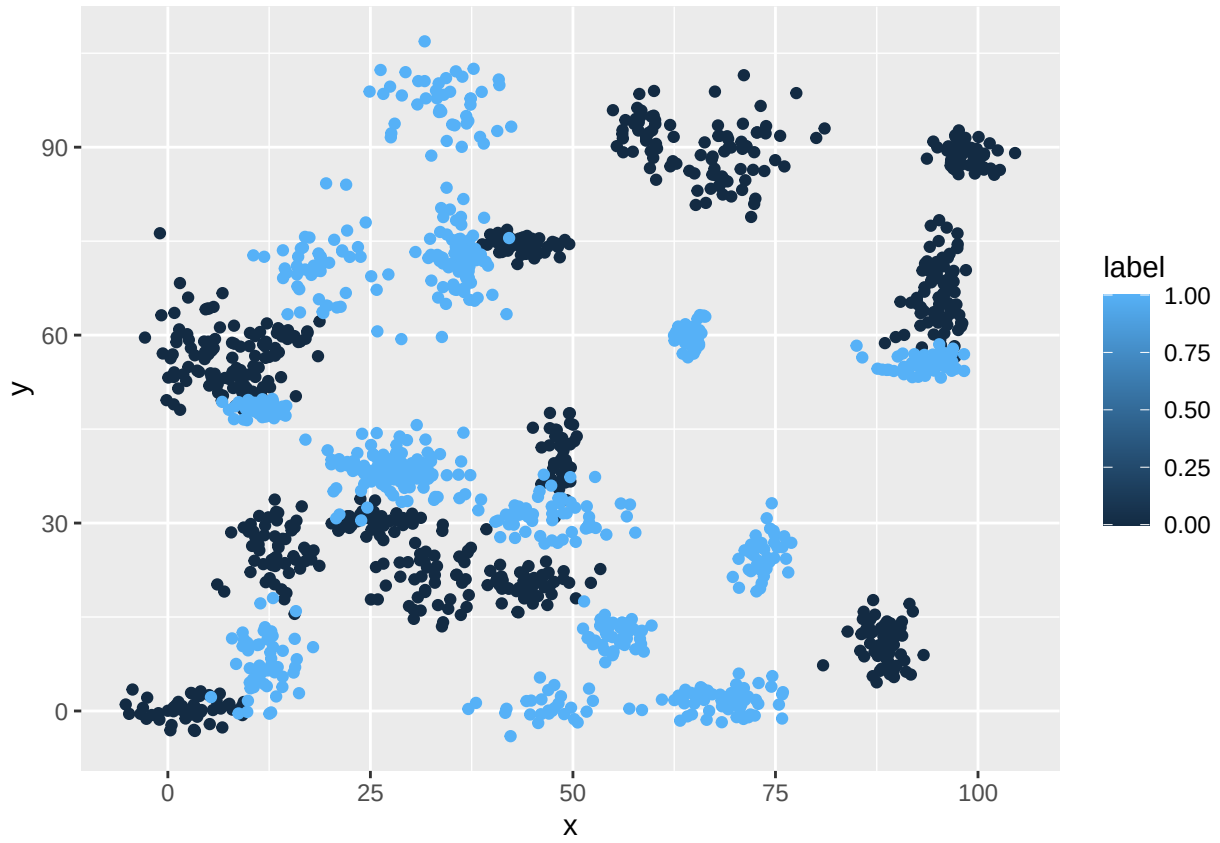
Exercise 10.2 Part 1

Alan Donahue

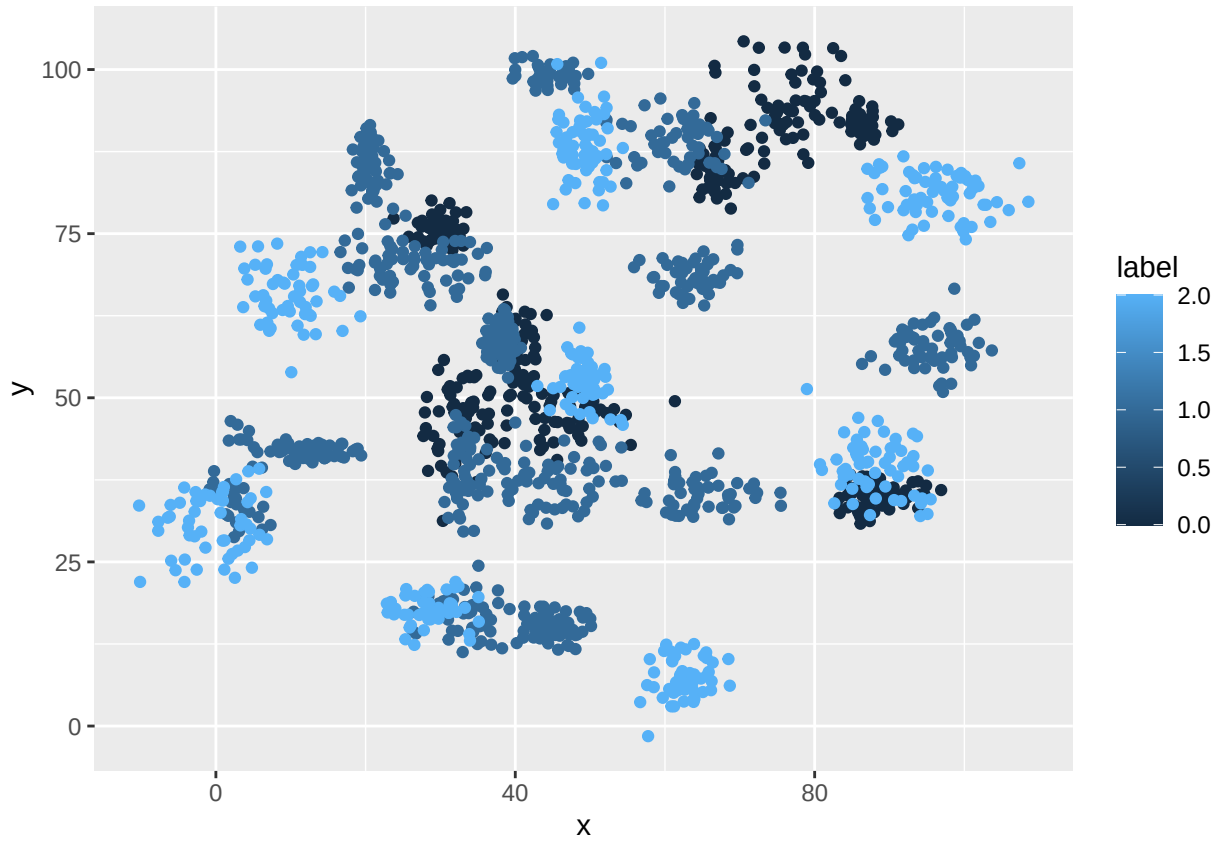
8/14/2021

Section 1

```
library(ggplot2)
library(foreign)
library(caTools)
library(class)
library(plyr)
library(useful)
library(cluster)

#setting the working directory
setwd("C:/Users/Alan Donahue/Documents/data science masters/DSC 520 Stats/GIT/dsc520")

#get the data into a dataframe
binary_df = read.csv("data/binary-classifier-data.csv")
trinary_df = read.csv("data/trinary-classifier-data.csv")

#plot scatter plots of each data
ggplot(binary_df, aes(x=x, y=y, color=label)) + geom_point()
```




```
ggplot(trinary_df, aes(x=x, y=y, color=label)) + geom_point()
```




```
#generating random number
bi.ran <- sample(1:nrow(binary_df), 0.9 * nrow(binary_df))
tri.ran <- sample(1:nrow(trinary_df), 0.9 * nrow(trinary_df))

#normalize function
nor <- function(x) { (x - min(x))/(max(x)-min(x)) }

#normalize the data
bi_norm <- as.data.frame(lapply(binary_df[,c(2,3)], nor))
tri_norm <- as.data.frame(lapply(trinary_df[,c(2,3)], nor))

summary(bi_norm)
```

```
##        x                y         
##  Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.2275   1st Qu.:0.2274  
##  Median :0.4278   Median :0.4386  
##  Mean   :0.4580   Mean   :0.4421  
##  3rd Qu.:0.6522   3rd Qu.:0.6556  
##  Max.   :1.0000   Max.   :1.0000
```

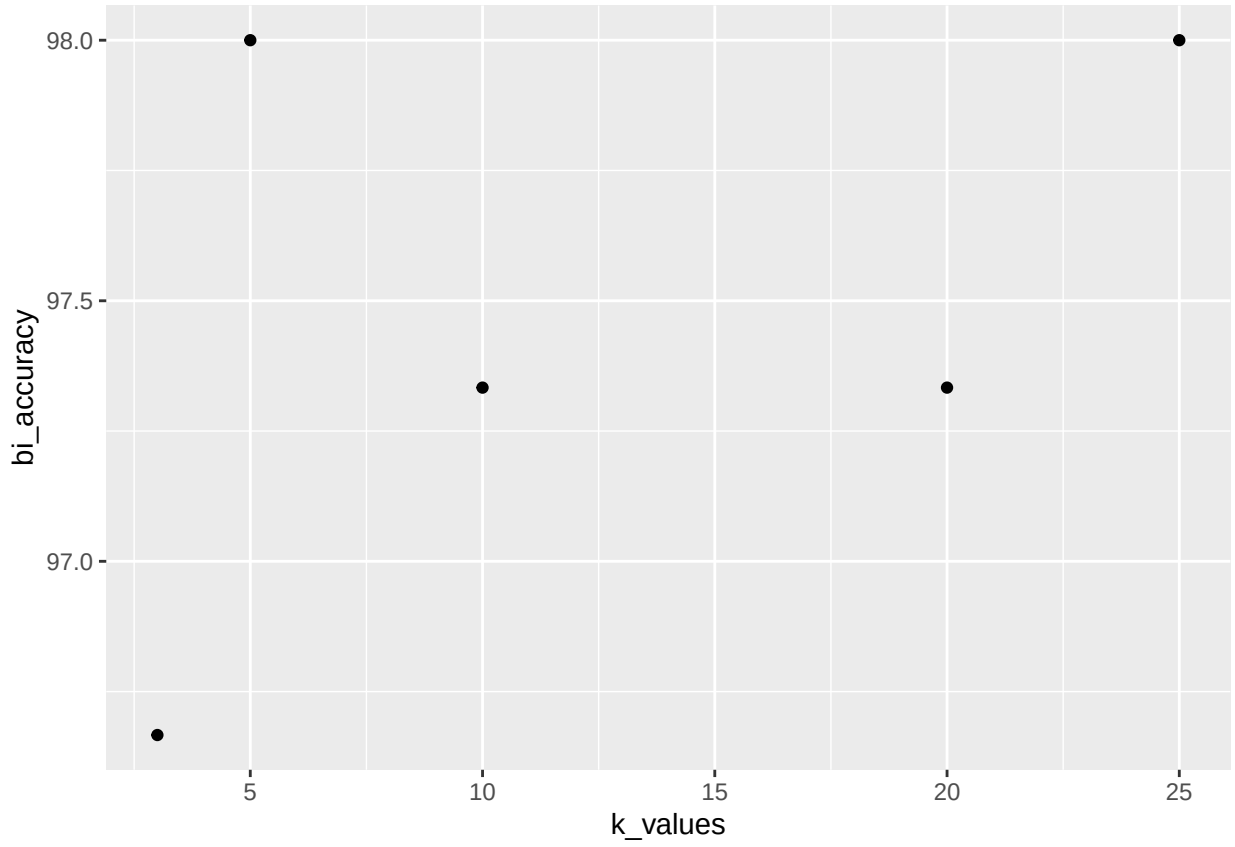
```
summary(tri_norm)
```

```
##        x                y         
##  Min.   :0.0000   Min.   :0.0000
```

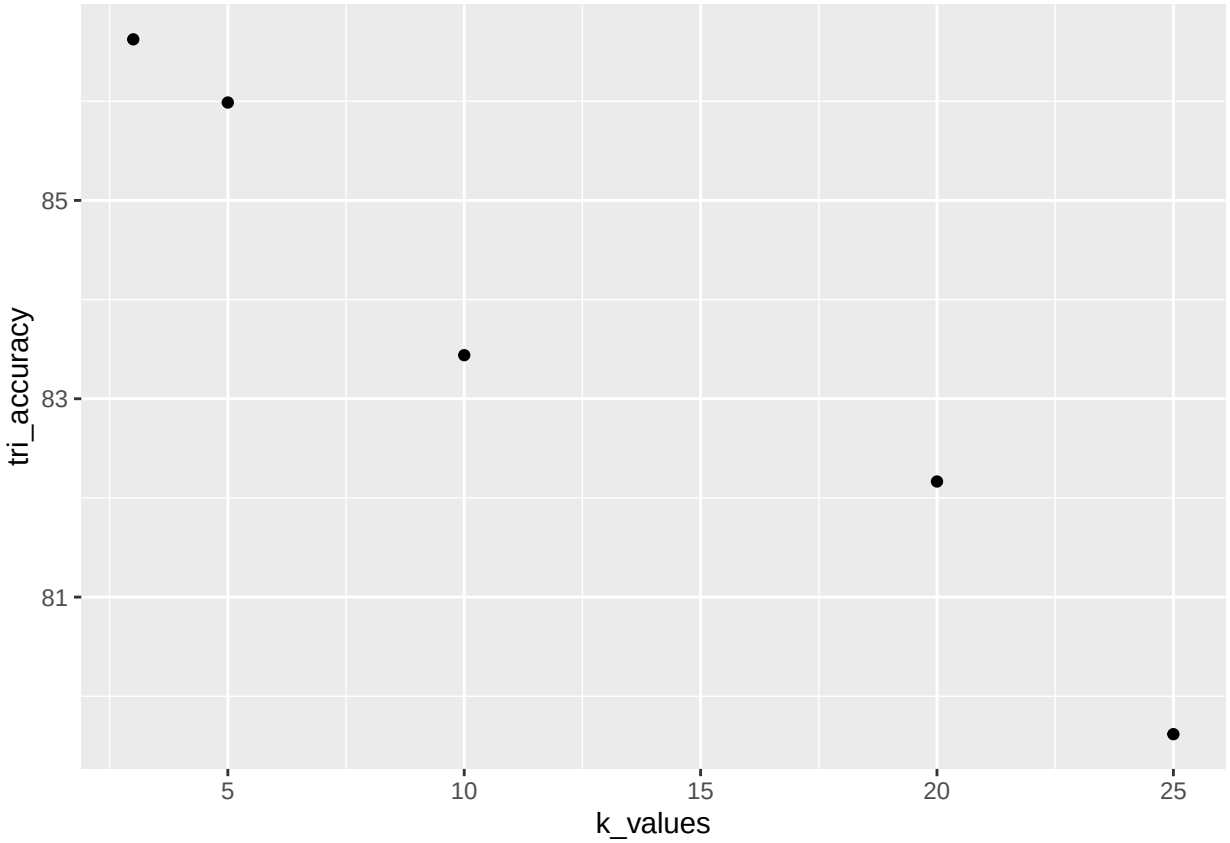
```
##  1st Qu.:0.3485   1st Qu.:0.3538
##  Median :0.4701   Median :0.5349
##  Mean   :0.4976   Mean   :0.5369
##  3rd Qu.:0.6441   3rd Qu.:0.7459
##  Max.   :1.0000   Max.   :1.0000
```

```r
#extracting the training set
bi_train <- bi_norm[bi.ran,]
tri_train <- tri_norm[tri.ran,]

#extracting the testing set
bi_test <- bi_norm[-bi.ran,]
tri_test <- tri_norm[-tri.ran,]

#extract label from train set to be used as 'cl'
bi_target_category <- binary_df[bi.ran,1]
tri_target_category <- trinary_df[tri.ran,1]

#extract label from test set to be used as 'cl'
bi_test_category <- binary_df[-bi.ran,1]
tri_test_category <- trinary_df[-tri.ran,1]

#run knn function for k=3,5,10,20,25
bi_3 <- knn(bi_train,bi_test,cl=bi_target_category,k=3)
bi_5 <- knn(bi_train,bi_test,cl=bi_target_category,k=5)
bi_10 <- knn(bi_train,bi_test,cl=bi_target_category,k=10)
bi_20 <- knn(bi_train,bi_test,cl=bi_target_category,k=20)
bi_25 <- knn(bi_train,bi_test,cl=bi_target_category,k=25)

tri_3 <- knn(tri_train,tri_test,cl=tri_target_category,k=3)
tri_5 <- knn(tri_train,tri_test,cl=tri_target_category,k=5)
tri_10 <- knn(tri_train,tri_test,cl=tri_target_category,k=10)
tri_20 <- knn(tri_train,tri_test,cl=tri_target_category,k=20)
tri_25 <- knn(tri_train,tri_test,cl=tri_target_category,k=25)

#create confusion matrix
bi_tab3 <- table(bi_3,bi_test_category)
bi_tab5 <- table(bi_5,bi_test_category)
bi_tab10 <- table(bi_10,bi_test_category)
bi_tab20 <- table(bi_20,bi_test_category)
bi_tab25 <- table(bi_25,bi_test_category)

tri_tab3 <- table(tri_3,tri_test_category)
tri_tab5 <- table(tri_5,tri_test_category)
tri_tab10 <- table(tri_10,tri_test_category)
tri_tab20 <- table(tri_20,tri_test_category)
tri_tab25 <- table(tri_25,tri_test_category)

#determine accuracy
accuracy <- function(x) {sum(diag(x)/(sum(rowSums(x)))) * 100}
acc_bi3 <- accuracy(bi_tab3)
acc_bi5 <- accuracy(bi_tab5)
acc_bi10 <- accuracy(bi_tab10)
acc_bi20 <- accuracy(bi_tab20)
```

```
acc_bi25 <- accuracy(bi_tab25)

acc_tri3 <- accuracy(tri_tab3)
acc_tri5 <- accuracy(tri_tab5)
acc_tri10 <- accuracy(tri_tab10)
acc_tri20 <- accuracy(tri_tab20)
acc_tri25 <- accuracy(tri_tab25)

#plotting the information
k_values <- c(3,5,10,20,25)
bi_accuracy <- c(acc_bi3, acc_bi5, acc_bi10, acc_bi20, acc_bi25)
tri_accuracy <- c(acc_tri3, acc_tri5, acc_tri10, acc_tri20, acc_tri25)

bi_plot <- cbind(k_values, bi_accuracy)
tri_plot <- cbind(k_values, tri_accuracy)

ggplot(as.data.frame(bi_plot), aes(x=k_values, y=bi_accuracy)) + geom_point()
```




```
ggplot(as.data.frame(tri_plot), aes(x=k_values, y=tri_accuracy)) + geom_point()
```



Compared to last week

Using the k-nearest neighbor is incredibly more accurate than using a logistic regression model. Last week's model produced a 58.5% accuracy while this one is no lower than 80%.