Exercise 2

Emilia Löscher

2022-09-28

Aim

The aim is to compare different approaches for classification based on the MSE with the help of cross validation. The models that we are going to compare are the following: - logistic regression (lr) - linear discriminant analysis (lda) - random forest (rf)

Set up

We are using the cardiovascular disease dataset of 253 patients (from the SLV course). The data set is split into a five parts, so that while performing cross validation, the training data set always consists of 80% and a test data set consists of 20% of the original data set. As this splitting of the data set is random, it requires using RNG.

Using cross validation, the three models are fit to the respective training data set, always leaving out one fifth in turn. Then predictions are made for the 20% of the data which make up the test data set. By comparing these predictions to the "true" outcomes, the MSE is obtained.

Hence, we obtain a 3x5-matrix containing the results. By taking the mean across the cross validation for each model, we can compare which model performs best by identifying the model with the lowest mean MSE.

Setting the seed

```r
set.seed(2010)
```

Packages required and data

```r
library(ggplot2)
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```
##
## Attache Paket: 'randomForest'

## Das folgende Objekt ist maskiert 'package:ggplot2':
##
##     margin
```

```
library(MASS)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## v purrr   0.3.5
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::combine()       masks randomForest::combine()
## x dplyr::filter()        masks stats::filter()
## x dplyr::lag()           masks stats::lag()
## x randomForest::margin() masks ggplot2::margin()
## x dplyr::select()        masks MASS::select()
```

```
library(utils)


cardio <- read.csv("cardiovascular_treatment.csv") %>%
  mutate(severity = as.factor(severity),
         gender   = as.factor(gender),
         dose     = as.factor(dose),
         response = as.factor(response))
```

Code for the cross validation and MSE

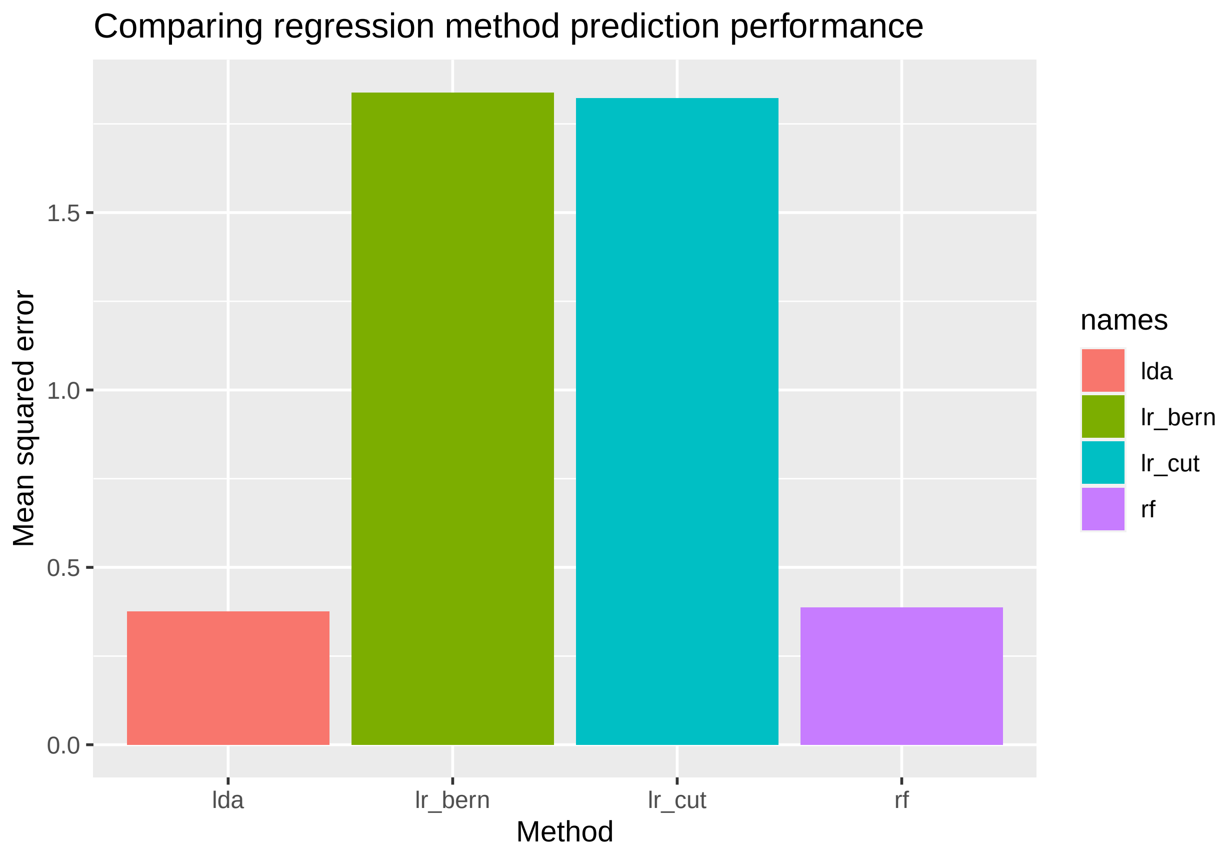
```
#MSE function
mse <- function(y_true, y_pred) mean((y_true - y_pred)^2)

myCV <- function(k = 5, data = cardio){
   mse_mat <- matrix(NA,  ncol = k, nrow = 4)

  split <- c(rep(1:k, floor(nrow(data)/k)), 1:(nrow(data)%%k))
  split_shuff <- sample(split, length(split) )
  #adding column to randomly split the data set
  data$split <- split_shuff
  for(i in 1:k){
    #splitting the data set for each k into train and test
    data_train <- data[which(data$split != i),]
    data_test <- data[which(data$split == i),]
  #fitting the different models
```

```r
    #logistic regression
    lr <- glm(response~ ., data = data_train, family= "binomial")

    #lda
    lda <- lda(response~. , data = data_train)

    #random forest
    rf <- randomForest(response ~., data = data_train)

    model_list <- list(lr, lda, rf)
    pred_list <- list()
    pred_list[[1]] <- ifelse(predict(lr, newdata = data_test) < 0.5, 0,1)
    pred_list[[2]] <- rbernoulli(nrow(data_test),p=  predict(lr, newdata = data_test))
    pred_list[[3]] <- predict(lda, newdata = data_test)$class
    pred_list[[4]] <- predict(rf, newdata = data_test)
    for(j in 1:4){
     mse_mat[j,i] <- mse(as.numeric(data_test$response), as.numeric(pred_list[[j]]))
    }
  }
  return(mse_mat)
}
mse_res <- myCV()
rownames(mse_res) <- c("lr_cut", "lr_bern", "lda", "rf")

rowMeans(mse_res)
```

```
##    lr_cut   lr_bern       lda        rf
## 1.8226667 1.8388235 0.3756078 0.3871373
```

Presentation, visualization, and discussion of results

```r
results <- data.frame("names"= rownames(mse_res), "mse" = rowMeans(mse_res))
ggplot(results, aes(x = names, y = mse, fill= names))+
  geom_bar(stat ="identity")+
   labs(
    x     = "Method",
    y     = "Mean squared error",
    title = "Comparing regression method prediction performance")
```



It can be seen that regarding the MSE, the linear discriminant analysis performs best (MSE = 0.3756). The Random Forest method has a similar performance with a MSE of 0.3871. The logistic regression methods perform worst. It does not make much of a difference if the classification is done according to a cut-off value of 0.5 or using a bernoulli distribution with the obtained probabilities. The MSEs are 1.8227 (cut-off) and 1.8388 (bernoulli).

Replication of the analysis

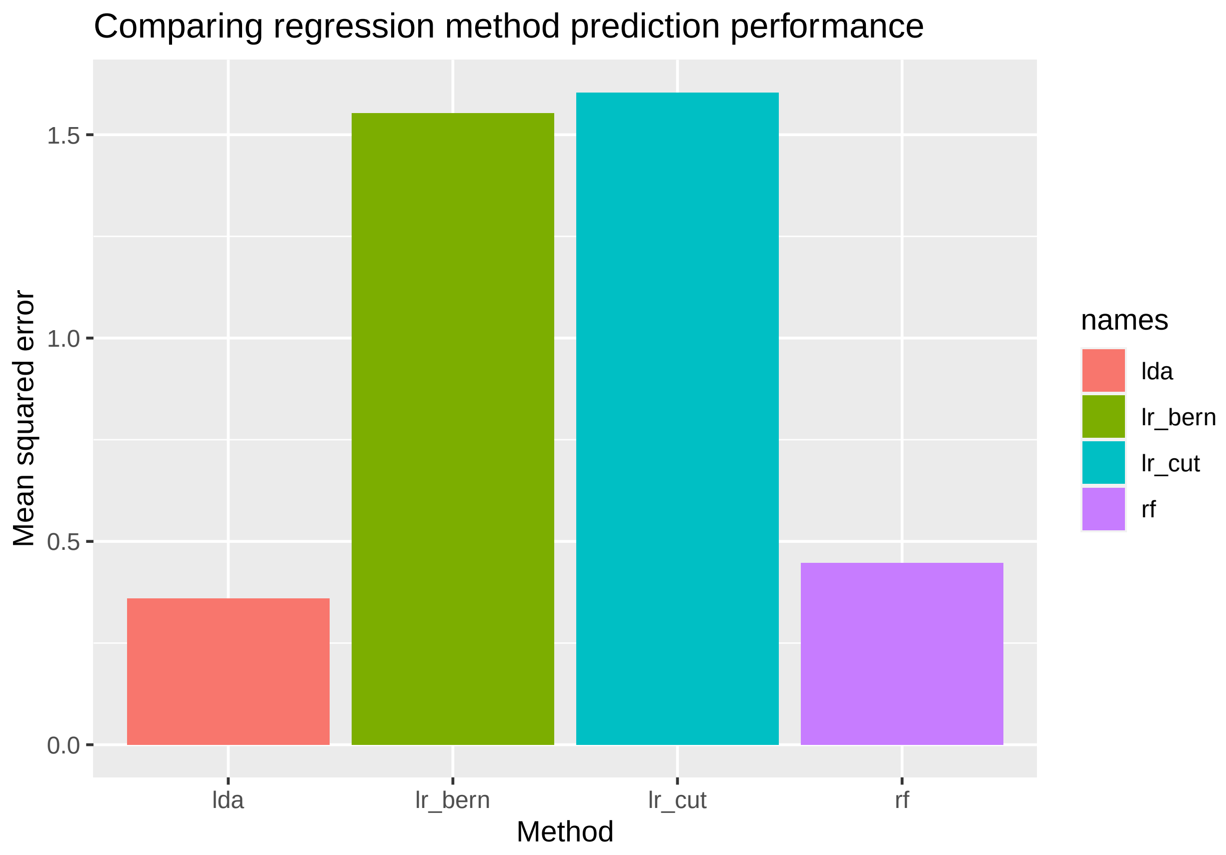
When we use the same code as before with another seed to see if we obtain the same results. They will be slightly different, but the order of performance of the different methods should be the same.

```
set.seed(2110)

library(ggplot2)
library(randomForest)
library(MASS)
library(tidyverse)
library(utils)


cardio <- read.csv("cardiovascular_treatment.csv") %>%
  mutate(severity = as.factor(severity),
         gender   = as.factor(gender),
         dose     = as.factor(dose),
         response = as.factor(response))
```

```r
#MSE function
mse <- function(y_true, y_pred) mean((y_true - y_pred)^2)

myCV <- function(k = 5, data = cardio){
   mse_mat <- matrix(NA,  ncol = k, nrow = 4)

  split <- c(rep(1:k, floor(nrow(data)/k)), 1:(nrow(data)%%k))
  split_shuff <- sample(split, length(split) )
  #adding column to randomly split the data set
  data$split <- split_shuff
  for(i in 1:k){
    #splitting the data set for each k into train and test
    data_train <- data[which(data$split != i),]
    data_test <- data[which(data$split == i),]
  #fitting the different models
    #logistic regression
    lr <- glm(response~ ., data = data_train, family= "binomial")

    #lda
    lda <- lda(response~. , data = data_train)

    #random forest
    rf <- randomForest(response ~., data = data_train)

    model_list <- list(lr, lda, rf)
    pred_list <- list()
    pred_list[[1]] <- ifelse(predict(lr, newdata = data_test) < 0.5, 0,1)
    pred_list[[2]] <- rbernoulli(nrow(data_test),p=  predict(lr, newdata = data_test))
    pred_list[[3]] <- predict(lda, newdata = data_test)$class
    pred_list[[4]] <- predict(rf, newdata = data_test)
    for(j in 1:4){
     mse_mat[j,i] <- mse(as.numeric(data_test$response), as.numeric(pred_list[[j]]))
    }
  }
  return(mse_mat)
}
mse_res <- myCV()
rownames(mse_res) <- c("lr_cut", "lr_bern", "lda", "rf")

print(rowMeans(mse_res))
```

```
##    lr_cut   lr_bern       lda        rf
## 1.6040784 1.5529412 0.3597647 0.4466667
```

```r
results <- data.frame("names"= rownames(mse_res), "mse" = rowMeans(mse_res))
ggplot(results, aes(x = names, y = mse, fill= names))+
  geom_bar(stat ="identity")+
   labs(
    x     = "Method",
    y     = "Mean squared error",
    title = "Comparing regression method prediction performance")
```



We see that the ranking of the methods is similar and the values of the MSEs vary a bit: Before: 1.8226667(lr_cut) 1.8388235 (lr_bern) 0.3756078 (lda) 0.3871373 (rf) Now: 1.6040784 (lr_cut) 1.5529412 (lr_bern) 0.3597647 (lda) 0.4466667 (rf)

There is now a larger difference between the linear discriminant analysis and the random forest method. The LDA still performs best with respect to the MSE. Now, the cut off logistic regression performs worse than the bernoulli logistic regression.

Session info

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=German_Germany.utf8  LC_CTYPE=German_Germany.utf8
## [3] LC_MONETARY=German_Germany.utf8 LC_NUMERIC=C
## [5] LC_TIME=German_Germany.utf8
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] forcats_0.5.2        stringr_1.4.1        dplyr_1.0.10
##  [4] purrr_0.3.5          readr_2.1.3          tidyr_1.2.1
##  [7] tibble_3.1.8         tidyverse_1.3.2      MASS_7.3-57
## [10] randomForest_4.7-1.1 ggplot2_3.3.6
##
## loaded via a namespace (and not attached):
##  [1] lubridate_1.8.0     assertthat_0.2.1    digest_0.6.29
##  [4] utf8_1.2.2          R6_2.5.1            cellranger_1.1.0
##  [7] backports_1.4.1     reprex_2.0.2        evaluate_0.17
## [10] httr_1.4.4          highr_0.9           pillar_1.8.1
## [13] rlang_1.0.6         googlesheets4_1.0.1 readxl_1.4.1
## [16] rstudioapi_0.14     rmarkdown_2.17      labeling_0.4.2
## [19] googledrive_2.0.0   munsell_0.5.0       broom_1.0.1
## [22] compiler_4.2.1      modelr_0.1.9        xfun_0.33
## [25] pkgconfig_2.0.3     htmltools_0.5.3     tidyselect_1.1.2
## [28] fansi_1.0.3         crayon_1.5.2        tzdb_0.3.0
## [31] dbplyr_2.2.1        withr_2.5.0         grid_4.2.1
## [34] jsonlite_1.8.2      gtable_0.3.1        lifecycle_1.0.3
## [37] DBI_1.1.3           magrittr_2.0.3      scales_1.2.1
## [40] cli_3.4.1           stringi_1.7.8       farver_2.1.1
## [43] fs_1.5.2            xml2_1.3.3          ellipsis_0.3.2
## [46] generics_0.1.3      vctrs_0.4.2         tools_4.2.1
## [49] glue_1.6.2          hms_1.1.2           fastmap_1.1.0
## [52] yaml_2.3.5          colorspace_2.0-3    gargle_1.2.1
## [55] rvest_1.0.3         knitr_1.40          haven_2.5.1
```