Từ trồng cây gây rừng đến XGBoost

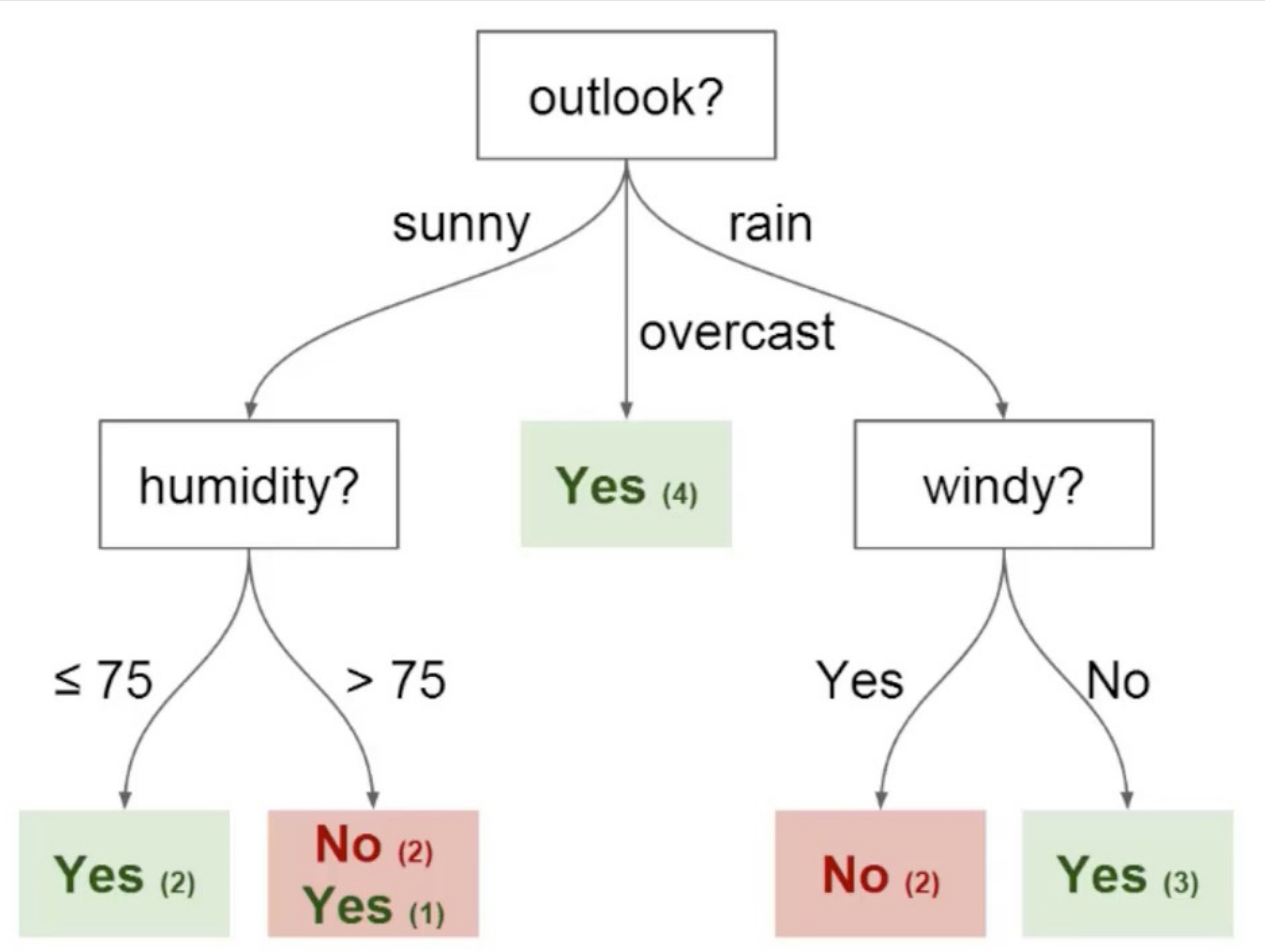
Mì AI

Decision Tree

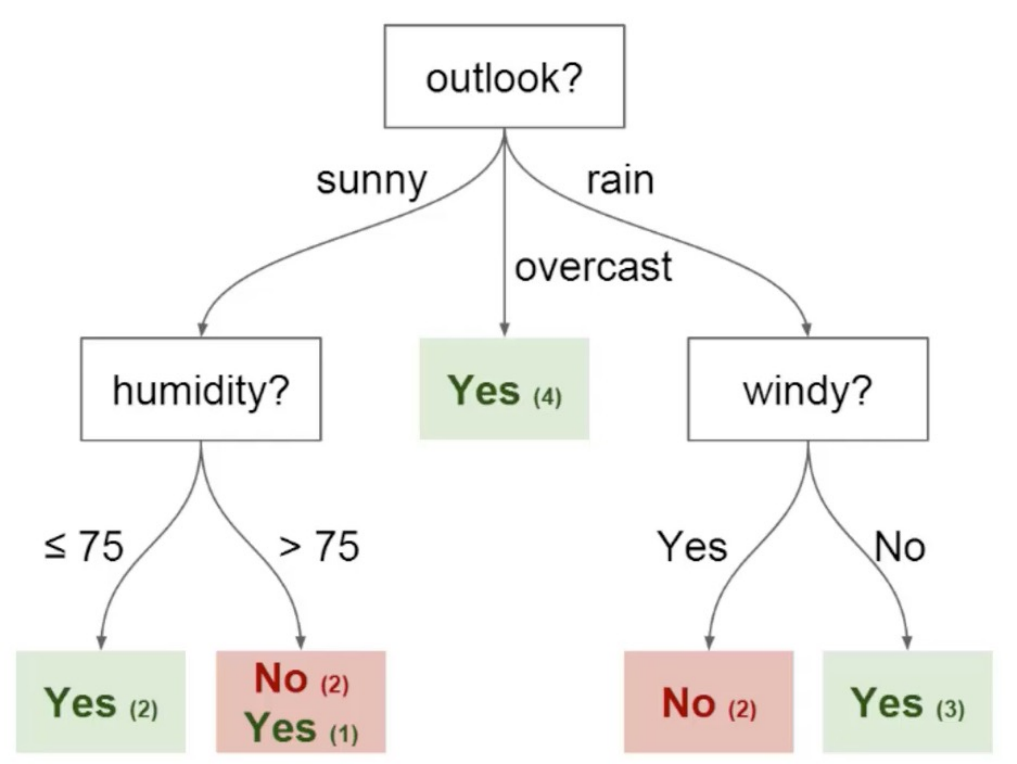
- *A Decision Tree is a supervised Machine learning algorithm. It is used in both classification and regression algorithms*.

- *Decision Trees usually implement exactly the human thinking ability while making a decision*, so it is easy to understand.

Decision Tree

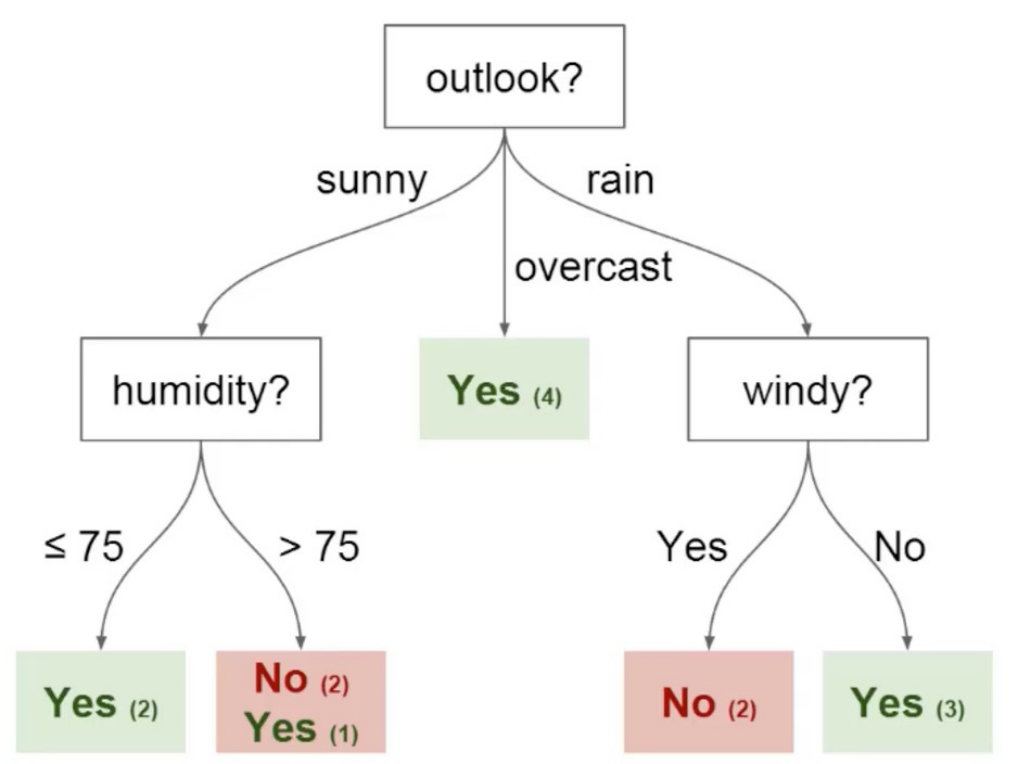


Decision Tree



- **Nodes:** It is The point where the tree splits according to the value of some attribute/feature of the dataset
- **Edges**: It directs the outcome of a split to the next node.

Decision Tree

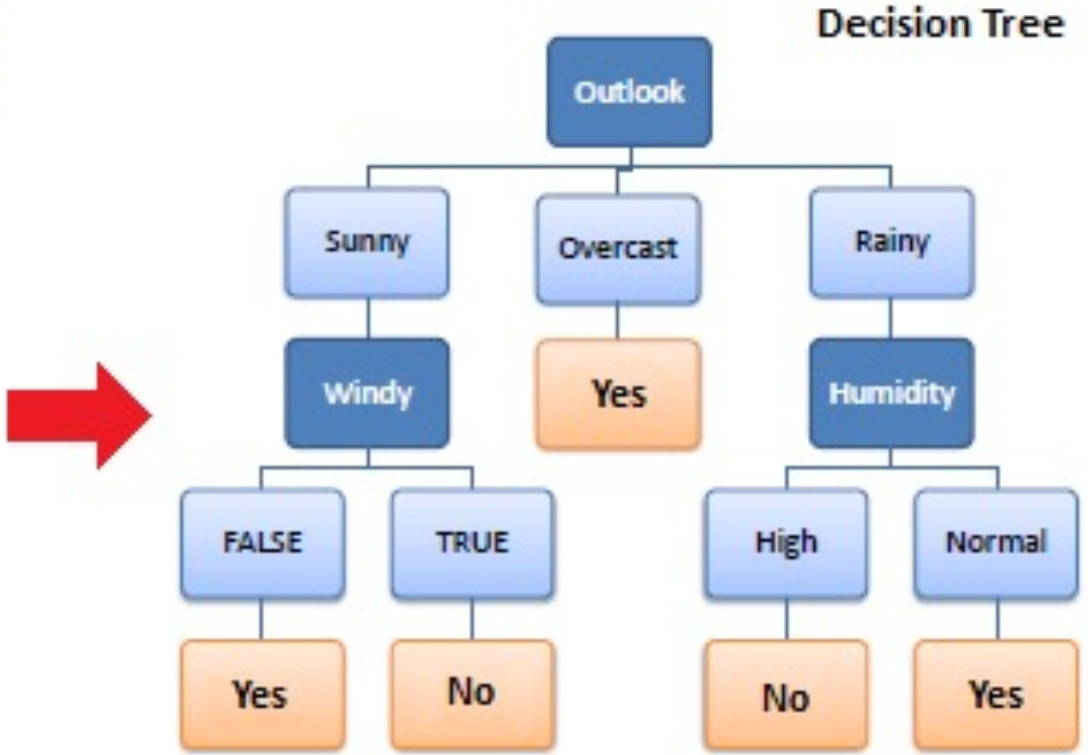


- **Root:** This is the node where the first split takes place
- **Leaves:** These are the terminal nodes that predict the outcome of the decision tree.

Decision Tree

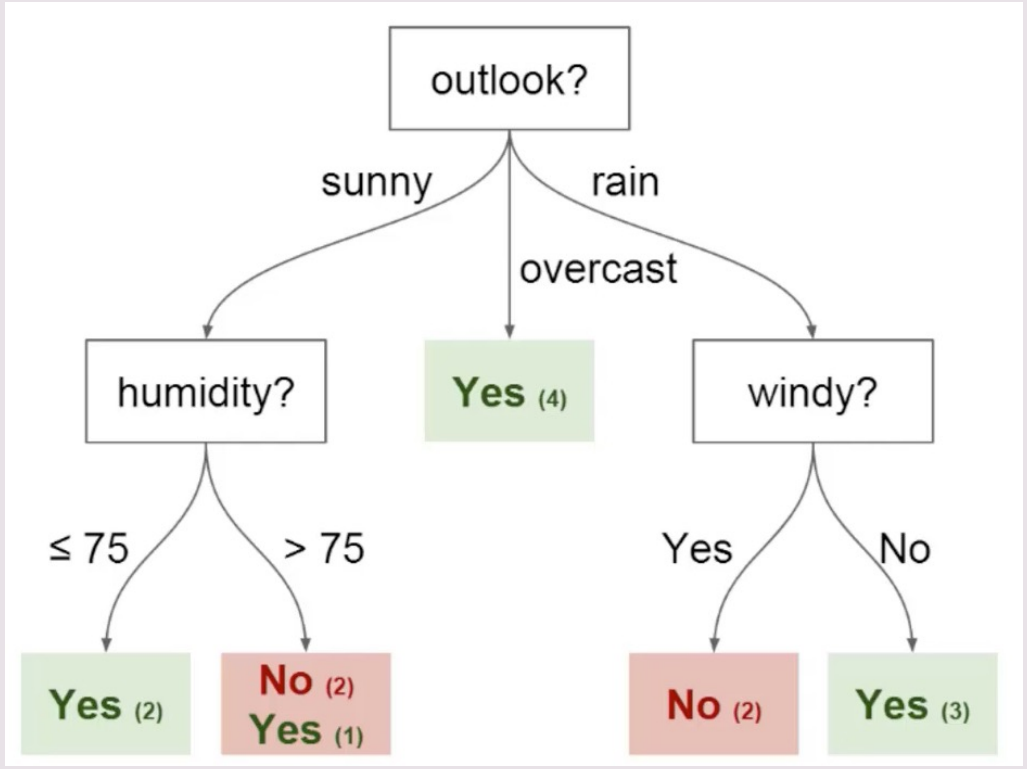
Predictors | Target

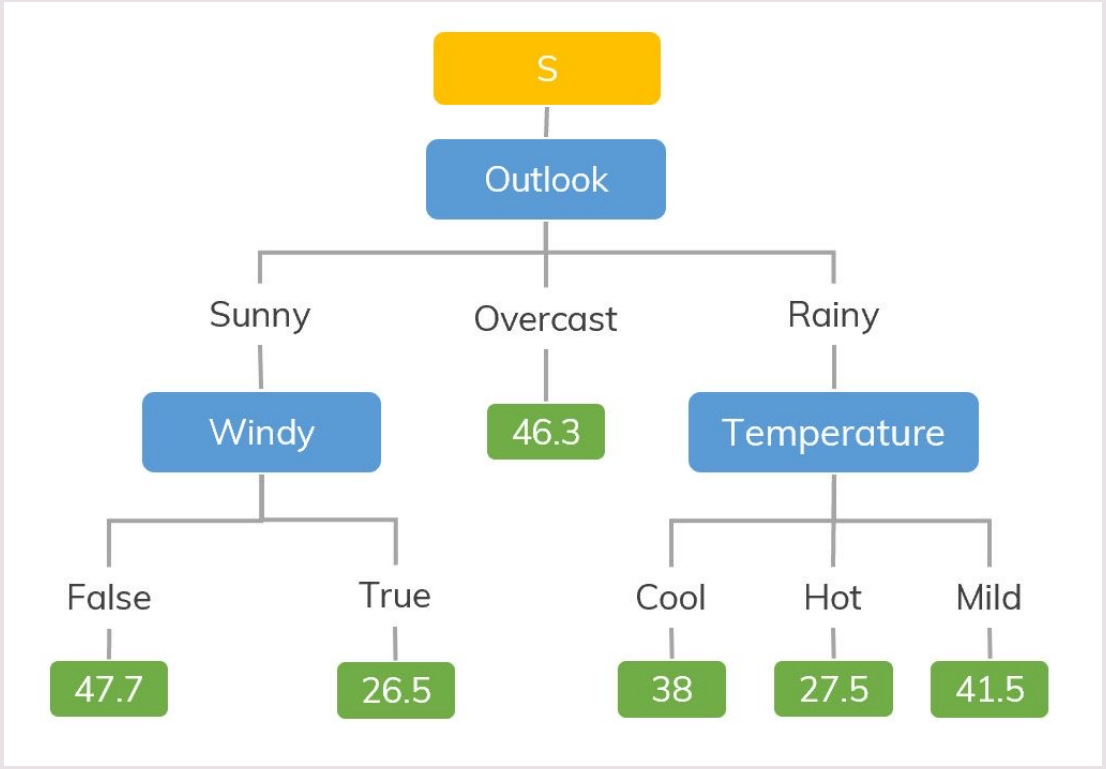
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



Rainy	Cool	Normal	False	???

Decision Tree Type





???

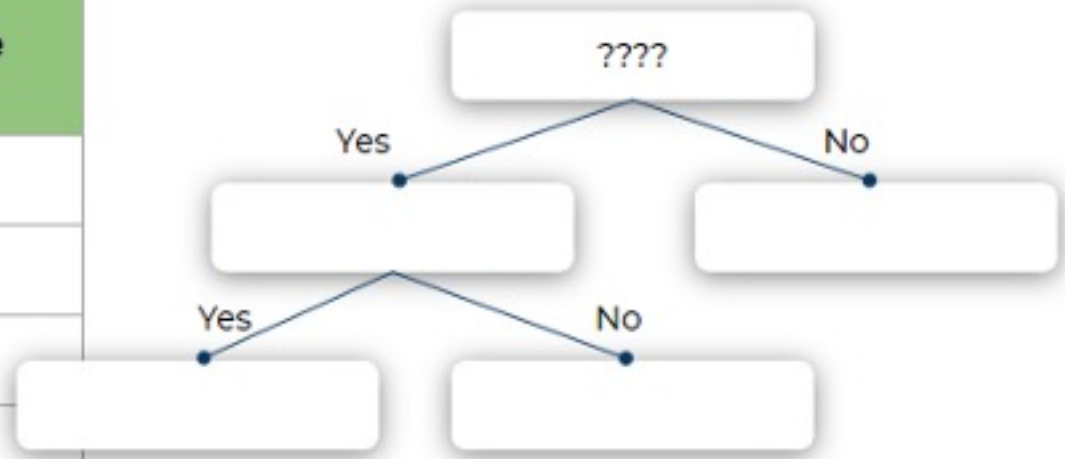
Classification Tree

- While building a Decision tree, the main thing is to select the best attribute from the total features list of the dataset for the root node as well as for sub-nodes. The selection of best attributes is being achieved with the help of a technique known as the Attribute selection measure (ASM).

- Methods:
  - Information Gain (ID3)
  - GINI

GINI

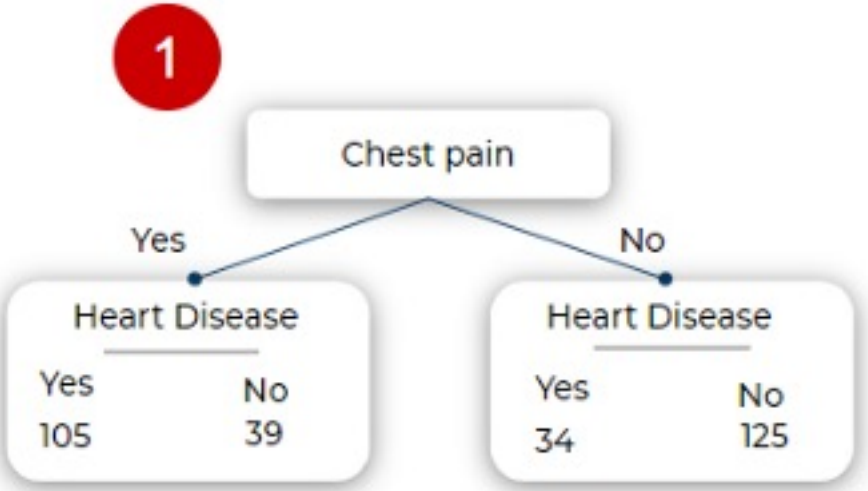
Chest pain	Good blood circulation	Blocked arteries	Heart disease
yes	no	yes	no
yes	yes	yes	yes
no	no	yes	no
yes	yes	no	yes
...	...	...	...



https://www.analyticsvidhya.com/blog/2020/10/all-about-decision-tree-from-scratch-with-python-implementation/

GINI

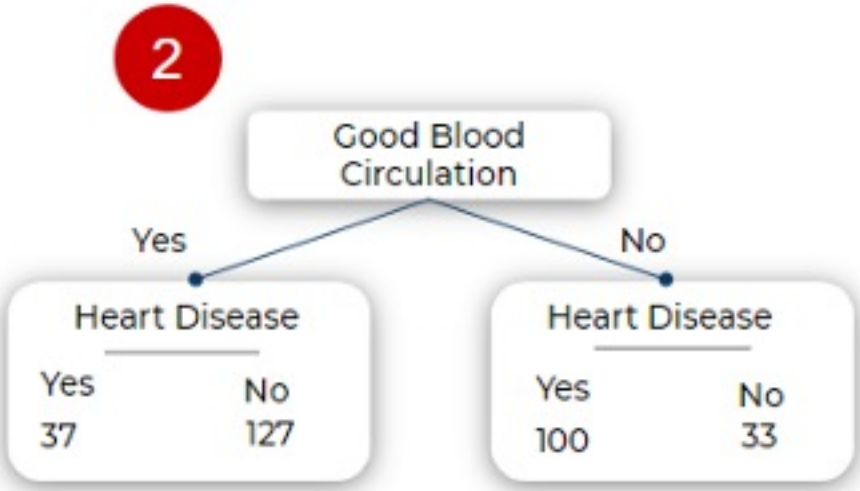
Chest pain	Good blood circulation	Blocked arteries	Heart disease
yes	no	yes	no
yes	yes	yes	yes
no	no	yes	no
yes	yes	no	yes
...	...	...	...



https://www.analyticsvidhya.com/blog/2020/10/all-about-decision-tree-from-scratch-with-python-implementation/

GINI

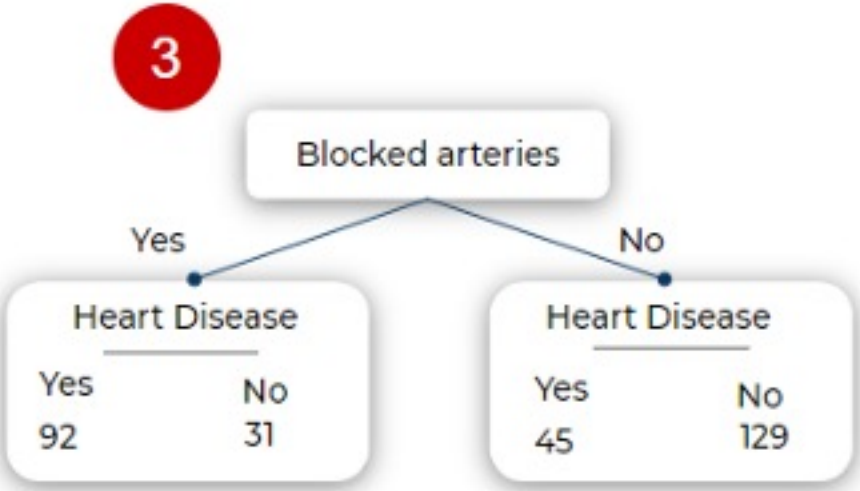
Chest pain	Good blood circulation	Blocked arteries	Heart disease
yes	no	yes	no
yes	yes	yes	yes
no	no	yes	no
yes	yes	no	yes
...	...	...	...



https://www.analyticsvidhya.com/blog/2020/10/all-about-decision-tree-from-scratch-with-python-implementation/

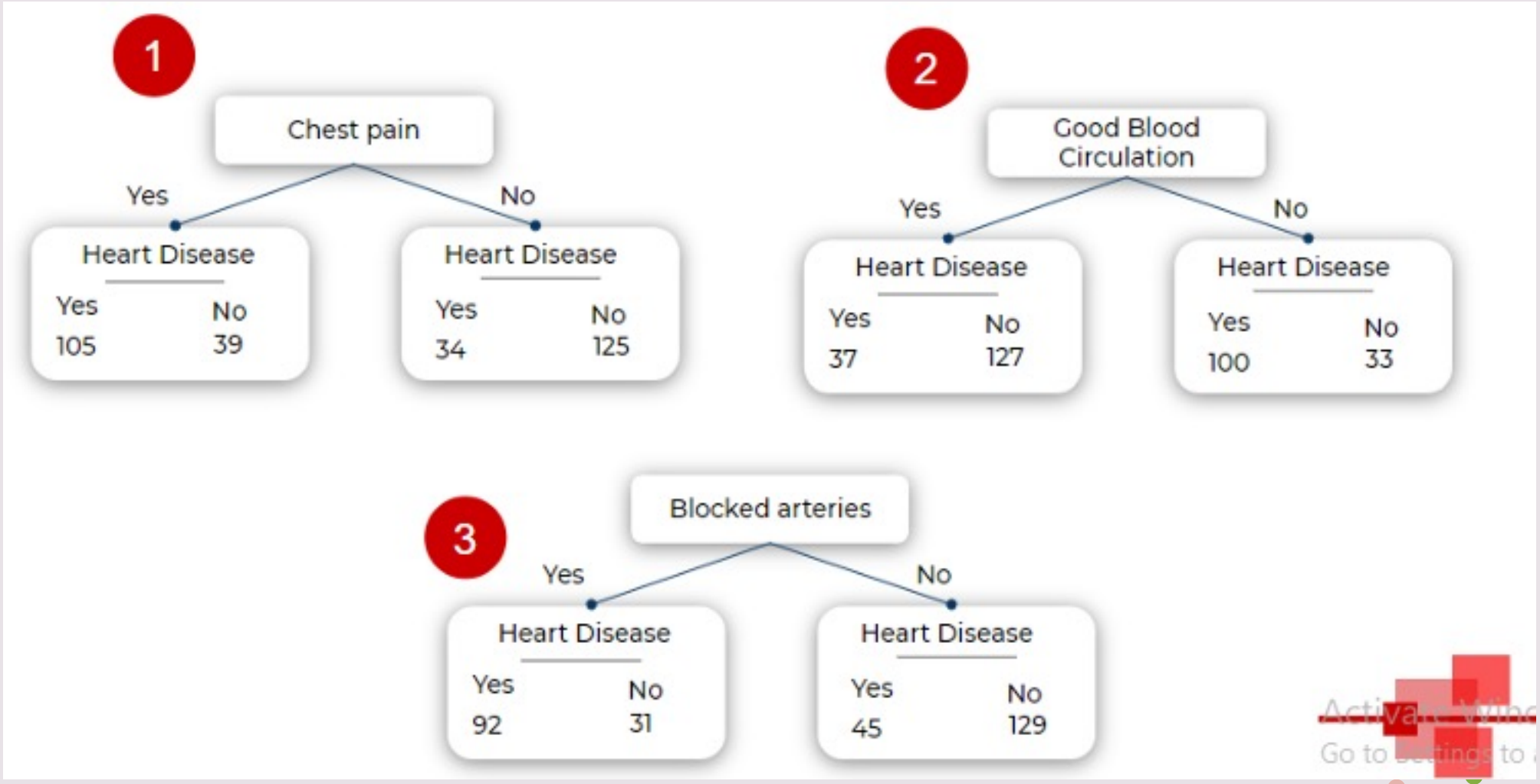
GINI

Chest pain	Good blood circulation	Blocked arteries	Heart disease
yes	no	yes	no
yes	yes	yes	yes
no	no	NA	no
yes	yes	no	yes
...	...	...	...



https://www.analyticsvidhya.com/blog/2020/10/all-about-decision-tree-from-scratch-with-python-implementation/

GINI



https://www.analyticsvidhya.com/blog/2020/10/all-about-decision-tree-from-scratch-with-python-implementation/

GINI

$$Gini\ impurity = 1 - [P(yes)]^2 - [P(no)]^2$$

$$Gini\ impurity = 1 - \left[\frac{105}{105+39}\right]^2 - \left[\frac{39}{105+39}\right]^2$$

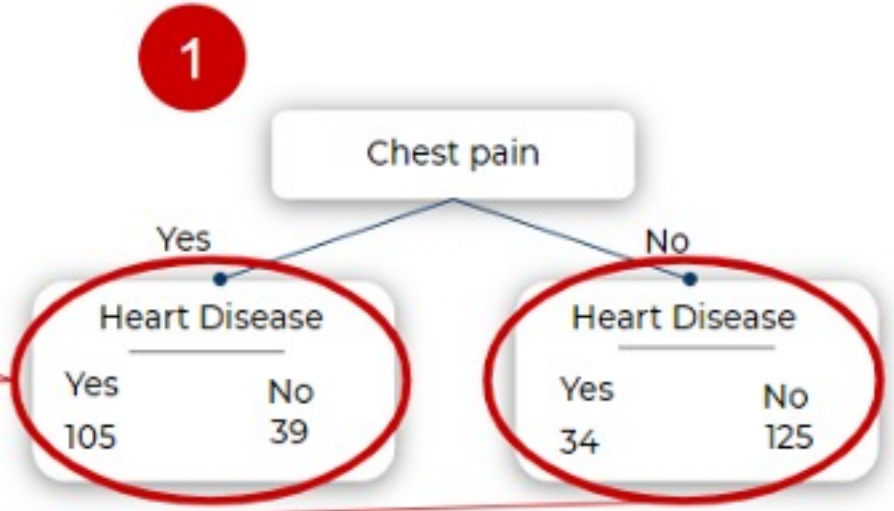
$$Gini\ impurity = 0.395$$

$$Gini\ impurity = 0.336$$

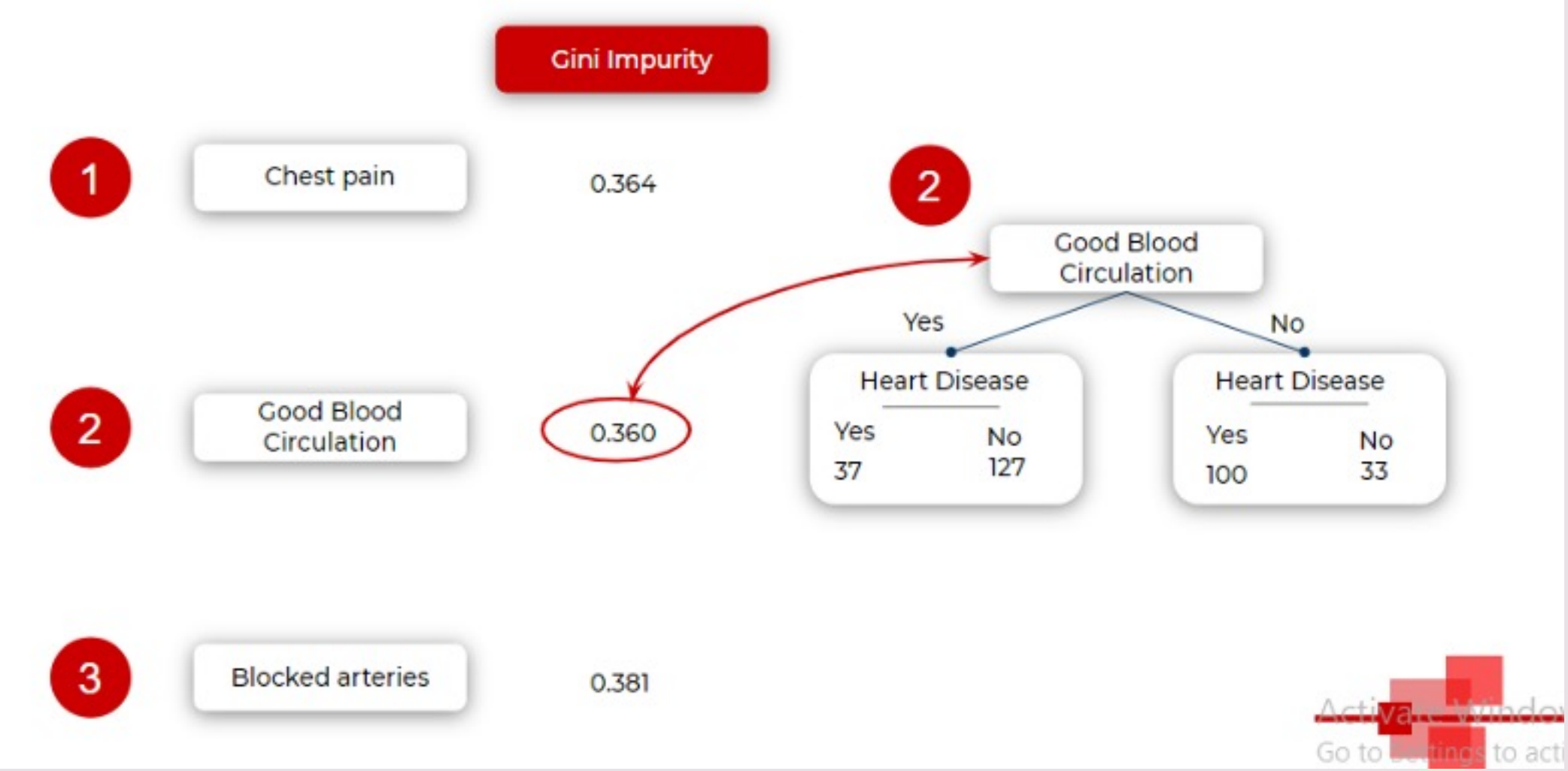
$$Gini\ impurity\ for\ chest\ pain = weighted\ avg.\ of\ leaf\ node$$

$$Gini\ impurity\ for\ chest\ pain = \left(\frac{144}{144+159}\right).0.395 + \left(\frac{159}{144+159}\right).0.336$$

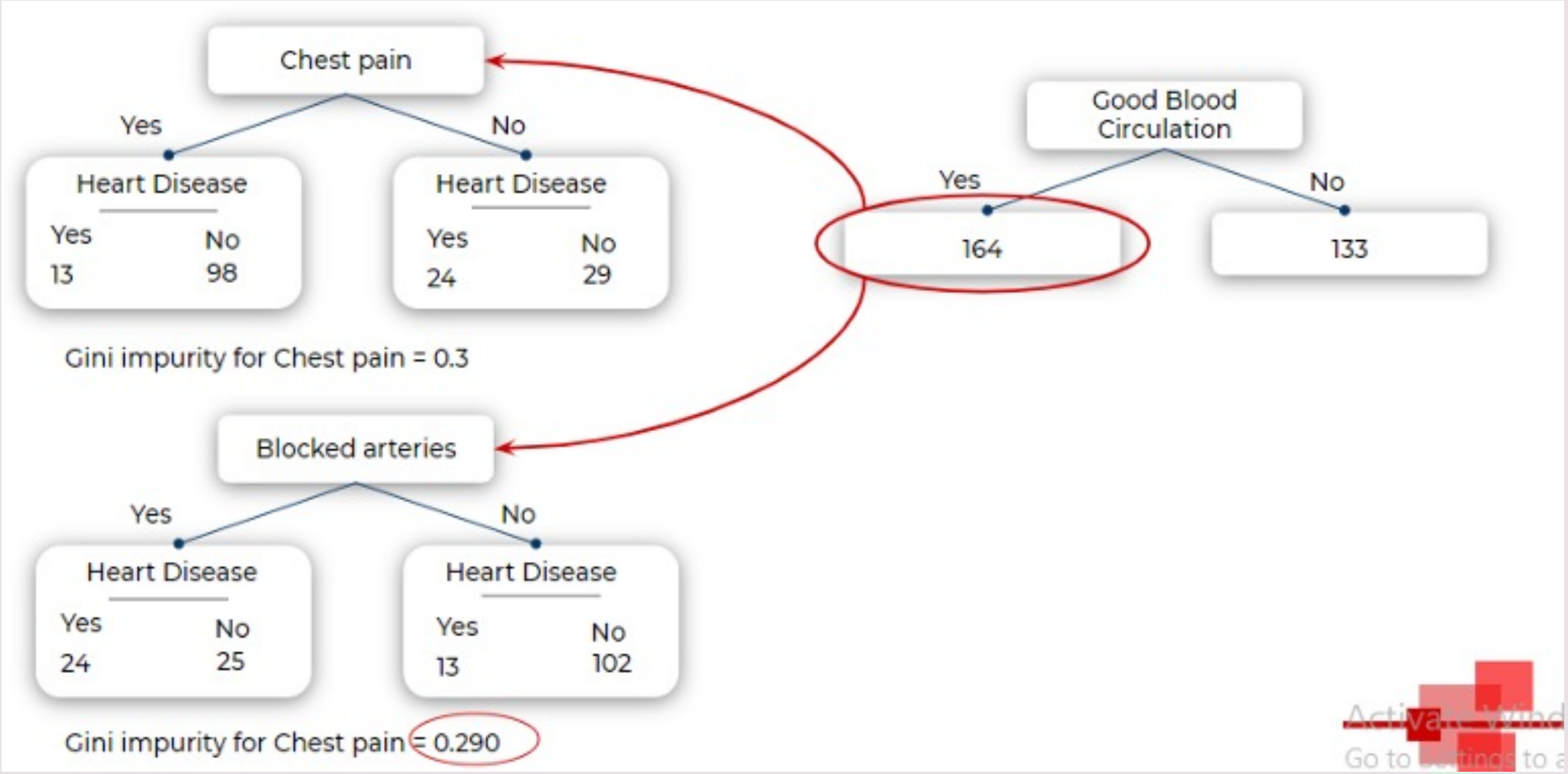
$$Gini\ impurity\ for\ chest\ pain = 0.364$$



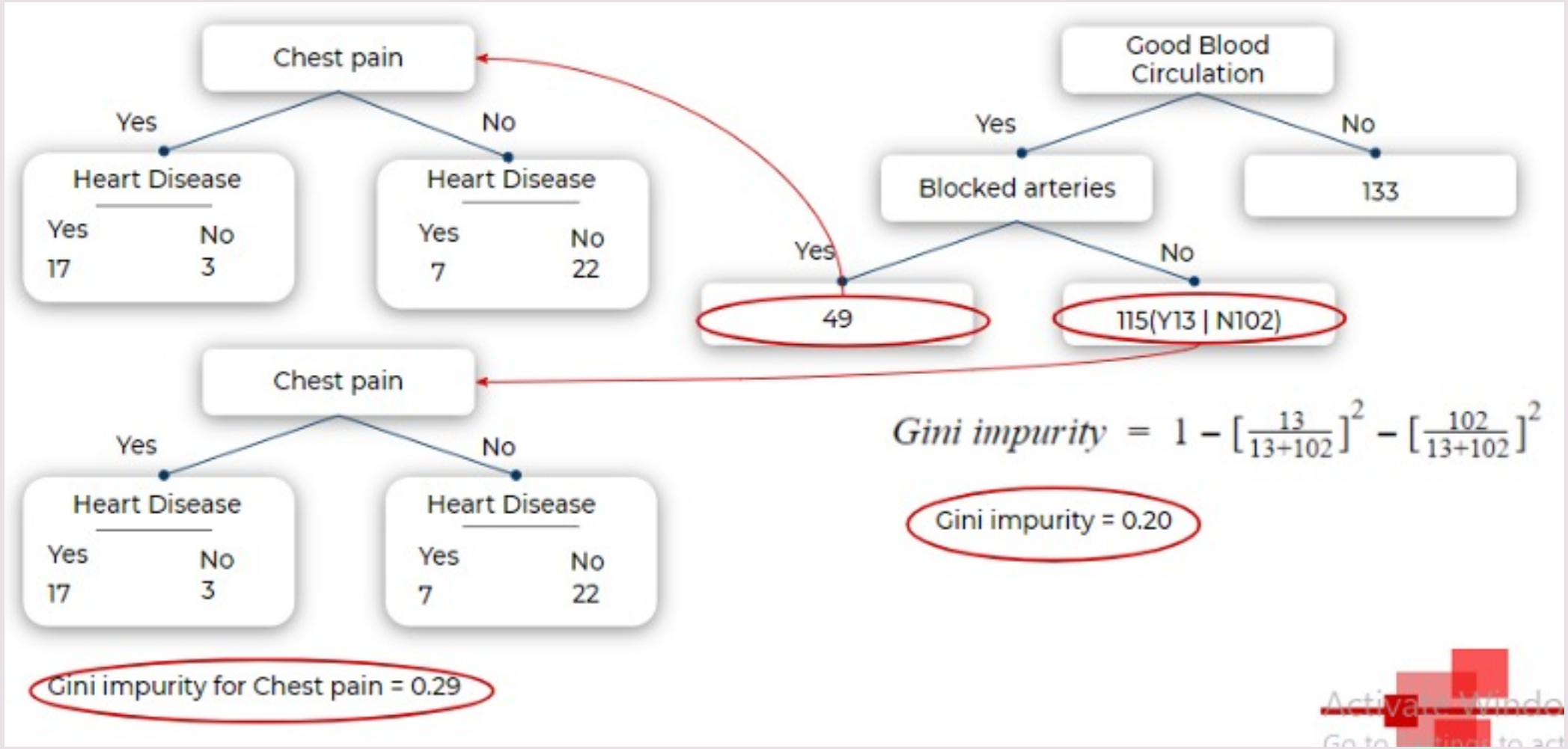
GINI


Gini Impurity
1
Chest pain
0.364
2
Good Blood Circulation
Yes
No
Heart Disease
Yes 37
No 127
Heart Disease
Yes 100
No 33
2
Good Blood Circulation
0.360
3
Blocked arteries
0.381

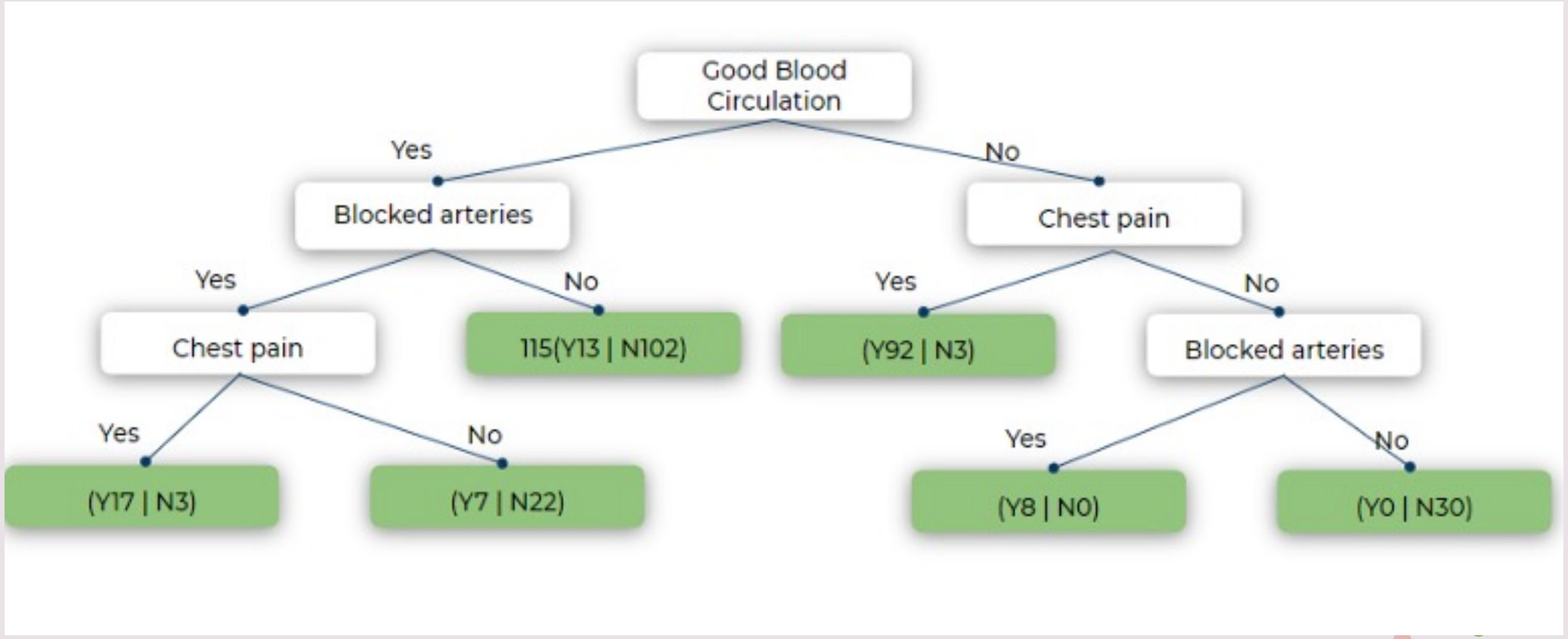
GINI


Chest pain
Yes
No
Heart Disease
Yes
No
13
98
Heart Disease
Yes
No
24
29
Gini impurity for Chest pain = 0.3
Good Blood Circulation
Yes
No
164
133
Blocked arteries
Yes
No
Heart Disease
Yes
No
24
25
Heart Disease
Yes
No
13
102
Gini impurity for Chest pain = 0.290

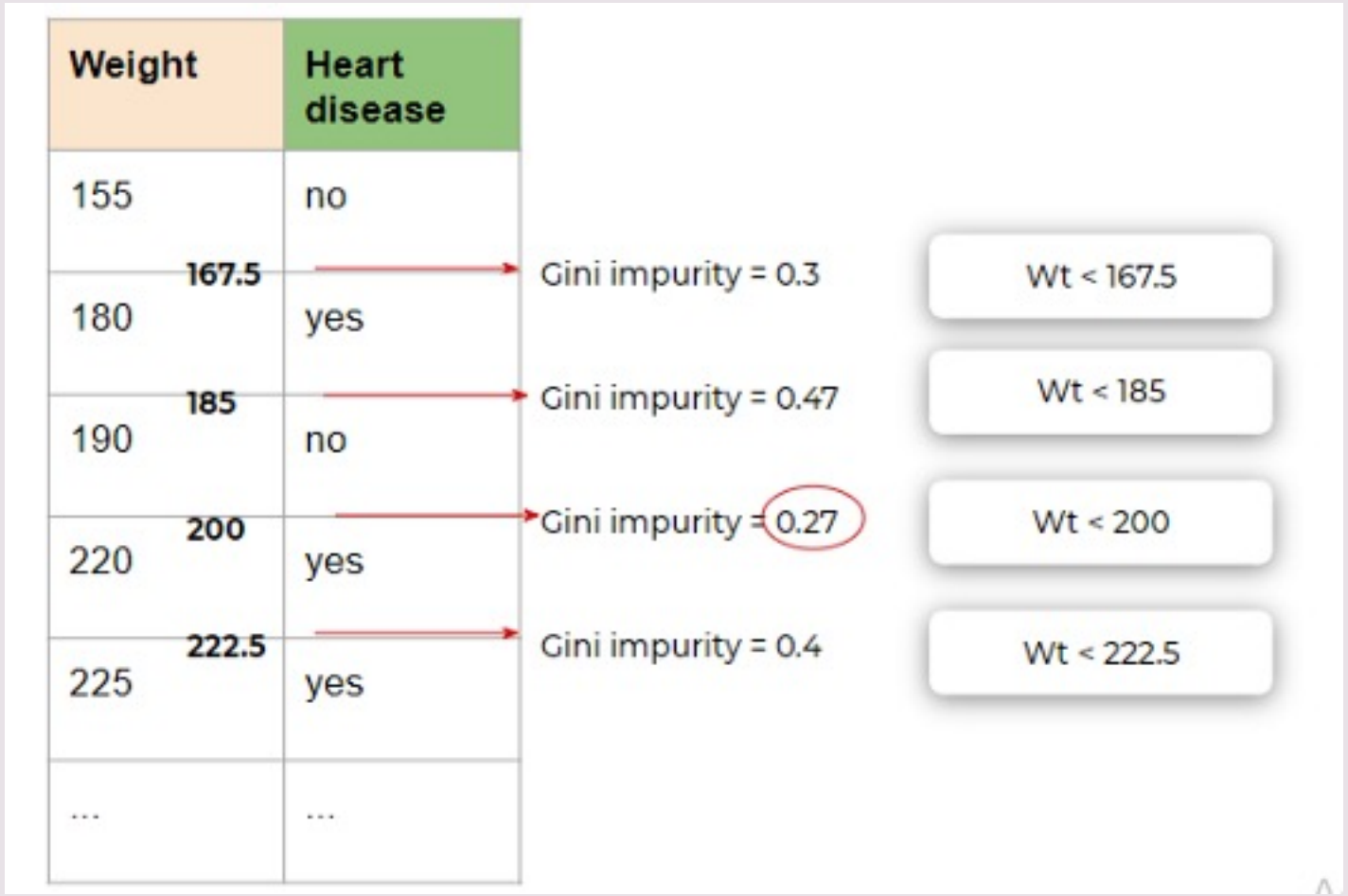
GINI


Chest pain
Yes
No
Heart Disease
Yes 17
No 3
Heart Disease
Yes 7
No 22
Good Blood Circulation
Yes
No
Blocked arteries
133
Yes
No
49
115(Y13 | N102)
Chest pain
Yes
No
Heart Disease
Yes 17
No 3
Heart Disease
Yes 7
No 22
Gini impurity for Chest pain = 0.29
Gini impurity = 1 − [13/(13+102)]² − [102/(13+102)]²
Gini impurity = 0.20

GINI


Good Blood Circulation
Yes
No
Blocked arteries
Chest pain
Yes
No
Yes
No
Chest pain
115(Y13 | N102)
(Y92 | N3)
Blocked arteries
Yes
No
Yes
No
(Y17 | N3)
(Y7 | N22)
(Y8 | N0)
(Y0 | N30)

GINI


Weight
Heart disease
155
no
167.5
180
yes
185
190
no
200
220
yes
222.5
225
yes
...
...
Gini impurity = 0.3
Gini impurity = 0.47
Gini impurity = 0.27
Gini impurity = 0.4
Wt < 167.5
Wt < 185
Wt < 200
Wt < 222.5

GINI

Rating	Heart disease
1	no
1	yes
3	no
2	yes
1	yes
...	...

Rating from 1 to 5

Gini impurity for 2 = 0.57 — Rating < 2

Gini impurity for 3 = 0.17 — Rating < 3

Gini impurity for 4 = 0.3 — Rating < 4

Note: We don't need to calculate Gini impurity for 1 & 5

GINI

Skin Color	Heart disease
White	no
Brown	yes
Brown	no
Black	yes
White	yes
...	...

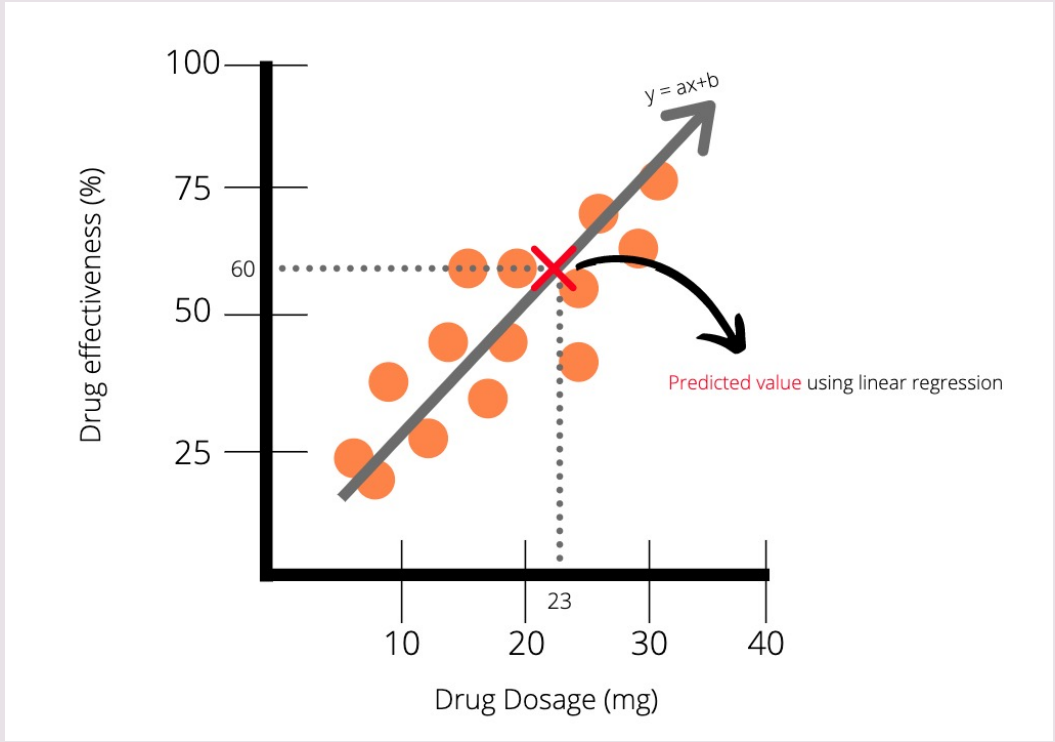
Skin color: White, Brown, Black

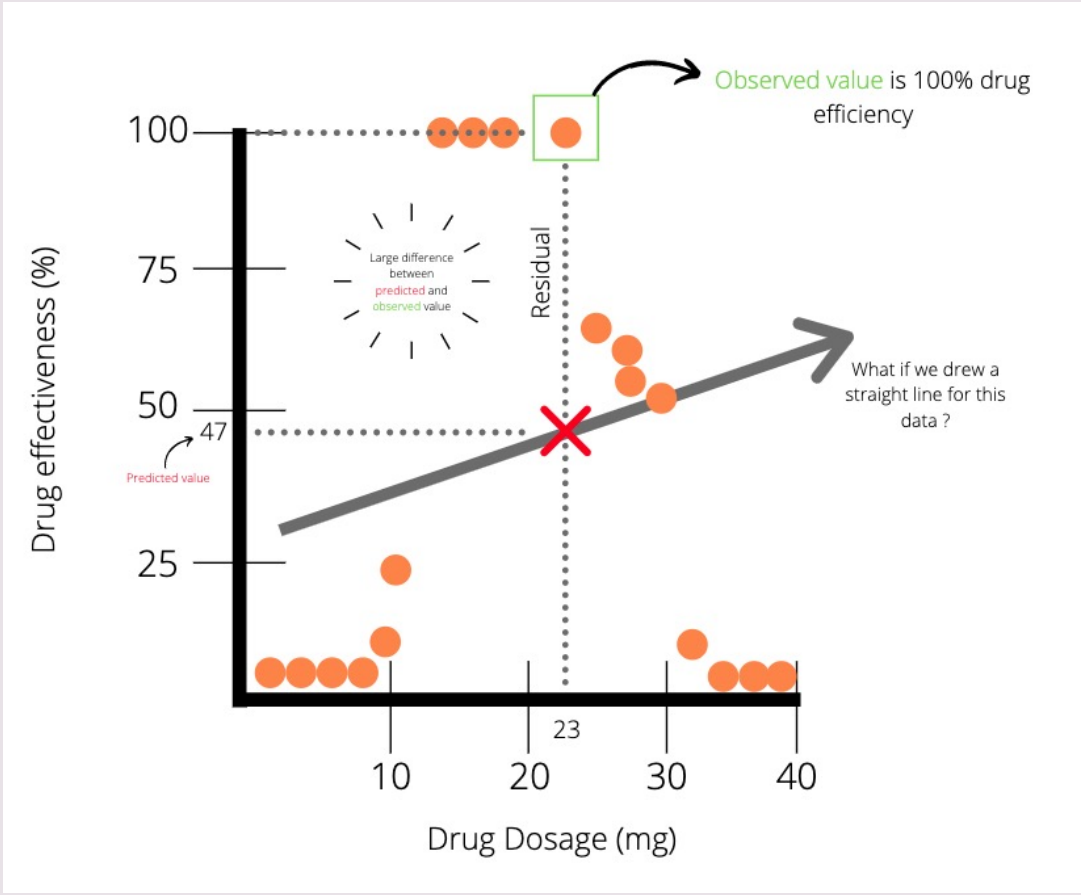
Gini impurity for all following

- White
- White or Brown
- Brown
- White or Black
- Black
- Black or Brown

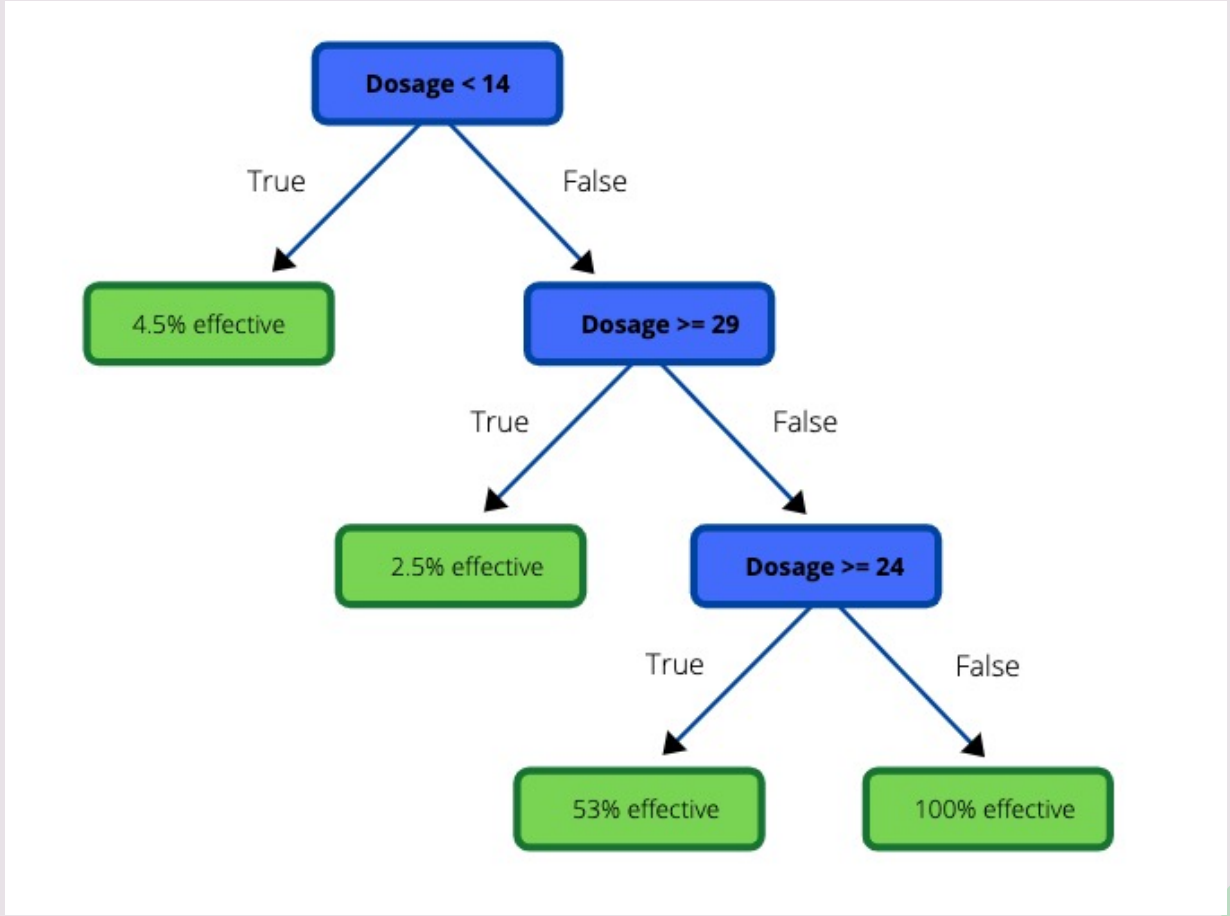
Note: We don't need to calculate Gini impurity for White or Brown or Black

Regresion Tree

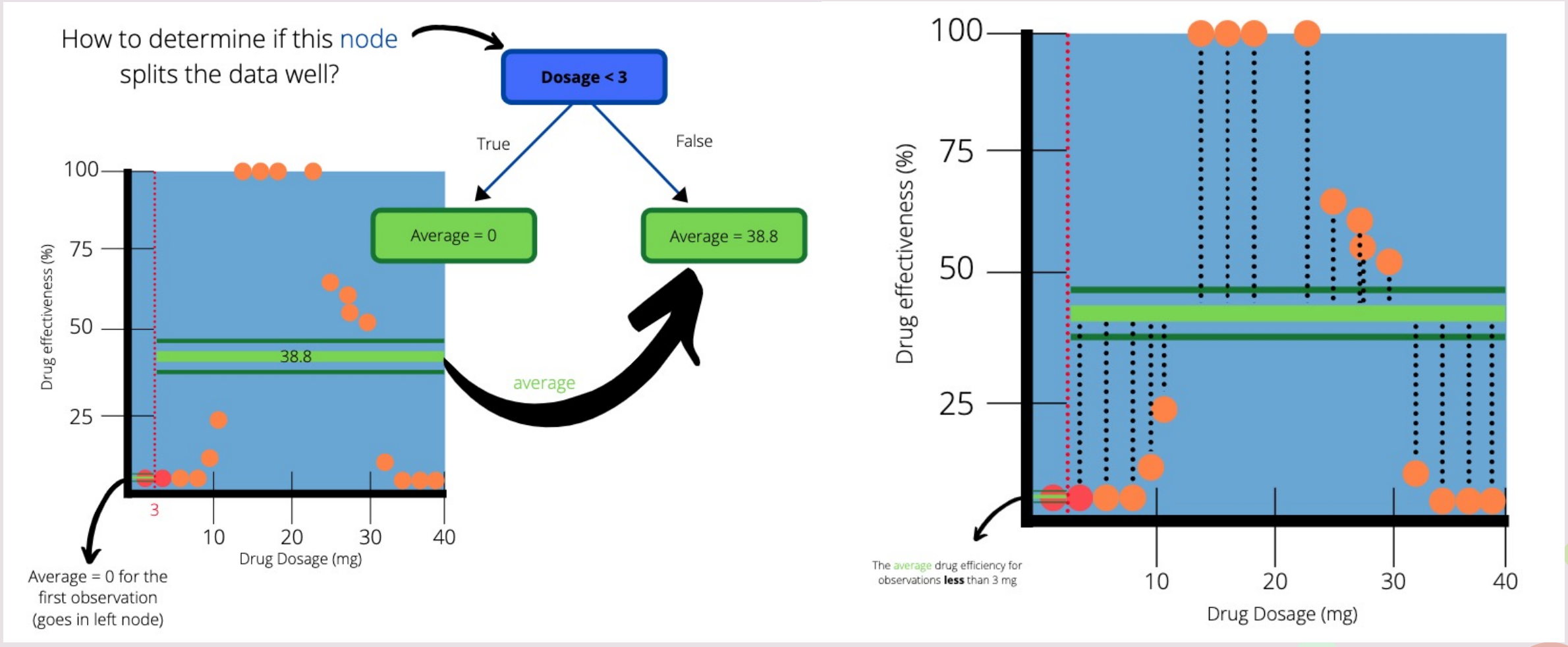




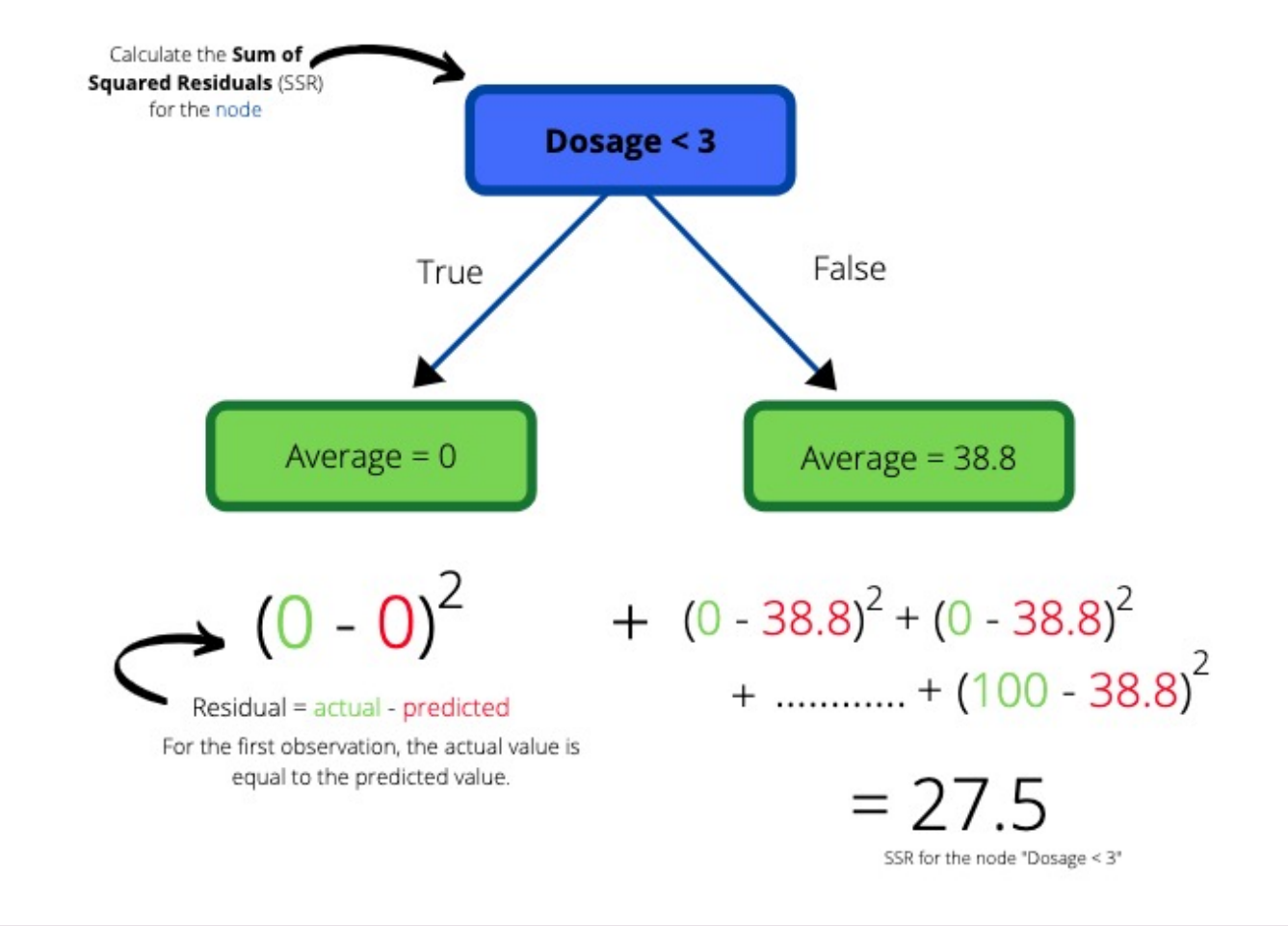
Regresion Tree



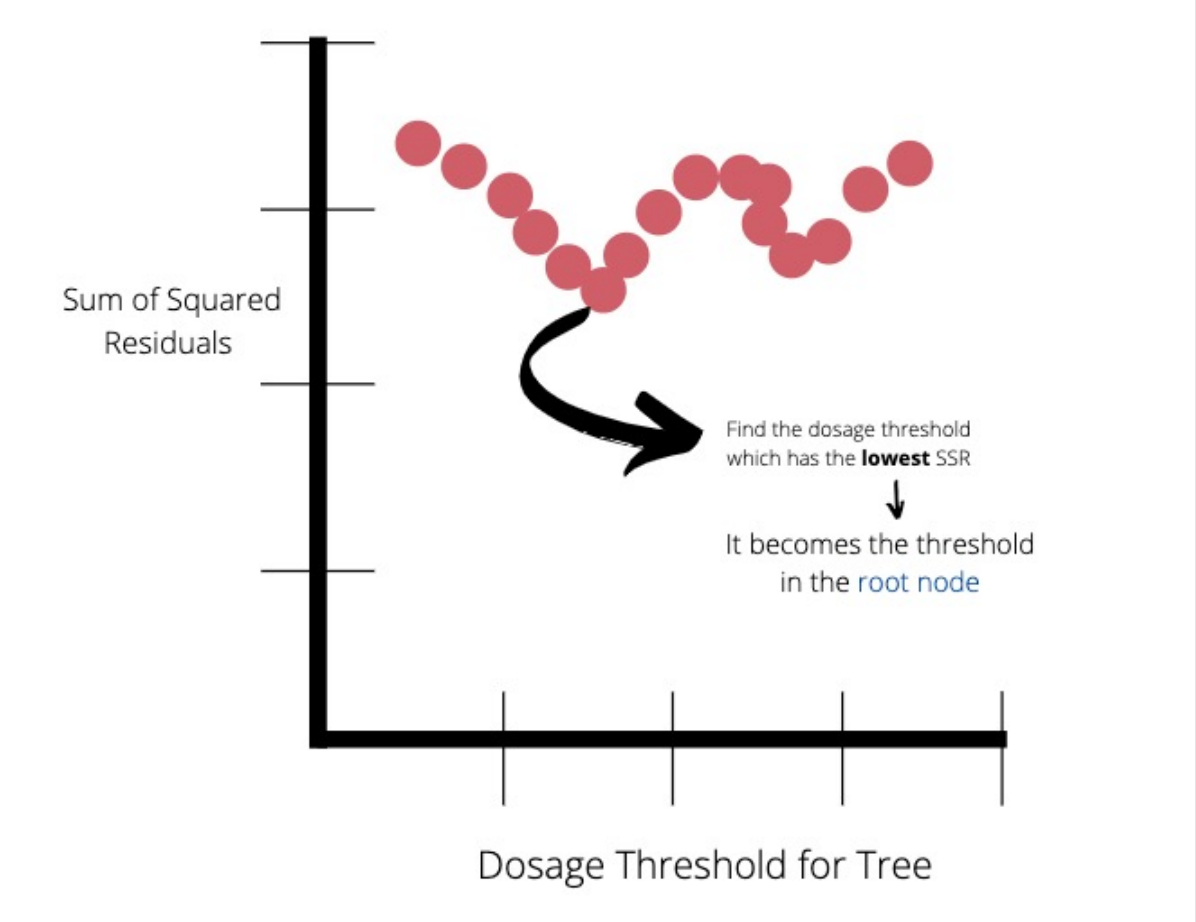
Regresion Tree



Regresion Tree



Regresion Tree



Pro and Cons

Advantages of a decision tree

- **Easy to visualize and interpret**: Its graphical representation is very intuitive to understand and it does not require any knowledge of statistics to interpret it.
- **Useful in data exploration**: We can easily identify the most significant variable and the relation between variables with a decision tree. It can help us create new variables or put some features in one bucket.
- **Less data cleaning required**: It is fairly immune to outliers and missing data, hence less data cleaning is needed.
- **The data type is not a constraint**: It can handle both categorical and numerical data.

Pro and Cons

Disadvantages of decision tree

- **Overfitting**: single decision tree tends to overfit the data which is solved by setting constraints on model parameters i.e. height of the tree and pruning. Sensitive to noisy data. It can overfit noisy data.

- **Not exact fit for continuous data**: It losses some of the information associated with numerical variables when it classifies them into different categories.

- The small variation(or variance) in data can result in the different decision tree. This can be reduced by bagging and boosting algorithms.
- Decision trees are biased with imbalance dataset, so it is recommended that balance out the dataset before creating the decision tree.

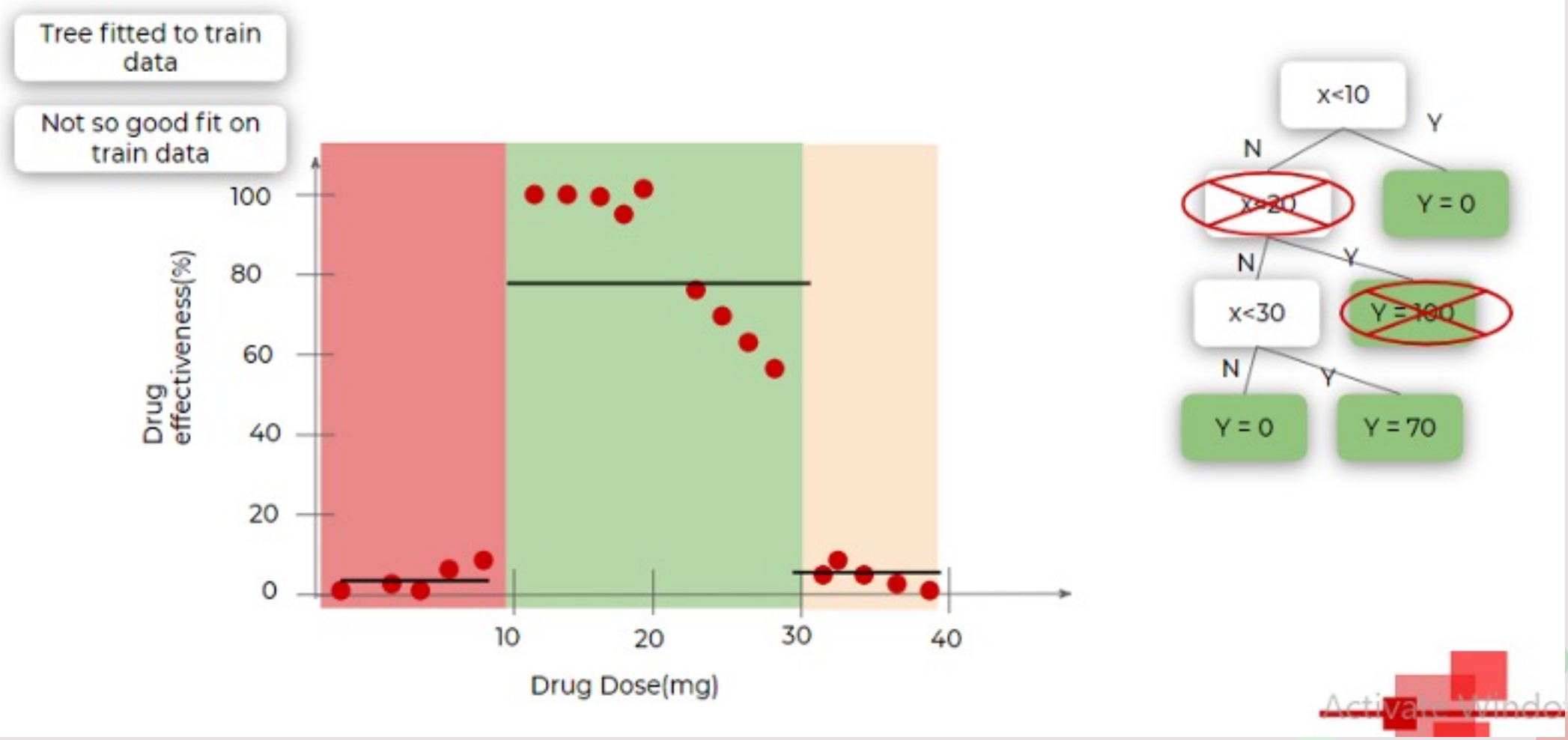
Overfit

Setting Constraints on tree size

- **Minimum samples for a node split**
- **Minimum samples for a leaf node**
- **Maximum depth of the tree (vertical depth)**
- **Maximum number of leaf nodes**
- **Maximum features to consider for a split**
  - The number of features to consider while searching for the best split. These will be randomly selected.
  - As a thumb-rule, the square root of the total number of features works great but we should check up to 30–40% of the total number of features.
  - Higher values can lead to over-fitting but depend on case to case.

Overfit

Pruning



Nấu Mì

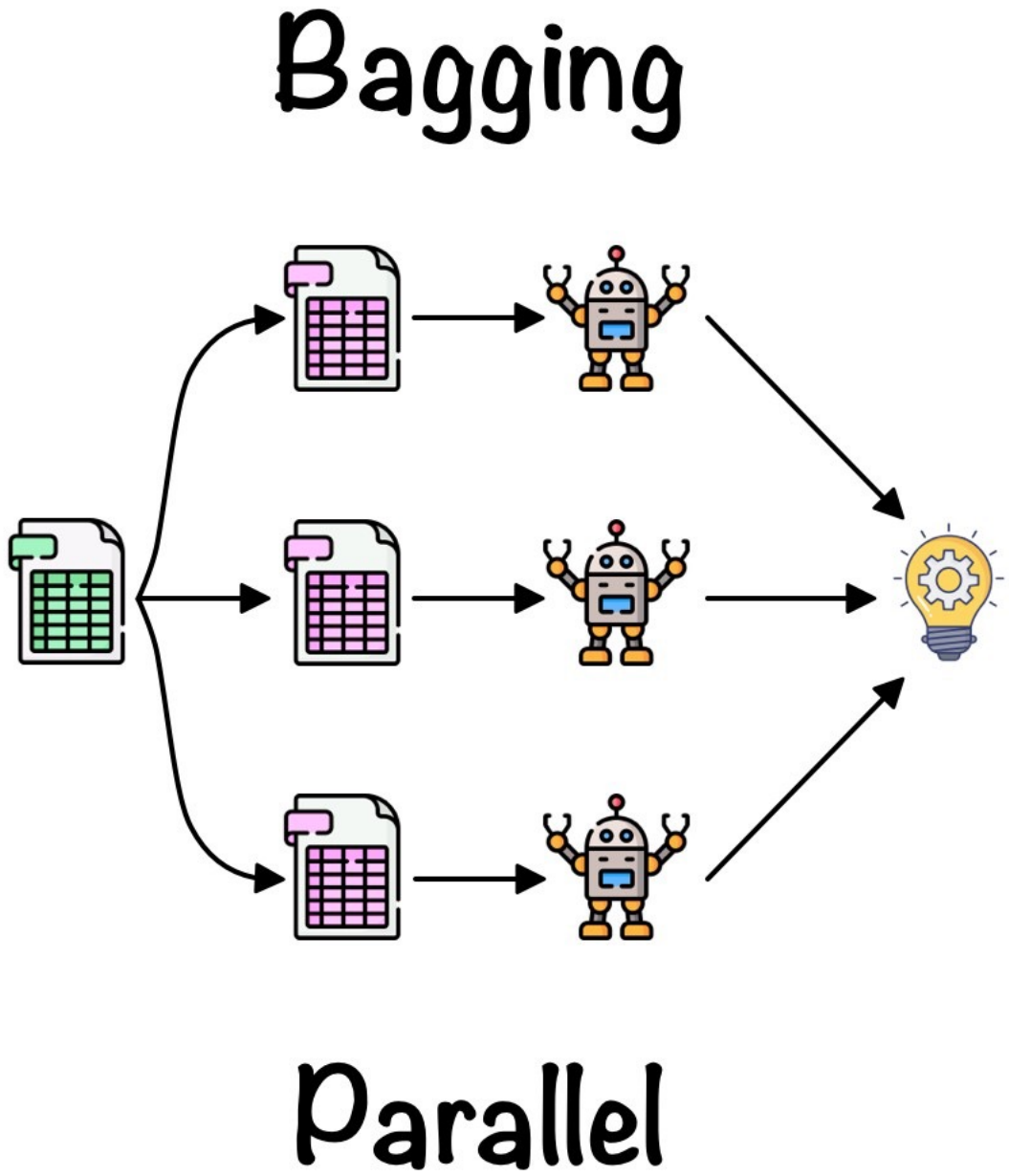
Ensemble Learning

- Ensemble methods is a machine learning technique that combines several base models in order to produce one optimal predictive model.

- EL có thể áp dụng với nhiều loại model, ở đây ta focus vào Decision Tree model (vì đề bài là XGBoost).

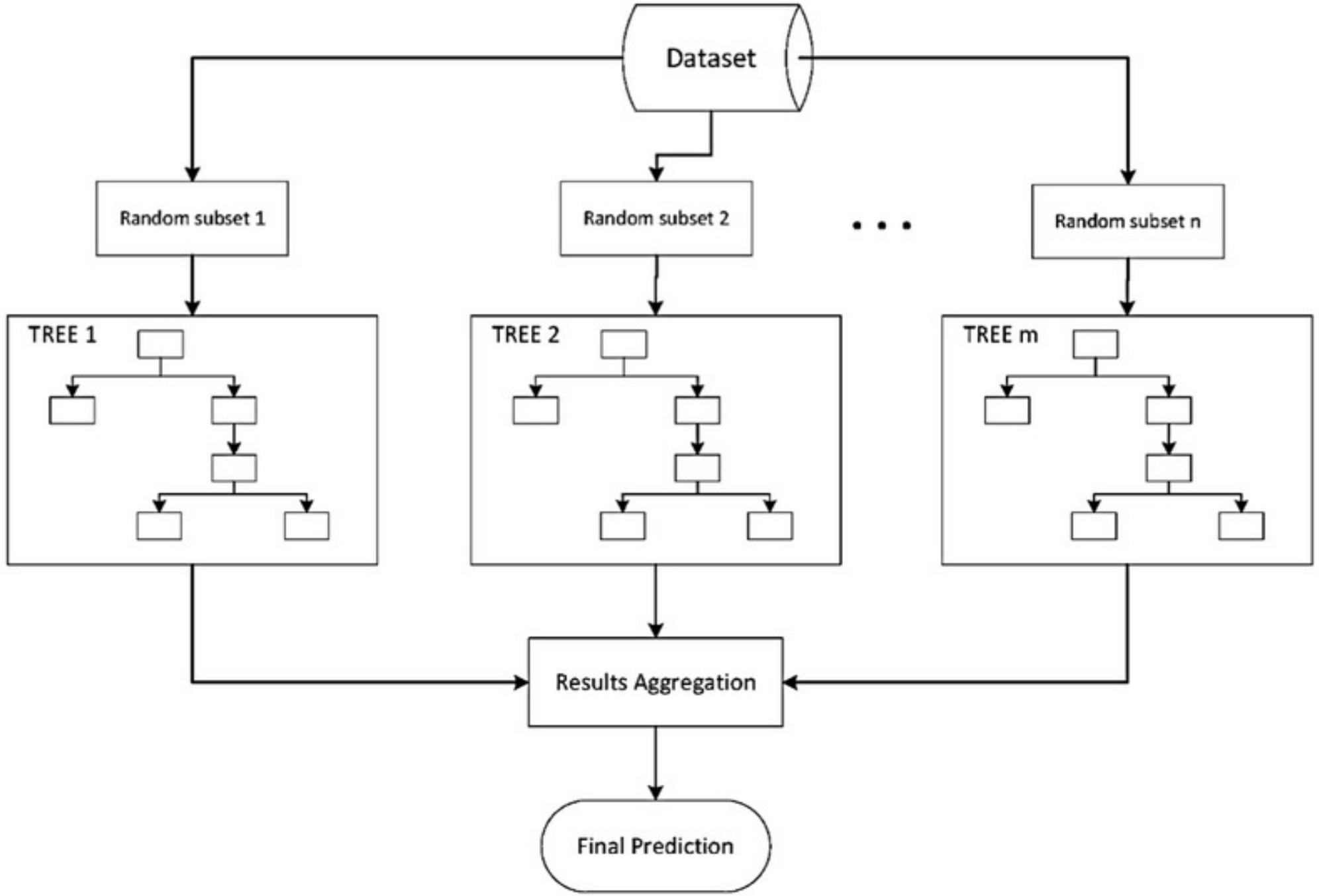
Bagging

- Các bước thực hiện của thuật toán :
  - Bước 1: Tạo ngẫu nhiên các N bags từ tập train set.
  - Bước 2: Tạo N model ”yếu” và train trên mỗi bag, độc lập với nhau.
  - Bước 3: Sử dụng các objects đã trained để dự đoán



Bagging

- Các bag được dựng lên bởi 2 tham số:
  - Max_samples: Số sample mỗi bag
  - Max_features: Số features sử dụng trong các bags.

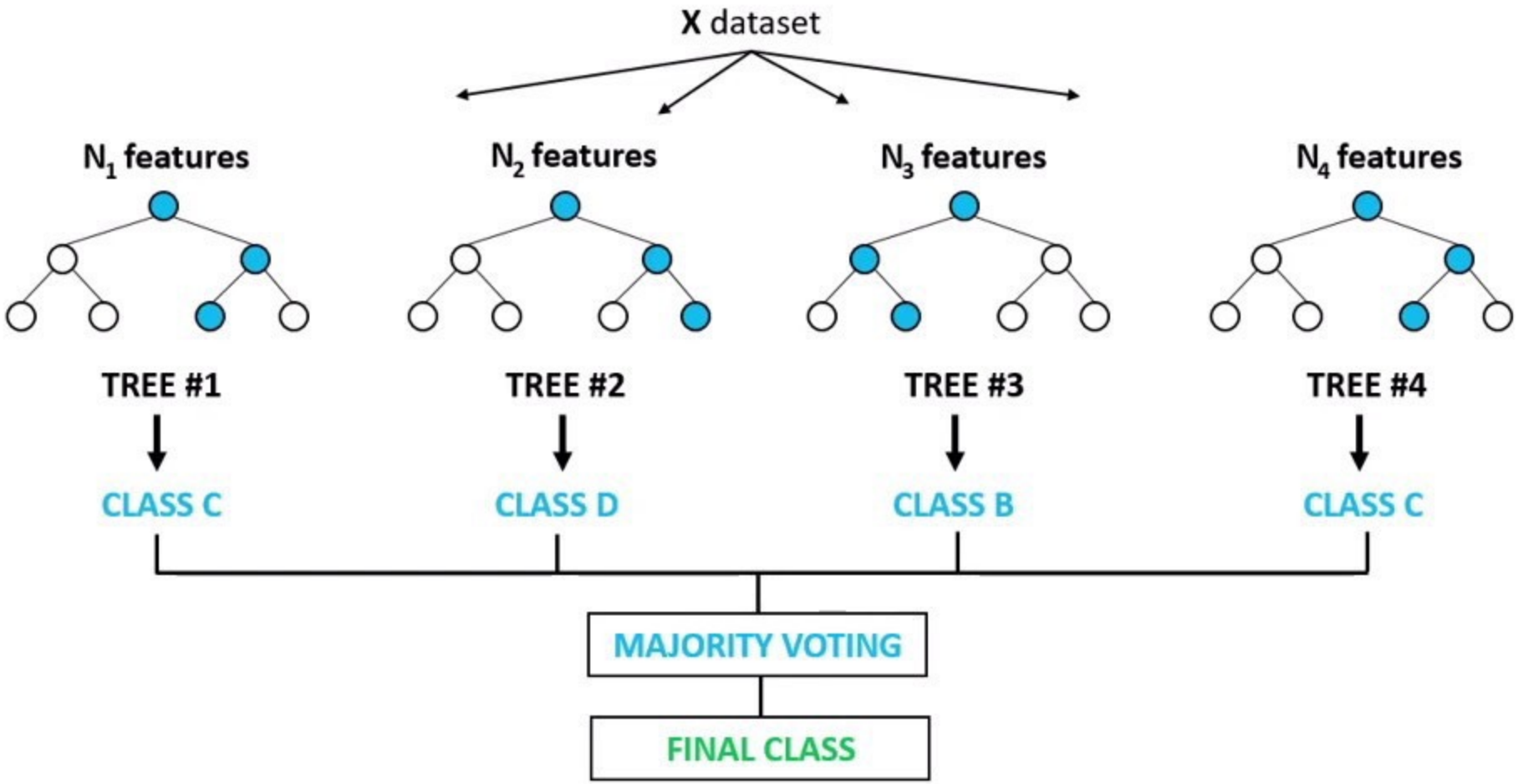

Dataset
Random subset 1
Random subset 2
Random subset n
TREE 1
TREE 2
TREE m
Results Aggregation
Final Prediction

Random Forest

- Random Forest same same Bagging.

- Khác biệt là tại mỗi node của tree trong Decision Tree, nó tạo ra một tập ngẫu nhiên các features và sử dụng tập này đê chọn hướng đi tiếp theo. (Bagging sử dụng tất cả features)

Random Forest Classifier

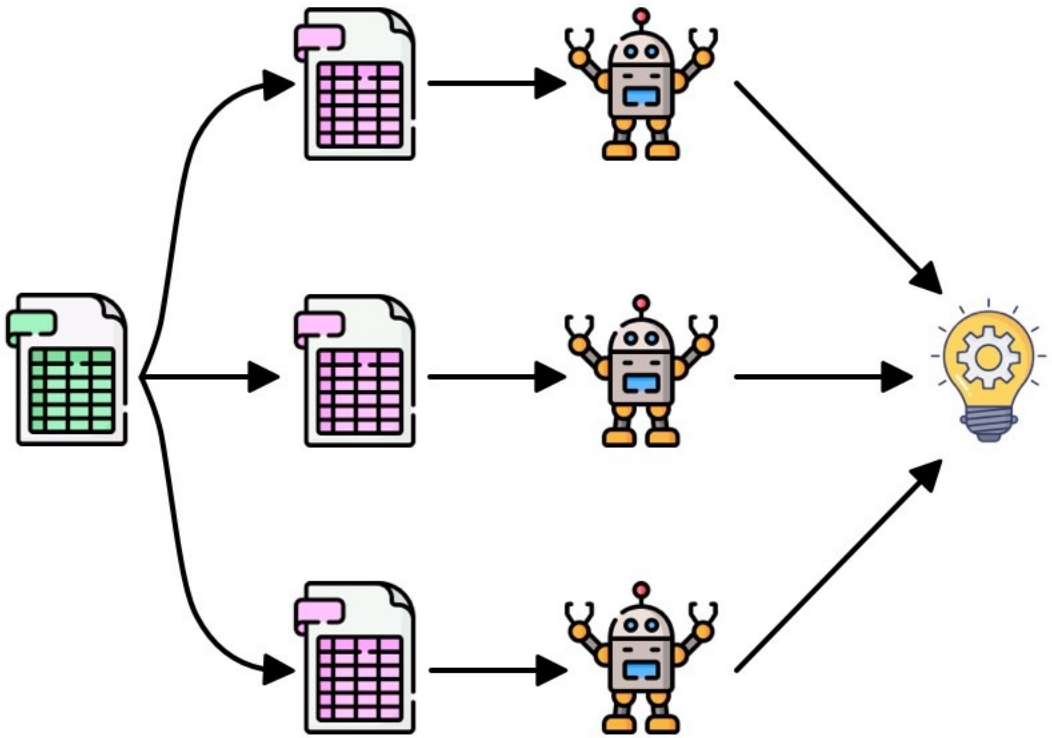


Random Forest

- Một số tham số chính:
  - N_estimators: Số lượng cây
  - Max_features: Số features sử dụng tại mỗi node để xác định phân nhánh

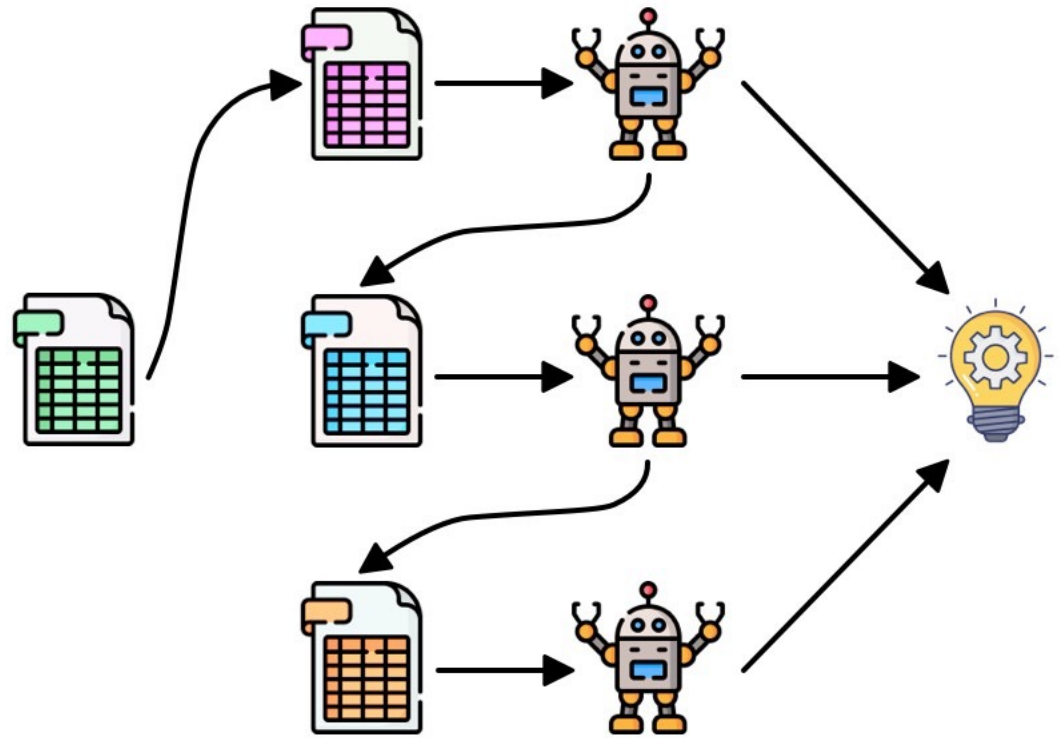
Boosting

Bagging



Parallel

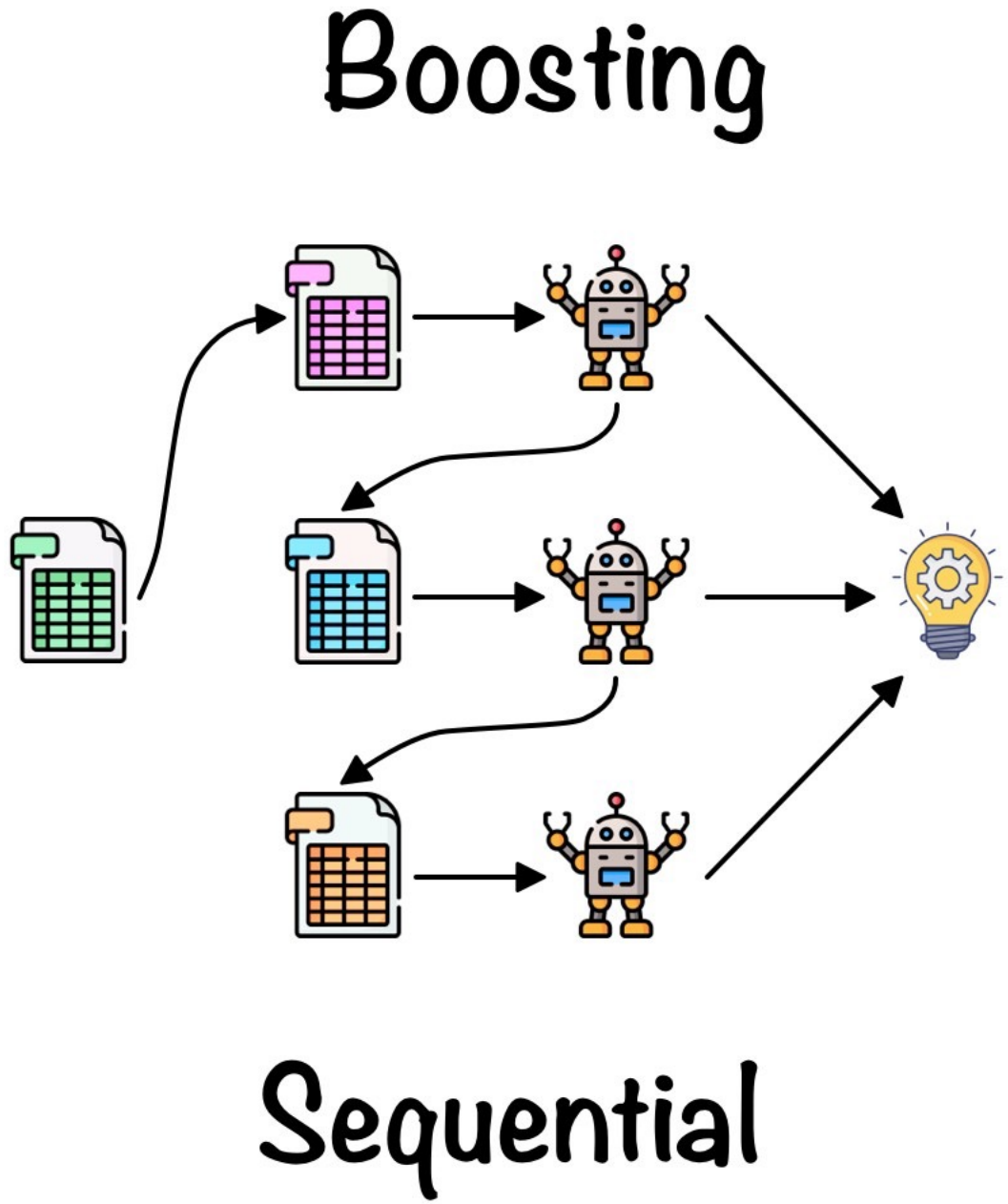
Boosting



Sequential

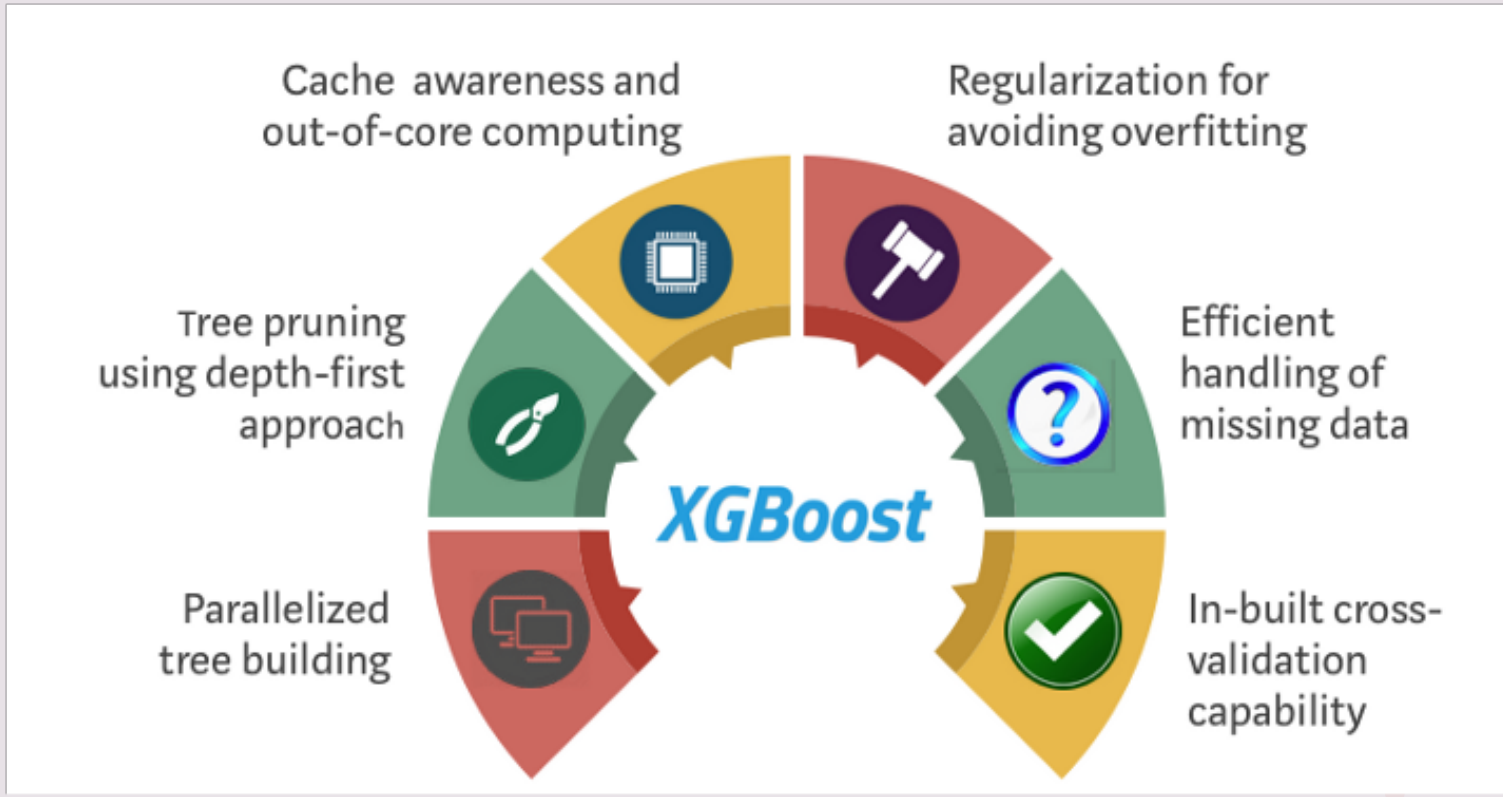
Boosting

- Boosting is an ensemble learning method that combines a set of weak learners into a strong learner to minimize training errors.
- In boosting, a random sample of data is selected, fitted with a model and then trained sequentially—that is, each model tries to compensate for the weaknesses of its predecessor



Boosting

- AdaBoost
- **Gradient Boosting (GBM)**
- **XGBoost**
- LightGBM
- CatBoost



GBM

Customer Age	Customer Salary	Customer Spending (target)
50	1000	100
20	500	50
40	2000	1000
60	100	1

Dự liệu cần dự đoán là Spending

GBM

Customer Age	Customer Salary	Customer Spending (target)	Customer Spending (Predict)
50	1000	100	95
20	500	50	40
40	2000	1000	920
60	100	1	5

B1. Train Decision Tree dự đoán Spending

GBM

Customer Age	Customer Salary	Customer Spending (target)	Customer Spending (Predict)	Error
50	1000	100	95	5
20	500	50	40	10
40	2000	1000	920	80
60	100	1	5	-4

B2. Tính Error

GBM

Customer Age	Customer Salary	Customer Spending (target)	Customer Spending (Predict)	Error	Error (Predict)
50	1000	100	95	5	2
20	500	50	40	10	12
40	2000	1000	920	80	76
60	100	1	5	-4	-2

B3. Train Decision Tree mới để dự doán Error
(same input, khác output)

GBM

Customer Age	Customer Salary	Customer Spending (target)	Customer Spending (Predict)	Error	Error (Predict)	Error of Error
50	1000	100	95	5	2	3
20	500	50	40	10	12	-2
40	2000	1000	920	80	76	4
60	100	1	5	-4	-2	-2

B4. Tính Error trên việc “Dự đoán Error”
Quay lại B2, coi Error of Error là Error và lặp lại quá trình

GBM

Customer Age	Customer Salary	Customer Spending (target)	Customer Spending (Predict)	Error	Error (Predict)	Error of Error
50	1000	100	95	5	2	3
20	500	50	40	10	12	-2
40	2000	1000	920	80	76	4
60	100	1	5	-4	-2	-2

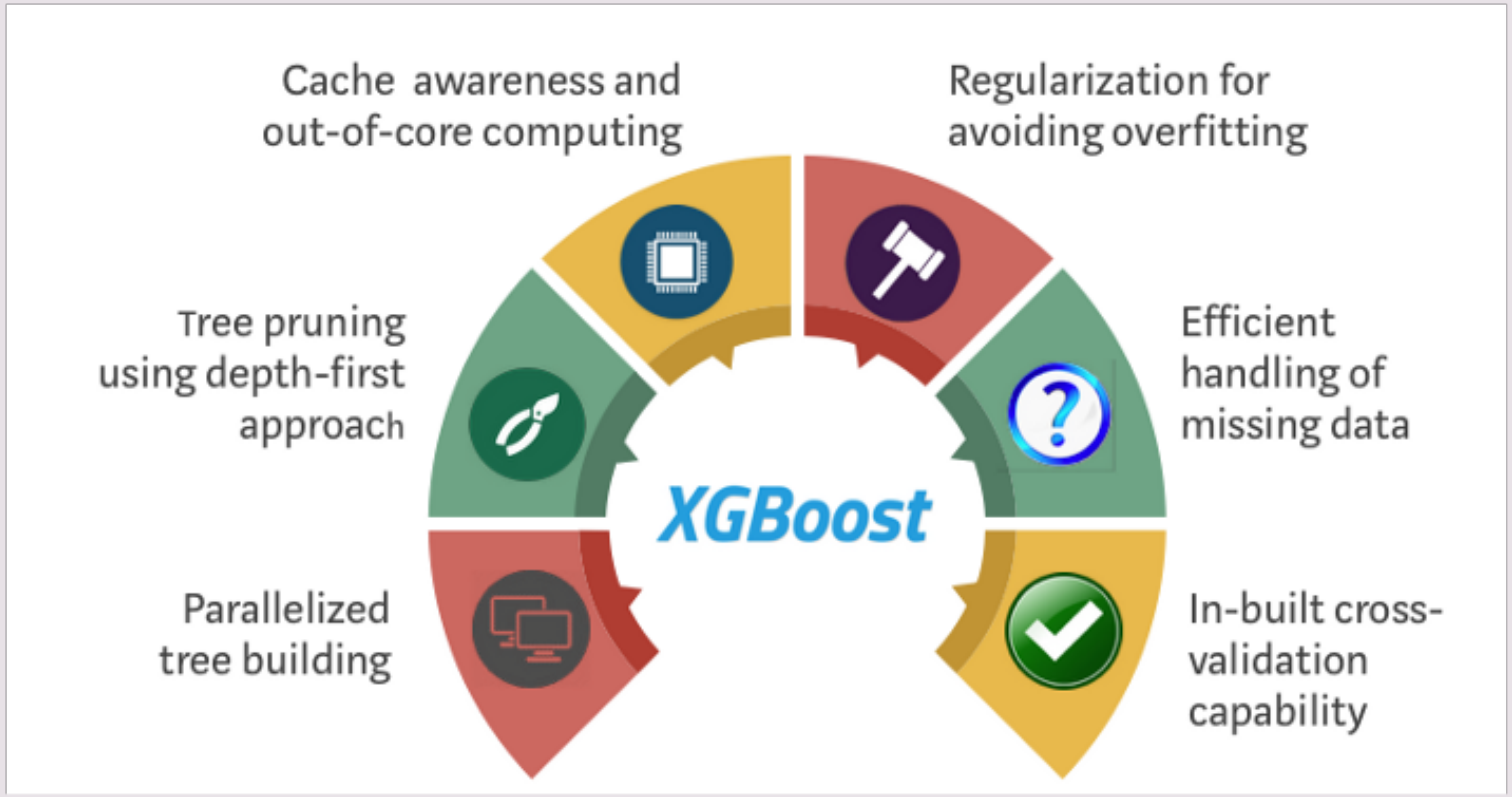
Quá trình lặp sẽ dừng lại khi Error không đổi hoặc sau khi số lượng cây bằng 1 giá trị đặt trước

XGBoost

- XGBoost (*extreme Gradient Boosting*) là phiên bản cải tiến của Gradient Boosting.

- Ưu điểm:
  - Tốc độ nhanh do tính toán song song
  - Tránh được Overfit bằng Regularization
  - Linh hoạt trong sử dụng hàm tối ưu
  - Tự động xử lý missing value
  - Tự động cắt tỉa cây (auto pruning). Tự động bỏ qua những leaves, nodes không mang giá trị tích cực trong quá trình mở rộng tree.

XGBoost


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Hands-on