```r
################################################################################
######################### AIC vs Early Stopping  ##############################
################################################################################
rm(list = ls())

library(MASS)
library(glmnet)
library(mboost)


# define a performance measure
MSE <- function(y, yhat){

  mean((y - yhat)^2)

}


M = 1000  # number of simulations
B = 250  # number of maximum boosting iterations
corr = FALSE  # decide whether to use an uncorrelated X or not

# to store performance of the models at every iter
MSEtrain = matrix(nrow = M, ncol = B)
MSEval = matrix(nrow = M, ncol = B)

# to store optimal stopping point for different criteria
mstop = data.frame('AIC' = double(M), 'estp_insp' = double(M), 'min' = double(M))

# to store MSE values for different criteria
error = data.frame('AIC' = double(M), 'estp_insp' = double(M), 'min' = double(M))


# to store performance of each optimal models
Acc = data.frame('AIC' = double(M), 'estp_insp' = double(M))
Corr = data.frame('AIC' = double(M), 'estp_insp' = double(M))

for (m in 1:M) {

  ########################################
  ## generate the data
  ########################################

  n = 20  # number of observations
  p = 10   # dimension
  s = 2  # error sd


  ## Generate the train data ##

  # X variables
  if (corr == FALSE) {

    # X without correlation
    Xtrain = mvrnorm(n = n, mu = rep(0,p), Sigma = diag(p))
    a = 1  # scaling factor
  } else {

    # define a block covariance matrix of X
    a = 0.779  # scaling factor
    b = 0.677  # correlation in the second diagonals
    c = 0.323  # correlation in the third diagonals
    Sig = diag(p)
    diag(Sig[-1,]) = b
    diag(Sig[,-1]) = b
    diag(Sig[-c(1,2),]) = c
    diag(Sig[,-c(1,2)]) = c

    # X  with block correlation
    Xtrain = mvrnorm(n = n, mu = rep(0,p), Sigma = Sig)
  }

  # define the underlying relationship f btw Y and X
  f <- function(X) {
    a * (1 + 5*X[,1] + 2*X[,2] + X[,3])
  }

  # eps and Y
  epsTrain = rnorm(n = n, mean = 0, sd = s)
  ftrain = f(Xtrain)
  Ytrain = f(Xtrain) + epsTrain

  # named data frame for the train data
  Dtrain = data.frame('Y' = Ytrain, 'X' = Xtrain)


  ## generate the validation data (for prediction) ##

  if (corr == FALSE){

    # X variables without correlation
    Xval = mvrnorm(n = n, mu = rep(0,p), Sigma = diag(p))

  } else {

    # X variables with block correlation
    Xval = mvrnorm(n = n, mu = rep(0,p), Sigma = Sig)
  }

  # generate eps and Y
  epsVal = rnorm(n = n, mean = 0, sd = s)
  fval = f(Xval)
  Yval = f(Xval) + epsVal

  # named data frame for the test data
  Dval = data.frame('Y' = Yval, 'X' = Xval)

  ########################################
  ## L2 Boosting
  ########################################

  # train the model
  # options(mboost_dftraceS = TRUE)
  L2 = glmboost(Y ~ ., data = Dtrain)
  estp_insp = c()  # save 1 if in sample MSE becomes smaller than a pre-set threshold, e.g. error variance

  for (b in 1:B) {
    l2 = L2[b]  # model at iteration b
    y_fit = predict(l2, newdata = Dtrain[-1])  # fit at iter b
    MSEtrain[m,b] = MSE(Dtrain$Y, y_fit)  # MSE on the training set
    estp_insp[b] = ifelse(MSEtrain[m,b] <= s^2, 1, 0)

    y_val = predict(l2, newdata = Dval[-1])  # predict at iter b on the validation set
    MSEval[m,b] = MSE(fval, y_val)

  }

  # in sample early stopping
  idx_insp = which(estp_insp == 1)[1]
  mstop$estp_insp[m] = idx_insp - 1
  error$estp_insp[m] = MSEval[m,mstop$estp_insp[m]]


  # corrected AIC
  mstop$AIC[m] = mstop(AIC(L2, method = 'corrected'))
  error$AIC[m] = MSEval[m,mstop$AIC[m]]


  # minimun MSE
  error$min[m] = min(MSEval[m,])


  ## save the graphics
  # setwd('C:/Users/Seokhee2/Desktop/graphics')
  # file.name = paste0('n',n,'p',p,'m',m,'B',B,'corr',corr,'.png')
  # png(file.name,width = 1200, height = 834, pointsize = 25)
  plot(MSEval[m,], type = 'l', ylim = c(0,20), xlab = '# of iteration',
       ylab = 'Mean MSE')
  abline(v = mstop$AIC[m], col = 'red')
  abline(h = error$AIC[m], col = 'red', lty='dotted')
  abline(v = mstop$estp_insp[m], col = 'blue')
  abline(h = error$estp_insp[m], col = 'blue', lty='dotted')
  abline(h = error$min[m], col = 'black', lty = 'dotted')
  abline(h = s^2, col = 'grey', lty = 'dotted')
  legend("topright",
         c("L2 Boosting / min MSE", "corrected AIC", 'in-sample early stopping', 'error variance'),
         fill = c("black", "red", 'blue', 'grey'))
  # dev.off()


}

# import the results as a .csv file
AIC_vs_ES = data.frame('mstop_AIC' = mstop$AIC, 'MSE_AIC' = error$AIC,
                       'mstop_ES' = mstop$estp_insp, 'MSE_ES' = error$estp_insp,
                       'MSE_min' = error$min)

# write.csv(AIC_vs_ES, 'AIC_vs_ES.csv')

# make a .gif of the graphical results
```

```r
# library(gifski)
# png_files <- list.files('C:/Users/Seokhee2/Desktop/graphics', pattern = ".*png$", full.names = TRUE)
# gifski(png_files, gif_file = "AIC_vs_ES.gif", width = 800, height = 600, delay = 2)
```