Metabolomic Data Analysis with MetaboAnalyst 6.0

Name: guest17082556790664 4379

May 13, 2024

1 Data Processing and Normalization

1.1 Reading and Processing the Raw Data

MetaboAnalyst accepts a variety of data types generated in metabolomic studies, including compound concentration data, binned NMR/MS spectra data, NMR/MS peak list data, as well as MS spectra (NetCDF, mzXML, mzDATA). Users need to specify the data types when uploading their data in order for MetaboAnalyst to select the correct algorithm to process them. Table 1 summarizes the result of the data processing steps.

1.1.1 Reading Binned Spectral Data

The binned spectra data should be uploaded in comma seperated values (.csv) format. Samples can be in rows or columns, with class labels immediately following the sample IDs.

Samples are in rows and features in columns The uploaded file is in comma separated values (.csv) format. The uploaded data file contains 50 (samples) by 200 (spectra bins) data matrix.

1.1.2 Data Integrity Check

Before data analysis, a data integrity check is performed to make sure that all the necessary information has been collected. The class labels must be present and contain only two classes. If samples are paired, the class label must be from -n/2 to -1 for one group, and 1 to n/2 for the other group (n is the sample number and must be an even number). Class labels with same absolute value are assumed to be pairs. Compound concentration or peak intensity values should all be non-negative numbers. By default, all missing values, zeros and negative values will be replaced by the half of the minimum positive value found within the data (see next section)

1.1.3 Missing value imputations

Too many zeroes or missing values will cause difficulties for downstream analysis. MetaboAnalyst offers several different methods for this purpose. The default method replaces all the missing and zero values with a small values (the half of the minimum positive values in the original data) assuming to be the detection limit. The assumption of this approach is that most missing values are caused by low abundance metabolites (i.e.below the detection limit). In addition, since zero values may cause problem for data normalization (i.e. log), they are also replaced with this small value. User can also specify other methods, such as replace by mean/median, or use K-Nearest Neighbours (KNN), Probabilistic PCA (PPCA), Bayesian PCA (BPCA) method, Singular Value Decomposition (SVD) method to impute the missing values [1]. Please choose the one that is the most appropriate for your data.

[1] Stacklies W, Redestig H, Scholz M, Walther D, Selbig J. *pcaMethods: a bioconductor package, providing PCA methods for incomplete data.*, Bioinformatics 2007 23(9):1164-1167

Zero or missing values were replaced by 1/5 of the min positive value for each variable. 0 variables were removed for threshold 50 percent. Missing variables were replaced by LoDs (1/5 of the min positive value for each variable)

1.1.4 Data Filtering

The purpose of the data filtering is to identify and remove variables that are unlikely to be of use when modeling the data. No phenotype information are used in the filtering process, so the result can be used with any downstream analysis. This step can usually improves the results. Data filter is strongly recommended for datasets with large number of variables ($>$ 250) datasets contain much noise (i.e.chemometrics data). Filtering can usually improve your results[2].

For data with number of variables $<$ 250, this step will reduce 5% of variables; For variable number between 250 and 500, 10% of variables will be removed; For variable number bwteen 500 and 1000, 25% of variables will be removed; And 40% of variabled will be removed for data with over 1000 variables. The None option is only for less than 5000 features. Over that, if you choose None, the IQR filter will still be applied. In addition, the maximum allowed number of variables is ***10000***

No data filtering was performed.

[2] Hackstadt AJ, Hess AM.*Filtering for increased power for microarray data analysis*, BMC Bioinformatics. 2009; 10: 11.

Table 1: Summary of data processing results

	Features (positive)	Missing/Zero	Features (processed)
C002	194	6	200
C004	189	11	200
C005	191	9	200
C006	195	5	200
C007	200	0	200
C009	186	14	200
C010	196	4	200
C011	177	23	200
C012	189	11	200
C015	188	12	200
C016	188	12	200
C017	198	2	200
C019	181	19	200
C020	184	16	200
C021	187	13	200
C022	191	9	200
C024	190	10	200
C026	195	5	200
C028	196	4	200
C029	192	8	200
C030	182	18	200
C031	179	21	200
C032	191	9	200
C033	189	11	200
C034	199	1	200
P002	195	5	200
P012	187	13	200
P014	200	0	200
P027	200	0	200
P034	198	2	200
P037	187	13	200
P038	195	5	200
P041	178	22	200
P042	198	2	200
P049	189	11	200
P056	190	10	200
P058	179	21	200
P060	190	10	200
P064	200	0	200
P065	198	2	200
P070	190	10	200
P080	196	4	200
P085	200	0	200
P086	193	7	200
P089	199	1	200
P092	191	9	200
P099	190	10	200
P113	152	48	200
P013b	191	9	200
P100b	199	1	200

1.2 Data Normalization

The data is stored as a table with one sample per row and one variable (bin/peak/metabolite) per column. The normalization procedures implemented below are grouped into four categories. Sample specific normalization allows users to manually adjust concentrations based on biological inputs (i.e. volume, mass); row-wise normalization allows general-purpose adjustment for differences among samples; data transformation and scaling are two different approaches to make features more comparable. You can use one or combine both to achieve better results.

The normalization consists of the following options:

1. Row-wise procedures:
   - Sample specific normalization (i.e. normalize by dry weight, volume)
   - Normalization by the sum
   - Normalization by the sample median
   - Normalization by a reference sample (probabilistic quotient normalization)[3]
   - Normalization by a pooled or average sample from a particular group
   - Normalization by a reference feature (i.e. creatinine, internal control)
   - Quantile normalization
2. Data transformation :
   - Log transformation (base 10)
   - Square root transformation
   - Cube root transformation
3. Data scaling:
   - Mean centering (mean-centered only)
   - Auto scaling (mean-centered and divided by standard deviation of each variable)
   - Pareto scaling (mean-centered and divided by the square root of standard deviation of each variable)
   - Range scaling (mean-centered and divided by the value range of each variable)

Figure 1 shows the effects before and after normalization.

Row-wise normalization: Normalization to constant sum; Data transformation: Cubic Root Transformation; Data scaling: Mean Centering.

[3] Dieterle F, Ross A, Schlotterbeck G, Senn H. *Probabilistic quotient normalization as robust method to account for dilution of complex biological mixtures. Application in 1H NMR metabonomics*, 2006, Anal Chem 78 (13);4281 - 4290

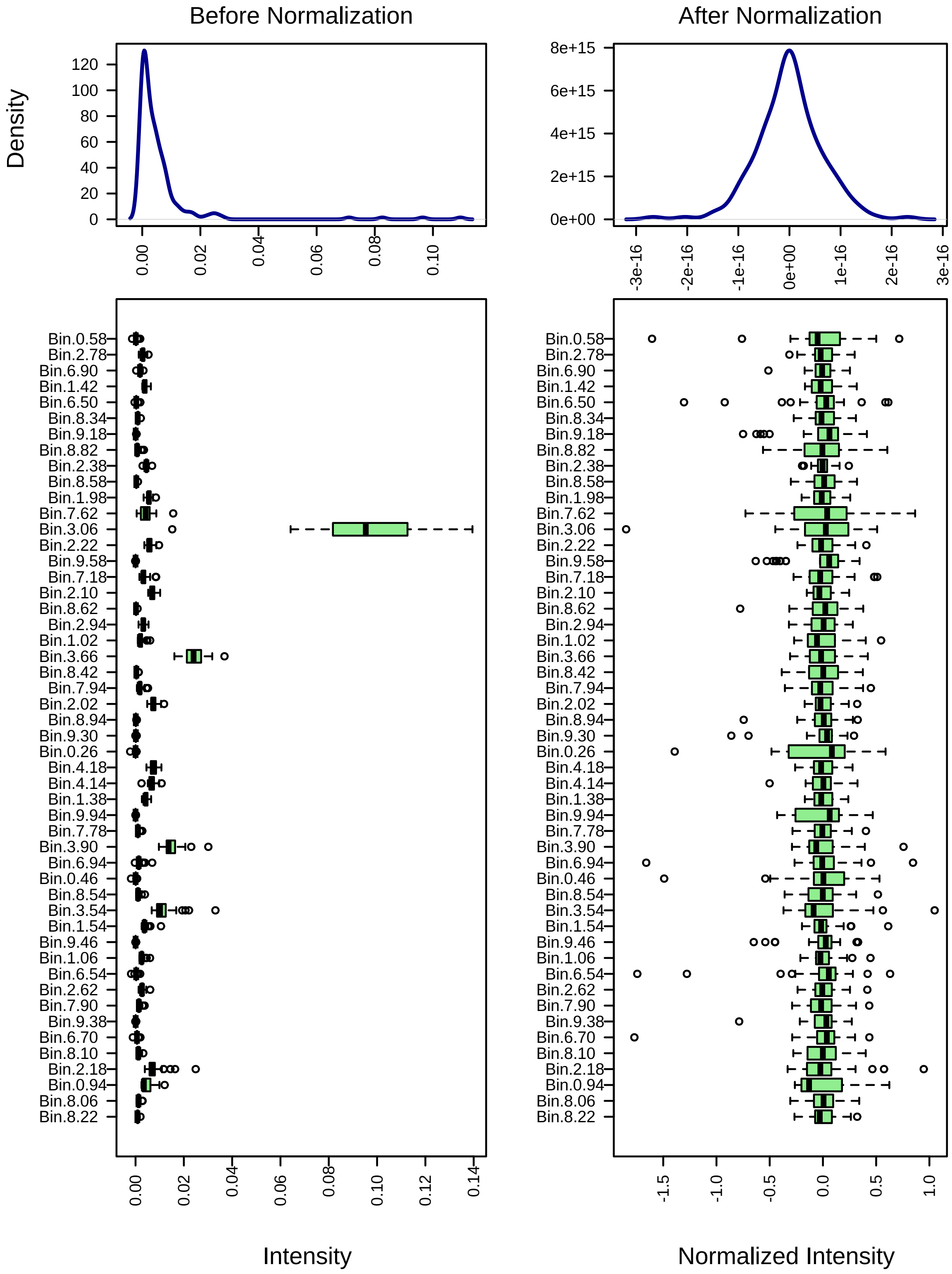

Figure 1: Box plots and kernel density plots before and after normalization. The boxplots show at most 50 features due to space limit. The density plots are based on all samples.

2 Statistical and Machine Learning Data Analysis

MetaboAnalyst offers a variety of methods commonly used in metabolomic data analyses. They include:

1. Univariate analysis methods:
   - Fold Change Analysis
   - T-tests
   - Volcano Plot
   - One-way ANOVA and post-hoc analysis
   - Correlation analysis
2. Multivariate analysis methods:
   - Principal Component Analysis (PCA)
   - Partial Least Squares - Discriminant Analysis (PLS-DA)
3. Robust Feature Selection Methods in microarray studies
   - Significance Analysis of Microarray (SAM)
   - Empirical Bayesian Analysis of Microarray (EBAM)
4. Clustering Analysis
   - Hierarchical Clustering
     - Dendrogram
     - Heatmap
   - Partitional Clustering
     - K-means Clustering
     - Self-Organizing Map (SOM)
5. Supervised Classification and Feature Selection methods
   - Random Forest
   - Support Vector Machine (SVM)

`Please note:  some advanced methods are available only for two-group sample analyais.`

2.1 Univariate Analysis

Univariate analysis methods are the most common methods used for exploratory data analysis. For two-group data, MetaboAnalyst provides Fold Change (FC) analysis, t-tests, and volcano plot which is a combination of the first two methods. All three these methods support both unpaired and paired analyses. For multi-group analysis, MetaboAnalyst provides two types of analysis - one-way analysis of variance (ANOVA) with associated post-hoc analyses, and correlation analysis to identify signficant compounds that follow a given pattern. The univariate analyses provide a preliminary overview about features that are potentially significant in discriminating the conditions under study.

For paired fold change analysis, the algorithm first counts the total number of pairs with fold changes that are consistently above/below the specified FC threshold for each variable. A variable will be reported as significant if this number is above a given count threshold (default > 75% of pairs/variable)

Figure 2 shows the important features identified by t-tests. Table 2 shows the details of these features;

Please note, the purpose of fold change is to compare absolute value changes between two group means. Therefore, the data before column normalization will be used instead. Also note, the result is plotted in log2 scale, so that same fold change (up/down regulated) will have the same distance to the zero baseline.

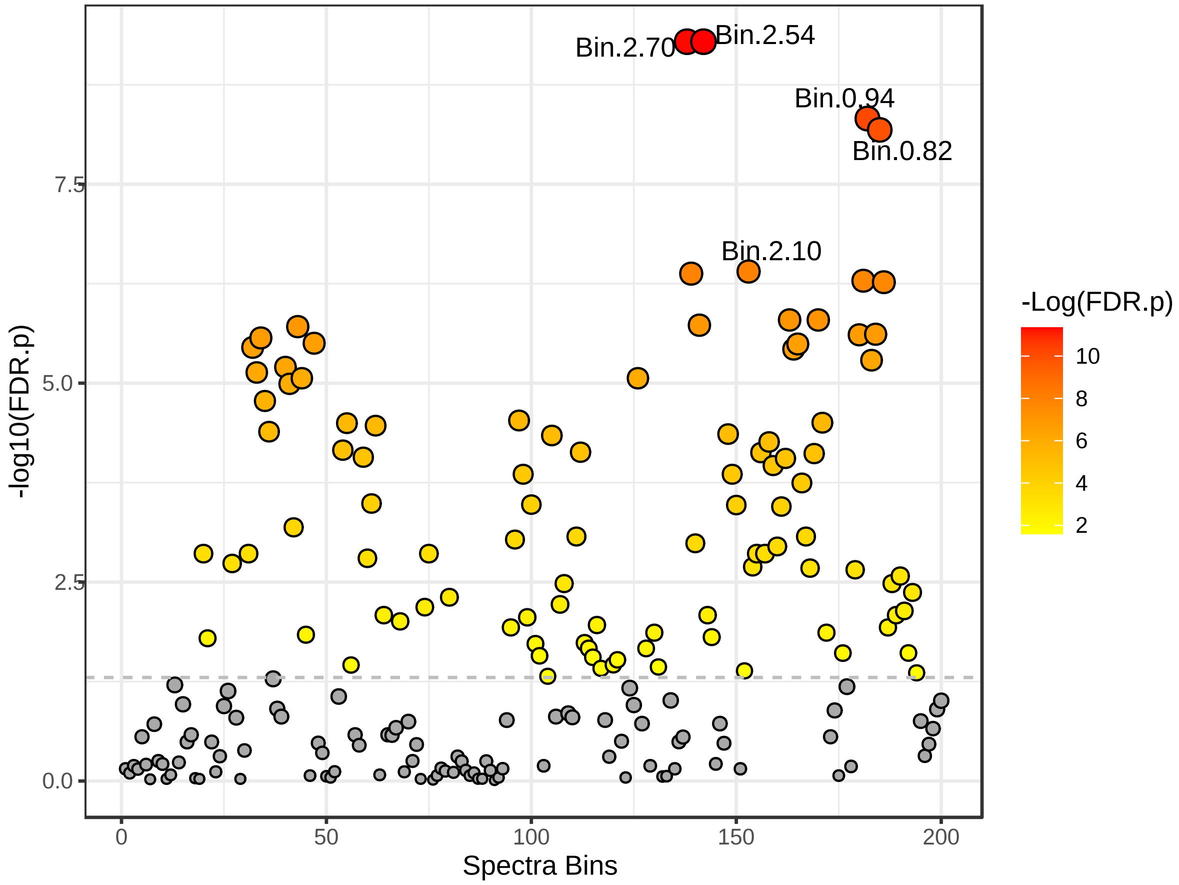

Figure 2: Important features selected by t-tests with threshold 0.05. The red circles represent features above the threshold. Note the p values are transformed by -log10 so that the more significant features (with smaller p values) will be plotted higher on the graph.

Table 2: Top 50 features identified by t-tests

	Spectra Bins	t.stat	p.value	-log10(p)	FDR
1	Bin.2.54	9.129	4.5843e-12	11.339	5.118e-10
2	Bin.2.70	9.0964	5.118e-12	11.291	5.118e-10
3	Bin.0.94	-8.3267	7.0984e-11	10.149	4.7323e-09
4	Bin.0.82	-8.1487	1.3141e-10	9.8814	6.5704e-09
5	Bin.2.10	-6.9162	9.8861e-09	8.005	3.9544e-07
6	Bin.2.66	6.8471	1.2615e-08	7.8991	4.2052e-07
7	Bin.0.98	-6.7454	1.8071e-08	7.743	5.1632e-07
8	Bin.0.78	-6.6956	2.1551e-08	7.6665	5.3877e-07
9	Bin.1.42	-6.3353	7.6934e-08	7.1139	1.6069e-06
10	Bin.1.70	-6.323	8.0345e-08	7.095	1.6069e-06
11	Bin.2.58	6.2531	1.0282e-07	6.9879	1.8694e-06
12	Bin.8.30	-6.2166	1.1694e-07	6.932	1.949e-06
13	Bin.0.86	-6.1314	1.5787e-07	6.8017	2.4288e-06
14	Bin.1.02	-6.1057	1.7286e-07	6.7623	2.4694e-06
15	Bin.8.66	-6.0604	2.0272e-07	6.6931	2.703e-06
16	Bin.8.14	-5.9985	2.5205e-07	6.5985	3.1506e-06
17	Bin.1.62	-5.975	2.7373e-07	6.5627	3.2203e-06
18	Bin.8.74	-5.9306	3.1993e-07	6.4949	3.5548e-06
19	Bin.1.66	-5.9005	3.5559e-07	6.4491	3.743e-06
20	Bin.0.90	-5.7949	5.1501e-07	6.2882	5.1501e-06
21	Bin.8.42	-5.7239	6.6021e-07	6.1803	6.2877e-06
22	Bin.8.70	-5.6664	8.0722e-07	6.093	7.3384e-06
23	Bin.8.26	-5.595	1.0357e-06	5.9848	8.6745e-06
24	Bin.3.14	5.5935	1.0409e-06	5.9826	8.6745e-06
25	Bin.8.38	-5.5351	1.2755e-06	5.8943	1.0204e-05
26	Bin.8.62	-5.3809	2.1783e-06	5.6619	1.6756e-05
27	Bin.4.26	-5.2063	3.979e-06	5.4002	2.9474e-05
28	Bin.1.38	-5.1772	4.3971e-06	5.3568	3.1408e-05
29	Bin.7.82	5.1631	4.6162e-06	5.3357	3.1836e-05
30	Bin.7.54	5.1321	5.1329e-06	5.2896	3.422e-05
31	Bin.8.58	-5.0716	6.3142e-06	5.1997	4.0737e-05
32	Bin.2.30	-5.042	6.9851e-06	5.1558	4.3657e-05
33	Bin.3.98	5.021	7.5055e-06	5.1246	4.5488e-05
34	Bin.1.90	-4.9574	9.3192e-06	5.0306	5.4819e-05
35	Bin.7.86	4.8788	1.2165e-05	4.9149	6.9514e-05
36	Bin.3.70	-4.853	1.3278e-05	4.8769	7.3764e-05
37	Bin.1.98	-4.8413	1.3809e-05	4.8598	7.4642e-05
38	Bin.1.46	-4.8251	1.4585e-05	4.8361	7.6761e-05
39	Bin.7.66	4.7862	1.663e-05	4.7791	8.5284e-05
40	Bin.1.74	-4.7684	1.7658e-05	4.7531	8.8289e-05
41	Bin.1.86	-4.6994	2.2258e-05	4.6525	0.00010858
42	Bin.2.26	-4.6113	2.987e-05	4.5248	0.00013961
43	Bin.4.22	-4.6098	3.0017e-05	4.5226	0.00013961
44	Bin.1.58	-4.5269	3.9537e-05	4.403	0.00017971
45	Bin.7.58	4.3392	7.3261e-05	4.1351	0.0003256
46	Bin.4.14	-4.3228	7.7289e-05	4.1119	0.00033604
47	Bin.2.22	-4.3122	7.9991e-05	4.097	0.00034039
48	Bin.1.78	-4.2925	8.5309e-05	4.069	0.00035545
49	Bin.8.34	-4.0996	0.00015886	3.799	0.00064841
50	Bin.3.74	-4.0085	0.00021221	3.6732	0.00084618

2.2 Correlation Analysis

Correlation analysis can be used to visualize the overall correlations between different features It can also be used to identify which features are correlated with a feature of interest. Correlation analysis can also be used to identify if certain features show particular patterns under different conditions. Users first need to define a pattern in the form of a series of hyphenated numbers. For example, in a time-series study with four time points, a pattern of of `1-2-3-4` is used to search compounds with increasing the concentration as time changes; while a pattern of `3-2-1-3` can be used to search compounds that decrease at first, then bounce back to the original level.

Figure 3 shows the overall correlation heatmap.

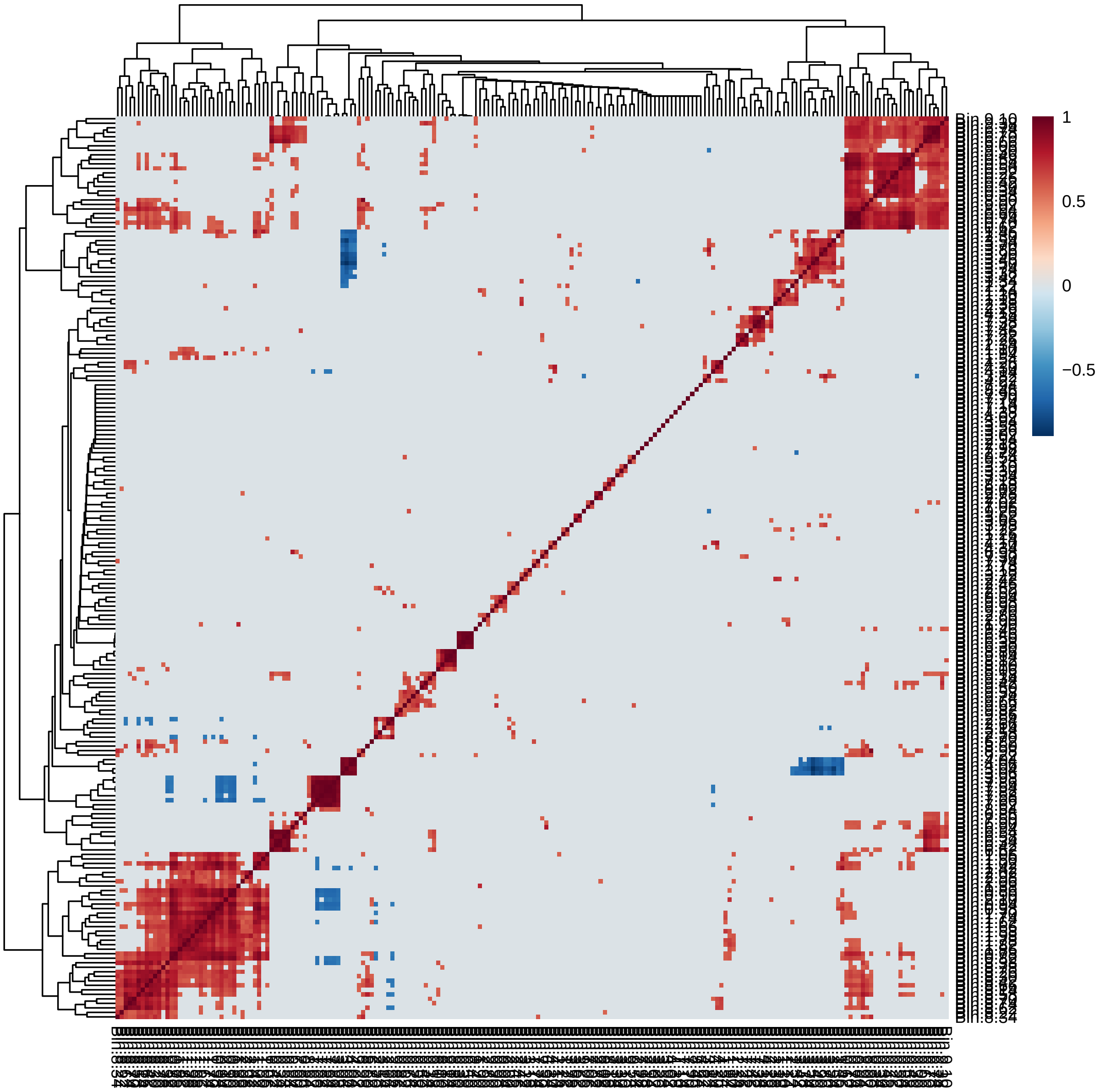

Figure 3: Correlation Heatmaps

2.3 Principal Component Analysis (PCA)

PCA is an unsupervised method aiming to find the directions that best explain the variance in a data set (X) without referring to class labels (Y). The data are summarized into much fewer variables called *scores* which are weighted average of the original variables. The weighting profiles are called *loadings*. The PCA analysis is performed using the `prcomp` package. The calculation is based on singular value decomposition.

The Rscript `chemometrics.R` is required. Figure 4 is pairwise score plots providing an overview of the various seperation patterns among the most significant PCs; Figure 5 is the scree plot showing the variances explained by the selected PCs; Figure 6 shows the 2-D scores plot between selected PCs; Figure 7 shows the biplot between the selected PCs. Interactive 3-D scores plots are not included here and can be directly downloaded from website.

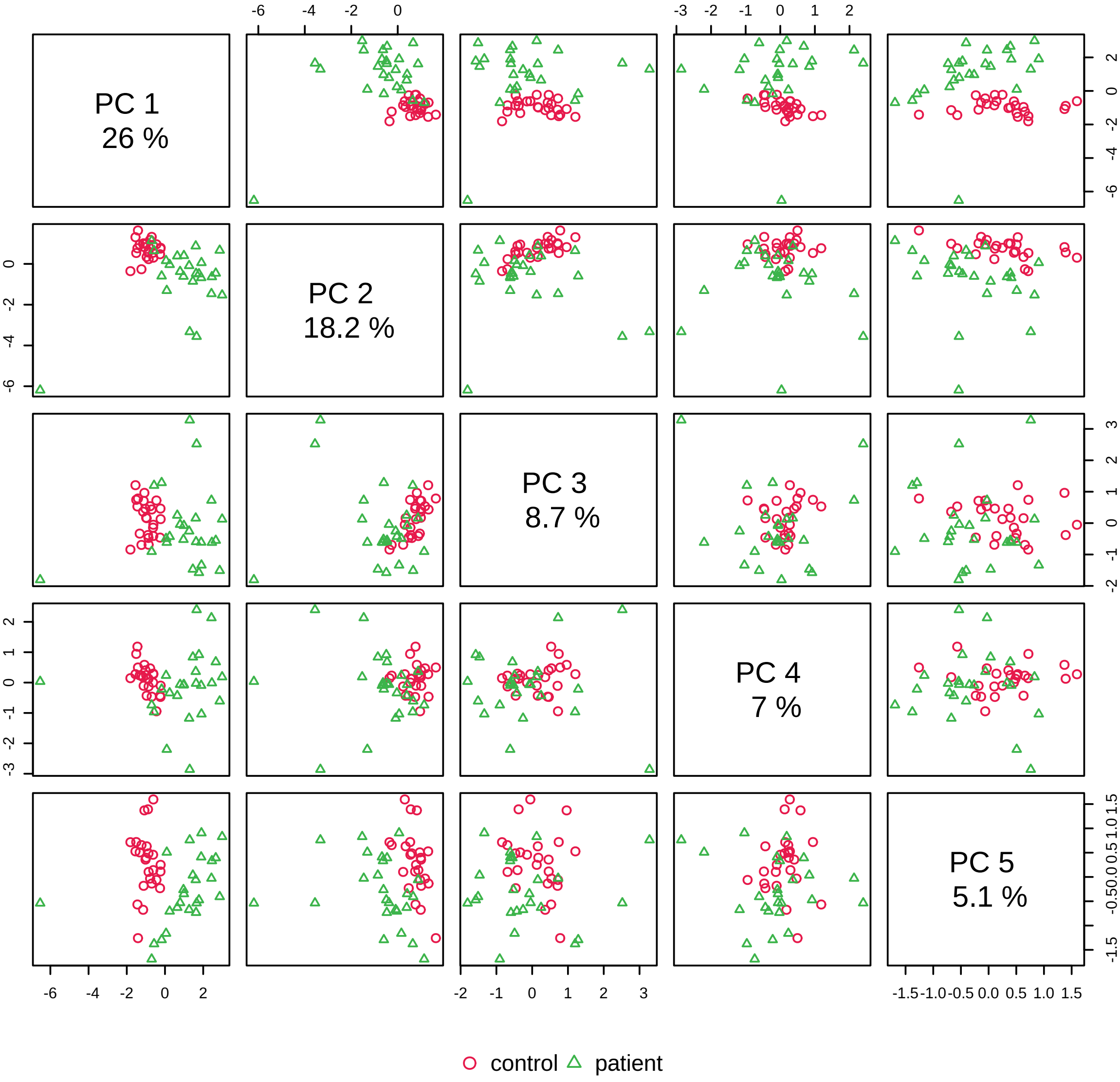

Figure 4: Pairwise score plots between the selected PCs. The explained variance of each PC is shown in the corresponding diagonal cell.

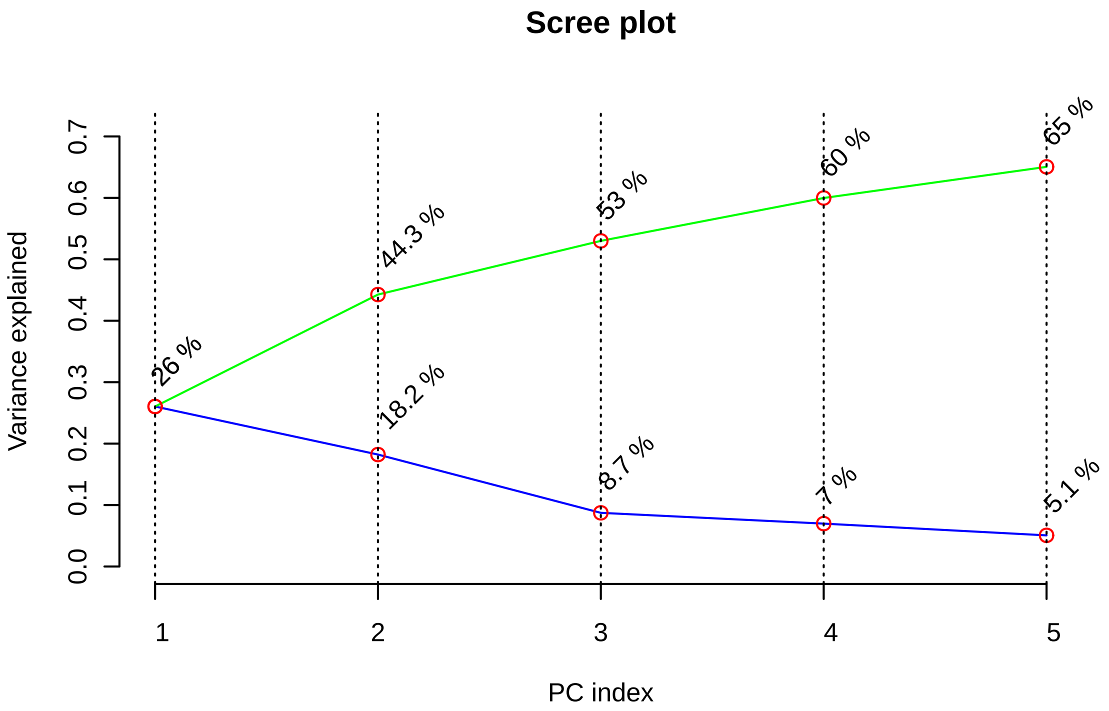

Figure 5: Scree plot shows the variance explained by PCs. The green line on top shows the accumulated variance explained; the blue line underneath shows the variance explained by individual PC.

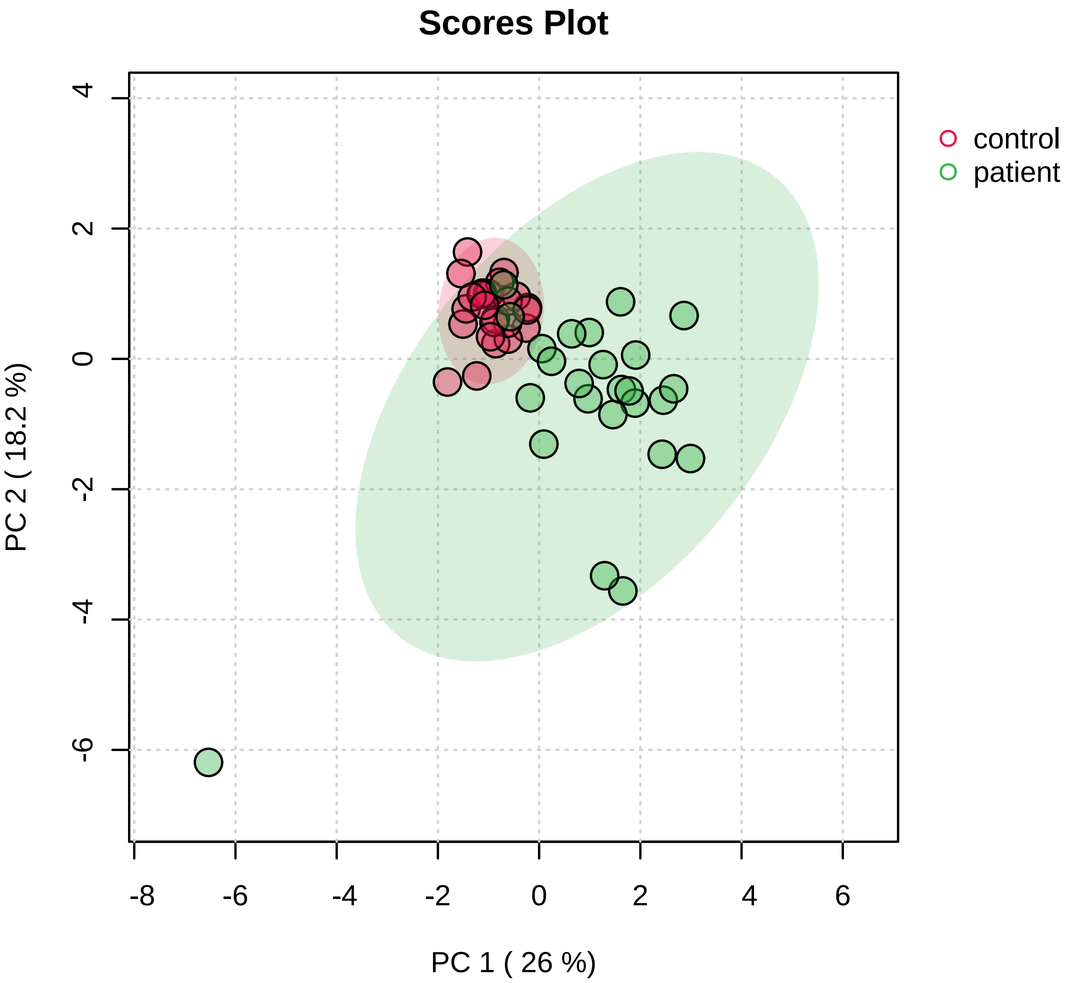

Figure 6: Scores plot between the selected PCs. The explained variances are shown in brackets.

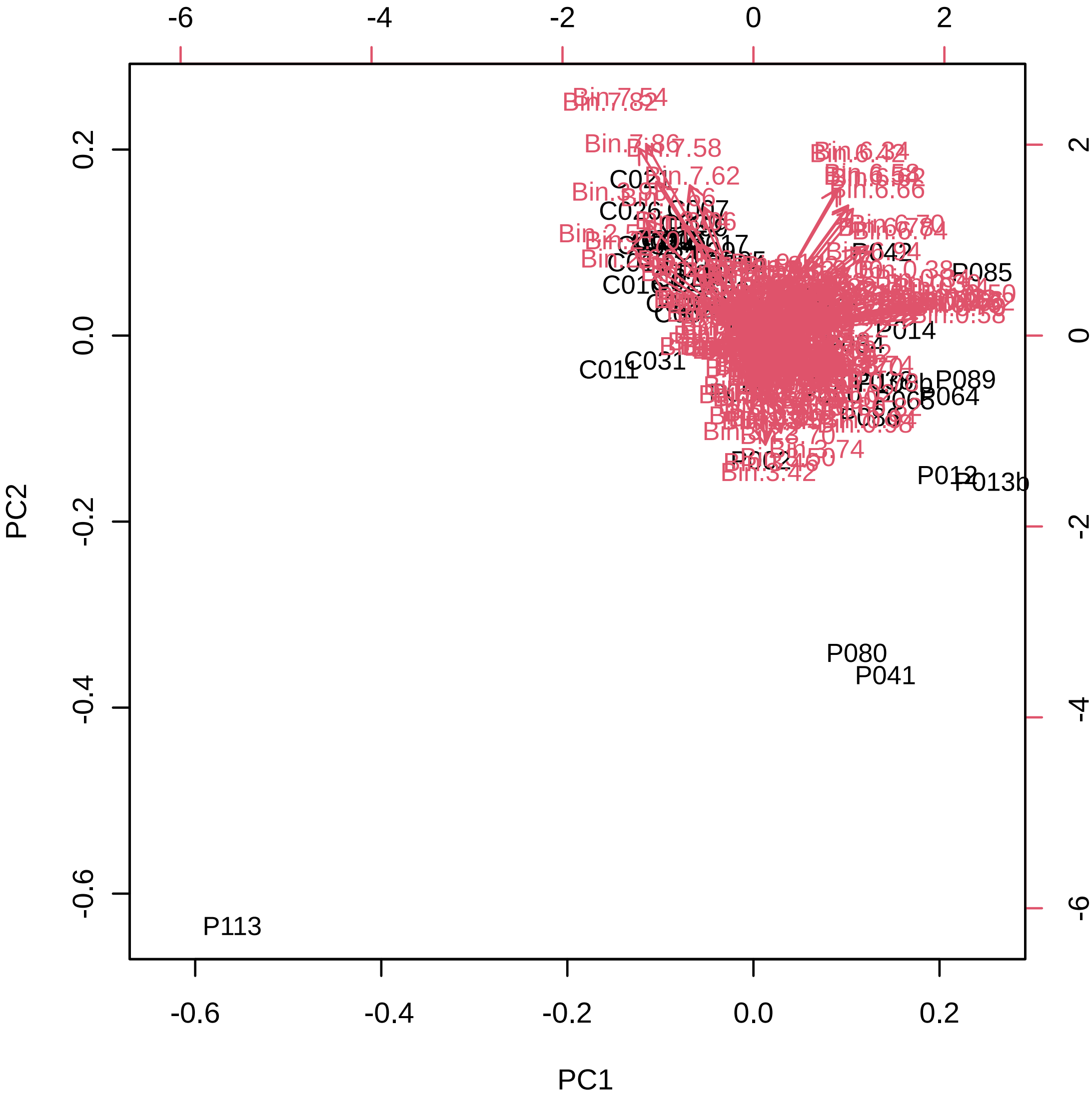

Figure 7: PCA biplot between the selected PCs. Note, you may want to test different centering and scaling normalization methods for the biplot to be displayed properly.

2.4 Hierarchical Clustering

In (agglomerative) hierarchical cluster analysis, each sample begins as a separate cluster and the algorithm proceeds to combine them until all samples belong to one cluster. Two parameters need to be considered when performing hierarchical clustering. The first one is similarity measure - Euclidean distance, Pearson's correlation, Spearman's rank correlation. The other parameter is clustering algorithms, including average linkage (clustering uses the centroids of the observations), complete linkage (clustering uses the farthest pair of observations between the two groups), single linkage (clustering uses the closest pair of observations) and Ward's linkage (clustering to minimize the sum of squares of any two clusters). Heatmap is often presented as a visual aid in addition to the dendrogram.

Hierachical clustering is performed with the `hclust` function in package `stat`. Figure 8 shows the clustering result in the form of a dendrogram. Figure 9 shows the clustering result in the form of a heatmap.

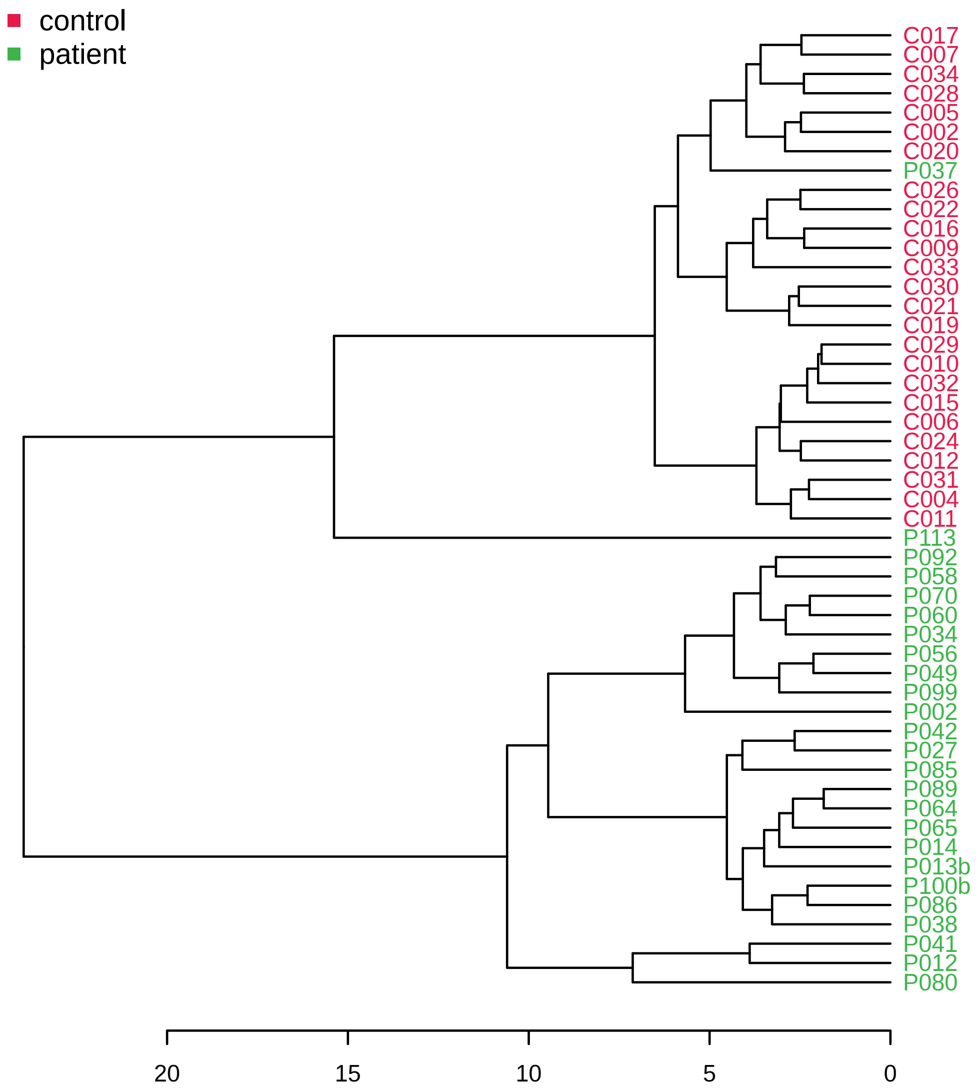

Figure 8: Clustering result shown as dendrogram (distance measure using `euclidean`, and clustering algorithm using `ward.D`).

2.5 Random Forest (RF)

Random Forest is a supervised learning algorithm suitable for high dimensional data analysis. It uses an ensemble of classification trees, each of which is grown by random feature selection from a bootstrap sample at each branch. Class prediction is based on the majority vote of the ensemble. RF also provides other useful information such as OOB (out-of-bag) error, variable importance measure, and outlier measures. During tree construction, about one-third of the instances are left out of the bootstrap sample. This OOB data is then used as test sample to obtain an unbiased estimate of the classification error (OOB error). Variable importance is evaluated by measuring the increase of the OOB error when it is permuted. The outlier measures are based on the proximities during tree construction.

RF analysis is performed using the `randomForest` package[4]. Table 3 shows the confusion matrix of random forest. Figure 10 shows the cumulative error rates of random forest analysis for given parameters. Figure 11 shows the important features ranked by random forest. Figure 12 shows the outlier measures of all samples for the given parameters. The OOB error is 0.08

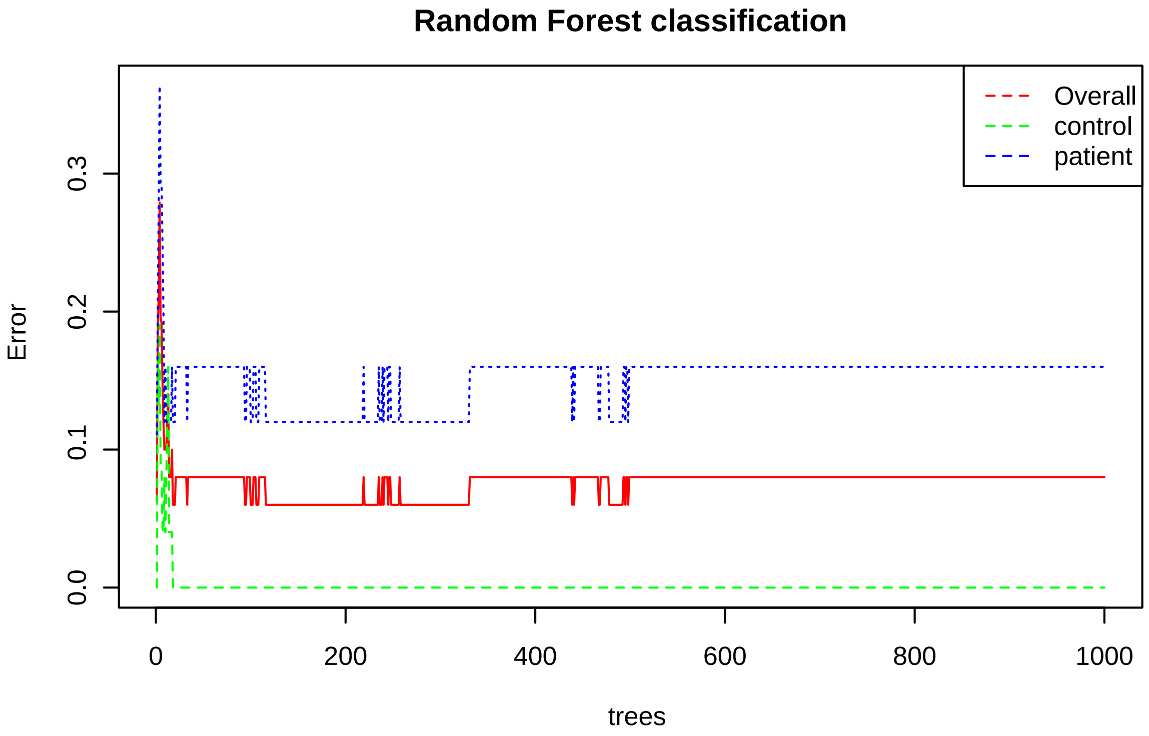

Figure 9: Cumulative error rates by Random Forest classification. The overall error rate is shown as the black line; the red and green lines represent the error rates for each class.

	control	patient	class.error
control	25.00	0.00	0.00
patient	4.00	21.00	0.16

Table 3: Random Forest Classification Performance

[4] Andy Liaw and Matthew Wiener. *Classification and Regression by randomForest*, 2002, R News

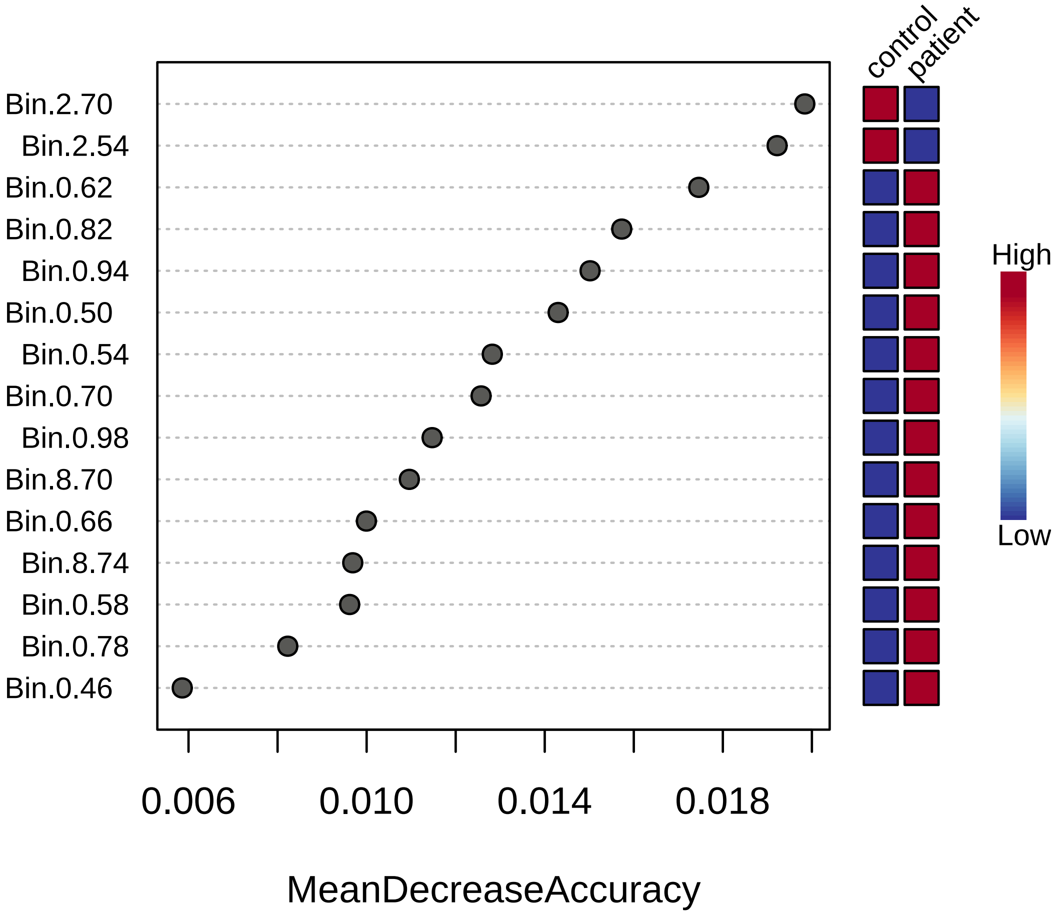

Figure 10: Significant features identified by Random Forest. The features are ranked by the mean decrease in classification accuracy when they are permuted.

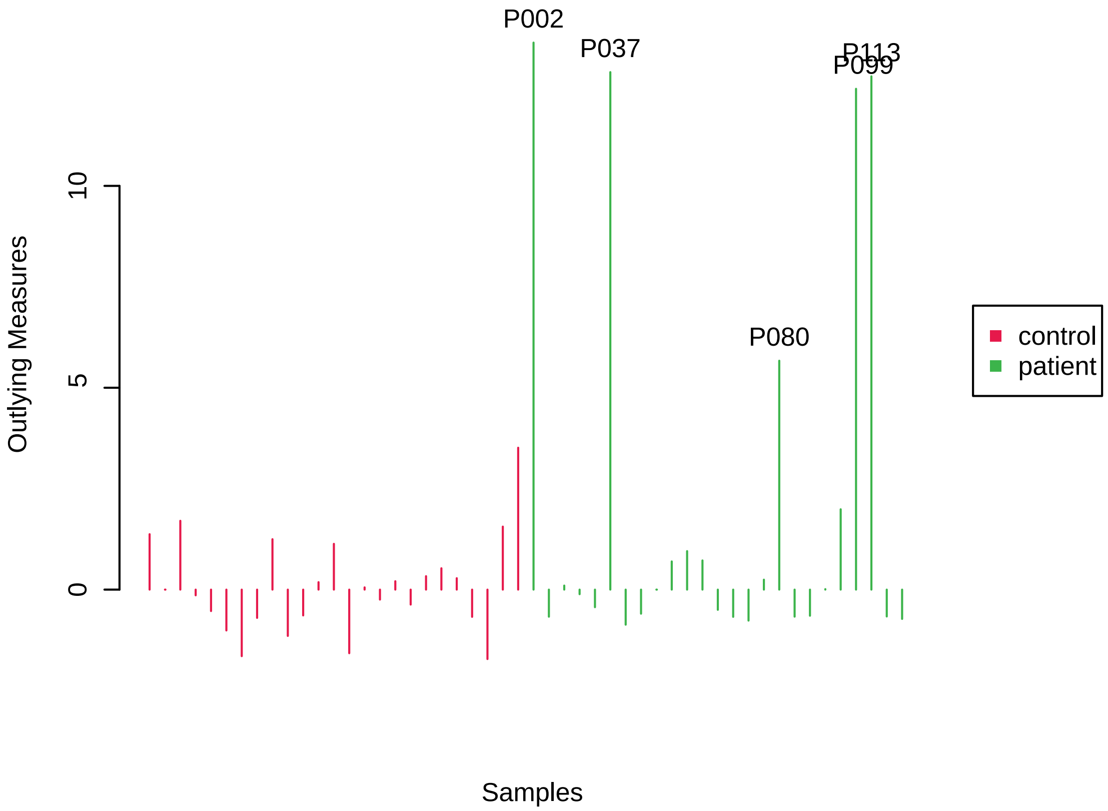

Figure 11: Potential outliers identified by Random Forest. Only the top five are labeled.

3 Appendix: R Command History

```
 [1] "mSet<-InitDataObjects(\"specbin\", \"stat\", FALSE)"
 [2] "mSet<-Read.TextData(mSet, \"Replacing_with_your_file_path\", \"rowu\", \"disc\");"
 [3] "mSet<-SanityCheckData(mSet)"
 [4] "mSet<-ReplaceMin(mSet);"
 [5] "mSet<-SanityCheckData(mSet)"
 [6] "mSet<-SanityCheckData(mSet)"
 [7] "mSet<-RemoveMissingPercent(mSet, percent=0.5)"
 [8] "mSet<-ImputeMissingVar(mSet, method=\"min\")"
 [9] "mSet<-SanityCheckData(mSet)"
[10] "mSet<-SanityCheckData(mSet)"
[11] "mSet<-FilterVariable(mSet, \"F\", 25, \"none\", -1, \"mean\", 0)"
[12] "mSet<-PreparePrenormData(mSet)"
[13] "mSet<-Normalization(mSet, \"NULL\", \"NULL\", \"NULL\", ratio=FALSE, ratioNum=20)"
[14] "mSet<-PlotNormSummary(mSet, \"norm_0_\", \"png\", 72, width=NA)"
[15] "mSet<-PlotSampleNormSummary(mSet, \"snorm_0_\", \"png\", 72, width=NA)"
[16] "mSet<-Normalization(mSet, \"SumNorm\", \"LogNorm\", \"MeanCenter\", ratio=FALSE, ratioNum=20)"
[17] "mSet<-PlotNormSummary(mSet, \"norm_1_\", \"png\", 72, width=NA)"
[18] "mSet<-PlotSampleNormSummary(mSet, \"snorm_1_\", \"png\", 72, width=NA)"
[19] "mSet<-Normalization(mSet, \"SumNorm\", \"CrNorm\", \"MeanCenter\", ratio=FALSE, ratioNum=20)"
[20] "mSet<-PlotNormSummary(mSet, \"norm_2_\", \"png\", 72, width=NA)"
[21] "mSet<-PlotSampleNormSummary(mSet, \"snorm_2_\", \"png\", 72, width=NA)"
[22] "mSet<-Normalization(mSet, \"SumNorm\", \"SrNorm\", \"MeanCenter\", ratio=FALSE, ratioNum=20)"
[23] "mSet<-PlotNormSummary(mSet, \"norm_3_\", \"png\", 72, width=NA)"
[24] "mSet<-PlotSampleNormSummary(mSet, \"snorm_3_\", \"png\", 72, width=NA)"
[25] "mSet<-Normalization(mSet, \"SumNorm\", \"CrNorm\", \"MeanCenter\", ratio=FALSE, ratioNum=20)"
[26] "mSet<-PlotNormSummary(mSet, \"norm_4_\", \"png\", 72, width=NA)"
[27] "mSet<-PlotSampleNormSummary(mSet, \"snorm_4_\", \"png\", 72, width=NA)"
[28] "mSet<-PCA.Anal(mSet)"
[29] "mSet<-PlotPCAPairSummary(mSet, \"pca_pair_0_\", \"png\", 72, width=NA, 5)"
[30] "mSet<-PlotPCAScree(mSet, \"pca_scree_0_\", \"png\", 72, width=NA, 5)"
[31] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_0_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[32] "mSet<-PlotPCALoading(mSet, \"pca_loading_0_\", \"png\", 72, width=NA, 1,2);"
[33] "mSet<-PlotPCABiplot(mSet, \"pca_biplot_0_\", \"png\", 72, width=NA, 1,2)"
[34] "mSet<-PlotPCA3DLoading(mSet, \"pca_loading3d_0_\", \"json\", 1,2,3)"
[35] "mSet<-RF.Anal(mSet, 500,7,1)"
[36] "mSet<-PlotRF.Classify(mSet, \"rf_cls_0_\", \"png\", 72, width=NA)"
[37] "mSet<-PlotRF.VIP(mSet, \"rf_imp_0_\", \"png\", 72, width=NA)"
[38] "mSet<-PlotRF.Outlier(mSet, \"rf_outlier_0_\", \"png\", 72, width=NA)"
[39] "mSet<-RF.Anal(mSet, 500,10,1)"
[40] "mSet<-PlotRF.Classify(mSet, \"rf_cls_1_\", \"png\", 72, width=NA)"
[41] "mSet<-PlotRF.VIP(mSet, \"rf_imp_1_\", \"png\", 72, width=NA)"
[42] "mSet<-PlotRF.Outlier(mSet, \"rf_outlier_1_\", \"png\", 72, width=NA)"
[43] "mSet<-RF.Anal(mSet, 1000,10,1)"
[44] "mSet<-PlotRF.Classify(mSet, \"rf_cls_2_\", \"png\", 72, width=NA)"
[45] "mSet<-PlotRF.VIP(mSet, \"rf_imp_2_\", \"png\", 72, width=NA)"
[46] "mSet<-PlotRF.Outlier(mSet, \"rf_outlier_2_\", \"png\", 72, width=NA)"
[47] "mSet<-PlotCorrHeatMap(mSet, \"corr_1_\", \"png\", 72, width=NA, \"col\", \"pearson\", \"bwm\",
[48] "mSet<-PlotHCTree(mSet, \"tree_0_\", \"png\", 72, width=NA, \"euclidean\", \"ward.D\")"
[49] "mSet<-PlotHCTree(mSet, \"tree_1_\", \"png\", 72, width=NA, \"euclidean\", \"ward.D\")"
[50] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[51] "mSet<-PlotTT(mSet, \"tt_0_\", \"png\", 72, width=NA)"
[52] "mSet<-SaveTransformedData(mSet)"
[53] "mSet<-PreparePDFReport(mSet, \"guest17082556790664379\")\n"
```

The report was generated on Mon May 13 10:55:20 2024 with R version 4.3.2 (2023-10-31), OS system: Linux, version: -Ubuntu SMP Tue Mar 5 20:16:58 UTC 2024 .