biosigner: A new method for signature discovery from omics data

Philippe Rinaudo and Etienne Thevenot

1 novembre 2016

Package version: biosigner 1.2.0

Contents

1 Introduction

High-throughput, non-targeted, technologies such as **transcriptomics**, **proteomics** and **metabolomics**, are widely used to **discover molecules** which allow to efficiently discriminate between biological or clinical conditions of interest (e.g., disease vs control states). Powerful **machine learning** approaches such as **Partial Least Square Discriminant Analysis** (PLS-DA), **Random Forest** (RF) and **Support Vector Machines** (SVM) have been shown to achieve high levels of prediction accuracy. **Feature selection**, i.e., the selection of the few features (i.e., the molecular signature) which are of highest discriminating value, is a critical step in building a robust and relevant classifier (Guyon and Elisseeff 2003): First, dimension reduction is usefull to **limit the risk of overfitting** and **increase the prediction stability** of the model; second, **intrepretation** of the molecular signature is facilitated; third, in case of the development of diagnostic product, a restricted list is required for the **subsequent validation** steps (Rifai, Gillette, and Carr 2006).

Since the comprehensive analysis of all combinations of features is not computationally tractable, several **selection techniques** have been described, including *filter* (e.g., *p*-values thresholding), *wrapper* (e.g., recursive feature elimination), and *embedded* (e.g., sparse PLS) approaches (Saeys, Inza, and Larranaga 2007). The major challenge for such methods is to be **fast** and extract **restricted and stable molecular signatures** which still provide **high performance** of the classifier (Gromski et al. 2014; Determan 2015).

2 The biosigner package

The biosigner package implements a new **wrapper** feature selection algorithm:

1. the dataset is split into training and testing subsets (by bootstraping, controling class proportion),
2. model is trained on the training set and balanced accuracy is evaluated on the test set,
3. the features are ranked according to their importance in the model,
4. the relevant feature subset at level f is found by a binary search: a feature subset is considered relevant if and only if, when randomly permuting the intensities of other features in the test subsets, the proportion of increased or equal prediction accuracies is lower than a defined threshold f,
5. the dataset is restricted to the selected features and steps 1 to 4 are repeated until the selected list of features is stable.

Three binary classifiers have been included in biosigner, namely **PLS-DA**, **RF** and **SVM**, as the performances of each machine learning approach may vary depending on the structure of the dataset (Determan 2015). The algorithm returns the **tier** of each feature for the selected classifer(s): tier **S** corresponds to the **final signature**, i.e., features which have been found significant in all the selection steps; features with tier *A* have been found significant in all but the last selection, and so on for tier *B* to *D*. Tier *E* regroup all previous round of selection.

As for a classical classification algorithm, the `biosign` method takes as input the `x` samples times features data frame (or matrix) of intensities, and the `y` factor (or character vector) of class labels (note that only binary classification is currently available). It returns the signature (`signatureLs`: selected feature names) and the trained model (`modelLs`) for each of the selected classifier. The `plot` method for `biosign` objects enable to visualize the individual boxplots of the selected features. Finally, the `predict` method allows to apply the trained classifier(s) on new datasets.

The algorithm has been successfully applied to **transcriptomics** and **metabolomics** data [Rinaudo et al. (2016); see also the *Hands-on* section below).

3 Hands-on

3.1 Loading

We first load the biosigner package:

```
library(biosigner)
```

We then use the **diaplasma** metabolomics dataset (Rinaudo et al. 2016) which results from the analysis of **plasma samples from 69 diabetic patients** were analyzed by reversed-phase liquid chromatography coupled to high-resolution mass spectrometry (**LC-HRMS**; Orbitrap Exactive) in the negative ionization mode. The raw data were pre-processed with XCMS and CAMERA (5,501 features), corrected for signal drift, log10 transformed, and annotated with an in-house spectral database. The patient's **age**, **body mass index**, and **diabetic type** are recorded (Rinaudo et al. 2016).

```
data(diaplasma)
```

We attach *diaplasma* to the search path and display a summary of the content of the *dataMatrix*, *sampleMetadata* and *variableMetadata* with the `strF` function from the (imported) ropls package:

```
attach(diaplasma)
library(ropls)
strF(dataMatrix)
##         dim  class    mode typeof   size NAs min mean median max
##  69 x 5,501 matrix numeric double 3.2 Mb   0   0  4.2    4.4 8.2
##           m096.009t01.6    m096.922t00.8 ...    m995.603t10.2
## DIA001 2.98126177377087 6.08172882312848 ... 3.93442594703862
## DIA002                0 6.13671997362279 ... 3.74201112636229
```

```
## ...                 ...              ... ...              ...
## DIA077                0 6.12515971273103 ... 4.55458598372024
## DIA078 4.69123816772499   6.134420482337 ...  4.1816445335704
##          m995.613t10.2
## DIA001 3.96424920154706
## DIA002 3.78128422428722
## ...                 ...
## DIA077 4.57310800324247
## DIA078 4.20696191303494
strF(sampleMetadata)
##    type     age     bmi
##  factor numeric numeric
##  nRow nCol size NAs
##    69    3 0 Mb   0
##        type age  bmi
## DIA001   T2  70 31.6
## DIA002   T2  67   28
## ...     ... ...  ...
## DIA077   T2  50   27
## DIA078   T2  65   29
strF(variableMetadata)
##    mzmed   rtmed ... pcgroup     spiDb
##  numeric numeric ... numeric character
##   nRow nCol   size NAs
##  5,501    6 0.7 Mb   0
##                     mzmed       rtmed ... pcgroup
## m096.009t01.6 96.00899361 93.92633015 ...    1984
## m096.922t00.8 96.92192011 48.93274877 ...       4
## ...                   ...         ... ...     ...
## m995.603t10.2 995.6030195 613.4388762 ...    7160
## m995.613t10.2 995.6134422 613.4446705 ...    7161
##                                           spiDb
## m096.009t01.6 N-Acetyl-L-aspartic acid_HMDB00812
## m096.922t00.8
## ...                                         ...
## m995.603t10.2
## m995.613t10.2
```

We see that the **diaplasma** list contains three objects:

1. **dataMatrix**: 69 samples x 5,501 matrix of numeric type containing the intensity profiles (log10 transformed),
2. **sampleMetadata**: a 69 x 3 data frame, with the patients'
   - **type**: diabetic type, factor
   - **age**: numeric
   - **bmi**: body mass index, numeric
3. **variableMetadata**: a 5,501 x 8 data frame, with the median m/z ('mzmed', numeric) and the median retention time in seconds ('rtmed', numeric) from XCMS, the 'isotopes' (character), 'adduct' (character) and 'pcgroups' (numeric) annotations from CAMERA, the names of the m/z and RT matching compounds from an in-house spectra of commercial metabolites ('name_hmdb', character), and the *p*-values resulting from the non-parametric hypothesis testing of difference in medians between types ('type_wilcox_fdr', numeric), and correlation with age ('age_spearman_fdr', numeric) and body mass index ('bmi_spearman_fdr', numeric), all corrected for multiple testing (False Discovery Rate).

We can observe that the 3 clinical covariates (diabetic *type*, *age*, and *bmi*) are stronlgy associated:

```
with(sampleMetadata,
plot(age, bmi, cex = 1.5, col = ifelse(type == "T1", "blue", "red"), pch = 16))
legend("topleft", cex = 1.5, legend = paste0("T", 1:2),
text.col = c("blue", "red"))
```

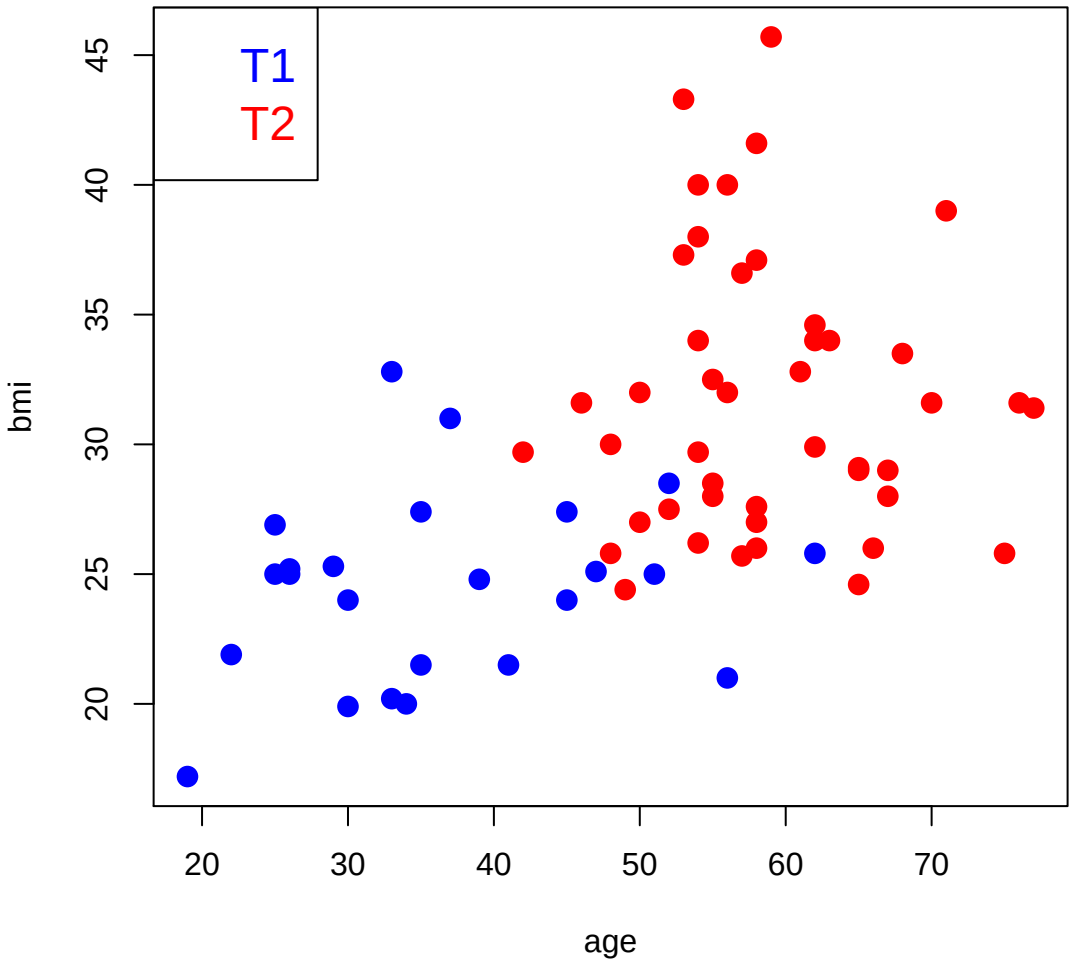


Figure 1: *age*, *body mass index (bmi)*, and diabetic *type* of the patients from the *diaplasma* cohort.

3.2 Molecular signatures

Let us look for signatures of *type* in the *diaplasma* dataset by using the `biosign` method. To speed up computations in this demo vignette, we restrict the number of features (from 5,501 to about 500) and the number of bootstraps (5 instead of 50 [default]); the selection on the whole dataset, 50 bootstraps, and the 3 classifiers, takes around 10 min.

```
featureSelVl <- variableMetadata[, "mzmed"] >= 450 &
variableMetadata[, "mzmed"] < 500
sum(featureSelVl)
## [1] 533
```

```
dataMatrix <- dataMatrix[, featureSelVl]
variableMetadata <- variableMetadata[featureSelVl, ]
```

```
set.seed(123)
diaSign <- biosign(dataMatrix, sampleMetadata[, "type"], bootI = 5)
set.seed(NULL)
```

```
## Significant features from 'S' groups:
##               plsda randomforest svm
## m471.241t07.6 "S"   "B"          "A"
## m497.284t08.1 "S"   "S"          "E"
## m495.261t08.7 "A"   "E"          "S"
## m497.275t08.1 "S"   "A"          "E"
## Accuracy:
##      plsda randomforest   svm
## Full 0.759        0.742 0.808
## AS   0.832        0.784 0.780
## S    0.911        0.738 0.752
```

The arguments are:

- `x`: the numerical matrix (or data frame) of intensities (samples as rows, variables as columns),
- `y`: the factor (or character) specifying the sample labels from the 2 classes,
- `methodVc`: the classifier(s) to be used; here, the default *all* value means that all classifiers available (*plsda*, *randomforest*, and *svm*) are selected,
- `bootI`: the number of bootstraps is set to 5 to speed up computations when generating this vignette; we however recommend to keep the default 50 value for your analyzes (otherwise signatures may be less stable).

Note:

1. If some features from the `x` matrix/data frame contain missing values (NA), these features will be removed prior to modeling with Random Forest and SVM (in contrast, the NIPALS algorithm from PLS-DA can handle missing values),
2. The `set.seed` command was used here to be sure that the results from this vignette can be reproduced exactly; by choosing alternative seeds (and the default `bootI` = 50), similar signatures are obtained, showing the stability of the selection.

The resulting signatures for the 3 selected classifiers are both printed and plotted as **tiers** from *S*, *A*, up to *E* by decreasing relevance. The (*S*) tier corresponds to the final signature, i.e. features which passed through all the backward selection steps. In contrast, features from the other tiers were discarded during the last (*A*) or previous (*B* to *E*) selection rounds.

Note that *tierMaxC* = *'A'* argument in the *print* and *plot* methods can be used to view the features from the larger *S+A* signatures (especially when no *S* features have been found, or when the performance of the *S* model is much lower than the *S+A* model).

The **performance** of the model built with the input dataset (*balanced accuracy*: mean of the *sensitivity* and *specificity*), or the subset restricted to the *S* or *S+A* signatures are shown. We see that with 1 to 5 *S* feature signatures (i.e., less than 1% of the input), the 3 classifiers achieve good performances (even higher than the full Random Forest and SVM models). Furthermore, reducing the number of features decreases the risk of building non-significant models (i.e., models which do not perform significantly better than those built after randomly permuting the labels). The signatures from the 3 classifiers have some distinct features, which highlights the interest of comparing various machine learning approaches.

The **individual boxplots** of the features from the *complete* signature can be visualized with:

```
plot(diaSign, typeC = "boxplot")
```

Let us see the metadata of the *complete* signature:

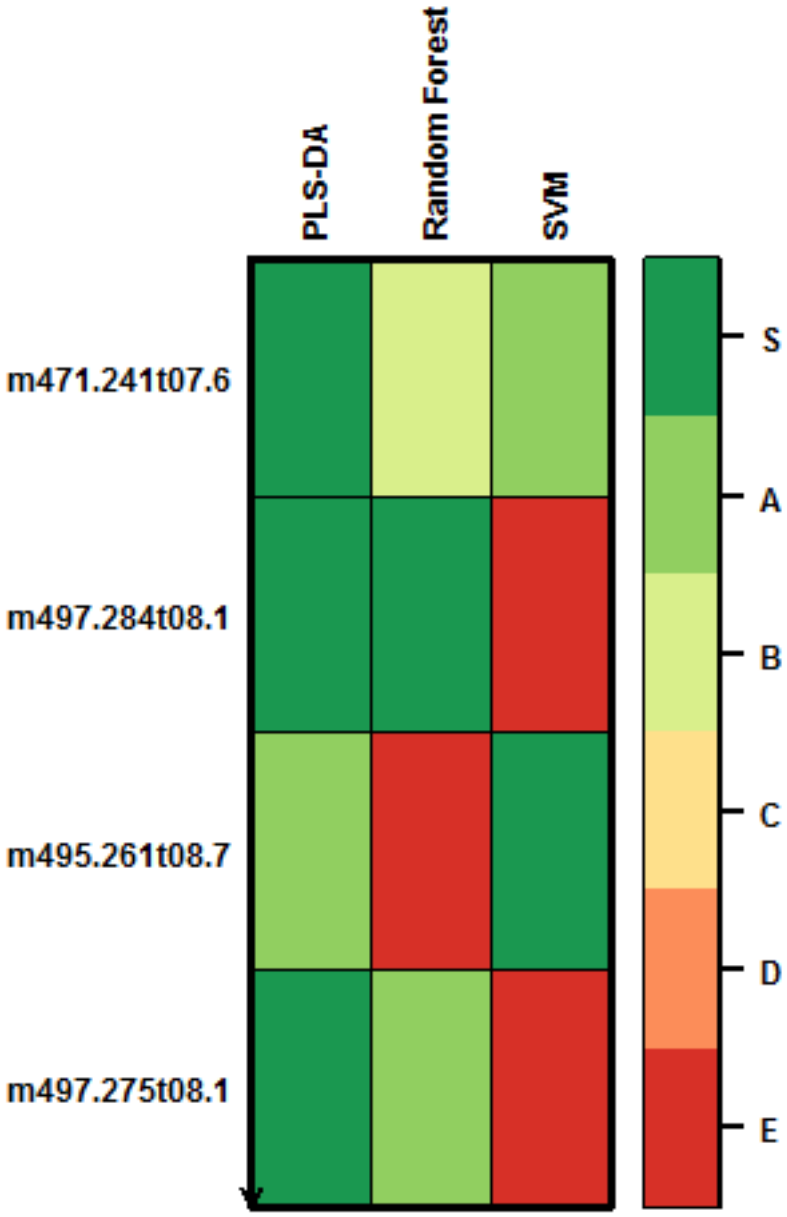


Figure 1: **Figure 2:** Relevant signatures for the *PLS-DA*, *Random Forest*, and *SVM* classifiers extracted from the diaplasma dataset. The *S* tier corresponds to the final metabolite signature, i.e., metabolites which passed through all the selection steps.

```
variableMetadata[getSignatureLs(diaSign)[["complete"]], ]
##                   mzmed    rtmed isotopes                        adduct
## m471.241t07.6 471.2408 455.5541
## m497.284t08.1 497.2840 486.5338          [M+Cl]- 462.31 [M-H]- 498.287
## m495.261t08.7 495.2609 524.1249
## m497.275t08.1 497.2755 486.5722          [M+Cl]- 462.31 [M-H]- 498.287
##               pcgroup                                 spiDb
## m471.241t07.6   10538
```

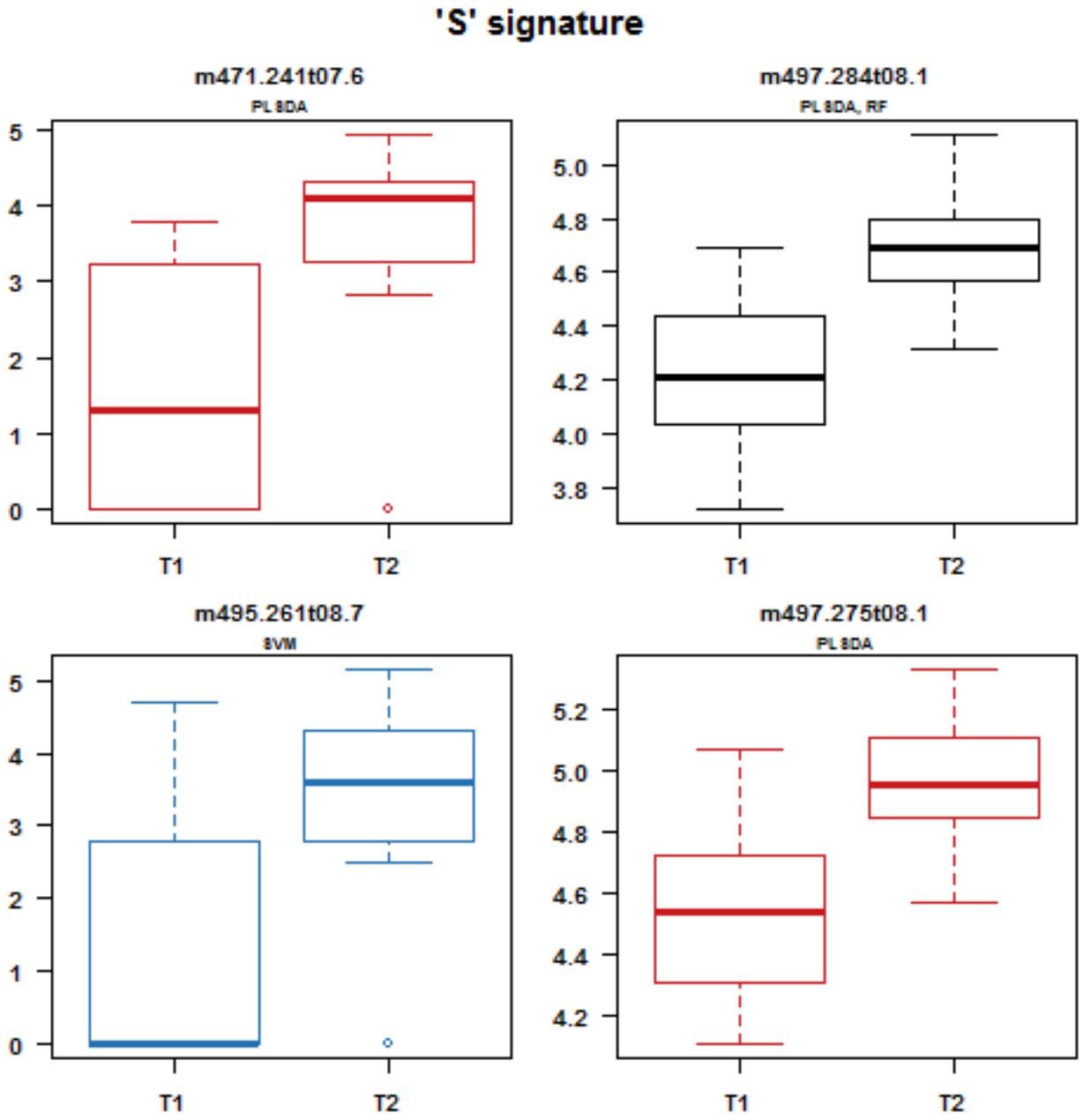


Figure 2: **Figure 3:** Individual boxplots of the features selected for at least one of the classification methods. Features selected for a single classifier are colored (*red* for PLS-DA, *green* for Random Forest and *blue* for SVM).

```
## m497.284t08.1     220
## m495.261t08.7    1655
## m497.275t08.1     220 Taurochenodeoxycholic acid_HMDB00951
```

We observe that the *taurochenodeoxycholic acid* has been annotated, in addition to another [M-H]- ion at 470.233 Da. Six out of the 8 features are very significant by univariate hypothesis testings of difference between *type* medians, and to a lesser extent, of the correlation with *age* and *body mass index*.

3.3 Predictions

Let us split the dataset into a training (the first 4/5th of the 183 samples) and a testing subsets, and extract the relevant features from the training subset:

```
trainVi <- 1:floor(0.8 * nrow(dataMatrix))
testVi <- setdiff(1:nrow(dataMatrix), trainVi)
```

```
set.seed(123)
diaTrain <- biosign(dataMatrix[trainVi, ], sampleMetadata[trainVi, "type"],
bootI = 5)
set.seed(NULL)
```

```
## Significant features from 'S' groups:
##                plsda randomforest svm
## m471.241t07.6 "B"   "A"          "S"
## m497.284t08.1 "S"   "S"          "D"
## m456.182t12.9 "E"   "E"          "S"
## m493.243t07.7 "E"   "E"          "S"
## Accuracy:
##      plsda randomforest   svm
## Full 0.794        0.777 0.793
## AS   0.827        0.848 0.783
## S    0.843        0.793 0.840
```

We extract the **fitted** types on the training dataset restricted to the *S* signatures:

```
diaFitDF <- predict(diaTrain)
```

We then print the confusion tables for each classifier:

```
lapply(diaFitDF, function(predFc) table(actual = sampleMetadata[trainVi,
"type"], predicted = predFc))
## $plsda
##       predicted
## actual T1 T2
##     T1 16  6
##     T2  4 29
##
## $randomforest
##       predicted
## actual T1 T2
##     T1 14  8
##     T2  7 26
##
## $svm
##       predicted
## actual T1 T2
##     T1 18  4
##     T2  4 29
```

and the corresponding balanced accuracies:

```
sapply(diaFitDF, function(predFc) {
conf <- table(sampleMetadata[trainVi, "type"], predFc)
conf <- sweep(conf, 1, rowSums(conf), "/")
round(mean(diag(conf)), 3)
})
```

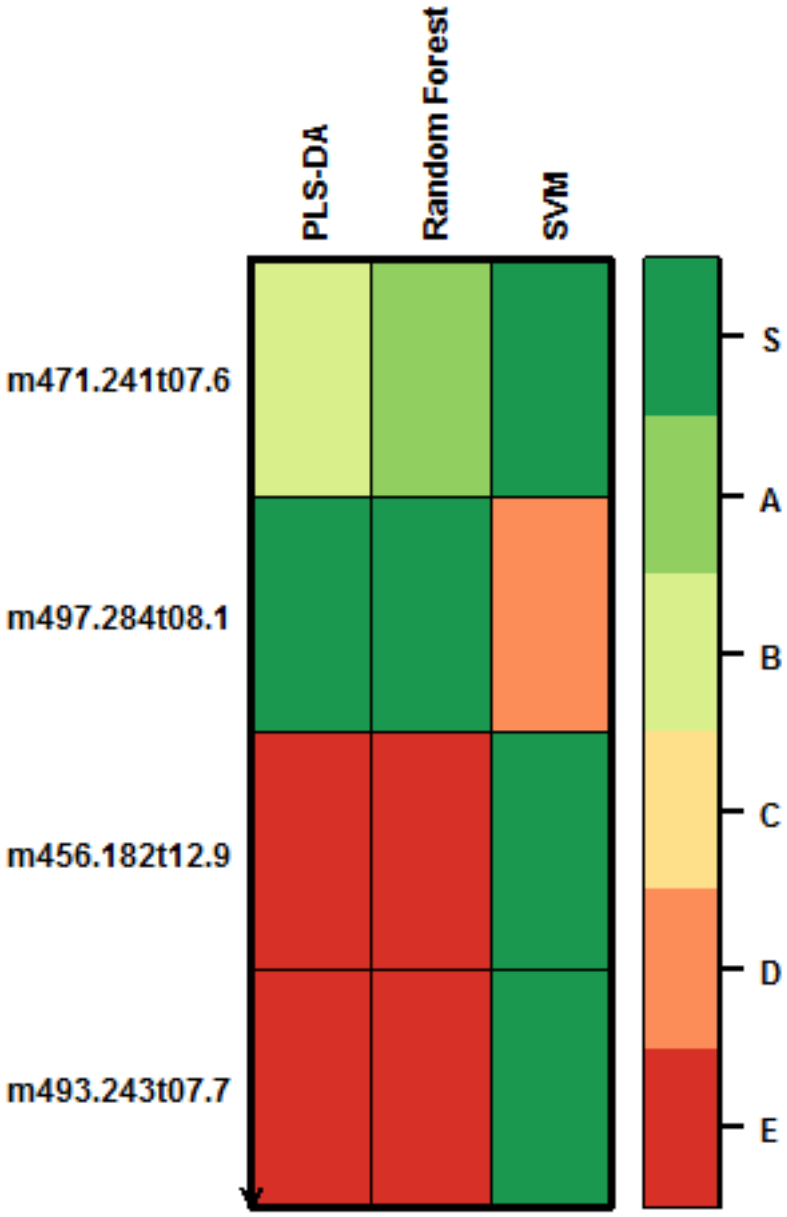


Figure 3: **Figure 4:** Signatures from the training data set

```
##        plsda randomforest          svm
##        0.803        0.712        0.848
```

Note that these values are slightly different from the accuracies returned by *biosign* because the latter are computed by using the resampling scheme selected by the *bootI* (or *crossvalI*) arguments:

```
round(getAccuracyMN(diaTrain)["S", ], 3)
##        plsda randomforest          svm
##        0.843        0.793        0.840
```

Finally, we can compute the performances on the test subset:

```
diaTestDF <- predict(diaTrain, newdata = dataMatrix[testVi, ])
sapply(diaTestDF, function(predFc) {
conf <- table(sampleMetadata[testVi, "type"], predFc)
conf <- sweep(conf, 1, rowSums(conf), "/")
round(mean(diag(conf)), 3)
})
##        plsda randomforest          svm
##        0.750        0.667        0.750
```

3.4 Working on *ExpressionSet* omics objects from bioconductor

The **ExpressionSet** class from the Biobase bioconductor package has been developed to conveniently handle preprocessed omics objects, including the variables x samples matrix of intensities, and data frames containing the sample and variable metadata (Huber et al. 2015). The matrix and the two data frames can be accessed by the `exprs`, `pData` and `fData` respectively (note that the data matrix is stored in the object with samples in columns).

The `biosign` method can be applied to an *ExpressionSet* object, by using the object as the `x` argument, and by indicating as the `y` argument the name of the phenoData column to be used as the response.

In the example below, we will first build a minimal *ExpressionSet* object from the *diaplasma* data set, and we subsequently perform signature discovery:

```
library(Biobase)
diaSet <- ExpressionSet(assayData = t(dataMatrix),
phenoData = new("AnnotatedDataFrame", data = sampleMetadata))
set.seed(123)
biosign(diaSet, "type", bootI = 5)
set.seed(NULL)
```

```
## Significant features from 'S' groups:
##                plsda randomforest svm
## m471.241t07.6 "S"   "B"          "A"
## m497.284t08.1 "S"   "S"          "E"
## m495.261t08.7 "A"   "E"          "S"
## m497.275t08.1 "S"   "A"          "E"
## Accuracy:
##      plsda randomforest   svm
## Full 0.759        0.742 0.808
## AS   0.832        0.784 0.780
## S    0.911        0.738 0.752
```

Note: Because of identical names in the two packages, the `combine` method from package Biobase is replaced by the `combine` method from package randomForest.

Before moving to new data sets, we detach *diaplasma* from the search path:

```
detach(diaplasma)
```

4 Extraction of biomarker signatures from other omics datasets

In this section, `biosign` is applied to two metabolomics and one transcriptomics data sets. Please refer to Rinaudo et al. (2016) for a full discussion of the methods and results.

4.1 Physiological variations of the human urine metabolome (metabolomics)

The **sacurine** LC-HRMS dataset from the dependent ropls package can also be used (Thevenot et al. 2015): Urine samples from a cohort of 183 adults were analyzed by using an LTQ Orbitrap in the negative ionization mode. A total of 109 metabolites were identified or annotated at the MSI level 1 or 2. Signal drift and batch effect were corrected, and each urine profile was normalized to the osmolality of the sample. Finally, the data were log10 transformed (see the ropls vignette for further details and examples).

We can for instance look for signatures of the *gender*:

```
data(sacurine)
set.seed(123) ##
sacSign <- biosign(sacurine[["dataMatrix"]],
sacurine[["sampleMetadata"]][, "gender"],
methodVc = "plsda")
set.seed(NULL)
```

```
## Significant features from 'S' groups:
##                            plsda
## Malic acid                 "S"
## p-Anisic acid              "S"
## Testosterone glucuronide "S"
## Accuracy:
##      plsda
## Full 0.870
## AS   0.861
## S    0.879
```

4.2 Apples spikes with known compounds (metabolomics)

The **spikedApples** dataset was obtained by LC-HRMS analysis (SYNAPT Q-TOF, Waters) of one control and three spiked groups of 10 apples each. The spiked mixtures consists in 2 compounds which were not naturally present in the matrix and 7 compounds aimed at achieving a final increase of 20%, 40% or 100% of the endogeneous concentrations. The authors identified 22 features (out of the 1,632 detected in the positive ionization mode; i.e. 1.3%) which came from the spiked compounds. The dataset is included in the BioMark R package (Franceschi et al. 2012). Let us use the *control* and *group1* samples (20 in total) in this study.

```
library(BioMark)
data(SpikePos)
group1Vi <- which(SpikePos[["classes"]] %in% c("control", "group1"))
appleMN <- SpikePos[["data"]][group1Vi, ]
spikeFc <- factor(SpikePos[["classes"]][group1Vi])
annotDF <- SpikePos[["annotation"]]
rownames(annotDF) <- colnames(appleMN)
```

We can check, by using the `opls` method from the ropls package for multivariate analysis, that:

1. no clear separation can be observed by PCA:

```
biomark.pca <- opls(appleMN, plotL = FALSE)
plot(biomark.pca, parAsColFcVn = spikeFc)
```

```
## PCA
## 20 samples x 1632 variables
## standard scaling of predictors
##       R2X(cum) pre ort
```

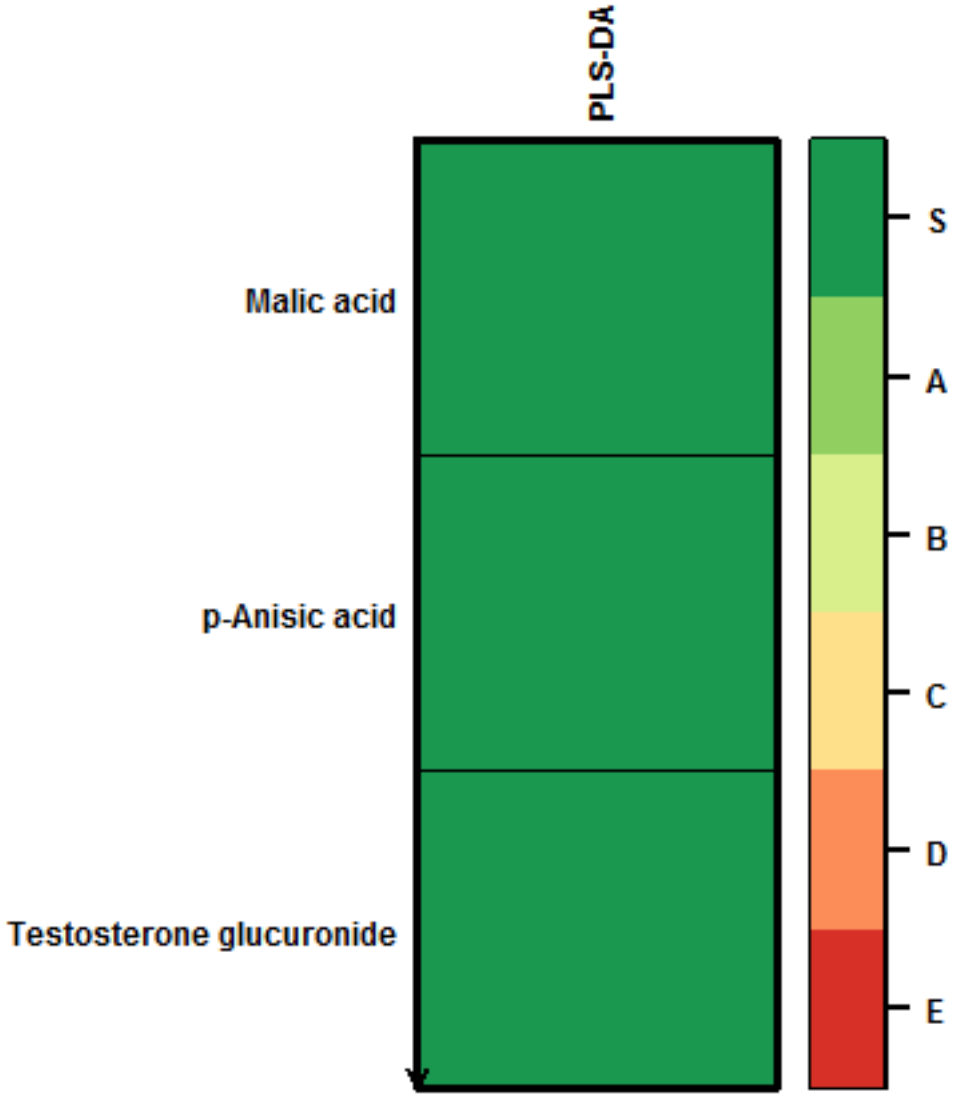
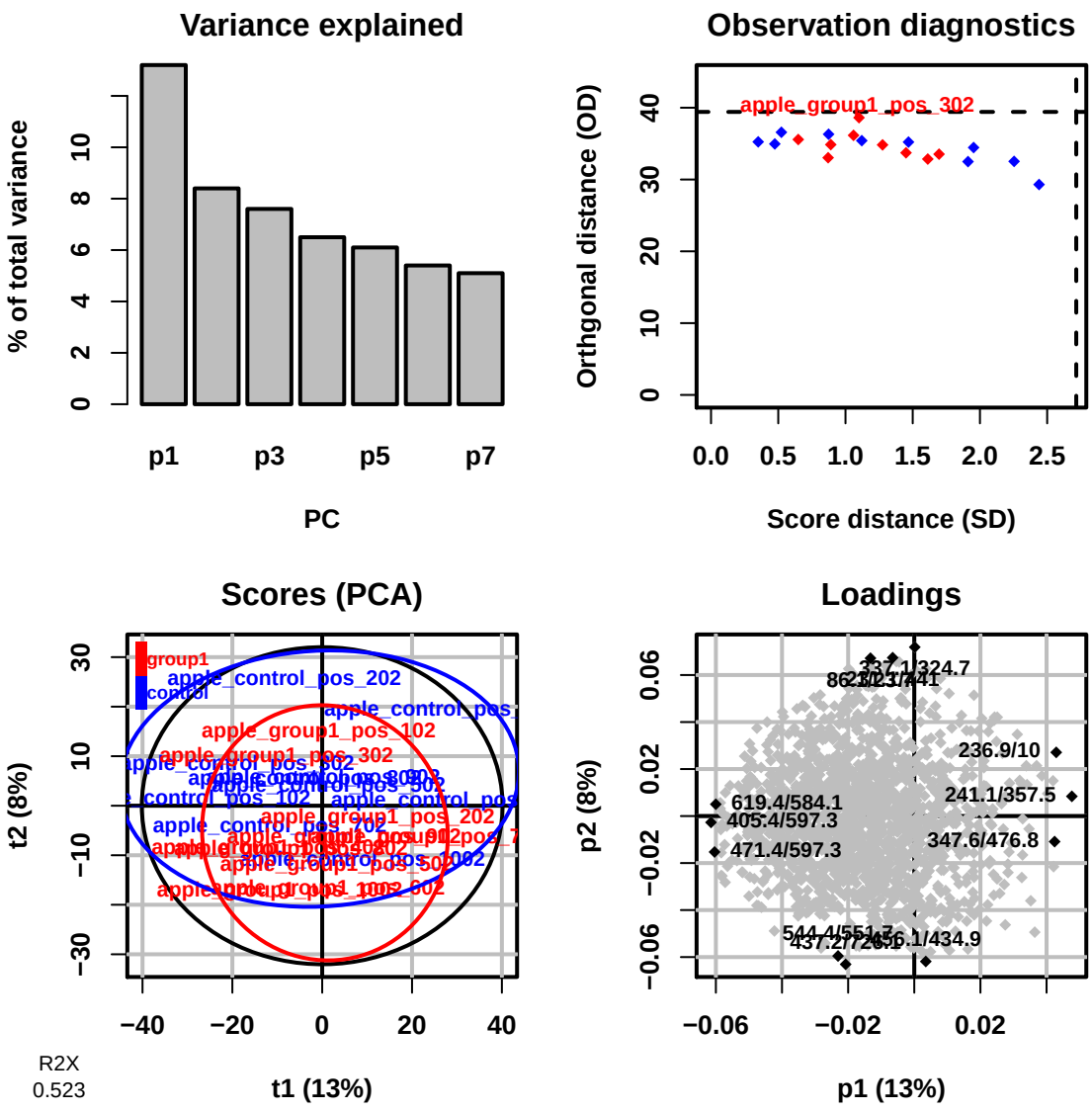


Figure 4: **Figure 5:** PLS-DA signature from the 'sacurine' data set.

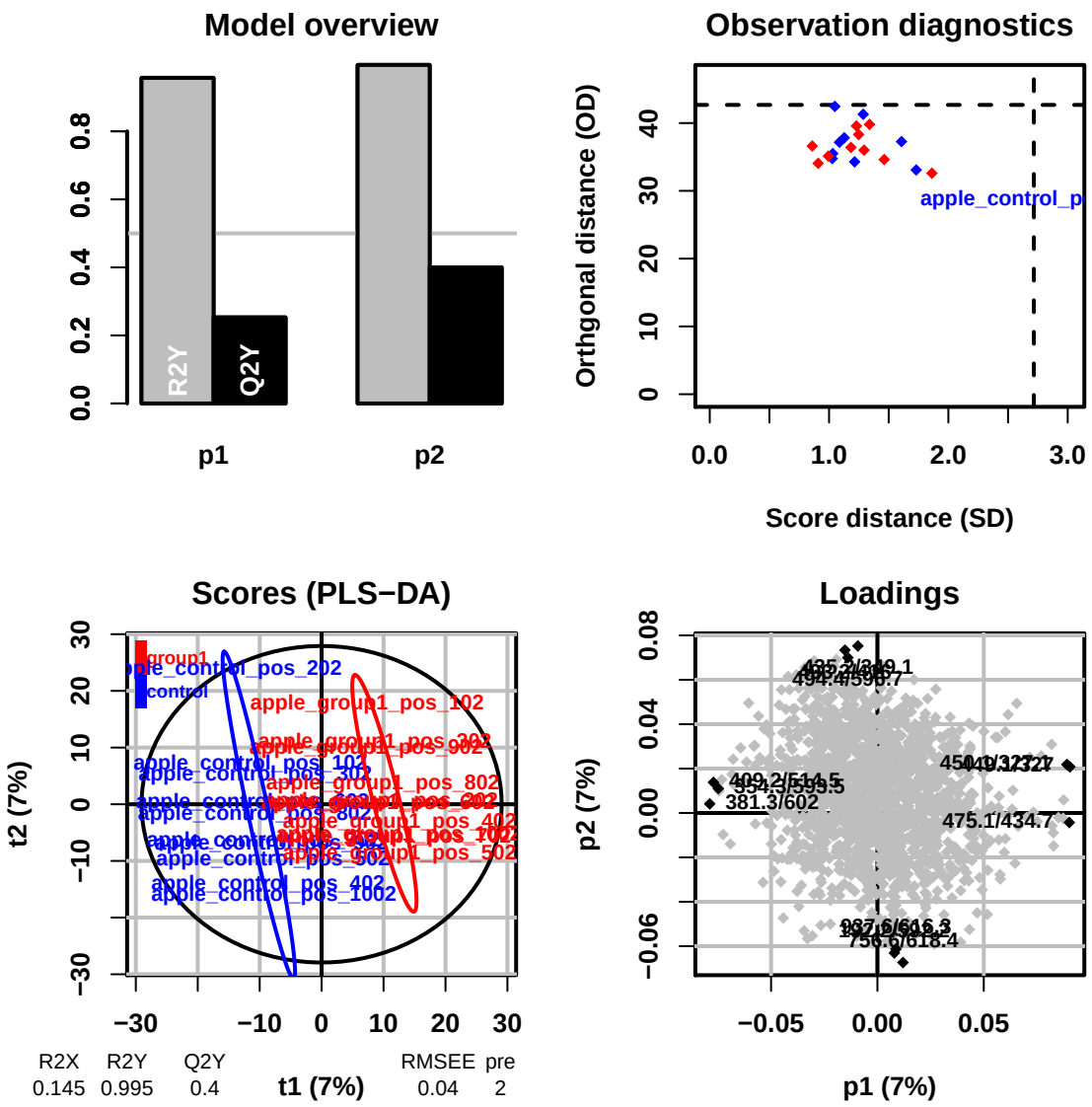
```
## Total    0.523   7   0
```



2. PLS-DA modeling with the full dataset is not significant (as seen on the top left plot: 7 out of 20 models trained after random permutations of the labels have Q2 values higher than the model trained with the true labels):

```
biomark.pls <- opls(appleMN, spikeFc)
```

```
## PLS-DA
## 20 samples x 1632 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum)  RMSEE pre ort pR2Y pQ2
## Total    0.145    0.995     0.4 0.0396   2   0 0.25 0.4
```



Let us now extract the molecular signatures:

```
set.seed(123)
appleSign <- biosign(appleMN, spikeFc)
set.seed(NULL)
```

The *449.1/327* corresponds to the Cyanidin-3-galactoside (absent in the control) and the *475.1/434.7* is probably a potassium adduct of the Phloridzin (80% concentration increase in group1; Franceschi et al. (2012)).

```
annotDF <- SpikePos[["annotation"]]
rownames(annotDF) <- colnames(appleMN)
annotDF[getSignatureLs(appleSign)[["complete"]], c("adduct", "found.in.standards")]
```

4.3 Bone marrow from acute leukemia patients (transcriptomics)

Samples from 47 patients with acute lymphoblastic leukemia (ALL) and 25 patients with acute myeloid leukemia (AML) have been analyzed using Affymetrix Hgu6800 chips resulting in expression data of 7,129 gene probes (Golub et al. 1999).

The **golub** dataset is available in the golubEsets package from Bioconductor. Let us compute for example the SVM signature (to speed up this demo example, the number of features is restricted to 500):

```
library(golubEsets)
data(Golub_Merge)
golubMN <- t(exprs(Golub_Merge))
leukemiaFc <- pData(Golub_Merge)[["ALL.AML"]]
table(leukemiaFc)
varSubVi <- 1501:2000
set.seed(123)
golubSign <- biosign(golubMN[, varSubVi], leukemiaFc, methodVc = "svm")
set.seed(NULL)
```

```
## leukemiaFc
## ALL AML
##  47  25
## Significant features from 'S' groups:
##           svm
## M19507_at "S"
## M27891_at "S"
## Accuracy:
##        svm
## Full 0.950
## AS   0.969
## S    0.944
```

The computation results in a signature of 2 features only and a sparse SVM model performing almost as well (94.4% accuracy) as the model trained on the dataset of 500 variables (95.0% accuracy).

The hu6800.db bioconductor package can be used to get the annotation of the selected probes (Carlson 2016):

```
library(hu6800.db)
sapply(getSignatureLs(golubSign)[["complete"]],
       function(probeC)
       get(probeC, env = hu6800GENENAME))
##         M19507_at         M27891_at
## "myeloperoxidase"      "cystatin C"
```

Cystatin C is part of the 50 gene signature selected by Golub and colleagues on the basis of a metric derived from the Student's statistic of mean differences between the AML and ALL groups (Golub et al. 1999). Interestingly, the second probe, myeloperoxidase, is a cytochemical marker for the diagnosis (and also potentially the prognosis) of acute myeloid leukemia (AML).

5 Session info

Here is the output of `sessionInfo()` on the system on which this document was compiled:

```
## R version 3.3.1 (2016-06-21)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## locale:
## [1] LC_COLLATE=French_France.1252  LC_CTYPE=French_France.1252
## [3] LC_MONETARY=French_France.1252 LC_NUMERIC=C
## [5] LC_TIME=French_France.1252
```

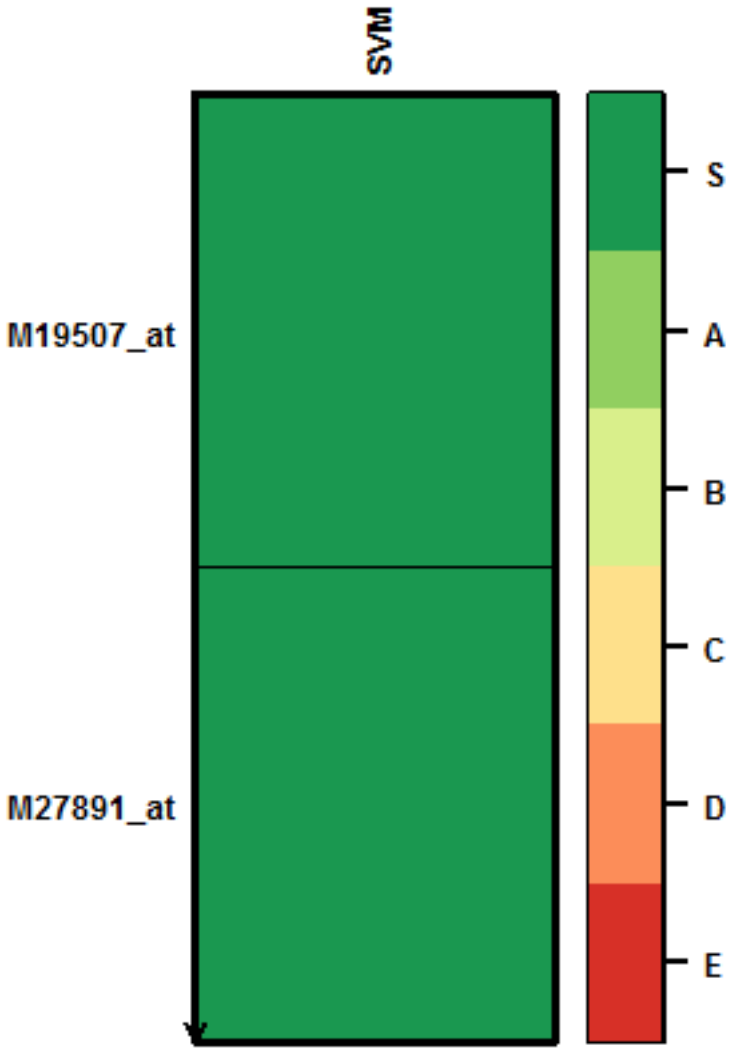


Figure 5: **Figure 6:** SVM signature from the *golub* data set.

```
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] hu6800.db_3.2.3      org.Hs.eg.db_3.4.0   AnnotationDbi_1.36.0
##  [4] IRanges_2.8.0        S4Vectors_0.12.0     golubEsets_1.16.0
##  [7] BioMark_0.4.5        st_1.2.5             sda_1.3.7
## [10] fdrtool_1.2.15       corpcor_1.6.8        entropy_1.2.1
```

```
## [13] MASS_7.3-45          glmnet_2.0-5         foreach_1.4.3
## [16] Matrix_1.2-7.1       pls_2.5-0            Biobase_2.34.0
## [19] BiocGenerics_0.20.0  ropls_1.6.0          biosigner_1.2.0
## [22] BiocStyle_2.2.0      devtools_1.12.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.7           BiocInstaller_1.24.0 formatR_1.4
##  [4] class_7.3-14          iterators_1.0.8      tools_3.3.1
##  [7] digest_0.6.10         RSQLite_1.0.0        evaluate_0.10
## [10] memoise_1.0.0         tibble_1.2           lattice_0.20-34
## [13] DBI_0.5-1             yaml_2.1.13          e1071_1.6-7
## [16] withr_1.0.2           stringr_1.1.0        knitr_1.14
## [19] grid_3.3.1            rmarkdown_1.1        magrittr_1.5
## [22] codetools_0.2-15      htmltools_0.3.5      randomForest_4.6-12
## [25] assertthat_0.1        stringi_1.1.2
```

References

Carlson, M. 2016. *Hu6800.db: Affymetrix Hugenefl Genome Array Annotation Data (Chip Hu6800)*.

Determan, CE. 2015. "Optimal Algorithm for Metabolomics Classification and Feature Selection Varies by Dataset." *International Journal of Biology* 7 (1): 100–115. doi:10.5539/ijb.v7n1p100.

Franceschi, P., D. Masuero, U. Vrhovsek, F. Mattivi, and R. Wehrens. 2012. "A Benchmark Spike-in Data Set for Biomarker Identification in Metabolomics." *Journal of Chemometrics* 26 (1-2): 16–24. doi:10.1002/cem.1420.

Golub, TR., DK. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, JP. Mesirov, H. Coller, et al. 1999. "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring." *Science* 286 (5439): 531. doi:10.1126/science.286.5439.531.

Gromski, PS., Y. Xu, E. Correa, DI. Ellis, ML. Turner, and R. Goodacre. 2014. "A Comparative Investigation of Modern Feature Selection and Classification Approaches for the Analysis of Mass Spectrometry Data." *Analytica Chimica Acta* 829 (0): 1–8. doi:10.1016/j.aca.2014.03.039.

Guyon, I., and A. Elisseeff. 2003. "An Introduction to Variable and Feature Selection." *Journal of Machine Learning Research* 3: 1157–82.

Huber, W., VJ. Carey, R. Gentleman, S. Anders, M. Carlson, BS. Carvalho, HC. Bravo, et al. 2015. "Orchestrating High-Throughput Genomic Analysis with Bioconductor." *Nature Methods* 12 (2): 115–21. doi:10.1038/nmeth.3252.

Rifai, N., MA. Gillette, and SA. Carr. 2006. "Protein Biomarker Discovery and Validation: The Long and Uncertain Path to Clinical Utility." *Nature Biotechnology*. doi:10.1038/nbt1235.

Rinaudo, P., S. Boudah, C. Junot, and EA. Thevenot. 2016. "Biosigner: A New Method for the Discovery of Significant Molecular Signatures from Omics Data." *Frontiers in Molecular Biosciences* 3. doi:10.3389/fmolb.2016.00026.

Saeys, Y., I. Inza, and P. Larranaga. 2007. "A Review of Feature Selection Techniques in Bioinformatics." *Bioinformatics* 23 (19): 2507–17. doi:10.1093/bioinformatics/btm344.

Thevenot, EA., A. Roux, Y. Xu, E. Ezan, and C. Junot. 2015. "Analysis of the Human Adult Urinary Metabolome Variations with Age, Body Mass Index, and Gender by Implementing a Comprehensive Workflow for Univariate and OPLS Statistical Analyses." *Journal of Proteome Research* 14 (8): 3322–35. doi:10.1021/acs.jproteome.5b00354.