```r
# Codes edited by Hui Han are following advices from Martin Schweinberger
# clean current workspace
rm(list=ls(all=T))
options(stringsAsFactors = F)         # no automatic data transformation
options("scipen" = 100, "digits" = 4) # suppress math annotation
# install libraries
install.packages(c("Boruta", "caret", "cowplot", "dplyr",
                   "ggplot2", "Gmisc", "grid", "Hmisc",
                   "knitr", "party", "partykit", "randomForest",
                   "tidyr", "stringr", "Rling", "DT", "RCurl"))
# activate libraries
library(partykit)
library(dplyr)
library(grid)
library(Gmisc)
library(ggplot2)
library(cowplot)
library(randomForest)
library(party)
library(Hmisc)
library(Boruta)
library(RCurl)
# set options
options(stringsAsFactors = F)
options(scipen = 999)
options(max.print=10000)
# install caret library
```

```
source("https://bioconductor.org/biocLite.R"); biocLite(); library(Biobase)

install.packages("Biobase", repos=c("http://rstudio.org/_packages",
"http://cran.rstudio.com",  "http://cran.rstudio.com/", dependencies=TRUE))

install.packages("dimRed", dependencies = TRUE)

install.packages('caret', dependencies = TRUE)

# activate caret library

library(caret)

# set the working directory (where the data are located)

setwd("/Users/huihan/Desktop/RF/Dataset")

# load data

rfdata <- read.csv("DataR_Cluster.csv", header=T, row.names=NULL, na.strings="")

names(rfdata)

# factorize variables (cit require factors instead of character vectors)

fcts <- c("SaaS", "PuC", "PrC", "CC", "HC", "ComS", "IS", "ConS", "ST", "LMST",
"SNT", "S4ML", "Sc", "Se", "Re", "IO", "EN", "UN", "CN", "LN", "PN", "IN",
"Cluster")

rfdata[fcts] <- lapply(rfdata[fcts], factor)
```

Gini index

```
# set.seed

set.seed(2019120205)

rfmodel2 <- randomForest(Cluster ~., data=rfdata, proximity=TRUE)

# plot result

varImpPlot(rfmodel2, main = "Importance of variables", pch = 20)

# check what mtry is optimal

tuneRF(rfdata[, !colnames(rfdata)== "Cluster"],

    rfdata[, colnames(rfdata)== "Cluster"],

    stepFactor = 4, # 4 has the lowest value to be optimal

    plot = T, ntreeTry = 500, trace = T, improve = .05)
```

```
# set.seed (to store random numbers and thus make results reproducible)

set.seed(2019120206)

# create a new model with same trees and that takes 1 variable at a time

rfmodel3 <- randomForest(Cluster ~ ., data=rfdata, ntree = 500, mtry = 22,
proximity=TRUE)

# inspect model

rfmodel3


# set.seed (to store random numbers and thus make results reproducible)

set.seed(2019120206)

# create a new model with same trees and that takes 1 variable at a time

rfmodel3 <- randomForest(Cluster ~ ., data=rfdata, ntree = 64, mtry = 22,
proximity=TRUE)

 # inspect model

rfmodel3


# set.seed (to store random numbers and thus make results reproducible)

set.seed(2019120206)

# create a new model with same trees and that takes 1 variable at a time

rfmodel3 <- randomForest(Cluster ~ ., data=rfdata, ntree = 128, mtry = 22,
proximity=TRUE)

# inspect model

rfmodel3


set.seed(2019120206)

# create a new model with same trees and that takes 1 variable at a time

rfmodel3 <- randomForest(Cluster ~ ., data=rfdata, ntree = 500, mtry = 22,
proximity=TRUE)

# inspect model
```

```
rfmodel3

# save what the model predicted in a new variable
rfdata$Prediction <- predict(rfmodel3, rfdata)
# create confusion matrix to check accuracy
confusionMatrix(rfdata$Prediction, rfdata$Cluster)
# plot variable importance
varImpPlot(rfmodel3, main = "Importance of variables", pch = 20)
# extract importance of individual variables
partialPlot(rfmodel3, rfdata, PuC)
partialPlot(rfmodel3, rfdata, IN)
```

Permutation importance

```
## load the library
install.packages("readxl")
library(readxl)
library(randomForest)
library(dplyr)
# Here we read in the data set
dat <- read.csv("DataR_small.csv", header=T, row.names=NULL, na.strings="")
names(dat)
# factorize variables (require factors instead of character vectors)
fcts <- c("SaaS", "PuC", "PrC", "CC", "HC", "ComS", "IS", "ConS", "ST", "LMST",
"SNT", "S4ML", "Sc", "Se", "Re", "IO", "EN", "UN", "CN", "LN", "PN", "IN")
dat[fcts] <- lapply(dat[fcts], factor)
#The unsupervised Random Forest algorithm was used to generate a proximity matrix
using all listed clinical variables
set.seed(23)
n <- nrow(dat)
```

```
datBS <- mutate_all(dat,funs(sample(.,replace=TRUE)))

y <- factor(c(rep(1, n), rep(2, n)))

rf<-randomForest(x=rbind(dat,datBS), y=y, proximity=TRUE)

varImpPlot(rf)


#rfPermute estimates the significance of importance metrics for a Random Forest model
by permuting the response variable. It will produce null distributions of importance
metrics for each predictor variable and p-value of observed.

install.packages("rfPermute")

if (!require('devtools')) install.packages('devtools')

library(rfPermute)

rfP <- rfPermute(x=rbind(dat,datBS), y=y, ntree = 100, na.action = na.omit, nrep = 50,
num.cores = 1)

plotNull(rfP)

plotImportance(rfP, scale = TRUE)


#We can now create our initial Boruta model and set a seed for reproducibility.

borutadata <- read.table("DataWeka_Cluster.csv", sep=",", header=T,
row.names=NULL, , na.strings="")


# factorize variables (boruta require factors instead of character vectors)

fcts <- c("SaaS", "PuC", "PrC", "CC", "HC", "ComS", "IS", "ConS", "ST", "LMST",
"SNT", "S4ML", "Sc", "Se", "Re", "IO", "EN", "UN", "CN", "LN", "PN", "IN",
"Cluster")

borutadata[fcts] <- lapply(borutadata[fcts], factor)

#We can now create our initial Boruta model and set a seed for reproducibility.

# set.seed

set.seed(2019120207)

# initial run

boruta1 <- Boruta(Cluster~.,data=borutadata)
```

```
print(boruta1)


# extract decision
getConfirmedFormula(boruta1)
```

In a next step, we inspect the history to check if any variables show drastic fluctuations in their importance assessment.

```
plotImpHistory(boruta1)


# remove irrelevant variables
rejected <- names(boruta1$finalDecision)[which(boruta1$finalDecision == "Rejected")]
# update data for boruta
borutadata <- borutadata %>%
  dplyr::select(-rejected)
# set.seed (to store random numbers and thus make results reproducible)
set.seed(2019120208)
# 2nd run
boruta2 <- Boruta(Cluster~.,data=borutadata)
print(boruta2)
# extract decision
getConfirmedFormula(boruta2)


library(ggplot2)
library(tidyr)
library(stringr)
borutadf <- as.data.frame(boruta2$ImpHistory) %>%
  gather(Variable, Importance, PuC:shadowMin) %>%
  mutate(Type = ifelse(str_detect(Variable, "shadow"), "Control", "Predictor")) %>%
  mutate(Type = factor(Type),
         Variable = factor(Variable))
```

```
ggplot(borutadf, aes(x = reorder(Variable, Importance, mean), y = Importance, fill =
Type)) +
  geom_boxplot() +
  geom_vline(xintercept=3.5, linetype="dashed", color = "black") +
  scale_fill_manual(values = c("gray80", "gray40")) +
  theme_bw() +
  theme(legend.position = "top",
        axis.text.x = element_text(angle=90)) +
  labs(x = "Variable")
```

#Codes edited by Hui Han are following advices from Tao Shi and Steve Horvath

```
# set the working directory (where the data are located)

setwd("/Users/huihan/Desktop/RF/Dataset")

## load the library

source("FunctionsRFclustering.txt")

# Here we read in the data set

mydata= read.csv("DataR.csv", header=T, row.names=NULL, na.strings="")

## the objects that will be clustered (number part).

datRF = mydata[,18:23]

attach(datRF)

names(datRF)

## Calculating RF distance between samples based on the above three measurements

## This will take quite long time depends how many tree and repetitions you choose

## We suggest to use relatively large number of forests with large number of trees

# Since we are mainly interested in the Addcl1 RF dissimilarity we set addcl1=T,
addcl2=F

# imp=T specifies that we are also interested in the importance measures.

no.trees=128

no.forests=10

datRF <- na.omit(datRF)

distRF = RFdist(datRF, mtry1=3, no.trees, no.forests, addcl1=T,addcl2=F,imp=T,
oob.prox1=T)

## PAM clustering based on the Addcl1 RF dissimilarity

no.clusters = 2

labelRF = pamNew(distRF$cl1, no.clusters)

## PAM clustering based on Euclidean distance

labelEuclid = pamNew(dist(datRF), no.clusters)

## Due to the randomness of RF procedure, the exact distance measure will vary a bit

## Therefore, we also include our RF clustering result in our data
## Check the agreement between RF cluster and Euclidean distance cluster

fisher.test(table(labelRF, labelEuclid)) ## Fisher’s exact p value

## Using rpart tree the dissect the relationship between RF clusters and markers expression value

## need library 'rpart'

library(rpart)

rp1 = rpart(factor(labelRF)~., datRF)

plot(rp1); text(rp1, all=T, use.n=T, cex=0.9)

summary(rp1)

## We can use two markers respectively to explain the RF clusters

plot(jitter(CN,5),jitter(IN,5), col= c("blue", "orange"), cex=1.5, xlab="CN", ylab="IN")

abline(h=81.67); abline(v=67.5)


plot(jitter(CN,5),jitter(UN,5), col= c("blue", "orange"), cex=1.5, xlab="CN", ylab="UN")

abline(h=81.67); abline(v=67.5)


plot(jitter(EN,5),jitter(IN,5), col= c("blue", "orange"), cex=1.5, xlab="EN", ylab="IN")

abline(h=81.67); abline(v=67.5)


plot(jitter(PN,5),jitter(LN,5), col= c("blue", "orange"), cex=1.5, xlab="PN", ylab="LN")

abline(h=81.67); abline(v=67.5)


## Classical multidimensional scaling based on RF distance

# we use 2 scaling dimensions.

cmd1 = cmdscale(as.dist(distRF$cl1),2)

## Due to the randomness of the RF clustering algorithm, you may get slightly different results

plot(cmd1,xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
```

```
#In the following, we will use PAM clustering on the 2 scaling coordinates

## PAM clustering based on the scaling coordinates

RFclusterLabel = pamNew(cmd1, 2)

# See the following plot.

# Classical MDS plot

# Clear Online platforms “SaaS” are labeled by “S” and non-supports mobile learning are
labelled by “N”

# Online platforms for higher education are colored by their RF clustering memberships

plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")

text(cmd1, label=ifelse(mydata$ SaaS==1, "S", "N"), col= RFclusterLabel)


# Clear Online platforms using “PuC” are labeled by “PuC” and non-“ PuC” are labelled
by “N”

# Online platforms for higher education are colored by their RF clustering memberships

plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")

text(cmd1, label=ifelse(mydata$ PuC==1, "PuC", "N"), col= RFclusterLabel)


# Clear Online platforms with “IO” are labeled by “I” and non-“IO” are labelled by “N”

# Online platforms for higher education are colored by their RF clustering memberships

plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")

text(cmd1, label=ifelse(mydata$IO==1, "I", "N"), col= RFclusterLabel)


# Clear Online platforms using “ST” are labeled by “ST” and non-“ST” are labelled by
“N”

# Online platforms for higher education are colored by their RF clustering memberships

plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")

text(cmd1, label=ifelse(mydata$ST==1, " ST", "N"), col= RFclusterLabel)


# Clear Online platforms using “PrC” are labeled by “PrC” and non-“PrC” are labelled by
“N”
```

```
# Online platforms for higher education are colored by their RF clustering memberships
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$PrC==1, " PrC", "N"), col= RFclusterLabel)

# Clear Online platforms using “CC” are labeled by “CC” and non-“CC” are labelled by “N”
# Online platforms for higher education are colored by their RF clustering memberships
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$CC==1, " CC", "N"), col= RFclusterLabel)

# Clear Online platforms using “HC” are labeled by “HC” and non-“HC” are labelled by “N”
# Online platforms for higher education are colored by their RF clustering memberships
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$HC==1, " HC", "N"), col= RFclusterLabel)

# Clear Online platforms using “ComS” are labeled by “ComS” and non-“ComS” are labelled by “N”
# Online platforms for higher education are colored by their RF clustering memberships
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$ComS==1, " ComS", "N"), col= RFclusterLabel)

# Clear Online platforms using “IS” are labeled by “IS” and non-“IS” are labelled by “N”
# Online platforms for higher education are colored by their RF clustering memberships
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$IS==1, " IS", "N"), col= RFclusterLabel)

# Clear Online platforms using “ConS” are labeled by “ConS” and non-“ConS” are labelled by “N”
# Online platforms for higher education are colored by their RF clustering memberships
```

```
plot(cmd1, type="n", xlab="Scaling Dimension 1",ylab="Scaling Dimension 2")
text(cmd1, label=ifelse(mydata$ConS==1, " ConS", "N"), col= RFclusterLabel)
```