assignment 2 bikes

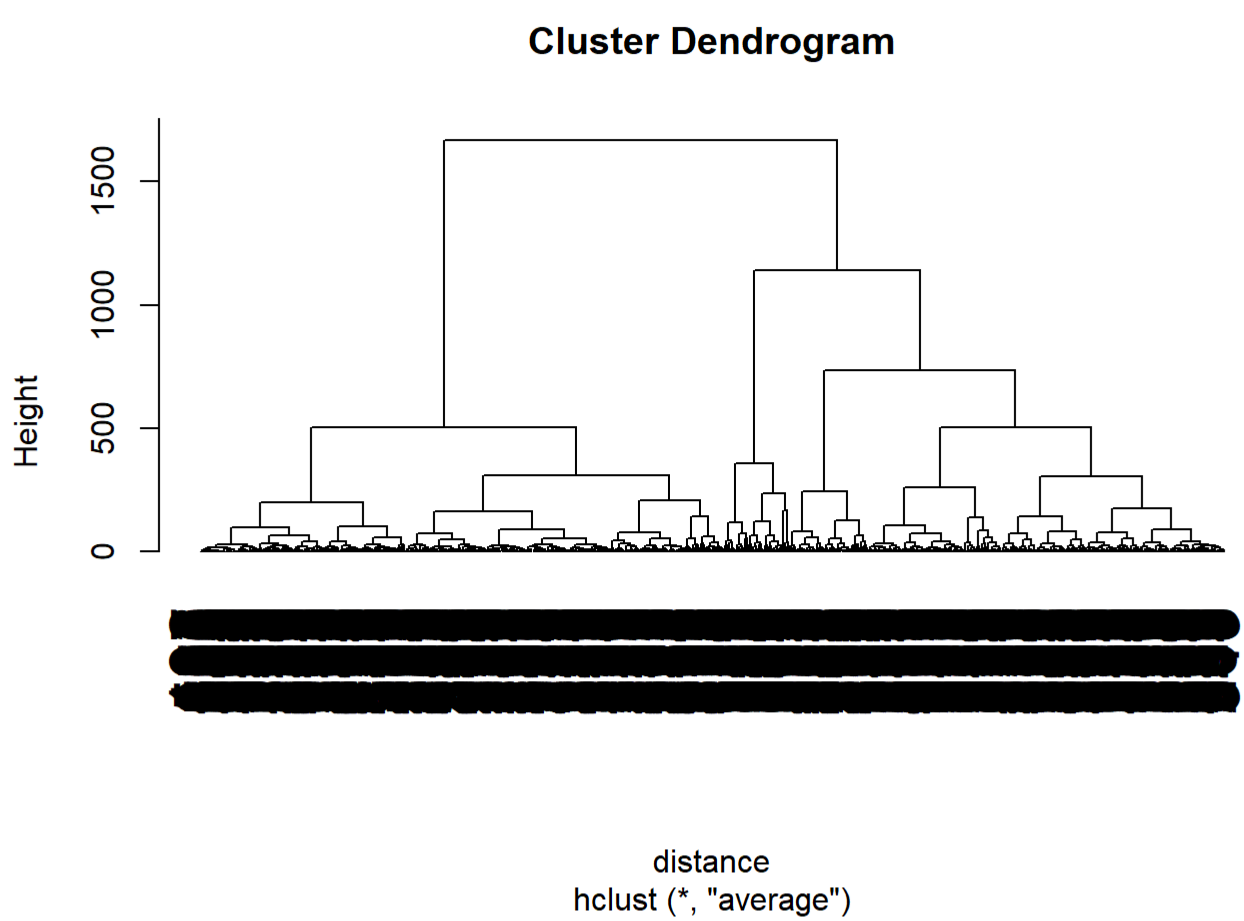
Drew Murray

10/1/2022

```
Bikes <- read.csv("C:/Users/dgmur/Downloads/SeoulBikes_F12022.csv")
View(Bikes)
bikes2 <- Bikes[,c(1:4,12)]
View(bikes2)
```

#a

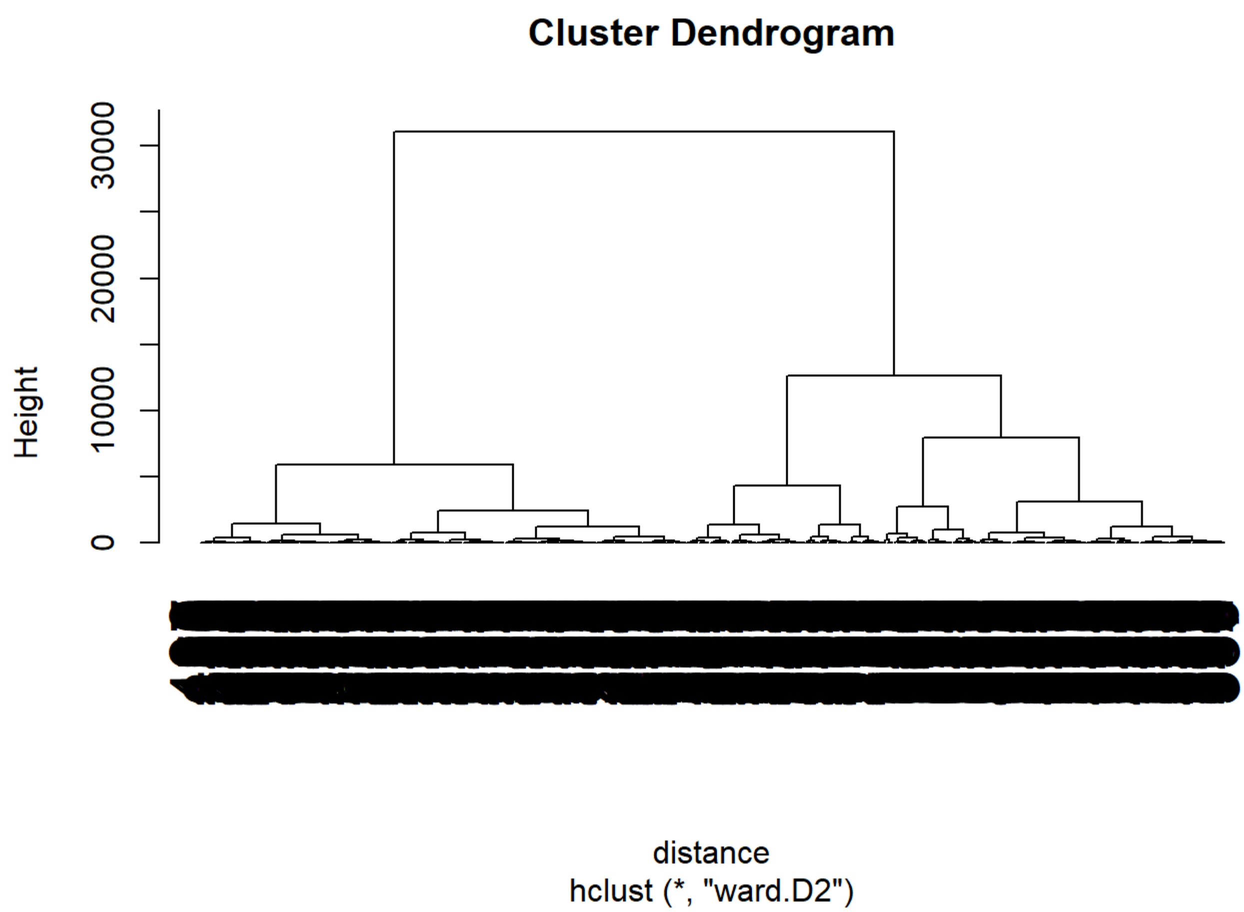
```
distance<- dist(bikes2)
a <- plot(hclust(distance, "average"), hang = -3,cex=2)
```




```
a
```

```
## NULL
```

```
b<- plot(hclust(distance, "ward.D2"), hang = -3,cex=2)
```



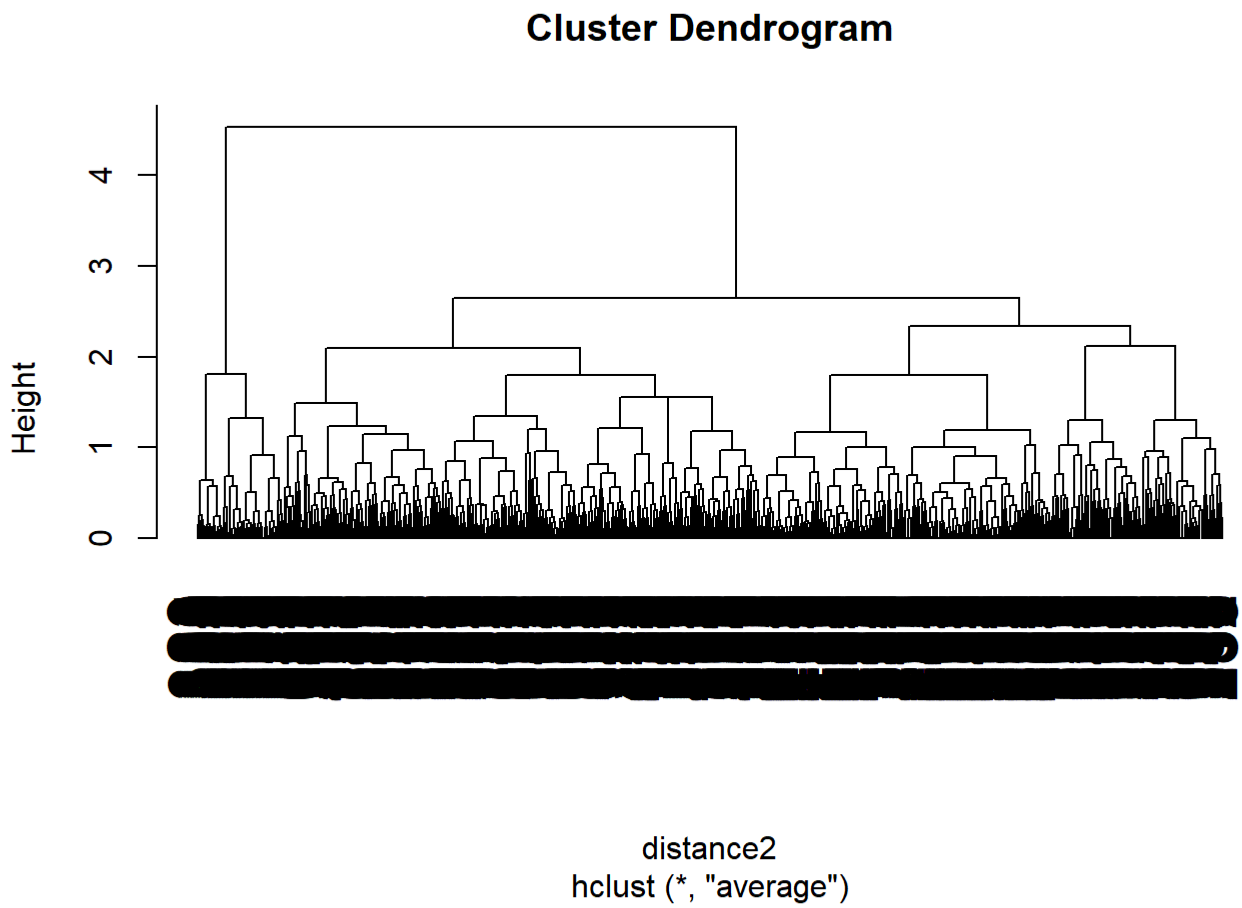

```
b
```

```
## NULL
```

#no, it’s a bit challenging to determine the number of clusters

#b

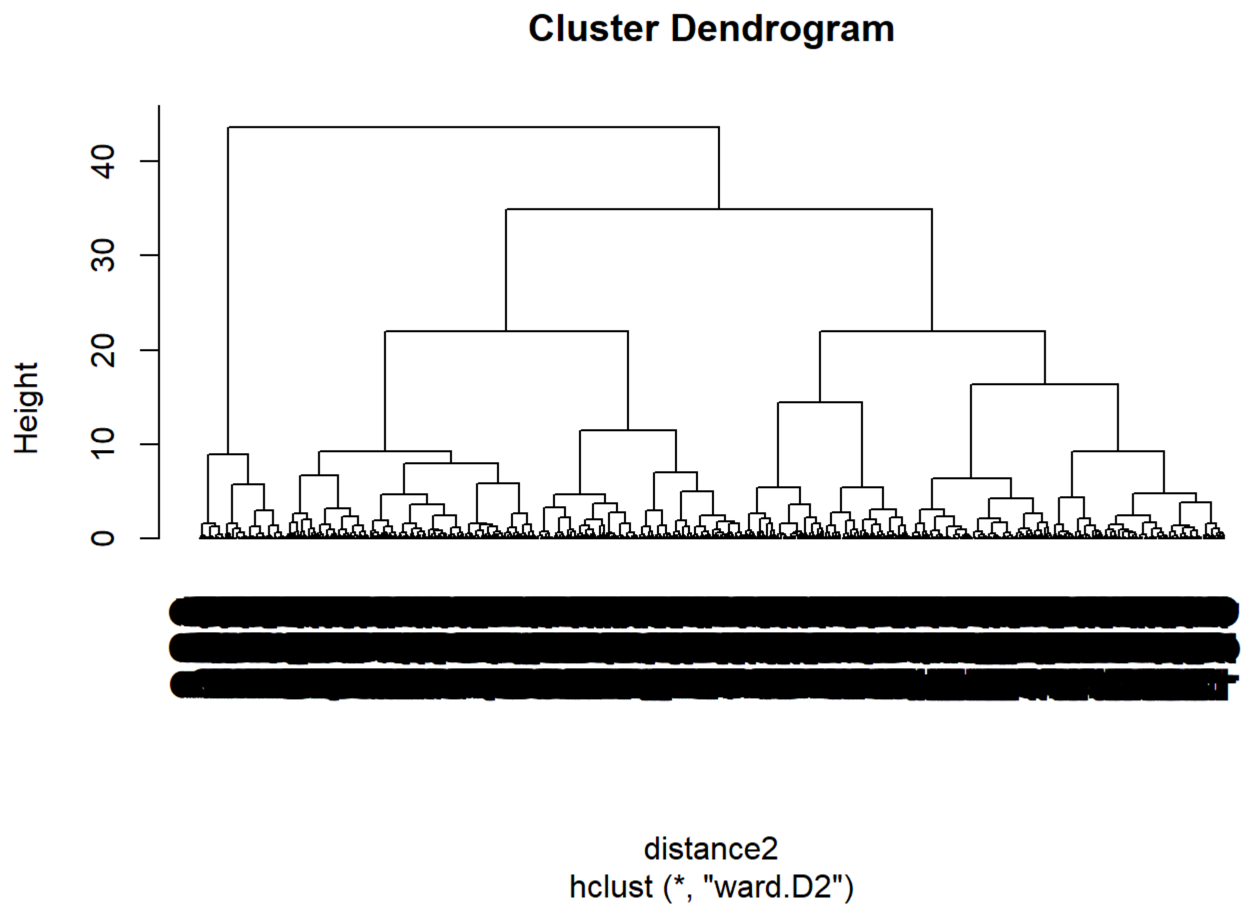
```
bikes2s <- apply(bikes2[ , 1:4], 2, function(x) (x-mean(x))/sd(x))
distance2 <- dist(bikes2s)
c<- plot(hclust(distance2, "average"), hang = -3,cex=2)
```




```
c
```

```
## NULL
```

```
d<- plot(hclust(distance2, "ward.D2"), hang = -3,cex=2)
```




```
d
```

```
## NULL
```

#I prefer the standardized because it’s easier to determine the actual amount of clusters in the dendrogram

#c

```
clust4<- kmeans(bikes2,4)
clust4
```

```
## K-means clustering with 4 clusters of sizes 214, 197, 200, 119
##
## Cluster means:
##        Day    Month     Year RentedBikeCount    Snowfall
## 1 17.17290 7.476636 2018.000       1876.0467 0.001401869
## 2 14.53299 6.035533 2017.848        771.1066 0.018274112
## 3 15.55000 5.715000 2017.840        216.6050 0.257500000
## 4 15.36134 6.991597 2018.000       2710.8908 0.000000000
##
## Clustering vector:
##   [1] 2 3 3 2 2 2 2 2 3 3 2 2 2 2 2 3 3 3 3 3 3 2 3 3 3 2 2 2 2 3 3 3 2 2 2 2 3
##  [38] 3 2 2 3 3 3 3 3 3 2 2 2 2 3 3 2 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 2 2 3 3 2
##  [75] 2 2 3 3 3 3 2 2 2 2 3 3 3 2 2 2 3 2 3 3 3 2 2 3 2 3 3 2 2 1 3 2 3 3 2 2 2
## [112] 2 2 3 3 2 2 1 2 1 3 3 1 1 2 3 2 3 3 2 1 3 1 1 3 3 1 1 1 3 1 2 3 3 1 1 1 1
## [149] 2 3 1 2 2 1 1 2 3 2 1 1 3 1 3 3 1 1 3 3 3 2 2 1 2 1 1 1 2 2 1 1 1 1 1 2 2
## [186] 1 4 2 4 4 2 2 4 4 2 3 4 2 2 4 4 4 4 1 2 2 4 3 1 1 1 2 3 3 1 1 1 1 2 2 2 1
## [223] 2 1 1 2 2 1 1 1 1 1 2 3 1 1 1 1 1 2 3 1 1 1 1 1 2 3 2 1 1 1 1 2 2 1 1 2 1
## [260] 1 2 2 1 1 1 1 2 2 2 3 1 1 1 1 2 2 1 1 4 1 1 2 2 1 4 4 4 1 2 3 1 3 3 3 3 2
## [297] 3 3 3 2 1 3 2 3 1 3 2 3 3 3 2 1 3 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 2 2 3 1
## [334] 1 1 1 1 3 3 1 3 2 3 3 2 3 1 1 1 1 1 2 3 1 1 1 1 1 3 3 1 1 1 1 1 2 3 3 2 2
## [371] 3 2 2 3 3 2 2 2 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 2 2 2 2 3 3 3 2 3 3
## [408] 3 3 3 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 2 2 3 3 3
## [445] 3 2 2 2 2 2 2 3 3 2 2 3 3 2 2 3 2 2 1 2 2 2 2 1 1 1 2 1 2 3 2 2 2 1 1 2 2 1
## [482] 1 1 1 1 1 2 4 1 4 3 2 2 3 4 1 3 4 4 2 3 4 1 1 1 1 1 3 3 4 4 4 4 1 1 4 1 2
## [519] 4 4 1 1 1 1 4 3 4 3 4 4 4 2 3 1 4 4 4 3 4 4 4 1 1 4 1 4 4 4 1 4 4 4 1 4 4
## [556] 1 4 1 4 4 4 4 4 4 2 4 4 4 4 1 1 4 3 4 2 4 3 3 2 4 4 4 4 1 4 3 1 4 4 4 1 1
## [593] 4 4 4 4 4 2 2 4 4 4 4 1 2 2 4 1 1 1 1 2 2 1 1 4 1 1 2 2 4 4 2 4 4 1 1 4 4
## [630] 4 1 2 1 1 1 3 3 1 4 1 1 3 4 4 4 4 1 4 4 4 4 4 4 1 2 4 3 3 1 1 1 1 2 1 4 4
## [667] 3 4 3 4 3 1 3 3 1 1 4 3 4 4 4 1 1 4 4 4 4 4 1 1 4 4 4 4 1 2 2 1 1 1 1 4 3
## [704] 1 4 3 1 3 3 2 2 1 1 1 1 1 2 2 1 1 1 1 1 3 2 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 14027256  9469312  5183165 14939972
##  (between_SS / total_SS =  93.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
clust3 <- kmeans(bikes2,3)
clust3
```

```
## K-means clustering with 3 clusters of sizes 149, 369, 212
##
## Cluster means:
##        Day    Month     Year RentedBikeCount    Snowfall
## 1 15.20134 7.040268 2018.000       2618.2349 0.000000000
## 2 15.08943 5.945799 2017.832        437.9295 0.149322493
## 3 17.18396 7.174528 2018.000       1733.8962 0.001415094
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 3 2 2 2 3 3
## [112] 3 3 2 2 2 3 3 3 3 2 2 3 3 3 2 3 2 2 3 3 2 3 3 2 2 3 3 3 2 3 2 2 2 3 3 3 3
## [149] 2 2 3 2 2 3 3 2 2 2 3 3 2 3 2 2 3 3 2 2 2 2 2 3 2 3 3 3 2 2 1 1 3 1 3 2 2
## [186] 1 1 2 1 1 2 2 1 1 2 2 1 2 2 1 1 1 1 1 2 2 1 2 3 1 3 2 2 2 3 1 3 1 2 2 2 3
## [223] 2 3 3 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 2 3 3 3 3 2 2 3 3 2 3
## [260] 3 2 2 3 3 3 3 2 2 2 2 3 3 3 3 2 2 3 3 1 1 1 2 2 1 1 1 1 1 2 2 1 2 2 2 2 2
## [297] 2 2 2 2 3 2 2 2 3 2 2 2 2 2 2 3 2 3 3 3 2 2 3 3 1 3 3 2 2 3 3 3 3 2 2 2 3
## [334] 3 3 3 3 2 2 3 2 3 2 2 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [408] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [445] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 2 2 3 3 3 2 3 2 2 3 3 2 3 3 2 2 3
## [482] 3 3 3 3 3 3 1 3 1 2 3 2 2 1 3 2 1 1 2 2 1 3 3 1 1 3 2 2 1 1 1 1 3 3 1 3 2
## [519] 1 1 3 3 1 3 1 2 1 2 1 1 1 2 2 1 1 1 1 2 1 1 1 3 1 1 3 1 1 1 1 1 1 1 3 1 1
## [556] 3 1 3 1 1 1 1 1 1 2 1 1 1 1 1 3 1 2 1 2 1 2 2 2 1 1 1 1 1 1 2 3 1 1 1 3 3
## [593] 1 1 1 1 1 3 3 1 1 1 1 1 2 3 1 3 3 3 3 2 3 3 1 1 3 1 3 3 1 1 2 1 1 3 3 1 1
## [630] 1 3 2 3 3 3 2 2 3 1 3 3 2 1 1 1 1 1 1 1 1 1 1 1 3 2 1 2 2 3 1 3 3 3 3 1 1
## [667] 2 1 2 1 2 1 2 2 3 3 1 2 1 1 1 3 3 1 1 1 1 1 3 3 1 1 1 1 3 2 2 3 3 3 1 1 2
## [704] 3 1 2 3 2 2 2 3 2 3 3 1 3 3 2 2 3 3 3 3 3 2 2 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
## [1] 20044550 29919366 18341768
##  (between_SS / total_SS =  89.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
clust2 <- kmeans(bikes2,2)
clust2
```

```
## K-means clustering with 2 clusters of sizes 331, 399
##
## Cluster means:
##        Day    Month     Year RentedBikeCount     Snowfall
## 1 16.52266 7.290030 2018.000       2179.4713 0.0009063444
## 2 15.05514 5.892231 2017.845        495.9749 0.1380952381
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2
## [112] 2 2 2 2 2 2 1 2 1 2 2 1 1 2 2 2 2 2 2 1 2 1 1 2 2 1 1 1 2 1 2 2 2 1 1 1 1
## [149] 2 2 1 2 2 1 1 2 2 2 1 1 2 1 2 2 1 1 2 2 2 2 2 1 2 1 1 1 2 2 1 1 1 1 1 2 2
## [186] 1 1 2 1 1 2 2 1 1 2 2 1 2 2 1 1 1 1 1 2 2 1 2 1 1 1 2 2 2 1 1 1 1 2 2 2 1
## [223] 2 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 2 1 1 1 1 2 2 1 1 2 1
## [260] 1 2 2 1 1 1 1 2 2 2 2 1 1 2 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 2 2 2 2 2
## [297] 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 1 2 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 2 2 2 1
## [334] 1 1 1 1 2 2 1 2 2 2 2 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [408] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [445] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 1 1 2 1 2 2 2 2 2 1 1 2 2 1
## [482] 1 1 1 1 1 2 1 1 1 2 2 2 2 1 1 2 1 1 2 2 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 2
## [519] 1 1 1 1 1 1 1 2 1 2 1 1 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [556] 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 2 1 2 2 2 1 1 1 1 1 1 2 1 1 1 1 1 1
## [593] 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 2 1 1 1 1 1 1
## [630] 1 1 2 1 1 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 1 1 1 1 2 1 1 1
## [667] 2 1 2 1 2 1 2 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2
## [704] 2 1 2 1 2 2 2 2 1 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 80841025 46574439
##  (between_SS / total_SS =  80.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

#I chose 2 clusters because had a highest cluster sum of squares, while the cluster mean >0

#d

```
clust5 <- kmeans(bikes2s,5)
clust5
```

```
## K-means clustering with 5 clusters of sizes 145, 62, 176, 154, 193
##
## Cluster means:
##          Day      Month       Year RentedBikeCount
## 1 -0.8246137 -1.0519563  0.3044458      -0.7876544
## 2  0.0317477  1.5865597 -3.2801577      -0.8613488
## 3  0.8515252  0.2197148  0.3044458       0.9809081
## 4  0.8697834 -0.5194636  0.3044458      -0.7512608
## 5 -0.8612147  0.4947903  0.3044458       0.5734086
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4
## [112] 4 4 4 4 4 4 4 4 4 4 1 1 5 1 1 1 1 1 1 5 1 5 5 1 1 3 3 3 4 3 4 4 4 4 3 3 3
## [149] 4 4 3 1 1 5 5 1 1 1 5 5 1 5 1 1 5 3 4 4 4 4 4 3 4 3 3 3 4 4 3 3 3 3 5 1 1
## [186] 5 5 1 5 5 1 1 5 5 1 1 3 4 4 3 3 3 3 3 4 4 3 4 3 3 3 4 1 1 5 5 5 5 5 1 1 5
## [223] 5 5 5 4 4 3 3 3 3 3 4 4 3 3 3 3 3 4 4 3 3 5 5 5 5 1 5 5 5 5 5 5 5 5 5 4 5
## [260] 3 4 4 3 3 3 3 4 4 4 4 3 3 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 4 3 4 4 4 4 4
## [297] 4 4 4 4 3 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 3 3 3 4 4 3 3 3 3 4 4 4 3
## [334] 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 4 3 3 3 3 3 4 4 3 3 3 3 3 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1
## [408] 1 1 1 1 1 1 1 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [445] 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 4 4 4 4 4 4 4 4 4 4
## [482] 4 3 3 3 4 1 5 5 5 1 1 1 1 5 5 1 5 5 1 1 3 3 3 3 3 3 3 4 4 3 3 3 3 3 3 3 5 1
## [519] 5 5 5 5 5 5 5 1 5 1 5 3 3 4 4 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5 5 5
## [556] 5 5 5 3 5 3 3 3 3 4 3 3 3 3 3 3 3 4 3 4 3 4 1 5 5 5 5 5 5 5 1 5 5 5 5 5 5
## [593] 3 3 3 3 3 4 4 3 3 3 3 3 4 4 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 3 3 3 3 3 3
## [630] 3 3 4 3 3 3 4 4 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 4 3 4 4 3 3 3 3 3 3 3 3
## [667] 4 3 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 3 3 3 3 3 3 3 3 3 3 3 3 4 3 3 3 5 5 5
## [704] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 3 3 3 3 3 4 4 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
## [1] 104.19797  68.50358 185.44553 185.27594 288.14813
##  (between_SS / total_SS =  71.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
clust10 <- kmeans(bikes2s,10)
clust10
```

```
## K-means clustering with 10 clusters of sizes 59, 73, 109, 63, 62, 64, 55, 79, 105, 61
##
## Cluster means:
##           Day      Month       Year RentedBikeCount
## 1  -0.9945660 -0.3243994  0.3044458       1.1051452
## 2   1.1257982 -0.4700918  0.3044458       0.8925944
## 3  -0.9521511 -1.1442897  0.3044458      -0.7966331
## 4   1.0091282  0.9286758  0.3044458       0.6205653
## 5   0.0317477  1.5865597 -3.2801577      -0.8613488
## 6   0.8323217  0.5540155  0.3044458      -0.7924549
## 7  -1.0237642  0.6696209  0.3044458      -0.8666002
## 8  -0.8080809  0.8161070  0.3044458       0.8085666
## 9   0.6852582 -1.1986357  0.3044458      -0.8200875
## 10  0.1583915  0.1706348  0.3044458       1.4982069
##
## Clustering vector:
##   [1]  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5
##  [26]  5  5  5  5  5  5  3  3  3  3  3  3  3  3  3  3  3  3  3  3  9  9  9  9  9
##  [51]  9  9  9  9  9  9  9  9  9  9  9  9  3  3  3  3  3  3  3  3  3  3  3  3  3
##  [76]  3  9  9  9  9  9  9  9  9  9  9  9  9  9  9  3  3  3  3  3  3  3  3  3  3
## [101]  3  3  3  3  9  9  9  9  9  9  9  9  9  9  9  9  9  2  2  2  9  3  1  1  3
## [126]  3  3  3  3  3  1  3  1  1  3  9  2  2  2  9  2  9  9  9  2  2  2  2  9  9
## [151]  2  3  3  1  1  3  3  3  1  1  3  1  3  3  1 10  9  9  9  9  9  2  9  2  2
## [176]  2  9  9  2  2  2  2  1  7  7  1  1  7  1  1  7  7  1  1  7  7 10  6  6 10
## [201] 10 10 10  2  6  6  2  6  2  2  2  6  7  7  1  1  1  1  7  7  7  1  7  8 10
## [226]  7  6 10 10 10 10 10  6  6  2  2  2  2  2  6  6  2  2  8  8  8  7  7  7  8
## [251]  8  8  8  7  7  8  8  6  8 10  6  6  4  4  4  4  6  6  6  6  4  4  4  4  7
## [276]  7  8  8  8  8  8  7  7  8  8  8  8  8  6  6 10  6  6  6  6  6  6  6  6  6
## [301]  4  6  6  6  8  7  7  7  7  7  7  8  7  8  8  8  7  7  8  8  4  4  4  6  6
## [326]  4  4  4  4  6  6  6  4  4  4  8  8  7  7  8  7  7  7  7  7  7  8  8  8  8
## [351]  8  6  6  4  4  4  4  4  6  6  4  4  4  4  4  5  5  5  5  5  5  5  5  5  5
## [376]  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  3  3  3  3
## [401]  3  3  3  3  3  3  3  3  3  3  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9
## [426]  9  9  3  3  3  3  3  3  3  3  3  3  3  3  3  3  9  9  9  9  9  9  9  9  9
## [451]  9  9  9  9  9  3  3  3  3  3  3  3  3  3  3  3  3  1  1  9  2  9  9  9  9
## [476]  9  2  2  9  9  2  2  2  2  2  2  3  1  1  1  3  3  3  3  1  1  3  1  1  3
## [501]  9 10  2  2  2  2  2  9  9  2  2  2  2  2  2  2  1  3  1  1  1  1  1  1  1
## [526]  3  1  3  1 10 10  9  9 10 10  2 10  9 10  2  2  2  2  2  2  2  2  1  1  1
## [551]  1  1  1  1  1  1  1  1 10 10 10 10 10 10  6 10 10 10 10  2  2 10  6  2  6
## [576]  2  6  7  7  1  1  1  1  1  1  7  8 10 10 10  8  8 10 10 10 10 10  6  6 10
## [601]  2  2  2  2  6  6  2  2  8  8  8  7  7  8  8  8  8  8  8  8 10 10  6 10 10
## [626] 10  4 10 10 10  4  6  4  4  4  6  6  4  2  8  8  7  8  8  8  8  8  8 10 10
## [651] 10 10 10  8  6 10  6  6  4  4  4  4  4  4  4 10  6  4  6  8  7  8  7  7  8
## [676]  8  8  7  8  8  8  8  8 10 10 10 10 10  4  4  4  4  4  4  4  6  6  4  4  4
## [701]  8  8  7  8  8  7  8  7  7  8  7  8  8  8  8  8  6  6  4  4  4  4  4  6  6
## [726]  4  4  4  4  4
##
## Within cluster sum of squares by cluster:
##  [1] 30.75580 44.36339 54.49731 26.31513 68.50358 39.31211 33.26050 47.71110
##  [9] 53.74947 34.85223
##  (between_SS / total_SS =  85.1 %)
##
```

```
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
clust100 <- kmeans(bikes2s,100)
clust100
```

```
## K-means clustering with 100 clusters of sizes 5, 9, 11, 7, 13, 10, 4, 8, 9, 16, 3, 6, 4, 7, 1
0, 4, 2, 8, 6, 6, 11, 5, 7, 2, 5, 8, 8, 3, 5, 14, 5, 2, 8, 9, 2, 4, 5, 9, 3, 7, 6, 8, 7, 6, 6,
7, 6, 11, 6, 10, 4, 10, 6, 7, 13, 4, 10, 8, 9, 7, 5, 11, 10, 4, 8, 8, 7, 8, 7, 10, 9, 11, 8, 8,
8, 6, 7, 9, 13, 3, 5, 8, 8, 9, 9, 3, 5, 11, 6, 11, 9, 6, 11, 10, 6, 7, 7, 7, 5, 6
##
## Cluster means:
##             Day        Month       Year RentedBikeCount
## 1   -0.64989403  0.948918350  0.3044458     1.359262530
## 2   -0.77612398 -1.472830685  0.3044458    -0.756259494
## 3   -0.28841736  1.191327465  0.3044458     0.823922304
## 4    1.47617898 -0.856352017  0.3044458     0.652479271
## 5   -1.48009870 -1.178039221  0.3044458    -1.128434411
## 6    0.58842177 -1.340793836  0.3044458    -0.562422610
## 7   -0.47948360  0.717048761  0.3044458     2.103626047
## 8   -0.90550968 -0.297380688  0.3044458     0.952530565
## 9    0.82699638  0.427211776  0.3044458     0.101878104
## 10   1.33822767  1.586559718 -3.2801577    -0.989617901
## 11  -1.63448764  0.137374790  0.3044458    -1.293766435
## 12   0.58084798 -1.021973152  0.3044458    -0.097093139
## 13   0.54297899 -0.152462195  0.3044458     2.317330412
## 14  -1.57497923 -1.187594286  0.3044458    -0.651471812
## 15  -0.19546621  1.586559718 -3.2801577    -0.787705962
## 16  -0.13866273 -0.007543702  0.3044458    -0.476567348
## 17   1.67904854 -0.587217673  0.3044458     1.238136711
## 18  -0.69249664  0.137374790  0.3044458    -0.575282223
## 19   0.14535465 -0.780442330  0.3044458     0.565274779
## 20   0.69445493 -0.780442330  0.3044458     0.441480990
## 21   0.40355228  0.374514142  0.3044458     0.716718435
## 22   1.21326002  1.238755335  0.3044458     0.412453480
## 23   0.24273204 -0.359488613  0.3044458    -0.712149063
## 24   1.50863811  0.572130269  0.3044458     1.852303311
## 25   0.39528995 -0.558233975  0.3044458     1.125334956
## 26  -1.50194619  1.224263486  0.3044458     0.929319230
## 27  -1.26053141 -0.840825036  0.3044458     0.011937885
## 28  -0.08185926 -0.538911509  0.3044458     1.684399091
## 29   1.50863811 -0.326364386  0.3044458     2.049733061
## 30   0.48617552  1.586559718 -3.2801577    -0.963471799
## 31  -0.26363039 -1.021973152  0.3044458    -0.009299153
## 32  -0.99071489  0.282293283  0.3044458    -1.104340594
## 33  -0.40847925 -0.732136166  0.3044458     0.622902922
## 34  -1.55874967  0.523824104  0.3044458     0.543812446
## 35  -0.02505578  0.717048761  0.3044458     1.912599424
## 36  -0.99071489  1.296722733  0.3044458     0.011137494
## 37   1.05421029  0.137374790  0.3044458     0.617140108
## 38  -1.43251972 -0.345686852  0.3044458     1.161239898
## 39   0.59978247  0.427211776  0.3044458     1.496929761
## 40   0.06420683  0.551427627  0.3044458    -0.636988548
## 41   0.16428914  0.233987119  0.3044458     1.504755805
## 42  -0.40847925  0.608359892  0.3044458     1.134752889
## 43   0.89191464 -0.193867479  0.3044458    -0.618693900
## 44   0.90273435 -0.490605345  0.3044458    -1.278825805
## 45   1.22462072  0.620436433  0.3044458     0.965292353
```

```
## 46  -0.48759838 -0.566515032  0.3044458    -1.334929394
## 47   0.42937204  1.200110404  0.3044458     0.639266469
## 48  -1.36251947  1.059583381  0.3044458    -1.258355202
## 49  -0.08185926 -0.490605345  0.3044458    -1.028925986
## 50   1.33822767  0.166358489  0.3044458    -0.722287348
## 51   1.11101377 -0.732136166  0.3044458     1.397681292
## 52  -0.68397612  0.311276982  0.3044458     0.402955509
## 53   1.03527580 -0.538911509  0.3044458     0.784226150
## 54   0.97306246 -1.228999570  0.3044458    -1.011266569
## 55   1.44746513 -1.579351970  0.3044458    -1.075649659
## 56  -0.76350099 -0.224921442  0.3044458     2.141511215
## 57   0.04310839 -1.311810137  0.3044458    -1.198786717
## 58  -1.45934358 -0.261151065  0.3044458    -0.555939443
## 59  -0.47317210 -0.410095071  0.3044458     1.255152428
## 60   1.42749028 -1.021973152  0.3044458     0.124221291
## 61  -0.30907317  0.021439996  0.3044458     1.790192979
## 62  -1.41415900  1.586559718 -3.2801577    -0.647826742
## 63   0.14535465 -1.601647123  0.3044458    -0.738615322
## 64  -1.30313402  0.789508008  0.3044458     1.947549826
## 65  -0.87710794  1.224263486  0.3044458    -1.318845349
## 66   1.43763376 -0.333610311  0.3044458     0.936789545
## 67  -1.38022445 -0.525109748  0.3044458     0.738159208
## 68  -0.18126534  0.246063660  0.3044458     0.520319491
## 69   0.97306246  0.013158939  0.3044458     1.121127187
## 70  -1.22928950  0.659081364  0.3044458    -0.504474310
## 71  -1.06645286  0.813661090  0.3044458     0.996655818
## 72  -1.06301023 -1.443554222  0.3044458    -0.981759518
## 73   0.41517117 -0.913284282  0.3044458    -0.903976077
## 74  -1.38833924 -1.601647123  0.3044458    -0.652119747
## 75   1.59384332  0.318522906  0.3044458     0.362375691
## 76   0.39150305  0.717048761  0.3044458    -1.268153927
## 77  -1.03940359 -0.649325599  0.3044458    -1.092220389
## 78   0.14535465  0.974681638  0.3044458     1.228235579
## 79  -0.35276815 -1.467876206  0.3044458    -0.893406813
## 80   1.54650709  0.137374790  0.3044458     1.011715024
## 81   0.89516055 -0.384331784  0.3044458     1.894137074
## 82   0.71338942  1.224263486  0.3044458     0.345967678
## 83   0.85539812  0.970656124  0.3044458     1.163433562
## 84  -0.82661596 -0.957564933  0.3044458    -0.311242165
## 85  -0.15759723  1.232314514  0.3044458    -0.668038633
## 86  -0.04399027 -0.925360823  0.3044458    -0.520055252
## 87   0.53161830 -0.094494798  0.3044458     1.204947169
## 88   0.77535685 -1.601647123  0.3044458    -1.040648883
## 89   1.62224506  1.006885747  0.3044458     0.760748018
## 90  -0.79448470  1.586559718 -3.2801577    -0.825270974
## 91   0.83961937  0.974681638  0.3044458    -0.886693563
## 92  -0.17653172  1.103498076  0.3044458     0.434722133
## 93   1.28142420 -1.101019602  0.3044458    -0.538391480
## 94  -1.11568254  0.224325886  0.3044458     1.025019299
## 95   1.50863811  1.296722733  0.3044458     0.292252557
## 96   1.26519463  0.965480463  0.3044458    -0.568993438
## 97   1.47617898  0.675643478  0.3044458    -1.309316886
```

```
## 98  -0.63366447 -1.187594286  0.3044458    -1.149543621
## 99   0.05446909 -0.210429592  0.3044458     1.146678713
## 100 -1.35047025 -0.007543702  0.3044458     2.024120553
##
## Clustering vector:
##   [1]  62  62  62  62  62  62  90  90  90  90  90  15  15  15  15  15  30  30
##  [19]  30  30  30  30  30  10  10  10  10  10  10  10  10   5  74  74  74  74
##  [37]  72  72   2   2   2  79  79  79  57  63  63  63  63  63  88  88   6  88
##  [55]  88  88  55  55  55  55  55  55  14  14   5   5  72  72  72   2   2  98
##  [73]  98  79  79  79  57  57  57  57   6   6   6   6  54  54  54  93  93  93
##  [91]   5  14   5   5   5  84  84  98  84  98  98  31  31  31  57  31  73  73
## [109]  73  12  12  12  12  54  54  93  60  60  60  60  93   5  27  67  27  77
## [127]  27  77  77  84  33  46  33  33  46  49  19  19  19  73  20  73  73  44
## [145]  20  53  53   4  93  93   4  58  58  67  67  58  77  77   8   8  46  33
## [163]  46  49  33  99  49  49  23  23  23  20  43  53  53  53  43  43  66  66
## [181]  66  17  38  58  58  38  38  58   8   8  18  18  59  59  16  49  99  16
## [199]  23  99  87  87  87  87  43  43  69  44  66  66  66  50  11  11  94  94
## [217]  94  94  18  18  18  52  18  68  68  16  16  68  21  21  21  21  43  43
## [235]  37  37  37  37  37  50  50  75  75  34  34  34  70  32  70  52  52  52
## [253]  52  18  18  68  68  40  68  21  40  40  21   9  21   9  50  50  50  50
## [271]  75  75  75  75  70  70  34  71  71  71  71  70  70  42  42  42  42  42
## [289]  40  40  78  76  76  76  76  91  91  91  91  96  45  97  96  97  26  48
## [307]  70  48  48  48  70  71  65   3   3   3  85  85   3  78  78  78  47  91
## [325]  91  83  82  83  83  96  96  97  89  89  89  26  26  48  48  26  65  36
## [343]  65  65  85  85  92   3   3  92  92  85  85  47  47  82  82  82  91  91
## [361]  22  22  95  95  95  62  62  62  62  62  90  90  90  90  90  90  15  15
## [379]  15  15  15  30  30  30  30  30  30  30  10  10  10  10  10  10  10  10
## [397]   5  74  74  74  74  72  72  72   2   2  79  79  79  79  63  63  63  63
## [415]  63  88  88  88  88  88  88  55  55  55  55  55  55  14  14   5   5  72
## [433]  72  72   2   2  98  98  79  79  79  57  57  57  57   6   6   6   6   6
## [451]  54  54  93  93  55   5  14  14   5  27  27  27  84  84  84  84  31  33
## [469]  33  86  19  86  73  12  12  73  20  20  93  93  60  60   4   4   4  60
## [487]  27  38  67  38  77  27  84  77  59  33  46  59  59  86  49  28  19  19
## [505]  25  25  20  44  44  51  51  51  51   4   4  17  67  58  38  38  67  67
## [523]   8   8  56  46  59  46  59  28  28  23  23  25  25  25  81  44  81  81
## [541]  81  53  66  29  66  29  29 100  38  38 100 100   8  56  56   8  59  52
## [559]  56  59  61  61  99  99  23  13  13  13  13  69  69  81  44  29  50  29
## [577]  50  11  58 100 100  94 100  94  94  32  52  61  61  61  68  68  41  41
## [595]  41  41  87   9   9  69  69  69  69  80  50  75  80  80  34  34  34  70
## [613]  70  94  94  94  52  42  52  52  42  42  40  41  41  21  21  39  39  39
## [631]   9   9  45   9  45  97  97  75  24  34  34  48  64  64  71  64  71   1
## [649]   7   7   7   7  35  92  40  35  76  76  21  83  21   9   9  45  45  24
## [667]  97  45  97  64  48  26  48  48  71  71   1  65   1   1   1  92  92  78
## [685]  78  78  78  78  47  82  83  83  83  83  22  96  96  89  89  89  26  26
## [703]  48  36  26  65  36  65  65  36  85   3   3   3   3   3  85  85  47  47
## [721]  82  82  82  91  96  22  22  95  95  95
##
## Within cluster sum of squares by cluster:
##   [1] 0.33675413 0.33822595 0.74457613 0.27349226 1.30308616 0.32548693
##   [7] 0.07867657 0.45255928 1.08167225 1.97199148 0.01415986 0.15529761
##  [13] 0.08999685 0.38083881 0.57327712 0.22624777 0.09214895 0.75468030
##  [19] 0.19679974 0.57768649 0.90087006 0.18615901 0.41665361 0.20283961
```

```
##  [25] 0.18469489 0.65768770 0.67010572 0.11811163 0.24984810 1.12639053
##  [31] 0.24229415 0.16151898 0.73813216 0.70836439 0.17390510 0.25724888
##  [37] 0.14250339 0.85208030 0.07972357 0.55004067 0.27353208 0.42061431
##  [43] 0.94386598 0.64321746 0.45536130 0.29596985 0.24866290 0.76737072
##  [49] 0.43094171 1.39706350 0.20089654 1.24062817 0.22581202 0.24915102
##  [55] 0.89760796 0.29086166 0.45937122 0.89083781 0.90403463 0.33335867
##  [61] 0.24334140 1.35767334 0.32355957 0.38350399 0.30262355 0.40116073
##  [67] 0.48790014 0.67454434 0.32373487 1.74927352 0.60184990 0.49091435
##  [73] 0.75879777 0.13347234 0.75289581 0.17393488 0.57479685 0.69100491
##  [79] 0.80433756 0.14989636 0.33517548 0.26162423 0.76199187 0.76338955
##  [85] 1.55517497 0.16158021 0.15342577 0.54495494 0.10255143 1.27100134
##  [91] 1.34916694 0.43466670 1.30939663 0.95188663 0.09107142 0.58231368
##  [97] 0.32212886 0.28617229 0.27965315 0.43555583
##  (between_SS / total_SS =  98.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

#I prefer unstandardized because it will give an exact number of clusters, while standardized seems to approach infinite

#e For hierarchical, they did not match at all, there were a lot more clusters than 4, this is because many of the observations are statistically different from each other. For non-hierarchical, they also did not match, the optimal amount was actually less than 4, this is because it specifically tried to find what number of groups are statistically similar while not having the cluster mean to be zero