hw3

Tinglei Wu

2/14/2022

```
library('splines')        ## for 'bs'
library('dplyr')          ## for 'select', 'filter', and others
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library('magrittr')       ## for '%<>%' operator
library('glmnet')         ## for 'glmnet'
```

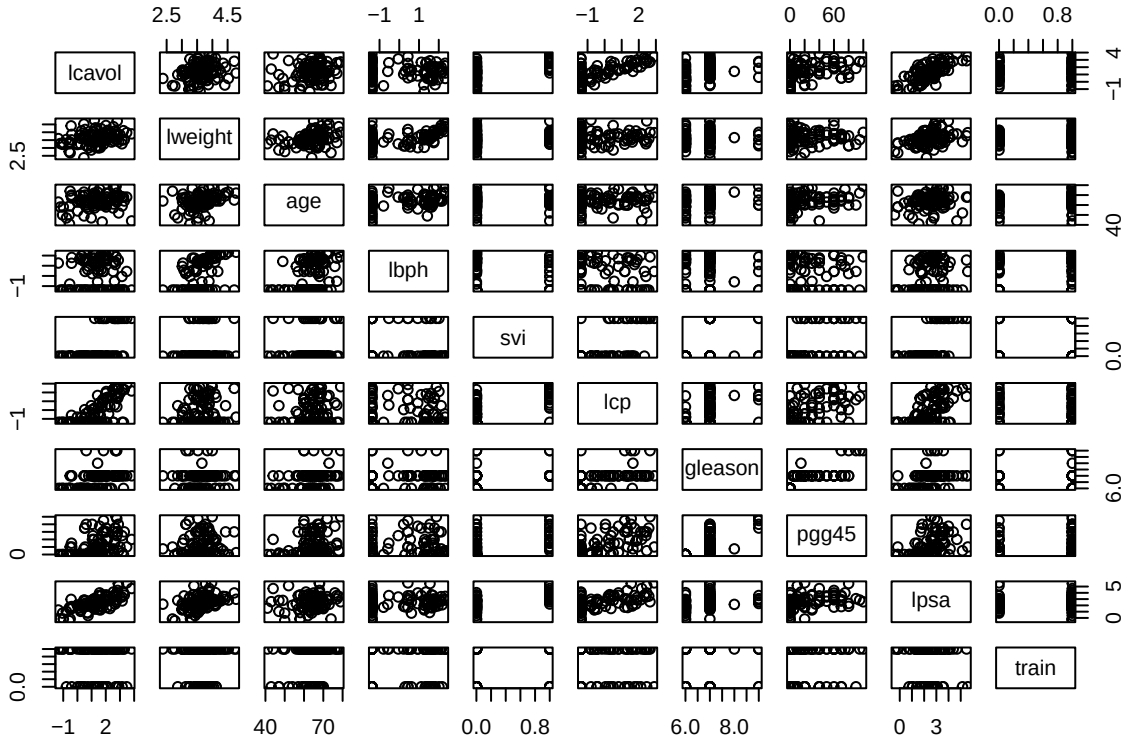
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-3
```

```
###  Linear regression examples ###

## load prostate data
prostate <-
  read.table(url(
    'https://web.stanford.edu/~hastie/ElemStatLearn/datasets/prostate.data'))

pairs(prostate)
```




```
## split prostate into testing and training subsets
prostate_train <- prostate %>%
  filter(train == TRUE) %>%
  select(-train)

summary(prostate_train)
```

```
##      lcavol           lweight           age              lbph        
##  Min.   :-1.3471   Min.   :2.375   Min.   :41.00   Min.   :-1.38629  
##  1st Qu.: 0.4883   1st Qu.:3.330   1st Qu.:61.00   1st Qu.:-1.38629  
##  Median : 1.4679   Median :3.599   Median :65.00   Median :-0.05129  
##  Mean   : 1.3135   Mean   :3.626   Mean   :64.75   Mean   : 0.07144  
##  3rd Qu.: 2.3491   3rd Qu.:3.884   3rd Qu.:69.00   3rd Qu.: 1.54751  
##  Max.   : 3.8210   Max.   :4.780   Max.   :79.00   Max.   : 2.32630  
##       svi              lcp              gleason          pgg45       
##  Min.   :0.0000   Min.   :-1.3863   Min.   :6.000   Min.   :  0.00  
##  1st Qu.:0.0000   1st Qu.:-1.3863   1st Qu.:6.000   1st Qu.:  0.00  
##  Median :0.0000   Median :-0.7985   Median :7.000   Median : 15.00  
##  Mean   :0.2239   Mean   :-0.2142   Mean   :6.731   Mean   : 26.27  
##  3rd Qu.:0.0000   3rd Qu.: 0.9948   3rd Qu.:7.000   3rd Qu.: 50.00  
##  Max.   :1.0000   Max.   : 2.6568   Max.   :9.000   Max.   :100.00  
##       lpsa        
##  Min.   :-0.4308  
##  1st Qu.: 1.6673  
##  Median : 2.5688  
##  Mean   : 2.4523  
##  3rd Qu.: 3.3652  
##  Max.   : 5.4775
```

```
prostate_test <- prostate %>%
  filter(train == FALSE) %>%
  select(-train)
```

##Question 2:

```
cor(prostate_train)
```

```
##             lcavol    lweight       age         lbph        svi         lcp
## lcavol  1.00000000 0.30023199 0.2863243  0.06316772  0.5929491  0.69204308
## lweight 0.30023199 1.00000000 0.3167235  0.43704154  0.1810545  0.15682859
## age     0.28632427 0.31672347 1.0000000  0.28734645  0.1289023  0.17295140
## lbph    0.06316772 0.43704154 0.2873464  1.00000000 -0.1391468 -0.08853456
## svi     0.59294913 0.18105448 0.1289023 -0.13914680  1.0000000  0.67124021
## lcp     0.69204308 0.15682859 0.1729514 -0.08853456  0.6712402  1.00000000
## gleason 0.42641407 0.02355821 0.3659151  0.03299215  0.3068754  0.47643684
## pgg45   0.48316136 0.07416632 0.2758057 -0.03040382  0.4813577  0.66253335
## lpsa    0.73315515 0.48521519 0.2276424  0.26293763  0.5568864  0.48920320
##             gleason       pgg45      lpsa
## lcavol  0.42641407  0.48316136 0.7331551
## lweight 0.02355821  0.07416632 0.4852152
## age     0.36591512  0.27580573 0.2276424
## lbph    0.03299215 -0.03040382 0.2629376
## svi     0.30687537  0.48135774 0.5568864
## lcp     0.47643684  0.66253335 0.4892032
## gleason 1.00000000  0.75705650 0.3424278
## pgg45   0.75705650  1.00000000 0.4480480
## lpsa    0.34242781  0.44804795 1.0000000
```

##Question 3,4:

```
lcavol_out<- lm(lcavol ~ ., data=prostate_train)
lcavol_out
```

```
##
## Call:
## lm(formula = lcavol ~ ., data = prostate_train)
##
## Coefficients:
## (Intercept)      lweight          age         lbph          svi          lcp
##   -2.173357    -0.113370     0.020102    -0.056981     0.035116     0.418455
##     gleason        pgg45         lpsa
##    0.224387    -0.009113     0.575455
```

##Question 5:

```
L2_loss <- function(y, yhat)
  (y-yhat)^2
err <- function(dat, fit, loss=L2_loss)
  mean(loss(dat$lcavol, predict(fit, newdata=dat)))
## training error
err(prostate_train, lcavol_out)
```

```
## [1] 0.4383709
```

```
## testing error
err(prostate_test, lcavol_out)
```

```
## [1] 0.5084068
```

##Question 6:

```
form  <- lcavol ~  lweight + age + lbph + lcp + pgg45 + lpsa + svi + gleason
x_inp <- model.matrix(form, data=prostate_train)
y_out <- prostate_train$lcavol
glmnet <- glmnet(x=x_inp, y=y_out, lambda=seq(0.8, 0, -0.05), alpha = 0)
print(glmnet$beta)
```

```
## 9 x 17 sparse Matrix of class "dgCMatrix"

##    [[ suppressing 17 column names 's0', 's1', 's2' ... ]]

##
## (Intercept)  .            .            .            .            .
## lweight      0.126233723  0.123269298  0.119863278  0.1158716819  0.1111931576
## age          0.011482623  0.011670731  0.011877123  0.0121010799  0.0123455384
## lbph        -0.007449919 -0.008367575 -0.009363254 -0.0104653587 -0.0116905450
## lcp          0.207036526  0.211463094  0.216338206  0.2216006088  0.2273130852
## pgg45        0.001473672  0.001329880  0.001167084  0.0009827841  0.0007732439
## lpsa         0.290917892  0.298034308  0.305629735  0.3138070249  0.3226494596
## svi          0.394617510  0.393975652  0.392543932  0.3903246504  0.3871479640
## gleason      0.132832538  0.133493765  0.134175184  0.1349324324  0.1357928799
##
## (Intercept)  .             .             .              .
## lweight      0.1057367423  0.0992845548  9.167569e-02  0.0826683974
## age          0.0126162948  0.0129137196  1.324527e-02  0.0136179442
## lbph        -0.0130587545 -0.0145982826 -1.633988e-02 -0.0183250234
## lcp          0.2336122365  0.2404842462  2.481056e-01  0.2566365841
## pgg45        0.0005320808  0.0002563424 -6.281817e-05 -0.0004352185
## lpsa         0.3322362737  0.3427336970  3.542772e-01  0.3670625821
## svi          0.3827345083  0.3769402232  3.693573e-01  0.3595330231
## gleason      0.1368039677  0.1380061651  1.394762e-01  0.1413173664
##
## (Intercept)  .             .             .             .             .
## lweight      0.0719539790  0.059127616  0.043652593  0.024777969  0.001504802
## age          0.0140406458  0.014526957  0.015088490  0.015748487  0.016532948
## lbph        -0.0206075810 -0.023258103 -0.026377963 -0.030098852 -0.034621150
## lcp          0.2662889676  0.277447149  0.290342311  0.305728439  0.324372008
## pgg45       -0.0008738898 -0.001398912 -0.002031353 -0.002810371 -0.003788173
## lpsa         0.3813402190  0.397429712  0.415786556  0.437009864  0.461951799
## svi          0.3468674177  0.330415198  0.309283880  0.281608260  0.245177911
## gleason      0.1436779613  0.146778188  0.150949425  0.156678907  0.164800413
##
## (Intercept)  .            .            .
## lweight     -0.027603986 -0.064680201 -0.113137304
```

```
## age          0.017480107  0.018643148  0.020098181
## lbph        -0.040241264 -0.047425776 -0.056962692
## lcp          0.347616547  0.377657417  0.418431830
## pgg45       -0.005050263 -0.006739814 -0.009116838
## lpsa         0.491849702  0.528596455  0.575318051
## svi          0.196427346  0.129711598  0.035342349
## gleason      0.176722769  0.194999807  0.224585243
```

```
## functions to compute testing error with glmnet
error <- function(dat, glmnet, lam, form, loss=L2_loss) {
  x_inp <- model.matrix(form, data=dat)
  y_out <- dat$lcavol
  y_hat <- predict(glmnet, newx=x_inp, s=lam)  ## see predict.elnet
  mean(loss(y_out, y_hat))
}

## training error at lambda=0.01
error(prostate_train, glmnet, lam=0.01, form=form)
```

```
## [1] 0.4385064
```

```
## testing error at lambda=0.01
error(prostate_test, glmnet, lam=0.01, form=form)
```

```
## [1] 0.5047688
```

```
## training error at lambda=0.1
error(prostate_train, glmnet, lam=0.1, form=form)
```

```
## [1] 0.4486907
```

```
## testing error at lambda=0.1
error(prostate_test, glmnet, lam=0.1, form=form)
```

```
## [1] 0.4914336
```

```
## training error at lambda=0.2
error(prostate_train, glmnet, lam=0.2, form=form)
```

```
## [1] 0.4653812
```

```
## testing error at lambda=0.2
error(prostate_test, glmnet, lam=0.2, form=form)
```

```
## [1] 0.4945718
```

```
## training error at lambda=0.3
error(prostate_train, glmnet, lam=0.3, form=form)
```

```
## [1] 0.481922
```

```
## testing error at lambda=0.3
error(prostate_test, glmnet, lam=0.3, form=form)
```

```
## [1] 0.5015496
```

```
## training error at lambda=0.4
error(prostate_train, glmnet, lam=0.4, form=form)
```

```
## [1] 0.4973256
```

```
## testing error at lambda=0.4
error(prostate_test, glmnet, lam=0.4, form=form)
```

```
## [1] 0.5088406
```

```
## training error at lambda=0.5
error(prostate_train, glmnet, lam=0.5, form=form)
```

```
## [1] 0.5115869
```

```
## testing error at lambda=0.5
error(prostate_test, glmnet, lam=0.5, form=form)
```

```
## [1] 0.5156281
```

```
## training error at lambda=0.6
error(prostate_train, glmnet, lam=0.6, form=form)
```

```
## [1] 0.524921
```

```
## testing error at lambda=0.6
error(prostate_test, glmnet, lam=0.6, form=form)
```

```
## [1] 0.5218096
```

```
## training error at lambda=0.7
error(prostate_train, glmnet, lam=0.7, form=form)
```

```
## [1] 0.5374869
```

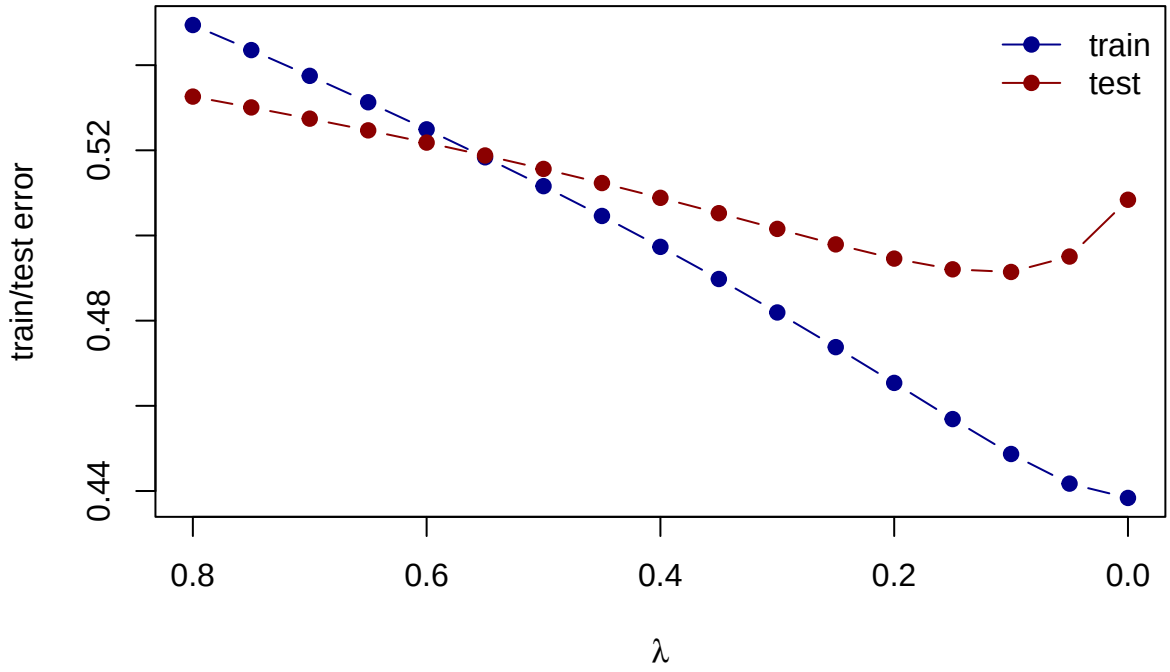
```
## testing error at lambda=0.7
error(prostate_test, glmnet, lam=0.7, form=form)
```

```
## [1] 0.5274336
```

##Question 7:

```
err_train_1 <- sapply(glmnet$lambda, function(lam)
  error(prostate_train, glmnet, lam, form))
err_test_1 <- sapply(glmnet$lambda, function(lam)
  error(prostate_test, glmnet, lam, form))

## plot test/train error
plot(x=range(glmnet$lambda),
     y=range(c(err_train_1, err_test_1)),
     xlim=rev(range(glmnet$lambda)),
     type='n',
     xlab=expression(lambda),
     ylab='train/test error')
points(glmnet$lambda, err_train_1, pch=19, type='b', col='darkblue')
points(glmnet$lambda, err_test_1, pch=19, type='b', col='darkred')
legend('topright', c('train','test'), lty=1, pch=19,
       col=c('darkblue','darkred'), bty='n')
```



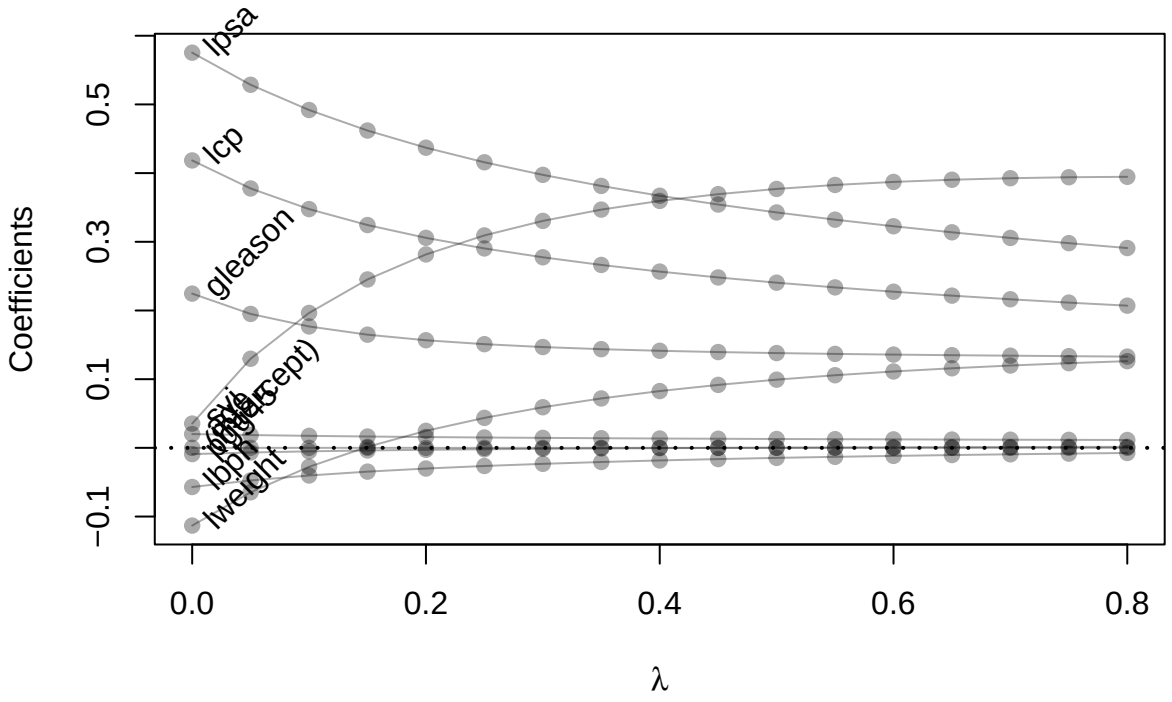

```
colnames(glmnet$beta) <- paste('lam =', glmnet$lambda)
print(glmnet$beta %>% as.matrix)
```

```
##                lam = 0.8   lam = 0.75    lam = 0.7    lam = 0.65     lam = 0.6
## (Intercept)  0.000000000  0.000000000  0.000000000  0.0000000000  0.0000000000
## lweight      0.126233723  0.123269298  0.119863278  0.1158716819  0.1111931576
## age          0.011482623  0.011670731  0.011877123  0.0121010799  0.0123455384
## lbph        -0.007449919 -0.008367575 -0.009363254 -0.0104653587 -0.0116905450
## lcp          0.207036526  0.211463094  0.216338206  0.2216006088  0.2273130852
## pgg45        0.001473672  0.001329880  0.001167084  0.0009827841  0.0007732439
## lpsa         0.290917892  0.298034308  0.305629735  0.3138070249  0.3226494596
## svi          0.394617510  0.393975652  0.392543932  0.3903246504  0.3871479640
## gleason      0.132832538  0.133493765  0.134175184  0.1349324324  0.1357928799
##                lam = 0.55     lam = 0.5    lam = 0.45     lam = 0.4
```

```
## (Intercept)  0.0000000000  0.0000000000  0.000000e+00  0.0000000000
## lweight      0.1057367423  0.0992845548  9.167569e-02  0.0826683974
## age          0.0126162948  0.0129137196  1.324527e-02  0.0136179442
## lbph        -0.0130587545 -0.0145982826 -1.633988e-02 -0.0183250234
## lcp          0.2336122365  0.2404842462  2.481056e-01  0.2566365841
## pgg45        0.0005320808  0.0002563424 -6.281817e-05 -0.0004352185
## lpsa         0.3322362737  0.3427336970  3.542772e-01  0.3670625821
## svi          0.3827345083  0.3769402232  3.693573e-01  0.3595330231
## gleason      0.1368039677  0.1380061651  1.394762e-01  0.1413173664
##                lam = 0.35    lam = 0.3   lam = 0.25    lam = 0.2   lam = 0.15
## (Intercept)  0.0000000000  0.000000000  0.000000000  0.000000000  0.000000000
## lweight      0.0719539790  0.059127616  0.043652593  0.024777969  0.001504802
## age          0.0140406458  0.014526957  0.015088490  0.015748487  0.016532948
## lbph        -0.0206075810 -0.023258103 -0.026377963 -0.030098852 -0.034621150
## lcp          0.2662889676  0.277447149  0.290342311  0.305728439  0.324372008
## pgg45       -0.0008738898 -0.001398912 -0.002031353 -0.002810371 -0.003788173
## lpsa         0.3813402190  0.397429712  0.415786556  0.437009864  0.461951799
## svi          0.3468674177  0.330415198  0.309283880  0.281608260  0.245177911
## gleason      0.1436779613  0.146778188  0.150949425  0.156678907  0.164800413
##                lam = 0.1   lam = 0.05      lam = 0
## (Intercept)  0.000000000  0.000000000  0.000000000
## lweight     -0.027603986 -0.064680201 -0.113137304
## age          0.017480107  0.018643148  0.020098181
## lbph        -0.040241264 -0.047425776 -0.056962692
## lcp          0.347616547  0.377657417  0.418431830
## pgg45       -0.005050263 -0.006739814 -0.009116838
## lpsa         0.491849702  0.528596455  0.575318051
## svi          0.196427346  0.129711598  0.035342349
## gleason      0.176722769  0.194999807  0.224585243
```

##Question 8:

```
plot(x=range(glmnet$lambda),
     y=range(as.matrix(glmnet$beta)),
     type='n',
     xlab=expression(lambda),
     ylab='Coefficients')
for(i in 1:nrow(glmnet$beta)) {
  points(x=glmnet$lambda, y=glmnet$beta[i,], pch=19, col='#00000055')
  lines(x=glmnet$lambda, y=glmnet$beta[i,], col='#00000055')
}
text(x=0, y=glmnet$beta[,ncol(glmnet$beta)],
     labels=rownames(glmnet$beta),
     xpd=NA, pos=4, srt=45)
abline(h=0, lty=3, lwd=2)
```


lpsa
lcp
gleason
(Intercept)
lweight
Coefficients
0.5
0.3
0.1
−0.1
0.0
0.2
0.4
0.6
0.8
λ