Lasso

Muyao GUO

2025-10-19

```r
rm(list=objects())

don=read.table("wdbc.data",sep=",",header=F)
dim(don) # 569  32
```

```
## [1] 569  32
```

```r
don=don[,-1]
names(don)[1]="Y"
don$Y=factor(don$Y,labels=c("0", "1")) # on peut garder "B" et M"
head(don)
```

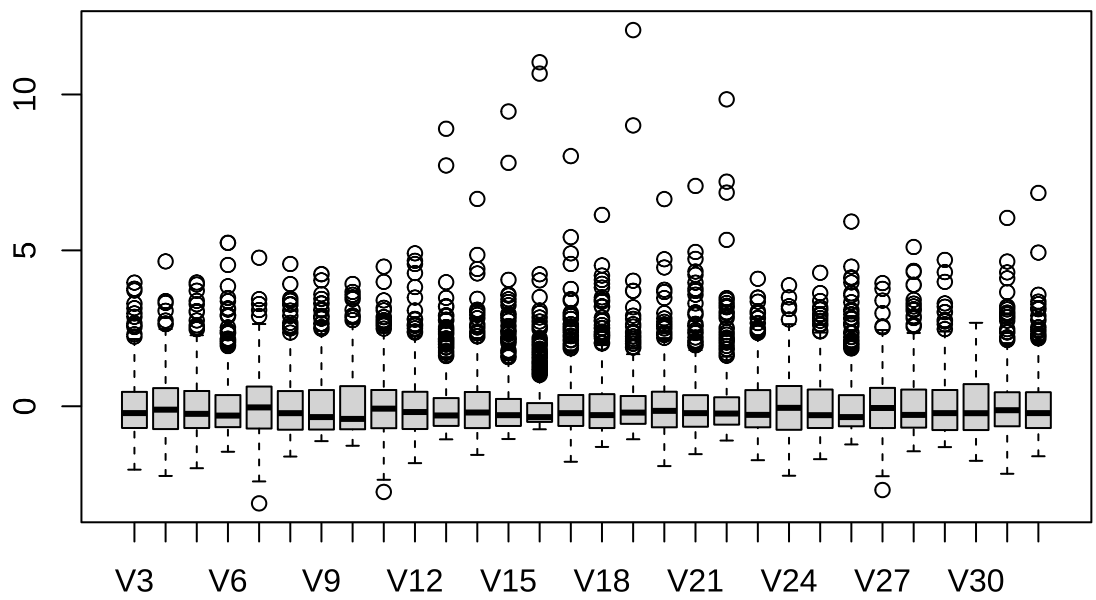
```
##   Y    V3    V4     V5     V6      V7      V8     V9     V10    V11     V12
## 1 1 17.99 10.38 122.80 1001.0 0.11840 0.27760 0.3001 0.14710 0.2419 0.07871
## 2 1 20.57 17.77 132.90 1326.0 0.08474 0.07864 0.0869 0.07017 0.1812 0.05667
## 3 1 19.69 21.25 130.00 1203.0 0.10960 0.15990 0.1974 0.12790 0.2069 0.05999
## 4 1 11.42 20.38  77.58  386.1 0.14250 0.28390 0.2414 0.10520 0.2597 0.09744
## 5 1 20.29 14.34 135.10 1297.0 0.10030 0.13280 0.1980 0.10430 0.1809 0.05883
## 6 1 12.45 15.70  82.57  477.1 0.12780 0.17000 0.1578 0.08089 0.2087 0.07613
##      V13    V14   V15    V16      V17     V18     V19     V20     V21      V22
## 1 1.0950 0.9053 8.589 153.40 0.006399 0.04904 0.05373 0.01587 0.03003 0.006193
## 2 0.5435 0.7339 3.398  74.08 0.005225 0.01308 0.01860 0.01340 0.01389 0.003532
## 3 0.7456 0.7869 4.585  94.03 0.006150 0.04006 0.03832 0.02058 0.02250 0.004571
## 4 0.4956 1.1560 3.445  27.23 0.009110 0.07458 0.05661 0.01867 0.05963 0.009208
## 5 0.7572 0.7813 5.438  94.44 0.011490 0.02461 0.05688 0.01885 0.01756 0.005115
## 6 0.3345 0.8902 2.217  27.19 0.007510 0.03345 0.03672 0.01137 0.02165 0.005082
##     V23   V24    V25    V26    V27    V28    V29    V30    V31     V32
## 1 25.38 17.33 184.60 2019.0 0.1622 0.6656 0.7119 0.2654 0.4601 0.11890
## 2 24.99 23.41 158.80 1956.0 0.1238 0.1866 0.2416 0.1860 0.2750 0.08902
## 3 23.57 25.53 152.50 1709.0 0.1444 0.4245 0.4504 0.2430 0.3613 0.08758
## 4 14.91 26.50  98.87  567.7 0.2098 0.8663 0.6869 0.2575 0.6638 0.17300
## 5 22.54 16.67 152.20 1575.0 0.1374 0.2050 0.4000 0.1625 0.2364 0.07678
## 6 15.47 23.75 103.40  741.6 0.1791 0.5249 0.5355 0.1741 0.3985 0.12440
```

```r
summary(don$Y)
```

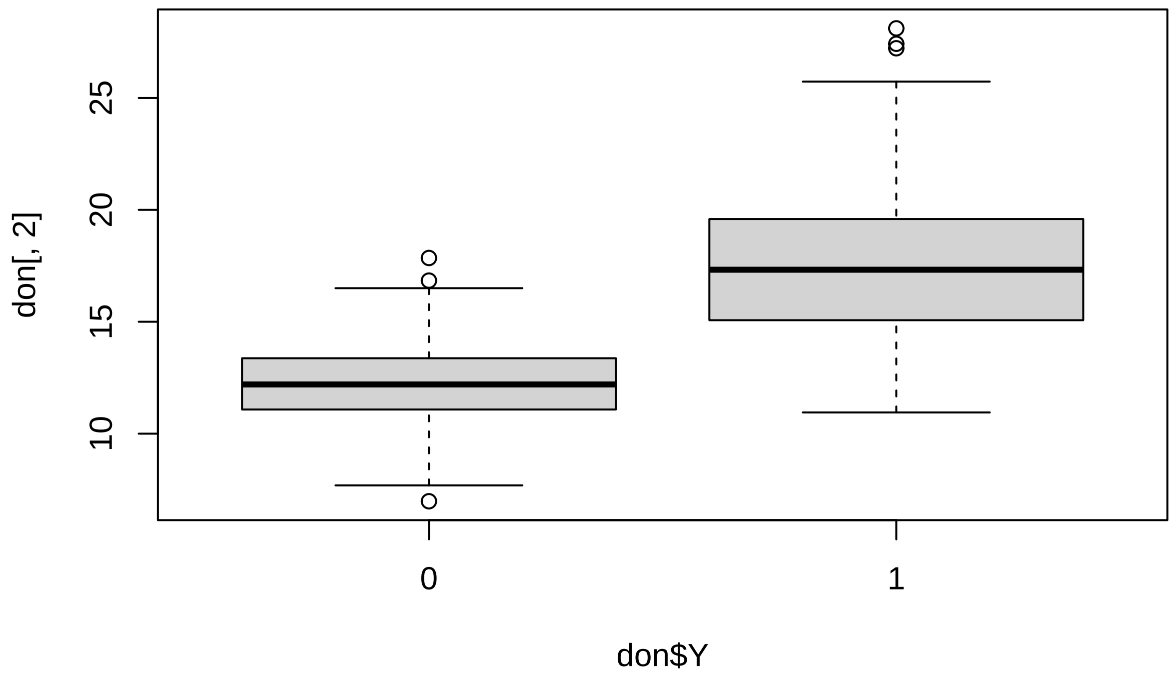
```
##   0   1
## 357 212
```

```
# B   M
# 357 212

boxplot(scale(don[,-1]))
```




```
boxplot(don[,2]~don$Y)
```




```r
## Apprentissage/test
set.seed(12345)
test = sample(1:length(don$Y),200)
train = -test
train = don[train, ] #369 observations
test = don[test,]
table(train$Y)
```

```
##
##   0   1
## 230 139
```

```r
# reg logistique
fit.glm=glm(Y~.,family=binomial,data=train);summary(fit.glm)
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Call:
## glm(formula = Y ~ ., family = binomial, data = train)
##
## Coefficients:
```

```
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.132e+03  9.953e+05  -0.001    0.999
## V3          -2.656e+02  2.698e+05  -0.001    0.999
## V4           9.649e+00  7.818e+03   0.001    0.999
## V5           4.611e+01  5.075e+04   0.001    0.999
## V6          -4.592e-01  1.237e+03   0.000    1.000
## V7           4.136e+03  3.553e+06   0.001    0.999
## V8          -5.135e+03  1.954e+06  -0.003    0.998
## V9           6.377e+02  8.711e+05   0.001    0.999
## V10          2.777e+02  3.311e+06   0.000    1.000
## V11         -1.434e+03  5.140e+05  -0.003    0.998
## V12          1.089e+04  7.568e+06   0.001    0.999
## V13          8.376e+02  3.809e+05   0.002    0.998
## V14         -1.431e-02  5.391e+04   0.000    1.000
## V15          2.952e+01  3.989e+04   0.001    0.999
## V16         -7.041e+00  4.376e+03  -0.002    0.999
## V17          3.880e+03  4.936e+06   0.001    0.999
## V18          5.133e+03  4.183e+06   0.001    0.999
## V19         -4.533e+03  1.997e+06  -0.002    0.998
## V20          1.091e+03  4.090e+06   0.000    1.000
## V21         -4.646e+03  3.404e+06  -0.001    0.999
## V22          1.818e+03  1.738e+07   0.000    1.000
## V23         -1.334e+01  9.223e+04   0.000    1.000
## V24          2.140e+00  7.580e+03   0.000    1.000
## V25         -3.879e+00  5.073e+03  -0.001    0.999
## V26          7.628e-01  8.084e+02   0.001    0.999
## V27         -8.221e+02  1.300e+06  -0.001    0.999
## V28         -3.597e+01  6.468e+05   0.000    1.000
## V29          6.288e+02  3.378e+05   0.002    0.999
## V30          1.374e+03  6.843e+05   0.002    0.998
## V31          1.377e+03  4.793e+05   0.003    0.998
## V32         -4.422e+03  2.675e+06  -0.002    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4.8887e+02  on 368  degrees of freedom
## Residual deviance: 1.3191e-07  on 338  degrees of freedom
## AIC: 62
##
## Number of Fisher Scoring iterations: 25
```

```r
# non convergence de l'algo

# les 30 covariables représentent les moyennes, écart-types et max
# de 10 features (voir wdbc.txt): il est probable qu'il y ait une
# colinéarité des covariables, la design matrix n'est pas de plein rang

# si on ne considère que les 10 premières covariables, pas de pb de convergence
fit1=glm(Y~V3+V4+V5+V6+V7+V8+V9+V10+V11+V12,family=binomial,data=train);summary(fit1)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
```

```
## Call:
## glm(formula = Y ~ V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 + V11 +
##     V12, family = binomial, data = train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -13.80871   16.45031  -0.839   0.4012
## V3            0.50270    5.41627   0.093   0.9261
## V4            0.48834    0.09287   5.258 1.45e-07 ***
## V5           -0.50361    0.73248  -0.688   0.4917
## V6            0.04389    0.02065   2.126   0.0335 *
## V7           83.07329   38.53036   2.156   0.0311 *
## V8           -9.59541   26.85511  -0.357   0.7209
## V9           11.90201    9.88574   1.204   0.2286
## V10          93.53589   37.76797   2.477   0.0133 *
## V11          27.59167   12.72113   2.169   0.0301 *
## V12         -28.78779  113.50325  -0.254   0.7998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 488.868  on 368  degrees of freedom
## Residual deviance:  90.781  on 358  degrees of freedom
## AIC: 112.78
##
## Number of Fisher Scoring iterations: 9
```

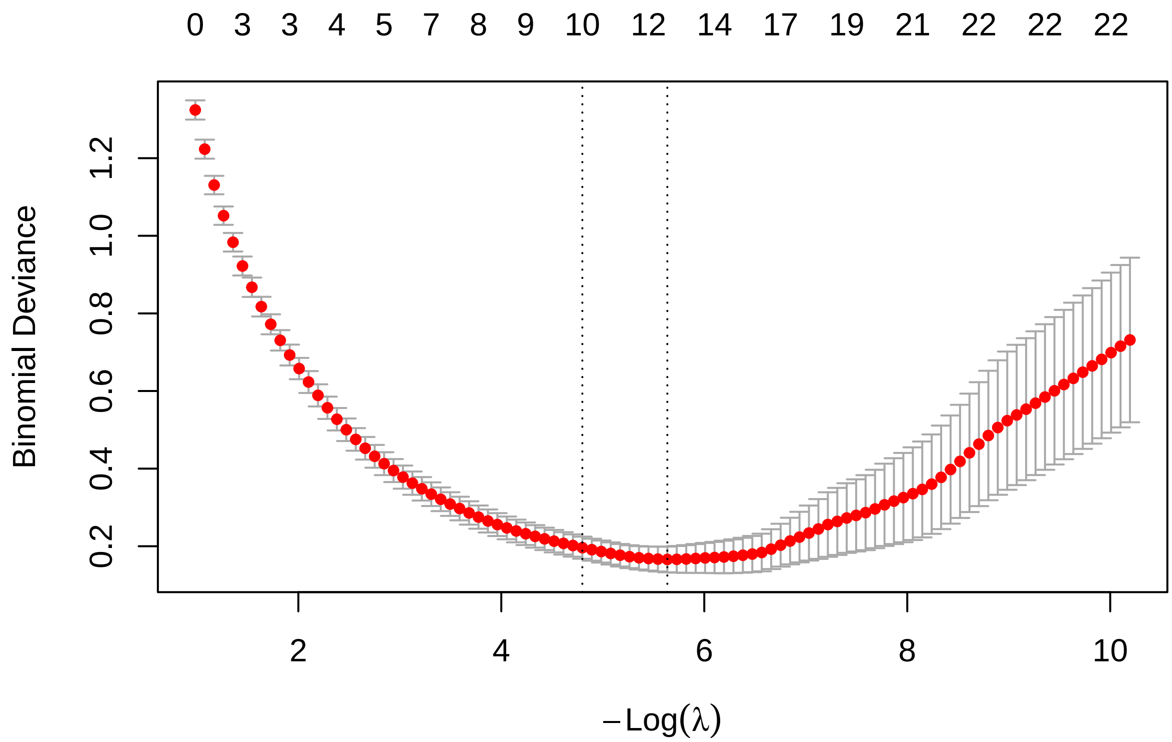
```r
## LASSO
xtrain=model.matrix (Y~.,train )[,-1]
ytrain=train$Y
xtest=model.matrix (Y~.,test )[,-1]
ytest=test$Y

library(glmnet)
```

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

```
## Loaded glmnet 4.1-10
```

```r
cv.out =cv.glmnet (xtrain,ytrain,alpha=1,family="binomial")
plot(cv.out)
```




```
bestlam =cv.out$lambda.min
bestlam
```

```
## [1] 0.00356702
```

```
# [1] 0.00356702 #log(bestlam) : -5.636025
fit.lasso=glmnet(xtrain,ytrain,family="binomial" ,alpha =1)
# help("predict.glmnet")
pred.lasso=predict(fit.lasso,s=bestlam,newx=xtest,type="response")

predict(fit.lasso,type="coefficients",s=bestlam)
```

```
## 31 x 1 sparse Matrix of class "dgCMatrix"
##              s=0.00356702
## (Intercept)  -33.17384495
## V3             .
## V4             0.09016524
## V5             .
## V6             .
## V7             .
## V8             .
## V9             .
## V10           14.21269344
## V11            .
## V12            .
```

```
## V13            8.63307248
## V14            .
## V15            .
## V16            .
## V17           21.61112530
## V18          -11.68481423
## V19            .
## V20            .
## V21           -9.72834497
## V22         -101.79763780
## V23            0.80800303
## V24            0.18420057
## V25            .
## V26            .
## V27           34.74873370
## V28            .
## V29            3.07229214
## V30           13.58937537
## V31           10.08673548
## V32            .
```

```
# certains paramètres sont estimés à 0 --> sélection de variables

length(predict(fit.lasso,type="nonzero",s=bestlam)[,1])
```

```
## [1] 13
```

```
# 13 paramètres estimés non nuls

# classification
glm_pred=rep("0",length(test$Y))
glm_pred[pred.lasso>0.5]="1"
table(test$Y,glm_pred)
```

```
##    glm_pred
##       0   1
##   0 125   2
##   1   3  70
```

```
# taux de mal classés sur données test
mean(glm_pred!=test$Y)
```

```
## [1] 0.025
```

```
# [1] 0.025

pred.lasso=predict(fit.lasso,s=bestlam,newx=xtest,type="class")
mean(pred.lasso!=test$Y)
```

```
## [1] 0.025
```

```r
# idem

# package randomForest
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```r
fit.rf=randomForest(Y~.,data=train)
# avec les paramètres par défaut, ntree=500 et mtry=5 (savoir dire pourquoi)
y.rf=predict(fit.rf,newdata=test,type="class")
mean(y.rf!=test$Y)
```

```
## [1] 0.04
```

```r
# [1] 0.04
```