

# 4일차. 분류 알고리즘 이해 및 실습

심선영 교수

# 강의 목표

- ❖ 머신러닝 시작예제 실습을 통해 머신러닝 기본 개념을 익힌다.
- ❖ Scikit-Learn 라이브러리를 주요 모듈을 익힌다.
- ❖ 머신러닝 주요개념을 이해하고 실습해 본다.
- ❖ 분류모델의 종류를 학습하고 실습해 본다.
- ❖ 함수 및 class를 import하는 방법을 복습한다.

# 강의 목차

## ❖ 머신러닝 시작 예제

- 학습예제 및 머신러닝의 데이터 셋, 사이킷런 주요 모듈 소개

## ❖ 머신러닝 주요개념

- 과대적합과 과소적합, 교차검증, 평가 지표

## ❖ 분류모델

- 의사결정나무, 앙상블, 랜덤포레스트, 로지스틱 회귀

## ❖ 하이퍼 파라미터 튜닝

## ❖ Import 리뷰

- 함수/클래스 import

# 강의 스케줄

시간	목차		활동
0.5h	Overview		PPT 학습
1h	머신러닝 시작예제		파이썬 실습
1.5h	머신러닝 주요개념		파이썬 실습
1h	분류 모델	의사결정나무	파이썬 실습
2h		앙상블 / 랜덤 포레스트	파이썬 실습
0.5h		로지스틱 회귀	
1h	import 정리 및 복습		파이썬 실습
0.5h	Wrap-Up		학습 정리

# 머신러닝 시작 예제

---

머신러닝 실습의 시작

# 머신러닝은 어떻게 스스로 학습하는가?

❖ 사례) 두 배우의 얼굴 분류



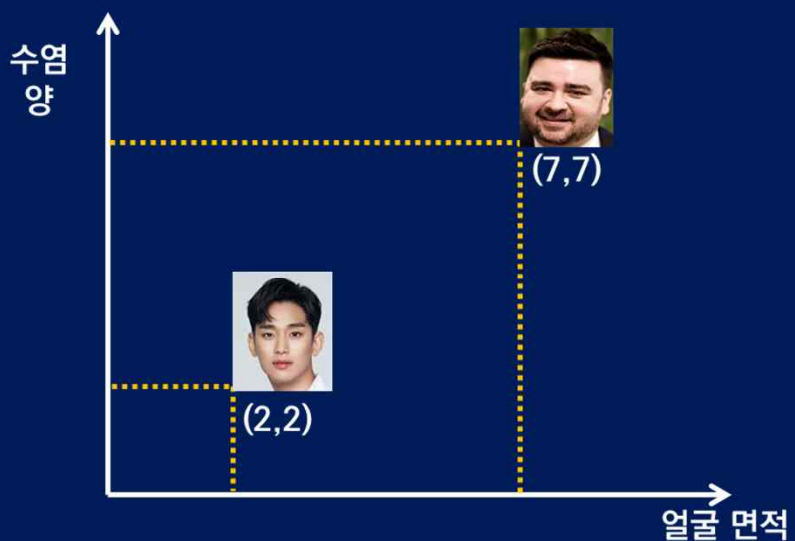
분류 기준: 얼굴 면적, 수염의 양

머신러닝 실습의 시작

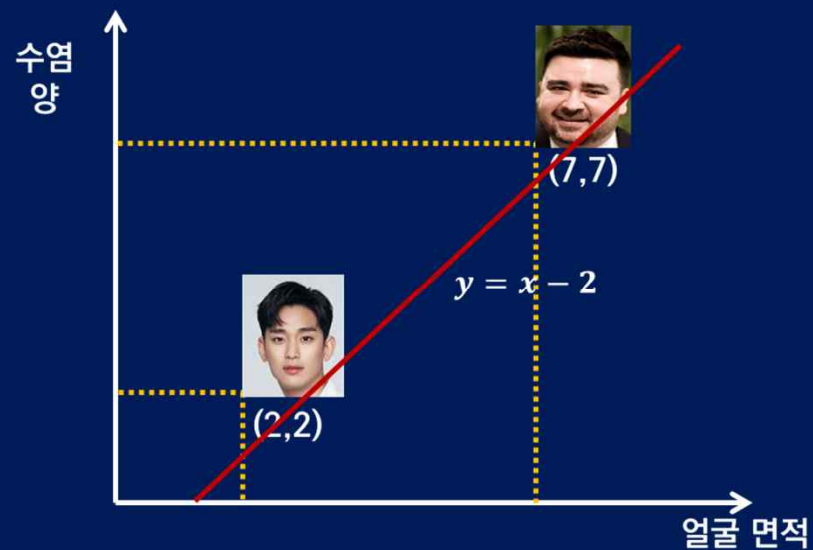
# 머신러닝은 어떻게 스스로 학습하는가?

❖ 사례) 두 배우의 얼굴 분류

## 1 데이터 준비



## 2 수학적 모델 생성



직선을 통해 두 사람의 위치가 분리되지 않음

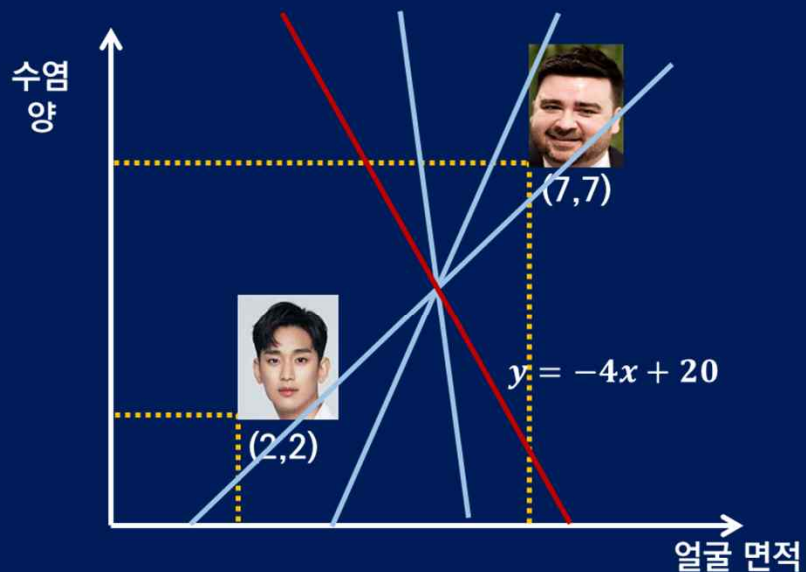
분류 불가

머신러닝 실습의 시작

# 머신러닝은 어떻게 스스로 학습하는가?

❖ 사례) 두 배우의 얼굴 분류

## 2 수학적 모델 생성



다량의 데이터

특징 파악

## 3 학습 완료



수학적 모델 구축

작업 수행



# 머신러닝 학습의 종류

## ❖ 지도학습(Supervised Learning)

- 훈련데이터(train data set)에 타겟(target)또는 레이블(label)이라고 불리는 정답이 포함된 학습방식
- 훈련이 끝나면 정답이 포함되지 않은 새로운 데이터 셋(test data set)을 사용하여 정답을 예측하게 함

## ❖ 비지도학습(Unsupervised Learning)

- 훈련데이터(train data set)가 타겟(target)또는 레이블(label)이라고 불리는 정답이 포함하지 않음
- 훈련데이터에 별다른 가이드라인이 없어 기계학습모델이 스스로 학습하여 모델을 만듦

머신러닝 실습의 시작

# 지도학습

## ❖ 개와 고양이의 분류 예시

레이블이 있는 데이터  
(Labeled Data)



개

고양이

개



고양이

개

고양이



결과



개



고양이

분류

머신러닝 실습의 시작

# 비지도학습

## ❖ 개와 고양이의 군집화 예시

레이블이 없는 데이터  
(Unlabeled Data)



결과

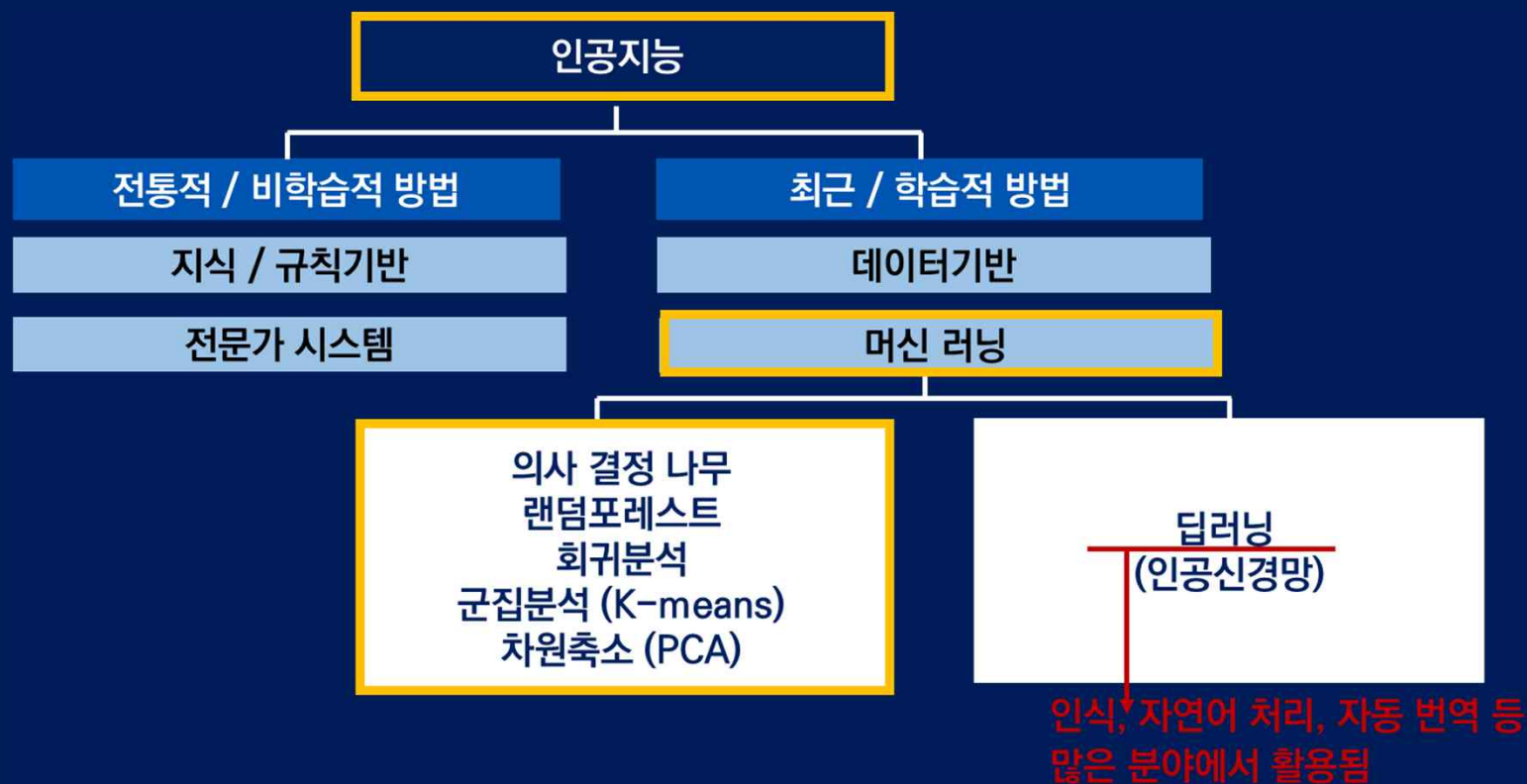
군집화



머신러닝 실습의 시작

# 인공지능 기술의 분류

❖ 인공지능 > 머신러닝 > 딥러닝

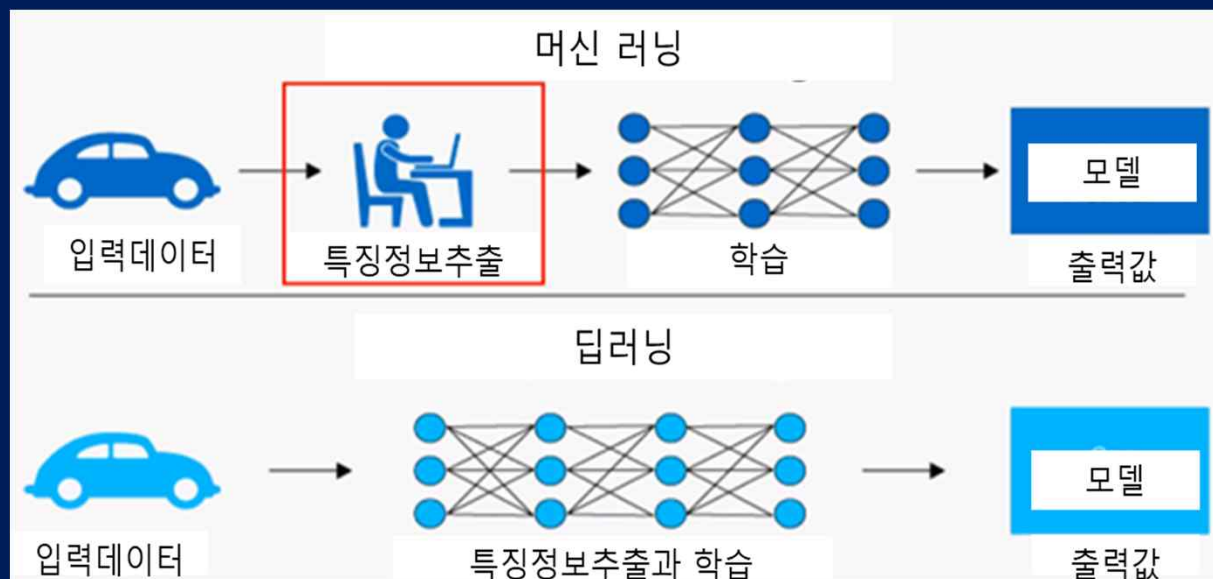


머신러닝 실습의 시작

# 머신러닝과 딥러닝 비교

❖ 전통적 머신러닝 - 기계가 학습을 하기 전에 미리 데이터 특성을 파악

■ 머신러닝에서는 특징추출이 중요!



머신러닝 실습의 시작


# Scikit-Learn 라이브러리

- ❖ 파이썬 머신러닝 라이브러리 중 가장 많이 사용되는 라이브러리
- ❖ 텐서플로(TensorFlow), 케라스(Keras)같은 딥러닝 전문 라이브러리가 최근 각광받지만 파이썬 기반 대표 머신러닝 라이브러리는 사이킷런임
- ❖ 아나콘다 설치 시 사이킷런도 같이 설치되므로 import하여 사용

머신러닝 실습의 시작

# Scikit-Learn 라이브러리

❖ <https://scikit-learn.org/stable/>

 [Install](#) [User Guide](#) [API](#) [Examples](#) [Community](#) [More](#)

## scikit-learn

Machine Learning in Python

[Getting Started](#) [Release Highlights for 1.0](#) [GitHub](#)

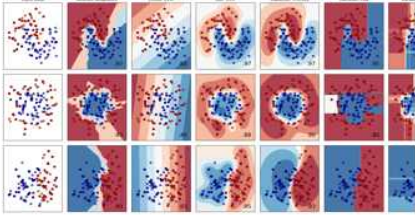
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

### Classification

Identifying which category an object belongs to.

**Applications:** Spam detection, image recognition.

**Algorithms:** SVM, nearest neighbors, random forest, and more...



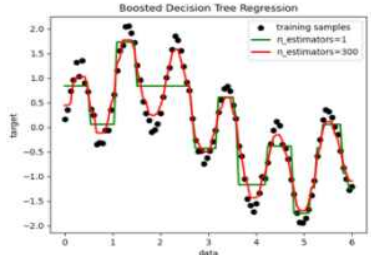
Examples

### Regression

Predicting a continuous-valued attribute associated with an object.

**Applications:** Drug response, Stock prices.

**Algorithms:** SVR, nearest neighbors, random forest, and more...




Examples

### Clustering

Automatic grouping of similar objects into sets.

**Applications:** Customer segmentation, Grouping experiment outcomes

**Algorithms:** k-Means, spectral clustering, mean-shift, and more...



Examples

머신러닝 실습의 시작

# 머신러닝의 전체 단계

1. 머신러닝 라이브러리 불러오기 (ex. 사이킷런)
2. 데이터 불러오기
3. 데이터 전처리 및 EDA를 통한 변수(feature) 설정
4. 학습데이터 셋(train data set)을 훈련 및 검증 데이터로 분리
5. 적합한 머신러닝 알고리즘의 선택하고 학습 (train)
6. 학습된 알고리즘에 검증 데이터에 적용하여 예측 (predict)
7. 예측값과 실제값을 비교해 오차를 측정하여 알고리즘의 성능 평가 (evaluation)
8. 3-8단계를 반복하며 알고리즘의 성능 고도화
9. 실제 예측을 원하는 데이터 셋(test data set)을 적용하여 최종 예측



머신러닝 시작 예제

# 사이킷런 주요 모듈

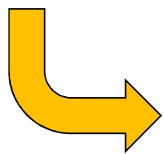
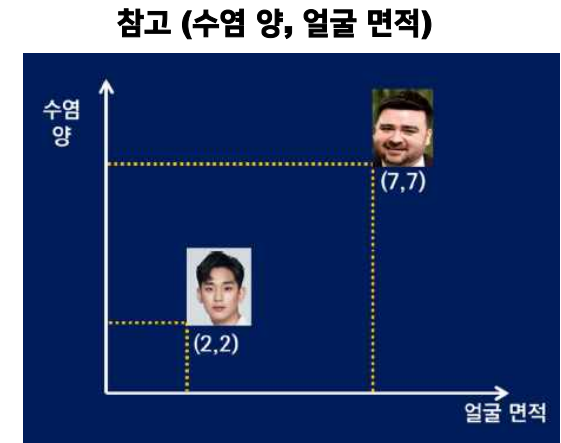
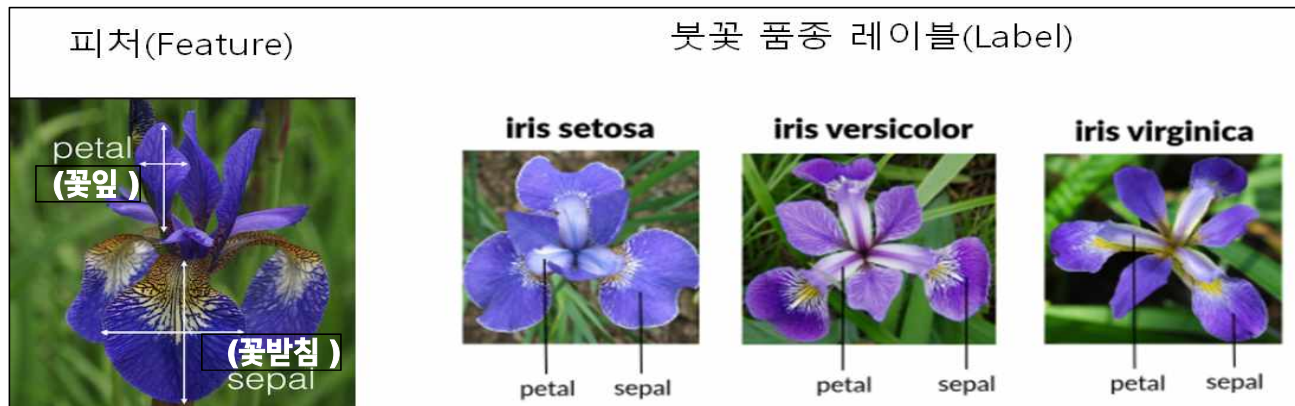
분류	모듈명	설명
예제 데이터	sklearn.datasets	사이킷런에 내장되어 예제로 제공하는 데이터 세트
데이터 분리, 검증 & 파라미터 튜닝	sklearn.model_selection	교차 검증을 위한 학습용/테스트용 분리, 그리드 서치(Grid Search)로 최적 파라미터 추출 등의 API 제공
피처 처리	sklearn.preprocessing	데이터 전처리에 필요한 다양한 가공 기능 제공(문자열을 숫자형 코드 값으로 인코딩, 정규화, 스케일링 등)
	sklearn.feature_selection	알고리즘에 큰 영향을 미치는 피처를 우선순위 대로 셀렉션 작업을 수행하는 다양한 기능 제공
	sklearn.feature_extraction	텍스트 데이터나 이미지 데이터의 벡터화된 피처를 추출하는 데 사용됨.
		예를 들어 텍스트 데이터에서 Count Vectorizer 나 Tf-Idf Vectorizer 등을 생성하는 기능 제공. 텍스트 데이터의 피처 추출은 sklearn.feature_extraction.text 모듈에, 이미지 데이터의 피처 추출은 sklearn.feature_extraction.image 모듈에 지원 API가 있음.
피처 처리 & 차원 축소	sklearn.decomposition	차원 축소와 관련한 알고리즘을 지원하는 모듈임. PCA, NMF, Truncated SVD 등을 통해 차원 축소 기능을 수행할 수 있음

분류	모듈명	설명
평가	sklearn.metrics	분류, 회귀, 클러스터링, 페어와이즈(Pairwise)에 대한 다양한 성능 측정 방법 제공 Accuracy, Precision, Recall, ROC-AUC, RMSE 등 제공
ML 알고리즘	sklearn.ensemble	앙상블 알고리즘 제공 랜덤 포레스트, 에이다 부스트, 그래디언트 부스팅 등을 제공
	sklearn.linear_model	주로 선형 회귀, 릿지(Ridge), 라쏘(Lasso) 및 로지스틱 회귀 등 회귀 관련 알고리즘을 지원. 또한 SGD(Stochastic Gradient Descent) 관련 알고리즘도 제공
	sklearn.naive_bayes	나이브 베이즈 알고리즘 제공. 가우시안 NB, 다항 분포 NB 등.
	sklearn.neighbors	최근접 이웃 알고리즘 제공. K-NN 등
	sklearn.svm	서포트 벡터 머신 알고리즘 제공
	sklearn.tree	의사 결정 트리 알고리즘 제공
	sklearn.cluster	비지도 클러스터링 알고리즘 제공 (K-평균, 계층형, DBSCAN 등)
	sklearn.pipeline	피처 처리 등의 변환과 ML 알고리즘 학습, 예측 등을 함께 묶어서 실행할 수 있는 유틸리티 제공

머신러닝 시작 예제

# 학습 예제 소개

## ❖ 학습예제의 데이터 셋



4개의 특성들(features)                      클래스(class)

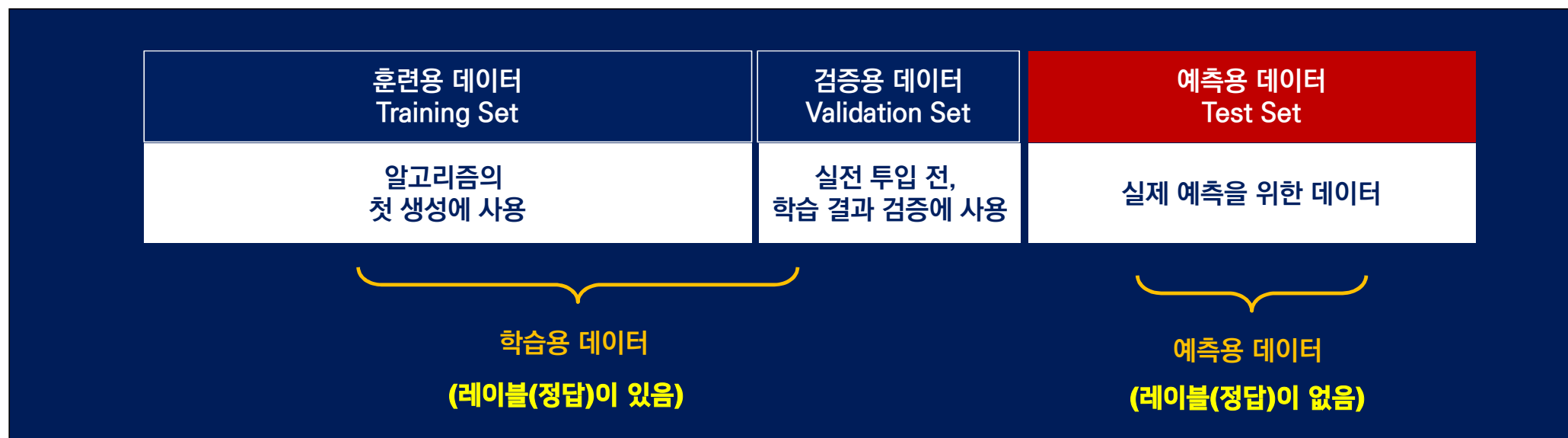
sepal_length	sepal_width	petal_length	petal_width	species (Label, Target)
5.8	4.0	1.2	0.2	setosa
5.1	2.5	3.0	1.1	versicolor
6.6	3.0	4.4	1.4	versicolor
5.4	3.9	1.3	0.4	setosa
7.9	3.8	6.4	2.0	virginica

0,1,2

머신러닝 시작 예제

# 머신러닝 데이터 셋

❖ 훈련용(train set), 검증용(validation set), 예측용 (test set)



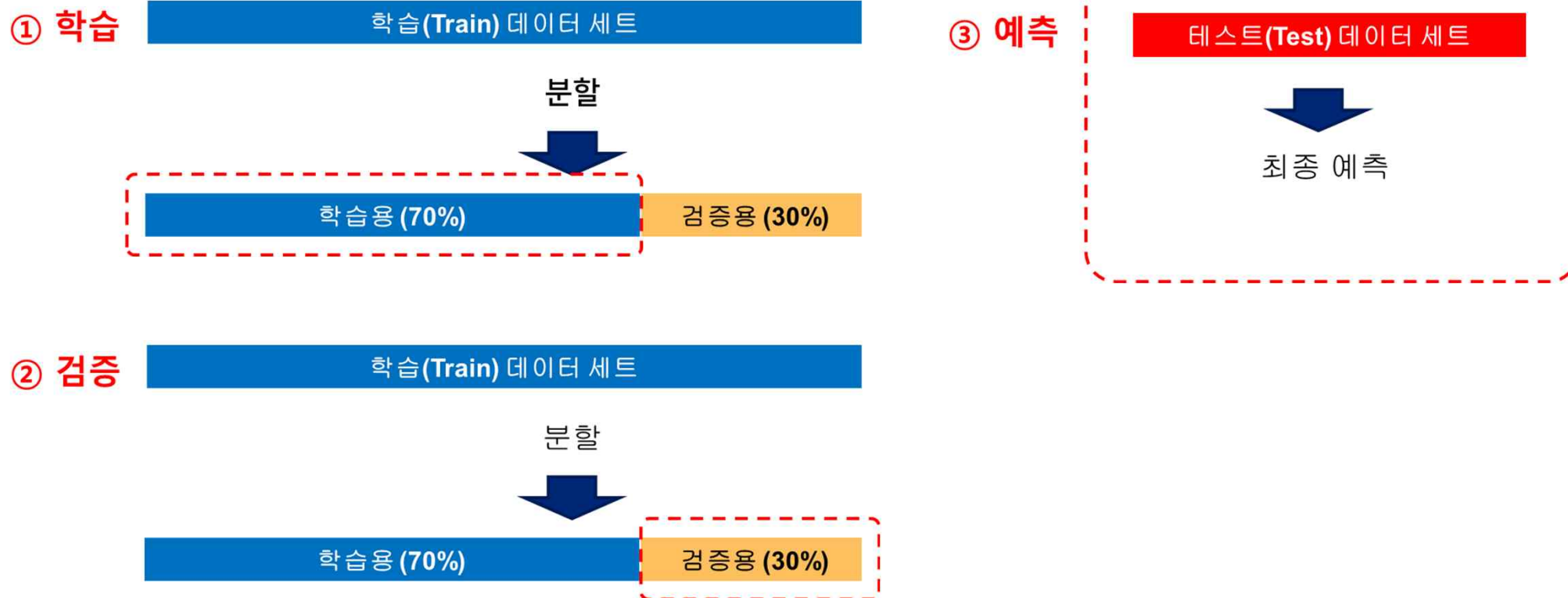
sepal_length	sepal_width	petal_length	petal_width	species
5.8	4.0	1.2	0.2	setosa
5.1	2.5	3.0	1.1	versicolor
6.6	3.0	4.4	1.4	versicolor
5.4	3.9	1.3	0.4	setosa
7.9	3.8	6.4	2.0	virginica

sepal_length	sepal_width	petal_length	petal_width
5.8	4.0	1.2	0.2
5.1	2.5	3.0	1.1
6.6	3.0	4.4	1.4
5.4	3.9	1.3	0.4
7.9	3.8	6.4	2.0

머신러닝 시작 예제

# 머신러닝의 데이터 셋

❖ 훈련용(train set), 검증용(validation set), 예측용 (test set)



# 머신러닝 주요 개념

---

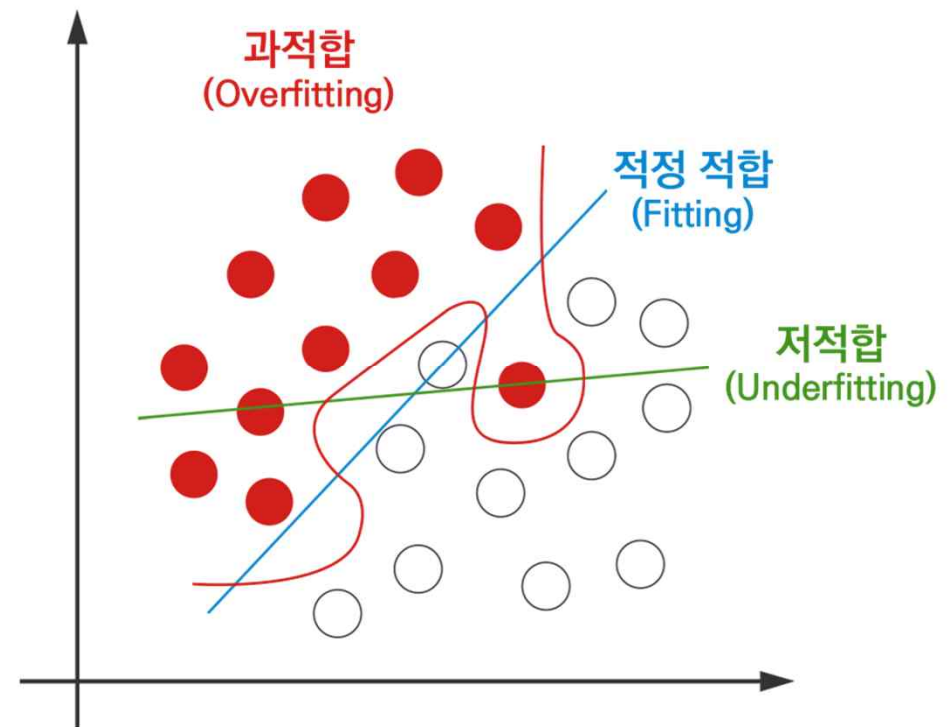
# 과적합(Overfitting)과 저적합(Underfitting)

## ❖ 과적합 (과대적합)

- 학습 모델이 현재의 훈련데이터에 지나치게 적합되어 학습하여 복잡한 모델을 생성
- 일반화 (새로운 데이터에 대응) 어려움

## ❖ 저적합 (과소적합)

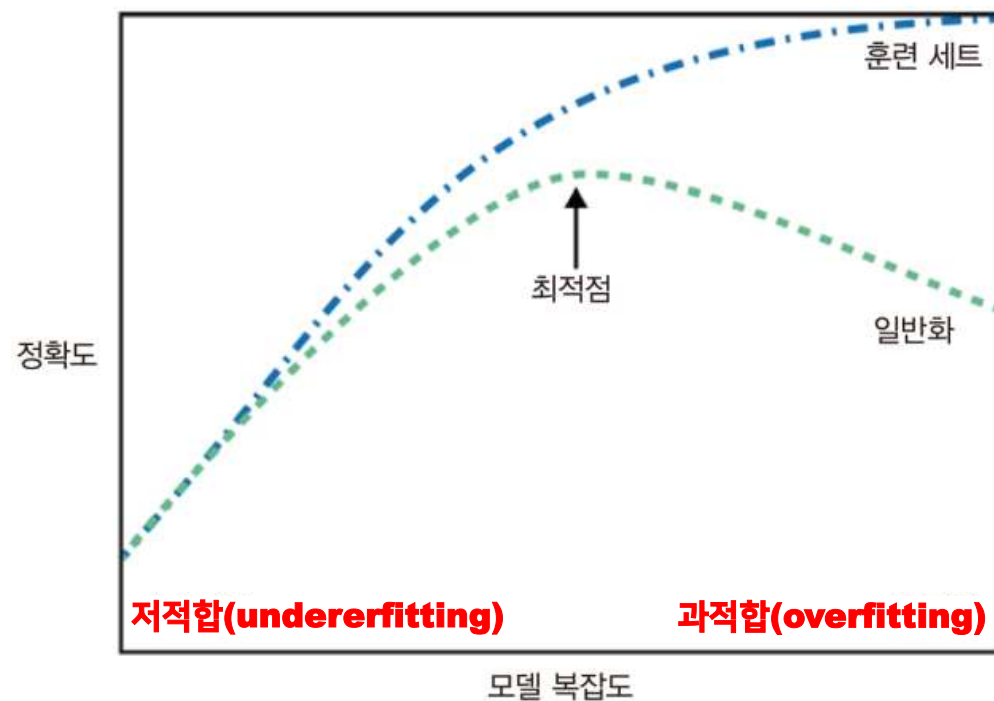
- 주어진 훈련데이터도 충분히 학습을 하지 못하여 (또는 훈련데이터가 충분치 않아서) 너무 간단한 모델을 생성
- 예측 성능 떨어짐



머신러닝 주요 개념

# 과적합(Overfitting)과 저적합(Underfitting)

❖ 모델 복잡도와 예측 정확도의 관계



# 교차 검증

## ❖ 과적합 문제를 해결하는 방법

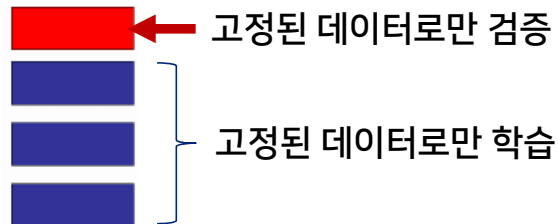
- 모델 복잡도를 너무 높이지 않는다
- 고정된 데이터로 학습과 검증을 하지 않는다.

## ❖ 고정된 학습데이터와 검증 데이터로 평가한다면?

→ 학습한 데이터에만 최적의 성능을 발휘하도록 편향되게 모델이 만들어짐 (과적합!)

→ 이를 방지하기 위해 **교차 검증**을 사용함

교차 검증이 아닌 경우

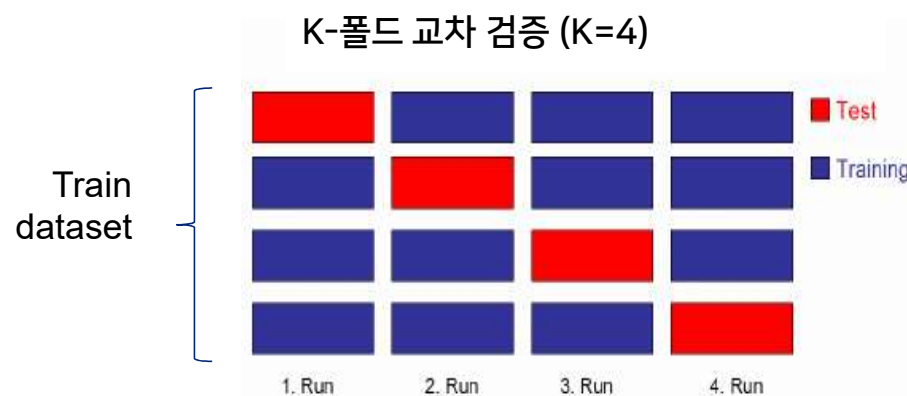
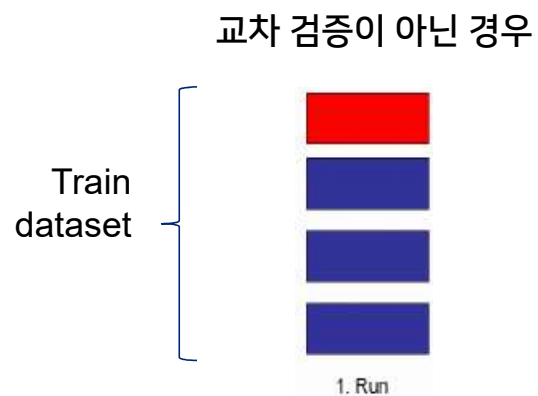




# 교차 검증

## ❖ K-폴드 교차검증

- 가장 보편적으로 사용하는 교차 검증 기법
- K개의 데이터 폴드 세트를 구성, K번 만큼 각 폴드 세트에서 학습과 검증을 반복적으로 수행
- 예를 들어, 4번을 반복수행 했다면, 이를 평균하여 K폴드 평가 결과로 사용함



# 평가 지표

## ❖ 정확도 (Accuracy)

■  $\text{정확도} = \frac{\text{예측 결과가 정확한 데이터 건수}}{\text{전체 예측 데이터 건수}}$

## ❖ 오차 행렬

■ 이진분류에서 성능지표로 활용

■  $\text{정확도} = \frac{TN+TP}{TN+FP+FN+TP}$

	예측: Negative	예측: Positive
실제: Negative	TN (True Negative)	FP (False Positive)
실제: Positive	FN (False Negative)	TP ( True Positive)

## ❖ 정밀도와 재현율

■  $\text{정밀도} = \frac{TP}{FP+TP}$  (예측을 positive 로 한 대상 중 정확하게 한 건수의 비율)

→(코로나) 양성으로 나온 사람 중 실제 양성자의 비율

→ positive 예측성능을 더욱 정밀하게 보기 위한 지표

■  $\text{재현율} = \frac{TP}{FN+TP}$  (실제 positive 대상 중 정확하게 positive로 예측 한 건수의 비율)

→ (코로나) 실제 양성인 사람 중 양성으로 정확하게 예측된 비율

→ 민감도

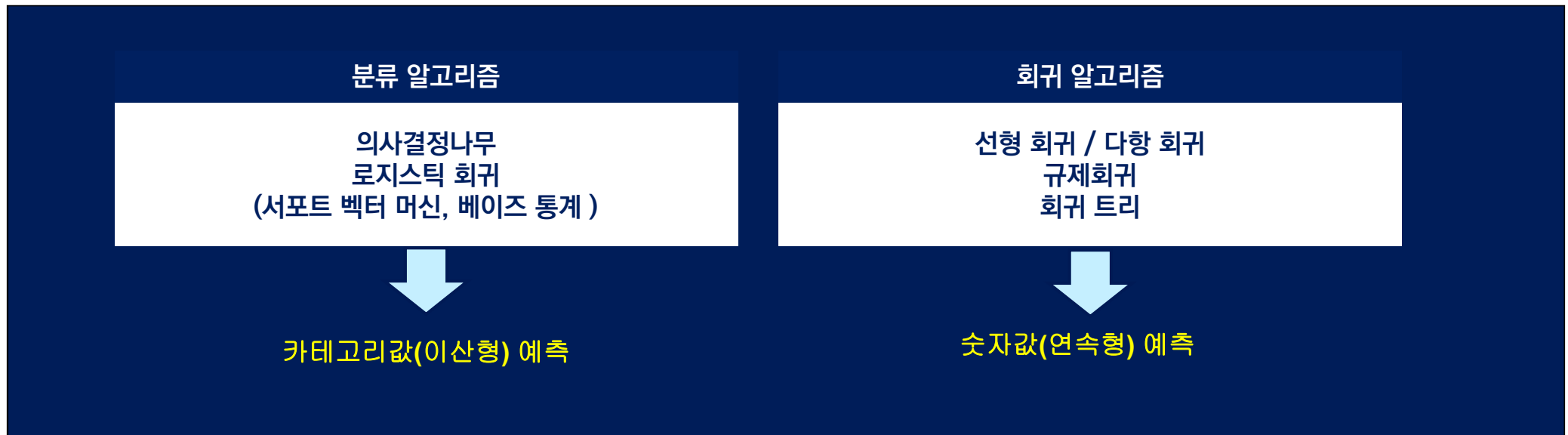
# 분류 모델

---

분류 모델

# 지도학습의 두 유형

- ❖ 분류 (classification) – 어떤 집단에 속하는 지 판별해 내는 것
- ❖ 회귀 (regression) – 주어진 값에 대한 결과값을 예측하는 것



분류 모델

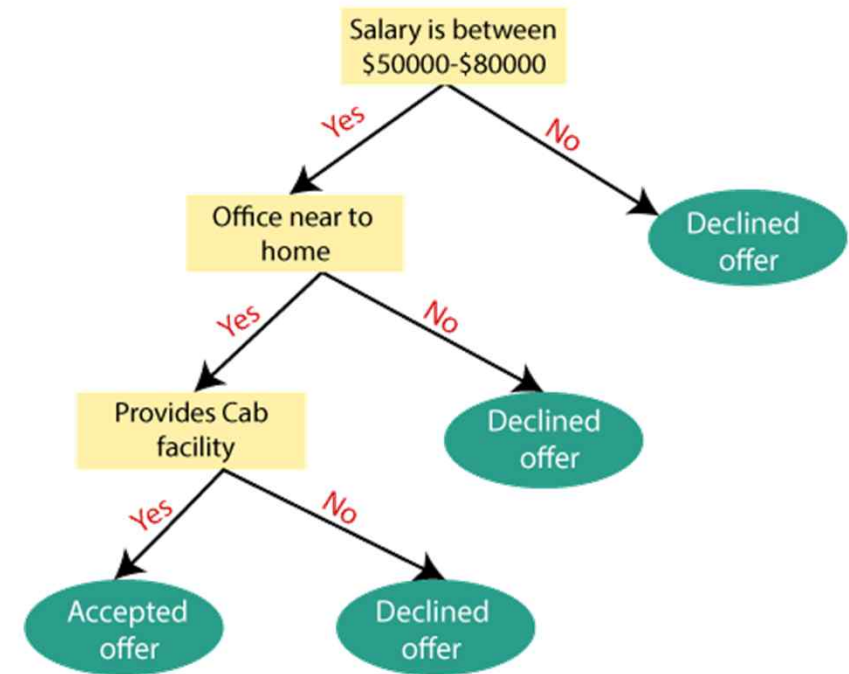
# 분류 모델의 종류

- ❖ 의사결정나무 (Decision Tree)
- ❖ 앙상블 (서로 같거나/다른 알고리즘 간 결합)
  - Bagging - Random Forest
  - Boosting - Gradient Boosting, xgBoost, LightGBM
- ❖ 로지스틱 회귀

분류 모델

# 의사결정나무

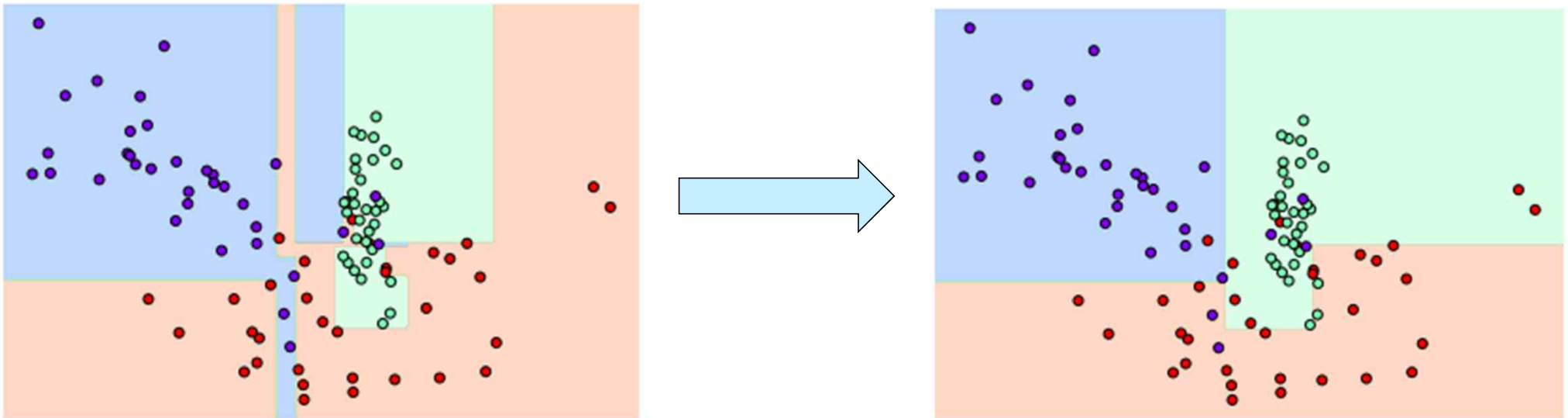
- ❖ 의사결정나무는 매우 쉽고, 스케일링이나 정규화 등 사전 데이터 가공의 영향이 적음
- ❖ 예측 성능을 향상시키기 위해 **복잡한 규칙 구조**를 가져야 하고 이로 인한 **과적합**이 발생, 예측성능 떨어질수도 있음
  - **depth**가 깊어질수록 의사결정나무의 예측성능은 저하될 수 있음



# 의사결정나무

❖ 복잡하고 엄격한 학습 규칙으로 인한 과적합 사례

■ 의사결정나무의 max depth를 제한하거나, 말단노드의 데이터 수를 완화하는 방식으로 해결



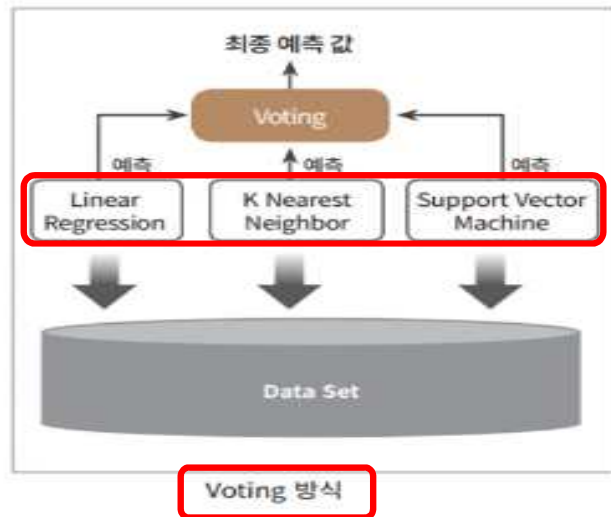
# 앙상블 (Ensemble)

❖ 학습 모델 간 결합을 통해 단순학습모델의 한계를 극복하고 성능을 향상시키는 방법

❖ 앙상블의 전통적 유형

## ■ 보팅(Voting)과 배깅(Bagging)

- 보팅(Voting) – 서로 **다른** 알고리즘을 가진 분류기를 결합
- 배깅(Bagging) – 모두 **같은** 알고리즘을 가진 분류기를 결합 (ex. 랜덤 포레스트)





# 앙상블에서 최종 예측값 결정

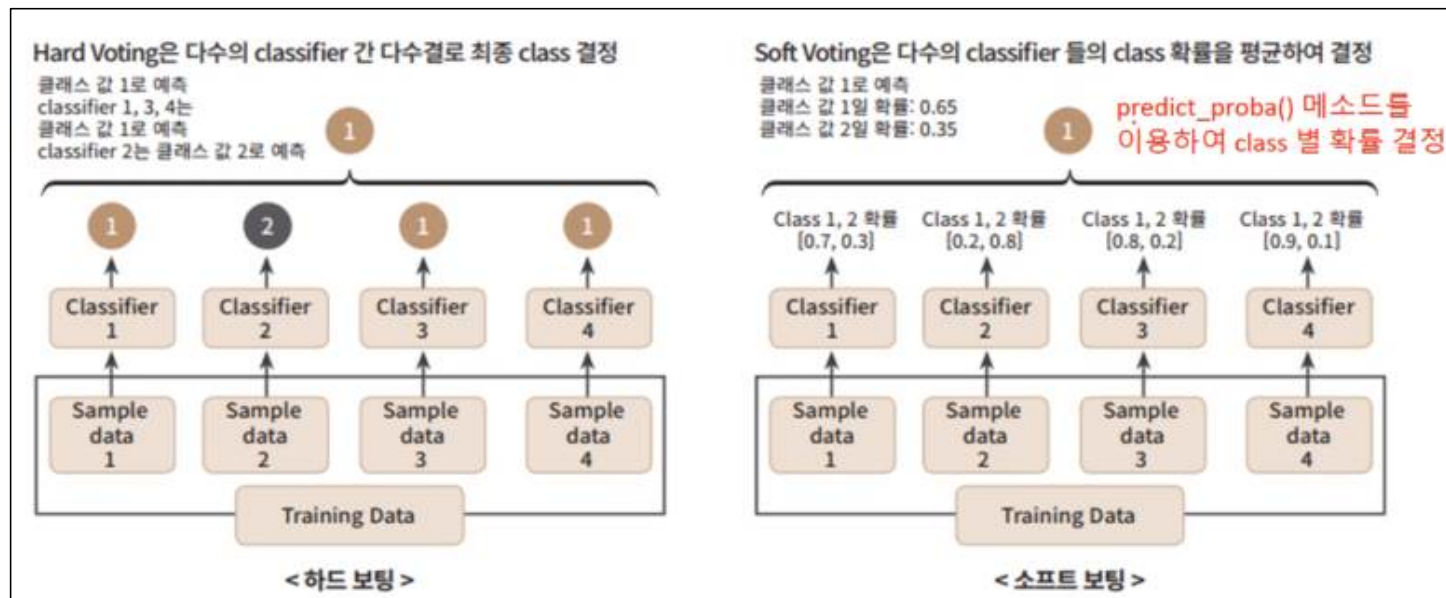
❖ 보팅 (Voting) – 여러 개 모델의 예측값을 이용하여 최종 예측값을 결정

## ■ 하드 보팅

- **다수결** 원칙과 유사, 다수의 모델이 예측한 값을 최종 예측값으로 결정

## ■ 소프트 보팅

- 각 모델들이 예측한 레이블별 결정 확률을 평균 → 확률이 가장 높은 레이블 값을 최종 예측값으로
- 일반적으로 많이 사용하는 방법



# 랜덤 포레스트 (Random Forest)

## ❖ 앙상블 중 배깅 기법의 하나

- 같은 알고리즘으로 여러 개의 모델을 만들고 보팅으로 최종 예측값 결정

## ❖ 개별 모델의 기반 알고리즘은 의사결정나무

## ❖ 개별 의사결정나무가 학습하는 데이터 세트는 Bootstrapping 분할된 데이터

- ✓ 단, 학습 데이터는 원본 데이터를 샘플링 추출하여 사용함 (Bootstrapping)
- ✓ 배깅 (샘플링 데이터간 중첩을 허용) VS 교차검증 (데이터 세트간의 중첩을 허용하지 않음)



분류 모델

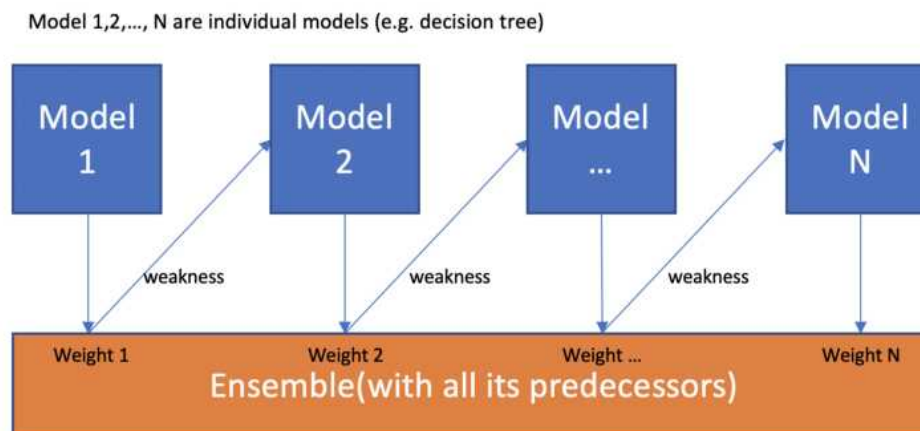
# 부스팅 (Boosting)

## ❖ 앙상블의 최신 유형

- 여러 개의 모델이 순차적으로 학습
- 이전 단계 학습에서 **저적합**되어 예측이 틀린 데이터에 대해서 다음 단계에서 가중치를 부여하여 학습
- 계속해서 **가중치를 “부스팅(boosting)”하면서 학습을 진행**한다는 의미

## ❖ 부스팅의 종류

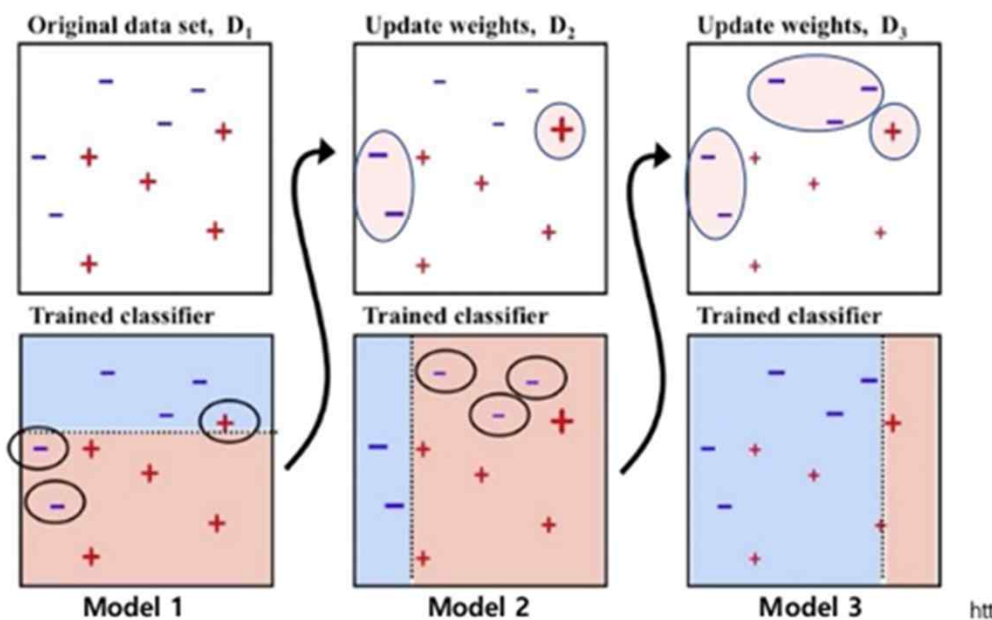
- AdaBoost
- GBM (Gradient Boost Machine)
- XGBoost (eXtra Gradient Boost)
- LightGBM (Light Gradient Boost)
  - 성능은 유사하지만 속도 빠름



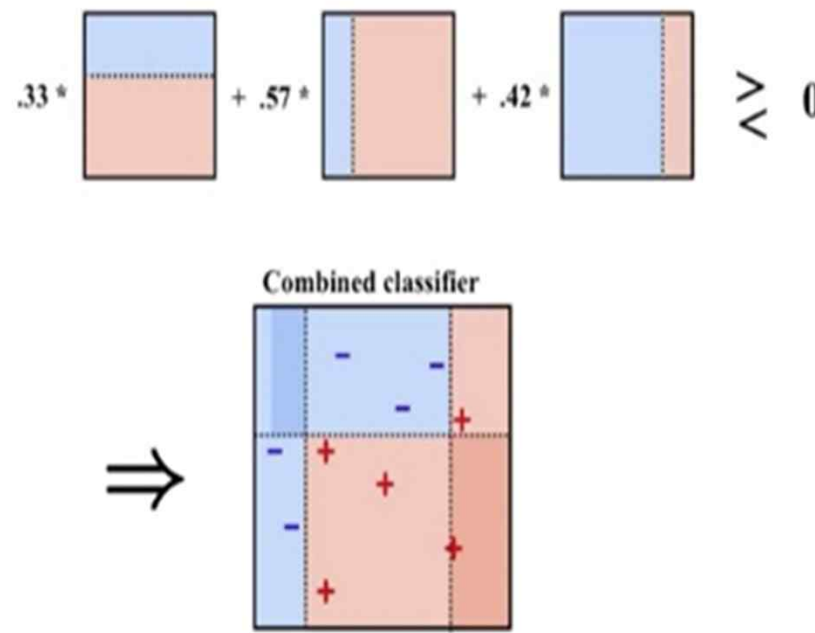
분류 모델

# AdaBoost 예시

- Model1에서 잘못 예측한 데이터에 가중치를 부여
- Model2는 잘못 예측한 데이터를 분류하는데 더 집중
- Model3는 Model1, 2가 잘못 예측한 데이터를 분류하는데 집중



**[3개의 모델에 각 가중치를 적용 후 합산하여 최종 모델 생성]**



# 앙상블의 다양한 사례

❖ 앙상블의 성능 향상을 기대하려면..

■ 개별학습 모델이 일정수준 이상의 성능을 확보 (사례3 X)

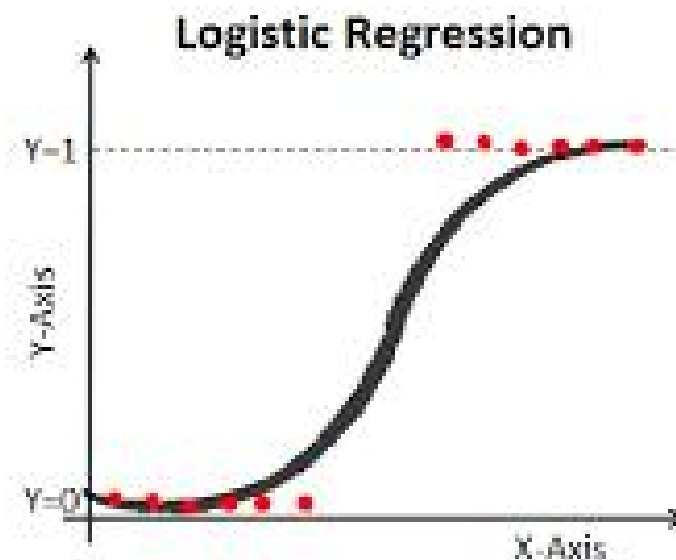
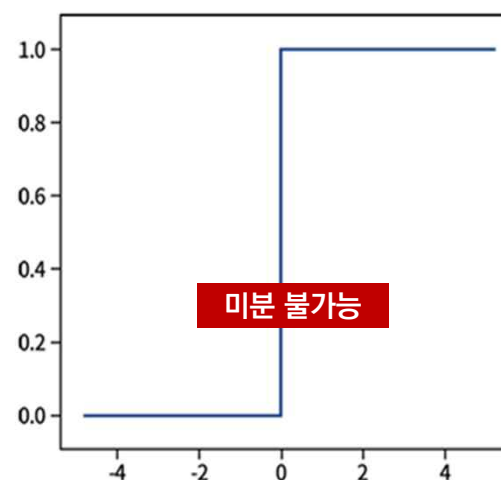
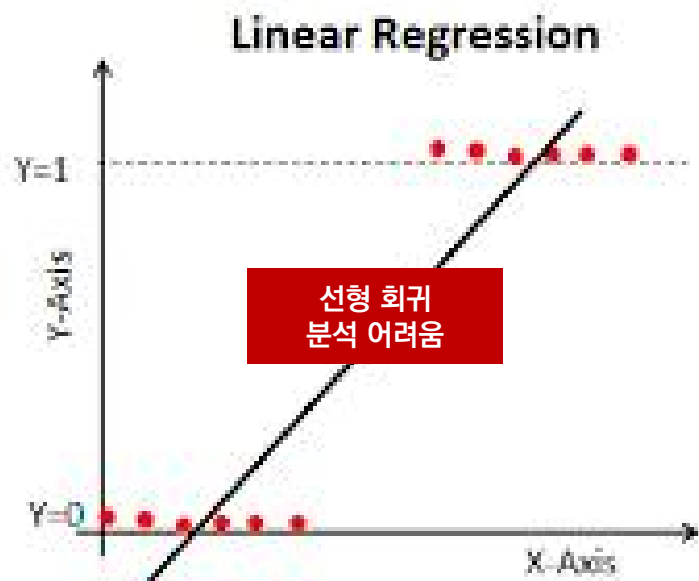
사례 1				사례 2				사례 3			
	테스트 샘플1	테스트 샘플2	테스트 샘플3		테스트 샘플1	테스트 샘플2	테스트 샘플3		테스트 샘플1	테스트 샘플2	테스트 샘플3
Model1	V	V	X	Model1	V	V	X	Model1	V	X	X
Model2	X	V	V	Model2	V	V	X	Model2	X	V	X
Model3	V	X	V	Model3	V	V	X	Model3	X	X	V
앙상블	V	V	V	앙상블	V	V	X	앙상블	X	X	X
◆ 단일모델 정확도 66% ◆ 앙상블 학습 결과 100% <b>앙상블 성능 향상</b>				◆ 단일모델 정확도 66% ◆ 앙상블 학습 결과 66% <b>앙상블 효과 없음</b>				◆ 단일모델 정확도 33% ◆ 앙상블 학습 결과 0% <b>앙상블 부작용</b>			

분류 모델

# 로지스틱 회귀

❖ Target 데이터가 이산적 형태를 보일 때 사용하는 회귀 모델

- 연속형 독립변수(X), 범주형 종속변수(Y)
- 예측 값  $> 0 \rightarrow 1$ 로 분류, 예측 값  $< 0 \rightarrow 0$ 으로 분류

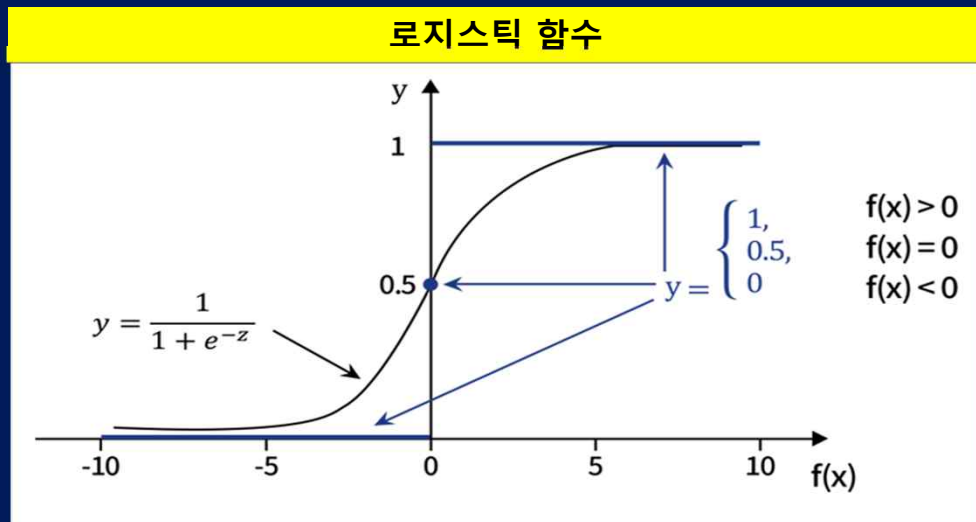


분류 모델

# 로지스틱 회귀

## ❖ 로지스틱 함수

■ 입력 값을 0이나 1에 근사한 출력 값으로 변환



$$y = \frac{1}{1 + e^{-f(x)}}$$

선형 회귀 분석 대입

$$= \frac{1}{1 + e^{-(w^T x + b)}}$$

$$h \frac{y}{1-y} = w^T x + b$$

$y$	$f(x)$ 가 양의 값일 가능성
$1 - y$	$f(x)$ 가 음의 값일 가능성
$\frac{y}{1-y}$	$f(x)$ 가 양의 값일 상대적 가능성

→ 오즈비  
(Odds Ratio)

분류 모델

# 로지스틱 회귀

$$\diamond y = \frac{1}{1+e^{-f(x)}}$$

$$\diamond y^{-1} = 1 + e^{-f(x)}$$

$$\diamond y^{-1} - 1 = e^{-f(x)} = \frac{1-y}{y}$$

$$\diamond e^{f(x)} = \frac{y}{1-y}$$

$$\diamond f(x) = \mathbf{h} \left( \frac{y}{1-y} \right)$$



❖ Day4. Wrap-up 설문

■ <https://forms.gle/CL6ZfcKNZM2f2ssS8>

(참고)하이퍼 파라미터 최적화

---

# 하이퍼 파라미터 (Hyper Parameter) 최적화

❖ 머신러닝 모델에서 우리가 직접 설정할 수 있는 값들 → 머신 러닝의 주요 구성요소

■ 하이퍼 파라미터 값을 조정하여 머신러닝 알고리즘의 성능 개선 → 하이퍼 파라미터 최적화

	파라미터 (Parameter)	하이퍼 파라미터 (Hyper Parameter)
의미	<ul style="list-style-type: none"><li>매개변수</li><li>모델 내부에서 결정되는 값</li><li>데이터로부터 학습하여 결정</li></ul>	<ul style="list-style-type: none"><li>초매개변수</li><li>모델 학습에 반영되는 값</li><li>학습 전에 미리 설정</li></ul>
예시	선형회귀 계수	비용함수의 종류, 학습률, 의사결정나무의 최대 깊이(max-depth)
설정 가능 여부	직접 설정 안됨 (X)	직접 설정 가능 (O)