Solution Tutorial to PCA and PLS

Benoit Liquet

15 April 2023

Contents

1 PCA

1.1 Data

The dataset includes gene expression data for 6830 genes from 64 cancer samples (from different cancer subtypes).

Data can be downloaded form http://www-stat.stanford.edu/~tibs/ElemStatLearn/

For this analysis (and to simplify the plots), 5 subtypes with only 1 samples have been removed: UNKNOWN, K562B-repro, K562A-repro, MCF7A-repro, MCF7D-repro

1.2 Data Management

- Read the data from the txt file `nci.data.txt`

```
dat.1 <- read.table("nci.data.txt")
```

- What is the dimension of your Data Frame ?

```
dim(dat.1)
## [1] 6830   64
```

- The names of the 64 samples are stored in the file `subtypes_names.Rdata`. Load the names and removed from the data frame the subtypes with only 1 samples:

UNKNOWN, K562B-repro, K562A-repro, MCF7A-repro, MCF7D-repro

```
names.data <- as.character(read.csv("names-sample.csv",header=FALSE,sep=" ")[,1])
names.data <- names.data[-c(36,38,55,52)]
save(names.data,file="subtypes_names.Rdata")
```

```
load("subtypes_names.Rdata")
```

Here an example:

```
## Remove subtype with only 1 sample
one.sample <- c("UNKNOWN", "K562B-repro", "K562A-repro", "MCF7A-repro", "MCF7D-repro")
ind <- which(names.data%in%one.sample)
names.final <- names.data[-ind]
dat.1 <- dat.1[,-ind]
dim(dat.1)
## [1] 6830   59
dat.2 <- t(dat.1)
dim(dat.2)
## [1]   59 6830
```

- So now you are working with 59 samples.

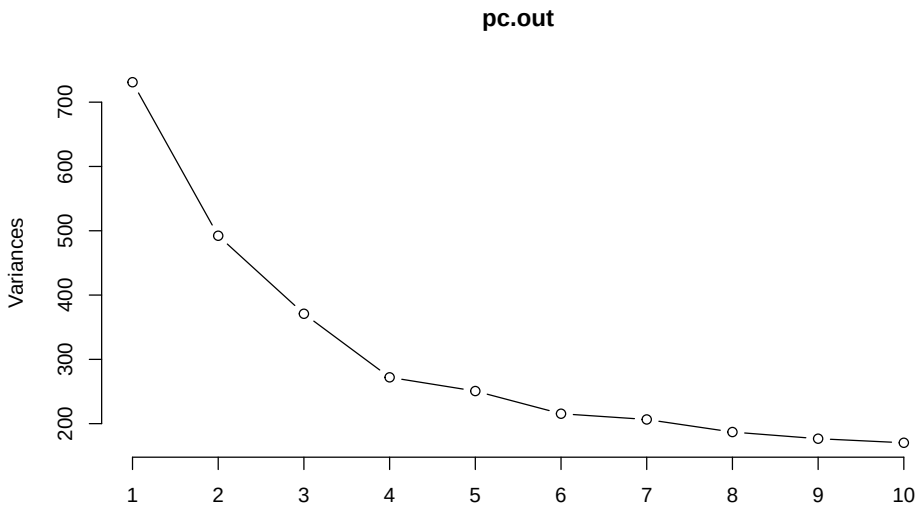
1.3 Run a PCA

- Run a PCA

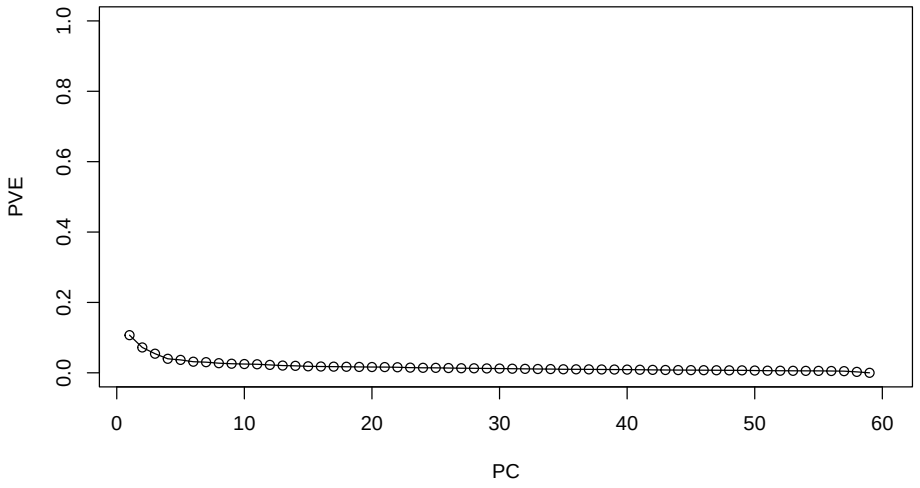
```
##### PCA with prcomp
pc.out <- prcomp(dat.2, scale=TRUE, retx=TRUE)
##### Choice of number of component
```

- Choice of number of component

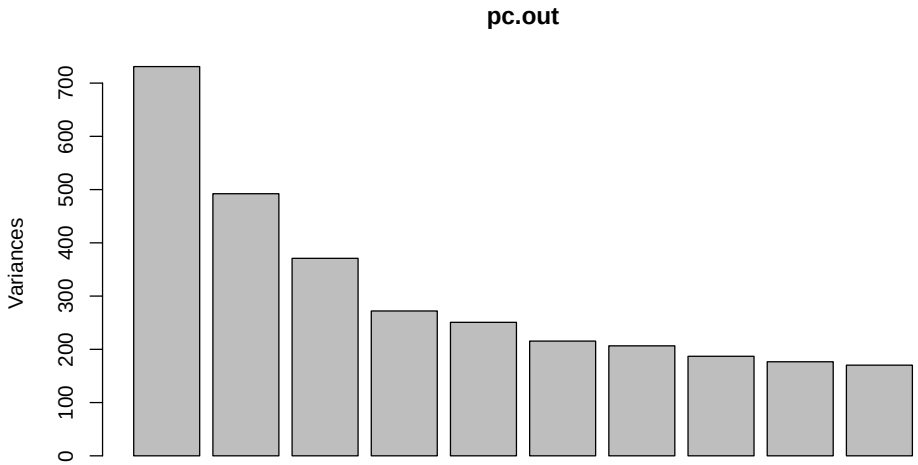
```
##### PCA with prcomp
pc.out <- prcomp(dat.2, scale=TRUE, retx=TRUE)
##### Choice of number of component
plot(pc.out,type="l")
```




```
plot((pc.out$sdev^2)/sum(pc.out$sdev^2), xlab="PC",
     ylab="PVE", ylim=c(0, 1), type='o',xlim=c(1,60))
```




```
screeplot(pc.out)
```



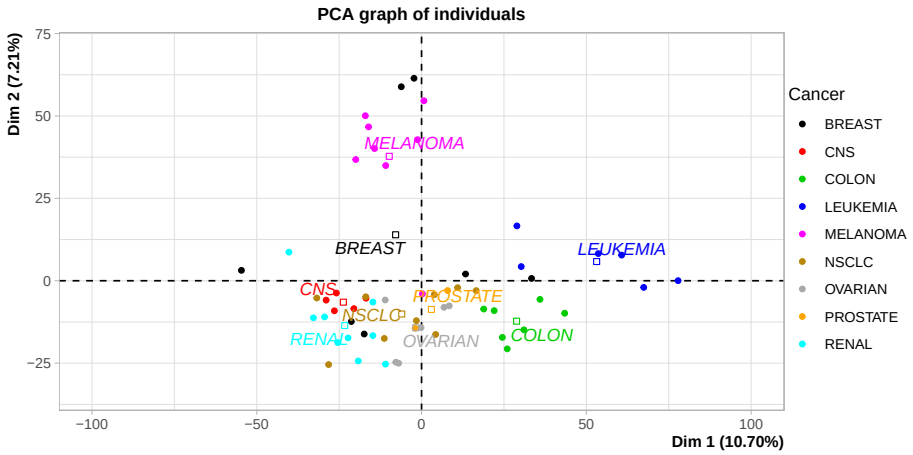

```
summary(pc.out)
## Importance of components:
##                            PC1      PC2      PC3      PC4      PC5      PC6
## Standard deviation     27.037 22.18643 19.25886 16.49400 15.83406 14.67861
## Proportion of Variance  0.107  0.07207  0.05431  0.03983  0.03671  0.03155
## Cumulative Proportion   0.107  0.17910  0.23341  0.27324  0.30995  0.34149
##                            PC7      PC8      PC9     PC10     PC11     PC12
## Standard deviation     14.37134 13.67336 13.29001 13.04881 12.87674 12.38813
## Proportion of Variance  0.03024  0.02737  0.02586  0.02493  0.02428  0.02247
## Cumulative Proportion   0.37173  0.39910  0.42496  0.44989  0.47417  0.49664
##                           PC13     PC14     PC15     PC16     PC17    PC18
## Standard deviation     11.86734 11.64578 11.23625 11.06547 10.93829 10.8378
## Proportion of Variance  0.02062  0.01986  0.01849  0.01793  0.01752  0.0172
## Cumulative Proportion   0.51726  0.53712  0.55560  0.57353  0.59105  0.6082
##                           PC19     PC20     PC21     PC22     PC23    PC24
## Standard deviation     10.69318 10.58879 10.54715 10.38115 10.02575 9.86413
## Proportion of Variance  0.01674  0.01642  0.01629  0.01578  0.01472 0.01425
## Cumulative Proportion   0.62499  0.64140  0.65769  0.67347  0.68819 0.70243
##                           PC25    PC26    PC27    PC28    PC29    PC30    PC31
## Standard deviation     9.81773 9.63323 9.43698 9.36499 9.22820 9.08380 8.93499
## Proportion of Variance 0.01411 0.01359 0.01304 0.01284 0.01247 0.01208 0.01169
## Cumulative Proportion  0.71654 0.73013 0.74317 0.75601 0.76848 0.78056 0.79225
##                           PC32    PC33    PC34    PC35    PC36   PC37    PC38
## Standard deviation     8.77378 8.60447 8.53749 8.28149 8.23875 8.1832 8.06190
## Proportion of Variance 0.01127 0.01084 0.01067 0.01004 0.00994 0.0098 0.00952
## Cumulative Proportion  0.80352 0.81436 0.82503 0.83507 0.84501 0.8548 0.86433
##                           PC39    PC40    PC41    PC42    PC43   PC44    PC45
## Standard deviation     7.97843 7.90143 7.72475 7.56830 7.41647 7.2966 7.23393
## Proportion of Variance 0.00932 0.00914 0.00874 0.00839 0.00805 0.0078 0.00766
## Cumulative Proportion  0.87365 0.88279 0.89153 0.89992 0.90797 0.9158 0.92343
##                           PC46    PC47    PC48    PC49    PC50   PC51   PC52
## Standard deviation     7.15244 7.05517 6.92794 6.90651 6.64532 6.4547 6.3462
## Proportion of Variance 0.00749 0.00729 0.00703 0.00698 0.00647 0.0061 0.0059
## Cumulative Proportion  0.93092 0.93820 0.94523 0.95222 0.95868 0.9648 0.9707
##                           PC53    PC54    PC55    PC56   PC57    PC58      PC59
## Standard deviation     6.2369 6.17022 6.14843 5.93041 5.6650 4.26996 2.122e-14
```

```
## Proportion of Variance 0.0057 0.00557 0.00553 0.00515 0.0047 0.00267 0.000e+00
## Cumulative Proportion  0.9764 0.98195 0.98748 0.99263 0.9973 1.00000 1.000e+00
```

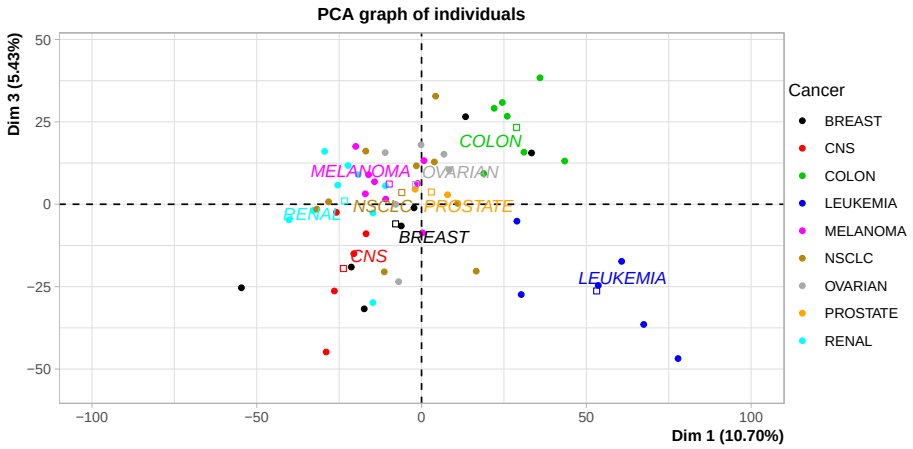
- Project the samples on the first 3 components

You should get some plots like the following

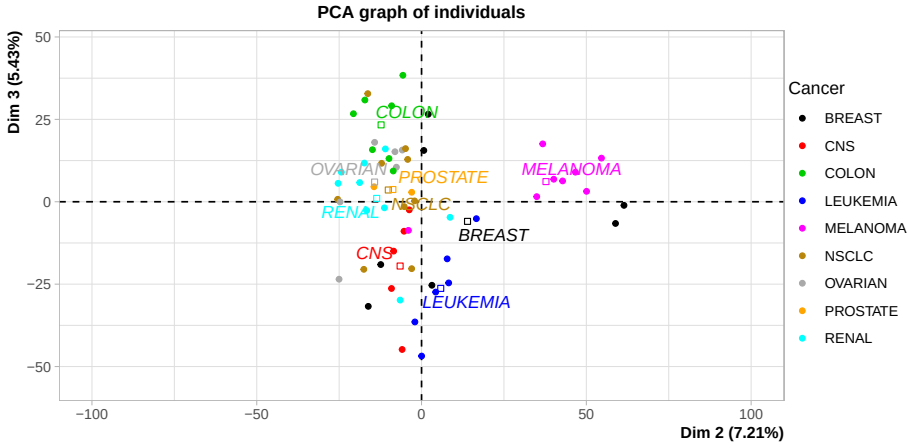
```
##### Projection of Samples
require(FactoMineR)
data.final <- data.frame(dat.2,Cancer=names.final)
res.pca = PCA(data.final, scale.unit=TRUE, ncp=4, graph=F,quali.sup=6831)
plot(res.pca,axes=c(1,2),choix="ind",habillage=6831,label="quali",xlim=c(-100,100))
```




```
plot(res.pca,axes=c(1,3),choix="ind",habillage=6831,label="quali",xlim=c(-100,100))
```




```
plot(res.pca,axes=c(2,3),choix="ind",habillage=6831,label="quali",,xlim=c(-100,100))
```



1.4 Sparse PCA

Find a way to select the most relevant genes by using a sparse PCA.

```
##### Sparse PCA
#install.packages("elasticnet")
library(elasticnet)
### Ready to wait !!
sparse.pca.result <- spca(dat.2, K = 3, type = "predictor", sparse = "varnum", para = c(10, 10,10))
```

```
ind <- which(sparse.pca.result$loadings[,]!=0,arr.ind=TRUE)
res <- sparse.pca.result$loadings[ind[,1],]
rownames(res) <- ind[,1]
res
##              PC1         PC2         PC3
## 5557 0.13680124  0.00000000  0.00000000
## 5586 0.05518063  0.00000000  0.00000000
## 5845 0.13273471  0.00000000  0.00000000
## 5913 0.16216825  0.00000000  0.00000000
## 5937 0.27398937  0.00000000  0.00000000
## 5942 0.74494222  0.00000000  0.00000000
## 5943 0.27195384  0.00000000  0.00000000
## 5948 0.08169883  0.00000000  0.00000000
## 5980 0.09861000  0.00000000  0.00000000
## 6393 0.46255472  0.00000000  0.00000000
## 113  0.00000000 -0.04167595  0.00000000
## 196  0.00000000 -0.19973132  0.00000000
## 224  0.00000000 -0.09795758  0.00000000
## 243  0.00000000 -0.05269887  0.00000000
## 252  0.00000000 -0.41575684  0.00000000
## 256  0.00000000 -0.83971810  0.00000000
## 266  0.00000000 -0.18494413  0.00000000
## 286  0.00000000 -0.17666885  0.00000000
## 4094 0.00000000  0.02899376  0.00000000
## 6149 0.00000000  0.04195914  0.00000000
## 2082 0.00000000  0.00000000  0.31836665
```

```
## 4244 0.00000000  0.00000000 -0.53083905
## 4288 0.00000000  0.00000000 -0.40178834
## 4344 0.00000000  0.00000000 -0.26659927
## 4354 0.00000000  0.00000000 -0.29350544
## 4388 0.00000000  0.00000000 -0.02157008
## 4701 0.00000000  0.00000000 -0.18458970
## 5868 0.00000000  0.00000000 -0.33792626
## 5869 0.00000000  0.00000000 -0.04309358
## 5884 0.00000000  0.00000000 -0.38419821
```

2 PLS-DA

```
table(names.final)
## names.final
##   BREAST      CNS    COLON LEUKEMIA MELANOMA    NSCLC  OVARIAN PROSTATE
##        7        5        7        6        8        9        6        2
##    RENAL
##        9
```

We will define a binary response variable:

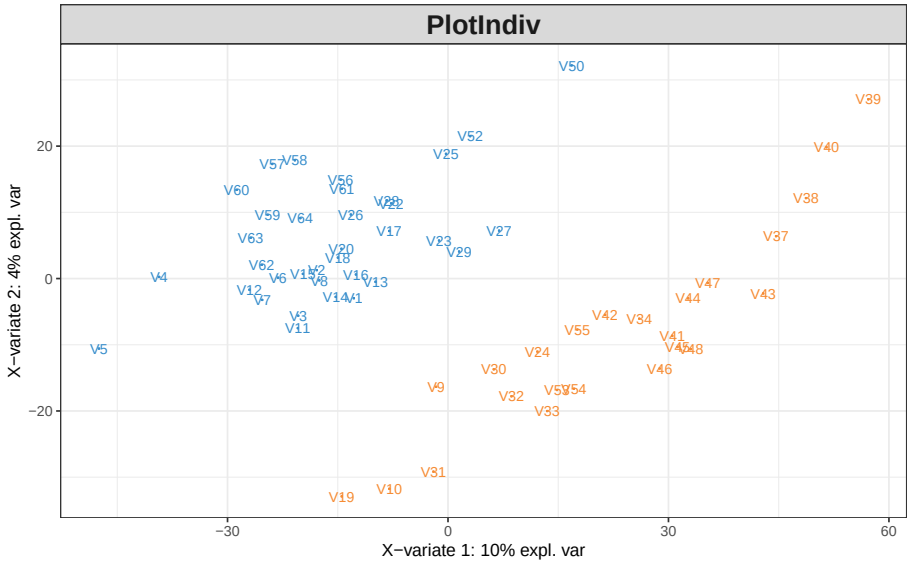
- 1 for subtypes cancer : Colon, Leukemia, Prostate, NSCLC
- 0 for: BREAST, CNS, MELANOMA, OVARIAN, RENAL

```
Y <- rep(0,59)
group1 <- which(names.final%in%c("COLON","LEUKEMIA","PROSTATE","NSCLC"))
Y[group1] <-1
table(Y)
## Y
##  0  1
## 35 24
```

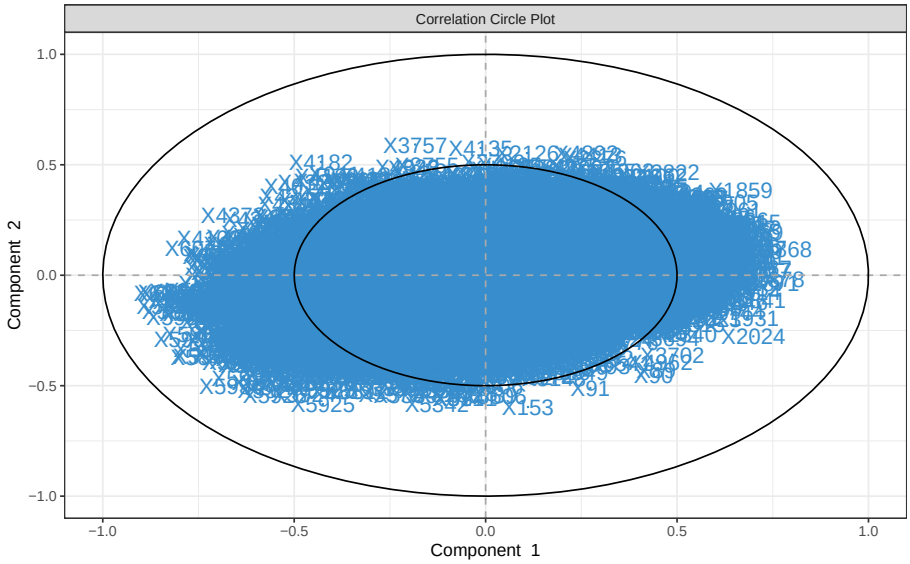
2.1 Run a PLS-DA model

```
library(mixOmics)
X <- dat.2
Y <- factor(Y)
summary(Y)
##  0  1
## 35 24
dim(X); length(Y)
## [1]   59 6830
## [1] 59
```

```
MyResult.plsda <- plsda(X, Y) # 1 Run the method
plotIndiv(MyResult.plsda)                          # 2 Plot the samples
```



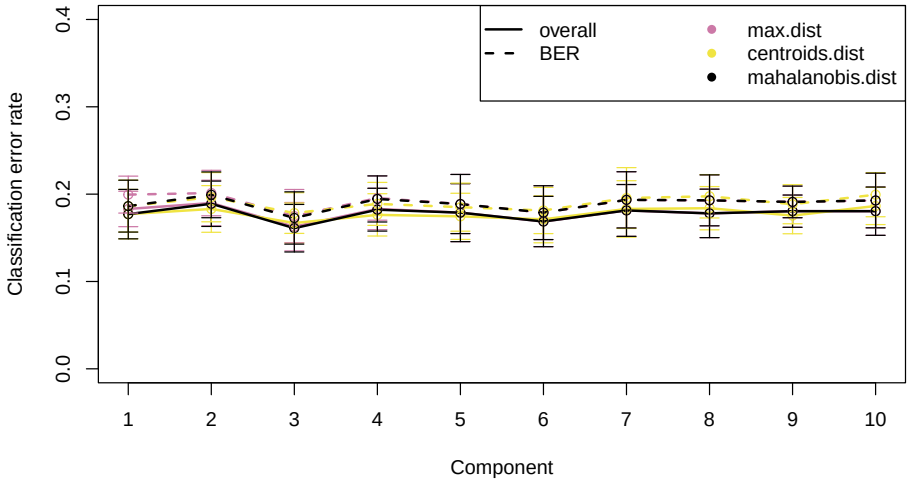

```
plotVar(MyResult.plsda)                    # 3 Plot the variables
```



2.2 choose the number of components

```
MyResult.plsda2 <- plsda(X,Y, ncomp=10)
set.seed(30) # for reproducibility in this vignette, otherwise increase nrepeat
MyPerf.plsda <- perf(MyResult.plsda2, validation = "Mfold", folds = 3,
                  progressBar = FALSE, nrepeat = 20) # we suggest nrepeat = 50

plot(MyPerf.plsda, col = color.mixo(5:7), sd = TRUE, legend.position = "horizontal",ylim=c(0,0.4))
```

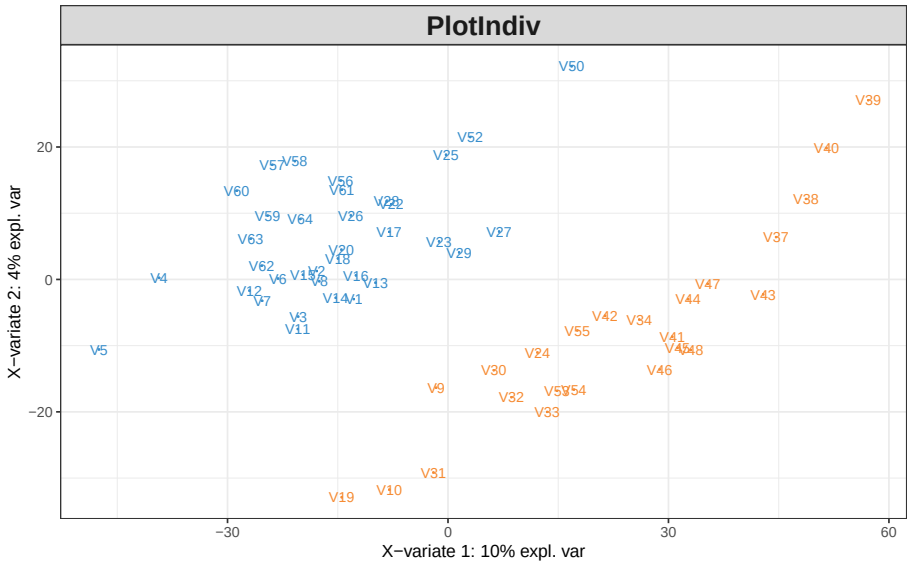



```
MyPerf.plsda
```

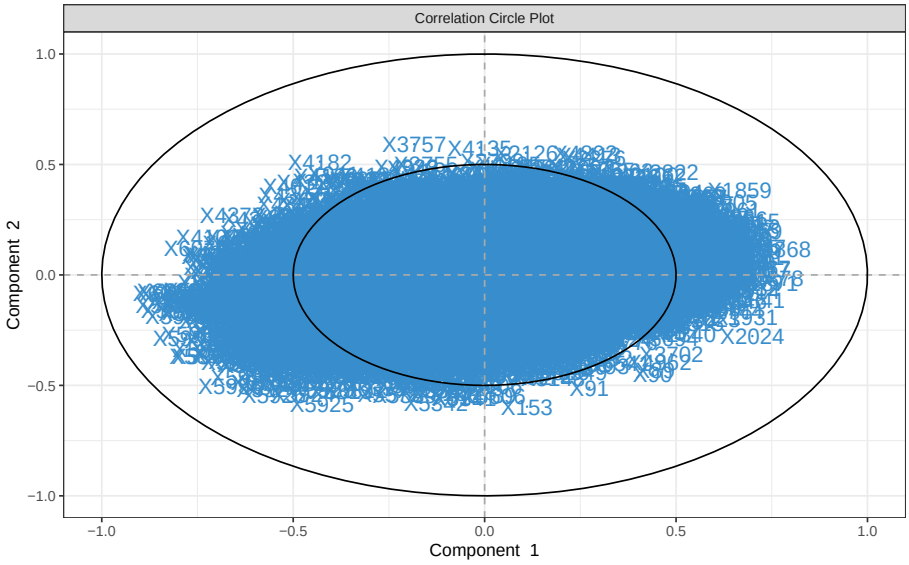
In this example, we retain only 2 components.

2.3 Project the samples on the first two components map

```
plotIndiv(MyResult.plsda)                    # 2 Plot the samples
```

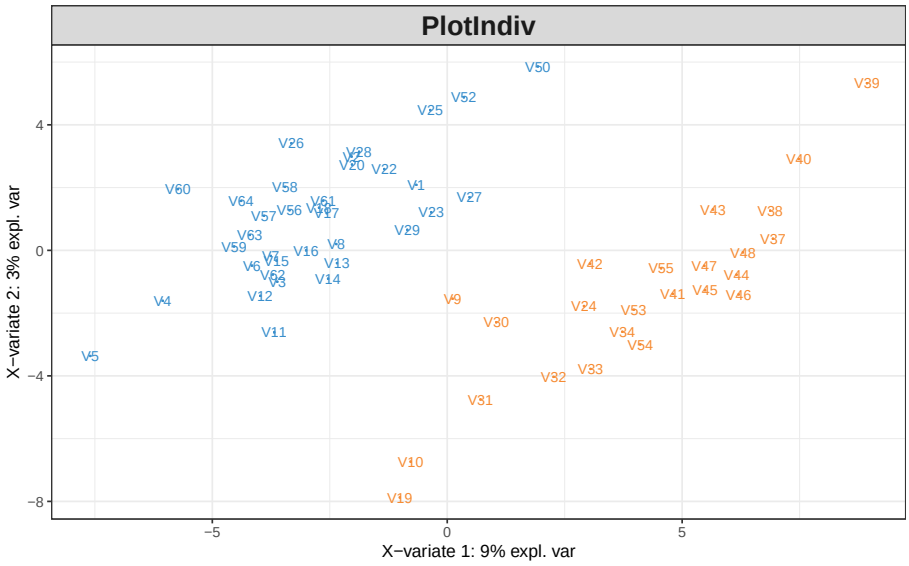



```
plotVar(MyResult.plsda)                      # 3 Plot the variables
```

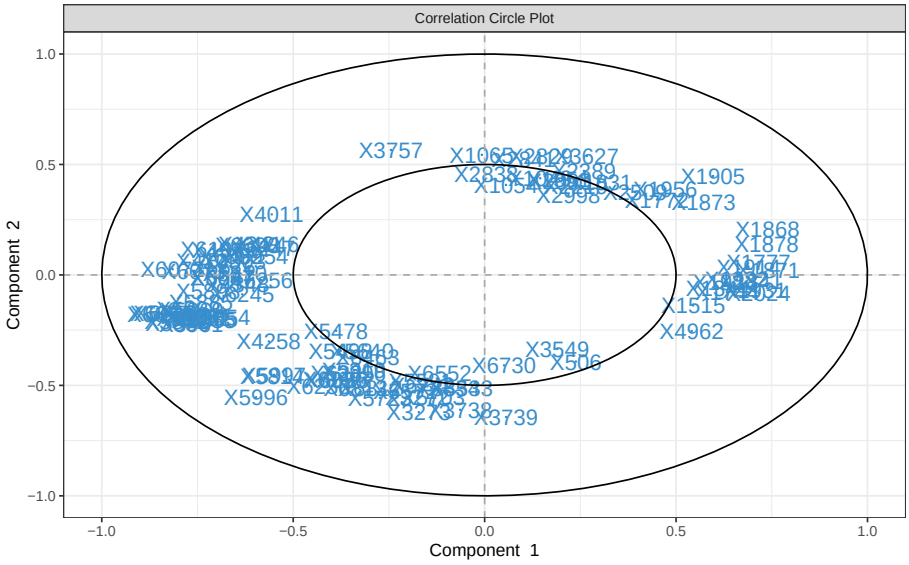


2.4 Run a sparse PLS-DA model

```
MyResult.splsda <- splsda(X, Y, keepX = c(50,50)) # 1 Run the method
plotIndiv(MyResult.splsda)                        # 2 Plot the samples
```




```
plotVar(MyResult.splsda)                          # 3 Plot the variables
```




```
#selectVar(MyResult.splsda, comp=1)$name            # Selected variables on component 1
```

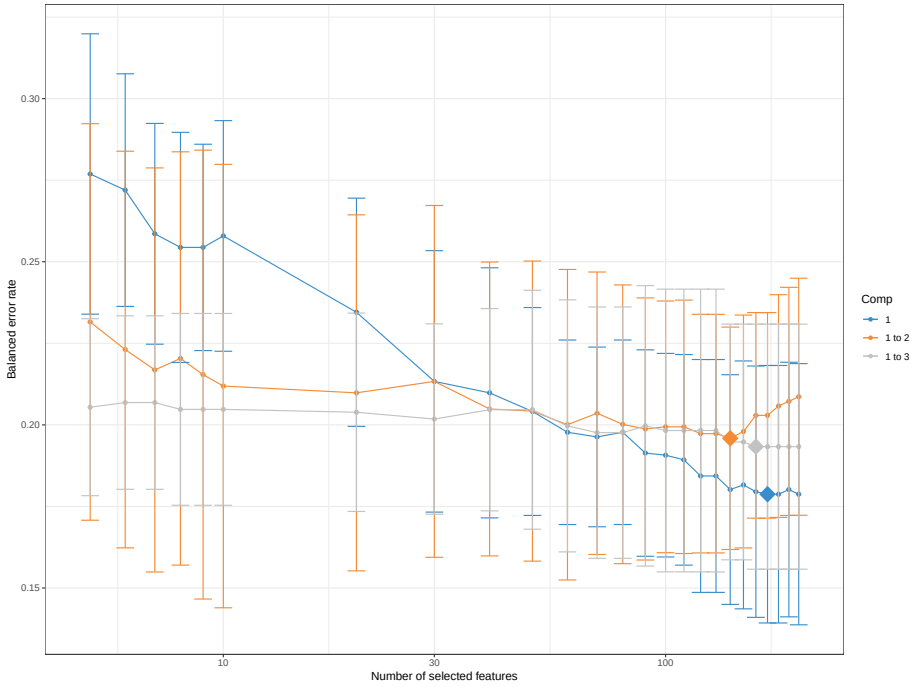
2.5 Choose the number of variables to select in each component

```
list.keepX <- c(5:10,  seq(20, 200, 10))
list.keepX # to output the grid of values tested
set.seed(30) # for reproducbility
tune.splsda.Tutorial <- tune.splsda(X, Y, ncomp = 3,
                          validation = 'Mfold',
                          folds = 3, dist = 'max.dist', progressBar = FALSE,
                          measure = "BER", test.keepX = list.keepX,
                          nrepeat = 10)   # we suggest nrepeat = 50
#save(tune.splsda.Tutorial,file="Tutorial_splsda_tune.RData")
```

```
##  [1]   5   6   7   8   9  10  20  30  40  50  60  70  80  90 100 110 120 130 140
## [20] 150 160 170 180 190 200
```

```
error <- tune.splsda.Tutorial$error.rate
ncomp <- tune.splsda.Tutorial$choice.ncomp$ncomp
# optimal number of components based on t-tests on the error rate
ncomp
## [1] 1
select.keepX <- tune.splsda.Tutorial$choice.keepX[1:ncomp]
# optimal number of variables to select
select.keepX
## comp1
##   170
```

```
plot(tune.splsda.Tutorial)#, col = color.jet(ncomp))
```



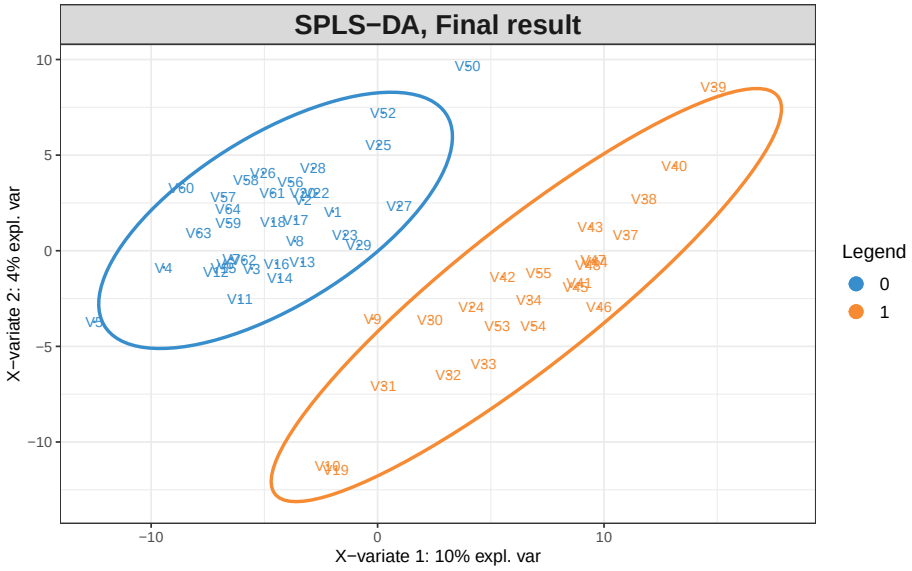
2.6 Final model

Based on those tuning results and the plot of the BER, we choose 2 components to visualise the results and run our final model:

```
select.keepX <- tune.splsda.Tutorial$choice.keepX[1:2]
# optimal number of variables to select
select.keepX
## comp1 comp2
##   170   140

MyResult.splsda.final <- splsda(X, Y, keepX = c(170,140))

plotIndiv(MyResult.splsda.final, legend=TRUE,
          ellipse = TRUE, title="SPLS-DA, Final result")
```


SPLS−DA, Final result
X−variate 2: 4% expl. var
X−variate 1: 10% expl. var
Legend
0
1