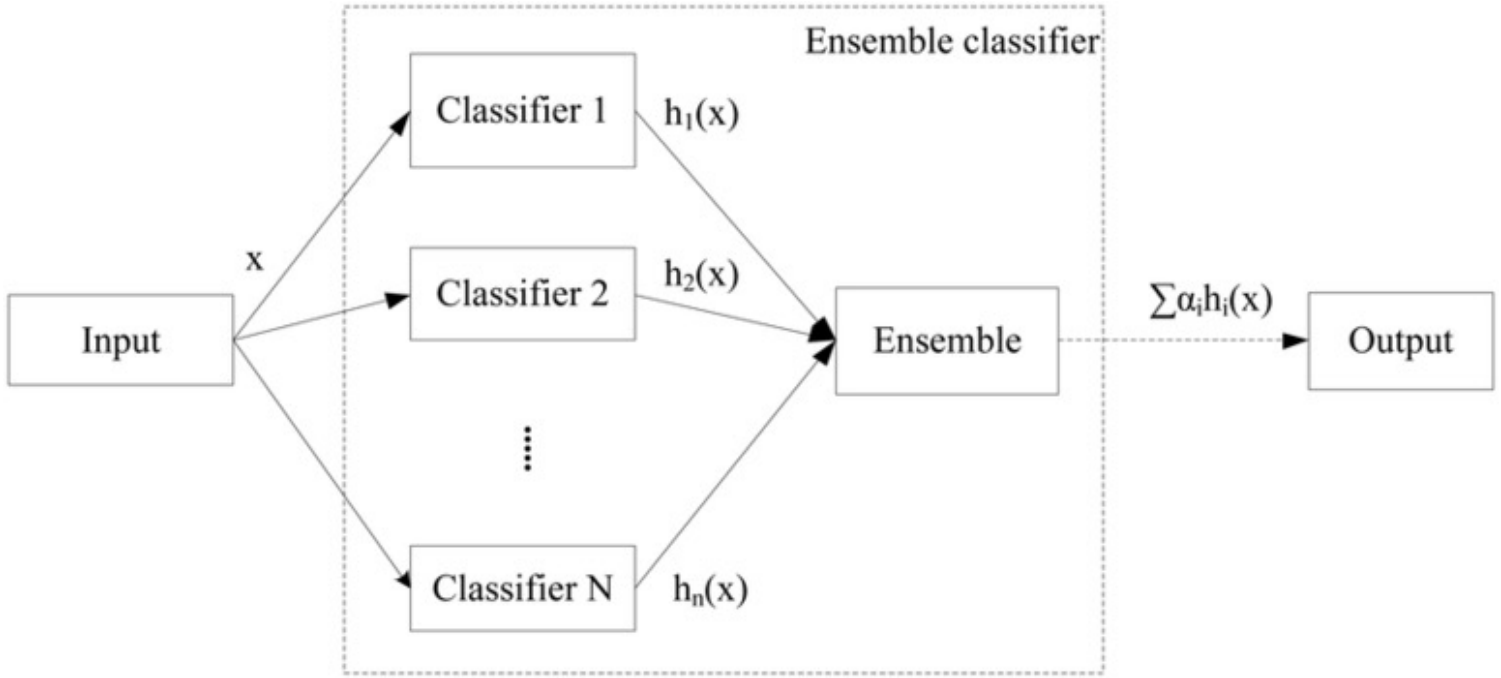
python™

Ensemble(앙상블)



- 머신러닝 알고리즘의 선택과 구성은 overfitting과의 전쟁이다!
- Ensemble(앙상블)은 여러 개의 약한 모델을 조합하여 최적의 모델로 일반화한다.

python™

Ensemble(앙상블)

- Ensemble(앙상블)
  : base model이 되는 weak learner을 여러 번 학습시킨 이후, strong learner을 적용시켜 성능을 극대화 하는 방법

- weak learner : 주로 overfitting의 우려가 없는 모형을 사용한다.
  - stump, KNN, Naïve Bayes ... 등

- strong learner : weak learner에 비해 성능이 월등한 모형을 사용한다.
  - Logistic Regression, SVM ... 등

- 훈련 및 최종 과정에 따라 voting, bagging, boosting, stacking 등으로 나뉜다.

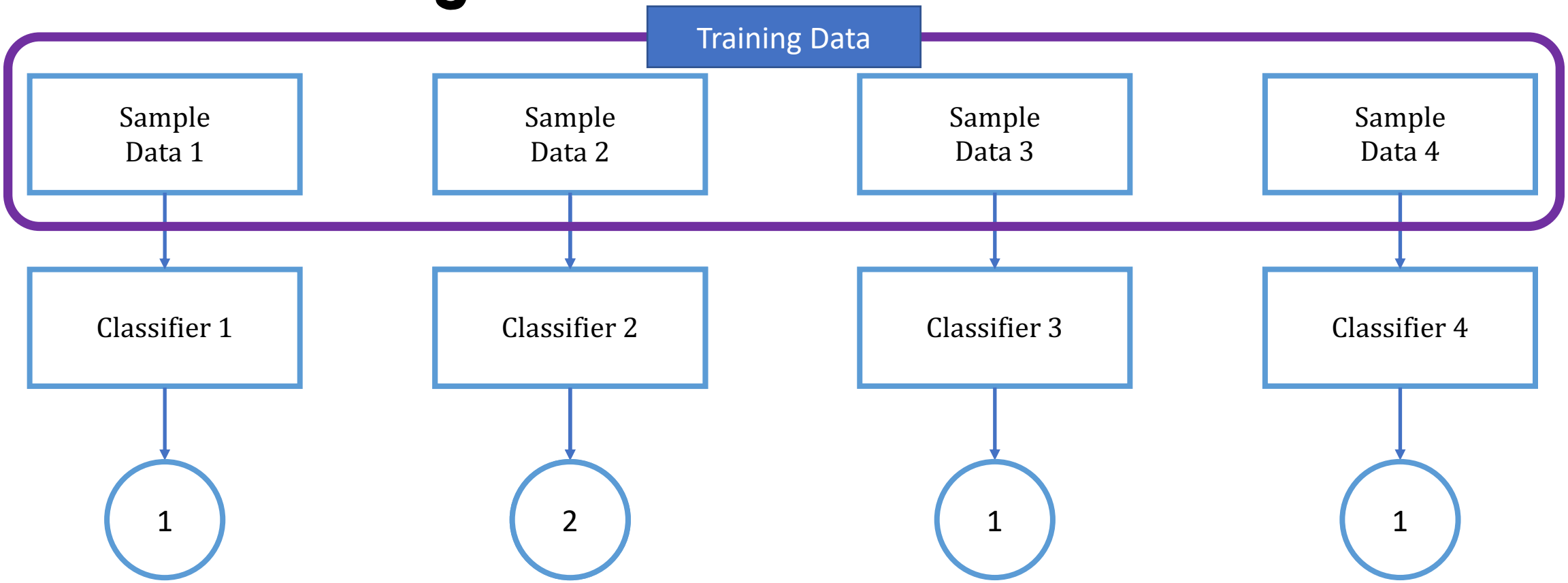
Voting

- Voting (투표)
  서로 다른 알고리즘을 가진 모델을 병렬로 결합
  최종 output이 continuous면 각 모델의 예측 값을 더해 평균을 출력
  최종 output이 categorical이면 hard voting / soft voting 중 하나를 선택

- Hard Voting
  : 예측한 output들 중 다수의 모델이 결정한 output을 최종 output으로 결정 (다수결)

- Soft Voting
  : 모델들의 output의 확률을 모두 더하고, 이를 평균을 낸 이후, 가장 높은 output을 결정

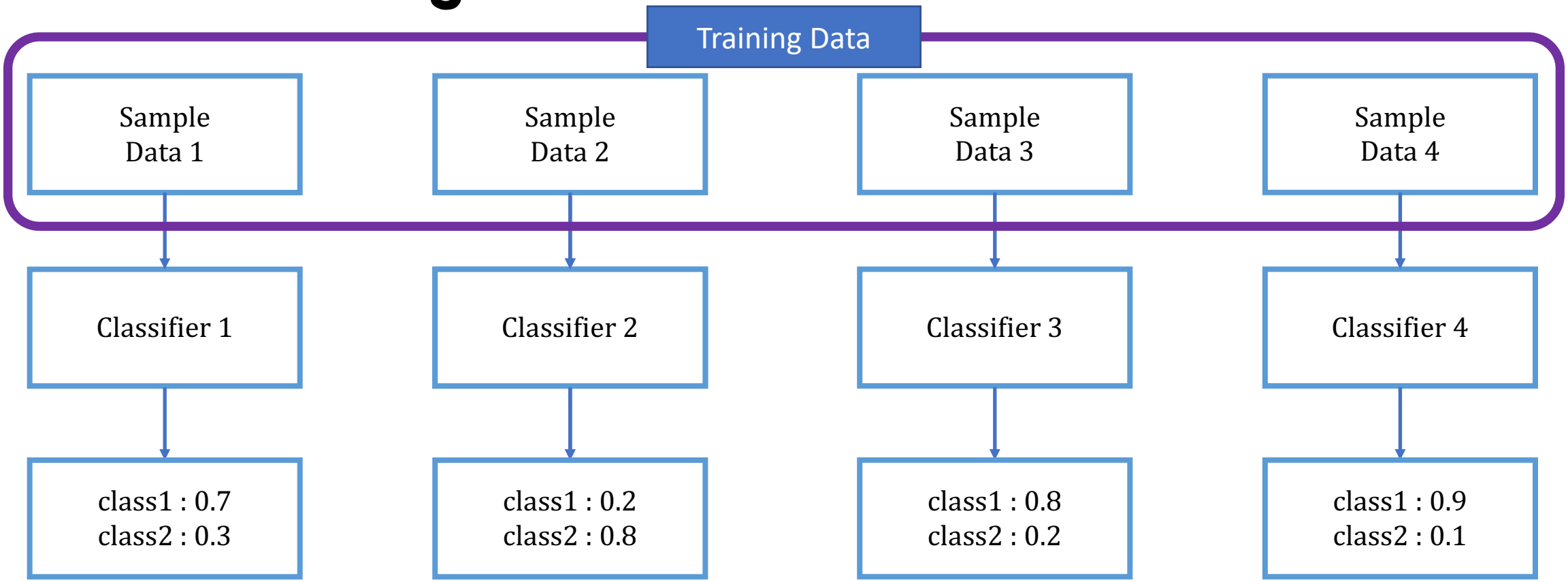
Hard Voting



- class1로 구분
  모델 1, 3, 4는 class 1로 구분 >> 모델 2는 class 2로 구분

python™

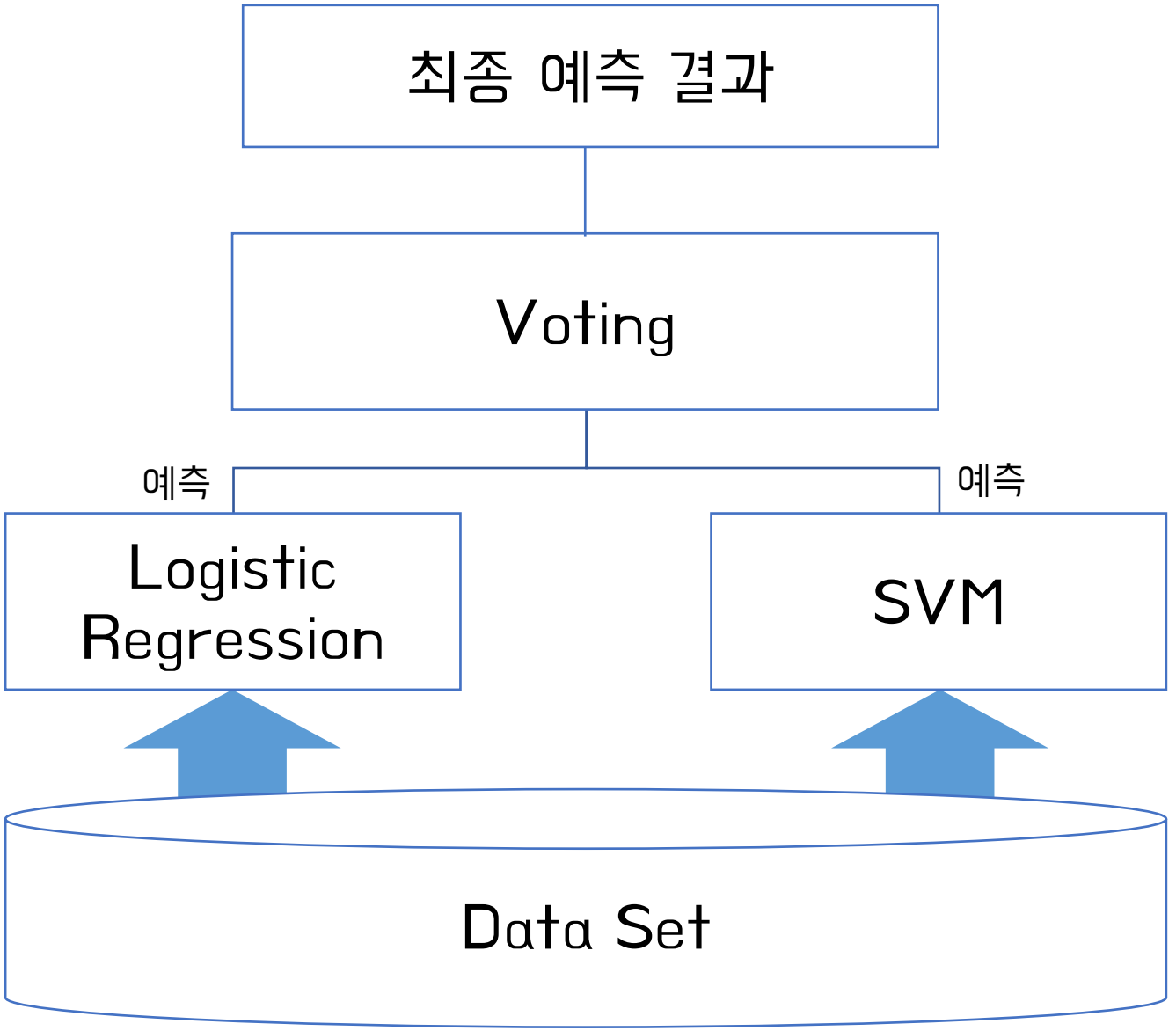
Soft Voting



- class1로 예측
  class1일 확률 : 0.65 >> class2일 확률 : 0.35

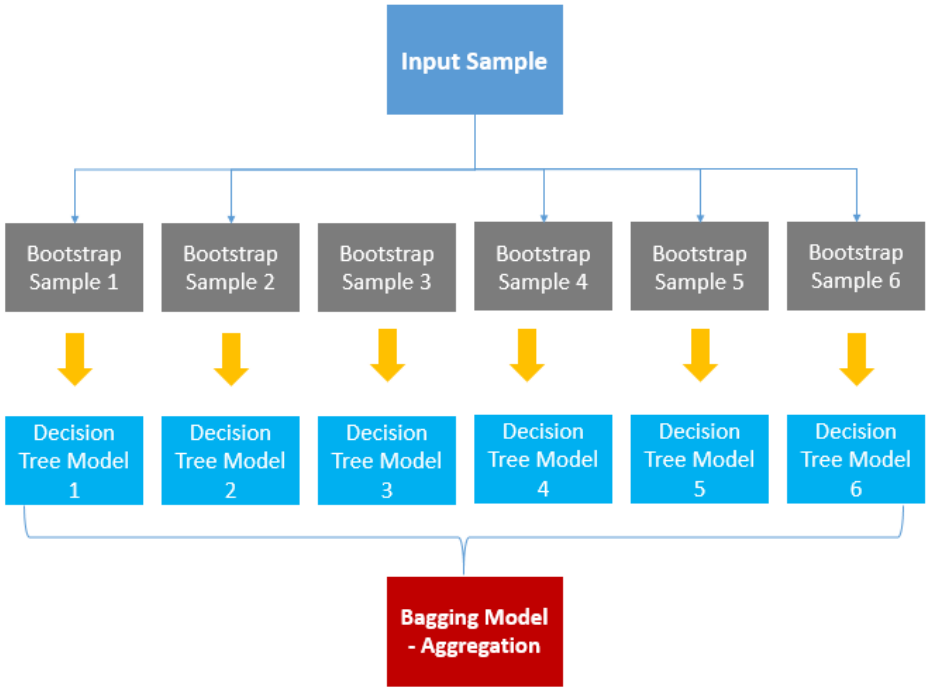
python™

Voting



python™

Bagging



- Bagging (Bootstrap aggregating)
  bootstrap : 복원추출
  aggregating : 집계

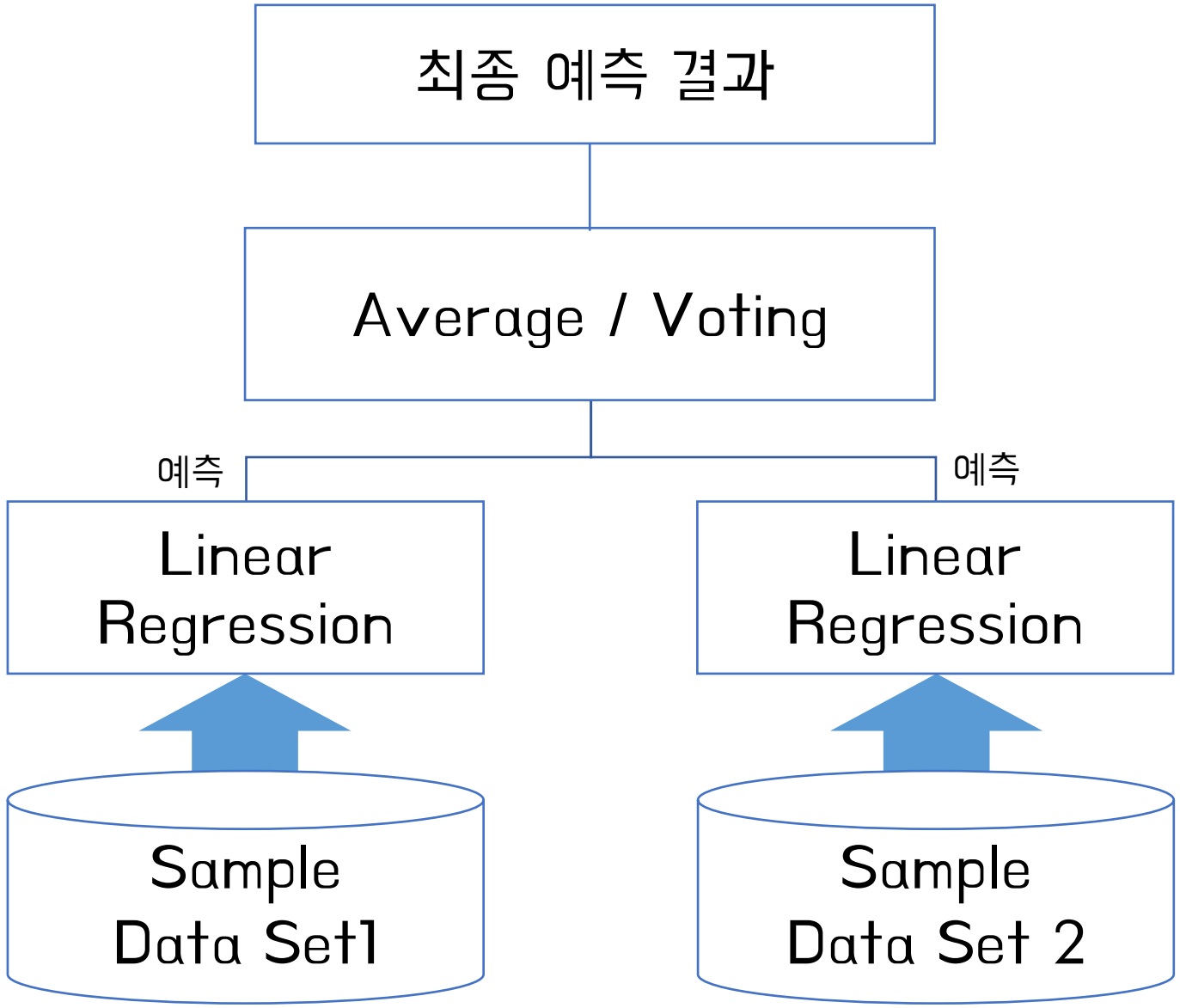
  sample을 여러 번 뽑아(bootstrap)
  같은 알고리즘을 가진 모델에 학습시켜
  결과를 집계(aggregate)

- Continuous : Average(평균)
- Categorical : Voting(투표)

- Random Forest가 대표적인 예

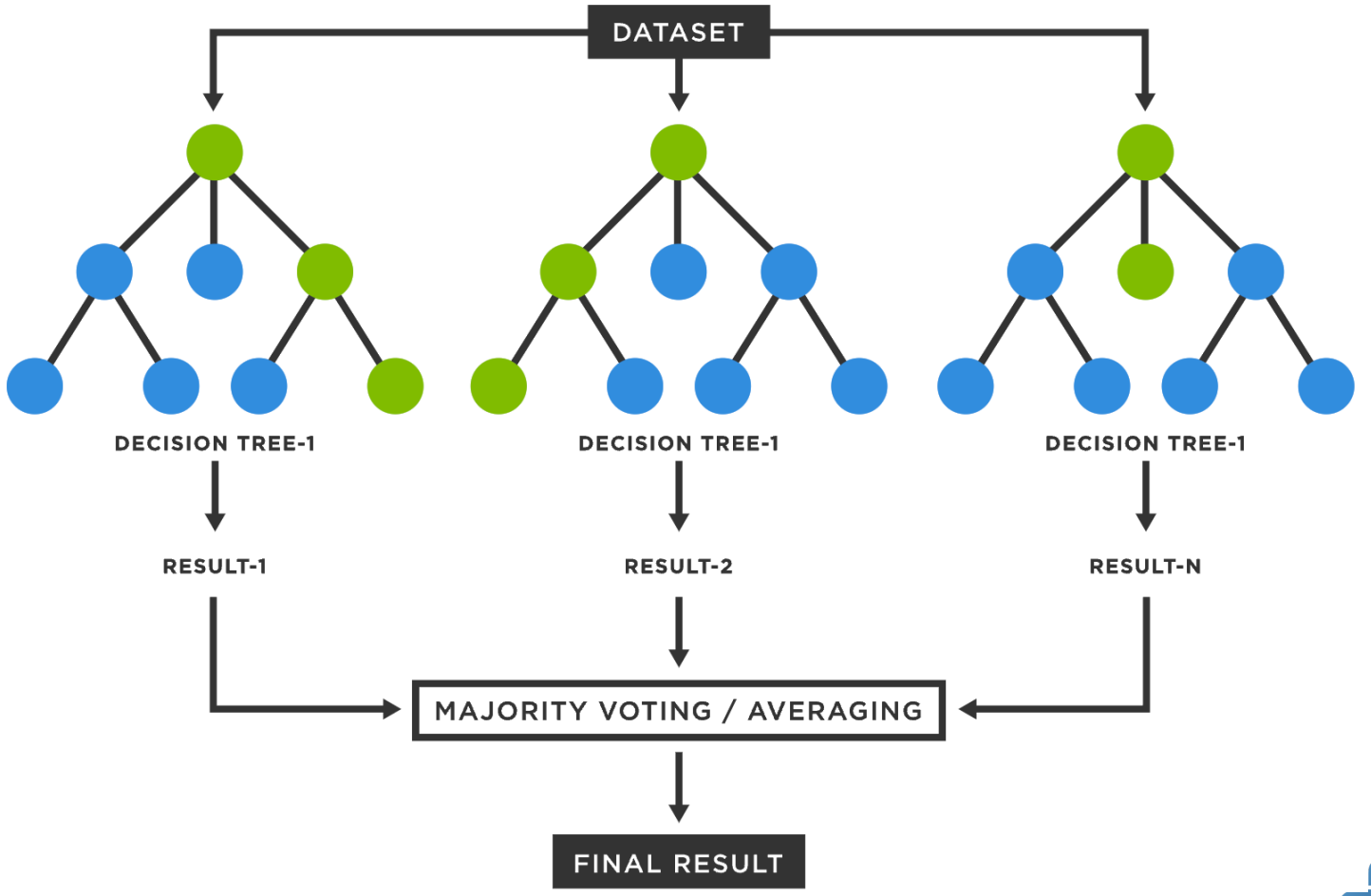
python™

Bagging



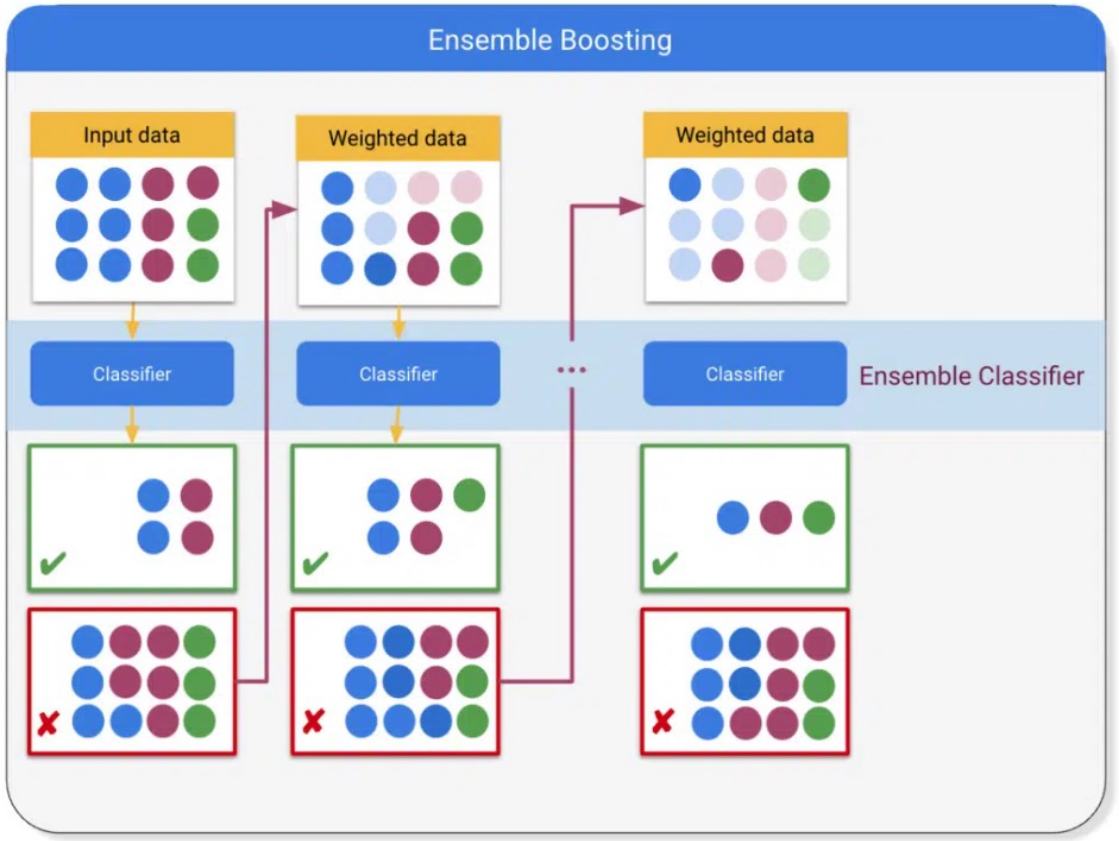
python™

Random Forest



python™

Boosting

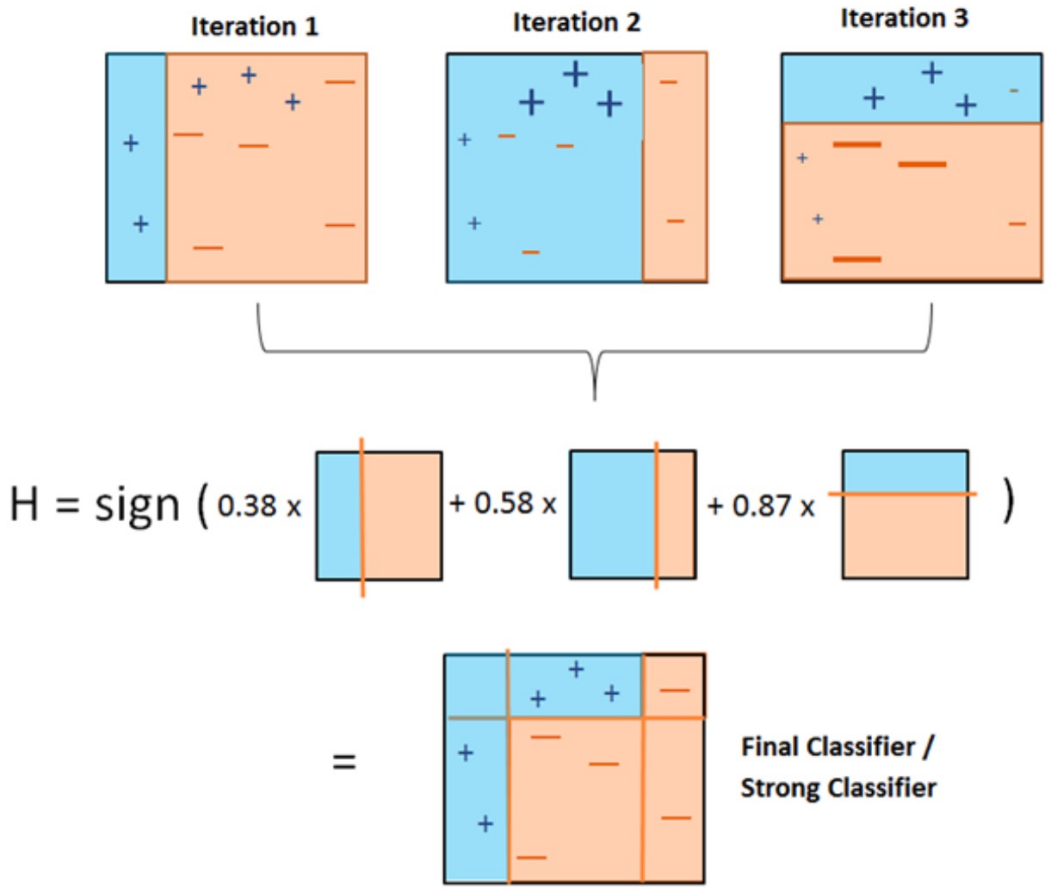


- Boosting
  여러 개의 weak learner가 순차적으로 학습,
  앞에서 학습한 learner의 예측이 틀린
  데이터에 대해 다음 learner가 가중치를 더해
  학습을 계속 이어가는 방식

- AdaBoost
- GBM(Gradient Boosting Machine)
- XGBoost
- LightGBM

python™

AdaBoosting



- AdaBoost(Adaptive Boosting)
  예측오차가 더 큰 표본에 더 큰 가중치를 부여하는 방식

  일반적으로 weak learner에 decision stump(를 자주 사용한다.

  stump : depth가 1인 Decision Tree 모형

  *Boosting 알고리즘의 시초.

python™

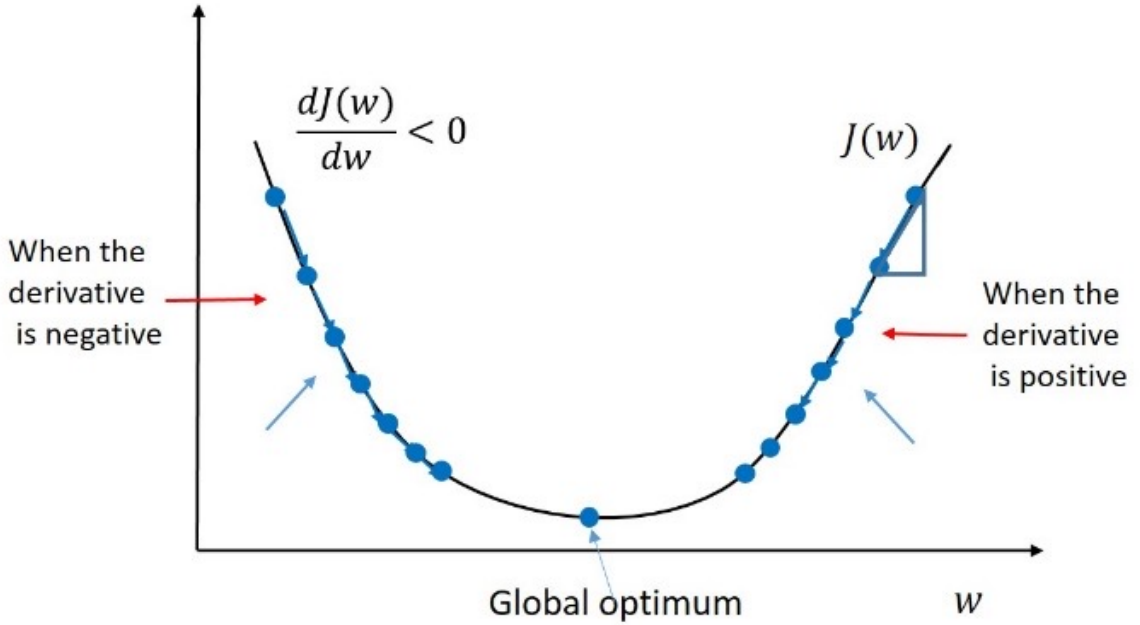
GBM

- Gradient Boosting Model
  Gradient Boosting을 Residual Fitting으로 이해하면 쉽다!

  1. 먼저 A라는 모델은 전체 데이터에서 target variable의 평균으로 예측값을 만든다
     그리고 실제값-예측값을 통하여 Residual을 구하고, 이는 B 모델이 학습할 정답이 된다.
  2. B모델은 A모델 학습에 사용됐던 features를 가지고 잔차를 맞추는 방식으로 학습한다.
     그리고 실제값-(A모델의 예측값 + LR*B모델의 예측값) = new residual을 구한다.
  3. C모델은 A모델 학습에 사용됐던 features를 가지고 새로운 잔차를 맞추는 방식으로 학습한다.
     그리고 실제값-(A모델의 예측값 + LR*B모델의 예측값 + LR*C모델의 예측값) = new residual을 구한다.
  4. 이를 계속 반복...

GBM

- 왜 Gradient Boosting인가?
  회귀문제에서는 loss function을 MSE로 자주 사용하는데, 이 MSE의 Gradient가 residual이 되기 때문이다.
  (계속 구해낸 새로운 Residual = Gradient)
  그리고 우리는 계속해서 Gradient를 줄이는 학습을 반복한다



python™

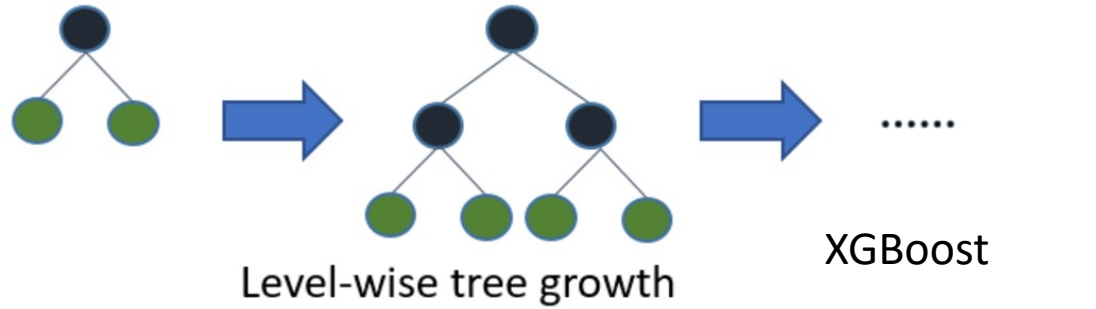
GBM 계열의 Ensemble 모델

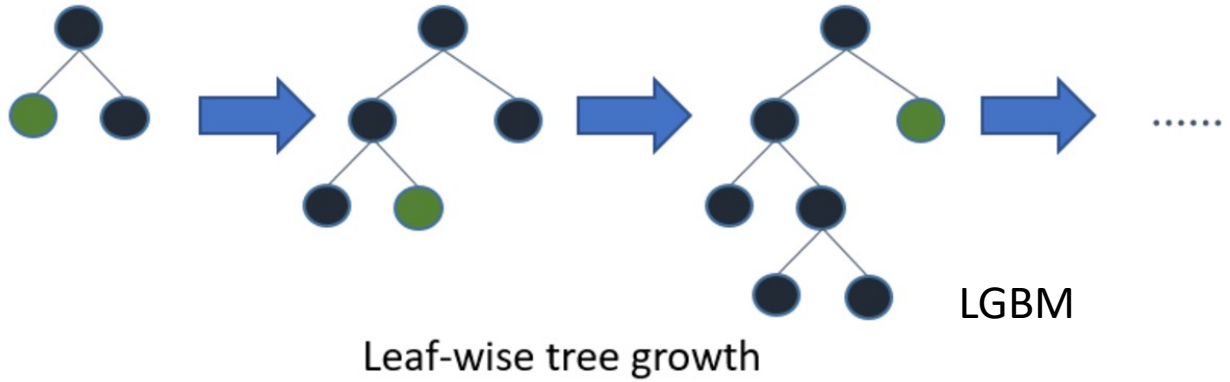
- GBM(Gradient Boosting Machine)
  Boosting에서 에러값들의 weight를 높이는 것 보다, gradient descent를 적용시킨 것

- XGBoost (Extreme Gradient Boosting)
  GBM 알고리즘을 병렬학습을 가능하게 하여 속도와 성능을 모두 잡은 알고리즘

- LGBM (Light Gradient Boosting Machine)
  Tree 모형이 수평이 아닌 수직으로 확장을 하면서 속도와 성능을 모두 잡은 알고리즘

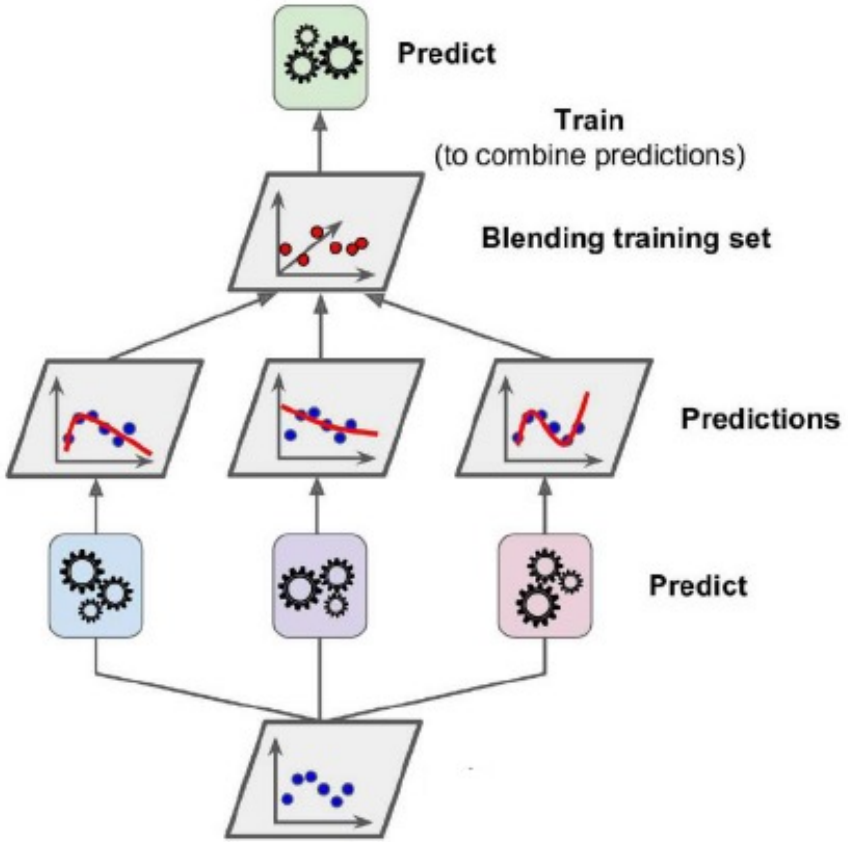
GBM, XGBoost, LGBM ...





python™

Stacking



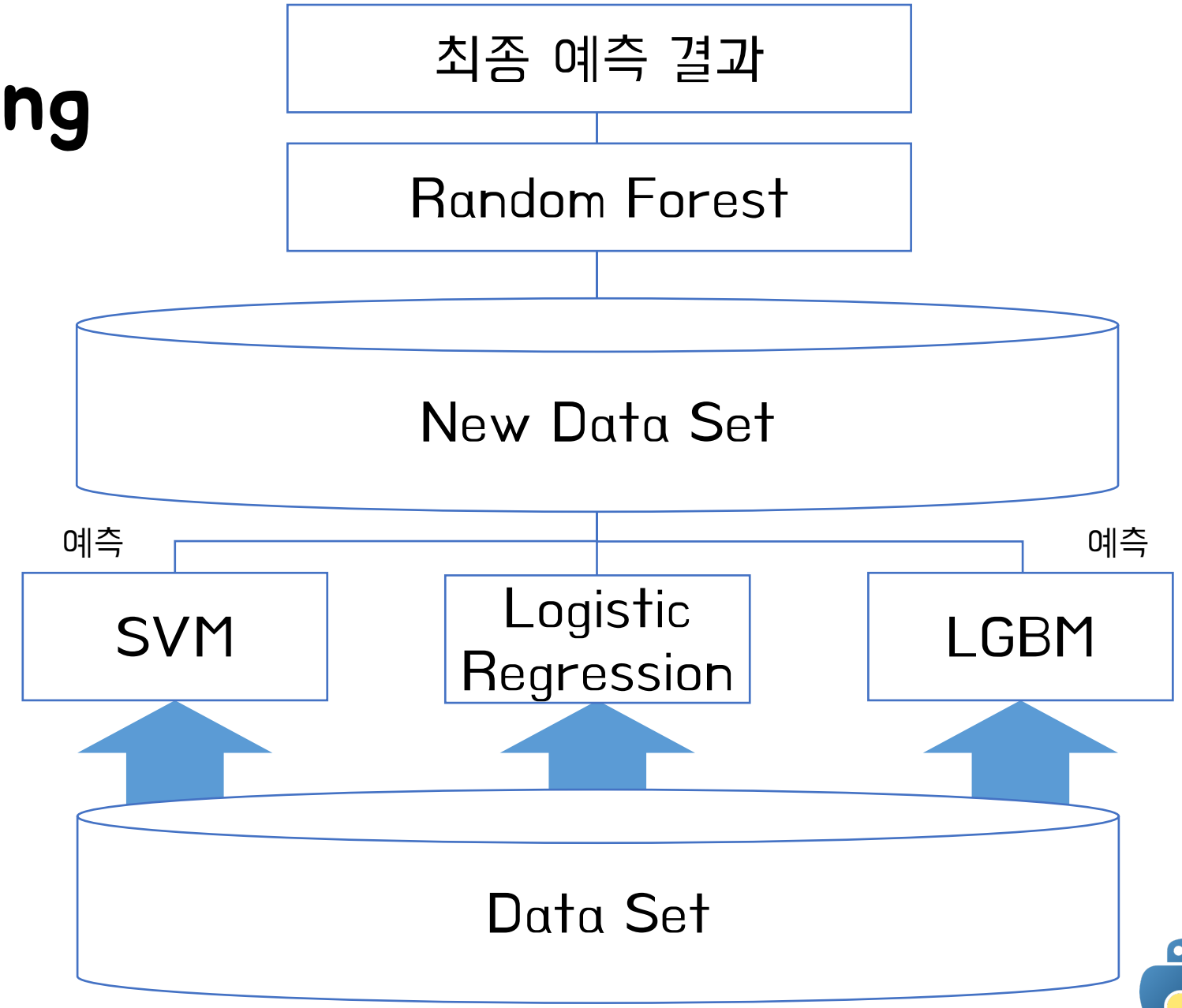
- Stacking
  개별적인 여러 개의 모델을 서로 결합하여
  예측 결과를 도출

  +

  개별 모델로 예측한 결과를
  다시 학습 데이터로 투입하여 다시 예측

python™

Stacking



python™

XGBoost 장단점

- 장점
  - GPU 지원 가능!!
  - 병렬처리로 인하여 GBM에 대하여 빠른 속도로 처리 가능
  - XGBoost 자체적으로 overfitting을 억제하는 regularization 기능이 존재
  - Regression / Classification 모두에서 뛰어난 성능
  - Early Stopping 기능 존재
  - 결측치를 내부적으로 처리해주는 기능이 존재

- 단점
  - 데이터셋이 작은 경우 overfitting 발생 가능
  - 해석이 어려움

LGBM 장단점

- 장점
  - 학습 시간이 매우 적게 걸리는 편
  - 메모리 사용량 또한 적게 걸리는 편
  - Categorical feature 들에 대해 자동 변환하는 기능이 존재
  - GPU 지원 가능 (근데 사전에 뭘 해야 하는 게 많음)

- 단점
  - 데이터셋이 작은 경우 overfitting 발생 가능
  - 해석이 어려움

python™