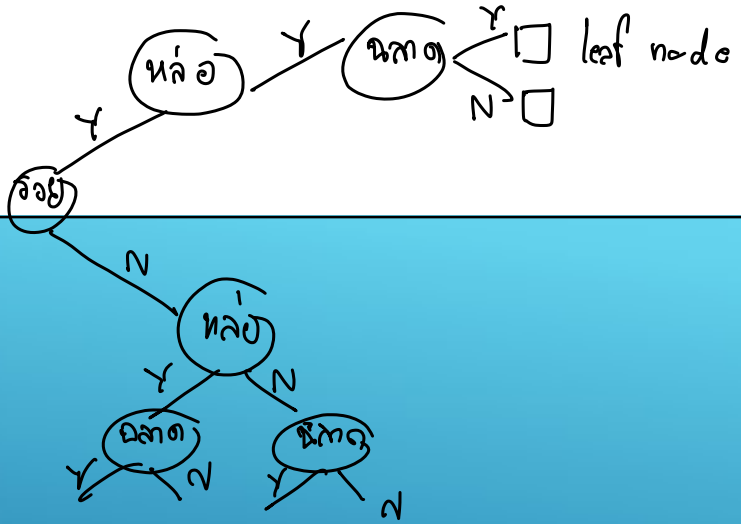


CHAPTER 6
DECISION TREE

PART I: DECISION TREE CLASSIFIERS
PART II: DECISION TREE ENSEMBLES

Associate Professor Yachai Limpiyakorn, Ph.D

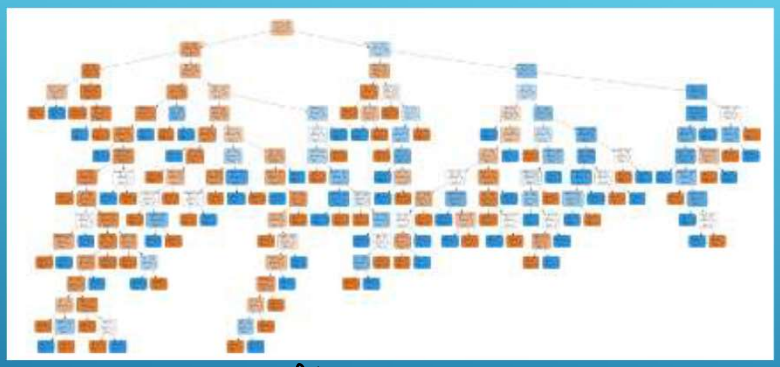
Diagram ≠ แผนผัง Chart

PART 1: DECISION TREE CLASSIFIERS

ตัวจำแนกประเภท

- A tree-like diagram illustrates all possible decision alternatives and the corresponding outcomes.
- Starting from the root of a tree,
  - **internal** node represents the basis on which a decision is made;
  - each **branch** of a node represents how a choice may lead to the next nodes;
  - terminal node, **leaf**, represents the outcome produced.
  - Paths from root to leaves represent **classification rules**

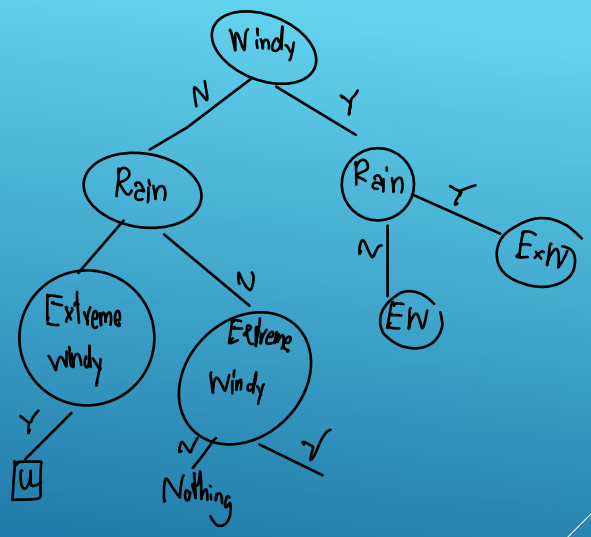
1. Tree Construction



2. Tree Pruning DT ตัดกิ่ง overfitting

DECISION TREE LEARNING

Rain	Windy	Xtreme windy	Decision
0	0	0	Nothing
0	0	1	Nothing
0	1	0	Nothing
0	1	1	Nothing
1	0	0	Umbrella
1	0	1	Umbrella
1	1	0	Rain jacket
1	1	1	Stay home



TREE CONSTRUCTION

Occam's Razor

สมมติฐานที่สั้นกว่าที่สามารถอธิบายข้อมูลได้เหมือนกัน จะเป็นสมมติฐานที่ดีกว่า

the simplest explanation is usually the best one

ก่อนเป็น classification rule ตอนเป็น hypothesis มาก่อน

~raining → don't bring anything

raining and ~windy → use an umbrella

raining and ~extremely windy → wear a rain jacket

raining and extremely windy → stay home

overfit กับ training data — model quality — generalization

มีความเป็น generalization

https://medium.com/@ml.at.berkeley/machine-learning-crash-course-part-5-decision-trees-and-ensemble-models-dcc5a36af8cd

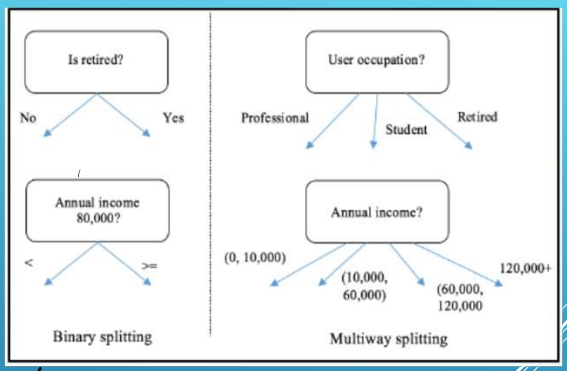
เลือก attribute ที่ให้ค่า ASM

Select the best attribute using Attribute Selection Measures(ASM) to split the records.

Make that attribute a decision node and breaks the dataset into smaller subsets.

Starts tree building by repeating this process recursively for each child until one of the condition will match: หยุดสร้างเมื่อ

- All the tuples belong to the same class. class เดียวกัน
- There are no more remaining attributes. ไม่มี attribute
- There are no more instances. ไม่มีตัวอย่าง



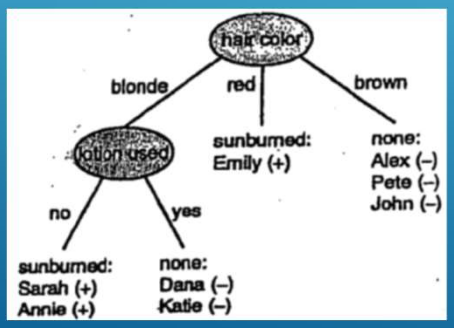
BASIC IDEA OF DECISION TREE ALGORITHM (TOP-DOWN CONSTRUCTION)

การสร้างจาก root ลงไป leaf

ASM ตัวอย่าง attribute ที่ดีที่สุด

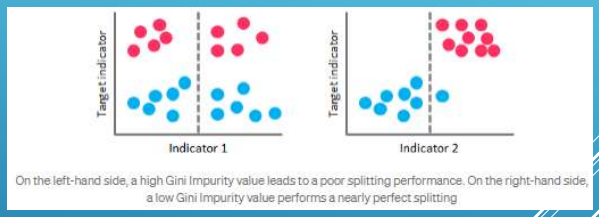
Multi-way Split

- Information Gain – ID3 [Ross Quinlan]
- Gain ratio – C4.5 [Ross Quinlan]



Binary Split

- GINI – CART (Classification and Regression Tree)



On the left-hand side, a high Gini Impurity value leads to a poor splitting performance. On the right-hand side, a low Gini Impurity value performs a nearly perfect splitting

SPLIT MEASURE/ ASM

M = {A, B, C, D}

กำหนด message M ประกอบด้วยค่าที่เป็นไปได้ $\{m_1, m_2, \ldots, m_n\}$ และ
ความน่าจะเป็นที่จะเกิดค่า $m_i = P(m_i)$ จะได้ว่า
จำนวนบิตน้อยที่สุดที่ใช้ encode m_i แต่ละตัว ที่ให้ค่าเฉลี่ยจำนวนบิตที่น้อยที่สุด คือ

เพราะมีค่าติดลบ

$$\text{Optimal code length } (m_i) = -\log_2 P(m_i)$$

$-\log_2 2^{-1} = 1$

ค่าสารสนเทศของ M หรือค่าเอนโทรปีของ M เขียนแทนด้วย $I(M)$ คำนวณโดย

สัมประสิทธิ์ถ่วง

$$I(M) = \sum_{i}^{n} -P(m_i)\log_2 P(m_i)$$

กรณีเทคสัญลักษณ์ 4 ตัว

Message	Probability	Standard Code	Optimal Code
A	$0.5\ 2^{-1}$	00	0
B	$0.25\ 2^{-2}$	01	10
C	$0.125\ 2^{-3}$	10	110
D	$0.125\ 2^{-3}$	11	111
Average Encoding Length		2 bits	1.75 bits

Average Encoding Length of Optimal Code is calculated by
=(-0.5*log0.5) +(-0.25*log0.25)+(-0.125*log0.125)+(-0.125*log0.125)
=(0.5*1)+(0.25*2)+(0.125*3)+(0.125*3)=1.75 bits

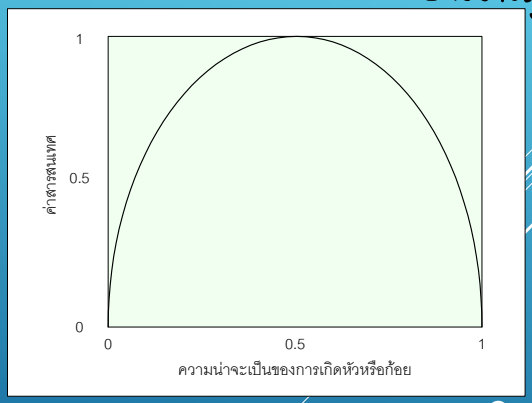
ENTROPY/ INFORMATION

INFORMATION/ ENTROPY OF COIN FLIP

$\log_2 1 = 0$
$2^0 = 1$

$$I(\text{การโยนหัวโ ยนก้อย}) = -P(\text{หัว})\log_2(P(\text{หัว})) - P(\text{ก้อย})\log_2(P(\text{ก้อย}))$$ บอกถึงความไร้ระเบียบของข้อมูล

- M1=HHHHHHHH
  I(M1) = (-1log$_2$1)+ (-0log$_2$ 0) = 0
- M2=TTTTTTTT
  I(M2) = (-0log$_2$ 0)+ (-1log$_2$ 1)= 0
- M3=HHHHTTTT
  I(M3) = (-0.5log$_2$ 0.5)+ (-0.5log$_2$ 0.5)= 1



Lower entropy implies a purer dataset. In a perfect case where the dataset contains only one class, the entropy is –(1*log$_2$1+0)=0

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

ID3 [J.R. QUINLAN] แต่เดิม ID3 ใช้ได้กับ categorical เท่านั้น

Id3Estimator

ID3 CONSTRUCTION (1)

① หา Entropy ของ class
② หา information ของแต่ละ attribute
③ เอา Entropy class - ②

S=[9+, 5-]

$I(S)= \frac{-9}{14} log_2 \frac{9}{14} + \left(-\frac{5}{14} log_2 \frac{5}{14}\right)=0.940$

Age

< 30 : LE 30 — S1=[2+, 3-]

[31..40] — S2=[4+, 0-]

> 40 : GT 40 — S3=[3+, 2-]

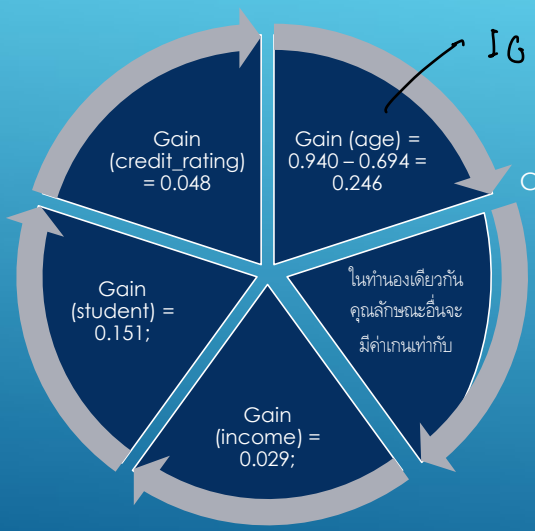
$I(S1)= \frac{-2}{5} log_2 \frac{2}{5} + \left(-\frac{3}{5} log_2 \frac{3}{5}\right)$

$I(S3)= -\frac{3}{5} log_2 \frac{3}{5} + \left(-\frac{2}{5} log_2 \frac{2}{5}\right)$

$I(S2)= \frac{-4}{4} log_2 \frac{4}{4} + (-0\ log_2\ 0)$

$I_{age}(S) = \frac{5}{14}\left(-\frac{2}{5}\log_2\frac{2}{5}-\frac{3}{5}\log_2\frac{3}{5}\right)+\frac{4}{14}(0)+\frac{5}{14}\left(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}\right)=0.694$

ID3 CONSTRUCTION (2)



$IG = I(S) - I_A(S)$

Choose attribute providing max GAIN → Age

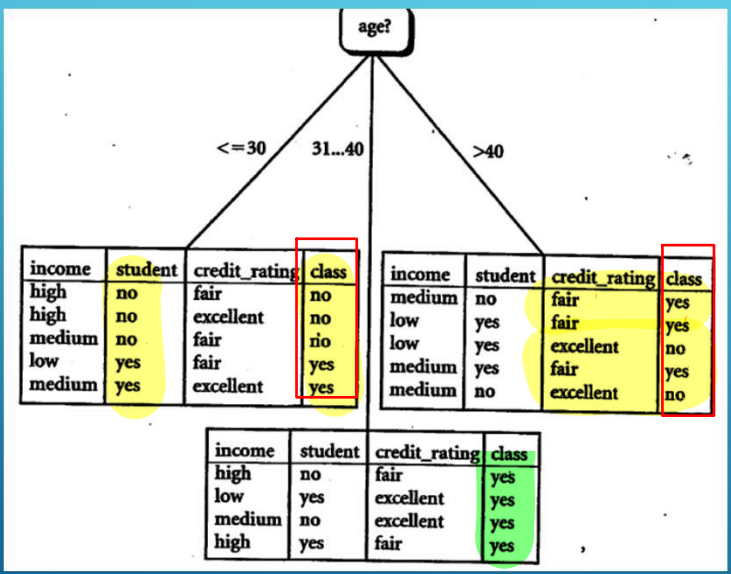
↓ profit

ID3 CONSTRUCTION (3)

$f(a_1, a_2, .., a_m) : c \in \{c_1, c_2\}$

$m - 1$

$1 - 1$

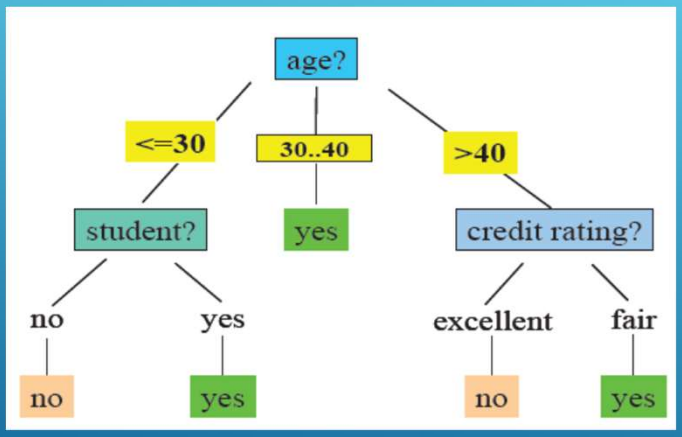


income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	rio
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

OUTPUT OF ID3 LEARNING (MULTI-WAY SPLIT)



ออกมา 5 กฎ

Gini Impurity ความไม่บริสุทธิ์ (เหลวแหม่)

เลือก attribute ที่ให้ค่า Gini ต่ำก่อน

- Lower Gini indicates a purer dataset
- For a dataset with **K** classes, suppose data from class $k(1 \le k \le K)$, take up a fraction $f_k (0 \le f_k \le 1)$ of the entire dataset:

$$GiniImpurity = 1 - \sum_{k=1}^{K} f_k^2$$

- To evaluate quality of a split, add up the Gini of all resulting subgroups, combining the proportions of each subgroup as corresponding weight factors.
- The smaller the weighted sum of Gini Impurity, the better the split.

User gender	Interested in tech	Click	Group by gender
M	True	1	Group 1
F	False	0	Group 2
F	True	1	Group 2
M	False	0	Group 1
M	False	1	Group 1

#1 split based on gender

User gender	Interested in tech	Click	Group by interest
M	True	1	Group 1
F	False	0	Group 2
F	True	1	Group 1
M	False	0	Group 2
M	False	1	Group 2

#2 split based on interest in tech

Weighted Gini Impurity of #1split based on gender

$$\#1\ Gini\ Impurity = \frac{3}{5}\left[1 - \left(\frac{2^2}{3} + \frac{1^2}{3}\right)\right] + \frac{2}{5}\left[1 - \left(\frac{1^2}{2} + \frac{1^2}{2}\right)\right] = 0.467$$

click 1st click

Weighted Gini Impurity of #2 split based on tech interest

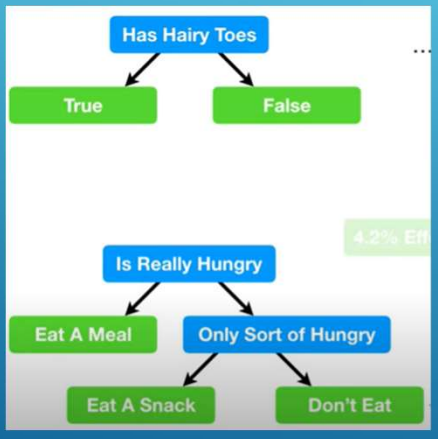
$$\#2\ Gini\ Impurity = \frac{2}{5}[1 - (1^2 + 0^2)] + \frac{3}{5}\left[1 - \left(\frac{1^2}{3} + \frac{2^2}{3}\right)\right] = 0.267$$

click 1st click

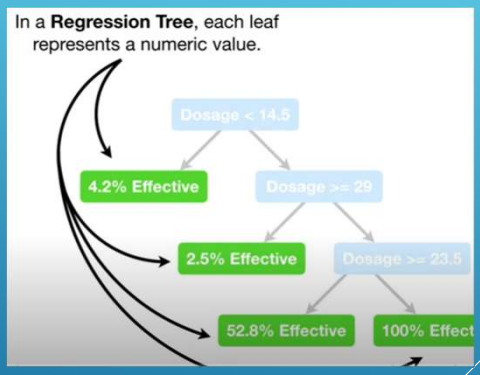
interest not interest

SPLIT METRICS: GINI INDEX

CART (Classification and Regression Tree)



Classification Tree



Regression Tree

- Most ML learning models in Python work with numerical data
- Three approaches to manage categorical data:
  - Drop categorical variables if NOT relevant
  - Label encoding or ranking in case of ordinal variables
  - One-Hot encoding

Label	Encoded Label
Africa	1
Asia	2
Europe	3
South America	4
North America	5
Other	6

Label encoding

	is_africa	is_asia	is_europe	is_sam	is_nam
Africa	1	0	0	0	0
Asia	0	1	0	0	0
Europe	0	0	1	0	0
South America	0	0	0	1	0
North America	0	0	0	0	1
Other	0	0	0	0	0

One-hot encoding

DATA PREPROCESSING

ข้อดี

- ช่วยให้การเรียนรู้ง่ายขึ้น แทนที่โมเดลจะเรียนรู้ Pattern เปลี่ยนมาเรียนรู้จากสวิตช์
- ความหมายของข้อมูลแบบ Nominal จะตรงขึ้น Europe 3 ไม่ได้ใกล้เคียง S.America 4 มากกว่า N. America 5
- สามารถ Dot Product กับ Matrix ที่ต้องการ

ข้อเสีย

- ความหมายของลำดับข้อมูลแบบ Ordinal จะหายไป เนื่องจากทุก Category แตกต่างกันเท่ากันหมด
- ถ้าข้อมูลมี Value หลากหลายมาก เช่น มีสีเสื้อ 10,000 สี จะทำให้มีปัญหาเปลืองเนื้อที่/Memory ที่เก็บค่า 0 เป็นส่วนใหญ่ เรียกว่า Sparse Matrix ต้อง column ใหม่เยอะมาก
- การเพิ่ม Categoryใหม่ ยิบย่อยอยู่ตลอด จะทำให้มีปัญหา เช่น เพิ่มสีเสื้อใหม่

ONE-HOT ENCODING

CART ALGORITHM

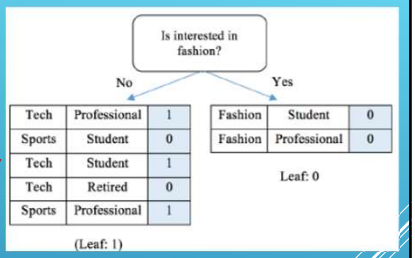
- Always produces **binary** splits
- **Gini index**. A Gini score of 0 indicates perfect purity and a score of 1 indicates maximum impurity.
- CART should be allowed to go till 7–8 tree depth in accordance with the nature of producing tall and skinny trees.
- Splitting stops when CART detects no further gain can be made, or some pre-set stopping rules are met. (Alternatively, the data are split as much as possible and then the tree is later pruned).
- The optimal Tree is identified by evaluating the performance of every Tree through test set; or performing k-fold cross-validation.

IMPLEMENTING A CART TREE (1)

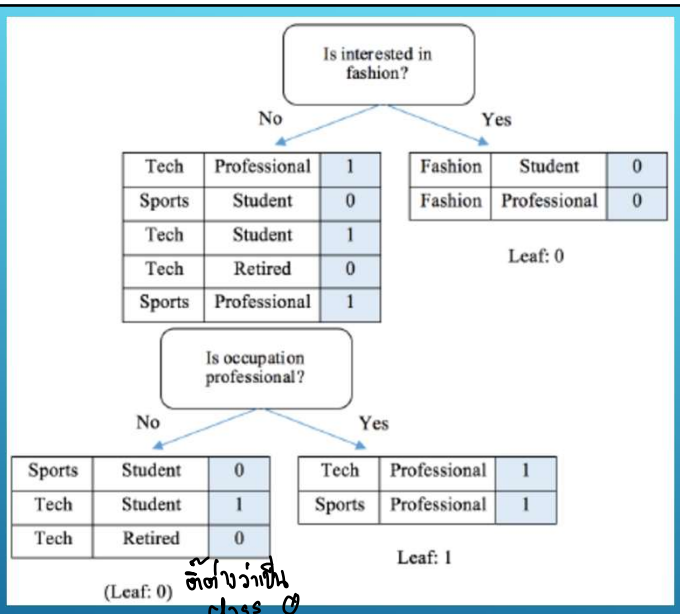
- Gini(interest, tech) = weighted_impurity([[1, 1, 0], [0, 0, 0, 1]]) = 0.405
- Gini(interest, Fashion) = weighted_impurity([[0, 0], [1, 0, 1, 0, 1]]) = 0.343
- Gini(interest, Sports) = weighted_impurity([[0, 1], [1, 0, 0, 1, 0]]) = 0.486
- Gini(occupation, professional) = weighted_impurity([[0, 0, 1, 0], [1, 0, 1]]) = 0.405
- Gini(occupation, student) = weighted_impurity([[1, 0, 0, 1], [0, 0, 1]]) = 0.476
- Gini(occupation, retired) = weighted_impurity([[1, 0, 0, 0, 1, 1], [0]]) = 0.429

User interest	User occupation	Click
Tech	Professional	1
Fashion	Student	0
Fashion	Professional	0
Sports	Student	0
Tech	Student	1
Tech	Retired	0
Sports	Professional	1

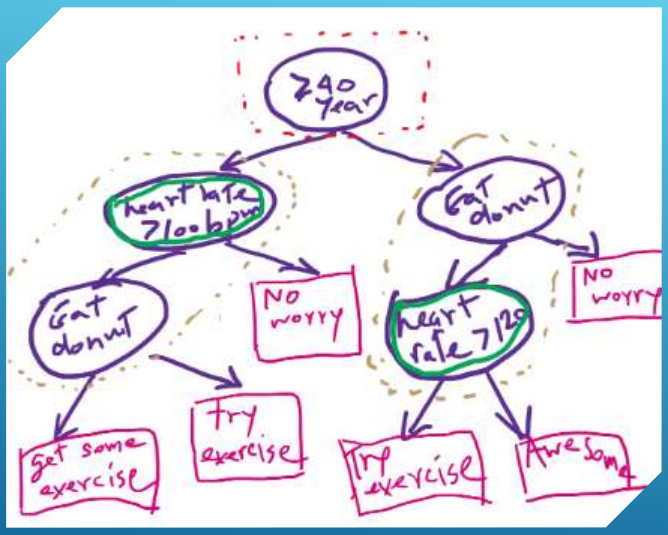
- Gini(interest, tech) = weighted_impurity([[0, 1], [1, 1, 0]]) = 0.467
- Gini(interest, Sports) = weighted_impurity([[1, 1, 0], [0, 1]]) = 0.467
- Gini(occupation, professional) = weighted_impurity([[0, 1, 0], [1, 1]]) = 0.267
- Gini(occupation, student) = weighted_impurity([[1, 0, 1], [0, 1]]) = 0.467
- Gini(occupation, retired) = weighted_impurity([[1, 0, 1, 1], [0]]) = 0.300



IMPLEMENTING A CART TREE (2)



IMPLEMENTING A CART TREE (3)



EXPLORATION

ต้องการแยกกลุ่ม numeric

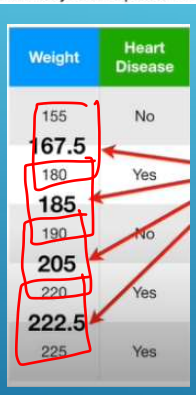
Weight	Heart Disease
220	Yes
180	Yes
225	Yes
190	No
155	No

	Weight	Heart Disease
Lowest	155	No
	180	Yes
	190	No
	220	Yes
Highest	225	Yes

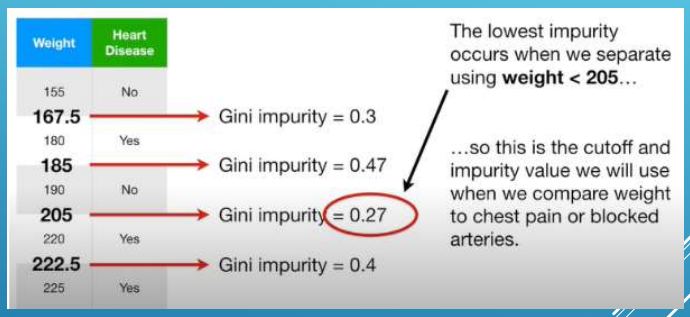
Step 1) Sort the patients by weight, lowest to highest.

NUMERIC VARIABLE: WHAT'S THE BEST WEIGHT USED TO DIVIDE THE PATIENT?

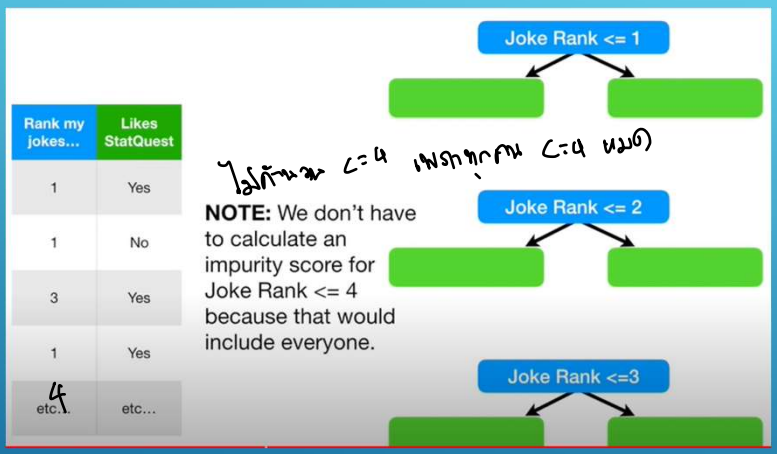
Step 2) Calculate the average weight for all adjacent patients.



Step 3) Calculate the impurity values for each average weight.



NUMERIC VARIABLE: WHAT'S THE BEST WEIGHT USED TO DIVIDE THE PATIENT?



ORDINAL VARIABLE

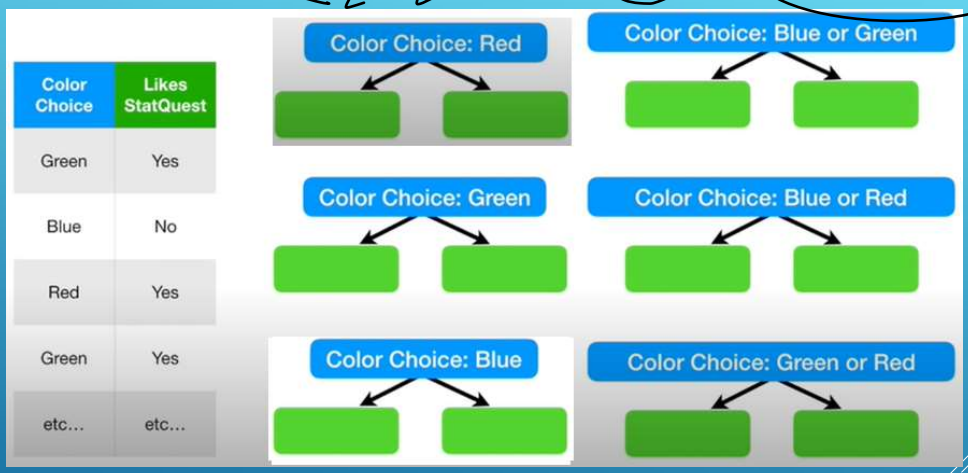
Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc...	etc...

When there are **multiple choices**, like "**color choice can be blue, green or red**", you calculate an impurity score for each one as well as each possible combination.

NOMINAL VARIABLE (1)

4 ค่าเป็นไปได้ 1 3 2 2 Green blue | Green | Yellow

Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc...	etc...

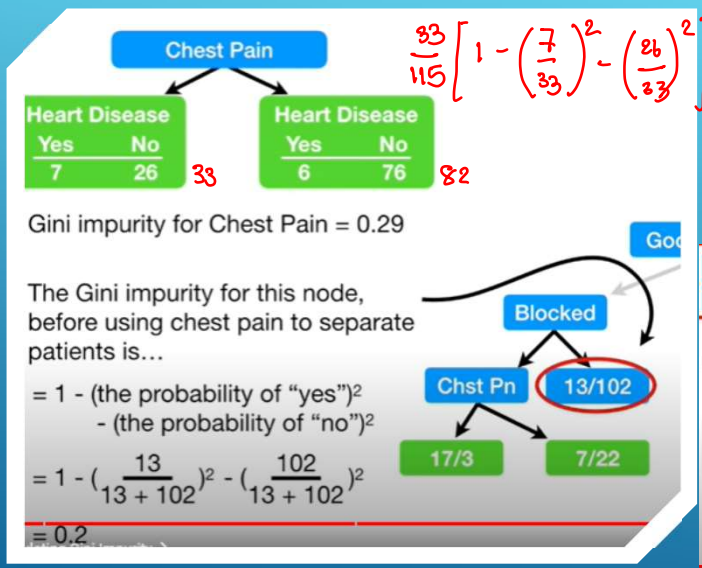


NOTE: We don't have to calculate an impurity score for "Color Choice: Blue or Green or Red" since that

NOMINAL VARIABLE (2)

- Pruning is a technique used to deal with overfitting, that reduces the size of DTs by removing sections of the Tree that provide little predictive or classification power.
- The goal is to reduce complexity and gain better accuracy by reducing the effects of overfitting and removing sections of the DT that may be based on noisy or erroneous data.
- There are two different strategies to perform pruning on DTs:
  - Pre-prune: When you stop growing DT branches when information becomes unreliable.
  - Post-prune: When you take a fully grown DT and then remove leaf nodes only if it results in a better model performance. This way, you stop removing nodes when no further improvements can be made.

TREE PRUNING bottom up



$\frac{33}{115}\left[1-\left(\frac{7}{33}\right)^2-\left(\frac{26}{33}\right)^2\right]+\frac{82}{115}\left[1-\left(\frac{6}{82}\right)^2-\left(\frac{76}{82}\right)^2\right]=0.29$

ใช้ root node

STOP SPLITTING WHEN NO FURTHER GAIN CAN BE MADE

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

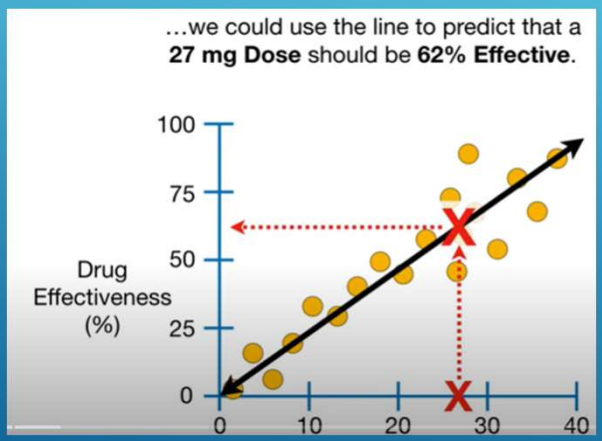
- Optimization of DT classifier performed by only pre-pruning using maximum depth of DT.
- **max_depth : int or None, (default=None) or Maximum Depth of a Tree**: If None, nodes are expanded until all the leaves contain less than min_samples_split samples. The higher value of maximum depth causes overfitting, and a lower value causes underfitting.

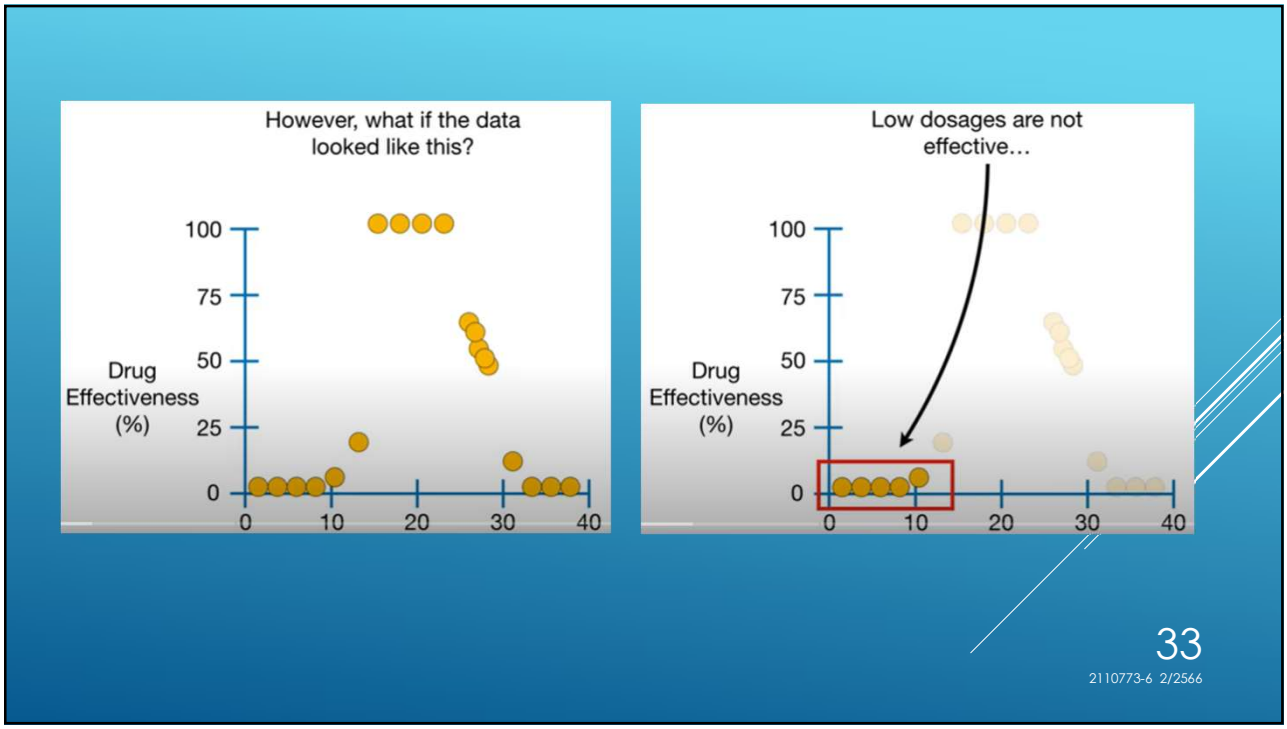
DECISION TREE (CLASSIFICATION) USING SCIKIT-LEARN

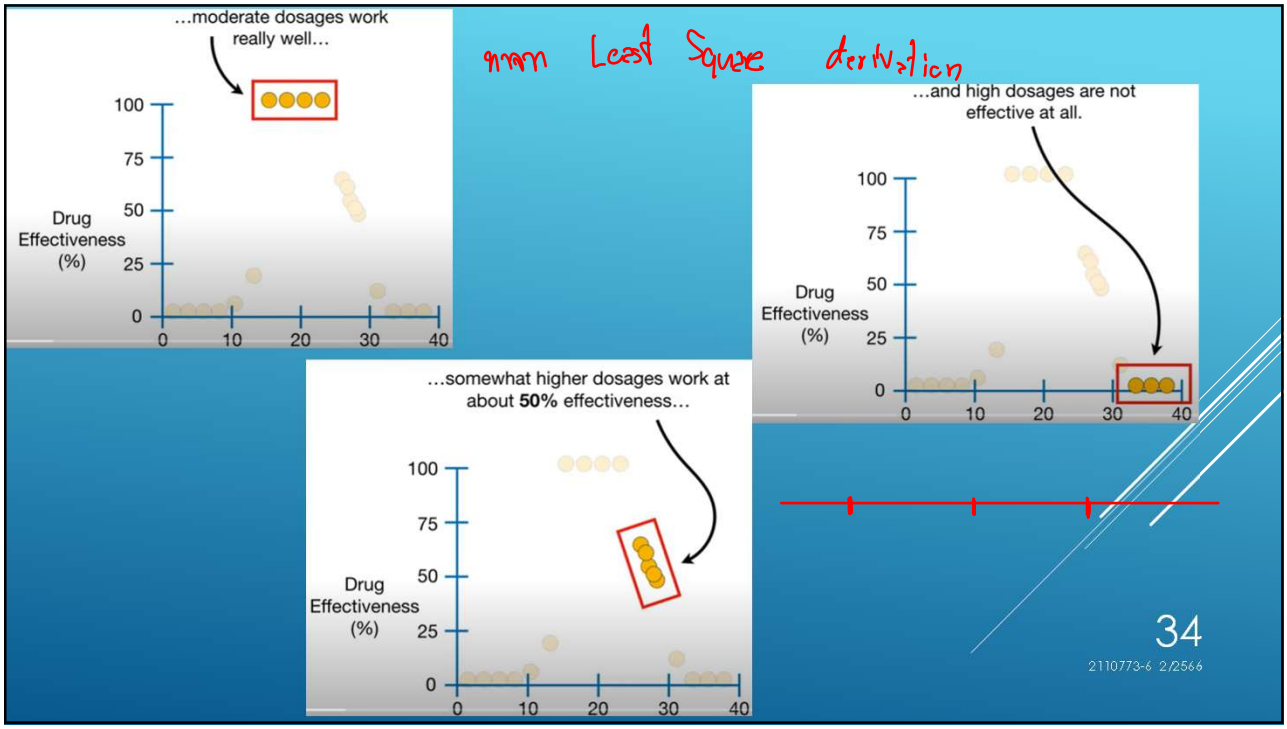
HTTPS://WWW.DATACAMP.COM/COMMUNITY/TUTORIALS/DECISION-TREE-CLASSIFICATION-PYTHON

LINEAR REGRESSION

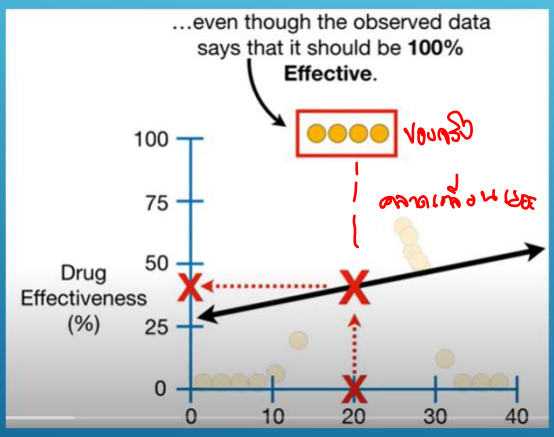
Easily fit a line to the data, the higher the dose, the more effective the drug...

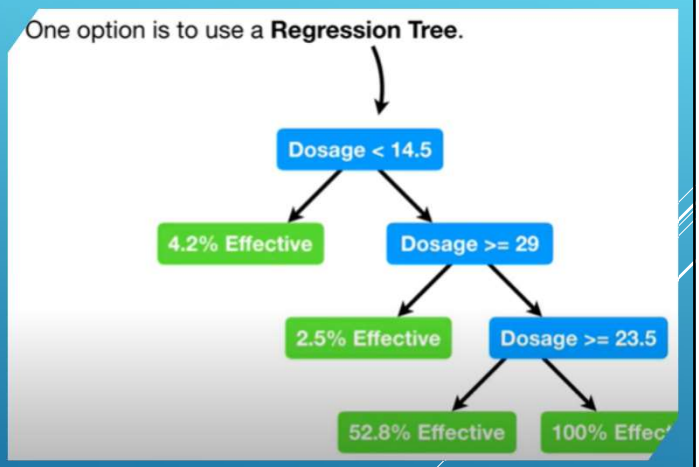



However, what if the data looked like this?
Drug Effectiveness (%)
100
75
50
25
0
0
10
20
30
40
Low dosages are not effective…
Drug Effectiveness (%)
100
75
50
25
0
0
10
20
30
40
33
2110773-6 2/2566


…moderate dosages work really well…
Drug Effectiveness (%)
100
75
50
25
0
0
10
20
30
40
mmm Least Square derivation
…somewhat higher dosages work at about 50% effectiveness…
Drug Effectiveness (%)
100
75
50
25
0
0
10
20
30
40
…and high dosages are not effective at all.
Drug Effectiveness (%)
100
75
50
25
0
0
10
20
30
40
34
2110773-6 2/2566

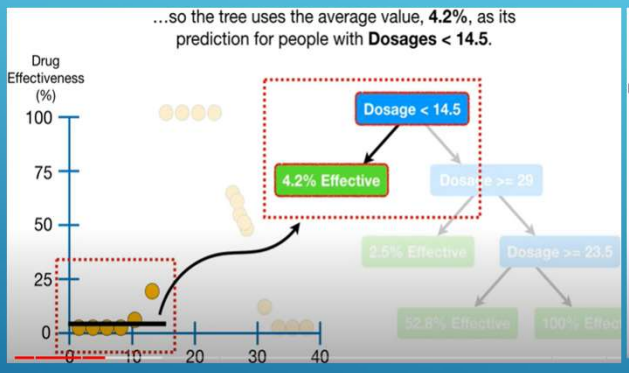
For example, if someone told us they were taking a 20 mg Dose…

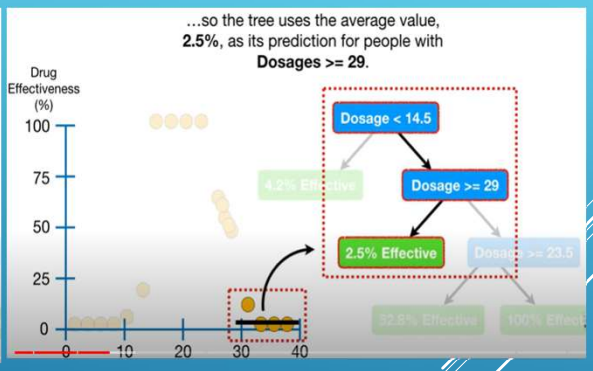




35

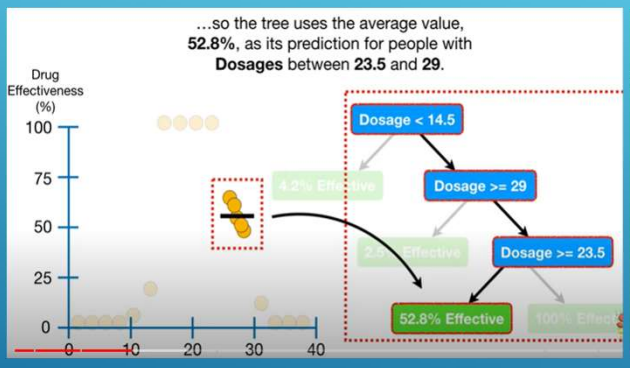
2110773-6 2/2566

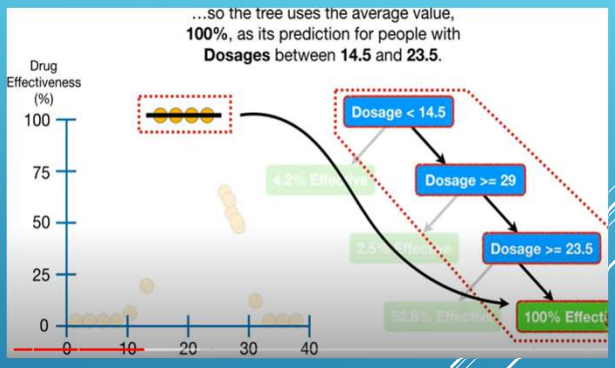




36

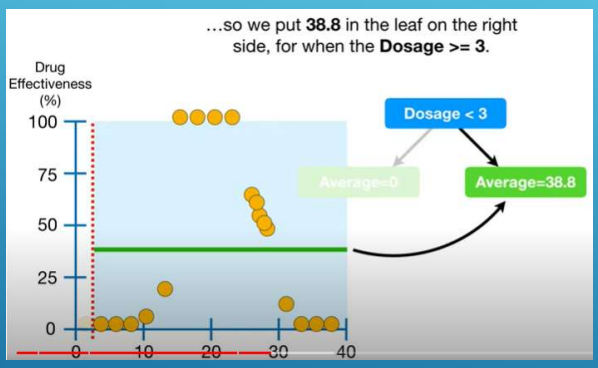
2110773-6 2/2566

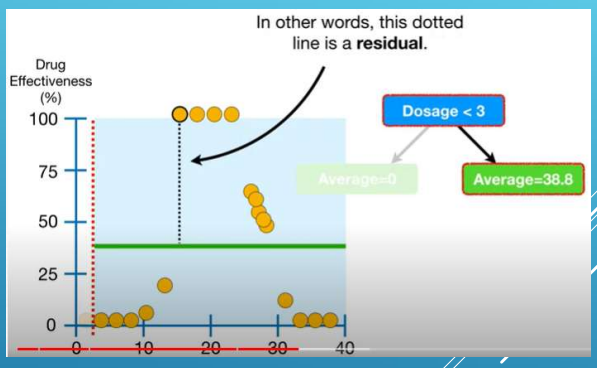




37

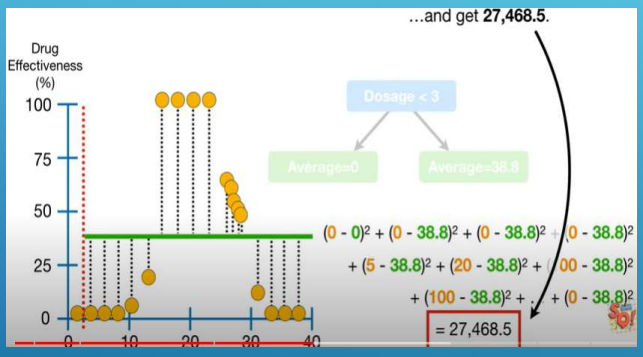
2110773-6 2/2566

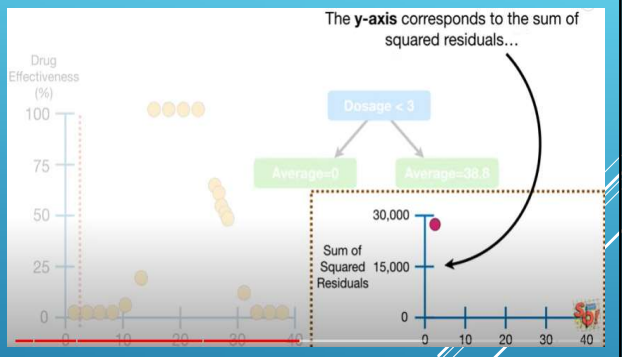




38

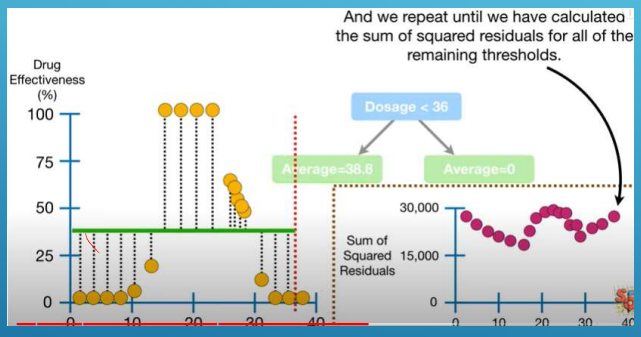
2110773-6 2/2566

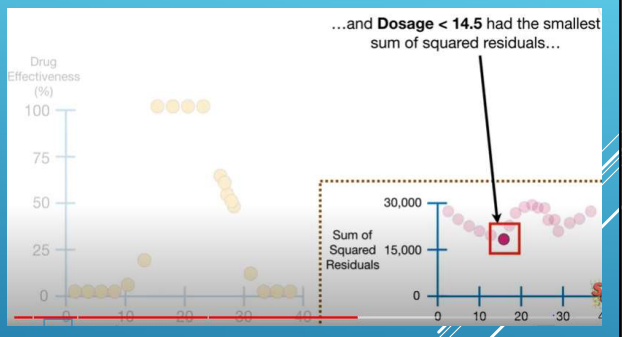




39

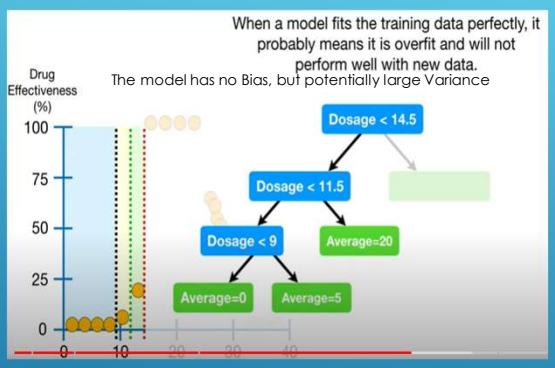
2110773-6 2/2566

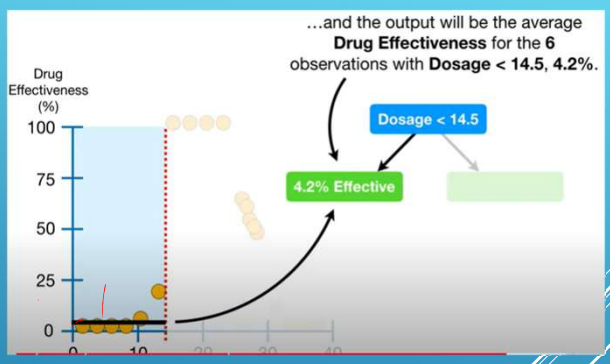




40

2110773-6 2/2566

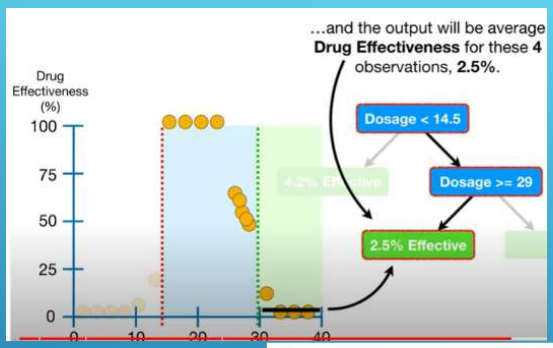


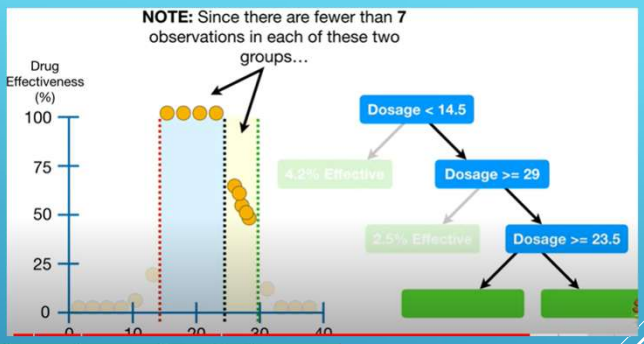


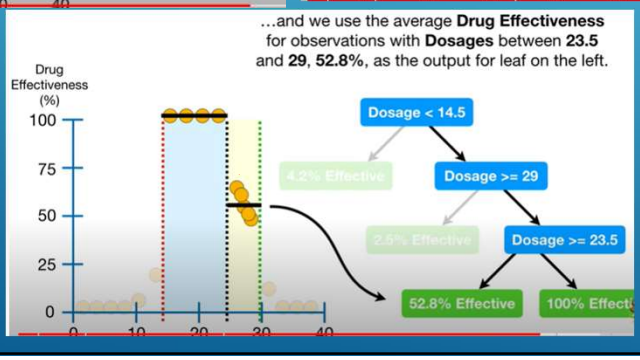
The simplest is to only split observations when there are more than some minimum number. Typically, the minimum number of observations to allow for a split is **20**.

However, since this example doesn't have many observations, I set the minimum to **7**.

41

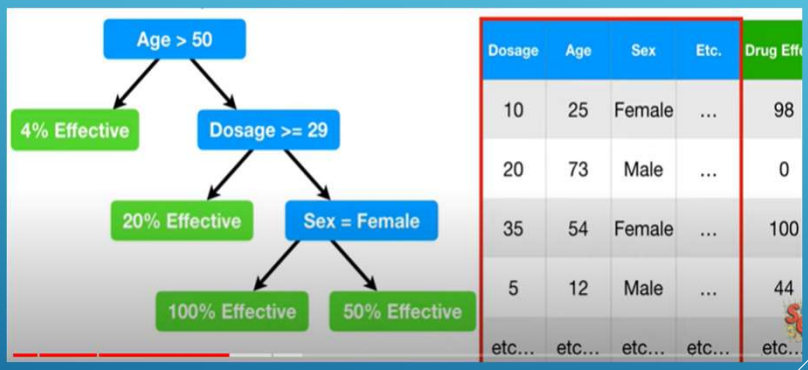






42

- A Regression tree looks for splits that minimize the Least Square Deviation (LSD), sometimes referred as "variance reduction", that implies the variance within the node.



REGRESSION TREE

Pro

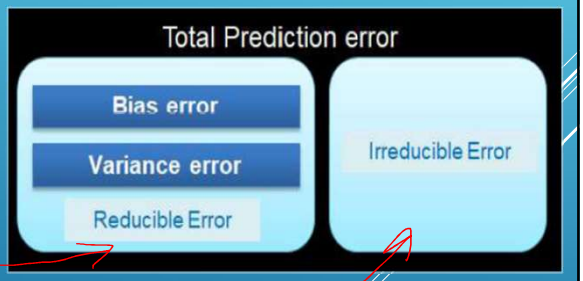
- easy to interpret and visualize
- easily capture Non-linear patterns with non-parametric nature of the algorithm.
- requires fewer data preprocessing, no need to normalize features
- **can be applied for variable selection**

Con

- Sensitive to noisy data. It can overfit noisy data.
- Biased with imbalanced dataset, balance out the dataset before creating DT is recommended balance dataset ก่อน
- **small variation(or variance) in data can result in different DT. This can be reduced by bagging and boosting algorithms.**

DT CLASSIFIER

- Every model has both bias and variance error components in addition to white noise.
- The ideal model will have both low bias and low variance.
- Bias and variance are inversely related to each other; while trying to reduce one component, the other component of the model will increase.
- The true art lies in creating a good fit by balancing both. balance & error



$$E\left(y_0 - \hat{f}(x_0)\right)^2 = Var\left(\hat{f}(x_0)\right) + \left[Bias(\hat{f}(x_0))\right]^2 + Var(\varepsilon)$$

Bias-error

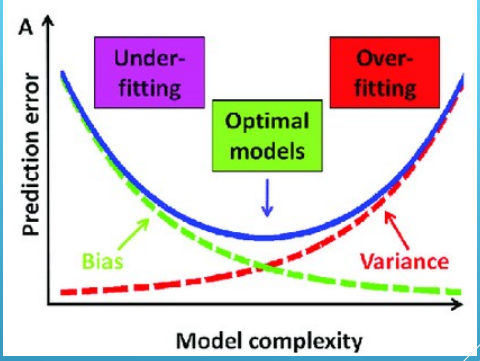
- Difference between predicted and actual data points caused by **oversimplified** model or unable to capture underlying pattern of data.
- It misses how the features in the training data set relate to the expected output.
- A model with high bias is too simple and has low number of predictors.

Variance-error

- High variance error of a model implies that it is highly sensitive to small fluctuations. This model flounders outside of its comfort zone(training data)
- Any model which has very large number of predictors will end up being a very **complex model** which will deliver very accurate predictions for the training data that it has seen already but this complexity makes the generalization of this model to unseen data very difficult, i.e. a high variance model. Thus, this model will perform very poorly on test data.

REDUCIBLE ERROR/ INADVERTENT MISTAKES

- On the one hand, we want our algorithm to model the training data very closely, otherwise we'll miss relevant features and interesting trends.
- However, on the other hand we don't want our model to fit too closely, and risk overinterpreting every outlier and irregularity.



Low Bias: Suggests less assumptions about the form of the target function.
High-Bias: Suggests more assumptions about the form of the target function.

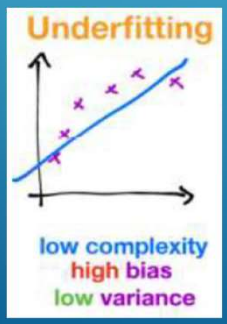
Low Variance: Suggests small changes to the estimate of the target function with changes to the training dataset.
High Variance: Suggests large changes to the estimate of the target function with changes to the training dataset.

BIAS-VARIANCE DILEMMA

47

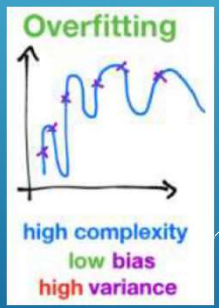
Bias-error

- Difference between predicted and actual data points caused by **oversimplified** model or unable to capture underlying pattern of data.
- A model with high bias is too simple and has low number of predictors.

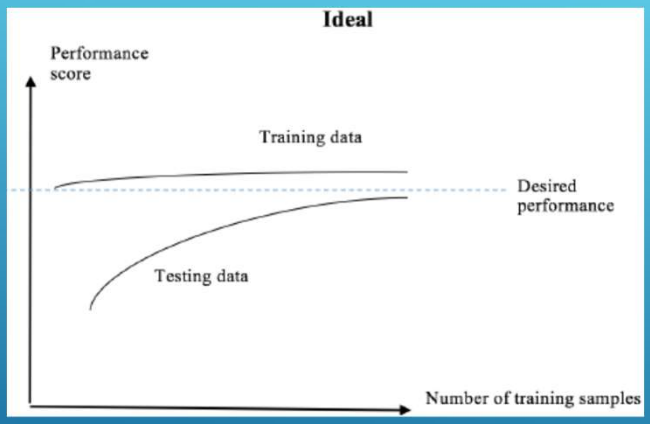


Variance-error

- High variance error implies highly sensitive to small fluctuations.
- Any model containing very large number of predictors will end up being a very **complex model** which flounders outside its comfort zone(training data)



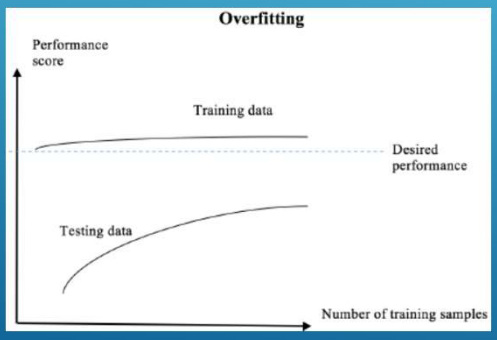
48



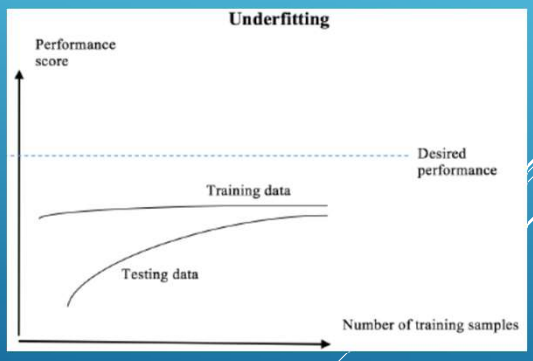
- A **learning curve** is usually used to evaluate the bias and variance of a model.
- For a model that fits well on the training samples, the performance of training samples should be above desire. Ideally, as the number of training samples increases, the model performance on testing samples improves; eventually the performance on testing samples becomes close to that on training samples.

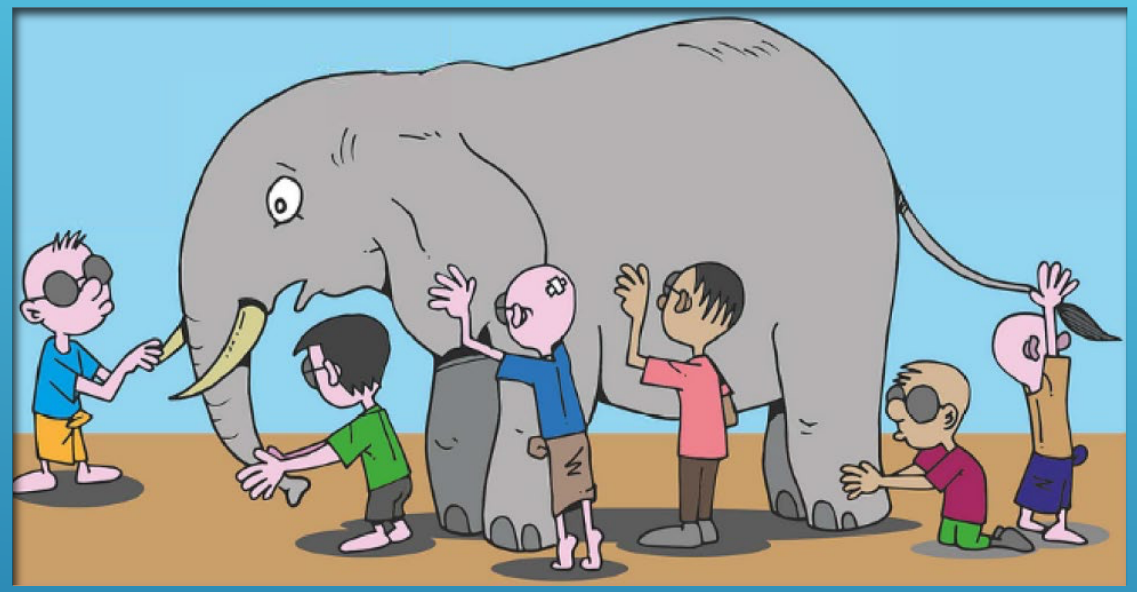
DIAGNOSING OVERFITTING AND UNDERFITTING

When the performance on testing samples converges at a value far from the performance on training samples, overfitting can be concluded. In this case, the model fails to generalize to instances that are not seen.



For a model that does not even fit well on the training samples, underfitting is easily spotted: both performances on training and testing samples are below desire in the learning curve.



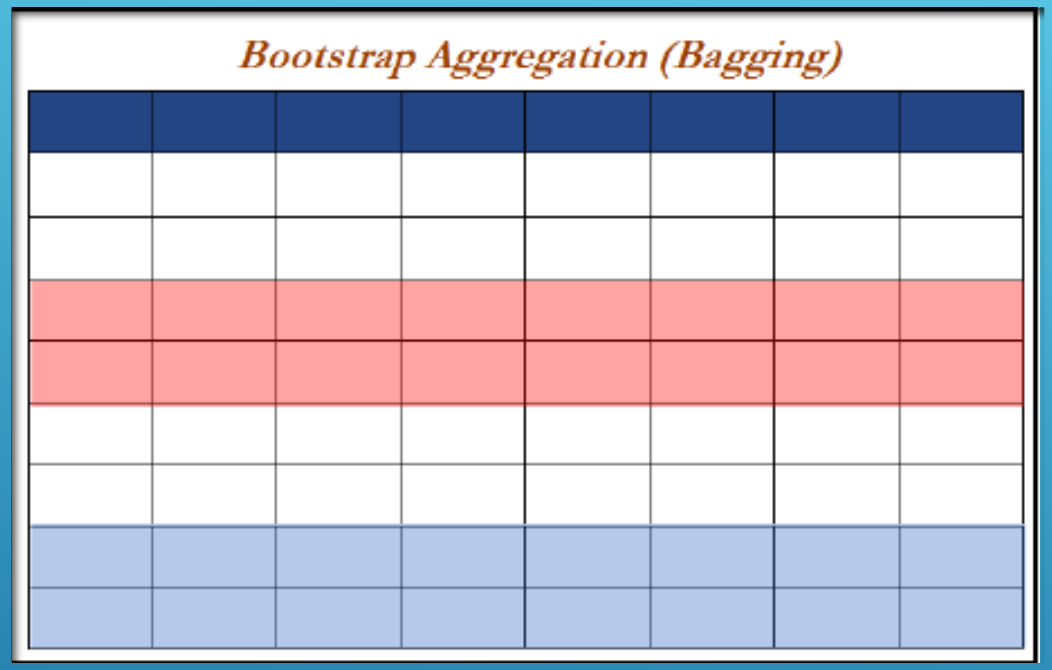


- Combine decisions from multiple models to improve overall performance
- Help minimize causes of error due to noise, bias and variance
- Major schemes:
  - Bagging
  - Boosting

PART2: DECISION TREE ENSEMBLES

หมู่คณะ

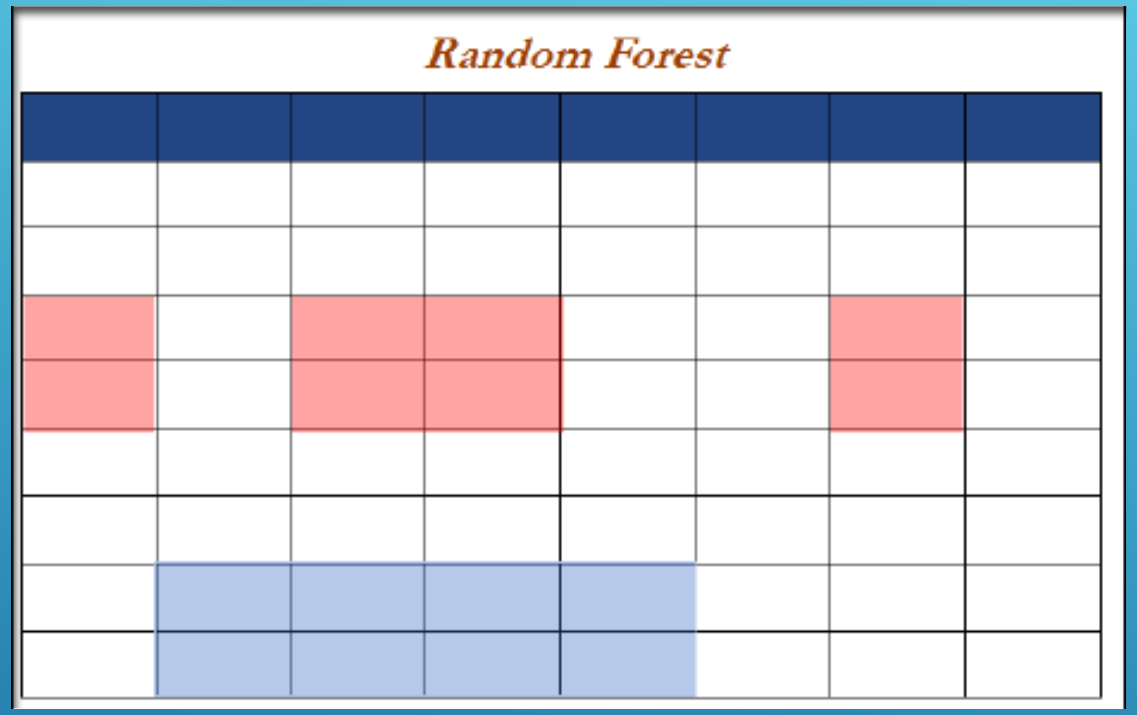
BAGGING



Two samples (pink, blue) with all variables

ยังคงเป็น attribute ชุดเดิมทุกต้น

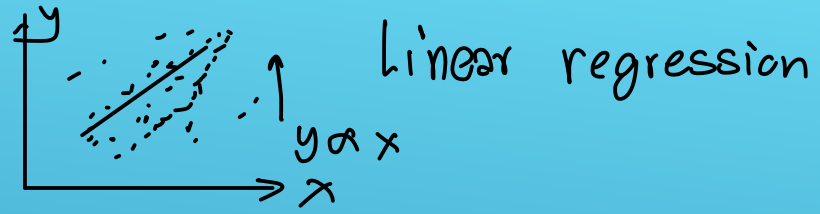
- **Bootstrap aggregation** or **bagging** introduced by Leo Breiman in 1994.
- Bootstrapped datasets are created by **sampling with replacement**.
- Build a number of decision trees on bootstrapped samples from training data ใช้ทุก column แต่ละ sample สร้างต้นไม้
- Combine the results of the models by **averaging** or **majority voting** ต้นไม้โหวตทุกต้น
- The algorithm aims to reduce the chance of overfitting. ลดโอกาส overfit
- Due to all variables selected, order of candidate/variable chosen to split remains more or less the same for all the individual trees.
- Variance reduction on correlated individual entities does not work effectively while aggregating them.



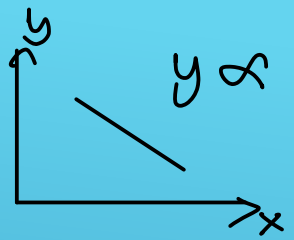
RANDOM FORESTS

RF พยายามลด feature Variance สร้างจาก feature ต่างกันจะดีกว่า

- In bagging, a random sample with replacement is selected to train every tree in the ensemble. The tree is trained on all the features.
- While each decision tree in the random forest is given a randomly selected subset of features and a randomly selected subset of the dataset for the selected features **to ensure low correlation among decision trees.** sure ว่าลดความสัมพันธ์ต่อกัน ต่างข้อมูล ต่าง variance เพิ่มความหลากหลาย



linear regression



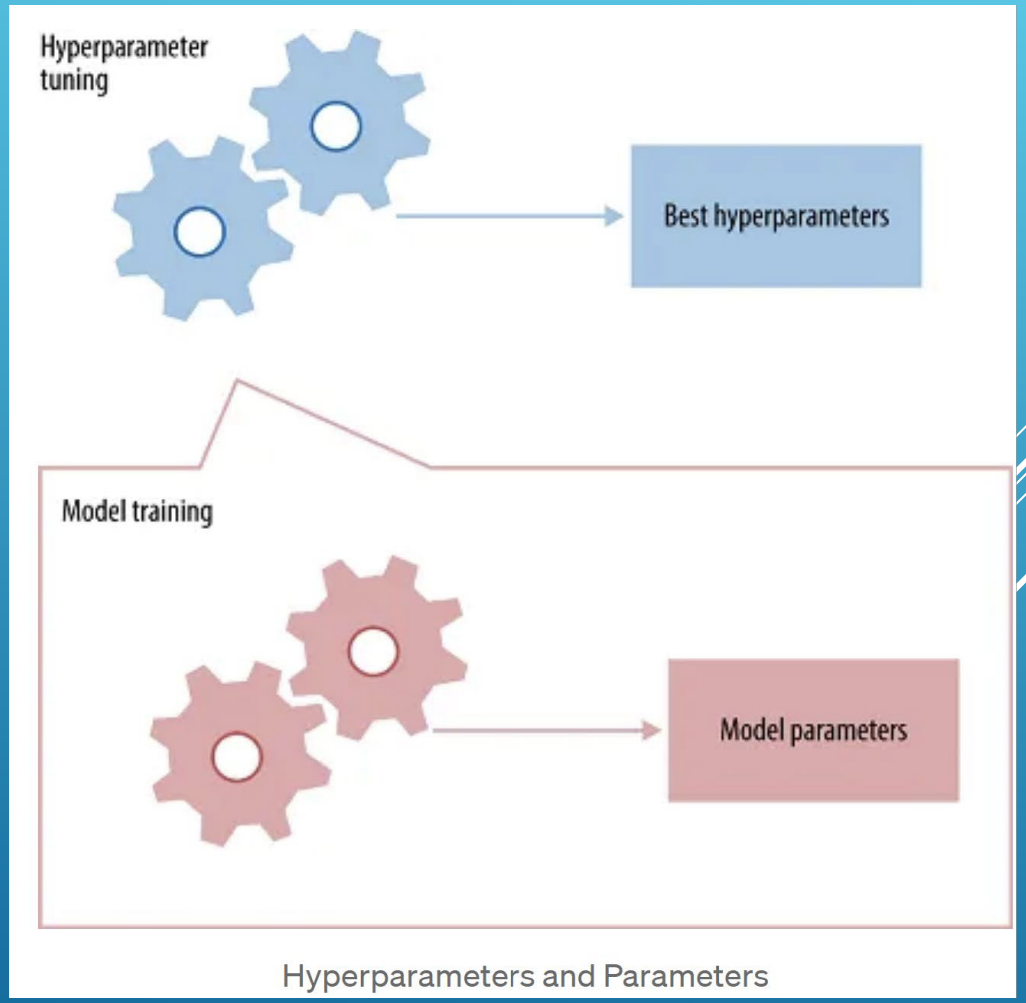
$y = mx + c$ → ค่าคงที่ของ y

slope

y - intercept

PARAMETER VS. HYPERPARAMETER

- model ***parameters*** are learned during training — such as the *slope* and *intercept* in a linear regression ใช้ระหว่าง train
- ***hyperparameters*** must be set by the data scientist **before** training ปรับ tuning ก่อน train
- Scikit-Learn implements a set of sensible default hyperparameters for all models, but these are not guaranteed to be optimal for a problem.
- Hyperparameter tuning relies more on experimental results than theory, and thus the best method to determine the optimal settings is to try many different combinations evaluate the performance of each model.



Hyperparameters and Parameters

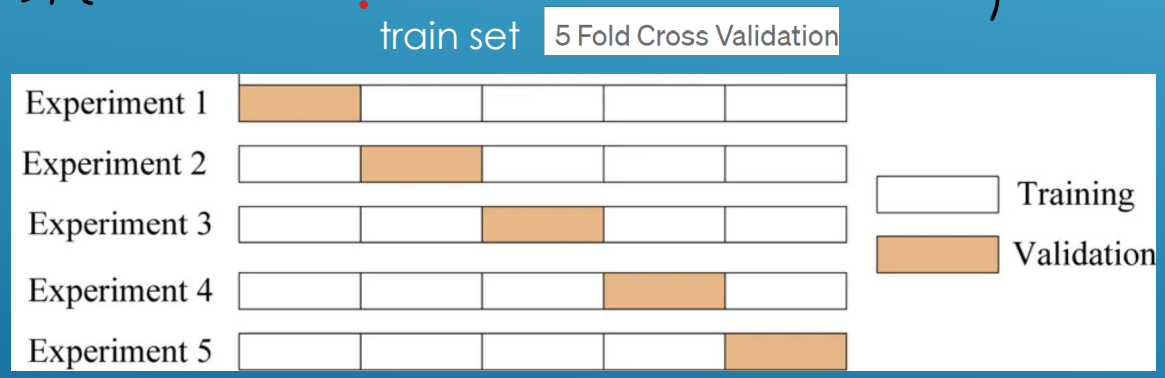
- n_estimators: number of trees considered for majority voting in the forest. The more trees, the better the performance, but more computation time. It is usually set as 100, 200, 500, and so on. จำนวนต้นไม้ที่ต้องการ
- max_features: max number of features consider for each best splitting point search. Typically, for an m-dimensional dataset, rounded sqrt(m) is a recommended value for max_features. This can be specified as max_features="sqrt" in scikit-learn. จำนวน feature แต่ละต้นที่ใช้ [ปัจจุบัน feature ทั้งหมด]
- max_depth: max number of levels in each DT. It tends to overfit if it is too deep, or to underfit if it is too shallow. ต้นไม้เตี้ย underfit สูงไป overfit
- min_samples_split: min number of data points placed in a node before the node is split. Too small a value tends to cause overfitting, while too large a value is likely to introduce underfitting. 10, 30, and 50 might be good options to start with. < 7 ก็ไม่ split
- min_samples_leaf = min number of data points allowed in a leaf node จำนวนใบที่น้อยที่สุด
- bootstrap = method for sampling data points (with or without replacement)
- Practically, application of **grid search** on tuning different combinations of hyperparameters will provide better and more robust results.

HYPERPARAMETERS IN RANDOM FOREST

grid k = {1, 2, 3, ..., 10} ทุกค่า
randomize : range (1 - 10) มีบางค่าไม่เอาไป test

- For hyperparameter tuning, many iterations of K-Fold CV process are performed, each time using different model settings. Then, compare all of the models, select the best one, train it on the full training set, and then evaluate on the testing set.
- If we have 10 sets of combinations of hyperparameters and are using 5-Fold CV, that represents 50 training loops.
- Fortunately, model tuning with K-Fold CV can be automatically implemented in Scikit-Learn.
- Using SK-Learn's RandomizedSearchCV method will randomly sample from the defined grid of hyperparameter ranges, and, performing K-Fold CV with each combination of values.
- GridserchCV method

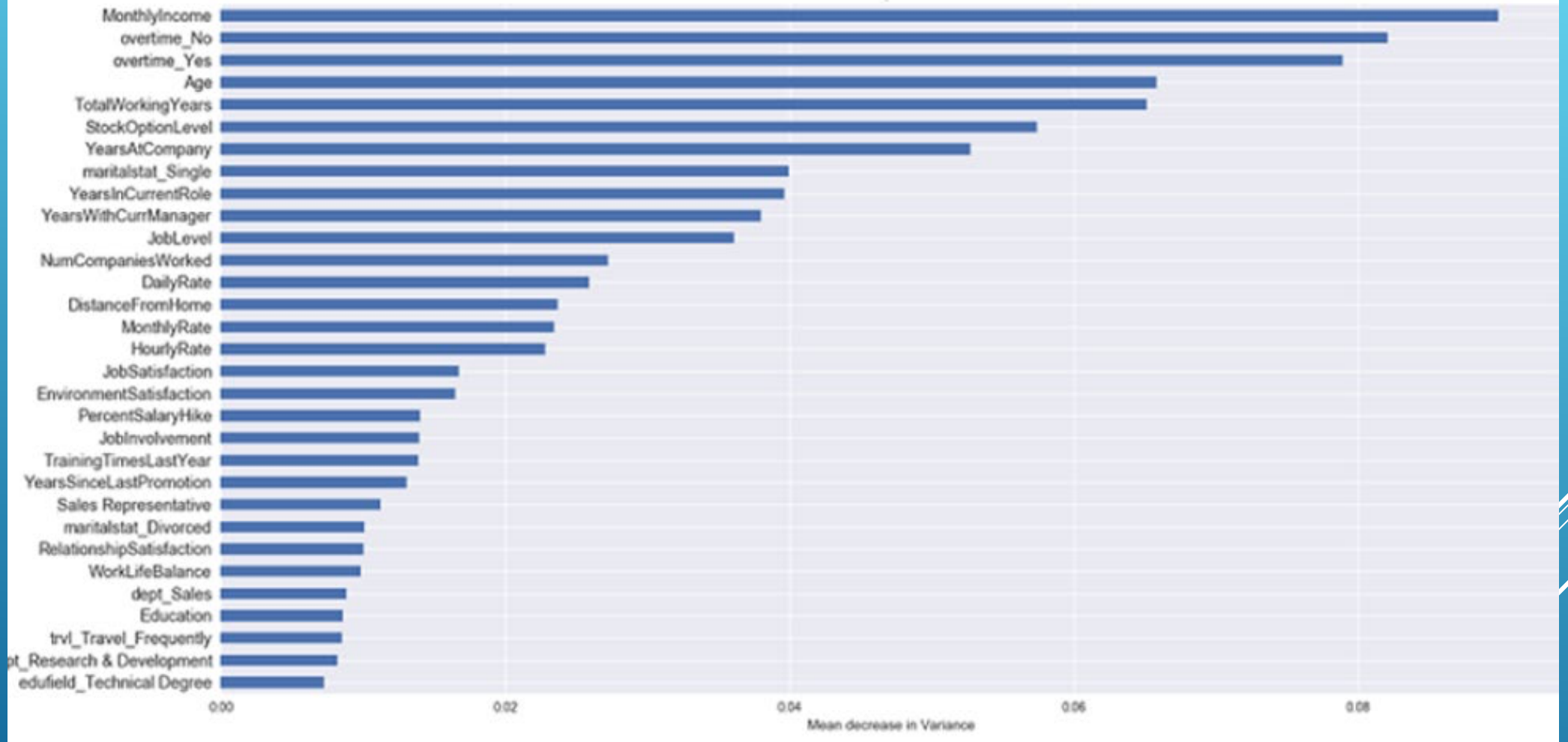
hp1 hp2 → 10 combination ที่ต้องหาว่า combination ไหนได้ค่าดีสุด
2 5



1 combination วน 10 รอบ

HYPERPARAMETER TUNING WITH CROSS VALIDATION

Variable Importance Plot from RF



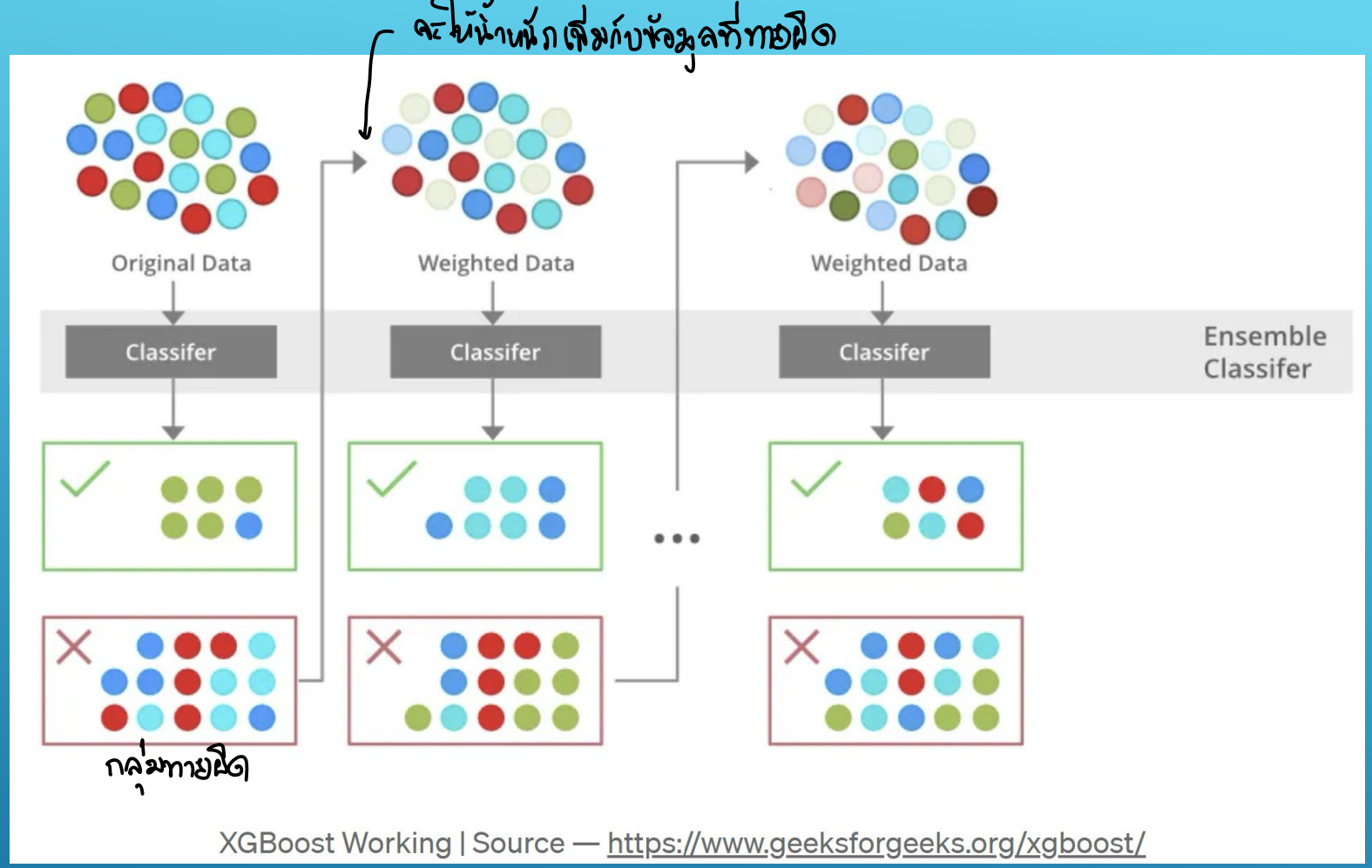
BOOSTING

- In boosting, all models are trained in sequence, instead of in parallel as in bagging.
- Each model is trained on the same dataset, but each data sample is under a different weight factoring, in the previous model's success.
- The weights are reassigned after a model is trained, which will be used for the next training round.
- In general, weights for mispredicted samples are increased to stress their prediction difficulty.
- There are many boosting algorithms e.g. AdaBoost, Gradient Boosting and XGBoost; boosting algorithms differ mostly in their weighting scheme.
- Boosting relies on creating a series of weak learners each of which might not be good for the entire data set but is good for some part of the data set. Thus, each model actually boosts the performance of the ensemble.
- Boosting has shown better predictive accuracy than bagging.

XGBOOST

- XGBoost is a gradient boosting algorithm, originally developed by *Tianqi Chen*.
- It works by combining a number of weak learners to form a strong learner.
- XGBoost works by training a number of decision trees. Each tree is trained on a subset of the data, and the predictions from each tree are combined to form the final prediction.
- A number of most important ~~parameters~~ hyperparameter include:

- ***max_depth:*** The maximum depth of the decision trees.
- ***eta:*** The learning rate. η อัตราการเรียนรู้ มักเป็นค่าน้อย ๆ e.g. .01 , .001
- ***gamma:*** The minimum loss reduction required to make a split.
- ***subsample:*** The fraction of the training data that is used to train each tree.



XGBoost Working | Source — https://www.geeksforgeeks.org/xgboost/

XGBoost has been shown to outperform other machine learning algorithms in a variety of tasks, including classification, regression and ranking.

DIFFERENCES BETWEEN BAGGING AND BOOSTING

	Bagging	Boosting
Differences	Individual models are built separately สร้างบน subset of dataset subset of attribute	Each new model is influenced by the performance of those built previously รับอิทธิพลจากต้นไม้ก่อนหน้า
	Equal weight is given to all models weight เท่ากันทุกต้น	Weights a model's contribution by its performance

CART → Classification and Regression Tree

ผู้บริหารเข้าใจความหมาย ตีความได้ง่าย

- Produce rules in simple English sentences, easily interpreted and presented to senior management without any editing.
- DT can be applied to either classification or regression problems.
- Able to handle both numerical and categorical variables.
- DT is a non-parametric model. ไม่ต้องระบุว่าข้อมูล underlying หรือ distribution เป็นอย่างไร
- No assumptions are made on the underlying distribution of the data.
- Useful in data exploration: DT is one of the fastest ways to identify the most significant variables.
- Overfitting/ high variance error is one of the most practical difficulties for DT models. The problem can be solved by pruning and ensemble techniques.

เวลาไปทำนายกับข้อมูลชุดใหม่ (ข้อมูลเปลี่ยนไป) จะมีโอกาสทายผิด (overfitting) แก้โดยการ pruning