TP2

Guangyue CHEN

Jiahui XU

Application THE Boston housing data set

(a)upload the data

```
rm(list=ls())
library(mlbench)
data(BostonHousing)
```

the first step, we try to use linear regression.

```
modreg<-lm(medv~.,BostonHousing)
summary(modreg)
```

```
##
## Call:
## lm(formula = medv ~ ., data = BostonHousing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas1        2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## b            9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

This linear model has the residual stadard error which is 4.745. But with the high R-squared and the small p-value of F-test, we don't refuse this mod. So we use the different ways to select our linear model:

Their aics are the same, we can choose no matter which one.

```r
AIC(regforward)
```

```
## [1] 3023.726
```

```r
AIC(regbackward)
```

```
## [1] 3023.726
```

```r
AIC(regbic)
```

```
## [1] 3023.726
```

```r
AIC(regboth)
```

```
## [1] 3023.726
```

```r
reg = lm(formula(regbackward), data = BostonHousing)
summary(reg)
```

```
##
## Call:
## lm(formula = formula(regbackward), data = BostonHousing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5984  -2.7386  -0.5046   1.7273  26.2373
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.341145   5.067492   7.171 2.73e-12 ***
## crim         -0.108413   0.032779  -3.307 0.001010 **
## zn            0.045845   0.013523   3.390 0.000754 ***
## chas1         2.718716   0.854240   3.183 0.001551 **
## nox         -17.376023   3.535243  -4.915 1.21e-06 ***
## rm            3.801579   0.406316   9.356  < 2e-16 ***
## dis          -1.492711   0.185731  -8.037 6.84e-15 ***
## rad           0.299608   0.063402   4.726 3.00e-06 ***
## tax          -0.011778   0.003372  -3.493 0.000521 ***
## ptratio      -0.946525   0.129066  -7.334 9.24e-13 ***
## b             0.009291   0.002674   3.475 0.000557 ***
## lstat        -0.522553   0.047424 -11.019  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.736 on 494 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7348
## F-statistic: 128.2 on 11 and 494 DF,  p-value: < 2.2e-16
```

```r
Y_esti<-predict(reg,BostonHousing)
Y<-BostonHousing$medv
Non_biased_residual<-function(Y,Y_esti,p){
sum=0

for(i in seq(1,length(Y))){
  sum<-sum+(Y_esti[i]-Y[i])^2
```

```
}
NBR<- sqrt(sum/(length(Y)-p+1))

return(NBR)
}
Non_biased_residual(Y,Y_esti,13)
```

```
##        1
## 4.736234
```

So that we obtain the model after the selection, with the function "predict" we can gain the estimation.

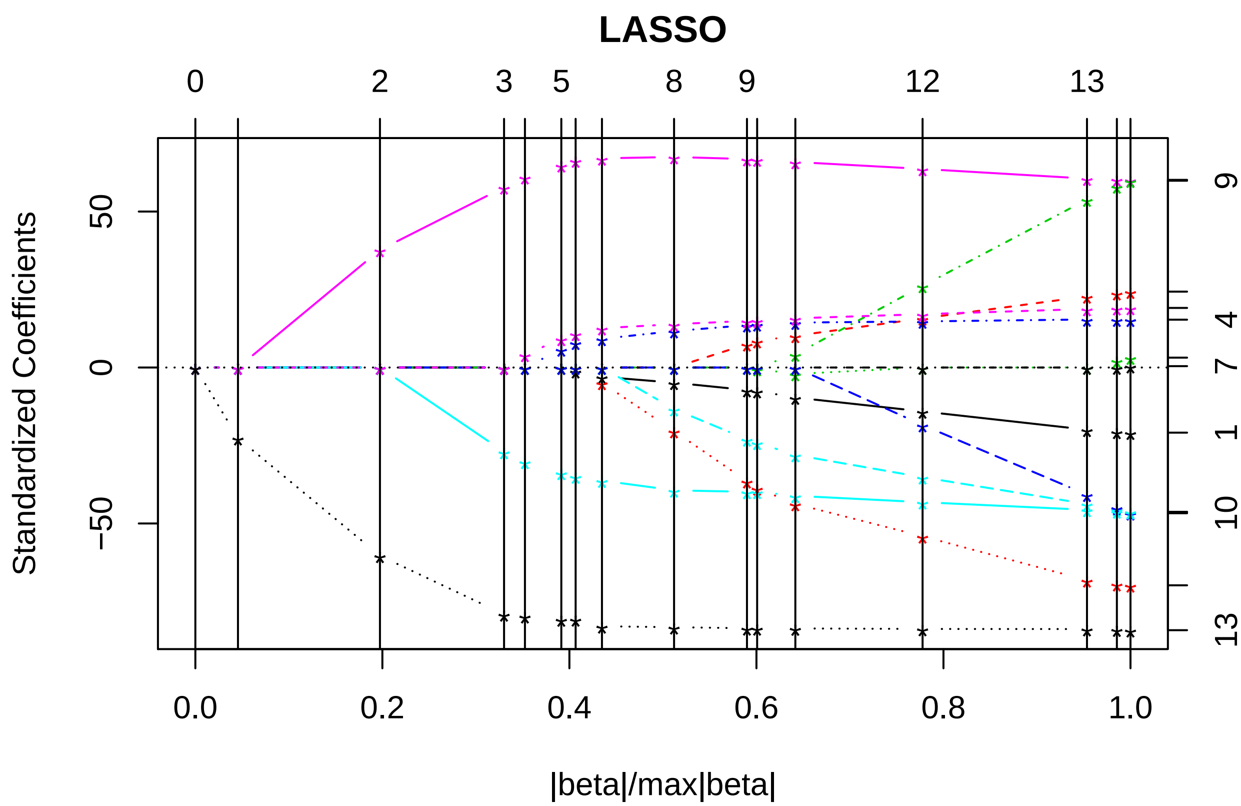
LASSO

The next step, we try the Lasso regression:

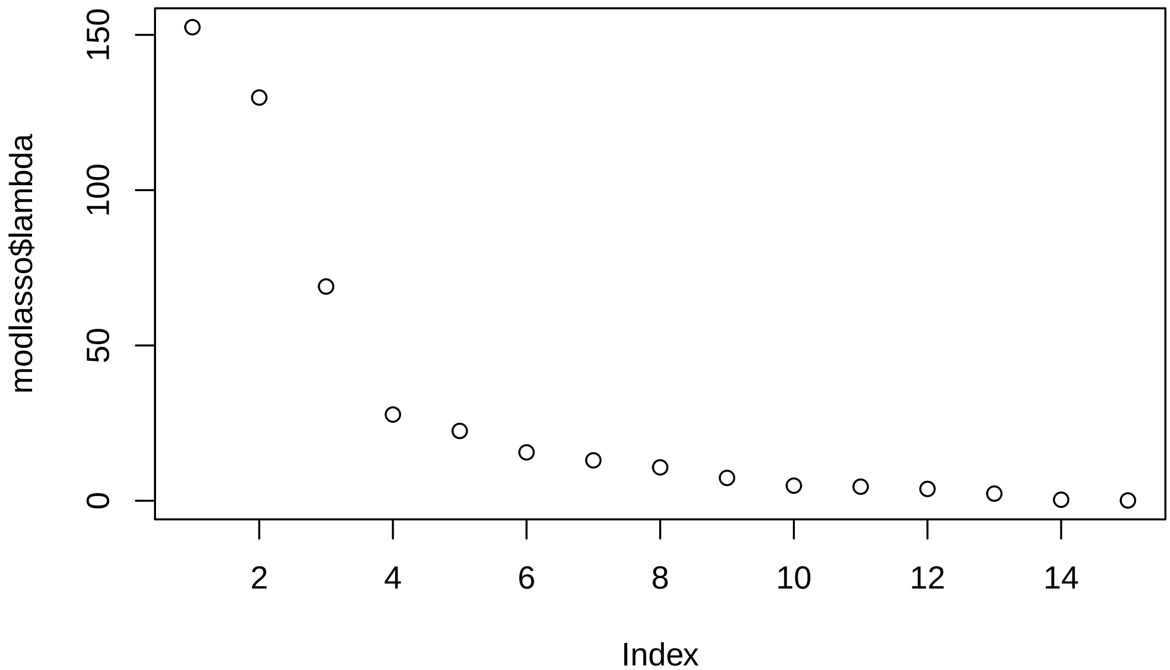
```
library(lars)
```

```
## Loaded lars 1.2
```

```
Y<-as.matrix(BostonHousing$medv)
X<-apply(as.matrix(subset(BostonHousing,select=-medv)),2,as.numeric)
modlasso=lars(x=X,y=Y,type="lasso")
plot(modlasso)
```




```
plot(modlasso$lambda)
```



From these two graphs, we can see the evolution of the values of the coefficients for different values of the penalized coefficient. And after the beta bigger than 13, the coefficients become more stable.

```
modlasso$lambda[which.min(modlasso$RSS)-1]
```

```
## [1] 0.0996448
```

With the help of criteria RSS, we choose the 16th lambda which is 0.0996448. And we found that the residual standard error is less than the Previous method but the difference is small.

```
coef<-predict.lars(modlasso,X,type="coefficient",mode="lambda",s=0.0996448)
coef$coefficients
```

```
##          crim            zn         indus          chas           nox
## -1.065847e-01  4.550621e-02  1.451309e-02  2.692123e+00 -1.744708e+01
##            rm           age           dis           rad           tax
##  3.820574e+00  2.723102e-11 -1.467646e+00  2.967960e-01 -1.186796e-02
##       ptratio             b         lstat
## -9.479889e-01  9.270514e-03 -5.234585e-01
```

```
Y_esti<-predict.lars(modlasso,X,type="fit",mode="lambda",s=0.0996448)
Y_esti<-Y_esti$fit
#data.frame(Y_esti,Y)
print("residual standard error")
```

```
## [1] "residual standard error"
```

```
Non_biased_residual(Y,Y_esti,13)
```

```
## [1] 4.735837
```

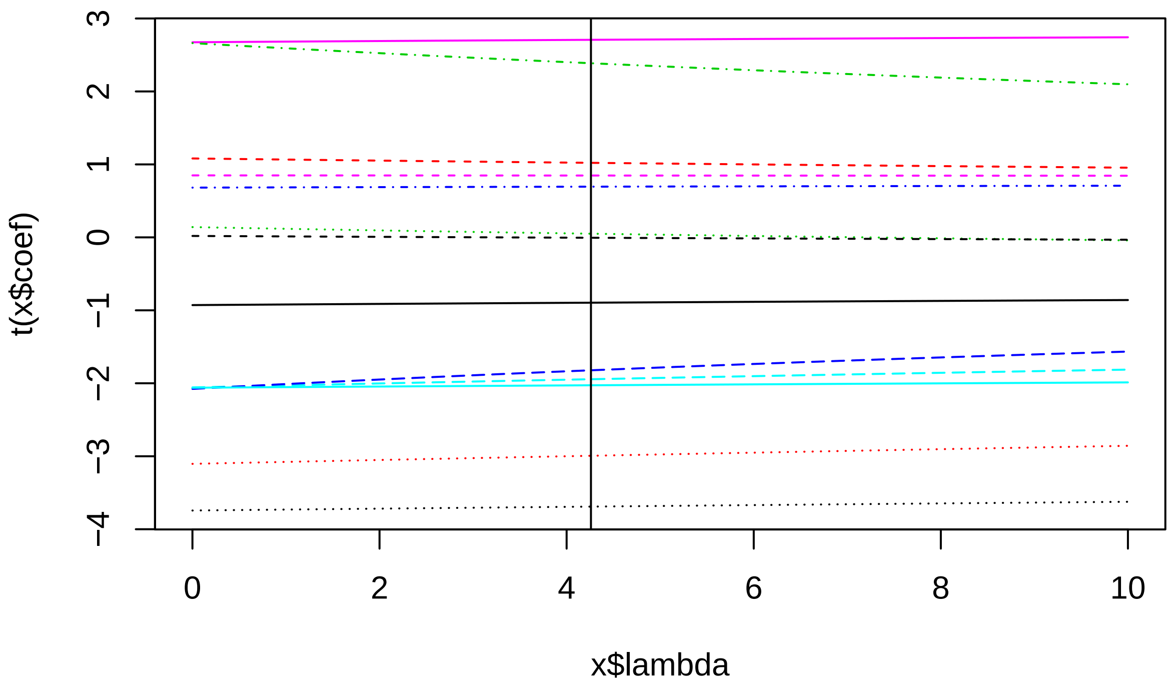
RIDGE

```
library(MASS)
```

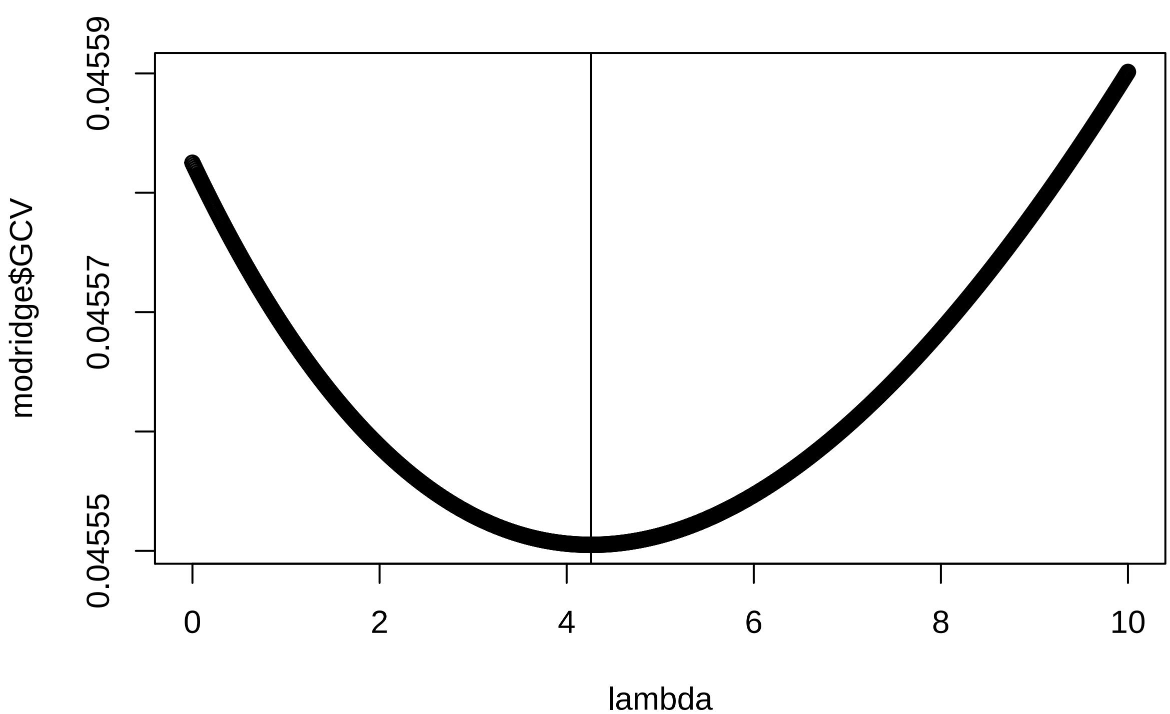
```
## Warning: package 'MASS' was built under R version 3.4.4
```

```r
modridge<-lm.ridge(medv~.,data=BostonHousing,lambda=seq(0,10,0.01))
plot(modridge)
lambda<-modridge$lambda[which.min(modridge$GCV)]

abline(v=lambda)
```




```r
plot(x=seq(0,10,0.01),modridge$GCV,xlab = "lambda")
abline(v=lambda)
```



For the ridge regression, with the smallest GCV, wo choose the lambda which is 4.26. So we can use the regression model whose lambda equals 4.26.

```
modridge<-lm.ridge(medv~.,data=BostonHousing,lambda=lambda)

coef<-coef(modridge)
coef
```

```
##                        crim            zn         indus         chas1
##  3.495372e+01 -1.041870e-01  4.384158e-02  7.326148e-03  2.738093e+00
##           nox            rm           age           dis           rad
## -1.679498e+01  3.857388e+00 -1.932605e-04 -1.422995e+00  2.743521e-01
##           tax       ptratio             b         lstat
## -1.081962e-02 -9.372977e-01  9.291544e-03 -5.172556e-01
```

```
un<-matrix(1,nrow=length(Y),ncol=1)
Y_esti<-cbind(un,X)%*%as.vector(coef)
Non_biased_residual(Y,Y_esti,13)
```

```
## [1] 4.737444
```

So we obtain the result.

What's more, I think about how about it with the new data.

```
smp1<-sample(nrow(BostonHousing), nrow(BostonHousing)*0.75)
train_data=BostonHousing[smp1,]
test_data=BostonHousing[-smp1,]
```

With linear regression

```
Y_esti<-predict(reg,newdata=test_data)
Y_test<-test_data$medv
Non_biased_residual(Y_test,Y_esti,13)
```

```
##        2
## 5.627472
```

LASSO

```
Y<-as.matrix(train_data$medv)
X<-apply(as.matrix(subset(train_data,select=-medv)),2,as.numeric)
modlasso=lars(x=X,y=Y,type="lasso")
X_test<-apply(as.matrix(subset(test_data,select=-medv)),2,as.numeric)
Y_esti<-predict.lars(modlasso,X_test,type="fit",mode="lambda",s=modlasso$lambda[which.min(modlasso$RSS)
Y_esti<-Y_esti$fit
Y_test<-test_data$medv
Non_biased_residual(Y_test,Y_esti,13)
```

```
## [1] 5.637635
```

Ridge

```
modridge<-lm.ridge(medv~.,data=train_data,lambda=seq(0,10,0.01))
lambda<-modridge$lambda[which.min(modridge$GCV)]
```

For the ridge regression, with the smallest GCV, wo choose the lambda which is 4.26. So we can use the regression model whose lambda equals 4.26.

```
modridge<-lm.ridge(medv~.,data=train_data,lambda=lambda)
X_test<-apply(as.matrix(subset(test_data,select=-medv)),2,as.numeric)
coef<-coef(modridge)
Y_test<-test_data$medv
un<-matrix(1,nrow=length(Y_test),ncol=1)
Y_esti<-cbind(un,X_test)%*%as.vector(coef)
Non_biased_residual(Y_test,Y_esti,13)
```

```
## [1] 5.639929
```

That's all. I find that for these new data, the linear regression and the ridge regression is better than Lasso regression. in general, ridge regression fit the new data better than linear regression. So I predict if the numbers of data is bigger, ridge regression will have the residual standard error.