

데이터마케팅팀

4팀

황유나
문서영
김지현
위재성
이진모

CONTENTS

1. Tree-based Models

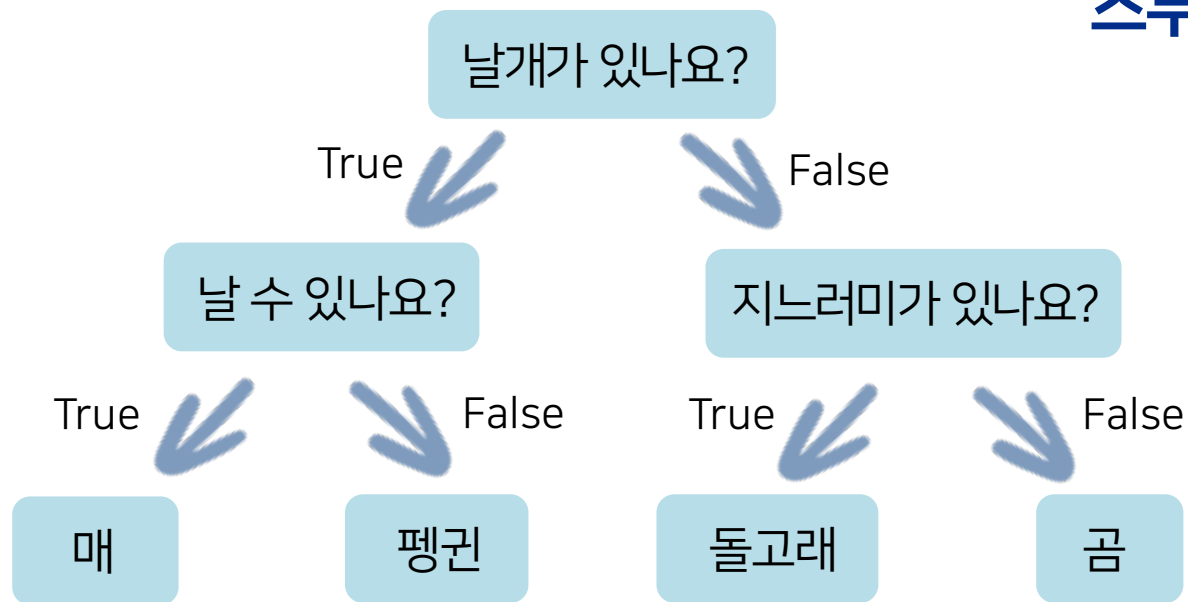
2. Ensemble Methods

1

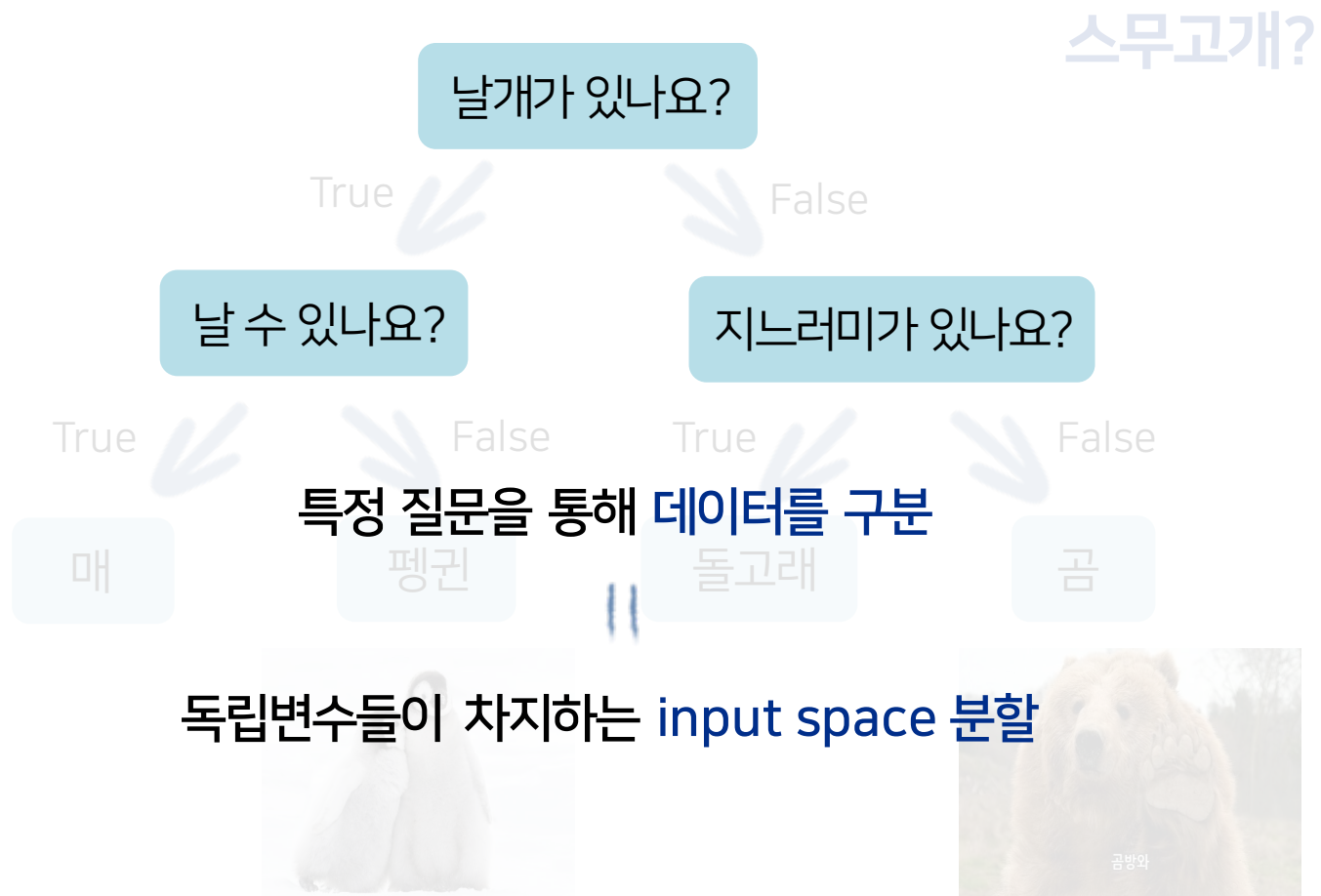
Tree-based Models

의사 결정 나무 (Decision Tree)

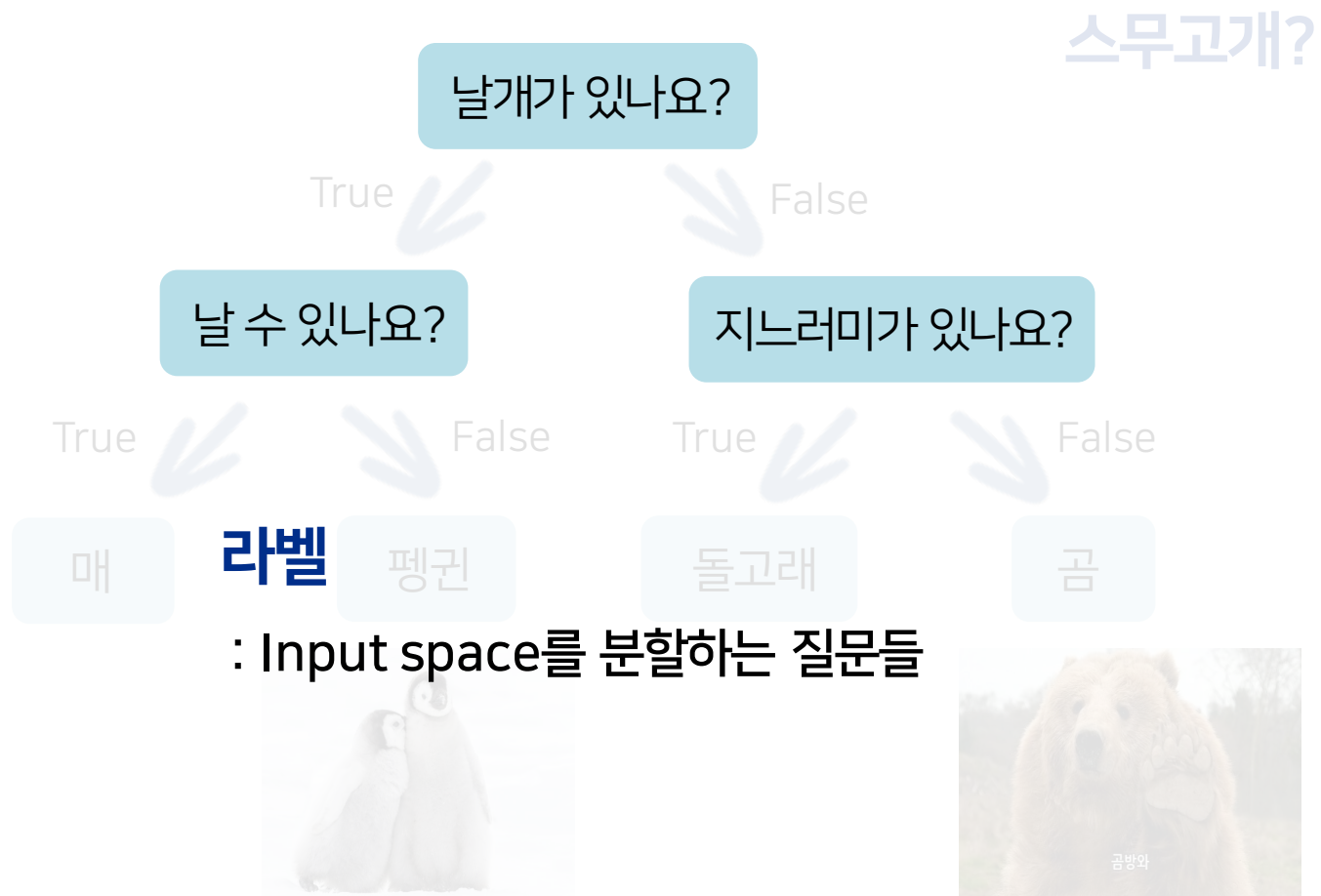
스무고개?



의사 결정 나무 (Decision Tree)



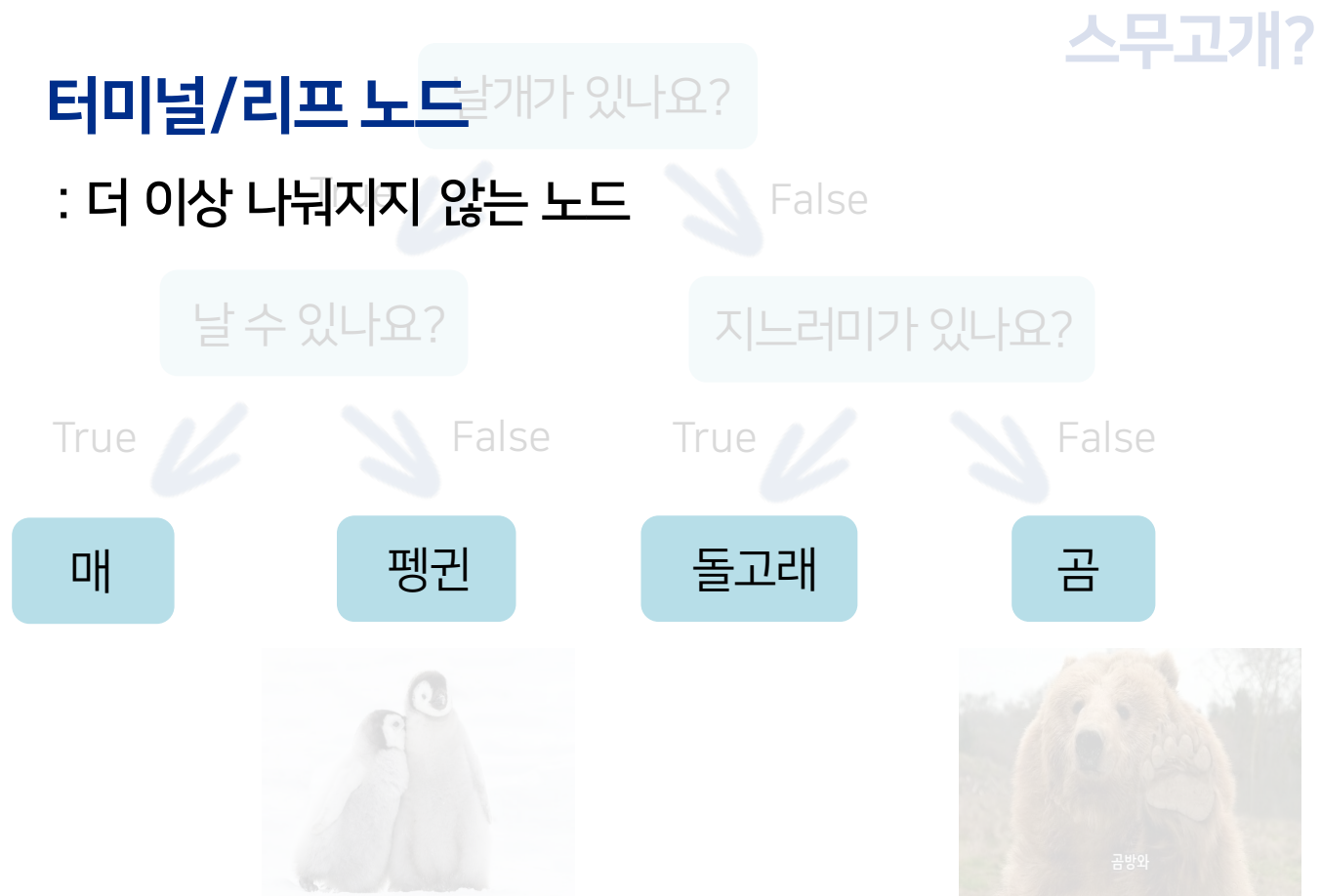
의사 결정 나무 (Decision Tree)



의사 결정 나무 (Decision Tree)

터미널/리프 노드

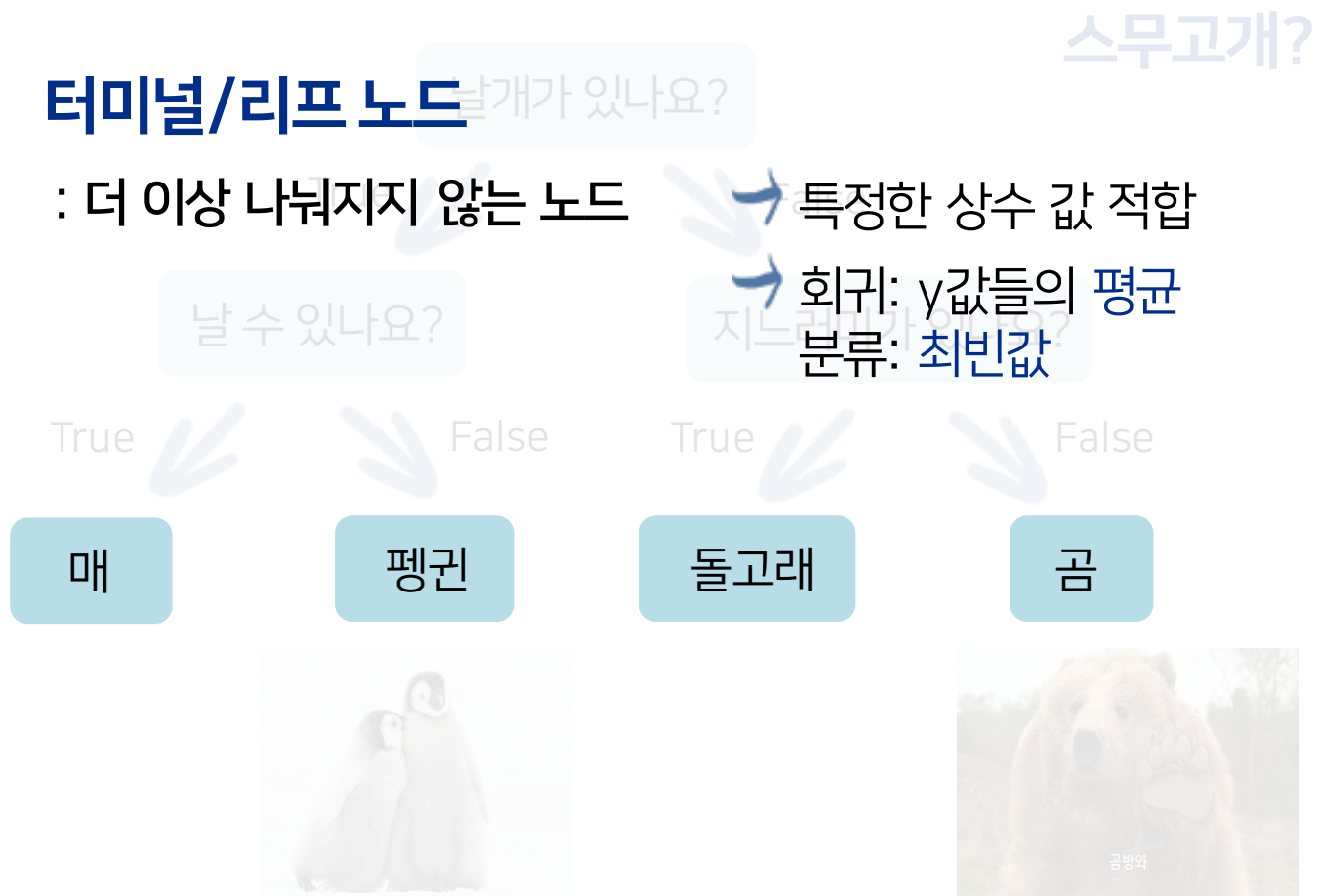
: 더 이상 나뉘지지 않는 노드



의사 결정 나무 (Decision Tree)

터미널/리프 노드

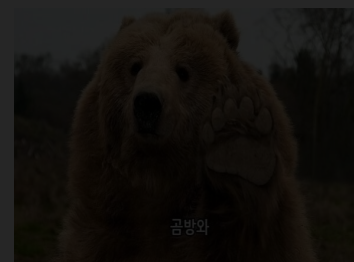
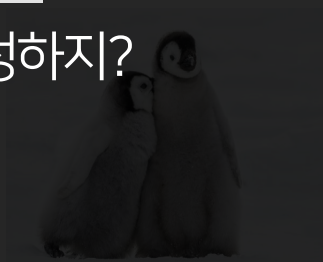
: 더 이상 나뉘지지 않는 노드



의사 결정 나무 (Decision Tree)



어떻게 트리를 생성하지?



스무고개?

날개가 있나요?

True

False

날개 있나요?

지느러미가 있나요?

False

False

펭귄

돌고래

곰

어떤 질문을 해야 할까?
어떻게 상수 값을 도출할 수 있을까?

CART(Classification And Regression Trees)

- : 트리 생성 알고리즘 중 하나
- 이진 트리로만 이루어져 있다.

? 터미널 노드에 적합한 상수 C 는 ?
어떻게 도출할 수 있을까?

CART(Classification And Regression Trees)

- : 트리 생성 알고리즘 중 하나
- 이진 트리로만 이루어져 있다.

? 터미널 노드에 적합한 상수 C 는
어떻게 도출할 수 있을까? ?

$$f(X) = \sum_{m=1}^M c_m I(X \in R_m)$$

when $\bigcup_{m=1}^M R_m = \mathbb{R}^p$ and $R_m \cap R_p = \emptyset$ (non-overlapping, distinct)

and \mathbb{R}^p is a p -dimensional input space

CART(Classification And Regression Trees)

- : 트리 생성 알고리즘 중 하나
- 이진 트리로만 이루어져 있다.

? 터미널 노드에 적합한 상수 C 는
어떻게 도출할 수 있을까? ?

$$f(X) = \sum_{m=1}^M c_m I(X \in R_m)$$

회귀: y 값들의 평균
분류: 최빈값

when $\bigcup_{m=1}^M R_m = \mathbb{R}^p$ and $R_m \cap R_p = \emptyset$ (non-overlapping, distinct)

and \mathbb{R}^p is a p -dimensional input space

1) Decision Tree Regressor

- 회귀 문제
- 실제 값과 예측 값의 오차 사용
- $RSS(\text{잔차제곱합})$ 을 줄여 나가는 방향으로 작동
- 탐욕적 알고리즘
: 매 순간마다 최적의 선택

$$\min_{c_m} \sum_{i=1}^N \{y_i - f(x_i)\}^2 = \min_{c_m} \sum_{i=1}^N [y_i - \sum_{m=1}^M c_m I(X \in R_m)]^2$$

1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

독립변수가 한 개일 때

1. 독립변수에서 기준점을 랜덤 선택
2. 기준점을 바탕으로 RSS 계산
3. RSS값이 더 이상 줄어들지 않을 때까지 1-2 반복
4. 가장 작은 RSS값을 도출하는 독립변수를 기준으로 선택

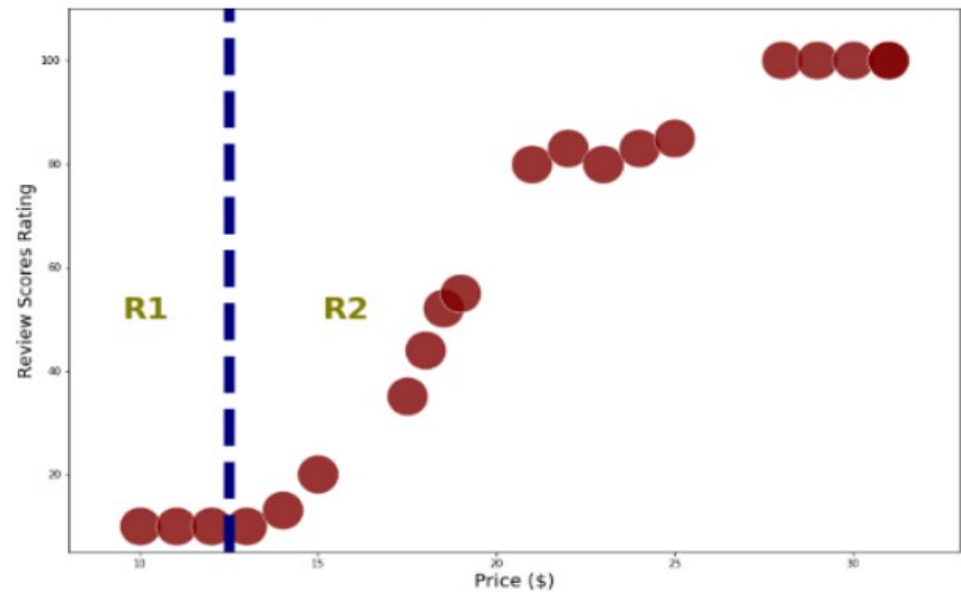
1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

⇒ 랜덤 기준1 Price: 13



1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

⇒ 랜덤 기준1 Price: 13



1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

⇒ 랜덤 기준1 Price: 13



$$RSS = (10 - 10)^2 + (10 - 10)^2 + \dots + (100 - 65)^2 + (100 - 65)^2 = \mathbf{15762.0}$$

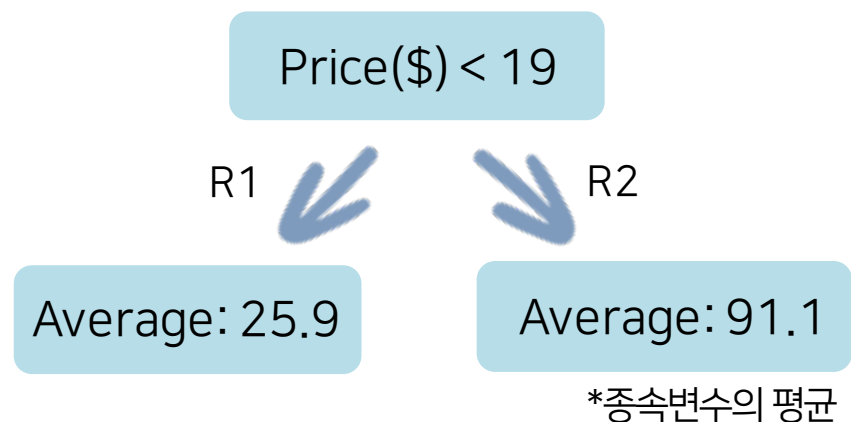
1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

⇒ 랜덤 기준2 Price: 19



$$RSS = (10 - 25.9)^2 + (10 - 25.9)^2 + \dots + (100 - 91.1)^2 + (100 - 91.1)^2 = \mathbf{3873.79}$$

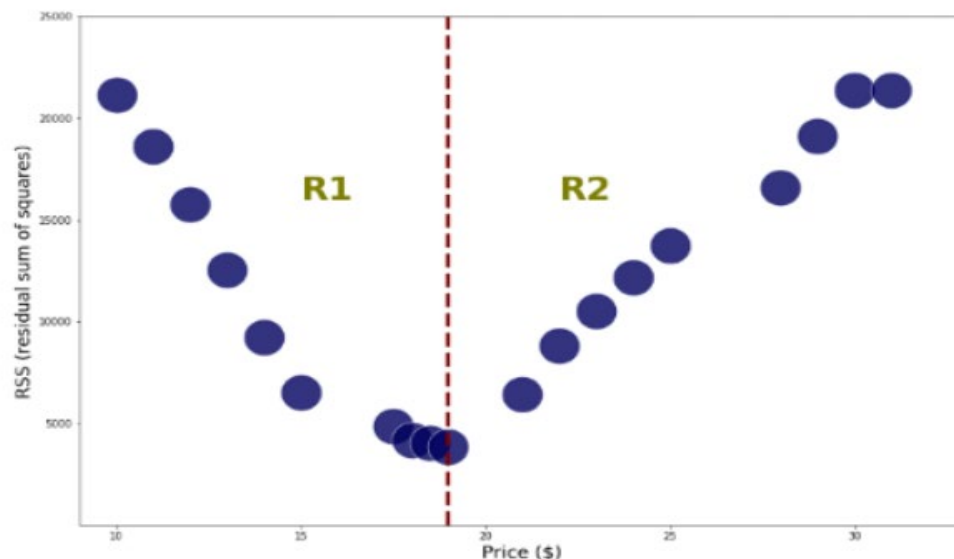
1) Decision Tree Regressor

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55

⋮

종속변수

➡ 랜덤 기준2 Price: 19



Price ≤ 19 를 기준으로 삼았을 때
RSS가 가장 작다

➡ 최종선택

1) Decision Tree Regressor

독립변수가 추가된다면,,?

	Price (\$)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55



	Price (\$)	Cleaning fee(\$)	License	Review Scores Rating
0	10.0	0	0	10
1	11.0	1	0	10
2	12.0	2	1	10
3	13.0	3	0	10
4	14.0	4	1	13
5	15.0	5	0	20
6	17.5	7.5	1	35
7	18.0	8	1	44
8	18.5	8.5	1	52
9	19.0	9	0	55

⋮

⋮

1) Decision Tree Regressor

	Price (\$)	Cleaning fee(\$)	License	Review Scores Rating
0	10.0	0	0	10
1	11.0	1	0	10
2	12.0	2	1	10
3	13.0	3	0	10
4	14.0	4	1	13
5	15.0	5	0	20
6	17.5	7.5	1	35
7	18.0	8	1	44
8	18.5	8.5	1	52
9	19.0	9	0	55

⋮

종속변수

독립변수가 여러 개

1. 독립변수 별로 가장 작은 RSS 도출하는 최적 기준 선택
2. 선택된 최적 기준들 중에서 가장 작은 RSS를 갖는 기준을 선택

1) Decision Tree Regressor

	Price (\$)	Cleaning fee(\$)	License	Review Scores Rating
0	10.0	0	0	10
1	11.0	1	0	10
2	12.0	2	1	10
3	13.0	3	0	10
4	14.0	4	1	13
5	15.0	5	0	20
6	17.5	7.5	1	35
7	18.0	8	1	44
8	18.5	8.5	1	52
9	19.0	9	0	55

⋮

종속변수



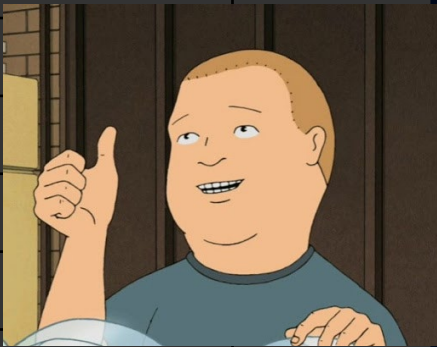
독립변수가 여러 개

Price 최적기준: 19
(RSS = 3873.79)

Cleaning_fee 최적기준: 9
(RSS = 64214.8)

License 최적기준: 0
(RSS = 11685.5)

1) Decision Tree Regressor

	Price (\$)	Cleaning fee(\$)	License	Review Scores Rating
0	10.0	0	0	10
1	11.0	1	0	10
2	12.0		0	10
3	13.0		0	10
4	14.0		0	13
5	15.0		0	20
6	17.5	7.5	1	35
7	18.0	8	1	44
8	18.5	8.5	1	52
9	19.0	9	0	55

⋮

종속변수

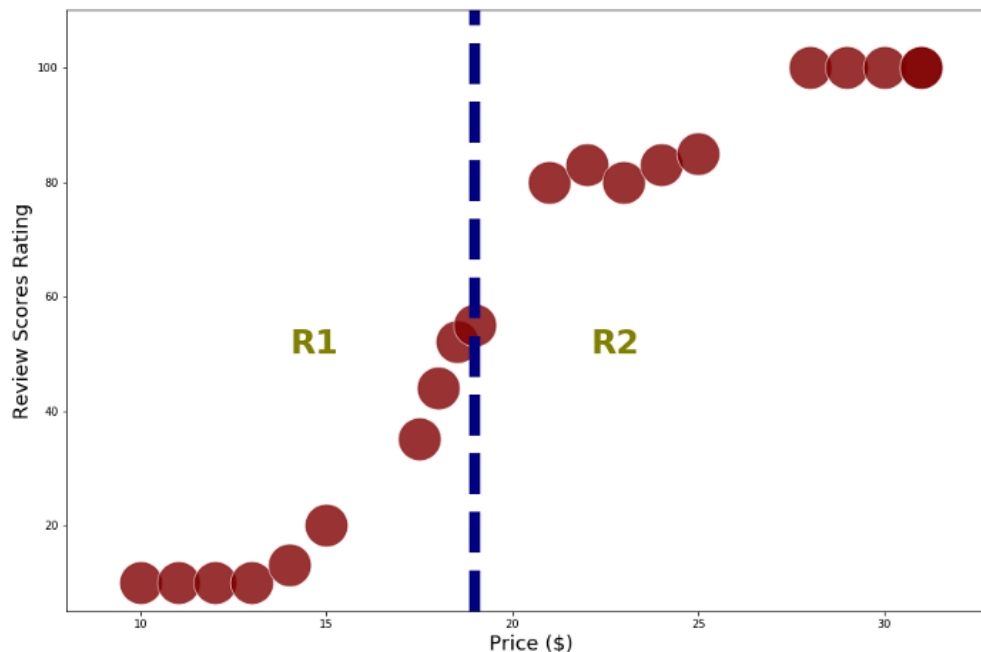
⇒ 독립변수가 여러 개

Price 최적기준: 19
(RSS = 3873.79)Cleaning_fee 최적기준: 9
(RSS = 64214.8)

⇒ 가장 최적!

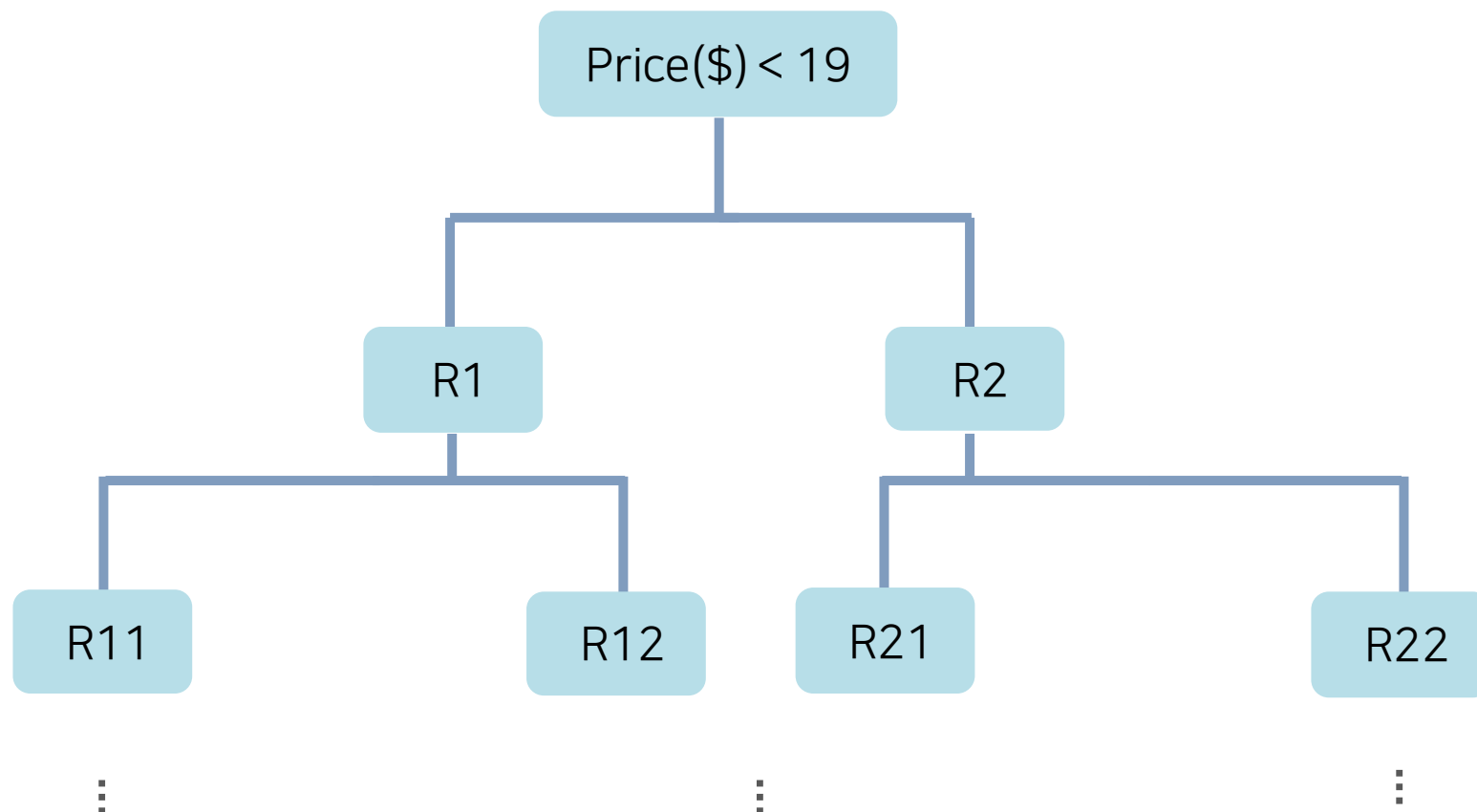
License 최적기준: 0
(RSS = 11685.5)

1) Decision Tree Regressor



→ R1, R2로 나뉜 input space에서
각각 최소 RSS 산출조건 다시 찾음

1) Decision Tree Regressor



→ RSS가 더 이상 작아지지 않을 때까지 반복

1) Decision Tree Regressor

수식으로 요약하자면,,

Find j and s satisfying

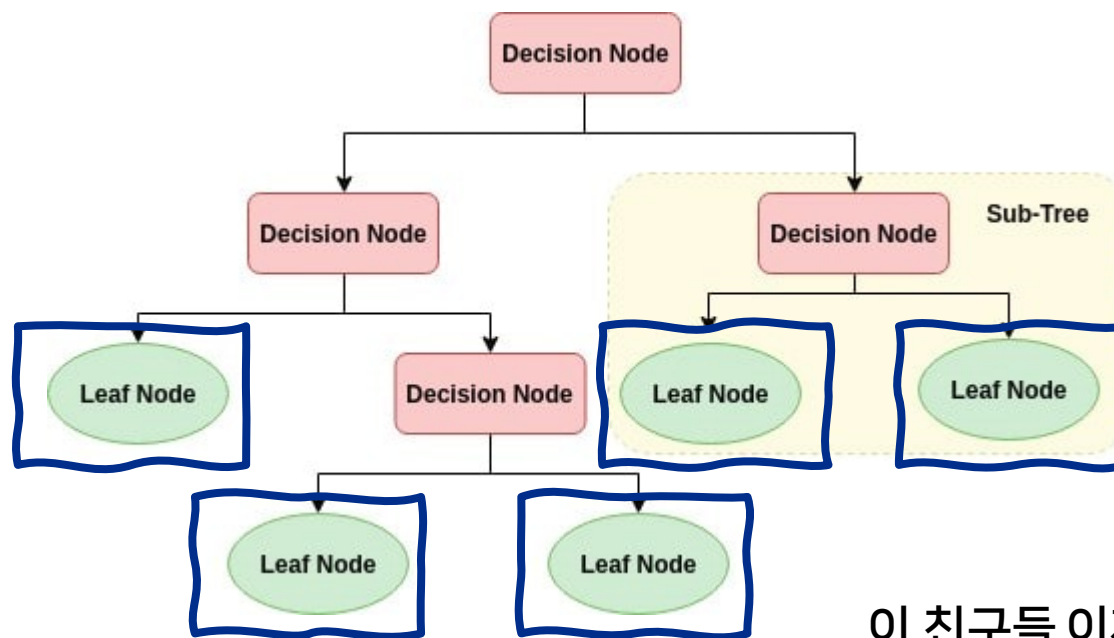
$$\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2$$

⇒ j : 특정 독립변수
 s : 분기의 기준이 되는 관측치의 종속변수 값

⇒ c_1, c_2 : terminal node의 종속변수 평균값

2) Decision Tree Classifier

분류모델에서 termial node의 적합값은 해당 분할에 포함된 범주들의 **최빈값**이다.



이 친구들 이제
Mean 아니고 Mode야!!

2) Decision Tree Classifier

비슷한 관측치들끼리 몰려 있는 형태가 이상적

회귀 모델

RSS가 줄어드는 방향으로 트리가 나뉘도록 한다

분류 모델

불순도가 줄어들도록 트리가 나뉘도록 하자

2) Decision Tree Classifier

비슷한 관측치들끼리 몰려 있는 형태가 이상적

? 불순도가 줄어듦

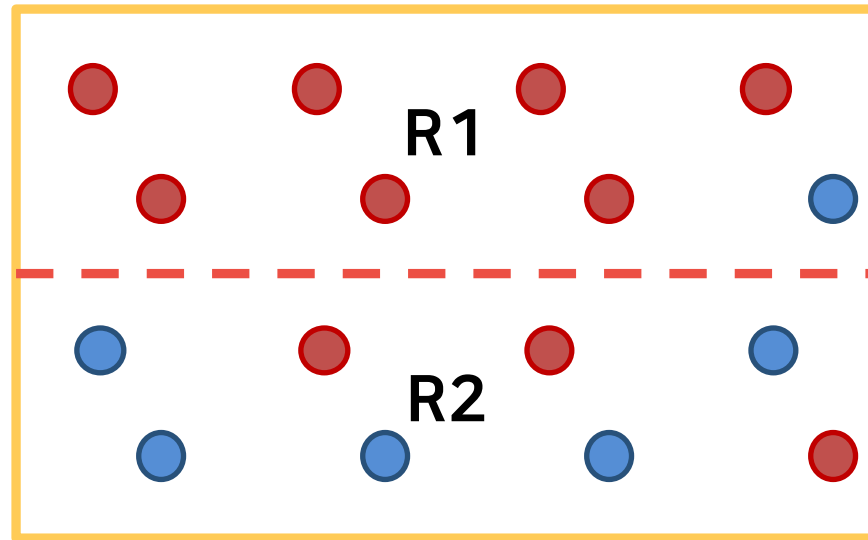
회귀 모델

RSS가 줄어드는 방향으로 트리가 나뉘도록 한다
: 관측치들이 어떤 범주에 속할 지에 대한
불확실성이 줄어드는 것

분류 모델

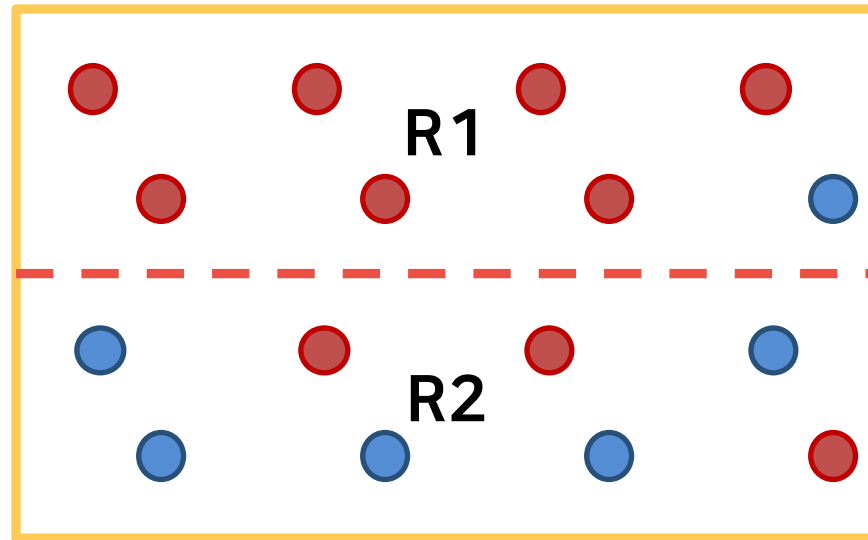
불순도가 줄어들도록 트리가 나뉘도록 하자
관측치들이 각자의 범주에 따라 적절히 분류 되었음을 의미

2) Decision Tree Classifier



예측값은 각 카테고리의 최빈값을 차지하는 범주를 선택해야 하므로 R1은 빨강, R2는 파랑

2) Decision Tree Classifier



R2에는 빨간 구슬과 파란 구슬이 적절히 분류되었다고 볼 수 없음

즉, R2의 불순도 > R1의 불순도

2) Decision Tree Classifier

분류가 잘 되었는지,
그러니까 분할된 공간의 '순도'는 어떻게 계산할 수 있을까?

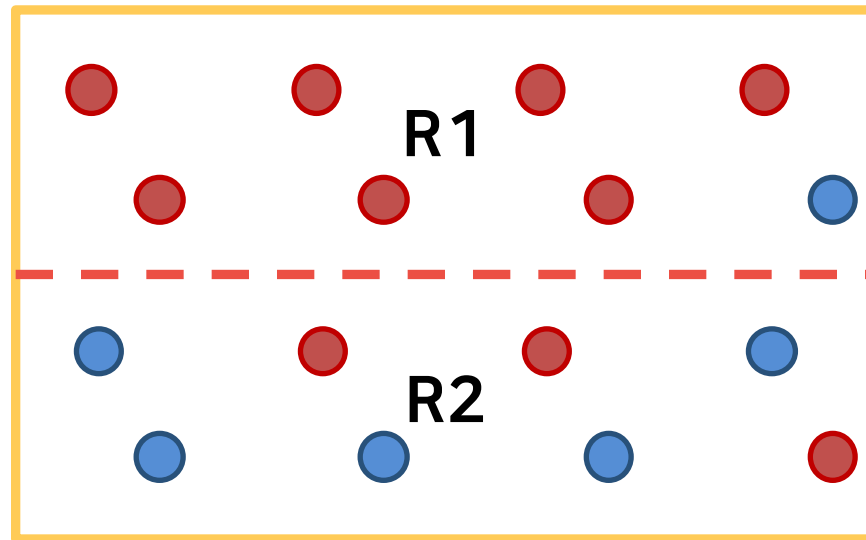
물리학에서의 '엔트로피'의 개념을 차용해보자



Dr. Strange의 엔트로피가 증가하고 있다.

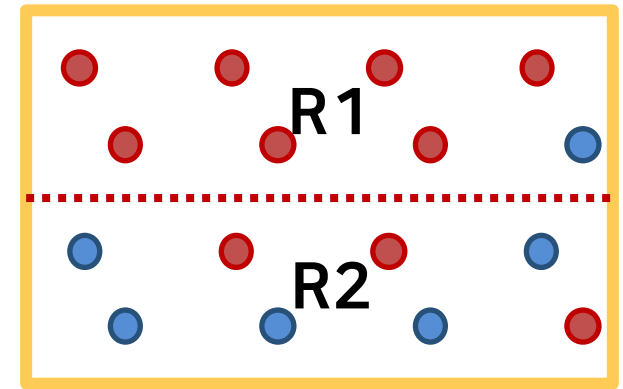
2) Decision Tree Classifier

$$Entropy(X) = - \sum_{k=1}^m p_k \log_2(p_k)$$



2) Decision Tree Classifier

$$\text{Entropy}(X) = - \sum_{k=1}^m p_k \log_2(p_k)$$



분할 전 Entropy:

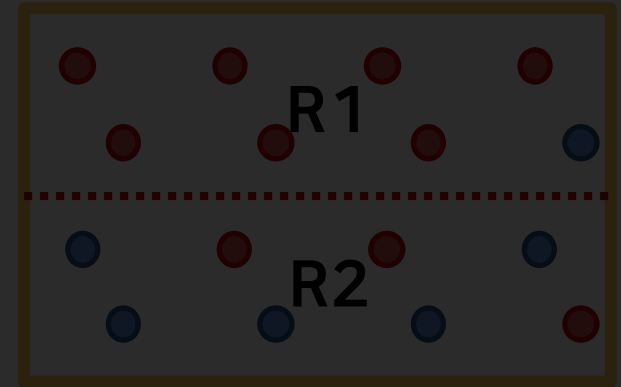
$$-\frac{10}{16} \log_2\left(\frac{10}{16}\right) - \frac{6}{16} \log_2\left(\frac{6}{16}\right) \approx \mathbf{0.95}$$

분할 후 Entropy:

$$0.5 \times \left(-\frac{7}{8} \log_2\left(\frac{7}{8}\right) - \frac{1}{8} \log_2\left(\frac{1}{8}\right)\right) + 0.5 \times \left(-\frac{3}{8} \log_2\left(\frac{3}{8}\right) - \frac{5}{8} \log_2\left(\frac{5}{8}\right)\right) \approx \mathbf{0.75}$$

2) Decision Tree Classifier

$$Entropy(X) = - \sum_{k=1}^m p_k \log_2(p_k)$$



트리를 분할한 후,
 분할 전 Entropy: 엔트로피가 약 0.2정도 감소

$$-\frac{10}{16} \log_2\left(\frac{10}{16}\right) - \frac{6}{16} \log_2\left(\frac{6}{16}\right) \approx 0.95$$

분할 후 Entropy: \Rightarrow 불순도가 감소하는 방향으로 분류 진행

$$0.5 \times \left(-\frac{7}{8} \log_2\left(\frac{7}{8}\right) - \frac{1}{8} \log_2\left(\frac{1}{8}\right)\right) + 0.5 \times \left(-\frac{3}{8} \log_2\left(\frac{3}{8}\right) - \frac{5}{8} \log_2\left(\frac{5}{8}\right)\right) \approx 0.75$$

3) Tree-based Model에서 과적합 피하기

복잡도 (Complexity): 트리모델 > 선형회귀 모델

? Why

분류가 진행되며 관측치들이 나뉘는 과정

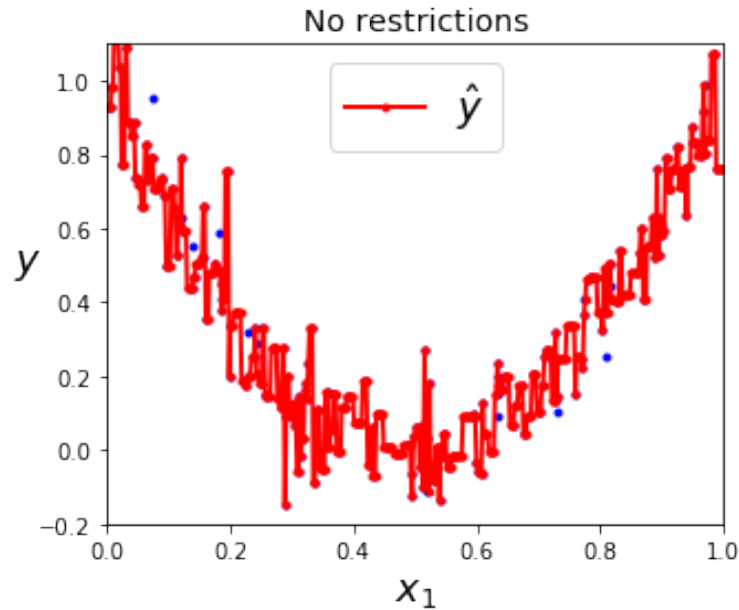
||

모델의 파라미터가 늘어나는 효과

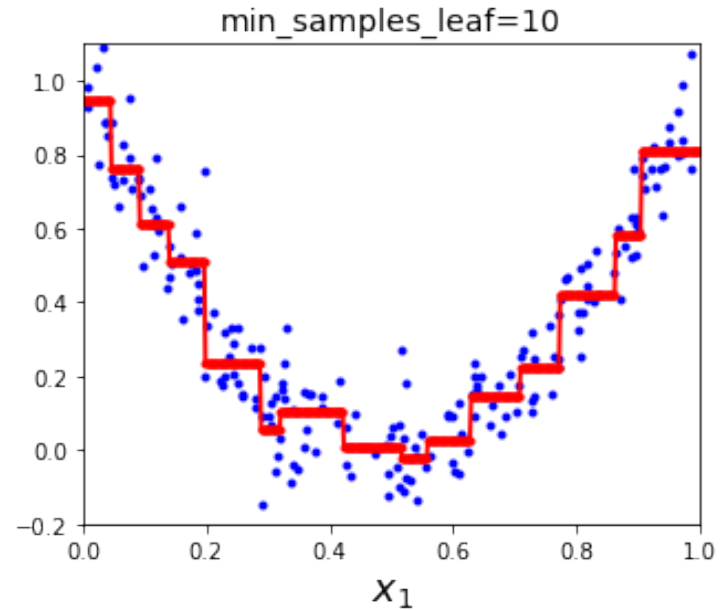


하이퍼 파라미터 값에 따라 모델 모양이 달라짐

3) Tree-based Model에서 과적합 피하기



[제약조건 없음]



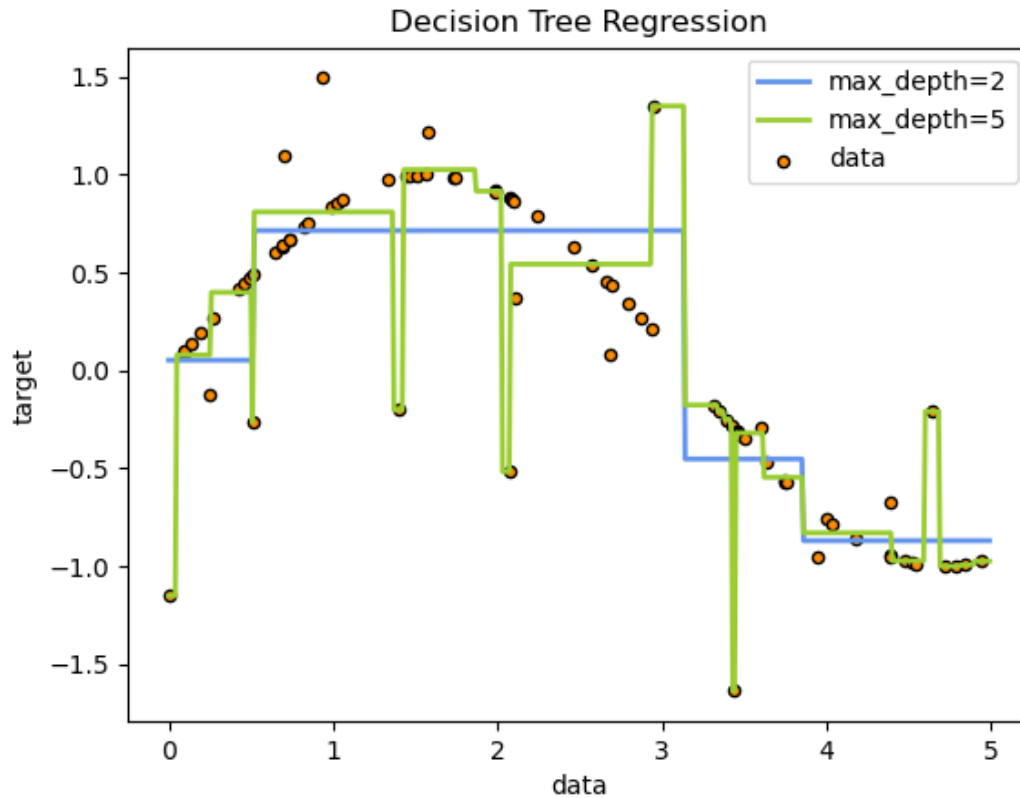
[min_samples_leaf=10]

복잡한 정도가 서로 다르다!

3) Tree-based Model에서 과적합 피하기

트리의 깊이를 의미하는

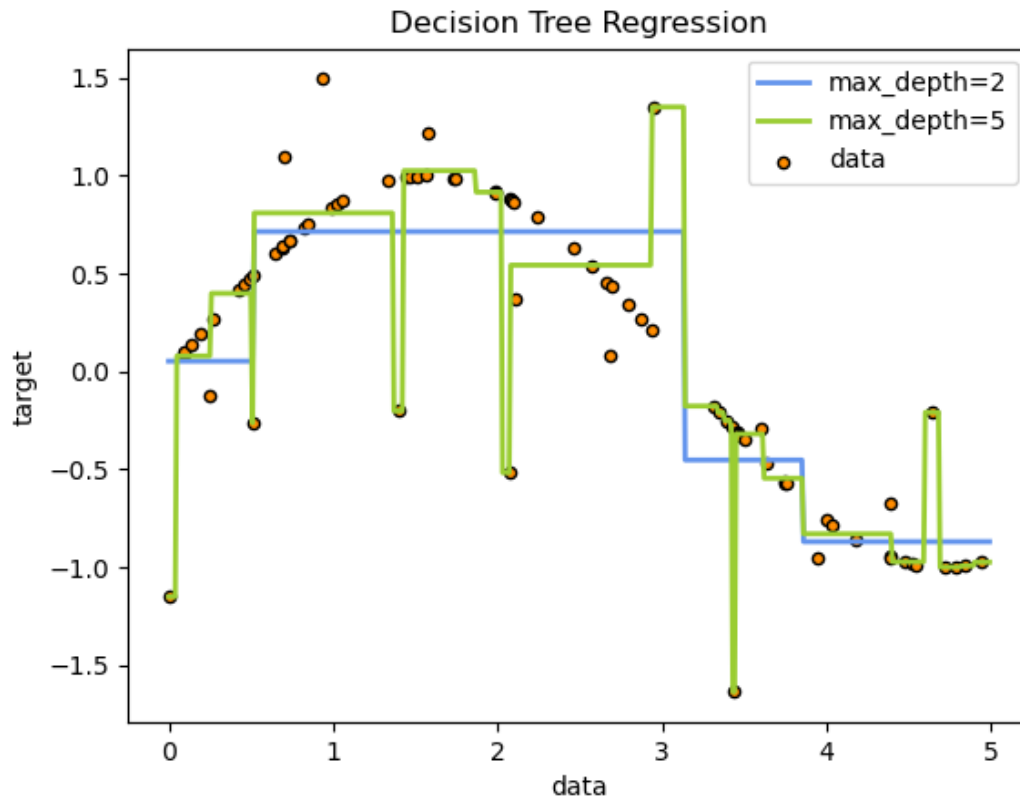
'max_depth' 파라미터에 다른 값을 준 경우



3) Tree-based Model에서 과적합 피하기

트리의 깊이를 의미하는

'max_depth' 파라미터에 다른 값을 준 경우



모델 간 분산이 크다

3) Tree-based Model에서 과적합 피하기

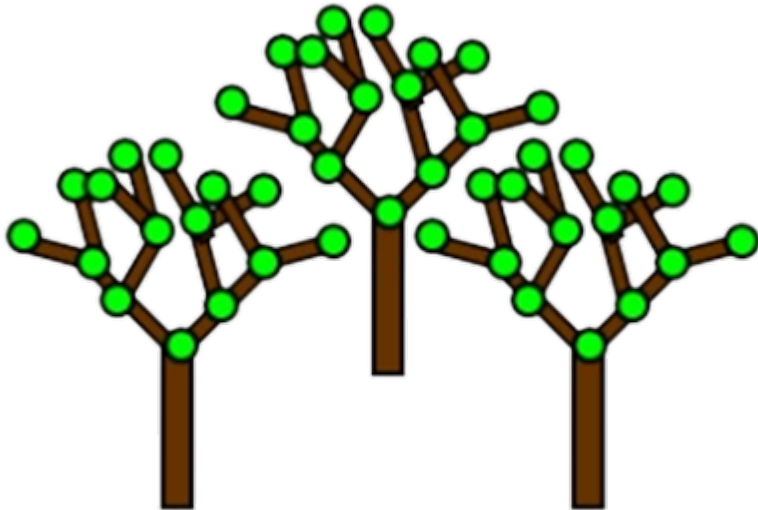
모델 간 분산이 큰 경우를 해결해주자

“

Idea

여러 개의 트리모델을 적합해
이들을 종합적으로 평가해보자!

”



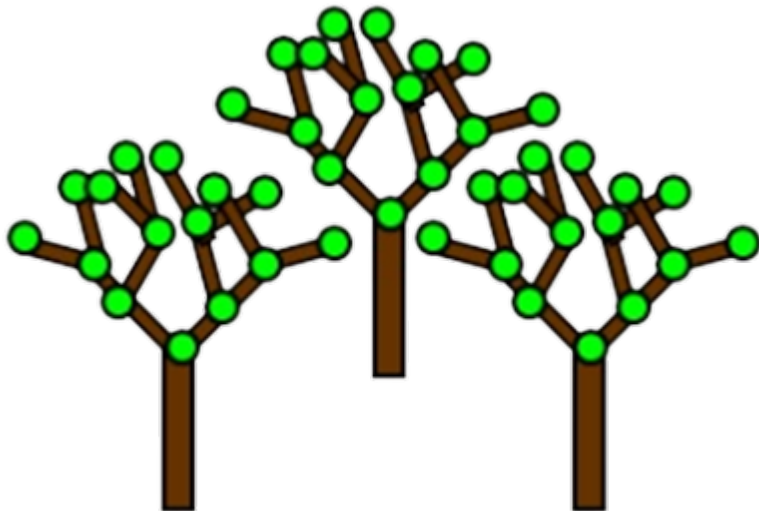
3) Tree-based Model에서 과적합 피하기

모델 간 분산이 큰 경우를 해결해주자



Idea

여러 개의 트리모델을 적합해
이들을 종합적으로 평가해보자!



랜덤 포레스트 모델링 기법

자세한 설명은 뒤에서..

3) Tree-based Model에서 과적합 피하기

너무 많이 나뉜 트리 모델 \equiv 너무 많은 변수가 사용된 모델

3) Tree-based Model에서 과적합 피하기

너무 많이 나뉜 트리 모델 = 너무 많은 변수가 사용된 모델



관측치들이 매우 자세하게 분류 됨

3) Tree-based Model에서 과적합 피하기

너무 많이 나뉜 트리 모델 = 너무 많은 변수가 사용된 모델

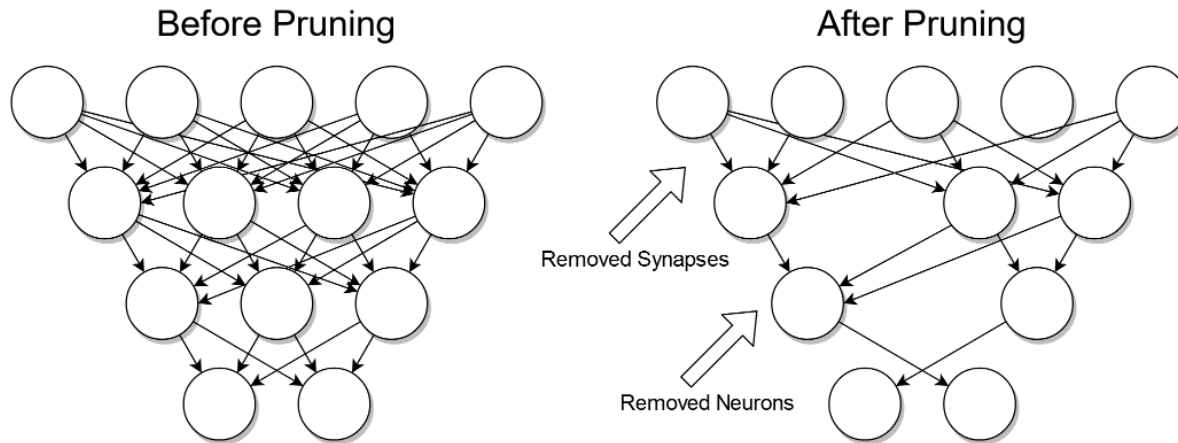


관측치들이 매우 자세하게 분류 됨



Overfit할 가능성이 높음

3) Tree-based Model에서 과적합 피하기



Pruning

: 지나치게 자세히 분류된 가지들은 다시 적절히 합쳐 **Overfit**을 방지
&
새로운 Data에 **탄력적으로** 반응하는 모델로 만듦

2

Ensemble Model

앙상블 모델(Ensemble Model)

지금까진 단일한 트리 기반 모델에 관한 내용이었다!

앙상블 기법은 여러 모델을 독립적으로 학습시킨 후,
각 모델의 결과를 조합하여 최종 결과를 생성

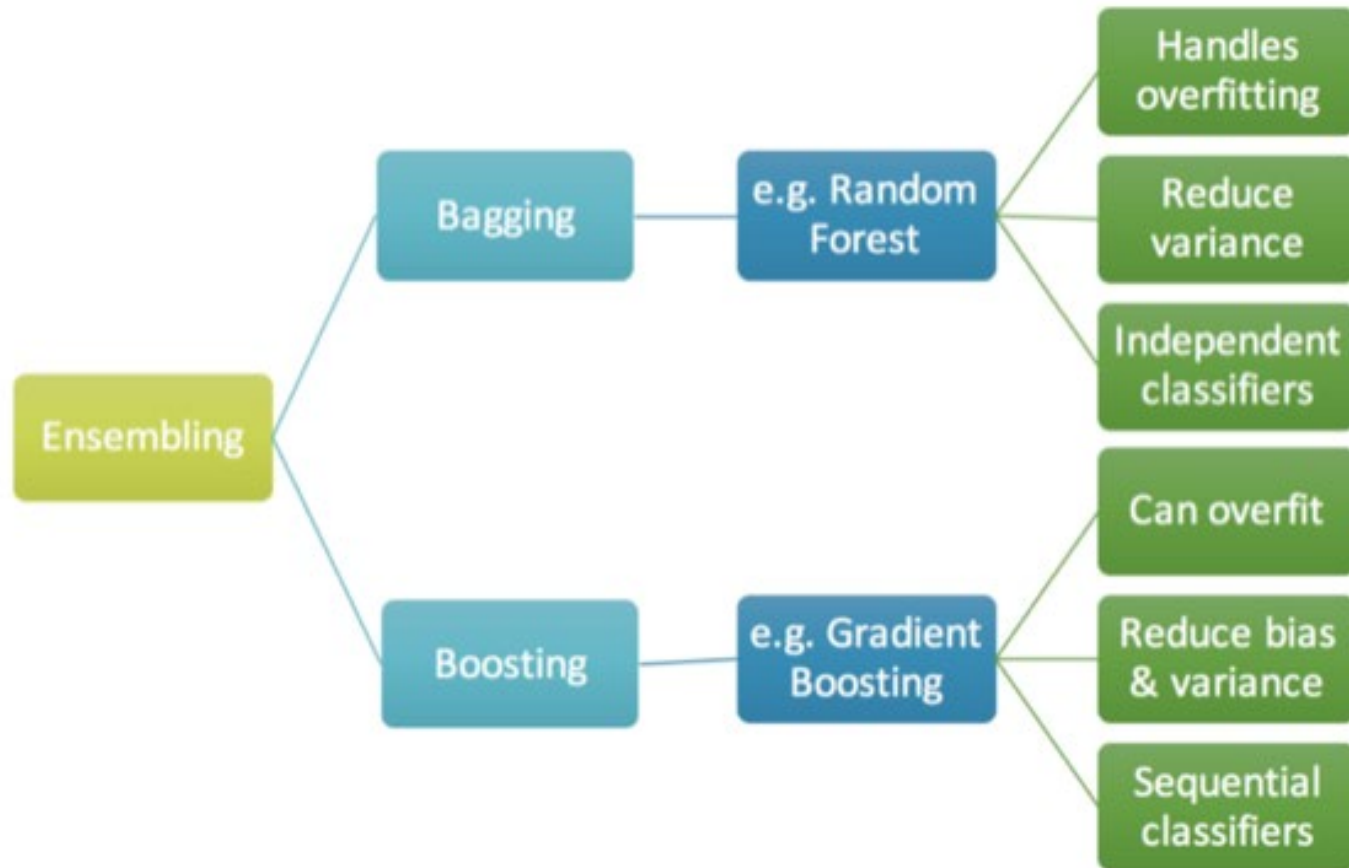
앙상블 모델(Ensemble Model)

지금까진 단일한 트리 기반 모델에 관한 내용이었다!

앙상블 기법은 여러 모델은 독립적으로 학습시킨 후,
각 모델의 결과를 조합하여 최종 결과를 생성
여러 개의 **약한 학습기**를 결합하여
더 좋은 성능을 내는 **강한 학습기**를 만든다

앙상블 모델(Ensemble Model)

앙상블 기법은 아래와 같은 갈래로 분류



앙상블 모델(Ensemble Model)

앙상블 기법에서 가장 먼저 이루어져야 할 작업은 어떤 회귀/분류기를 사용할지 선택하는 것이다.

앙상블 모델(Ensemble Model)

앙상블 기법에서 가장 먼저 이루어져야 할 작업은 어떤 회귀/분류기를 사용할지 선택하는 것이다.

Homogeneous Ensemble

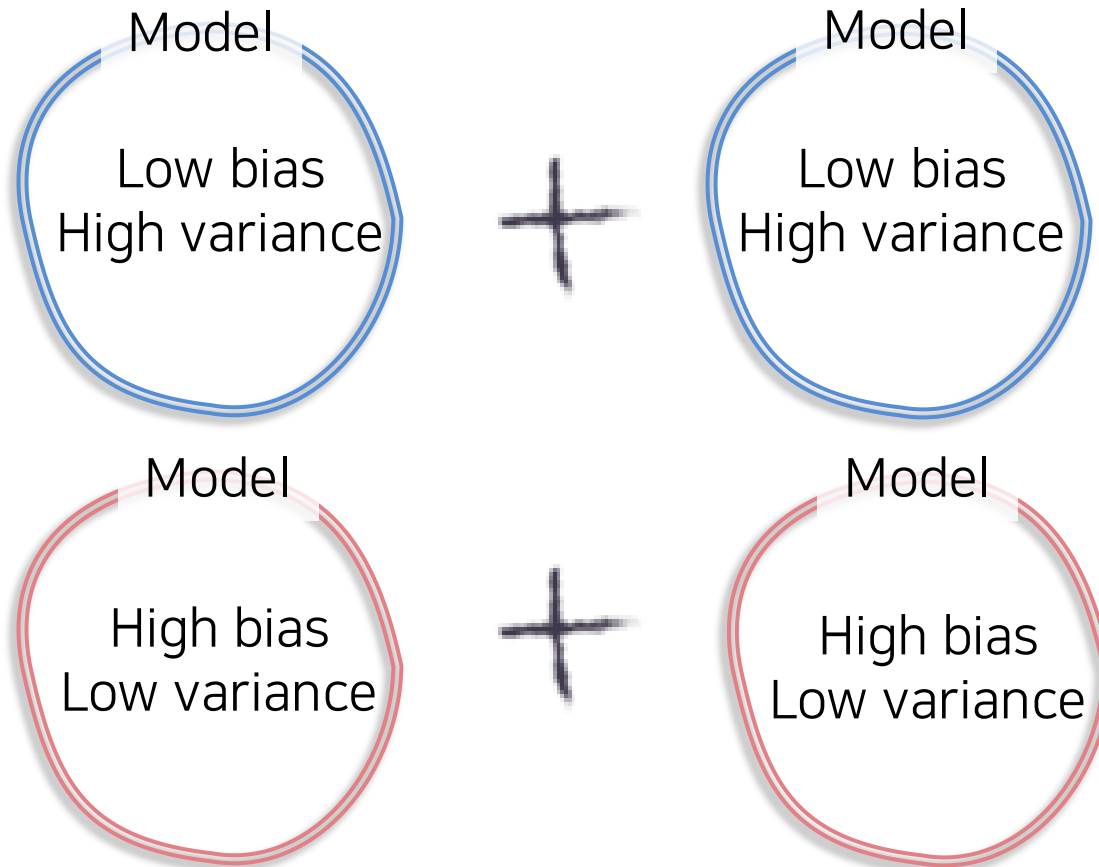
: 한종류의 model을 여러개결합(주로사용됨)

Heterogeneous Ensemble

: 동일한데이터셋에서로다른종류의model을결합

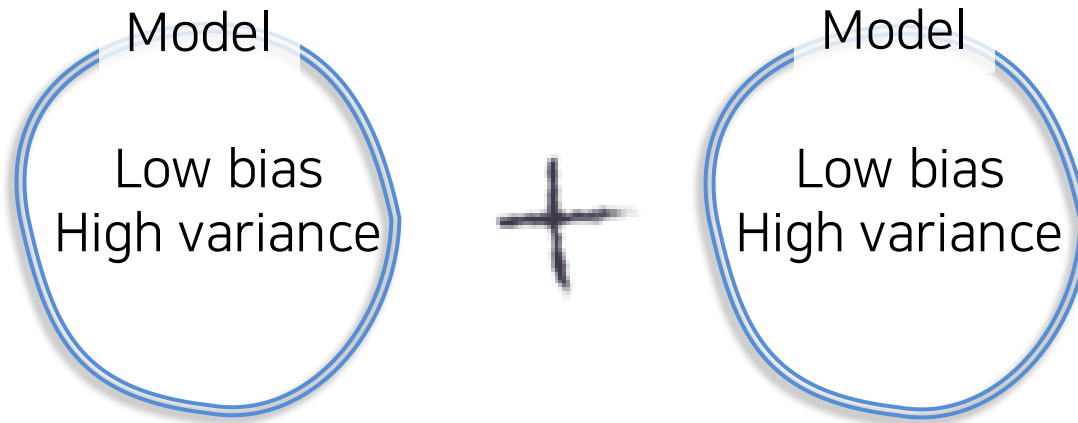
앙상블 모델(Ensemble Model)

앙상블을 위한 estimator를 선정할 때 **주의해야 할 점**



앙상블 모델(Ensemble Model)

앙상블을 위한 estimator를 선정할 때 **주의해야 할 점**



모델의 **분산**을 크게 줄여
더욱 강건한 모델을 설계해 성능 개선

앙상블 모델(Ensemble Model)

앙상블을 위한 estimator를 선정할 때 **주의해야 할 점**



모델의 **편향**을 크게 줄여
더욱 강건한 모델을 설계해 성능 개선

배깅(Bagging)

배깅(Bagging)은 Bootstrap+Aggregating의 의미

부트스트랩 방법을 통해
샘플링한 표본들을 바탕으로 모델링을 실시하는 기법

배깅(Bagging)

배깅(Bagging)은 Bootstrap+Aggregating의 의미

부트스트랩 방법을 통해
샘플링한 표본들을 바탕으로 모델링을 실시하는 기법



부트스트랩 방법이란?

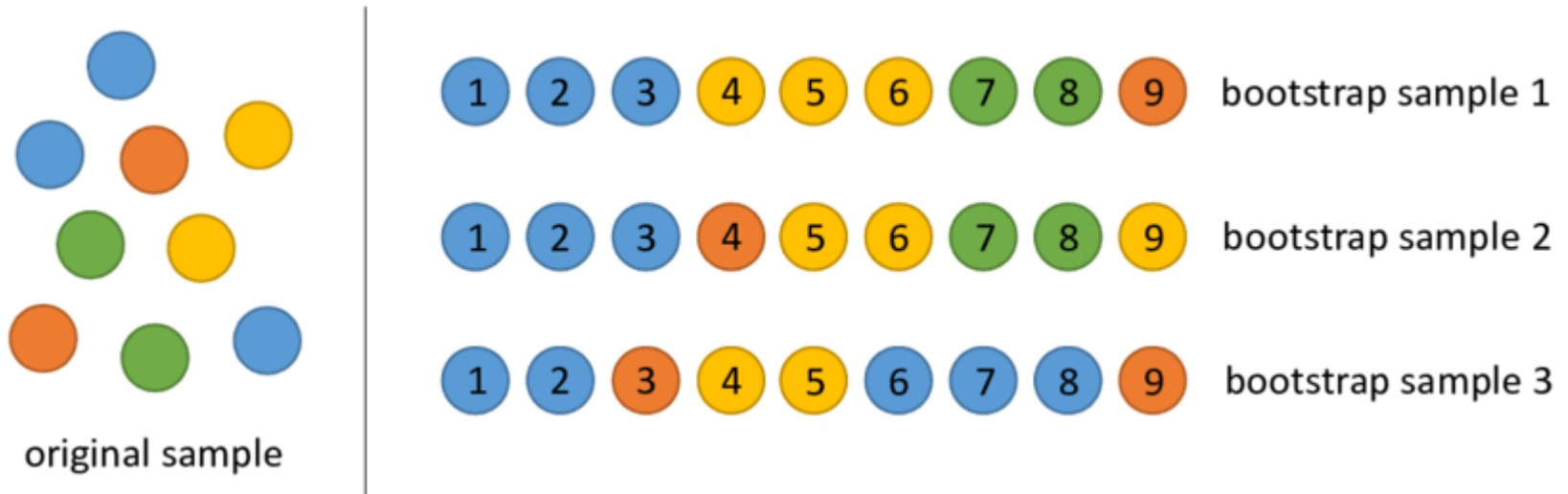
배깅(Bagging)



Bootstrap Sampling은
복원 추출이 가능한 랜덤 샘플링

배깅(Bagging)

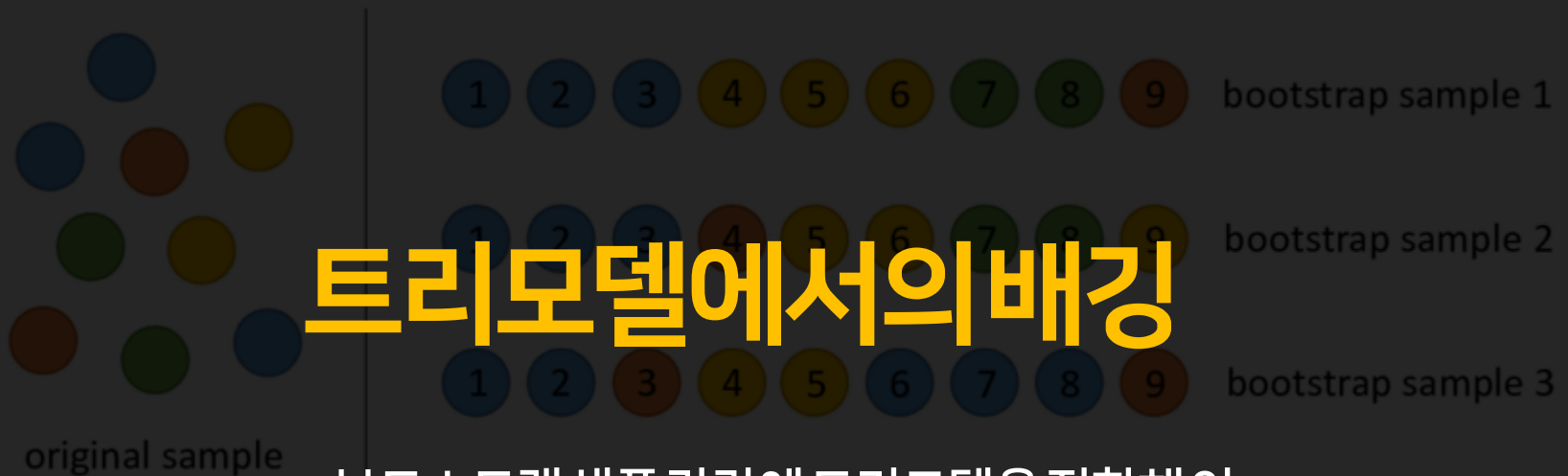
Bootstrap sampling을 하는 이유?



샘플 데이터셋을 각각 다르게 해
모델에 적합 시켰을 때 발생할 수 있는 **모델 variance**를 최소화

배깅(Bagging)

Bootstrap sampling을 하는 이유?

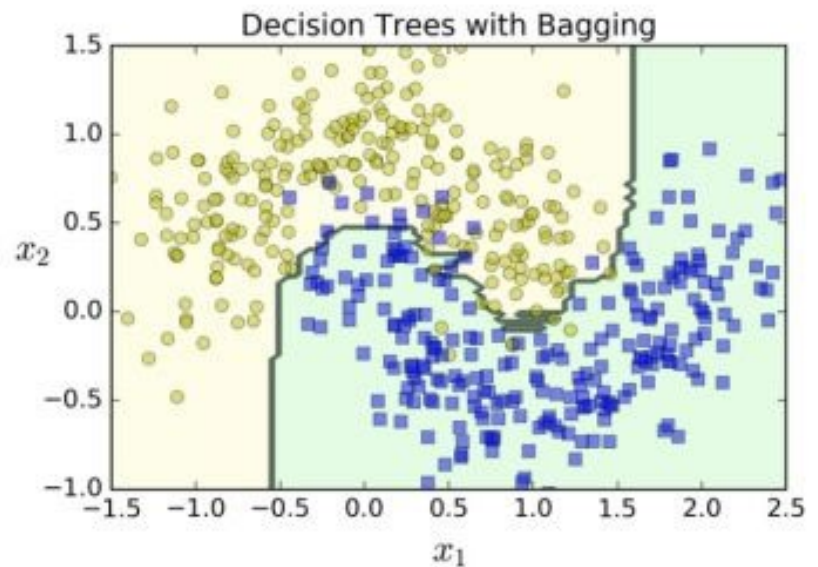
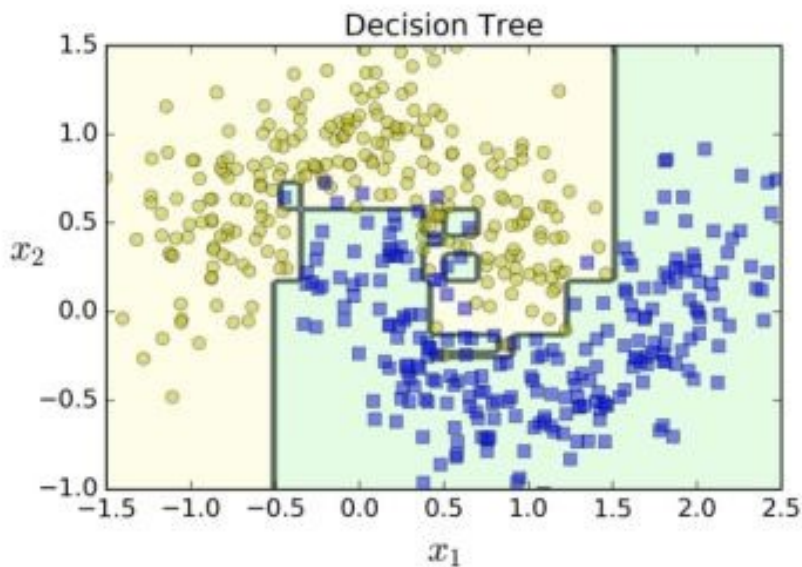


: 부트스트랩 샘플 각각에 트리모델을 적합해 이들의 평균(또는 최빈값)을 예측값으로 생각

샘플 데이터셋이 각각 다르게 해
모델에 적합 시켰을 때 발생할 수 있는 모델 variance를 최소화

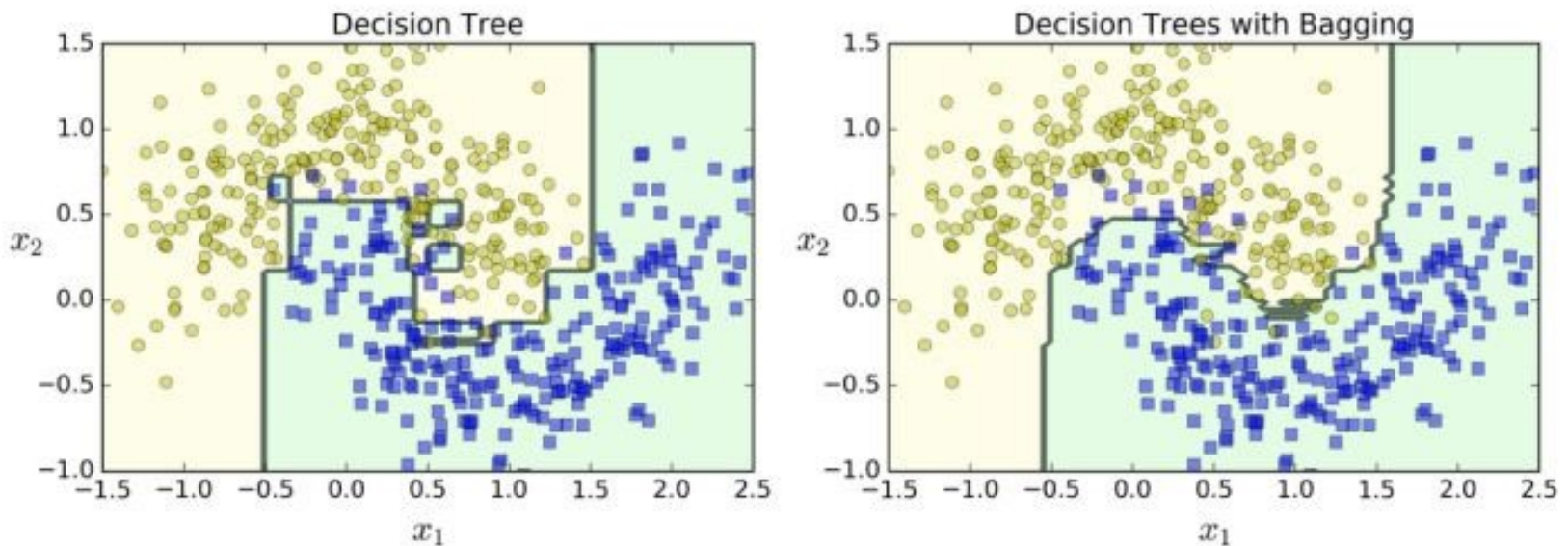
배깅(Bagging)

배깅을 통해 의사결정나무를 적합한 경우 더 **작은 variance**를 가짐



배깅(Bagging)

배깅을 통해 의사결정나무를 적합한 경우 더 **작은 variance**를 가짐

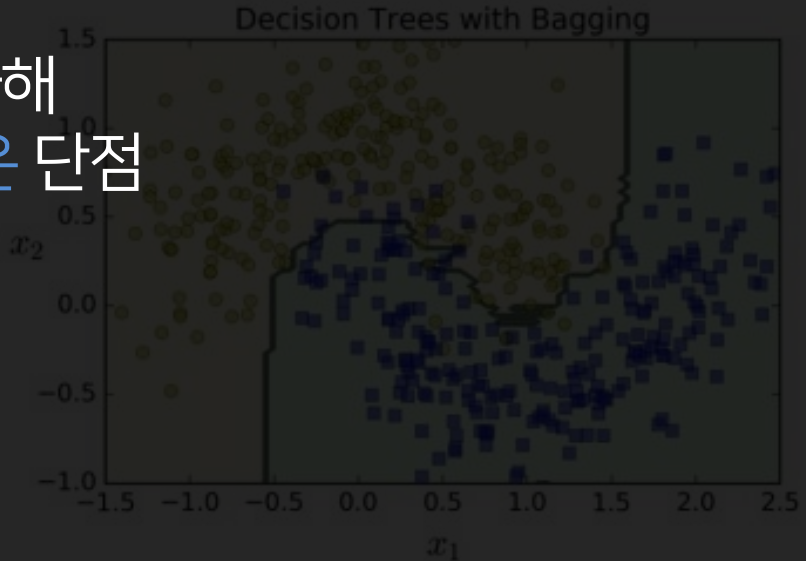
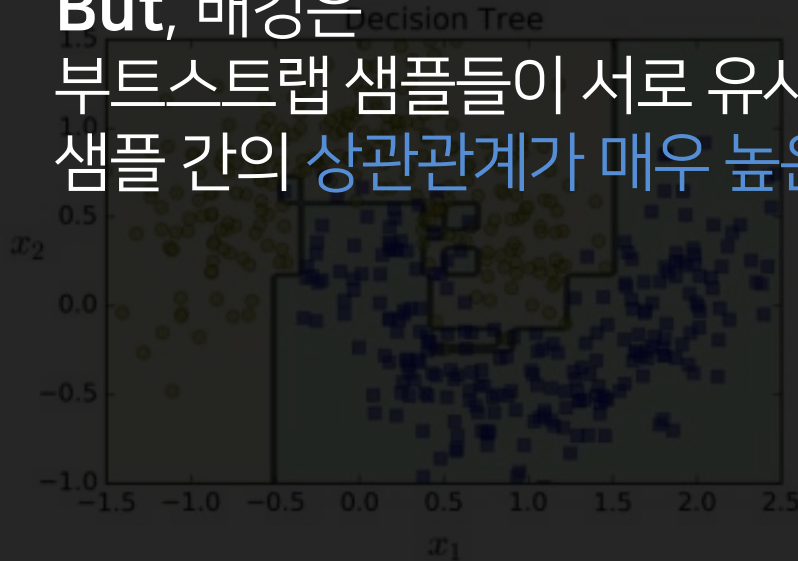


모집단에서 여러 개의 표본을 추출할 경우
표본평균의 분산인 $\frac{\sigma^2}{N}$ 의 값은 줄어들기 때문

배깅(Bagging)

배깅을 통해 의사결정나무를 적합한 경우 더 작은 variance를 가짐

But, 배깅은
부트스트랩 샘플들이 서로 유사해
샘플 간의 상관관계가 매우 높은 단점



모집단에서 여러 개의 표본을 추출할 경우
표본평균의 분산인 $\frac{\sigma^2}{N}$ 의 값은 줄어들기 때문

배깅(Bagging)

배깅을 통해 의사결정나무를 적합한 경우 더 작은 variance를 가짐

But, 배깅은
부트스트랩 샘플들이 서로 유사해
샘플 간의 상관관계가 매우 높은 단점

이를 해결하기 위해

랜덤 포레스트

두두 등장

모집단에서 여러 개의 표본을 추출할 경우
표본의 분산인 $\frac{\sigma^2}{N}$ 의 값은 줄어들기 때문



랜덤 포레스트(Random Forest)

랜덤 포레스트는 매 모델링마다 사용할 독립변수를 임의로 선택

랜덤 포레스트(Random Forest)

랜덤 포레스트는 매 모델링마다 사용할 독립변수를 임의로 선택

부트스트랩 샘플에 대한 배깅 실시



트리 적합 과정에서 일부 X변수만을 사용

랜덤 포레스트(Random Forest)

랜덤 포레스트는 매 모델링마다 사용할 독립변수를 임의로 선택

부트스트랩 샘플에 대한 배깅 실시

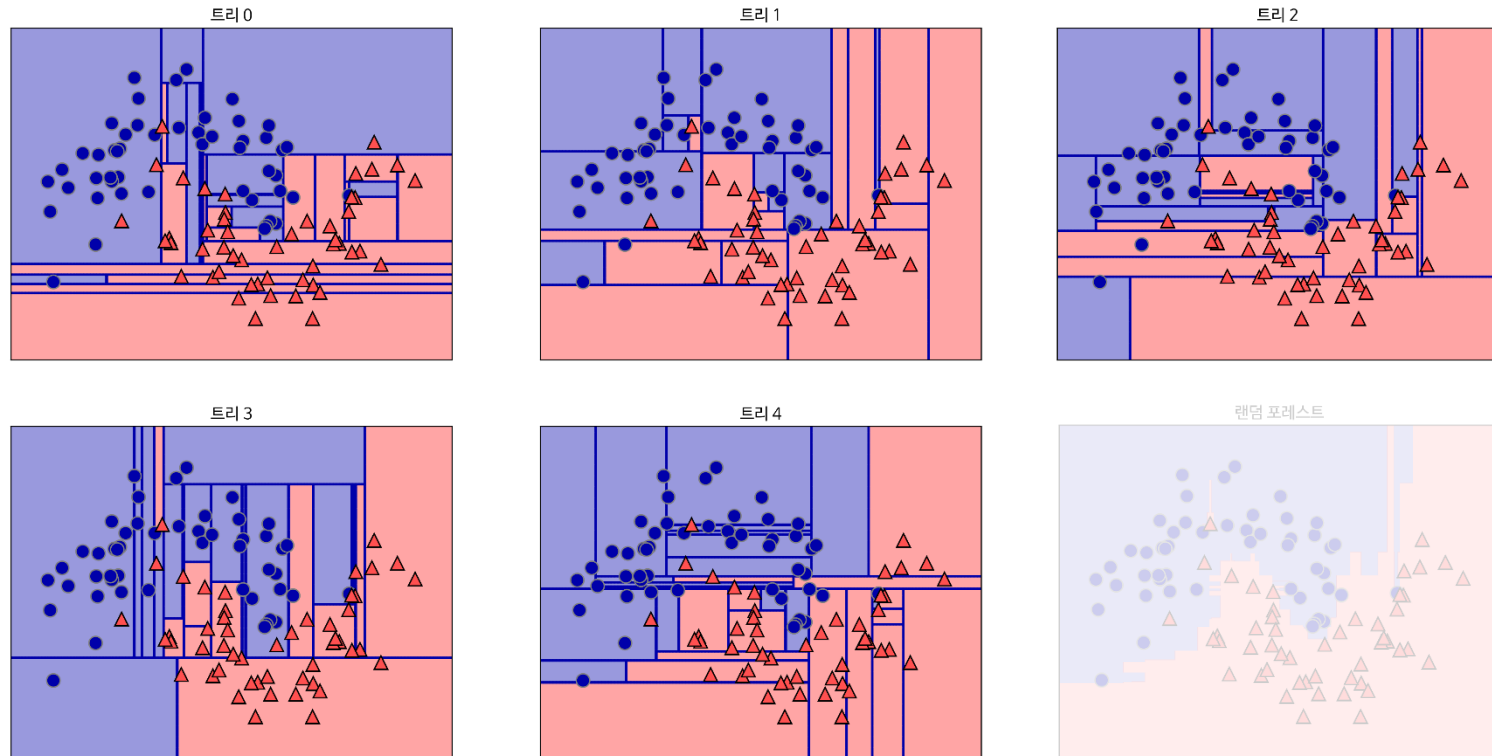


트리 적합 과정에서 일부 X변수만을 사용

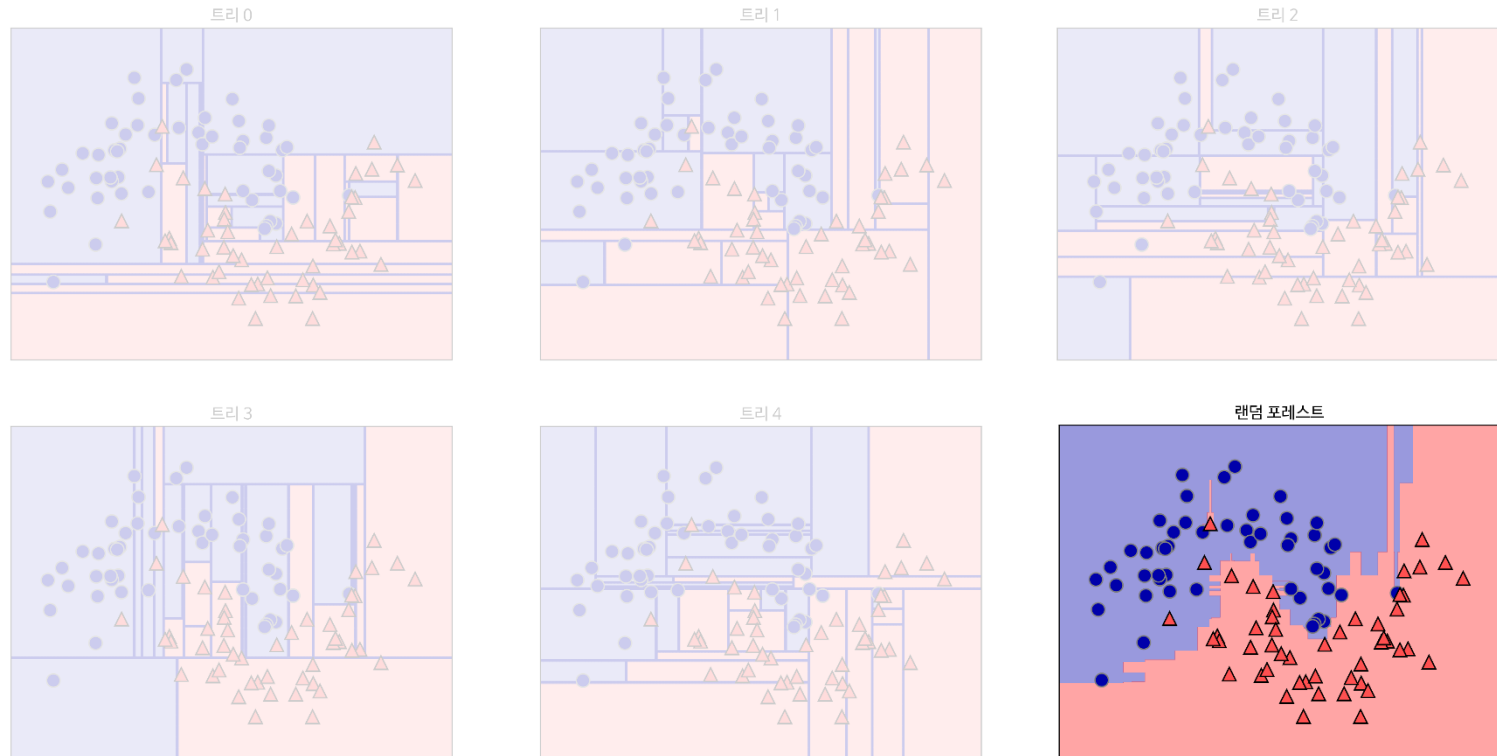


변수의 종류가 다양해지면서 Decorrelation 달성

랜덤 포레스트(Random Forest)



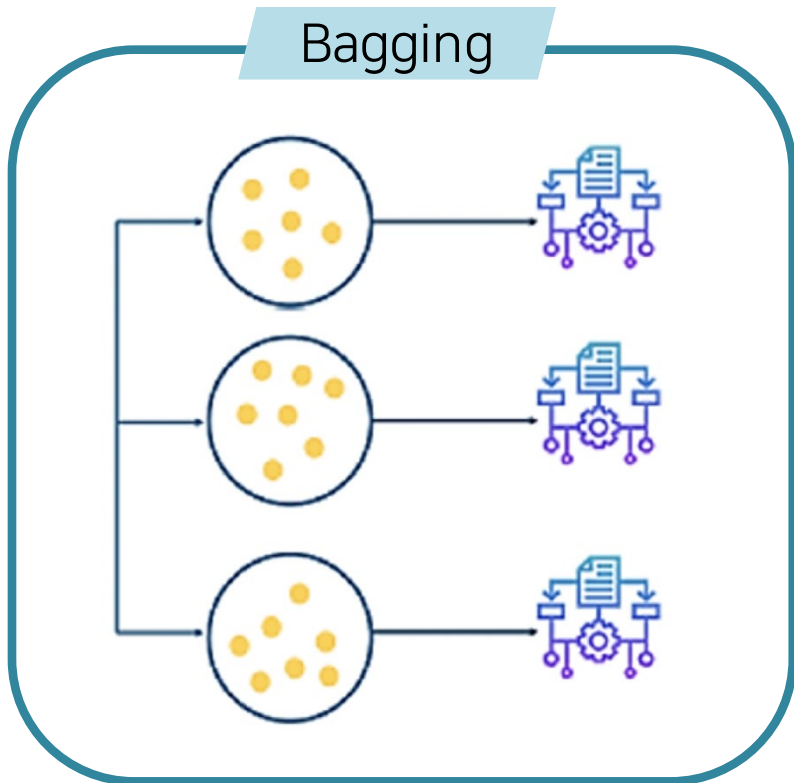
랜덤 포레스트(Random Forest)



각각의 트리보다 이들을 함께 적합한 **랜덤포레스트** 모형이 더 Robust 하다.

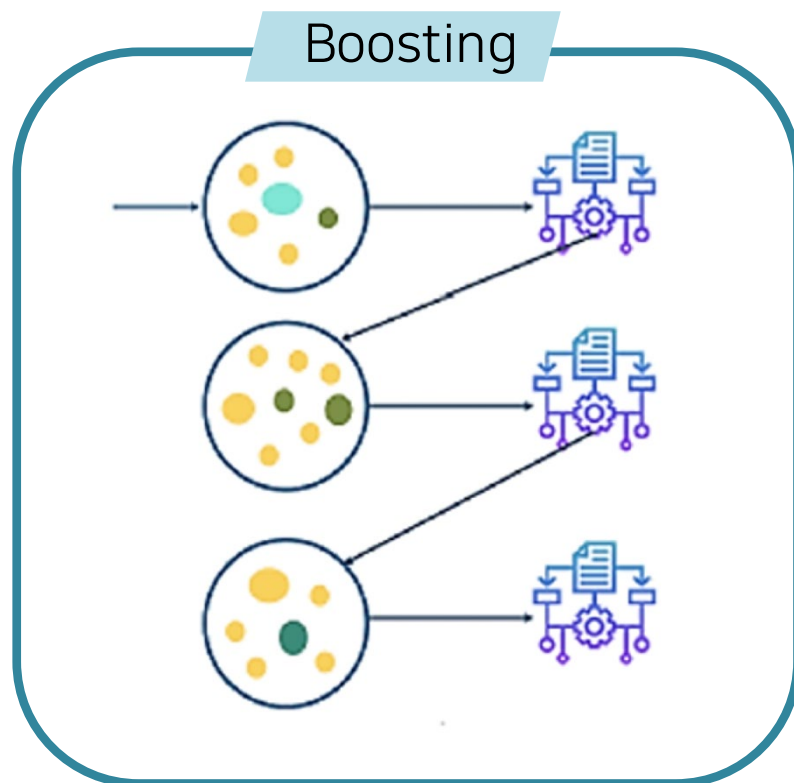
Boosting

Bagging



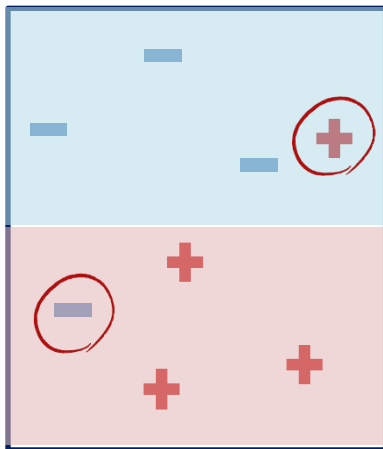
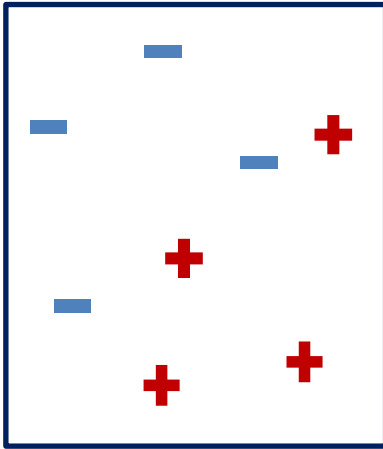
병렬적, 독립적인 학습

Boosting

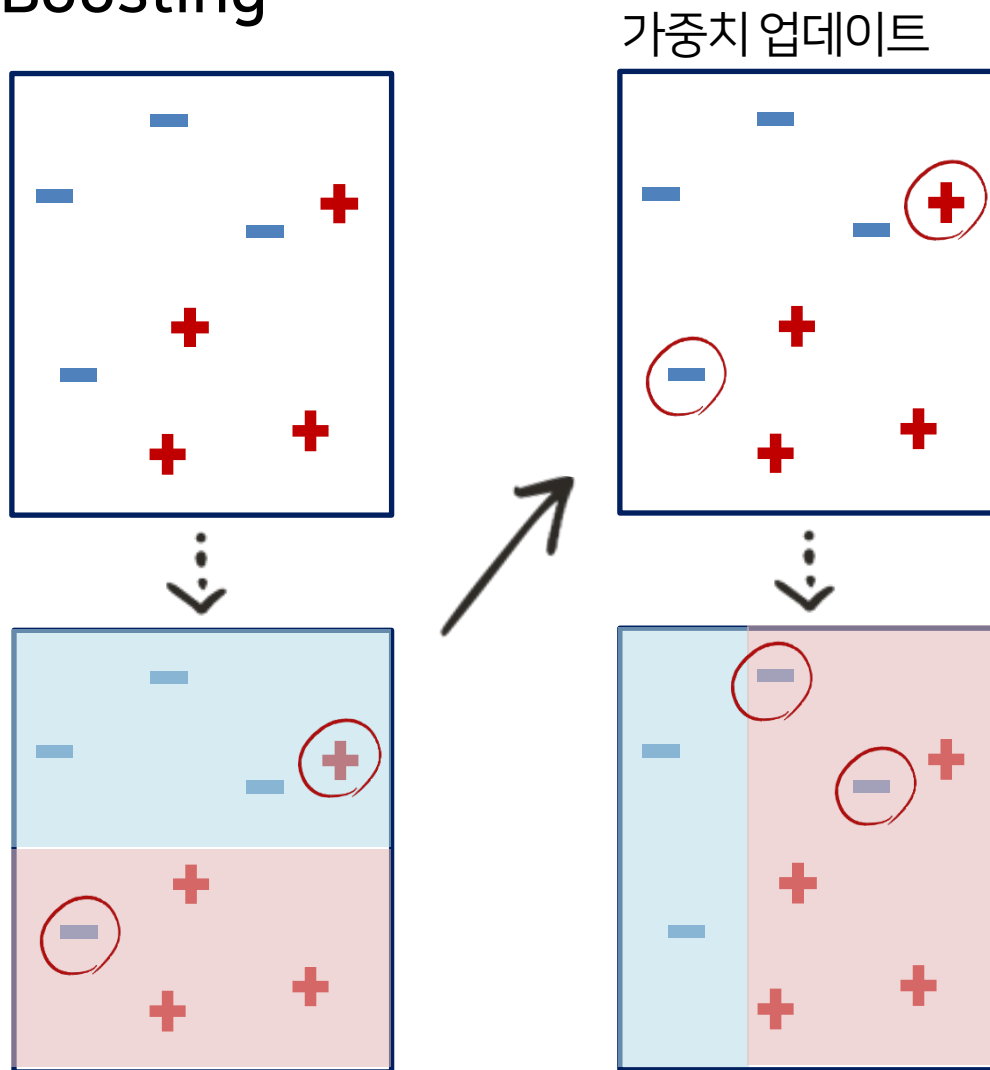


가중치를 활용한 순차적 학습

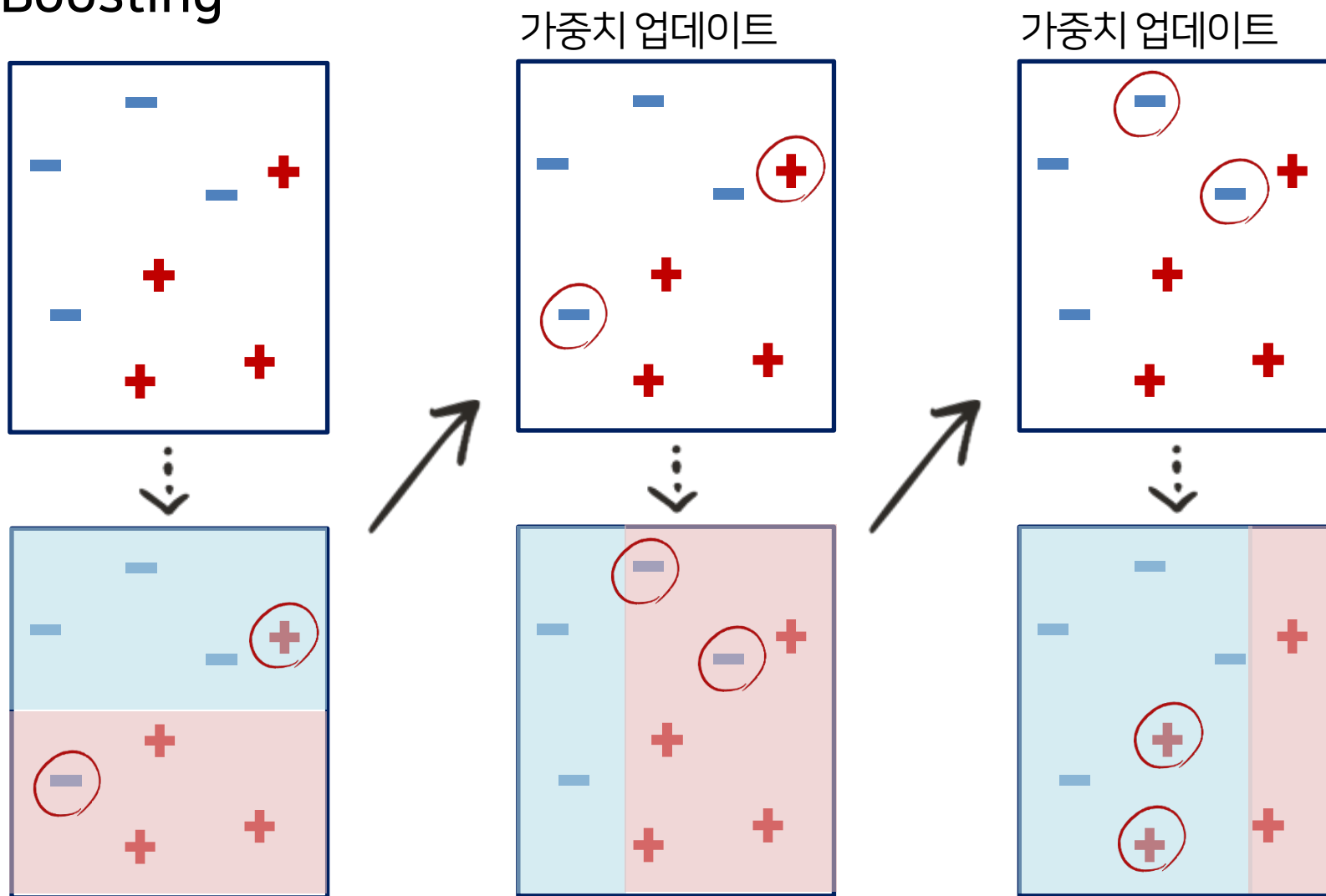
Boosting



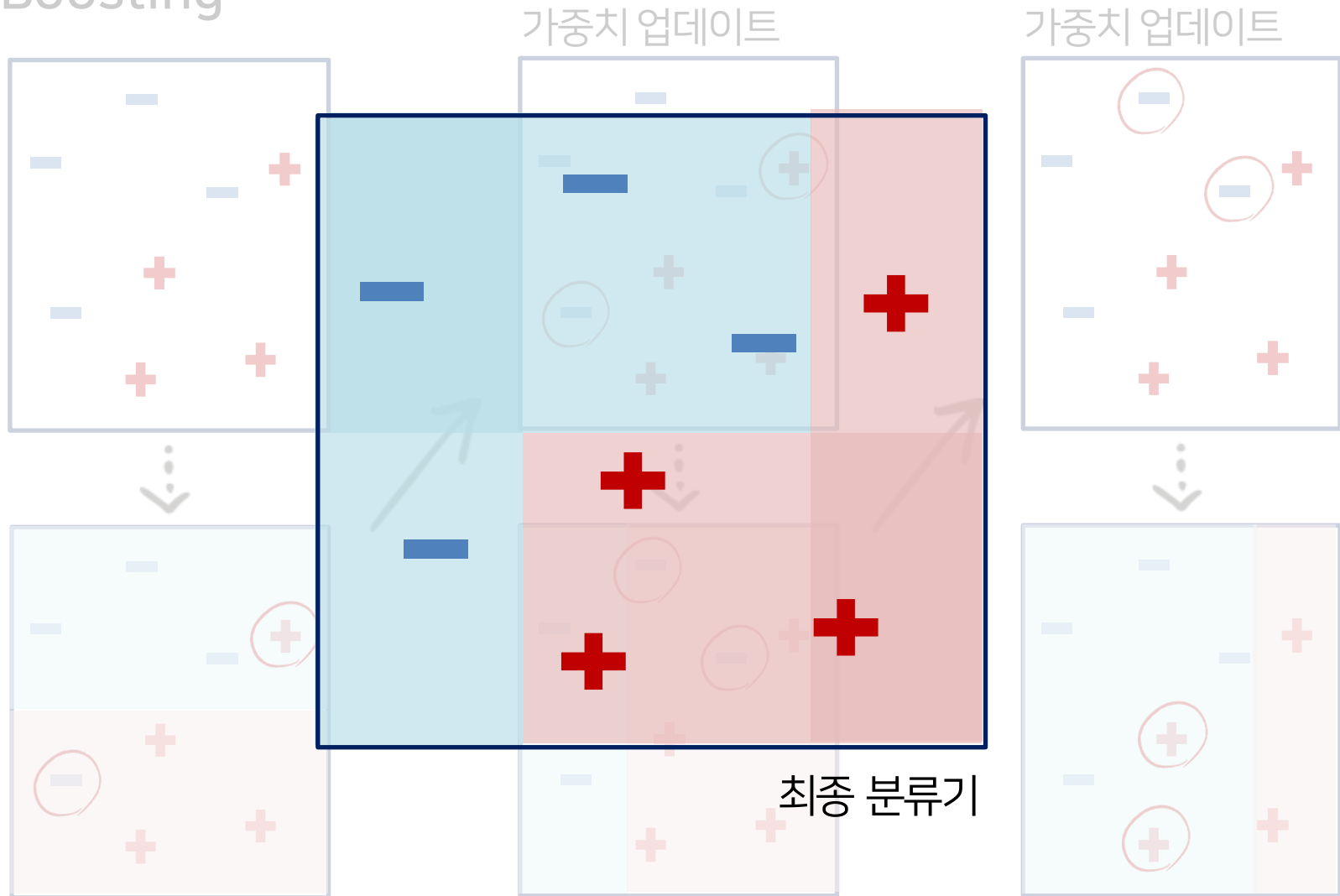
Boosting



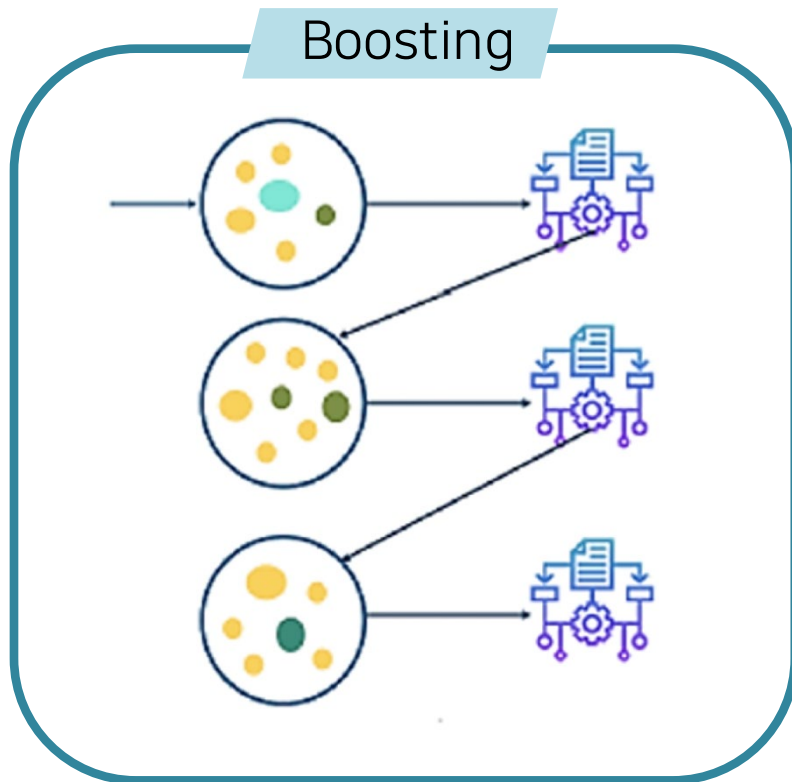
Boosting



Boosting



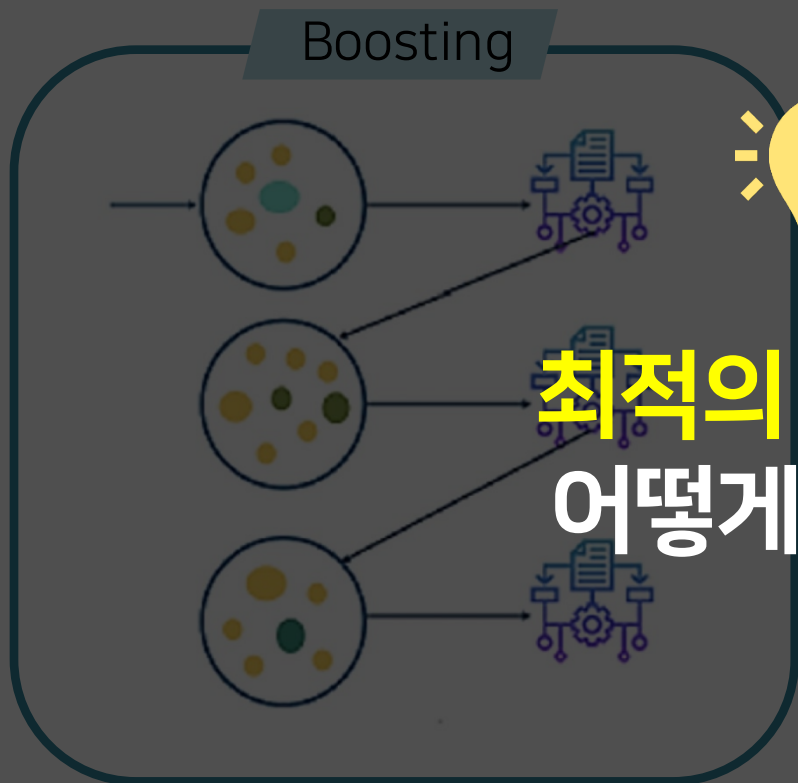
Boosting



Bagging에 비해 좋은 성능

느린 속도
오버피팅이 될 가능성

Boosting



Bagging에 비해 좋은 성능

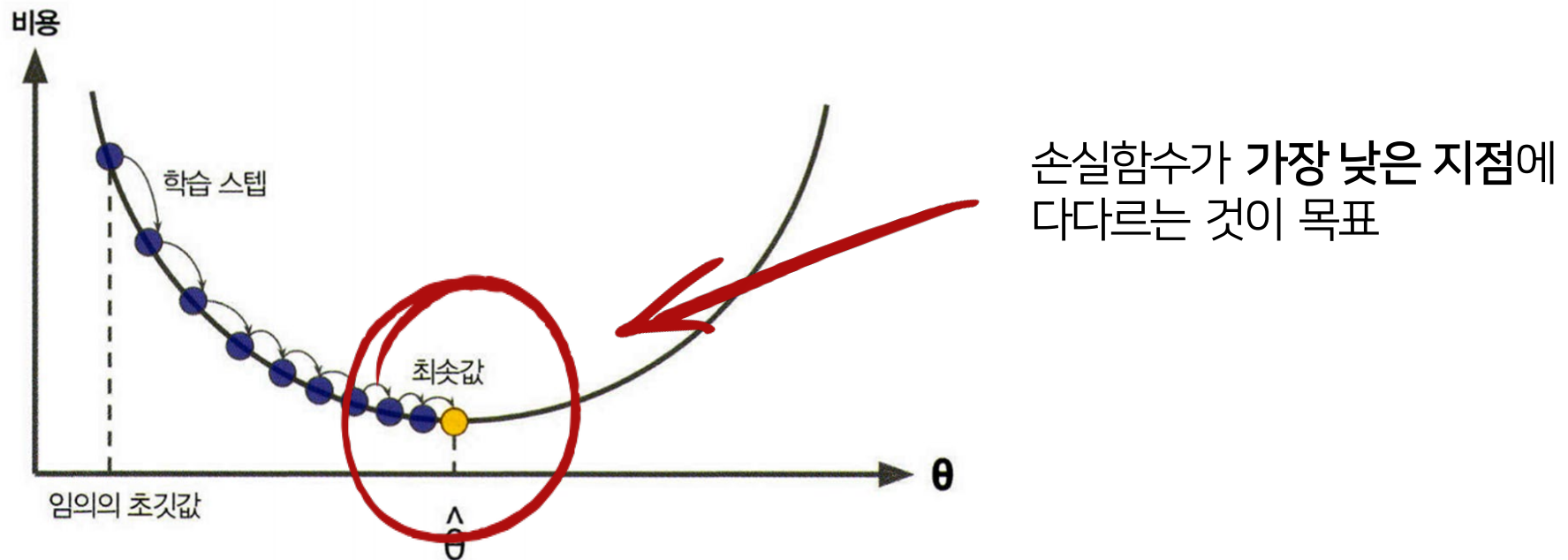
**최적의 가중치는
어떻게 찾을까?**

느린 속도
과다 피팅이 될 가능성

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

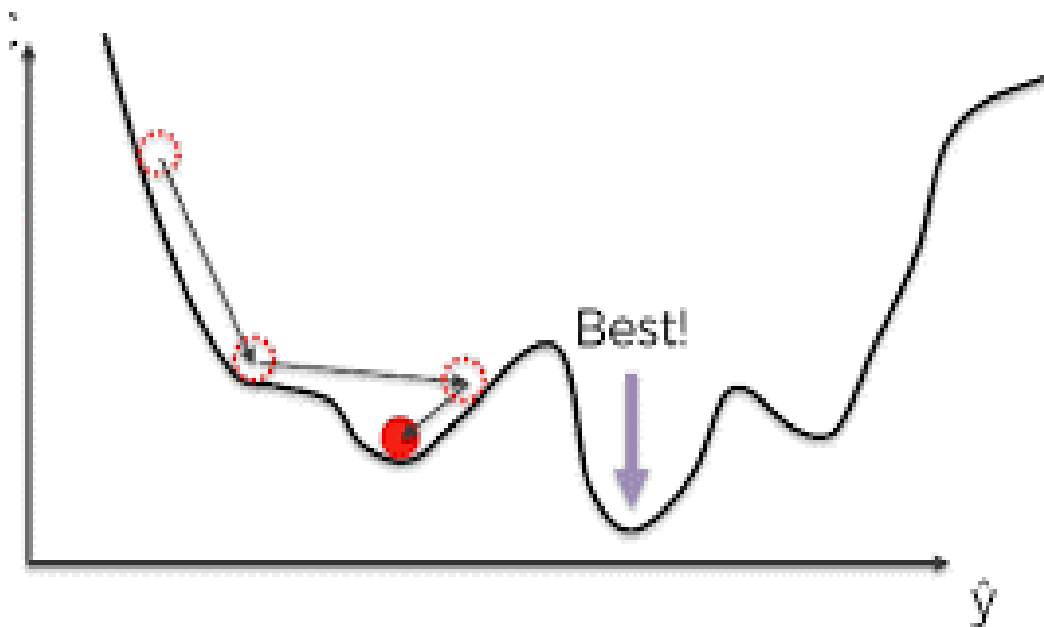


* 손실함수? 예측값과 실제 값의 차이(잔차)를 표현한 함수

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제



Learning rate

- 너무 작게 설정 :
지역 최솟값(local minimum)
에 빠짐
- 너무 크게 설정 :
수렴하지 못할 위험이 존재

* Learning rate? 딥러닝팀이 잘 알려줄 것이다!

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

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

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

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



평균

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

Height (m)	Favorite Color	Gender	Weight (kg)	(pseudo) Residual
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	4.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	75	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

(pseudo) Residual = 종속변수 - 평균값

모델이 적합됨에 따라 변화하는 residual을 예측하는 과정

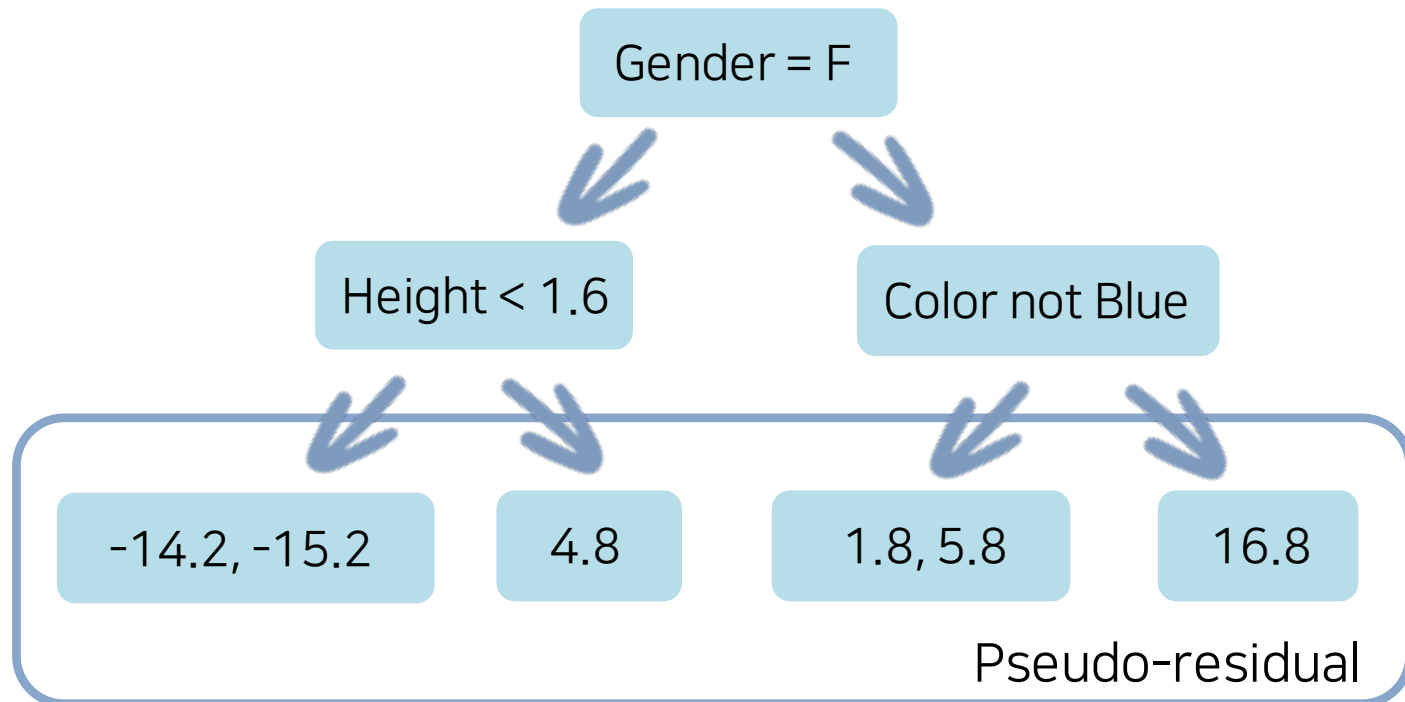
* Residual 값은 각 단계마다 새롭게 계산된다!

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

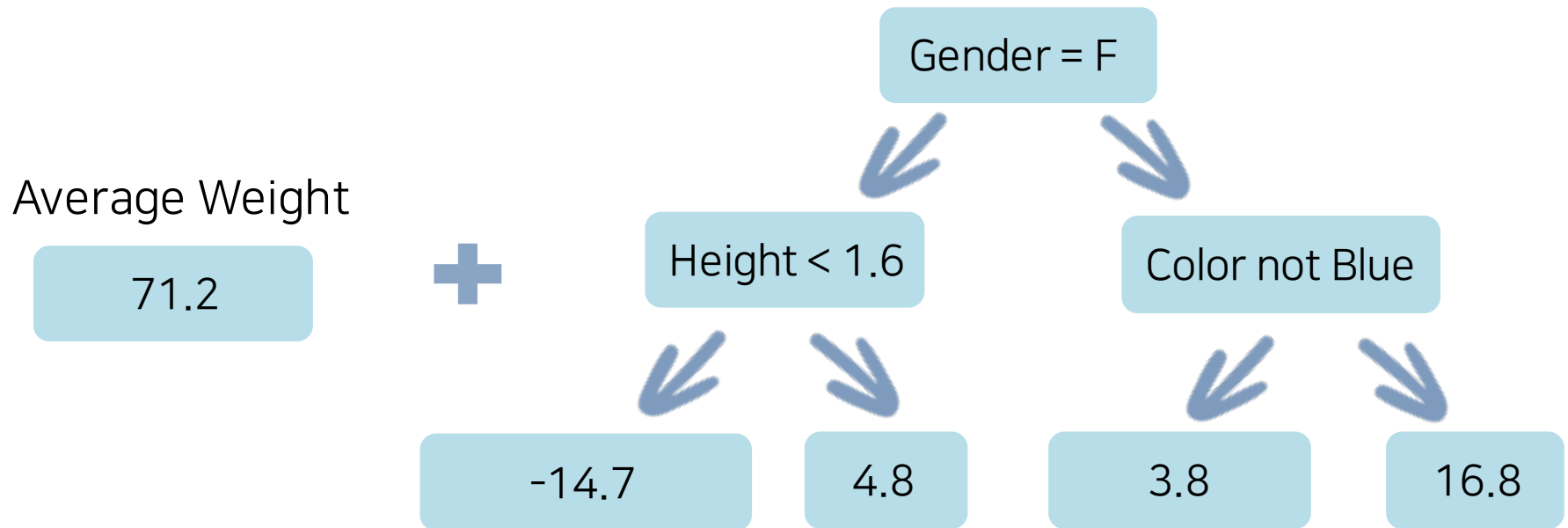


1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]



* 마지막 node에 두 개의 residual 값이 있는 경우 평균내주자!

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88



$$71.2 + 16.8 = 88$$

⋮

Average Weight

71.2

+

Height < 1.6

-14.7

4.8

Gender = F

Color not Blue

3.8

16.8

1) LGBM

Light Gradient Boosting Machine

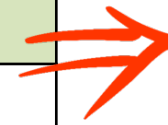
- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]

Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88

⋮

< Overfitting >



$$71.2 + 16.8 = 88$$

Average Weight

71.2

+

Height < 1.6

-14.7

4.8

Color not Blue

3.8

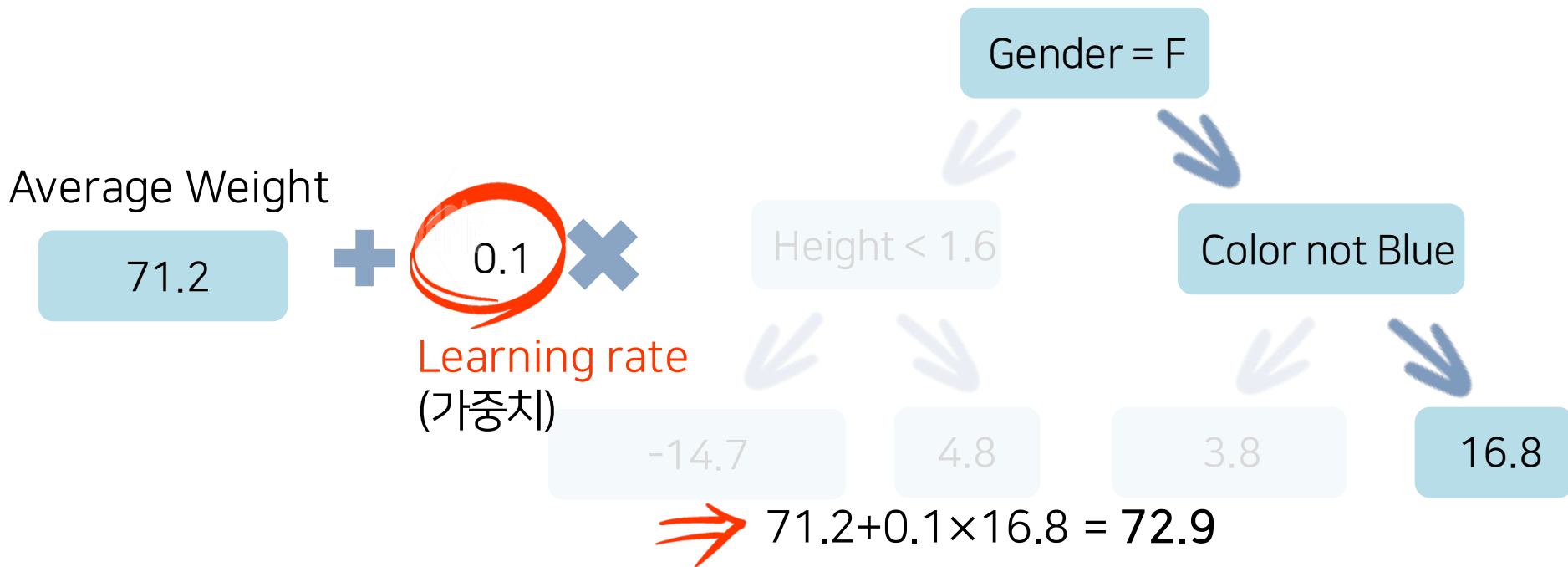
16.8

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

[키, 좋아하는 색, 성별을 통해 두고 몸무게를 예측]



1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

Average Weight

71.2

+

0.1

×

Gender = F

Height < 1.6

Color not Blue

-14.7

4.8

3.8

16.8

Residual

16.8

4.8

-15.2

1.8

5.8

-14.2

Residual

15.1

4.3

-13.7

1.4

5.4

-12.7

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

Average Weight

71.2

+

0.1

✖

Height < 1.6

Color not Blue

잔차(residual)가 더 이상 작아지지 않을 때까지

OR

사용자가 지정한 iteration 횟수에 도달할 때까지
반복하며 순차적으로 전개됨

Residual

16.8

4.8

-15.2

1.8

5.8

-14.2

15.1

4.3

-13.7

1.4

5.4

-12.7

-14.7

4.8

3.8

16.8

1) LGBM

Light Gradient Boosting Machine

- 경사하강법을 이용한 최적화 문제

Average Weight

71.2

+

0.1

×

Height < 1.6

Color not Blue

LGBM : GBM을 개선한 모델

Residual
16.8
4.8
-15.2
1.8
5.8
-14.2

Residual
15.1
4.3
-13.7
1.4
5.4
-12.7

- 빠른 속도
- 적은 메모리 차지
- GOSS 방법을 취함

Gender = F

-14.7

4.8

3.8

16.8

1) LGBM

Light Gradient Boosting Machine

GOSS (Gradient Based **O**ne **S**ide **S**ampling)

: 큰 error를 보이는 관측치들의 error를 줄이는데 집중

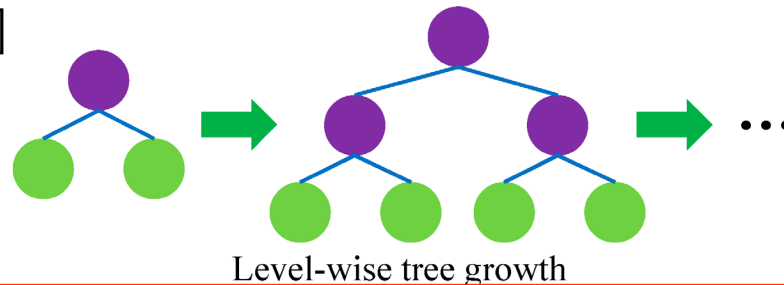
1) LGBM

Light Gradient Boosting Machine

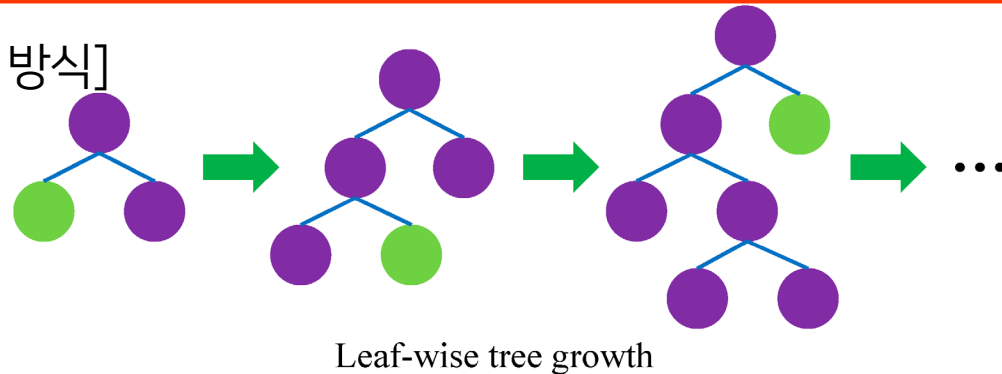
GOSS (Gradient Based One Side Sampling)

: 큰 error를 보이는 관측치들의 error를 줄이는데 집중

[너비 우선 방식]



[깊이 우선 방식]



1) LGBM

Light Gradient Boosting Machine

GOSS (Gradient Based One Side Sampling)

: 큰 error를 보이는 관측치들의 error를 줄이는데 집중

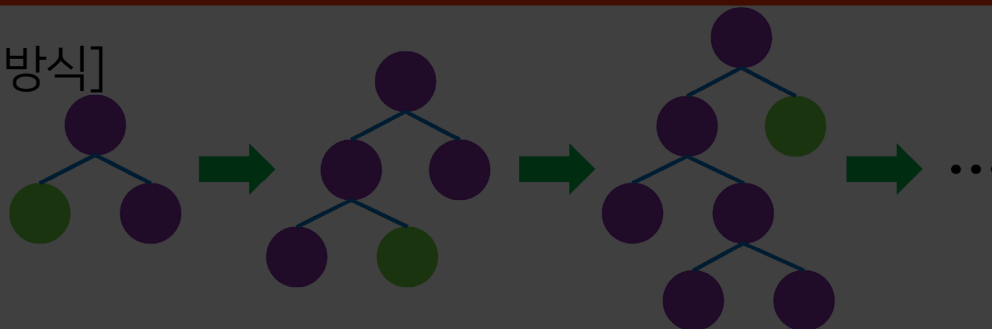
[너비 우선 방식]

LGBM

: 예측 오류 최소화가 목표

Level-wise tree growth

[깊이 우선 방식]



Leaf-wise tree growth

2) XGBoost

a.k.a. GBM Killer



회귀, 분류 문제 모두 수행 가능

2) XGBoost

a.k.a. GBM Killer





THANK YOU

