기계학습

3강 선형회귀(2), 선형분류

장필훈 교수

학습목차

01

편향분산분해(2)

1 편향분산분해(2)

$$\{y(x;D) - h(x)\}^2$$

$$\begin{aligned}
&\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] + \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\
&\quad = \quad \{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2 + \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\
&\qquad\quad +2\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}.
\end{aligned}$$

1 편향분산분해(2)

$$2\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}$$

$$E[2(y - E_D[y])(E_D[y] - h(x))]$$

$$=2E[yE_D[y] - h(x)y - E_D[y]E_D[y] + E_D[y]h(x)]$$

$$=2E\big[yE[y]\big] - 2E[h(x)y] - 2E\big[E[y]E[y]\big] + 2E[E[y]h(x)]$$

$$=2E[y]E[y] - 2E[h(x)]E[y] - 2E[y]E[y] + 2E[y]E[h(x)]$$

$$=0$$

1 편향분산분해(2)

- 정리하면,

$$E_D[\{y(x;D) - h(x)\}^2] =$$

$$\{E_D[y(x;D)] - h(x)\}^2 + E_D[\{y(x;D) - E_D[y(x;D)]\}^2]$$

편향^2 분산

1 편향분산분해(2)

- 편향과 분산

편향: 전체 데이터집합에 대한 평균예측과 회귀함수의 차

분산: 각각 데이터집합에서 구한 해와 전체평균의 차

데이터 집합에 따른 함수y의 민감도를 나타냄.

2강의 기대제곱오류에 편향, 분산을 넣으면,

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t$$

1 편향분산분해(2)

$$\text{기대오류} = \text{편향}^2 + \text{분산} + \text{노이즈}$$

편향과 분산 사이의 트레이드 오프:

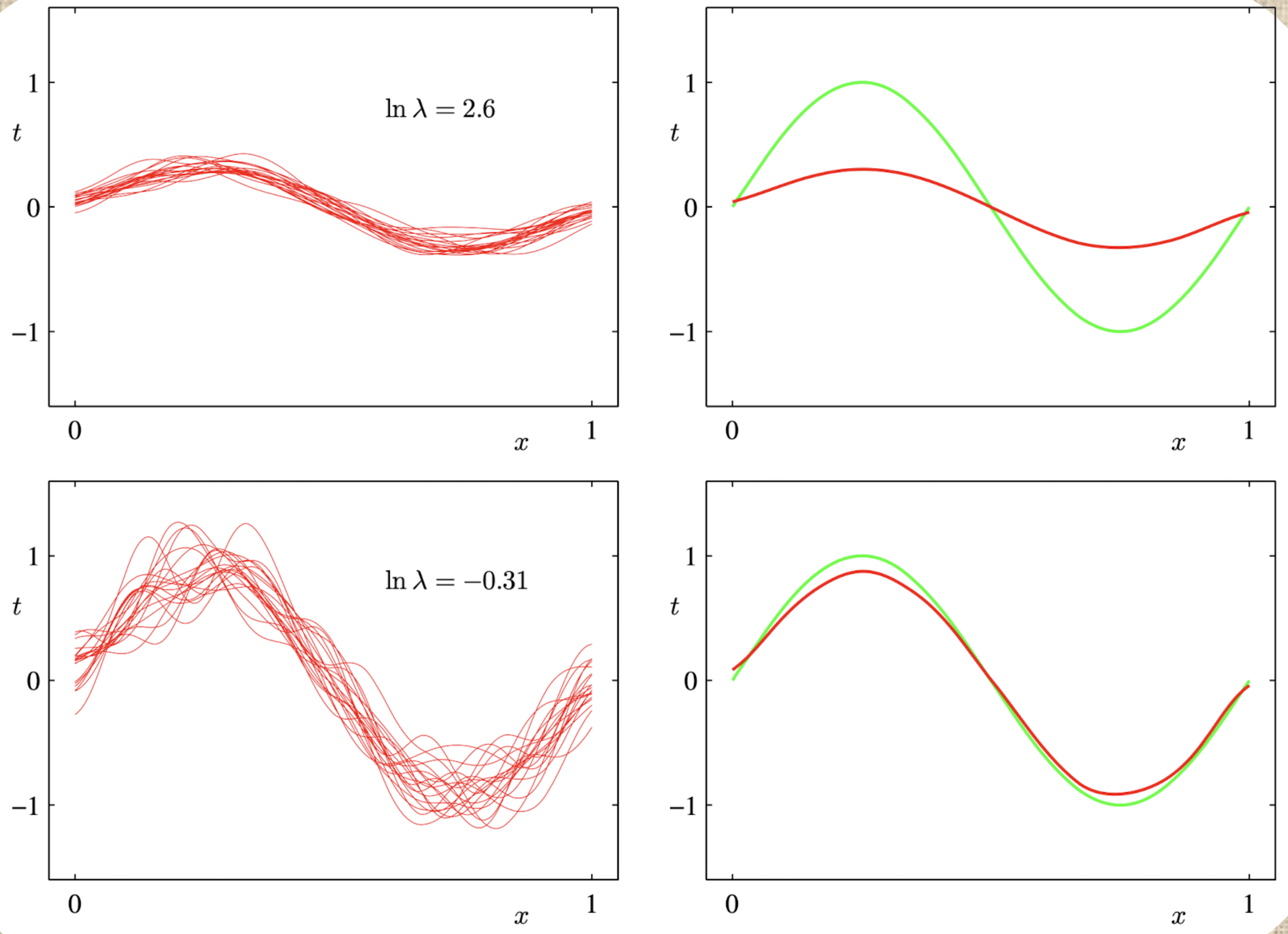
유연한 모델은 낮은 편향과 높은 분산값

엄격한 모델은 반대.

→ 둘 사이에 밸런스가 좋아야 좋은 모델이다.

여러 데이터집합들을 대상으로 한 결과. 적용이 쉽지 않음.

1 편향분산분해(2)



Bishop, Christopher M. Pattern Recognition and Machine Learning. New York :Springer, 2006. p.170

02

베이지안선형회귀

2 베이지안선형회귀

$$y(x, w) = w_0 + w_1 x$$

- 관찰 포인트
  - 데이터 집합의 크기에 따른 베이지안 학습의 결과
  - 사후분포가 새로운 데이터포인트 관측 후
    새로운 사전분포가 되는것

2 베이지안선형회귀

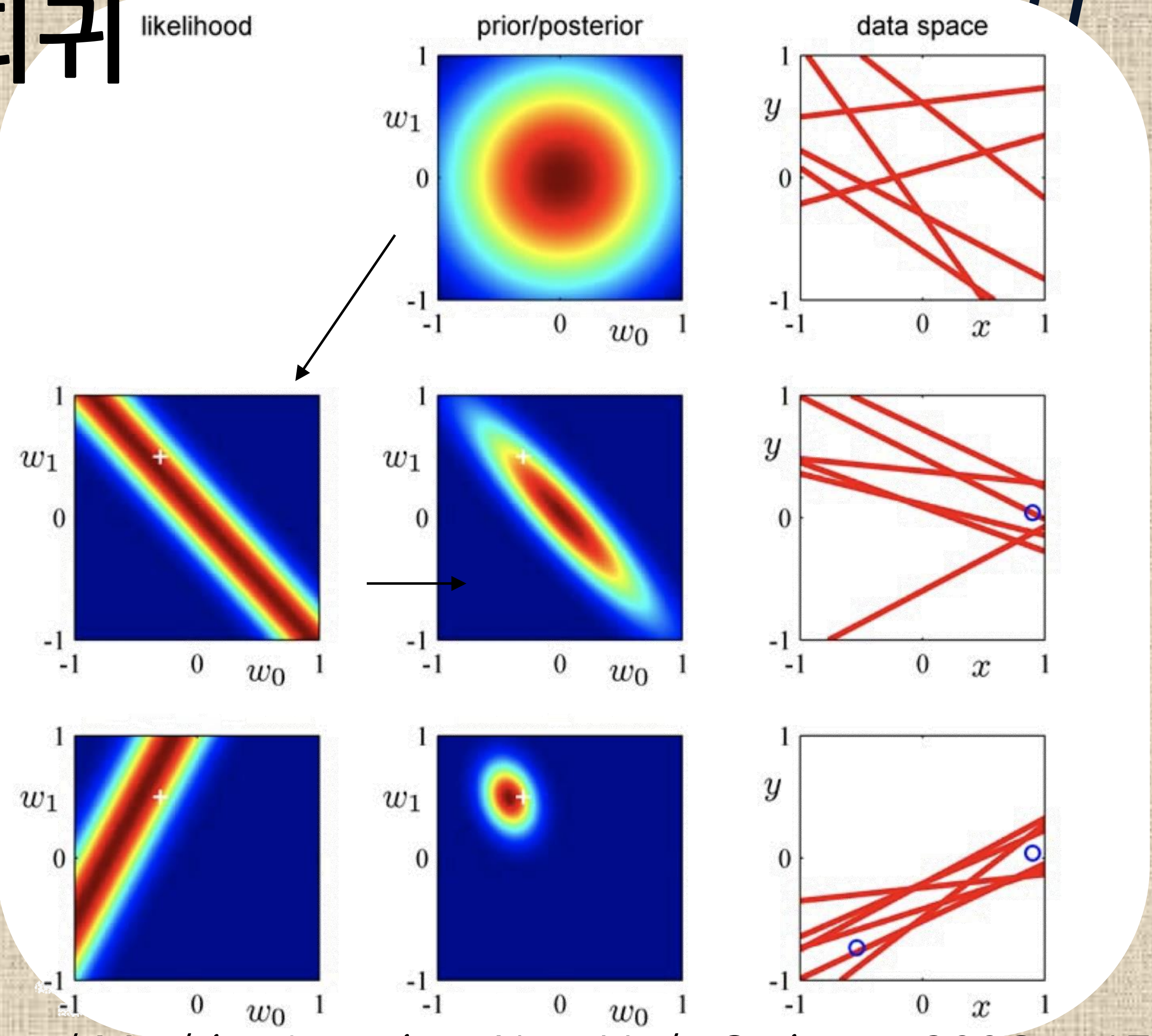
- 가운데가 사전분포,

왼쪽이 가능도 함수,

가운데는 사후분포의 정규화

(반복)

사후분포가 점점 모인다.



Bishop, Christopher M. Pattern Recognition and Machine Learning. New York :Springer, 2006. p.175

03

etc.

3-1 차원의 저주

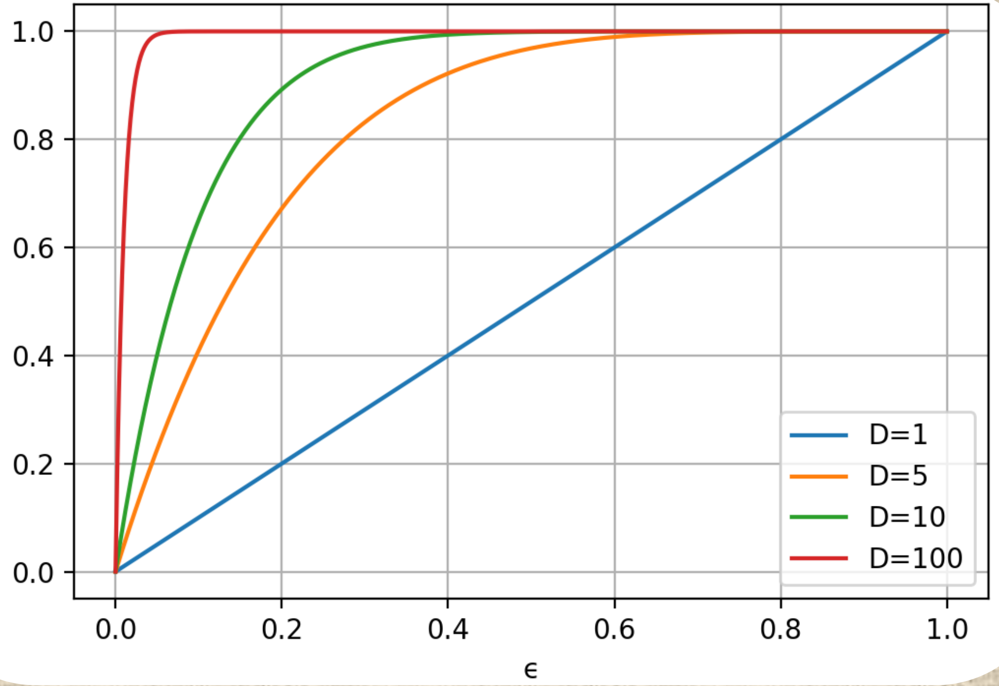
- D차원의 반지름 $r = 1$인 구를 가정하자.
- 반지름 $r = 1 - \epsilon$에서 $r = 1$사이에 존재하는 부피의 비율은?
- 반지름 r을 가진 구의 부피는 r^D에 비례하여 증가하므로

$$V_D(r) = K_D r^D$$

로 적을 수 있다.

3-1 차원의 저주

$$\frac{V_D(1) - V_D(1-\epsilon)}{V_D(1)} = \frac{K_D - K_D(1-\epsilon)^D}{K_D} = 1 - (1-\epsilon)^D$$



3-2 모델선택

- 집합을 셋으로 나눈다 : 훈련, 검증, 시험
- 모델을 비교하는 기준: 독립된 데이터를 사용하여 모델평가

  → 어떤 모델이 오류함수를 최소화하는가

  (홀드아웃 방법)

  - 이 자체가 과적합을 만들 수 있음 : 성능평가는 test set

    → 성능평가는 단 한번!

3-2 모델선택

- 앞의 곡선 피팅 문제에 적용하면?
- Competition할때도 셋으로 나눌까?
- 검증법은 여러가지가 가능
  - 예) k-fold cross-validation,

    Monte Carlo cross-validation

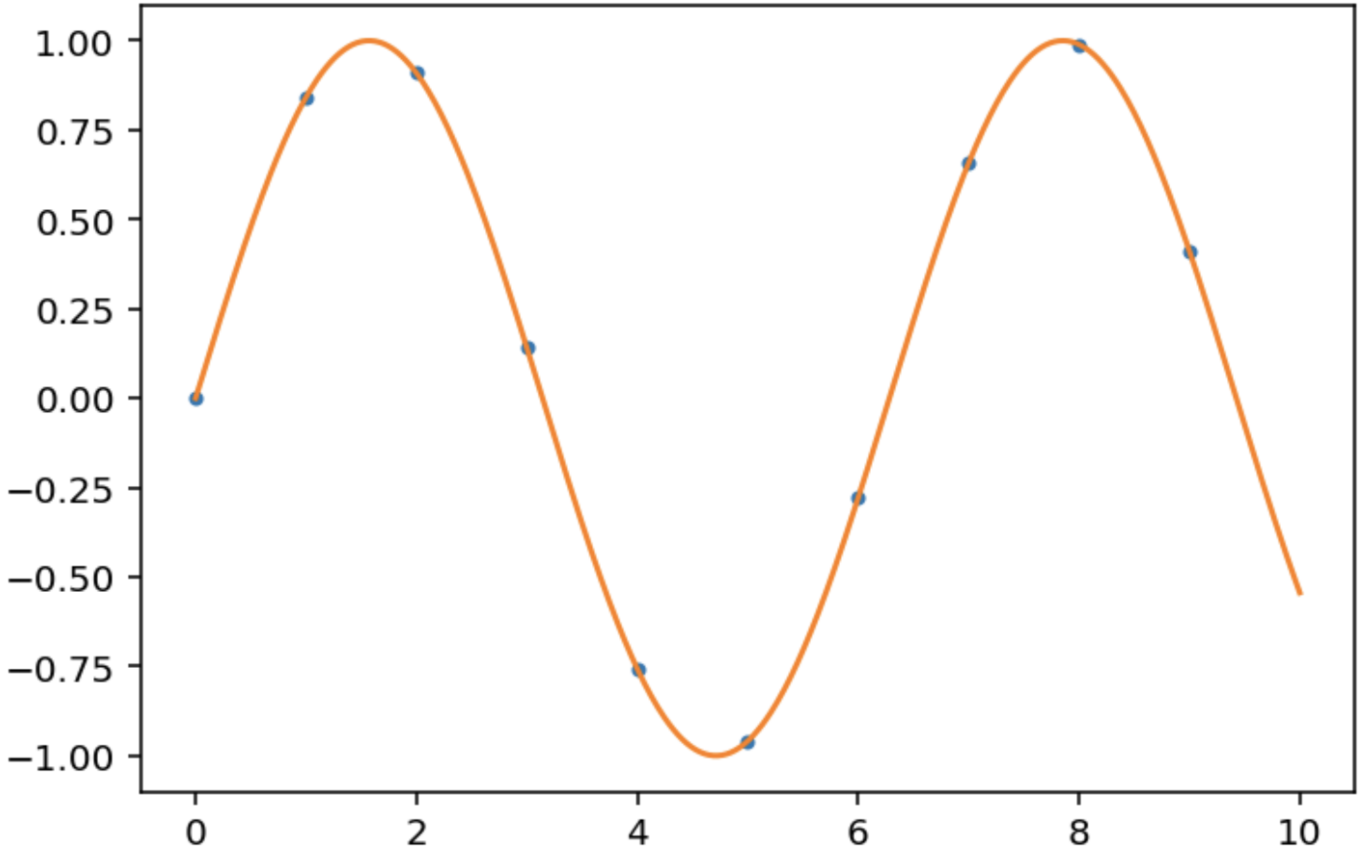
3-3 python

- 거의 모든 기초적인 알고리즘이 이미 구현되어 있음.
  - 상대적으로 사용하기 용이한 scikit learn
- 배운것들을 직접 실험해보는 것을 강력추천
  - 수업에서 자세히 다루지는 못함
- jupyter notebook/lab이 가장 많이 쓰임

3-3 python

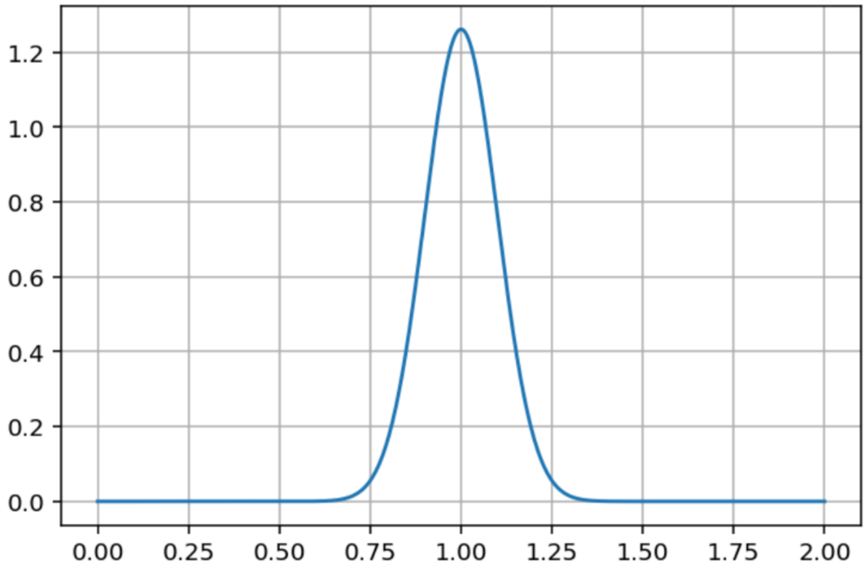
- anaconda
- jupyter notebook
- jupyter lab

```
x = np.linspace(0,10,500)
plt.plot(range(10), num10, '.', x, np.sin(x), '-')
plt.savefig('sol.png')
```



3-3 python

```
from matplotlib import pyplot as plt
%config InlineBackend.figure_format = 'retina'
x=np.linspace(0,2,1000)
ga=1/(np.sqrt(2*np.pi*0.1))*np.exp(-1/(2*0.1*0.1) * (x-1)*(x-1) )
plt.grid()
plt.plot(x,ga)
plt.savefig('gaussian1_0.1.jpg', dpi=300)
```



04

선형 분류(1)

4 선형분류

- 분류의 목표: (클래스) 할당
- 겹치지 않고, 하나에는 해당한다고 가정
  - 그 경계: 결정경계, 결정표면
  - 결정표면은 D차원 입력공간상의 D-1차원 초평면
- ‘선형분리 가능한 집합’

4 선형분류

- 타겟변수 t
  1. $t \in \{0,1\}$
  2. $0 \leq t \leq 1$, P값으로 해석
  3. etc
- $k > 0$의 경우는 one-hot encoding
  - $t = (0, 1, 0, 0, 0)^T$ or $t = (p_1, p_2, \ldots)^T$ or $\cdots$

4 선형분류

- 분류문제를 푸는 세가지 방법
  1. 결합밀도 p(x, t)를 구하고 p(t|x)를 알아낸 뒤,
     $y_t = E_t[t|x]$를 구함
  2. p(t|x)를 구하고 $y_t = E_t[t|x]$를 구함
  3. 판별함수 y_t(discriminant func.)를 구함

4 선형분류

- 방법1은 ‘생성모델’(generative model)이라고도 한다.
  1. 클래스별로 $p(x|C_k)$와 $p(C_k)$를 모두 구함
  2. 1을 이용해 $p(C_k|x)$를 알 수 있다.(∵베이즈정리)
  3. 입출력의 분포를 모델링했으므로,

     이 분포로부터 새로운 데이터를 만들어낼 수 있다.
- 방법 2는 $p(C_k|x)$를 바로 계산하는 ‘판별모델’

4 선형분류

- 0~1 사이의 값을 얻기 위해 함수 $f(\cdot)$를 다음과 같이 구성

$$y(\vec{x}) = f(\vec{w}^T\vec{x} + w_0)$$

$$0 \le y(x) \le 1$$

  - 이때 함수f를 activation function

    link function이라고도 함.

- 주의: 결정경계면은 f 와 무관하게 선형함수다.

4-1 판별함수

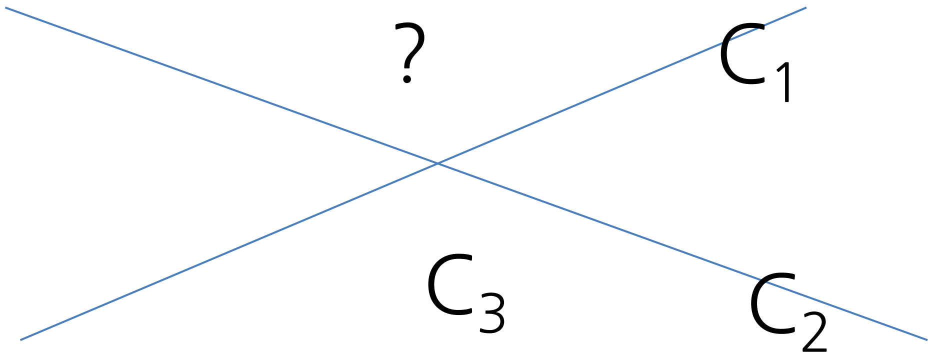
- K=2
  - $y(\vec{x}) = f(\vec{w}^T\vec{x} + w_0)$를 사용하여 $y(\boldsymbol{\vec{x}}) \geq 0$ 이면 C_1에, 아니면 C_2에 배정.
  - w_0(bias)는 조절 가능하므로 기준값이 0이라고 고정
  - 따라서 결정경계는 $y(\boldsymbol{\vec{x}}) = 0$인 hyperplane.
  - w_0가 위치 결정.

4-1 판별함수

- $x_0 = 1, \vec{w}' = (w_0, \vec{w}), \vec{x}' = (x_0, \vec{x})$으로 두면,

  $y(\vec{x}) = \vec{w}'^T \vec{x}'$(원점을 지나는 초평면)

- K>2 (다중클래스)
  - 2클래스 분류기를 K-1개 사용하면? 「일대다 분류기」

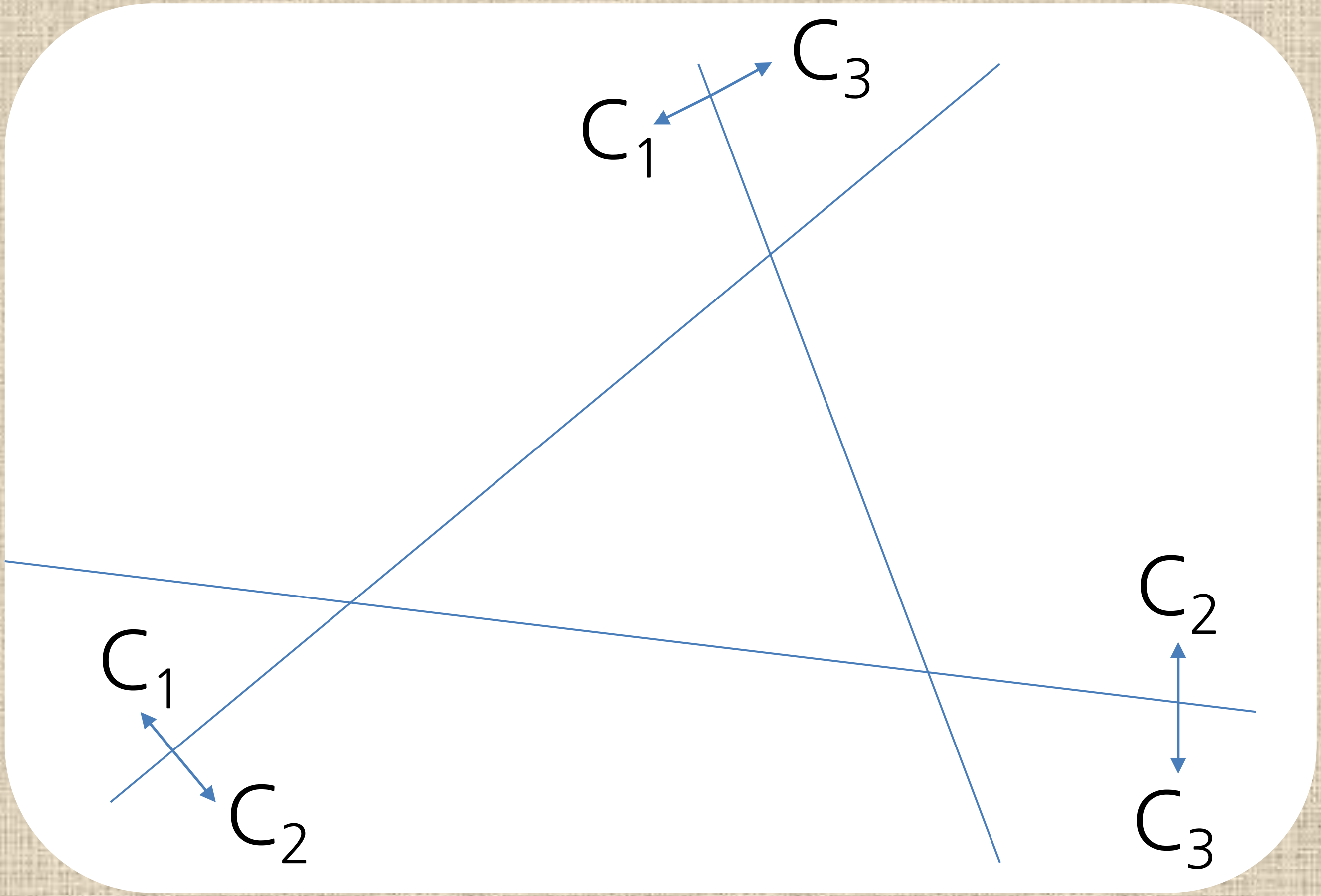


not C_1 & not C_2 → C_3

C_1 & C_2 → ?

4-1 판별함수

- K(K-1)/2 개의 모든 쌍에 대한 분류 : 일대일 분류기



4-1 판별함수

- 해결책

  K개의 선형함수로 이루어진 하나의 K클래스 판별함수

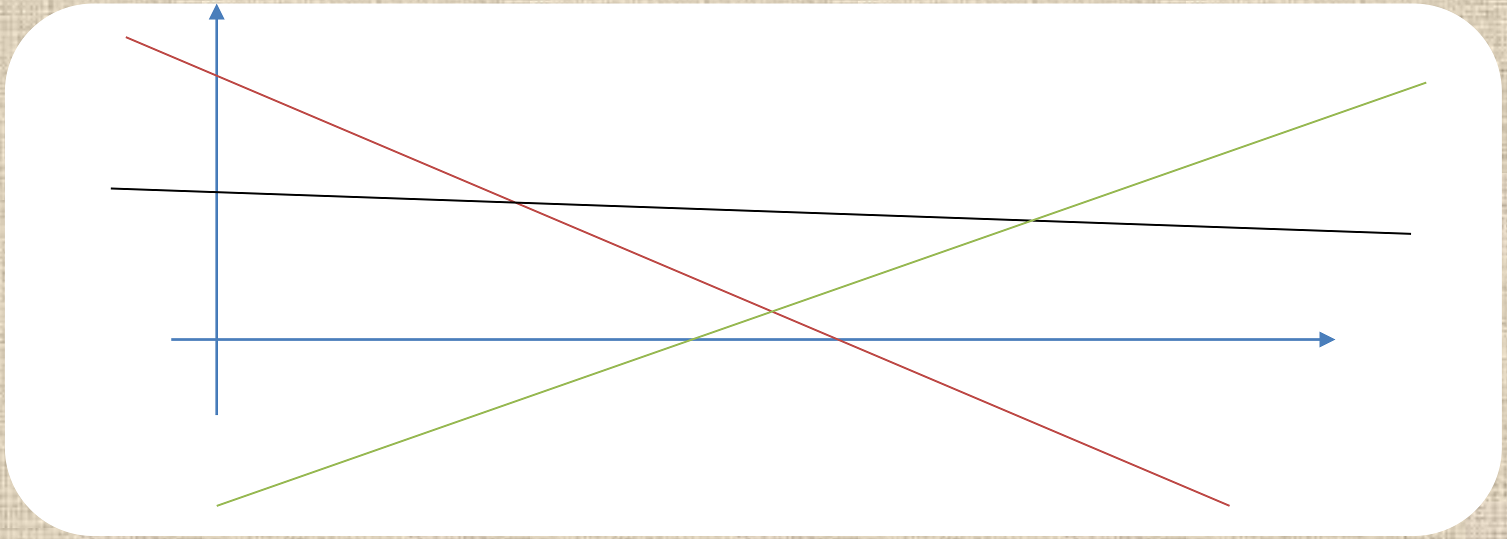
  $y_k(\vec{x}) = \vec{w}_k^T \vec{x} + w_{k0}$

  $j \neq k$인 모든 j에 대해 $y_k(\vec{x}) > y_j(\vec{x})$면 $\vec{x}$을 C_k에 배정

- C_k와 C_j사이의 결정경계: $y_k(\vec{x}) = y_j(\vec{x})$
  - 이 초평면은 $(\vec{w}_k - \vec{w}_j)^T \vec{x} + (w_{k0} - w_{j0}) = 0$

4-1 판별함수

- 개념적 설명



- 다차원에서도 이렇게 깔끔하게? 영역의 단일성은?

4-1 판별함수

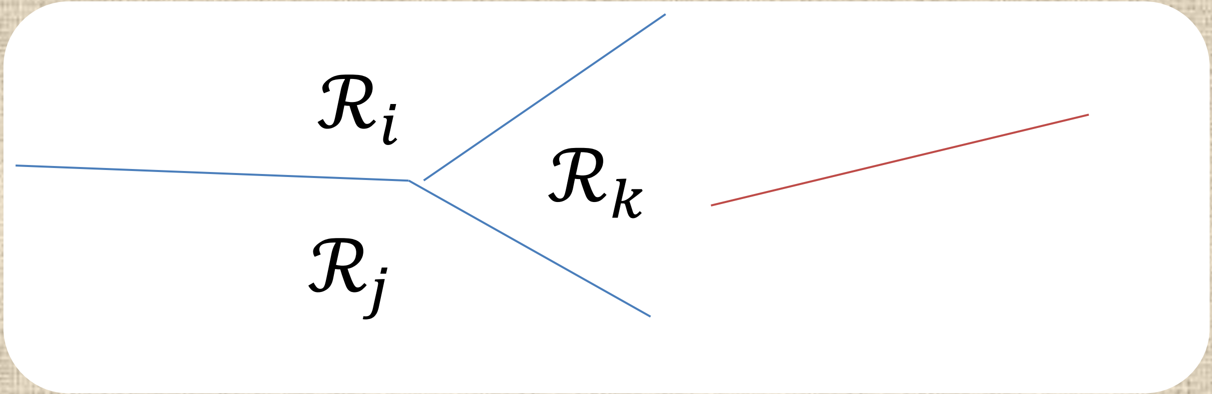
- 증명

두 점$(\vec{x}_A, \vec{x}_B)$이 한 영역$(\mathcal{R}_k)$에 속한다고 가정하면,

그 사이의 점은 $\hat{x} = \lambda\vec{x}_A + (1-\lambda)\vec{x}_B$로 표현되고, $(0 \le \lambda \le 1)$

$$y_k(\hat{x}) = \lambda y_k(\vec{x}_A) + (1-\lambda) y_k(\vec{x}_B)\text{이다.}$$

($\because$판별함수의 선형성)



4-1 판별함수

- 증명(cont.)

  $\vec{x}_A, \vec{x}_B$가 모두 $\mathcal{R}_k$에 속하므로, k가 아닌 모든 j에 대해

  $y_k(\vec{x}_A) > y_j(\vec{x}_A)$이고, $y_k(\vec{x}_B) > y_j(\vec{x}_B)$.

  따라서 $y_k(\vec{x}) > y_j(\vec{x})$이고, $\hat{x}$도 $\mathcal{R}_k$에 존재한다.

  그러므로 $\mathcal{R}_k$는 단일하게 연결되어 있고, 볼록하다.

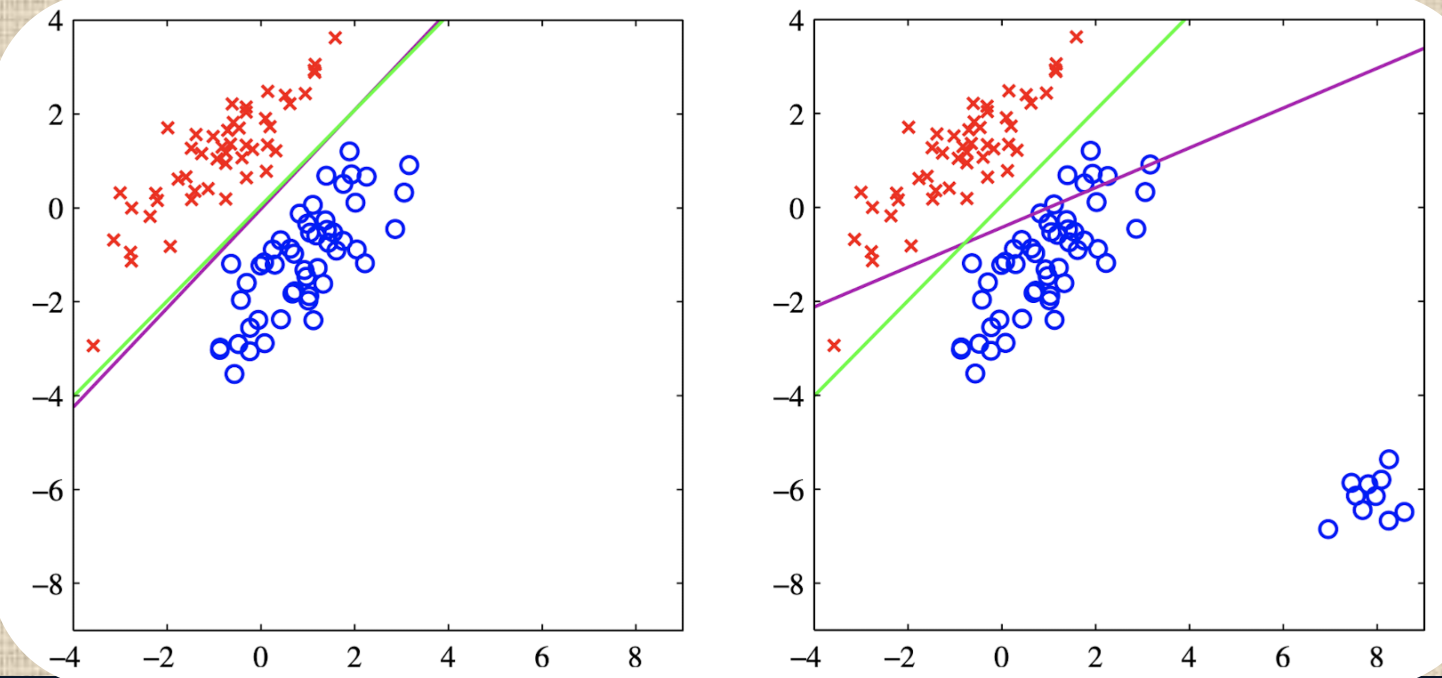
4-1 판별함수

- 선형판별함수의 매개변수를 학습하는 법

1. 최소제곱법
2. 피셔의 선형판별법
3. 퍼셉트론 알고리즘

4-2 최소제곱법

- 앞서나온 최소제곱법을 이용 : $y(\vec{x}) = \vec{w}'^T \vec{x}'$
  - outlier에 민감하다.



Bishop, Christopher M. Pattern Recognition and Machine Learning. New York :Springer, 2006. fig.4.4

4-3 피셔의 선형판별법

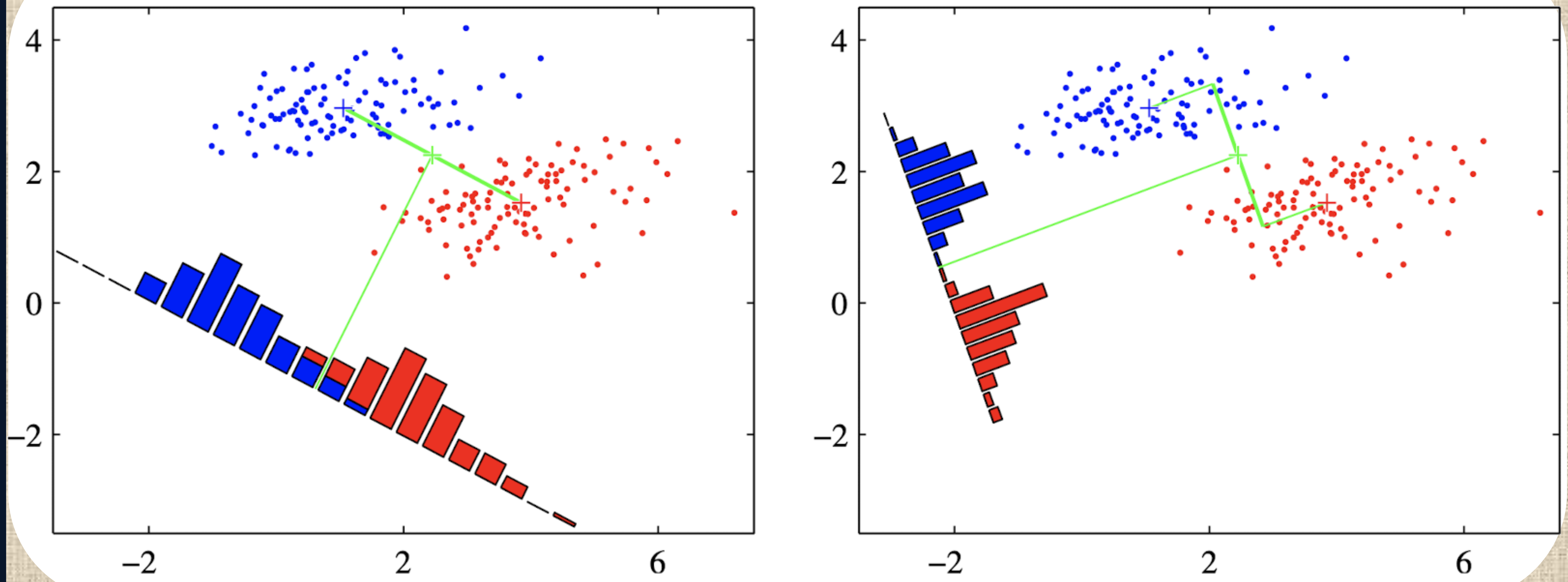
- 비확률적 방법
- 아이디어:

투영된 클래스의 평균사이 분리정도 최대화,

동시에 클래스 내 분산을 작게 한다. (피셔기준)

$$J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}, \qquad s_k^2 = \sum_{n \in C_k} (y_k - m_k)^2$$

4-3 피셔의 선형판별법



Bishop, Christopher M. Pattern Recognition and Machine Learning. New York :Springer, 2006. fig.4.6

4-3 피셔의 선형판별법

- 피셔기준은 정확한 계산이 가능하다

$$w \propto S_w^{-1}(m_2 - m_1)$$

$$\text{where } S_w = \sum_{n \in C_1}(x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_2}(x_n - m_2)(x_n - m_2)^T$$

- 전체 클래스 내(within) 공분산 행렬(S_w) 방향에만 의존한다.

다음시간

4강

- 퍼셉트론 알고리즘
- 확률적 생성/판별 모델
- 중심극한정리
- 신경망(1)