Exercise 11

Alissa Trujillo

2022-05-24

Exercise 11

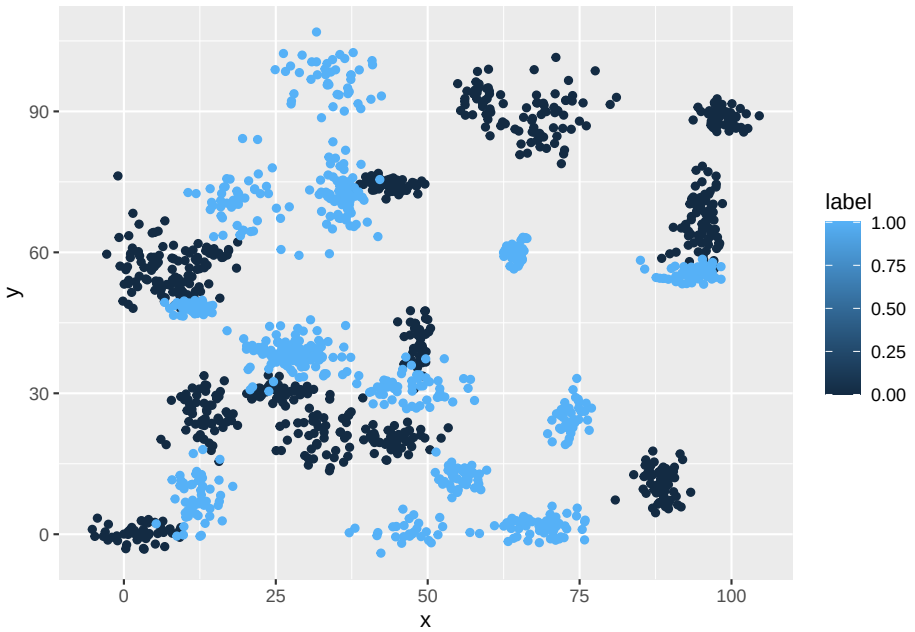
Rmd file: https://github.com/alisssaa/dsc520/blob/master/completed/Exercise11/Exercise11.Rmd

1. Machine Learning

i. Scatter Plots

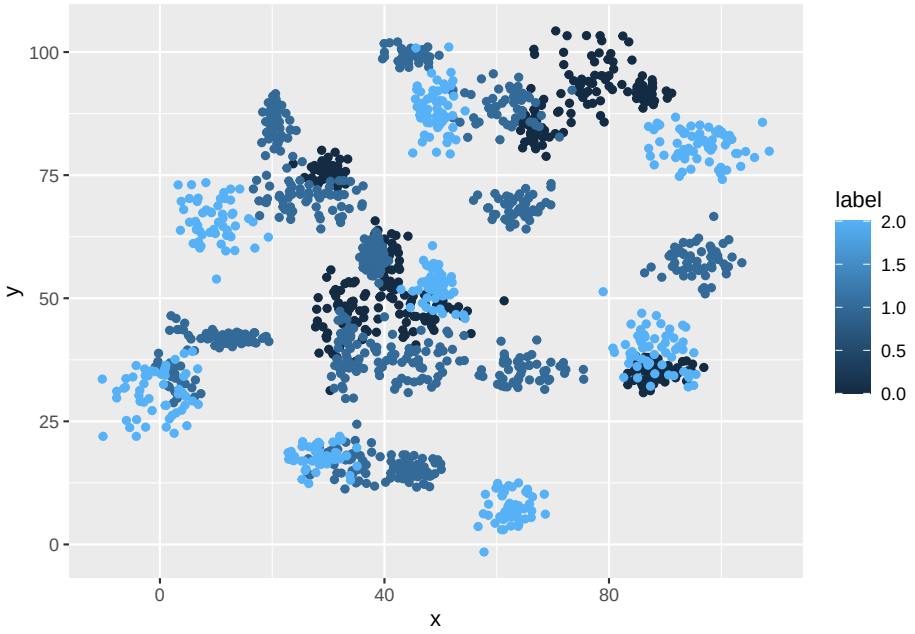
Binary Classifier Plot

```
setwd("/Users/alissa/Documents/Grad/DSC 520/dsc520")
library(ggplot2)
library(caTools)

binary_df <- read.csv("data/binary-classifier-data.csv", header = TRUE)
trinary_df <- read.csv("data/trinary-classifier-data.csv",
    header = TRUE)

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



Trinary Classifier Plot

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



ii. K-Nearest Neighbors

Binary Classifier K Nearest Neigbors

```
binarySplit <- sample.split(binary_df, SplitRatio = 0.75)
binaryTrain <- subset(binary_df, binarySplit == "TRUE")
binaryValidate <- subset(binary_df, binarySplit == "FALSE")

library(class)
binaryK3 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 3)
binaryK5 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 5)
binaryK10 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 10)
binaryK15 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 15)
binaryK20 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 20)
binaryK25 <- knn(train = binaryTrain, test = binaryValidate,
    cl = binaryTrain$label, k = 25)

ACCbK3 <- 100 * sum(binaryValidate$label == binaryK3)/nrow(binaryValidate)
ACCbK5 <- 100 * sum(binaryValidate$label == binaryK5)/nrow(binaryValidate)
ACCbK10 <- 100 * sum(binaryValidate$label == binaryK10)/nrow(binaryValidate)
ACCbK15 <- 100 * sum(binaryValidate$label == binaryK15)/nrow(binaryValidate)
ACCbK20 <- 100 * sum(binaryValidate$label == binaryK20)/nrow(binaryValidate)
```

```
ACCbK25 <- 100 * sum(binaryValidate$label == binaryK25)/nrow(binaryValidate)

ACCbK3
```

```
## [1] 97.39479
```

```
ACCbK5
```

```
## [1] 97.39479
```

```
ACCbK10
```

```
## [1] 96.99399
```

```
ACCbK15
```

```
## [1] 96.59319
```

```
ACCbK20
```

```
## [1] 96.79359
```

```
ACCbK25
```

```
## [1] 96.19238
```

Trinary Classifier K Nearest Neighbors

```
trinarySplit <- sample.split(trinary_df, SplitRatio = 0.75)
trinaryTrain <- subset(trinary_df, trinarySplit == "TRUE")
trinaryValidate <- subset(trinary_df, trinarySplit == "FALSE")

library(class)
trinaryK3 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 3)
trinaryK5 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 5)
trinaryK10 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 10)
trinaryK15 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 15)
trinaryK20 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 20)
trinaryK25 <- knn(train = trinaryTrain, test = trinaryValidate,
    cl = trinaryTrain$label, k = 25)

ACCtK3 <- 100 * sum(trinaryValidate$label == trinaryK3)/nrow(trinaryValidate)
ACCtK5 <- 100 * sum(trinaryValidate$label == trinaryK5)/nrow(trinaryValidate)
ACCtK10 <- 100 * sum(trinaryValidate$label == trinaryK10)/nrow(trinaryValidate)
ACCtK15 <- 100 * sum(trinaryValidate$label == trinaryK15)/nrow(trinaryValidate)
ACCtK20 <- 100 * sum(trinaryValidate$label == trinaryK20)/nrow(trinaryValidate)
ACCtK25 <- 100 * sum(trinaryValidate$label == trinaryK25)/nrow(trinaryValidate)

ACCtK3
```

```
## [1] 92.73423
```

```
ACCtK5
```

```
## [1] 92.73423
```

```
ACCtK10
```

```
## [1] 90.05736
```

```
ACCtK15
```

```
## [1] 87.5717
```

```
ACCtK20
```

```
## [1] 87.5717
```

```
ACCtK25
```

```
## [1] 87.76291
```

iii. Linear Classifier

A linear classifier would not make sense for these data sets. The points are distributed in a way that points with different labels overlap. There is no way to create a line that makes a reliable separation of the different categories.

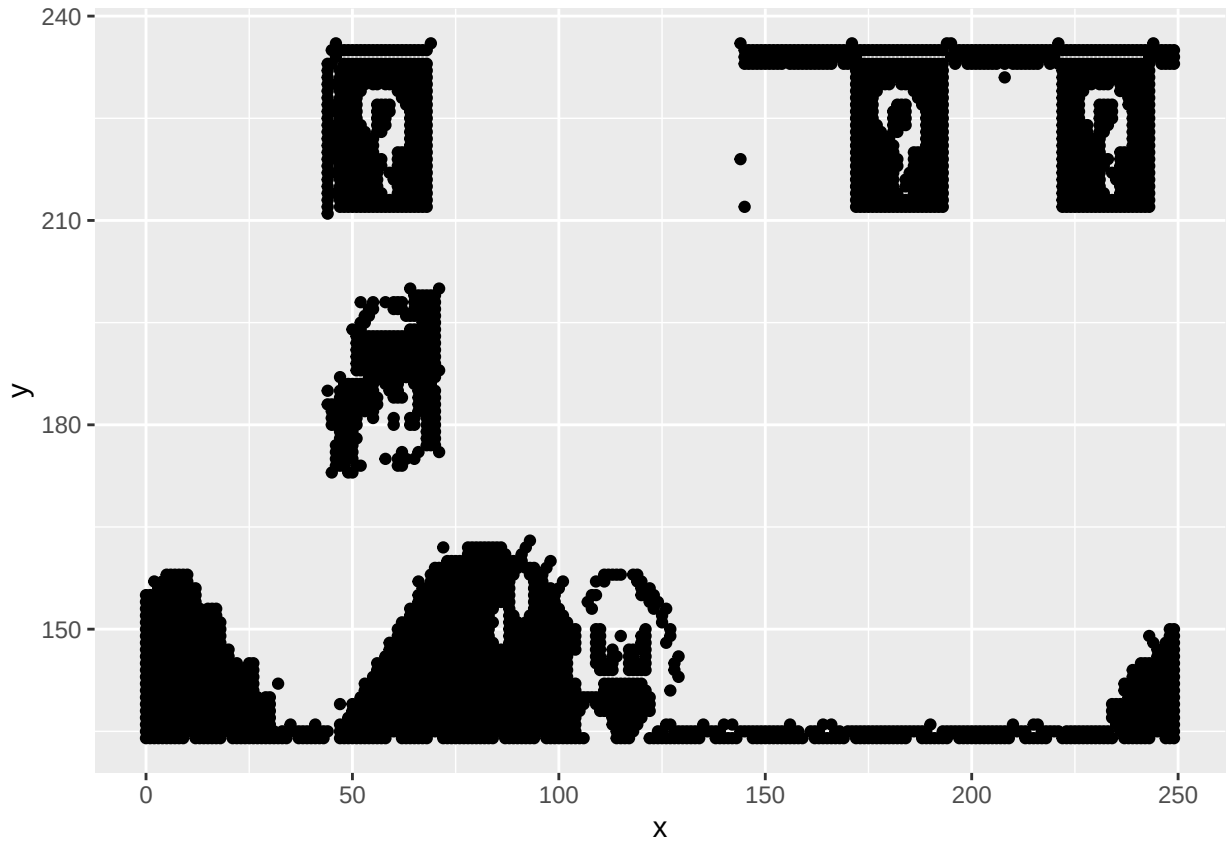
iv. Accuracy: Logistic vs. K-Nearest Neighbor

The accuracy of my original binary logistic model last week was 58%. We have significantly improved the accuracy by using the K Nearest Neighbor method to predict the classifiers. This is because of the methods that the models use to predict values. Regression models rely on a linear or quasi-linear trend appearing in the data. The data in this particular set appear in clusters distributed throughout the graphical environment, in which each classifier has multiple clusters that are spread out in a non-linear pattern. Due to this, it is much more reliably predicted by k nearest neighbors. The best use for a linear or logistic regression model is when certain variables correlate highly with the classifier (when a variable increases, so does the likelihood of the classifier). On the other hand, the best use for k nearest neighbors is when data points in the same classifier share similarities with each other, but an increase in the variable does not correlate with the strength of the classifier. The accuracy for the binary model peaks at 98.4% with k=10 clusters, while the accuracy for the trinary model peaks at 92.7% with k=3 Clusters.

2. Clustering

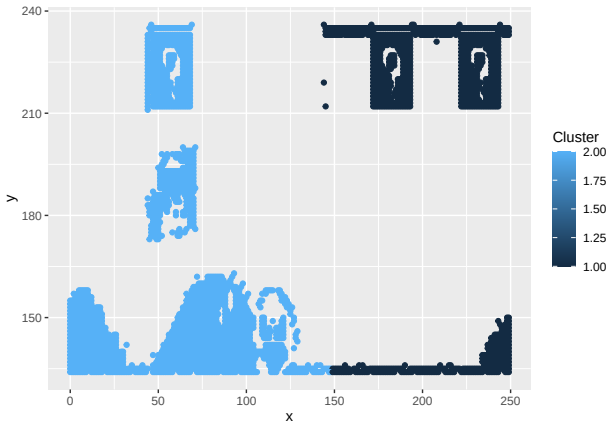
i. Scatter Plot

```
setwd("/Users/alissa/Documents/Grad/DSC 520/dsc520")
clustering_df <- read.csv("data/clustering-data.csv", header = TRUE)

ggplot(clustering_df, aes(x = x, y = y)) + geom_point()
```

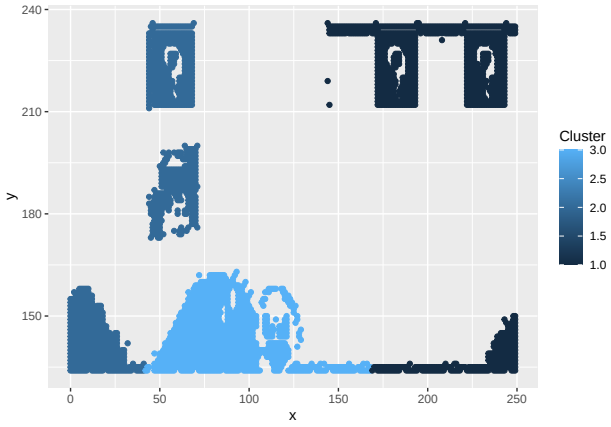


ii. K-Means Clustering

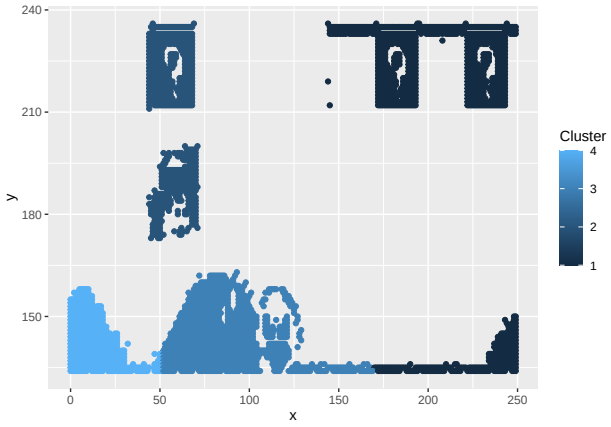
```
clusteringK2 <- kmeans(x = clustering_df, centers = 2)
clusteringK3 <- kmeans(x = clustering_df, centers = 3)
clusteringK4 <- kmeans(x = clustering_df, centers = 4)
clusteringK5 <- kmeans(x = clustering_df, centers = 5)
clusteringK6 <- kmeans(x = clustering_df, centers = 6)
clusteringK7 <- kmeans(x = clustering_df, centers = 7)
clusteringK8 <- kmeans(x = clustering_df, centers = 8)
clusteringK9 <- kmeans(x = clustering_df, centers = 9)
clusteringK10 <- kmeans(x = clustering_df, centers = 10)
clusteringK11 <- kmeans(x = clustering_df, centers = 11)
clusteringK12 <- kmeans(x = clustering_df, centers = 12)

clusteringC2 <- clustering_df
clusteringC2$Cluster <- clusteringK2$cluster
ggplot(clusteringC2, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



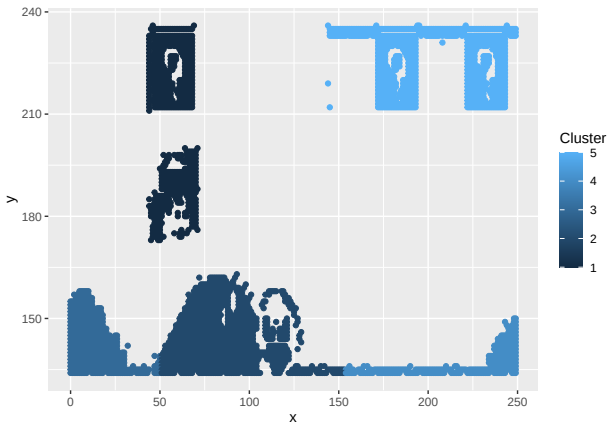

```
clusteringC3 <- clustering_df
clusteringC3$Cluster <- clusteringK3$cluster
ggplot(clusteringC3, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



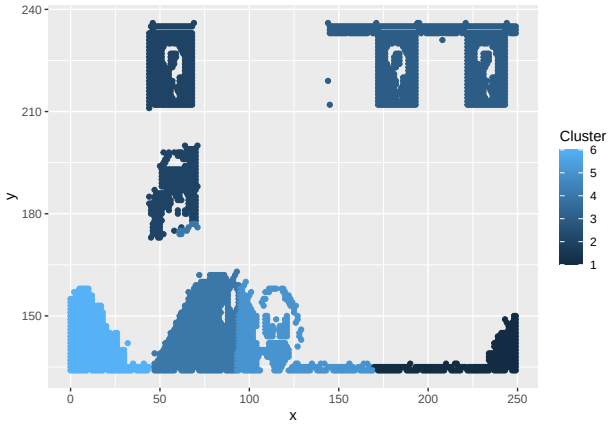

```
clusteringC4 <- clustering_df
clusteringC4$Cluster <- clusteringK4$cluster
ggplot(clusteringC4, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



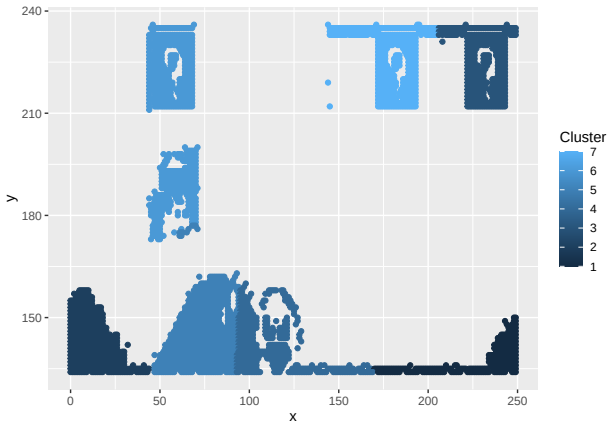
```
clusteringC5 <- clustering_df
clusteringC5$Cluster <- clusteringK5$cluster
ggplot(clusteringC5, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



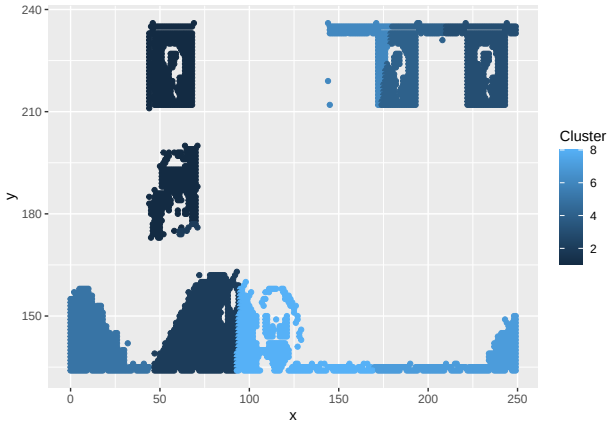

```
clusteringC6 <- clustering_df
clusteringC6$Cluster <- clusteringK6$cluster
ggplot(clusteringC6, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



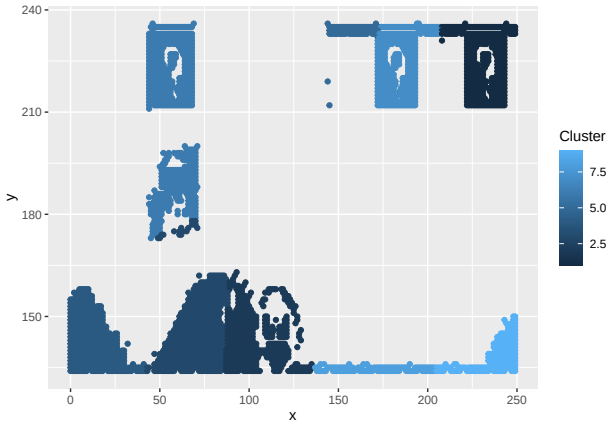

```
clusteringC7 <- clustering_df
clusteringC7$Cluster <- clusteringK7$cluster
ggplot(clusteringC7, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



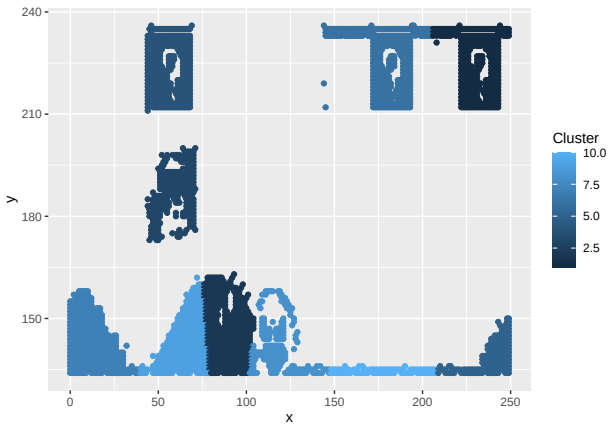

```
clusteringC8 <- clustering_df
clusteringC8$Cluster <- clusteringK8$cluster
ggplot(clusteringC8, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



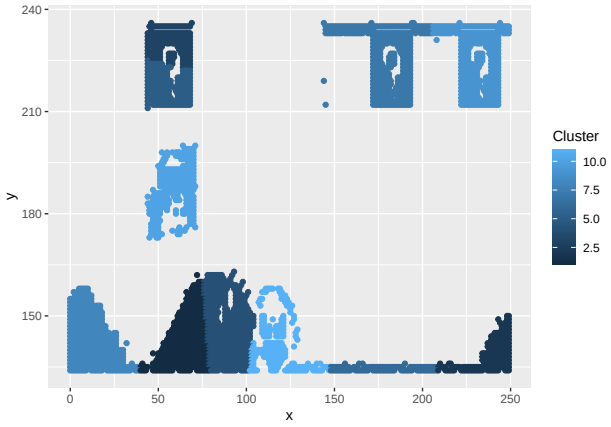

```
clusteringC9 <- clustering_df
clusteringC9$Cluster <- clusteringK9$cluster
ggplot(clusteringC9, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



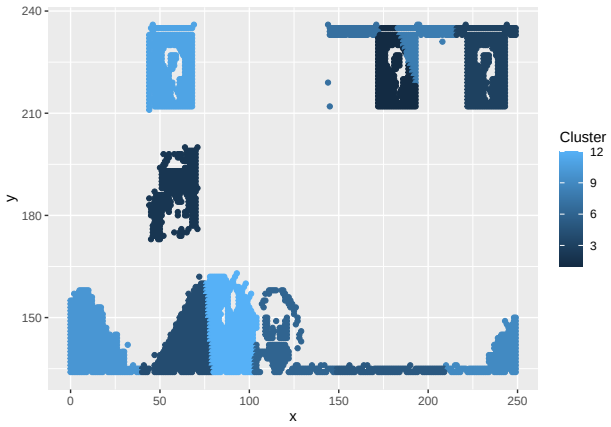
```
clusteringC10 <- clustering_df
clusteringC10$Cluster <- clusteringK10$cluster
ggplot(clusteringC10, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```




```
clusteringC11 <- clustering_df
clusteringC11$Cluster <- clusteringK11$cluster
ggplot(clusteringC11, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```




```
clusteringC12 <- clustering_df
clusteringC12$Cluster <- clusteringK12$cluster
ggplot(clusteringC12, aes(x = x, y = y, color = Cluster)) +
    geom_point()
```



iii. Measuring WSS

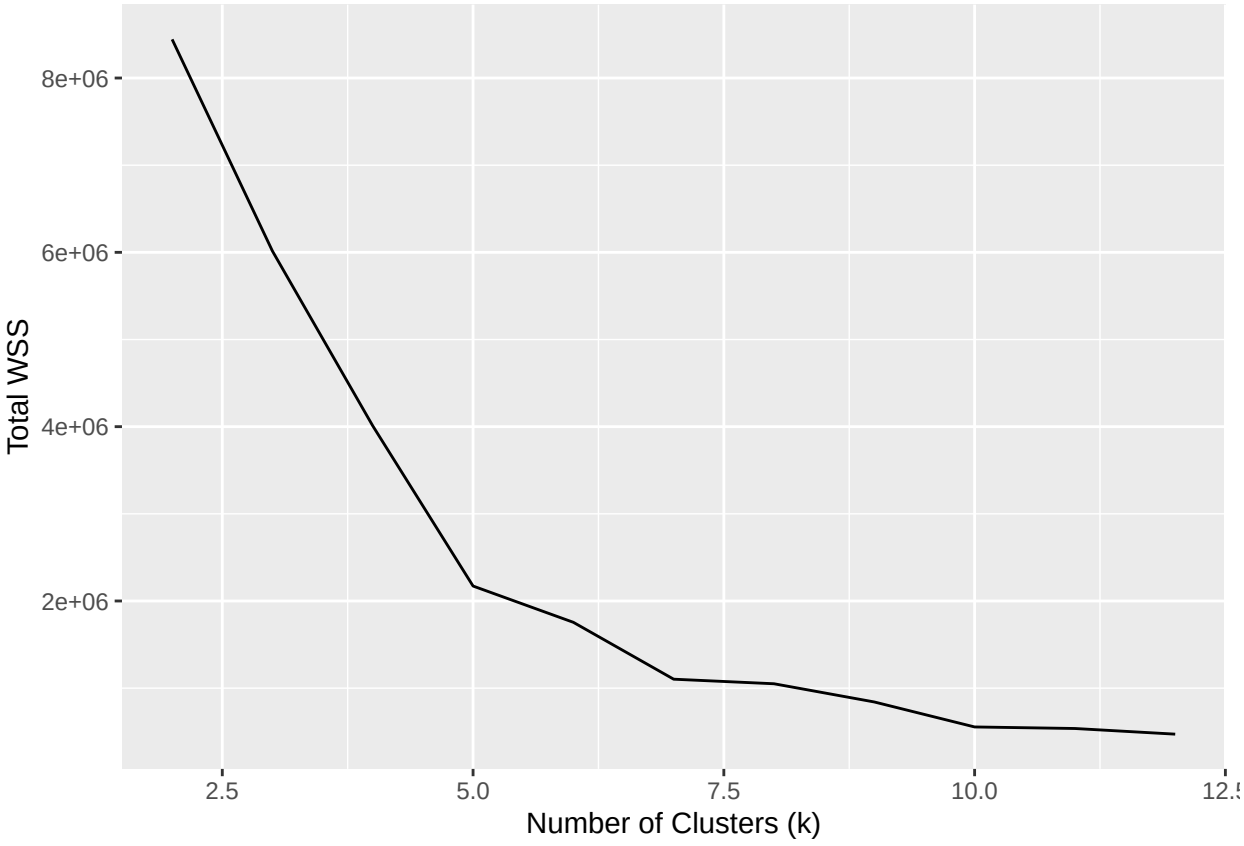
```
clusteringX <- c(2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
clusteringY <- c(clusteringK2$tot.withinss, clusteringK3$tot.withinss,
    clusteringK4$tot.withinss, clusteringK5$tot.withinss, clusteringK6$tot.withinss,
    clusteringK7$tot.withinss, clusteringK8$tot.withinss, clusteringK9$tot.withinss,
    clusteringK10$tot.withinss, clusteringK11$tot.withinss,
    clusteringK12$tot.withinss)

clusteringKWSS <- data.frame(clusteringX, clusteringY)
clusteringKWSS
```

```
##    clusteringX clusteringY
## 1            2   8443681.1
## 2            3   6014377.9
## 3            4   4009678.4
## 4            5   2171612.8
## 5            6   1755439.4
## 6            7   1102869.9
## 7            8   1050483.4
## 8            9    841593.1
## 9           10    554716.4
## 10          11    537210.3
## 11          12    472215.3
```

This table displays the within-cluster sum of squares for each value of K (denoted as clusteringX).

e. K-Value vs. WSS

```
ggplot(clusteringKWSS, aes(x = clusteringX, y = clusteringY)) +
    geom_line() + xlab("Number of Clusters (k)") + ylab("Total WSS")
```



f. Elbow Point

Our graph demonstrates an elbow point at k=5. This means that 5 clusters is ideal for this data set. Additional clusters would not bolster our findings enough to warrant dividing the data any further.