Notebook 3 Clustering

Dillon Carter

03/25

The dataset used for this project is an online retail set pulled from UCI. It contains tuples containing data on Customer ID, quantity purchased, price purchased at, description of the data, and country of origin of purchaser alongside some other columns.

Importing required packages into the project.

```
if(!require('tidyverse')){
  install.packages('tidyverse')
}
```

```
## Loading required package: tidyverse
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.1     ✓ purrr   1.0.1
## ✓ tibble  3.1.8     ✓ dplyr   1.1.0
## ✓ tidyr   1.3.0     ✓ stringr 1.5.0
## ✓ readr   2.1.4     ✓ forcats 1.0.0
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
```

```
if(!require('e1071')){
  install.packages('e1071')
}
```

```
## Loading required package: e1071
```

```
if(!require('caret')){
  install.packages('caret')
}
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require('word2vec')){
  install.packages('word2vec')
}
```

```
## Loading required package: word2vec
```

```
## Warning: package 'word2vec' was built under R version 4.2.3
```

```
if(!require('cluster')){
  install.packages('cluster')
}
```

```
## Loading required package: cluster
```

```
if(!require('mclust')){
  install.packages('mclust')
}
```

```
## Loading required package: mclust
```

```
## Warning: package 'mclust' was built under R version 4.2.3
```

```
## Package 'mclust' version 6.0.0
## Type 'citation("mclust")' for citing this R package in publications.
##
## Attaching package: 'mclust'
##
## The following object is masked from 'package:purrr':
##
##     map
```

```
library('word2vec')
library('caret')
library('tidyverse')
library('e1071')
library('cluster')
library('mclust')
data <- read.csv("data/online_retail.csv", header=TRUE)
set.seed(1234)
str(data)
```

```
## 'data.frame':    541909 obs. of  8 variables:
##  $ InvoiceNo  : chr  "536365" "536365" "536365" "536365" ...
##  $ StockCode  : chr  "85123A" "71053" "84406B" "84029G" ...
##  $ Description: chr  "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUP
ID HEARTS COAT HANGER" "KNITTED UNION FLAG HOT WATER BOTTLE" ...
##  $ Quantity   : int  6 6 8 6 6 2 6 6 6 32 ...
##  $ InvoiceDate: chr  "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26"
...
##  $ UnitPrice  : num  2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...
##  $ CustomerID : int  17850 17850 17850 17850 17850 17850 17850 17850 17850 13047 ...
##  $ Country    : chr  "United Kingdom" "United Kingdom" "United Kingdom" "United Kingdom"
...
```

```
normalize <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}
```

I'm taking the Customer ID, Price, Quantity, and Country of Origin as the values to evaluate on. I'm omitting NA data instead of replacing it, as there are enough values already in the dataset to where removing some will not affect the data in a major way. Additionally, I get rid of some values that are not possible, i.e. removing negative bought items or items with a negative purchase price.

```
zero <- 0.0
data <- data[, c(4,6,7,8)]
data <- na.omit(data)
data <- data[data$Quantity > 0,]
data <- data[c(data$UnitPrice > as.numeric(zero)),]
```

I'm leaving this in, even though it failed. What I had attempted was to use the Word to Vec library to attempt to turn the character strings of Descriptions into a format that could be used in clustering. I wanted to create prediction matrices that could be used to help determine the likelyhood of a user ID purchasing something based on similar words. This was extremely time intensive and in the end, I couldn't get something to work in the time I had allotted.

```
##temp <- data$Description
#word_vector <- word2vec(temp, min_count = 1L, split=c(" \t", "\n."))
#emb <- as.matrix(word_vector)
```

Here is checking to make sure that the data cleaning steps worked.

```
i <- which.min(data$Quantity)
print(data[i,])
```

```
##     Quantity UnitPrice CustomerID        Country
## 114        1      1.25      15311 United Kingdom
```

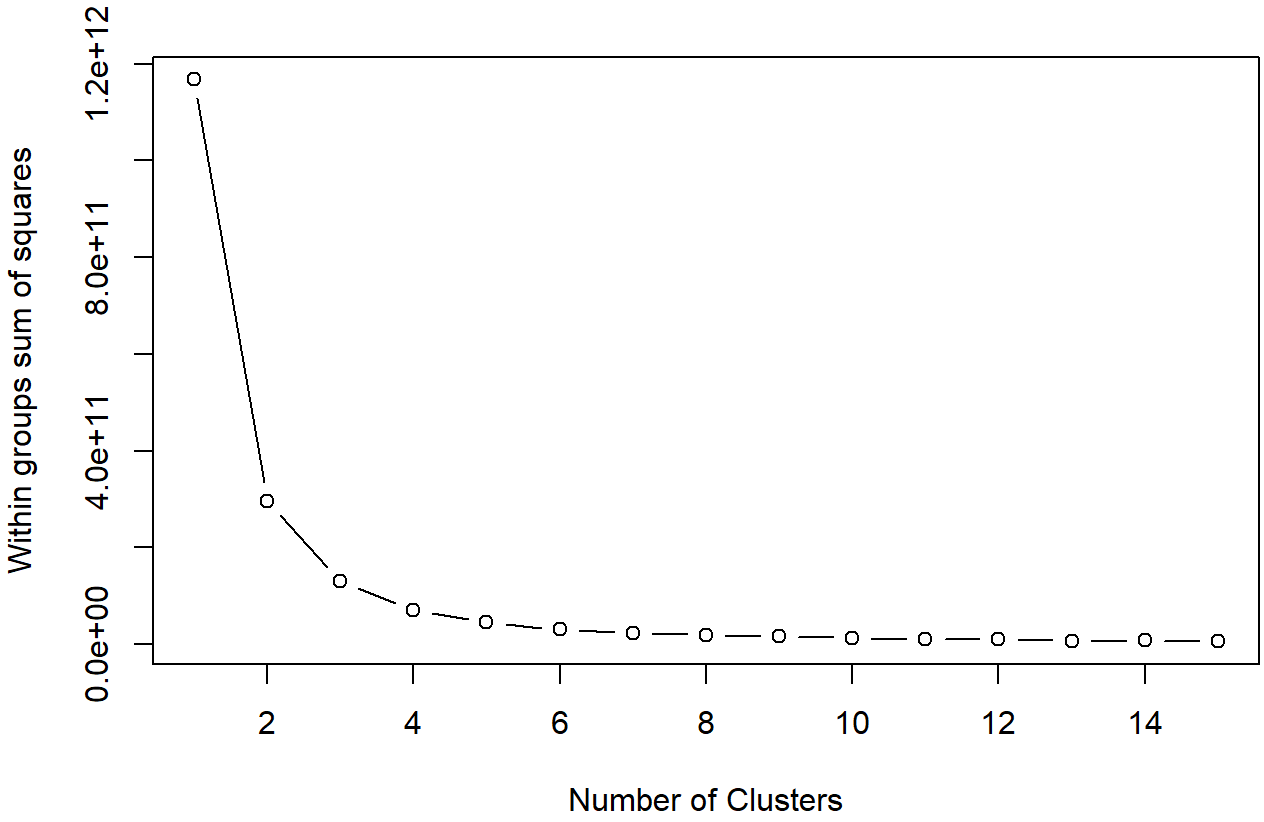
```
i <- which.max(data$Quantity)
print(data[i,])
```

```
##        Quantity UnitPrice CustomerID        Country
## 540422    80995      2.08      16446 United Kingdom
```

```
data_norm <- as.data.frame(lapply(data[,1:2], normalize))
data_norm$CustomerID <- data$CustomerID
#data_norm$Country <- factor(data$Country)

##Attempted to convert character vectors into a usuable format for clustering
##Unable to find a good solution
#for(x in data$Description){
#  split <- unlist(lapply(sapply(x, strsplit, "\\s+|\\.+", USE.NAMES=FALSE), function(Z) {Z
[Z!= ""]}))
#  vector <- rep(0, 50)
#  for(y in split){
#    predictions = tryCatch({
#      vector <- vector + emb[y, ]
#    }, error = function(e) "")
#  }
#  predictions <- rbind(predict(word_vector, vector, type='embedding'))
#}
```

With clustering, it is good to know how many clusters to push everything into. Plotting by the sum of squares will give a good approximation of the number of clusters to form. Of course, this isn't the absolute value to follow so I will try with some other values.

```
wss <- (nrow(data_norm)-1)*sum(apply(data_norm,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(data_norm,
   centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
  ylab="Within groups sum of squares")
```



It would seem that around 2 - 4 clusters is a good idea. The main change in the slope occurs between 2 and 3 clusters but doesn't really level off until around 4 to 5.

```
retail2KCluster <- kmeans(data_norm, 2, nstart=20)
retail3KCluster <- kmeans(data_norm, 3, nstart=20)
retail4KCluster <- kmeans(data_norm, 4, nstart=20)
aggregate(data_norm,by=list(retail2KCluster$cluster),FUN=mean)
```

```
##   Group.1     Quantity    UnitPrice CustomerID
## 1       1 0.0001372414 0.0003591180   16874.32
## 2       2 0.0001574700 0.0004032289   13907.60
```

```
aggregate(data_norm,by=list(retail3KCluster$cluster),FUN=mean)
```

```
##   Group.1     Quantity    UnitPrice CustomerID
## 1       1 0.0001376345 0.0003766579   15148.45
## 2       2 0.0001408467 0.0003495202   17275.62
## 3       3 0.0001676361 0.0004258166   13274.43
```

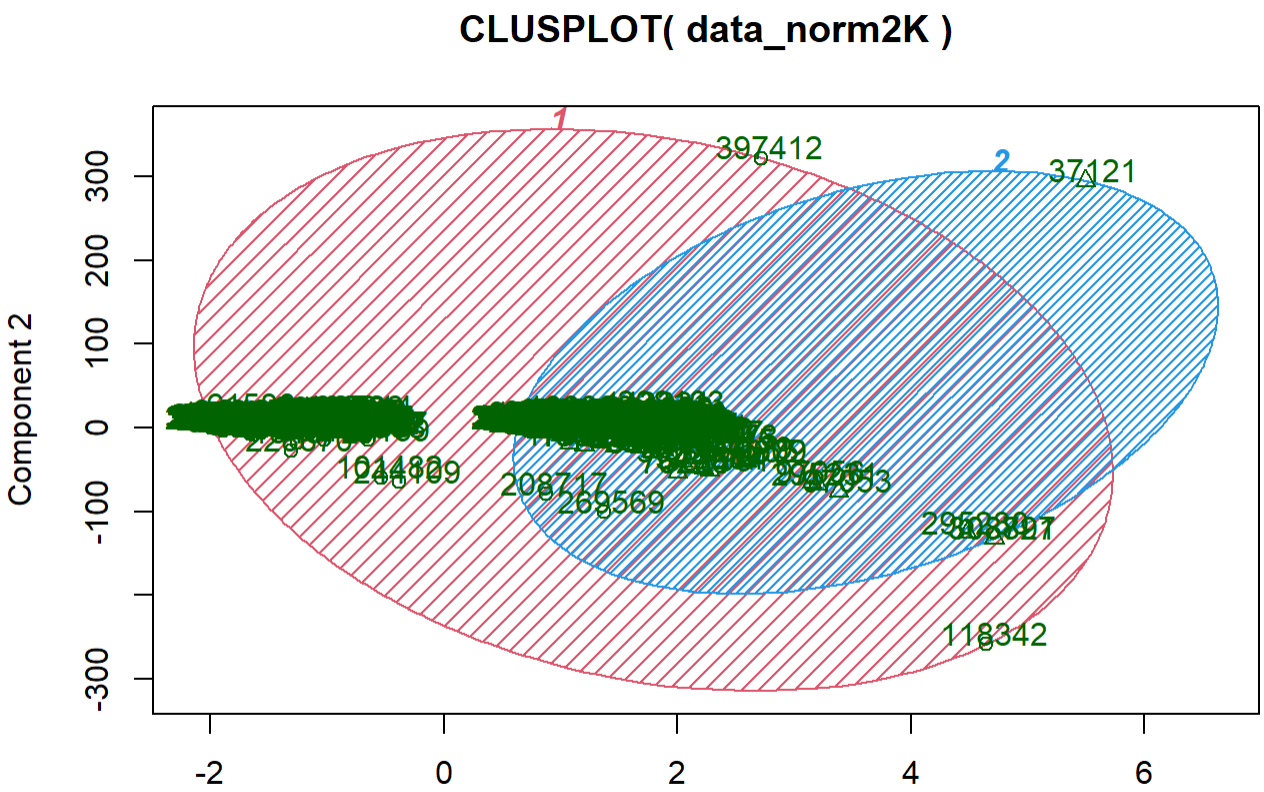
```
aggregate(data_norm,by=list(retail4KCluster$cluster),FUN=mean)
```

```
##   Group.1     Quantity    UnitPrice CustomerID
## 1       1 0.0001321754 0.0003526209   17562.37
## 2       2 0.0001416246 0.0003649616   16091.85
## 3       3 0.0001432305 0.0003924587   14572.58
## 4       4 0.0001771505 0.0004194005   13017.31
```

```
data_norm2K <- data.frame(data_norm, retail2KCluster$cluster)
data_norm3K <- data.frame(data_norm, retail3KCluster$cluster)
data_norm4K <- data.frame(data_norm, retail4KCluster$cluster)
```

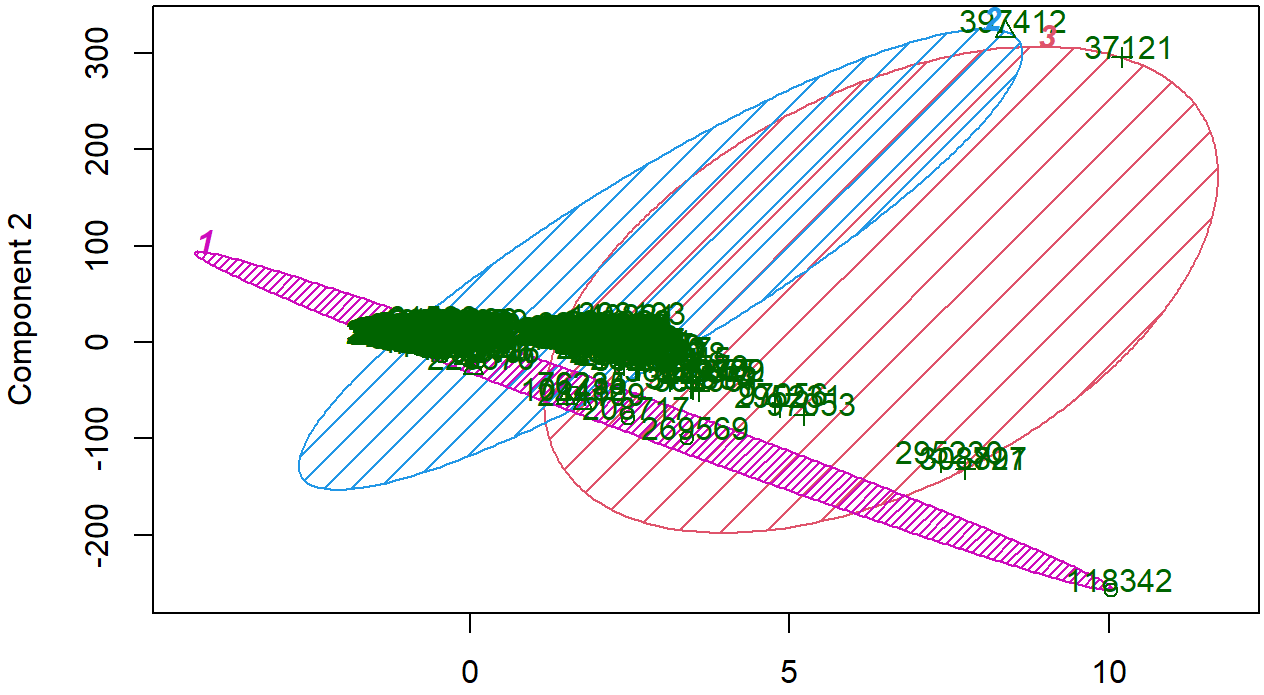
Now lets check the results of the clustering.

```
clusplot(data_norm2K, retail2KCluster$cluster, color=TRUE, shade=TRUE,
   labels=2, lines=0)
```

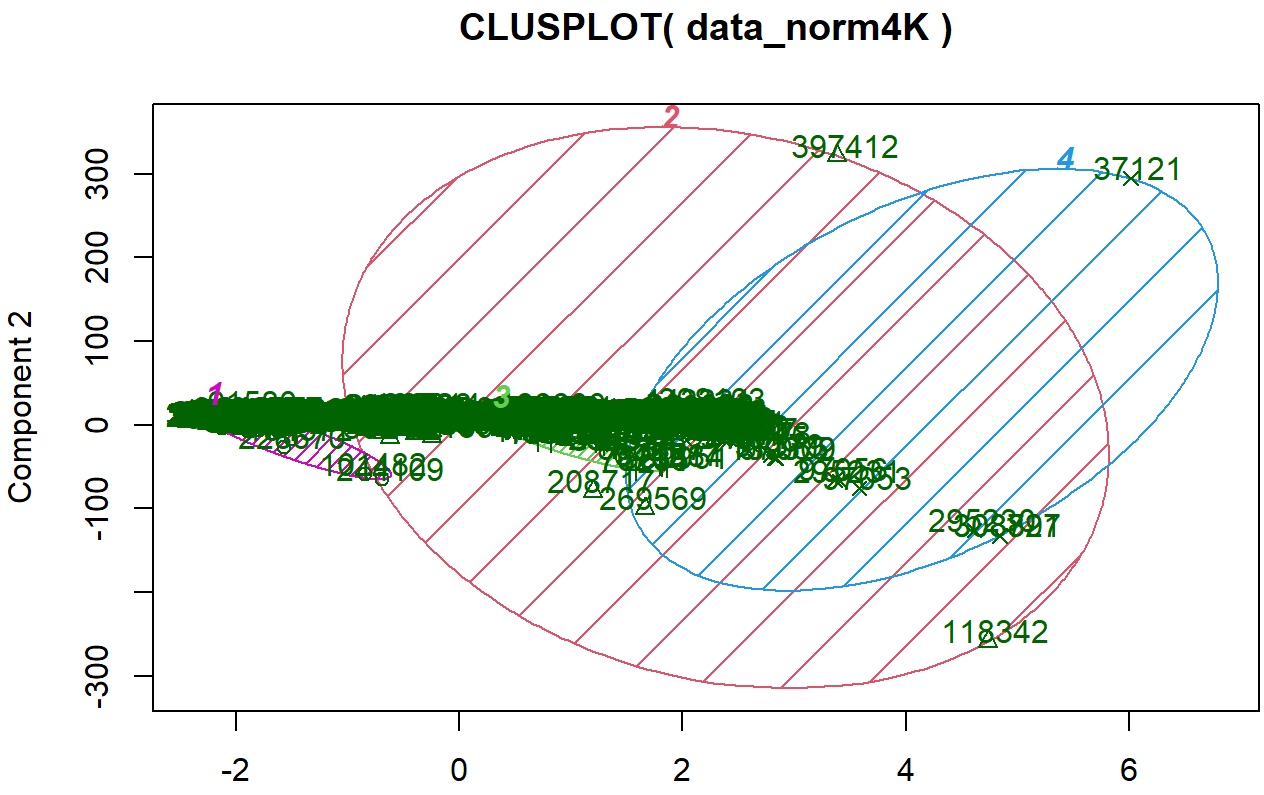


These two components explain 71.72 % of the point variability.

```
clusplot(data_norm3K, retail3KCluster$cluster, color=TRUE, shade=TRUE,
   labels=2, lines=0)
```




```
clusplot(data_norm4K, retail4KCluster$cluster, color=TRUE, shade=TRUE,
   labels=2, lines=0)
```




```
summary(retail2KCluster)
```

```
##              Length Class  Mode
## cluster      397884 -none- numeric
## centers           6 -none- numeric
## totss             1 -none- numeric
## withinss          2 -none- numeric
## tot.withinss      1 -none- numeric
## betweenss         1 -none- numeric
## size              2 -none- numeric
## iter              1 -none- numeric
## ifault            1 -none- numeric
```

```
summary(retail3KCluster)
```

```
##              Length Class  Mode
## cluster      397884 -none- numeric
## centers           9 -none- numeric
## totss             1 -none- numeric
## withinss          3 -none- numeric
## tot.withinss      1 -none- numeric
## betweenss         1 -none- numeric
## size              3 -none- numeric
## iter              1 -none- numeric
## ifault            1 -none- numeric
```

```
summary(retail4KCluster)
```

```
##              Length Class  Mode
## cluster      397884 -none- numeric
## centers          12 -none- numeric
## totss             1 -none- numeric
## withinss          4 -none- numeric
## tot.withinss      1 -none- numeric
## betweenss         1 -none- numeric
## size              4 -none- numeric
## iter              1 -none- numeric
## ifault            1 -none- numeric
```

Heirachical Agglomerative The data's far too big for this clustering method, so I'm cutting it down drastically.

```
temp <- sample(1:nrow(data_norm), 0.1*nrow(data_norm), replace=FALSE)
reduced_normalized <- data[temp,]
```

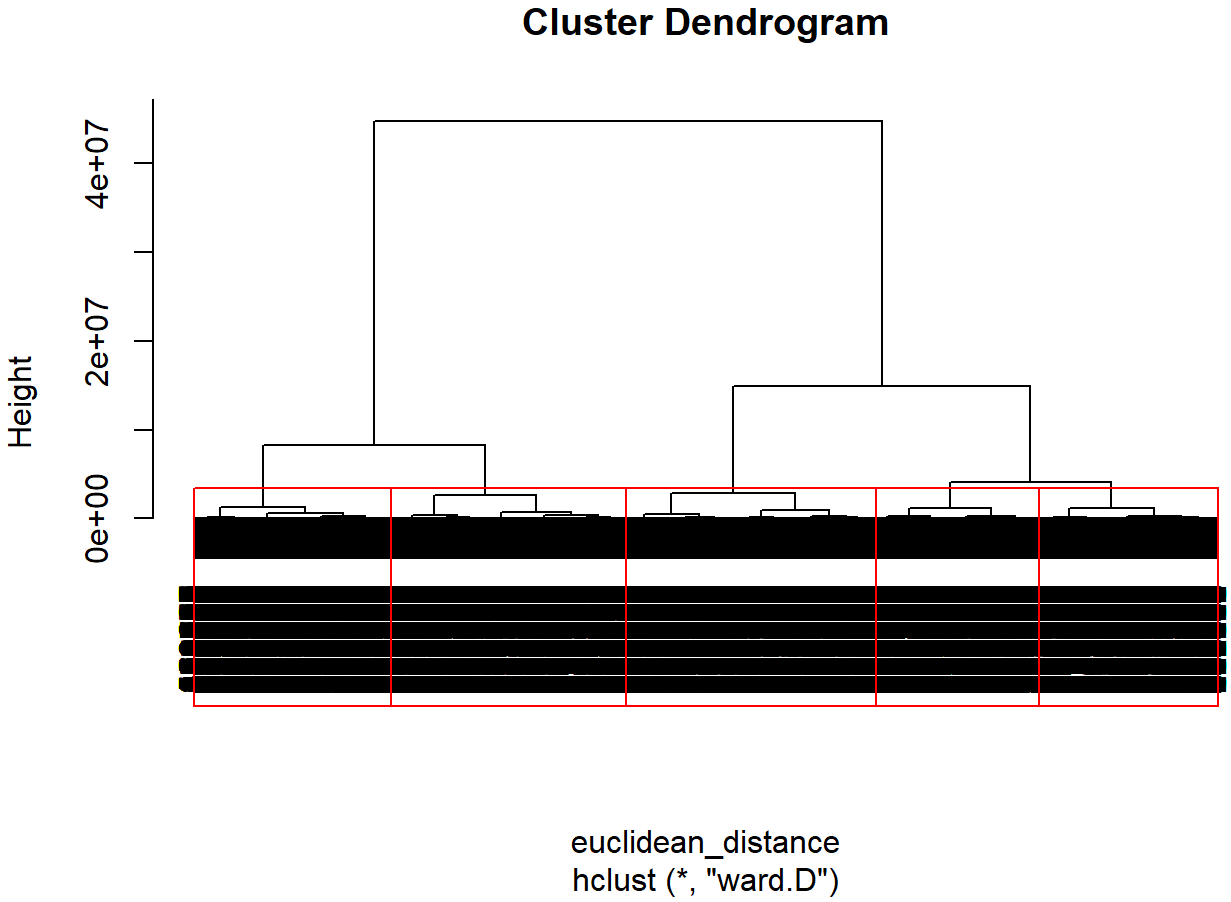
```
# Ward Hierarchical Clustering
euclidean_distance <- dist(reduced_normalized, method = "euclidean") # distance matrix
```

```
## Warning in dist(reduced_normalized, method = "euclidean"): NAs introduced by
## coercion
```

```
retailHeirachical <- hclust(euclidean_distance, method="ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
plot(retailHeirachical) # display dendogram
groups <- cutree(retailHeirachical, k=5) # cut tree into 5 clusters
# draw dendogram with red borders around the 5 clusters
rect.hclust(retailHeirachical, k=5, border="red")
```

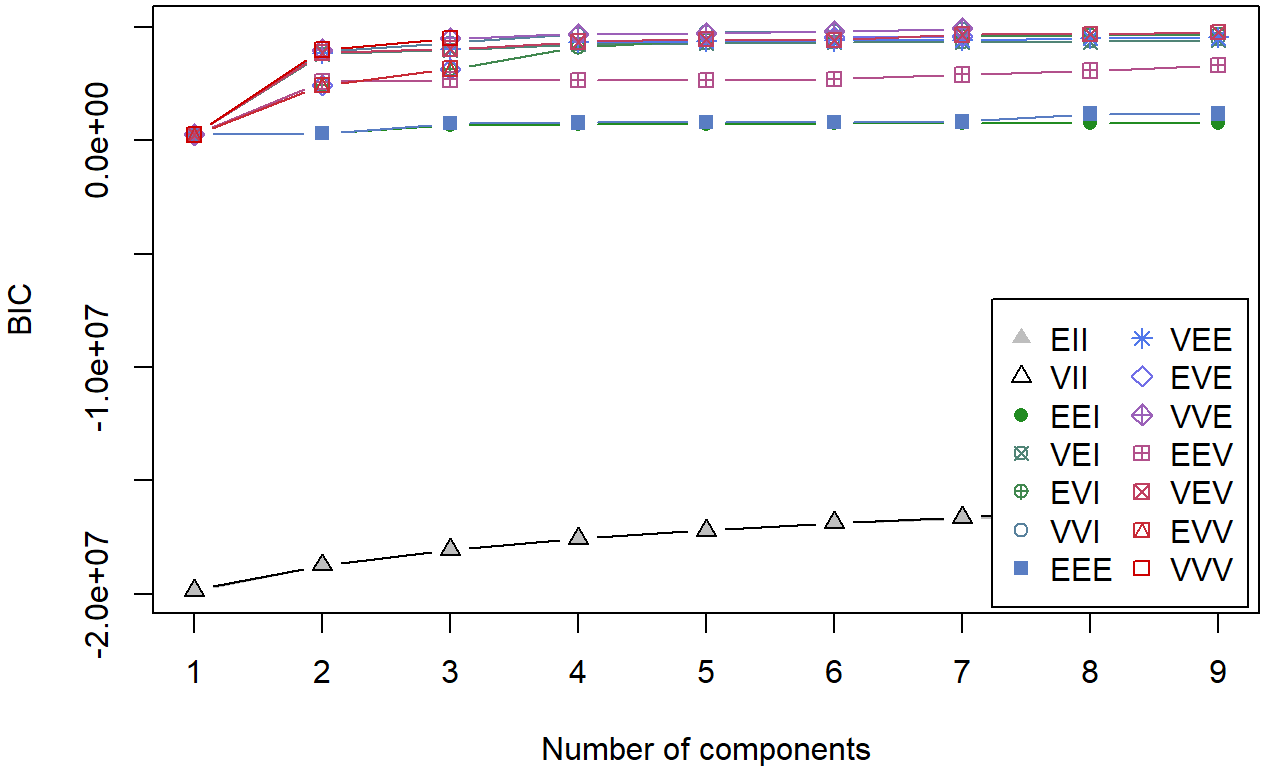


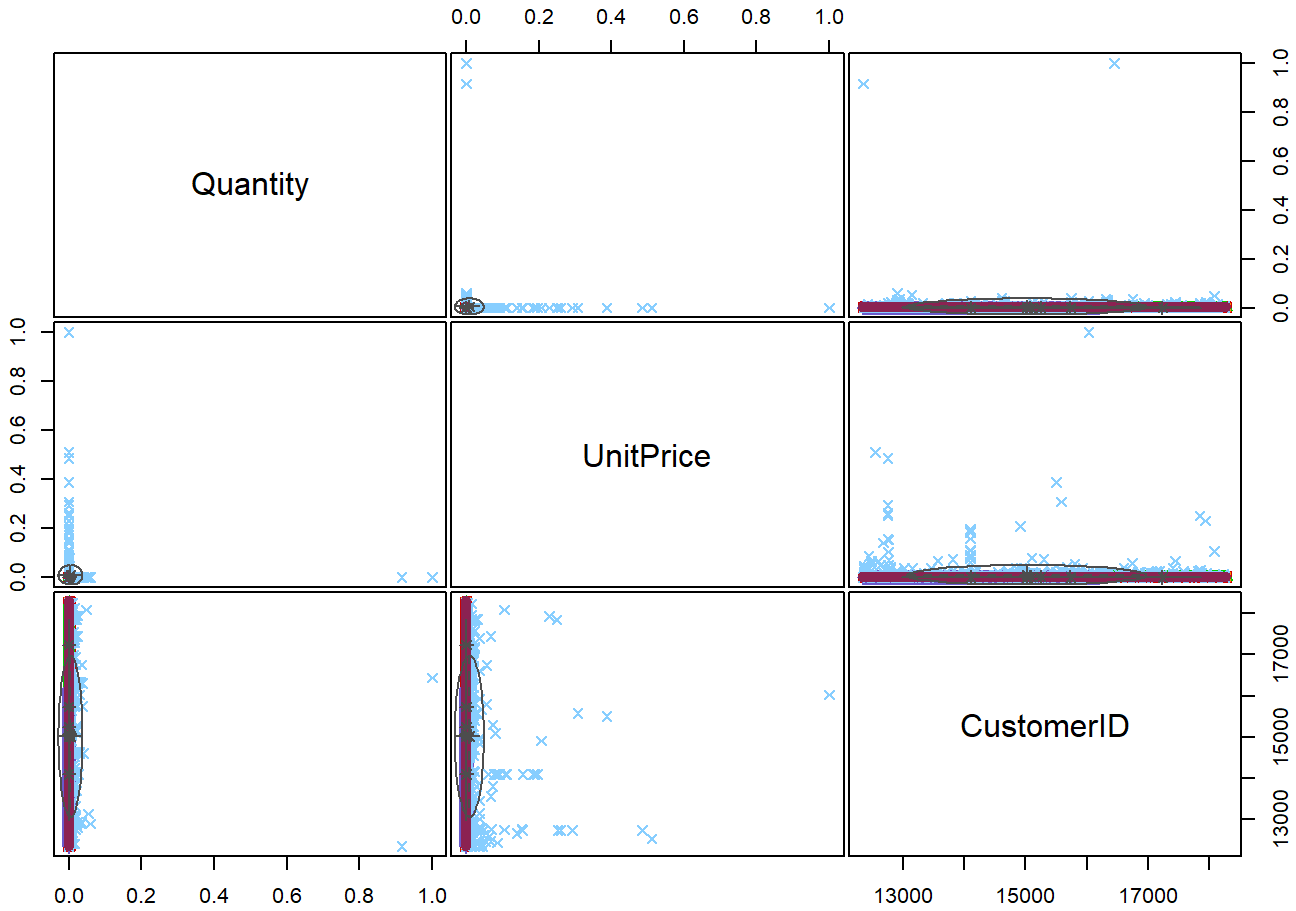
Model Based

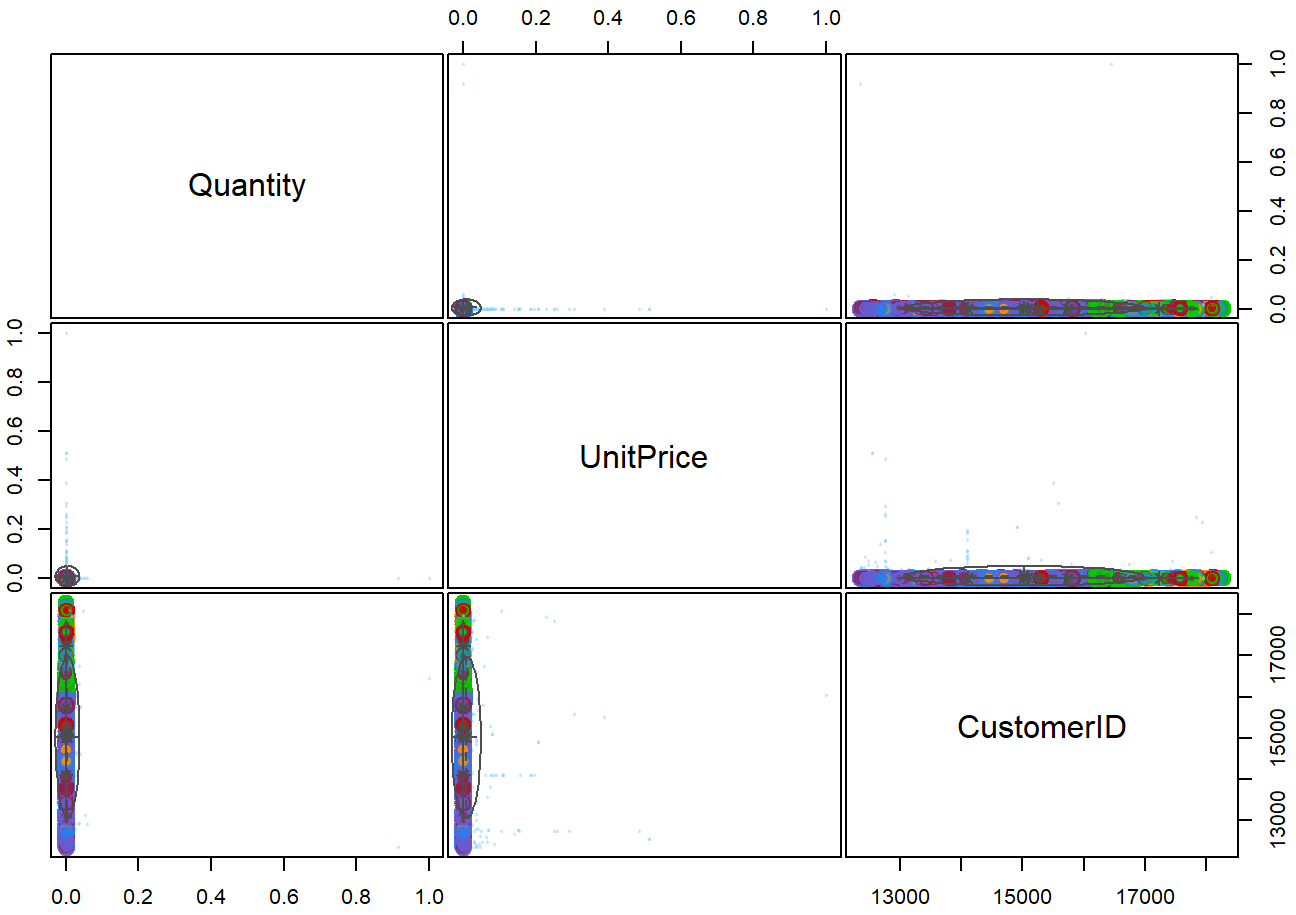
```
# Model Based Clustering

retailModel <- Mclust(data_norm)
summary(retailModel) # display the best model
```

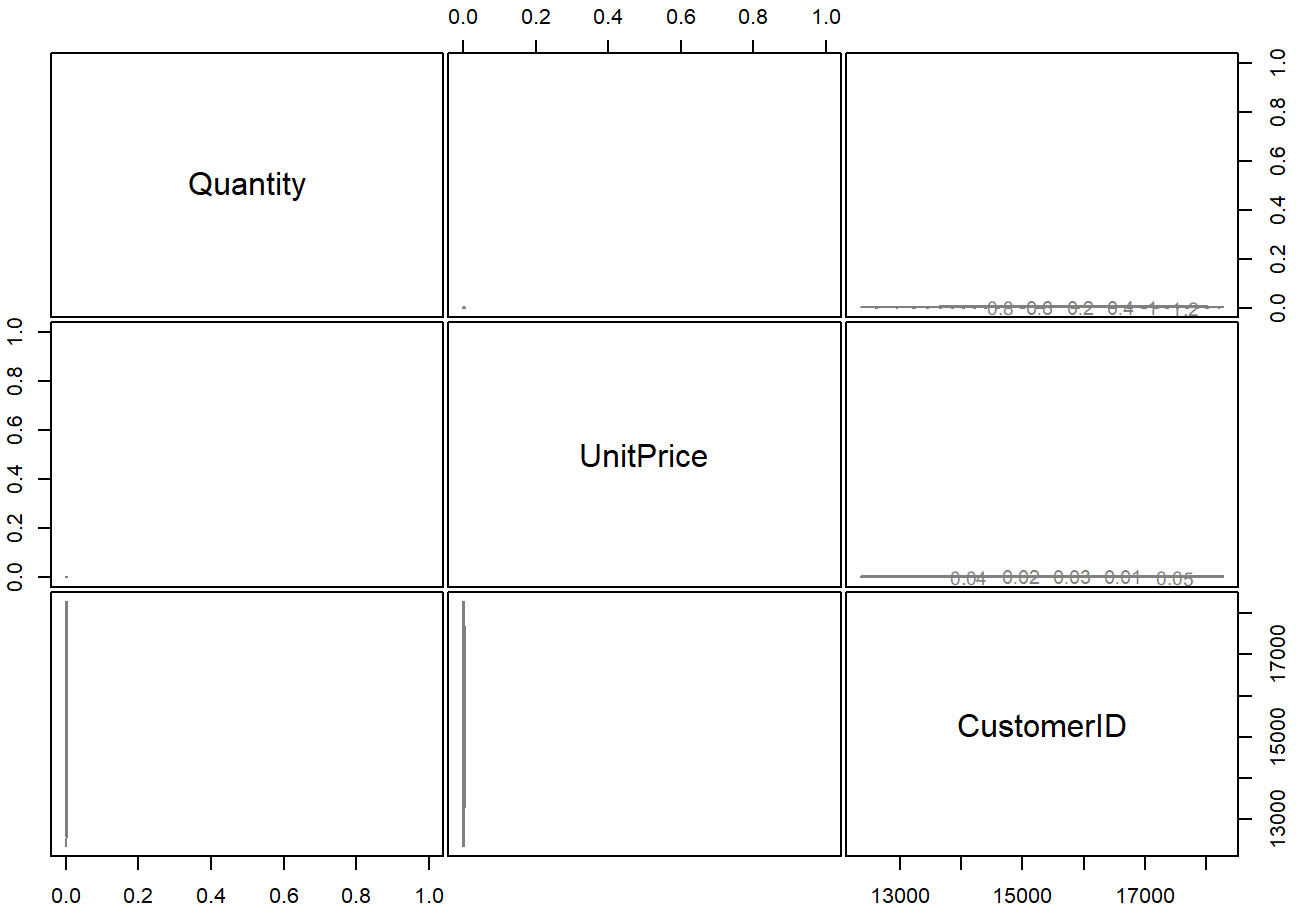
```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVE (ellipsoidal, equal orientation) model with 7 components:
##
##  log-likelihood      n df     BIC     ICL
##         2467357 397884 51 4934057 4788163
##
## Clustering table:
##     1     2     3     4     5     6     7
## 91586 20463 52536 98097 85396  1722 48084
```

```
plot(retailModel)
```


BIC
0.0e+00
-1.0e+07
-2.0e+07
1
2
3
4
5
6
7
8
9
Number of components
EII
VII
EEI
VEI
EVI
VVI
EEE
VEE
EVE
VVE
EEV
VEV
EVV
VVV


Quantity
UnitPrice
CustomerID
0.0
0.2
0.4
0.6
0.8
1.0
13000
15000
17000


Quantity
UnitPrice
CustomerID
0.0 0.2 0.4 0.6 0.8 1.0
13000 15000 17000


Quantity
UnitPrice
CustomerID
0.0 0.2 0.4 0.6 0.8 1.0
13000 15000 17000

The results for each of the models varied in its usefulness. The most useful I would say for this data set was the K means clustering algorithm. Thought I had to specify the number of clusters to use, the sum of squares algorithm identified three clusters that might be useful in 2, 3, and 4 clusters. 3 and 4 did not provide any useful information. But 2 classified information into 2 usable clusters that could be used with predictions. The heirarchical clustering is not useful in a dataset of this size. I attempted to cut the data set down to a 1/10th of its original size and still could not create a legible clustering graph. Going down to smaller sizes provided a more legible graph but due to the random nature of picking such a small number of values from the dataset, it did not provide a consistent method of identifying groups. Finally, the model based clustering was better than the heirachical but due to its more general nature, did not compare with the output produced by the K means cluster.