[2장 기계 학습과 수학]

기계 학습 알고리즘을 설계한다는 말은 (1) 목적함수를 정의하고 (2) 최적화 방법을 선택하며 (3) 적절한 제어 (규제, 모멘텀, 학습률, 멈춤조건 등) 기능을 추가하는 작업.

2.1 선형 대수 (Linear Algebra)

1. 벡터와 행렬 (vector and matrix)

   - Iris 데이터베이스의 붓꽃 샘플은 꽃받침의 길이, 꽃받침의 너비, 꽃잎의 길이, 꽃잎의 너비라는 “4개의 실숫값”을 가지고, 총 “150개의 샘플”을 제시하고 있음.
   - 기계 학습에서는 입력된 샘플을 특징 벡터로 (아무 말이 없으면 “열벡터”(column vector)) 표현한다. Iris 데이터베이스의 1개 샘플을 벡터로 표시하면,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{bmatrix} \in \Re^4$$

   - Iris 데이터베이스의 샘플들을 순차적으로 벡터로 표시하면

$$\mathbf{x}_1 = \begin{bmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} 4.9 \\ 3.0 \\ 1.4 \\ 0.2 \end{bmatrix} \quad \mathbf{x}_3 = \begin{bmatrix} 4.7 \\ 3.2 \\ 1.3 \\ 0.2 \end{bmatrix} \quad \cdots \quad \mathbf{x}_{150} = \begin{bmatrix} 5.9 \\ 3.0 \\ 5.1 \\ 1.8 \end{bmatrix}$$

- 행렬은 여러 개의 벡터는 담을 수 있다. Iris 데이터베이스를 행렬로 표시하면

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \vdots \\ \mathbf{x}_{150}^T \end{bmatrix} = \begin{bmatrix} 5.1 & 3.5 & 1.4 & 0.2 \\ 4.9 & 3.0 & 1.4 & 0.2 \\ 4.7 & 3.2 & 1.3 & 0.2 \\ \vdots & \vdots & \vdots & \vdots \\ 5.9 & 3.0 & 5.1 & 1.8 \end{bmatrix} \in \Re^{150\times 4}$$

  여기서 $\mathbf{x}^T$는 행과 열을 교환한 형태로 $\mathbf{x}$의 “전치”(transpose)라고 한다.

- 기계 학습에서는 훈련집합을 담은 행렬을 “설계행렬”(design matrix)이라고 한다.
- (예제 2-1)

$$\begin{aligned} f(\mathbf{x}) &= f(x_1, x_2, x_3) \\ &= 2x_1^2 - 4x_1x_2 + 3x_1x_3 + x_2x_1 + 2x_2^2 + 6x_2x_3 - 2x_3x_1 + 3x_3x_2 + 2x_3^2 + 2x_1 + 3x_2 - 4x_3 + 5 \\ &= \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} 2 & -4 & 3 \\ 1 & 2 & 6 \\ -2 & 3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 2 & 3 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + 5 \\ &= \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c \end{aligned}$$

- 특수한 행렬
  a) 정사각행렬(square matrix): 행의 개수와 열의 개수가 같은 행렬
  b) 대각행렬(diagonal matrix): 주 대각선을 제외한 요소가 모두 0인 행렬
  c) 단위행렬(identity matrix): 모든 주대각선 요소가 1이고, 나머지 요소가 모두 0인 정사각행렬, $\mathbf{I}$로 표기
  d) 대칭행렬(symmetric matrix): a_{ij}와 a_{ji}가 같은 정사각행렬, $\mathbf{A}^T = \mathbf{A}$인 행렬

- 행렬 연산 (matrix operation)

  a) 두 행렬의 덧셈은 해당하는 요소끼리 더함. $\mathbf{A}+\mathbf{B}$

  b) 행렬에 스칼라를 곱할 때는 스칼라 값을 요소별로 곱함. $\alpha\mathbf{A}$

  c) 행렬의 곱

$$\mathbf{C}=\mathbf{AB} \qquad c_{ij}=\sum_{k=1}^{s} a_{ik}b_{kj}=a_{i1}b_{1j}+a_{i2}b_{2j}+\cdots+a_{is}b_{sj}$$

  d) 행렬 곱셈의 교환법칙은 성립하지 않지만, "분배법칙과 결합법칙은 성립"함.

$$\mathbf{AB}\neq\mathbf{BA} \qquad \mathbf{A}(\mathbf{B}+\mathbf{C})=\mathbf{AB}+\mathbf{AC} \qquad \mathbf{A}(\mathbf{BC})=(\mathbf{AB})\mathbf{C}$$

  e) 두 벡터의 "내적" (dot product, inner product, scalar product)

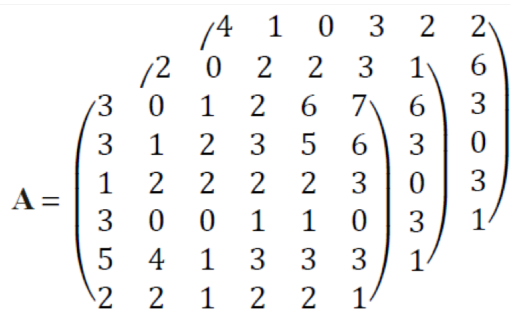
$$\mathbf{a}\cdot\mathbf{b}=\mathbf{a}^T\mathbf{b}=\sum_{k=1}^{d} a_k b_k = a_1b_1+a_2b_2+\cdots+a_db_d$$

  f) (예제 2-2)

$$\mathbf{A}=\begin{bmatrix}3&4&1\\0&5&2\end{bmatrix} \qquad \mathbf{B}=\begin{bmatrix}2&0&1\\1&0&5\\4&5&1\end{bmatrix} \qquad \mathbf{a}=\begin{bmatrix}2\\1\\3\end{bmatrix} \qquad \mathbf{x}_1=\begin{bmatrix}5.1\\3.5\\1.4\\0.2\end{bmatrix} \qquad \mathbf{x}_2=\begin{bmatrix}4.9\\3.0\\1.4\\0.2\end{bmatrix}$$

$$\mathbf{AB}=\begin{bmatrix}3&4&1\\0&5&2\end{bmatrix}\begin{bmatrix}2&0&1\\1&0&5\\4&5&1\end{bmatrix}=\begin{bmatrix}14&5&24\\13&10&27\end{bmatrix} \qquad \mathbf{Aa}=\begin{bmatrix}3&4&1\\0&5&2\end{bmatrix}\begin{bmatrix}2\\1\\3\end{bmatrix}=\begin{bmatrix}13\\11\end{bmatrix} \qquad \mathbf{x}_1^T\mathbf{x}_2=37.49$$

- 텐서 (tensor)
  a) 0차원 텐서(0th-order tensor): 스칼라 (scalar)
  b) 1차원 텐서(1st-order tensor): 벡터 (vector)
  c) 2차원 텐서(2nd-order tensor): 행렬 (흑백 이미지) (matrix)
  d) 3차원 텐서(3rd-order tensor): 3차원 숫자 배열 (컬러 이미지 : RGB 영상을 입력받는 경우 2차원 행렬이 3장 있는 셈)



2. 놈(norm)과 유사도 (similarity)

- 기계 학습이 사용하는 중요한 연산 중 하나는 두 샘플의 "유사도" 측정이다.
- 놈 : 벡터의 크기

a) p차 놈 $\|\mathbf{x}\|_p = \left(\sum_{i=1}^{d} |x_i|^p\right)^{\frac{1}{p}}$, for $p = 1, 2, \cdots, \infty$

$$\|\mathbf{x}\|_1 = \left(\sum_{i=1}^{d} |x_i|\right) = |x_1| + |x_2| + \cdots + |x_d|$$

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{d} |x_i|^2\right)^{\frac{1}{2}} = \sqrt{|x_1|^2 + |x_2|^2 + \cdots + |x_d|^2}$$

$$\vdots$$

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

놈 부등식 (norm inequality)

$$\|\mathbf{x}\|_\infty \le \|\mathbf{x}\|_2 \le \|\mathbf{x}\|_1$$

예를 들어, $\mathbf{x} = \begin{bmatrix} 3 & -4 & 1 \end{bmatrix}^T$

$$\|\mathbf{x}\|_1 = |3| + |-4| + |1| = 8$$
$$\|\mathbf{x}\|_2 = \sqrt{|3|^2 + |-4|^2 + |1|^2} = \sqrt{26} = 5.099$$
$$\|\mathbf{x}\|_\infty = \max\{|3|, |-4|, |1|\} = 4$$

b) 단위 벡터 (unit vector) : 크기가 1인 벡터 $\frac{\mathbf{x}}{\|\mathbf{x}\|_2}$

$$\text{단위벡터 :}\frac{\mathbf{x}}{\|\mathbf{x}\|_2} = \begin{bmatrix} \frac{3}{5.099} \\ \frac{-4}{5.099} \\ \frac{1}{5.099} \end{bmatrix}$$

c) 행렬 놈 (matrix norm) : 프로베니우스 놈 (Frobenius norm)

$$\|\mathbf{A}\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij}^2 \right)^{\frac{1}{2}} = \sqrt{a_{11}^2 + a_{12}^2 + \cdots + a_{nm}^2}$$

예를 들어, $\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 6 & 4 \end{bmatrix}$

$$\|\mathbf{A}\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij}^2 \right)^{\frac{1}{2}} = \sqrt{2^2 + 1^2 + 6^2 + 4^2} = 7.55$$

- 유사도와 거리 (similarity and distance)
  a) 내적은 두 벡터가 똑같을 때 가장 큰 값이 되고, 두 벡터가 수직일 때 0이 되어 가장 작은 값이 된다.
  b) 내적은 두 벡터의 방향이 달라질수록 값이 작아지므로 유사도 측정에 사용할 수 있다.
  c) 하지만 벡터의 크기가 크면 유사도가 증가하는 경향이 있어서 "단위벡터로 변환"한 다음, 단위벡터의 "내적을 사용"하는데, 이런 유사도 측정 방법을 "코사인 유사도"(cosine similarity) 라고 한다.

$$cosine\_similarity(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a}}{\|\mathbf{a}\|} \cdot \frac{\mathbf{b}}{\|\mathbf{b}\|} = \cos(\theta)$$

예를 들어, 그림 2-2(a)에서 $\mathbf{x}$는 $\mathbf{x}_1$과 $\mathbf{x}_2$ 중 $\mathbf{x}_1$에 더 유사하다.

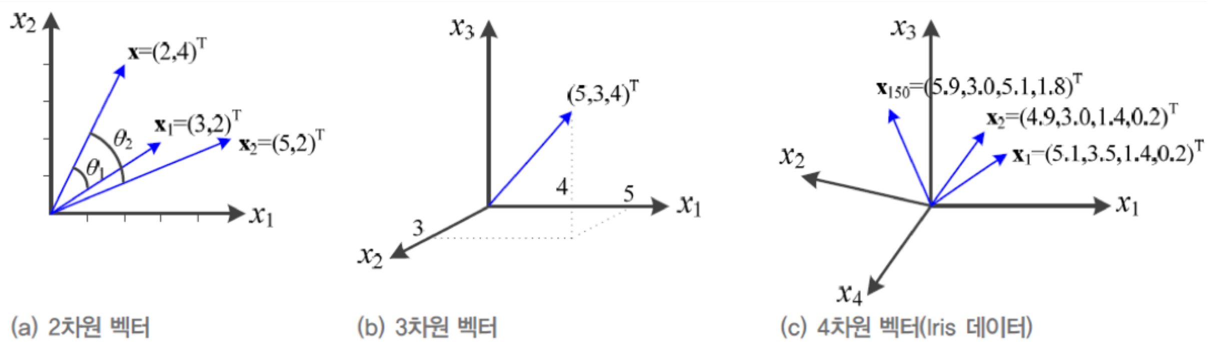


그림 2-2 벡터를 기하학적으로 해석

$$\cos(\theta_1) = \frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot \frac{\mathbf{x}_1}{\|\mathbf{x}_1\|} = \begin{bmatrix} \frac{2}{\sqrt{20}} \\ \frac{4}{\sqrt{20}} \end{bmatrix} \cdot \begin{bmatrix} \frac{3}{\sqrt{13}} \\ \frac{2}{\sqrt{13}} \end{bmatrix} = 0.8682$$

$$\cos(\theta_2) = \frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|} = \begin{bmatrix} \frac{2}{\sqrt{20}} \\ \frac{4}{\sqrt{20}} \end{bmatrix} \cdot \begin{bmatrix} \frac{5}{\sqrt{29}} \\ \frac{2}{\sqrt{29}} \end{bmatrix} = 0.7474$$

d) 코사인 유사도는 정보검색 분야에서 주로 두 문서의 유사도를 계산하는 데 사용

e) 두 벡터 사이의 거리는 “유클리디언 거리”(Euclidean distance)를 사용

$$dist(\mathbf{a}, \mathbf{b}) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_d - b_d)^2}$$

f) 해밍 거리 (Hamming distance): 2진 벡터인 경우 서로 다른 값을 가진 요소의 개수

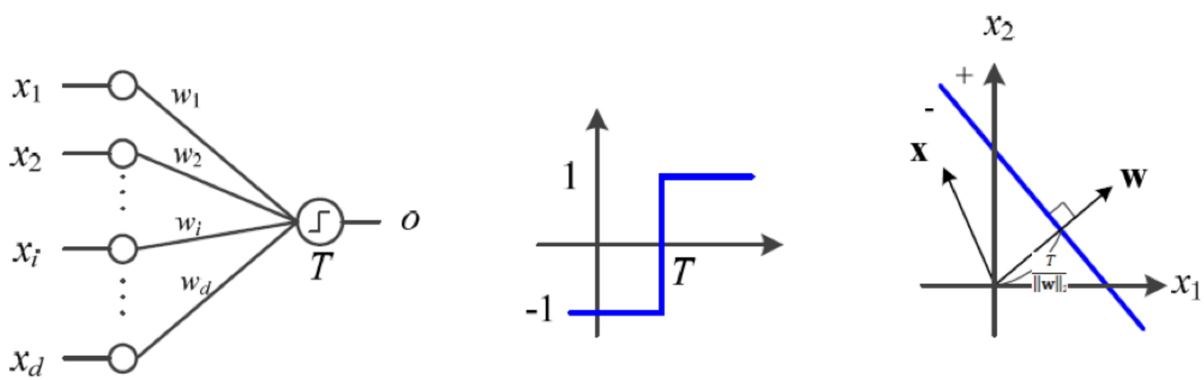
g) 거리를 유사도로 변환하려면 아래의 3가지 중 하나를 선택하여 사용할 수 있음.

$$-dist(\mathbf{a}, \mathbf{b}) \quad \text{or} \quad R - dist(\mathbf{a}, \mathbf{b}) \quad \text{or} \quad \frac{1}{dist(\mathbf{a}, \mathbf{b})}$$

여기서 R은 발생 가능한 최대 거리 값이다.

3. 퍼셉트론의 해석

- 로젠블렛이 고안한 최초의 퍼셉트론 : d차원의 특징 벡터 $\mathbf{x}$를 입력할 수 있도록 d개의 입력 노드가 있고, o로 표시된 1개의 출력노드가 있으며, 입력노드와 출력노드는 "에지"로 연결되어 있는데, 에지마다 "가중치"가 부여되어 있음.



(a) 퍼셉트론 구조 (b) 계단형 활성함수(비선형) (c) 퍼셉트론의 공간 분할

그림 2-3 퍼셉트론의 구조와 동작

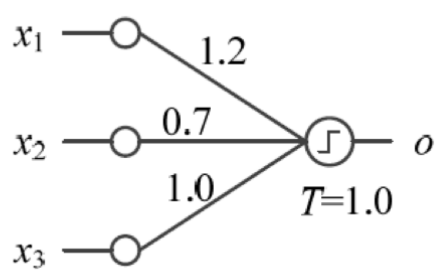
- 퍼셉트론 동작 : $\mathbf{w} \cdot \mathbf{x}$는 입력노드와 해당 에지의 가중치를 곱하여 더한 값을 "활성값"이라고 하고, 이를 활성함수에 넣었을 때 출력되는 값이 o이며, 이때 활성함수는 계단함수이다.

$$o = \tau(\mathbf{w} \cdot \mathbf{x}) = \tau(w_1 x_1 + w_2 x_2 + \cdots + w_d x_d) \qquad \textbf{where} \qquad \tau(a) = \begin{cases} 1 & a \geq T \\ -1 & a < T \end{cases}$$

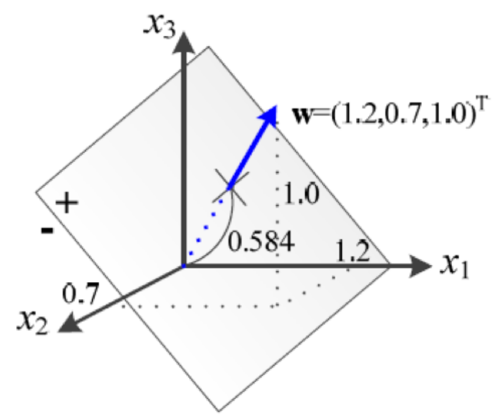
- 퍼셉트론이 하는 일 : 전체 공간을 +1인 부분공간과 -1인 부분공간으로 구분하는 "결정직선"(decision line)을 생성한다. 결정직선은 $\mathbf{w}$에 "수직"이고 원점으로 부터 $\frac{T}{\|\mathbf{w}\|_2}$ 만큼 떨어져 있다.
  - a) $\mathbf{w} \cdot \mathbf{x} = T$: 결정직선에 위치한 점
  - b) $\mathbf{w} \cdot \mathbf{x} > T$: 결정직선의 "위쪽"
  - c) $\mathbf{w} \cdot \mathbf{x} < T$: 결정직선의 "아래쪽"

- 퍼셉트론은 특징 벡터 $\mathbf{x}$를 두 부류 중 하나로 분류하는 "분류기"(classifier)이다.
  a) 2차원 특징벡터 : 결정직선
  b) 3차원 특징벡터 : 결정평면 (decision plane)
  c) 4차원 이상 특징벡터 : 결정초평면 (decision hyperplane)
- (예제 2-3) 에지의 가중치 벡터 $\mathbf{w} = [1.2,\ 0.7,\ 1.0]^T$이고 임계값 $T = 1.0$이라 가정하면, 퍼셉트론은 특징벡터 $\mathbf{x} = [x_1,\ x_2,\ x_3]^T$에 대해서 평면의 방정식을 형성한다.

$$\mathbf{w} \cdot \mathbf{x} = T \quad \rightarrow \quad 1.2x_1 + 0.7x_2 + 1.0x_3 = 1.0$$



(a) 퍼셉트론



(b) 공간 분할(2부류 분류)

그림 2-4 퍼셉트론의 예(3차원)

a) $\mathbf{x} = [1.0,\ 1.0,\ 1.0]^T$이면 +로 분류
b) $\mathbf{x} = [-1.0,\ 1.0,\ 1.0]^T$이면 -로 분류

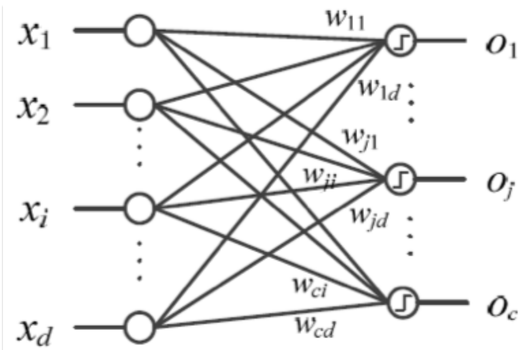


그림 2-5 출력이 여러 개인 퍼셉트론

- 여러개의 퍼셉트론을 묶어 그림 2-5처럼 확장해 보자

$$\begin{aligned} o_1 &= \tau(w_{11}x_1 + w_{12}x_2 + \cdots + w_{1d}x_d) \\ o_2 &= \tau(w_{21}x_1 + w_{22}x_2 + \cdots + w_{2d}x_d) \\ &\vdots \\ o_c &= \tau(w_{c1}x_1 + w_{c2}x_2 + \cdots + w_{cd}x_d) \end{aligned} \qquad \begin{aligned} o_1 &= \tau(\mathbf{w}_1^T\mathbf{x}) \\ o_2 &= \tau(\mathbf{w}_2^T\mathbf{x}) \\ &\vdots \\ o_c &= \tau(\mathbf{w}_c^T\mathbf{x}) \end{aligned}$$

줄여서

$$\mathbf{o} = \tau\left(\begin{bmatrix} \mathbf{w}_1^T\mathbf{x} \\ \mathbf{w}_2^T\mathbf{x} \\ \vdots \\ \mathbf{w}_c^T\mathbf{x} \end{bmatrix}\right) = \tau(\mathbf{W}\mathbf{x})$$

여기서 $\mathbf{o} \in \Re^c$, $\mathbf{W} \in \Re^{c\times d}$ **and** $\mathbf{x} \in \Re^d$.

- 그림 2-5의 신경망은 입력된 특징 벡터 $\mathbf{x}$에 대해 c개의 부류와 유사도를 계산한다. 값이 가장 큰 출력노드를 분류 결과로 취한다면, 이 신경망을 “c 부류 분류기”로 사용할 수 있다.

- 학습의 정의

  a) 분류 ($\mathbf{o}$ 찾는 것) : 특징벡터 $\mathbf{x}$와 $\mathbf{W}$를 알고 있을 때 $\mathbf{o}$를 알아내는 것

  b) 학습 ($\mathbf{W}$ 찾는 것) : 특징벡터 $\mathbf{x}$와 부류 정보 $\mathbf{o}$의 쌍이 주어졌을 때, 샘플을 제대로 분류하는 $\mathbf{W}$를 구하는 문제가 바로 기계 학습이다.

$$\mathbf{o} = \tau(\mathbf{W}\mathbf{x})$$

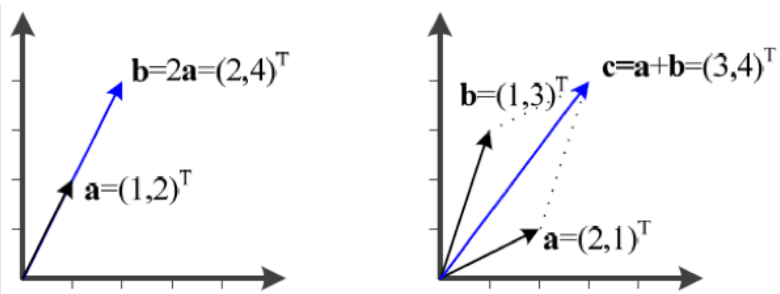
  아무리 복잡하고 깊은 신경망이라도 퍼셉트론을 여러 층 확장한 것에 불과하다.

4. 선형결합과 벡터공간 (linear combination and vector space)

- 벡터의 기본 연산 : 스칼라 곱 (scalar multiplcation), 덧셈 (addition)
- 벡터 공간 (vector space) : 두 벡터의 선형 결합으로 만들어지는 공간

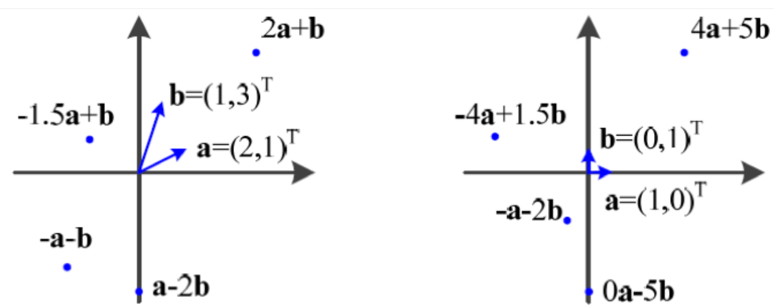
$$\mathbf{c} = \alpha_1 \mathbf{a} + \alpha_2 \mathbf{b}$$

여기서 벡터공간을 만드는 벡터들을 기저 벡터(basis vector)라고 하며, 크기가 1이고 서로 수직인 기저 벡터를 정규직교 기저 벡터(orthonormal basis vector)라고 한다.



(a) 벡터에 스칼라 곱 (b) 두 벡터의 덧셈

그림 2-6 벡터의 연산



(a) 기저 벡터와 벡터공간 (b) 정규직교 기저 벡터

그림 2-7 벡터공간

- 한 기저 벡터를 나머지 기저 벡터(basis vector)들의 선형결합으로 만들 수 없을 때, 두 기저 벡터는 선형 독립(linearly independent)이라고 한다.
- 행렬의 계수 (rank) : 벡터 공간을 펼치는 선형 독립인 기저 벡터의 개수. 계수가 1이면 1차원 직선이고, 2이면 평면을, 3이면 3차원 공간을 덮는다. 행렬의 계수가 벡터의 차원과 같으면 행렬이 최대 계수(full rank)를 가진다고 한다.
- 기계 학습의 공간 변환 : 기계 학습에서 행렬의 가장 중요한 역할은 "공간 변환"이다.

  a) 그림 2-3의 퍼셉트론은 d차원의 특징공간을 1차원으로 변환

b) 그림 2-5의 신경망은 d차원의 특징공간을 c차원 부류공간으로 변환
c) 퍼셉트론은 한 단계의 변환만 수행하는 원시적인 구조
d) 깊은 신경망은 여러 단계의 은닉층을 거치면서 특징공간을 점점 목적에 적합한 형태로 변환. 이러한 공간변환 능력 때문에 높은 성능을 보장한다.

5. 역행렬

- 역행렬의 정의 : $AB = BA = I$를 만족하면 B는 A의 역행렬이다.
- 특이행렬 (singular matrix) : 역행렬이 없는 행렬
- 행렬식 (determinant) : 도형의 넓이 또는 부피를 확대하는 비율을 나타낸다.

$$\det\begin{bmatrix} a & b \\ c & d \end{bmatrix} = a\cdot(-1)^{1+1}\det(d) + b\cdot(-1)^{1+2}\det(c) = ad - bc$$

$$\begin{aligned}\det\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} &= a\cdot(-1)^{1+1}\det\begin{bmatrix} e & f \\ h & i \end{bmatrix} + b\cdot(-1)^{1+2}\det\begin{bmatrix} d & f \\ g & i \end{bmatrix} + c\cdot(-1)^{1+3}\det\begin{bmatrix} d & e \\ g & h \end{bmatrix} \\ &= a(ei - fh) - b(di - fg) + c(dh - eg)\end{aligned}$$

- 역행렬

$$A^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{adj(A)}{\det(A)} = \frac{1}{\det(A)}\begin{bmatrix} (-1)^{1+1}\det(d) & (-1)^{1+2}\det(b) \\ (-1)^{2+1}\det(c) & (-1)^{2+2}\det(a) \end{bmatrix} = \frac{1}{ad - bd}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}^{-1} = \frac{adj(A)}{\det(A)} = \frac{1}{\det(A)}\begin{bmatrix} (-1)^{1+1}\det\begin{bmatrix} e & f \\ h & i \end{bmatrix} & (-1)^{1+2}\det\begin{bmatrix} b & c \\ h & i \end{bmatrix} & (-1)^{1+3}\det\begin{bmatrix} b & c \\ e & f \end{bmatrix} \\ (-1)^{2+1}\det\begin{bmatrix} d & f \\ g & i \end{bmatrix} & (-1)^{2+2}\det\begin{bmatrix} a & c \\ g & i \end{bmatrix} & (-1)^{2+3}\det\begin{bmatrix} a & c \\ d & f \end{bmatrix} \\ (-1)^{3+1}\det\begin{bmatrix} d & e \\ g & h \end{bmatrix} & (-1)^{3+2}\det\begin{bmatrix} a & b \\ g & h \end{bmatrix} & (-1)^{3+3}\det\begin{bmatrix} a & b \\ d & e \end{bmatrix} \end{bmatrix}$$

- 정부호 행렬 (definite matrix) : 스칼라 처럼 행렬에 부여하는 “부호”

  a) 양의 정부호 행렬 (positive definite matrix) : 0이 아닌 모든 벡터에 대해서 $\mathbf{x}^T A\mathbf{x} > 0$ (고윳값이 모두 양수)

  b) 양의 준정부호 행렬 (positive semi-definite matrix) : 0이 아닌 모든 벡터에 대해서 $\mathbf{x}^T A\mathbf{x} \geq 0$ (고윳값이 양수와 0)

  c) 음의 정부호 행렬 (negative definite matrix) : 0이 아닌 모든 벡터에 대해서 $\mathbf{x}^T A\mathbf{x} < 0$ (고윳값이 모두 음수)

  d) 음의 준정부호 행렬 (negative semi-definite matrix) : 0이 아닌 모든 벡터에 대해서 $\mathbf{x}^T A\mathbf{x} \leq 0$ (고윳값이 음수와 0)

  e) 부정부호 행렬 (indefinite matrix) : 고윳값이 양수도 있고, 음수도 있는 행렬

6. 행렬 분해 (matrix decomposition)

- 고윳값과 고유벡터 (eigenvalue and eigenvector) : $A \in \Re^{n\times n}$ 행렬에 어떤 벡터 방향을 곱할 때 같은 방향으로 결과가 형성되는 경우, 그 방향 벡터를 고유벡터라고 하고, 고유벡터에 대응하는 배수 λ를 고윳값이라고 한다.

$$A\mathbf{v}_i = \lambda_i \mathbf{v}_i \qquad i = 1, 2, 3, \cdots, n$$

- (예제 2-5) $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

$$(A - \lambda I)\mathbf{v} = \begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} \mathbf{v} = 0$$

특이행렬 $\det(A - \lambda I) = (2-\lambda)^2 - 1 = \lambda^2 - 4\lambda + 3 = (\lambda - 3)(\lambda - 1) = 0$

$$\lambda_1 = 3 \quad (A - \lambda_1 I)\mathbf{v}_1 = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \mathbf{v}_1 = 0 \quad \rightarrow \quad \mathbf{v}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$\lambda_2 = 1 \quad (A - \lambda_2 I)\mathbf{v}_2 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \mathbf{v}_2 = 0 \quad \rightarrow \quad \mathbf{v}_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$$

- 고유벡터들은 항상 서로 “직교”(orthogonal)한다.

7. 고윳값 분해 (eigenvalue decomposition)

$$A \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & & & 0 \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \quad \rightarrow \quad AQ = Q\Lambda \quad \rightarrow \quad A = Q\Lambda Q^{-1}$$

예를 들어,

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^{-1} = Q\Lambda Q^{-1}$$

고윳값 분해는 정사각행렬(square matrix)이 아니면 적용할 수 없다.

8. 특이값 분해 (singular value decomposition)

- 정사각행렬이 아니어도 적용할 수 있다. $A \in \Re^{n\times m}$

$$A = U\Sigma V^T$$

- 왼쪽 특이행렬 $U \in \Re^{n\times n}$: AA^T의 고유벡터를 열에 배치한 행렬
- 오른쪽 특이행렬 $V \in \Re^{m\times m}$: A^TA의 고유벡터를 열에 배치한 행렬
- 특이행렬 $\Sigma \in \Re^{n\times m}$: AA^T의 고윳값의 제곱근을 대각선에 배치한 대각 행렬

2.2 최적화

기계 학습에서는 데이터로 "미분"(differential)을 계산하면서 목적함수 or 비용함수(objective function or cost function)의 최적점을 찾아가는 "스토케스틱 경사하강법"(stochastic gradient decent rule)을 사용한다.

1. 매개변수 공간의 탐색

   - 기계 학습은 적절한 모델을 선택하고 목적함수를 정의하며, 모델의 매개변수 공간을 탐색하여 목적함수가 최저가 되는 최적점을 찾아가는 전략을 사용
   - 모델의 매개변수 공간은 특징 공간보다 수배~수만배 넓다.
   - 기계 학습 알고리즘이 해야할 일

$$J(\boldsymbol{\Theta})\text{를 최소로 하는 최적해 }\hat{\boldsymbol{\Theta}}\text{을 찾아라. 즉, } \hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}}\, J(\boldsymbol{\Theta}) \tag{2.50}$$

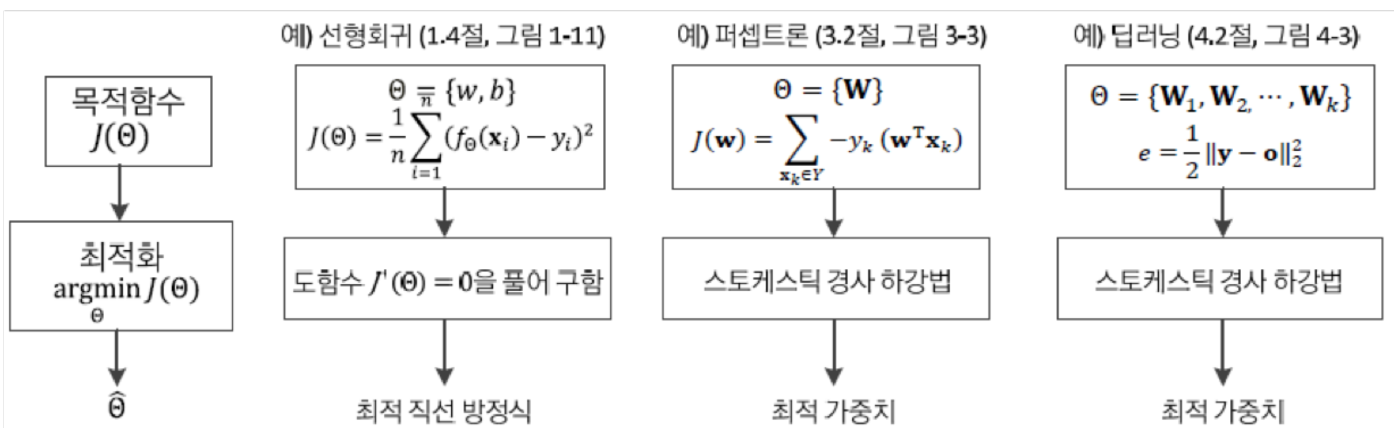


그림 2-22 최적화를 이용한 기계 학습의 문제풀이 과정

알고리즘 2-3 기계 학습이 사용하는 전형적인 탐색 알고리즘(1장의 [알고리즘 1-1]과 같음)

입력 : 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$
출력 : 최적해 $\hat{\mathbf{\Theta}}$

```
1  난수를 생성하여 초기해 Θ을 설정한다.
2  repeat
3      J(Θ)가 작아지는 방향 dΘ를 구한다.
4      Θ = Θ + dΘ
5  until(멈춤 조건)
6  Θ̂ = Θ
```

- 기계 학습이 사용하는 전형적인 알고리즘
- 알고리즘 2-3의 라인 3에서는 목적함수가 작아지는 방향을 주로 "미분"으로 찾아냄

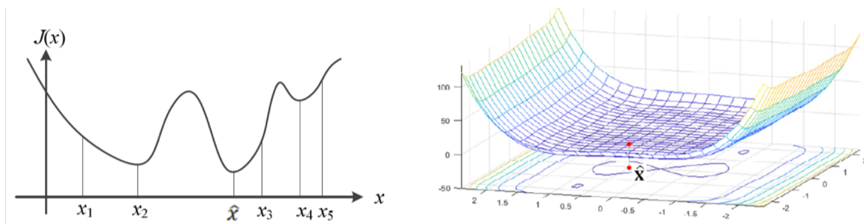


그림 2-23 최적해 탐색

2. 미분

- 미분에 의한 최적화
  a) 목적함수의 최저점을 찾아가는 길잡이 : 미분
  b) 도함수 값이 +이면 -방향으로 가야 최저점을 만나고, -이면 +방향으로 가야 최저점을 만나게 된다. $-f'(x)$ 방향으로 가야 최저점을 찾을 수 있다. (경사하강법의 핵심원리)
  c) 알고리즘 2-3과 같이 $d\Theta$ 만큼 조금씩 이동하는 일을 반복하여 최저점을 찾는 접근방법을 수치적 방법이라고 한다.

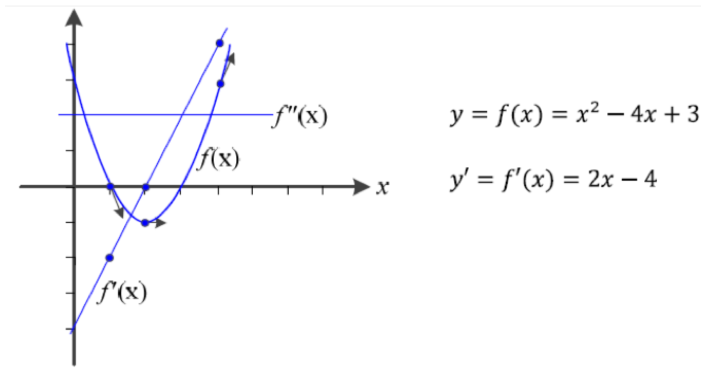


그림 2-24 간단한 미분 예제

- 편미분 (partial differential)

  a) 편미분 : 여러변수의 함수일때, 변수 각각에 대해 독립적으로 하는 미분

  b) 그레이디언트 (gradient) : 편미분이 이루는 벡터, $\nabla f = \frac{df}{d\mathbf{x}} = f'(\mathbf{x})$

$$f(\mathbf{x}) = f(x_1, x_2) = 5x_1^2 + 2x_1x_2 + 7x_2^2$$
$$\nabla f = \frac{df}{d\mathbf{x}} = \begin{bmatrix} \frac{df}{dx_1} \\ \frac{df}{dx_2} \end{bmatrix} = \begin{bmatrix} 10x_1 + 2x_2 \\ 2x_1 + 14x_2 \end{bmatrix}$$

- 독립변수(independent variable)와 종속변수(dependent variable)의 구분 & 연쇄법칙 (chain rule)

  a) 함수 $y = f(x) = x^2 - 4x + 3$로 가정하면, y는 종속변수이고, x는 독립변수이다. 이의 미분은 다음과 같다

$$y' = f'(x) = \frac{df}{dx} = 2x - 4$$

b) 합성함수 $y = g(h(x))$로 가정하면, 미분을 적용하는데 있어서 "연쇄법칙"을 적용한다.

$$y' = g'(x) = \frac{dg}{dx} = \frac{dg}{dh}\frac{dh}{dx}$$

c) (예) $y = 3(2x^2-1)^2 - 2(2x^2-1) + 5$이면, 여기서 $h(x) = 2x^2 - 1$로 가정하면 $y = 3h^2(x) - 2h(x) + 5$로 표시할 수 있다. 이의 미분은 다음과 같다.

$$\begin{aligned} y' = g'(x) &= \frac{dg}{dx} \\ &= \frac{dg}{dh}\frac{dh}{dx} \\ &= [6h(x)-2][4x] = [6(2x^2-1)-2][4x] \\ &= 48x^3 - 32x \end{aligned}$$

d) 합성함수 $y = g(h(i(x)))$로 가정하면, 미분을 적용하는데 있어서 연속적으로 연쇄법칙을 적용한다.

$$y' = g'(x) = \frac{dg}{dx} = \frac{dg}{dh}\frac{dh}{di}\frac{di}{dx}$$

e) 다층 퍼셉트론 구조에서 $\frac{\partial o_1}{\partial u^1_{23}}$를 계산할 때 연쇄법칙 사용

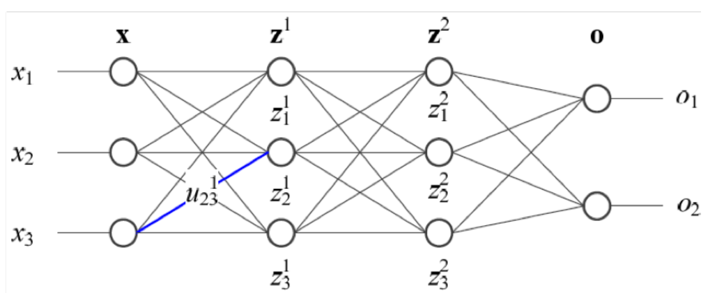


그림 2-26 다층 퍼셉트론은 합성함수

- 자코비안 (Jacobian)

  a) d차원 벡터함수 $\mathbf{f} : \Re^d \rightarrow \Re^m$라면, 이를 벡터 $\mathbf{x} \in \Re^d$로 미분하면 행렬을 얻을 수 있는데, 이를 자코비안이라 한다.

$$J(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}^T} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_d} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_d} \\ \vdots & & & \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_d} \end{bmatrix} \quad \textbf{when} \quad \mathbf{f} = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix} \quad \textbf{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

  b) (예)

$$J(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}^T} = \begin{bmatrix} 2 & 2x_2 \\ -2x_1 & 3 \\ 4x_2 & 4x_1 \end{bmatrix} \quad \textbf{when} \quad \mathbf{f} = \begin{bmatrix} 2x_1 + x_2^2 \\ -x_1^2 + 3x_2 \\ 4x_1x_2 \end{bmatrix} \quad \textbf{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- 헤시안 (2계 도함수) (Hessian)

  a) 벡터함수에 대한 헤시안은 3차원 텐서가 되며, 스칼라함수에 대한 헤시안은 2차원 텐서인 행렬이 된다.

$$H(\mathbf{x}) = \frac{\partial^2 f}{\partial \mathbf{x}^2} = \frac{\partial}{\partial \mathbf{x}^T}\left(\frac{\partial f}{\partial \mathbf{x}^T}\right)^T = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & & \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial^2 x_n} \end{bmatrix} \quad \textbf{when} \quad f(\mathbf{x}) \in \Re \quad \textbf{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

b) (예) $f(\mathbf{x}) = x_1^2 + 2x_1x_2 + 3x_2^2$ **and** $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$$J(\mathbf{x}) = \frac{\partial f}{\partial \mathbf{x}^T} = \begin{bmatrix} 2x_1 + 2x_2 & 2x_1 + 6x_2 \end{bmatrix} \in \Re^{1\times 2}$$
$$H(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}^T}\left(\frac{\partial f}{\partial \mathbf{x}^T}\right)^T = \begin{bmatrix} 2 & 2 \\ 2 & 6 \end{bmatrix} \in \Re^{2\times 2}$$

- 테일러 급수 **(Taylor series)**

a) 어떤 점 x에서의 함수값 $f(x)$와 미분 $f'(x)$가 주어지고, 이웃한 점에서의 함수값을 추정해야하는 경우 테일러 급수를 사용할 수 있다

$$\begin{aligned} f(x + \Delta x) &= f(x) + f'(x)\Delta x + \frac{1}{2!}f''(x)\Delta x^2 + \cdots \\ &\approx f(x) + f'(x)\Delta x \end{aligned}$$

b) (예제 **2-11**) $f(x) = \frac{1}{2}x^2$일 때, $f(1.0) = 0.5$와 $f'(1.0) = 1.0$을 알고 있을 때, $f(1.5)$를 추정하라?

$$f(1.0 + 0.5) \approx f(1.0) + f'(1.0) \cdot 0.5 = 1.0$$

실제 함수값은 $f(1.5) = \frac{1}{2} \cdot 1.5^2 = 1.125$로 추정치 1.0과 유사하다.

3. 경사 하강 알고리즘 (gradient decent rule)

- 미분으로 알아낸 그레이디언트 $\mathbf{g} = \frac{\partial J}{\partial \Theta}$는 오르막 방향을 가르킴. 반대 방향으로 이동하여야 최소값을 만날 수 있음.

$$\Theta_{new} = \Theta - \rho \mathbf{g}$$

  여기서 ρ는 "학습률"(learning rate)을 의미한다.

- 배치 경사 하강 알고리즘 (batch gradient decent) : 샘플의 그레이디언를 평균한 후 한꺼번에 갱신
- 스토캐스틱 경사 하강 알고리즘 (stochastic gradient decent) : 한 샘플의 그레이디언트를 계산한 후 즉시 갱신

알고리즘 2-4 배치 경사 하강 알고리즘(BGD)

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

난수를 생성하여 초기해 $\mathbf{\Theta}$를 설정한다.
repeat
  $\mathbb{X}$에 있는 샘플의 그레이디언트 $\nabla_1, \nabla_2, \cdots, \nabla_n$을 계산한다.
  $\nabla_{total} = \frac{1}{n}\sum_{i=1,n} \nabla_i$ // 그레이디언트 평균을 계산
  $\mathbf{\Theta} = \mathbf{\Theta} - \rho \nabla_{total}$
until(멈춤 조건)
$\hat{\mathbf{\Theta}} = \mathbf{\Theta}$

알고리즘 2-5 스토캐스틱 경사 하강 알고리즘(SGD)

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

난수를 생성하여 초기해 $\mathbf{\Theta}$를 설정한다.
repeat
  $\mathbb{X}$의 샘플의 순서를 섞는다.
  for (i=1 to n)
    i번째 샘플에 대한 그레이디언트 ∇_i를 계산한다.
    $\mathbf{\Theta} = \mathbf{\Theta} - \rho \nabla_i$
until(멈춤 조건)
$\hat{\mathbf{\Theta}} = \mathbf{\Theta}$