311-2 Homework 3

Tina Zhang

2019/5/10

```
knitr::opts_chunk$set(echo = TRUE)
packages <- c("readr","proxy","dplyr","tidytext","ggplot2","SnowballC","viridis",
              "fields","tidyr","mixtools","tm")
load.packages <- function(x) {
  if (!require(x, character.only = TRUE)) {
    install.packages(x, dependencies = TRUE)
    library(x, character.only = TRUE)
  }
}
lapply(packages, load.packages)
setwd("C:/Users/tzwhi/Desktop/Northwestern/311-2/R")
```

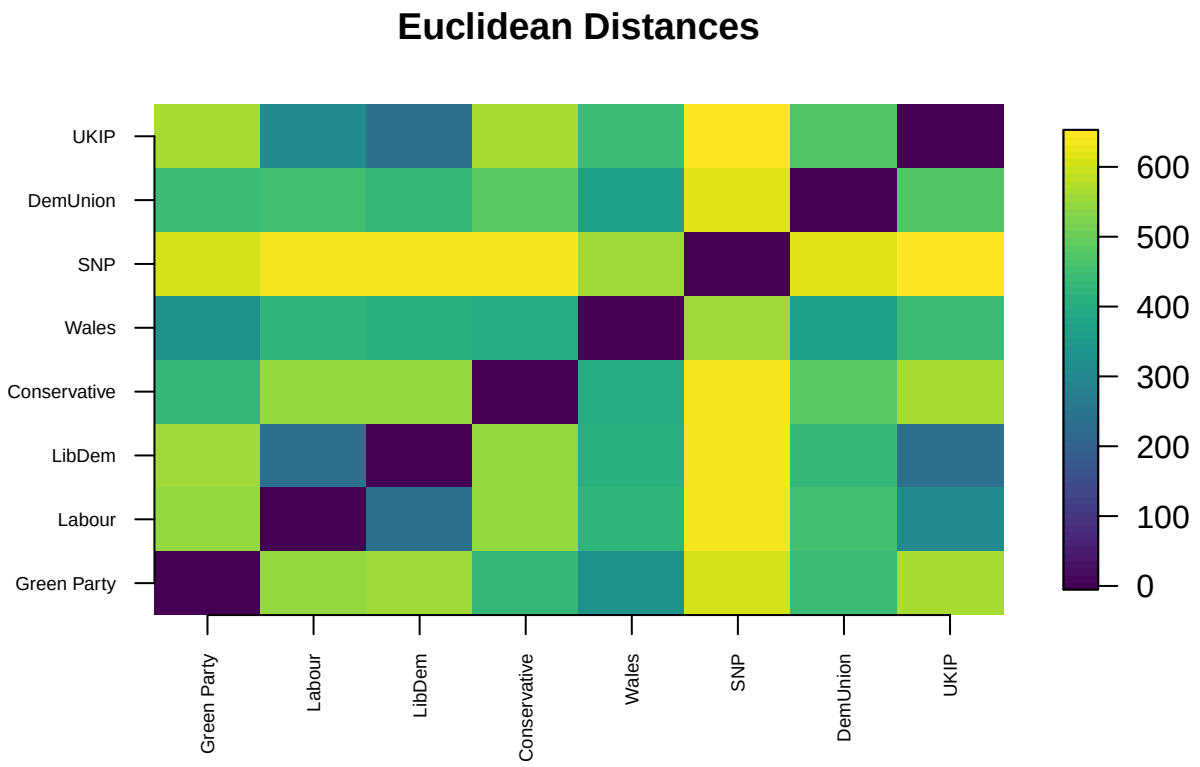
1. Distance/Similarity

```
#1
mani <- read_csv("manifestos.csv")
tokenmani <- unnest_tokens(mani, word, text) %>%
  anti_join(stop_words) %>% mutate(word = wordStem(word))
dtm <- tokenmani %>% count(doc_id, word) %>%
  cast_dtm(doc_id, word, n) %>% removeSparseTerms(0.99) %>% as.matrix()

#2
euc <- dist(dtm) %>% as.matrix()
print(euc)
```

```
##              Conservative DemUnion Green Party   Labour   LibDem      SNP
## Conservative       0.0000 540.1870    552.5305 431.7163 331.6580 606.6655
## DemUnion         540.1870   0.0000    233.6450 549.1894 417.0552 638.5319
## Green Party      552.5305 233.6450      0.0000 544.5852 407.7389 638.7699
## Labour           431.7163 549.1894    544.5852   0.0000 400.9913 638.7347
## LibDem           331.6580 417.0552    407.7389 400.9913   0.0000 556.4459
## SNP              606.6655 638.5319    638.7699 638.7347 556.4459   0.0000
## UKIP             438.5316 453.6937    434.9322 485.9105 375.1573 621.5714
## Wales            565.3645 312.4532    233.5573 563.4625 438.7756 647.8271
##                    UKIP    Wales
## Conservative 438.5316 565.3645
## DemUnion     453.6937 312.4532
## Green Party  434.9322 233.5573
## Labour       485.9105 563.4625
## LibDem       375.1573 438.7756
## SNP          621.5714 647.8271
## UKIP           0.0000 468.4816
## Wales        468.4816   0.0000
```

```
image.plot(1:ncol(euc), 1:ncol(euc), euc, axes = F, xlab = "", ylab = "",
           main = "Euclidean Distances", col = viridis(64))
axis(1, 1:ncol(euc), mani$doc_id, cex.axis = 0.6, las=3)
axis(2, 1:ncol(euc), mani$doc_id, cex.axis = 0.6, las=1)
```



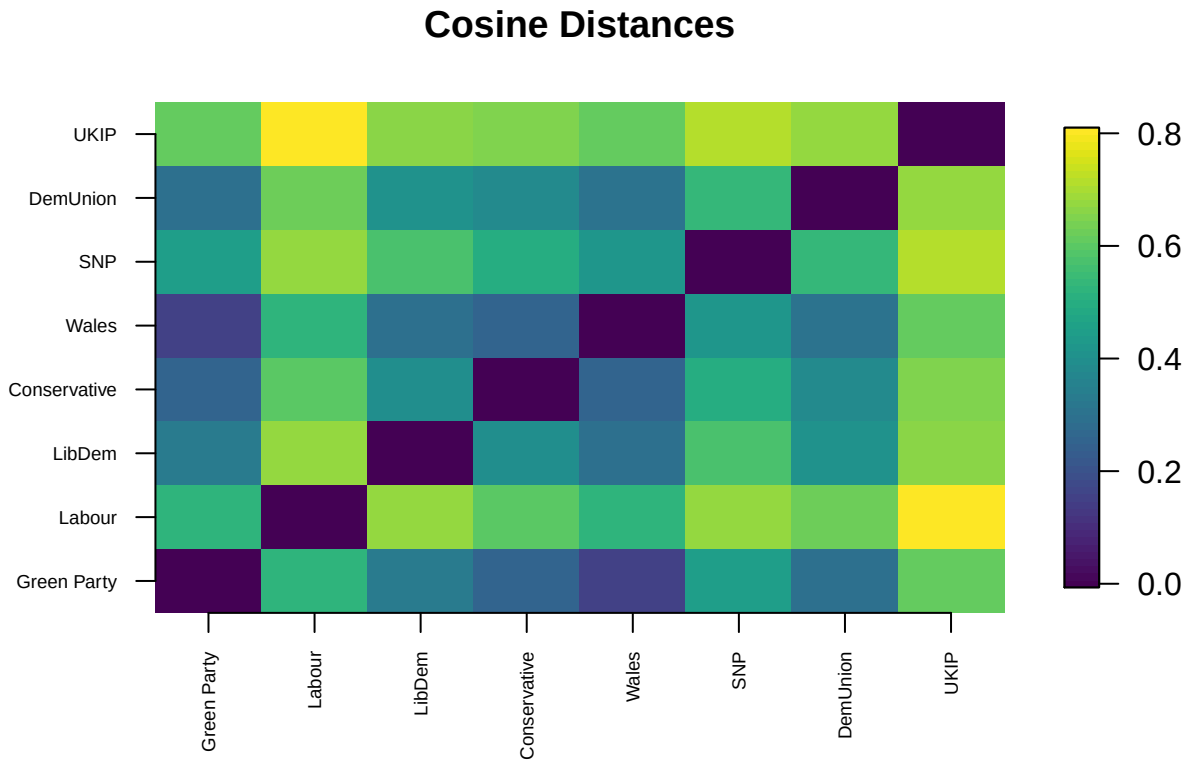
By Euclidean distance, the top 5 closest pairings, in order of most to least close, are Green Party-Wales, Green Party-DemUnion, DemUnion-Wales, Conservative-LibDem, and UKIP-LibDem.

```
#3
cos <- dist(dtm, method = "cosine") %>% as.matrix()
print(cos)
```

```
##              Conservative  DemUnion Green Party    Labour    LibDem
## Conservative    0.0000000 0.5204065   0.3351602 0.2536549 0.1556048
## DemUnion        0.5204065 0.0000000   0.6764215 0.6024194 0.5289530
## Green Party     0.3351602 0.6764215   0.0000000 0.3959646 0.2881437
## Labour          0.2536549 0.6024194   0.3959646 0.0000000 0.2563073
## LibDem          0.1556048 0.5289530   0.2881437 0.2563073 0.0000000
## SNP             0.4403148 0.6749508   0.5732773 0.4980957 0.4217511
## UKIP            0.2955111 0.6195540   0.4070560 0.3841075 0.3062600
## Wales           0.6125894 0.8035864   0.6693755 0.6561363 0.6085862
##                      SNP      UKIP     Wales
## Conservative 0.4403148 0.2955111 0.6125894
## DemUnion     0.6749508 0.6195540 0.8035864
## Green Party  0.5732773 0.4070560 0.6693755
## Labour       0.4980957 0.3841075 0.6561363
```

```
## LibDem       0.4217511 0.3062600 0.6085862
## SNP          0.0000000 0.5349073 0.7107423
## UKIP         0.5349073 0.0000000 0.6760242
## Wales        0.7107423 0.6760242 0.0000000
```

```
image.plot(1:ncol(cos), 1:ncol(cos), cos, axes = F, xlab="", ylab="",
           main = "Cosine Distances", col = viridis(64))
axis(1, 1:ncol(cos), mani$doc_id, cex.axis = 0.6, las=3)
axis(2, 1:ncol(cos), mani$doc_id, cex.axis = 0.6, las=1)
```



By cosine distance, the top 5 closest pairings, in order of most to least close, are Conservative-LibDem, Conservative-Labour, Labour-LibDem, Green Party-LibDem, and Conservative-UKIP.

```
rowSums(dtm)
```

```
## Conservative     DemUnion  Green Party       Labour       LibDem
##        12955         3657         1599        11672        10620
##          SNP         UKIP        Wales
##        10092        12145         2870
```

Overall, using cosine distance seems most appropriate because the different parties' manifestos have different lengths. For example, by word counts in the Term Document Matrix, the Green Party's manifesto has only 1599 words, while Conservative's has 12955 words, nearly 10 times more. Note that the length of manifestos is unlikely to be relevant to the parties' beliefs/ideals. Cosine distance looks at the angle between vectors and the proportion of word count frequencies among terms, and it doesn't consider the magnitude of the word counts, so it corrects for the unequal lengths of different manifestos. Euclidean distance is inappropriate

when the texts differ significantly in length because it will consider manifesto 1 to be more related to a term X than manifesto 2 if the word count of X is higher in 1 than in 2, which is erroneous because 1 may simply be longer than 2.

2. Clustering

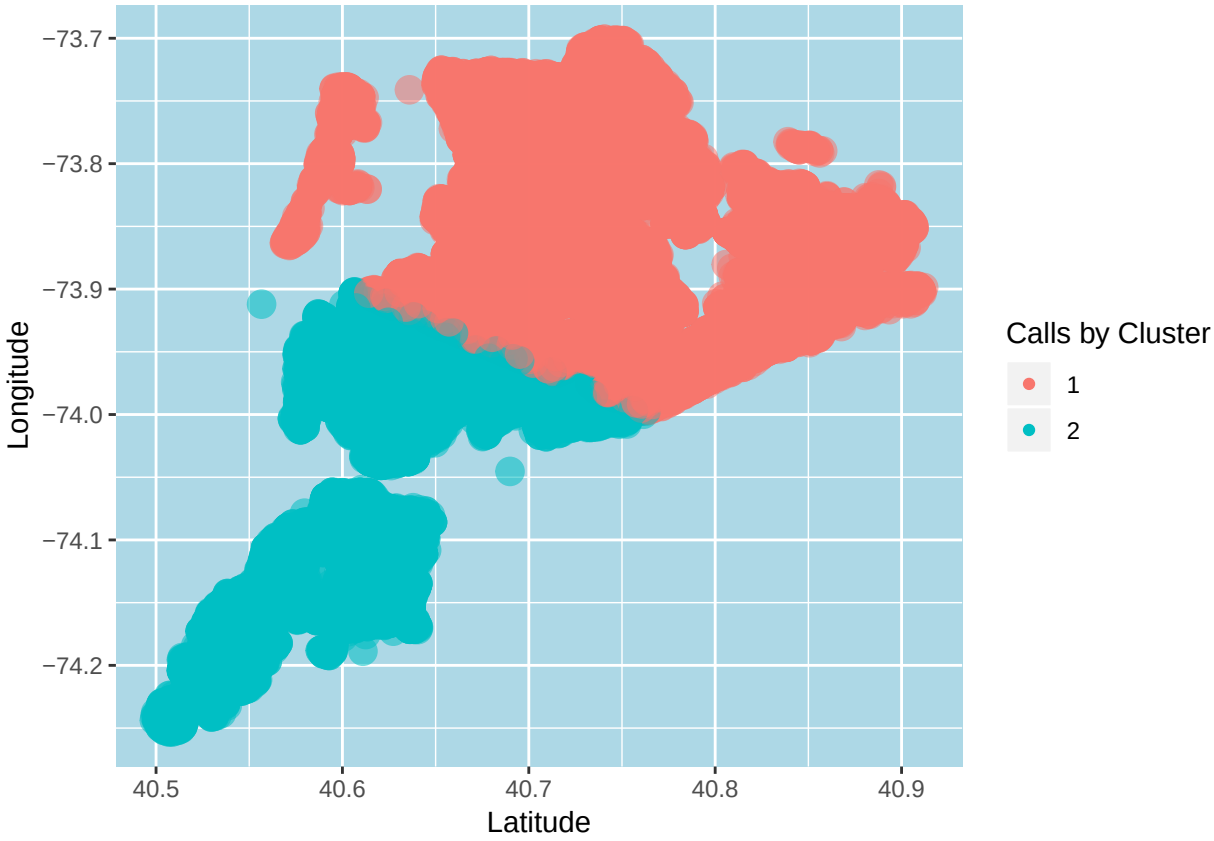
```
#1
sani <- read.csv("311_sanitation_requests_2019.csv") %>%
  drop_na(Latitude,Longitude)
set.seed(25)
k2 <- kmeans(sani[,c("Latitude","Longitude")], 2)
sani$cluster <- k2$cluster

#2
table(Borough = sani$Borough, Cluster = sani$cluster)
```

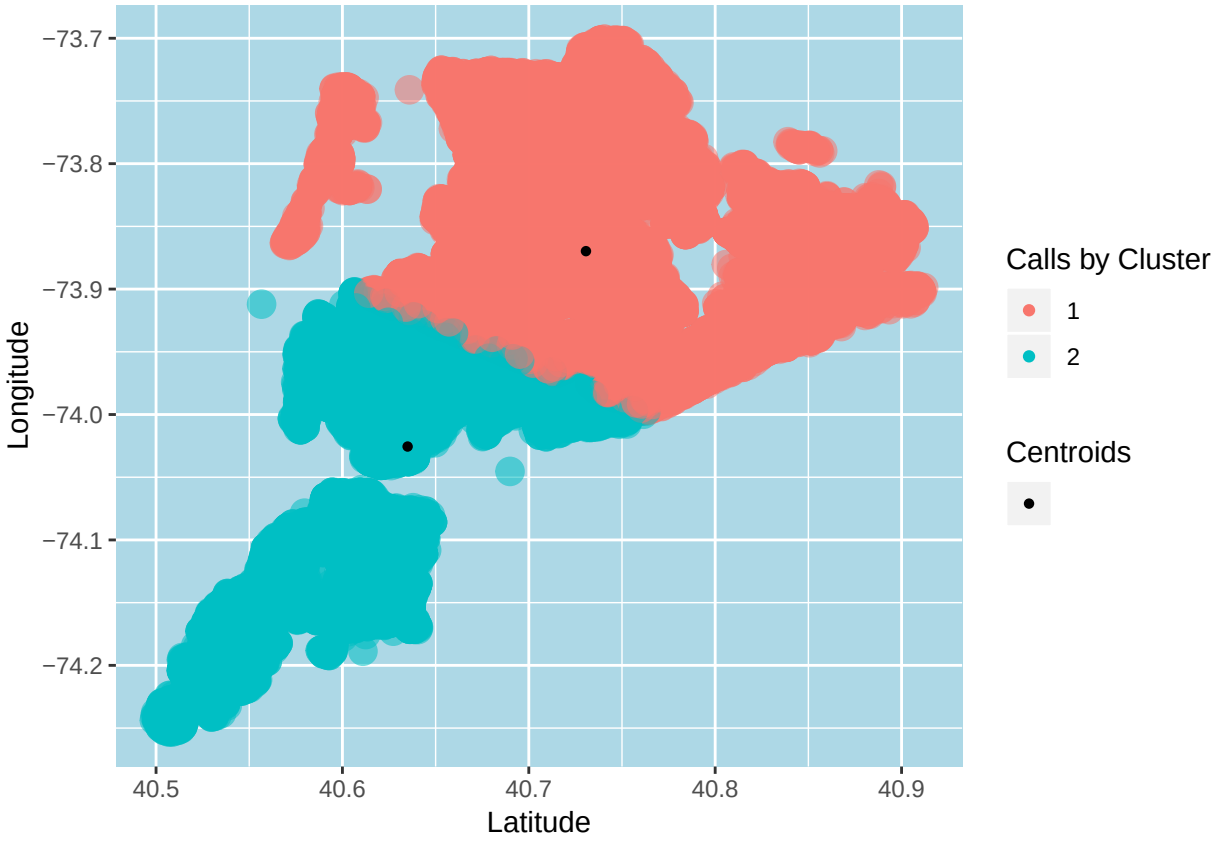
```
##                Cluster
## Borough             1     2
##   BRONX          5045     0
##   BROOKLYN      11248 22234
##   MANHATTAN      4868  3669
##   QUEENS        27448     1
##   STATEN ISLAND     0 10049
```

As shown by the table above, the clusters mostly match the political boundaries, and the only boroughs that were split up in a significant way were Brooklyn and Manhattan.

```
#3
nocentroids <- ggplot()+
  geom_point(data = sani,aes(Latitude,Longitude,
                      color=as.factor(cluster),size = 0.5,alpha = 0.1)) +
  labs(x = "Latitude", y = "Longitude", color = "Calls by Cluster") +
  guides(alpha = FALSE, size = FALSE) +
  theme(panel.background = element_rect("lightblue"))
print(nocentroids)
```




```
#4
centroids <- as.data.frame(k2$centers) %>% mutate (name = "")
withcentroids <- nocentroids +
  geom_point(data = centroids, aes(Latitude,Longitude, shape = name)) +
  labs(shape = "Centroids")
print(withcentroids)
```

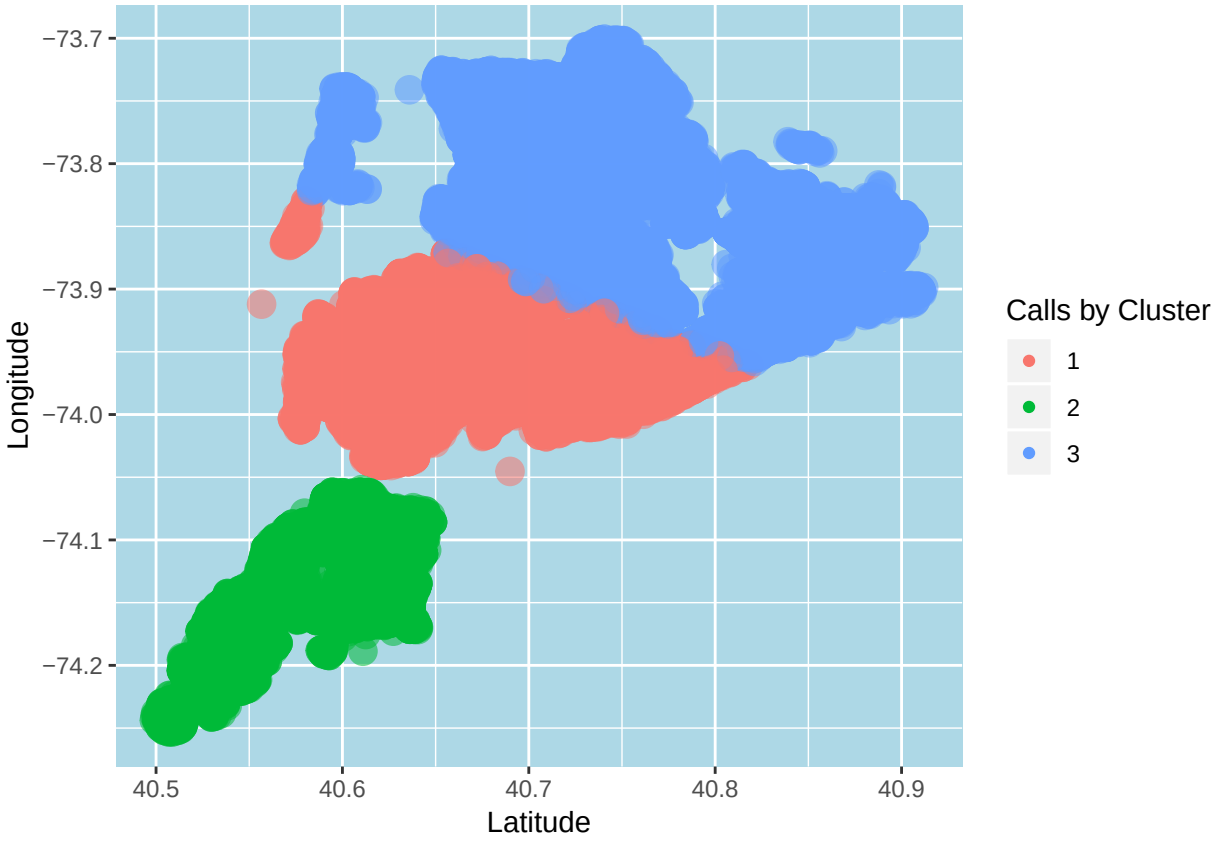



```
#5
set.seed(25)
k3 <- kmeans(sani[,c("Latitude","Longitude")], 3)
sani$cluster3 <- k3$cluster
table(Borough = sani$Borough, Cluster = sani$cluster3)
```

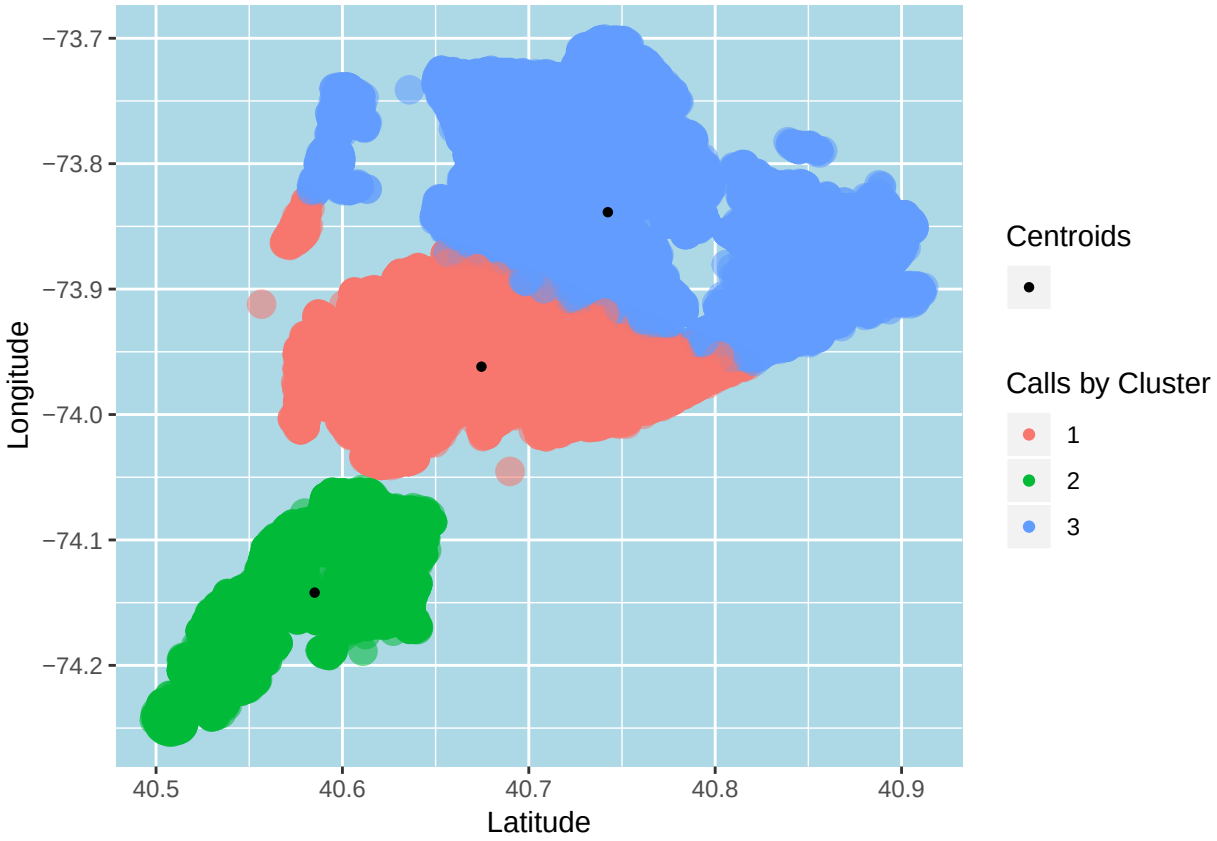
```
##                Cluster
## Borough             1     2     3
##   BRONX             0     0  5045
##   BROOKLYN      32668     0   814
##   MANHATTAN      6968     0  1569
##   QUEENS         2422     0 25027
##   STATEN ISLAND     0 10049     0
```

In the table above, boroughs Brooklyn, Manhattan, and Queens were all split up into two clusters, while Bronx and Staten Island weren't split up. This is actually an increase in the number of split boroughs compared to the results from using 2 clusters.

```
nocentroids3 <- ggplot()+
  geom_point(data = sani,aes(Latitude,Longitude,
                        color=as.factor(cluster3),size = 0.5,alpha = 0.1)) +
  labs(x = "Latitude", y = "Longitude", color = "Calls by Cluster") +
  guides(alpha = FALSE, size = FALSE) +
  theme(panel.background = element_rect("lightblue"))
print(nocentroids3)
```

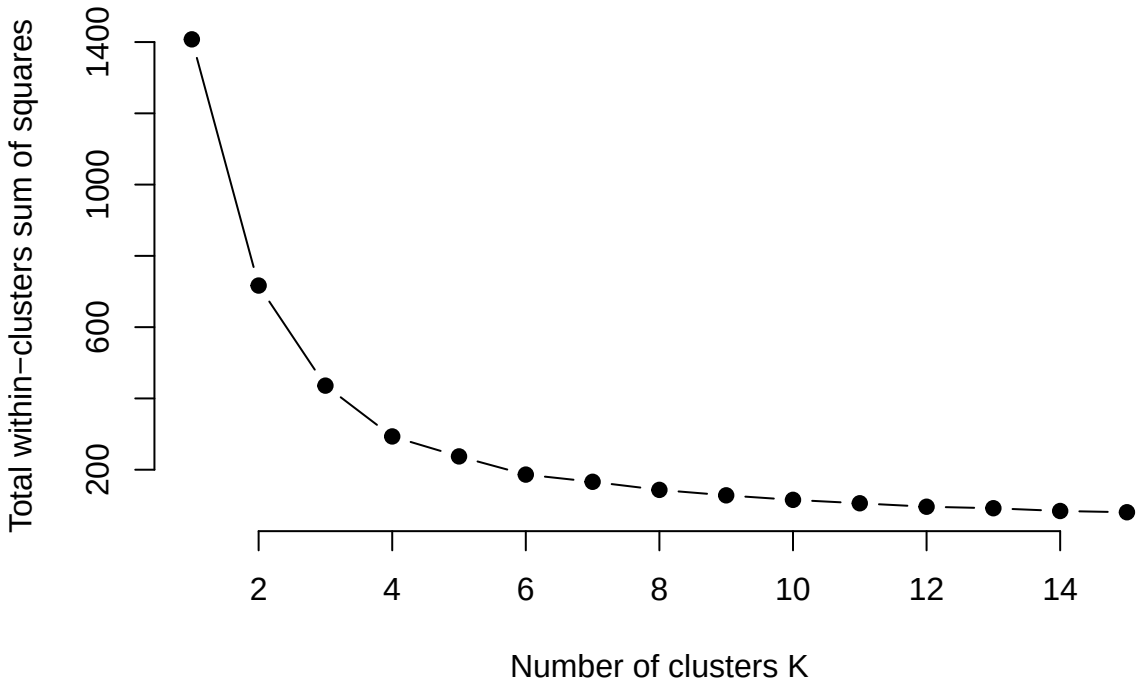



```
centroids3 <- as.data.frame(k3$centers) %>% mutate (name = "")
withcentroids <- nocentroids3 +
  geom_point(data = centroids3, aes(Latitude,Longitude, shape = name)) +
  labs(shape = "Centroids")
print(withcentroids)
```



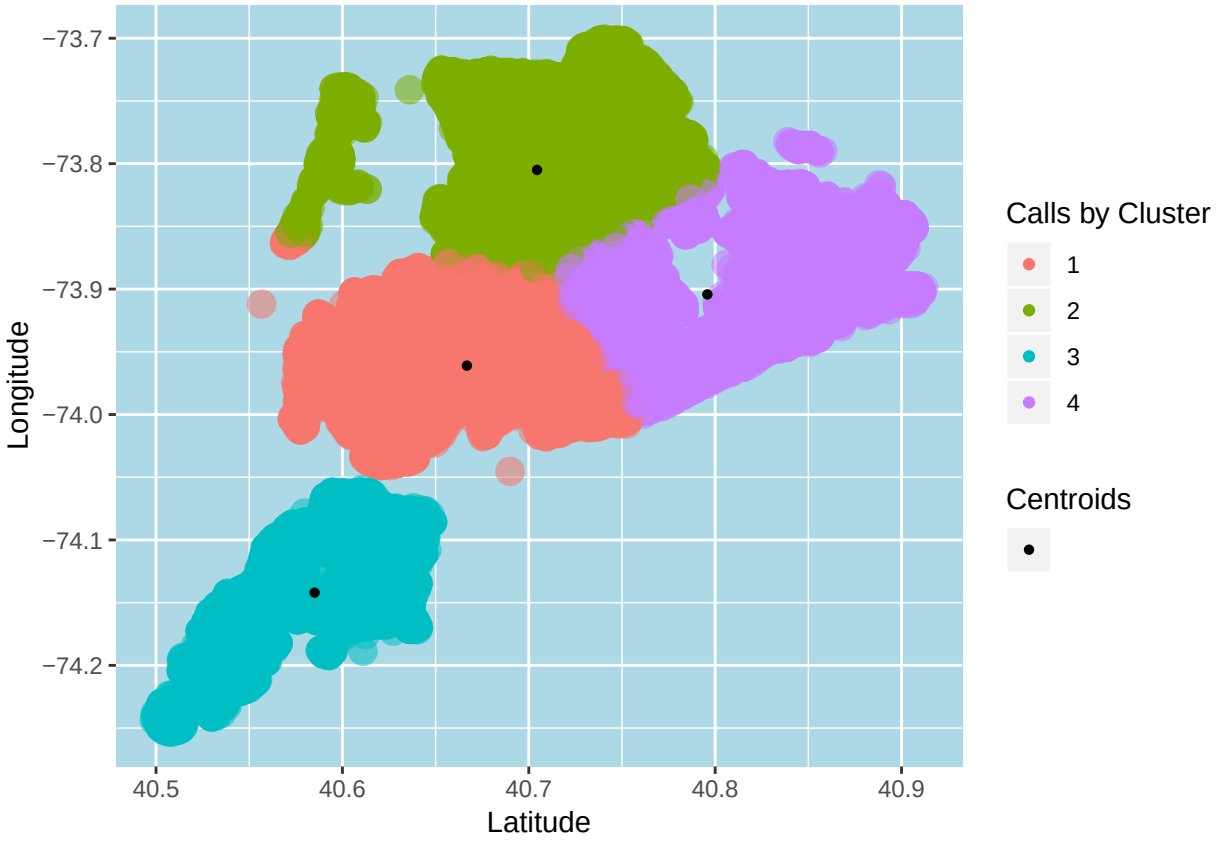

```
#6
wcss <- c()
for (i in 1:15){
  set.seed(57)
  k <- kmeans(sani[,c("Latitude","Longitude")], centers = i, iter.max = 30)
  sani[[paste0("k",i)]] <- k$cluster
  wcss[i] <- k$tot.withinss
}

#7
plot(1:15, wcss, type="b", pch=19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")
```



Total within-clusters sum of squares is a measure of the total amount of variation among data points within the same cluster, so we want to minimize this number. In the plot above, the total within-clusters sum of squares still drops noticeably after the inclusion of the 4th cluster(facility) but doesn't change substantially with any more increases in the number of facilities, so building more facilities probably aren't worth the cost. Thus, I recommend building 4 facilities.

```
#8
set.seed(57)
k4 <- kmeans(sani[,c("Latitude","Longitude")], centers = 4, iter.max = 30)
centroids4 <- as.data.frame(k4$centers) %>% mutate (name = "")
ggplot()+ geom_point(data = sani,aes(Latitude,Longitude,
                     color=as.factor(k4),size = 0.5,alpha = 0.1)) +
  geom_point(data = centroids4, aes(Latitude,Longitude, shape = name)) +
  labs(x = "Latitude", y = "Longitude", color = "Calls by Cluster",
       shape = "Centroids") + guides(alpha = FALSE, size = FALSE) +
  theme(panel.background = element_rect("lightblue"))
```



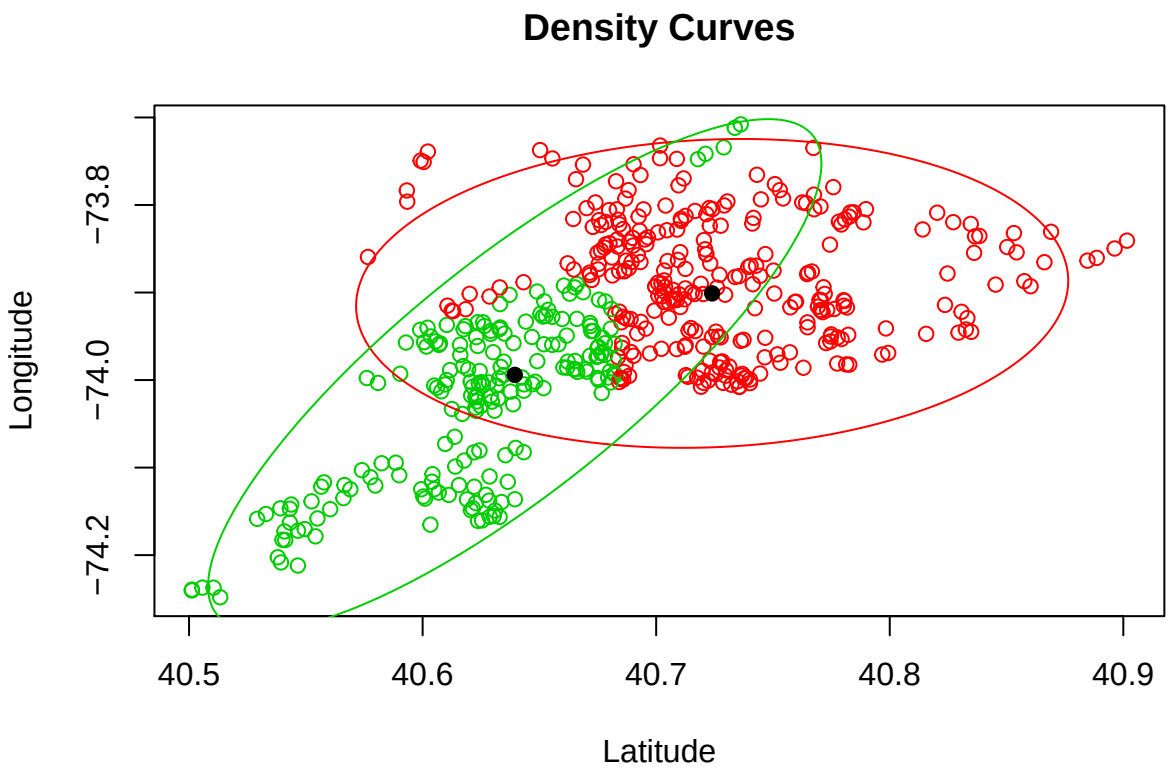
9. Euclidean distance measures the length of the straight line segment connecting two points, but in real life, people can rarely travel between two points via a straight line. There can be a big difference between Euclidean distance and travel time, especially after considering potentially complex routes and traffic, which are what really matters when responding to customer calls.

3. EM

```
#1
set.seed(52)
sani500 <- sani[sample(nrow(sani),500),]
#2
mod2 <- mvnormalmixEM(select(sani500, Latitude, Longitude), k=2)
```

```
## number of iterations= 94
```

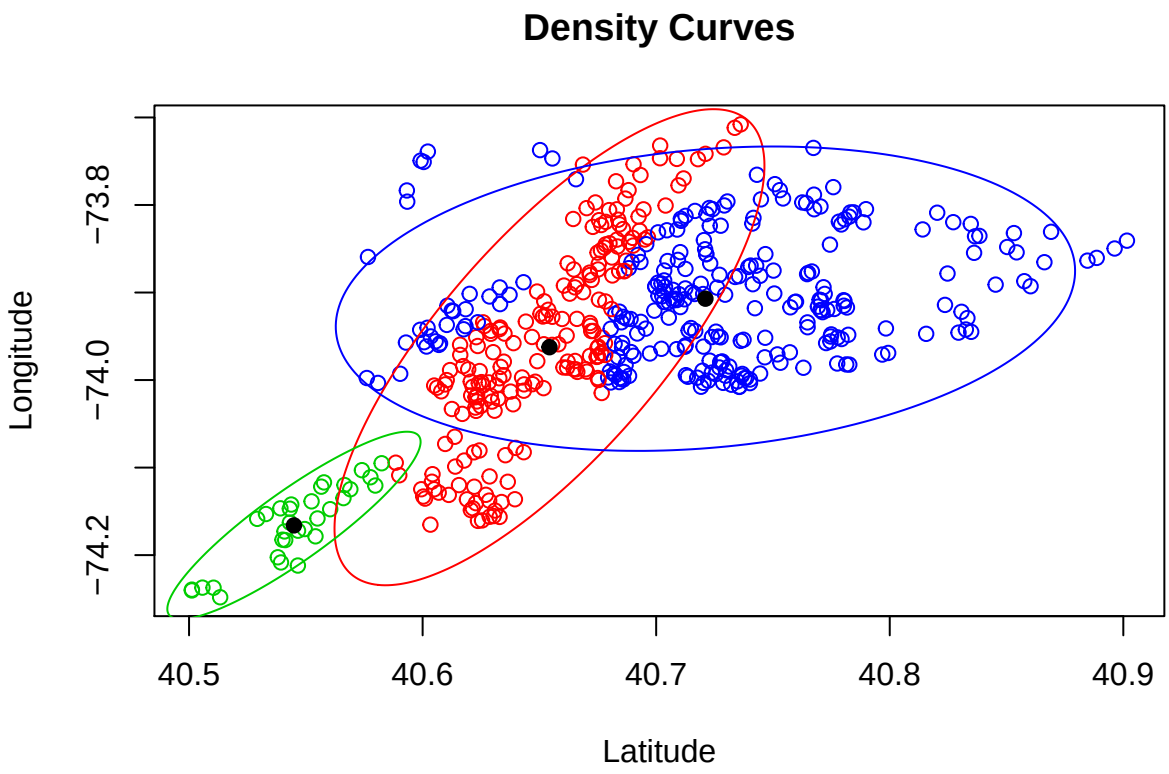
```
plot(mod2, whichplots = 2, xlab2="Latitude", ylab2="Longitude")
```




```
#3
mod3 <- mvnormalmixEM(select(sani500, Latitude, Longitude), k=3)
```

```
## number of iterations= 157
```

```
plot(mod3, whichplots = 2, xlab2="Latitude", ylab2="Longitude")
```


Density Curves
Longitude
−73.8
−74.0
−74.2
40.5
40.6
40.7
40.8
40.9
Latitude