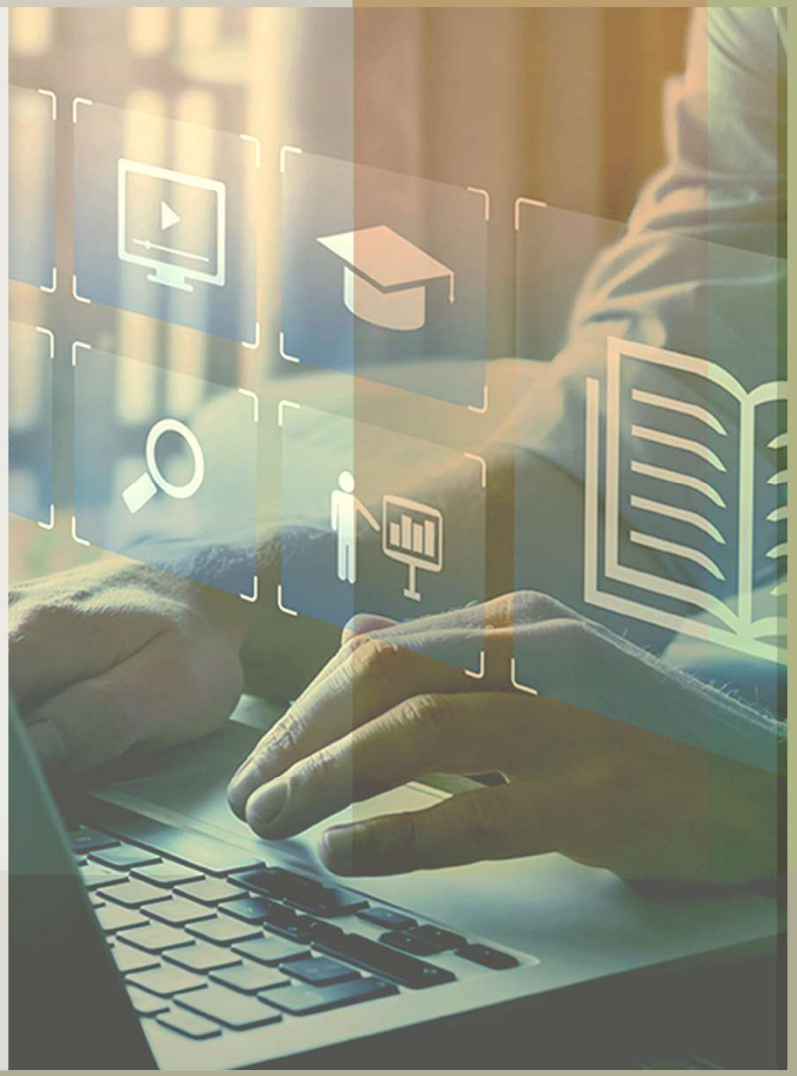
01

딥러닝

신경망의 개요

방송대 컴퓨터과학과 이병래 교수



학습목차

01

인공 신경망의 개념

1. 인공지능을 구현하기 위한 접근 방법

기호 인공지능

- 문제 및 지식을 기호 형식으로 표현함
- 논리적 추론, 탐색 등의 방법을 통해 문제를 해결함
- 적용 예: 지식기반 시스템, 전문가시스템, 온톨로지, 시맨틱 웹 등

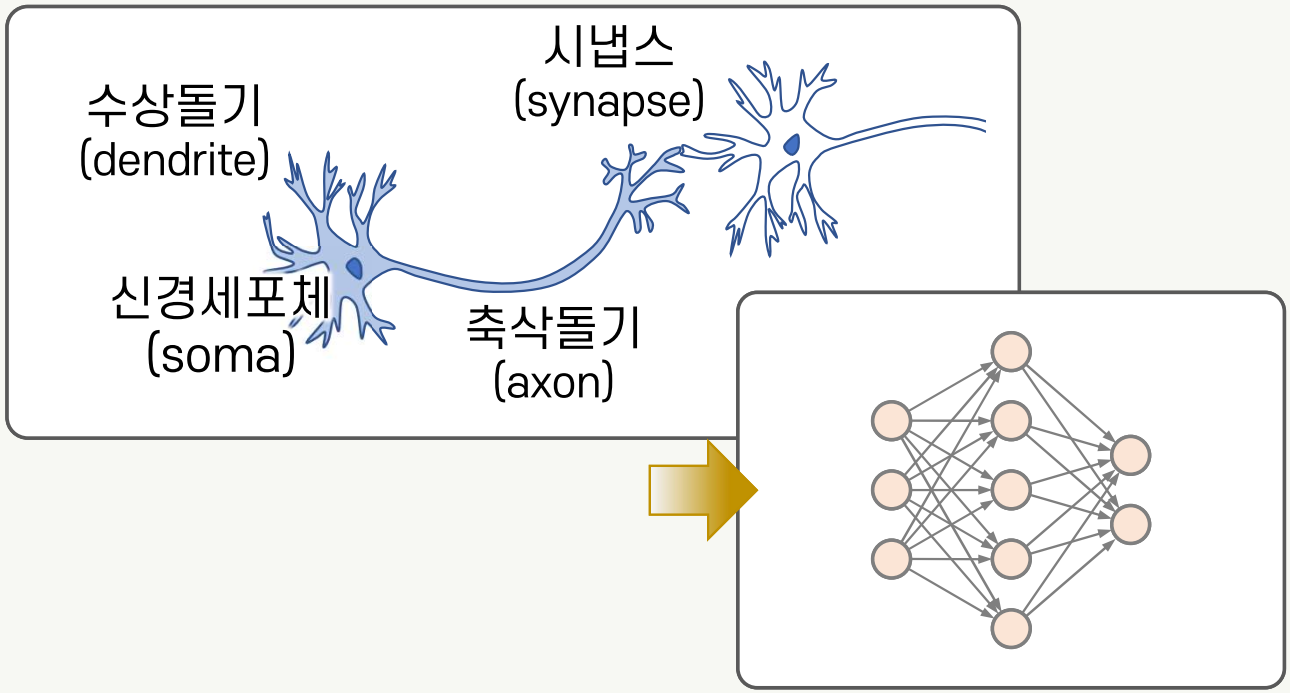
연결주의 인공지능

- 복잡하게 연결된 신경 구조에 착안한 모델
- 수치적으로 표현된 대량의 데이터를 기반으로 한 학습
- 적용 예: 딥러닝을 바탕으로 한 컴퓨터 시각, 의료 영상 분석, 음성 인식, 자연어 처리 등

2. 인공 신경망

인공 신경망(artificial neural networks)이란?

- 인간과 동물의 두뇌를 구성하는 생물학적 신경 시스템의 원리를 바탕으로 설계된 계산 시스템



2. 인공 신경망

초기의 신경망 연구

- 1943년 Warren McCulloch 등 : '임계치 논리(threshold logic)'라는 알고리즘을 바탕으로 한 신경망 계산 모델 제안
- 1949년 Donald Hebb : 헵의 학습(Hebbian learning) 이론 제시
- 1957년 Frank Rosenblatt: 퍼셉트론(Perceptron) 학습 모델
  - Mark 1 perceptron 구현
  - 이진 분류기를 학습할 수 있는 학습모델

> "걷고, 말하고, 보고, 쓰고, 스스로를 복제하고, 자신의 존재를 의식할 수 있을 것으로 기대되는 전자 컴퓨터의 배아(embryo)"
>
> - *The New York Times* (1958)

2. 인공 신경망

퍼셉트론의 한계와 역전파

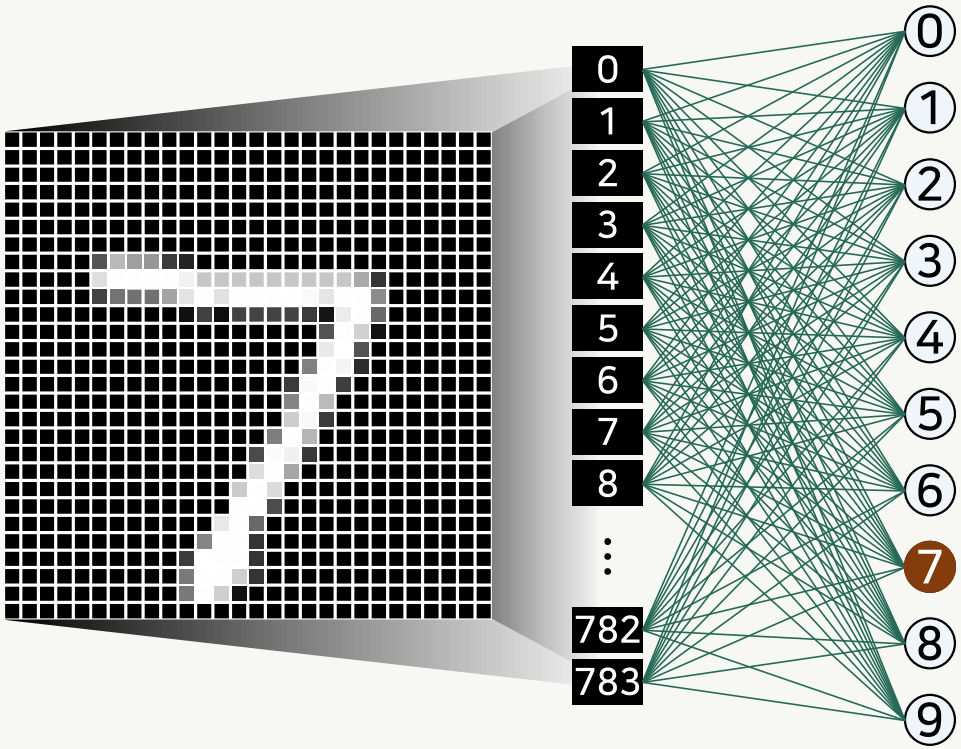
- 1969년 Marvin Minsky 등은 단층 퍼셉트론의 한계를 지적
  - 단층 퍼셉트론은 배타적 논리합(XOR)과 같은 간단한 문제도 해결하지 못함

  ⇨ 신경망의 연구가 침체기에 들어서게 됨

- 1974년 Paul Werbos, 1986년 David Rumelhart 등에 의해 다층 퍼셉트론 구조의 학습 방법이 발표됨
  - 역전파(backpropagation) 알고리즘

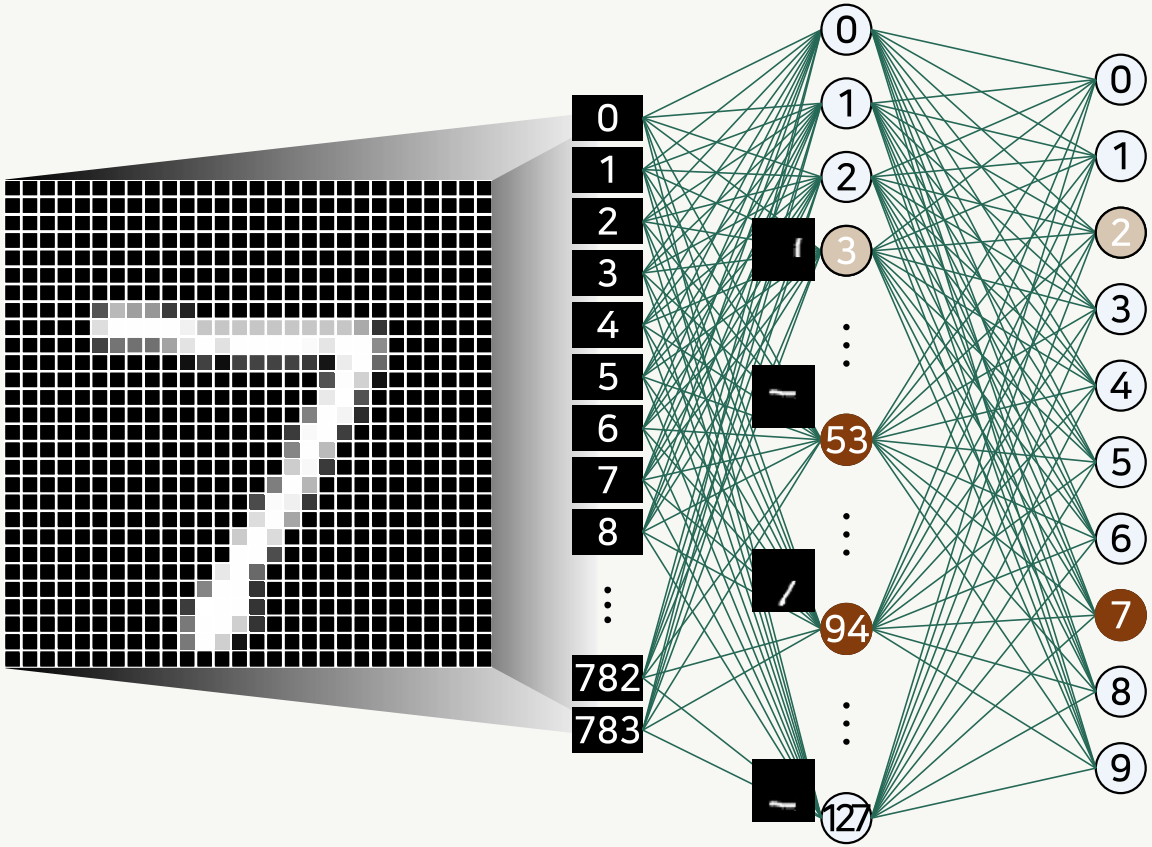
2. 인공 신경망

신경망의 층과 표현력



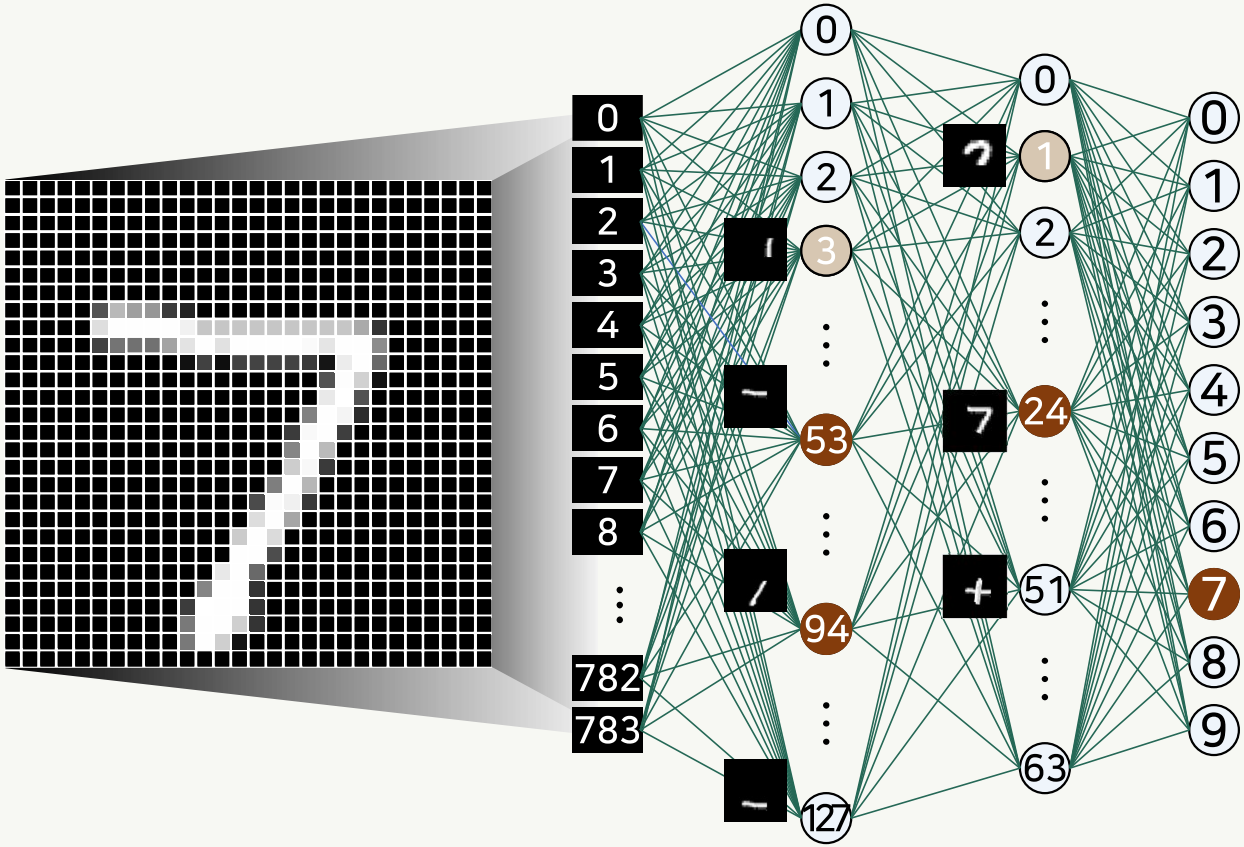
2. 인공 신경망

신경망의 층과 표현력



2. 인공 신경망

신경망의 층과 표현력



2. 인공 신경망

신경망의 층과 표현력

- 많은 수의 층으로 구성하면 더욱 높은 차원의 표현이 가능함
  - ➪ 심층 신경망(deep neural network)

딥러닝(deep learning)

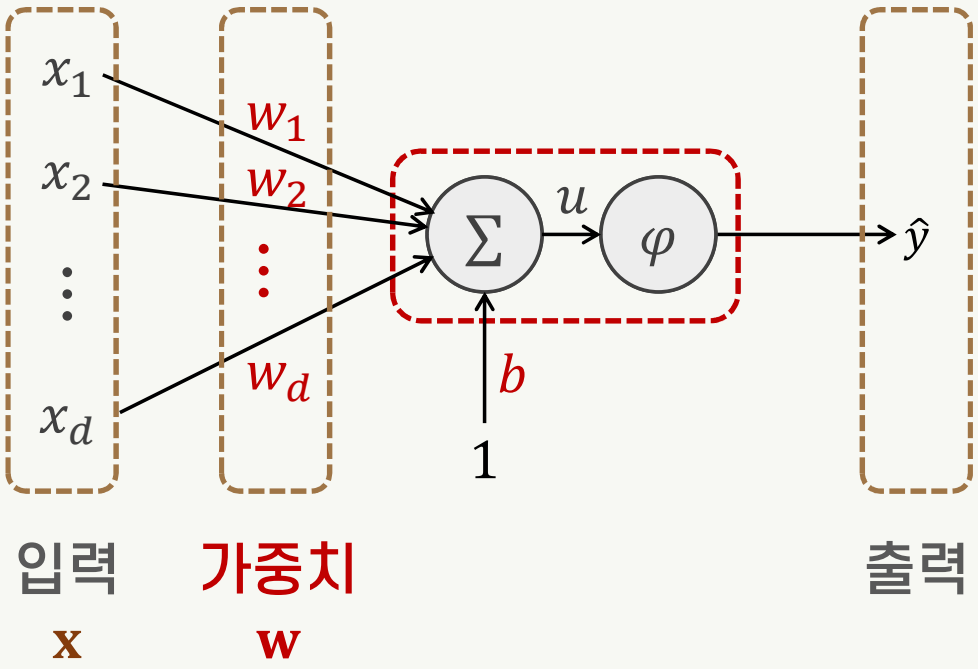
- 심층 신경망을 학습하기 위해 활용되는 기계학습 알고리즘
- 학습 성능을 제한하는 제반 문제의 개선이 필요함
  - 부족한 학습 데이터, 불안정한 경사, 과적합, 방대한 계산량

02

신경망의 기본 구조

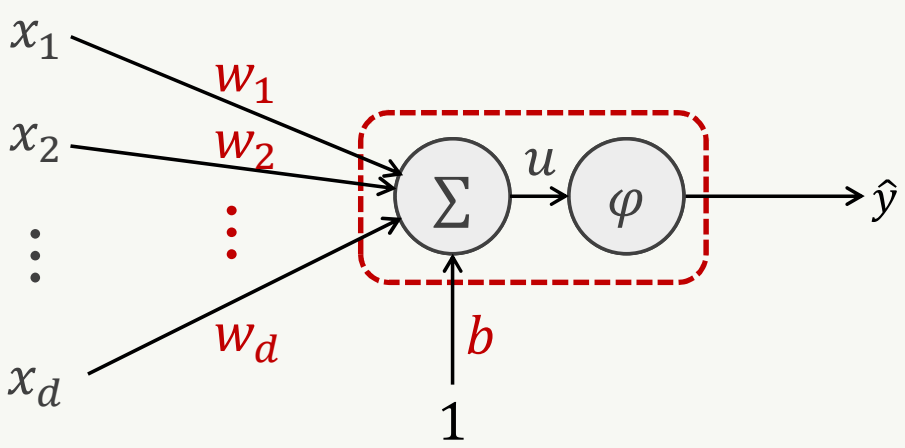
1. 뉴런의 기본 구조

인공 뉴런



1. 뉴런의 기본 구조

인공 뉴런



$$u = \sum_{i=1}^{d} w_i x_i + b$$

바이어스

활성함수

뉴런의 출력 ⇒ $\hat{y} = \varphi(u)$

1. 뉴런의 기본 구조

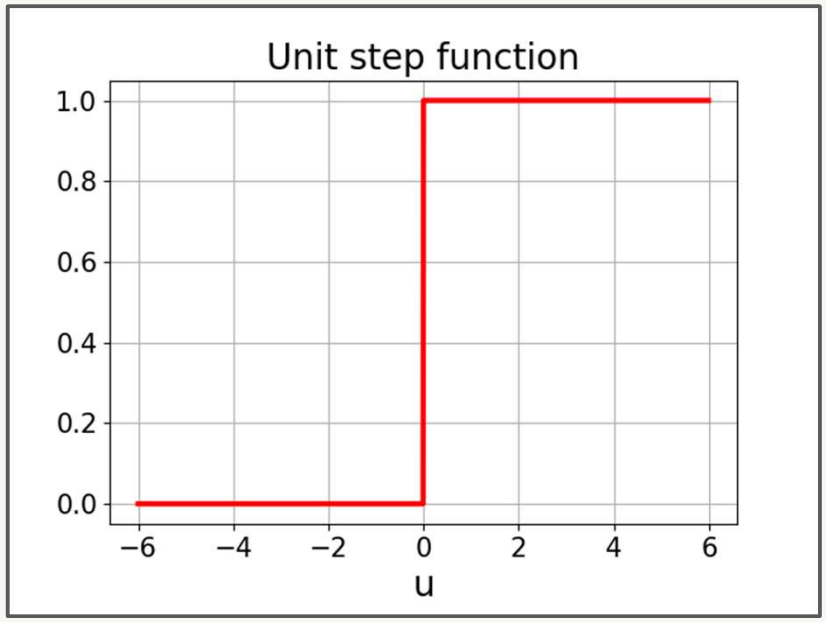
- 활성함수(activation function) $\varphi(u)$
  - 일반적으로 비선형 특성을 갖는 함수를 사용함
  - u의 값이 0보다 작으면 출력을 억제하고 0보다 크면 출력을 내도록 설계함
- 바이어스(bias)
  - 뉴런이 활성화되는 $\mathbf{w}^T\mathbf{x}$의 레벨을 조정함

2. 활성함수의 종류

계단 함수

- 정의역을 유한한 개수의 구간으로 나누어서 각 구간에서 상수인 함수
- 단위 계단 함수(unit step function, Heaviside step function)

$$\varphi_{step}(u) = \begin{cases} 1, & u \geq 0 \\ 0, & u < 0 \end{cases}$$



2. 활성함수의 종류

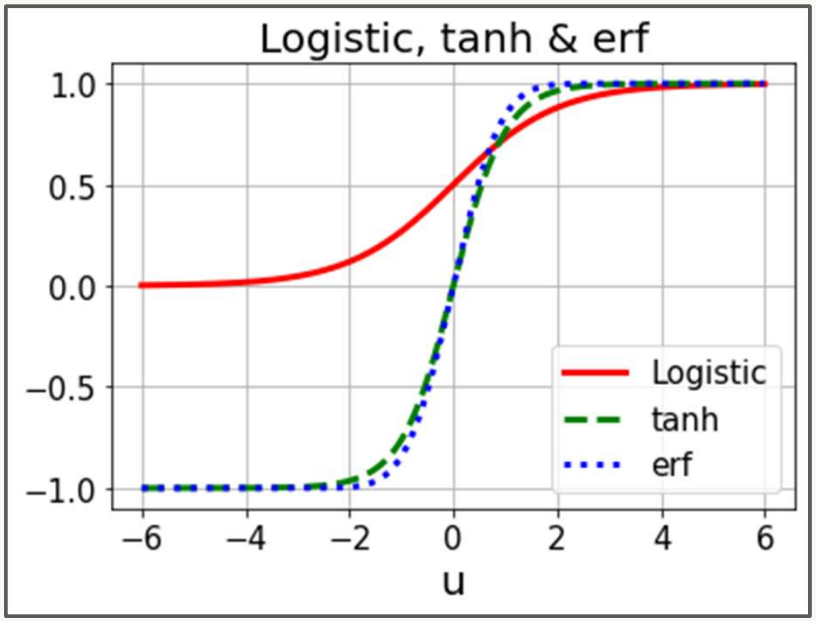
시그모이드 함수

- ‘S’자 형태의 곡선 함수
- 로지스틱(logistic) 함수, 쌍곡탄젠트(tanh), 오차 함수(erf) 등

$$\varphi_{Logistic}(u) = \frac{1}{1+e^{-u}}$$

$$\varphi_{tanh}(u) = \frac{e^{u}-e^{-u}}{e^{u}+e^{-u}}$$

$$\varphi_{erf}(u) = \frac{2}{\sqrt{\pi}}\int_{0}^{u} e^{-t^2}dt$$



모든 u에 대해 미분 가능

‘경사 소멸’ 문제의 원인이 됨

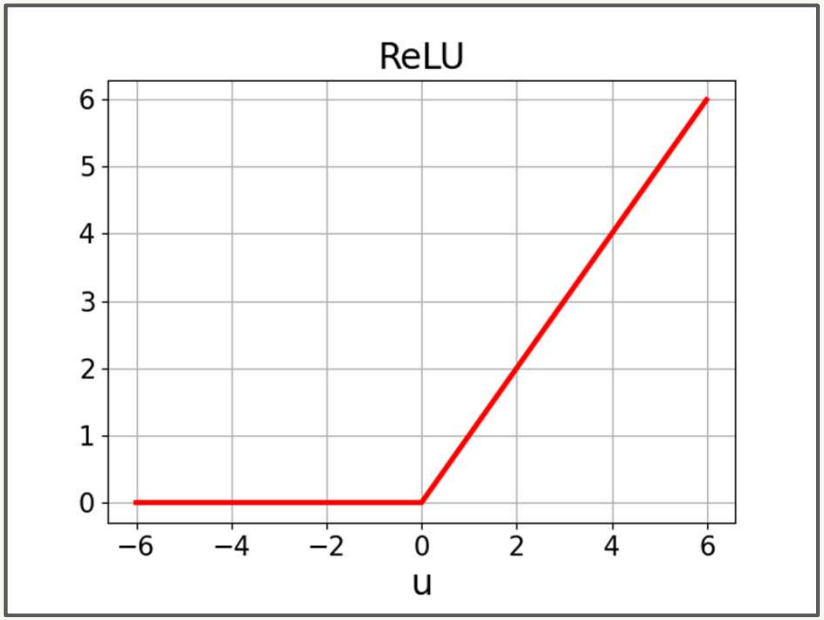
2. 활성함수의 종류

ReLU

- 심층망에서 경사 소멸 문제를 개선하기 위해 시그모이드의 대안으로 활용되는 활성함수

$$\varphi_{ReLU}(u) = \begin{cases} 0, & u < 0 \\ u, & u \geq 0 \end{cases}$$

또는 $\varphi_{ReLU}(u) = \max(0, u)$



문제점

- $u = 0$에서 미분 불가
- 'Dying ReLU' 문제

2. 활성함수의 종류

ReLU의 변형

Leaky ReLU

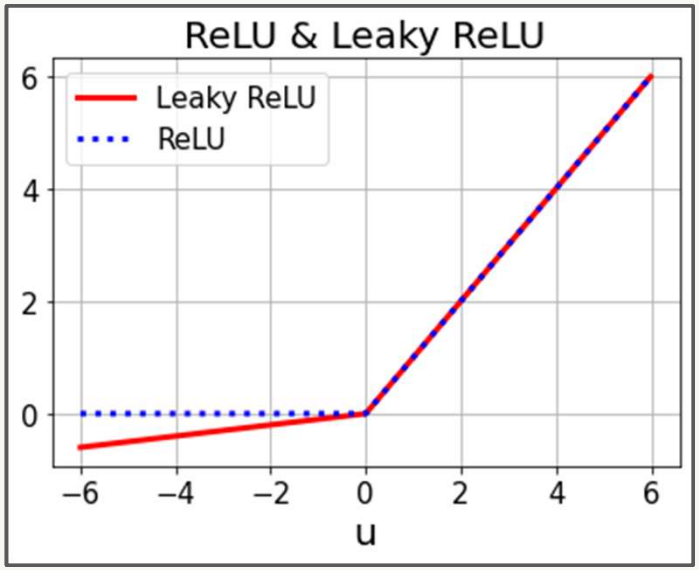
$$\varphi_{LeakeReLU}(u) = \max(\alpha u, u)$$

α: 1보다 작은 양수(예: 0.1)

PReLU(parametric ReLU)

$$\varphi_{PReLU}(u) = \max(\alpha u, u)$$

α: 학습에 의해 결정되게 함



2. 활성함수의 종류

ReLU의 변형

- **GELU(Gaussian-error linear unit)**

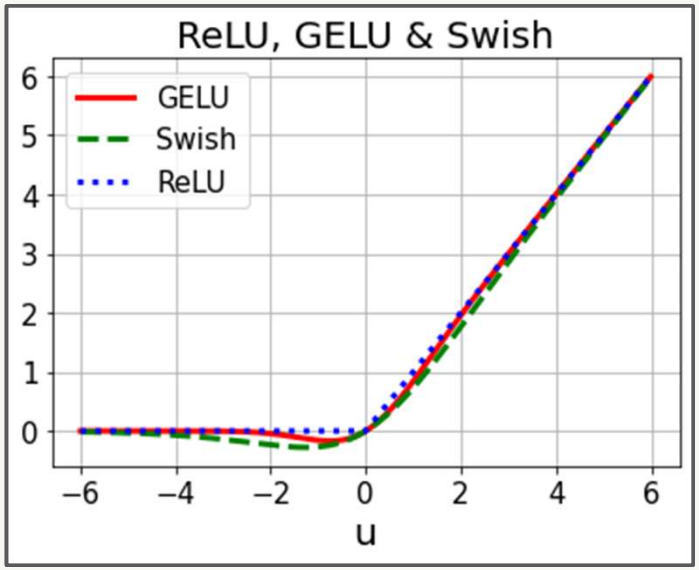
$$\varphi_{GELU}(u) = u \cdot \Phi(u),$$

  - $\Phi(u)$: 표준 정규분포의 누적분포함수

- **Swish**

$$\varphi_{swish}(u) = u \cdot \varphi_{Logistic}(\beta u)$$

  - $\beta = 1$인 경우 SiLU(sigmoid LU)

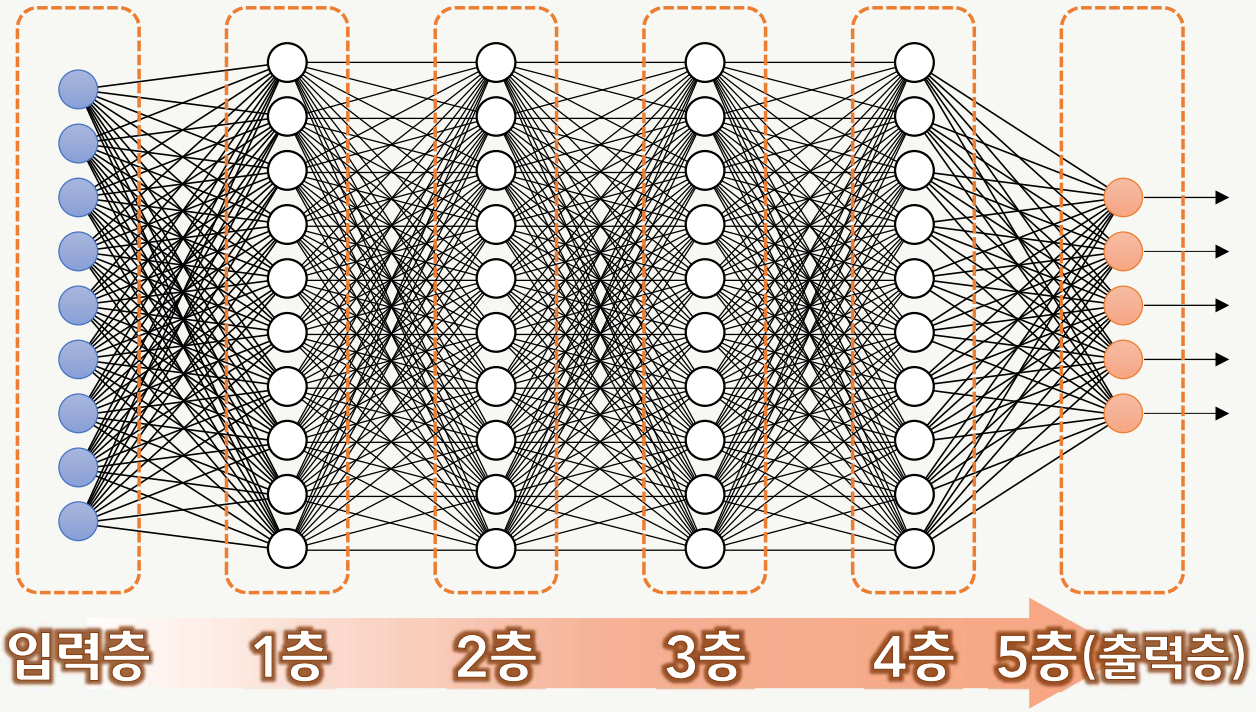


03

단층 피드포워드 신경망

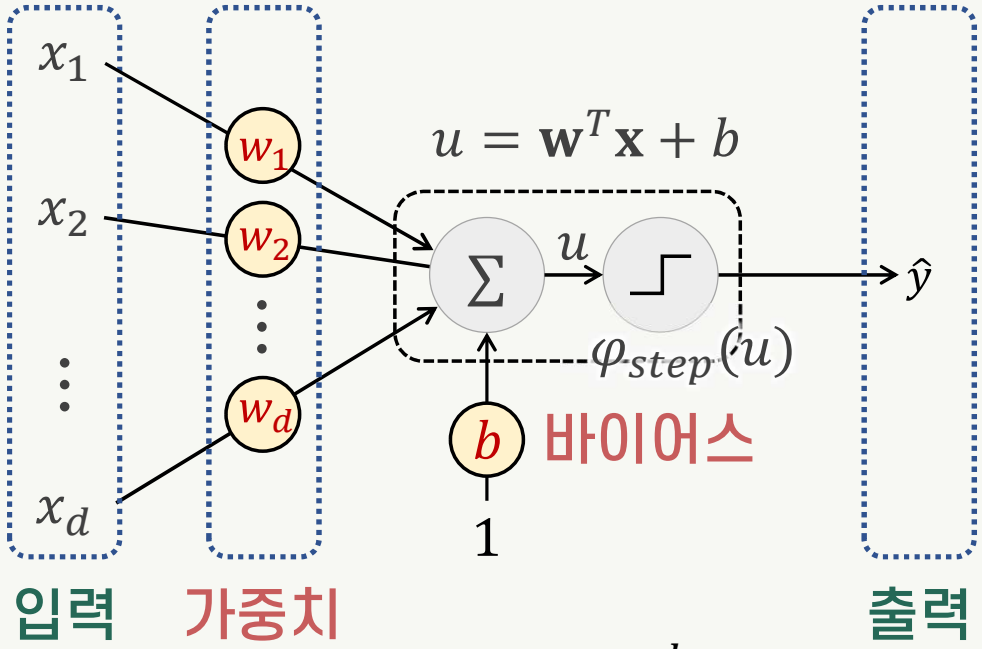
1. 피드포워드 신경망

- 입력층에서 출력층 방향으로 뉴런 층의 연결이 이루어지는 구조



2. 단층 퍼셉트론(Single-layer perceptron)

- 하나의 층으로 구성된 가장 기본적인 피드포워드 신경망 구조



$$\hat{y} = \varphi_{step}(u), \quad u = \sum_{i=1}^{d} w_i\, x_i + b$$

2. 단층 퍼셉트론(Single-layer perceptron)

퍼셉트론의 학습

- 지도학습 방식으로 학습 : 학습표본 $(\mathbf{x}, y)$의 집합
  - $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_d]^T$: 입력
  - y : 레이블
- 학습 대상 파라미터
  - $\mathbf{w} = [w_1 \quad w_2 \quad \cdots \quad w_d]^T$: 입력이 뉴런에 전달되는 연결의 가중치
  - b : 바이어스

2. 단층 퍼셉트론(Single-layer perceptron)

퍼셉트론의 학습

- Step1. 파라미터(w와 b) 초기화 - 작은 크기의 랜덤 값으로 초기화함
- Step2. 모든 학습표본을 대상으로 파라미터 업데이트를 반복

1. t번 반복 업데이트된 $\mathbf{w}(t)$와 $b(t)$를 이용하여 k번째 학습표본 $\mathbf{x}^{(k)}$에 대한 출력 $\hat{y}^{(k)}$를 예측함

$$\hat{y}^{(k)} = \varphi_{step}\left(\sum_{i=1}^{d} w_i(t)x_i^{(k)} + b(t)\right)$$

2. 오차 $\delta = \hat{y}^{(k)} - y^{(k)}$와 입력의 곱에 비례하여 $\mathbf{w}$와 b를 업데이트함

$$w_i(t+1) = w_i(t) - \eta\delta x_i^{(k)}, \qquad i = 1,2,\cdots,d$$
$$b(t+1) = b(t) - \eta\delta$$

η: 학습률

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

피셔의 붓꽃 데이터 집합(Fisher's Iris flower data set)

- Edgar Anderson이 수집한 붓꽃 데이터 집합
- 'setosa', 'virginica,' 'versicolor'라는 세 종류 붓꽃의 특징 데이터
  - 특징 : 꽃받침(sepal)과 꽃잎(petal)의 길이와 폭
  - 각 종류별로 50개의 데이터로 구성됨

setosa

virginica

versicolor

(사진 출처: 위키백과)

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

실습 프로그램의 구성 - 함수 목록

- prepare_data(target)
  - 붓꽃 데이터의 특징과 레이블을 각각 numpy 배열 형태로 반환함
    - 특징 : 꽃잎의 길이와 폭
    - 레이블 : target에 지정된 종류의 붓꽃이면 1, 그 외의 종류이면 0으로 지정
- step(x)
  - 단위 계단 함수 : x가 0 이상이면 1, 그렇지 않으면 0을 반환

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

실습 프로그램의 구성 - 함수 목록

- `visualize(net, X, y, multi_class, labels, class_id, colors, xlabel, ylabel, legend_loc='lower right')`
  - 결과의 그래프를 출력하는 시각화 함수
  - `net` : 학습된 퍼셉트론
  - `X, y` : 특징 및 레이블의 배열
  - `multi_class` : 3개 이상의 클래스로 분류하는 경우 `True`
  - `labels` : 클래스 레이블 리스트
  - `class_id` : 클래스 이름을 출력할 스트링 리스트
  - `colors` : 클래스를 구분할 색상 리스트
  - `xlabel, ylabel` : x, y축에 표시할 레이블
  - `legend_loc` : 범례를 표시할 위치

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

실습 프로그램의 구성 - 클래스 선언

- class Perceptron
  - 퍼셉트론 객체를 만들기 위한 클래스
  - 인스턴스 변수
    - self.dim : 입력층 입력의 수
    - self.activation : 활성함수
    - self.w : 가중치
    - self.b : 바이어스

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

실습 프로그램의 구성 - 클래스 선언

- class Perceptron
  - 메소드 목록
    - __init__(self, dim, activation) : 퍼셉트론 객체 초기화
    - printW(self) : 가중치 및 바이어스 값 출력
    - predict(self, x) : 입력 표본 x에 대한 퍼셉트론의 출력
    - fit(self, X, y, N, ephocs, eta=0.01) : 주어진 학습표본 집합을 이용하여 퍼셉트론 객체를 훈련함

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [1] 필요한 패키지 불러오기

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3  from sklearn.datasets import load_iris
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [2] 데이터 준비 함수 정의하기

```
1  def prepare_data(target):
2      iris = load_iris()          # iris data set 읽기
3      X_tr = iris.data[:, 2:]     # 4개의 특징 중 꽃잎의 길이와 폭 선택
4      labels = iris.target_names # 'setosa', 'versicolor', 'virginica'
5      y = iris.target
6
7      # 학습표본의 레이블 지정 - target에 지정된 레이블이면 1, 그 외는 0
8      y_tr = []
9      for i in range(150):
10         y_tr.append(labels[y[i]] == target)
11     y_tr = np.array(y_tr, dtype=int)
12     return X_tr, y_tr, ['(1) '+target, '(0) the others']
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [3] 활성함수 - 단위 계단 함수

```
1  def step(x):
2      return int(x >= 0)
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [4] 퍼셉트론 클래스 선언

```
1  class Perceptron():
2      def __init__(self, dim, activation):
3          rnd = np.random.default_rng()
4          self.dim = dim
5          self.activation = activation
6          # 가중치(w)와 바이어스(b)를 He normal 방식으로 초기화
7          self.w = rnd.normal(scale = np.sqrt(2.0 / dim), size=dim)
8          self.b = rnd.normal(scale = np.sqrt(2.0 / dim))
9
10     def printW(self):
11         for i in range(self.dim):
12             ......
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [4] 퍼셉트론 클래스 선언

```
10      def printW(self):
11          for i in range(self.dim):
12              print('  w{} = {:6.3f}'.format(i+1, self.w[i]), end='')
13          print('  b = {:6.3f}'.format(self.b))
14
15      def predict(self, x):   # numpy 배열 x에 저장된 표본의 출력 계산
16          return np.array([self.activation(np.dot(self.w, x[i]) + self.b)
17                           for i in range(len(x))])
18
19      def fit(self, X, y, N, epochs, eta=0.01):
20          # 학습표본의 인덱스를 무작위 순서로 섞음
21          ......
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [4] 퍼셉트론 클래스 선언

```
19      def fit(self, X, y, N, epochs, eta=0.01):
20          # 학습표본의 인덱스를 무작위 순서로 섞음
21          idx = list(range(N))
22          np.random.shuffle(idx)
23          X = np.array([X[idx[i]] for i in range(N)])
24          y = np.array([y[idx[i]] for i in range(N)])
25
26          f = 'Epochs = {:4d}    Loss = {:8.5f}'
27          print('w의 초깃값  ', end='')
28          self.printW()
29          for j in range(epochs):
30              ......
```

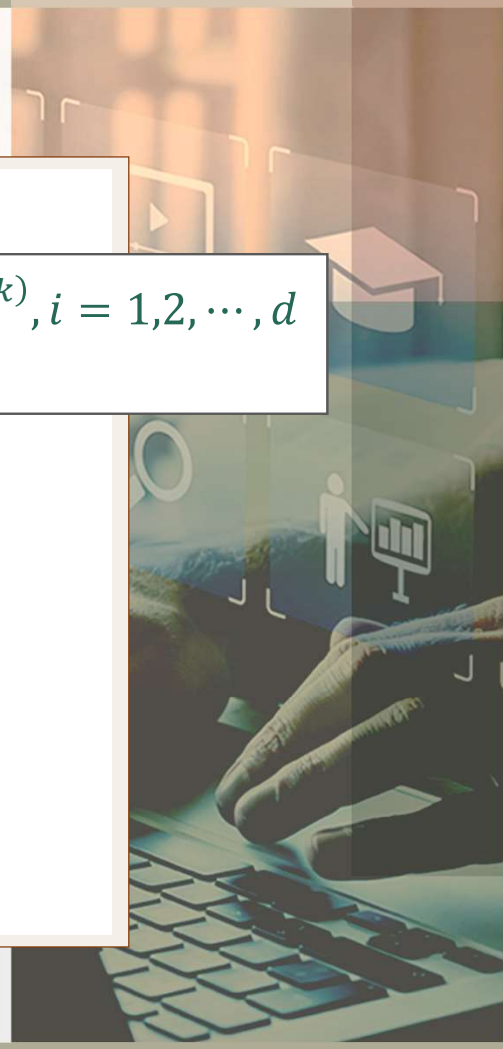
3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [4] 퍼셉트론 클래스 선언

```
10      def fit(self, X, y, N, epochs, eta=0.01):
...         ......
29          for j in range(epochs):
30              for i in range(N):
31                  # X[i]에 대한 출력 오차 계산
32                  delta = self.predict([X[i]])[0] - y[i]
33                  self.w -= eta * delta * X[i]
34                  self.b -= eta * delta
35              # 학습 과정 출력
36              if j < 10 or (j+1) % 100 == 0:
37                  loss = self.predict(X) - y
38                  ......
```

$$\delta = \hat{y}^{(k)} - y^{(k)}$$

$$w_i(t+1) = w_i(t) - \eta\delta x_i^{(k)}, i = 1,2,\cdots,d$$
$$b(t+1) = b(t) - \eta\delta$$



3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [4] 퍼셉트론 클래스 선언

```
10      def fit(self, X, y, N, epochs, eta=0.01):
...         ......
29          for j in range(epochs):
30              for i in range(N):
31                  # X[i]에 대한 출력 오차 계산
...                 ......
35              # 학습 과정 출력
36              if j < 10 or (j+1) % 100 == 0:
37                  loss = self.predict(X) - y
38                  loss = (loss * loss).sum() / N
39                  print(f.format(j+1, loss), end='')
40                  self.printW()
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
1  def visualize(net, X, y, multi_class, labels, class_id, colors,
2                xlabel, ylabel, legend_loc='lower right'):
3      # 데이터의 최소~최대 범위를 0.05 간격의 좌표값으로 나열
4      x_max = np.ceil(np.max(X[:, 0])).astype(int)
5      x_min = np.floor(np.min(X[:, 0])).astype(int)
6      y_max = np.ceil(np.max(X[:, 1])).astype(int)
7      y_min = np.floor(np.min(X[:, 1])).astype(int)
8      x_lin = np.linspace(x_min, x_max, (x_max-x_min)*20+1)
9      y_lin = np.linspace(y_min, y_max, (y_max-y_min)*20+1)
10
11     # x_lin과 y_lin의 격자좌표의 x와 y 값 구하기
12     x_mesh, y_mesh = np.meshgrid(x_lin, y_lin)
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
def visualize(net, X, y, multi_class, labels, class_id, colors,
              xlabel, ylabel, legend_loc='lower right'):
    .......
    # (x, y) 좌표의 배열로 만들어 신경망의 입력 구성
    X_test = np.column_stack([x_mesh.ravel(), y_mesh.ravel()])

    # 학습된 신경망으로 X_test에 대한 출력 계산
    if multi_class:
        y_hat = net.predict(X_test)
        y_hat = np.array([np.argmax(y_hat[k])
                          for k in range(len(y_hat))], dtype=int)
    else:
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
 1  def visualize(net, X, y, multi_class, labels, class_id, colors,
 2                xlabel, ylabel, legend_loc='lower right'):
...     ......
17      # 학습된 신경망으로 X_test에 대한 출력 계산
18      if multi_class:
19          y_hat = net.predict(X_test)
20          y_hat = np.array([np.argmax(y_hat[k])
21                            for k in range(len(y_hat))], dtype=int)
22      else:
23          y_hat = (net.predict(X_test) >= 0.5).astype(int)
24          y_hat = y_hat.reshape(len(y_hat))
25      ......
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
def visualize(net, X, y, multi_class, labels, class_id, colors,
              xlabel, ylabel, legend_loc='lower right'):
    ......
    # 출력할 그래프의 수평/수직 범위 설정
    plt.xlim(x_min, x_max)
    plt.ylim(y_min, y_max)

    # 클래스별로 산점도 그리기
    for c, i, c_name in zip(colors, labels, class_id):
        # 격자 좌표의 클래스별 산점도
        plt.scatter(X_test[y_hat == i, 0], X_test[y_hat == i, 1],
            ......
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
def visualize(net, X, y, multi_class, labels, class_id, colors,
              xlabel, ylabel, legend_loc='lower right'):
    .......
    # 클래스별로 산점도 그리기
    for c, i, c_name in zip(colors, labels, class_id):
        # 격자 좌표의 클래스별 산점도
        plt.scatter(X_test[y_hat == i, 0], X_test[y_hat == i, 1],
                    c = c, s = 5, alpha = 0.3, edgecolors = 'none')
        # 학습 표본의 클래스별 산점도
        plt.scatter(X[y == i, 0], X[y == i, 1],
                    c = c, s = 20, label=c_name)
     .......
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [5] 퍼셉트론의 분류 결과 시각화

```
1   def visualize(net, X, y, multi_class, labels, class_id, colors,
2                 xlabel, ylabel, legend_loc='lower right'):
...     .......
35          # 학습 표본의 클래스별 산점도
36          plt.scatter(X[y == i, 0], X[y == i, 1],
37                      c = c, s = 20, label=c_name)
38      # 범례의 표시 위치 지정
39      plt.legend(loc=legend_loc)
40      # x축과 y축의 레이블을 지정한 후 그래프 출력
41      plt.xlabel(xlabel, size=12)
42      plt.ylabel(ylabel, size=12)
43      plt.show()
```

3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [6] 훈련 데이터 준비하기

```
1  nSamples = 150
2  nDim = 2
3  target = 'setosa'          # 식별하고자 하는 붓꽃 종류 지정
4  X_tr, y_tr, labels = prepare_data(target)
```

코드 1-1 [7] 퍼셉트론 객체 생성 및 학습

```
1  p = Perceptron(nDim, activation=step)
2  p.fit(X_tr, y_tr, nSamples, epochs=1000, eta=0.01)
```

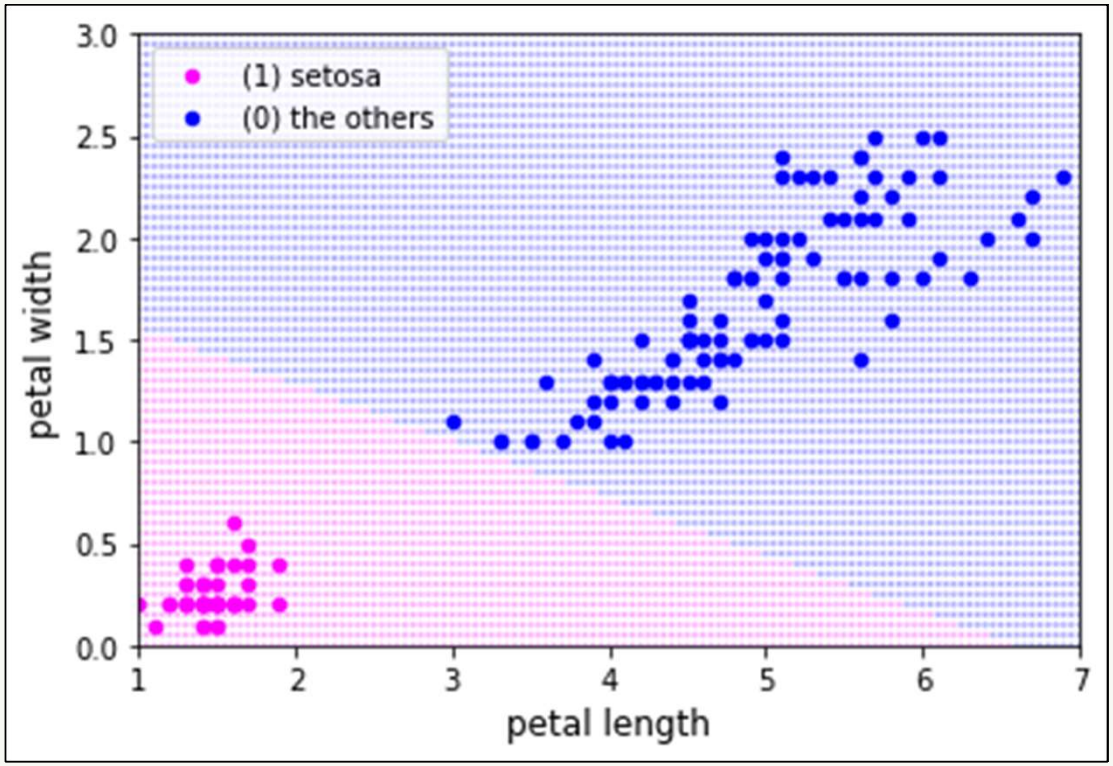
3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

코드 1-1 [8] 특징 공간 결정 영역 시각화

```
visualize(p, X_tr, y_tr,
          multi_class=False,
          class_id=labels,
          labels=[1, 0],
          colors=['magenta', 'blue'],
          xlabel='petal length',
          ylabel='petal width',
          legend_loc='upper left')
```

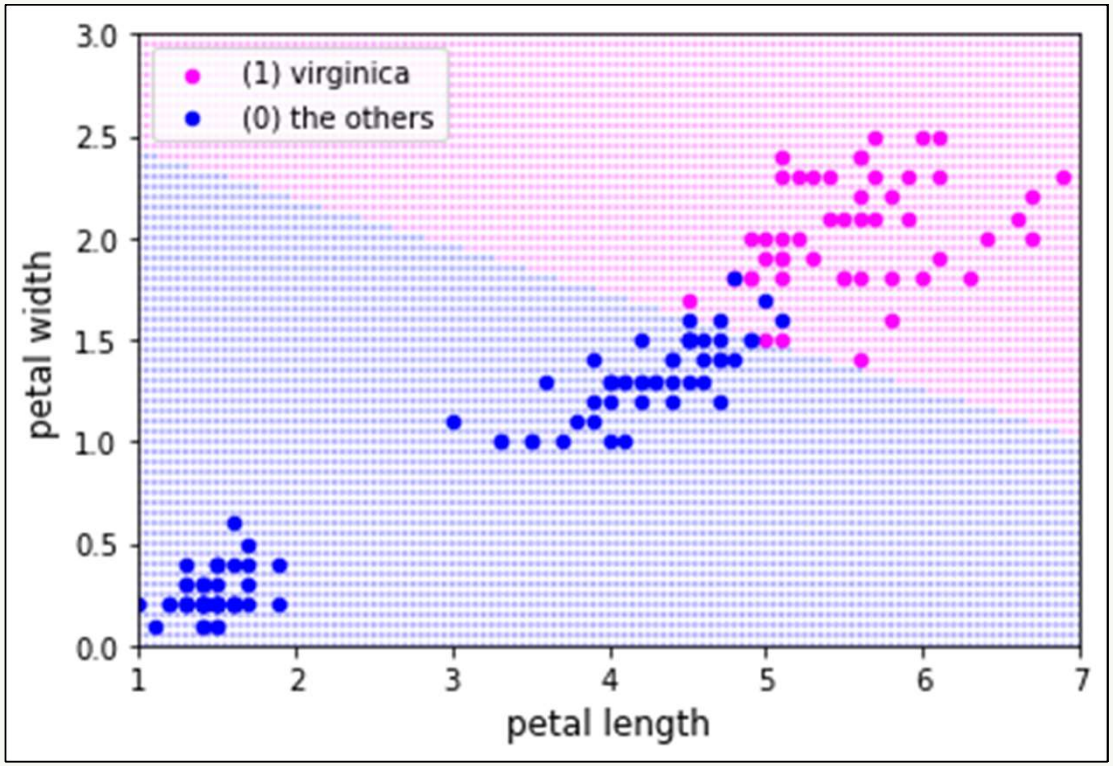
3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

- ‘setosa’를 식별하도록 학습된 퍼셉트론의 결정경계



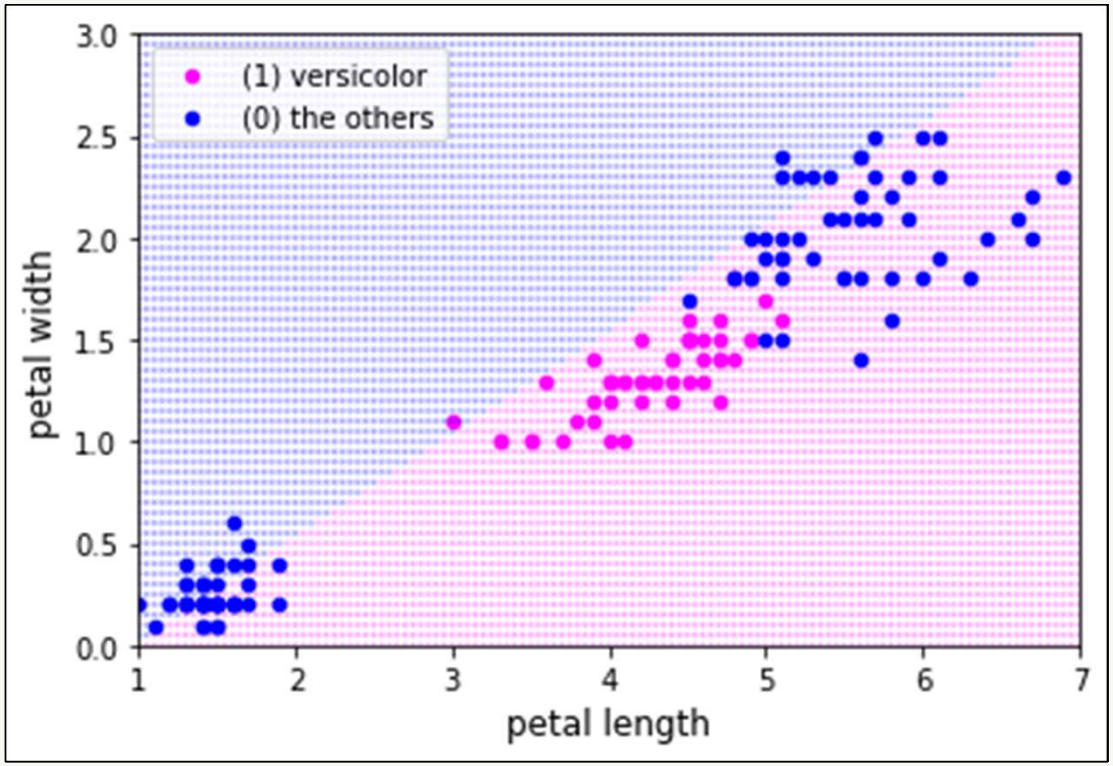
3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

- ‘virginica’를 식별하도록 학습된 퍼셉트론의 결정경계



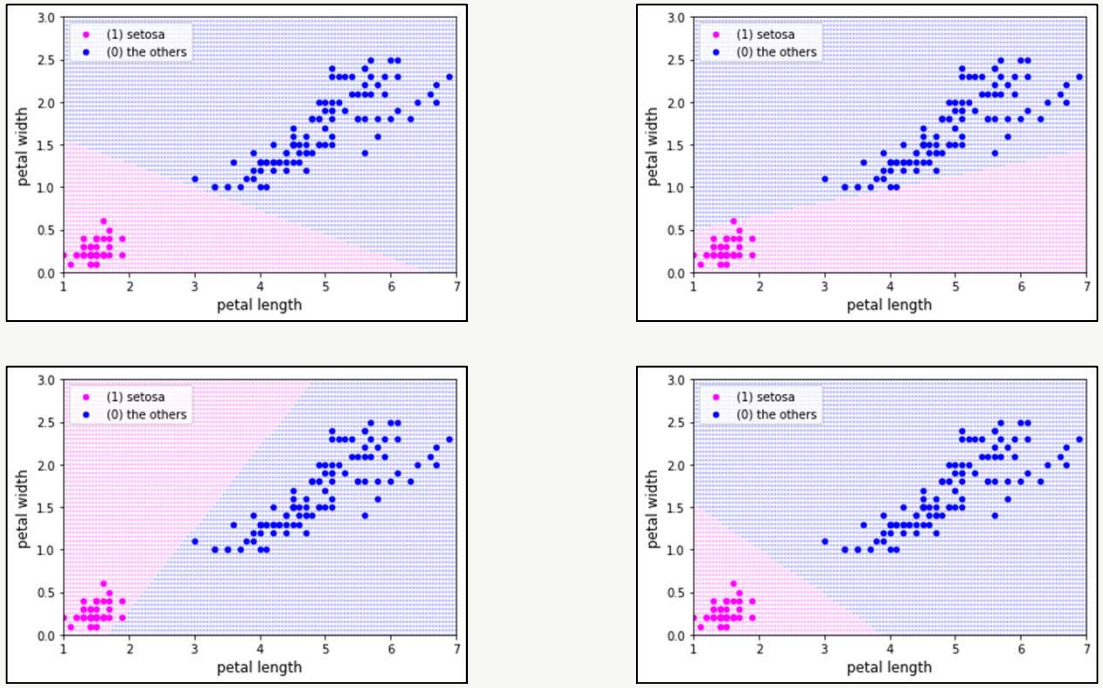
3. 실습 - 퍼셉트론을 이용한 붓꽃 식별

- ‘versicolor’를 식별하도록 학습된 퍼셉트론의 결정경계



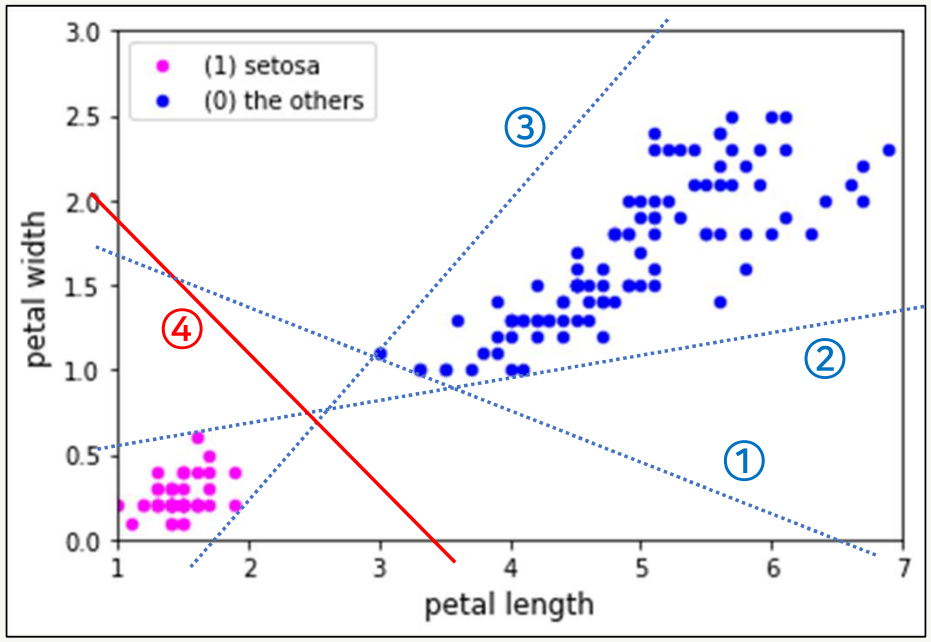
4. 퍼셉트론의 일반화 오류 문제

- 단위 계단 함수는 0 또는 1의 출력만 냄
  - 학습 표본에 대한 분류가 맞다면 오차 δ가 0이므로 파라미터의 변화가 없음



4. 퍼셉트론의 일반화 오류 문제

- 단위 계단 함수는 0 또는 1의 출력만 냄
  - 학습 표본에 대한 분류가 맞다면 오차 δ가 0이므로 파라미터의 변화가 없음



정리하기

- 인공신경망은 인간과 동물의 두뇌를 구성하는 생물학적 신경 시스템의 원리를 바탕으로 설계된 계산 시스템이다.
- 신경망을 많은 수의 층으로 구성하면 더욱 높은 차원의 표현을 할 수 있다.
- 딥러닝은 심층 신경망을 학습하기 위해 활용되는 기계학습 알고리즘이다.
- 기본적인 뉴런의 구조는 다수의 입력이 각각 가중치를 적용하여 전달되면 이를 합산한 후 활성함수를 거쳐 출력을 만들어 내는 형태이다.

정리하기

- 일반적인 활성함수는 비선형 특성을 갖는 함수이며, 입력이 0보다 작으면 출력을 억제하고, 0보다 크면 출력을 내는 역할을 한다.
- 피드포워드 신경망은 입력층에서 출력층 방향으로 뉴런 층의 연결이 이루어지는 구조이다.
- 단층 퍼셉트론은 특징 공간을 선형 결정경계로 분할할 수 있다.

다음시간안내

02

다층 퍼셉트론과 역전파