```
#=========================================================

Section1  BLCA mRNA cluster analysis

#=========================================================
#load data , delete useless genes
setwd("E:\\Rwork")

require(dplyr)
require(tidyr)
load("mRNA_exprSet.Rda")
mRNA <- mRNA_exprSet
index <- duplicated(mRNA$gene_name)
mRNA.data <- mRNA [!index,]
dim(mRNA.data)

#=========================================================
#select tumor data
index <- as.numeric(substr(colnames(mRNA.data),start = 14, stop = 15))
mRNA.data1 <-  mRNA.data[,which(index == 1)]
mRNA.data <- cbind(mRNA.data[,1],mRNA.data1 )

write.csv(mRNA.data, file = "BLCAmRNA.FPKM.csv", row.names = F)

#=========================================================
# log2 and delete low expression genes

rt <-   read.csv("BLCAmRNA.FPKM.csv",header = T,row.names = 1)
rt=log2(rt[,1:ncol(rt)]+1)
flag <- apply(rt, 1, function(x) sum( x == 0) < 250)
rt <- rt[which(flag),]

#=========================================================
# delete  genes by row mean

means <- apply(rt,1,mean)
rt_mean <- rt[rev(order(means))[1:10000],]
dim(rt_mean)

#=========================================================
# delete  genes by row SD

mads=apply(rt_mean,1,mad)
rt_mads = rt_mean[rev(order(mads))[1:3000],]
```

```
#=====================================================
#apply ConsensusCluster analysis

rt_nor=sweep(rt_mads,1, apply(rt_mads,1,median,na.rm=T)) #
library(ConsensusClusterPlus)
title=tempdir()
title="77"
rt_nor <- as.matrix(rt_nor)
results = ConsensusClusterPlus(rt_nor,maxK=6,reps=50,pItem=0.8,pFeature=1,
                               clusterAlg="hc",
                               distance="pearson",
                               innerLinkage="complete",
                               seed=1262118388.71279,
                               plot="pdf")
#save the result
results2 <-results[[2]]["consensusClass"]
results3 <-results[[3]]["consensusClass"]
results4 <-results[[4]]["consensusClass"]
results5 <-results[[5]]["consensusClass"]
results6 <-results[[6]]["consensusClass"]
as.data.frame(results2)
as.data.frame(results3)
as.data.frame(results4)
as.data.frame(results5)
as.data.frame(results6)
new<-data.frame(results2,results3,results4,results5,results6)
write.csv(new,file = "BLCAmRNAcluster.csv")




#=====================================================

Section2  BLCA miRNA cluster analysis


#=====================================================
#load data, delete useless genes such as duplicated genes

setwd("E:\\Rwork")
miRNA<- read.table("BLCA.miRseq_mature_RPM.txt",header = T,
```

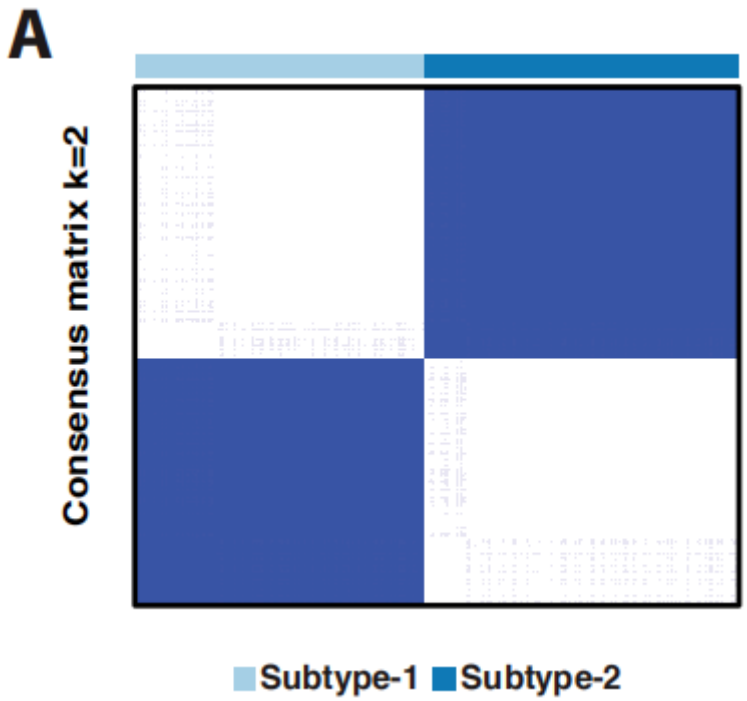
```
                    stringsAsFactors = F, quote = "")



index <- duplicated(miRNA$Gene)
miRNA.data <- miRNA [!index,]

#====================================================
#select expression data of tumor samples from normal samples

index <- as.numeric(substr(colnames(miRNA.data),start = 14, stop = 15))
miRNA.data1 <-  miRNA.data[,which(index == 1)]
miRNA.data <- cbind(miRNA.data[,1],miRNA.data1 )
rt <- miRNA.data

#====================================================
# log2 and delete low expression genes

rt [is.na(rt)] <- 0
flag <- apply(rt, 1, function(x) sum( x == 0) < 300)
rt <- rt[which(flag),]
rt=log2(rt[,1:ncol(rt)]+1)
fix(rt)

#====================================================
# delete 25% genes by row mean

means <- apply(rt,1,mean)
rt_mean <- rt[rev(order(means))[1:400],]
dim(rt_mean)
#====================================================
# delete 25% genes by row mad

mads=apply(rt_mean,1,mad)
rt_mads = rt_mean[rev(order(mads))[1:300],]
rt_nor=sweep(rt_mads,1, apply(rt_mads,1,median,na.rm=T))

#====================================================
#ConsensusCluster was uterlized

library(ConsensusClusterPlus)
title=tempdir()
title="miRNA"
rt_nor <- as.matrix(rt_nor)
```

```
results = ConsensusClusterPlus(rt_nor,maxK=6,reps=50,pItem=0.8,pFeature=1,
                               clusterAlg="hc",
                               distance="pearson",
                               innerLinkage="complete",
                               seed=1262118388.71279,
                               plot="pdf")    figure separately

#=======================================================

#=======================================================
#save the result
results2 <-results[[2]]["consensusClass"]
results3 <-results[[3]]["consensusClass"]
results4 <-results[[4]]["consensusClass"]
results5 <-results[[5]]["consensusClass"]
results6 <-results[[6]]["consensusClass"]
as.data.frame(results2)
as.data.frame(results3)
as.data.frame(results4)
as.data.frame(results5)
as.data.frame(results6)
new<-data.frame(results2,results3,results4,results5,results6)
write.csv(new,file = "BLCAmiRNAcluster.csv")

#=======================================================




#=======================================================


Section3  BLCA lncRNA cluster analysis


#=======================================================
#load data , delete useless genes
setwd("E:\\Rwork")

require(dplyr)
require(tidyr)
load("LncRNA_exprSet.Rda")
LncRNA <- LncRNA_exprSet
```

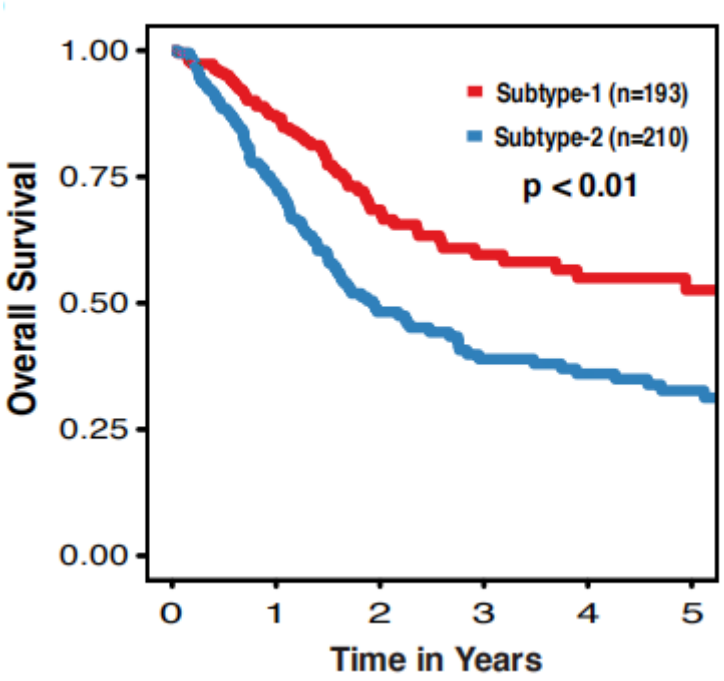
```
index <- duplicated(LncRNA$gene_name)
LncRNA.data <- LncRNA[!index,]

#=========================================================
#select tumor data

index <- as.numeric(substr(colnames(LncRNA.data),start = 14, stop = 15))
LncRNA.data1 <-  LncRNA.data[,which(index == 1)]
LncRNA.data <- cbind(LncRNA.data[,1],LncRNA.data1 )

#=========================================================
# log2 and delete low expression genes

rt <-   read.csv("BLCALncRNA.FPKM.csv",
                 header = T,row.names = 1)
rt=log2(rt[,1:ncol(rt)]+1)
flag <- apply(rt, 1, function(x) sum( x == 0) < 250)
rt <- rt[which(flag),]

#=========================================================
# delete 25% genes by row mean

means <- apply(rt,1,mean)
rt_mean <- rt[rev(order(means))[1:10000],]
dim(rt_mean)

#=========================================================
# delete 25% genes by row SD

mads=apply(rt_mean,1,mad)
rt_mads = rt_mean[rev(order(mads))[1:3000],]

#=========================================================
#apply ConsensusCluster analysis

rt_nor=sweep(rt_mads,1, apply(rt_mads,1,median,na.rm=T))
library(ConsensusClusterPlus)
title=tempdir()
title="77"
rt_nor <- as.matrix(rt_nor)
results = ConsensusClusterPlus(rt_nor,maxK=6,reps=50,pItem=0.8,pFeature=1,
                               clusterAlg="hc",
                               distance="pearson",
                               innerLinkage="complete",
```

```
                                seed=1262118388.71279,
                                plot="pdf")

results2 <-results[[2]]["consensusClass"]
results3 <-results[[3]]["consensusClass"]
results4 <-results[[4]]["consensusClass"]
results5 <-results[[5]]["consensusClass"]
results6 <-results[[6]]["consensusClass"]
as.data.frame(results2)
as.data.frame(results3)
as.data.frame(results4)
as.data.frame(results5)
as.data.frame(results6)
new<-data.frame(results2,results3,results4,results5,results6)
write.csv(new,file = "BLCALncRNAcluster.csv")
```

```
#=================================================

Section1  Cluster of Cluster analysis

#=================================================
#load data

setwd("E:\\Rwork")
library(ConsensusClusterPlus)
rt_nor <- read.csv("BLCA.Cluster.csv",header = T,
                   row.names = 1,
                   quote = "",stringsAsFactors = F)

rt_nor <- t(rt_nor)

#=================================================
#apply cluster of cluster analysis

title=tempdir()
title="77"
rt_nor <- as.matrix(rt_nor)
results = ConsensusClusterPlus(rt_nor,maxK=6,reps=50,pItem=0.8,pFeature=1,# notice which maxK you
set
                               clusterAlg="hc", ### "pam","hc","km"
                               distance="pearson",
                               innerLinkage="complete",#implement consensus clustering with
innerLinkage="complete".
                               seed=1262118388.71279,
                               plot="pdf")   # plot="png" will keep figure separately
#save the result
results2 <-results[[2]]["consensusClass"]
results3 <-results[[3]]["consensusClass"]
results4 <-results[[4]]["consensusClass"]
results5 <-results[[5]]["consensusClass"]
results6 <-results[[6]]["consensusClass"]
as.data.frame(results2)
as.data.frame(results3)
as.data.frame(results4)
as.data.frame(results5)
as.data.frame(results6)
new<-data.frame(results2,results3,results4,results5,results6)  ### check
write.csv(new,file = "BLCA.ClusterofCluster.csv")
```




```
#=================================================

#Survival analysis of subtype

#=================================================
#load data

setwd("E:\\RworK")
library(survival)
library(survminer)
data <- read.csv("BLCA_Cluster_survival.csv",header = T)

#=================================================

pdf(file="TCGA_subtype_survival.pdf",height = 4, width = 4)
fit <- survfit(Surv(futime, fustat) ~ subtype,
               data = data)
ggsurv <- ggsurvplot(fit, data = data,
                     pval = T,
                     xlim = c(0,1825),
                     break.time.by = 365,
                     xlab = "Time in days")

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))
ggsurv
dev.off()
```

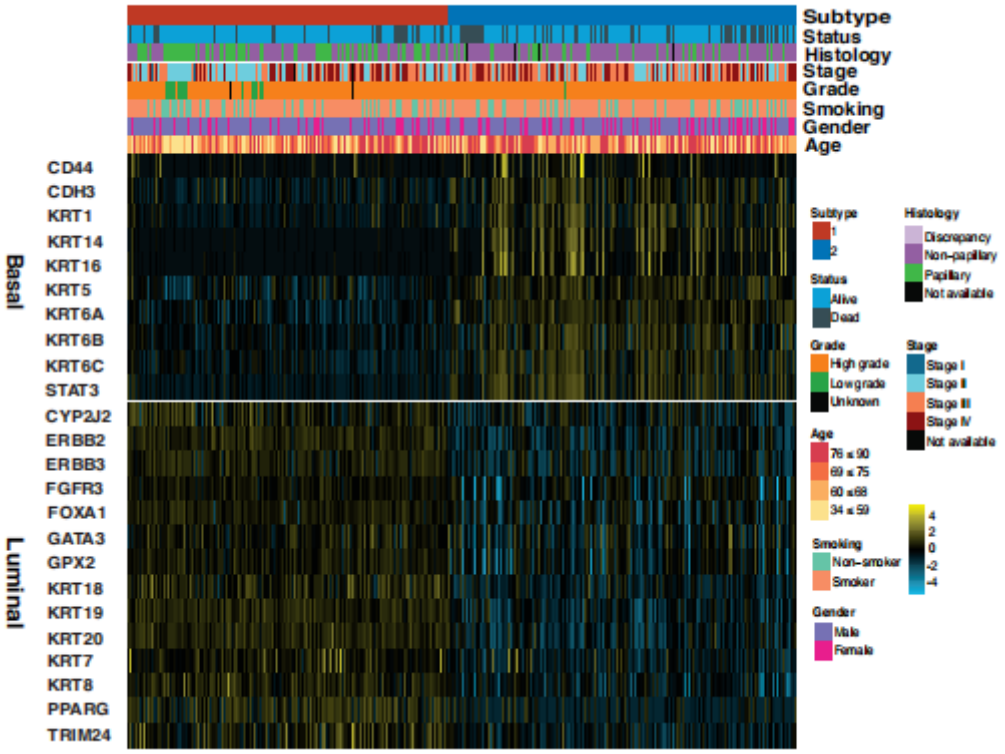

1.00
0.75
0.50
0.25
0.00
Overall Survival
0
1
2
3
4
5
Time in Years
Subtype-1 (n=193)
Subtype-2 (n=210)
p < 0.01

```
setwd("E:\\Rwork")

library(pheatmap)
library(RColorBrewer)


annotation_col <- read.csv("BLCA.anno.heatmap.genename.csv",
                           header = T,row.names = 1)
library(dplyr)
annotation_col <- annotation_col %>%
  dplyr::select(Age,Gender,Smoking,Grade,
                Stage,Hiotology,Status,Subtype,)
annotation_col <- data.frame(annotation_col)

annotation_col$miRNA_cluster <- NULL
annotation_col$mRNA_cluster <- NULL
annotation_col$lncRNA_cluster <- NULL
annotation_col$Subtype <- as.factor(annotation_col$Subtype)
annotation_col$Stage  <- as.factor(annotation_col$Stage)
annotation_col$Hiotology <- as.factor(annotation_col$Hiotology)
annotation_col$Status <- as.factor(annotation_col$Status)
annotation_col$Age  <- as.integer(annotation_col$Age)
annotation_col$Gender <- as.factor(annotation_col$Gender)
annotation_col$Smoking <- as.factor(annotation_col$Smoking)
annotation_col$Grade   <- as.factor(annotation_col$Grade)
annotation_col$Status <- ifelse(annotation_col$Status == "Not Applicable",
                                "Alive","Dead")
annotation_col$Smoking <- ifelse(annotation_col$Smoking == 1,
                                "non-smoker","smoker")
annotation_col$Status <- as.factor(annotation_col$Status)
annotation_col$Smoking <- as.factor(annotation_col$Smoking)
annotation_col$Age  <-  ifelse(annotation_col$Age %in% (76:90), 4,
                        ifelse(annotation_col$Age %in% (69:75),3,
                        ifelse(annotation_col$Age %in% (60:69),2,1)))
annotation_col$Age  <- as.factor(annotation_col$Age)



library(RColorBrewer)
set1 <- c(brewer.pal(12,"Paired"),brewer.pal(8,"Dark2"))
set2 <- c(brewer.pal(9,"Spectral"))
# Specify colors
Subtype = c("#BC3C28", "#0072B5")
names(Subtype) = c("1", "2")
```

```
Stage = c("deepskyblue4","cyan","coral","firebrick4","gray7")
names(Stage) = c("Stage I","Stage II",
                 "Stage III","Stage IV","Not Available")
Hiotology  = c(set1[9],'#925E9F','#42B540',"gray7")
names(Hiotology) = c("Discrepancy","Non-Papillary",
                     "Papillary","Not Available")
Status = c("#00A1D5","#374E55")
names(Status) = c("Alive","Dead")

Smoking = brewer.pal(5,"Set2")[1:2]
names(Smoking) = c("non-smoker","smoker")
Gender = brewer.pal(5,"Dark2")[3:4]
names(Gender) = c("MALE","FEMALE")
Age = set2[1:4]
names(Age) = c("4","3","2","1")
Grade = c("#FF7F0D","#2CA02C","gray7")
names(Grade)  = c("High Grade","Low Grade","Unknown")


ann_colors = list(Subtype = Subtype,
                  Age = Age,
                  Gender = Gender,
                  Smoking= Smoking,
                  Grade = Grade,
                  # miRNA_cluster = miRNA_cluster,
                  # miRNA_cluster=miRNA_cluster,
                  # lncRNA_cluster=lncRNA_cluster,
                  Stage = Stage,
                  Status = Status,
                  Hiotology = Hiotology )




basal_luminal <- read.csv("basal_luminal_marker.csv",
                          header = T,
                          row.names = 1)

basal_luminal <- t(basal_luminal)
basal_luminal <- basal_luminal[-1,]
basal <- basal_luminal[1:10,]
luminal <- basal_luminal[11:24,]
basal =(log2(basal+1))
luminal = (log2(luminal+1))
```

```
# bk = unique(c(seq(-3,5, length=50)))
pdf("basal.pdf",width = 18,height = 18)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 = pheatmap(basal, scale = 'row',
         cluster_cols = FALSE,
         # breaks = bk,
         cluster_row = T,
         border_color  = NA,
         annotation_colors = ann_colors,
         show_colnames     = FALSE,
         show_rownames     = FALSE,
         color = colorRampPalette(c("#20B6E2",
                                    "#020303",
                                    "#F5EB17"))(50),
         legend = FALSE,
         annotation_col = annotation_col,
         annotation_legend = T,treeheight_row=0,
         treeheight_col=0,
         fontsize = 16,
         fontsize_row=6,
         fontsize_col = 6)
dev.off()
# bk = unique(c(seq(0,12, length=50)))
pdf("luminal.pdf",width = 8,height = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
library(pheatmap)
pheatmap(luminal, cluster_cols = FALSE,
         scale = "row",
         cluster_row = T,
         border_color  = NA,
         # breaks = bk,
         show_colnames     = FALSE,
         show_rownames     = FALSE,
         color = colorRampPalette(c("#20B6E2",
                                    "#020303",
                                    "#F5EB17"))(50),
         legend = FALSE,
          treeheight_row=0,
         treeheight_col=0,
         fontsize = 6.5,
         fontsize_row=6,
         fontsize_col = 6)

dev.off()
```

C

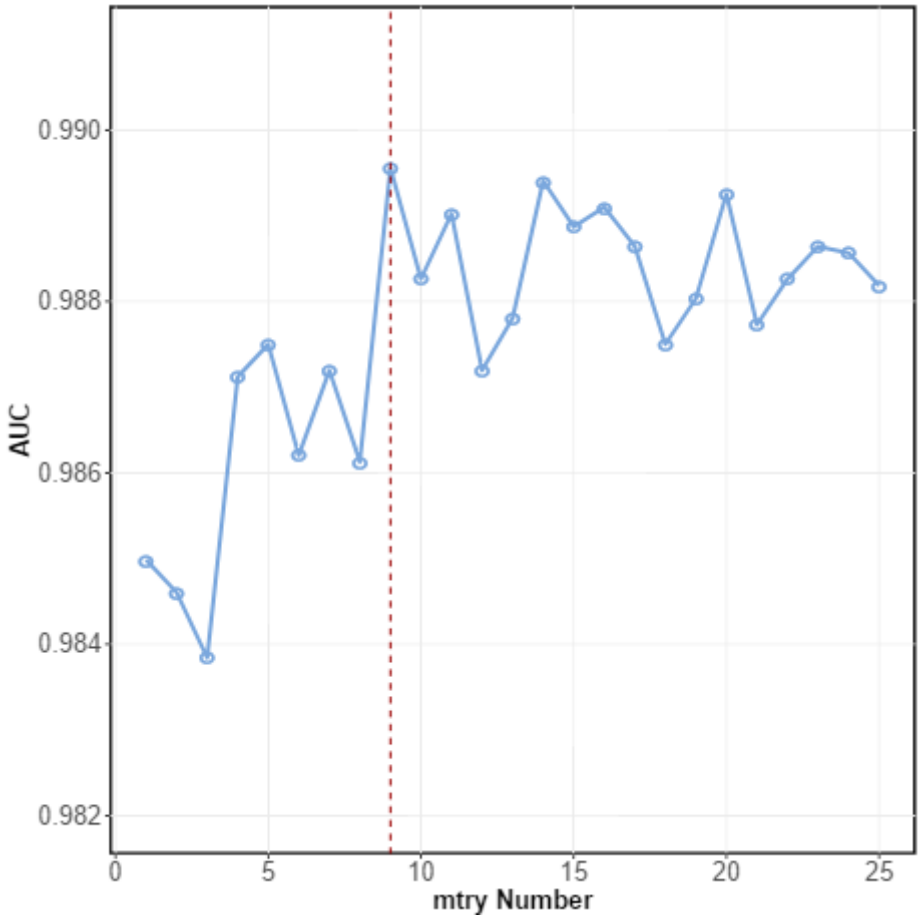

Subtype
Status
Histology
Stage
Grade
Smoking
Gender
Age
CD44
CDH3
KRT1
KRT14
KRT16
KRT5
KRT6A
KRT6B
KRT6C
STAT3
CYP2J2
ERBB2
ERBB3
FGFR3
FOXA1
GATA3
GPX2
KRT18
KRT19
KRT20
KRT7
KRT8
PPARG
TRIM24
Basal
Luminal
Subtype
1
2
Status
Alive
Dead
Grade
High grade
Low grade
Unknown
Age
76 ≤ 90
69 ≤ 75
60 ≤ 68
34 ≤ 59
Smoking
Non-smoker
Smoker
Gender
Male
Female
Histology
Discrepancy
Non-papillary
Papillary
Not available
Stage
Stage I
Stage II
Stage III
Stage IV
Not available
4
2
0
-2
-4

```r
# ============================================
# Ensemble Learning on mRNA expression data
# ============================================


#Section1 Decision Trees Model


# ============================================

setwd("E:\\Rwork")
library(MASS)
library(C50)
DT_data <- read.csv("mRNA_for_DT.csv",header = T)
DT_data <- DT_data[,-1]
DT_data[is.na(DT_data)] <- 0

set.seed(1234)
#60% data for train and 40% data for test
index <- sample(nrow(DT_data),0.6*nrow(DT_data))
DT_train <- DT_data[index,]
DT_test <- DT_data[-index,]
DT_train$subtype <- as.character(DT_train$subtype)
DT_train$subtype <- as.factor(DT_train$subtype)
DT_test$subtype <- as.character(DT_test$subtype)
DT_test$subtype <- as.factor(DT_test$subtype)

set.seed(1234)
tc <- C5.0Control(subset =F,
                  CF=0.25,
                  winnow=F,
                  noGlobalPruning=F,
                  minCases =20)


DT <- C5.0( subtype ~.,
               data    = DT_train,
               rules   = F,
               control = tc)
summary(DT)

# ============================================
# Create the confusion matrix
```

```
pred1 <- predict(DT, DT_test)
Freq1 <- table(pred1,DT_test$subtype)
sum (diag(Freq1)) / sum(Freq1)

# ================================================
#Calculate AUC

library(ROCR)
DT_testp <- predict(DT,DT_test,type='prob')[,2]
DT_pred<-prediction(DT_testp,DT_test$subtype)
DT_perf <- performance(DT_pred,"tpr","fpr")
plot(DT_perf, col='blue',lty=2)
auc <- performance(DT_pred,'auc')
auc = unlist(slot(auc,"y.values"))

plot(DT_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For DT = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)

# ================================================


#Section2 Randome Forest Model


# ================================================

#load neccessary packages
library(ggplot2)
library(RColorBrewer)
library(ggsci)
library(randomForest)
rf_data <- DT_data

# ================================================
#data was divided to train set(60%) and test set (40%)

index <- sample(nrow(rf_data),0.6*nrow(rf_data))
rf_train <- rf_data[index,]
rf_test <- rf_data[-index,]
rf_train$subtype <- as.character(rf_train$subtype)
rf_train$subtype <- as.factor(rf_train$subtype)
```

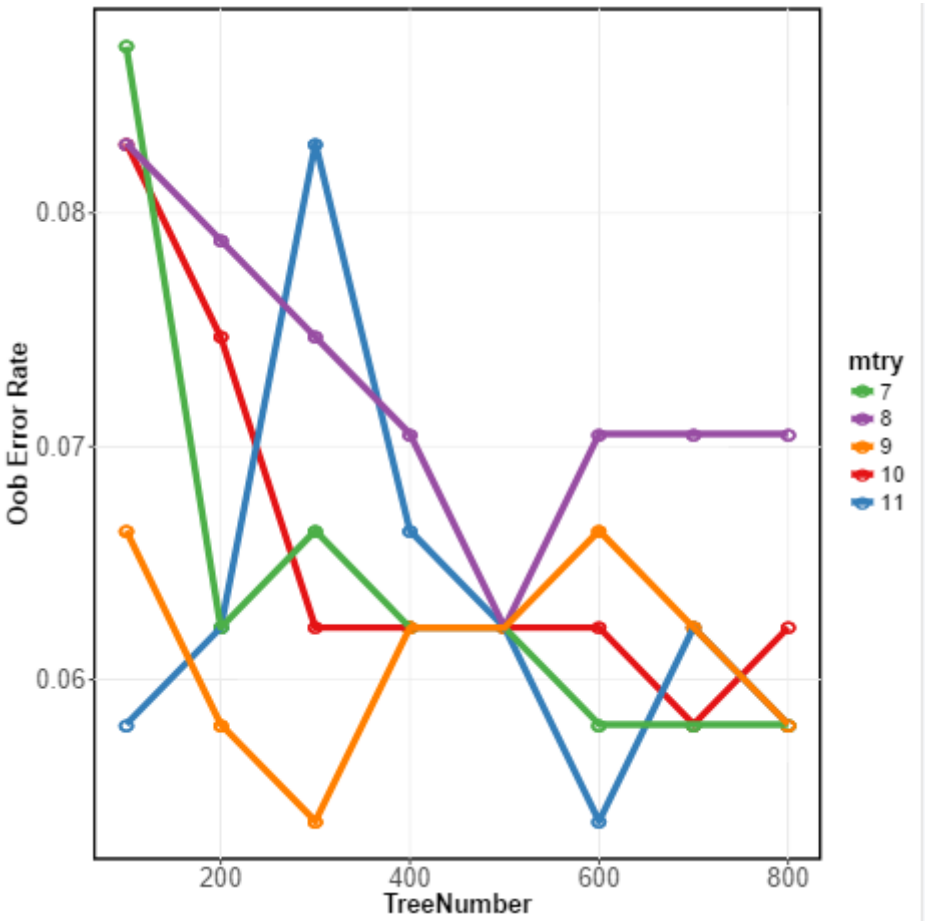
```
rf_test$subtype <- as.character(rf_test$subtype)
rf_test$subtype <- as.factor(rf_test$subtype)


# ============================================
#-------------find the best mtry value---------------------

n <- 25
set.seed(1234)
library(tcltk)
pb<-tkProgressBar("Processing","Completed%",0,2500)
for (i in 1:(n)){
  info<- sprintf("Processing %d%%", round(i*10/length(n)))
  setTkProgressBar(pb, i*100/length(n), sprintf("Completed (%s)", info),info)
  mtry_fit <- randomForest(subtype~.,data = rf_train,mtry = i)
  rf_testp <- predict(mtry_fit,rf_test,type='prob')[,2]
  rf_pred<-prediction(rf_testp,rf_test$subtype)
  rf_perf <- performance(rf_pred,"tpr","fpr")
  auc <- performance(rf_pred,'auc')
  auc = unlist(slot(auc,"y.values"))
  print(auc )
}


RF.mtry <- read.csv("mRNA.RF.mtry.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)

pdf(file="mRNA.RF.mtry.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(RF.mtry , aes(x=mtry, y=AUC))+
  geom_point(size = 3.8,shape=1,color='#7AA6DC',stroke =1.2)

p1 <- p1 +coord_cartesian(ylim=c(0.975, 0.995))
scale_y_continuous(breaks=seq(0.975, 0.995, 0.007))

p1 <- p1+scale_x_continuous(breaks=seq(0, 25, 5))+
  geom_line(color='#7AA6DC', size=1.2)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())+
```

```
  geom_vline(aes(xintercept=7),
             colour="#990000",
             linetype="dashed")
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5 ))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("mtry Number") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("AUC")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))


p1
dev.off()
```




```
# ============================================
#-------------find the best number of ntree---------------------

m =  "best mtry number"
for (i in c(100,200,300,400,500,600,700,800)){
  for (j in c(m-2,m-1, m, m+1,m+2)){
    set.seed(123)
    rf=randomForest(subtype ~., data=rf_train,
                    mtry=j, ntree=i)
    error_rate <- rf$err.rate[i]
    if (exists('oob_err')==FALSE){
      oob_err = c(i,j,error_rate)
    }
    else{
      oob_err = rbind(oob_err, c(i,j,error_rate)) mtry
    }
  }
}
oob_err <- as.data.frame(oob_err)
names(oob_err) <- c('ntree', 'mtry', 'oob_error_rate')
oob_err$mtry <- as.factor(oob_err$mtry)
library (ggplot2)
oob_err <- oob_err[order(oob_err$mtry),]
set1 <- c(brewer.pal(5,"Set1"))


pdf(file="mRNA.RF.ntree.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(oob_err, aes(x=ntree, y=oob_error_rate, color=mtry))+
```

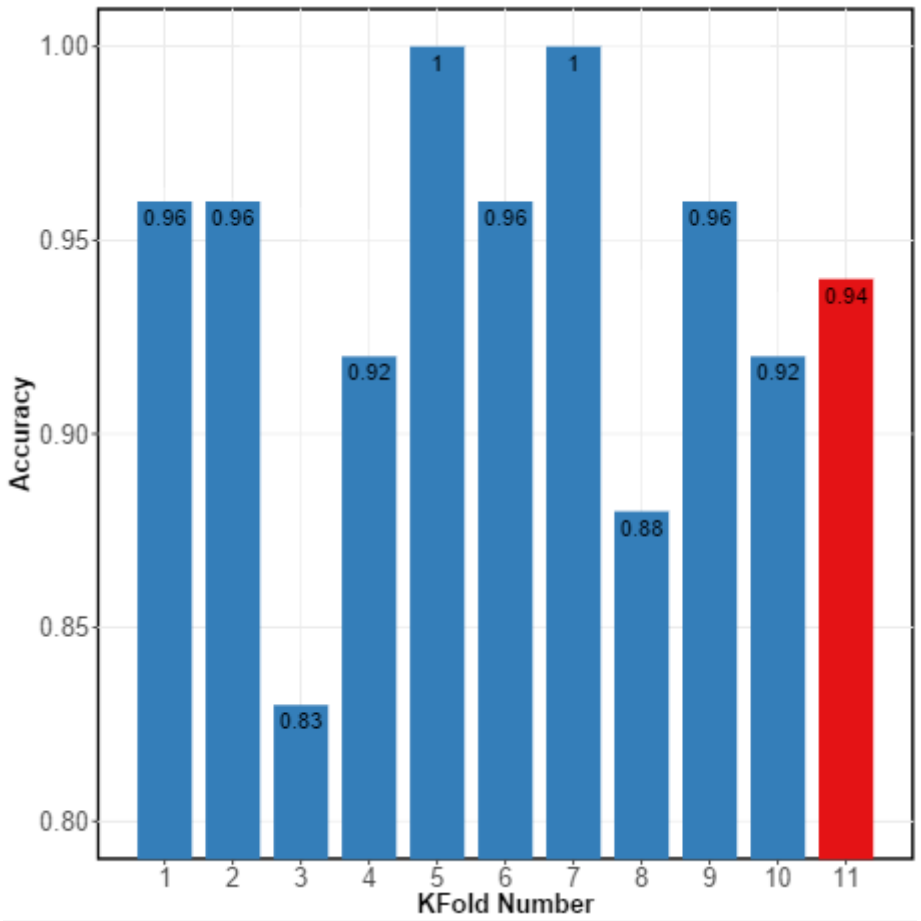
```
  geom_point(size=3.8,shape=1,stroke =1.2)+
  geom_line(size=2)+scale_color_manual(breaks = c("7", "8","9",
                                                  "10", "11"),
                                       values=set1) +
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(legend.title   = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+
  theme(legend.text = element_text(size = 14))+
  theme(legend.position = "right")



p1 <- p1 +coord_cartesian(ylim=c(0.04, 0.1))
scale_y_continuous(breaks=seq(0.04, 0.1, 0.01))

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5
))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("TreeNumber") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Oob Error Rate")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1
dev.off()
```




```
# ============================================
#--------------------establish the model-------------------------------

set.seed(1234)
rf <- randomForest(subtype~.,
                   data = rf_train,
                   mtry = 9,
                   ntree = 300,
                   importance= TRUE,
                   proximity = TRUE)
rf

# ============================================
# Create the confusion matrix


pred1 <- predict(rf,rf_test)
Freq1 <- table(pred1,rf_test$subtype)
sum (diag(table(predict(rf,rf_test),
                rf_test$subtype))) / sum(Freq1)

# ============================================
#-------- calculate AUC -------------------------

library(ROCR)
rf_testp <- predict(rf,rf_test,type='prob')[,2]
rf_pred<-prediction(rf_testp,rf_test$subtype)
rf_perf <- performance(rf_pred, "tpr", "fpr")
plot(rf_perf, col='blue',lty=2)
```

```
auc <- performance(rf_pred,'auc')
auc = unlist(slot(auc,"y.values"))
plot(rf_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For RF = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)

# ============================================
#-------- 10-fold Crossvalidation -------------------------

data <- rf_train
library("caret")
set.seed(1234)
folds<-createFolds(y=data$subtype,k=10)
re <- {}
for(i in 1:10){
  traindata <- data[-folds[[i]],]
  testdata <- data[folds[[i]],]
  rf <- randomForest(subtype ~ ., data=traindata,
                     ntree=200, proximity=TRUE,
                     mtry = 16)

  pred1 <- predict(rf,testdata)
  Freq1 <- table(pred1,testdata$subtype)
  temp<- sum(diag(Freq1))/sum(Freq1)
  re <- c(re,temp)
}
re <- as.data.frame(re)
re$K.fold  <- row.names(re)
names(re)[1] <- c("ACC")
library (ggplot2)
mi_rf_Kfold <- re

set1 <- c(brewer.pal(5,"Set1"))
pdf(file="mRNA.RF.Kfold.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
rf_Kfold$K.fold <- as.numeric(rf_Kfold$K.fold)
rf_Kfold$pos <- rf_Kfold$K.fold == 11
rf_Kfold$pos  <- as.factor(rf_Kfold$pos)
rf_Kfold$ACC   <- as.numeric(rf_Kfold$ACC)
rf_Kfold$ACC <- round(rf_Kfold$ACC,2)



p1 <- ggplot(rf_Kfold, aes(x=K.fold, y=ACC))+
  geom_bar(aes(fill=pos),stat = "identity", width = 0.8)+
  scale_fill_manual(values=set1[2:1])
p1 <- p1 +  coord_cartesian(ylim=c(0.8, 1))+
  scale_y_continuous(breaks=seq(0.8, 1, 0.04)) +
  scale_x_continuous(breaks = seq(1,11,1))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
```

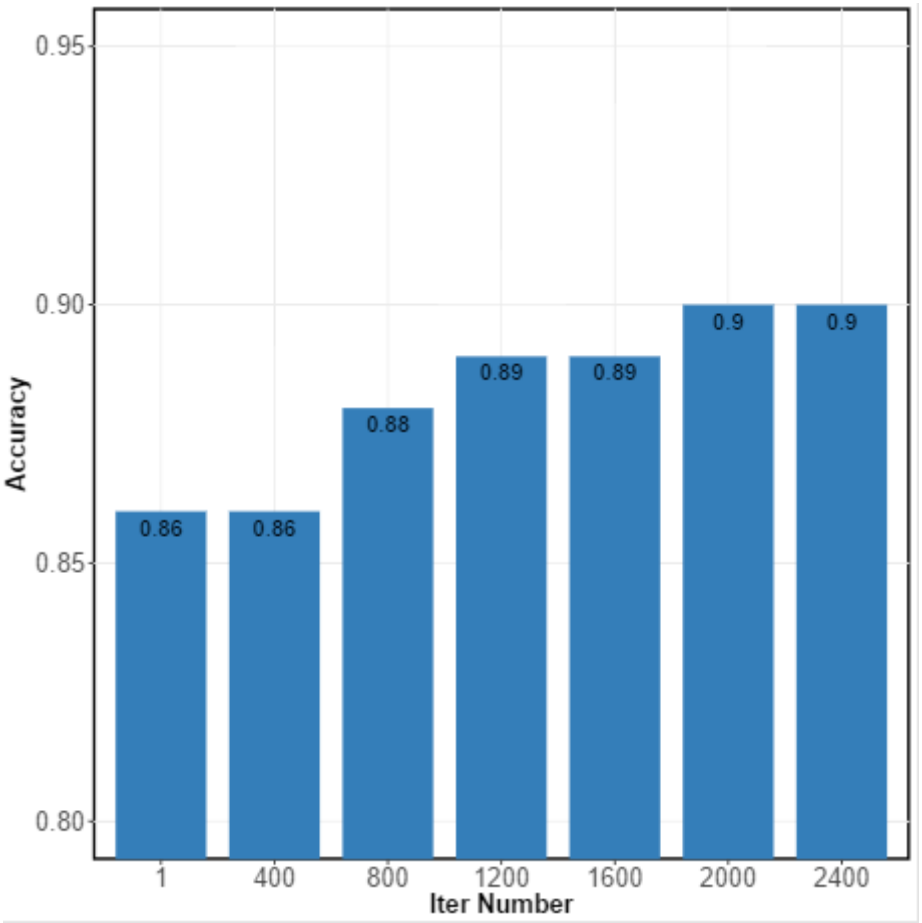
```
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("KFold Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16,  face = "bold"))+
  theme(legend.background    = element_rect(fill = "white"))+

  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
p1
dev.off()
```




```
# ============================================
#select important genes

rf_importance <- as.data.frame(importance(rf , type=1))
rf_importance$symbol <- row.names(rf_importance)

rf_importance <- rf_importance[order(rf_importance$MeanDecreaseAccuracy,
                                     decreasing = T),]
rf_importance <-  rf_importance[which(rf_importance$MeanDecreaseAccuracy > 0),]




# ============================================


#Section3 XGboost Model


# ============================================



mydata <- DT_data
mydata$subtype <- as.numeric(mydata$subtype)
mydata$subtype <- mydata$subtype - 1
set.seed(1234)
```

```
trainIndex   <- createDataPartition(mydata$subtype,
                                    p=0.6,
                                    list=FALSE,
                                    times=1)
train   <- mydata[trainIndex,]
test    <- mydata[-trainIndex,]
train.label  <- train$subtype
test.label   <- test$subtype
library(Matrix)
dtrain  <- sparse.model.matrix(subtype ~ .-1, data=train)
dtest   <- sparse.model.matrix(subtype ~ .-1, data=test)

# ============================================
#xgboost model constructed

library(xgboost)
param <- list(objective   = "binary:logistic",
              eval_metric = "auc",
              max_depth   = 14,
              eta         = 0.001,
              gammma      = 1,
              colsample_bytree = 0.6,
              min_child_weight = 1,
              seed = 1234)

cv.res = xgb.cv(data = dtrain, nfold = 2,
                nrounds = 5000,
                label   = train.label,
                objective = "binary:logistic",
                eval_metric = "auc")


set1 <- c(brewer.pal(5,"Set1"))
xg.cv <- read.csv("mRNA.xg.CV.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)
pdf(file="xg.cv.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
xval = xg.cv$IterNumber/100
xg.cv$ACC  =  round(xg.cv$ACC,2)
p1 <- ggplot(xg.cv, aes(x=factor(IterNumber), y=ACC))+
  geom_bar(fill=set1[2],stat = "identity", width = 0.8)

p1 <- p1 + coord_cartesian(ylim=c(0.80, 095))+
  scale_y_continuous(breaks=seq(0.80, 0.95, 0.04))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())
```

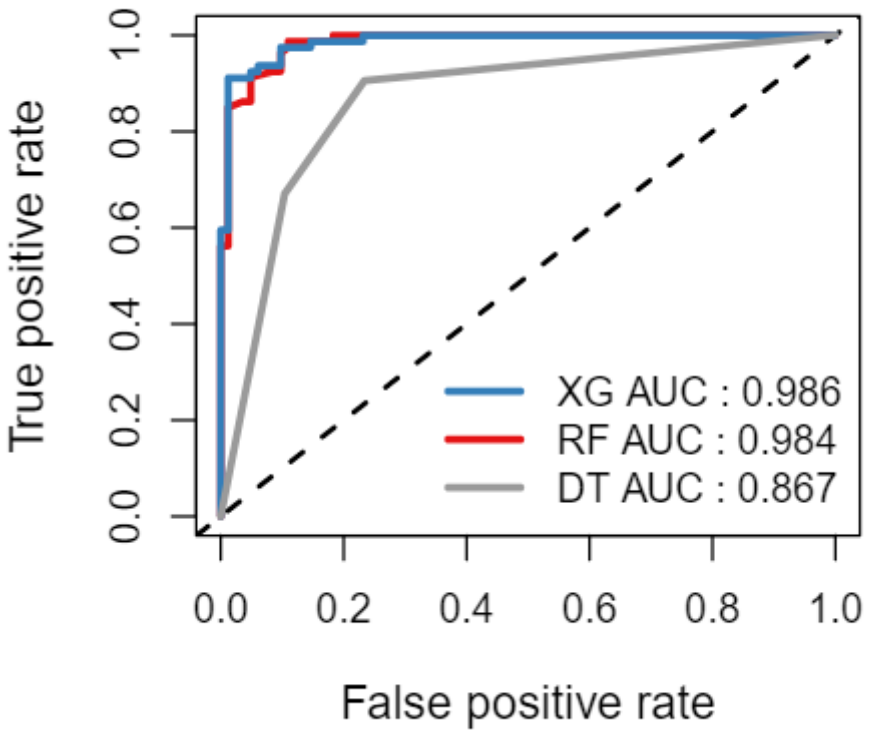
```
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("Iter Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+

  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
p1
dev.off()
```




```
system.time(xgb <- xgboost(params  = param,
                           data    = dtrain,
                           label   = train.label,
                           nrounds = 2000,
                           print_every_n = 10,
                           verbose = 1))

# ================================================
# Create the confusion matrix

pred_xg <- predict(xgb, dtest)
pred.resp <- ifelse(pred_xg >= 0.5, 1, 0)
confusionMatrix(pred.resp, test.label, positive="1")

# ================================================
#xgboost auc

library(ROCR)
# Use ROCR package to plot ROC Curve
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
plot(xgb.perf, col='blue',lty=2)
auc <- performance(xgb.pred,'auc')
auc = unlist(slot(auc,"y.values"))
plot(xgb.perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve for xgboost (", "AUC = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)
```

```
# ================================================
#select importance genes

model <- xgb.dump(xgb, with_stats=TRUE)
names <- dimnames(dtrain)[[2]]
xg_importance_matrix <- xgb.importance(names,
                                       model = xgb)

xg_importance_matrix <- xg_importance_matrix[order(xg_importance_matrix$Gain,
                                                   decreasing = T),]

names(xg_importance_matrix)[1] <- c("symbol")
inter_importance <- merge(rf_importance,
                          xg_importance_matrix,
                          by="symbol")
write.csv(inter_importance,
          file = "blca.mRNA.rf.xg.intersect.csv",
          row.names = F)

# ================================================
#plot auc

library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(8, "Dark2"))

pdf(file="mRNA.xg.rf_auc.pdf")
library(ROCR)
testp_rf <- predict(rf,rf_test,type='prob')[,2]
pred_rf <- prediction(testp_rf,rf_test$subtype)
perf_rf <- performance(pred_rf,"tpr","fpr")
auc_rf <- performance(pred_rf,'auc')
auc_rf = unlist(slot(auc_rf,"y.values"))
plot(perf_rf,
     xlim =c(0,1), ylim=c(0,1),col=set1[1],
     main = "",
     lwd = 2,
     cex.main=1.3,
     cex.lab=1.2,
     cex.axis=1.2,
     font=1.2,
)
graphics::abline(a = 0,  b = 1,lwd = 2,lty=2)
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
auc_xg <- performance(xgb.pred,'auc')
auc_xg = unlist(slot(auc_xg,"y.values"))
plot(xgb.perf,col=set1[2],add = TRUE,lwd = 2)
plot(DT_perf, col=set1[9],add = TRUE,lwd = 2)
legend("bottomright", bty = "n" ,
       col=c(set1[1],set1[2], set1[9]),
       lty=1, c("XG Auc : 0.986",
```

```
                "RF Auc : 0.984",
                "DT Auc : 0.867"),lwd=2)

dev.off()
```

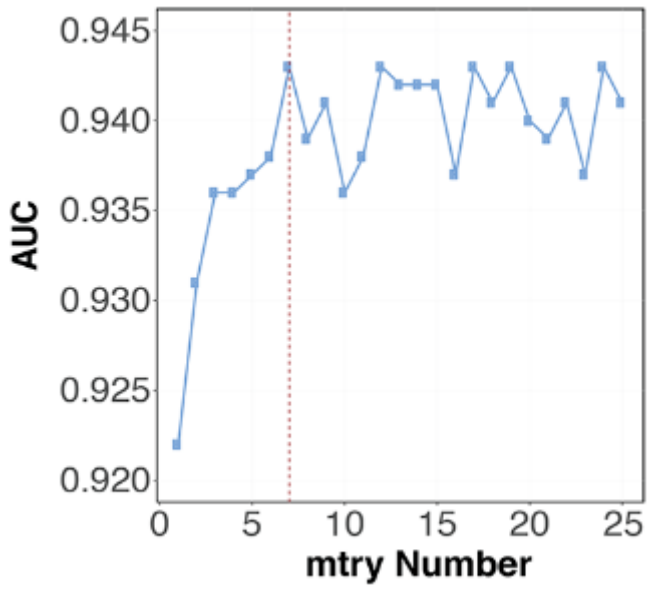


```r
# ============================================
# Ensemble Learning on miRNA expression data
# ============================================


#Section1 Decision Trees Model


# ============================================

setwd("E:\\Rwork")
library(MASS)
library(C50)
DT_data <- read.csv("miRNA_for_DT.csv",header = T)
DT_data <- DT_data[,-1]
DT_data[is.na(DT_data)] <- 0

set.seed(1234)
#60% data for train and 40% data for test
index <- sample(nrow(DT_data),0.6*nrow(DT_data))
DT_train <- DT_data[index,]
DT_test <- DT_data[-index,]
DT_train$subtype <- as.character(DT_train$subtype)
DT_train$subtype <- as.factor(DT_train$subtype)
DT_test$subtype <- as.character(DT_test$subtype)
DT_test$subtype <- as.factor(DT_test$subtype)

set.seed(1234)
tc <- C5.0Control(subset =F,
                  CF=0.25,
                  winnow=F,
                  noGlobalPruning=F,
                  minCases =20)

DT <- C5.0( subtype ~.,
               data    = DT_train,
               rules   = F,
               control = tc)
summary(lnc_DT)

# ============================================
# Create the confusion matrix

pred1 <- predict(DT, DT_test)
Freq1 <- table(pred1,DT_test$subtype)
sum (diag(Freq1)) / sum(Freq1)

# ============================================
#Calculate AUC

library(ROCR)
```

```
DT_testp <- predict(DT,DT_test,type='prob')[,2]
DT_pred<-prediction(DT_testp,DT_test$subtype)
DT_perf <- performance(DT_pred,"tpr","fpr")
plot(DT_perf, col='blue',lty=2)
auc <- performance(DT_pred,'auc')
auc = unlist(slot(auc,"y.values"))

plot(DT_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For DT = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)

# ============================================


#Section2 Randome Forest Model


# ============================================

#load neccessary packages
library(ggplot2)
library(RColorBrewer)
library(ggsci)
library(randomForest)
rf_data <- DT_data

# ============================================
#data was divided to train set(60%) and test set (40%)

index <- sample(nrow(rf_data),0.6*nrow(rf_data))
rf_train <- rf_data[index,]
rf_test <- rf_data[-index,]
rf_train$subtype <- as.character(rf_train$subtype)
rf_train$subtype <- as.factor(rf_train$subtype)
rf_test$subtype <- as.character(rf_test$subtype)
rf_test$subtype <- as.factor(rf_test$subtype)


# ============================================
#------------find the best mtry value--------------------

n <- 25
set.seed(1234)
library(tcltk)
pb<-tkProgressBar("Processing","Completed%",0,2500)
for (i in 1:(n)){
  info<- sprintf("Processing %d%%", round(i*10/length(n)))
  setTkProgressBar(pb, i*100/length(n), sprintf("Completed (%s)", info),info)
  mtry_fit <- randomForest(subtype~.,data = rf_train,mtry = i)
  rf_testp <- predict(mtry_fit,rf_test,type='prob')[,2]
  rf_pred<-prediction(rf_testp,rf_test$subtype)
  rf_perf <- performance(rf_pred,"tpr","fpr")
```

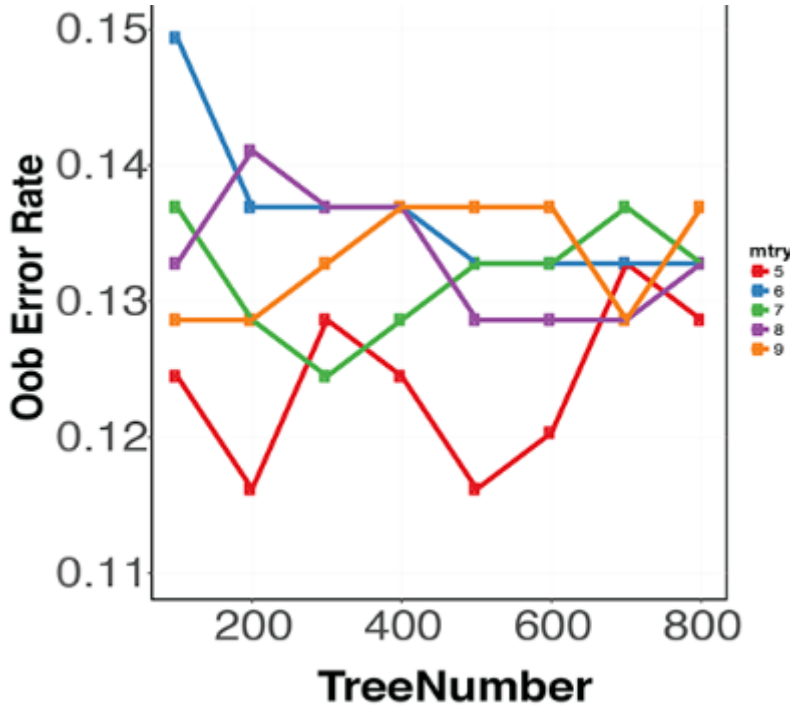
```
  auc <- performance(rf_pred,'auc')
  auc = unlist(slot(auc,"y.values"))
  print(auc )
}


RF.mtry <- read.csv("mRNA.RF.mtry.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)

pdf(file="miRNA.RF.mtry.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(RF.mtry , aes(x=mtry, y=AUC))+
  geom_point(size = 3.8,shape=1,color='#7AA6DC',stroke =1.2)

p1 <- p1 +coord_cartesian(ylim=c(0.975, 0.995))
scale_y_continuous(breaks=seq(0.975, 0.995, 0.007))

p1 <- p1+scale_x_continuous(breaks=seq(0, 25, 5))+
  geom_line(color='#7AA6DC', size=1.2)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())+
  geom_vline(aes(xintercept=7),
             colour="#990000",
             linetype="dashed")
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5 ))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("mtry Number") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("AUC")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))
```

```
p1
dev.off()
```




```
# ==========================================
#-------------find the best number of ntree--------------------

m =  "best mtry number"
for (i in c(100,200,300,400,500,600,700,800)){
  for (j in c(m-2,m-1, m, m+1,m+2)){
    set.seed(123)
    rf=randomForest(subtype ~., data=rf_train,
                    mtry=j, ntree=i)
    error_rate <- rf$err.rate[i]
    if (exists('oob_err')==FALSE){
      oob_err = c(i,j,error_rate)
    }
    else{
      oob_err = rbind(oob_err, c(i,j,error_rate)) mtry
    }
  }
}
oob_err <- as.data.frame(oob_err)
names(oob_err) <- c('ntree', 'mtry', 'oob_error_rate')
oob_err$mtry <- as.factor(oob_err$mtry)
library (ggplot2)
oob_err <- oob_err[order(oob_err$mtry),]
set1 <- c(brewer.pal(5,"Set1"))


pdf(file="miRNA.RF.ntree.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(oob_err, aes(x=ntree, y=oob_error_rate, color=mtry))+
  geom_point(size=3.8,shape=1,stroke =1.2)+
  geom_line(size=2)+scale_color_manual(breaks = c("5", "6","7",
                                                  "8", "9"),
                                       values=set1) +
```

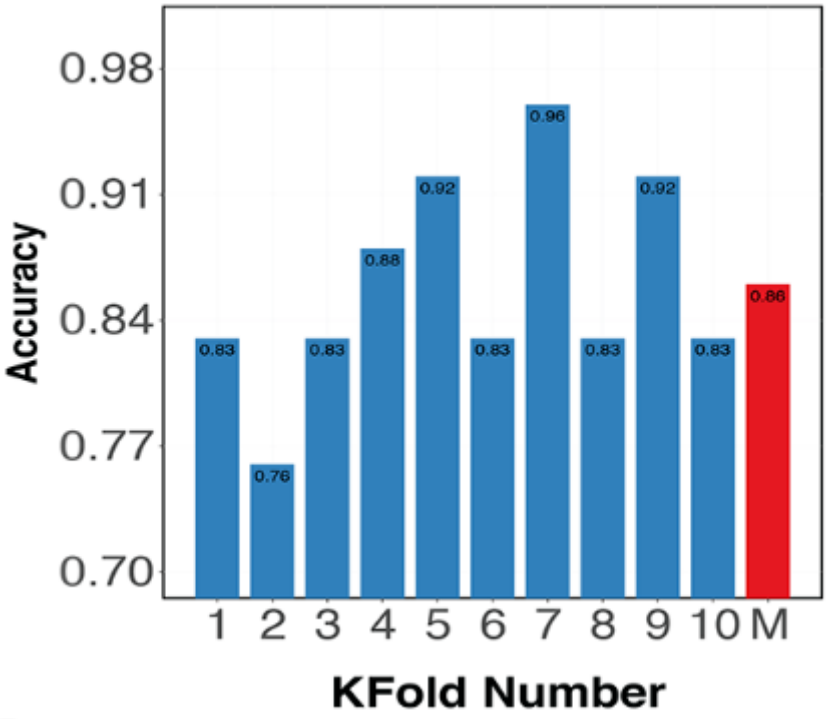
```
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(legend.title   = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+
  theme(legend.text = element_text(size = 14))+
  theme(legend.position = "right")



p1 <- p1 +coord_cartesian(ylim=c(0.11, 0.15))
scale_y_continuous(breaks=seq(0.11, 0.15, 0.01))

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5
))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("TreeNumber") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Oob Error Rate")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1
dev.off()
```




```
# ==============================================
#--------------------establish the model-------------------------------

set.seed(1234)
rf <- randomForest(subtype~.,
                   data = rf_train,
                   mtry = 7,
                   ntree = 300,
                   importance= TRUE,
                   proximity = TRUE)
rf

# ==============================================
# Create the confusion matrix


pred1 <- predict(rf,rf_test)
Freq1 <- table(pred1,rf_test$subtype)
sum (diag(table(predict(rf,rf_test),
                rf_test$subtype))) / sum(Freq1)

# ==============================================
#-------- calculate AUC -------------------------

library(ROCR)
rf_testp <- predict(rf,rf_test,type='prob')[,2]
rf_pred<-prediction(rf_testp,rf_test$subtype)
rf_perf <- performance(rf_pred, "tpr", "fpr")
plot(rf_perf, col='blue',lty=2)
auc <- performance(rf_pred,'auc')
auc = unlist(slot(auc, "y.values"))
plot(rf_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For RF = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
```

```
abline(0,1)

# ============================================
#-------- 10-fold Crossvalidation ------------------------

data <- rf_train
library("caret")
set.seed(1234)
folds<-createFolds(y=data$subtype,k=10)
re <- {}
for(i in 1:10){
  traindata <- data[-folds[[i]],]
  testdata <- data[folds[[i]],]
  rf <- randomForest(subtype ~ ., data=traindata,
                     ntree=200, proximity=TRUE,
                     mtry = 16)

  pred1 <- predict(rf,testdata)
  Freq1 <- table(pred1,testdata$subtype)
  temp<- sum(diag(Freq1))/sum(Freq1)
  re <- c(re, temp)
}
re <- as.data.frame(re)
re$K.fold  <- row.names(re)
names(re)[1] <- c("ACC")
library (ggplot2)
mi_rf_Kfold <- re

set1 <- c(brewer.pal(5, "Set1"))
pdf(file="miRNA.RF.Kfold.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
rf_Kfold$K.fold <- as.numeric(rf_Kfold$K.fold)
rf_Kfold$pos <- rf_Kfold$K.fold == 11
rf_Kfold$pos  <- as.factor(rf_Kfold$pos)
rf_Kfold$ACC   <- as.numeric(rf_Kfold$ACC)
rf_Kfold$ACC <- round(rf_Kfold$ACC,2)



p1 <- ggplot(rf_Kfold, aes(x=K.fold, y=ACC))+
  geom_bar(aes(fill=pos),stat = "identity", width = 0.8)+
  scale_fill_manual(values=set1[2:1])
p1 <- p1 +  coord_cartesian(ylim=c(0.8, 1))+
  scale_y_continuous(breaks=seq(0.8, 1, 0.04)) +
  scale_x_continuous(breaks = seq(1,11,1))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())
```

```
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("KFold Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16,  face = "bold"))+
  theme(legend.background    = element_rect(fill = "white"))+

  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
p1
dev.off()
```

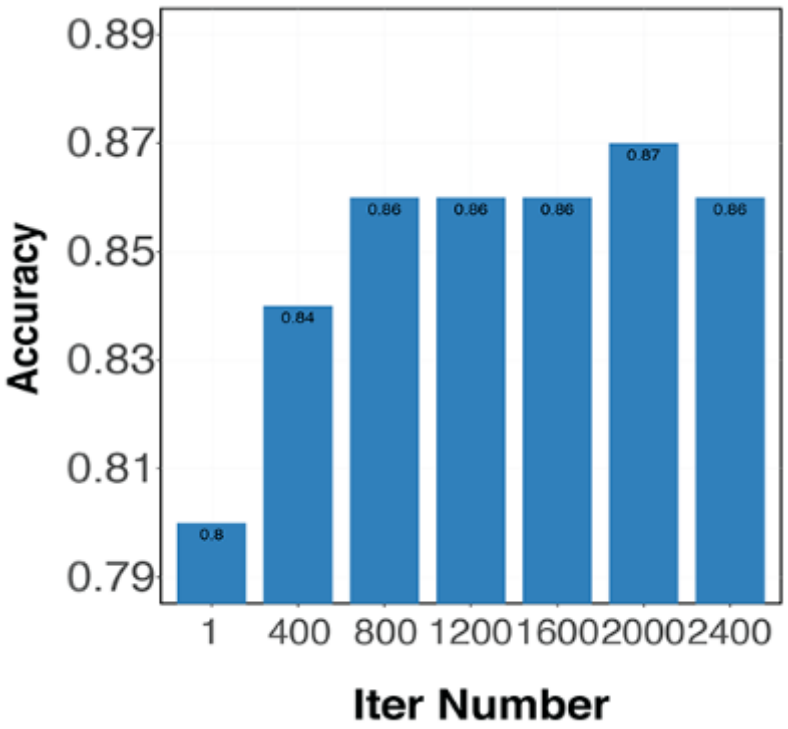



```
# ==========================================
#select important genes
```

```
rf_importance <- as.data.frame(importance(rf , type=1))
rf_importance$symbol <- row.names(rf_importance)

rf_importance <- rf_importance[order(rf_importance$MeanDecreaseAccuracy,
                                     decreasing = T),]
rf_importance <-  rf_importance[which(rf_importance$MeanDecreaseAccuracy > 0),]




# ==============================================


#Section3 XGboost Model


# ==============================================



mydata <- DT_data
mydata$subtype <- as.numeric(mydata$subtype)
mydata$subtype <- mydata$subtype - 1
set.seed(1234)
trainIndex   <- createDataPartition(mydata$subtype,
                                    p=0.6,
                                    list=FALSE,
                                    times=1)
train   <- mydata[trainIndex,]
test    <- mydata[-trainIndex,]
train.label  <- train$subtype
test.label   <- test$subtype
library(Matrix)
dtrain  <- sparse.model.matrix(subtype ~ .-1, data=train)
dtest   <- sparse.model.matrix(subtype ~ .-1, data=test)

# ==============================================
#xgboost model constructed

library(xgboost)
param <- list(objective   = "binary:logistic",
              eval_metric = "auc",
              max_depth   = 14,
              eta         = 0.001,
              gammma      = 1,
              colsample_bytree = 0.6,
              min_child_weight = 1,
              seed = 1234)

cv.res = xgb.cv(data = dtrain, nfold = 2,
                nrounds = 5000,
                label   = train.label,
```

```
                objective = "binary:logistic",
                eval_metric = "auc")


set1 <- c(brewer.pal(5,"Set1"))
xg.cv <- read.csv("miRNA.xg.CV.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)
pdf(file="mi.xg.cv.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
xval = miRNA.xg.cv$IterNumber/100
xg.cv$ACC  =  round(xg.cv$ACC,2)
p1 <- ggplot(xg.cv, aes(x=factor(IterNumber), y=ACC))+
  geom_bar(fill=set1[2],stat = "identity", width = 0.8)

p1 <- p1 + coord_cartesian(ylim=c(0.79, 0.89))+
  scale_y_continuous(breaks=seq(0.79, 0.89, 0.02))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("Iter Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+
```

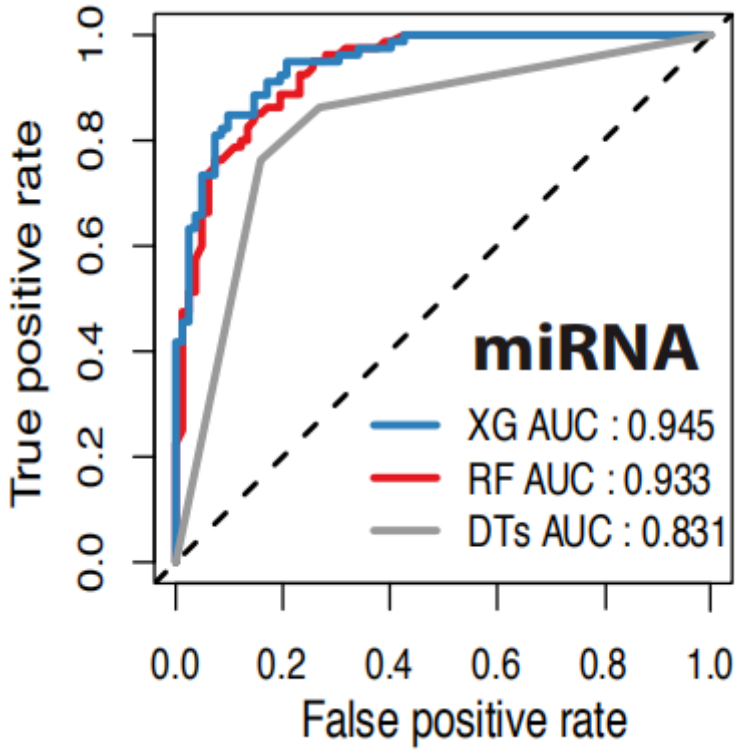
```
  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
pl
dev.off()
```




```
system.time(xgb <- xgboost(params  = param,
                           data    = dtrain,
                           label   = train.label,
                           nrounds = 2000,
                           print_every_n = 10,
                           verbose = 1))

# ==============================================
# Create the confusion matrix

pred_xg <- predict(xgb, dtest)
pred.resp <- ifelse(pred_xg >= 0.5, 1, 0)
confusionMatrix(pred.resp, test.label, positive="1")

# ==============================================
#xgboost auc

library(ROCR)
# Use ROCR package to plot ROC Curve
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
plot(xgb.perf, col='blue',lty=2)
auc <- performance(xgb.pred,'auc')
auc = unlist(slot(auc,"y.values"))
plot(xgb.perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve for xgboost (", "AUC = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
```

```
abline(0,1)


# ==============================================
#select importance genes

model <- xgb.dump(xgb, with_stats=TRUE)
names <- dimnames(dtrain)[[2]]
xg_importance_matrix <- xgb.importance(names,
                                       model = xgb)

xg_importance_matrix <- xg_importance_matrix[order(xg_importance_matrix$Gain,
                                                   decreasing = T),]

names(xg_importance_matrix)[1] <- c("symbol")
inter_importance <- merge(rf_importance,
                          xg_importance_matrix,
                          by="symbol")
write.csv(inter_importance,
          file = "blca.miRNA.rf.xg.intersect.csv",
          row.names = F)

# ==============================================
#plot auc

library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(8, "Dark2"))

pdf(file="miRNA.xg.rf_auc.pdf")
library(ROCR)
testp_rf <- predict(rf,rf_test,type='prob')[,2]
pred_rf <- prediction(testp_rf,rf_test$subtype)
perf_rf <- performance(pred_rf,"tpr","fpr")
auc_rf <- performance(pred_rf,'auc')
auc_rf = unlist(slot(auc_rf,"y.values"))
plot(perf_rf,
     xlim =c(0,1), ylim=c(0,1),col=set1[1],
     main = "",
     lwd = 2,
     cex.main=1.3,
     cex.lab=1.2,
     cex.axis=1.2,
     font=1.2,
)
graphics::abline(a = 0,  b = 1,lwd = 2,lty=2)
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
auc_xg <- performance(xgb.pred,'auc')
auc_xg = unlist(slot(auc_xg,"y.values"))
plot(xgb.perf,col=set1[2],add = TRUE,lwd = 2)
plot(DT_perf, col=set1[9],add = TRUE,lwd = 2)
legend("bottomright", bty = "n" ,
       col=c(set1[1],set1[2], set1[9]),
```

```
       lty=1, c("XG Auc : 0.986",
                "RF Auc : 0.984",
                "DT Auc : 0.867"),lwd=2)

dev.off()
```

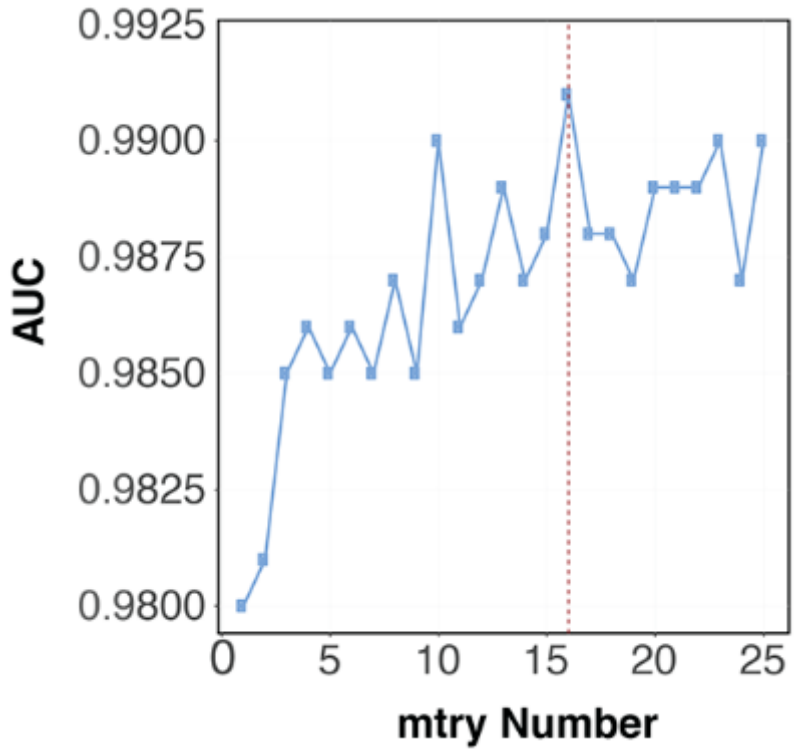


```
# ============================================
# Ensemble Learning on lncRNA expression data
# ============================================


#Section1 Decision Trees Model


# ============================================

setwd("E:\\Rwork")
library(MASS)
library(C50)
DT_data <- read.csv("lncRNA_for_DT.csv",header = T)
DT_data <- DT_data[,-1]
DT_data[is.na(DT_data)] <- 0

set.seed(1234)
#60% data for train and 40% data for test
index <- sample(nrow(DT_data),0.6*nrow(DT_data))
DT_train <- DT_data[index,]
DT_test <- DT_data[-index,]
DT_train$subtype <- as.character(DT_train$subtype)
DT_train$subtype <- as.factor(DT_train$subtype)
DT_test$subtype <- as.character(DT_test$subtype)
DT_test$subtype <- as.factor(DT_test$subtype)

set.seed(1234)
tc <- C5.0Control(subset =F,
                  CF=0.25,
                  winnow=F,
                  noGlobalPruning=F,
                  minCases =20)


lnc_DT <- C5.0( subtype ~.,
                data    = DT_train,
                rules   = F,
                control = tc)
summary(lnc_DT)

# ============================================
# Create the confusion matrix
```

```
pred1 <- predict(lnc_DT, DT_test)
Freq1 <- table(pred1,DT_test$subtype)
sum (diag(Freq1)) / sum(Freq1)

# ================================================
#Calculate AUC

library(ROCR)
DT_testp <- predict(lnc_DT,DT_test,type='prob')[,2]
DT_pred<-prediction(DT_testp,DT_test$subtype)
DT_perf <- performance(DT_pred,"tpr","fpr")
plot(DT_perf, col='blue',lty=2)
auc <- performance(DT_pred,'auc')
auc = unlist(slot(auc,"y.values"))

plot(DT_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For DT = ",auc,")"),
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)

# ================================================


#Section2 Randome Forest Model


# ================================================

#load neccessary packages
library(ggplot2)
library(RColorBrewer)
library(ggsci)
library(randomForest)
rf_data <- DT_data

# ================================================
#data was divided to train set(60%) and test set (40%)

index <- sample(nrow(rf_data),0.6*nrow(rf_data))
rf_train <- rf_data[index,]
rf_test <- rf_data[-index,]
rf_train$subtype <- as.character(rf_train$subtype)
rf_train$subtype <- as.factor(rf_train$subtype)
```

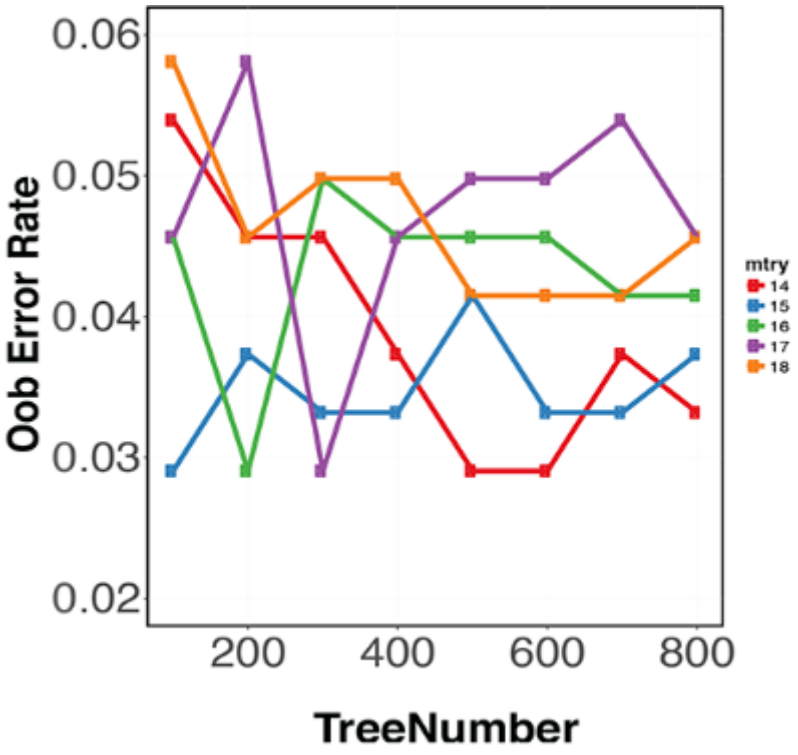
```
rf_test$subtype <- as.character(rf_test$subtype)
rf_test$subtype <- as.factor(rf_test$subtype)


# ============================================
#-------------find the best mtry value---------------------

n <- 25
set.seed(1234)
library(tcltk)
pb<-tkProgressBar("Processing","Completed %",0,2500)
for (i in 1:(n)){
  info<- sprintf("Processing %d%%", round(i*10/length(n)))
  setTkProgressBar(pb, i*100/length(n), sprintf("Completed (%s)", info),info)
  mtry_fit <- randomForest(subtype~.,data = rf_train,mtry = i)
  rf_testp <- predict(mtry_fit,rf_test,type='prob')[,2]
  rf_pred<-prediction(rf_testp,rf_test$subtype)
  rf_perf <- performance(rf_pred,"tpr","fpr")
  auc <- performance(rf_pred,'auc')
  auc = unlist(slot(auc,"y.values"))
  print(auc )
}


lnc.RF.mtry <- read.csv("lnc.RF.mtry.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)

pdf(file="lnc.RF.mtry.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(lnc.RF.mtry , aes(x=mtry, y=AUC))+
  geom_point(size = 3.8,shape=1,color='#7AA6DC',stroke =1.2)

p1 <- p1 +coord_cartesian(ylim=c(0.980, 0.9925))
scale_y_continuous(breaks=seq(0.980, 0.9925, 0.006))

p1 <- p1+scale_x_continuous(breaks=seq(0, 25, 5))+
  geom_line(color='#7AA6DC', size=1.2)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())+
```

```
  geom_vline(aes(xintercept=16),
             colour="#990000",
             linetype="dashed")
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5 ))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("mtry Number") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("AUC")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))


p1
dev.off()
```




```
# ============================================
#-------------find the best number of ntree--------------------

m =  "best mtry number"
for (i in c(100,200,300,400,500,600,700,800)){
  for (j in c(m-2,m-1, m, m+1,m+2)){
    set.seed(123)
    rf=randomForest(subtype ~., data=rf_train,
                    mtry=j, ntree=i)
    error_rate <- rf$err.rate[i]
    if (exists('oob_err')==FALSE){
      oob_err = c(i,j,error_rate)
    }
    else{
      oob_err = rbind(oob_err, c(i,j,error_rate))
    }
  }
}
oob_err <- as.data.frame(oob_err)
names(oob_err) <- c('ntree', 'mtry', 'oob_error_rate')
oob_err$mtry <- as.factor(oob_err$mtry)
library (ggplot2)
oob_err <- oob_err[order(oob_err$mtry),]
set1 <- c(brewer.pal(5,"Set1"))


pdf(file="lnc.RF.ntree.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
p1 <- ggplot(oob_err, aes(x=ntree, y=oob_error_rate, color=mtry))+
  geom_point(size=3.8,shape=1,stroke =1.2)+
  geom_line(size=2)+scale_color_manual(breaks = c(14", "15","16",
                                                  "17", "18"),
```

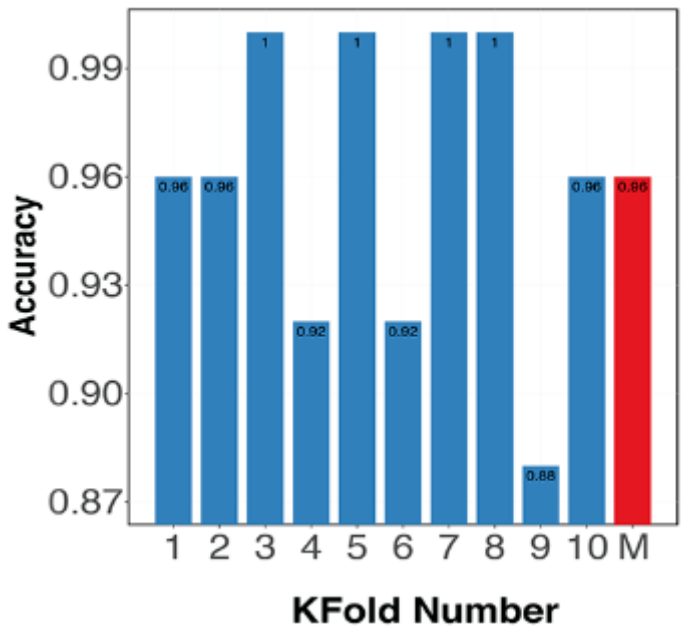
```
                                    values=set1) +
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(legend.title   = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+
  theme(legend.text = element_text(size = 14))+
  theme(legend.position = "right")



p1 <- p1 +coord_cartesian(ylim=c(0.02, 0.06))
scale_y_continuous(breaks=seq(0.02, 0.06, 0.01))

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5
))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("TreeNumber") +
  theme(axis.title.x = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Oob Error Rate")+
  theme(axis.title.y = element_text(size = 16,

                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1
dev.off()
```




```
# ================================================
#--------------------establish the model-------------------------------

set.seed(1234)
rf <- randomForest(subtype~.,
                   data = rf_train,
                   mtry = 16,
                   ntree = 200,
                   importance= TRUE,
                   proximity = TRUE)
rf

# ================================================
# Create the confusion matrix


pred1 <- predict(rf,rf_test)
Freq1 <- table(pred1,rf_test$subtype)
sum (diag(table(predict(rf,rf_test),
                rf_test$subtype))) / sum(Freq1)

# ================================================
#-------- calculate AUC -------------------------

library(ROCR)
rf_testp <- predict(rf,rf_test,type='prob')[,2]
rf_pred<-prediction(rf_testp,rf_test$subtype)
rf_perf <- performance(rf_pred, "tpr", "fpr")
plot(rf_perf, col='blue',lty=2)
auc <- performance(rf_pred,'auc')
auc = unlist(slot(auc, "y.values"))
plot(rf_perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve (", "AUC For RF = ",auc,")"),
```

```
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)

# ============================================
#-------- 10-fold Crossvalidation -------------------------

data <- rf_train
library("caret")
set.seed(1234)
folds<-createFolds(y=data$subtype,k=10)
re <- {}
for(i in 1:10){
  traindata <- data[-folds[[i]],]
  testdata <- data[folds[[i]],]
  rf <- randomForest(subtype ~ ., data=traindata,
                     ntree=200, proximity=TRUE,
                     mtry = 16)

  pred1 <- predict(rf,testdata)
  Freq1 <- table(pred1,testdata$subtype)
  temp<- sum(diag(Freq1))/sum(Freq1)
  re <- c(re,temp)
}
re <- as.data.frame(re)
re$K.fold  <- row.names(re)
names(re)[1] <- c("ACC")
library (ggplot2)
mi_rf_Kfold <- re

set1 <- c(brewer.pal(5,"Set1"))
pdf(file="lnc.RF.Kfold.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
rf_Kfold$K.fold <- as.numeric(rf_Kfold$K.fold)
rf_Kfold$pos <- rf_Kfold$K.fold == 11
rf_Kfold$pos  <- as.factor(rf_Kfold$pos)
rf_Kfold$ACC   <- as.numeric(rf_Kfold$ACC)
rf_Kfold$ACC <- round(rf_Kfold$ACC,2)



p1 <- ggplot(rf_Kfold, aes(x=K.fold, y=ACC))+
  geom_bar(aes(fill=pos),stat = "identity", width = 0.8)+
  scale_fill_manual(values=set1[2:1])
p1 <- p1 +  coord_cartesian(ylim=c(0.87, 1))+
  scale_y_continuous(breaks=seq(0.87, 1, 0.03)) +
  scale_x_continuous(breaks = seq(1,11,1))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())
```

```
p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("KFold Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16, face = "bold"))+
  theme(legend.background    = element_rect(fill = "white"))+

  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
p1
dev.off()
```




```
# ============================================
#select important genes

rf_importance <- as.data.frame(importance(rf , type=1))
rf_importance$symbol <- row.names(rf_importance)
```

```
rf_importance <- rf_importance[order(rf_importance$MeanDecreaseAccuracy,
                                     decreasing = T),]
rf_importance <-  rf_importance[which(rf_importance$MeanDecreaseAccuracy > 0),]
```

```
# ============================================
```

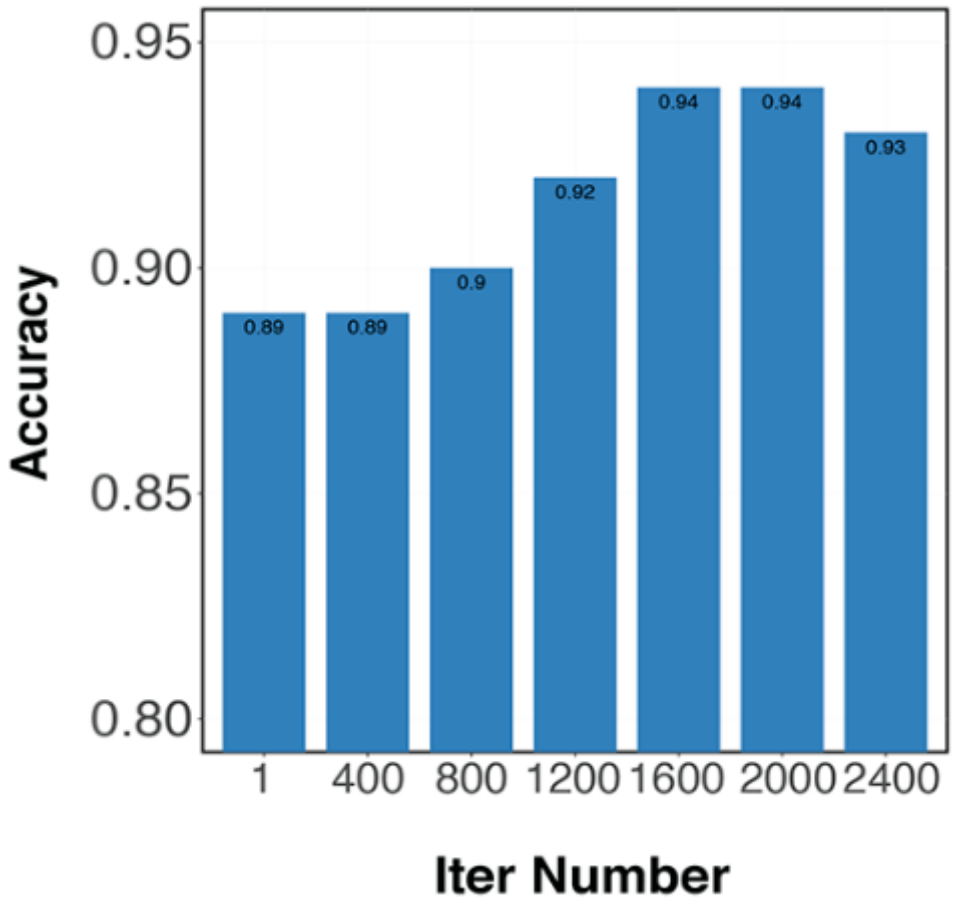
```
#Section3 XGboost Model
```

```
# ============================================
```

```
mydata <- DT_data
mydata$subtype <- as.numeric(mydata$subtype)
mydata$subtype <- mydata$subtype - 1
set.seed(1234)
trainIndex   <- createDataPartition(mydata$subtype,
                                    p=0.6,
                                    list=FALSE,
                                    times=1)
train   <- mydata[trainIndex,]
test    <- mydata[-trainIndex,]
train.label  <- train$subtype
test.label   <- test$subtype
library(Matrix)
dtrain  <- sparse.model.matrix(subtype ~ .-1, data=train)
dtest   <- sparse.model.matrix(subtype ~ .-1, data=test)

# ============================================
#xgboost model constructed

library(xgboost)
param <- list(objective   = "binary:logistic",
              eval_metric = "auc",
              max_depth   = 14,
              eta         = 0.001,
              gammma      = 1,
              colsample_bytree = 0.6,
              min_child_weight = 1,
              seed = 1234)

cv.res = xgb.cv(data = dtrain, nfold = 2,
                nrounds = 5000,
                label   = train.label,
                objective = "binary:logistic",
                eval_metric = "auc")
```

```
set1 <- c(brewer.pal(5,"Set1"))
xg.cv <- read.csv("lncRNA.xg.CV.csv")
library(ggplot2)
library(RColorBrewer)
library(ggsci)
pdf(file="lnc.xg.cv.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))
xval = xg.cv$IterNumber/100
xg.cv$ACC  =  round(xg.cv$ACC,2)
p1 <- ggplot(xg.cv, aes(x=factor(IterNumber), y=ACC))+
  geom_bar(fill=set1[2],stat = "identity", width = 0.8)

p1 <- p1 + coord_cartesian(ylim=c(0.80, 095))+
  scale_y_continuous(breaks=seq(0.80, 0.95, 0.05))+
  geom_text(aes(label = ACC), vjust = 1.5,
            colour = "black", position = position_dodge(.9),
            size = 5)+
  theme_bw() +
  theme(panel.background = element_rect(colour = "black",
                                        size = 1.5))+
  theme(panel.grid.major  = element_line( size=0.1)) +
  theme(panel.grid.minor = element_blank())

p1 <- p1 + theme(axis.text.x = element_text(size = 16,
                                            vjust = 0.5,
                                            hjust = 0.5))+
  theme(axis.text.y = element_text(size = 16,
                                   vjust = 0.5,
                                   hjust = 0.5))+
  theme(axis.text.y.right = element_text(size = 16,
                                         vjust = 0.5,
                                         hjust = 0.5))



p1 <- p1 + xlab("Iter Number") +
  theme(axis.title.x = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))+

  ylab("Accuracy")+
  theme(axis.title.y = element_text(size = 16,
                                    face = "bold",
                                    vjust = 0.5,
                                    hjust = 0.5))

p1 <- p1 + theme(legend.title = element_text(size = 16, face = "bold"))+
  theme(legend.background   = element_rect(fill = "white"))+

  theme(legend.text = element_text(size = 14))+
  guides(fill=FALSE)
p1
```

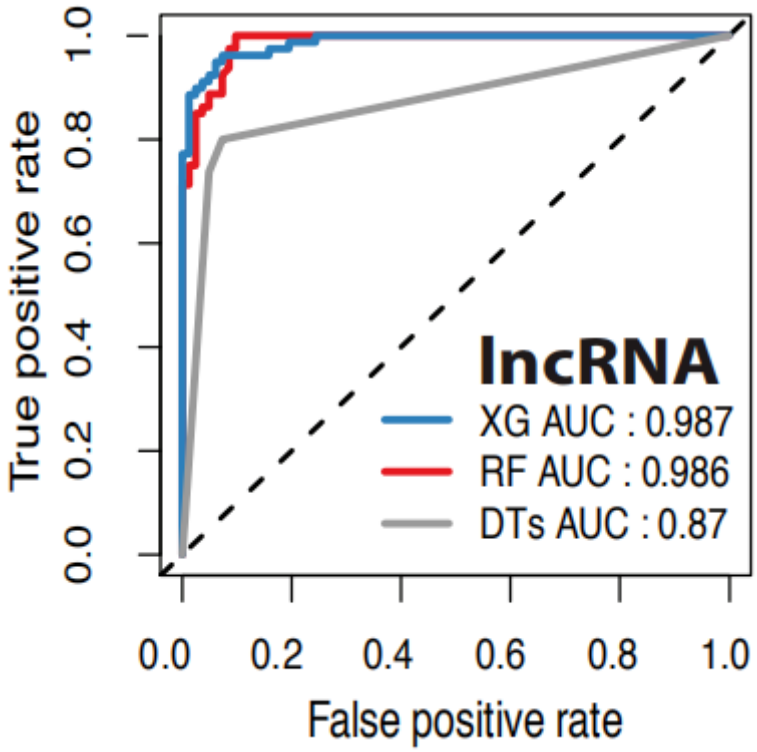
```
dev.off()
```




```
system.time(xgb <- xgboost(params  = param,
                           data    = dtrain,
                           label   = train.label,
                           nrounds = 1600,
                           print_every_n = 10,
                           verbose = 1))

# ==============================================
# Create the confusion matrix

pred_xg <- predict(xgb, dtest)
pred.resp <- ifelse(pred_xg >= 0.5, 1, 0)
confusionMatrix(pred.resp, test.label, positive="1")

# ==============================================
#xgboost auc

library(ROCR)
# Use ROCR package to plot ROC Curve
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
plot(xgb.perf, col='blue',lty=2)
auc <- performance(xgb.pred,'auc')
auc = unlist(slot(auc,"y.values"))
plot(xgb.perf,
     xlim=c(0,1), ylim=c(0,1),col='red',
     main=paste("ROC curve for xgboost (", "AUC = ",auc,")"),
```

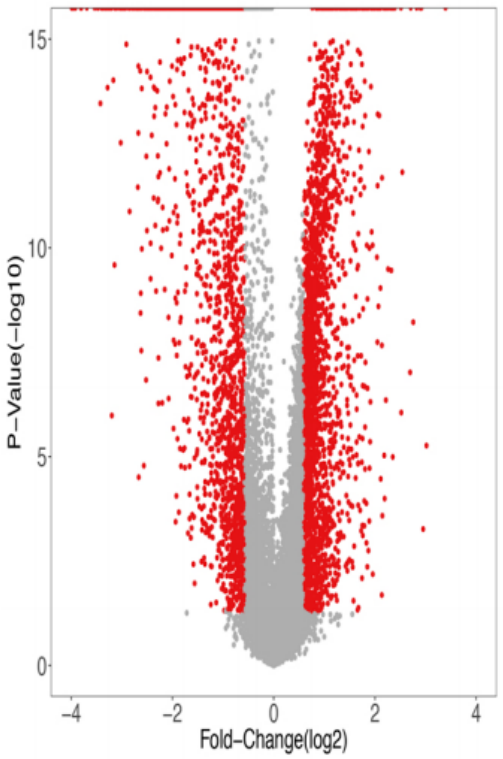
```
     lwd = 2, cex.main=1.3, cex.lab=1.2, cex.axis=1.2, font=1.2)
abline(0,1)


# ============================================
#select importance genes

model <- xgb.dump(xgb, with_stats=TRUE)
names <- dimnames(dtrain)[[2]]
xg_importance_matrix <- xgb.importance(names,
                                       model = xgb)

xg_importance_matrix <- xg_importance_matrix[order(xg_importance_matrix$Gain,
                                                   decreasing = T),]

names(xg_importance_matrix)[1] <- c("symbol")
inter_importance <- merge(rf_importance,
                          xg_importance_matrix,
                          by="symbol")
write.csv(inter_importance,
          file = "blca.lncRNA.rf.xg.intersect.csv",
          row.names = F)

# ============================================
#plot auc

library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(8, "Dark2"))

pdf(file="lncRNA.xg.rf_auc.pdf")
library(ROCR)
testp_rf <- predict(rf,rf_test,type='prob')[,2]
pred_rf <- prediction(testp_rf,rf_test$subtype)
perf_rf <- performance(pred_rf,"tpr","fpr")
auc_rf <- performance(pred_rf,'auc')
auc_rf = unlist(slot(auc_rf,"y.values"))
plot(perf_rf,
     xlim =c(0,1), ylim=c(0,1),col=set1[1],
     main = "",
     lwd = 2,
     cex.main=1.3,
     cex.lab=1.2,
     cex.axis=1.2,
     font=1.2,
)
graphics::abline(a = 0,  b = 1,lwd = 2,lty=2)
xgb.pred <- prediction(pred_xg, test.label)
xgb.perf <- performance(xgb.pred, "tpr", "fpr")
auc_xg <- performance(xgb.pred,'auc')
auc_xg = unlist(slot(auc_xg,"y.values"))
plot(xgb.perf,col=set1[2],add = TRUE,lwd = 2)
plot(DT_perf, col=set1[9],add = TRUE,lwd = 2)
legend("bottomright", bty = "n" ,
```

```
        col=c(set1[1],set1[2], set1[9]),
        lty=1, c("XG Auc : 0.986",
                 "RF Auc : 0.984",
                 "DT Auc : 0.867"),lwd=2)

dev.off()
```



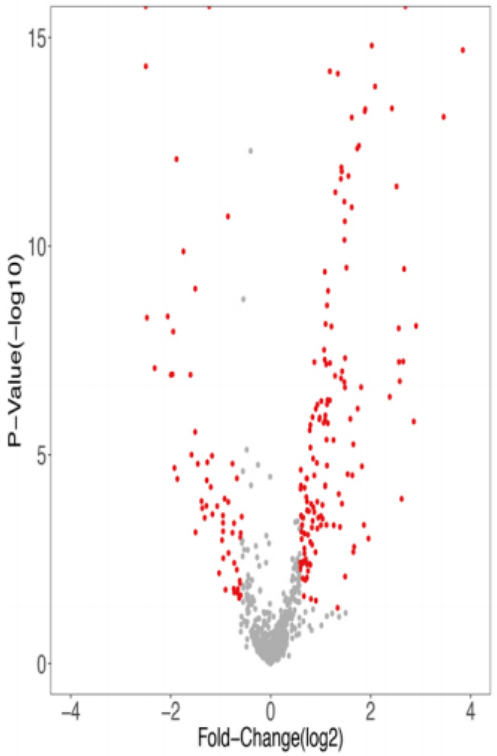
```
# ============================================================


Section1   mRNA DEG


# ============================================================

setwd("E:\\Rwork")
library(tidyr)
library('ballgown')
load("mRNA_exprSet.Rda")
index <- duplicated(mRNA_exprSet$gene_name)
mRNA.data <- mRNA_exprSet[!index,]

# ============================================================

BLCA_fpkm_data = mRNA.data
rownames(BLCA_fpkm_data) = BLCA_fpkm_data[,1]
BLCA_fpkm_data  =  BLCA_fpkm_data[c(-1)]
load("mRNA_exprSet.Rda")
metadata <- data.frame(names(mRNA_exprSet)[-1])
for (i in 1:length(metadata[,1])) {
  num <- as.numeric(substring(metadata[i,1],14,15))
  if (num %in% seq(1,9)) {metadata[i,2] <- "T"}
  if (num %in% seq(10,29)) {metadata[i,2] <- "N"}
}
names(metadata) <- c("TCGA_id","group")
metadata$group <- as.factor(metadata$group)

# ============================================================

result_diff = stattest(gowntable = BLCA_fpkm_data ,
                       pData = metadata ,
                       covariate = "group" ,
                       getFC = TRUE ,
                       log =TRUE,
                       meas='FPKM',
                       feature="gene")
diffSig=result_diff[which(result_diff$pval<0.05  & (result_diff$fc>1.5 | result_diff$fc<0.67)),]
inter_importance <- read.csv("blca.mRNA.rf.xg.intersect.genename.csv",
                             header = T)
merge_gene <- merge(inter_importance, diffSig, by= "id")
write.csv(merge_gene, file = "merge_mRNA.csv")

# ============================================================

library(ggplot2)
library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(12, "Set3"))
pdf(file = "mRNA_volcano.pdf",width=8,height=8)
theme_set(theme_bw())
threshold<-as.factor((result_diff$fc>1.5|result_diff$fc<0.67) & result_diff$pval<0.05)
p <- ggplot(result_diff,aes(x = log2(result_diff$fc),
                            y= -1*log10(result_diff$pval),
                            colour = threshold))+xlab("Fold-Change(log2)")+
                             ylab("P-Value(-log10)")+geom_point()
p <- p + scale_color_manual(breaks = c("FALSE", "TRUE"),
```

```
                        values=c("gray68", set1[1]))
p <- p +theme(panel.grid =element_blank())+
                  theme(axis.line = element_line(size=0))+
                 xlim(-4,4)+
                  ylim(0,15)
p <- p  +guides(colour = FALSE)
p <- p +theme(axis.text=element_text(size=20),
               axis.title=element_text(size=20))
p
dev.off()
```




```
# ======================================================


Section2   miRNA DEG


# ======================================================

setwd("E:\\Rwork")
library(tidyr)
library(ballgown)

miRNA<- read.table("BLCA.miRseq_mature_RPM.txt",header = T,
                   quote = "")
miRNA[is.na(miRNA)] <- 0
index <- duplicated(miRNA$Gene)
miRNA.data <- miRNA [!index,]
names(miRNA.data)[1] <- 'genename'
BLCA_fpkm_data <- miRNA.data

# ======================================================

metadata <- data.frame(names(miRNA)[-1])
for (i in 1:length(metadata[,1])) {
  num <- as.numeric(substring(metadata[i,1],14,15))
  if (num %in% seq(1,9)) {metadata[i,2] <- "T"}
  if (num %in% seq(10,29)) {metadata[i,2] <- "N"}
}
names(metadata) <- c("TCGA_id","group")
metadata$group <- as.factor(metadata$group)
result_diff = stattest(gowntable = BLCA_fpkm_data ,
                       pData = metadata ,
```

```
                        covariate = "group" ,
                        getFC = TRUE ,
                        log =TRUE,
                        feature="gene")
write.csv(result_diff,'TCGA_BLCA_mi_fpkm_result_diff.csv')
diffSig=result_diff[which(result_diff$pval<0.05  & (result_diff$fc>1.5 | result_diff$fc<0.67)),]
write.csv(diffSig, file="diffSig_miRNA_BLCA.csv")

#=============================================================================================================================


inter_importance <- read.csv("blca.mi.rf.xg.intersect.genename.csv",
                             header = T)
stringr::str_sub(diffSig$gene_id, -12)
inter_importance$id <- stringr::str_sub(inter_importance$symbol, -12)
diffSig$id  <- stringr::str_sub(diffSig$id,-12)
merge_gene <- merge(inter_importance, diffSig, by= "id")
write.csv(merge_gene, file = "merge_mi.csv")

#============================================================

library(ggplot2)
library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(12, "Set3"))
pdf(file = "mi_volcano.pdf",width=8,height=8)
theme_set(theme_bw())
threshold<-as.factor((result_diff$fc>1.5|result_diff$fc<0.67) & result_diff$pval<0.05)

p <- ggplot(result_diff,aes(x = log2(result_diff$fc),
                            y= -1*log10(result_diff$pval),
                            colour = threshold))+xlab("Fold-Change(log2)")+
  ylab("P-Value(-log10)")+geom_point()


p <- p + scale_color_manual(breaks = c("FALSE", "TRUE"),
                            values=c("gray68", set1[1]))

p <- p +theme(panel.grid =element_blank())+
  theme(axis.line = element_line(size=0))+
  xlim(-4,4)+
  ylim(0,15)

p <- p  +guides(colour = FALSE)
p <- p +theme(axis.text=element_text(size=20),
              axis.title=element_text(size=20))
p
dev.off()
```

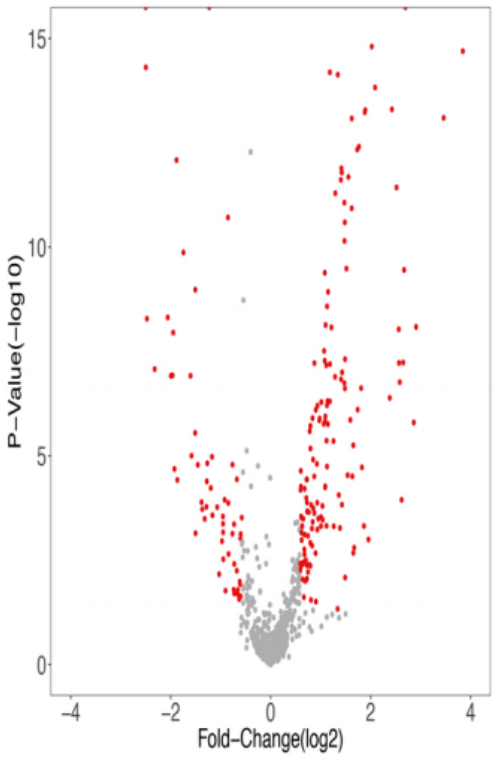



```
# ======================================================
```

Section3 lncRNA DEG

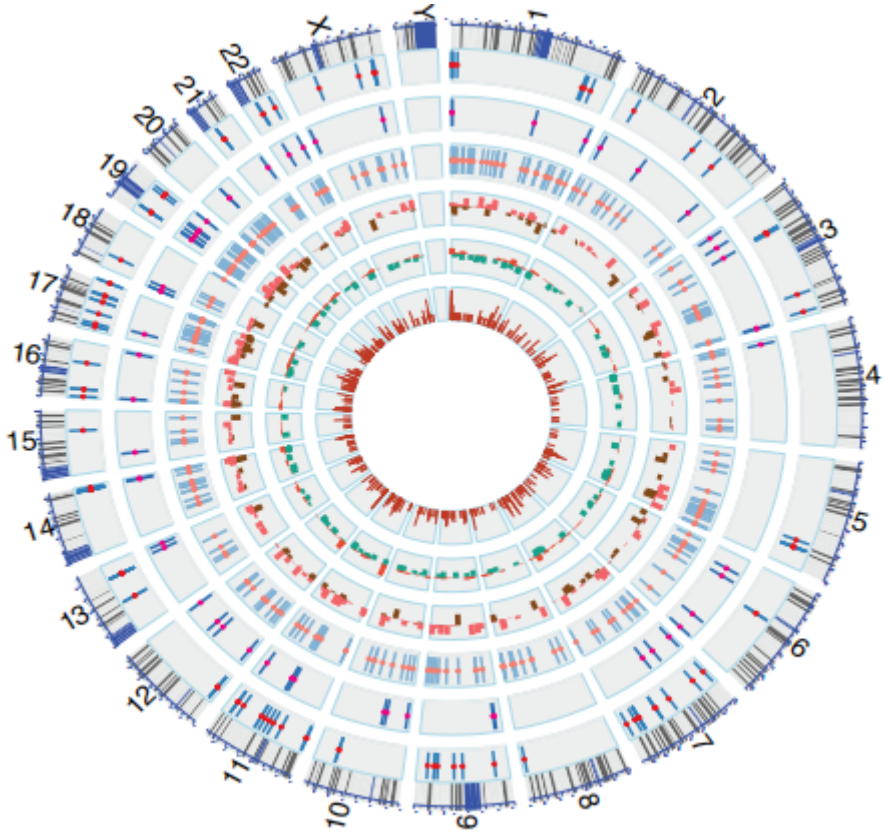
```
# ======================================================

setwd("E:\\Rwork")
library(tidyr)
library(ballgown)
load("LncRNA_exprSet.Rda")
index <- duplicated(LncRNA_exprSet$gene_name)
LncRNA_exprSet1 <- LncRNA_exprSet[!index,]
BLCA_fpkm_data = LncRNA_exprSet1
rownames(BLCA_fpkm_data) = BLCA_fpkm_data[,1]
BLCA_fpkm_data  =  BLCA_fpkm_data[c(-1)]

metadata <- data.frame(names(LncRNA_exprSet)[-1])
for (i in 1:length(metadata[,1])) {
  num <- as.numeric(substring(metadata[i,1],14,15))
  if (num %in% seq(1,9)) {metadata[i,2] <- "T"}
  if (num %in% seq(10,29)) {metadata[i,2] <- "N"}
}
names(metadata) <- c("TCGA_id","group")
metadata$group <- as.factor(metadata$group)

metadata <- metadata[order(metadata$group),]
result_diff = stattest(gowntable = BLCA_fpkm_data ,
                       pData = metadata ,
                       covariate = "group" ,
                       getFC = TRUE ,
                       log =TRUE,
                       meas='FPKM',
                       feature="gene")



write.csv (result_diff,
           "TCGA_BLCA_lnc_fpkm_result_diff.csv",
           row.names = F)

diffSig=result_diff[which(result_diff$pval<0.05  & (result_diff$fc>1.5 | result_diff$fc<0.67)),]
```

```
inter_importance <- read.csv("blca.lnc.rf.xg.intersect.genename.csv",
                             header = T)
merge_gene <- merge(inter_importance, diffSig, by= "id")


library(ggplot2)
library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"),
         brewer.pal(12, "Set3"))
pdf(file = "lnc_volcano.pdf",width=8,height=8)
theme_set(theme_bw())
threshold<-as.factor((result_diff$fc>1.5|result_diff$fc<0.67) & result_diff$pval<0.05)

p <- ggplot(result_diff,aes(x = log2(result_diff$fc),
                            y= -1*log10(result_diff$pval),
                            colour = threshold))+xlab("Fold-Change(log2)")+
  ylab("P-Value(-log10)")+geom_point()


p <- p + scale_color_manual(breaks = c("FALSE", "TRUE"),
                            values=c("gray68", set1[1]))

p <- p +theme(panel.grid =element_blank())+
  theme(axis.line = element_line(size=0))+
  xlim(-4,4)+
  ylim(0,15)

p <- p  +guides(colour = FALSE)
p <- p +theme(axis.text=element_text(size=20),
              axis.title=element_text(size=20))
p
dev.off()
```



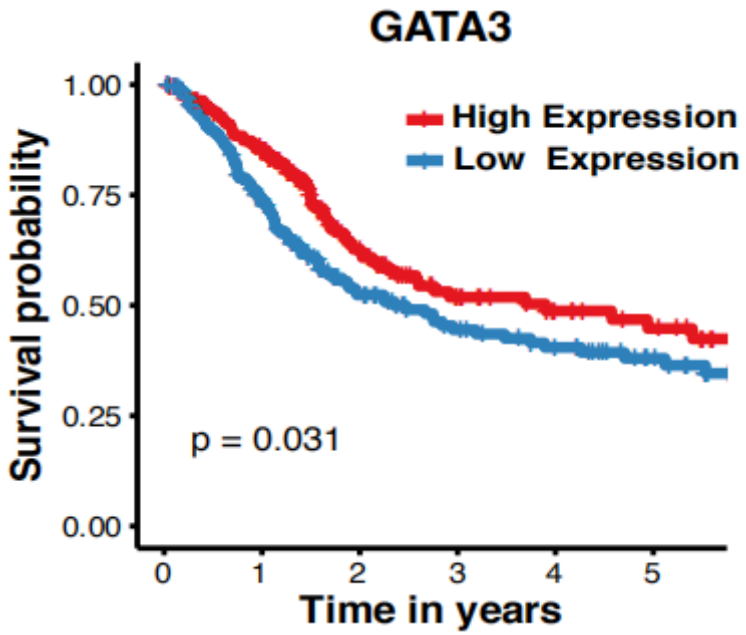
```
#================================================
setwd("E:\\Rwork")
options(stringsAsFactors = FALSE)
set.seed(999)
library("OmicCircos")
circle <- read.csv("new_circle.csv",header = T)
dim(circle)
sur <- read.csv("univariateCox.csv",header = T)
names(sur)[1] <- 'id'
circle <- merge(circle,sur, by = 'id')
load("importance.Rda")

circle <- merge(importance, circle , by = 'id')
circle <- circle %>%
  dplyr::select(Chromosome,chromStart,chromEnd,id)
diff_fc <- circle %>%
  dplyr::select(c(Chromosome,chromStart,fc))
sur_P <-  circle %>%
  dplyr::select(c(Chromosome,chromStart,pvalue))
gene_im <- circle %>%
  dplyr::select(c(Chromosome,chromStart,importance))



circle_new <- read.csv("new_circle.csv",header = T,row.names = 1)
mRNA_name <- row.names(circle_new)[1:278]
mi_name <- row.names(circle_new)[279:335]
lnc_name <- row.names(circle_new)[336:455]
mRNA_loc <- circle1[which(circle1$id %in% mRNA_name),]
lnc_loc <- circle1[which(circle1$id %in% lnc_name),]
mi_loc <- circle1[which(circle1$id %in% mi_name),]



library(RColorBrewer)
set1 = c(brewer.pal(9,"Set1"))
set2 = c(brewer.pal(8,'Accent'))
set3 = c(brewer.pal(8,'Set3'))

type = "chr"
par(mar=c(1,1,1,1))
plot(c(1,800),c(1,800),type="n",axes=FALSE,xlab="",ylab="")
#chromsom
circos(R=400,  cir="hg19",type="chr",W=10,scale=TRUE,print.chr.lab = TRUE)
```

```
#gene location
circos(R=350, cir="hg19",type="b3",W=40,mapping=lnc_loc,B=TRUE,
       col.v=2,col=set1[2])
circos(R=350, cir="hg19",type="s2",W=40,mapping=lnc_loc,B=FALSE,
       col=set1[1],cex=0.5)
circos(R=300, cir="hg19",type="b3",W=40,mapping=mi_loc,B=TRUE,
       col.v=2,col=set2[5])
circos(R=300, cir="hg19",type="s2",W=40,mapping=mi_loc,B=FALSE,
       col=set2[6],cex=0.5)
circos(R=250, cir="hg19",type="b3",W=40,mapping=mRNA_loc,B=TRUE,
       col.v=2,col=set3[5])
circos(R=250, cir="hg19",type="s2",W=40,mapping=mRNA_loc,B=FALSE,
       col=set3[4],cex=0.5)
circos(R=200, cir="hg19",type="b2",W=40,mapping=sur_P,B=TRUE,
       col=c('#FB6467','#82491E'),
       lwd=2,cutoff=0.05, col.v=3)
circos(R=150, cir="hg19",type="b2",W=40, mapping=diff_fc,B=TRUE,
       col=c('#E64B35','#01A087'), lwd=2,cutoff=1,col.v = 3)
circos(R=100, cir="hg19",type="b",W=40,
       mapping=gene_im,B=TRUE, col='#BC3C28',lwd=0.01,col.v=3)
```

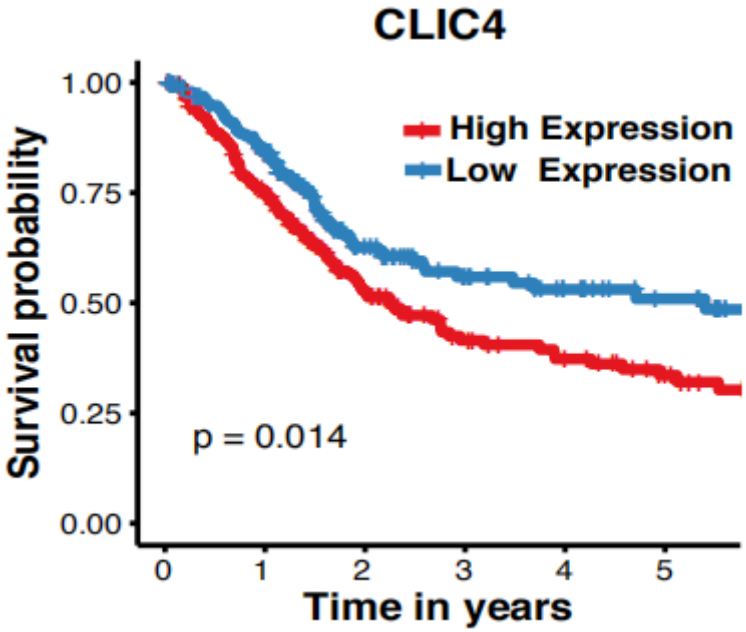


```
rm(list=ls())
options(stringsAsFactors = F)
rt <- read.csv("TCGAForSur.csv",header = T)
library(survminer)
library(survival)

rt$CLIC4 <- ifelse(rt$CLIC4 > median(rt[,"CLIC4"]),"high","low")
rt$GATA3 <- ifelse(rt$GATA3  > median(rt[,"GATA3"]), "high","low")
rt$PALLD <- ifelse(rt$PALLD > median(rt[,"PALLD"]),"high","low")
rt$MIR100HG <- ifelse(rt$MIR100HG > median(rt[,"MIR100HG"]),"high","low")
rt$AC010326.3 <- ifelse(rt$AC010326.3  > median(rt[,"AC010326.3"]), "high","low")
rt$AC073335.2 <- ifelse(rt$AC073335.2> median(rt[,"AC073335.2"]),"high","low")
rt$hsa-miR-141-3p <- ifelse(rt$hsa-miR-141-3p > median(rt[,"hsa-miR-141-3p"]),"high","low")
rt$hsa-miR-200c-3p<- ifelse(rt$hsa-miR-200c-3p  > median(rt[,"hsa-miR-200c-3p"]), "high","low")
rt$hsa-miR-141-5p <- ifelse(rt$hsa-miR-141-5p > median(rt[,"hsa-miR-141-5p"]),"high","low")



fit_GATA3 <- survfit(Surv(futime, fustat) ~ GATA3 , data = rt)
fit_PALLD <- survfit(Surv(futime, fustat) ~ PALLD , data = rt)
fit_CLIC4 <- survfit(Surv(futime, fustat) ~ CLIC4 , data = rt)
fit_GATA3 <- survfit(Surv(futime, fustat) ~ GATA3 , data = rt)
fit_MIR100HG <- survfit(Surv(futime, fustat) ~ MIR100HG, data = rt)
fit_AC010326.3 <- survfit(Surv(futime, fustat) ~ AC010326.3 , data = rt)
fit_hsa-miR-141-3p <- survfit(Surv(futime, fustat) ~ hsa-miR-141-3p , data = rt)
fit_hsa-miR-200c-3p <- survfit(Surv(futime, fustat) ~ hsa-miR-200c-3p , data = rt)
fit_hsa-miR-141-5p <- survfit(Surv(futime, fustat) ~ hsa-miR-141-5p , data = rt)


pdf(file="GATA3_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_GATA3, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c( "#377EB8","#E41A1C"),
                     legend.labs =c("GATA3 Low Expression", "GATA3 HighExpression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
```

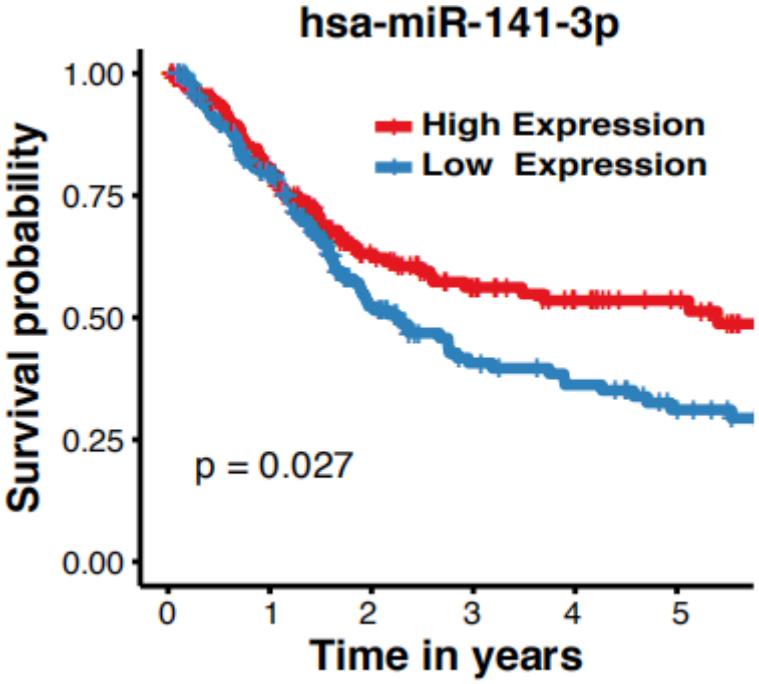
```
                font.legend = c(16, "bold"))

ggsurv
dev.off()
```



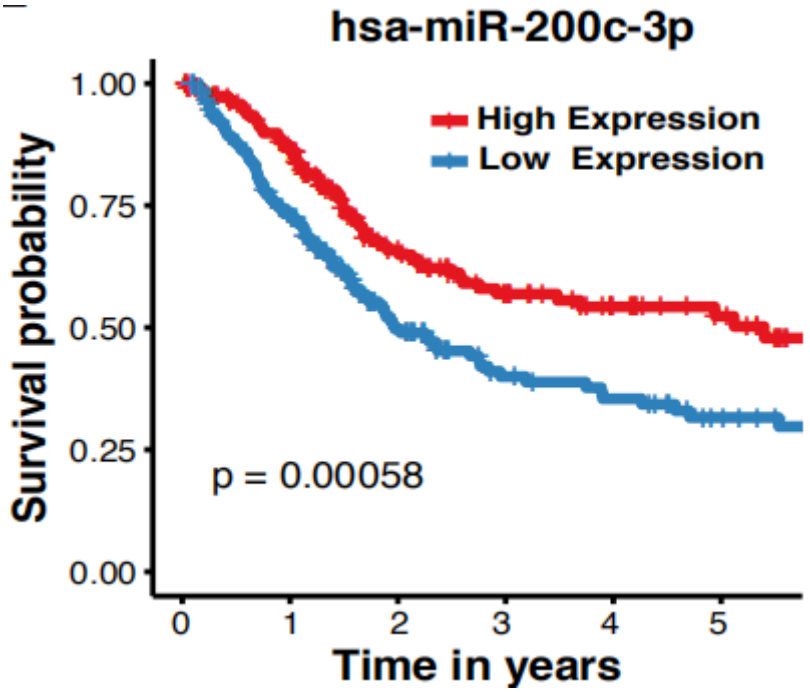

```
pdf(file="CLIC4_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_CLIC4, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("GATA3 HighExpression", "GATA3 LowExpression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```



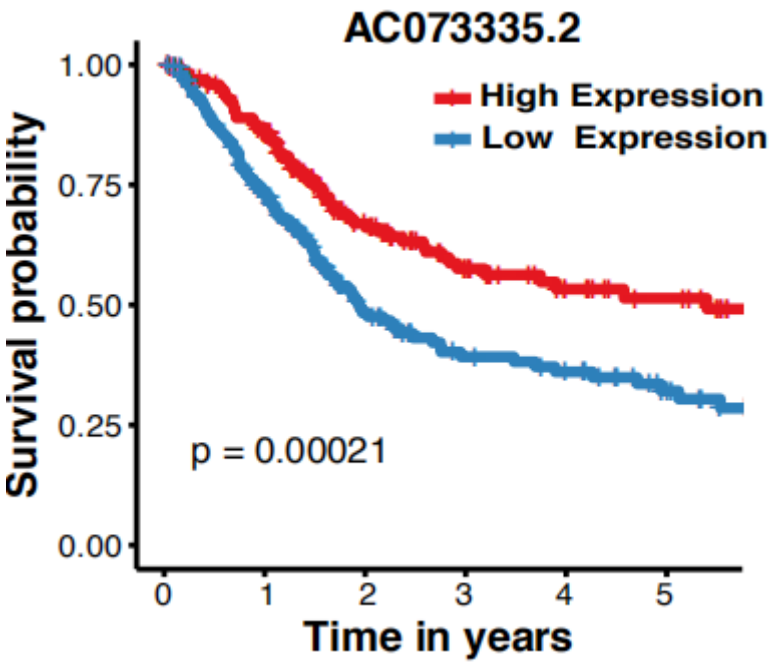

```
pdf(file="hsa-miR-141-3p_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_hsa-miR-141-3p, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("hsa-miR-141-3p High Expression", "GATA3 Low Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```



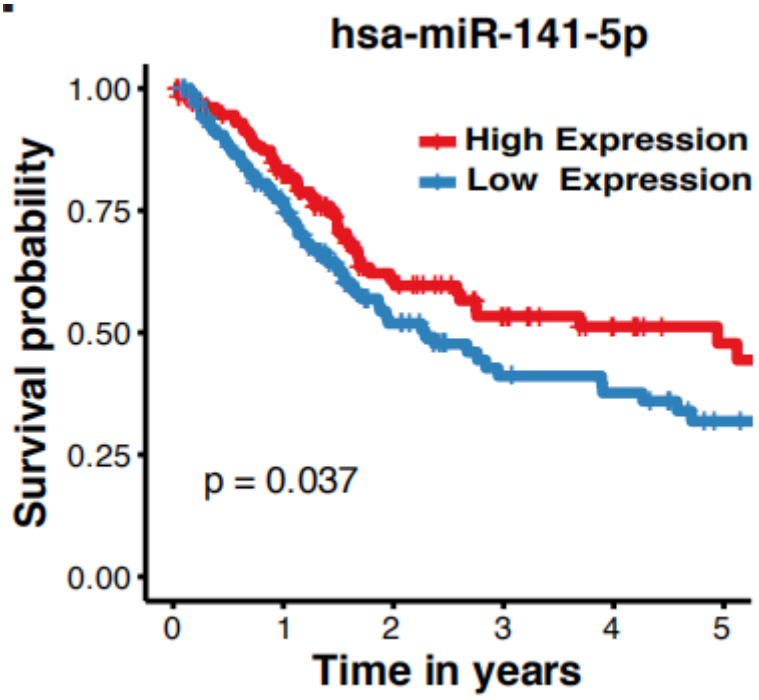

```
pdf(file="hsa-miR-200c-3p_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_hsa-miR-200c-3p, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("hsa-miR-200c-3p High Expression", "hsa-miR-200c-3p Low
Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```



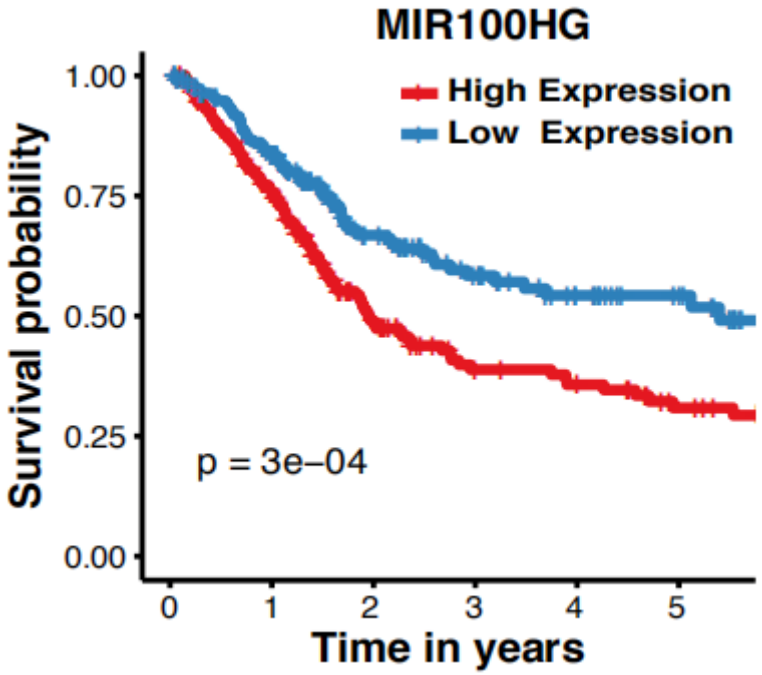

```
pdf(file="AC073335.2_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_AC073335.2, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("AC073335.2 High Expression", "AC073335.2 Low Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```



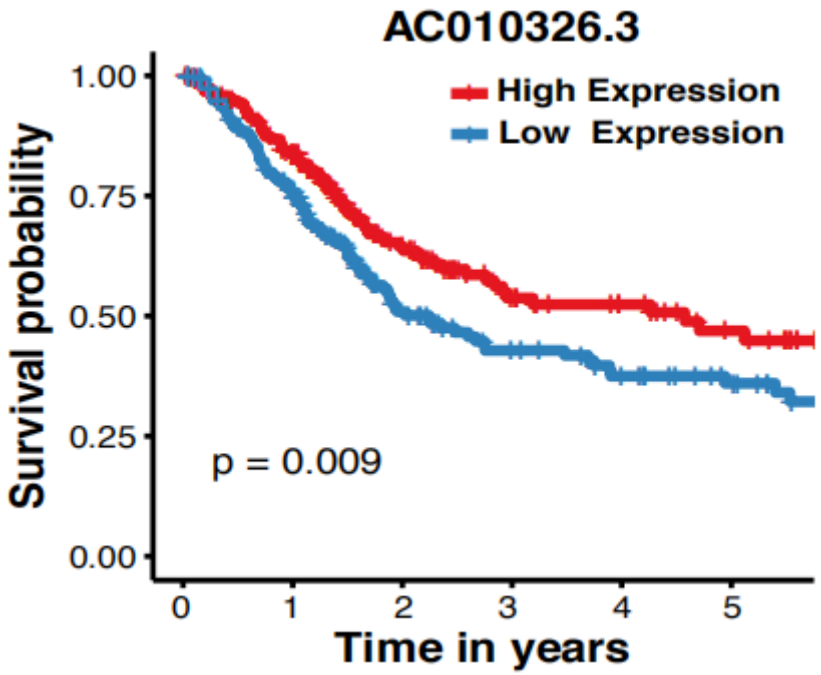

```
pdf(file="hsa-miR-141-5p_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_hsa-miR-141-5p, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("hsa-miR-141-5p High Expression", "hsa-miR-141-5p Low
Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```




```
pdf(file="MIR100HG_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_MIR100HG, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("MIR100HG High Expression", "MIR100HG Low Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```




```
pdf(file="AC010326.3_survival.pdf",height = 8, width = 8)
par(oma=c(2,2,1,2),mar=c(5,4,4,2))

ggsurv <- ggsurvplot(fit_AC010326.3, data = rt,
                     pval = T,
                     xlim = c(0,2000),
                     break.time.by = 365,
                     xlab = "Time in days",
                     palette = c("#E41A1C", "#377EB8"),
                     legend.labs =c("AC010326.3 High Expression", "AC010326.3 Low Expression"))

ggsurv <- ggpar( ggsurv,
                 font.y  = c(16, "bold"),
                 font.x  = c(16, "bold"),
                 legend = "top",
                 font.legend = c(16, "bold"))

ggsurv
dev.off()
```


AC010326.3
High Expression
Low Expression
Survival probability
1.00
0.75
0.50
0.25
0.00
p = 0.009
0
1
2
3
4
5
Time in years