인공지능 : 인간의 학습, 추론, 지각, 자연언어 이해 등의 지능적 능력을 기기로 실현한 기술

학습 : 경험의 결과로 나타나는 비교적 지속적인 행동의 변화나 그 잠재력의 변화, 지식을 습득하는 과정

컴퓨터가 경험을 통해 학습 -> 프로그램이 작업을 수행하여 성능을 평가했을 때 경험을 통해 개선 = 최적의 프로그램(알고리즘)을 찾는 행위 (경험 E * 작업 T = 성능 P)

input, program -> 전통적 프로그래밍 -> result
input, desired result -> 기계 학습 -> Program

기계학습(데이터를 설명할 수 있는 학습 모델을 찾아내는 과정)

기계학습의 훈련
주어진 문제인 예측을 가장 정확하게 할 수 있는 최적의 매개변수를 찾는 작업
임의의 매개변수 값에서 시작하고 개선하여 정량적인 최적 성능에 도달

추론
새로운 특징에 대응되는 목표치의 예측에 사용
기계학습의 목표 : 훈련집합에 없는 새로운 데이터(테스트 집합)에 대한 오류 최소화
일반화 : 테스트 집합에 대한 높은 성능

기계학습 필수 요소 3가지 : 학습할 수 있는 데이터, 데이터 규칙 존재, 수학적으로 설명 불가능

차원의 저주 : 차원이 높아짐에 따라 발생하는 현실적인 문제(차원이 높아질수록 필요 데이터가 지수적으로 많아짐)

표현 학습 : 좋은 특징 공간을 자동으로 찾는 작업

초인공지능(인간을 넘어서는 특이점) : 모든 인류의 지성을 합친 것보다 더 뛰어남
강인공지능 : 인간이 할 수 있는 어떠한 지적인 업무도 성공적으로 해결
약인공지능 : 인간이 지시한 명령의 틀안에서 일함

적은 양의 데이터베이스로 높은 성능 달성하는 방법
1. 데이터 희소특성 가정
2. 매니폴드 가정 : 고차원의 데이터는 관련된 낮은 차원의 매니폴드에 가깝게 집중되어있음

평균제곱오차

$$J(\Theta) = \frac{1}{n}\sum_{i=1}^{n}(f_\Theta(\mathbf{x}_i) - y_i)^2$$

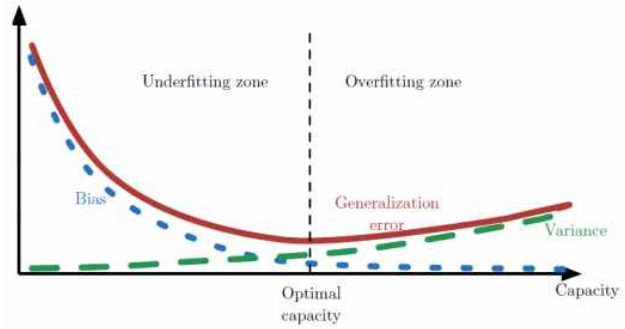
- 훈련집합

$\mathbb{X} = \{x_1 = (2.0), x_2 = (4.0), x_3 = (6.0), x_4 = (8.0)\}$,
$\mathbb{Y} = \{y_1 = 3.0, y_2 = 4.0, y_3 = 5.0, y_4 = 6.0\}$

- 초기 직선의 매개변수 $\Theta_1 = (0.1, 4.0)^T$라 가정

$x_1, y_1 \rightarrow (f_{\Theta_1}(2.0) - 3.0)^2 = ((0.1 * 2.0 + 4.0) - 3.0)^2 = 1.44$
$x_2, y_2 \rightarrow (f_{\Theta_1}(4.0) - 4.0)^2 = ((0.1 * 4.0 + 4.0) - 4.0)^2 = 0.16$
$x_3, y_3 \rightarrow (f_{\Theta_1}(6.0) - 5.0)^2 = ((0.1 * 6.0 + 4.0) - 5.0)^2 = 0.16$ $\longrightarrow$ $J(\Theta_1) = 0.8$
$x_4, y_4 \rightarrow (f_{\Theta_1}(8.0) - 6.0)^2 = ((0.1 * 8.0 + 4.0) - 6.0)^2 = 1.44$

과소적합(underfitting) : 모델의 용량이 작아 오차가 클 수밖에 없는 현상
과잉적합(overfitting) : 모델의 용량이 커 잡음까지 수용



용량증가 -> 편향 감소, 분산 증가
일반화 오차 성능(편향 + 분산)은 U형의 곡선을 가짐

훈련집합과 테스트집합과 별도로
검증집합(데이터 양 많음)모델 사용
vs
비용문제로 별도의 검증집합이 없는 상황 교차검증 모델 사용
(훈련집합을 등분하여 학습과 평가과정을 여러번 반복한 후 평균 사용)
vs
부트스트랩 모델(임의의 복원 추출 샘플링 반복(데이터 분포가 불균형일 때 사용)

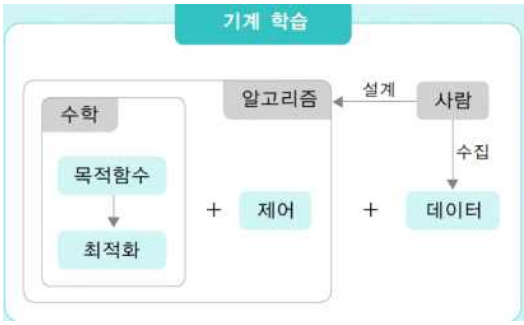
현대 기계학습 전략 : 용량이 충분히 큰 모델을 선택 후 모델이 정상을 벗어나지 않도록 여러 규제 기법을 적용
1. 데이터 확대 : 데이터를 더 많이 수집하면 일반화 능력이 향상, 하지만 데이터 수집은 많은 비용이들어 훈련집합에 있는 샘플을 변형함(회전, 왜곡)

2. 가중치 감쇠 : 개선된 목적함수를 이용하여 가중치를 작게 조절하는 규제 기법

지도 학습 : 특정 벡터 X와 목표치 Y(정답)가 모두 주어진 상황(회귀와 분류 구분)
비지도 학습 : 특징 벡터 X는 있지만 목표치 Y가 주어지지 않는 상황
군집화 과업(고객 성향에 따른 홍보 등)
밀도 추정, 특징공간 변환 과업
강화 학습 : (상대적)목표치가 주어지지만 지도학습과 다른 형태(보상이 주어짐)
준지도 학습 : 일부는 X와 Y를 모두 가지지만, 나머지는 X만 가진 상황, X의 수집은 쉽지만 Y는 수작업이 필요하여 최근 중요성 부각

오프라인 학습 vs 온라인 학습(iot 등에서 추가로 발생하는 데이터 샘플로 점증적 학습 수행)
결정론적 학습(같은 데이터를 가지고 다시 학습하면 같은 예측 모델 만듬) vs 확률적 학습 (학습 과정에서 확률 분포를 사용하니 다른 예측 모델이 만들어짐 ex. RBM, DBN)
분별 모델(예측에만 관심) vs 생성 모델(새로운 샘플을 생성할 수 있음ex. GAN, RBM)

수학은 목적함수를 정의하고 목적함수의 최저점을 찾아주는 최적화 이론 제공
최적화 이론에 학습률, 멈춤조건과 같은 제어를 추가하여 알고리즘 구축
사람은 알고리즘을 설계하고 데이터를 수집

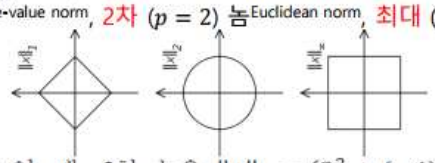


0차 = 수
1차 = 벡터
2차 = 행렬(훈련집합을 담은 행렬 = 설계행렬)
텐서 : 3차원 이상의 구조를 가진 숫자 배열

- 벡터의 p차 놈

p차 놈: $\|\mathbf{x}\|_p = \left(\sum_{i=1,d} |x_i|^p\right)^{\frac{1}{p}}$ 최대 놈: $\|\mathbf{x}\|_\infty = \max(|x_1|, |x_2|, \cdots, |x_d|)$

- 1차 ($p = 1$) 놈absolute-value norm, 2차 ($p = 2$) 놈Euclidean norm, 최대 ($p = infinite$) 놈max norm



- 예) $\mathbf{x} = (3 \ -4 \ 1)$ 일 때, 2차 놈은 $\|\mathbf{x}\|_2 = (3^2 + (-4)^2 + 1^2)^{1/2} = 5.099$

- 행렬의 프로베니우스 놈Frobenius norm: 행렬의 크기를 측정

프로베니우스 놈: $\|\mathbf{A}\|_F = \left(\sum_{i=1,n}\sum_{j=1,m} a_{ij}^2\right)^{\frac{1}{2}}$

- 예) $\left\|\begin{pmatrix} 2 & 1 \\ 6 & 4 \end{pmatrix}\right\|_F = \sqrt{2^2 + 1^2 + 6^2 + 4^2} = 7.550$

■ 학습의 정의

- 추론inferring: 식 (2.10)은 학습을 마친 알고리즘을 현장의 새로운 데이터에 적용하는 작업

분류라는 과업: $\hat{\mathbf{o}} \overset{?}{=} \boldsymbol{\tau}(\hat{\mathbf{W}} \overset{\text{앎}}{\hat{\mathbf{x}}})$ (2.10)

- 훈련training은 훈련집합의 샘플에 대해 식 (2.11)을 가장 잘 만족하는 $\mathbf{W}$를 찾아내는 작업

학습이라는 과업: $\overset{\text{앎}}{\hat{\mathbf{o}}} = \boldsymbol{\tau}(\overset{?}{\hat{\mathbf{W}}} \overset{\text{앎}}{\hat{\mathbf{x}}})$ (2.11)

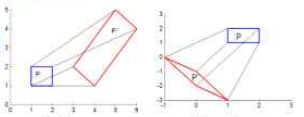
■ 현대 기계 학습에서 심층학습은 퍼셉트론을 여러 층으로 확장하여 만듦

■ 행렬 $\mathbf{A}$의 행렬식determinant $det(\mathbf{A})$

$$det\begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$
$$det\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = aei + bfg + cdh - ceg - bdi - afh \quad (2.15)$$

예를 들어 $\begin{pmatrix} 2 & 1 \\ 6 & 4 \end{pmatrix}$의 행렬식은 2*4−1*6=2

- 역행렬의 존재 유무
  - det(A) = 0 역행렬 없음
  - det(A) ≠ 0 역행렬 존재
- 기하학적 의미: 행렬식은 주어진 행렬의 곱에 의한 공간의 확장 또는 축소 해석
  - 만약 det(A) = 0, 하나의 차원을 따라 축소되어 부피를 잃게 됨
  - 만약 det(A) = 1, 부피 유지한 변환/방향 보존 됨
  - 만약 det(A) = -1, 부피 유지한 변환/방향 보존 안됨
  - 만약 det(A) = 5, 5배 부피 확장되며 방향 보존



$A = \begin{pmatrix} 1 & 2 \\ -1 & 3 \end{pmatrix}$, $det(A)=5$ $A = \begin{pmatrix} 1 & -1 \\ -2 & 1 \end{pmatrix}$, $det(A)=-1$

(변환된 P' 공간 부피) = $|det(\mathbf{A})|$·(원래 P 공간 부피)

베이즈 정리 (식 (2.26))

- 베이즈 정리를 적용하면, $\hat{y} = \underset{y}{\operatorname{argmax}} P(y|x = \text{하양}) = \underset{y}{\operatorname{argmax}} \frac{P(x = \text{하양}|y)P(y)}{P(x = \text{하양})}$
- 세 가지 경우에 대해 확률을 계산하면,

$$P(①|\text{하양}) = \frac{P(\text{하양}|①)P(①)}{P(\text{하양})} = \frac{\frac{9}{12}\frac{1}{8}}{\frac{43}{96}} = \frac{9}{43}$$

$$P(②|\text{하양}) = \frac{P(\text{하양}|②)P(②)}{P(\text{하양})} = \frac{\frac{5}{15}\frac{4}{8}}{\frac{43}{96}} = \frac{16}{43}$$

$$P(③|\text{하양}) = \frac{P(\text{하양}|③)P(③)}{P(\text{하양})} = \frac{\frac{3}{6}\frac{3}{8}}{\frac{43}{96}} = \frac{18}{43} \longrightarrow ③\text{번 병일 확률이 가장 높음}$$

베이즈 정리의 해석

- 사후posteriori확률=우도likelihood확률*사전prior확률 → $\overbrace{P(y|x)}^{\text{사후확률}} = \frac{\overbrace{P(x|y)}^{\text{우도}}\overbrace{P(y)}^{\text{사전확률}}}{P(x)}$

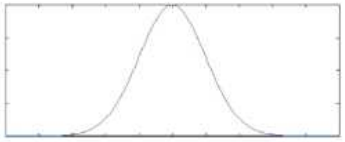
사후확률을 직접 추정하는 일은 아주 단순한 경우를 제외하고 불가능
따라서 베이즈 정리를 이용해 추정한다.

최대우도 : 어떤 확률변수의 관찰된 값들을 토대로 그 확률변수의 매개변수를 구하는 방법

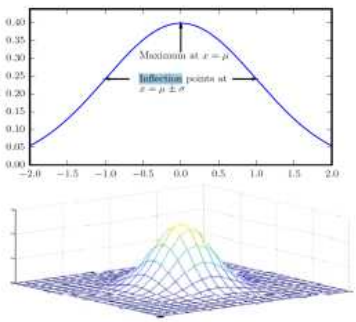
가우시안 분포Gaussian distribution

- 평균 μ와 분산 σ^2으로 정의

$$N(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$



(a) 1차원



(b) 2차원

그림 2-19 가우시안 분포

- 다차원 가우시안 분포: 평균벡터 $\boldsymbol{\mu}$와 공분산행렬 $\boldsymbol{\Sigma}$로 정의

$$N(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{|\boldsymbol{\Sigma}|}\sqrt{(2\pi)^d}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

베르누이 분포Bernoulli distribution

- 성공(x=1) 확률 p이고 실패(x=0) 확률이 1-p인 분포

$$Ber(x;p) = p^x(1-p)^{1-x} = \begin{cases} p, & x = 1\text{일 때} \\ 1-p, & x = 0\text{일 때} \end{cases}$$

이항 분포Binomial distribution

- 성공 확률이 p인 베르누이 실험을 m번 수행할 때 성공할 횟수의 확률분포

$$B(x;m,p) = C_m^x p^x(1-p)^{m-x} = \frac{m!}{x!\,(m-x)!}p^x(1-p)^{m-x}$$

- 확률질량함수

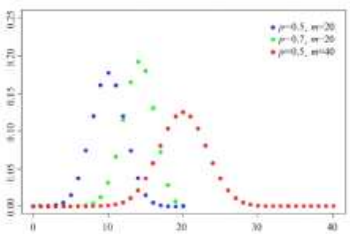
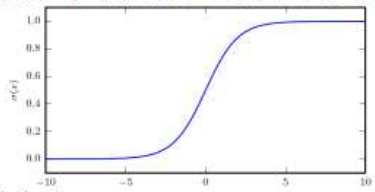


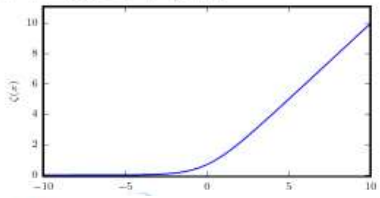
그림 2-20 이항 분포

확률 분포와 연관된 유용한 함수들

- 로지스틱 시그모이드 함수logistic sigmoid function
  - 일반적으로 베르누이 분포의 매개변수를 조정을 통해 얻어짐



- 소프트플러스 함수softplus function
  - 정규 분포의 매개변수의 조정을 통해 얻어짐



지수 분포exponential distribution

$$p(x;\lambda) = \lambda \mathbf{1}_{x\geq 0} \exp(-\lambda x)$$

라플라스 분포Laplace distribution

$$\text{Laplace}(x;\mu,\gamma) = \frac{1}{2\gamma}\exp\left(-\frac{|x-\mu|}{\gamma}\right)$$

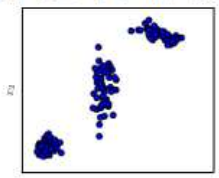
디랙 분포Dirac distribution

$$p(x) = \delta(x-\mu)$$

혼합 분포들Mixture Distributions

$$P(\mathrm{x}) = \sum_i P(\mathrm{c}=i)P(\mathrm{x}\mid \mathrm{c}=i)$$

- 3개의 요소를 가진 가우시안 혼합 분포 예 ← 가우시안 혼합 모델 추청 가능



변수 변환change of variables

- 기존 확률변수를 새로운 확률 변수로 바꾸는 것
- 변환 $y=g(x)$와 가역성을 가진 g에 의해 정의되는 x, y 두 확률변수를 가정할 때, 두 확률 변수는 다음과 같이 상호 정의될 수 있음

$$p_x(\boldsymbol{x}) = p_y(g(\boldsymbol{x}))\left|\det\left(\frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)\right|$$

- 예) 확률변수 x의 확률질량함수가 다음과 같을 때,

$$\left(\frac{4}{5}\right)\left(\frac{1}{5}\right)^{x-1},\ x=1,2,\ \ldots$$

새로운 확률변수$y=x^2$의 확률질량함수는 다음과 같이 정의됨

$y=x^2 \ \Rightarrow\ x=\sqrt{y}$

$$f(x)=f(\sqrt{y})=\left(\frac{4}{5}\right)\left(\frac{1}{5}\right)^{\sqrt{y}-1}=g(y) \longrightarrow g(y)=\begin{cases}\left(\frac{4}{5}\right)\left(\frac{1}{5}\right)^{\sqrt{y}-1}, & y=1,4,9,\ldots \\ 0, & \text{elsewhere}\end{cases}$$

정보이론과 확률통계는 많은 교차점을 가짐
확률통계 : 기계학습의 기초적인 근간 제공(해당 확률 분포 추정, 확률 분포간의 유사성 정량화)
정보이론 관점에서도 기계학습 접근 가능(불확실성을 정량화 하여 정보이론 방법을 기계학습
ex. 엔트로피 교차 엔트로피 KL발산, 상대 엔트로피)

정보이론 : 사건이 지닌 정보를 정량화 할 수 있나?
(확률이 작을수록 많은 정보 - 잘 일어나지 않는 사건의 정보량이 많다.)

자기정보 : 사건의 정보량
엔트로피 : 확률변수 x의 불확실성을 나타내는 엔트로피, 모든 사건 정보량의 기대 값으로 표현

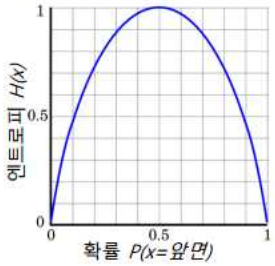
이산 확률분포 $H(x) = -\sum_{i=1,k} P(e_i)\log_2 P(e_i)$ 또는 $H(x) = -\sum_{i=1,k} P(e_i)\log_e P(e_i)$ (2.45)

연속 확률분포 $H(x) = -\int_{\mathbb{R}} P(x)\log_2 P(x)$ 또는 $H(x) = -\int_{\mathbb{R}} P(x)\log_e P(x)$ (2.46)

- 예) 동전의 앞뒤의 발생 확률이 동일한 경우의 엔트로피는 다음과 같음

$$\begin{aligned} H(x) &= -\sum_x P(x)\log P(x) \\ &= -(0.5 \times \log_2 0.5 + 0.5 \times \log_2 0.5) \\ &= -\log_2 0.5 \\ &= -(-1) \end{aligned}$$

- 동전의 발생 확률에 따른 엔트로피 변화
  - 공평한 동전을 사용할 때에 가장 큰 엔트로피를 구할 수 있음
  - 동전 던지기 결과 전송에는 최대 1비트가 필요함을 의미



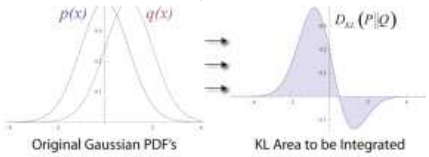
→ 모든 사건이 동일한 확률을 가질 때
즉, 불확실성이 가장 높은 경우, 엔트로피가 최고임

국민대학교 KOOKMIN UNIVERSITY

주사위가 윷놀이보다 엔트로피가 높은 이유 - 주사위는 모든 사건이 동일한 확률이라 윷놀이보다 예측이 어려움 -> 무질서하고 불확실성이 큼 -> 엔트로피가 높음

교차 엔트로피(두 확률분포의 교차 엔트로피, 심층학습의 손실함수로 많이 사용)

■ KL 발산
- 식 (2.48)은 P와 Q 사이의 KL 발산
- 두 확률분포 사이의 거리를 계산할 때 주로 사용



$$KL(P \parallel Q) = \sum_x P(x)\log_2 \frac{P(x)}{Q(x)} \quad (2.48)$$

■ 교차 엔트로피와 KL 발산의 관계

$$\begin{aligned} P\text{와 } Q\text{의 교차 엔트로피 } H(P,Q) &= H(P) + \sum_x P(x)\log_2 \frac{P(x)}{Q(x)} \\ &= P\text{의 엔트로피} + P\text{와 } Q \text{ 간의 } KL \text{ 다이버전스} \end{aligned} \quad (2.49)$$

→ 즉, 가지고 있는 데이터 분포 $P(x)$와
추정한 데이터 분포 $Q(x)$간의 차이 최소화하는데 교차 엔트로피 사용

[그림 2-21]과 같이 정상적인 주사위와 찌그러진 주사위가 있는데, 정상적인 주사위의 확률분포는 P, 찌그러진 주사위의 확률분포는 Q를 따르며, P와 Q가 다음과 같이 분포한다고 가정하자.

$$P(1)=\frac{1}{6},\ P(2)=\frac{1}{6},\ P(3)=\frac{1}{6},\ P(4)=\frac{1}{6},\ P(5)=\frac{1}{6},\ P(6)=\frac{1}{6}$$
$$Q(1)=\frac{3}{12},\ Q(2)=\frac{1}{12},\ Q(3)=\frac{1}{12},\ Q(4)=\frac{1}{12},\ Q(5)=\frac{3}{12},\ Q(6)=\frac{3}{12}$$

(a) 정상 주사위 (b) 찌그러진 주사위

그림 2-21 확률분포가 다른 두 주사위

확률분포 P와 Q 사이의 교차 엔트로피와 KL 다이버전스는 다음과 같다.

$$H(P,Q)=-\left(\frac{1}{6}\log_2\frac{3}{12}+\frac{1}{6}\log_2\frac{1}{12}+\frac{1}{6}\log_2\frac{1}{12}+\frac{1}{6}\log_2\frac{1}{12}+\frac{1}{6}\log_2\frac{3}{12}+\frac{1}{6}\log_2\frac{3}{12}\right)=2.7925$$
$$KL(P\,\|\,Q)=\frac{1}{6}\log_2\frac{2}{3}+\frac{1}{6}\log_2 2+\frac{1}{6}\log_2 2+\frac{1}{6}\log_2 2+\frac{1}{6}\log_2\frac{2}{3}+\frac{1}{6}\log_2\frac{2}{3}=0.2075$$

[예제 2-8]에서 P의 엔트로피 $H(P)$는 2.585이었다. 따라서 식 (2.49)가 성립함을 알 수 있다.

최적화 : 기계 학습의 최적화는 단지 훈련집합이 주어지고, 훈련집합에 따라 정해지는 목적함수의 최저점으로 만드는 모델의 매개변수를 찾아야 함

최적화는 예측 단계가 아닌 학습 단계에 필요

- 식 (2.58)은 경사 하강법gradient descent이 낮은 곳을 찾아가는 원리
  - $\mathbf{g}=d\boldsymbol{\Theta}=\frac{\partial J}{\partial \boldsymbol{\Theta}}$ (기울기)이고, ρ는 학습률learning rate (이동 거리 조절)

$$\boldsymbol{\Theta}=\boldsymbol{\Theta}-\rho\mathbf{g} \qquad (2.58)$$

  - 함수의 기울기 (경사)를 구하여 기울기가 낮은 쪽으로 반복적으로 이동하여 최소값에 도달
- 집단(무리)batch 경사 하강 알고리즘
  - 샘플의 경사도를 구하고 평균한 후 한꺼번에 갱신
  - 훈련집합 전체를 다 봐야 갱신이 일어나므로 학습 과정이 오래 걸리는 단점

알고리즘 2-4 배치 경사 하강 알고리즘(BGD)

입력: 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 ρ
출력: 최적해 $\hat{\boldsymbol{\Theta}}$

```
1 난수를 생성하여 초기해 Θ를 설정한다.
2 repeat
3   𝕏에 있는 샘플의 그레이디언트 ∇1, ∇2, ..., ∇n을 계산한다.
4   ∇total = (1/n) Σ(i=1,n) ∇i   // 그레이디언트 평균을 계산
5   Θ = Θ − ρ∇total
6 until(멈춤 조건)
7 Θ̂ = Θ
```

훈련집합

$$\mathbb{X}=\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n\}$$
$$\mathbb{Y}=\{y_1,y_2,\cdots,y_n\}$$

- 확률론적 경사 하강SGD (stochastic gradient descent) 알고리즘
  - 한 샘플 혹은 작은 집단(무리)mini-batch의 경사도를 계산한 후 즉시 갱신
  - 작은 무리 단위: 3~6번째 줄을 한 번 반복하는 일을 한 세대epoch라 부름 (훈련집합==세대)

알고리즘 2-5 스토케스틱 경사 하강 알고리즘(SGD)

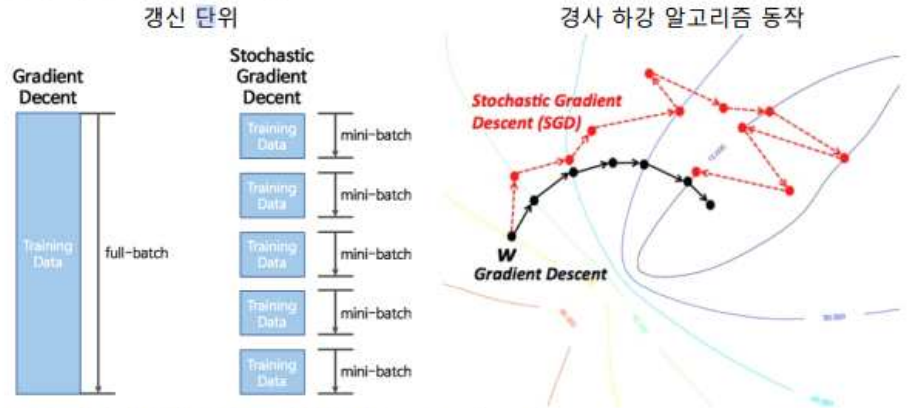
입력: 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 ρ
출력: 최적해 $\hat{\boldsymbol{\Theta}}$

```
1 난수를 생성하여 초기해 Θ를 설정한다.
2 repeat
3   𝕏의 샘플의 순서를 섞는다.
4   for (i=1 to n)
5       i번째 샘플에 대한 그레이디언트 ∇i를 계산한다.
6       Θ = Θ − ρ∇i
7 until(멈춤 조건)
8 Θ̂ = Θ
```

  - 한 샘플 단위: 다른 방식의 구현([알고리즘 2-5]의 3~6번째 줄을 다음 코드로 대치)

```
3   𝕏에서 임의로 샘플 하나를 뽑는다.
4   뽑힌 샘플의 그레이디언트 ∇를 계산한다.
5   Θ = Θ − ρ∇
```

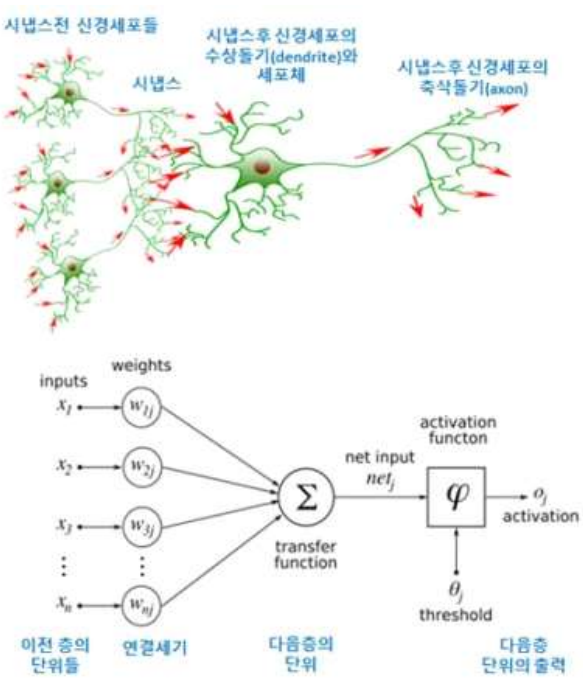
■ 경사 하강 알고리즘 비교



- 집단 경사 하강 알고리즘: 정확한 방향으로 수렴. 느림
- 확률론적 경사 하강 알고리즘: 수렴이 다소 헤맬 수 있음. 빠름

인공신경망 : 기계학습 역사에서 가장 오래된 기계 학습 모델(뉴런)
퍼셉트론(인공두뇌학) -> 다층 퍼셉트론(결합설) -> 깊은 인공신경망(심층학습)

컴퓨터가 사람 뇌의 정보처리를 모방하여 지능적 행위를 할 수 있는 인공지능에 도전
뉴런의 동작을 모방해 인공 신경망 연구 시작(페셉트론 고안)



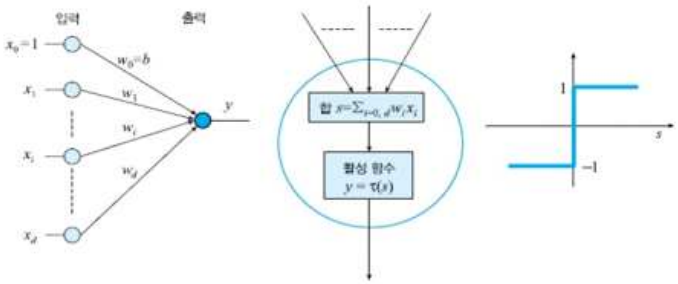
사람 신경망	인공 신경망
세포체	노드
수상돌기	입력
축삭	출력
시냅스	가중치

전방신경망 vs 순환신경망
얕은 신경망 vs 깊은 신경망
결정론 신경망 - 모델의 매개변수와 조건에 의해 출력이 완전히 결정되는 신경망
확률론적 신경망-고유의 임의성을 가지고 매개변수와 조건이 같더라도 다른 출력을 가지는 신경망

퍼셉트론(절, 가중치, 층과 같은 구조, 깊은 신경망의 토대)

■ 퍼셉트론의 구조

- 입력
  - i번째 노드는 특징 벡터 $\mathbf{x} = (x_1, x_2, \cdots, x_d)^T$의 요소 x_i를 담당
  - 항상 1이 입력되는 편향bias 노드 포함
- 입력과 출력 사이에 연산하는 구조를 가짐
  - i번째 입력 노드와 출력 노드를 연결하는 변edge는 가중치 w_i를 가짐
  - 퍼셉트론은 단일 층 구조라고 간주함
- 출력
  - 한 개의 노드에 의해 수치(+1 혹은 -1) 출력



■ 퍼셉트론의 동작

- 선형 연산 → 비선형 연산
  - [선형] 입력(특징)값과 가중치를 곱하고 모두 더해 s를 구함
  - [비선형] 활성함수 τ를 적용
    - 활성함수 τ로 계단 함수step function를 사용 → 출력 y=+1 또는 y=-1
- 수식

$$y = \tau(s)$$

$$\text{이때 } s = w_0 + \sum_{i=1}^{d} w_i x_i, \qquad \tau(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases}$$

- 경사도 계산
  - 식 (2.58)의 일반화된 가중치 갱신 규칙 $\boldsymbol{\theta} = \boldsymbol{\theta} - \rho\mathbf{g}$를 적용하려면 경사도 $\mathbf{g}$가 필요
  - 식 (3.7)을 편미분하면,

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} \frac{\partial(-y_k(w_0 x_{k0} + w_1 x_{k1} + \cdots + w_i x_{ki} + \cdots + w_d x_{kd}))}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} -y_k x_{ki}$$

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} -y_k x_{ki}, \qquad i = 0,1,\cdots,d \tag{3.8}$$

← x_{ki}는 $\mathbf{x}_k = (x_{k0}, x_{k1}, \ldots, x_{kd})^T$의 i번째 요소임

- 편미분 결과인 식 (3.8)을 식 (2.58)에 대입하면,

$$\text{델타 규칙:} \quad w_i = w_i + \rho \sum_{\mathbf{x}_k \in Y} y_k x_{ki}, \qquad i = 0,1,\cdots,d \tag{3.9}$$

→ 델타규칙delta rule은 퍼셉트론의 학습 방법

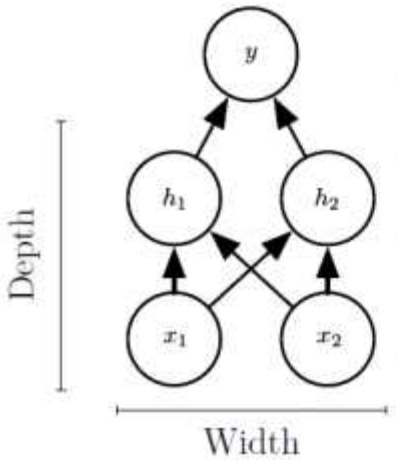
퍼셉트론 학습 알고리즘(확률론적 형태) - 샘플 순서를 섞고 틀린 샘플이 발생하면 즉시 갱신

퍼셉트론 학습 알고리즘(무리 형태) - 훈련집합의 샘플을 모두 맞출 때까지 세대를 반복

퍼셉트론 : 선형 분류기의 한계 -> 다층 퍼셉트론 이론 정립, 신경망 재부활

다층퍼셉트론(은닉층을 둔다, 유리한 새로운 특징 공간으로 변환한다,
시그모이드 활성함수 도입 - 연성에서는 출력이 연속값
오류 역전파 알고리즘 사용 - 한 층씩 그레이디언트를 계산하고 가중치 갱신)

은닉층 - 특징 추출기 : 특징 벡터를 분류에 더 유리한 새로운 특징 공간으로 변환
특징학습이라 부름



- 범용적 근사 이론universal approximation theorem
  - 하나의 은닉층은 함수의 근사를 표현

→ 다층 퍼셉트론도 공간을 변환하는 근사 함수

- 얕은 은닉층의 구조
  - 지수적으로 더 넓은 폭width이 필요할 수 있음
  - 더 과잉적합 되기 쉬움

→ 일반적으로 깊은 은닉층의 구조가 좋은 성능을 가짐

■ Hornik 주장[1989]

- 은닉층을 하나만 가진 다층 퍼셉트론은 범용근사자universal approximator

"… standard multilayer feedforward network architectures using arbitrary squashing functions can approximate virtually any function of interest to any desired degree of accuracy, provided sufficiently many hidden units are available, … 은닉 노드가 충분히 많다면, 포화함수(활성함수)로 무엇을 사용하든 표준 다층 퍼셉트론은 어떤 함수라도 원하는 정확도만큼 근사화할 수 있다."

3.7.3 성능 향상을 위한 경험의 중요성

■ 순수한 최적화 알고리즘으로는 높은 성능 불가능

- 데이터 희소성, 잡음, 미숙한 신경망 구조 등 이유
- 성능 향상을 위한 다양한 경험heuristics을 개발하고 공유함

→ 예) 「Neural Networks: Tricks of the Trade」 [2012]

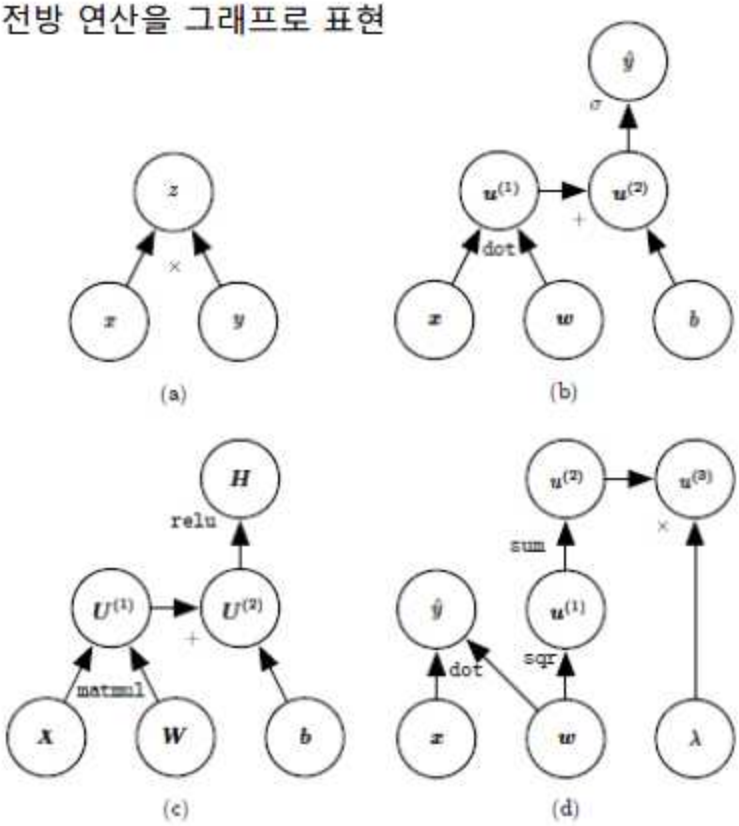
■ 신경망의 경험적 개발에서 중요 쟁점

- 아키텍처 은닉층과 은닉 노드의 개수를 정해야 한다. 은닉층과 은닉 노드를 늘리면 신경망의 용량은 커지는 대신, 추정할 매개변수가 많아지고 학습 과정에서 과잉적합할 가능성이 커진다. 1.6절에서 소개한 바와 같이 현대 기계 학습은 복잡한 모델을 사용하되, 적절한 규제 기법을 적용하는 경향이 있다.
- 초깃값 [알고리즘 3-4]의 라인 1에서 가중치를 초기화한다. 보통 난수를 생성하여 설정하는데, 값의 범위와 분포가 중요하다. 이 주제는 5.2.2절에서 다룬다.
- 학습률 처음부터 끝까지 같은 학습률을 사용하는 방식과 처음에는 큰 값으로 시작하고 점점 줄이는 적응적 방식이 있다. 5.2.4절에서 여러 가지 적응적 학습률 기법을 소개한다.
- 활성함수 초창기 다층 퍼셉트론은 주로 로지스틱 시그모이드나 tanh 함수를 사용했는데, 은닉층의 개수를 늘림에 따라 그레이디언트 소멸과 같은 몇 가지 문제가 발생한다. 따라서 깊은 신경망은 주로 ReLU 함수를 사용한다. 5.2.5절에서 여러 가지 ReLU 함수를 설명한다.

기계학습의 목표 - 모든 샘플을 옳게 분류하는 함수 f를 찾는 일

연산 그래프computational graph의 예

- 전방 연산을 그래프로 표현



(a) $z = xy$

(b) logistic regression

- $\hat{y} = \sigma\left(x^\top w + b\right)$

(c) ReLU activation

- $H = \max\{0, XW + b\}$

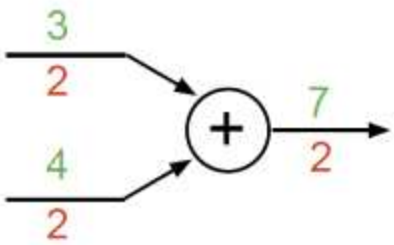
(d) linear regression

- w : weights
- $\hat{y}$: prediction
- $\lambda \sum_i w_i^2$: weight decay penalty

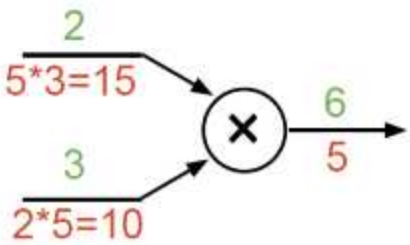
오류 역전파 알고리즘 - 출력의 오류를 역방향(왼쪽)으로 전파하며 경사도를 계산하는 알고리즘
반복되는 부분식들의 경사도의 지수적 폭발, 사라짐을 피해야 함

역전파 주요 예

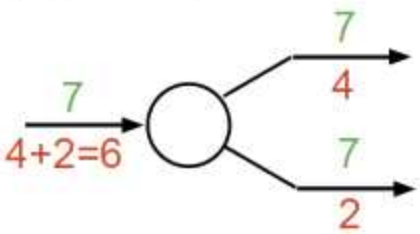
add gate: gradient distributor



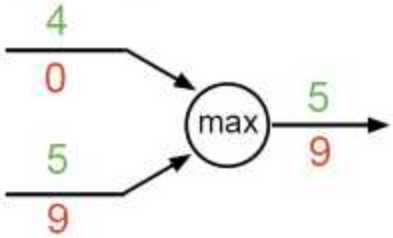
mul gate: "swap multiplier"



copy gate: gradient adder



max gate: gradient router



간단한 전방전파와 오류역전파

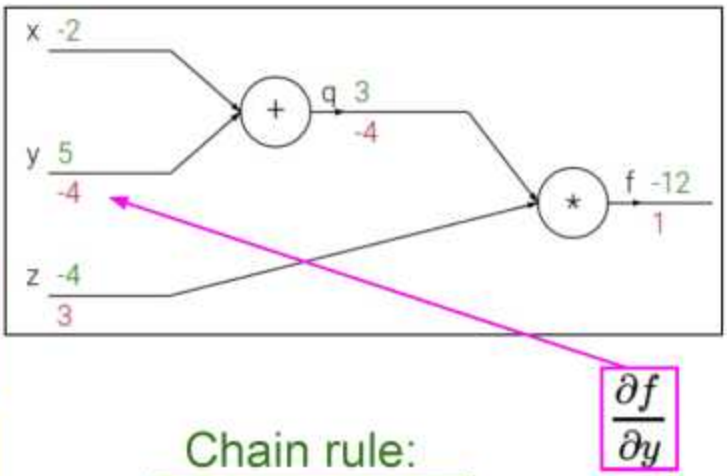
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Chain rule:

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y}$$

Upstream gradient

Local gradient

1) torch.rand()

2) torch.zeros()

3) torch.tensor

4) torch.from_numpy(a)

5) torch.as_tensor(a)

6) torch.cat()

7) torch.stack()

8) torch.eye()

9) requires_grad=True

10) torch.autograd.backward()

11) torch.autograd.grad()

12) torch.norm()

13) torch.no_grad()

14) torch.detach()

1) x.size()

2) x[:, 1]

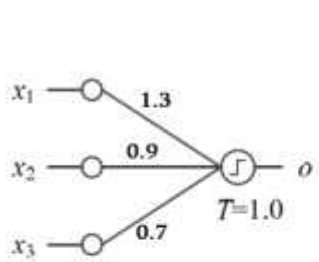
3) x.view(2, 10) | torch.reshape(2, 10)

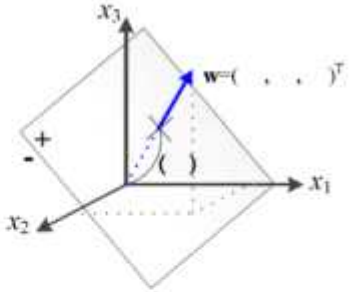
4) device = torch.device('cuda')

   x.to(device)

   x.cuda()

8. [퍼셉트론] 다음 그림의 퍼셉트론 (perceptron) 입니다. (6점, 각 3점)





(1) 해당 퍼셉트론에 의해 결정되는 결정평면의 방향과 원점에서의 거리를 구하세요.

(2) T=0.0으로 바꾸기 위해 퍼셉트론을 수정하고, 결정평면의 변화를 설명하세요.

9. [PLA] PLA 가중치 갱신 법칙 $w(t+1) = w(t) + y(t)x(t)$를 보고 다음 문제의 답을 보이세요.
(18점, 각 6점)

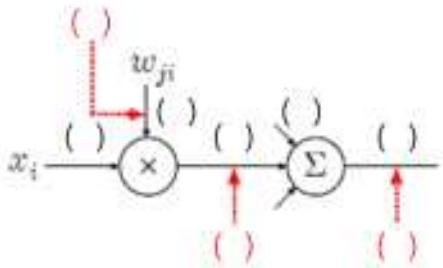
(1) $y(t)w^T(t)x(t) < 0$임을 보이세요. (Hint: $x(t)$는 $w(t)$에 의해 오분류 됨)

(2) $y(t)w^T(t+1)x(t) > y(t)w^T(t)\,x(t)$ 임을 보이세요. (Hint: $w(t+1) = w(t) + y(t)x(t)$ 이용)

(3) $w(t)$에서 $w(t+1)$로 이동하는 것이 $x(t)$를 분류하는데 올바른 방향으로 이동함을 설명하세요.

11. 아래 그림의 연산 그래프 예처럼 f(x, y, z)= (x+y)z 연산에 대한 연산 그래프를 새롭게 생성하고, x=-2, y=5, z=-4인 경우에 전방 전파와 이에 대응되는 오류 역전파를 각 가중치마다 계산하세요. (10점)

[아래 예에 표시된 것처럼 전방 전파 연산 결과는 검은색 빈칸, 오류 역전파 연산 결과는 빨간색 빈칸으로 표시하여 구분하세요]

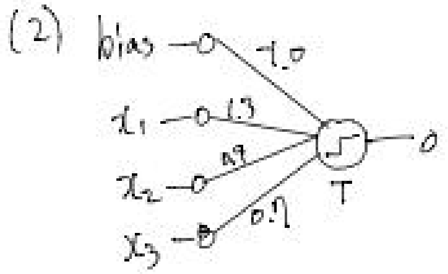


8번

(1) • $w = (1.3, 0.9, 0.7)$

• 원점에서의 거리 : $\dfrac{1}{\sqrt{1.3^2+0.9^2+0.7^2}} \doteqdot 0.578$

(2)



→ 이렇게 표현하게 바꾸면 평면 방정식이 원점을 지나게 됨

• bias = 1

9번

(1) t번째에 관해서 $x(t)$는 $w(t)$에 의해 이미 오분류되었음

$w^T(t)\,x(t)$의 부호와 $y(t)$의 부호는 달라야 하기 때문에

$y(t)\,w^T(t)\,x(t) < 0$

(2) $y(t)\,w^T(t+1)\,x(t)$

$= y(t)\{w(t) + y(t)x(t)\}^T x(t)$

$= y(t)\{w^T(t) + x^T(t)y^T(t)\}x(t)$

$= y(t)w^T(t)x(t) + y(t)x^T(t)y^T(t)x(t) > y(t)w^T(t)x(t)$

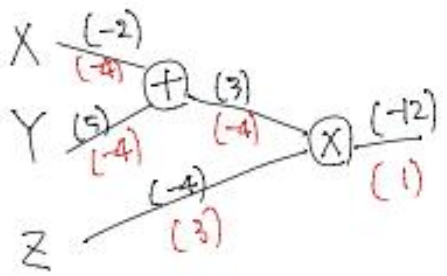
$\therefore y(t)x^T(t)y^T(t)x(t) \geq 0$

$= \underbrace{y(t)y^T(t)}_{\approx y^2} \underbrace{x^T(t)x(t)}_{\approx x^2} > 0$

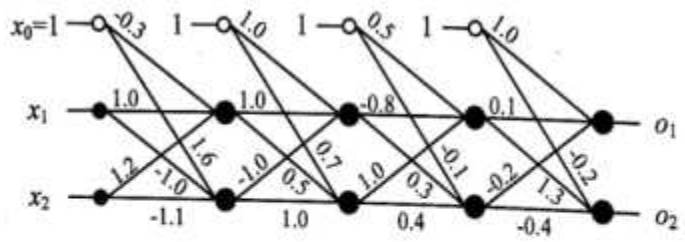
(3) $x(t)$가 입력층의 샘플에 따라서 목적함수는 $-y(t)w^T(t)x(t)$로 생각할 수 있고 $-y(t)w^T(t+1)x(t) < -y(t)w^T(t)x(t)$이므로 $w(t)$에서 $w(t+1)$로 이동하는 것이 $x(t)$를 분류하는데 좋은 방향으로 이동하게 된다.

(목적함수는 작아지는 방향으로 갱신되어야 한다).

II번



10. [MLP] 다음은 은닉층이 3개인 MLP입니다. (16점, 각 4점)



(1) 가중치 행렬 U^1, U^2, U^3, U^4를 쓰세요.

(2) $x=(1,0)^T$가 입력되었을 때 출력 $o=(o_1,o_2)^T$를 구하세요. 활성함수는 로지스틱 시그모이드를 사용하세요.

(3) $x=(1,0)^T$가 입력되었을 때 출력 $o=(o_1,o_2)^T$를 구하세요. 활성함수는 ReLU를 사용하세요.

(4) $x=(1,0)^T$의 기대 출력이 $o=(0,1)$일 때, 현재 1.0인 u^3_{12} 가중치를 0.9로 줄이면 오류에 어떤 영향을 미치는지 설명하세요.

(1) 가중치 행렬 U^1, U^2, U^3, U^4를 쓰세요.

U1 = [-0.3 1.0 1.2]

[1.6 -1.0 -1.1]

U2 = [1.0 1.0 -1.0]

[0.7 0.5 1.0]

U3 = [0.5 -0.8 -0.1]

[-0.1 0.3 0.4]

U1 = [1.0 1.0 -0.2]

[-0.2 1.3 -0.4]