

지지벡터기계(support vector machine)

서울대학교
통계학과

2018년 8월 23일

노트. 다루는 내용

지지 벡터 기계는 1990년대에 컴퓨터과학계에서 개발한 분류기이다. 다양한 문제에서 수행능력이 좋게 나타나, 가장 좋은 알고리즘 중 하나로 알려져 있다(one of best "out of the box" classifier). 아래의 세 가지 분류기를 통칭해서 지지 벡터 기계들이라고 하기도 하는데, 여기서는 분류를 정확히 한다.

1. 최대 여백 분류기(maximal margin classifier) linear separable
2. 지지 벡터 분류기(support vector classifier) margin soft
3. 지지 벡터 기계(support vector machine; SVM) soft margin x kernel

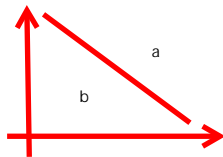
초평면(hyperplanes) I

정의 hyperplanes: $p-1$

\mathbb{R}^p 에서

$$\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0$$

를 만족하는 $x = (x_1, \dots, x_p)$ 들의 집합.



a:
b:
-
a
-hyperplane

예

\mathbb{R}^2 의 경우

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

를 만족하는 x 들의 집합은 직선이다. 3차원에서는 평면이다.

초평면(hyperplanes) II

성질

초평면은 전체공간 \mathbb{R}^p 를 두 개의 집합으로 분리를 한다.

$$\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p > 0$$

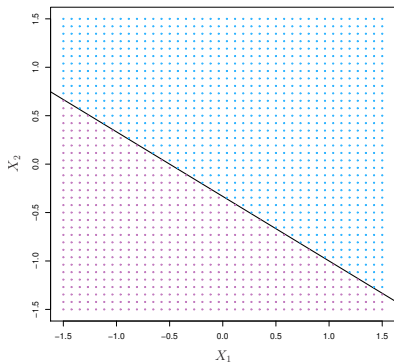
인 것과

$$\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p < 0$$

인 것으로 나눈다.

\Rightarrow 관측치 x 가 어디에 속하는지 위의 식을 구해서 알아볼 수 있다.


초평면(hyperplanes) III



그림에서는 $1 + 2x_1 + 3x_2 = 0$ 의 초평면이 $1 + 2x_1 + 3x_2 > 0$ 인 부분 (파란색)과 $1 + 2x_1 + 3x_2 < 0$ 인 부분(보라색)으로 나뉘는 것을 보여준다.

분리초평면(separating hyperplanes) I

자료

1. p 개의 설명변수와 n 개의 관측치.
2. 관측치들의 설명변수는 $x_1 = (x_{11}, \dots, x_{1p})', \dots, x_n = (x_{n1}, \dots, x_{np})'$ 으로 표현한다.

3. 반응변수 y_1, \dots, y_n 는 1 또는 -1의 값을 갖는다.

목적

위의 자료에 기반하여 새로운 설명변수 값 $x^* = (x_1^*, \dots, x_p^*)'$ 에 대해서, 이 값에 해당하는 반응변수 y^* 가 1 혹은 -1인지 분류하는 것이 목적이다.

분리초평면(separating hyperplanes) II

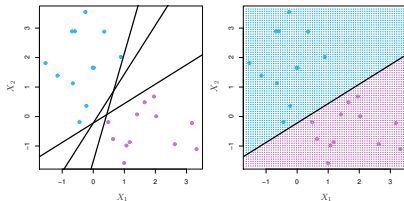
분리초평면

주어진 관측치들을 반응변수의 값에 따라 완벽하게 분류하는 초평면을 분리초평면이라고 한다. 즉,

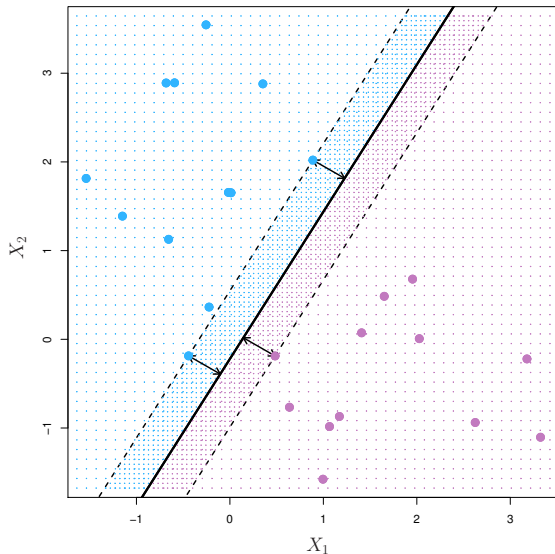
$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) > 0, \forall i = 1, 2, \dots, n$$

"for all"

을 만족하는 초평면을 분리초평면이라고 한다.



최대여백분류기(maximal margin classifier) II



1. linear seperable
- 2.

최대여백분류기(maximal margin classifier) III

최적화 문제를 통한 최대여백분류기의 수리적 정의

$$\begin{aligned} & \max_{\beta_0, \dots, \beta_p} M \quad \text{margin} = 1 / \|\beta\| \\ \text{subject to } & \sum_{j=1}^p \beta_j^2 = 1, \quad \rightarrow \quad \text{가 1} \\ & y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M, \quad \forall i = 1, \dots, n \end{aligned}$$

최대여백분류기(maximal margin classifier) IV

참고

1. $\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0$ 로 정의되는 초평면은 모든 $k > 0$ 에 대해, $k(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p) = 0$ 로 정의되는 초평면과 동일하다.

이러한 모호함을 없애기 위해, $\sum_{j=1}^p \beta_j^2 = 1$ 조건을 두었다.

2. $\sum_{j=1}^p \beta_j^2 = 1$ 조건하에서, $y_i(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p)$ 는 관측치 i 와 초평면 사이의 거리이다. \Rightarrow 다음 페이지 참조

최대여백분류기(maximal margin classifier) V

간단한 사실들

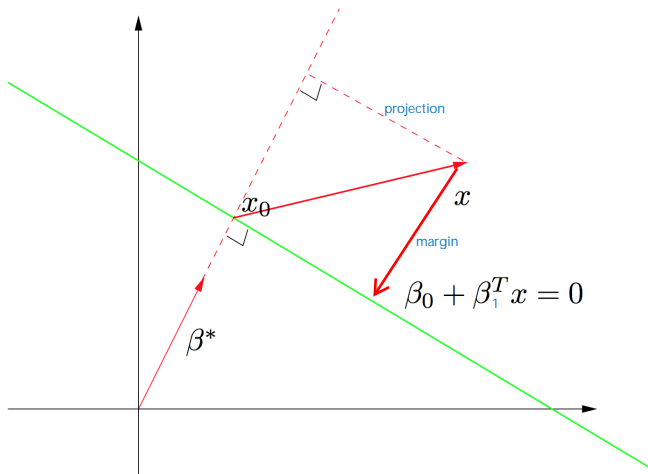
초평면 $L := \{x : \beta_0 + \beta_1'x = 0\}$ 을 고려하자.

1. (수직벡터) $x_1, x_2 \in L$ 이면, $\beta_1'(x_1 - x_2) = 0$ 이므로, $\beta^* = \beta_1 / \|\beta_1\|$ 는 L 에 수직인 단위벡터이다.
2. (임의의 점 x 에서 L 까지의 거리) L 상의 임의의 점 x_0 를 고려하자. x 에서 L 까지의 거리 (다음 페이지 그림 참조)는 $x - x_0$ 와 L 의 수직벡터 β^* 의 내적의 절대값이다. 내적을 구해보자.

$$\begin{aligned}\beta^{*'}(x - x_0) &= \left(\frac{\beta_1}{\|\beta_1\|}\right)'(x - x_0) \\ (\text{, } x \text{ } L \text{ }) &= \frac{1}{\|\beta_1\|}(\beta_1'x - \beta_1'x_0) \\ &\quad \text{모든 } x_0 \in L \text{에 대해, } \beta_1'x_0 = -\beta_0 \text{ 이므로} \\ &= \frac{1}{\|\beta_1\|}(\beta_1'x + \beta_0)\end{aligned}$$

위의 값은 거리에 부호가 붙은 값이다.

최대여백분류기(maximal margin classifier) VI



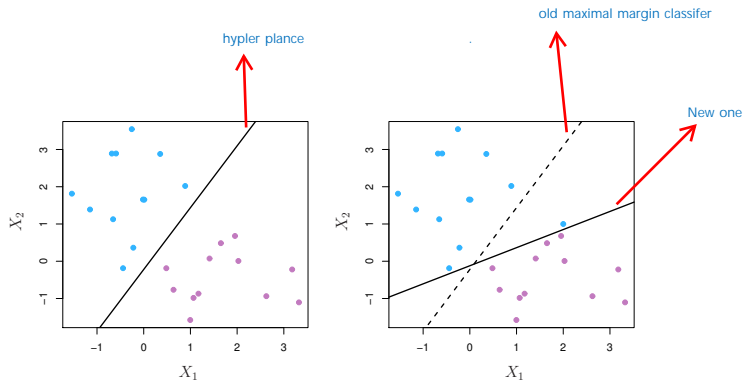
지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) |

가 가 , 가

동기

1. 최대여백분류기는 분리초평면이 존재하는 경우만 적용이 되는데, 분리초평면이 존재하지 않는 경우가 많다.
2. 최대여백분류기는 한두 개의 관측치에 크게 영향을 받을 수 있다. 즉, 로버스트(robust)하지 않다. SVM가
3. 몇 개의 관측치를 오분류하더라도 로버스트한 분류기가 필요하다.

지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) II



지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) III

지지벡터분류기의 수리적 정의

$$\begin{aligned} & \max_{\beta_0, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n} M \\ \text{subject to } & \sum_{j=1}^p \beta_j^2 = 1, \\ & y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \quad i = 1, \dots, n \\ & \epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \end{aligned}$$

x 0



C=0 hard margin .

지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) IV

설명

1. $\epsilon_1, \dots, \epsilon_n$ 을 여유 변수(slack variable)들이라 한다.
2. 여유변수들의 값에 따라 관측치의 위치를 알 수 있다. 관측치 i 가 지지벡터와 초평면 사이에 있으면 잘못된 여백에 있다고 하고, 지지벡터보다 먼 쪽에 있으면, 올바른 여백에 있다고 한다. 초평면 건너편에 있으면 잘못된 쪽에 위치한다고 한다.
 - 2.1 $\epsilon_i = 0$ 이면, 관측치 i 는 올바른 여백에 있고,
 - 2.2 $0 < \epsilon_i \leq 1$ 이면, 관측치 i 는 잘못된 여백에 있고, (wrong side of the margin)
 - 2.3 $\epsilon > 1$ 이면, 관측치 i 는 잘못된 쪽에 위치한다. (wrong side of the hyperplane)
3. C 는 선택해야 하는 조율파라미터로, 예산(budget)이라고 하기도 한다. Cross Validation 교차검증으로 선택한다.
4. 지지벡터의 해는 잘못된 여백 혹은 여백 상에 있는 관측치에만 영향을 받는다. 초평면에서 먼 관측치들의 변화에 로버스트하다.

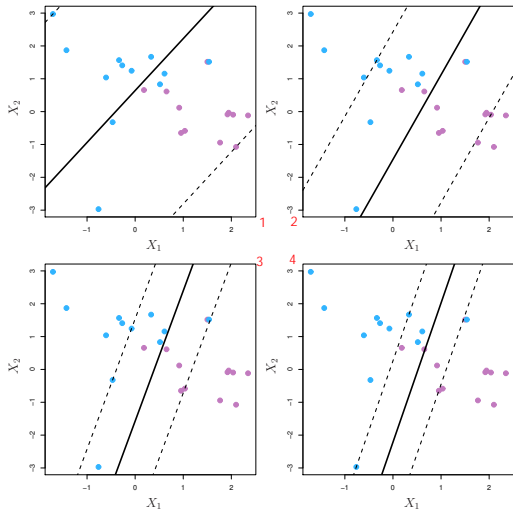
지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) V

hyperplane
C가



5. C가 커지면 지지벡터의 수가 커져서 분산이 작아진다. C가 작아지면 지지벡터의 수가 작아져서 분산이 커지고, 편이가 작아진다.

지지벡터분류기 혹은 폭신한 여백 분류기(soft margin classifier) VI



1 C가 가
4 C가 가

예산 C 값의 크기에 따른 분류기의 변화를 보여준다.

지지벡터분류기 R 코드 I

e1071 패키지 소개

- SVM이 처음 구현된 패키지
- 주요 함수 : `svm()`, `plot()`, `tune()`
CV

지지벡터분류기 R 코드 I

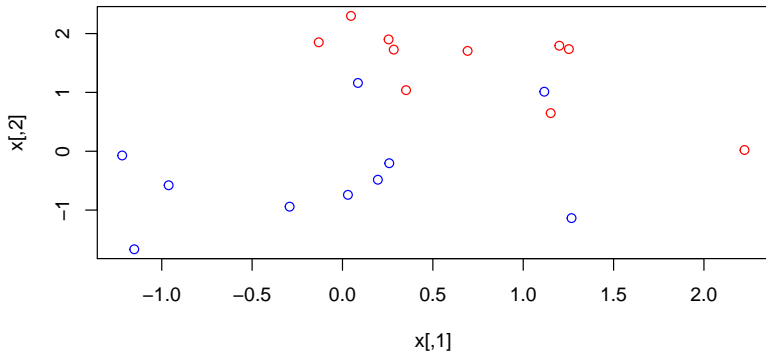
자료의 생성

```
set.seed(1)
x=matrix(rnorm(20*2), ncol=2)
y=c(rep(-1,10), rep(1,10))
x[y==1,]=x[y==1,] + 1
```

20개의 2차원 자료를 생성해서 2차원 설명변수 x 를 구성하고, 첫 10개에는 $y = 1$ 을 나머지는 $y = -1$ 을 배정하였다. 그리고 $y = 1$ 인 설명변수는 일괄적으로 1을 더했다. 그림은 다음과 같다.

```
plot(x, col=(3-y))
```

지지벡터분류기 R 코드 II



y값에 따라 색깔이 달리 표시되게 하였다.

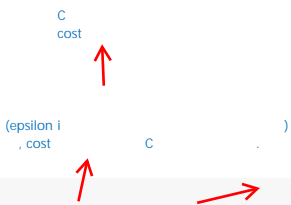
```
dat=data.frame(x=x, y=as.factor(y))
```

지지벡터분류기 R 코드 III

자료를 데이터프레임으로 만들었다.

지지벡터분류기의 적합

```
library(e1071)
svmfit=svm(y~., data=dat, kernel="linear", cost=10, scale=FALSE)
```

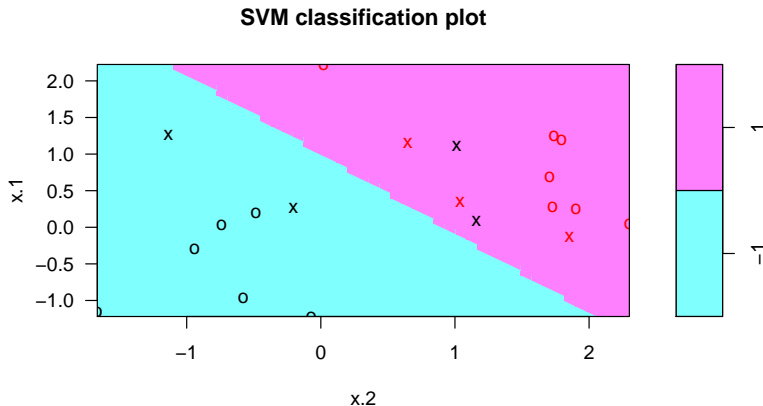


e1071 패키지 안의 svm 함수를 이용한다. svm 함수의 옵션은 다음을 의미한다.

1. kernel ="linear" 는 결정경계가 선형이라는 뜻이다.
2. cost는 '여백 위반비용'을 설정하는 것으로, 이 값이 작아지면 정해진 예산하에서 여백이 커지고 서포트벡터도 많고, 여백 위반하는 관측치도 많아진다.
3. scale =FALSE 는 설명변수를 표준화하지 않는다는 뜻이다. 문제에 따라서는 표준화하는 것이 좋을 때도 많다.

지지벡터분류기 R 코드 IV

```
plot(svmfit, dat)
```



그림의 x 표시는 지지벡터를 나타내고, 그 외의 벡터들은 o로 표시되었다.
자료의 색깔은 y 값을 표시한다. 그림이 위에서 그린 자료의 그림과

지지벡터분류기 R 코드 V

달라보이는데, 이는 좌표축의 선택이 다르기 때문이다. 여기서는 x_2 가 x 축에 x_1 이 y 축에 그려졌다.

```
svmfit$index
```

```
## [1] 1 2 5 7 14 16 17
```

⇒ 지지벡터가 되는 관측치 번호를 알려준다.

지시벡터분류기 R 코드 VI

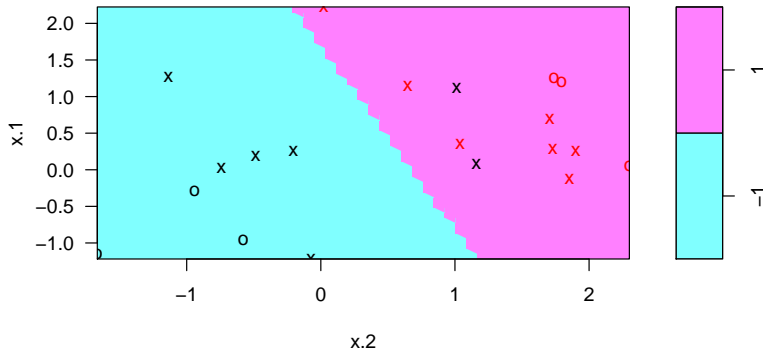
```
summary(svmfit)

##
## Call:
## svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10,
##      scale = FALSE)
##
##
## Parameters:
##      SVM-Type:  C-classification
##      SVM-Kernel: linear
##              cost: 10
##              gamma: 0.5
##
## Number of Support Vectors: 7
##
##      ( 4 3 )
##
##
## Number of Classes: 2
##
## Levels:
##      -1 1
```

지지벡터분류기 R 코드 VII

```
svmfit=svm(y~., data=dat, kernel="linear", cost=0.1, scale=FALSE)  
plot(svmfit, dat)
```

SVM classification plot



지지벡터분류기 R 코드 VIII

```
svmfit$index
```

```
## [1] 1 2 3 4 5 7 9 10 12 13 14 15 16 17 18 20
```

⇒ cost=0.1로 적합한 경우 여백이 크게 나타나고, 지지벡터의 수가 늘어났다.

교차검증

```
set.seed(1)
```

```
tune.out=tune(svm,y~.,data=dat,kernel="linear",ranges=list(cost=c(0.001, 0.01, 0.1, 1,5,10
```

교차검증은 tune 함수를 이용한다. range는 교차검증할 때 비교할 cost의 값들을 지정한다.

지지벡터분류기 R 코드 IX

```
summary(tune.out)

##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   cost
##   0.1
##
## - best performance: 0.1
##
## - Detailed performance results:
##   cost error dispersion
## 1 1e-03  0.65  0.4743416
## 2 1e-02  0.65  0.4743416
## 3 1e-01  0.10  0.2108185
## 4 1e+00  0.15  0.2415229
## 5 5e+00  0.10  0.2108185
## 6 1e+01  0.10  0.2108185
## 7 1e+02  0.15  0.2415229
```

지지벡터분류기 R 코드 X

```
bestmod=tune.out$best.model
summary(bestmod)

##
## Call:
## best.tune(method = svm, train.x = y ~ ., data = dat, ranges = list(cost = c(0.001,
##      0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
##
## Parameters:
##   SVM-Type:  C-classification
##   SVM-Kernel: linear
##       cost:  0.1
##      gamma:  0.5
##
## Number of Support Vectors:  14
##
##   ( 7 7 )
##
##
## Number of Classes:  2
##
## Levels:
##   -1 1
```

지지벡터분류기 R 코드 XI

예측

```
xtest=matrix(rnorm(20*2), ncol=2)
ytest=sample(c(-1,1), 20, rep=TRUE)
xtest[ytest==1,]=xtest[ytest==1,] + 1
testdat=data.frame(x=xtest, y=as.factor(ytest))
ypred=predict(bestmod,testdat)
table(predict=ypred, truth=testdat$y)
```

```
##          truth
## predict -1  1
##        -1 10  1
##         1  1  8
```

20개 중 18개를 맞추고 2개를 틀렸다. 아래는 cost = 0.01일 때의 예측결과이다. 20개 중 16개를 맞추었다.

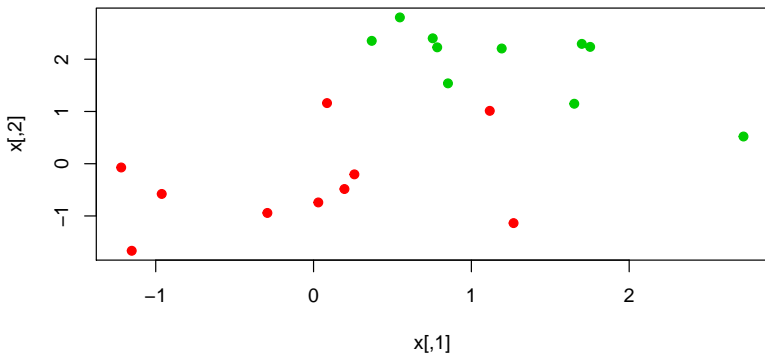
```
svmfit=svm(y~., data=dat, kernel="linear", cost=.01,scale=FALSE)
ypred=predict(svmfit,testdat)
table(predict=ypred, truth=testdat$y)
```

```
##          truth
## predict -1  1
##        -1  7  0
##         1  4  9
```

지지벡터분류기 R 코드 XII

아래는 자료가 완전히 분리되는 경우 분리초평면을 어떻게 구하는가를 나타낸다. 자료는 완전 분리가 된다.

```
x[y==1,]=x[y==1,]+0.5  
plot(x, col=(y+5)/2, pch=19)
```



지시벡터분류기 R 코드 XIII

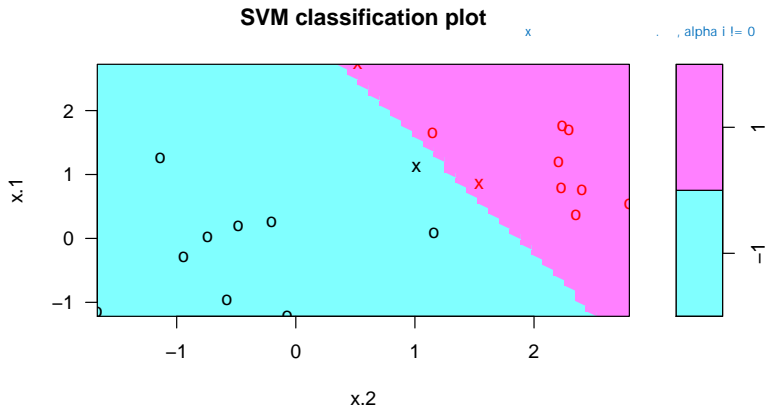
```
dat=data.frame(x=x, y=as.factor(y))
svmfit=svm(y~., data=dat, kernel="linear", cost=1e5)
summary(svmfit)

##
## Call:
## svm(formula = y ~ ., data = dat, kernel = "linear", cost = 1e+05)
##
##
## Parameters:
##   SVM-Type:  C-classification
##   SVM-Kernel: linear
##           cost: 1e+05
##           gamma: 0.5
##
## Number of Support Vectors:  3
##
##   ( 1 2 )
##
##
## Number of Classes:  2
##
## Levels:
##   -1 1
```

예산을 작게하면(cost를 크게하면) 여백이 작게 되어서 분리초평면을 구하게 된다.

지지벡터분류기 R 코드 XIV

```
plot(svmfit, dat)
```



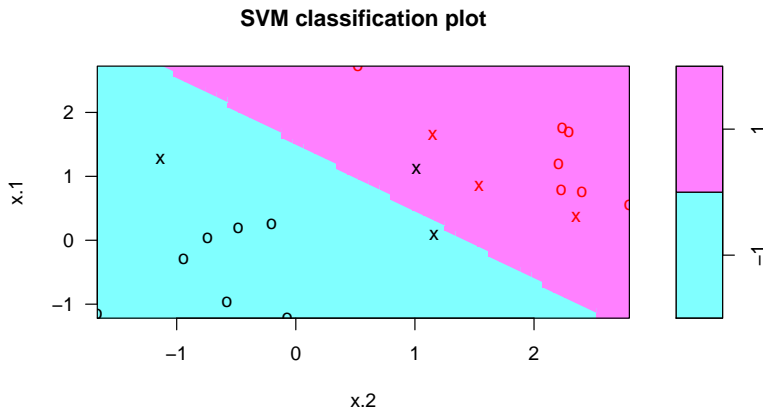
지지벡터분류기 R 코드 XV

```
svmfit=svm(y~., data=dat, kernel="linear", cost=1)
summary(svmfit)

##
## Call:
## svm(formula = y ~ ., data = dat, kernel = "linear", cost = 1)
##
##
## Parameters:
##   SVM-Type:  C-classification
##   SVM-Kernel: linear
##         cost:  1
##        gamma: 0.5
##
## Number of Support Vectors:  6
##
##   ( 3 3 )
##
##
## Number of Classes:  2
##
## Levels:
##   -1 1

plot(svmfit,dat)
```

지지벡터분류기 R 코드 XVI

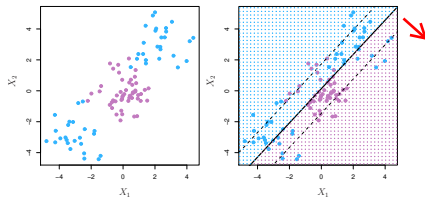


예산을 크게하면(cost를 작게하면) 여백이 크게 되어서 오분류하는 관측치가 생기게 된다. 한 개의 오분류가 보인다.

지지벡터기계 I

가

동기



가

SVM

가

결정경계가 비선형인 경우 지지벡터분류기의 성능에 문제가 있을 수 있다.

지지벡터기계 II

다항회귀모형과 같이 설명변수들을 $x_1, \dots, x_p, x_1^2, \dots, x_p^2$ 로 확장하여 다음과 같이 지지벡터분류기를 구축할 수도 있다.

$$\sum_{j=1}^p \beta_{j1}^2 + \sum_{j=1}^p \beta_{j2}^2 = 1,$$

$$y_i(\beta_0 + \sum_{j=1}^p \beta_{j1}x_{ij} + \sum_{j=1}^p \beta_{j2}x_{ij}^2) \geq M(1 - \epsilon_i), \quad i = 1, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

x 가

의 조건하에서,

M 을 $\beta_0, \beta_{11}, \beta_{21}, \dots, \beta_{p1}, \beta_{12}, \beta_{22}, \dots, \beta_{p2}$ 와 $\epsilon_1, \dots, \epsilon_n$ 에 관해서 최대화한다.

지지벡터기계 III



효과

이 경우, 결정경계가 확장된 설명변수($x_1, \dots, x_n, x_1^2, \dots, x_n^2$)의 공간에서는 선형이지만 이를 원래의 설명변수의 공간(x_1, \dots, x_n)으로 환원시키면 비선형이 된다.

문제점

이 경우 변수의 개수가 너무 많아지면 계산이 힘들어 질 수 있는데, 지지벡터기계는 이 계산을 쉽게 하도록 한 것이다.

동기

x 를 $h(x)$ 로 변환하여 $h(x)$ 의 공간에서 지지벡터분류기를 적용하면 $h(x)$ 의 형태에 따라 x 의 공간에서는 비선형분류기를 생성할 수 있다.

지지벡터기계 IV

최적화이론에 의하면 $h(x)$ 를 새로운 변수로 지지벡터분류기를 적용하는 문제는

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j'=1}^n \alpha_i \alpha_{j'} y_i y_{j'} < \underbrace{h(x_i), h(x_{j'})}_{\text{inner product}} > \\ \text{subject to} \quad & \alpha_i \geq 0, i = 1, 2, \dots, n \end{aligned}$$

와 같고,
구해진 초평면은

$$h(x)' \beta + \beta_0 = \sum_{i=1}^n \alpha_i y_i < h(x), h(x_i) > + \beta_0$$

와 같이 표현된다. 즉, 최적화문제와 초평면이 모두 $h(x)$ -공간에서의 내적으로 표현된다. 만약,

$$K(x, y) = \langle h(x), h(y) \rangle, \forall x, y$$

와 같이 표현된다면, 함수 h 를 몰라도 K 를 이용해 지지벡터분류기를 구할 수 있다. 이와 같은 방법이 지지벡터기계이다.

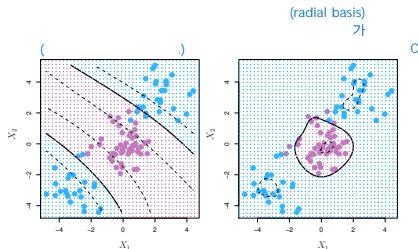
지지벡터기계 V

커널

많이 쓰는 커널은 다음과 같다.

1. (d 차 다항식) $K(x, y) = (1 + \langle x, y \rangle)^d$
2. (방사기저[radial baiss]) $K(x, y) = e^{-\|x-y\|^2/c}$
3. (신경망) $K(x, y) = \tanh(\kappa_1 \langle x, y \rangle + \kappa_2)$

예



왼쪽은 차수가 3인 다항식 커널을 적용한 결과이고, 오른쪽은 방사커널을 적용한 결과이다.

지지벡터기계와 로지스틱회귀모형의 관계 I

지지벡터기계의 또다른 최적화 문제

$f(x) = \beta_0 + \beta'x$ 라고 하면, 지지벡터기계의 분류기는 다음과 같이 구할 수 있다.

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^n \max(0, 1 - y_i f(x_i)) + \lambda \|\beta\|^2 \right\}$$

위의 형태는 마치

손실함수 + 벌점

혹은

로그가능도 + 사전분포

의 형태와 같다. 왼쪽항을 경첩 손실함수(hinge loss)라고 한다.
경첩손실함수는

$$l(f(x)) = \max(0, 1 - yf(x))$$

로 정의된다. y 는 반응변수의 값이다.

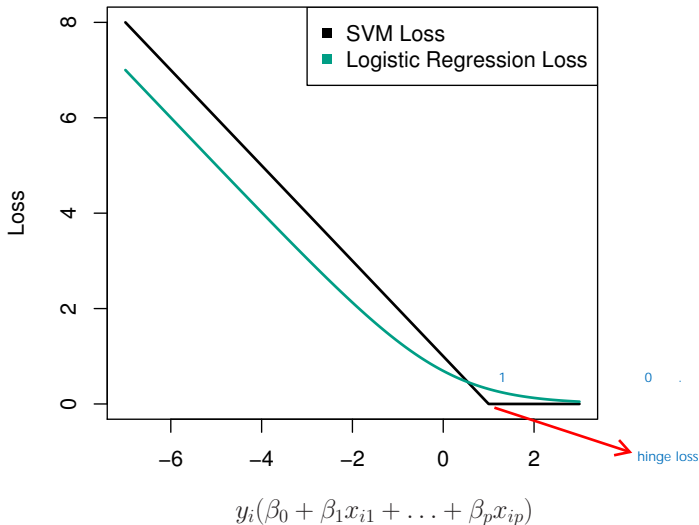
로지스틱모형의 로그가능도와 비교해보면 아래와 같아 둘이 유사함을 알 수 있다.

지지벡터기계와 로지스틱회귀모형의 관계 II

"linear seperable

SVM

가

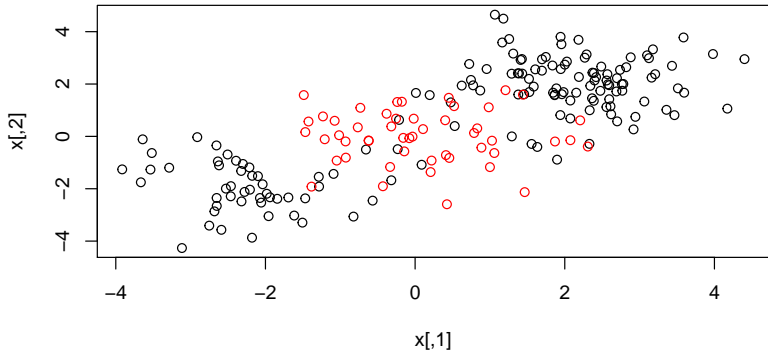


지시벡터기계R 코드 I

자료의 생성

```
set.seed(1)
x=matrix(rnorm(200*2), ncol=2)
x[1:100,]=x[1:100,]+2
x[101:150,]=x[101:150,]-2
y=c(rep(1,150),rep(2,50))
dat=data.frame(x=x,y=as.factor(y))
plot(x, col=y)
```

지지벡터기계R 코드 II



지지벡터기계의 적합

지지벡터기계R 코드 III

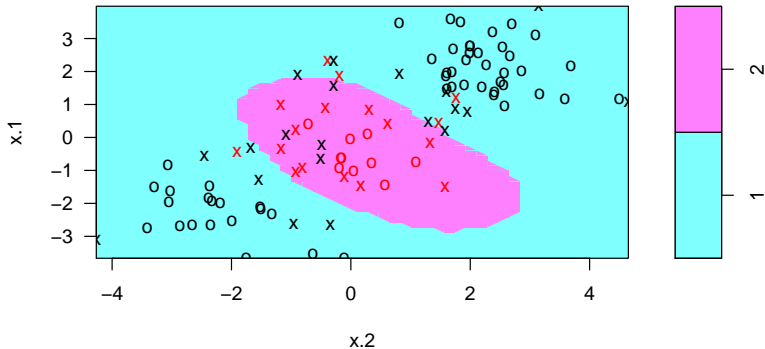
```
train=sample(200,100)
svmfit=svm(y~., data=dat[train,], kernel="radial", gamma=1, cost=1)
```

지지벡터기계의 적합은 svm 함수를 이용하면 되고 단지 커널만 바꾸면 된다.

```
plot(svmfit, dat[train,])
```

지지벡터기계R 코드 IV

SVM classification plot



지지벡터기계R 코드 V

```
summary(svmfit)

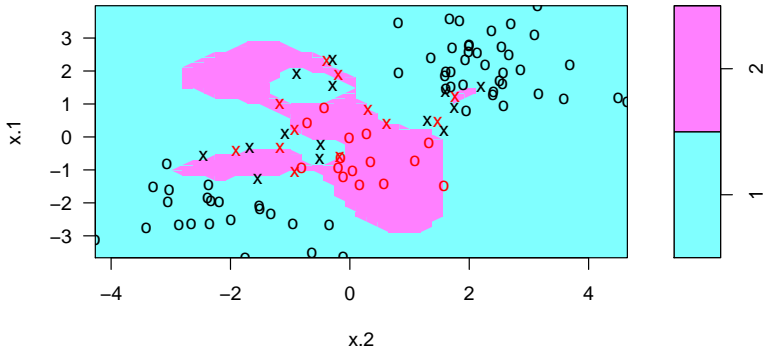
##
## Call:
## svm(formula = y ~ ., data = dat[train, ], kernel = "radial",
##      gamma = 1, cost = 1)
##
##
## Parameters:
##      SVM-Type:  C-classification
##      SVM-Kernel: radial
##              cost:  1
##              gamma: 1
##
## Number of Support Vectors:  37
##
## ( 17 20 )
##
##
## Number of Classes:  2
##
## Levels:
##  1 2
```

cost의 값을 크게 하고 다시 적합하였다.

지지벡터기계R 코드 VI

```
svmfit=svm(y~., data=dat[train,], kernel="radial",gamma=1,cost=1e5)  
plot(svmfit,dat[train,])
```

SVM classification plot



비용과 감마의 값을 교차검증을 이용해서 결정한다.

지시벡터기계R 코드 VII

```
set.seed(1)
tune.out=tune(svm, y~., data=dat[train,], kernel="radial", ranges=list(cost=c(0.1,1,10,100
```

```
summary(tune.out)
```

```
table(true=dat[-train,"y"], pred=predict(tune.out$best.model,newx=dat[-train,]))
```

```
##      pred
## true  1  2
##      1 56 21
##      2 18  5
```

참고문헌

아래의 책에서 제공하는 그림들을 사용하였다.

1. Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Springer, 2013.