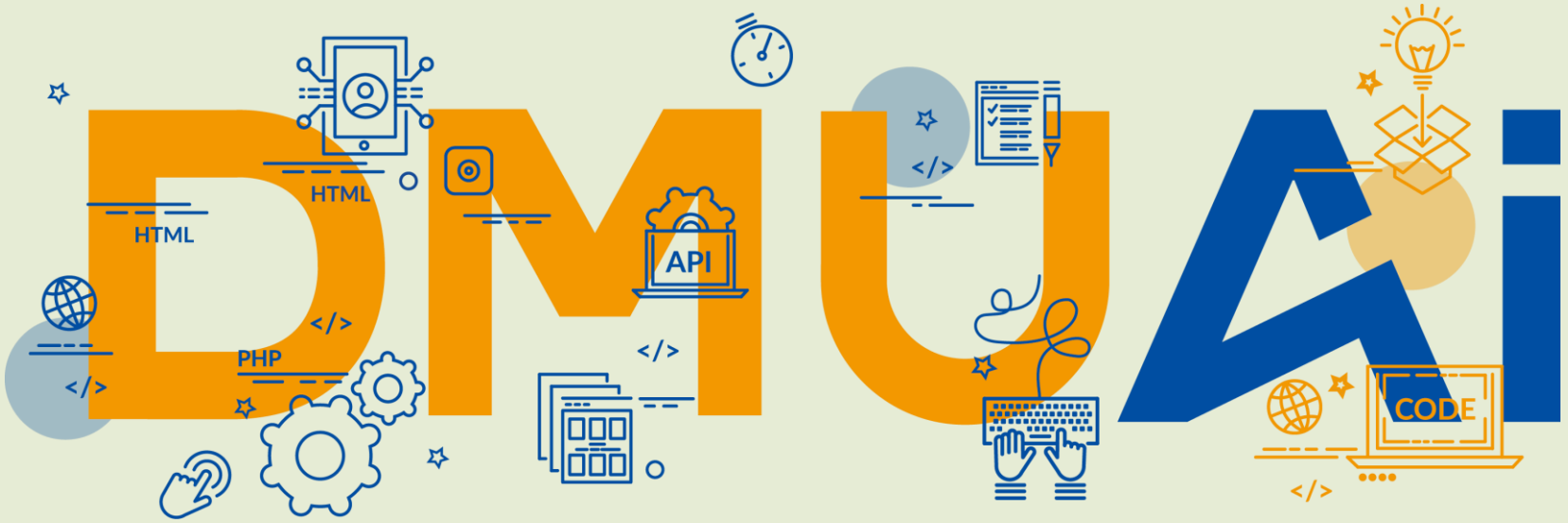
동양미래대학교 인공지능소프트웨어학과

CPU GPU TPU

Dongyang Mirae University
Dept. Of Artificial Intelligence



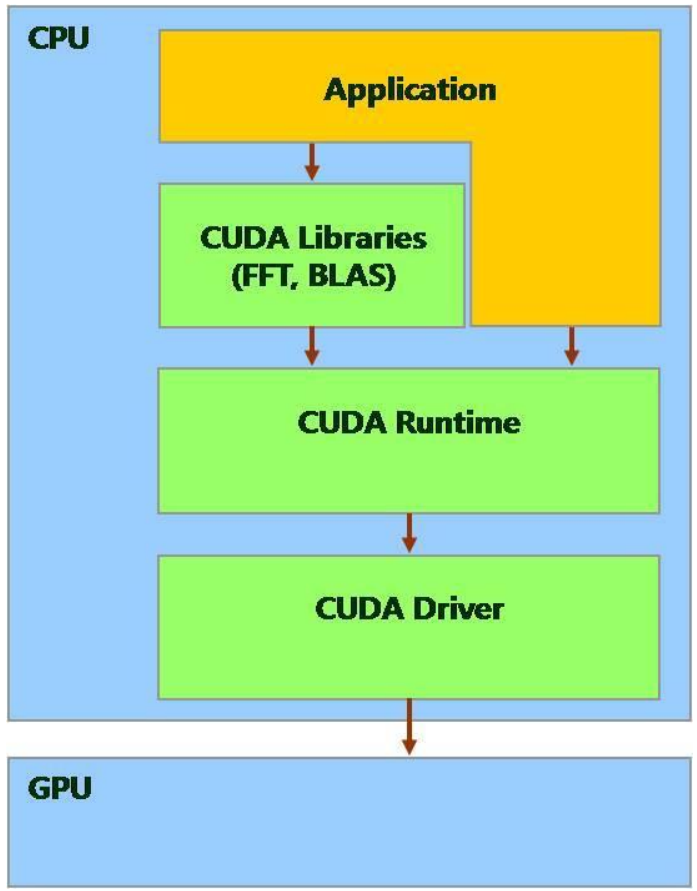
그래픽처리 장치 GPU의 인기

GPU란 용어는 1999년 엔비디아(Nvidia)에서 처음 사용

- **그래픽 처리 장치 GPU**
  - Graphics Processing Unit
  - 그래픽 연산 처리를 하는 전용 프로세서
- **GPGPU**
  - General Purpose Graphic Processing Unit
    - **일반 CPU 프로세서를 돕는 보조프로세서 (coprocessor)로서의 GPU**
  - 중앙 처리 장치(CPU)가 맡았던 응용 프로그램들의 계산에 GPU를 사용하는 기술
    - **GPU 컴퓨팅이란 GPGPU를 연산에 참여**
    - **고속의 병렬처리**
      - **대량의 행렬과 벡터를 다루는 데 뛰어난 성능을 발휘**
  - 딥러닝의 심층신경망에서 빅데이터를 처리
    - **대량의 행렬과 벡터를 사용**
    - **GPU 사용이 매우 효과적**
  - 12개 GPU가 2,000개의 CPU와 비슷한 계산 능력
- **GPU 병렬 처리 동영상**
  - https://www.youtube.com/watch?v=-P28LKWTzrI

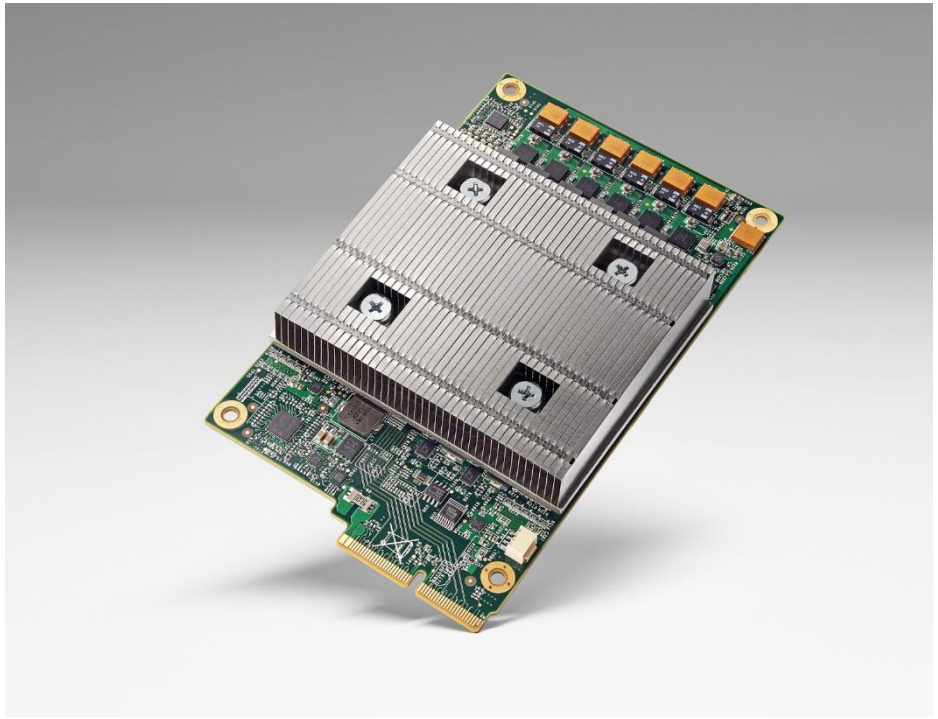
CUDA

- **GPU 업체인 NVIDIA의 GPU를 사용하기 위한 라이브러리 소프트웨어**
  - Compute Unified Device Architecture의 약자



구글의 TPU

- **구글은 2016년**
  - 텐서 처리 장치(Tensor Processing Unit)를 발표
  - 텐서란 벡터·행렬을 의미
  - TPU는 데이터 분석 및 딥러닝용 칩으로서 벡터·행렬연산의 병렬처리에 특화
  - 텐서플로(TensorFlow)
    - **TPU를 위한 소프트웨어**

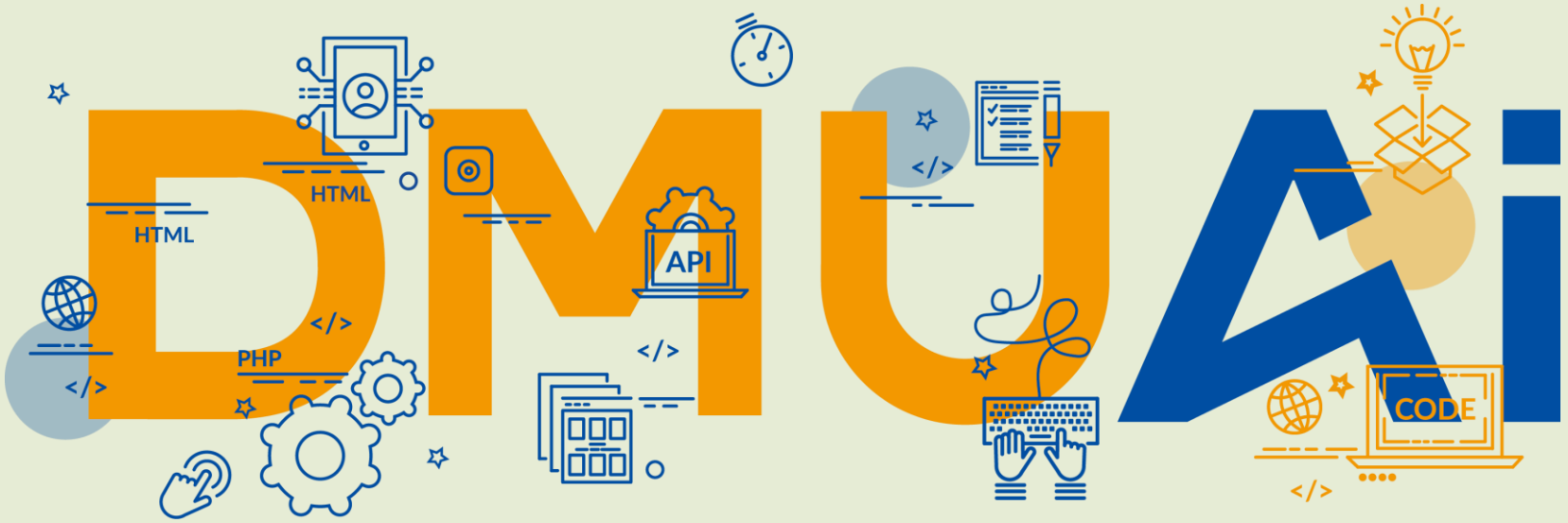


동양미래대학교

동양미래대학교 인공지능소프트웨어학과

머신러닝

Dongyang Mirae University
Dept. Of Artificial Intelligence



선형 모델

통계 분야에서 오랫동안 연구된 선형 모델은 현재 머신러닝 분야에서도 널리 쓰이고 있는 기법

- **선형 모델**
  - 데이터 특성에 대한 선형 함수를 만들어 예측하는 기법

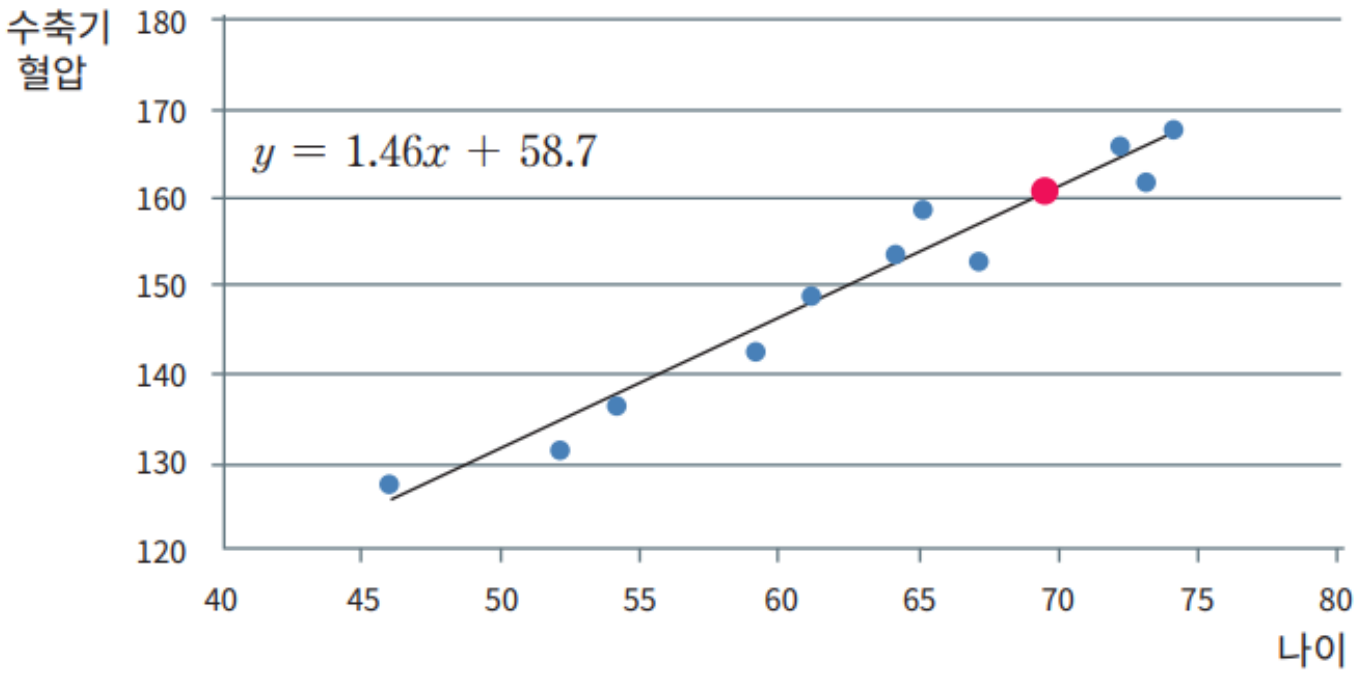
만일 입력 데이터의 특성의 수(차수)가 n이라면 선형 모델의 식은 다음과 같다. 선형 모델이 학습을 통해 구해야 할 값이 $w_1, w_2, \cdots w_n, b$ 등 많아진다.

$$y = w_1x_1 + w_1x_2 + w_3x_3 + \cdots + w_nx_n + b$$

나이로 혈압을 예측하는 문제

나이처럼 데이터 특성이 하나라면 선형 모델은 다음 1차 함수로 간단하게 표현

- **입력 자료로 파란색 데이터를 학습**
  - 모델은 학습으로 알지 못했던 매개변수인 a와 b를 각각 1.46, 58.7로 결정
    - **1차 함수에서 a는 기울기(또는 가중치), b는 y축과 만 나는 절편(또는 편향)**
- **나이 70의 혈압을 예측**
  - 모델은 학습으로 파악한 1차 함수를 통해 160.9(1.46×70 + 58.7)를 예측, 붉은 점이 (70, 160.9)



$$y = ax + b$$

y: 예측값

x: 특성

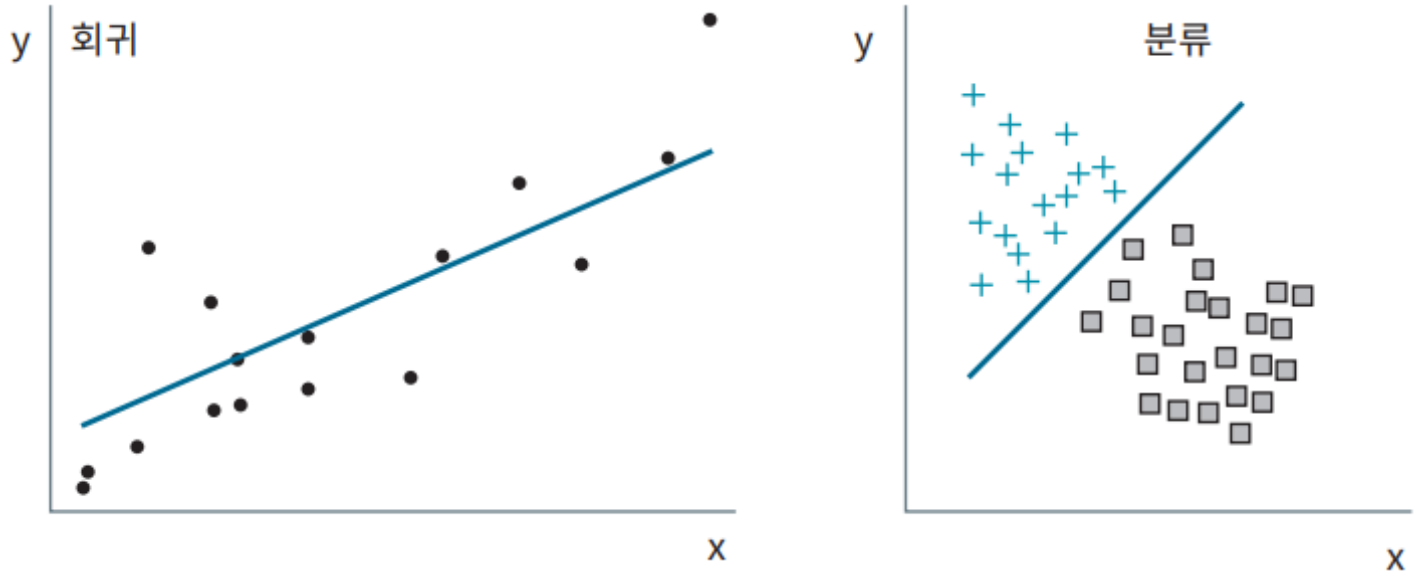
a: 기울기(가중치)

b: 절편(편향)

회귀와 분류에 사용되는 선형 모델

선형 모델은 회귀 뿐 아니라 분류에도 활용

- **선형 회귀**
  - 데이터에 가장 적합한 선을 찾는 방법
- **선형 분류**
  - 유형 클래스를 분류하는 가장 적합한 선을 찾는 방법



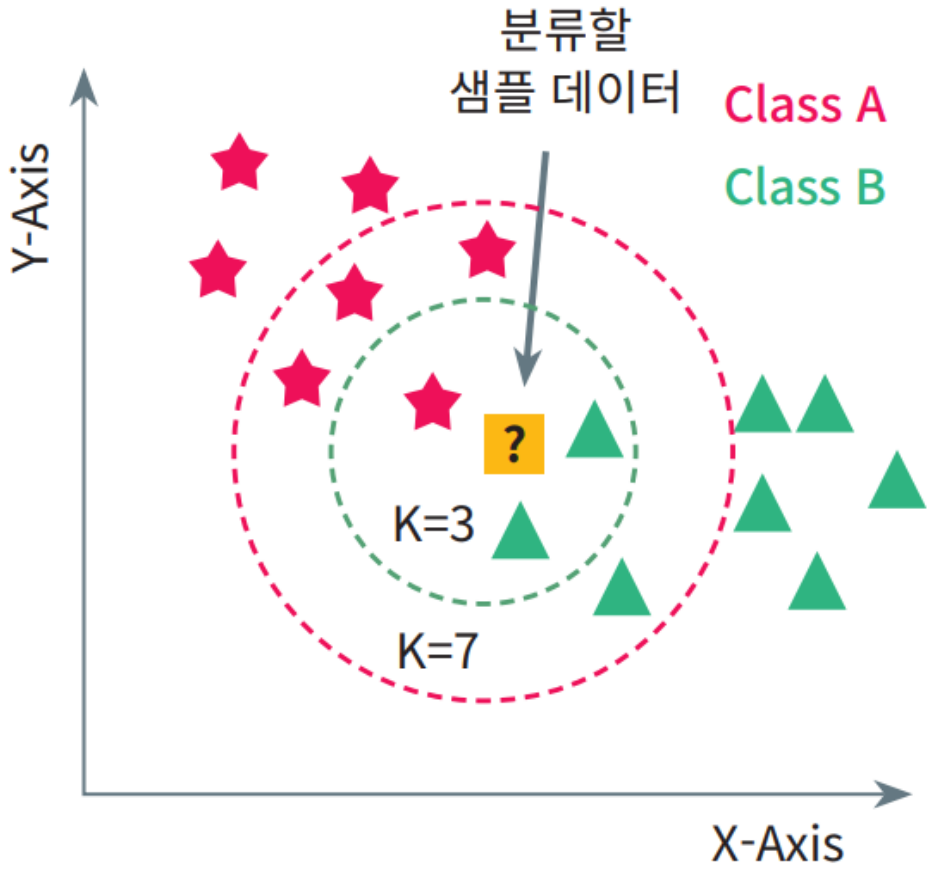
만일 입력 데이터의 특성의 수(차수)가 n이라면 선형 모델의 식은 다음과 같다. 선형 모델이 학습을 통해 구해야 할 값이 $w_1, w_2, \cdots w_n, b$ 등 많아진다.

$$y = w_1x_1 + w_1x_2 + w_3x_3 + \cdots + w_nx_n + b$$

K-최근접 이웃(KNN) 알고리즘

지도 학습인 K-최근접 이웃 알고리즘(KNN: K-Nearest Neighbors)은 가장 간단한 알고리즘으로 회귀와 분류에 모두 사용

- **KNN**
  - 새로운 자료에 대해
    - **기준으로 정한 가장 근접한 이웃 K개 중에서**
    - **가장 많은 유형으로 분류하는 방법**
- **사례**
  - 두 개의 특징 X, Y의 데이터에서
    - **A와 B, 2개 클래스로 분류하는 문제**
    - **별표는 클래스 A, 삼각형은 클래스 B**
      - **새로운 데이터 ?**
        - A와 B 중 무엇으로 분류해야 할까?

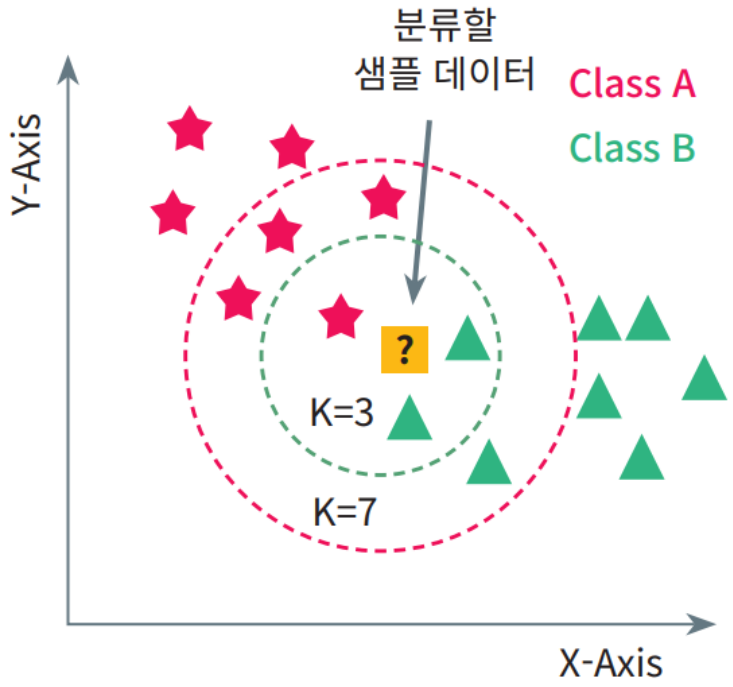


K-최근접 이웃(KNN) 알고리즘

주위에 있는 데이터 분류 다수결에 의해 결정

- **특징 X, Y의 데이터에서**
  - A와 B, 2개 클래스로 분류하는 문제
  - 새로운 데이터 ?
    - **A와 B 중 무엇으로 분류해야 할까?**
- **KNN 알고리즘**
  - K가 3
    - **?와 가장 가까운 이웃 3개**
      - **녹색 점선 원으로 표시 중**
      - **A는 1개, B는 2개**
    - **다수결에 의해 데이터 ?**
      - **클래스 B로 예측**
  - K가 7
    - **A는 4개, B는 3개이므로**
      - **A 로 예측**
  - 기준 K 값에 따라 예측이 달라질 수 있음

- **다양한 지도 학습의 다른 방법**
  - 서포트 벡터 머신
  - 의사결정트리
  - 랜덤 포레스트(random forest) 알고리즘



K-평균(means) 군집화 알고리즘

K-평균 군집화 알고리즘(K-means clustering algorithm)은 주어진 데이터를 비슷한 K개의 군집(클러스터)으로 묶는 알고리즘으로, 간단히 KC(K-means Clustering) 또는 K-평균 알고리즘

- **비지도 학습의 대표적 알고리즘**
  - 정답이 없는 입력 데이터에 정답을 붙이는 역할에 주로 사용
  - 옆 그림, 군집화를 수행하는 과정의 이해
- **K**
  - 알고리즘 적용 이전에 군집 수 K가 결정
    - **주어진 데이터로부터 그룹화할 그룹 수**
    - **군집의 수**
  - 각 군집은 하나의 중심(centroid)을 가짐
  - 동일한 자료에 대해 군집 수 K가 변하면 전혀 다른 결과
- **평균**
  - 각 군집의 중심과 데이터들의 평균 거리

K-평균(means) 군집화 알고리즘

군집화를 수행하는 과정을 다음 그림으로 간단히 살펴보면

- **처음(a)**
- **임의로 정한 중심(b)**
- **근접한 군집(c)으로 나눔**
  - 중심에 가까운 군집으로
- **다음 각각의 군집에서**
  - 다시 중심(d)을 구함
    - **데이터의 평균 값으로**
- **새로 정한 중심을 기준으로 다시**
  - 근접한 군집(e)으로 나눔
- **이러한 과정을 반복**
  - 군집에 소속된 데이터가 바뀌지 않거나,
  - 무게중심이 변하지 않으면 종료(f)

차원 축소

특징 수가 많은 고차원 데이터는 그만큼 데이터 저장 양도 많으며, 머신러닝 모델도 학습 시간이 훨씬 많이 소요되며 원하는 결과가 안 나올 수 있음

- **특징 수를 줄이는 차원 축소(dimensionality reduction)**
  - 높은 차원의 원시 데이터(raw data)에서 중복되거나 관련성이 떨어지는 특성을 파악해
    - **차원 수를 줄이는 방식**
      - **모델의 이해를 높이고 계산 속도를 향상**
      - **모델의 성능을 향상시킬 수 있음**
  - 데이터의 차원 축소는 정보의 손실이 발생 가능
    - **그러므로 데이터의 차원을 축소하면서 정보 손실을 최소로 한다면 금상첨화**
- **차원 축소 예, 그림자 투영에 비유**
  - 간단히 x, y, z, 3개의 특징 수인 3차원을 x, y, 2개인 2차원으로 줄이는 예

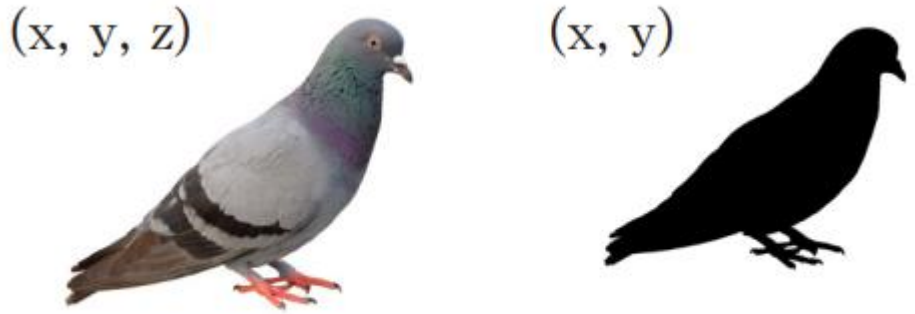


그림 6.47 ▸ 그림자 투영(3차원을 2차원으로 차원 축소)

비지도학습 종류

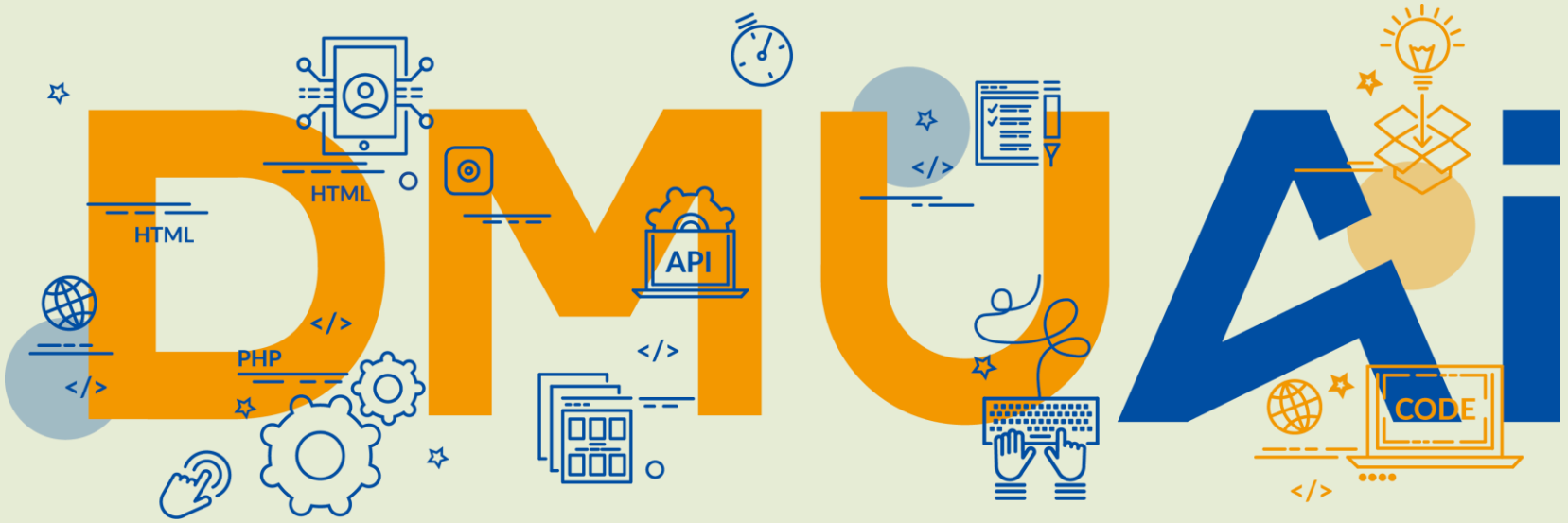
- **비지도학습**
  - 군집 (Clustering)
    - **- k-평균 (k-Means)**
    - **- 계층 군집 분석 (Hierarchical Cluster Analysis (HCA))**
    - **- 기댓값 최대화 (Expectation Maximization)**
  - 시각화 (Visualization)와 차원 축소(Dimensionality reduction)
    - **- 주성분 분석 (Principal Component Analysis (PCA))**
    - **- 커널 (kernel PCA)**
    - **- 지역적 선형 임베딩 (Locally-Linear Embedding (LLE))**
  - 연관 규칙 학습 (Association rule learning)
    - **- 어프라이어리 (Apriori)**
    - **- 이클렛 (Eclat)**

동양미래대학교

동양미래대학교 인공지능소프트웨어학과

인공신경망과 딥러닝

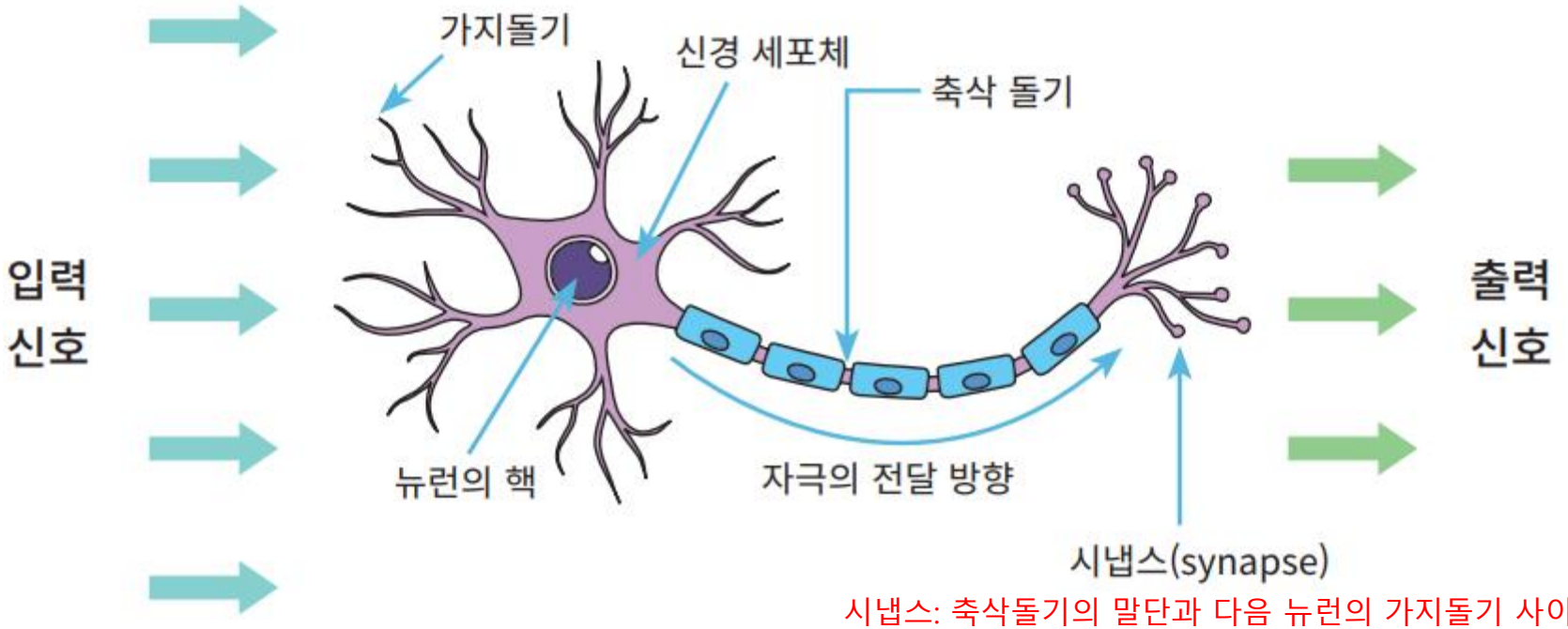
Dongyang Mirae University
Dept. Of Artificial Intelligence



뉴런과 퍼셉트론 1/2

우리 인간의 뇌에는 1천억 개의 뉴런이라는 신경세포가 있음

- **뉴런**
  - 가지돌기(dendrite)(또는 수상돌기)에 들어온 여러 신호는 하나로 통합되어
    - **어떤 임계 값을 초과하면 축삭돌기(axon)를 통해 이동되고 다음 신경 세포에 전달**



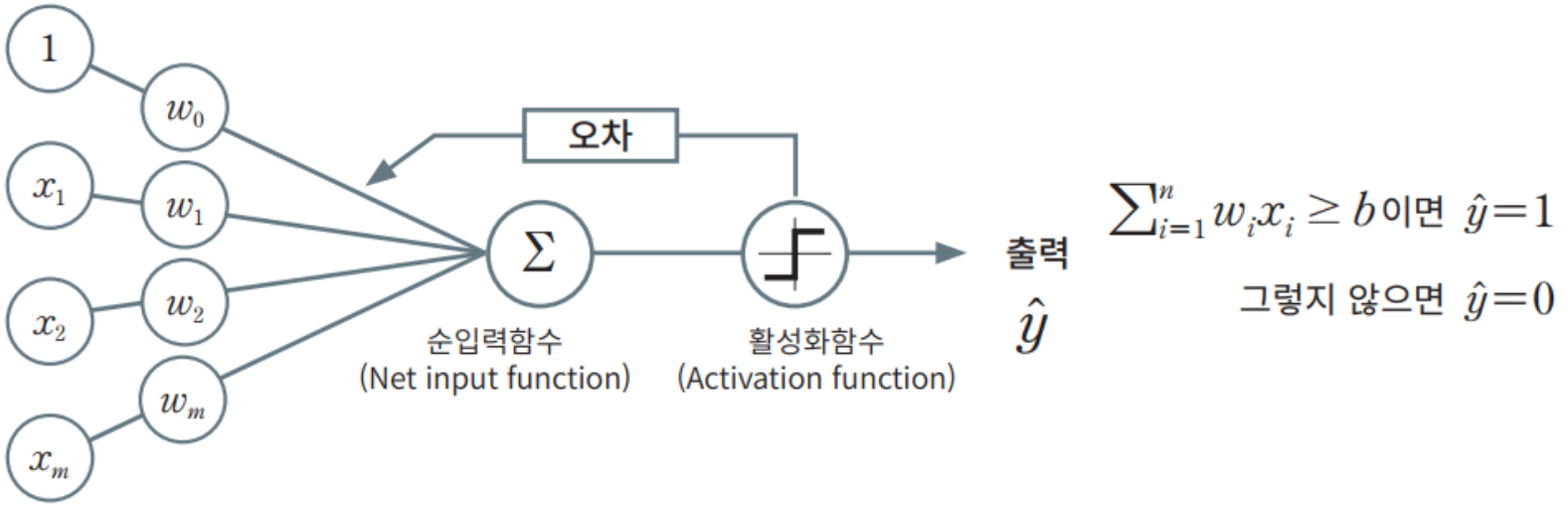
시냅스: 축삭돌기의 말단과 다음 뉴런의 가지돌기 사이 부분으로
전 뉴런에서 다음 뉴런으로 전달되는 부위를 말함

뉴런과 퍼셉트론 2/2

1957년 심리학자인 프랭크 로젠블랫(Frank Rosenblatt)는 세계 최초로 인공신경망인 퍼셉트론(perceptron)을 제시

- **퍼셉트론**
  - 다수 입력(input), 하나의 출력(output)
  - 가중치(weights)
    - **입력의 강도(중요도)를 표현**
  - 스스로 학습하는 능력
    - **초기 가중치는 임의의 값 지정**
    - **뉴런의 결과를 목표값과 비교**
      - **그 차이인 오차(error)를 사용**
        - 다시 그 다음 단계의 가중치에 반영

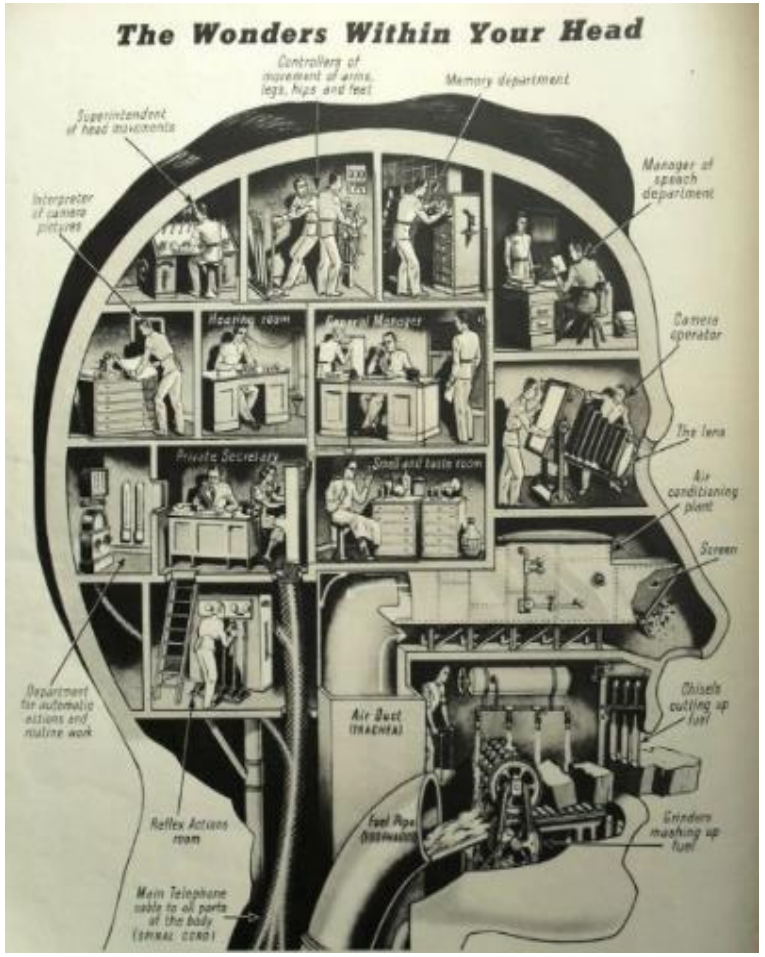
- **임계값 –b**
  - 가중치와 입력 곱의 합
    - **가중 합계(weighted sum)**
      - **w0 + x1w1 + x2w2 + … + xnwn**
    - **값이 임계값(threshold) b(=-w0)보다 크면**
      - **뉴런은 실행(fire), 결과값이 1**
    - **그렇지 않으면 0이 결과값**



연결주의와 인간의 뇌

연결주의의 시작이 바로 1943년에 제시된 맥컬럭-피츠 뉴런(MCP 또는 MP neuron)

- **연결주의(connectionism)**
  - 인간의 중추 신경계인 뉴런에서의 처리를 추상화한 인공신경망을 사용
    - **지적 능력을 설명하고자 하는 인지 과학의 운동**
  - 연결주의의 신경망
    - **단위 뉴런 간의 연결 강도를 표현하는 가중치와 함께 여러 뉴런으로 구성된 뇌를 단순화시킨 모델**
- **1938년의 인포그래픽**
  - 인간의 뇌는 여러 기능을 수행하는 여러 방으로 구성
  - 인간 뇌의 경이로움을 표현



가중치와 편향

퍼셉트론의 수식 wx+b에서 가중치 w를 기울기 a로 생각하면 결국 ax+b가 되어 기울기가 a, y절편이 b인 1차 함수와 같음

- **퍼셉트론 계산식**
  - 입력 특성은 x 하나이고 w를 b로 표현
    - **다음과 같은 간단한 수식의 퍼셉트론**
  - 입력 특성이 하나
    - **가중치는 기울기, 편향은 절편**

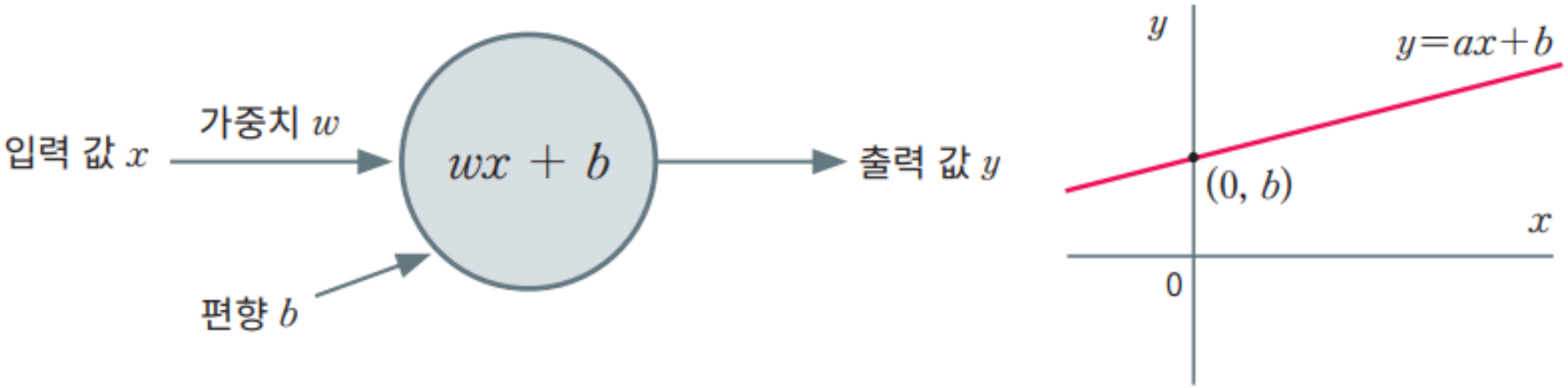


그림 6.52 ▸ 입력 특성이 하나인 간단한 퍼셉트론

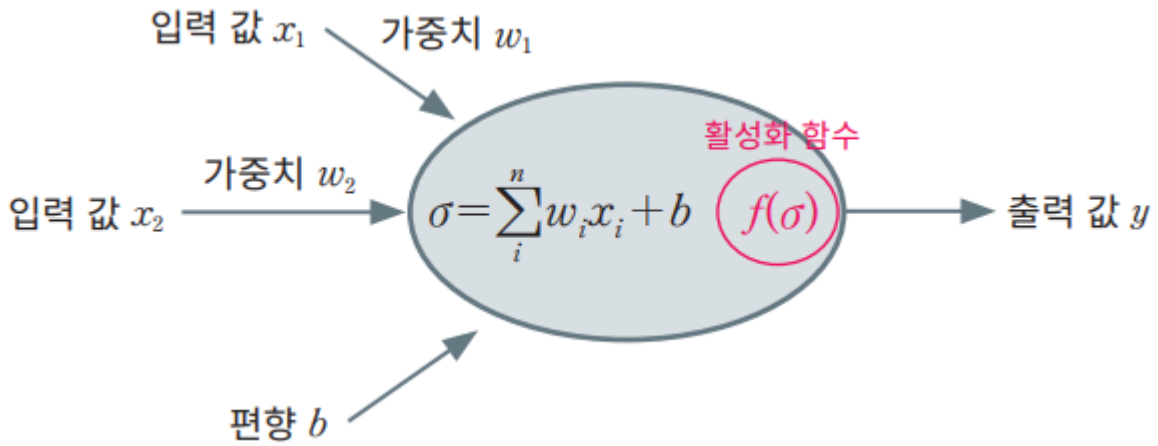
활성화 함수(activation function)

활성화 함수는 퍼셉트론의 최종 값의 세기를 조절하는 데 사용되며 활성화 함수 출력값은 다음 퍼셉트론의 입력으로 사용

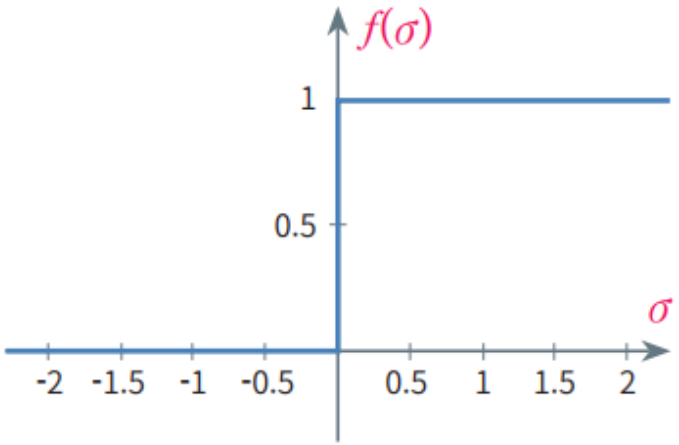
- **퍼셉트론에서 최종 값을 결정하는 함수**
  - 퍼셉트론의 입력값과 가중치를 곱한 것에 편향을 모두 합한 값에 대한 함수

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

  - **함수** $f(\sigma)$



- **단위 계단 함수(unit step function)**
  - 로젠블랫의 활성화 함수
  - 가중 합계(weighted sum)
    - **0 이상이면 결과는 1**
    - **0 미만이면 0**



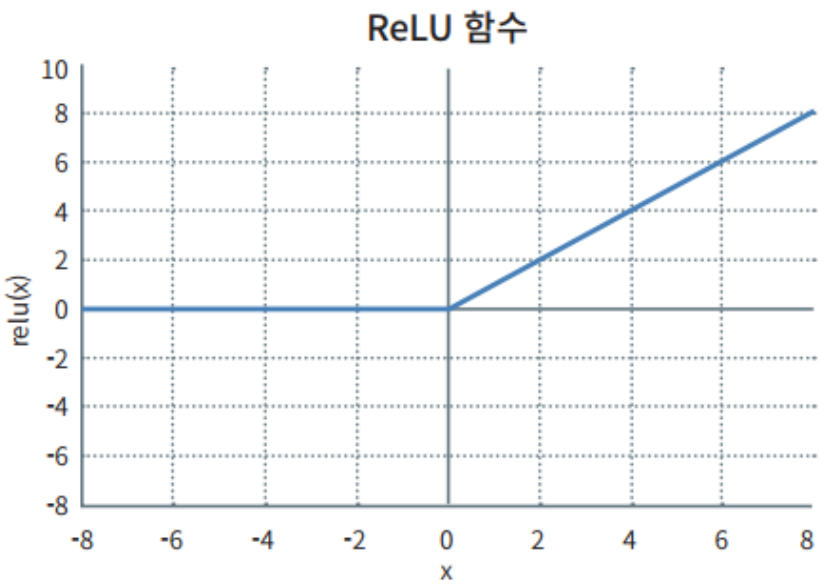
$$f(\sigma) = \begin{cases} \sigma \geq 0 \text{이면 } 1 \\ \sigma < 0 \text{이면 } 0 \end{cases}$$

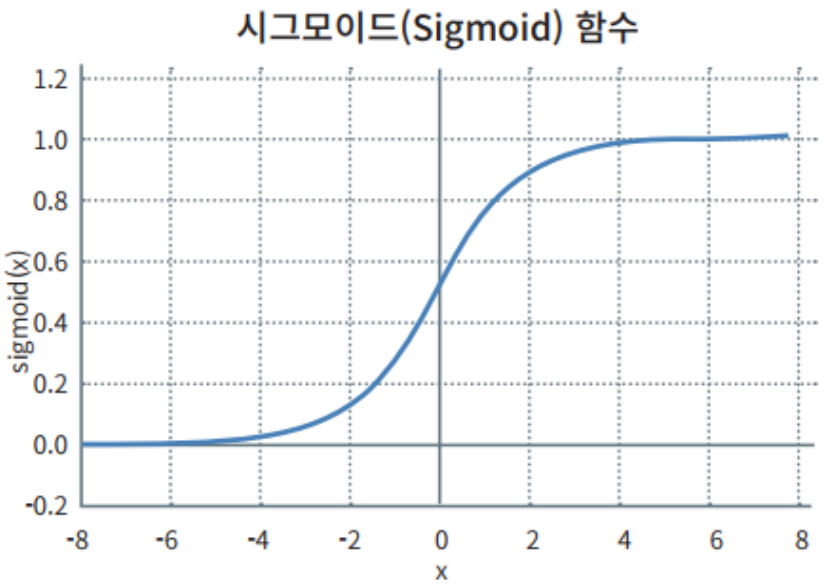
그림 6.54 ▸ 로젠블랫의 활성화 함수: 단위 계단 함수

ReLU와 시그모이드 활성화 함수

활성화 함수는 다양한데 최근에 많이 사용하는 ReLU(Rectified Linear Unit) 함수

- **항등 함수(identity function)**
  - 가장 단순한 활성화 함수는 입력 그대로 출력하는 f(x)=x
- **ReLU 함수**
  - 양수 부분은 항등함수와 동일, 0 이하는 0
- **시그모이드(sigmoid) 함수와 하이퍼볼릭탄젠트(tanh) 함수**





간단한 인공신경망 이해

가장 간단한 인공신경망(ANN: Artificial Neural Network)은 입력층(input layer)과 퍼셉트론의 출력층(output layer)으로 구성

- **아파트 가격 예측, 선형 회귀 문제**
  - 입력층, 4차원의 입력 특성
    - **크기, 방의 개수, 도심과의 거리, 건축 연수**
      - **특징 수인 차수: 4**
  - 출력층, 하나의 퍼셉트론
    - **활성화 함수: 항등 함수**
      - **입력의 가중치 곱의 합이 아파트 가격**
  - 편향은 그림에서 생략
  - 학습
    - **훈련 데이터로 가중치와 편향을 결정**
      - **4개의 가중치와 하나의 편향 값을 찾음**
  - 예측
    - **테스트 데이터로 출력인 아파트 가격을 예측**

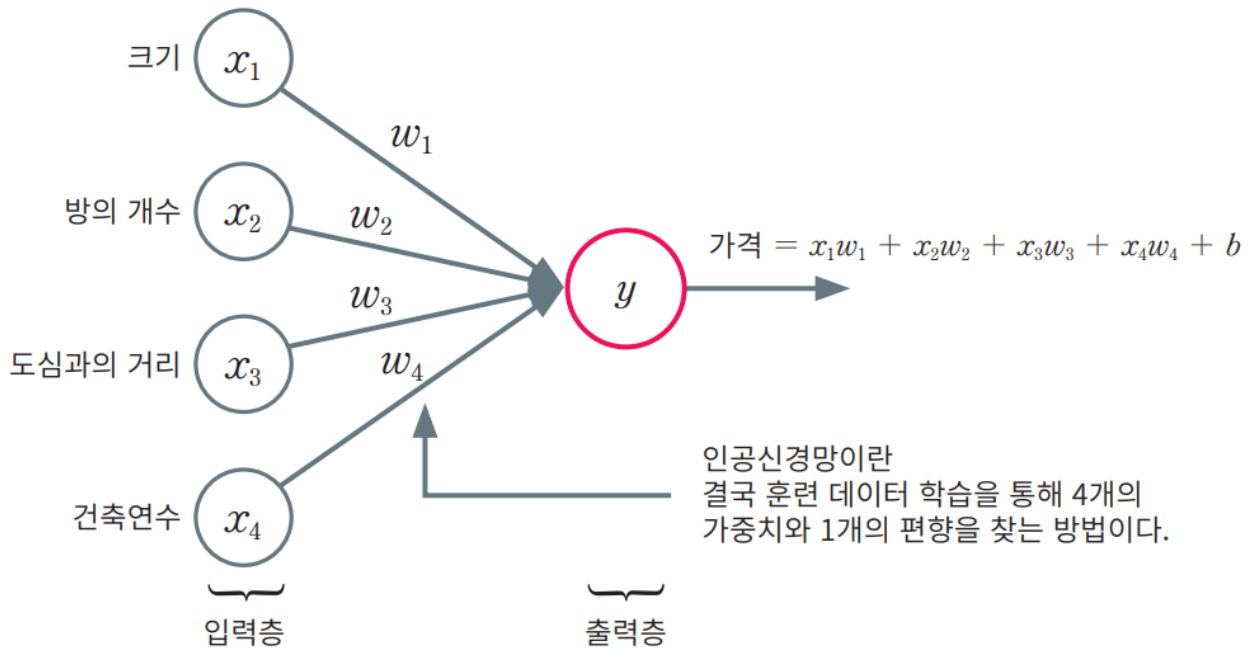


그림 6.56 ▸ 입력의 차원이 4인 단순 선형 회귀 문제의 인공신경망

다층 퍼셉트론(MLP)과 심층신경망(DNN)

여러 값이 입력되는 입력층(input layer)과 여러 개의 뉴런으로 구성되는 출력층(output layer)을 구성

- **다층 신경망(MLP: Multi-Layer Perceptron)**
  - 인공신경망을 확장해 가로로 여러 개 연결한 층(layer)을 쌓음
  - 은닉층(hidden layer) 또는 중간층
    - **입력층과 출력층 사이를 여러 층으로 연결, 뉴런이 여러 개 연결되어 '깊은 사고'를 하는 것을 의미**
    - **처리할 계산량이 기하급수적으로 늘어나 그만큼 정확도가 높아진 결과를 얻을 수 있음**
  - 깊은 딥러닝: 여러 층의 은닉층의 단계가 많아질수록

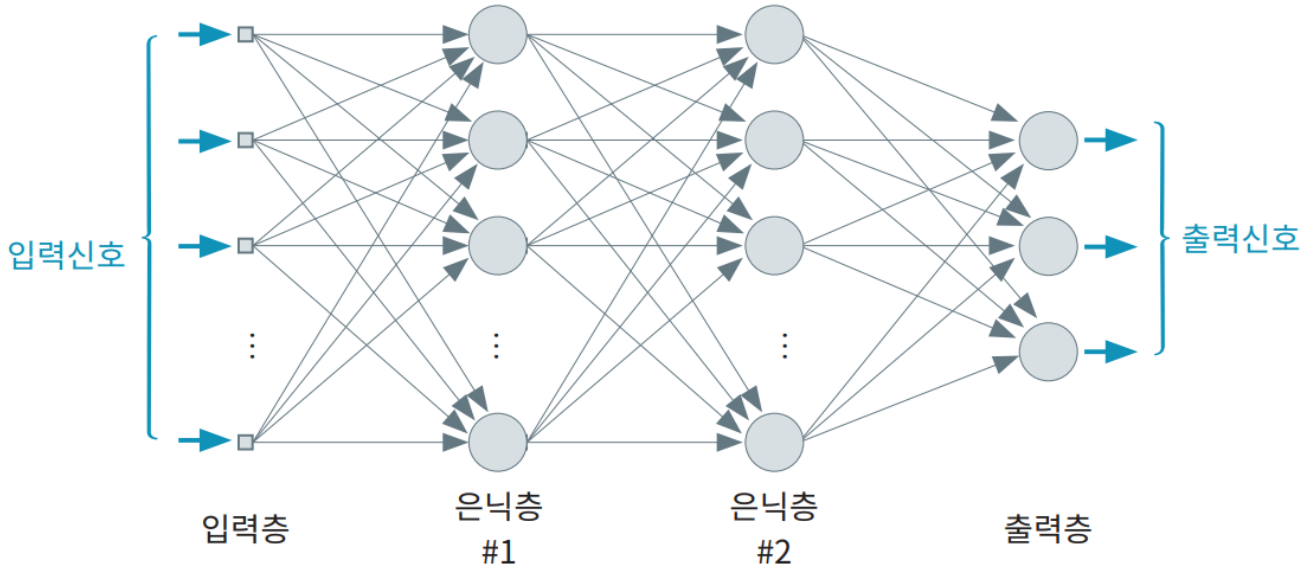


그림 6.57 ▸ 다층 퍼셉트론: MLP

아파트 가격을 예측하는 인공신경망 MLP

인공신경망은 입력층과 은닉층 그리고 1개의 예측 값이 있는 출력층으로 구성된 사례

- **은닉층으로 5개의 퍼셉트론으로 구성**
  - 아파트 가격이 출력으로 나오는 출력층

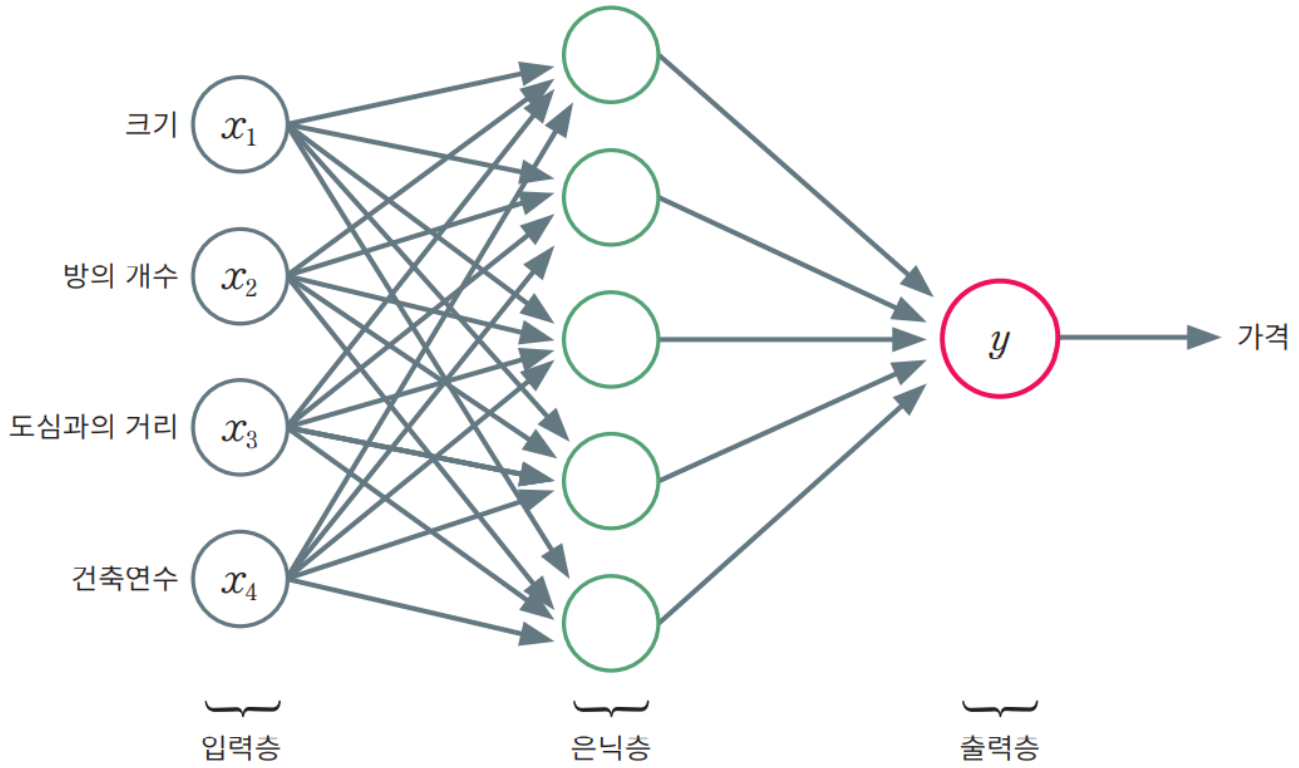
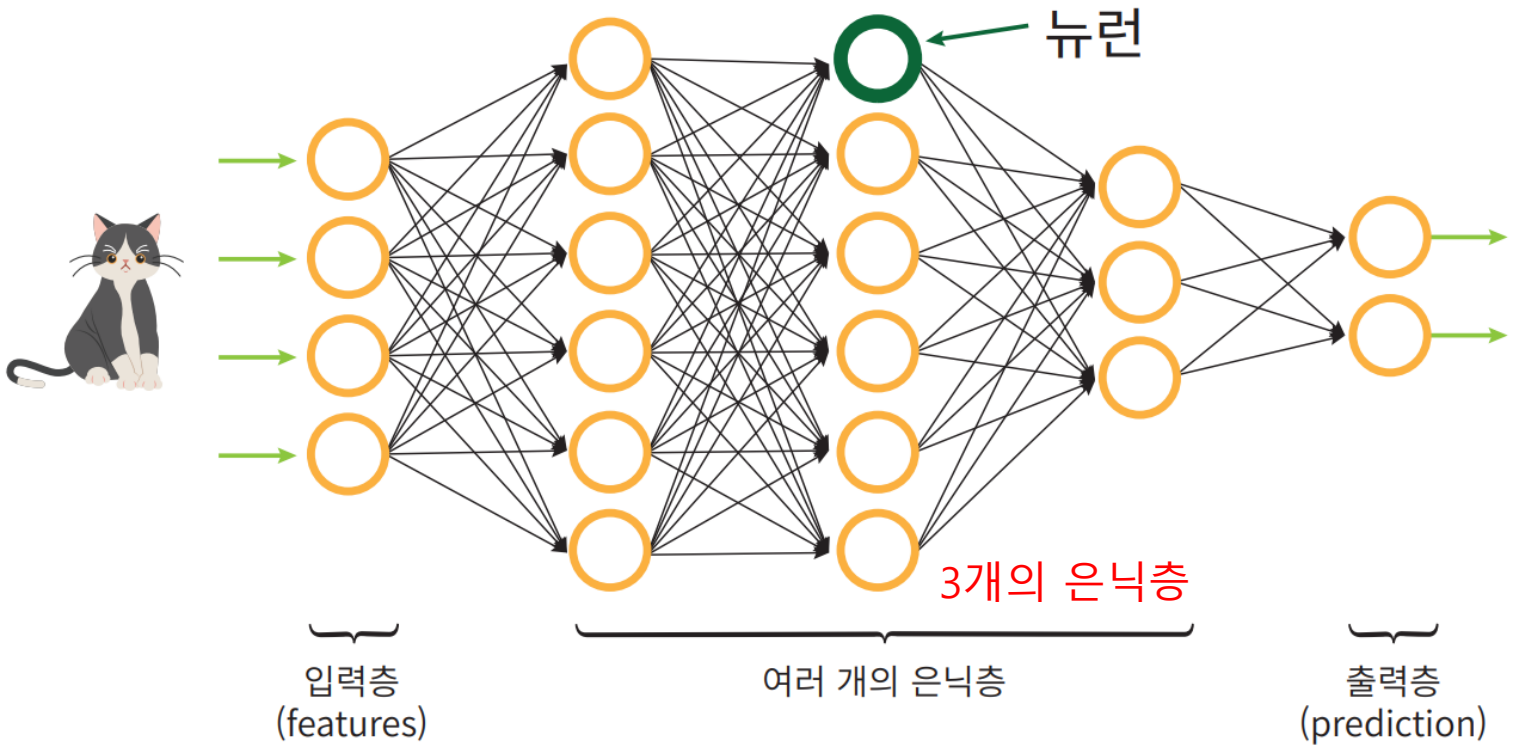


그림 6.58 ▸ 입력의 차원이 4인 선형 회귀 문제에서 5개의 퍼셉트론을 은닉층으로 구성한 인공신경망

딥러닝(deep learning)

심층신경망(DNN: Deep Neural Network)은 입력층(input layer)과 출력층(output layer) 사이에 여러 개의 은닉층(hidden layer)들로 이뤄진 인공신경망으로 딥러닝이라고 불리움

- **심층신경망(DNN: Deep Neural Network)**
  - 현재의 딥러닝은 적게는 몇 개에서 많게는 수백 개의 은닉층으로 구성
  - 다양한 활용 분야
    - **항공기나 드론, 자동차의 자율비행, 필체인식 또는 음성인식 등 다양하게 이용**



인공신경망 연산 과정

입력 특성(차원)이 n개이고 k개의 퍼셉트론으로 출력층을 구성하는 인공신경망을 표현

- **훈련 데이터에서 각각의 샘플이 입력층으로 입력**

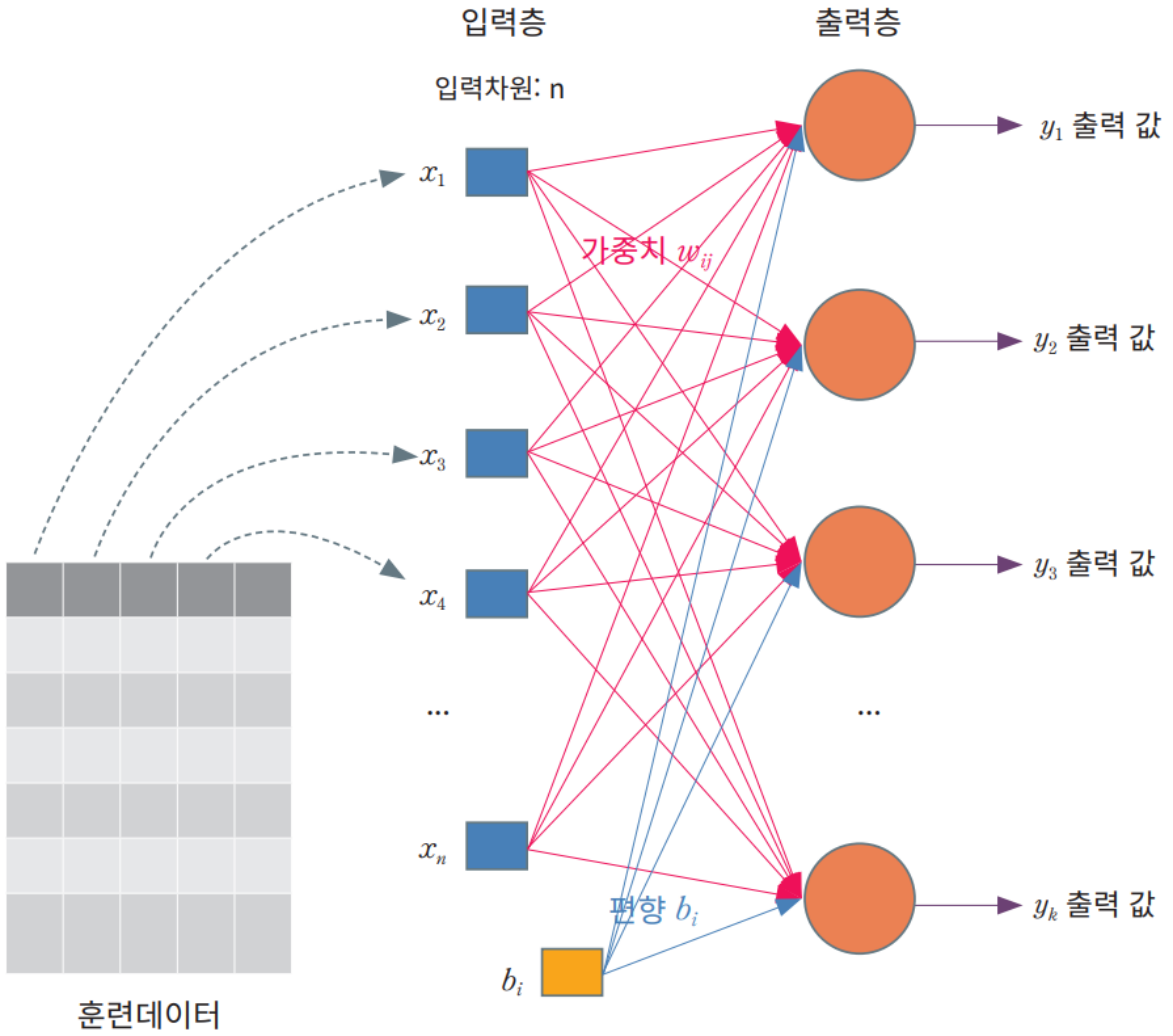


그림 6.60 ▸ 입력층과 출력층으로 구성된 인공신경망

선형 회귀와 인공신경망 모델

6개의 데이터로 학습시켜 2년 또는 12년 등의 원하는 연수의 월 급여를 예측

- **SW경력(연수)에 따른 월 급여 예측 사례**
  - '소프트웨어 개발 분야 급여 수준' 데이터
    - **산포도(scatter)**

소프트웨어 개발 분야 급여 수준 자료

		신입사원	초급기능사	고급기능사	초급기술자	고급기술자	기술사
월급여(백만원)	y	2.6	3.1	4.3	6.5	8.4	11.2
sw경력(년수)	x	0	3	5	8	10	13

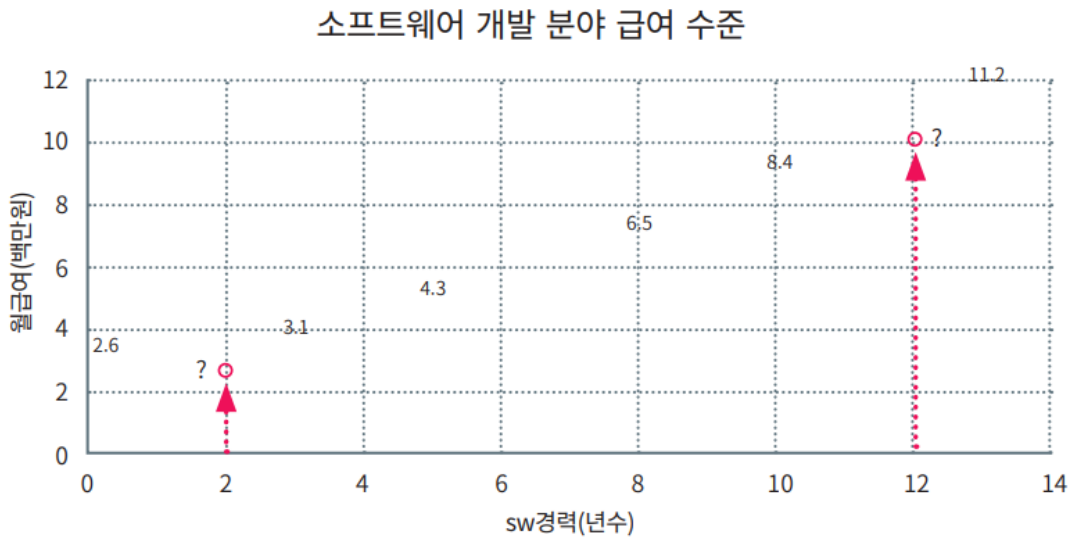


그림 6.61 ▸ '소프트웨어 개발 분야 급여 수준' 자료와 산포도

단순 선형회귀와 다중 선형회귀

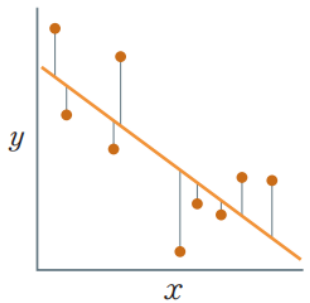
위 예와 같이 입력 속성이 x 하나인 선형회귀를 단순 선형회귀라 하며, '프로젝트 개발 경험'과 '관련 자격증' 등의 $x_1, x_2, x_3, \cdots$ 여러 입력 속성이 있는 경우 이를 다중 선형회귀라 한다.

회귀

단순 선형회귀	$y = b_0 + b_1 \times x_1$
다중 선형회귀	$y = b_0 + b_1 \times x_1 + b_2 \times x_2 + \cdots + b_n \times x_n$

종속 변수 == 결과 값

단순 선형회귀



다중 선형회귀
(2개의 독립 변수, x_1, x_2)

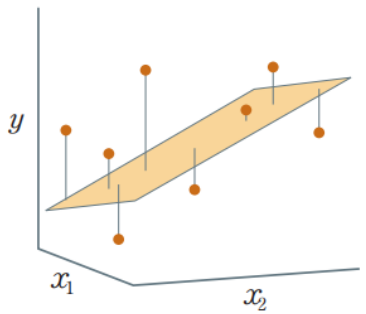


그림 6.64 ▸ 단순 선형회귀와 다중 선형회귀

손실함수의 이해

매개변수 w와 b가 결정된 4개의 직선 중 어느 직선이 훈련 데이터를 표현하고 다른 데이터를 예측하는 가장 좋은 직선일까?

- **수식 y=wx+b에서 어떻게 w와 b를 찾을 수 있을까?**
  - 데이터 지점(파란색 원)을 가장 근접하는 직선
    - **실제값 y와 예측값 $\hat{y}(y\ \text{hat})$ 차이인 오류를 최소화**
- **손실 함수(loss function)**
  - 비용 함수(cost function)
  - 학습의 목표가 되는 기준 지표의 계산식, 목적 함수(objective function)
    - **오류를 최소화하는 방향으로 학습을 수행**
  - 손실과 비용이라는 용어
    - **값이 작아질수록 목표에 접근**

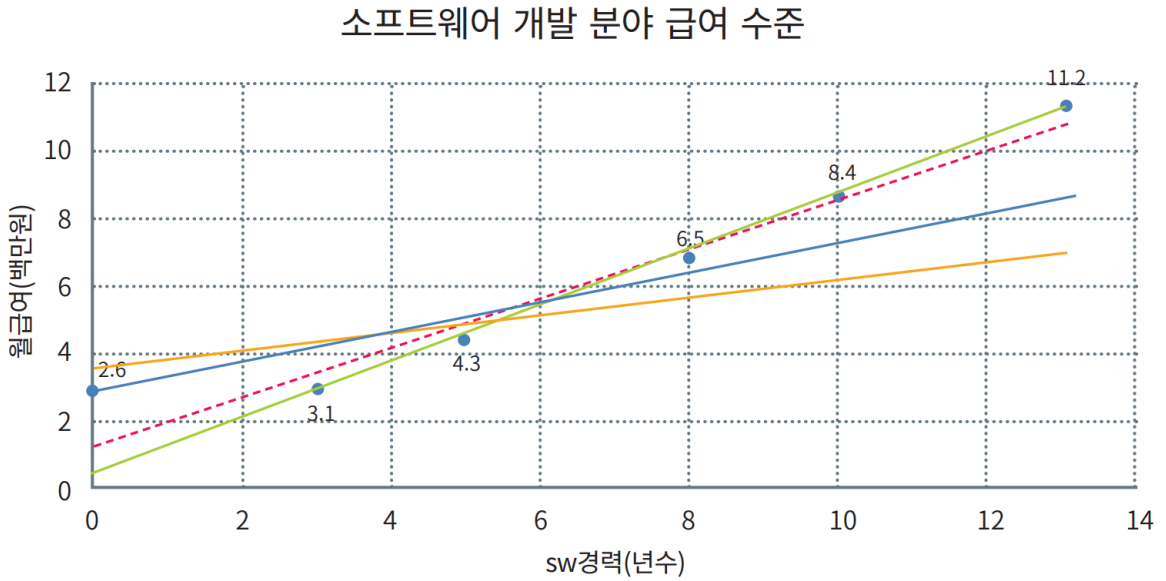


그림 6.65 ▸ 데이터를 표현하고 예측하는 여러 직선

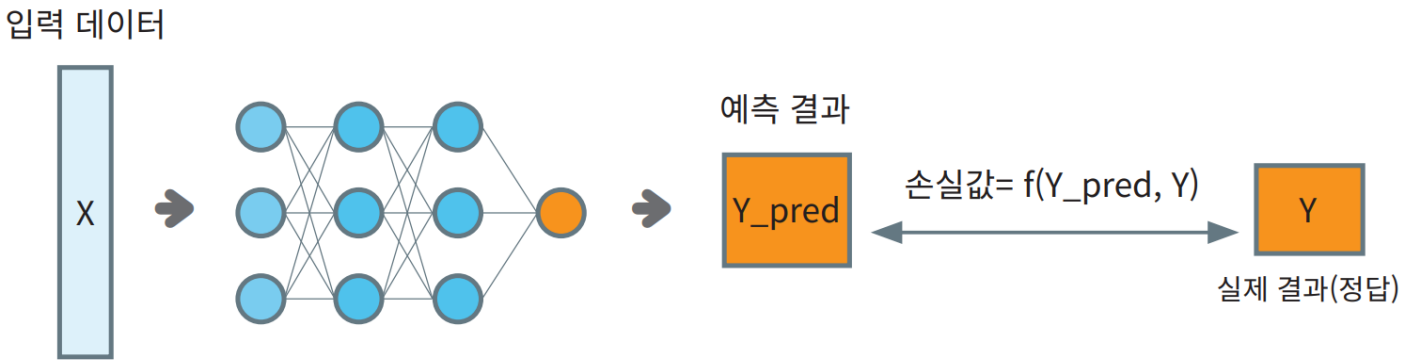


그림 6.66 ▸ 손실함수 개념

경사하강법: 손실 값을 줄이는 방법

인공신경망은 학습을 통해 지표인 손실 값을 줄이는 방향으로 가중치 w와 편향 b를 찾아야 함

- **경사하강법(傾斜下降法, gradient descent)**
  - 말 그대로 '경사 따라 내려가기' 방식
    - **기울기를 계산해 기울어진 방향으로 조금씩 이동하는 과정을 반복적으로 수행**
      - **기울기가 최소가 되는 지점으로 이동하는 방식**
  - 손실함수의 값이 최소가 되는 지점을 찾아 가는 방법

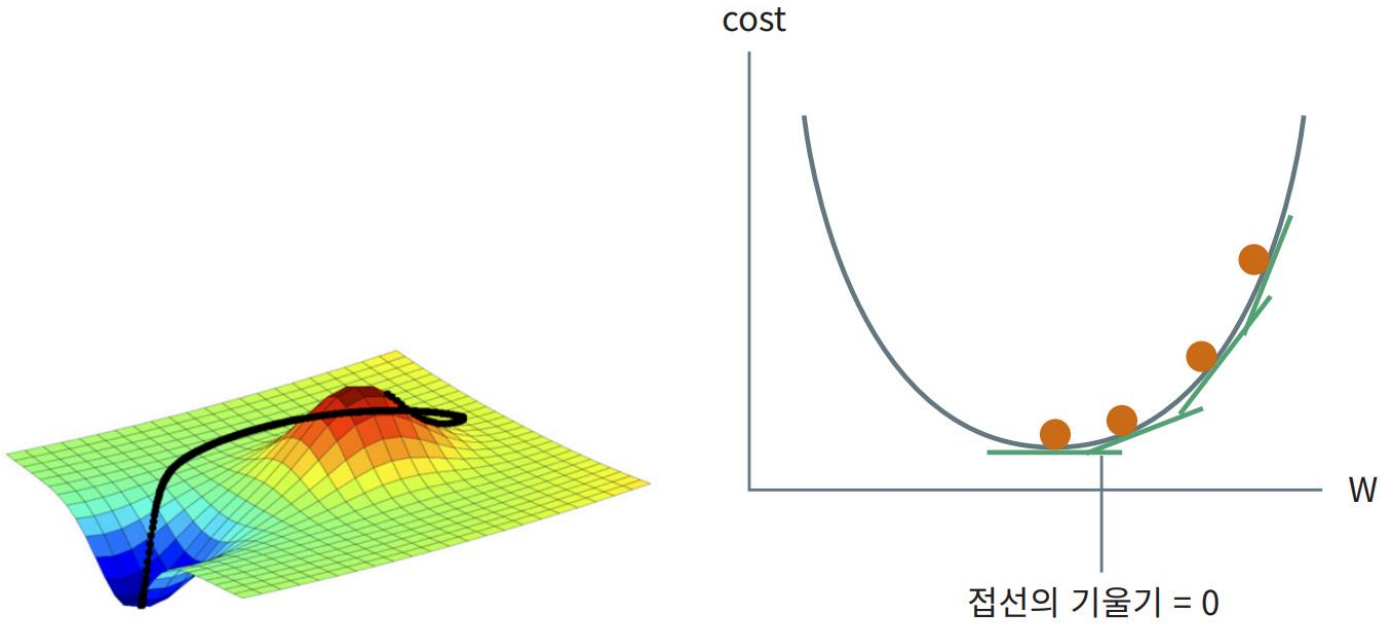
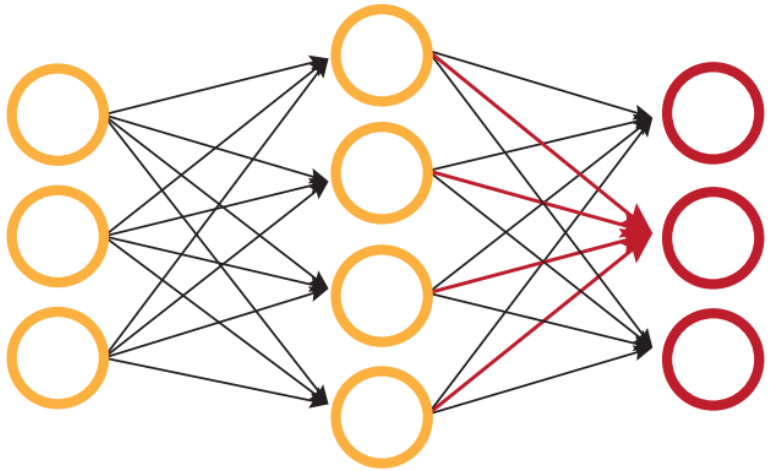


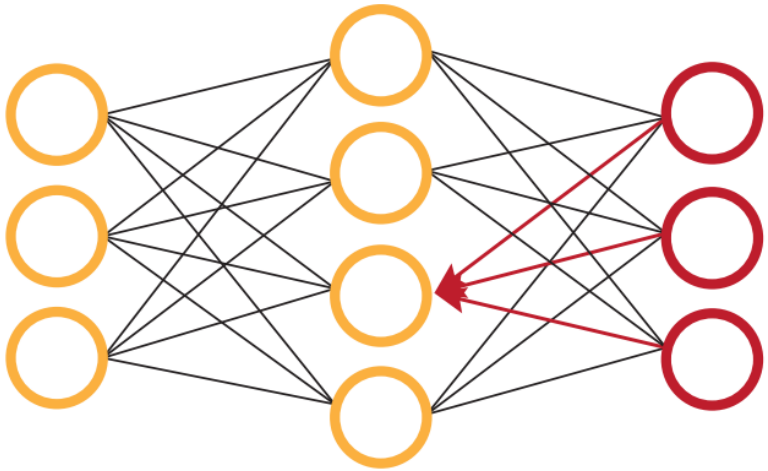
그림 6.67 ▸ 경사하강법 개념

순전파와 오차역전파

가중치 w를 수정하는 과정을 살펴보면 결과값과 실제값 사이의 오차를 구하고, 경사하강법에 따라 가중치를 수정하며, 오차가 더이상 줄어들지 않을 때까지 이 과정을 반복

- **순전파(forward propagation)**
  - 처음에는 임의의 초기 가중치 w로 주어진 입력으로 은닉층을 거쳐 결과값을 계산
  - 신경망에서 [입력층 → 은닉층 → 출력층]의 방향인 좌측에서 우측으로의 진행
- **오차역전파(error backpropagation)**
  - 가중치와 편향을 계산하는 방식(다변 함수의 미분법인 연쇄법칙(chain rule)을 사용)
    - **예측값과 실제값을 비교해 오차를 출력층에서 입력층으로 역으로 전파**
  - 1986년, 제프리 힌튼 교수에 의해 도입되어 심층신경망에서 실제 학습이 실현
    - **'후진 방식 자동 미분'이라는 수학 방식 사용**

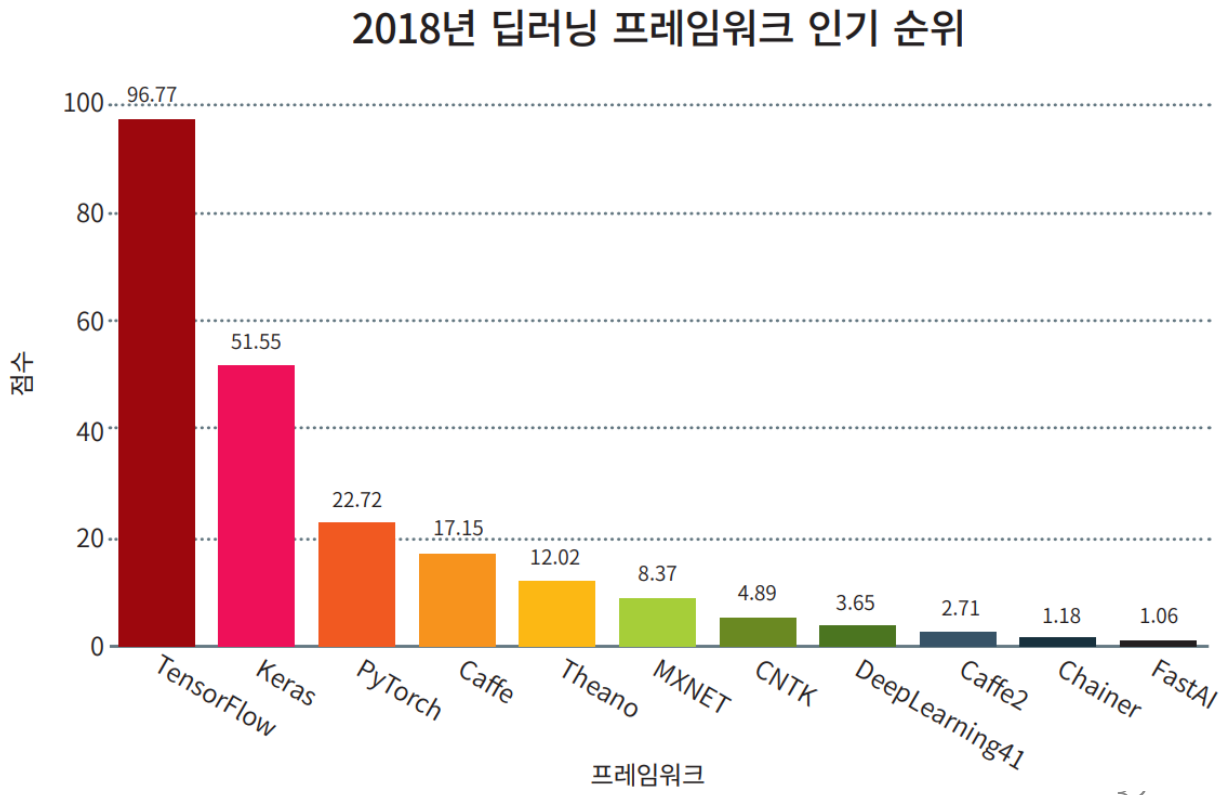




다양한 인공신경망 라이브러리

프레임워크는 툴킷(toolkit) 또는 라이브러리(library)라는 용어로도 사용

- **인공신경망을 위한 공개된 프레임워크(framework)는 매우 다양**
  - 오픈소스 소프트웨어는 소스가 공개되어 누구나 활용 가능
  - 구글의 텐서플로(tensorflow)와 페이스북의 파이토치(Pytorch)



구글의 텐서플로

원래 구글 브레인(brain) 팀이 구글 내부에서 연구와 제품 개발을 위한 목적으로 만들어 사용하다가 공개한 오픈소스 소프트웨어

- **구글이 공개**
  - 머신러닝과 딥러닝을 위한 오픈소스 플랫폼
    - **2015년 11월**
  - 인공신경망 중심의 머신러닝 구현 공개 라이브러리
  - 다차원 데이터인 텐서(tensor)와 연산의 흐름
    - **데이터 흐름 그래프(data flow graph)를 사용한 수치 연산**
  - 데스크탑, 서버 혹은 모바일 기기에서 실행
  - CPU(Central Processing Unit)나 GPU(Graphical Processing Unit)를 사용

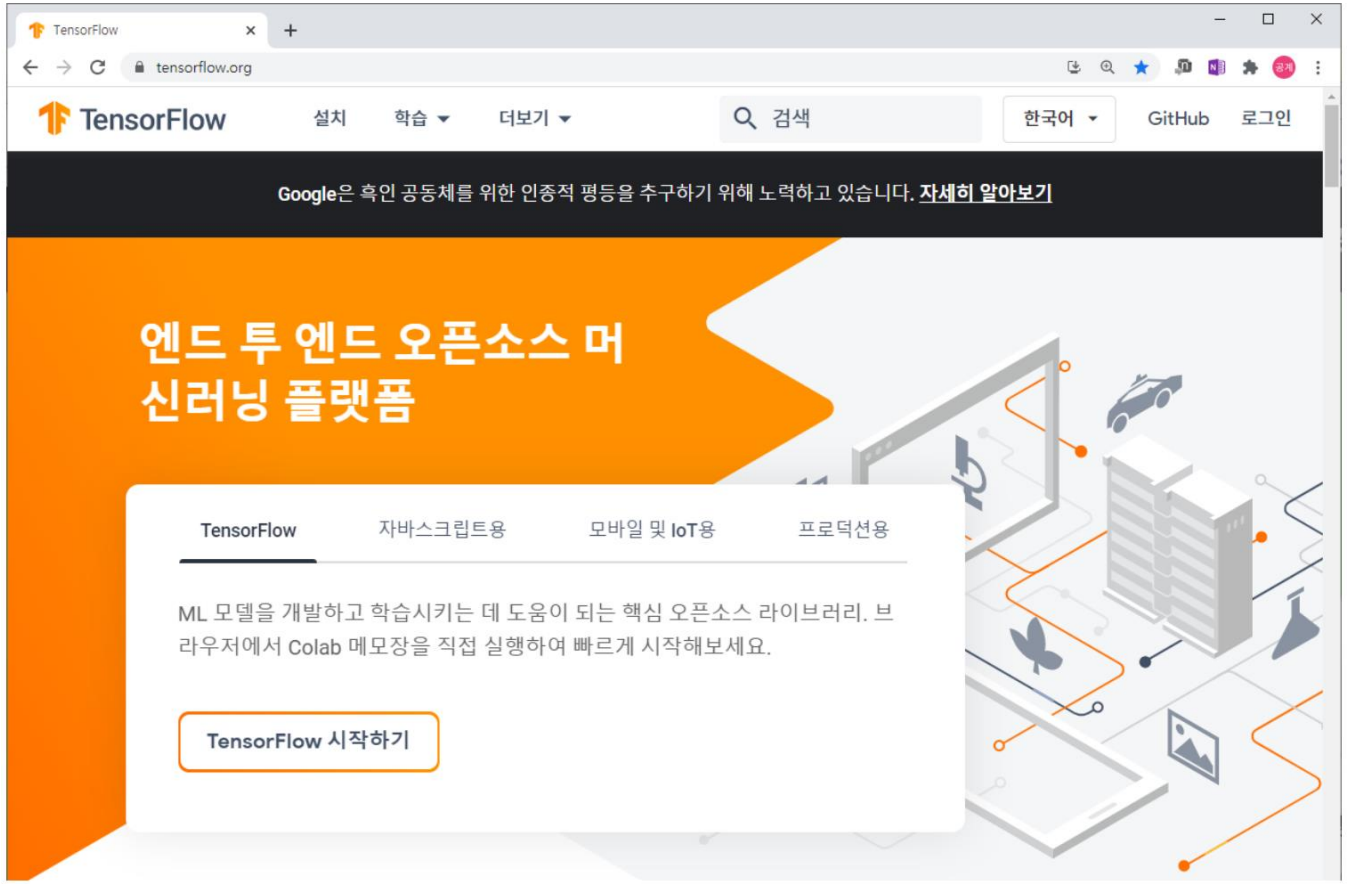


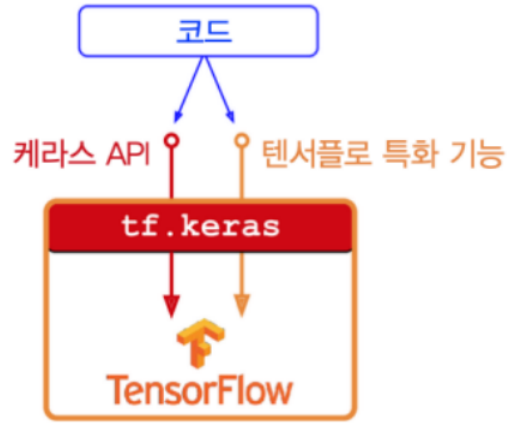
그림 6.71 ▸ 텐서플로 홈페이지(www.tensorflow.org)

케라스(Keras)

케라스(Keras)는 프랑소아 숄레(Francois Chollet)가 파이썬으로 개발한 오픈소스 라이브러리

- **고수준 API(Application Programming Interface) 라이브러리**
  - 간결하면서 직관적인 API를 제공
    - **초보자도 쉽게 이해**
    - **수리적으로 복잡한 딥러닝 이론을 모르더라도 인공지능 모델을 쉽게 구현**
  - 2017년 텐서플로 코어 라이브러리로 사용
    - **버전 2.0부터는 텐서플로의 고수준 API**
- **원래 몬트리올 대학의 시애노(theano) 딥러닝 엔진을 쉽게 사용하려고 개발**
  - 시애노 외에 다음에서도 사용 가능
    - **마이크로소프트의 인지 툴킷(CNTK)**
    - **아마존이 지원하는 아파치(Apache)의 맥엑스넷(MXNET)**
    - **인텔(Intel)이 주도하는 OpenCL의 PlaidML**

- **다양한 딥러닝 샘플 데이터 제공**
  - MNIST 손글씨 숫자 이미지, MNIST 패션
  - cifar10 및 cifar100 소형 컬러 이미지
  - IMDB 영화 리뷰
  - 로이터 뉴스와이어 주요 기사
  - 보스턴 주택 가격



파이토치(PyTorch)

컴퓨터 비전 및 자연어처리와 같은 응용 분야에 사용되는 토치(torch) 라이브러리를 기반으로 하는 파이썬을 위한 오픈소스 머신 러닝 라이브러리

- **페이스북 인공지능 연구팀(FAIR: Facebook AI Research)에서 개발**
  - 2016년에 오픈소스로 공개
    - **2018년 12월 1.0 버전을 정식 출시**
  - 풍부한 성능 개선
  - 모바일 플랫폼용 개발자 친화적인 지원
    - **최근 사용 비중이 높아지고 있음**
- **파이토치로 만들어진 대표적인 인공지능 소프트웨어**
  - 테슬라의 오토파일럿(Autopilot)
  - 우버의 파이로(Pyro)
- **OpenAI(openai.org)의 주개발 플랫폼**
  - 2015년 10월, 일론 머스크 등이 인류에게 이익을 주며 인간 친화적인 인공지능 연구를 목표로 설립

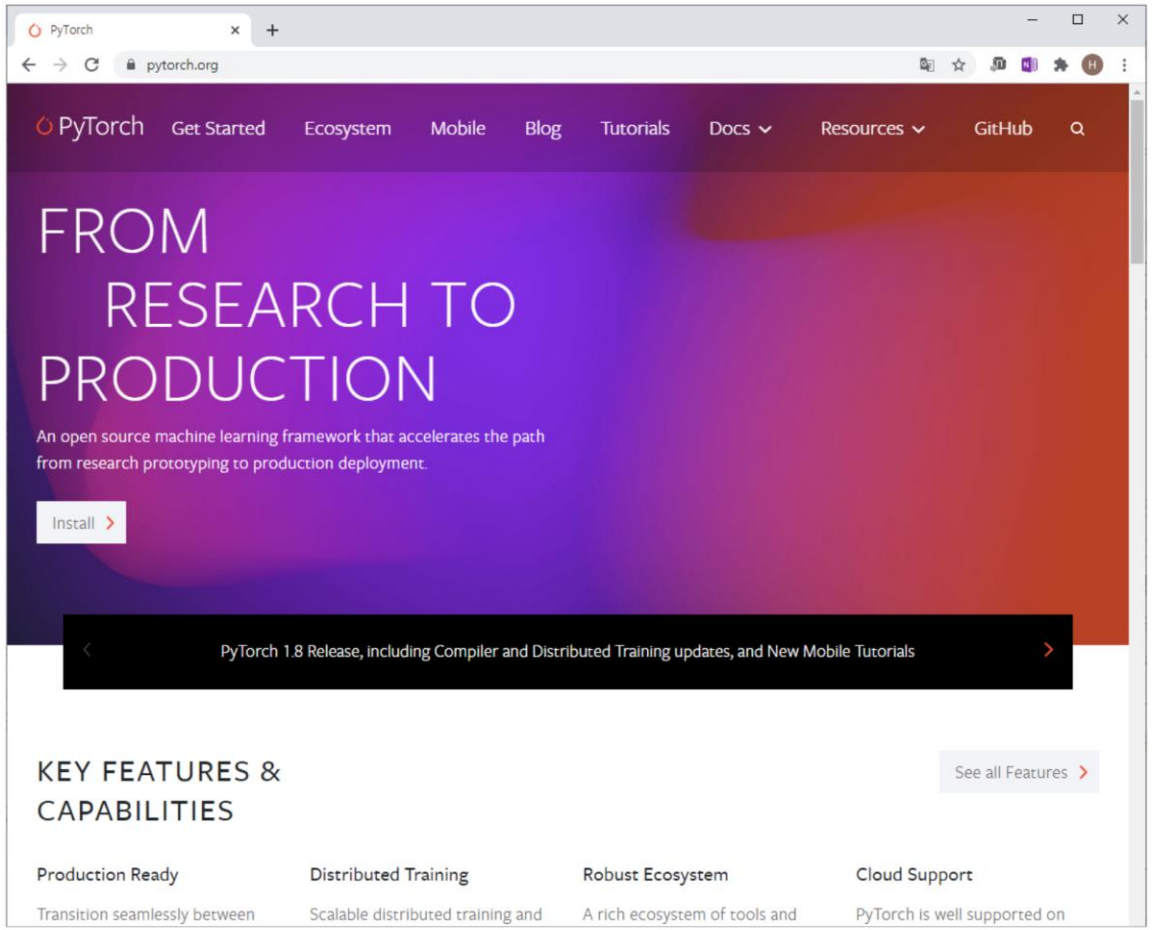


그림 6.74 ▸ 파이토치 홈페이지(pytorch.org)