Example report of *OUKS*

This R Markdown document was provided as an example to reproduce the *OUKS* code script. Fig. 4 from article can be reproduced with the following code. Preliminarily download *"xcms after IPO MVI QC-XGB filter repeats annot+filtr LMM adj KEGG.csv"* and *"8 peaks.csv"* files into your working directory (for example: ("D:/OUKS/").

Prepare environment

First, set the folder for the working directory and load the packages.

```
setwd("D:/OUKS/")

library(data.table)

## Warning: package 'data.table' was built under R version 4.0.2

library(factoextra)

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'tibble':
##   method     from
##   format.tbl pillar
##   print.tbl  pillar

## Welcome! Want to learn more? See two factoextra-related books at https://goo.
gl/ve3WBa

library(FactoMineR)
library(dendextend)

##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili
/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendex
tend))
## ---------------------

##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:data.table':
##
##     set

## The following object is masked from 'package:stats':
##
##     cutree

library(rafalib)
library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

library(parallel)
library(doParallel)

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 4.0.3

## Loading required package: iterators

## Warning: package 'iterators' was built under R version 4.0.3

library(grid)
library(caret)

## Loading required package: lattice

library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(MKinfer)

## Warning: package 'MKinfer' was built under R version 4.0.2
```

```
library(limma)
library(ggplot2)
library(cowplot)

##
## ************************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## ************************************************************

library(ggsci)
```

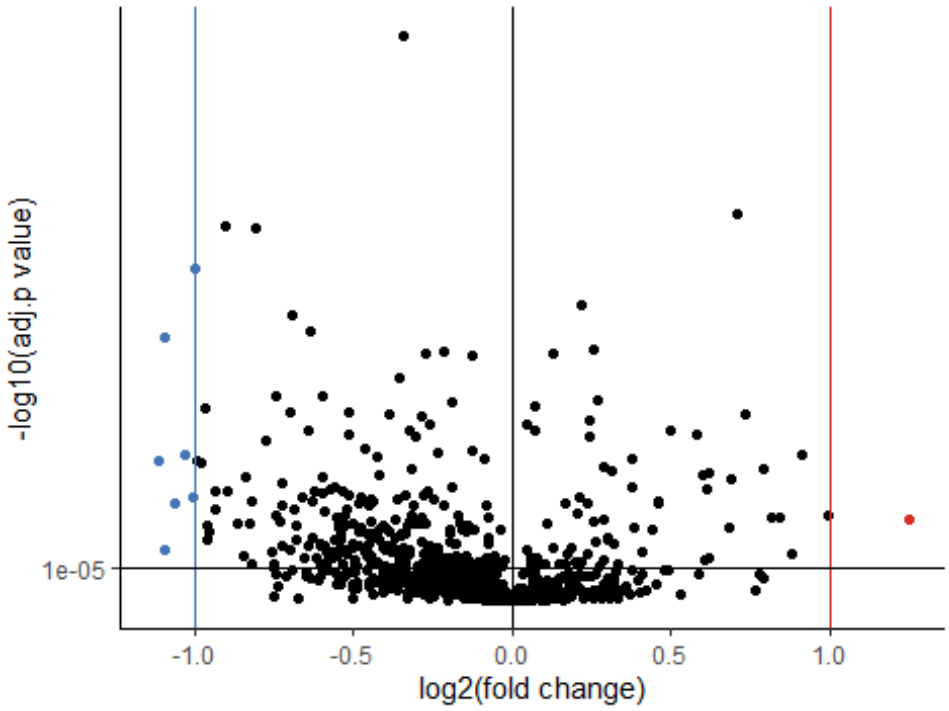
Load datasets

```
ds <- as.data.frame(fread(input = "8 peaks.csv", header=T))
rownames(ds) <- ds[,1]
ds <- ds[,-1]
colnames(ds)[1] <-"Label"
ds[,-1] <- sapply(ds[,-1], as.numeric)
ds$Label <- as.factor(ds$Label)

ds2 <- as.data.frame(fread(input = "xcms after IPO MVI QC-XGB filter repeats ann
ot+filtr LMM adj KEGG.csv", header=T))
ds2 <- ds2[-c(1:12),]
rownames(ds2) <- ds2[,5]
ds2 <- ds2[,-c(1,3:5)]
ds2[,-1] <- sapply(ds2[,-1], as.numeric)
ds2$Label <- as.factor(ds2$Label)
```

Plot volcano plot

```
ds_log <- as.data.frame(log2(ds2[,-1]))
ds_log <- cbind(Label = ds2[,1], ds_log)

FOLD.CHANGE <- function(data) {
  ds_log_subsets <- lapply(1:length(unique(data[,1])), function(y) filter(data[,
-1], data$Label == unique(data[,1])[y]))
  mean_r_l <- lapply(1:length(ds_log_subsets), function(y) apply(ds_log_subsets[
[y]], 2, mean, na.rm = T))
  foldchange <- mean_r_l[[1]] - mean_r_l[[2]]
  fc_res <- as.data.frame(foldchange)
  return(fc_res)
}

fc_res <- FOLD.CHANGE(ds_log)
foldchange <- as.numeric(fc_res$foldchange)

mdl_mtrx <- model.matrix(~Label, ds2)
lmf <- lmFit(t(ds2[,-1]), method = "robust", design = mdl_mtrx, maxit = 1000)
```

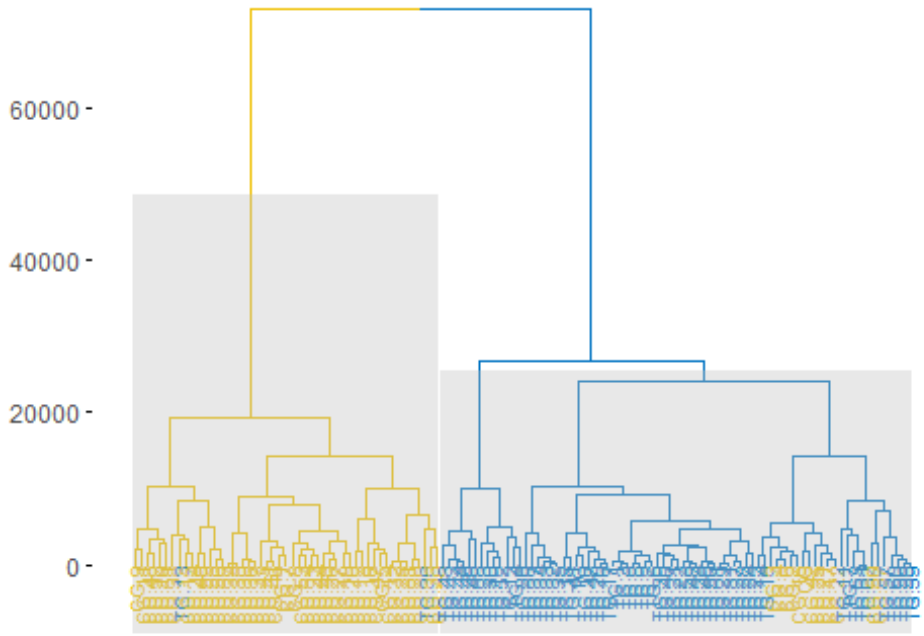
```
efit <- eBayes(lmf)
tableTop <- topTable(efit, coef = 2, adjust = "BH",  number = ncol(ds2), sort.by
= "none")
pval <- as.numeric(tableTop$adj.P.Val)

f <- volcano(foldchange, pval, effect.low = -1.0, effect.high = 1.0, sig.level =
0.00001,
             xlab = "log2(fold change)", ylab = "-log10(adj.p value)",  title =
"") + theme_classic() +  theme(legend.position="none")
f
```



HCA plot

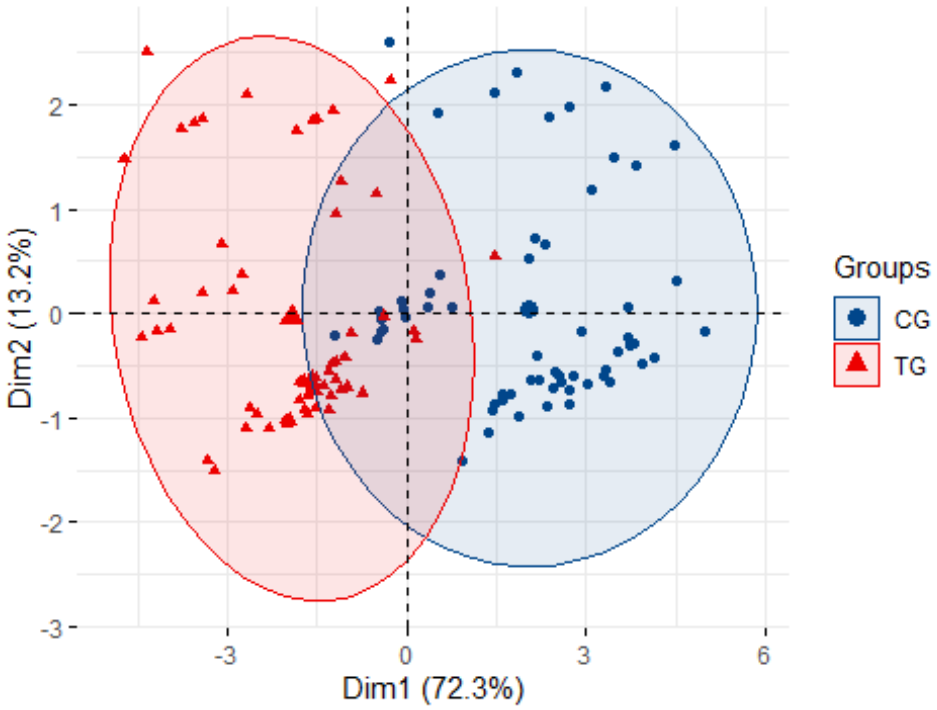
```
base1 <- ds
mtrx1 <- ds[,-1]
grp1 <- as.character(base1[,1])

k <- length(unique(grp1))
Cols = function(vec, ord){
  cols = pal_jco(palette = c("default"), alpha = 1)(length(unique(vec)))
  return(cols[as.fumeric(vec)[ord]])}
mtrx1_1 <- mtrx1
rownames(mtrx1_1) = make.names(grp1, unique=TRUE)
res.dist1 <- dist(mtrx1_1, method = "manhattan")
res.hc1 <- hclust(d = res.dist1, method = "ward.D2")

b <- fviz_dend(res.hc1, k = k,
                 cex = 0.55,
                 k_colors = unique(Cols(grp1,res.hc1$order)),
```

```
                  color_labels_by_k = F,
                  label_cols = Cols(grp1,res.hc1$order),
                  rect = T,
                  rect_fill = T,
                  horiz = F,
                  lwd = 0.7,
                  show_labels = T,
                  main = "",
                  ylab = "")
b
```



PCA plot

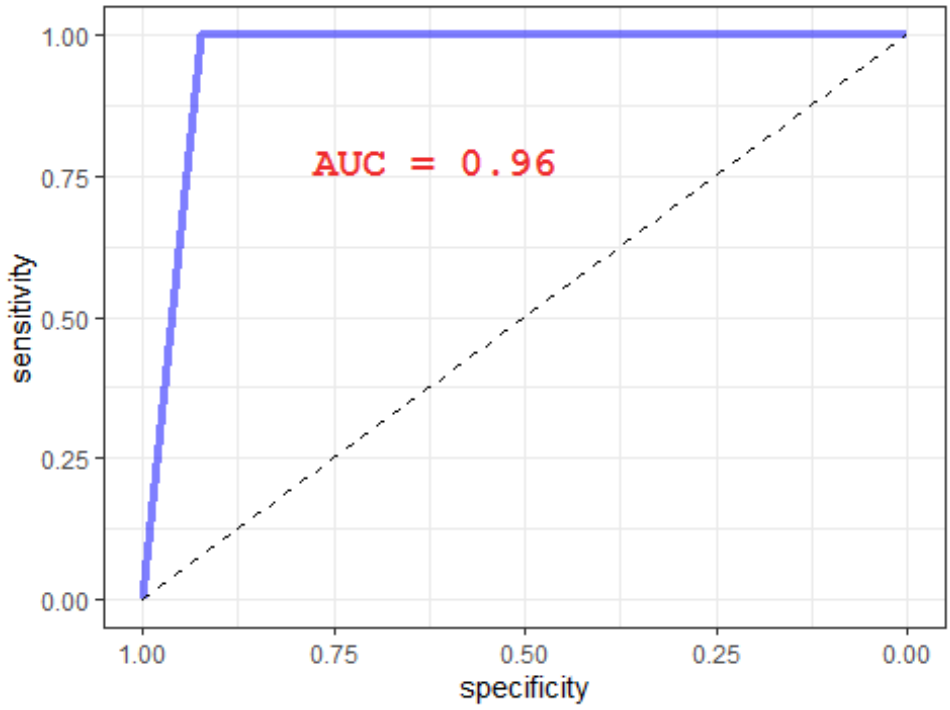
```
base1 <- ds
mtrx1 <- ds[,-1]
grp1 <- as.character(base1[,1])
palette_pca <- "lancet"

pca.ds1 <- PCA(mtrx1, scale.unit = T, graph = F)
a <- fviz_pca_ind(pca.ds1,
                  title = "",
                  geom.ind = "point",
                  col.ind = grp1,
                  palette = palette_pca,
                  addEllipses = T,
                  legend.title = "Groups")
a
```



ROC curve

```
# start parallel processing
fc <- as.numeric(detectCores(logical = F))
cl <- makePSOCKcluster(fc+1)
registerDoParallel(cl)

# cross-validation
set.seed(1234)
trainIndex <- createDataPartition(ds$Label, p = 0.8, list = F, times = 1)
dsTrain <- ds[ trainIndex,]
dsValid <- ds[-trainIndex,]
trainControl <- trainControl(method="repeatedcv", number=10, repeats=10, classPr
obs = F)
metric <- "Accuracy"

# machine learning
set.seed(1234)
fit.cl <- train(Label~., data=dsTrain, method="svmRadial", metric=metric, trCont
rol=trainControl, tuneLength = 10)
predicted.classes <- predict(fit.cl, newdata=dsValid)

# ROC curve
res.roc <- roc(as.numeric(as.factor(predicted.classes)), as.numeric(dsValid$Labe
l))

## Setting levels: control = 1, case = 2

## Setting direction: controls < cases
```

```
auroc <- round(as.numeric(auc(res.roc)),2)
grob <- grobTree(textGrob(paste0("AUC = ", auroc), x=0.25,  y=0.75, hjust=0,
                          gp=gpar(col="firebrick2", fontsize=15, fontface=11)))
c <- ggroc(res.roc, alpha = 0.5, colour = "blue1", linetype = 1, size = 1.5) +th
eme_bw() + ggtitle("") +
  geom_segment(aes(x = 1, xend = 0, y = 0, yend = 1), color="black", linetype="d
ashed")  + annotation_custom(grob)
c
```



bootstrap histogram

```
# start parallel processing
fc <- as.numeric(detectCores(logical = F))
cl <- makePSOCKcluster(fc+1)
registerDoParallel(cl)

# bootstrap
set.seed(1234)
trainControl <-trainControl(method="boot", number=1000)
metric <- "Accuracy"

# machine learning
set.seed(1234)
fit.cl <- train(Label~., data=ds, method="svmRadial", metric=metric, trControl=t
rainControl)
results <- resamples(list(svm=fit.cl, svm1=fit.cl), trControl = trainControl, me
tric=metric)

# histogram
```

```
d <-ggplot(results$values, aes(x=results$values[,2])) +
  geom_histogram(colour="blue", fill="white") + theme_bw() + xlab("Accuracy") +
ylab("Frequency")
d

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```