워크북

교과목명 : 머신 러닝

차시명: 3차시

◈ 담당교수: **장 필 훈**

⊙ 세부목차

- 편향분산분해(2)
- 베이지안 선형회귀
- 모델선택
- 차원의 저주
- python
- 선형분류(1)

학습에 앞서

■ 학습개요

패턴인식의 가장 기본적인 선형회귀를 마저 배우고 마무리한다. 이전 시간에 하던 편향분산분해를 식 유도과정과 함께 모두 이해하고 편형과 분산 각각에 대해 어떤 의미를 가지는지 이해한다.

선형회귀의 마무리로 베이지안 선형회귀의 과정을 대략 살펴본다.

선형회귀/분류와 직접적으로 관련있지는 않지만, 데이터과학의 기본적인 사항인 차원의 저주, 모델선택에 관해 배운다. 모델선택은 훈련/검증/시험집합 각각의 의미에 관해 배우고, 실제로 어떻게 사용되는지 이해한다.

선형회귀에 이어 선형분류에 관해 배운다. 선형분류는 회귀보다 실무에 더 자주 쓰인다. 생성모델, 판별모델에 관해 각각 자세히 배운다.

■ 학습목표

1	편향과 분산이 식에 어떻게 나타나는지 관찰하고 의미를 이해한다.
2	베이지안 선형회귀의 과정을 정성적으로 이해한다.
3	차원의 저주를 식으로 이해한다.
4	데이터를 훈련/검증/시험집합으로 분류하는 이유를 이해한다.
5	선형분류의 개념과 방법을 이해한다.

■ 주요용어

용어	해설
차원의 저주	고차원 입력공간에서 변수의 영향이 저차원의 직관적인 입력변수들의 영향과는 매우 다를 수 있다는 것을 비유적으로 나타낸 말. 저차원공간에서 발전시킨 아이디어는 고차원에 바로 적용하기 전에 충분한 검토가 필요하다.
훈련/검증/시험집합	모델을 선택할 때, 데이터를 보통 train, validation, test 이렇게 세 집합으로 나눈다. 데이터 전체를 학습(train)으로 사용하면 어떤 모델을 선택해야 할지 기준이 없게 되고, 두 개로만 나누어서 사용하게 되면 model 의 hyper-parameter 등을 조절하여 두 번째 집합에 적절하게 overfit 하게 되므로 모델을 공정하게 선택할 수 없다. 따라서, 훈련집합으로 학습 후 검증집합으로 성능을 평가하고 마지막으로 시험집합에 실제로 적용한다. 보통 시험집합은 주어지지 않거나 실제 서비스 자체를 지칭하므로 그것을 고려하지 않는다면 데이터를 둘로만 나눈다고 생각할 수 있다.
선형분류	선형모델을 사용하여 목표값을 예측하는 것이 아니라, 목표값을 분류하는 것. 몇종류로 분류하는가를 class 의 개수로 나타내며 여러개의 클래스로 분류하는 문제도 선형분류문제에 속한다.

	학습하기

<편향분산분해>

(이전시간에서 계속) 무한히 많은 데이터포인트를 가진 데이터 D의 분포를 추정해내고자 할때, 무한한 데이터를 모두 관찰할 수는 없으므로 여러번 데이터집합을 추출해서 y를 추정하는 과정을 반복합니다. 이때 제곱오류는 여러번 시행의 평균값으로 정의됩니다.

이것을 식으로 나타내면 다음과 같습니다.

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

여기서 $y(x;D)$는 데이터셋 D로부터 추정한 함수를 뜻합니다. 여러번 시행하여 평균을 보는 것이므로 E를 구합니다. 우선 E의 내부에 있는 제곱오류함수를 다음과같이 변형할 수 있습니다.

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

여기에 기댓값을 취하면, 교차항이 사라집니다. 교차항이 사라지는 것은 다음과 같이 확인해 볼 수 있습니다.

$$\begin{aligned}&E[2(y - E_D[y])(E_D[y] - h(x))] \\ &=2E[yE_D[y] - h(x)y - E_D[y]E_D[y] + E_D[y]h(x)] \\ &=2E[yE[y]] - 2E[h(x)y] - 2E[E[y]E[y]] + 2E[E[y]h(x)] \\ &=2E[y]E[y] - 2E[h(x)]E[y] - 2E[y]E[y] + 2E[y]E[h(x)] \\ &=0\end{aligned}$$

정리하면 다음과 같고, 여기서 첫번째 항이 편향의 제곱, 두번째 항이 분산입니다.

$$\begin{aligned}&E_D[\{y(x;D) - h(x)\}^2] = \\ &\{E_D[y(x;D)] - h(x)\}^2 + E_D[\{y(x;D) - E_D[y(x;D)]\}^2]\end{aligned}$$

말로 풀어 설명 하자면, 편향은 '전체 데이터 집합에 대한 평균예측과 회귀함수의 차'이고 분산은 '각각 데이터 집합에서 구한 해와 전체평균의 차이'입니다 즉 분산은 '데이터 집합에 따른 함수y의 민감도'를 나타냅니다.

여기에 노이즈까지 더해서, '기대오류 = 편향의 제곱 + 분산 + 노이즈'로 나타내기도 합니다.

편향과 분산 사이에는 트레이드오프 관계가 있습니다. 유연한 모델은 낮은 편향과 높은 분산값을 가지고 엄격한 모델은 그 반대입니다. 이 둘의 밸런스가 좋아야 좋은 모델입니다. 우리가 이처럼 무한한 데이터로부터 여러번 제곱오차를 구할 수 있으면 좋겠지만 실제로는 데이터의 개수가 절대적으로 부족할 때가 많아서 이정도도 할 수 없습니다.

<베이지안 선형회귀>

지금까지 배운 선형회귀를 베이지안 관점에서 접근하여 시도해 볼 수도 있습니다. 관찰의 포인트는 일단 데이터집합의 크기에 따른 베이지안 학습의 결과, 그리고 데이터 포인트 관측이 이루어질 때마다 이전의 사후분포가 새로운 사전분포가 되는 과정입니다. 녹화된 강의의 ppt중에 베이지안 선형회귀를 시각적으로 나타낸 그램을 끼워 넣었으니 참고 하세요. Bishop책 175쪽에 있는 그림입니다.

<차원의 저주>

회귀와 직접적인 연관이 있는 것은 아니지만 널리 알려진 개념 중 하나인 차원의 저주에 대해 배워보겠습니다.

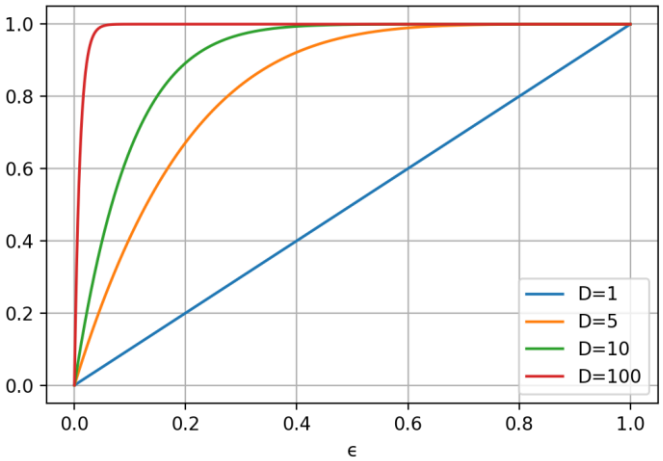
D차원의 반지름 1인 구를 가정합니다. 이때 반지름 r=1−ϵ에서 r=1사이에 존재하는 부피의 비율과 D사이의 관계는 어떻게 될까요. 반지름 r을 가진 구의 부피는 r의 D승에 비례하여 증가하므로 부피를 다음과 같이 나타낼 수 있습니다.

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

그러면 우리가 구하고자 하는 부피(1− ϵ과 1사이에 존재하는 부피)는 다음과 같습니다.

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

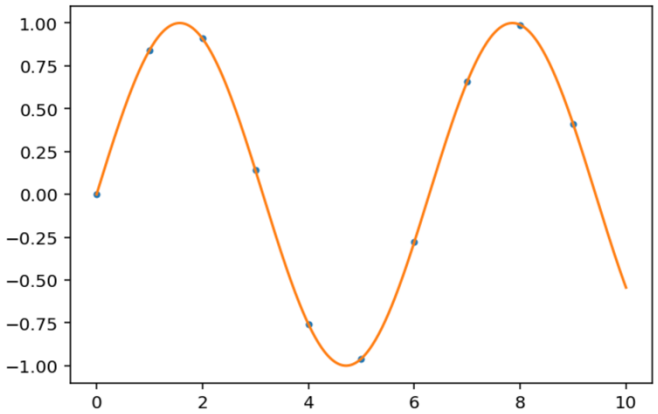
그림으로 나타내면 다음과 같습니다



그림은 파이썬을 이용해서 그린 것입니다 보통은 주피터 노트북으로 그리게 됩니다. 파이썬에는 scikit-learn이라는 좋은 라이브러리가 있기 때문에 우리가 수업시간에 배운 것들을 대부분 직접 해볼 수 있습니다. 꼭 익혀두시기 바랍니다.

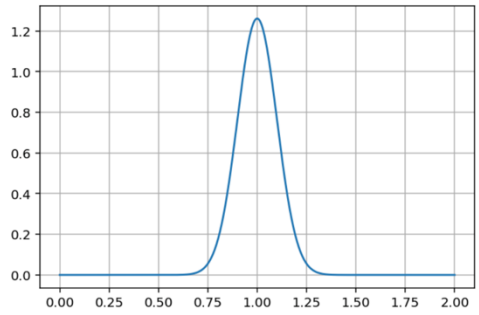
아래는 우리가 첫시간에 배웠던 사인곡선에서 추출된 점 열개를 그려본 것입니다. 코드 단 세줄로 될 정도로 쉽습니다.

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



아래는 가우시언 분포를 그려본 것입니다. 실제로는 더 간단하게 가우시언 분포 자체가 함수로 들어있지만, 식을 넣는 방식을 사용해 보았습니다. 이렇게 되면 식으로 나타나는 모든 분포를 그려볼 수 있습니다.

```
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)
```



<선형분류>

선형분류의 목표는 클래스 할당입니다. 회귀는 구체적인 값을 추정하는 것이 목표지만, 분류는 주어진 데이터 포인트가 어떤 class에 속하는지만 알면 됩니다.

분류할 때는 보통 데이터포인트가 클래스 하나에만 속한다고 가정합니다.(복수개의 클래스도 할당 가능한 분류기가 있지만 우리 수업에서는 다루지 않습니다.) 그래서, 데이터포인트들 사이에 그것들을 경계짓는 초평면이 있게 됩니다. 쉽게말해 '경계선'인데, 일반적인 경우를 포괄해서 수학적으로 정확하게 표현하다보니 초평면이라고 말하게 됩니다. D차원 입력공간상의 D-1차원 초평면이 결정경계, 또는 결정표면이 됩니다.

선형분류에서 또 하나 가정하는 것은, '선형분류'라는 이름에서도 알 수 있듯 데이터가 선형분리 가능해야 한다는 전제가 있습니다.

선형회귀와 달리 선형분류에서 함수의 출력은 0 아니면 1이 나오게 할 수도 있고, 0~1사이의 연속적인 값으로 할 수도 있고(이 경우 확률값으로 해석하게 됩니다) 임의의 threshold를 정하고 그 값을 결정경계로 보는 방법도 있을 수 있습니다. 정하기 나름입니다. 멀티클래스의 경우 1-hot

encoding을 씁니다.

선형분류를 푸는 방법을 크게 세가지로 나누어볼 수 있습니다. (1) 결합밀도 p(x,t)를 구하고 p(t|x)fmㄹ 알아낸 뒤, y=E[t|x]를 구하는 법 (2) p(t|x)를 구하고 y=E[t|x]를 구하는 법, (3) 판별함수를 구하는 법.

(1)방법은 아예 데이터포인트들의 분포 자체를 알아내는 것인데, 가능하면 좋겠지만 불가능하거나 지나치게 어려울 때가 많습니다. 분류만이 목적이라면 지나친 수고를 하는 것일수도 있습니다. 하지만, 만일 할수만 있다면 데이터를 새로 만드는것까지도 가능합니다. 분포를 전부 알고 있다면 해당 분포에서 샘플링만 하면 되니까요. 그래서 (1)은 '생성모델'이라고 불리기도 합니다.

(2)번 방법은 판별모델이라고 합니다. 데이터의 분포는 다 알수 없지만 데이터 포인트가 주어질 때 어떤 클래스에 속하는지 확률은 알 수 있을 때 사용하는 방법입니다.

(3)의 방법이 가장 쉬워서 가장 많이 사용하는데, 수식으로 나타내면 다음과 같습니다.

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

이때 함수 f를 activation function이라고 부르고 activation function 내의 선형함수가 내는 값을 0~1사이의 값으로 바꾸는 역할을 합니다. 쉽게말해, 어떤 '판별식'하나를 만들고 데이터포인트를 입력으로 주면 타겟클래스에 속할 확률을 출력으로 주게 됩니다.

판별모델을 사용할 때, 다중클래스의 경우 문제가 됩니다. 이때는 여러 방법이 있겠지만 개념적으로는 '각 클래스에 대해 확률값을 주는 함수 여러개(클래스의 갯수만큼)를 디자인하고 각 구간에서 최댓값을 가지는 함수에 해당하는 클래스를 배정'하는 방식을 씁니다. 좀 더 수학적으로는 다음과 같이 표현합니다.

K개의 선형함수 $y_k(\vec{x}) = \vec{w}_k^T\vec{x} + w_{k0}$를 만들고 모든 j에 대해 (j를 제외한) 특정 k에서 $y_k(\vec{x}) > y_j(\vec{x})$가 성립하면 x를 해당 클래스에 배정.

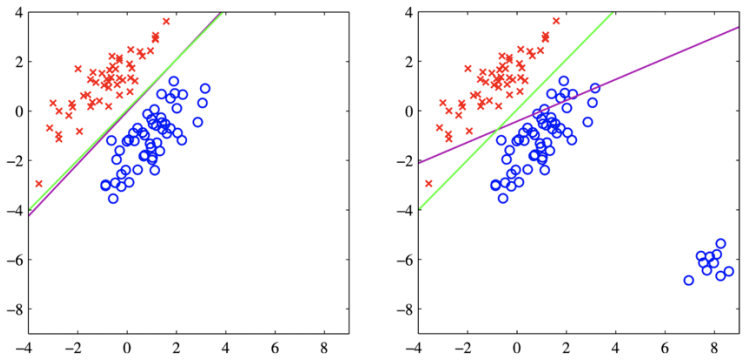
이때 클래스간 결정경계는 $y_k(\vec{x}) = y_j(\vec{x})$ 인 초평면이므로 $(\vec{w}_k - \vec{w}_j)^T\vec{x} + (w_{k0} - w_{j0}) = 0$ 으로 나타낼 수 있습니다.

이렇게 했을 때 모든 영역이 각각 하나의 영역으로 이루어짐(쪼개지지 않음)을 보일 수 있습니다. 물론 이 영역들이 쪼개진다고 해도 결과적으로 예측력이 좋으면 괜찮다고 할 수도 있겠습니다만, 이런 이상한(?!) 결정함수를 좋아할 수학자나 데이터과학자는 없을 것입니다. 더군다나 선형판별기가 이런 이상한 결과를 준다면 뭔가 중간에 잘못이 있다고 생각하는 편이 맞습니다. 다행히도 단일성은 증명할 수 있고 강의시간에 자세히 다루었으니 영상을 참고하기 바랍니다.

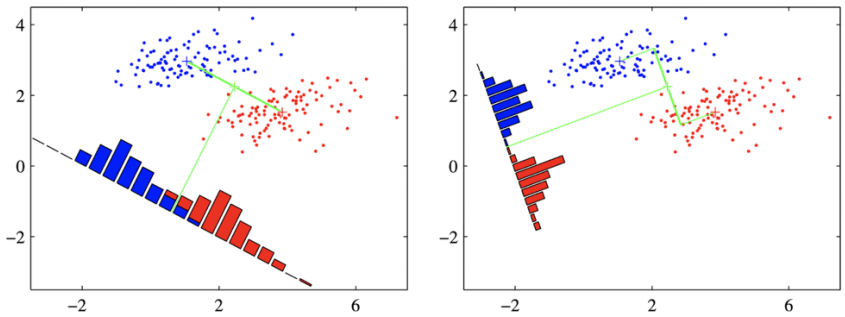
그러면 선형판별함수를 구체적으로 어떻게 학습시킬까요? 다시말해, 매개변수를 결정하는 구체적인 방법이 무엇이 있을까요. 여러가지가 있겠지만 우리가 앞으로 다룰것은 1.최소제곱법, 2.피셔의 선형판별법, 3. 퍼셉트론 알고리즘, 이 세가지 입니다.

최소제곱법은 우리가 앞서 계속 봐왔던 바로 그 방법을 말합니다. 오류함수를 최소제곱함수로 잡고 오차를 최소화 하는 방향으로 매개변수(w)를 조정합니다. 이 방법은 outlier에 민감하다는 단점을 가지고 있습니다. 오류가 결정경계에서 멀어질수록 제곱으로 증가하기 때문에 멀리 떨어지면 떨어질수록 함수에 영향을 많이 주게 됩니다.

아래 그림은 Bishop책의 fig4.4인데, 오른쪽 그림의 파란점이 결정경계에서 멀리 떨어져 있어서 최소제곱오차를 최소화하는 결정경계가 녹색선에서 보라색선으로 이동했음을 보여줍니다. 결과적으로 분류기의 성능은 (최소제곱오차를 더 줄였음에도 불구하고) 더 안좋아졌습니다.



피셔의 선형판별법은 클래스의 분리 정도를 최대화하는 직선을 그려보자는 아이디어입니다. 위그림을 보면 알 수 있듯이 전체 데이터의 오류를 모두 고려하는 것보다 분리를 가장 잘 해내기만 하면 판별함수로서는 더 적절하다고 볼 수 있습니다. 피셔의 아이디어는 클래스의 분리 정도를 최대화하면서 동시에 클래스 내 분산을 작게 하는 벡터를 찾아내서 기준으로 삼자(피셔기준)는 것입니다. 비숍책 fig 4.6의 그림이 아래와 같은데 직관적으로 이해하기 좋습니다.



오른쪽 그림을 보면 왼쪽 그림보다 분리도가 좋습니다. 이렇게 분산을 작게 하는 방향으로 벡터를 잡고, 분리도를 최대화하고자 하는 것이 아이디어이고 식으로 정확한 해를 구해낼 수 있습니다. 얼핏 두가지 다른 기준같지만 사실 이 둘은 동일하고, 결과적으로 피셔기준은 전체클래스 내의 공분산행렬의 방향에만 의존한다는 것을 보일 수 있습니다. 두개의 클래스 각각의 분산을 고려할 필요가 없다는 점을 눈여겨 보세요.

<모델선택>

우리가 데이터셋에 대해 최선의 모델을 찾기 위해 여러 가지를 실험해보게 됩니다. 이때 모델들을 공정히 비교하기 위해서 (당연하게도) 데이터셋을 똑같이 사용하게 됩니다. 어떤 모델이 오류함수를 최소화하는지 보면 됩니다. 데이터 셋은 기본적으로 훈련셋과 시험셋으로 나뉘는데, 대부분의 모델은 하이퍼파라미터(우리가 임의로 조정하는 파라미터)를 가지기 때문에 훈련셋을 다시 둘로 나눕니다. 다시 말해 훈련셋의 일부로 학습을 진행하고, 다른 일부로 하이퍼파라미터를 정합니다. 이것을 데이터셋 하나에서 모두 할 수 없는 이유는, 학습에 사용된 데이터하나로 하이퍼 파라미터 튜닝을 할 경우 과적합이 발생하기 때문입니다. 답을 보고 변수를 조정하는 것과 같습니다. 그래서 결론적으로 데이

터셋을 셋을 훈련/검증/시험 이렇게 셋으로 나누게 됩니다.

만약 데이터싸이언스 컴피티션에 나가게 되었다면, 그때도 데이터를 셋으로 나눠야 할까요? 그렇지 않습니다. 그때는 주최측에서 공개하지 않은 테스트셋을 가지고 있기 때문에, 우리는 훈련과 검증셋만 있으면 됩니다.

검증법은 여러 가지가 있습니다. 주로 사용되는 것은 K-fold cross-validation 입니다. 예를들어 5-fold라면, 데이터를 5등분 한 뒤, 각 조각을 제외하고 학습을 진행하는 것입니다. 따라서 학습이 총 4회 이루어지게 됩니다. 각각의 학습을 모두 종합해서 최적의 하이퍼 파라미터값을 얻습니다.

연습문제

1. (OX 문제) 유연한 모델은 낮은 편향과 높은 분산을 가진다.

   O 맞는 설명

2. (OX 문제) 편향 분산 분해에서, 각각 데이터 집합에서 구한 해와 전체평균의 차를 편향이라고 한다.

   X 분산에 대한 설명이다.

3. (OX 문제) 선형분류에서, 입력이 D 차원으로 주어진다면 그 결정경계는 D-1 차원보다 더 낮을 수 없다.

   X 더 낮을 수 있다. 이 경우 입력차원에서 쓸데 없는 정보가 더 주어져있거나 다른 차원에 의존하는 차원이 존재한다는 뜻이다. 결정경계는 설명과는 반대로 D-1 차원보다 더 높은 것이 불가능하다.

4. (OX 문제) 선형분류문제를 풀 때, 사전확률과 가능도를 모두 구하면, 새로운 데이터를 생성해낼 수도 있다.

   O 맞는 설명. 그래서 이 경우를 '생성모델'이라고 부르기도 한다.

5. (OX 문제) 일대일 분류기를 디자인할 때 모든 클래스의 쌍에 대한 분류기를 만들면, 모든 영역이 적어도 어느 한 영역에 속하게 되므로 분류에 문제가 없다.

   X 모호성문제가 발생한다. ppt 의 30 쪽 참고.

6. (OX 문제) 피셔의 선형판별법은 확률적으로 주어진 데이터를 분류해내는 방법이다

   X 비확률적 방법이다.

정리하기

1. 제곱오류를 편향과 분산으로 분해할 수 있다.

2. 편향은 전체 데이터집합에 대한 평균예측과 회귀함수의 차를 말한다.
3. 분산은 각각 데이터집합에서 구한 해와 전체평균의 차를 말한다.
4. 편향과 분산 사이에는 트레이드 오프 관계가 있다.
5. 다차원에서 구의 부피를 계산해보면, 고차원에서는 대부분의 부피가 껍질쪽에 쏠려 있음을 알 수 있다.
6. 여러 가지 모델을 실험 후, 가장 좋은 모델을 선택하기 위해 보통 데이터를 2~3 등분 해서 실험하고 비교한다.
7. 검증법은 k-fold cross validation, Monte Carlo, Nested k-fold 등 여러 가지가 가능하다.
8. 선형분류에서 입력공간이 D 차원이면, 결정표면은 보통 D-1 차원의 초평면이다.
9. 분류문제는 세가지 방법으로 풀 수 있다. (1) 결합밀도를 구하고 타겟변수의 확률변수 자체를 모델링 하는 방법, (2) 결합밀도 없이 조건부확률만을 가지고 타겟변수의 확률변수를 모델링 하는 방법, (3) 확률변수의 모델링 없이 판별함수를 바로 구하는 방법.
10. 선형분류에서 다중클래스의 경우 판별함수를 디자인하고, 비선형 입력변수도 선형으로 분류 가능하다(비선형 activation function 을 이용한다)
11. 선형분류에서, 최소제곱법을 사용하면 outlier 에 민감하다.
12. 피셔의 선형판별법은 투영된 클래스의 평균사이분리정도의 최대화, 동시에 클래스 내 분산을 작게 하는 것이 아이디어다.
13. 피셔기준은 전체 클래스 내 공분산행렬의 방향에만 의존한다.

참고하기

Bishop, C. M. "Bishop-Pattern Recognition and Machine Learning-Springer 2006." Antimicrob. Agents Chemother (2014): 03728-14.

다음 차시 예고

- 퍼셉트론 알고리즘
- 확률적 생성모델/판별모델
- 로지스틱 회귀
- 다층 퍼셉트론