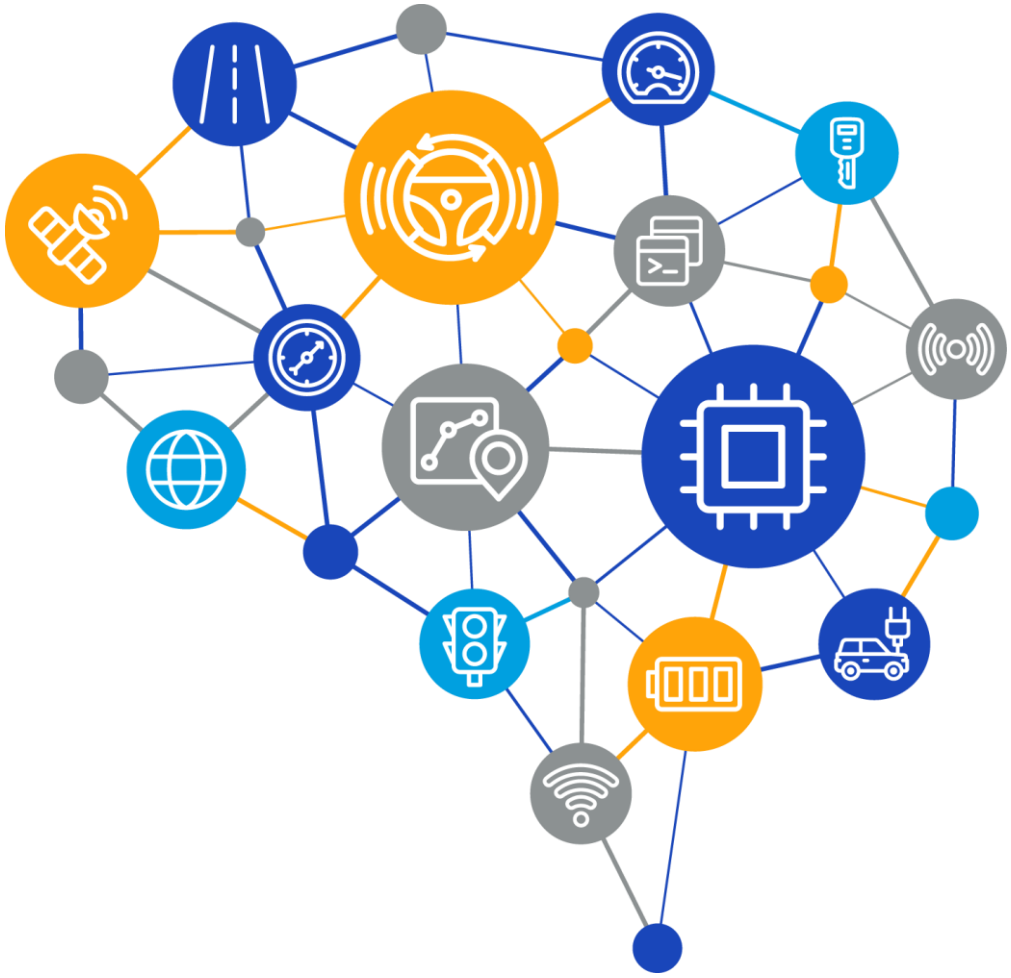
머신러닝
지도학습 기법



Machine Learning
Supervisor learning

2024.11

강환수 교수

DMUAi 동양미래대학교
인공지능소프트웨어학과

AI Experts Who Lead
The Future

AI Experts
Who Lead
The Future

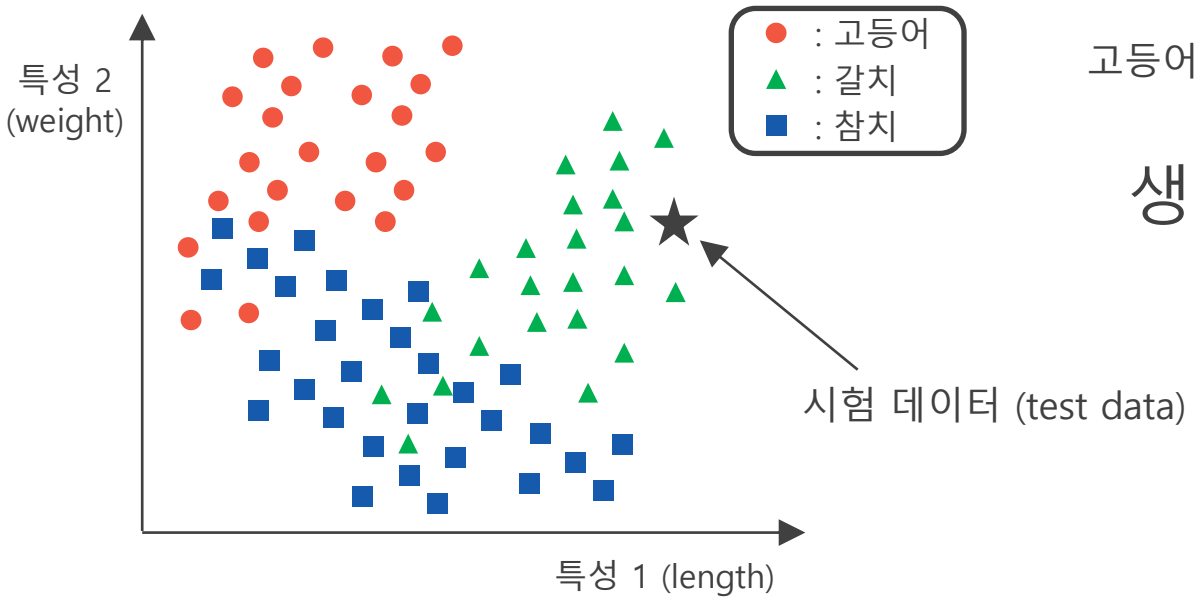
01

지도학습의 분류 알고리즘

분류 (Classification) 문제를 해결하기 위한 기법

- 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



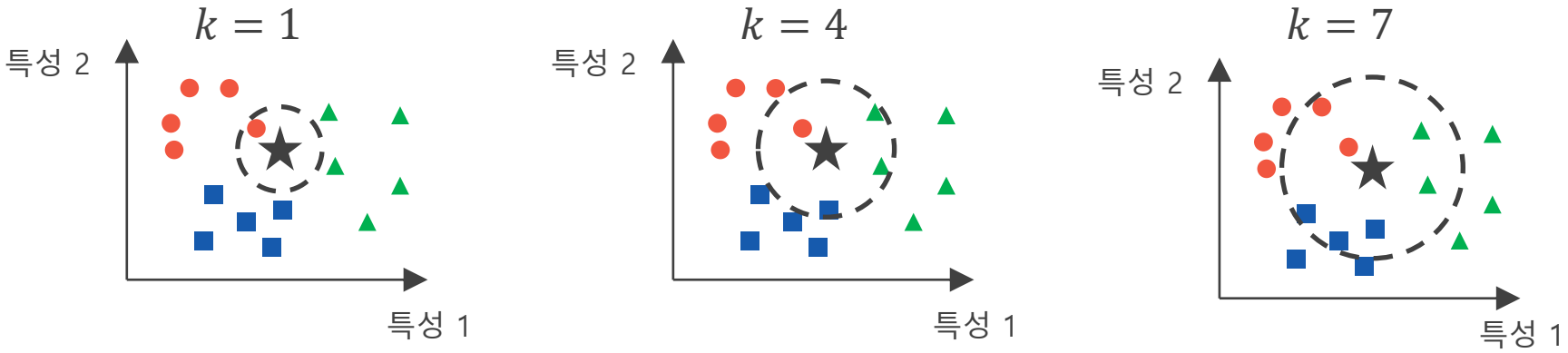
수산시장에서 일하고 있는 동양이는
고등어, 갈치, 참치를 분류하는 장비를 개발하려고 한다.

생선 ★의 Label은 무엇일까?

★ 주변에 갈치 ▲가 많은데...

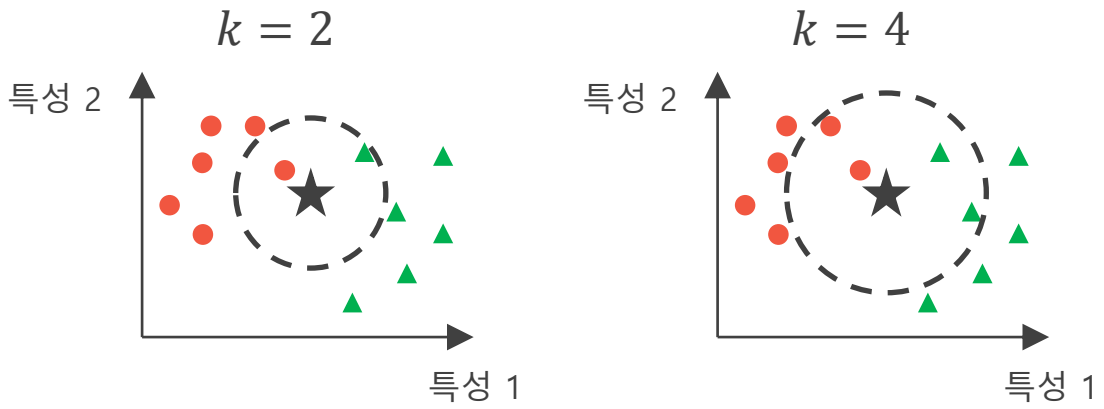
① k-최근접 이웃 (k-Nearest Neighbors, k-NN) (2/10)

- ① k-최근접 이웃 (k-Nearest Neighbors, k-NN) (2/10)
  - 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k개의 학습 데이터의 Labels을 참조하여 시험 데이터가 어떤 Label에 속하는지 분류하는 알고리즘
  - 예를 들어,
    - $k = 1$일 때, 시험 데이터는 ● 빨간색 원으로 분류됨
    - $k = 4$일 때, 시험 데이터는 ▲ 초록색 삼각형으로 분류됨
    - $k = 7$일 때, 시험 데이터는 ■ 파란색 사각형으로 분류됨



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

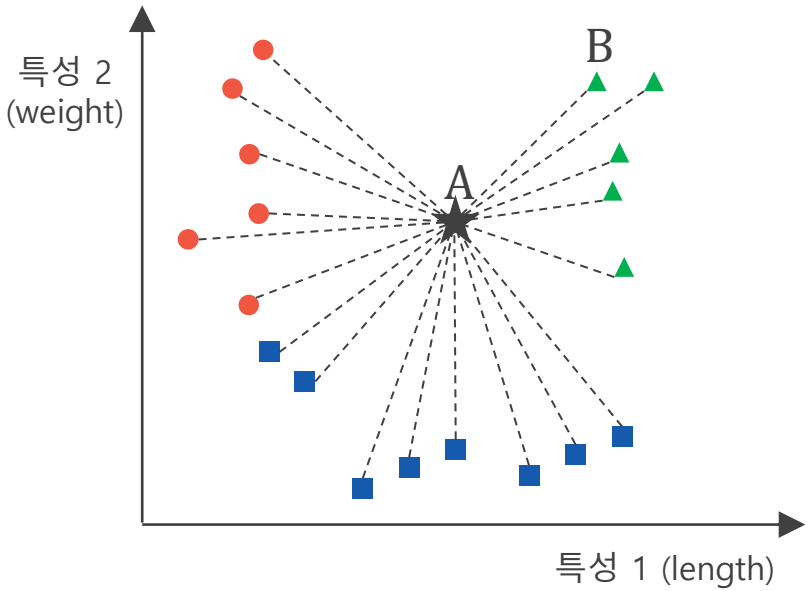
- ① k-최근접 이웃 (k-Nearest Neighbors, k-NN) (3/10)
  - 어떤 값이 최적의 k인지 불분명하기 때문에 데이터의 특성에 맞게 k 값을 임의로 선정해야 하는 단점이 있음
  - 일반적으로 이진 분류(Binary Classification) 문제에서는 k 값을 홀수로 선정함



이진 분류 문제에서 k 값이 짝수인 경우, ★을 분류할 수 없다

① k-최근접 이웃 (k-Nearest Neighbors, k-NN) (4/10)

- **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

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

- 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 계산

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

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

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 확인

Class	거리 (d)
0	**1.14**
0	**0.81**
0	**0.41**

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

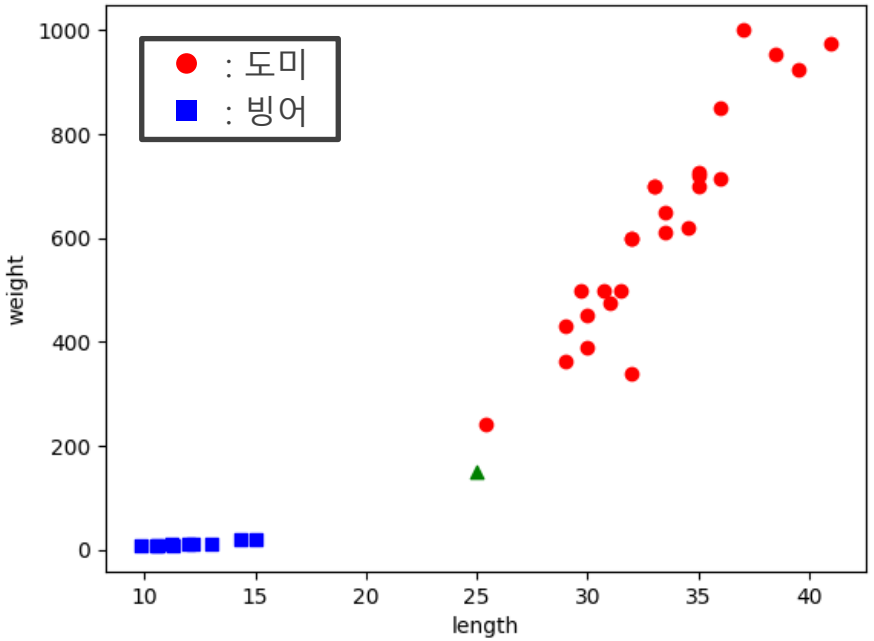
시험 데이터
(Test Data)

(특성 1, 특성 2) = (24.3, 3.4)

→ Class 0 (갈치)로 분류!

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

- (주의 사항) 특성들 사이에 스케일 (Scale)을 맞춰야 한다



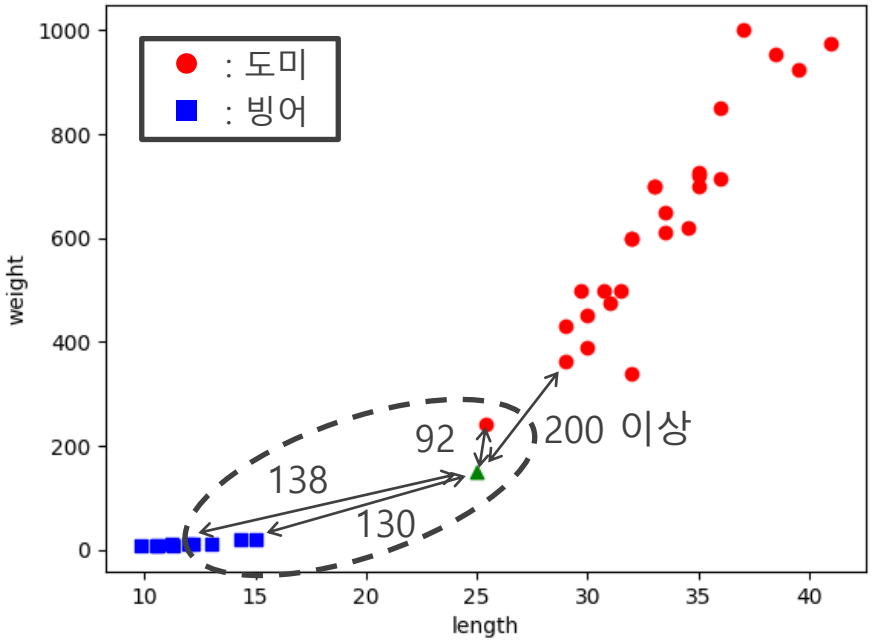
$k = 5$라고 가정해 보자!

생선 ▲의 Label은 무엇일까?

●랑 더 가까운 것 같은데...

① k-최근접 이웃 (k-Nearest Neighbors, k-NN) (9/10)

- (주의 사항) 특성들 사이에 스케일(Scale)을 맞춰야 한다



$k = 5$라고 가정해 보자!

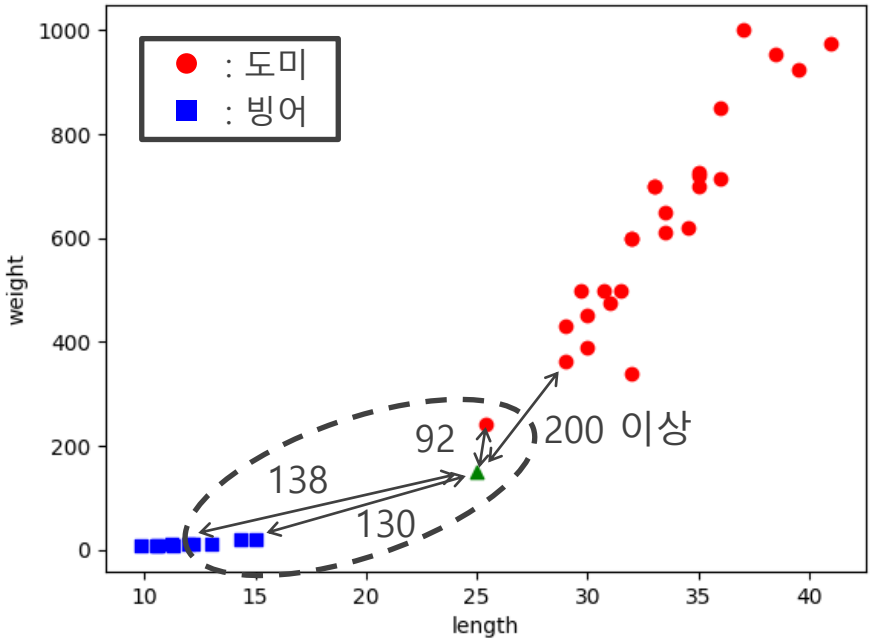
▲랑 가까운 5개의 데이터를 표시해 보면?!

■ (빙어) 4개, ● (도미) 1개로 ▲는 빙어로 분류된다?!

왜 그럴까?!

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

- **(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: $x_{new} = \dfrac{x - x_{min}}{x_{max} - x_{min}}$

② 결정 트리 (Decision Tree) (1/8)

- 의사결정 규칙을 나무 형태로 분류하는 분석 방법
- 다음 그림과 같이 상위 노드에서 시작하여 분류 기준값에 따라 하위 노드로 확장하는 방식이 “나무”를 닮았다고 하여 “의사 결정 나무”라고도 불림

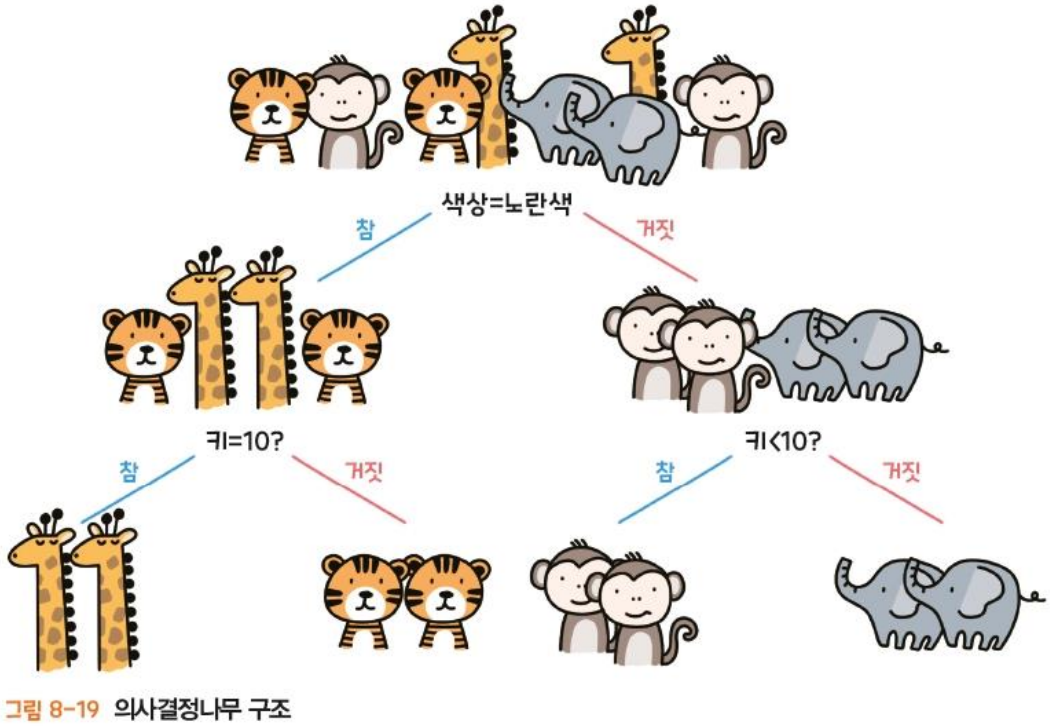


그림 8-19 의사결정나무 구조

② 결정 트리(Decision Tree) (2/8)

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

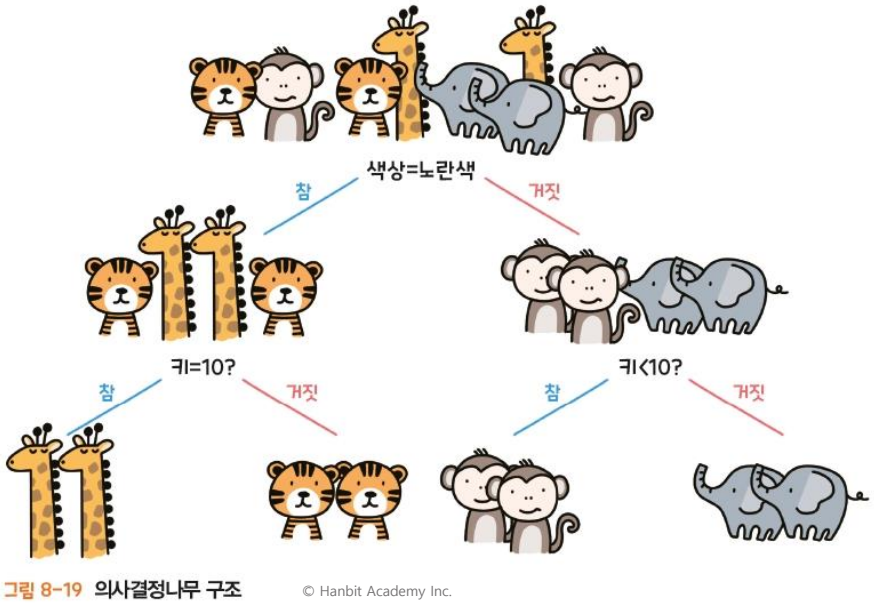
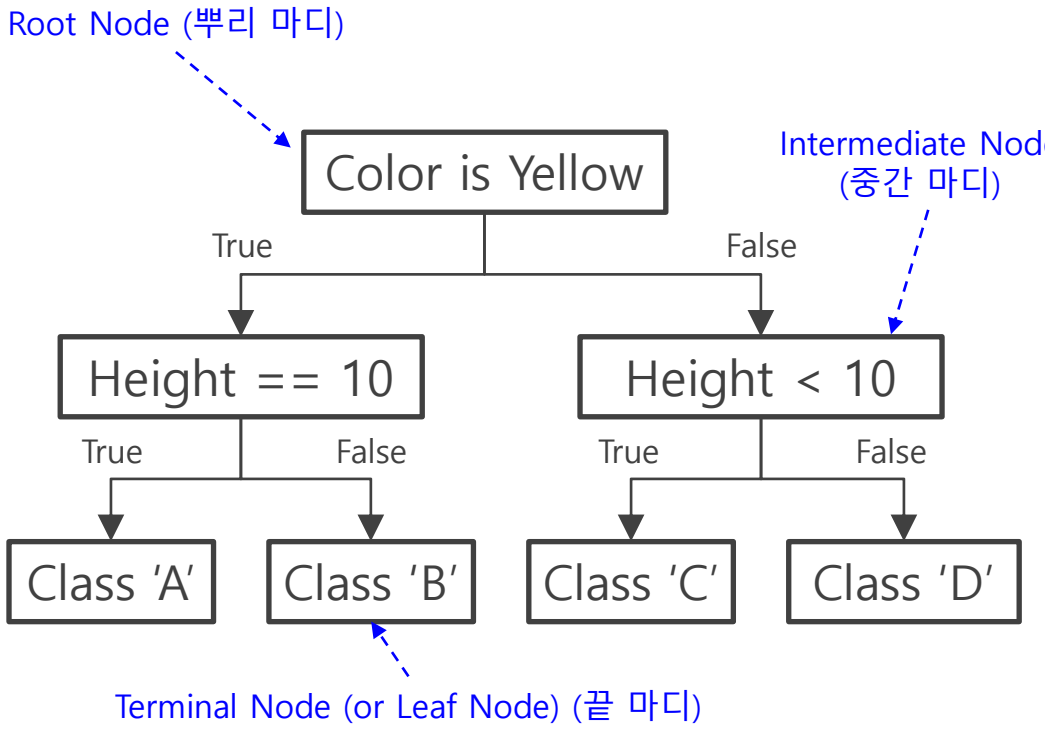
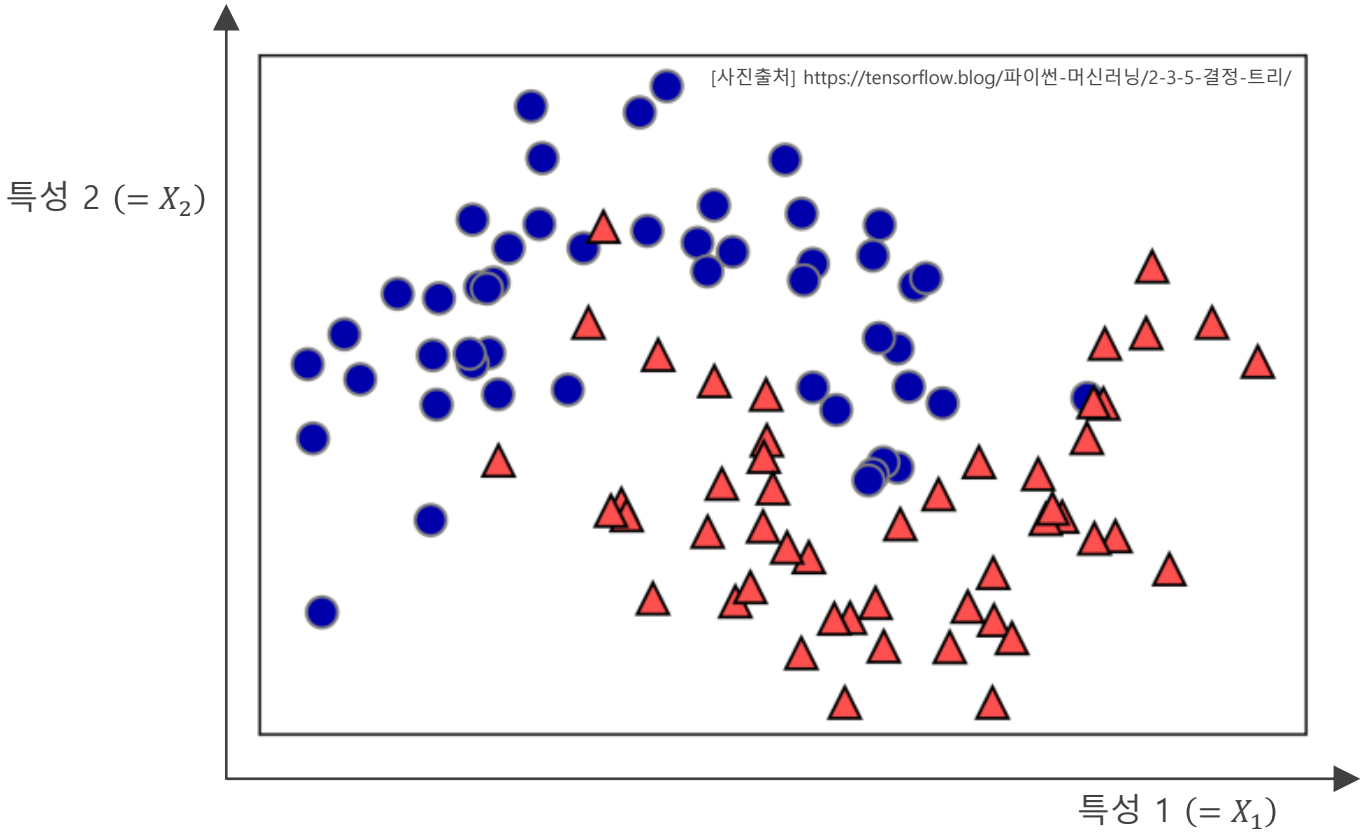


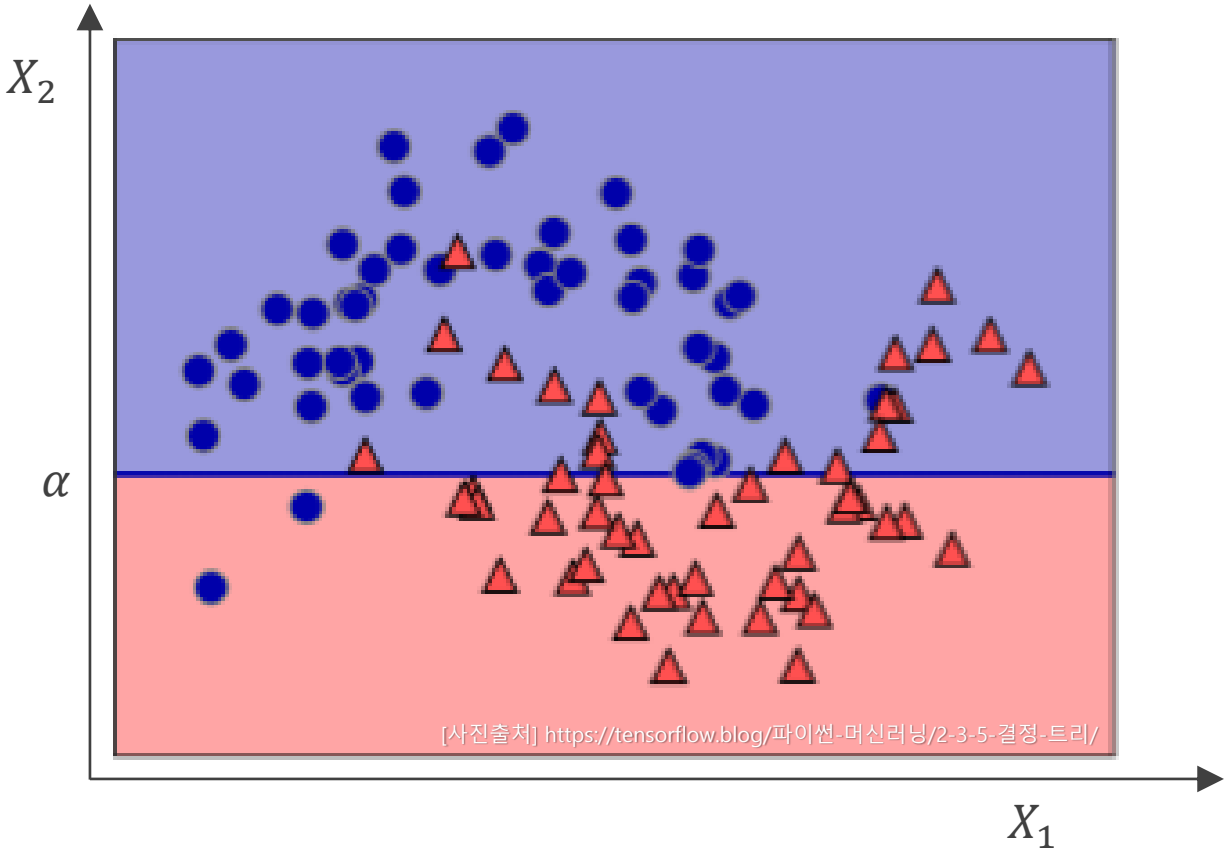
그림 8-19 의사결정나무 구조



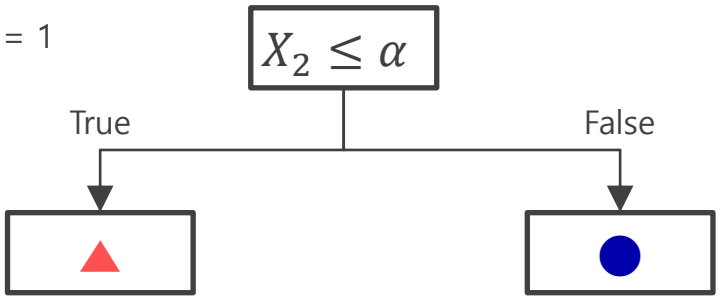
② 결정 트리(Decision Tree) (3/8)



② 결정 트리(Decision Tree) (4/8)

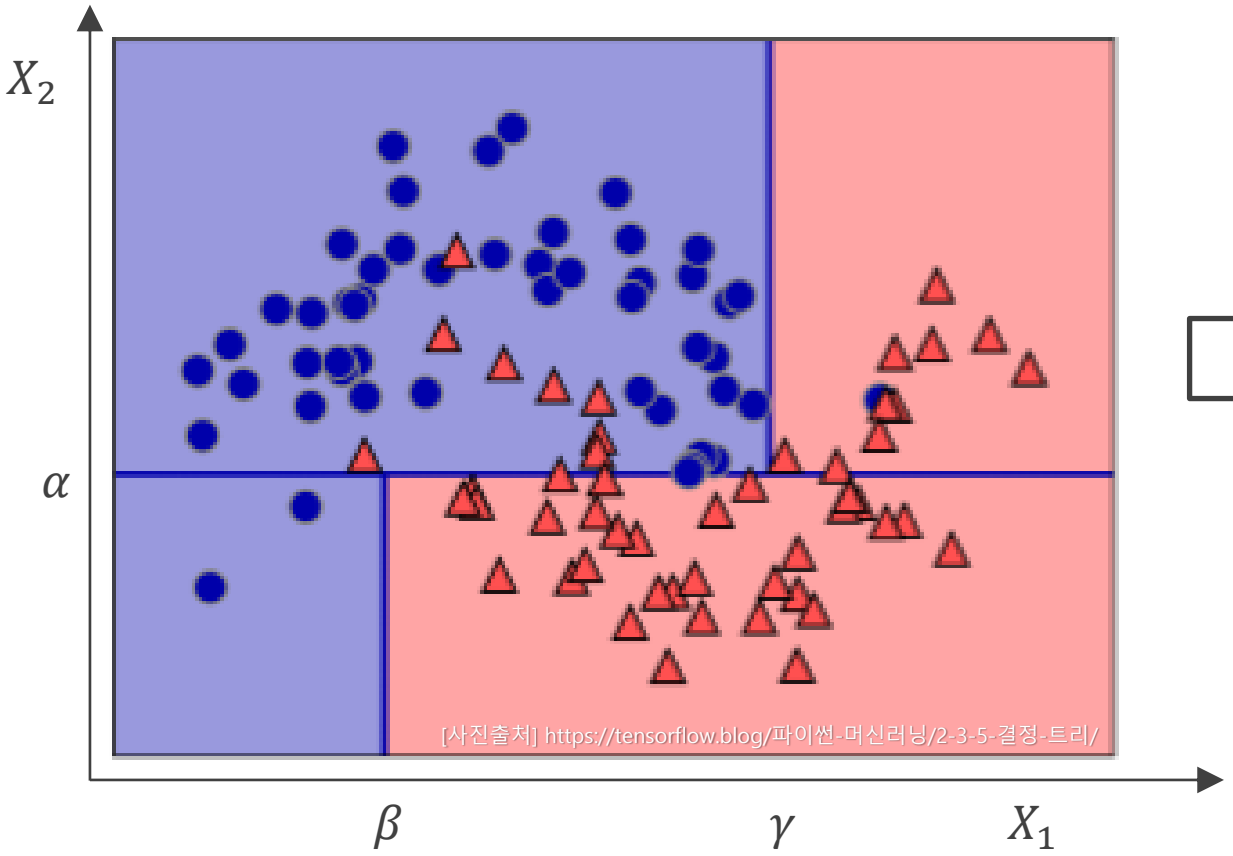


깊이 (Depth) = 1

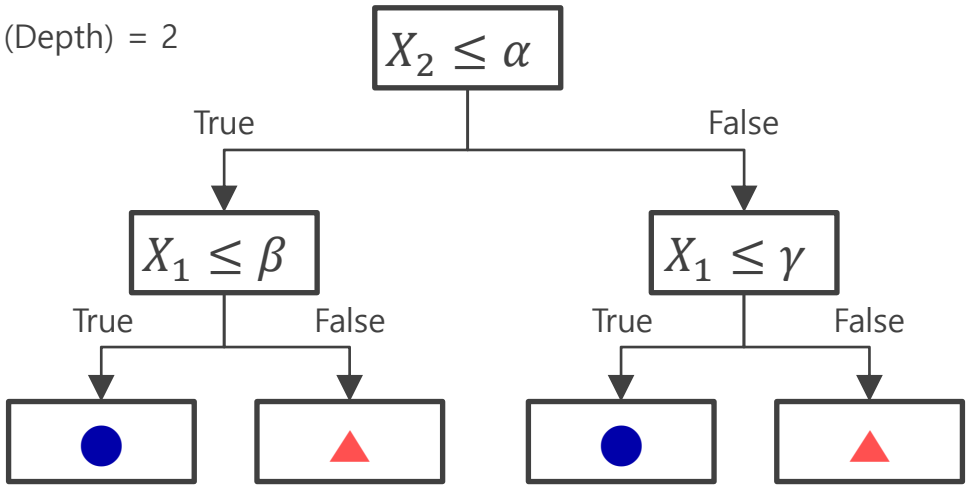


② 결정 트리(Decision Tree) (5/8)

- ② 결정 트리 (Decision Tree) (5/8)

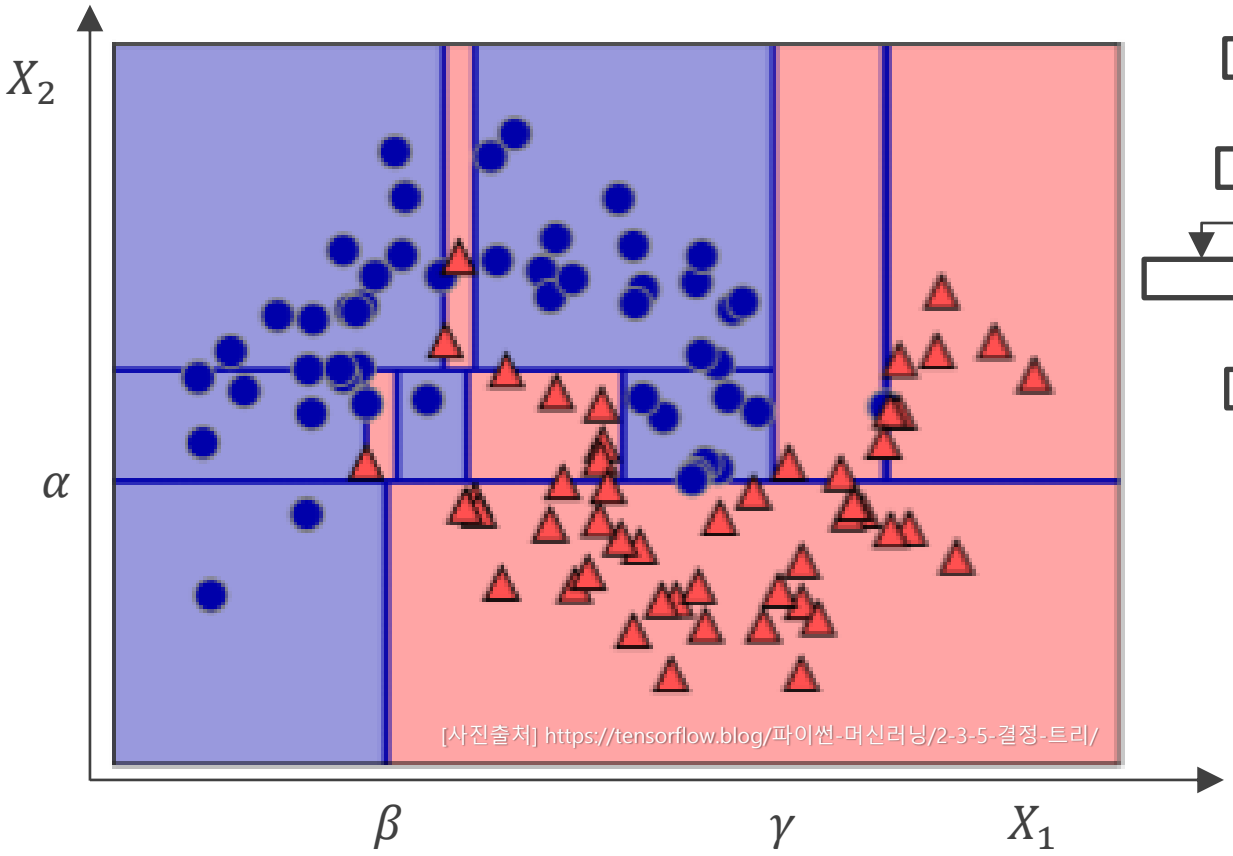


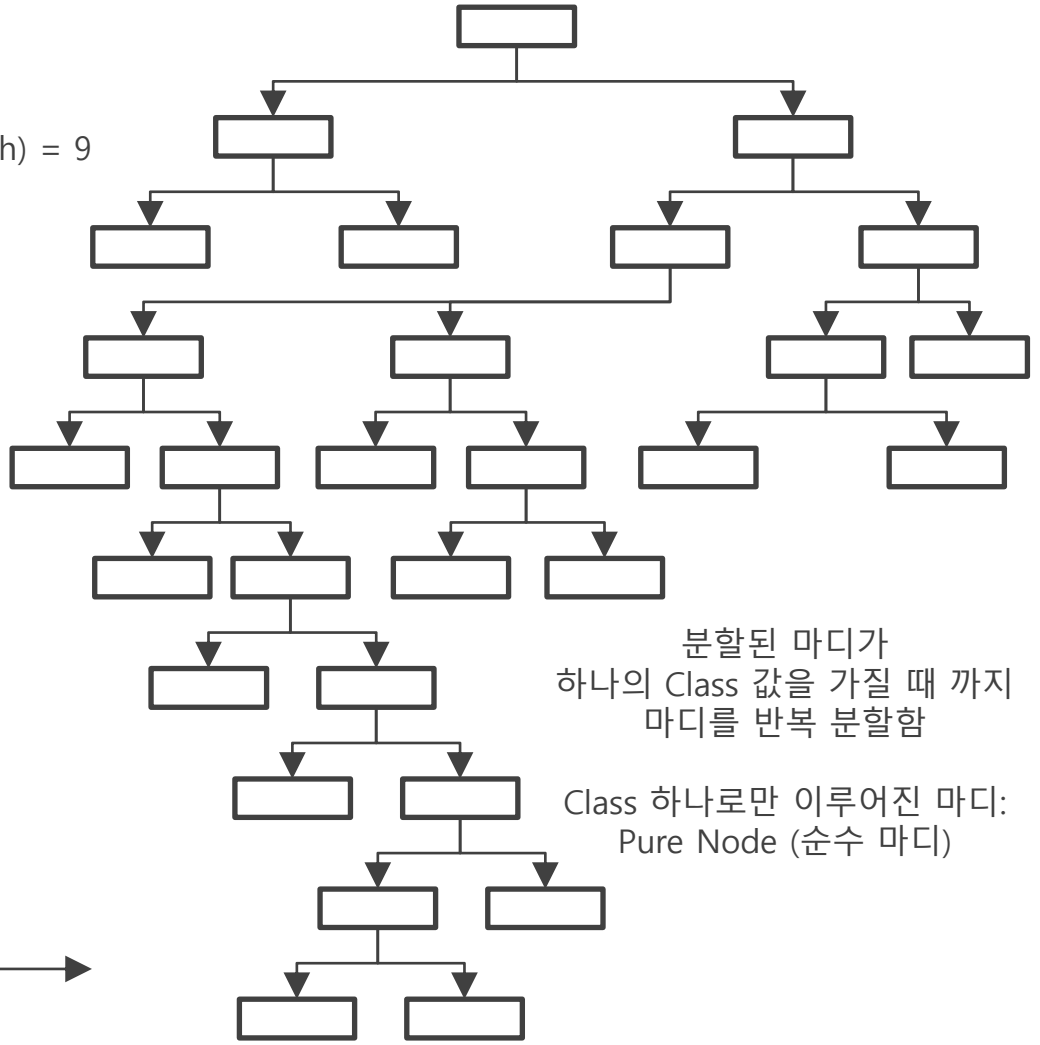
깊이 (Depth) = 2



② 결정 트리(Decision Tree) (6/8)

- ② 결정 트리 (Decision Tree) (6/8)





② 결정 트리(Decision Tree) (7/8)

- 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 복잡해지고 훈련 데이터 (Training Data)에 과대적합 (Overfitting)하게 됨
  - 해결 책으로 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)

- 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

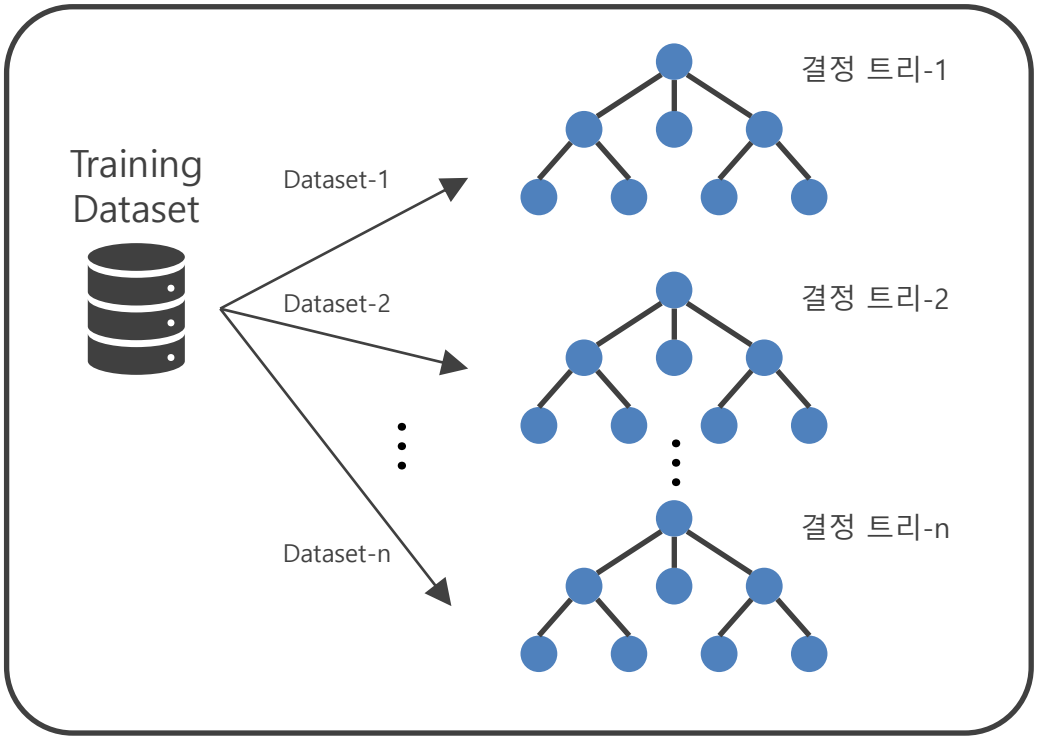
③ 랜덤 포레스트 (Random Forest) (1/3)

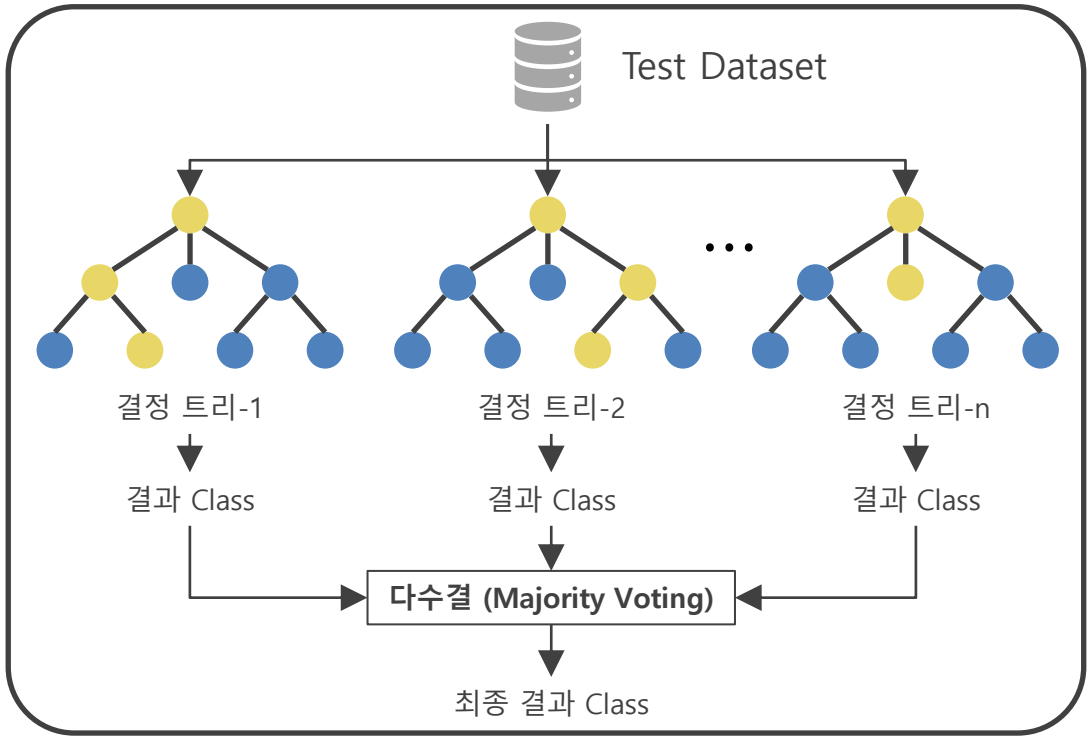
- **③ 랜덤 포레스트 (Random Forest) (1/3)**
  - 앙상블 (Ensemble) 학습 방법의 일종으로, 훈련 (Training) 과정에서 구성한 다수의 결정 트리 (Decision Tree)로부터 출력된 분류 결과로부터 다수결의 원칙에 따라 최종 분류 결과를 결정하는 방식으로 동작

나무 (Tree)가 모여서 숲 (Forest)을 이룬다

③ 랜덤 포레스트 (Random Forest) (2/3)

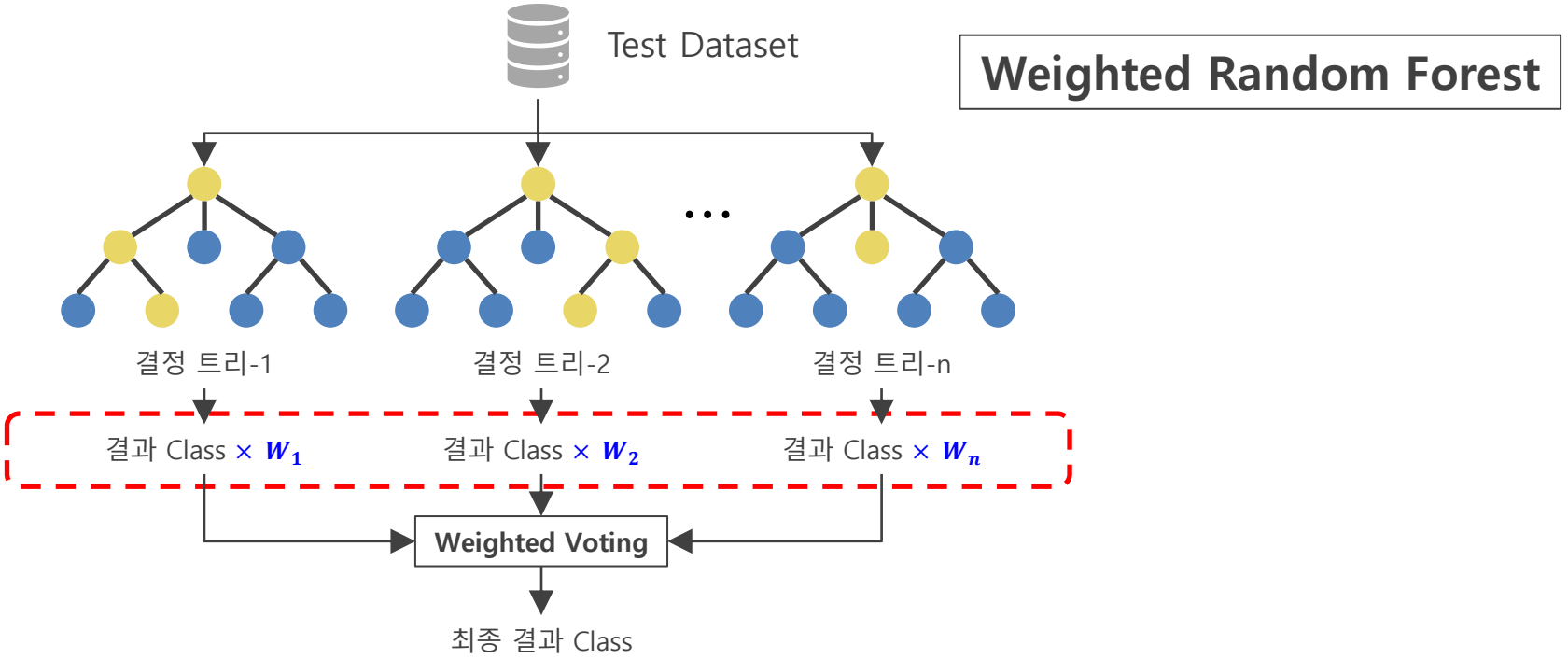
- **앙상블 (Ensemble) 학습 방법의 일종으로,**
  - 훈련 (Training) 과정에서 구성한 다수의 결정 트리 (Decision Tree)로부터 출력된 분류 결과로부터
    - 다수결의 원칙에 따라 최종 분류 결과를 결정하는 방식으로 동작





③ 랜덤 포레스트 (Random Forest) (3/3)

- **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



AI Experts
Who Lead
The Future

02

지도학습의 회귀 알고리즘

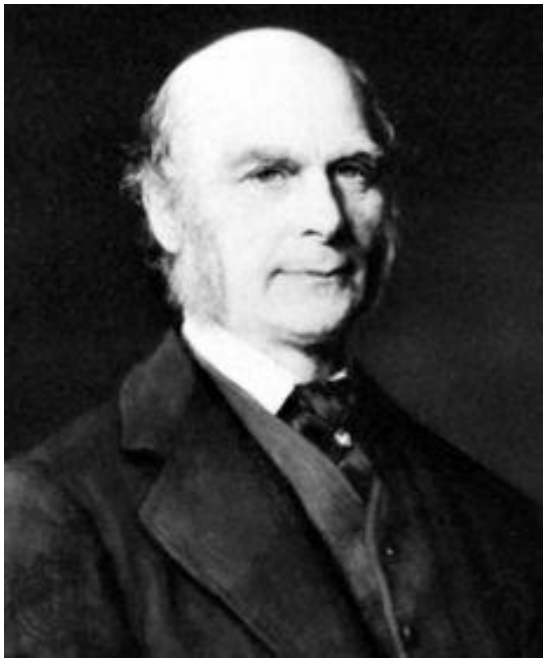
회귀와 분류

- **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 개요

- **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

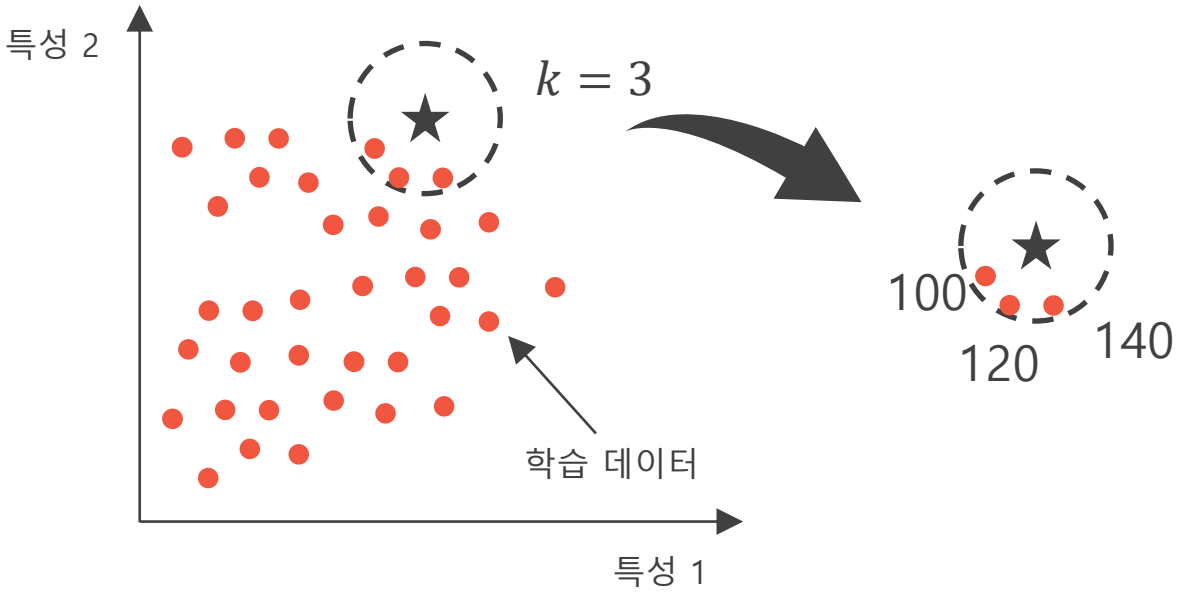
변수 사이의 관계를 분석하는 방법을
역사적인 이유 때문에 "회귀 (Regression)" 라고 부르게 되었다!

회귀 (Regression) 문제를 해결하기 위한 기법

- 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)

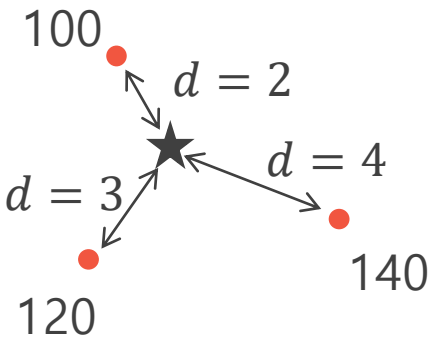
- 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k개의 학습 데이터의 Labels의 평균 값을 시험 데이터의 Label 값으로 결정하는 알고리즘
- 예를 들어,
  - $k = 3$일 때, 시험 데이터 ★에 대한 결과 값은 가장 가까운 3개의 학습 데이터의 평균 값 120으로 결정



$$\frac{100 + 120 + 140}{3} = \frac{360}{3} = 120$$

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

- **가중 회귀 (Weighted Regression)**
  - 시험 데이터가 주어졌을 때, 시험 데이터로부터 거리가 가장 가까운 k개의 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



시험 데이터와 멀리 떨어져 있는
학습 데이터일수록 영향력을 줄이겠다!

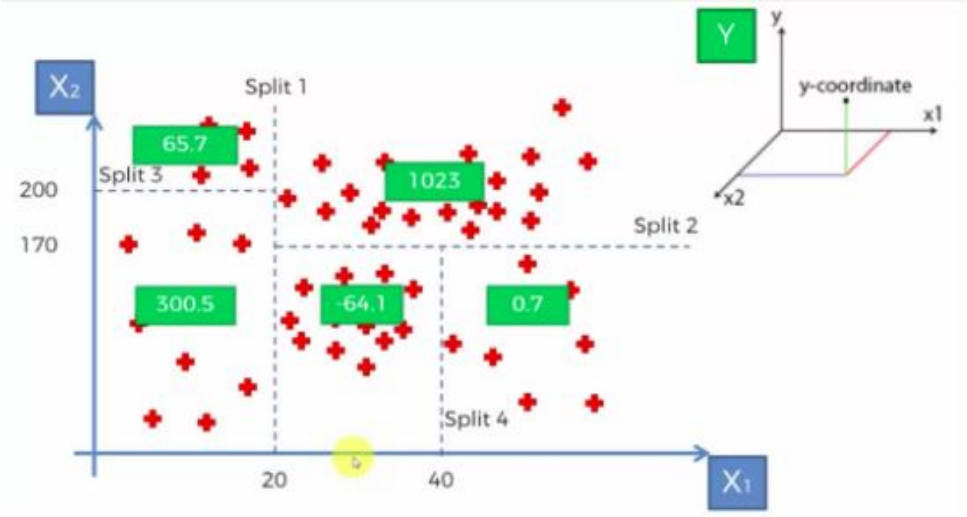
Labels의 평균 값: $\frac{100 + 120 + 140}{3} = \frac{360}{3} = 120$

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

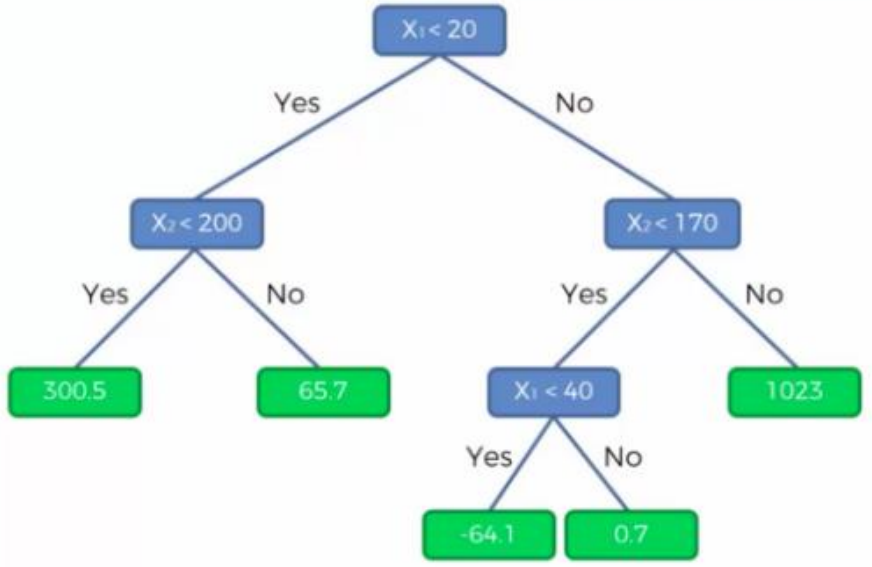
② 결정 트리 (Decision Tree) (1/2)

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

Y: 해당 영역의 평균 값



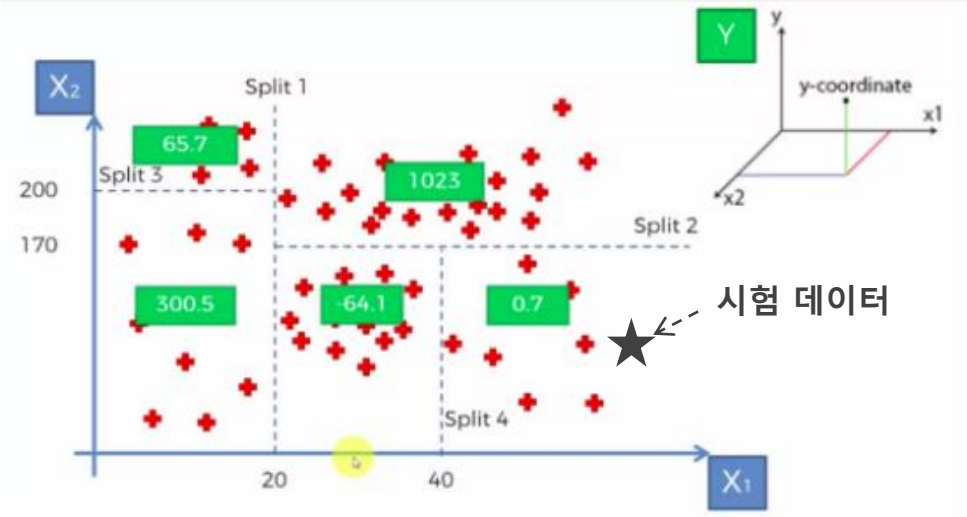
[사진출처] https://riverzayden.tistory.com/6



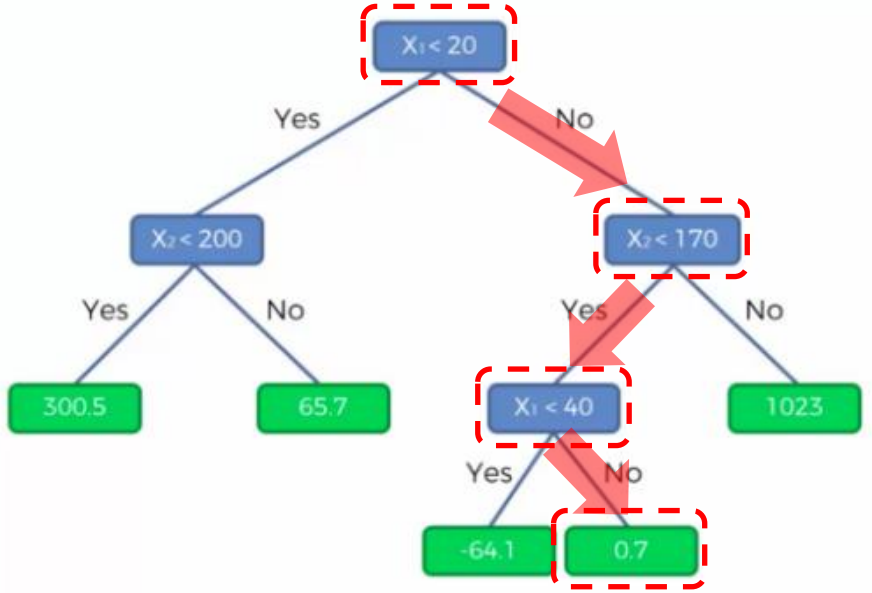
② 결정 트리 (Decision Tree) (2/2)

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

Y: 해당 영역의 평균 값



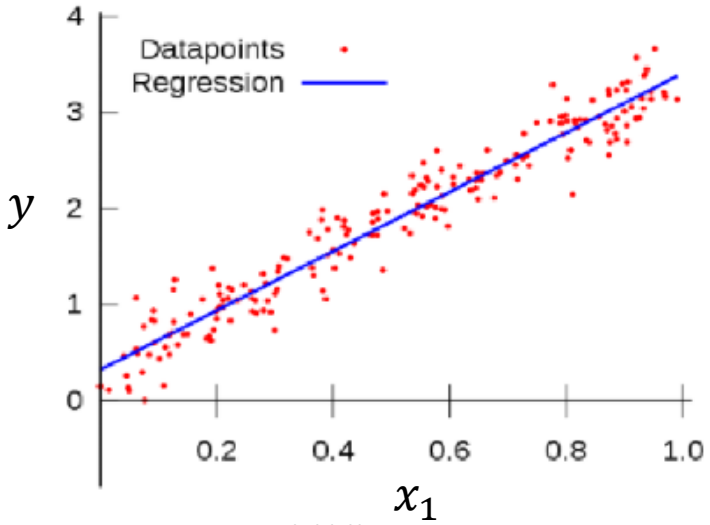
[사진출처] https://riverzayden.tistory.com/6



시험 데이터 ★의 회귀 결과값: 0.7

③ 선형 회귀 (Linear Regression) (1/4)

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



[사진출처] https://bangu4.tistory.com/100

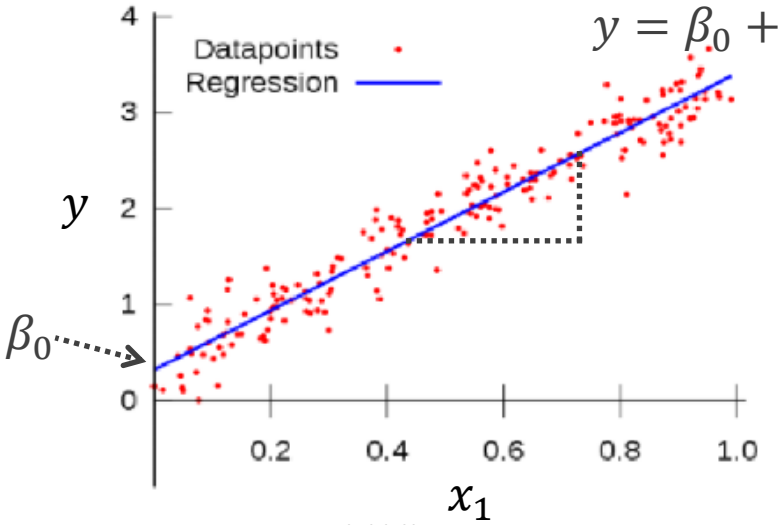
빨간색 점들이 학습 데이터이고,

이 데이터들을 잘 표현하는 파란색 직선을

찾아내는 것이 "선형 회귀"가 하는 역할임

③ 선형 회귀 (Linear Regression) (2/4)

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



[사진출처] https://bangu4.tistory.com/100

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

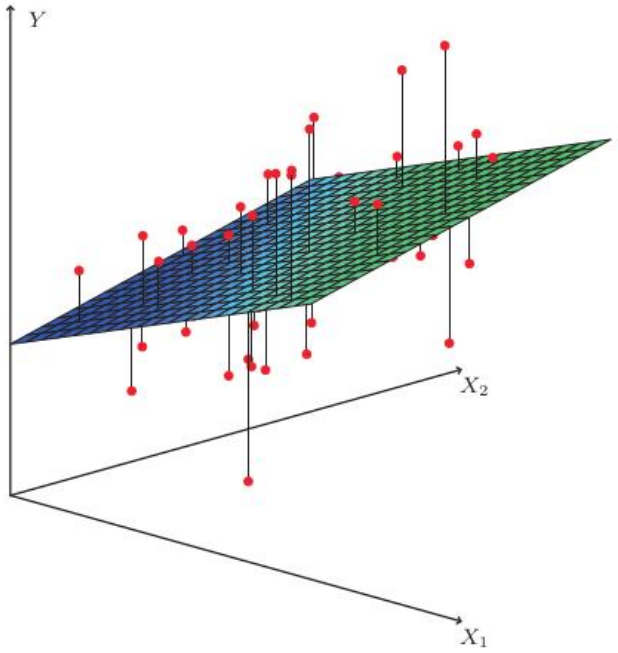
왼쪽 예제는 독립 변수의 개수가 1개인 경우임

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

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

③ 선형 회귀 (Linear Regression) (3/4)

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



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

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 β_0, β_1, β_2 값을 찾아냄

③ 선형 회귀 (Linear Regression) (4/4)

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

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

$$y = \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_2 + \cdots + \beta_i \times x_i + \cdots + \beta_n \times x_n$$

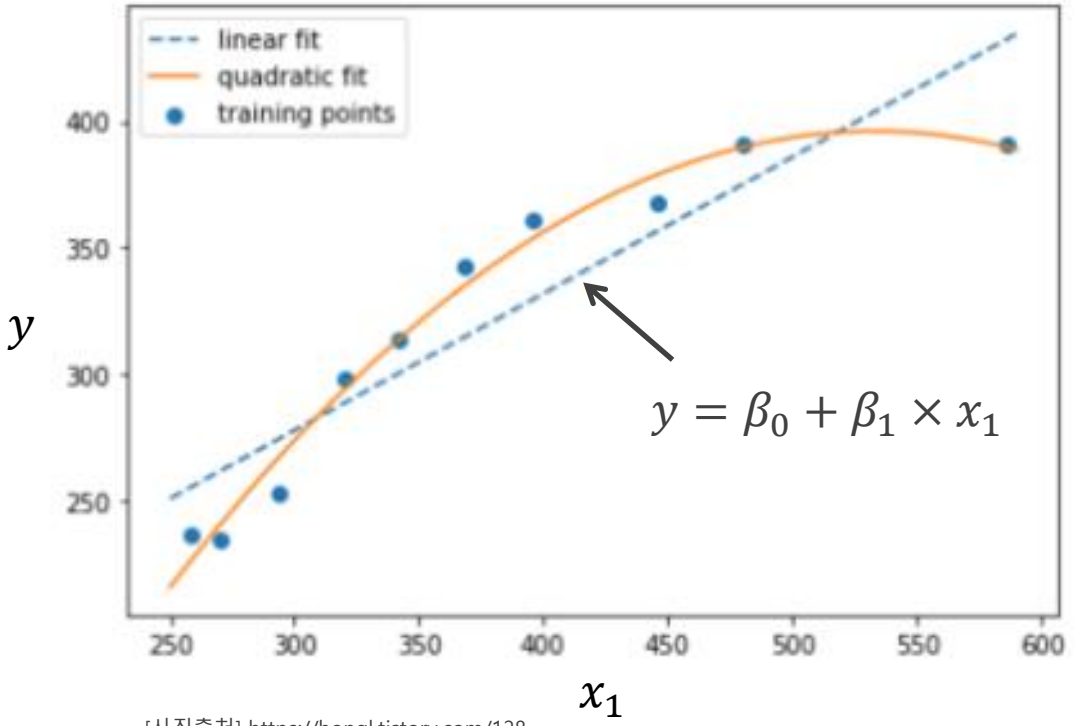
$$= \beta_0 + \sum_{i=1}^{n} (\beta_i \times x_i)$$

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)

- **선형 회귀의 단점**
  - 학습 데이터 내, 종속 변수 y와 독립 변수 x 사이의 상관 관계가 선형이 아닐 수 있다



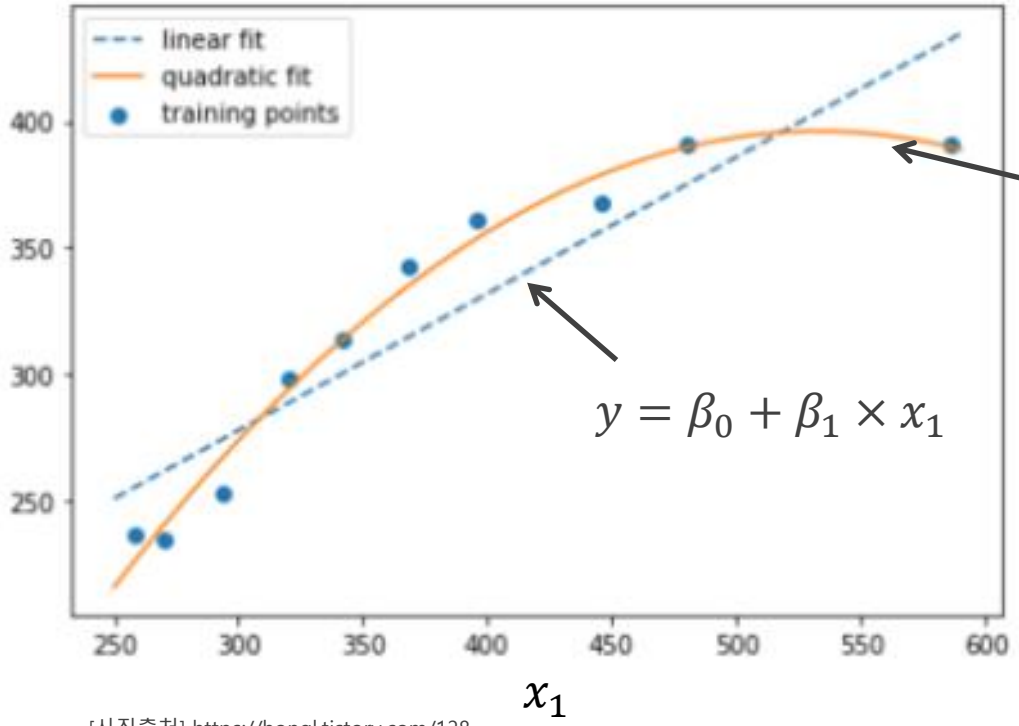
[사진출처] https://hongl.tistory.com/128

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)

- 각 독립 변수 x에 대한 고차원의 다항식을 이용하여 종속 변수 y의 관계를 비선형적 (Non-linear)으로 모델링하는 방법



[사진출처] https://hongl.tistory.com/128

왼쪽 예제는 독립 변수의 개수가 1개인 경우임

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

다항 회귀 알고리즘 수행을 통해서,

학습 데이터를 가장 잘 표현 하는

$\beta_0, \beta_1, \cdots, \beta_n$ 값을 찾아냄

AI Experts
Who Lead
The Future

03

붓꽃 분류 문제 체험

인공지능 라이브러리 패키지

- **Tensorflow(구글) vs Pytorch(메타)**
  - 딥러닝 관련 API 제공

특징	TensorFlow	PyTorch
개발자	구글 브레인 팀	페이스북의 인공지능 연구소
발표일	2015년 11월 9일	2016년 9월
프로그래밍 언어	파이썬, C++, CUDA	파이썬, CUDA
플랫폼	리눅스, macOS, 마이크로소프트 윈도우, 안드로이드, 자바스크립트	리눅스, macOS, 윈도우
종류	기계 학습 라이브러리	딥러닝 프레임워크
라이선스	아파치 2.0 오픈 소스 라이선스	BSD-3-Clause 라이선스
장점	유연하고 확장성이 뛰어나며, 대규모 데이터셋과 복잡한 모델을 처리하는 데 적합	사용자 친화적이고 직관적이며, 연구 및 프로토타이핑에 용이
단점	초보자에게는 다소 복잡하고 사용하기 어려울 수 있음	텐서플로에 비해 사용자층이 얕고, 예제 및 자료가 적을 수 있음

인공지능 라이브러리 패키지

- **Scikit-learn**
  - 데이터 분석 및 머신 러닝 모델 개발에 널리 사용
    - **다양한 알고리즘 제공**
      - 선형 회귀, 로지스틱 회귀, 의사결정나무, SVM, KNN 등 다양한 머신 러닝 알고리즘을 제공
    - **데이터 전처리 기능**
      - 데이터 정규화, 결측치 처리, 데이터 분할 등 데이터 전처리에 필요한 기능을 제공
    - **모델 평가 기능**
      - 모델의 성능을 평가할 수 있는 다양한 지표(정확도, 정밀도, 재현율, F1 스코어 등)를 제공
    - **사용하기 쉬움**
      - 직관적인 API를 제공하여 초보자도 쉽게 사용 가능
    - **오픈 소스**
      - 무료로 사용할 수 있는 오픈 소스 라이브러리
    - **다양한 데이터 형식 지원**
      - CSV, Excel, SQL 등 다양한 데이터 형식을 지원
    - **다양한 플랫폼 지원**
      - Windows, Linux, macOS 등 다양한 운영체제에서 사용
    - **커뮤니티 지원**
      - 활발한 커뮤니티 지원을 통해 문제 해결에 도움을 받을 수 있음

Sklearn 데이터셋

- 라이브러리 데이터셋
  - 빌트인(built-in) 데이터 셋들이 존재
    - 튜토리얼 진행을 위한 수준이므로, 규모가 크지 않음
    - Toy Dataset

- Sklearn 데이터 셋 종류
  - load_boston: 보스톤 집값 데이터
  - load_iris: 아이리스 붓꽃 데이터
  - load_diabetes: 당뇨병 환자 데이터
  - load_digits: 손글씨 데이터
  - load_linnerud: multi-output regression 용 데이터
  - load_wine: 와인 데이터
  - load_breast_cancer: 위스콘신 유방암 환자 데이터

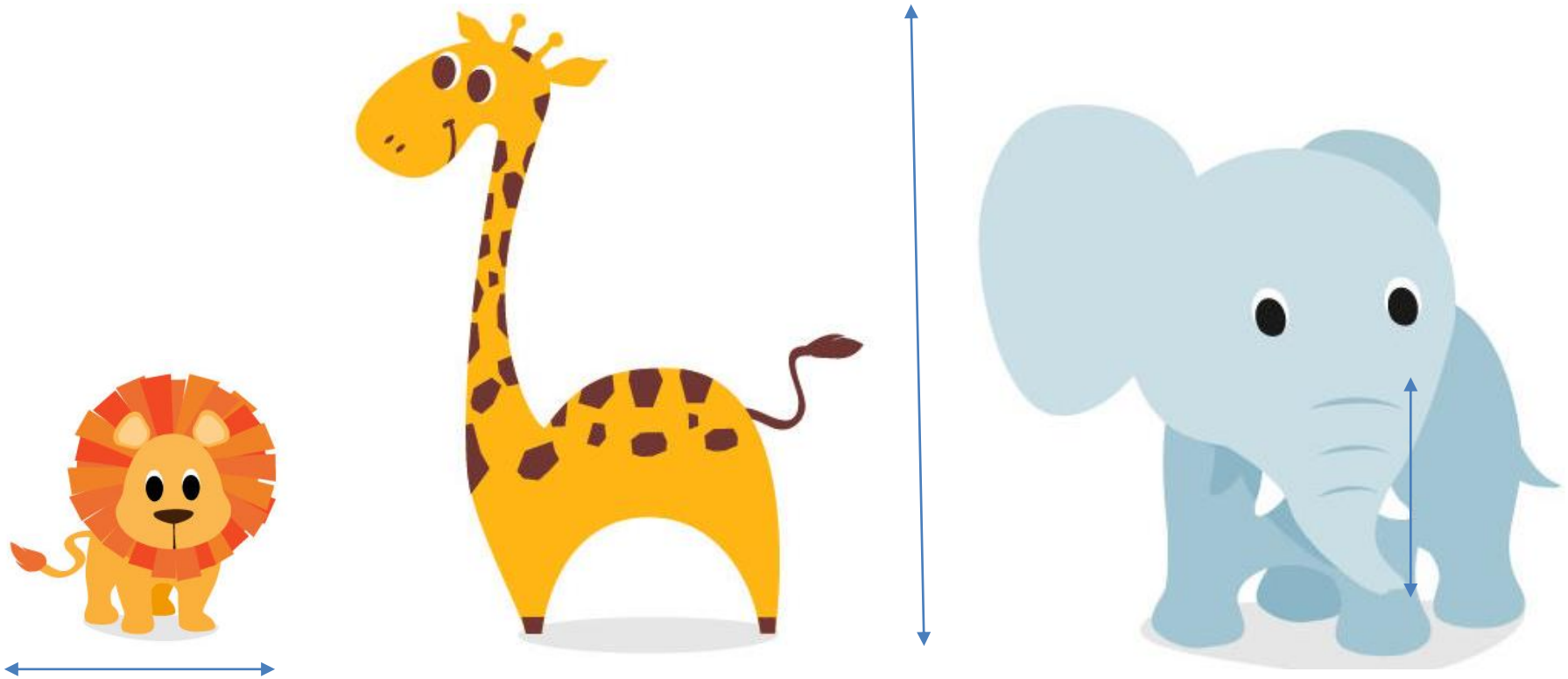
데이터 셋 조회

- 빌트인 데이터셋은 sklearn.utils.Bunch 라는 자료구조를 활용
  - key-value 형식으로 구성
  - 사전(dict)형 타입과 유사한 구조

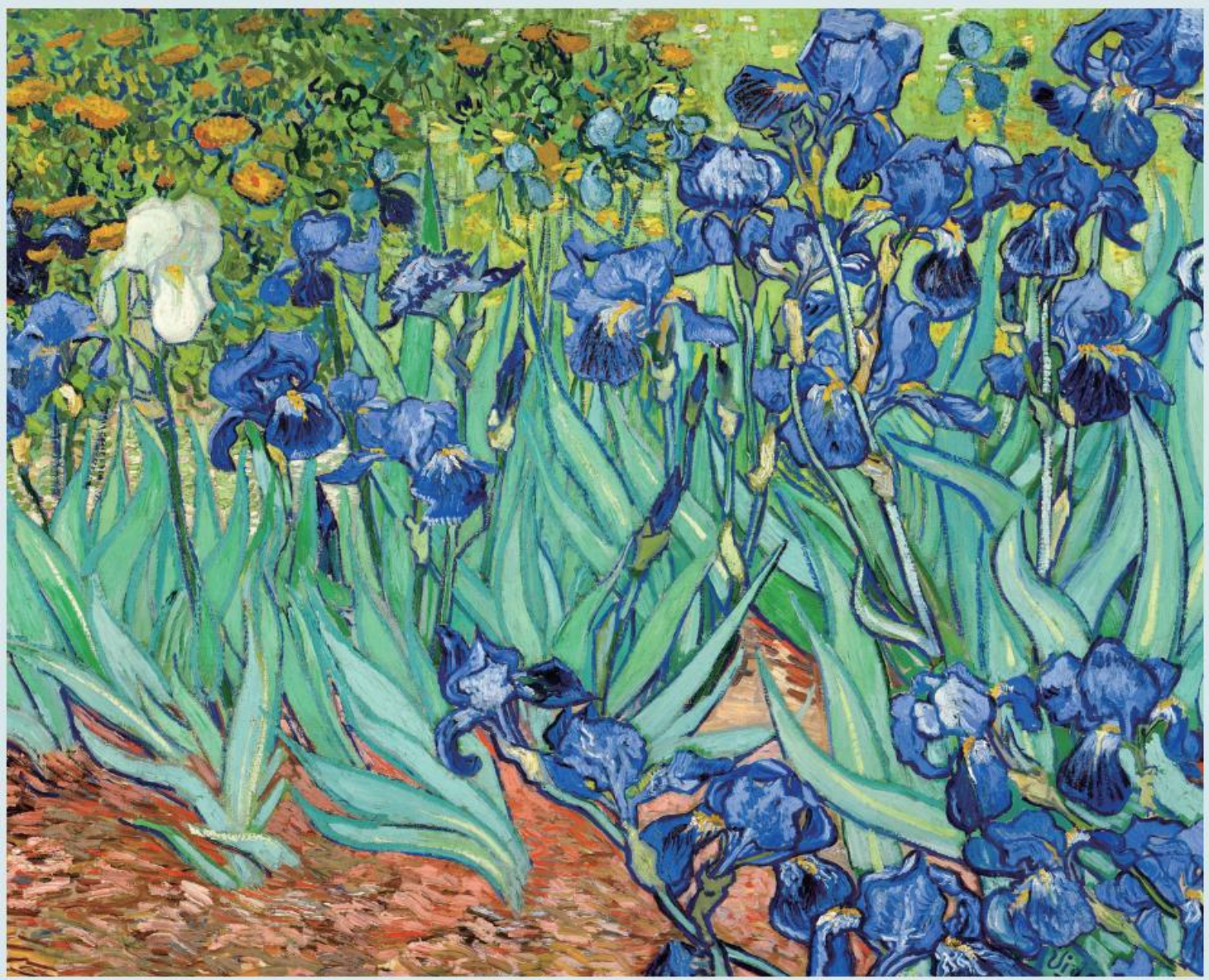
- 공통 key
  - data: 샘플 데이터, Numpy 배열로 구성
  - target: Label 데이터, Numpy 배열로 구성
  - feature_names: Feature 데이터의 이름
  - target_names: Label 데이터의 이름
  - DESCR: 데이터 셋의 설명
  - filename: 데이터 셋의 파일 저장 위치 (csv)

동물 분류

- 사자, 코끼리, 기린 3가지의 분류
  - 특징: 키, 길이, 몸무게, 코길이
  - 분류: 사자, 코끼리, 기린
  - 분류(classification)
    - # of classes = 3



다음 붓꽃의 품종을 알 수 있을까요?



붓꽃(Iris, 1889 빈센트 반 고흐)

붓꽃 분류 개요

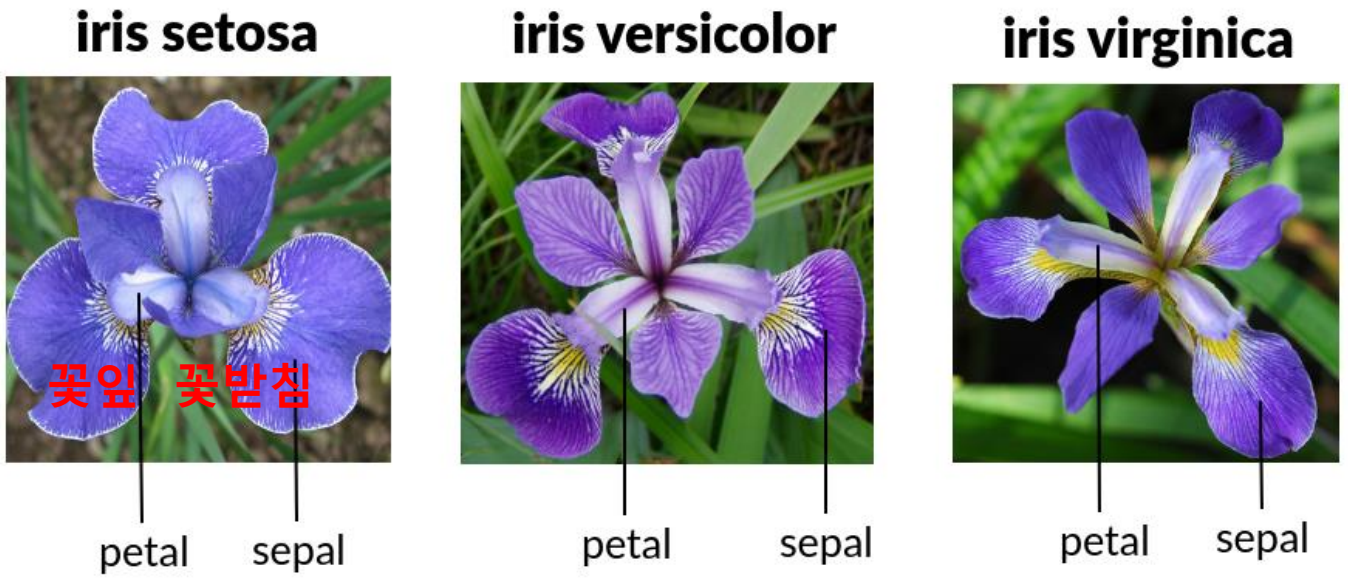
- 붓꽃의 품종 판별
  - 분류(classification)
    - # of classes = 3
      - Setosa, versicolor, virginica

그림 3-2 iris의 세 가지 품종(왼쪽부터 Setosa, Versicolor, Virginica)

레이블과 특징 수

- 꽃잎과 꽃받침의 너비와 길이
  - 4
  - 꽃잎
    - Petal
  - 꽃받침
    - Sepal
- 레이블
  - 3개의 붓꽃 중 하나

sepal.length	sepal.width	petal.length	petal.width



붓꽃 데이터

- **3 종류의 꽃잎과 꽃받침, 너비와 길이**
  - **Iris.csv**

sepal.length	sepal.width	petal.length	petal.width	variety
5.1	3.5	1.4	0.2	Setosa
4.9	3	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
4.6	3.1	1.5	0.2	Setosa
5	3.6	1.4	0.2	Setosa
5.4	3.9	1.7	0.4	Setosa
4.6	3.4	1.4	0.3	Setosa
5	3.4	1.5	0.2	Setosa
4.4	2.9	1.4	0.2	Setosa

Scikit-learn

- 홈페이지
  - https://scikit-learn.org
- 클래스 Bunch
  - 키로 속성을 참조하는 컨테이너 객체.
    - bunch["value_key"]
    - bunch.value_key.

sklearn.utils.Bunch

sklearn.utils.Bunch(**kwargs) [source]

Container object exposing keys as attributes.

Bunch objects are sometimes used as an output for functions and methods. They extend dictionaries by enabling values to be accessed by key, `bunch["value_key"]`, or by an attribute, `bunch.value_key`.

Examples

```
>>> from sklearn.utils import Bunch
>>> b = Bunch(a=1, b=2)
>>> b['b']
2
>>> b.b
2
>>> b.a = 3
>>> b['a']
3
>>> b.c = 6
>>> b['c']
6
```

datasets.load_iris()

- **자료형 sklearn.utils.Bunch**
  - 파이썬의 사전과 유사

```
# skleran 데이터셋에서 iris 데이터셋 로딩
from sklearn import datasets
iris = datasets.load_iris()

iris
```

```
{'DESCR': '.. _iris_dataset:₩n₩nIris plants dataset₩n------
 'data': array([[5.1, 3.5, 1.4, 0.2],
        [4.9, 3. , 1.4, 0.2],
        [4.7, 3.2, 1.3, 0.2],
        [4.6, 3.1, 1.5, 0.2],
        [5. , 3.6, 1.4, 0.2],
```

```
[3] type(iris)
```

```
sklearn.utils.Bunch
```

```
[ ] # iris 데이터셋은 딕셔너리 형태이므로, key 값을 확인
    iris.keys()
```

```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename'])
```

```
[ ] iris['filename']
```

```
'/usr/local/lib/python3.7/dist-packages/sklearn/datasets/data/iris.csv'
```

키 DESCR

- **iris['DESCR']**

```
Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================

:Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988
...
```

150개의 샘플

네 개의 특징(feature)

세 개의 부류

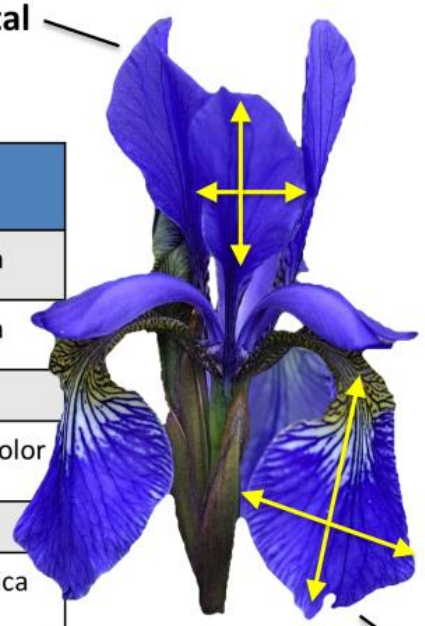
붓꽃 분류 문제

꽃잎

Petal

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica



Sepal

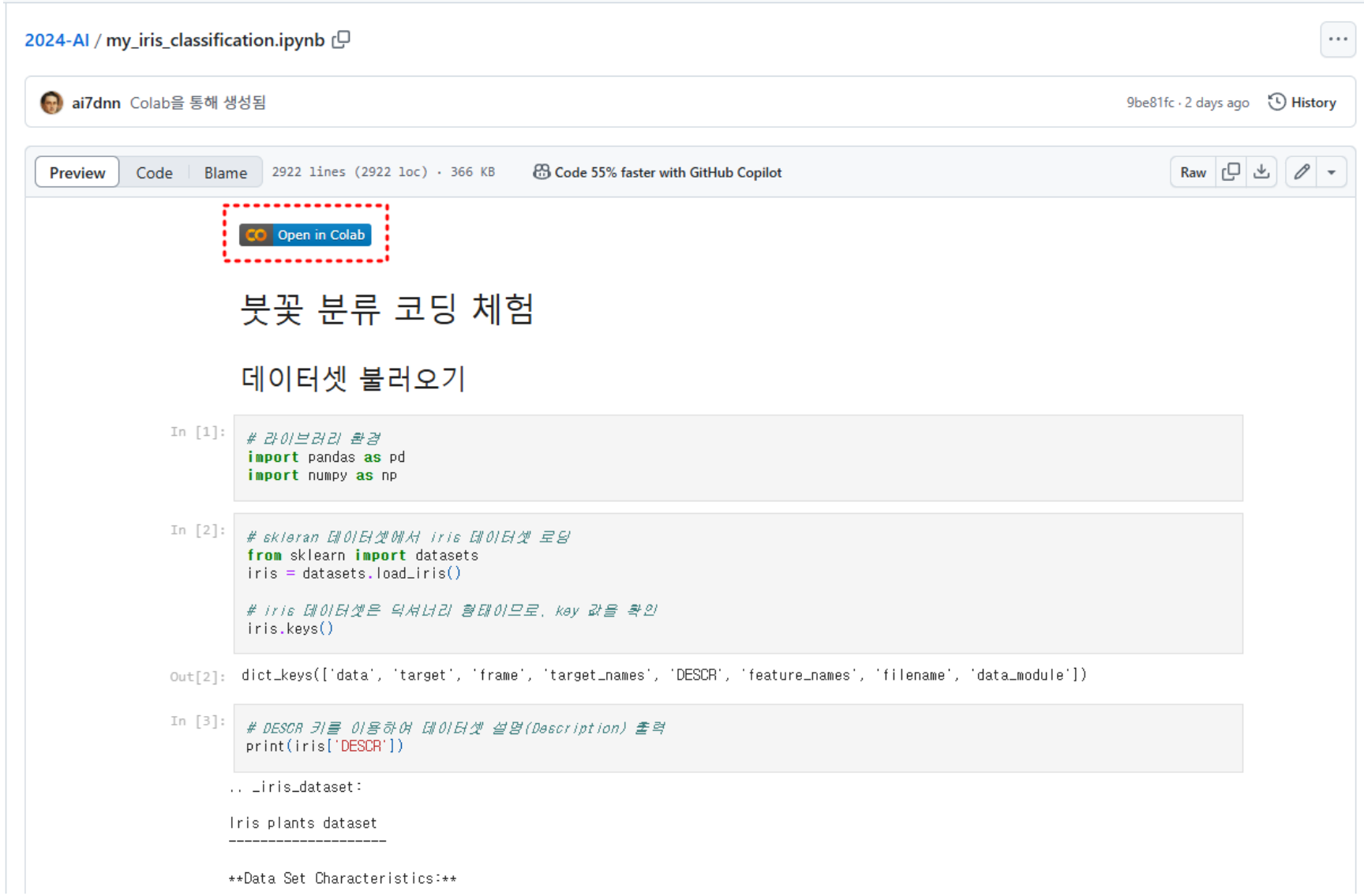
꽃받침

Class labels
(targets)

Features
(attributes, measurements, dimensions)

인공지능 실습 체험

- https://github.com/ai7dnn/2024-AI/blob/main/my_iris_classification.ipynb



2024-AI / my_iris_classification.ipynb

ai7dnn Colab을 통해 생성됨 9be81fc · 2 days ago History

Preview Code Blame 2922 lines (2922 loc) · 366 KB Code 55% faster with GitHub Copilot Raw

Open in Colab

붓꽃 분류 코딩 체험

데이터셋 불러오기

In [1]:
```
# 라이브러리 환경
import pandas as pd
import numpy as np
```

In [2]:
```
# sklearn 데이터셋에서 iris 데이터셋 로딩
from sklearn import datasets
iris = datasets.load_iris()

# iris 데이터셋은 딕셔너리 형태이므로, key 값을 확인
iris.keys()
```

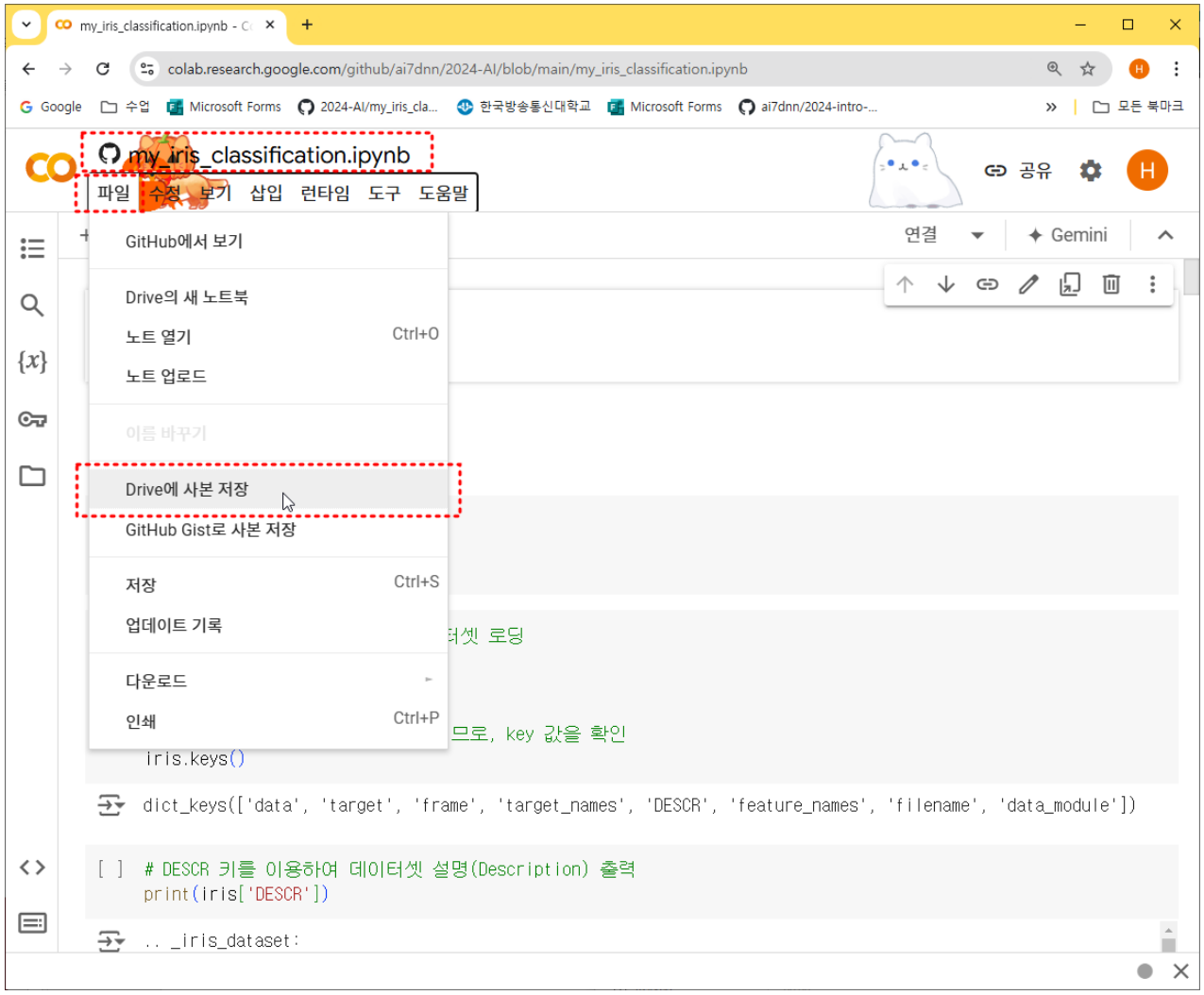
Out[2]:
```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
```

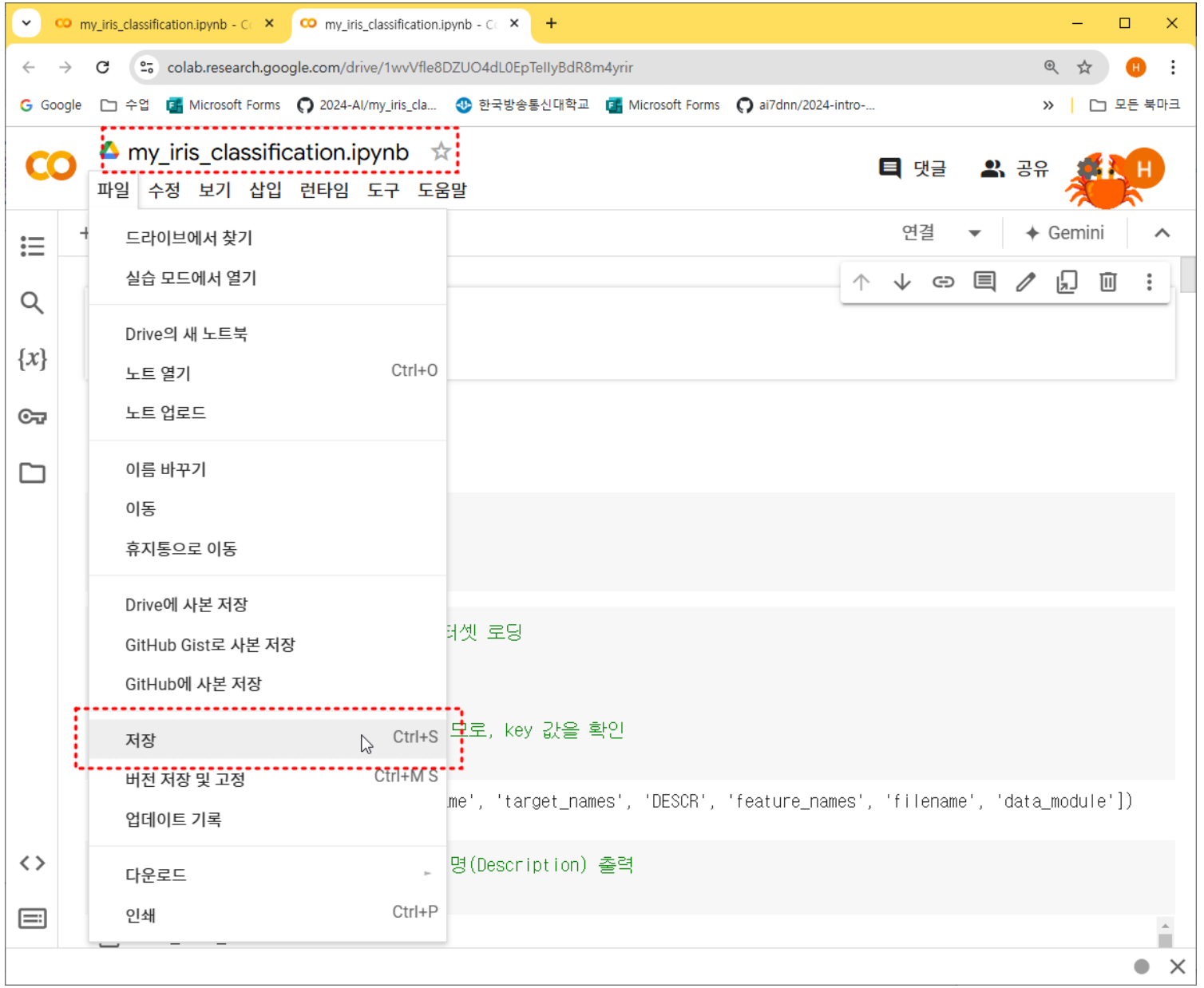
In [3]:
```
# DESCR 키를 이용하여 데이터셋 설명(Description) 출력
print(iris['DESCR'])
```

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**
```

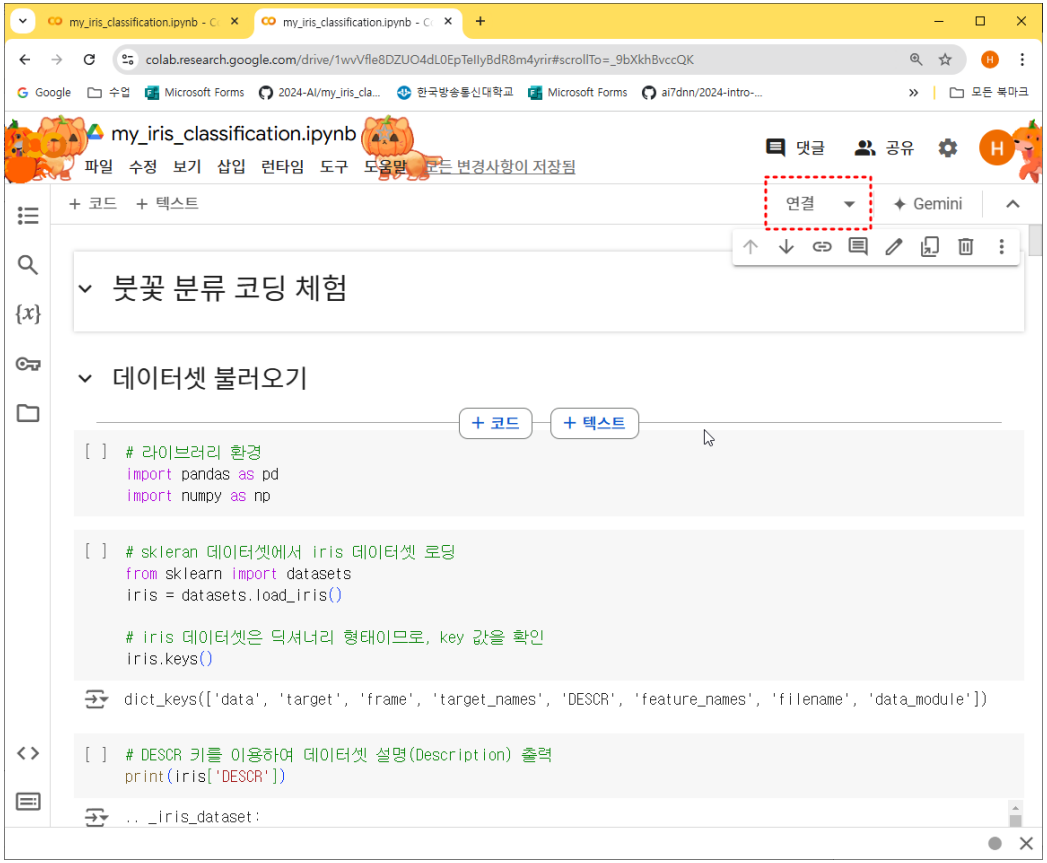
자신의 구글 드라이브에 소스 파일 저장

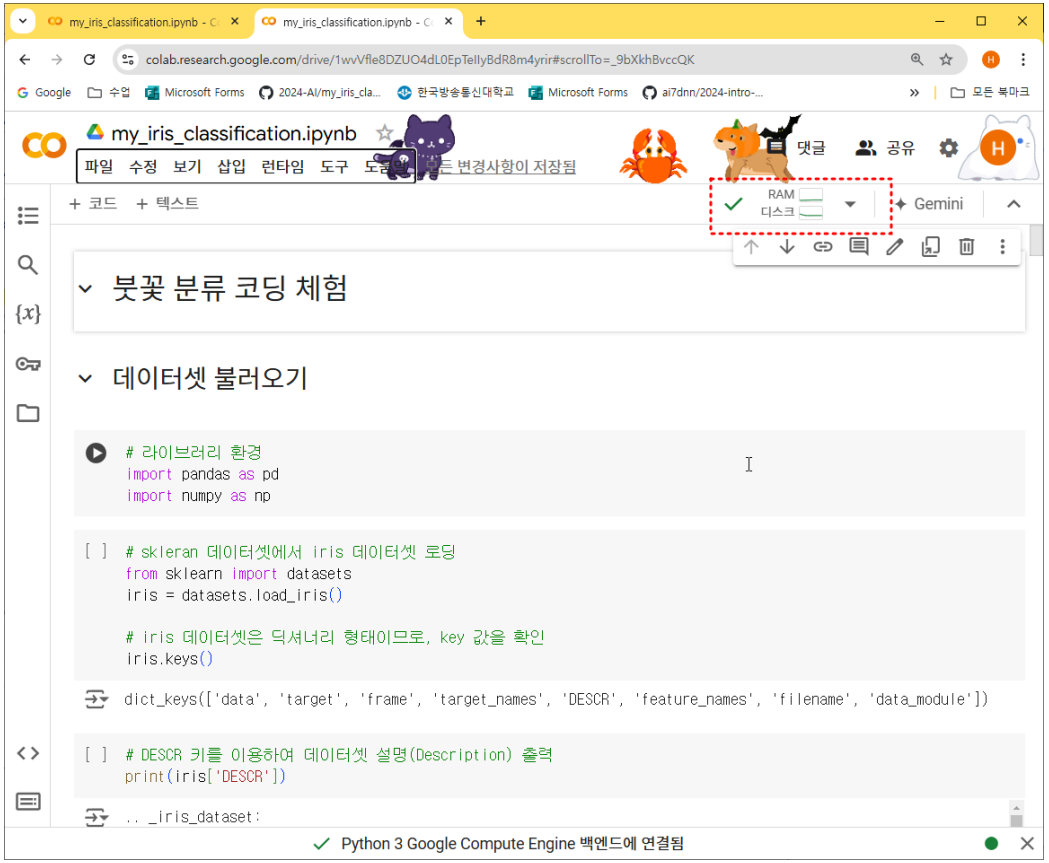
- 메뉴 파일 | Drive에 사본 저장




my_iris_classification.ipynb - C
my_iris_classification.ipynb - C
colab.research.google.com/drive/1wvVfle8DZUO4dL0EpTeIlyBdR8m4yrir
Google
수업
Microsoft Forms
2024-AI/my_iris_cla...
한국방송통신대학교
Microsoft Forms
ai7dnn/2024-intro-...
모든 북마크
my_iris_classification.ipynb
파일 수정 보기 삽입 런타임 도구 도움말
댓글
공유
연결
Gemini
드라이브에서 찾기
실습 모드에서 열기
Drive의 새 노트북
노트 열기 Ctrl+O
노트 업로드
이름 바꾸기
이동
휴지통으로 이동
Drive에 사본 저장
GitHub Gist로 사본 저장
GitHub에 사본 저장
저장 Ctrl+S
버전 저장 및 고정 Ctrl+M S
업데이트 기록
다운로드
인쇄 Ctrl+P
터셋 로딩
로, key 값을 확인
me', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
명(Description) 출력

연결 클릭


my_iris_classification.ipynb
파일 수정 보기 삽입 런타임 도구 도움말
댓글
공유
+ 코드 + 텍스트
연결
Gemini
붓꽃 분류 코딩 체험
데이터셋 불러오기
+ 코드 + 텍스트
# 라이브러리 환경
import pandas as pd
import numpy as np
# skleran 데이터셋에서 iris 데이터셋 로딩
from sklearn import datasets
iris = datasets.load_iris()
# iris 데이터셋은 딕셔너리 형태이므로, key 값을 확인
iris.keys()
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
# DESCR 키를 이용하여 데이터셋 설명(Description) 출력
print(iris['DESCR'])
.. _iris_dataset:


my_iris_classification.ipynb
파일 수정 보기 삽입 런타임 도구
댓글
공유
RAM
디스크
+ 코드 + 텍스트
Gemini
붓꽃 분류 코딩 체험
데이터셋 불러오기
# 라이브러리 환경
import pandas as pd
import numpy as np
# skleran 데이터셋에서 iris 데이터셋 로딩
from sklearn import datasets
iris = datasets.load_iris()
# iris 데이터셋은 딕셔너리 형태이므로, key 값을 확인
iris.keys()
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
# DESCR 키를 이용하여 데이터셋 설명(Description) 출력
print(iris['DESCR'])
.. _iris_dataset:
Python 3 Google Compute Engine 백엔드에 연결됨

순차적으로 셀 실행

- **셀 화살표 클릭**
  - **shift + enter**