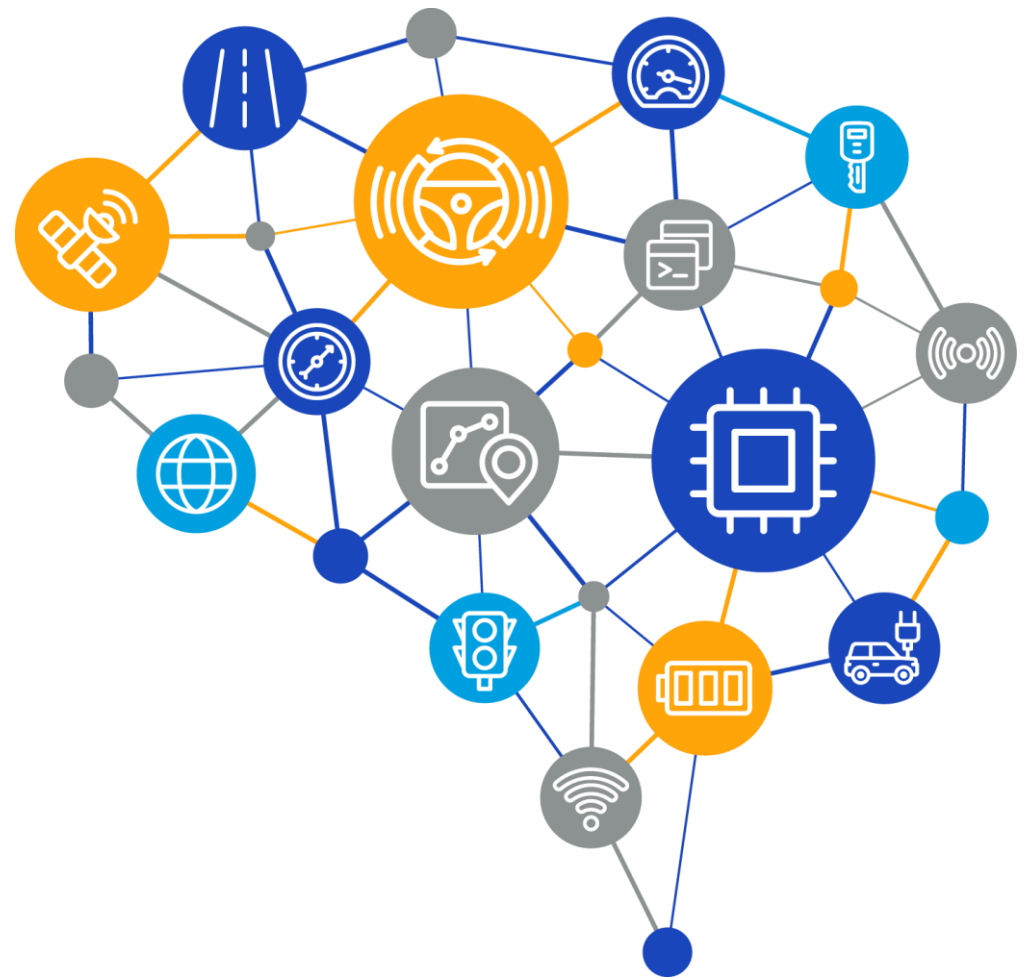


머신러닝 지도학습

Machine Learning
Supervisor learning

2022.05

강환수 교수



AI Experts Who Lead The Future

CONTENTS

- 01 | 머신러닝이란?
- 02 | 머신러닝 학습 방법
- 03 | 지도학습의 분류 알고리즘
- 04 | 지도학습의 회귀 알고리즘
- 05 | 라
- 06 | 동

AI Experts
Who Lead
The Future

01

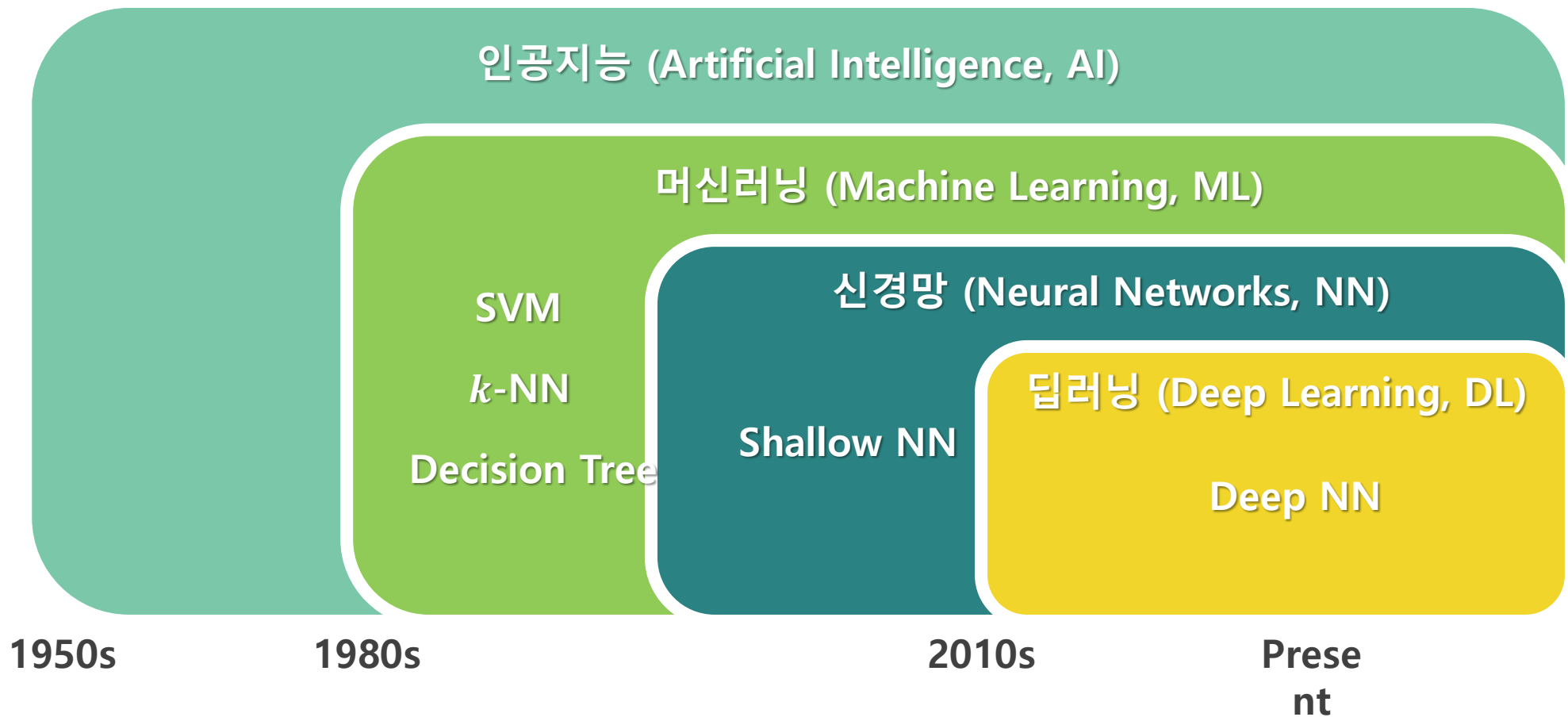
머신러닝이란?

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

01. 머신러닝이란?

인공지능 활용 Python language

- 인공지능, 머신러닝, 딥러닝의 관계



01. 머신러닝이란?

인공지능 활용 Python language

• 머신러닝의 개념 (1/5)

- 머신러닝 (Machine Learning, ML)

- 컴퓨터가 명시적으로 프로그램 되지 않고도 학습할 수 있도록 하는 연구 분야를 말함
- 용어는 1959년에 아서 사무엘이 학술지 <IBM Journal of Research and Development>에 기고한 논문에서 처음 사용함



[사진출처] https://en.wikipedia.org/wiki/Arthur_Samuel

아서 사무엘의 머신러닝 정의

“머신러닝은 컴퓨터가 명시적으로 프로그램되지 않고도 학습할 수 있도록 하는 연구 분야를 말합니다.”

(Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.)

01. 머신러닝이란?

인공지능 활용 Python language

- 머신러닝의 개념 (2/5)
 - 일반적인 프로그래밍 작업
 - 입력값에 따라 원하는 결과값이 출력되도록 사람이 내부 동작을 작성
 - 머신러닝
 - 사람이 컴퓨터에게 입력값과 결과값만 충분히 전달해 주면
컴퓨터가 스스로 입력값과 결과값의 관계를 만족시키는 내부 동작을 찾아냄
 - 학습 데이터 (훈련 데이터, Training Data): 내부 동작을 만들 때 사용한 데이터 (입력값과 결과값)
 - 시험 데이터 (Test Data): 만들어진 내부 동작의 성능을 평가할 때 사용하는 데이터

01. 머신러닝이란?

머신러닝의 개념 (3/5)

- 머신러닝은 여러 개의 입력값과 결과값을 컴퓨터에 제공하기만 하면 이 데이터를 바탕으로 컴퓨터가 스스로 내부 동작을 만들어 냄
- 이를 위해서 사람은 양질의 많은 학습 데이터를 공급해야 함

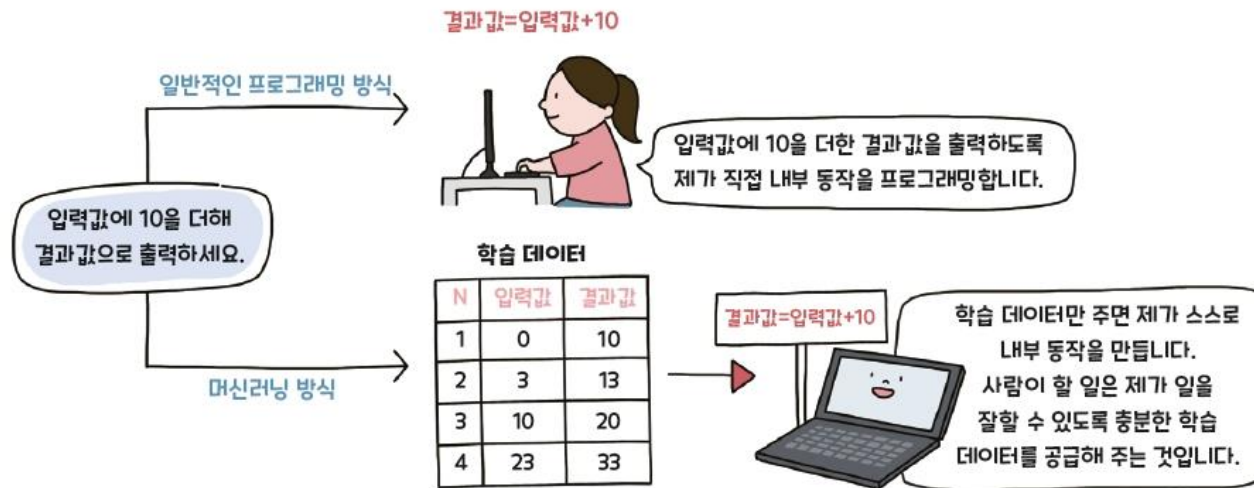


그림 2-2 일반적인 프로그래밍 방식과 머신러닝 방식 비교 © Hanbit Academy Inc.

01. 머신러닝이란?

인공지능 활용 Python language

- 머신러닝의 개념 (4/5)
 - 인공지능 시대의 소프트웨어 역량
 - 머신러닝 방식에서 우수한 개발자란 데이터에 대한 통찰력과 수집 능력이 우수하여 머신러닝 기계에 양질의 학습 데이터를 충분히 공급할 수 있는 사람임
 - 일반적인 프로그래밍 방식에서 우수한 개발자란 논리적인 사고를 통해 프로그래밍 언어를 자유자재로 사용할 수 있는 사람이었음
 - 앞으로는 쓸모 있는 데이터를 누가 더 많이 보유하고 있느냐에 따라 소프트웨어 역량이 결정 될 것

01. 머신러닝이란?

머신러닝의 개념 (5/5)

- 카네기 멜론 대학교 교수인 톰 미첼은 머신러닝을 다음과 같이 정의함



"만약 컴퓨터 프로그래밍이 어떤 작업 T를 수행할 때, 경험 E를 통해 성능이 향상된다면(성능 측정 방법은 P), 그 컴퓨터 프로그램은 작업 T를 수행할 때 방법 P로 측정되는 성능 향상을 경험 E로부터 학습한다고 말할 수 있습니다."

(A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.)

그림 2-3 톰 미첼의 머신러닝 정의

© Hanbit Academy Inc.

- 만약 머신러닝 방식으로 고양이 판별 프로그램을 개발한다면, 고양이 판별 정확도를 높이기 위해 프로그램에 수많은 고양이 사진들을 입력해 훈련시켜야 함
- 이를 톰 미첼의 정의로 해석하면 [표 2-1]과 같음

표 2-1 고양이 판별 프로그램 예를 활용한 톰 미첼의 머신러닝 정의 해석

© Hanbit Academy Inc.

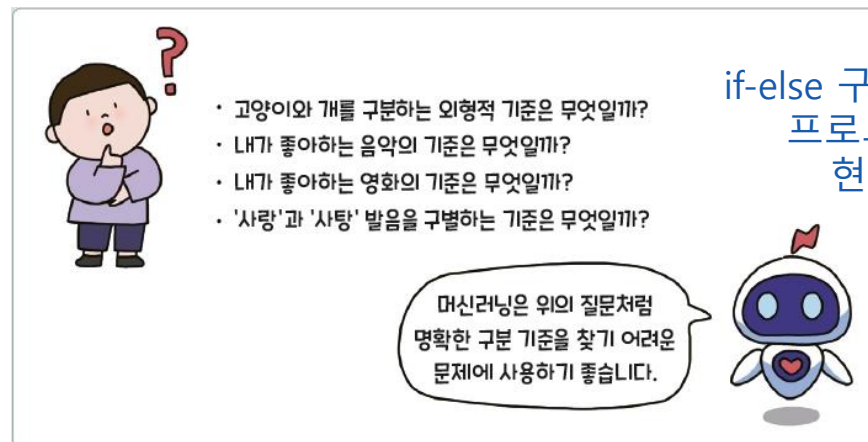
작업 T, 성능 측정 P, 경험 E	해석
<ul style="list-style-type: none"> • T : 고양이 판별하기 • P : 고양이 판별 정확도 • E : 고양이 사진 입력 	<p>〈고양이 판별하기〉작업을 수행할 때 '고양이 사진 입력'으로 '고양이 판별 정확도'가 향상된다면, 이 프로그램은 〈고양이 판별하기〉작업을 수행할 때 '고양이 판별 정확도'의 성능을 향상시키기 위해 '고양이 사진 입력'이란 경험으로부터 학습합니다.</p>

01. 머신러닝이란?

인공지능 활용 Python language

머신러닝은 언제 주로 사용될까? (1/2)

- 명시적으로 알고리즘을 설계하고 프로그래밍 하는 것이 어렵거나 불가능 한 경우에 주로 사용됨
 - ① 문제에서 나타날 수 있는 경우의 수가 너무 많은 경우
 - ② 규칙 기반 프로그램으로 답을 내기가 어려운 경우
- 예를 들어
 - ① 바둑 경기에서 모든 경우의 수를 찾아서 if-else와 같은 문장으로 처리하는 것은 거의 불가능함
 - ② 스팸 메일을 자동으로 걸러내는 작업에도 많은 경우가 있기 때문에 프로그래밍하는 것은 거의 불가능함



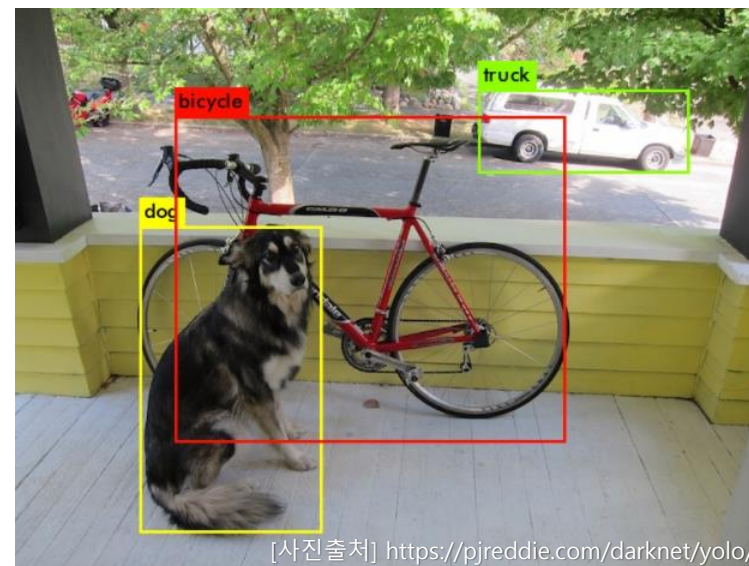
© Hanbit Academy Inc.

01. 머신러닝이란?

• 머신러닝은 언제 주로 사용될까? (2/2)

- 머신러닝의 응용 분야

- 주로, 복잡한 데이터들이 있고 이 데이터들에 기반하여 결정을 내려야 하는 분야
- 머신러닝 모델을 학습 (Learning) 시키려면 많은 데이터가 필요하기 때문에 머신러닝은 빅데이터 (Big Data)와 아주 밀접한 관계가 있음
- 예를 들어
 - 객체 검출 (Object Detection)
 - 음성 인식 (Voice Recognition)
 - 글자 인식 (Character Recognition)



01. 머신러닝이란?

머신러닝 구현 과정 예제 (1/4)

- 사람의 키를 입력했을 때 몸무게를 추측하는 작업을 머신러닝으로 구현하면?
- ① 일반적으로 키가 커지면 몸무게도 늘어날 것이라 가정해,
[몸무게 = $a \times \text{키} + b$]와 같은 직선의 방정식을 만들어 머신러닝 모델로 가정

$$\text{몸무게} = a \times \text{키} + b$$

$$y = a \times x + b$$



그림 2-5 머신러닝 구현 과정 예제의 학습 데이터

© Hanbit Academy Inc.

No.	이름	x	y
		키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

01. 머신러닝이란?

인공지능 활용 Python language

머신러닝 구현 과정 예제 (2/4)

- 사람의 키를 입력했을 때 몸무게를 추측하는 작업을 머신러닝으로 구현하면?
- ② 머신러닝 모델로 가정한 직선의 방정식은 아직 기울기 a 와 y 절편 b 의 값이 결정되지 않은 상태로, 학습 데이터를 확보하여 최적화된 직선을 구함
 - 학습 데이터와 최적화된 직선을 구한다는 것은 학습 데이터와 오차가 가장 적은 직선의 기울기 ($= a$)와 y 절편 ($= b$)을 구한다는 의미

No.	이름	x	y
		키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

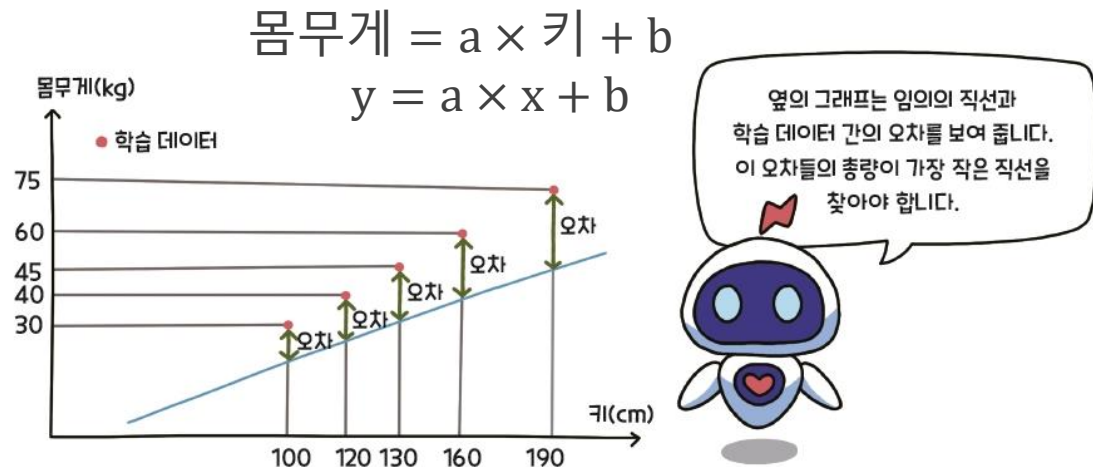


그림 2-6 학습 데이터와 직선 간의 오차 © Hanbit Academy Inc.

01. 머신러닝이란?

인공지능 활용 Python language

머신러닝 구현 과정 예제 (3/4)

- 사람의 키를 입력했을 때 몸무게를 추측하는 작업을 머신러닝으로 구현하면?

② 머신러닝 모델로 가정한 직선의 방정식은 아직 기울기 a 와 y 절편 b 의 값이 결정되지 않은 상태로, 학습 데이터를 확보하여 최적화된 직선을 구함

• 학습 결과, 직선의 방정식 [몸무게 = $(0.5 \times \text{키}) - 20$]이 해당 학습 데이터에 최적화된 함수임

No.	이름	x	y
		키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

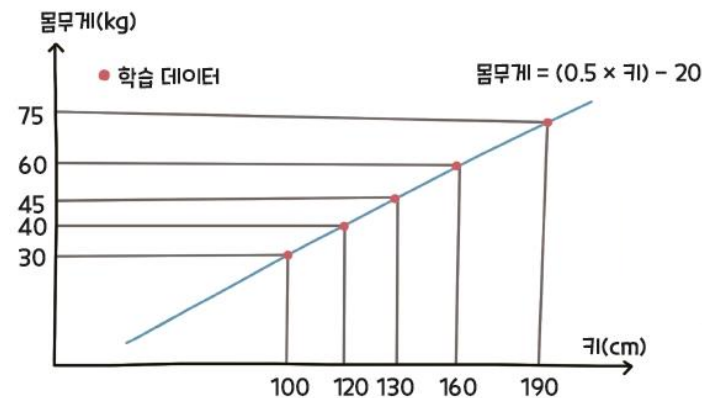


그림 2-7 몸무게와 키의 상관관계에 최적화된 직선의 방정식 © Hanbit Academy Inc.

$$\text{몸무게} = a \times \text{키} + b$$

$$y = a \times x + b$$

최적화 (Optimization)

$$a = 0.5$$

$$b = -20$$

$$y = 0.5 \times x - 20$$

01. 머신러닝이란?

인공지능 활용 Python language

머신러닝 구현 과정 예제 (4/4)

- 사람의 키를 입력했을 때 몸무게를 추측하는 작업을 머신러닝으로 구현하면?

③ 최적화된 머신러닝 모델을 실제 문제에 적용해 확인

- 학습 데이터에는 없던 키 180cm를 방정식에 대입하면 예측되는 몸무게는 $[70\text{kg} = (0.5 \times 180) - 20]$ 이 나옴

Q) 키가 180cm 인 사람은 몸무게가 몇 kg일까?

No.	이름	x	y
		키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

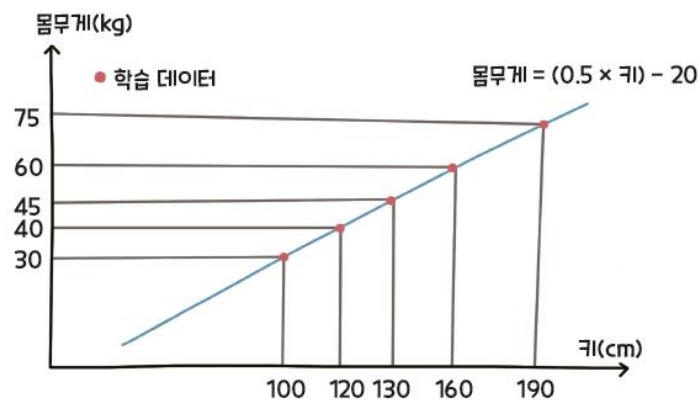


그림 2-7 몸무게와 키의 상관관계에 최적화된 직선의 방정식 © Hanbit Academy Inc.

$$y = 0.5 \times x - 20$$

$$\downarrow$$

$$x = 180 \text{ 대입}$$

$$\downarrow$$

$$y = 0.5 \times 180 - 20$$

$$= 90 - 20$$

$$= 70$$

AI Experts
Who Lead
The Future

02

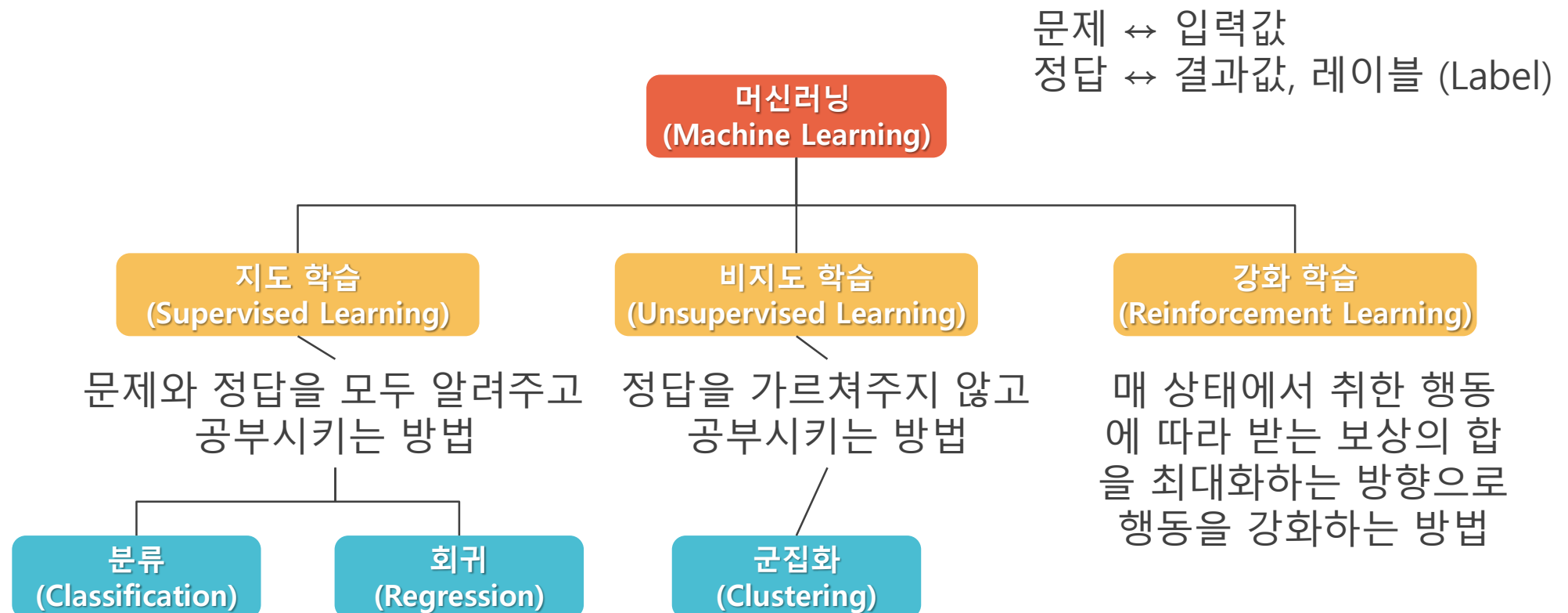
머신러닝 학습방법

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

머신러닝의 종류

인공지능 활용 Python language

- 머신러닝 모델을 학습시킴에 있어
 - 입력값에 대한 결과값을 알고 있는지 여부에 따라
 - “지도학습”과 “비지도 학습”으로 나누어짐



① 지도학습 (Supervised Learning)

인공지능 활용 Python language

- 입력값에 대한 정답 (=결과값)을 알고 있는 학습 데이터를 활용하여 머신러닝 모델을 학습시키는 방식
- 입력값에 대한 정답 (=결과값)을 레이블 (Label)이라고 함



그림 2-8 구분된 학습 데이터를 활용하는 지도학습의 예

© Hanbit Academy Inc.

- **분류 (Classification)와 회귀 (Regression)라는 2가지 유형**
 - ① **분류 (Classification)**
 - 어떤 입력 데이터가 들어오더라도 학습에 사용한 레이블 (Label)중 하나로 결과값을 결정
 - 레이블 (Label)이 이산적인 (Discrete) 경우 (즉, [0, 1, 2, 3, ...]와 같이 유한한 경우)
 - Ex) 이미지가 주어졌을 때, 고양이 (Class = 0) 이미지인지 또는 개 (Class = 1) 이미지인지 분류
 - ② **회귀 (Regression)**
 - 입력 데이터에 대한 결과값으로 학습에 사용한 레이블 이외의 값이 나올 수 있음
 - 레이블 (Label)이 실수인 경우
 - Ex) 키 (Height) 정보가 주어졌을 때, 몸무게를 예측

표 2-2 지도학습을 적용한 머신러닝 모델의 작업

작업	내용
분류	입력값에 대한 결과값이 정해진 레이블 중 하나로 결정되는 작업을 의미
회귀	입력값에 대한 결과값이 학습에 사용된 레이블 외의 값도 나올 수 있는 작업을 의미

㉠ 분류 (Classification) 작업

인공지능 활용 Python language

• 대표적인 예: 필기체 (손글씨) 인식

- 0부터 9까지의 손글씨 숫자 이미지와 레이블 (Label) 정보를 학습 데이터로 사용
- 어떤 이미지라도 (심지어는 숫자 이미지가 아니더라도) 0부터 9까지의 레이블 중 하나로 결과값을 결정함

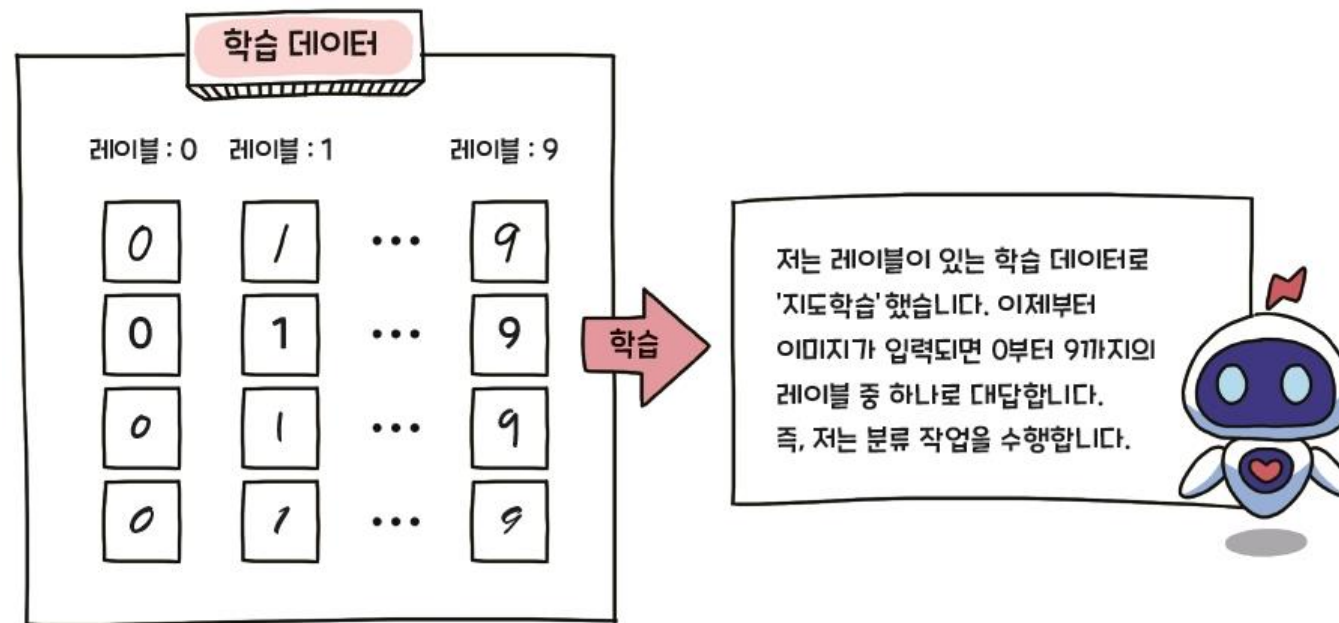


그림 2-9 머신러닝의 분류 작업 예: 숫자 필기체 인식하기

© Hanbit Academy Inc.

㉞ 회귀 (Regression) 작업

인공지능 활용 Python language

• 대표적인 예: 몸무게 예측 작업

- 여러 명의 키와 그에 대응하는 몸무게를 학습 데이터로 사용하는데, 입력된 키에 대한 몸무게를 결과값으로 출력하기 때문에 몸무게가 레이블 (Label)이 됨
- 다양한 키에 대해서 레이블에 포함되지 않은 몸무게도 결과값으로 출력 가능

No.	이름	키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

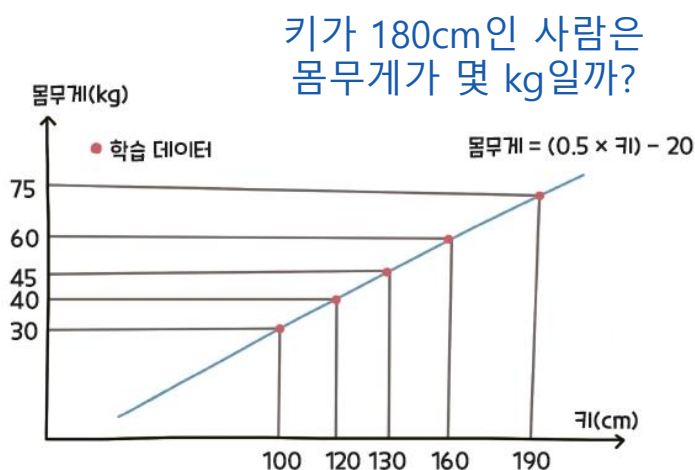


그림 2-7 몸무게와 키의 상관관계에 최적화된 직선의 방정식 © Hanbit Academy Inc.

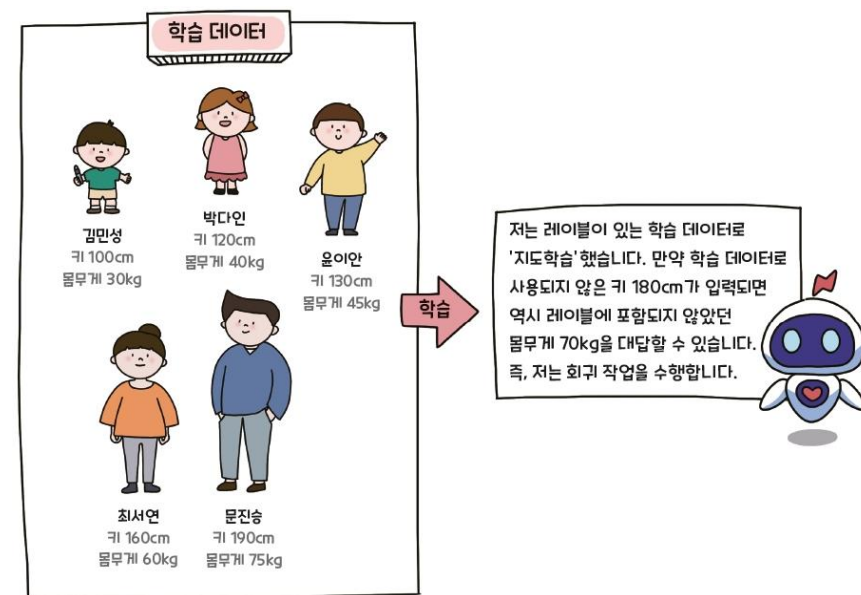


그림 2-10 머신러닝의 회귀 작업 예 : 키로 몸무게 예측하기 © Hanbit Academy Inc.

② 비지도학습 (Unsupervised Learning)

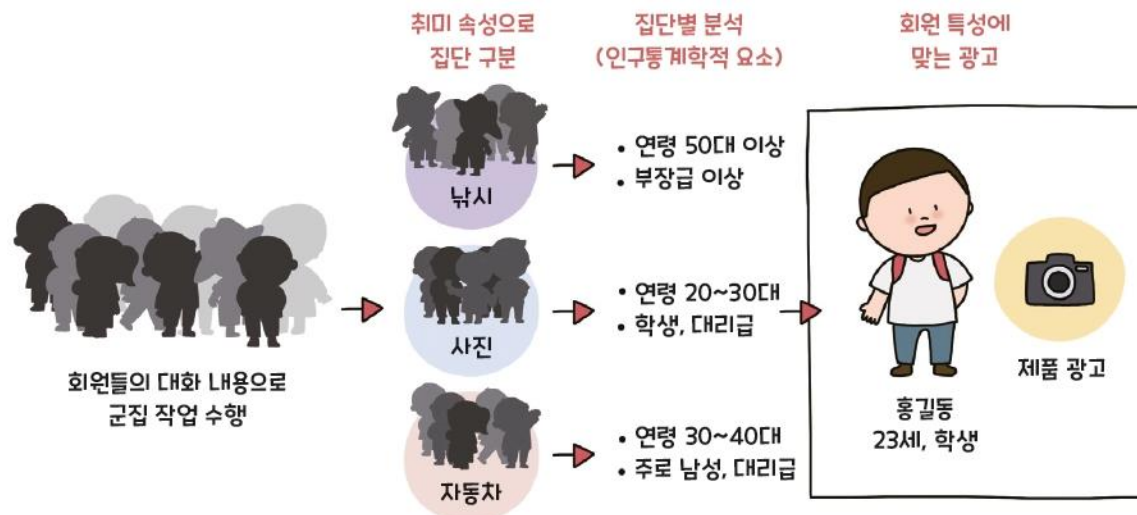
인공지능 활용 Python language

정의

- 입력값에 대한 정답 (=결과값)이 없는, 즉 레이블 (Label)이 없는 데이터를 사용하여 머신러닝 모델을 학습시키는 방식
- 비지도학습에는 대표적으로 군집화 (Clustering)라는 방법이 있음

군집화 (Clustering)

- 레이블 (Label) 없이
 - 데이터 간에 존재하는 특성을 바탕으로 비슷한 데이터 구분하여 비슷한 집단으로 묶는 작업
- 군집 작업은 마케팅 분야에서 많이 응용
 - 예를 들어, 채팅 사이트를 운영하는 기업이 사람들의 대화 내용을 기반으로 군집 작업을 수행해 취미를 속성으로 묶었다면, 집단별로 인구통계학적 특성을 파악해 적절한 취미용품을 광고할 수 있음



③ 강화학습 (Reinforcement Learning) (1/2)

인공지능 활용 Python language

- 입력값에 대한 정답 (=결과값) 대신 어떤 일을 잘했을 때 보상 (Reward)을 주는 것으로 머신러닝 모델을 학습시키는 방식
 - 핵심 목표
 - 시행착오를 통해 보상의 총합이 최대가 되는 일련의 행동을 찾는 것
 - 나중에 더 큰 이익을 기대하며 지금의 손해를 감수하는 식의 전략적 행동을 탐색할 수 있다는 의미와 같음
 - 응용분야
 - 게임 전략, 금융시장의 투자 전략, 광고 노출 전략 등과 같은 사업 분석 분야
 - 로봇 팔 제어, 이족보행 제어와 같은 로봇틱스 분야에서 널리 응용되고 있으며,
 - 자율주행 분야에서도 핵심 기술로 활용되고 있음

③ 강화학습 (Reinforcement Learning) 사례

인공지능 활용 Python language



(a) 슈퍼마리오(<https://www.youtube.com/watch?v=WzxmH1Cx2Yg>)



(a) 로봇 팔 제어 : 탁구(<https://www.youtube.com/watch?v=SH3bADiB7uQ>)



(b) 쿠키런(<https://www.youtube.com/watch?v=exXD6wJLJ6s>)

그림 2-13 강화학습을 통한 게임 제어 활용 예 © Hanbit Academy Inc.



(b) 로봇 팔 제어 : 팬케이크 뒤집기(https://www.youtube.com/watch?v=W_gxLKSSsIE)

© Hanbit Academy Inc.

AI Experts
Who Lead
The Future

03

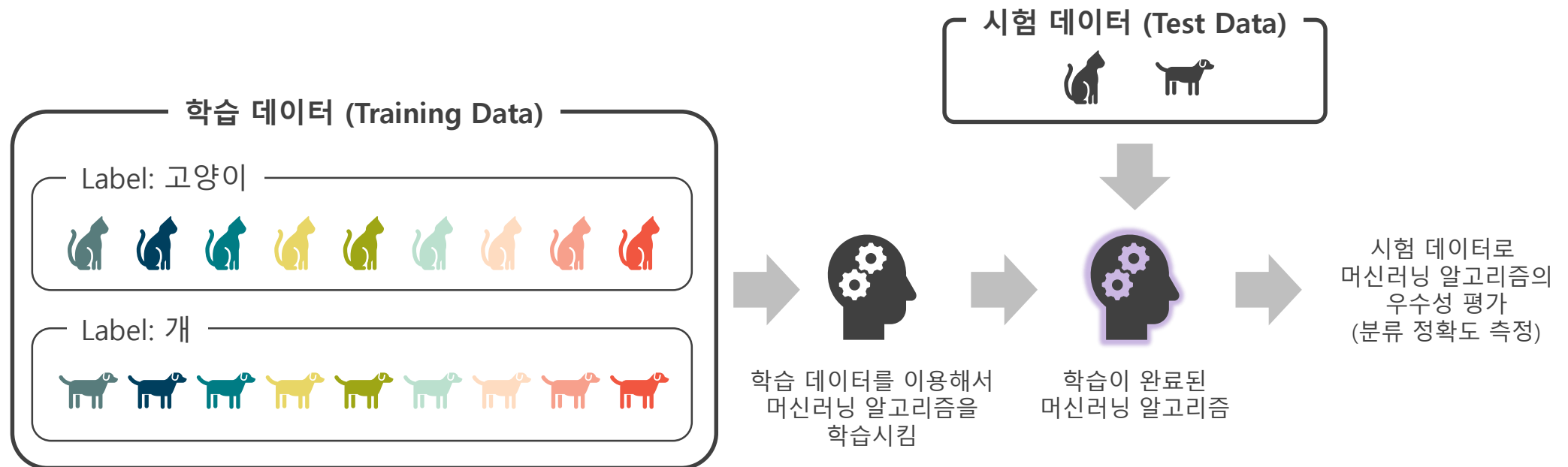
지도학습의 분류 알고리즘

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

분류 (Classification) 개요

인공지능 활용 Python language

- 레이블이 포함된 데이터를 학습하고 유사한 성질을 갖는 데이터끼리 분류한 후, 새로 입력된 데이터가 어느 그룹에 속하는지를 찾아내는 기법
 - 어떤 입력 데이터가 들어오더라도 학습에 사용한 레이블 (Label) 중 하나로 결과값을 결정
 - 레이블 (Label)이 이산적인 (Discrete) 경우 (즉, [0, 1, 2, 3, ...]와 같이 유한한 경우)



분류 (Classification)의 종류

인공지능 활용 Python language

이진 분류 (Binary Classification)

- 데이터를 2개의 그룹 (Class)으로 분류

Label: 고양이



Label: 개



"고양이" or "개"

둘 중에 하나로 분류

다중 분류 (Multiclass Classification)

- 데이터를 3개의 그룹 (Class) 이상으로 분류

Label: 고양이



Label: 개



Label: 토끼



"고양이" or "개" or "토끼"

셋 중에 하나로 분류

1. Define Machine Learning Problem and Construct Dataset

2. Training Phase

Training Dataset



Label

Raw Data
(텍스트, 이미지 등)
1-D or 2-D Vector

특성/특징 추출
(Feature Extraction)

Feature
Vector

머신러닝
모델

학습 (Training)

3. Testing Phase

Test Dataset



Raw Data
(텍스트, 이미지 등)
1-D or 2-D Vector

특성/특징 추출
(Feature Extraction)

Feature
Vector

머신러닝
모델

학습이 완료된
모델

(예측 or 분류)
결과값

특성/특징 추출 (Feature Extraction) (1/4)

인공지능 활용 Python language

상상해 봅시다!

아래 이미지들을 보고 연상되는 동물은?



꼬리



귀



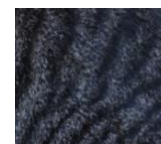
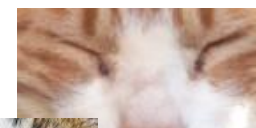
발바닥



코



눈



무늬



수염

특성/특징 추출 (Feature Extraction) (2/4)

인공지능 활용 Python language

- 사람은 어떻게 고양이 신체의 일부분 이미지들만 보고도, "고양이"라는 것을 어떻게 추론해 낼까요?

사람은 지금까지 살아오면서
다양한 고양이를 봐왔고,
"고양이"라면 갖고 있는 특징들을
이미 학습을 통해 알고 있다.



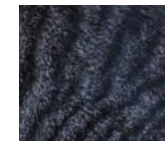
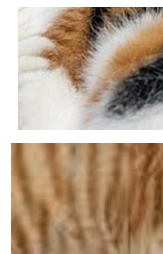
특징 #2: 긴 꼬리



특징 #1: 뾰족한 귀



특징 #3: 핑크색 발바닥



...

특징 #4: 무늬 (삼색, 치즈색, 고등어색)

<고양이라면 갖고 있는 몇가지 특징들>

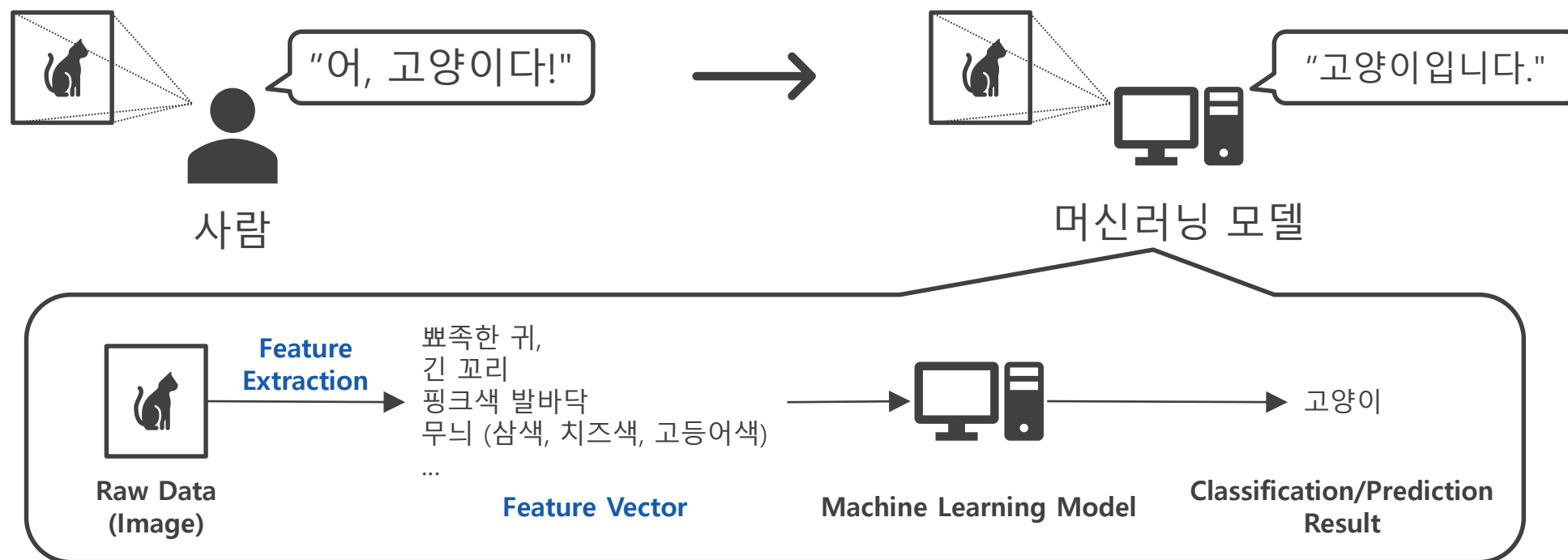
특성/특징 추출 (Feature Extraction) (3/4)

인공지능 활용 Python language

- 그렇다면, 머신러닝 모델이 주어진 이미지를 보고 고양이로 인지 할 수 있게 하려면 어떻게 해야 할까?

- Feature Extraction

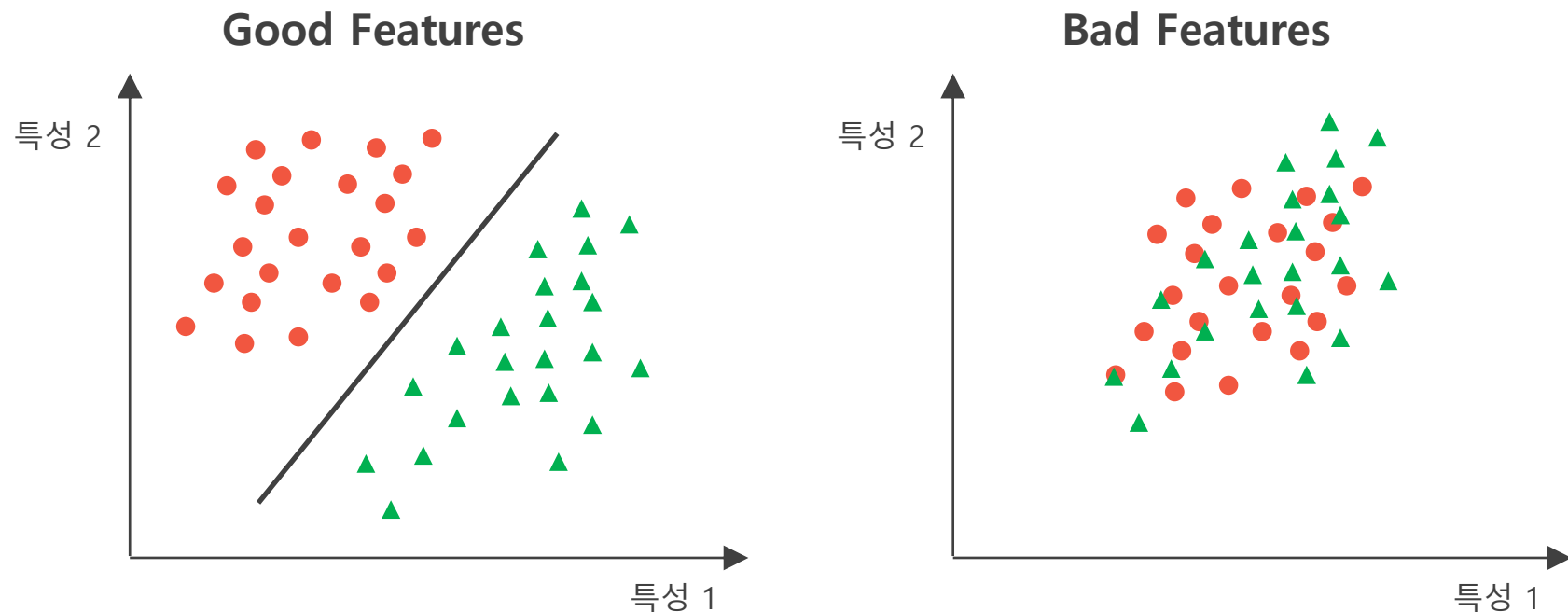
- 머신러닝 모델이 잘 학습할 수 있도록 학습 데이터 (Training Data)로부터 특성/특징 정보를 뽑아내는 작업



특성/특징 추출 (Feature Extraction) (4/4)

인공지능 활용 Python language

- Good and Bad Features



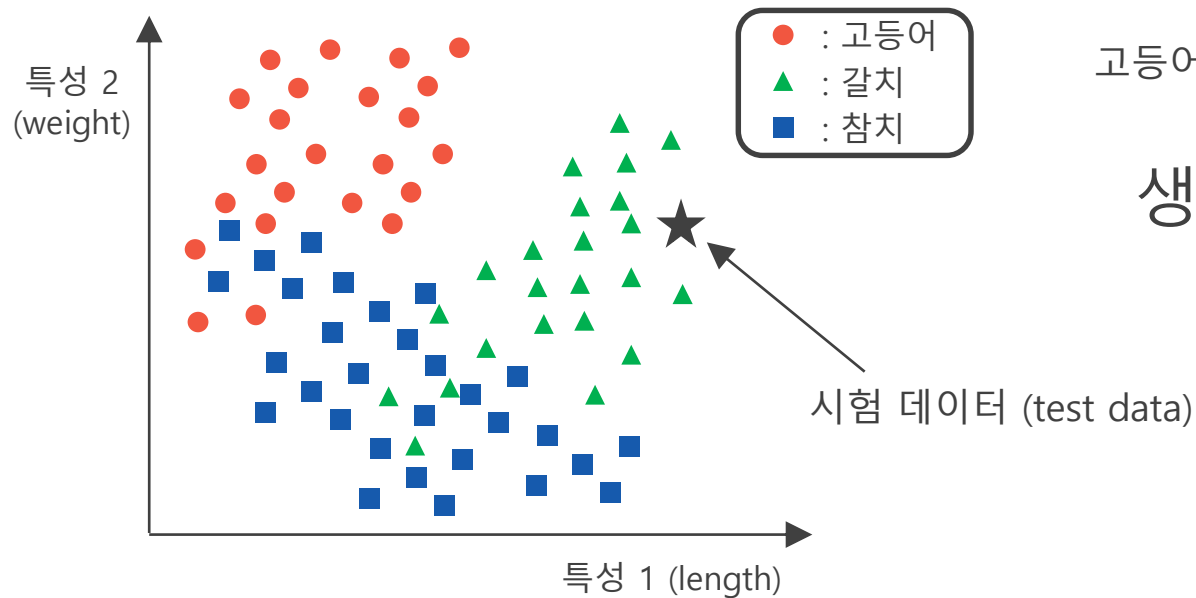
좋은 Features를 설계 (Design) 하는 것이 매우 중요하다!

분류 (Classification) 문제를 해결하기 위한 기법

인공지능 활용 Python language

- ① k -최근접 이웃 (k -Nearest Neighbors, k -NN)
- ② 결정 트리 (Decision Tree)
- ③ 랜덤 포레스트 (Random Forest)
- ④ 서포트 벡터 머신 (Support Vector Machine, SVM)
- ⑤ 앙상블 학습 (Ensemble Learning)

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (1/10) 인공지능 활용 Python language



수산시장에서 일하고 있는 동양이는
고등어, 갈치, 참치를 분류하는 장비를 개발하려고 한다.

생선 ★의 Label은 무엇일까?

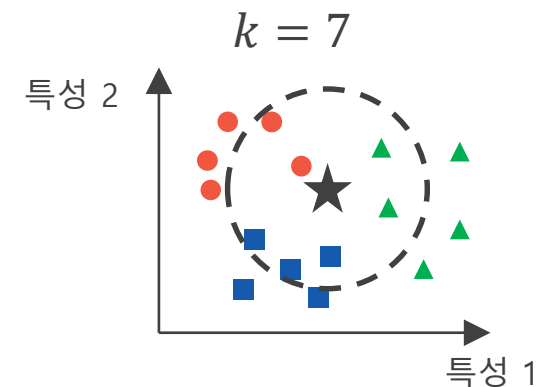
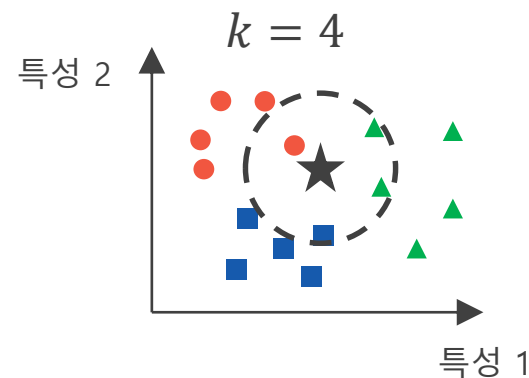
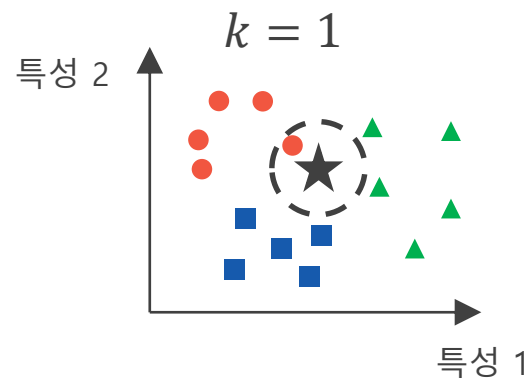
★ 주변에 갈치 ▲가 많은데...

시험 데이터 (test data)

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (2/10) 인공지능 활용 Python language

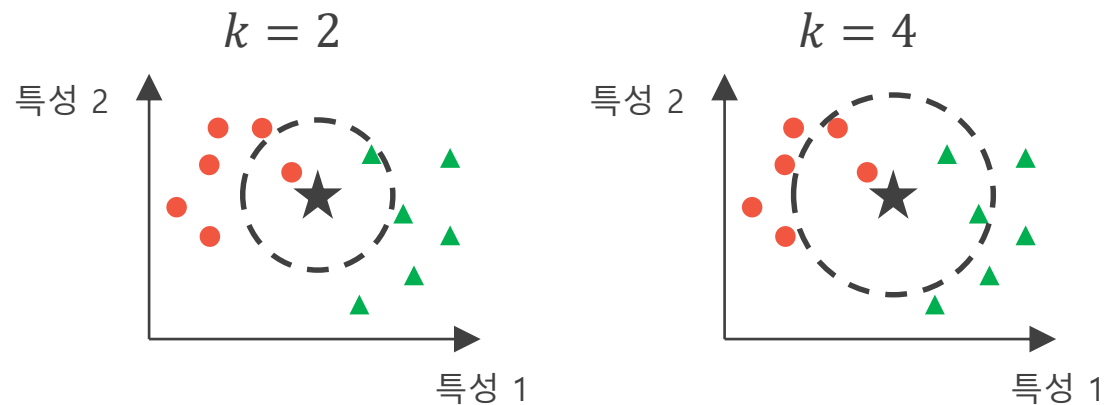
• ① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (2/10)

- 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k 개의 학습 데이터의 Labels을 참조하여 시험 데이터가 어떤 Label에 속하는지 분류하는 알고리즘
- 예를 들어,
 - $k = 1$ 일 때, 시험 데이터는 ● 빨간색 원으로 분류됨
 - $k = 3$ 일 때, 시험 데이터는 ▲ 초록색 삼각형으로 분류됨
 - $k = 6$ 일 때, 시험 데이터는 ■ 파란색 사각형으로 분류됨



① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (3/10) 인공지능 활용 Python language

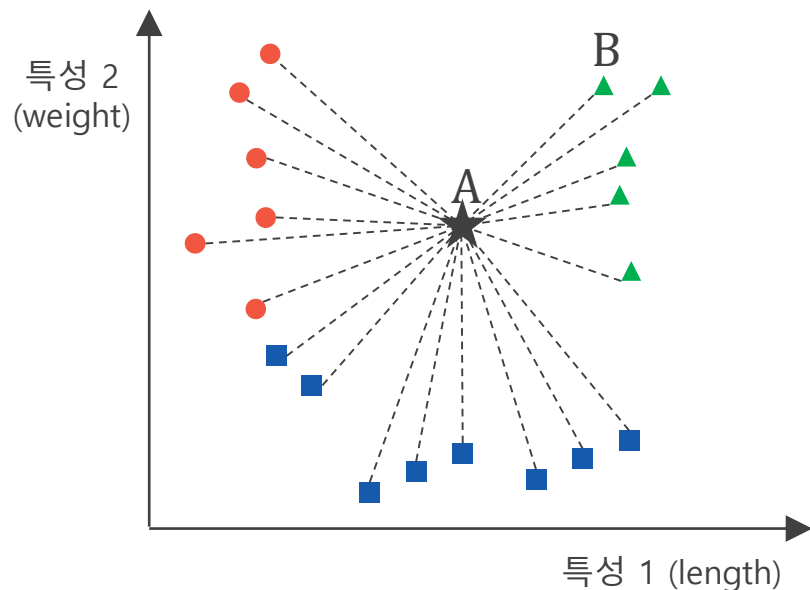
- ① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (3/10)
 - 어떤 값이 최적의 k 인지 불분명하기 때문에 데이터의 특성에 맞게 k 값을 임의로 선정해야 하는 단점이 있음
 - 일반적으로 이진 분류 (Binary Classification) 문제에서는 k 값을 홀수로 선정함



이진 분류 문제에서 k 값이 짝수인 경우, ★을 분류할 수 없다

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (4/10) 인공지능 활용 Python language

- 데이터 사이의 거리를 계산하기 위해서, 유클리드 거리 (Euclidean Distance)가 주로 사용됨



- Dimension of Feature Vector = 2

$$A(p_1, p_2)$$

$$B(q_1, q_2)$$

$$d = \overline{AB} = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

- Dimension of Feature Vector = 3

$$A(p_1, p_2, p_3)$$

$$B(q_1, q_2, q_3)$$

$$d = \overline{AB} = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}$$

- Dimension of Feature Vector = n

$$d = \overline{AB} = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_i - q_i)^2 + \dots + (p_n - q_n)^2}$$

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (5/10) 인공지능 활용 Python language

• ① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (5/10)

학습 데이터셋 (Training Dataset)

	Class	특성 1 (length)	특성 2 (weight)
갈치	0	25.4	3.7
	0	23.5	3.3
	0	24.7	3.5

고등어	1	10.9	2.4
	1	9.7	2.1
	1	10.2	2.3

시험 데이터 (Test Data)

(특성 1, 특성 2) = (24.3, 3.4)

Class 0 (갈치)에 속할까?

아니면

Class 1 (고등어)에 속할까?

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (1/10) 인공지능 활용 Python language

학습 데이터셋 (Training Dataset)

	Class	특성 1 (length)	특성 2 (weight)	거리 (d)
갈치	0	25.4	3.7	1.14
	0	23.5	3.3	0.81
	0	24.7	3.5	0.41

고등어	1	10.9	2.4	13.44
	1	9.7	2.1	14.66
	1	10.2	2.3	14.14

시험 데이터 (Test Data)

(특성 1, 특성 2) = (24.3, 3.4)

Step 1) 학습 데이터셋에 있는 각 데이터들과 거리를 계

$$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

$$d = \sqrt{(24.3 - 25.4)^2 + (3.4 - 3.7)^2} = 1.14$$

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (7/10) 인공지능 활용 Python language

학습 데이터셋 (Training Dataset)

	Class	특성 1 (length)	특성 2 (weight)	거리 (d)
갈치	0	25.4	3.7	1.14
	0	23.5	3.3	0.81
	0	24.7	3.5	0.41

고등어	1	10.9	2.4	13.44
	1	9.7	2.1	14.66
	1	10.2	2.3	14.14

Step 2) $k = 3$ 이라고 하면, 가장 작은 d 값 3개의 Class

Class	거리 (d)
0	1.14
0	0.81
0	0.41

선별된 3개의 Class 중에서
다수를 차지하고 있는 Class를
시험 데이터의 Class로 결정

시험 데이터
(Test Data)

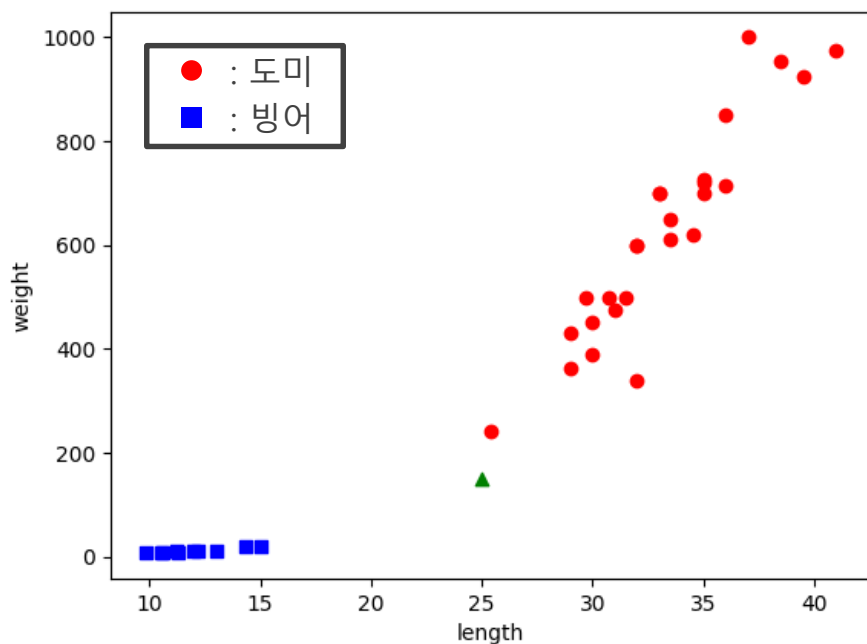


Class 0 (갈치)로 분류!

(특성 1, 특성 2) = (24.3, 3.4)

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (8/10) 인공지능 활용 Python language

- (주의 사항) 특성들 사이에 스케일 (Scale)을 맞춰야 한다



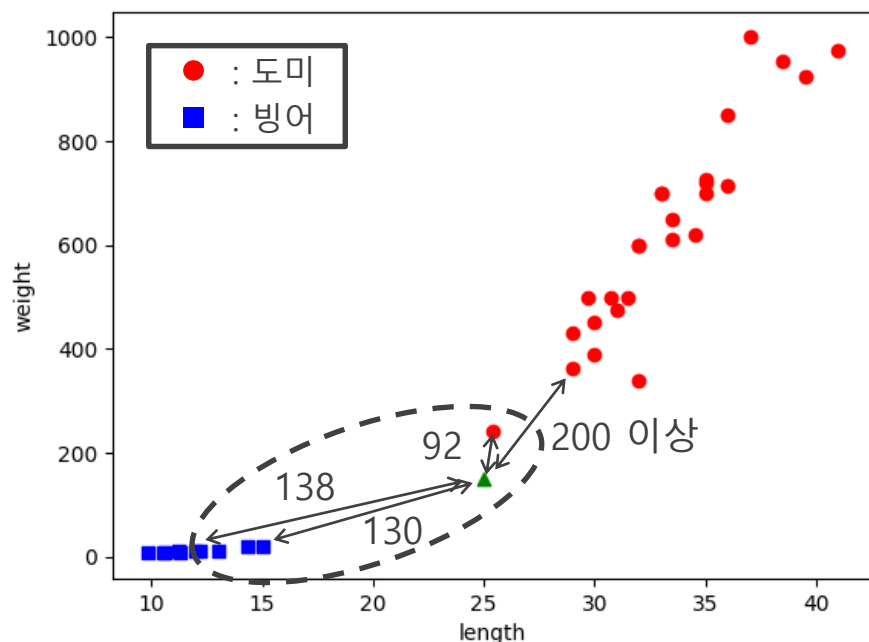
$k = 5$ 라고 가정해 보자!

생선 ▲의 Label은 무엇일까?

●랑 더 가까운 것 같은데...

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (9/10) 인공지능 활용 Python language

- (주의 사항) 특성들 사이에 스케일 (Scale)을 맞춰야 한다



$k = 5$ 라고 가정해 보자!

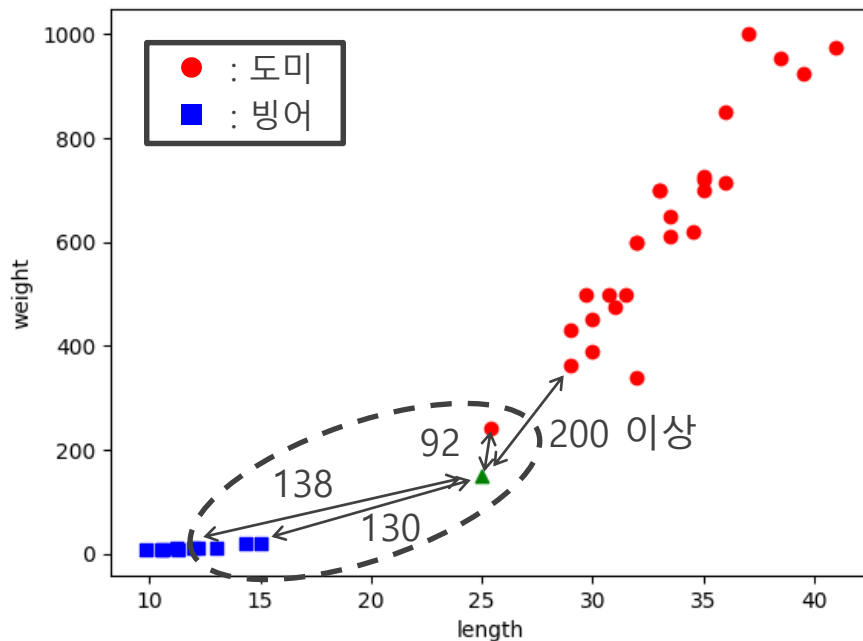
▲랑 가까운 5개의 데이터를 표시해 보면?!

■ (빙어) 4개, ● (도미) 1개로 ▲는 빙어로 분류된다?!

왜 그럴까?!

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (10/10) 인공지능 활용 Python language

- (주의 사항) 특성들 사이에 스케일 (Scale)을 맞춰야 한다



특성 1 (length)와 특성 2 (weight)의 Scale이 다르다!

쉽게 말해서, 두 특성이 갖는 값의 범위가 다르다!

$$0 < \text{length} < 50$$

$$0 < \text{weight} < 1000$$

Scale이 서로 다른 상태에서 유클리드 거리를 계산하면 k -NN 알고리즘이 올바르게 분류 작업을 할 수 없다!

Scale을 맞춰주는 작업:
$$x_{\text{new}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

② 결정 트리 (Decision Tree) (1/8)

인공지능 활용 Python language

- 의사결정 규칙을 나무 형태로 분류하는 분석 방법
- [그림 8-19]와 같이 상위 노드에서 시작하여 분류 기준값에 따라 하위 노드로 확장하는 방식
이
“나무”를 닮았다고 하여 “의사 결정 나무”라고도 불림

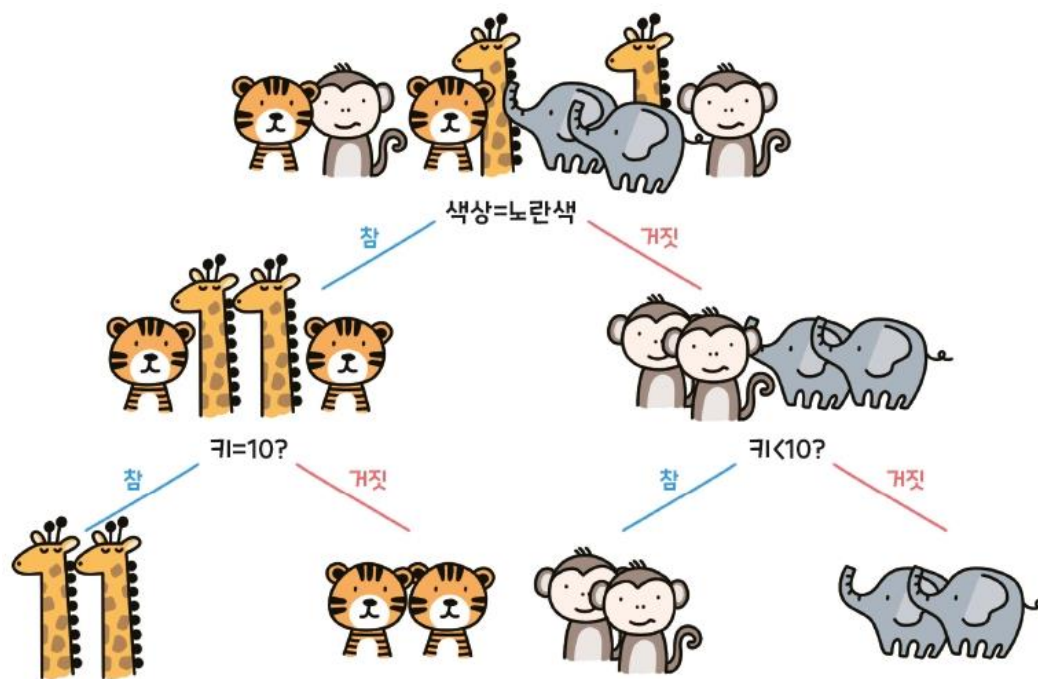


그림 8-19 의사결정나무 구조

© Hanbit Academy Inc.

② 결정 트리 (Decision Tree) (2/8)

인공지능 활용 Python language

- Root Node
- Intermediate Node
- Terminal Node (Leaf Node)

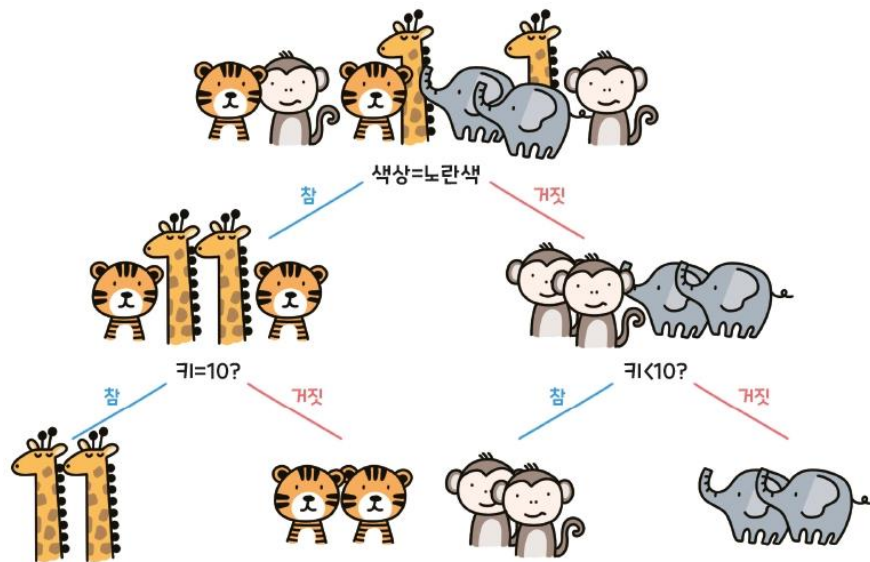
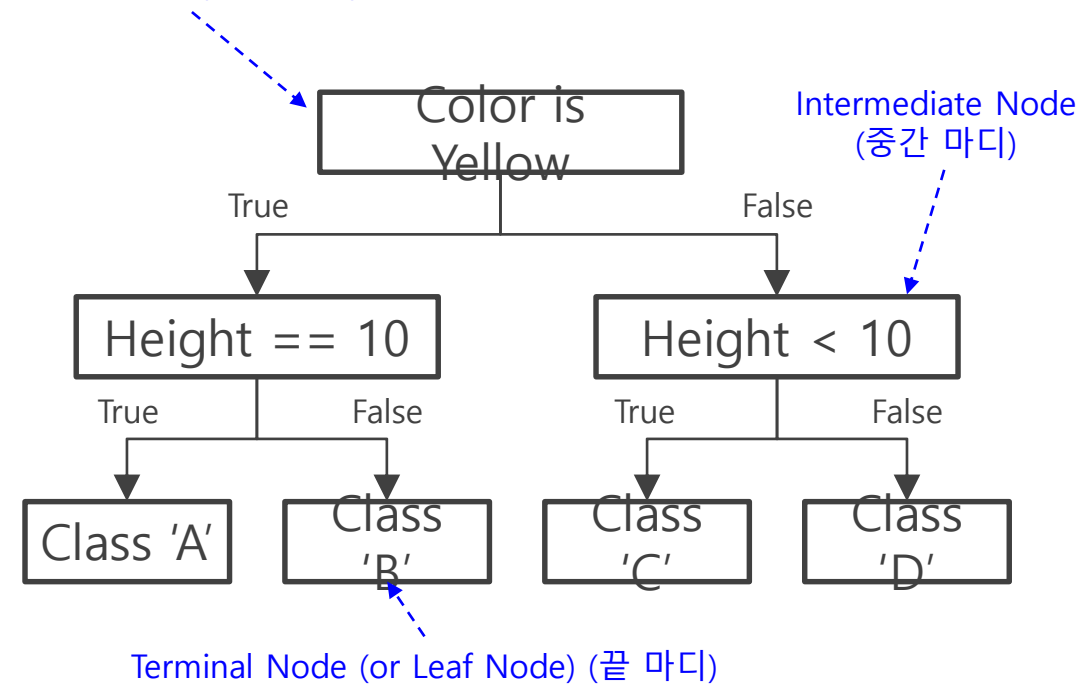


그림 8-19 의사결정나무 구조

© Hanbit Academy Inc.

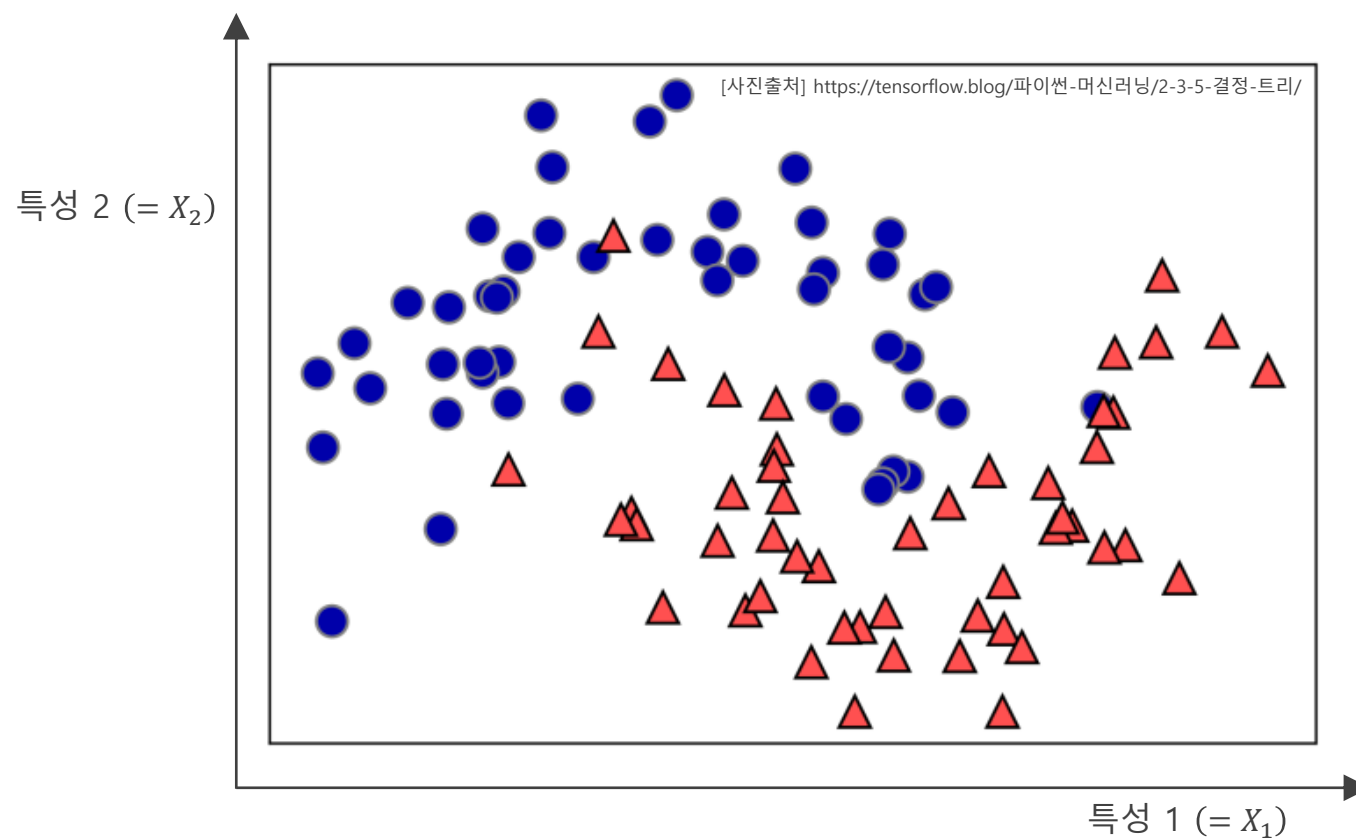
Root Node (뿌리 마디)



Terminal Node (or Leaf Node) (끝 마디)

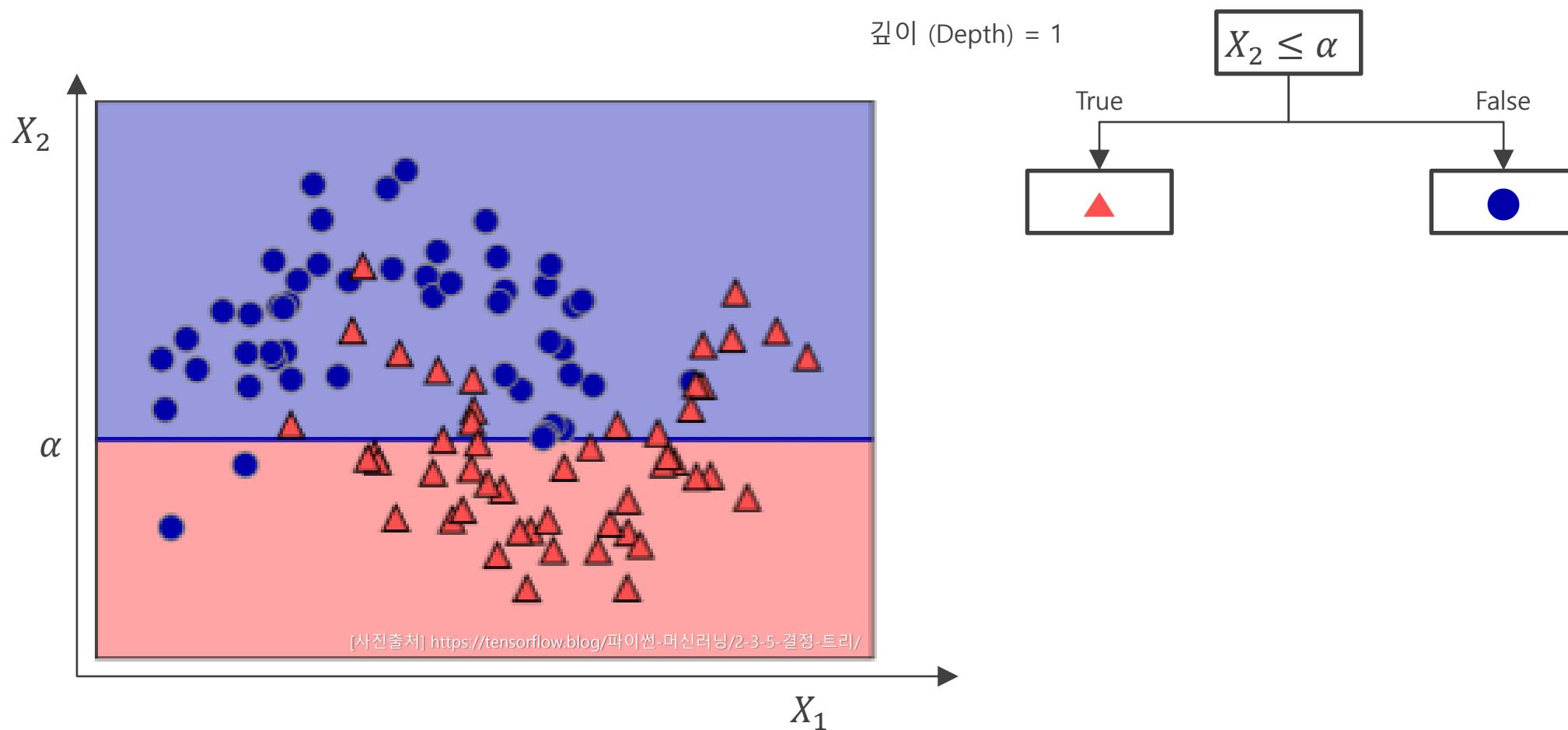
② 결정 트리 (Decision Tree) (3/8)

인공지능 활용 Python language



② 결정 트리 (Decision Tree) (4/8)

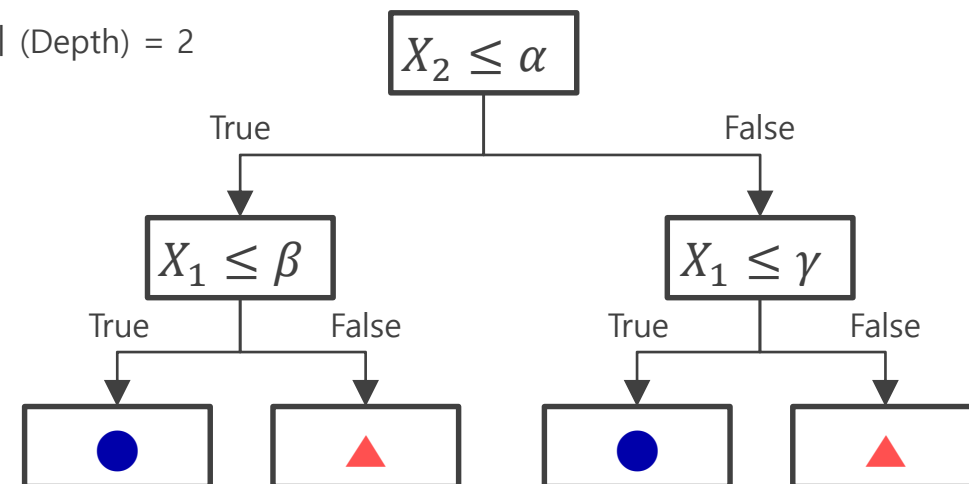
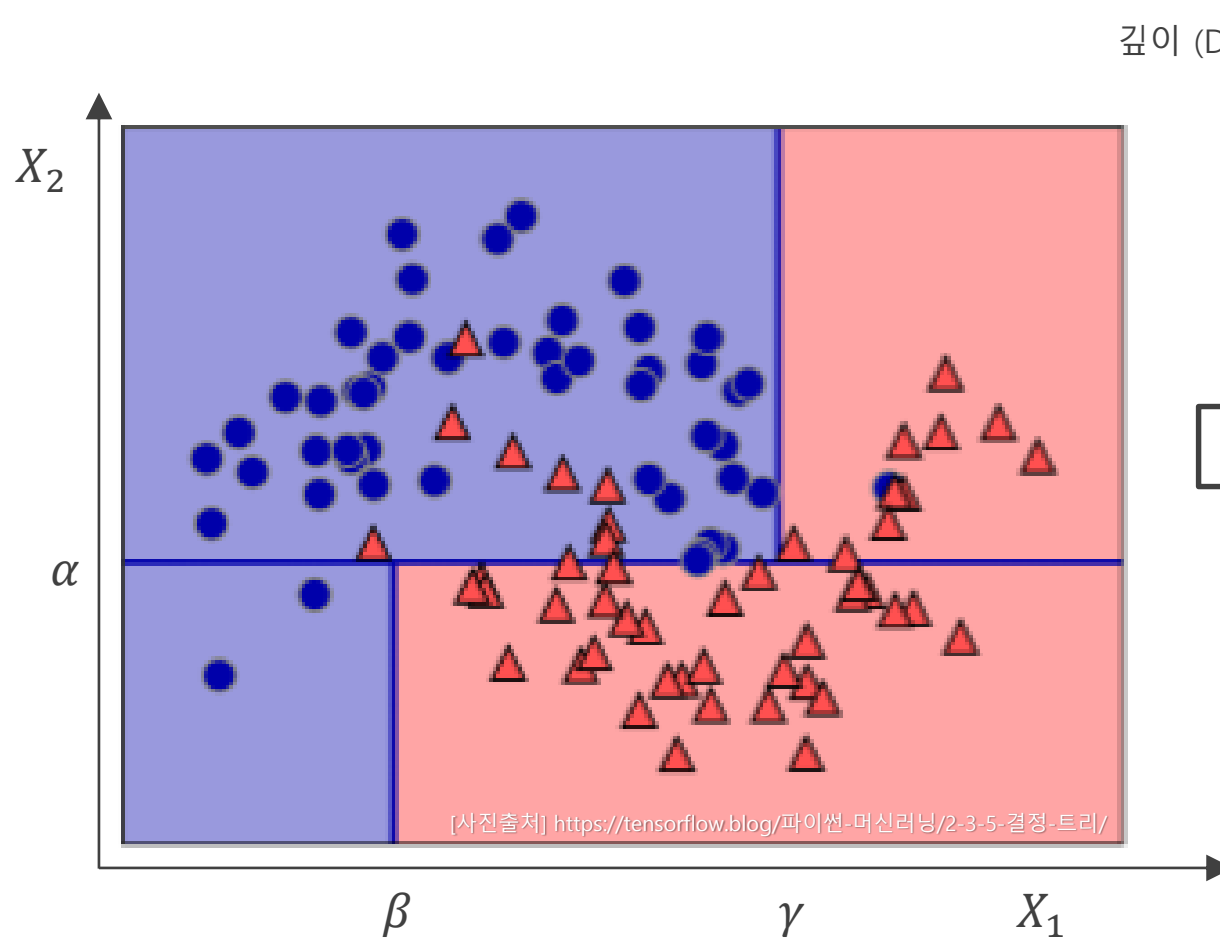
인공지능 활용 Python language



② 결정 트리 (Decision Tree) (5/8)

인공지능 활용 Python language

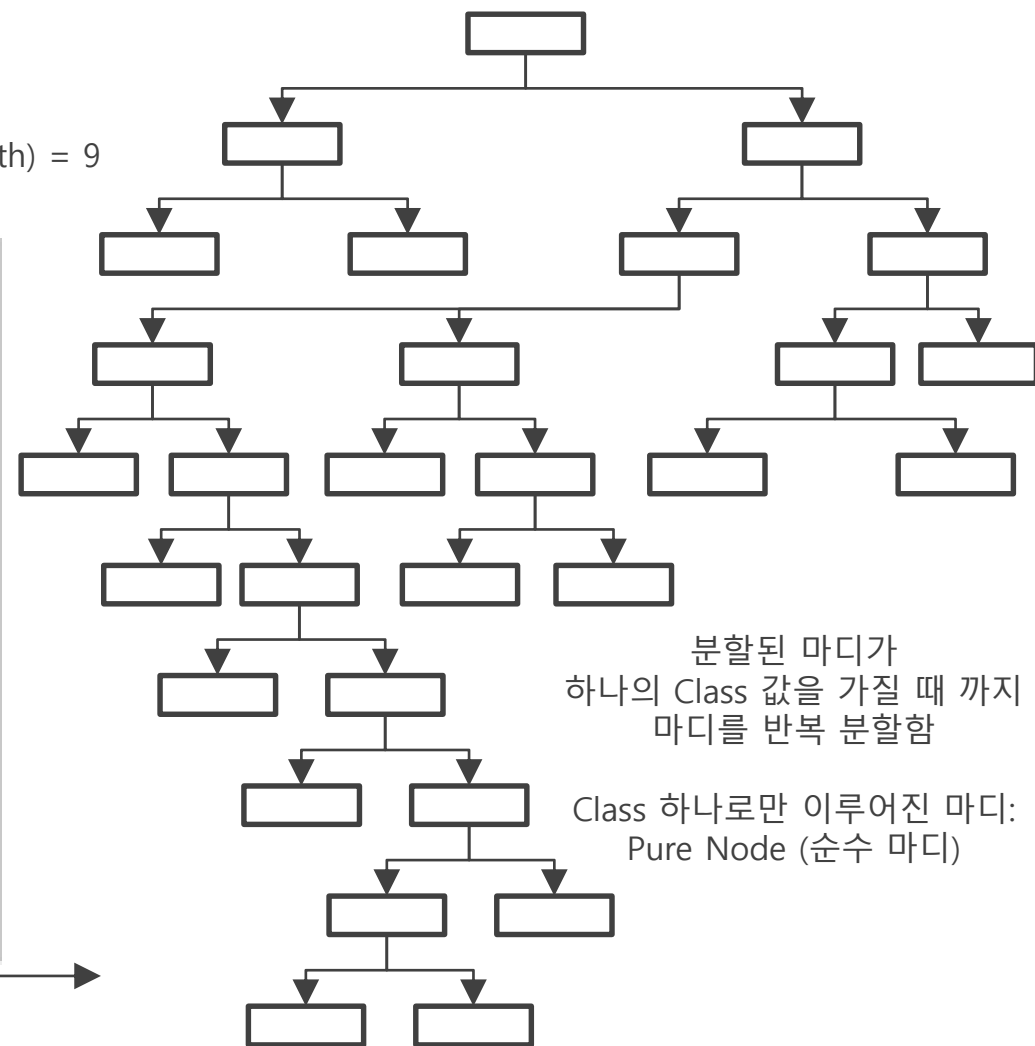
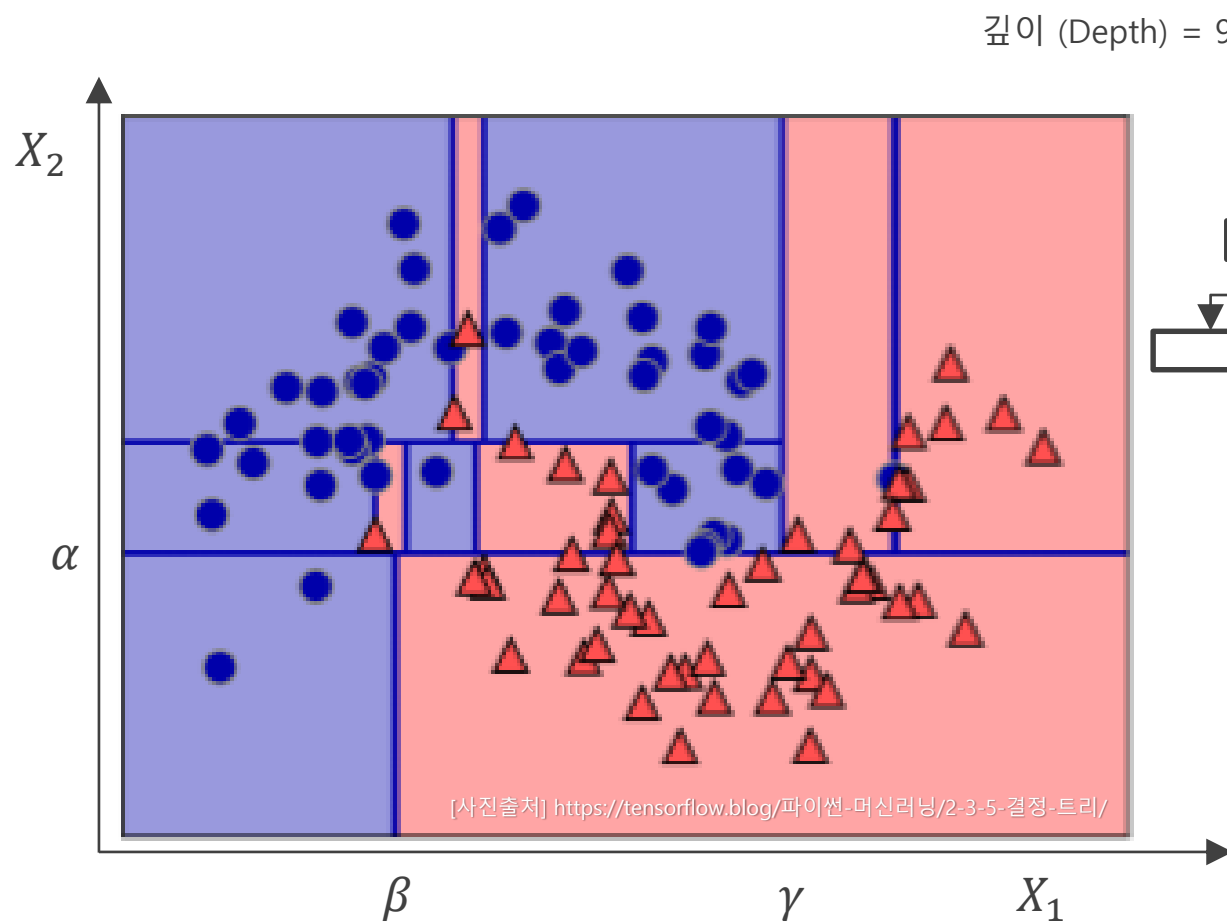
② 결정 트리 (Decision Tree) (5/8)



② 결정 트리 (Decision Tree) (6/8)

인공지능 활용 Python language

• ② 결정 트리 (Decision Tree) (6/8)



② 결정 트리 (Decision Tree) (7/8)

인공지능 활용 Python language

- ② 결정 트리 (Decision Tree) (7/8)

- 모든 끝 마디 (Terminal Node or Leaf Node)가 순수 마디 (Pure Node)가 될 때 까지, 가지 분할을 계속하게 되면 결정 트리 모델이 복잡해 지고 훈련 데이터 (Training Data)에 과대적합 (Overfitting)하게 됨
- 해결책으로 크게 2가지가 있음
 - 최대 깊이 (Depth)를 사전에 정하고 결정 트리 모델을 생성 (사전 가지치기, Pre-pruning)
 - 결정 트리 모델을 만들고 나서, 데이터 수가 적은 마디 (Node)를 제거 (사후 가지치기, Post-pruning)

② 결정 트리 (Decision Tree) (8/8)

인공지능 활용 Python language

- ② 결정 트리 (Decision Tree) (8/8)
 - 결정 트리는 분석 과정이 직관적이고 이해하기 쉬움
 - 인공신경망의 경우 분석 결과에 대한 설명이 어려운 블랙박스 모델인 반면, 결정 트리는 분석 과정을 눈으로도 관측할 수 있음
 - 그래서 결과에 대한 명확한 설명이 필요할 때 많이 사용함
 - k -NN 모델과 다르게 결정 트리 모델은 데이터의 Scale에 대한 전처리 (Pre-processing) 과정이 불필요

③ 랜덤 포레스트 (Random Forest) (1/3)

인공지능 활용 Python language

- ③ 랜덤 포레스트 (Random Forest) (1/3)

- 앙상블 (Ensemble) 학습 방법의 일종으로, 훈련 (Training) 과정에서 구성한 다수의 결정 트리 (Decision Tree)로부터 출력된 분류 결과로부터 다수결의 원칙에 따라 최종 분류 결과를 결정하는 방식으로 동작



나무 1
(Tree 1)



나무 2
(Tree 2)

...



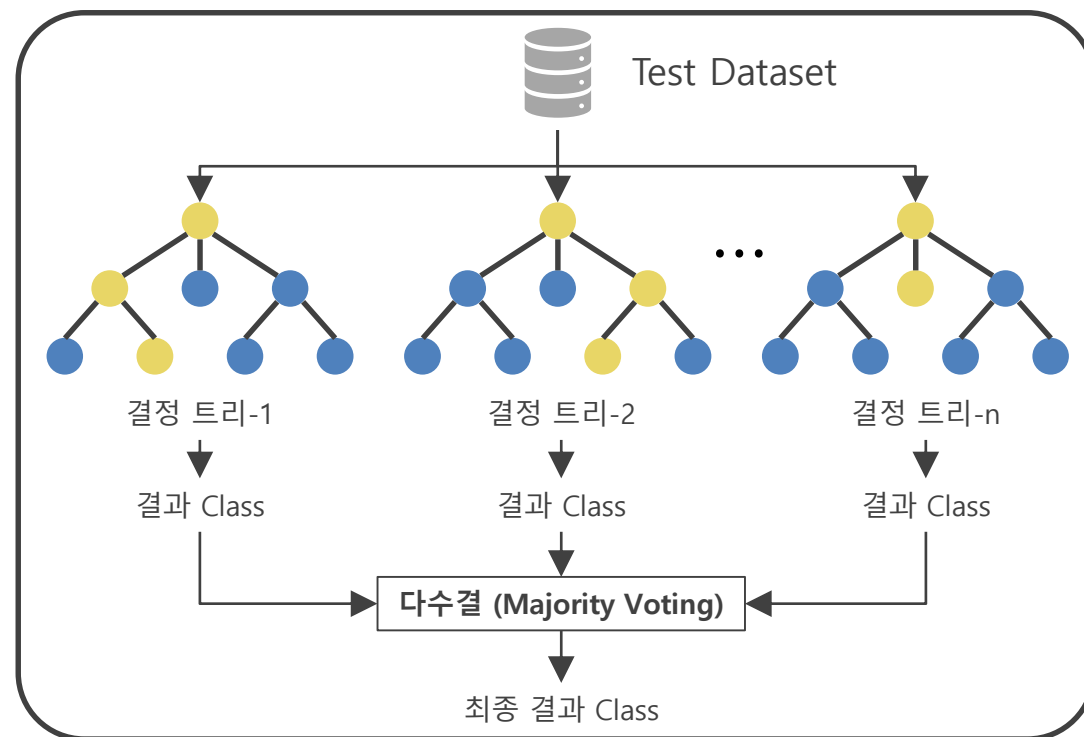
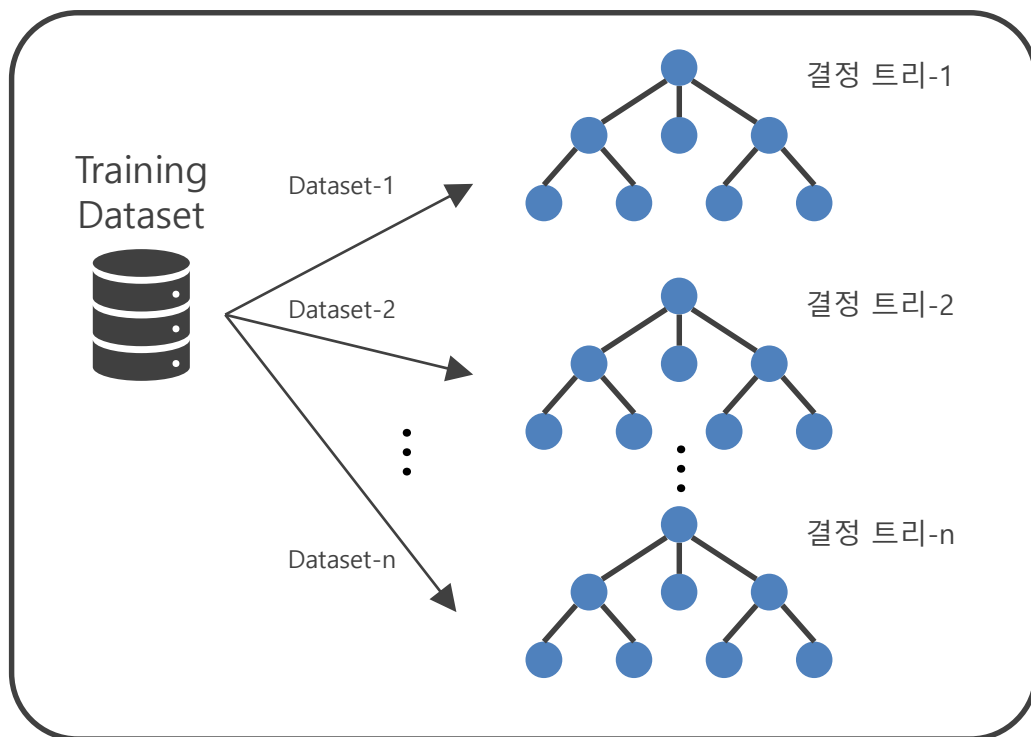
나무 n
(Tree n)

나무 (Tree)가 모여서 숲 (Forest)을 이룬다

③ 랜덤 포레스트 (Random Forest) (2/3)

인공지능 활용 Python language

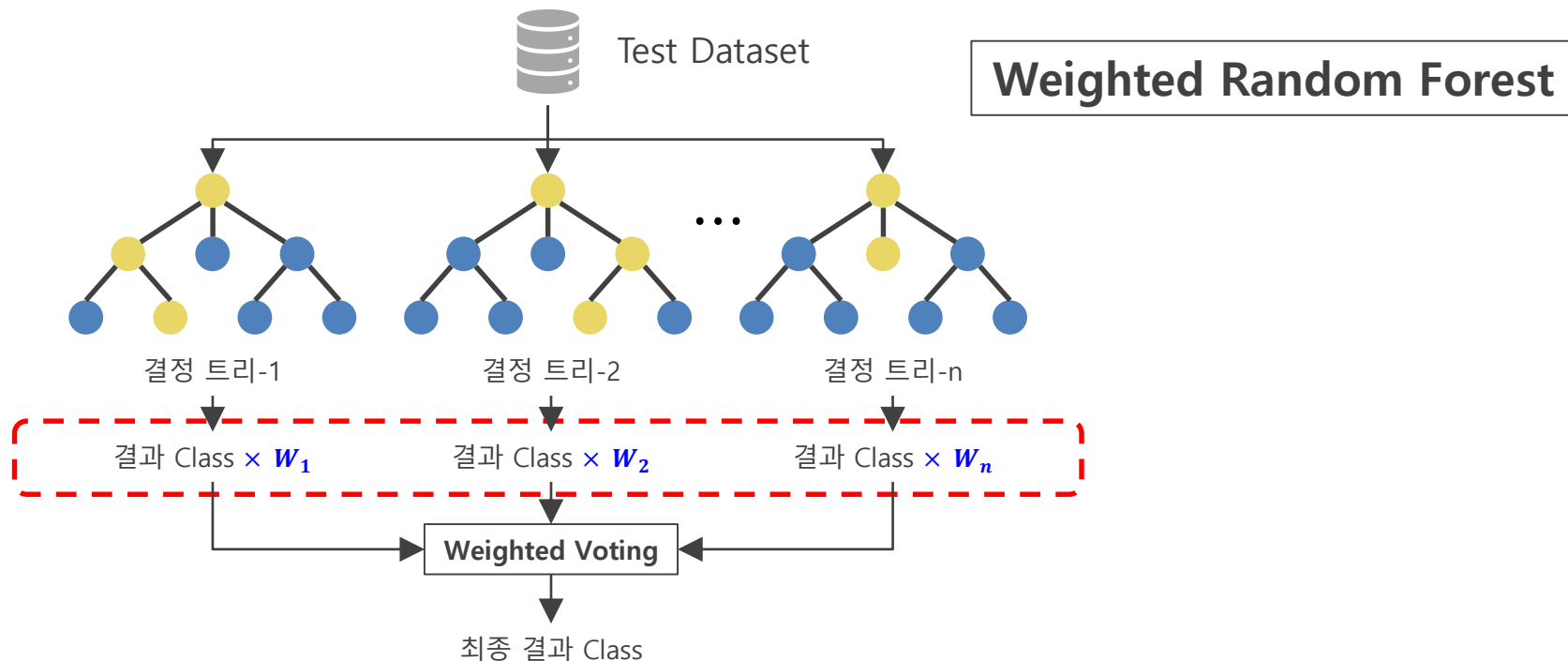
- 앙상블 (Ensemble) 학습 방법의 일종으로,
 - 훈련 (Training) 과정에서 구성한 다수의 결정 트리 (Decision Tree)로부터 출력된 분류 결과로부터
 - 다수결의 원칙에 따라 최종 분류 결과를 결정하는 방식으로 동작



③ 랜덤 포레스트 (Random Forest) (3/3)

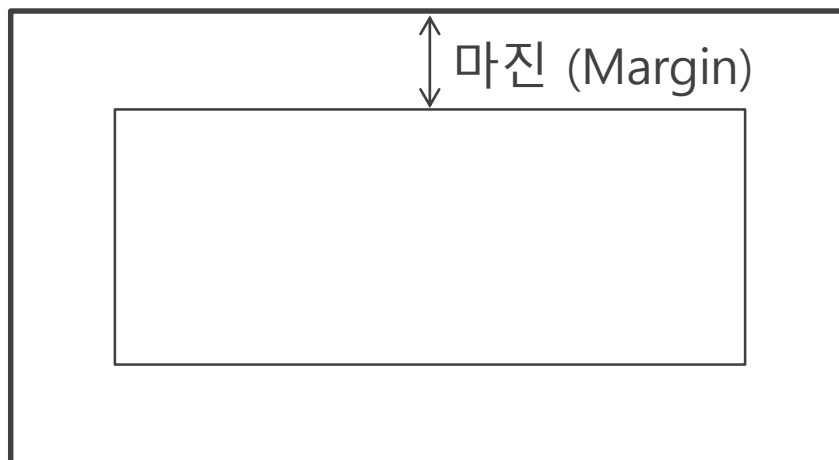
인공지능 활용 Python language

- 결정 트리 (Decision Tree)마다 서로 다른 가중치 (Weight)를 부여하여,
 - 가중치가 큰 결정 트리로부터의 출력 분류 결과가 최종 분류 결과에 영향력을 더 많이 가할 수 있게 할 수도 있다



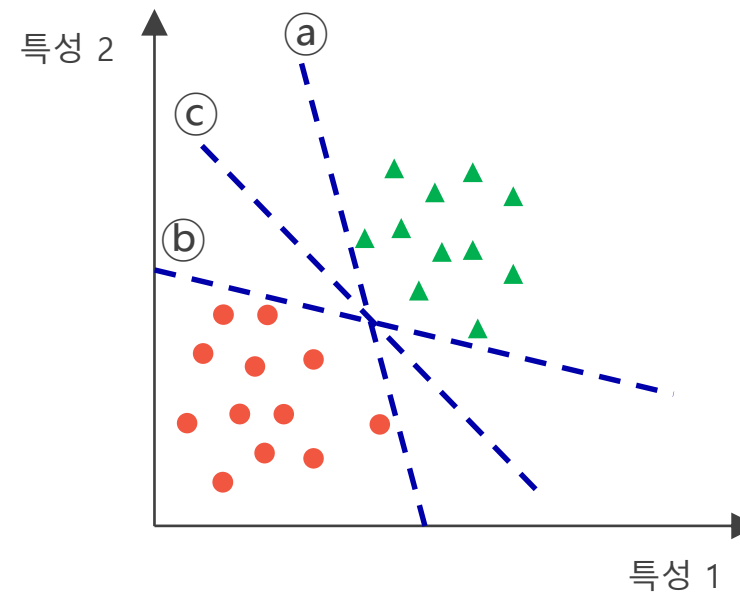
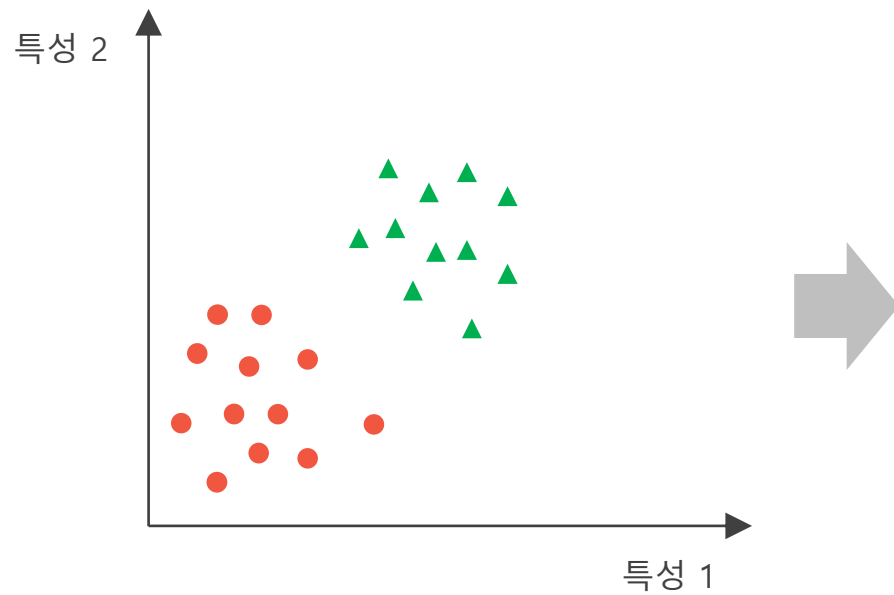
④ 서포트 벡터 머신 (Support Vector Machine, SVM) (1/12) 인공지능 활용 Python language

- ④ 서포트 벡터 머신 (Support Vector Machine, SVM) (1/12)
 - 주어진 데이터가 어느 그룹에 속하는지 분류하는 모델
 - 두 분류 (Classes) 사이의 여백을 의미하는 마진 (Margin)을 최대화하는 방향으로 데이터를 분류
 - SVM은 마진을 극대화하는 선을 찾아 분류하므로
 - 마진이 크면 클수록 새로운 데이터가 들어오더라도 잘 분류할 가능성이 높아짐
 - SVM은 사용 방법이 쉽고 예측 정확도가 높다는 장점
 - 하지만 모델 구축에 시간이 오래 걸리고 결과에 대한 설명력이 떨어지는 단점



④ 서포트 벡터 머신 (Support Vector Machine, SVM) (2/12) 인공지능 활용 Python language

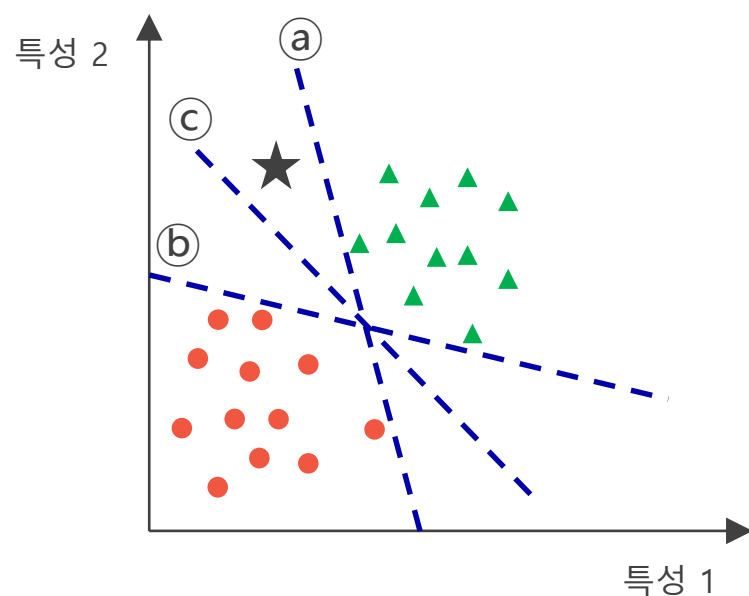
●과 ▲는 학습 데이터 (Training Data)



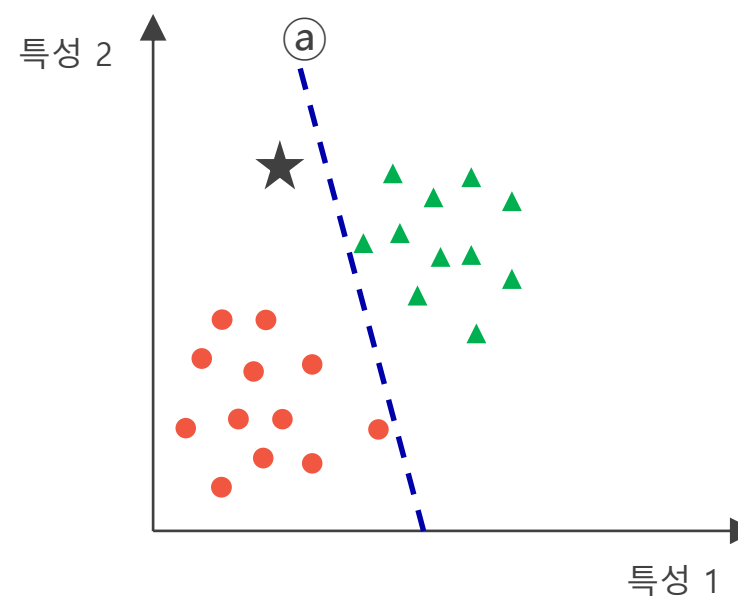
①, ②, ③ 모두 ●과 ▲를 잘 분류하고 있다

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (3/12) 인공지능 활용 Python language

시험 데이터 ★이 주어졌다.



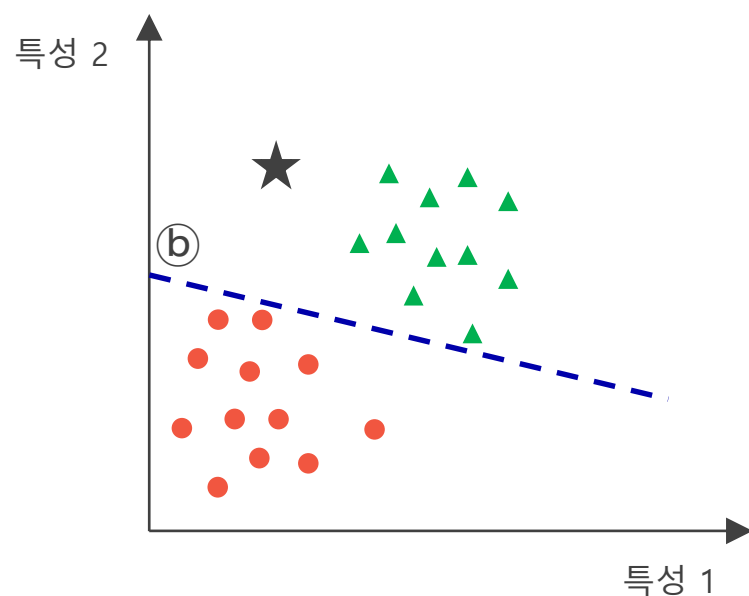
분류기 (Classifier) ①를 살펴보자.



①는 ★을 ●로 분류한다

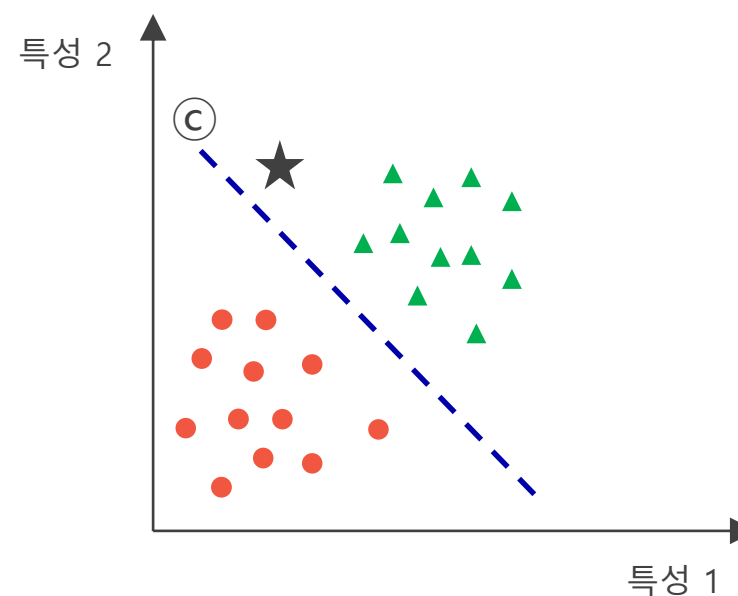
④ 서포트 벡터 머신 (Support Vector Machine, SVM) (4/12) 인공지능 활용 Python language

분류기 (Classifier) ⑥를 살펴보자.



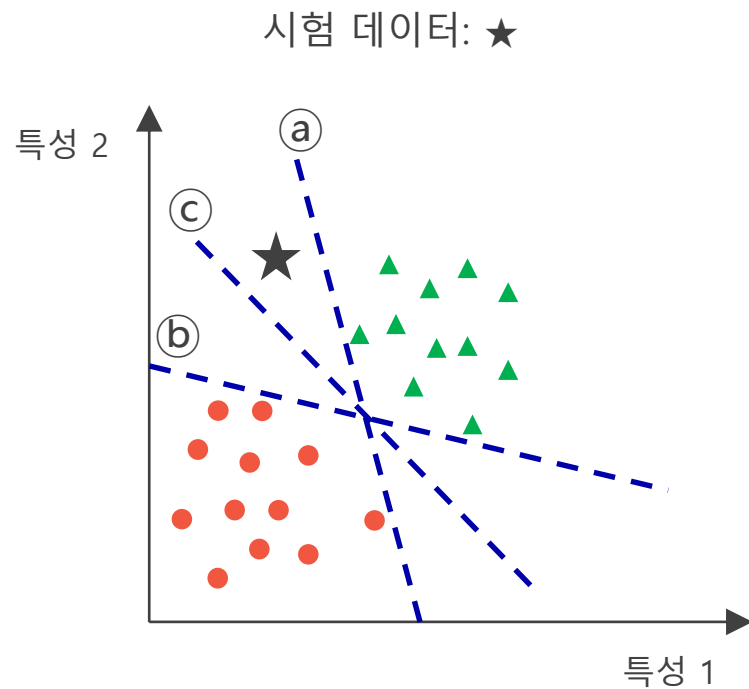
⑥는 ★을 ▲로 분류한다

분류기 (Classifier) ⑦를 살펴보자.



⑦는 ★을 ▲로 분류한다

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (5/12) 인공지능 활용 Python language



①는 ★을 ●로 분류한다

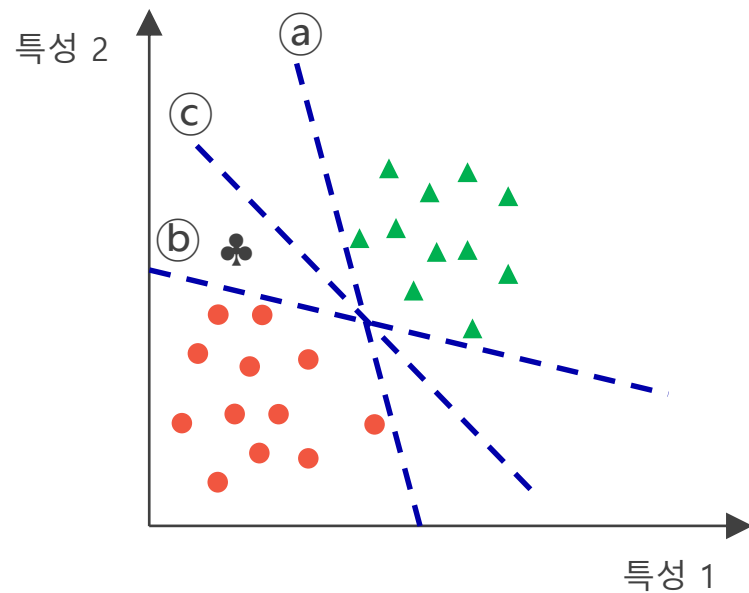
②는 ★을 ▲로 분류한다

③는 ★을 ▲로 분류한다

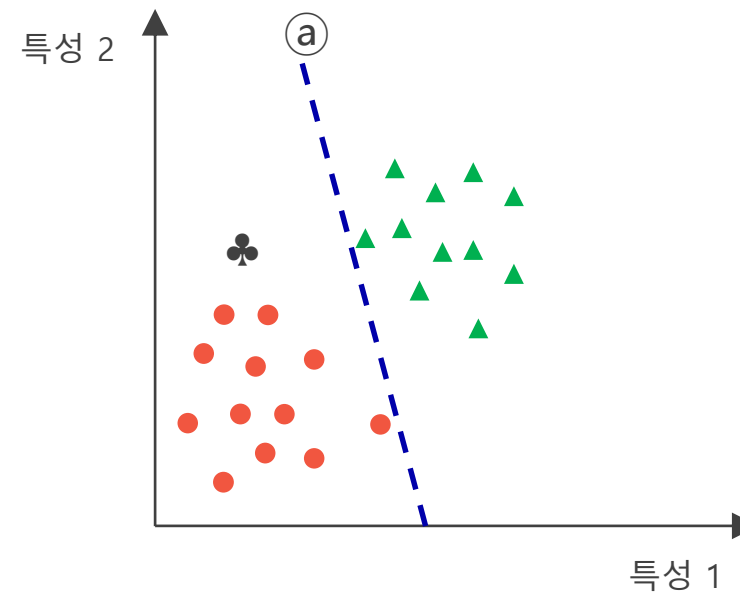
①, ②, ③ 중에 어떤 모델이 더 우수한 걸까?

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (6/12) 인공지능 활용 Python language

New 시험 데이터: ♣



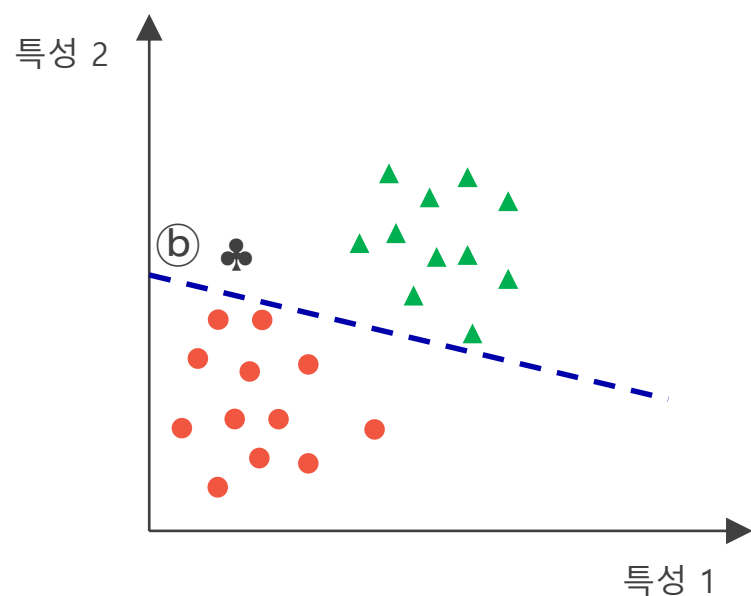
분류기 (Classifier) (a)를 살펴보자.



①는 ♣을 ●로 분류한다

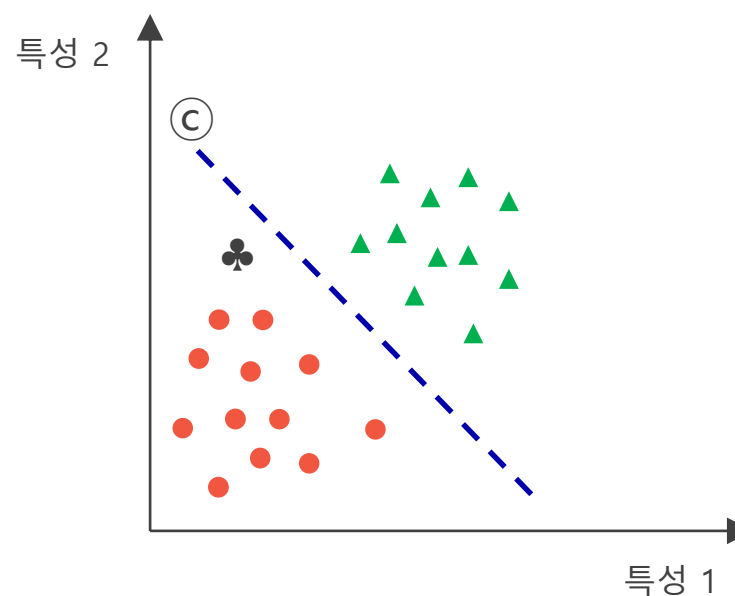
④ 서포트 벡터 머신 (Support Vector Machine, SVM) (7/12) 인공지능 활용 Python language

분류기 (Classifier) ⑥를 살펴보자.



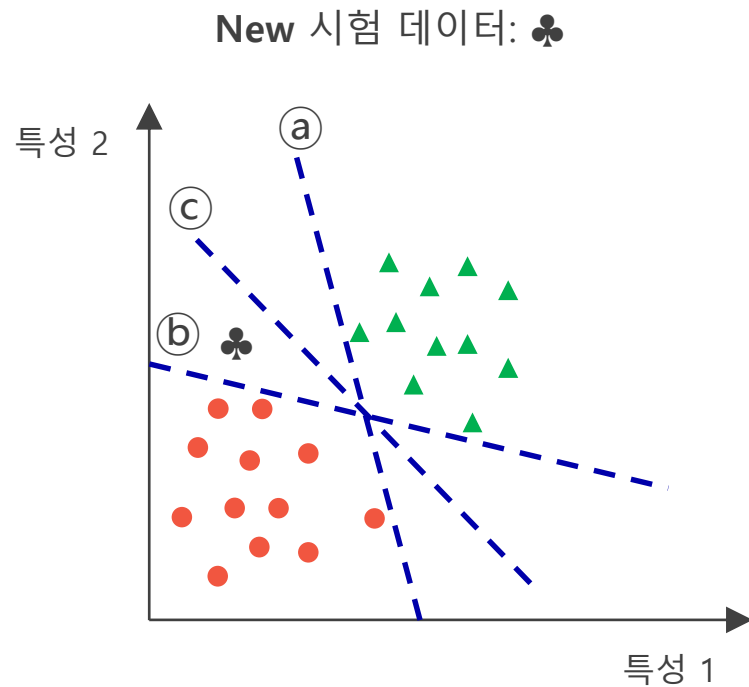
⑥는 ♣을 ▲로 분류한다

분류기 (Classifier) ⑦를 살펴보자.



⑦는 ♣을 ●로 분류한다

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (8/12) 인공지능 활용 Python language



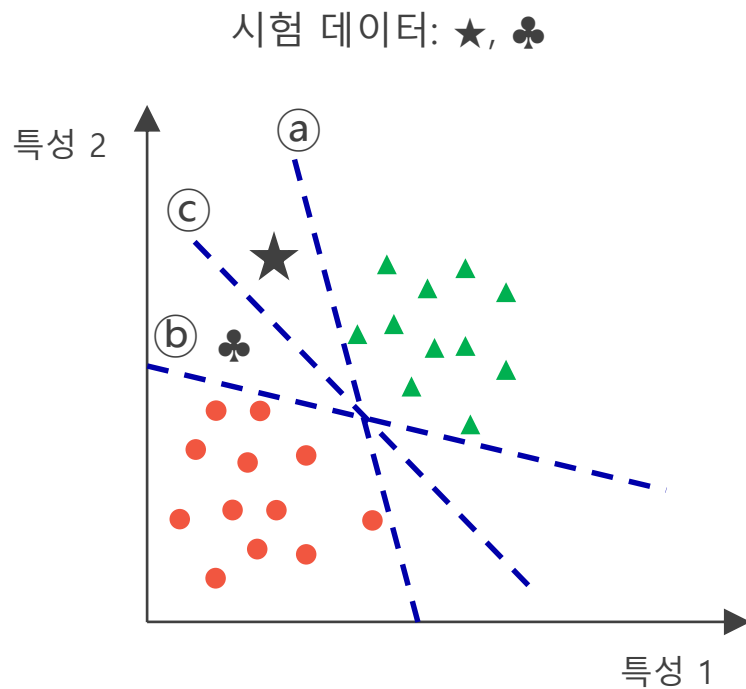
①는 ♣을 ●로 분류한다

②는 ♣을 ▲로 분류한다

③는 ♣을 ●로 분류한다

①, ②, ③ 중에 어떤 모델이 더 우수한 걸까?

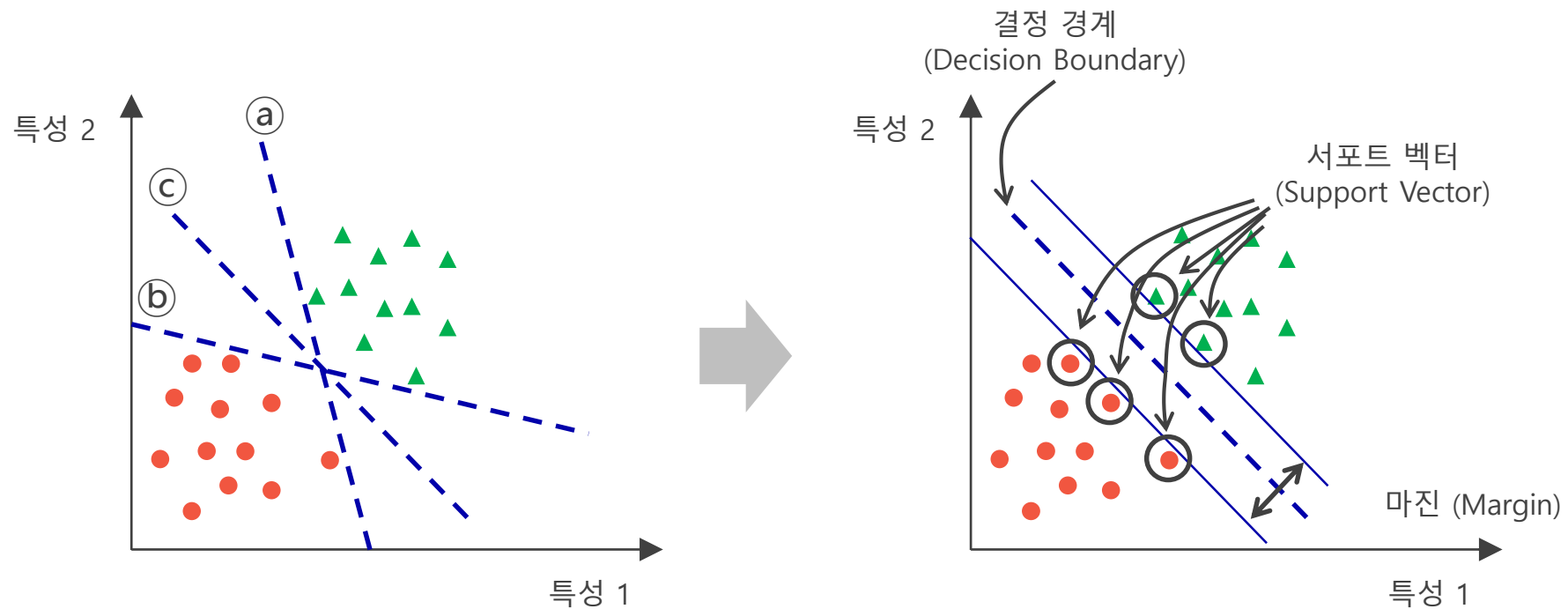
④ 서포트 벡터 머신 (Support Vector Machine, SVM) (9/12) 인공지능 활용 Python language



분류기 (Classifier)	시험 데이터 ★	시험 데이터 ♣	정확도
① a	● (오분류)	●	50%
② b	▲	▲ (오분류)	50%
③ c	▲	●	100%

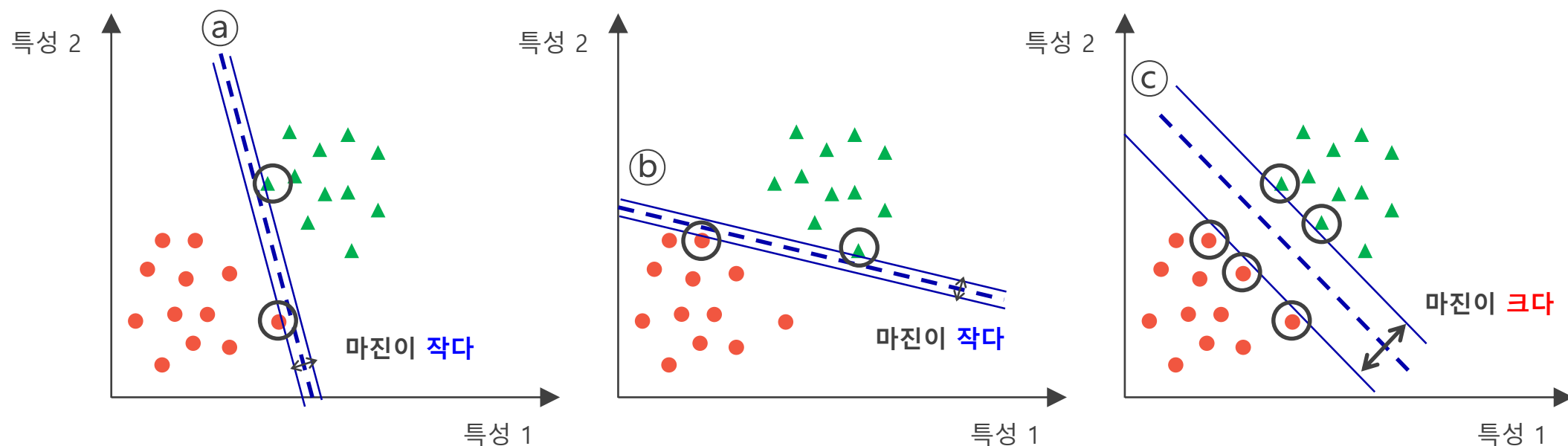
③가 가장 우수한 모델인 것 같다!

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (10/12) 인공지능 활용 Python language



- 결정 경계 (Decision Boundary): 분류를 위한 기준선
- 서포트 벡터 (Support Vector): 결정 경계와 가장 가까운 위치에 있는 데이터
- 마진 (Margin): 결정 경계와 서포트 벡터 사이의 거리

④ 서포트 벡터 머신 (Support Vector Machine, SVM) (11/12) 인공지능 활용 Python language



마진 (Margin)을 극대화 하는 선을 찾으면
새로운 데이터 (시험 데이터)가 들어오더라도 잘 분류 하겠구나!

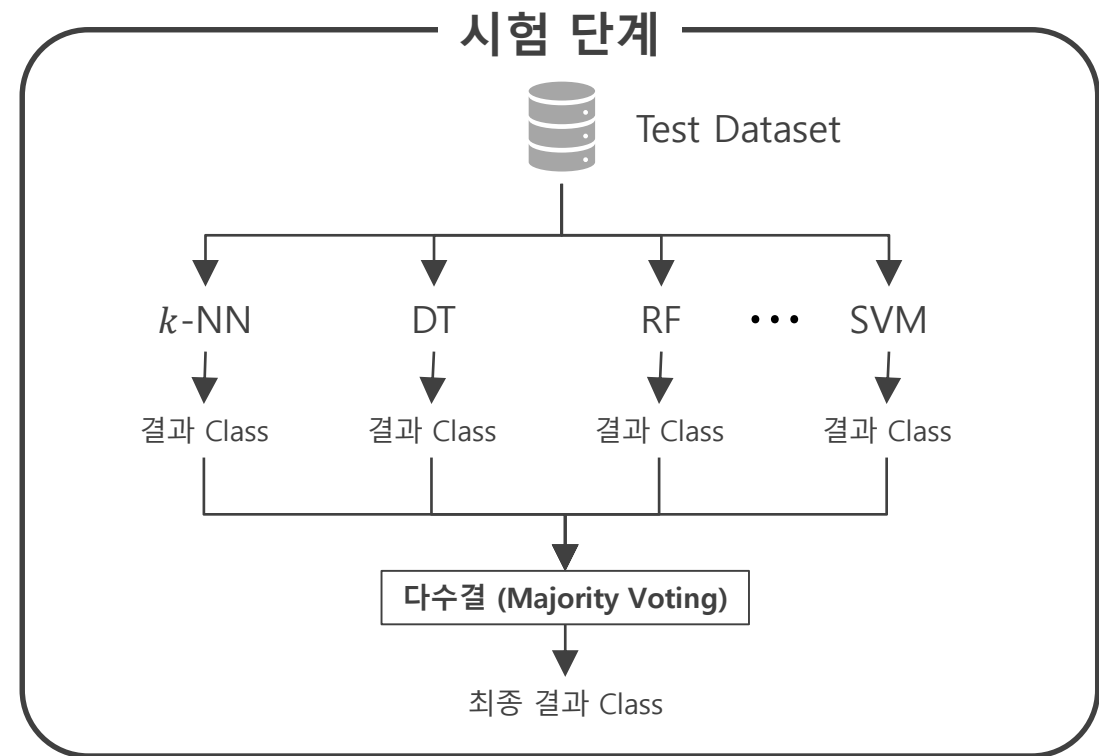
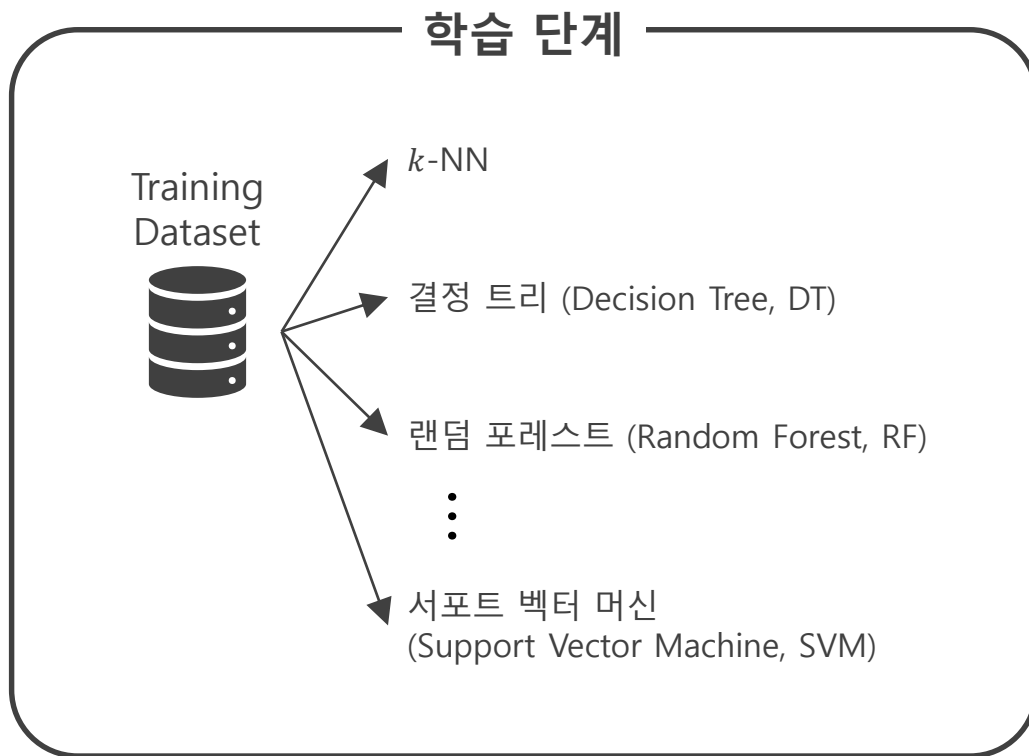
④ 서포트 벡터 머신 (Support Vector Machine, SVM) (12/12) 인공지능 활용 Python language

- 주어진 데이터가 어느 그룹에 속하는지 분류하는 모델
- 두 분류 (Classes) 사이의 여백을 의미하는 **마진 (Margin)**을 최대화하는 방향으로 데이터를 **분류**
- **SVM은 마진을 극대화하는 선을 찾아 분류하므로 마진이 크면 클수록 새로운 데이터가 들어오더라도 잘 분류할 가능성이 높아짐**
- SVM은 사용 방법이 쉽고 예측 정확도가 높다는 장점
- 하지만 모델 구축에 시간이 오래 걸리고 결과에 대한 설명력이 떨어지는 단점

⑤ 앙상블 학습 (Ensemble Learning)

인공지능 활용 Python language

- 학습 알고리즘들을 따로 쓰는 경우에 비해 더 좋은 성능을 얻기 위해 다수의 학습 알고리즘을 사용하는 방법



- 혼동 행렬 (Confusion Matrix)
- 분류 정확도 (Accuracy)

이진 분류 (Binary Classification) 문제를 가정

		학습 알고리즘의 답	
		Class 1	Class 2
실제 정답	Class 1	A (30)	B (20)
	Class 2	C (10)	D (40)

혼동 행렬 (Confusion Matrix)라고 부릅니다

$$\text{Accuracy} = \frac{A+B}{A+B+C+D} = \frac{30+40}{30+20+10+40} = \frac{70}{100} = 0.7$$

$$\text{Accuracy (\%)} = 0.7 \times 100 (\%) = 70\%$$

일반적으로 분류 정확도 (Accuracy)를 이용해서
학습 알고리즘의 우수성을 평가합니다

AI Experts
Who Lead
The Future

04

지도학습의 회귀 알고리즘

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

- **분류 (Classification)**

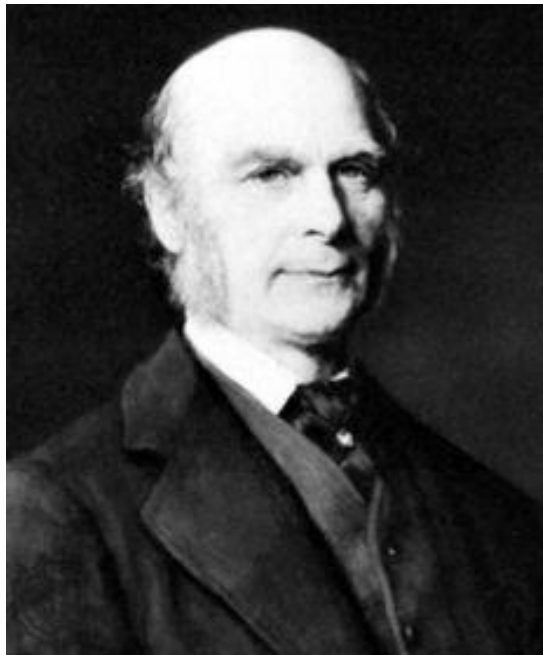
- 어떤 입력 데이터가 들어오더라도 학습에 사용한 레이블 (Label) 중 하나로 결과값을 결정
- 레이블 (Label)이 이산적인 (Discrete) 경우 (즉, [0, 1, 2, 3, ...]와 같이 유한한 경우)

- **회귀 (Regression)**

- 입력 데이터에 대한 결과값으로 학습에 사용한 레이블 이외의 값이 나올 수 있음
- 레이블 (Label)이 실수인 경우
 - 키 (Height) 정보가 주어졌을 때, 몸무게를 예측
 - 공부한 시간 정보가 주어졌을 때, 시험 성적 예측
 - 커피를 몇 잔 마셨는지에 대한 정보가 주어졌을 때, 수면 시간 예측

• 회귀 (Regression)

- 19세기, 통계학자이자 인류학자인 프랜시스 골턴 (Francis Galton)이 처음 사용



[사진출처] https://en.wikipedia.org/wiki/Francis_Galton

프랜시스 골턴

- 아버지와 자식의 키를 분석함
- 사람의 키 (Height)는 세대를 거듭할 수록 평균에 가까워지는 경향이 있다는 것을 발견
- 키가 큰 아버지의 자식은 아버지보다 키가 작고, 키가 작은 아버지의 자식은 아버지보다 키가 크다
- 즉, 세대를 거듭할 수록 큰 키는 작아지고, 작은 키는 커지고 평균에 수렴한다
- 이를 프랜시스 골턴은 "평균으로 돌아간다 (=회귀)"라고 표현함

변수 사이의 관계를 분석하는 방법을
역사적인 이유 때문에 "회귀 (Regression)" 라고 부르
게 되었다!

회귀 (Regression) 문제를 해결하기 위한 기법

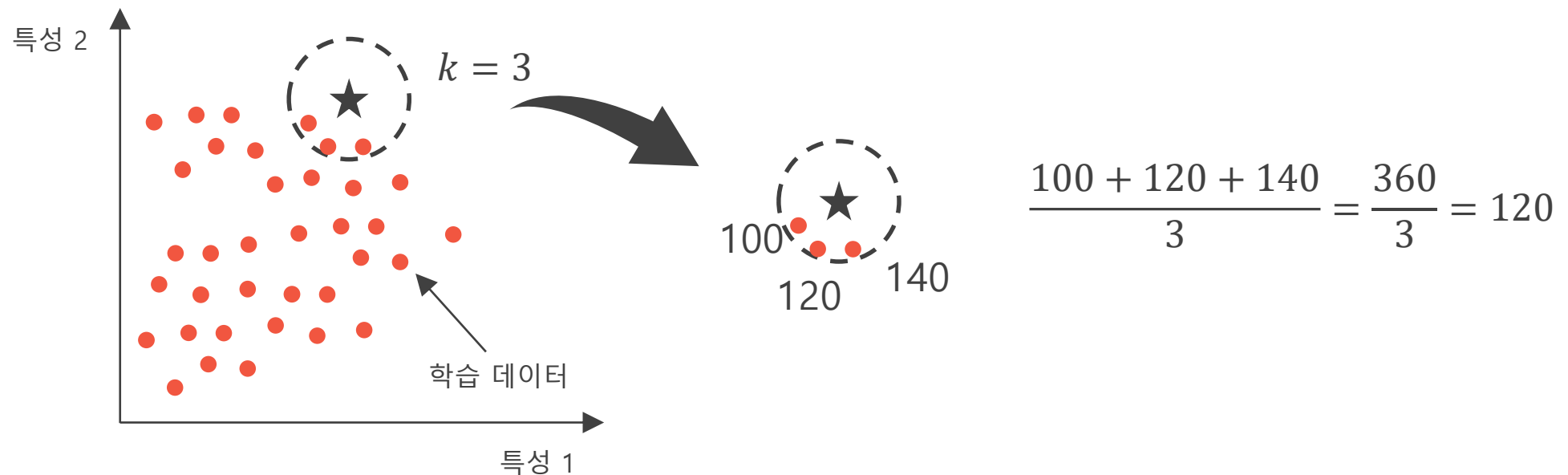
인공지능 활용 Python language

- ① k -최근접 이웃 (k -Nearest Neighbors, k -NN)
- ② 결정 트리 (Decision Tree)
- ③ 선형 회귀 (Linear Regression)
- ④ 다항 회귀 (Polynomial Regression)

① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (1/2)

인공지능 활용 Python language

- 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k 개의 학습 데이터의 Labels의 평균 값을 시험 데이터의 Label 값으로 결정하는 알고리즘
- 예를 들어,
 - $k = 3$ 일 때, 시험 데이터 ★에 대한 결과 값은 가장 가까운 3개의 학습 데이터의 평균 값 120으로 결정



① k -최근접 이웃 (k -Nearest Neighbors, k -NN) (2/2)

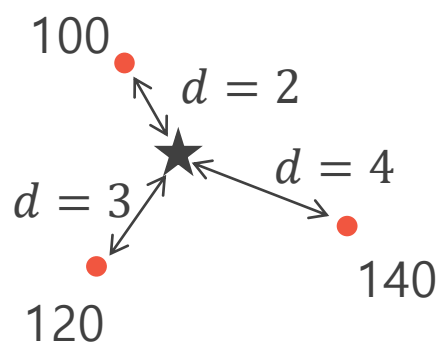
인공지능 활용 Python language

가중 회귀 (Weighted Regression)

- 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k 개의 학습 데이터를 찾음
- 단순히 학습데이터의 Labels의 평균 값을 계산하는 것이 아니라,
 - 시험 데이터와 학습 데이터 사이의 거리를 고려하여 가중합으로 Label 값으로 결정

예를 들어,

- $k = 3$ 일 때, 시험 데이터 ★에 대한 가중 회귀 결과값은 115.4로 결정



Labels의 평균 값: $\frac{100 + 120 + 140}{3} = \frac{360}{3} = 120$

가중 회귀 결과값: $\frac{\frac{100}{2} + \frac{120}{3} + \frac{140}{4}}{\frac{1}{2} + \frac{1}{3} + \frac{1}{4}} = \frac{\frac{600}{12} + \frac{480}{12} + \frac{420}{12}}{\frac{13}{12}} = \frac{1500}{13} \approx 115.4$

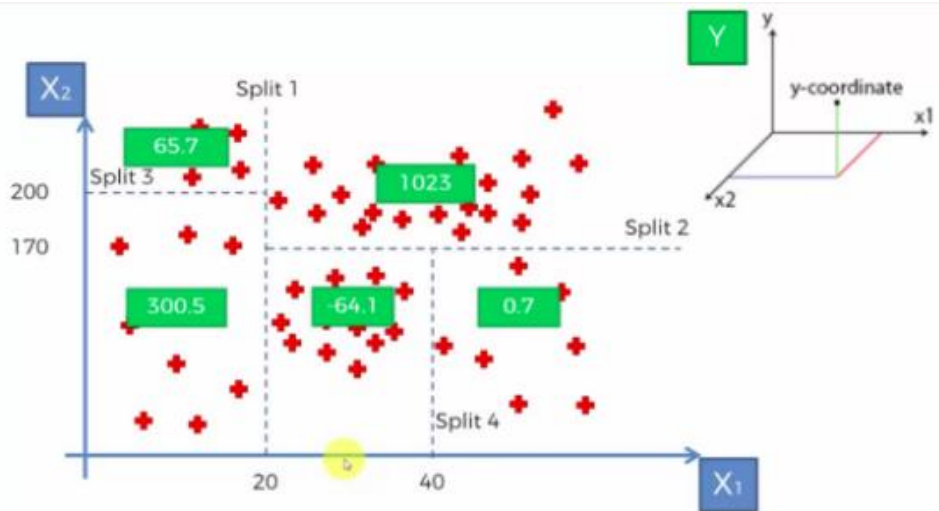
시험 데이터와 멀리 떨어져 있는
학습 데이터일수록 영향력을 줄이겠다!

② 결정 트리 (Decision Tree) (1/2)

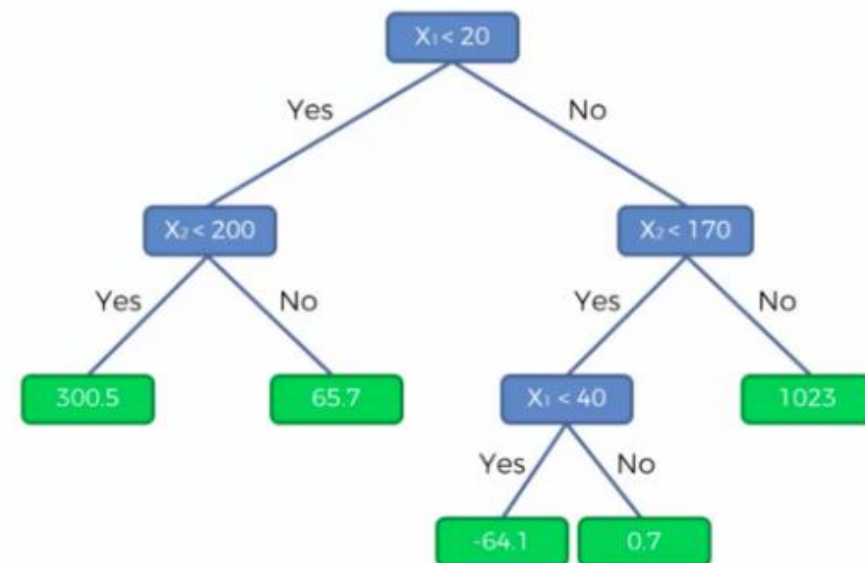
인공지능 활용 Python language

- 학습 데이터로부터 결정 트리를 생성하고, 각 끝 마디 (Terminal Node or Leaf Node)에서 해당 영역의 평균 값을 계산

Y: 해당 영역의 평균 값



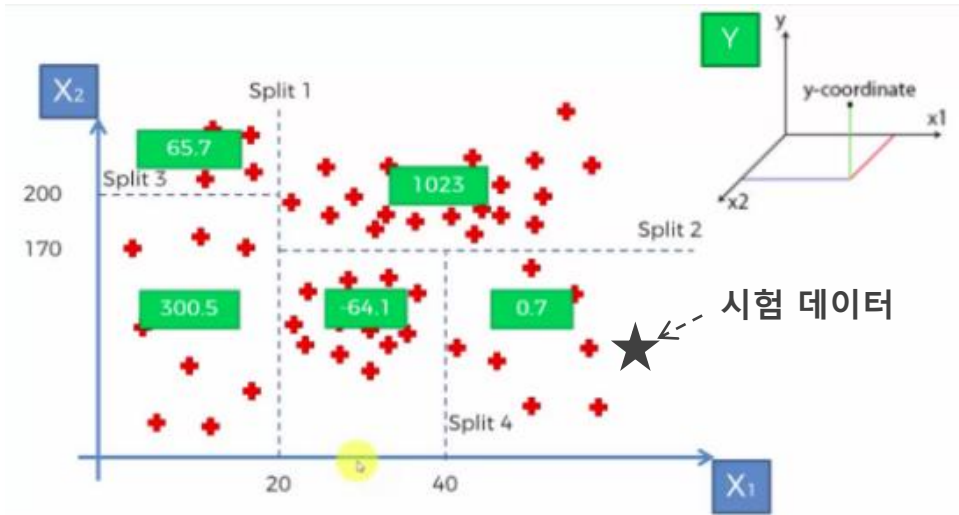
[사진출처] <https://riverzayden.tistory.com/6>



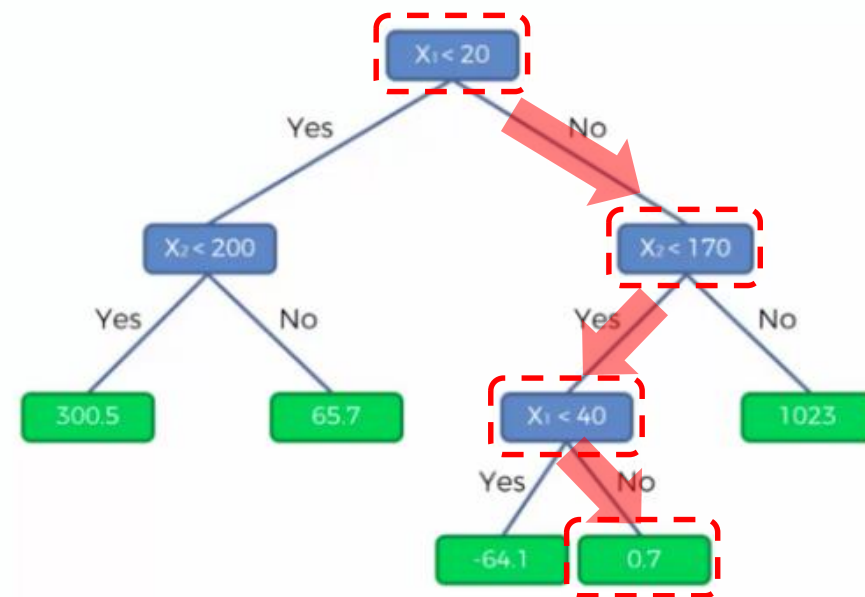
② 결정 트리 (Decision Tree) (2/2)

- 시험 데이터 (Test Data)가 주어졌을 때, 시험 데이터가 속한 영역의 평균 값으로 회귀 결과값을 결정하는 알고리즘

Y: 해당 영역의 평균 값



[사진출처] <https://riverzayden.tistory.com/6>

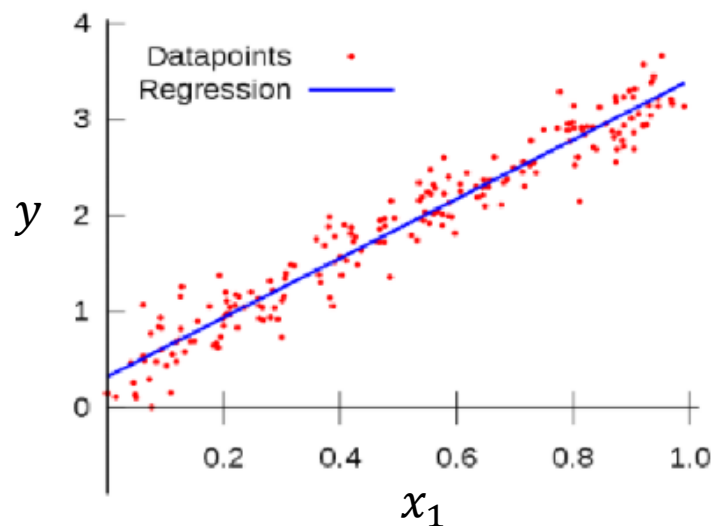


시험 데이터 ★의 회귀 결과값: 0.7

③ 선형 회귀 (Linear Regression) (1/4)

인공지능 활용 Python language

- 학습 데이터 (Training Data)로부터 종속 변수 y 와 한 개 이상의 독립 변수 x 와의 선형 상관 관계를 모델링하는 방법
(쉽게 이해하면, 학습 데이터를 잘 표현하는 직선 하나를 찾아내겠다는 의미)



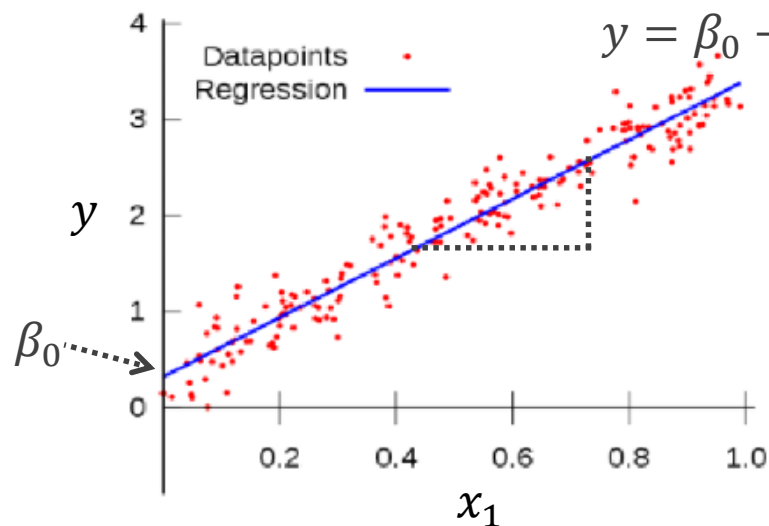
[사진출처] <https://bangu4.tistory.com/100>

빨간색 점들이 학습 데이터이고,
이 데이터들을 잘 표현하는 파란색 직선을
찾아내는 것이 "선형 회귀"가 하는 역할임

③ 선형 회귀 (Linear Regression) (2/4)

인공지능 활용 Python language

- 독립 변수 x 의 개수에 따라 선형 회귀는 아래와 같이 분류됨
 - 단순 선형 회귀 (Simple Linear Regression): 독립 변수 x 의 개수가 1개
 - 다중 선형 회귀 (Multiple Linear Regression): 독립 변수 x 의 개수가 2개 이상



[사진출처] <https://bangu4.tistory.com/100>

왼쪽 예제는 독립 변수의 개수가 1개인 경우임

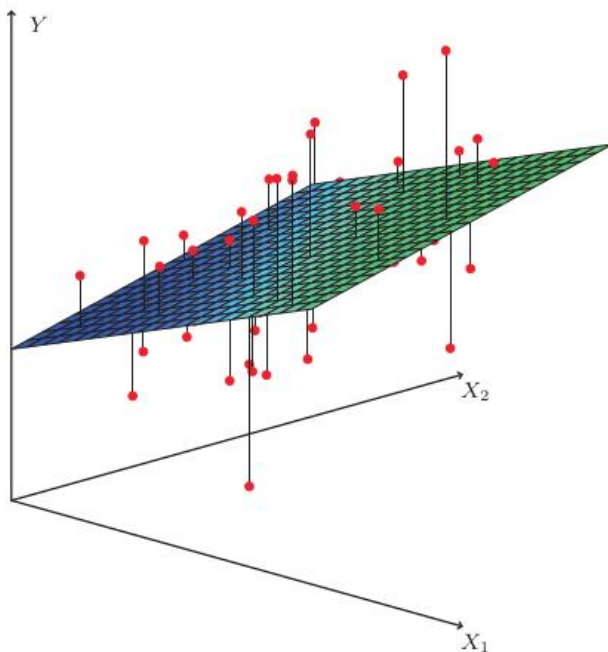
$$y = \beta_0 + \beta_1 \times x_1$$

위 직선에서 기울기 ($= \beta_1$) 값과 y절편 ($= \beta_0$) 값을 찾는 것이 선형 회귀 알고리즘에서 수행하는 동작

③ 선형 회귀 (Linear Regression) (3/4)

인공지능 활용 Python language

- 독립 변수 x 의 개수에 따라 선형 회귀는 아래와 같이 분류됨
 - 단순 선형 회귀 (Simple Linear Regression): 독립 변수 x 의 개수가 1개
 - 다중 선형 회귀 (Multiple Linear Regression): 독립 변수 x 의 개수가 2개 이상



왼쪽 예제는 독립 변수의 개수가 2개 (x_1, x_2)인 경우임

$$y = \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_2$$

선형 회귀 알고리즘 수행을 통해서,
학습 데이터를 가장 잘 표현 하는 $\beta_0, \beta_1, \beta_2$ 값을 찾아냄

[사진출처] <https://p829911.github.io/2020/01/16/3.2.1/>

③ 선형 회귀 (Linear Regression) (4/4)

- 독립 변수 x 의 개수에 따라 선형 회귀는 아래와 같이 분류됨
 - 단순 선형 회귀 (Simple Linear Regression): 독립 변수 x 의 개수가 1개
 - 다중 선형 회귀 (Multiple Linear Regression): 독립 변수 x 의 개수가 2개 이상

독립 변수의 개수가 3개를 넘어가게 되면, 시각적으로 표현하기가 어려워 짐

$$\begin{aligned}
 y &= \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_2 + \cdots + \beta_i \times x_i + \cdots + \beta_n \times x_n \\
 &= \beta_0 + \sum_{i=1}^n (\beta_i \times x_i)
 \end{aligned}$$

이 경우에도 선형 회귀 알고리즘 수행을 통해서,

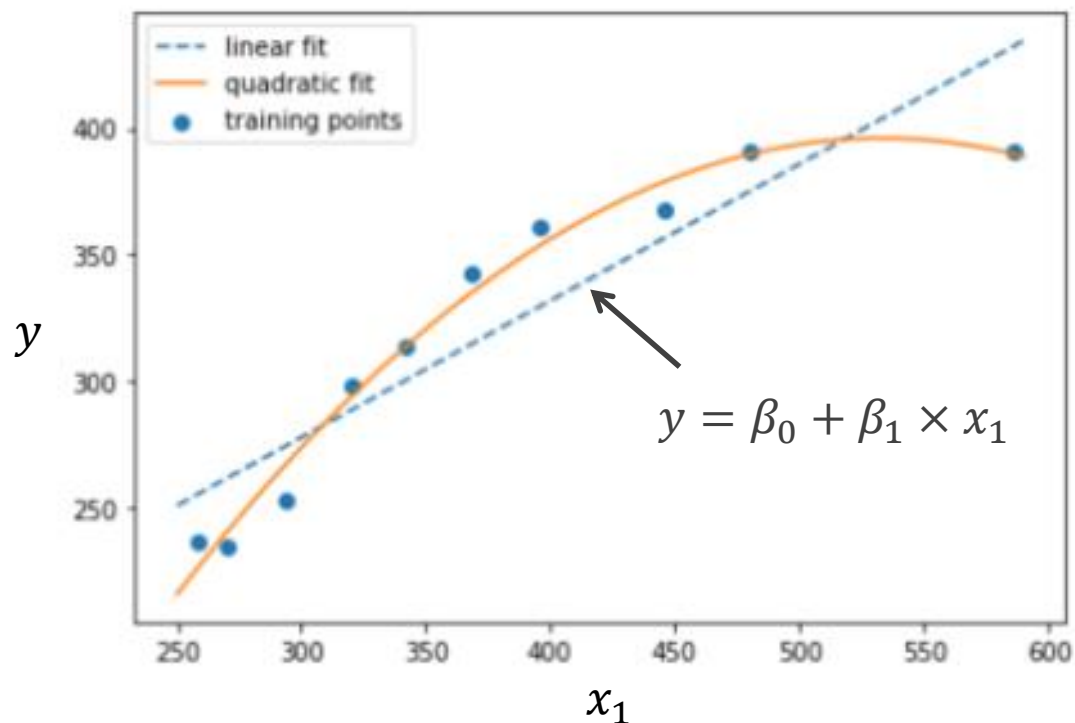
학습 데이터를 가장 잘 표현 하는 $\beta_0, \beta_1, \cdots, \beta_n$ 값을 찾아냄

④ 다항 회귀 (Polynomial Regression) (1/2)

인공지능 활용 Python language

• 선형 회귀의 단점

- 학습 데이터 내, 종속 변수 y 와 독립 변수 x 사이의 상관 관계가 선형이 아닐 수 있다



[사진출처] <https://hongl.tistory.com/128>

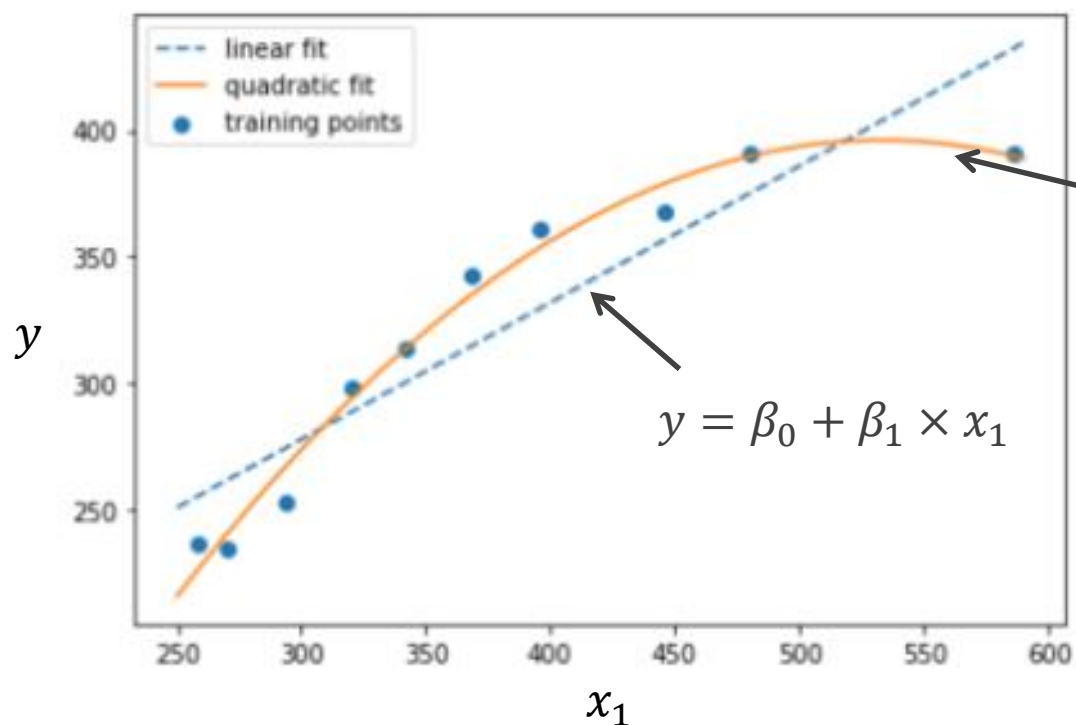
학습 데이터 ●가 선형 회귀 모형 (점선)으로 잘 표현되지 않고 있다 (오차가 크다)

오히려 주황색 실선이 학습 데이터 ●를 잘 표현하고 있다 (오차가 작다)

④ 다항 회귀 (Polynomial Regression) (2/2)

인공지능 활용 Python language

- 각 독립 변수 x 에 대한 고차원의 다항식을 이용하여 종속 변수 y 의 관계를 비선형적 (Non-linear)으로 모델링하는 방법



[사진출처] <https://hongl.tistory.com/128>

왼쪽 예제는 독립 변수의 개수가 1개인 경우임

$$y = \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_1^2 + \dots + \beta_n \times x_1^n$$

다항 회귀 알고리즘 수행을 통해서,
학습 데이터를 가장 잘 표현 하는
 $\beta_0, \beta_1, \dots, \beta_n$ 값을 찾아냄