Disaster Relief Project Part 1 and 2

Yihnew Eshetu

2/26/2020

Disaster Relief Project Part 1

```
library(ISLR)
library(regclass)
library(caret)
library(MASS)
library(class)
library(pROC)
library(ROCR)
library(nnet)
library(randomForest)
library(e1071)
```

Read in Haiti csv file

```
haiti.image = read.csv("HaitiPixels.csv")
attach(haiti.image)
```

Set seed and splitting

```
set.seed(10)
data.size = nrow(haiti.image)
split.size = data.size/10
permutation = sample(data.size)
haiti.image$Class = ifelse(haiti.image$Class == 'Blue Tarp', 'B', 'NB')
haiti.image$Class = as.factor(haiti.image$Class)
```

KNN 10x Cross-Validation

```
knn.function = function(kvalue) {
  knn.acc = 0
  for (i in 1:10) {
    haiti.image.sample = permutation[((i - 1) * split.size + 1) : (i * split.size)]

    haiti.image.test = as.data.frame(haiti.image[haiti.image.sample,])
    haiti.image.train = as.data.frame(haiti.image[-haiti.image.sample,])

    knn.pred = knn(haiti.image.train[-1], haiti.image.test[-1], cl =  haiti.image.train[,1], kvalue)
    knn.acc = knn.acc + mean(knn.pred == haiti.image.test[,1])
  }
  return((knn.acc/10)*100)
}

k.value.acc = c('1' = 0, '2' = 0, '3' = 0,
```

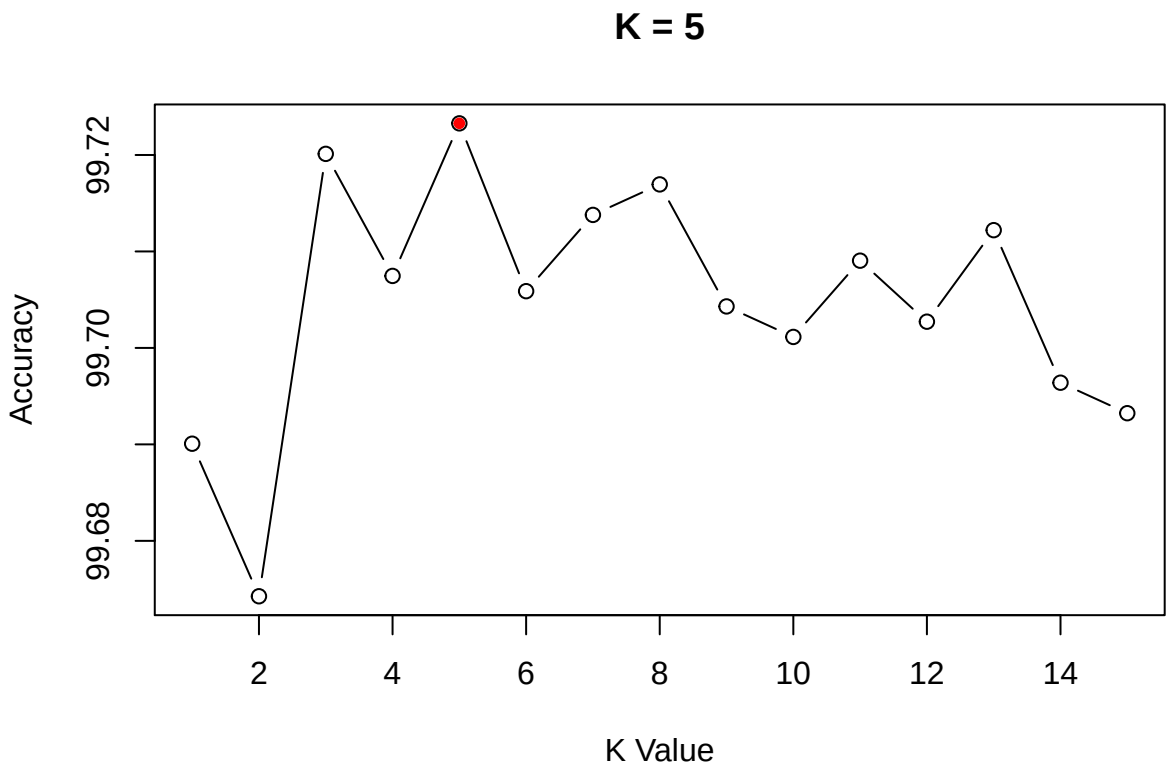
```
                  '4' = 0, '5' = 0, '6' = 0,
                  '7' = 0, '8' = 0, '9' = 0,
                  '10' = 0, '11' = 0, '12' = 0,
                  '13' = 0, '13' = 0, '15' = 0)

start_time = Sys.time()
for (k in 1:15){
  k.value.acc[k] = knn.function(k)
}
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 5.139288 mins
```

```
k.value.acc[which.max(k.value.acc)]
```

```
##        5
## 99.72328
```

```
plot(seq(1,15), k.value.acc,type = 'b', xlab = 'K Value', ylab = 'Accuracy', main = paste0('K = ', which
points(which.max(k.value.acc), k.value.acc[which.max(k.value.acc)], col ='red', pch = 20)
```



LDA 10x Cross-Validation

```
lda.function = function() {
  lda.acc = 0
  for (i in 1:10) {
    haiti.image.sample = permutation[((i - 1) * split.size + 1) : (i * split.size)]

    haiti.image.test = as.data.frame(haiti.image[haiti.image.sample,])
```

```
    haiti.image.train = as.data.frame(haiti.image[-haiti.image.sample,])

    lda.fit = lda(Class ~ Red + Green + Blue, data = haiti.image.train)
    lda.pred = predict(lda.fit, haiti.image.test)
    lda.acc = lda.acc + mean(lda.pred$class == haiti.image.test[,1])
  }
  return((lda.acc/10)*100)
}
start_time = Sys.time()
lda.function()
```

```
## [1] 98.39342
```

```
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 1.482143 secs
```

QDA 10x Cross-Validation

```
qda.function = function() {
  qda.acc = 0
  for (i in 1:10) {
    haiti.image.sample = permutation[((i - 1) * split.size + 1) : (i * split.size)]

    haiti.image.test = as.data.frame(haiti.image[haiti.image.sample,])
    haiti.image.train = as.data.frame(haiti.image[-haiti.image.sample,])

    qda.fit = qda(Class ~ Red + Green + Blue, data = haiti.image.train)
    qda.pred = predict(qda.fit, haiti.image.test)
    qda.acc = qda.acc + mean(qda.pred$class == haiti.image.test[,1])
  }
  return((qda.acc/10)*100)
}
start_time = Sys.time()
qda.function()
```

```
## [1] 99.45446
```

```
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 1.343 secs
```

Logistic Regression

```
log.function = function() {
  log.acc = 0
  for (i in 1:10) {
    haiti.image.sample = permutation[((i - 1) * split.size + 1) : (i * split.size)]

    haiti.image$Class = factor(haiti.image$Class)
    levels(haiti.image$Class)
    haiti.image.test = haiti.image[haiti.image.sample,]
    haiti.image.train = haiti.image[-haiti.image.sample,]
```

```
    log.fit = multinom(Class ~ Red + Green + Blue, data = haiti.image.train, family = binomial)
    log.pred = predict(log.fit, type = 'class', haiti.image.test)
    log.acc = log.acc + mean(log.pred == haiti.image.test[,1])
  }
  return((log.acc/10)*100)
}
start_time = Sys.time()
log.function()
```

```
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1177.543963
## iter  20 value 797.577923
## final  value 797.571866
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1162.717942
## iter  20 value 784.381302
## final  value 784.367786
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1198.958949
## iter  20 value 818.465065
## final  value 818.405685
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1159.226498
## iter  20 value 787.177774
## final  value 787.165317
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1171.158962
## iter  20 value 797.364037
## final  value 797.361038
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1188.779683
## iter  20 value 807.864165
## final  value 807.855856
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1187.867810
## iter  20 value 801.445716
## final  value 801.441734
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
```

```
## iter  10 value 1187.308613
## iter  20 value 800.540539
## final  value 800.530705
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1168.874654
## iter  20 value 766.390685
## final  value 766.351913
## converged
## # weights:  5 (4 variable)
## initial  value 39451.858076
## iter  10 value 1182.589362
## iter  20 value 799.483603
## final  value 799.473340
## converged
```

```
## [1] 99.52878
```

```
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 4.288774 secs
```

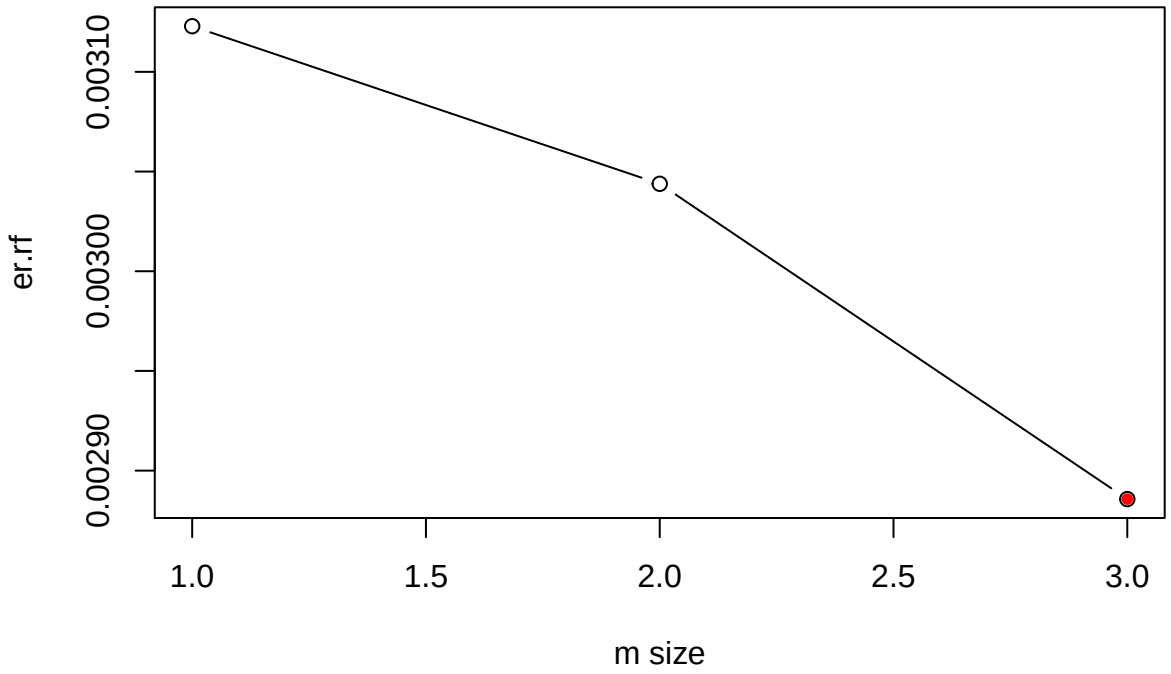
Disaster Relief Project Part 2

Random Forest 10x Cross-Validation

Pick the best m and ntree value

```
er.rf = NA
ntree.rf = NA
for (i in 1:3){
  set.seed(10)
  train.set = sample(nrow(haiti.image), size = nrow(haiti.image)*.6)
  haiti.image.train = haiti.image[train.set,]
  haiti.image.test = haiti.image[-train.set,]

  for (j in c(100, 200, 300, 400, 500)){
    low.er = 100
    low.ntree = 0

    haiti.image.rf = randomForest(Class ~ Red + Green + Blue, data = haiti.image.train,
                                  mtry = i, ntree = j, importance = TRUE)

    yhat.rf =  predict(haiti.image.rf, newdata = haiti.image.test)
    er = 1 - mean(yhat.rf == haiti.image.test$Class)
    low.er = ifelse(low.er < er, low.er, er)
    low.ntree = ifelse(low.er < er, low.ntree, j)}

  er.rf[i] = low.er
  ntree.rf[i] = low.ntree
}
plot(1:3, er.rf, type = 'b', xlab = 'm size')
points(which.min(er.rf),
```

```
      er.rf[which.min(er.rf)], col = 'red', pch = 20)
```



Random Forest 10x Cross-Validation

```
rf.function = function(m, n) {
  rf.acc = 0
  for (i in 1:10) {
    haiti.image.sample = permutation[((i - 1) * split.size + 1) : (i * split.size)]

    haiti.image.test = as.data.frame(haiti.image[haiti.image.sample,])
    haiti.image.train = as.data.frame(haiti.image[-haiti.image.sample,])

    rf.fit = randomForest(Class ~ Red + Green + Blue, data = haiti.image.train,
                          mtry = m, ntree = n, importance = TRUE)
    rf.pred = predict(rf.fit, newdata = haiti.image.test)
    rf.acc = rf.acc + mean(rf.pred == haiti.image.test[,1])
  }
  return((rf.acc/10)*100)
}
start_time = Sys.time()
which.min(er.rf)
```

```
## [1] 3
```

```
ntree.rf[which.min(er.rf)]
```

```
## [1] 500
```

```
rf.function(which.min(er.rf), ntree.rf[which.min(er.rf)])
```

```
## [1] 99.68691
```

```
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 2.865297 mins
```

SVM 10x Cross-Validation

Linear

```
cross = tune.control(cross = 10)
start_time = Sys.time()
haiti.image.svm.linear = tune(svm, Class ~ ., data =  haiti.image, kernel = 'linear',
                ranges = list(cost = c(1, 10, 25, 50)),
                tunecontrol = cross)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 1.593538 mins
```

```
haiti.image.svm.linear$best.parameters
```

```
##   cost
## 4   50
```

```
haiti.image.svm.linear.acc = (1 - haiti.image.svm.linear$best.performance)*100
haiti.image.svm.linear.acc
```

```
## [1] 99.53353
```

Polynomial

```
start_time = Sys.time()
haiti.image.svm.polynomial = tune(svm, Class ~ ., data = haiti.image, kernel = 'polynomial',
                          ranges = list(cost = c(1, 5, 10, 50),
                                        degree = c(2, 4, 6)),  tunecontrol = cross)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 7.648502 mins
```

```
haiti.image.svm.polynomial$best.parameters
```

```
##   cost degree
## 4   50      2
```

```
haiti.image.svm.polynomial.acc = (1 - haiti.image.svm.polynomial$best.performance)*100
haiti.image.svm.polynomial.acc
```

```
## [1] 99.38647
```

Radial

```
start_time = Sys.time()
haiti.image.svm.radial = tune(svm, Class ~ ., data = haiti.image, kernel = 'radial',
                        ranges = list(cost = c(11, 14, 16),
                                      gamma = c(8, 9, 10)),  tunecontrol = cross)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 7.466551 mins
```

```
haiti.image.svm.radial$best.parameters
```

```
##   cost gamma
## 9   16    10
```

```
haiti.image.svm.radial.acc = (1 - haiti.image.svm.radial$best.performance)*100
haiti.image.svm.radial.acc
```

```
## [1] 99.74226
```

Hold Out Data

Preprocessing and Cleaning of csv files

```
NB.57 = read.csv('orthovnir057_ROI_NON_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
NB.57$Class = 'NB'

B.67 = read.csv('orthovnir067_ROI_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
B.67$Class = 'B'

NB.67 = read.csv('orthovnir067_ROI_NOT_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
NB.67$Class = 'NB'

B.69 = read.csv('orthovnir069_ROI_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
B.69$Class = 'B'

NB.69 = read.csv('orthovnir069_ROI_NOT_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
NB.69$Class = 'NB'

B.78 = read.csv('orthovnir078_ROI_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
B.78$Class = 'B'

NB.78 = read.csv('orthovnir078_ROI_NON_Blue_Tarps.csv', skip = 7)[, c('B1', 'B2', 'B3')]
NB.78$Class = 'NB'
```

Merging all csv files into dataframe

```
hold.out.data = as.data.frame(rbind(NB.57, B.67, NB.67, B.69, NB.69, B.78, NB.78))
hold.out.data$Class = as.factor(hold.out.data$Class)
colnames(hold.out.data) <- c('Red', 'Green', 'Blue', 'Class')
```

KNN on Hold Out Data

```
start_time = Sys.time()
knn.pred = knn(haiti.image[-1], hold.out.data[-4], cl = haiti.image[,1], k = which.max(k.value.acc))
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 10.87045 mins
```

```
knn.acc = mean(knn.pred == hold.out.data[,4])*100
knn.acc
```

```
## [1] 99.23061
```

LDA on Hold Out Data

```
start_time = Sys.time()
lda.fit = lda(Class ~ ., data = haiti.image)
lda.pred = predict(lda.fit, hold.out.data)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 5.519053 secs
```

```
lda.acc = mean(lda.pred$class == hold.out.data[,4])*100
lda.acc
```

```
## [1] 98.17496
```

QDA on Hold Out Data

```
start_time = Sys.time()
qda.fit = qda(Class ~ ., data = haiti.image)
qda.pred = predict(qda.fit, hold.out.data)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 4.528015 secs
```

```
qda.acc = mean(qda.pred$class == hold.out.data[,4])*100
qda.acc
```

```
## [1] 99.59719
```

Logistic Regression on Hold Out Data

```
start_time = Sys.time()
log.fit = multinom(Class ~ Red + Green + Blue, data = haiti.image, family = binomial)
```

```
## # weights:  5 (4 variable)
## initial  value 43835.320846
## iter  10 value 1210.137462
## iter  20 value 884.763967
## final  value 884.762363
## converged
```

```
log.pred = predict(log.fit, type = 'class', hold.out.data)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 4.092085 secs
```

```
log.acc = mean(log.pred == hold.out.data[,4])*100
log.acc
```

```
## [1] 98.97793
```

Random Forest on Hold Out Data

```
start_time = Sys.time()
rf.fit = randomForest(Class ~ ., data = haiti.image,
```

```
                          mtry = which.min(er.rf), ntree = ntree.rf[which.min(er.rf)], importance = TRU
rf.pred = predict(rf.fit, newdata = hold.out.data)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 46.28119 secs
```

```
rf.acc =  mean(rf.pred == hold.out.data[,4])*100
rf.acc
```

```
## [1] 99.20546
```

SVM:Radial on Hold Out Data

```
start_time = Sys.time()
haiti.image.svm.radial = svm(Class ~ ., data = haiti.image, kernel = 'radial', cost = 14, gamma = 9)
hold.out.svm.radial.pred = predict(haiti.image.svm.radial, newdata = hold.out.data)
end_time = Sys.time()
end_time - start_time
```

```
## Time difference of 18.86518 secs
```

```
hold.out.svm.radial.pred.acc = mean(hold.out.svm.radial.pred == hold.out.data$Class)*100
hold.out.svm.radial.pred.acc
```

```
## [1] 98.94765
```