```
#### Lab 1 - Assignment 1 - Spam classification with k-nearest
neighbors

### Libraries
library(kknn)

### Setup
data.spambase = read.csv("spambase.csv", dec=',')
n = dim(data.spambase)[1]
set.seed(12345)
ids.train = sample(1:n, floor(n/2)) # Sample 50%
train = data.spambase[ids.train,] # Select the 50% to be train data
test = data.spambase[-ids.train,] # The Test data is the remaining
data
y.test = test[,49]

### Functions

## Returns estimated spam probability of given train data, test data
and k
knearest = function(data.train, data.pred, k){
  # Setup
  n.train = dim(data.train)[1] # number of observations in training
data
  n.pred = dim(data.pred)[1] # number of observations in prediction
data
  p = dim(data.train)[2] # index of the class in the data(s)
  X = as.matrix(data.train[,-p]) # all data in training data excluding
the class
  Y = as.matrix(data.pred[,-p]) # all data in prediction data
excluding the class

  # Calculate the distances
  X.hat = X/matrix(sqrt(rowSums(X^2)), nrow=n.train, ncol=p-1)
  Y.hat = Y/matrix(sqrt(rowSums(Y^2)), nrow=n.pred, ncol=p-1)
  C = X.hat%*%t(Y.hat)
  D = 1-C # Calculate the distances between training data and pred
data

  # Calculate the probabilities
  prob = numeric(n.pred)
  for(i in 1:n.pred){
    k.nearest.neighbors = order(D[,i])[1:k]
    prob[i] = sum(data.train[k.nearest.neighbors, p])/k
  }
  return(prob)
}

## Returns the predictions of given probabilties and probability p
pred = function(prob, p) {
  fit = ifelse(prob>p, 1, 0)
  return(fit)
}
```

```
## Returns the confusion matrix
confusion_matrix = function(y, yfit) {
  return(table(y, yfit, dnn=c("TRUE", "PRED")))
}

## Returns the missclassifcationrate
mcr = function(cm){
  return(1-sum(diag(cm))/sum(cm))
}

## TPR and FPR values for ROC Curve
ROC = function(y, p, prob){
  m = length(p)
  TPR = numeric(m)
  FPR = numeric(m)
  for(i in 1:m){
    pred1 = pred(prob, p[i])
    cm = confusion_matrix(y, pred1)
    print(cm)
    TPR[i] = cm[2,2]/sum(cm[2,]) # PREDICTED TRUE AND IT WAS TRUE /
ALL ACTUAL TRUE
    FPR[i] = cm[1,2]/sum(cm[1,]) # PREDICTED TRUE BUT IT WAS FALSE /
ALL ACTUAL FALSE
  }
  return(list(FPR=FPR, TPR=TPR))
}

### Implementation

## knearest, k=5
prob.knearest.k5 = knearest(train, test, 5)
pred.knearest.k5 = pred(prob.knearest.k5, 0.5)
cm.knearest.k5 = confusion_matrix(y.test, pred.knearest.k5)
mcr.knearest.k5  = mcr(cm.knearest.k5)

## knearest, k=1
prob.knearest.k1 = knearest(train, test, 1)
pred.knearest.k1 = pred(prob.knearest.k1, 0.5)
cm.knearest.k1 = confusion_matrix(y.test, pred.knearest.k1)
mcr.knearest.k1  = mcr(cm.knearest.k1)

## kknn, k=5
prob.kknn.k5 = kknn(Spam~., train=train, test=test, k=5)$fitted.values
pred.kknn.k5 = pred(prob.kknn.k5, 0.5)
cm.kknn.k5 = confusion_matrix(y.test, pred.kknn.k5)
mcr.kknn.k5  = mcr(cm.kknn.k5)

## TPR and FPR
p.seq = seq(0.05, 0.95, 0.05)
ROC.knearest = ROC(y.test, p.seq, prob.knearest.k5)
ROC.kknn = ROC(y.test, p.seq, prob.kknn.k5)

## ROC Curves
plot(x=ROC.knearest$FPR, y=ROC.knearest$TPR, xlim=c(0,1), ylim=c(0,1),
type='b', col="green"
```

```
     ,xlab="FPR", ylab="TPR", main="ROC Curves")
lines(x=ROC.kknn$FPR, y=ROC.kknn$TPR, type="b", col="blue")
legend("topright", legend=c("knearest, k=5", "kknn, k=5"),
col=c("green", "blue"), lty=1)
# Comment: No clear "winner" both are too similar to determine the
superior classification

## Specificity
specificity.knearest = 1-ROC.knearest$FPR
specificity.kknn = 1-ROC.kknn$FPR

plot(p.seq, specificity.knearest , xlab="p", ylab="Specifity (1-FPR)",
main="Specificity", xlim=c(0,1), ylim=c(0,1), type="l", col="blue")
legend("bottomright", legend=c("knearest, k=5", "kknn, k=5"),
col=c("blue", "green"), lty=1)
lines(p.seq, specificity.kknn, col="green")

## Sensitivity
sensitivity.knearest = ROC.knearest$TPR
sensitivity.kknn = ROC.kknn$TPR

plot(p.seq, sensitivity.knearest, xlab="p", ylab="Sensitivity (TPR)",
main="Sensitivity", xlim=c(0,1), ylim=c(0,1), type="l", col="blue")
legend("topright", legend=c("(blue) knearest, k=5", "(green) kknn,
k=5"), col=c("blue", "green"), lty=1)
lines(p.seq, sensitivity.kknn, col="green")
```