비타민 7주차 세션

비타민 10기 5조
최대상, 심정훈, 조은정, 조예진

세션 진행 순서

1. **복습**
   - 6주차 복습과제 리뷰
   - 로지스틱 회귀
   - 퍼셉트론
2. **진도**
   - 손실함수
   - 경사하강법
3. **실습**

6주차 복습과제 리뷰

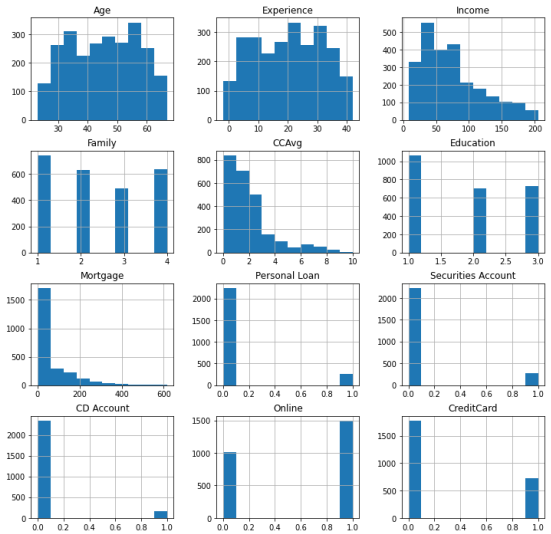
6주차 과제 리뷰

Part 1. 이진분류 – Personal Loan

[Q1-1] loan의 각 변수별 분포를 확인

반복문 사용 외에도 아래처럼 한 줄로 간단하게 작성할 수도 있음

```
loan.hist(figsize=(12,12))
```



[Q1-2] 연속형 변수의 Outlier 제거

```
KeyError: '[1130] not found in axis'
```

Drop할 때 x="Income"에서의 index와 x="CCAvg"에서의 index가 겹치면서 위의 에러 발생.

Solution 1

```
for x in continuous:
    loan2 = loan2.drop(outliers(loan2[x]), axis=0)
    loan2 = loan2.reset_index(drop=True)
```

Solution 2

```
outlier=[] #이 리스트에 outlier 인덱스를 모두 담아줄 겁니다.
for x in continuous:
  outlier=outlier+list(outliers(loan2[x]))

outlier=list(set(outlier)) # set(outlier)를 통해 중복되는 index를 처리해줍니다.
loan2.drop(outlier,inplace=True)
loan2.info()
```

6주차 과제 리뷰

[Q3-1] 분류결과 확인

```
z_vals=logr.decision_function(X_train)
probs=logr.predict_proba(X_train)
pred=logr. predict(X_train)
```

$$z_{vals} = \hat{\alpha} + \hat{\beta} X_{train}$$

$$probs = sigmoid(z_{vals}) = \frac{1}{1+e^{(-z)}}$$

Prediction은 probs가 0.5보다 크면 1(양성 클래스)

작으면 0(음성 클래스)로 저장

threshold=0.5

[Q4-4] Classify with Threshold

logr.predict(X_train)의 threshold가 0.5로 고정되어 있는 것을 사용자 지정 threshold를 사용할 수 있게 해주는 함수

```
def classify_with_threshold(p,threshold=0.5):
  p2=p.copy() # 확률 p를 카피해줍니다.
  # 여기서 매개변수 p는 양성 클래스로 분류될 확률입니다.
  # 즉, logr.predict(X_train)[:,1]
  p2[p2>threshold]=1 # threshold보다 큰 p2는 1로 대체
  p2[p2<threshold]=0 # threshold보다 작은 p2는 0으로 대체
  return p2 #반환되는 p2의 element는 0 or 1이 됩니다.
```

Cf) 리스트 컴프리헨션을 이용하여 한 줄로 작성 가능

```
def classify_with_threshold(p, threshold=0.5):
  p2 = p.copy()
  p2 = [1 if i>threshold else 0 for i in p2]

  return p2
```

6주차 과제 리뷰

Part 2. MNIST 다중분류

[Q4] True '0'와 False 'not0' 를 구분하는 plot

```
num0_index=np.where(y_test[:300]=='0')[0]
          # target이 '0'인 인덱스를 저장합니다.
plt.figure(figsize=(15,17))
for i,j in enumerate(num0_index):
    tmp=X_test[j].reshape(28,28)
    plt.subplot(5,5,i+1)
    if y_pred[j]!='0':
        # 예측값이 '0'이 아닌경우, 즉 False not 0인경우
        plt.imshow(tmp,cmap=plt.cm.binary)
        # 흑백으로 출력합니다.
    else:
        # 예측값이 '0'인 경우, 즉 True 0 인 경우
        plt.imshow(tmp)
        # default 색상으로 출력합니다.
    plt.title(f'prediction : {y_pred[j]}')
```

plt.imshow( ) : matplotlib에서 이미지를 표시하는 메소드

행렬을 만들어서 각 칸을 특정 색으로 채움

cmap : 이미지의 색상을 조정하는 옵션

default='viridis' 최솟값 보라, 최댓값 노랑을 표시

cmap=plt.cm.binary 로 변경시 흑백으로 이미지 출력

6주차 복습 – 로지스틱 회귀

로지스틱 회귀 Logistic Regression

로지스틱 회귀

회귀를 통해 데이터가 어떤 범주로 분류될지 확률값을 예측하고,
산출된 확률값을 근거로 분류하는 지도학습 알고리즘

이진 분류

- 타깃 클래스가 2개
- 시그모이드 함수로 z 값을 확률로 변환
- 출력을 0~1 사이의 값으로

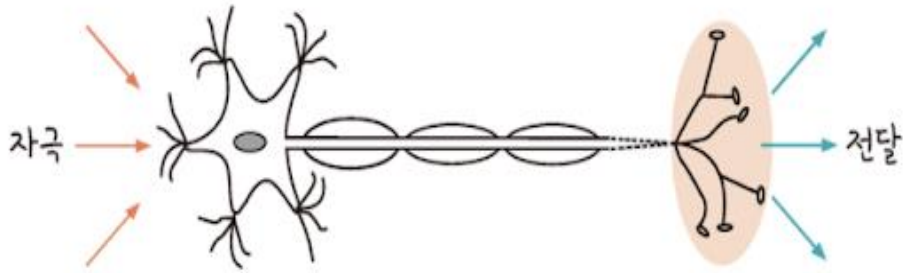
다중 분류

- 타깃 클래스가 3개 이상
- 소프트맥스 함수로 z 값을 확률로 변환
- 출력 결과의 합이 1이 되도록

퍼셉트론

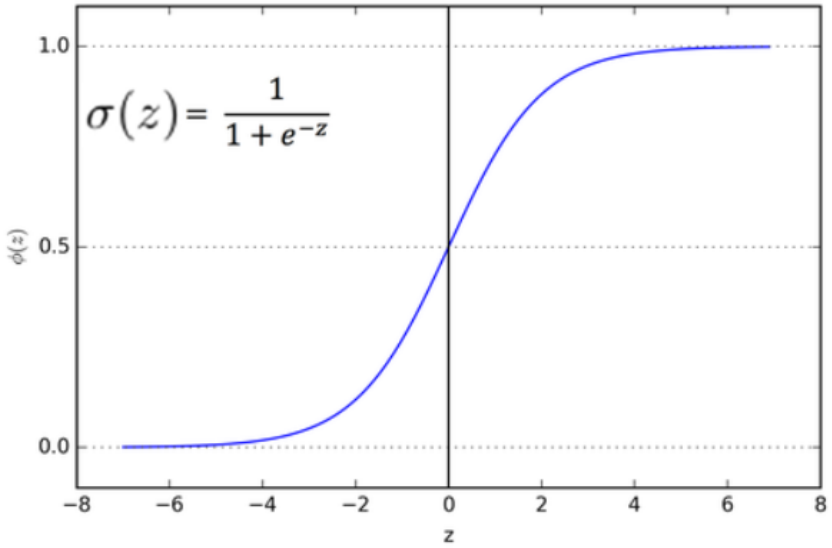
퍼셉트론 perceptron

뉴런의 신호 전달



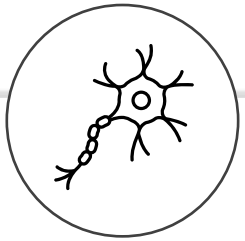
시냅스에서 나온 화학 물질에 의한 전위 변화가
임계값을 넘으면 다음 뉴런으로 신호 전달
임계값에 미치지 못하면 전달 X

로지스틱 회귀



확률값을 임계값을 기준으로 분류

퍼셉트론 perceptron



수많은 뉴런의
조합의 결과가
'생각'

뉴런과 비슷한
매커니즘을 사용하여
인공적으로 '생각' 하는
"인공 신경망" 연구 시작

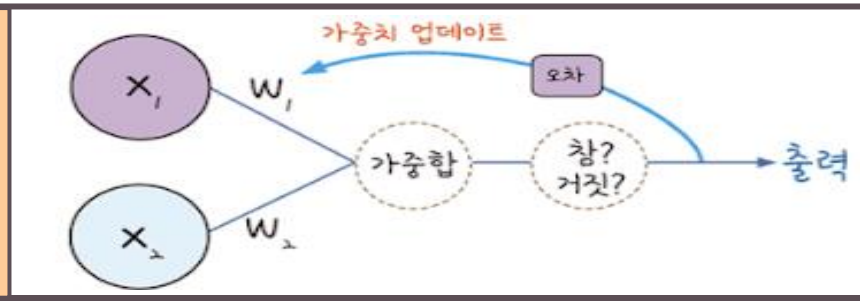
1943년
맥컬럭-월터 피츠
'켜고 끄는 기능이 있는
신경'을 그물망 형태로
연결하면 사람 뇌처럼
동작할 수 있다고 주장

1957년
미국의 신경 생물학자
프랑크 로젠블랫
앞선 개념을
실제 장치로 개발

퍼셉트론
perceptron

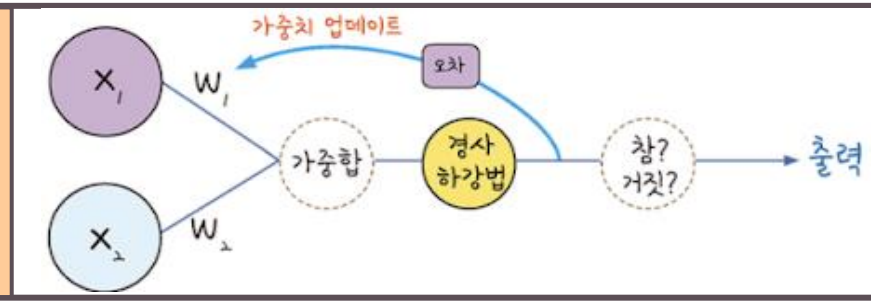
퍼셉트론 perceptron

퍼셉트론



입력 값에 가중치를 조절할 수 있게 만들어 최초의 '학습'을 하게 함

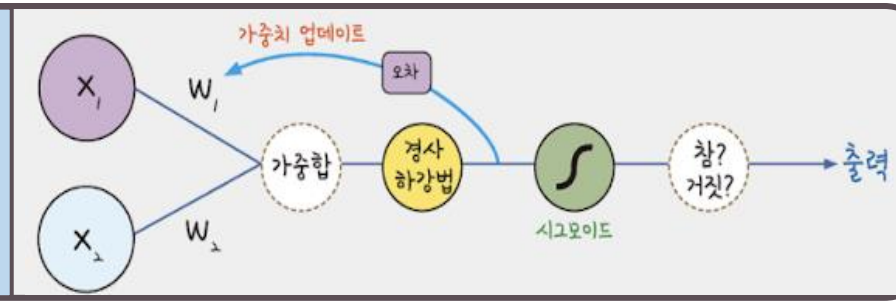
아달라인



퍼셉트론에 경사하강법을 도입해 최적의 경계선을 그릴 수 있게 한 아달라인 개발

※ 아달라인은 이후 서포트벡터머신 등 머신 러닝의 중요한 알고리즘으로 발전

로지스틱 회귀



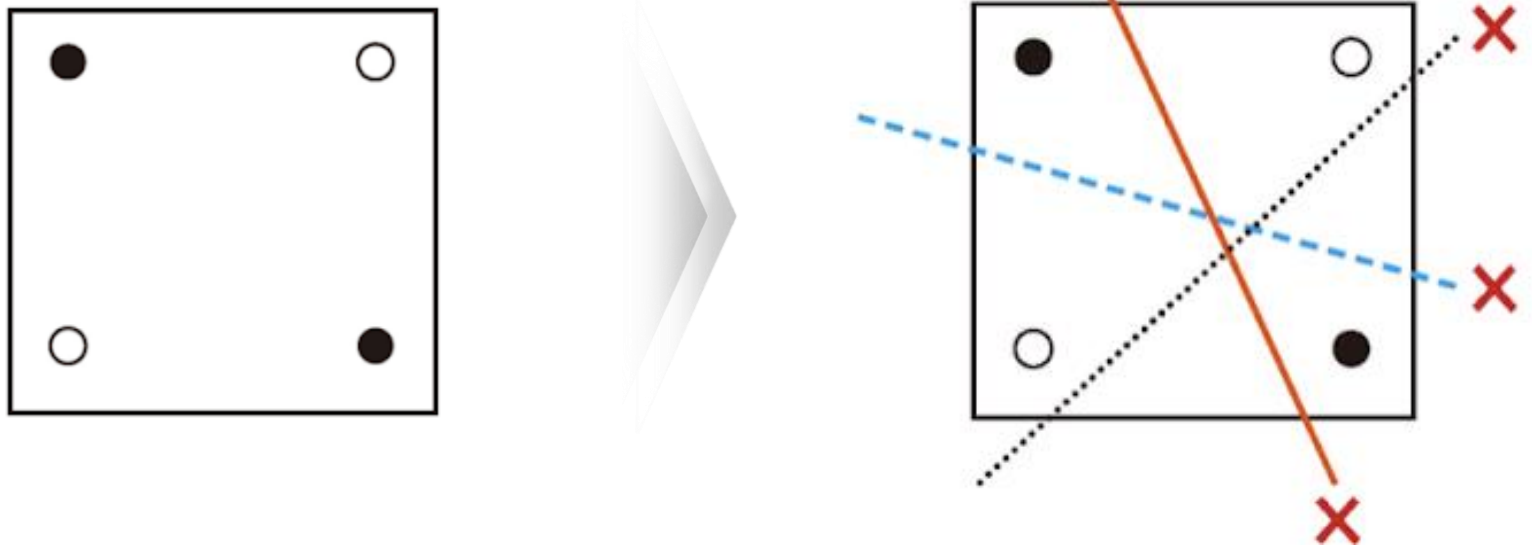
활성화 함수로 시그모이드를 사용한 것이 로지스틱 회귀

퍼셉트론 perceptron

퍼셉트론 - 한계

※ 퍼셉트론은 2차원 평면상에 직선을 긋는 것만 가능

“ 검은색 점과 흰색 점을 나눌 하나의 직선을 긋는다면? ”



퍼셉트론 perceptron

퍼셉트론 - XOR

※ XOR(exclusive OR) : 배타적 논리합

AND
(논리곱)
두 개 모두 1일 때 1

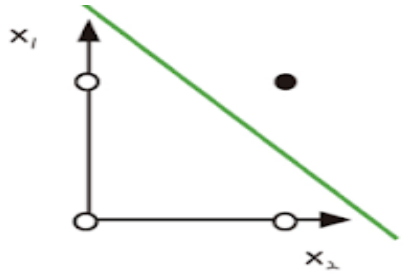
x_1	x_2	결괏값
0	0	0
0	1	0
1	0	0
1	1	1

OR
(논리합)
두 개 중 한 개라도 1이면 1

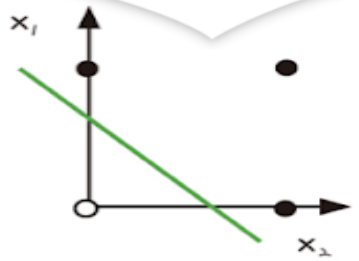
x_1	x_2	결괏값
0	0	0
0	1	1
1	0	1
1	1	1

XOR
(배타적 논리합)
하나만 1이어야 1

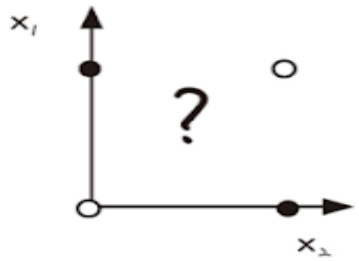
x_1	x_2	결괏값
0	0	0
0	1	1
1	0	1
1	1	0



AND



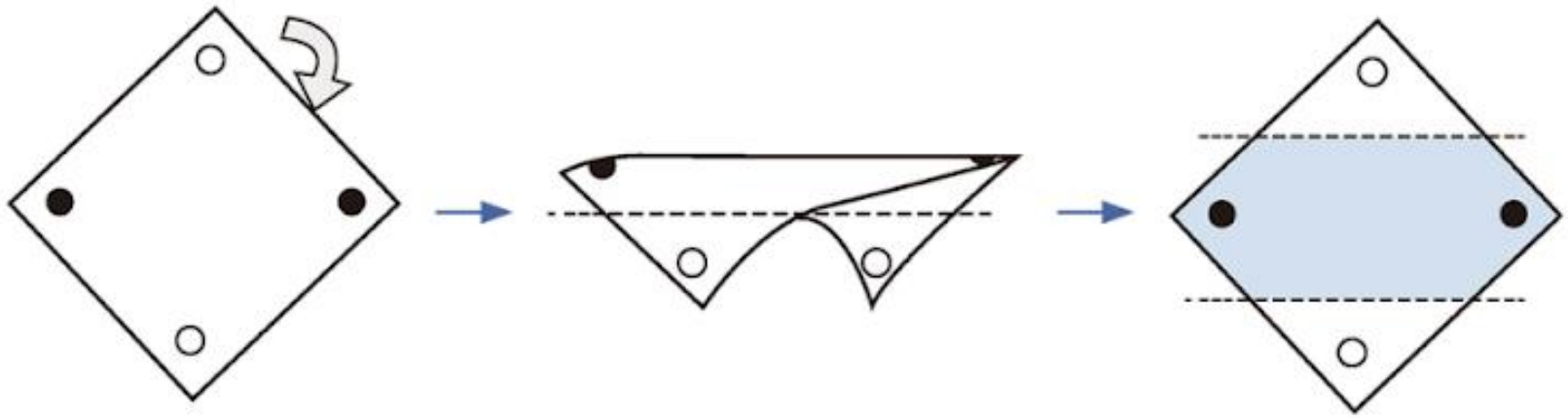
OR



XOR

퍼셉트론 perceptron

다층 퍼셉트론



'퍼셉트론 두 개를 한번에 계산'

→ 퍼셉트론 두 개를 각각 처리하는 은닉층을 만듦.

퍼셉트론 perceptron

다층 퍼셉트론

XOR
(배타적 논리합)
하나만 1이어야 1

x_1	x_2	결괏값
0	0	0
0	1	1
1	0	1
1	1	0

❷ NAND
(부정 논리곱)
하나라도 0이면 1

x_1	x_2	결괏값 1
0	0	1
0	1	1
1	0	1
1	1	0

❸ OR
(논리합)
두 개 중 한 개라도 1이면 1

x_1	x_2	결괏값 2
0	0	0
0	1	1
1	0	1
1	1	1

❹ AND
(논리곱)
두 개 모두 1일 때 1

결괏값 1	결괏값 2	결괏값
1	0	0
1	1	1
1	1	1
0	1	0

❶

은닉층이 XOR 문제를 해결하는 원리

① x1 과 x2를 두 연산으로 각각 보냄

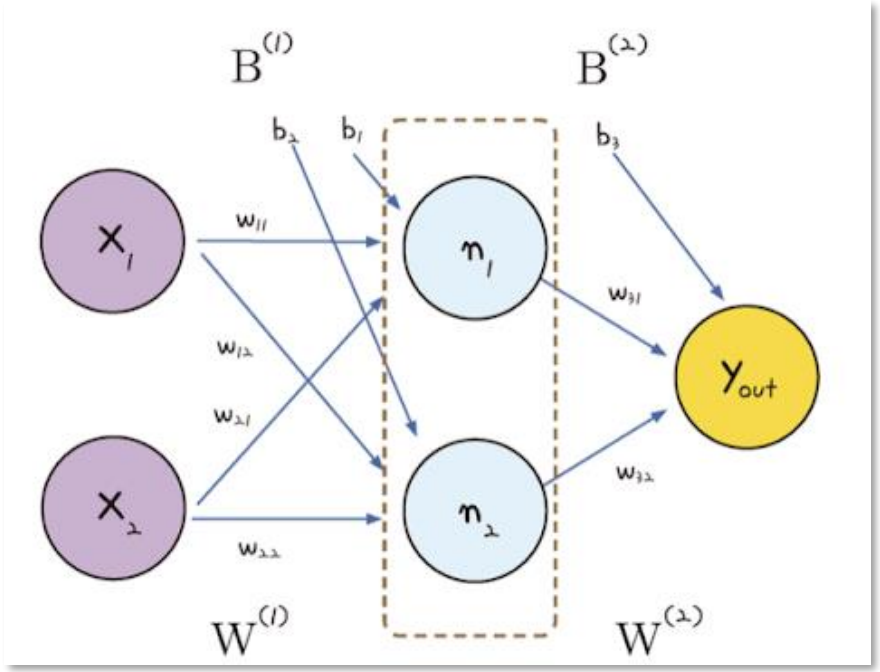
② 첫 번째 연산에는 NAND 처리

③ 동시에 두번째 연산에서 OR 처리

④ ②와 ③을 통해 구한 결과값 1과 결과값2를 가지고 AND 처리

퍼셉트론 perceptron

다층 퍼셉트론



입력층과 출력층 사이에 은닉층이 있는 구조

다층 퍼셉트론을 사용할 경우 XOR 문제는 해결

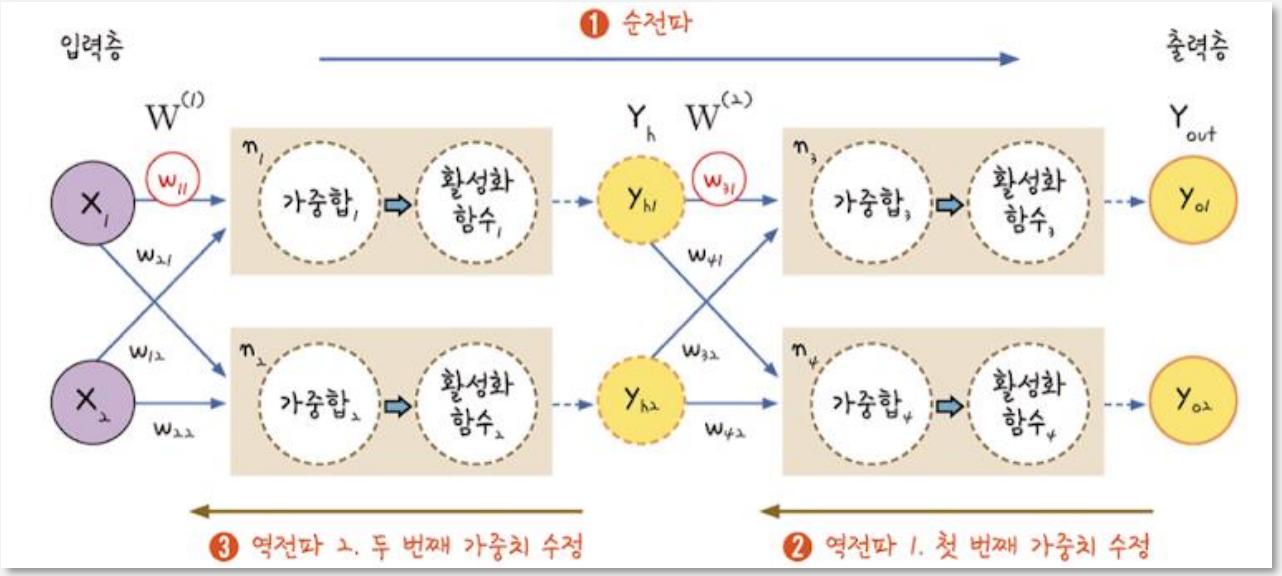
하지만, 당시에는 은닉층에 들어 있는 가중치를 데이터를 통해 학습하는 방법이 아직 X

오차 역전파

퍼셉트론 perceptron

오차 역전파

※ 목표: 출력층의 오차 최소화



① 순전파 > 가중치의 초깃값이 정해짐. 이 초깃값으로 만들어진 값과 실제 값을 비교해 출력층의 오차 계산

② 역전파1. 첫 번째 가중치 수정

③ 역전파2. 두 번째 가중치 수정

깊은 층을 통한 학습 가능

딥러닝

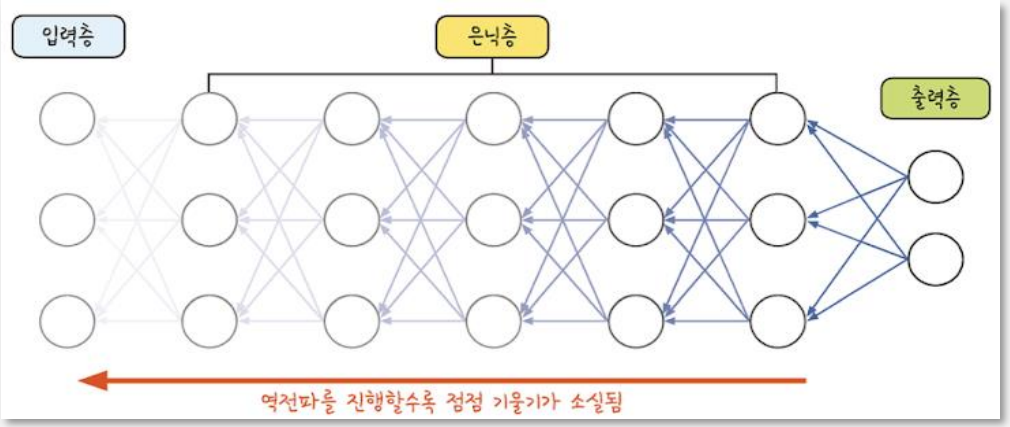
퍼셉트론 perceptron

활성화 함수

※ 활성화 함수로 시그모이드 사용시 문제점 발생

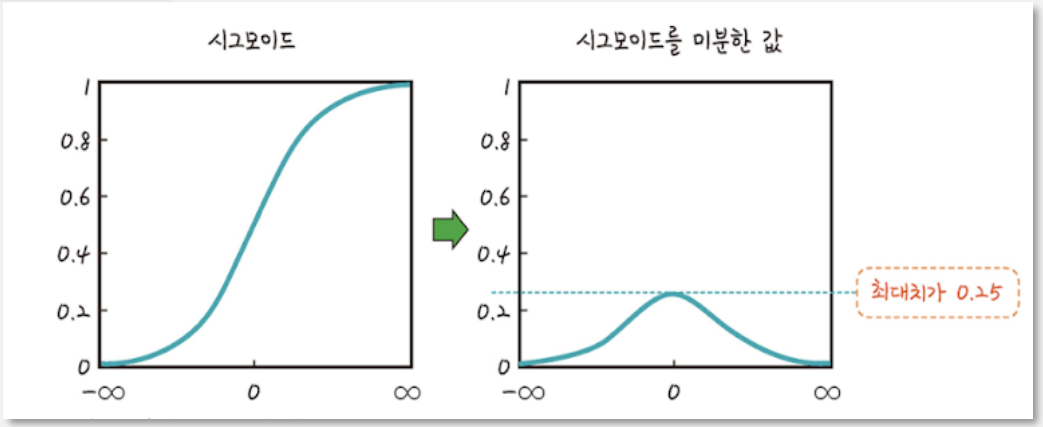
문제점

깊은 층을 만들다 보니 출력층에서 시작된 가중치 업데이트가 처음 층까지 전달되지 않는 현상이 발생



원인

활성화 함수로 사용한 시그모이드 함수의 특성

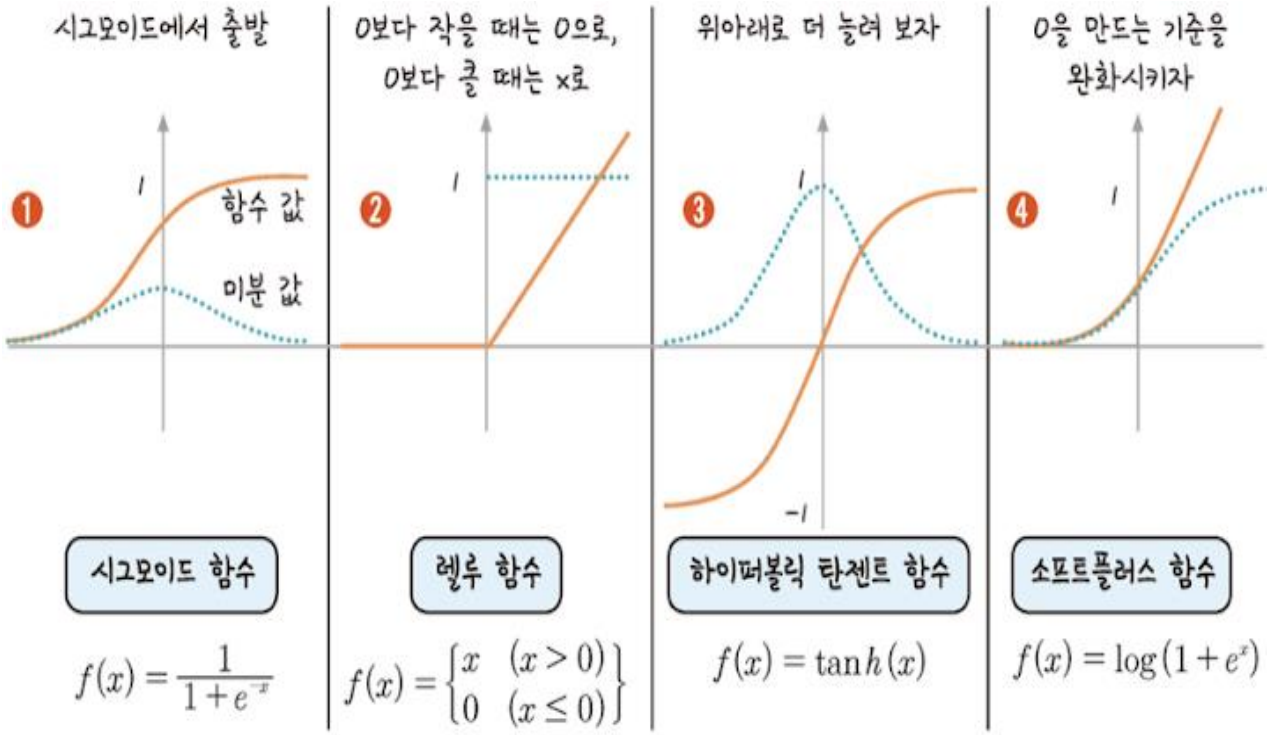


시그모이드 함수를 미분하면 최대치는 0.25
1보다 작으므로 계속 곱하다 보면 0에 가까워짐

여러 층을 거칠수록 기울기가 사라져
가중치 수정이 어려워짐

퍼셉트론 perceptron

활성화 함수



가중치 수정 문제를 해결하고자 제프리 힌튼 교수에 의해 렐루(ReLU)라는 활성화 함수가 개발됨.

②번의 렐루는 x가 0보다 작을 때 모든 값을 0으로 처리하고, 0보다 큰 값은 x를 그대로 사용.

파란색 점선(미분한 결과)을 보면, x가 0보다 크기만 하면 미분 값은 1

활성화 함수로 렐루를 쓰면 여러 번 오차 역전파가 진행되어도 맨 처음 층까지 값이 남음.

더 깊은 층을 만들 수 있게 됨

감사합니다 ☺