



MACHINE 기계 학습 LEARNING

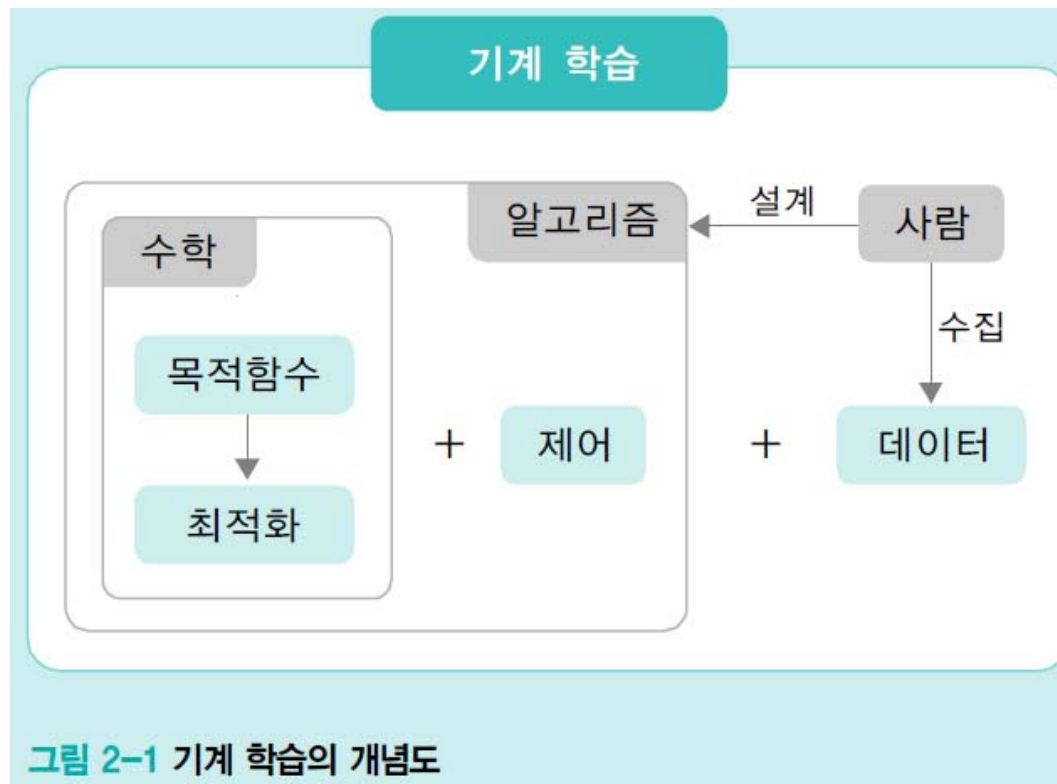
오일석 지음



PREVIEW

■ 기계 학습에서 수학의 역할

- **수학**은 목적함수를 정의하고, 목적함수가 최저가 되는 점을 찾아주는 최적화 이론 제공
- 최적화 이론에 규제, 모멘텀, 학습률, 멈춤조건과 같은 제어를 추가하여 **알고리즘** 구축
- **사람**은 알고리즘을 설계하고 데이터를 수집함



2.1 선형대수

- 2.1.1 벡터와 행렬
- 2.1.2 놈과 유사도
- 2.1.3 퍼셉트론의 해석
- 2.1.4 선형결합과 벡터공간
- 2.1.5 역행렬
- 2.1.6 행렬 분해

2.1.1 벡터와 행렬

■ 벡터

- 샘플을 특징 벡터로 feature vector 표현
- 예) Iris 데이터에서 꽃받침의 길이, 꽃받침의 너비, 꽃잎의 길이, 꽃잎의 너비라는 4개의 특징이 각각 5.1, 3.5, 1.4, 0.2인 샘플

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{pmatrix}$$

- 여러 개의 특징 벡터를 첨자로 구분

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

2.1.1 벡터와 행렬

■ 행렬

- 여러 개의 벡터를 담음
- 훈련집합을 담은 행렬을 설계행렬이라 부름
- 예) Iris 데이터에 있는 150개의 샘플을 설계 행렬 \mathbf{X} 로 표현

$$\mathbf{X} = \begin{pmatrix} 5.1 & 3.5 & 1.4 & 0.2 \\ 4.9 & 3.0 & 1.4 & 0.2 \\ 4.7 & 3.2 & 1.3 & 0.2 \\ 4.6 & 3.1 & 1.5 & 0.2 \\ \vdots & \vdots & \vdots & \vdots \\ 6.2 & 3.4 & 5.4 & 2.3 \\ 5.9 & 3.0 & 5.1 & 1.8 \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} \\ x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} \\ x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} \\ \vdots & \vdots & \vdots & \vdots \\ x_{149,1} & x_{149,2} & x_{149,3} & x_{149,4} \\ x_{150,1} & x_{150,2} & x_{150,3} & x_{150,4} \end{pmatrix}$$

← 행 row

↑
열 column

2.1.1 벡터와 행렬

- 행렬 \mathbf{A} 의 전치행렬 \mathbf{A}^T

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix}, \quad \mathbf{A}^T = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{pmatrix}$$

예를 들어, $\mathbf{A} = \begin{pmatrix} 3 & 4 & 1 \\ 0 & 5 & 2 \end{pmatrix}$ 라면 $\mathbf{A}^T = \begin{pmatrix} 3 & 0 \\ 4 & 5 \\ 1 & 2 \end{pmatrix}$

- Iris의 설계 행렬을 전치행렬 표기에 따라 표현하면,

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_{150}^T \end{pmatrix}$$

2.1.1 벡터와 행렬

■ 행렬을 이용하면 수학을 간결하게 표현할 수 있음

- 예) 다항식의 행렬 표현

$$f(\mathbf{x}) = f(x_1, x_2, x_3)$$

$$= 2x_1x_1 - 4x_1x_2 + 3x_1x_3 + x_2x_1 + 2x_2x_2 + 6x_2x_3 - 2x_3x_1 + 3x_3x_2 + 2x_3x_3 + 2x_1 + 3x_2 - 4x_3 + 5$$

$$= (x_1 \ x_2 \ x_3) \begin{pmatrix} 2 & -4 & 3 \\ 1 & 2 & 6 \\ -2 & 3 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + (2 \ 3 \ -4) \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + 5$$

$$= \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$$

■ 특수한 행렬들

$$\text{정사각행렬} \begin{pmatrix} 2 & 0 & 1 \\ 1 & 21 & 5 \\ 4 & 5 & 12 \end{pmatrix}, \quad \text{대각행렬} \begin{pmatrix} 50 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 8 \end{pmatrix},$$

$$\text{단위행렬} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \text{대칭행렬} \begin{pmatrix} 1 & 2 & 11 \\ 2 & 21 & 5 \\ 11 & 5 & 1 \end{pmatrix}$$

2.1.1 벡터와 행렬

■ 행렬 연산

■ 행렬 곱셈 $\mathbf{C} = \mathbf{AB}$, 이때 $c_{ij} = \sum_{k=1,s} a_{ik} b_{kj}$ (2.1)

2*3 행렬 $\mathbf{A} = \begin{pmatrix} 3 & 4 & 1 \\ 0 & 5 & 2 \end{pmatrix}$ 와 3*3행렬 $\mathbf{B} = \begin{pmatrix} 2 & 0 & 1 \\ 1 & 0 & 5 \\ 4 & 5 & 1 \end{pmatrix}$ 을 곱하면 2*3 행렬 $\mathbf{C} = \mathbf{AB} = \begin{pmatrix} 14 & 5 & 24 \\ 13 & 10 & 27 \end{pmatrix}$

- 교환법칙 성립하지 않음: $\mathbf{AB} \neq \mathbf{BA}$
- 분배법칙과 결합법칙 성립: $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$ 이고 $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$

■ 벡터의 내적

벡터의 내적 $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b} = \sum_{k=1,d} a_k b_k$ (2.2)

$\mathbf{x}_1 = \begin{pmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{pmatrix}$ 와 $\mathbf{x}_2 = \begin{pmatrix} 4.9 \\ 3.0 \\ 1.4 \\ 0.2 \end{pmatrix}$ 의 내적 $\mathbf{x}_1 \cdot \mathbf{x}_2$ 는 37.49

2.1.1 벡터와 행렬

■ 텐서

- 3차원 이상의 구조를 가진 숫자 배열
- 예) 3차원 구조의 RGB 컬러 영상

$$\mathbf{A} = \begin{pmatrix} 4 & 1 & 0 & 3 & 2 & 2 \\ 2 & 0 & 2 & 2 & 3 & 1 \\ 3 & 0 & 1 & 2 & 6 & 7 \\ 3 & 1 & 2 & 3 & 5 & 6 \\ 1 & 2 & 2 & 2 & 2 & 3 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 5 & 4 & 1 & 3 & 3 & 3 \\ 2 & 2 & 1 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 6 \\ 3 \\ 0 \\ 3 \\ 1 \end{pmatrix}$$

2.1.2 놈과 유사도

■ 벡터와 행렬의 크기를 놈으로 측정

■ 벡터의 p 차 놈

$$p\text{차 놈: } \|\mathbf{x}\|_p = \left(\sum_{i=1,d} |x_i|^p \right)^{\frac{1}{p}} \quad (2.3)$$

$$\text{최대 놈: } \|\mathbf{x}\|_\infty = \max(|x_1|, |x_2|, \dots, |x_d|) \quad (2.4)$$

- 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 놈

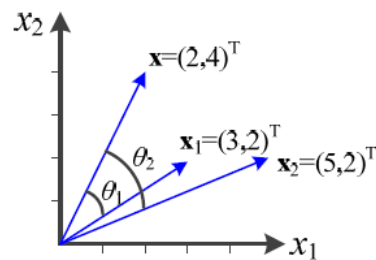
$$\text{프로베니우스 놈: } \|\mathbf{A}\|_F = \left(\sum_{i=1,n} \sum_{j=1,m} a_{ij}^2 \right)^{\frac{1}{2}} \quad (2.6)$$

$$\text{예를 들어, } \left\| \begin{pmatrix} 2 & 1 \\ 6 & 4 \end{pmatrix} \right\|_F = \sqrt{2^2 + 1^2 + 6^2 + 4^2} = 7.550$$

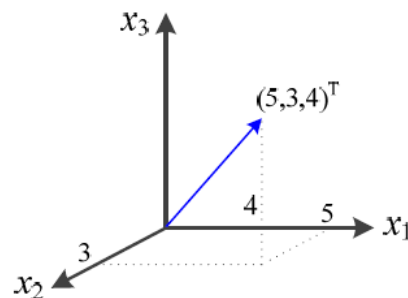
2.1.2 놀과 유사도

■ 유사도와 거리

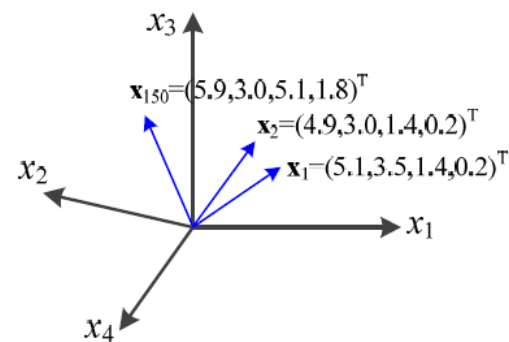
- 벡터를 기하학적으로 해석



(a) 2차원 벡터



(b) 3차원 벡터



(c) 4차원 벡터(Iris 데이터)

그림 2-2 벡터를 기하학적으로 해석

- 코사인 유사도

$$\text{cosine_similarity}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a}}{\|\mathbf{a}\|} \cdot \frac{\mathbf{b}}{\|\mathbf{b}\|} = \cos(\theta) \quad (2.7)$$

2.1.3 퍼셉트론의 해석

■ 퍼셉트론

- 1958년 로젠블랫이 고안한 분류기 모델

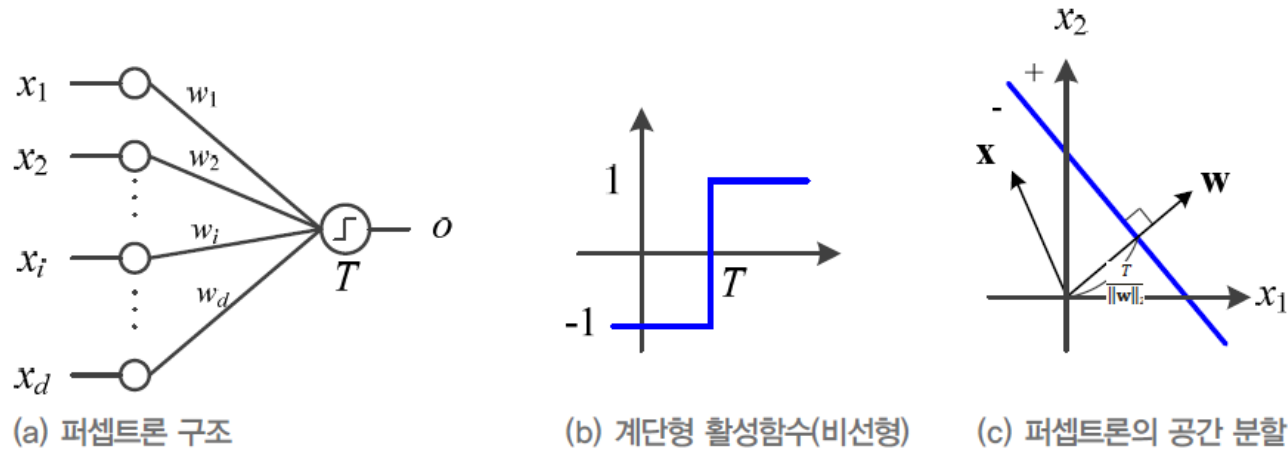


그림 2-3 퍼셉트론의 구조와 동작

- 퍼셉트론의 동작을 수식으로 표현하면,

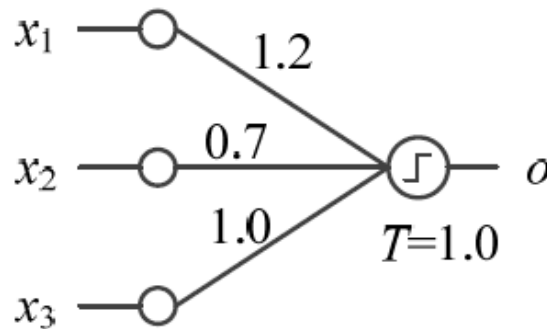
$$o = \tau(\mathbf{w} \cdot \mathbf{x}), \quad \text{이때} \quad \tau(a) = \begin{cases} 1, & a \geq T \\ -1, & a < T \end{cases} \quad (2.8)$$

- 활성 함수 τ 로는 계단함수 사용

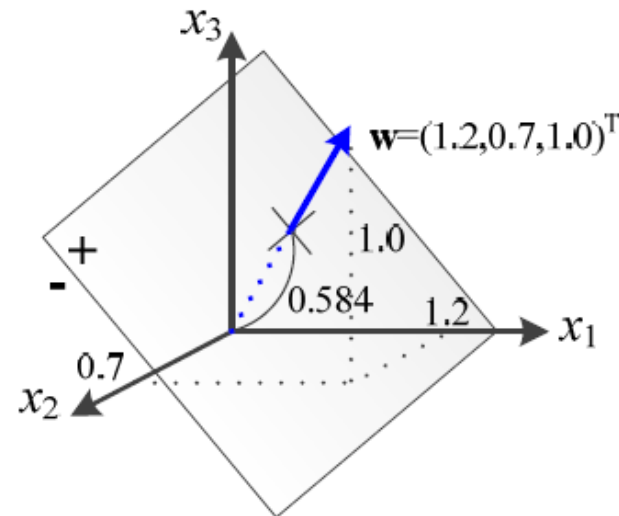
2.1.3 퍼셉트론의 해석

■ 퍼셉트론

- [그림 2-3(c)]의 파란 직선은 두 개의 부분공간을 나누는 결정직선decision line
 - \mathbf{w} 에 수직이고 $\frac{T}{\|\mathbf{w}\|_2}$ 만큼 떨어져 있음
- 3차원 특징공간은 결정평면decision plane, 4차원 이상은 결정 초평면decision hyperplane
- 예) 3차원 특징공간을 위한 퍼셉트론



(a) 퍼셉트론

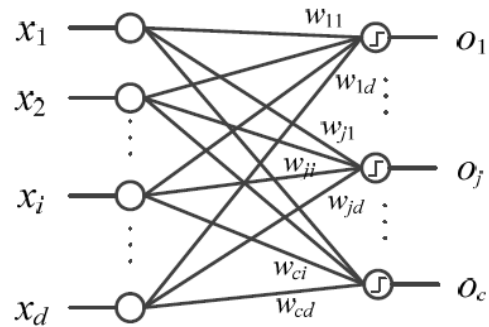


(b) 공간 분할(2부류 분류)

그림 2-4 퍼셉트론의 예(3차원)

2.1.3 퍼셉트론의 해석

■ 출력이 여러 개인 퍼셉트론



출력은 벡터 $\mathbf{o} = (o_1, o_2, \dots, o_c)^T$ 로 표기

j 번째 퍼셉트론의 가중치 벡터를

$\mathbf{w}_j = (w_{j1}, w_{j2}, \dots, w_{jd})^T$ 와 같이 표기

그림 2-5 출력이 여러 개인 퍼셉트론

- 동작을 수식으로 표현하면,

$$\mathbf{o} = \tau \begin{pmatrix} \mathbf{w}_1 \cdot \mathbf{x} \\ \mathbf{w}_2 \cdot \mathbf{x} \\ \vdots \\ \mathbf{w}_c \cdot \mathbf{x} \end{pmatrix}$$

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

$$\text{이때 } \mathbf{W} = \begin{pmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_c^T \end{pmatrix}$$

- 가중치 벡터를 각 부류의 기준 벡터로 간주하면, c 개 부류의 유사도를 계산하는 셈

2.1.3 퍼셉트론의 해석

■ 학습의 정의

- 식 (2.10)은 학습을 마친 프로그램을 현장에 설치했을 때 일어나는 과정

분류라는 과정: $\overset{?}{\tilde{\mathbf{o}}} = \tau(\overset{\text{앞}}{\tilde{\mathbf{W}}} \overset{\text{앞}}{\tilde{\mathbf{x}}})$ (2.10)

- 식 (2.11)은 학습 과정

- 학습은 훈련집합의 샘플에 대해 식 (2.11)을 가장 잘 만족하는 \mathbf{w} 를 찾아내는 작업

학습이라는 과정: $\overset{\text{앞}}{\tilde{\mathbf{o}}} = \tau(\overset{?}{\tilde{\mathbf{W}}} \overset{\text{앞}}{\tilde{\mathbf{x}}})$ (2.11)

■ 현대 기계 학습에서 퍼셉트론의 중요성

- 딥러닝은 퍼셉트론을 여러 층으로 확장하여 만들

2.1.4 선형결합과 벡터공간

■ 벡터

- 공간상의 한 점으로 화살표 끝이 벡터의 좌표에 해당

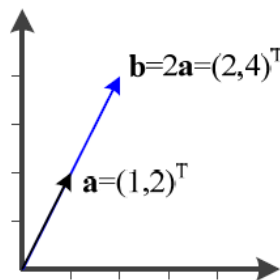
■ 선형결합이 만드는 벡터공간

- 기저벡터 \mathbf{a} 와 \mathbf{b} 의 선형결합

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

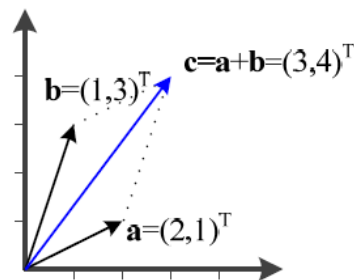
(2.12)

- 선형결합으로 만들어지는 공간을 **벡터공간**이라 부름

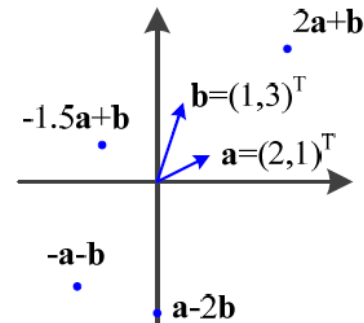


(a) 벡터에 스칼라 곱

그림 2-6 벡터의 연산

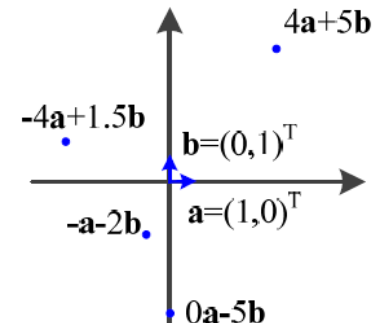


(b) 두 벡터의 덧셈



(a) 기저 벡터와 벡터공간

그림 2-7 벡터공간



(b) 정규직교 기저 벡터

2.1.5 역행렬

■ 역행렬의 원리

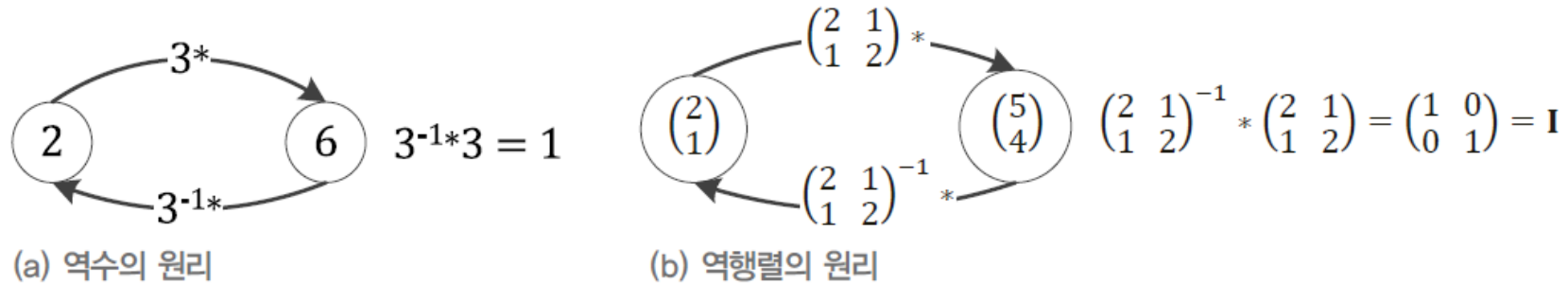


그림 2-9 역행렬

- 정사각행렬 A 의 역행렬 A^{-1}

$$A^{-1}A = AA^{-1} = I$$

- 예를 들어, $\begin{pmatrix} 2 & 1 \\ 6 & 4 \end{pmatrix}$ 의 역행렬은 $\begin{pmatrix} 2 & -0.5 \\ -3 & 1 \end{pmatrix}$

2.3.1 매개변수 공간의 탐색

■ 학습 모델의 매개변수 공간

- 높은 차원에 비해 훈련집합의 크기가 작아 참인 확률분포를 구하는 일은 불가능함
- 따라서 기계 학습은 적절한 모델을 선택하고, 목적함수를 정의하고, 모델의 매개변수 공간을 탐색하여 목적함수가 최저가 되는 최적점을 찾는 전략 사용 → 특징 공간에서 해야 하는 일을 모델의 매개변수 공간에서 하는 일로 대치한 셈
- [그림 2-22]는 여러 예제 (θ 는 매개변수, $J(\theta)$ 는 목적함수)

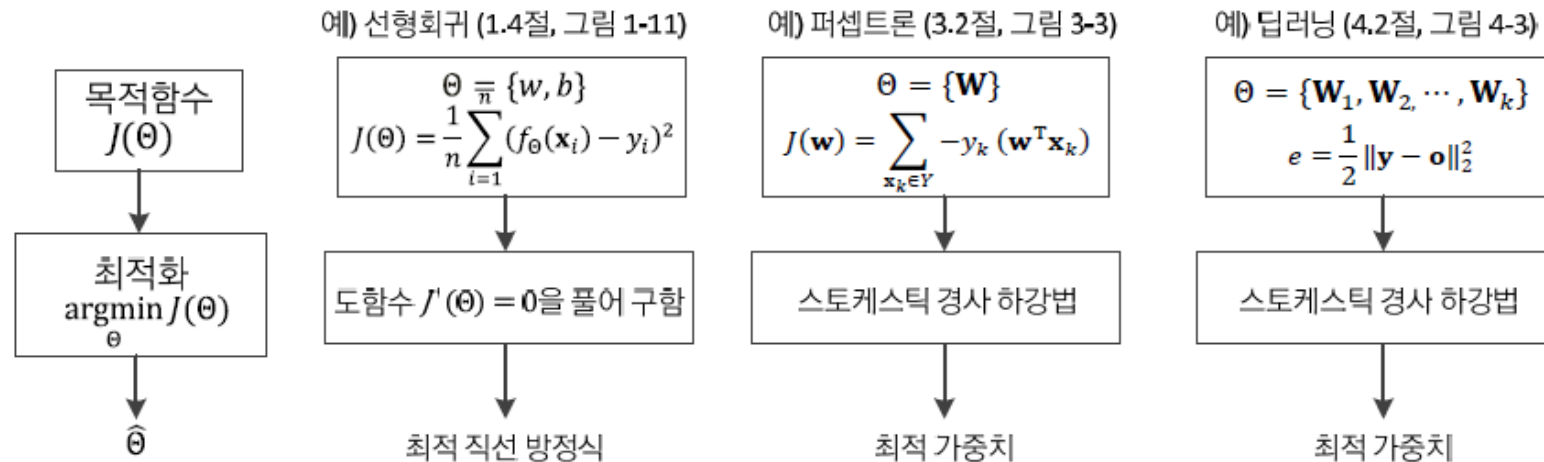


그림 2-22 최적화를 이용한 기계 학습의 문제풀이 과정

2.3.1 매개변수 공간의 탐색

■ 학습 모델의 매개변수 공간

- 특징 공간보다 수 배~수만 배 넓음
 - [그림 2-22]의 선형회귀에서는 특징 공간은 1차원, 매개변수 공간은 2차원
 - MNIST 인식하는 딥러닝 모델은 784차원 특징 공간, 수십만~수백만 차원의 매개변수 공간
- [그림 2-23] 개념도의 매개변수 공간: \hat{x} 은 전역 최적해, x_2 와 x_4 는 지역 최적해
- x_2 와 같이 전역 최적해에 가까운 지역 최적해를 찾고 만족하는 경우 많음

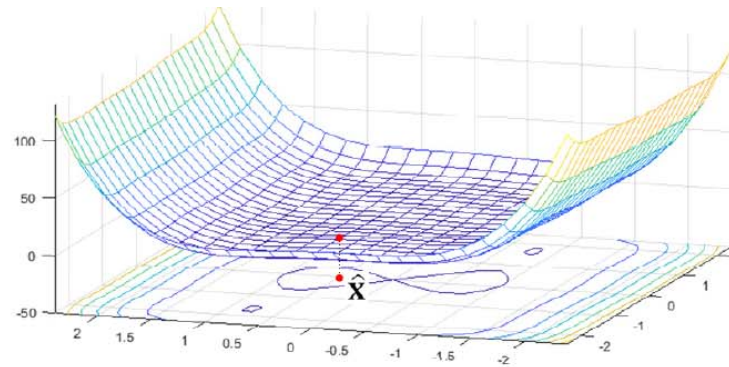
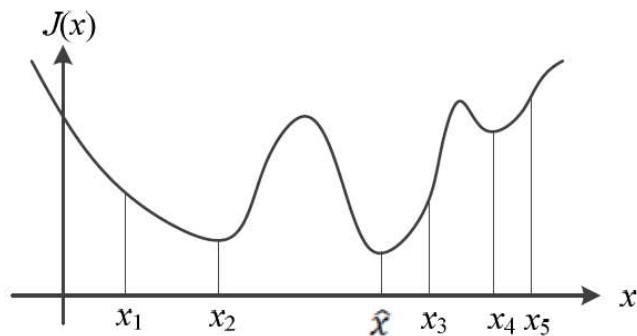


그림 2-23 최적해 탐색

■ 기계 학습이 해야 할 일을 식으로 정의하면,

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

2.3.1 매개변수 공간의 탐색

■ 최적화 문제 해결

- 낱낱탐색 exhaustive search 알고리즘
 - 차원이 조금만 높아져도 적용 불가능
 - 예) 4차원 Iris에서 각 차원을 1000 구간으로 나눈다면 총 1000^4 개의 점을 평가해야 함
- 무작위 탐색 알고리즘
 - 아무 전략이 없는 순진한 알고리즘

알고리즘 2-1 낱낱탐색 알고리즘

입력: 훈련집합 \mathbb{X} 와 \mathbb{Y}

출력: 최적해 $\hat{\theta}$

```
1 가능한 해를 모두 생성하여 집합  $S$ 에 저장한다.
2  $min$ 을 충분히 큰 값으로 초기화한다.
3 for ( $S$ 에 속하는 각 점  $\theta_{current}$ 에 대해)
4     if( $J(\theta_{current}) < min$ )  $min = J(\theta_{current})$ ,  $\theta_{best} = \theta_{current}$ 
5  $\hat{\theta} = \theta_{best}$ 
```

알고리즘 2-2 무작위 탐색 알고리즘

입력: 훈련집합 \mathbb{X} 와 \mathbb{Y}

출력: 최적해 $\hat{\theta}$

```
1  $min$ 을 충분히 큰 값으로 초기화한다.
2 repeat
3     무작위로 해를 하나 생성하고  $\theta_{current}$ 라 한다.
4     if( $J(\theta_{current}) < min$ )  $min = J(\theta_{current})$ ,  $\theta_{best} = \theta_{current}$ 
5 until(멈춤 조건)
6  $\hat{\theta} = \theta_{best}$ 
```

2.3.1 매개변수 공간의 탐색

- [알고리즘 2-3]은 기계 학습이 사용하는 전형적인 알고리즘
 - 라인 3에서는 목적함수가 작아지는 방향을 주로 미분으로 찾아냄

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

입력: 훈련집합 \mathbb{X} 와 \mathbb{Y}

출력: 최적해 $\hat{\theta}$

```
1  난수를 생성하여 초기해  $\theta$ 을 설정한다.  
2  repeat  
3       $J(\theta)$ 가 작아지는 방향  $d\theta$ 를 구한다.  
4       $\theta = \theta + d\theta$   
5  until(멈춤 조건)  
6   $\hat{\theta} = \theta$ 
```

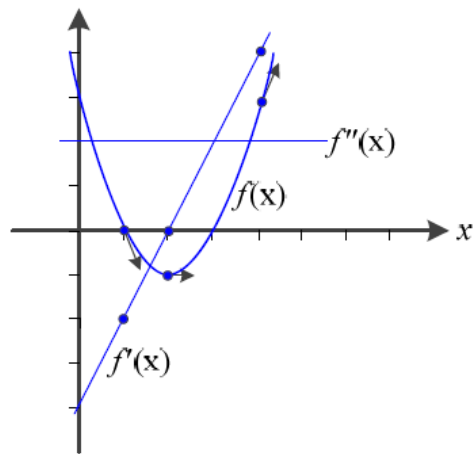
2.3.2 미분

■ 미분에 의한 최적화

■ 미분의 정의

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}, \quad f''(x) = \lim_{\Delta x \rightarrow 0} \frac{f'(x + \Delta x) - f'(x)}{\Delta x} \quad (2.51)$$

- 1차 도함수 $f'(x)$ 는 함수의 기울기, 즉 값이 커지는 방향을 지시함
- 따라서 $-f'(x)$ 방향에 목적함수의 최저점이 존재
- [알고리즘 2-3]에서 **d0**로 $-f'(x)$ 를 사용함 ← 경사 하강 알고리즘의 핵심 원리



$$y = f(x) = x^2 - 4x + 3$$

$$y' = f'(x) = 2x - 4$$

그림 2-24 간단한 미분 예제

2.3.2 미분

■ 편미분

- 변수가 여러 개인 함수의 미분
- 미분값이 이루는 벡터를 **그레디언트**라 부름
- 여러 가지 표기: $\nabla f, \frac{\partial f}{\partial \mathbf{x}}, \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)^T$
- 예)

$$\left. \begin{aligned} f(\mathbf{x}) &= f(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \\ \nabla f = f'(\mathbf{x}) &= \frac{\partial f}{\partial \mathbf{x}} = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)^T = (2x_1^5 - 8.4x_1^3 + 8x_1 + x_2, 16x_2^3 - 8x_2 + x_1)^T \end{aligned} \right\} \quad (2.52)$$

■ 기계 학습에서 편미분

- 매개변수 집합 Θ 에 많은 변수가 있으므로 편미분을 많이 사용

2.3.2 미분

■ 편미분으로 얻은 그레이디언트에 따라 최저점을 찾아가는 예제

예제 2-10

초기점 $\mathbf{x}_0 = (-0.5, 0.5)^T$ 라고 하자. \mathbf{x}_0 에서의 그레이디언트는 $f'(\mathbf{x}_0) = (-2.5125, -2.5)^T$ 즉, $\nabla f|_{\mathbf{x}_0} = (-2.5125, -2.5)^T$ 이다. [그림 2-25]는 \mathbf{x}_0 에서 그레이디언트를 화살표로 표시하고 있어, $-f'(\mathbf{x}_0)$ 은 최저점의 방향을 제대로 가리키는 것을 확인할 수 있다. 하지만 얼마만큼 이동하여 다음 점 \mathbf{x}_1 로 옮겨갈지에 대한 방안은 아직 없다. 2.3.3절에서 공부하는 경사 하강법은 이에 대한 답을 제공한다.

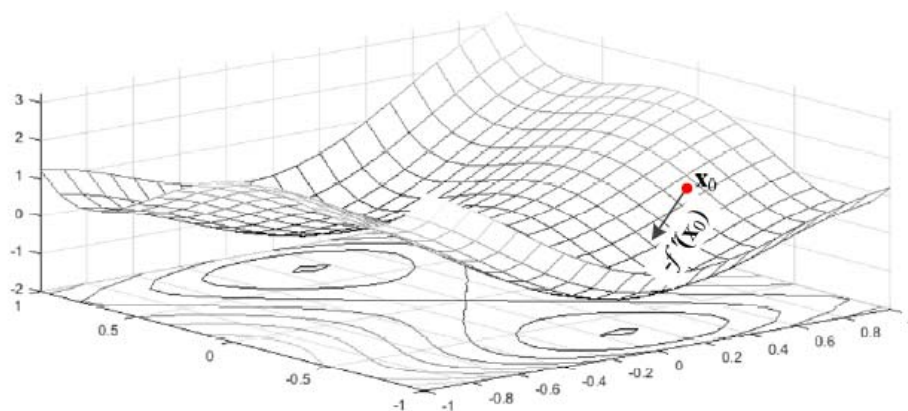


그림 2-25 그레이디언트는 최저점으로 가는 방향을 알려 줌

2.3.2 미분

■ 연쇄법칙

- 합성함수 $f(x) = g(h(x))$ 의 미분

$$\left. \begin{aligned} f'(x) &= g'(h(x))h'(x) \\ f'(x) &= g'(h(i(x)))h'(i(x))i'(x) \end{aligned} \right\} \quad (2.53)$$

- 예) $f(x) = 3(2x^2 - 1)^2 - 2(2x^2 - 1) + 5$ 일 때 $h(x) = 2x^2 - 1$ 로 두면,

$$f'(x) = \underbrace{(3 * 2(2x^2 - 1) - 2)}_{g'(h(x))} \underbrace{(2 * 2x)}_{h'(x)} = 48x^3 - 32x$$

■ 다층 퍼셉트론은 합성함수

- $\frac{\partial o_i}{\partial u_{23}^1}$ 를 계산할 때 연쇄법칙 적용
- 3.4절(오류 역전파)에서 설명

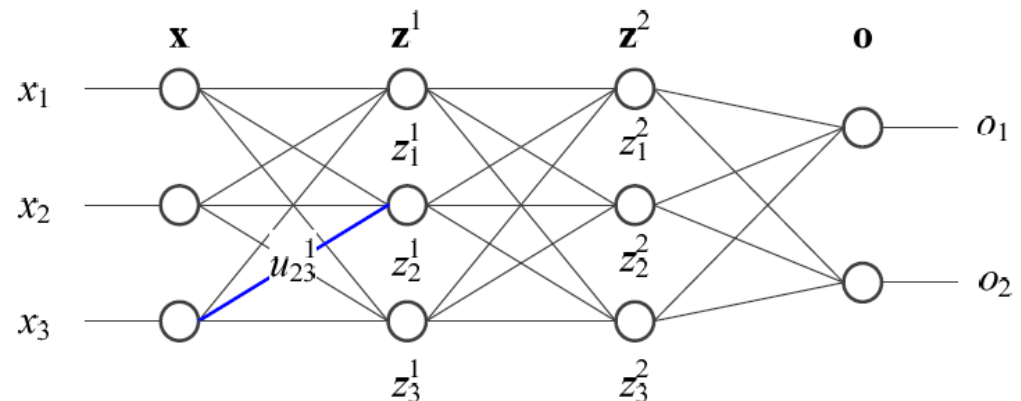


그림 2-26 다층 퍼셉트론은 합성함수

2.3.2 미분

■ 야코비안 행렬

- 함수 $\mathbf{f}: \mathbb{R}^d \mapsto \mathbb{R}^m$ 을 미분하여 얻은 행렬

$$\text{야코비안 행렬 } \mathbf{J} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_d} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_d} \end{pmatrix}$$

예)

$$\mathbf{f}: \mathbb{R}^2 \mapsto \mathbb{R}^3 \text{ 인 } \mathbf{f}(\mathbf{x}) = (2x_1 + x_2^2, -x_1^2 + 3x_2, 4x_1x_2)^T$$

$$\mathbf{J} = \begin{pmatrix} 2 & 2x_2 \\ -2x_1 & 3 \\ 4x_2 & 4x_1 \end{pmatrix} \quad \mathbf{J}|_{(2,1)^T} = \begin{pmatrix} 2 & 2 \\ -4 & 3 \\ 4 & 8 \end{pmatrix}$$

■ 헤시안 행렬

- 2차 편도함수

$$\text{헤시안 행렬 } \mathbf{H} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 x_1} & \frac{\partial^2 f}{\partial x_1 x_2} & \dots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2 x_2} & \dots & \frac{\partial^2 f}{\partial x_2 x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \frac{\partial^2 f}{\partial x_n x_2} & \dots & \frac{\partial^2 f}{\partial x_n x_n} \end{pmatrix}$$

예)

$$\begin{aligned} f(\mathbf{x}) &= f(x_1, x_2) \\ &= \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \end{aligned}$$

$$\mathbf{H} = \begin{pmatrix} 10x_1^4 - 25.2x_1^2 + 8 & 1 \\ 1 & 48x_2^2 - 8 \end{pmatrix}$$

$$\mathbf{H}|_{(0,1)^T} = \begin{pmatrix} 8 & 1 \\ 1 & 40 \end{pmatrix}$$

2.3.3 경사 하강 알고리즘

- 식 (2.58)은 경사 하강법이 낮은 곳을 찾아가는 원리

- $\mathbf{g} = d\boldsymbol{\theta} = \frac{\partial J}{\partial \boldsymbol{\theta}}$ 이고, ρ 는 학습률

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \rho \mathbf{g}$$

(2.58)

- 배치 경사 하강 알고리즘

- 샘플의 그래디언트를 평균한 후 한꺼번에 갱신

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  $\boldsymbol{\theta}$ 를 설정한다.
2  repeat
3       $\mathbb{X}$ 에 있는 샘플의 그래디언트  $\nabla_1, \nabla_2, \dots, \nabla_n$ 을 계산한다.
4       $\nabla_{total} = \frac{1}{n} \sum_{i=1, n} \nabla_i$  // 그래디언트 평균을 계산
5       $\boldsymbol{\theta} = \boldsymbol{\theta} - \rho \nabla_{total}$ 
6  until(멈춤 조건)
7   $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$ 
```

훈련집합

$$\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$$

$$\mathbb{Y} = \{y_1, y_2, \dots, y_n\}$$

2.3.3 경사 하강 알고리즘

■ 스토캐스틱 경사 하강 SGD(stochastic gradient descent) 알고리즘

- 한 샘플의 그레이디언트를 계산한 후 즉시 갱신
- 라인 3~6을 한 번 반복하는 일을 한 세대라 부름

알고리즘 2-5 스토캐스틱 경사 하강 알고리즘(SGD)

입력: 훈련집합 \mathbb{X} 와 \mathbb{Y} , 학습률 ρ

출력: 최적해 $\hat{\Theta}$

```
1  난수를 생성하여 초기해  $\Theta$ 를 설정한다.
2  repeat
3     $\mathbb{X}$ 의 샘플의 순서를 섞는다.
4    for ( $i=1$  to  $n$ )
5       $i$ 번째 샘플에 대한 그레이디언트  $\nabla_i$ 를 계산한다.
6       $\Theta = \Theta - \rho \nabla_i$ 
7  until(멈춤 조건)
8   $\hat{\Theta} = \Theta$ 
```

- 다른 방식의 구현

```
3   $\mathbb{X}$ 에서 임의로 샘플 하나를 뽑는다.
4  뽑힌 샘플의 그레이디언트  $\nabla$ 를 계산한다.
5   $\Theta = \Theta - \rho \nabla$ 
```