ML for Cancer prediction

Edris Sharif Rahamani

```
# Package installation
# install.packages(c("MBMethPred","umap"))

# Loading package
suppressMessages({
  require(readr)
  require(caTools)
  require(caret)
  require(randomForest)
  require(MBMethPred)
  require(rgl)
  require(umap)
  require(ggplot2)
  require(parallel)
})
set.seed(1234)
# Reading the train and label data
train <- data.frame(read_tsv("train_data.tsv"))
```

```
## New names:
## Rows: 16340 Columns: 1515
## -- Column specification
## -------------------------------------------------------- Delimiter: "\t" chr
## (1): ...1 dbl (1514): S1343, S1344, S1345, S1348, S135, S1353, S1354, S1355,
## S1356, S1...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
rownames(train) <- train[,1]
train <- train[,-1]
train <- data.frame(t(train))

class <- data.frame(read_tsv("train_label.tsv"))
```

```
## New names:
## Rows: 1514 Columns: 4
## -- Column specification
## -------------------------------------------------------- Delimiter: "\t" chr
## (4): ...1, id, sample_type, X_primary_disease
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
colnames(class)[1] <- "sample"
```

```r
identical(rownames(train), class$sample)
```

```
## [1] TRUE
```

```r
# Median Absolute Deviation (MAD) analysis -->
# To select 3000 most variable genes

mad <- data.frame(mad = sapply(train, mad))
mad <- cbind(rownames(mad), mad)
mad <- mad[order(mad$mad, decreasing = T),]

# Selecting the top 3000 genes and sub-setting the train set
mad <- rownames(mad)[1:3000]
train <- train[, which(colnames(train) %in% mad)]

# Adding the labels to the train data
table(class$X_primary_disease)
```

```
##
##    breast invasive carcinoma          lung adenocarcinoma
##                          897                          315
## lung squamous cell carcinoma
##                          302
```

```r
train$class <- factor(class$X_primary_disease,
                      labels = c("breast_invasive",
                                 "lung_adenocarcinoma",
                          "lung_squamous_cell_carcinoma"))

# Feature selection using Random Forest model
fac <- ncol(train)
split <- sample.split(train[, fac], SplitRatio = 0.8)
training_set <- subset(train, split == TRUE)
test_set <- subset(train, split == FALSE)
rfimp <- randomForest(class ~ .,
                      data = training_set,
                      ntree = 300,
                      importance=TRUE)
y_pred <- predict(rfimp, newdata = test_set[-fac])
MBMethPred::ConfusionMatrix(y_true = test_set[, fac],
                            y_pred = y_pred)
```

```
##                               y_pred
## y_true                         breast_invasive lung_adenocarcinoma
##   breast_invasive                          179                   0
##   lung_adenocarcinoma                        1                  61
##   lung_squamous_cell_carcinoma               0                   3
##                               y_pred
## y_true                         lung_squamous_cell_carcinoma
##   breast_invasive                                         0
##   lung_adenocarcinoma                                     1
##   lung_squamous_cell_carcinoma                           57
```

```
##                              Accuracy Precision Sensitivity F1_Score
## breast_invasive                 0.997     0.994       1.000    0.997
## lung_adenocarcinoma             0.983     0.953       0.968    0.961
```

```
## lung_squamous_cell_carcinoma    0.987     0.983       0.950    0.966
##                              Specificity AUC_average
## breast_invasive                    0.992       0.987
## lung_adenocarcinoma                0.987       0.987
## lung_squamous_cell_carcinoma       0.996       0.987
```

```
imp <- varImp(rfimp)
imp <- imp[imp != 0,]
imp <- na.omit(imp)
# Top genes
head(rownames(imp))
```

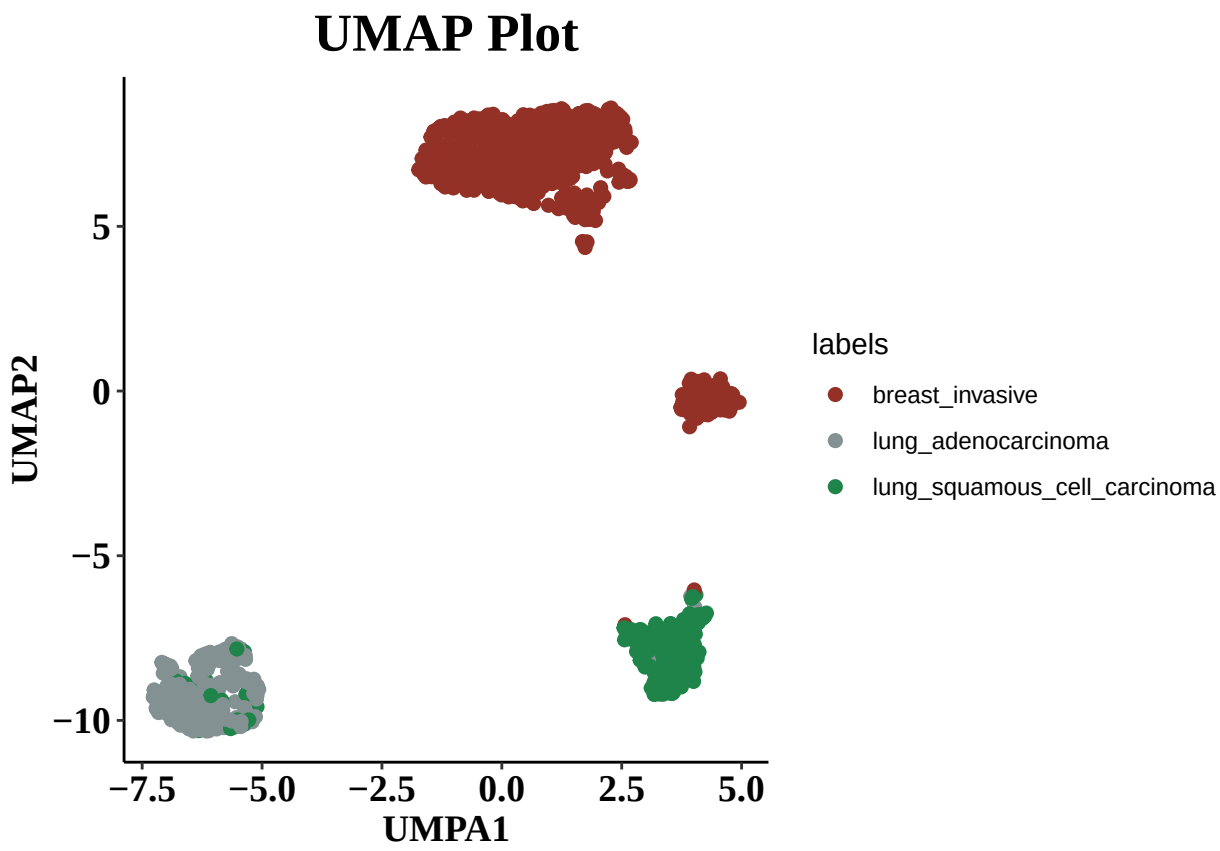
```
## [1] "A2ML1"  "AADAC"  "ABAT"   "ABCA12" "ABCA13" "ABCA4"
```

```
nrow(imp)
```

```
## [1] 1365
```

```
# Sub-setting the train data based on the important features
train <- train[, c(which(colnames(train) %in% rownames(imp)), fac)]
# 3D visualization using t-SNE and K-Means clustering
MBMethPred::TSNEPlot(File = train[, -ncol(train)], NCluster = 3)
# Leave the R window open and run the below line to save the 3D plot
rgl::rgl.snapshot('tsne3d.png', fmt = 'png')

# Visualization of train data using UMAP and sample labels
umap_plot <- function(counts,
                      labels,
                      n_neighbors = 10){
  umap_result <- umap(counts,
                      n_neighbors = n_neighbors,
                      labels = labels)
  umap_df <- data.frame(umap_result$layout, Clusters = labels)
  my_pal <- c("#943126", "#839192", "#1E8449", "#9C640C")
  g <- ggplot(umap_df, aes(x = X1, y = X2, color = labels, fill = labels)) +
    geom_point(size = 2) +
    labs(x= "UMPA1", y = "UMAP2", title = "UMAP plot") +
    scale_fill_manual(values = my_pal) +
    scale_color_manual(values = my_pal) +
    theme_classic() +
    theme(axis.line = element_line(linetype = "solid"),
                  axis.title = element_text(family = "Times",
                  size = 14, face = "bold"), axis.text = element_text(family = "Times",
                  size = 14, face = "bold", colour = "black"),
                  plot.title = element_text(family = "Times",
                  size = 20, face = "bold", hjust = 0.5)) +
    labs(title = "UMAP Plot")
  pdf("UMAP.pdf", width = 10, height = 8)
  print(g)
  dev.off()
  return(g)
}
umap_plot(counts = train[, -ncol(train)],
          labels = train$class,
          n_neighbors = 30)
```




```
# Training a Support Vector Machines model with 10-folds cross-validation
# And predication of the test_data
SupportVectorMachineModel <- function(Data,
                                      SplitRatio,
                                      CV,
                                      NCores,
                                      NewData){
  Data$class<- factor(Data$class)
  fac <- ncol(Data)
  if(!is.null(NewData)){
    if(colnames(NewData)[1] != "Gene") {
      stop('Please prodide correct NewData file.')
    } else {
      rownames(NewData) <- NewData$Gene
      NewData <- NewData[,-1]
      common_gene <- which(colnames(Data) %in% rownames(NewData))
      common_new <- which(rownames(NewData) %in% colnames(Data)[-fac])
      Data <- Data[, c(common_gene, fac)]
      NewData <- NewData[common_new, ] %>%
        t() %>%
        data.frame()
    }
  }
  fac <- ncol(Data)
  split <- sample.split(Data[, fac], SplitRatio = SplitRatio)
  training_set <- subset(Data, split == TRUE)
```

```
  test_set <- subset(Data, split == FALSE)
  folds <- createFolds(Data[,fac] , CV)
  cv <- mclapply(folds, function(x){
    training_fold <- training_set[-x, ]
    test_fold <- test_set[-x, ]
    formula <- as.formula(paste0(names(Data)[fac], " ~ ."))
    classifier <- e1071::svm(formula = formula,
                             data = training_fold,
                             type = "C-classification",
                             kernel = "linear",
                             cost = 0.01,
                             epsilon = 0.001,
                             na.action = na.omit,
                             scale = FALSE)
    y_pred <- predict(classifier, newdata = test_fold[-fac])
    conta <- table(test_fold[, fac], y_pred)
    result <- ConfusionMatrix(test_fold[, fac], y_pred)
    if(!is.null(NewData)) {
      y_pred_NewData <- predict(classifier, newdata = NewData)
    } else {
      y_pred_NewData <- NULL
    }
    allresult <- list(ConfusionMat = conta, result = result, pnewdata = y_pred_NewData)
    return(allresult)
  }, mc.cores = NCores)
}

# Reading the test data for predicting the Id labels
test_data <- data.frame(read_tsv("test_data.tsv"))
```

```
## New names:
## Rows: 16340 Columns: 601
## -- Column specification
## -------------------------------------------------------- Delimiter: "\t" chr
## (1): ...1 dbl (600): S1, S1001, S1005, S1006, S1012, S1020, S1021, S1037,
## S1039, S1040...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
colnames(test_data)[1] <- "Gene"

# Running the function
svm <- SupportVectorMachineModel(Data = train,
                                 SplitRatio = 0.8,
                                 CV = 10,
                                 NCores = 2,
                                 NewData = test_data)
# Model performance
ModelMetrics(Model = svm)
```

```
## $ConfusionMatrix
##                              y_pred
##                               breast_invasive lung_adenocarcinoma
##   breast_invasive                         157                   0
```

```
##   lung_adenocarcinoma                        0                  53
##   lung_squamous_cell_carcinoma               0                   5
##                               y_pred
##                                lung_squamous_cell_carcinoma
##   breast_invasive                                         0
##   lung_adenocarcinoma                                     3
##   lung_squamous_cell_carcinoma                           47
##
## $ModelPerformance
##                              Accuracy Precision Sensitivity F1_Score
## breast_invasive                 0.998     0.997       1.000    0.998
## lung_adenocarcinoma             0.969     0.913       0.942    0.927
## lung_squamous_cell_carcinoma    0.971     0.946       0.906    0.925
##                              Specificity AUC_average
## breast_invasive                    0.996       0.975
## lung_adenocarcinoma                0.976       0.975
## lung_squamous_cell_carcinoma       0.987       0.975
```

```r
# Predicting the test_data results
prediction <- NewDataPredictionResult(Model = svm)
prediction <- cbind(rownames(prediction), prediction)
colnames(prediction) <- c("ID", "label")
write_csv(prediction, "ML_predicton.csv")
```