Summary statistics: Lymphoma predictive modeling

Sleiman Bassim, PhD

Project started in December 2017
Original version published in August 2018
Last updated on September 12, 2018

Contents

Loading packages.

```
pkgs <- c('gdata','lattice','latticeExtra',
          'ggplot2', 'dplyr', 'tidyr', 'RColorBrewer','igraph',
          'DescTools', 'scales', 'brotools', 'Hmisc', 'finalfit',
          'plyr', 'paletteer', 'ggrepel', 'ggExtra', 'ggpubr',
          'cowplot', 'ggridges', 'reshape')
lapply(pkgs, require, character.only = TRUE)
```

1 Exploratory Data Analysis

Data is from individuals with Lymphoma tumors, either undergone or not a Rituximab CHOP treatment. Some individuals show relapse after treatment. Tumors migrate though nodal (lymphnodes) or extranodal tissues. Tumors involve two different subtypes of cells of origin, ABC or GCB. **The first aim is to find correlation genes that respond differently to treatment, nodal transmission, and cell subtypes.**

OR: Odds ratio. HR: Hazard ratio

```
#read.table("data/phenodata", sep = "\t", header = T) %>%
#    dplyr::select(SAMPLE_ID, Timepoint,
#    GROUP, SITE, Score, Prediction, ABClikelihood) %>%
#    brotools::describe()

print_summary_table <- function(features, dependent, df, execute = TRUE) {
    if ( execute == TRUE ) {
        x <- df %>%
            summary_factorlist(dependent, features, p=FALSE, add_dependent_label=TRUE)
        ## print latex table
        Hmisc::latex(x, file = "", booktabs = TRUE, title = "")
    } else {
        cat("LaTeX summary table printed\n")
    }
}

dfs <- read.table("data/phenodata", sep = "\t", header = T)
print_summary_table(features= c("Score", "ABClikelihood", "GROUP"),
                    dependent= c("Prediction"),
                    df = dfs,
                    execute = F)

LaTeX summary table printed
```

	Dependent: Prediction		ABC	GCB	U
10	Score	Mean (SD)	3156.3 (475.5)	506.4 (721.1)	2162.8 (143.6)
1	ABClikelihood	Mean (SD)	1 (0)	0 (0)	0.5 (0.4)
2	GROUP	CNS DIAGNOSIS	4 (33.3)	6 (50.0)	2 (16.7)
3		CNS RELAPSE CHOP or EQUIVALENT	6 (60.0)	3 (30.0)	1 (10.0)
4		CNS RELAPSE RCHOP	17 (44.7)	13 (34.2)	8 (21.1)
5		NO RELAPSE	27 (28.1)	52 (54.2)	17 (17.7)
6		NORMAL ABC CONTROL	2 (100.0)	0 (0.0)	0 (0.0)
7		NORMAL GCB CONTROL	0 (0.0)	4 (100.0)	0 (0.0)
8		SYTEMIC RELAPSE NO CNS	31 (48.4)	25 (39.1)	8 (12.5)
9		TESTICULAR NO CNS RELAPSE	9 (75.0)	0 (0.0)	3 (25.0)

1.1 Data reformating

In the first steps of the analysis, the samples will be labeled (supervised) into the following categories (based on patients diagnosis).

```
metadata <- read.table("data/phenodata", sep = "\t", header = T) %>%
    dplyr::select(SAMPLE_ID, Timepoint, GROUP, SITE, Score, Prediction, ABClikelihood) %>%
    filter(Timepoint != "T2") %>%
    mutate(Groups = case_when(GROUP %in% c("CNS_RELAPSE_RCHOP",
                                            "CNS_RELAPSE_CHOPorEQUIVALENT",
                                            "CNS_DIAGNOSIS") ~ "CNS",
                               GROUP %in% c("TESTICULAR_NO_CNS_RELAPSE", "NO_RELAPSE") ~ "NOREL",
                               GROUP == "SYTEMIC_RELAPSE_NO_CNS" ~ "SYST",
                               TRUE ~ "CTRL")) %>%
    filter(Groups != "CTRL") %>%
    mutate(ABClassify = case_when(ABClikelihood >= .9 ~ "ABC",
                                  ABClikelihood <= .1 ~ "GCB",
                                  TRUE ~ "U")) %>%
    mutate(ABCScore = case_when(Score > 2412 ~ "ABC",
                                Score <= 1900 ~ "GCB",
#                                Score == NA ~ "NA",
                                TRUE ~ "U")) %>%
    mutate(Nodes = case_when(SITE == "LN" ~ "LN",
                             SITE == "TO" ~ "LN",
                             SITE == "SP" ~ "LN",
                             TRUE ~ "EN")) %>%
    mutate(Lymphnodes = case_when(Nodes == "LN" ~ 1, TRUE ~ 0))

# factorize
metadata$Groups <- as.factor(metadata$Groups)
metadata$ABClassify <- as.factor(metadata$ABClassify)
metadata$ABCScore <- as.factor(metadata$ABCScore)
metadata$Nodes <- as.factor(metadata$Nodes)
metadata$Lymphnodes <- as.factor(metadata$Lymphnodes)

## reorder sample names
ids <- read.table("data/sampleIDs")
metadata <- arrange(metadata, factor(SAMPLE_ID, levels = ids$V1))
meta.selected <- metadata %>%
    mutate(Contrast1 = as.factor(paste0(Groups, ".", Prediction))) %>%
    mutate(Contrast2 = as.factor(paste0(Groups, ".", Nodes)))

#brotools::describe(metadata)
print_summary_table(c("ABCScore", "ABClassify", "GROUP",
                      "Contrast1", "Contrast2"), c("Nodes"), meta.selected, execute = F)

LaTeX summary table printed
```

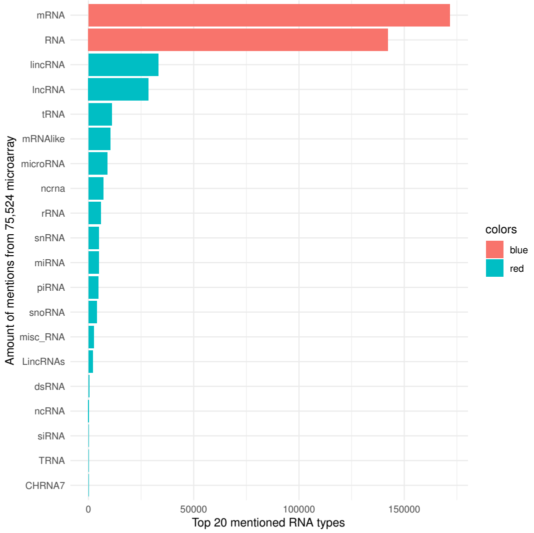
	Dependent: Nodes		EN	LN
4	ABCScore	ABC	34 (37.0)	58 (63.0)
5		GCB	36 (35.0)	67 (65.0)
6		U	16 (39.0)	25 (61.0)
1	ABClassify	ABC	37 (35.9)	66 (64.1)
2		GCB	38 (32.5)	79 (67.5)
3		U	11 (68.8)	5 (31.2)
7	GROUP	CNS DIAGNOSIS	7 (63.6)	4 (36.4)
8		CNS RELAPSE CHOP or EQUIVALENT	5 (62.5)	3 (37.5)
9		CNS RELAPSE RCHOP	20 (51.3)	19 (48.7)
10		NO RELAPSE	30 (31.2)	66 (68.8)
11		NORMAL ABC CONTROL	2 (NA)	0 (0.0)
12		NORMAL GCB CONTROL	0 (0.0)	4 (100.0)
13		SYTEMIC RELAPSE NO CNS	10 (15.6)	54 (84.4)
14		TESTICULAR NO CNS RELAPSE	12 (100.0)	0 (0.0)

1.1.1 On-array content of non-coding RNA

One microarray contains 70,524 probes, of which 12,969 genes do not contain any of the terms related to non-coding RNAs. The graph shows in red that the coding genes are categorized as mRNA or RNA. The non-coding RNAs are in blue and there is over 600 different mentions among the probes (large intergenic non-coding RNAs, lincRNAs make up most of the long ncRNAs). Even though they are less *mentioned* in gene annotations (smaller bars) they do however cover 70% of the annotated probes.

↰ Two or more mentions (patterns) can be recognized in one gene

```
colors <- c(rep("blue",2), rep("red",18))
read.table("./data/rna.patterns.annotated.array.txt", header = F) %>%
    mutate(percent = (V1/5 / (sum(V1)/5)) * 100) %>%
    slice(1:20) %>%
    ggplot(aes(x = reorder(V2, percent),
               y = V1,
               fill = colors)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    coord_flip() +
    theme_minimal() +
    labs(x = "Amount of mentions from 75,524 microarray",
        y = "Top 20 mentioned RNA types")
```



1.1.2 Regression analyses to quantify diagnosis connections

Logistic regression of binomial factoring between nodal/extranodal diagnosis and individuals labels for cell-of-origin classification and CNS relapse or systemic relapse. Regression model summary with odds ratio with 95% confidence interval to quantify how much nodal and extranodal diagnosis is associated with the cell-of-origin ABC or GCB nature in DLBCL individuals with CNS, systemic or no relapse.

```
fit_summary_table <- function(features, dependent, df, method, execute = TRUE) {
    if ( execute == TRUE ) {
        if ( method == "glm" || method == "cox" ) {
            x <- df %>%
                finalfit(dependent, features)
        } else if ( execute == "glmer" ) {
            x <- df %>%
                finalfit(dependent, features,
                         mixed, random_effect)
        }
        ## print latex table
        Hmisc::latex(x, file = "", booktabs = TRUE, title = "")
    } else {
        cat("LaTeX summary table printed\n")
    }
}


fit_summary_table(features= c("ABCScore", "ABClassify", "GROUP"),
                  dependent= c("Nodes"),
                  df = metadata,
                  method = "glm",
                  execute = F)

LaTeX summary table printed
```

	Dependent: Nodes		EN	LN	OR (univariable)	OR (multivariable)
4	ABCScore	ABC	34 (39.5)	58 (38.7)	-	-
5		GCB	36 (41.9)	67 (44.7)	1.09 (0.61-1.96, p=0.771)	0.44 (0.06-3.23, p=0.408)
6		U	16 (18.6)	25 (16.7)	0.92 (0.43-1.97, p=0.820)	0.96 (0.25-4.74, p=0.952)
1	ABClassify	ABC	37 (43.0)	66 (44.0)	-	-
2		GCB	38 (44.2)	79 (52.7)	1.17 (0.67-2.04, p=0.591)	1.61 (0.24-10.98, p=0.615)
3		U	11 (12.8)	5 (3.3)	0.25 (0.08-0.76, p=0.018)	0.52 (0.07-2.97, p=0.473)
7	GROUP	CNS DIAGNOSIS	7 (8.1)	4 (2.7)	-	-
8		CNS RELAPSE CHOP or EQUIVALENT	5 (5.8)	3 (2.0)	1.05 (0.15-7.08, p=0.960)	0.97 (0.13-6.76, p=0.979)
9		CNS RELAPSE RCHOP	20 (23.3)	19 (12.7)	1.66 (0.43-7.21, p=0.470)	1.71 (0.42-7.73, p=0.461)
10		NO RELAPSE	30 (34.9)	66 (44.0)	3.85 (1.08-15.64, p=0.042)	3.40 (0.91-14.2, p=0.074)
11		NORMAL ABC CONTROL	2 (2.3)	0 (0.0)	0.00 (NA-NA, p=0.995)	0.00 (NA-NA, p=0.995)
12		NORMAL GCB CONTROL	0 (0.0)	4 (2.7)	74.56 (0.00-NA, p=0.993)	79.25 (0.00-NA, p=0.993)
13		SYTEMIC RELAPSE NO CNS	10 (11.6)	54 (36.0)	9.45 (2.42-NA, p=0.002)	8.07 (1.98-NA, p=0.004)
14		TESTICULAR NO CNS RELAPSE	12 (14.0)	0 (0.0)	0.00 (0.00-NA, p=0.988)	0.00 (0.00-NA, p=0.988)

Mixed effects multilevel logistic regression model fit to find connections between individuals (CNS relapse, systemic, and no relapse) and cell-of-origin predictions (ABC, GCB likelihoods), while considering nodal and extranodal involvement in the relapse (diagnosed tissue sites with cancer invasion).

```
mixed = c("GROUP")
random_effect = c("SITE")
fit_summary_table(features= c("Prediction", "GROUP"),
                  dependent= c("Nodes"),
                  df = metadata,
                  method = "glmer",
                  execute = F)

LaTeX summary table printed
```

	Dependent: Nodes		EN	LN	OR (univariable)	OR (multilevel)
9	Prediction	ABC	34 (40.5)	58 (38.7)	-	-
10		GCB	36 (42.9)	67 (44.7)	1.09 (0.61-1.96, p=0.771)	-
11		U	14 (16.7)	25 (16.7)	1.05 (0.48-2.32, p=0.908)	-
1	GROUP	CNS DIAGNOSIS	7 (8.1)	4 (2.7)	-	-
2		CNS RELAPSE CHOP or EQUIVALENT	5 (5.8)	3 (2.0)	1.05 (0.15-7.08, p=0.960)	0.38 (0.00-NA, p=0.989)
3		CNS RELAPSE RCHOP	20 (23.3)	19 (12.7)	1.66 (0.43-7.21, p=0.470)	0.49 (0.00-NA, p=0.988)
4		NO RELAPSE	30 (34.9)	66 (44.0)	3.85 (1.08-15.64, p=0.042)	1.70 (0.00-NA, p=0.989)
5		NORMAL ABC CONTROL	2 (2.3)	0 (0.0)	0.00 (NA, p=0.995)	0.00 (0.00-Inf, p=1.000)
6		NORMAL GCB CONTROL	0 (0.0)	4 (2.7)	NA (0.00-NA, p=0.993)	285412.87 (0.00-Inf, p=0.999)
7		SYTEMIC RELAPSE NO CNS	10 (11.6)	54 (36.0)	9.45 (2.42-42.22, p=0.002)	1.76 (0.00-NA, p=0.989)
8		TESTICULAR NO CNS RELAPSE	12 (14.0)	0 (0.0)	0.00 (0.00-NA, p=0.988)	0.00 (0.00-Inf, p=1.000)

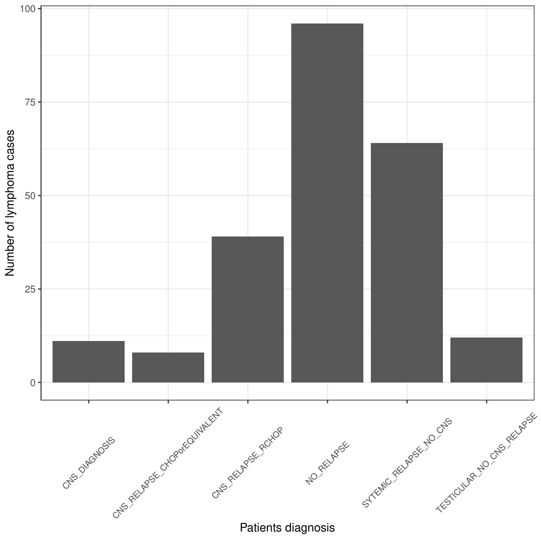
1.2 Featured data and groups of sample cases

Difference in cases being indexed based on their *cell-of-origin* association subtypes using either of the following features: prediction, ABClassify, ABCScore.

```
metadata %>%
    select(Prediction, ABClassify, ABCScore) %>%
    summary

 Prediction ABClassify ABCScore
 ABC :90    ABC:101    ABC:90
 GCB :99    GCB:113    GCB:99
 U   :39    U  : 16    U  :41
 NA's: 2
```

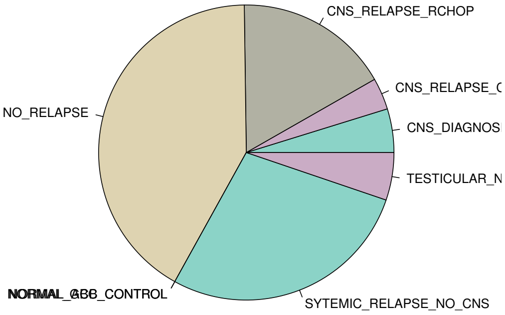
Distribution of samples with different treatments.

```
metadata %>%
    select(GROUP) %>%
    ggplot(aes(x = GROUP)) +
    geom_histogram(stat = "count") +
    labs(y = "Number of lymphoma cases",
         x = "Patients diagnosis") +
    theme_bw() +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))

## Warning: Ignoring unknown parameters: binwidth, bins, pad
```



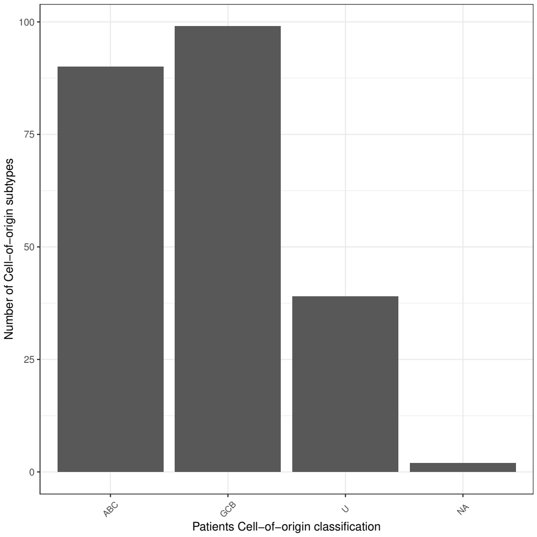
Or as a pie chart.

```
palette.pies <- brewer.pal(12, name = "Set3")
palette.pies.adj <- colorRampPalette(palette.pies)(length(unique(metadata$GROUP)))
pie(table(metadata$GROUP), col=palette.pies.adj)
```



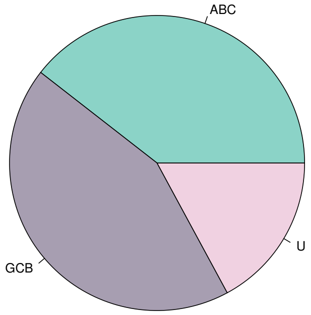
Distribution of samples with different cells of origin subtypes.

```
metadata %>%
    select(Prediction) %>%
    ggplot(aes(x = Prediction)) +
    geom_histogram(stat = "count") +
    labs(y = "Number of Cell-of-origin subtypes",
         x = "Patients Cell-of-origin classification") +
    theme_bw() +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))

Warning: Ignoring unknown parameters: binwidth, bins, pad
```



Or as pie chart.

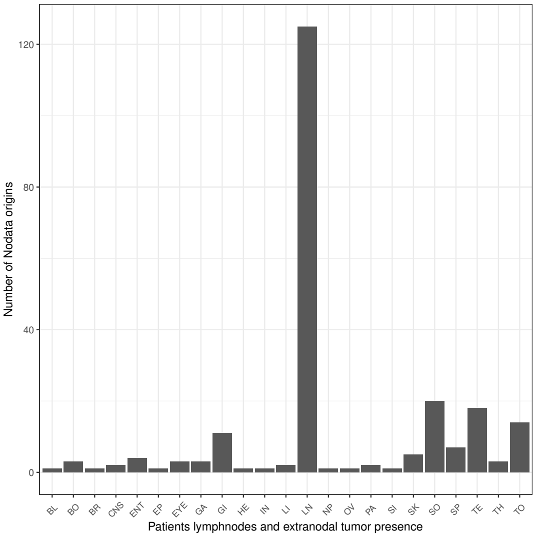
```
palette.pies <- brewer.pal(12, name = "Set3")
palette.pies.adj <- colorRampPalette(palette.pies)(length(unique(metadata$Prediction)))
pie(table(metadata$Prediction), col=palette.pies.adj)
```



Distribution of samples with different lymphnodes and extranodal cancer metastasis.

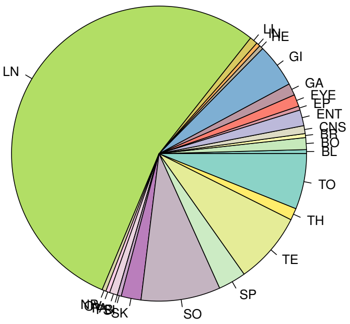
```
par(mfrow=c(2,2))
```

```
metadata %>%
    select(SITE) %>%
    ggplot(aes(x = SITE)) +
    geom_histogram(stat = "count") +
    labs(y = "Number of Nodata origins",
         x = "Patients lymphnodes and extranodal tumor presence") +
    theme_bw() +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))

Warning:  Ignoring unknown parameters:  binwidth, bins, pad
```



Or as a pie chart.

```
palette.pies <- brewer.pal(12, name = "Set3")
palette.pies.adj <- colorRampPalette(palette.pies)(length(unique(metadata$SITE)))
pie(table(metadata$SITE), col=palette.pies.adj)
```



2 Differential expression of microarray Affymetrix data

Genes have been fitted in a model that is based on an Empirical Bayes approach. Ranking of the genes determine if they are statistically significant. Bonferroni correction is used to control the false discovery rate (FDR). Moderated t-statistics, FDR, and fold change (log2) are implemented to reduce selection of false positives.

- **adjpval** is the adjusted P-value to control the FDR using Bonferroni correction. **Genes selected here based on their adjpval are also greater than or equal to the bstat threshold**.

- **avgex** is the average expression the ordinary arithmetic average of the log2-expression values for the probe, across all arrays. **Genes selected here based on their avgex are also greater than or equal to the bstat threshold**.

- **bstat** is the moderated t-statistics using an Empirical Bayes approach generating B-statistics scores.

```
expression <- read.table("data/summary.full.409794.txt", sep = "\t", header = T) %>%
    select(Design, Model, Bthreshold, adjPval, Category, Parameter, Transcripts) %>%
    filter(Category == "total")
summary(expression)

            Design                             Model
 CNSvsNOREL    : 12   systemicRelapse             : 36
 CNSvsNOREL_ABC: 12   systemicRelapseCOOprediction:108
 CNSvsNOREL_EN : 12   systemicRelapseNodes        :108
 CNSvsNOREL_GCB: 12
 CNSvsNOREL_LN : 12
 CNSvsSYST     : 12
 (Other)       :180
   Bthreshold        adjPval      Category     Parameter
 Min.   :-4.00   Min.   :0.1   total:252   adjpval:84
 1st Qu.:-2.50   1st Qu.:0.1               avgex  :84
 Median :-1.00   Median :0.1               bval   :84
 Mean   :-1.25   Mean   :0.1
 3rd Qu.: 0.25   3rd Qu.:0.1
 Max.   : 1.00   Max.   :0.1

  Transcripts
 Min.   :   0
 1st Qu.:   0
 Median :   6
 Mean   : 398
 3rd Qu.: 120
 Max.   :6414
```

Number of transcripts when comparing B-statistics scores, which represent confidence in selecting each significantly expressed gene.

```
aggregate( Transcripts ~ Bthreshold, data=expression, FUN=range)

  Bthreshold Transcripts.1 Transcripts.2
1         -4             0          6414
2         -2             0           954
3          0             0           154
4          1             0            65
```

Number of transcripts when samples are classed into groups, which are based on clinical data (e.g., cell-of-origin, CNS relapse, and nodal/extranodal tumor transmission).

```
aggregate( Transcripts ~ Model, data=expression, FUN=range)

                         Model Transcripts.1 Transcripts.2
1              systemicRelapse             0          6414
2 systemicRelapseCOOprediction             0          5664
3         systemicRelapseNodes             0          4059
```

Number of transcripts found when comparing different sample cases indexed based on their clinical data.

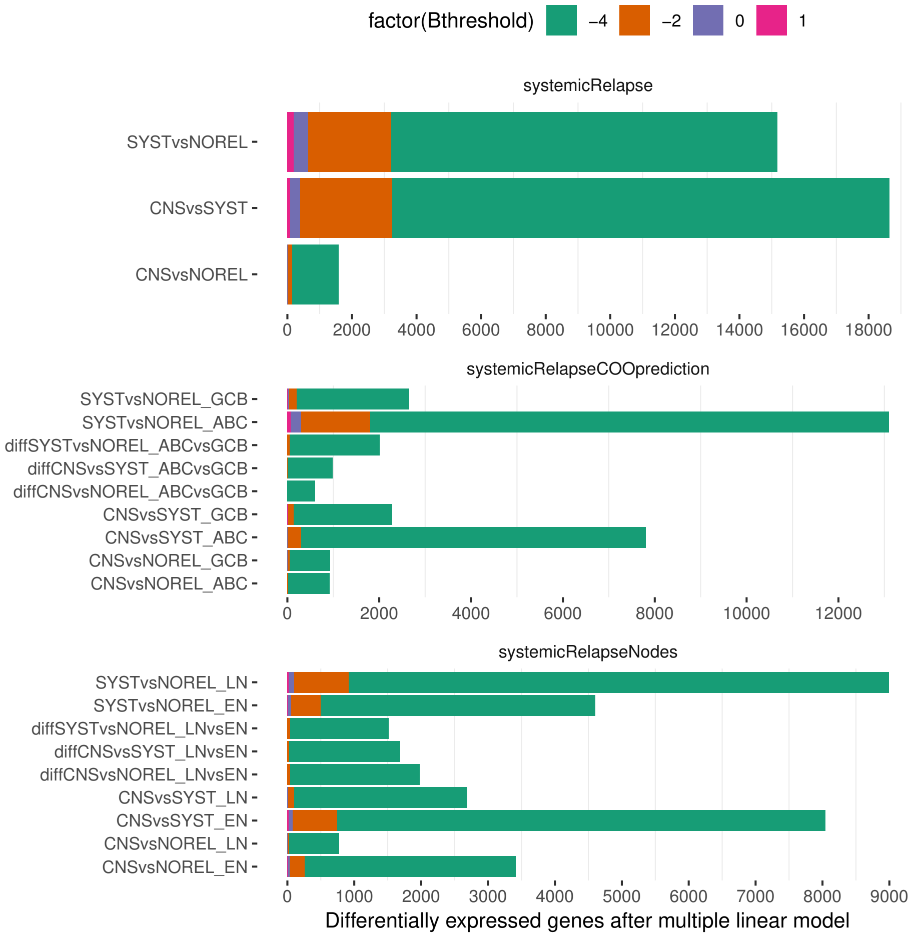
```
aggregate( Transcripts ~ Design, data=expression, FUN=range)
```

```
                      Design Transcripts.1 Transcripts.2
1                 CNSvsNOREL             0           716
2             CNSvsNOREL_ABC             0           453
3              CNSvsNOREL_EN             0          1585
4             CNSvsNOREL_GCB             0           438
5              CNSvsNOREL_LN             0           374
6                  CNSvsSYST            30          6414
7              CNSvsSYST_ABC             0          3770
8               CNSvsSYST_EN             1          3663
9              CNSvsSYST_GCB             2          1081
10              CNSvsSYST_LN             0          1302
11  diffCNSvsNOREL_ABCvsGCB             0           308
12    diffCNSvsNOREL_LNvsEN             0           977
13   diffCNSvsSYST_ABCvsGCB             0           498
14     diffCNSvsSYST_LNvsEN             0           831
15 diffSYSTvsNOREL_ABCvsGCB             0           982
16   diffSYSTvsNOREL_LNvsEN             0           738
17               SYSTvsNOREL            65          5299
18           SYSTvsNOREL_ABC            26          5664
19            SYSTvsNOREL_EN             1          2056
20           SYSTvsNOREL_GCB             0          1227
21            SYSTvsNOREL_LN             0          4059
```

Number of genes that respond to treatment, cell subtypes, and nodal transmission.

↰ Min adjusted p-value of 0.1

```
expression %>%
    ggplot(aes(
        x = Design,
        y = Transcripts,
        fill = factor(Bthreshold))) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "stack") +
    coord_flip() +
    facet_wrap( ~ Model,
              ncol = 1,
               scales = "free") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "",
         y = "Differentially expressed genes after multiple linear model") +
    theme(legend.position = "top",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank")) +
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10))
```



2.1 Cleaning and removing non-essential genes

Subsetting the data by reducing the number of gene profiles improves interpretation and reduces noise. It is well established that many machine learning models used for classification can be sensitive to high number of *irrelevant* genes, others like support vector machines and random forests are less so (Statnikov 2008).

↰ σ^2 is the average of the squared differences from the μ

Each array contains probes of 75,523 functional and non-functional RNAs. Either ncRNA, mRNA, and non annotated genes. More than 53.32% of the probes are non-coding. For interpretation purpose, ncRNAs profiles were discarded before fitting the expressions. In addition, the variation from the mean of each transcript was assessed and the spread of expression were all used to discard top and bottom variants. Individual genes that vary widely from the mean of the array were removed thus reducing the spread of the expression across profiles. Transcripts with potential biased high expressions were thus flagged and discarded thus improving correlation of other transcripts. Subsetting was done after normalization of all datasets, all arrays. This would reduce technical errors appearing significant when comparing arrays between each others. Data was transformed (standardization protocol) before calculating means and variances. This helps a better signal recovery from a large dataset with potential expression bias.

↰ Each array correspond to a DLBCL case

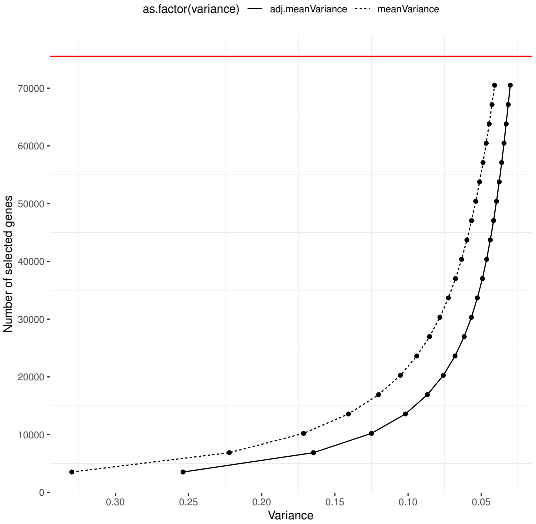
2.1.1 Variance optimization for each array

Full probe list accounting for 75,523 genes (red horizontal line). The full line represents the variance after being adjusted by iteratively discarding top/low variant expression profiles. The dotted line represent the original variance before discarding genes.

The graph below shows that by discarding highly variant expressions and selecting only the top 1613 genes for example, the mean variance of the whole array (0.27) is higher than a ranked subset of 10,811 (0.09). Ideally, the reduction of the data is on both, the mean variance and mean standard deviation of the whole array.

↰ The smaller the variance, the better

```
read.table("./data/summary.139102.adjusted.means.subsetting.txt", header = T) %>%
  select(dimension, meanVariance, adj.meanVariance) %>%
  gather("variance", "count", 2:3) %>%
  ggplot(aes(x = count,
             y = dimension)) +
  theme_bw() +
  geom_line(aes(linetype = as.factor(variance))) +
  geom_point() +
  scale_x_continuous(trans = "reverse",
                     breaks = scales::pretty_breaks(n = 10)) +
  scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
  geom_hline(aes(yintercept = 75523), colour = "red") +
  labs(y = "Number of selected genes",
       x = "Variance") +
  theme(legend.position = "top",
        strip.background = element_rect(linetype = "blank",
                                        fill = "white"),
        panel.border = element_rect(linetype = "blank",
                                    fill = NA),
        panel.grid.major = element_line(linetype = "blank"))
```

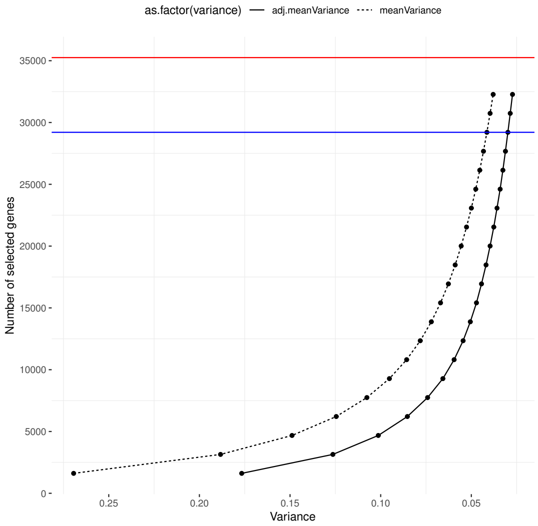


Same plot description as above however we removed ncRNA which account for 53.32% of the probes. The total number of transcripts is now 35,253 (46%, red horizontal line). The blue horizontal line represents the threshold that was selected for subsequent analysis.

↰ 29,207 genes were selected for clustering and nets

By discarding 1198 transcripts from the 35,253 the top outliers with high variance are not included in the clustering process. More rare expression signals will get distinguished. Also, the size of the dataset was reduced to 29,207 by removing transcripts with little deviation from the mean of each array. The total number of transcripts by array was kept above 25k to increase the sizes of the clusters (modules and networks) in later analyses. For example, network analysis on 20k transcripts generated network sizes between 200 and 500. At 29k networks have a total size over 700 nodes.

```
read.table("./data/summary.149317.adjusted.means.subsetting.txt", header = T) %>%
    select(dimension, meanVariance, adj.meanVariance) %>%
    gather("variance", "count", 2:3) %>%
    ggplot(aes(x = count,
               y = dimension)) +
    theme_bw() +
    geom_line(aes(linetype = as.factor(variance))) +
    geom_point() +
    scale_x_continuous(trans = "reverse",
                       breaks = scales::pretty_breaks(n = 8)) +
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
    geom_hline(aes(yintercept = 35253), color = "red") +
    geom_hline(aes(yintercept = 29207), color = "blue") +
    labs(y = "Number of selected genes",
         x = "Variance") +
    theme(legend.position = "top",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank"))
```



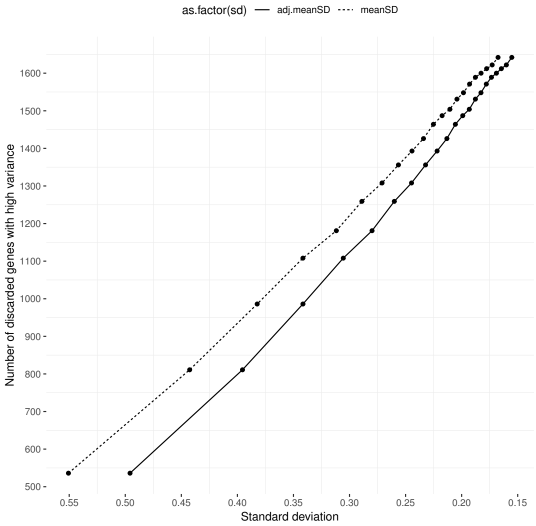
2.1.2 Standard deviation optimization for each array

The spread of the gene expression scores is dependent on their variance, their deviation from each array's mean (population mean). By removing potentially noisy expressions we are reducing the spread of the arrays numbers, hence improving recognition of rare gene regulations. Below, the plot shows how the standard deviation, **spread** of the data is getting smaller the more we discard genes with high and low variance.

↰ Best if small spread between 2 SDs

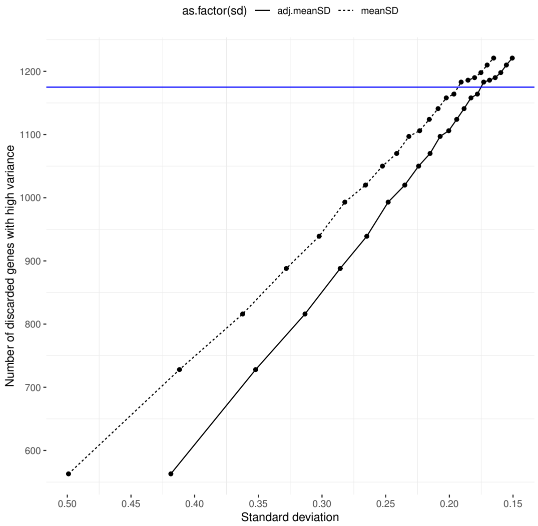
All array probes with all RNAs.

```
read.table("./data/summary.139102.adjusted.means.subsetting.txt", header = T) %>%
    select(discarded, meanSD, adj.meanSD) %>%
    gather("sd", "count", 2:3) %>%
    ggplot(aes(x = count,
               y = discarded)) +
    theme_bw() +
    geom_line(aes(linetype = as.factor(sd))) +
    geom_point() +
    scale_x_continuous(trans = "reverse",
                       breaks = scales::pretty_breaks(n = 8)) +
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
    labs(y = "Number of discarded genes with high variance",
         x = "Standard deviation") +
    theme(legend.position = "top",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank"))
```



Without the ncRNAs. Blue horizontal line is the number of genes discarded when variance shrinking was used to eliminate biased genes.

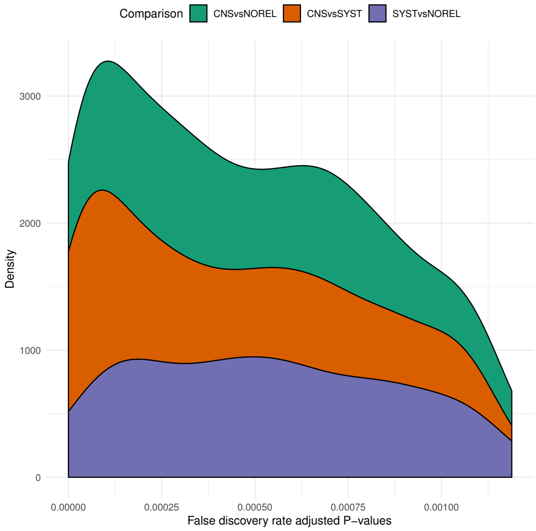
```
read.table("./data/summary.149317.adjusted.means.subsetting.txt", header = T) %>%
    select(discarded, meanSD, adj.meanSD) %>%
    gather("sd", "count", 2:3) %>%
    ggplot(aes(x = count,
               y = discarded)) +
    theme_bw() +
    geom_line(aes(linetype = as.factor(sd))) +
    geom_point() +
    geom_hline(aes(yintercept = 1175), colour = "blue") +
    scale_x_continuous(trans = "reverse",
                       breaks = scales::pretty_breaks(n = 8)) +
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
    labs(y = "Number of discarded genes with high variance",
         x = "Standard deviation") +
    theme(legend.position = "top",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank"))
```



2.2 Distribution of p-values in pairwise comparisons

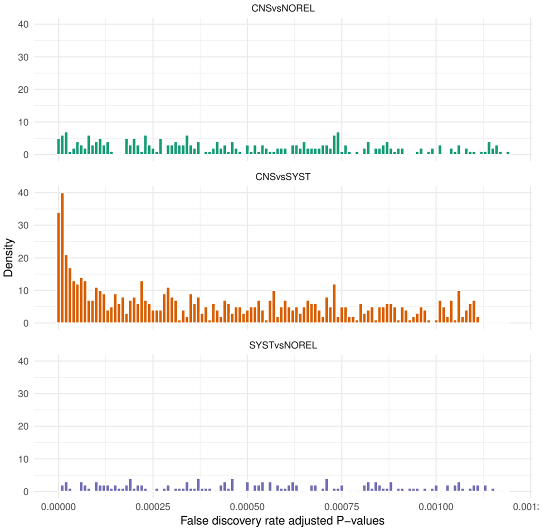
Distribution of p-values in pairwise comparisons between individuals recognized as CNS relapse, systemic, and did not relapse.

```
pvals <- read.table("./data/summary.lmfit.fdrAdjpval.txt", header = TRUE, fill = TRUE)
pvals %>%
    filter(Contrast == "systemicRelapse") %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_density(position = "stack") +
    theme_minimal() +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")
```



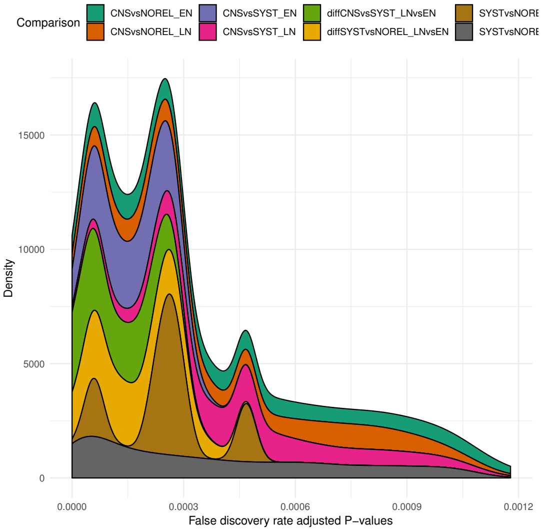
Distribution of p-values in pairwise comparisons between individuals recognized as CNS relapse, systemic, and did not relapse.

```
pvals %>%
    filter(Contrast == "systemicRelapse") %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_histogram(binwidth = 10^-5,
                   col=I("white")) +
    theme_minimal() +
    facet_wrap( ~ Comparison,
               ncol = 1) +
    theme(legend.position = "none") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")
```



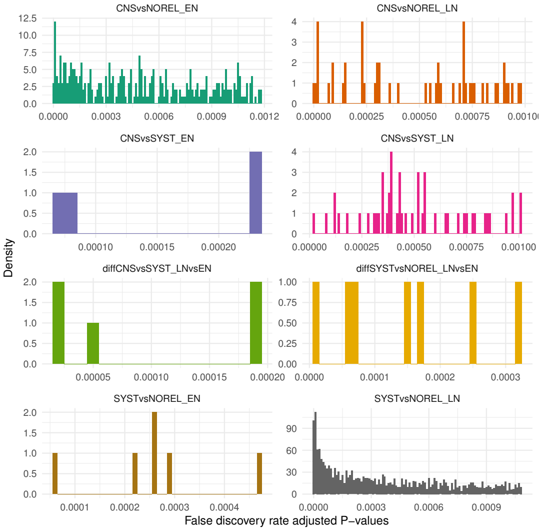
Distribution of p-values in pairwise comparisons between individuals recognized with lymph nodes and extranodal implication.

```
pvals %>%
    filter(Contrast == c("systemicRelapseNodes")) %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_density(position = "stack") +
    theme_minimal() +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")
```



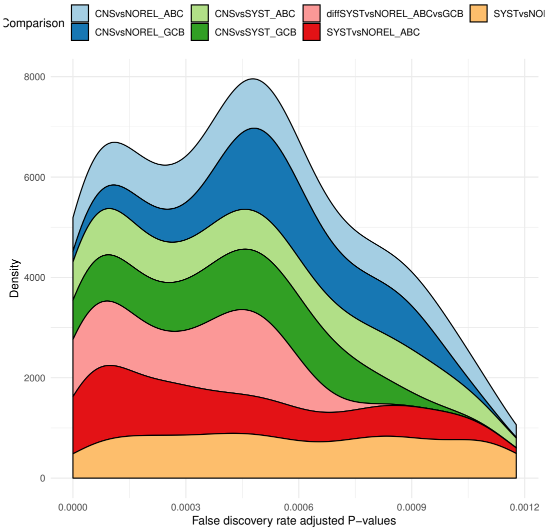
Distribution of p-values in pairwise comparisons between individuals recognized with lymph nodes and extranodal implication.

```
pvals %>%
    filter(Contrast == "systemicRelapseNodes") %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_histogram(binwidth = 10^-5) +
    theme_minimal() +
    facet_wrap( ~ Comparison,
               ncol = 2, scales = "free") +
    theme(legend.position = "none") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")
```



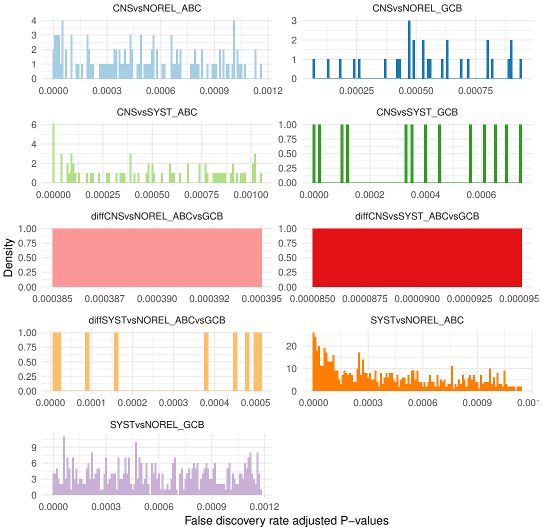
Distribution of p-values in pairwise comparisons between individuals recognized with cell-of-origin based on ABC or GCB.

```
pvals %>%
    filter(Contrast == c("systemicRelapseCOOprediction")) %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_density(position = "stack") +
    theme_minimal() +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Paired") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")

Warning:  Groups with fewer than two data points have been dropped.
Warning:  Groups with fewer than two data points have been dropped.
Warning:  Removed 2 rows containing missing values (position_stack).
```



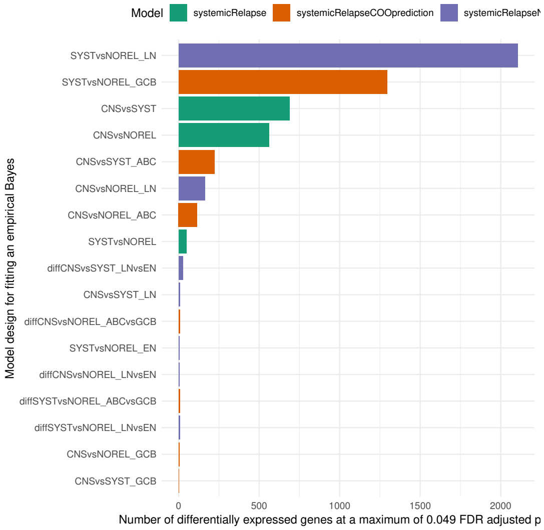
Distribution of p-values in pairwise comparisons between individuals recognized with cell-of-origin based on ABC or GCB.

```
pvals %>%
    filter(Contrast == "systemicRelapseCOOprediction") %>%
    ggplot(aes(x = Pval,
               fill = Comparison)) +
    geom_histogram(binwidth = 10^-5) +
    theme_minimal() +
    facet_wrap( ~ Comparison,
               ncol = 2, scales = "free") +
    theme(legend.position = "none") +
    scale_fill_brewer(palette = "Paired") +
    labs(x = "False discovery rate adjusted P-values",
         y = "Density")
```



Across all contrasts class comparison, the number of genes significantly expressed at FDR multi test corrections.

```
read.table("./data/summary.lmfit.bval.txt", header = TRUE, fill = TRUE) %>%
  filter(Model == c("systemicRelapse","systemicRelapseNodes","systemicRelapseCOOprediction")) %>%
  ggplot(aes(x = reorder(Design, Transcripts),
             y = Transcripts,
             fill = Model)) +
  geom_bar(stat = "identity",
           position = "dodge") +
  coord_flip() +
  theme_minimal() +
  theme(legend.position = "top") +
  scale_fill_brewer(palette = "Dark2") +
  labs(y = "Number of differentially expressed genes at a maximum of 0.049 FDR adjusted p-value",
       x = "Model design for fitting an empirical Bayes")
```



3 Clustering and network analyses

The number of clusters and modules per networks are assigned by designing first a similarity matrix between differentially expressed gene for any two conditions (eg., relapse vs no relapse individual cases). An adjacency matrix is then constructed by weighting the previously inferred measures. The data is transformed to increase the correlation coefficient therefore improving detection of strong correlated patterns. (Example of the strength of data transformation and correlation, visit the following online page).

The final methods selected were best according to our output. Convergence and reproducibility of each trend in gene number per module or scale of module per network helped narrow down our selection criteria for which method to incorporate in our clustering technique. The best configuration involve a Pearson correlation at stringent thresholds. Hellinger expression data standardization was better than other methods in helping to estimate similarity between genes. The correlation power was set to 6 at an overall threshold of recall of 0.5 and a complete clustering distribution to improve estimating adjacency distribution.

↰Overfitting is a source of bias.

- **MaxEdgesPerGene**, maximum number of correlations per genes
- **NbNodes**, number of genes found for each edge connection bracket
- **Normalization**, method that focuses on creating complete clusters. We tested methods ranging from Complete clustering, Average, and Ward. Each method is detailed here. Only Complete clustering was retained. All other methods overfitted the data.
- **Correlation**, finding ranges from linear to non-linear trends. We tested Pearson and Spearman correlation.
- **Standardization**, data transformation method. We tested transformation by Hellinger, Standardize, Range, and Logarithmic scaling. Each method is detailed here.
- **MaxGenePerModule**, how many genes assigned by cluster (module)
- **SimilaritySize**, number of initial differentially expressed genes
- **EdgeThreshold**, parameter to limit the weight of the edges

- **CorrelationPower**, power transformation of the data

↰Effect of correlation methods is seen on module content

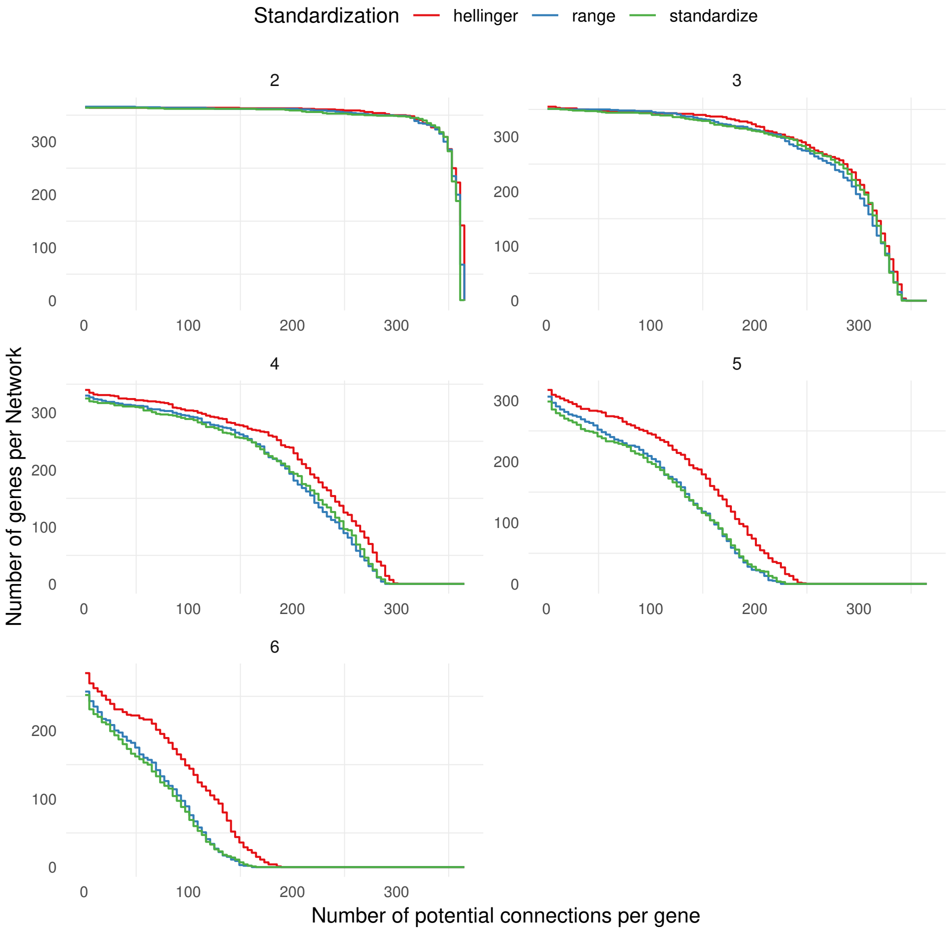
```
ns <- read.table("./data/networks.summary.437006.txt", header = T)
summary(ns)
```

```
 MaxEdgesPerGene    NbNodes      Normalization    Correlation
 Min.   :  1     Min.   :  0   complete:4140   pearson:4140
 1st Qu.: 92     1st Qu.:  4
 Median :183     Median :234
 Mean   :183     Mean   :194
 3rd Qu.:274     3rd Qu.:338
 Max.   :365     Max.   :366
   Standardization MaxGenesPerModule SimilaritySize EdgeThreshold
 hellinger  :1380   Min.   : 5.0      Min.   :366    Min.   :0.5
 range      :1380   1st Qu.: 5.0      1st Qu.:366    1st Qu.:0.5
 standardize:1380   Median :20.0      Median :366    Median :0.5
                    Mean   :19.3      Mean   :366    Mean   :0.5
                    3rd Qu.:33.0      3rd Qu.:366    3rd Qu.:0.5
                    Max.   :33.0      Max.   :366    Max.   :0.5
 CorrelationPower
 Min.   :2
 1st Qu.:3
 Median :4
 Mean   :4
 3rd Qu.:5
 Max.   :6
```

Difference between methods used for network inference. Are we able to generate convergence of the output of all iterations across all methods? After many trials the below two graphs (in this section) represent the number of networks, genes, clustered, and modules for the selected configuration (between normalization, clustering, and data transformation).

↰ Definitive graphs

```
networks.comparison <- function(data = ns){
    data %>%
    ggplot(aes(
        x = MaxEdgesPerGene,
        y = NbNodes,
        fill = Standardization)) +
    theme_minimal() +
    geom_step(aes(color = Standardization),
              stat = "identity") +
    facet_wrap( ~ CorrelationPower,
               ncol = 2,
               scales = "free") +
    scale_color_brewer(type = "qual", palette = 6) +
    labs(x = "Number of potential connections per gene",
         y = "Number of genes per Network") +
    theme(legend.position = "top",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank"),
          axis.text.x = element_text(size = 8),
          axis.text.y = element_text(size = 8))
}

networks.comparison(data = ns)
```

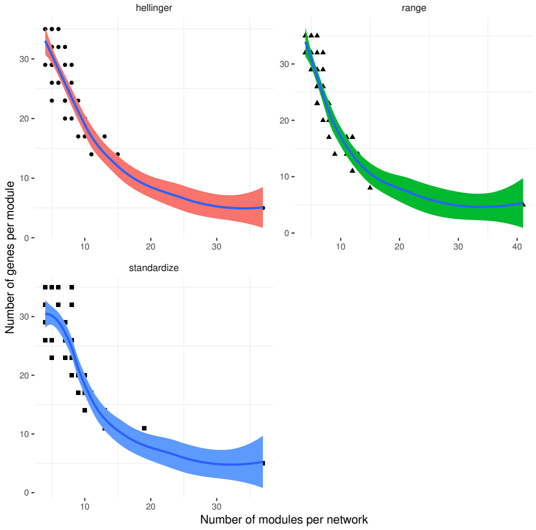


Showing the number of modules per network and the number of genes per module. Each module contains differing number of nodes based on their correlation strength. Each cluster contains at least one module. Each network contains at least one cluster. One module can be assigned to nodes that belong to more than one cluster. The Lowess curves show if the trend in the data is linear or not. The wave around Lowess curves represents the level of confidence of the data points (the narrower the interval the better, less variability = more accuracy).

↰Points=iterations. With less iterations comes high variability of the curve

```
ms <- read.table("./data/modules.summary.437006.txt", header = TRUE)
```

```
modules.comparison <- function(data = ms){
data %>%
    ggplot(aes(
        x = NbModules,
        y = MaxGenesPerModule,
        fill = Standardization)) +
    theme_bw() +
    geom_point(aes(shape = Standardization)) +
    scale_color_brewer(type = "qual", palette = 6) +
    labs(x = "Number of modules per network",
         y = "Number of genes per module") +
    facet_wrap( ~ Standardization,
                ncol = 2,
                scales = "free") +
    theme(legend.position = "none",
          strip.background = element_rect(linetype = "blank",
                                          fill = "white"),
          panel.border = element_rect(linetype = "blank",
                                      fill = NA),
          panel.grid.major = element_line(linetype = "blank"),
          axis.text.x = element_text(size = 8),
          axis.text.y = element_text(size = 8)) +
    geom_smooth(method = 'loess', alpha = 1)
}

modules.comparison(data = ms)
```



3.1 Network analysis for Spearman-related correlations (relaxed)

Thresholds based on the Empirical Bayes approach to rank genes and determine if a gene is significantly expressed by shrinking the sample variance Smyth 2004. Limma implementation.

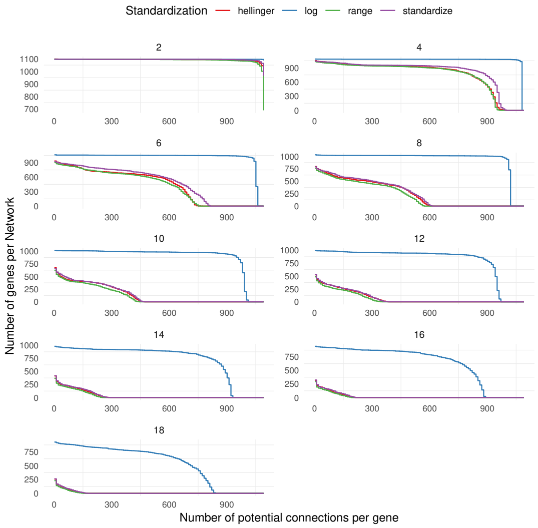
↰ Different iterations to test which normalization & clustering are best

- **Average Expression**: 5
- **Adjusted P-value**: equal or less than 0.045
- **Log Fold Change**: 1
- **B-statisitcs**: 1.5

3.1.1 Nodal versus extra-nodal lymphoma

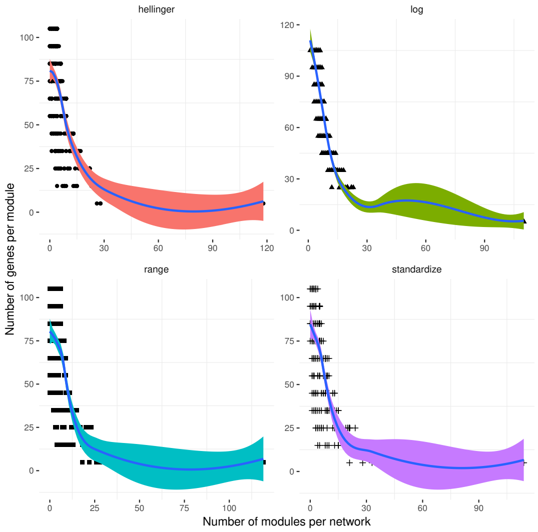
Genetic networks from differentially expressed genes selected by comparing sample cases with nodal and extranodal lymphoma.

```
ns <- read.table("./data/networks.summary.104859.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

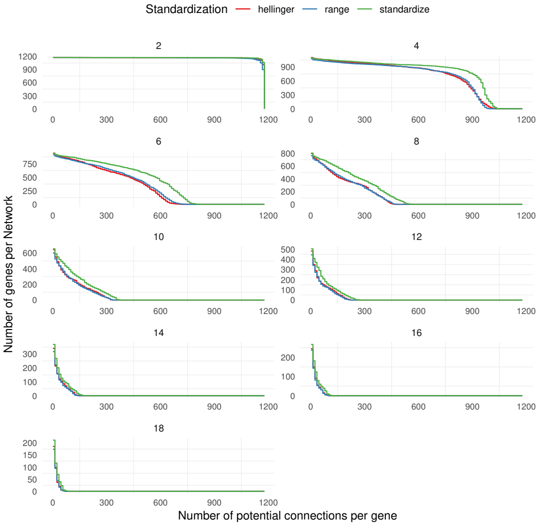
```r
ms <- read.table("./data/modules.summary.104859.txt", header = TRUE)
modules.comparison(ms)
```



3.1.2 Relapsed versus no CNS relapsed cases

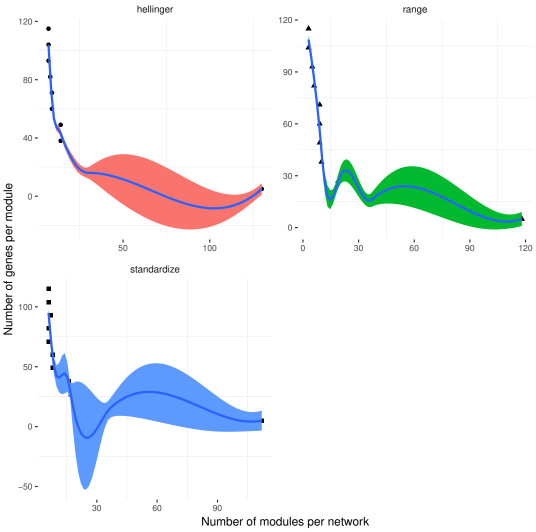
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```r
ns <- read.table("./data/networks.summary.114018.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

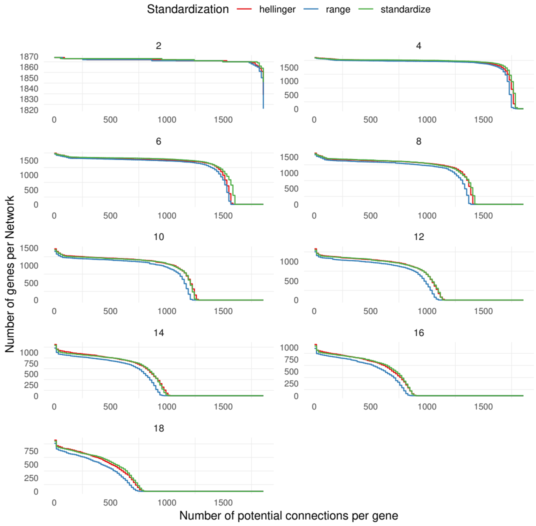
```r
ms <- read.table("./data/modules.summary.114018.txt", header = TRUE)
modules.comparison(ms)
```



3.1.3 Lymphoma cases classified by Cell-of-origin subtypes

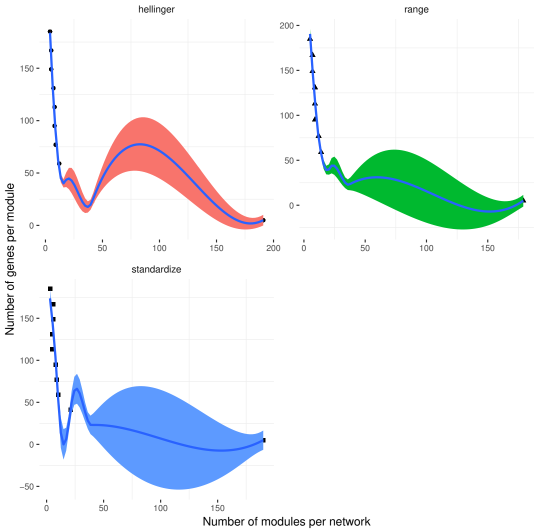
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```r
ns <- read.table("./data/networks.summary.114017.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

```
ms <- read.table("./data/modules.summary.114017.txt", header = TRUE)
modules.comparison(ms)
```



3.2 Network analysis for Pearson-related correlations (relaxed)

Thresholds based on the Empirical Bayes approach to rank genes and determine if a gene is significantly expressed. Limma implementation.

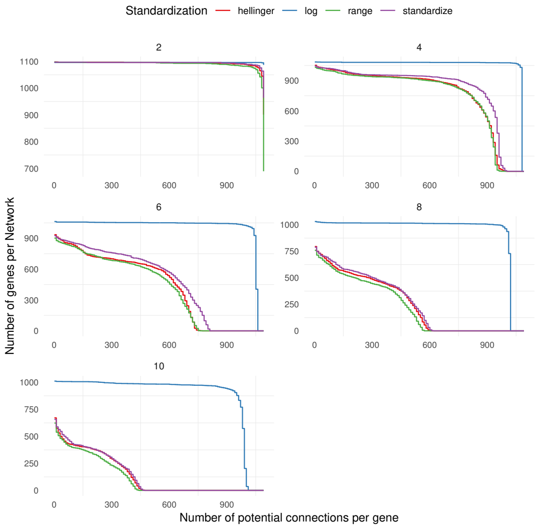
↰With pearson, we can only raise the data to power 10. All are discarded after 10.

- **Average Expression**: 5
- **Adjusted P-value**: equal or less than 0.045
- **Log Fold Change**: 1
- **B-statisitcs**: 1.5

3.2.1 Nodal versus extra-nodal lymphoma

Genetic networks from differentially expressed genes selected by comparing sample cases with nodal and extranodal lymphoma.

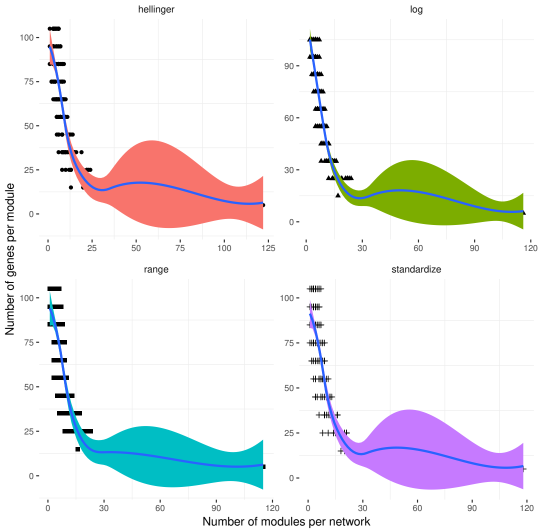
```
ns <- read.table("./data/networks.summary.104862.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

↰Since Lowess ranks by confidence, Log transformation seems the best, ie, low variability. For this, Log is removed from further tests.

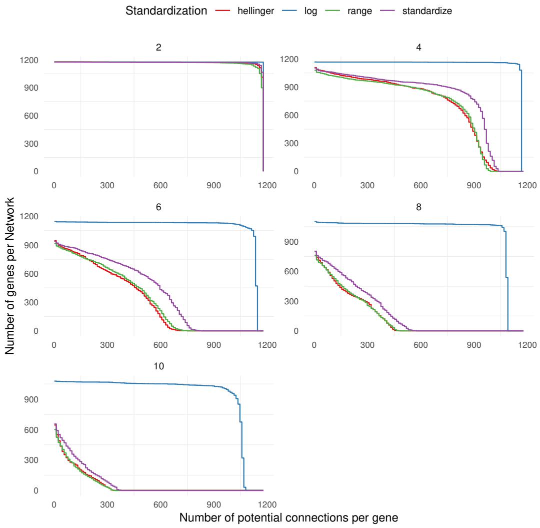
```r
ms <- read.table("./data/modules.summary.104862.txt", header = TRUE)
modules.comparison(ms)
```



3.2.2 Relapsed versus no CNS relapsed cases

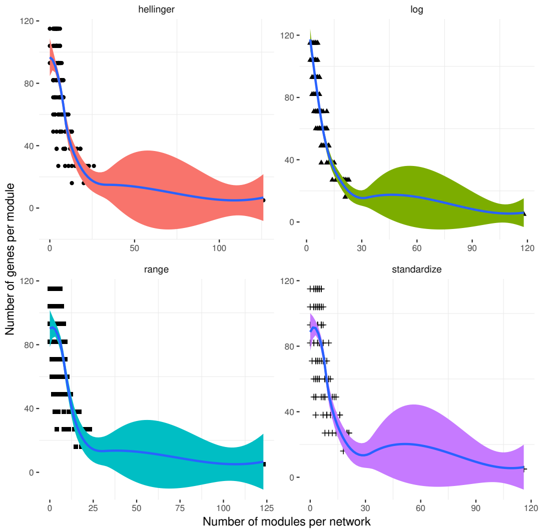
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```r
ns <- read.table("./data/networks.summary.104863.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

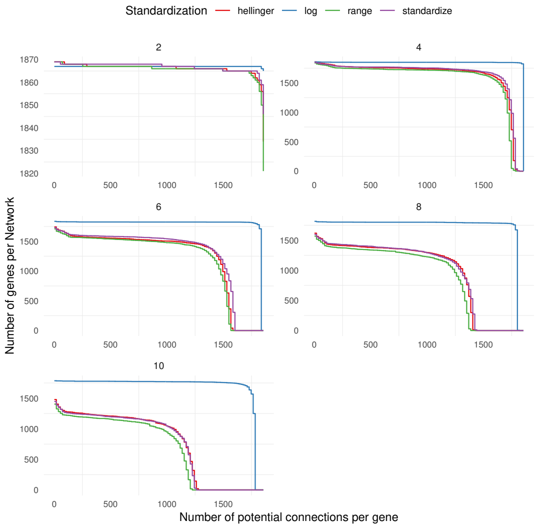
```r
ms <- read.table("./data/modules.summary.104863.txt", header = TRUE)
modules.comparison(ms)
```



3.2.3 Lymphoma cases classified by Cell-of-origin subtypes

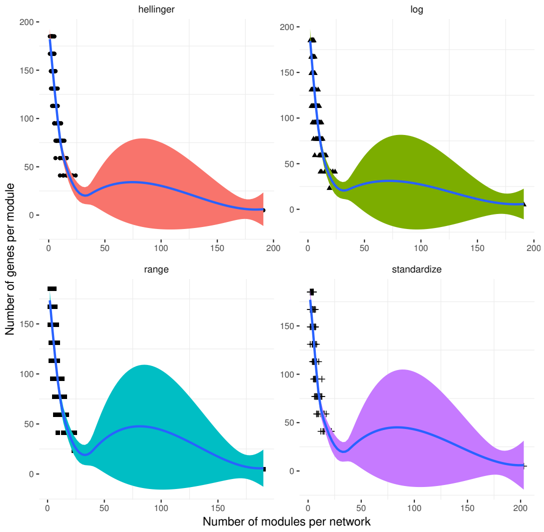
Genetic networks from differentially expressed genes selected by comparing sample cases with cell of origin classification based on ABC or GCB subtypes.

```r
ns <- read.table("./data/networks.summary.104864.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

```
ms <- read.table("./data/modules.summary.104864.txt", header = TRUE)
modules.comparison(ms)
```



3.3 Network analysis for Spearman-related correlations (stringent)

Thresholds based on the Empirical Bayes approach to rank genes and determine if a gene is significantly expressed. Limma implementation.

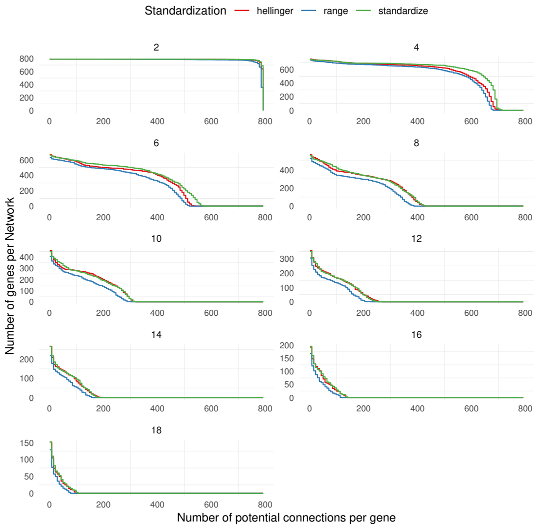
↰Same analysis with more stringent parameters

- **Average Expression**: 10
- **Adjusted P-value**: equal or less than 0.030
- **Log Fold Change**: 1
- **B-statisitcs**: 2

3.3.1 Nodal versus extra-nodal lymphoma

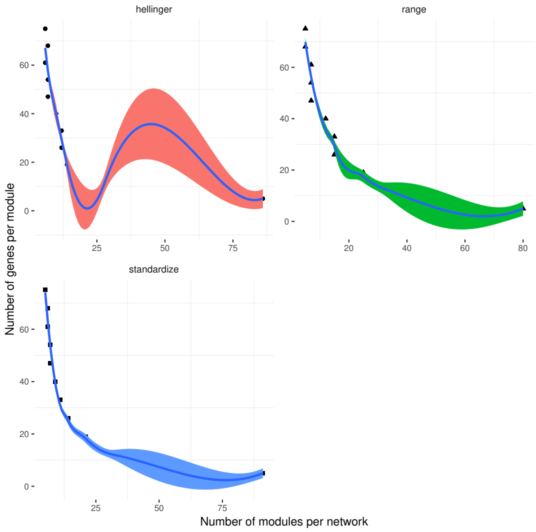
Genetic networks from differentially expressed genes selected by comparing sample cases with nodal and extranodal lymphoma.

```
ns <- read.table("./data/networks.summary.119759.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

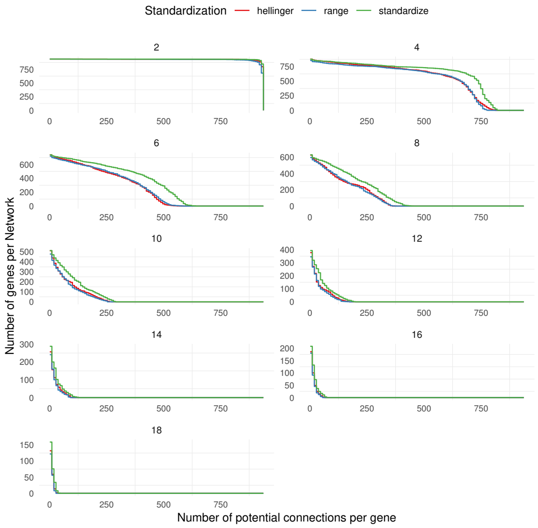
```r
ms <- read.table("./data/modules.summary.119759.txt", header = TRUE)
modules.comparison(ms)
```



3.3.2 Relapsed versus no CNS relapsed cases

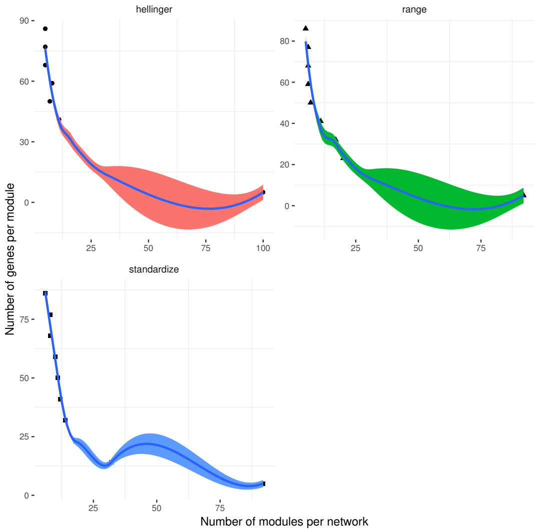
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```r
ns <- read.table("./data/networks.summary.119760.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

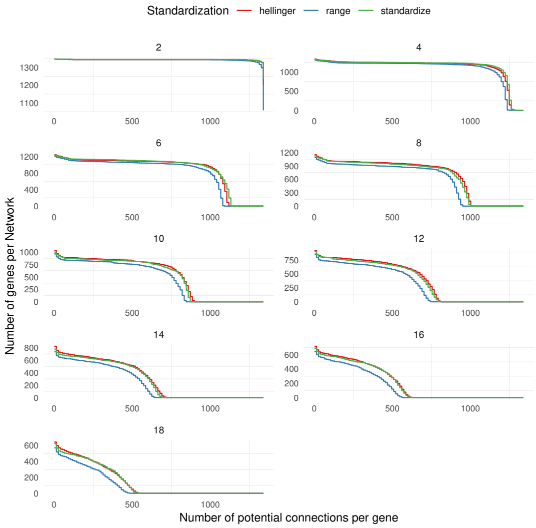
```r
ms <- read.table("./data/modules.summary.119760.txt", header = TRUE)
modules.comparison(ms)
```



3.3.3 Lymphoma cases classified by Cell-of-origin subtypes

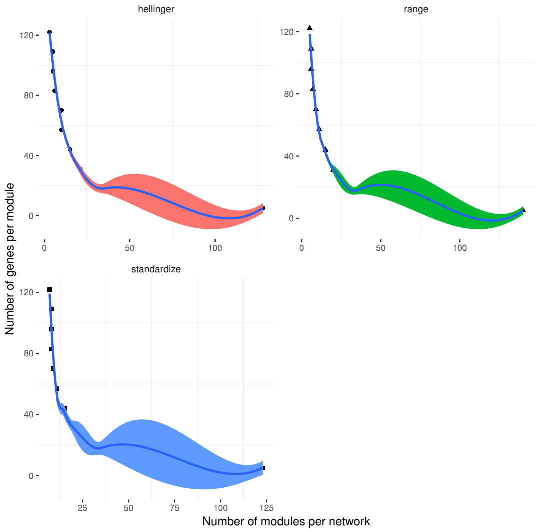
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```r
ns <- read.table("./data/networks.summary.119758.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

```
ms <- read.table("./data/modules.summary.119758.txt", header = TRUE)
modules.comparison(ms)
```



3.4 Network analysis for Pearson-related correlations (stringent)

Thresholds based on the Empirical Bayes approach to rank genes and determine if a gene is significantly expressed. Limma implementation.

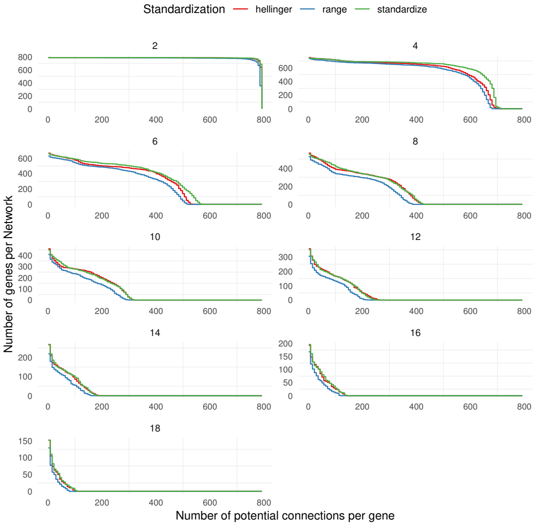
↰Same analysis with more stringent parameters

- **Average Expression**: 10
- **Adjusted P-value**: equal or less than 0.030
- **Log Fold Change**: 1
- **B-statisitcs**: 2

3.4.1 Nodal versus extra-nodal lymphoma

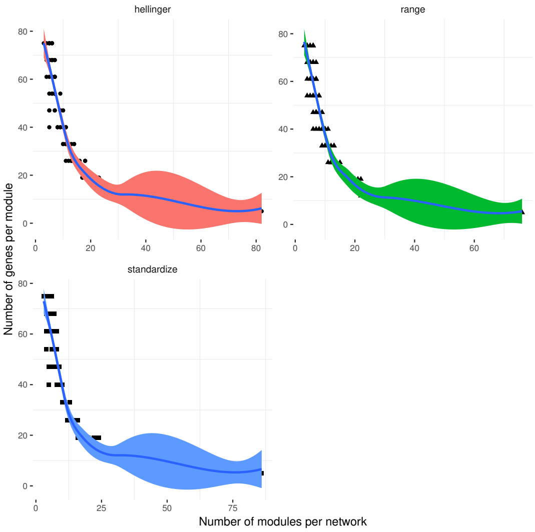
Genetic networks from differentially expressed genes selected by comparing sample cases with nodal and extranodal lymphoma.

```
ns <- read.table("./data/networks.summary.119755.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

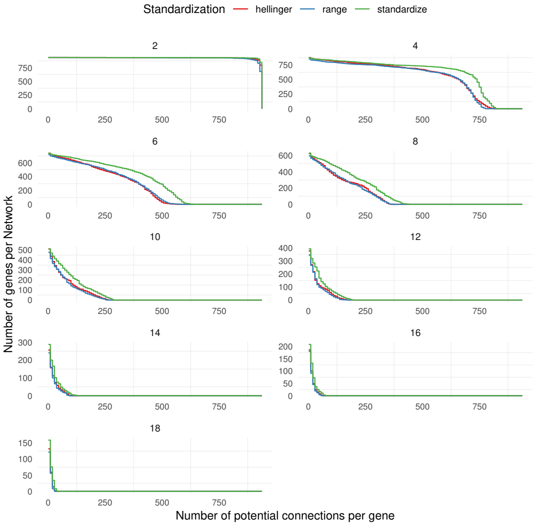
```
ms <- read.table("./data/modules.summary.119755.txt", header = TRUE)
modules.comparison(ms)
```



3.4.2 Relapsed versus no CNS relapsed cases

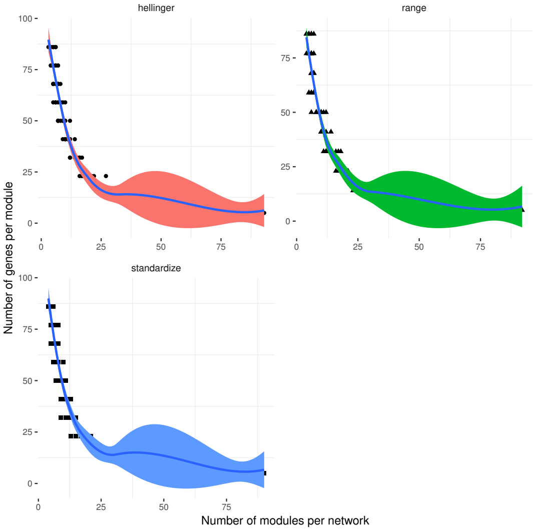
Genetic networks from differentially expressed genes selected by comparing sample cases with systemic or no CNS relapse lymphoma.

```
ns <- read.table("./data/networks.summary.119754.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

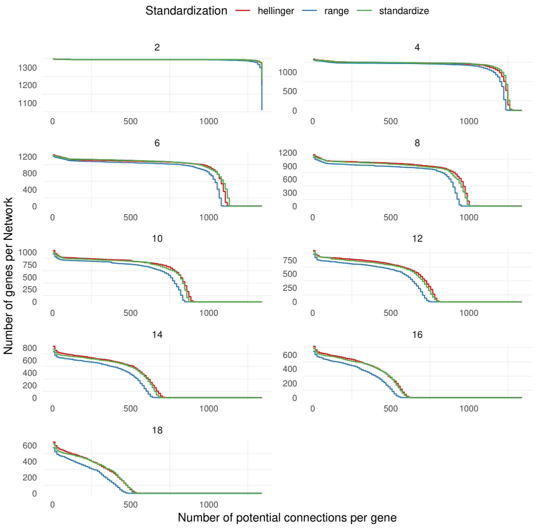
```r
ms <- read.table("./data/modules.summary.119754.txt", header = TRUE)
modules.comparison(ms)
```



3.4.3 Lymphoma cases classified by Cell-of-origin subtypes

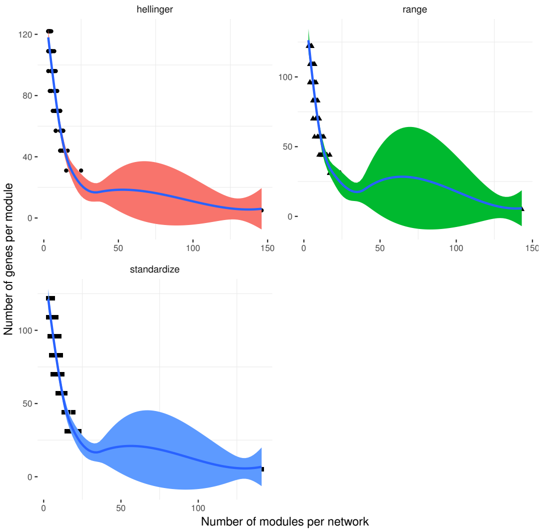
Genetic networks from differentially expressed genes selected by comparing sample cases with cell of origin classification based on ABC or GCB subtypes.

```r
ns <- read.table("./data/networks.summary.119757.txt", header = TRUE)
networks.comparison(ns)
```



Showing the number of modules per network and the number of genes per module.

```
ms <- read.table("./data/modules.summary.119757.txt", header = TRUE)
modules.comparison(ms)
```



4 Machine Learning

Machine learning models were used for classification of cases into systemic relapse of DLBCL, CNS relapse or no relapse. Data are gene expression from Affymetrix arrays of 240 individuals with a form of DLBCL. Subsets of the whole number of microarray probes will be used for classification.

4.1 Regularization

Least absolute shrinkage and selection operator (LASSO) was used for dimension reduction. Gene expression profiles were extracted from networks with significant connectivity. Subset selection using lasso, penalizes genes based on coefficient estimates, to increase accuracy of classification. Briefly, cases are assigned to either diagnosis category, systemic relapse (SYST), CNS relapse (CNS), and no relapse (NOREL). During each iteration, a prediction is made to assign a category. Then a probability is calculated for having an accuracy performance for that iteration. A single iteration has a different random seed, which generates a different set of lambda coefficients for adjusting the lasso penalty. The best lambda across a grid of coefficients with the best accuracy classification is then selected based on accuracy. Adjusting the lambda score also adjusts the subset of genes used for the classification. For one best lambda there is one subset of significantly expressed genes and each gene has a different probability. For one best lambda there is one mean probability registered for that subset of genes.

↰If a subset has 50 genes, the reported probabilities are the mean of each gene individual probability to predict all individual cases

4.1.1 Uncertainty estimation for selected genes from expression networks

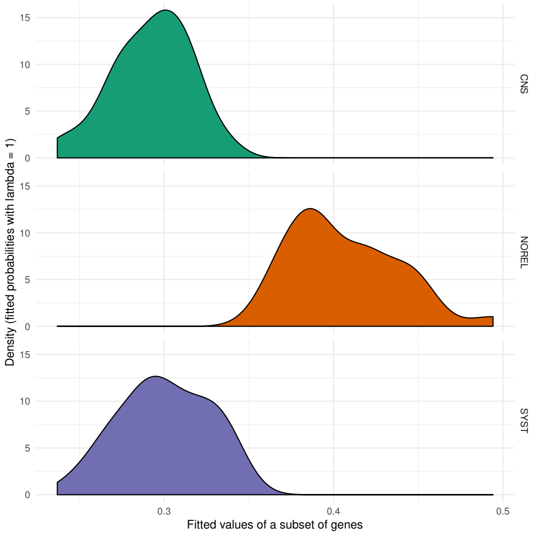
The chart below shows how many iterations (dots) were executed for each sample group before selecting a subset of genes through regularization.

The probabilities are the fitted values of either a multinomial or binomial model at the best lambda (λ), shrinking parameter determined by tuning and cross-validation resampling. When predictions were made with lasso, the least squares were penalized. Lasso zero out coefficient estimates thus reducing the data.

The fitted values are compared to the outcome to follow the proportion of variance "explained" by the model and the proportion of variance "not explained".

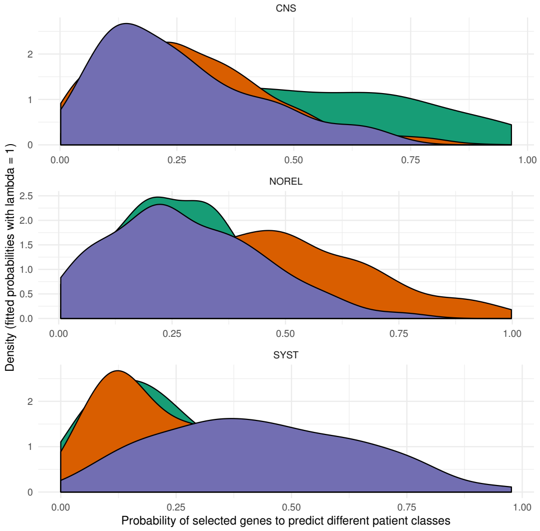
Peaks in density represents variance fitted at best λ between sample groups. Probabilities are compared to the residuals of the data, the outcome is the fitted values. As long as the peaks differ between groups, then the prediction is possible between samples. There is an overlay between CNS and SYST groups, which indicates the presence of some bias in differentiating between them.

```r
read.table("./data/logSummary.lambda.iterations30.multinomial.probabilities.txt",
           row.names = 1, header = T) %>%
    ggplot(aes(x = probabilityScore,
               fill = class)) +
    geom_density() +
    theme_minimal() +
    facet_grid(class ~ .) +
    theme(legend.position = "none") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "Fitted values of a subset of genes",
         y = "Density (fitted probabilities with lambda = 1)")
```



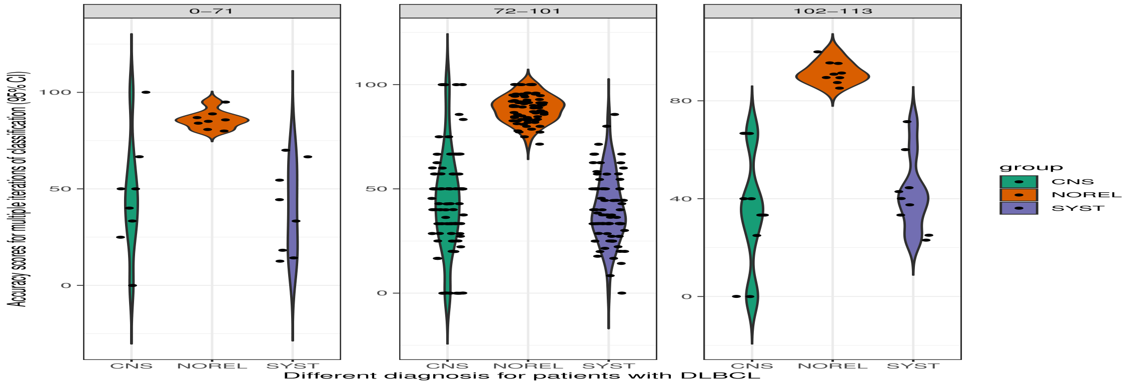
Density plot summarizing the distribution of how many times the CNS/NOREL/SYST sample classes were correctly predicted against other classes using the list of genes that supposedly represent individuals with each type of prognosis.

```r
read.table("./data/logSummary.lambda.iterations30.multinomial.densities.txt",
           row.names = 1, header = T) %>%
    filter(classes == c("CNS", "SYST", "NOREL")) %>%
    ggplot(aes(x = probabilities,
               fill = classes)) +
    geom_density() +
    theme_minimal() +
    facet_wrap( ~ genes,
               ncol = 1,
               scales = "free") +
    theme(legend.position = "none") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Density (fitted probabilities with lambda = 1)")
```



Plot showing the accuracy of assigning an individual to its correct class (or diagnosis) based on lambda calculation for lasso regularization. Each facet represents an accuracy for multiple iterations with a specific number of genes.

```r
df <- read.table("./data/logSummary.lambda.iterations30.multinomial.accuracies.txt",
                 row.names = 1, header = T)
mir <- min(df$regNgenes)
mar <- max(df$regNgenes)
q1 <- floor((mir+mar)/2.5)
q2 <- floor((mir+mar)/1.75)
df$grouped <- cut(df$regNgenes, c(0, q1, q2, mar))
levels(df$grouped) <- c(paste0(0,"-",q1),
                        paste0(q1+1,"-",q2),
                        paste0(q2+1,"-",mar))
df %>%
    ggplot(aes(x = group,
               y = accuracy,
               fill = group)) +
    geom_violin(trim = FALSE) +
    geom_jitter(shape=16, position=position_jitter(0.2)) +
    scale_fill_brewer(palette = "Dark2") +
    theme_bw() +
    labs(x = "Different diagnosis for patients with DLBCL",
         y = "Accuracy scores for multiple iterations of classification (95% CI)") +
##    theme(legend.position = "top") +
    facet_wrap( ~ grouped,
               ncol = 3,
               scales = "free")
```



4.2 Selected machine and deep learning models

Selection of learners was based on historical efficiency of such algorithms. Also, the training of classifiers starts from simple logistic regression, naive bayes, nearest neighbors, going to more flexible models, such as trees and deep neural nets that require more hyperparameter tuning. This strategy lets assess with less bias the overfitting issues that may arise.

Table 1: Machine learning models

#	Classifiers trained	R package*	Parameters† tuned	Abbreviation
1	Naive bayes	naivebayes	laplace, usekernel, adjust	naive_bayes
2	Weighted k-Nearest Neighbors	kknn	kmax, distance, kernel	kknn
3	Penalized multinomial regression	nnet	decay	multinom
4	Random forest	randomForest	mtry	rf
5	Regularized random forest	RRF	mtry, coefReg, coefImp	RRF
6	Linear discriminant analysis (LDA)	MASS	dimen	lda2
7	Multilayer perceptron network by stochastic gradient descent	FCNN4R	size, l2reg, lambda, learn_rate, momentum, gamma, minibatchsz, repeats	mlpSGD
8	Flexible discriminant analysis (FDA)	mda	degree, nprune	fda
9	Bagged FDA	mda	degree, nprune	bagFDA
10	Bagged FDA using gCV pruning	earth	degree	bagFDAGCV
11	Penalized discriminant analysis	mda	lambda	pda
12	Partial least squares	pls	ncomp	kernelpls
13	Support vector machines (SVM) with linear kernel	kernlab	C	svmLinear
14	L2 regularized SVM (dual) with linear kernel	LiblineaR	cost, loss	svmLinear3
15	SVM with polynomial kernel	kernlab	degree, scale, C	svmPoly
16	SVM with radial basis function kernel	kernlab	sigma, C	svmRadialSigma
17	Neural network (NN)	nnet	size, decay	nnet
18	NN with feature extraction	nnet	size, decay	pcaNNet
19	Monotone multi-layer perceptron NN	monmlp	hidden1, n.ensemble	monmlp
20	Stacked autoencoder deep NN	deepnet	layer1, layer2, layer3, hidden_dropout, visible_dropout	dnn
21	Stochastic gradient boosting	gbm	n.trees, interaction.depth, shrinkage, n.minobsinnode	gbm

* The version of each package is shared under section 4.10. Links are forwarded to the CRAN page (except those imported from Tensorflow and H2O) of each package for assessment of version, vignettes, advanced functionality, and description. †Parameters are crucial to optimize for accuracy (95% CI). Similar models have different parameters. Multi-layered neural networks are used for deep learning. In some instances, only the layer 1 is used. For such instances the classifier is considered a neural network.

4.3 Available and tuned hyperparameters for each model

4.4 Machine learning performance benchmarks

Please follow up on performance metrics for classification problem by reading Sokolova 2009.

↰ Link to metrics definitions

- **Sensitivity**, is how many true cases are correctly classified to their expected class. Or **recall**, is the fraction of events where we correctly declared i form all cases where the true of state of the world is i. $TP/(TP+FN)$
- **Specificity**, is how many wrong cases are correctly classified elsewhere. $TN/(TN+FP)$
- **Precision**, is the fraction of events where we correctly declared i out of all instances where the algorithm declared i. $TP/(TP+FP)$
- **Accuracy**, is an overall measure that assesses the predictive model by comparing predicted classes to observed expected classes. Scores can also be shown as the area under the receiver operator curve (AUROC, 95% CI). $(TN+TP)/(TP+TN+FP+FN)$

4.4.1 Creating the baseline of models performance

Machine learning models were trained only without tuning for hyperparameter optimization. Metrics generated show the raw performance of each model.

↰ Precision and recall are best for multi class learning

For this type of nominal data, classification models (not regression) are used, see Section /refsubsec:models. The performance metrics for this type of models are an accuracy score and kappa, which takes into account the possibility of the agreement occurring by chance (the kappa score however reflects the adequate agreement). Standard error (SE in red) bars for the kappa significance per model reproducible across 10 cross-validation each repeated 5 times. Minimum and maximum accuracy thresholds are held at 95% confidence intervals.

↰ Kappa is Cohen's (unweighted) Kappa statistic averaged across the resampling results

Load standard error and deviation equations.

```
summary_SE <- function(data=NULL, measurevar, groupvars=NULL, na.rm=FALSE,
```

Table 2: Hyperparameters for each classifier.

Classifier trained	Hyperparameter	Scenario A*	Scenario B**	Scenario C***
Naive bayes	laplace			
	usekernel			
	adjust			
Weighted k-Nearest Neighbors	kmax			
	distance			
	kernel			
Penalized multinomial regression	decay			
Random forest	mtry			
Regularized random forest	mtry			
	coefReg			
	coefImp			
Linear discriminant analysis (LDA)	dimen			
Localized LDA	k			
Flexible discriminant analysis (FDA)	degree			
	nprune			
Bagged FDA	degree			
	nprune			
Bagged FDA using gCV pruning	degree			
Penalized discriminant analysis	lambda			
Partial least squares	ncomp			
Support vector machines (SVM) with linear kernel	C	0.1		
L2 regularized SVM (dual) with linear kernel	cost			
	loss			
SVM with polynomial kernel	degree			
	scale			
	C			
SVM with radial basis function kernel	sigma			
	C			
Neural network (NN)	size			
	decay			
NN with feature extraction	size			
	decay			
Monotone multi-layer perceptron NN	hidden1			
	n.ensemble			
Stacked autoencoder deep NN	layer1			
	layer2			
	layer3			
	hidden_dropout			
	visible_dropout			
Boosted logistic regression	nIter			
Stochastic gradient boosting	n.trees			
	interaction.depth			
	shrinkage			
	n.minobsinnode			
Multilayer perceptron network by stochastic gradient descent	size			
	l2reg			
	lambda			
	learn_rate			
	momentum			
	gamma			
	minibatchsz			
	repeats			

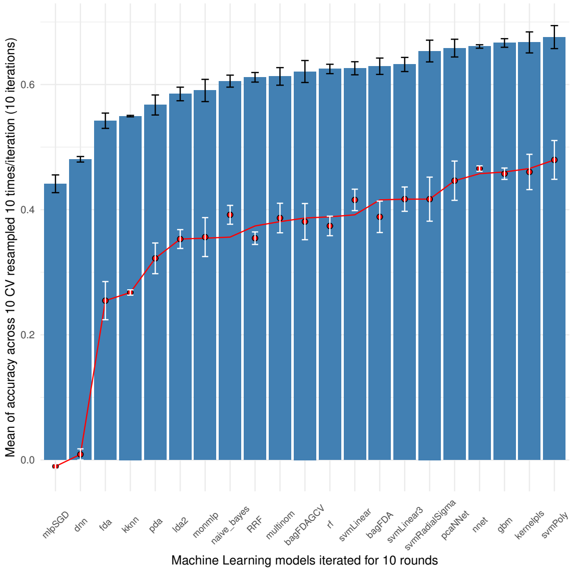
*Comparing samples with individuals grouped by either having a central nervous system relapse (CNS), systemic relapse (SYST), no relapse (NOREL) or controls. **Comparing samples with individuals grouped by either having an activated B-cell (ABC) or germinal center B-cell (GCB) diagnosis. ***Comparing samples with individuals grouped by either having an tumor occurrence in lymph nodes (LN) in contrast to extra-nodal (EN) presence. The tuning parameters were iterated across a random selection grid for best tuning. Grid configurations are found on Github.

```
                      conf.interval=.95, .drop=TRUE) {

    length2 <- function (x, na.rm=FALSE) {
        if (na.rm) sum(!is.na(x))
        else       length(x)
    }
    datac <- ddply(data, groupvars, .drop=.drop,
                   .fun= function(xx, col, na.rm) {
                       c( N    = length2(xx[,col], na.rm=na.rm),
                         mean = mean   (xx[,col], na.rm=na.rm),
                         sd   = sd     (xx[,col], na.rm=na.rm)
                         )
                   },
                   measurevar,
                   na.rm
                   )
    datac <- rename(datac, c("mean"=measurevar))
    # Calculate standard error of the mean
    datac$se <- datac$sd / sqrt(datac$N)
    # Confidence interval multiplier for standard error
    # Calculate t-statistic for confidence interval:
    ciMult <- qt(conf.interval/2 + .5, datac$N-1)
    datac$ci <- datac$se * ciMult
    return(datac)
}
```

Metrics for classification performance without tuning for hyperparameter optimization. Quick comparison of statistical learning on the DLBCL data.

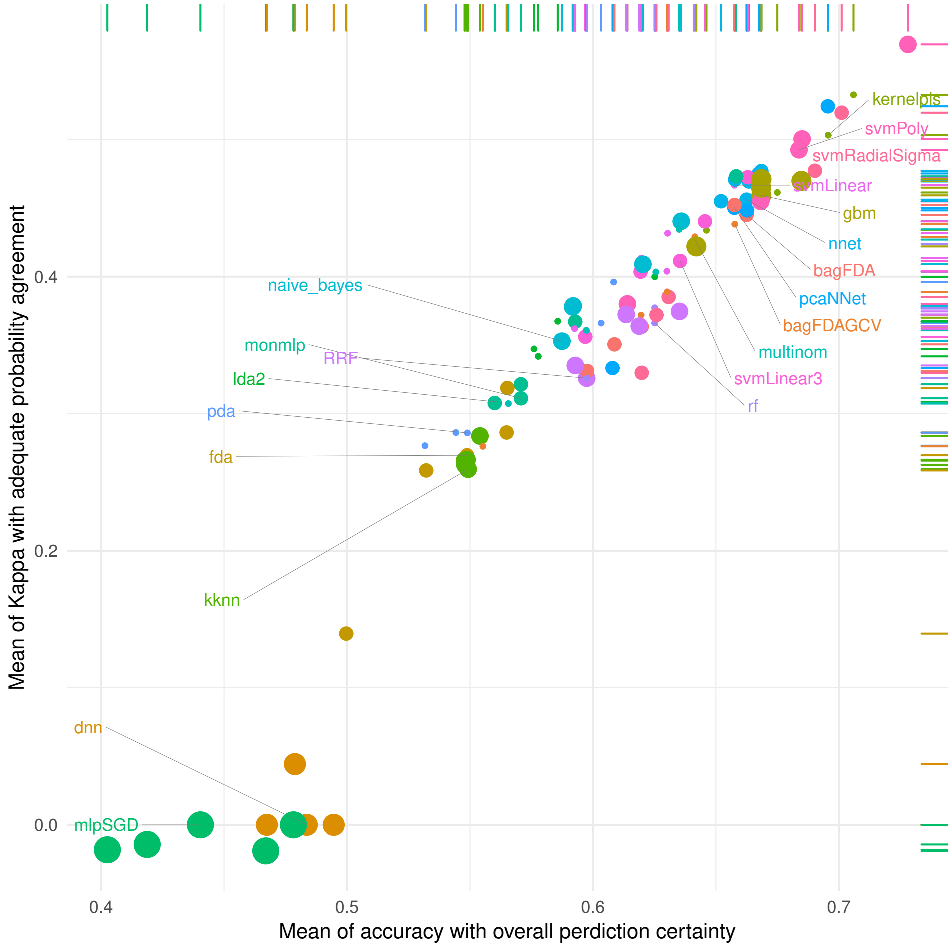
↰ Accuracy, is the true prediction rate averaged over cross-validation iterations

```
accuracy.pre <- read.table("./data/reports.437017/log.Accuracy.performance1.multianalysis.seed5437259.4
kappa <- read.table("./data/reports.437017/log.Kappa.performance1.multianalysis.seed5437259.437017.txt"

accuracy.se <- summary_SE(accuracy.pre, measurevar = "Mean", groupvars = "model")
kappa.se <- summary_SE(kappa, measurevar = "Mean", groupvars = "model") %>%
    arrange(Mean) %>%
    mutate(model = factor(model, unique(model)))

accuracy.se %>%
    arrange(Mean) %>%
    mutate(model = factor(model, unique(model))) %>%
    ggplot(aes(x = model,
               y = Mean)) +
    geom_bar(position="dodge",
             stat="identity",
             fill = "steelblue") +
    geom_errorbar(data = accuracy.se,
                  aes(ymin=Mean-se,
                      ymax=Mean+se),
                  width=.3,
                  position=position_dodge(.9)) +
    geom_line(data = kappa.se,
              aes(x = as.integer(model),
                  y = Mean),
              color = "red") +
    geom_point(data = kappa.se,
               size=2, shape=21, fill="red") +
    geom_errorbar(data = kappa.se,
                  aes(ymin=Mean-se,
                      ymax=Mean+se),
                  width=.25,
                  position=position_dodge(.9),
                  color = "white") +
    theme_minimal() +
    ylab("Mean of accuracy across 10 CV resampled 10 times/iteration (10 iterations)") +
    xlab("Machine Learning models iterated for 10 rounds") +
    theme(legend.position = "top") +
    guides(fill=guide_legend(title="Number of parameters per model")) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))
```



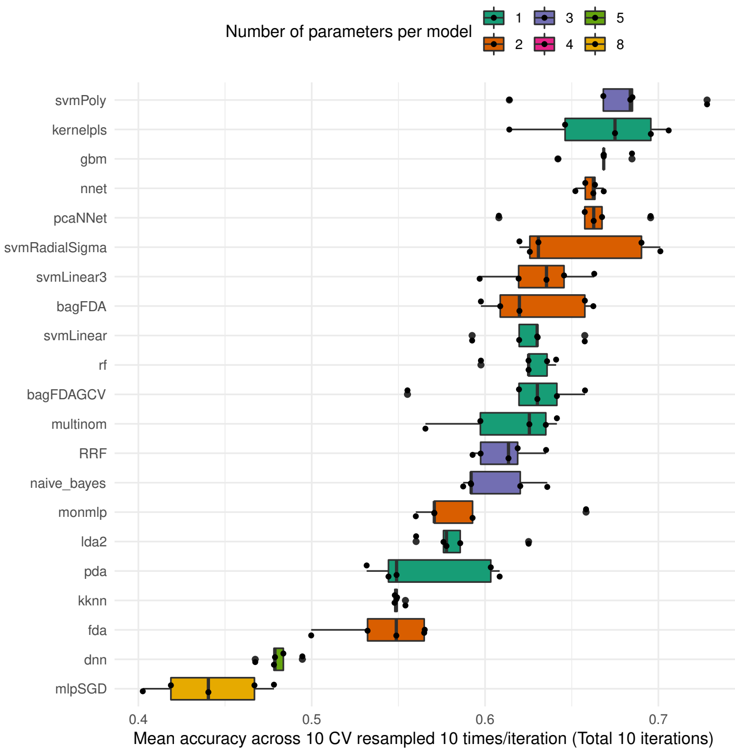
Kappa (vertical axis) and accuracy (horizontal axis) calculated from the performance tests of machine learning models. The higher kappa is the stronger agreement for a prediction and classification. Although, this chart shows little correlation between prediction accuracy and outcome of each model, its important to highlight the number of parameters tune for each classifier. Because, based on a recent paper from Google in Nature, Rajkomar 2018, deep learning models with many layers and hyperparameters perform well enough as a regularized logistic regression model (Supplementary text). In the paper however, little was mentioned about this discrepancy in the published results. Lastly, it remains important to address the value of the parameters that are manipulated and the amount of time spent training the model based on the tuning process.

```r
## get single instance label names
model.names <- data.frame(accuracy.pre, kappa) %>%
    select(model, iteration, Mean, Mean.1) %>%
    filter(iteration == 1)

data.frame(accuracy.pre, kappa) %>%
    ggplot(aes(x = Mean,
               y = Mean.1,
               color = model,
               label = model)) +
    geom_point(aes(size=parameters, fill = model), shape=21) +
    geom_rug(aes(color = as.character(model)),
             sides = "tr",
             show.legend = F) +
    geom_text_repel(data = subset(model.names, Mean >= .6),
                    nudge_x = .04,
                    nudge_y = -.03,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 3) +
    geom_text_repel(data = subset(model.names, Mean < .6),
                    nudge_x = -.1,
                    force = 8,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 3) +
    theme_minimal() +
    ylab("Mean of Kappa with adequate probability agreement") +
    xlab("Mean of accuracy with overall perdiction certainty") +
    theme(legend.position = "none")
```



Mean accuracy (95% CI) registered for machine learning fitting for all classification groups. Iterations for reproducibility were executed 10 times.

```
accuracy.pre %>%
    ggplot(aes(x = reorder(model, Mean),
               y = Mean,
               fill = as.character(parameters))) +
    geom_boxplot() +
    geom_jitter(shape=16, position=position_jitter(0.2), cex = 1.5) +
    coord_flip() +
    scale_fill_brewer(palette = "Dark2") +
    theme_minimal() +
    ylab("Mean accuracy across 10 CV resampled 10 times/iteration (Total 10 iterations)") +
    xlab("") +
    theme(legend.position = "top") +
    guides(fill=guide_legend(title="Number of parameters per model"))
```

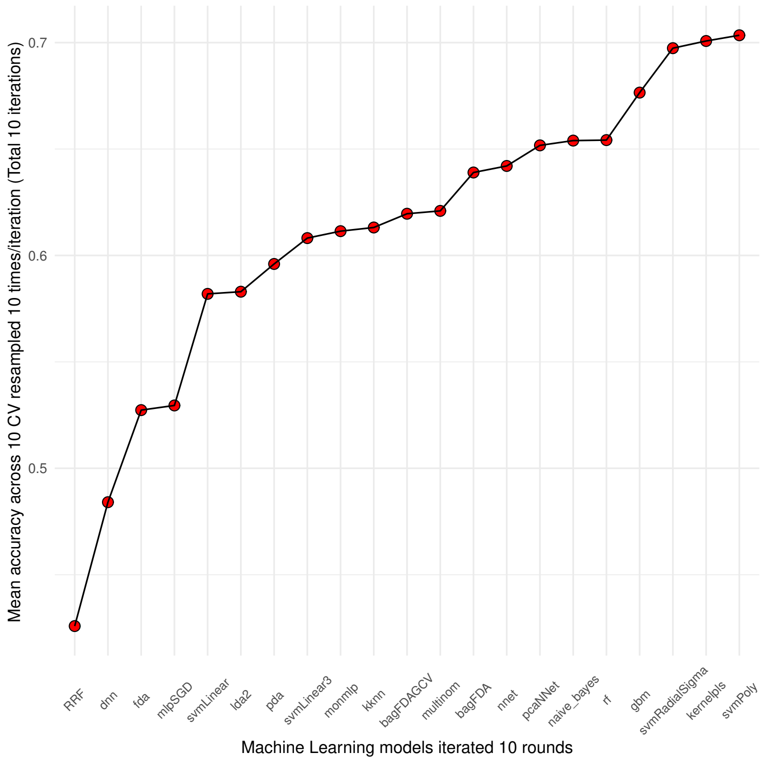


4.4.2 Models performance with hyperparameter tuning

Compared to the baseline, the parameters available for each learner should increase its performance at predicting each expected outcome. Tuning these hyperparameters will leverage the results with increased accuracy.

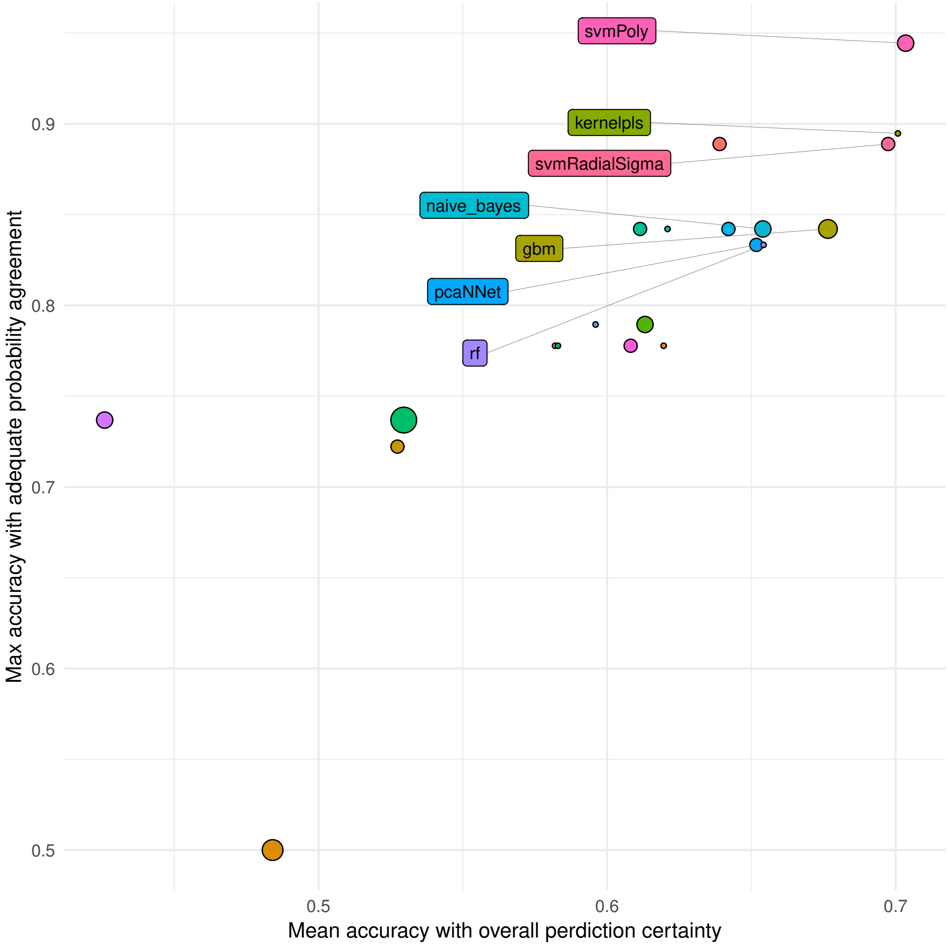
Mean accuracy (95% CI) registered for machine learning fitting for all classification groups. Iterations for reproducibility were executed 10 times.

```
accuracy.pst <- read.table("./data/reports.437017/log.Accuracy.performance2.hyperTuning.seed5437259.437
accuracy.pst %>%
    arrange(Mean) %>%
    mutate(model = factor(model, unique(model))) %>%
    ggplot(aes(x = model,
               y = Mean)) +
    geom_point(size = 3, shape = 21, fill = "red") +
    geom_line(aes(x = as.integer(model),
                  y = Mean)) +
    scale_fill_brewer(palette = "Dark2") +
    theme_minimal() +
    ylab("Mean accuracy across 10 CV resampled 10 times/iteration (Total 10 iterations)") +
    xlab("Machine Learning models iterated 10 rounds") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))
```



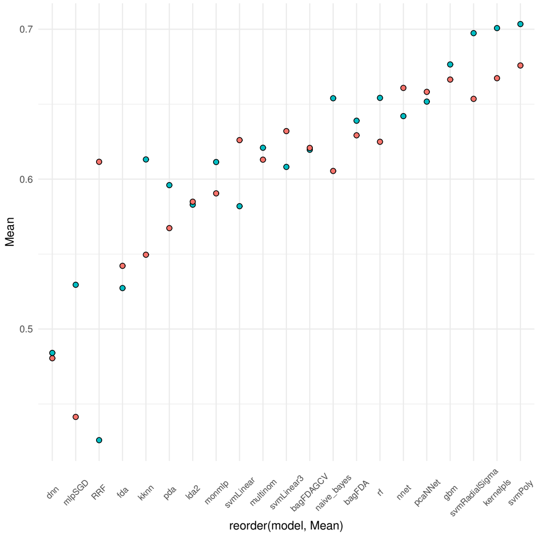
Showing the mean accuracy of each model across its maximum predictive performance.

```
accuracy.pst %>%
    ggplot(aes(x = Mean,
               y = Max.,
               fill = model,
               label = as.factor(model))) +
    geom_point(aes(size=parameters), shape=21) +
    geom_label_repel(data = subset(accuracy.pst, Mean >= .65),
                     nudge_x = -.1,
                    force = 10,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 3) +
    theme_minimal() +
    ylab("Max accuracy with adequate probability agreement") +
    xlab("Mean accuracy with overall perdiction certainty") +
    theme(legend.position = "none")
```



Comparing the performance of each model during baseline testing versus optimization, both during the training process.

```
pre.tune <- accuracy.se %>%
    select(model, Mean)
post.tune <- accuracy.pst %>%
    select(model, Mean)

full_join(pre.tune, post.tune, by = "model") %>%
    gather("condition", "Mean", 2:3) %>%
    arrange(Mean) %>%
    mutate(model = factor(model, post.tune$model)) %>%
    ggplot(aes(x = reorder(model, Mean),
           y = Mean,
           fill = condition)) +
    geom_point(shape = 21, size = 2) +
    theme_minimal() +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8),
          legend.position = "none")
```

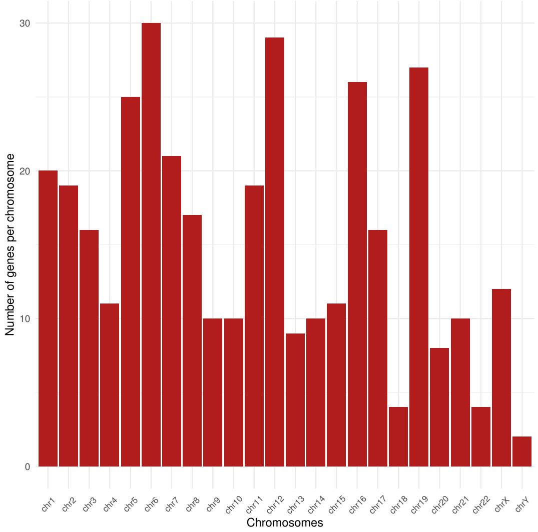


4.4.3 Genomic representation of selected genes

Distribution of all the genes that were selected after differential expression and network analysis.

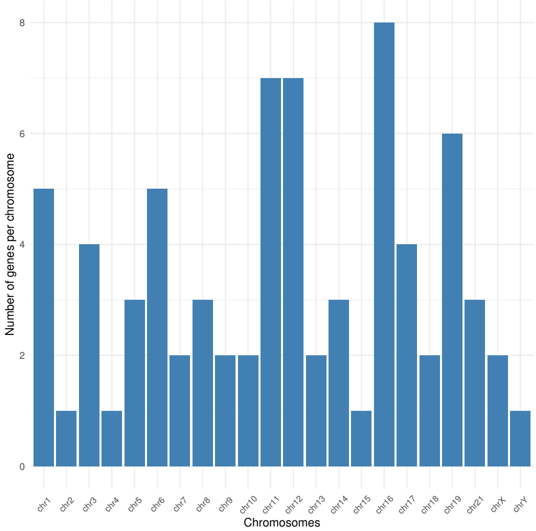
```
read.table("./data/log.gene.names_allnetworks.txt", header = T) %>%
    mutate(order = gsub("chr", "", chromosome)) %>%
    mutate(order = gsub("X", "24", order)) %>%
    mutate(order = gsub("Y", "25", order)) %>%
    arrange(as.numeric(order)) %>%
    ggplot(aes(x = reorder(chromosome, as.numeric(order)))) +
    geom_histogram(stat = "count", fill = "firebrick") +
    theme_minimal() +
    labs(x = "Chromosomes",
         y = "Number of genes per chromosome") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))
```

```
Warning:  Ignoring unknown parameters:  binwidth, bins, pad
```



Distribution of the genes with preferential transcriptional characteristics to predict individuals who are at risk or not of CNS relapse after diagnosis and DLBCL treatment.

```
read.table("./data/log.gene.names_bestLambda.txt", header = T) %>%
    mutate(order = gsub("chr", "", chromosome)) %>%
    mutate(order = gsub("X", "24", order)) %>%
    mutate(order = gsub("Y", "25", order)) %>%
    arrange(as.numeric(order)) %>%
    ggplot(aes(x = reorder(chromosome, as.numeric(order)))) +
    geom_histogram(stat = "count", fill = "steelblue") +
    theme_minimal() +
    labs(x = "Chromosomes",
         y = "Number of genes per chromosome") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))

Warning:  Ignoring unknown parameters:  binwidth, bins, pad
```



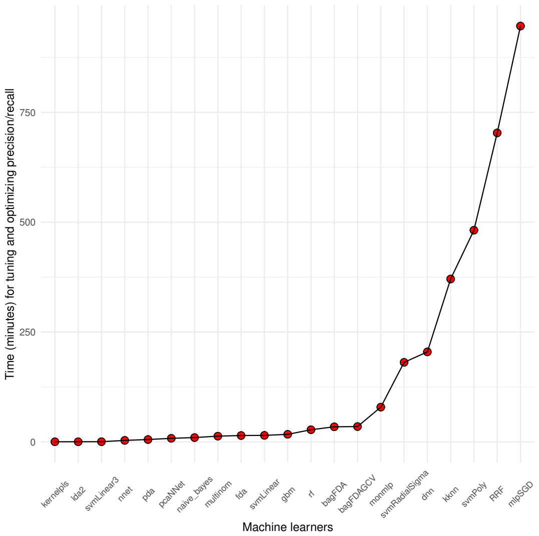
4.4.4 Prediction accuracy of each classifier at optimal parameters

Final report of the actual accuracy (95% CI) for each machine learning model from comparing predicted values and expected outcomes.

How long (seconds) a statistical learner requires to optimize the hyperparameters and gets the highest significant accuracy on expected data.

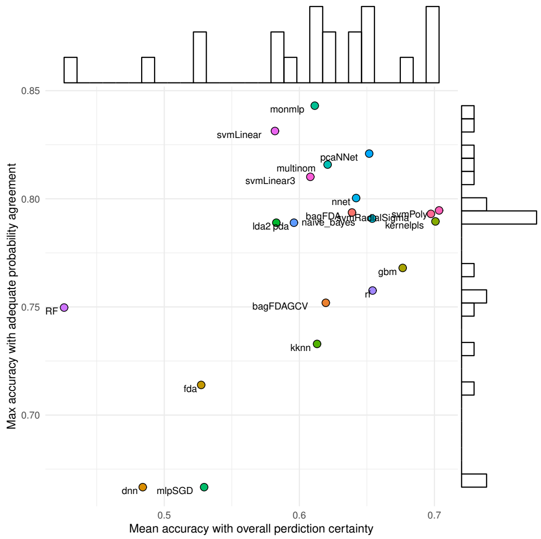
↰ Data are retrieved from Confusion Matrix

```
df <- read.table("./data/reports.437017/log.performance3.full.hyperTuning.seed5437259.437017.txt", heade
df$group <- gsub("[0-9]", "", df$group)

df %>%
    arrange(durationMinutes) %>%
    mutate(model = factor(model, unique(model))) %>%
    ggplot(aes(x = model,
               y = durationMinutes)) +
    geom_point(stat = "identity",
               size=3, shape=21,
               fill = "red") +
    geom_line(aes(x = as.integer(model))) +
    theme_minimal() +
    xlab("Machine learners") +
    ylab("Time (minutes) for tuning and optimizing precision/recall") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8))
```



Showing the mean accuracy of each model in contrast to their specificity for predicting individuals classes.

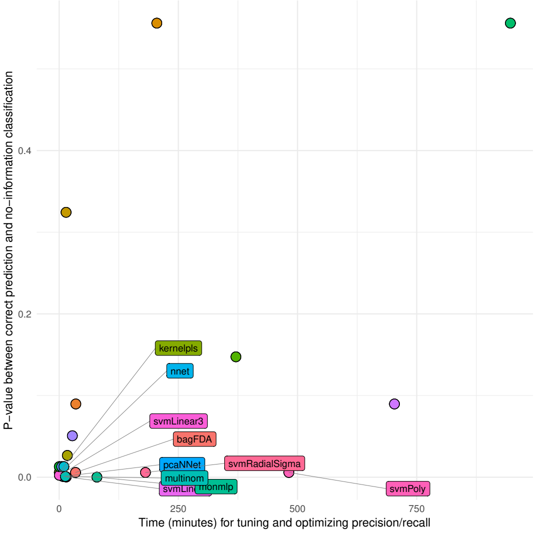
```
specificity.mean <- df %>%
    select(model, Specificity) %>%
    summary_SE(measurevar = "Specificity", groupvars = "model") %>%
    select(model, Specificity)

accuracy.mean <- accuracy.pst %>%
    select(model, Mean)

p <- full_join(specificity.mean, accuracy.mean, by = "model") %>%
        ggplot(aes(x = Mean,
               y = Specificity,
               fill = as.factor(model))) +
    geom_point(shape=21, size = 3) +
    geom_text(aes(x = Mean,
                  y = Specificity,
                  label = model,
                  hjust = 1.3,
                  vjust = 1), size = 3) +
    theme_minimal() +
    ylab("Max accuracy with adequate probability agreement") +
    xlab("Mean accuracy with overall perdiction certainty") +
    theme(legend.position = "none")

    ggMarginal(p, type = "histogram", fill="transparent")
```



How is time training a model deliver on the significance of its accuracy? The p-value evaluates whether the overall accuracy rate is greater than the rate of the largest class. Proportions between classes (if one group of samples is larger than an other) is also considered in the hypothesis testing.

↰ Link documentation Section 17.2

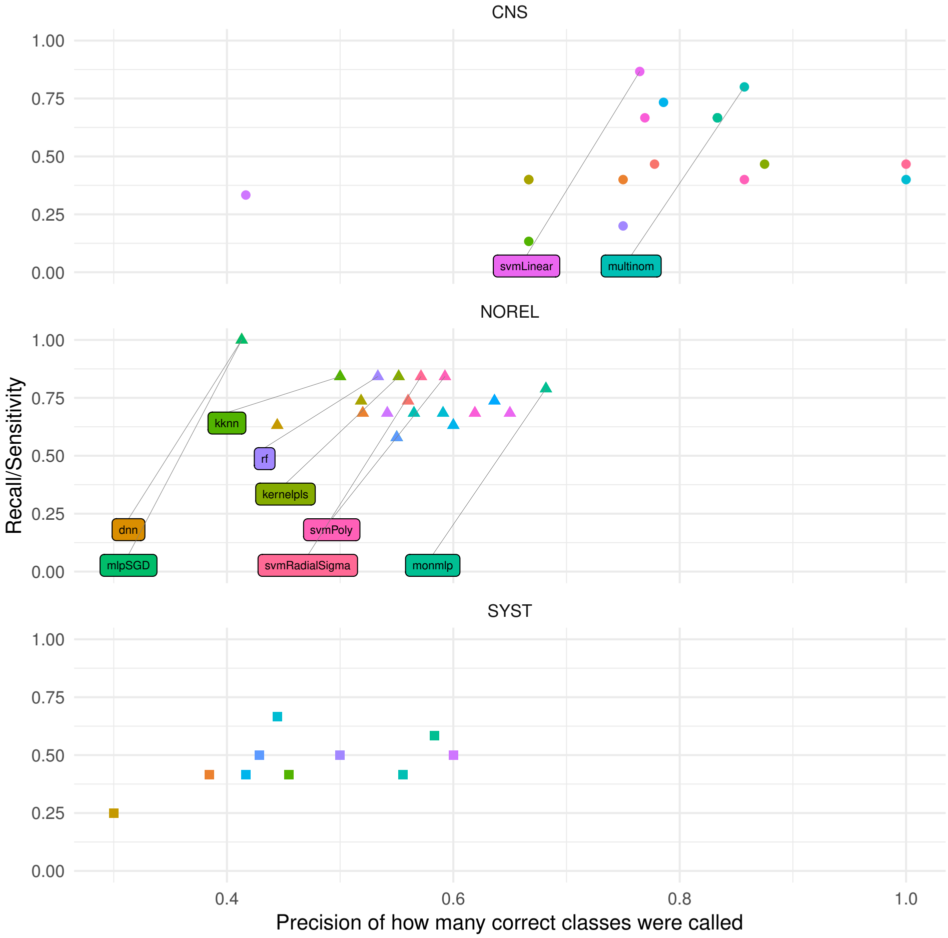
```
unique.acc <- df %>%
    select(model, accuracyPval, durationMinutes) %>%
    unique()

df %>%
    ggplot(aes(y = accuracyPval,
               x= durationMinutes,
               fill = as.character(model),
               label = as.factor(model))) +
    geom_point(size = 4,
               shape = 21) +
    geom_label_repel(data = subset(unique.acc, accuracyPval <= .01),
                     nudge_y = -1,
                     nudge_x = 250,
                     force = 5,
                     segment.color = "grey50",
                     direction = "y",
                     segment.size = 0.1,
                     size = 3) +
    theme_minimal() +
    xlab("Time (minutes) for tuning and optimizing precision/recall") +
    ylab("P-value between correct prediction and no-information classification") +
    theme(legend.position = "none")
```



Precision versus recall across all sample groups for a multi-class classification.

↰True/False Positives/Negatives

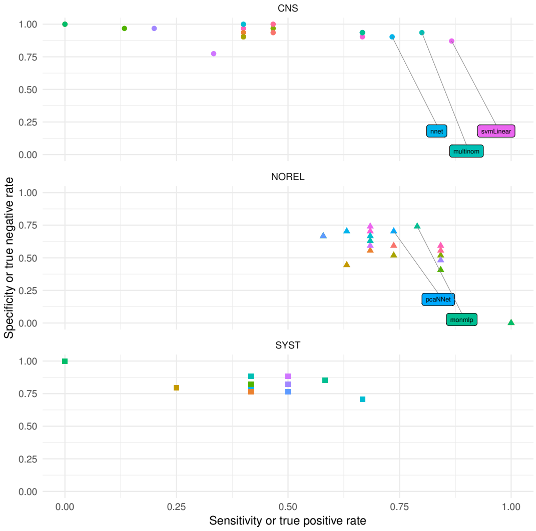
```
prec.rec <- df %>%
    select(group, model, Precision, Recall)

df %>%
    ggplot(aes(x = Precision,
               y = Recall,
               fill = model,
               label = as.factor(model),
               na.rm = T)) +
    geom_point(aes(shape = group, color = model),
               size = 2) +
    geom_label_repel(data = subset(prec.rec, Recall >= .75),
                     nudge_y = -1,
                     nudge_x = -.1,
                     force = 1,
                     segment.color = "grey50",
                     direction = "y",
                     segment.size = 0.1,
                     size = 2) +
    facet_wrap(~ group, ncol = 1) +
    theme_minimal() +
    xlab("Precision of how many correct classes were called") +
    ylab("Recall/Sensitivity") +
    theme(legend.position = "none")

Warning:  Removed 4 rows containing missing values (geom_point).
```



382

383 Specificity and sensitivity across all sample groups.

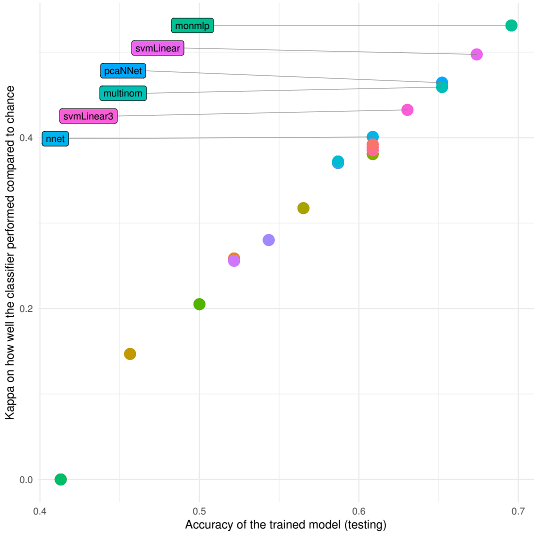
```
spec.sens <- df %>%
    select(group, model, Sensitivity, Specificity)

df %>%
    ggplot(aes(x = Sensitivity,
               y = Specificity,
               fill = model,
               label = as.factor(model))) +
    geom_point(aes(shape = group,
                   color = model),
               size = 2) +
    geom_label_repel(data = subset(spec.sens, Specificity >= .7 & Sensitivity >= .7),
                    nudge_y = -1,
                    nudge_x = .1,
                    force = 1,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2) +
    facet_wrap(~ group, ncol = 1) +
    theme_minimal() +
    xlab("Sensitivity or true positive rate") +
    ylab("Specificity or true negative rate") +
    theme(legend.position = "none")
```



Accuracy and Kappa of each of the optimized model across all sample groups after grid tuning search.

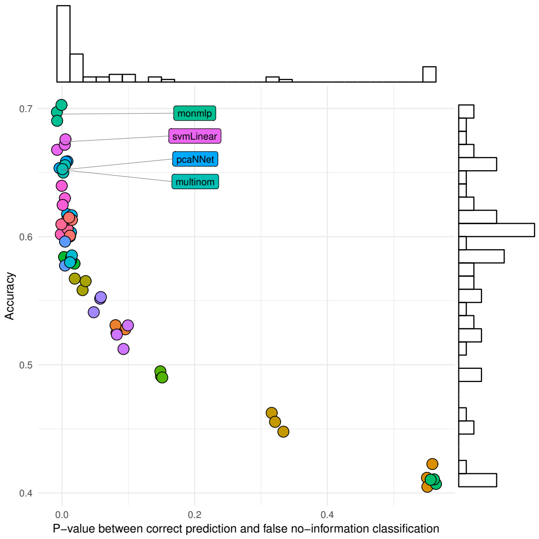
↰ Optimized model

```
pak <- df %>%
    select(model, accuracy, kappa) %>%
    unique
df %>%
    ggplot(aes(x = accuracy,
               y = kappa,
               fill = model,
               label = as.factor(model))) +
    geom_point(aes(size = 4,
               color = model)) +
    geom_label_repel(data = subset(pak, accuracy >= .6 & kappa >= .4),
                     nudge_x = -.2,
                     force = 1,
                     segment.color = "grey50",
                     direction = "y",
                     segment.size = 0.1,
                     size = 3) +
    theme_minimal() +
    xlab("Accuracy of the trained model (testing)") +
    ylab("Kappa on how well the classifier performed compared to chance") +
    theme(legend.position = "none")
```



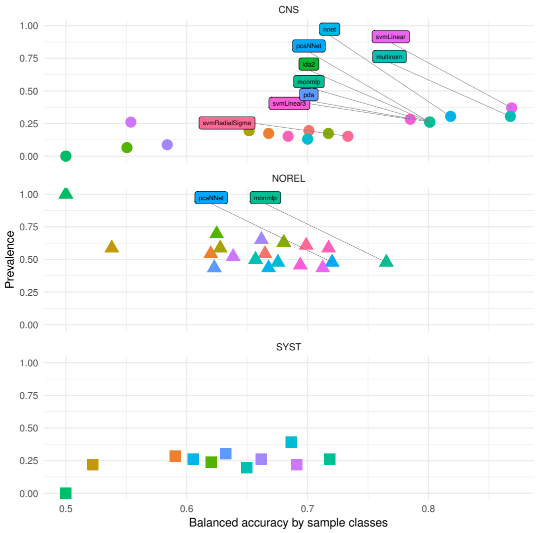
Accuracy versus the p-value of each classification. The p-value is a hypothesis test between predicting expected samples and the probability that the classification is biased by disproportionate class sizes (one group of samples is larger than an other).

```
paap <- df %>%
    select(model, accuracy, accuracyPval) %>%
    unique
p <- df %>%
    ggplot(aes(x = accuracyPval,
               y = accuracy,
               label = as.factor(model),
               fill = model)) +
    geom_point(aes(size = 3,
               fill = model),
               shape = 21,
               position=position_jitter(width=.01,height=.01)) +
    geom_label_repel(data = subset(paap, accuracy >= .65 & accuracyPval <= .01),
                    nudge_x = .2,
                    force = 1,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 3) +
##    scale_x_reverse(limits = rev(levels(df$accuracyPval))) +
    theme_minimal() +
    ylab("Accuracy") +
    xlab("P-value between correct prediction and false no-information classification") +
    theme(legend.position = "none")

    ggMarginal(p, type = "histogram", fill="transparent")
```



Prevalence of cases for each classifier. Were the classes perfectly balanced? A positive predictive score is similar to precision while accounting for disproportionality of the classes.

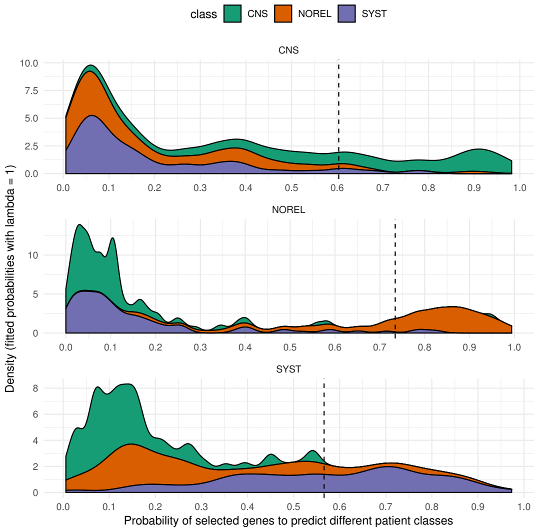
```
pbd <- df %>%
    select(group, model, Balanced.Accuracy, Detection.Prevalence)
df %>%
    ggplot(aes(x = Balanced.Accuracy,
               y = Detection.Prevalence,
               label = as.factor(model),
               fill = model)) +
    geom_point(aes(size = 2,
               color = model,
               shape = as.factor(group))) +
    geom_label_repel(data = subset(pbd, Balanced.Accuracy >= .72),
                     nudge_x = -.1,
                     nudge_y = .5,
                     force = 15,
                     segment.color = "grey50",
                     direction = "y",
                     segment.size = 0.1,
                     size = 2) +
    facet_wrap(~ group, ncol = 1) +
    theme_minimal() +
    xlab("Balanced accuracy by sample classes") +
    ylab("Prevalence") +
    theme(legend.position = "none")
```



4.4.5 Probability to classify samples with the best model

Using the best model, support vector machines with a polynomial kernel, gets the cleanest predictions of individuals with of CNS and systemic relapse.

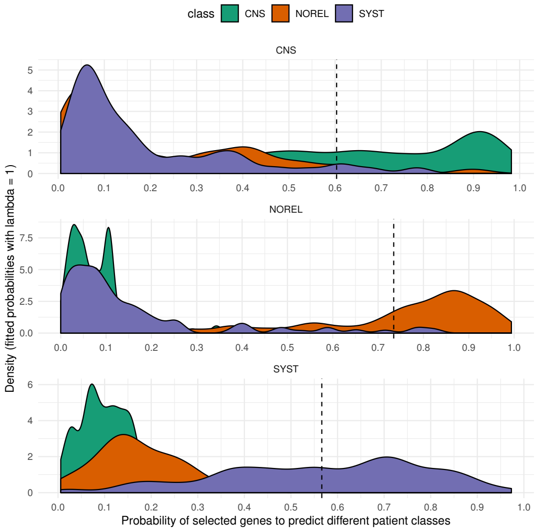
```
dfp <- read.table("./data/reports.437017/log.performance6.class_probabilities.seed5437259.437017.txt", 
dfm = NULL
classes.s <- c("CNS", "NOREL", "SYST")
for (cp in 1:3) {
    dfmx <- dfp %>%
        select(one_of("original", classes.s[[cp]])) %>%
        filter(original == classes.s[[cp]])
    dfm <- rbind(dfm, data.frame(original = classes.s[[cp]],
                                 grp.mean = mean(dfmx[,2])))
}

dfp %>%
    gather("class", "prob", 3:5) %>%
    ggplot(aes(x = prob,
               fill = class)) +
##    geom_density(alpha = 0.3) +
    geom_density(position = "stack") +
##    geom_area(aes(fill = class), stat ="bin", alpha=0.6) +
    geom_vline(data = dfm, mapping = aes(xintercept=grp.mean), linetype = "dashed") +
    scale_x_continuous(breaks = pretty(dfp$CNS, n = 10)) +
    theme_minimal() +
    facet_wrap( ~ original,
               ncol = 1,
               scales = "free") +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Density (fitted probabilities with lambda = 1)")
```



Similar to the above but non stacked densities.

↰ similar & optional representation

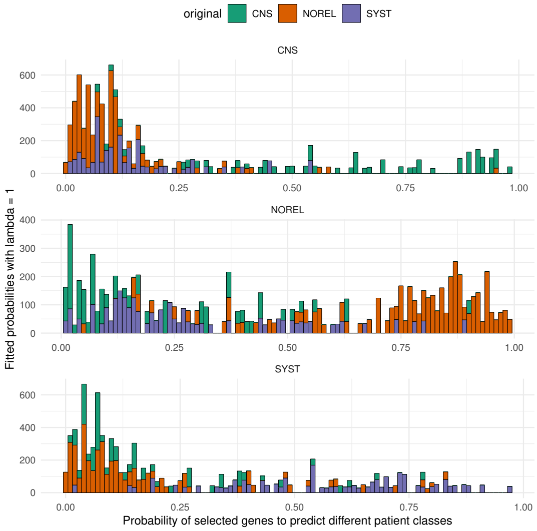
```
dfp %>%
    gather("class", "prob", 3:5) %>%
    ggplot(aes(x = prob,
               fill = class)) +
    geom_density() +
    geom_vline(data = dfm, mapping = aes(xintercept=grp.mean), linetype = "dashed") +
    scale_x_continuous(breaks = pretty(dfp$CNS, n = 10)) +
    theme_minimal() +
    facet_wrap( ~ original,
               ncol = 1,
               scales = "free") +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Density (fitted probabilities with lambda = 1)")
```



Bars showing concentration of good predictions by individual for each outcome class.

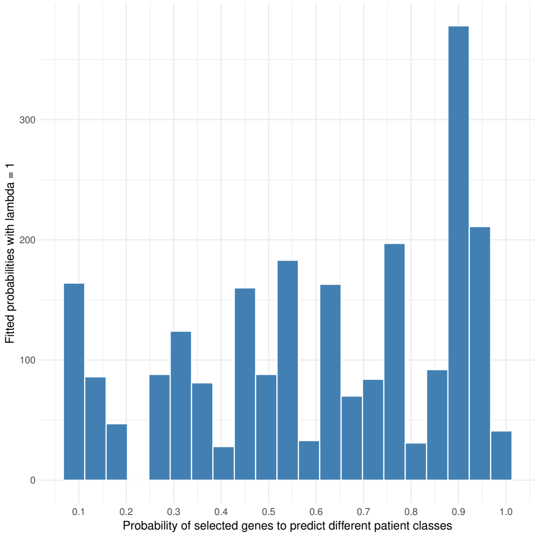
↰ similar & optional representation

```
dfp %>%
    gather("class", "prob", 3:5) %>%
    ggplot(aes(x = prob,
               fill = original)) +
    geom_histogram(binwidth = 0.01, col = "black", size = .1) +
    theme_minimal() +
    facet_wrap( ~ class,
               ncol = 1,
               scales = "free") +
    theme(legend.position = "top") +
    scale_fill_brewer(palette = "Dark2") +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Fitted probabilities with lambda = 1")
```



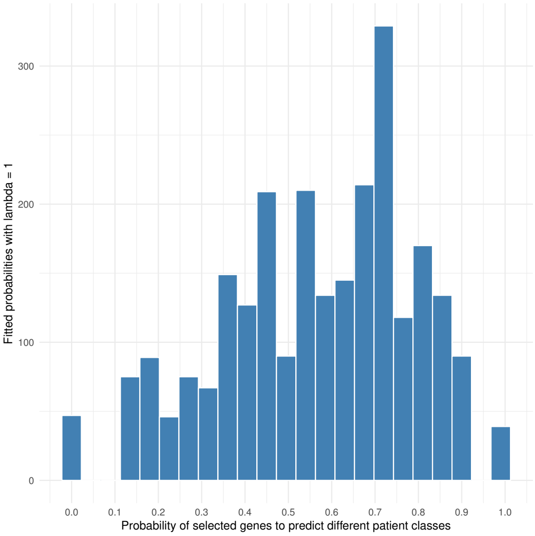
Distribution of probabilities for individuals identified with risk of central nervous system relapse.

```
dfp %>%
    select(original, CNS) %>%
    filter(original == "CNS") %>%
    ggplot(aes(x=CNS)) +
    geom_histogram(binwidth = 0.045, color = "white", fill = "steelblue") +
    theme_minimal() +
    scale_x_continuous(breaks = pretty(dfp$CNS, n = 10)) +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Fitted probabilities with lambda = 1")
```



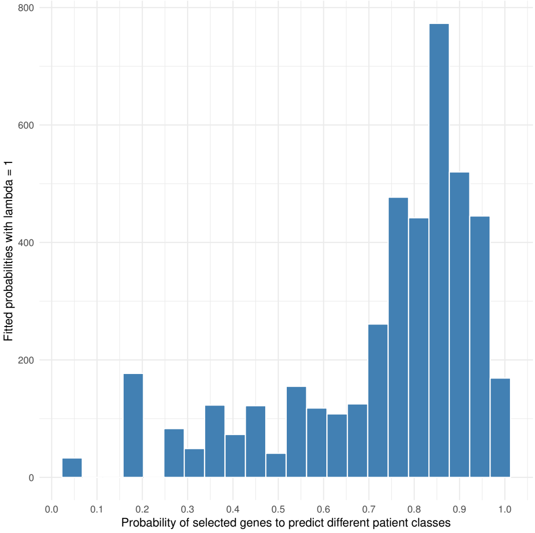
Distribution of probabilities for individuals identified with risk of a systemic relapse.

```
dfp %>%
    select(original, SYST) %>%
    filter(original == "SYST") %>%
    ggplot(aes(x=SYST)) +
    geom_histogram(binwidth = 0.045, color = "white", fill = "steelblue") +
    theme_minimal() +
    scale_x_continuous(breaks = pretty(dfp$SYST, n = 10)) +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Fitted probabilities with lambda = 1")
```



Distribution of probabilities for individuals identified with a no risk of relapse after treatment, at the time of diagnosis.

```
dfp %>%
    select(original, NOREL) %>%
    filter(original == "NOREL") %>%
    ggplot(aes(x=NOREL)) +
    geom_histogram(binwidth = 0.045, color = "white", fill = "steelblue") +
    theme_minimal() +
    scale_x_continuous(breaks = pretty(dfp$NOREL, n = 10)) +
    labs(x = "Probability of selected genes to predict different patient classes",
         y = "Fitted probabilities with lambda = 1")
```

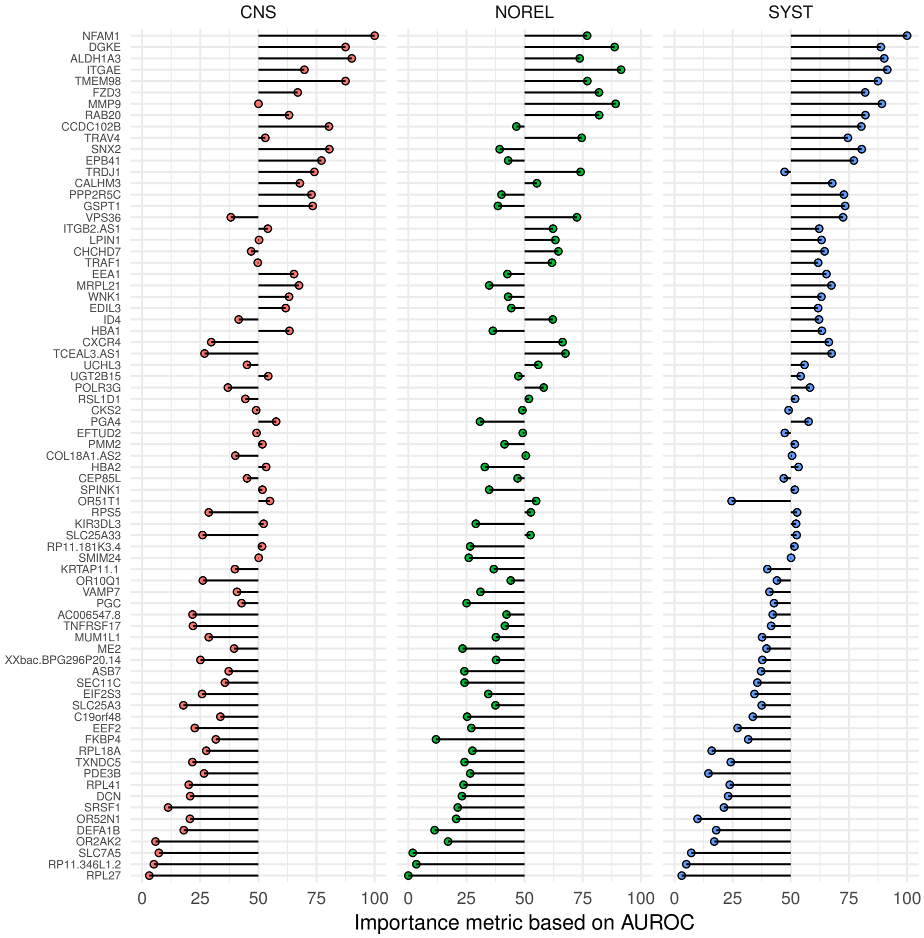


4.5 Importance scope of gene contribution

Importance scores of each gene is calculated based on that variable significance to the model performance. The score is thus model-based therefore taking into account the estimated correlation between predictors. For each class, a gene is attributed an importance. All counts are scaled to 100. A trapezoid rule is used to approximate an integral with linear guidelines. This helps estimate an area under the ROC curve. The area is used to measure the variable importance.

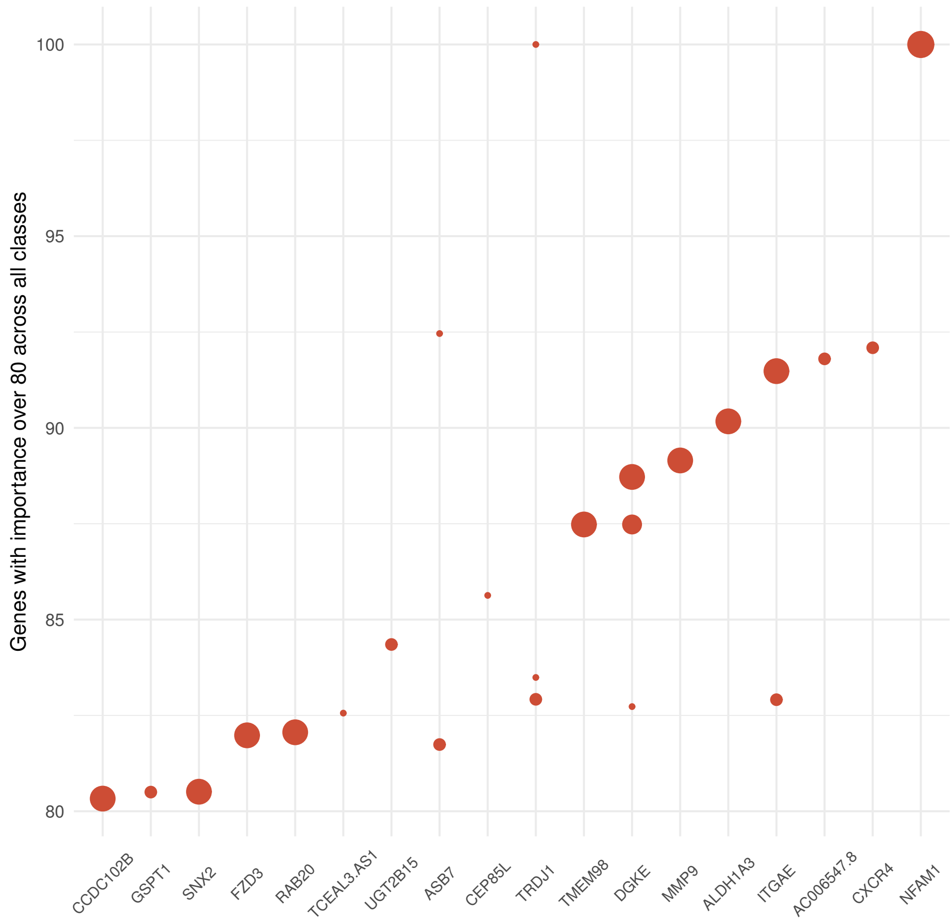
↰Multiclass pairwise decomposition AUROC

```
imp.model <- read.table("./data/reports.437017/log.performance5.importance.seed5437259.437017.txt", hea
imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    filter(model == "svmPoly") %>%
    gather("groups", "importance", 3:5) %>%
    ggplot(aes(x = reorder(genes, importance),
               y = importance,
               fill = as.character(groups))) +
    geom_point(shape=21) +
    geom_segment(aes(y = 50,
                     x = genes,
                     yend = importance,
                     xend = genes),
                 color = "black") +
    facet_wrap(~ groups, ncol = 3) +
    theme_minimal() +
    coord_flip() +
    ylab("Importance metric based on AUROC") +
    xlab("") +
    theme(axis.text.y = element_text(size = 6)) +
    theme(legend.position = "none")
```



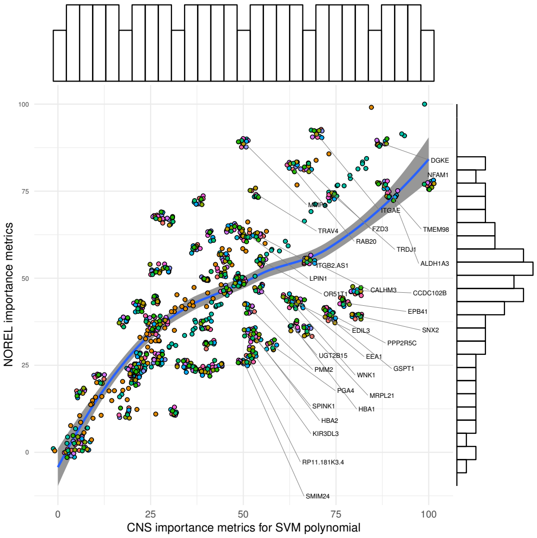
Genes ranked by importance across all classes. The bigger the bubble, the more models were to rely on that one gene.

```
imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    gather("groups", "importance", 3:5) %>%
    filter(importance >= 80) %>%
    ggplot(aes(x = reorder(genes, importance),
               y = importance,
               color = groups)) +
    geom_count(col="tomato3", show.legend=F) +
    theme_minimal() +
    ylab("Genes with importance over 80 across all classes") +
    xlab("") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 8)) +
    theme(legend.position = "top")
```



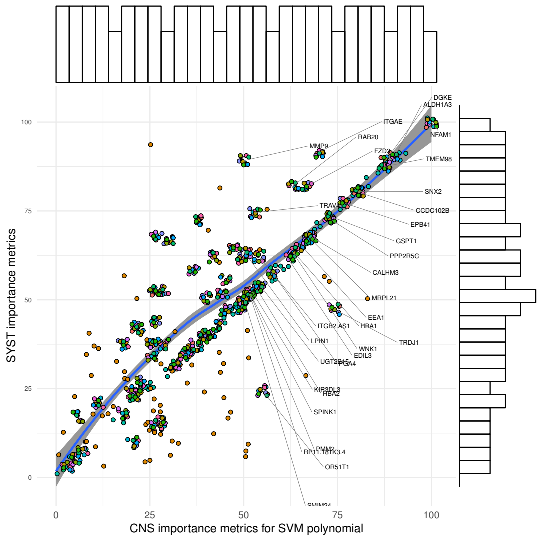
Comparing the importance of genes across all model trained and optimized, annotated by genes, only for the SVM model with a polynomial kernel. Model comparison between individuals diagnosed with CNS relapse or no relapse.

```
model.names <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    select(genes, CNS, NOREL, model) %>%
    filter(model == "svmPoly")

p <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = CNS,
               y = NOREL,
               label = genes)) +
    geom_smooth(method = 'loess', alpha=1) +
    geom_point(aes(fill = model), shape=21,
               position=position_jitter(width=1.5,height=1.5)) +
    theme_minimal() +
    geom_text_repel(data = subset(model.names, CNS >= 50),
                    nudge_x = 20,
                    nudge_y = -20,
                    force = 5,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2) +
    xlab("CNS importance metrics for SVM polynomial") +
    ylab("NOREL importance metrics") +
    theme(axis.text.y = element_text(size = 6)) +
    theme(legend.position = "none")

    ggMarginal(p, type = "histogram", fill="transparent")
```



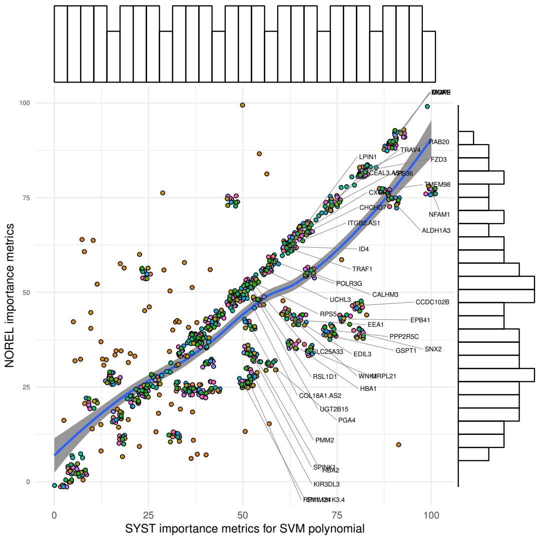
Comparing the importance of genes across all model trained and optimized, annotated by genes, only for the SVM model with a polynomial kernel. Model comparison between individuals diagnosed with CNS relapse or systemic.

```
model.names <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    select(genes, CNS, SYST, model) %>%
    filter(model == "svmPoly")

p <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = CNS,
               y = SYST,
               label = genes)) +
    geom_smooth(method = 'loess', alpha=1) +
    geom_point(aes(fill = model), shape=21,
               position=position_jitter(width=1.5,height=1.5)) +
    theme_minimal() +
    geom_text_repel(data = subset(model.names, CNS >= 50),
                    nudge_x = 20,
                    nudge_y = -20,
                    force = 5,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2) +
    xlab("CNS importance metrics for SVM polynomial") +
    ylab("SYST importance metrics") +
    theme(axis.text.y = element_text(size = 6)) +
    theme(legend.position = "none")

    ggMarginal(p, type = "histogram", fill="transparent")
```



Comparing the importance of genes across all model trained and optimized, annotated by genes, only for the SVM model with a polynomial kernel. Model comparison between individuals diagnosed with systemic or no relapse.

```
model.names <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    select(genes, SYST, NOREL, model) %>%
    filter(model == "svmPoly")

p <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = SYST,
               y = NOREL,
               label = genes)) +
    geom_smooth(method = 'loess', alpha=1) +
    geom_point(aes(fill = model), shape=21,
               position=position_jitter(width=1.5,height=1.5)) +
    theme_minimal() +
    geom_text_repel(data = subset(model.names, SYST >= 50),
                    nudge_x = 20,
                    nudge_y = -20,
                    force = 5,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2) +
    xlab("SYST importance metrics for SVM polynomial") +
    ylab("NOREL importance metrics") +
    theme(axis.text.y = element_text(size = 6)) +
    theme(legend.position = "none")

    ggMarginal(p, type = "histogram", fill="transparent")
```

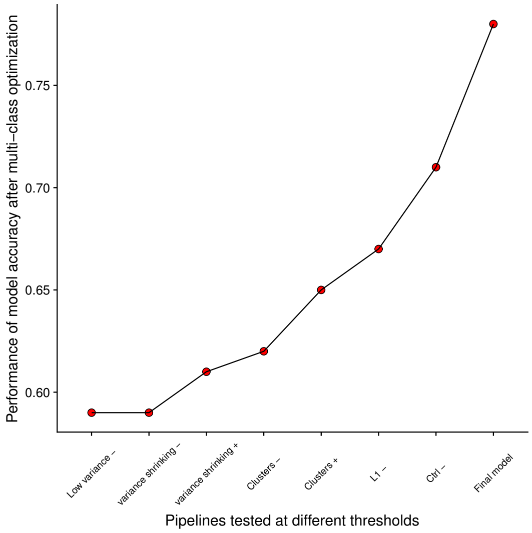


4.6 Accuracy measured across pipelines

Different strategies for gene filtering can be used, either at different thresholds for some methods and implementing or not a method can produce a different accuracy. The performance was measured for SVM model with a polynomial kernel. Metrics were registered manually after the execution of the whole pipeline (from gene expression, to networks, and finally classification).

↰ Final model is wo variance shrinking, wo ctrl, w nets, w L1

```
pipelines <- c("variance shrinking +",
               "Low variance -",
               "variance shrinking -",
               "Clusters +",
               "Clusters -",
               "L1 -",
               "Ctrl -",
               "Final model")
perf.estimated <- c(.61, .59, .59, .65, .62, .67, .71, .78)
dfpe <- data.frame(pipelines = as.factor(pipelines), accuracy = perf.estimated)
dfpe %>%
    arrange(accuracy) %>%
    mutate(pipelines = factor(pipelines, pipelines)) %>%
    ggplot(aes(x = reorder(pipelines, accuracy),
               y = accuracy)) +
    geom_point(stat = "identity",
               size=3, shape=21,
               fill = "red") +
    geom_line(aes(x = as.integer(pipelines))) +
    xlab("Pipelines tested at different thresholds") +
    ylab("Performance of model accuracy after multi-class optimization") +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))
```



4.7 Expression of important genes summarized by RMA scores

The expression is based on RMA normalized likelihoods for 230 individuals. Genes with importance score above 90.

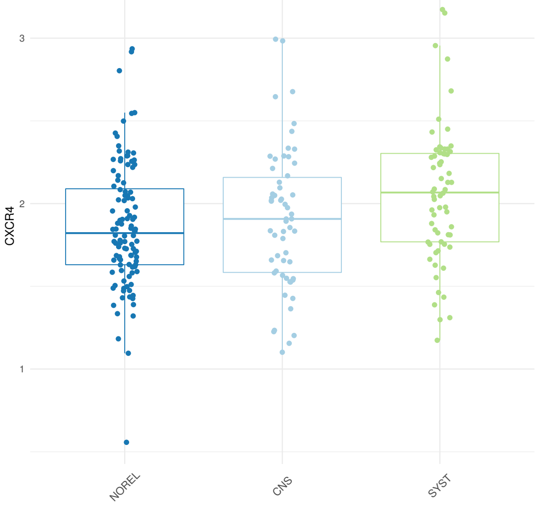
```
imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    gather("groups", "importance", 3:5) %>%
    filter(importance >= 90) %>%
    select(genes) %>%
    unique

        genes
1       CXCR4
2       NFAM1
3  AC006547.8
4     ALDH1A3
29      ITGAE
31      TRDJ1
50       ASB7
```

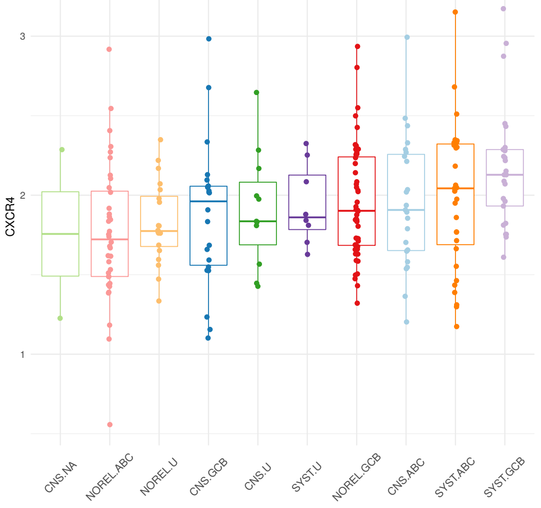
Define function that combines samples, RMA-normalized log2 fold changes, and prognosis classes. Generates plots.

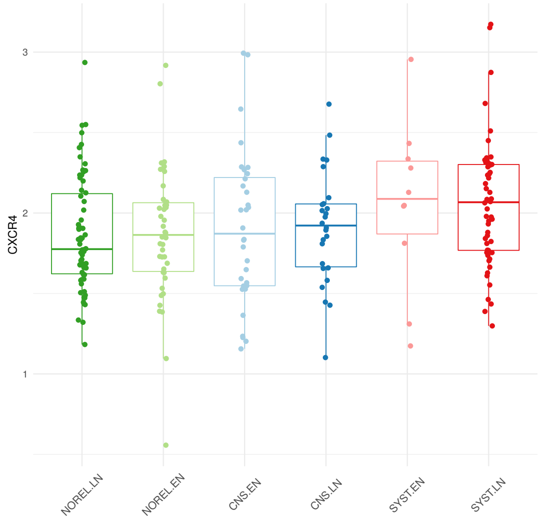
```
rma.genes <- read.table("./data/expressions.epochs50", header = TRUE)

gene.by.class <- function(data = rma.genes, y = meta.selected$Groups, geneSel = 1, implev = 90){
    y.samples <- y
    dat <- data.frame(y.samples, t(rma.genes))

    sel.genes <- imp.model %>%
        mutate(genes = gsub(".TC.*$","",genes)) %>%
        gather("groups", "importance", 3:5) %>%
        filter(importance >= implev) %>%
        select(genes) %>%
        unique

    sel.box <- dat %>%
        select(contains(sel.genes$genes[geneSel])) %>%
        mutate(groups = y.samples)
    colnames(sel.box) <- c("gene", "groups")

    sel.box %>%
        arrange(gene) %>%
        ggplot(aes(x = reorder(groups, gene),
                   y = gene,
                   color = groups)) +
        geom_boxplot(outlier.colour = NA, lwd = .4) +
        geom_jitter(aes(color = factor(groups)),
                    shape=16,
                    position=position_jitterdodge(dodge.width=.8),
                    cex = 2) +
        scale_color_brewer(palette="Paired") +
        theme_minimal() +
        theme(legend.position = "none") +
        xlab("") + ylab(paste0(sel.genes$genes[geneSel])) +
        theme(axis.text.x = element_text(vjust = .5,
                                         angle = 45,
                                         size = 10))
}
```

We choose the genes with the top important score, above than 90. CXRC4

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 1, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 1, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 1, implev = 90)
```

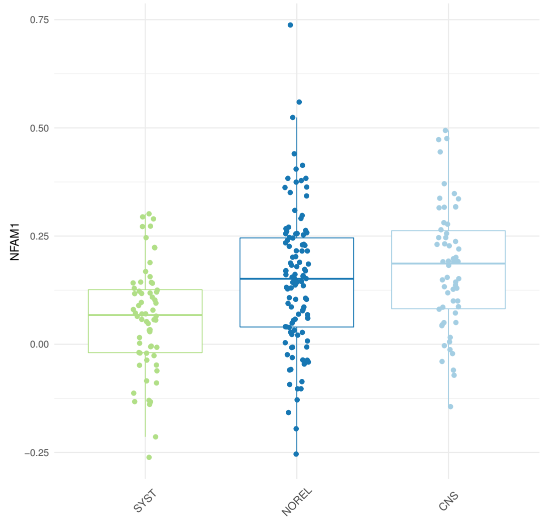


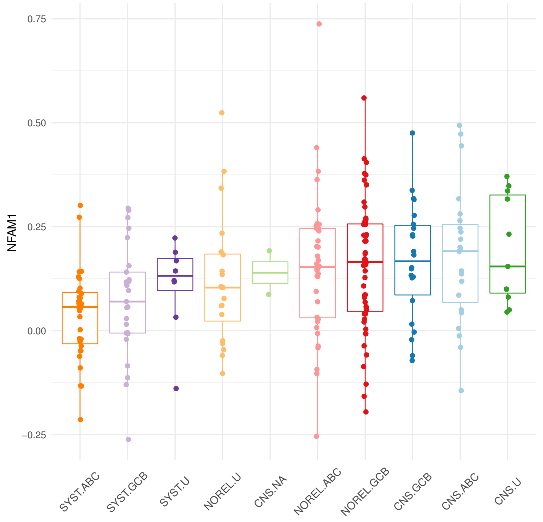


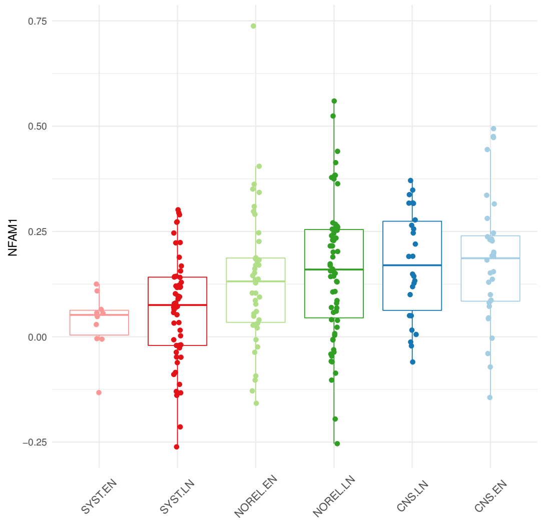


NFMA1

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 2, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 2, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 2, implev = 90)
```

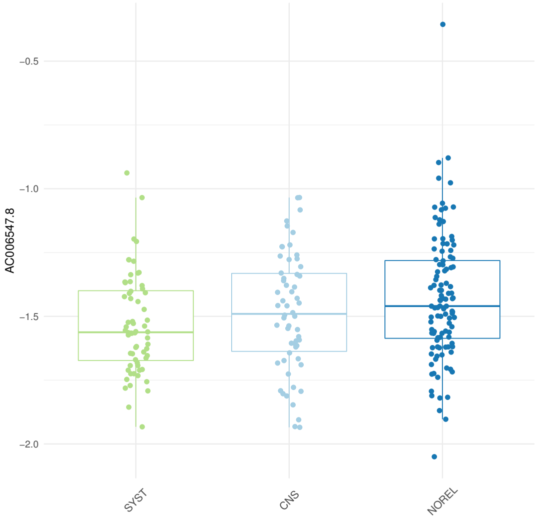


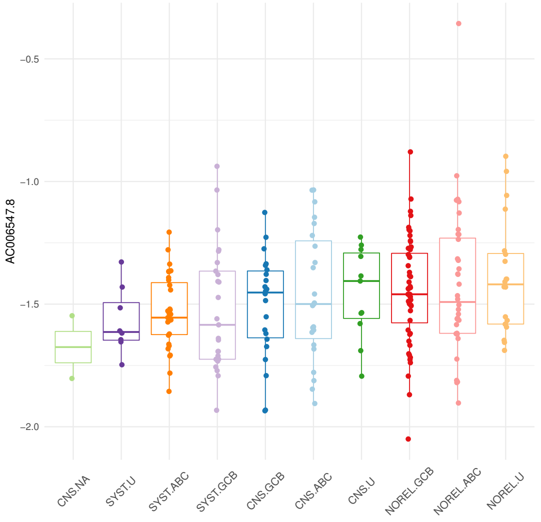


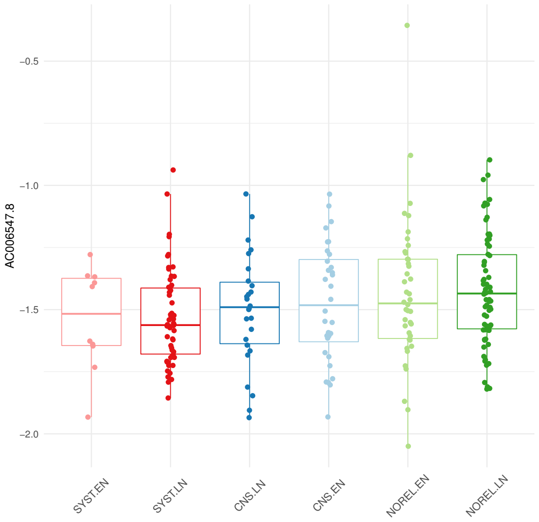


AC006547.8

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 3, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 3, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 3, implev = 90)
```

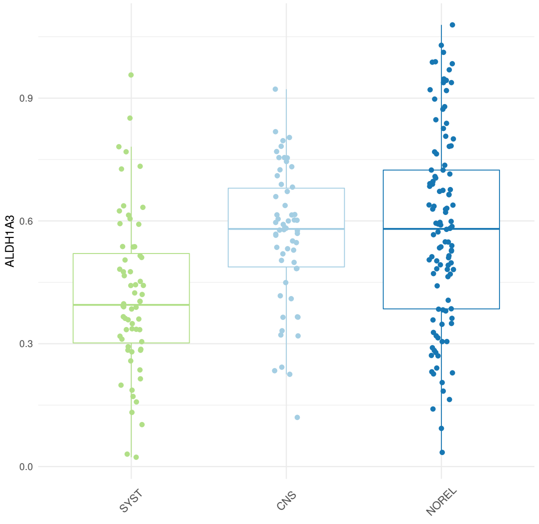


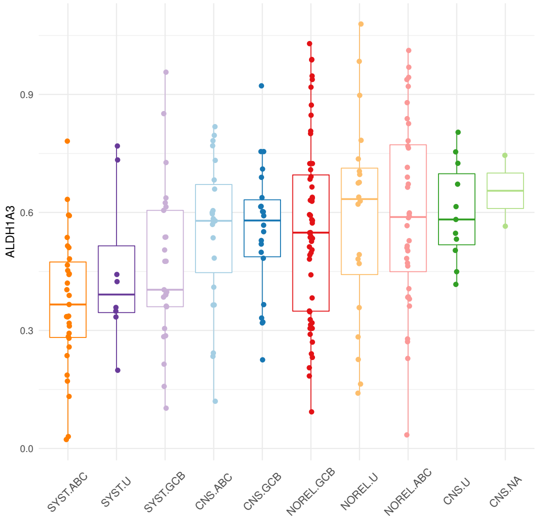


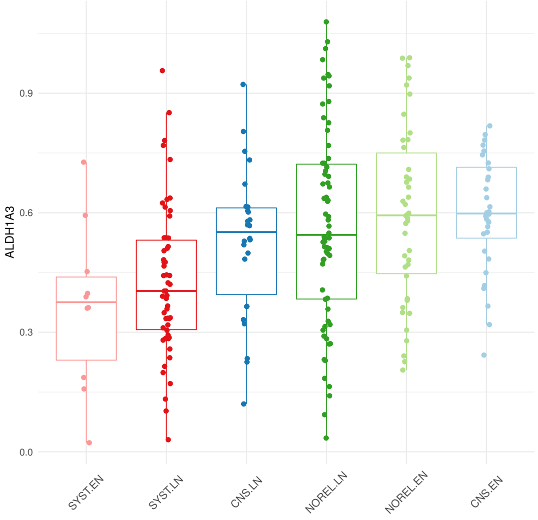


ALDH1A3

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 4, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 4, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 4, implev = 90)
```

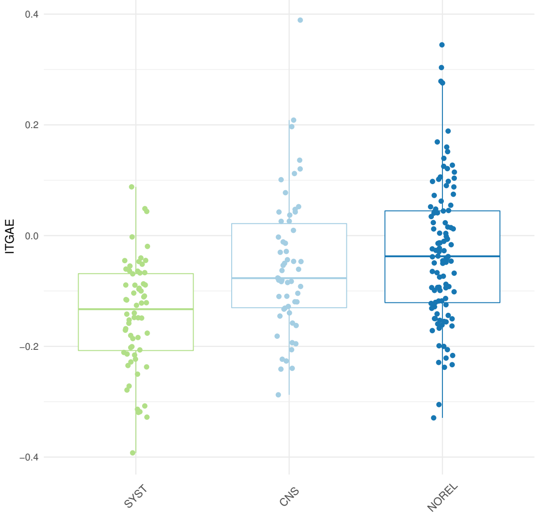


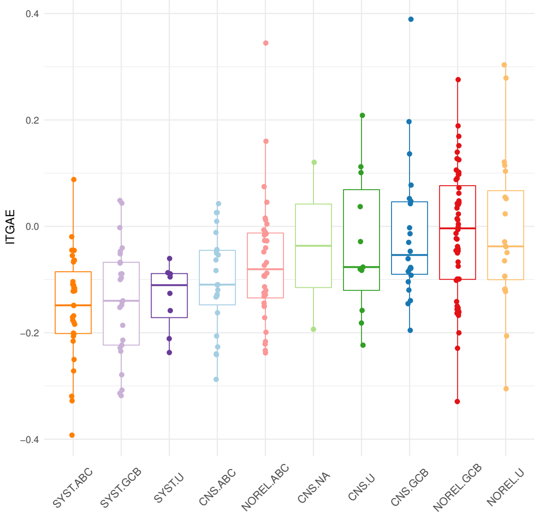


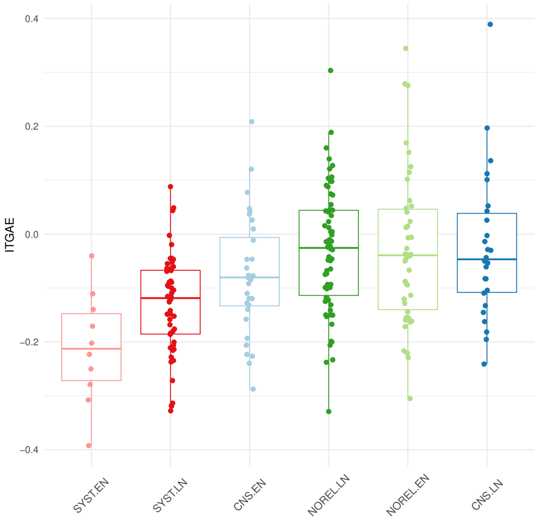


ITGAE

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 5, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 5, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 5, implev = 90)
```

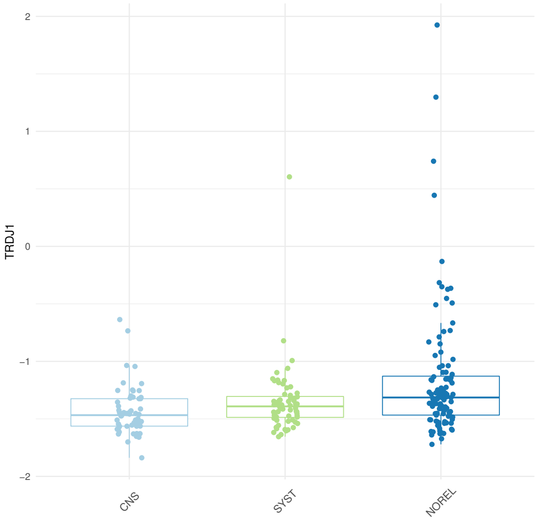


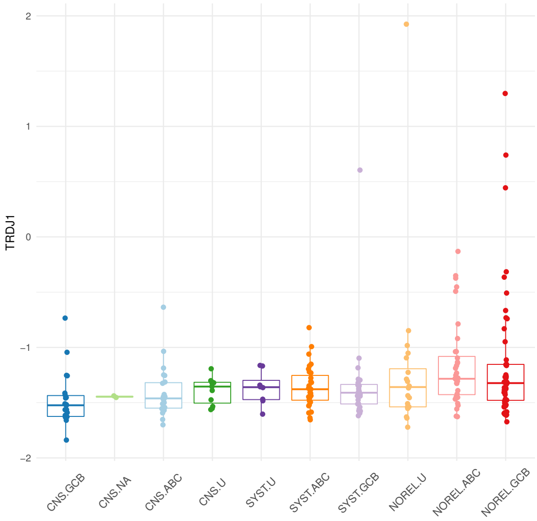


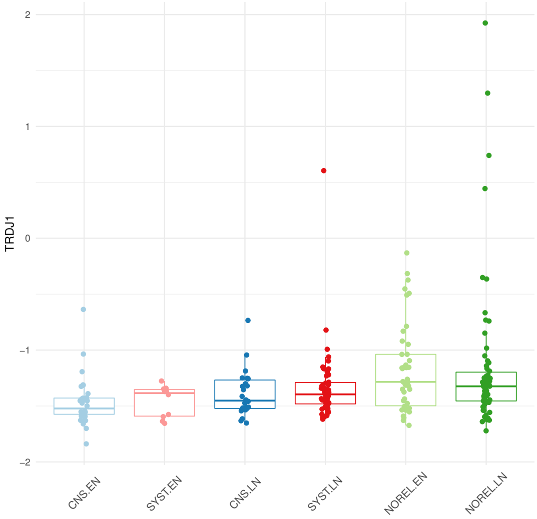


TRDJ1

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 6, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 6, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 6, implev = 90)
```

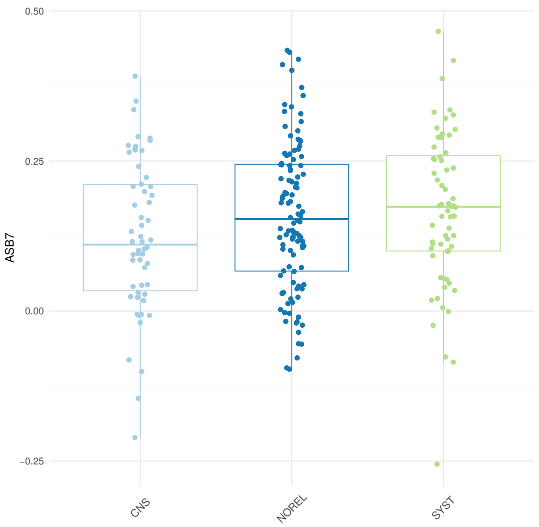


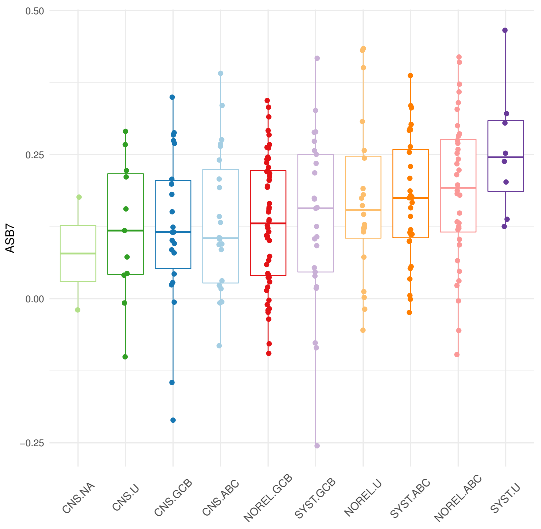


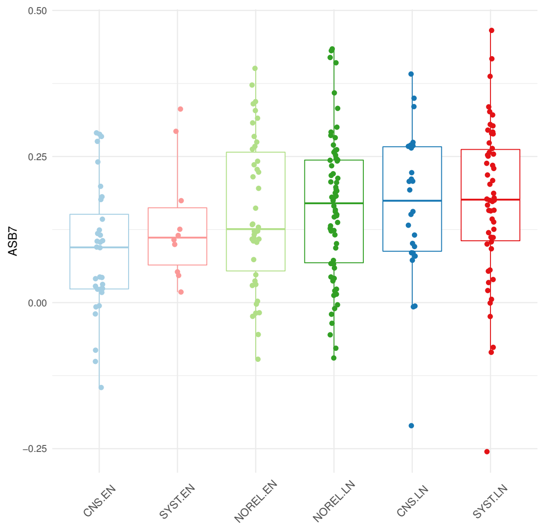


ASB7

```
gene.by.class(data = rma.genes, y = meta.selected$Groups, geneSel = 7, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast1, geneSel = 7, implev = 90)
gene.by.class(data = rma.genes, y = meta.selected$Contrast2, geneSel = 7, implev = 90)
```



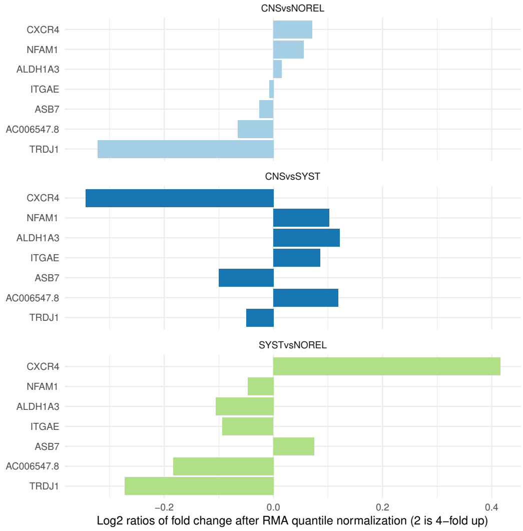




4.8 Expression of important genes summarized by limma scores

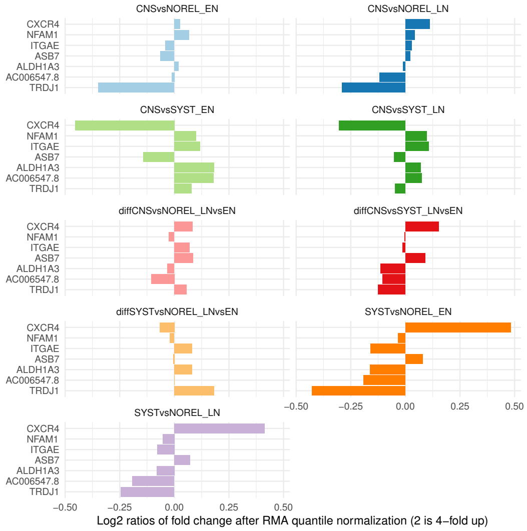
Expression based on log2 fold change for individuals compared based on a relapse contrast.

```
pvals <- read.table("./data/pvals.lmfit", header = TRUE)

pvals %>%
    filter(Contrast == "systemicRelapse") %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = reorder(genes, LogFC),
               y = LogFC,
               fill = factor(Comparison))) +
    geom_bar(stat = "identity",
             position = "dodge") +
    coord_flip() +
    scale_fill_brewer(palette="Paired") +
    facet_wrap(~ Comparison, ncol = 1) +
    theme_minimal() +
    theme(legend.position = "none") +
    xlab("") +
    ylab("Log2 ratios of fold change after RMA quantile normalization (2 is 4-fold up)")
```



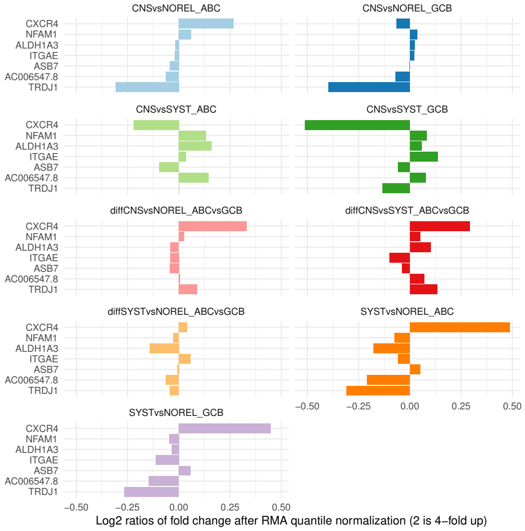
Expression based on log2 fold change for individuals compared based on involvement of lymphnodes or extra nodal sites.

```
pvals %>%
    filter(Contrast == "systemicRelapseNodes") %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = reorder(genes, LogFC),
               y = LogFC,
               fill = factor(Comparison))) +
    geom_bar(stat = "identity",
             position = "dodge") +
    coord_flip() +
    scale_fill_brewer(palette="Paired") +
    facet_wrap(~ Comparison, ncol = 2) +
    theme_minimal() +
    theme(legend.position = "none") +
    xlab("") +
    ylab("Log2 ratios of fold change after RMA quantile normalization (2 is 4-fold up)")
```



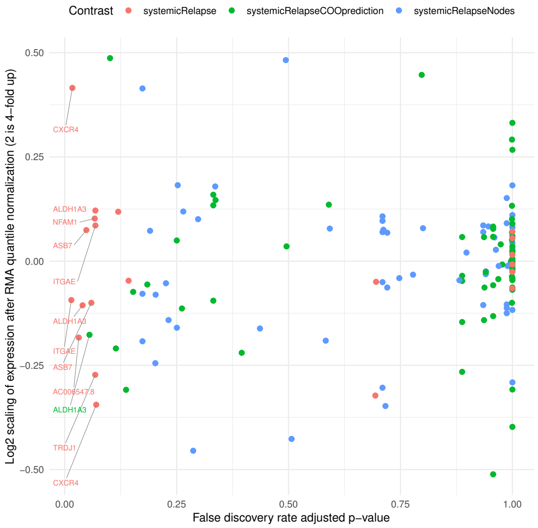
Expression based on log2 fold change for individuals compared based on involvement of cell-of-origin.

```
pvals %>%
    filter(Contrast == "systemicRelapseCOOprediction") %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    ggplot(aes(x = reorder(genes, LogFC),
               y = LogFC,
               fill = factor(Comparison))) +
    geom_bar(stat = "identity",
             position = "dodge") +
    coord_flip() +
    scale_fill_brewer(palette="Paired") +
    facet_wrap(~ Comparison, ncol = 2) +
    theme_minimal() +
    theme(legend.position = "none") +
    xlab("") +
    ylab("Log2 ratios of fold change after RMA quantile normalization (2 is 4-fold up)")
```



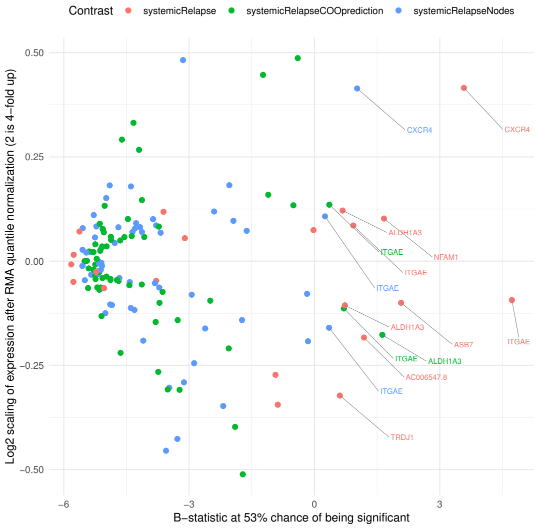
Expression based on log2fold and adjusted p-values. Labeled genes have a minimum of 0.1 adjusted p-value. Duplication in gene names is possible, because each contrast uses different comparisons between the groups of individuals.

```
model.names <- pvals %>%
    filter(FDRadjPval <= .1)

pvals %>%
    ggplot(aes(x = FDRadjPval,
               y = LogFC,
               color = Contrast,
               label = Symbol)) +
    geom_point(aes(color = Contrast),
#               shape = as.factor(Contrast)),
               size = 2) +
    geom_text_repel(data = subset(model.names, FDRadjPval <= .1),
                    nudge_x = -.5,
                    nudge_y = -.1,
                    force = 5,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2.5) +
    scale_fill_brewer(palette="Paired") +
    theme_minimal() +
    theme(legend.position = "top") +
    xlab("False discovery rate adjusted p-value") +
    ylab("Log2 scaling of expression after RMA quantile normalization (2 is 4-fold up)")
```



Expression based on log2fold and B-statistics. Labeled genes have a minimum of 0 B-statistic score (ie., 53% probability of being significant). Duplication in gene names is possible, because each contrast uses different comparisons between the groups of individuals.

```
model.names <- pvals %>%
    filter(B >= 0)

pvals %>%
    ggplot(aes(x = B,
               y = LogFC,
               color = Contrast,
               label = Symbol)) +
    geom_point(aes(color = Contrast),
#               shape = as.factor(Contrast)),
               size = 2) +
    geom_text_repel(data = subset(model.names, B >= 0),
                    nudge_x = 1.5,
                    nudge_y = -.1,
                    force = 5,
                    segment.color = "grey50",
                    direction = "y",
                    segment.size = 0.1,
                    size = 2.5) +
    scale_fill_brewer(palette="Dark2") +
    theme_minimal() +
    theme(legend.position = "top") +
    xlab("B-statistic at 53% chance of being significant") +
    ylab("Log2 scaling of expression after RMA quantile normalization (2 is 4-fold up)")
```



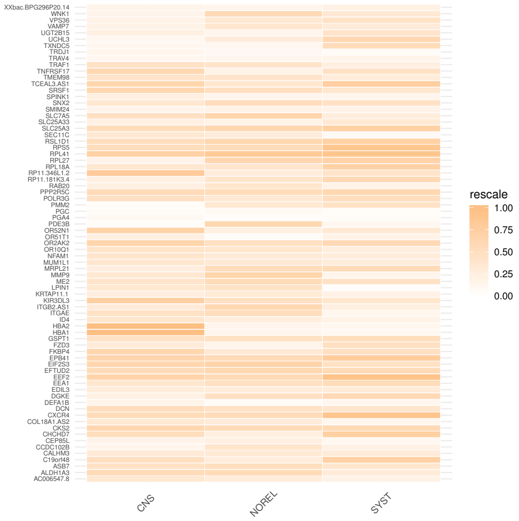
4.9 All classifying genes visualized by prognosis classes

Without clustering, visualizing the expression of all selected genes used for classification. Log2 FC was scaled to 0-1 for better interpretability.

4.9.1 Top classifier genes

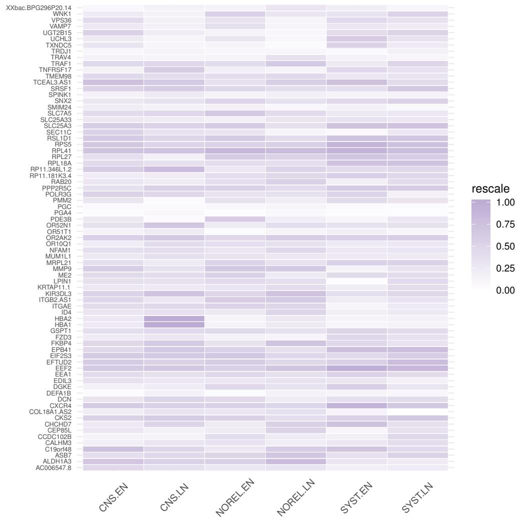
Gene log2 ratios for individuals when grouped according to CNS or systemic relapse, or no relapse.

```
data.frame(y=meta.selected$Groups, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "white", high = "#fdc086") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))
```



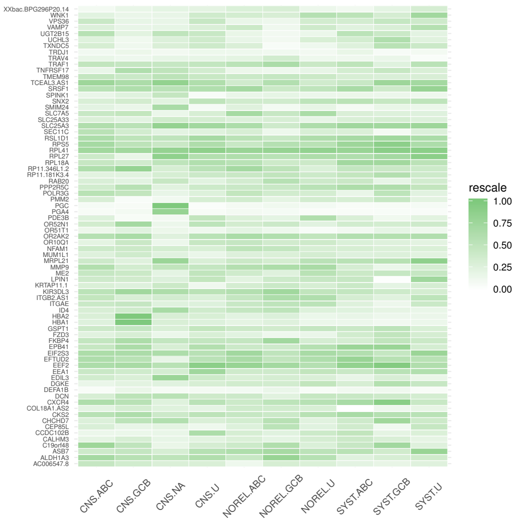
Gene log2 ratios for individuals when grouped according to lymphnodes or extranodal involvement.

```
data.frame(y=meta.selected$Contrast2, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "white", high = "#beaed4") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))
```



Gene log2 ratios for individuals when grouped according to cell-of-origin.

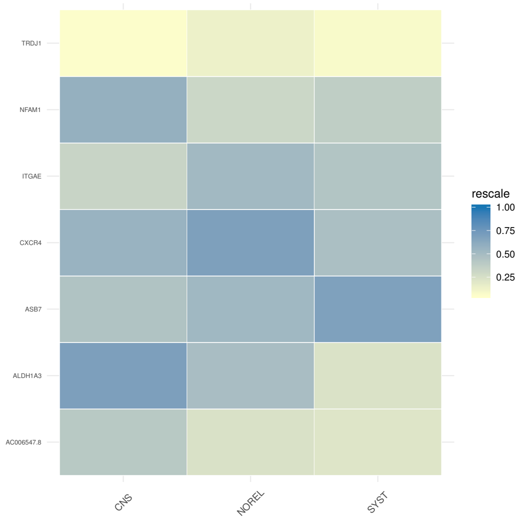
```
data.frame(y=meta.selected$Contrast1, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "white", high = "#7fc97f") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))
```



4.9.2 Top importance genes

The selected genes with highest importance score across models, showing log2 ratios for individuals when grouped according to relapse classes.

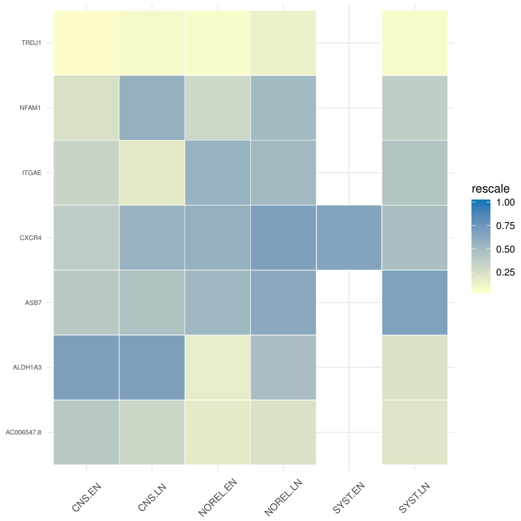
```
sel.genes <- imp.model %>%
    mutate(genes = gsub(".TC.*$","",genes)) %>%
    gather("groups", "importance", 3:5) %>%
    filter(importance >= 90) %>%
    select(genes) %>%
    unique

data.frame(y=meta.selected$Groups, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    filter(genes == sel.genes$genes) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "#ffffcc", high = "#1f78b4") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))

Warning in genes == sel.genes$genes:  longer object length is not a multiple of
shorter object length
```



The selected genes with highest importance score across models, showing log2 ratios for individuals when grouped according to nodal classes.

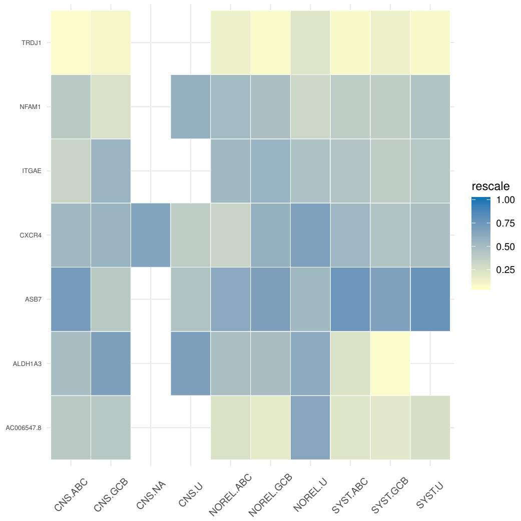
```
data.frame(y=meta.selected$Contrast2, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    filter(genes == sel.genes$genes) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "#ffffcc", high = "#1f78b4") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))
```

```
Warning in genes == sel.genes$genes:  longer object length is not a multiple of
shorter object length
```



The selected genes with highest importance score across models, showing log2 ratios for individuals when grouped according to cell-of-origin.

```
data.frame(y=meta.selected$Contrast1, t(rma.genes)) %>%
    melt() %>%
    ddply( .(variable), transform, rescale = rescale(value)) %>%
    mutate(genes = gsub(".TC.*$","",variable)) %>%
    filter(genes == sel.genes$genes) %>%
    ggplot(aes(y, genes)) +
    geom_tile(aes(fill = rescale), colour = "white") +
    scale_fill_gradient(low = "#ffffcc", high = "#1f78b4") +
    xlab("") + ylab("") +
    theme_minimal() +
    theme(axis.text.y = element_text(size = 6)) +
    theme(axis.text.x = element_text(vjust = .5,
                                     angle = 45,
                                     size = 9))

Warning in genes == sel.genes$genes:  longer object length is not a multiple of
shorter object length
```



4.10 Version of machine learning models on high performance clusters

↰ Version of R packages used in the machine learning pipeline

```
##############################################################################
```

```
## R version 3.5.0 (2018-04-23)                                                       ##
## Platform: x86_64-pc-linux-gnu (64-bit)                                             ##
## Running under: Ubuntu 16.04.4 LTS                                                  ##
##                                                                                    ##
## Matrix products: default                                                           ##
## BLAS: /usr/lib/openblas-base/libblas.so.3                                          ##
## LAPACK: /usr/lib/libopenblasp-r0.2.18.so                                           ##
##                                                                                    ##
## locale:                                                                            ##
##  [1] LC_CTYPE=en_CA.UTF-8       LC_NUMERIC=C                                       ##
##  [3] LC_TIME=en_CA.UTF-8        LC_COLLATE=en_CA.UTF-8                             ##
##  [5] LC_MONETARY=en_CA.UTF-8    LC_MESSAGES=en_CA.UTF-8                            ##
##  [7] LC_PAPER=en_CA.UTF-8       LC_NAME=C                                          ##
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C                                     ##
## [11] LC_MEASUREMENT=en_CA.UTF-8 LC_IDENTIFICATION=C                                ##
##                                                                                    ##
## attached base packages:                                                            ##
## [1] stats     graphics  grDevices utils     datasets  methods   base               ##
##                                                                                    ##
## other attached packages:                                                           ##
##  [1] pls_2.6-0          bindrcpp_0.2.2     plot3D_1.1.1       plyr_1.8.4           ##
##  [5] tidyr_0.8.1        reshape2_1.4.3     vegan_2.5-2        permute_0.9-4        ##
##  [9] earth_4.6.3        plotmo_3.4.0       TeachingDemos_2.10 plotrix_3.7-2        ##
## [13] ROCR_1.0-7         doSNOW_1.0.16      snow_0.4-2         iterators_1.0.9      ##
## [17] caret_6.0-80       ggplot2_3.0.0      lattice_0.20-35    glmnet_2.0-16        ##
## [21] foreach_1.4.4      Matrix_1.2-14      dplyr_0.7.6        gplots_3.0.1         ##
## [25] pvclust_2.0-0      RColorBrewer_1.1-2                                         ##
########################################################################################
```

```
########################################################################
##  [1] minqa_1.2.4         colorspace_1.3-2    class_7.3-14          ##
##  [4] DRR_0.0.3           svUnit_0.7-12       prodlim_2018.04.18    ##
##  [7] lubridate_1.7.4     codetools_0.2-15    splines_3.5.0         ##
## [10] mnormt_1.5-5        robustbase_0.93-0   RcppRoll_0.2.2        ##
## [13] mda_0.4-10          broom_0.4.4         ddalpha_1.3.3         ##
## [16] cluster_2.0.7-1     kernlab_0.9-26      sfsmisc_1.1-2         ##
## [19] compiler_3.5.0      assertthat_0.2.0    lazyeval_0.2.1        ##
## [22] FCNN4R_0.6.2        Rcgmin_2013-2.21    optextras_2016-8.8    ##
## [25] tools_3.5.0         igraph_1.2.1        misc3d_0.8-4          ##
## [28] gtable_0.2.0        glue_1.2.0          LiblineaR_2.10-8      ##
## [31] naivebayes_0.9.2    Rcpp_0.12.18        gdata_2.18.0          ##
## [34] nlme_3.1-137        setRNG_2013.9-1     psych_1.8.4           ##
## [37] timeDate_3043.102   gower_0.1.2         stringr_1.3.1         ##
## [40] kknn_1.3.1          gtools_3.5.0        DEoptimR_1.0-8        ##
## [43] Rvmmin_2018-4.17    MASS_7.3-50         scales_1.0.0          ##
## [46] ipred_0.9-6         parallel_3.5.0      monmlp_1.1.5          ##
## [49] rpart_4.1-13        stringi_1.2.2       ucminf_1.1-4          ##
## [52] randomForest_4.6-14 deepnet_0.2         e1071_1.6-8           ##
## [55] BB_2014.10-1        caTools_1.17.1      optimx_2013.8.7       ##
## [58] lava_1.6.1          geometry_0.3-6      rlang_0.2.1           ##
## [61] pkgconfig_2.0.1     bitops_1.0-6        purrr_0.2.5           ##
## [64] bindr_0.1.1         recipes_0.1.2       labeling_0.3          ##
## [67] CVST_0.2-2          tidyselect_0.2.4    gbm_2.1.3             ##
## [70] magrittr_1.5        R6_2.2.2            dimRed_0.1.0          ##
## [73] pillar_1.2.3        foreign_0.8-70      withr_2.1.2           ##
## [76] mgcv_1.8-23         survival_2.42-3     abind_1.4-5           ##
## [79] nnet_7.3-12         tibble_1.4.2        KernSmooth_2.23-15    ##
## [82] RRF_1.7             grid_3.5.0          ModelMetrics_1.1.0    ##
## [85] digest_0.6.15       dfoptim_2018.2-1    numDeriv_2016.8-1     ##
## [88] stats4_3.5.0        munsell_0.5.0       magic_1.5-8           ##
## [91] quadprog_1.5-5                                                 ##
########################################################################
```

5 System information for this report

The version number of R and packages loaded for generating the vignette were:

```
###save(list=ls(pattern=".*|.*"),file="PD.Rdata")
sessionInfo()
```

```
R version 3.4.4 (2018-03-15)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: elementary OS 0.4.1 Loki

Matrix products: default
BLAS: /usr/lib/libblas/libblas.so.3.6.0
LAPACK: /usr/lib/lapack/liblapack.so.3.6.0

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

other attached packages:
 [1] bindrcpp_0.2.2      reshape_0.8.7       ggridges_0.5.0
 [4] cowplot_0.9.3       ggpubr_0.1.7        magrittr_1.5
 [7] ggExtra_0.8         ggrepel_0.8.0       paletteer_0.1.0
[10] plyr_1.8.4          finalfit_0.7.4      Hmisc_4.1-1
[13] Formula_1.2-3       survival_2.42-6     brotools_0.2
[16] scales_1.0.0        DescTools_0.99.23   igraph_1.1.2
[19] tidyr_0.8.0         dplyr_0.7.6         ggplot2_3.0.0
[22] latticeExtra_0.6-28 RColorBrewer_1.1-2  lattice_0.20-35
[25] gdata_2.18.0        knitr_1.20

loaded via a namespace (and not attached):
 [1] splines_3.4.4     gtools_3.5.0      shiny_1.0.5
 [4] assertthat_0.2.0  expm_0.999-2      highr_0.6
 [7] pillar_1.1.0      backports_1.1.1   glue_1.3.0
[10] digest_0.6.12     checkmate_1.8.5   colorspace_1.3-2
[13] htmltools_0.3.6   httpuv_1.3.5      Matrix_1.2-11
[16] pkgconfig_2.0.2   purrr_0.2.4       xtable_1.8-2
[19] mvtnorm_1.0-7     manipulate_1.0.1  htmlTable_1.11.2
[22] tibble_1.4.2      withr_2.1.2       nnet_7.3-12
[25] lazyeval_0.2.1    mime_0.5          evaluate_0.10.1
[28] MASS_7.3-47       foreign_0.8-70    tools_3.4.4
[31] data.table_1.11.2 stringr_1.3.1     munsell_0.5.0
[34] cluster_2.0.7-1   compiler_3.4.4    rlang_0.2.2
[37] grid_3.4.4        rstudioapi_0.7    htmlwidgets_1.2
[40] miniUI_0.1.1.1    base64enc_0.1-3   labeling_0.3
[43] boot_1.3-20       gtable_0.2.0      reshape2_1.4.3
[46] R6_2.2.2          gridExtra_2.3     bindr_0.1.1
[49] stringi_1.2.2     Rcpp_0.12.18      rpart_4.1-13
[52] acepack_1.4.1     tidyselect_0.2.4
```