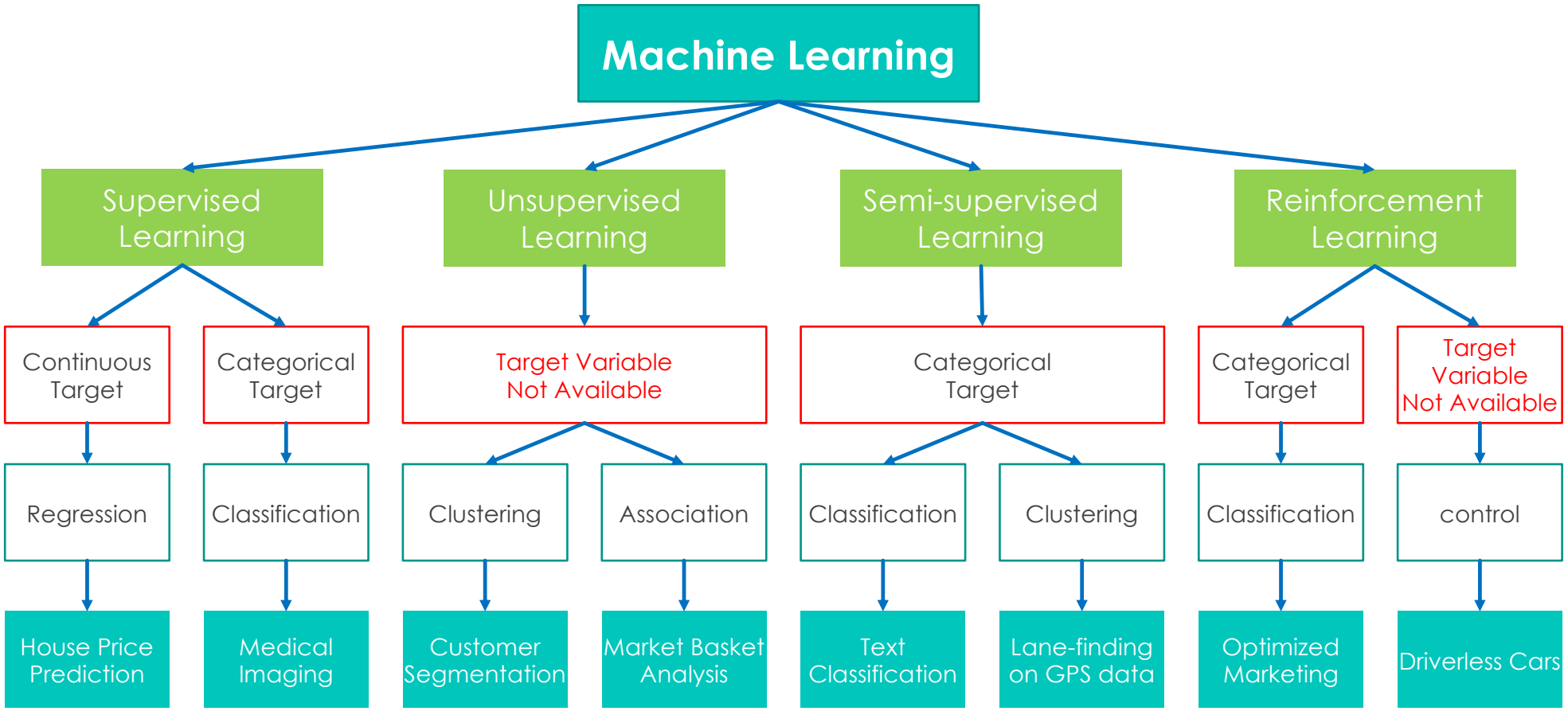
4주차 모델링

Introduction Machine Learning

Machine Learning



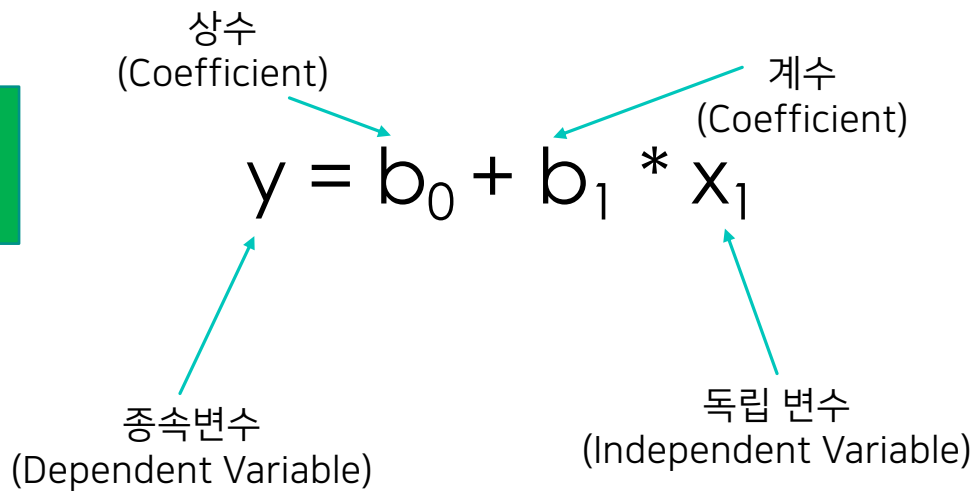
Deep Learning?

Layered Perceptron

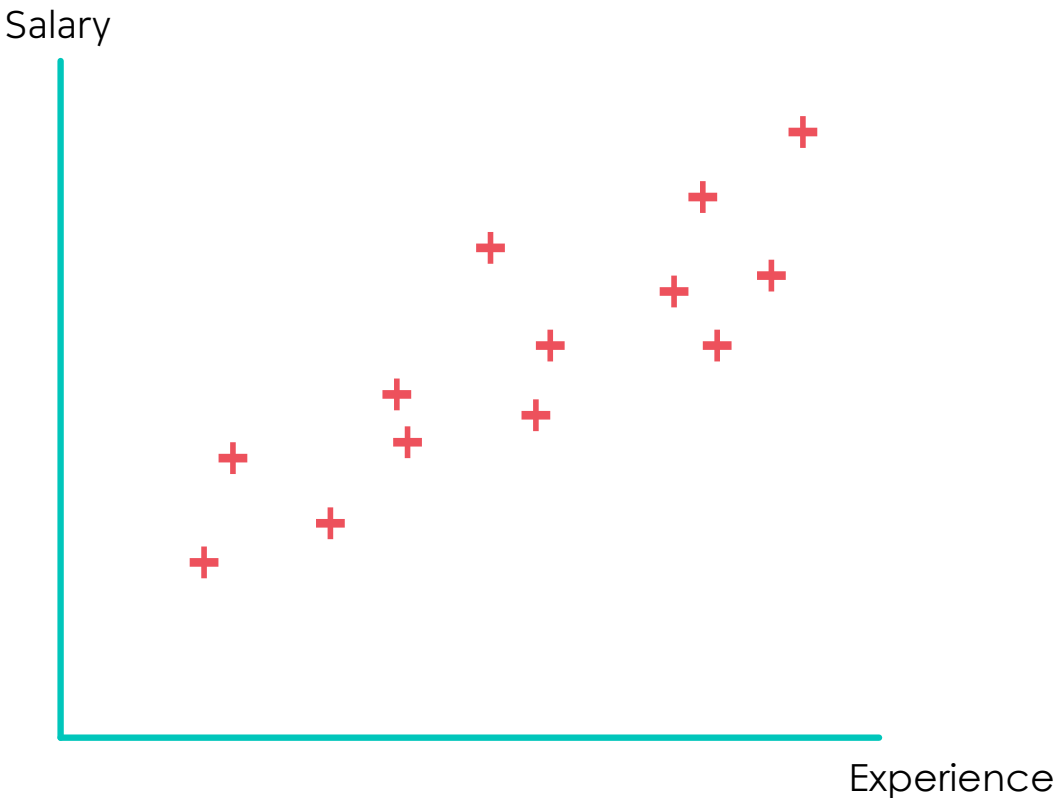
01. Linear Regression

Linear Regression Concept

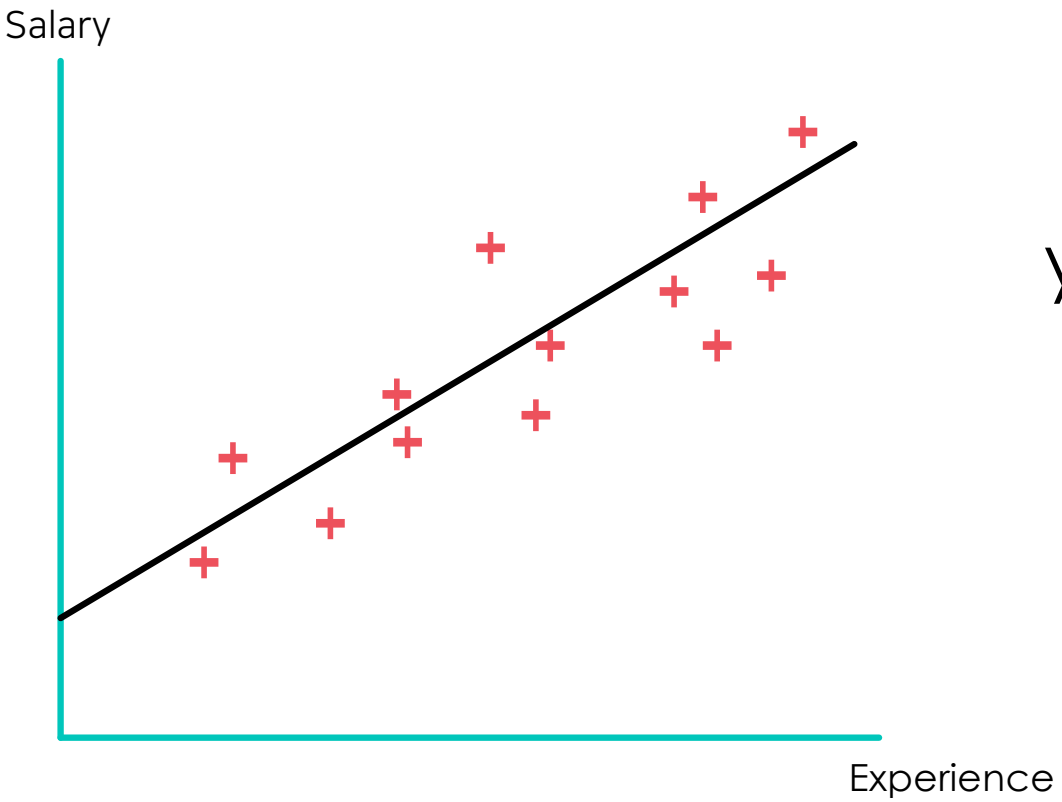
Univariate Linear Regression



Linear Regression Concept

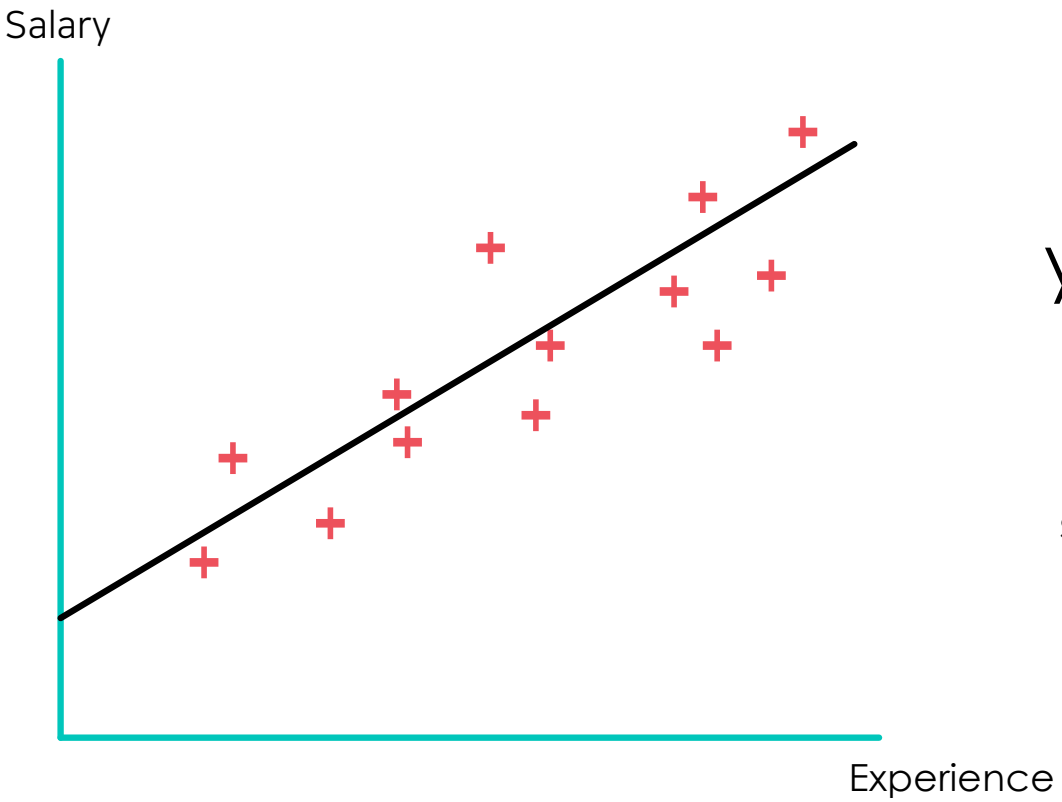


Linear Regression Concept



$$y = b_0 + b_1 * x_1$$

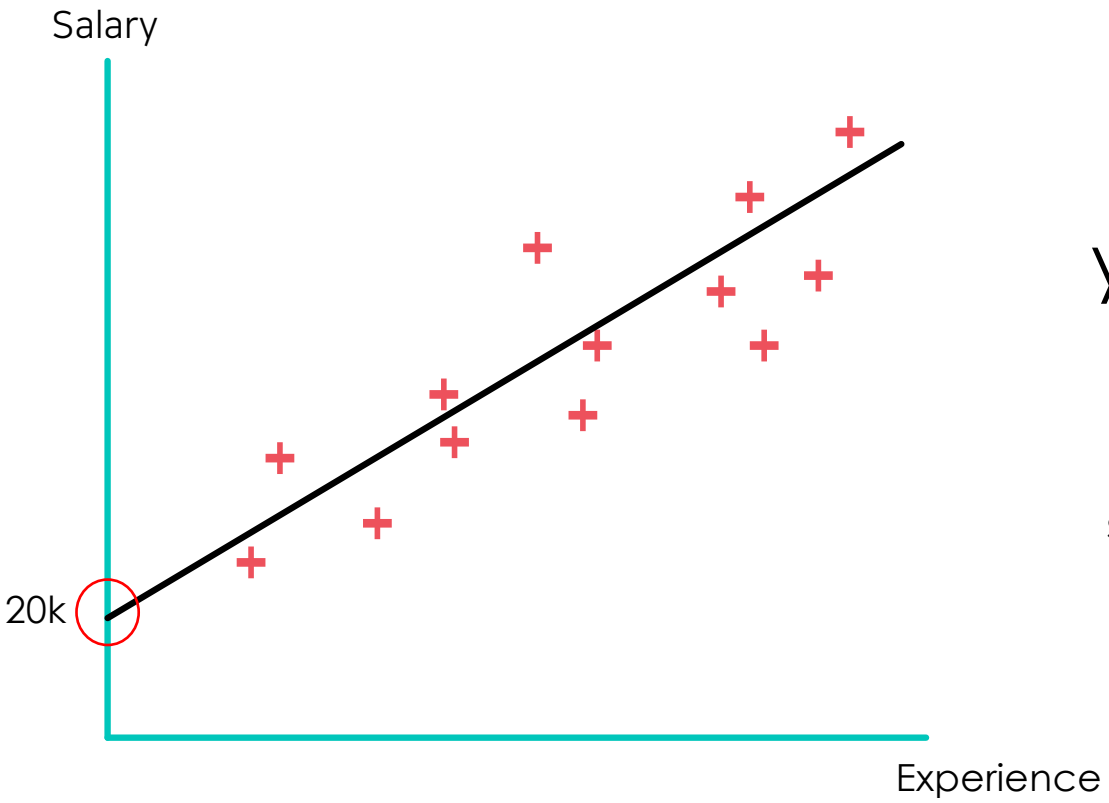
Linear Regression Concept



$$y = b_0 + b_1 * x_1$$

salary = b_0 + b_1 * Experience

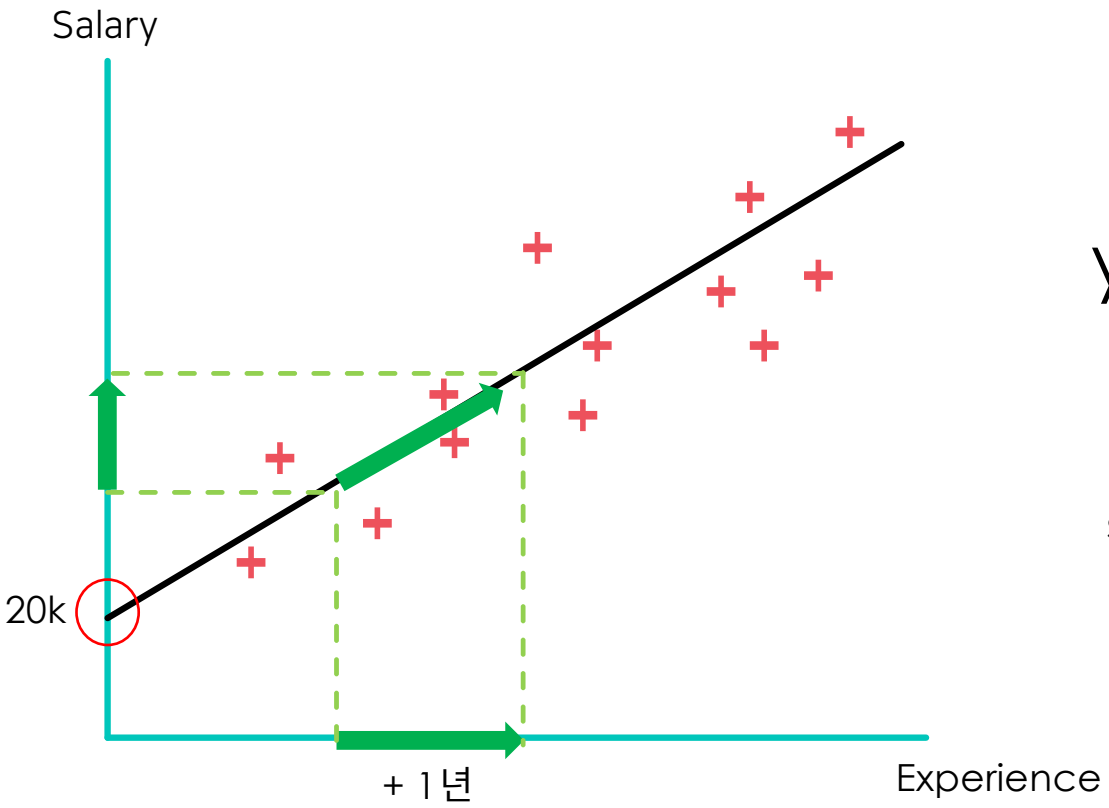
Linear Regression Concept



$$y = b_0 + b_1 * x_1$$

salary = b_0 + b_1 * Experience

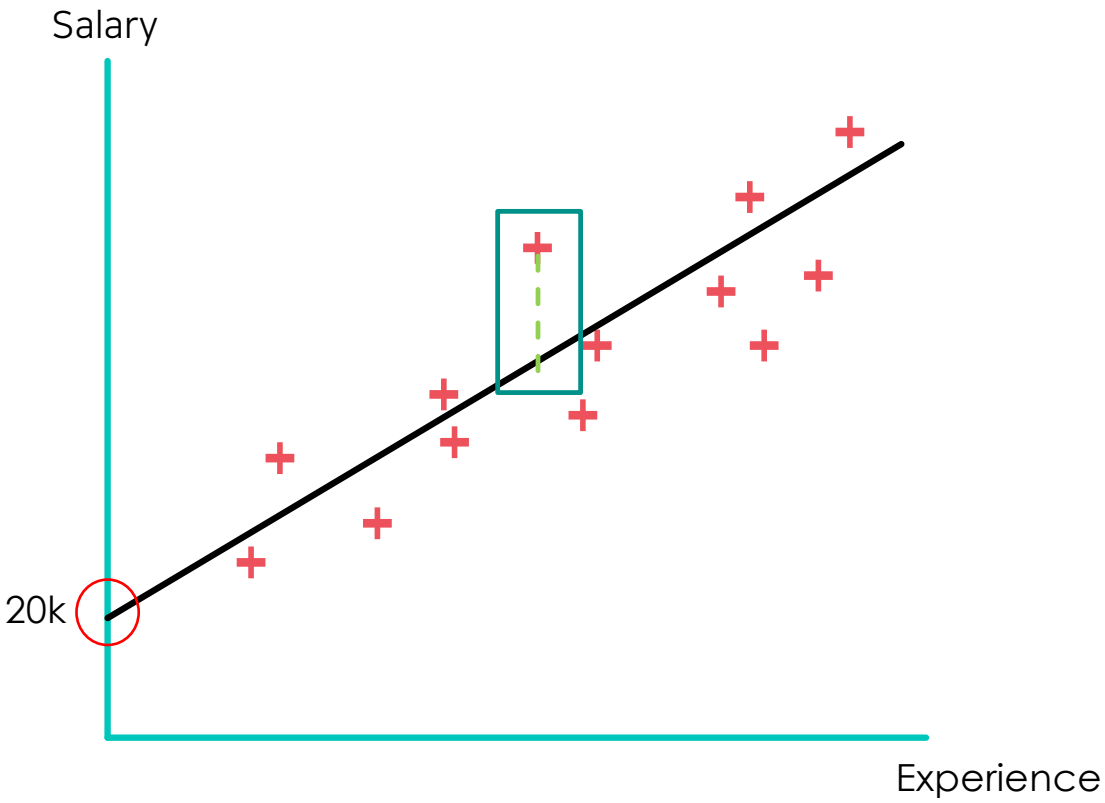
Linear Regression Concept



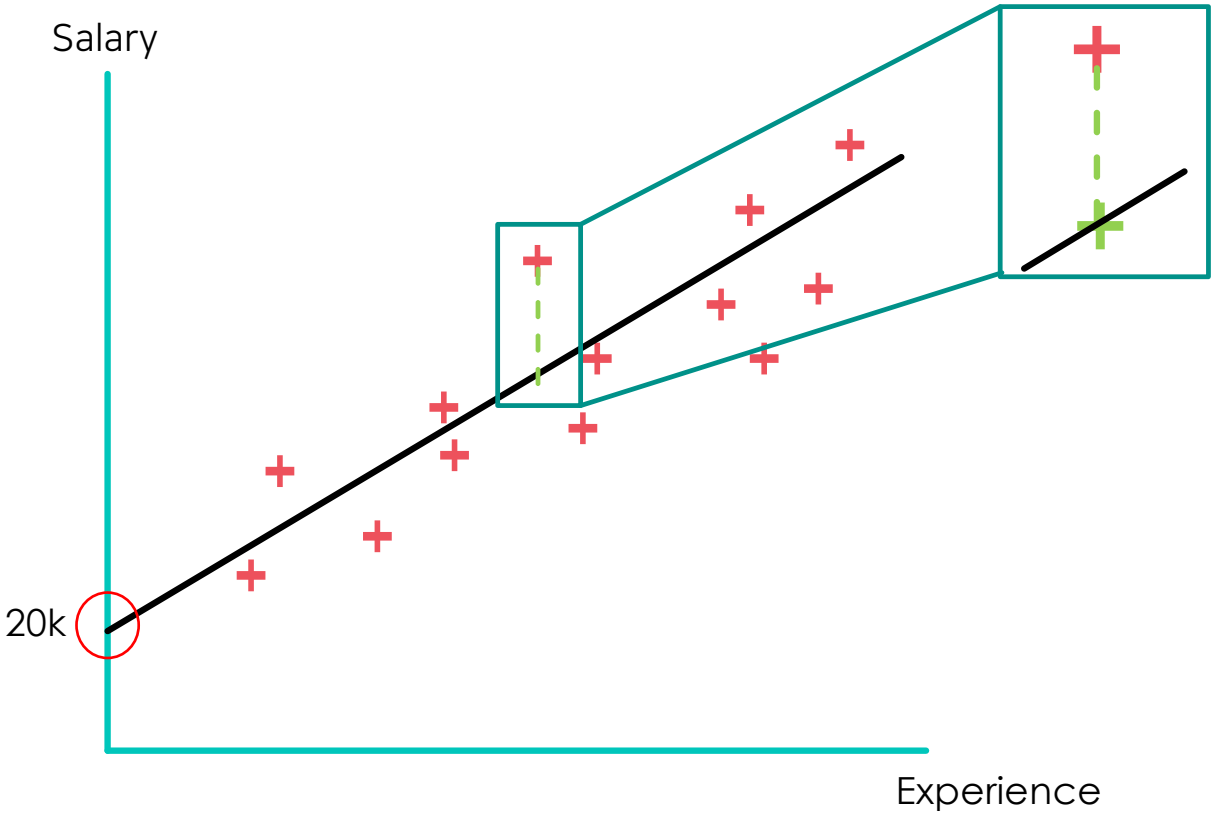
$$y = b_0 + b_1 * x_1$$

salary = b_0 + b_1 * Experience

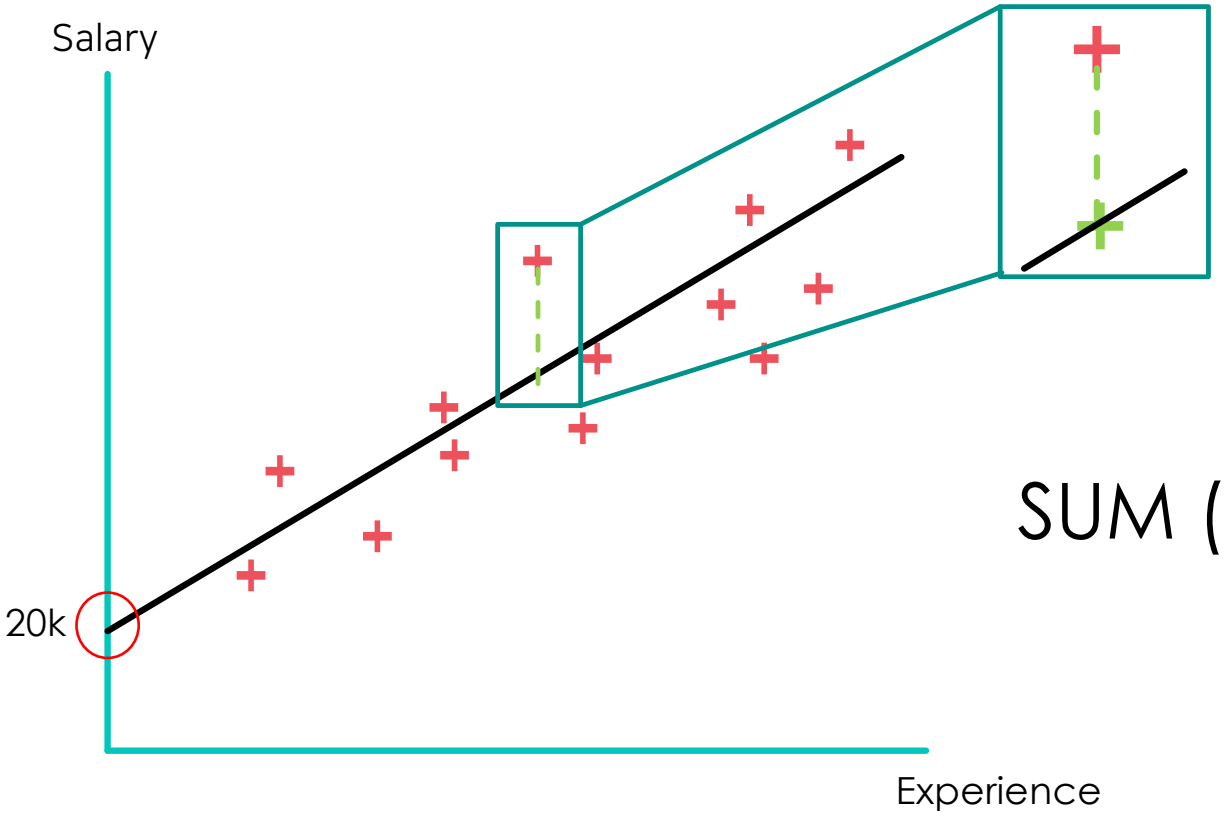
Linear Regression Concept



Linear Regression Concept



Linear Regression Concept



$$SUM\ (y - y\hat{})^2$$ ➡ 최소화

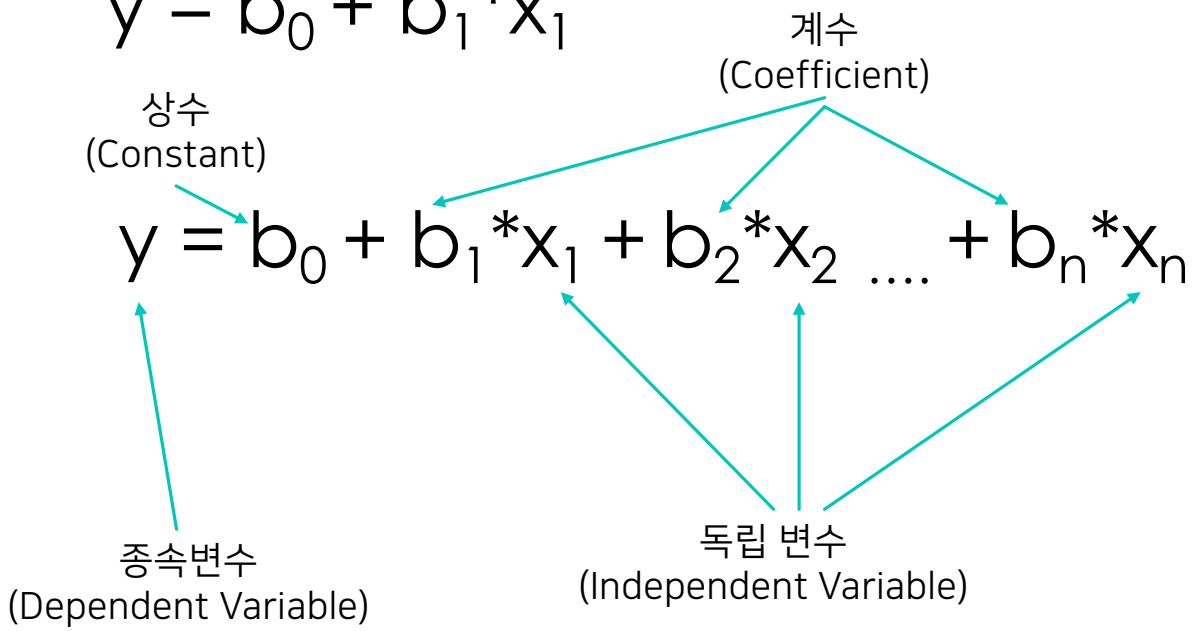
Multivariate Linear Regression Concept

Univariate Linear Regression

Multivariate Linear Regression

$$y = b_0 + b_1 * x_1$$

$$y = b_0 + b_1 * x_1 + b_2 * x_2 \ .... + b_n * x_n$$



Multivariate Linear Regression Concept – 범주형 변수

범주형(Category)

Profit	R&D Spend	Admin	Marketing	State
192261.83	165349.2	136897.8	471784.1	New York
191792.06	162597.7	151377.59	443898.53	California
191050.39	153441.51	101145.55	407934.54	Florida
182901.99	144372.41	118671.85	383199.62	New York
166187.94	142107.34	91391.77	366168.42	Florida

Multivariate Linear Regression Concept – 범주형 변수

범주형(Category)

Profit	R&D Spend	Admin	Marketing	State
192261.83	165349.2	136897.8	471784.1	New York
191792.06	162597.7	151377.59	443898.53	California
191050.39	153441.51	101145.55	407934.54	Florida
182901.99	144372.41	118671.85	383199.62	New York
166187.94	142107.34	91391.77	366168.42	Florida

One-Hot-Encoding

New York	California
1	0
0	1
0	1
1	0
0	1

Multivariate Linear Regression Concept – 범주형 변수

범주형(Category)

Profit	R&D Spend	Admin	Marketing	State
192261.83	165349.2	136897.8	471784.1	New York
191792.06	162597.7	151377.59	443898.53	California
191050.39	153441.51	101145.55	407934.54	Florida
182901.99	144372.41	118671.85	383199.62	New York
166187.94	142107.34	91391.77	366168.42	Florida

One-Hot-Encoding

New York	California
1	0
0	1
0	1
1	0
0	1

1 : New York
2 : California

State
1
2
2
1
2

Polynomial Regression

Univariate Linear Regression

$$y = b_0 + b_1 * x_1$$

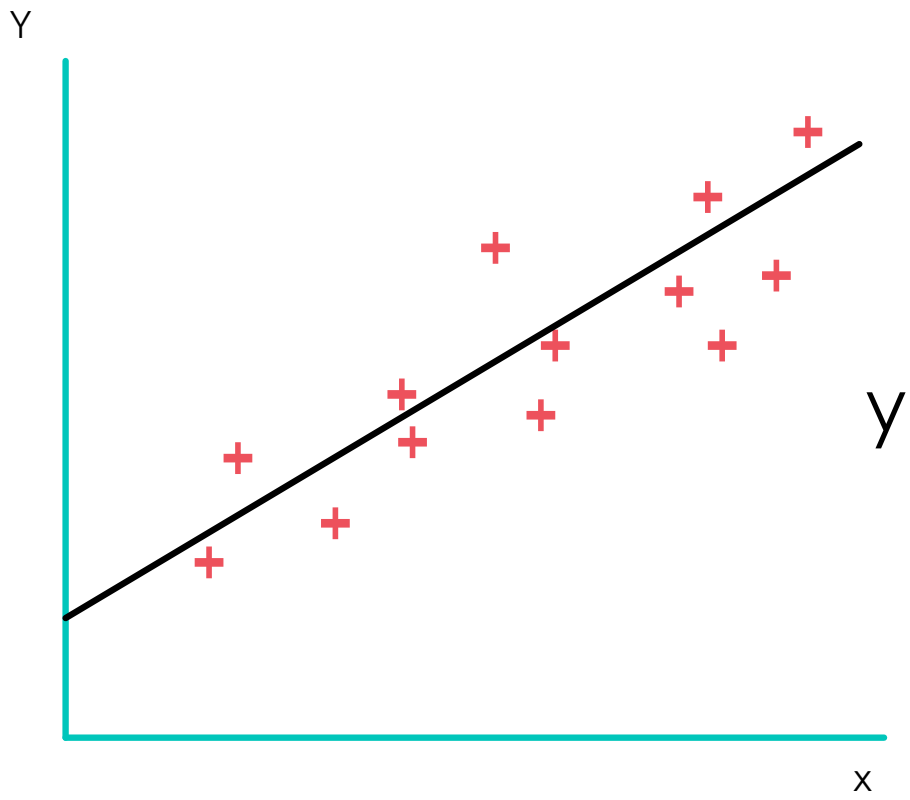
Multivariate Linear Regression

$$y = b_0 + b_1 * x_1 + b_2 * x_2 \ .... + b_n * x_n$$

Polynomial Regression

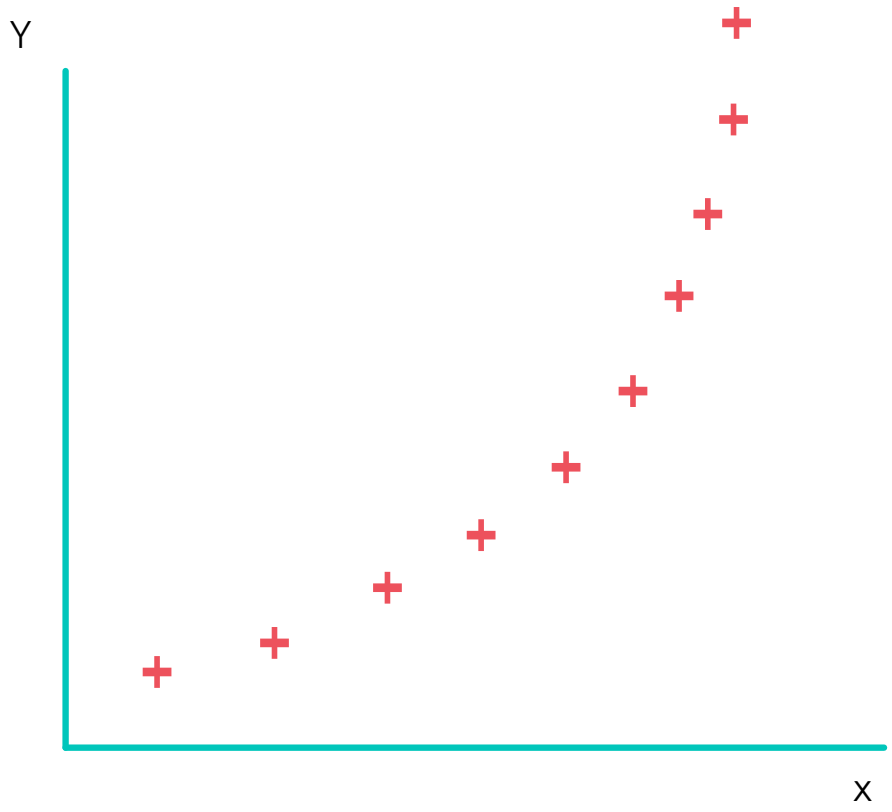
$$y = b_0 + b_1 * x_1 + b_2 * x^2_1 \ .... + b_n * x^2_1$$

Linear Regression Concept

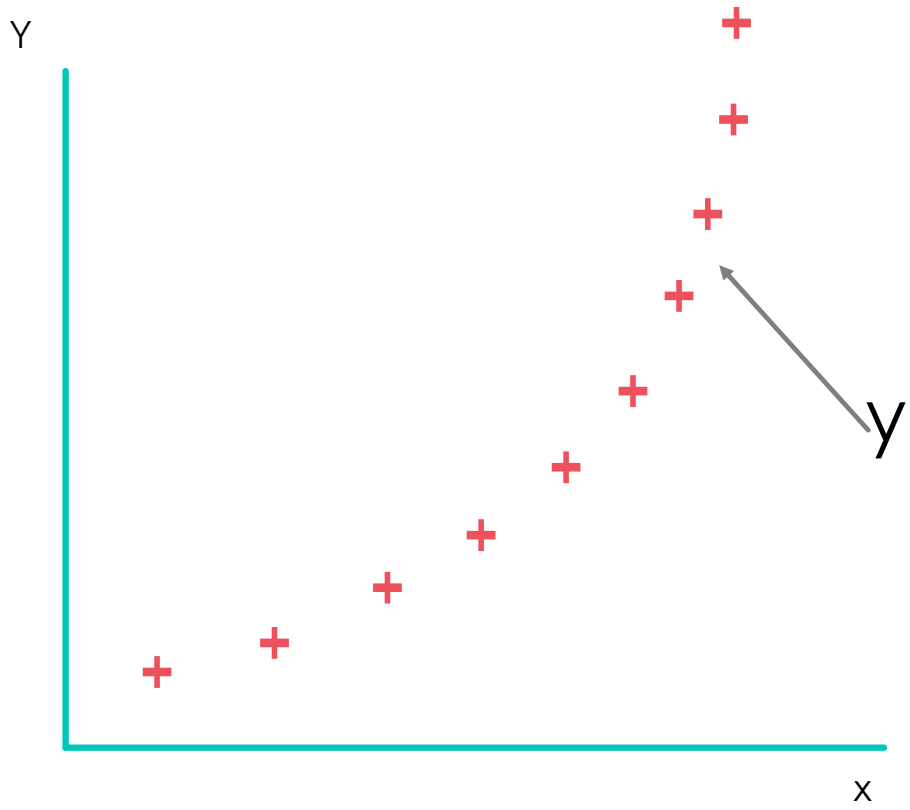


$$y = b_0 + b_1 * x_1$$

Linear Regression Concept



Linear Regression Concept

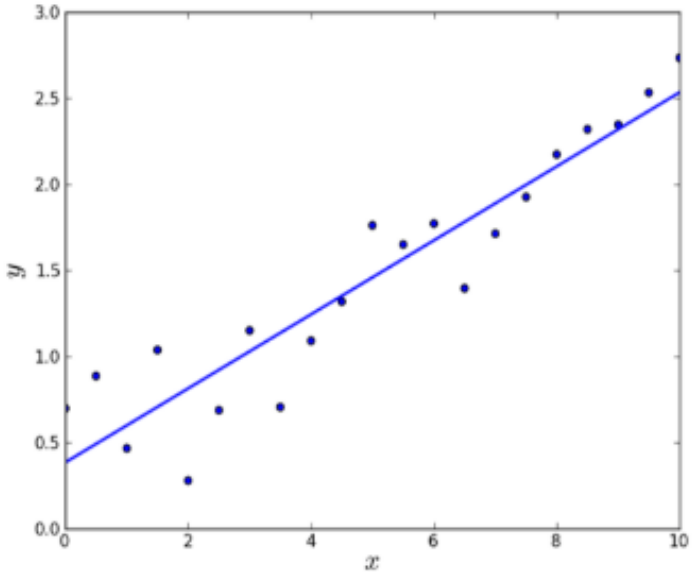


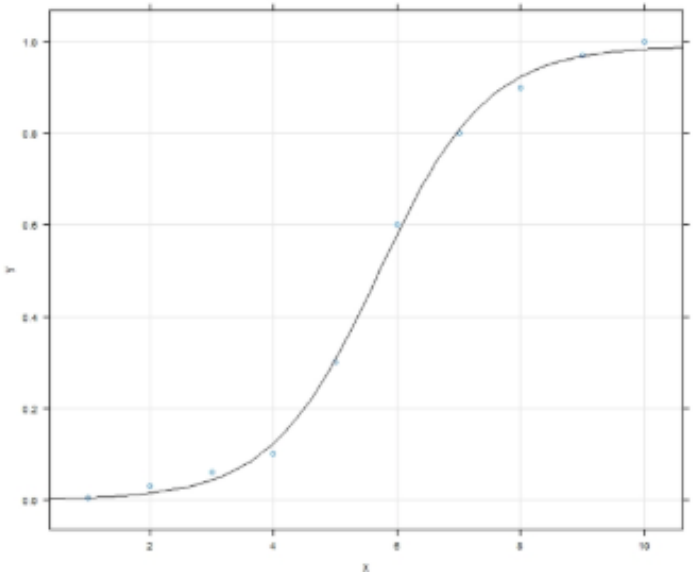
질병 전염병 확산 등

$$y = b_0 + b_1 x_1 * b_2 x_1^2$$

Linear(선형) 이 의미 하는 것은 무엇일까요?

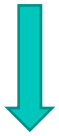
- 선형 회귀 는 “회귀 계수(regression coefficient) 를 선형 결합으로 표현할 수 있는 모델
- 독립변수 X 와 종속변수 Y 의 관계가 일차식 이기 때문에 선형 회기가 아님





Linear(선형) 이 의미 하는 것은 무엇일까요?

- $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$
- $y = \beta_0 x^{\beta_1}$
- $y = \frac{e^{\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3}}{1 + e^{\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3}}$



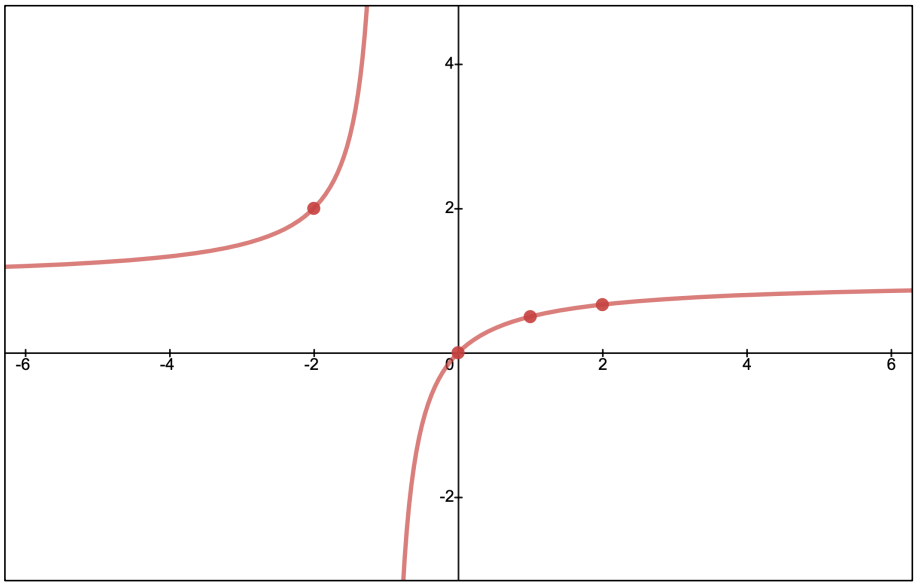
아래와 같이 변형 가능함

- $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 \Rightarrow \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$
- $y = \beta_0 x^{\beta_1} \Rightarrow log(y) = log\left(\beta_0 x^{\beta_1}\right) \Rightarrow log\beta_0 + \beta_1 \log(x) \Rightarrow y^* = \beta_0^* + \beta_1 x^*$
- $y = \frac{e^{\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3}}{1 + e^{\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3}} \Rightarrow \frac{y}{1-y} = e^{\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3} \Rightarrow \log\left(\frac{y}{1-y}\right) = y^* = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$

Linear(선형) 이 의미 하는 것은 무엇일까요?

- 종속 변수와 독립변수 의 관계가 1차 식 이지만 선형 회귀가 아님

$$y = \frac{\beta_1 x}{\beta_2 + x}$$

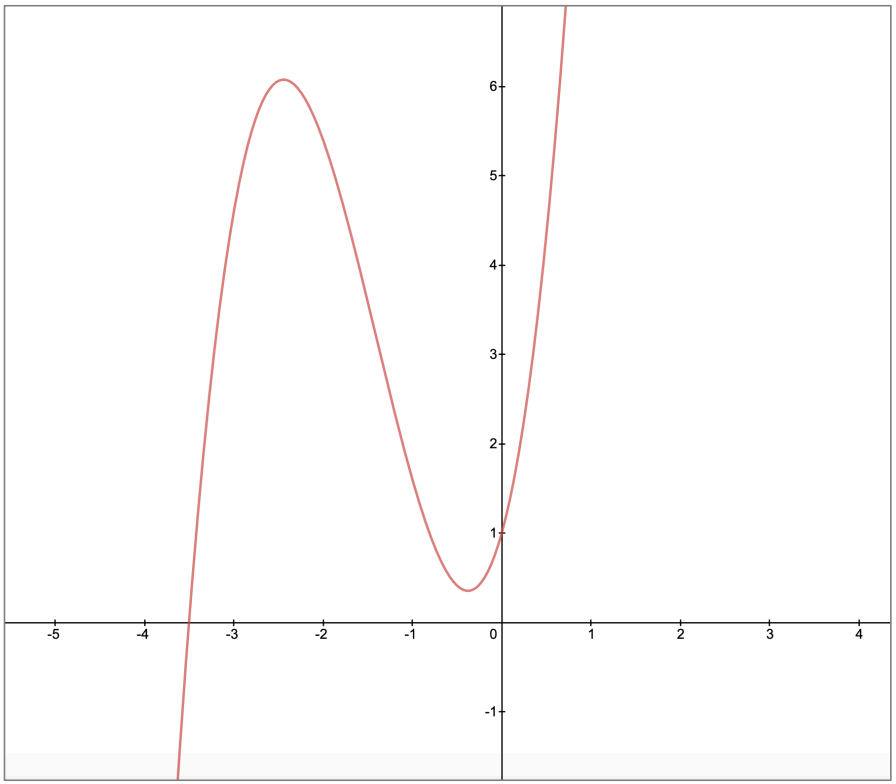


https://www.desmos.com/

Linear(선형) 이 의미 하는 것은 무엇일까요?

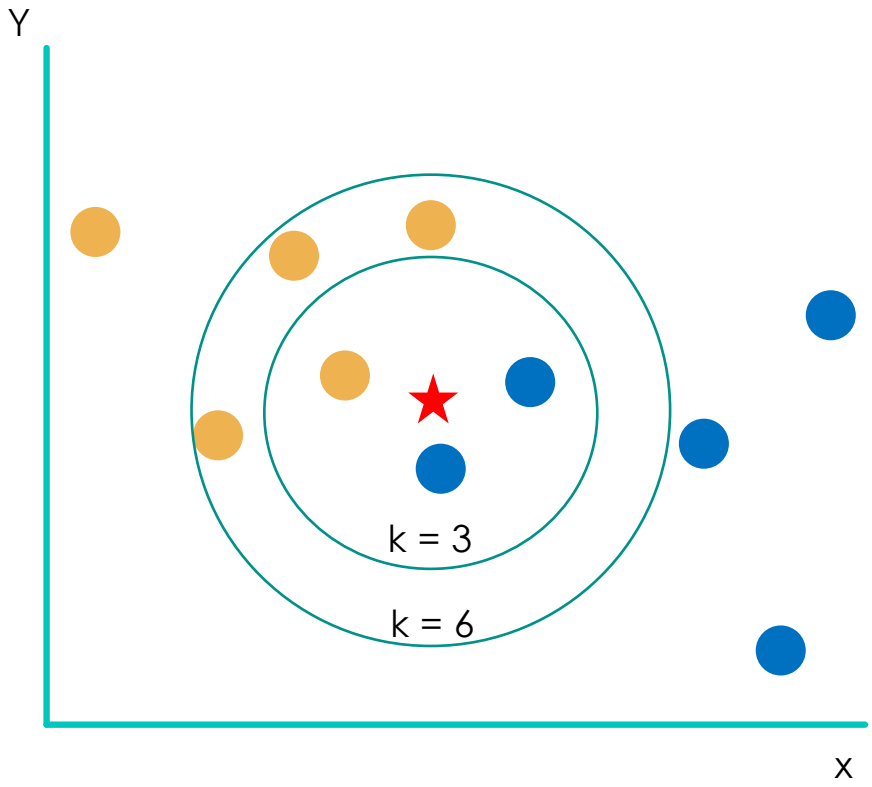
www.demos.com 에서 (a + bx + cx^2 + d^3) 을 입력하고 그래프를 그려보세요

Linear(선형) 이 의미 하는 것은 무엇일까요?



9장. KNN(K Nearest Neighbors)

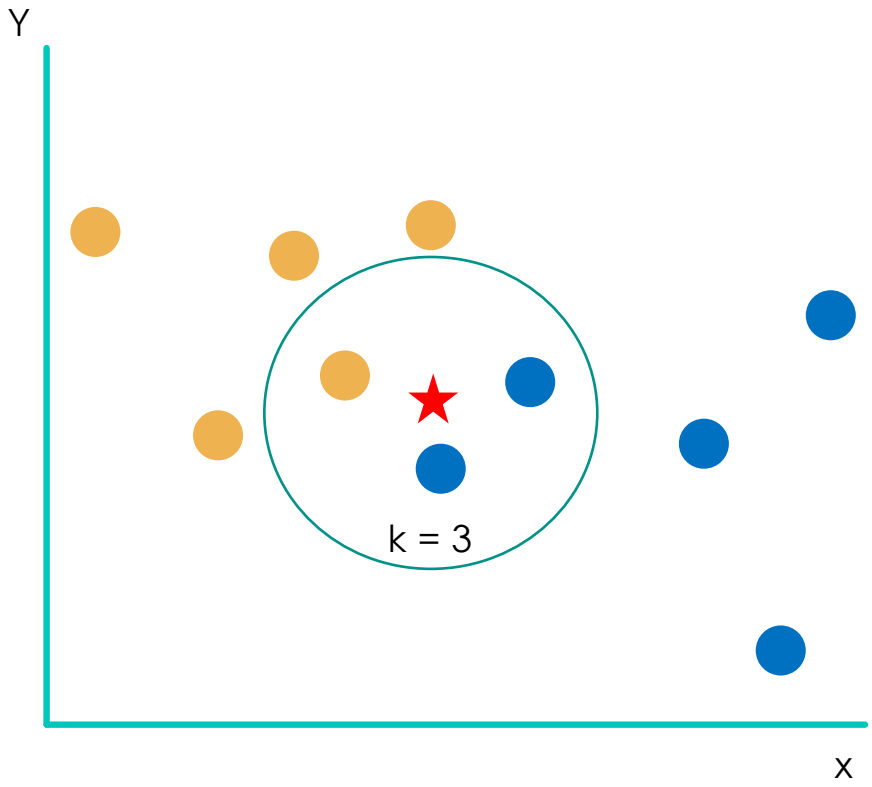
KNN



CLASS A

CLASS B

KNN



CLASS A

CLASS B

KNN

Training Algorithm:

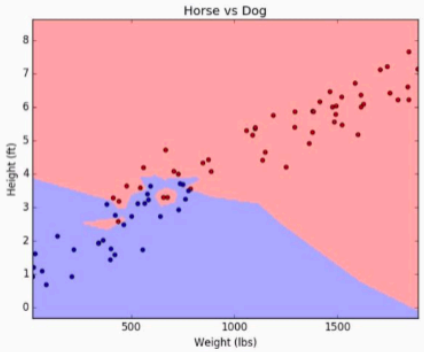
1. 모든 데이터를 저장 합니다. (Training 알고리즘이 없습니다)

Prediction Algorithm:

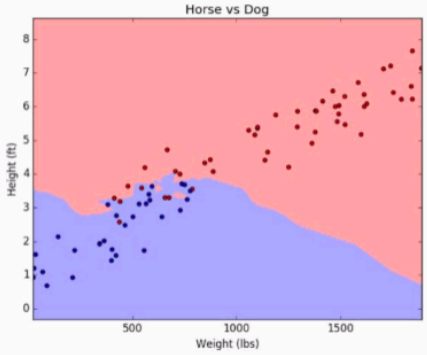
1. 데이터 x 에서 모든 점까지의 거리를 계산합니다.
2. x로 부터 거리를 늘려가며 데이터의 점을 정렬합니다.
3. k 개만큼 가까운 이웃 점들의 라벨(결과값) 중 과반수에 해당하는 값으로 예측 결정 합니다.

KNN

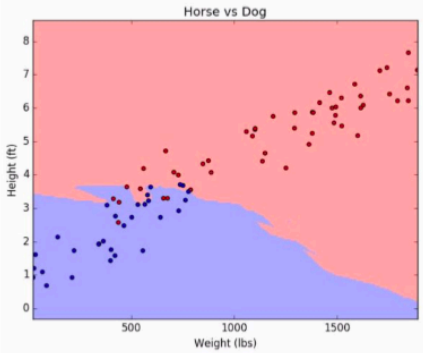
k=1



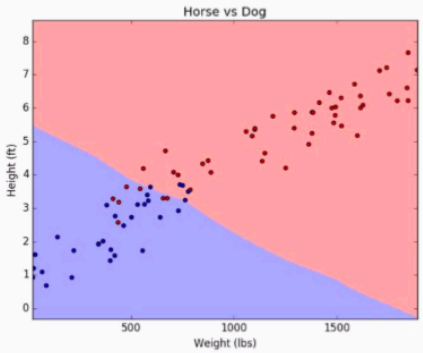
k=5



k=10



k=50



KNN Pros and Cons

Pros

- 알고리즘 구현이 쉽다.
- 데이터 추가에 용이
- 수치 형 데이터에 잘 맞음
- 튜닝 파라메터가 적습니다.
  - K
  - Distance Metric : Distance 를 정의 하는 방법 (예 : Euclidean distance, Mahalanobis Distance Metric)

Euclidean distance

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots}$$

Mahalanobis Distance Metric

$$d(p,q) = \sqrt{(\vec{p} - \vec{q})^\top \Omega (\vec{p} - \vec{q})}$$

Cons

- 데이터 셋이 클 경우 고비용 (전체 데이터를 스캔)
- 차원이 많은 데이터 셋에 부적합 합니다. (역시 계산량이 많아 집니다)
- 범주형 변수와는 잘 맞지 않습니다.(모델 성능이 나오지 않음

KNN Pros and Cons

- Euclidean distance

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots}$$

ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	*sqrt((6 - 8)^2 + (4 - 5)^2) = 2.2*
green bean	3	7	vegetable	*sqrt((6 - 3)^2 + (4 - 7)^2) = 4.2*
nuts	3	6	protein	*sqrt((6 - 3)^2 + (4 - 6)^2) = 3.6*
orange	7	3	fruit	*sqrt((6 - 7)^2 + (4 - 3)^2) = 1.4*

토마토의 당도(sweetness) 가 6이고 , 바삭함(crunchiness)이 4일때, 녹두의 당도가 3이고 바삭함이 7 이라면 당도는 당도 끼리, 바삭함은 바삭함 끼리 빼주면 두 개체 간의 유클리디안 거리 값이 나옵니다.

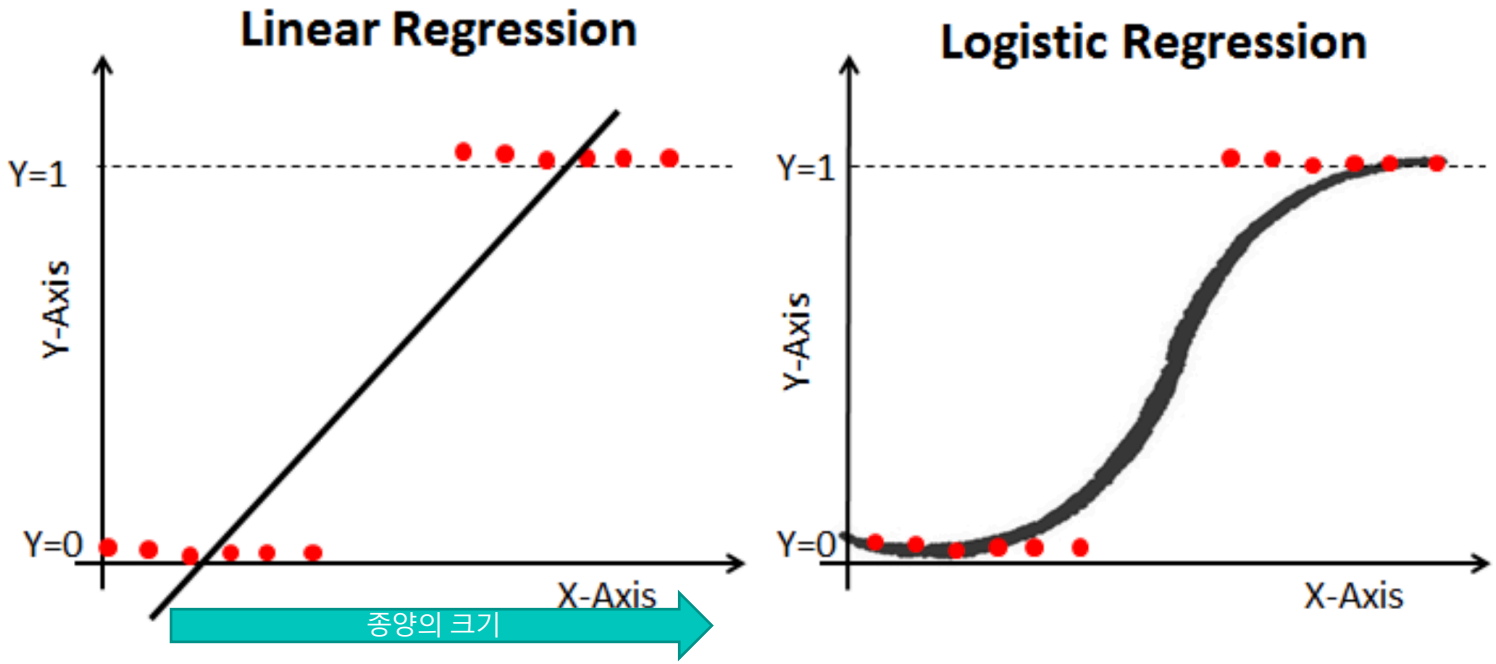
$$\text{dist}(tomato, green\ bean) = \sqrt{(6-3)^2 + (4-7)^2} = 4.2$$

유클리디안 거리가 작을수록 해당 데이터는 유사성을 가진다고 할 수 있습니다.

9장. Logistic Regression (binary)

Logistic Regression

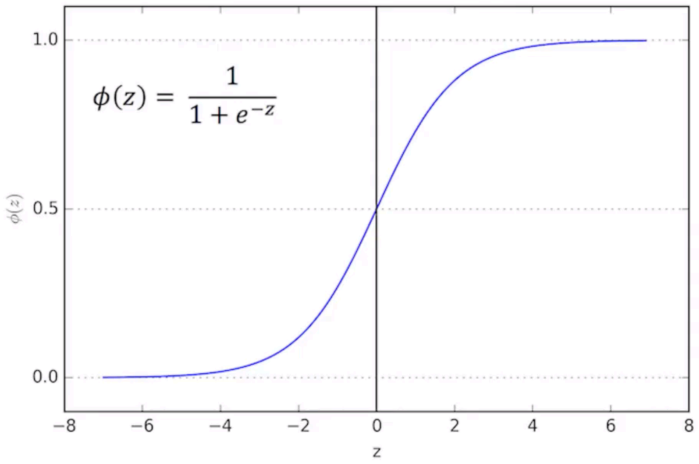
- 왼쪽 그림의 경우 Y 가 0 또는 1 (예 : 사망/생존, 합격/불합격 등) 이라면 선형회귀로는 Fitting 하기가 어렵습니다.
- 곡선으로 fitting 하기 위해 사용 하는 것이 logistic 함수 또는 로짓 변환 이라고 합니다.

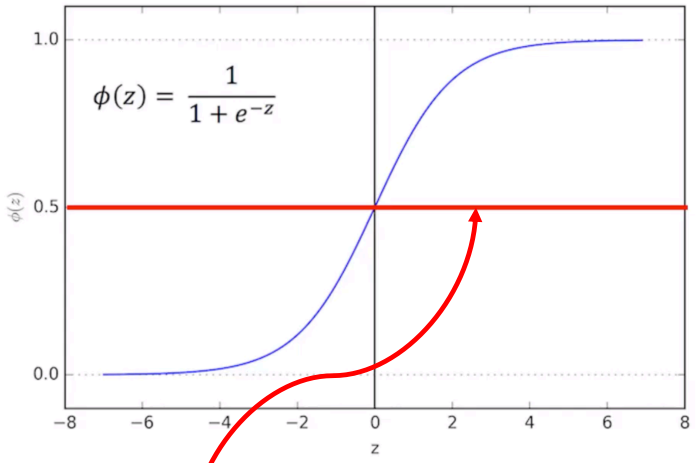


종양의 크기에 따른 양성/악성 종양을 분류 한다고 해봅시다

KNN Pros and Cons

- 임계점 을 정하고 Logistic 회귀에서 나온 결과를바탕으로 2진 분류를 합니다.
- 0.5 이상이면 “참”, ”0.5” 이하면 거짓

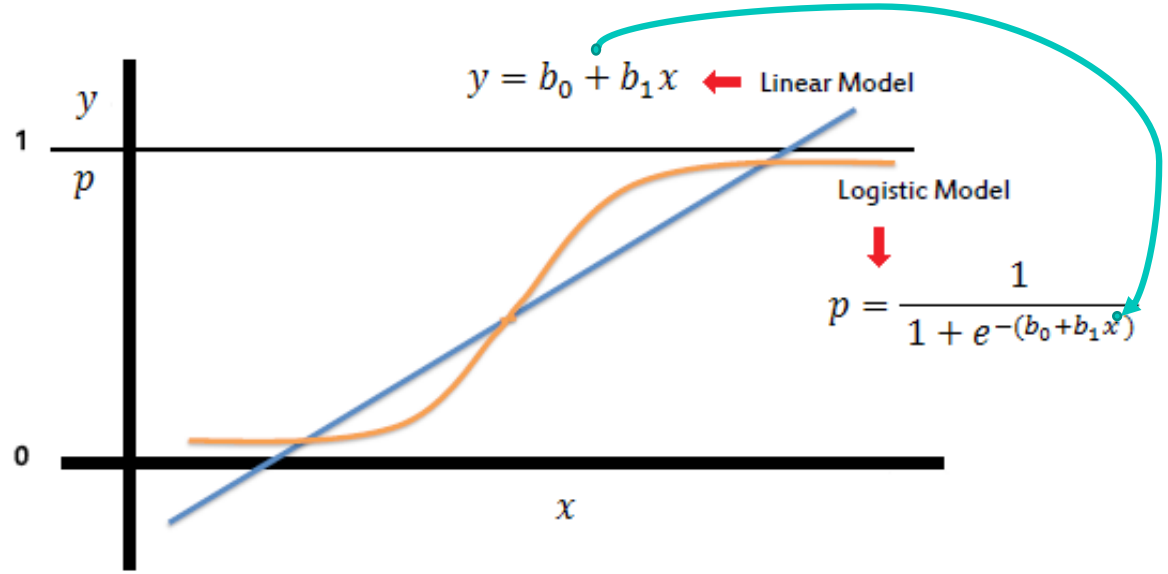




Cut off point : 0.5 이상이면 “참” 0.5 이하면 “거짓” 으로 분류

Logistic Regression

- 선형회귀 함수가 Logistic 함수의 인자로 들어가게 되면 선형이 휘게 됩니다.

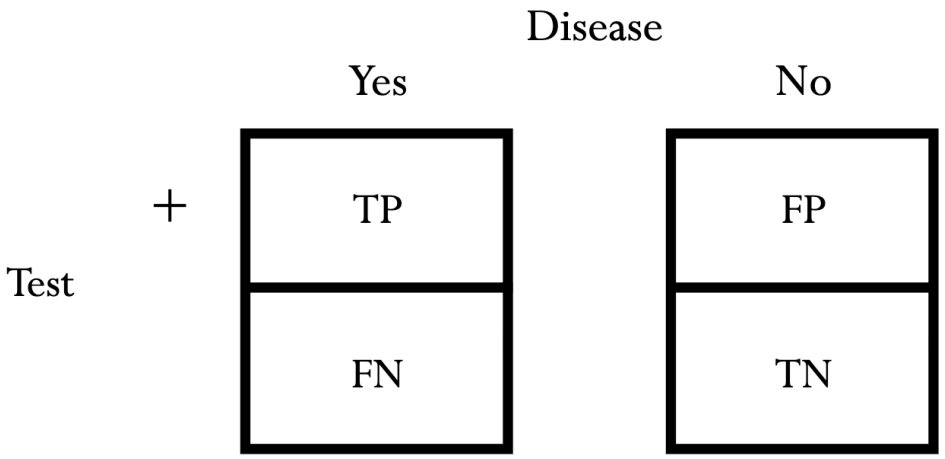


y 값이 1과 0 사이의 확률 로 나옵니다

sigmoid 함수 라고도 합니다.

ROC Curve - 민감도(Sensitive) , 특이도(Specifityfy)

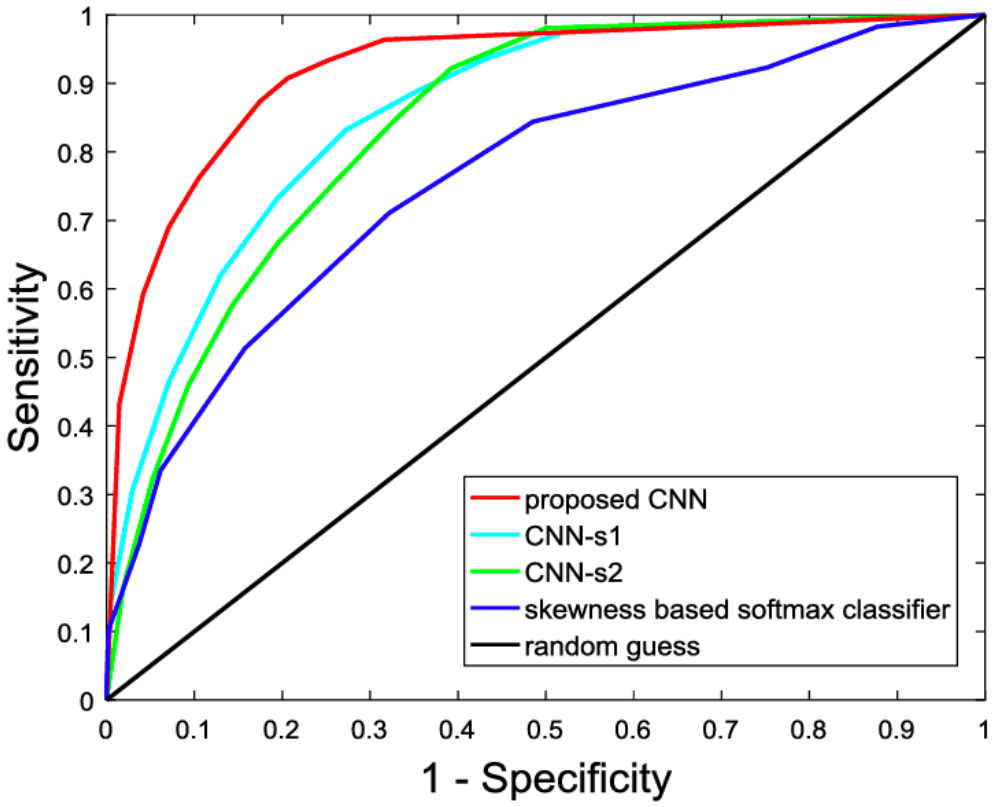
ROC Curve 는 민감도와 특이도 를 구해서 그래프로 표현한 것 입니다.
예측 결과에 대해 민감도(Sensitivity) 와 특이도(Specificity) 의 비율을 그래프로 그려 보겠습니다.



$$Se = \frac{TP}{TP + FN} \qquad Sp = \frac{TN}{TN + FP}$$

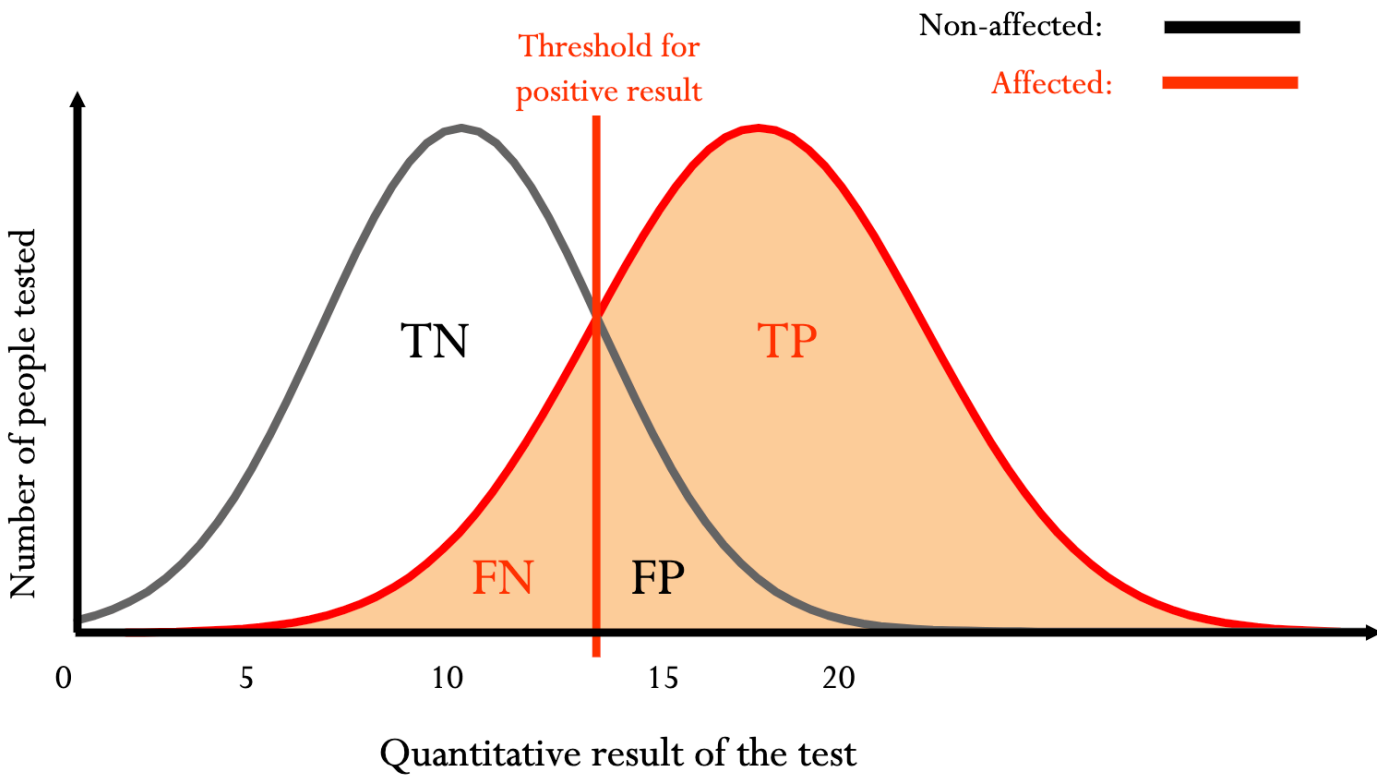
ROC Curve - 해석

ROC Curve 의 아래의 면적을 AUC 라고 합니다. AUC 가 넓을수록 좋은 모델입니다.



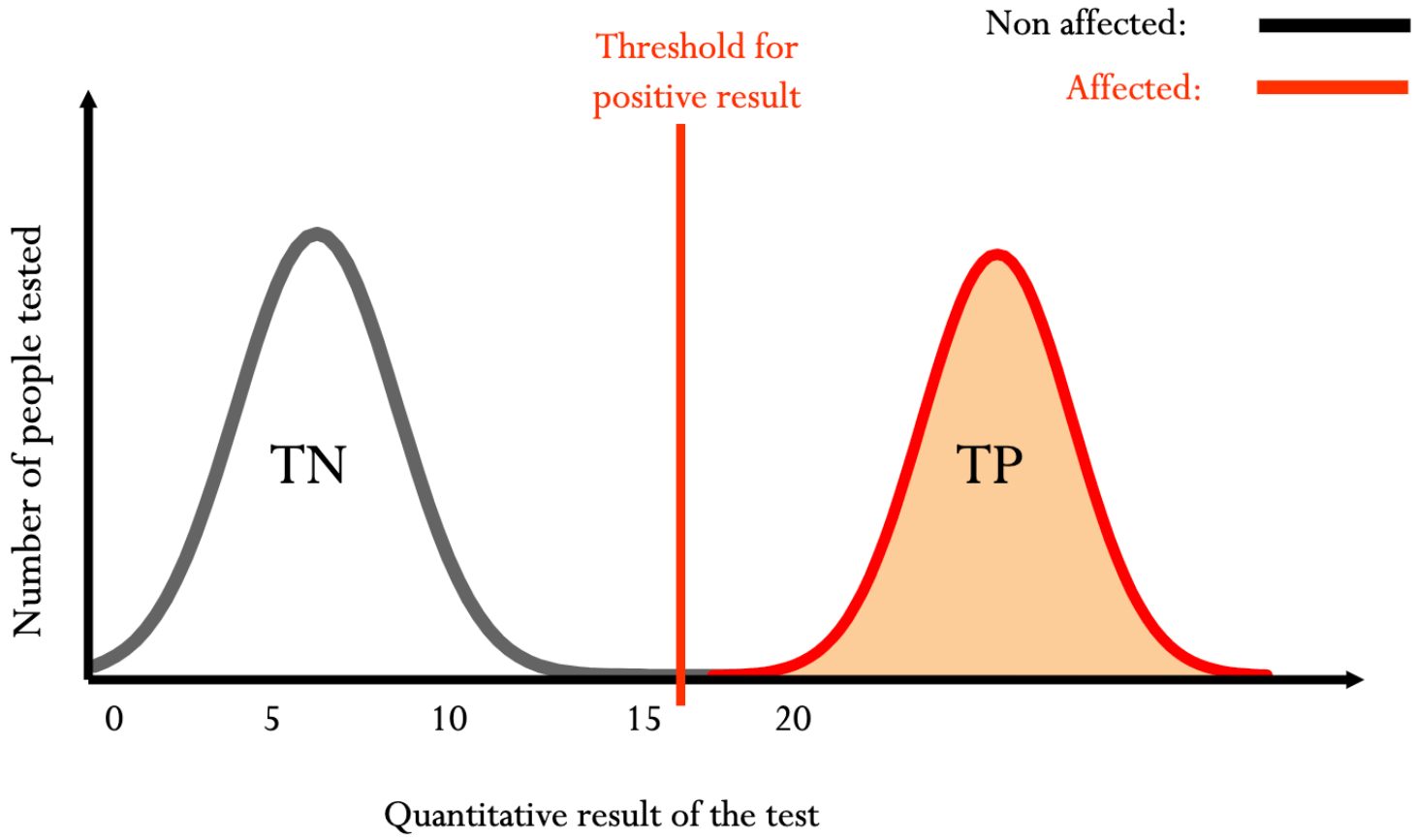
ROC Curve 의 AUC 가 적은 모델

- ROC Curve 의 AUC 가 적은 모델



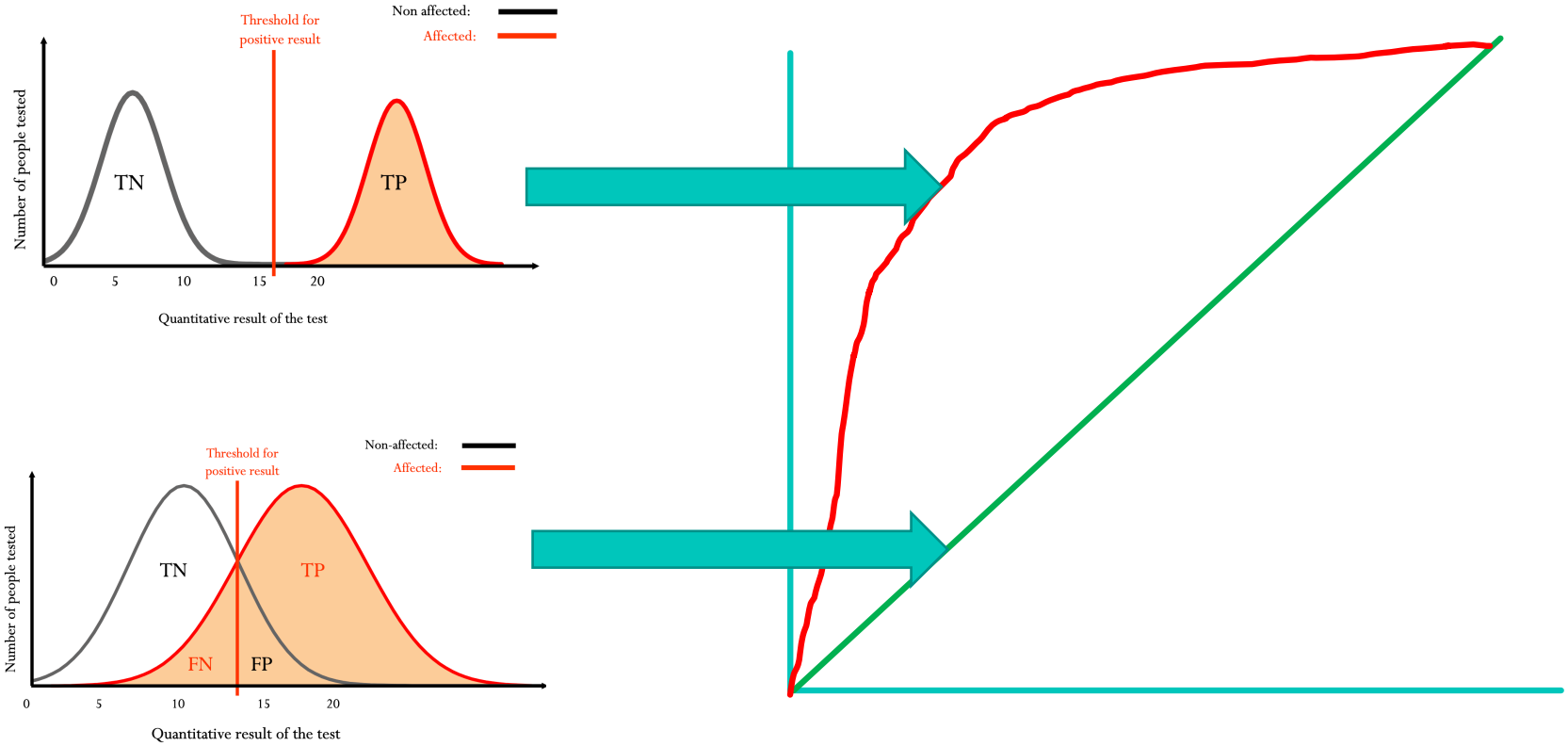
ROC Curve 의 AUC 가 높은 모델 (ideal)

- ROC Curve 의 AUC 가 높은 모델



ROC Curve 의 AUC 가 높은 모델 (ideal)

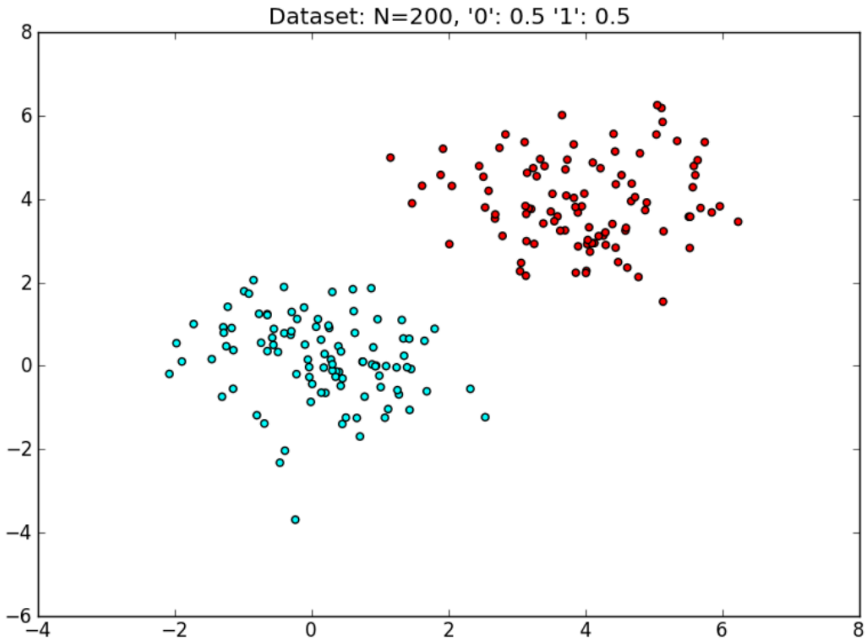
- ROC Curve 의 AUC 가 높은 모델



9장. Support Vector Machine

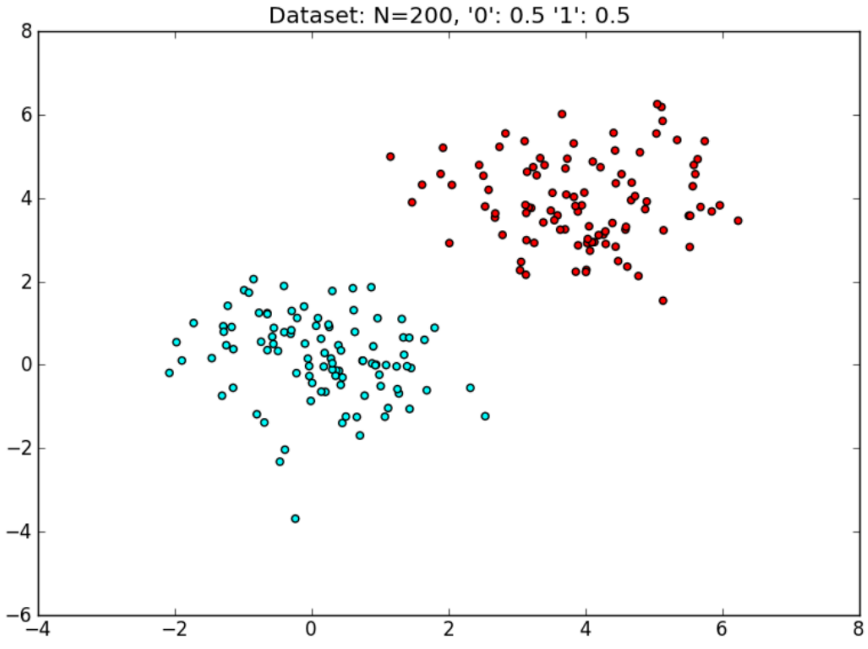
Support Vector Machine Concept

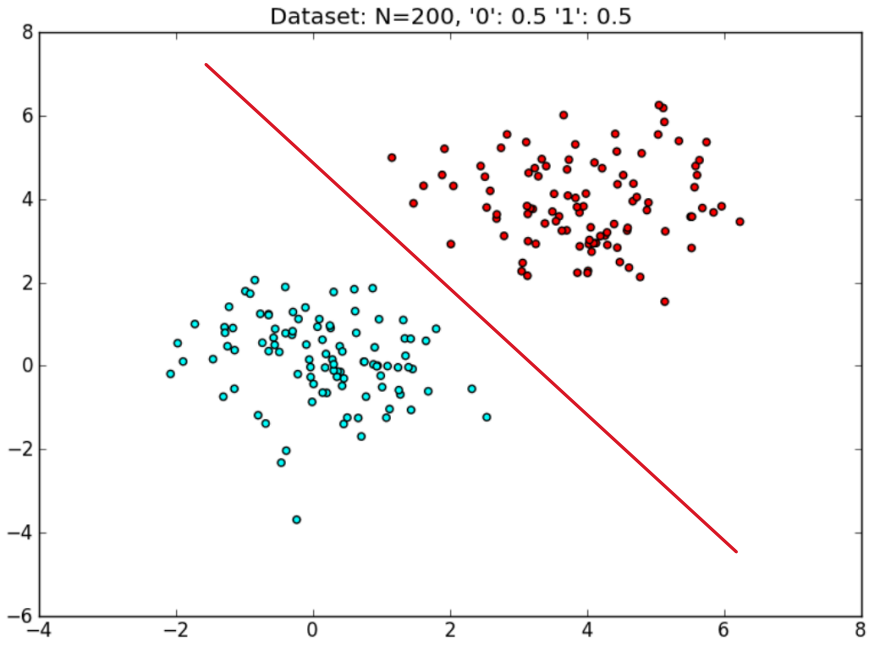
- 결과 값이 있는(레이블 된) 데이터 셋을 아래 와 같이 그려 봅시다.



KNN

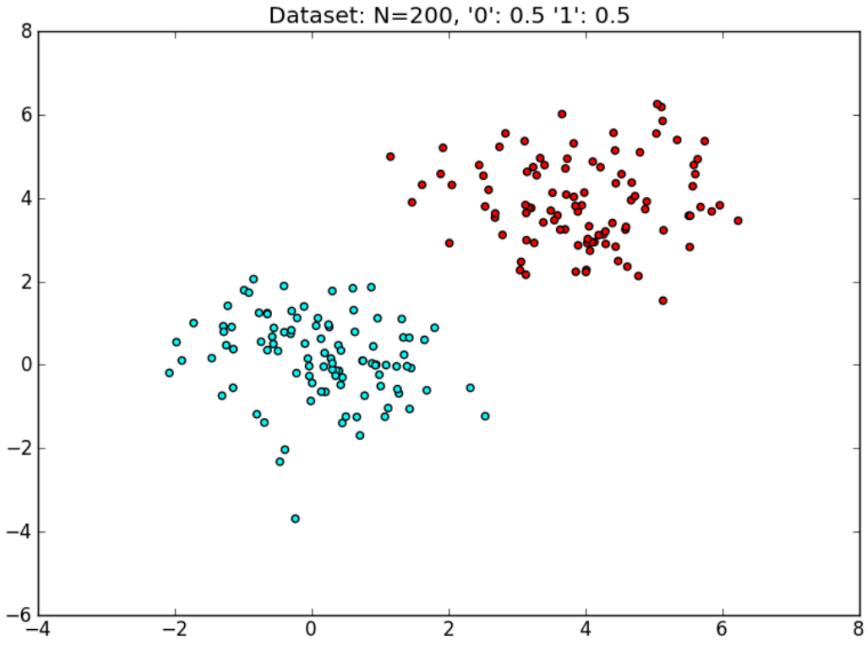
- 두개의 군집 사이에 선을 그려 구분하여 분리 할 수 있습니다.

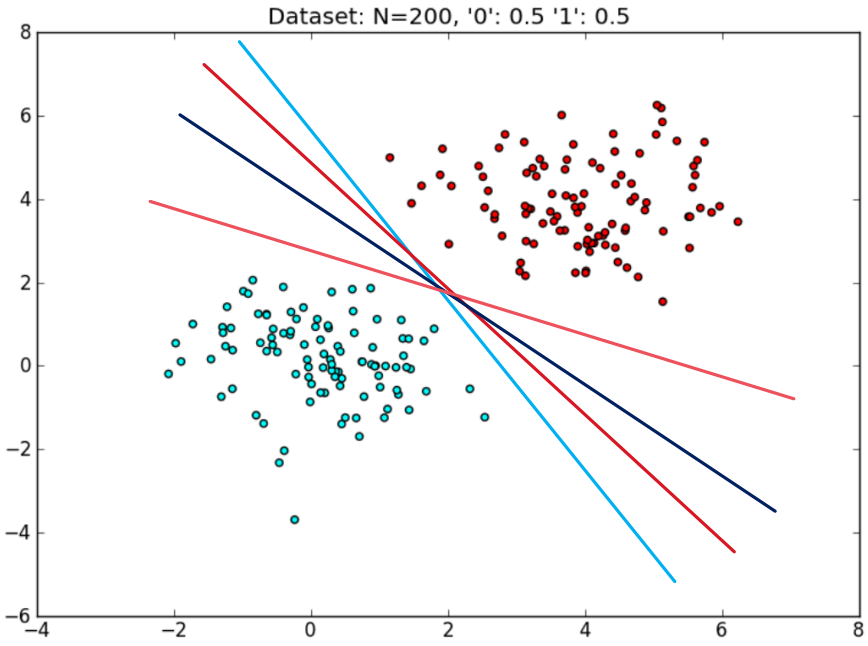




Support Vector Machine Concept

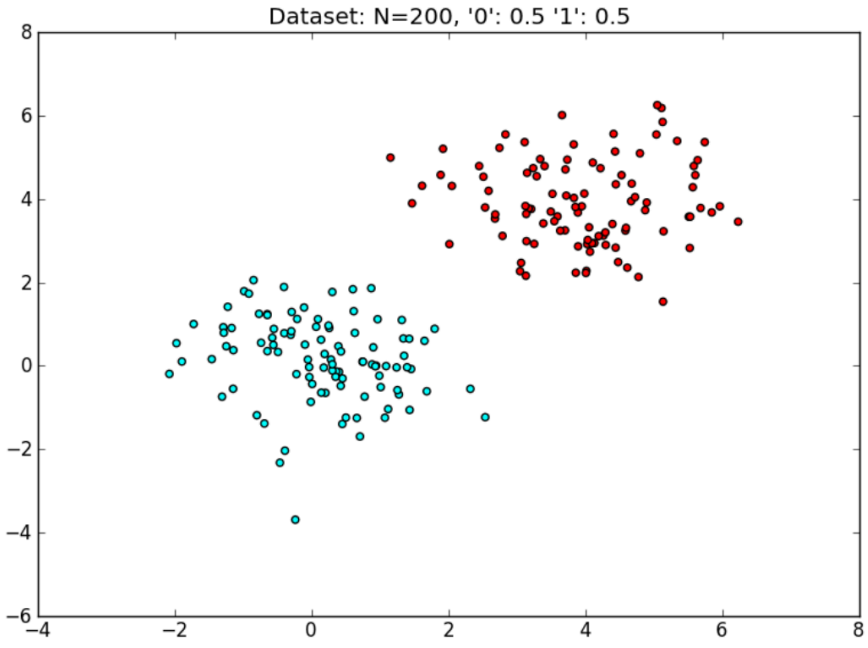
- 두개의 군집을 구분하는 수많은 선을 그릴 수 있습니다.

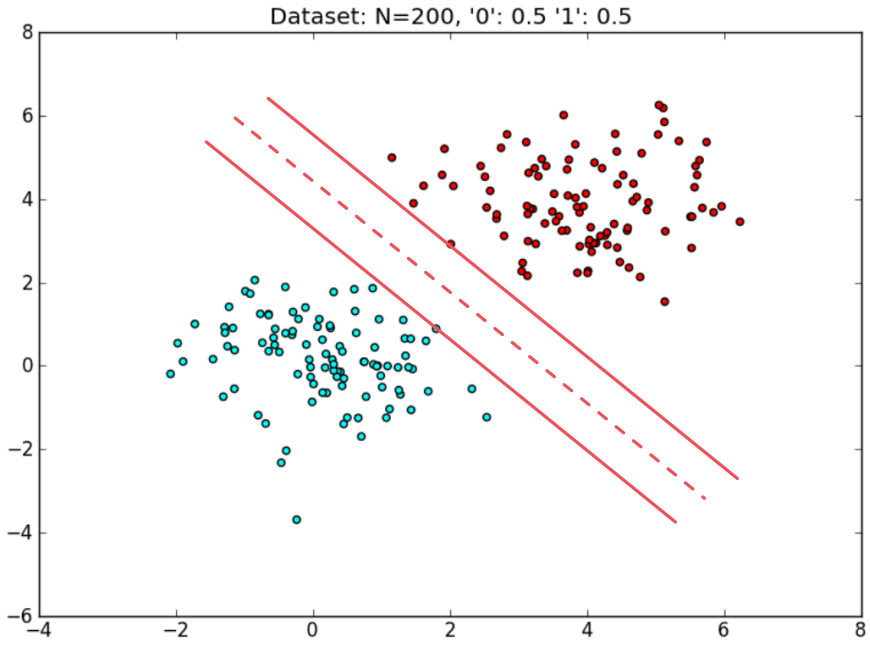




Support Vector Machine Concept

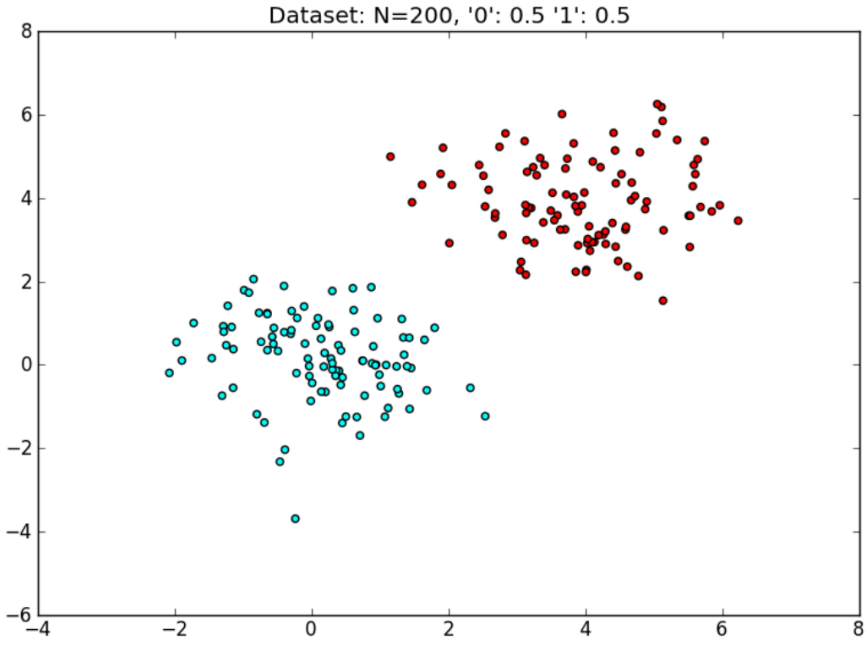
- 두개의 군집간의 간격을 최대한 크게 하는Hyperplane(초 평면) 고르는 것이 가장 정확 합니다.

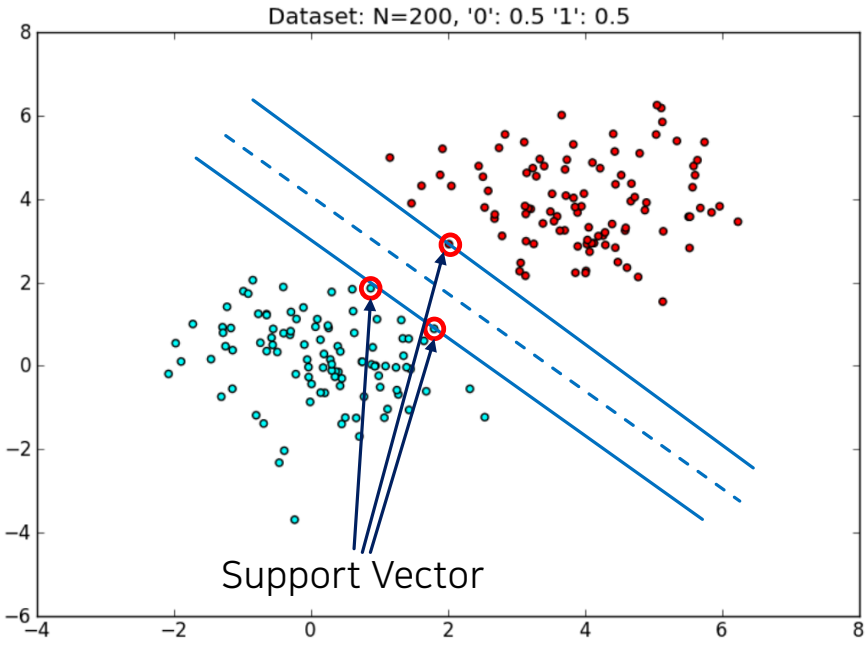




Support Vector Machine Concept

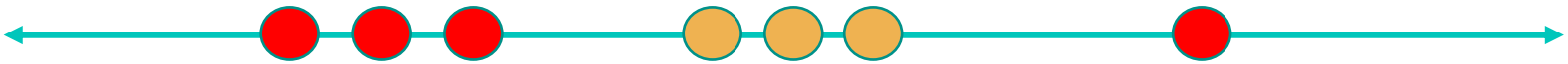
- 두개의 군집간의 간격을 최대한 크게 하는Hyperplane(초 평면) 고르는 것이 가장 정확 합니다.





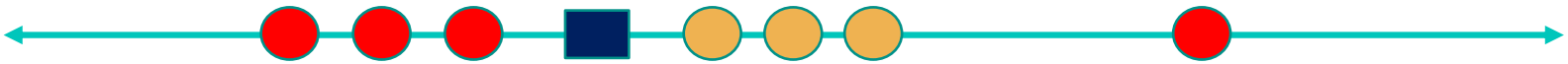
Support Vector Machine Concept – Kernel Trick (one dimension)

- 1차원의 공간에서 하나의 점을 찍어 여러 개의 데이터를 구분 해봅시다



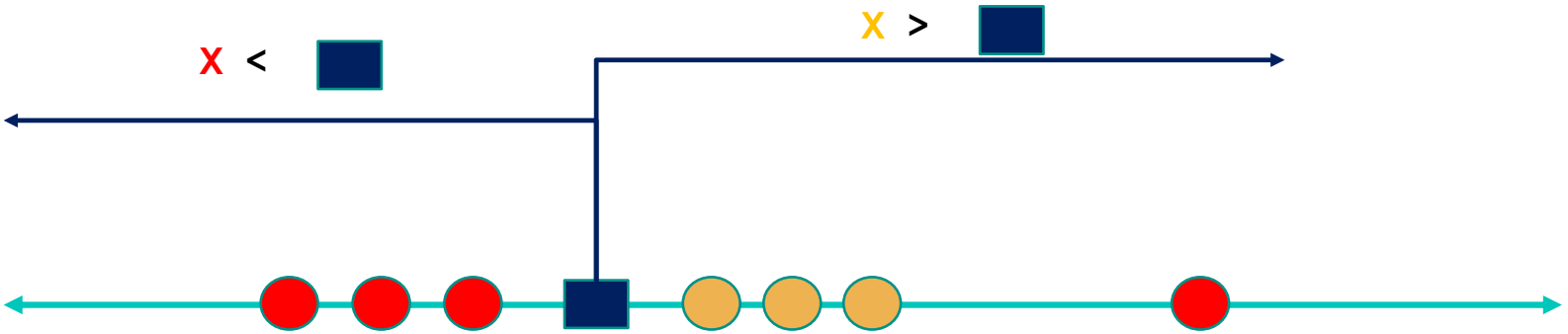
Support Vector Machine Concept – Kernel Trick (one dimension)

- 1차원의 공간에서 하나의 점을 찍어 여러 개의 데이터를 구분 해봅시다



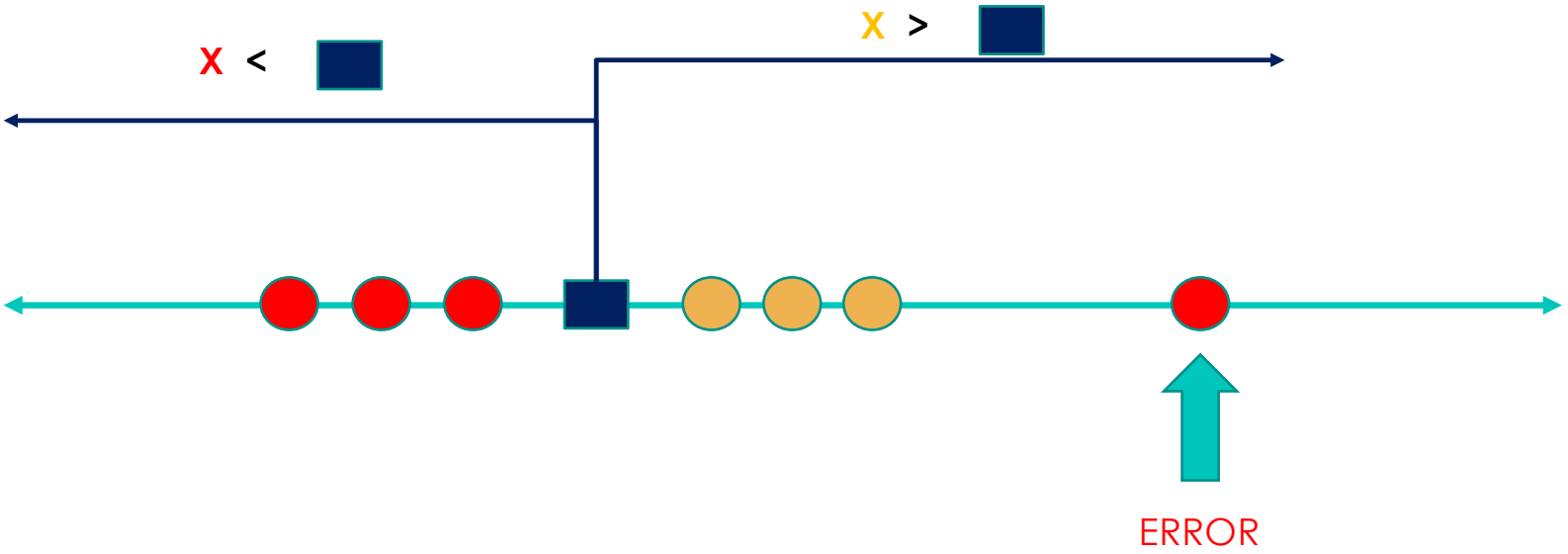
Support Vector Machine Concept – Kernel Trick (one dimension)

- 1차원의 공간에서 하나의 점을 찍어 여러 개의 데이터를 구분 해봅시다



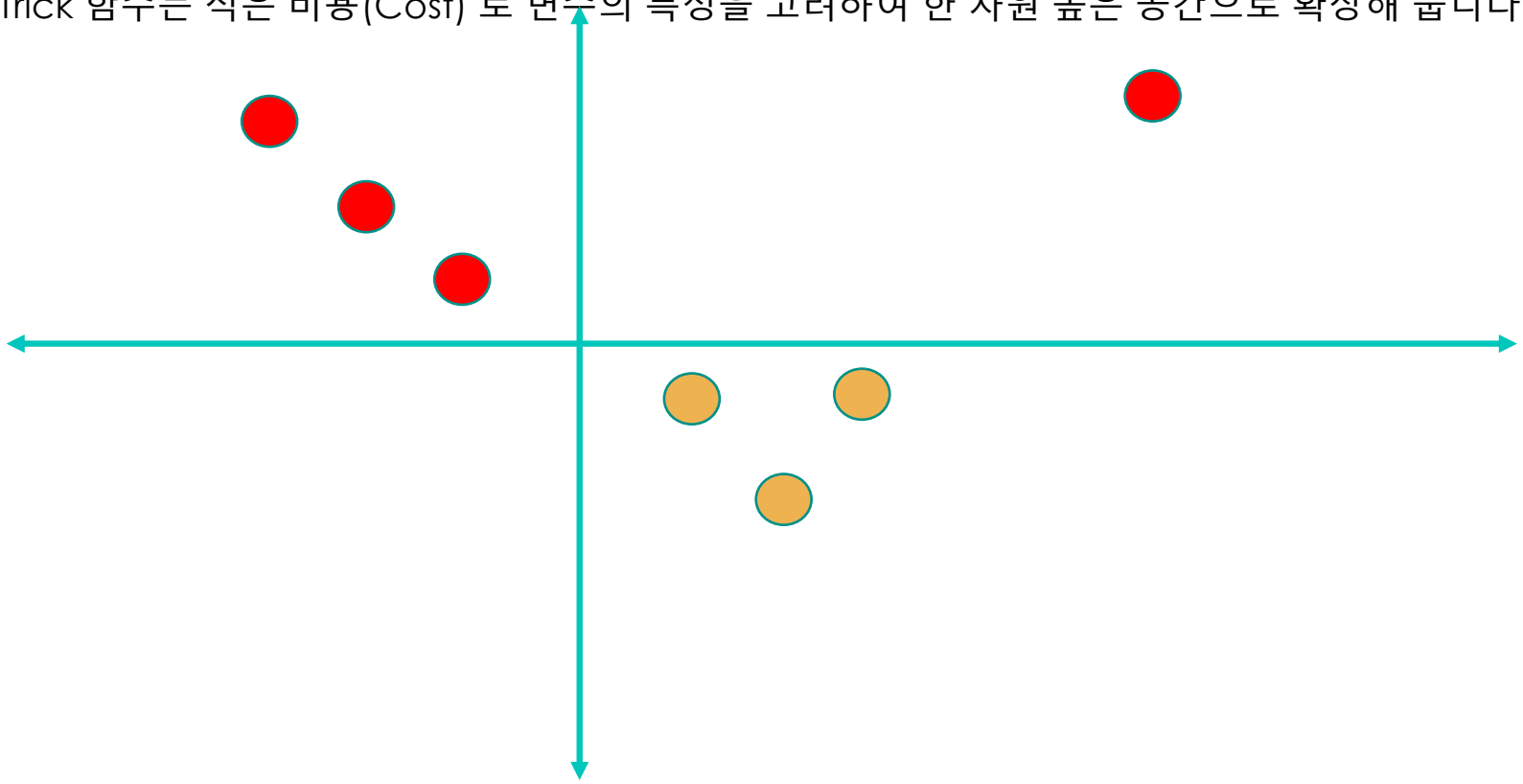
Support Vector Machine Concept – Kernel Trick (one dimension)

- 1차원의 공간에서 하나의 점을 찍어 여러 개의 데이터를 구분 해봅시다



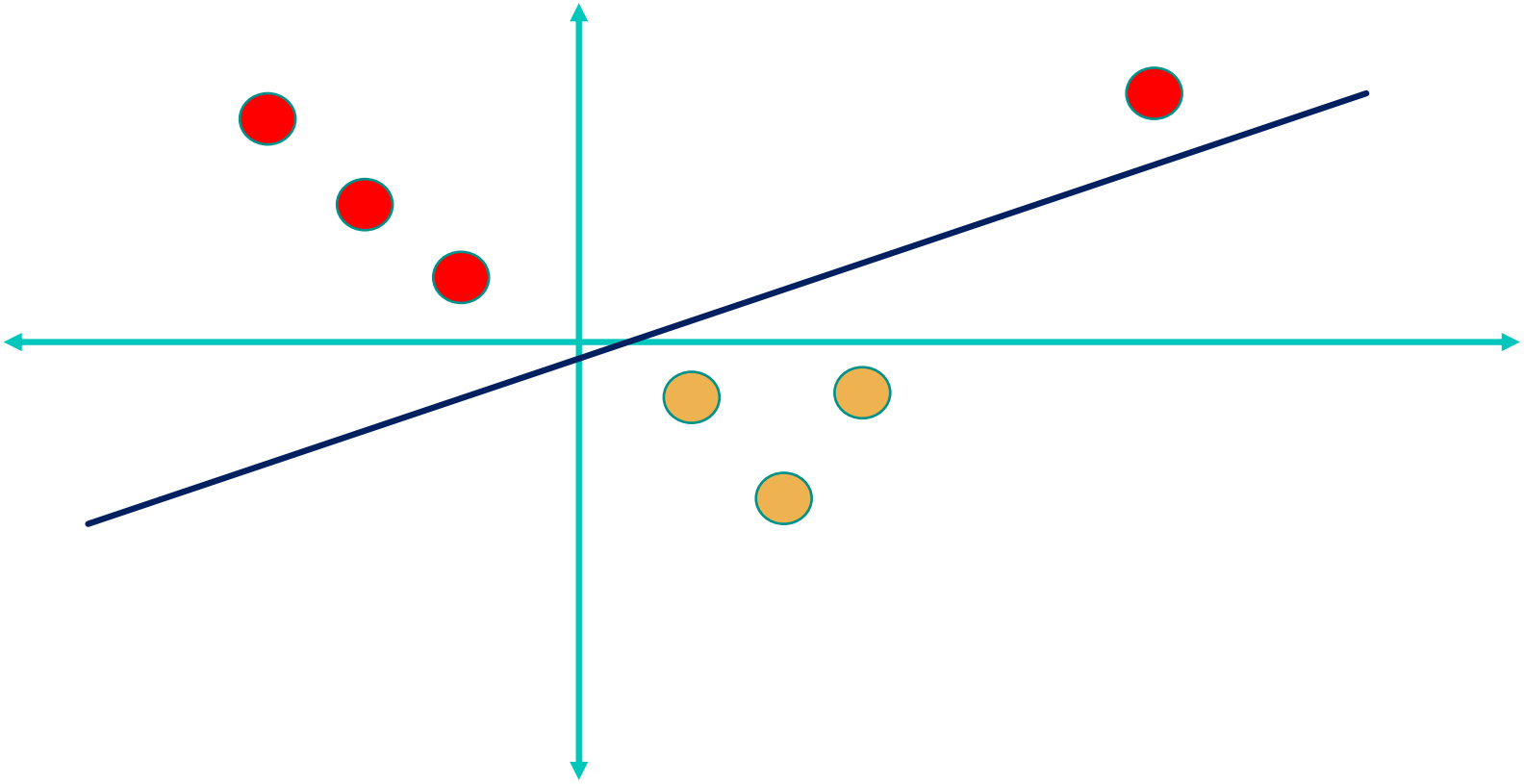
Support Vector Machine Concept – Kernel Trick (one dimension)

- 1차원 점들을 Kernel Trick 함수를 통해 2차원 으로 확장 할 수 있습니다.
- Kernel Trick 함수는 적은 비용(Cost) 로 변수의 특성을 고려하여 한 차원 높은 공간으로 확장해 줍니다.



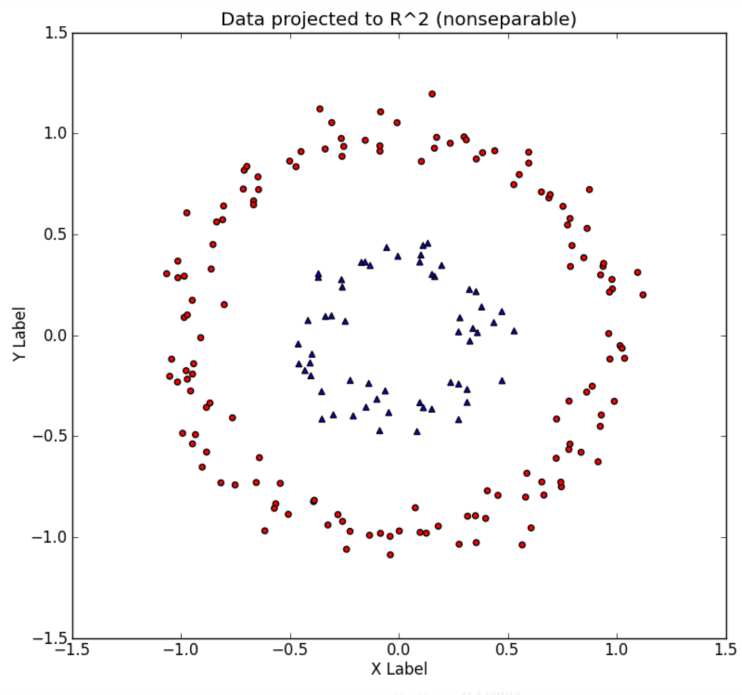
Support Vector Machine Concept – Kernel Trick (one dimension)

- 차원을 확장하니 1차원에서 구분 할 수 없었던 점들의 특성을 손쉽게 구분 하게 됩니다.



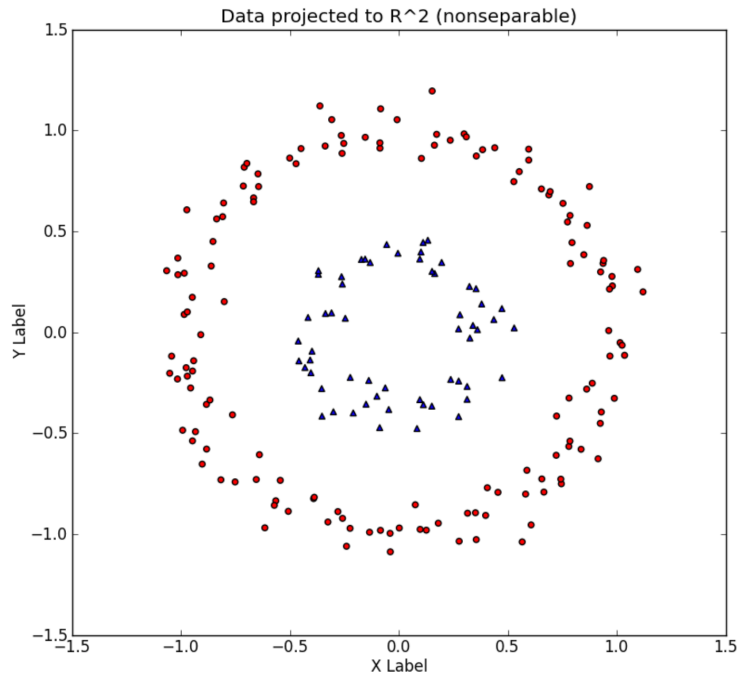
Support Vector Machine Concept – Kernel Trick (high Dimension)

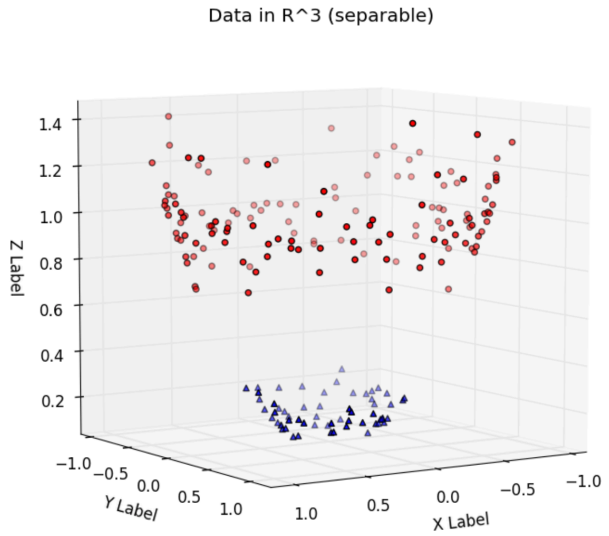
- n-1 차원의 hyperplane(여기서는 1차원인 선) 을 이용해서 데이터를 구분 할 수 가 없습니다.



Support Vector Machine Concept – Kernel Trick (high Dimension)

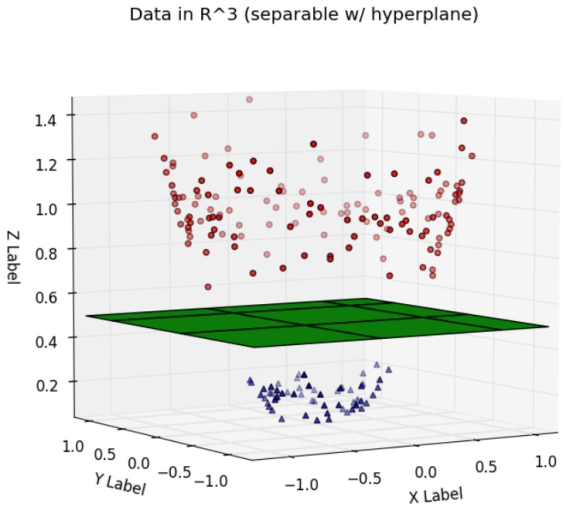
- 3차원으로 확장해 데이터를 그려 보면 두개의 집합에 대해 구분이 가능 합니다.

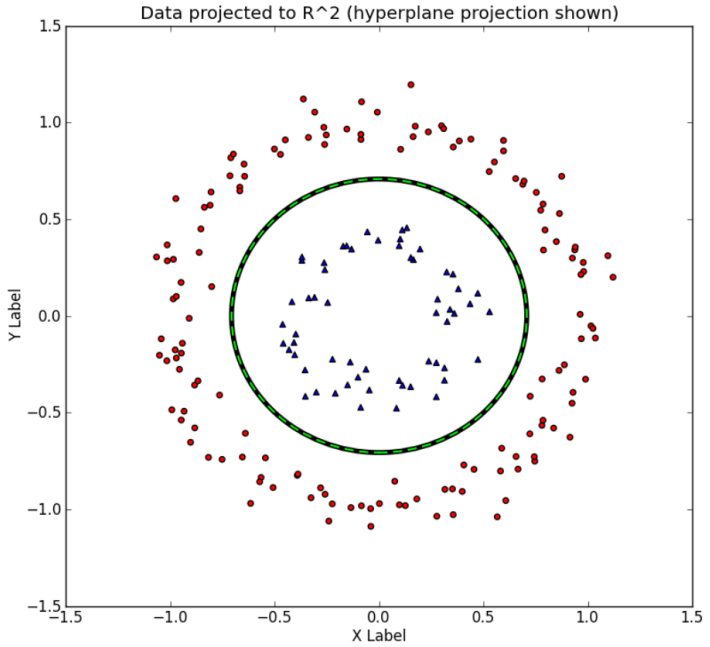




Support Vector Machine Concept – Kernel Trick (high Dimension)

- 3차원으로 에서 n-1 차원인 면을 통해 두개의 군집을 구분 할 수 있습니다.
- 구분된 군집을 2차원에서 보면 아래와 같습니다.
- 이렇게 n 차원의 데이터를 n+1 차월으로 투영해 주는 것을 Kernel Trick 이라고 부릅니다.

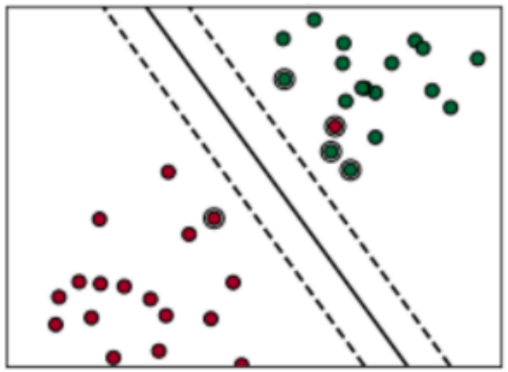




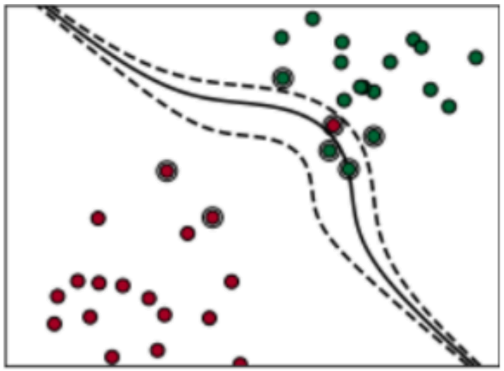
Support Vector Machine Concept – Type of Kernel Trick

- 커널 트릭 함수에는 크게 3가지 종류가 있습니다. : Linear / Polynomial / RBF(radial basis function)

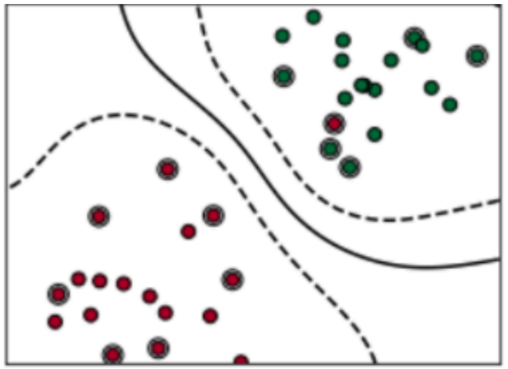
Decision Boundary : linear



Decision Boundary : Poly



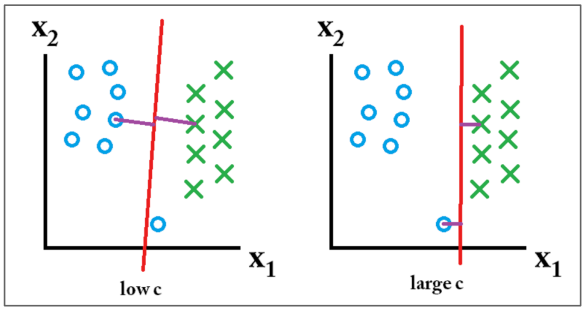
Decision Boundary : RBF



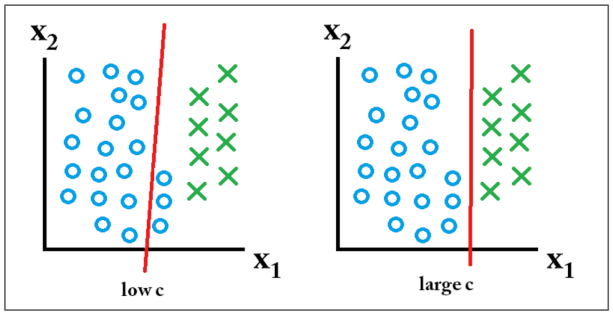
Support Vector Machine Concept - – C 값 Gamma 값

- SVM 에서 좀더 많은 그룹 객체를 구분 하기 위해 Margin 의 범위를 C 파라메터에 선언된 만큼 더 작게 가져 하는 것을 허용 하는 것입니다.
- 아래는 C 값을 작게 했을때 좋은 사례와 나쁜 사례의 예시 입니다.

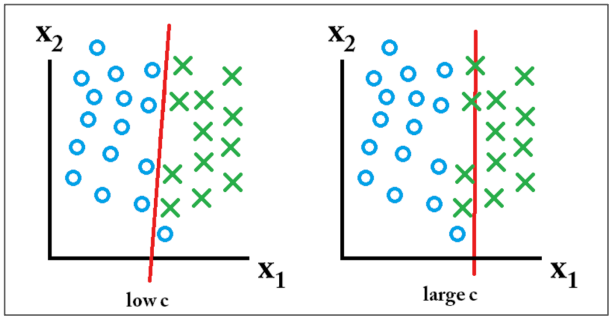
GOOD



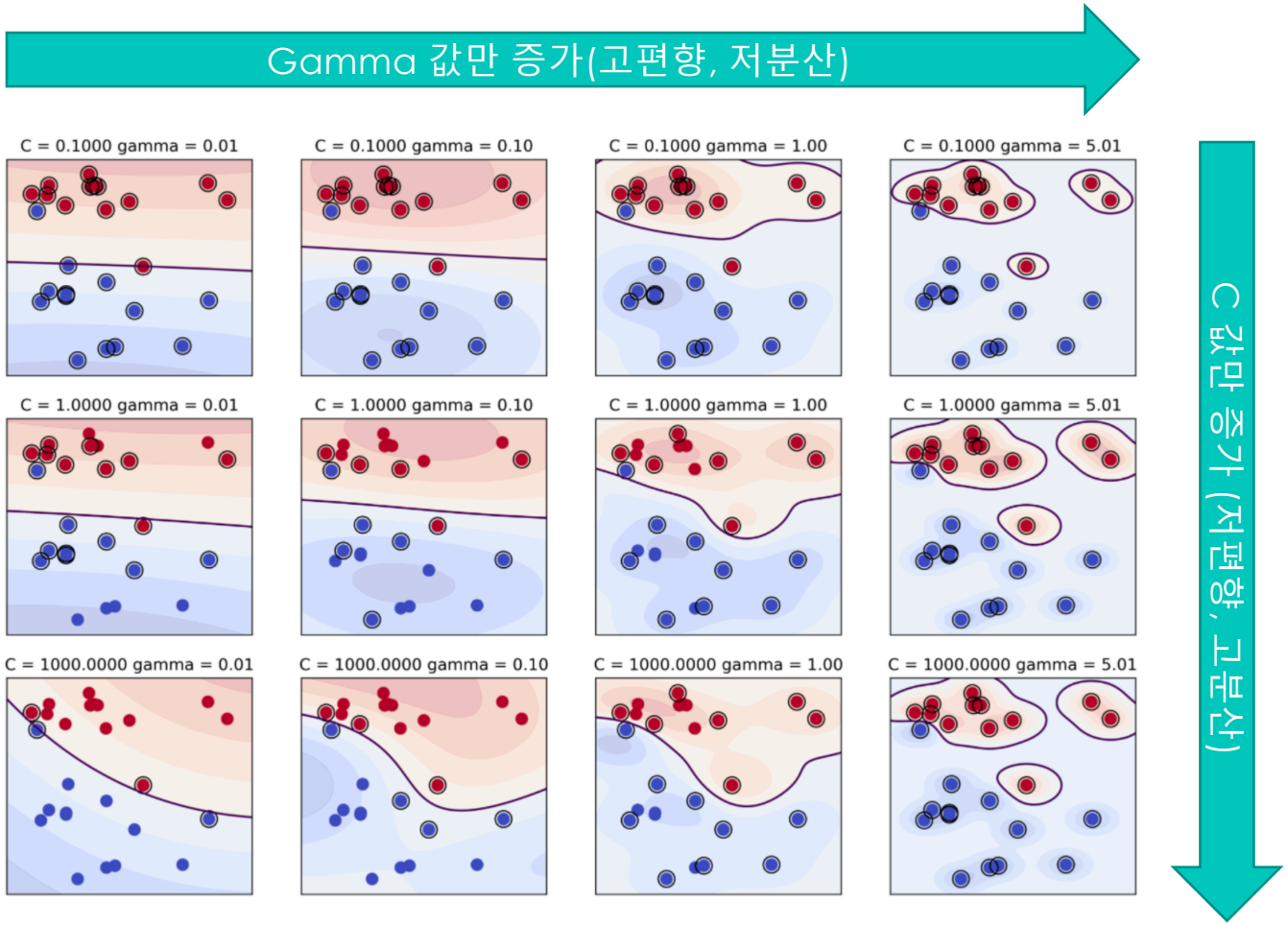
GOOD



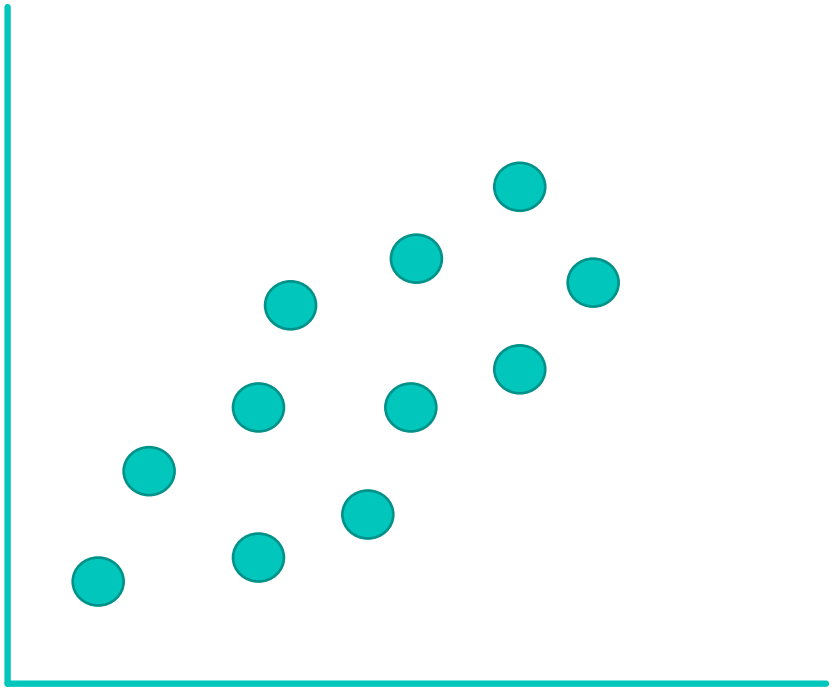
BAD



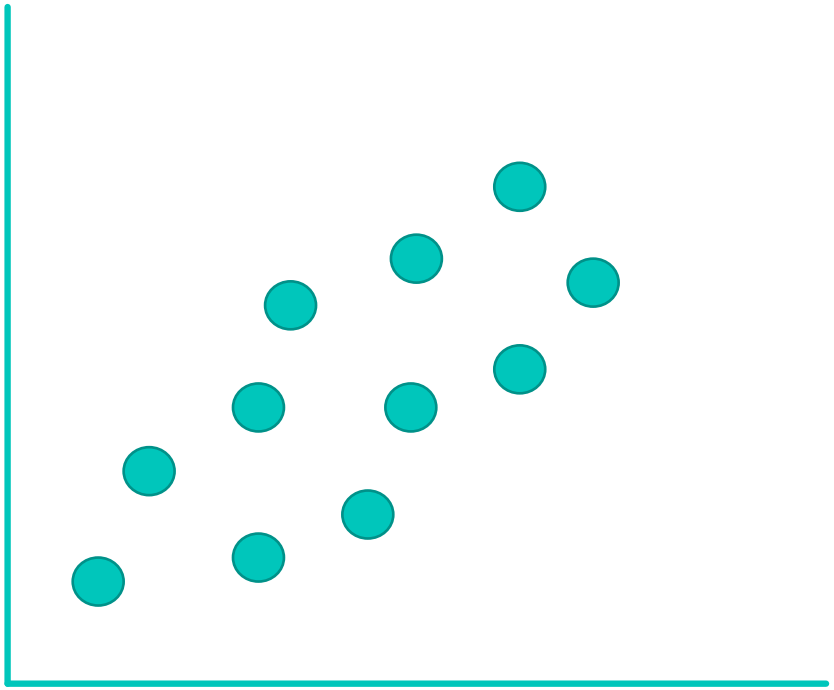
Support Vector Machine Concept - – C 값 Gamma 값



Support Vector Machine Concept – Variance and Bias

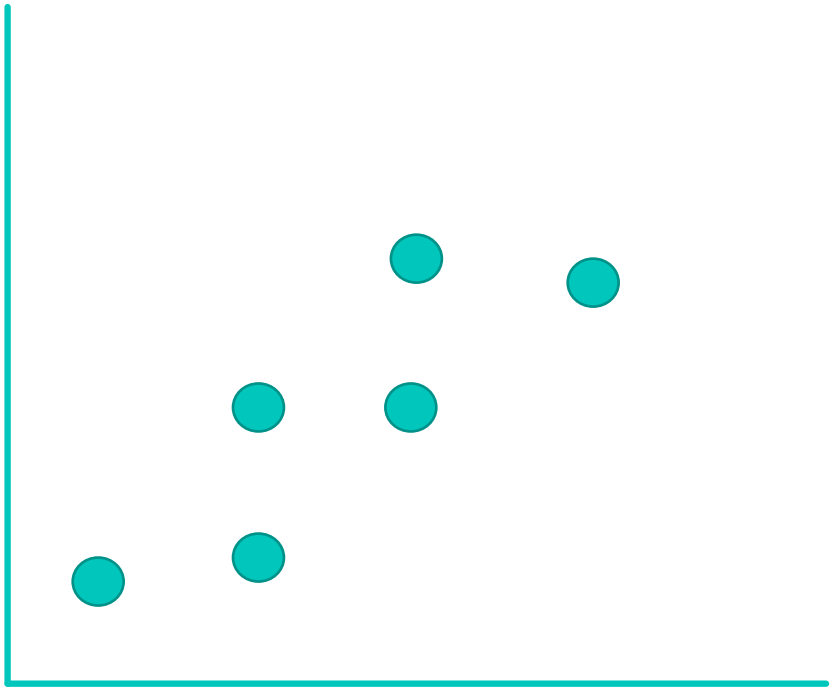


Support Vector Machine Concept – Variance and Bias

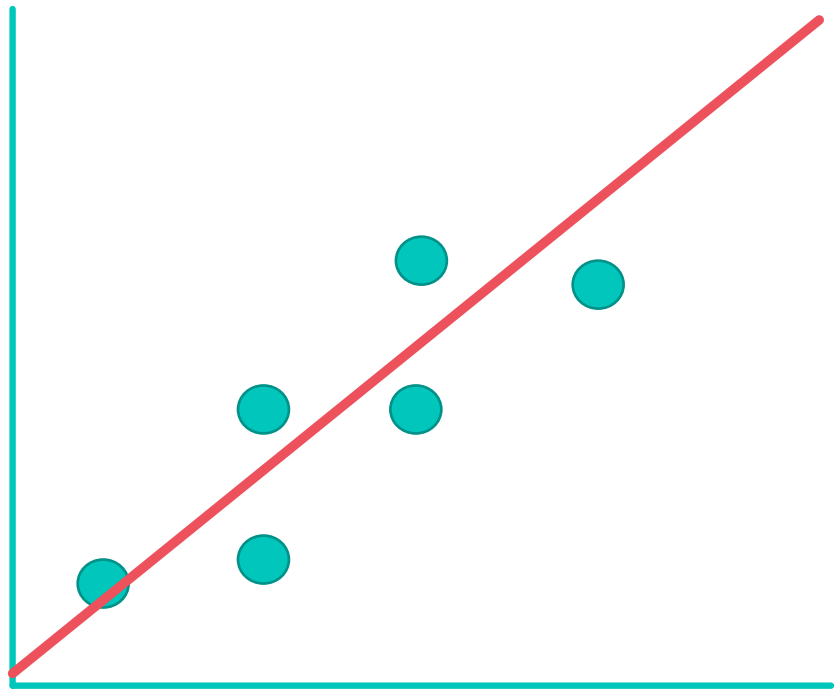


Sampling 수행

Support Vector Machine Concept – Variance and Bias

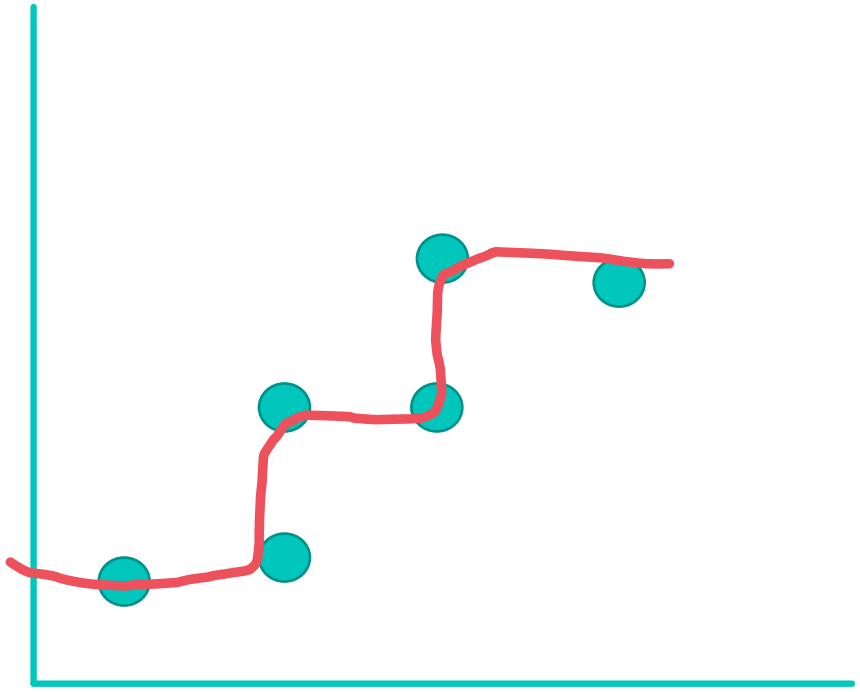


Support Vector Machine Concept – Variance and Bias



선형 모델 생성
잔차 50

Support Vector Machine Concept – Variance and Bias



비선형 모델 생성

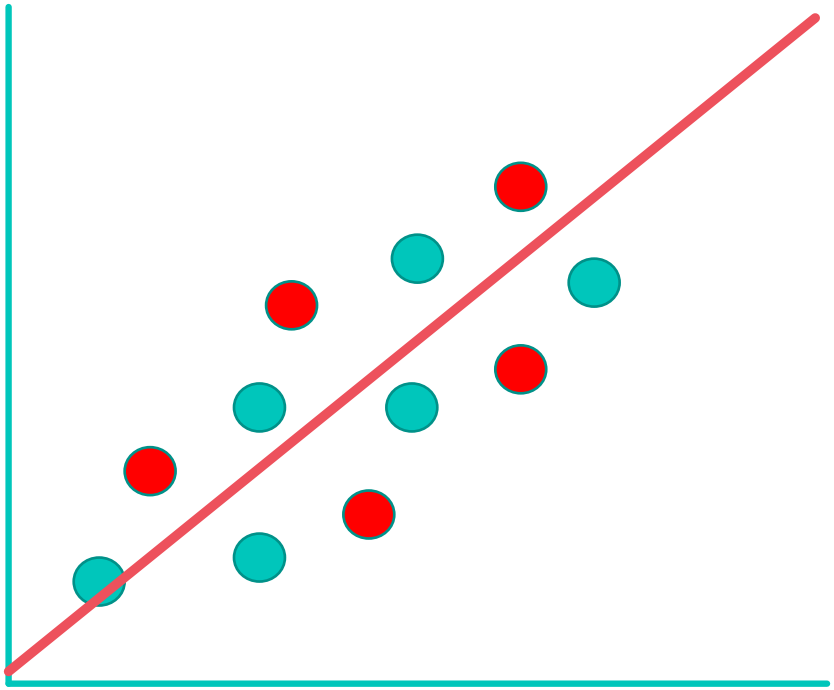
잔차 0.1

Support Vector Machine Concept – Variance and Bias

BUT,

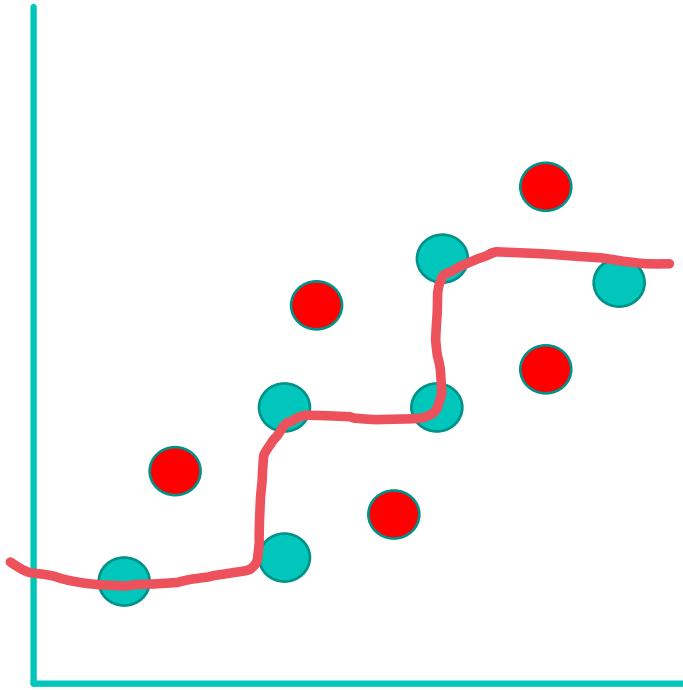
BUT, 테스트

Support Vector Machine Concept – Variance and Bias



선형 모델 테스트
잔차 50

Support Vector Machine Concept –Bias and Variance



비선형 모델 테스트

잔차 : 120

오버피팅(overfitting) :
훈련은 잘하나, 실전은 약한모델
훈련데이터에 너무 많이 적응
bias 가 심함

Support Vector Machine Concept –Bias and Variance

Bias 와 Variance 는 Trade-Off 관계 이며,
최적의 Cut-Off 지점을 찾는 것이 중요

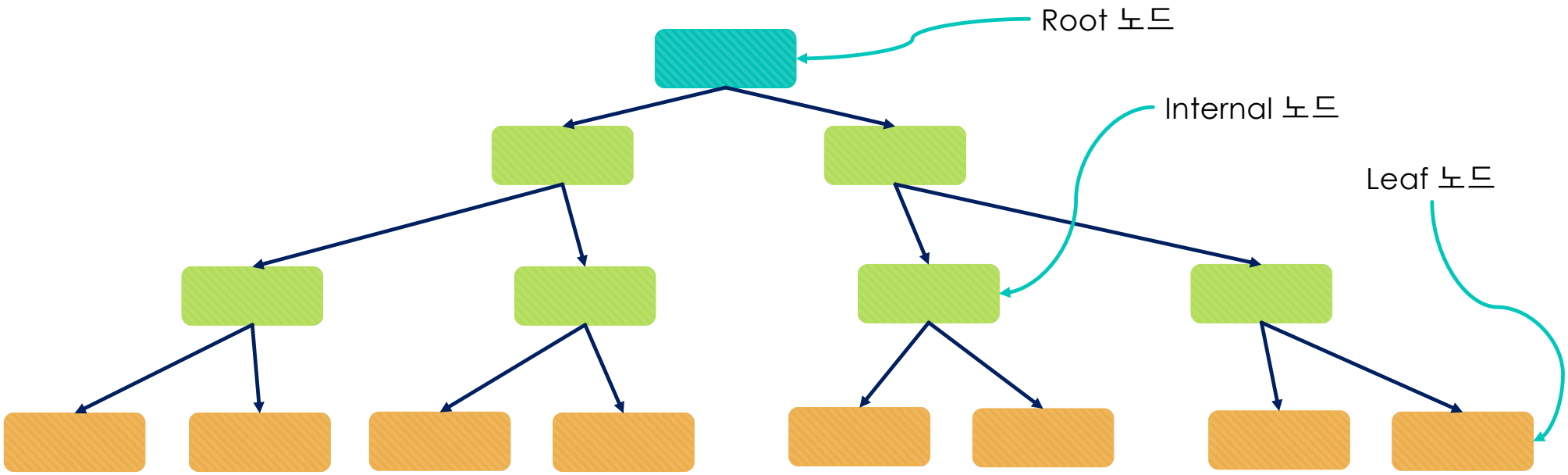
Support Vector Machine Concept - – C 값 Gmma 값

민감도 특이도

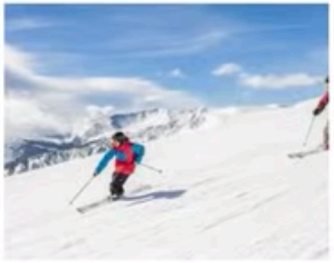
장. Decision Tree

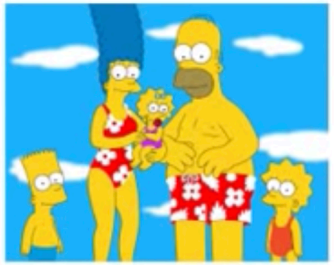
Decision Tree Concept

- 첫번째 노드를 root 노드
- 중간에 있는 노드를 Internal 노드
- 마지막 노드를 leaf 노드



Decision Tree Concept



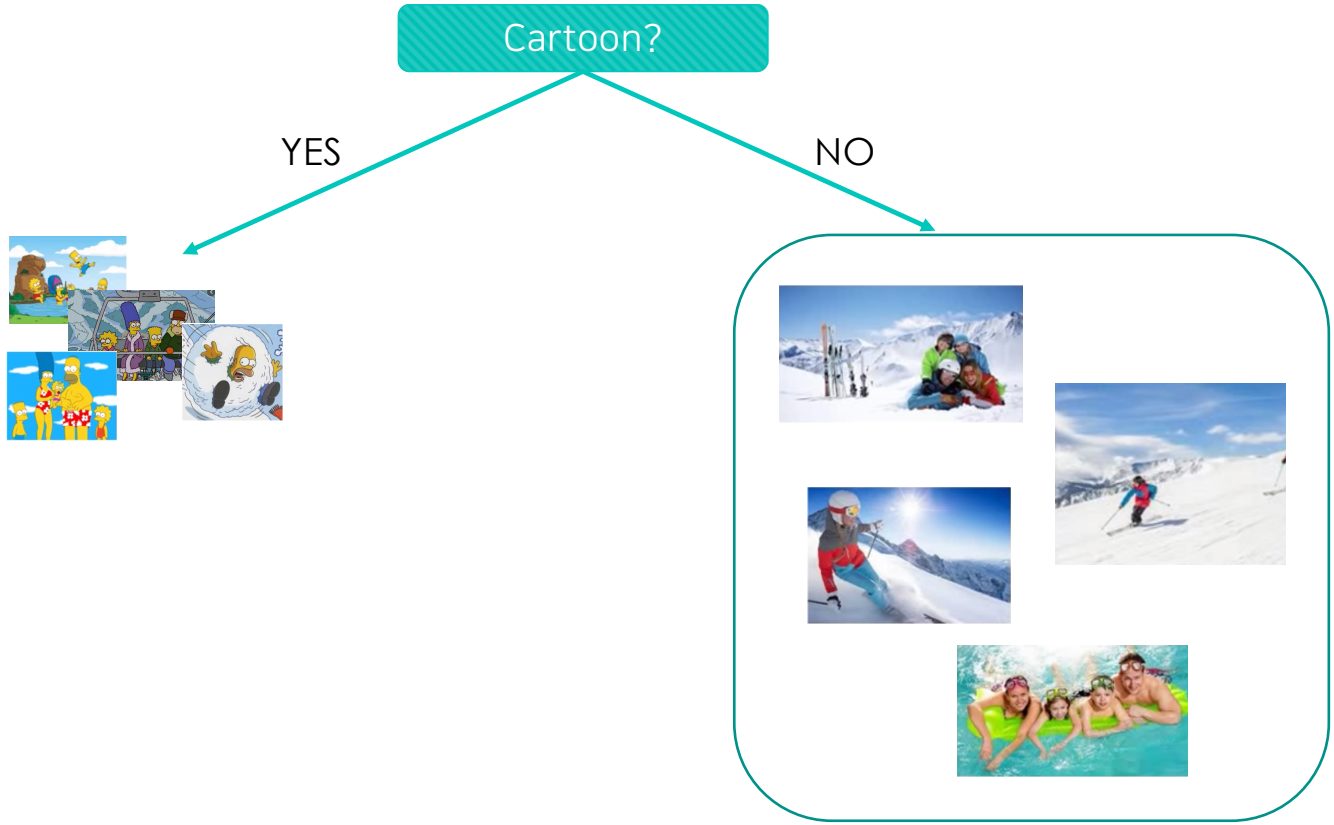




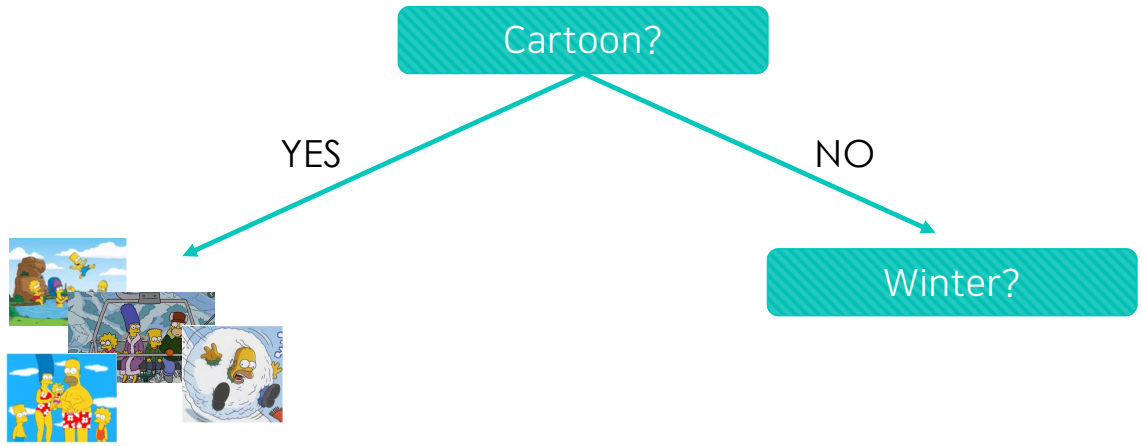

Decision Tree Concept

Cartoon?

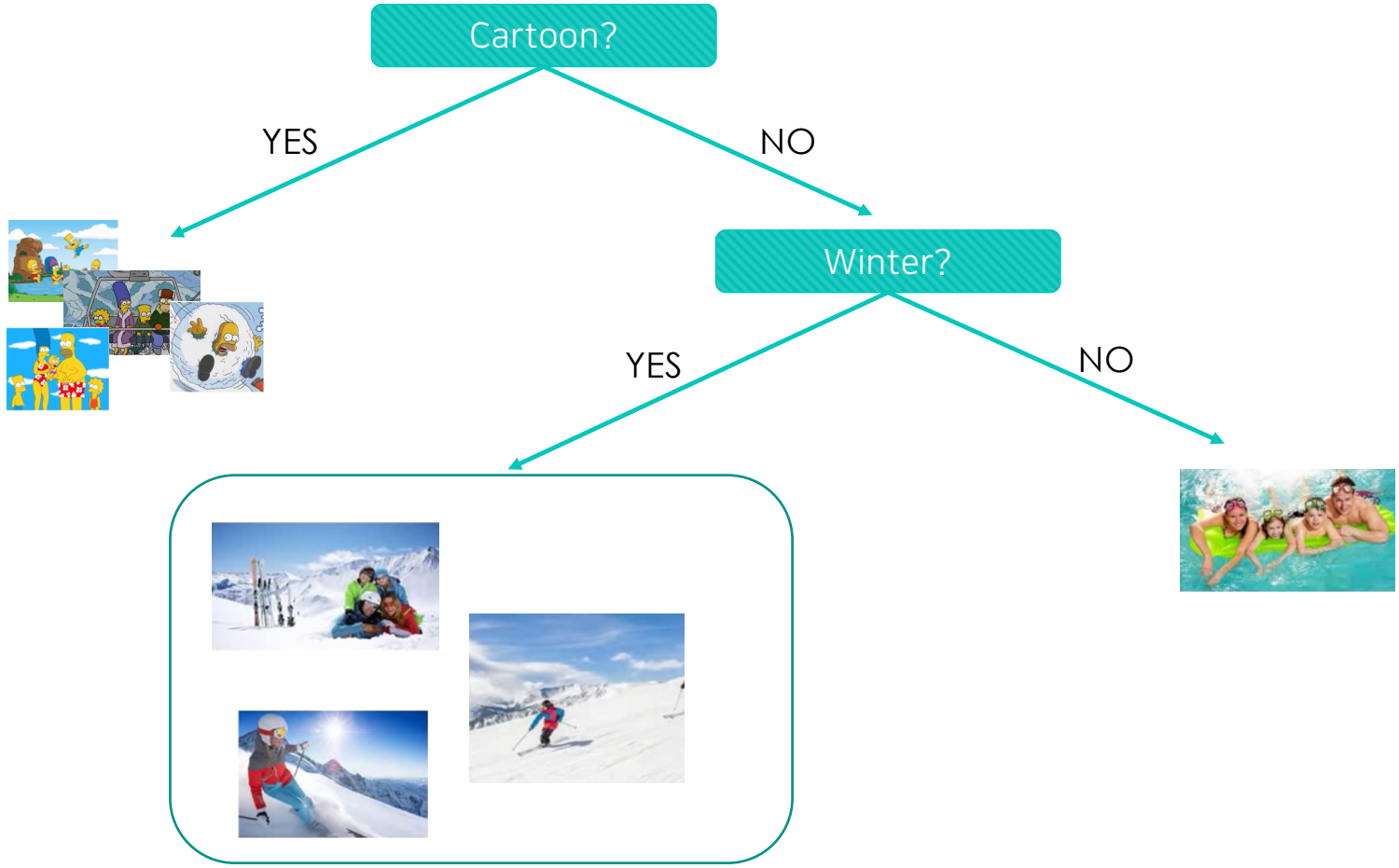
Decision Tree Concept



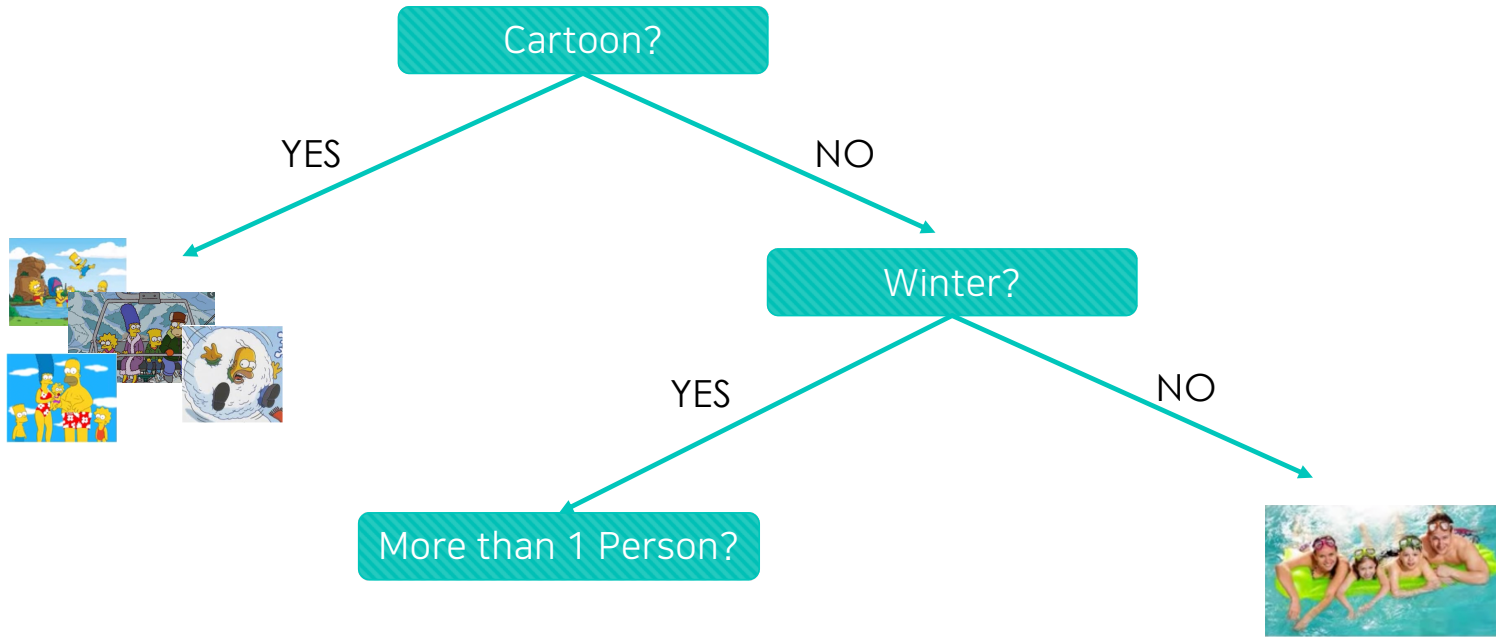
Decision Tree Concept



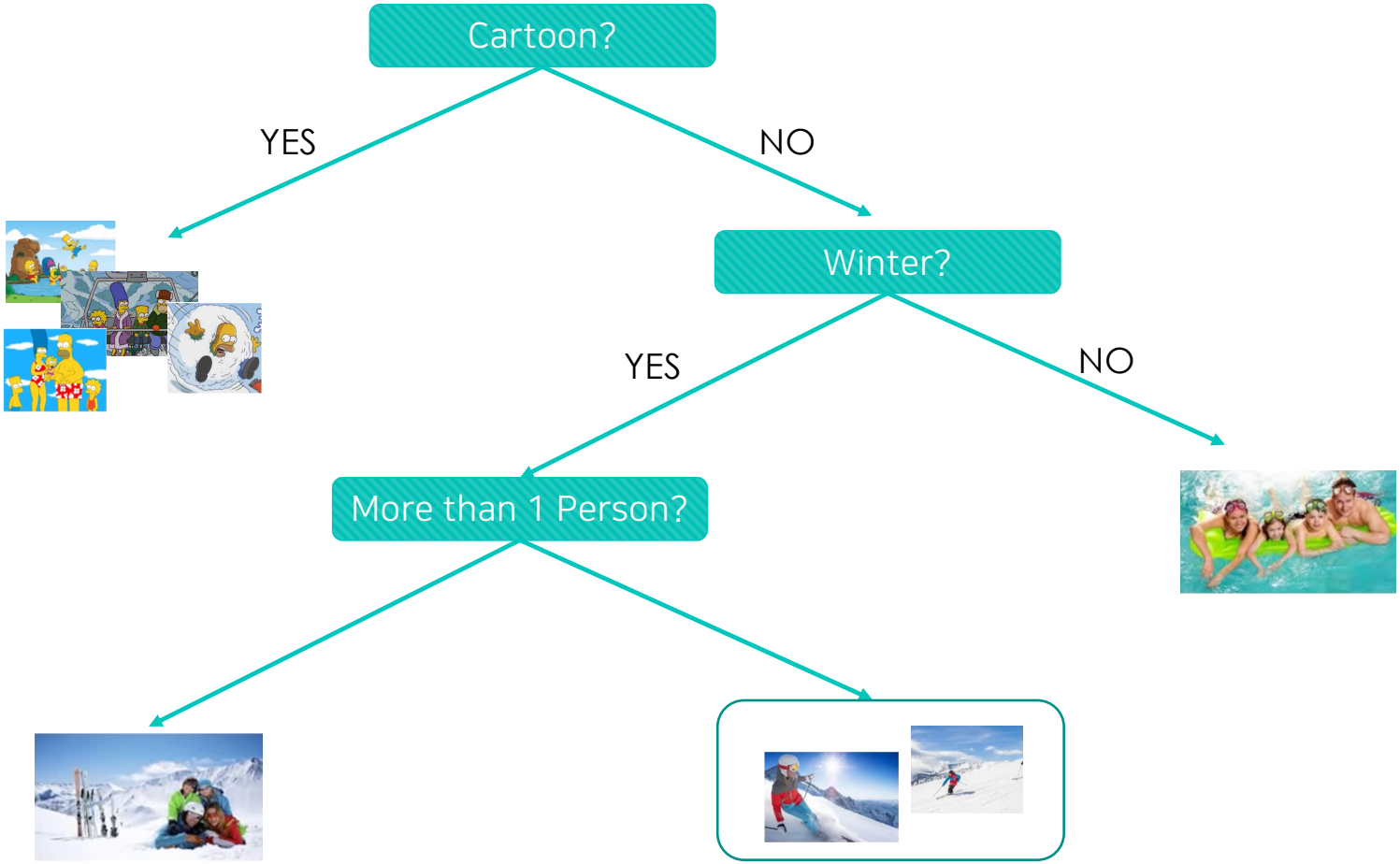
Decision Tree Concept



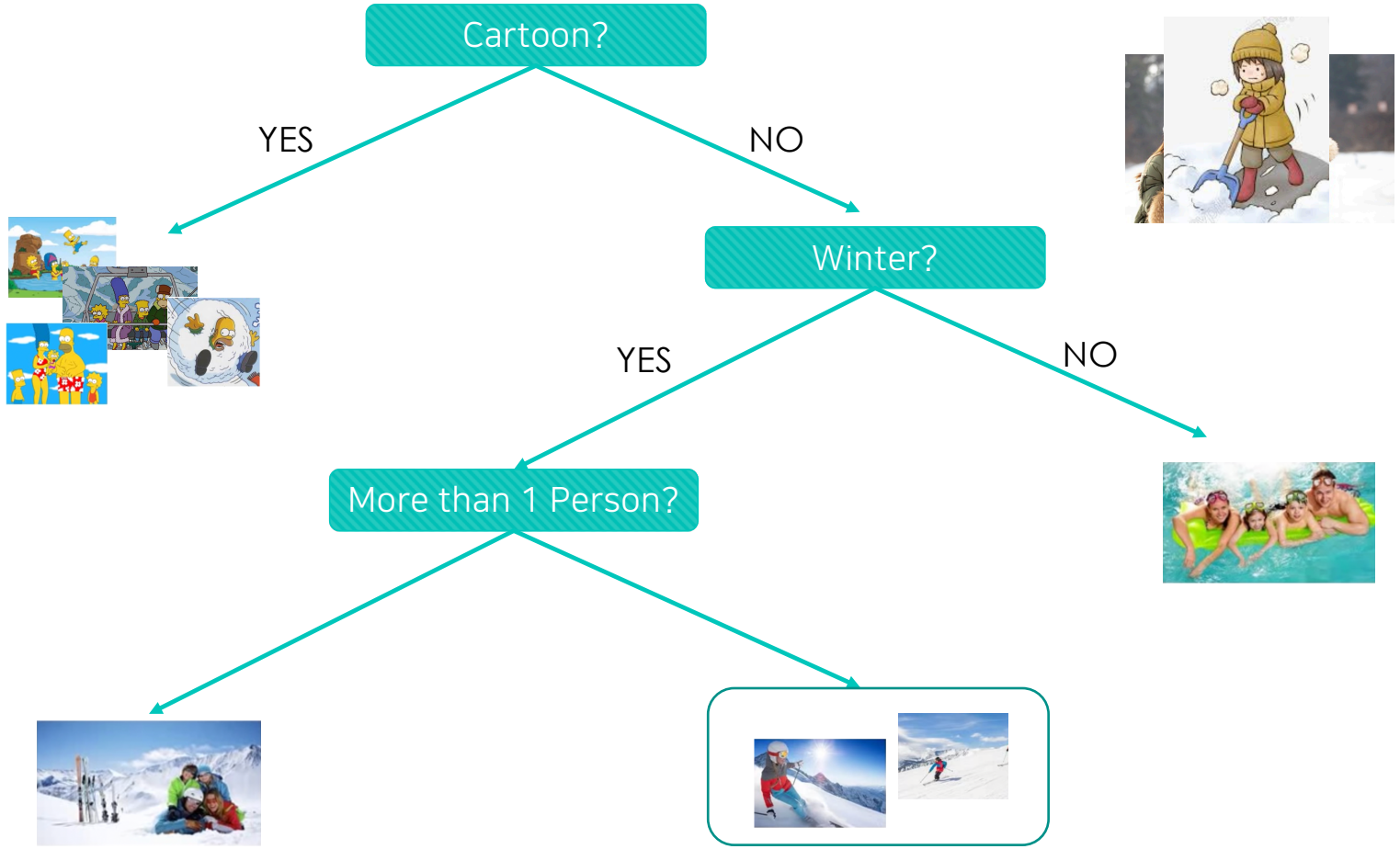
Decision Tree Concept



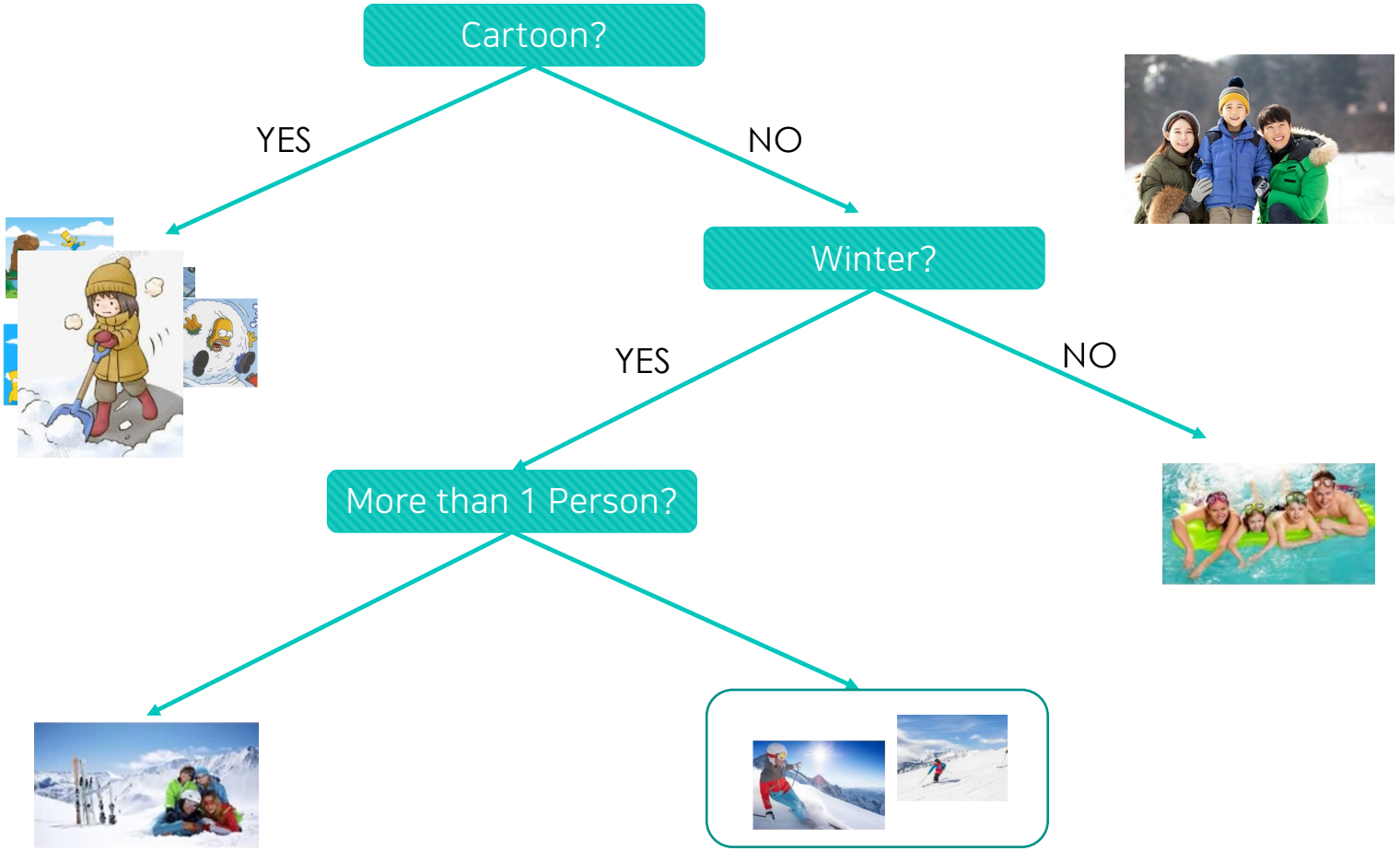
Decision Tree Concept



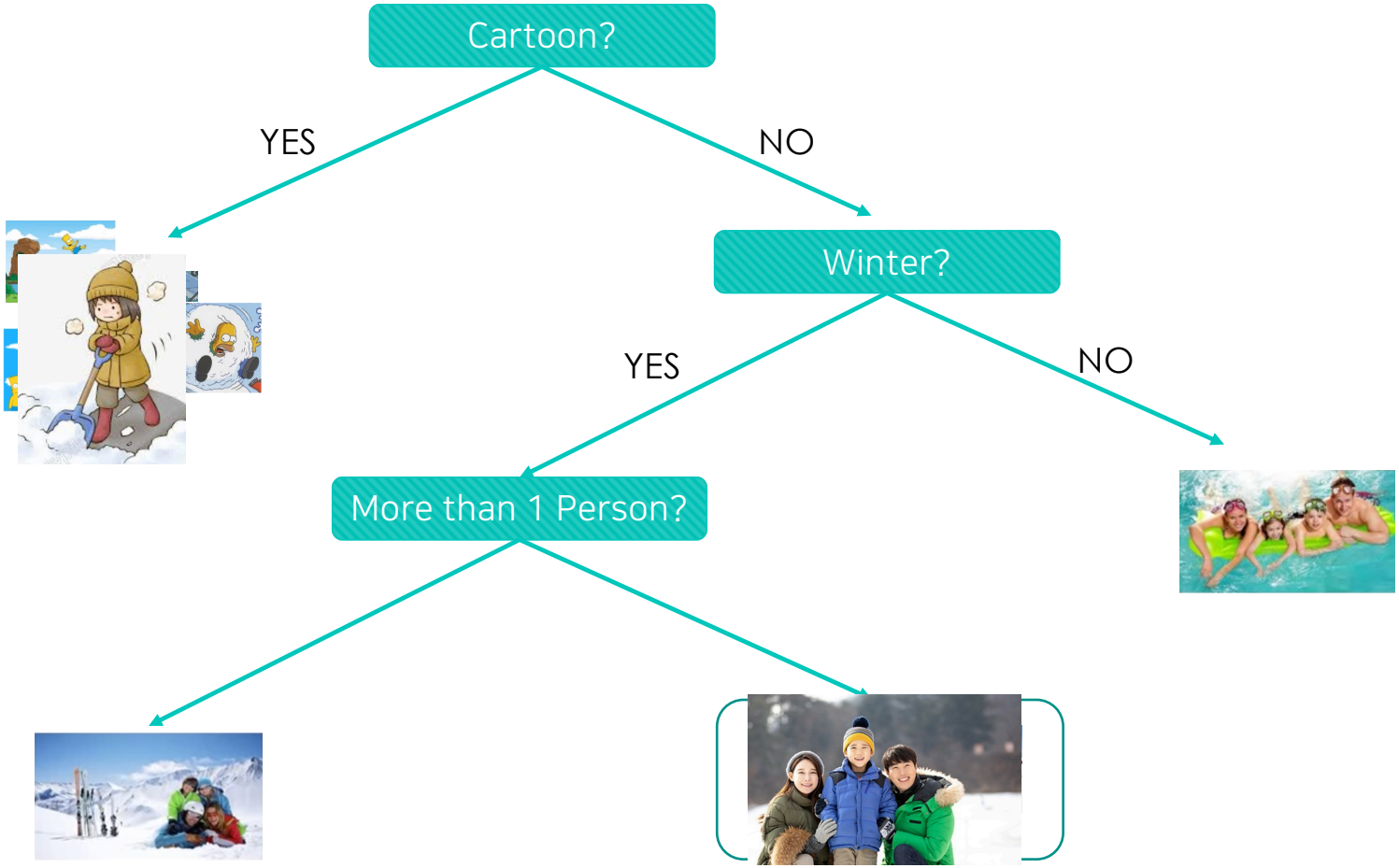
Decision Tree Concept



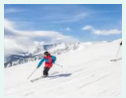
Decision Tree Concept



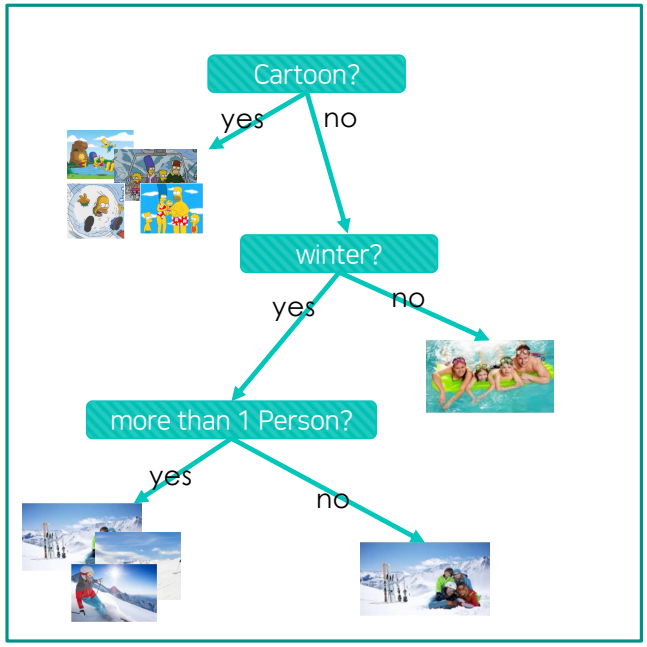
Decision Tree Concept

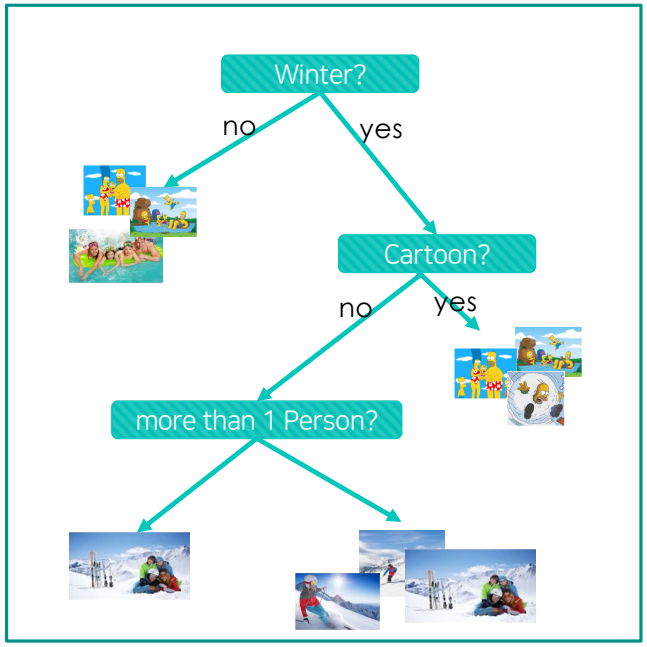


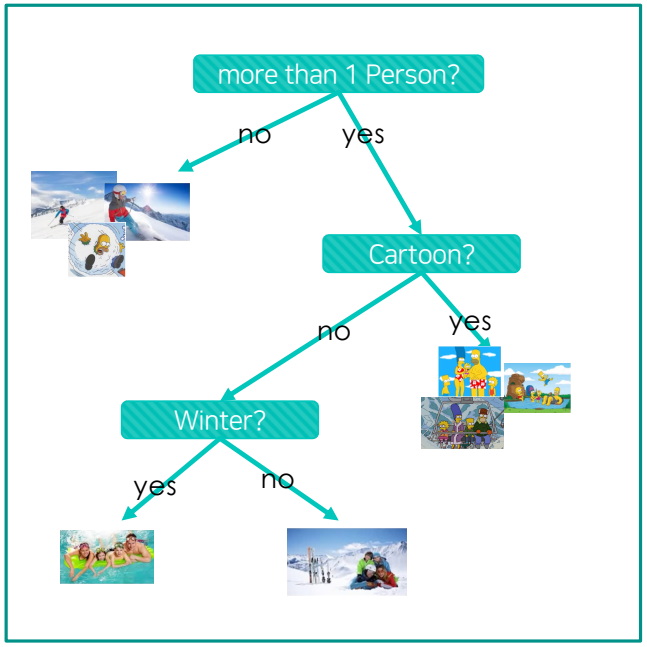
Decision Tree Concept

image	Cartoon	winter	person >1	겨울가족사진
	No	Yes	Yes	Yes
	No	Yes	No	No
	Yes	No	Yes	No
	Yes	Yes	Yes	No
	No	Yes	No	No
	No	No	Yes	No
	Yes	No	Yes	No
	Yes	Yes	No	No

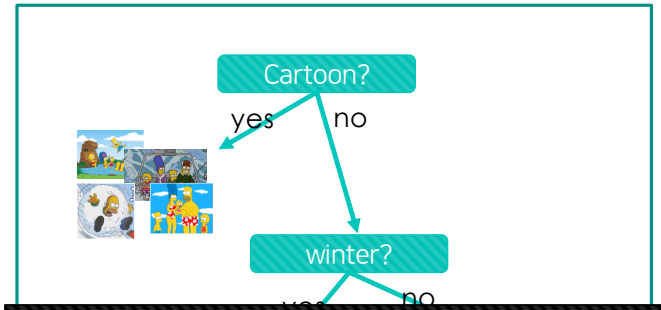
Decision Tree Concept



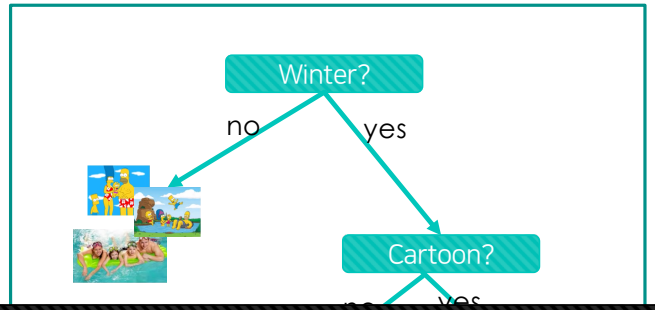




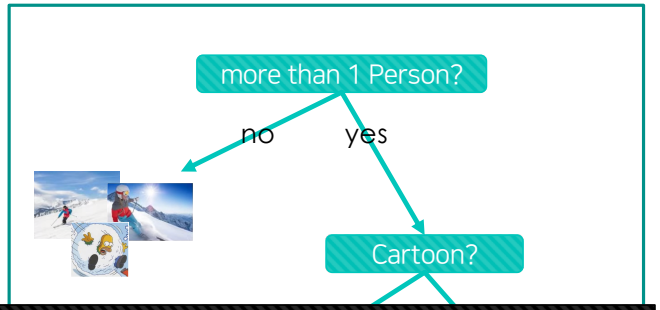
Decision Tree Concept



8장 사진 중
4장 남음

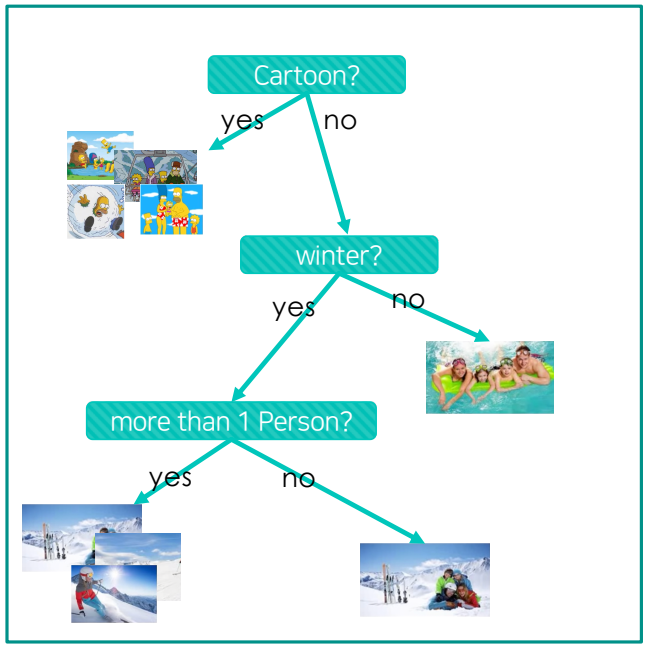


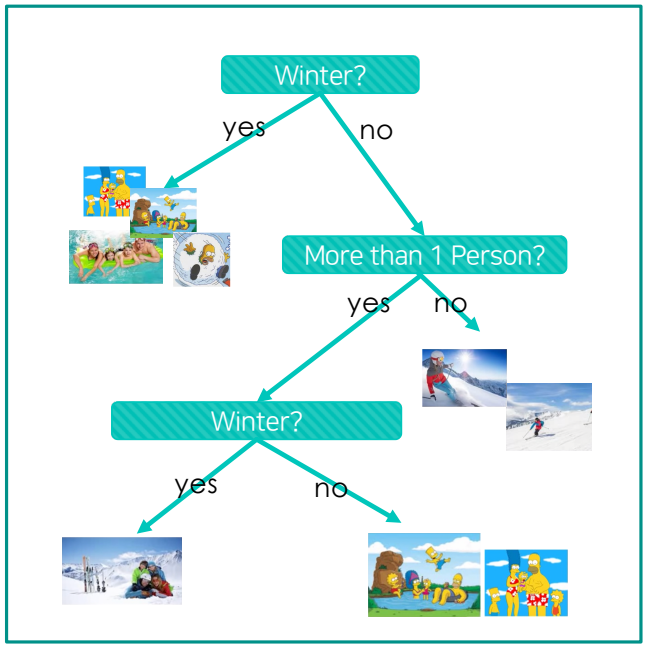
8장 사진 중
3장 남음



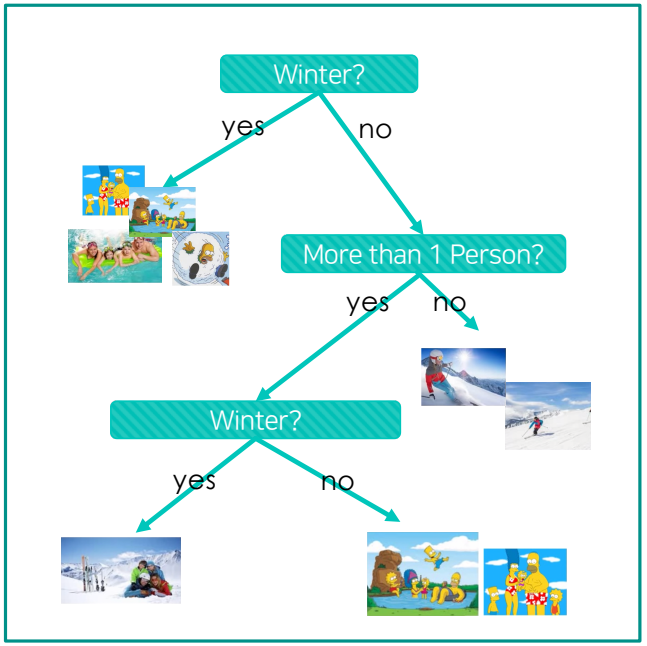
8장 사진 중
3장 남음

Decision Tree Concept

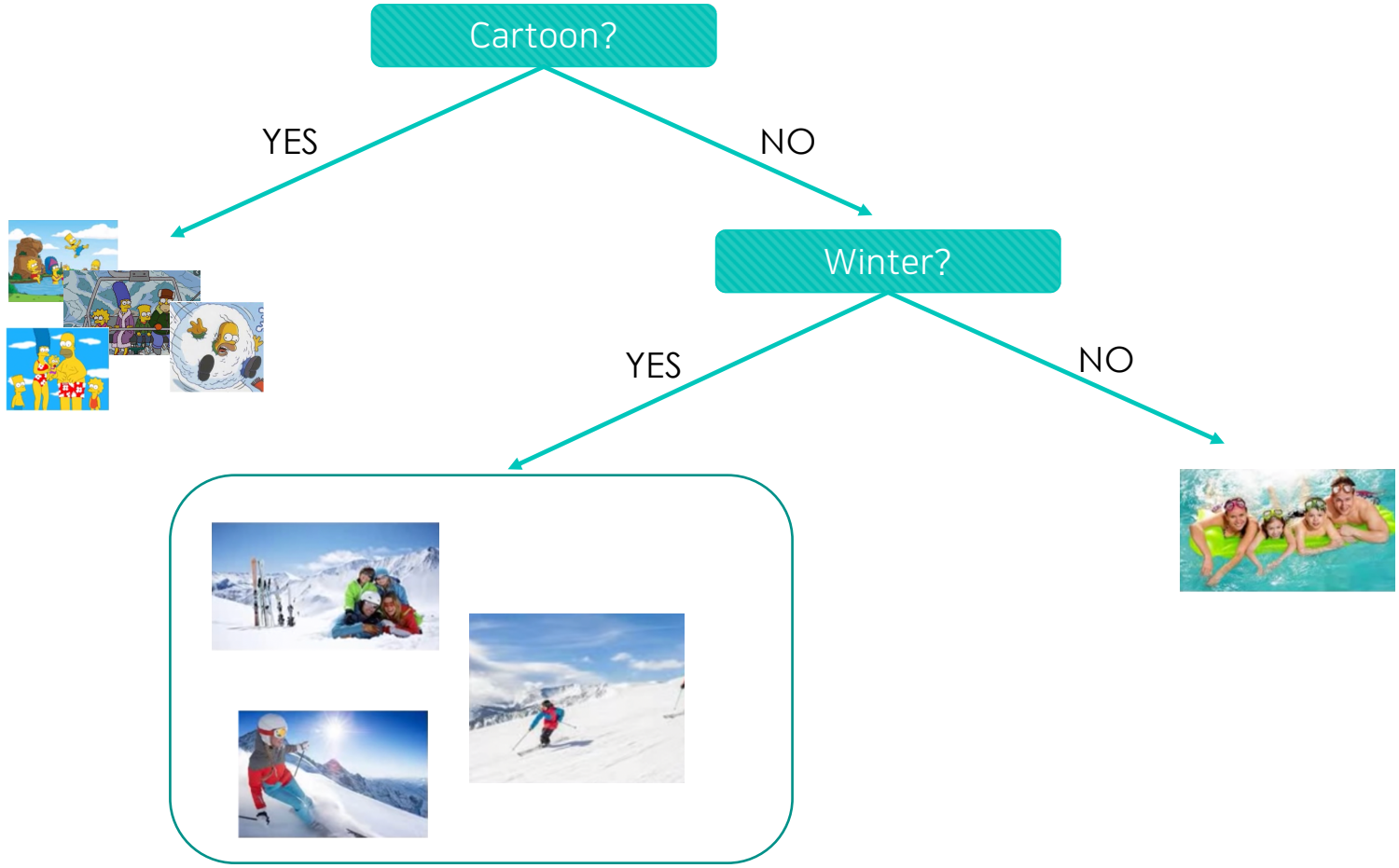




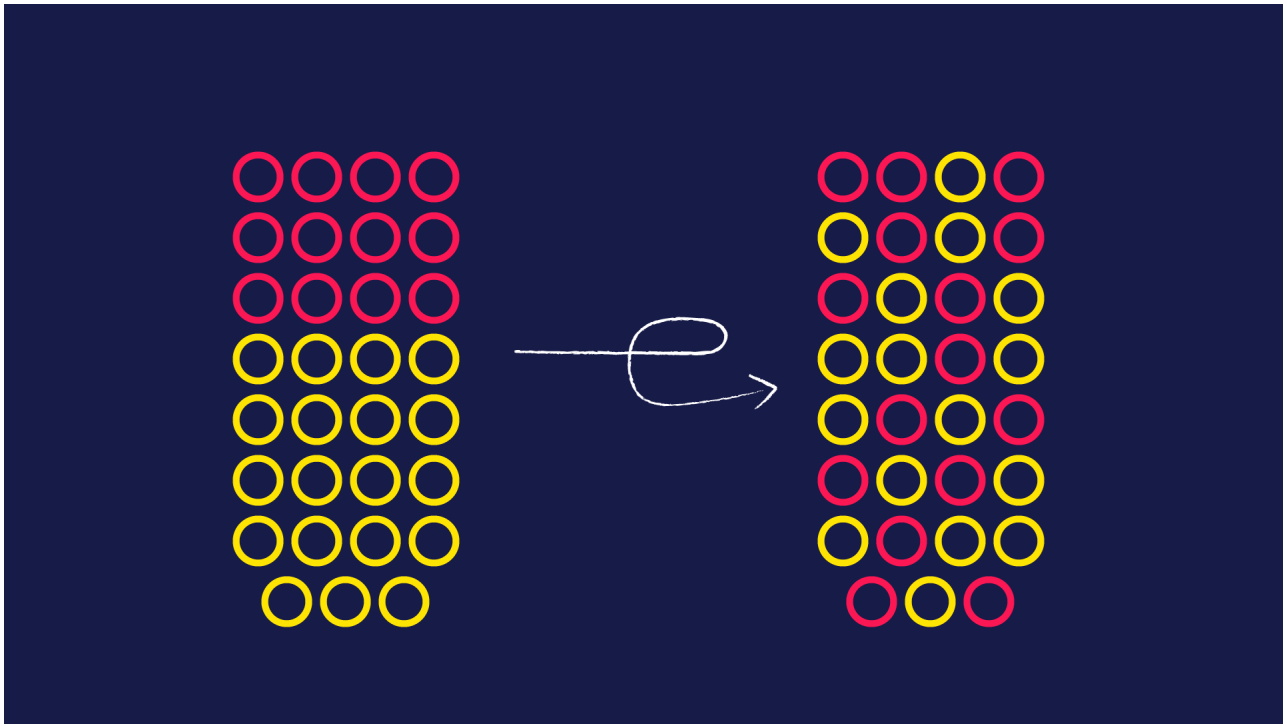
Decision Tree Concept



Decision Tree Concept

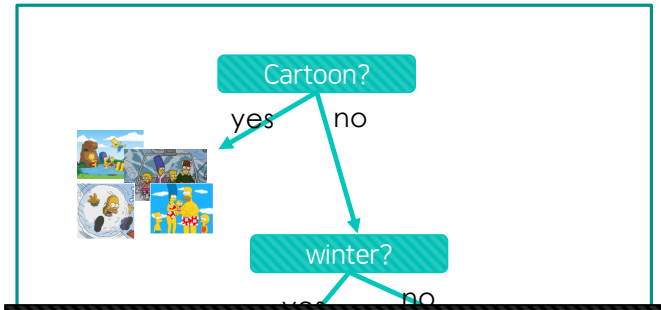


Decision Tree Concept - Entropy

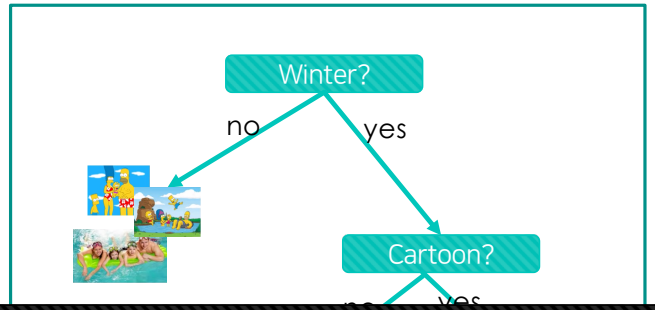


Decision Tree Concept - Entropy

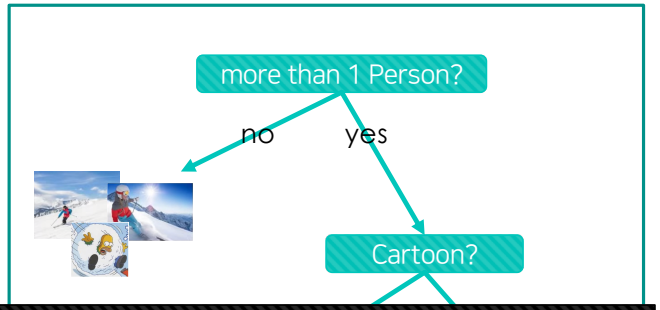
- 사진을 더 많이 걸러낼수록 이후 Entropy 가 감소하게 됩니다.
- Entropy를 낮게 할수록 좋은 노드를 선택 하는 것입니다.



8장 사진 중
4장 남음



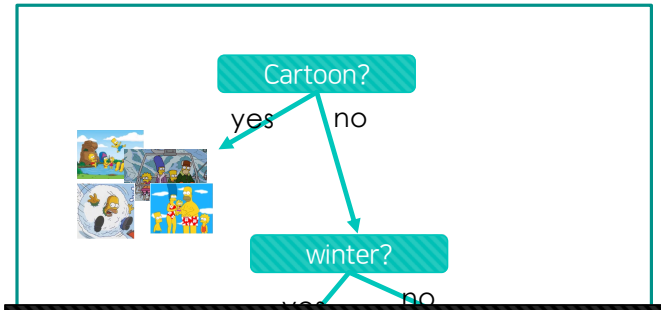
8장 사진 중
3장 남음



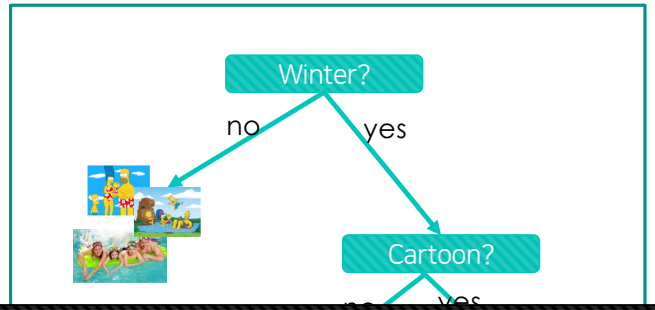
8장 사진 중
3장 남음

Decision Tree Concept - Entropy

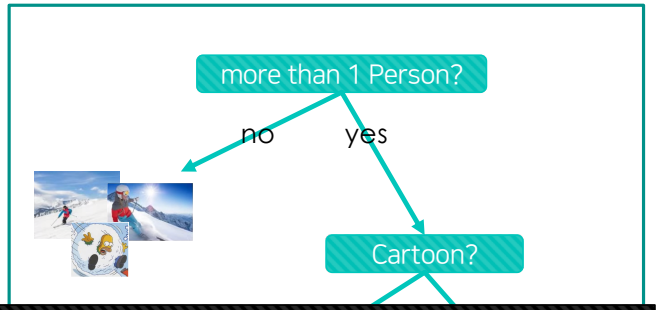
- Information Gain 은 [Base Entropy – New Entropy]
- Information Gain 이 높을 수록 좋은 선택



8장 사진 중
4장 남음
8 - 4 = 4



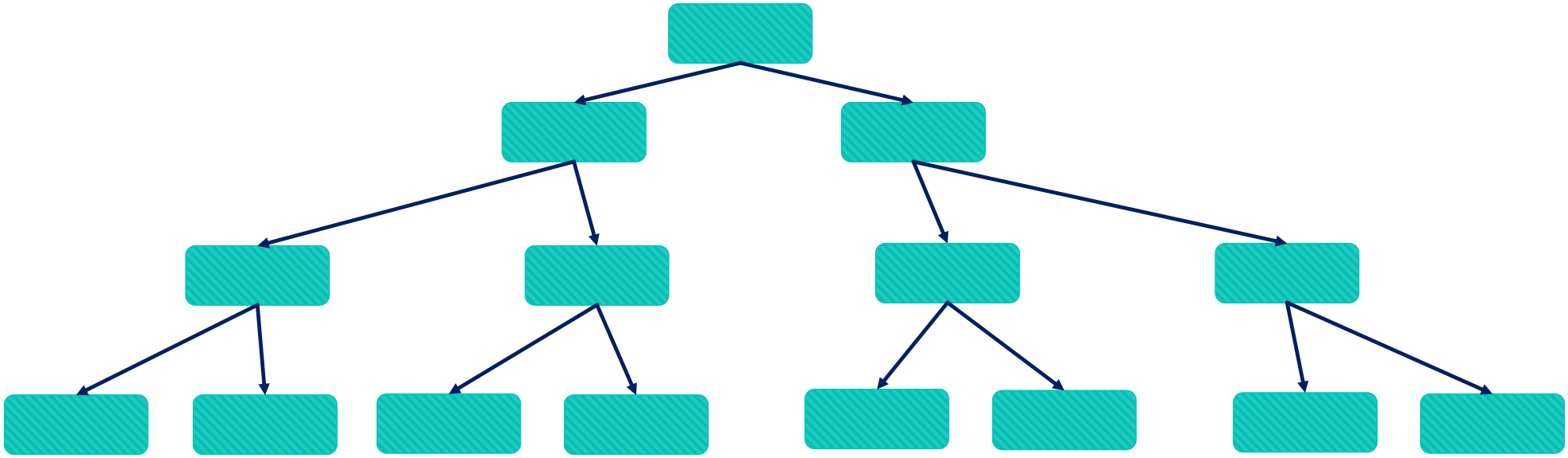
8장 사진 중
5장 남음
8 - 5 = 3



8장 사진 중
5장 남음
8 – 5 = 3

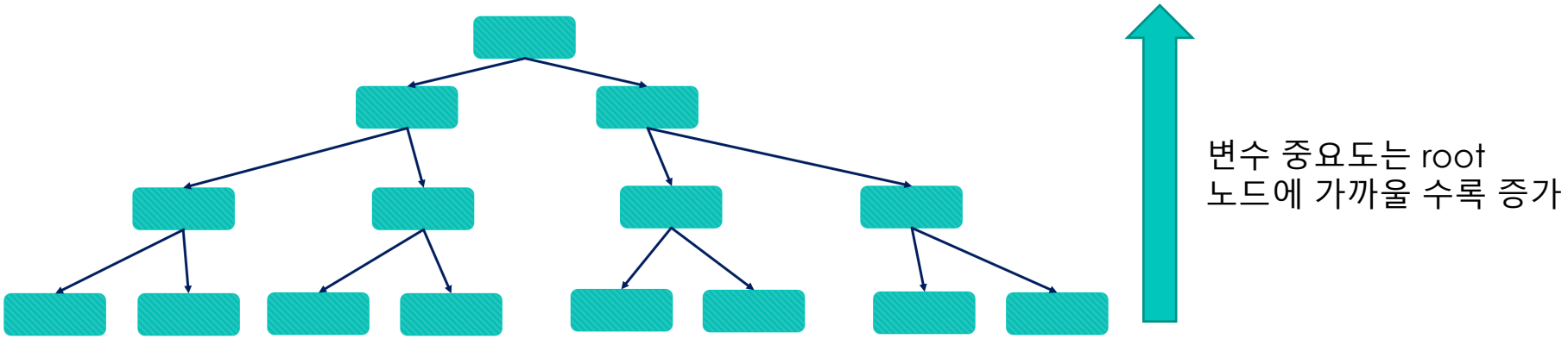
Decision Tree Concept

- Decision Tree 의 경우 Information Gain 의 순서에 따라 root 노드부터 차례로 선택 됩니다.
- 결국 Information Gain 을 계산하는 것은 **어떤 질문의 먼저 할 것 인가의 문제**



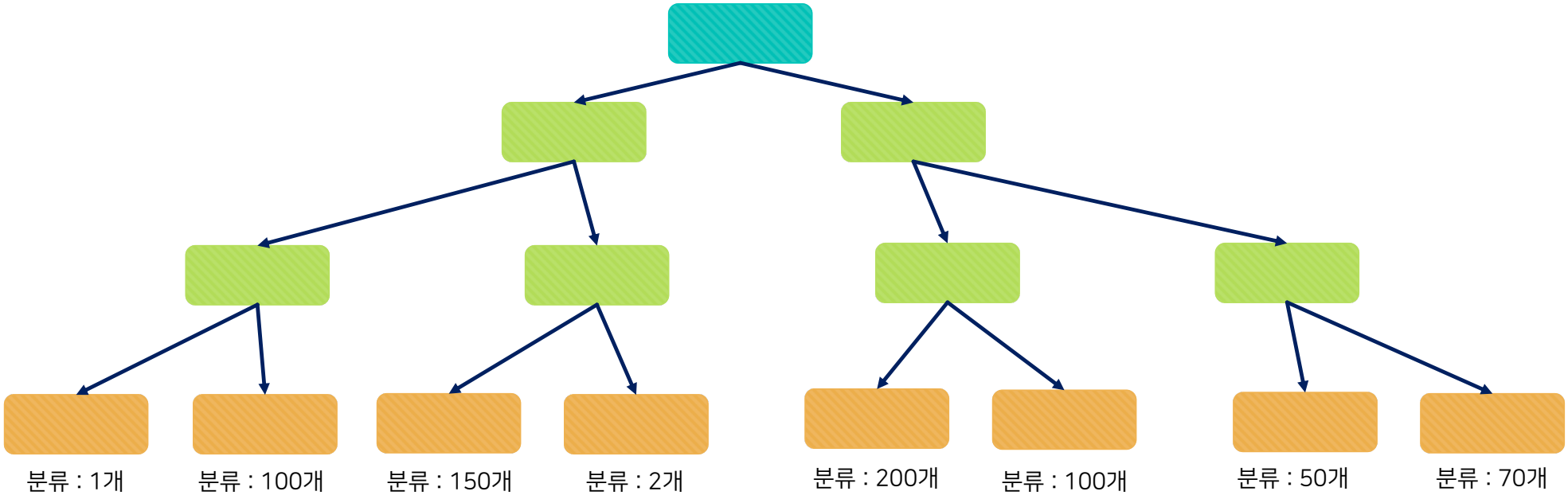
Decision Tree Concept - Entropy

- 변수 중요도는 root 노드에 가까울 수록 증가 합니다.



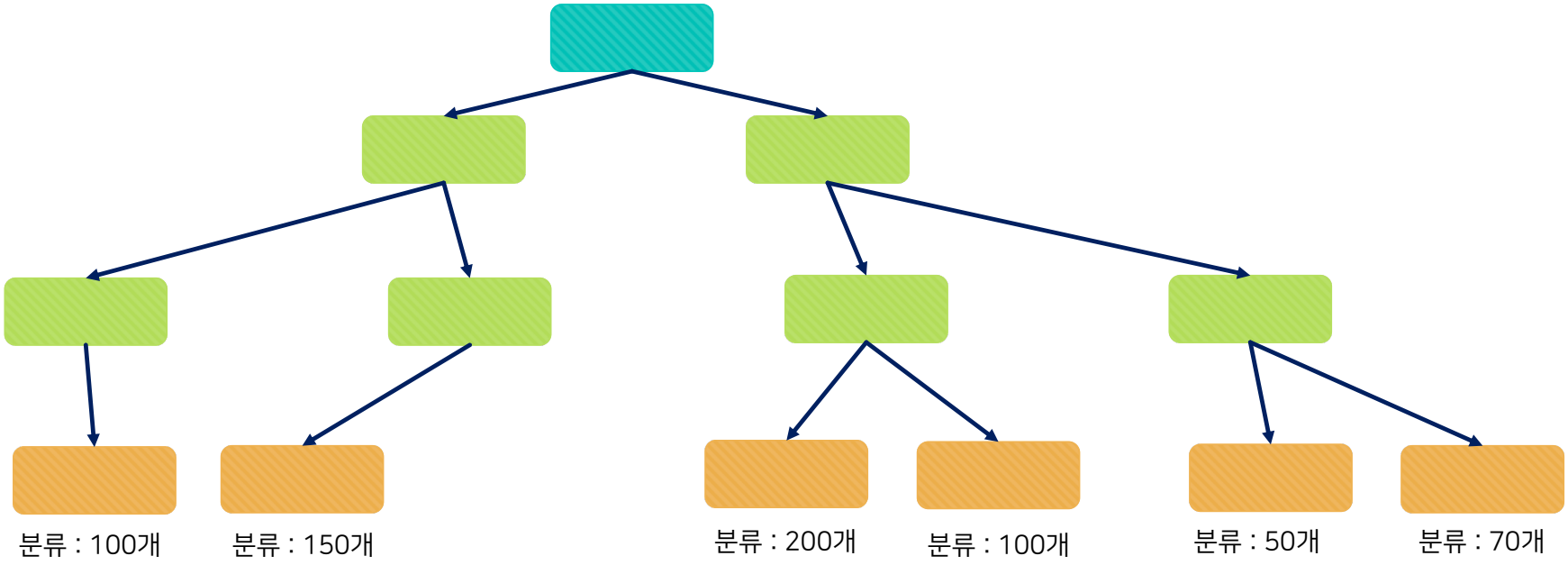
Decision Tree Concept – Pruning

- Leaf 노드의 분류 개수가 다른 노드에 비해 너무 작을 때 Over fitting 될 가능 성이 많습니다.



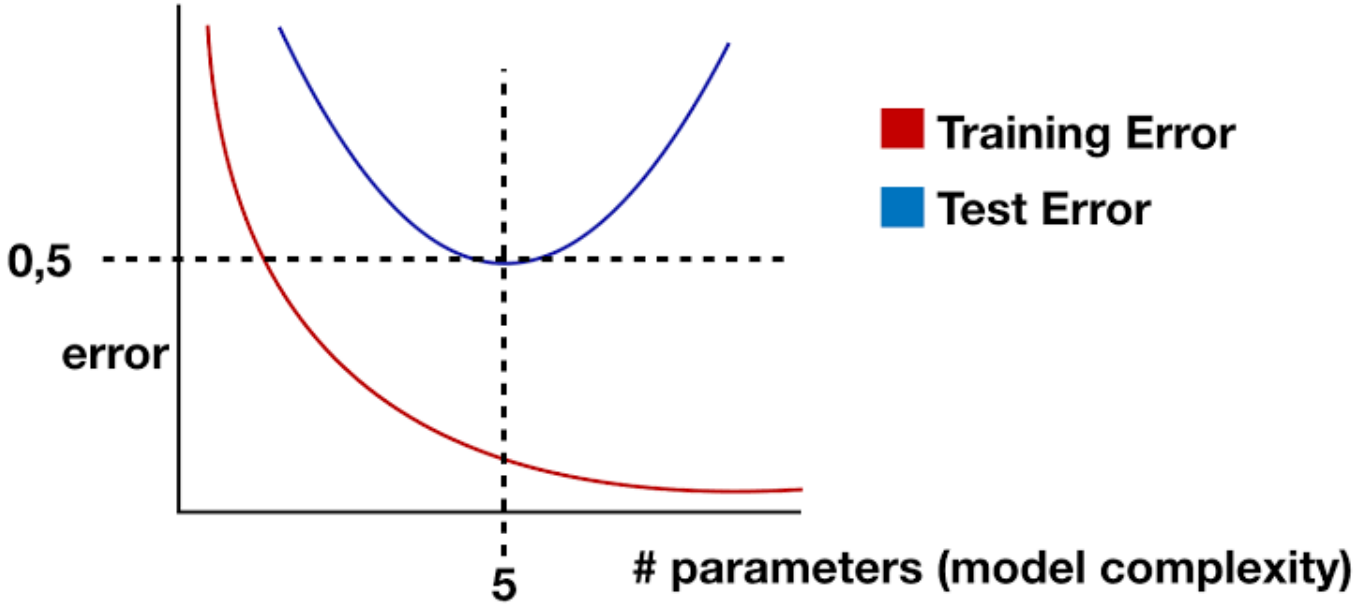
Decision Tree Concept – Pruning

- Leaf 노드의 분류 개수가 다른 노드에 비해 너무 작을 때 Over fitting 될 가능 성이 많습니다.
- 이때는 모델의 성능을 위해 해당 노드를 제거 하기도 합니다.
- Training 데이터셋 을 가지고 트리 개수를 무한정 늘리게 되면 후련 데이터 에만 맞는 모델이 됩니다.



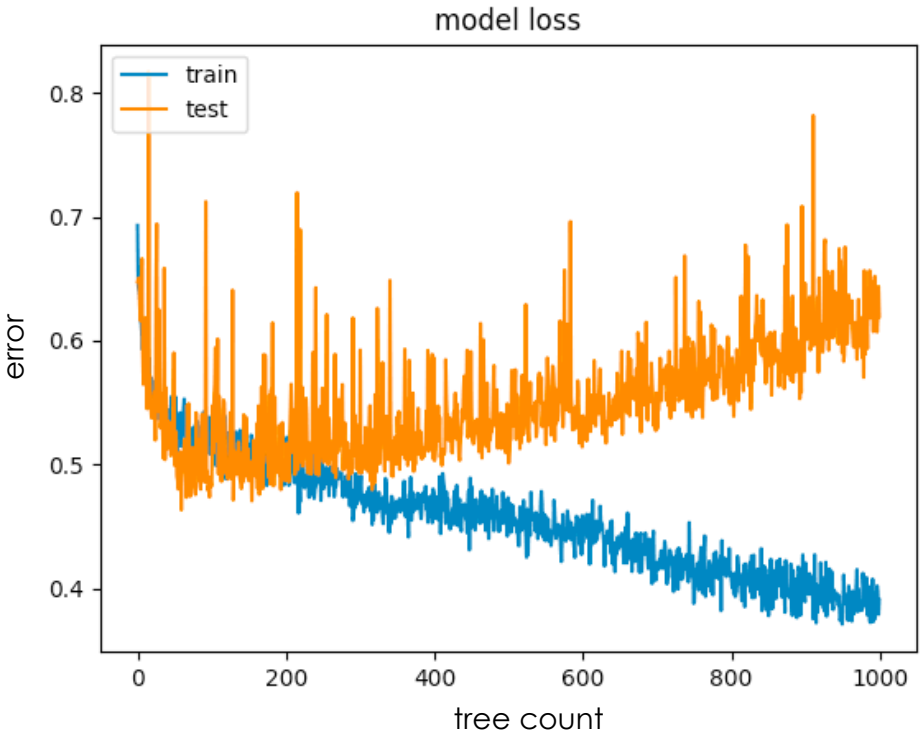
Over fitting for model Concept

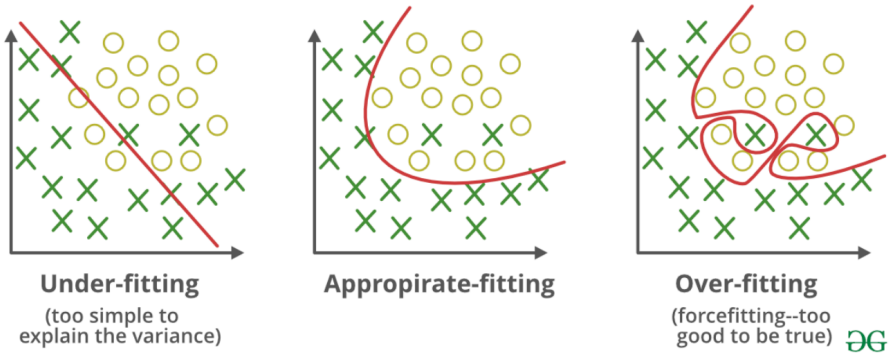
- 훈련 데이터 셋과 테스트 셋을 트리 개수를 늘려가며 성능 측정을 해보면 아래와 같은 그림을 그릴 수 있습니다.
- 따라서 적당은 트리 갯수는 테스트 데이터 셋의 오류가 가장 적은 지점 입니다.



Over fitting for model on real world

- 아래는 실제 iris 데이터셋을 훈련 셋과 테스트 셋을 나누고 트리 갯수를 늘려 갈 때의 성능 오류 율 입니다.





ID3 (Entropy and Information Gain)

image	Cartoon	winter	person >1	겨울가족사진
	No	Yes	Yes	Yes
	No	Yes	No	No
	Yes	No	Yes	No
	Yes	Yes	Yes	No
	No	Yes	No	No
	No	No	Yes	No
	Yes	No	Yes	No
	Yes	Yes	No	NO

총 사진 개수 : 8장
겨울가족사진 개수(정답) : 1장
겨울가족사진 아닌 것 : 7장

Total Entropy = ([1 + , 7 -])
= - (1 / 8) * log (1 / 8) – (7 / 8) * log (/ 8)
= 0.543

Entropy = - P(+) * log (p (+)) – P(-) * log(p (-))

P(+) : 정답일 확률
P(-) : 오답일 확률

ID3 (Entropy and Information Gain)

image	Cartoon	winter	person >1	겨울가족사진
	No	Yes	Yes	Yes
	No	Yes	No	No
	Yes	No	Yes	No
	Yes	Yes	Yes	No
	No	Yes	No	No
	No	No	Yes	No
	Yes	No	Yes	No
	Yes	Yes	No	NO

information gain
= total Entropy – entropy when choos feature

information gain
= total Entropy – E (choose Cartoon)
= 0.543 – (4/8 * E([0+,4-] + 4/8 * E([1+, 3-])
= 0.138

information gain
total Entropy – E (choose Winter)
= 0.543 – (5/8 * E([1+,4-]) + 3/8 * E([0+, 3-])
= 0.093

information gain
total Entropy – E (choose Person)
= 0.542 – (5/8 * E([1+,4-]) + 3/8 * E([0+,3-])
= 0.093

ID3 (Entropy and Information Gain)

평면에서 ID3 를 도식화 하면 쉬울 것임

ID3 Alternative

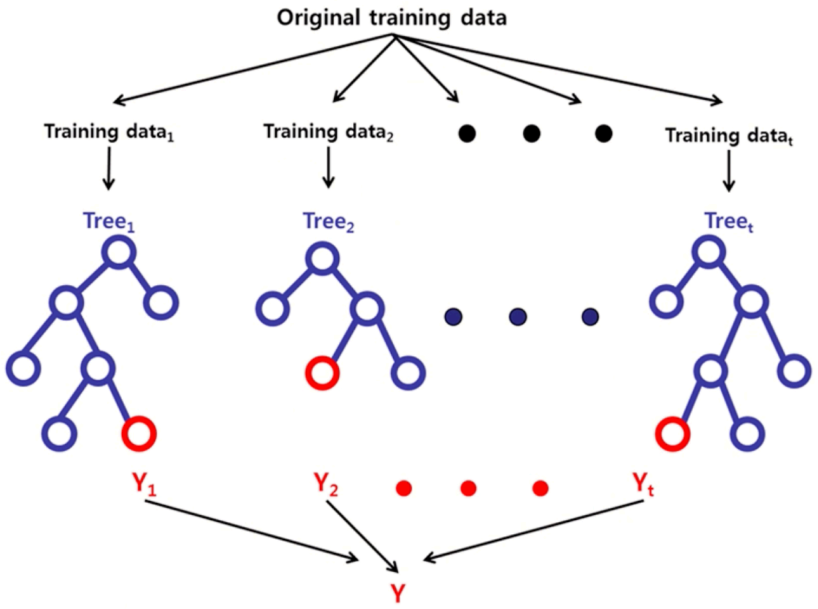
- ID3 는 Feature 의 카테고리의 개수가 많을 수록 Information Gain 이 증가하여 왜곡 가능성이 있습니다.
- 이를 보완하기 위해 나온 것이 C4.5 , CART 등이 있습니다. 각 알고리즘의 특성은 아래와 같습니다.

	분할 조건	Feature 타입	결측	Pruning 전략	Outlier
ID3	Information Gain	범주형 만 허용	결측 허용 안함	No Pruning	Outlier 에 영향 받음
CART	Towing Criteria	범주형 및 연속형 허용	결측 허용	고비용 Pruning	비교적 Outlier 에 견고함
C4.5	Gain Ration	범주형 및 연속형 허용	결측 허용	에러 기반 Pruning	Outlier 에 영향 받음

장. Random Forrest

Random Forrest Concept

- 랜덤 하게 선택된 Training data 를 통해 다수의 트리를 생성
- Decision Tree 보도 일반적으로 성능 이 좋음
- 관측치가 작아 변수 선택이 어려울 때 중요 변수를 선택 하는 용도로도 쓰임



1. Bootstrap 기법을 이용하여 다수의 **training data** 생성

2. 생성된 **training data**로 **decision tree** 모델 구축 (무작위 변수를 사용하여)

3. 예측 종합

핵심 아이디어 : Diversity and Random

- 여러 개의 Training 데이터 로부터 다수의 Decision Tree 를 생성
- Decision Tree 생성시 무작위 Feature 선택

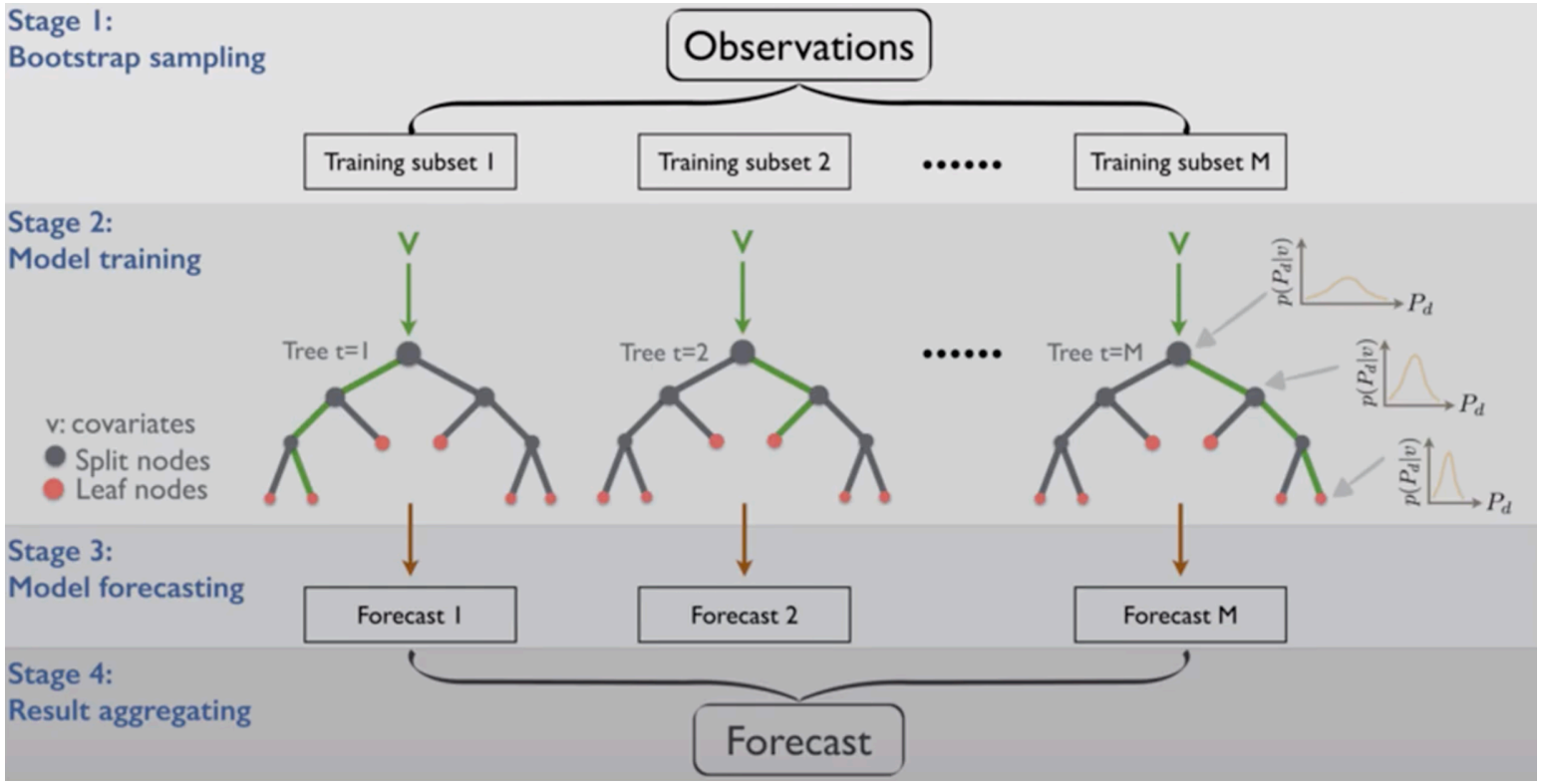
핵심 아이디어 : Diversity and Random

- 여러 개의 Training 데이터 로부터 다수의 Decision Tree 를 생성 **Bagging**
- Decision Tree 생성시 무작위 Feature 선택 **Random Subspace**

Random Forrest Concept

- Bagging(**B**ootstrap **Agg**regat**ing**) :

  Bootstrapping 을 통해 여러개 데이터 및 트리를 생성하고 각각의 트리에서 도출된 결과를 취합함



Random Forrest Concept - Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

Bootstrap 2

• • •

Bootstrap B

Random Forrest Concept - Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y^{10}
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

Bootstrap 2

Bootstrap B

Random Forrest Concept - Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y^{10}
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

Bootstrap 2

x^7	y^7
x^1	y^1
x^{10}	y^{10}
x^1	y^1
x^8	y^8
x^6	y^6
x^2	y^2
x^6	y^6
x^4	y^4
x^9	y^9

Bootstrap B

Random Forrest Concept - Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y^{10}
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

Bootstrap 2

x^7	y^7
x^1	y^1
x^{10}	y^{10}
x^1	y^1
x^8	y^8
x^6	y^6
x^2	y^2
x^6	y^6
x^4	y^4
x^9	y^9

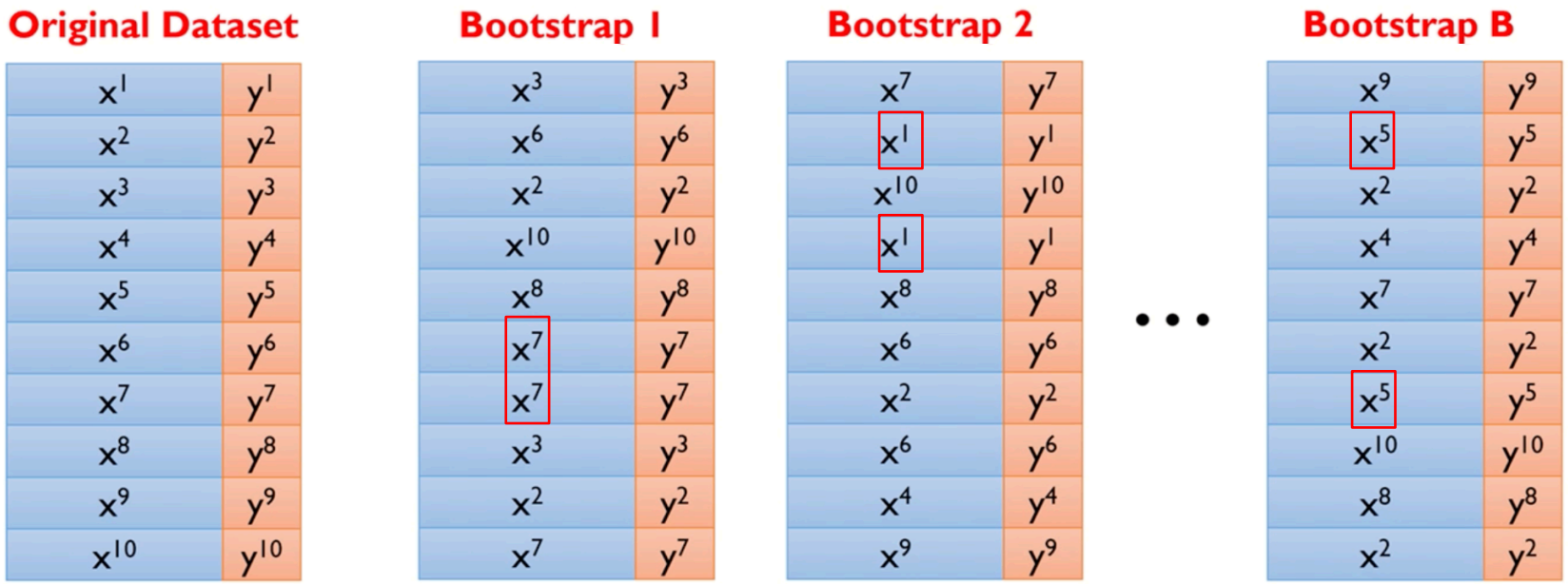
• • •

Bootstrap B

x^9	y^9
x^5	y^5
x^2	y^2
x^4	y^4
x^7	y^7
x^2	y^2
x^5	y^5
x^{10}	y^{10}
x^8	y^8
x^2	y^2

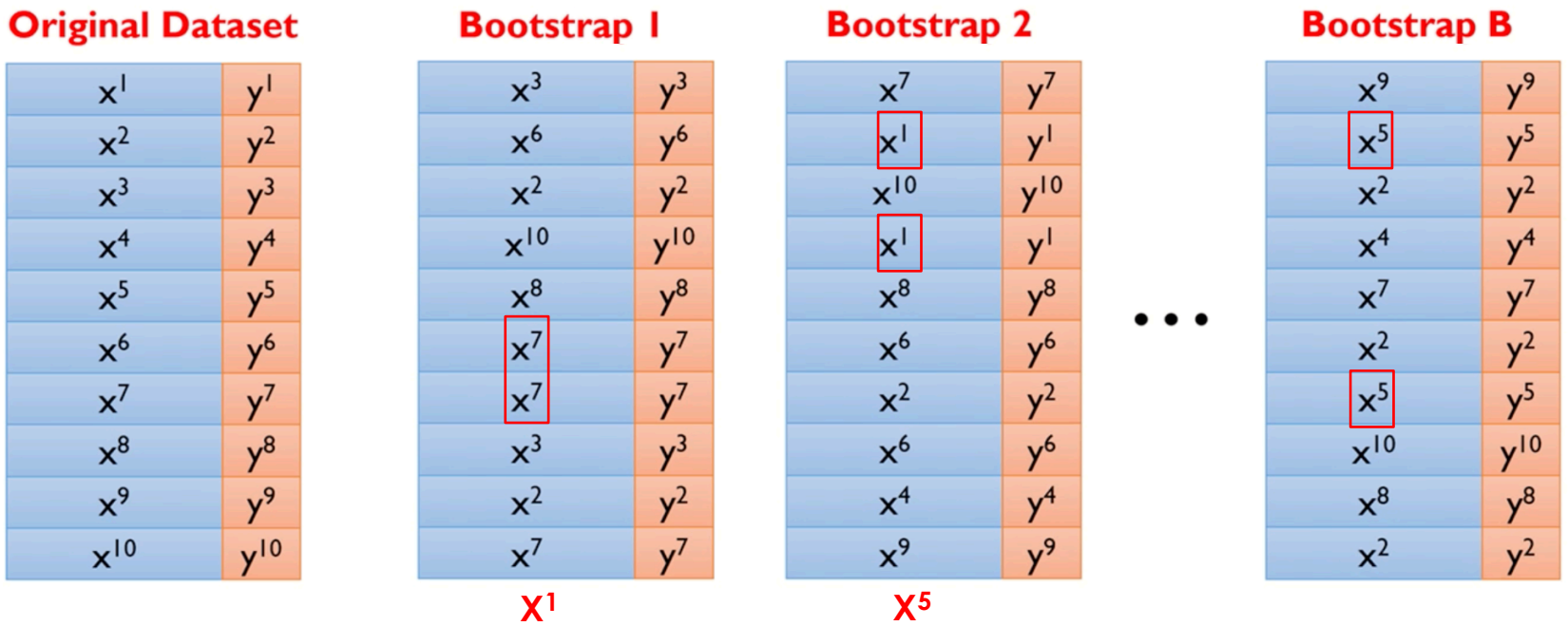
Random Forrest Concept - Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.



Random Forrest Concept – Bootstrapping

- 각가의 Decision Tree 서로 다른 데이터 셋을 사용
- 각 데이타 셋은 복원 추출을 통해 원래 데이타셋 만큼의 크기를 갖는다.

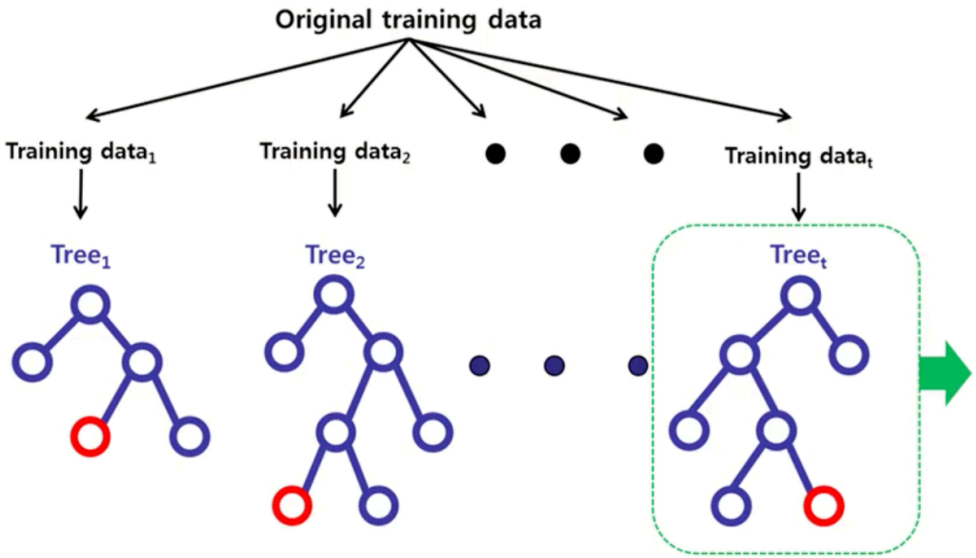


Random Forrest Concept – Bootstrapping

- 이론적으로 하나의 Feature가 한번도 선택 되지 않을 확률

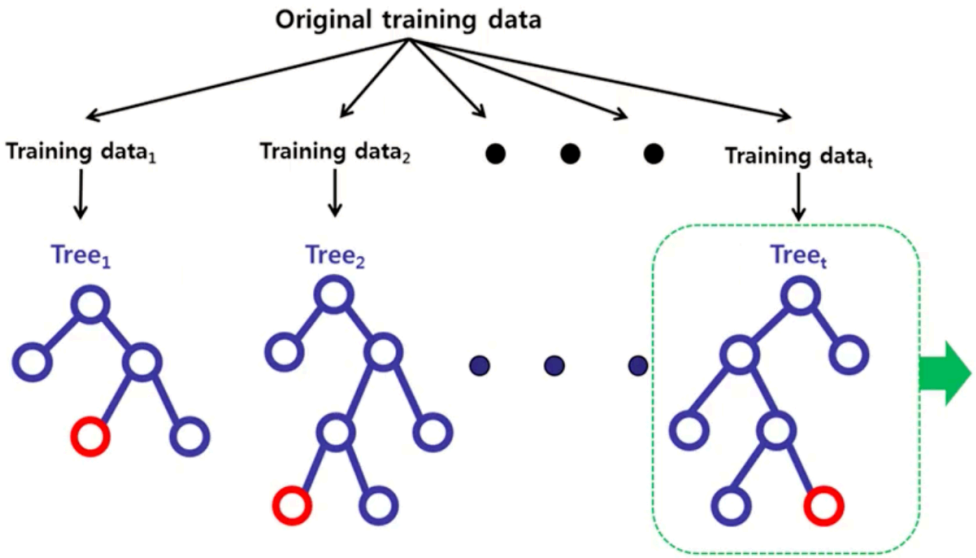
$$p = \left(1 - \frac{1}{N}\right)^N \rightarrow \lim_{N \to \infty} \left(1 - \frac{1}{N}\right)^N = e^{-1} = 0.368$$

Random Forrest Concept – Random Subspace



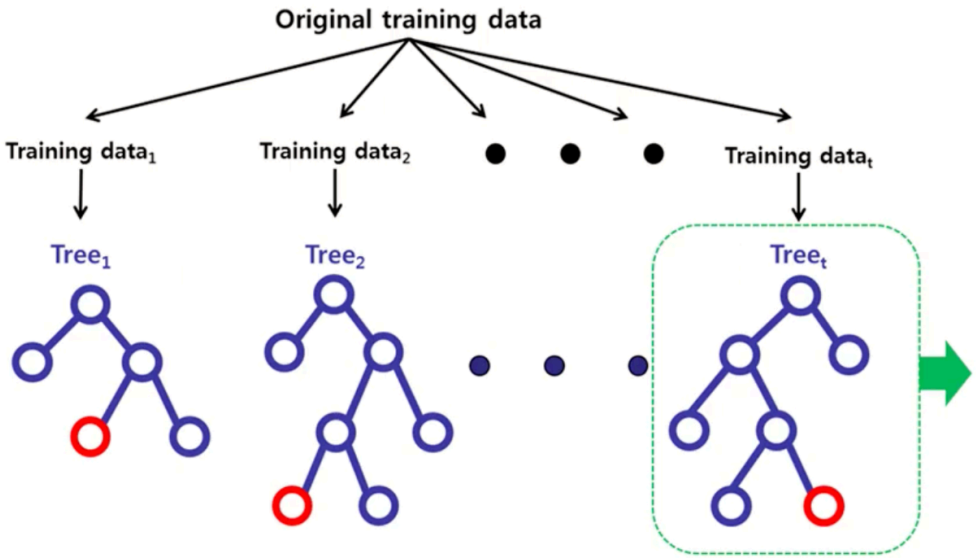
Tree

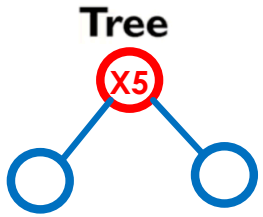
Random Forrest Concept – Random Subspace



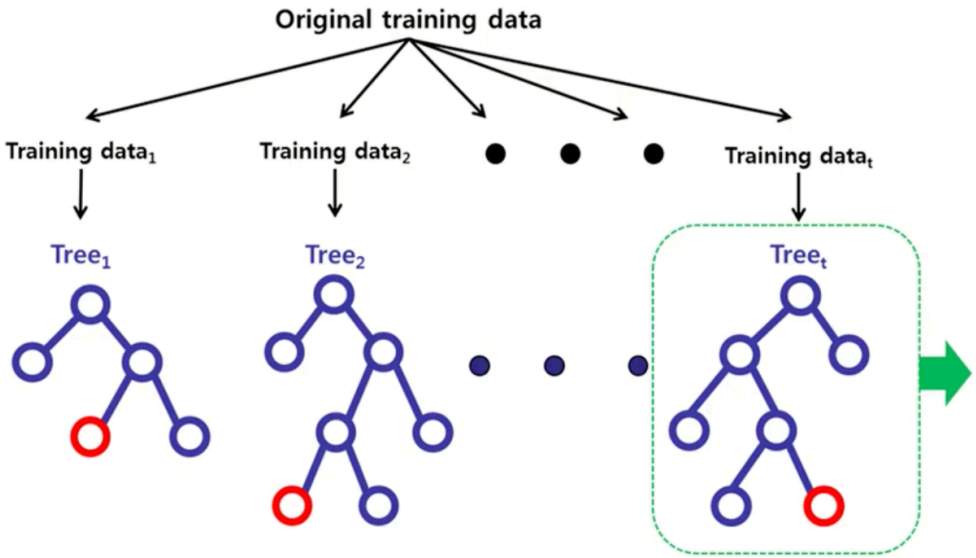
Tree

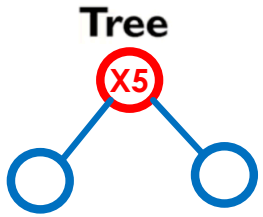
Random Forrest Concept – Random Subspace



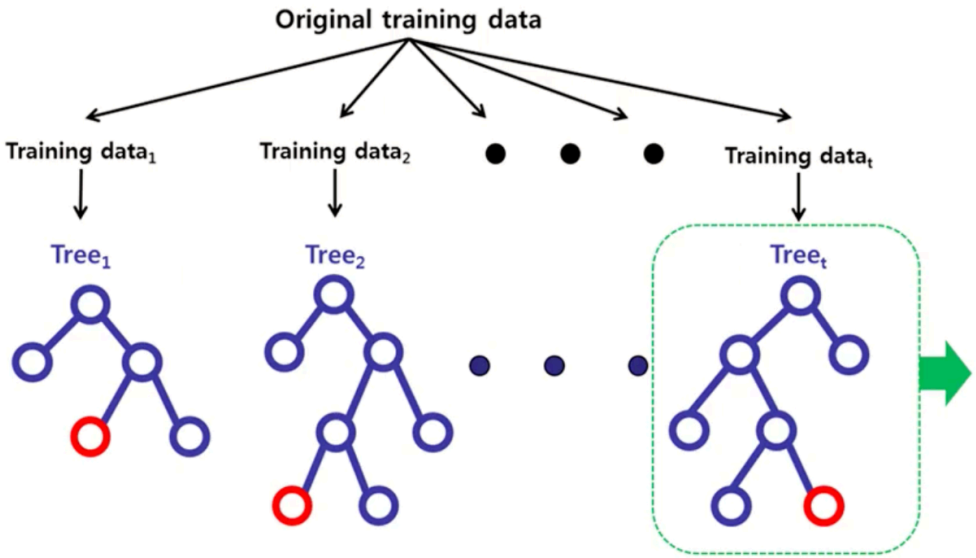


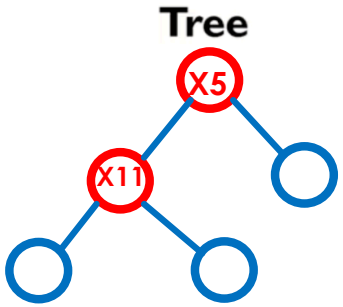
Random Forrest Concept – Random Subspace



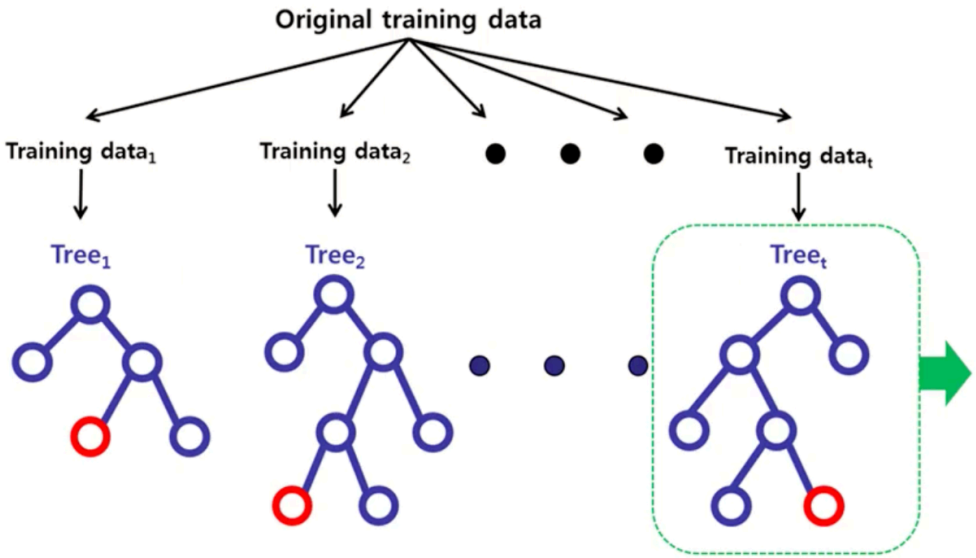


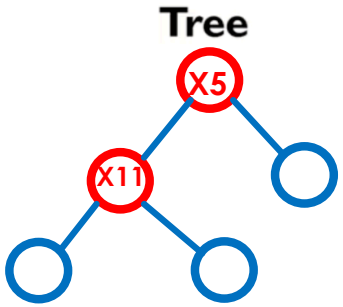
Random Forrest Concept – Random Subspace



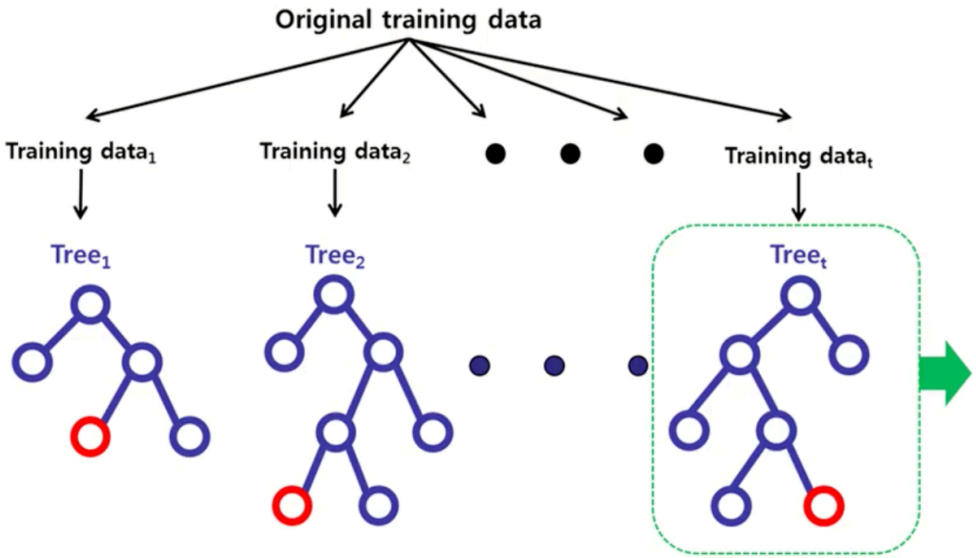


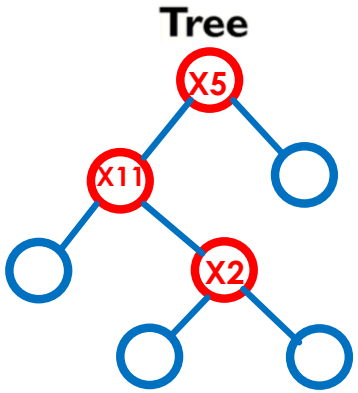
Random Forrest Concept – Random Subspace



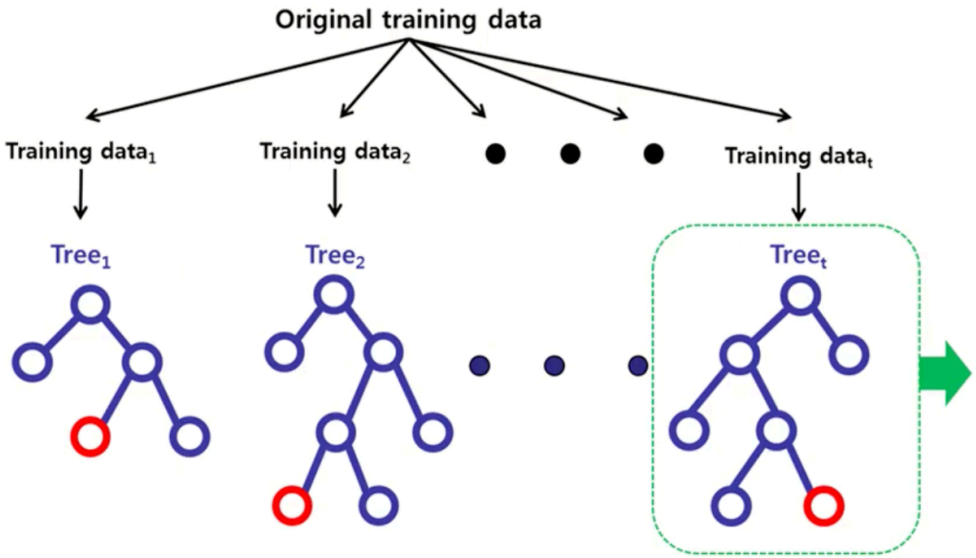


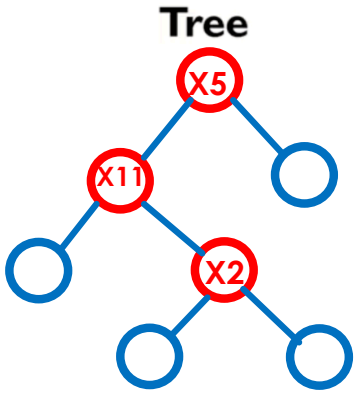
Random Forrest Concept – Random Subspace





Random Forrest Concept – Random Subspace





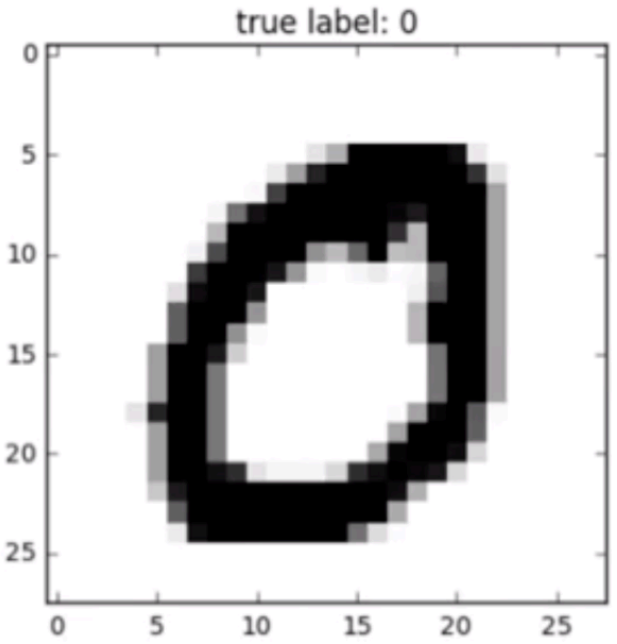
원래 변수	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16
		X2			X5						X11					

Random Forrest Concept – Aggregating (Ensemble)

Random Forrest Concept – Aggregating

- Soft Voting : rkr

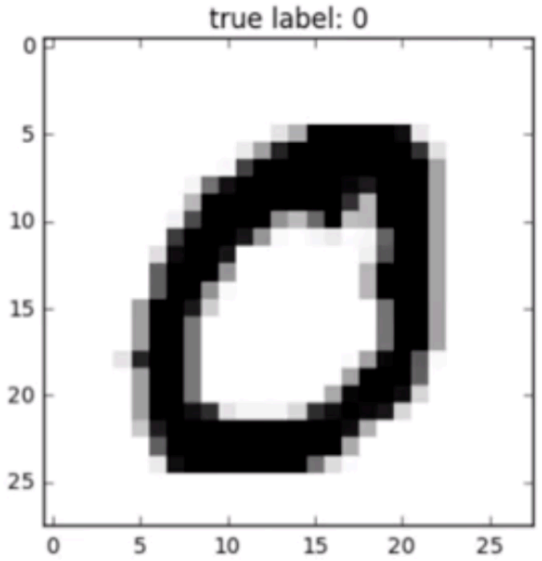
Tree Model	Prediction
Tree1	0
Tree2	0
Tree3	9
Tree4	0
Tree5	0



Random Forrest Concept – Aggregating

- Hard Voting : 각각의 모델들이 가장 많은 투표를 한 결과를 사용

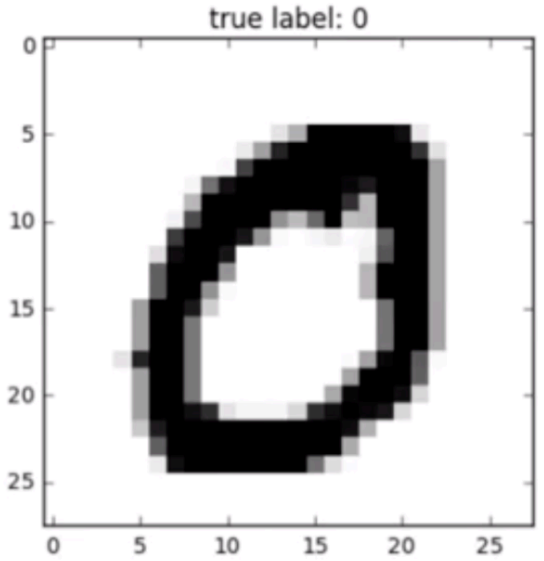
Tree	0	1	2	3	4	5	6	7	8	9
Tree1	0.9	0	0	0	0	0	0	0	0	0.1
Tree2	0.8	0	0	0	0	0	0	0	0	0.2
Tree3	0.3	0	0	0	0	0	0.1	0	0	0.6
Tree4	0.4	0	0	0	0	0	0.1	0	0	0.5



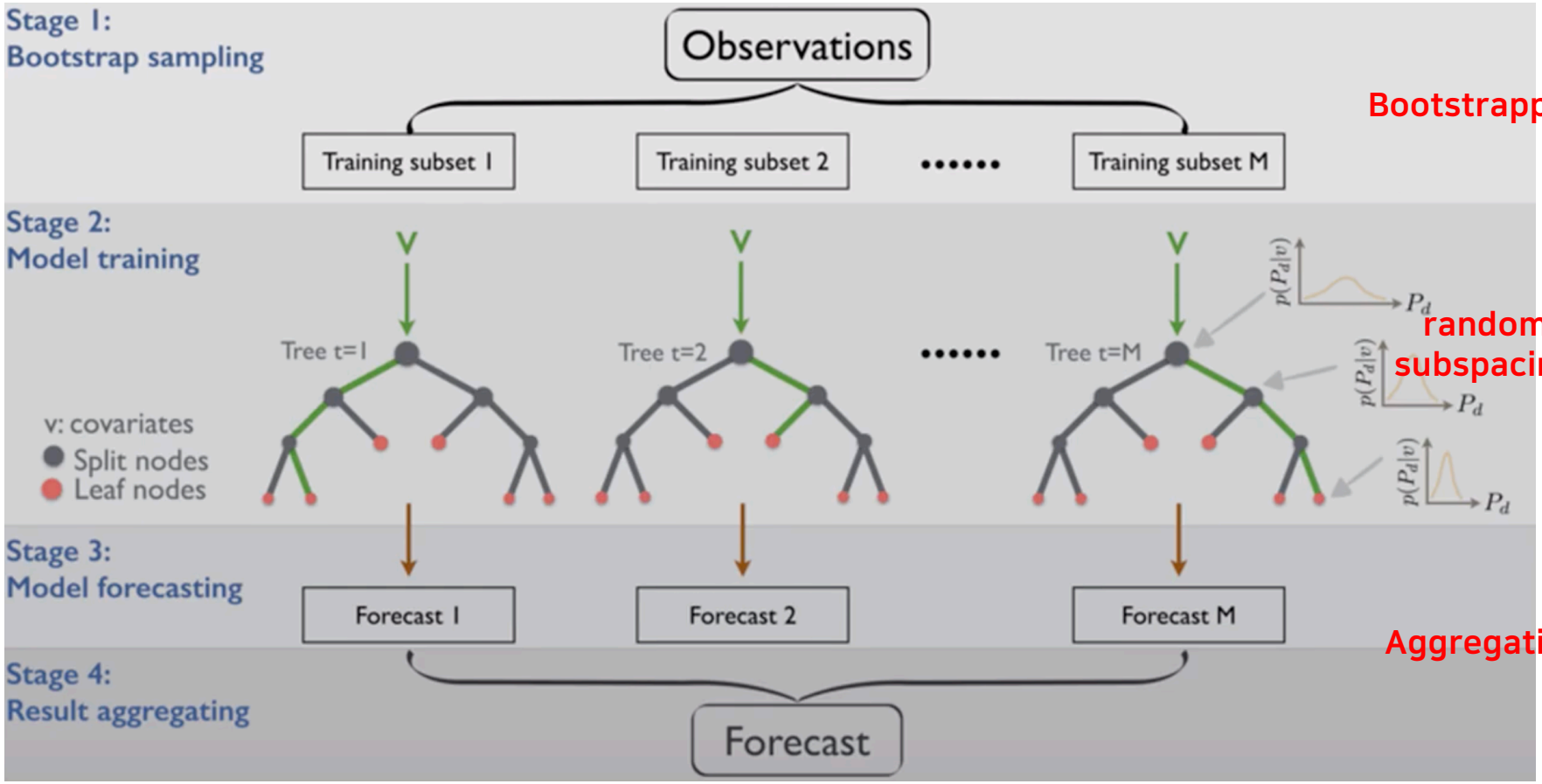
Random Forrest Concept – Aggregating

- Hard Voting : 각각의 모델들이 가장 많은 투표를 한 결과를 사용

Tree	0	1	2	3	4	5	6	7	8	9
Tree1	0.9	0	0	0	0	0	0	0	0	0.1
Tree2	0.8	0	0	0	0	0	0	0	0	0.2
Tree3	0.3	0	0	0	0	0	0.1	0	0	0.6
Tree4	0.4	0	0	0	0	0	0.1	0	0	0.5
투표 평균	**0.6**	**0**	**0**	**0**	**0**	**0**	**0.05**	**0**	**0**	**0.35**



Random Forrest Concept - Final

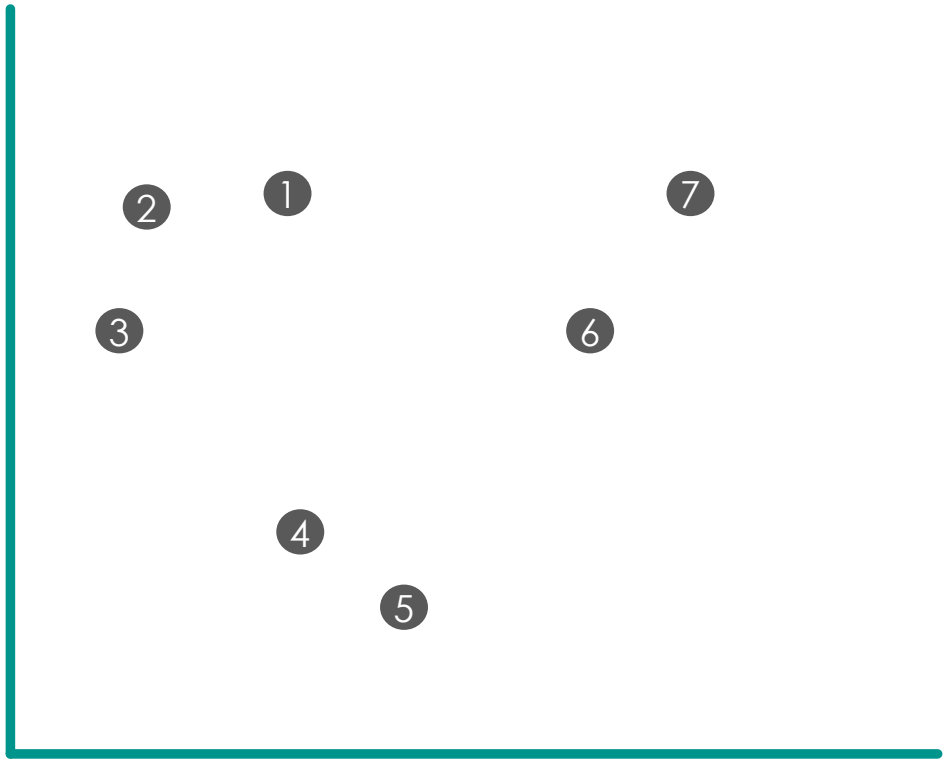


K-Means Clustering

K-Means Clustering

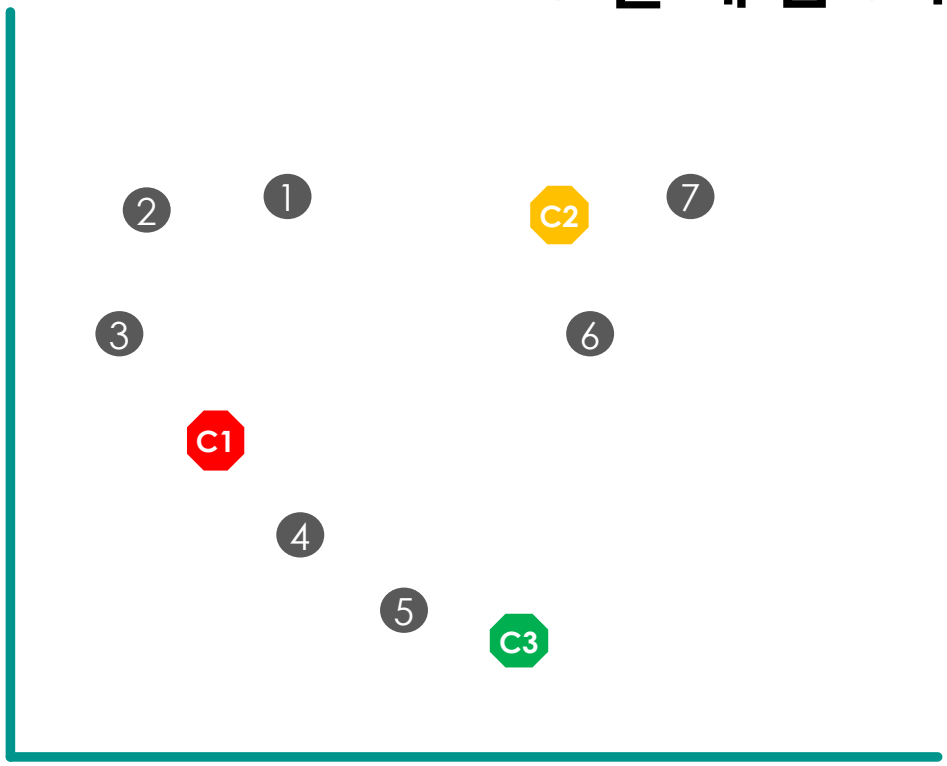
1. k 를 몇개로 할 것인지 정합니다.
2. 초기 클러스터의 중심을 설정 합니다. (Centeroid)
   2-1. 랜덤 선택
   2-2. Kmeans++ 선택
3. 설정한 중심에서 가장 가까운 점들은 구분해 클러스터를 만듭니다.
4. 각 군집의 평균으로 중심을 옮깁니다.

K-Means Clustering

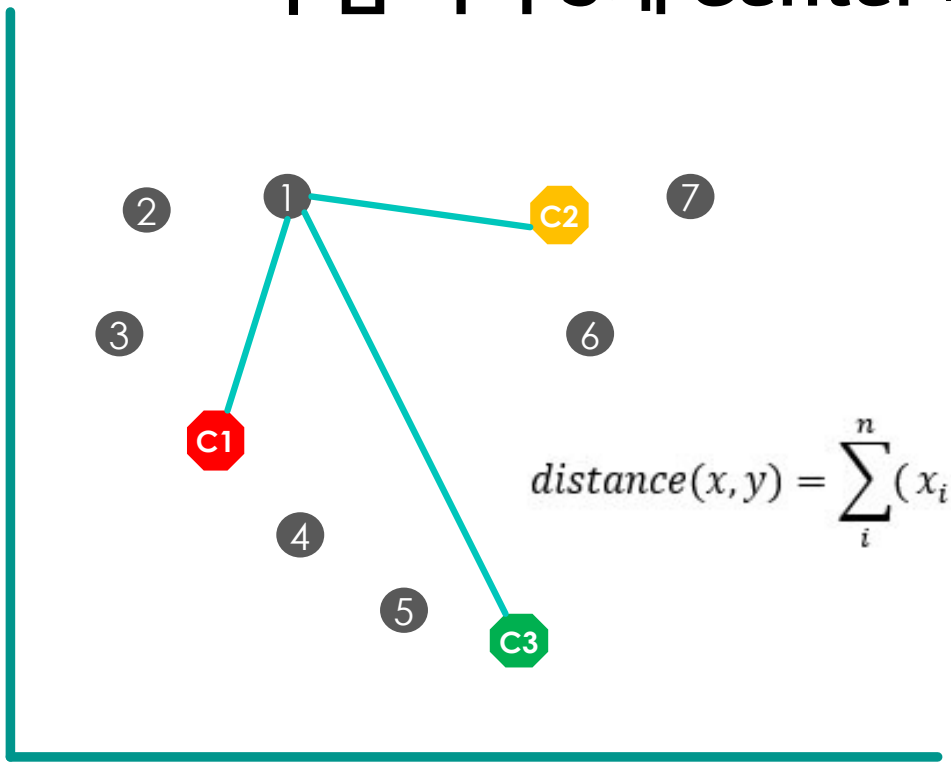


K-Means Clustering

k = 3 일 때 점 3개를 임의 로 잡음



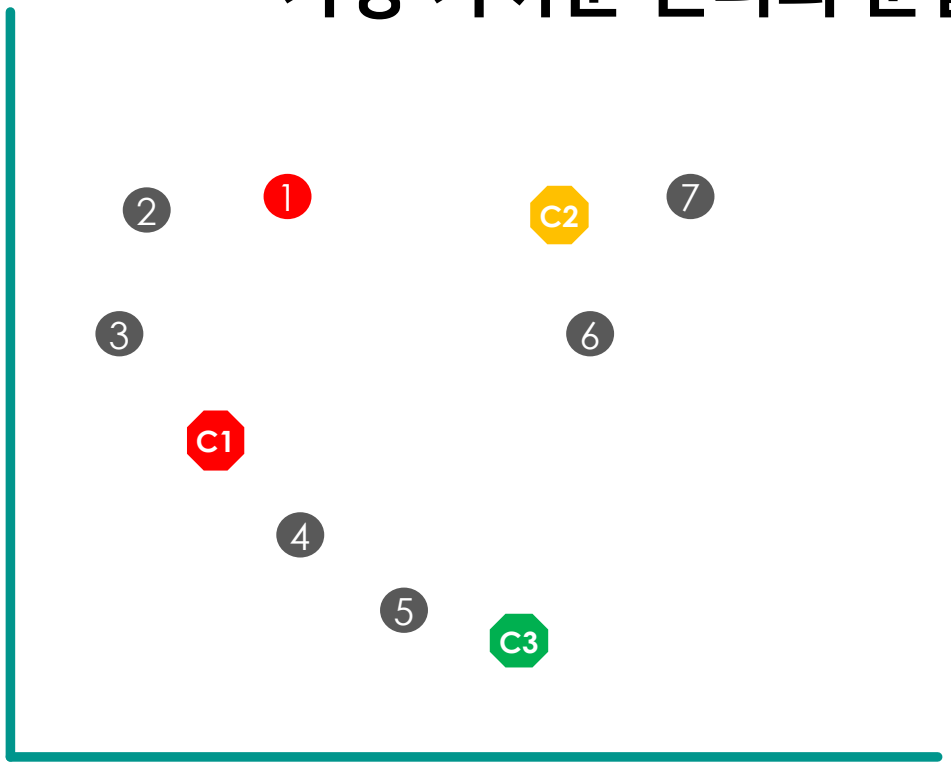
각 점 마다 3개 Center 와 의 거리 측정



$$distance(x,y) = \sum_{i}^{n} (x_i - y_i)^2$$

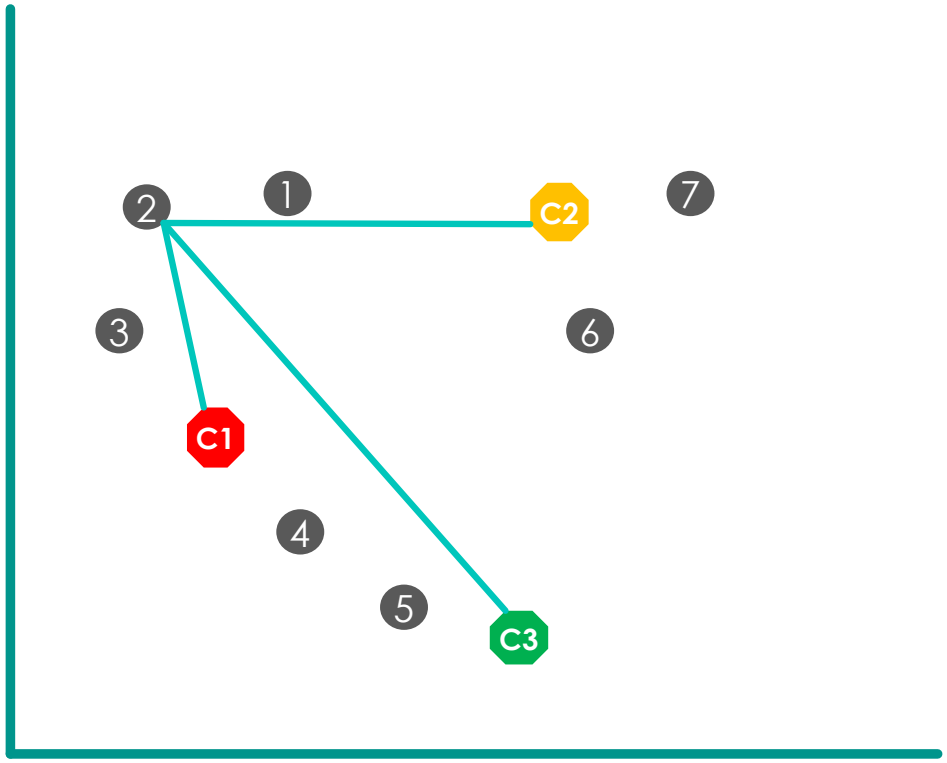
K-Means Clustering

가장 가까운 센터의 군집으로 분류



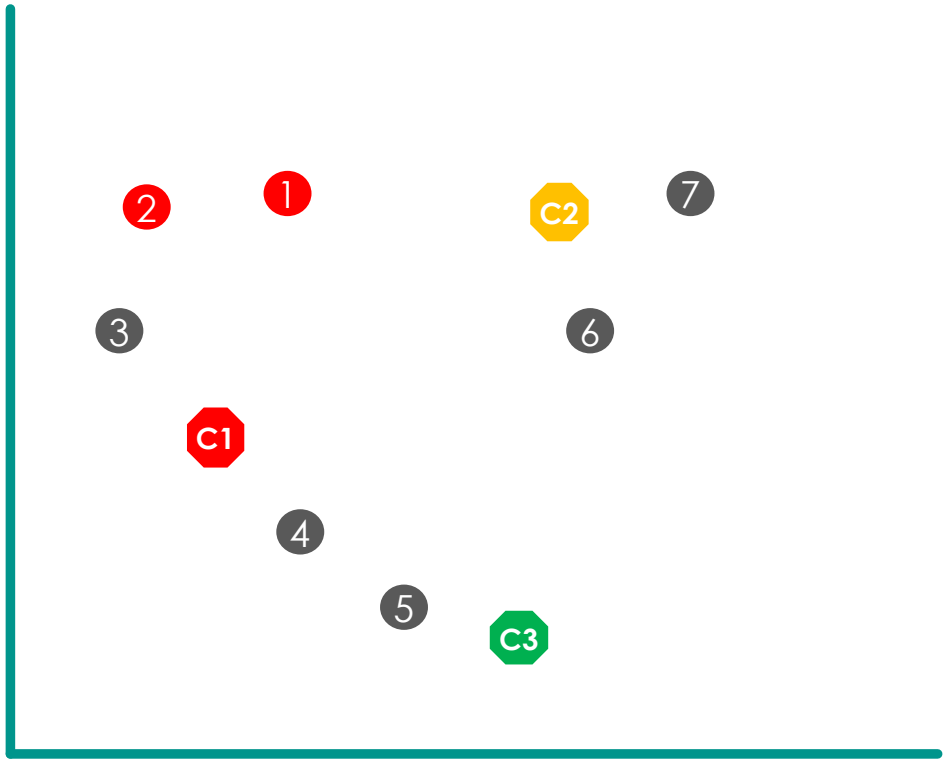
K-Means Clustering

반복



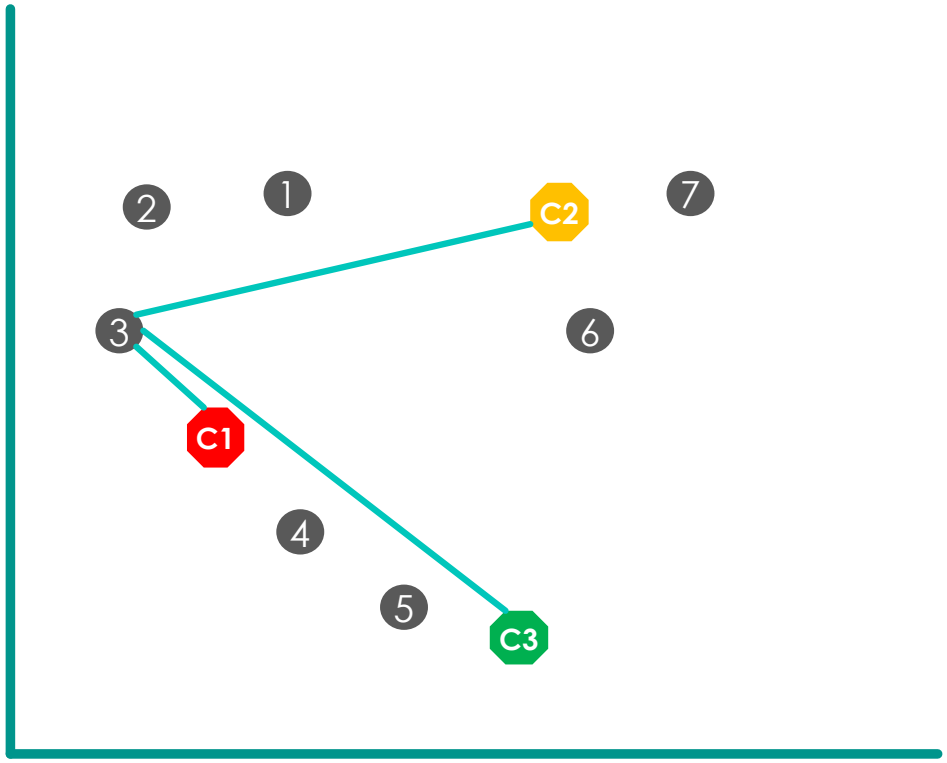
K-Means Clustering

반복



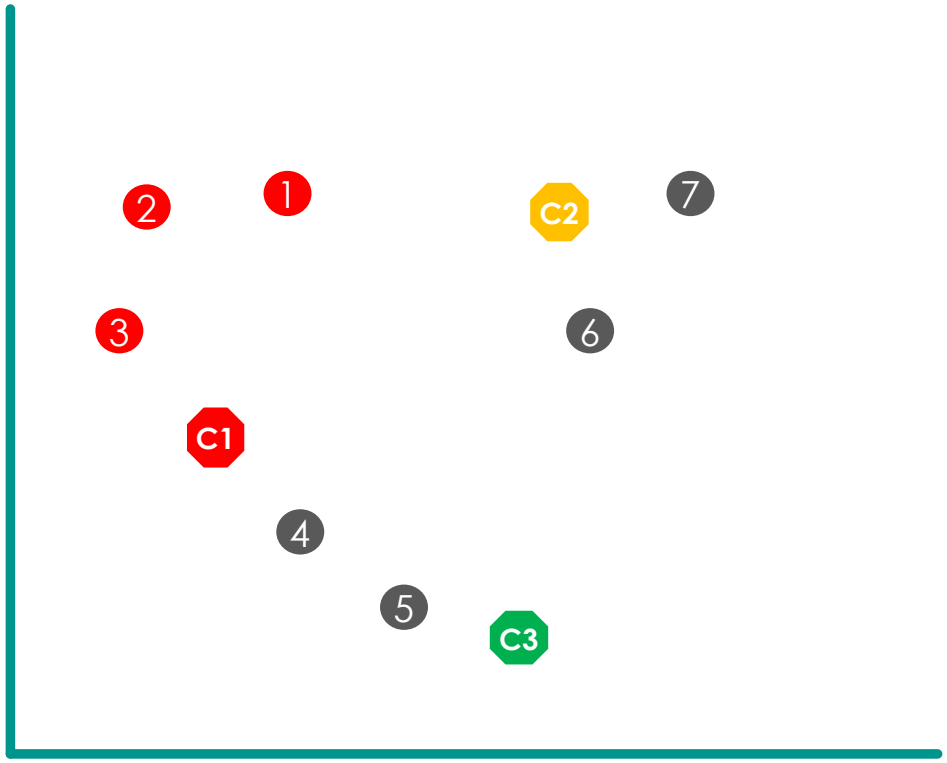
K-Means Clustering

반복



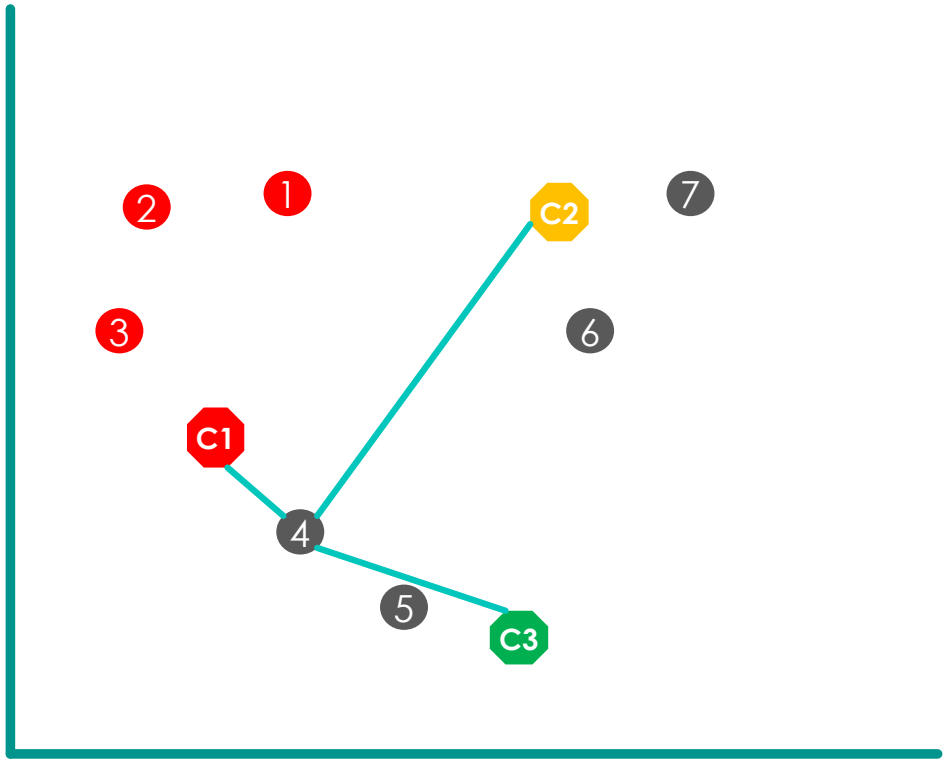
K-Means Clustering

반복



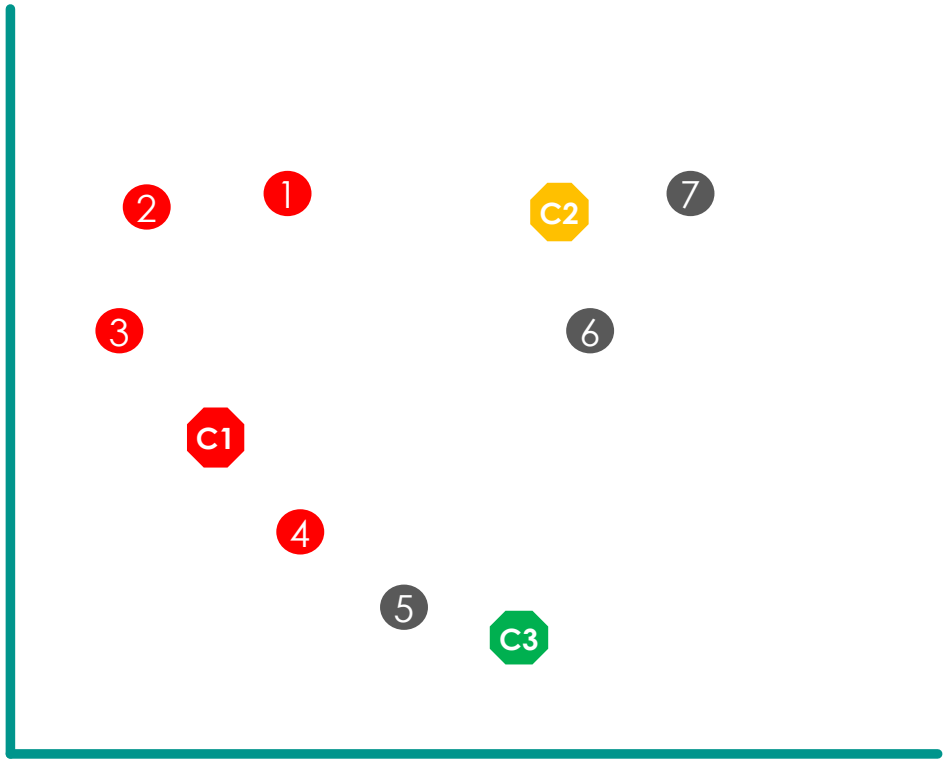
K-Means Clustering

반복



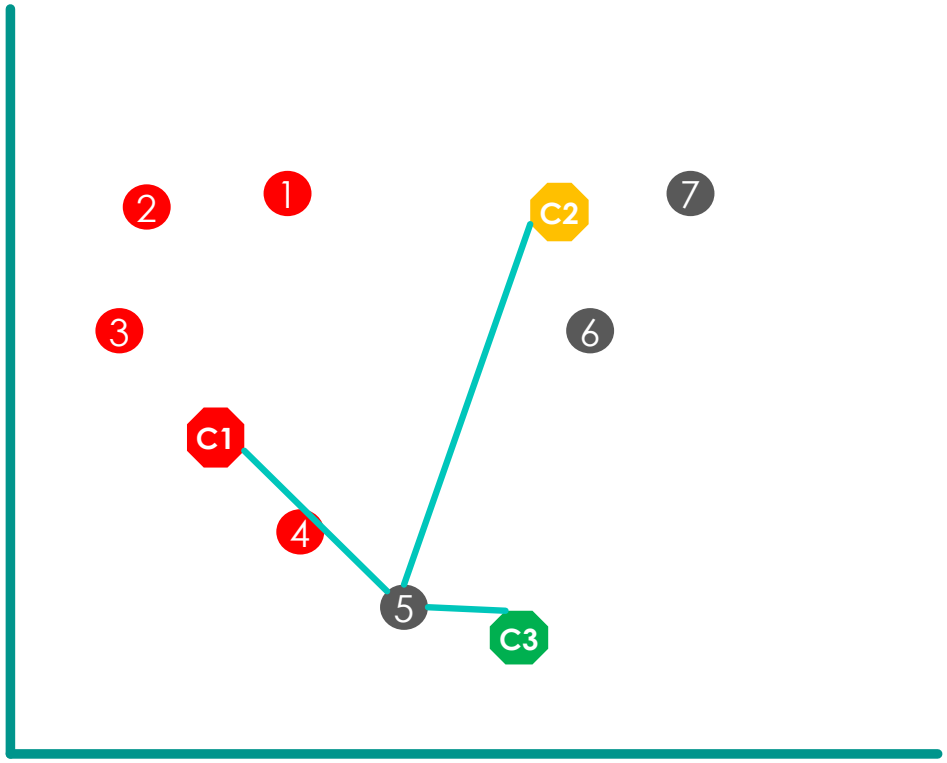
K-Means Clustering

반복



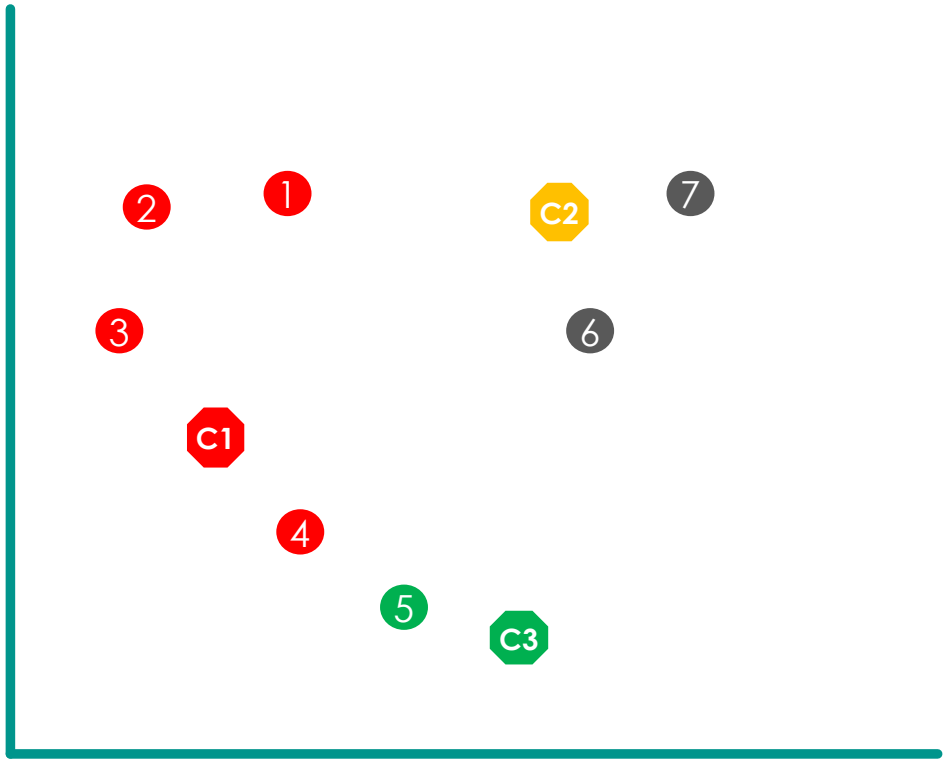
K-Means Clustering

반복



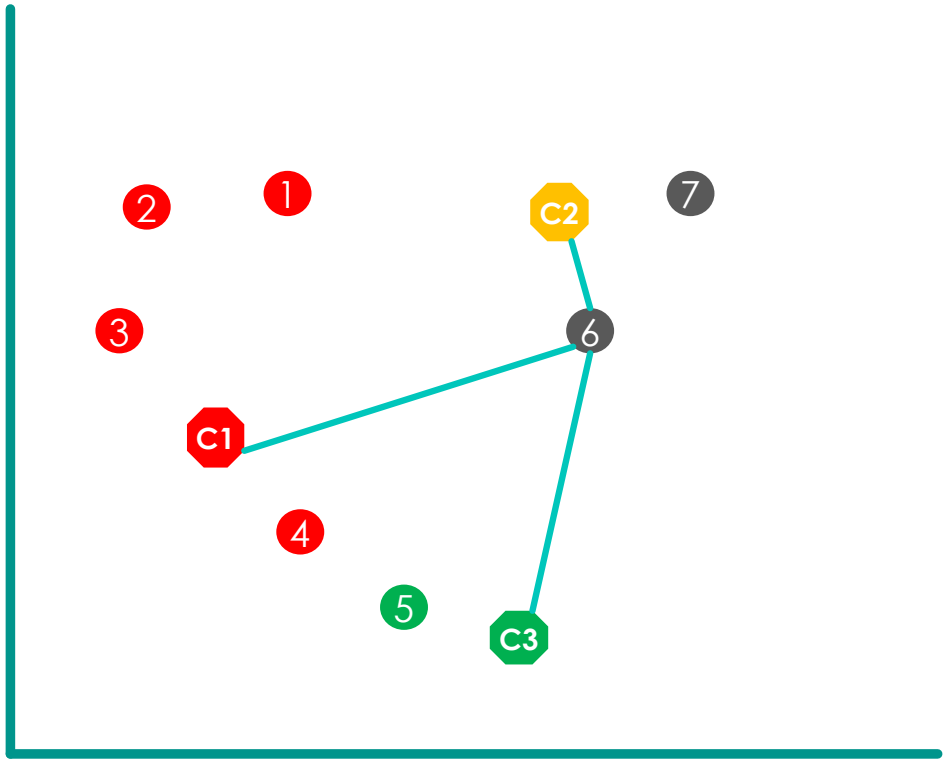
K-Means Clustering

반복



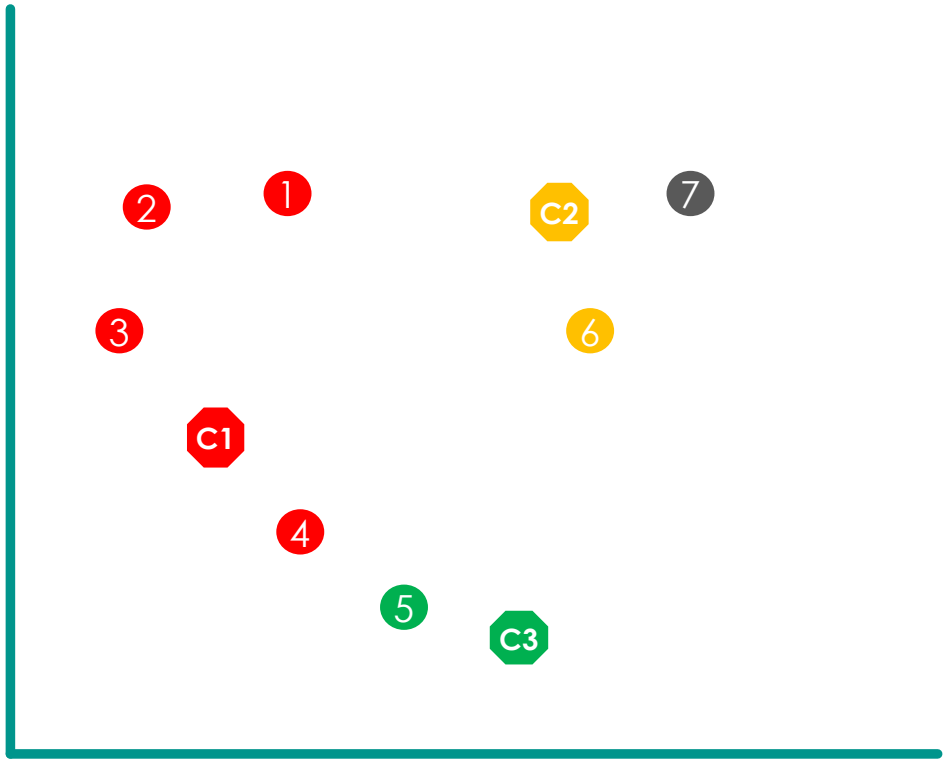
K-Means Clustering

반복



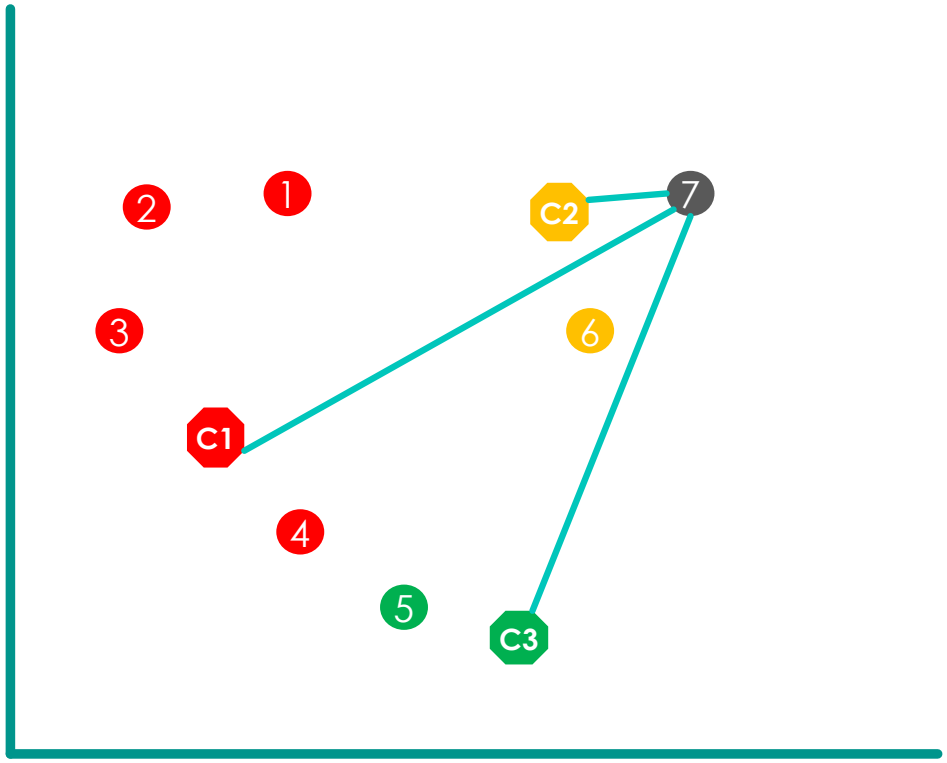
K-Means Clustering

반복



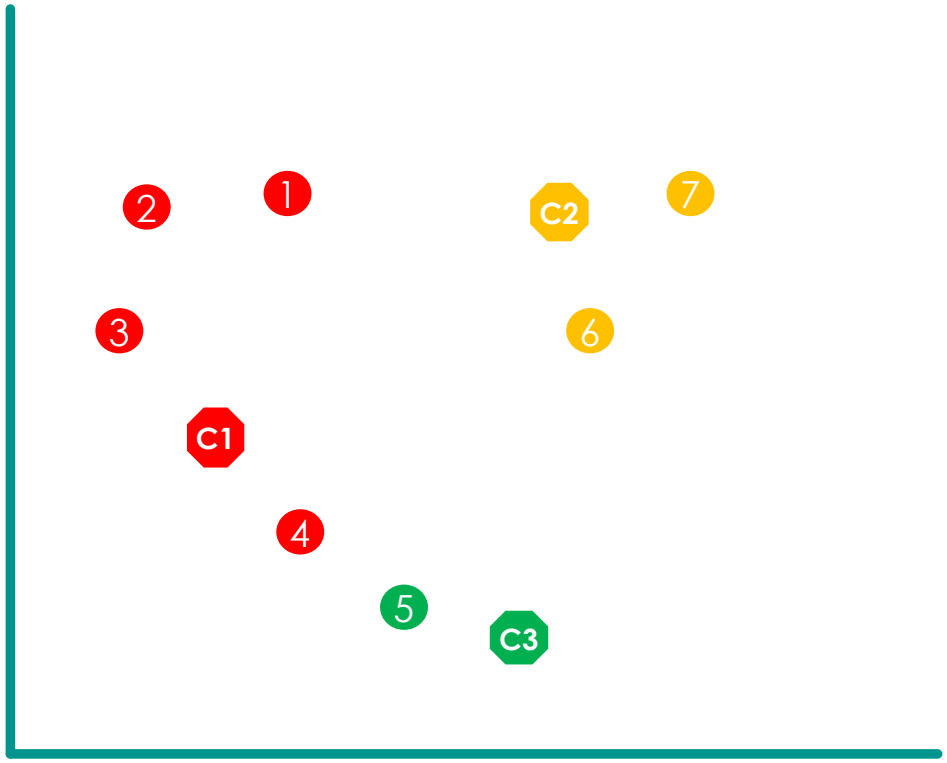
K-Means Clustering

반복

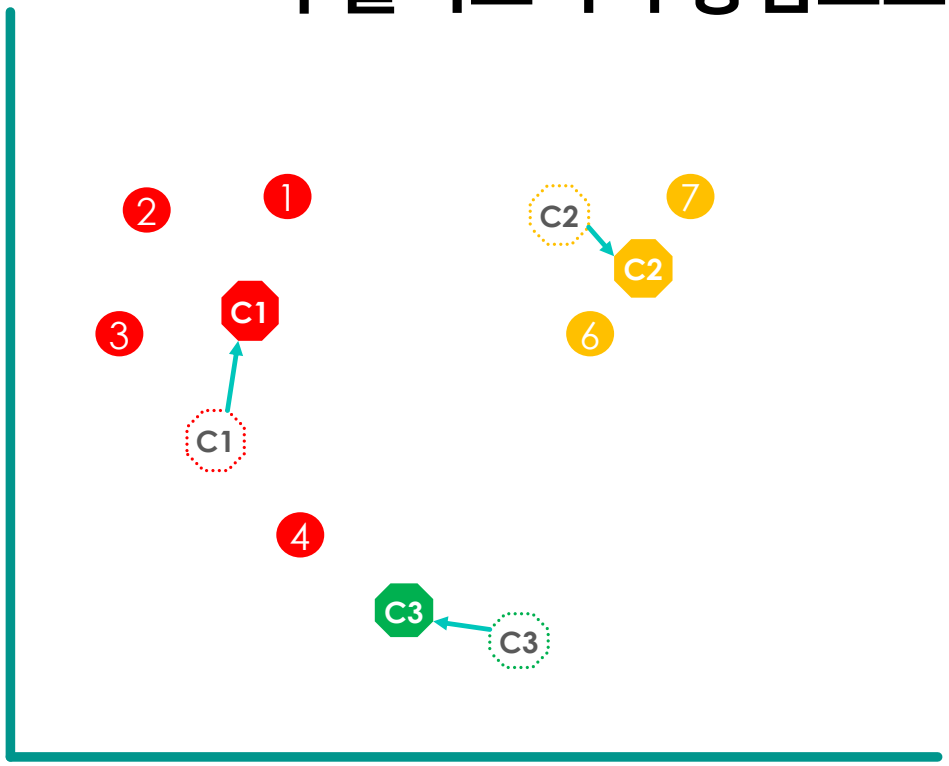


K-Means Clustering

반복

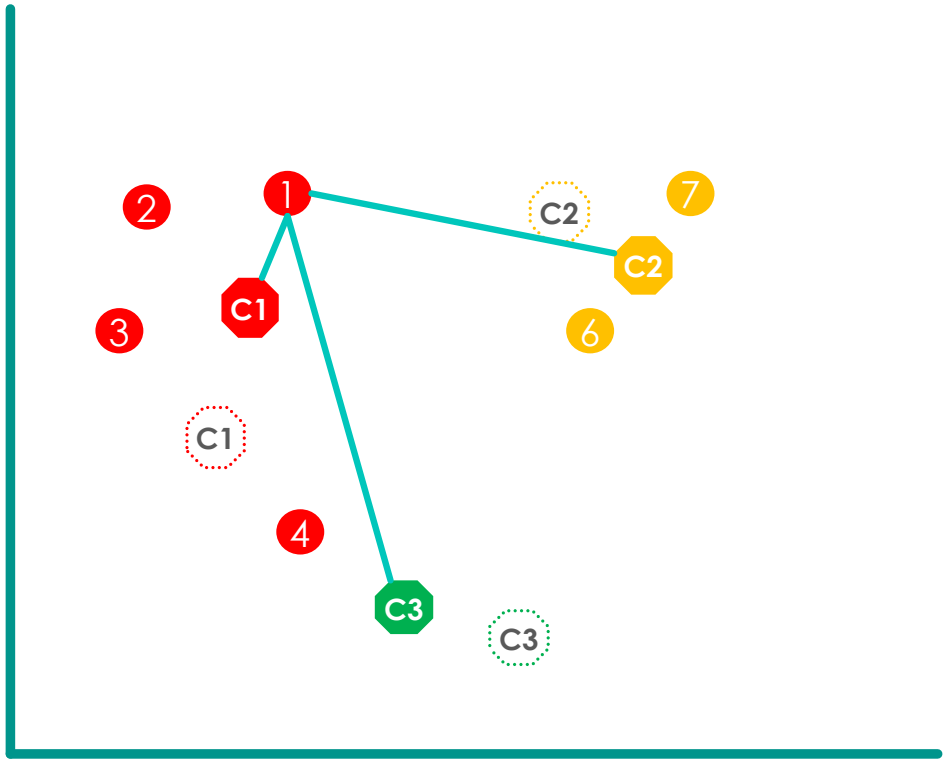


각 클러스터의 중심으로 Center를 옮김



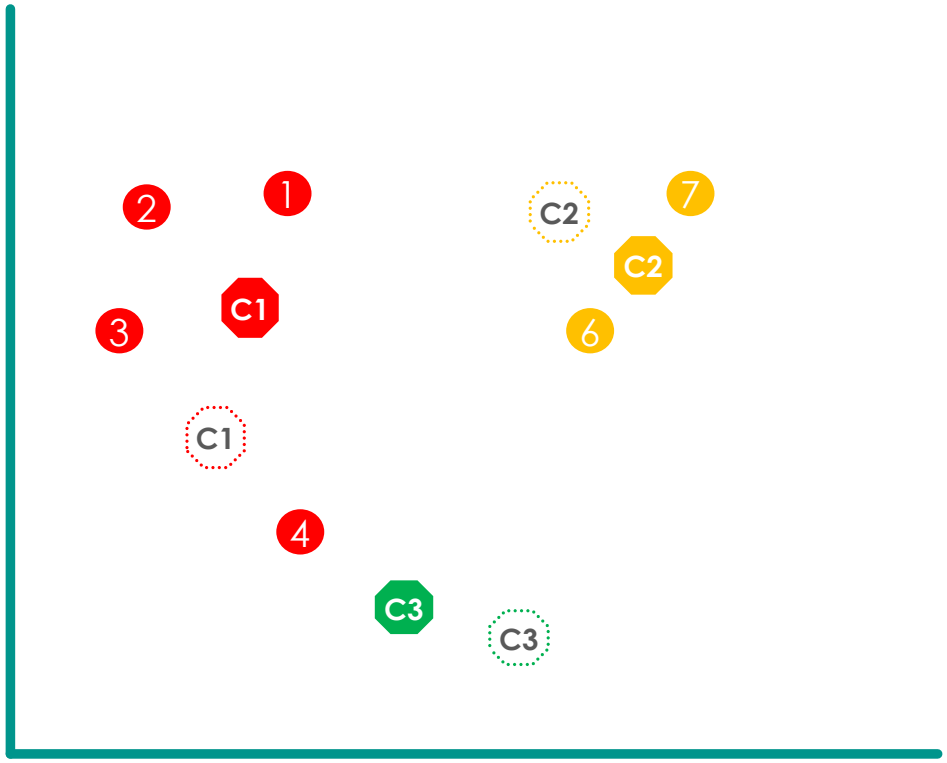
K-Means Clustering

반복



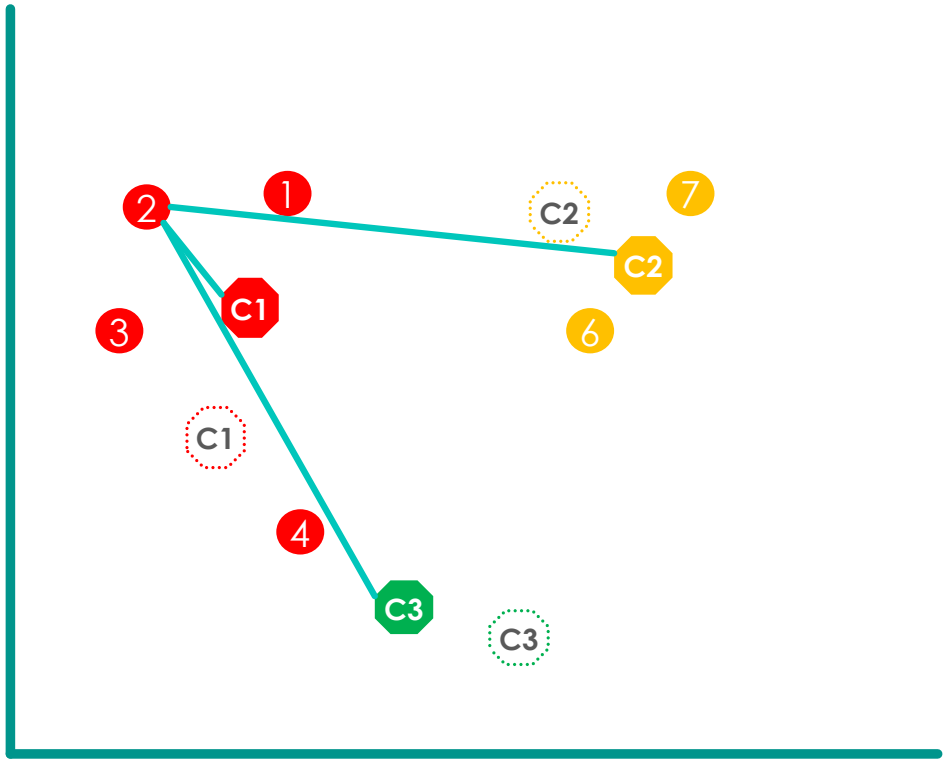
K-Means Clustering

반복



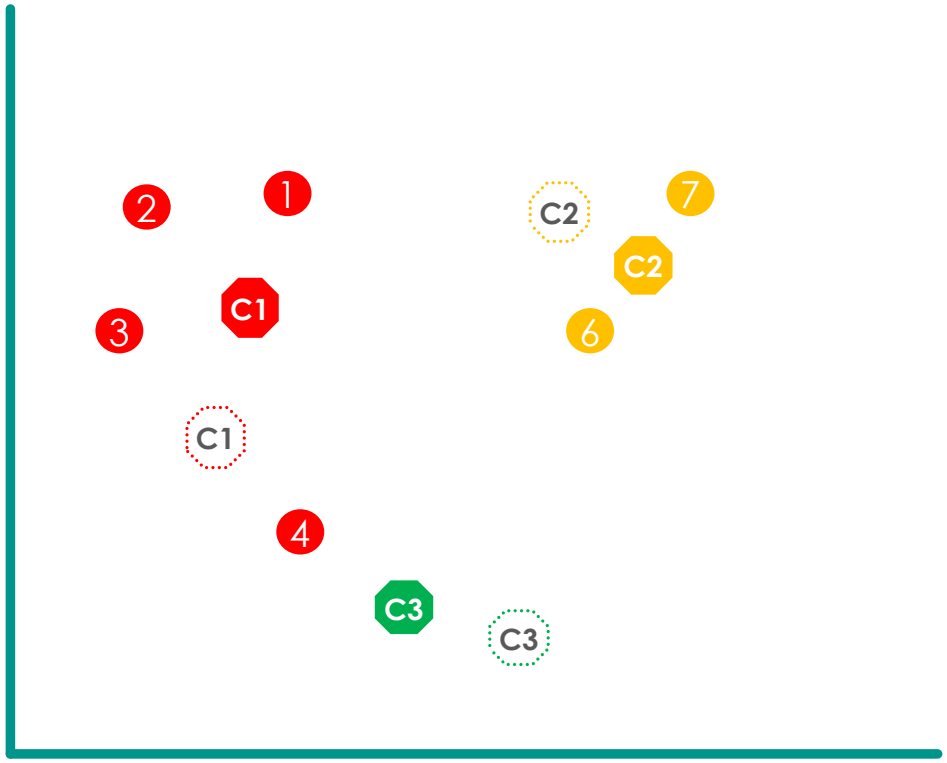
K-Means Clustering

반복



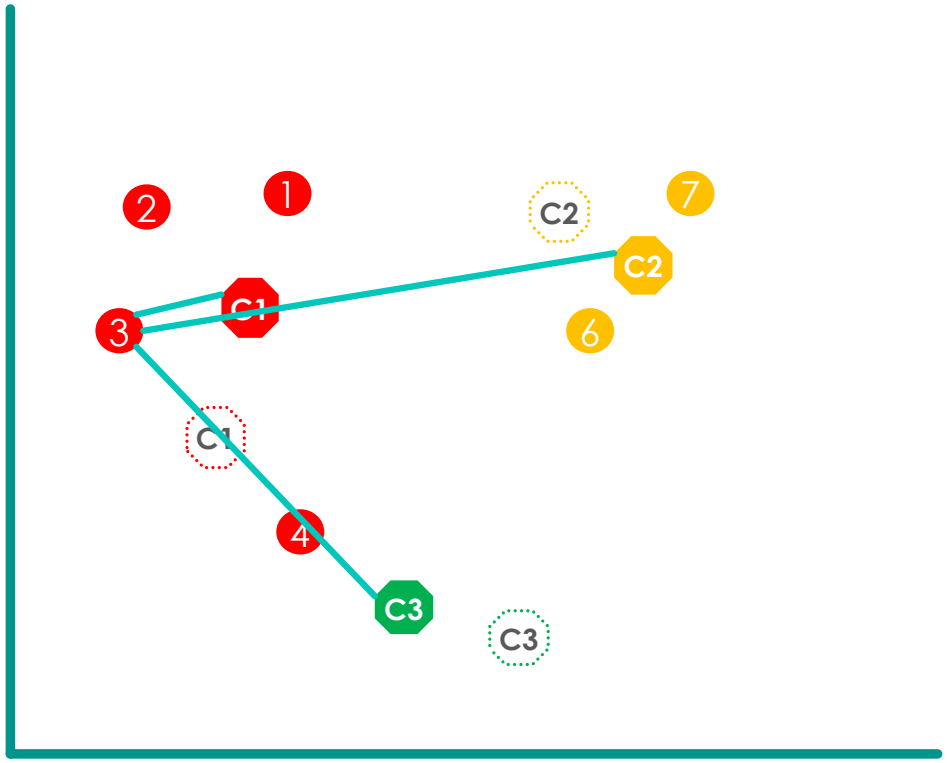
K-Means Clustering

반복



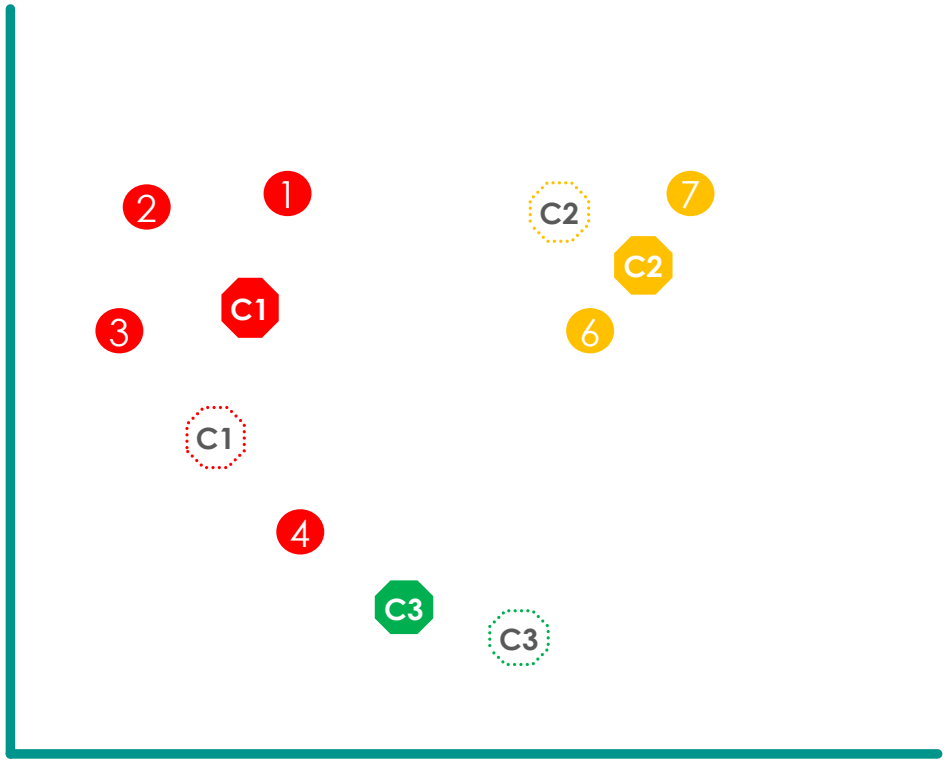
K-Means Clustering

반복

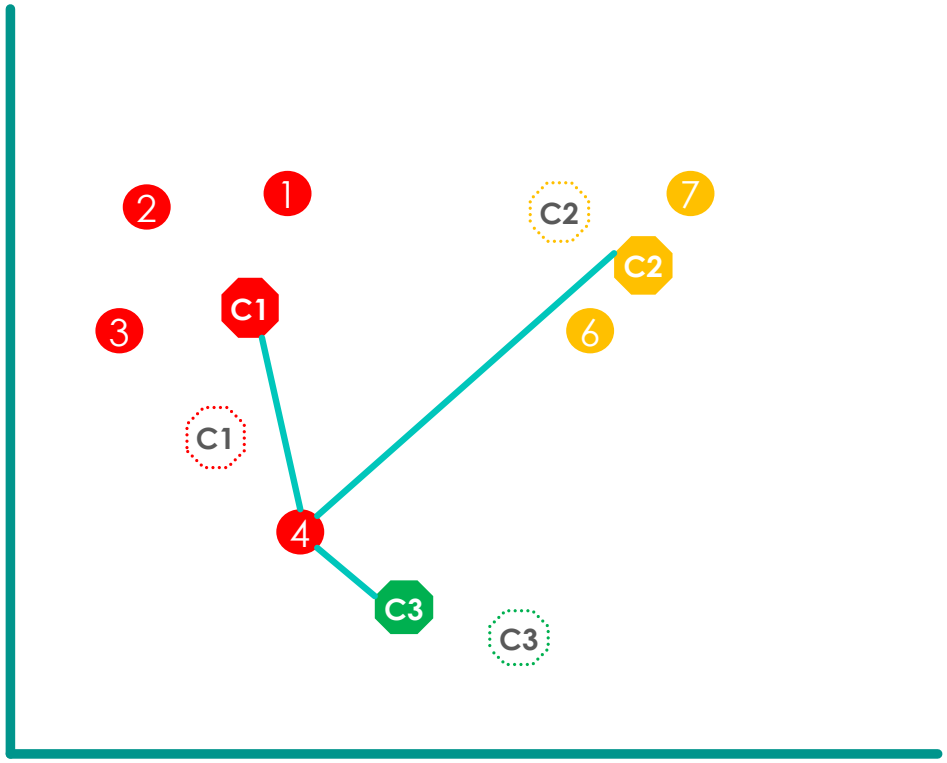


K-Means Clustering

반복

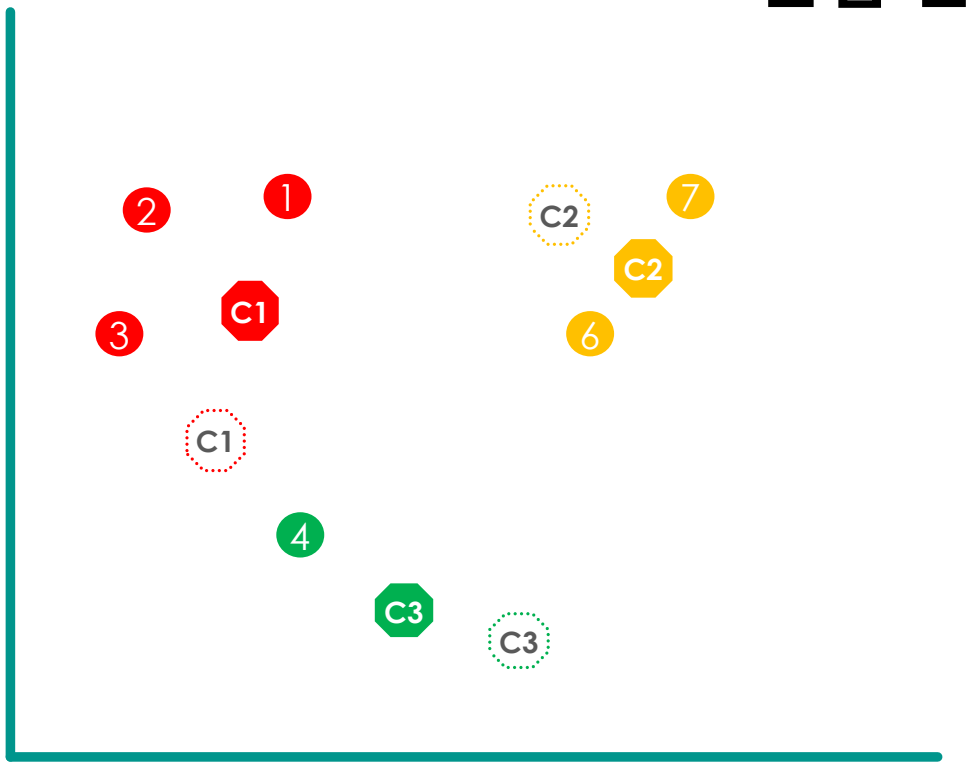


센터가 옮겨 지면서 새로운 군집에 가까운 점 생김

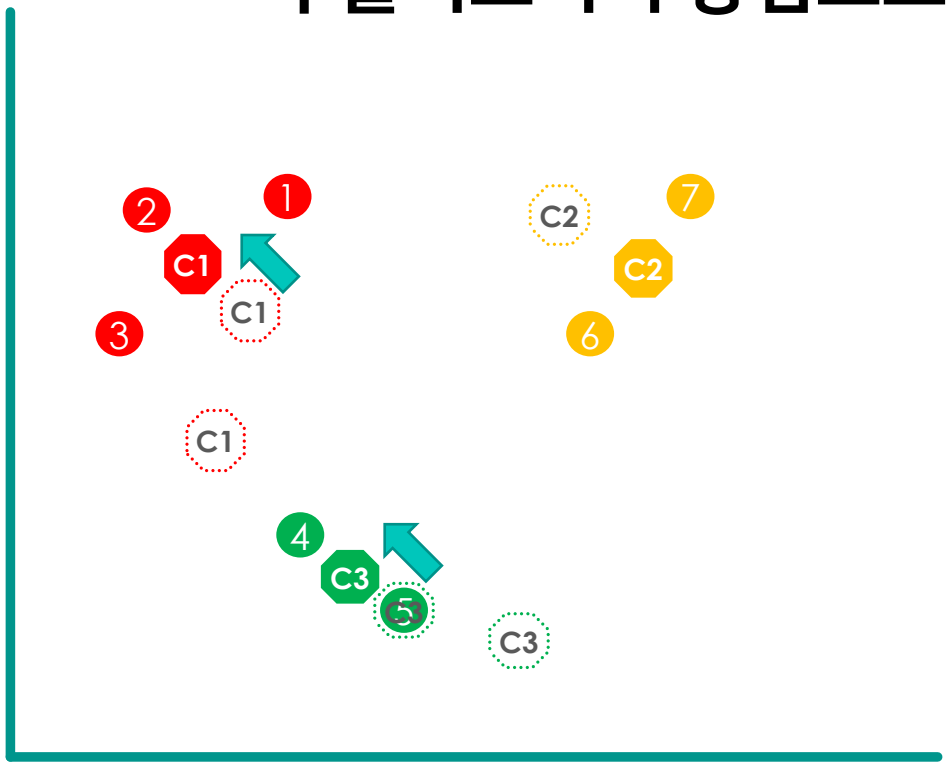


K-Means Clustering

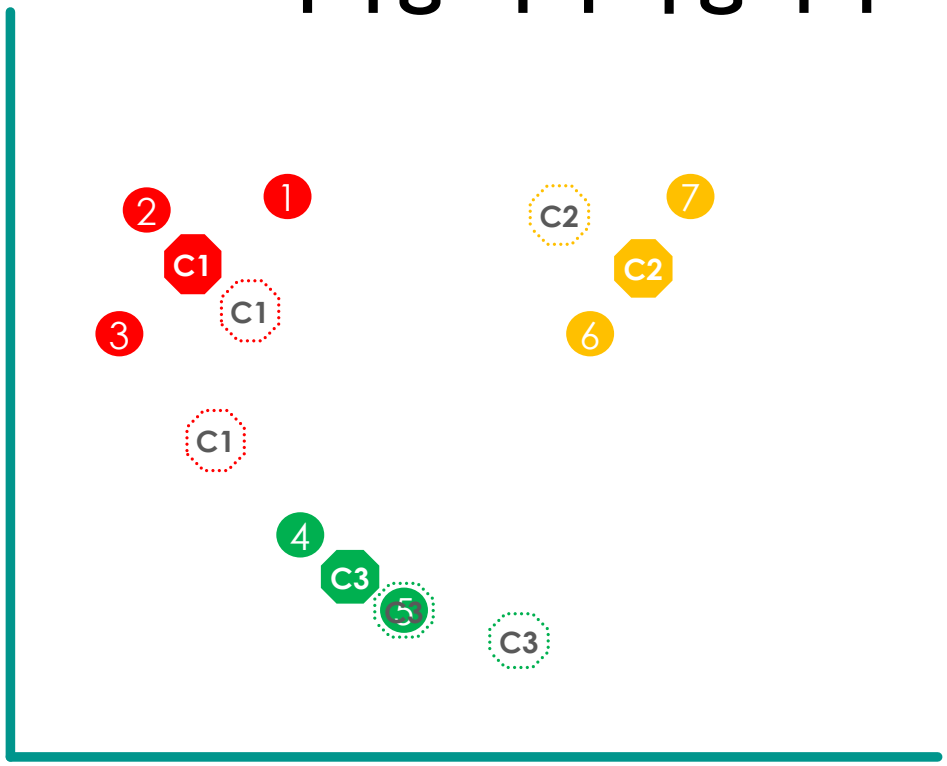
군집 변경 발생



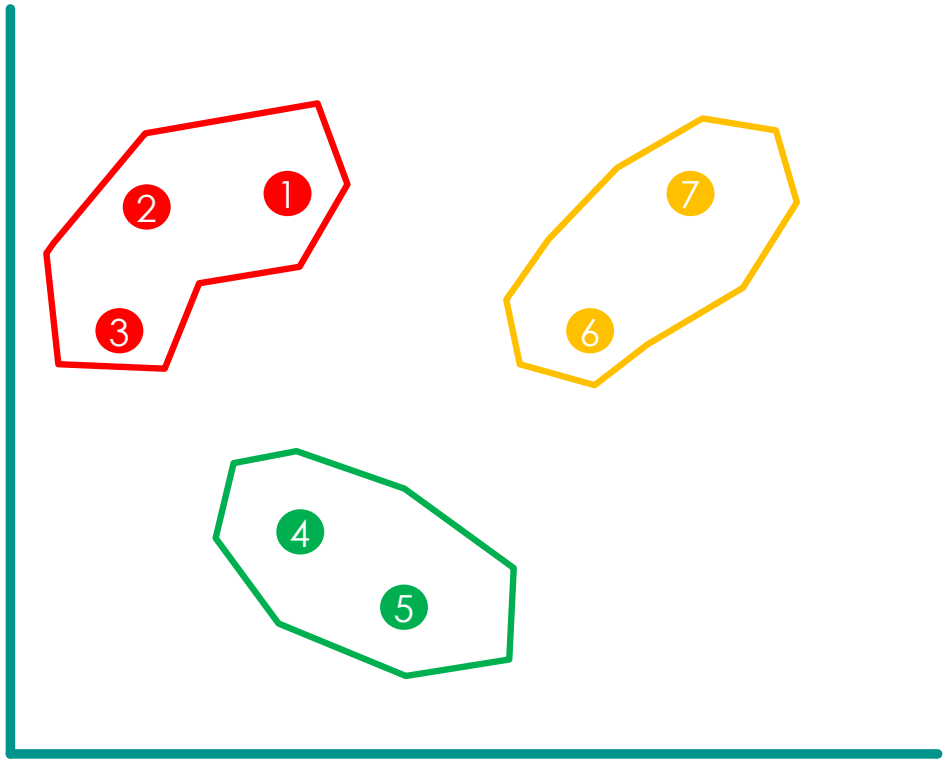
각 클러스터의 중심으로 Center를 옮김



더이상 거리 측정이 무의미 하면 중단

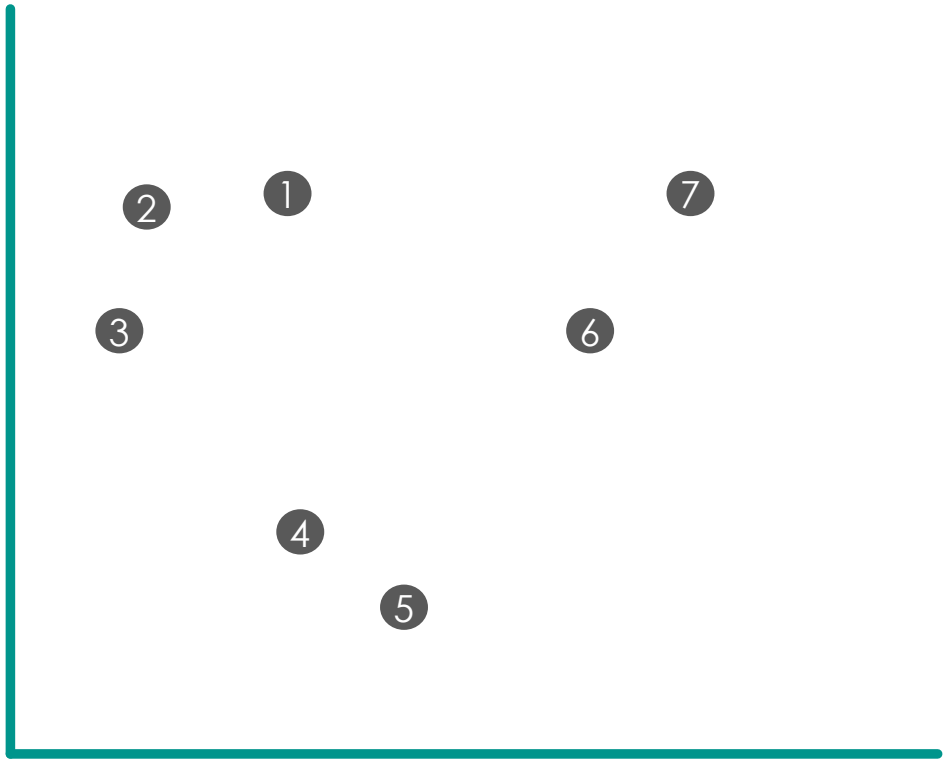


군집 결정



KMeans++

K-Means Clustering

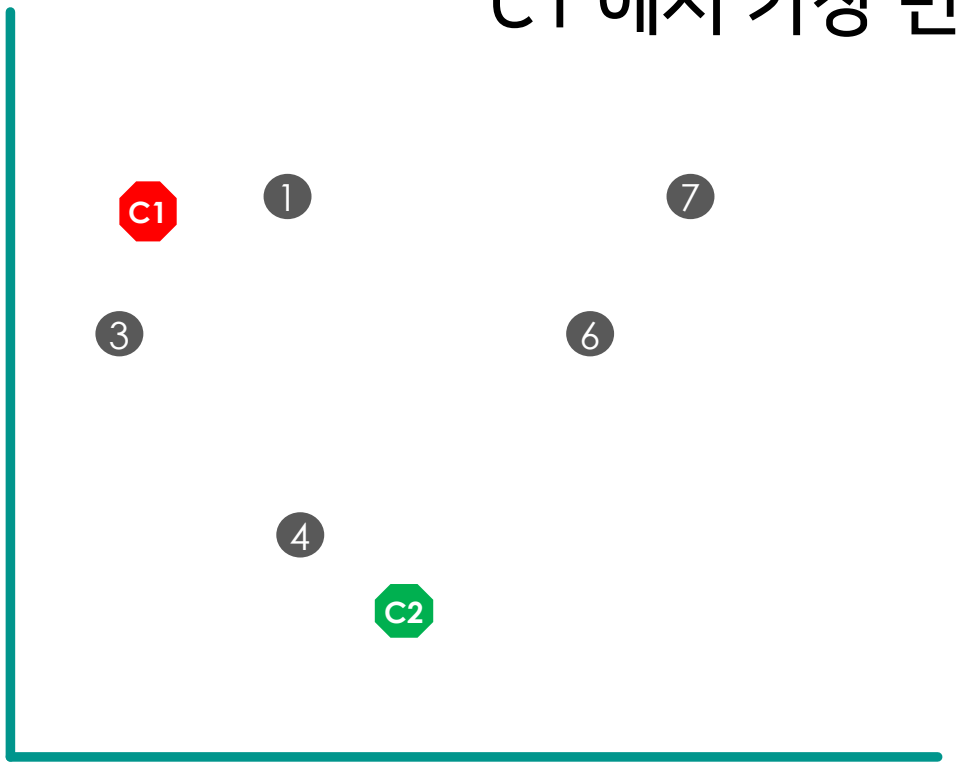


K-Means Clustering

임의의 점을 선택 해서 중심 설정

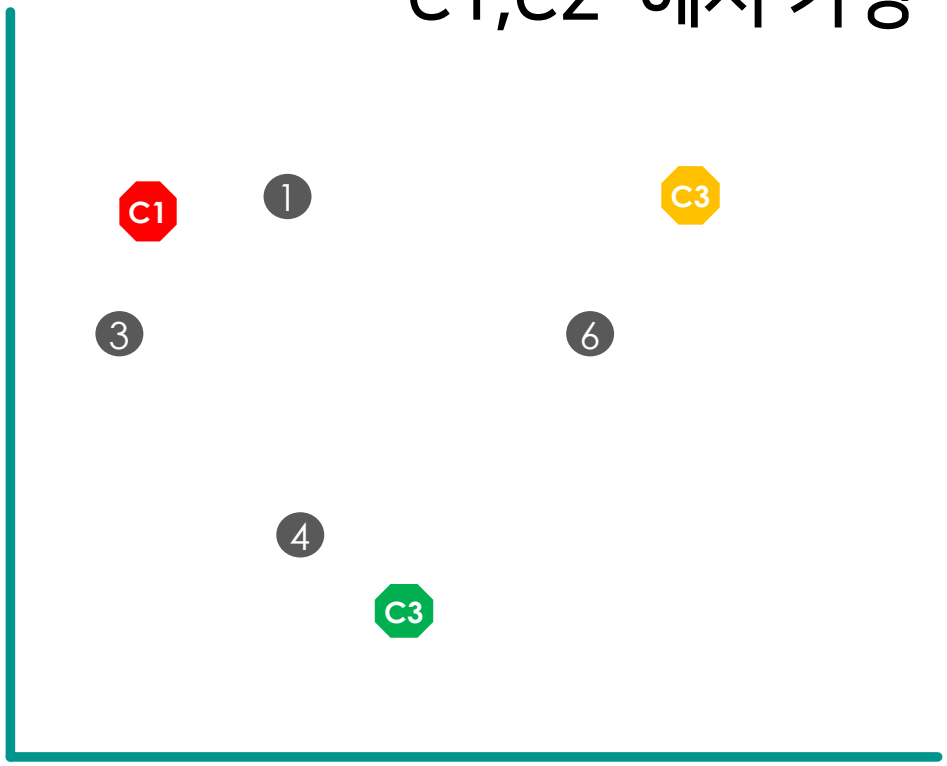
K-Means Clustering

C1 에서 가장 먼 점에 중심 설정



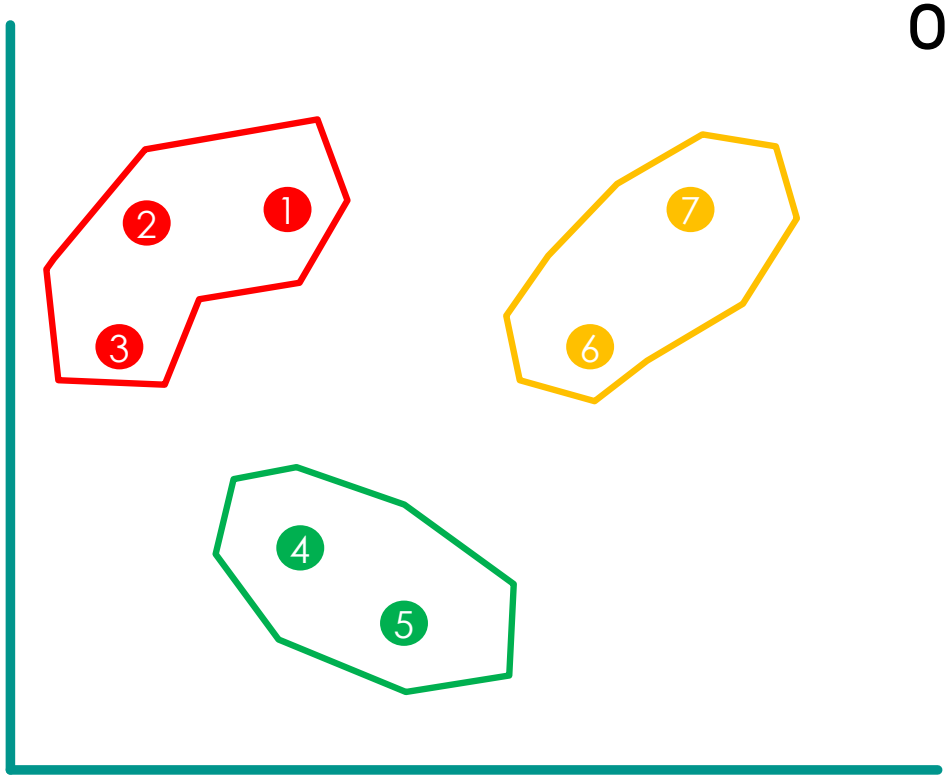
K-Means Clustering

C1,C2 에서 가장 먼 점에 중심 설정



K-Means Clustering

이후 과정 동일



클러스터의 개수는 몇개가 가장 좋을까요?

K-Means Clustering

클러스터의 개수는 몇개가 가장 좋을까요?

$$cluster = \sqrt{2/n}$$