Laboratory 3: Mixed Methods or Semi-Supervised Learning

This lab is intended to help you understand how to integrate unsupervised and supervised learning techniques. You'll start by considering unsupervised learning using k-Means clustering. Then, you'll use the results of that analysis as input for a supervised learning analysis, specifically a classification analysis. This is the basis for a mixed methods or semi-supervised learning analysis. To complete this lab you need to submit a report of your analysis as described in Parts I, II, III and IV below.

Part I

This part is relatively easy. It is intended to give you some practice running a k-Means analysis. Use the Wholesale customers dataset from UCI at: https://archive.ics.uci.edu/ml/datasets/Wholesale+customers. This is a .csv file.

The R script to use in the analysis is in Part IV

Now let's get back to our problem of news popularity and see if we can apply KNN (K- nearest neighbors) to improve the accuracy of the model. Use the same strategy of training and testing that we did on first 2 labs, and don't forget that whenever it is required you should use: *set.seed(12345).*

For designing the model, use following command:

knn(train = <training features>, test = <testing features>, cl=<training labels>, k = custom value)

It appears that everything is straightforward. The only unknown value is k and it is your job to find the best value of k. In your report you should build this model, test it on your test data and write about the model evaluation and why you think the k that yoou have chosen is the best value.

Appendix A: R script for Wholesale customers dataset and uploaded to our Moodle site. When you use this script it will generate several tables and plots. You will need to consider the within (cluster) sum of squares and between (cluster) sum of squares to determine the value for "k," i.e. the number of clusters that provides the best result for this analysis.

Given this is an imperfect real-world, you need to determine what you believe is the best value for "k" and write-up this portion of your lab report.

You should include a brief discussion of your k-Means analysis as well as the best value of "k" that you determine. You should include what mixture of variables within the clusters that this value of "k" results in. That is, you need to interpret your k-Means analysis and discuss what it means.

Part II

Let's look at a better behaved dataset, i.e. the wine dataset from UCI at https://archive.ics.uci.edu/ml/datasets/wine. As before, I have provided you with the R script to do a k-Means analysis on this dataset. The entire script for both Part II and Part III of this lab is in

Appendix B: R script for Wine dataset.

Essentially, the script reads in the data. There is a short function, *wssplot()*, that will plot up the within (cluster) sum of squares versus different values of "k". The script will use this to generate a table and a barplot for your use in determining what the best number of clusters is for this dataset. You will be asked to enter the value for "k" you want to use. The script produces a variety of plots, a confusion or truth table, as well as a plot of the resulting clusters.

You will need to write-up this portion of your lab report to include a brief description of the k-Means analysis conducted and the results of the k-Means analysis conducted. Make sure you understand the clusters you find from this k-Means analysis. You will use them in the next part of the lab as the "labels" for the observations in the dataset.

Part III

For this part of Laboratory #3 we want to train a classifier to classify wines using the clusters we obtained in Part II above. The first thing we'll do is label the data frame we have been using with the cluster labels. That is, use the command:

```
> df <- data.frame(k=fit.km$cluster, df)
```

to label the observations or records in the wine dataset with the cluster labels we just found. You can view this labeled dataset using:

```
> View(df)
```

To complete this part of the lab you can use any of the classification techniques that were covered in Laboratory #1. In this documentation, for illustration of classification, we'll use the classification technique covered during our lecture on classification. That is, we'll use the classification in the *rpart* package in R. If you haven't installed all the packages we used during lecture now is a good time to do that. You'll need:

- rpart
- rpart.plot
- RColorBrewer
- rattle

Of these packages the only one that should give you any trouble is *rattle*.

Since the observations in the dataset are essentially sorted by class the first thing you'll need to do is to randomize the dataset, otherwise you'll wind up training the model on a subset of the classes with one class omitted. You can use the following command to do this:

```
> rdf <- df[sample(1:nrow(df)), ]
```

Note that this will randomize the entire dataset through the total number of observations using nrow(df). The next thing is to split the dataset into training and test sets. I've used an 80/20 split. You can change that to any appropriate split. The commands include the calculation of the number of observations as follows:

```
> train <- rdf[1:(as.integer(0.8*nrow(rdf))-1), ]
> test <- rdf[(as.integer(0.8*nrow(rdf))):nrow(rdf), ]
```

If you are going to change the split just change the value 0.8. Fitting the classification model is the same as we covered during our lecture. In this case I've only included a plot using the *fancyRpartPlot()* function.

The biggest difference in the script for this part of Laboratory #3 is that I've included a confusion or truth table to evaluate the performance of the model. That is, I've included the commands:

```
> pred <- predict(fit, test, type="class")
> table(pred, test$k)
```

This is of particular interest because, if you compare the original wine dataset from UCI with the classes assigned based on the k-Means clustering results, you'll see that there are some "misclassifications" due to the cluster assignments. In your written report for this part of the lab you need to include some discussion of what you think the impact of this misclassification is. For example, do misclassifications due to the cluster assignments flow through the entire analysis. Can the extent any error in the model due to these misclassifications be estimated? By that I mean can you estimate the number or percentage of future misclassifications made by the model based on the error that will propagate through the model from the cluster analysis?

Therefore, in your written report include a brief discussion of the analysis you conducted for Part III of this lab. Be sure to include a discussion of your evaluation of the model as you discuss the results of this analysis. And, include a discussion of any known errors and their impact on future predictions using this classification model.

Part IV

Now let's get back to our problem of news popularity and see if we can apply KNN (K- nearest neighbors) to improve the accuracy of the model. Use the same strategy of training and testing that we did on first 2 labs, and don't forget that whenever it is required you should use: *set.seed(12345).*

For designing the model, use following command:

knn(train = <training features>, test = <testing features>, cl=<training labels>, k = custom value)

It appears that everything is straightforward. The only unknown value is k and it is your job to find the best value of k. In your report you should build this model, test it on your test data and write about the model evaluation and why you think the k that yoou have chosen is the best value.

Appendix A: R script for Wholesale customers dataset

```
#Requires some data frame and the top N to remove
#Convert to data frame
data <- data.frame(Wholesale_customers_data)

top.n.custs <- function (data,cols,n=5) {

  #Initialize a vector to hold customers being removed
  idx.to.remove <-integer(0)

  for (c in cols){

    # For every column in the data we passed to this function
    #Sort column "c" in descending order (bigger on top)
    #Order returns the sorted index (e.g. row 15, 3, 7, 1, ...) rather than the actual values sorted.

    col.order <-order(data[,c],decreasing=T)


    #Take the first n of the sorted column C to
    #combine and de-duplicate the row ids that need to be removed

    idx <-head(col.order, n)
    idx.to.remove <-union(idx.to.remove,idx)

  }

  #Return the indexes of customers to be removed
  return(idx.to.remove)

}

#How Many Customers to be Removed?
top.custs <-top.n.custs(data, cols = 1:5,n=5)
length(top.custs)

#Examine the customers
data[top.custs,]

#Remove the Customers
data.rm.top<-data[-c(top.custs),]

#Examine summary stats for the remaining data
print(summary(data.rm.top))

#Set the seed for reproducibility
```

```
set.seed(76964057)

#Try K from 2 to 20
rng<-2:20

#Number of times to run the K Means algorithm
tries <-100

#Set up an empty vector to hold all of points
avg.totw.ss <-integer(length(rng))
avg.totb.ss <- integer(length(rng))
avg.tot.ss <- integer(length(rng))

# For each value of the range variable
for(v in rng){

  #Set up an empty vectors to hold the tries
  v.totw.ss <-integer(tries)
  b.totb.ss <- integer(tries)
  tot.ss <- integer(tries)

  #Run kmeans
  for(i in 1:tries){
    k.temp <-kmeans(data.rm.top,centers=v)

    #Store the total withinss
    v.totw.ss[i] <-k.temp$tot.withinss

    #Store the betweenss
    b.totb.ss[i] <- k.temp$betweenss

    #Store the total sum of squares
    tot.ss[i] <- k.temp$totss
  }

  #Average the withinss and betweenss
  avg.totw.ss[v-1] <-mean(v.totw.ss)
  avg.totb.ss[v-1] <-mean(b.totb.ss)
  avg.tot.ss[v-1] <- mean(tot.ss)
}

plot(rng,avg.totw.ss,type="b", main="Total Within SS by Various K",
     ylab="Average Total Within Sum of Squares",
     xlab="Value of K")

plot(rng,avg.totb.ss,type="b", main="Total between SS by Various K",
```

```
    ylab="Average Total Between Sum of Squares",
    xlab="Value of K")

#Plot the ratio of betweenss/total ss and withinss / total ss for evaluation
plot(rng,avg.totb.ss/avg.tot.ss,type="b", main="Ratio of between ss / the total ss by Various K",
    ylab="Ratio Between SS / Total SS",
    xlab="Value of K")
abline(h=0.85, col="red")

plot(rng,avg.totw.ss/avg.tot.ss,type="b", main="Ratio of within ss / the total ss by Various K",
    ylab="Ratio Between SS / Total SS",
    xlab="Value of K")
abline(h=0.15, col="red")


#Create the best number of clusters, Remove columns 1 and 2
n <- readline(prompt = "Enter the best number of clusters: ")
return(as.integer(n))

k <-kmeans(data.rm.top[,-c(1,2)], centers=n)

#Display cluster centers
print(k$centers)

#Give a count of data points in each cluster
print(table(k$cluster))


#Generate a plot of the clusters
library(cluster)
clusplot(data.rm.top, k$cluster, main='2D representation of the Cluster solution',
        color=TRUE, shade=TRUE,
        labels=2, lines=0)
```

Appendix B: R script for Wine dataset

```
#This R script will perform a k-Means analysis on the wine dataset from UCI.  This dataset
#has 13 chemical measurements on 178 observations of Italian wine.

#Plot the within (cluster) sum of squares to determine the initial value for "k"

wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")}


#Load data into R/RStudio and view it

wine <- read.csv("wine.csv")
df <- scale(wine[-1])


#Examine the data frame and plot the within sum of squares

head(df)
wssplot(df)


#Start the k-Means analysis using the variable "nc" for the number of clusters

library(NbClust)
set.seed(1234)
nc <- NbClust(df, min.nc=2, max.nc = 15, method = "kmeans")

print(table(nc$Best.n[1,]))

barplot(table(nc$Best.n[1,]), xlab = "Number of Clusters", ylab = "Number of Criteria", main =
"Number of Clusters Chosen by 26 Criteria")


#Enter the best number of clusters based on the information in the table and barplot

n <- readline(prompt = "Enter the best number of clusters:  ")
n <- as.integer(n)
```

```
#Conduct the k-Means analysis using the best number of clusters

set.seed(1234)
fit.km <- kmeans(df, n, nstart=25)

print(fit.km$size)

print(fit.km$centers)

print(aggregate(wine[-1], by=list(cluster=fit.km$cluster), mean))


#Use a confusion or truth table to evaluate how well the k-Means analysis performed

ct.km <- table(wine$Type, fit.km$cluster)
print(ct.km)


#Generate a plot of the clusters

library(cluster)
clusplot(df, fit.km$cluster, main='2D representation of the Cluster solution',
         color=TRUE, shade=TRUE,
         labels=2, lines=0)


#Set-up to train a model for classification of wines

library(rpart)

df <- data.frame(k=fit.km$cluster, df)
print(str(df))

#Randomize the dataset
rdf <- df[sample(1:nrow(df)), ]
print(head(rdf))

train <- rdf[1:(as.integer(.8*nrow(rdf))-1), ]
test <- rdf[(as.integer(.8*nrow(rdf))):nrow(rdf), ]

#Train the classifier and plot the results
fit <- rpart(k ~ ., data=train, method="class")

library(rpart.plot)
library(RColorBrewer)
```

```
library(rattle)

fancyRpartPlot(fit)

#Now use the predict() function to see how well the model works
predict(fit, test, type="class")

print(table(pred, test$k))
```