```
# Part 1 - learning the model

randomForestMods = function(x, t, n, d) {
  # x is training data
  # t is number of trees to build
  # n is number of instances to select for each tree
  # d is number of attributes to use

  ls = list() # rpart models

  for (idx in 1:t) {

    # randomly selected with replacement
    xSample = x[sample(nrow(x), n, replace = TRUE),]

    # attributes randomly selected without replacement
    a = sample(names(x)[-length(names(x))], d, replace = FALSE)

    # a = sort(a)

    # build formula
    frm = paste0(names(x)[length(names(x))], "~")
    for (j in 1:length(a)) {
      frm = paste0(frm, a[j], "+")

    }
    frm = substr(frm, 1, nchar(frm)-1) # remove last character
    print(frm)

    # build model
    a = append(a, names(x)[length(names(x))])
    mod = rpart(frm, xSample[, a])

    # append to result ls
    ls[[idx]] = mod
  }

  return(ls)
}

# Part 2 - predicting new instances

predictIns = function(ls, x) {
  # ls is the list of random forest models
  # x is the test data

  # Store the prediction in data frame
  predsDf = data.frame(matrix(ncol = length(ls), nrow = nrow(x)))

  # set the column names
  cols = c()
  for (i in 1:length(ls)) {
    cols = c(cols, paste("mod", i))
  }
  colnames(predsDf) = cols

  # perform predictions
  for(i in 1:length(ls)) {
    pred = predict(ls[[i]], x, type="vector")
    predsDf[i] = pred
  }

  # label predictions
  for(i in 1:ncol(predsDf)) {
    ind = predsDf[, i] <= 0.5
    predsDf[ind, i] = -1
    predsDf[!ind, i] = 1
  }

  # calculate majority vote
  sum = rowSums(predsDf)
  output = sign(sum)

  # add sum, output column to predictions data frame
  predsDf["sum"] = sum
  predsDf["output"] = output

  return(predsDf)
}

# load the data set
trainDs = read.table("hw06dataTrain.txt", header = TRUE)
testDs = read.table("hw06dataTest.txt", header = TRUE)

# get the models
lsMods = randomForestMods(x = trainDs, t = 21, n = 3000, d = 3)

# predict the new instances
resPredictIns = predictIns(ls = lsMods, x = testDs)
print(resPredictIns)
```