2022/02/03
17기 분석 권예은

Ensemble

Boosting & Stacking

목차

01

앙상블 분류

앙상블 기법

- Bagging
- Voting
- Boosting
- Stacking

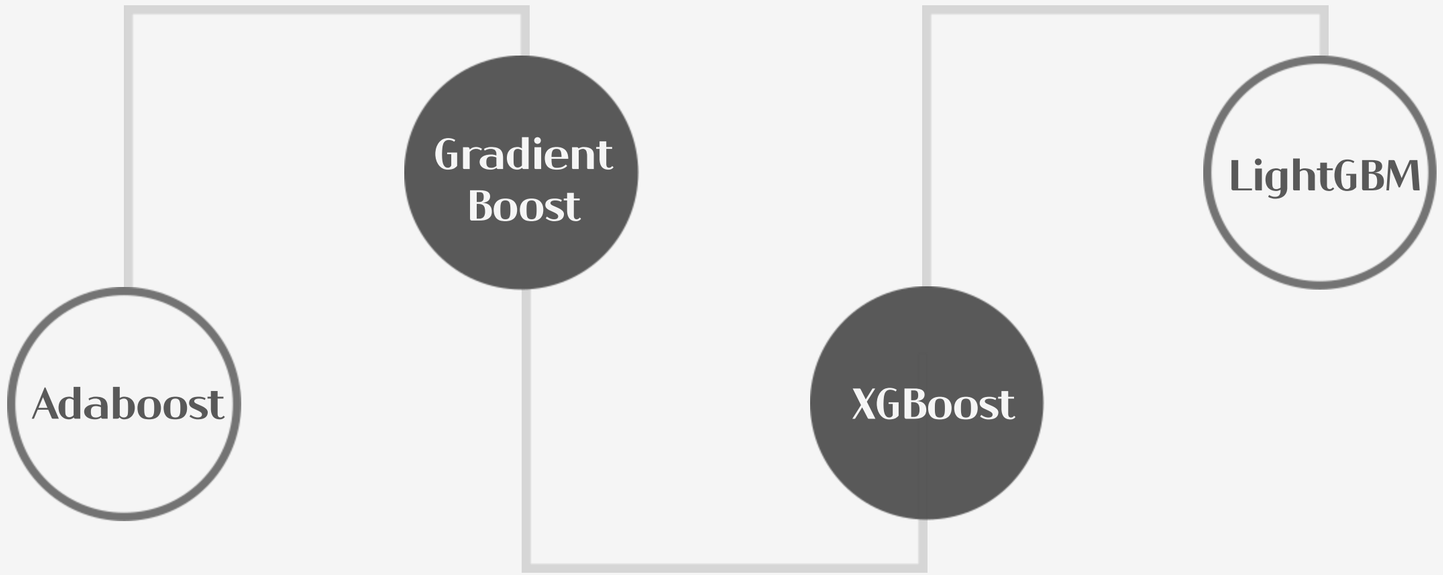
같은 알고리즘의 모델을 여러 번 사용하는 기법
- Bagging (Ex RandomForest)
- Boosting

다른 알고리즘의 모델을 결합해서 사용하는 기법
- Voting
- Stacking

02

Boosting

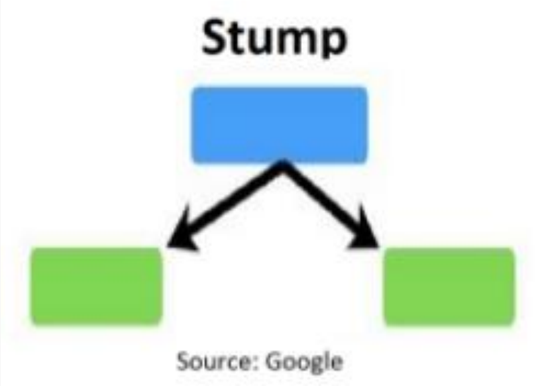
Boosting 기본 모델



Boosting

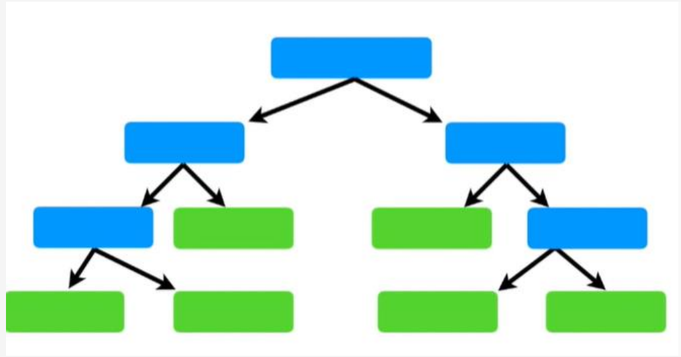
약한 학습기들을 모아서 강한 학습기를 만드는 방법으로
Bagging과 다르게 순차적으로 연결되어 있어 이전의 학습기가
다음의 학습기에 영향을 미친다.

- 약한 학습기란?



Split이 한번만 시행된 tree 형태의 모델인 stump
Boosting 모델은 분기를 최소화

이러한 모델들을 여러 개를 이용하는 방식이 Boosting!



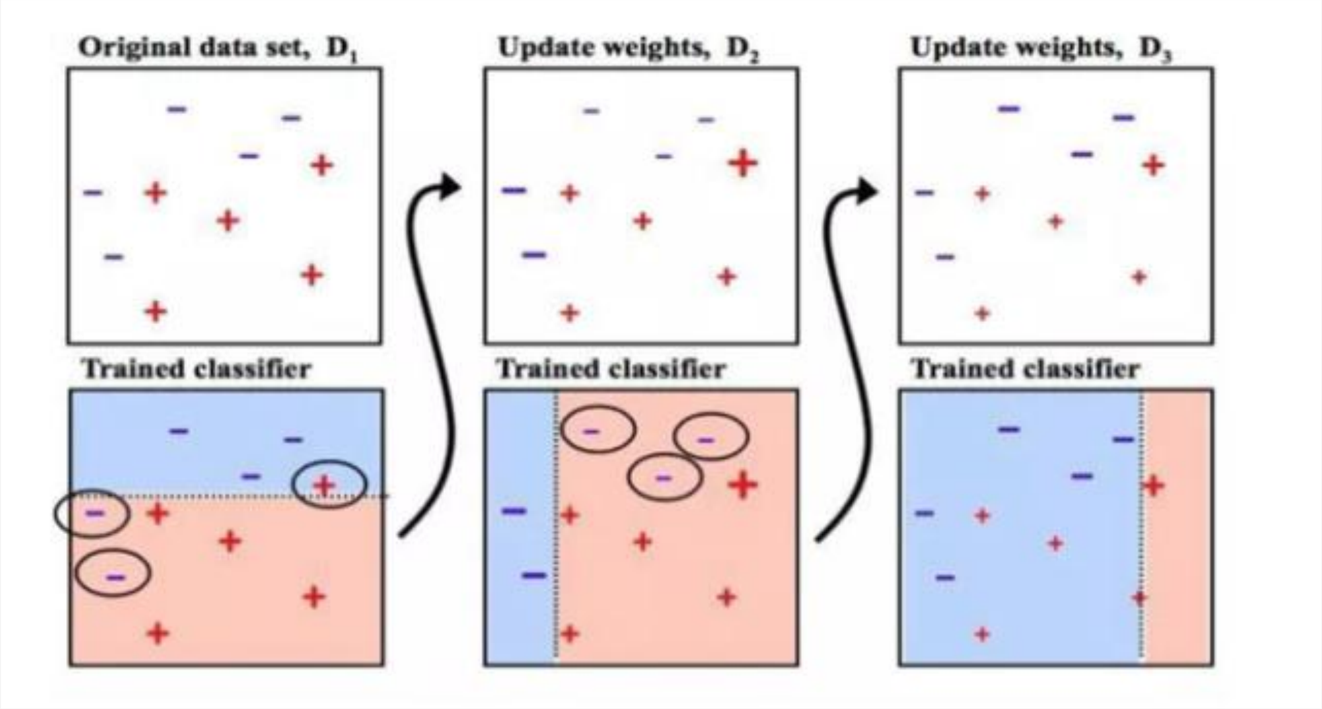
> 한 명의 천재 vs 평범한 사람 100명

08

AdaBoost

-Adaptive Boosting

이전의 모델이 잘못 분류한 샘플에 가중치를 높여서
다음 모델에 반영하는 알고리즘 -> 마지막 모델이 항상 BEST는 아니다.



09

AdaBoost

-Adaptive Boosting

Total loss = 잘못 분류된 데이터 샘플들의 가중치 총합

$$New\ sample\ weight(incorrectly) = sample\ weight * e^{\text{Amount of say}}$$

$$New\ sample\ weight(correctly) = sample\ weight * e^{-\text{Amount of say}}$$

$$Amount\ of\ say = \frac{1}{2} * \log(\frac{1 - Total\ loss}{Total\ loss})$$

- Process

1. Feature의 개수만큼 stump를 만들고 Gini index가 가장 낮은 stump 먼저 학습

   (Gini index는 Decision Tree에서의 분류 기준으로 불순도를 의미)

2. 처음에는 모든 샘플 가중치를 1/N
3. Weak learner로 데이터 학습
4. Weak learner의 Total loss를 구하고 Weak learner의 가중치(Amount of say)를 계산
5. Weak learner의 가중치에 따라 새롭게 sample weight가 업데이트

-> 이때 오분류된 샘플의 가중치가 더 커진다.

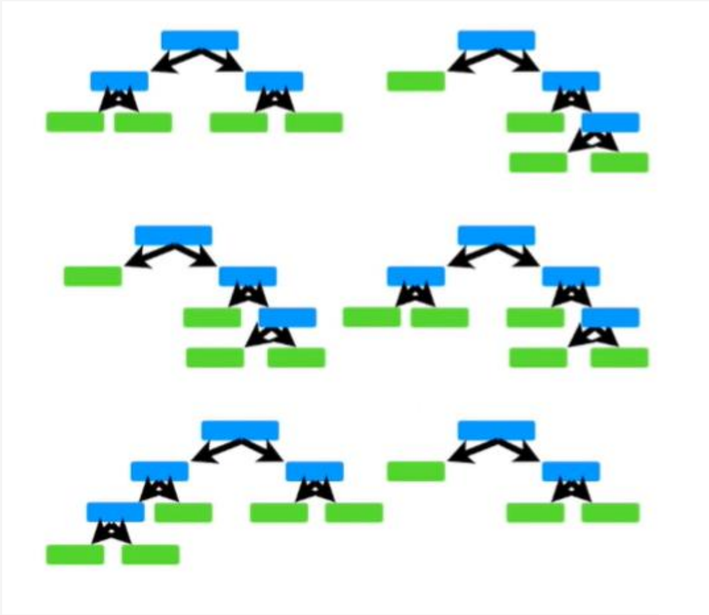
6. 새로운 sample weight를 정규화 & 다음 학습기로 전달
7. 3~7번 과정을 반복한다
8. 지금까지 학습된 Weak learner들의 가중치에 따라 결합하여 최종 예측

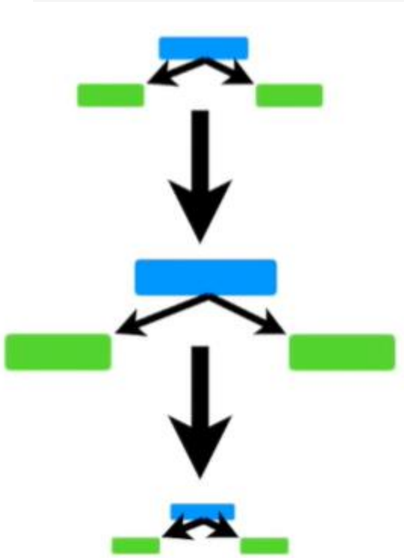
AdaBoost

-Adaptive Boosting

- RandomForest와 차이점

1. RandomForest에서는 모델이 서로 독립적이었다면
   Adaboost에서는 순차적
   -> 모델의 순서가 중요해진다.
2. Stump 형태의 weak learner를 이용한다
3. 마지막 예측에서 정확도에 따라서 모델들의 가중치가 반영





11

Gradient Boost

-Residual fitting

LightGBM,XGBoost의 기반이 된 모델
Target인 y값을 예측하는 것이 아니라 잔차가 점점 작아지도록 학습

잔차 = 실제 값 y - 예측 값 F(x)

$$y_i - f(x_i)$$

- Gradient란?

Loss 함수에 대한 기울기
->Loss를 최소로 만들기 위해서 F(x)가 움직이는 방향
->residual가 작아지도록 학습하는 건 결국 loss를 최소로 만드는 방향

$$j(y_i, f(x_i)) = \frac{1}{2}(y_i - f(x_i))^2$$

$$\frac{\partial j(y_i, f(x_i))}{\partial f(x_i)} = \frac{\partial\left[\frac{1}{2}(y_i - f(x_i))^2\right]}{\partial f(x_i)} = f(x_i) - y_i$$

Gradient Boost

-Residual fitting

- Process

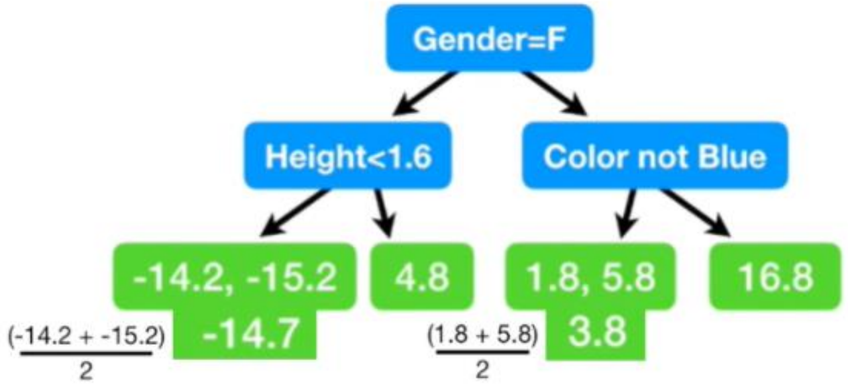
초기 예측값은 y값의 평균
잔차를 구하는 모델을 생성 -> leaf노드는 평균값

Average Weight

71.2

Height (m)	Favorite Color	Gender	Weight (kg)	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	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

(88 - 71.2) = 16.8



Gradient Boost

-Residual fitting

새로운 예측값= 이전의 예측값 + learning_rate * residual
새로운 잔차값= y값 - 새로운 예측값

- Learning _rate란

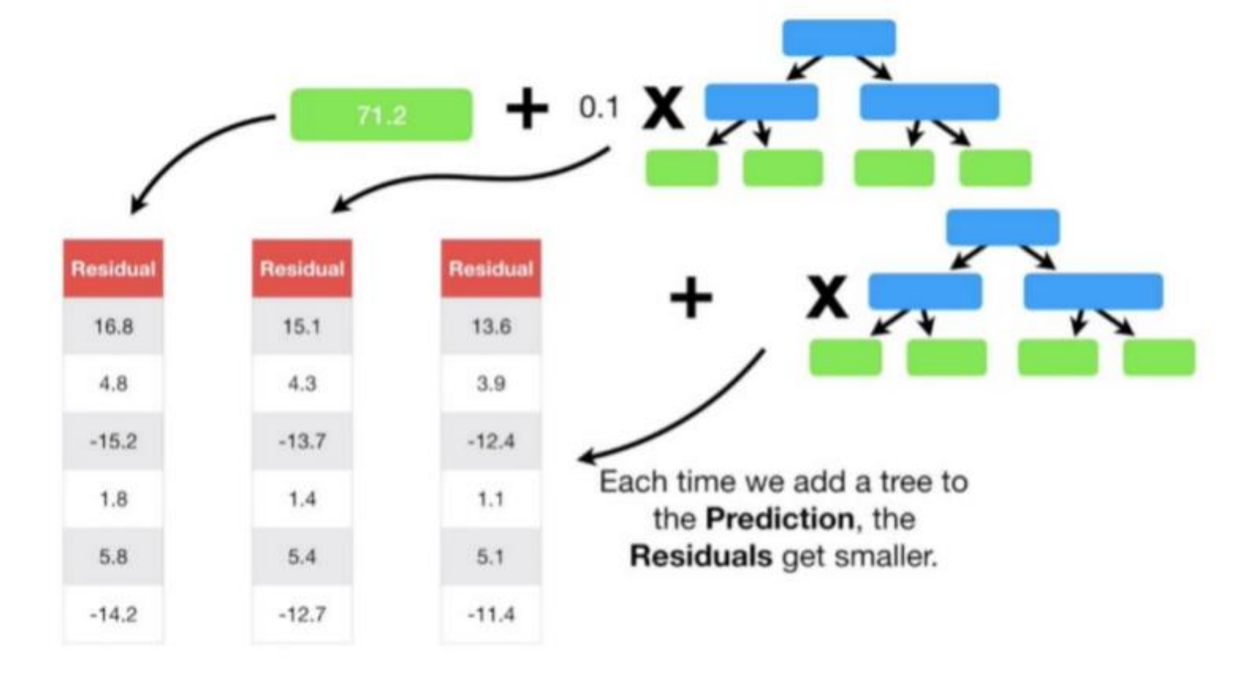
과적합을 제어하기 위해 사용하는 매개변수
이전 트리의 오차를 얼마나 강하게 보정할 것인지를 결정.
0과 1사이의 값

Gradient Boost

-Residual fitting

이 과정을 반복하여 잔차가 작아지는 방향으로 학습

지정된 iteration이 끝나거나 잔차가 더 이상 작아지지 않으면 학습 종료



Adaboost와의 차이

1.Stump가 아니라 tree 형태

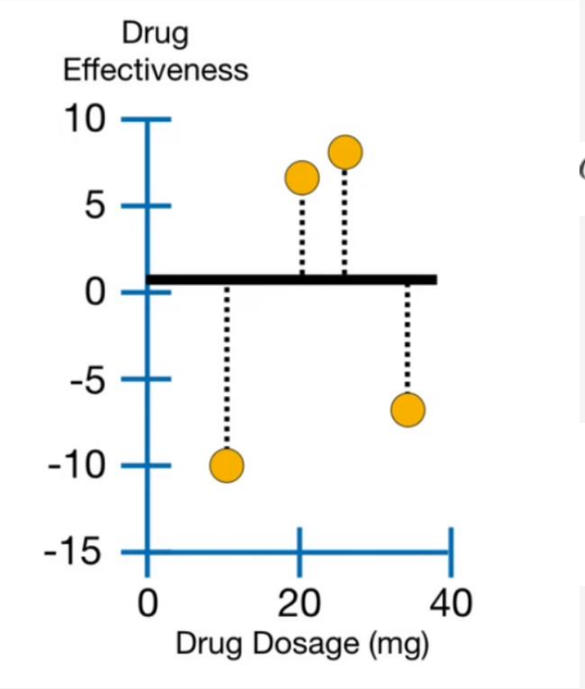
2. AdaBoost는 데이터의 가중치를 조정하여 예측

Gradient Boost는 잔차 오류에 맞추려고 한다.

XGBoost

-Extreme Gradient Boosting

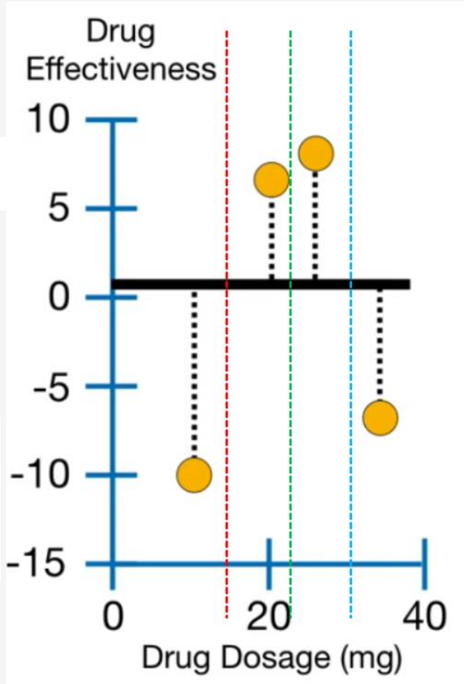
Gradient boosting을 병렬적으로 처리하는 알고리즘
모델 자체에서 과적합 규제가 가능
병렬처리로 Gradinet Boosting보다 빠른 속도
많은 파라미터로, 파라미터 튜닝으로 최적 모델 생성



$$Gain = (New\ Similarity\ Score) - (Old\ Similarity\ Score)$$

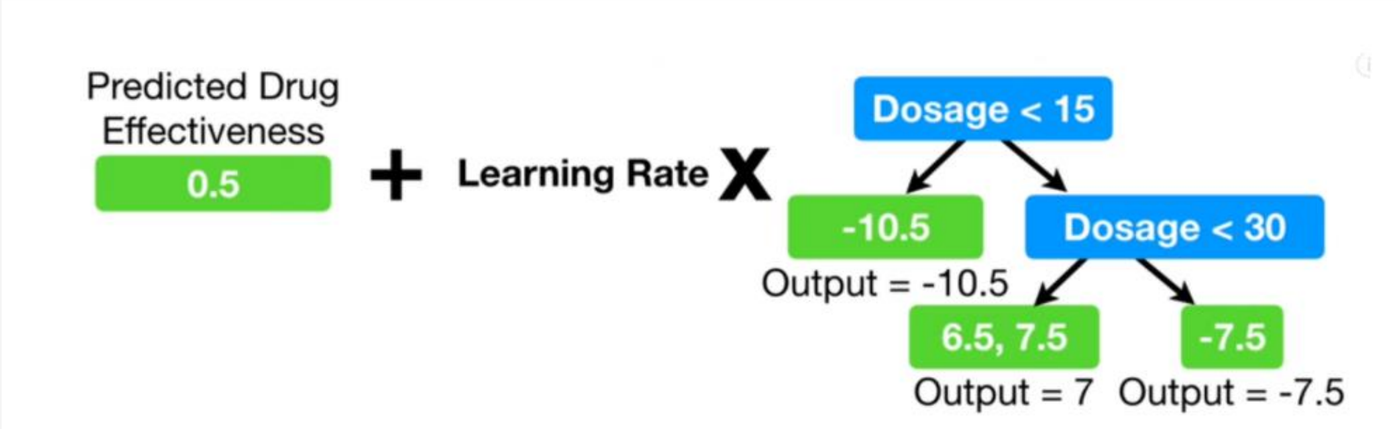
Gain을 극대화하는 기준으로 분기
-> 재귀적으로 분기

$$\text{Similarity Score} = \frac{\text{Sum of Residuals, Squared}}{\text{Number of Residuals} + \lambda}$$



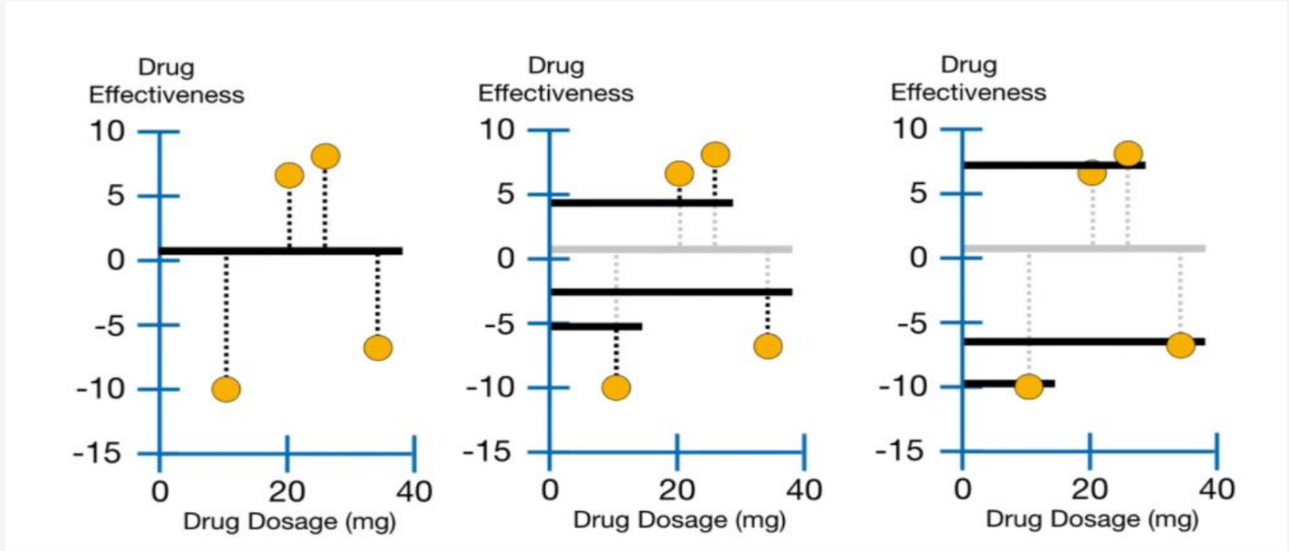
XGBoost

-Extreme Gradient Boosting



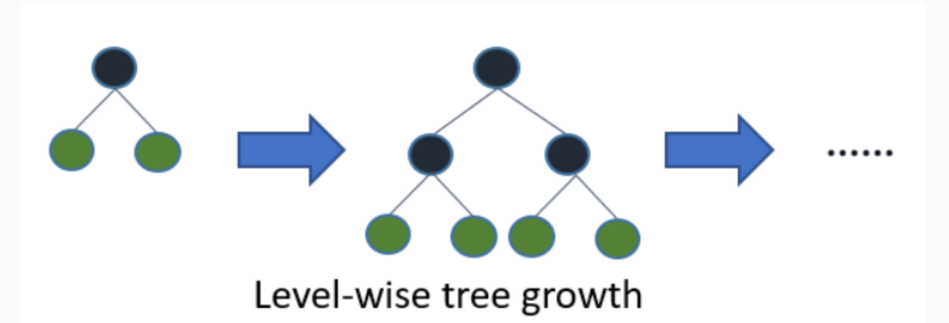
모든 Gain값이 음수가 되면 종료
새로운 예측값 계산

순차적으로 예측값의 변화

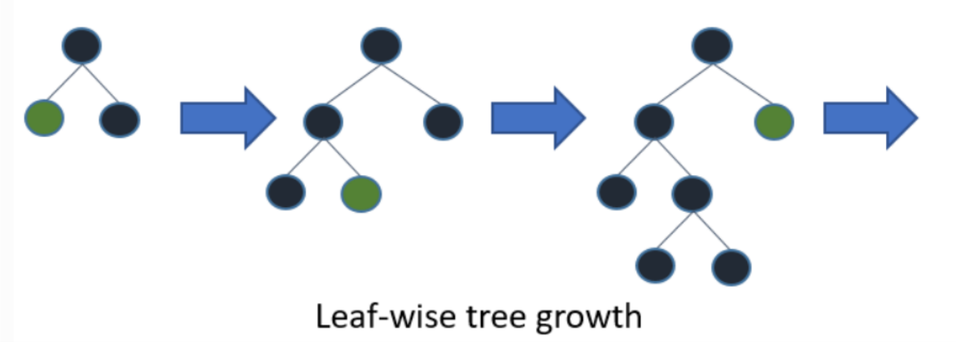


LightGBM

Leaf wise로 분류를 하는 분할 방식의 트리 기반 알고리즘
XGBoost에 비해 적은 학습시간과 메모리 사용량



일반적인 Depth wise tree기반 앙상블



과적합이 발생하기 쉬운 Leaf wise tree기반 앙상블

LightGBM

- Histogram based algorithm

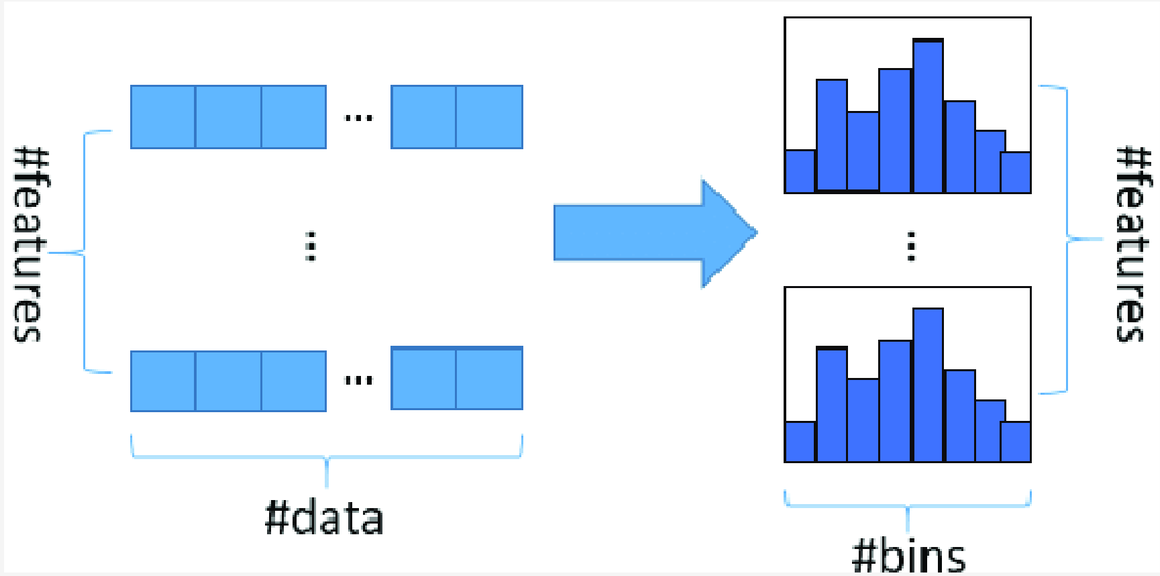
연속적인 변수 값을 이산적인 구간으로 feature histogram을
구성하여 최적의 split을 찾는다

$$\begin{bmatrix} 1.5 & 0.0 \\ 0.0 & 5.5 \\ 0.3 & 7.0 \\ 5.5 & 8.5 \end{bmatrix}$$

Continuous values

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 2 \\ 2 & 2 \end{bmatrix}$$

Discrete values



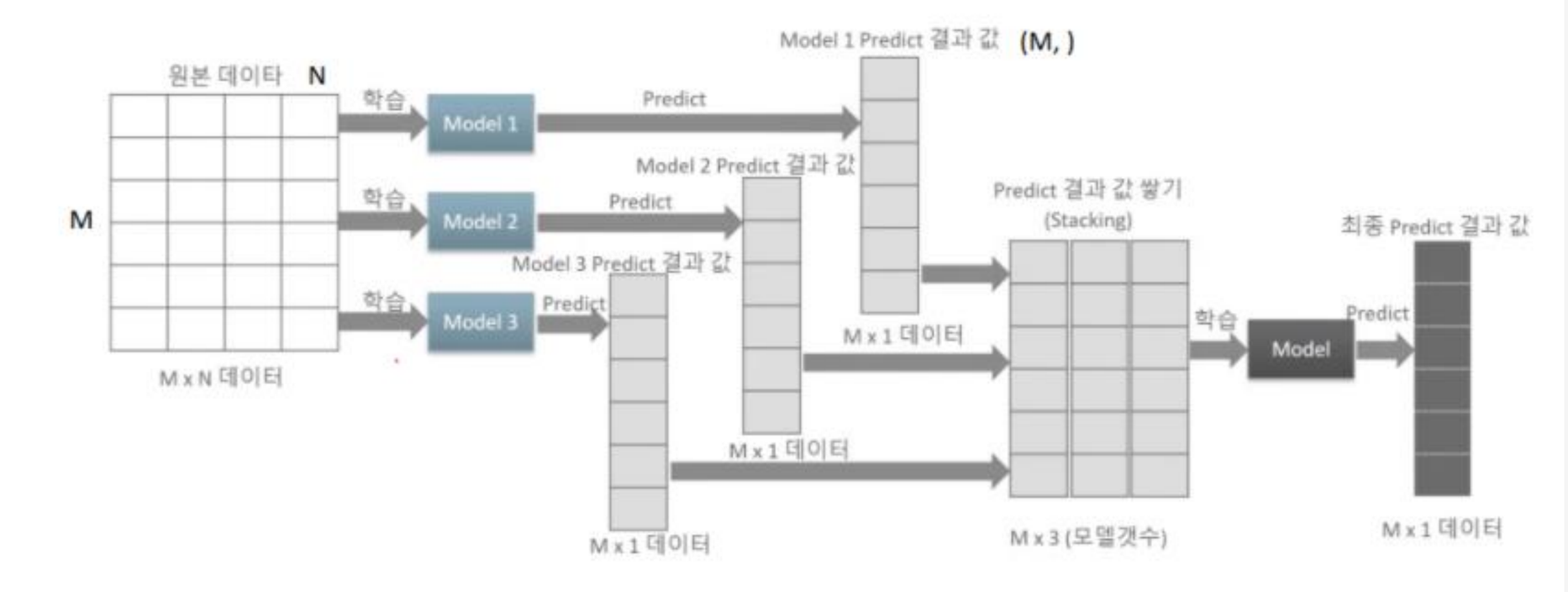
한번 histogram을 생성하면 계산해야하는 데이터 수가 적어지기 때문에 Gain을 계산하는 비용이 감소

03

Stacking

Stacking

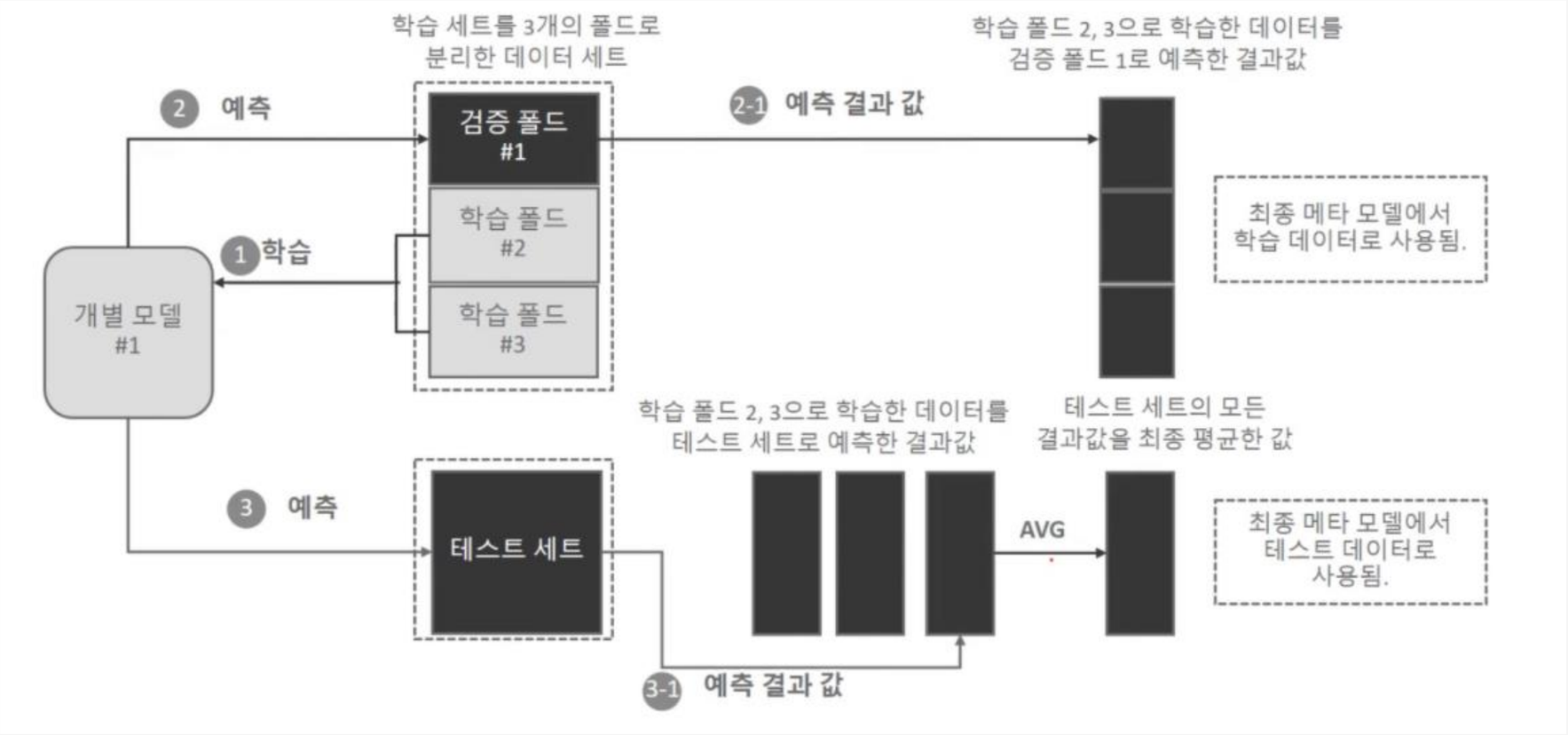
서로 다른 알고리즘의 모델로 학습한 후,
그 결과를 쌓아(Stacking) Meta dataset을 만들어 최종 모델(Meta model)로
학습시키는 알고리즘



StackingCV

- 교차 검증 세트 기반의 Stacking

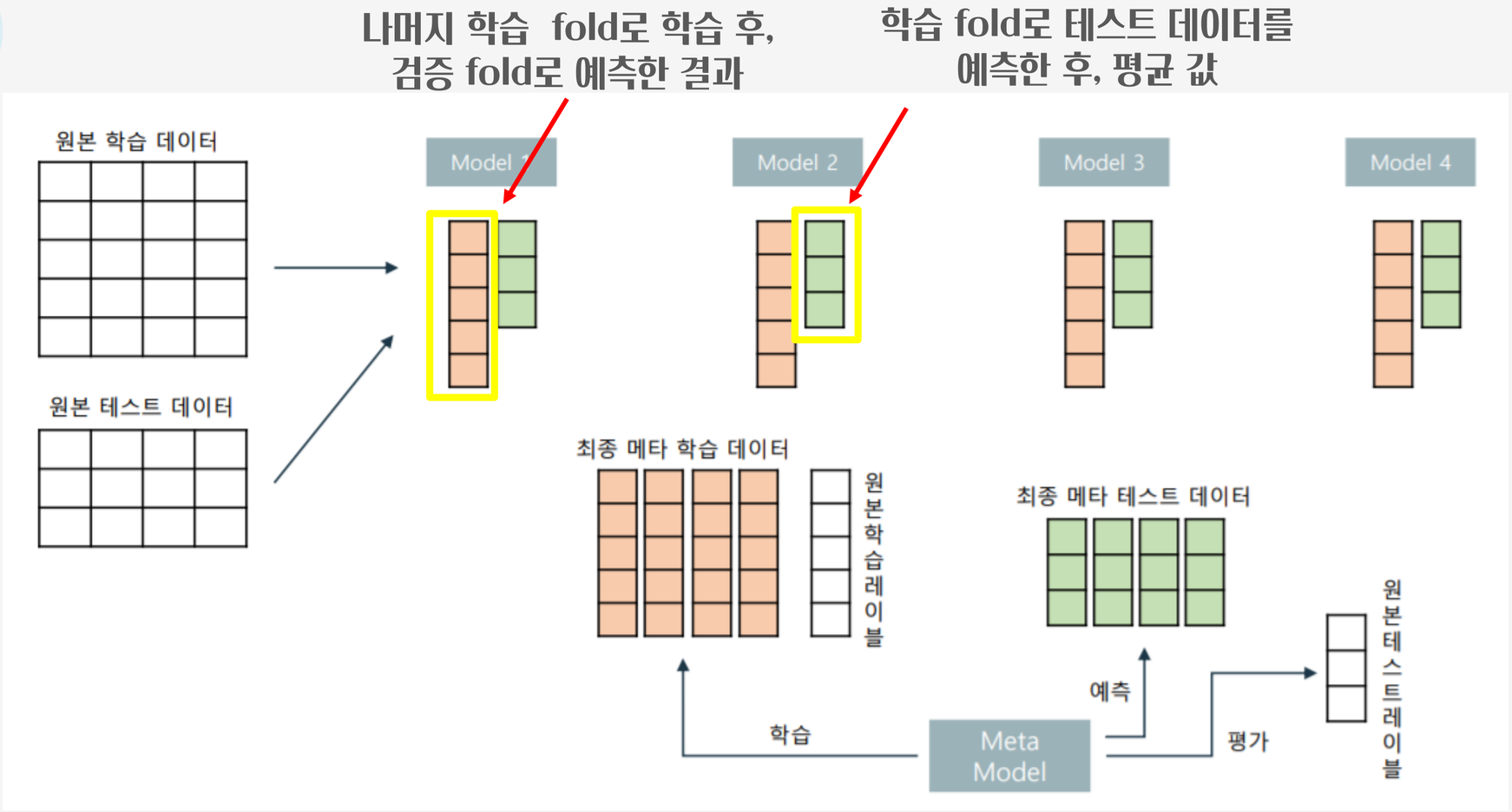
Step 1



StackingCV

- 교차 검증 세트 기반의 Stacking

Step 2



Q&A

감사합니다 :)