**4. 분류 ( Classification )**

**- 분류 알고리즘**

**Bayes 통계와 생성 모델에 기반한 Naïve Bayes** : 예

**독립변수와 종속변수의 선형 관계성에 기반한 Logistic Regression**

**데이터 균일도에 따른 규칙 기반의 Decision Tree(결정 트리)**

**개별 클래스 간의 최대 분류 마진을 효과적으로 찾아주는 Support Vector Machine**

**근접 거리를 기준으로 하는 Nearest Neighbor(최소 근접) 알고리즘**

**심층 연결 기반의 Neural Network(신경망)**

**서로 다른(or 같은) 머신러닝 알고리즘을 결합한 Ensemble(앙상블)**

**- 결정 트리와 앙상블**

**결정트리 Decision Tree** : 데이터에 있는 규칙을 학습을 통해 자동으로 찾아내 트리 Tree 기반의 분류 규칙 생성( If-Else 기반 규칙)

데이터의 스케일링 정규화의 영향이 매우 적음

하지만 예측 성능향상을 위해 복잡한 규칙 구조를 가져야 하며, 이로 인해 과적합(Overfitting)이 발생해 예측성능이 오히려 저하될 수 있음

**트리 분할을 위한 데이터의 균일도 :** 한가지의 데이터로만 채워진 상태 : 가장 균일

**- 결정 트리와 앙상블**

**정보이득 Information Gain** : 1-엔트로피 지수 /결정트리는 **정보이득이 높은속성을** 기준으로 분할

\*엔트로피 : 데이터 집합의 혼잡도

(서로 **다른 값**이 섞여 있으면 엔트로피 **↑** // 서로 **같은 값**이 섞여 있으면 엔트로피 **↓** )

**지니 계수** : 0이 평등 1로 갈수록 불평등 // 머신러닝 🡪 지니계수 낮을수록 데이터 균일

**지니 계수가 낮은 속성을** 기준으로 분할

**앙상블 Ensemble** : 위의 단점이 앙상블에서는 장점으로 적용

약한 학습기(Week Learner)(예측 성능이 떨어지는 학습 알고리즘)를 결합해 확률적 보완과 오류가 발생한 부분에 가중치를 계속 업데이트하며, 예측 성능을 향상시킴. (GBM, XGBoost, LightGBM)

**- 결정 트리의 규칙 노드 생성 프로세스**

텍스트이(가) 표시된 사진

자동 생성된 설명

**- 결정 트리의 특징**

**장점** : 쉽다 직관적이다 피처의 스케일링,정규화 등 사전 가공영향도가 크지 않다

**단점** : 과적합으로 알고리즘 성능이 떨어진다.

이를 극복하기 위해 트리의 크기를 사전에 제한하는 튜닝 필요

**- 결정 트리의 Feature 선택 중요도 dt\_clf.feature\_importances\_**

**- 결정 트리 과적합**

트리 생성 제약을 주어 데이터 과적합 해소 가능

**- 앙상블 Ensemble Learning**

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

**단일 분류기보다 신뢰성이 높은 예측 값을 얻는 것이 목표**

**Voting(보팅)**

**Bagging(배깅)** – 랜덤 포레스트(Random Forest) 알고리즘

**Boosing(부스팅)** – 에이다부스팅, 그래디언트부스팅, XGBoost, LightGBM

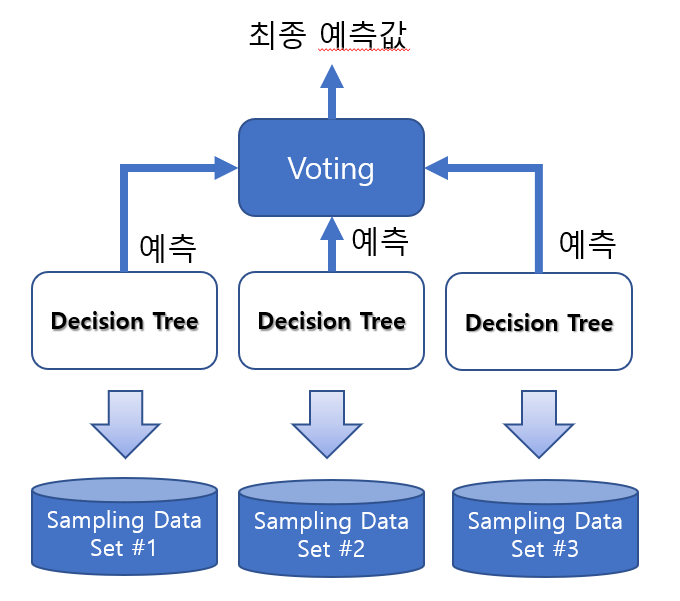
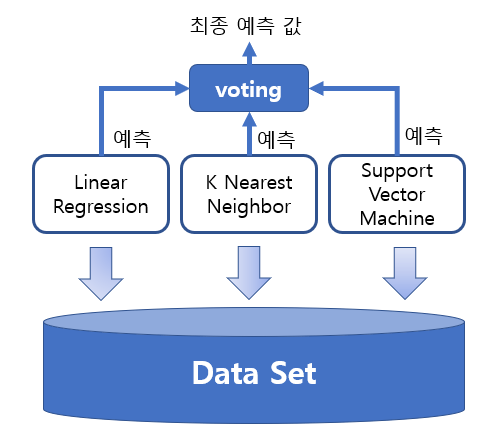
정형 데이터의 분류,회귀에서는 GBM 부스팅 계열의 앙상블이 전반적으로 높은 예측 성능을 나타냄.

**- 앙상블 Ensemble Learning 의 특징**

* 단일 모델의 약점을 다수의 모델들을 결합하여 보완
* 뛰어난 성능을 가진 모델들로만 구성하는 것보다 성능이 떨어지더라도 **서로 다른 유형의 모델을 섞는 것**이 오히려 전체 성능이 도움이 될 수 있음.
* 랜덤 포레스트 및 뛰어난 부스팅 알고리즘들은 모두 **결정 트리 알고리즘을 기반** 알고리즘으로 적용함.
* 결정 트리의 단점인 과적함(오버 피팅)을 수십~수천개의 많은 분류기를 결합해 보완하고 장점인 직관적인 분류 기준은 강화됨.

**- 보팅(Voting)과 배깅(Bagging) 개요**

보팅과 배깅은 모두 여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정하는 방식



**[ 보팅(Voting) ] [ 배깅(Bagging) ]**

* **Votin(보팅)** : 서로 다른 알고리즘을 가진 분류기를 결합 // **VotingClassifier**

**Hard Voting :** classifier 간 다수결로 최종 class 결정

**Soft Voting** : classifie들의 class 확률의 평균으로 결정

predict\_proba()를 이용해 class 별 확률 추출

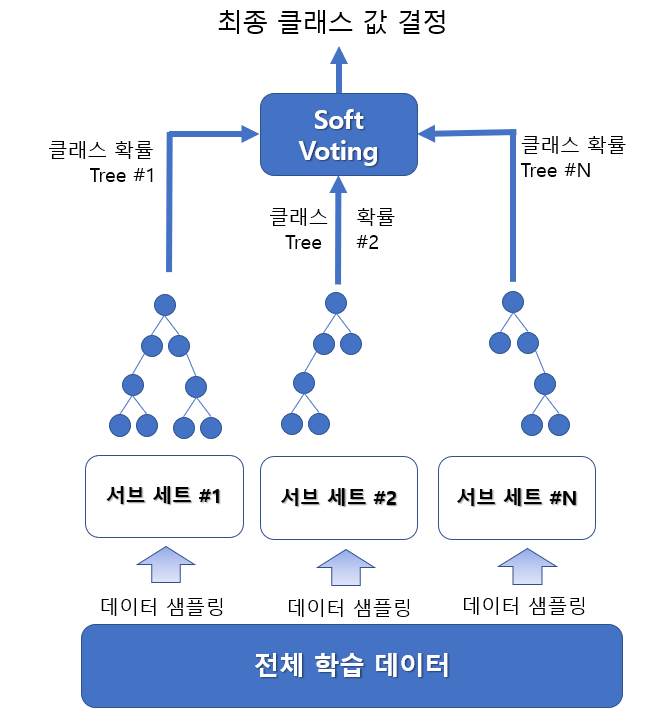
일반적으로 하드 보팅보다는 **소프트 보팅이 예측 성능이 상대적으로 우수하여** 주로 사용됨.

* **Bagging(배깅)** : 각각의 분류기가 모두 같은 유형의 알고리즘 기반이지만, 데이터 샘플링을 서로 다르게 가져가면서 학습을 수행

여러 개의 결정 트리 분류기가 배깅 방식으로 **각자의 데이터를 샘플링**하여 **개별적으로 학습**한 뒤 **최종적으로 모든 분류기가 소프트 보팅을 통해 예측을 결정함.**

**랜덤 포레스트 (대표적인 알고리즘)**

앙상블 알고리즘 중 비교적 빠른 수행속도를 가지며 높은 예측 성능을 보임



* **랜덤 포레스트의 부트스트래핑 분할 RandomForesClassifier()**

랜덤 포레스트의 개별 트리가 학습하는 데이터 세트는 전체 데이터에서 일부가 중첩되게

샘플링된 데이터 세트임.

**Bootstrapping(부트스트래핑) :** 평균의 신뢰도

여러 개의 데이터 세트를 중첩되게 분리하는 분할 방식

* **RandomForesClassifier()의 하이퍼 파라미터**

**n\_estimators :** 약한 학습기의 개수 (default : 10 ) 많이 설정할수록 무조건 좋은 것은 아니다.

**max\_features :** ‘sqrt’ 와 같음. 즉 랜덤 포레스트의 트리를 분할할 때 sqrt 만큼만 참조함.

Ex) 전체가 16개이면 피처 4개만 참조

**max\_depth**

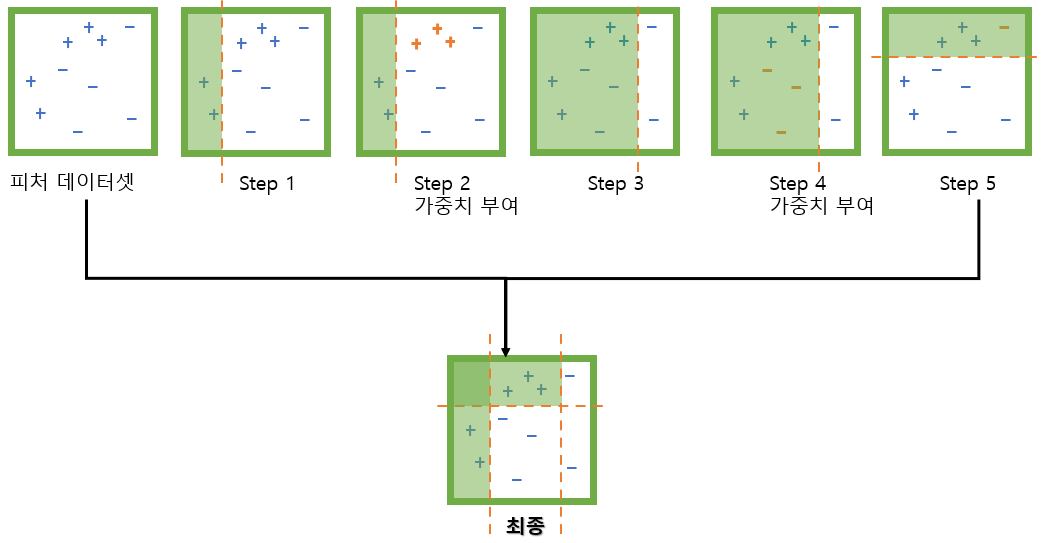
**min\_samples\_leaf**

* **부스팅(Boosting)**

여러 개의 약한 학습기(week learner)를 순차적으로 학습-예측하며 **잘못 예측한 데이터에 가중치 부여로** 오류를 개선해 나가면서 학습하는 방식

대표적인 구현 : AdaBoost(Adaptive boosting) , 그래디언트 부스트

* **AdaBoosting : 가중치를 지정**



**GBM(Gradient Boost Machine) GradientBoostingClassifier()**

가중치 업데이트 시, **경사 하강법 (Gradient Descent) 이용**

**\*경사 하강법 :** 분류의 실제 결과 값 y // 피처 X1, X2, X3,…. // 피처에 기반한 예측함수 F(x)

***(오류) h(x) = y – F(x)*** // 오류 값 = 실제 값 – 예측 값

**오류 값을 최소화**하며 반복적으로 ***가중치 값을 업데이트*** 하는 것

**learing\_rate :** GBM이 학습을 진행할 때마다 적용하는 학습률 default = 0.1

Weak learner가 순차적으로 오류 값을 보정하는 데 적용하는 계수

너무 작게 설정 🡪 최소 오류 값을 찾지 못할 수도

너무 크게 설정 🡪 최소 오류 값을 찾지 못하고 예측 성능이 떨어질수도 but 빠른 수행 가능

**n\_estimators :** weak learner 의 개수 default = 100

**subsample :** weak learner가 학습에 사용하는 데이터 샘플링 비율