MACQUARIE
University

Solution Tutorial, PLS and PLS-DA

Benoit mates[*1]

[1]Macquarie University

[*]benoit.mates-weiland@mq.edu.au

Contents

1 Prediction using PLS-DA

In this practice we will use the dataset `Sonar` from the `mlbench` R package. The Sonar data consist of 208 data points collected on 60 predictors. The goal is to predict the two classes M for metal cylinder or R for rock).

```
library(mlbench)
library(caret)
data(Sonar)
```

We first split the data into train/test data split

```
set.seed(107)
inTrain <- createDataPartition(
  y = Sonar$Class,
  ## the outcome data are needed
  p = .75,
  ## The percentage of data in the
  ## training set
  list = FALSE
)
```

By default, createDataPartition does a stratified random split of the data. To partition the data:

```
training <- Sonar[ inTrain,]
testing  <- Sonar[-inTrain,]
nrow(training)
[1] 157
nrow(testing)
[1] 51
```

(a) Here, a partial least squares discriminant analysis (PLSDA) model will be tuned over the number of PLS components that should be retained. Using a 10-fold cross-validation with 3 repetitions. Explore the argument `trainControl` form the `train()` function from caret package.

(b) Based on the previous results, decide the number of components to retain.

(c) Using your selected model, predict the label of the test data.

(d) Provide the confusion matrix

(e) Use the package `mixOmics` to perform the same analysis. You will use the function `plsda` from this package.

(f) Project the samples on the first two components. Use the function `plotIndiv()`

(g) Tune the number of component using a K-fold cross-validation approach by optimizing the area under the curve (auc). Help: use the `perf` function.

(h) Using your selected model, predict the label of the test data by using the `centroid.dist` distance. Provide the confusion matrix as well.

(i) Provide the roc curve evaluated on the test set using `auroc()` function.

2 PLS2 using mixOmics package

This data set contains the expression measure of 3116 genes and 10 clinical measurements for 64 subjects (rats) that were exposed to non-toxic, moderately toxic or severely toxic doses of acetaminophen in a controlled experiment.

```
library(mixOmics)
data(liver.toxicity)
X <- liver.toxicity$gene
Y <- liver.toxicity$clinic
help(liver.toxicity)
```

In this practice we will use PLS2 to model the relation between the X and Y variables. Here are the dimensions of the matrices that includes clinical parameters associated with liver failure.

```
dim(X)
[1]   64 3116
dim(Y)
[1] 64 10
```

(a) First start by tuning the number of components to select by using the `perf()` function and the Q^2 criterion using repeated cross-validation.

(b) Run the model with 2 components.

(c) The amount of explained variance can be extracted for each dimension and each data set:

(d) Using the `plotIndiv()` function, display the sample and metadata information using the arguments group (colour) and pch (symbol) to better understand the similarities between samples modelled with sPLS2. Interpret the results.

(e) Using the `perf()` function and a cross-validation approach provide the RMSE of the clinical variables.

(f) Provide the correlation circle plot by using a cut off of 0.5 to display high correlation.

3 Solution Prediction using PLS-DA

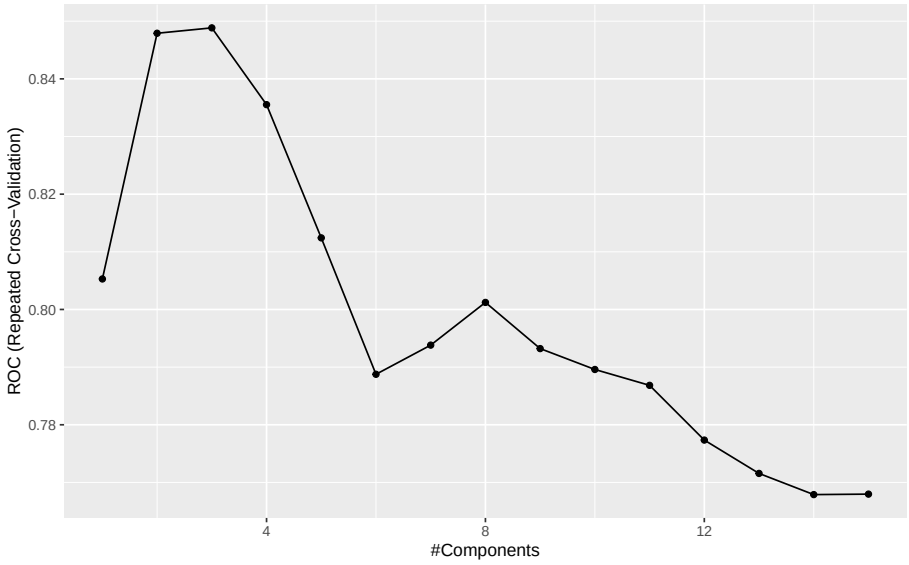
(a) Tune the number of component

```
detach(name="package:mixOmics")

ctrl <- trainControl(
  method = "repeatedcv",
  repeats = 3,
  classProbs = TRUE,
  summaryFunction = twoClassSummary
)

set.seed(123)
plsFit <- train(
  Class ~ .,
```

```
  data = training,
  method = "pls",
  preProc = c("center", "scale"),
  tuneLength = 15,
  trControl = ctrl,
  metric = "ROC"
)

plsFit
Partial Least Squares

157 samples
 60 predictor
  2 classes: 'M', 'R'

Pre-processing: centered (60), scaled (60)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 141, 141, 142, 142, 141, 142, ...
Resampling results across tuning parameters:

  ncomp  ROC        Sens       Spec
   1     0.8052910  0.7259259  0.6904762
   2     0.8479084  0.7495370  0.8005952
   3     0.8488426  0.7638889  0.7476190
   4     0.8355241  0.7652778  0.7357143
   5     0.8124173  0.7481481  0.7547619
   6     0.7887566  0.7236111  0.6988095
   7     0.7938161  0.7439815  0.6892857
   8     0.8012235  0.7393519  0.6982143
   9     0.7932126  0.7578704  0.6767857
  10     0.7895916  0.7412037  0.6904762
  11     0.7868386  0.7416667  0.7101190
  12     0.7773479  0.7365741  0.7148810
  13     0.7715608  0.7375000  0.7000000
  14     0.7678902  0.7175926  0.6904762
  15     0.7679729  0.7148148  0.6898810

ROC was used to select the optimal model using the largest value.
The final value used for the model was ncomp = 3.
```

(b) We plot the results here:

```
ggplot(plsFit)
```



In this output the grid of results are the average resampled estimates of performance. The note at the bottom tells the user that 3 PLS components were found to be optimal. Based on this value, a final PLS model is fit to the whole data set using this specification and this is the model that is used to predict future samples.

(c) Prediction on the test data. To predict new samples, `predict.train` can be used. For classification models, the default behavior is to calculate the predicted class. The option `type = "prob"` can be used to compute class probabilities from the model.

```
plsClasses <- predict(plsFit, newdata = testing)
str(plsClasses)
 Factor w/ 2 levels "M","R": 2 1 1 1 2 2 1 2 2 2 ...
plsProbs <- predict(plsFit, newdata = testing, type = "prob")
head(plsProbs)
           M         R
6  0.2878477 0.7121523
8  0.6484796 0.3515204
9  0.6590434 0.3409566
15 0.5285970 0.4714030
26 0.4299690 0.5700310
27 0.4916393 0.5083607
```

(d) Confusion matrix

```
confusionMatrix(data = plsClasses, testing$Class)
Confusion Matrix and Statistics

          Reference
Prediction  M  R
         M 21  7
         R  6 17

               Accuracy : 0.7451
                 95% CI : (0.6037, 0.8567)
    No Information Rate : 0.5294
```

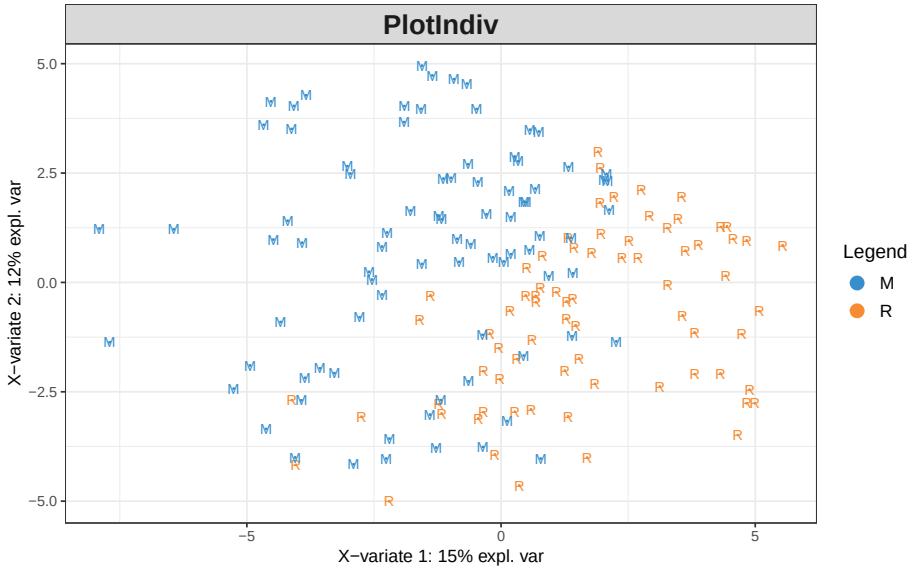
```
          P-Value [Acc > NIR] : 0.001311

                       Kappa : 0.4872

      Mcnemar's Test P-Value : 1.000000

                 Sensitivity : 0.7778
                 Specificity : 0.7083
              Pos Pred Value : 0.7500
              Neg Pred Value : 0.7391
                  Prevalence : 0.5294
              Detection Rate : 0.4118
        Detection Prevalence : 0.5490
           Balanced Accuracy : 0.7431

            'Positive' Class : M
```

(e) Using `mixOmics` package

```
library(mixOmics)
Ytrain <- training$Class
Xtrain <- as.matrix(training[,-61])
model <- mixOmics::plsda(Y=Ytrain ,X=Xtrain,ncomp=15)
```

(f) The projection samples on the two first component:

```
plotIndiv(model, ind.names = training$Class, ellipse = FALSE, legend = TRUE)
```



(g) We tune the number of component using cross-validation approach.

```
set.seed(45)
error <- perf(model, validation = "Mfold", folds = 10, dist="all", auc = TRUE,nrepeat = 3)
```

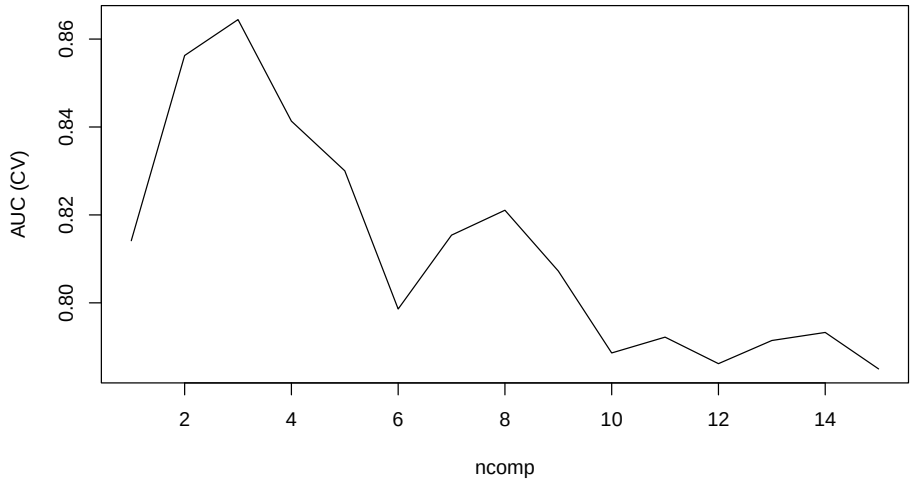
The results of the AUC are stored in

```
error$auc
$comp1
  AUC.mean     AUC.sd
0.81413333 0.00695006

$comp2
   AUC.mean      AUC.sd
0.856266667 0.004735328

$comp3
   AUC.mean      AUC.sd
0.864433333 0.004445597

$comp4
 AUC.mean    AUC.sd
0.8413333 0.0123314

$comp5
  AUC.mean     AUC.sd
0.83003333 0.00998816

$comp6
  AUC.mean     AUC.sd
0.79860000 0.02135158

$comp7
  AUC.mean     AUC.sd
0.81540000 0.01045323

$comp8
  AUC.mean     AUC.sd
0.82106667 0.00685298

$comp9
 AUC.mean    AUC.sd
0.8072333 0.0112269

$comp10
AUC.mean   AUC.sd
  0.7886   0.0004

$comp11
  AUC.mean     AUC.sd
0.79220000 0.02171704

$comp12
 AUC.mean    AUC.sd
0.7861667 0.0134318

$comp13
```

```
   AUC.mean      AUC.sd
0.791433333 0.007552704

$comp14
  AUC.mean     AUC.sd
0.79326667 0.01016923

$comp15
  AUC.mean     AUC.sd
0.78496667 0.01206496
```

We can plot it using:

```
plot(unlist(error$auc)[seq(1,30,by=2)],ylab="AUC (CV)",xlab="ncomp",type="l")
```

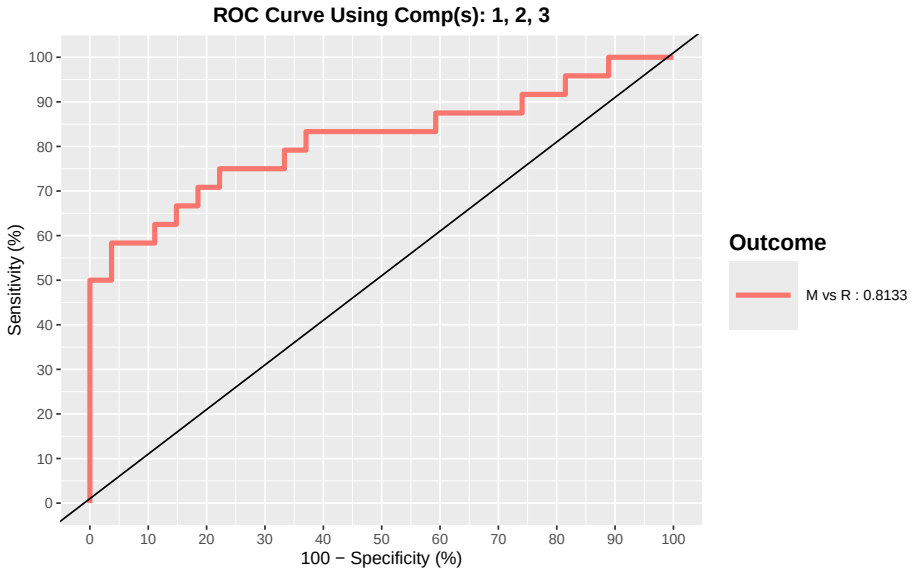


(h) We evaluate our model on the test set using the `centroids.dist` distance.

```
Ytest <- testing$Class
Xtest <- as.matrix(testing[,-61])
test.predict <- predict(model,newdata=Xtest,dist = "centroids.dist")
Prediction <- test.predict$class$centroids.dist[, 2]
table(Y = as.character(Ytest), Prediction)
   Prediction
Y    M  R
  M 21  6
  R  5 19
```

(i) Provide the roc curve evaluated on the test set using `auroc()` function

```
res <- auroc(model,newdata=Xtest,outcome.test=Ytest,roc.comp=3,print=FALSE)
```



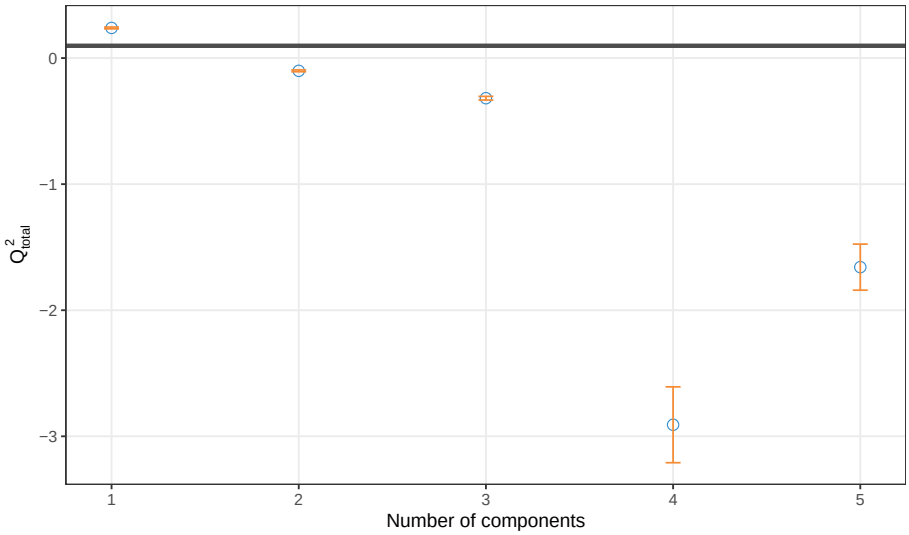
4 Solution: PLS2

(a) Run a PLS model with 6 Components

```
library(mixOmics)
data(liver.toxicity)
X <- liver.toxicity$gene
Y <- liver.toxicity$clinic
dim(X)
[1]   64 3116
dim(Y)
[1] 64 10
```

(b) Number of dimensions using the Q^2 criterion

```
tune.pls.liver <- pls(X = X, Y = Y, ncomp = 5, mode = 'regression')

set.seed(33)  # For reproducibility with this handbook, remove otherwise
Q2.pls.liver <- perf(tune.pls.liver, validation = 'Mfold', folds = 10,
                     nrepeat = 5)
plot(Q2.pls.liver, criterion = 'Q2.total')
```



This plots shows that one dimension should be sufficient in PLS2. We will include a second dimension in the graphical outputs, whilst focusing our interpretation on the first dimension.

(b) Run the model with 2 components

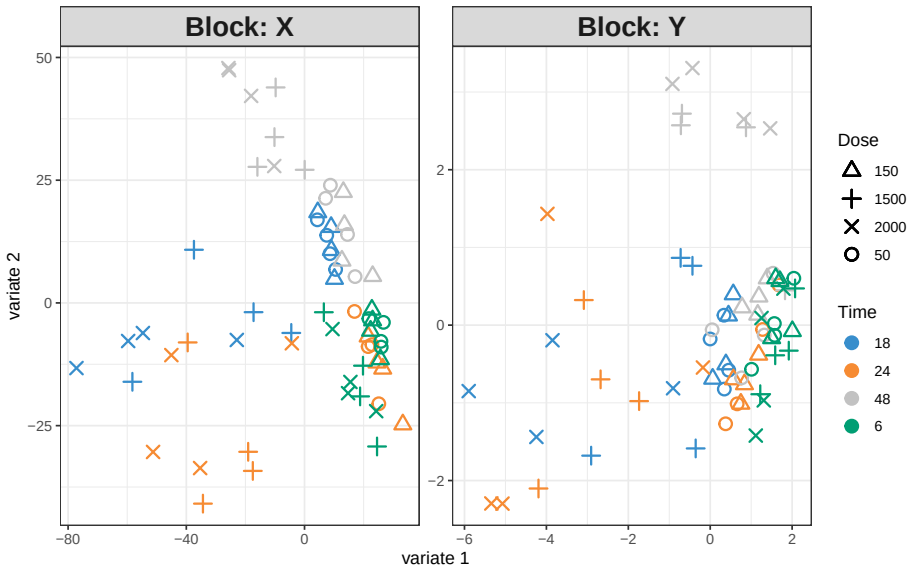
```
pls.liver <- pls(X, Y, ncomp = 2,
                 mode = "regression")
```

(c) The amount of explained variance can be extracted for each dimension and each data set:

```
pls.liver$prop_expl_var
$X
    comp1     comp2
0.2461708 0.1595936

$Y
    comp1     comp2
0.4070319 0.1803284
```

(d) Using the `plotIndiv()` function, we display the sample and metadata information using the arguments group (colour) and pch (symbol) to better understand the similarities between samples modelled with PLS2.

```
plotIndiv(pls.liver, ind.names = FALSE,
          group = liver.toxicity$treatment$Time.Group,
          pch = as.factor(liver.toxicity$treatment$Dose.Group),
          col.per.group = color.mixo(1:4),
          legend = TRUE, legend.title = 'Time',
          legend.title.pch = 'Dose')
```



Samples are projected into the space spanned by the components associated to each data set (or block). We observe some agreement between the data sets, and a separation of the 1500 and 2000 mg doses ($+$ and $\times$)
in the 18h, 24h time points, and the 48h time point.

We also observe an effect of low vs. high doses of acetaminophen (component 1) as well as time of necropsy (component 2). There is some level of agreement between the two data sets, but it is not perfect!

(e) Performance of the model. We provide the root mean square error evaluated by cross-validation for each component of Y.

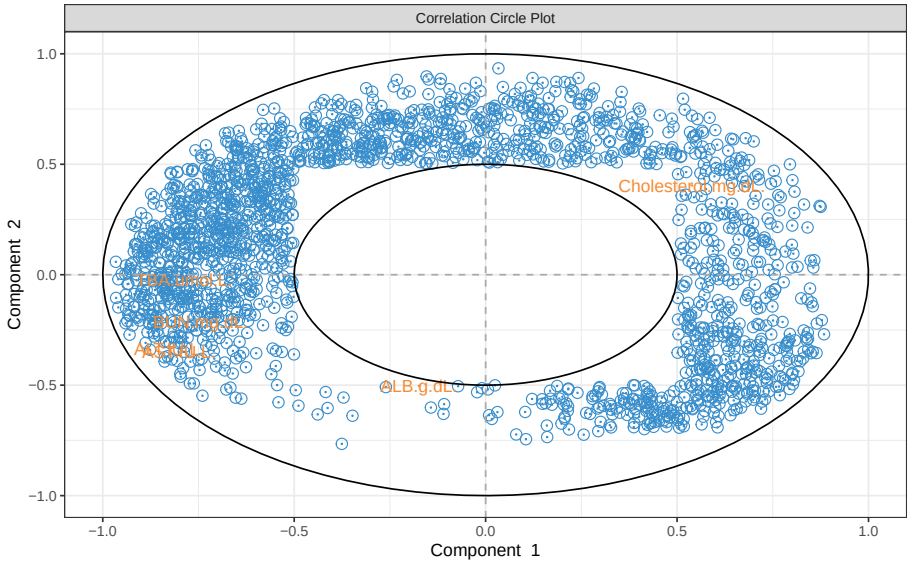
```
## PLS
perf.pls.liver <-  perf(pls.liver, validation = 'Mfold', folds = 10,
                        nrepeat = 5)
```

```
perf.pls.liver$measures$RMSEP$summary
              feature comp      mean          sd
1           ALB.g.dL.    1 1.0159146 0.008984257
2           ALB.g.dL.    2 1.0094319 0.008868516
3           ALP.IU.L.    1 0.9238382 0.010156017
4           ALP.IU.L.    2 0.9499867 0.014694009
5           ALT.IU.L.    1 0.6272040 0.010138777
6           ALT.IU.L.    2 0.5285712 0.013412777
7           AST.IU.L.    1 0.6586374 0.014934533
8           AST.IU.L.    2 0.5520485 0.017946659
9          BUN.mg.dL.    1 0.7001732 0.004767982
10         BUN.mg.dL.    2 0.7035315 0.015726603
11 Cholesterol.mg.dL.    1 0.8785277 0.011325991
12 Cholesterol.mg.dL.    2 0.8320630 0.007596465
13       Creat.mg.dL.    1 0.9871319 0.009985080
14       Creat.mg.dL.    2 1.0179872 0.017684603
15          SDH.IU.L.    1 1.0135889 0.015052044
16          SDH.IU.L.    2 1.0774257 0.039293167
```

```
17        TBA.umol.L.    1 0.6574753 0.010769027
18        TBA.umol.L.    2 0.6586374 0.024042230
19           TP.g.dL.    1 1.0252676 0.010640779
20           TP.g.dL.    2 1.0471932 0.018798523
```

(f) Correlation circle plot from the PLS2 performed with two components. The plot highlights correlations between genes and clinical parameters on each dimension of PLS2. We only provide genes and parameters with correlations greater than 0.5.

```
plotVar(pls.liver, cex = c(3,4), var.names = c(FALSE, TRUE),cutoff = 0.5)
```



This plot is very difficult to interpret due to the number of genes. We should try in a future analysis a sparse version of PLS to select the most relevant genes.