Final

Arielle

05/01/2020

# Part 1: Neighborhood environments in New York City

### Question 1: Run an appropriate unsupervised learning analysis that will address the above research question. Your analysis should include a data-driven approach to determine the optimal number of outputs that are retained/produced by the chosen learning technique.

First, I read in the nyc environment dataset and removed any of the NAs in the dataset (there were no missing data in this data set). I then checked if scaling was needed by looking at the means and standard devations. Some of the means and standard devations are different from each other which makes it necesarry to scale the data.

nyc.env<-read.csv("nyc\_env\_correct.csv", header=TRUE)  
  
nyc.env.nomiss<-na.omit(nyc.env)  
  
#Check means and SDs to determine if scaling is necessary  
colMeans(nyc.env, na.rm=TRUE)

## Pct.Low.Inc Diesel.Conc   
## 0.3663891 3.9260364   
## NPL.Site.Count..wt.by.dist Pct.pre1960.Housing   
## 0.1573208 0.6720128

apply(nyc.env, 2, sd, na.rm=TRUE)

## Pct.Low.Inc Diesel.Conc   
## 0.1932175 1.5077004   
## NPL.Site.Count..wt.by.dist Pct.pre1960.Housing   
## 0.1724710 0.2312597

nyc.env.scale =   
 scale(nyc.env)

A clustering analysis was done next using k-means clustering to assess if there are any clusters in the data and where they are. The gap statistic determined that 8 clusters were needed for this data and that was entered to the final cluster model.

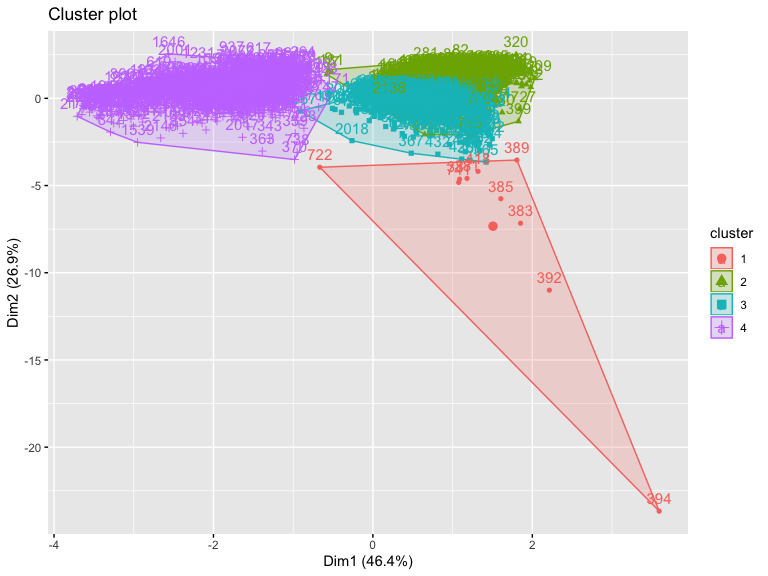
set.seed(100)  
clusters<-kmeans(nyc.env.scale, 4, nstart=25)  
str(clusters)

## List of 9  
## $ cluster : int [1:2140] 4 4 4 4 4 2 4 4 2 2 ...  
## $ centers : num [1:4, 1:4] -0.783 0.955 -0.613 -0.149 0.628 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ : chr [1:4] "1" "2" "3" "4"  
## .. ..$ : chr [1:4] "Pct.Low.Inc" "Diesel.Conc" "NPL.Site.Count..wt.by.dist" "Pct.pre1960.Housing"  
## $ totss : num 8556  
## $ withinss : num [1:4] 571 529 821 1469  
## $ tot.withinss: num 3389  
## $ betweenss : num 5167  
## $ size : int [1:4] 10 646 837 647  
## $ iter : int 3  
## $ ifault : int 0  
## - attr(\*, "class")= chr "kmeans"

clusters

## K-means clustering with 4 clusters of sizes 10, 646, 837, 647  
##   
## Cluster means:  
## Pct.Low.Inc Diesel.Conc NPL.Site.Count..wt.by.dist Pct.pre1960.Housing  
## 1 -0.7828869 0.6275689 8.91461917 0.5750571  
## 2 0.9550593 0.6719798 -0.09958896 0.5213409  
## 3 -0.6128879 0.5912988 0.02153468 0.5728717  
## 4 -0.1486125 -1.4455822 -0.06620749 -1.2705261  
##   
## Clustering vector:  
## [1] 4 4 4 4 4 2 4 4 2 2 4 4 2 4 2 4 4 4 3 4 4 4 2 4 2 2 2 2 2 4 2 2 2 2  
## [35] 4 2 2 4 2 2 4 4 2 4 2 2 4 4 2 4 2 4 2 4 2 2 4 2 3 2 4 2 3 4 4 4 4 4  
## [69] 4 4 4 4 4 4 4 4 4 2 2 4 4 2 4 2 2 2 2 2 2 4 2 4 4 4 4 2 3 3 4 3 4 4  
## [103] 4 2 2 2 2 2 2 2 2 2 3 4 2 2 2 2 2 2 2 2 4 2 4 4 4 2 2 3 2 3 2 4 2 2  
## [137] 2 3 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 3 2 2  
## [171] 2 2 3 2 2 3 3 3 3 2 3 2 3 2 3 2 3 2 3 2 4 3 2 2 2 2 3 4 4 2 3 3 2 2  
## [205] 2 3 2 3 3 4 4 4 4 3 3 4 4 4 4 4 3 3 3 3 4 3 2 3 2 2 4 4 2 3 2 3 2 2  
## [239] 4 3 4 4 3 3 3 3 4 4 3 4 4 3 4 2 4 4 4 2 3 4 3 2 2 4 2 4 2 4 2 2 2 2  
## [273] 2 4 2 3 2 4 2 2 2 2 4 4 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 4 2 2 3 2 3 2  
## [307] 2 2 2 2 2 2 3 3 2 2 2 2 3 2 3 2 3 3 3 3 3 3 2 2 2 4 4 4 4 4 3 2 3 3  
## [341] 3 3 4 4 4 4 4 2 4 2 2 2 3 4 3 3 3 3 4 4 3 3 4 4 3 4 3 3 3 4 4 4 3 3  
## [375] 4 4 3 3 3 3 1 2 1 3 1 2 1 4 1 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 3 2  
## [409] 2 2 2 2 2 2 2 3 2 1 2 3 2 2 4 2 3 3 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 2  
## [443] 3 3 3 3 3 3 3 3 3 3 3 3 3 4 3 3 3 4 3 3 3 3 4 3 4 3 4 2 3 2 3 3 2 3  
## [477] 3 3 2 2 2 4 2 3 2 3 3 3 3 3 3 4 3 3 3 3 3 2 2 2 3 2 3 2 2 2 2 2 2 4  
## [511] 2 2 2 2 3 2 3 2 3 2 4 2 2 2 3 4 2 2 3 2 3 3 2 2 2 2 3 2 2 2 4 2 4 4  
## [545] 3 4 3 2 2 2 3 3 2 2 3 4 3 2 2 3 2 2 2 3 4 4 4 4 2 3 2 3 2 2 2 2 3 4  
## [579] 4 4 3 3 3 2 4 3 2 3 4 4 2 4 2 4 4 2 2 4 2 2 2 2 2 3 2 4 3 4 2 4 2 2  
## [613] 2 3 2 2 4 3 4 2 2 4 4 2 4 2 4 2 4 2 4 4 4 4 2 2 4 2 4 2 4 4 2 3 2 3  
## [647] 2 2 3 2 2 3 4 2 2 3 2 2 2 2 2 3 4 2 3 3 2 3 2 3 2 2 2 2 2 3 2 3 4 2  
## [681] 2 3 4 3 2 3 3 2 3 4 2 4 2 2 3 2 2 4 2 2 2 2 2 2 2 4 2 2 2 2 2 2 3 2  
## [715] 3 2 2 2 3 3 4 1 3 3 2 3 2 3 3 4 2 2 4 2 4 2 2 4 2 2 1 2 2 2 2 2 4 2  
## [749] 4 2 4 3 3 2 3 2 3 3 3 3 3 3 4 2 4 2 2 2 4 2 2 4 2 3 2 4 2 2 3 3 3 2  
## [783] 2 2 3 2 3 2 3 4 3 2 2 2 4 2 4 2 4 2 4 2 4 2 2 2 3 2 3 2 4 3 3 2 2 3  
## [817] 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 2 4 2 4 3 3 3 3 3 3 3 3 4 3 2 4 4  
## [851] 3 2 3 4 3 4 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 3 3 3 3 3 2 3 3 3 3 4  
## [885] 4 4 3 4 4 4 4 4 3 3 3 3 3 3 3 2 3 3 3 3 3 2 2 3 3 3 3 3 3 3 2 3 3 3  
## [919] 2 3 3 3 3 2 2 2 3 3 3 3 2 2 2 2 2 3 4 3 3 3 2 2 2 2 2 3 3 3 3 3 3 3  
## [953] 3 3 3 4 3 3 3 2 3 3 2 3 2 2 2 3 2 3 2 2 2 3 2 2 2 2 4 4 4 4 4 4 2 4  
## [987] 4 4 2 4 3 3 3 3 3 2 3 2 3 4 4 4 4 3 3 4 3 4 4 2 4 4 4 3 4 4 4 4 4 4  
## [1021] 4 4 3 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 4 4 4 2 2 2 2 4 4 4 3 2 2 2  
## [1055] 4 4 4 4 4 4 4 4 2 2 2 2 2 2 2 2 4 2 2 2 2 2 2 2 2 2 2 2 2 4 2 2 4 4  
## [1089] 3 3 4 4 2 4 4 2 4 4 4 3 3 4 4 4 4 2 2 2 4 4 4 4 2 4 4 4 4 2 3 4 4 3  
## [1123] 3 3 4 2 4 3 4 3 2 4 3 3 3 3 4 3 3 3 4 4 3 4 4 4 3 3 3 4 3 4 3 4 3 3  
## [1157] 4 4 3 4 3 3 4 3 3 4 3 3 3 3 4 4 4 4 3 4 3 4 3 3 3 4 3 4 4 4 4 3 4 4  
## [1191] 4 3 4 4 4 3 4 4 4 4 4 4 3 4 4 4 4 4 2 4 3 3 4 4 4 3 3 4 3 4 4 4 4 4  
## [1225] 4 3 3 3 3 3 4 3 4 4 4 4 4 4 3 4 3 3 4 4 4 4 4 4 4 3 3 4 3 3 3 3 4 3  
## [1259] 4 3 2 3 2 3 4 3 4 3 4 4 3 4 2 3 2 4 2 3 4 4 2 3 4 2 4 3 2 2 4 3 4 4  
## [1293] 4 4 3 3 3 3 4 3 2 3 2 2 4 3 4 2 4 4 3 4 2 4 2 3 4 2 4 2 3 2 2 3 2 2  
## [1327] 2 4 3 2 2 2 4 3 4 4 2 4 3 4 2 4 2 2 2 2 2 3 3 4 2 2 2 2 2 2 3 4 2 2  
## [1361] 3 2 2 3 2 2 3 4 2 3 3 2 4 4 4 4 4 3 3 2 3 3 3 3 3 2 2 4 3 3 3 2 3 3  
## [1395] 3 3 3 4 3 2 4 2 3 2 3 2 4 3 2 4 2 3 3 3 2 3 4 3 3 4 4 2 3 3 4 3 3 2  
## [1429] 4 2 4 4 2 3 2 3 4 2 3 2 3 3 4 3 3 3 3 2 3 3 3 2 3 4 3 2 2 3 3 2 3 3  
## [1463] 3 2 3 2 3 3 3 3 3 3 3 4 2 4 3 3 3 2 3 3 3 3 3 2 3 3 3 2 3 2 3 3 2 2  
## [1497] 3 3 2 2 3 3 3 3 2 4 3 3 3 2 3 2 3 3 3 3 4 3 3 3 3 3 3 2 3 3 3 2 2 4  
## [1531] 2 3 3 3 3 2 3 3 4 3 3 4 2 3 2 2 2 2 3 4 3 3 3 3 2 4 3 3 3 2 2 2 4 3  
## [1565] 4 2 4 3 2 2 4 2 3 2 4 2 2 2 4 3 3 3 3 2 4 3 3 3 3 2 2 4 3 3 4 2 3 4  
## [1599] 3 4 2 4 3 3 4 4 2 3 2 3 3 3 2 3 2 3 2 3 3 3 3 4 2 3 2 2 2 4 3 3 3 2  
## [1633] 3 4 3 4 3 2 2 2 2 2 2 4 3 4 2 3 3 4 4 4 2 4 4 2 4 2 2 3 3 2 4 3 4 3  
## [1667] 4 2 4 4 2 3 2 4 2 2 2 3 2 3 4 4 3 3 3 3 2 3 4 3 2 3 3 4 3 3 3 4 2 3  
## [1701] 3 3 3 4 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 2 3 2 2  
## [1735] 3 2 2 3 2 3 3 3 2 3 4 3 4 3 3 2 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 2 3 3  
## [1769] 3 3 3 3 3 4 3 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3 3  
## [1803] 3 3 3 3 3 3 3 4 4 3 3 3 4 3 4 4 3 3 3 3 3 3 3 4 3 4 3 4 3 3 3 3 3 3  
## [1837] 3 3 3 3 3 3 4 2 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 4 3 3 4 3 4  
## [1871] 4 4 4 4 4 4 3 3 3 2 2 3 3 4 3 2 4 4 4 4 2 3 4 3 4 4 2 3 4 4 4 4 4 4  
## [1905] 3 3 3 4 4 4 4 4 4 4 4 4 3 4 4 4 4 4 4 4 4 3 4 3 4 4 4 4 4 4 4 4 4 3  
## [1939] 4 4 3 3 3 4 3 3 3 4 3 3 4 3 3 3 3 3 3 3 2 3 3 4 4 4 2 3 4 4 4 3 4 2  
## [1973] 3 3 3 3 4 3 2 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 3 3 3 3 3  
## [2007] 3 3 3 3 3 3 3 3 3 4 4 3 3 4 4 3 4 3 4 3 3 3 3 3 3 3 4 4 4 4 4 2 4 4  
## [2041] 4 4 4 4 4 4 4 3 4 4 4 3 4 3 4 4 2 4 4 4 4 4 3 4 4 4 4 3 3 3 4 4 4 4  
## [2075] 4 4 4 3 3 4 4 4 4 4 4 4 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3  
## [2109] 4 3 4 4 4 4 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 4 4  
##   
## Within cluster sum of squares by cluster:  
## [1] 570.5973 529.3671 820.6747 1468.6461  
## (between\_SS / total\_SS = 60.4 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss"   
## [5] "tot.withinss" "betweenss" "size" "iter"   
## [9] "ifault"

fviz\_cluster(clusters, data=nyc.env.scale)



#Conduct a gap\_statistic analysis to determine optimal number of clusters  
gap\_stat<-clusGap(nyc.env.scale, FUN=kmeans, nstart=25, K.max=9, B=50)  
print(gap\_stat, method="firstmax")

## Clustering Gap statistic ["clusGap"] from call:  
## clusGap(x = nyc.env.scale, FUNcluster = kmeans, K.max = 9, B = 50, nstart = 25)  
## B=50 simulated reference sets, k = 1..9; spaceH0="scaledPCA"  
## --> Number of clusters (method 'firstmax'): 2  
## logW E.logW gap SE.sim  
## [1,] 7.153008 8.817508 1.664500 0.005286697  
## [2,] 6.834087 8.523273 1.689187 0.006066177  
## [3,] 6.682157 8.363967 1.681810 0.005876694  
## [4,] 6.628777 8.217023 1.588246 0.005057143  
## [5,] 6.562368 8.130070 1.567702 0.004098779  
## [6,] 6.491211 8.054833 1.563622 0.005549765  
## [7,] 6.426818 8.005593 1.578775 0.004709737  
## [8,] 6.372718 7.959780 1.587061 0.004523679  
## [9,] 6.346023 7.918436 1.572413 0.004702976

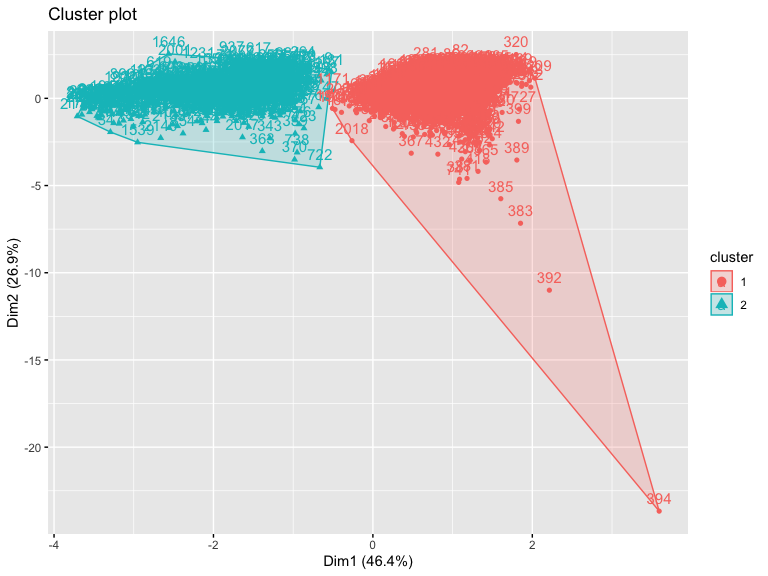
clusters.1<-kmeans(nyc.env.scale, 2, nstart=25)  
str(clusters.1)

## List of 9  
## $ cluster : int [1:2140] 2 2 2 2 2 1 2 2 2 1 ...  
## $ centers : num [1:2, 1:4] 0.0618 -0.1414 0.6306 -1.4423 0.0257 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ : chr [1:2] "1" "2"  
## .. ..$ : chr [1:4] "Pct.Low.Inc" "Diesel.Conc" "NPL.Site.Count..wt.by.dist" "Pct.pre1960.Housing"  
## $ totss : num 8556  
## $ withinss : num [1:2] 3572 1516  
## $ tot.withinss: num 5088  
## $ betweenss : num 3468  
## $ size : int [1:2] 1489 651  
## $ iter : int 1  
## $ ifault : int 0  
## - attr(\*, "class")= chr "kmeans"

clusters.1

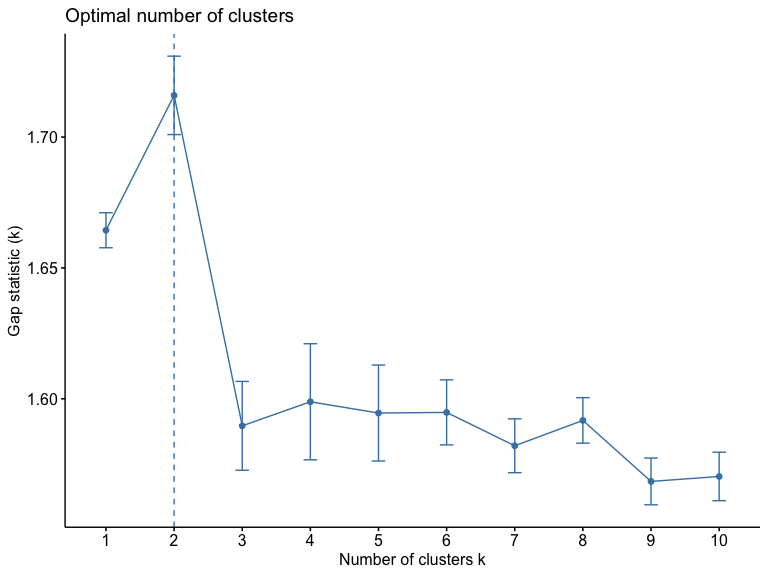
## K-means clustering with 2 clusters of sizes 1489, 651  
##   
## Cluster means:  
## Pct.Low.Inc Diesel.Conc NPL.Site.Count..wt.by.dist Pct.pre1960.Housing  
## 1 0.06183817 0.6305858 0.02570246 0.5534846  
## 2 -0.14143937 -1.4423077 -0.05878795 -1.2659578  
##   
## Clustering vector:  
## [1] 2 2 2 2 2 1 2 2 2 1 2 2 1 2 1 2 2 2 1 2 2 2 1 2 1 1 1 1 1 2 1 1 1 1  
## [35] 2 1 1 2 1 1 2 2 1 2 1 1 2 2 1 2 1 2 1 2 1 1 2 1 1 1 2 1 1 2 2 2 2 2  
## [69] 2 2 2 2 2 2 2 2 2 1 1 2 2 1 2 1 1 1 1 1 1 2 1 2 2 2 2 1 1 1 2 1 2 2  
## [103] 2 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 2 1 2 2 2 1 1 1 2 1 1 2 1 1  
## [137] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [171] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 2 1 1 1 1 1  
## [205] 1 1 1 1 1 2 2 2 2 1 1 2 2 2 2 2 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 1 1 1  
## [239] 2 1 2 2 1 1 1 1 2 2 1 2 2 1 2 1 2 2 2 1 1 2 1 1 1 2 1 2 1 2 1 1 1 1  
## [273] 1 2 1 1 1 2 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1  
## [307] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1  
## [341] 1 1 2 2 2 2 2 1 2 1 1 1 1 2 1 1 1 1 2 2 1 1 2 2 1 2 1 1 1 2 2 2 1 1  
## [375] 2 2 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [409] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [443] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 1 1 1 1 2 1 2 1 2 1 1 1 1 1 1 1  
## [477] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2  
## [511] 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2  
## [545] 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 2  
## [579] 2 2 1 1 1 1 2 1 1 1 2 2 1 2 1 2 2 1 1 2 1 1 1 1 1 1 1 2 1 2 1 2 1 1  
## [613] 1 1 1 1 2 1 2 1 1 2 2 1 2 1 2 1 2 1 2 2 2 2 1 1 2 1 2 1 2 2 1 1 1 1  
## [647] 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1  
## [681] 1 1 2 1 1 1 1 1 1 2 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1  
## [715] 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 2 1 1 2 1 2 1 1 2 1 1 1 1 1 1 1 1 2 1  
## [749] 2 1 2 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 2 1 1 2 1 1 1 2 1 1 1 1 1 1  
## [783] 1 1 1 1 1 1 1 2 1 1 1 1 2 1 2 1 2 1 2 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1  
## [817] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 2 1 1 2 2  
## [851] 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2  
## [885] 2 2 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [919] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [953] 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 2  
## [987] 2 2 1 2 1 1 1 1 1 1 1 1 1 2 2 2 2 1 1 2 1 2 2 1 2 2 2 1 2 2 2 2 2 2  
## [1021] 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 2 2 2 1 1 1 1 2 2 2 1 1 1 1  
## [1055] 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 2  
## [1089] 1 1 2 2 1 2 2 1 2 2 2 1 1 2 2 2 2 1 1 1 2 2 2 2 1 2 2 2 2 1 1 2 2 1  
## [1123] 1 1 2 1 2 1 2 1 1 2 1 1 1 1 2 1 1 1 2 2 1 2 2 2 1 1 1 2 1 2 1 2 1 1  
## [1157] 2 2 1 2 1 1 2 1 1 2 2 1 1 1 1 2 2 2 1 2 1 2 1 1 1 2 1 2 2 2 2 1 2 2  
## [1191] 2 1 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 1 1 2 2 2 1 1 2 1 2 2 2 2 2  
## [1225] 2 1 1 1 1 1 2 1 2 2 2 2 2 2 1 2 1 1 2 2 2 2 2 2 2 1 1 2 1 1 1 1 2 1  
## [1259] 2 1 1 1 1 1 2 1 2 1 2 2 1 2 1 1 1 2 1 1 2 2 1 1 2 1 2 1 1 1 2 1 2 2  
## [1293] 2 2 1 1 1 1 2 1 1 1 1 1 2 1 2 1 2 2 1 2 1 2 1 1 2 1 2 1 1 1 1 1 1 1  
## [1327] 1 2 1 1 1 1 2 1 2 2 1 2 1 2 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1  
## [1361] 1 1 1 1 1 1 1 2 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1  
## [1395] 1 1 1 2 1 1 2 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 1 2 1 1 1  
## [1429] 2 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1  
## [1463] 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [1497] 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2  
## [1531] 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1  
## [1565] 2 1 2 1 1 1 2 1 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 2 1 1 2  
## [1599] 1 2 1 2 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1  
## [1633] 1 2 1 2 1 1 1 1 1 1 1 2 1 2 1 1 1 2 2 2 1 2 2 1 2 1 1 1 1 1 2 1 2 1  
## [1667] 2 1 2 2 1 1 1 2 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1  
## [1701] 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [1735] 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [1769] 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1  
## [1803] 1 1 1 1 1 1 1 2 2 1 1 1 2 1 2 2 1 1 1 1 1 1 1 2 1 2 1 2 1 1 1 1 1 1  
## [1837] 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 2  
## [1871] 2 2 2 2 2 2 1 1 1 1 1 1 1 2 1 1 2 2 2 2 1 1 2 1 2 2 1 1 2 2 2 2 2 2  
## [1905] 1 1 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 1  
## [1939] 2 2 1 1 1 2 1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 2 2 2 1 2 1  
## [1973] 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1  
## [2007] 1 1 1 1 1 1 1 1 1 2 2 1 1 2 2 1 2 1 2 1 1 1 1 1 1 1 2 2 2 2 2 1 2 2  
## [2041] 2 2 2 2 2 2 2 1 2 2 2 1 2 1 2 2 1 2 2 2 2 2 1 2 2 2 2 1 1 1 2 2 2 2  
## [2075] 2 2 2 1 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1  
## [2109] 2 1 2 2 2 2 1 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  
##   
## Within cluster sum of squares by cluster:  
## [1] 3572.237 1516.013  
## (between\_SS / total\_SS = 40.5 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss"   
## [5] "tot.withinss" "betweenss" "size" "iter"   
## [9] "ifault"

fviz\_cluster(clusters.1, data=nyc.env.scale)

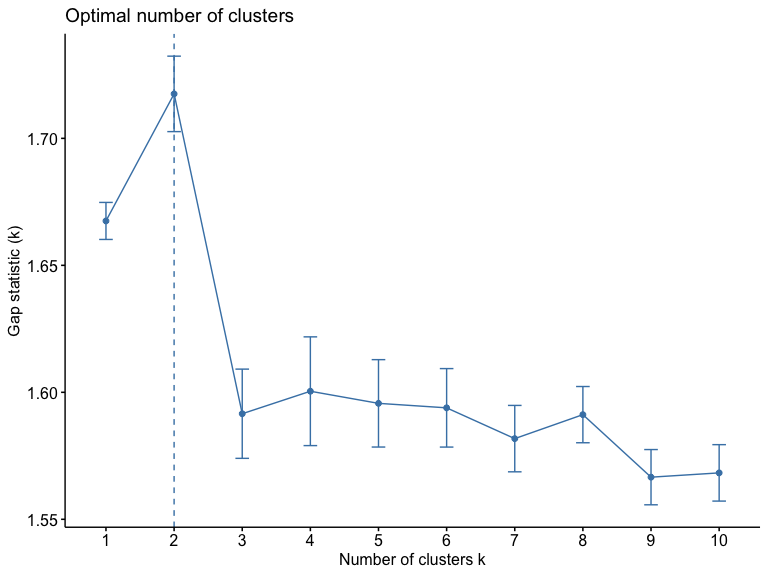


The next thing that I did was Hierarichical clustering setting the distance Euclidian methods since the data was all continuous and numeric. A dissimilarity matrix was used to get the distance and looked at the difference clustering methods: Complete, Single, and average A gap statistic was also used to get the optimal number of clusters.

set.seed(100)  
  
# Create Dissimilarity matrix  
diss.matrix <- dist(nyc.env.scale, method = "euclidean")  
  
gap\_stat <- clusGap(nyc.env.scale, FUN = hcut, nstart = 25, K.max = 10, B = 50, mc\_metric = "euclidean")  
fviz\_gap\_stat(gap\_stat)



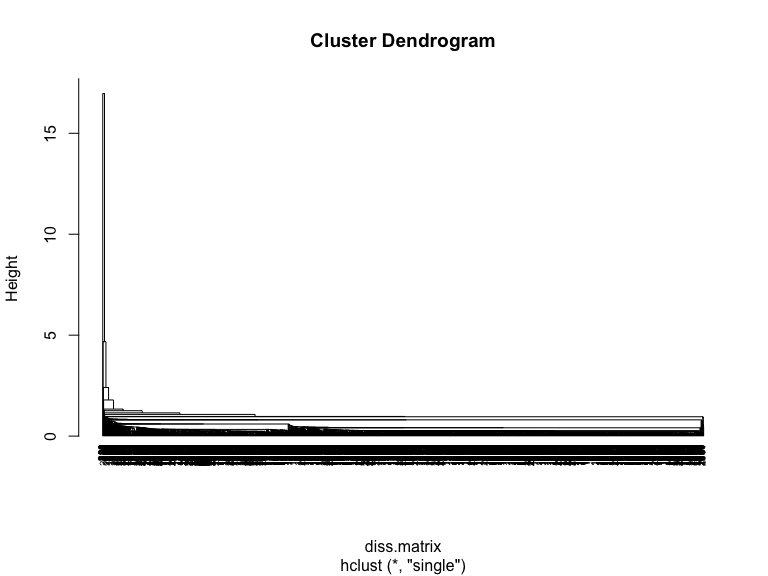
# Hierarchical clustering using Single Linkage  
gap\_stat1 <- clusGap(nyc.env.scale, FUN = hcut, nstart = 25, K.max = 10, B = 50, mc\_metric = "euclidean", method = "single")  
fviz\_gap\_stat(gap\_stat1)



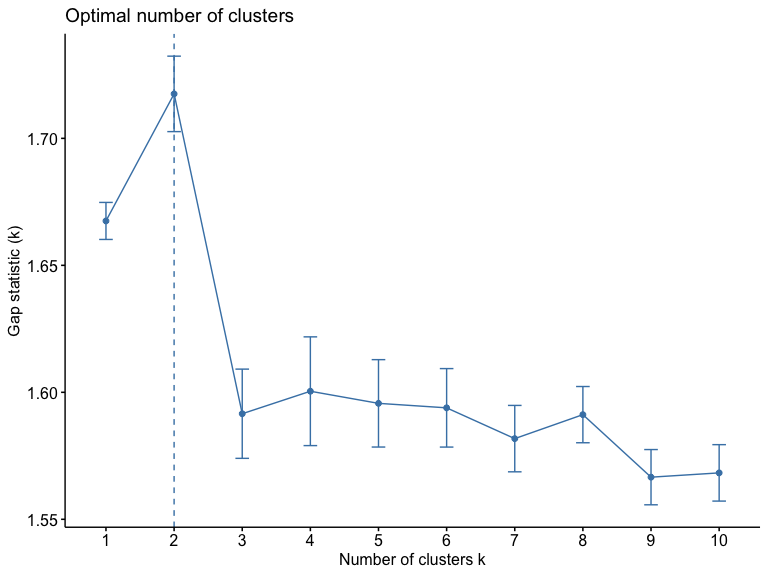
# Create Dissimilarity matrix  
diss.matrix <- dist(nyc.env.scale, method = "euclidean")  
  
hc1 <- hclust(diss.matrix, method = "single" )  
  
groups.1<-cutree(hc1,4)  
aggregate(nyc.env.scale,list(groups.1),mean)

## Group.1 Pct.Low.Inc Diesel.Conc NPL.Site.Count..wt.by.dist  
## 1 1 0.001584854 -0.001222701 -0.02428273  
## 2 2 -0.921153793 0.872121335 8.54242052  
## 3 3 -1.389996824 0.867652022 13.19376437  
## 4 4 -1.075681561 0.873137794 30.15601400  
## Pct.pre1960.Housing  
## 1 -0.001139374  
## 2 0.922220182  
## 3 0.956895487  
## 4 0.555726512

# Plot the obtained dendrogram using Single  
plot(hc1, cex = 0.6, hang = -1)



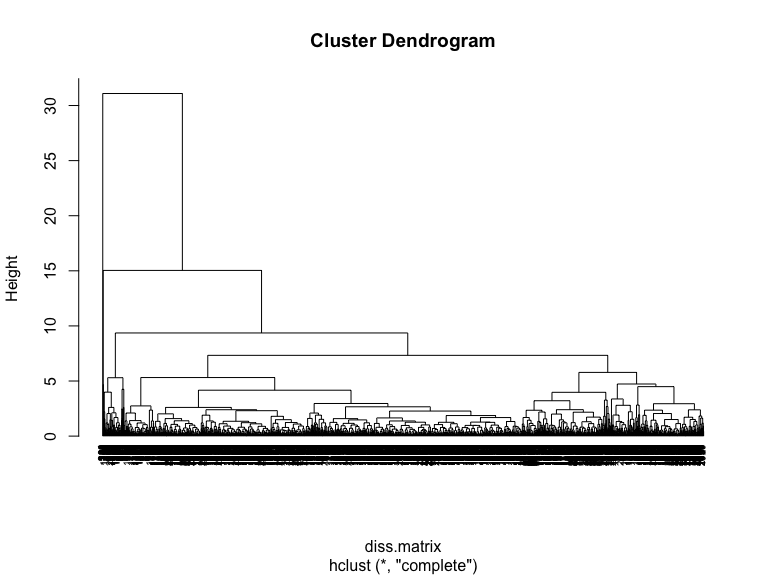
# Hierarchical clustering using Complete Linkage  
  
gap\_stat2 <- clusGap(nyc.env.scale, FUN = hcut, nstart = 25, K.max = 10, B = 50, mc\_metric = "euclidean", method = "complete")  
fviz\_gap\_stat(gap\_stat2)



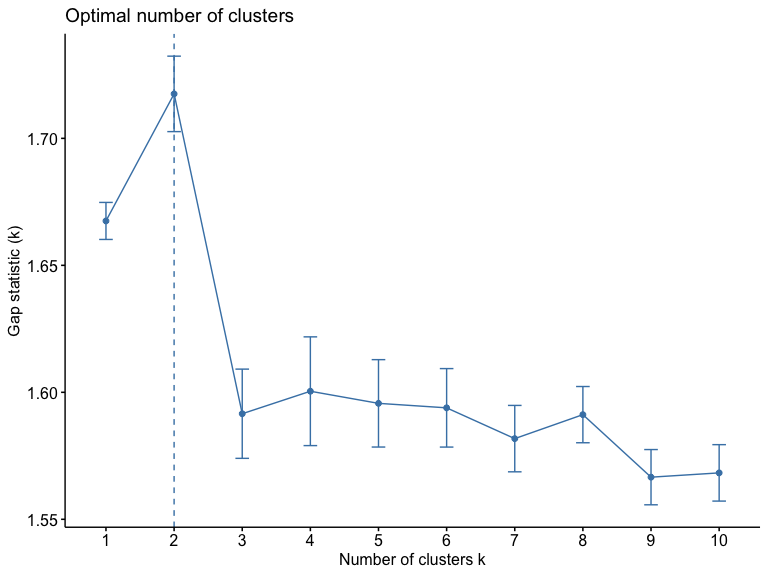
hc2 <- hclust(diss.matrix, method = "complete" )  
  
groups.2 <-cutree(hc2,4)  
aggregate(nyc.env.scale,list(groups.2),mean)

## Group.1 Pct.Low.Inc Diesel.Conc NPL.Site.Count..wt.by.dist  
## 1 1 0.02591283 -0.02490863 -0.1145826  
## 2 2 -0.66727225 0.64998234 2.4583610  
## 3 3 -1.15557531 0.86988668 10.8680924  
## 4 4 -1.07568156 0.87313779 30.1560140  
## Pct.pre1960.Housing  
## 1 -0.02463258  
## 2 0.64476713  
## 3 0.93955783  
## 4 0.55572651

# Plot the obtained dendrogram using Complete  
plot(hc2, cex = 0.6, hang = -1)



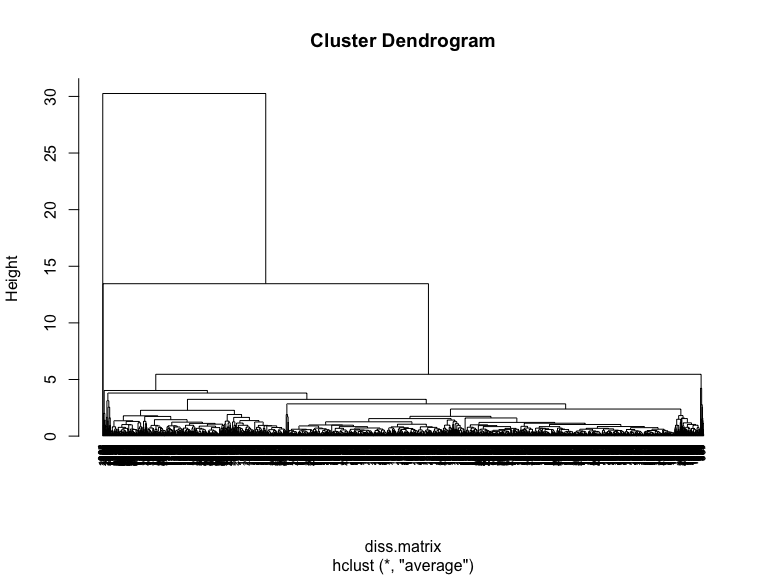
# Hierarchical clustering using Average Linkage  
gap\_stat3 <- clusGap(nyc.env.scale, FUN = hcut, nstart = 25, K.max = 10, B = 50, mc\_metric = "euclidean", method = "average")  
fviz\_gap\_stat(gap\_stat3)



hc3 <- hclust(diss.matrix, method = "average" )  
  
groups.3<-cutree(hc3,4)  
aggregate(nyc.env.scale,list(groups.3),mean)

## Group.1 Pct.Low.Inc Diesel.Conc NPL.Site.Count..wt.by.dist  
## 1 1 0.002048087 -0.004237551 -0.04815852  
## 2 2 -0.157379495 0.605686967 4.91960229  
## 3 3 -1.389996824 0.867652022 13.19376437  
## 4 4 -1.075681561 0.873137794 30.15601400  
## Pct.pre1960.Housing  
## 1 -0.003360289  
## 2 0.469279412  
## 3 0.956895487  
## 4 0.555726512

# Plot the obtained dendrogram using Average  
plot(hc3, cex = 0.6, hang = -1)



### Question 2: Describe the outputs of the analysis in terms of their composition of the input features.

The optimal number of cluster based on the k means clustering plot are 2 cluster, this consistent with the optimal number of cluster based on hierarchical clustering. For this type of clustering, the gap statistic stated that two was also the optimal number of cluster to have. Two clusters was consistent across the different types of linkage methods, single, complete and average. Although the cluster dendrograms all look different the results from all of them are the same.

The means for perecnt of low income and percentage of housing stock built before 1960 are similar across all the linkaage methods. The means observed were similar to each other, there was a little more varaition in the means but not a huge varaition for air concentration of diesal extract and proximity to hazardous waste sites.

### Question 3: List a subsequent research question that could be addressed using the output of this analysis. The output can be used as an exposure, outcome or confounding variable. State what type of research question it is: descriptive, explanatory or predictive.

A subsequential research question that can be addressed using the outbut of this analysis is a explanatory question which is if a proportion of housing stocking built before 1960 as a proxy for potential lead exposure is associated with increase ambient air concentrations of diesel exhaust, while controling for proportion of neighborhood with household income lwoer than the city median.

# Part 2: Choose your own supervised adventure

### B) Researchers are interested in uncovering nutritional factors that influence the development of diabetes. To generate hypotheses to guide future research, they ask you to utilize feature selection methods to identify the factors that are most relevant to fasting blood glucose. (The dataset you are given is glucose.csv.) You should try two different algorithms for feature selection and comment if they identify the same set of variables as being important for serum glucose.

The first thing that will be done is import and clean the data.

glucose.raw<-read.csv("glucose.csv", header=TRUE)  
  
glucose =  
 glucose.raw %>%   
 drop\_na() %>%   
 janitor::clean\_names()

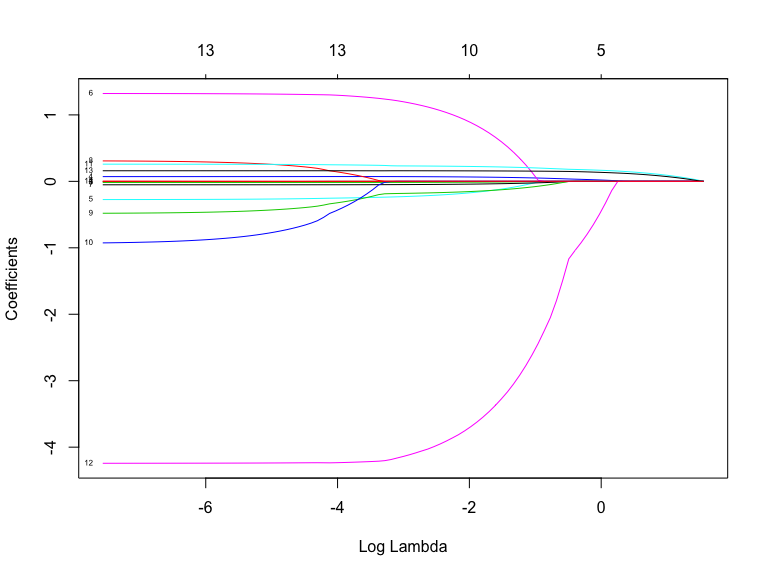
Next I will partition the data so that there is a 70/30 split and set the seed to 100.

set.seed(100)  
training.data.glucose<-glucose$lbxsgl %>% createDataPartition(p=0.7, list=F)  
train.data.glucose<-glucose[training.data.glucose, ]  
test.data.glucose<-glucose[-training.data.glucose, ]  
  
#Store outcome   
glucose.train<-train.data.glucose$lbxsgl  
glucose.test<-test.data.glucose$lbxsgl  
  
# Store the outcome in one train and test and the predictors in another   
#model matrix- will create indicator variables for categorical varaibles, it does not do anything to the continuous variables  
x.train<-model.matrix(lbxsgl~., train.data.glucose)[,-1]  
x.test<-model.matrix(lbxsgl~., test.data.glucose)[,-1]

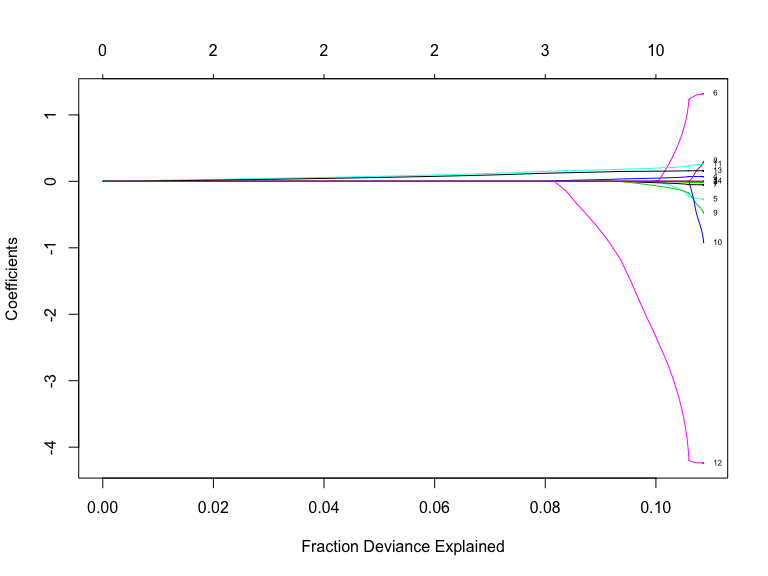
I first performed a regulaized regression using LASSO and Elastic net. First I performed a model with LASSO setting my alpha to one, by doing this it shrinks the features in the model to 0 only leaving the ones that were most important for the model. The second regularized regression that I did was Elastic net, this is done by setting the alpha to 0.5.

REGULARIZED REGRESSION USING LASSO

#LASSO  
model.2.1<-glmnet(x.train, glucose.train, alpha=1, standardize = TRUE)  
  
plot(model.2.1, xvar="lambda", label=TRUE)



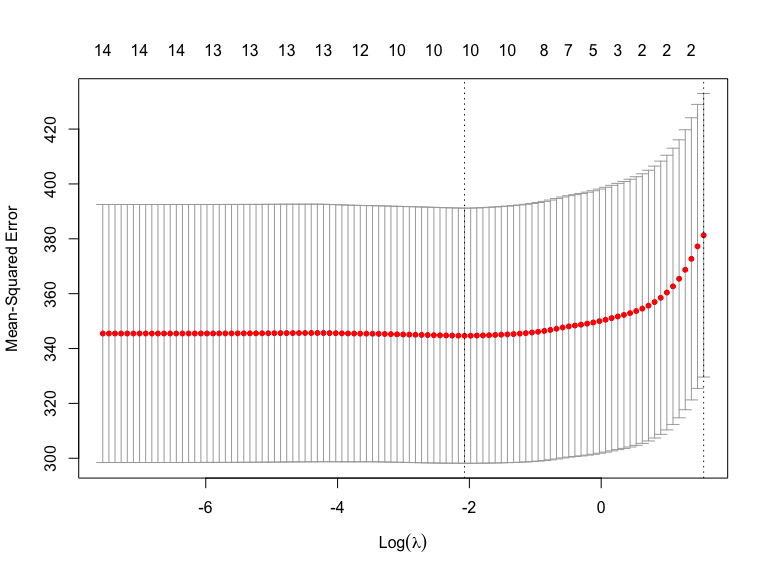
plot(model.2.1, xvar="dev", label=TRUE)



model.2.1$beta[,1]

## x seqn lbxscu lbxsse lbxmg lbxlyc lbxszn bmxwt   
## 0 0 0 0 0 0 0 0   
## bmxht bmxbmi bmxwaist riagendr ridageyr ridreth1   
## 0 0 0 0 0 0

model.1.cv<-cv.glmnet(x.train, glucose.train, alpha=1)  
plot(model.1.cv)



model.1.cv$lambda.min

## [1] 0.1257162

model.1.cv$lambda.1se

## [1] 4.733134

model.1.train.final<-glmnet(x.train, glucose.train, alpha=1, lambda=model.1.cv$lambda.min)  
coef(model.1.train.final)

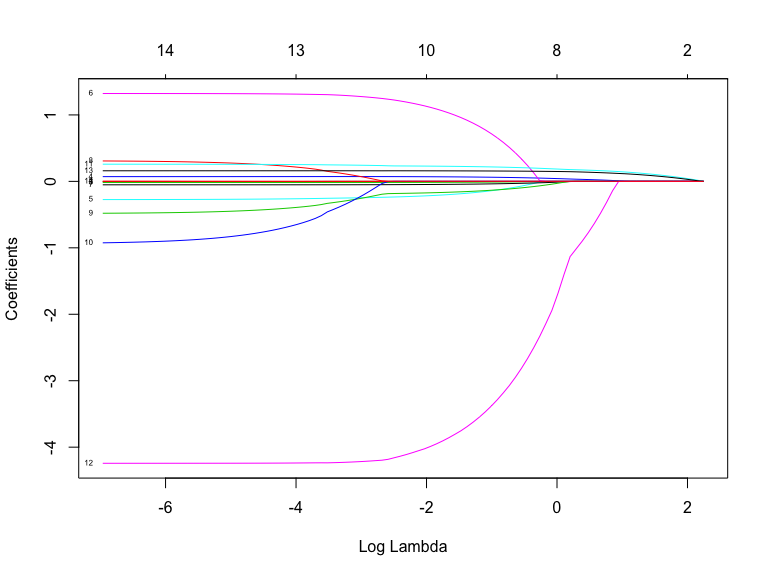
## 15 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## (Intercept) 85.0397702856  
## x .   
## seqn 0.0001321868  
## lbxscu -0.0132769689  
## lbxsse 0.0664025402  
## lbxmg -0.1793991957  
## lbxlyc 0.9290736244  
## lbxszn -0.0426124384  
## bmxwt .   
## bmxht -0.1571261065  
## bmxbmi .   
## bmxwaist 0.2252242823  
## riagendr -3.7588951448  
## ridageyr 0.1579758545  
## ridreth1 .

model.1.test.pred<-model.1.train.final %>% predict(x.test) %>% as.vector()  
data.frame(RMSE=RMSE(model.1.test.pred, glucose.test), RSQ=R2(model.1.test.pred, glucose.test))

## RMSE RSQ  
## 1 16.58982 0.115314

REGULARIZED REGRESSION USING ELASTIC NET

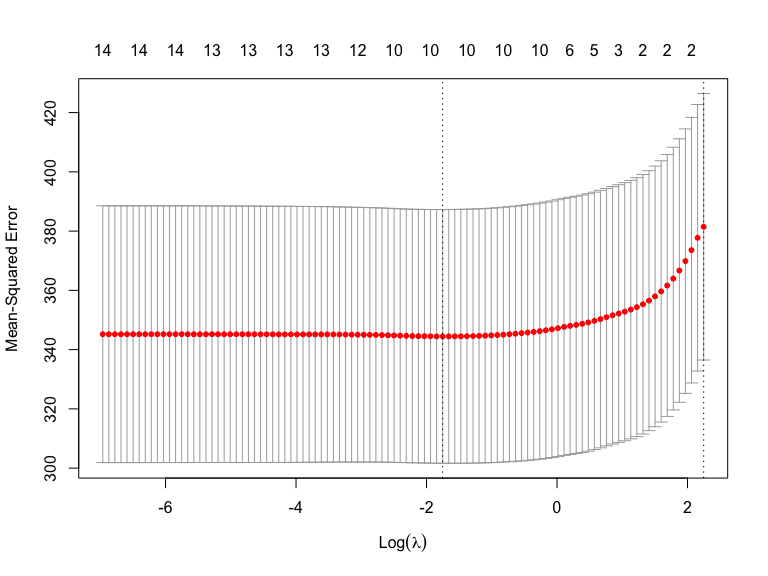
#Elastic Net  
model.2.2<-glmnet(x.train, glucose.train, alpha=0.5, standardize = TRUE)  
  
plot(model.2.2, xvar="lambda", label=TRUE)



model.2.2$beta[,1]

## x seqn lbxscu lbxsse lbxmg lbxlyc lbxszn bmxwt   
## 0 0 0 0 0 0 0 0   
## bmxht bmxbmi bmxwaist riagendr ridageyr ridreth1   
## 0 0 0 0 0 0

model.2.cv<-cv.glmnet(x.train, glucose.train, alpha=0.5)  
plot(model.2.cv)



model.2.cv$lambda.min

## [1] 0.1733026

model.2.cv$lambda.1se

## [1] 9.466267

model.2.train.final<-glmnet(x.train, glucose.train, alpha=0.5, lambda=model.1.cv$lambda.min)  
coef(model.2.train.final)

## 15 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## (Intercept) 87.4874928543  
## x .   
## seqn 0.0001408574  
## lbxscu -0.0154609763  
## lbxsse 0.0702467118  
## lbxmg -0.2205433793  
## lbxlyc 1.1464028002  
## lbxszn -0.0474222372  
## bmxwt .   
## bmxht -0.1753242810  
## bmxbmi .   
## bmxwaist 0.2303053593  
## riagendr -4.0438239450  
## ridageyr 0.1588633637  
## ridreth1 .

model.2.test.pred<-model.2.train.final %>% predict(x.test) %>% as.vector()  
data.frame(RMSE=RMSE(model.2.test.pred, glucose.test), RSQ=R2(model.2.test.pred, glucose.test))

## RMSE RSQ  
## 1 16.59604 0.1149167

I then fitted a support vector model to compare to the other two models to see which model performed the best. The support vector machine used the largest cost of 100 that shows there is a lot of misclassiofcation in in this model.

set.seed (100)  
svm.glucose<-svm(lbxsgl ~ ., data=train.data.glucose, kernel="linear", cost=1, scale=TRUE)  
print(svm.glucose)

##   
## Call:  
## svm(formula = lbxsgl ~ ., data = train.data.glucose, kernel = "linear",   
## cost = 1, scale = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: linear   
## cost: 1   
## gamma: 0.07142857   
## epsilon: 0.1   
##   
##   
## Number of Support Vectors: 2005

svm.pred<-predict(svm.glucose, newdata=train.data.glucose)  
  
rmse(svm.glucose, test.data.glucose)

## [1] 16.78708

features<-x.train  
outcome<-train.data.glucose$lbxsgl  
  
svm\_tune <- tune(svm, train.x=features, train.y=outcome, kernel="linear", range=list(cost=10^(-1:2)))  
  
summary(svm\_tune)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 100  
##   
## - best performance: 352.6775   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 0.1 353.0315 141.0055  
## 2 1.0 352.8538 140.7250  
## 3 10.0 352.7076 140.6454  
## 4 100.0 352.6775 140.6988

svm.glucose.new<-svm(lbxsgl ~ ., data=train.data.glucose, kernel="linear", cost= 100, scale=TRUE)  
  
print(svm.glucose.new)

##   
## Call:  
## svm(formula = lbxsgl ~ ., data = train.data.glucose, kernel = "linear",   
## cost = 100, scale = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: linear   
## cost: 100   
## gamma: 0.07142857   
## epsilon: 0.1   
##   
##   
## Number of Support Vectors: 2006

svm.pred.new<-predict(svm.glucose.new, newdata=test.data.glucose)  
  
rmse(svm.glucose.new, test.data.glucose)

## [1] 16.79641

Based on these three alogorithms root mean square errors I would say that the best model to go with to further this analysis would be the LASSO regularized regression because it gave the smallest root mean squared error.

# Part 3: Ethical considerations of data-driven analyses in social epidemiology

The following is an excerpt from an article on the ethical tensions in using social media data to characterize individuals’ and communities’ mental health. After reading the brief excerpt, address both questions listed below. The response (to both questions) should be limited to one page.

“Powered by machine learning techniques, social media provides an unobtrusive lens into individual behaviors,emotions, and psychological states. Recent research has successfully employed social media data to predictmental health states of individuals, ranging from the presence and severity of mental disorders like depression to the risk of suicide. These algorithmic inferences hold great potential in supporting early detection and treatment of mental disorders and in the design of interventions. At the same time, the outcomes of this research can pose great risk…”

### Question 1: Describe one potential risk to either individuals, communities or specific populations that could arise from research or public health practice that utilizes data-driven analyses of social-media data.

One potential risk for individuals who use social media that could arise from using research or public health practices that utilizes data-driven ananlyses of social media data would be a violations of privacy. People who use social media and use the platforms that are being used ti run theses algorithms should be aware of what is happening. Although using social media could be beneficial to creating algorithms it could also be harmful if the data is not properly stored. If there are multiple platforms that an individual is using and those data are merged together to form a full data set then data that was once de-indentified might become identifiable if the the researchers are not careful. Every individual has a right to their privacy and by using social media to run data driven analysis could be a potential breach to the individual and put them at a higher risk.

### Question 2: Describe one potential safeguard that could be implemented to prevent the risk you describe.

One safegaurd that could be implemented to protect individuals right to privacy is by setting guiedliens with HIPPA. If guidleiens are in place then it would be a way to protect the individuals right to privacy. There would have to be a system that regulates a way to ensure that the data remains de-identifiable. The patients’ privacy should be one of the first concerns that researchers should be thinking about when dealing with big data from social media sources. HIPPA could set these guidelines across the board and ensure privacy for everyone while also getting information to make these predictions.