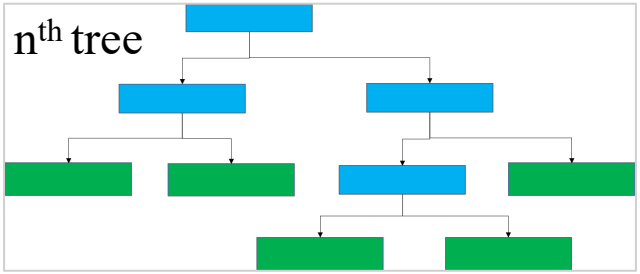




Predict



Predict



Predict

Heart Disease

Yes	No
7	2

Out-of-bag Dataset

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

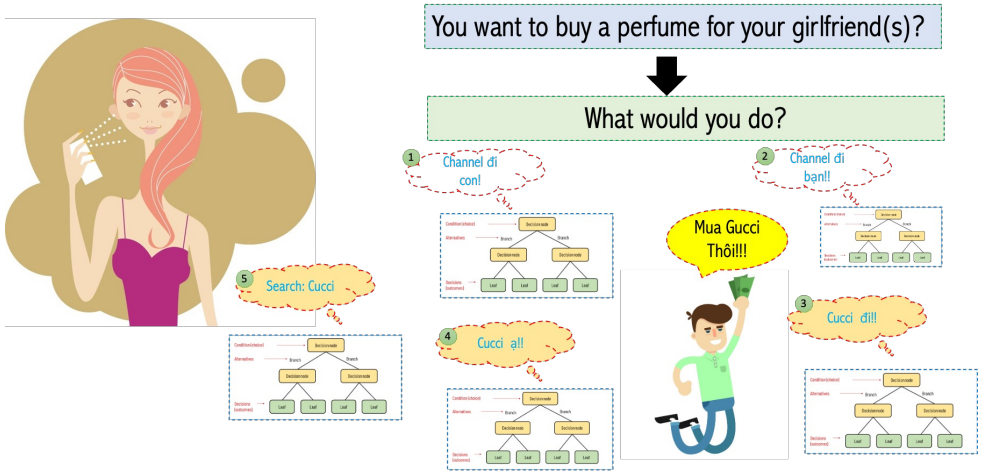
CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	NO	210	NO

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

OUT-OF-BAG ERROR

1. How to create out-of-bag dataset?
2. Please show me its benefits?

Random Forest



1. Can we use RF for filling missing data? And How?
2. How many trees in RF?
3. RF instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features?
4. What are limitations of RF?

The random forest is a classification algorithm consisting of many decisions trees. **It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees** whose prediction by committee is more accurate than that of any individual tree.

Why Random Feature to Build a Tree: RF

AIO2023 use a uniformly distributed random number generator to produce a number.
If the number AIO2023 generates, is greater than or equal to 40, you win (so you have a 60% chance of victory) and AIO2023 pays you some money. If it is below 40, AIO2023 win, and you pay AIO2023 the same amount

Game 1
Play 100 times, betting $1 each time.

Game 2
Play 10 times, betting $10 each time.

Game 3
Play 1 times, betting $100 each time.

Which game would you pick?

Why Random Feature to Build a Tree: RF

Game 1
Play 100 times, betting $1 each time.

Expected Value:
= (0.60*1 + 0.40*-1)*100 = 20

Game 2
Play 10 times, betting $10 each time.

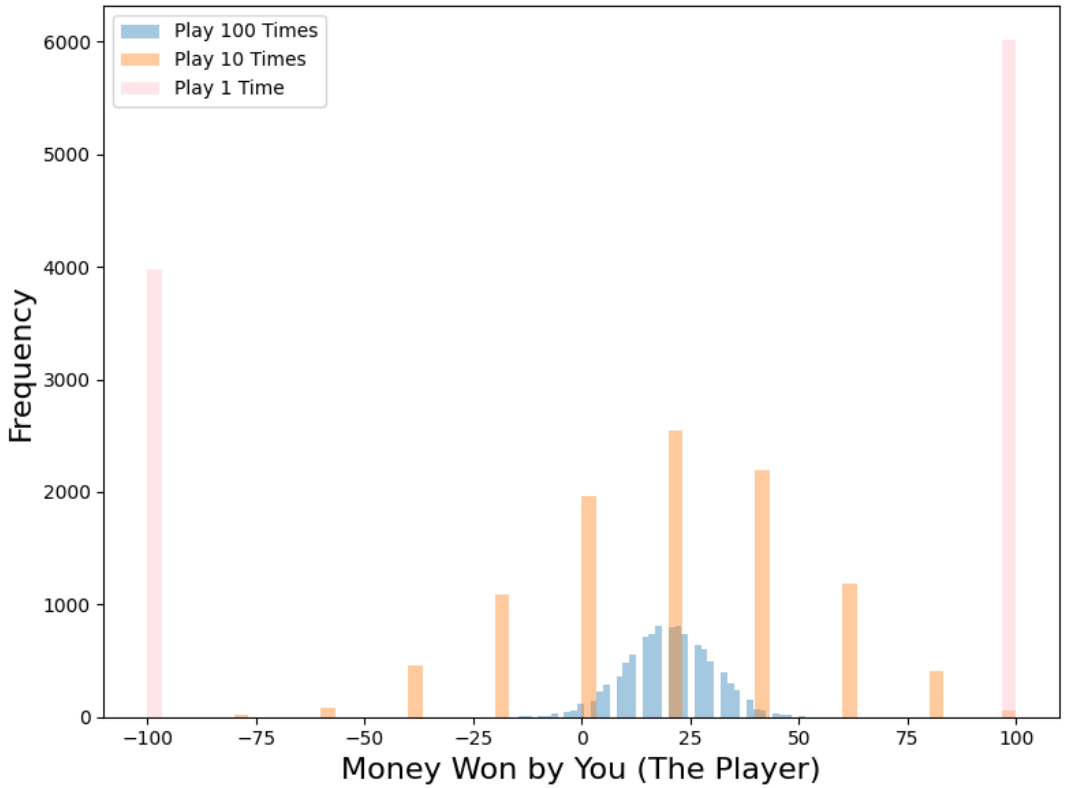
Expected Value:
(0.60*10 + 0.40*-10)*10 = 20

Game 3
Play 1 times, betting $100 each time.

Expected Value:
= 0.60*100 + 0.40*-100 = 20

What about the distributions?

We will run 10,000 simulations of each game type; **for example, we will simulate 10,000 times the 100 plays of Game 1**



Which game would you pick?

Why Random Feature to Build a Tree: RF

Game 1
Play 100 times, betting $1 each time.

Expected Value:
= (0.60*1 + 0.40*-1)*100 = 20

You make money in 97% of them

Game 2
Play 10 times, betting $10 each time.

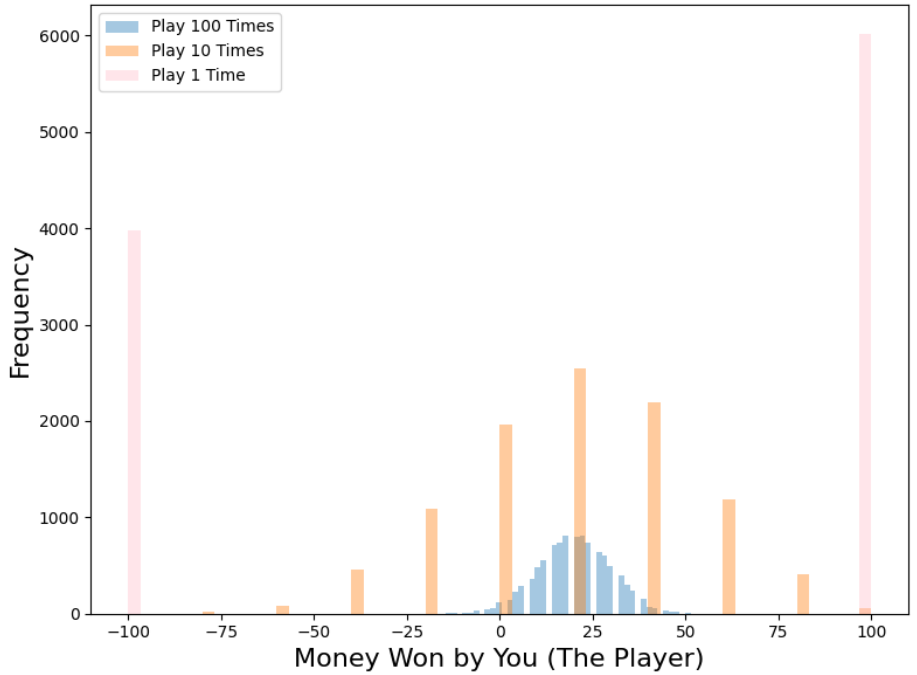
Expected Value:
(0.60*10 + 0.40*-10)*10 = 20

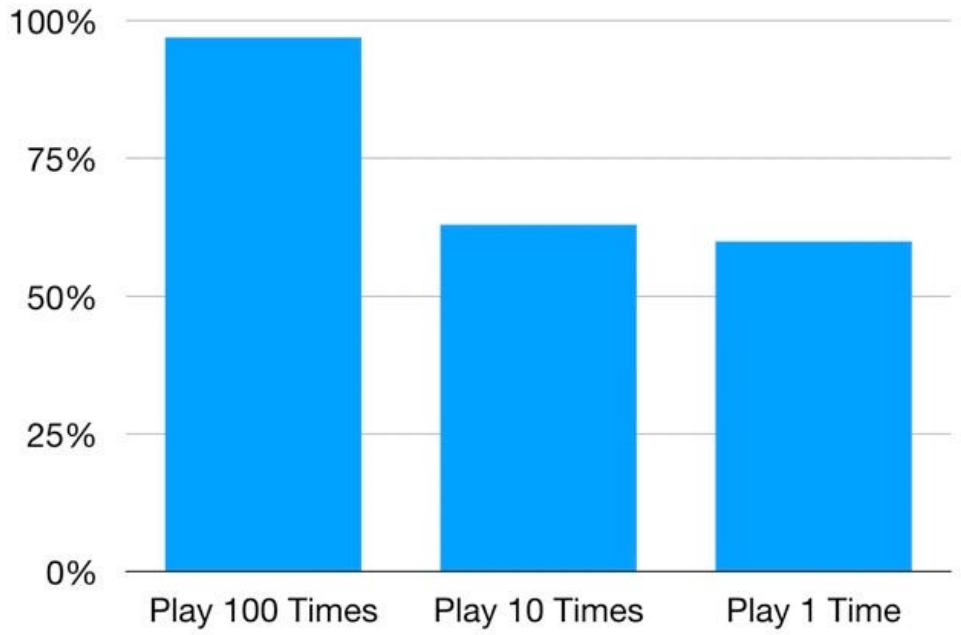
You make money in 63% of them

Game 3
Play 1 times, betting $100 each time.

Expected Value:
= 0.60*100 + 0.40*-100 = 20

You make money in 60% of them





Why Random Feature to Build a Tree: RF

Game 1	Game 2	Game 3
Play 100 times, betting $1 each time.	Play 10 times, betting $10 each time.	Play 1 times, betting $100 each time.
Expected Value: = (0.60*1 + 0.40*-1)*100 = 20	Expected Value: (0.60*10 + 0.40*-10)*10 = 20	Expected Value: = 0.60*100 + 0.40*-100 = 20
You make money in 97% of them	You make money in 63% of them	You make money in 60% of them

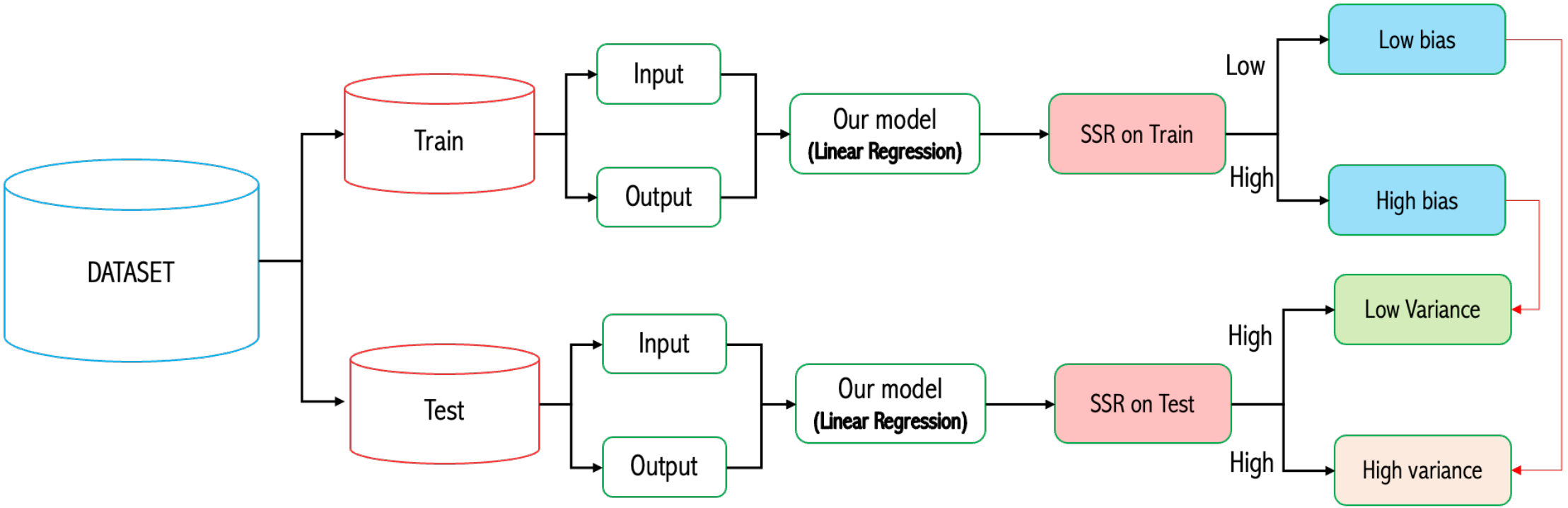
The more we split up our $100 bet into different plays, the more confident we can be that we will make money. As mentioned previously, this works because each play is independent of the other ones

Random forest is the same — each tree is like one play in our game earlier. We just saw how our chances of making money increased the more times we played. Similarly, with a random forest model, our chances of making correct predictions increase with the number of uncorrelated trees in our model.

Outline

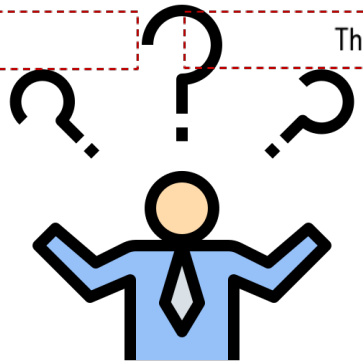
- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

Bias-Variance Trade-off



Bias as the error rate of the training data.

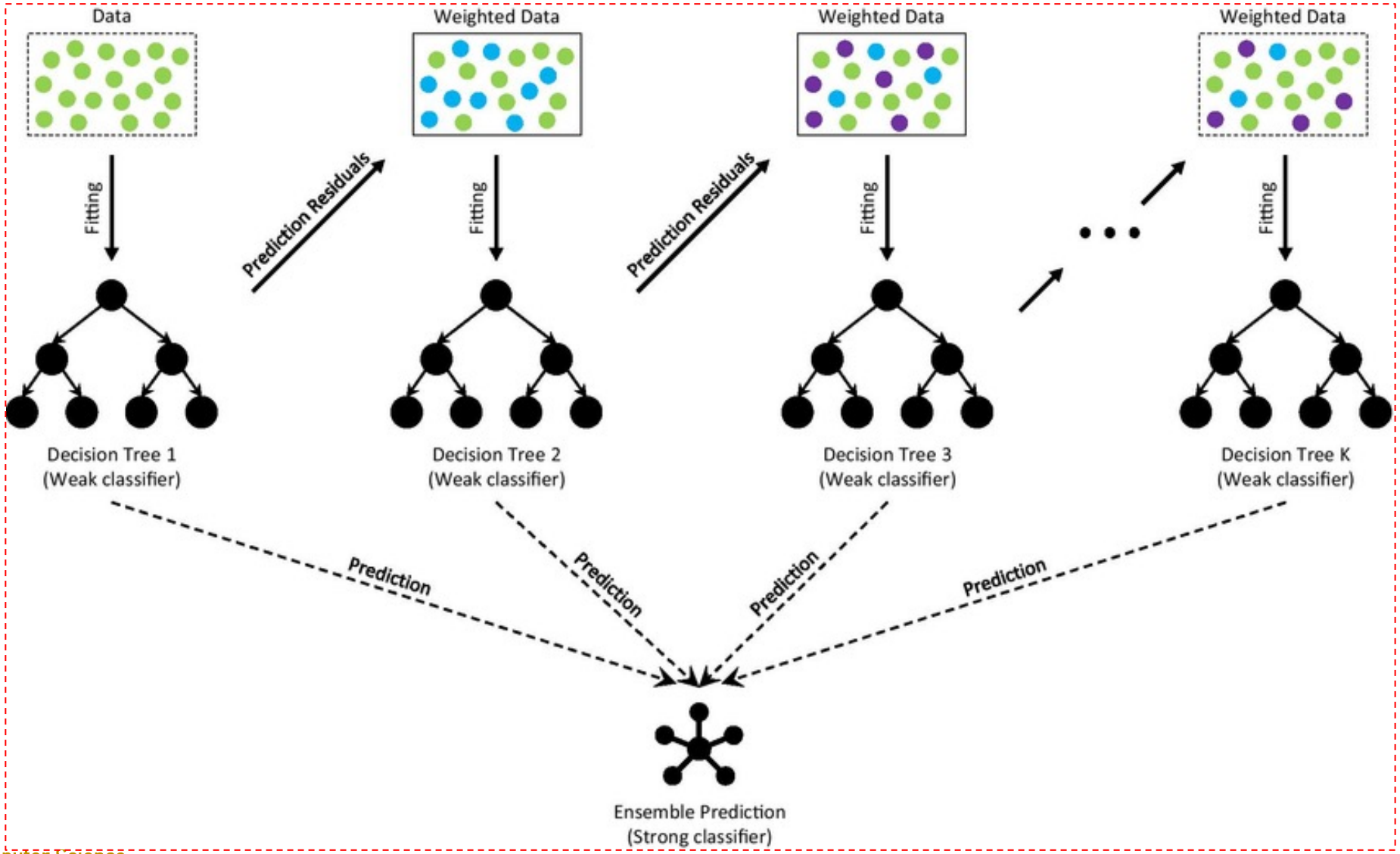
The different in fits between datatasets is call variance



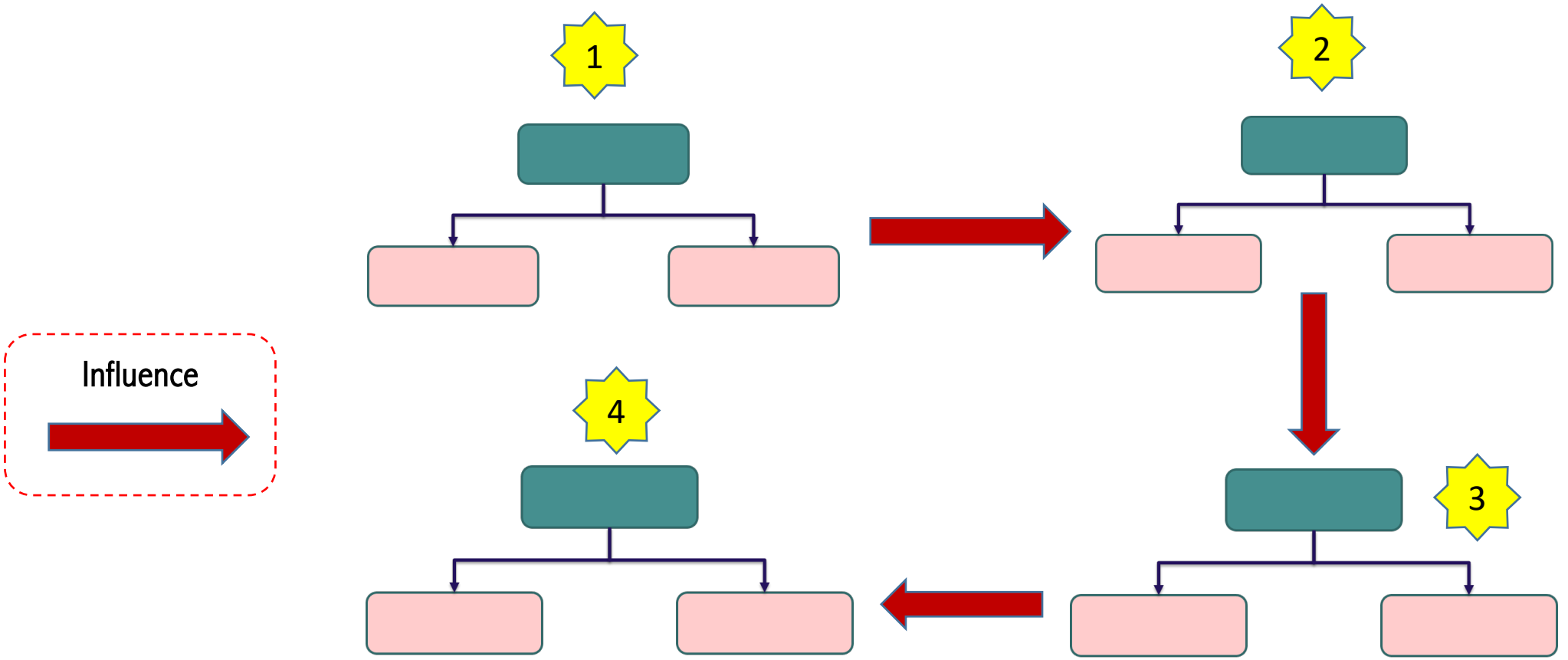
RF decreases the variance and helps to avoid overfitting.
Aim to decrease variance, not bias

Boosting techniques aims to decrease bias, not variance.

Boosting-Based Method



AdaBoost: FOREST OF STUMP



Adaboost builds a stump based on the the error made by previous stumps

Heart Disease Dataset

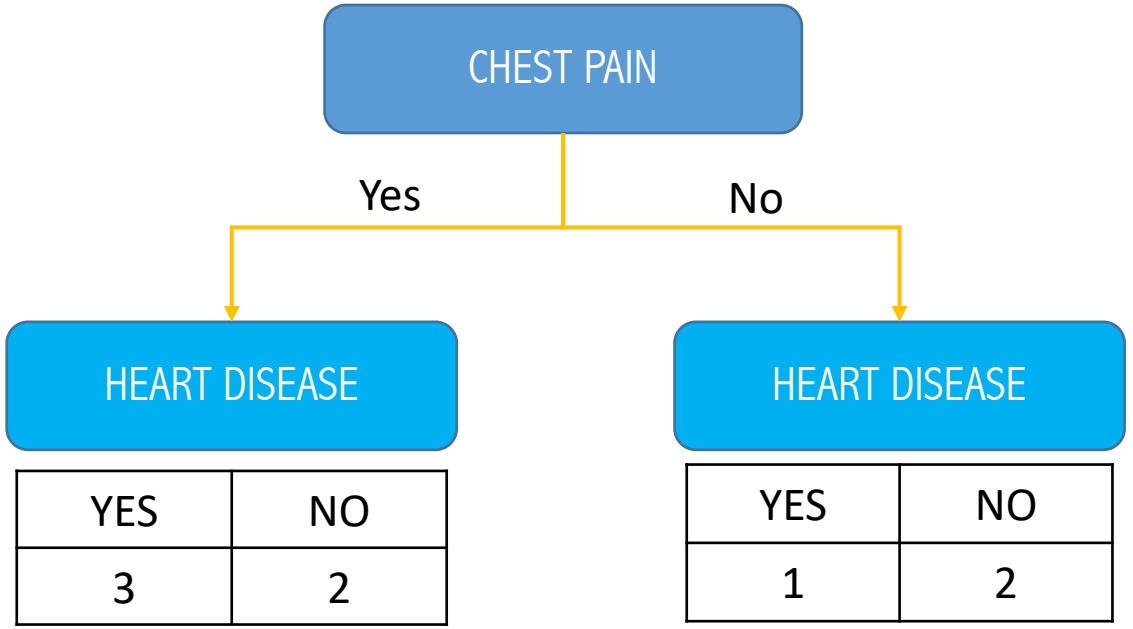
Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	No	168	No
Yes	Yes	172	No

Important of sample = Sample weight = 1 / number of samples = 1/8

Compute Gini Index For Chest Pain

Chest Pain	Heart Diease	Sample Weight
Yes	Yes	1/8
No	Yes	1/8
Yes	Yes	1/8
Yes	Yes	1/8
No	No	1/8
No	No	1/8
Yes	No	1/8
Yes	No	1/8

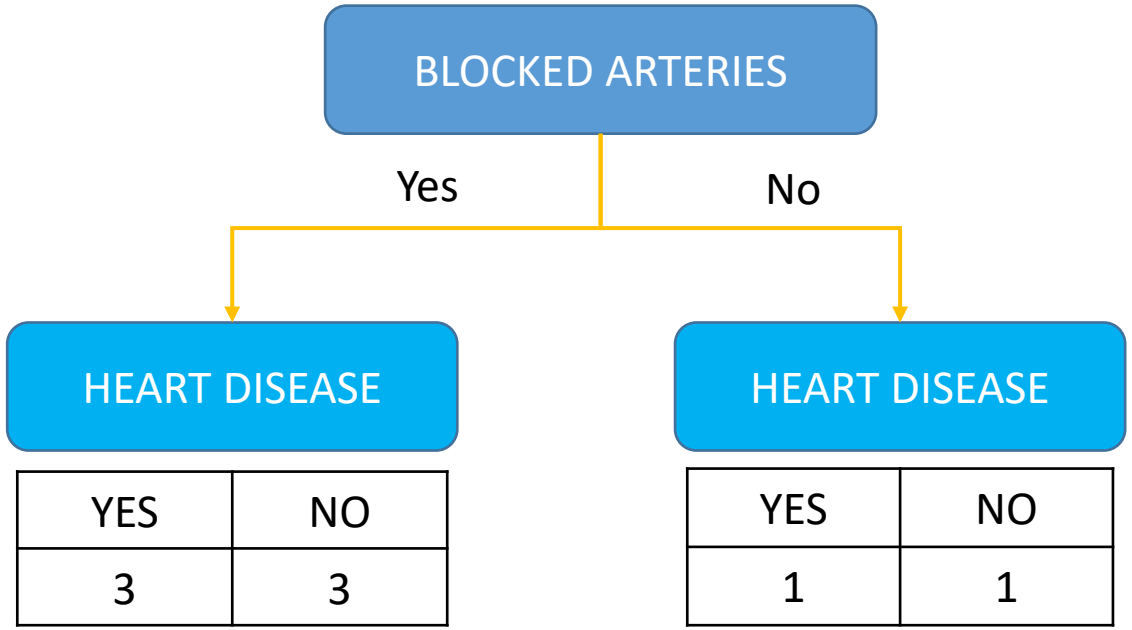
Gini index = 5/8 * (1 – (3/5)^2 – (2/5)^2) + 3/8* (1 - (1/3)^2 - (2/3)^2) = **0.57**



GINI INDEX FOR BLOCKED ARTERIES

Blocked Arteries	Heart Diease	Sample Weight
Yes	Yes	1/8
Yes	Yes	1/8
No	Yes	1/8
Yes	Yes	1/8
Yes	No	1/8
Yes	No	1/8
No	No	1/8
Yes	No	1/8

Gini index = 6/8 * (1 – (3/6)^2 – (3/6)^2) + 2/8* (1 - (1/2)^2 - (1/2)^2) = **0.5**



YES	NO
3	3

YES	NO
1	1

Gini Index for Heart Disease

Patient Weight	Heart Diease	Sample Weight
205	Yes	1/8
180	Yes	1/8
210	Yes	1/8
167	Yes	1/8
156	No	1/8
125	No	1/8
168	No	1/8
172	No	1/8

Gini index = =4/8 * (1-(1/4)^2 - (3/4)^2) + 4/8* (1-(1/4)^2 -(3/4)^2) = 0.375

192.5

195

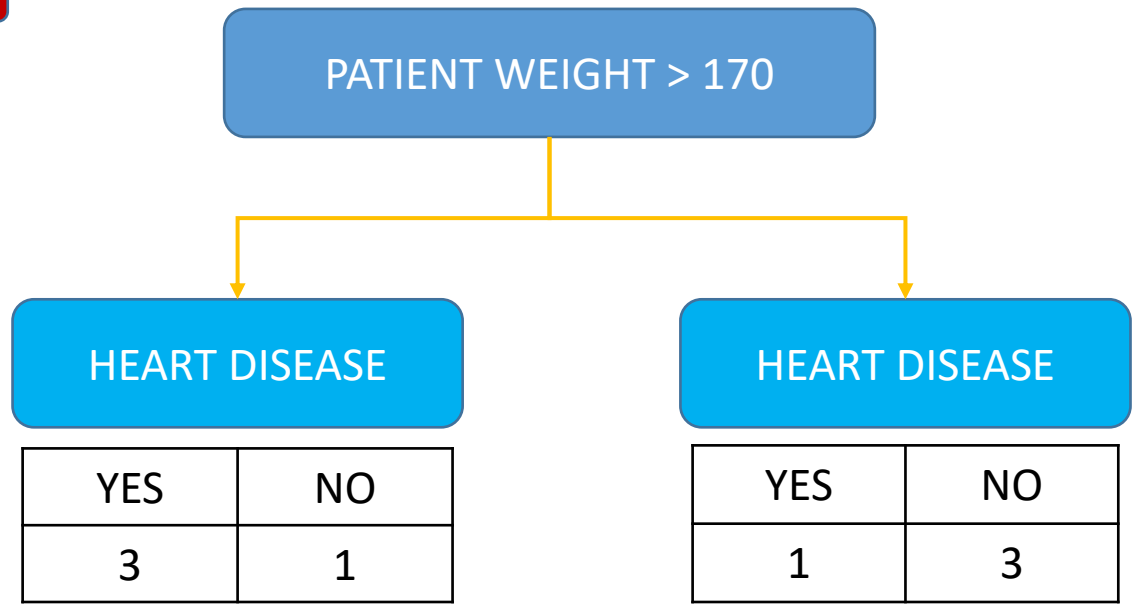
188.5

161.5

140.5

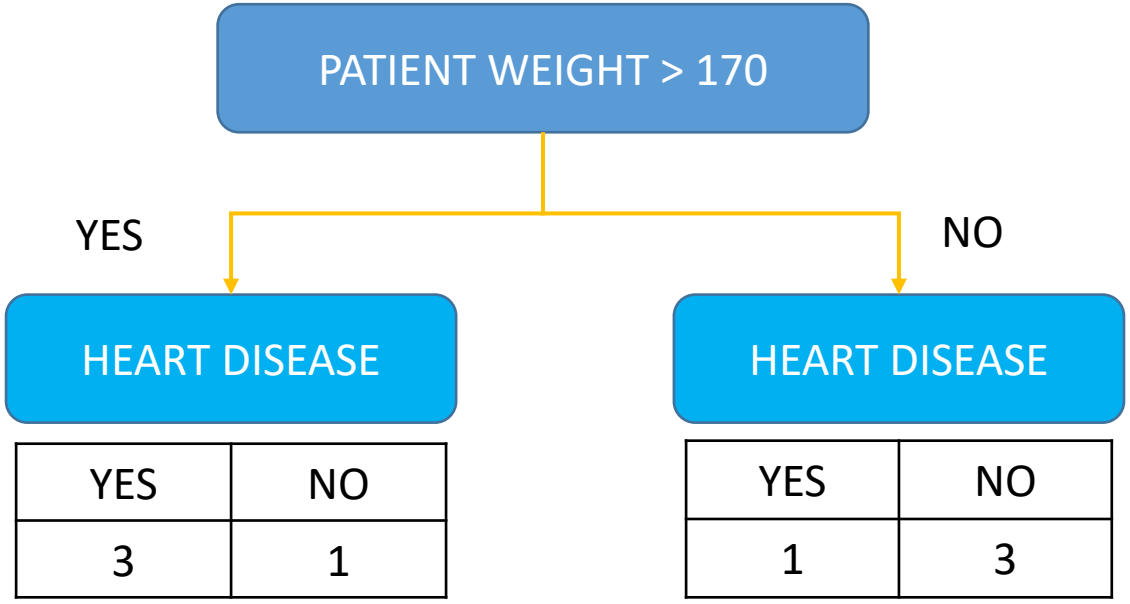
146.5

170



Amount of Say

How was this stump contribute to the final decision (classification)?



YES	NO
3	1

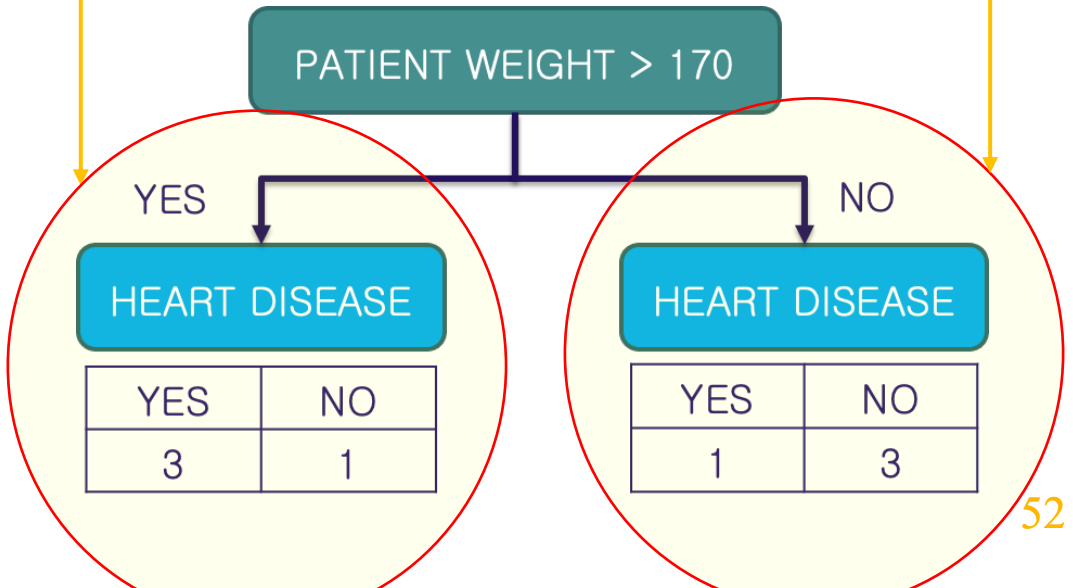
YES	NO
1	3

$$\text{Amount of Say} = \frac{1}{2}\log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

Amount of Say: Patient Weight

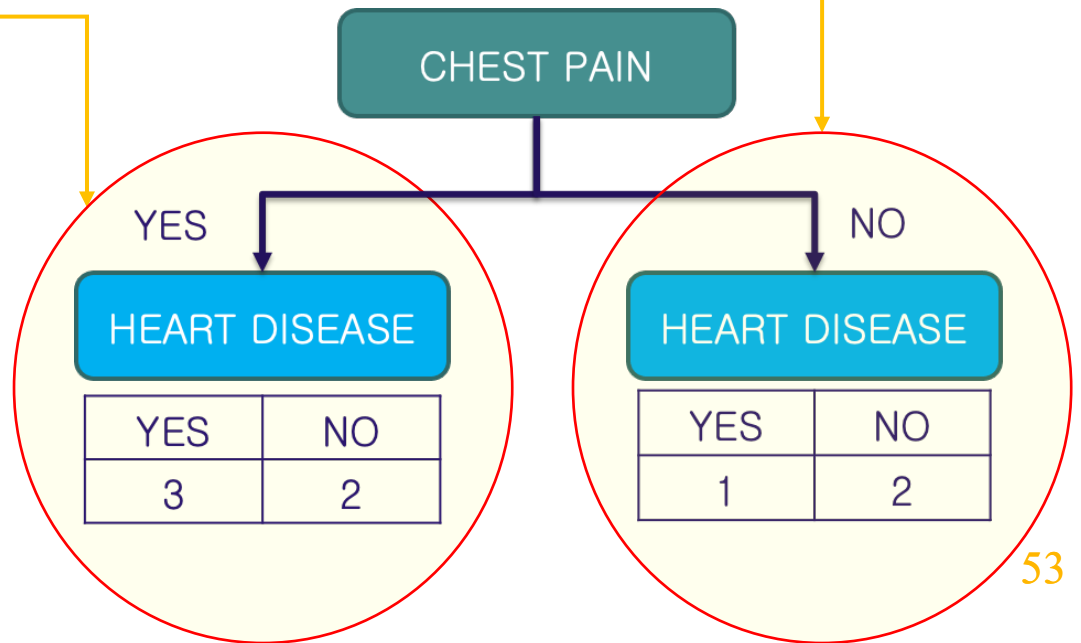
Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

- Total error is equal to the sum of the weights of the incorrect classified
- Amount of say = 1/2*log((1-2/8) / (2/8)) = 0.55



Amount of say: Chest Pain

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

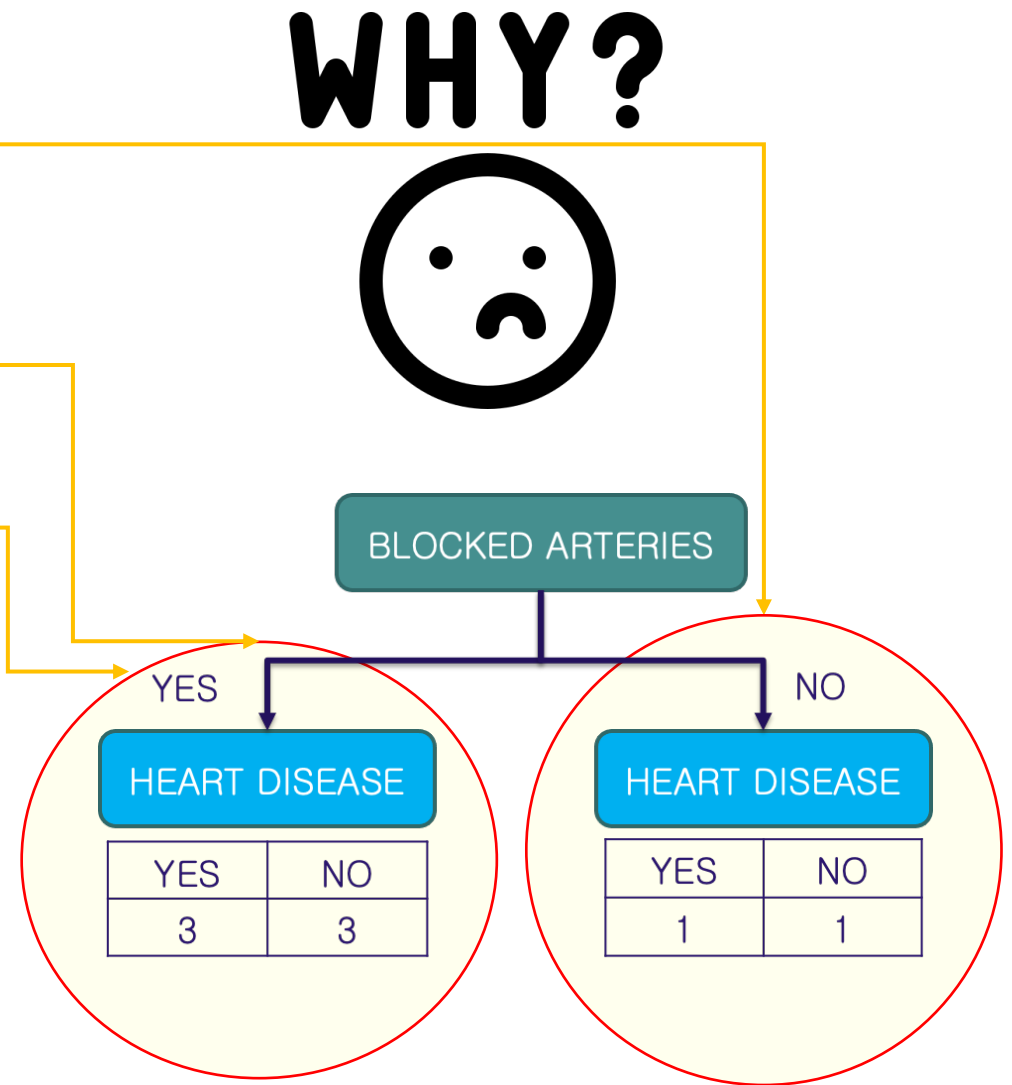


- Total error is equal to the sum of the weights of the incorrect classified
- Amount of say = 1/2*log((1-3/8) / (3/8)) = 0.25

$$\log(\text{Odds}) = \log\left(\frac{1 - \text{Probability of incorrect prediction}}{\text{Probability of incorrect prediction}}\right)$$

Amout of Say: Blocked Arteries

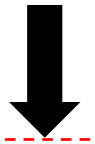
Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8



- Total error is equal to the sum of the weights of the incorrect classified
- Amount of say = 1/2*log((1-4/8) / (4/8)) = 0

Assumptions

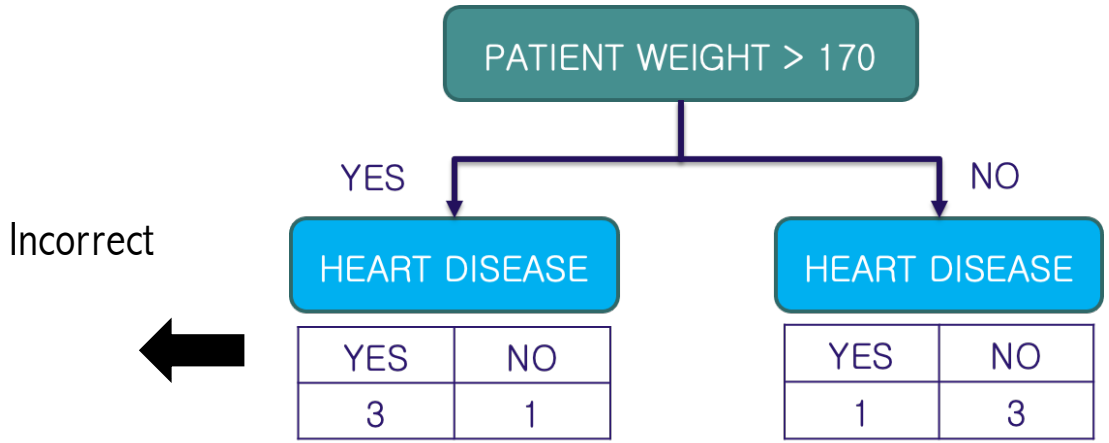
Known: weight cho các sample dự đoán sai được sử dụng để tính "Amount of Say" cho từng stump hiện tại.



Unknown: Tiếp theo, chúng ta cần làm thế nào để sử dụng thông tin các weight của sample dự đoán sai này để xây dựng stump tiếp và khắc phục các dự đoán sai này

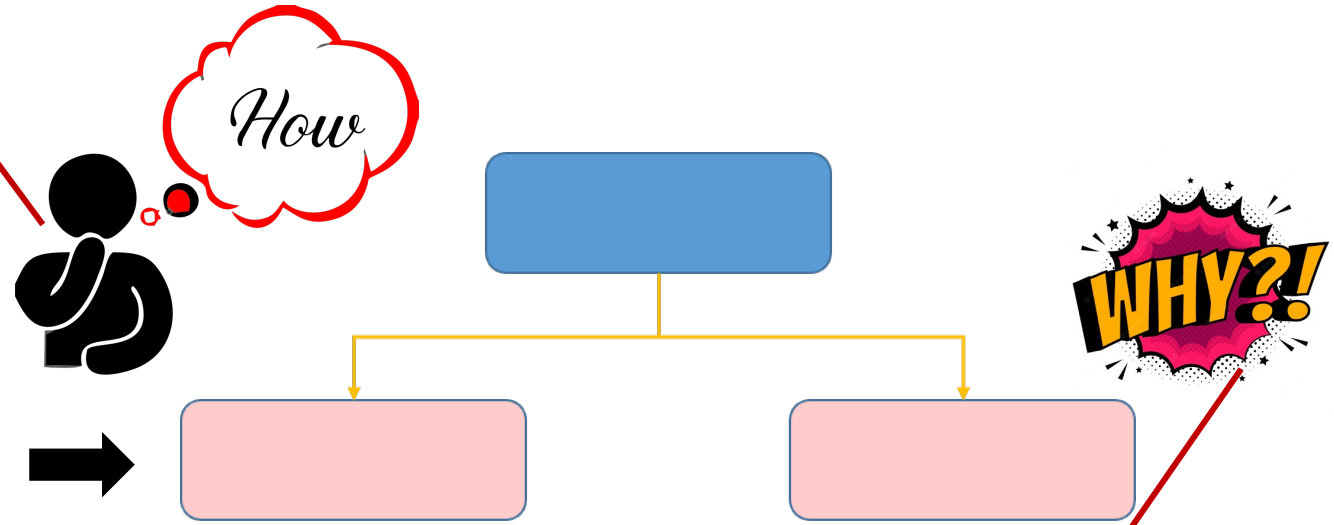
Idea: Improved Bootstrapped Dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	No	168	No
Yes	Yes	172	No



Create new dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
Yes	Yes	167	Yes
Yes	Yes	167	Yes
Yes	Yes	172	No
Yes	Yes	172	No

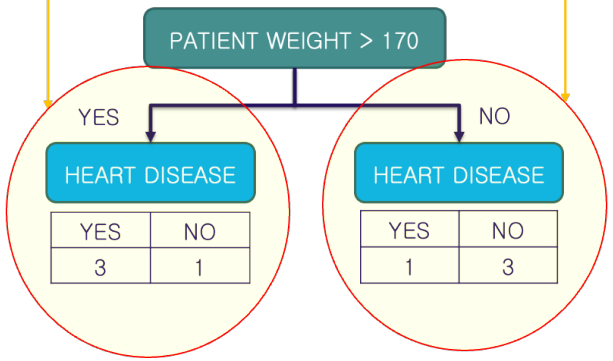


How to build next Stump

Increase the sample weights of samples that were incorrectly classified and decrease sample weights of samples that were correctly classified. Label {-1, 1}

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

- Total error is equal to the sum of the weights of the incorrect classified
- Amount of say = 1/2*log((1-2/8) / (2/8)) = 0.55



$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

New sample weight = 1/8 * e^{0.55} = 0.22

```
def update_weights_formular1(w_i, alpha, y, y_pred):
    result = w_i * np.exp(-alpha * y * y_pred)
    w_norm = result / np.sum(result)
    return w_norm
```

How to build next Stump

Increase the sample weights of samples that were incorrectly classified and decrease the sample weights of samples that were correctly classified. Label {-1, 1}

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	No	168	No
Yes	Yes	172	No

$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{-amount of say}}$$

New sample weight = 1/8 * e^{-0.55} = 0.07

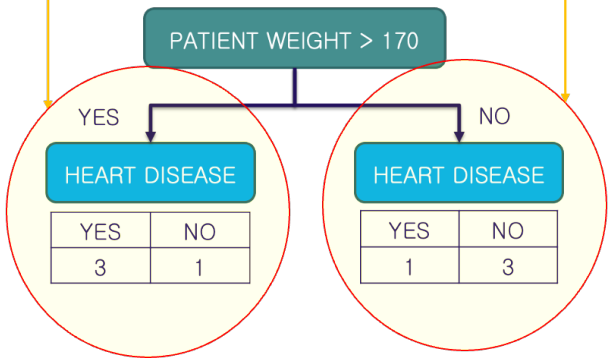
```
def update_weights_formular1(w_i, alpha, y, y_pred):
    result = w_i * np.exp(-alpha * y * y_pred)
    w_norm = result / np.sum(result)
    return w_norm
```

How to build next Stump

Increase the sample weights of samples that were incorrectly classified and keep the sample weights of samples that were correctly classified. Label {0, 1}

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

- Total error is equal to the sum of the weights of the incorrect classified
- Amount of say = 1/2*log((1-2/8) / (2/8)) = 0.55



$$w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],$$

New sample weight = 1/8 * e^{0.55*1} = 0.22

```
def update_weights_formular2(w_i, alpha, y, y_pred):
    result = w_i * np.exp(alpha * (
        np.not_equal(y, y_pred)).astype(int))
    w_norm = result / np.sum(result)
    return w_norm
```

How to build next Stump

Increase the sample weights of samples that were incorrectly classified and keep the sample weights of samples that were correctly classified. Label {0, 1}

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	No	168	No
Yes	Yes	172	No

$$w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],$$

New sample weight = 1/8 * e^{0.55*0} = 0.125

```
def update_weights_formular2(w_i, alpha, y, y_pred):
    result = w_i * np.exp(alpha * (
        np.not_equal(y, y_pred)).astype(int))
    w_norm = result / np.sum(result)
    return w_norm
```

New Sample Weight

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Sample Weight	New Weight	Normal Weight
Yes	Yes	205	Yes	1/8	0.07	0.08
No	Yes	180	Yes	1/8	0.07	0.08
Yes	No	210	Yes	1/8	0.07	0.08
Yes	Yes	167	Yes	1/8	0.22	0.25
No	Yes	156	No	1/8	0.07	0.08
No	Yes	125	No	1/8	0.07	0.08
Yes	No	168	No	1/8	0.07	0.08
Yes	Yes	172	No	1/8	0.22	0.25
Sum				~1.0	0.86	~1.0

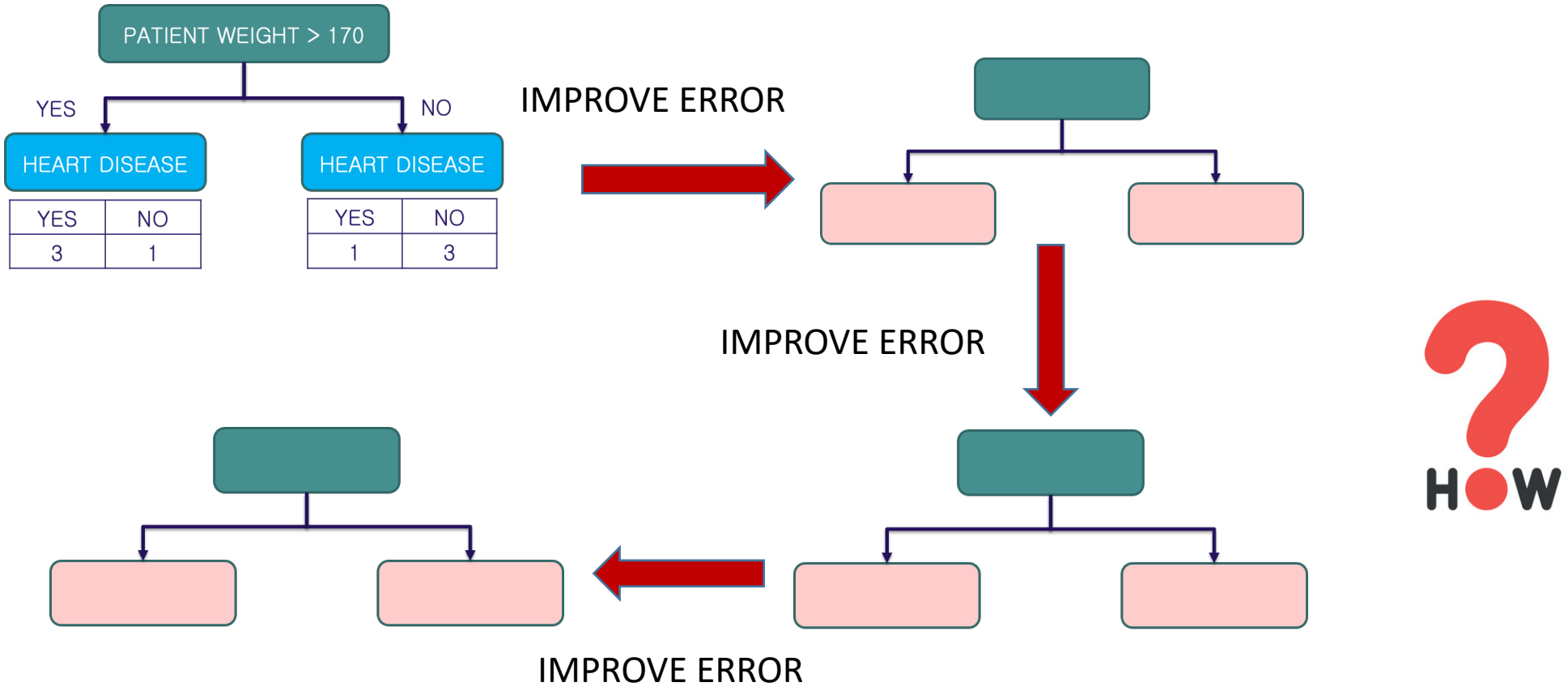
$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

Update

New Sample Weight

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	New Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

AdaBoost: FOREST OF STUMPS



New Dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

New Dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

Random [0,1] to select samples

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

New Dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight	Range
Yes	Yes	205	Yes	0.08	[0-0.08]
No	Yes	180	Yes	0.08	(0.08-0.16]
Yes	No	210	Yes	0.08	(0.16-0.24]
Yes	Yes	167	Yes	0.25	(0.24-0.495]
No	Yes	156	No	0.08	(0.495-0.575]
No	Yes	125	No	0.08	(0.575-0.655]
Yes	No	168	No	0.08	(0.655-0.735]
Yes	Yes	172	No	0.25	(0.735-1.0]
Sum				~1.0	

New Dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease	Normal Weight
Yes	Yes	205	Yes	0.08
No	Yes	180	Yes	0.08
Yes	No	210	Yes	0.08
Yes	Yes	167	Yes	0.25
No	Yes	156	No	0.08
No	Yes	125	No	0.08
Yes	No	168	No	0.08
Yes	Yes	172	No	0.25
Sum				~1.0

Old dataset

Random [0,1] to select samples

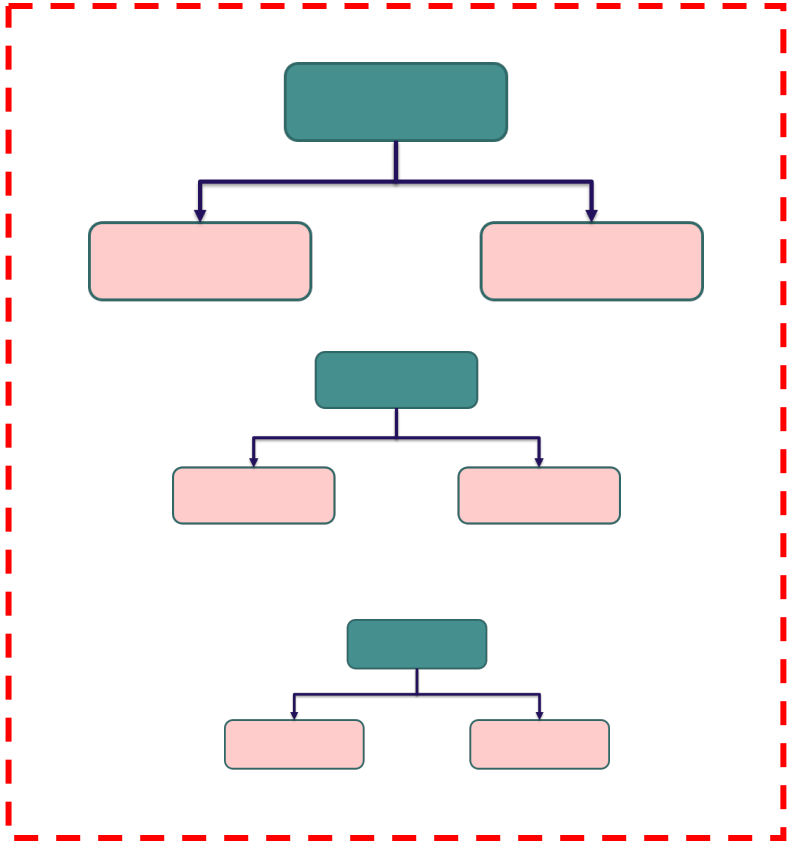
CONTINUE TO BUILD THE NEXT STUMP

Chest Pain	Blocked Arteries	Patient Weight	Heart Diease
Yes	Yes	167	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	Yes	172	No
Yes	Yes	172	No

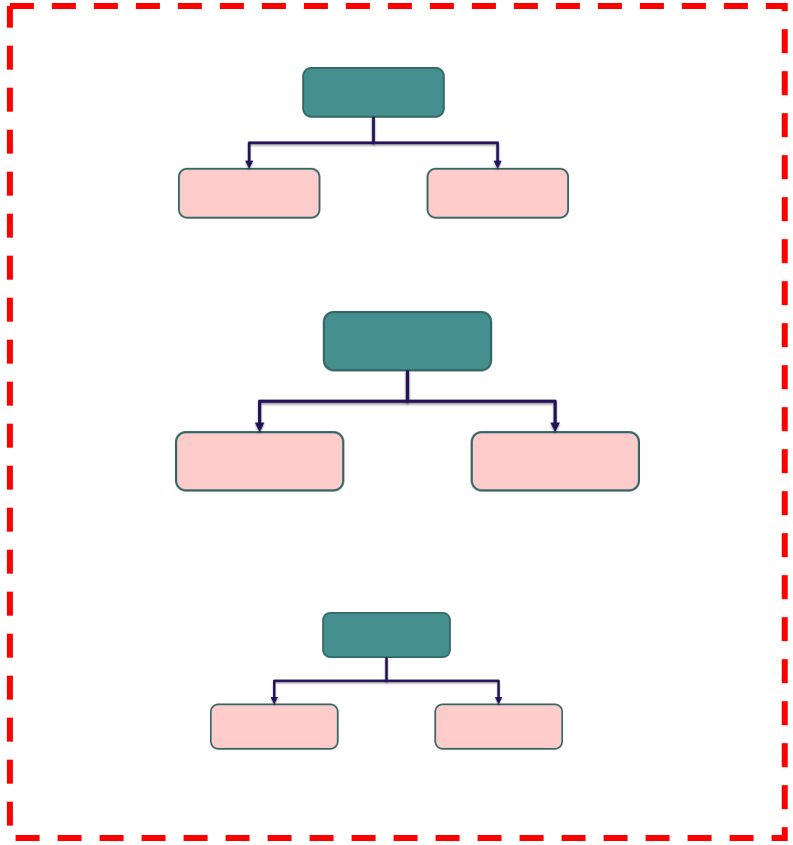
New dataset

Ý tưởng: các sample bị phân loại sai, sẽ được được nhiều hơn vào dataset mới

How to Classify The Final Result



These stumps for predicting heart disease



These stumps for predicting no heart disease

How to Classify The Final Result

Algorithm 1 *AdaBoost (Freund & Schapire 1997)*

1. *Initialize the observation weights* $w_i = 1/n,\ i = 1, 2, \ldots, n.$
2. *For* $m = 1$ *to* M*:*
   (a) *Fit a classifier* $T^{(m)}(\boldsymbol{x})$ *to the training data using weights* w_i.
   (b) *Compute*
   $$err^{(m)} = \sum_{i=1}^{n} w_i \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right) / \sum_{i=1}^{n} w_i.$$
   (c) *Compute*
   $$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}}.$$
   (d) *Set*
   $$w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \cdot \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right)\right),\ i = 1, 2, \ldots, n.$$
   (e) *Re-normalize* w_i.
3. *Output*
   $$C(\boldsymbol{x}) = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathbb{I}(T^{(m)}(\boldsymbol{x}) = k).$$

https://hastie.su.domains/Papers/samme.pdf

Adaboost: the **weight of each sample is** modified during each iteration to reduce the prediction error, and the **weight of each tree** is different when making final classification.

1. Can AdaBoost handle overfitting?
2. AdaBoost can be sensitive to outliers / label noise?
3. When should we stop the Adaboost?

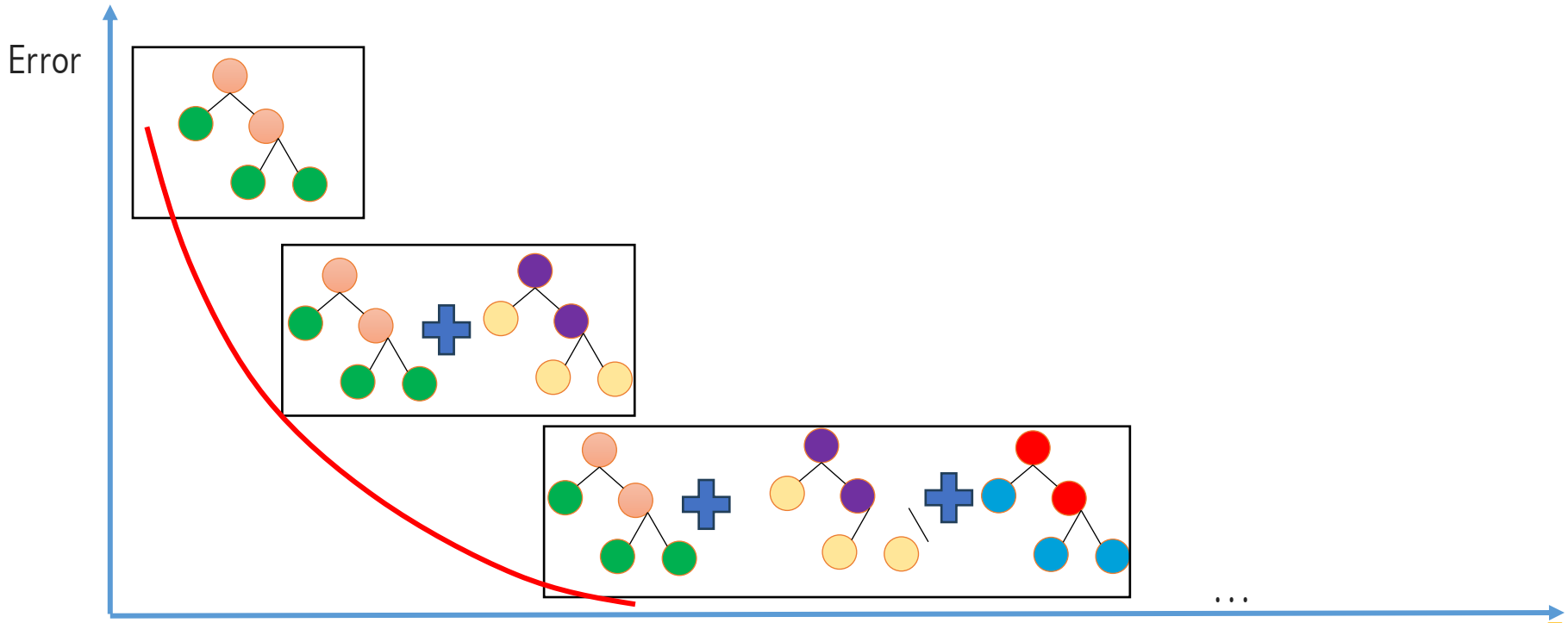
Outline

- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

Gradient Boosting

Gradient + Boosting

Applies the concepts of **logistic regression**. It uses **log-odds** to make a prediction, converts log-odds to **probabilities** through logistic function, then make a classification based on self-defined **threshold**.



Gradient Boost For Regression

Height	Favorite Color	Gender	Weight
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

Input

Output

Tree-based Gradient Boost

- Step 1: Build 1st tree
  - Calculate the average of weights

Height	Favorite Color	Gender	Weight
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

Node of 1st Tree Average of Weights: 71.17

Gradient Boost: Behind The Scenes

- Initialize a model with a constant value:
  - $F_{0(x)} = argmin \sum_{i=1}^{n} L(y, \delta)$

δ

y

Height	Favorite Color	Gender	Weight
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56

SSR = 1/2 {(88 – δ)^2 + (76 – δ)^2 + (56 - δ)^2}

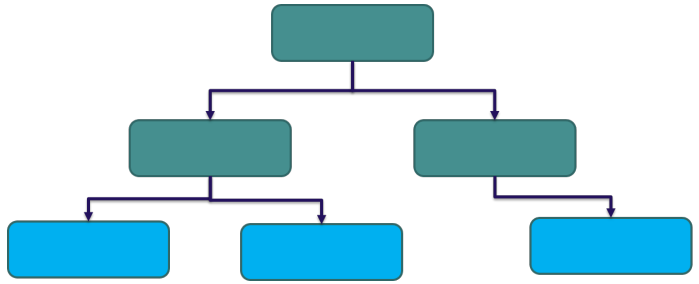
$\frac{dSSR}{d\delta}$ = -(88 - δ) – (76 - δ) – (56 - δ) = 0

$\delta = \frac{88+76+56}{3}$ = 73.3 = average of all sample' weights

Tree-based Gradient Boost

- Step 2:
  - Build 2nd tree

Average of weights: 71.17

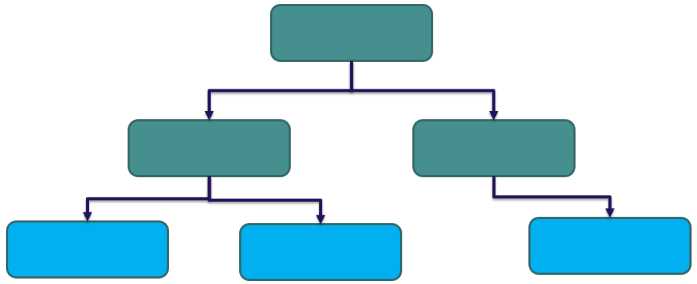


Height	Favorite Color	Gender	Weight
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

Tree-based Gradient Boost

- Step 2:
  - Build 2nd tree

Average of weights: 71.2

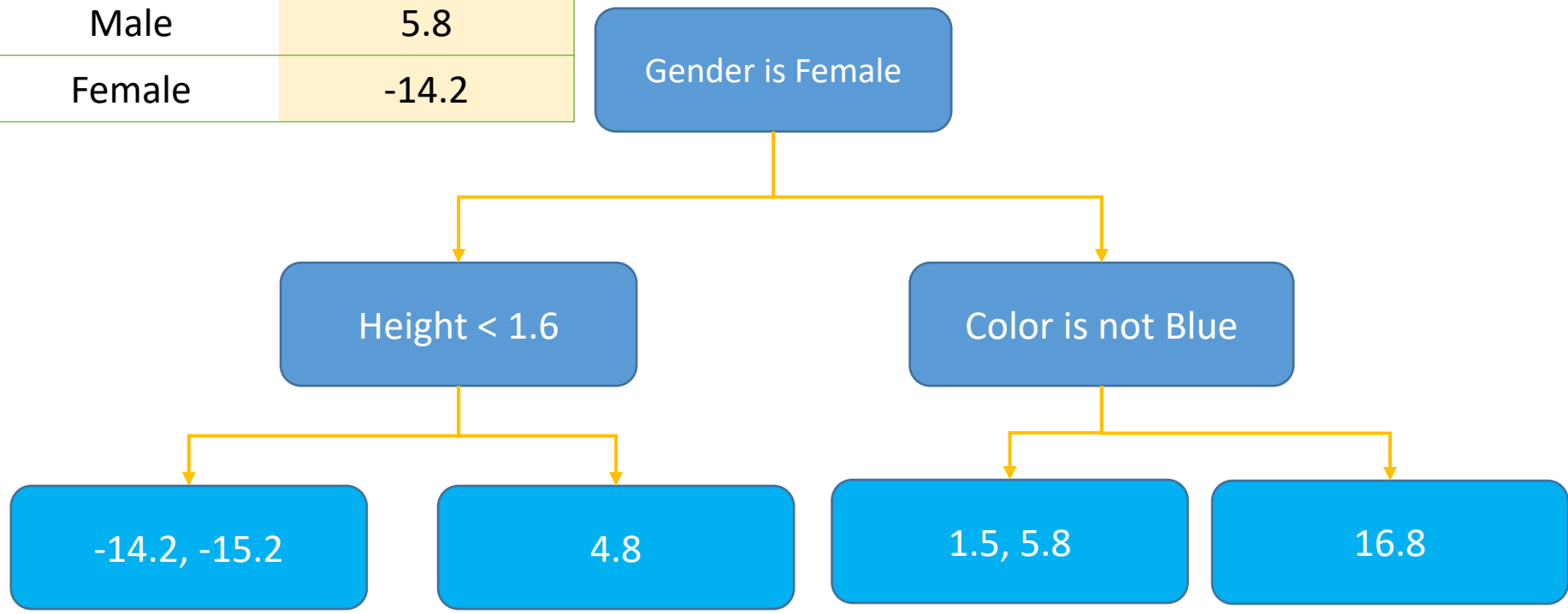


Height	Favorite Color	Gender	Weight	Residual Error
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	1.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

Tree-based Gradient Boost

Height	Favorite Color	Gender	Residual Error
1.6	Blue	Male	16.8
1.6	Green	Female	1.8
1.5	Blue	Female	-15.2
1.8	Red	Male	1.8
1.5	Green	Male	5.8
1.4	Blue	Female	-14.2

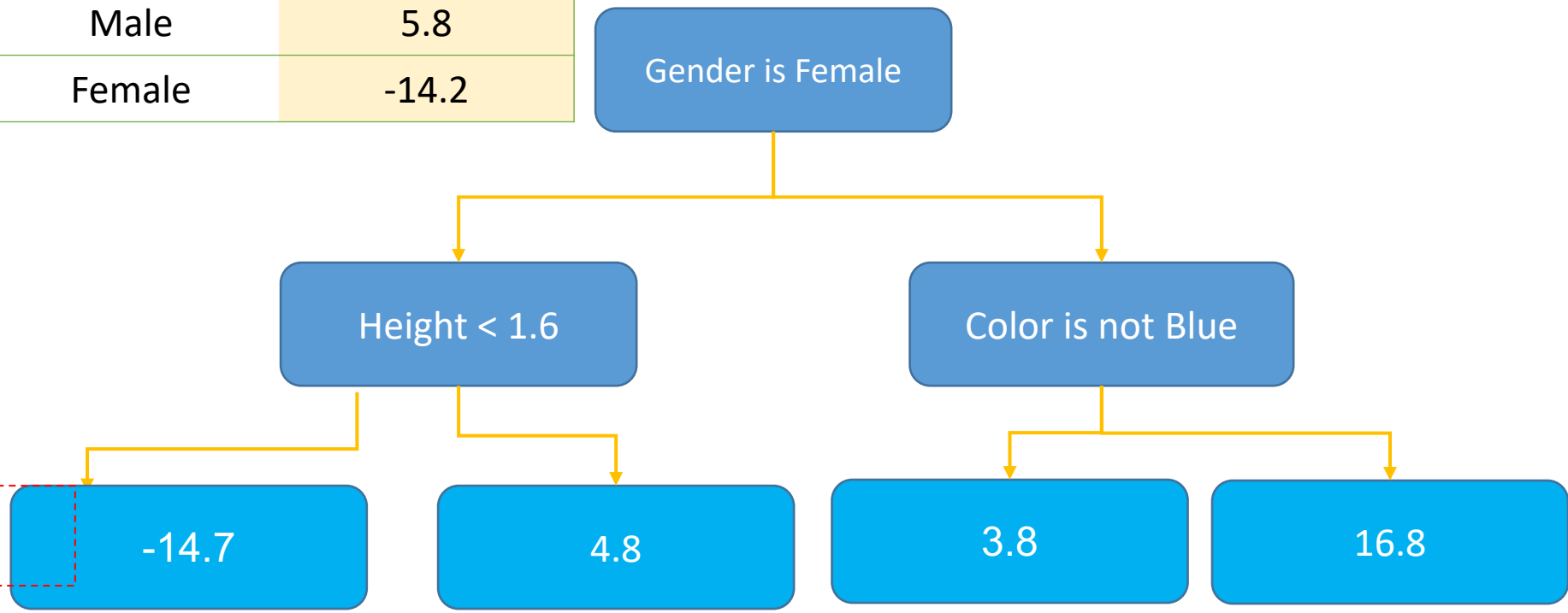
Tại sao lại xây dựng cây dự đoán Residual Error



Tree-based Gradient Boost

Height	Favorite Color	Gender	Residual Error
1.6	Blue	Male	16.8
1.6	Green	Female	1.8
1.5	Blue	Female	-15.2
1.8	Red	Male	1.8
1.5	Green	Male	5.8
1.4	Blue	Female	-14.2

Tại sao lại xây dựng cây dự đoán Residual Error



Trung bình residual

Tree-based Gradient Boost

Điều gì sẽ xảy ra, nếu chung ta tiếp tục xây dựng cây theo cách này?

1st Tree

Average of weights: 71.2

2nd Tree

Gender is Female

Height < 1.6

Color is not Blue

-14.7

4.8

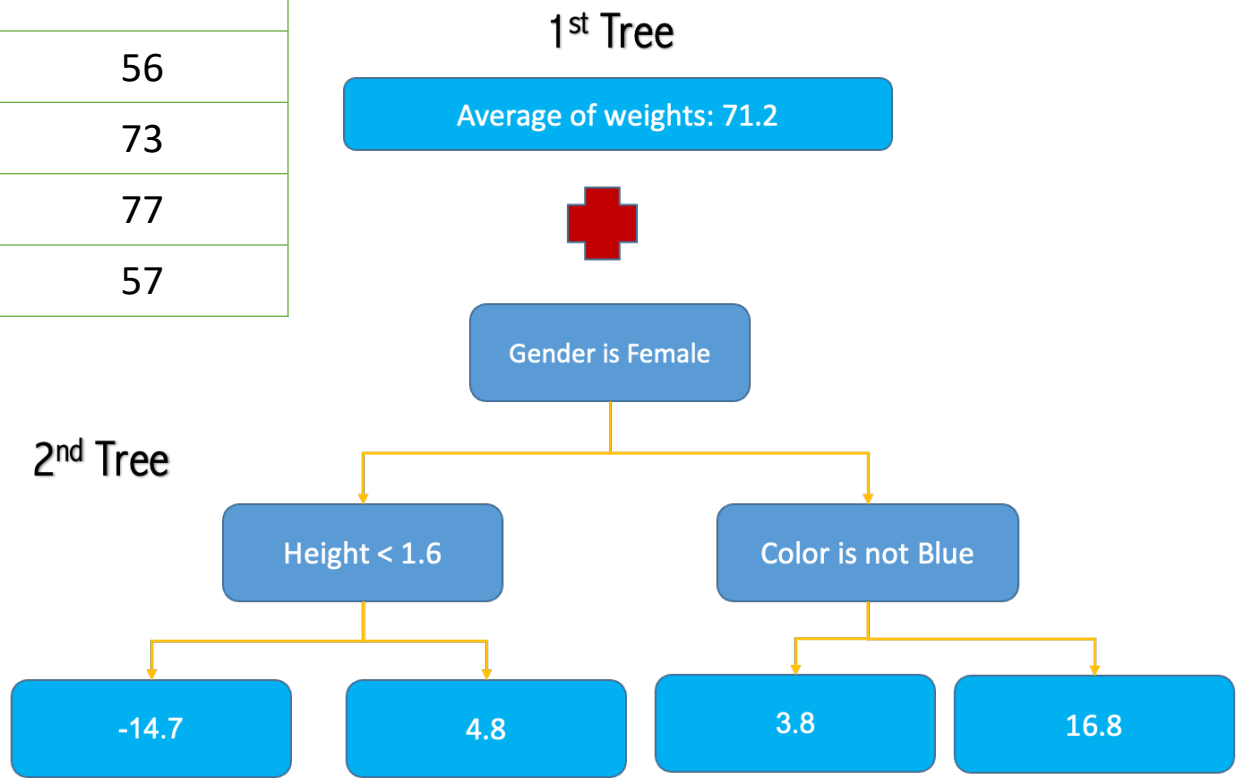
3.8

16.8

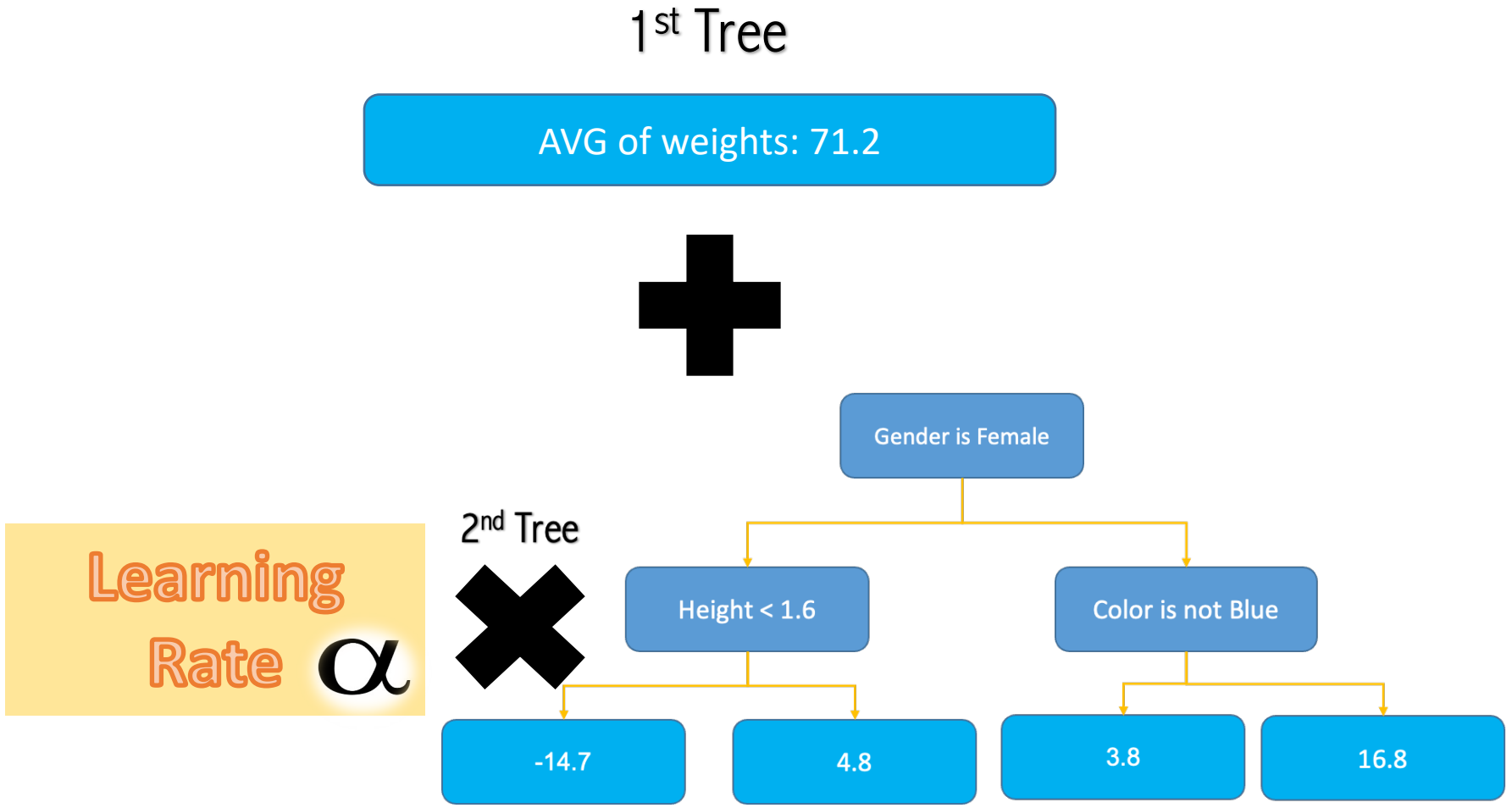
Prediction

Height	Favorite Color	Gender	Weight	Prediction
1.6	Blue	Male	88	88
1.6	Green	Female	76	76
1.5	Blue	Female	56	56
1.8	Red	Male	73	73
1.5	Green	Male	77	77
1.4	Blue	Female	57	57

OVER FITTING



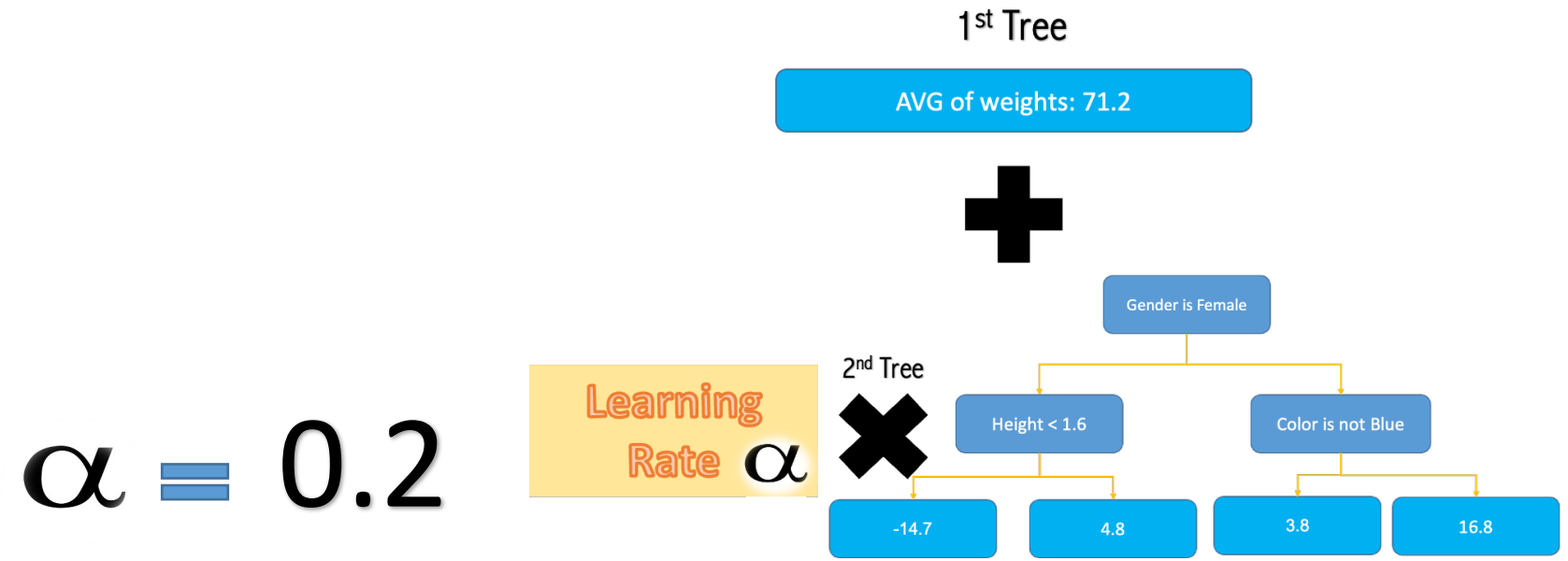
Prediction


1st Tree
AVG of weights: 71.2
Learning Rate α
2nd Tree
Gender is Female
Height < 1.6
Color is not Blue
-14.7
4.8
3.8
16.8

Prediction

Height	Favorite Color	Gender	Weight	Prediction
1.6	Blue	Male	88	74.56

Prediction = 71.2 + 0.2 * 16.8 = 74.56

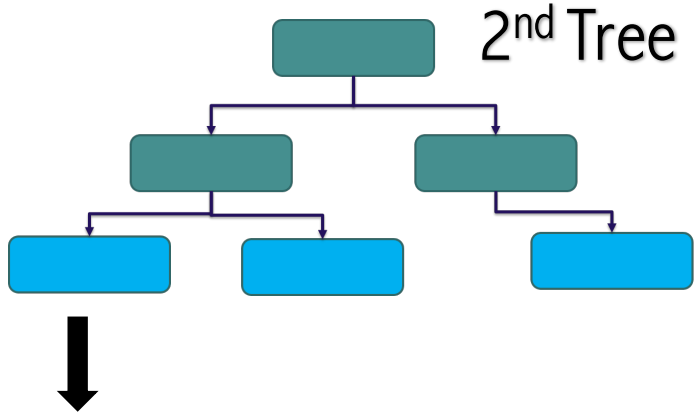


Tree-based Gradient Boost

- Step 3:
  - Build 3rd tree

1st Tree

Average of weights: 71.2



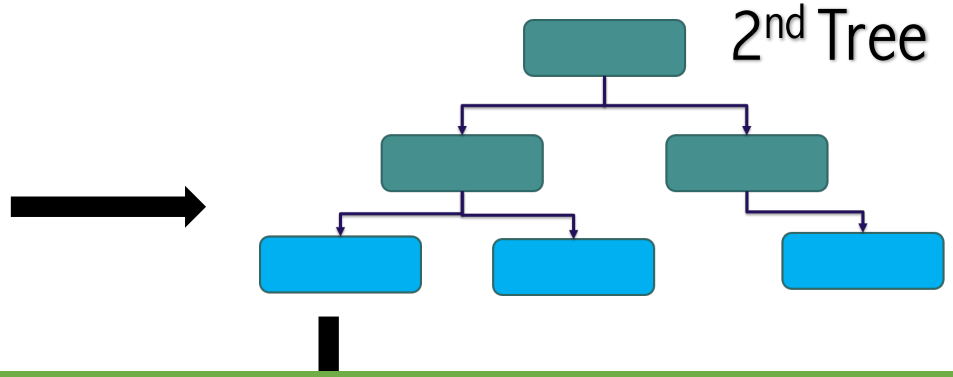
Height	Favorite Color	Gender	Weight	Predicted Weight	Residual Error
1.6	Blue	Male	88	74.56	12.44
1.6	Green	Female	76	…	…
1.5	Blue	Female	56	…	…
1.8	Red	Male	73	…	…
1.5	Green	Male	77	…	…
1.4	Blue	Female	57	…	…

Tree-based Gradient Boost

- Step 3:
  - Build 3rd tree

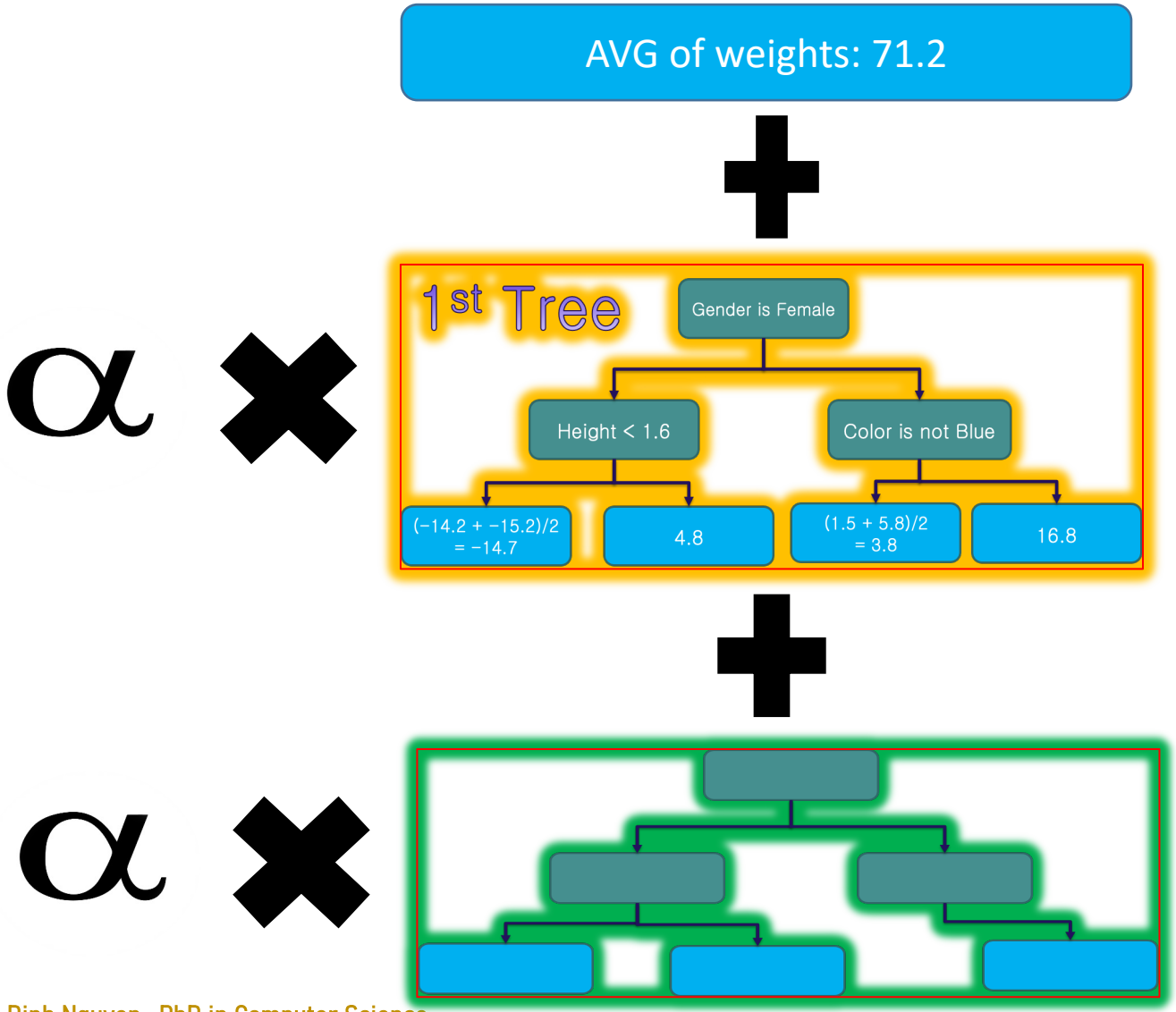
1st Tree

Average of weights: 71.2



Height	Favorite Color	Gender	First Tree Residual	Second Tree Residual	Third Tree Residual
1.6	Blue	Male	16.8	12.44	
1.6	Green	Female	1.8	…	
1.5	Blue	Female	-15.2	…	
1.8	Red	Male	1.8	…	
1.5	Green	Male	5.8	…	
1.4	Blue	Female	-14.2	…	

Prediction



1. How to select α ($learning\ rate$) ?

2. Differences between Gradient Descent and Gradient Boosting

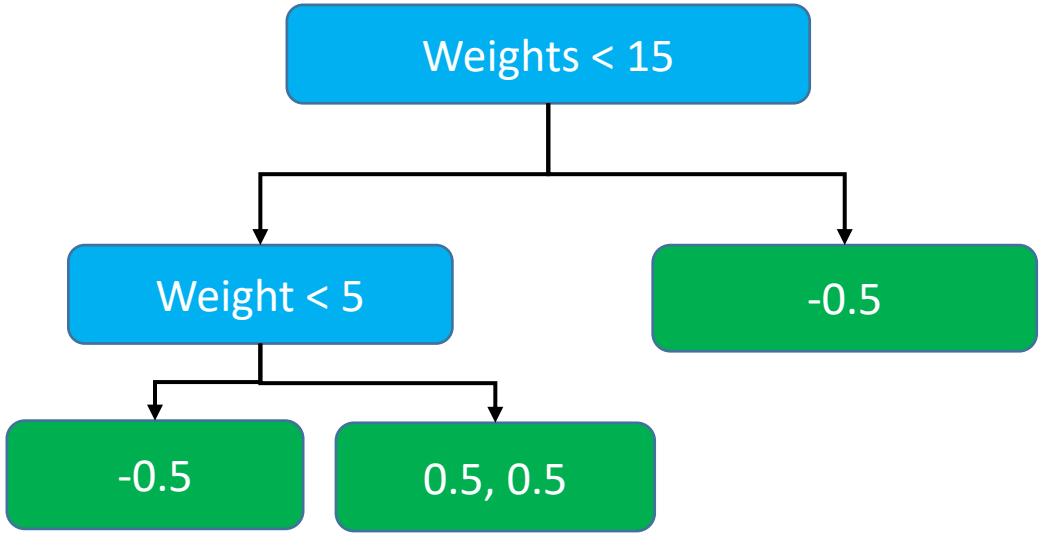
3. Limitations of Gradient Boosting?

Outline

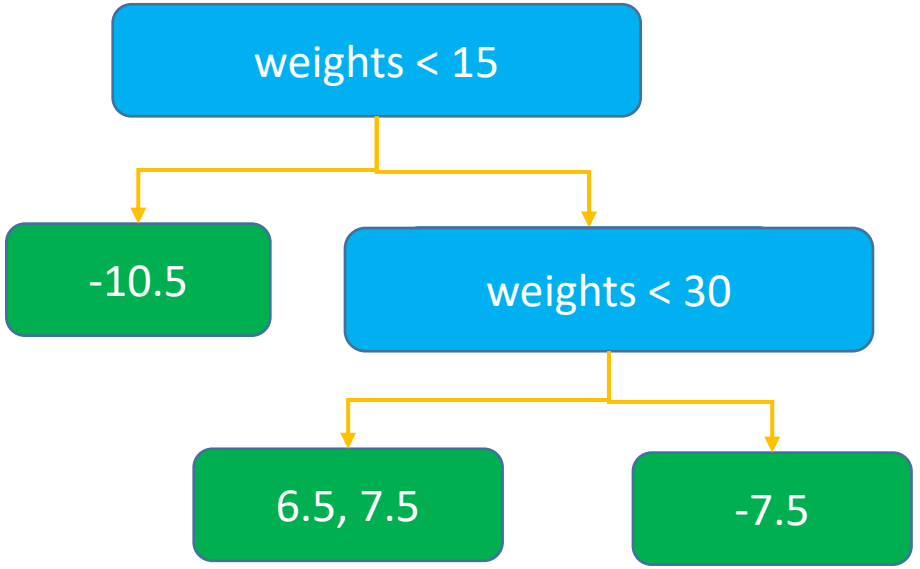
- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost

XGBoost

Classification



Regression



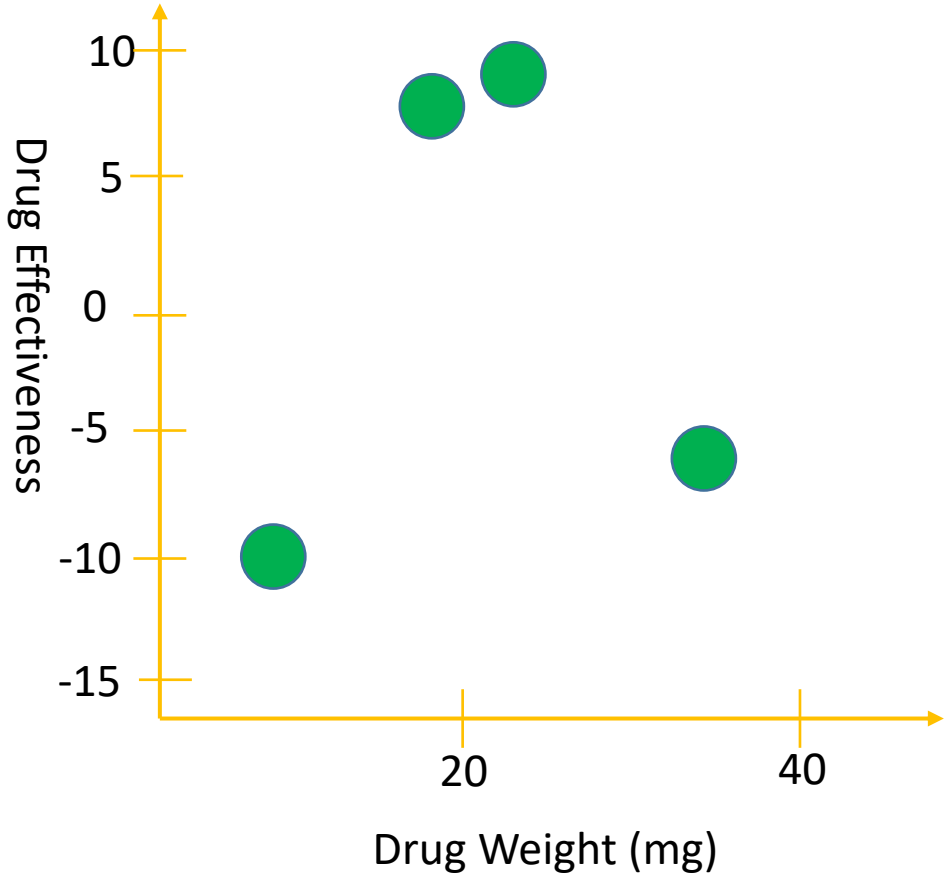
$$Similarity\ Score\ = \frac{(\sum \text{Residual})^2}{\sum \bar{y}_i \times (1-\bar{y}_i) + \lambda}$$

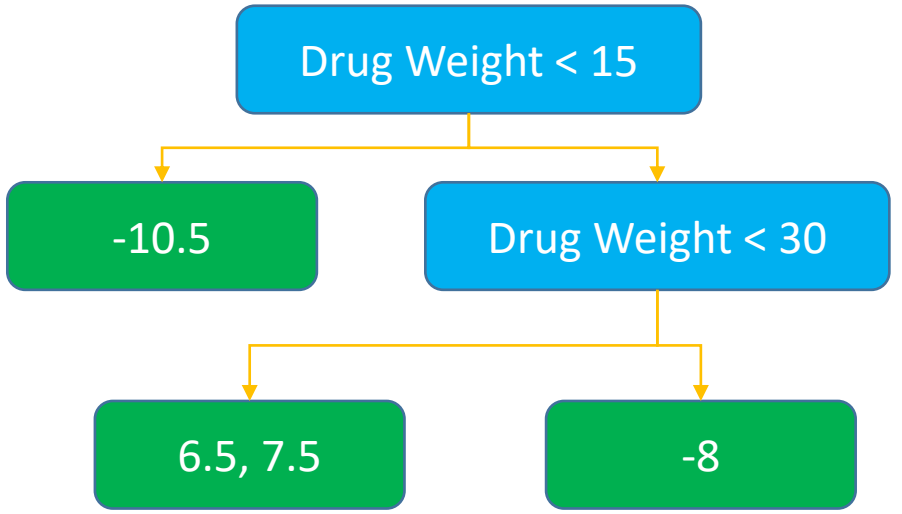
$$Ouput\ value\ = \frac{(\sum \text{Residual})}{\sum \bar{y}_i \times (1-\bar{y}_i) + \lambda}$$

$$Similarity\ Score\ = \frac{(\sum \text{Residual})^2}{\text{Number of Residual} + \lambda}$$

$$\text{Output Value} = \frac{(\sum \text{Residual})}{\text{Number of Residual} + \lambda}$$

XGBoost For Regression

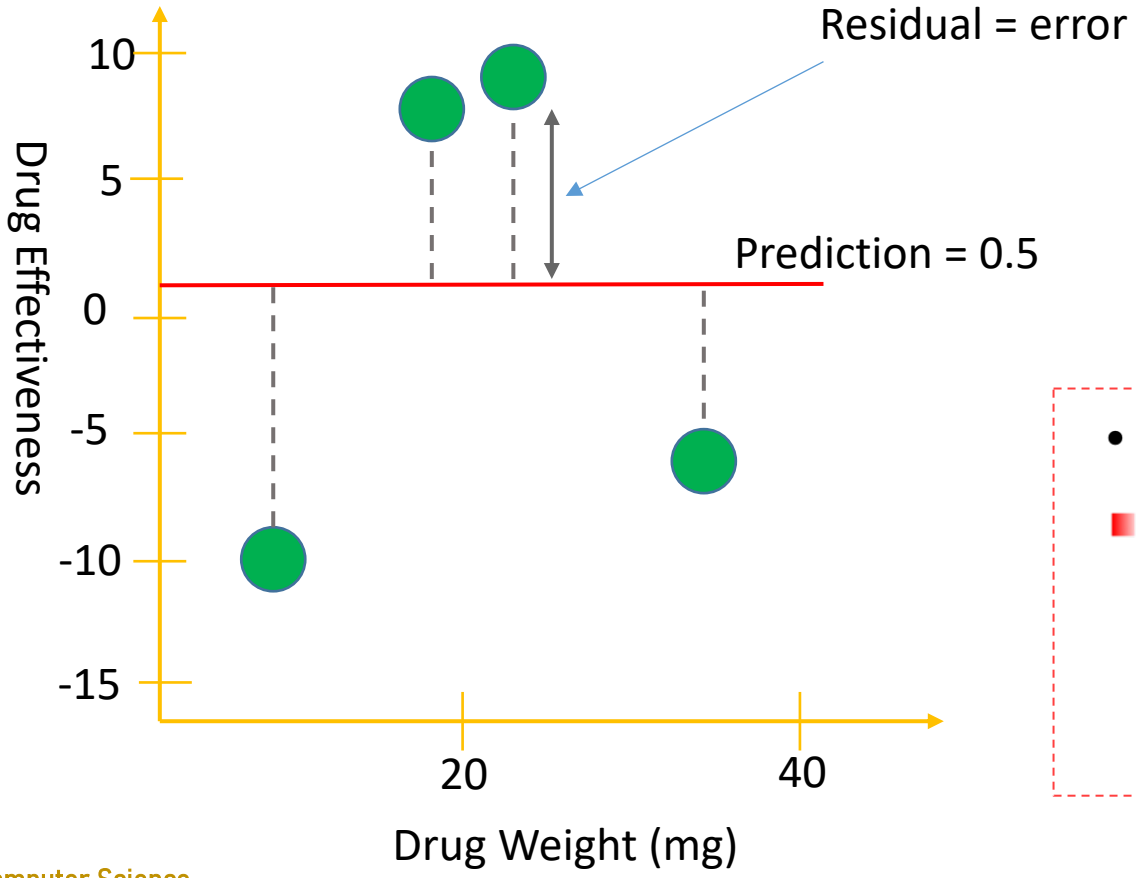




XGBoost For Regression

Step 1

- Initialize the first prediction for drug effectiveness
- Any number, for default, we set 1st prediction = 0.5



$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$$

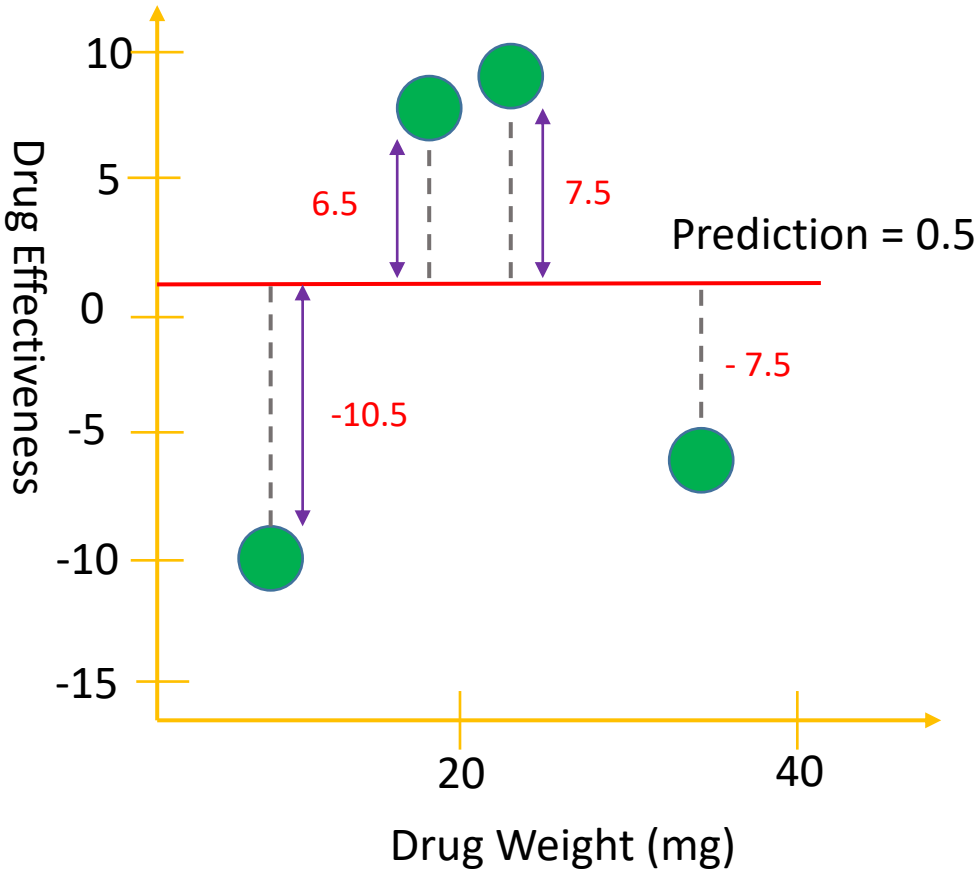
- $\{(x_i, y_i)\}^n_1$
- Loss function = $L(y_i, F(x)) = 1/2 * (\text{Output} - \text{Predicted})^2$

$$\frac{dL}{dPredicted} = 2/2\ (\text{Output} - \text{Predicted}) * -1 = -(\text{Output} - \text{Predicted})$$

Tricky implementation here

XGBoost For Regression

Step 1



Start with single Leaf of residuals

-10.5, 6.5, 7.5, -7.5

Compute Similarity Score

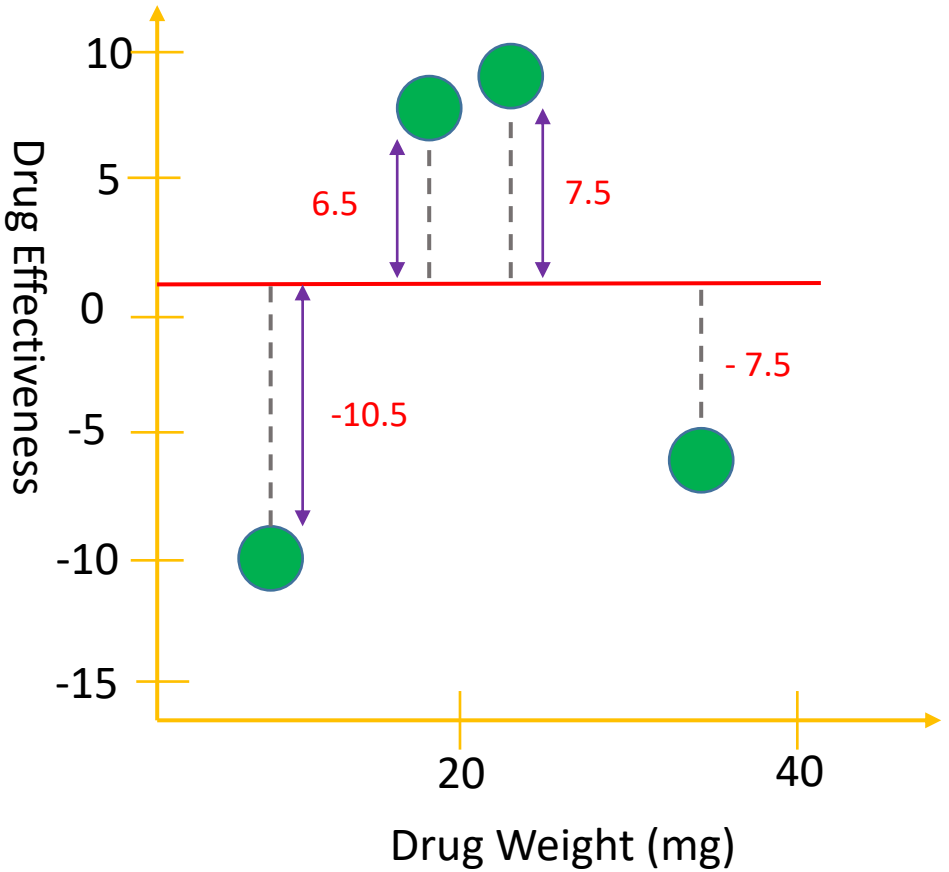
$$SC = \frac{[\sum (output - predicted)]^2}{m + \lambda}$$

m: number of samples

λ : *regularization parameters*

XGBoost For Regression

Step 1



Start with single Leaf of residuals

-10.5, 6.5, 7.5, -7.5

m = 4
$\lambda = 0$

Compute Similarity Score

$$SC = \frac{[\sum (output - predicted)]^2}{m + \lambda}$$

$$SC = \frac{[-10.5+7.5+6.5+(-7.5)]^2}{4} = 4$$

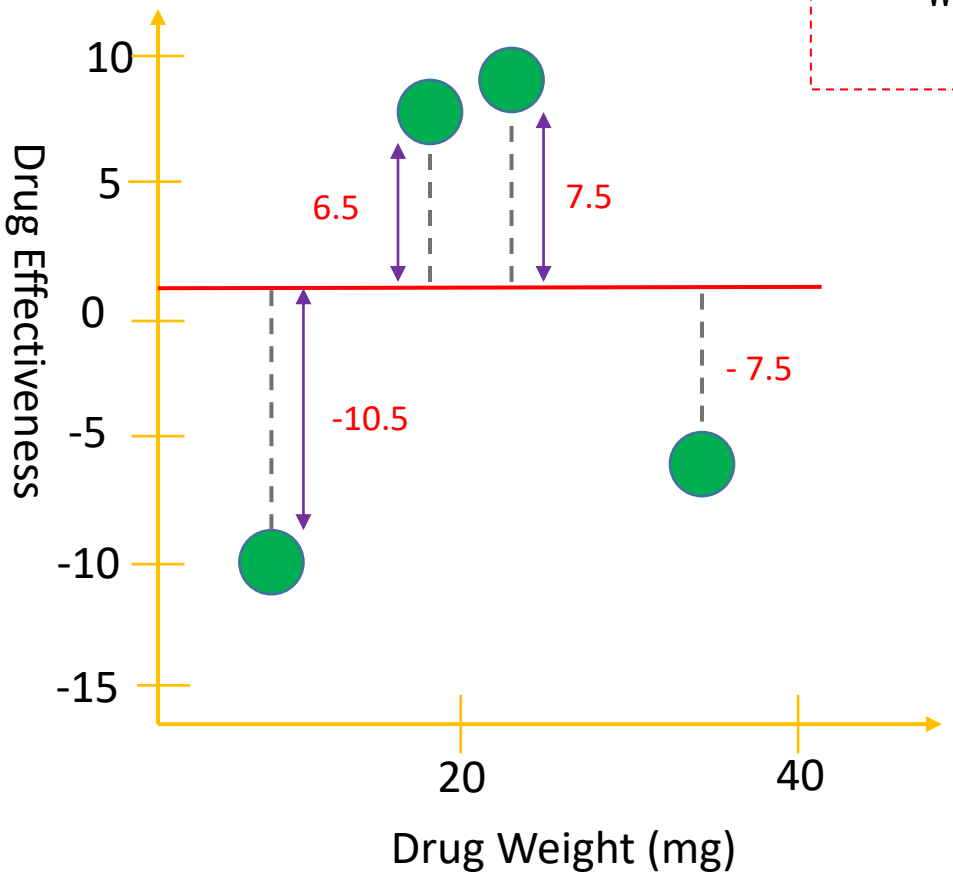
XGBoost For Regression

Step 1

SC = 4

-10.5, 6.5, 7.5, -7.5

What happens if we try to split residuals into two groups => measure the similarity score



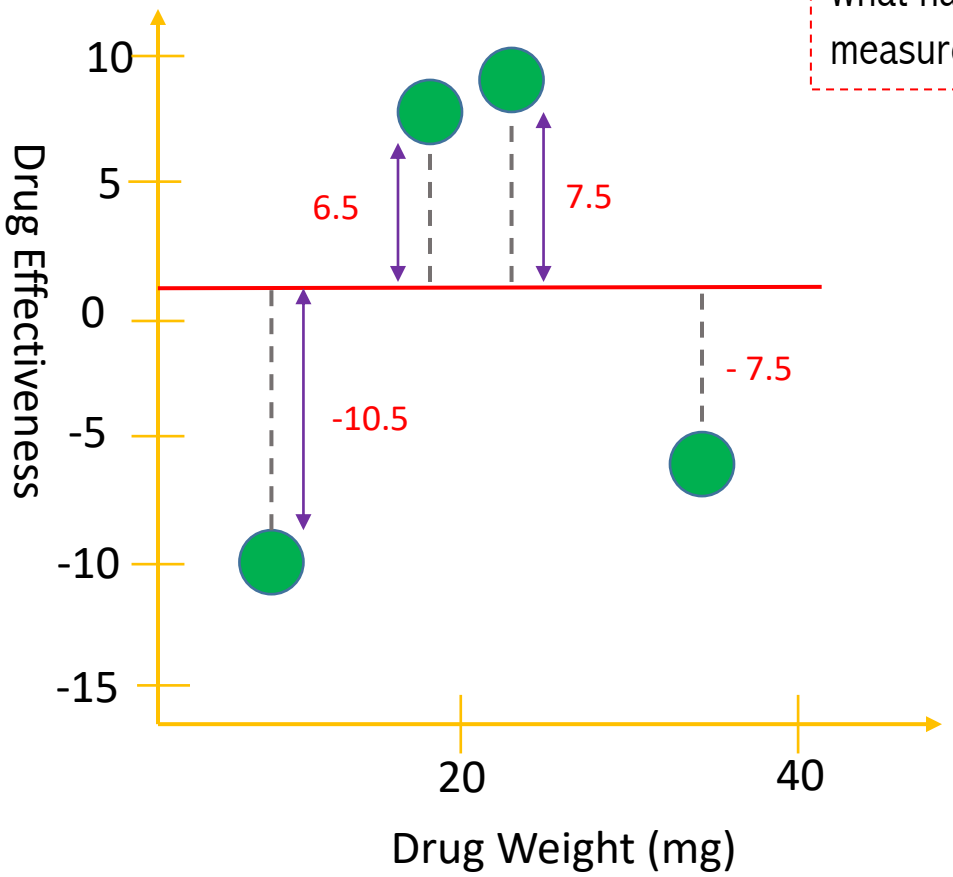
XGBoost For Regression

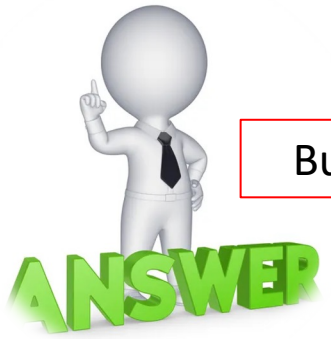
Step 1

SC = 4

-10.5, 6.5, 7.5, -7.5

What happens if we try to split residuals into two groups => measure the similarity score

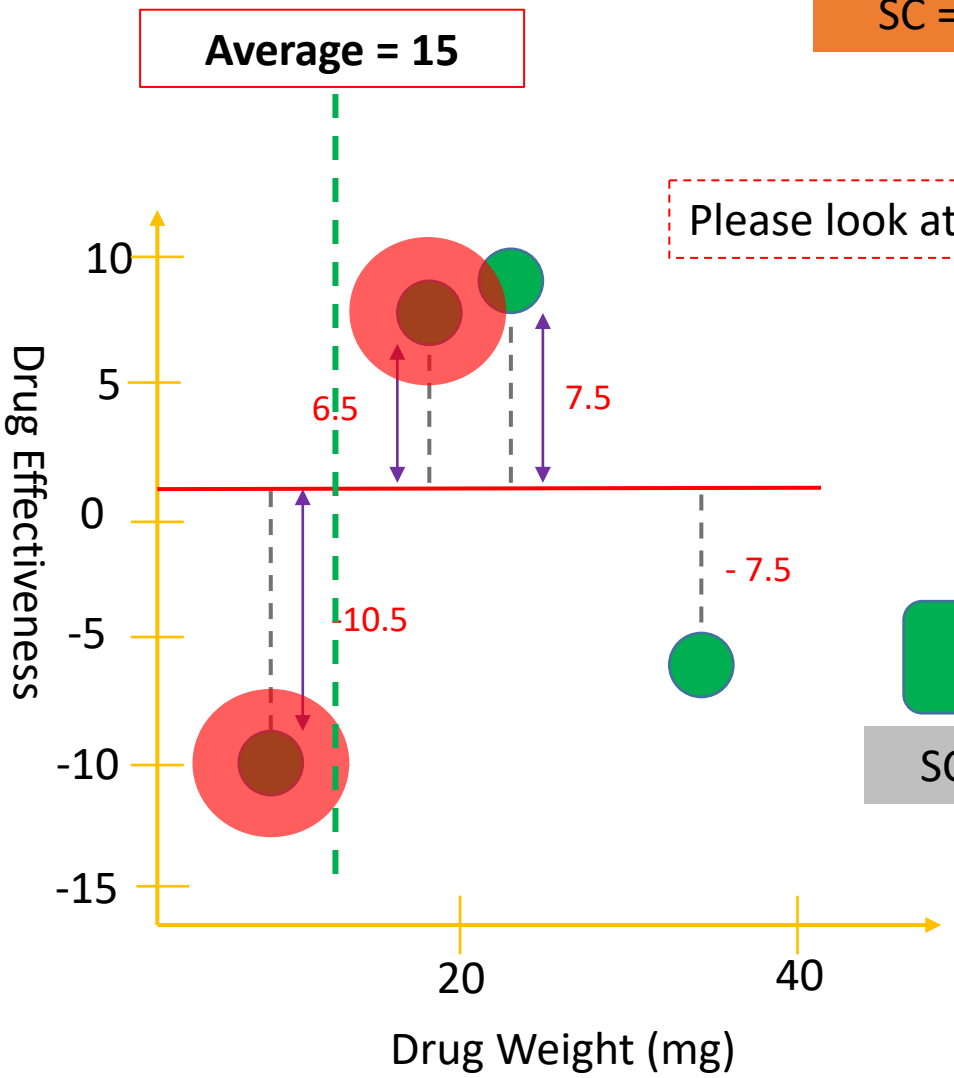




Build a tree on it

XGBoost For Regression

Step 1



SC = 4

-10.5, 6.5, 7.5, -7.5

Please look at the two outputs with lowest drug weights

Drug weights < 15

SC = 4

-10.5

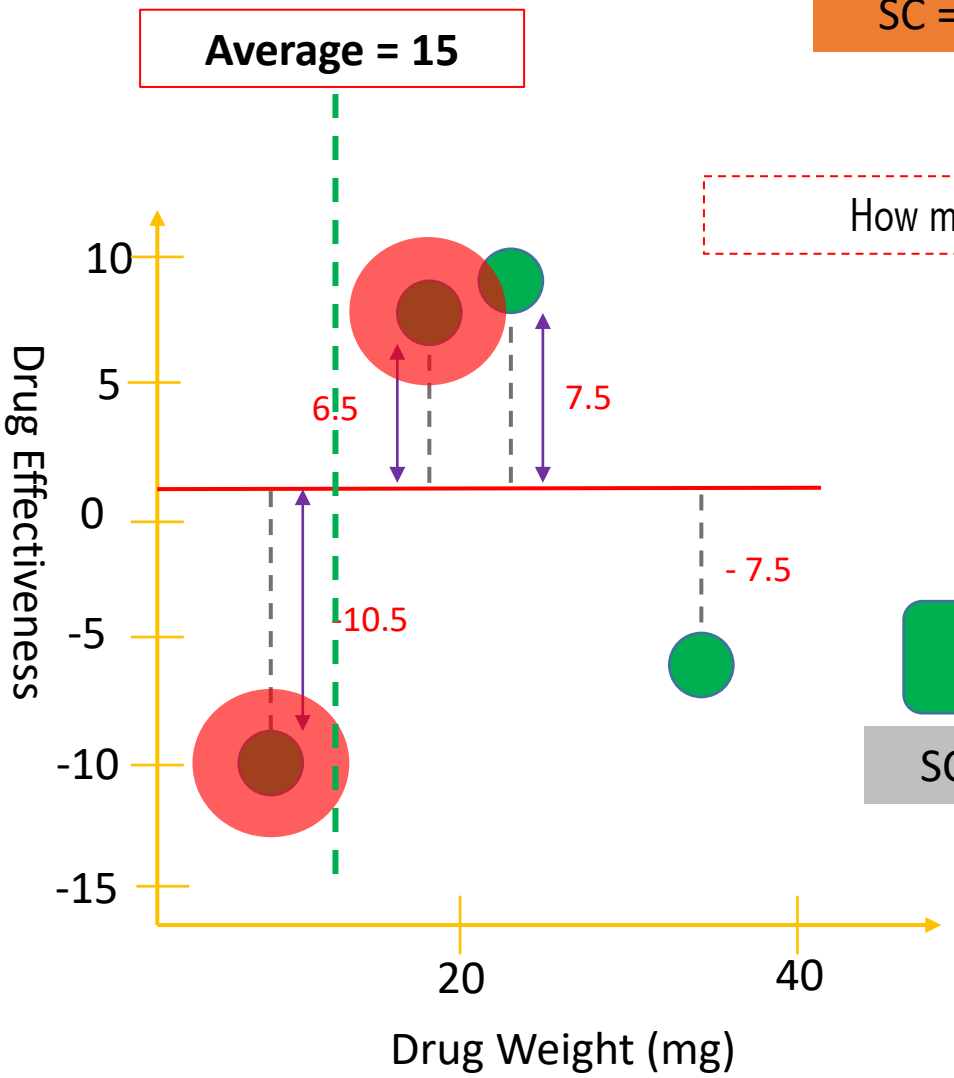
SC = 110.25

6.5, 7.5, -7.5

SC = 14.08

XGBoost For Regression

Step 1



SC = 4

-10.5, 6.5, 7.5, -7.5

How much better leaves cluster similar Residual than the root?

Drug weights < 15

SC = 4

-10.5

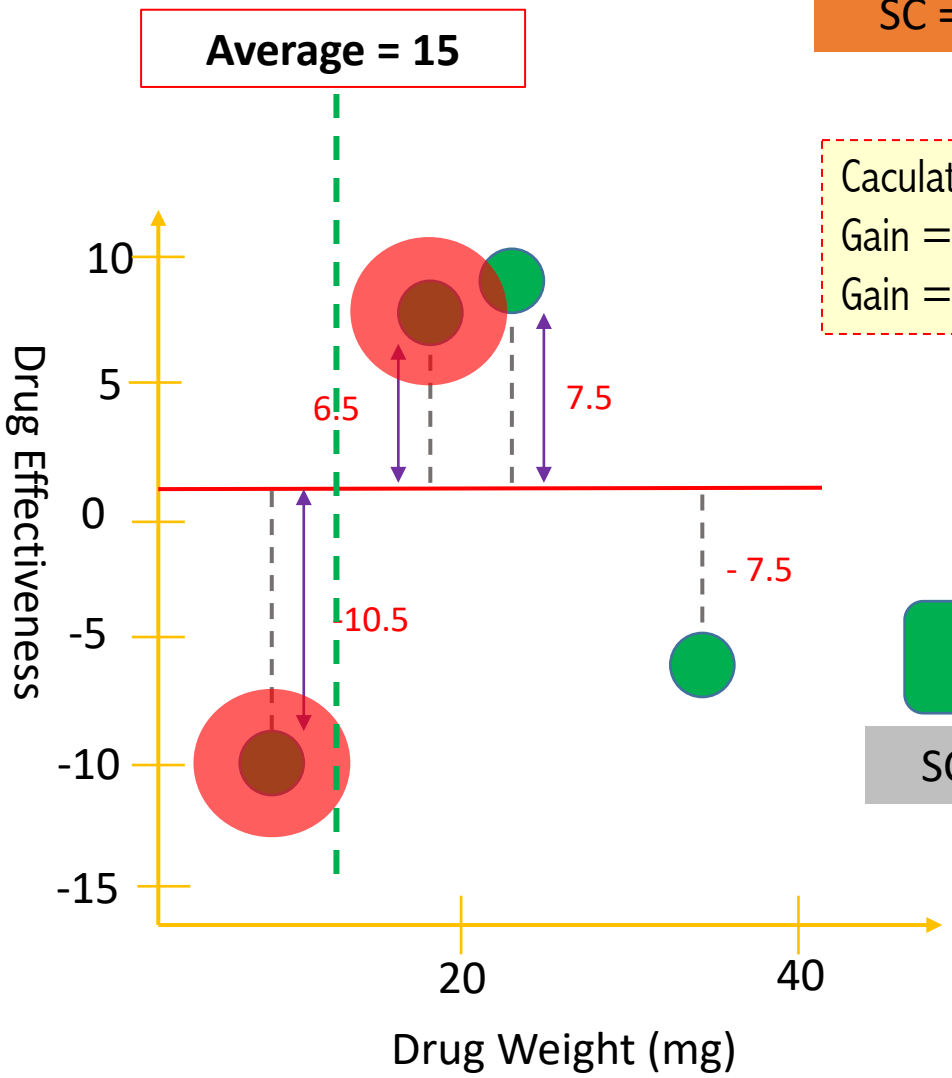
SC = 110.25

6.5, 7.5, -7.5

SC = 14.08

XGBoost For Regression

❑ *Step 1*



SC = 4

-10.5, 6.5, 7.5, -7.5

Residual rất khác nhau, triệt tiêu lẫn nhau, nên SC nhỏ

Caculate the Gain.
Gain = Left SC + Right SC - Root SC
Gain = 120.33

Drug weights < 15

SC = 4

-10.5

SC = 110.25

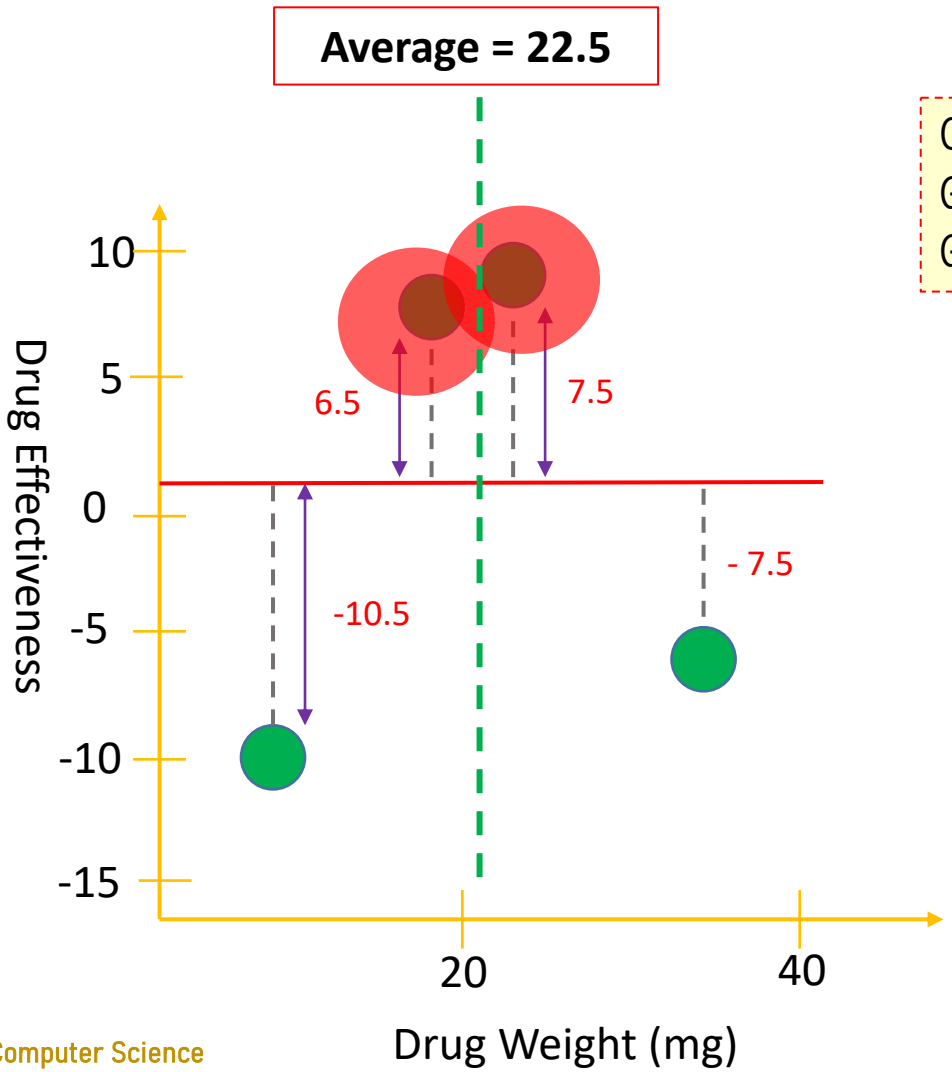
6.5, 7.5, -7.5

SC = 14.08

Residual giống nhau hoặc không triệt tiêu lẫn nhau, nên SC khá lớn

XGBoost For Regression

❑ *Step 1*

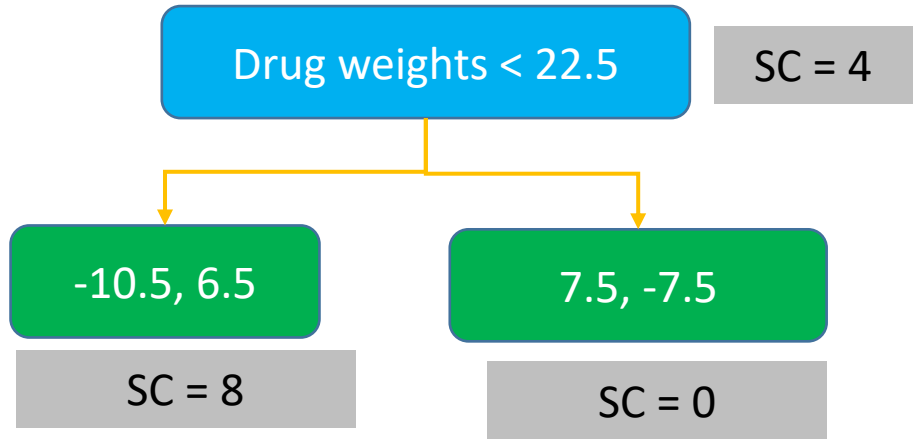


SC = 4

-10.5, 6.5, 7.5, -7.5

Residual rất khác nhau, triệt tiêu lẫn nhau, nên SC nhỏ

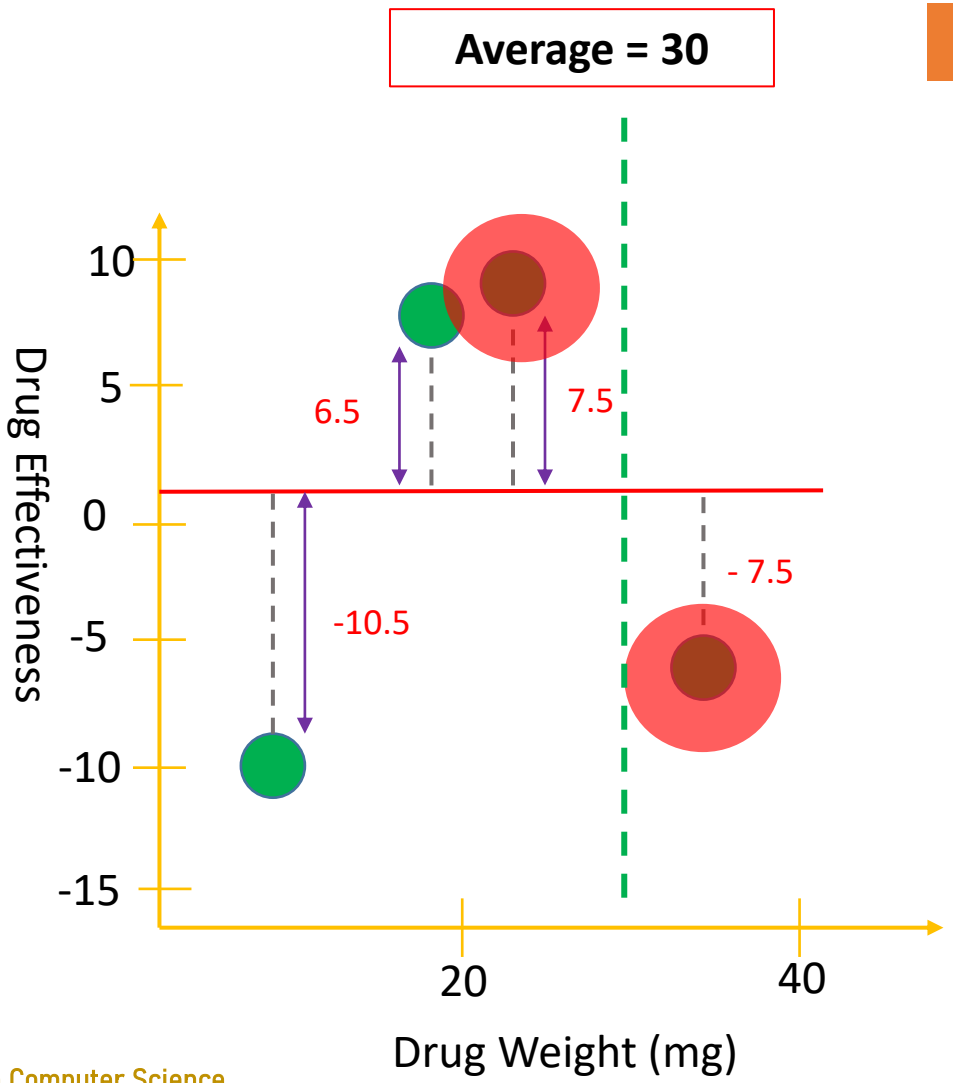
Caculate the Gain.
Gain = Left SC + Right SC - Root SC
Gain = 4.0



Residual giống nhau hoặc không triệt tiêu lẫn nhau, nên SC khá lớn

XGBoost For Regression

❑ *Step 1*

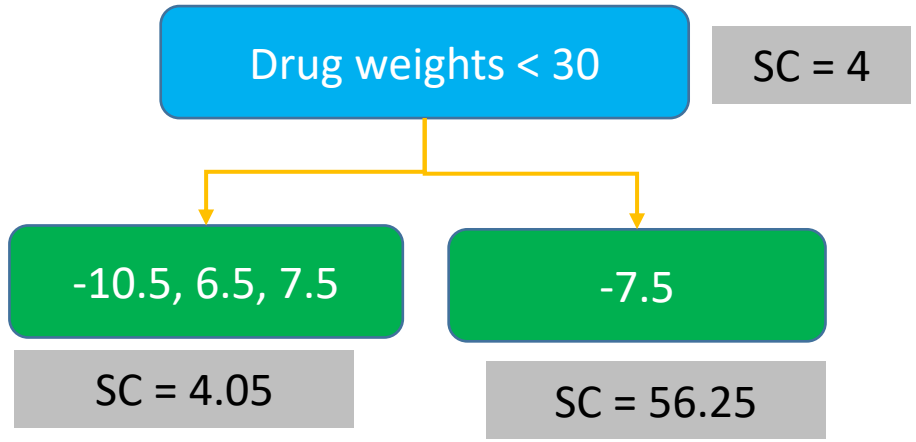


SC = 4

-10.5, 6.5, 7.5, -7.5

Residual rất khác nhau, triệt tiêu lẫn nhau, nên SC nhỏ

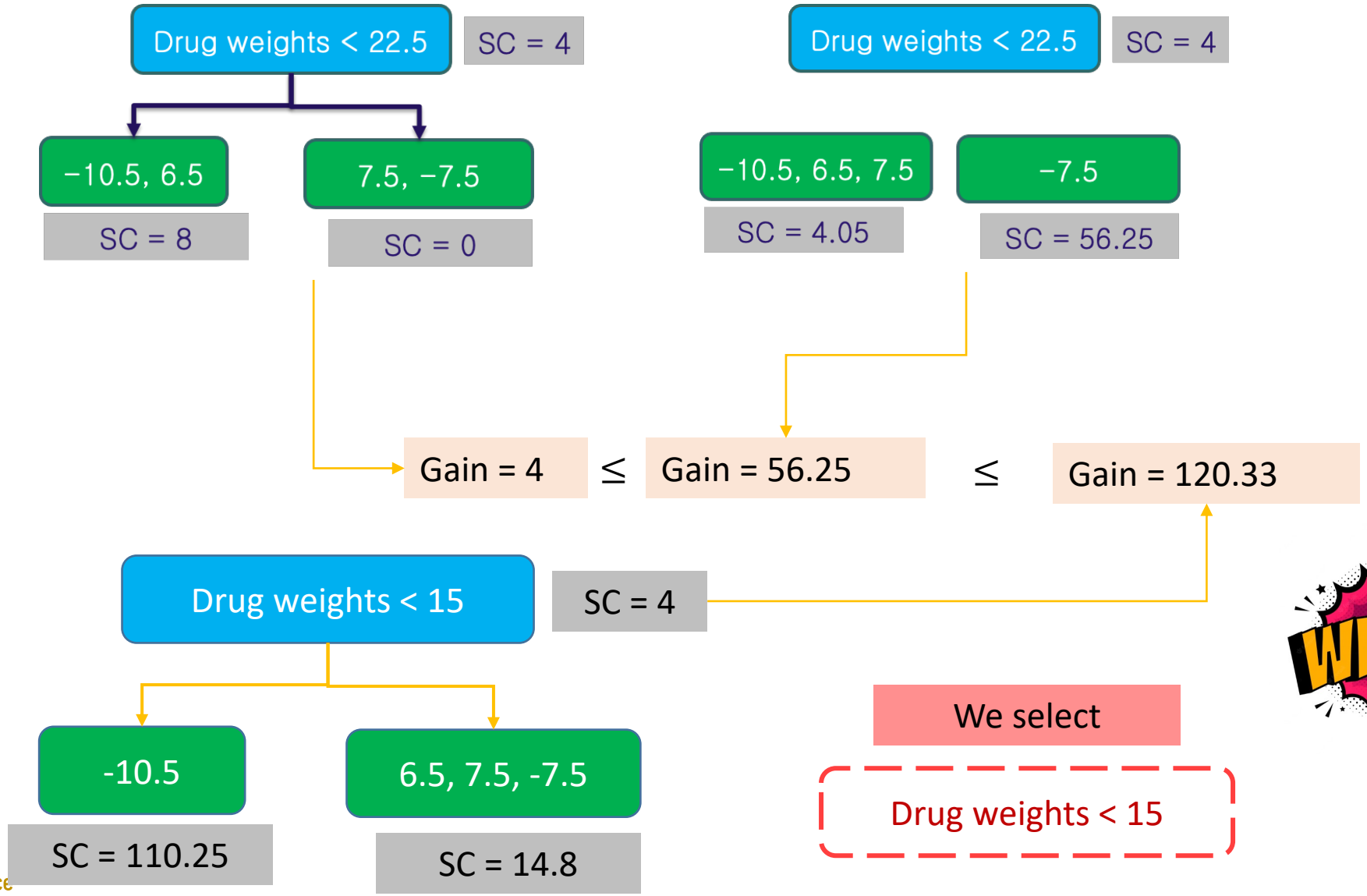
Caculate the Gain.
Gain = Left SC + Right SC - Root SC
Gain = 56.33



Residual giống nhau hoặc không triệt tiêu lẫn nhau, nên SC khá lơn

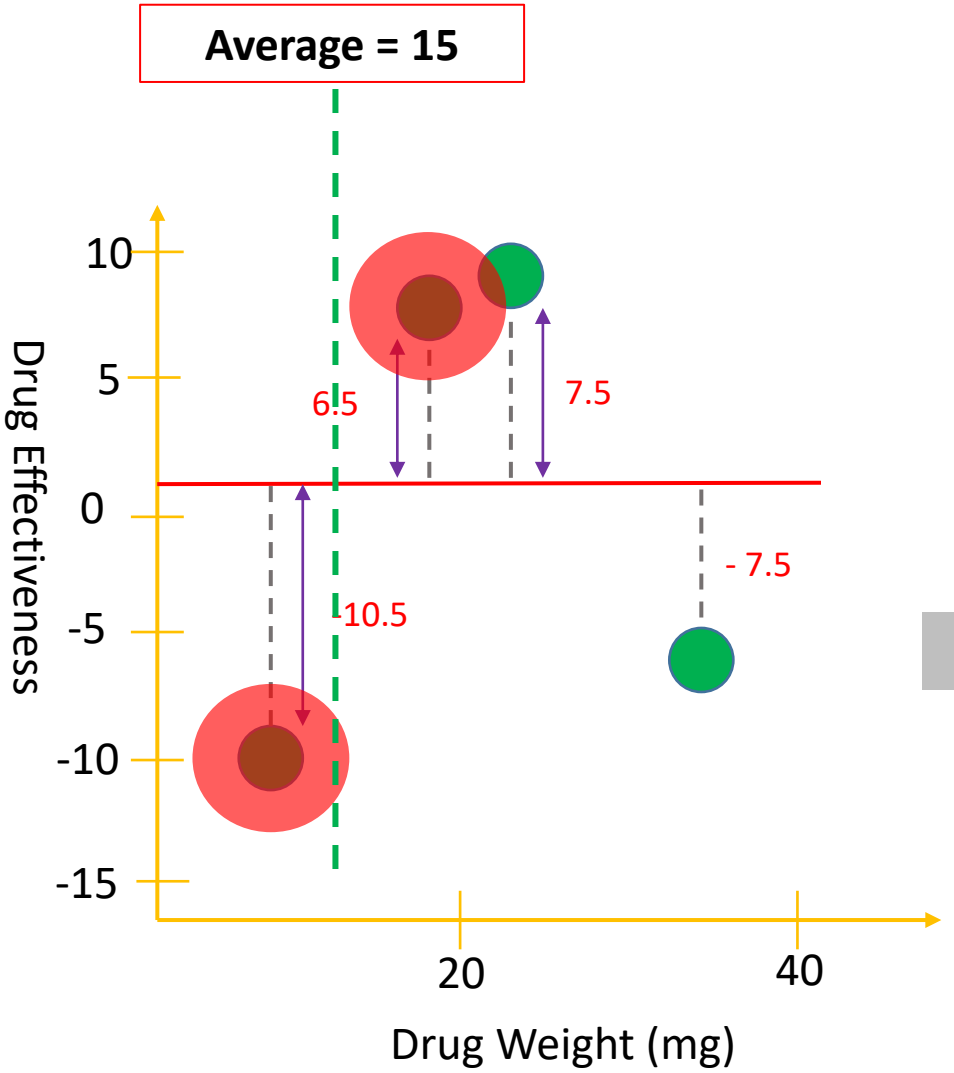
XGBoost For Regression

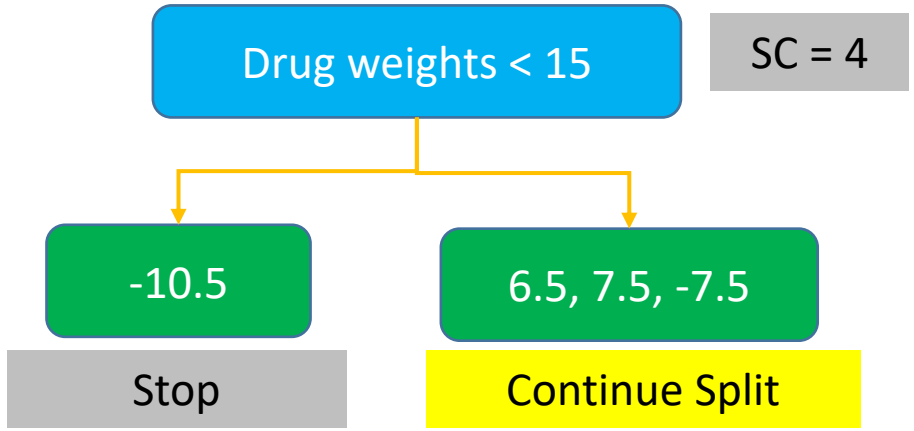
❑ *Step 1*



XGBoost For Regression

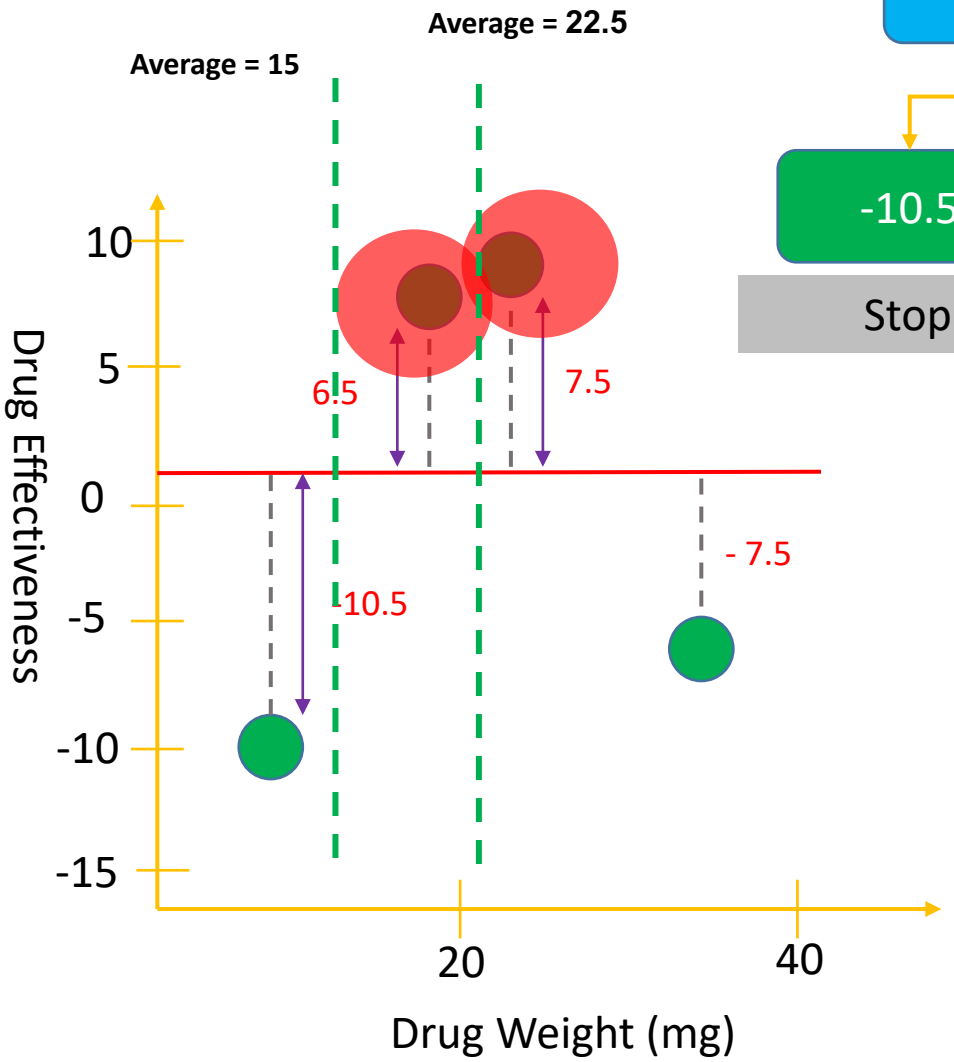
❑ *Step 1*

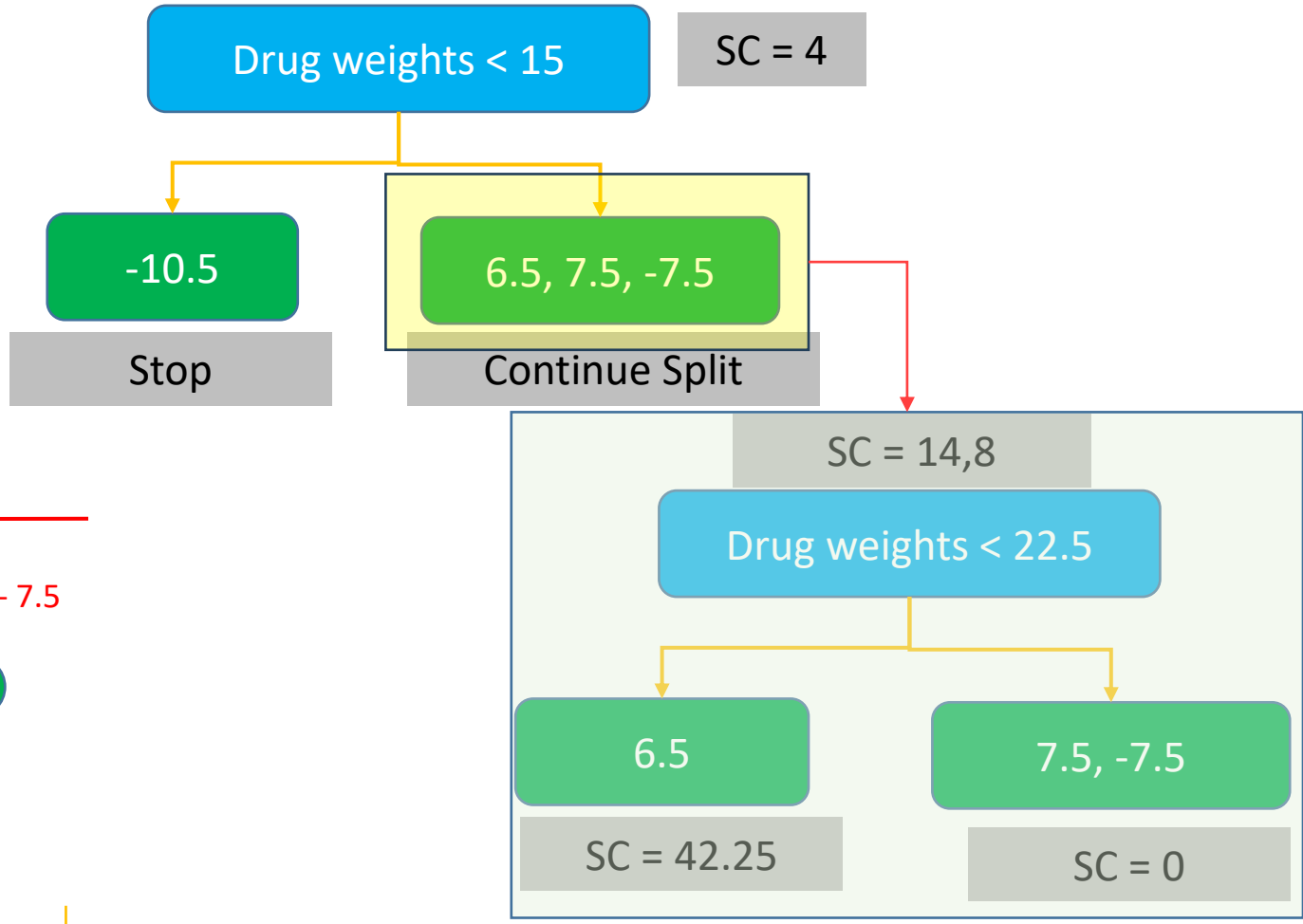




XGBoost For Regression

❑ *Step 1*

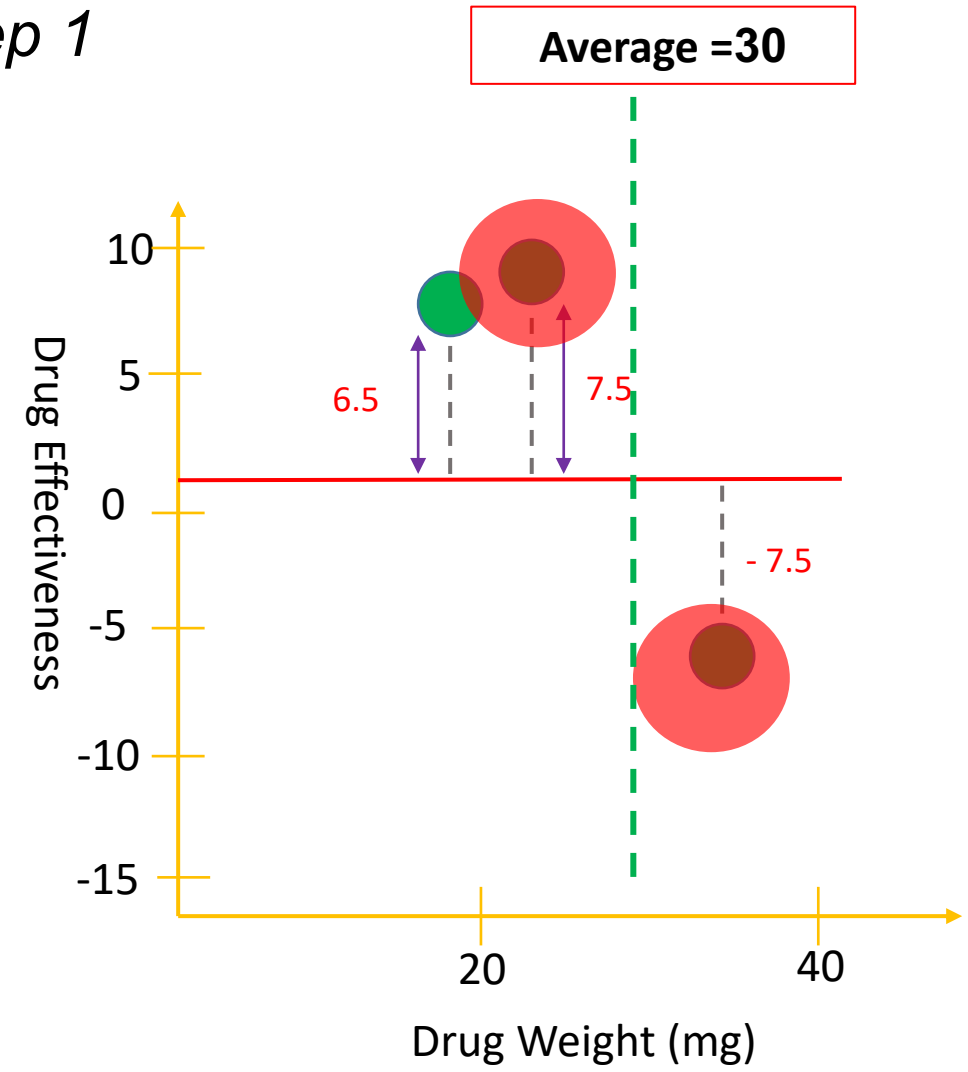


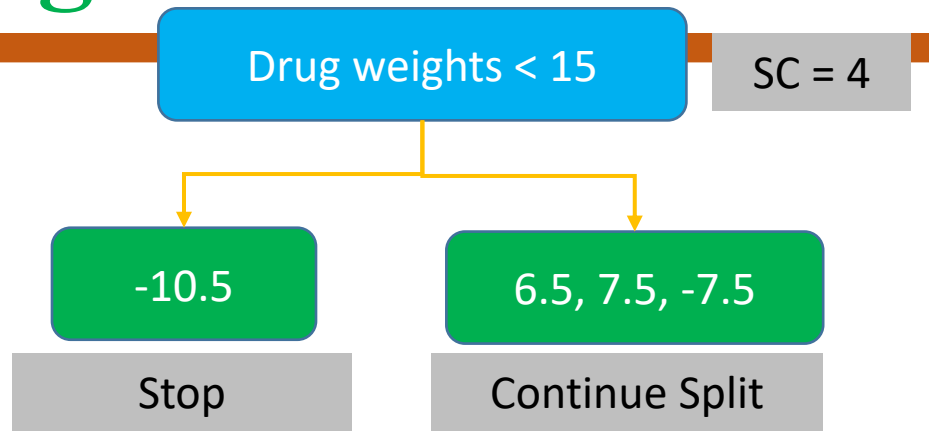


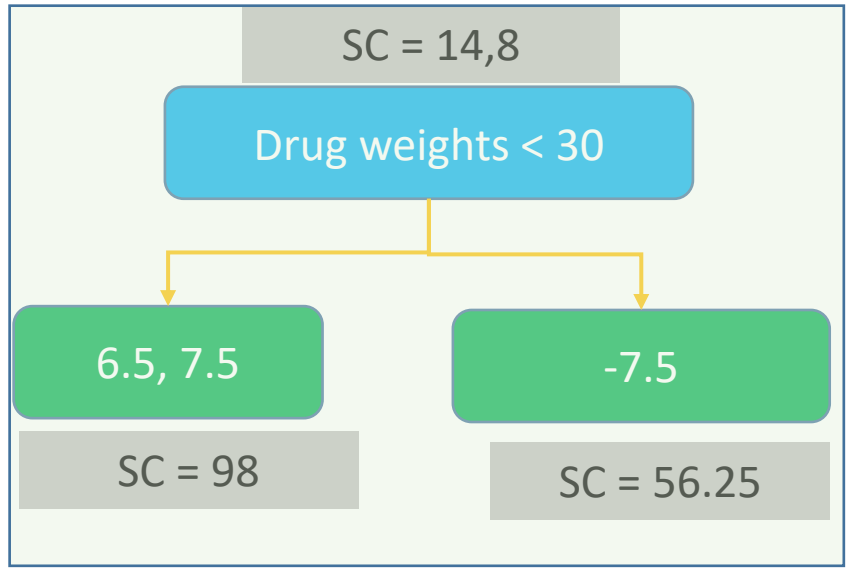
Caculate the Gain
Gain = Left SC + Right SC - Root SC
Gain = 28.17

XGBoost For Regression

❑ *Step 1*



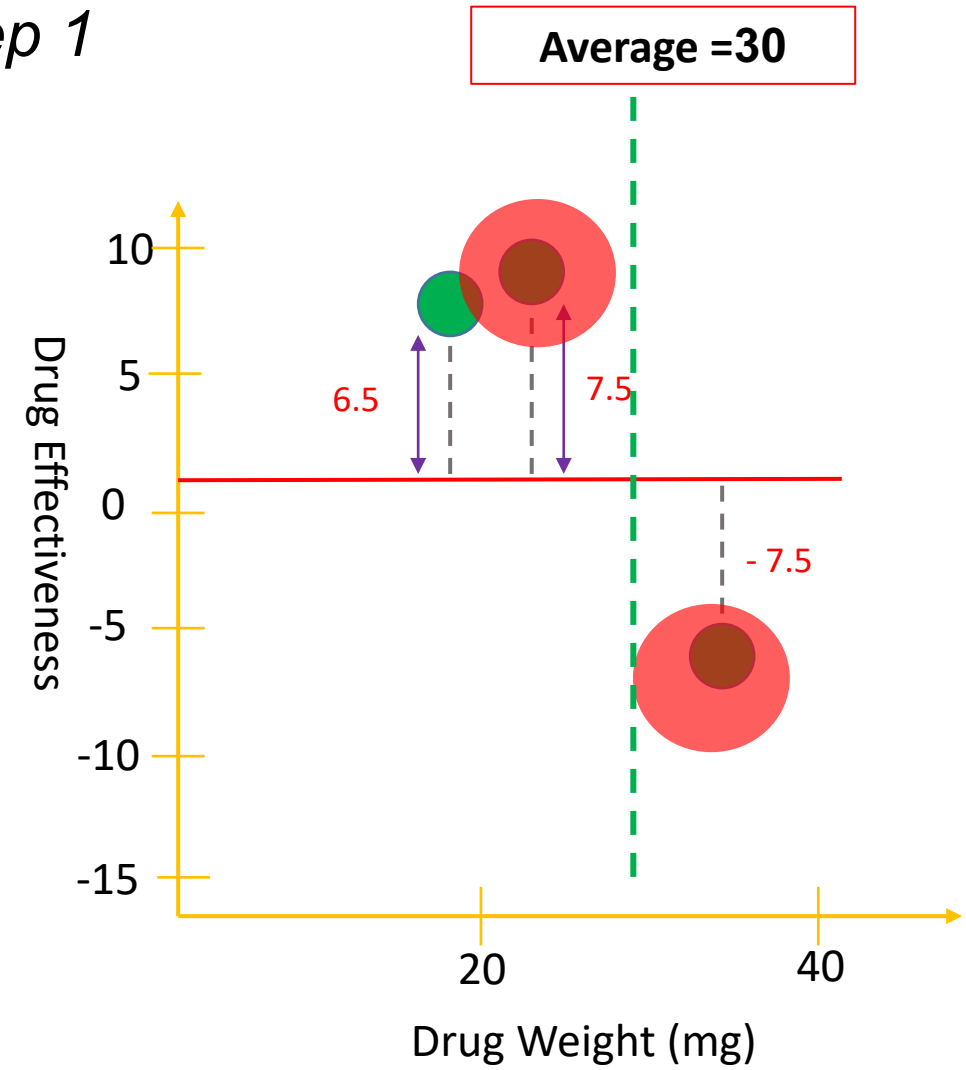


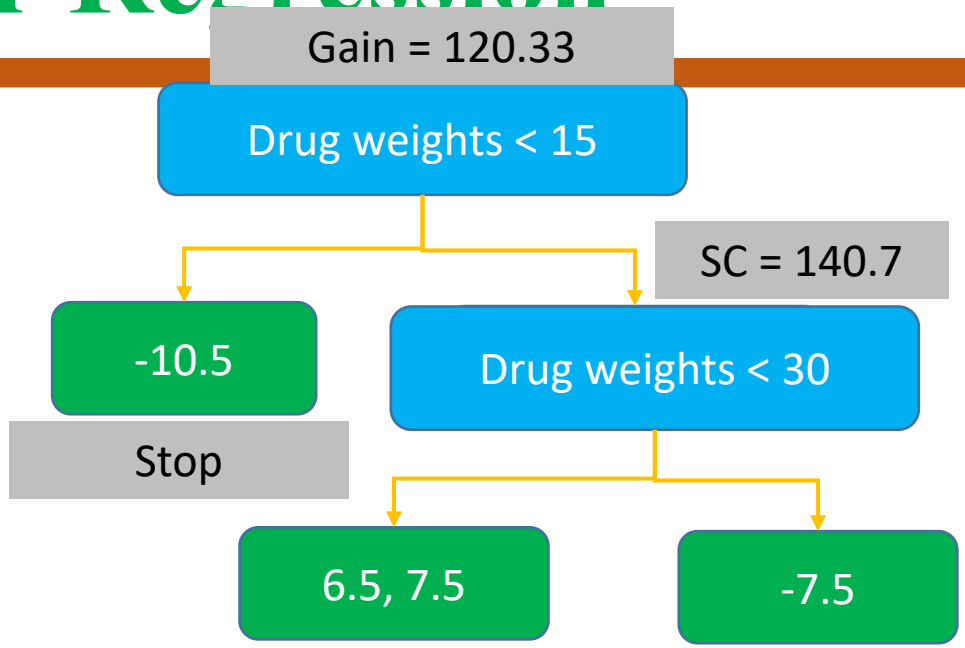


Caculate the Gain
Gain = Left SC + Right SC - Root SC
Gain = 140.17

XGBoost For Regression

❑ *Step 1*



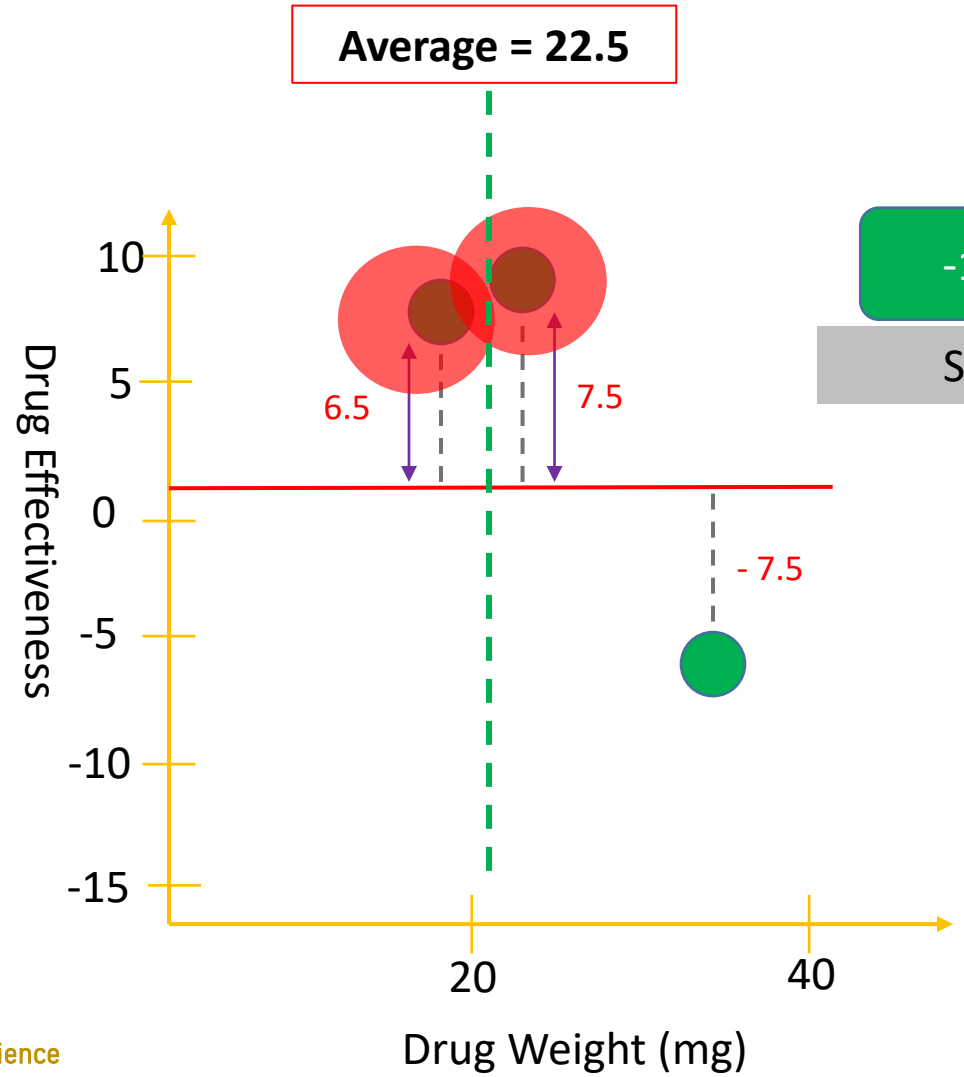


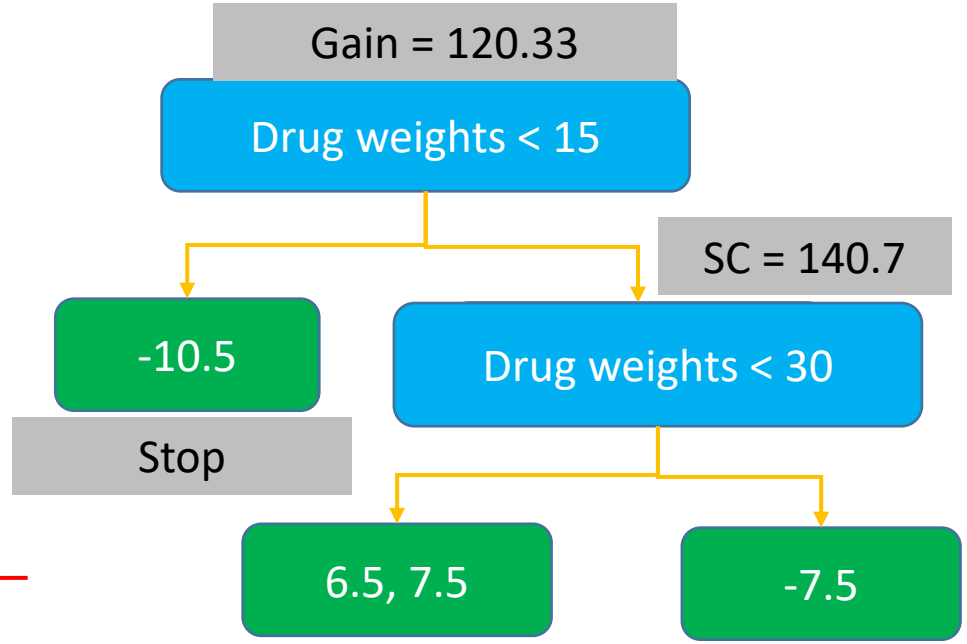
How to prune the tree to prevent Overfitting ? Gain information

$\gamma = 130$

XGBoost For Regression

❑ *Step 1*





How to prune the tree to prevent Overfitting ? Gain information

$\gamma = 130$

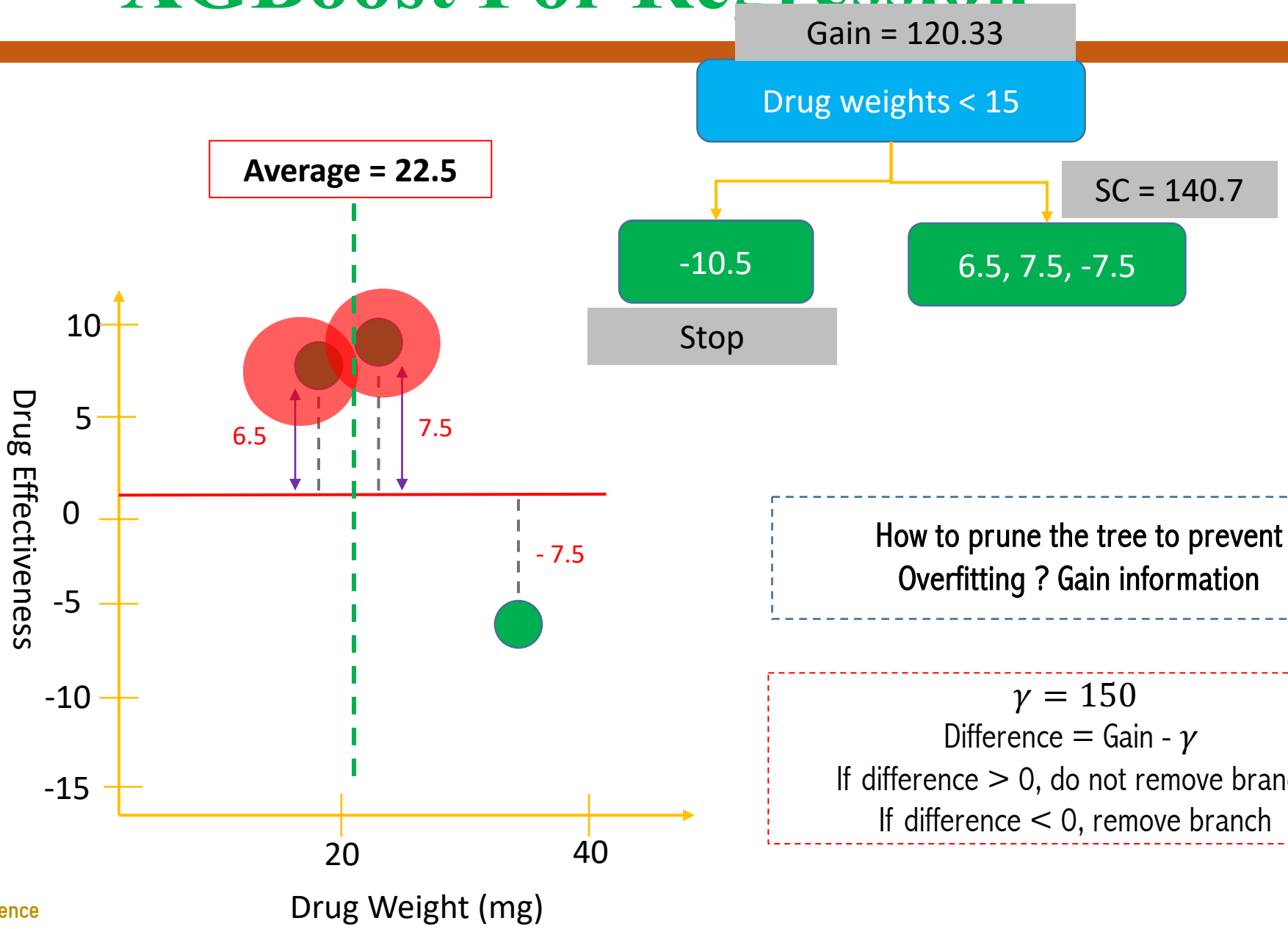
Difference = Gain - γ

If difference > 0, do not remove branch

If difference < 0, remove branch

XGBoost For Regression

❑ *Step 1*



How to prune the tree to prevent Overfitting ? Gain information

$\gamma = 150$

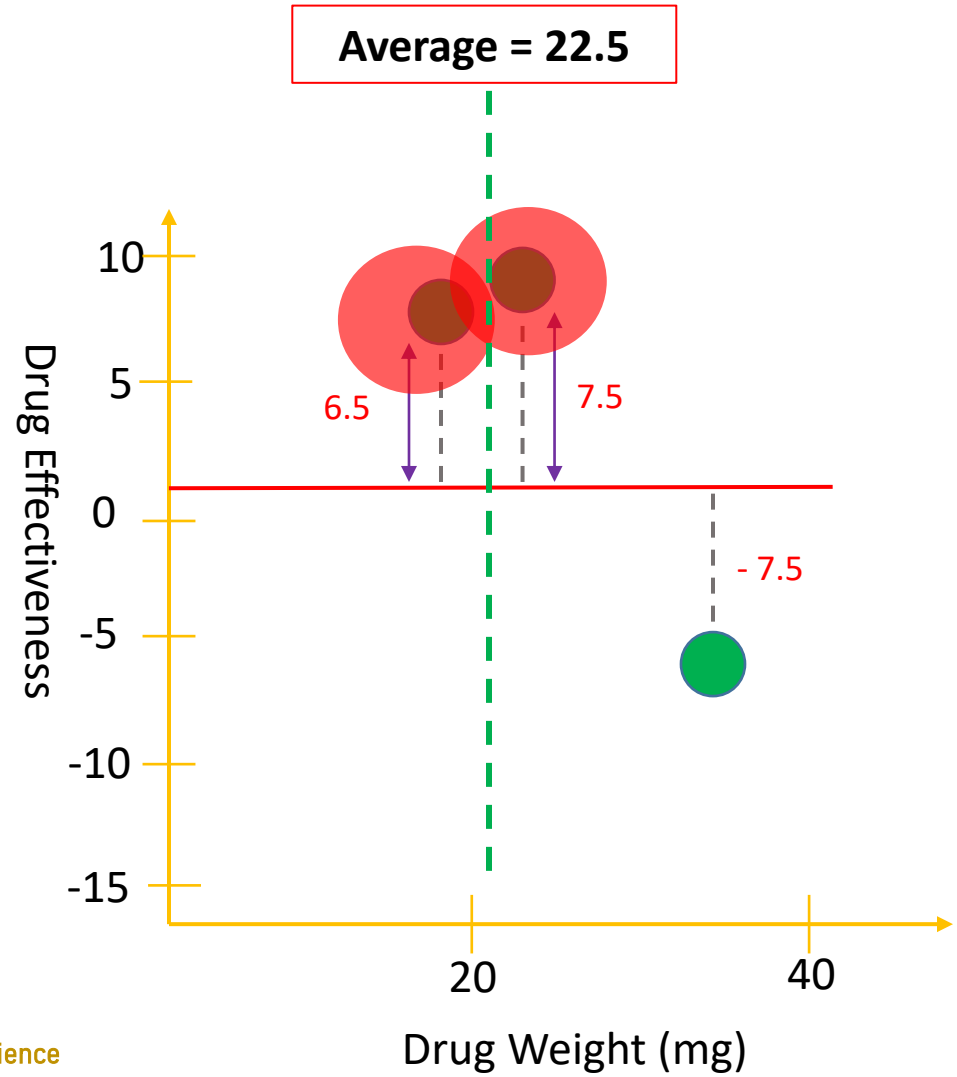
Difference = Gain - γ

If difference > 0, do not remove branch

If difference < 0, remove branch

XGBoost For Regression

❑ *Step 1*



0.5

How to prune the tree to prevent Overfitting ? Gain information

$\gamma = 150$

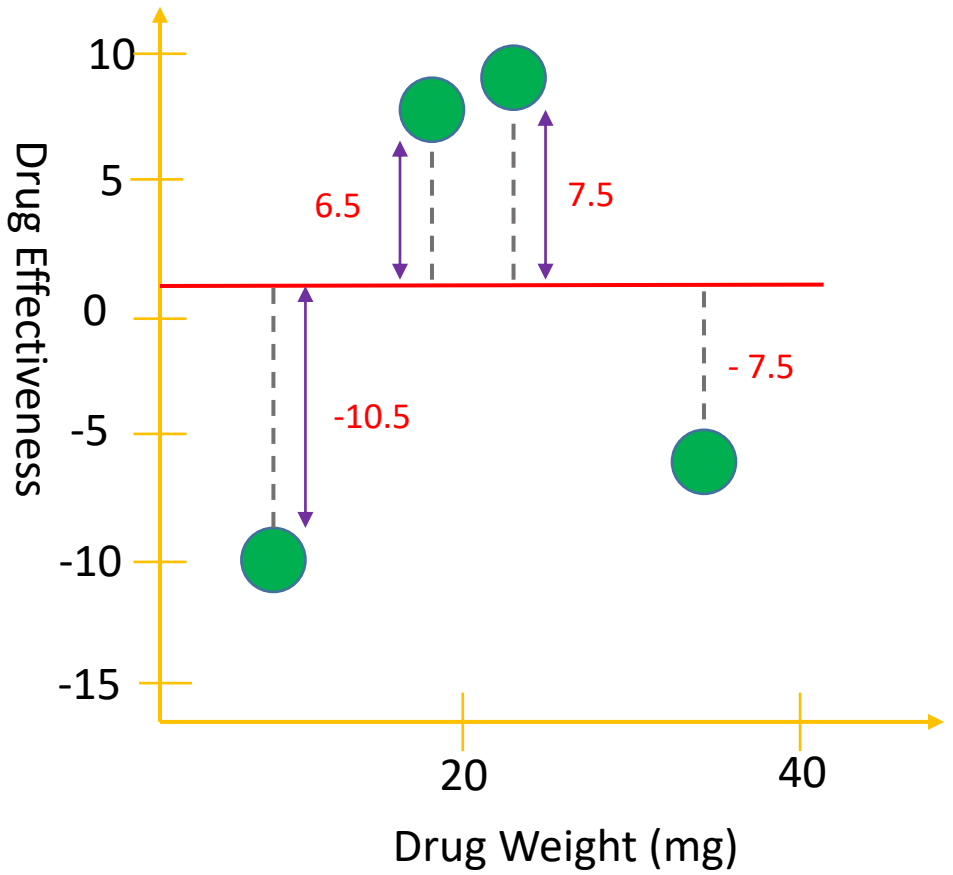
Difference = Gain - γ

If difference > 0, do not remove branch

If difference < 0, remove branch

XGBoost For Regression

❑ *Step 1*



Start with single Leaf of residuals

-10.5, 6.5, 7.5, -7.5

m = 4
λ = 1

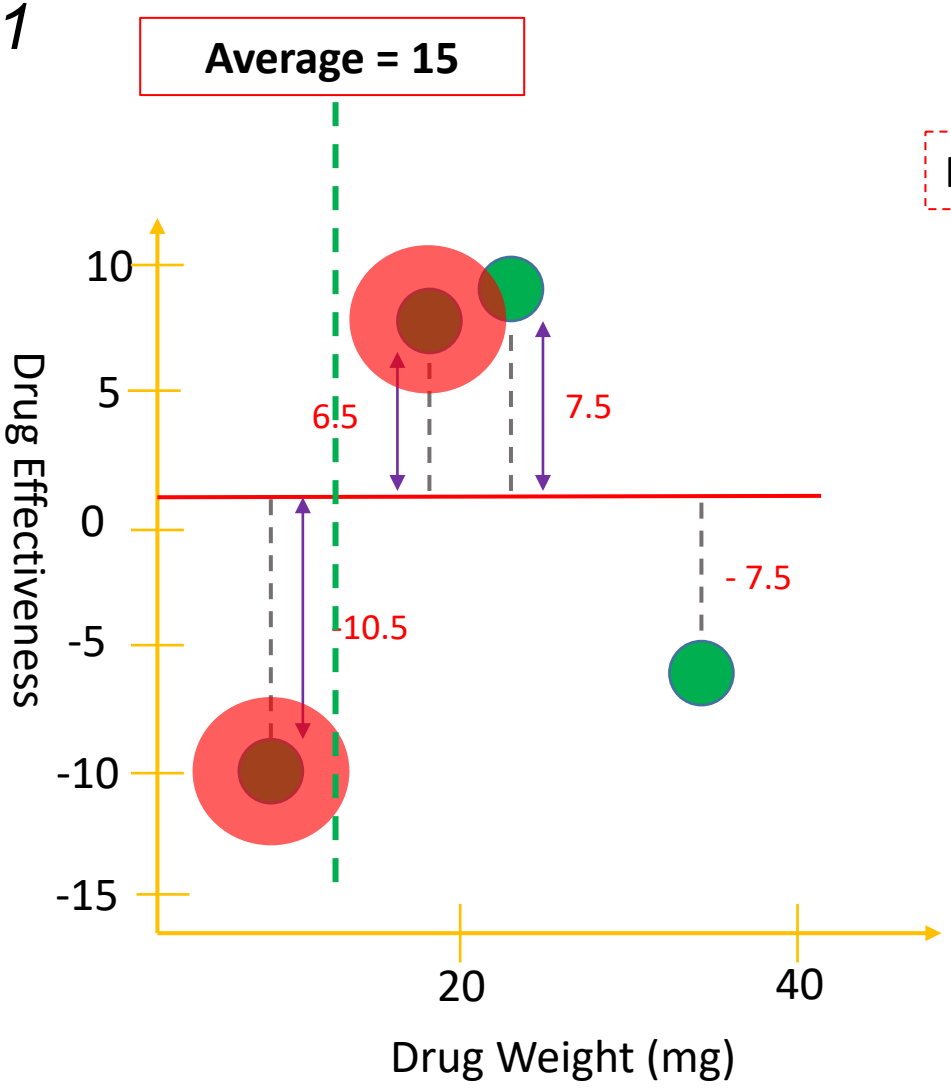
Compute Similarity Score

$$SC = \frac{\sum (output - predicted)^2}{m + \lambda}$$

$$SC = \frac{[-10.5+7.5+6.5+(-7.5)]^2}{4+1} = 3.2$$

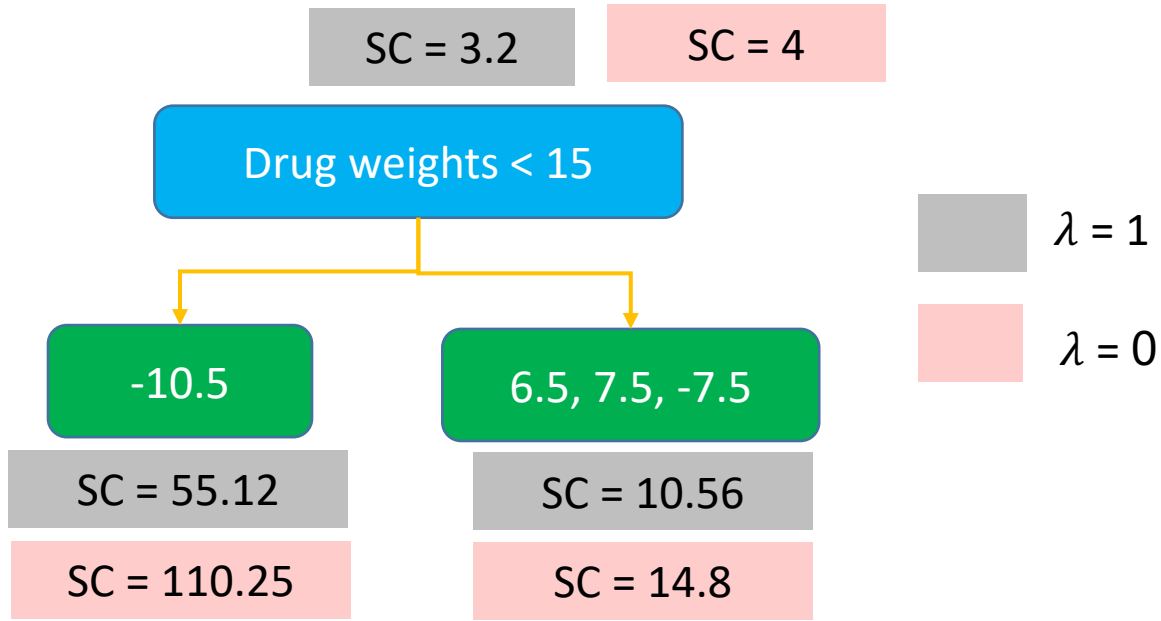
XGBoost For Regression

❑ *Step 1*



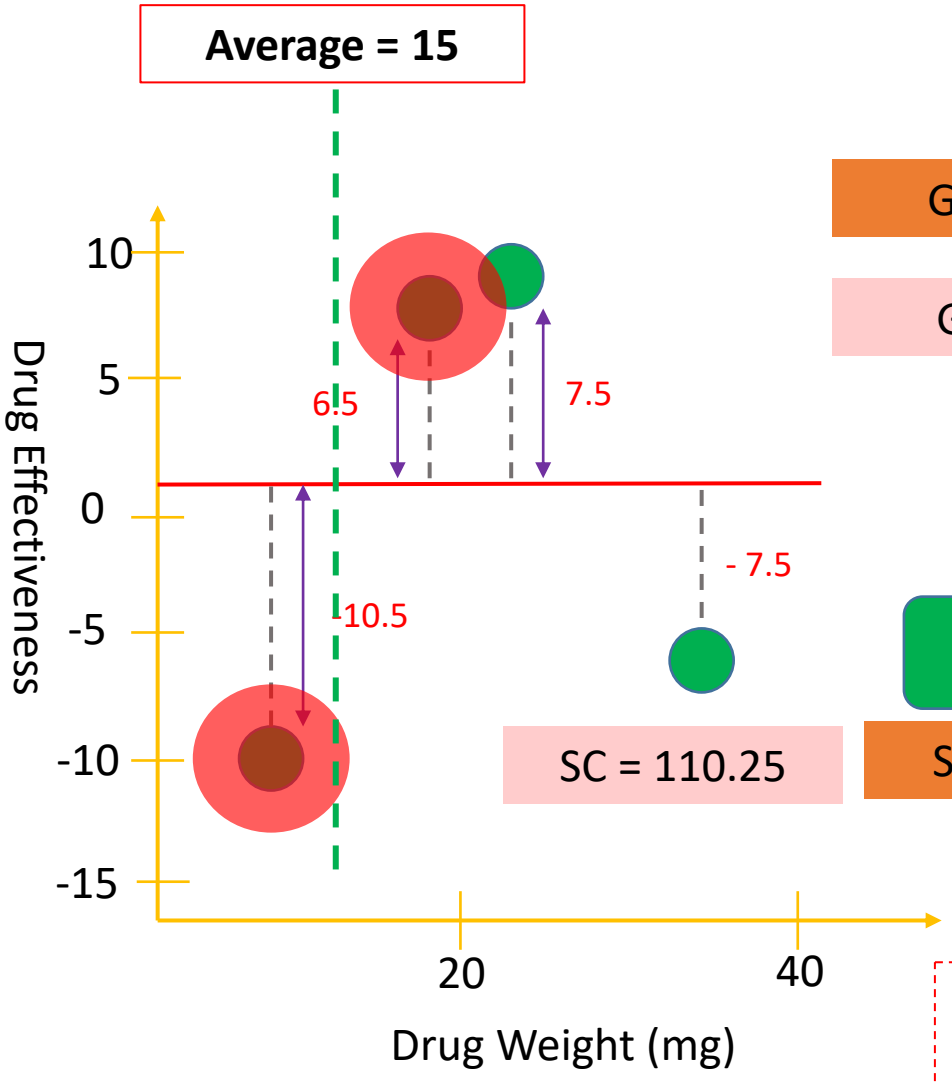
SC = 3.2 | -10.5, 6.5, 7.5, -7.5

Please look at the two outputs with lowest drug weights



When $\lambda > 0$, the similarity score are smaller
Inversely proportional to the number of residuals

XGBoost For Regression



$\lambda = 1$ $\lambda = 0$

Gain = 55.12 + 10.56 – 3.2 = 62.48

Gain = 55.12 + 10.56 – 4 = 120.33

Drug weights < 15

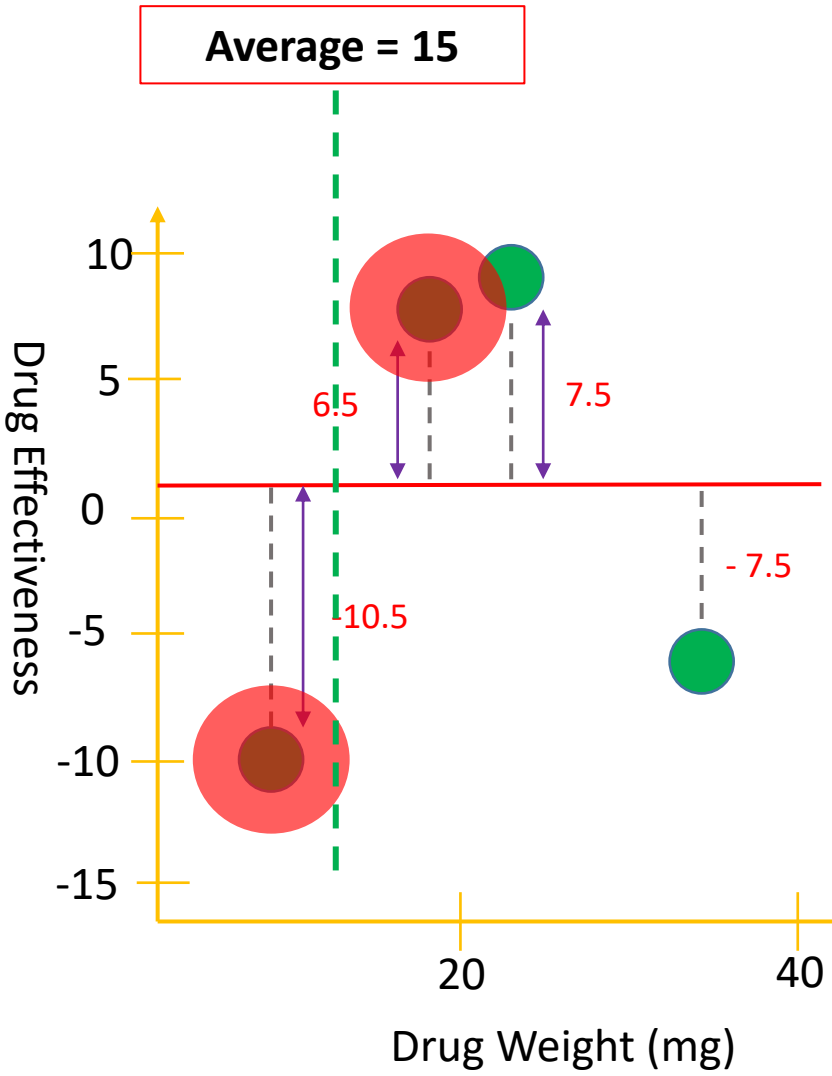
Gain = 82.9 Gain = 140.17

-10.5 6.5, 7.5, -7.5

SC = 110.25 SC = 55.12 SC = 10.56 SC = 14.8

The amount of decrease is invertly propotional to the number of Residual in the nodes

XGBoost For Regression



$\lambda = 0$

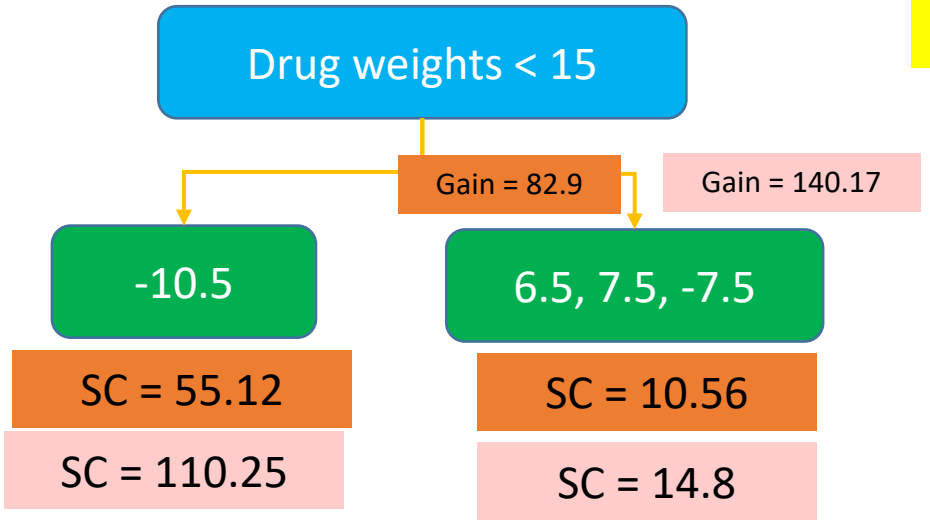
$\lambda = 1$

$\lambda > 0$: easy to prune the tree
Prevent overffiting

Gain = 55.12 + 10.56 – 4 = 62.48

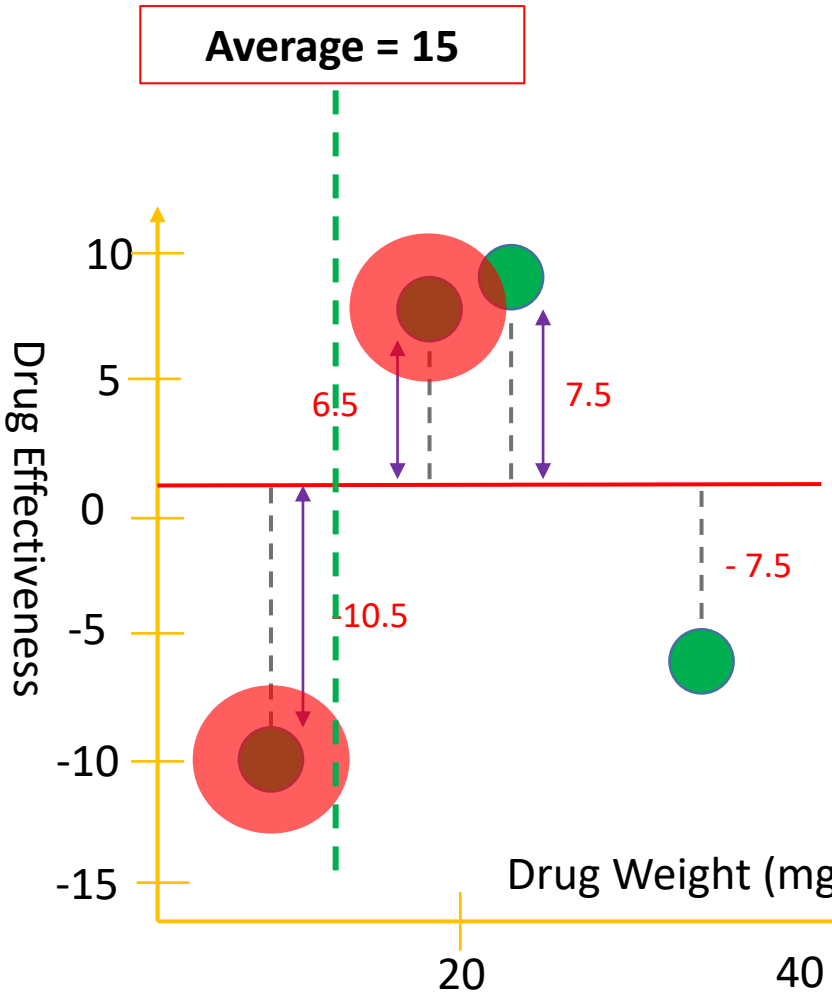
Gain = 55.12 + 10.56 – 4 = 120.33

Prunning parameter: $\gamma = 130$



The amount of decrease is invertly propotional to the number of Residual in the nodes

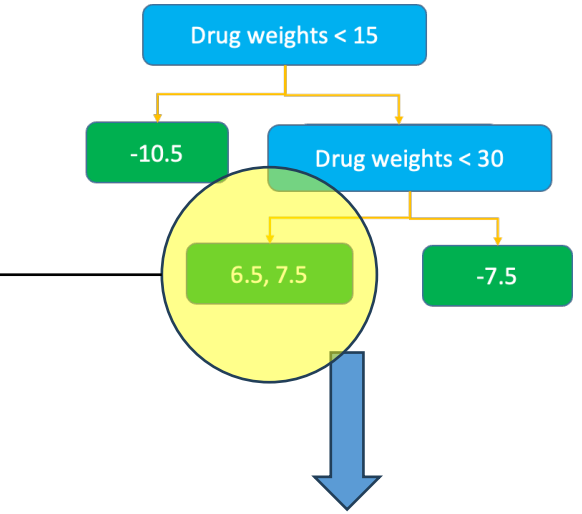
XGBoost For Regression



Setting $\gamma = 0$ do not turn off prunning

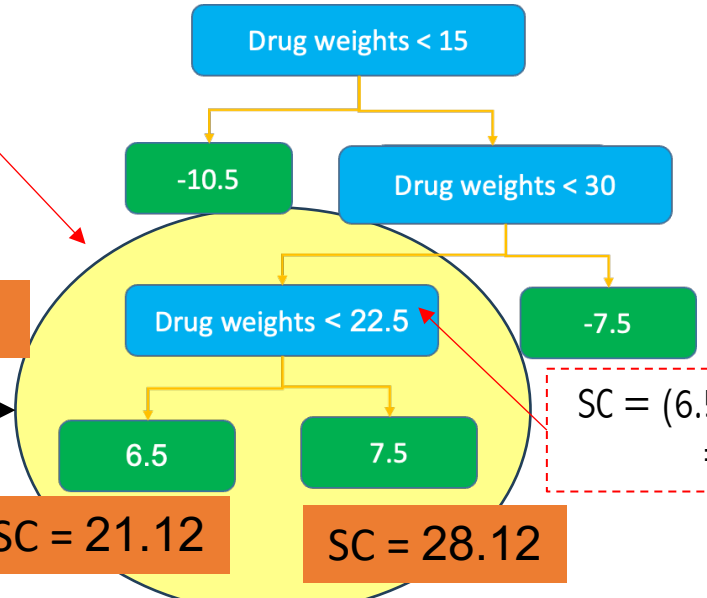
We will remove this branch: -16.06- $\gamma < 0$

Prunning parameter: $\gamma = 0$



$\lambda = 1$

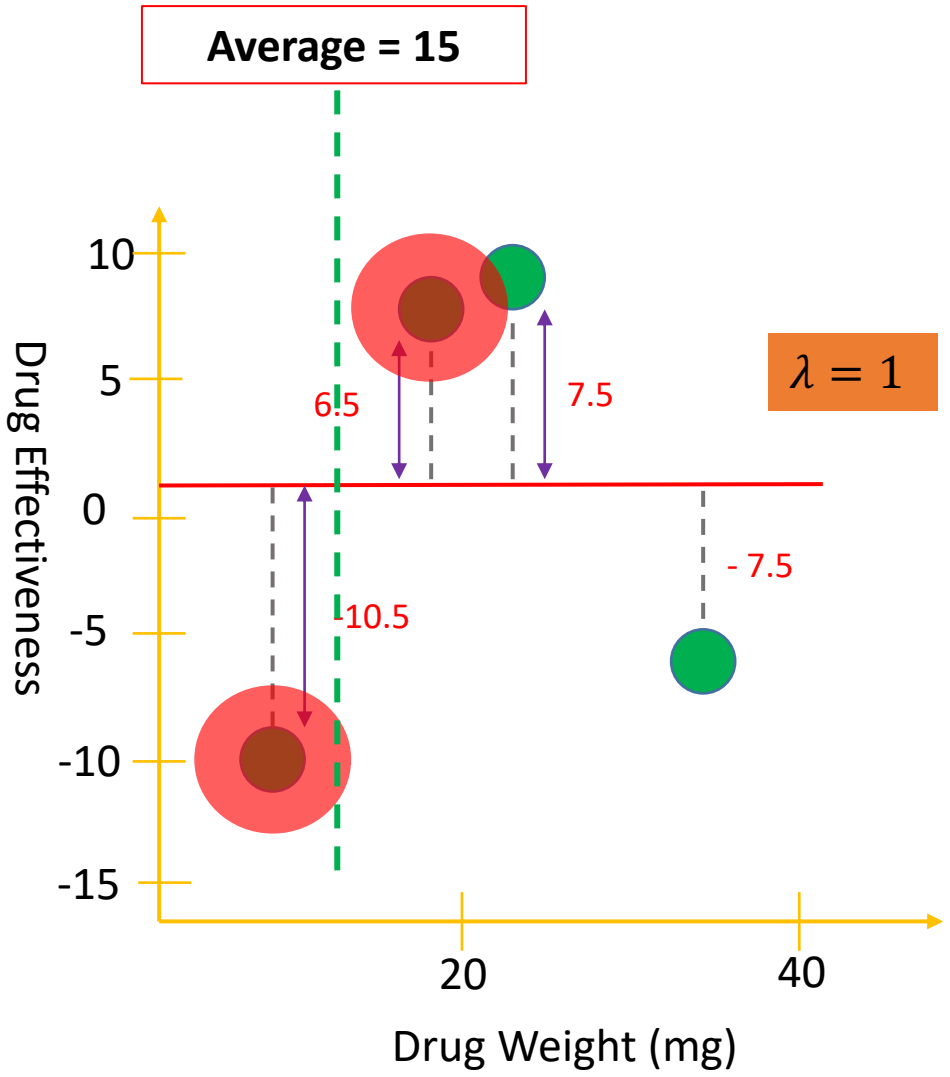
SC = 65.3



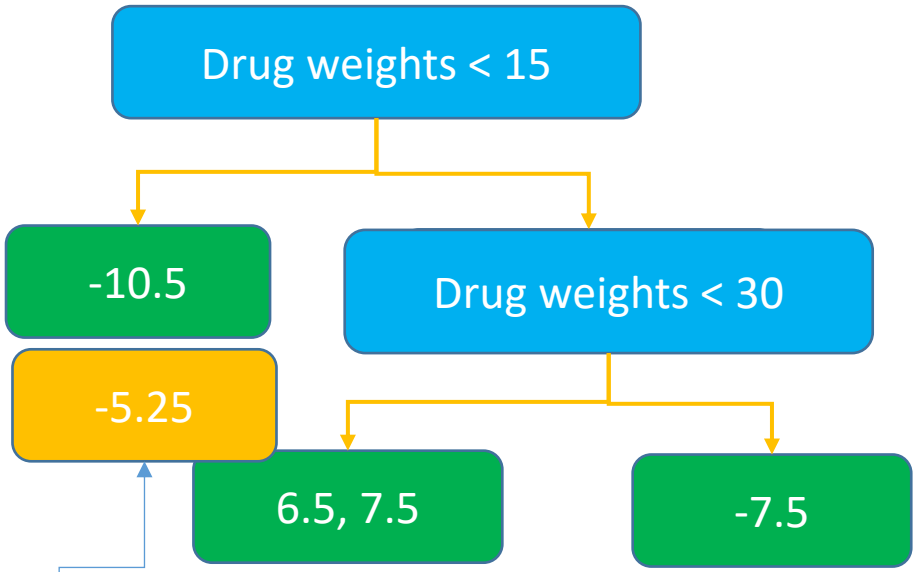
SC = $(6.5+7.5)^2/(2+1)$ = 65.3

Gain = 21.12 + 28.12 – 65.3 = -16.06

How to Predict Value



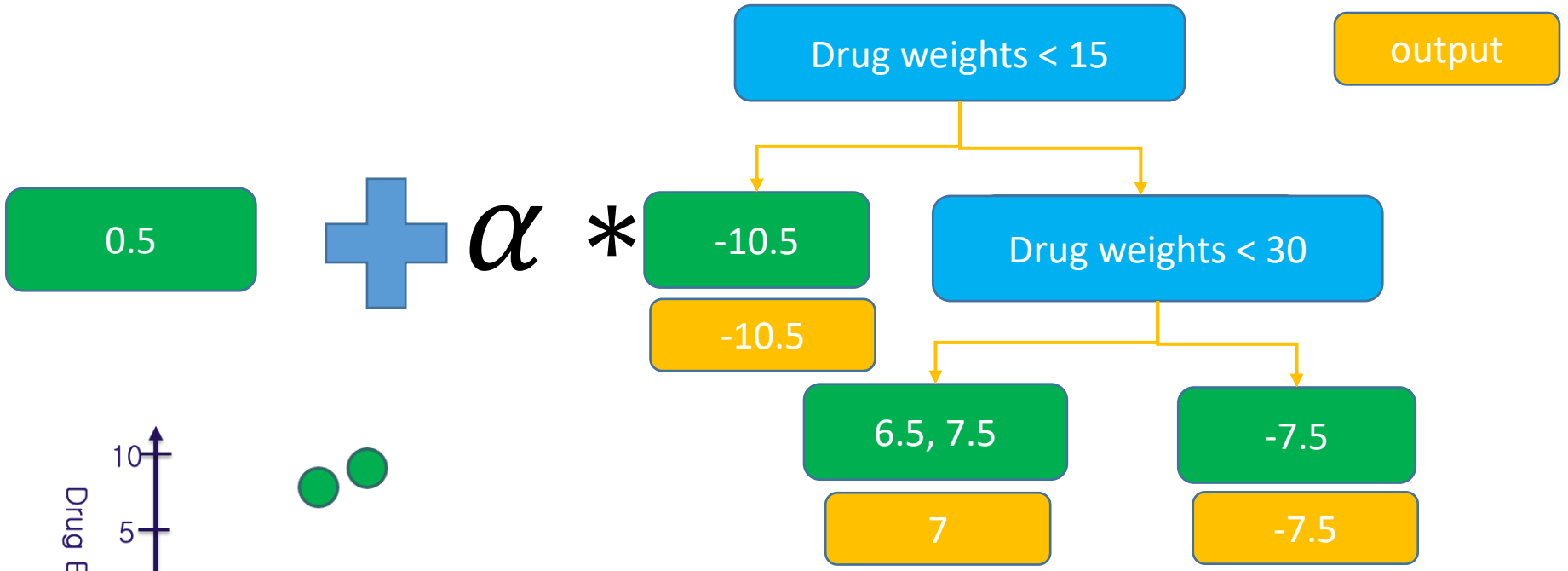
$\lambda = 1$

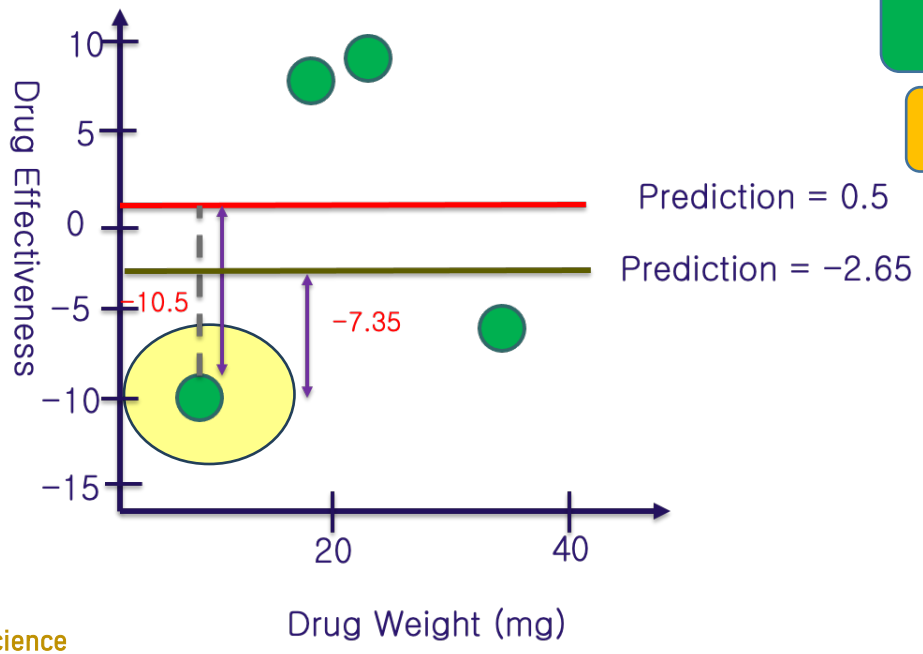


$$\text{Output Value} = \frac{\text{Sum of Residuals}}{\text{Number of Residuals} + \lambda}$$

When $\lambda > 0$, it will reduce the amount that this indiviual observation add to the overal prediction

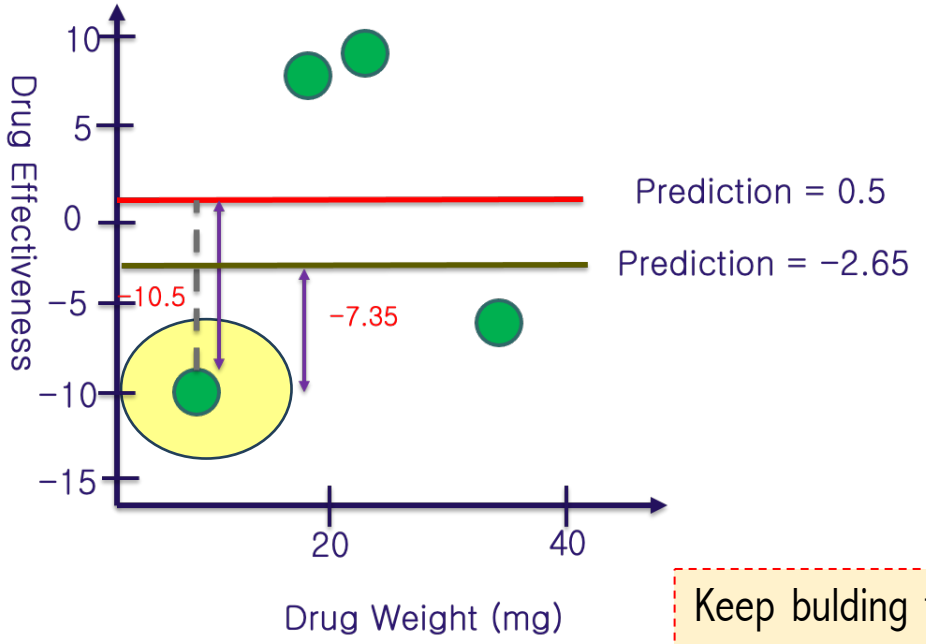
How to Predict Value

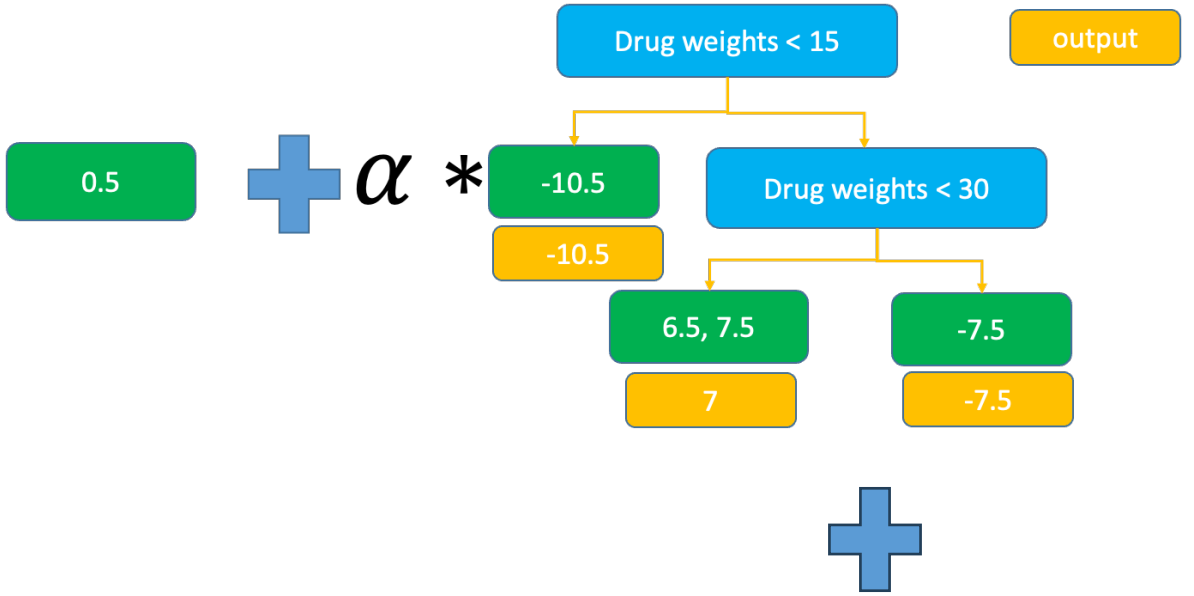




$\lambda = 0$ $\alpha = 0.3$

Building the Next Tree

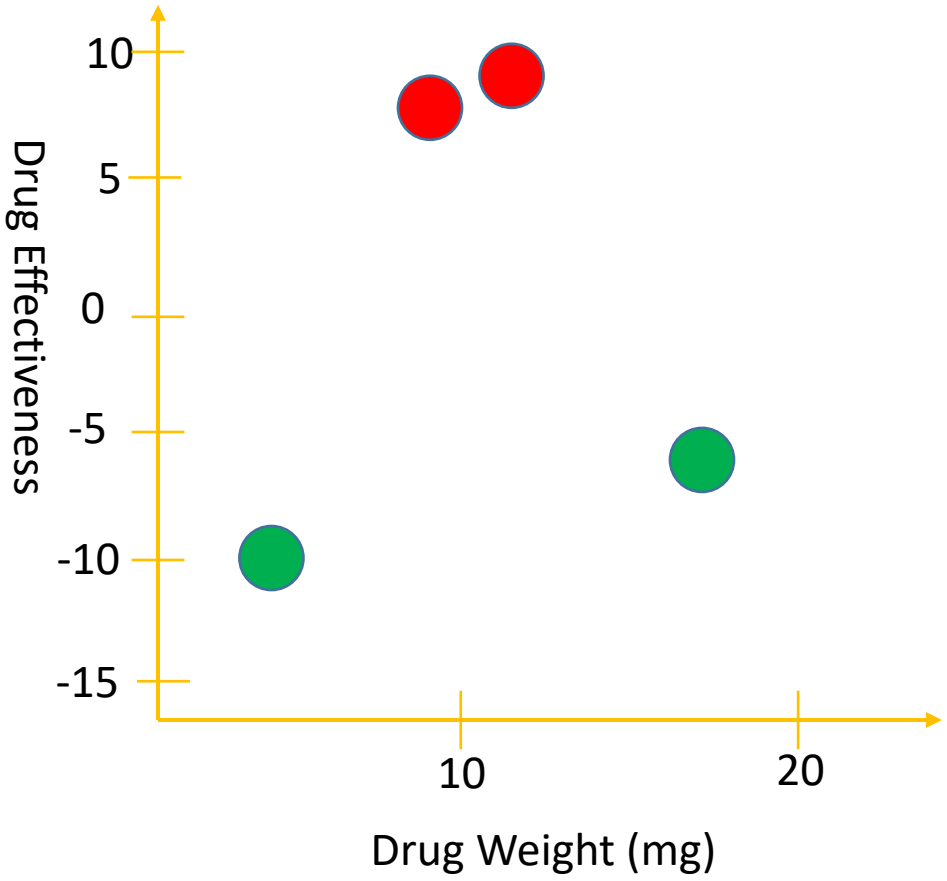




$\alpha *$ Next Tree Result

Keep bulding the Tree until the Residual are reach the predefined threshold. Or we reach to the maximum number of Tree

XGBoot For Classification



Not effectiviness

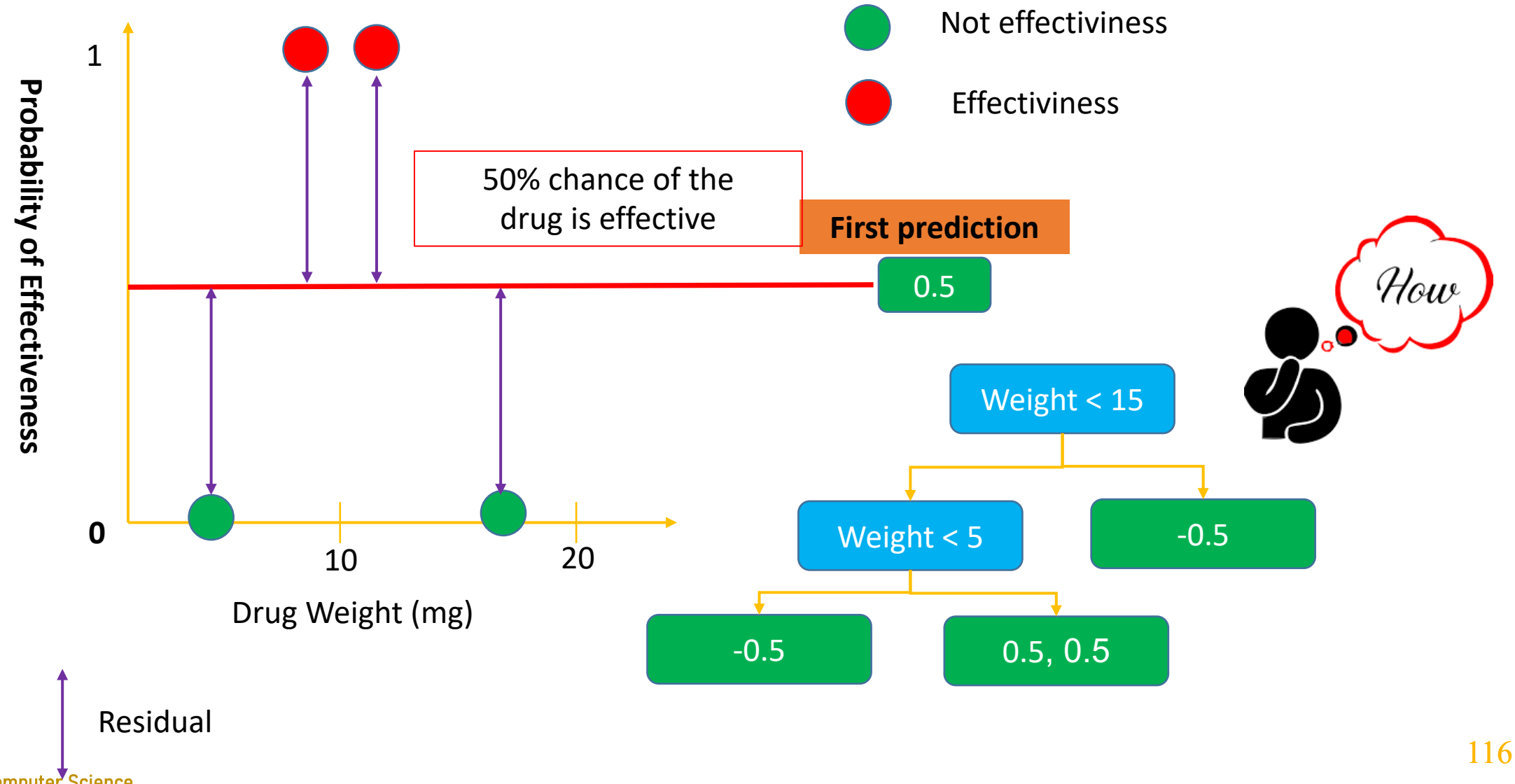
Effectiviness

First prediction

0.5

50% chance of the drug is effective

XGBoot For Classification



XGBoot For Classification

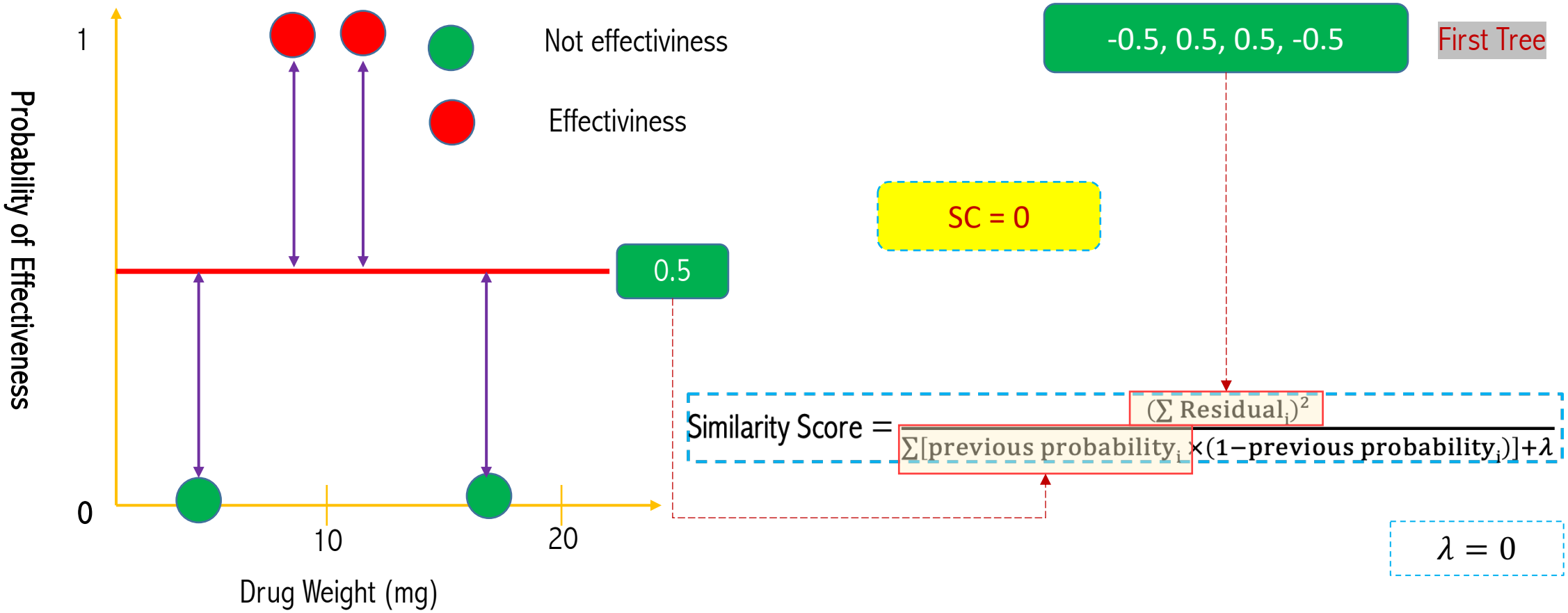
Similarity Score for Classification:

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

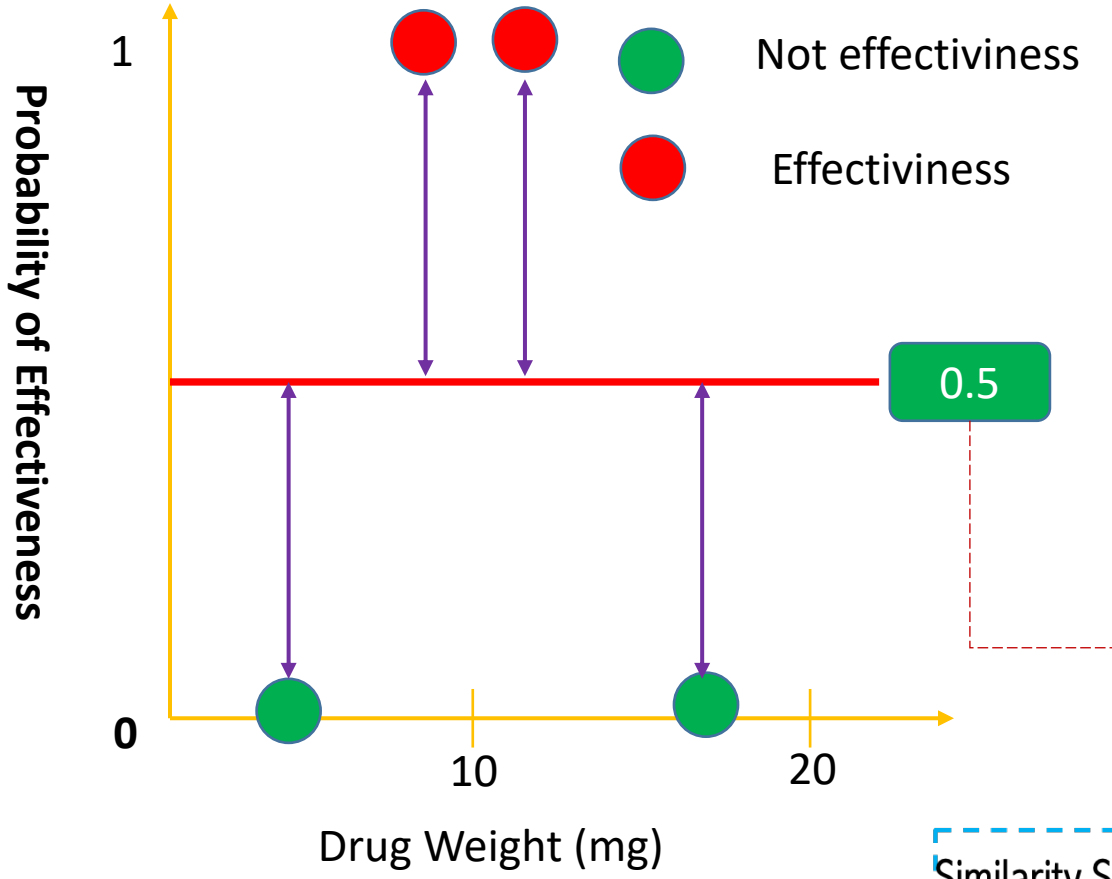
Similarity Score for Prediction (regression):

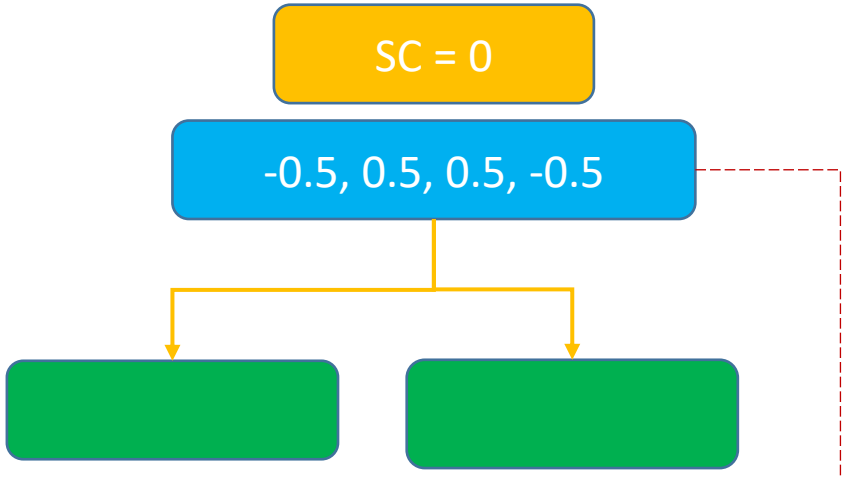
$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\text{number of residual} + \lambda}$$

XGBoost for Classification



XGBoost For Classification

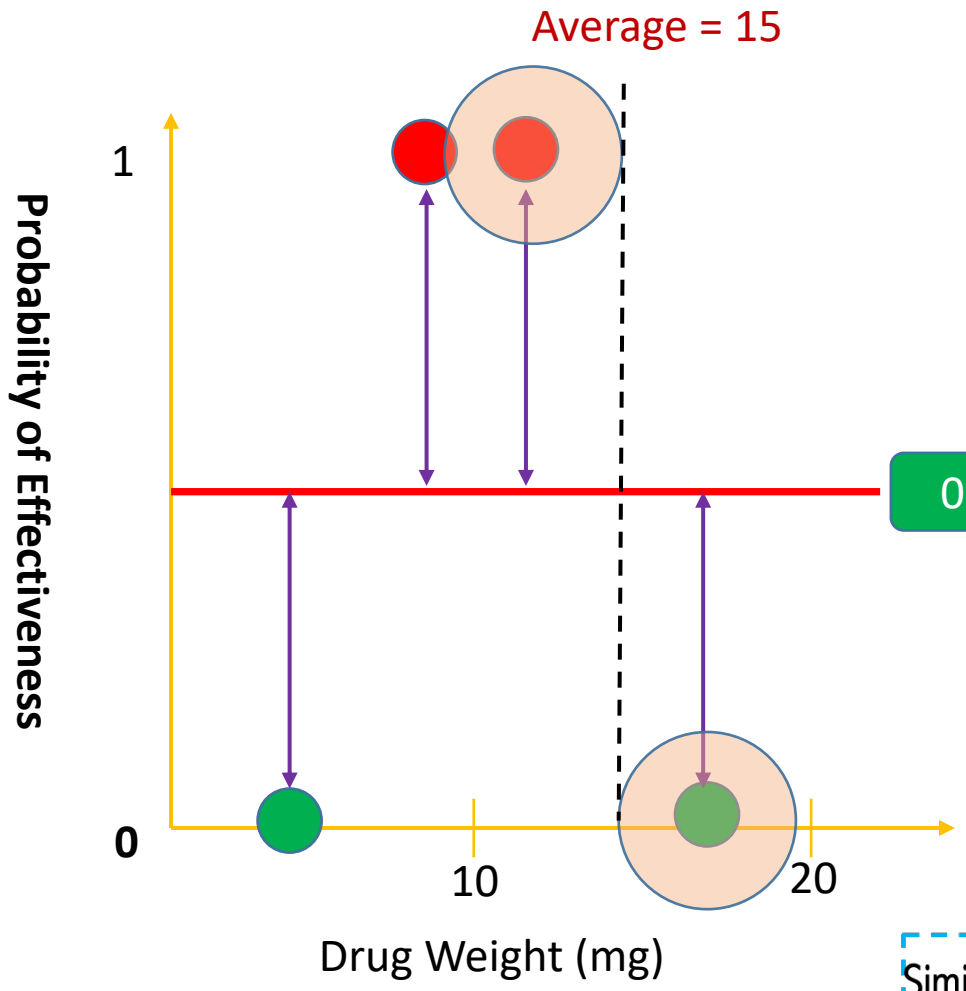




$$\lambda = 0$$

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

XGBoost For Classification



0.5

SC = 0

Weights < 15

-0.5, 0.5, 0.5

-0.5

SC = 0.33

SC = 1

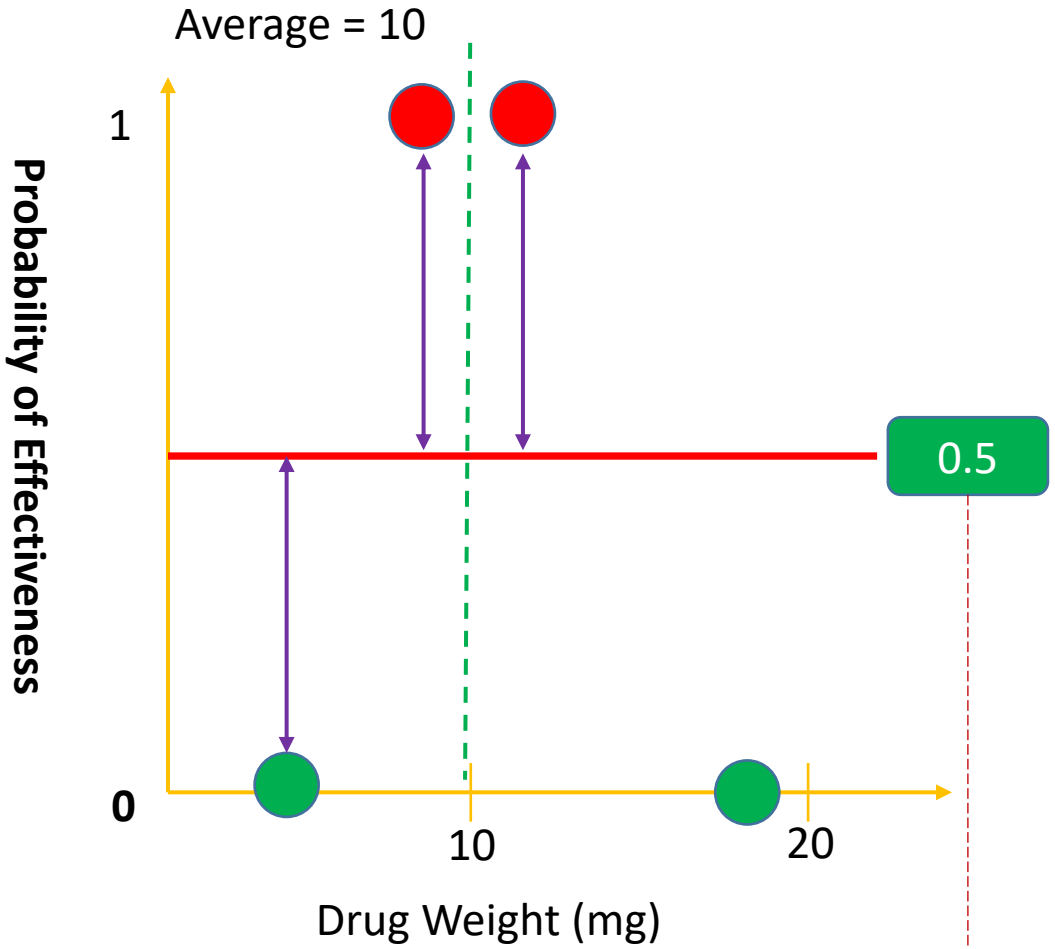
Gain = 0.33 + 1 – 0 =1.33

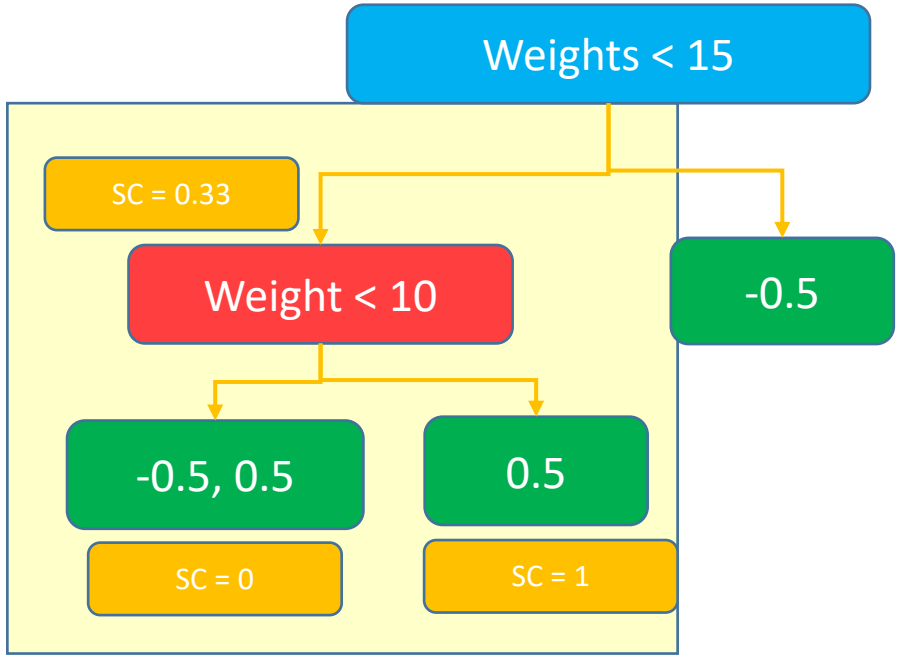
Supposing that weights < 15 is best threshold

$\lambda = 0$

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

XGBoost For Classification



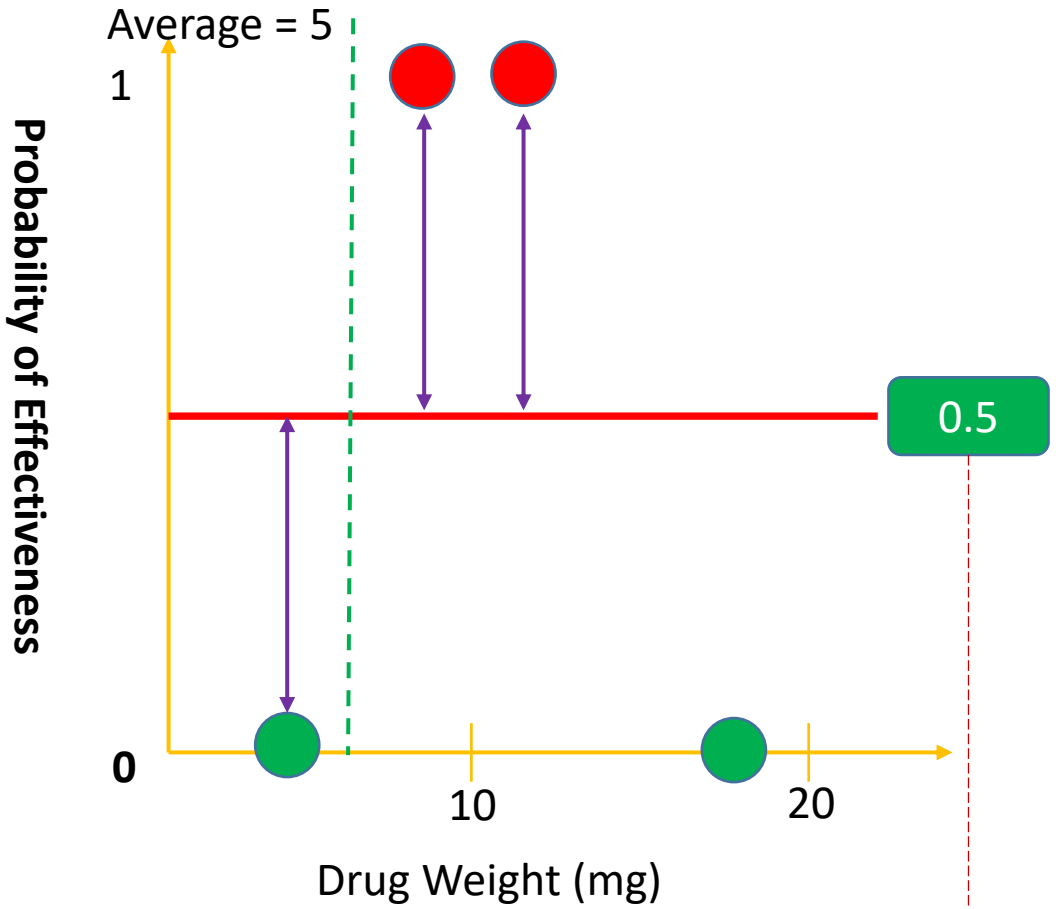


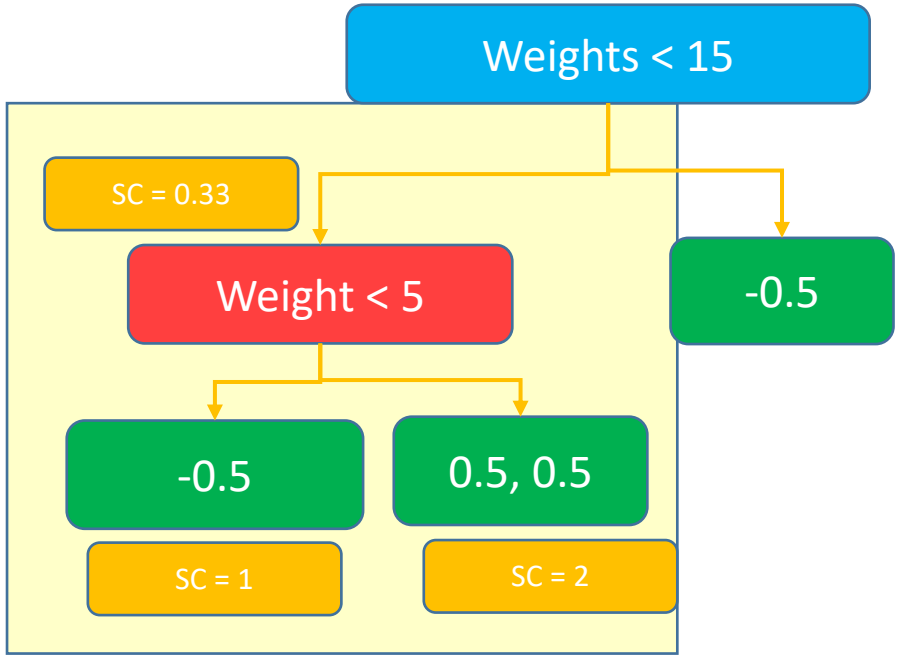
Gain = 0 + 1 – 0.33 = 0.66

$\lambda = 0$

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

XGBoost for Classification

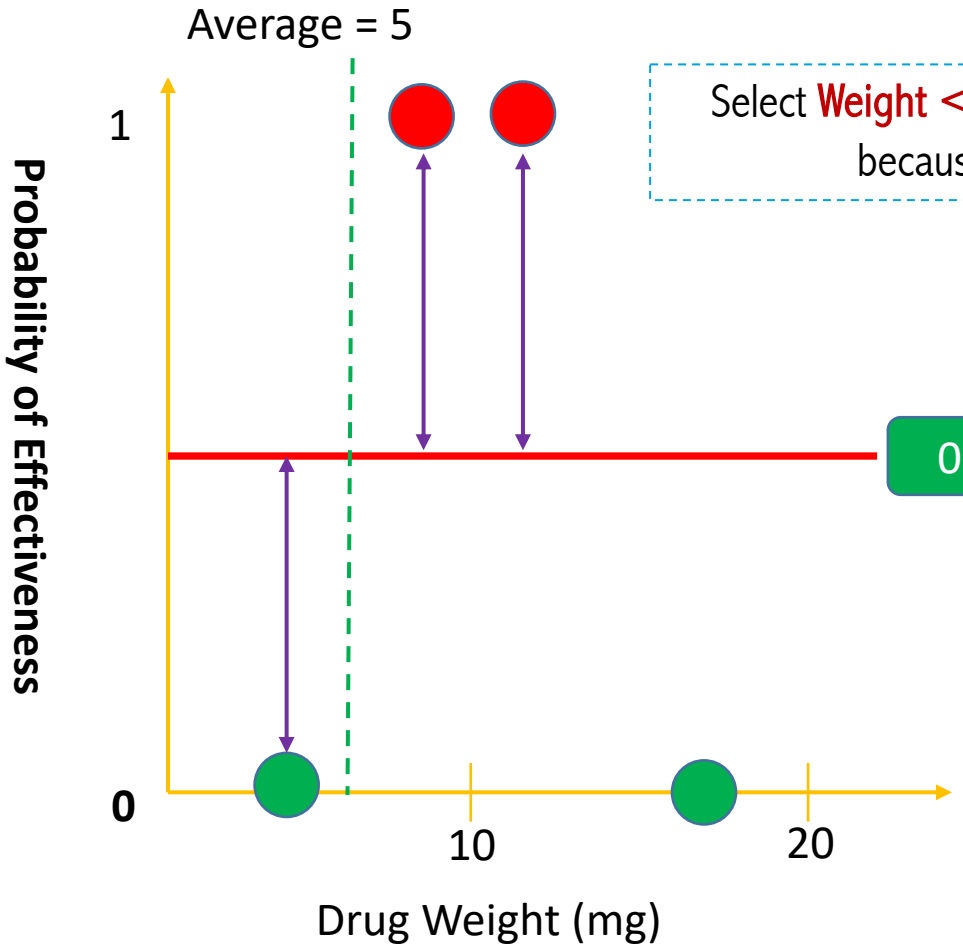




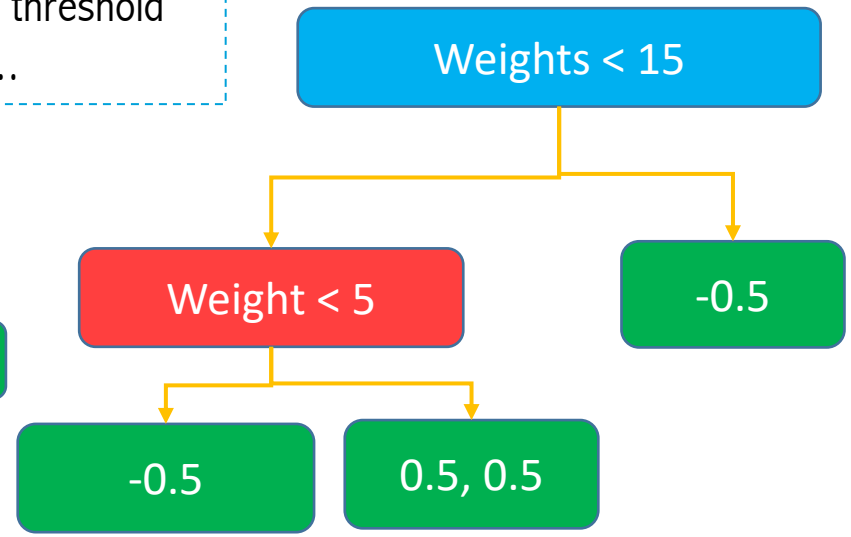
Gain = 1 + 2 – 0.33 = 2.66

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

XGBoost for Classification



Select **Weight < 5** is threshold because ....



Giả sử quy định depth level = 2, dừng xây dựng Tree

How to estimate the minimum number of Residuals in each leaf
=> **XGBoot Cover**

By default: Mininmum XGBoot Cover is set to 1

What is a Cover

Similarity Score for Classification:

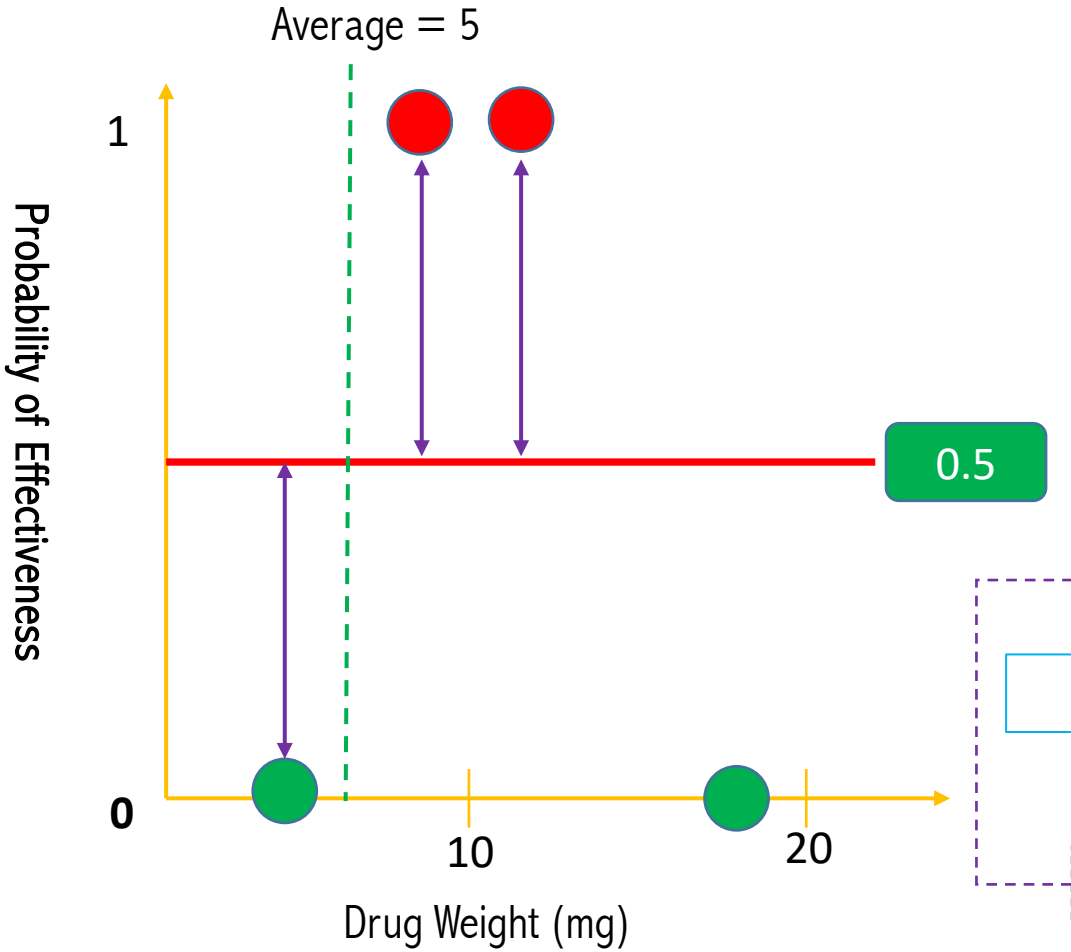
$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\sum[\text{previous probability}_i \times (1 - \text{previous probability}_i)] + \lambda}$$

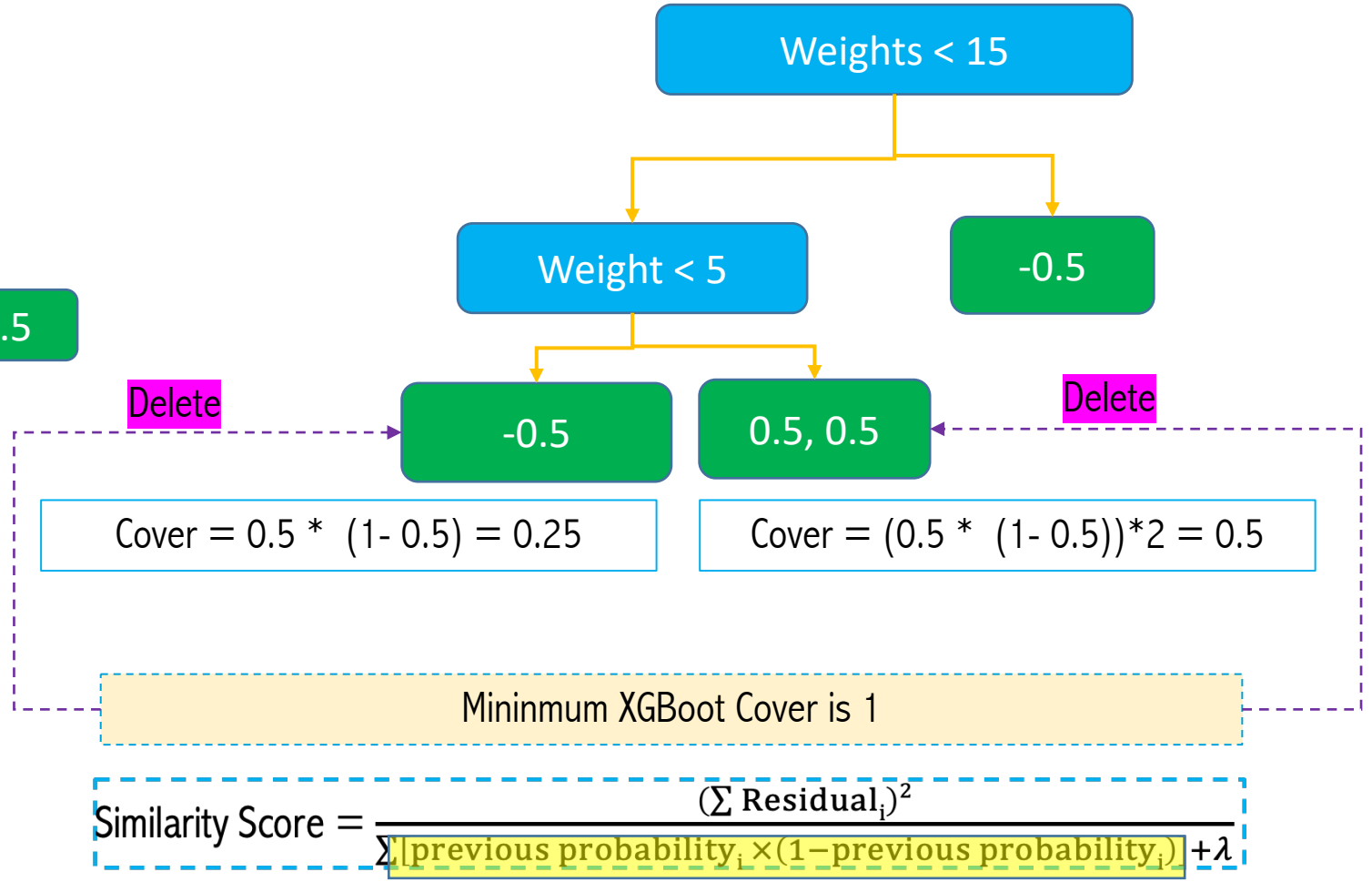
Cover

Similarity Score for Prediction:

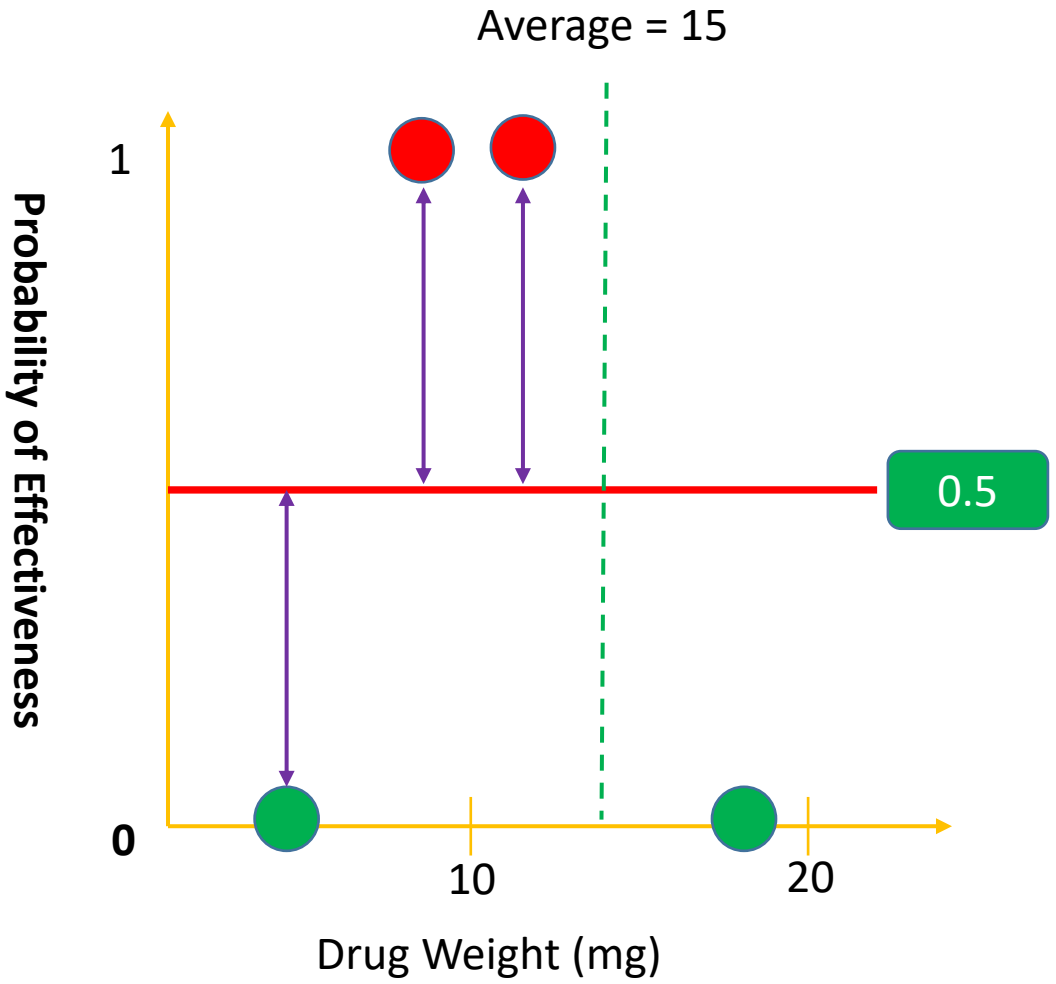
$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\text{number of residual} + \lambda}$$

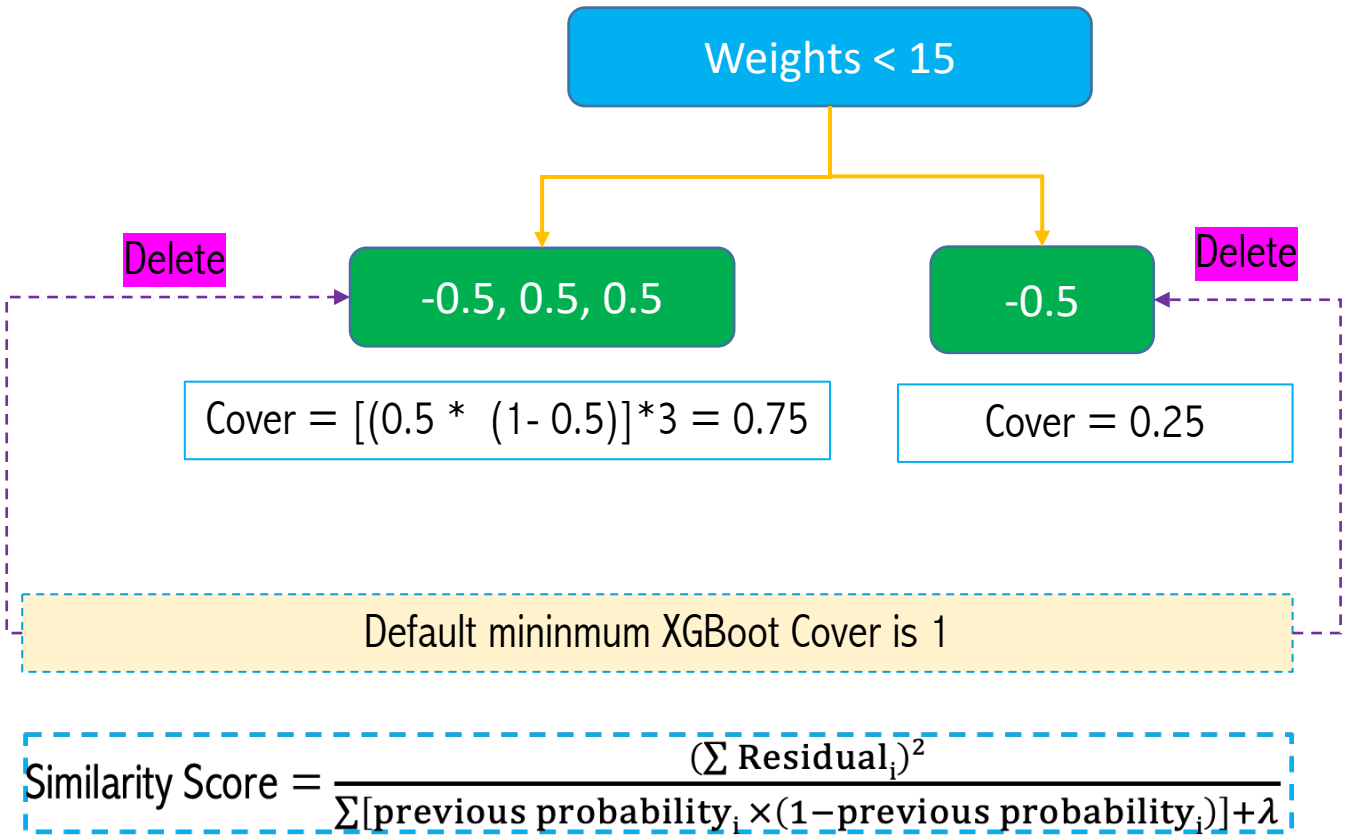
What is a Cover



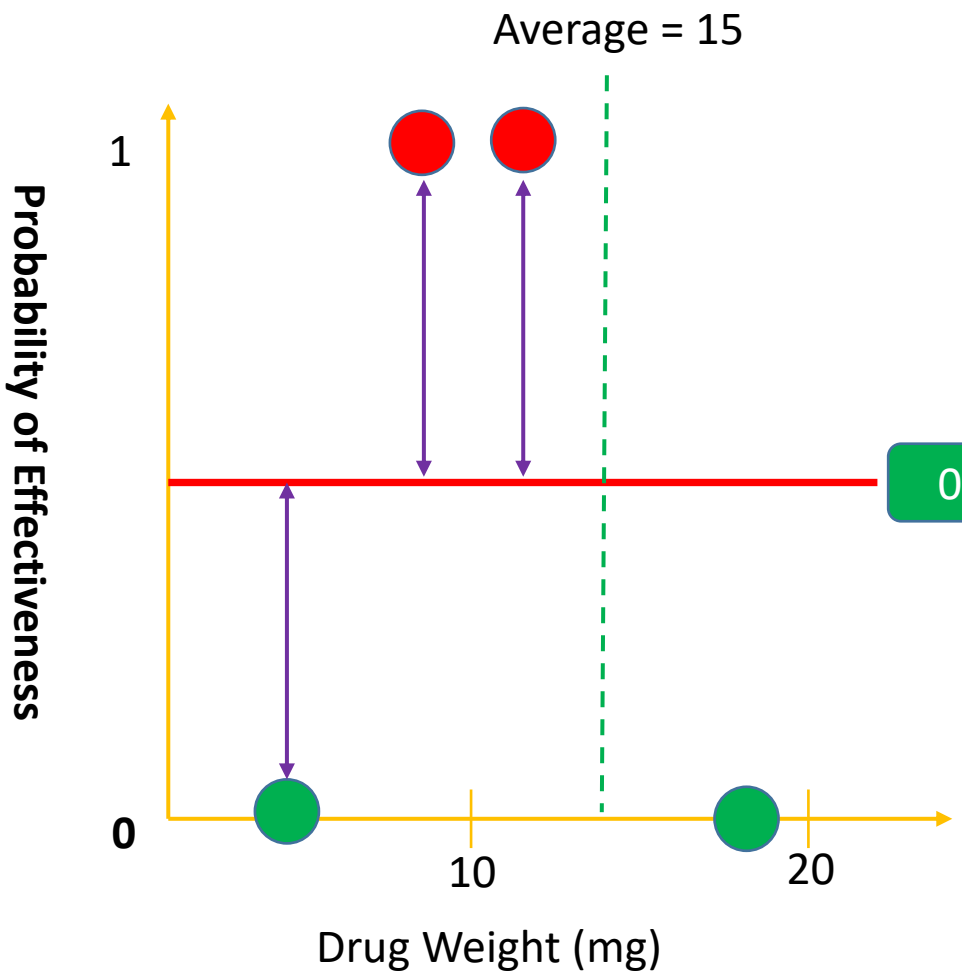


Xgboost for Classification





Xgboost for Classification



Keep this node

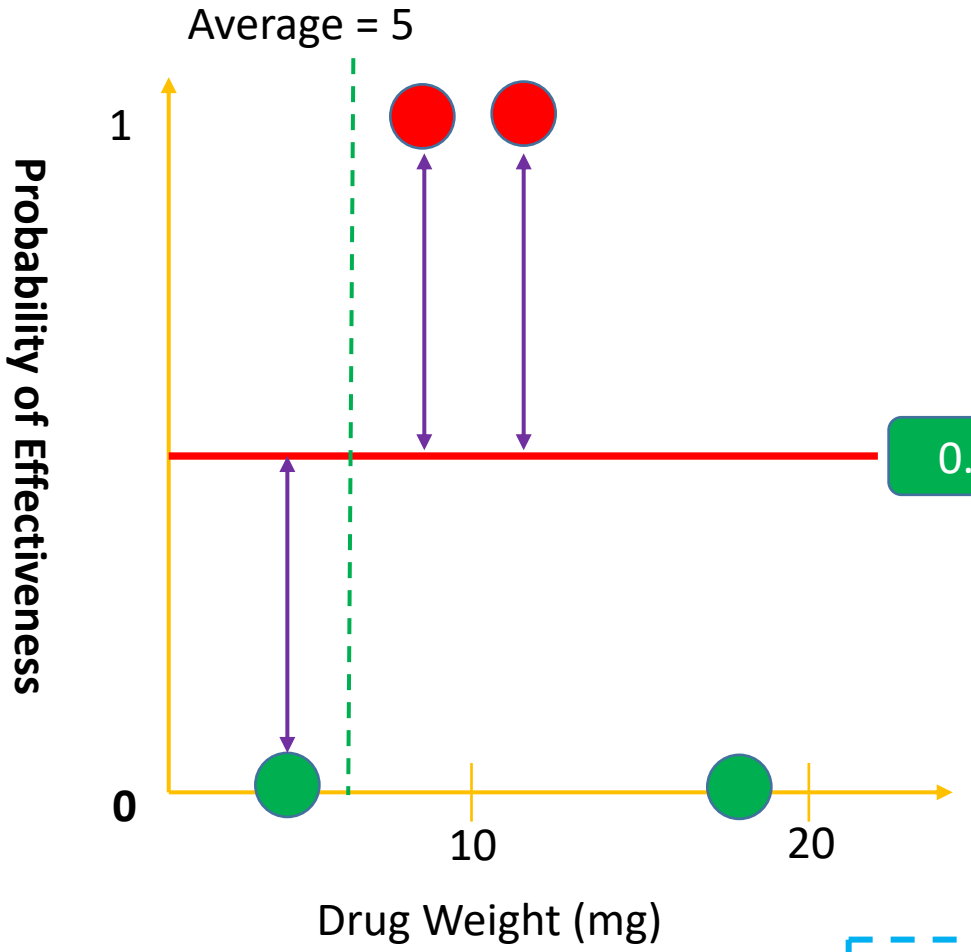
-0.5, 0.5, 0.5, -0.5

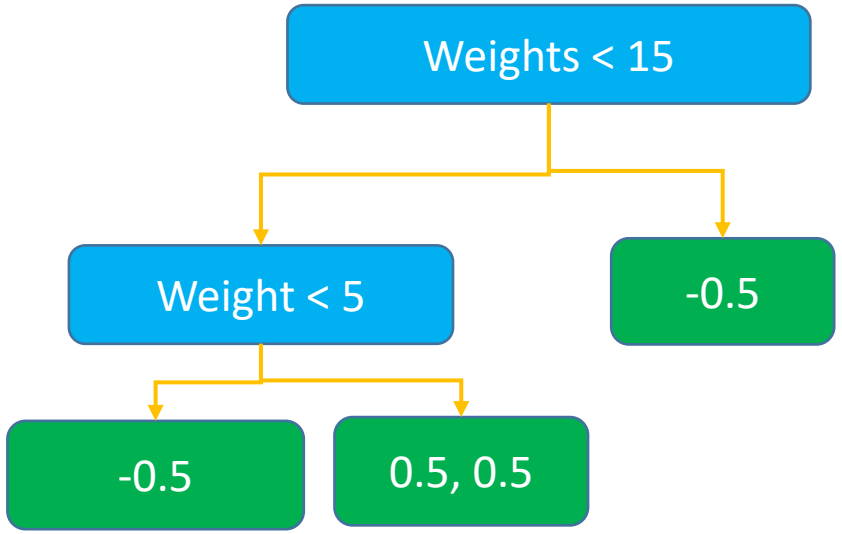
Cover = [(0.5 * (1- 0.5)]*4 = 1

Default mininmum XGBoot Cover is 1

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

How to predict the value





$$\text{Output Value} = \frac{(\sum \text{Residual}_i)}{\sum[\text{previous probability}_i \times (1 - \text{previous probability}_i)] + \lambda}$$

How to predict the value

Drug Weight	Drug Effectiveness
	No
	Yes
	Yes
	No

Initial prediction is that the probability of drug effective is 50%

2 Yes and 2 No => Probablity Yes = 2/4 = 1/2 = 0.5

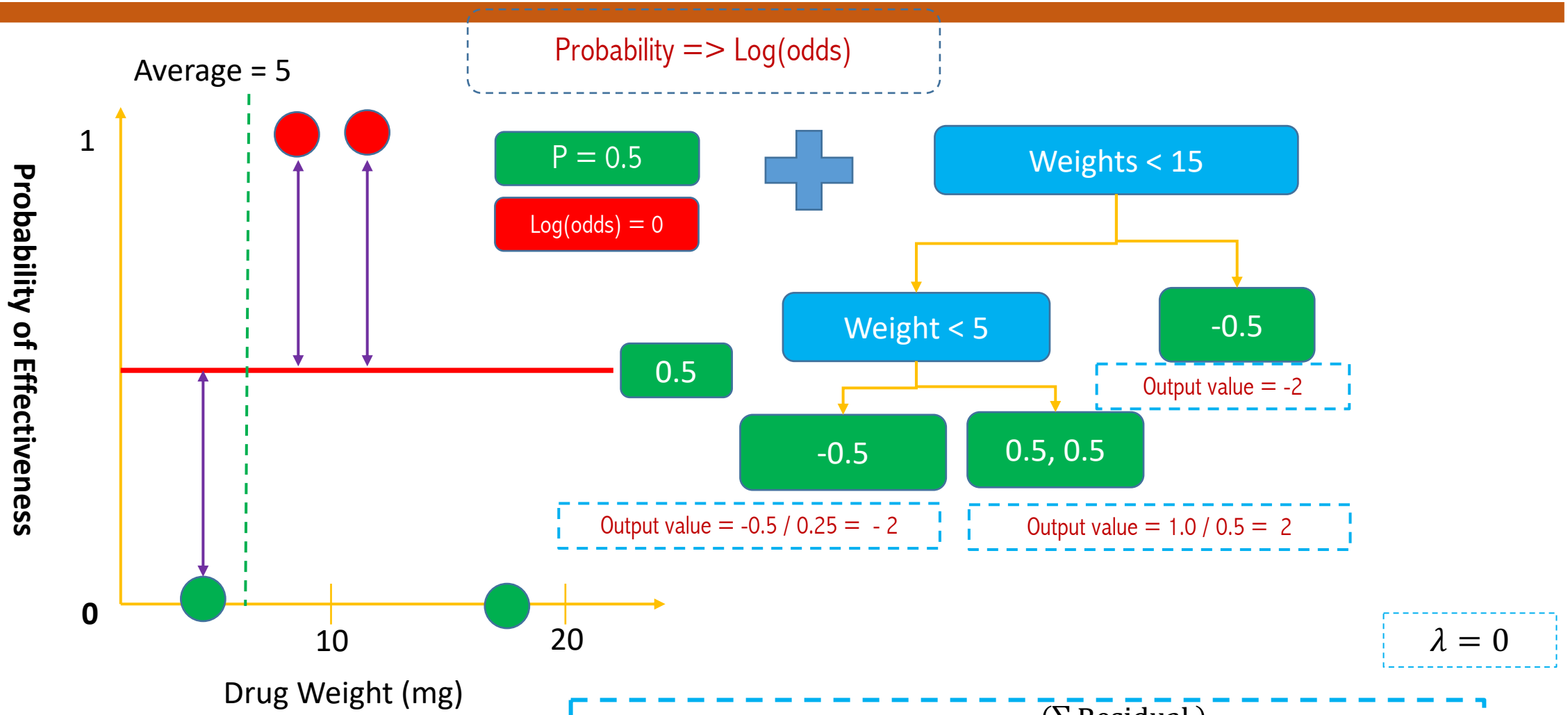
$$\text{Log(odds)} = \log\left(\frac{\text{Probablity Yes}}{\text{Probablity No}}\right) = 0$$

In XGBoost (or Gradient Boost), the initial prediction is that the log(odds)

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

$$\text{Probability of Drug Effectiveness} = \frac{e^{0}}{1+e^{0}} = 0.5$$

How to predict the value



Tranformation formular for getting value at a leaf.

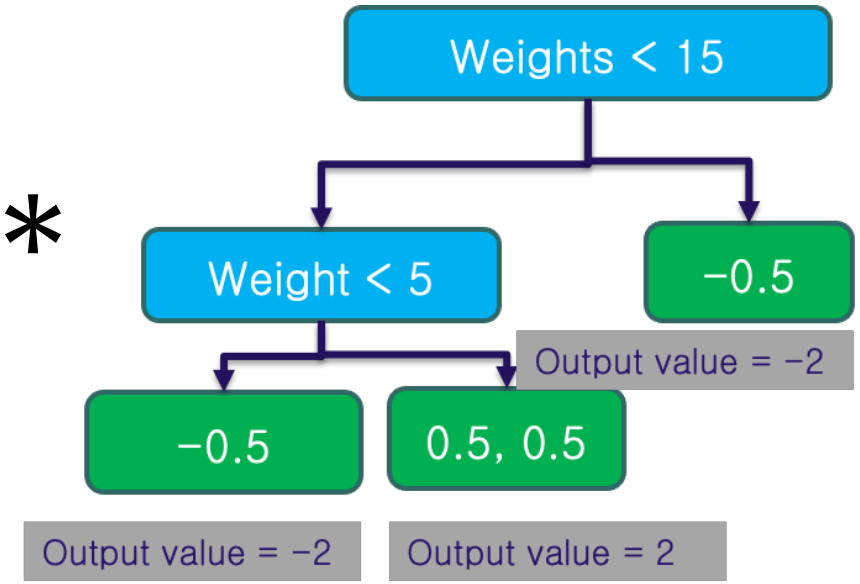
How to predict the value

Probability => Log(odds)

P = 0.5

Log(odds) = 0

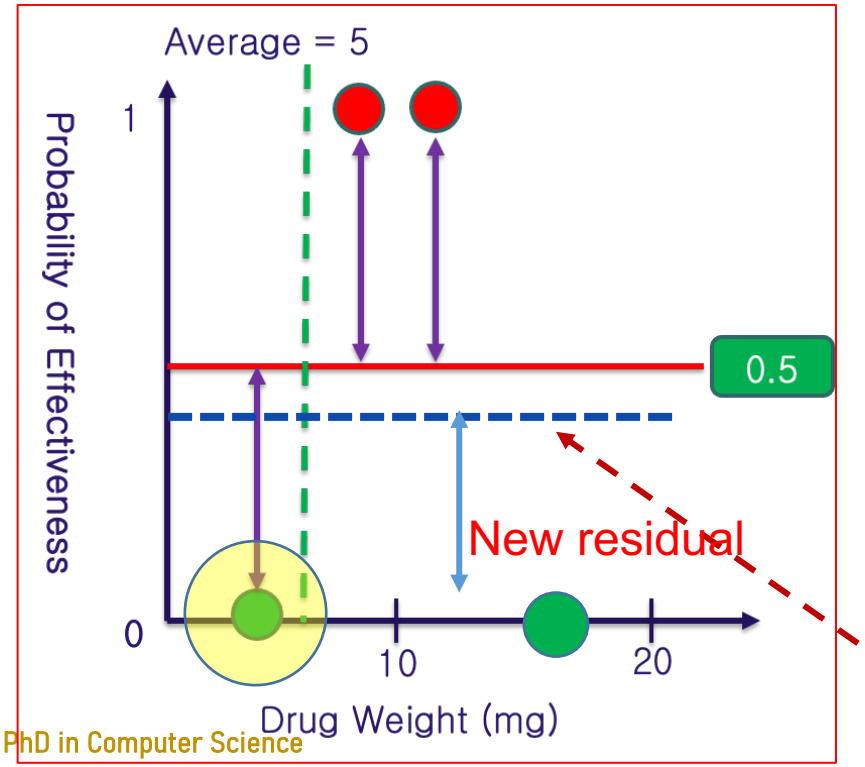
α *



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

$Log(\frac{p}{1-p})$ = log(odds)

α = 0.3



Prediction = 0 + 0.3 * (-2) = -0.6

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

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

How to predict the value

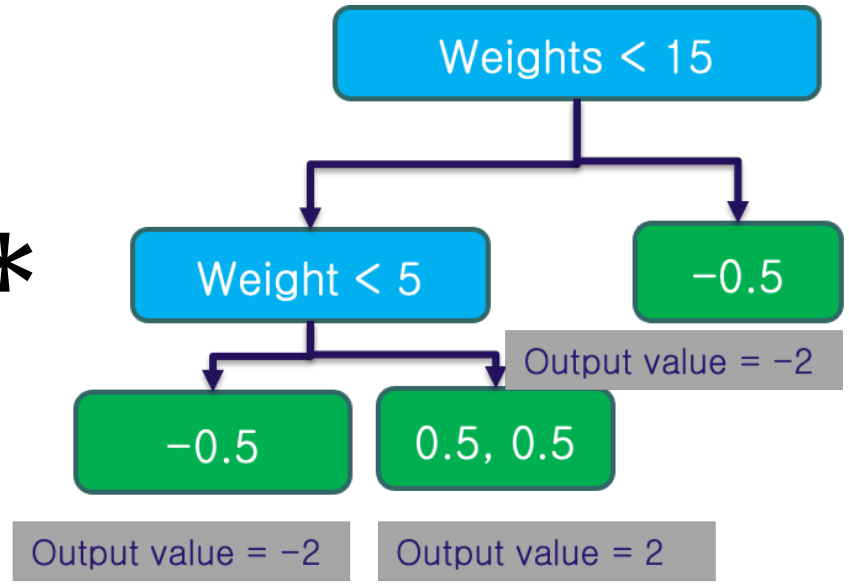
Probability => Log(odds)

Can we change P?

P = 0.5

Log(odds) = 0

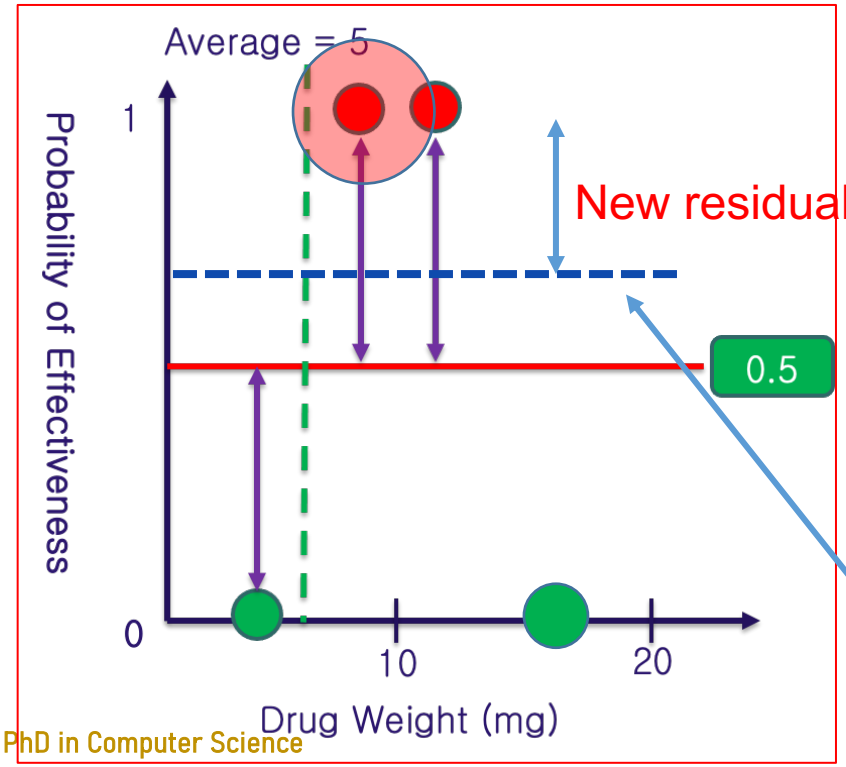
α *



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

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

α = 0.3



Log(odds) = Prediction = 0 + 0.3 * (2) = 0.6

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

$$\text{Probability} = \frac{e^{0.6}}{1+e^{0.6}} = 0.65$$

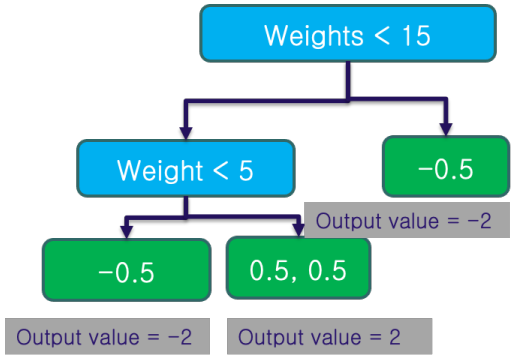
Build 2nd Tree

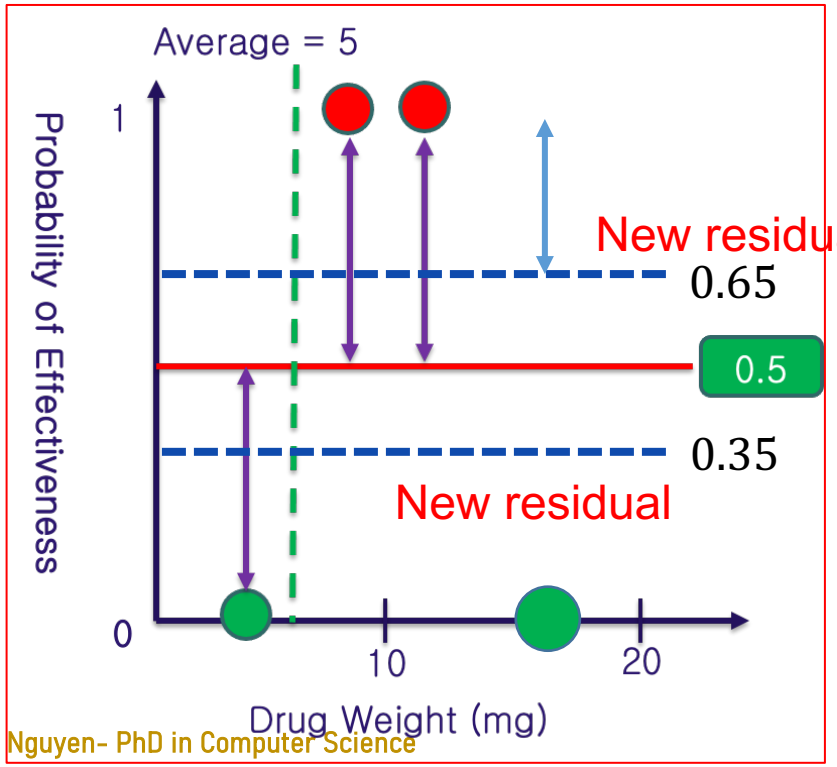
Probability => Log(odds)

P = 0.5

Log(odds) = 0

$\propto *$





$\propto *$

-0.35, 0.35, 0.35, -0.35

$$\text{Similarity Score} = \frac{(-0.35 + 0.35 + 0.35 - 0.35)^2}{0.35\times(1-0.35)+0.65\times(1-0.65)+0.65\times(1-0.65)+0.35\times(1-0.35)}$$

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

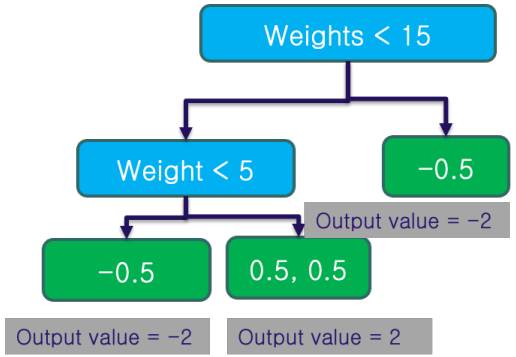
Build 2nd Tree

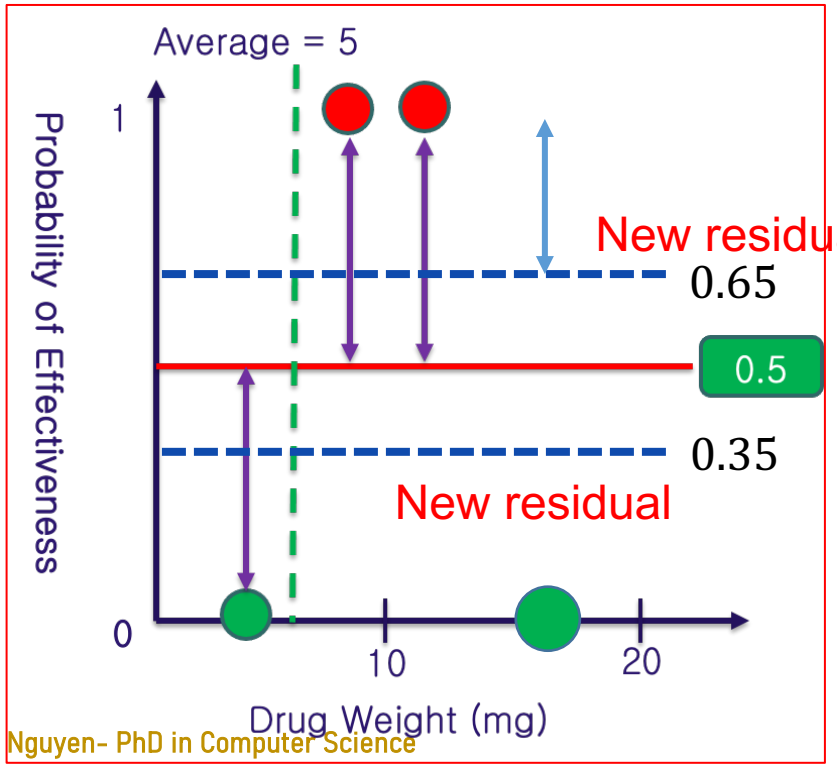
Probability => Log(odds)

P = 0.5

Log(odds) = 0

$\propto *$





$\propto *$

-0.35, 0.35, 0.35, -0.35

$$\text{Output Score} = \frac{(-0.35 + 0.35 + 0.35 - 0.35)}{0.35\times(1-0.35)+0.65\times(1-0.65)+0.65\times(1-0.65)+0.35\times(1-0.35)+\lambda}$$

$$\text{Output Score} = \frac{(\sum \text{Residual}_i)}{\sum[\text{previous probability}_i \times(1-\text{previous probability}_i)]+\lambda}$$

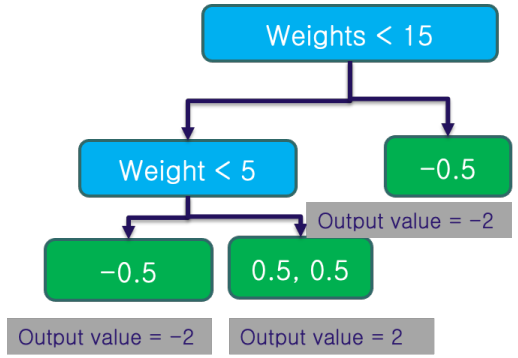
Build 2nd Tree

Probability => Log(odds)

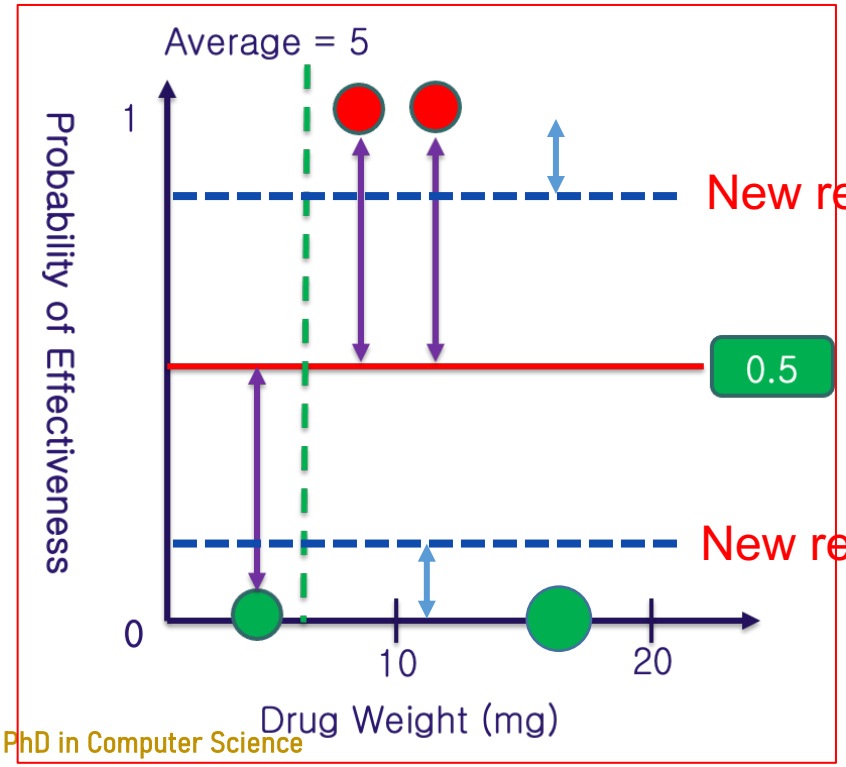
P = 0.5

Log(odds) = 0

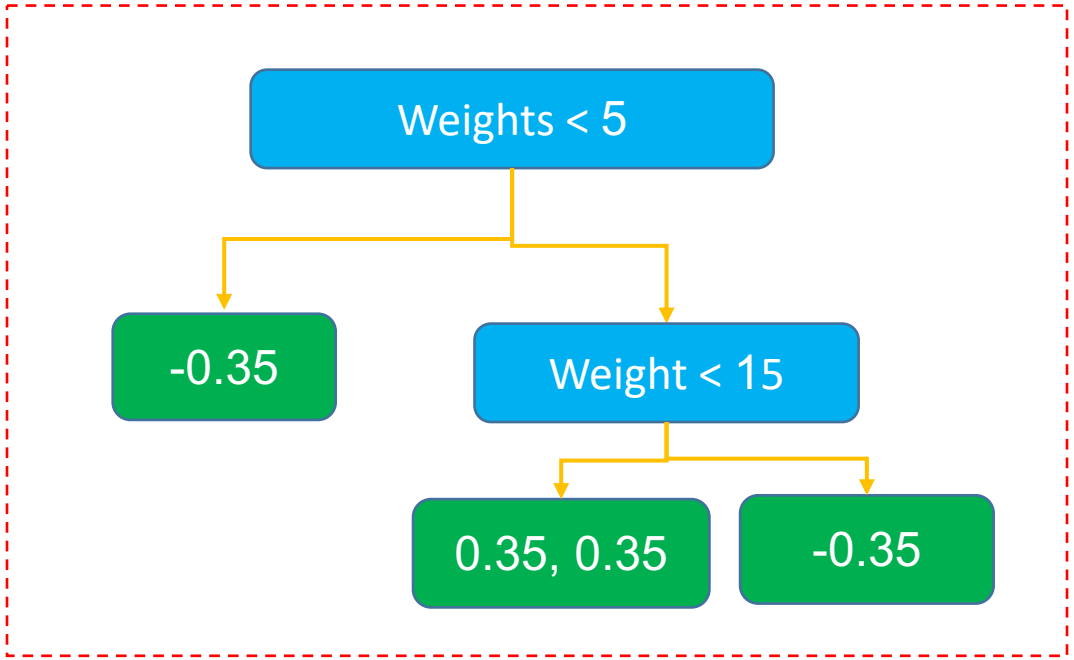
$+ \propto *$



$+$



$\propto *$



XGBoost Review Questions

1. When do you stop to build the Tree

2. What's happen when $\lambda > 0$

3. How to select γ?

Case if γ is very large?

Case if γ is very small?

4. Start with 0 and check CV error rate. If you see test error is much larger than train error. What will you do? Decrease or increase γ

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

XGBoost Cheat sheet

Parameter	Range	Default Value	Value	Effect	Bias	Variance
eta	[0, 1]	0,3	Decreasing the value	Less contribution of each tree to final result	+	-
			Increasing the value	More contribution of each tree to final result	-	+
gamma	[0, ∞]	0	Decreasing the value	Less pruning: trees tend to be bigger	-	+
			Increasing the value	More pruning: trees tend to be smaller	+	-
max_depth	[0, ∞]	6	Decreasing the value	Smaller trees (except value 0)	+	-
			Increasing the value	Bigger trees	-	+
min_child_weight	[0, ∞]	1	Decreasing the value	Bigger trees	-	+
			Increasing the value	Smaller trees (except value 0)	+	-
subsample	(0, 1]	1	Decreasing the value	Smaller samples feeding each tree	+	-
			Increasing the value	Bigger samples feeding each tree	-	+

1. Sự khác biệt giữa AdaBoost và XGBoost là gì?

2. Lý do đằng sau việc không sử dụng các stumps trong gradient boosting?

3. Làm thế nào để có thể cải độ chính xác của thuật toán gradient boossting ?

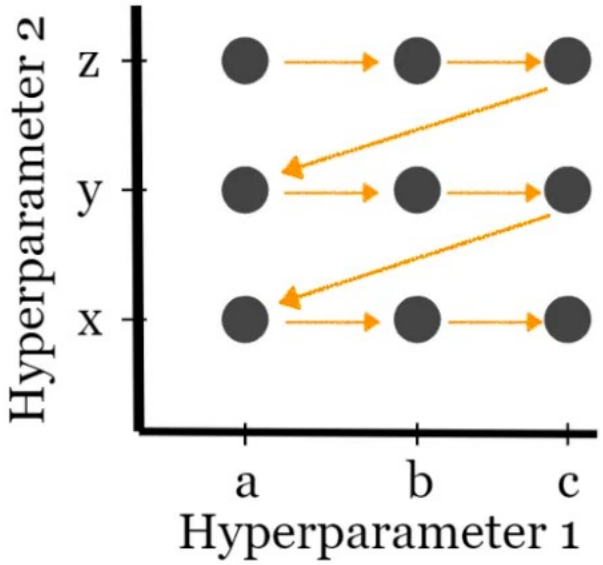
4. Sự khác biệt giữa Gradient Boosting và XGBoost là gì?

5. Có bao nhiêu loại boosting algorithm?

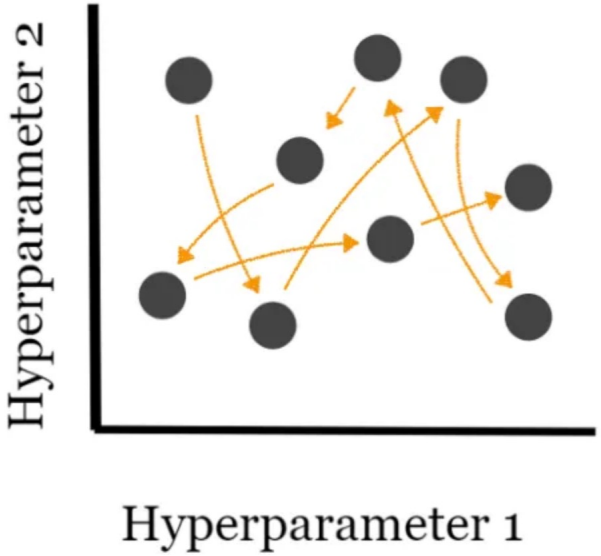
6. Sự khác biệt giữa Random Forest và XGBoost là gì?

7. Làm cách nào để điều chỉnh siêu tham số trong XGBoost?

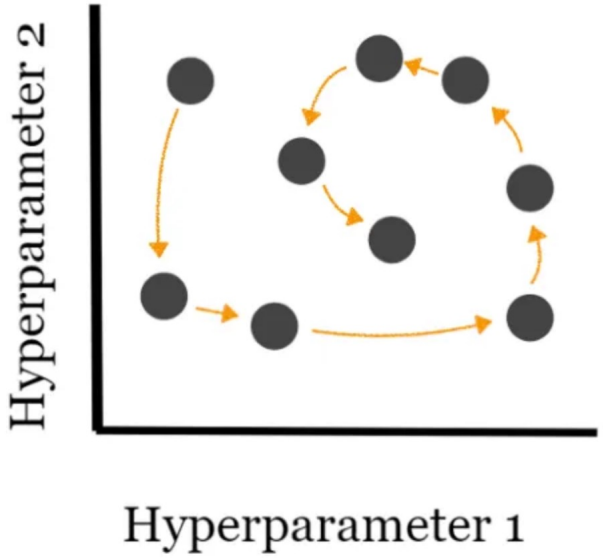
Hyperparameter optimization/tuning



Grid Search — trying out all the possible combinations



Random Search tries random combinations



Bayes Search

Other methods: **Evolutionary optimization, Early stopping-based,…**

https://towardsdatascience.com/a-practical-introduction-to-grid-search-random-search-and-bayes-search-d5580b1d941d


Thank you