HW11

M072040019 梅瀚中

## 第四題

# (a)

1. Steadily increase. 因為越大，被限制的越多，模型越不靈活，training RSS越大。

# (b)

1. Decrease initially, and then eventually start increasing in a U shape. 因為越大，被限制的越多，模型越不靈活，一開始test RSS會下降，但過度不靈活的模型可能配適不佳，所以test RSS會漸漸增加。

# (c)

1. Steadily decrease. 因為越大，被限制的越多，模型越不靈活，variance會越小。

# (d)

1. Steadily increase. 因為越大，被限制的越多，模型越不靈活，bias會越大。

# (e)

1. Remain constant.irreducible error會保持常數。

## 第五題

# (a)

minimize

# (b)

令，

對偏微分，並令其為0

得到

兩式相減

因此

# (c)

minimize

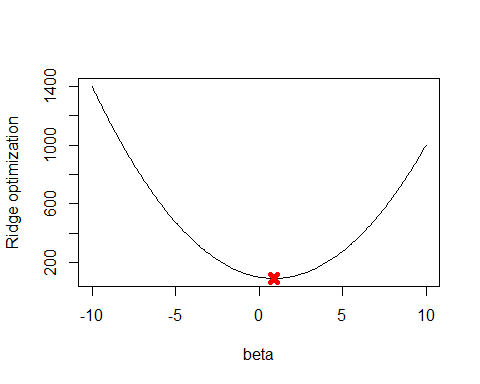
# (d)

作法同(b)， 若，則 若，則

## 第六題

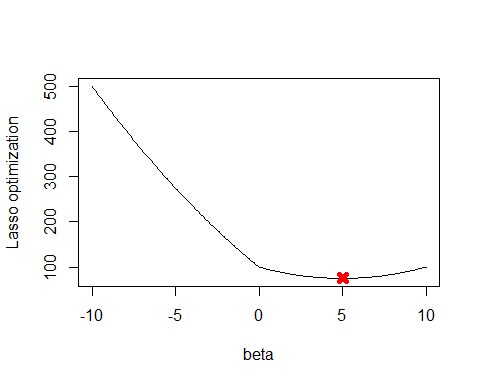
# (a)

y <- 10  
lambda <- 10  
beta <- seq(-10, 10, 0.1)  
plot(beta, (y - beta)^2 + lambda \* beta^2, type = "l", xlab = "beta", ylab = "Ridge optimization")  
beta.est <- y / (1 + lambda)  
points(beta.est, (y - beta.est)^2 + lambda \* beta.est^2, col = "red", pch = 4, lwd = 5)



# (b)

y <- 10  
lambda <- 10  
beta <- seq(-10, 10, 0.1)  
plot(beta, (y - beta)^2 + lambda \* abs(beta), type = "l", xlab = "beta", ylab = "Lasso optimization")  
beta.est <- y - lambda / 2  
points(beta.est, (y - beta.est)^2 + lambda \* abs(beta.est), col = "red", pch = 4, lwd = 5)



## 第十題

# (a)

set.seed(1)  
x <- matrix(rnorm(1000 \* 20), 1000, 20)  
b <- rnorm(20)  
z <- sample(1:20, 5)  
b[z] <- 0  
eps <- rnorm(1000)  
y <- x %\*% b + eps

# (b)

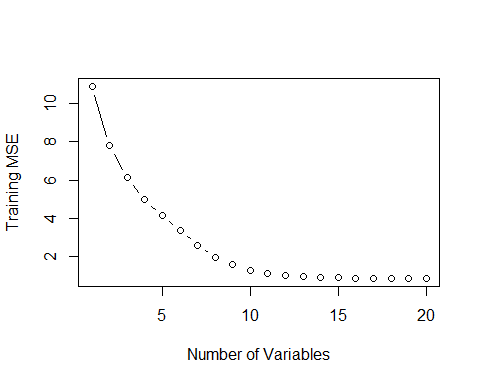
set.seed(1)  
train <- sample(1:1000, 100)  
x.train <- x[train,]  
x.test <- x[-train,]  
y.train <- y[train,]  
y.test <- y[-train,]

# (c)

library(leaps)

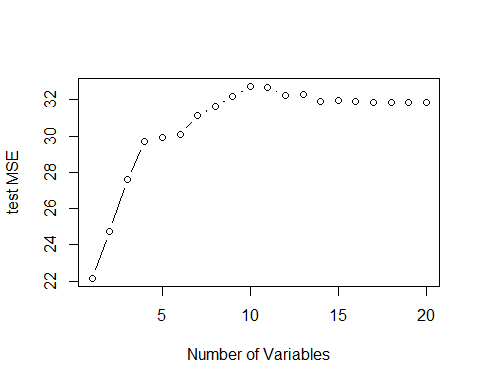
## Warning: package 'leaps' was built under R version 3.4.4

data.train <- data.frame(y = y.train, x = x.train)  
regfit.full <- regsubsets(y~., data = data.train, nvmax = 20)  
train.mat <- model.matrix(y ~ ., data = data.train, nvmax = 20)  
val.errors <- rep(NA, 20)  
for (i in 1:20) {  
 coefi <- coef(regfit.full, id = i)  
 pred <- train.mat[, names(coefi)] %\*% coefi  
 val.errors[i] <- mean((pred - y.train)^2)  
}  
plot(val.errors, xlab = "Number of Variables", ylab = "Training MSE", type = "b")



# (d)

library(leaps)  
data.test <- data.frame(y = y.test, x = x.test)  
regfit.full <- regsubsets(y~., data = data.train, nvmax = 20)  
test.mat <- model.matrix(y ~ ., data = data.test, nvmax = 20)  
val.errors <- rep(NA, 20)  
for (i in 1:20) {  
 coefi <- coef(regfit.full, id = i)  
 pred <- test.mat[, names(coefi)] %\*% coefi  
 val.errors[i] <- mean((pred - y.train)^2)  
}  
plot(val.errors, xlab = "Number of Variables", ylab = "test MSE", type = "b")



# (e)

最小test MSE的變數個數

which.min(val.errors)

## [1] 1

# (f)

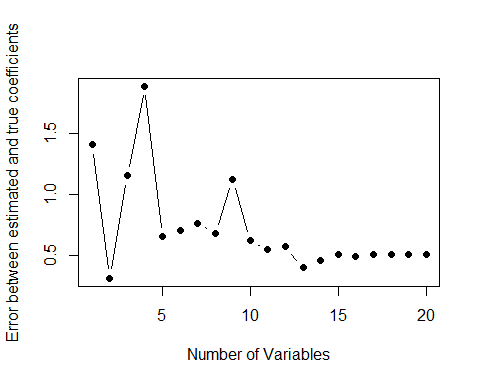
最小test MSE的模型係數

coef(regfit.full, which.min(val.errors))

## (Intercept) x.4   
## 0.003065861 -2.150241056

# (g)

val.errors <- rep(NA, 20)  
x\_cols = colnames(x, do.NULL = FALSE, prefix = "x.")  
for (i in 1:20) {  
 coefi <- coef(regfit.full, id = i)  
 val.errors[i] <- sqrt(sum((b[x\_cols %in% names(coefi)] - coefi[names(coefi) %in% x\_cols])^2) + sum(b[!(x\_cols %in% names(coefi))])^2)  
}  
plot(val.errors, xlab = "Number of Variables", ylab = "Error between estimated and true coefficients", pch = 19, type = "b")

 真實與估計誤差差距最小值出現在變數個數為2，而最小test MSE的模型係數為1，表示模型配適越好並不表示有越小的test MSE。

## 第十一題

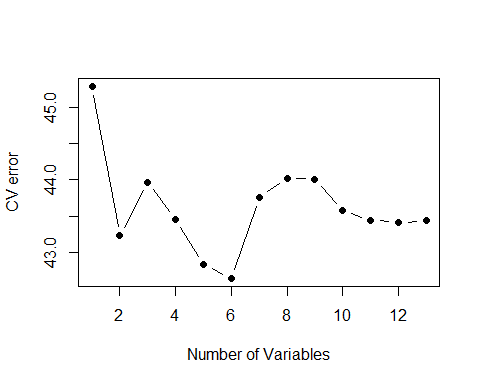
# (a)

library(MASS)

## Warning: package 'MASS' was built under R version 3.4.4

1. best subset selection

library(leaps)  
#使用交叉驗證方法  
set.seed(1)  
train <- sample(c(TRUE, FALSE), nrow(Boston), rep=TRUE)  
regfit.best <- regsubsets(crim~., data = Boston[train,], nvmax = 13)  
  
test.mat <- model.matrix(crim~., data = Boston[-train,])  
val.errors <- rep(NA,13)  
for (i in 1:13) {  
 coefi = coef(regfit.best, id=i)  
 pred = test.mat[,names(coefi)]%\*%coefi  
 val.errors[i] = mean((Boston$crim[-train]-pred)^2)  
}  
  
plot(val.errors, xlab = "Number of Variables", ylab = "CV error", pch = 19, type = "b")

 使用best subset selection，當變數個數為6個時，有最低的test error。 最小test MSE

val.errors[which.min(val.errors)]

## [1] 42.64486

1. lasso

library(glmnet)

## Warning: package 'glmnet' was built under R version 3.4.4

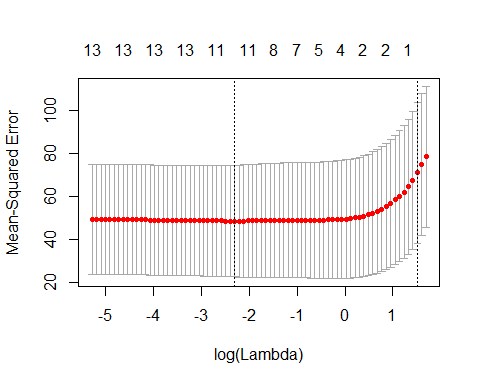
## Loading required package: Matrix

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.4.4

## Loaded glmnet 2.0-16

x <- model.matrix(crim~., Boston)[,-1]  
y <- Boston$crim  
grid <- 10^seq(10, -2, length = 100)  
set.seed(1)  
train <- sample(1:nrow(x), nrow(x)/2)  
y.test <- y[-train]  
  
lasso.mod <- glmnet(x[train,], y[train], alpha = 1, lambda = grid)  
set.seed(1)  
cv.out <- cv.glmnet(x[train,], y[train], alpha = 1)  
plot(cv.out)

 最佳

bestlam <- cv.out$lambda.min  
bestlam

## [1] 0.09979553

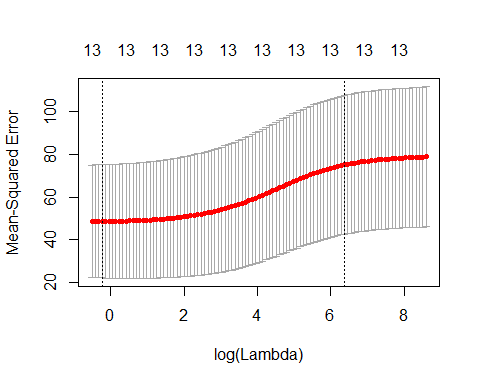
最小test MSE

lasso.pred <- predict(lasso.mod, s=bestlam, newx = x[-train,])  
mean((lasso.pred - y.test)^2)

## [1] 38.3096

1. ridge

library(glmnet)  
x <- model.matrix(crim~., Boston)[,-1]  
y <- Boston$crim  
grid <- 10^seq(10, -2, length = 100)  
set.seed(1)  
train <- sample(1:nrow(x), nrow(x)/2)  
y.test <- y[-train]  
  
ridge.mod <- glmnet(x[train,], y[train], alpha = 0, lambda = grid)  
set.seed(1)  
cv.out <- cv.glmnet(x[train,], y[train], alpha = 0)  
plot(cv.out)

 最佳

bestlam <- cv.out$lambda.min  
bestlam

## [1] 0.7908625

最小test MSE

ridge.pred <- predict(ridge.mod, s=bestlam, newx = x[-train,])  
mean((ridge.pred - y.test)^2)

## [1] 38.3679

1. PCR

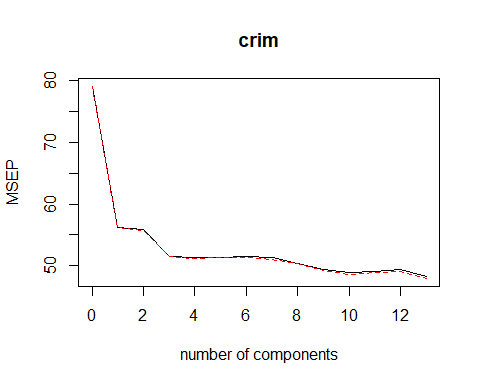
library(pls)

## Warning: package 'pls' was built under R version 3.4.4

##   
## Attaching package: 'pls'

## The following object is masked from 'package:stats':  
##   
## loadings

set.seed(1)  
x <- model.matrix(crim~., Boston)[,-1]  
y <- Boston$crim  
train <- sample(1:nrow(x), nrow(x)/2)  
test <- (-train)  
y.test <- y[-train]  
pcr.fit <- pcr(crim~., data = Boston, subset = train, scale = TRUE, validation = "CV")  
validationplot(pcr.fit, val.type = "MSEP")

 最小test MSE

pcr.pred <- predict(pcr.fit, x[test,], ncomp = 7) #M=7有最小cross-validation error  
mean((pcr.pred - y.test)^2) #test MSE

## [1] 41.53621

# (b)

lasso為最佳模型，因為其test MSE相對最小

# (c)

的lasso模型係數

library(glmnet)  
x <- model.matrix(crim~., Boston)[,-1]  
y <- Boston$crim  
grid <- 10^seq(10, -2, length = 100)  
set.seed(1)  
train <- sample(1:nrow(x), nrow(x)/2)  
y.test <- y[-train]  
lasso.mod <- glmnet(x[train,], y[train], alpha = 1, lambda = grid)  
set.seed(1)  
cv.out <- cv.glmnet(x[train,], y[train], alpha = 1)  
bestlam <- cv.out$lambda.min  
out <- glmnet(x,y,alpha = 1, lambda = grid)  
lasso.coef <- predict(out, type = "coefficient", s=bestlam)[1:14,]  
lasso.coef

## (Intercept) zn indus chas nox   
## 9.262700913 0.031356409 -0.051023135 -0.512648901 -3.755451657   
## rm age dis rad tax   
## 0.041320041 0.000000000 -0.600700390 0.494793892 0.000000000   
## ptratio black lstat medv   
## -0.107509984 -0.007556396 0.118431941 -0.126165598

所選的lasso模型沒有用到所有變數