Features Selection - LOVO K-medians

IV. Leave One Variable Out - Kmedians

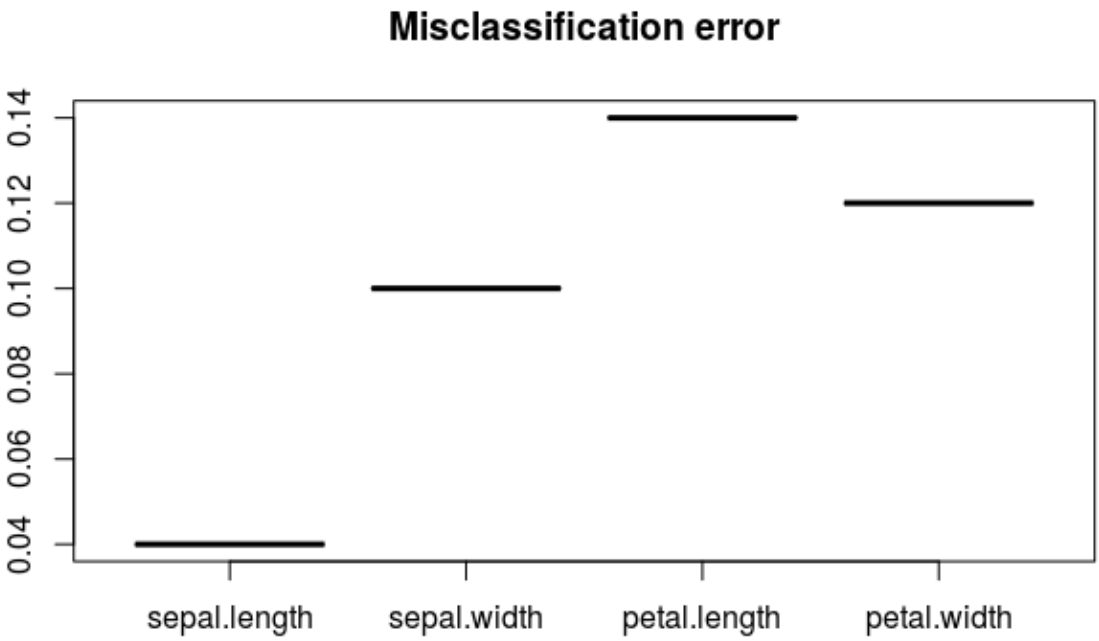
In this section we will apply the LOVO K-medians to rank the different variable of the data set "iris". Hence, we will build 4 K-medians clustering, where each clustering corresponds to the dataset without one of the variable. Finally the features will be ranked according to the misclassification error or the rand index. Indeed, the most important feature is the one for which a clustering without it lead to the greatest misclassification error rate (inversely for rand index).

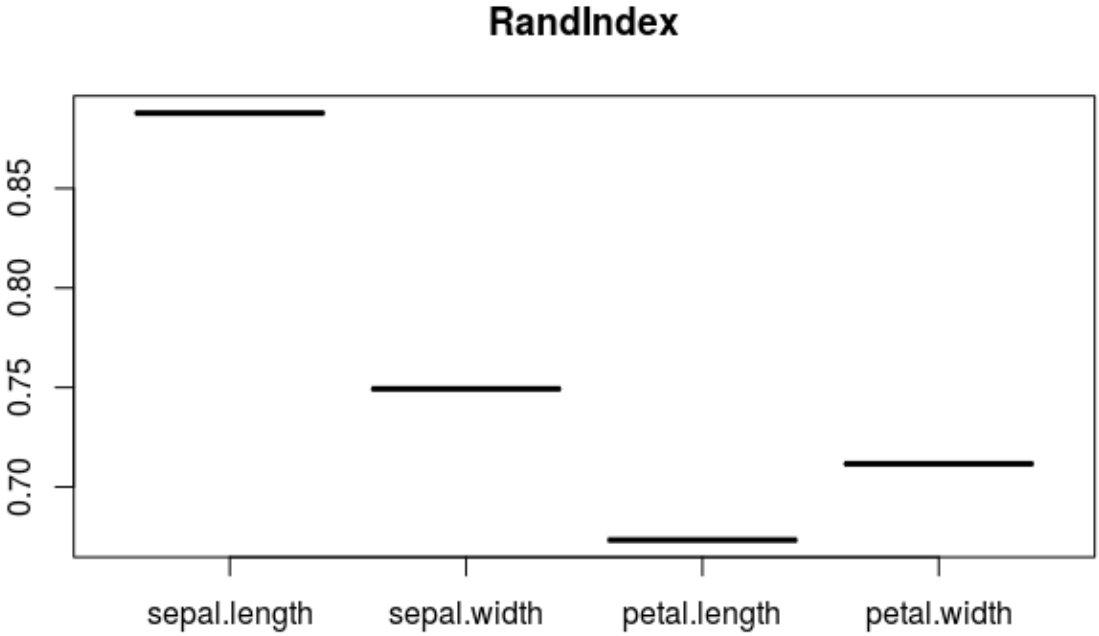
The boxplot highlights the fact that the most important variable are:

1. petal.length
2. petal.width
3. sepal.width

```
data(iris)
dataset <- iris
M <- 100
randind.km.sep.l <- NULL
randind.km.sep.w <- NULL
randind.km.pet.l <- NULL
randind.km.pet.w <- NULL

err.km.sep.l <- NULL
err.km.sep.w <- NULL
err.km.pet.l <- NULL
err.km.pet.w <- NULL

M<-100
for (i in 1:M){
n <- nrow(iris)
  set.seed(1)
  train.index <- sample(1:nrow(dataset),size = floor(2/3 * nrow(dataset)))
  x.train <- dataset[train.index,1:ncol(dataset)-1]
  x.test <- dataset[-train.index,1:ncol(dataset)-1]
  y.train <- dataset[train.index,ncol(dataset)]
  y.test <- dataset[-train.index,ncol(dataset)]

  x.train.sep.l <- x.train[,-1]
  x.test.sep.l <- x.test[,-1]

  x.train.sep.w <- x.train[,-2]
  x.test.sep.w <- x.test[,-2]

  x.train.pet.l <- x.train[,-3]
  x.test.pet.l <- x.test[,-3]

  x.train.pet.w <- x.train[,-4]
  x.test.pet.w <- x.test[,-4]

  ## Kmedians
```

```
  km.sep.l <- kcca(x.train.sep.l, k=3, kccaFamily("kmedians"))
  km.sep.w <- kcca(x.train.sep.w, k=3, kccaFamily("kmedians"))
  km.pet.l <- kcca(x.train.pet.l, k=3, kccaFamily("kmedians"))
  km.pet.w <- kcca(x.train.pet.w, k=3, kccaFamily("kmedians"))

  pred.sep.l <- predict(km.sep.l, newdata = x.test.sep.l)
  pred.sep.w <- predict(km.sep.w, newdata = x.test.sep.w)
  pred.pet.l <- predict(km.pet.l, newdata = x.test.pet.l)
  pred.pet.w <- predict(km.pet.w, newdata = x.test.pet.w)

  randind.km.sep.l <- c(randind.km.sep.l, randIndex(pred.sep.l,y.test))
  randind.km.sep.w <- c(randind.km.sep.w, randIndex(pred.sep.w,y.test))
  randind.km.pet.l <- c(randind.km.pet.l, randIndex(pred.pet.l,y.test))
  randind.km.pet.w <- c(randind.km.pet.w, randIndex(pred.pet.w,y.test))


  err.km.sep.l <- c(err.km.sep.l, error(pred.sep.l,y.test, print = FALSE)[1])
  err.km.sep.w <- c(err.km.sep.w, error(pred.sep.w,y.test, print = FALSE)[1])
  err.km.pet.l <- c(err.km.pet.l, error(pred.pet.l,y.test, print = FALSE)[1])
  err.km.pet.w <- c(err.km.pet.w, error(pred.pet.w,y.test, print = FALSE)[1])


}

## Plotting misclassification errors and rand index boxplot
errors.models <- list(err.km.sep.l,err.km.sep.w,err.km.pet.l,err.km.pet.w)
randindex.models <- list(randind.km.sep.l,randind.km.sep.w,randind.km.pet.l,randind.km.pet.w)

names(errors.models) <- c('sepal.length','sepal.width','petal.length','petal.width')
names(randindex.models) <- c('sepal.length','sepal.width','petal.length','petal.width')

boxplot(errors.models,main="Misclassification error")
boxplot(randindex.models,main="RandIndex")
```


Misclassification error
0.04
0.06
0.08
0.10
0.12
0.14
sepal.length
sepal.width
petal.length
petal.width


RandIndex
0.70
0.75
0.80
0.85
sepal.length
sepal.width
petal.length
petal.width