Metabolomic Data Analysis with MetaboAnalyst 6.0

Name: guest3445752532594411690

April 4, 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.

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.

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

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

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 sample median; Data transformation: Square Root Transformation; Data scaling: Autoscaling.

[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

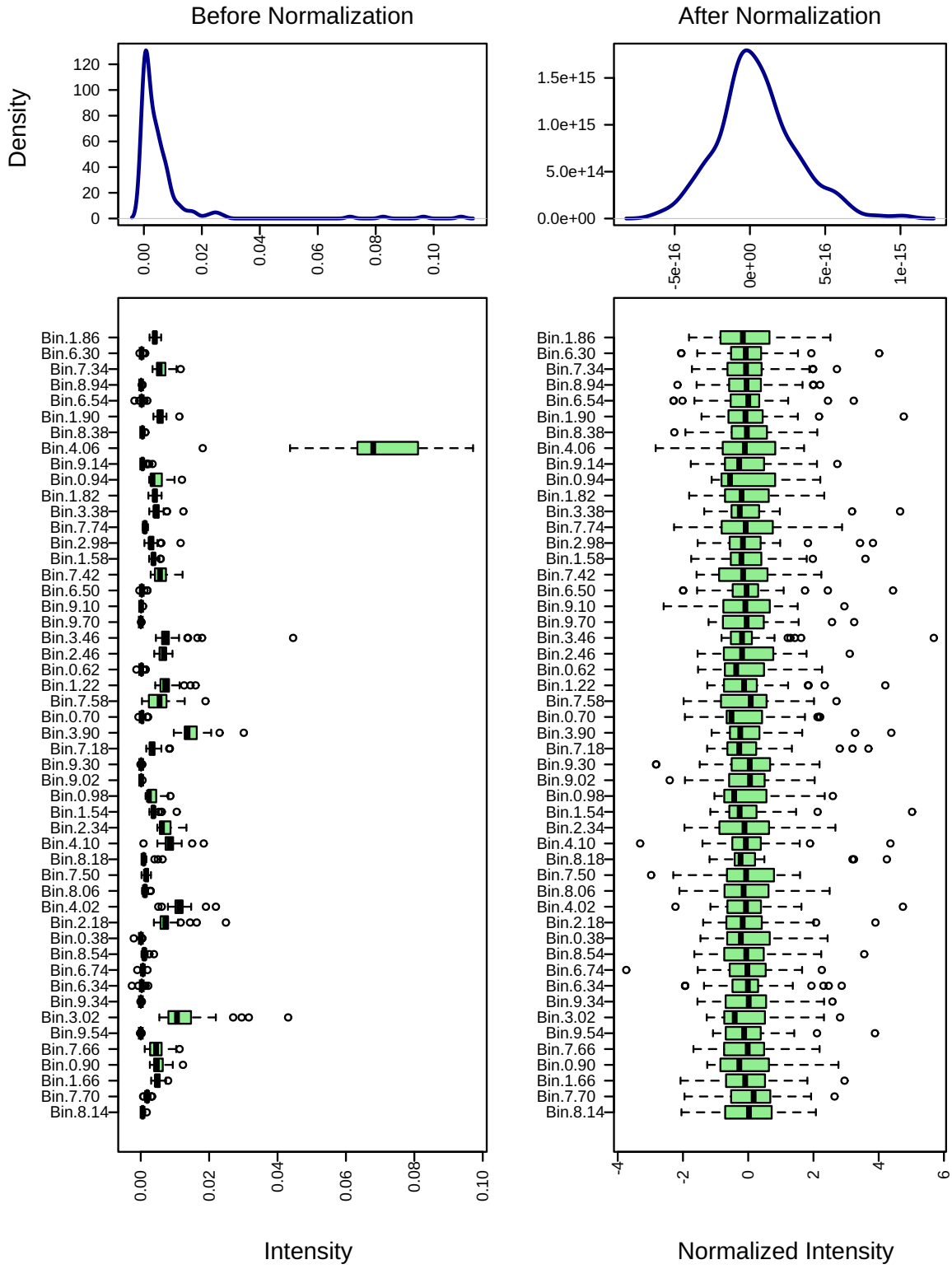

Figure 1: Box plots and kernel density plots before and after normalization. The boxplots show at most 50 features due to space limit. The density plots are based on all samples.

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 fold change analysis. Table 2 shows the details of these features; Figure 3 shows the important features identified by t-tests. Table 3 shows the details of these features; Figure 4 shows the important features identified by volcano plot. Table 4 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.

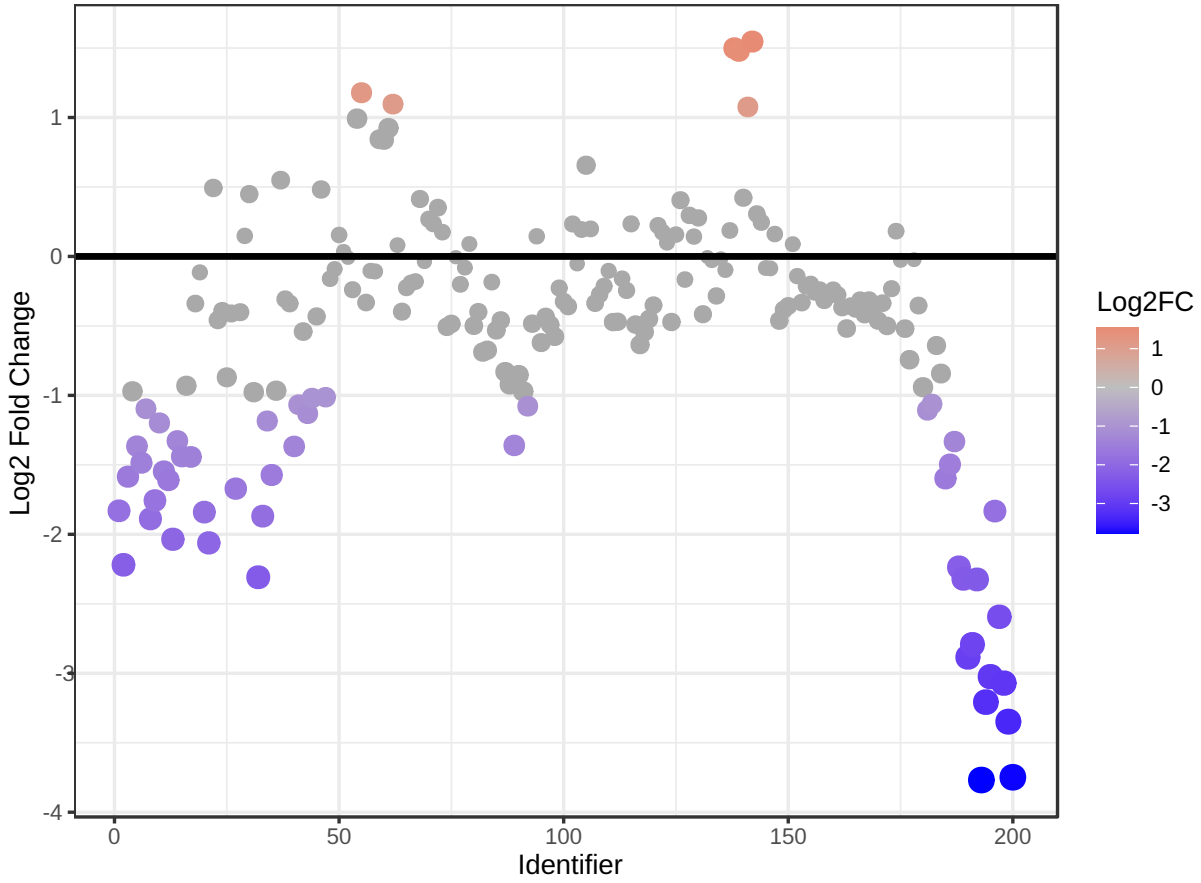

Figure 2: Important features selected by fold-change analysis with threshold 2. The red circles represent features above the threshold. Note the values are on log scale, so that both up-regulated and down-regulated features can be plotted in a symmetrical way

Table 2: Top 50 features identified by fold change analysis

	Spectra Bins	Fold Change	log2(FC)
1	Bin.0.50	0.073381	-3.7684
2	Bin.0.22	0.074396	-3.7486
3	Bin.0.26	0.098188	-3.3483
4	Bin.0.46	0.10825	-3.2076
5	Bin.0.30	0.11901	-3.0708
6	Bin.0.42	0.12291	-3.0243
7	Bin.0.62	0.13546	-2.8841
8	Bin.0.58	0.14434	-2.7925
9	Bin.0.34	0.16568	-2.5935
10	Bin.0.54	0.19961	-2.3248
11	Bin.0.66	0.20028	-2.3199
12	Bin.8.74	0.2018	-2.309
13	Bin.0.70	0.21209	-2.2373
14	Bin.9.94	0.21475	-2.2193
15	Bin.9.18	0.23955	-2.0616
16	Bin.9.50	0.2439	-2.0356
17	Bin.9.70	0.27004	-1.8887
18	Bin.8.70	0.27372	-1.8692
19	Bin.9.22	0.27928	-1.8402
20	Bin.0.38	0.28071	-1.8328
21	Bin.9.98	0.2811	-1.8308
22	Bin.9.66	0.29597	-1.7565
23	Bin.8.94	0.31398	-1.6713
24	Bin.9.54	0.32773	-1.6094
25	Bin.0.82	0.33046	-1.5975
26	Bin.9.90	0.33333	-1.585
27	Bin.8.62	0.33612	-1.573
28	Bin.9.58	0.34217	-1.5472
29	Bin.2.54	2.9209	1.5464
30	Bin.2.70	2.8251	1.4983
31	Bin.0.78	0.35423	-1.4972
32	Bin.9.78	0.35721	-1.4851
33	Bin.2.66	2.788	1.4792
34	Bin.9.34	0.36779	-1.443
35	Bin.9.42	0.3686	-1.4399
36	Bin.8.42	0.3876	-1.3674
37	Bin.9.82	0.38804	-1.3657
38	Bin.6.46	0.38956	-1.3601
39	Bin.0.74	0.39715	-1.3323
40	Bin.9.46	0.39847	-1.3274
41	Bin.9.62	0.43577	-1.1984
42	Bin.8.66	0.44002	-1.1844
43	Bin.7.82	2.263	1.1782
44	Bin.8.30	0.45651	-1.1313
45	Bin.0.98	0.46426	-1.107
46	Bin.9.74	0.46771	-1.0963
47	Bin.7.54	2.1377	1.0961
48	Bin.6.34	0.47367	-1.078
49	Bin.2.58	2.1077	1.0757
50	Bin.8.38	0.47723	-1.0672

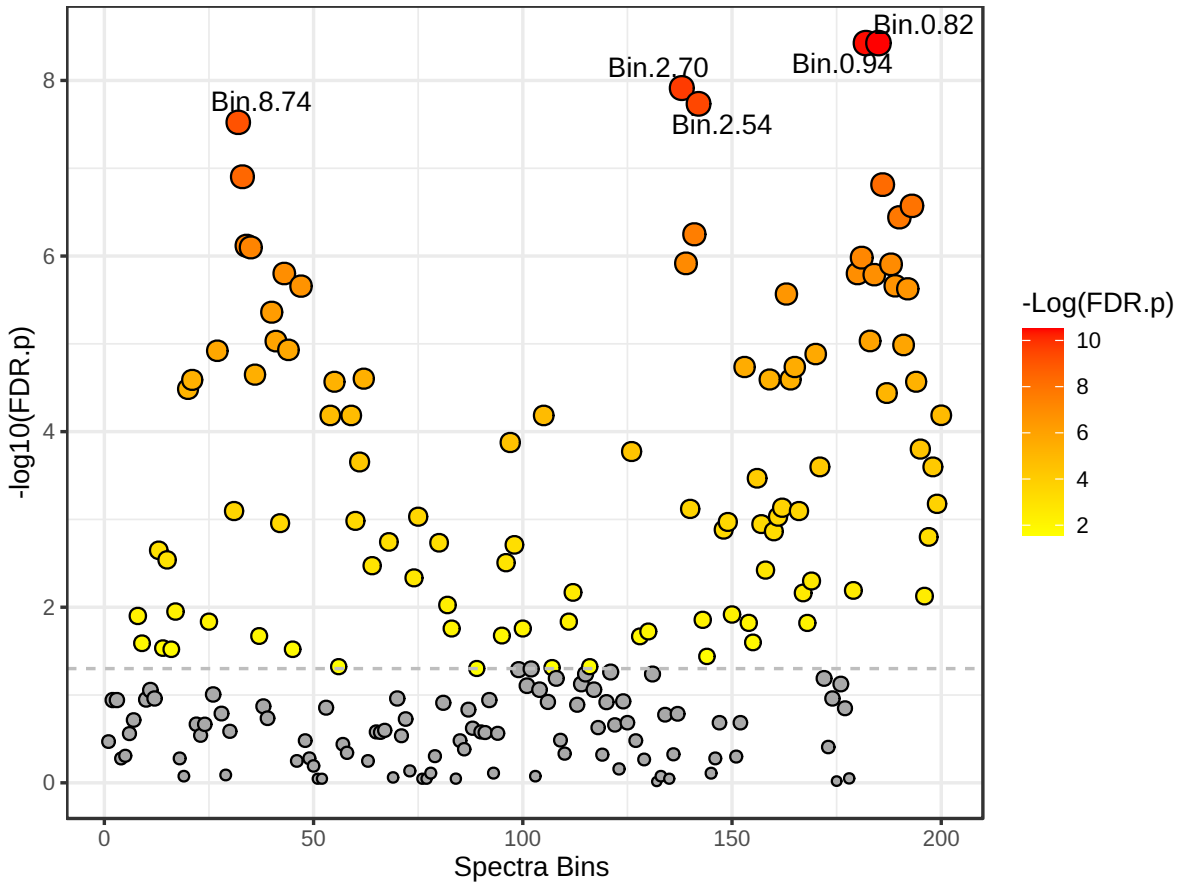

Figure 3: Important features selected by t-tests with threshold 0.05. The red circles represent features above the threshold. Note the p values are transformed by -log10 so that the more significant features (with smaller p values) will be plotted higher on the graph.

Table 3: Top 50 features identified by t-tests

	Spectra Bins	t.stat	p.value	-log10(p)	FDR
1	Bin.0.82	-8.5573	3.2089e-11	10.494	3.753e-09
2	Bin.0.94	-8.5117	3.753e-11	10.426	3.753e-09
3	Bin.2.70	8.053	1.8312e-10	9.7373	1.2208e-08
4	Bin.2.54	7.8517	3.6913e-10	9.4328	1.8456e-08
5	Bin.8.74	-7.6489	7.5003e-10	9.1249	3.0001e-08
6	Bin.8.70	-7.1909	3.7498e-09	8.426	1.2499e-07
7	Bin.0.78	-7.0887	5.3764e-09	8.2695	1.5361e-07
8	Bin.0.50	-6.8932	1.072e-08	7.9698	2.6801e-07
9	Bin.0.62	-6.7748	1.6289e-08	7.7881	3.6199e-07
10	Bin.2.58	6.6189	2.8257e-08	7.5489	5.6515e-07
11	Bin.8.66	-6.5071	4.1947e-08	7.3773	7.6267e-07
12	Bin.8.62	-6.4696	4.7886e-08	7.3198	7.9809e-07
13	Bin.0.98	-6.3712	6.7777e-08	7.1689	1.0427e-06
14	Bin.2.66	6.3077	8.482e-08	7.0715	1.2117e-06
15	Bin.0.70	-6.2811	9.3151e-08	7.0308	1.242e-06
16	Bin.8.30	-6.1791	1.3348e-07	6.8746	1.5808e-06
17	Bin.1.02	-6.1772	1.3437e-07	6.8717	1.5808e-06
18	Bin.0.86	-6.1525	1.466e-07	6.8339	1.6289e-06
19	Bin.0.66	-6.0538	2.0751e-07	6.683	2.1843e-06
20	Bin.8.14	-6.0371	2.2004e-07	6.6575	2.2004e-06
21	Bin.0.54	-6.0032	2.4794e-07	6.6057	2.3613e-06
22	Bin.1.70	-5.951	2.9783e-07	6.526	2.7076e-06
23	Bin.8.42	-5.8025	5.0151e-07	6.2997	4.361e-06
24	Bin.0.90	-5.5743	1.1128e-06	5.9536	9.273e-06
25	Bin.8.38	-5.5616	1.1635e-06	5.9343	9.3076e-06
26	Bin.0.58	-5.5212	1.3388e-06	5.8733	1.0298e-05
27	Bin.8.26	-5.4734	1.5809e-06	5.8011	1.171e-05
28	Bin.8.94	-5.4561	1.6783e-06	5.7751	1.1988e-05
29	Bin.1.42	-5.421	1.8958e-06	5.7222	1.3074e-05
30	Bin.2.10	-5.3052	2.8303e-06	5.5482	1.836e-05
31	Bin.1.62	-5.3036	2.8459e-06	5.5458	1.836e-05
32	Bin.8.58	-5.2363	3.5882e-06	5.4451	2.2427e-05
33	Bin.7.54	5.1967	4.1125e-06	5.3859	2.4924e-05
34	Bin.1.66	-5.1807	4.3447e-06	5.362	2.5467e-05
35	Bin.1.86	-5.1733	4.4568e-06	5.351	2.5467e-05
36	Bin.9.18	-5.162	4.6329e-06	5.3342	2.5738e-05
37	Bin.7.82	5.1335	5.1088e-06	5.2917	2.7035e-05
38	Bin.0.46	-5.1319	5.1366e-06	5.2893	2.7035e-05
39	Bin.9.22	-5.0687	6.3764e-06	5.1954	3.2699e-05
40	Bin.0.74	-5.0304	7.2692e-06	5.1385	3.6346e-05
41	Bin.0.22	-4.8515	1.3342e-05	4.8748	6.5084e-05
42	Bin.3.98	4.835	1.4106e-05	4.8506	6.5414e-05
43	Bin.7.86	4.8298	1.4358e-05	4.8429	6.5414e-05
44	Bin.7.66	4.8291	1.4391e-05	4.8419	6.5414e-05
45	Bin.4.26	-4.6109	2.9914e-05	4.5241	0.00013295
46	Bin.0.42	-4.5523	3.6335e-05	4.4397	0.00015798
47	Bin.3.14	4.5264	3.9595e-05	4.4024	0.00016849
48	Bin.7.58	4.4369	5.32e-05	4.2741	0.00022167
49	Bin.0.30	-4.3903	6.1985e-05	4.2077	0.00025212
50	Bin.1.38	-4.3852	6.303e-05	4.2005	0.00025212

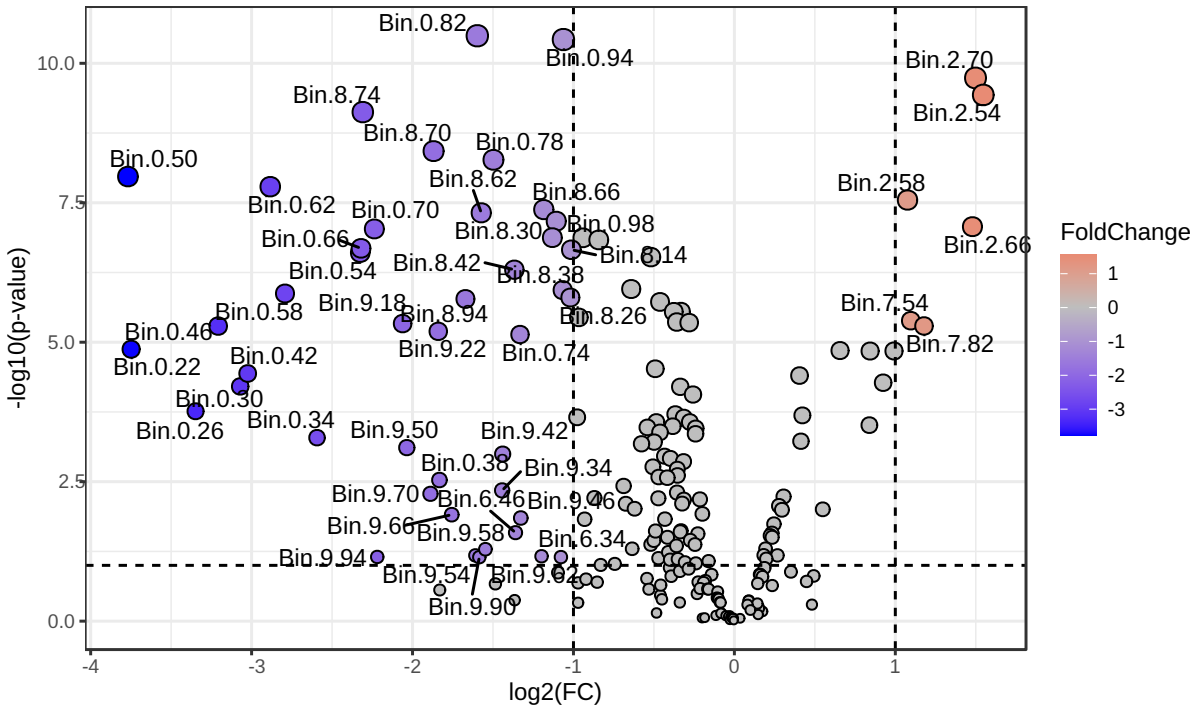

Figure 4: Important features selected by volcano plot with fold change threshold (x) 2 and t-tests threshold (y) 0.1. The red circles represent features above the threshold. Note both fold changes and p values are log transformed. The further its position away from the (0,0), the more significant the feature is.

Table 4: Important features identified by volcano plot

	Spectra Bins	FC	log2(FC)	raw.pval	-log10(p)
1	Bin.0.82	0.33046	-1.5975	3.2089e-11	10.494
2	Bin.0.94	0.47882	-1.0624	3.753e-11	10.426
3	Bin.2.70	2.8251	1.4983	1.8312e-10	9.7373
4	Bin.2.54	2.9209	1.5464	3.6913e-10	9.4328
5	Bin.8.74	0.2018	-2.309	7.5003e-10	9.1249
6	Bin.8.70	0.27372	-1.8692	3.7498e-09	8.426
7	Bin.0.78	0.35423	-1.4972	5.3764e-09	8.2695
8	Bin.0.50	0.073381	-3.7684	1.072e-08	7.9698
9	Bin.0.62	0.13546	-2.8841	1.6289e-08	7.7881
10	Bin.2.58	2.1077	1.0757	2.8257e-08	7.5489
11	Bin.8.66	0.44002	-1.1844	4.1947e-08	7.3773
12	Bin.8.62	0.33612	-1.573	4.7886e-08	7.3198
13	Bin.0.98	0.46426	-1.107	6.7777e-08	7.1689
14	Bin.2.66	2.788	1.4792	8.482e-08	7.0715
15	Bin.0.70	0.21209	-2.2373	9.3151e-08	7.0308
16	Bin.8.30	0.45651	-1.1313	1.3348e-07	6.8746
17	Bin.0.66	0.20028	-2.3199	2.0751e-07	6.683
18	Bin.8.14	0.49549	-1.0131	2.2004e-07	6.6575
19	Bin.0.54	0.19961	-2.3248	2.4794e-07	6.6057
20	Bin.8.42	0.3876	-1.3674	5.0151e-07	6.2997
21	Bin.8.38	0.47723	-1.0672	1.1635e-06	5.9343
22	Bin.0.58	0.14434	-2.7925	1.3388e-06	5.8733
23	Bin.8.26	0.4934	-1.0192	1.5809e-06	5.8011
24	Bin.8.94	0.31398	-1.6713	1.6783e-06	5.7751
25	Bin.7.54	2.1377	1.0961	4.1125e-06	5.3859
26	Bin.9.18	0.23955	-2.0616	4.6329e-06	5.3342
27	Bin.7.82	2.263	1.1782	5.1088e-06	5.2917
28	Bin.0.46	0.10825	-3.2076	5.1366e-06	5.2893
29	Bin.9.22	0.27928	-1.8402	6.3764e-06	5.1954
30	Bin.0.74	0.39715	-1.3323	7.2692e-06	5.1385
31	Bin.0.22	0.074396	-3.7486	1.3342e-05	4.8748
32	Bin.0.42	0.12291	-3.0243	3.6335e-05	4.4397
33	Bin.0.30	0.11901	-3.0708	6.1985e-05	4.2077
34	Bin.0.26	0.098188	-3.3483	0.00017276	3.7626
35	Bin.0.34	0.16568	-2.5935	0.00051411	3.2889
36	Bin.9.50	0.2439	-2.0356	0.00077495	3.1107
37	Bin.9.42	0.3686	-1.4399	0.0010112	2.9952
38	Bin.0.38	0.28071	-1.8328	0.0029506	2.5301
39	Bin.9.34	0.36779	-1.443	0.0045343	2.3435
40	Bin.9.70	0.27004	-1.8887	0.0052184	2.2825
41	Bin.9.66	0.29597	-1.7565	0.012348	1.9084
42	Bin.9.46	0.39847	-1.3274	0.014164	1.8488
43	Bin.6.46	0.38956	-1.3601	0.025931	1.5862
44	Bin.9.58	0.34217	-1.5472	0.051424	1.2888
45	Bin.9.54	0.32773	-1.6094	0.06624	1.1789
46	Bin.9.62	0.43577	-1.1984	0.068678	1.1632
47	Bin.9.94	0.21475	-2.2193	0.070626	1.151
48	Bin.6.34	0.47367	-1.078	0.071057	1.1484
49	Bin.9.90	0.33333	-1.585	0.071343	1.1466

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 5 shows the overall correlation heatmap.

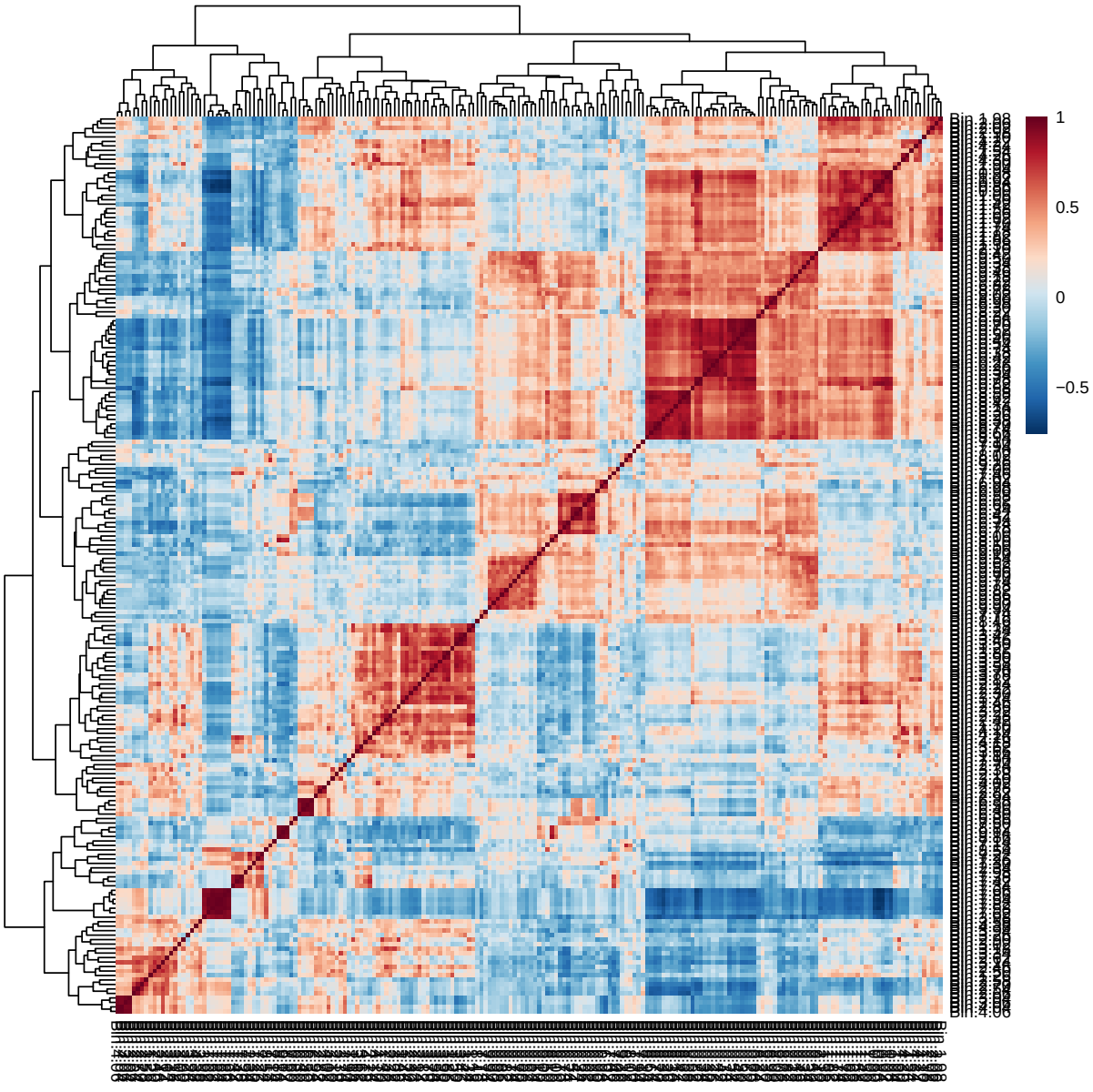

Figure 5: Correlation Heatmaps

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 6 is pairwise score plots providing an overview of the various seperation patterns among the most significant PCs; Figure 7 is the scree plot showing the variances explained by the selected PCs; Figure 8 shows the 2-D scores plot between selected PCs; Figure 9 shows the biplot between the selected PCs. Interactive 3-D scores plots are not included here and can be directly downloaded from website.

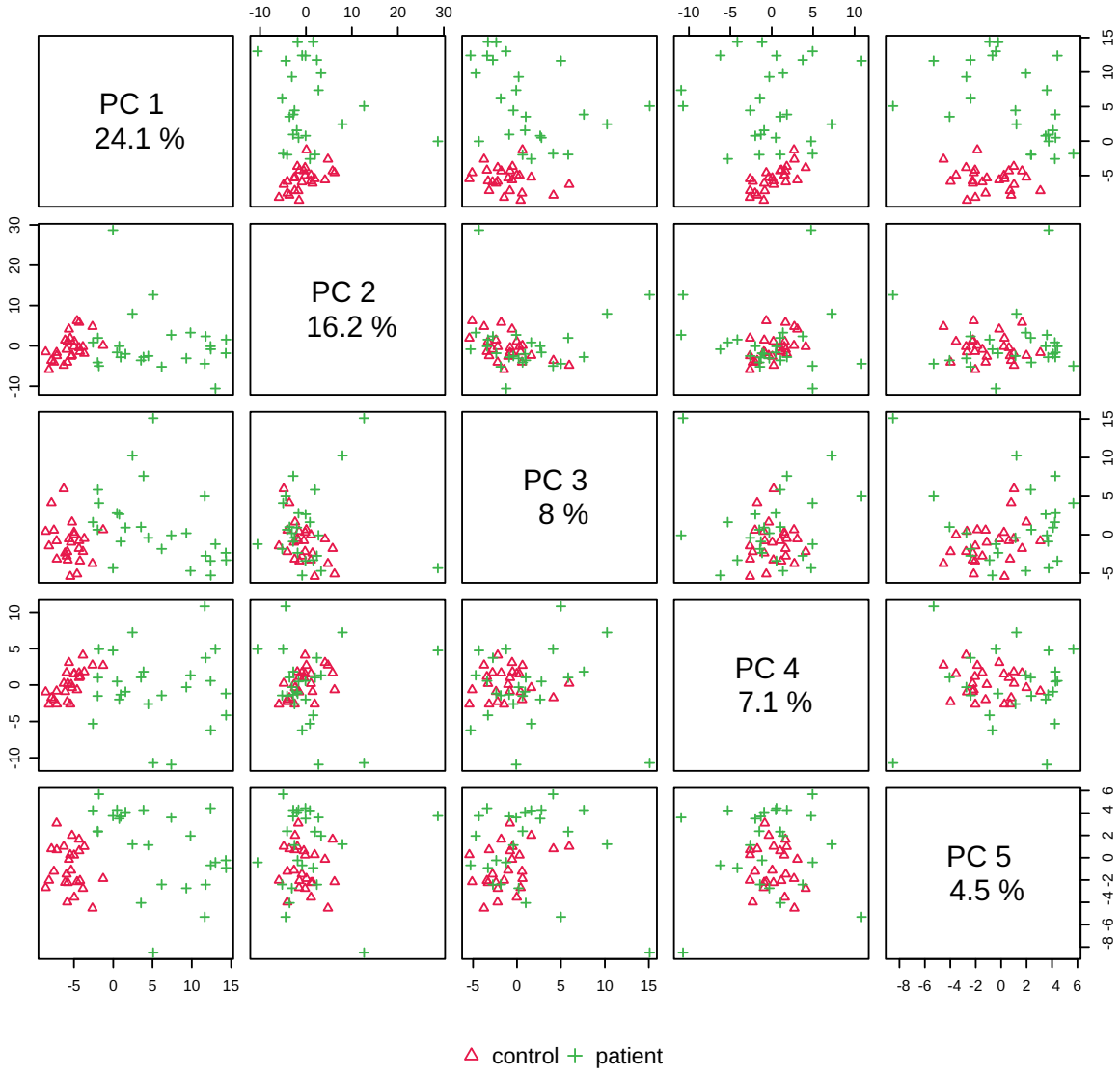

Figure 6: Pairwise score plots between the selected PCs. The explained variance of each PC is shown in the corresponding diagonal cell.

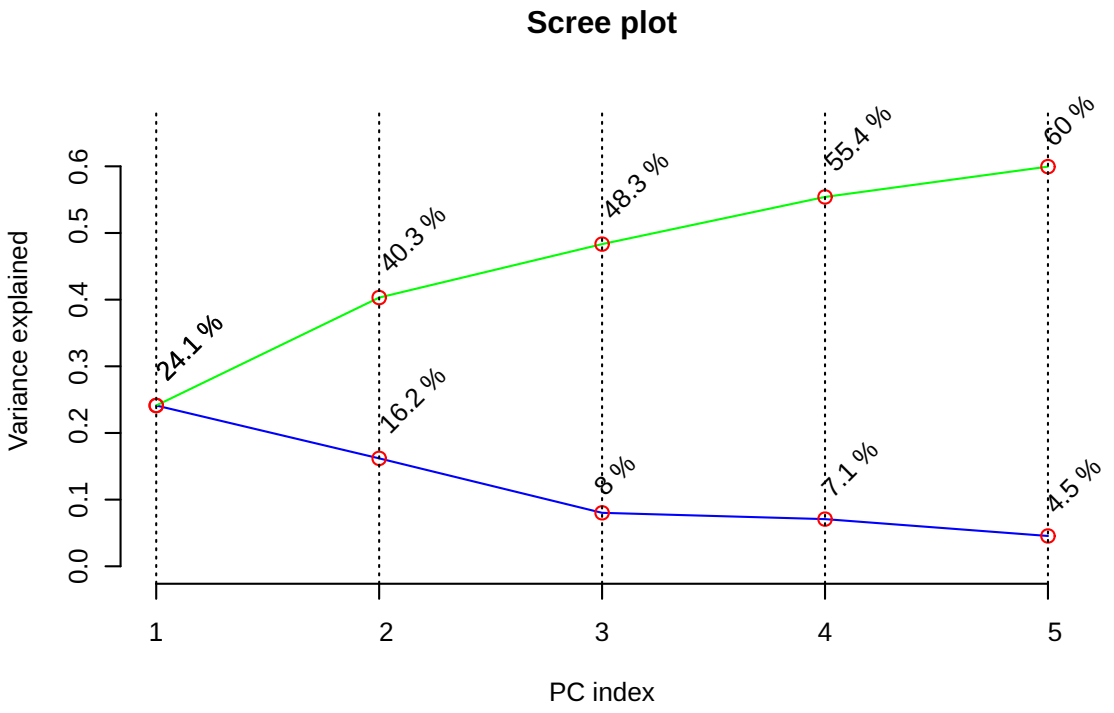

Figure 7: Scree plot shows the variance explained by PCs. The green line on top shows the accumulated variance explained; the blue line underneath shows the variance explained by individual PC.

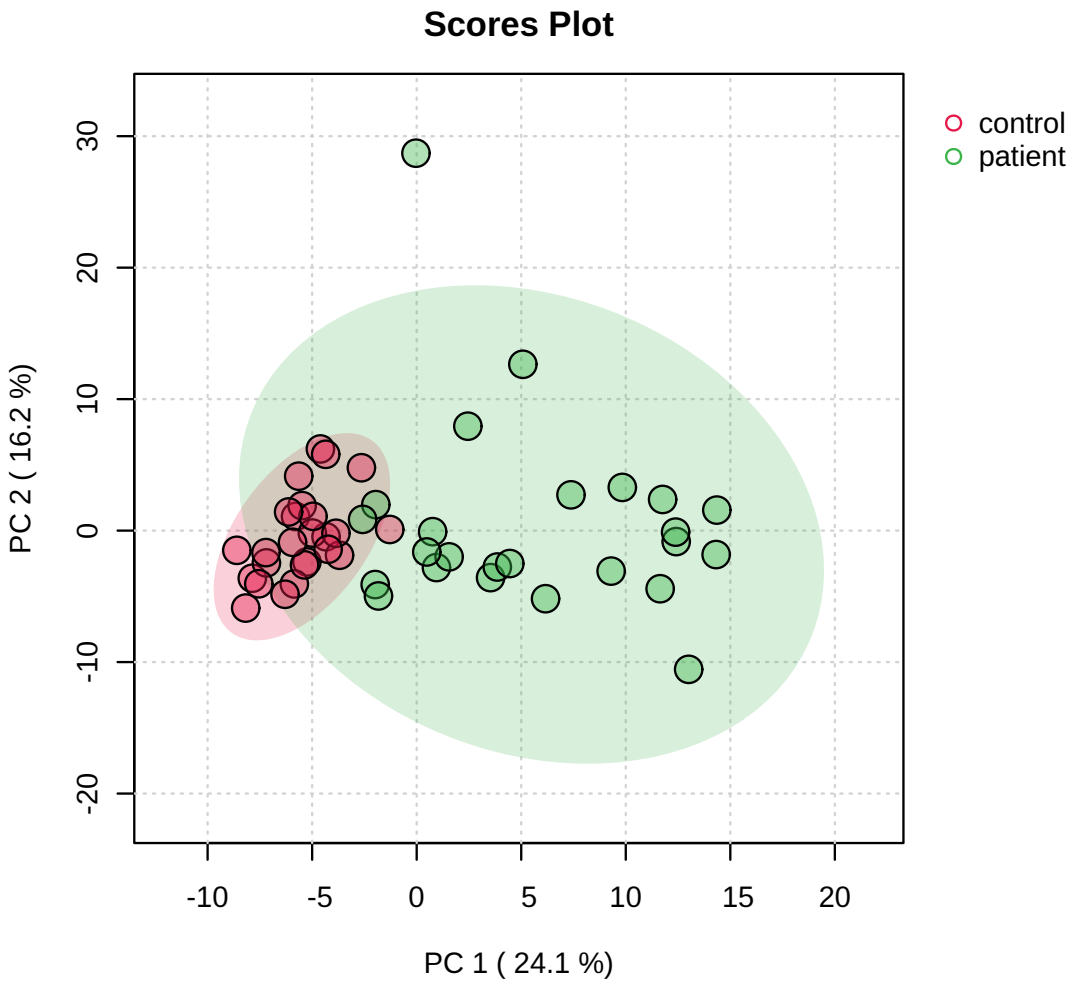

Figure 8: Scores plot between the selected PCs. The explained variances are shown in brackets.

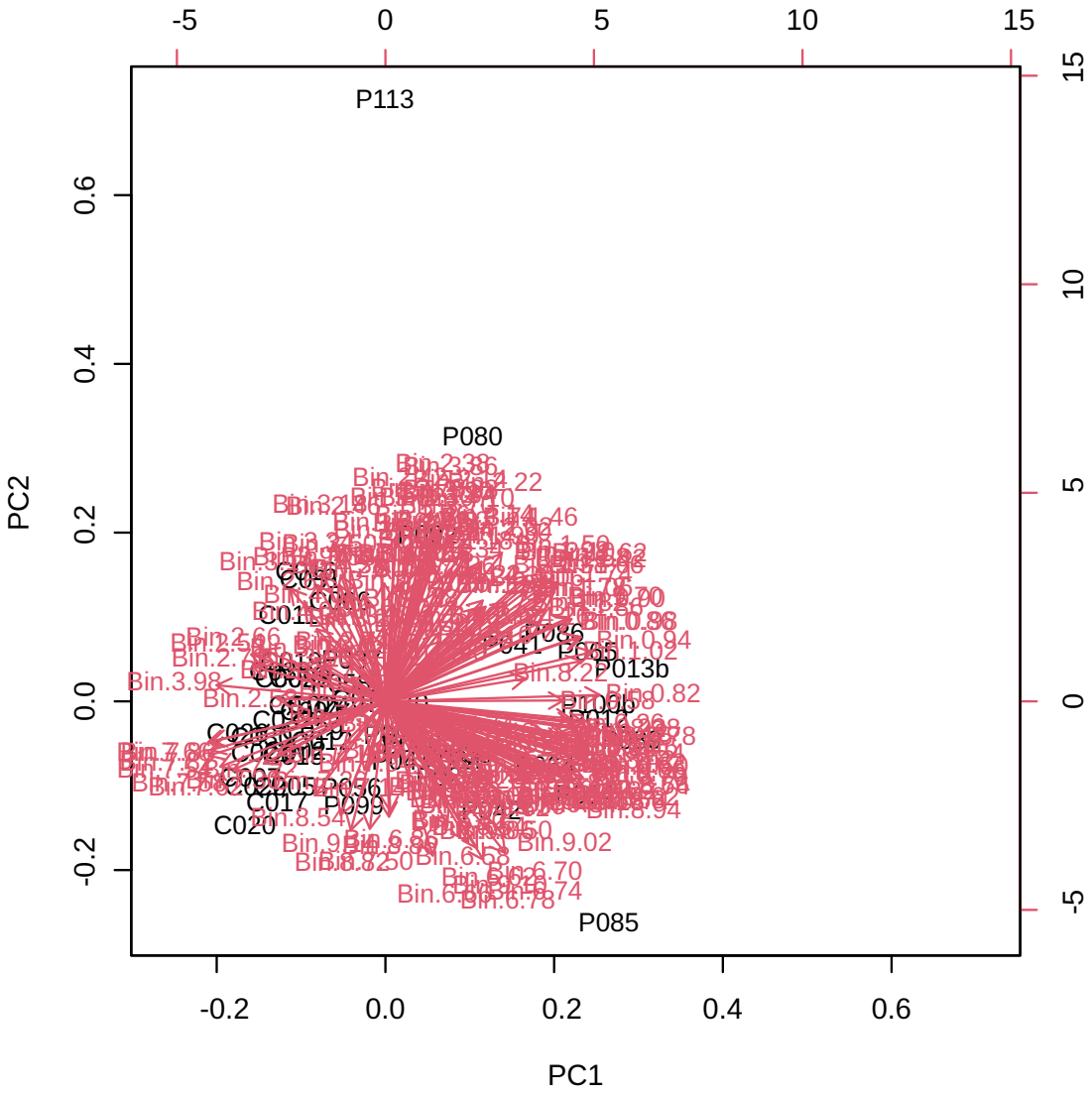

Figure 9: PCA biplot between the selected PCs. Note, you may want to test different centering and scaling normalization methods for the biplot to be displayed properly.

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 10 shows the clustering result in the form of a dendrogram. Figure 11 shows the clustering result in the form of a heatmap.

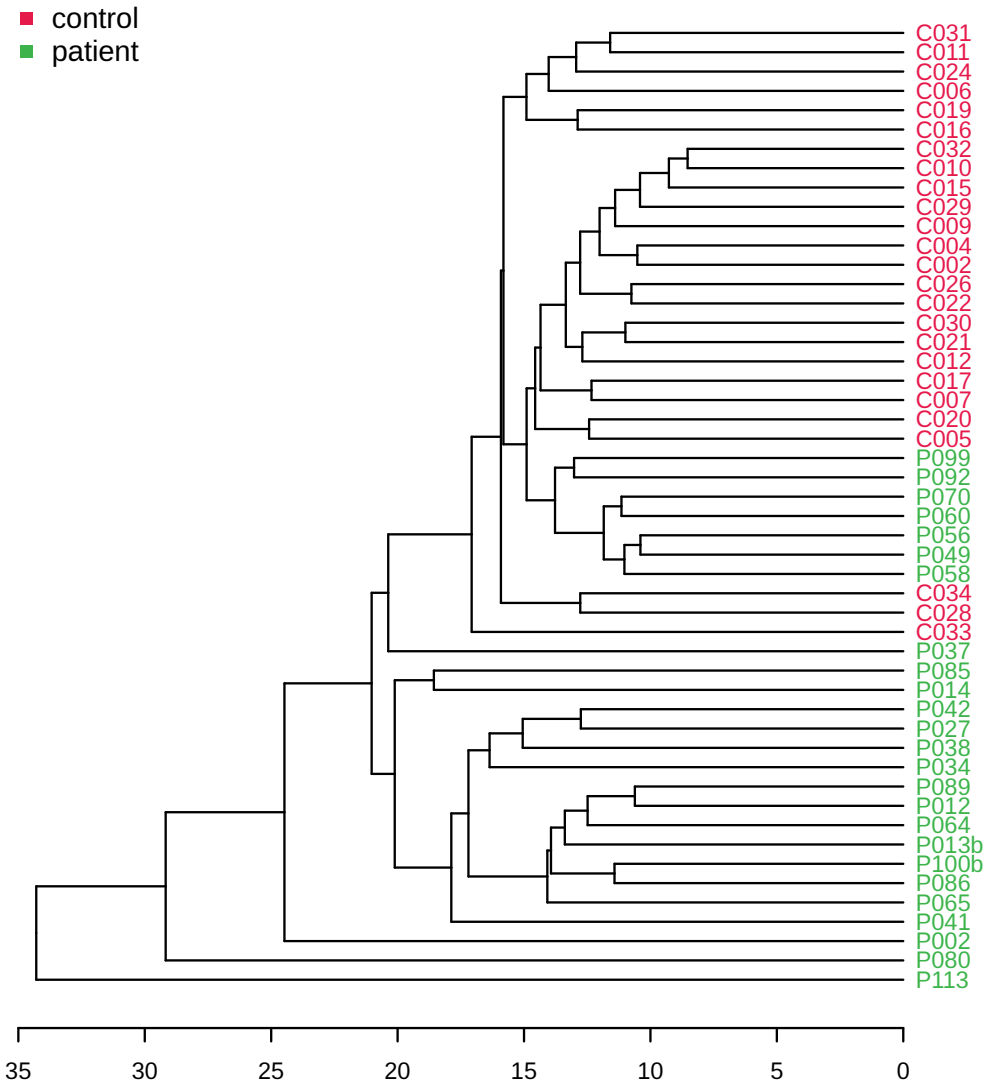

Figure 10: Clustering result shown as dendrogram (distance measure using `euclidean`, and clustering algorithm using `average`).

2.5 K-means Clustering

K-means clustering is a nonhierarchical clustering technique. It begins by creating k random clusters (k is supplied by user). The program then calculates the mean of each cluster. If an observation is closer to the centroid of another cluster then the observation is made a member of that cluster. This process is repeated until none of the observations are reassigned to a different cluster.

K-means analysis is performed using the `kmeans` function in the package `stat`. Figure 12 shows clustering the results. Table 5 shows the members in each cluster from K-means analysis.

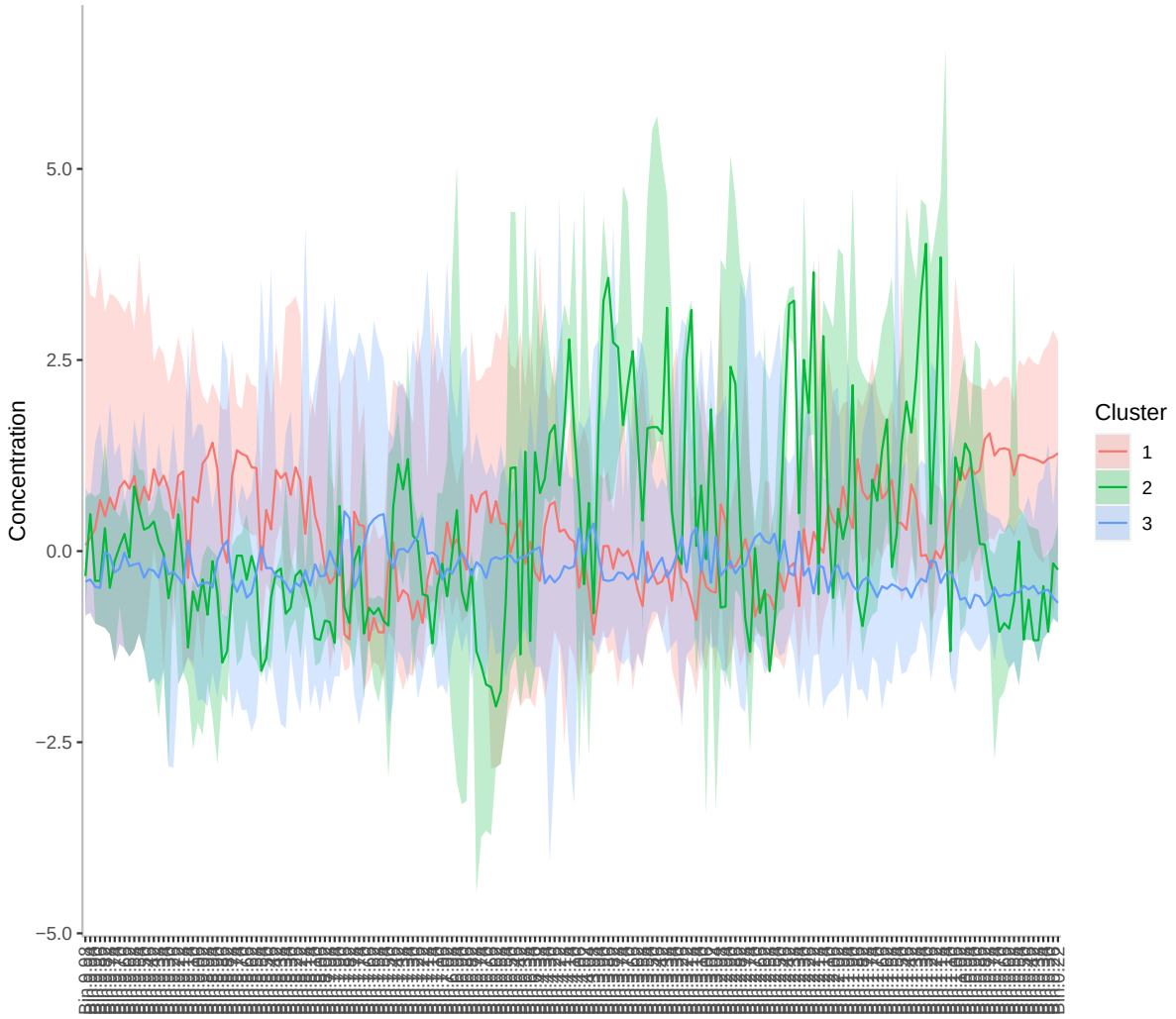

Figure 11: K-means cluster analysis. The x-axes are variable indices and y-axes are relative intensities. The blue lines represent median intensities of corresponding clusters

Table 5: Clustering result using K-means

	Samples in each cluster
Cluster(1)	P012 P014 P027 P034 P038 P041 P042 P064 P065 P085 P086 P089 P013b P100b
Cluster(2)	P002 P080 P113
Cluster(3)	C002 C004 C005 C006 C007 C009 C010 C011 C012 C015 C016 C017 C019 C020 C021 C022 C024 C026 C028 C029 C030 C031 C032 C033 C034 P037 P049 P056 P058 P060 P070 P092 P099

2.6 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 6 shows the confusion matrix of random forest. Figure 13 shows the cumulative error rates of random forest analysis for given parameters. Figure 14 shows the important features ranked by random forest. Figure 15 shows the outlier measures of all samples for the given parameters. The OOB error is 0.04

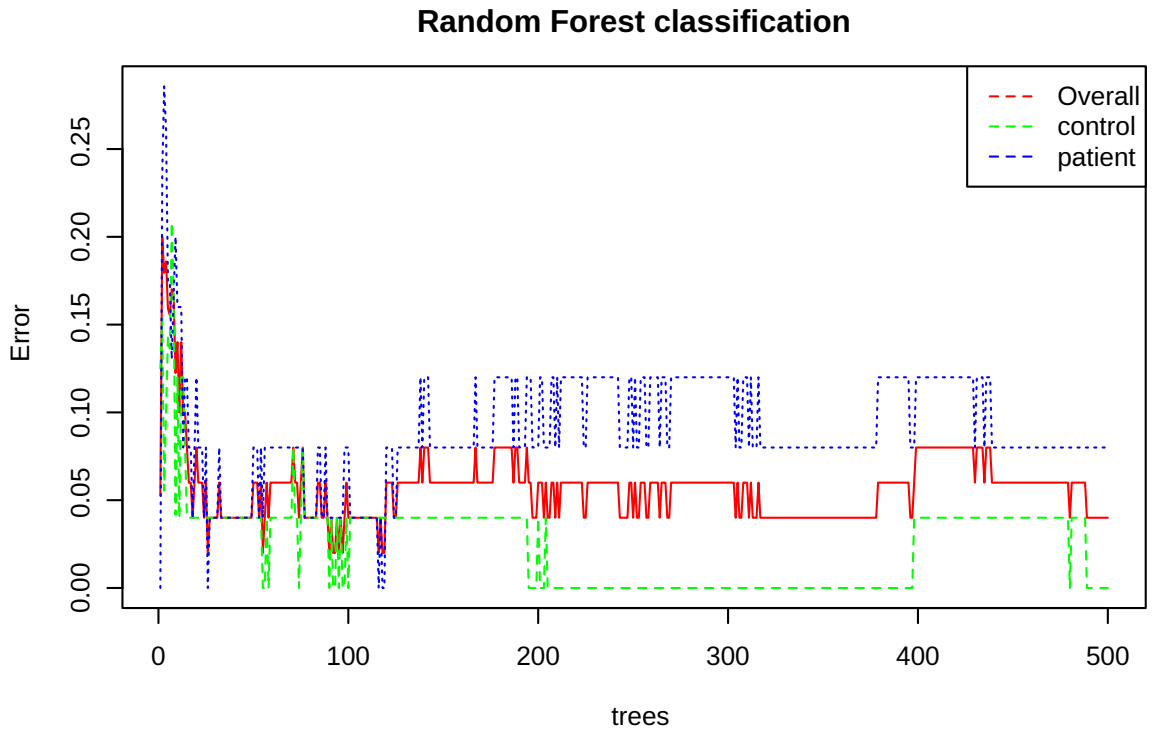

Figure 12: Cumulative error rates by Random Forest classification. The overall error rate is shown as the black line; the red and green lines represent the error rates for each class.

	control	patient	class.error
control	25.00	0.00	0.00
patient	2.00	23.00	0.08

Table 6: Random Forest Classification Performance

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

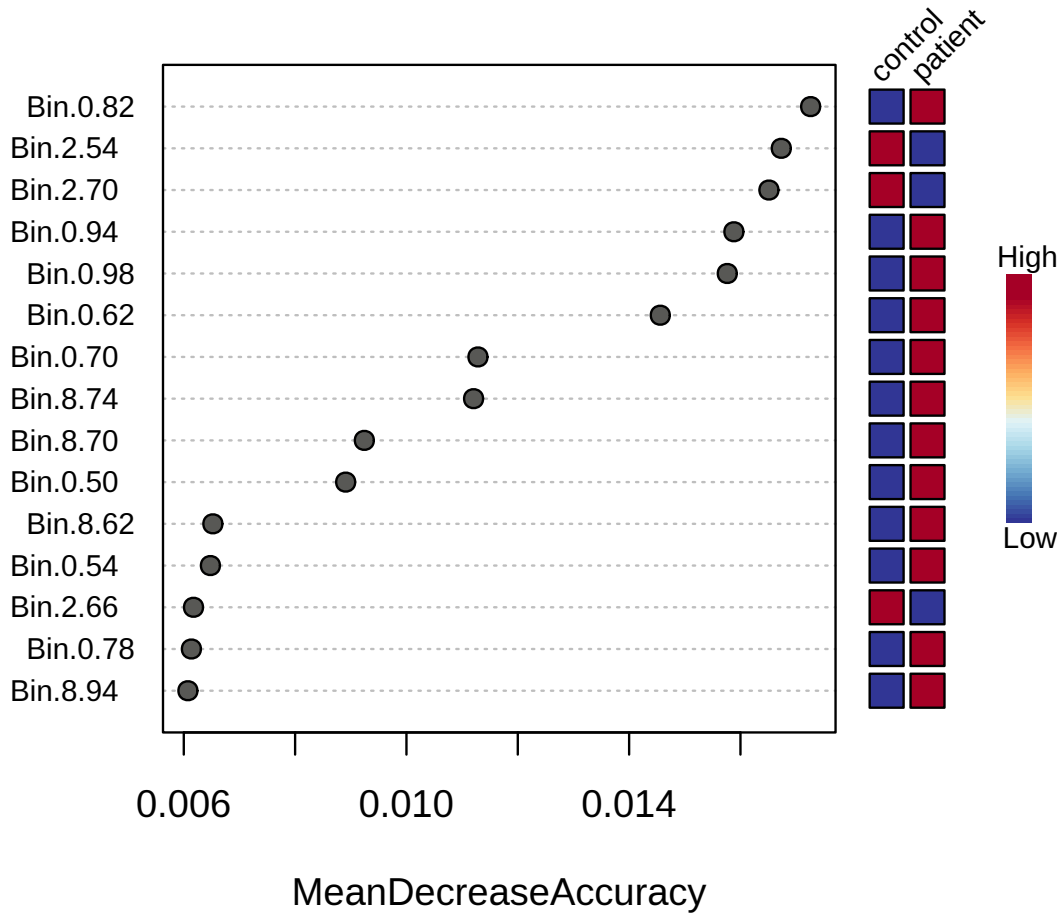

Figure 13: Significant features identified by Random Forest. The features are ranked by the mean decrease in classification accuracy when they are permuted.

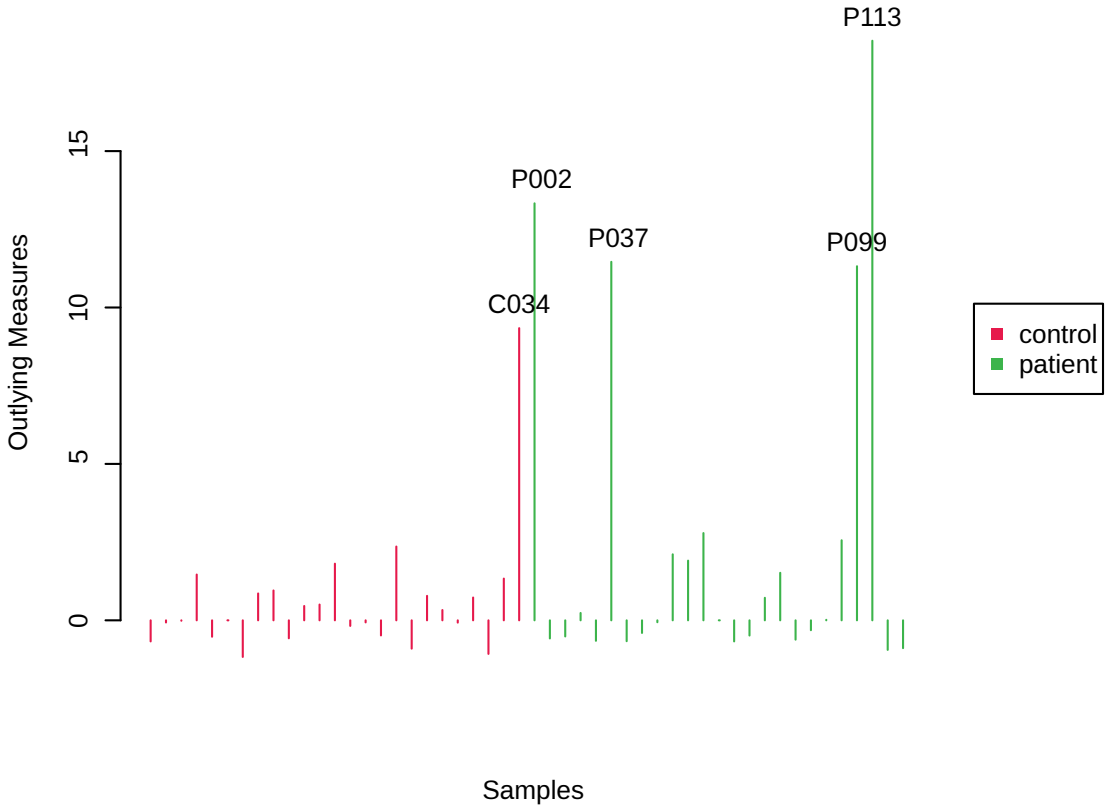

Figure 14: Potential outliers identified by Random Forest. Only the top five are labeled.

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<-SanityCheckData(mSet)"
 [5] "mSet<-ReplaceMin(mSet);"
 [6] "mSet<-SanityCheckData(mSet)"
 [7] "mSet<-FilterVariable(mSet, \"F\", 25, \"none\", -1, \"mean\", 0)"
 [8] "mSet<-PreparePrenormData(mSet)"
 [9] "mSet<-Normalization(mSet, \"MedianNorm\", \"SrNorm\", \"AutoNorm\", ratio=FALSE, ratioNum=20)"
[10] "mSet<-PlotNormSummary(mSet, \"norm_0_\", \"png\", 72, width=NA)"
[11] "mSet<-PlotSampleNormSummary(mSet, \"snorm_0_\", \"png\", 72, width=NA)"
[12] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[13] "mSet<-PlotTT(mSet, \"tt_0_\", \"png\", 72, width=NA)"
[14] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[15] "mSet<-PlotTT(mSet, \"tt_1_\", \"png\", 72, width=NA)"
[16] "mSet<-PlotCorrHeatMap(mSet, \"corr_1_\", \"png\", 72, width=NA, \"col\", \"pearson\", \"bwm\",
[17] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[18] "mSet<-PlotTT(mSet, \"tt_2_\", \"png\", 72, width=NA)"
[19] "mSet<-PlotTT(mSet, \"tt_2_\", \"png\", 300, width=NA)"
[20] "mSet<-Kmeans.Anal(mSet, 3)"
[21] "mSet<-PlotKmeans(mSet, \"km_0_\", \"png\", 72, width=NA, \"default\", \"F\")"
[22] "mSet<-PlotClustPCA(mSet, \"km_pca_0_\", \"png\", 72, width=NA, \"default\", \"km\", \"F\")"
[23] "mSet<-FC.Anal(mSet, 2.0, 0, FALSE)"
[24] "mSet<-PlotFC(mSet, \"fc_0_\", \"png\", 72, width=NA)"
[25] "mSet<-PlotCorrHeatMap(mSet, \"corr_2_\", \"png\", 72, width=NA, \"col\", \"pearson\", \"bwm\",
[26] "mSet<-RF.Anal(mSet, 500,7,1)"
[27] "mSet<-PlotRF.Classify(mSet, \"rf_cls_0_\", \"png\", 72, width=NA)"
[28] "mSet<-PlotRF.VIP(mSet, \"rf_imp_0_\", \"png\", 72, width=NA)"
[29] "mSet<-PlotRF.Outlier(mSet, \"rf_outlier_0_\", \"png\", 72, width=NA)"
[30] "mSet<-RF.Anal(mSet, 500,7,1)"
[31] "mSet<-PlotRF.Classify(mSet, \"rf_cls_1_\", \"png\", 72, width=NA)"
[32] "mSet<-PlotRF.VIP(mSet, \"rf_imp_1_\", \"png\", 72, width=NA)"
[33] "mSet<-PlotRF.Outlier(mSet, \"rf_outlier_1_\", \"png\", 72, width=NA)"
[34] "mSet<-PlotHCTree(mSet, \"tree_0_\", \"png\", 72, width=NA, \"euclidean\", \"ward.D\")"
[35] "mSet<-PlotHCTree(mSet, \"tree_1_\", \"png\", 72, width=NA, \"euclidean\", \"ward.D\")"
[36] "mSet<-PlotHCTree(mSet, \"tree_2_\", \"png\", 72, width=NA, \"euclidean\", \"average\")"
[37] "mSet<-GetGroupNames(mSet, \"null\")"
[38] "mSet<-Volcano.Anal(mSet, FALSE, 2.0, 0, F, 0.1, TRUE, \"raw\")"
[39] "mSet<-PlotVolcano(mSet, \"volcano_0_\",1, 0, \"png\", 72, width=NA, -1)"
[40] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[41] "mSet<-PlotTT(mSet, \"tt_3_\", \"png\", 72, width=NA)"
[42] "mSet<-UpdateLoadingCmpd(mSet, \"Bin.8.74\")"
[43] "mSet<-SetCmpdSummaryType(mSet, \"violin\")"
[44] "mSet<-PlotCmpdSummary(mSet, \"Bin.8.74\",\"NA\",\"NA\", 0, \"png\", 72)"
[45] "mSet<-Ttests.Anal(mSet, T, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[46] "mSet<-PlotTT(mSet, \"tt_4_\", \"png\", 72, width=NA)"
[47] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, FALSE, \"fdr\", FALSE)"
[48] "mSet<-PlotTT(mSet, \"tt_5_\", \"png\", 72, width=NA)"
[49] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[50] "mSet<-PlotTT(mSet, \"tt_6_\", \"png\", 72, width=NA)"
[51] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"raw\", FALSE)"
[52] "mSet<-PlotTT(mSet, \"tt_7_\", \"png\", 72, width=NA)"
[53] "mSet<-Ttests.Anal(mSet, F, 0.05, FALSE, TRUE, \"fdr\", FALSE)"
[54] "mSet<-PlotTT(mSet, \"tt_8_\", \"png\", 72, width=NA)"
[55] "mSet<-PCA.Anal(mSet)"
[56] "mSet<-PlotPCAPairSummary(mSet, \"pca_pair_0_\", \"png\", 72, width=NA, 5)"
```

```
[57] "mSet<-PlotPCAScree(mSet, \"pca_scree_0_\", \"png\", 72, width=NA, 5)"
[58] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_0_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[59] "mSet<-PlotPCALoading(mSet, \"pca_loading_0_\", \"png\", 72, width=NA, 1,2);"
[60] "mSet<-PlotPCABiplot(mSet, \"pca_biplot_0_\", \"png\", 72, width=NA, 1,2)"
[61] "mSet<-PlotPCA3DLoading(mSet, \"pca_loading3d_0_\", \"json\", 1,2,3)"
[62] "mSet<-PlotPCAPairSummary(mSet, \"pca_pair_1_\", \"png\", 72, width=NA, 5)"
[63] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_1_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[64] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_2_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[65] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_3_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[66] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_4_\", \"png\", 72, width=NA, 1,2,0.95,0,0, \"na\")"
[67] "mSet<-SaveTransformedData(mSet)"
[68] "mSet<-PreparePDFReport(mSet, \"guest34457525325944116 90\")\n"
```

The report was generated on Thu Apr 4 13:37:17 2024 with R version 4.3.2 (2023-10-31), OS system: Linux, version: -Ubuntu SMP Tue Mar 5 20:16:58 UTC 2024 .