

# 훈련 세트와 테스트 세트

비타민 2주차 세션1

10기 교육부 이소현, 유근태

# 01

## 지난 시간 내용 복습

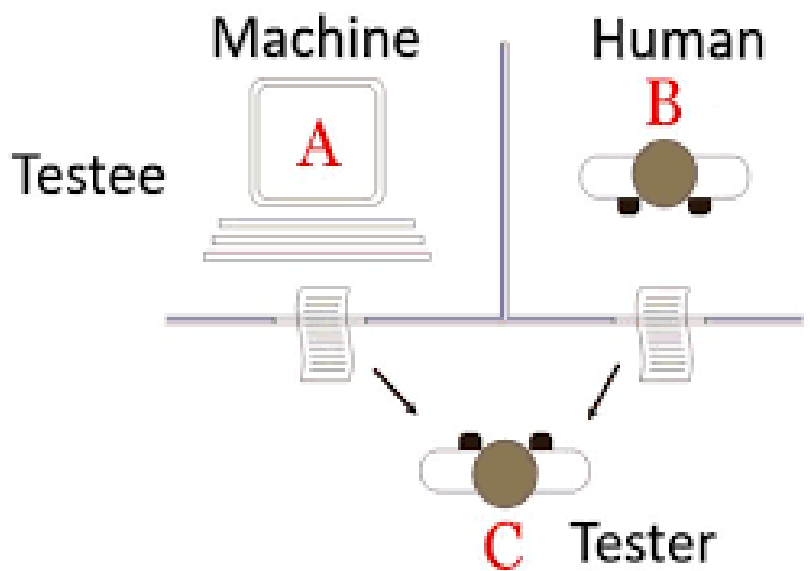
- 인공지능, 머신러닝, 딥러닝
- 데이터 분석을 위한 3종 패키지
- 복습 과제 풀이 예시

# 1. 인공지능, 머신러닝, 딥러닝

## Turing Test



기계에도 지능이 있는가?



기계에 지능이 있다고 말하기 위해서는,  
질문자가 상대와 여러 질의응답을 한 후, 어느 쪽이  
사람 or 컴퓨터인지 판별 불가능한 경우

## » 강인공지능(Strong AI)

인간과 동일한 지능이 구현된 인공지능  
사람과 구별이 어려운 지능을 가진 컴퓨터 시스템

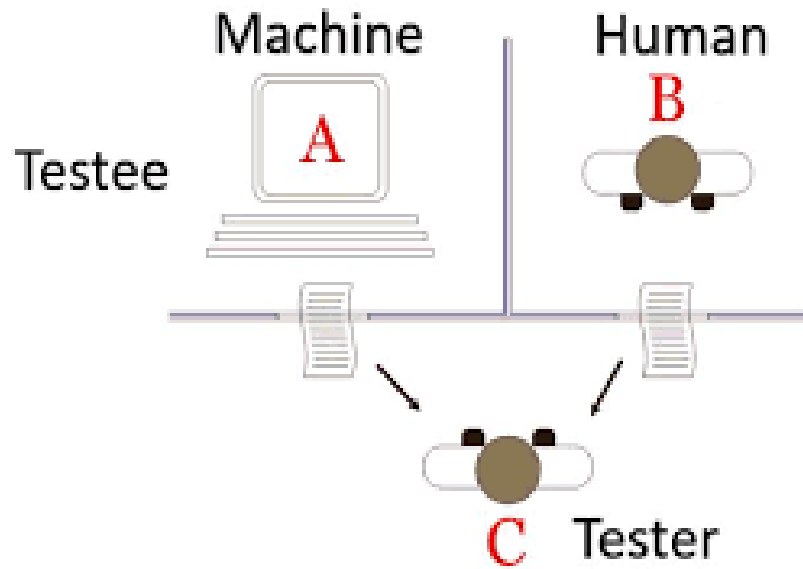
1950년 Alan Turing이 제시한 인공지능 판별법.  
인공지능(AI)의 우수성을 측정하기 위해 사용.

# 1. 인공지능, 머신러닝, 딥러닝

## Turing Test



기계에도 지능이 있는가?



기계에 지능이 있다고 말하기 위해서는,  
질문자가 상대와 여러 질의응답을 한 후, 어느 쪽이  
사람 or 컴퓨터인지 판별 불가능한 경우

## » 약인공지능 (Strong AI)

인간의 지능으로 가능한 작업의 일부를 컴퓨터로  
수행하게 하는 것.

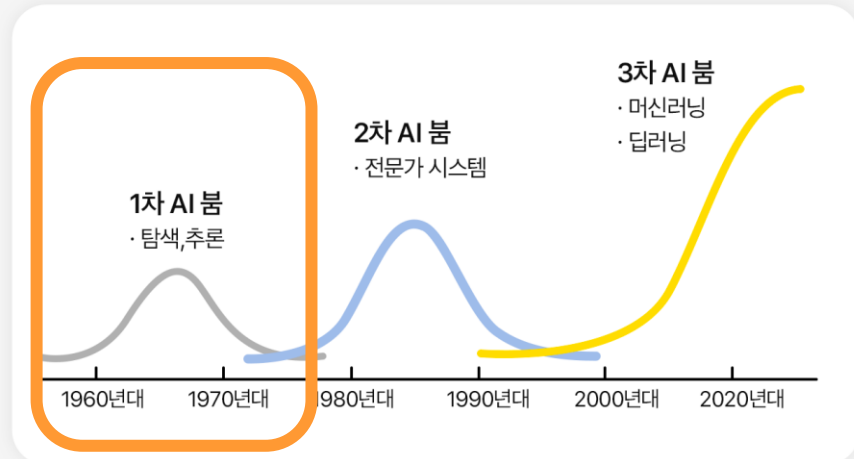
사람의 일을 도와주는 보조적인 역할 수행.

1950년 Alan Turing이 제시한 인공지능 판별법.  
인공지능(AI)의 우수성을 측정하기 위해 사용.

# 1. 인공지능, 머신러닝, 딥러닝

## 1차 AI 붐: 추론과 탐색

### AI 발전의 역사



## “지능이란 곧 기호 처리이다.”

컴퓨터상에서 기호 처리를 통해 퍼즐과 미로를 풀고 간단한 수학의 정리를 증명하거나 체스를 두는 등 지적 활동을 배울 수 있는 단계

Ex) 컴퓨터가 미로를 푸는 경우

1) 모든 길을 샅샅이 조사해 갈 수 있는 길의 패턴 **탐색**

+

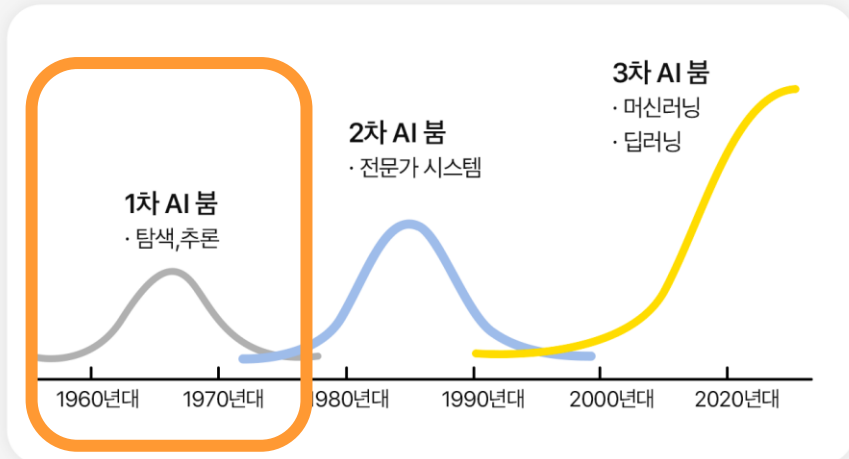
2) 목적지에 도착하는 경로를 예측하는 **추론**

⇒ AI는 “**추론과 탐색**”의 조합으로 미로를 풀 때, 가능한 모든 선택지를 탐색하여 목표에 도달하는 방법 찾아냄.

# 1. 인공지능, 머신러닝, 딥러닝

## 1차 AI 붐: 추론과 탐색

### AI 발전의 역사



“지능이란 곧 기호 처리이다.”

컴퓨터상에서 기호 처리를 통해 퍼즐과 미로를 풀고 간단한 수학의 정리를 증명하거나 체스를 두는 등 지적 활동을 배울 수 있는 단계

### \* 한계점

규칙과 목표가 명확하게 정해져 있는 문제에 적용이 한정됨.

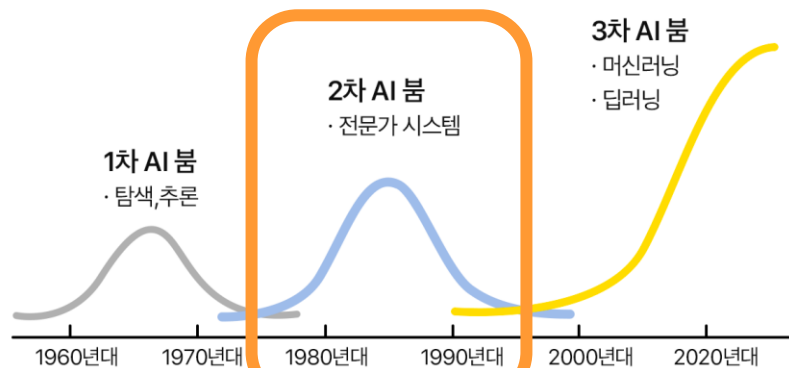
규칙과 목표가 복잡하고 모호한 실제 현실의 문제 해결 불가.

⇒ 계산을 효율화 하는 이론의 부족 및 컴퓨터 처리 능력의 부족

# 1. 인공지능, 머신러닝, 딥러닝

## 2차 AI 붐: 전문가 시스템

### AI 발전의 역사



## “지능이란 곧 지식이다.”

컴퓨터가 스스로 현실 세계에 대해 고려하기는 힘들다는 것이 판명되었으니, 인간이 컴퓨터를 가르쳐서 똑똑하게 만들자!

⇒ 전문가의 지식을 컴퓨터에 집어넣은 뒤, 그 지식에 기초하여 판단을 내리는 프로그램 설계

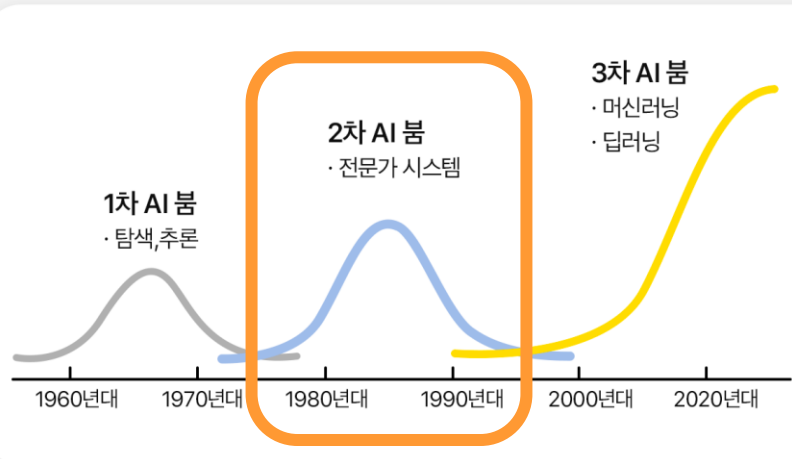
Ex) 의사가 환자를 진찰할 때, 여러가지 증상을 관찰 후 적절한 약물과 치료법을 판단함.

⇒ 이때, 의사가 사용하는 지식을 컴퓨터에 입력하여 컴퓨터가 마치 의사처럼 판단을 내리게 하는 것.

# 1. 인공지능, 머신러닝, 딥러닝

## 2차 AI 붐: 전문가 시스템

### AI 발전의 역사



## “지능이란 곧 지식이다.”

컴퓨터가 스스로 현실 세계에 대해 고려하기는 힘들다는 것이 판명되었으니, 인간이 컴퓨터를 가르쳐서 똑똑하게 만들자!

⇒ 전문가의 지식을 컴퓨터에 집어넣은 뒤, 그 지식에 기초하여 판단을 내리는 프로그램 설계

### \* 한계점

지식 병목 현상(Knowledge acquisition bottleneck)

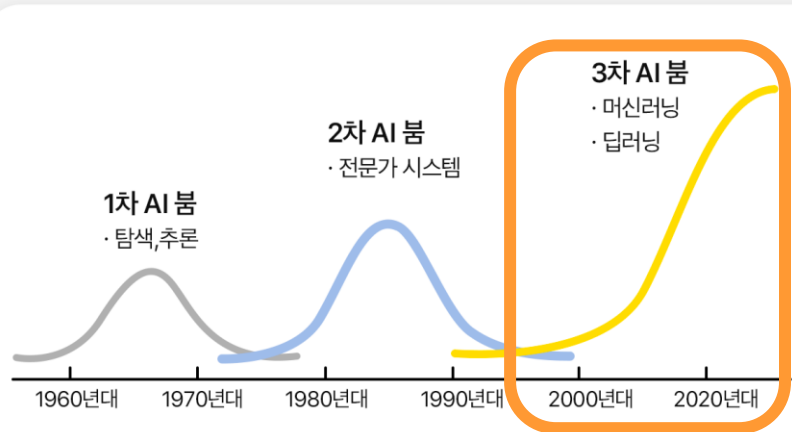
: 예를 들어, 생산 공장 현장에서는 단순한 지식보다 오랜 경험과 직감이 중요한 경우가 많음. 인간이 가진 전문 지식을 적절히 표현하여 시스템에 입력하는 것의 어려움. => 노력 대비 가치가 있는 것인가?



# 1. 인공지능, 머신러닝, 딥러닝

## 3차 AI 붐: 머신러닝, 딥러닝

### AI 발전의 역사



“지능이란 곧 학습이다.”

- 머신러닝의 방법 중 하나인 딥러닝은, 머신러닝의 정밀도를 비약적으로 증가시킴.
- 딥러닝은 인간이 존재하지 않아도 입력 데이터에서 스스로 특징을 판별하고, 특정 지식이나 패턴을 기억시키지 않아도 학습이 가능함.

Ex) IBM의 Watson, Google의 AlphaGo,

Facebook의 DeepFace

## 2. 데이터 분석 3종 패키지

### Numpy

```
import numpy as np
```

파이썬에서 배열을 사용하기 위한 표준 패키지.

수치 해석용 파이썬 패키지로, 벡터/행렬 사용하는 선형대수 계산에 주로 사용.

### Pandas

```
import pandas as pd
```

고수준의 자료 구조와 빠르고 쉬운 데이터 분석 도구를 제공하는 파이썬 라이브러리.

Pandas의 자료 구조에는 series, dataframe이 있음.

### matplotlib

```
import matplotlib.pyplot as plt
```

파이썬에서 자료를 차트나 플롯으로 시각화하는 패키지

# 3. 복습 과제 풀이 예시

## 문제 1

place라는 새로운 데이터 프레임 생성 후, place 데이터를 출력하세요. 이 데이터 프레임은 지역, 인구라는 두 열을 가지고 있습니다.

- 지역이 서울이면 인구는 9904312
- 지역이 부산이면 인구는 3448737
- 지역이 인천이면 인구는 2890451
- 지역이 대구이면 인구는 2466052

```
import pandas as pd
place=pd.DataFrame({'지역': ['서울', '부산', '인천', '대구'], '인구':['9904312', '3448737', '2890451', '2466052']})
place
```



## Dataframe 생성

### 1) 하나의 열 데이터를 일차원 배열(리스트)로 준비

지역 열에 해당하는 데이터: ['서울', '부산', '인천', '대구']

인구 열에 해당하는 데이터: ['9904312', '3448737', '2890451', '2466052']

### 2) 각 열에 대한 이름을 키로 가지는 딕셔너리 생성

{'지역': ['서울', '부산', '인천', '대구'], '인구': ['9904312', '3448737', '2890451', '2466052']}

### 3) dataframe 생성자에 데이터 넣기

pd.DataFrame({'지역': ['서울', '부산', '인천', '대구'], '인구': ['9904312', '3448737', '2890451', '2466052']})

# 02

## 훈련 세트와 테스트 세트

- 지도 학습, 비지도 학습
- 지도 학습에서의 훈련 세트 & 테스트 세트

# 1. 지도 학습, 비지도 학습

## 머신러닝

인공지능의 한 분야로, 컴퓨터가 스스로 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타깃을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

### 2) 비지도 학습(unsupervised learning)

- 정답(target)없이 입력 데이터만 사용
- 정답을 사용하지 않기 때문에, 무언가를 맞출 수는 없음.
- 데이터를 잘 파악하거나 변형하는데 도움.

이상값 감지  
(Anomaly  
Detection)

그룹화  
(clustering)

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.



분류  
(classification)

예측  
(regression)

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

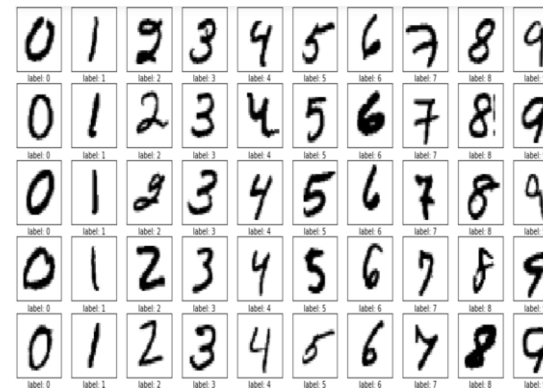
- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

### \* 분류란?

레이블이 달린 학습 데이터로 학습한 후에 새로 입력된 데이터가 학습했던 레이블 중 어떤 그룹에 속하는 지를 찾아냄.

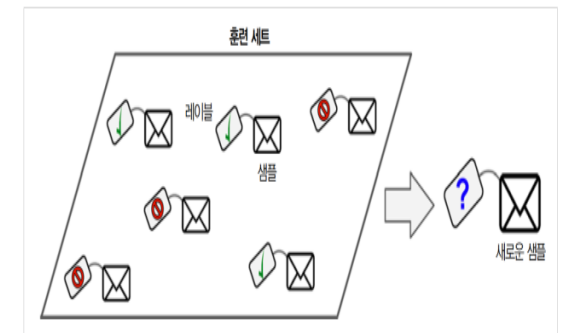
### \* 예시

#### 손글씨 이미지 분류



#### 스팸 메일 분류

그림 1-5 지도 학습에서 레이블된 훈련 세트(예를 들면 스팸 분류)



# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

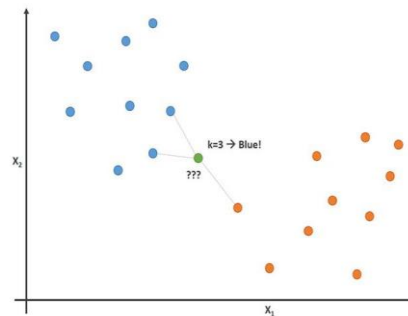
\* 분류란?

레이블이 달린 학습 데이터로 학습한 후에 새로 입력된 데이터가 학습했던 레이블 중 어떤 그룹에 속하는 지를 찾아냄.

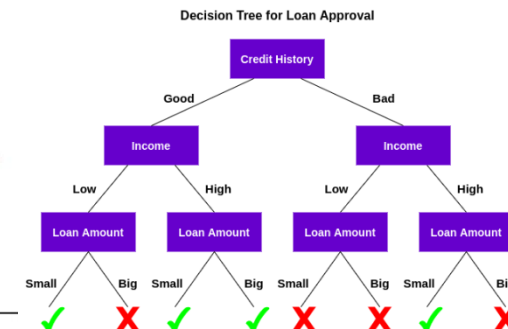
\* 알고리즘



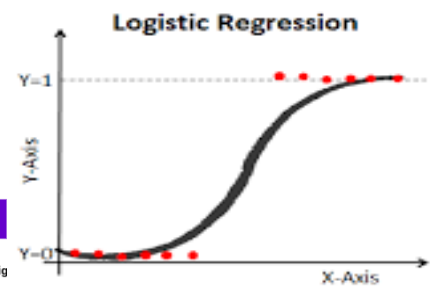
KNN



Decision Tree



Logistic Regression





# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

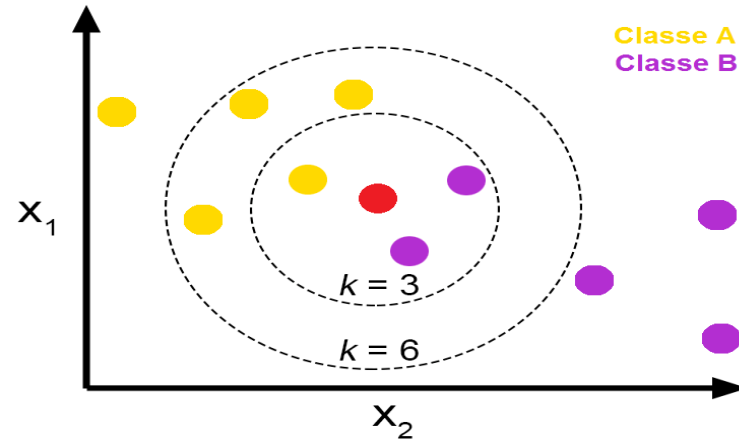
### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

\* 알고리즘: KNN(K-최근접 이웃: Chapter 3)



- 어떤 데이터가 주어지면 그 주변의 데이터를 살펴본 뒤 더 많은 데이터가 포함되는 범주로 분류
- K값에 따라 살펴보는 주변의 데이터 수 달라짐.
- 위의 그림에서, 빨간 점을 분류하고자 할 때, k=3인 경우 보라색 점(class B), k=6인 경우 노란색 점(class A)로 분류됨.

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

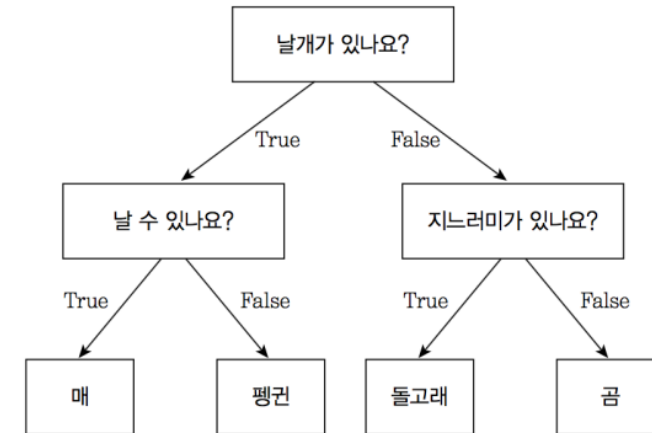
### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

\* 알고리즘: Decision Tree(의사결정트리: Chapter5)



- 예/아니오로 대답할 수 있는 질문을 이어 나가며 학습
- 특정 기준(질문)에 따라 데이터를 구분하는 모델
- 불순도(ex. gini, entropy)를 기준으로 정보 이득(information gain)이 최대가 되도록 노드를 분할

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

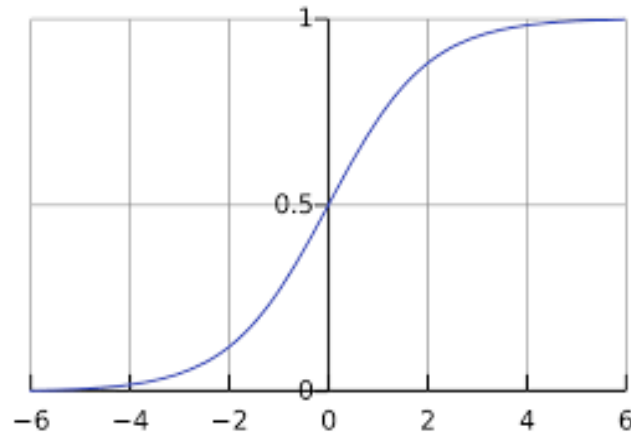
### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

\* 알고리즘: Logistic regression(로지스틱 회귀: Chapter4)



$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

- 시그모이드 함수를 사용해 함수가 0과 1사이의 값을 가지도록 함.
- Z가 무한하게 큰 음수일 경우 0에, 무한하게 큰 양수일 경우 1에 가까워짐.
- 이진 분류의 경우, Sigmoid 함수의 출력이 0.5보다 크면 양성 클래스, 0.5보다 작으면 음성 클래스

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

### \* 예측이란?

레이블이 달린 학습 데이터로 학습한 후에 특징(feature)과 레이블 사이의 상관 관계가 함수식으로 나타남.

⇒ 연속적인 범위 내의 값에서 그 결과값을 예측하는 문제에 활용.

### \* 예시

주가 예측



환율 분석



# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

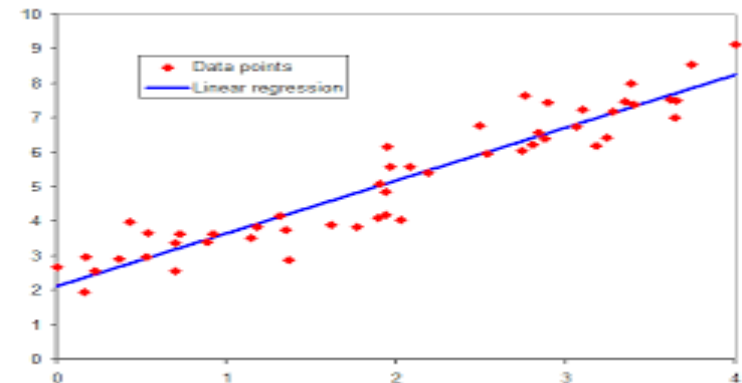
### \* 예측이란?

레이블이 달린 학습 데이터로 학습한 후에 특징(feature)과 레이블 사이의 상관 관계가 함수식으로 나타남.

⇒ 연속적인 범위 내의 값에서 그 결과값을 예측하는 문제에 활용.

### \* 알고리즘

Linear Regression(선형 회귀)



# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

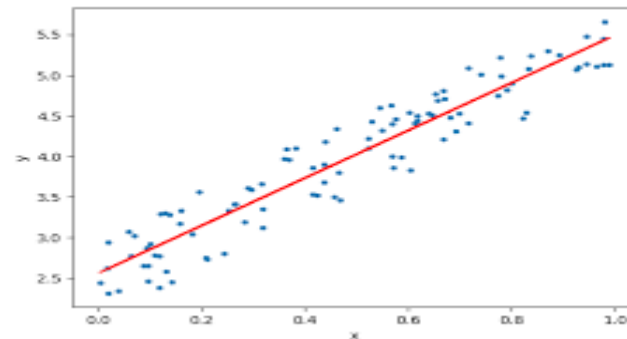
### 1) 지도 학습(supervised learning)

- 훈련하기 위한 데이터(input)와 정답(target)이 필요  
=> 입력과 타겟을 합쳐 훈련 데이터(training data)라고 함.
- 정답이 있기 때문에 알고리즘이 정답을 맞히는 것을 학습함.

분류  
(classification)

예측  
(regression)

\* 알고리즘: Linear Regression(선형 회귀, Chapter3)



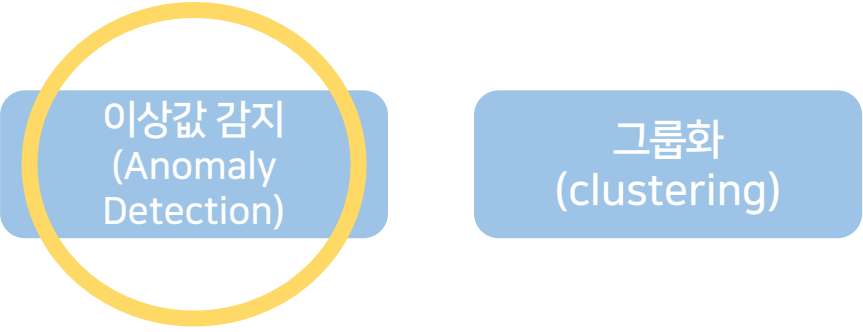
- 데이터를 가장 잘 설명할 수 있는 직선식 ( $y=mx+b$ ) 찾기  
⇒ 모든 데이터로부터 나타나는 오차의 평균을 최소화할 수 있는 최적의 기울기와 절편 찾기

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 2) 비지도 학습(unsupervised learning)

- 정답(target)없이 입력 데이터만 사용
- 정답을 사용하지 않기 때문에, 무언가를 맞출 수는 없음.
- 데이터를 잘 파악하거나 변형하는데 도움.



이상값 감지  
(Anomaly  
Detection)

그룹화  
(clustering)

#### \* 이상값 감지란?

정상 데이터를 학습하여 비정상 데이터를 감지함.

#### \* 예시

제조 라인에서 결함 제품 감지

#### \* 알고리즘

GMM, PCA, isolation forest



# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

### 2) 비지도 학습(unsupervised learning)

- 정답(target)없이 입력 데이터만 사용
- 정답을 사용하지 않기 때문에, 무언가를 맞출 수는 없음.
- 데이터를 잘 파악하거나 변형하는데 도움.

이상값 감지  
(Anomaly  
Detection)

그룹화  
(clustering)

#### \* 그룹화란?

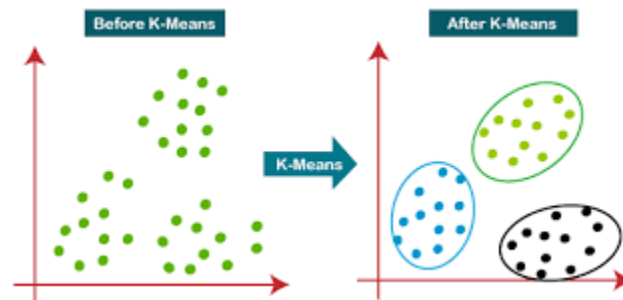
라벨링이 되어 있지 않은 데이터들 내에서 비슷한 특징이나 패턴을 가진 데이터들끼리 그룹화한 후, 새로운 데이터가 어떤 그룹에 속하는지 판단.

#### \* 예시

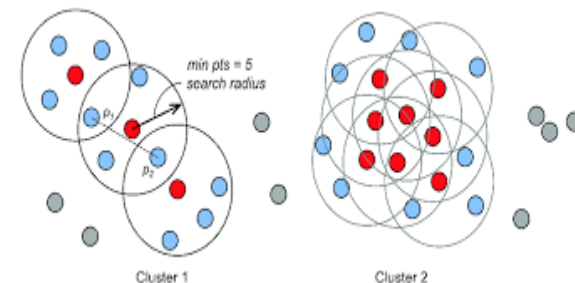
카드사에서 고객 정보와 카드 결제 내역을 바탕으로 고객 유형 분류

#### \* 알고리즘

##### K- means clustering



##### DBSCAN clustering



# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

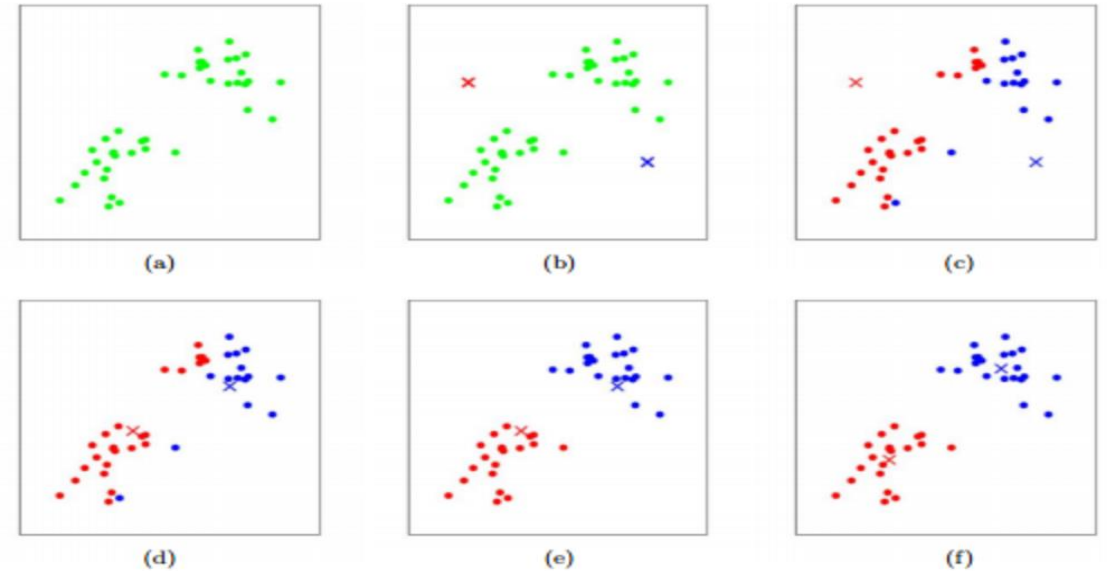
### 2) 비지도 학습(unsupervised learning)

- 정답(target)없이 입력 데이터만 사용
- 정답을 사용하지 않기 때문에, 무언가를 맞출 수는 없음.
- 데이터를 잘 파악하거나 변형하는데 도움.

이상값 감지  
(Anomaly  
Detection)

그룹화  
(clustering)

\* 알고리즘: K-means clustering(K-평균, Chapter6)



- 각 데이터로부터 그 데이터가 속한 클러스터 중심까지의 평균 거리 최소화
- 1) K개의 임의의 중심점(centroid) 배치
  - 2) 각 데이터를 가장 가까운 중심점으로 할당
  - 3) 군집에 속한 샘플의 평균값으로 클러스터의 중심점 업데이트
  - 4) 2,3번 단계를 수렴이 될 때까지(중심점이 업데이트 되지 않을 때까지) 반복

# 1. 지도 학습, 비지도 학습

## 머신러닝으로 해결할 수 있는 문제

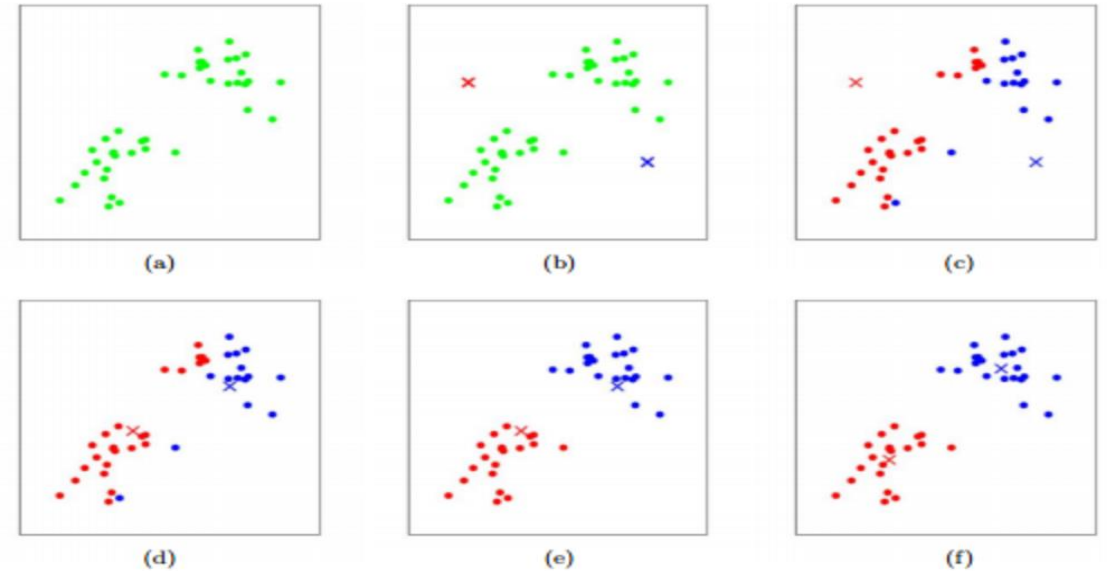
### 2) 비지도 학습(unsupervised learning)

- 정답(target)없이 입력 데이터만 사용
- 정답을 사용하지 않기 때문에, 무언가를 맞출 수는 없음.
- 데이터를 잘 파악하거나 변형하는데 도움.

이상값 감지  
(Anomaly  
Detection)

그룹화  
(clustering)

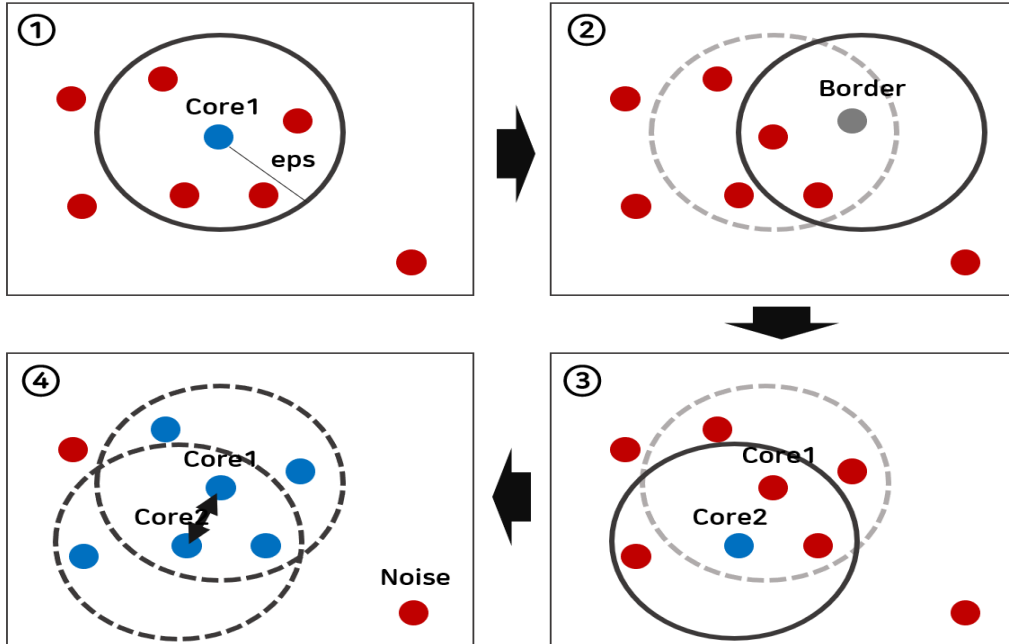
\* 알고리즘: K-means clustering(K-평균, Chapter6)



- 각 데이터로부터 그 데이터가 속한 클러스터 중심까지의 평균 거리 최소화
- 1) K개의 임의의 중심점(centroid) 배치
  - 2) 각 데이터를 가장 가까운 중심점으로 할당
  - 3) 군집에 속한 샘플의 평균값으로 클러스터의 중심점 업데이트
  - 4) 2,3번 단계를 수렴이 될 때까지(중심점이 업데이트 되지 않을 때까지) 반복

# 1. 지도 학습, 비지도 학습

\* 알고리즘: DBSCAN clustering(교재에 없음)



minPts(반경 내 최소 point 수), Core: 중심점(minPts를 만족할 경우),  
Border: 경계점(minPts를 만족하진 않지만, 어느 core 반경에 속한 경우),  
Noise: 어느 군집에도 속하지 않은 점

- 1) 하나의 점(파란색)을 중심으로 반경 내에 최소 점이 4개(minPts=4)이상 있으면 하나의 군집으로 판단하고, 해당 점(파란색)은 core가 된다.
- 2) 회색 점은 반경 내에 점이 3개 뿐이므로 core가 되지는 못하지만, core1의 군집에 포함된 점이므로 border가 된다.
- 3) 파란색 점은 반경 내에 최소 점이 4개 이상이기 때문에 core가 된다.
- 4) 그러나, 반경 내의 점 중에 core1이 포함되어 있으므로 두 군집은 연결되어 하나의 군집으로 묶인다.

=> 이와 같은 방식으로 군집의 확산을 반복하면서, 자동으로 최적의 군집수가 도출.

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

알고리즘의 성능을 제대로 평가하려면 훈련 데이터와 평가에 사용할 데이터가 달라야 함.

### 1) 훈련 세트(train set)

- 모델을 훈련할 때 사용하는 데이터(입력 + 정답(target, label))
- 훈련 세트는 클수록 좋음.

### 2) 테스트 세트(test set)

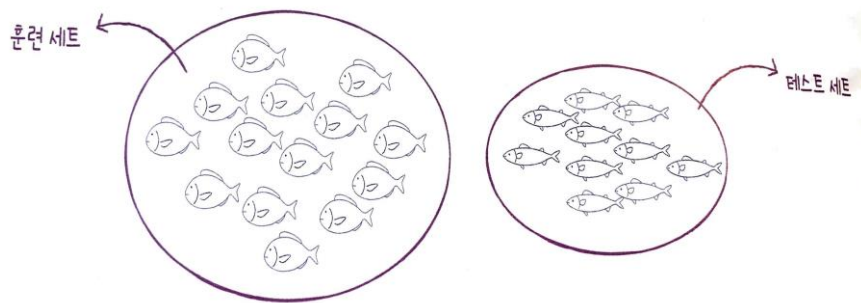
- 훈련 세트로 학습시킨 모델의 성능 평가에 사용되는 데이터
- 주로 훈련 데이터에서 일부를 떼어 내어 테스트 세트로 활용.

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

### ★ 훈련 세트 & 테스트 세트를 나눌 때 주의 사항

- 1) 훈련 세트의 데이터가 테스트 세트의 데이터보다 많아야 함.
- 2) 훈련 세트와 테스트 세트에 중복되는 데이터가 최대한 없어야 함.
- 3) 샘플링 편향(sampling bias)

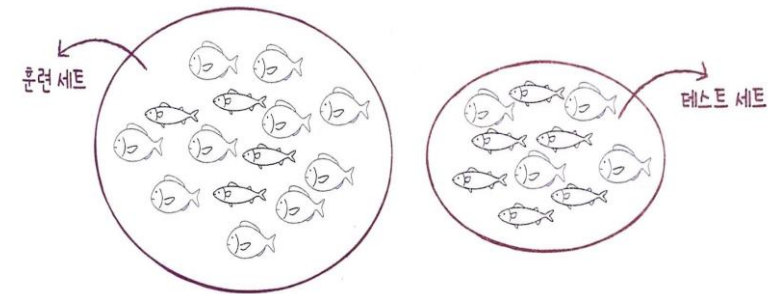
잘못된 훈련 데이터



- 훈련 세트에는 도미 데이터만, 테스트 세트에는 빙어 데이터만 들어있음.
- 빙어 데이터 없이 모델을 훈련하면 빙어 올바르게 분류 불가.

=> 샘플링 편향

올바른 훈련 데이터



- 훈련 세트에 도미와 빙어 데이터가 골고루 섞여 있음.
- 도미와 빙어 모두 올바르게 분류 가능

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

### 훈련 세트와 테스트 세트로 나누는 법

#### 1) Numpy 배열의 인덱스 섞기

```
[ ] 1 # 도미 데이터
    2 bream_length = [25.4, 26.3, 26.5, 29.0, 29.0, 29.7, 30.0, 30.0, 30.7, 31.0, 31.0, 31.5, 32.0, 32.0]
    3 bream_weight = [242.0, 290.0, 340.0, 363.0, 430.0, 450.0, 500.0, 390.0, 450.0, 500.0, 475.0, 500.0, 340.0, 600.0]
    4
    5 # 빙어 데이터
    6 smelt_length = [9.8, 10.5, 10.6, 11.0, 11.2, 11.3, 11.8, 11.8, 12.0, 12.2, 12.4, 13.0, 14.3, 15.0]
    7 smelt_weight = [6.7, 7.5, 7.0, 9.7, 9.8, 8.7, 10.0, 9.9, 9.8, 12.2, 13.4, 12.2, 19.7, 19.9]
    8
    9 # 도미와 빙어 데이터를 하나의 데이터로 합침
   10 length = bream_length + smelt_length
   11 weight = bream_weight + smelt_weight
   12
   13 fish_data = [[l,w] for l,w in zip(length, weight)]
   14 fish_target = [1]*15 + [0]*14
```

- 도미 데이터 15개, 빙어 데이터 14개 선언.
- 도미 데이터와 빙어 데이터를 각각 length와 weight로 합침.
- zip 함수를 사용해 생선의 길이와 무게를 하나의 리스트로 담은 2차원 리스트 fish\_data를 만들고, 이에 해당하는 정답값 fish\_target을 생성.

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

### 훈련 세트와 테스트 세트로 나누는 법

#### 1) Numpy 배열의 인덱스 섞기

```
[ ] 1 # 넘파이 사용하기
      2 import numpy as np
      3
      4 input_arr = np.array(fish_data)
      5 target_arr = np.array(fish_target)
      6
      7 print(input_arr)
```

#### 출력 결과

```
[[ 25.4 242. ]
 [ 26.3 290. ]
 [ 26.5 340. ]
 [ 29. 363. ]
 [ 29. 430. ]
 [ 29.7 450. ]
 [ 30. 500. ]
 [ 30. 390. ]
 [ 30.7 450. ]
 [ 31. 500. ]
 [ 31. 475. ]
 [ 31.5 500. ]
 [ 32. 500. ]]
```

- 넘파이 array() 함수에 파이썬 리스트를 전달하여, 파이썬 리스트를 넘파이 배열로 바꿈.



## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

### 훈련 세트와 테스트 세트로 나누는 법

#### 1) Numpy 배열의 인덱스 섞기

```
[ ] 1 # 데이터 섞기
    2 index = np.arange(29) #인덱스 생성
    3 np.random.shuffle(index)
    4 print(index)
    5
    6 train_input = input_arr[index[:15]]
    7 train_target = target_arr[index[:15]]
    8 test_input = input_arr[index[15:]]
    9 test_target = target_arr[index[15:]]
```

- 넘파이 arrange() 함수를 사용해 0부터 28까지 1씩 증가하는 인덱스 만듦.
- shuffle() 함수를 사용해 주어진 배열을 무작위로 섞음.
- 앞서 만든 인덱스 배열의 처음 15개를 train\_input과 train\_target에 전달하여 랜덤하게 15개의 샘플을 훈련 세트로 만듦.
- 나머지 14개의 데이터를 테스트 세트로 만듦.

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

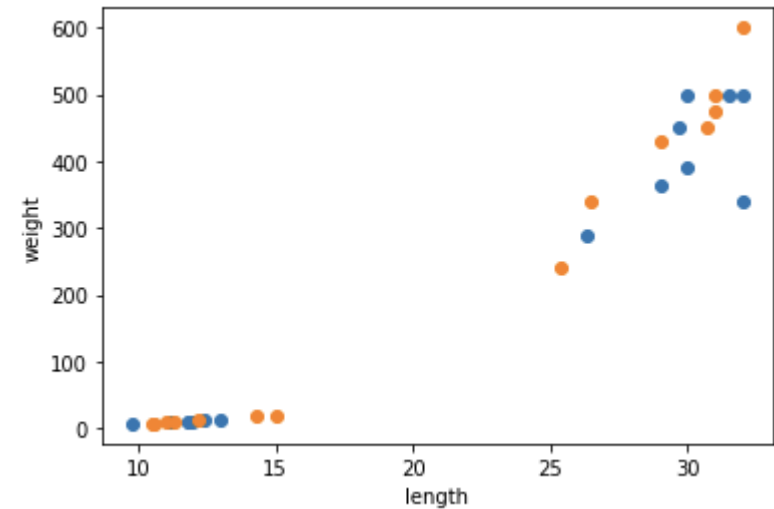
### 훈련 세트와 테스트 세트로 나누는 법

#### 1) Numpy 배열의 인덱스 섞기

```
[ ] 1 # 데이터 나누고 확인하기
     2 import matplotlib.pyplot as plt
     3
     4 plt.scatter(train_input[:,0], train_input[:,1])
     5 plt.scatter(test_input[:,0], test_input[:,1])
     6 plt.xlabel('length')
     7 plt.ylabel('weight')
     8 plt.show()
```

- 마지막으로 훈련 세트와 테스트 세트에 도미와 빙어가 잘 섞여 있는지 확인!

출력 결과



파란색이 훈련 세트이고 주황색이 테스트 세트

## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

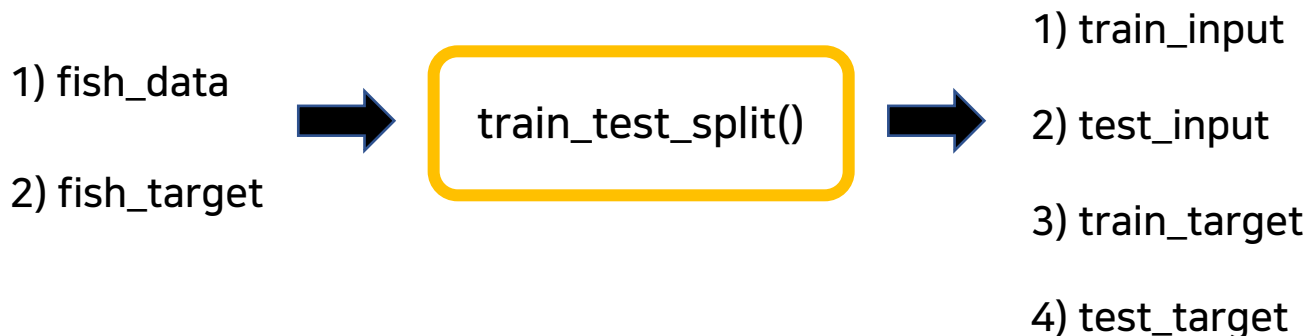
### 훈련 세트와 테스트 세트로 나누는 법

#### 2) 사이킷런 패키지의 train\_test\_split() 함수 이용

```
from sklearn.model_selection import train_test_split

train_input, test_input, train_target, test_target = train_test_split(fish_data, fish_target, stratify=fish_target, random_state=42)
```

- train\_test\_split() 함수는 사이킷런의 model\_selection 모듈 안에 있으므로 import.
- 위와 같이 훈련 세트와 테스트 세트로 나눔.
- fish\_data와 fish\_target이라는 두 개의 배열을 전달했으므로 2개씩 나뉘어 총 4개의 배열이 반환됨.



## 2. 지도 학습에서의 훈련 세트 & 테스트 세트

### 훈련 세트와 테스트 세트로 나누는 법

#### 2) 사이킷런 패키지의 train\_test\_split() 함수 이용

```
from sklearn.model_selection import train_test_split

train_input, test_input, train_target, test_target = train_test_split(fish_data, fish_target, stratify=fish_target, random_state=42)
```

- train\_test\_split() 함수는 사이킷런의 model\_selection 모듈 안에 있으므로 import
- 위와 같이 훈련 세트와 테스트 세트로 나눔.
- fish\_data와 fish\_target이라는 두 개의 배열을 전달했으므로 2개씩 나뉘어 총 4개의 배열이 반환됨.
- test size 매개 변수 지정 가능(기본값은 0.25) => 전체 데이터의 25%를 test set으로 지정
- stratify 매개변수를 target 데이터로 지정 시, 클래스 비율에 맞게 훈련 세트와 테스트 세트를 나눔.

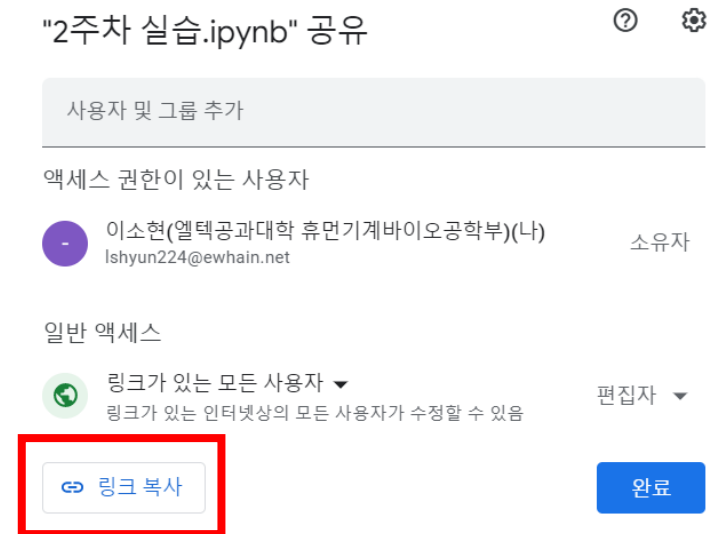
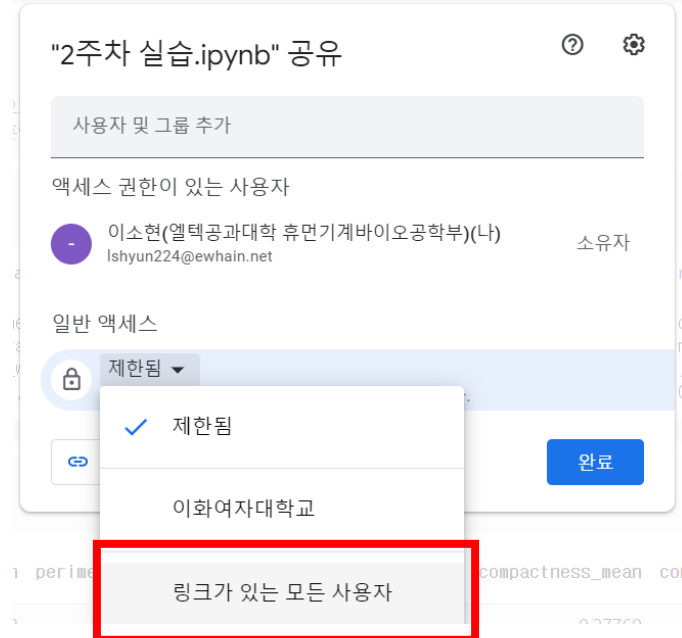
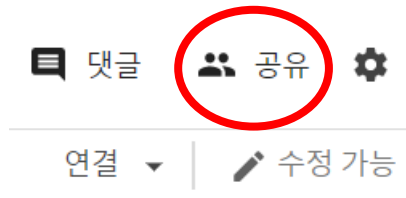


실습

**Using Google Colab**

# 실습

- 1) 카페에서 실습 파일 및 csv 파일 다운 받아 코랩에 세팅!
- 2) 코랩 오른쪽 상단에 있는 '공유' 누르고, '엑세스 권한 링크가 있는 모든 사용자/편집자'로 변경 후 링크 복사 누르기



# 실습

- 1) 카페에서 실습 코드 및 csv 파일 다운 받아 코랩에 세팅!
- 2) 코랩 오른쪽 상단에 있는 ‘공유’ 누르고, ‘엑세스 권한 링크가 있는 모든 사용자/편집자’로 변경 후 링크 복사 누르기
- 3) <https://forms.gle/KtVMEk6YverxSPkc9>  
=> 위의 구글폼 링크에 복사한 링크 붙여 넣어 제출

정답 여부를 보는 것이 아닌, 실습에 잘 참여하고 계신지 확인하기 위한 용도이므로 답을 맞춰야 한다는 부담 없이 편하게 제출해주시면 좋을 것 같습니다:)