Step 0

```r
library("caret")
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library("kernlab")
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```r
credit_data <- read.csv("credit_data.csv", stringsAsFactors = TRUE)
credit_data$credit_history <- ordered(credit_data$credit_history,
                                      levels = c("critical", "poor", "good","very good","perfect"))
credit_data$employment_duration <-
  ordered(credit_data$employment_duration,
  levels = c("unemployed" , "< 1 year", "1 - 4 years", "4 - 7 years","> 7 years"))
```

Step 1

KNN

```r
set.seed(100)

knn_credit <- train(default ~ ., data=credit_data,
                   method = "knn",
                   preProcess = c("center", "scale"),
                   tuneGrid=expand.grid(k=1:50),
                   trControl = trainControl(method = "cv", number=10))

pred_knn <- predict(knn_credit, newdata = credit_data)
confusionMatrix(data = pred_knn, credit_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  628 249
##        yes   2  21
##
##                Accuracy : 0.7211
##                  95% CI : (0.6906, 0.7502)
##     No Information Rate : 0.7
```

```
##     P-Value [Acc > NIR] : 0.08854
##
##                   Kappa : 0.101
##
##  Mcnemar's Test P-Value : < 2e-16
##
##             Sensitivity : 0.99683
##             Specificity : 0.07778
##          Pos Pred Value : 0.71608
##          Neg Pred Value : 0.91304
##              Prevalence : 0.70000
##          Detection Rate : 0.69778
##    Detection Prevalence : 0.97444
##       Balanced Accuracy : 0.53730
##
##        'Positive' Class : no
##
```

SVM linear

```
set.seed(23)

svm_credit = train(default ~ ., data=credit_data,
               method="svmLinear",
               tuneGrid=expand.grid(C = c(0.01,0.1,1,10)),
               preProcess=c("center","scale"),
               trControl= trainControl(method = "cv", number=10, classProbs = TRUE))

pred_svm <-predict(svm_credit, newdata = credit_data)
confusionMatrix(data = pred_svm, credit_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  599 219
##        yes  31  51
##
##                Accuracy : 0.7222
##                  95% CI : (0.6917, 0.7513)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.07728
##
##                   Kappa : 0.1744
##
##  Mcnemar's Test P-Value : < 2e-16
##
##             Sensitivity : 0.9508
##             Specificity : 0.1889
##          Pos Pred Value : 0.7323
##          Neg Pred Value : 0.6220
##              Prevalence : 0.7000
##          Detection Rate : 0.6656
##    Detection Prevalence : 0.9089
##       Balanced Accuracy : 0.5698
```

```
##
##        'Positive' Class : no
##
```

Decision tree

```
set.seed(100)

C5.0_credit <- train(default ~ ., data=credit_data,
                     method = "C5.0",
                     preProcess = c("center", "scale"),
                     tuneGrid = expand.grid( .winnow = c(TRUE,FALSE), .trials=1, .model="tree"),
                     trControl = trainControl(method = "cv", number=10))

pred_tree <- predict(C5.0_credit, newdata = credit_data)
confusionMatrix(data = pred_tree, credit_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  613 103
##        yes  17 167
##
##                Accuracy : 0.8667
##                  95% CI : (0.8427, 0.8882)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6508
##
##  Mcnemar's Test P-Value : 8.533e-15
##
##             Sensitivity : 0.9730
##             Specificity : 0.6185
##          Pos Pred Value : 0.8561
##          Neg Pred Value : 0.9076
##              Prevalence : 0.7000
##          Detection Rate : 0.6811
##    Detection Prevalence : 0.7956
##       Balanced Accuracy : 0.7958
##
##        'Positive' Class : no
##
```

Step 2

KNN

```
pred_probs_knn <-predict(knn_credit, credit_data, type = "prob")
knnROC <- roc(credit_data$default,
              pred_probs_knn[,"yes"],
              levels = c('no','yes'),
              direction ='<')
```

SVM linear

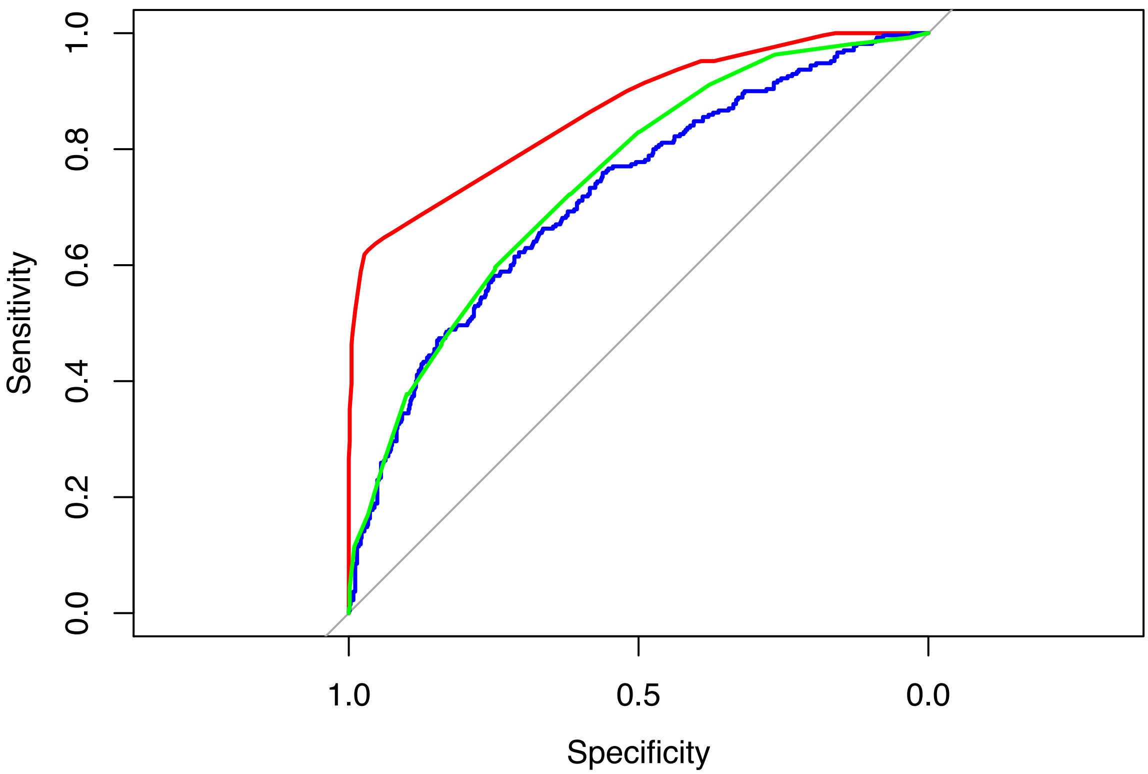
```
pred_probs_svm <-predict(svm_credit, credit_data, type = "prob")
svmLinearROC = roc(predictor=pred_probs_svm$yes,
                   response=credit_data$default,
                   levels=levels(credit_data$default),
                   direction = "<")
```

Decision tree

```
pred_probs_tree <- predict(C5.0_credit, credit_data, type = "prob")
treeROC <- roc(credit_data$default, pred_probs_tree[, "yes"],
                levels = c('no','yes'),
                direction = '<')
```

ROC plots

```
plot(treeROC, col="red")
plot(svmLinearROC, add=TRUE, col="blue")
plot(knnROC, add=TRUE, col="green")
```




Step 3

KNN auc

```
auc(knnROC)
```

```
## Area under the curve: 0.7454
```

SVM linear auc

```
auc(svmLinearROC)
```

```
## Area under the curve: 0.7204
```

Decision tree auc

```r
auc(treeROC)
```

```
## Area under the curve: 0.865
```

We would choose the decision tree algorithm because it has the highest AUC (.865)

Step 4

```r
new_customers_data <- read.csv("new_customers.csv", stringsAsFactors = TRUE)
new_customers_data$credit_history <- ordered(new_customers_data$credit_history,
                                     levels = c("critical", "poor", "good","very good","perfect"))
new_customers_data$employment_duration <-
  ordered(new_customers_data$employment_duration,
  levels = c("unemployed" , "< 1 year", "1 - 4 years", "4 - 7 years","> 7 years"))
```

KNN

```r
pred_knn_new_customers <-predict(knn_credit, newdata = new_customers_data)
pred_knn_new_customers
```

```
## [1] yes no  no  no  no
## Levels: no yes
```

SVM linear

```r
pred_svm_new_customers <-predict(svm_credit, newdata = new_customers_data)
pred_svm_new_customers
```

```
## [1] yes no  yes yes no
## Levels: no yes
```

Decision tree

```r
pred_decision_tree_new_customers <-predict(C5.0_credit, newdata = new_customers_data)
pred_decision_tree_new_customers
```

```
## [1] yes yes yes no  no
## Levels: no yes
```

According to majority rule, the following prediction would be made for the five customers in the new customers dataset: yes no yes no no (in order of customers listed in the new customers dataset)

Step 5

Calculating individual accuracies via confusion matrix

KNN

```r
confusionMatrix(data = pred_knn_new_customers, new_customers_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no   3   1
##        yes  0   1
##
##                Accuracy : 0.8
##                  95% CI : (0.2836, 0.9949)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.337
```

```
##
##                   Kappa : 0.5455
##
##  Mcnemar's Test P-Value : 1.000
##
##             Sensitivity : 1.00
##             Specificity : 0.50
##          Pos Pred Value : 0.75
##          Neg Pred Value : 1.00
##              Prevalence : 0.60
##          Detection Rate : 0.60
##    Detection Prevalence : 0.80
##       Balanced Accuracy : 0.75
##
##        'Positive' Class : no
##
```

SVM linear

```
confusionMatrix(data = pred_svm_new_customers, new_customers_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no   2   0
##        yes  1   2
##
##                Accuracy : 0.8
##                  95% CI : (0.2836, 0.9949)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.337
##
##                   Kappa : 0.6154
##
##  Mcnemar's Test P-Value : 1.000
##
##             Sensitivity : 0.6667
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.6667
##              Prevalence : 0.6000
##          Detection Rate : 0.4000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.8333
##
##        'Positive' Class : no
##
```

Decision Tree

```
confusionMatrix(data = pred_decision_tree_new_customers, new_customers_data$default)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction no yes
##        no   2   0
##        yes  1   2
##
##                Accuracy : 0.8
##                  95% CI : (0.2836, 0.9949)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.337
##
##                   Kappa : 0.6154
##
##  Mcnemar's Test P-Value : 1.000
##
##             Sensitivity : 0.6667
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.6667
##              Prevalence : 0.6000
##          Detection Rate : 0.4000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.8333
##
##        'Positive' Class : no
##
```

Calculating accuracy for majority rule manually

Majority value predictions in order listed (predictions in brackets are correct): [yes] [no] [yes] [no] [no] Majority rule accuracy = 5/5 = 100% The majority rule accuracy of 100% is greater than the individual accuracy of the algorithms which are all equal to 80%

# DMBA HW 4 Part B

## Iteration 1

| point x | point y | distance to centroid 1: (1,1) | distance to centroid 2: (8,8) |
|---|---|---|---|
| 8 | 4 | $\|8-1\|+\|4-1\| = 7+3 = 10$ | $\|8-8\|+\|4-8\| = 0+4 = 4$ |
| 3 | 3 | $\|3-1\|+\|3-1\| = 2+2 = 4$ | $\|3-8\|+\|3-8\| = 5+5 = 10$ |
| 4 | 5 | $\|4-1\|+\|5-1\| = 3+4 = 7$ | $\|4-8\|+\|5-8\| = 4+3 = 7$ |
| 0 | 1 | $1+0 = 1$ | $8+7 = 15$ |
| 10 | 2 | $9+1 = 10$ | $2+6 = 8$ |
| 3 | 7 | $2+6 = 8$ | $5+1 = 6$ |
| 0 | 9 | $1+8 = 9$ | $8+1 = 9$ |
| 8 | 1 | $7+0 = 7$ | $0+7 = 7$ |
| 4 | 3 | $3+2 = 5$ | $4+5 = 9$ |
| 9 | 4 | $8+3 = 11$ | $1+4 = 5$ |

centroid 1: (1,1)

(3,3)
(4,5)
(0,1)
(0,9)
(8,1)
(4,3)

centroid 2: (8,8)

(8,4)
(10,2)
(3,7)
(9,4)

## New centroids:

centroid 1

$$\frac{3+4+0+0+8+4}{6} = \frac{7+12}{6} = \frac{19}{6}$$

$$\frac{3+5+1+9+1+3}{6} = \frac{8+10+4}{6} = \frac{22}{6}$$

$\Rightarrow \left(\frac{19}{6}, \frac{22}{6}\right)$

centroid 2

$$\frac{8+10+3+9}{4} = \frac{18+12}{4} = \frac{30}{4} = 7.5$$

$$\frac{4+2+7+4}{4} = \frac{6+11}{4} = \frac{17}{4} = 4.25$$

$\Rightarrow (7.5, 4.25)$

# Iteration 2

| point x | point y | distance to centroid 1: $(\frac{19}{6}, \frac{22}{6})$ | distance to centroid 2: (7.5, 4.25) |
| --- | --- | --- | --- |
| 8 | 4 | $\lvert 8-\frac{19}{6}\rvert + \lvert 4-\frac{22}{6}\rvert = \frac{31}{6} = 5.17$ | $\lvert 8-7.5\rvert + \lvert 4-4.25\rvert = .75$ |
| 3 | 3 | $\lvert 3-\frac{19}{6}\rvert + \lvert 3-\frac{22}{6}\rvert = \frac{5}{6}$ | $\lvert 3-7.5\rvert + \lvert 3-4.25\rvert = 5.75$ |
| 4 | 5 | $\lvert 4-\frac{19}{6}\rvert + \lvert 5-\frac{22}{6}\rvert = \frac{13}{6} = 2.17$ | $\lvert 4-7.5\rvert + \lvert 5-4.25\rvert = 4.25$ |
| 0 | 1 | $\frac{35}{6} = 5.83$ | 10.75 |
| 10 | 2 | $\frac{17}{2} = 8.5$ | 4.75 |
| 3 | 7 | $\frac{7}{2} = 3.5$ | 7.25 |
| 0 | 9 | $\frac{17}{2} = 8.5$ | 12.25 |
| 8 | 1 | $\frac{15}{2} = 7.5$ | 3.75 |
| 4 | 3 | $\frac{3}{2} = 1.5$ | 4.75 |
| 9 | 4 | $\frac{37}{6} = 6.17$ | 1.75 |

centroid 1:

(3,3)
(4,5)
(0,1)
(3,7)
(0,9)
(4,3)

centroid 2:

(8,4)
(10,2)
(8,1)
(9,4)

## New centroids:

centroid 1

$$\frac{3+4+0+3+0+4}{6} = \frac{7+7}{6} = \frac{14}{6}$$

$$\frac{3+5+1+7+9+3}{6} = \frac{8+8+12}{6} = \frac{28}{6}$$

$\Rightarrow (\frac{14}{6}, \frac{28}{6})$

centroid 2

$$\frac{8+10+8+9}{4} = \frac{18+17}{4} = \frac{35}{4} = 8.75$$

$$\frac{4+2+1+4}{4} = \frac{6+5}{4} = \frac{11}{4} = 2.75$$

$\Rightarrow (8.75, 2.75)$

```
data<-read.csv("/Users/joonghochoi/Desktop/prospects.csv")
df<-as.data.frame(data)
df<-na.omit(df)

df$GENDER<-ifelse(df$GENDER=="M",1,0)
df <- df[,-c(1,8)]

## --- standardize all the variables
library(caret)
# Estimate Pre-processing transformation (centering, scaling etc.)
#    Estimated from the training data and applied to any data set with the same variables.
preproc = preProcess(df) # default: method = c("center", "scale")
df.norm = predict(preproc, df)

set.seed(100)

km <- kmeans(df.norm, centers = 4, iter.max=1000)

Q1)
km$size
table(km$cluster) #number of points in which cluster

Output:
  1    2    3    4
935 1142 1485 1359

Q2)
km$centers

#For the 1st cluster, its center's gender is -1.0700134 and married is -1.1267097.
#This means the center is likely female and not married

#For the 2nd cluster, its center's gender is 0.9343779 and married is -1.1418361.
#This means the center is likely male and not married.

#For the 3rd cluster, its center's gender is 0.9343779 and married is 0.8227884.
#This means the center is likely male and married.

#For the 4th cluster, its center's gender is -1.0700134 and married is 0.8356215.
#This means the center is likely female and married.
```