Assignment 3

Computational Statistics

12/2019

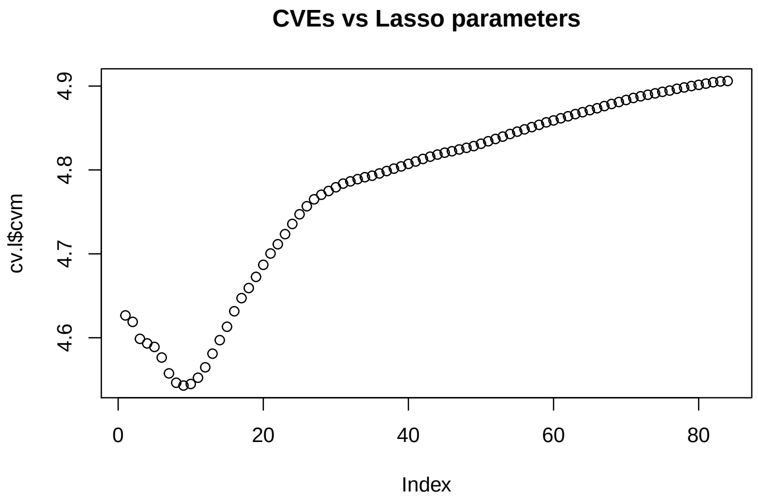
Names	SNR	ANR
Fernando Catala	2048042	u270800
Francesco Lauria	2041074	u819639
Konstantinos Soiledis	2037001	u226393

Part I

Question 1

We used a LOOCV because since the number of observations is very low (24), splitting the dataset would result in very few observations for both training and testing. Then, what LOOCV does is to train on 23 observations and validate on the remaining one, and iterates this 24 times. The low number of observations allows us to afford a computation of k=n, which otherwise would take very long. Also, LOOCV has the benefit of having low bias, since it doesn't overestimate the test set's MSE.

Plotting the CVE against the different values of lasso tuning parameters:



The optimal lasso parameter value is 11.48. The number of non-zero coefficients is 46.

The way we obtained the optimal λ value was through a 24 fold LOOCV. We tried a value of λ, for each run of LOOCV, and eventually we got the λ that corresponds to the lowest MSE. In other words, glmnet tries by default different values of λ for each run of LOOCV.

Question 2

The estimated prediction error of 2008 data is 3.12. We fitted the model with the optimal λ , and then used that model to predict the class. Finally we calculated the MSE.

Question 3

The estimated prediction error of the 2007 data is 5.36.

Part II

Question 1

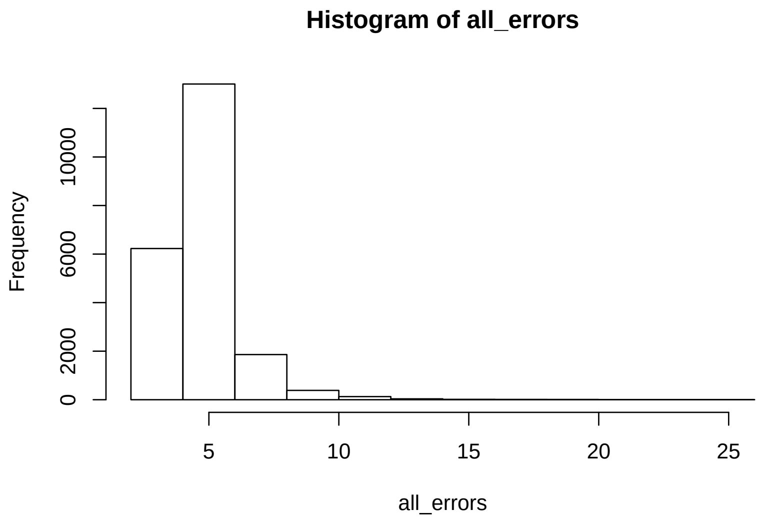
The amount of regression models we need to select the best model is 21699. We used the combn function five times, to create the models to be used for predicting the true value of vaccine efficacy. That function basically creates each possible combination from 1 to 5 features. Finally, we sum up the models to have the total number of models.

Question 2

The best performing models according to their CVEs were:

```
##                                                Model              MSE
## 1                                     Outcome ~ PAX8  3.2684768946448
## 2                             Outcome ~ PAX8 + FOXP3 2.91223941810903
## 3                   Outcome ~ ESRRA + IL17RA + FOXP3  2.4587329986644
## 4              Outcome ~ PAX8 + PTPN21 + CCL2 + IRF8 2.39860697291651
## 5 Outcome ~ PAX8 + PTPN21 + STAT3 + CCL2 + IRF8 2.38603377372105
```

And among them, the best performing one with the lowest CVE is the the last one Outcome ~ PAX8 + PTPN21 + STAT3 + CCL2 + IRF8, which uses 5 predictors and has a CVE of 2.38603377372105 .



Question 3

Regression coefficients:

```
##   (Intercept)          PAX8        PTPN21         STAT3          CCL2
##  7.705345e-16  1.306751e+00  4.676034e-01  2.800965e-01 -7.711971e-01
##          IRF8
## -7.931440e-01
```

Question 4

The predicted score for each of the 9 observations in the test set is in the second column, as compared to the true values in the first column:

```
##     True Predicted
## V1 -0.22      0.99
## V2  0.78     -0.32
## V3 -4.22      0.71
## V4 -0.22      0.73
## V5  4.78      1.79
## V6  0.78     -1.26
## V7  1.78     -0.45
## V8 -0.22      0.47
## V9 -3.22     -2.65
```

The overall MSE of the prediction is 5.1936736.

Question 5

The best subset selection approach favors to use the full set of variables, as this should guarantee higher R2 and lower RSS. However, it has its disadvantages:

1) Computation becomes infeasible for more than 40 variables, as the possible subsets are far over a million.

- Also, using a very large K is computationally expensive

2) Its strengths lead to a lower training error, but not necessarily to a lower test error (therefore, prediction power). The larger the number of variables, the higher the chance of finding models that fit well the training data, but do not have prediction power on the test data. Coefficient estimates may be overfitted and with high variance.

APPENDIX CODE

```
knitr::opts_chunk$set(echo = TRUE)
library(MLmetrics)
library(tidyverse)
library(caret)
library(glmnet)

#### LOAD DATA
set_2008<-read.table("input/TIV2008affyPLM.txt", header = F, sep = "",
                        dec = ".")

set_2007<-read.table("input/TIV2007affyPLM.txt", header = F, sep = "",
                     dec = ".")

outcome_2008<-read.table("input/TITER2008_centered.txt", header = FALSE, sep = "",
                         dec = ".")

outcome_2007<-read.table("input/TITER2007_centered.txt", header = FALSE, sep = "",
                         dec = ".")

set_2008 <-t(set_2008)
set_2007 <- t (set_2007)

outcome_2008 <- as.vector(t(outcome_2008))
set_2008 <- set_2008[,1:54675]
# tuning
set.seed(123)

cv.l <- cv.glmnet(set_2008, outcome_2008, alpha = 0.1, nfolds = 24)
index_min_lambda = which(cv.l$lambda == cv.l$lambda.min)
#cv.l$nzero[index_min_lambda]

#fitting
model.lasso <- glmnet(set_2008, outcome_2008, alpha = 0.10, lambda = cv.l$lambda.min)

#prediction in 2008
p1 <- predict(model.lasso, newx = set_2008)
mse1=MSE(y_pred = p1, y_true = outcome_2008) ## MSE

# prediction in 2007
p2 <-predict(model.lasso, newx = set_2007)
mse2=MSE(y_pred = p2, y_true = t(outcome_2007)) ## MSE


plot(cv.l$cvm, main = "CVEs vs Lasso parameters")
names_col = c("RFC2", "HSPA6", "PAX8", "GUCA1A", "THRA", "PTPN21",
              "CCL5", "CYP2E1", "EPHB3", "ESRRA", "IL17RA", "SERPIND1",
              "IFNGR2", "MAPK1", "IL25", "CTSG", "FOXP3", "STAT3", "CCL2",
              "IRF8")

names_col_2 = c("RFC2", "HSPA6", "PAX8", "GUCA1A", "THRA", "PTPN21",
                "CCL5", "CYP2E1", "EPHB3", "ESRRA", "IL17RA", "SERPIND1",
```

```r
                "IFNGR2", "MAPK1", "IL25", "CTSG", "FOXP3", "STAT3",
                "CCL2", "IRF8","Outcome")


Subset_2007<-read.table("input/subset2007.txt", header = FALSE, sep = "",
                        dec = ".", col.names = names_col)

Subset_2008 <- read.table("input/subset2008.txt", header = FALSE, sep = "",
                          dec = ".", col.names = names_col)

outcome_2008<-read.table("input/TITER2008_centered.txt",
                         header = FALSE, sep = "",
                         dec = ".")

outcome_2007<-read.table("input/TITER2007_centered.txt",
                         header = FALSE, sep = "",
                         dec = ".")
######### CREATE THE TRAIN: adding the outcome value
train<-cbind(Subset_2008,outcome_2008)

colnames(train)<-names_col_2

######### CREATE THE TEST
test<-Subset_2007
test$Outcome = NA

set.seed(123)

model1=combn(names_col,1,simplify = F)
model2=combn(names_col,2,simplify = F)
model3=combn(names_col,3,simplify = F)
model4=combn(names_col,4,simplify = F)
model5=combn(names_col,5,simplify = F)

total_models=length(model1)+length(model2)+length(model3)+length(model4)+length(model5)

######### COMBINE MODELS AS STRING

# from lists to matrix
model2_matrix <- matrix(unlist(model2), ncol = length(model2), byrow = F)
model2_matrix = t(model2_matrix)

model3_matrix <- matrix(unlist(model3), ncol = length(model3), byrow = F)
model3_matrix = t(model3_matrix)

model4_matrix <- matrix(unlist(model4), ncol = length(model4), byrow = F)
model4_matrix = t(model4_matrix)

model5_matrix <- matrix(unlist(model5), ncol = length(model5), byrow = F)
model5_matrix = t(model5_matrix)

# concatenate name columns
model2.2=rep(0,nrow(model2_matrix))
```

```r
# create models with two variables
for(row in 1:nrow(model2_matrix)){
  first_variable=model2_matrix[row,1]
  second_variable=model2_matrix[row,2]
  model2.2[row]=paste(first_variable,second_variable,sep=" + ")
}
model3.2=rep(0,nrow(model3_matrix))

# create models with two variables
for(row in 1:nrow(model3_matrix)){
  first_variable=model3_matrix[row,1]
  second_variable=model3_matrix[row,2]
  third_variable=model3_matrix[row,3]
  model3.2[row]=paste(first_variable,second_variable,
                      third_variable ,sep=" + ")
}
model4.2=rep(0,nrow(model4_matrix))

# create models with two variables
for(row in 1:nrow(model4_matrix)){
  first_variable=model4_matrix[row,1]
  second_variable=model4_matrix[row,2]
  third_variable=model4_matrix[row,3]
  fourth_variable=model4_matrix[row,4]
  model4.2[row]=paste(first_variable,second_variable,
                      third_variable, fourth_variable, sep=" + ")
}
model5.2=rep(0,nrow(model5_matrix))

# create models with two variables
for(row in 1:nrow(model5_matrix)){
  first_variable=model5_matrix[row,1]
  second_variable=model5_matrix[row,2]
  third_variable=model5_matrix[row,3]
  fourth_variable=model5_matrix[row,4]
  fifth_variable=model5_matrix[row,5]
  model5.2[row]=paste(first_variable,second_variable,
                      third_variable, fourth_variable,
                      fifth_variable, sep=" + ")
}
K <- 5
N <- nrow(Subset_2008)

## Create a partition vector:
part <- sample(rep(1 : K, N / K))

set.seed(123)

# --------- models with 1 feature ---------- #

## Loop over candidate models:
matrix1_cve<-matrix(nrow=length(model1),ncol=2)
for(m in 1 : length(model1)) {
  ## Loop over K repititions:
  y<-"Outcome ~ "
  model<-paste0(y,model1[m])
  mse <- c()
  for(k in 1 : K) {
    ## Partition data:
    dat0 <- train[part != k, ]
    dat1 <- train[part == k, ]
    ## Fit model and generate predictions:
    fit  <- lm(model, data = dat0)
    pred <- predict(fit, newdata = dat1)

    ## Save MSE:
    mse[k] <- MSE(y_pred = pred, y_true = dat1$Outcome)
  }
  ## Save the CVE:
  matrix1_cve[m,]<-c(model,mean(mse))
}

index=which.min(matrix1_cve[,2])
model1_min=matrix1_cve[index,]

# --------- models with 2 features ---------- #

matrix2_cve<-matrix(nrow=length(model2.2),ncol=2)
for(m in 1 : length(model2.2)) {
  ## Loop over K repititions:
  y<-"Outcome ~ "
  model<-paste0(y,model2.2[m])
  mse <- c()
  for(k in 1 : K) {
    ## Partition data:
    dat0 <- train[part != k, ]
    dat1 <- train[part == k, ]
    ## Fit model and generate predictions:
    fit  <- lm(model, data = dat0)
    pred <- predict(fit, newdata = dat1)

    ## Save MSE:
    mse[k] <- MSE(y_pred = pred, y_true = dat1$Outcome)
  }
  ## Save the CVE:
  matrix2_cve[m,]<-c(model,mean(mse))
}

index=which.min(matrix2_cve[,2])
model2_min=matrix2_cve[index,]

# --------- models with 3 features ---------- #

matrix3_cve<-matrix(nrow=length(model3.2),ncol=2)
for(m in 1 : length(model3.2)) {
  ## Loop over K repititions:
```

```r
  y<-"Outcome ~ "
  model<-paste0(y,model3.2[m])
  mse <- c()
  for(k in 1 : K) {
    ## Partition data:
    dat0 <- train[part != k, ]
    dat1 <- train[part == k, ]
    ## Fit model and generate predictions:
    fit  <- lm(model, data = dat0)
    pred <- predict(fit, newdata = dat1)

    ## Save MSE:
    mse[k] <- MSE(y_pred = pred, y_true = dat1$Outcome)
  }
  ## Save the CVE:
  matrix3_cve[m,]<-c(model,mean(mse))
}
index=which.min(matrix3_cve[,2])
model3_min=matrix3_cve[index,]

# --------- models with 4 features ---------- #

matrix4_cve<-matrix(nrow=length(model4.2),ncol=2)
for(m in 1 : length(model4.2)) {
  ## Loop over K repititions:
  y<-"Outcome ~ "
  model<-paste0(y,model4.2[m])
  mse <- c()
  for(k in 1 : K) {
    ## Partition data:
    dat0 <- train[part != k, ]
    dat1 <- train[part == k, ]
    ## Fit model and generate predictions:
    fit  <- lm(model, data = dat0)
    pred <- predict(fit, newdata = dat1)

    ## Save MSE:
    mse[k] <- MSE(y_pred = pred, y_true = dat1$Outcome)
  }
  ## Save the CVE:
  matrix4_cve[m,]<-c(model,mean(mse))
}
index=which.min(matrix4_cve[,2])
model4_min=matrix4_cve[index,]


# --------- models with 5 features ---------- #

matrix5_cve<-matrix(nrow=length(model5.2),ncol=2)
for(m in 1 : length(model5.2)) {
  ## Loop over K repititions:
  y<-"Outcome ~ "
  model<-paste0(y,model5.2[m])
  mse <- c()
  for(k in 1 : K) {
    ## Partition data:
    dat0 <- train[part != k, ]
    dat1 <- train[part == k, ]
    ## Fit model and generate predictions:
    fit  <- lm(model, data = dat0)
    pred <- predict(fit, newdata = dat1)

    ## Save MSE:
    mse[k] <- MSE(y_pred = pred, y_true = dat1$Outcome)
  }
  ## Save the CVE:
  matrix5_cve[m,]<-c(model,mean(mse))
}

index=which.min(matrix5_cve[,2])
model5_min=matrix5_cve[index,]

################# BEST MODEL ########################
best_models = c(model1_min,model2_min,model3_min,model4_min,model5_min)

even_indexes<-seq(2,10,2)
odd_indexes<-seq(1,9,2)

c1 <- best_models[odd_indexes]
c2 <- best_models[even_indexes]

Best_models_df<- data.frame("Model" = c1, "MSE" = c2)

min_index <- which.min(Best_models_df[,2])

Best_models_df
################## HISTOGRAM OF ALL ERRORS ########################

all_errors = c(matrix1_cve[,2],matrix2_cve[,2],
               matrix3_cve[,2],matrix4_cve[,2],
               matrix5_cve[,2])

all_errors=as.numeric(all_errors)
hist(all_errors)

################################### FIT THE BETS MODEL

model=best_models[9]
fit  <- lm(model, data = train)
coefficients=coef(fit)

coefficients
##### PREDICT
p1 <- predict(fit, newdata = test)
true_value <-outcome_2007
true_value=as.numeric(true_value)
```

```r
mse=MSE(y_pred = p1, y_true = true_value) ## MSE

Comparison_df <- cbind(t(round(outcome_2007, 2)),
                       round(p1, 2))
colnames(Comparison_df) <- c("True", "Predicted")


Comparison_df
```