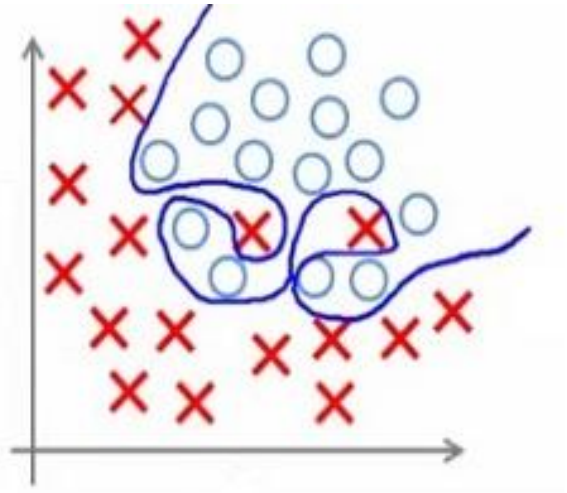
앙상블(Ensenbles)

- Bagging
- Boosting

결정트리의 최대 단점!!

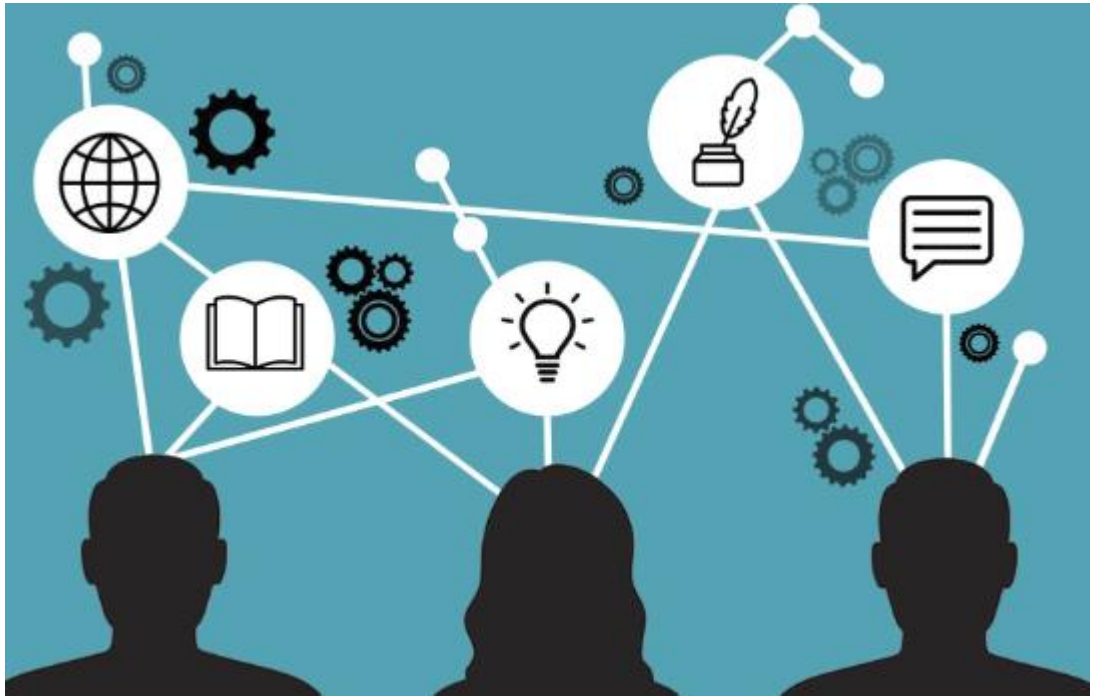
과적합(오버피팅)이 너무 잘 일어난다!!



Over-fitting

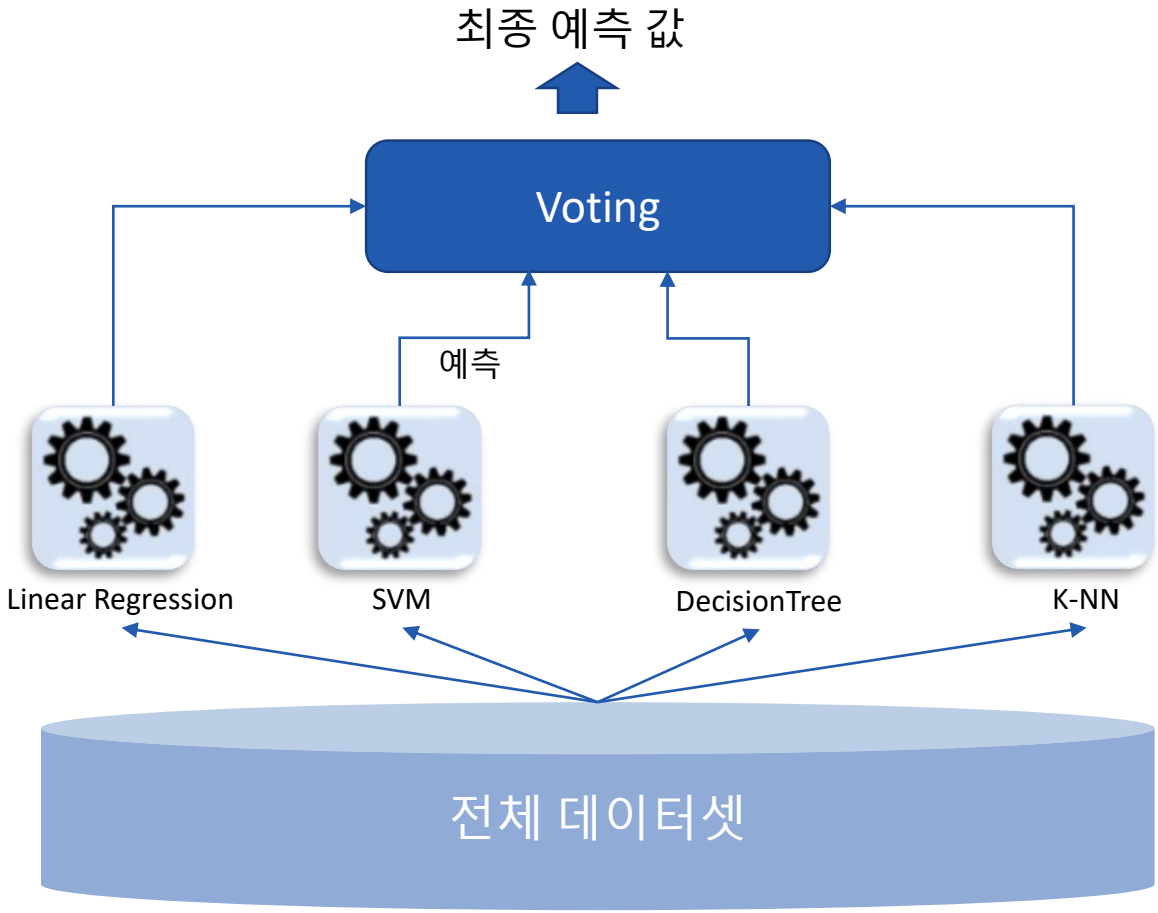
앙상블 학습 (Ensenble Learning)

여러 개의 약한 분류기(Classifier)를 생성하고, 그 예측을 결합함으로써 보다 정확한 최종 예측을 도출하는 기법



Ensenbles (앙상블) (1)– 보팅(Voting)

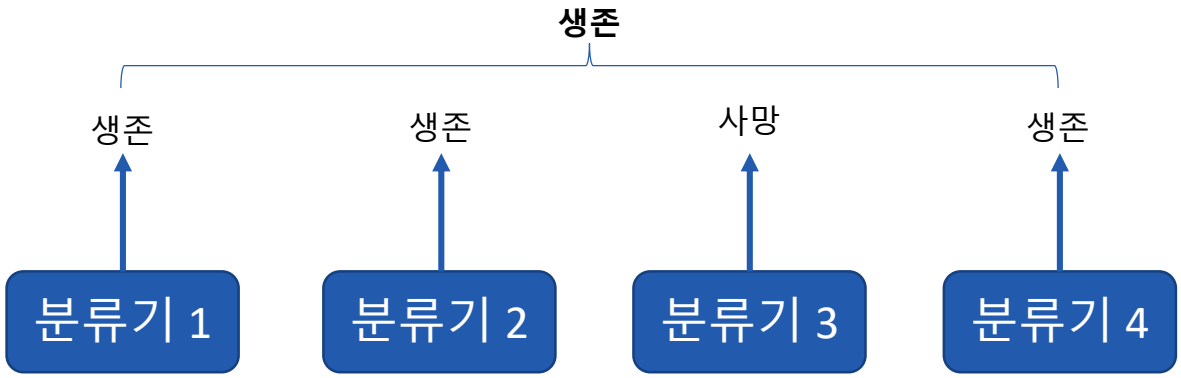
서로 다른 ML 알고리즘으로 여러 개의 분류기를 생성하고, 투표(Vote)를 통해 최종 예측 결과를 결정하는 방식



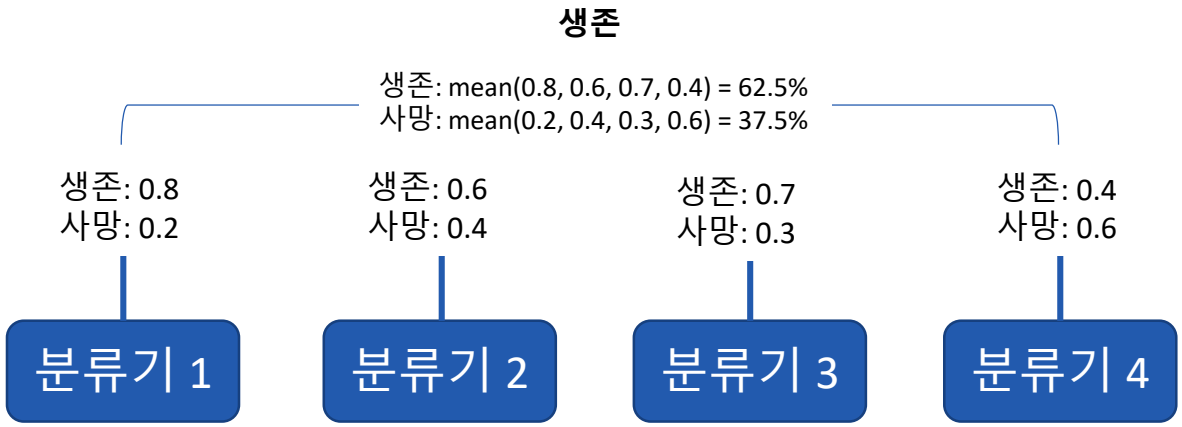
사이킷런에서는 **VotingClassifier** 클래스로 제공됨

하드 보팅(Hard Voting)과 소프트 보팅(Soft Voting)

하드 보팅(Hard Voting) – 분류기 간 다수결로 최종 분류 값 예측



소프트 보팅(Soft Voting) – 분류기들의 분류 확률들을 평균 내어 최종 분류 값 예측



Ensenbles (앙상블) (2) – 배깅(Bagging)

동양미래대학교

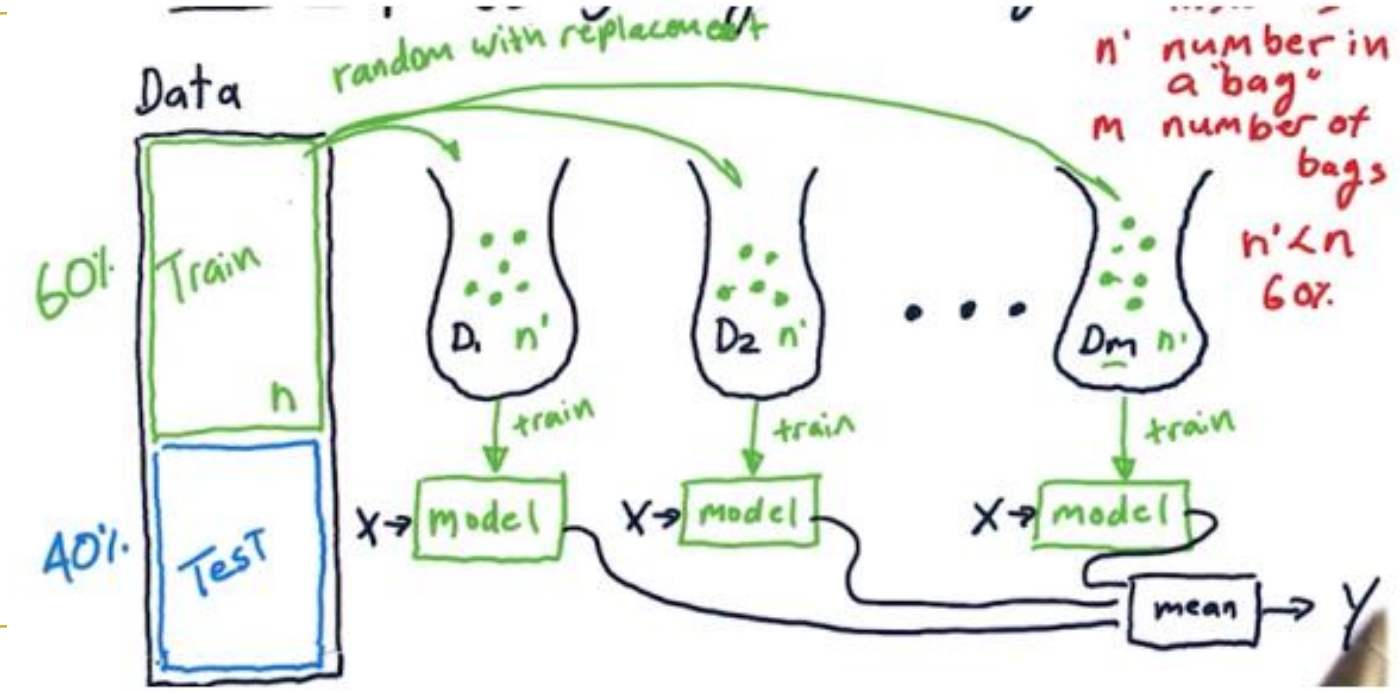
여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정

- 분류기는 모두 같은 유형의 알고리즘 기반
- 각각의 분류기는 서로 다른 데이터로 학습 (부트스트랩)
- 가장 대표적인 것은 **RandomForest** 알고리즘

1)

2)

3)



다른 데이

feature만 고

RandomForest의 주요 인자

동양미래대학교

n_estimators : integer, optional (default=10)

결정 트리의 개수. 값이 클수록 좋은 성능을 기대할 수 있지만, 학습 시간이 오래 걸림

The number of trees in the forest.

criterion : string, optional (default="gini")

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain. Note: this parameter is tree-specific.

max_features : int, float, string or None, optional (default="auto")

기본 값이 auto ➔ sqrt(# of features)

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.
- If float, then max_features is a percentage and int(max_features * n_features) features are considered at each split.
- If "auto", then max_features=sqrt(n_features).
- If "sqrt", then max_features=sqrt(n_features) (same as "auto").
- If "log2", then max_features=log2(n_features).
- If None, then max_features=n_features.

Note: the search for a split does not stop until at least one valid partition of the node samples is found, even if it requires to effectively inspect more than `max_features` features.

max_depth : integer or None, optional (default=None)

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

bootstrap : boolean, optional (default=True) Bootstrap 사용 여부

Whether bootstrap samples are used when building trees.

n_jobs : integer, optional (default=1) 병렬 처리할 job의 수

The number of jobs to run in parallel for both fit and predict. If -1, then the number of jobs is set to the number of cores.

모델의 성능에는 영향 x
학습 시간에 영향 o

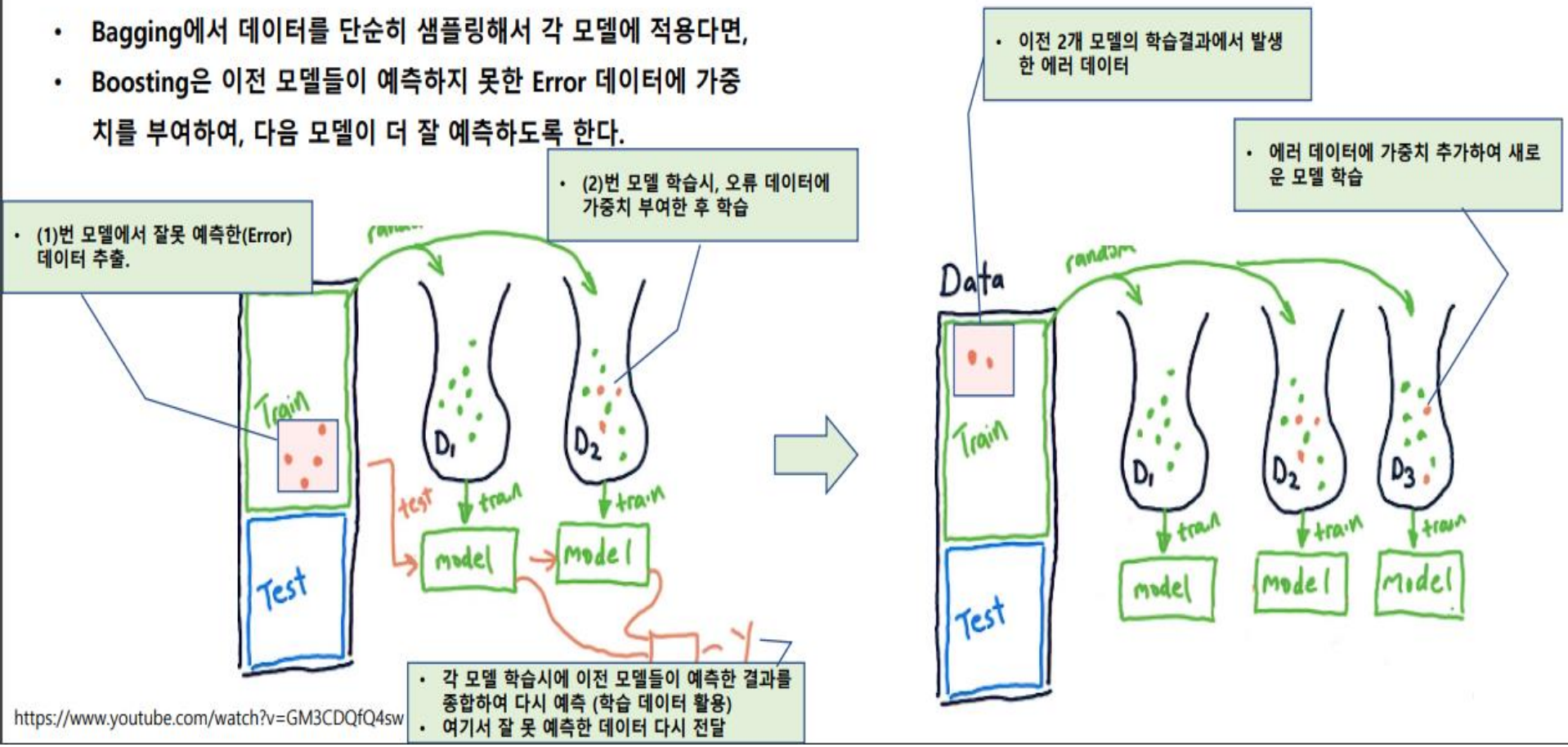
그 외에 min_samples_leaf, min_samples_split 등 결정 트리에서 과적합을 개선하기 위한 하이퍼 파라미터들을 동일하게 적용 가능

Ensenbles (앙상블) (3) – 부스팅(Boosting)

동양미래대학교

이전 트리의 오차를 보완하는 방식으로 순차적으로 트리를 개선시키는 방식
- 여러 개의 분류기가 순차적으로 학습을 수행
- 이전 분류기에서 예측이 틀렸던 데이터를 올바르게 예측할 수 있도록 다음 분류기에게는 가중치(weight)를 부여하면서 학습
- 예측 성능이 뛰어나지만 병렬 처리가 힘들어 학습 속도가 매우 느림

- Bagging에서 데이터를 단순히 샘플링해서 각 모델에 적용다면,
- Boosting은 이전 모델들이 예측하지 못한 Error 데이터에 가중치를 부여하여, 다음 모델이 더 잘 예측하도록 한다.



Boosting 알고리즘들

동양미래대학교

알고리즘	특징
AdaBoost	• 다수결을 통한 정답 분류 및 오답에 가중치 부여
GBM	• Loss Function의 gradient를 통해 오답에 가중치 부여
Xgboost	• GBM 대비 성능향상 • 시스템 자원 효율적 활용 (CPU, Mem) • Kaggle을 통한 성능 검증 (많은 상위 랭커가 사용)
Light GBM	• Xgboost 대비 성능향상 및 자원소모 최소화 • Xgboost가 처리하지 못하는 대용량 데이터 학습 가능 • Approximates the split (근사치의 분할)을 통한 성능 향상

Scikit-learn 에서 제공

- Scikit-learn에서 제공X
- 별도의 라이브러리 설치
- 최근 가장 많이 활용되는 boosting 방법들

GBM(Gradient Boosting Machine) 하이퍼 파라미터 튜닝

파라미터명[기본값]	설명
loss [deviance]	경사 하강법에서 사용할 비용함수 지정. 특별한 이유가 없다면 기본값 사용
learning_rate [0.1]	GBM이 학습을 진행할 때마다 적용하는 학습률. (0부터 1사이의 값) - 오류값을 보정해 나가는 속도. . 값이 작으면 예측성능이 높아질 가능성이 높으나, 학습 시간이 오래 걸림. . 값이 너무 작으면 weak learner의 반복이 완료되어도 최소 오류 값을 찾지 못할 수 있음. . 값이 크면, 최소 오류값을 찾지 못하고 그냥 지나쳐 버려 예측 성능이 떨어질 가능성이 높으나, 학습 속도가 빠름.
n_estimators [100]	weak learner의 개수. - 값이 클수록 예측 ㅓㅅㅇ능이 일정 수준까지는 좋아질 수 있으나, 학습 시간이 오래 걸림
Subsample [1]	Weak learne가 학습에 사용할 데이터의 샘플링 비율. - 기본값 = 1: 전체 학습 데이터를 기반으로 학습 - 과적합이 우려되는 경우, 1보다 작은 값으로 설정하는 것을 권장함.

Random Forest(Bagging)와 GBM(Boosting) 방법 비교

동양미래대학교

	Random Forest (Bagging)	GBM (Boosting)
장점	✓ 모델 성능이 뛰어남. ✓ 파라미터 최적화를 많이 하지 않아도 훌륭한 성능을 보장 ✓ 병렬 처리 ✓ 데이터 스케일을 맞출 필요도 없음	✓ 지도 학습에서 가장 강력하고 널리 사용되는 모델 중 하나 ✓ 데이터 스케일을 맞출 필요도 없음 ✓ 파라미터 최적화 시, 매우 훌륭한 성능
단점	✓ 메모리 사용량이 많음.	✓ 파라미터 최적화가 필수. (미 최적화 시, 성능 보장 X) ✓ 학습 시간이 김. ✓ 병렬 처리가 어려움

XGBoost (eXtreme Gradient Boost)

트리 기반의 앙상블 학습에서 최근 가장 각광받고 있는 알고리즘

- Kaggle에서 상위 데이터 과학자들이 많이 활용하면서 널리 알려짐
- GBM 기반으로 구현 & GBM 문제점 해결
- 핵심 라이브러리는 C/C++로 구현. 패키지명은 xgboost.
- 출시 초기에는 scikit-learn과 호환되지 않아, scikit-learn에서 제공해 주는 다양한 유틸리티 기능과 함께 사용 X
- 지금은 scikit-learn과 연동 가능한 래퍼 클래스(Warpper Class)를 제공. 클래스명은 XGBClassifier, XGBRegressor.

XGBoost 주요 장점

동양미래대학교

뛰어난 예측 성능	- 분류와 회귀 영역에서 뛰어난 예측 성능 발휘
GBM 대비 빠른 수행 시간	- 일반적인 GBM은 순차적으로 weak learne가 가중치를 증감하는 방법으로 학습하기 때문에 병렬 수행이 불가능하여 학습 속도가 느림 - XGBoost는 병렬 수행 및 다양한 기능으로 빠른 학습 성능 보장. (GBM 대비 빠른 것이지, 결정트리, 랜덤포레스트 등 타 ML 알고리즘에 비해서는 느림)
과적합 규제	- 일반적인 GBM은 과적합 규제 기능 X
나무 가지치기 (Tree Pruning)	- 일반적인 GBM은 분할 시 부정 손실이 발생하면, 분할 수행 X. (부정 손실이란 트리를 분할하면 오히려 엔트로피나 지니지수와 같은 정보 이득이 줄어듬을 의미함.) - XGBoost는 max_depth까지 진행한 뒤 loss function 에서의 개선이 일정 threshold에 못미칠 경우까지 역방향으로 pruning과정을 수행.
자체 내장된 교차 검증	- 반복 수행 시마다 내부적으로 교차검증을 수행 - 조기 중단 기능 있음. (지정된 반복 횟수가 아니라 교차검증을 통해 최적화되면 반복을 멈춤)
결손값 자체 처리	결손값을 자체 처리할 수 있는 기능

XGBoost Parameters - 일반 파라미터 (General parameters)

동양미래대학교

파라미터명[기본값]	설명
booster [gbtree]	모델의 종류 - **gbtree**: tree-based models - **gblinear**: linear models
silent [0]:	메시지 출력 여부. If Silent = 1, 로그 출력 X
nthread [CPU의 전체 스레드 사용]	- CPU의 실행 스레드 개수 - 병렬 처리를 위한 인자 - 미 설정 시, 자동으로 시스템에서 사용할 수 있는 최대의 CPU를 사용함. - 전체 CPU를 사용하지 않고, 일부만 사용해 ML애플리케이션 구동 시에 변경

공식 사이트: https://xgboost.readthedocs.io/en/latest/parameter.html
XGBoost 최적화 관련 : https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

XGBoost Parameters – 부스터 파라미터 (Booster parameters)

파라미터명 [기본값]	설명
learning_rate (eta) [0.3]	- learning rate와 동일. XGBClassifier를 이용할 경우, learning_rate 인자로 사용. - 각 스텝 별로 weight를 감소시키는 정도. 클수록 빨리 감소시킴. - 0과 1 사이의 값 지정 가능. 일반적으로 [0.01, 0.2] 정도로 설정
min_child_weight [default=1]	- Child의 weights의 최소 합 - 더 높은 값을 설정할수록, overfitting을 방지함. - 하지만 너무 높은 값으로 설정하면 under-fitting 발생
max_depth [default=6]	트리의 최대 Depth Typical values: [3, 10]
min_split_loss(gamma) [default=0]	- 트리의 리프 노드를 추가적으로 확장(분할)할 것인지 결정할 최소 손실 감소 값. - 해당 값보다 큰 손실(loss)이 감소된 경우, 리프 노드 분리 - 값이 클수록 과적합 방지.
sub_sample(subsample) [default=1]	- 트리 확장 시, 고려되는 샘플의 비율 - Overfitting 발생 시, 이 값을 낮추면 overfitting을 방지할 수 있다. - 너무 낮은 값은 under fitting 유발할 가능성이 있음.
colsample_bytree[default=1]	- Max_features와 동일 - 보통 0.5~ 1 설정
reg_lambda(lambda) [default=1]	- L2 regularization 적용값 - 피처 개수가 많을 경우 검토. 자주 사용되진 않음. - 과적합 감소 효과
reg_alpha(alpha) [default=0]	- L1 regularization 적용값 - 피처 개수가 많을 경우 검토. - 과적합 감소 효과

XGBoost Parameters – 학습 파라미터 (Learning Task parameters)

파라미터명	설명
objective [default=reg:squarederror]	손실 함수(Loss function) 을 정의. 주로 이진/다중 분류 여부에 따라 선택. - **reg:squarederror**: regression with squared loss - **binary:logistic** –logistic regression for binary classification, returns predicted probability (not class) - **multi:softmax** –multiclass classification using the softmax objective, returns predicted class (not probabilities) - **multi:softprob** –same as softmax, but returns predicted probability of each data point belonging to each class. - ….
eval_metric [default: rmse for regression **error** for classification]	검증에 사용되는 함수를 정의. - **rmse**: root mean square error - **mae**: mean absolute error - **logloss**: negative log-likelihood - **error**: Binary classification error rate (0.5 threshold) - **merror**: Multiclass classification error rate - **mlogloss**: Multiclass logloss - **auc**: Area under the curve - ,…
seed [default=0]	랜덤 값 생성을 위한 Seed값

XGBoost의 과적합 개선 팁.

과적합을 방지하는 방법은 학습 모델을 단순화 하는 방향으로 하이퍼 파라미터 튜닝

- learning_rate 값은 낮춤과 함께 n_estimators는 높여줌.
- max_depth 값을 낮춤
- min_child_weight 값을 높임
- min_split_loss 값을 높임
- sub_sample 값을 줄임
- colsample_bytree 값을 줄임