Appendix 2: Code

# Handling Missing Data + Cleaning

library(naniar)

se = read.csv("2016 School Explorer.csv", header = TRUE)

#Convert 'N/A' strings to NA

se[ se == "N/A" ] = NA

#Remove dollar signs

se$School.Income.Estimate = as.numeric(gsub('[$,]', '', se$School.Income.Estimate))

#Removing Percent symbols

percent\_cols =dplyr::select(se,Percent.Asian,Percent.Black,Percent.Black...Hispanic,Percent.ELL,Percent.Hispanic,Percent.of.Students.Chronically.Absent,Percent.White,Supportive.Environment..,Rigorous.Instruction..,Collaborative.Teachers..,Effective.School.Leadership..,Trust..,Strong.Family.Community.Ties..,Student.Attendance.Rate)

percent\_cols = apply(percent\_cols, 2, function(y) as.numeric(gsub("%", "", y)))

se$Percent.Asian = percent\_cols[, 1]

se$Percent.Black = percent\_cols[, 2]

se$Percent.Black...Hispanic = percent\_cols[, 3]

se$Percent.ELL = percent\_cols[, 4]

se$Percent.Hispanic = percent\_cols[, 5]

se$Percent.of.Students.Chronically.Absent = percent\_cols[, 6]

se$Percent.White = percent\_cols[, 7]

se$Supportive.Environment.. = percent\_cols[, 8]

se$Rigorous.Instruction.. = percent\_cols[, 9]

se$Collaborative.Teachers.. = percent\_cols[, 10]

se$Effective.School.Leadership.. = percent\_cols[, 11]

se$Trust.. = percent\_cols[, 12]

se$Strong.Family.Community.Ties.. = percent\_cols[, 13]

se$Student.Attendance.Rate = percent\_cols[, 14]

# after doing a colSums(is.na(se)), I found that there were over 100 rows with missing values across the following 4 columns, so I removed them.

se = se[!is.na(se$Rigorous.Instruction.Rating) ,]

se = se[!is.na(se$Student.Achievement.Rating) ,]

se = se[!is.na(se$Trust.Rating) ,]

se = se[!is.na(se$Supportive.Environment.Rating) ,]

# Here I handle all na values in School income by assigning them the average income of their corresponding district.

Districts = se$District

means = aggregate(.~Districts, data=se, mean)

income\_means = means$School.Income.Estimate

income = se$School.Income.Estimate

d = se$District

for (x in 1:length(income))

if (is.na(income[x]))

income[x] = income\_means[d[x]]

se$School.Income.Estimate = income

#converting decimal data from <fctr> to <dbl>

se$Economic.Need.Index = as.numeric(as.character(se$Economic.Need.Index))

se$Average.ELA.Proficiency = as.numeric(as.character(se$Average.ELA.Proficiency))

se$Average.Math.Proficiency = as.numeric(as.character(se$Average.Math.Proficiency))

#Creating dummy variables for categorical data

se$Community.School.= as.numeric(se$Community.School.)

se$Rigorous.Instruction.Rating = as.numeric(se$Rigorous.Instruction.Rating)

se$Collaborative.Teachers.Rating = as.numeric(se$Collaborative.Teachers.Rating)

se$Supportive.Environment.Rating = as.numeric(se$Supportive.Environment.Rating)

se$Effective.School.Leadership.Rating = as.numeric(se$Effective.School.Leadership.Rating)

se$Strong.Family.Community.Ties.Rating = as.numeric(se$Strong.Family.Community.Ties.Rating)

se$Trust.Rating = as.numeric(se$Trust.Rating)

se$Student.Achievement.Rating = as.numeric(se$Student.Achievement.Rating)

#now all of the missing values have been handled, thanks mostly to Will for working all this out and making the data workable!!!

# CCA Code

# I am performing a CCA with school demographic information (income + ethnic makeup) as the dependent variables and school quality information (ratings + test scores) as the independent variables.

Demographic = se[, c(13,14,15,17,18,19,21)]

Quality = se[, c(24,26,28,30,32,34,37,38)]

cca1 = cca(Demographic,Quality,xscale = TRUE, yscale = TRUE)

helio.plot(cca1, cv = 1, xvlab = cca1$xlab, yvlab = cca1$ylab,

x.name = "", y.name = "", lab.cex = 0.50,

wid.fact = 0.75, main = "Demographics Quality",

sub = paste("Test Scores", sep = ""), zero.rad = 30,

range.rad = 20, name.padding = 1, name.cex = 1.0,

axis.circ = c(-1, 1), x.group = rep(0, dim(cca1$xstructcorr)[1]),

y.group = rep(0, dim(cca1$ystructcorr)[1]), type = "correlation")

helio.plot(cca1, cv = 2, xvlab = cca1$xlab, yvlab = cca1$ylab,

x.name = "", y.name = "", lab.cex = 0.5,

wid.fact = 0.75, main = "Demographics Quality",

sub = paste("Community Involvement", sep = ""), zero.rad = 30,

range.rad = 20, name.padding = 1, name.cex = 1.0,

axis.circ = c(-1, 1), x.group = rep(0, dim(cca1$xstructcorr)[1]),

y.group = rep(0, dim(cca1$ystructcorr)[1]), type = "correlation")

helio.plot(cca1, cv = 3, xvlab = cca1$xlab, yvlab = cca1$ylab,

x.name = "", y.name = "", lab.cex = 0.5,

wid.fact = 0.75, main = "Demographics Quality",

sub = paste("Supportive Environment", sep = ""), zero.rad = 30,

range.rad = 20, name.padding = 1, name.cex = 1.0,

axis.circ = c(-1, 1), x.group = rep(0, dim(cca1$xstructcorr)[1]),

y.group = rep(0, dim(cca1$ystructcorr)[1]), type = "correlation")

helio.plot(cca1, cv = 4, xvlab = cca1$xlab, yvlab = cca1$ylab,

x.name = "", y.name = "", lab.cex = 0.5,

wid.fact = 0.75, main = "Demographics Quality",

sub = paste("Teachers", sep = ""), zero.rad = 30,

range.rad = 20, name.padding = 1, name.cex = 1.0,

axis.circ = c(-1, 1), x.group = rep(0, dim(cca1$xstructcorr)[1]),

y.group = rep(0, dim(cca1$ystructcorr)[1]), type = "correlation")

#summary of CCA analysis

summary(cca1)

#structural loadings on x variate

cca1$xstructcorr[, c(1,2,3,4)]

#Structural loadings on y variate

cca1$ystructcorr[, c(1,2,3,4)]

#We can look at the redundancies

cca1$xvrd

cca1$yvrd

#Looking at the coefficients for the variables within each variate.

cca1$xcoef[, c(1,2,3,4)]

cca1$ycoef[, c(1,2,3,4)]

# Clustering Code

library(rlang) # data manipulation

library(cluster) # clustering algorithms

library(factoextra) # clustering algorithms & visualization

library(dplyr)

library(MASS)

library(readxl)

# Here I used the missing data + cleaning that Will worked out

# imported data is named se

rownames(se) <- make.names(se[,1], unique = TRUE)

se[,1] <- NULL

str(se)

#data exploration

pairs(~School.Income.Estimate+ Economic.Need.Index + Grade.3.ELA...All.Students.Tested + Grade.3.Math...All.Students.tested + Grade.5.ELA...All.Students.Tested + Grade.5.Math...All.Students.Tested,data=se,

main="Scatterplot Matrix")

pairs(~School.Income.Estimate+ Economic.Need.Index + Districts + Average.ELA.Proficiency+Average.Math.Proficiency ,data=se,

main="Scatterplot Matrix")

#Choose the data variables we want to use for the clustering

Schools = se[c(14,36,37)]

str(Schools)

# Let's first scale the data by dividing each variable by its standard deviation:

std <- sapply(Schools, sd, na.rm = TRUE) # finding standard deviations of variables

head(std)

Schools.std <- sweep(Schools, 2, std, '/')

Schools.std[is.na(Schools.std)]<- 0

head(Schools.std)

str(Schools.std)

######################

# how many clusters? #

######################

fviz\_nbclust(Schools, kmeans,

method = "gap\_stat")

fviz\_nbclust(Schools, kmeans, method = "wss")

fviz\_nbclust(Schools, kmeans, method = "silhouette")

# A K-means clustering with k = 2:

Schools.k2 <- kmeans(Schools.std, centers=2, iter.max=100, nstart=25)

Schools.k2

# Visualize

fviz\_cluster(Schools.k2, data = Schools,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal())

# A K-means clustering with k = 3:

Schools.k3 <- kmeans(Schools.std, centers=3, iter.max=100, nstart=25)

Schools.k3

# Visualize

fviz\_cluster(Schools.k3, data = Schools,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal())

# A K-means clustering with k = 4:

Schools.k4 <- kmeans(Schools.std, centers=4, iter.max=100, nstart=25)

Schools.k4

# Visualize

fviz\_cluster(Schools.k4, data = Schools,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal())

# A K-means clustering with k = 5:

Schools.k5 <- kmeans(Schools.std, centers=5, iter.max=100, nstart=25)

Schools.k5

# Visualize

fviz\_cluster(Schools.k5, data = Schools,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal())

Schools.k5$betweenss/Schools.k5$totss

Schools.k4$betweenss/Schools.k4$totss

Schools.k3$betweenss/Schools.k3$totss

Schools.k2$betweenss/Schools.k2$totss

#choose k = 2

Schools %>%

as\_tibble() %>%

mutate(cluster = Schools.k2$cluster,

state = row.names(Schools)) %>%

ggplot(aes(Average.ELA.Proficiency, Average.Math.Proficiency, color = factor(cluster), label = state)) +

geom\_text() + ggtitle("Schools in two clusters") +

xlab("Average ELA Scores") + ylab("Average Math Scores") + labs(color = "Cluster")

Schools %>%

as\_tibble() %>%

mutate(cluster = Schools.k2$cluster,

state = row.names(Schools)) %>%

ggplot(aes(School.Income.Estimate, Average.Math.Proficiency, color = factor(cluster), label = state)) +

geom\_text()+ ggtitle("Schools in two clusters") +

xlab("School Income Estimate") + ylab("Average Math Scores") + labs(color = "Cluster")

Schools %>%

as\_tibble() %>%

mutate(cluster = Schools.k2$cluster,

state = row.names(Schools)) %>%

ggplot(aes(School.Income.Estimate, Average.ELA.Proficiency, color = factor(cluster), label = state)) +

geom\_text()+ ggtitle("Schools in two clusters") +

xlab("School Income Estimate") + ylab("Average ELA Scores") + labs(color = "Cluster")

################################

# exploring the two clusters separately #

################################

# add cluster number to dataset

groups = Schools.k2$cluster

se$group <- groups

#explore based on cluster

plot( se$District, se$group,main="Scatterplot cluster districts",

ylab="Cluster", xlab="School District")

pairs(~se$group + se$Percent.Asian + se$Percent.Black + se$Percent.Black...Hispanic + se$Percent.ELL + se$Percent.Hispanic + se$Percent.White)

pairs(~se$group + se$Student.Attendance.Rate + se$Trust.. + se$Strong.Family.Community.Ties.. + se$Effective.School.Leadership.. + se$Collaborative.Teachers.. + se$Rigorous.Instruction..)

pairs(~se$Trust.. + se$Strong.Family.Community.Ties.. + se$Rigorous.Instruction.. + se$Percent.ELL)

i = 1

BinomialGroup =vector (,1167)

for (c in se$group){

if (se$group[i] == 2){

BinomialGroup[i] = 0

i = i + 1

} else {

BinomialGroup[i] = 1

i = i + 1

}

}

BinomialGroup

mylogit <- glm(BinomialGroup ~ Percent.Asian + Percent.ELL + Percent.Hispanic + Percent.White + Student.Attendance.Rate + Strong.Family.Community.Ties.. + Rigorous.Instruction.. + Percent.ELL, data = se, family = "binomial")

summary(mylogit)

## odds ratios and 95% CI

exp(cbind(OR = coef(mylogit), confint(mylogit)))

######################

# separate clusters #

######################

cluster1 <- se[ which(se$group=='1'), ]

cluster2 <- se[ which(se$group=='2'), ]

newC1 <- cluster1[c(31,23,15,16,18,20,21)]

head(newC1)

newC2 <- cluster2[c(31,23,15,16,18,20,21)]

head(newC2)

# Let's first scale the data by dividing each variable by its standard deviation:

std1 <- sapply(newC1, sd, na.rm = TRUE) # finding standard deviations of variables

head(std1)

std2 <- sapply(newC2, sd, na.rm = TRUE) # finding standard deviations of variables

head(std2)

######################

# choose clusters for C1 #

######################

C1.std <- sweep(newC1, 2, std1, '/')

C1.std[is.na(C1.std)]<- 0

head(C1.std)

str(C1.std)

# how many clusters?

my.data.matrix <- C1.std

my.k.choices <- 2:8

n <- length(my.data.matrix[,1])

wss1 <- (n-1)\*sum(apply(my.data.matrix,2,var))

wss <- numeric(0)

for(i in my.k.choices) {

W <- sum(kmeans(my.data.matrix,i)$withinss)

wss <- c(wss,W)

}

wss <- c(wss1,wss)

plot(c(1,my.k.choices),wss,type='l',xlab='Number of clusters', ylab='Within-groups sum-of-squares', lwd=2)

### A little function to calculate the average silhouette width

### for a variety of choices of k:

my.k.choices <- 2:8

avg.sil.width <- rep(0, times=length(my.k.choices))

for (ii in (1:length(my.k.choices)) ){

avg.sil.width[ii] <- pam(C1.std, k=my.k.choices[ii])$silinfo$avg.width

}

print( cbind(my.k.choices, avg.sil.width) )

# A LARGE average silhouette width indicates that the observations are properly clustered.

######################

# choose clusters for C2 #

######################

#cluster 2

C2.std <- sweep(newC2, 2, std2, '/')

C2.std[is.na(C2.std)]<- 0

head(C2.std)

str(C2.std)

# how many clusters?

my.data.matrix <- C2.std

my.k.choices <- 2:8

n <- length(my.data.matrix[,1])

wss1 <- (n-1)\*sum(apply(my.data.matrix,2,var))

wss <- numeric(0)

for(i in my.k.choices) {

W <- sum(kmeans(my.data.matrix,i)$withinss)

wss <- c(wss,W)

}

wss <- c(wss1,wss)

plot(c(1,my.k.choices),wss,type='l',xlab='Number of clusters', ylab='Within-groups sum-of-squares', lwd=2)

### A little function to calculate the average silhouette width

### for a variety of choices of k:

my.k.choices <- 2:8

avg.sil.width <- rep(0, times=length(my.k.choices))

for (ii in (1:length(my.k.choices)) ){

avg.sil.width[ii] <- pam(C2.std, k=my.k.choices[ii])$silinfo$avg.width

}

print( cbind(my.k.choices, avg.sil.width) )

# A LARGE average silhouette width indicates that the observations are properly clustered.

######################

# C2 final #

######################

C2.kmed.3 <- pam(C2.std, k=3, diss=F)

C2.kmed.3

C2.kmed.3$clustering # printing the "clustering vector"

C2.kmed.3$silinfo$avg.width #printing the average silhouette width

C2.3.clust <- lapply(1:3, function(nc) row.names(se)[C1.kmed.3$clustering==nc])

C2.3.clust # printing the clusters in terms of the car names

############# Visualization of Clusters:

## Built-in plots available with the pam function:

# The "clusplot":

plot(C2.kmed.3, which.plots=1, main = "High Scoring Cluster Plot (PAM) where k = 3")

# The "silhouette plot":

plot(C2.kmed.3, which.plots=2)

# This shows which observations are "best clustered."

plot(newC2, col=C2.kmed.3$cluster, cex=.2, cex.labels=.8, gap = .2, labels = colnames(newC2) , main = "Relationship between 3 clusters with high test scores")

######################

# C1 final #

######################

#now do the low values cluster

C1.kmed.5 <- pam(C1.std, k=5, diss=F)

C1.kmed.5

C1.kmed.5$clustering # printing the "clustering vector"

C1.kmed.5$silinfo$avg.width #printing the average silhouette width

C1.5.clust <- lapply(1:3, function(nc) row.names(se)[C2.kmed.5$clustering==nc])

C1.5.clust # printing the clusters in terms of the car names

############# Visualization of Clusters:

## Built-in plots available with the pam function:

# The "clusplot":

plot(C1.kmed.5, which.plots=1, main = "")

title(main = "Low Scoring Cluster Plot (PAM) where k = 5")

# The "silhouette plot":

plot(C1.kmed.5, which.plots=2)

# This shows which observations are "best clustered."

plot(newC1, col=C1.kmed.5$cluster, cex=.2, cex.labels=.5, gap = .2, labels = colnames(newC1), main = "Relationship between 5 clusters with low test scores")

######################

# end vis #

######################

# Visualize

fviz\_cluster(C1.kmed.5, data = newC1,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal(),

title = "Low Scoring Cluster Plot (PAM) where k = 5")

fviz\_cluster(C2.kmed.3, data = newC2,

ellipse.type = "convex",

palette = "jco",

repel = TRUE,

geom = "point",

ggtheme = theme\_minimal(),

title = "High Scoring Cluster Plot (PAM) where k = 3")

# LDA Code

```{r setup, include=FALSE}  
knitr::opts\_chunk$set(echo = TRUE)  
library(MASS)  
library(psych)  
library(devtools)  
install\_github('fawda123/ggord')  
library(ggord)  
```  
#Import the Data  
```{r import and select data}  
  
#The data was imported from Excel after the data pre-processing step was completed.  
df = readxl::read\_xlsx("C:/Users/O7010/OneDrive/DSC424\_Fall2018/Final Project/nyds2.xlsx", sheet = 1)  
  
#Pullout the rows for the analysis and display a portion of the data.  
df2 = df[c(17,27,39,35,40,41)]  
head(df2)  
```  
#Plot the Data   
\*\*Plot the data to show the scatterplots, distribution, and correlations between the pairs of numeric variables.\*\*  
```{r warning=FALSE}  
pairs.panels(df2[-c(3)], gap = 0, hist.col="cyan", bg = c("red", "green", "blue", "yellow", "orange")[df2$`Effective School Leadership Rating`], pch = 21)  
```  
#Partition the Data  
\*\*Because the data is over 1,000 rows, create training and validation sets.\*\*  
```{r create training and testing sets}  
#Set a seed to get the same results each time.The split is 70/30.  
set.seed(224)  
ind = sample(2,nrow(df2), replace = TRUE, prob = c(0.7,0.3))  
tdata = df2[ind==1,]  
vdata = df2[ind==2,]  
```  
#Run Linear Discriminant Analysis on Training Data  
```{r begin analysis on training data}  
tdata.lda = lda(tdata$`Student Achievement Rating` ~ ., data = tdata)  
tdata.lda  
```  
```{r}  
plot(tdata.lda, gap = 0, col = c("red", "green", "blue"), main = "Training Data LDA Plot")  
round((tdata.lda$scaling),2)  
tdata.lda$counts  
```  
#Predict the Rating and Show LDA Histogram  
```{r}  
#Try to predict the class from the original data  
p = predict(tdata.lda, newdata=tdata[,1:6])$class  
tpredict = predict(tdata.lda, tdata)  
ldahist(data = tpredict$x[,1], g = tdata$`Student Achievement Rating`, main = "Histogram of Student Achievement Rating", col = "orange", ylim = c(0,.2,.4,.6,.8))  
```  
#Create a Bi-plot  
```{r, warning=FALSE}  
ggord(tdata.lda,tdata$`Student Achievement Rating`, xlim = c(-10,10))  
```  
#Create the Confusion Matrix  
\*\*Compare the results of the prediction using the Confusion Matrix.\*\*  
```{r}  
tconf = table(p, tdata$`Student Achievement Rating`)  
tconf  
sum(diag(tconf))/sum(tconf)  
```  
#Run the Model on the Validation Set  
```{r}  
p2 = predict(tdata.lda, newdata = vdata[,1:6])$class  
vconf = table(p2, vdata$`Student Achievement Rating`)  
vconf  
sum(diag(vconf))/sum(vconf)  
```

# Multiple Regression Code

MyDataNumeric = df[, c(1, 3:19)]  
head(MyDataNumeric)  
  
#Check data types  
str(MyDataNumeric)  
  
#Compute the correlation matrix and visualize it  
cor.df = cor(MyDataNumeric)  
cor.df  
corrplot(cor.df, method="ellipse")  
corrplot(cor.df, method="number")  
corrplot(cor.df, method="color", col=brewer.pal(n=8, name="Blues"))  
  
#Full model  
fullFit = lm(Economic.Need.Index ~ ., data=MyDataNumeric)  
summary(fullFit)  
autoplot(fullFit)  
  
#Checking for Influential points and outliers  
cooksd <- cooks.distance(fullFit)  
plot(cooksd, pch="\*", cex=2, main="Influential Obs by Cooks distance") # plot cook's distance  
abline(h = 4\*mean(cooksd, na.rm=T), col="red") #   
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4\*mean(cooksd, na.rm=T),names(cooksd),""), col="red")  
  
car::outlierTest(fullFit)  
  
#Removing Observations  
MyDataNumeric.Out<-MyDataNumeric[-c(32,44,1001),]  
str(MyDataNumeric)  
str(MyDataNumeric.Out)  
  
#Split dataset and fit model  
set.seed(121)  
samplev = sample.split(MyDataNumeric.Out$Economic.Need.Index, SplitRatio= 0.80)  
train = subset(MyDataNumeric.Out, samplev == TRUE)  
test = subset(MyDataNumeric.Out, samplev == FALSE)  
  
#Stepwise  
null = lm(Economic.Need.Index ~ 1, data=train)  
null  
full = lm(Economic.Need.Index ~ ., data=train)  
full  
MyDataStep = step(null, scope = list(upper=full), direction="both")  
summary(MyDataStep)  
vif(MyDataStep)  
  
#Drop high VIF scores  
fit1 <- lm(Economic.Need.Index ~ School.Income.Estimate + Percent.White + Percent.of.Students.Chronically.Absent + Student.Attendance.Rate + Collaborative.Teachers + Strong.Family.Community.Ties + Percent.Hispanic+ Percent.ELL + Percent.Black.Hispanic + Average.Math.Proficiency , data=train)  
summary(fit1)  
vif(fit1)  
  
#Drop high VIF scores  
fit <- lm(Economic.Need.Index ~ School.Income.Estimate + Percent.White + Student.Attendance.Rate + Strong.Family.Community.Ties + Percent.Hispanic+ Percent.ELL + Average.Math.Proficiency , data=train)  
summary(fit)  
vif(fit)  
autoplot(fit)  
  
#Model Validation  
ENIPred <- predict(fit,test)  
actuals\_preds <- data.frame(cbind(actuals=test$Economic.Need.Index,predicteds=ENIPred))  
correlation\_accuracy <- cor(actuals\_preds)  
correlation\_accuracy  
  
head(actuals\_preds)  
  
min\_max\_accuracy <- mean(apply(actuals\_preds, 1, min) / apply(actuals\_preds, 1, max))   
min\_max\_accuracy  
mape <- mean(abs((actuals\_preds$predicteds - actuals\_preds$actuals))/actuals\_preds$actuals)  
mape

# PCA Code

require(dplyr)

require(stats)

require(factoextra)

library(FactoMineR)

require(psych)

require(corrplot)

# ---- C2a ----

df <- read.csv("C:\\Users\\jmoy001\\Documents\\DePaul\\CSC424\\Jeff\\Project\\2016 School Explorer.csv")

#set cells of N/A as NA

df[df=="N/A"] <- NA

colnames(df)

test <- df %>%

mutate(School.Income.Estimate=as.numeric(gsub('[$,]', '', .$School.Income.Estimate))) %>%

mutate(Percent.ELL= as.numeric(gsub('[%]', '', .$Percent.ELL))) %>%

mutate(Percent.Asian= as.numeric(gsub('[%]', '', .$Percent.Asian))) %>%

mutate(Percent.Black= as.numeric(gsub('[%]', '', .$Percent.Black))) %>%

mutate(Percent.Hispanic= as.numeric(gsub('[%]', '', .$Percent.Hispanic))) %>%

mutate(Percent.Black...Hispanic= as.numeric(gsub('[%]', '', .$Percent.Black...Hispanic))) %>%

mutate(Percent.White= as.numeric(gsub('[%]', '', .$Percent.White))) %>%

mutate(Student.Attendance.Rate= as.numeric(gsub('[%]', '', .$Student.Attendance.Rate))) %>%

mutate(Percent.of.Students.Chronically.Absent= as.numeric(gsub('[%]', '', .$Percent.of.Students.Chronically.Absent))) %>%

mutate(Rigorous.Instruction..= as.numeric(gsub('[%]', '', .$Rigorous.Instruction..))) %>%

mutate(Collaborative.Teachers..= as.numeric(gsub('[%]', '', .$Collaborative.Teachers..))) %>%

mutate(Supportive.Environment..= as.numeric(gsub('[%]', '', .$Supportive.Environment..))) %>%

mutate(Effective.School.Leadership..= as.numeric(gsub('[%]', '', .$Effective.School.Leadership..))) %>%

mutate(Strong.Family.Community.Ties..= as.numeric(gsub('[%]', '', .$Strong.Family.Community.Ties..))) %>%

mutate(Trust..= as.numeric(gsub('[%]', '', .$Trust..))) %>%

mutate(Collaborative.Teachers..= as.numeric(gsub('[%]', '', .$Collaborative.Teachers..)))

apply(df,2, function(y) as.numeric)

class(df$Trust..)

#works if you select the columns in a separate variable

#maybe <-apply(df, 2, function(y) as.numeric(gsub('[$%,]', '', y)))

test <- df %>%

mutate(School.Income.Estimate=as.numeric(gsub('[$,]', '', .$School.Income.Estimate))) %>%

mutate(Percent.ELL= as.numeric(gsub('[%]', '', .$Percent.ELL))) %>%

mutate(Percent.Asian= as.numeric(gsub('[%]', '', .$Percent.Asian))) %>%

mutate(Percent.Black= as.numeric(gsub('[%]', '', .$Percent.Black))) %>%

mutate(Percent.Hispanic= as.numeric(gsub('[%]', '', .$Percent.Hispanic))) %>%

mutate(Percent.Black...Hispanic= as.numeric(gsub('[%]', '', .$Percent.Black...Hispanic))) %>%

mutate(Percent.White= as.numeric(gsub('[%]', '', .$Percent.White))) %>%

mutate(Percent.of.Student.Attendance.Rate= as.numeric(gsub('[%]', '', .$Student.Attendance.Rate))) %>%

mutate(Percent.of.Students.Chronically.Absent= as.numeric(gsub('[%]', '', .$Percent.of.Students.Chronically.Absent))) %>%

mutate(Percent.of.Rigorous.Instruction..= as.numeric(gsub('[%]', '', .$Rigorous.Instruction..))) %>%

mutate(Percent.of.Collaborative.Teachers..= as.numeric(gsub('[%]', '', .$Collaborative.Teachers..))) %>%

mutate(Percent.of.Supportive.Environment..= as.numeric(gsub('[%]', '', .$Supportive.Environment..))) %>%

mutate(Percent.of.Effective.School.Leadership..= as.numeric(gsub('[%]', '', .$Effective.School.Leadership..))) %>%

mutate(Percent.of.Strong.Family.Community.Ties..= as.numeric(gsub('[%]', '', .$Strong.Family.Community.Ties..))) %>%

mutate(Percent.of.Trust..= as.numeric(gsub('[%]', '', .$Trust..))) %>%

mutate(Percent.of.Collaborative.Teachers..= as.numeric(gsub('[%]', '', .$Collaborative.Teachers..))) %>%

#will remove variables, all new variables added from above with Percent.of string added will be added after the last original variables

select(-c(Student.Attendance.Rate,

Rigorous.Instruction..,

Collaborative.Teachers..,

Supportive.Environment..,

Effective.School.Leadership..,

Strong.Family.Community.Ties..,

Trust..,

Collaborative.Teachers..))

#use code below to see all variables, truncated otherwise

str(test, list.len= 200)

#total

sum(is.na(df))

sum(is.na(df$Economic.Need.Index))

sapply(df, function(x) sum(is.na(x)))

test <- df %>%

select(starts\_with("Grade"))

#Remove Grades, Grade Low, Grade high

#Grades shows all grades the school has data on

#Grade low is the lowest grade taught, Grade high is the highest grade taught

test <- test[,-c(1,2,3)]

#remove columns of sum total

final <- test %>%

select(-contains("All"))

#check correlation sum

corrplot(cor(final), method="number")

#both plots are similar to measure the correlation. In terms of PCA though, it does make a difference scaling the data to determine the number of components for the model.

#seems that there are only two variables that are highly correlated. Let's now see what the p values are for each bi-variate relationship to test for significance

MCorrTest = corr.test(final, adjust="none")

MCorrTest

M = MCorrTest$p

#remember, testing p values whether independent variables are significant to each other!

MTest = ifelse(M < .01, T, F)

# Now lets see how many significant correlations there are for each variable. We can do this by summing the columns of the matrix

# minus 1 due to correlation on diagonal

colSums(MTest) - 1

colnames(final)

#how to check for Factorability: Steps to test if PCA can be done with this data.

#http://minato.sip21c.org/swtips/factor-in-R.pdf

kmo <- function(x)

{

x <- subset(x, complete.cases(x)) # Omit missing values

r <- cor(x) # Correlation matrix

r2 <- r^2 # Squared correlation coefficients

i <- solve(r) # Inverse matrix of correlation matrix

d <- diag(i) # Diagonal elements of inverse matrix

p2 <- (-i/sqrt(outer(d, d)))^2 # Squared partial correlation coefficients

diag(r2) <- diag(p2) <- 0 # Delete diagonal elements

KMO <- sum(r2)/(sum(r2)+sum(p2))

MSA <- colSums(r2)/(colSums(r2)+colSums(p2))

return(list(KMO=KMO, MSA=MSA))

}

kmo(final)

#KMO tests the sampling adequacy to verify that the data has a large enough sample. The overall test provided a value of .81, which is greater than .70. Thus, the data meets the sampling adequacy.

Bartlett.sphericity.test <- function(x)

{

method <- "Bartlett's test of sphericity"

data.name <- deparse(substitute(x))

x <- subset(x, complete.cases(x)) # Omit missing values

n <- nrow(x)

p <- ncol(x)

chisq <- (1-n+(2\*p+5)/6)\*log(det(cor(x)))

df <- p\*(p-1)/2

p.value <- pchisq(chisq, df, lower.tail=FALSE)

names(chisq) <- "X-squared"

names(df) <- "df"

return(structure(list(statistic=chisq, parameter=df, p.value=p.value,

method=method, data.name=data.name), class="htest"))

}

Bartlett.sphericity.test(final)

#Barlett's test of Sphericity tests if there are enough shared variance within the data set. With the p-value being less than .05, there is sufficient variance in the data to run a PCA.

psych::alpha(final, check.keys= TRUE)

#Cronbach's Alpha is used to test the reliability, which means do the variables and the groupings make sense (theory and conceptually) and can be used for later analysis.

#With the reliability value at .9, it shows that the groupings of the variables is sufficient enough for PCA

#PCA

PCA\_2a <- prcomp(final, center=T, scale=T)

plot(PCA\_2a, main="Screeplot of Eigenvalue", xlab="Components")

abline(1,0)

#it seems that the first 10 components have an eigen value greater than 1. Let's now look at the cumulative proportion of variance that each component has.

summary(PCA\_2a)

# with 9 components, 62.2% of the variance is explained. Lastly, lets look at the knee of the plot.

pCA\_2A1 <- PCA(final, graph = FALSE)

fviz\_eig(pCA\_2A1, addlabels = TRUE, ylim = c(0, 35), ncp=30)

#It looks like after 6 components, the plot levels off.

# ---- C2b ----

pca\_2b <- psych::principal(final, rotate="varimax", nfactors=6)

pca\_2b

print(pca\_2b$loadings, cutoff=.44, sort=T)

#It seems that RC1 has Hispanic or Latino grouped with Asian or Pacitic Islander. Perhaps we can increase the groupings to see if more components would separate the grade and ethnicity for each variable.

#increase components to 8

pca\_2b <- psych::principal(final, rotate="varimax", nfactors=8)

print(pca\_2b$loadings, cutoff=.46, sort=T)

#this gives some pretty interesting results. Although some variables did not get placed into a component due to small loadings.

# ---- C2b1 ----

#Prof follow up question:

#re-run PCA when removing variables that had low loadings, will we get the same result?

final2 <- final

final2$Grade.3.ELA.4s...American.Indian.or.Alaska.Native <- NULL

final2$Grade.3.ELA.4s...Limited.English.Proficient<- NULL

final2$Grade.3.Math.4s...American.Indian.or.Alaska.Native <- NULL

final2$Grade.4.ELA.4s...Limited.English.Proficient <- NULL

final2$Grade.5.ELA.4s...Multiracial <- NULL

final2$Grade.5.ELA.4s...Limited.English.Proficient <- NULL

final2$Grade.5.Math.4s...Multiracial <- NULL

final2$Grade.6.ELA.4s...Limited.English.Proficient <- NULL

final2$Grade.7.ELA.4s...Limited.English.Proficient <- NULL

final2$Grade.8.ELA.4s...Limited.English.Proficient <- NULL

final2$Grade.8.Math.4s...American.Indian.or.Alaska.Native <- NULL

pca\_2ba <- psych::principal(final2, rotate="varimax", nfactors=8)

print(pca\_2ba$loadings, cutoff=.46, sort=T)

#Variables are placed in the same component! The beta coefficients changed a bit but still the same component grouping. This provides us with a better model as all variables were placed into a component.

# ---- C2b2 ----

#Prof follow up question:

#add visualization of top influential variable for each component

plot\_df <-data.frame(Component = c("RC1", "RC5", "RC2", "RC4", "RC3", "RC9", "RC7", "RC6", "RC8"), Loading = c(.845, .889, .948, .927, .846, .849, .865, .921, .941), Class=c("Grade\_6\_Math\_Asian", "Grade\_7\_Math\_Asian", "Grade\_5\_Math\_Asian", "Grade\_4\_ELA\_White", "Grade\_4\_Math\_Black", "Grade\_6\_Math\_Hispanic\_Latino", "Grade\_4\_ELA\_Hispanic\_Latino", "Grade\_6\_ELA\_Indian", "Grade\_5\_Math\_Indian"))

plot <- ggplot(plot\_df, aes(x=Component, y=Loading, fill=Class)) + geom\_bar(stat='identity') + labs(title="Most Influential Variable per Component", x="Components", y="Beta Coefficient") + theme(plot.title = element\_text(hjust=.5))

plot <- plot + theme(legend.position="bottom")

plot(plot)

# ---- C2c ----

#ensure that varimax was the appropriate rotation.

round(cor(pca\_2b$scores),2)

corrplot(cor(pca\_2b$scores), method = "number")

# 