```
# HEAVY METAL CLASSIFICATION CODE

# This version compiles all relevant code for submission

#### PACKAGE SETUP ####
# Load relevant packages (may need to install first)
library(MASS)
library(ggplot2)
library(Rtsne)
library(e1071)
library(tidyr)
library(Heatplus)
library(gplots)

#### FIG 5 & TABLE 1 SOURCE CODE ####

# NOTE: This is designed for four concentration levels
# and three replicates per concentration by default.
# This can apply to other configurations with some adjustments

# Import data from source
metal <- read.csv("~/Dropbox/Heavy Metals Classification/heavy_metals_new.csv")

# Set as data.frame
metal = as.data.frame(metal)

# Remove control cases (where concentration is 0)
metal = metal[-which(metal$Concentration==0),]

# Remove concentration column
metal = metal[,-2]

# Set variables according to parameters
c = 4 # Number of concentrations
m = 11 # Number of ions
r = 3 # Number of replicates per concentration
n = 30 # Number of iterations of SVM to perform

# Test run for a single, non-random instance of SVM
index = 1:nrow(metal)
testindex = c(seq(1,c*m*r,c*r),seq(1+c,c*m*r,c*r),seq(1+2*c,c*m*r,c*r),seq(1+3*c,c*m*r,c*r))
trainset = metal[-testindex,]
testset = metal[testindex,]
```

```
svm.model = svm(Metal ~., data=trainset)
svm.pred = predict(svm.model, testset[,-1])
# confusion matrix
tab.metal = table(pred=svm.pred, true = testset[,1])
tab.metal
classAgreement(tab.metal,match.names=T)

svm.pred = as.character(svm.pred)
svm.actual = metal[testindex,1]

diag = sum(svm.pred == svm.actual)/dim(testset)[1] # Equivalent of diag
kappa = (diag-(1/dim(testset)[1])) / (1-(1/dim(testset)[1])) # Equivalent of Cohen's kappa

# The resulting table can now serve as a template for the complete version
tab.metal[,] = 0 # clear the table
data = metal

set.seed(0)
for(i in 1:n)
{
  # Randomly select 1 of the 3 replicates in each metal-concentration
  # combination to serve as test data
  # The remaining replicates will act as training data
  testindex = c(seq(1,c*m*r,c*r),seq(1+c,c*m*r,c*r),
                seq(1+2*c,c*m*r,c*r),seq(1+3*c,c*m*r,c*r)) +
    sample(0:(c-1),c*m,T)
  testset = data[testindex,]
  trainset = data[-testindex,]

  svm.model = svm(Metal ~., data=trainset)
  svm.pred = predict(svm.model, testset[,-1])
  #confusion matrix
  tab.metal = tab.metal+ table(pred=svm.pred, true = testset[,1])
}
tab.metal

# TABLE 1 FOLLOW-UP
# Percentage correctly classified
diag(tab.metal)*100/(n*4)


#### CODE FOR CLASSIFICATION RATES BY ION & CONCENTRATION ####

metal <- read.csv("~/Dropbox/Heavy Metals Classification/heavy_metals_new.csv")
```

```
metal = as.data.frame(metal)

# Remove control cases (where concentration is 0)
metal = metal[-which(metal$Concentration==0),]

# NOTE: Use this code to combine the names of Metal and Concentration
# This allows each Metal-Concentration combo to be a separate factor level
metal.all = metal # Make a separate copy for the all concentrations version
metal.all$Metal <- paste(metal.all$Metal,metal.all$Concentration, sep="-")

metal.all$Metal = as.factor(metal.all$Metal)


metal = metal[,-2]
metal.all = metal.all[,-2]

testindex = c(seq(1,132,12),seq(4,132,12),seq(7,132,12),
          seq(10,132,12)) + sample(0:2,44,T)
testset = metal.all[testindex,]
trainset = metal.all[-testindex,]

svm.model = svm(Metal ~., data=trainset)
svm.pred = predict(svm.model, testset[,-1])
#confusion matrix
tab.metal = table(pred=svm.pred, true = testset[,1])


svm.metals1= function(metal,g,n, tab.metal){
 if(g!= 4)
 {
  stop("Invalid number of groups g!")
 }

 tab.metal[,]=0

 for(i in 1:n)
 {
  if(g==4)
  {
   testindex = c(seq(1,132,12),seq(4,132,12),seq(7,132,12),
              seq(10,132,12)) + sample(0:2,44,T)
  }
  testset = metal.all[testindex,]
  trainset = metal.all[-testindex,]
```

```
   svm.model = svm(Metal ~., data=trainset)
   svm.pred = predict(svm.model, testset[,-1])
   #confusion matrix
   tab.metal = tab.metal + table(pred=svm.pred, true = testset[,1])
  }
  return(tab.metal)
}

# Median Results
set.seed(0)
summary4 = svm.metals1(metal,4,1000, tab.metal) # 1-1000    diag = .2222      kappa = .2000

write.csv(summary4, file="individual_ion_conc.csv")

#### FIG 6 SOURCE CODE ####

#IMPORTANT: Must Import Mixture Dataset
#mixtureChoice = 1 denotes Pb and Cu
#mixtureChoice = 2 denotes Cd and Ni
mixtureChoice = 2
mixture_dataset = as.data.frame(Mixture_Data_PbCu_CdNi) #Import Mixture Dataset
metal = as.data.frame(Compiled_Heavy_Metals_all_STATA) #Import Single Species Dataset
metal = metal[-which(metal$Concentration==0),] # Remove control cases (where concentration
is 0)


#Single Species Grouping
#Reformat single species grouping
trainset = metal
trainset = trainset[order(trainset$Metal), ]
rownames(trainset) = NULL
Trainset1 = trainset[which(trainset$Metal == "Cr(VI)"), ]
Trainset1 = rbind(Trainset1, trainset[which(trainset$Metal == "Cu(II)"), ])
Trainset1 = rbind(Trainset1, trainset[which(trainset$Metal == "Ni(II)"), ])
Trainset1 = rbind(Trainset1, trainset[which(trainset$Metal == "Pb(II)"), ])
Trainset1 = rbind(Trainset1, trainset[which(trainset$Metal == "Zn(II)"), ])
Trainset1 = Trainset1[order(Trainset1$Metal), ]
rownames(Trainset1) = seq(length = nrow(Trainset1))
Trainset1 = Trainset1[order(Trainset1$Metal), ]
Trainset2 = trainset[which(trainset$Metal == "Co(II)"), ]
Trainset2 = rbind(Trainset2, trainset[which(trainset$Metal == "Mg"), ])
rownames(Trainset2) = seq(length = nrow(Trainset2))
Trainset3 = trainset[which(trainset$Metal == "Cd(II)"), ]
```

```
Trainset3 = rbind(Trainset3, trainset[which(trainset$Metal == "K"), ])
Trainset3 = rbind(Trainset3, trainset[which(trainset$Metal == "Fe(III)"), ])
rownames(Trainset3) = seq(length = nrow(Trainset3))
SingleSpeciesGroups = Trainset1
SingleSpeciesGroups = rbind(SingleSpeciesGroups, Trainset2)
SingleSpeciesGroups = rbind(SingleSpeciesGroups, Trainset3)

#Heavy Metal Formatting
metal_mix = mixture_dataset[4:9,]
metal_mix = rbind(metal_mix, mixture_dataset[13:15,])
rownames(metal_mix) = seq(length=nrow(metal_mix))
metal_mix$Ion = NA
metal_mix[1:3,1] = 'Pb(II) + Cu(II) (Low)'
metal_mix[4:6,1] = 'Pb(II) + Cu(II) (High)'
metal_mix[7:9,1] = 'Cd(II) + Ni(II) (Mine)'
SingleSpeciesGroups$Metal = paste(SingleSpeciesGroups$Metal, "-",
SingleSpeciesGroups$Concentration, "ppb")
SingleSpeciesGroups = rbind(SingleSpeciesGroups, metal_mix)
rownames(SingleSpeciesGroups) = seq(length=nrow(SingleSpeciesGroups))
SingleSpeciesGroups = SingleSpeciesGroups[,-c(3:4)]

# tSNE for Mixture Samples
data_input = SingleSpeciesGroups
type = data_input[,1]    # Labels for elements
data = data_input[,3:6]  # All other variables except concentration
# Color Key for elements in plots
# ggplot2's normal color picker process
gg_color_hue <- function(n) {
  hues = seq(15, 375, length = n + 1)
  hcl(h = hues, l = 65, c = 100)[1:n]
}

# Select 9 distinct hues for metals (black for K, Mg)
cols = gg_color_hue(6)
m.color = c(cols[1:10], "#000000", "#000000", cols[11:20])

# Shape Key for elements in plots
# m.shape = c(rep(16,6),15,17,rep(16,3)) # all circles except controls
m.shape = seq(0,7,1) # Different shapes for each, but not as bold

m1 = as.matrix(data)
tsne.m1 = Rtsne(m1, verbose=T, check_duplicates = F, perplexity = 25,
                theta=0.5, max_iter=2000)
```

```
plotset = tsne.m1$Y
plotset = data.frame(plotset)
x=plotset[,1]
y=plotset[,2]
tsne_out=cbind(plotset[,1:2], type)
tsne_out=data.frame(tsne_out)

if(mixtureChoice == 1){
  to.1 = subset(tsne_out, type == 'Pb(II) - 100 ppb', select=c(X1,X2,type))
  to.1 = rbind(to.1,subset(tsne_out, type == 'Pb(II) - 1000 ppb', select=c(X1,X2,type)))
  to.1 = rbind(to.1,subset(tsne_out, type == 'Cu(II) - 100 ppb', select=c(X1,X2,type)))
  to.1 = rbind(to.1,subset(tsne_out, type == 'Cu(II) - 1000 ppb', select=c(X1,X2,type)))
  to.1 = rbind(to.1,subset(tsne_out, type == 'Pb(II) + Cu(II) (High)', select=c(X1,X2,type)))
  to.1 = rbind(to.1,subset(tsne_out, type == 'Pb(II) + Cu(II) (Low)', select=c(X1,X2,type)))

  plot1 = ggplot(to.1, main="Dim 2 vs Dim 1",aes(X1,X2)) +
    geom_point(aes(color=type, shape=type),size=3, alpha=1, stroke=1) +
    scale_color_manual(values=m.color) +
    scale_shape_manual(values=m.shape) +
    theme(panel.background = element_rect(fill = "white"),
          panel.grid.major=element_line(colour="grey", size=0.1),
          panel.grid.minor=element_line(colour="grey", size=0.1)) +
    labs(title = "t-SNE Plot: Pb(II) and Cu(II) - Mixture and Individual Data")

  plot1
}


if(mixtureChoice == 2){
  to.2 = subset(tsne_out, type == 'Cd(II) - 100 ppb', select=c(X1,X2,type))
  to.2 = rbind(to.2,subset(tsne_out, type == 'Cd(II) - 1000 ppb', select=c(X1,X2,type)))
  to.2 = rbind(to.2,subset(tsne_out, type == 'Ni(II) - 100 ppb', select=c(X1,X2,type)))
  to.2 = rbind(to.2,subset(tsne_out, type == 'Ni(II) - 1000 ppb', select=c(X1,X2,type)))
  to.2 = rbind(to.2,subset(tsne_out, type == 'Cd(II) + Ni(II) (Mine)', select=c(X1,X2,type)))

  plot1 = ggplot(to.2, main="Dim 2 vs Dim 1",aes(X1,X2)) +
    geom_point(aes(color=type, shape=type),size=3, alpha=1, stroke=1) +
    scale_color_manual(values=m.color) +
    scale_shape_manual(values=m.shape) +
    theme(panel.background = element_rect(fill = "white"),
          panel.grid.major=element_line(colour="grey", size=0.1),
          panel.grid.minor=element_line(colour="grey", size=0.1)) +
    labs(title = "t-SNE Plot: Cd(II) and Ni(II) - Mixture and Individual Data")
```

```
  plot1
}
```