

▼ 머신러닝 이론

▼ 인공지능 VS 머신러닝 VS 딥러닝

인공지능(AI, Artificial Intelligence)

정의: 인간의 지능적 행동(추론, 학습, 문제 해결, 의사결정 등)을 기계가 모방하도록 하는 기술·연구 분야

범위: 가장 큰 개념으로, 규칙 기반 시스템부터 최신 신경망 기반 기법까지 모두 포함

머신러닝(ML, Machine Learning)

정의: AI의 하위 분야로, 데이터를 통해 스스로 패턴을 학습하고 예측·판단하는 알고리즘 연구

특징:

명시적인 규칙 작성 대신 데이터 학습으로 모델 생성

지도학습, 비지도학습, 강화학습 등 다양한 방법 존재

예시: 스팸 메일 분류, 고객 이탈 예측, 추천 시스템

딥러닝(DL, Deep Learning)

정의: 머신러닝의 하위 분야로, **인공신경망(ANN, Artificial Neural Network)**을 깊고 복잡하게 쌓아 학습하는 기법

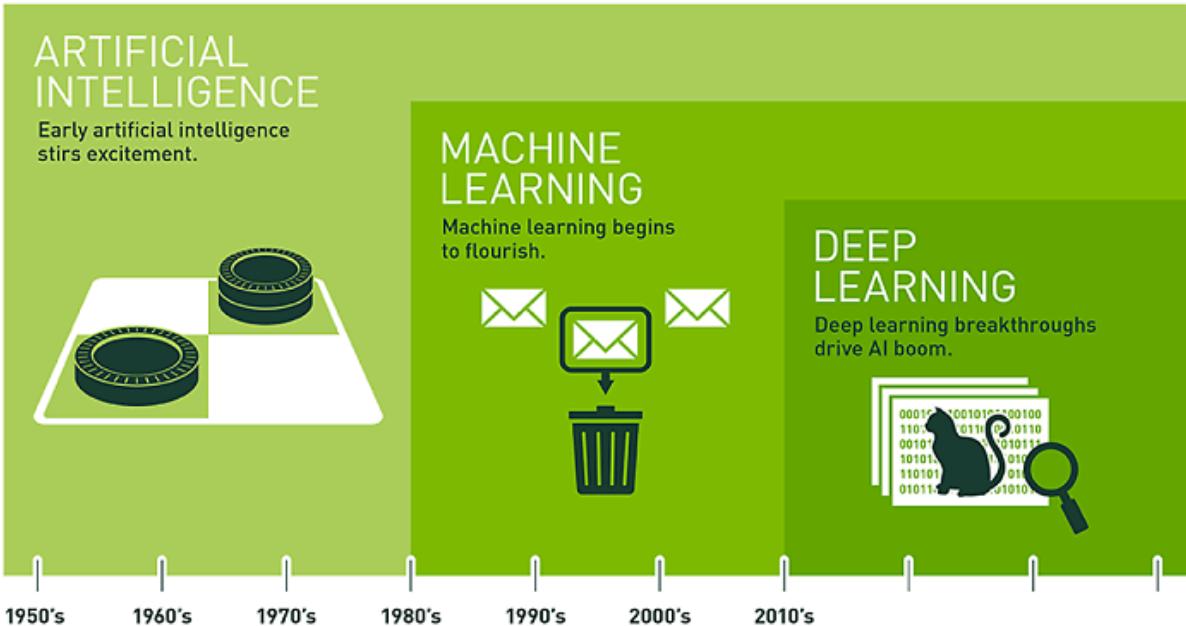
특징:

대규모 데이터와 고성능 컴퓨팅 환경에서 강력한 성능

자동으로 특징(feature)을 추출하여 고차원 문제 해결 가능

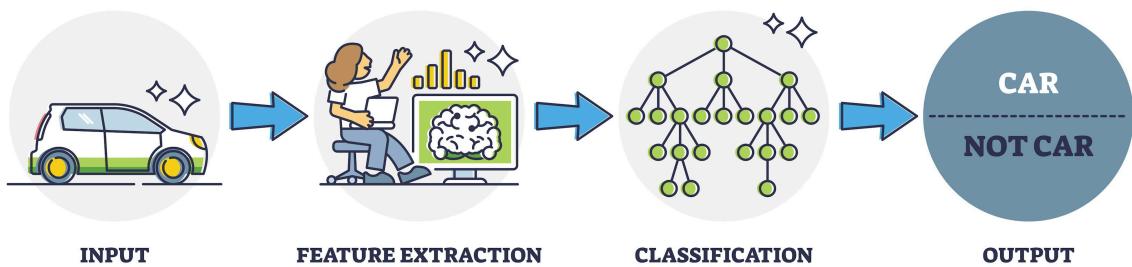
CNN, RNN, Transformer 등 다양한 구조 활용

예시: 이미지 인식, 자연어 처리, 자율주행 차량

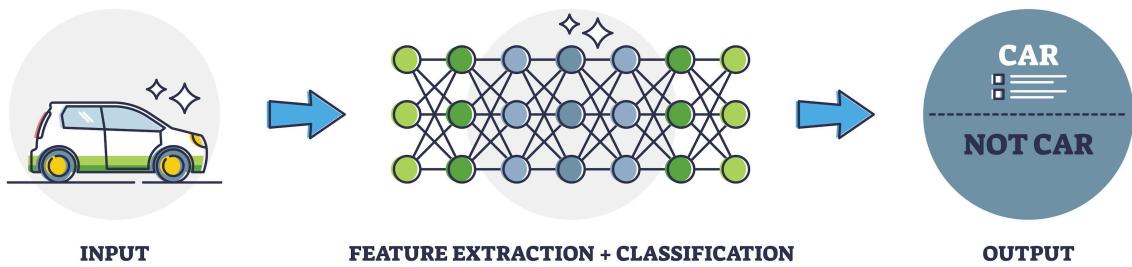


Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

MACHINE LEARNING



DEEP LEARNING



지도학습(Supervised Learning)

- 입력 X 와 라벨(정답) y 이 모두 주어진 데이터로 학습
- 새로운 데이터 입력 $X \rightarrow$ 정답 y 를 예측하여 예측 오차(손실) 를 최소화하는 것을 목표로 함

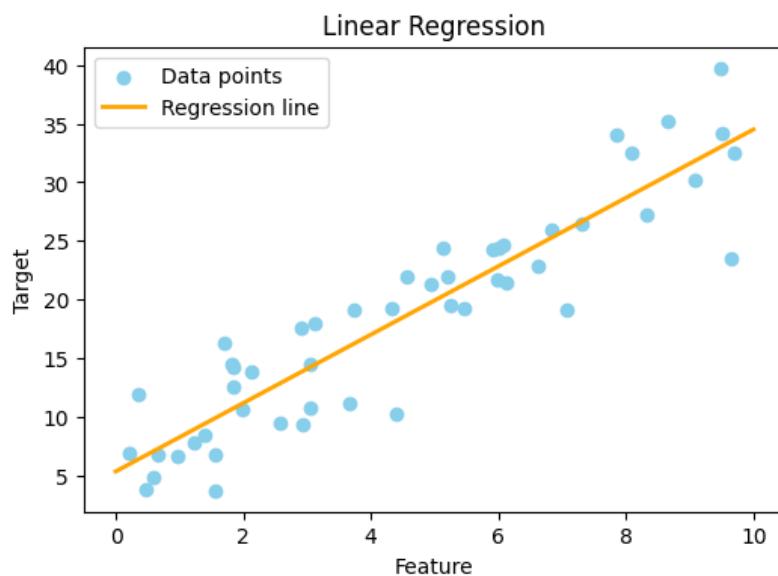
비지도학습(Unsupervised Learning)

- 라벨이 없는 데이터 X 만으로 학습하여 데이터의 구조나 패턴을 파악
- 내재된 규칙성을 바탕으로 유사한 샘플을 군집화하고, 분포를 추정하거나 차원을 축소하는 등의 작업에 활용

선형 회귀 (Linear Regression)

선형 회귀란?

- 연속형 숫자 값(수치형 데이터)을 예측하기 위해 사용하는 지도 학습(Supervised Learning) 기법
- 독립 변수(입력 변수, X) 와 종속 변수(출력 변수, y) 사이의 선형 관계(직선적 관계)를 학습
- 주어진 데이터를 통해 최적의 직선을 찾아 예측을 수행



기본 개념

- 단순 선형 회귀 (Simple Linear Regression): 독립 변수가 1개일 때
- 다중 선형 회귀 (Multiple Linear Regression): 독립 변수가 2개 이상일 때
- 모델식: $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$
 \hat{y} : 종속 변수의 예측값
 x_i : 독립 변수
 β_i : 회귀 계수 (기울기, 가중치)
 β_0 : 절편 (intercept)

목표

- 회귀 계수 β 를 추정하여 실제 값과 예측 값의 차이를 최소화 하는 것
- 일반적으로 최소제곱법 (Ordinary Least Squares, OLS)을 사용
- 목적 함수: $\min_{\beta} \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \min_{\beta} \frac{1}{n} \sum_{i=1}^n \epsilon^2$
 y_i : 실제값 (측정을 통해 얻은 값)

\hat{y}_i : 예측값 (모델을 통해 예측된 값)

ϵ : 예측 오차

성능 평가 지표 - 회귀

- **MSE (Mean Squared Error)** : 실제값과 예측값의 오차를 제곱하여 평균낸 값

$$MSE = \frac{1}{n} \sum (y_i - \hat{y}_i)^2$$

- **RMSE (Root Mean Squared Error)** : MSE의 제곱근, 평균적인 오차 크기

$$RMSE = \sqrt{MSE}$$

- **MAE (Mean Absolute Error)** : 절대오차의 평균, RMSE보다 이상치에 덜 민감

$$MAE = \frac{1}{n} \sum |y_i - \hat{y}_i|$$

- **R² (결정계수)** : 종속변수 분산 중 모델이 설명한 비율, 모델의 설명력 $R^2 = 1 - \frac{\sum(y_i - \bar{y})^2}{\sum(y_i - \hat{y}_i)^2} \rightarrow 1$ 에 가까울수록 설명력이 높음

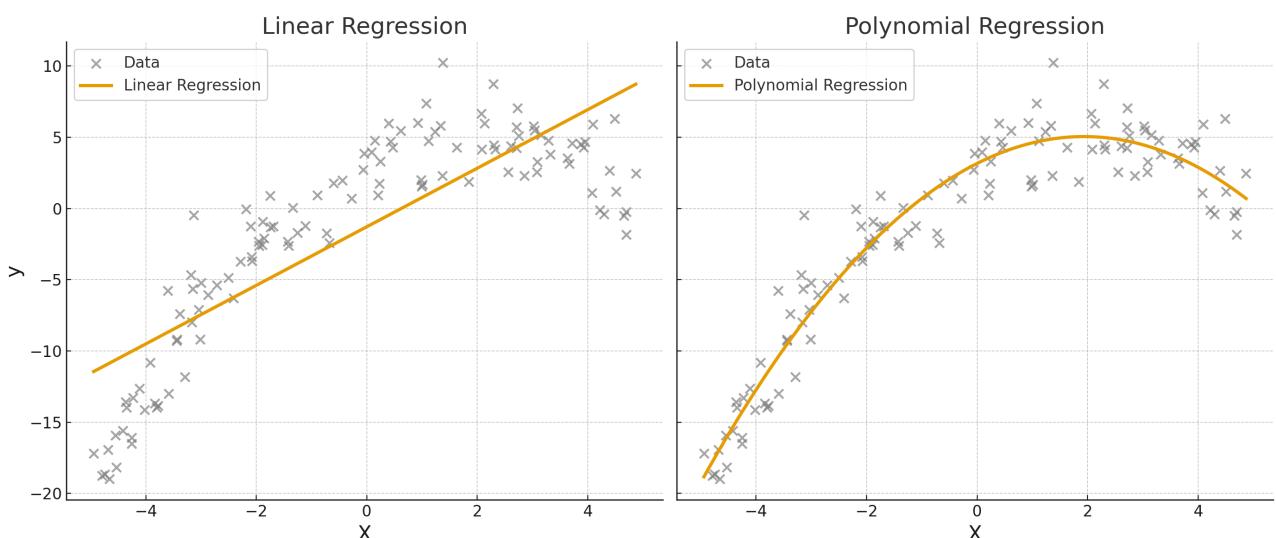
활용 예시

- 집값 예측 (면적, 방 개수 → 가격)
- 학생 성적 예측 (공부 시간, 출석률 → 점수)
- 매출 예측 (광고비, 프로모션 → 판매량)

다항식 회귀 (Polynomial Regression)

개념

- 회귀 분석(Regression Analysis)의 한 기법
- 입력 변수 x와 출력 변수 y의 관계를 **다항식(polynomial)** 형태로 모델링
- 직선이 아닌 곡선 형태의 데이터 패턴을 설명할 수 있음



모델 수식

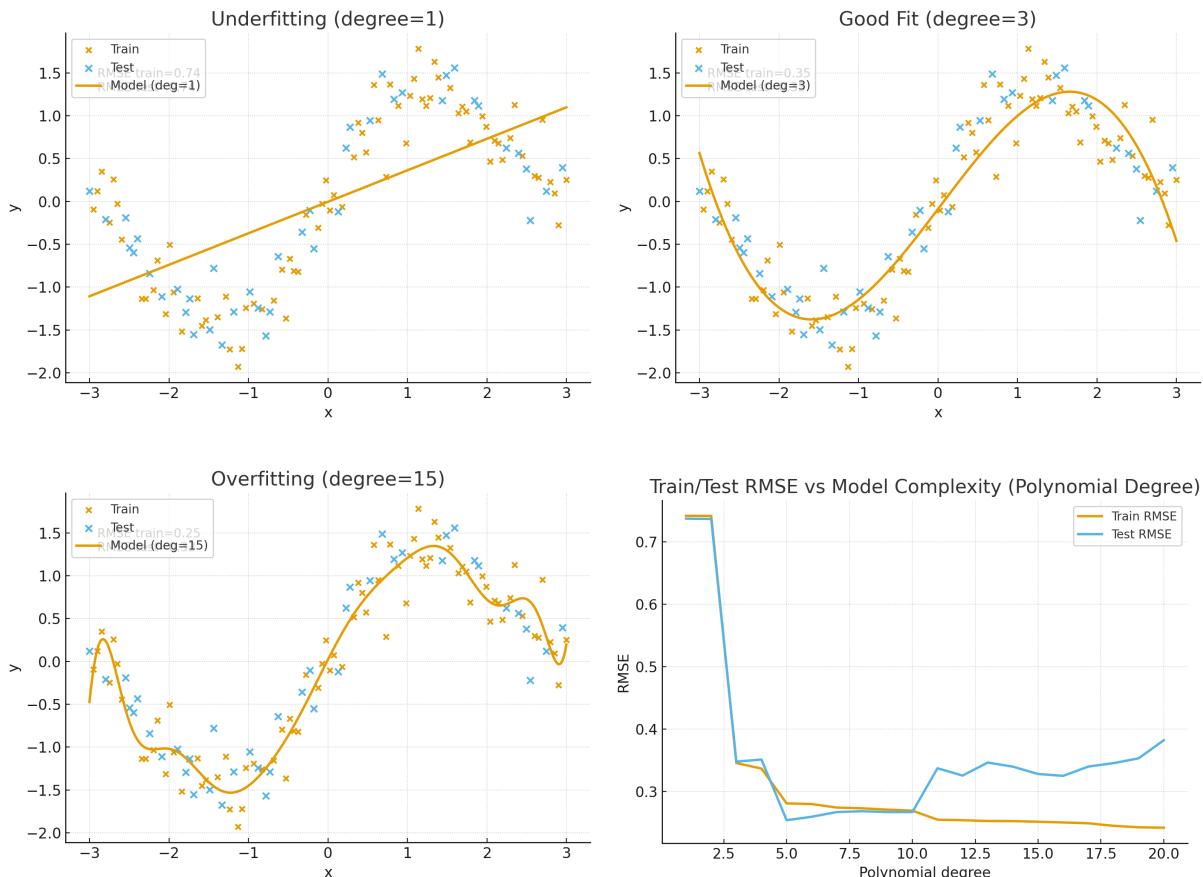
다항식 회귀는 입력 변수 x 의 제곱, 세제곱 등의 고차항을 포함한 선형 결합으로 표현

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_d x^d$$

- d : 다항식의 차수 (degree)
- β_i : 회귀 계수 (모수)

적절한 차수 선택

- 교차 검증(Cross Validation)으로 최적의 차수 선택
- 너무 낮은 차수 → 과소적합(Underfitting)
- 너무 높은 차수 → 과적합(Overfitting)



규제 회귀(Lasso, Ridge, ElasticNet)

규제(Regularization)는 계수 크기에 패널티를 부여 해 모델 복잡도를 낮추고 일반화(Generalization) 성능 향상

Ridge (릿지, L2 회귀)

- 목적 함수 : $\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^p \beta_j^2$
- 계수를 0으로 만들진 않지만 **작게 수축**
- 변수는 **대체로 유지**

- 다중공선성(독립 변수들 사이에 강한 상관관계가 있는 상태)으로 인해 회귀 계수가 불안정해지는 문제 해결 (안정화)

Lasso (라쏘, L1 회귀)

- 목적 함수 : $\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^p |\beta_j|$
- 일부 계수를 정확히 0으로 만들어 **변수 선택** 효과

ElasticNet (L1 + L2 혼합)

- 목적 함수 : $J \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \left[\alpha \sum_{j=1}^p |\beta_j| + (1 - \alpha) \sum_{j=1}^p \beta_j^2 \right]$
 - Lasso의 변수 선택 + Ridge의 안정성 결합
 - λ 가 클수록 규제가 강해져 계수의 절대값 감소
-

분류(Classification)

- 데이터를 미리 정해진 범주(클래스, **category**)로 나누는 지도학습(Supervised Learning) 방법
- 입력 변수(독립 변수, X)를 통해 출력 변수(종속 변수, y)의 범주형 값을 예측
- 활용 예시: 스팸 필터링, 이미지 분류 등

로지스틱 회귀 (Logistic Regression)

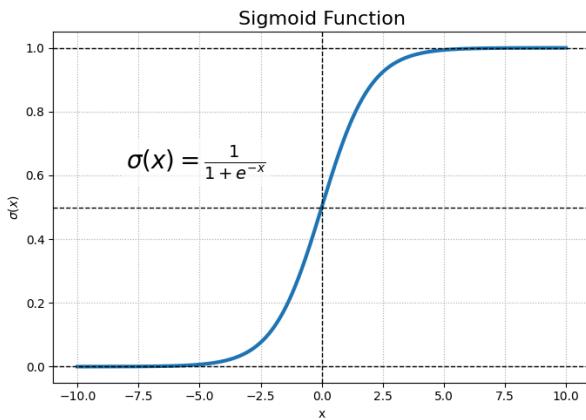
로지스틱 회귀란?

- 이름은 "회귀(Regression)"이지만, 실제로는 **분류(Classification)** 알고리즘
- 종속 변수가 **이진(Binary: 0/1)** 또는 **다중(Multinomial)** 일 때 사용
- 선형 회귀처럼 독립 변수들의 선형 결합을 이용하지만, 결과를 **확률(0~1)**로 변환

기본 개념

- 선형 회귀식: $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$
- 로지스틱 회귀는 이 z 값을 **시그모이드(Sigmoid)** 함수에 통과시켜 **확률**을 출력

$$P(y = 1|x) = \sigma(z) = \frac{1}{1+e^{-z}}$$
- 결과:
 - $P(y = 1|x) \geq 0.5 \rightarrow$ 클래스 1
 - $P(y = 1|x) < 0.5 \rightarrow$ 클래스 0



- 시그모이드(Sigmoid) 함수

- 출력 범위: $(0, 1)$
- 곡선 형태: S자 모양
- 해석: 입력 값이 작으면 0에 가까워지고, 크면 1에 가까워짐

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

성능 평가 지표 - 분류

- 예측 결과를 혼동 행렬 (Confusion Matrix)로 정리

Confusion Matrix

		Real	
		실제 Positive (1)	실제 Negative (0)
Predict	예측 Positive (1)	True Positive (TP)	False Positive (FP)
	예측 Negative (0)	False Negative (FN)	True Negative (TN)

- 정확도 (Accuracy) : 전체 예측 중에서 맞게 예측한 비율

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

- 정밀도 (Precision) : 모델이 Positive라고 예측한 것 중에서 실제로 Positive인 비율

$$Precision = \frac{TP}{TP+FP}$$

- 재현율 (Recall) : 실제 Positive 중에서 모델이 올바르게 Positive라고 예측한 비율

$$Recall = \frac{TP}{TP+FN}$$

- F1 Score : 정밀도(Precision)와 재현율(Recall)의 조화평균(Harmonic Mean), 두 지표의 균형을 보기 위한 지표

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

활용 예시

- 스팸 메일 분류 (스팸=1, 정상=0)
- 신용카드 사기 탐지 (사기=1, 정상=0)

- 환자 진단 (질병 있음=1, 없음=0)
-

K-최근접 이웃 (K-Nearest Neighbors, KNN)

KNN이란?

- 가장 단순하면서도 직관적인 **지도 학습(Supervised Learning)** 알고리즘
- 새로운 데이터가 들어왔을 때, 기존 데이터 중 **가장 가까운 K개의 이웃**을 찾아 **다수결(분류)** 또는 **평균(회귀)**으로 결과를 예측
- 모델 학습 과정에서 **명시적인 수학적 추정(파라미터 학습)**이 없는 **비모수적(Non-parametric)** 방법

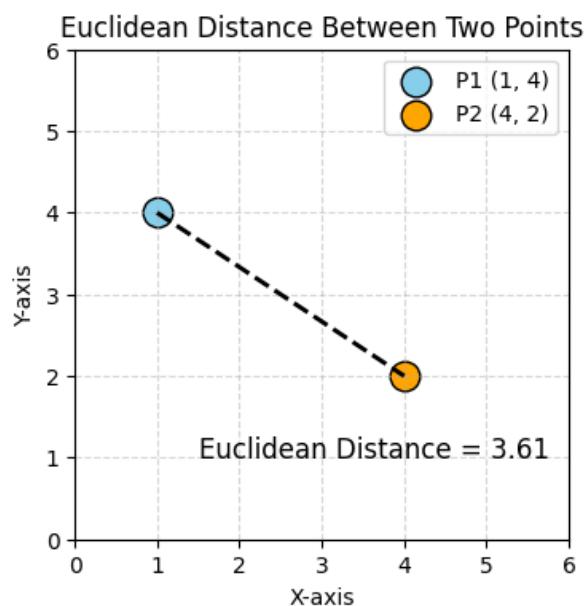
동작 원리

1. 새로운 데이터 포인트가 주어짐
2. 기존 데이터들과의 거리를 계산 (유클리드 거리, 맨해튼 거리 등)
3. 가장 가까운 K개의 이웃을 선택
4. 선택된 이웃들의 **다수결** (분류) 또는 **평균** (회귀)으로 결과 결정

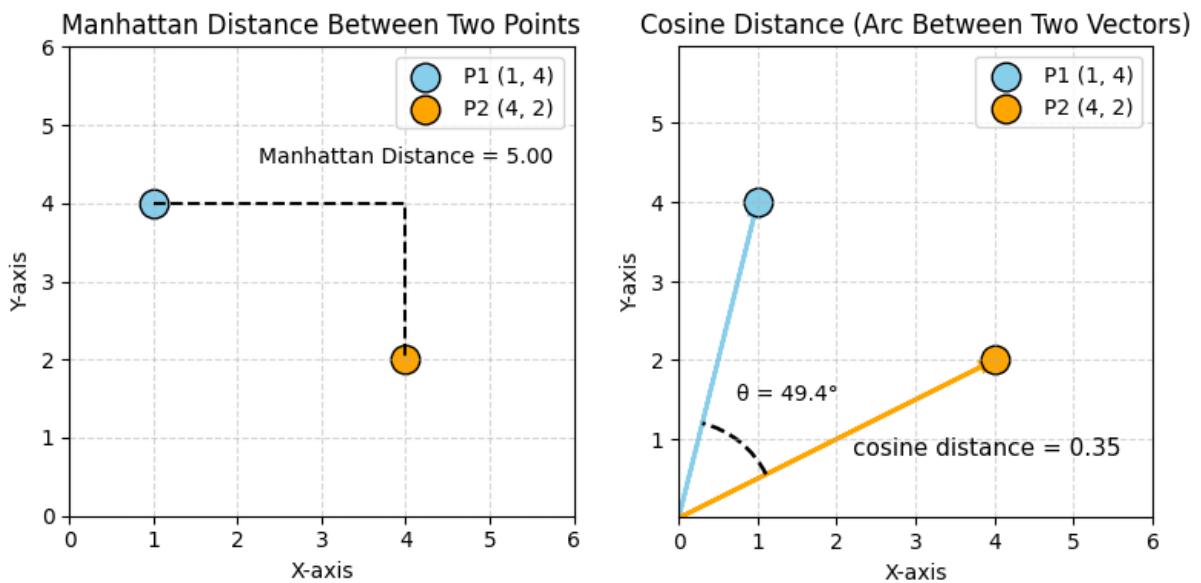
수학적 정의

- 거리 측정: 일반적으로 **유클리드 거리(Euclidean Distance)** 사용

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

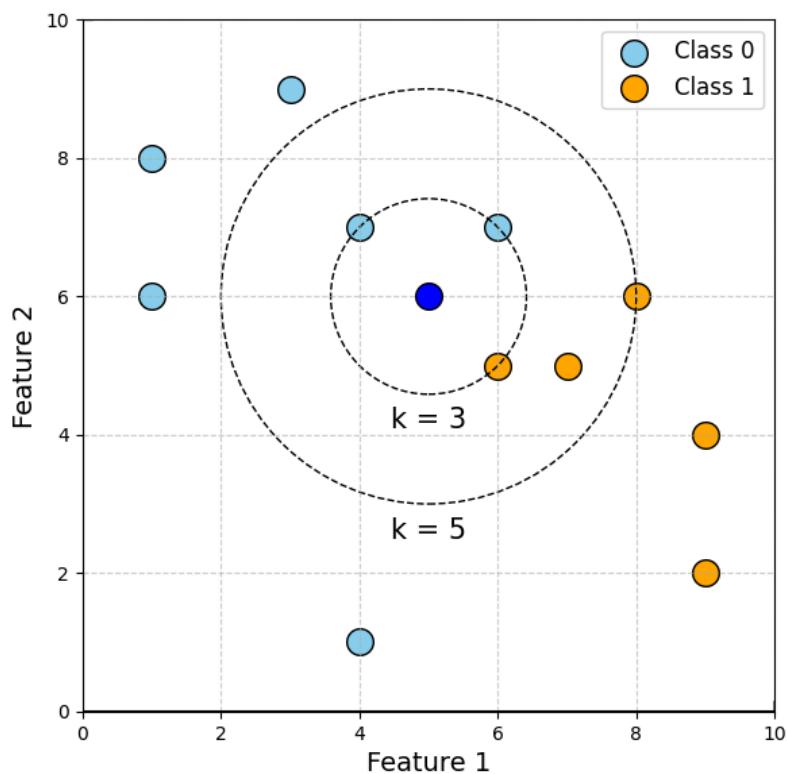


- 다양한 거리 척도 적용 가능 (맨하탄 거리, 코사인 거리 등)
 - 맨하탄 거리: $d(x, y) = \sum_{i=1}^n |x_i - y_i|$
 - 코사인 거리: $d(x, y) = 1 - \frac{x \cdot y}{|x||y|}$



K 값의 영향

- K가 너무 작을 때
 - 모델이 데이터에 과도하게 민감 (과적합 위험 ↑)
 - 노이즈에 영향을 많이 받음
- K가 너무 클 때
 - 너무 많은 이웃을 고려 → 경계가 흐려져 일반화 부족 (과소적합 위험 ↑)
- 보통 홀수 선택 (이진 분류에서 동점 방지)



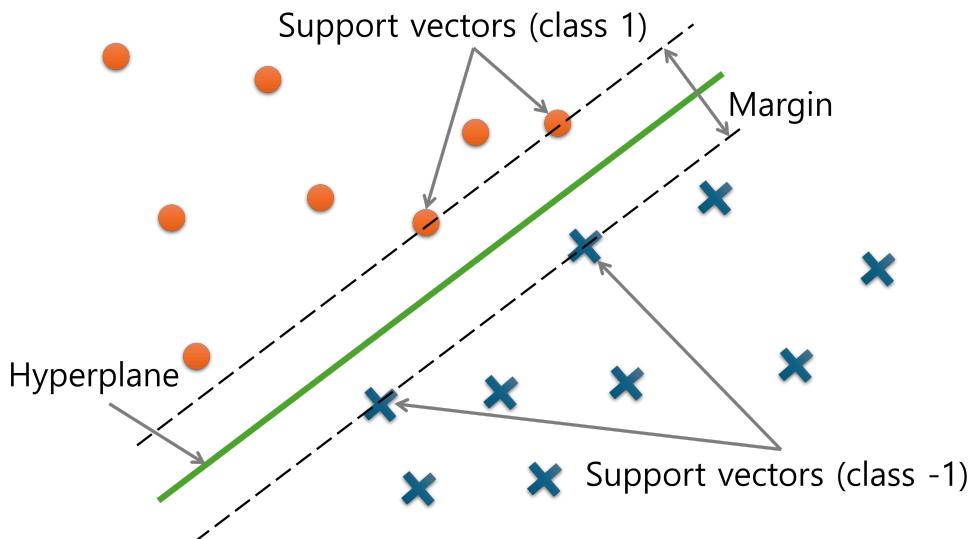
서포트 벡터 머신 (Support Vector Machine, SVM)

SVM이란?

- 지도 학습(Supervised Learning) 기반의 강력한 분류(Classification) 알고리즘
- 데이터를 가장 잘 구분하는 초평면(Hyperplane)을 찾는 방법
- 목적: 두 클래스 사이의 마진(Margin, 여백)을 최대화 하는 결정 경계(Decision Boundary)를 설정

핵심 개념

- 선형 분리가 가능한 경우: 두 클래스를 나누는 여러 직선(또는 초평면) 중에서 마진이 가장 넓은 것을 선택
- 마진: 결정 경계와 가장 가까운 데이터 점(서포트 벡터) 사이의 거리
- 이때, 경계를 결정하는 데이터 포인트들(초평면에 가장 가까이에 있는 데이터 포인트)을 서포트 벡터(Support Vectors)라고 부름



수학적 정의

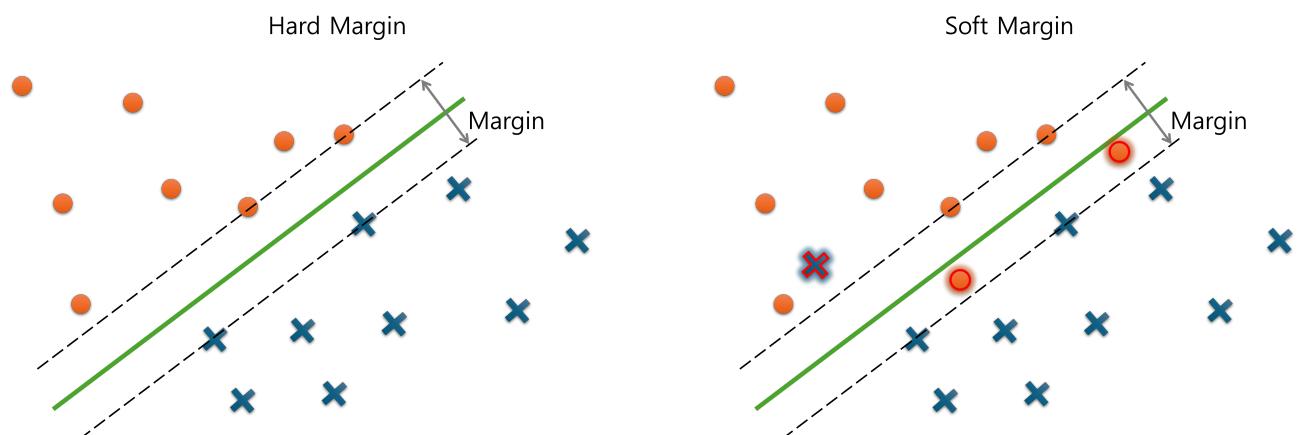
- 결정 경계(초평면): $w \cdot x + b = 0$
 - x : 데이터 포인트 축
 - w : 결정 경계를 정의하는 가중치 벡터 (초평면의 방향)
 - b : 초평면의 절편
- 결정 함수: $f(x) = w \cdot x + b$
 - $f(x) > 0$: 클래스 +1
 - $f(x) < 0$: 클래스 -1
 - $f(x) = 0$: 결정 경계에 위치 즉, 결정함수의 부호(sign)가 클래스 예측을 결정
- 데이터 포인트 x_i 가 올바르게 분류되려면, $y_i(w \cdot x + b) \geq 1 \quad \forall i, \quad y_i \in \{+1, -1\}$
 - y_i : 정답 클래스

- 서포트 벡터와 초평면 사이 거리: $\frac{1}{\|w\|}$
- 마진을 최대화하는 (w, b) 를 찾는 문제: 마진 $\frac{2}{\|w\|}$ 을 최대화하는 것은 $\frac{1}{2}\|w\|^2$ 을 최소화하는 것과 동일

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

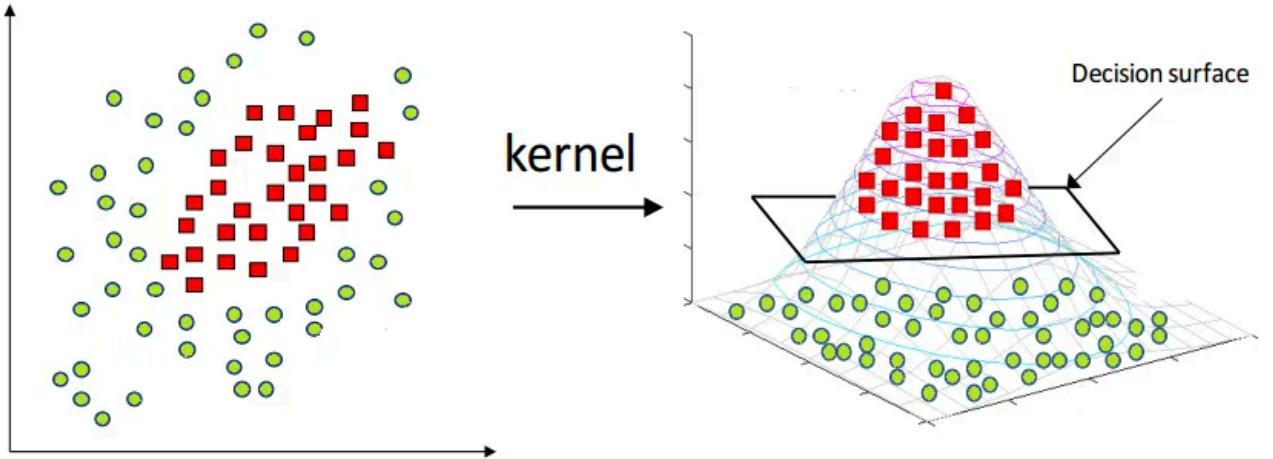
소프트 마진 (Soft Margin)

- 실제 데이터는 완벽하게 선형 분리가 되지 않을 수 있음
- 일부 오차(ξ)를 허용하여 더 유연한 결정 경계 설정
- 목적 함수: $\min \frac{1}{2}\|w\|^2 + C \sum_{i=1}^n \xi_i$
 - C : 오차 허용 정도를 조절하는 규제 파라미터
 - **C 가 크면** → 오차를 최소화하려고 해서 과적합 가능성 ↑
 - **C 가 작으면** → 오차를 좀 더 허용하고 마진을 크게 가져감



커널 트릭 (Kernel Trick)

- 선형 분리가 불가능한 경우, 데이터를 고차원 공간으로 매팅하여 선형 분리가 가능하게 함
- 대표적인 커널 함수
 - **선형 커널 (Linear Kernel)** $K(x_i, x_j) = x_i \cdot x_j$
 - **다항식 커널 (Polynomial Kernel)** $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$
 - **RBF 커널 (Gaussian Kernel)** $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$
 - **시그모이드 커널 (Sigmoid Kernel)** $K(x_i, x_j) = \tanh(\gamma(x_i \cdot x_j) + c)$



[Image source: Medium, What is the kernel trick? Why is it important?](#)

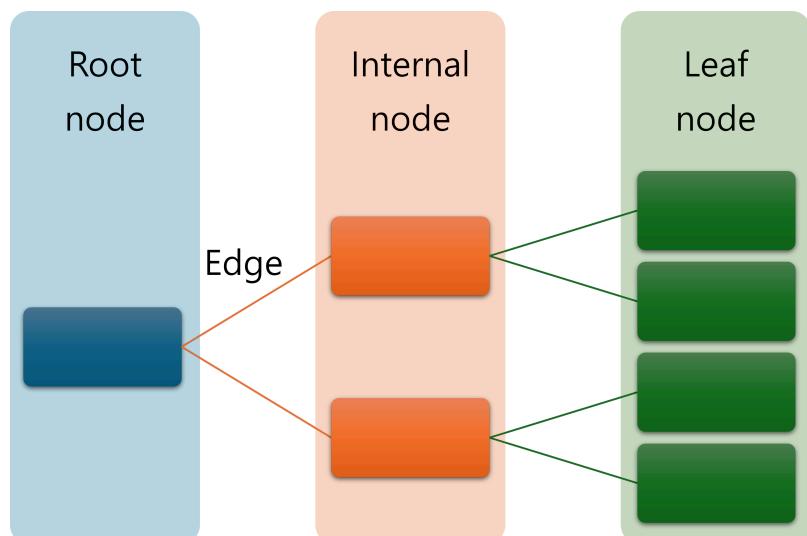
결정 트리 (Decision Tree)

결정 트리란?

- 데이터를 여러 기준(feature)에 따라 **분기(branch)** 시켜 나무(tree) 구조로 분류 또는 예측하는 알고리즘
- 분류(Classification)** 와 **회귀(Regression)** 모두 가능
- 사람의 의사결정 과정을 모방한 직관적 모델

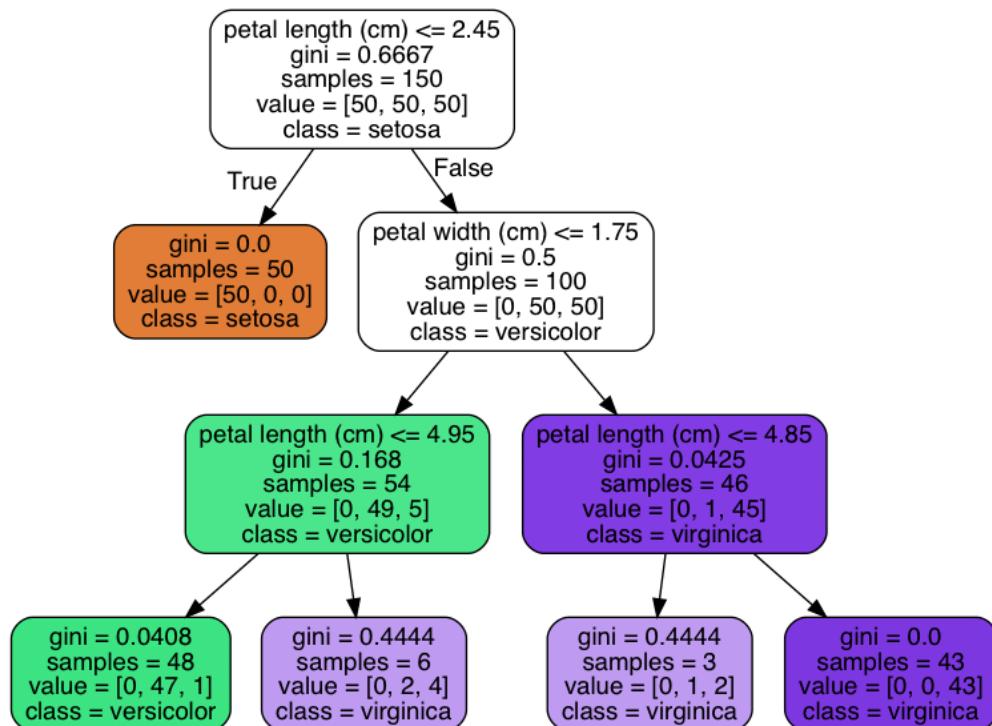
트리 구조

- 루트 노드(Root Node)**: 처음 시작하는 노드 (전체 데이터 집합)
- 내부 노드(Internal Node)**: 특정 기준(조건)에 따라 데이터를 나누는 분기점
- 엣지(Edge)**: 분기 조건을 따른 연결선
- 리프 노드(Leaf Node)**: 더 이상 분할되지 않고 최종 예측(클래스/값)을 제공



동작 원리

- 모든 데이터를 대상으로 특정 기준에 따라 분할
- 분할 후, 각 그룹에서 불순도가 가장 낮아지는 방향으로 다시 분할
- 이를 반복하다가 멈추는 시점(깊이 제한, 최소 샘플 수 도달 등)에서 리프 노드 생성



[Image source: Textklassification - Decision Trees & Random Forests - Scientific Figure on ResearchGate](#)

분할 기준 (불순도 측정 방법)

트리는 데이터를 나눌 때, 데이터가 얼마나 “순수해지는가”를 기준으로 합니다.

(1) 지니 불순도 (Gini Impurity)

$$Gini = 1 - \sum_{k=1}^K p_k^2$$

- p_k : 클래스 k의 비율
- 값이 작을수록 한 클래스에 치우쳐 있음 → 분류 성능 좋음

(2) 엔트로피 (Entropy, 정보 이득 기반)

$$Entropy = - \sum_{k=1}^K p_k \log_2(p_k)$$

- 정보 이론 기반
- 데이터의 불확실성(혼잡도)을 측정
- 엔트로피 감소량 = 정보 이득(Information Gain)

과적합 방지 방법 (Pruning, 가지치기)

- 사전 가지치기 (Pre-Pruning):** 트리 성장 제한
 - 최대 깊이(max_depth) 제한
 - 리프 노드 최소 샘플 수 지정
- 사후 가지치기 (Post-Pruning):** 완성된 트리에서 불필요한 가지 제거

인공 신경망 (Artificial Neural Network, ANN)

인공 신경망이란?

- 인간 뇌의 뉴런(Neuron) 구조를 모방한 기계 학습 알고리즘
- 여러 개의 노드(뉴런)가 층(layer)으로 연결되어 정보를 전달하고 변환
- 복잡한 패턴을 학습하고, 분류(Classification)·회귀(Regression)·특징 추출 등 다양한 문제에 활용 가능

구조

1. 입력층(Input Layer)

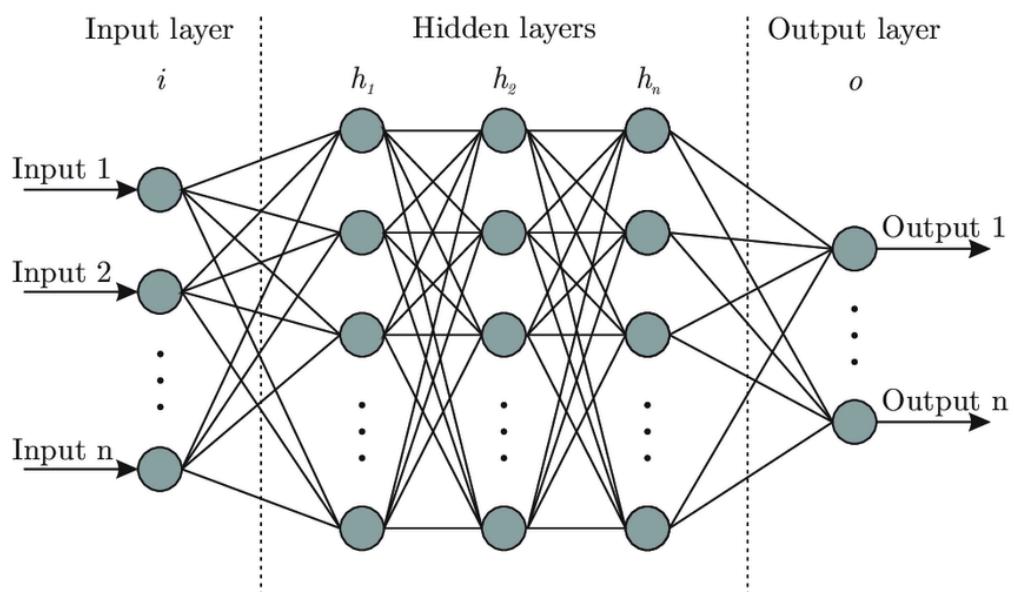
- 입력 데이터(feature)가 들어오는 층
- 각 노드는 하나의 변수와 대응

2. 은닉층(Hidden Layer)

- 입력을 가중치·활성화 함수를 거쳐 비선형적으로 변환
- 여러 층을 쌓아 복잡한 패턴 학습 가능

3. 출력층(Output Layer)

- 최종 예측 결과를 출력 (분류/회귀에 따라 달라짐)
- 분류 문제 → 소프트맥스(Softmax) 함수 등 사용



[Image Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks -](#)
[Scientific Figure on ResearchGate](#)

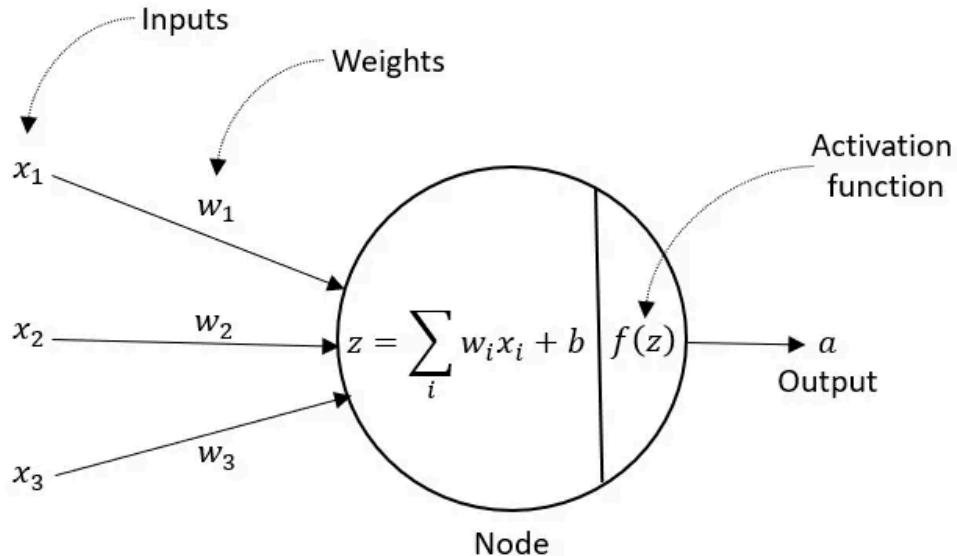
뉴런의 동작 원리

각 노드(뉴런)는 다음 계산을 수행합니다

$$z = \sum_{i=1}^n w_i x_i + b$$

$$y = f(z)$$

- x_i : 입력값
- w_i : 가중치 (Weight)
- b : 편향 (Bias)
- $f(\cdot)$: 활성화 함수 (Activation Function)
- y : 뉴런 출력



[Image source: A Review of the Math Used in Training a Neural Network, Medium](#)

모델 학습

순전파 (Forward Pass):

1. 입력부터 출력까지 순서대로 계산
2. 각 층에서 $z = wx + b$, $a = f(z)$ 계산

역전파 (Backward Pass):

1. 출력층에서 오차 계산: $\delta^L = \nabla_a C \odot \sigma'(z^L)$
2. 이전 층으로 오차 전파: $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$
3. 가중치 업데이트: $w^l = w^l - \eta \frac{\partial C}{\partial w^l}$

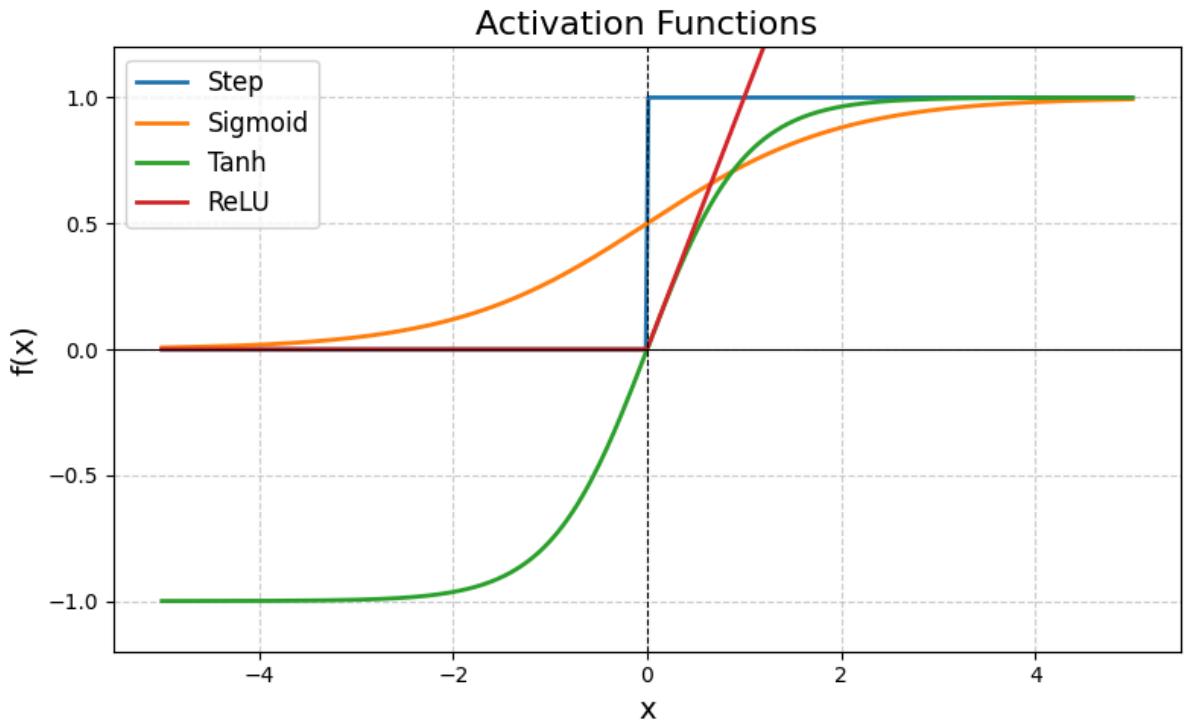
연쇄법칙 적용하여 그레디언트 계산

활성화 함수 (Activation Function)

- 뉴런의 출력에 비선형성을 주어, 신경망이 복잡한 패턴을 학습할 수 있도록 함

대표적인 함수들:

- **Sigmoid**: $\sigma(z) = \frac{1}{1+e^{-z}}$, 출력 범위 (0,1)
- **tanh**: $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ 출력 범위 (-1,1)
- **ReLU (Rectified Linear Unit)**: $f(z) = \max(0, z)$
- **Softmax**: $\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$ 다중 클래스 분류에서 확률로 변환, 출력층 전용



과적합 방지

- 드롭아웃 (Dropout)
 - 학습 과정에서 무작위로 일부 뉴런(노드)을 확률적으로 제거(drop) 하는 방법
 - 특정 뉴런이나 경로에 과도하게 의존하는 현상을 줄여 과적합 방지에 효과적
-

군집화(Clustering)

정의

- 레이블(label, 정답)이 없는 데이터를 비슷한 특성을 가진 그룹(군집, cluster)으로 묶는 **비지도 학습(Unsupervised Learning)** 알고리즘
- 데이터의 숨겨진 패턴이나 구조를 찾아내어 데이터 간 유사성을 기반으로 분류하는 과정
- 유사성(Similarity) 기반 거리(유clidean 거리, 코사인 유사도 등)나 확률적 분포를 사용해 데이터 간의 유사도를 측정

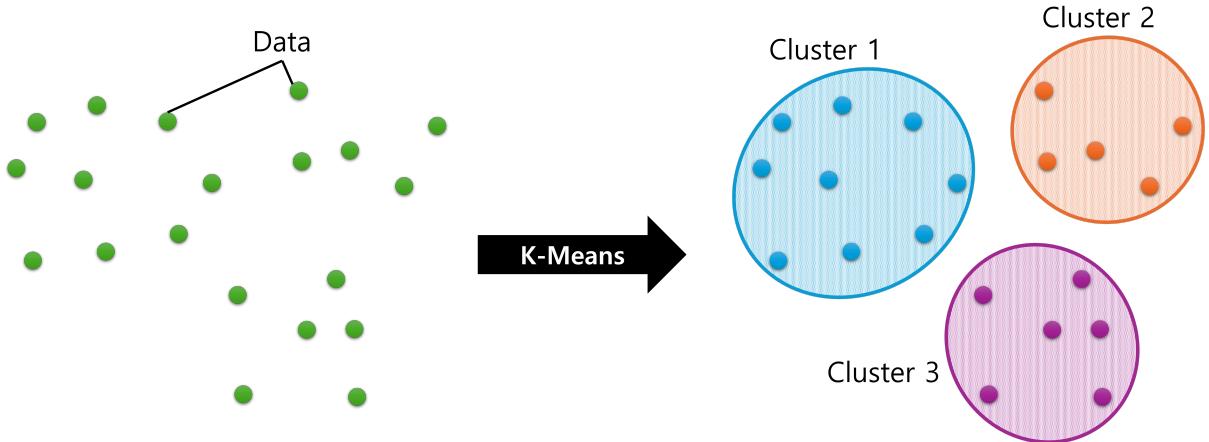
예시

- 고객 데이터를 나이, 소득, 소비 패턴에 따라 여러 집단으로 나누기
- 뉴스 기사를 주제별로 자동 분류하기 (스포츠, 정치, 경제 등)
- 이미지에서 색상을 그룹으로 묶어 대표 색상 추출

K-Means 클러스터링 (K-Means Clustering)

K-Means란?

- 대표적인 비지도 학습(Unsupervised Learning) 알고리즘
- 데이터를 K개의 그룹(클러스터)으로 나누는 방법
- 각 그룹은 하나의 중심점(centroid, 센트로이드)을 기준으로 형성됨



모델 학습

1. 초기 중심 선택 (Initialization)

- 무작위로 K개의 중심점(centroid) 선택

2. 할당 단계 (Assignment step)

- 각 데이터 포인트를 가장 가까운 중심점에 할당
- 거리 측정은 보통 유클리드 거리(Euclidean Distance) 사용

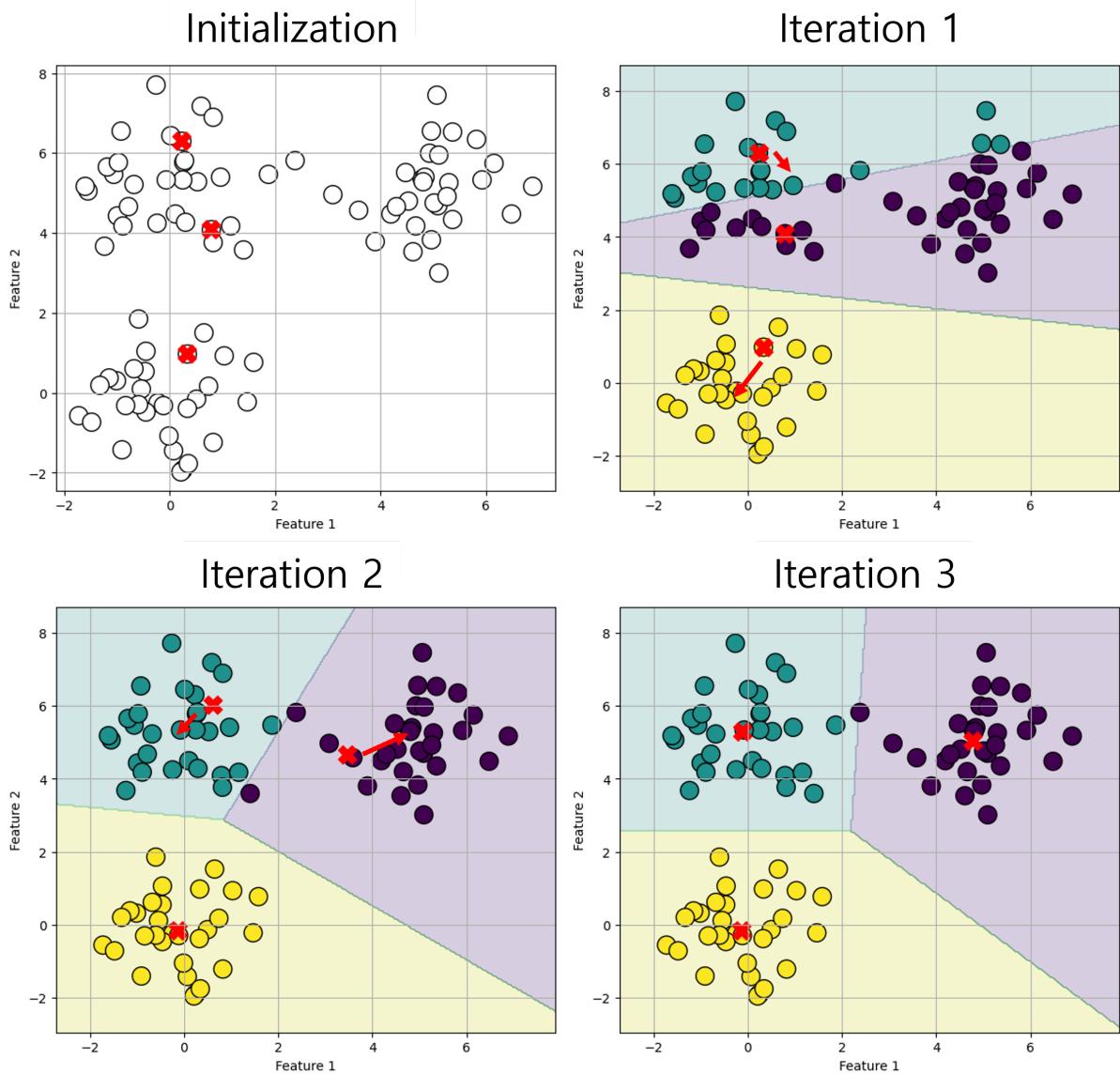
$$d(x, c_k) = \sqrt{\sum_{i=1}^n (x_i - c_{k,i})^2}$$

3. 업데이트 단계 (Update step)

- 각 클러스터에 속한 데이터들의 평균을 계산
- 새로운 중심점으로 갱신

4. 반복 (Iteration)

- 중심점이 더 이상 변하지 않거나, 변화가 일정 기준 이하일 때 종료



평가 지표

군집화의 품질을 정량적으로 평가하기 위해 실루엣 계수(Silhouette Coefficient)를 사용

$$s = \frac{1}{n} \sum_{i=1}^n \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

$a(i)$: 데이터 포인트가 자신의 군집 내 다른 데이터 포인트들과의 평균 거리 (응집도, Cohesion)
같은 군집의 데이터가 얼마나 가깝게 뭉쳐있는가

$b(i)$: 데이터 포인트가 속하지 않은 다른 군집의 데이터 포인트들과의 평균 거리 중 최솟값 (분리도, Separation)

다른 군집의 데이터가 얼마나 멀리 떨어져 있는가

- -1~1 범위를 지님
- 1에 가까울수록, 데이터 포인트가 자신의 군집에 잘 속해있고 다른 군집과 잘 분리됨을 의미

장점

- 구현이 간단하고 빠름 (대규모 데이터에도 확장 가능)
- 직관적이며 결과 해석이 쉬움
- 다른 클러스터링 알고리즘과 비교해 계산 효율성이 높음

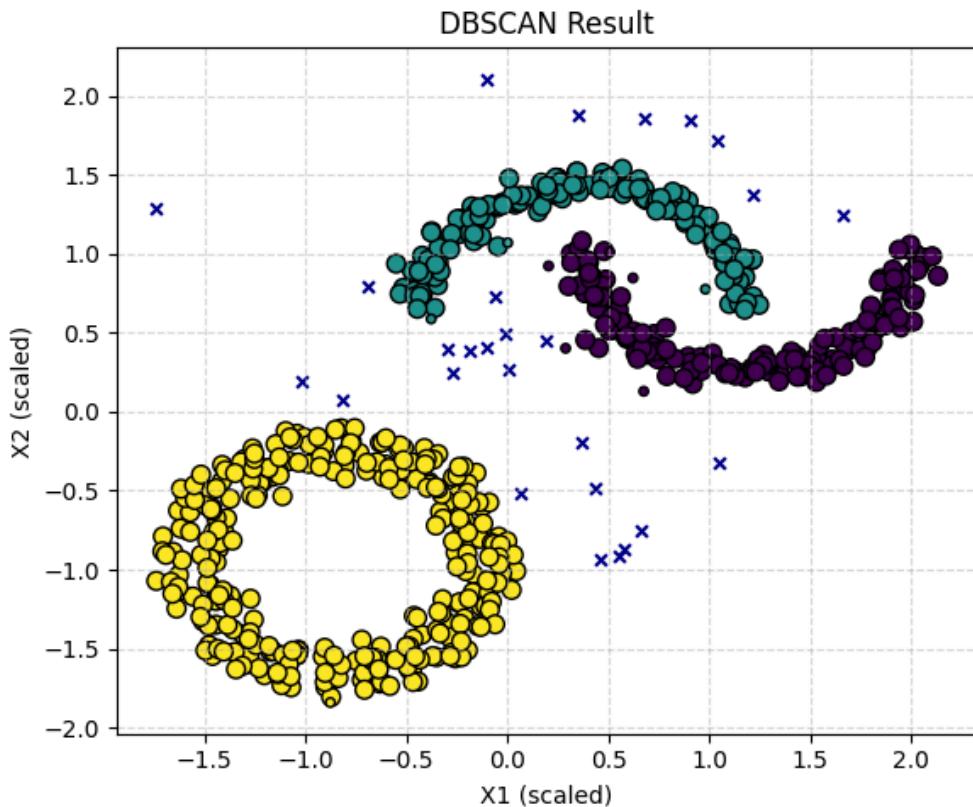
단점

- K(클러스터 개수)를 사전에 지정해야 함
 - 초기 중심점 선택에 따라 결과가 달라질 수 있음
 - 원형(구형) 클러스터에는 잘 작동하지만, 복잡한 모양의 데이터 분포에는 한계
-

DBSCAN (Density-based Spatial Clustering, 밀도 기반 클러스터링)

DBSCAN이란?

- 데이터의 **밀도(density)** 를 기준으로 클러스터를 형성하는 비지도학습 기반 클러스터링 알고리즘
- K-Means와 달리 **클러스터 개수(K)를 미리 지정할 필요 없음**



핵심 개념

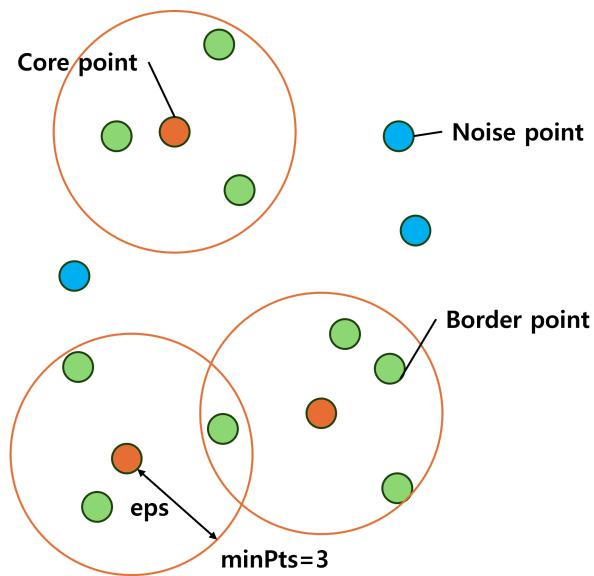
DBSCAN은 두 개의 핵심 파라미터에 의해 동작

- **ϵ (Epsilon):** 두 점이 같은 이웃(neighborhood)에 속한다고 볼 수 있는 최대 거리
- **MinPts (Minimum Points) :** 한 점이 "밀집 지역"의 중심점이 되기 위해 필요한 최소 이웃 수

주요 정의

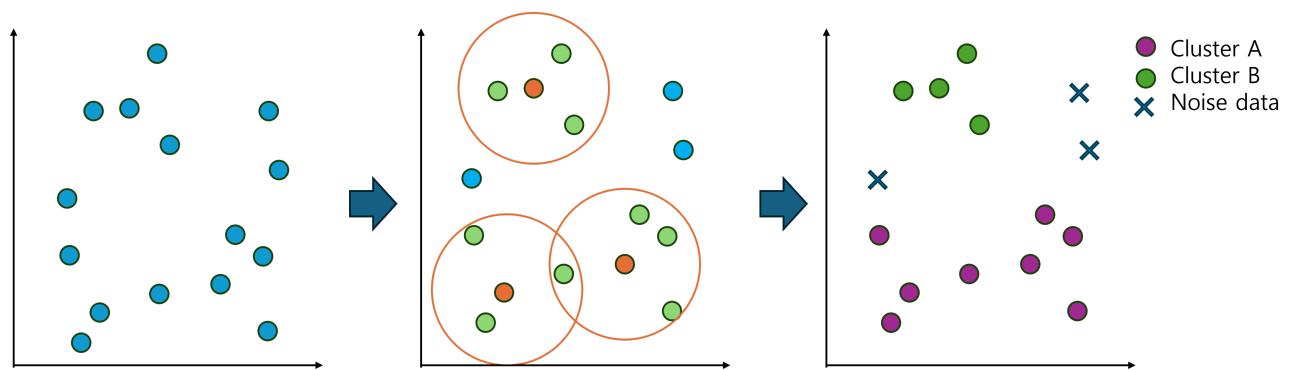
- **코어 포인트(Core Point):** 반경 ϵ 안에 MinPts 이상의 점이 있는 경우

- 경계 포인트(Border Point): 코어 포인트의 이웃이지만, 자기 자신은 코어 조건을 만족하지 않는 점
- 노이즈 포인트(Noise Point): 어떤 클러스터에도 속하지 않는 점



모델 학습

1. 데이터에서 아직 방문하지 않은 임의의 점 선택
2. 해당 점이 **코어 포인트**인지 확인 (ϵ 반경 내 MinPts 개수 이상 존재하는지 검사)
 - 코어 포인트라면 새로운 클러스터 생성
 - 이웃 점들을 재귀적으로 클러스터에 포함
3. 코어 포인트가 아니면:
 - Border Point → 기존 클러스터에 포함
 - Noise Point → 잡음으로 표시
4. 모든 점이 처리될 때까지 반복

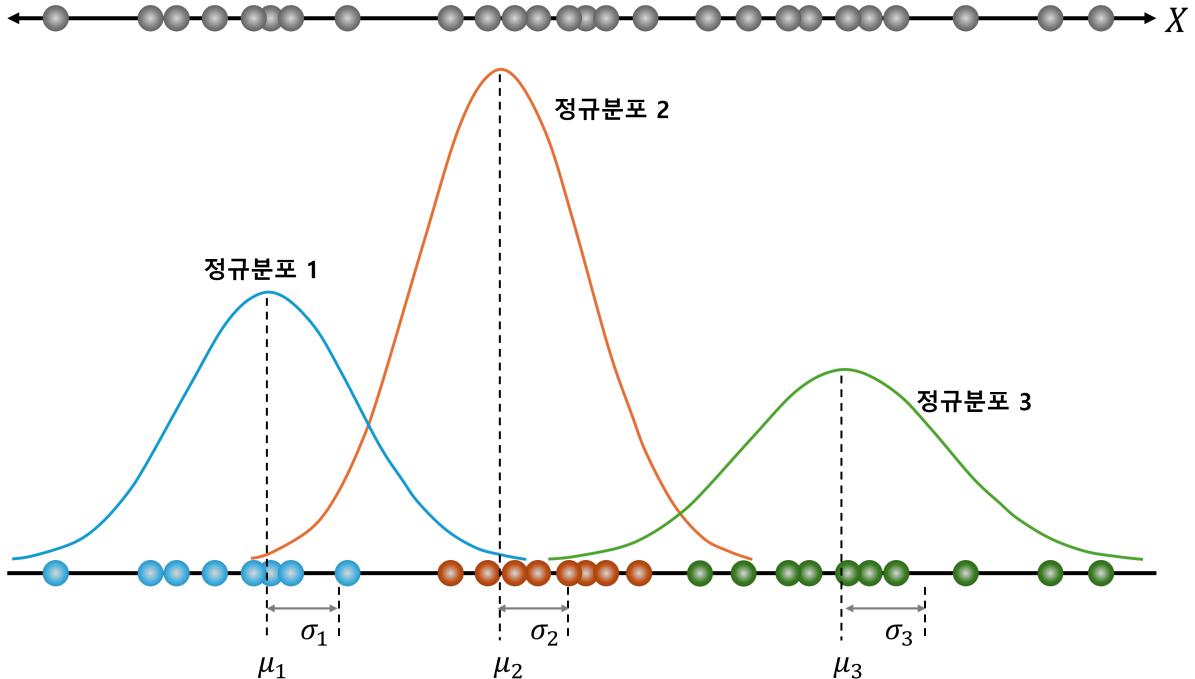


가우시안 혼합 모델 (Gaussian Mixture Model, GMM)

GMM이란?

- 비지도 학습(Unsupervised Learning) 알고리즘

- 데이터가 여러 개의 **가우시안 분포(정규분포)**의 혼합으로 이루어져 있다고 가정
- 각 데이터가 어느 분포(클러스터)에 속할 확률을 추정하여 클러스터링 수행



핵심 아이디어

- 데이터 분포를 여러 개의 가우시안 분포(평균 μ , 분산 σ)를 섞어서 모델링

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \sigma_k)$$
- $p(x)$: 데이터 x 가 혼합 분포에서 나올 확률
- K : 가우시안 성분의 개수 (클러스터 수)
- π_k : 각 분포의 가중치, $\sum \pi_k = 1$
- $\mathcal{N}(\mu_k, \sigma_k)$: 평균 μ_k , 분산 σ_k 를 가지는 정규 분포

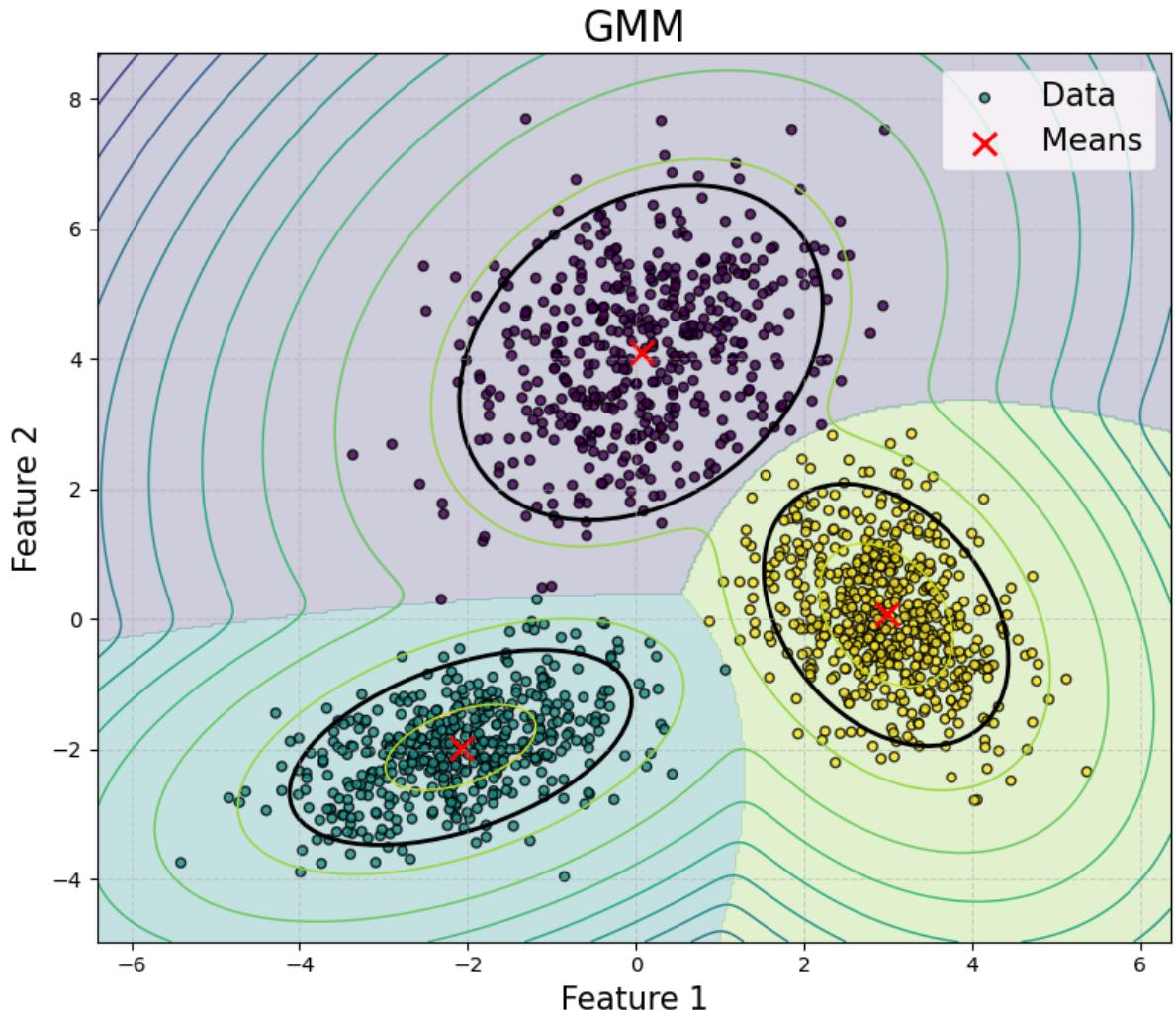
따라서, 각 데이터 포인트가 해당 클러스터에 속할 확률을 계산

모델 학습: EM 알고리즘

GMM은 Expectation-Maximization (EM) 알고리즘으로 학습

- 초기화: 평균(μ), 공분산(σ), 혼합계수(π) 초기값 설정
- E-step (기댓값 단계):
 - 데이터 x_i 가 클러스터 k 에 속할 확률 계산

$$\gamma_{ik} = \frac{\pi_k \mathcal{N}(x_i | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_i | \mu_j, \Sigma_j)}$$
- M-step (최대화 단계):
 - E-step에서 계산한 값을 이용하여 현재 데이터를 가장 잘 설명하는 평균, 공분산, 혼합계수의 값을 계산하고 반영
- 반복: 수렴할 때까지 2~3단계 반복



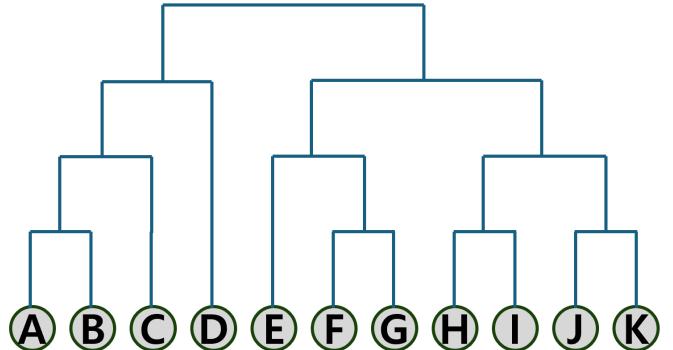
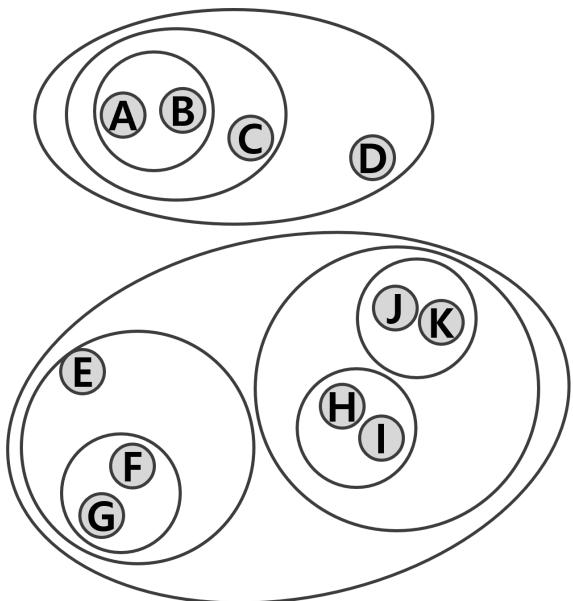
계층적 클러스터링 (Hierarchical Clustering)

계층적 클러스터링이란?

- **비지도 학습(Unsupervised Learning)** 클러스터링 기법 중 하나
- 데이터들을 계층적으로 병합하거나 분할하면서 **트리 구조(덴드로그램, Dendrogram)**를 형성
- 최종적으로 원하는 수준에서 클러스터를 결정

알고리즘 동작 과정

1. 초기화: 각 데이터 포인트를 개별 클러스터로 설정 (N 개의 데이터 $\rightarrow N$ 개의 클러스터)
2. 가장 가까운 클러스터 쌍 선택: 클러스터 간 거리를 계산하여 가장 가까운 두 클러스터 선택
3. 클러스터 병합: 선택된 두 클러스터를 합쳐 하나의 클러스터 생성
4. 반복: 클러스터가 K 개로 줄어들 때까지 반복



거리 측정 방법 (유사도 계산)

계층적 클러스터링은 데이터 간 또는 클러스터 간의 **거리 정의 방식**에 따라 결과가 달라집니다.

- **단일 연결법 (Single Linkage)** : 두 클러스터 간 가장 가까운 점들의 거리
- **완전 연결법 (Complete Linkage)** : 두 클러스터 간 가장 먼 점들의 거리
- **평균 연결법 (Average Linkage)** : 모든 점들 간 평균 거리
- **중심 연결법 (Centroid Linkage)** : 클러스터 중심간 거리
- **와드 연결법 (Ward's Linkage)** : 두 클러스터를 병합한 후에 발생되는 군집 내 제곱오차합, 클러스터 내 분산이 최소화되도록 합침

