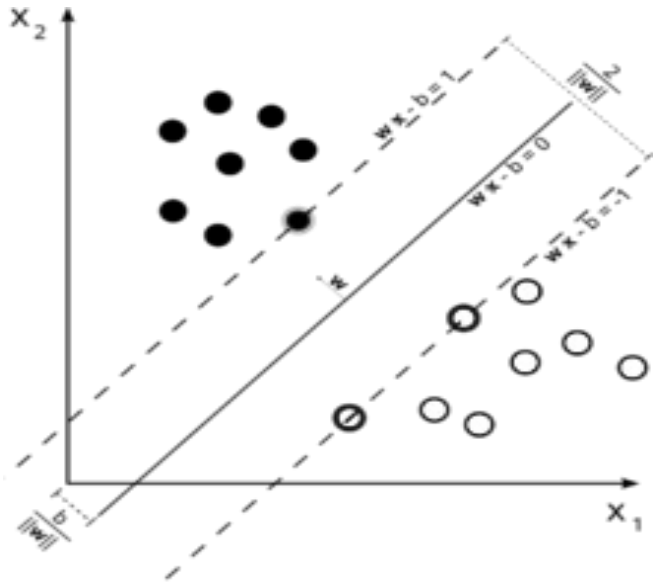
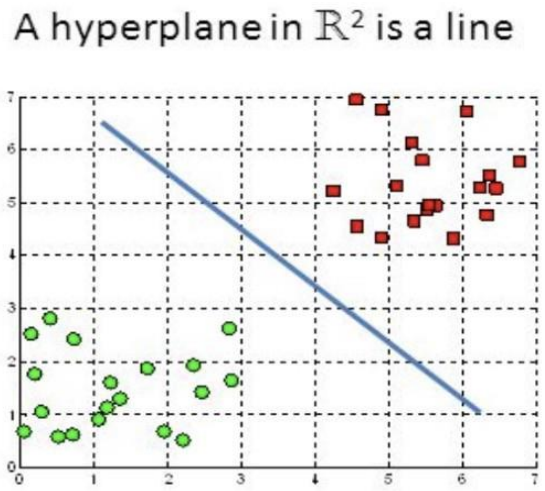
Support Vector Machine

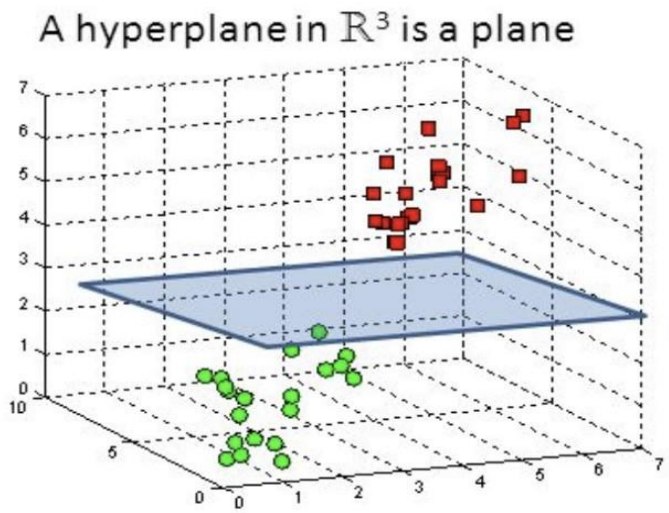
서포트 벡터 머신(SVM)

서포트 벡터 머신은 여백(Margin)을 최대화하는 초평면(Hyperplane)을 찾는 지도 학습 알고리즘

딥러닝의 대두 이전까지 분류, 회귀에 모두 사용할 수 있는 매우 강력한 모델







서포트 벡터 머신(SVM)

장점

- 상대적으로 데이터의 이해도가 떨어져도 사용이 용이함
- 예측 정확도가 통상적으로 높음

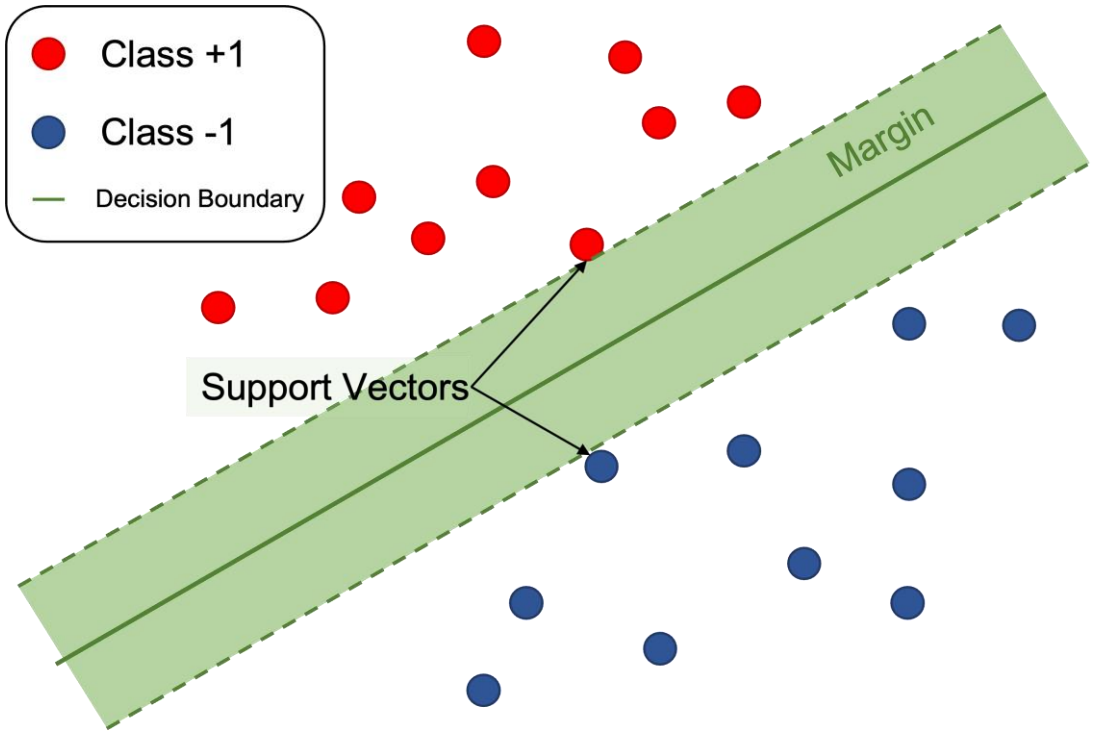
단점

- C(에러에 부여하는 가중치)를 결정해야 함
- 파라미터의 결정과 모형의 구축에 시간이 오래걸림

여백(Margin)의 의미

주어진 데이터가 오류를 발생시키지 않고 움직일 수 있는 최대공간

분류 문제와 회귀 문제 각각의 문제에 따라 정의가 다름



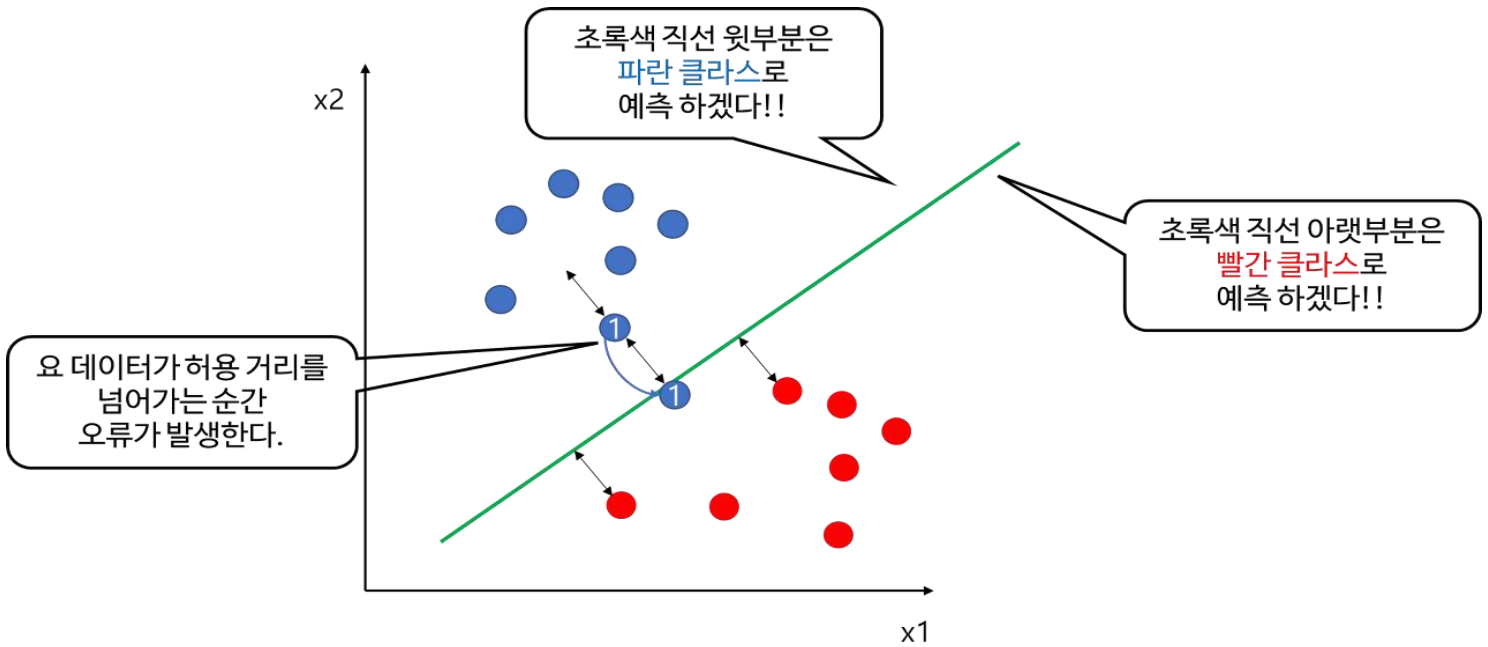
분류에서의 여백(Margin)

데이터가 2차원이며, 라벨은 파란색과 빨간색 두개의 클래스가 있다고 가정

이를 분류하는 초평면은 초록색 직선이고, 1번 데이터를 기준으로 해당 데이터가 직선과 수직으로 움직인다면

1번 데이터가 직선을 초과해 움직이면, 해당 데이터를 직선이 빨간색 데이터로 예측해 오류가 발생

즉, 파란색 1번 데이터와 직선과의 거리가 여백(Margin)임



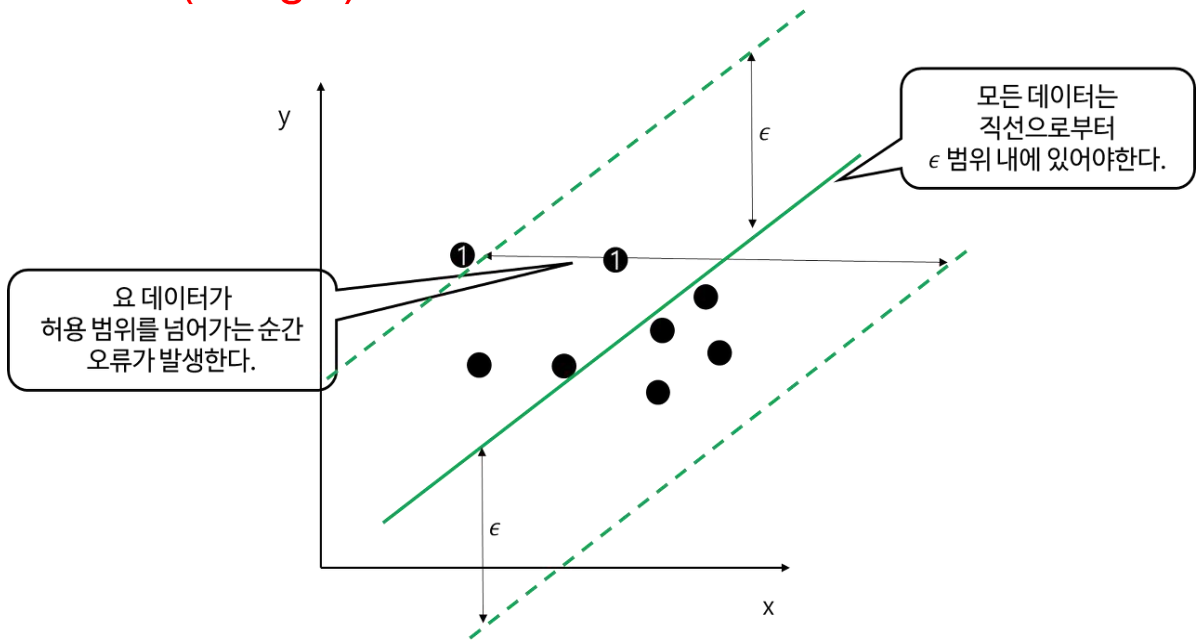
회귀에서의 여백(Margin)

단순 선형 모델(설명 변수 1개)를 생성하고, 데이터들이 초평면으로 부터 $\epsilon > 0$ 범위 내에 있다고 가정

데이터들은 초록색 직선으로 부터 ϵ 범위(양쪽 점선 사이)에 있으며, 데이터들은 x축 상에서 움직임

양 점선 사이의 수평거리를 데이터가 넘어가게 되면 ϵ 범위를 넘어가게 되어 오류를 발생시킴

즉, 양 점선 사이의 수평거리가 여백(Margin)임



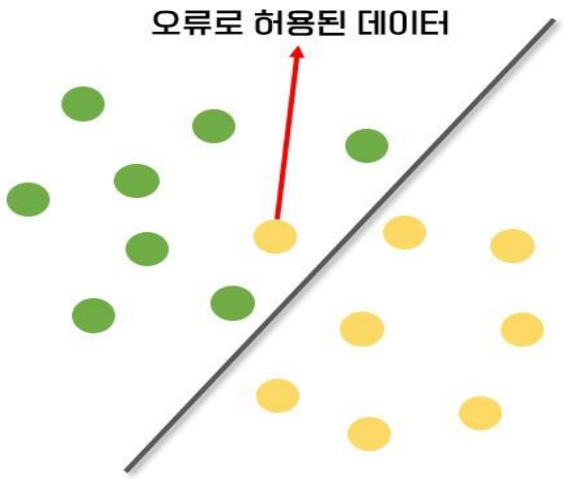
Soft Margin VS Hard Margin

Soft Margin

Soft Margin은 결정 경계를 조금씩 넘어가는 데이터들을 어느 정도 허용하여 유연한 결정 경계를 만듦

아래와 같은 데이터 분포는 직선으로 두 클래스를 분류하기 어렵기 때문에 어느 정도의 비용(Cost, C)을 감수하면서 가장 최선의 결정 경계를 찾음

Cost는 모델링을 하면서 설정이 가능하고, 이 값이 크면 클수록 Soft Margin을, 작으면 작을수록 Hard Margin을 만듦



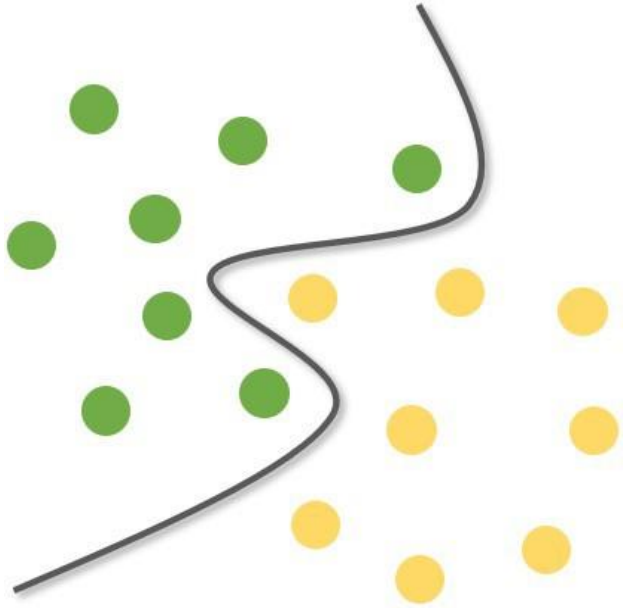
Soft Margin VS Hard Margin

Hard Margin

Hard Margin은 이상치들을 허용하지 않고, 분명하게 나누는 결정 경계를 만듦

과적합(Overfitting)의 오류가 발생하기 쉬움

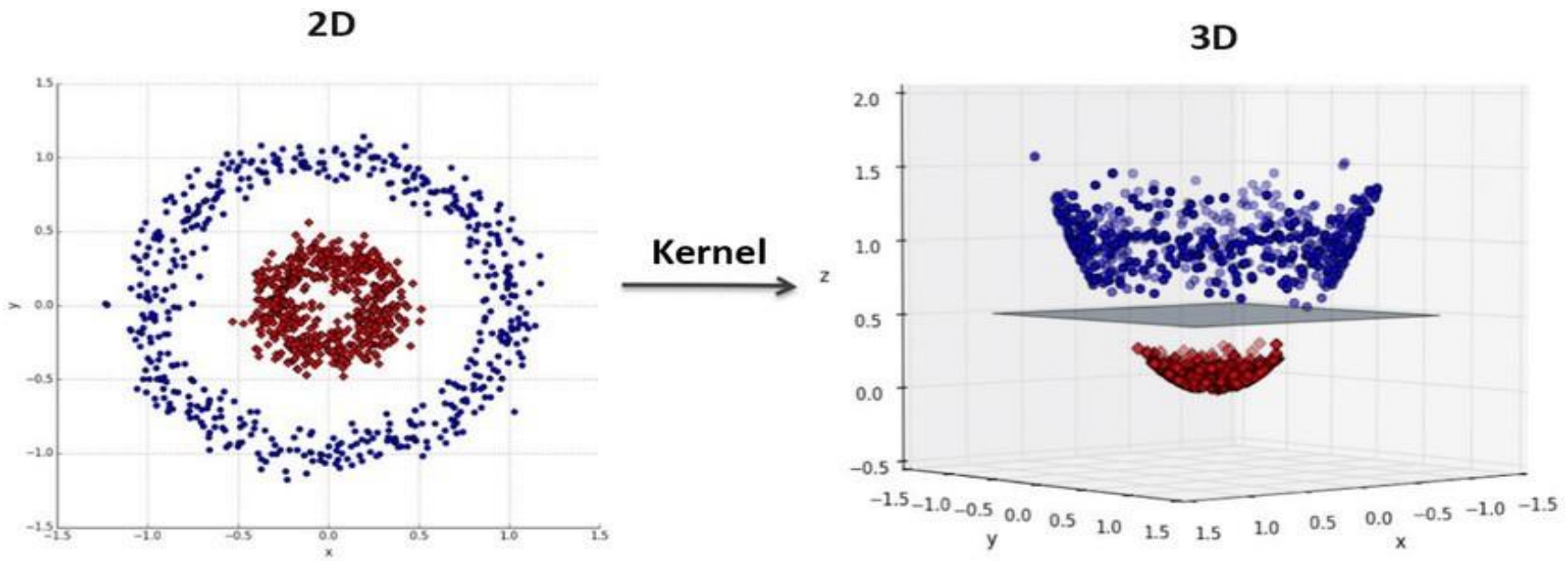
노이즈로 인해 최적의 결정 경계를 잘못 설정하거나 못 찾는 경우가 발생할 수 있음



Kernel Trick

SVM은 기본적으로 선형 분류를 위한 결정 경계를 만들지만, Kernel Trick을 사용한다면 비선형 분류도 가능

아래와 같은 데이터 분포는 저차원(2D)에서는 선형 분리가 되지 않을 수 있지만, 고차원(3D)에서는 선형 분리가 가능



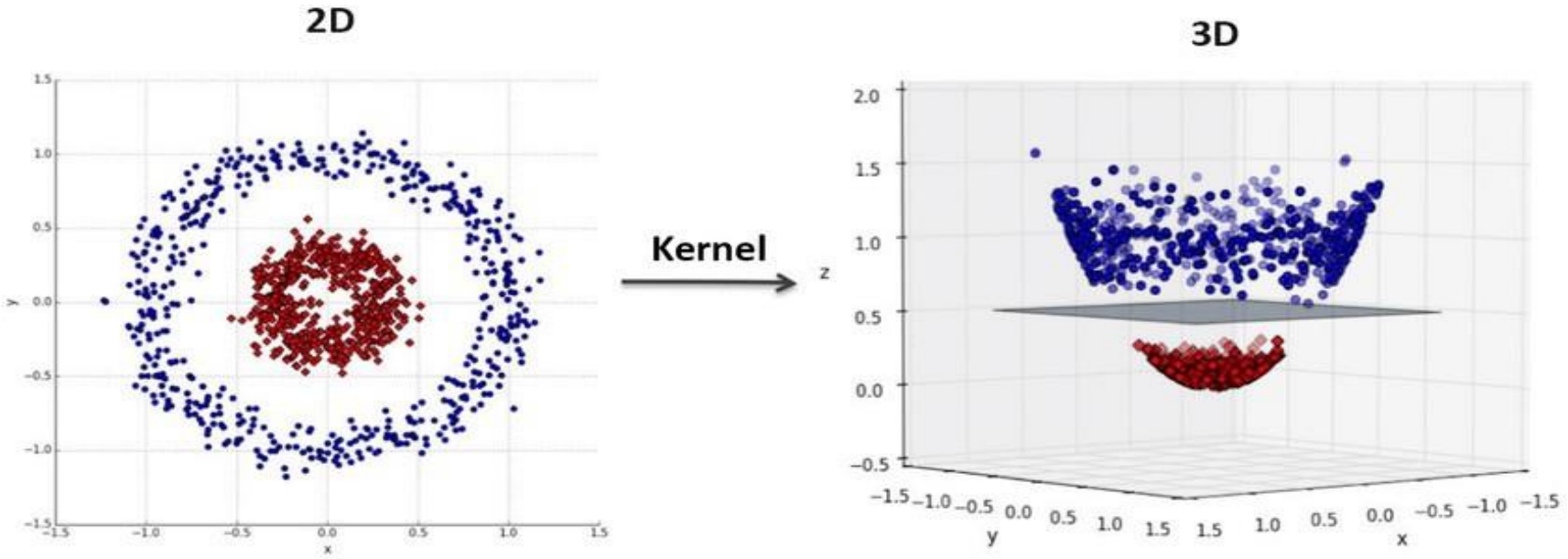
Kernel Trick

데이터를 고차원으로 보내서 서포트 벡터를 구하고 다시 저차원으로 축소하는 과정은 복잡하고

많은 연산량이 필요하기 때문에 Kernel Trick을 사용

Kernel Trick은 선형 분리가 불가능한 저차원 데이터를 함수화하여 고차원으로 보내 선형 분리를 하는 기법

Kernel Trick은 고차원 Mapping과 고차원에서의 내적 연산이 동시에 가능함



대표적인 Kernel 함수

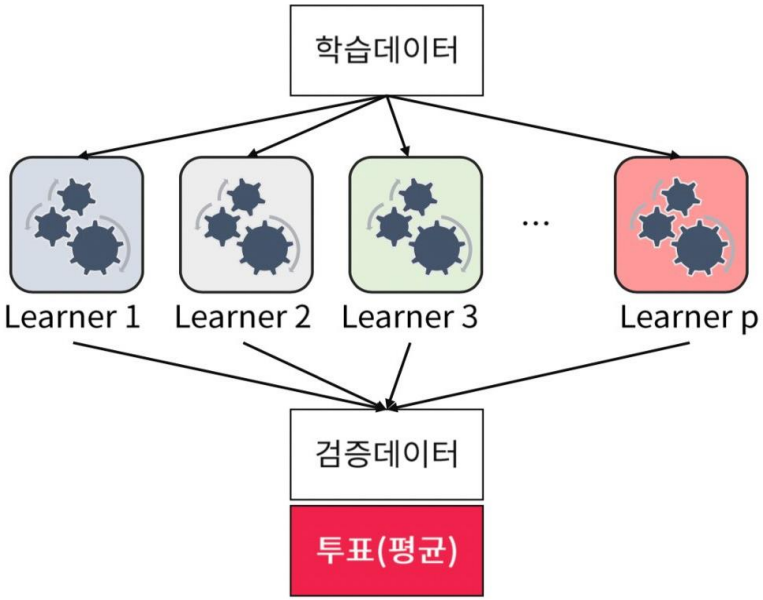
- Linear : 선형 함수
- Poly : 다항식 함수
- RBF : 방사 기저 함수
- Hyperbolic Tangent : 쌍곡선 탄젠트

Tree 기반 모델과 앙상블 기법

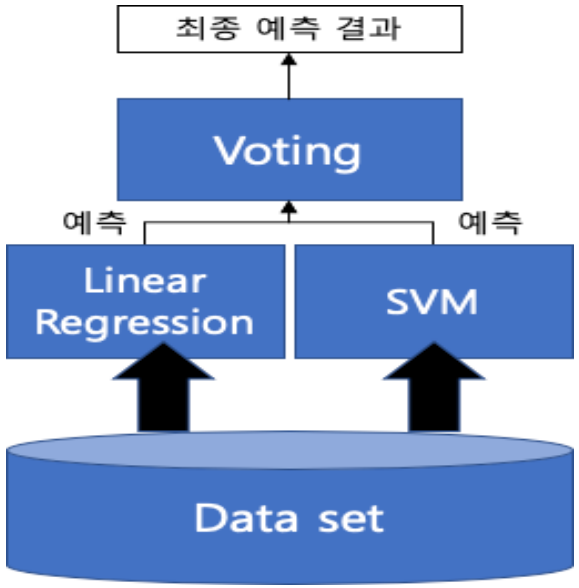
앙상블 학습(Ensemble Learning)

여러 개의 분류기를 생성하고, 그 예측을 결합함으로써 보다 정확한 예측을 도출하는 기법
강력한 하나의 모델을 사용하는대신 보다 약한 모델 여러개를 조합하여 더 정확한 예측에 도움을 주는 방식

보팅(Voting), 배깅(Bagging), 부스팅(Boosting) 세 가지의 유형이 존재



앙상블 학습(Ensemble Learning)



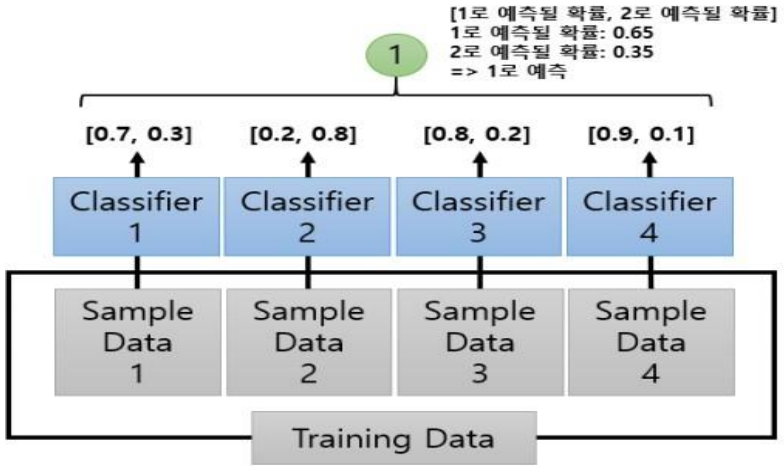
보팅(Voting)

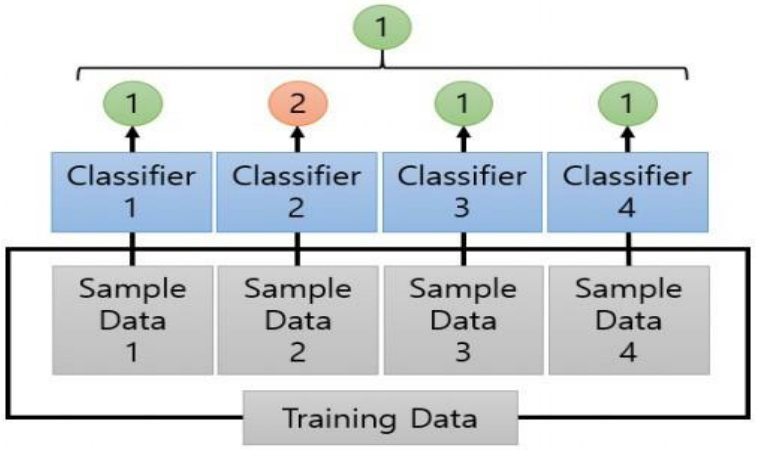
여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정하는 방식

서로 다른 알고리즘을 여러 개 결합하여 사용

보팅 방식

- 하드 보팅(Hard Voting) : 다수의 분류기가 예측한 결과값을 최종 결과로 선정
- 소프트 보팅(Soft Voting) : 모든 분류기가 예측한 레이블 값의 결정 확률 평균을 구한 뒤 가장 확률이 높은 레이블 값을 최종 결과로 선정



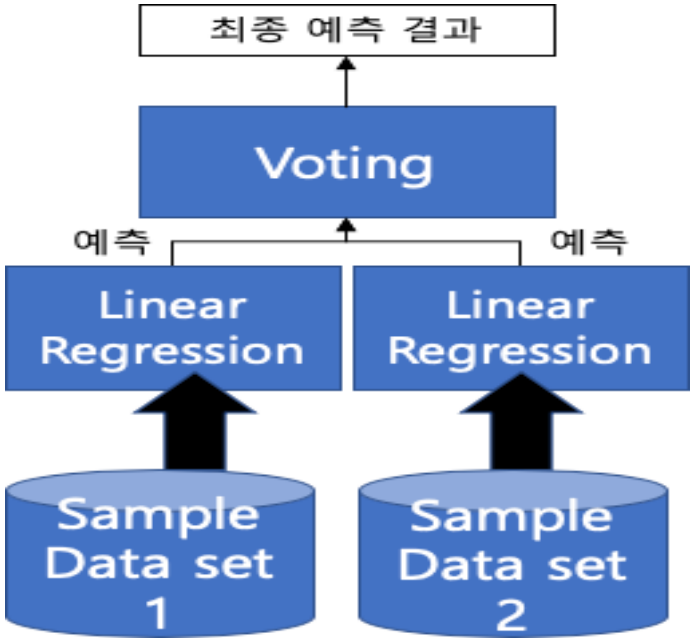


앙상블 학습(Ensemble Learning)

배깅(Bootstrap AGGregatING, Bagging)

- 데이터 샘플링(Bootsrap)을 통해 모델을 학습시키고 결과를 집계(Aggregation)하는 방법으로
  모두 같은 유형의 알고리즘 기반의 분류기를 사용
- 데이터 분할 시 중복을 허용

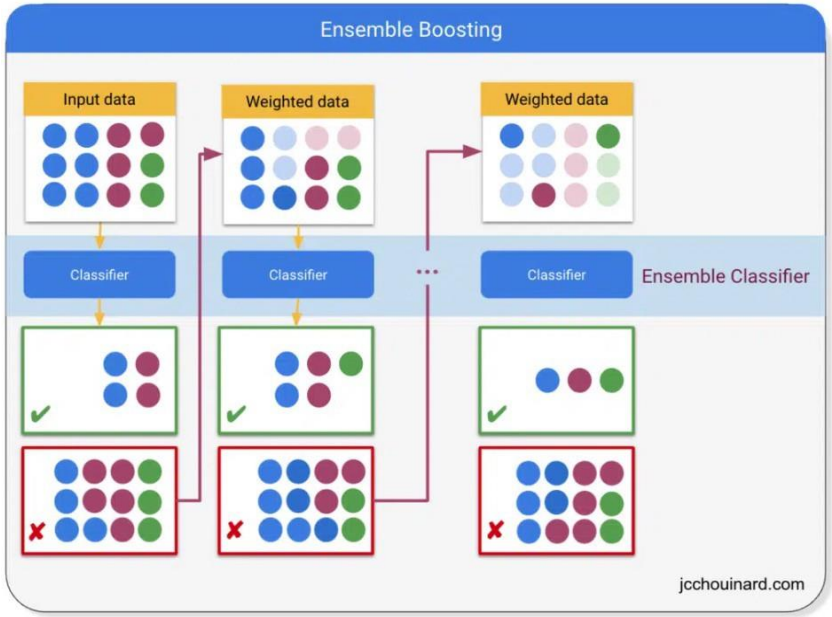
- 범주형 데이터는 다수결 투표 방식으로 결과 집계
- 연속형 데이터는 평균값 집계를 활용
  과적합(Overfitting) 방지에 효과적



앙상블 학습(Ensemble Learning)

부스팅(Boosting)

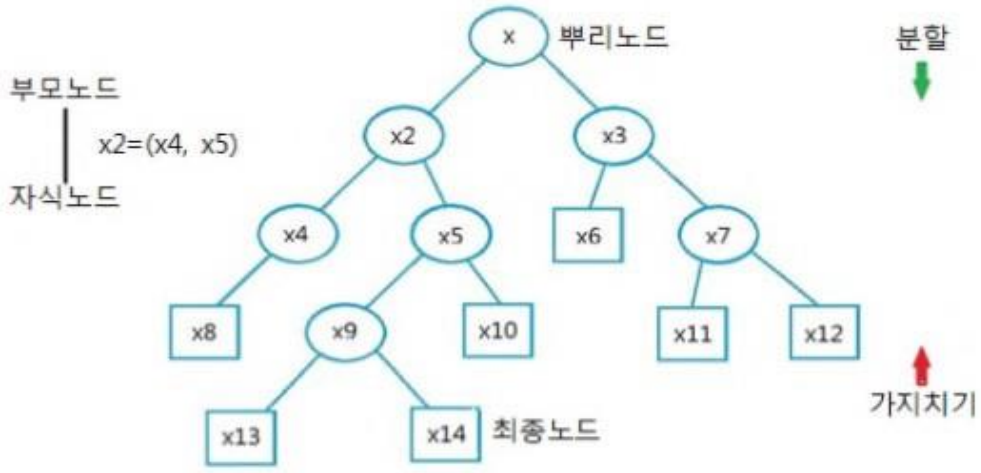
- 여러 개의 분류기가 순차적으로 학습을 수행
- 이전 분류기가 예측이 틀린 데이터에 대해서 올바르게 예측할 수 있도록 다음 분류기에게 가중치(weight)를 부여하면서 학습과 예측을 진행
- 예측 성능이 뛰어나 앙상블 학습을 주도
- 보통 부스팅 방식은 배깅에 비해 성능이 좋지만, 속도가 느리고 과적합이 발생할 가능성이 존재하므로 상황에 따라 적절하게 사용해야 함
- 대표적인 부스팅 모듈 kW XGBoost, AdaBoost, Gradient Boosting

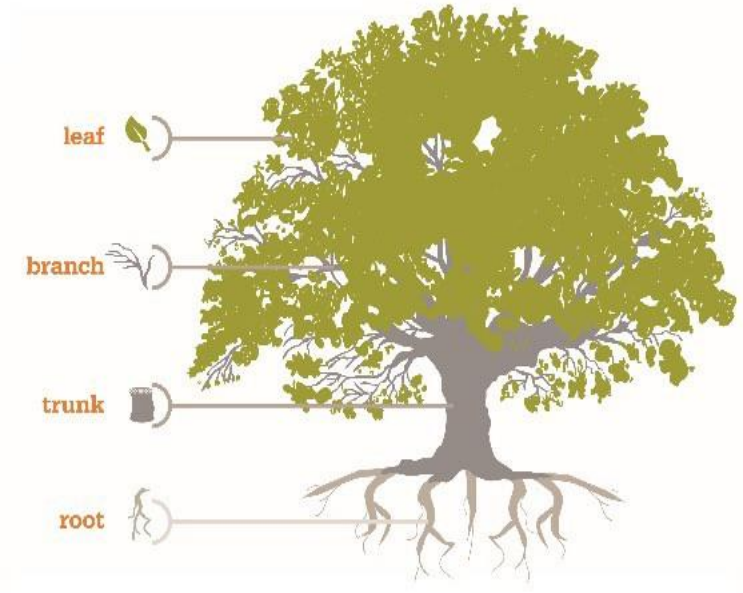


의사결정 트리(Decision Tree)

의사결정 트리(Decision Tree)는 분류, 회귀 문제에 모두 사용할 수 있는 모델

의사결정 트리는 입력 변수를 특정한 기준으로 분기해 트리 형태의 구조로 분류하는 모델





의사결정 트리(Decision Tree) 장단점

장점

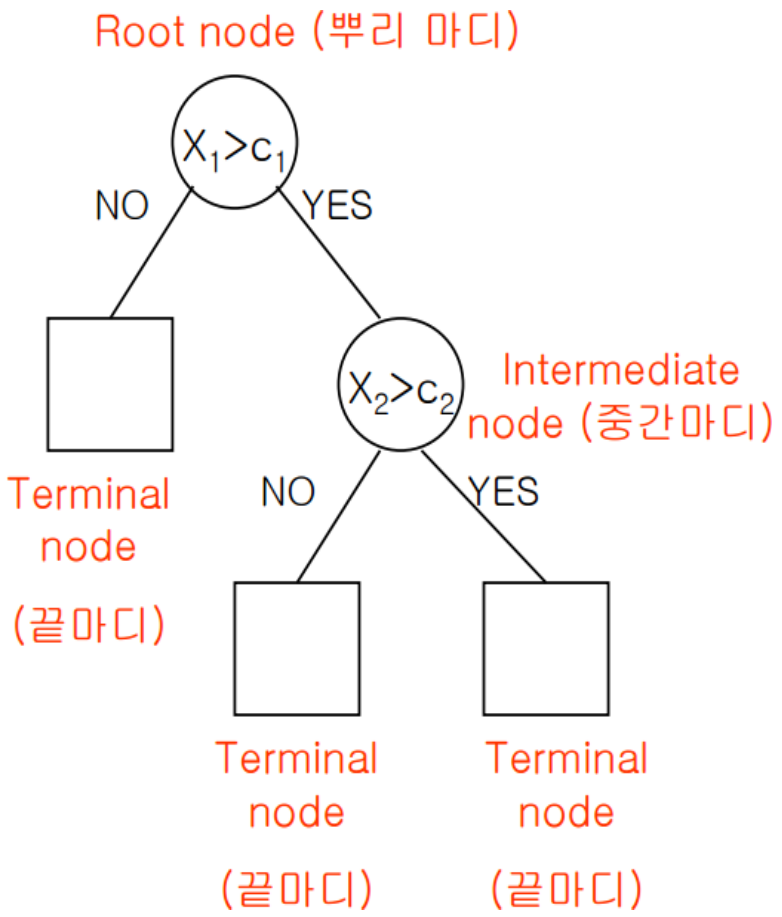
- 해석이 쉬움
- 입력값이 주어졌을 때 설명 변수의 영역의 흐름을 따라 출력값이 어떻게 나오는지 파악하기 용이

단점

- 예측력이 떨어짐
- 단순히 평균 또는 다수결 법칙에 의해 예측을 수행
- 회귀모델에서 반응 변수의 평균을 예측값으로 추정할 때 평균을 사용해 이상치에 영향을 많이 받음

의사결정 트리 구성

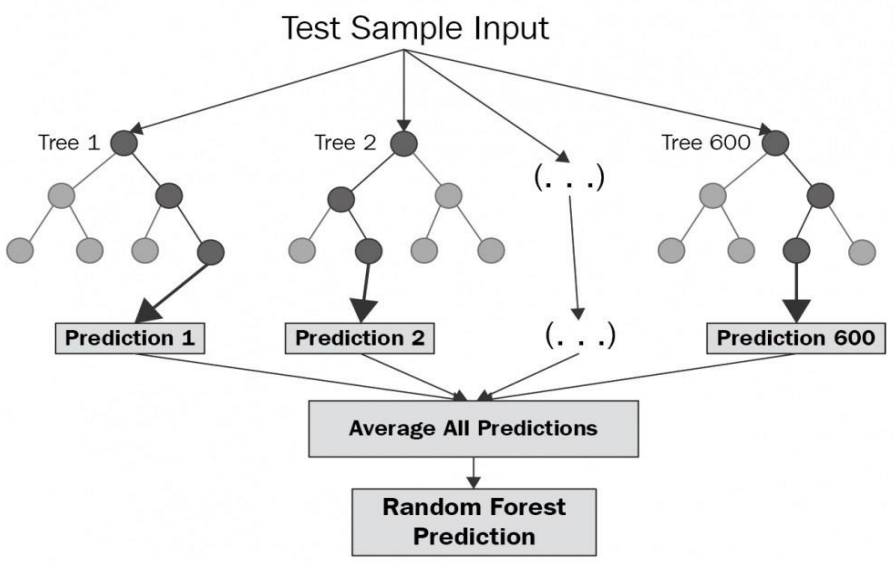
- 뿌리마디(root node) : 시작되는 마디로 전체 자료를 포함
- 자식마디(child node) : 하나의 마디로부터 분리되어 나간 2개 이상의 마디들
- 부모마디(parent node) : 주어진 마디의 상위마디
- 끝마디(terminal node) : 자식마디가 없는 마디
- 중간마디(internal node) : 부모마디와 자식마디가 모두 있는 마디
- 가지(branch) : 뿌리마디로부터 끝마디까지 연결된 마디들
- 깊이(depth) : 뿌리마디부터 끝마디까지의 중간마디의 수

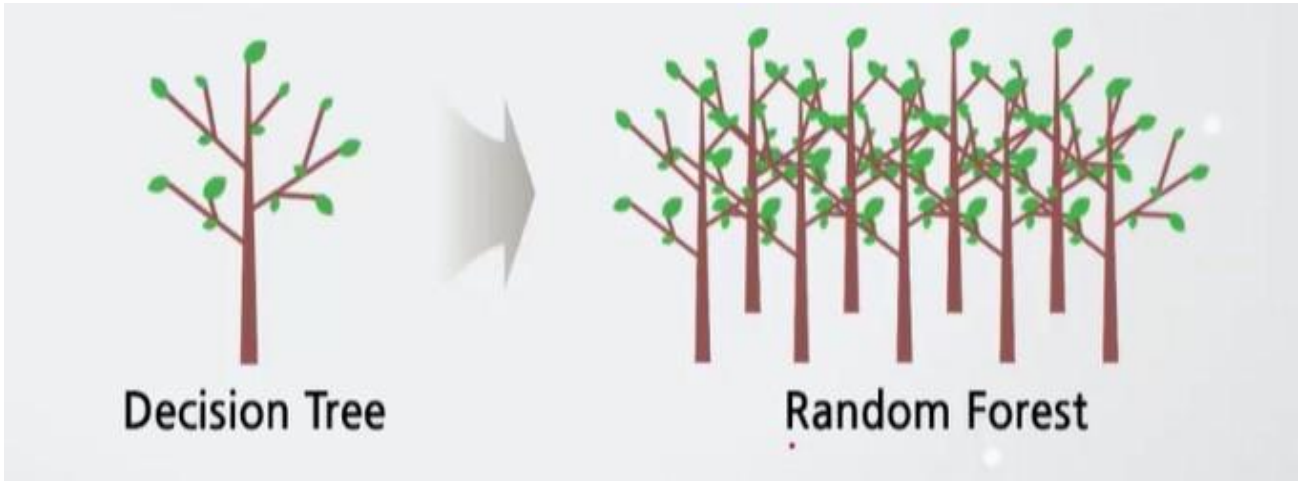


랜덤 포레스트(Random Forest)

여러 개의 결정 트리들을 임의적으로 학습하는 방식의 앙상블 방법

여러가지 학습기들을 생성한 후 이를 선형 결합하여 최종 학습기를 만드는 방법





랜덤 포레스트(Random Forest)

기반 기술

의사결정 트리 : 여러가지 요소를 기준으로 갈라지는 가지를 트리형태로 구성하여 분석하는 기법

앙상블 학습 : 주어진 데이터를 여러 모델로 학습하고 종합하여 정확도를 높이는 기법

배깅(Bagging) : 같은 알고리즘으로 여러 개의 분류기를 만들어서 결합하는 앙상블 학습 기법

의사결정 트리의 한계

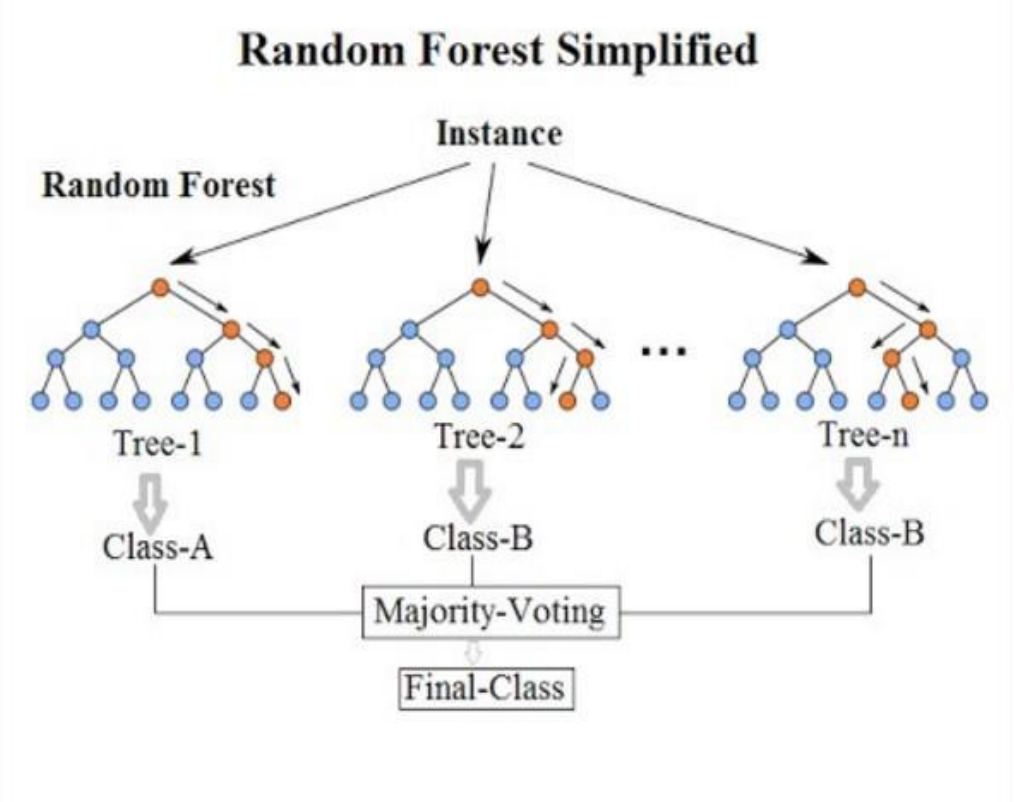
학습 데이터에 따라 생성되는 결정 트리가 크게 달라져 일반화가 어려운 과적합 문제 발생

계층적 접근방식으로, 중간에 에러 발생 시 하위 계층으로 에러가 누적

랜덤 포레스트 사용!!!!!

랜덤 포레스트 특징

- 임의성 : 서로 조금씩 다른 특성의 트리들로 구성
- 비상관화 : 각 트리들의 예측이 서로 연관되지 않음
- 견고성 : 오류가 전파되지 않아 노이즈에 강함
- 일반화 : 임의화를 통한 과적합 문제 극복



랜덤 포레스트 장단점

장점

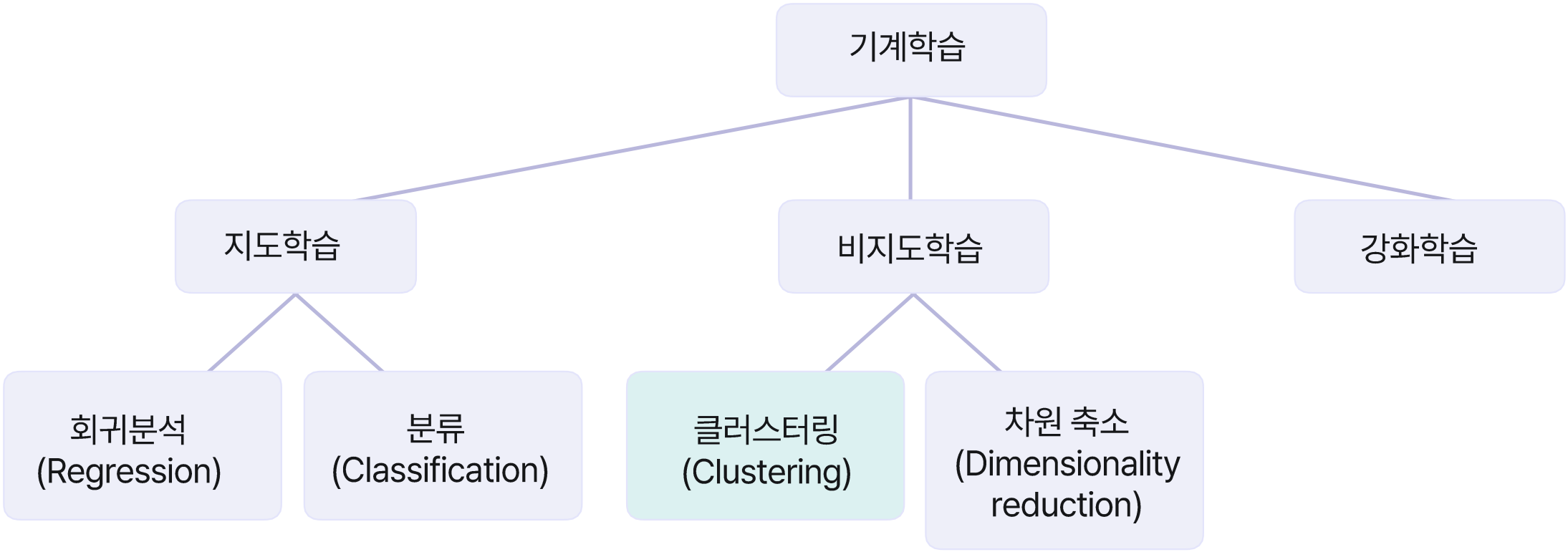
- 과적합이 잘 일어나지 않음
- 결측치나 이상치에 강함
- 의사결정나무 알고리즘에 기반한 기법이기 때문에 scaling, 정규화 과정이 필요 없음
- 비선형적 데이터에 강함
- 새로운 데이터가 들어와도 크게 영향을 받지 않음

단점

- 수 많은 트리를 계산하기 때문에 학습 시간과 계산 연산량이 큼

비지도학습과 군집화

기계 학습


기계학습
지도학습
비지도학습
강화학습
회귀분석
(Regression)
분류
(Classification)
클러스터링
(Clustering)
차원 축소
(Dimensionality
reduction)

지도학습의 한계

- 지도 학습을 통해 객체를 분류하는 것이 가능.

- 하지만 이는 **많은 양**의 **질 좋은 학습 데이터**를 필요로 한다.
- 데이터가 충분하지 않은 경우에 직접 커스텀 데이터를 만들어야 함.

비지도학습

- 정답(Label)이 지정되지 않은 데이터 세트를 분석하고 군집화

- 학습 데이터가 필요하지 않음.

  – 색깔을 기준으로 군집화 하면 2개의 군집이 발생.

  – 모양을 기준으로 군집화 하면 4개의 군집이 발생.

어떤 것이 좋다는 정답은 없다.

군집화와 MBTI

질문을 통해 인간의 성격을 16가지로 군집화

1. 각 군집마다 제목을 짓고, 특징 설명

2. 개개인의 MBTI를 의사소통에 활용

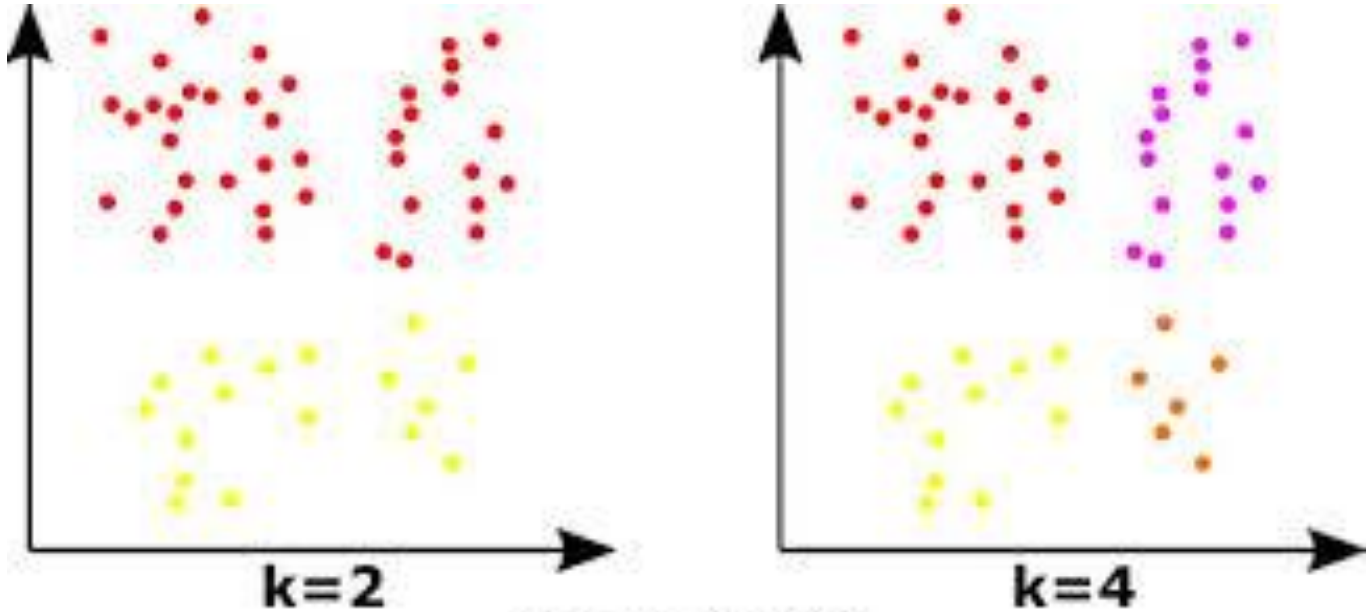
군집 분류 및 활용

K-means

K-means

- **평균**(means)를 활용해, **K개의 군집**을 만드는 알고리즘

- 평균은 군집의 중심좌표와 군집 안의 데이터들 사이의 평균 거리를 의미



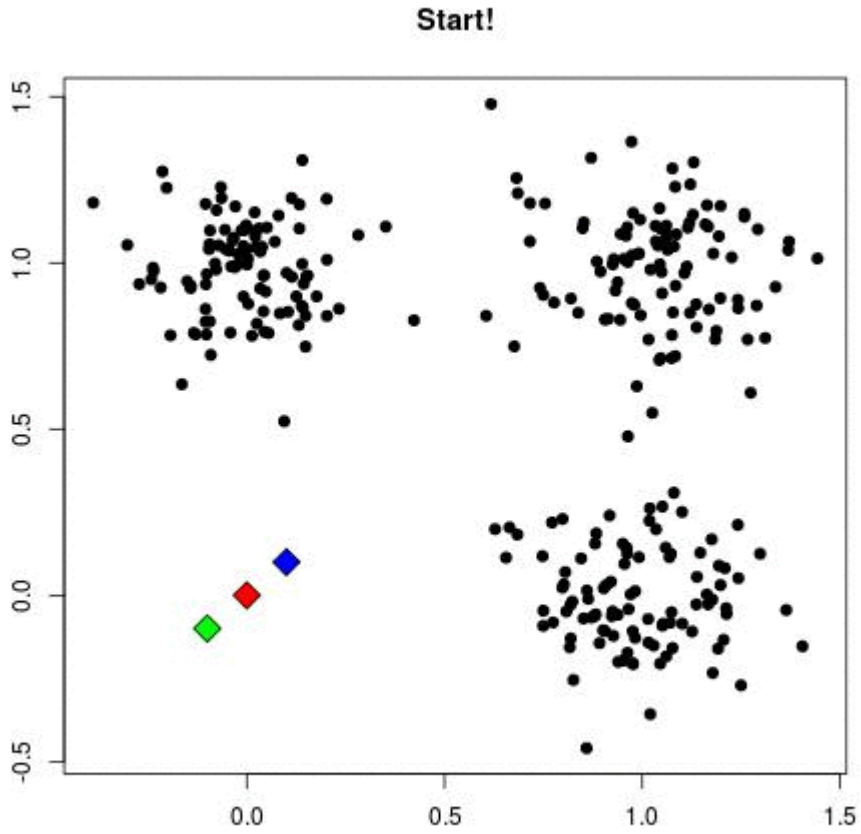
K-means

1. 군집의 개수와 초기 중심점을 선택한다.

2. 데이터를 거리 상 가장 가까운 주심점이 속한 군집으로 배정한다

3. 중심점을 해당 군집의 데이터의 평균 위치로 재설정한다.
   - 중심점이 변하지 않으면 종료한다 .

4. 2, 3과정을 반복한다.



출처 : https://github.com/bheemnitd/2D-KMeans-C

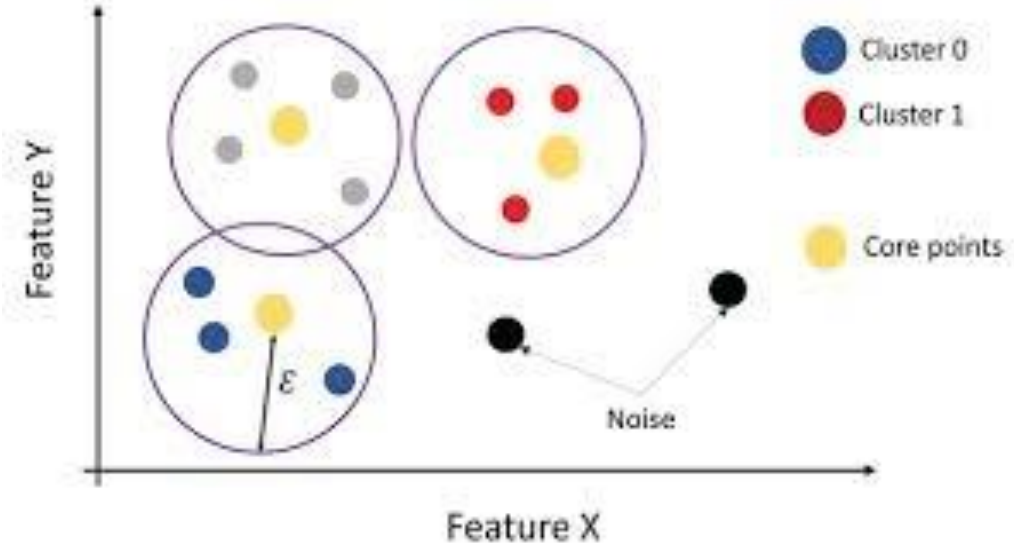
K-means 특징

- 알고리즘이 쉽고 간결, 가장 많이 활용됨
- 모든 요소가 군집에 속하게 됨
- K값을 사용자가 직접 지정(군집 개수 설정의 어려움)
- 속성의 개수가 매우 많을 경우 군집화 정확도가 감소

DBSCAN

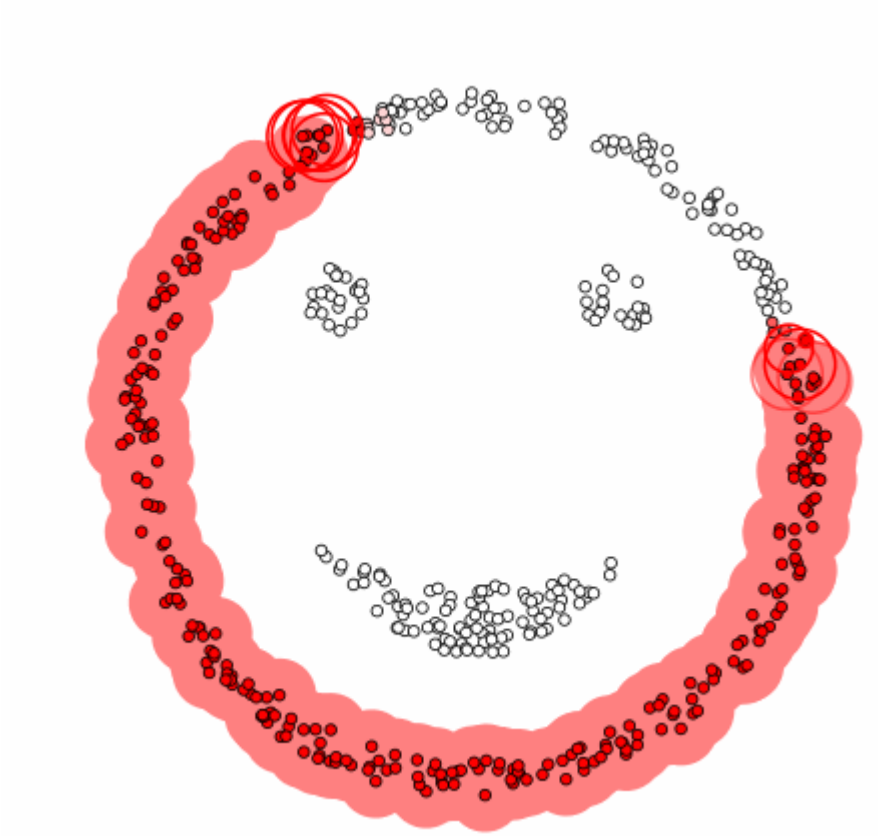
DBSCAN(Density-Based Spatial Clustering of Applications with Noise)

- 단위 공간 안에 모여 있는 데이터의 **밀도(Danse)**를 기반으로 군집화
- 중심점의 일정 **거리(Epsilon)** 안에 **샘플 개수(Min Points)**가 일정 수 이상 들어 있을 경우 군집화.



DBSCAN

1. 데이터 중, 임의의 포인트를 선택한다.
2. 선택한 데이터와 Epsilon 거리 내에 있는 모든 데이터를 찾는다.
3. 데이터 수가 Min Points 이상이면, 해당 포인트를 중심으로 하는 군집을 생성한다.
4. 군집안에 존재하는 다른 점 중, 다른 군집의 중심이 되는 데이터 포인트가 존재한다면 두 군집을 합 친다.
5. 1~4번을 모든 포인트에 대해서 반복한다.
6. 어느 Cluster에도 포함되지 않는 데이터 포인트는 이상치로 처리한다.



출처 : https://www.linkedin.com/pulse/role-dbscan-machine-learning-deepak-kur

DBSCAN 특징

- 복잡한 기하학적 분포도를 가진 데이터 세트에 대한 성능 우수
- 군집에 속하지 않은 요소(Noise Points) 발생
- 적절한 Epsilon, Min Point 지정의 어려움

머신러닝 알고리즘 실습