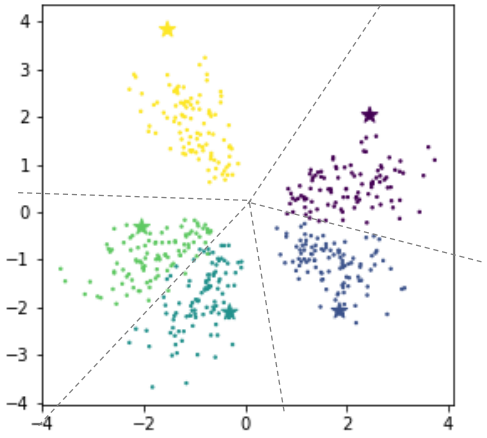
Decision Tree for Classification and Regression

Hyunjoong Kim

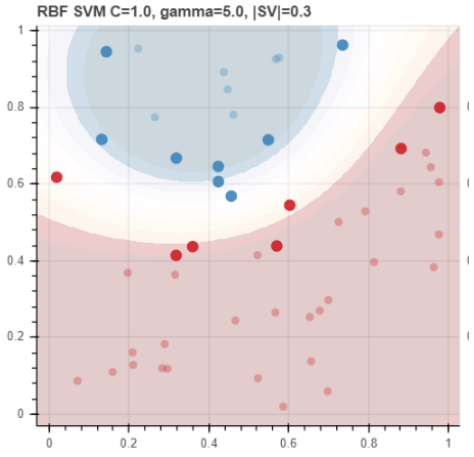
soy.lovit@gmail.com

github.com/lovit

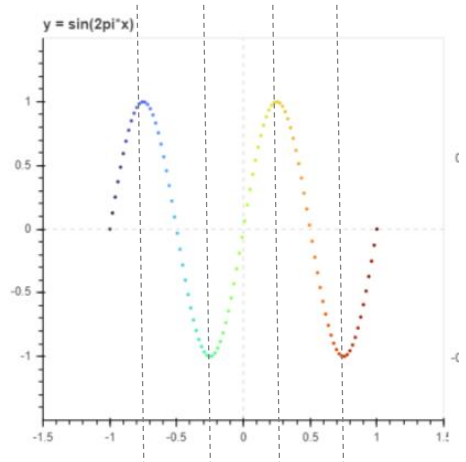
- 뉴럴넷, SVM 은 분류/회귀 모델은 데이터를 “특정 지역을 표현하는 (hidden) representation” 으로 변환한 뒤, 선형 분류/회귀 모델을 적용합니다.
  - 특정 지역 (spatial) 에 반복되는 패턴을 적은 수의 패러매터로 표현하려 합니다.
  - 각 지역은 동질적인 데이터들의 집합입니다.



Softmax decision boundary

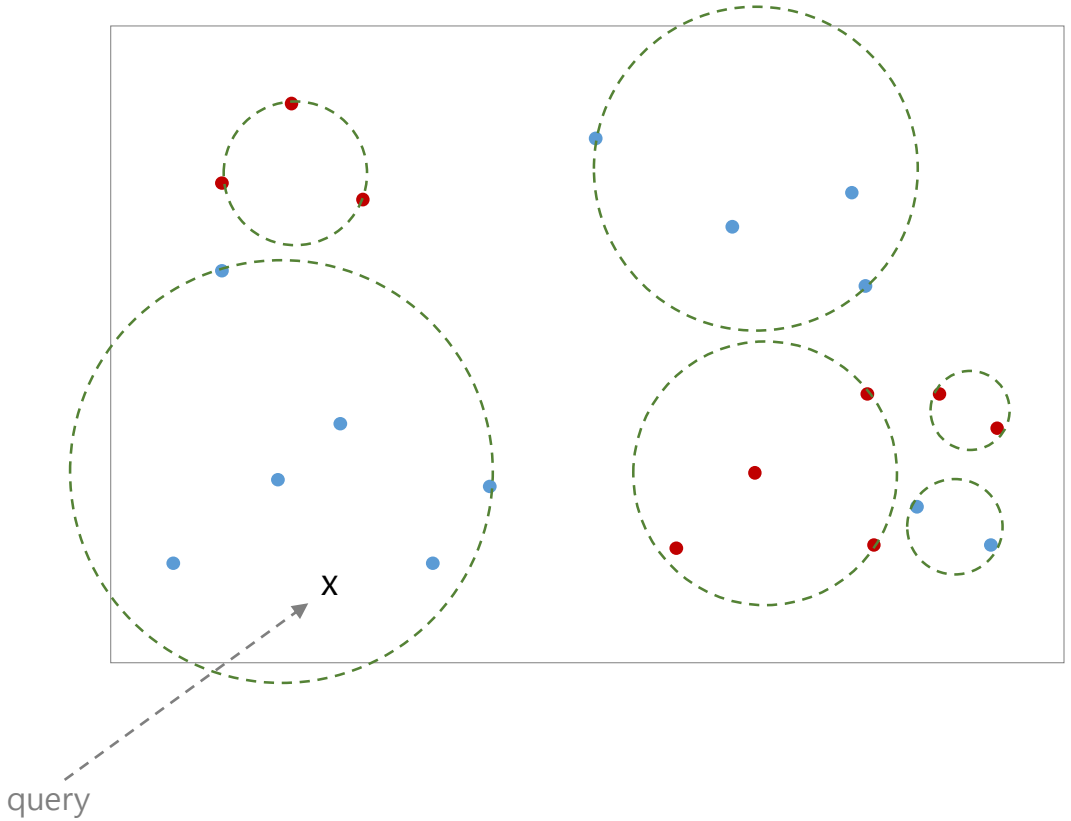


SVM decision boundary

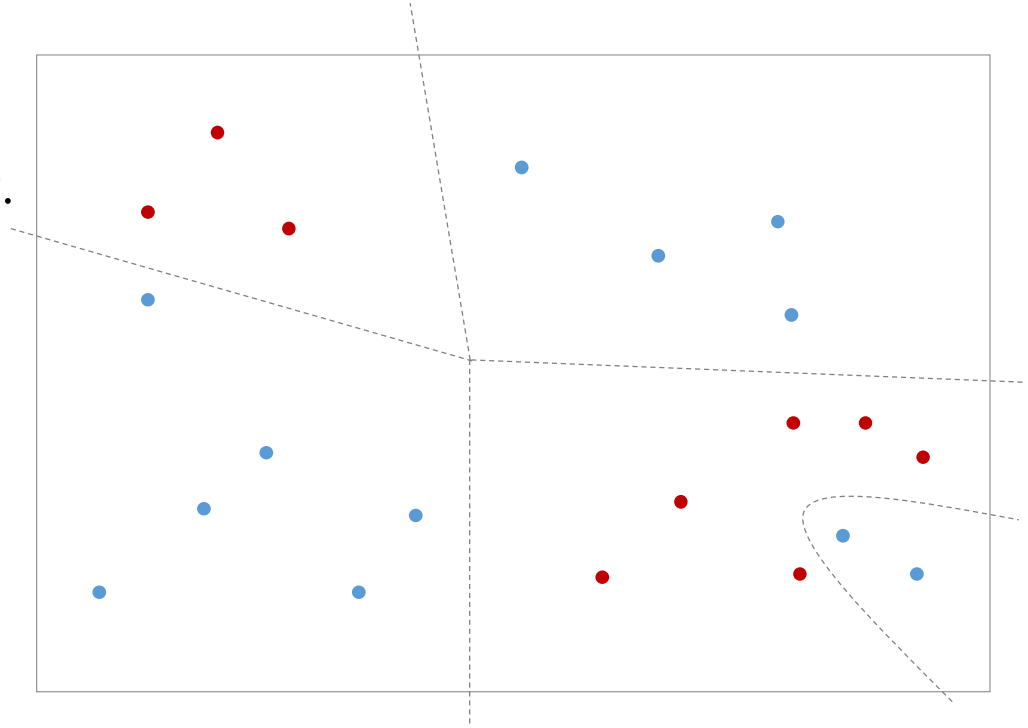


Feed-forward 1st hidden, h=(5,)

- Linear inseparable 하게 데이터가 분포하더라도 동질적인 데이터로 구성된 영역을 특정할 수 있습니다.
  - 같은 레이블을 지닌 점들을 감싸는 원 (구, sphere) 를 만들 수 있습니다.
  - query 벡터가 어떤 영역에 포함되는지 확인하면 클래스를 판단할 수 있습니다.

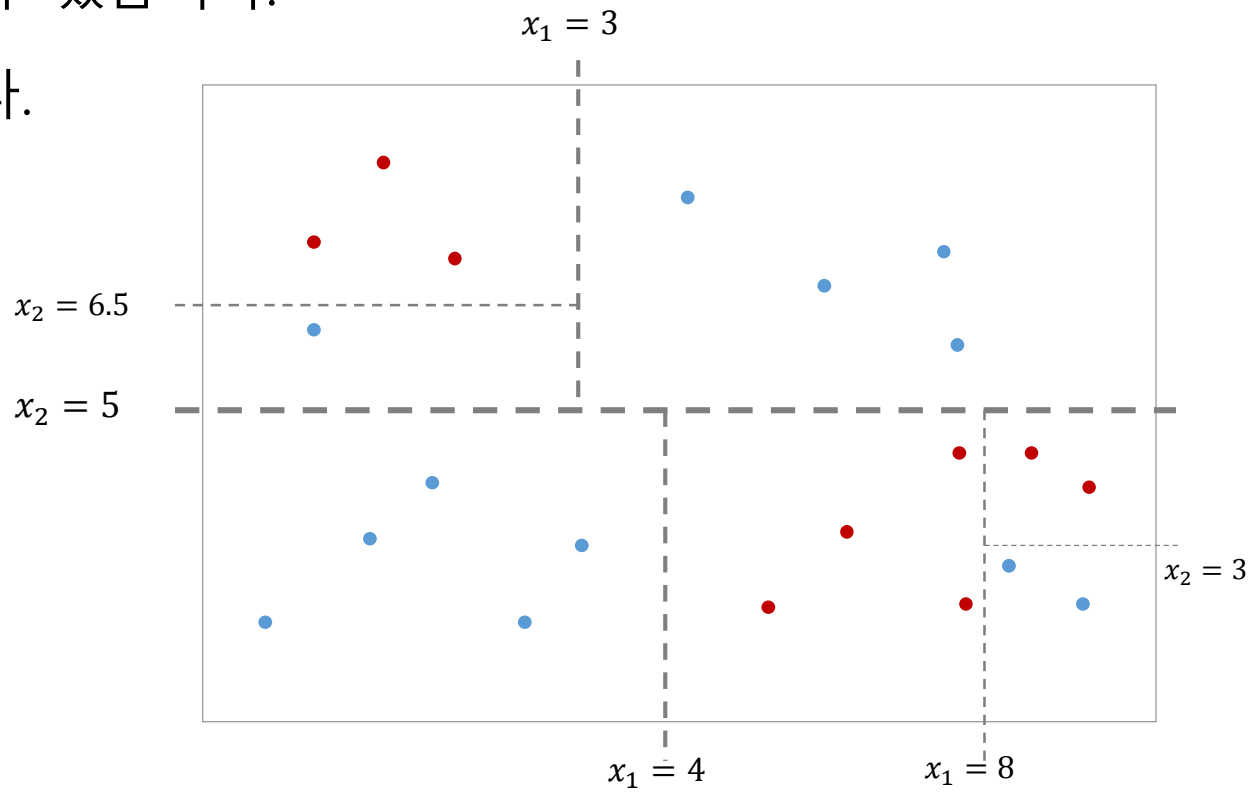


- Neural networks 로도 영역을 특정할 수 있습니다.
  - Kernel SVM 이나 여러 층의 뉴럴넷으로 non-linear hyper-plane 을 만들 수 있습니다.

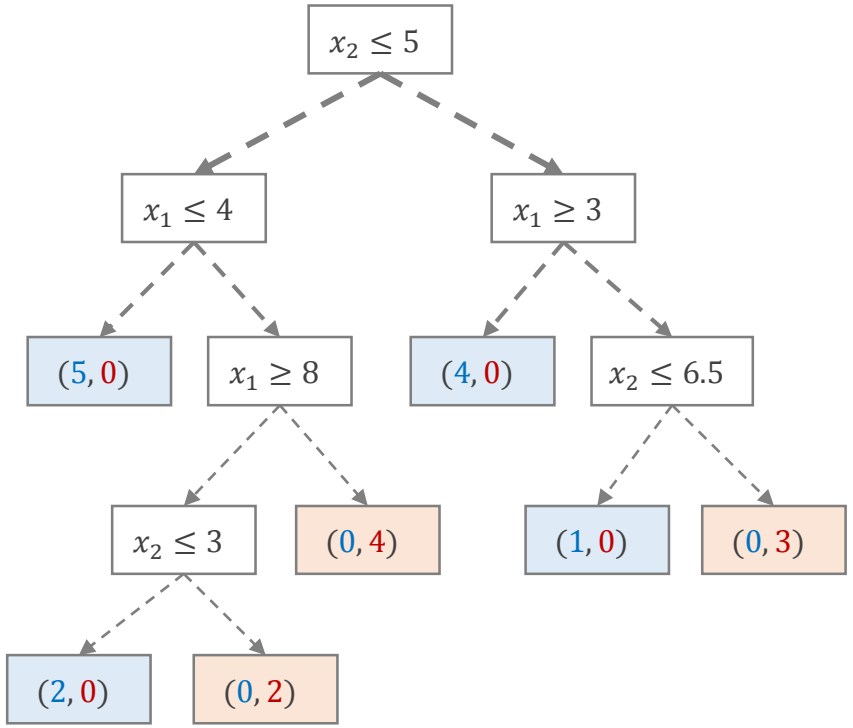


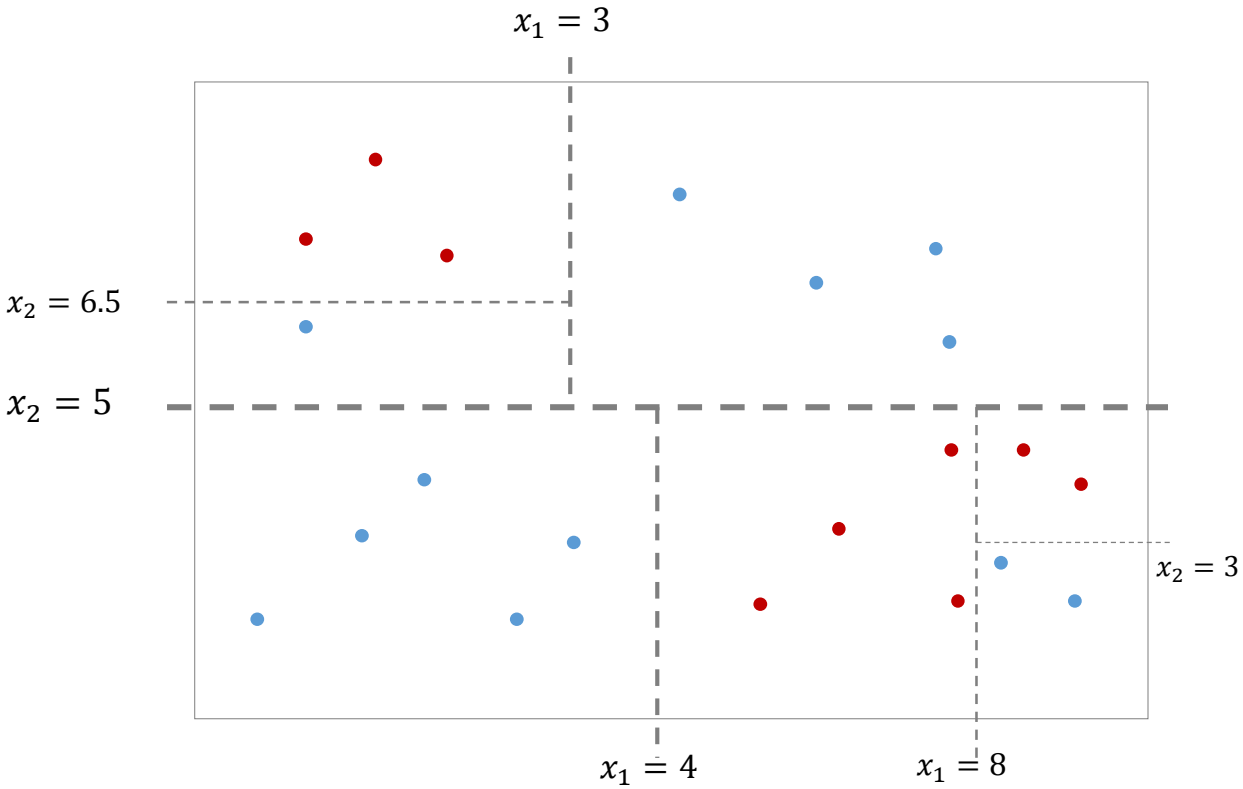
- Rectangular 로 동질적인 영역을 특정할 수 있습니다.
  - 직관적이며 영역을 표현하기 쉽습니다.

    IF $(x_1 \leq 4 \,\&\, x_2 \leq 5)$, THEN $blue$

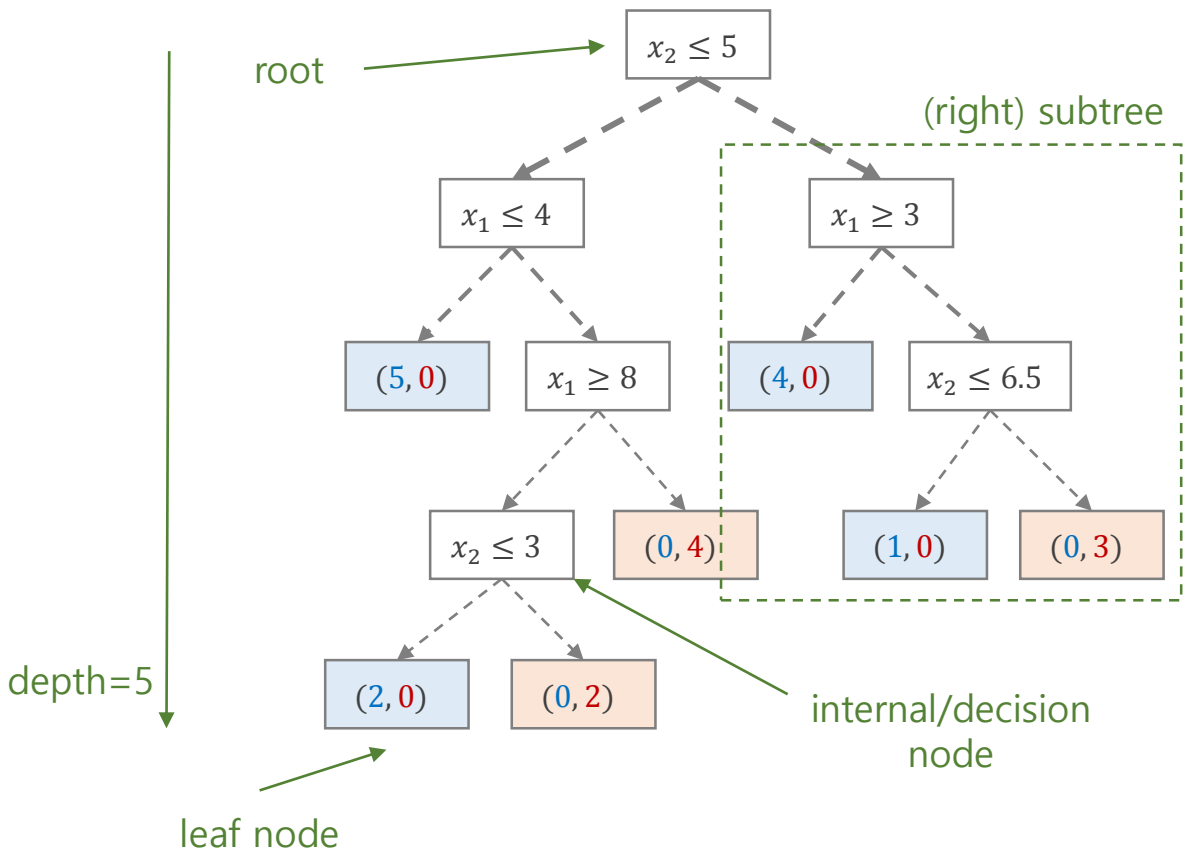


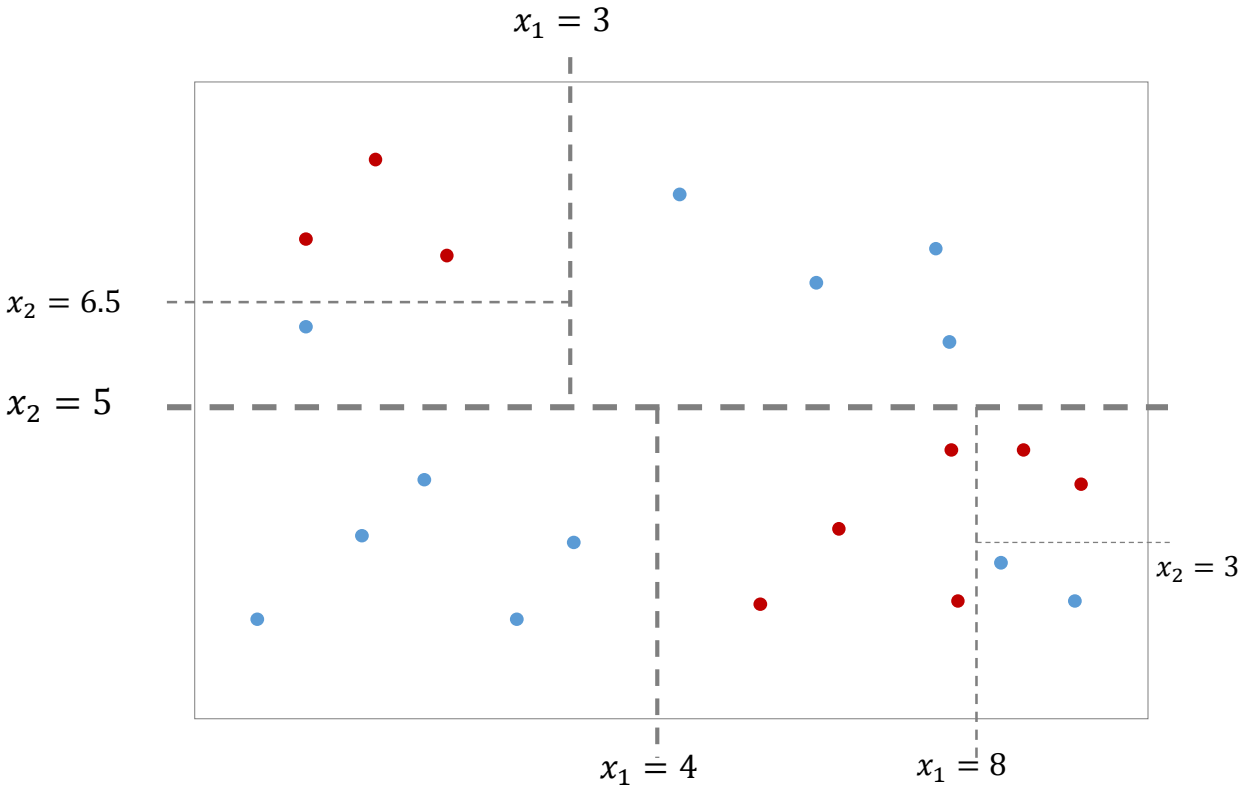
Tree traverse





Tree traverse



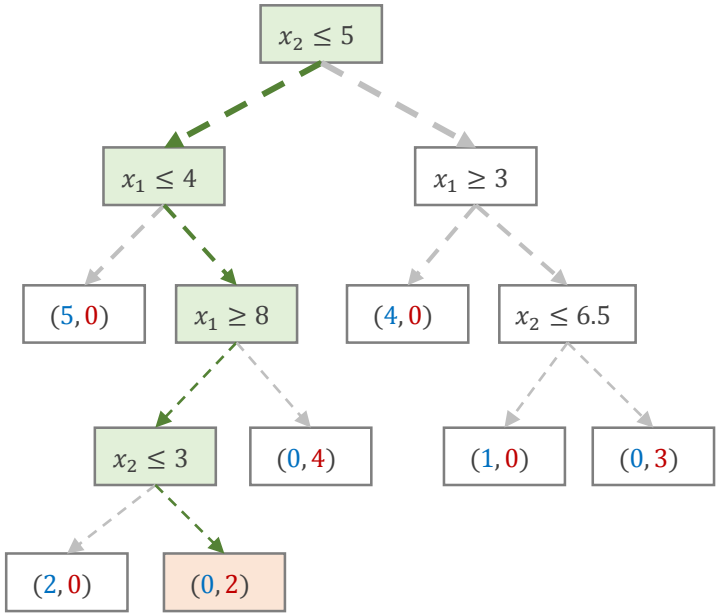


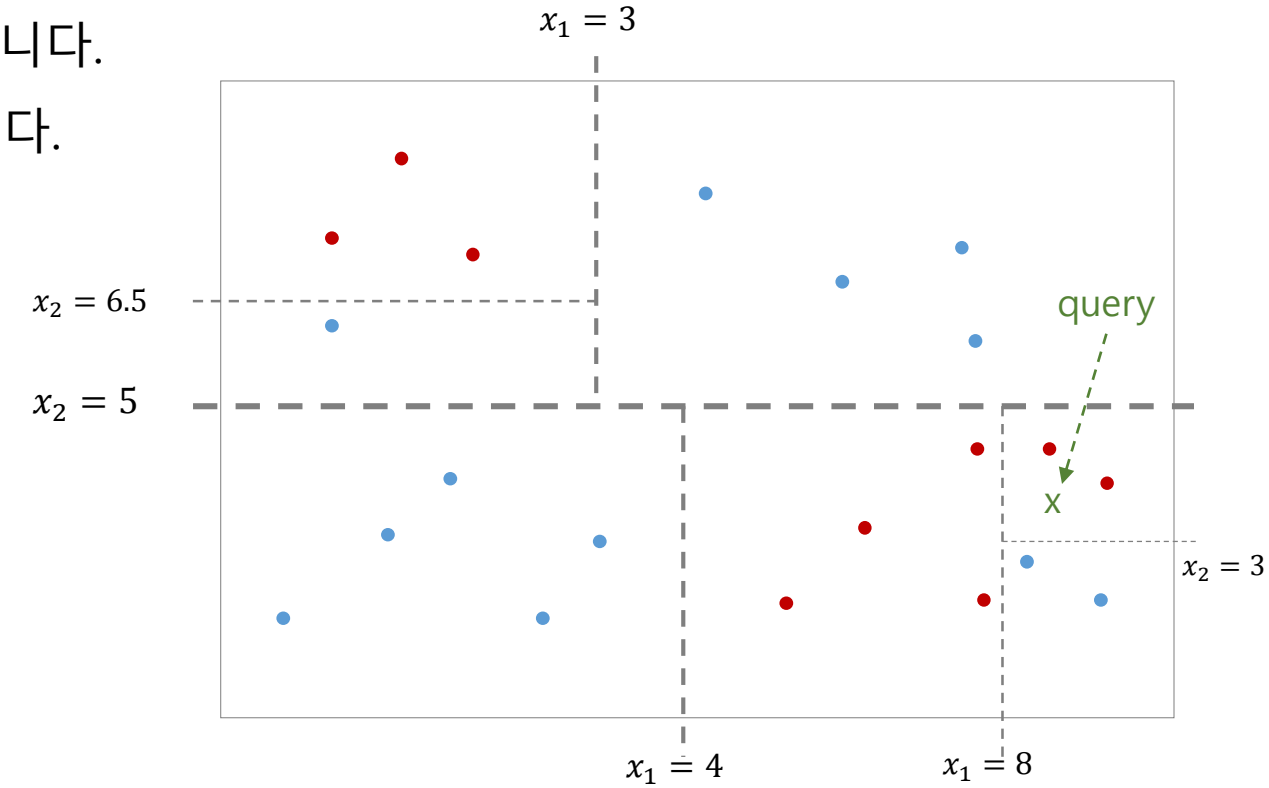
Tree traverse

- Query 에 대해서 tree 를 따라가며 분류를 합니다.
- 궤적을 해석 가능한 규칙으로 만들 수 있습니다.

$x_2 \leq 5 \;\&\; x_1 \leq 4 \;\&\; x_1 \geq 8 \;\&\; x_2 \leq 3$

$\rightarrow\; 8 \leq\; x_1 \;\&\; 3 \leq x_2 \leq 5$





Split

- 의사결정나무는 각 영역을 구성하는 데이터들의 클래스가 동일하도록 직사각형 모양의 부분영역들을 재귀적으로 탐색합니다.
  - 각 영역 내 클래스의 이질성을 줄이는 것이 목표입니다.
  - 영역 내 이질성을 측정하는 척도를 $Impurity\ (I)$ 라 합니다.
  - 영역 N 이 N_1, N_2 로 나뉘었을 때 감소하는 impurity 를 $Information\ Gain\ (IG)$ 라 합니다.

$$IG(N_1, N_2) = (w_1 + w_2)I(N) - w_1 I(N_1) - w_2 I(N_2)$$

Split

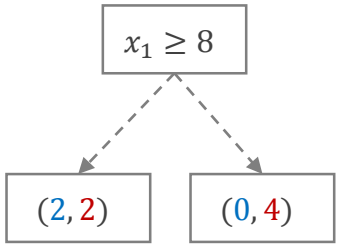
- 마디의 분기는 기준 이상의 이득 (information gain) 이 발생할 때에 이뤄집니다.
  - I 로 entropy 를 이용할 수 있습니다.

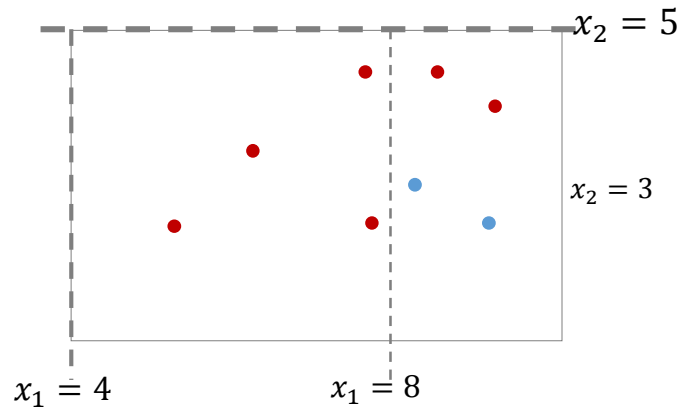
$$ent(p_i) = -\sum p_i \log p_i$$

$$IG(N_1, N_2) = I(N_1 + N_2) - w_1 I(N_1) - w_2 I(N_2)$$

$$= ent\left(\frac{6}{8}, \frac{2}{8}\right) - \frac{4}{8} ent\left(\frac{4}{4}, \frac{0}{4}\right) - \frac{4}{8} ent\left(\frac{2}{4}, \frac{2}{4}\right)$$

$$= 0.244 - 0 - 0.151 = 0.093 > 0$$





Split

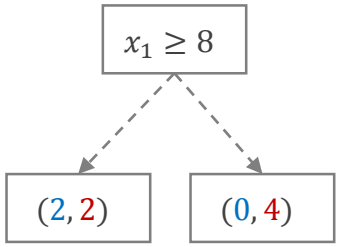
- 마디의 분기는 기준 이상의 이득 (information gain) 이 발생할 때에 이뤄집니다.
  - I 로 Gini Index 를 이용할 수 있습니다.

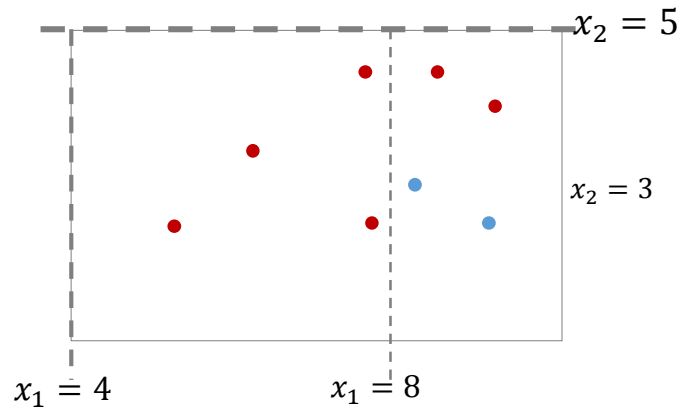
$$G(p_i) = 1 - \sum p_i^2$$

$$IG(N_1, N_2) = G(N_1 + N_2) - w_1 G(N_1) - w_2 G(N_2)$$

$$= G\left(\frac{6}{8}, \frac{2}{8}\right) - \frac{4}{8} G\left(\frac{4}{4}, \frac{0}{4}\right) - \frac{4}{8} G\left(\frac{2}{4}, \frac{2}{4}\right)$$

$$= 0.375 - 0 - 0.25 = 0.125 > 0$$





Split

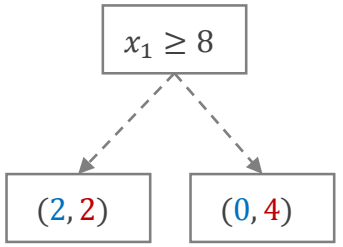
- 마디의 분기는 기준 이상의 이득 (information gain) 이 발생할 때에 이뤄집니다.
  - I 로 임의의 기준을 이용할 수 있습니다.

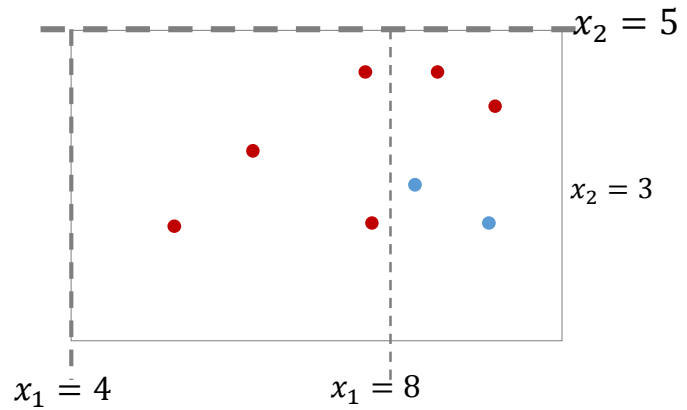
$$I(p_i) = 1 - \max(p)^2$$

$$IG(N_1, N_2) = I(N_1 + N_2) - w_1 I(N_1) - w_2 I(N_2)$$

$$= I\left(\frac{6}{8}, \frac{2}{8}\right) - \frac{4}{8} I\left(\frac{4}{4}, \frac{0}{4}\right) - \frac{4}{8} I\left(\frac{2}{4}, \frac{2}{4}\right)$$

$$= 0.4375 - 0 - 0.375 = 0.0625 > 0$$





Split

- 분기 기준을 탐색하기 위하여 가능한 모든 점들마다 IG 를 계산합니다.
  - 의사결정나무의 학습 과정 중 가장 오랜 시간이 걸리는 부분입니다.

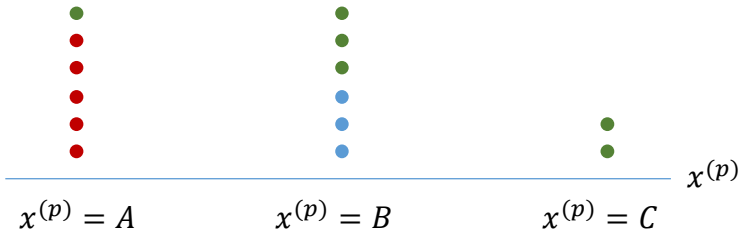
$x^{(p)}$
53
27
43
72
59
37
87
15
32

$\overrightarrow{x^{(p)}}$
15
27
32
37
43
53
59
72
87

Split point	IG
21	…
29.5	…
34.5	…
40	…
48	…
56	…
65.5	…
79.5	…

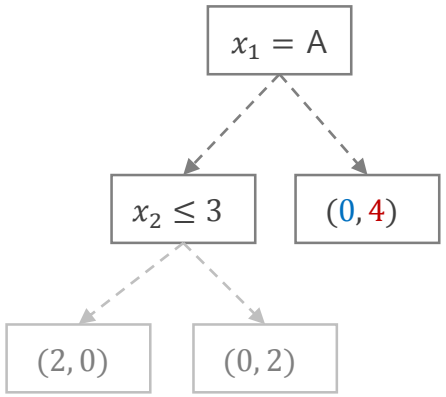
Split

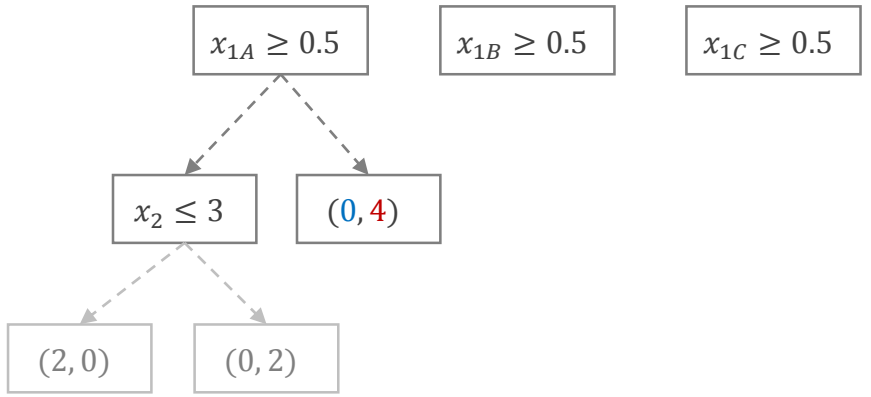
- 의사결정나무는 multi-class classification 을 다룰 수 있습니다.
  - Entropy 와 Gini index 는 확률 분포를 이용한 품질 평가 기준입니다.
- Categorical variable 에 대해서도 분기가 가능합니다.
  - IF $x^{(p)} = A$ THEN $y = red$



Split

- 대부분 구현체가 numerical matrix 를 입력받기 때문에 dummy coding 합니다.
  이 둘은 동일한 의미를 지닙니다.





Split

- k 개의 값을 지니는 명목형 변수를 $k-1$ 개의 dummy variables 로 변환하면 안됩니다.
  - 다음처럼 변수를 변환하면 "IF $x_{i,p} = A$" 인 경우가 제대로 학습되지 않을 수 있습니다.

$$x^{(p)} \in [A, B, C] \rightarrow \{A: (0,0), B: (0,1), C: (1,0)\}$$

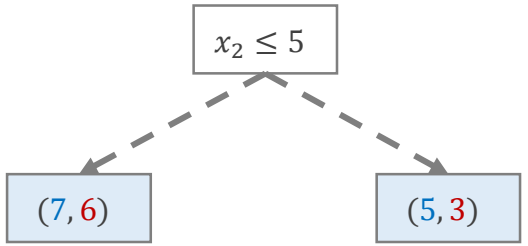
$$\text{IF } x_{i,pC} \leq 0.5 \rightarrow \{A, B\} \text{ 인 경우}$$

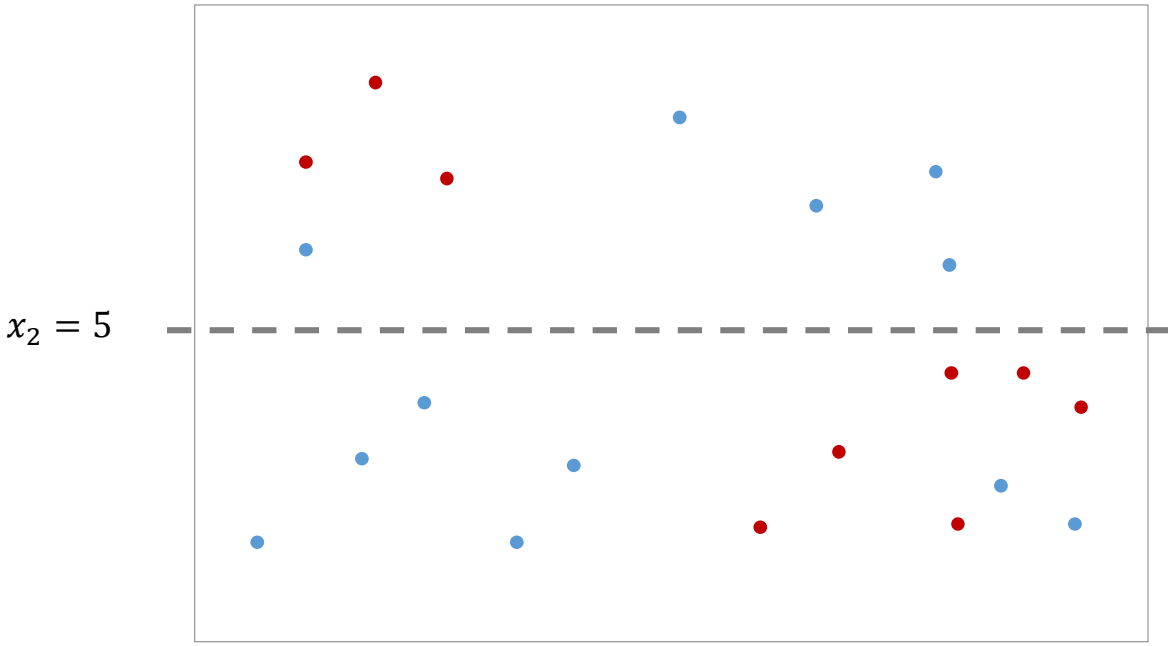
Split

- 그러나 dummy variable 은 부분집합 관계를 표현하기 어렵습니다.
  - $x_1 \in [A, B, C, D]$
  - $\rightarrow (x_{1A}, x_{1B}, x_{1C}, x_{1D})$

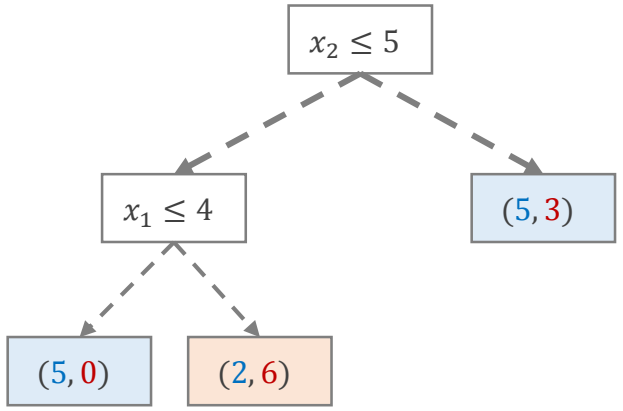
    $\rightarrow (x_{1,AB}, x_{1,AC}, x_{1,AD}, x_{1,BC}, x_{1,BD}, x_{1,CD}, x_{1,ABC}, \dots)$

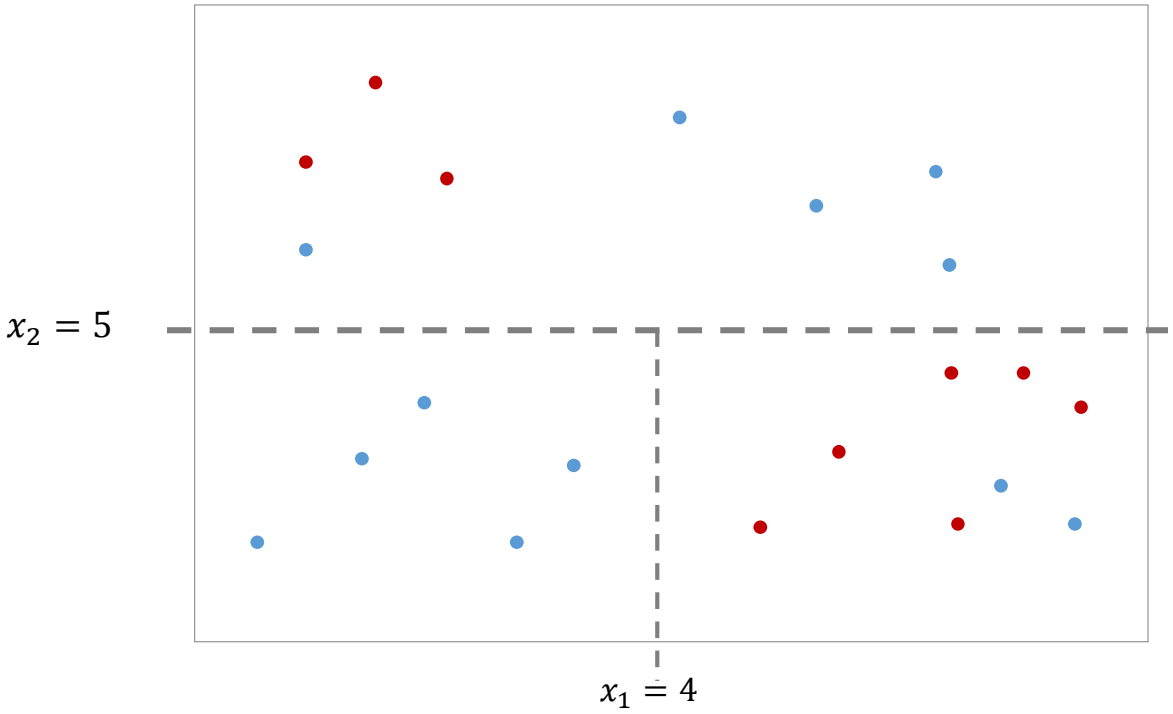
Demo



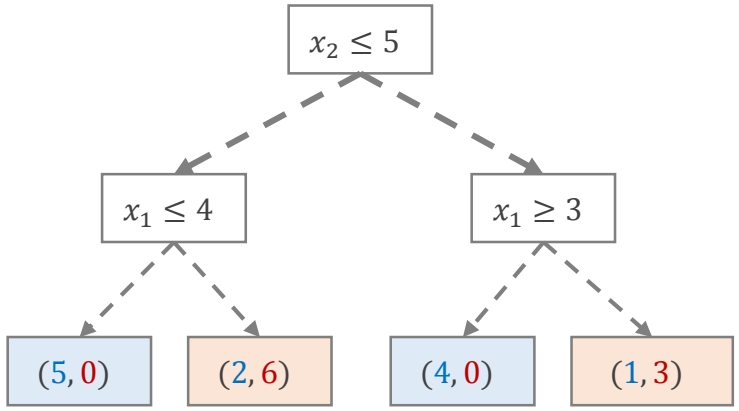


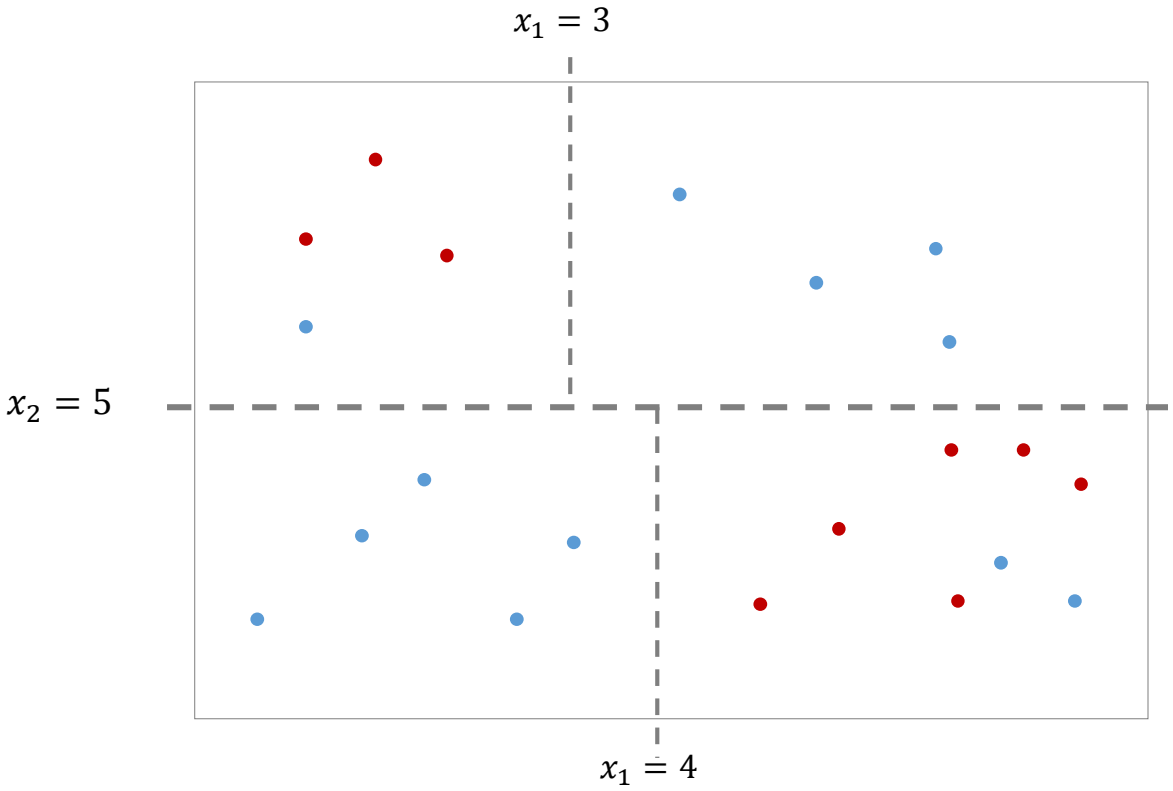
Demo



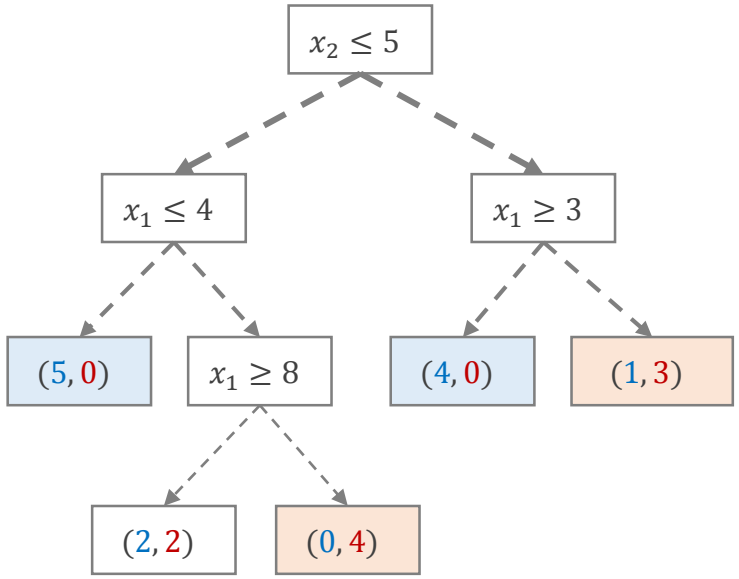


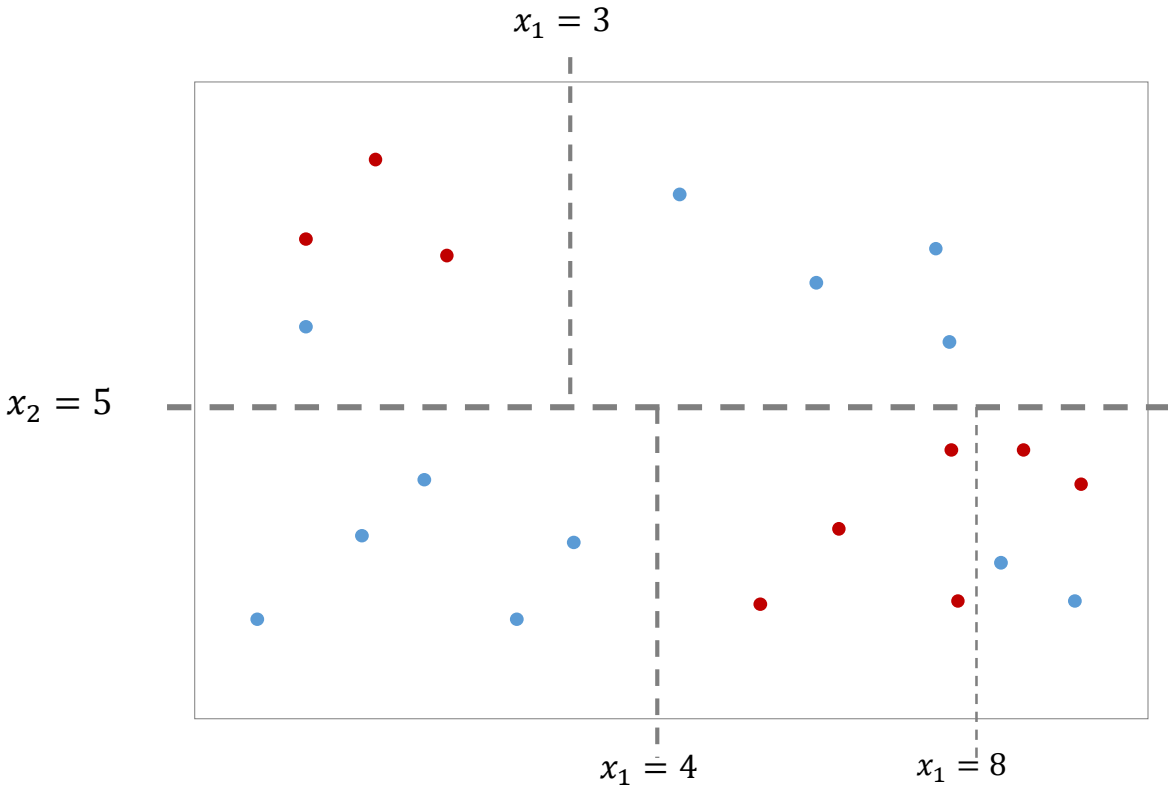
Demo



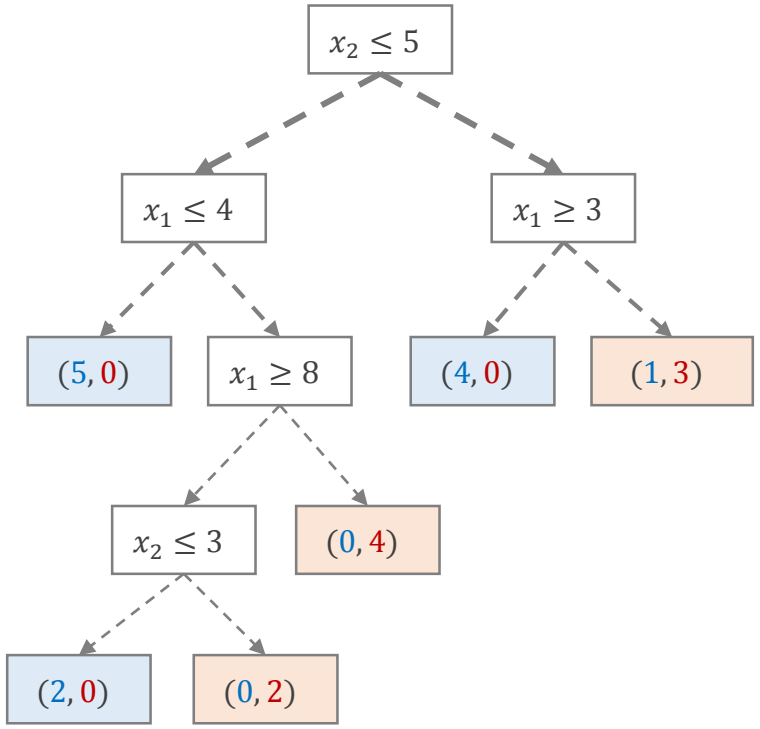


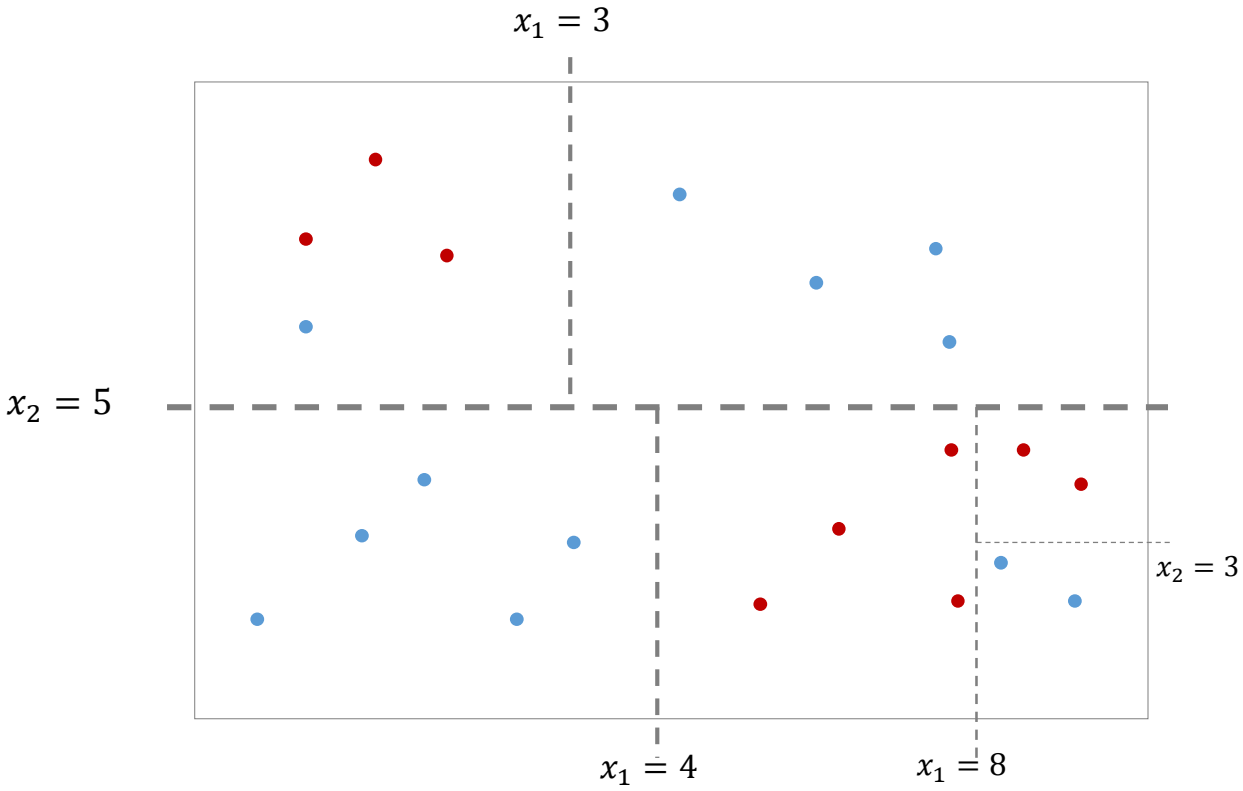
Demo



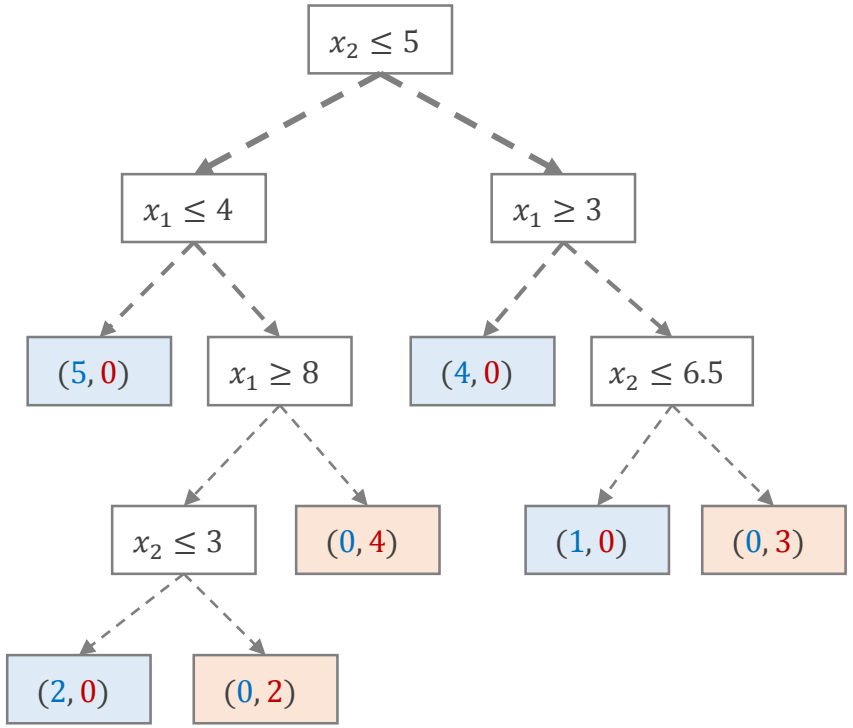


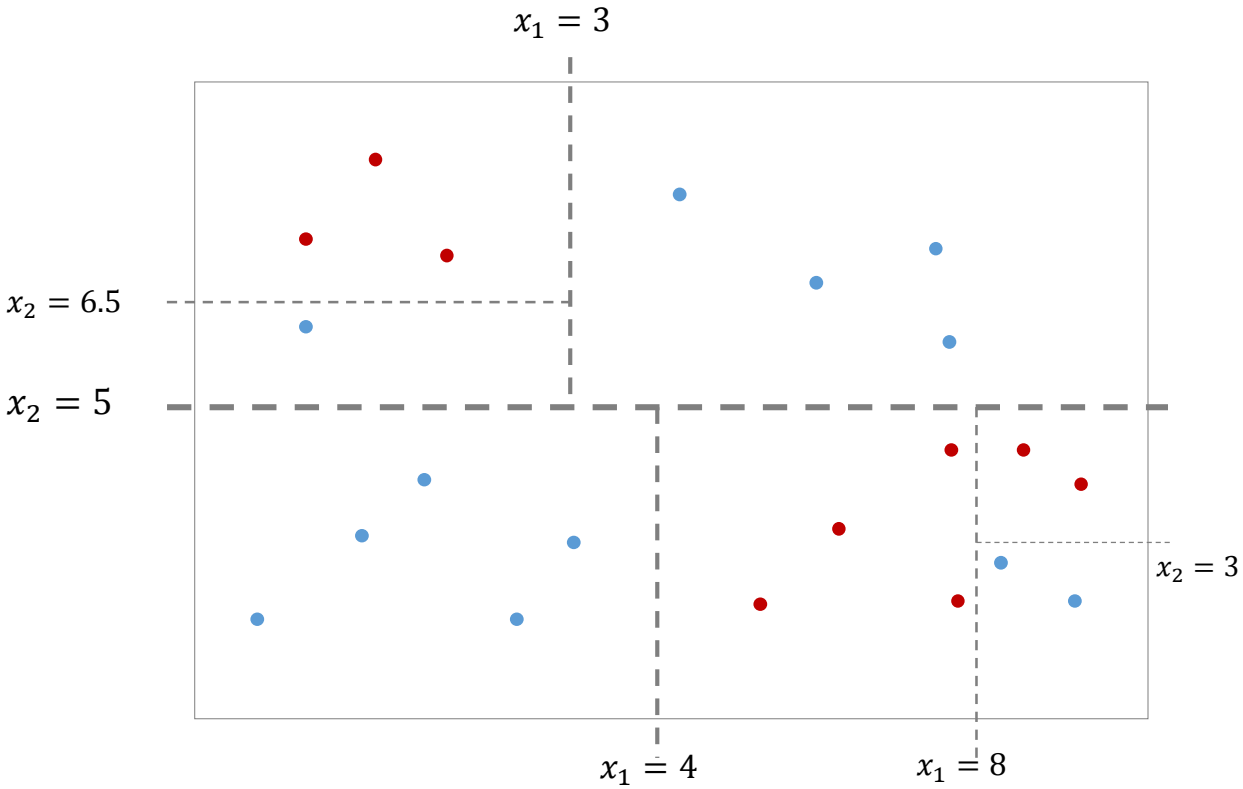
Demo


$x_2 \leq 5$
$x_1 \leq 4$
$x_1 \geq 3$
(5, 0)
$x_1 \geq 8$
(4, 0)
(1, 3)
$x_2 \leq 3$
(0, 4)
(2, 0)
(0, 2)


$x_1 = 3$
$x_2 = 5$
$x_2 = 3$
$x_1 = 4$
$x_1 = 8$

Demo


$x_2 \leq 5$
$x_1 \leq 4$
$x_1 \geq 3$
$(5, 0)$
$x_1 \geq 8$
$(4, 0)$
$x_2 \leq 6.5$
$x_2 \leq 3$
$(0, 4)$
$(1, 0)$
$(0, 3)$
$(2, 0)$
$(0, 2)$


$x_1 = 3$
$x_2 = 6.5$
$x_2 = 5$
$x_2 = 3$
$x_1 = 4$
$x_1 = 8$

Missing Value Handling

- Missing value 도 하나의 값으로 인식하여 처리할 수 있습니다.
  - IF $x_{i,p} = NaN$ THEN ...
  - 그러나 구현체마다 처리 방식이 다릅니다.
  - 벡터 형식의 학습데이터를 입력받는 scikit-learn 은 사용자가 missing value 를 특정 값으로 모두 바꿔야 합니다.
  - 구현체에 따라서는 NaN 을 0 으로 인식하기도 합니다.

Missing Value Handling

- Most frequent value
  - 한 변수 $x^{(p)}$ 에 가장 많이 등장한 값으로 missing value 를 대체합니다.
  - 일반적으로 가장 성능이 좋지 않은 방법입니다.
  - Using correlated other variables
  - Reconstruction

Missing Value Handling

- Using correlated variables
  - 한 변수 $x^{(p)}$ 와 상관성이 높은 다른 변수 $x^{(q)}$ 를 이용하여 $x^{(p)}$ 를 추정합니다.
  - 두 변수 $x^{(p)}$, $x^{(q)}$ 가 선형 관계일 때 이용할 수 있습니다.

$$x_{i,p} = a_{pq}x_{i,q} + b_{pq}$$

  - $x^{(p)}$ 를 제외한 다른 변수들을 이용하여 $x^{(p)}$ 를 예측하는 모델을 학습합니다.
    여러 변수간 관계를 반영할 수 있으며, 비선형 관계일때에도 이용 가능합니다.

$$x_{i,p} = f\left(x_{i,-p}\right)$$

Missing Value Handling

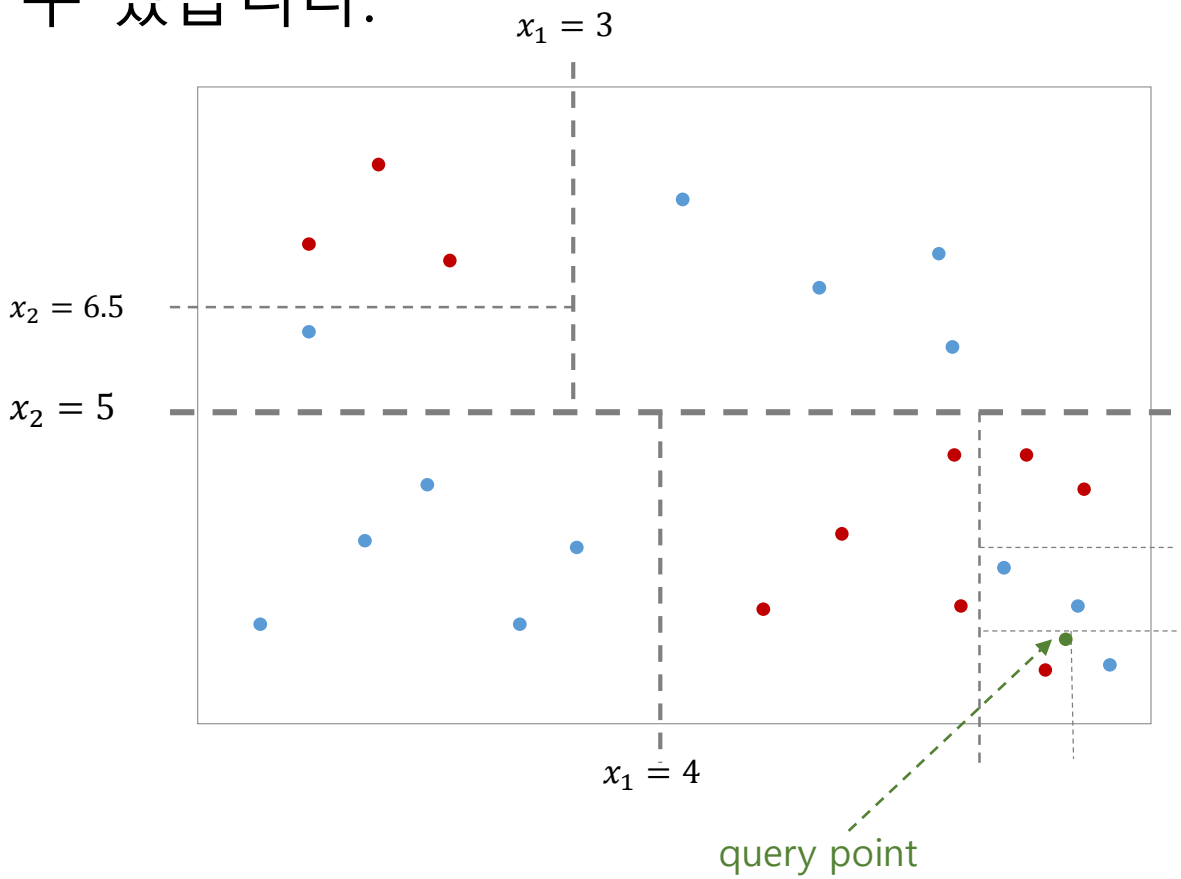
- Reconstruction
  - 한 변수 $x^{(p)}$ 를 제외한 다른 변수 $x^{(-p)}$ 를 이용하여 가까운 다른 점들을 탐색합니다. 이들의 평균벡터를 이용하여 missing value 를 추정합니다

$$x_{i,p} = \sum_{x_j \in N(x_i)} w\left(x_i^{(-p)}, x_j^{(-p)}\right) x_{j,p}$$

  - 변수 별 상관성이 반영되며, 비선형 관계에서도 이용할 수 있습니다.
  - 이외에도 다양한 reconstruction 기반 방법들이 이용될 수 있습니다.

Regularization

- 과적합된 모델은 잘못된 추정을 할 수 있습니다.
  - 사전/사후 정규화 방법을 이용하여 과적합을 방지할 수 있습니다.



Regularization

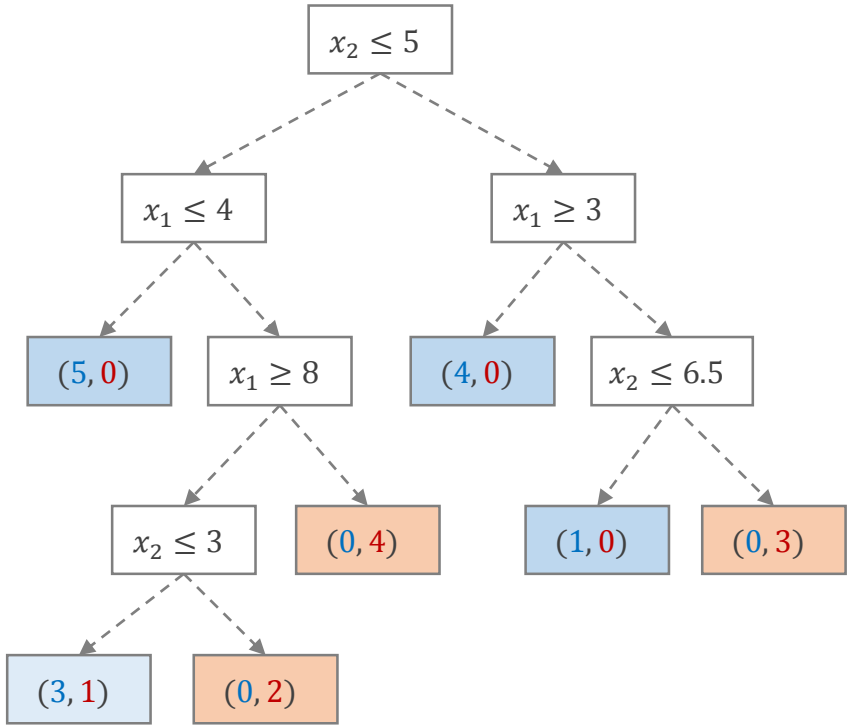
- Decision Tree 의 비용은 나무의 마디 개수와 정확도 향상의 차이입니다.

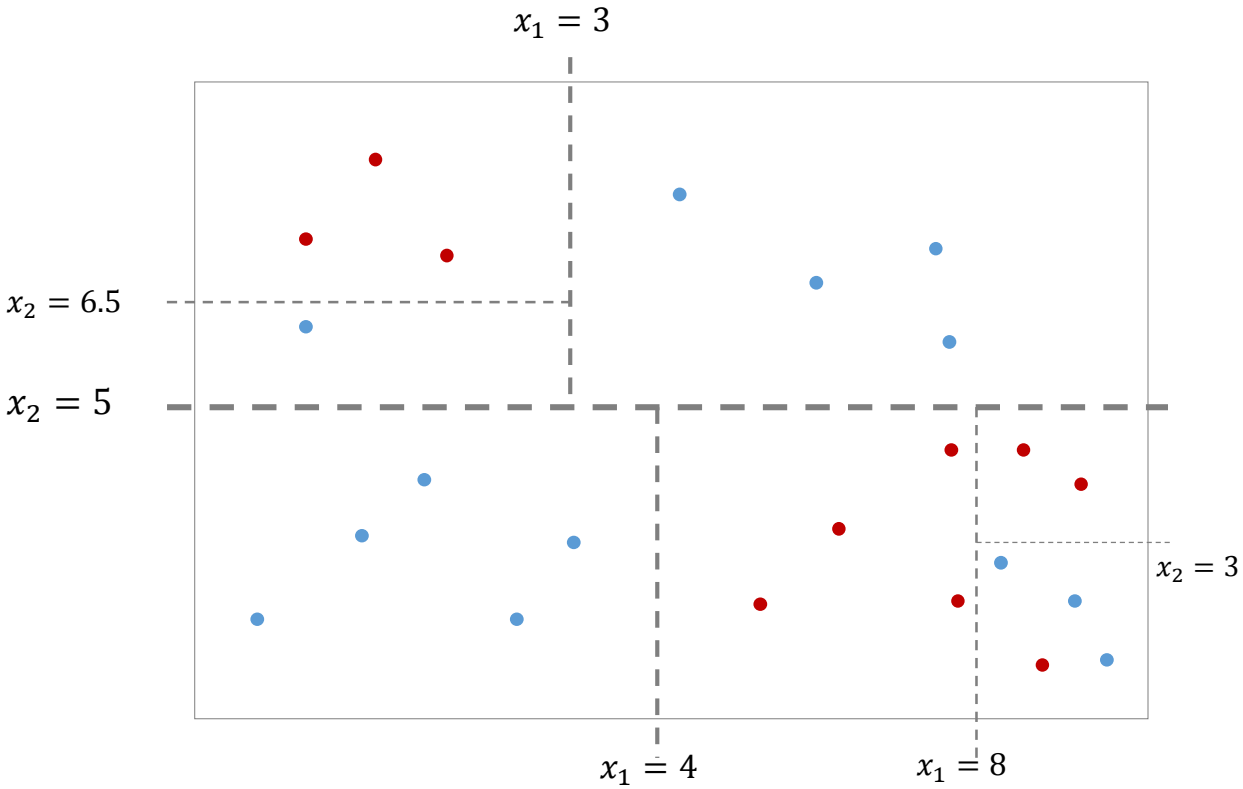
$$C(T) = -\sum IG(N) + \alpha|T|$$

$|T|$: number of leaves

  - Information gain 이 α 보다 작으면 분기하지 않습니다.
  - 지나치게 작은 마디로 나뉘어지는 것 (과적합) 을 방지합니다.

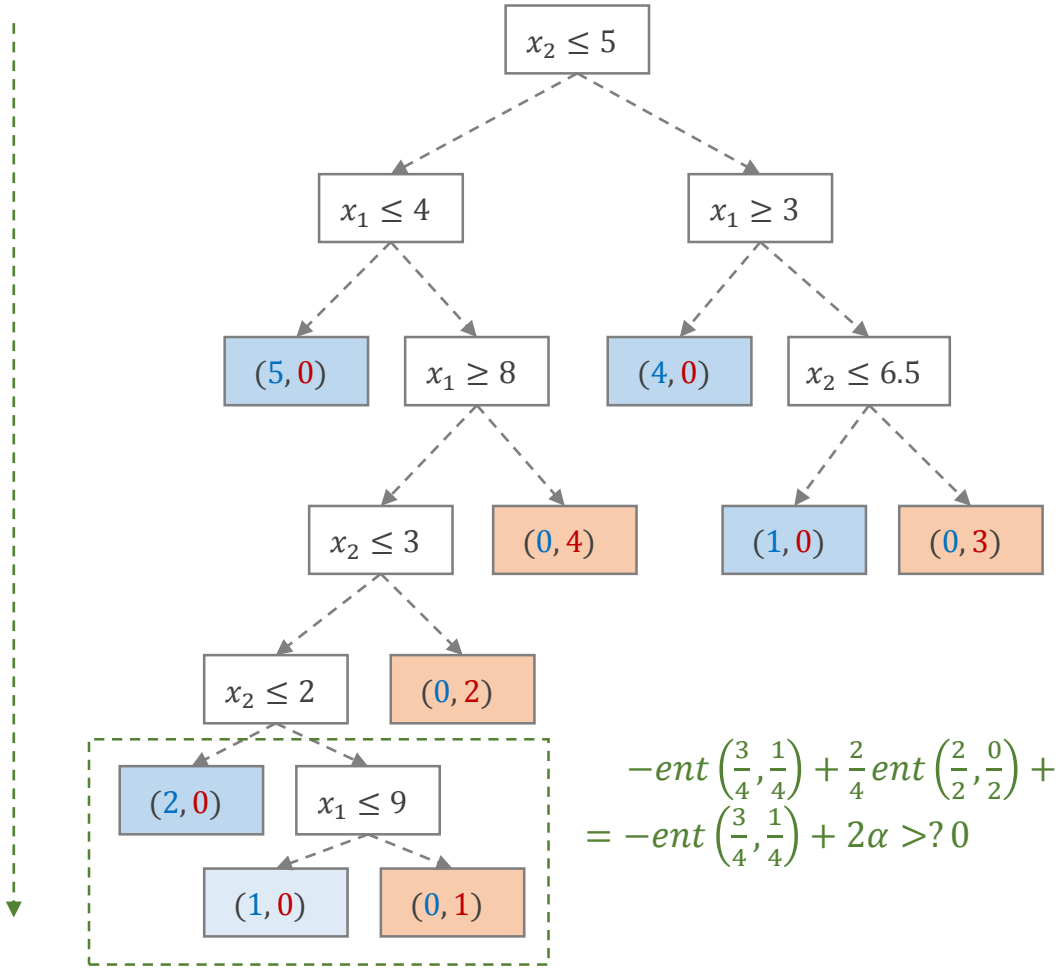
Regularization

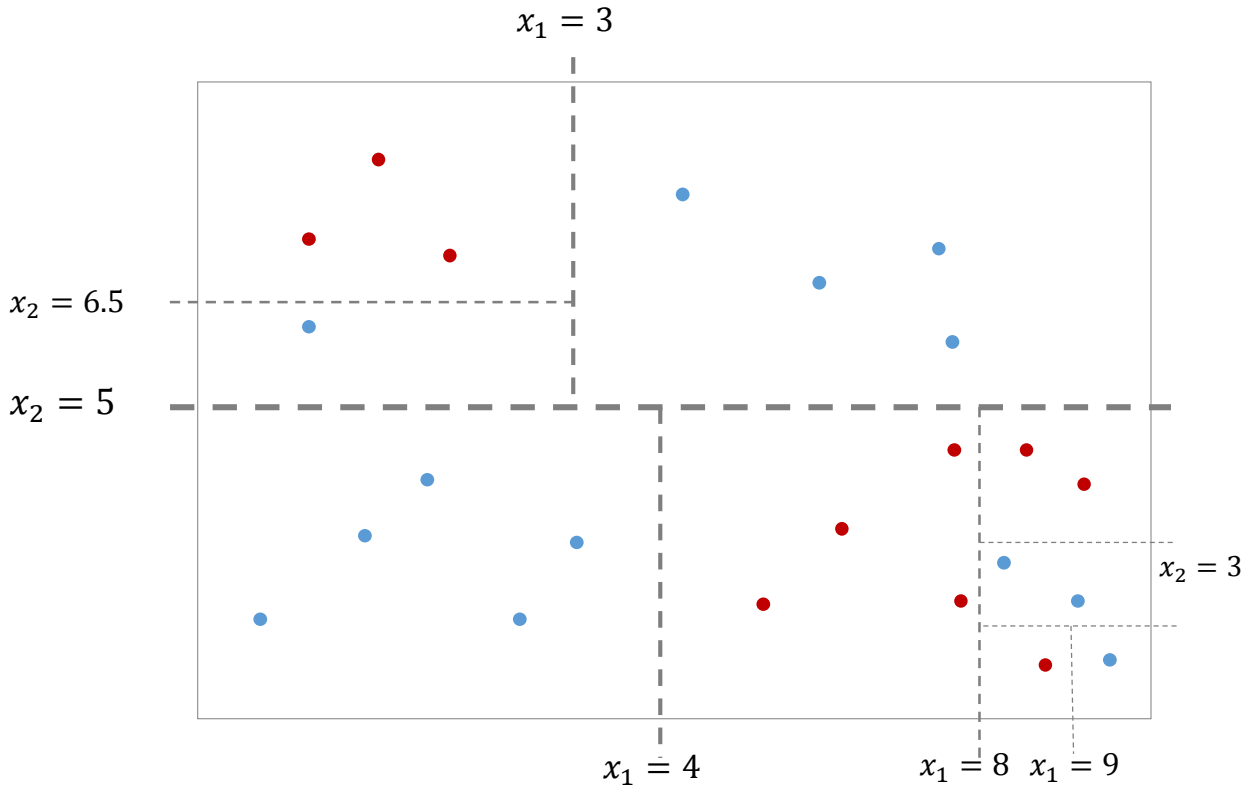




Regularization

- 한쪽 방향으로 나무가 지나치게 깊습니다.
- 분기를 통하여 큰 이득이 발생하지 않습니다.

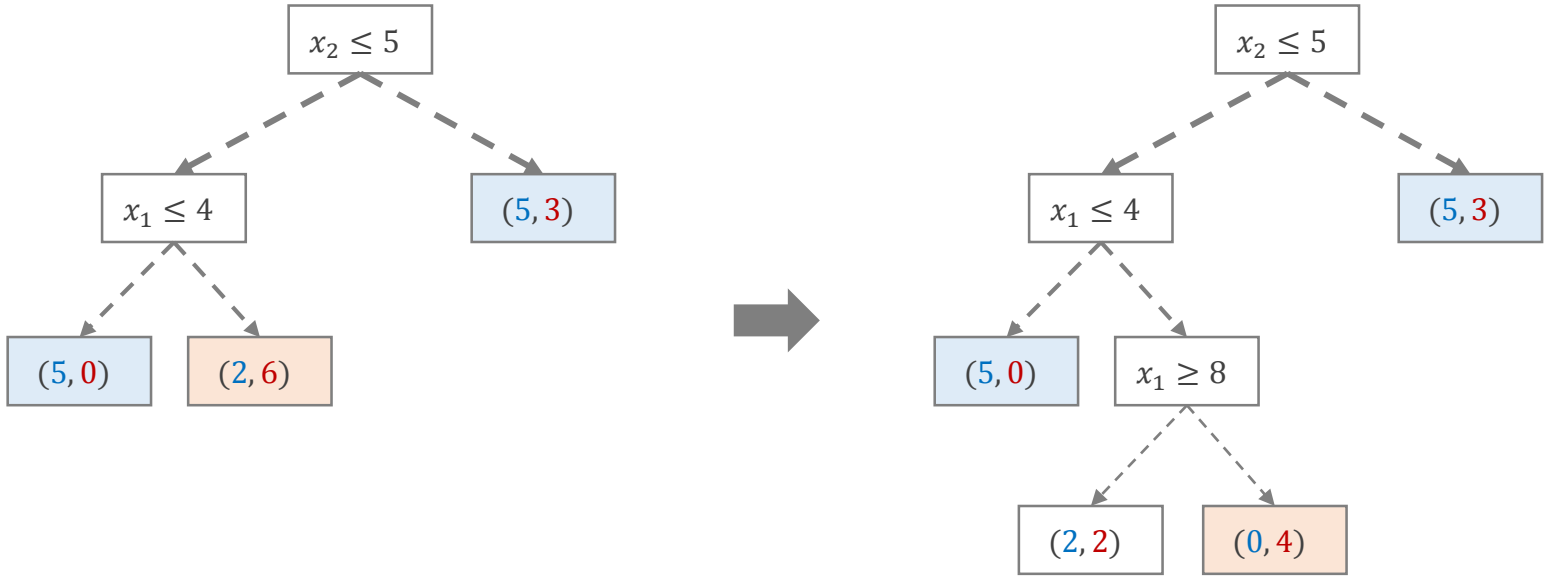




$$-ent\left(\frac{3}{4}, \frac{1}{4}\right) + \frac{2}{4} ent\left(\frac{2}{2}, \frac{0}{2}\right) + \frac{1}{2} ent\left(\frac{1}{1}, \frac{0}{1}\right) + \frac{1}{2} ent\left(\frac{1}{1}, \frac{0}{1}\right) + 2\alpha$$

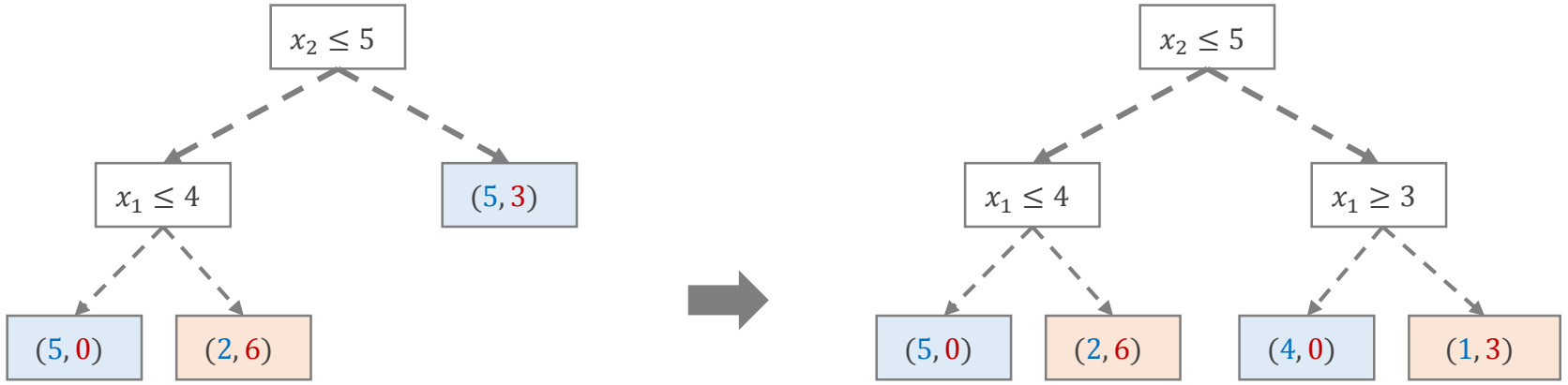
$$= -ent\left(\frac{3}{4}, \frac{1}{4}\right) + 2\alpha >? 0$$

Regularization



- Leaf-wise split 방식은 IG 가 큰 마디부터 분기합니다.
- 한 방향의 subtree 만 자라날 수 있습니다 (imbalanced tree)

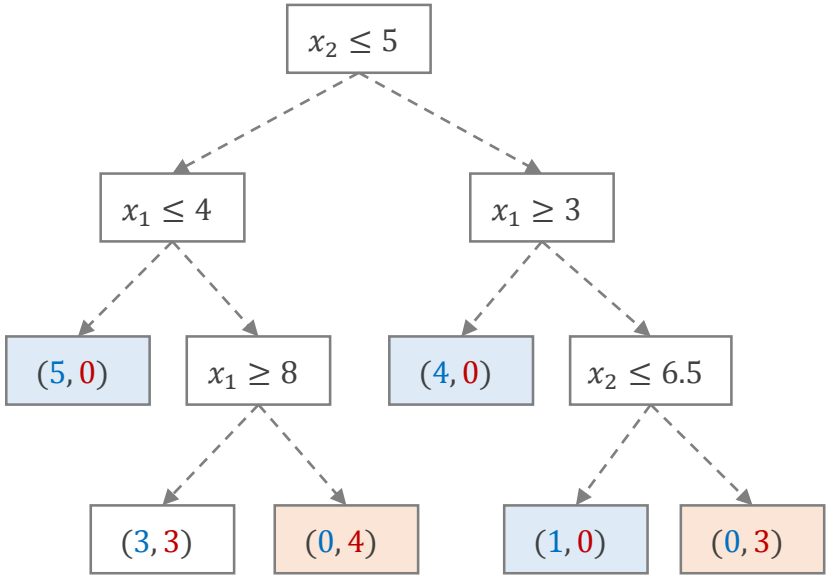
Regularization



- Level-wise split 방식은 같은 깊이의 마디가 모두 분기되도록 유도합니다.
- 나무의 깊이를 제한함으로써 한 지역에 과적합이 발생하는 것을 방지합니다.

Regularization

- max depth & number of leaves 를 통하여 과적합을 방지할 수 있습니다.
  - max depth = 4, number of leaves = 8 로 설정하면 $(3, 3)$ 인 마디가 더이상 분기하지 않습니다.



Regularization

- 그 외에도 다양한 사전 정규화를 위한 하이퍼 패러매터가 제공됩니다.
  - minimum split size
  - minimum node size
  - minimum information gain

Regularization

- 사후 정규화 방법을 이용할 수도 있습니다.
  - 의사결정나무를 최대한 학습시킨 뒤 (full grown tree), validation data 를 이용하여 일반화 성능이 감소되는 지점의 분기를 취소합니다.

Pros & Cons

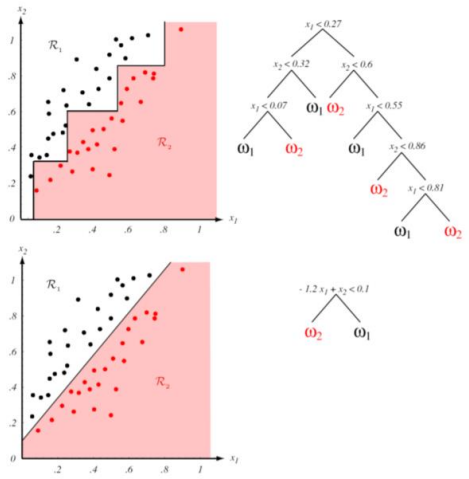
- 의사결정나무는 해석력이 있으며, 모델 피팅이 편리합니다.
  - 스캐일링, missing value 등의 전처리 과정에 영향이 적습니다.
  - (가능한) 적은 수의 분류에 중요한 변수를 선택합니다.
  - 선택된 변수와 경계값을 통하여 해석이 용이한 규칙이 도출됩니다.
  - 학습된 모델의 패러매터 크기가 매우 적습니다. Inference 가 빠릅니다.
    - # features = 100, # classes = 10

      Logistic regression # param. = 1000

      DT 는 분류기준과 leaves 의 확률값만 모델에 저장합니다.

Pros & Cons

- 노이즈와 데이터의 분포에 민감합니다.
  - 값이 조금만 바뀌어도 branch 에 이용되는 변수가 바뀔 수도 있습니다.
- 동시에 하나의 변수만 고려 가능합니다.
  - 소수의 변수만 이용하는 것이 목적이라면 LASSO 모델이 더 효과적입니다.
  - 비선형/계단형 관계를 표현하기 위하여 나무의 깊이가 깊어야 합니다.



Fat/sparse data + Decision Tree

- 의사결정나무는 Sparse vector 에 과적합을 하거나 학습이 제대로 되지 않습니다.
  - 의사결정나무는 적은 수의 변수로 데이터를 분류하려 합니다.
  - Sparse 형식의 데이터에는 각 점마다 다양한 변수가 등장하여 많은 점들이 자신이 지닌 변수를 활용하지 못한 채 분류됩니다.

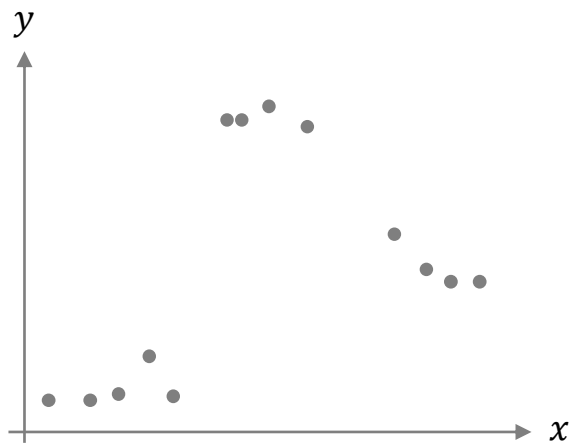
Fat/sparse data + Decision Tree

- $\{T_1, T_3, T_{13}, T_{14}\}$ 만 이용하면 이 변수가 등장하지 않은 데이터는 잘못 분류될 수 있습니다.

	y	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14
	0	5	3													
	0	3	2		5		1			2						
	0	2	4		4											
	0	5		2				4	5	3						
misclassified	0	1		1		2										
	0	4			1											
	1	2								2					2	
	1	3							5					4	4	
misclassified	1	5								1	1		3			
	1	1								2			2			3
	1	3								4		1		2	1	1
	1	2								4				1		2

Decision Tree for Regression

- 의사결정나무는 회귀분석에도 이용할 수 있습니다.
  - y 가 비슷한 지역들을 특정한 뒤, $\bar{y}$ 를 계산합니다.
  - Classification And Regression Tree (CART) 라 불립니다.

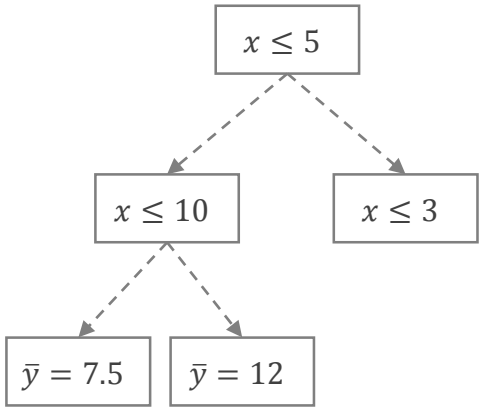


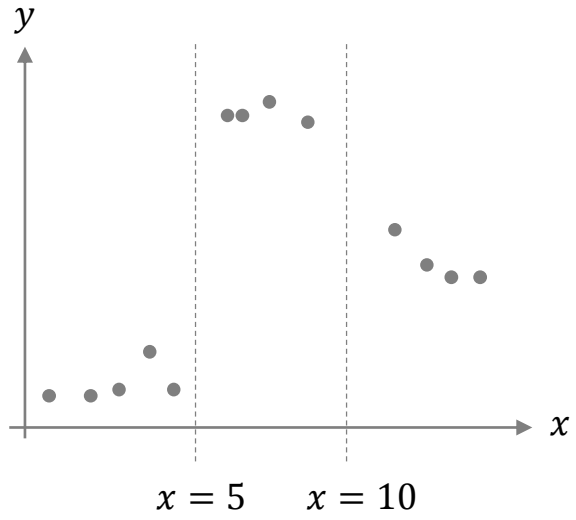
Decision Tree for Regression

- $Impurity\ (loss)$ 로 회귀분석의 품질 평가 기준들을 이용할 수 있습니다.
  - Squared errors, MAP, … 등 모두 이용 가능합니다.

$$L(N) = \sum_{i \in N}(y_i - \bar{y})^2$$

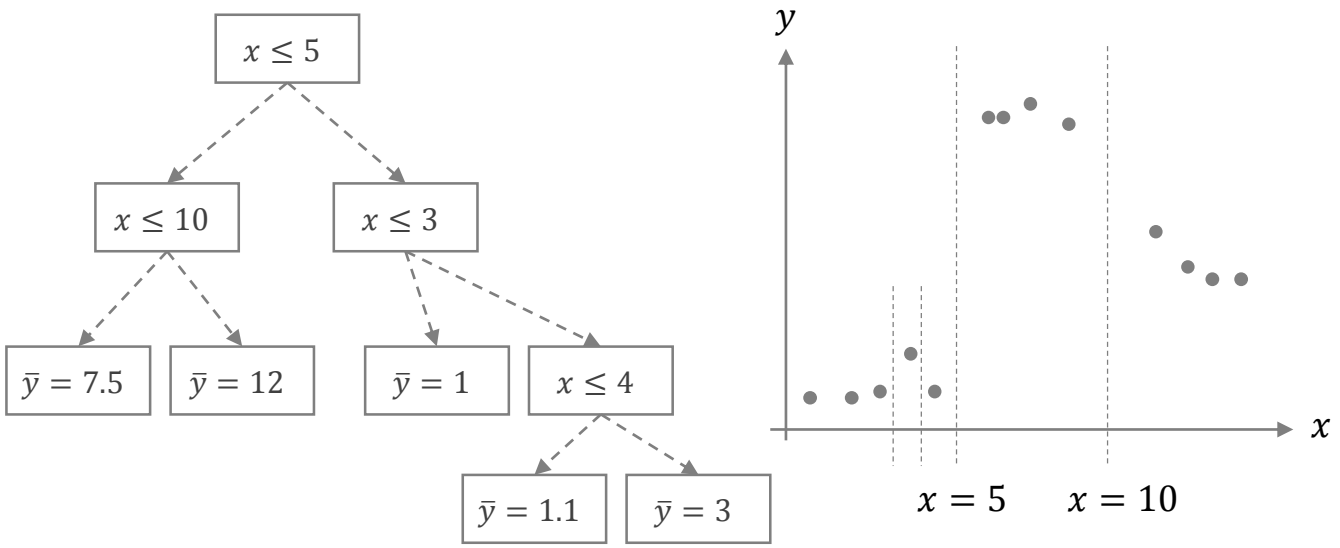
$$IG(N_1, N_2) = L(N_1 + N_2) - L(N_1) - N(N_2)$$





Decision Tree for Regression

- 회귀모델에서도 과적합이 발생할 수 있으며 분류문제와 동일한 regularization 이 이용됩니다.



Decision Tree for Regression

- Sub-linear 데이터를 제대로 학습하지 못합니다.
  - 부분적으로 x, y 가 선형 관계에 있지만 의사결정나무는 $\bar{y}$ 를 회귀값으로 이용합니다.