Alexandra/Lexie's Code

```
#cleaning file from initial dataset
#written by alexandra degrandchamp
#libraries used
library(tidyverse)
library(naniar)

#importing data, first peeks
summary(X2015)

#joining in states (copied from pdf as separate .txt file)
states_clean <- states %>%
          select(Value,Value_1)
states_clean <- na.omit(states_clean)
states_clean$Value <- as.double(states_clean$Value)

dataset2015 <- X2015 %>%
         left_join(states_clean, by = c('_STATE' = 'Value'))

#cleaning date values
dataset2015$IDATE <- str_sub(dataset2015$IDATE,start = 3,-1)
dataset2015$IDATE <- mdy(dataset2015$IDATE)
dataset2015$IMONTH <- month(dataset2015$IDATE)
dataset2015$IDAY <- day(dataset2015$IDATE)
dataset2015$IYEAR <- year(dataset2015$IDATE)

#gathering only complete rows
completedInterviews <- dataset2015 %>%
            filter(DISPCODE == 1100)

#taking a look at some intro question dispersions
summary(completedInterviews$CTELENUM)
summary(completedInterviews$PVTRESD1)
summary(completedInterviews$COLGHOUS)

#cleaning up and factorizing state values
completedInterviews <- completedInterviews %>%
            rename('StateText' = Value_1)
```

```
completedInterviews$StateText <- as.factor(completedInterviews$StateText)

#checking for empty columns
summary(completedInterviews)

#not all qs asked in 2015
interviewsNoBlankColumns <- completedInterviews %>%
                    select(-c('COLGHOUS','LADULT','PAINACT2',
                          'QLMENTL2','QLSTRES2','QLHLTH2',
                          'ASERVIST','ASDRVIST'))


#looking for incomplete cases based on survey flow
testingState <- interviewsNoBlankColumns %>%
            filter(STATERES != 1)
rm(testingState) #0 results
testingCell <- interviewsNoBlankColumns %>%
            filter(CELLFON3 == 2)
rm(testingCell) #doesn't actually terminate survey

#testing cell a second time
testingCell2 <- interviewsNoBlankColumns %>%
            filter(CELLFON3 == 2 & CELLFON2 == 2)
rm(testingCell2) #0 results

#rounding some funky looking should-be-integers-but-aren't
interviewsNoBlankColumns$NUMADULT <-
as.integer(interviewsNoBlankColumns$NUMADULT)
interviewsNoBlankColumns$NUMMEN <- as.integer(interviewsNoBlankColumns$NUMMEN)
interviewsNoBlankColumns$NUMWOMEN <-
as.integer(interviewsNoBlankColumns$NUMWOMEN)

#"duplicate" column cleanup
#this dataset aggregates the land-line and cell-phone surveys, with different variables for each
#this merges columns together or removes them if they are irrelevant

#CTELENUM - land line ask if number is correct
#CTELNUM1 - cell phone ask if number is correct
summary(interviewsNoBlankColumns$CTELENUM)
summary(interviewsNoBlankColumns$CTELNUM1)
```

```
#these are irrelevant columns

deDupe <- interviewsNoBlankColumns %>%
        select(-c('CTELENUM','CTELNUM1'))

#PVTRESD1 - land line ask if number is private residence
#PVTRESD2 - cell phone ask if number is private residence
summary(deDupe$PVTRESD1)
summary(deDupe$PVTRESD2)

deDupe2 <- deDupe %>%
        unite('PVTRES',c('PVTRESD1','PVTRESD2'), na.rm = TRUE)
deDupe2$PVTRES <- as.integer(deDupe2$PVTRES)
summary(deDupe2$PVTRES)

#COLGHOUS - land line ask if number is college housing, removed from set
#CCLGHOUS - cell phone ask if number is college housing, kept in set
#no changes here just noting the duplication has been handled

#NUMADULT - land line ask for number of adults in household
#HHADULT - cell phone ask for number of adults in household
summary(deDupe2$NUMADULT)
summary(deDupe2$HHADULT)

deDupe3 <- deDupe2 %>%
        unite('NUMADULTS',c('NUMADULT','HHADULT'), na.rm = TRUE)
#getting NAs when converting to numeric, so seeing what is up there
deDupe3$NUMADULTS <- as.factor(deDupe3$NUMADULTS)
numAdultsCheck <- deDupe3 %>%
                group_by(NUMADULTS) %>%
                summarize(n = n())
#further cleaning - 77 is don't know, 99 is refused to answer - will turn those to N/As for now
#will also turn '' into N/A
deDupe3 <- deDupe3 %>%
        replace_with_na(replace = list(NUMADULTS = c('77','99','')))
#there's also one row with a 4_3, which means on landline someone said 4
#and the same someone said 3 on a cellphone
#that shouldn't happen either, but deleting it is giving me trouble, so changing it to NA by
coercion
deDupe3$NUMADULTS <- as.character(deDupe3$NUMADULTS)
```

```
deDupe3$NUMADULTS <- as.integer(deDupe3$NUMADULTS)
summary(deDupe3$NUMADULTS)

#id or duplicated columns - will create subset without
irrelevantColumns = c('IMONTH','IDAY','IYEAR','DISPCODE','FMONTH','_STATE',
'STATERES','CELLFON3',
                    'CELLFON2')
#note: keeping SEQNO and PSU to join back together if data is separated

relevantData <- deDupe3 %>%
               select(-all_of(irrelevantColumns))

#starting on actual questions cleaning
#GENHLTH - categorical where 9 is refused
cleaningVars <- relevantData %>%
               replace_with_na(replace = list(GENHLTH = 9))
cleaningVars$GENHLTH <- as.factor(cleaningVars$GENHLTH)

#PHYSHLTH - replacing 88 with 0, 77 and 99 with N/A
cleaningVars$PHYSHLTH[cleaningVars$PHYSHLTH == 88] <- 0
cleaningVars <- cleaningVars %>%
               replace_with_na(replace = list(PHYSHLTH = c(77,99)))

#MENTHLTH - replacing 88 with 0, 77 and 99 with N/A
cleaningVars$MENTHLTH[cleaningVars$MENTHLTH == 88] <- 0
cleaningVars <- cleaningVars %>%
               replace_with_na(replace = list(MENTHLTH = c(77,99)))

#POORHLTH - replacing 88 with 0, 77 and 99 with N/A
#note: skipped q if 0 days in either PHYSHLTH or MENTHLTH
cleaningVars$POORHLTH[cleaningVars$POORHLTH == 88] <- 0
cleaningVars <- cleaningVars %>%
               replace_with_na(replace = list(POORHLTH = c(77,99)))

#HLTHPLN1 - changing 7 and 9 with N/A. This is a categorical variable
cleaningVars <- cleaningVars %>%
               replace_with_na(replace = list(HLTHPLN1 = c(7,9)))
cleaningVars$HLTHPLN1 <- as.factor(cleaningVars$HLTHPLN1)

#PERSDOC2 - changing 9 with N/A. This is a categorical variable
```

```
cleaningVars <- cleaningVars %>%
            replace_with_na(replace = list(PERSDOC2 = c(9)))
cleaningVars$PERSDOC2 <- as.factor(cleaningVars$PERSDOC2)

#MEDCOST - changing 9 with N/A. This is a categorical variable
cleaningVars <- cleaningVars %>%
            replace_with_na(replace = list(MEDCOST = c(9)))
cleaningVars$MEDCOST <- as.factor(cleaningVars$MEDCOST)

#CHECKUP1 - changing 9 to N/A. This is a categorical variable
cleaningVars <- cleaningVars %>%
            replace_with_na(replace = list(CHECKUP1 = c(9)))
cleaningVars$MEDCOST <- as.factor(cleaningVars$MEDCOST)

#this is about the point where I realized this was going to take 100 years this way
#switching it up
#read through the codebook and put each variable in either quant or cat
#I also classified weighting/coding variables and identified calculated fields separately (for
multicollinearity concerns)
#IDATE, SEQNO, _PSU kept in each set as identifiers

quantVars <- cleaningVars %>%
        select(IDATE, SEQNO, `_PSU`, NUMADULTS, NUMMEN, NUMWOMEN,
PHYSHLTH,
              MENTHLTH, POORHLTH, DIABAGE2, NUMPHON2, CHILDREN, WEIGHT2,
              HEIGHT3, ALCDAY5, AVEDRNK2, DRNK3GE5, MAXDRNKS, FRUITJU1,
              FRUIT1, FVBEANS, FVGREEN, FVORANG, VEGETAB1, EXEROFT1,
EXERHMM1, EXEROFT2,
              EXERHMM2, STRENGTH, BLDSUGAR, FEETCHK2, DOCTDIAB,
CHKHEMO3, FEETCHK, LONGWTCH,
              ASTHMAGE, ASRCHKUP, ASACTLIM, SCNTWRK1, SCNTLWK1,
ADPLEASR, ADDOWN, ADSLEEP,
              ADENERGY, ADEAT1, ADFAIL, ADTHINK, ADMOVE)

catVars <- cleaningVars %>%
        select(IDATE, SEQNO, `_PSU`, StateText, PVTRES, CCLGHOUS, CSTATE,
LANDLINE,
              GENHLTH, HLTHPLN1, PERSDOC2, MEDCOST, CHECKUP1, BPHIGH4,
BPMEDS,
```

BLOODCHO, CHOLCHK, TOLDHI2, CVDINFR4, CVDCRHD4, CVDSTRK3, ASTHMA3,
ASTHNOW, CHCSCNCR, CHCOCNCR, CHCCOPD1, HAVARTH3, ADDEPEV2,
CHCKIDNY, DIABETE3, SEX, MARITAL, EDUCA, RENTHOM1, NUMHHOL2,
CPDEMO1, VETERAN3, EMPLOY1, INCOME2, INTERNET, PREGNANT, QLACTLM2,
USEEQUIP, BLIND, DECIDE, DIFFWALK, DIFFDRES, DIFFALON, SMOKE100,
SMOKDAY2, STOPSMK2, LASTSMK2, USENOW3, EXERANY2, EXRACT11, EXRACT21,
LMTJOIN3, ARTHDIS2, ARTHSOCL, JOINPAIN, SEATBELT, FLUSHOT6, FLSHTMY2, IMFVPLAC,
PNEUVAC3, HIVTST6, HIVTSTD3, WHRTST10, PDIABTST, PREDIAB1, INSULIN,
EYEEXAM, DIABEYE, DIABEDU, CAREGIV1, CRGVREL1, CRGVLNG1, CRGVHRS1,
CRGVPRB1, CRGVPERS, CRGVHOUS, CRGVMST2, CRGVEXPT, VIDFCLT2, VIREDIF3,
VIPRFVS2, VINOCRE2, VIEYEXM2, VIINSUR2, VICTRCT4, VIGLUMA2, VIMACDG2,
CIMEMLOS, CDHOUSE, CDASSIST, CDHELP, CDSOCIAL, CDDISCUS, WTCHSALT,
DRADVISE, ASATTACK, ASYMPTOM, ASNOSLEP, ASTHMED3, ASINHALR, HAREHAB1,
STREHAB1, CVDASPRN, ASPUNSAF, RLIVPAIN, RDUCHART, RDUCSTRK, ARTTODAY,
ARTHWGT, ARTHEXER, ARTHEDU, TETANUS, HPVADVC2, HPVADSHT, SHINGLE2,HADMAM,
HOWLONG, HADPAP2, LASTPAP2, HPVTEST, HPLSTTST, HADHYST2, PROFEXAM,LENGEXAM,
BLDSTOOL, LSTBLDS3, HADSIGM3, HADSGCO1, LASTSIG3, PCPSAAD2, PCPSADI1,
PCPSARE1, PSATEST1, PSATIME, PCPSARS1, PCPSADE1, PCDMDECN, SCNTMNY1,
SCNTMEL1, SCNTPAID, SCNTLPAD, SXORIENT, TRNSGNDR, RCSGENDR, RCSRLTN2, CASTHDX2,
CASTHNO2, EMTSUPRT, LSATISFY, MISTMNT, ADANXEV, QSTVER, QSTLANG, EXACTOT1,

```
                EXACTOT2, `_CHISPNC`, `_CRACE1`, `_CPRACE`, `_DUALUSE`)

codingVars <- cleaningVars %>%
            select(IDATE, SEQNO, `_PSU`, MSCODE, `_STSTR`, `_STRWT`, `_RAWRAKE`,
                 `_WT2RAKE`, `_CLLCPWT`, `_DUALCOR`, `_LLCPWT`)

calcVars <- cleaningVars %>%
          select(IDATE, SEQNO, `_PSU`, `_RFHLTH`, `_HCVU651`, `_RFHYPE5`,
`_CHOLCHK`,
                `_RFCHOL`, `_MICHD`, `_LTASTH1`, `_CASTHM1`, `_ASTHMS1`,
`_DRDXAR1`,
                `_PRACE1`, `_MRACE1`, `_HISPANC`, `_RACE`, `_RACEG21`, `_RACEGR3`,
                `_RACE_G1`, `_AGEG5YR`, `_AGE65YR`, `_AGE80`, `_AGE_G`, HTIN4,
HTM4,
                WTKG3, `_BMI5`, `_BMI5CAT`, `_RFBMI5`, `_CHLDCNT`, `_EDUCAG`,
`_INCOMG`,
                `_SMOKER3`, `_RFSMOK3`, DRNKANY5, DROCDY3_, `_RFBING5`,
`_DRNKWEK`,
                `_RFDRHV5`, FTJUDA1_, FRUTDA1_, BEANDAY_, GRENDAY_,
ORNGDAY_, VEGEDA1_,
                `_MISFRTN`, `_MISVEGN`, `_FRTRESP`, `_VEGRESP`, `_FRUTSUM`,
`_VEGESUM`,
                `_FRTLT1`, `_VEGLT1`, `_FRT16`, `_VEG23`, `_FRUITEX`, `_VEGETEX`,
                `_TOTINDA`, METVL11_, METVL21_, MAXVO2_, FC60_, ACTIN11_,
ACTIN21_,
                PADUR1_, PADUR2_, PAFREQ1_, PAFREQ2_, `_MINAC11`, `_MINAC21`,
STRFREQ_,
                PAMISS1_, PAMIN11_, PAMIN21_, PA1MIN_, PAVIG11_, PAVIG21_,
PA1VIGM_,
                `_PACAT1`, `_PAINDX1`, `_PA150R2`, `_PA300R2`, `_PA30021`, `_PASTRNG`,
                `_PAREC1`, `_PASTAE1`, `_LMTACT1`, `_LMTWRK1`, `_LMTSCL1`,
`_RFSEAT2`,
                `_RFSEAT3`, `_FLSHOT6`, `_PNEUMO2`, `_AIDTST3`)


#exporting clean data set and subsets to csv files
setwd('/Users/alexandradegrandchamp/Documents/GradSchool/DSC424/Final
Project/FinalistDatasets/BRFSS')
cwd <- getwd()
fullFN <- 'cleanedFullSet.csv'
```

```
write.csv(cleaningVars, paste(cwd,fullFN,sep='/'), row.names = FALSE)

quantFN <- 'quantData.csv'
catFN <- 'categoricalData.csv'
codingFN <- 'codingVariables.csv'
calcFN <- 'calculatedFields.csv'

write.csv(quantVars, paste(cwd,quantFN,sep='/'), row.names = FALSE)
write.csv(catVars, paste(cwd,catFN,sep='/'), row.names = FALSE)
write.csv(codingVars, paste(cwd,codingFN,sep='/'), row.names = FALSE)
write.csv(calcVars, paste(cwd,calcFN,sep='/'), row.names = FALSE)
```

```
#cleaning file for clustering and women’s health
#written by alexandra degrandchamp

#research question: what are the clusters of behaviors women exhibit when it comes to
preventative health?

library(readr)
cleanedFullSet <- read_csv("Documents/GradSchool/DSC424/Final
Project/FinalistDatasets/BRFSS/cleanedFullSet.csv")

library(tidyverse)

codVars <- c('MSCODE','_STSTR','_STRWT','_RAWRAKE','_WT2RAKE','_CLLCPWT',
          '_DUALCOR','_LLCPWT')

allVars <- cleanedFullSet %>%
         select(-all_of(codVars))

#breast/cervical cancer - PAP/HPV test and other behaviors
#filter for women
#include variables in modules 12, 14, 15

womensHealth <- allVars %>%
              filter(SEX == 2)


relevantVars <- c('HPVADVC2','HPVADSHT','HPVTEST',
              'HADPAP2','HADMAM','HOWLONG',
              'LASTPAP2','HPLSTTST','HADHYST2',
              'PROFEXAM','LENGEXAM','GENHLTH',
              'PHYSHLTH','MENTHLTH','POORHLTH',
              'HLTHPLN1','MEDCOST','CHECKUP1',
              'EDUCA','EMPLOY1','CHILDREN',
              'INCOME2','INTERNET','WEIGHT2',
              'HTIN4','PREGNANT','SMOKE100',
              'SMOKDAY2','USENOW3','ALCDAY5',
              'AVEDRNK2','DRNK3GE5','MAXDRNKS',
              '_FRUTSUM','_VEGESUM','STRENGTH',
              'EXEROFT1','EXERHMM1','EXEROFT2',
              'EXERHMM2','MAXVO2_','FC60_',
                'ACTIN11_','ACTIN21_','_PAINDX1',
                'SEATBELT','FLUSHOT6','HIVTST6',
                'CAREGIV1','SCNTWRK1','SXORIENT',
                'TRNSGNDR','EMTSUPRT','ADDOWN',
                'ADSLEEP','ADPLEASR','ADENERGY',
                'ADEAT1', '_AGEG5YR')

limitedVars <- womensHealth %>%
              select(all_of(relevantVars))

#combining HADPAP2 and HPVTEST for measure of 1 or the other
summary(limitedVars$HADPAP2)
summary(limitedVars$HPVTEST)
summary(limitedVars)

cleaningVars <- limitedVars %>%
                mutate(HADHPVORPAP = case_when(HPVTEST == 1 | HADPAP2 == 1 ~ 1,
                                HPVTEST == 2 | HADPAP2 == 2 ~ 2,
                                HPVTEST == 7 | HADPAP2 == 7 ~ 99,
                                HPVTEST == 9 | HADPAP2 == 9 ~ 99,
                                .default = 99))
summary(cleaningVars$HADHPVORPAP)

combinedTest <- cleaningVars %>%
                #filter(HPVTEST == 1 | HADPAP2 == 1) %>%
                mutate(HPVTEST2 = ifelse(HPVTEST != 1,0,1),
                    HADPAP22 = ifelse(HADPAP2 != 1,0,1)) %>%
                group_by(HADHPVORPAP, HPVTEST, HADPAP2) %>%
                summarize(n = n(),
                        sumHPV = sum(HPVTEST2),
                        sumPap = sum(HADPAP22))

#eliminating all 99s from data set
testSet <- cleaningVars %>%
            filter(HADHPVORPAP != 99) %>%
            select(-c(HPVTEST, HADPAP2))

summary(testSet)

#whittling variables based on %N/A - any variable with more than 11k NAs should go
```

```
highlyNAVars <-
c('HPVADVC2','HPVADSHT','HPLSTTST','PROFEXAM','LENGEXAM','PREGNANT',

'SMOKDAY2','AVEDRNK2','DRNK3GE5','MAXDRNKS','EXEROFT2','EXERHMM2',
             'SCNTWRK1','SXORIENT','TRNSGNDR','EMTSUPRT','ADDOWN','ADSLEEP',
             'ADPLEASR','ADENERGY','ADEAT1')

testSet2 <- testSet %>%
          select(-all_of(highlyNAVars))
summary(testSet2)

#need to clean individual variables - can't do no missing cases, so will need to be case by case
summary(testSet2)

#HOWLONG: time since last mammogram within last X years (if val = 1-3). 4 is last 5,
#5 is more than 5. 7 is don't know, 9 is refused, NA is missing.
testSet2 %>% select(HOWLONG) %>% group_by(HOWLONG) %>% summarise(n = n())

testSet2$HOWLONG <- replace(testSet2$HOWLONG, is.na(testSet2$HOWLONG), 0)
testSet2$HOWLONG <- replace(testSet2$HOWLONG, testSet2$HOWLONG==9, 0)
testSet2$HOWLONG <- replace(testSet2$HOWLONG, testSet2$HOWLONG==7, 0)
testSet2$HOWLONG <- replace(testSet2$HOWLONG, testSet2$HOWLONG==5, 6)
testSet2$HOWLONG <- replace(testSet2$HOWLONG, testSet2$HOWLONG==4, 5)
summary(testSet2$HOWLONG)

#LASTPAP2: same scale as HOWLONG, will clean the same
testSet2 %>% select(LASTPAP2) %>% group_by(LASTPAP2) %>% summarise(n = n())
testSet2$LASTPAP2 <- replace(testSet2$LASTPAP2, is.na(testSet2$LASTPAP2), 0)
testSet2$LASTPAP2 <- replace(testSet2$LASTPAP2, testSet2$LASTPAP2==9, 0)
testSet2$LASTPAP2 <- replace(testSet2$LASTPAP2, testSet2$LASTPAP2==7, 0)
testSet2$LASTPAP2 <- replace(testSet2$LASTPAP2, testSet2$LASTPAP2==5, 6)
testSet2$LASTPAP2 <- replace(testSet2$LASTPAP2, testSet2$LASTPAP2==4, 5)
summary(testSet2$LASTPAP2)

#HADHYST2: 1 is yes, 2 is no. Would like to keep as binary. Few refusals
#coding refusals as "no hysterectomy"
testSet2 %>% select(HADHYST2) %>% group_by(HADHYST2) %>% summarise(n = n())
testSet2$HADHYST2 <- replace(testSet2$HADHYST2, is.na(testSet2$HADHYST2), 2)
testSet2$HADHYST2 <- replace(testSet2$HADHYST2, testSet2$HADHYST2==9, 2)
testSet2$HADHYST2 <- replace(testSet2$HADHYST2, testSet2$HADHYST2==7, 2)
```

```
#GENHLTH: categorical with 5 levels. Only a handful rows with N/A or don't know, will
remove
testSet2 %>% select(GENHLTH) %>% group_by(GENHLTH) %>% summarise(n = n())
testSet3 <- testSet2 %>%
        filter(GENHLTH != 7,
           !is.na(GENHLTH))
testSet3 %>% select(GENHLTH) %>% group_by(GENHLTH) %>% summarise(n = n())

#PHYSHLTH: quant variable with values 1-30 as valid (X days in 30 physical health not good)
#88 is none, 77 is don't know, 99 refused
#none of those three remaining
#460 NAs - a bit too many to remove, so will code 0 (median val)
summary(testSet3$PHYSHLTH)
testSet3$PHYSHLTH <- replace(testSet3$PHYSHLTH, is.na(testSet3$PHYSHLTH),0)

#MENTHLTH: quant variable, same scale as PHYSHLTH
#coding NAs to median of 0
summary(testSet3$MENTHLTH)
testSet3$MENTHLTH <- replace(testSet3$MENTHLTH, is.na(testSet3$MENTHLTH),0)

#POORHLTH: same scale as PHYS and MENT, skipped if 0s for previous
#will replace NA with 0 here
testSet3$POORHLTH <- replace(testSet3$POORHLTH, is.na(testSet3$POORHLTH),0)

#HLTHPLN1: binary variable about existence of health insurance
#want to be a true binary; N/As are going into yes (overwhelming majority)
testSet3 %>% select(HLTHPLN1) %>% group_by(HLTHPLN1) %>% summarize(n = n())
testSet3$HLTHPLN1 <- replace(testSet3$HLTHPLN1,is.na(testSet3$HLTHPLN1),1)

#MEDCOST - binary variable asking if doctor was prohibitive due to cost
#want to make this a true binary - don't knows and not sures to be removed
testSet3 %>% select(MEDCOST) %>% group_by(MEDCOST) %>% summarize(n = n())
testSet4 <- testSet3 %>%
        filter(MEDCOST != 7,
           !is.na(MEDCOST))
testSet4 %>% select(MEDCOST) %>% group_by(MEDCOST) %>% summarize(n = n())

#CHECKUP1: similar scale as other 1-4 qs.
#1 and 2 are within X years ago, 3 is within 5 years ago (switch to 5)
```

```
#4 is more than 5 years ago (switch to 6)
#7 is don't know, 8 is never. 4 NAs
testSet4 %>% select(CHECKUP1) %>% group_by(CHECKUP1) %>% summarize(n = n())
testSet4$CHECKUP1 <- replace(testSet4$CHECKUP1, is.na(testSet4$CHECKUP1),0)
testSet4$CHECKUP1 <- replace(testSet4$CHECKUP1, testSet4$CHECKUP1 == 8,0)
testSet4$CHECKUP1 <- replace(testSet4$CHECKUP1, testSet4$CHECKUP1 == 7,0)
testSet4$CHECKUP1 <- replace(testSet4$CHECKUP1, testSet4$CHECKUP1 == 4,6)
testSet4$CHECKUP1 <- replace(testSet4$CHECKUP1, testSet4$CHECKUP1 == 3,5)
summary(testSet4$CHECKUP1)

#EDUCA: categorical variable with 1 = no school, 2 = elementary, 3 = high school (some),
#4 = high school graduate, 5 = some college, 6 = college grad/beyond. 9 = refused
#going to remove the 64 9s here
testSet4 %>% select(EDUCA) %>% group_by(EDUCA) %>% summarize(n = n())
testSet5 <- testSet4 %>% filter(EDUCA != 9)

#EMPLOY1: categorical variable: 1 = employed, 2 = self-employed, 3 = out of work 1+ yrs,
#4 = out of work < 1 year, 5 = homemaker, 6 = student, 7 = retired, 8 = unable to work
#9 = refused
#I want to combine 3 and 4 into just "out of work" == 3
#and I will remove NAs here as well
testSet5 %>% select(EMPLOY1) %>% group_by(EMPLOY1) %>% summarize(n = n())
testSet6 <- testSet5 %>% filter(EMPLOY1 != 9)
testSet6$EMPLOY1 <- replace(testSet6$EMPLOY1, testSet6$EMPLOY1 == 4, 3)
testSet6 %>% select(EMPLOY1) %>% group_by(EMPLOY1) %>% summarize(n = n())

#CHILDREN: quantitative variable, number of children in house
kids <- testSet6 %>% select(CHILDREN) %>% group_by(CHILDREN) %>% summarize(n =
n())
#88 is 0, 99 is refused - will code both to 0 (0 is median)
testSet6$CHILDREN <- replace(testSet6$CHILDREN, testSet6$CHILDREN == 88, 0)
summary(testSet6$CHILDREN)
testSet6$CHILDREN <- replace(testSet6$CHILDREN, testSet6$CHILDREN == 99, 0)

#INCOME2: categorical variable representing income brackets on household income
#levels are <10k, <15k, <20k, <25k, <35k, <50k, <75k, >=75k
testSet6 %>% select(INCOME2) %>% group_by(INCOME2) %>% summarize(n = n())
#this is highly categorical and I'm kicking myself for not using _INCOMG instead
#will calculate _INCOMG and remove INCOME2
#_INCOMG will have 5 levels: <15k, <25k, <35k, <50k, 50k+
#there are a lot of NAs here so will replace with median after re-grouping

testSet7 <- testSet6 %>%
        mutate(INCOMG = case_when(INCOME2 == 1 | INCOME2 == 2 ~ 1,
                        INCOME2 == 3 | INCOME2 == 4 ~ 2,
                        INCOME2 == 5 ~ 3,
                        INCOME2 == 6 ~ 4,
                        INCOME2 == 7 | INCOME2 == 8 ~ 5,
                        .default = 99)) %>%
        select(-INCOME2)

testSet7 %>% filter(INCOMG != 99) %>% summarize(med = median(INCOMG))
#median value is 4, will replace 99s with 4
testSet7$INCOMG <- replace(testSet7$INCOMG, testSet7$INCOMG == 99,4)

#INTERNET: Binary variable about internet usage in past 30 days
#making a true binary, moving 7s to no and 9s to yes
testSet7 %>% select(INTERNET) %>% group_by(INTERNET) %>% summarise(n = n())
testSet7$INTERNET <- replace(testSet7$INTERNET, testSet7$INTERNET == 7, 2)
testSet7$INTERNET <- replace(testSet7$INTERNET, testSet7$INTERNET == 9, 1)

#WEIGHT2: 'weight without shoes", measured in either pounds or kgs, whichever was provided
#going to convert to pounds, about 1320 need conversion from kgs or from a null value
#will convert missing to 0 and find median of remaining (non-0 values), then sub missing with
median
testSet7 %>% filter(WEIGHT2 >= 9000) %>% summarise(n = n())
testSet7 <- testSet7 %>%
        mutate(WEIGHTLBS = case_when(WEIGHT2 == 9999 ~ 0,
                        WEIGHT2 == 7777 ~ 0,
                        WEIGHT2 < 9000 ~ WEIGHT2,
                        .default = (WEIGHT2 - 9000) * 2.20462))
testSet7 %>% select(WEIGHTLBS) %>% filter(WEIGHTLBS != 0) %>% summarise(med =
median(WEIGHTLBS))
testSet7$WEIGHTLBS <- replace(testSet7$WEIGHTLBS, testSet7$WEIGHTLBS == 0, 150)
testSet7 <- testSet7 %>% select(-WEIGHT2)
summary(testSet7$WEIGHTLBS)
#confirmed there are several outliers of sizable weight, incl. 700 pounds, in original data

#HTIN4: height in inches (this is a cheating calc field)
summary(testSet7$HTIN4)
```

```
#converting NAs to median
testSet7 %>% filter(!is.na(HTIN4)) %>% summarize(med = median(HTIN4))
testSet7$HTIN4 <- replace(testSet7$HTIN4, is.na(testSet7$HTIN4),64)

#SMOKE100: binary - have you smoked 100 cigarettes/5 packs in your life?
#7 and 9s are few, converting them to 2s
testSet7 %>% group_by(SMOKE100) %>% summarise(n = n())
testSet7$SMOKE100 <- replace(testSet7$SMOKE100, testSet7$SMOKE100 == 7, 2)
testSet7$SMOKE100 <- replace(testSet7$SMOKE100, testSet7$SMOKE100 == 9, 2)

#I actually didn't realize USENOW3 was just smokeless tobacco and not all forms of tobacco
#this is incredibly limited so I'm going to remove the variable
testSet8 <- testSet7 %>%
        select(-USENOW3)

#ALCDAY5: quantitative variable that is days per week OR days in the last month with drinking
#converting missing, don't know, and never to 0
#converting "in last month" to itself minus prefix
#converting "per week" to itself minus prefix * 4 weeks
#resulting column will be days in last 30 days
alc <- testSet9 %>%
      group_by(ALCDAYSCLEAN) %>%
      summarize(n = n())
testSet9 <- testSet8 %>%
        mutate(ALCDAYSCLEAN = case_when(ALCDAY5 == 999 ~ 0,
                                  ALCDAY5 == 888 ~ 0,
                                  ALCDAY5 == 777 ~ 0,
                                  ALCDAY5 > 200 ~ ALCDAY5 - 200,
                                  .default = (ALCDAY5 - 100) * 4)) %>%
        select(-ALCDAY5)

#need a new df name to make autocorrect easier lol
#_FRUTSUM: calculated fruits per day
#this needs to be converted down to the two decimal places - dividing by 100
summary(testSet9$`_FRUTSUM`)
newTestSet <- testSet9 %>%
         mutate(SUMFRUIT = `_FRUTSUM`/100) %>%
         select(-`_FRUTSUM`)
summary(newTestSet$SUMFRUIT)
#will convert NAs to median value
```

```
newTestSet %>% filter(!is.na(SUMFRUIT)) %>% summarise(med = median(SUMFRUIT))
newTestSet$SUMFRUIT <- replace(newTestSet$SUMFRUIT,
is.na(newTestSet$SUMFRUIT),1.07)
#note: there's an outlier here with someone who's chomping on 109 fruits per day

#_VEGESUM: calculated veg per day
#same conversion as _FRUTSUM
newTestSet <- newTestSet %>%
        mutate(SUMVEG = `_VEGESUM`/100) %>%
        select(-`_VEGESUM`)
summary(newTestSet$SUMVEG)
newTestSet %>% filter(!is.na(SUMVEG)) %>% summarise(med = median(SUMVEG))
newTestSet$SUMVEG <- replace(newTestSet$SUMVEG, is.na(newTestSet$SUMVEG),1.86)

#STRENGTH:how many times per day/week do you do strength exercise?
#needs converting like drinks
strength <- newTestSet2 %>%
       group_by(STRENCLEAN) %>%
       summarize(n = n())
newTestSet2 <- newTestSet %>%
         mutate(STRENCLEAN = case_when(STRENGTH == 999 ~ 0,
                        STRENGTH == 888 ~ 0,
                        STRENGTH == 777 ~ 0,
                        STRENGTH > 200 ~ STRENGTH - 200,
                        .default = (STRENGTH - 100) * 4)) %>%
         select(-STRENGTH)
summary(newTestSet2$STRENCLEAN)
#there are now a handful of outliers that I'm going to convert to 30;
#I think this is multiple times per day
#should be considered equivalent to number of days in a month that exercise is performed

newTestSet2$STRENCLEAN <- replace(newTestSet2$STRENCLEAN,
newTestSet2$STRENCLEAN == 240, 30)
newTestSet2$STRENCLEAN <- replace(newTestSet2$STRENCLEAN,
newTestSet2$STRENCLEAN == 396, 30)
newTestSet2$STRENCLEAN <- replace(newTestSet2$STRENCLEAN,
newTestSet2$STRENCLEAN > 30, 30)

#EXEROFT1: times per week or month doing the single physical activity performed most
#will clean similar to STRENGTH and ALCDAY5
```

```
exercise <- newTestSet3 %>%
        group_by(EXERTIMES) %>%
        summarise(n = n())

#there are outliers here that are multiple times per day, so will round those down to 30
#like alcohol and strength, want this to be number of days in a month
newTestSet3 <- newTestSet2 %>%
         mutate(EXERTIMES = case_when(EXEROFT1 == 999 ~ 0,
                         is.na(EXEROFT1) ~ 0,
                         EXEROFT1 == 777 ~ 0,
                         EXEROFT1 > 230 ~ 30,
                         EXEROFT1 > 200 ~ EXEROFT1 - 200,
                         EXEROFT1 > 107 ~ 30,
                         .default = (EXEROFT1 - 100) * 4)) %>%
         select(-EXEROFT1)


#I don't understand the coding on EXERHMM1 so removing
newTestSet4 <- newTestSet3 %>% select(-EXERHMM1)

#MAXVO2_: estimated maximum oxygen consumption
#needs missing values recoded and needs to be divided by 100
#will code missing values to the median
newTestSet4 <- newTestSet4 %>%
         mutate(MAXVO2 = MAXVO2_/100)

o2 <- newTestSet4 %>% group_by(MAXVO2) %>% summarise(n = n())
newTestSet4 %>% filter(MAXVO2 != 999) %>% summarise(med = median(MAXVO2))
newTestSet4$MAXVO2 <- replace(newTestSet4$MAXVO2, newTestSet4$MAXVO2 == 999,
26.2)
newTestSet4 <- newTestSet4 %>% select(-MAXVO2_)

#FC60: estimated functional capacity in two decimal places
#like MAXVO2, will divide by 100 and code any missing values to the median
newTestSet4 <- newTestSet4 %>%
         mutate(FUNCCAP = FC60_/100)
summary(newTestSet4$FUNCCAP)
newTestSet4 %>% filter(FUNCCAP != 999) %>% summarise(med = median(FUNCCAP))
newTestSet4$FUNCCAP <- replace(newTestSet4$FUNCCAP, newTestSet4$FUNCCAP ==
999, 4.49)
```

```
newTestSet4 <- newTestSet4 %>% select(-FC60_)

#to simplify the analysis, I removed the "second activity" variables
#however, ACTIN11_ and ACTIN21_ classify both activities as moderate, vigorous, or neither
#I want to combine these into a categorical variable: moderate activity, vigorous activity, or no
activity
#1 = moderate, 2 = vigorous, 0 = none, blank = missing
newTestSet4$ACTIN11_ <- as.integer(newTestSet4$ACTIN11_)
newTestSet4$ACTIN21_ <- as.integer(newTestSet4$ACTIN21_)

summary(newTestSet4$ACTIN11_)
summary(newTestSet4$ACTIN21_)


newTestSet5 <- newTestSet4 %>%
        mutate(ACTIVLVL = case_when(ACTIN11_ == 2 | ACTIN21_ == 2 ~ 2,
                       ACTIN11_ == 1 | ACTIN21_ == 1 ~ 1,
                       ACTIN11_ == 0 ~ 0,
                       ACTIN21_ == 0 ~ 0,
                       .default = 99))

activTest <- newTestSet5 %>%
         mutate(Activity1 = ifelse(ACTIN11_ != 1 | ACTIN11_ != 2,0,1),
            Activity2 = ifelse(ACTIN21_ != 1 | ACTIN21_ != 2,0,1)) %>%
         group_by(ACTIVLVL, ACTIN11_, ACTIN21_) %>%
         summarize(n = n(),
              sumAct1 = sum(Activity1),
              sumAct2 = sum(Activity2))
#final clean: convert 99 to 0, remove ACTIN11_ and ACTIN21_
newTestSet5$ACTIVLVL <- replace(newTestSet5$ACTIVLVL, newTestSet5$ACTIVLVL ==
99, 0)
newTestSet5 <- newTestSet5 %>% select(-ACTIN11_)
newTestSet5 <- newTestSet5 %>% select(-ACTIN21_)

#_PAINDX1: calculated binary variable, want to clean to true binary
#measure of physical activity index - 1 is meets aerobic recs, 2 is does not
#there are about 800 9s in this group, but the split is quite close between 1 and 2
#I'll move 9s where activity level is 0 and 1 to "did not meet"
#9s where activity level is 2 to "did meet"
newTestSet5 %>% group_by(ACTIVLVL, `_PAINDX1`) %>% summarize(n = n())
```

```
newTestSet6 <- newTestSet5 %>%
        mutate(AERORECS = case_when(ACTIVLVL == 0 & `_PAINDX1` == 9 ~ 2,
                        ACTIVLVL == 2 & `_PAINDX1` == 9 ~ 1,
                        ACTIVLVL == 1 & `_PAINDX1` == 9 ~ 2,
                        .default = `_PAINDX1`)) %>%
        select(-`_PAINDX1`)

summary(newTestSet6$AERORECS)
newTestSet6 %>% group_by(AERORECS) %>% summarize(n = n())

#SEATBELT: how often do you use a seatbelt? Categorical variable with 5 levels
#7 is don't know, 8 is "never ride in a car", 9 is refused
#there are less than 100 responses with 7,8, or 9 - will just remove
summary(newTestSet6$SEATBELT)
newTestSet6 %>% group_by(SEATBELT) %>% summarise(n = n())

newTestSet6 <- newTestSet6 %>%
        filter(SEATBELT < 6)

#FLUSHOT6 - binary variable, have you had a flu vaccine in last 12 months?
#want to turn into a true binary - will move "don't know" to no,
#will remove 8 rows without response
summary(newTestSet6$FLUSHOT6)
newTestSet6 %>% group_by(FLUSHOT6) %>% summarise(n = n())

newTestSet6$FLUSHOT6 <- replace(newTestSet6$FLUSHOT6, newTestSet6$FLUSHOT6 ==
7,2)
newTestSet6 <- newTestSet6 %>%
        filter(FLUSHOT6 != 9)

#HIVTST6 - have you ever been tested for HIV? binary variable, 1 is yes and 2 is no
#will add 7s and 9s to median value of no
summary(newTestSet6$HIVTST6)
newTestSet6 %>% filter(HIVTST6 < 3) %>% summarize(med = median(HIVTST6))
newTestSet6 %>% group_by(HIVTST6) %>% summarise(n = n())

newTestSet6$HIVTST6 <- replace(newTestSet6$HIVTST6, newTestSet6$HIVTST6 == 7, 2)
newTestSet6$HIVTST6 <- replace(newTestSet6$HIVTST6, newTestSet6$HIVTST6 == 9, 2)
```

```
#CAREGIV1: have you provided regular care or assistance to someone with a disability in last
30 days?
#replacing 7 and 9 with 2, na with 2, 8 with 1
summary(newTestSet6$CAREGIV1)
newTestSet6 %>% group_by(CAREGIV1) %>% summarise(n = n())
newTestSet6$CAREGIV1 <- replace(newTestSet6$CAREGIV1, newTestSet6$CAREGIV1 ==
7, 2)
newTestSet6$CAREGIV1 <- replace(newTestSet6$CAREGIV1, newTestSet6$CAREGIV1 ==
8, 1)
newTestSet6$CAREGIV1 <- replace(newTestSet6$CAREGIV1, newTestSet6$CAREGIV1 ==
9, 2)
newTestSet6$CAREGIV1 <- replace(newTestSet6$CAREGIV1,
is.na(newTestSet6$CAREGIV1), 2)

#_ageg5yr: 5-year age group from 18 to 80+
ages <- newTestSet6 %>%
        group_by(`_AGEG5YR`) %>%
        summarize(n = n())
#251 without age, will remove those
newTestSet6 <- newTestSet6 %>% filter(`_AGEG5YR` != 14)

#HADMAM: binary asking if they have ever had a mammogram
#a handful of 7s and 9s; will just remove those
newTestSet6 <- newTestSet6 %>% filter(HADMAM < 3)
testSet2 %>% group_by(HADMAM) %>% summarise(n = n())

summary(newTestSet6)

#need to scale variables
#will scale variables that are of the form "X days in last 30 days" with decimal scaling
#htin and weight will be scaled with z-score
#min-max used for sumfruit, sumveg, maxvo2

#starting with z-scores
meanWt <- mean(newTestSet6$WEIGHTLBS)
sdWt <- sd(newTestSet6$WEIGHTLBS)

normTestSet1 <- newTestSet6 %>%
              mutate(WEIGHTNORM = (WEIGHTLBS - meanWt)/sdWt)
summary(normTestSet1$WEIGHTNORM)
```

```
normTestSet1 <- normTestSet1 %>% select(-WEIGHTLBS)

meanHt <- mean(normTestSet1$HTIN4)
sdHt <- sd(normTestSet1$HTIN4)

normTestSet1 <- normTestSet1 %>%
        mutate(HEIGHTNORM = (HTIN4 - meanHt)/sdHt)
summary(normTestSet1$HEIGHTNORM)
normTestSet1 <- normTestSet1 %>% select(-HTIN4)

#min-max scaling
#will scale each from 0-6 - in line with scale of other variables

targetMin <- 0
targetMax <- 6
targetRange <- targetMax - targetMin

minFruit <- min(normTestSet1$SUMFRUIT)
maxFruit <- max(normTestSet1$SUMFRUIT)
rangeFruit <- maxFruit - minFruit

minVeg <- min(normTestSet1$SUMVEG)
maxVeg <- max(normTestSet1$SUMVEG)
rangeVeg <- maxVeg - minVeg

minO2 <- min(normTestSet1$MAXVO2)
maxO2 <- max(normTestSet1$MAXVO2)
rangeO2 <- maxO2 - minO2

normTestSet2 <- normTestSet1 %>%
        mutate(FRUITNORM = (((SUMFRUIT - minFruit)/rangeFruit)*targetRange),
            VEGNORM = (((SUMVEG - minVeg)/rangeVeg)*targetRange),
            MAXVO2NORM = (((MAXVO2 - minO2)/rangeO2)*targetRange),
        ) %>%
        select(-all_of(c('SUMFRUIT','SUMVEG','MAXVO2')))
summary(normTestSet2)

#decimal scaling for variables in the X days in last 30 days group
#max here is 30 so will divide by 10 - max will be 3, min stays 0
```

```
normTestSet3 <- normTestSet2 %>%
        mutate(NORMPHYS = PHYSHLTH/10,
           NORMMENT = MENTHLTH/10,
           NORMPOOR = POORHLTH/10,
           NORMALC = ALCDAYSCLEAN/10,
           NORMSTREN = STRENCLEAN/10,
           NORMEXER = EXERTIMES/10) %>%
        select(-all_of(c('PHYSHLTH','MENTHLTH','POORHLTH','ALCDAYSCLEAN',
                'STRENCLEAN','EXERTIMES')))

summary(normTestSet3)
#all variables now in same scale, ~-2 to ~13


#final look and converting to .csv
finalTestSet <- normTestSet3

#happy with this dataset - will convert to csv and start new file for analysis
setwd('/Users/alexandradegrandchamp/Documents/GradSchool/DSC424/Final
Project/SelfAnalysis')
cwd <- getwd()
fullFN <- 'clusteringSet.csv'
write.csv(finalTestSet, paste(cwd,fullFN,sep='/'), row.names = FALSE)
```

```
#analysis file - clustering and women's health
#written by alexandra degrandchamp

#Clustering for Final Project
#research question: what are the clusters of behavior women exhibit when it comes to
preventative health?

#libraries
library(readr)
library(tidyverse)
library(stats)
library(caret)
library(factoextra)
library(reshape2)
library(ggthemes)

#data set - precleaned
clusteringSet <- read_csv("Documents/GradSchool/DSC424/Final
Project/SelfAnalysis/clusteringSet.csv")
summary(clusteringSet)

#variable types
#binary - hadmam, hadhyst2, hlthpln1, medcost, internet, smoke100, flushot6, hivtst6, caregiv1,
#hadhpvorpap, activlvl, aerorecs
#interval - howlong, lastpap2, checkup1
#ordinal - genhlth, educa, employ1, seatbelt, _ageg5yr, incomg
#continuous - physhlth, mentlhlth, poorhlth, children, htin4, weightlbs, alcdaysclean,
#sumfruit, sumveg, strenclean, exertimes, maxvo2, funccap

#mostly binary and continuous variables - all pre-cleaned to scale similarly

#want to try a couple of distances and linkages before settling on a final answer
#distances: binary, euclidean, manhattan
#linkages: complete, median, Ward-D

#binary/complete
dist_bin1 <- dist(clusteringSet, method = 'binary')
hfit1 <- hclust(dist_bin1, method = 'complete')
plot(hfit1)
```

```
#binary/median
hfit2 <- hclust(dist_bin1, method = 'median')
plot(hfit2)

#binary/ward-D
hfit3 <- hclust(dist_bin1, method = 'ward.D')
plot(hfit3)

#euclidean/complete
dist_euc1 <- dist(clusteringSet, method = 'euclidean')
hfit4 <- hclust(dist_euc1, method = 'complete')
plot(hfit4)

#euclidean/median
hfit5 <- hclust(dist_euc1, method = 'median')
plot(hfit5)

#euclidean/ward-D
hfit6 <- hclust(dist_euc1, method = 'ward.D')
plot(hfit6)

#manhattan/complete
dist_man1 <- dist(clusteringSet, method = 'manhattan')
hfit7 <- hclust(dist_man1, method = 'complete')
plot(hfit7)

#manhattan/median
hfit8 <- hclust(dist_man1, method = 'median')
plot(hfit8)

#manhattan/ward-D
hfit9 <- hclust(dist_man1, method = 'ward.D')
plot(hfit9)

#initial results
#median linkage resulted in skewed dendrograms, will not use
#euclidean complete also had a high degree of skew, will not use
#manhattan complete is skewed, will not use
#binary complete is a lot of noise
```

```r
#ward-d linkage has the best noise and skew control
#Euclidean is less skewed than binary/manhattan
#hfit6

#pca for visualization
pca <- prcomp(clusteringSet)
rotated_pca <- as.data.frame(pca$x)

#modeling various cluster sizes - want to look at between 4 and 8
h4_eucD <- cutree(hfit6, k = 4)

rotated_pca$ClusterEucWardD <- as.factor(h4_eucD)

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = ClusterEucWardD)) +
  geom_point(alpha = 0.5)

h5_eucD <- cutree(hfit6, k = 5)
h6_eucD <- cutree(hfit6, k = 6)
h7_eucD <- cutree(hfit6, k = 7)
#actually will not bother with 8 because 7 is already too blurred

rotated_pca$MD4 <- as.factor(h4_eucD)
rotated_pca$MD5 <- as.factor(h5_eucD)
rotated_pca$MD6 <- as.factor(h6_eucD)
rotated_pca$MD7 <- as.factor(h7_eucD)

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = MD4)) +
  geom_point(alpha = 0.5)

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = MD5)) +
  geom_point(alpha = 0.5)

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = MD6)) +
  geom_point(alpha = 0.5)

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = MD7)) +
  geom_point(alpha = 0.5)

#4 results in the most distinct clusters
```

```
#adding clusters to original data set to understand cluster characteristics

numberedSet <- clusteringSet %>% mutate(rowNum = row_number())
dataWithClusters <- lapply(1:4, function(nc) numberedSet$rowNum[h4_eucD==nc])

#most observations fall in cluster 2 (8542), with 1 having 6779, 3 3436, and 4 2875

numberedSet <- numberedSet %>% rename('AGEGROUP' = `_AGEG5YR`)

#plotting density plots of all variables by cluster to understand unique characteristics

cluster1 <- numberedSet %>% filter(rowNum %in% dataWithClusters[[1]]) %>%
select(-rowNum)
cluster2 <- numberedSet %>% filter(rowNum %in% dataWithClusters[[2]]) %>%
select(-rowNum)
cluster3 <- numberedSet %>% filter(rowNum %in% dataWithClusters[[3]]) %>%
select(-rowNum)
cluster4 <- numberedSet %>% filter(rowNum %in% dataWithClusters[[4]]) %>%
select(-rowNum)

meltClust1 <- melt(cluster1)
meltClust2 <- melt(cluster2)
meltClust3 <- melt(cluster3)
meltClust4 <- melt(cluster4)

ggplot(data = meltClust1, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = 'free') +
  theme_clean() +
  ggtitle('Cluster 1 Densities')

ggplot(data = meltClust2, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = 'free') +
  theme_clean() +
  ggtitle('Cluster 2 Densities')

ggplot(data = meltClust3, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = 'free') +
  theme_clean() +
  ggtitle('Cluster 3 Densities')

ggplot(data = meltClust4, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = 'free') +
  theme_clean() +
  ggtitle('Cluster 4 Densities')

#four characteristics can be loosely defined on weight, health, and activity level:
#Cluster 1 - Younger Retirees
#Cluster 2 - Mid-Life Workers
#Cluster 3 - Older Retirees
#Cluster 4 - Young Professionals

#cleaning up a presentation version of 4 clusters scatter plot
rotated_pca <- rotated_pca %>% rename('MD4' = 'Clusters')
rotated_pca <- rotated_pca %>%
  mutate(Clusters = case_when(MD4 == 1 ~ 'Younger Retirees',
                              MD4 == 2 ~ 'Mid-Life Workers',
                              MD4 == 3 ~ 'Older Retirees',
                              MD4 == 4 ~ 'Young Professionals'
  ))

ggplot(rotated_pca, aes(x = PC1, y = PC2, col = Clusters)) +
  geom_point(alpha = 0.5) +
  theme_clean() +
  theme(legend.position = 'bottom') +
  ggtitle('Clusters by PC')
```

<u>Rasa Willette's Code</u>

<u>File 1:</u>

```
#libraries used
library(tidyverse)
library(naniar)

# read in the file

# # raw file from kaggle
# X2015 = read.table("2015.csv", sep=",", header=T)
# summary(X2015)

# cleaned R script
cleanedDataset = read.table("cleanedFullSet.csv", sep=",", header=T)
summary(cleanedDataset)


# #joining in states (copied from pdf as separate .txt file)
# states_clean <- states %>%
#               select(Value,Value_1)
# states_clean <- na.omit(states_clean)
# states_clean$Value <- as.double(states_clean$Value)
#
# dataset2015 <- X2015 %>%
#             left_join(states_clean, by = c('_STATE' = 'Value'))
#
# #cleaning date values
# dataset2015$IDATE <- str_sub(dataset2015$IDATE,start = 3,-1)
# dataset2015$IDATE <- mdy(dataset2015$IDATE)
# dataset2015$IMONTH <- month(dataset2015$IDATE)
# dataset2015$IDAY <- day(dataset2015$IDATE)
# dataset2015$IYEAR <- year(dataset2015$IDATE)
#
# #gathering only complete rows
# completedInterviews <- dataset2015 %>%
#                   filter(DISPCODE == 1100)
# #taking a look at some intro question dispersions
# summary(completedInterviews$CTELENUM)
# summary(completedInterviews$PVTRESD1)
# summary(completedInterviews$COLGHOUS)
#
# #cleaning up and factorizing state values
# completedInterviews <- completedInterviews %>%
#                 rename('StateText' = Value_1)
# completedInterviews$StateText <- as.factor(completedInterviews$StateText)
#
# #checking for empty columns
# summary(completedInterviews)
#
# #not all qs asked in 2015
# interviewsNoBlankColumns <- completedInterviews %>%
#                     select(-c('COLGHOUS','LADULT','PAINACT2',
#                           'QLMENTL2','QLSTRES2','QLHLTH2',
#                           'ASERVIST','ASDRVIST'))


# #looking for incomplete cases based on survey flow
# testingState <- interviewsNoBlankColumns %>%
#             filter(STATERES != 1)
# rm(testingState) #0 results
# testingCell <- interviewsNoBlankColumns %>%
#             filter(CELLFON3 == 2)
# rm(testingCell) #doesn't actually terminate survey
#
# #testing cell a second time
# testingCell2 <- interviewsNoBlankColumns %>%
#             filter(CELLFON3 == 2 & CELLFON2 == 2)
# rm(testingCell2) #0 results
#
# #rounding some funky looking should-be-integers-but-aren't
# interviewsNoBlankColumns$NUMADULT <-
as.integer(interviewsNoBlankColumns$NUMADULT)
# interviewsNoBlankColumns$NUMMEN <-
as.integer(interviewsNoBlankColumns$NUMMEN)
# interviewsNoBlankColumns$NUMWOMEN <-
as.integer(interviewsNoBlankColumns$NUMWOMEN)
```

```
#
# #"duplicate" column cleanup
# #this dataset aggregates the land-line and cell-phone surveys, with different variables for each
# #this merges columns together or removes them if they are irrelevant
#
# #CTELENUM - land line ask if number is correct
# #CTELNUM1 - cell phone ask if number is correct
# summary(interviewsNoBlankColumns$CTELENUM)
# summary(interviewsNoBlankColumns$CTELNUM1)
# #these are irrelevant columns
#
# deDupe <- interviewsNoBlankColumns %>%
#           select(-c('CTELENUM','CTELNUM1'))

# #PVTRESD1 - land line ask if number is private residence
# #PVTRESD2 - cell phone ask if number is private residence
# summary(deDupe$PVTRESD1)
# summary(deDupe$PVTRESD2)
#
# deDupe2 <- deDupe %>%
#           unite('PVTRES',c('PVTRESD1','PVTRESD2'), na.rm = TRUE)
# deDupe2$PVTRES <- as.integer(deDupe2$PVTRES)
# summary(deDupe2$PVTRES)
#
# #COLGHOUS - land line ask if number is college housing, removed from set
# #CCLGHOUS - cell phone ask if number is college housing, kept in set
# #no changes here just noting the duplication has been handled
#
# #NUMADULT - land line ask for number of adults in household
# #HHADULT - cell phone ask for number of adults in household
# summary(deDupe2$NUMADULT)
# summary(deDupe2$HHADULT)

# deDupe3 <- deDupe2 %>%
#           unite('NUMADULTS',c('NUMADULT','HHADULT'), na.rm = TRUE)
# #getting NAs when converting to numeric, so seeing what is up there
# deDupe3$NUMADULTS <- as.factor(deDupe3$NUMADULTS)
# numAdultsCheck <- deDupe3 %>%
#                group_by(NUMADULTS) %>%
#                summarize(n = n())
# #further cleaning - 7 is don't know, 99 is refused to answer - will turn those to N/As for now
# #will also turn '' into N/A
# deDupe3 <- deDupe3 %>%
#           replace_with_na(replace = list(NUMADULTS = c('77','99','')))
# #there's also one row with a 4_3, which means on landline someone said 4
# #and the same someone said 3 on a cellphone
# #that shouldn't happen either, but deleting it is giving me trouble, so changing it to NA by
coercion
# deDupe3$NUMADULTS <- as.character(deDupe3$NUMADULTS)
# deDupe3$NUMADULTS <- as.integer(deDupe3$NUMADULTS)
# summary(deDupe3$NUMADULTS)
#
# #id or duplicated columns - will create subset without
# irrelevantColumns = c('IMONTH','IDAY','IYEAR','DISPCODE','FMONTH','_STATE',
'STATERES','CELLFON3',
#                   'CELLFON2')
# #note: keeping SEQNO and PSU to join back together if data is separated
#
# relevantData <- deDupe3 %>%
#               select(-all_of(irrelevantColumns))

# #starting on actual questions cleaning
# #GENHLTH - categorical where 9 is refused
# cleaningVars <- relevantData %>%
#               replace_with_na(replace = list(GENHLTH = 9))
# cleaningVars$GENHLTH <- as.factor(cleaningVars$GENHLTH)
#
# #PHYSHLTH - replacing 88 with 0, 77 and 99 with N/A
# cleaningVars$PHYSHLTH[cleaningVars$PHYSHLTH == 88] <- 0
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(PHYSHLTH = c(77,99)))
#
# #MENTHLTH - replacing 88 with 0, 77 and 99 with N/A
# cleaningVars$MENTHLTH[cleaningVars$MENTHLTH == 88] <- 0
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(MENTHLTH = c(77,99)))
#
# #POORHLTH - replacing 88 with 0, 77 and 99 with N/A
# #note: skipped q if 0 days in either PHYSHLTH or MENTHLTH
# cleaningVars$POORHLTH[cleaningVars$POORHLTH == 88] <- 0
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(POORHLTH = c(77,99)))
#
# #HLTHPLN1 - changing 7 and 9 with N/A. This is a categorical variable
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(HLTHPLN1 = c(7,9)))
# cleaningVars$HLTHPLN1 <- as.factor(cleaningVars$HLTHPLN1)

# #PERSDOC2 - changing 9 with N/A. This is a categorical variable
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(PERSDOC2 = c(9)))
# cleaningVars$PERSDOC2 <- as.factor(cleaningVars$PERSDOC2)
#
# #MEDCOST - changing 9 with N/A. This is a categorical variable
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(MEDCOST = c(9)))
# cleaningVars$MEDCOST <- as.factor(cleaningVars$MEDCOST)
#
# #CHECKUP1 - changing 9 to N/A. This is a categorical variable
# cleaningVars <- cleaningVars %>%
#               replace_with_na(replace = list(CHECKUP1 = c(9)))
# cleaningVars$MEDCOST <- as.factor(cleaningVars$MEDCOST)

#this is about the point where I realized this was going to take 100 years this way
#switching it up
#read through the codebook and put each variable in either quant or cat
#I also classified weighting/coding variables and identified calculated fields separately (for
multicollinearity concerns)
#IDATE, SEQNO, X_PSU kept in each set as identifiers




#quantVars <- cleaningVars %>%
quantVars <- cleanedDataset %>%
          select(IDATE, SEQNO, `X_PSU`, NUMADULTS, NUMMEN, NUMWOMEN,
PHYSHLTH,
              MENTHLTH, POORHLTH, DIABAGE2, NUMPHON2, CHILDREN, WEIGHT2,
              HEIGHT3, ALCDAY5, AVEDRNK2, DRNK3GE5, MAXDRNKS, FRUITJU1,
                FRUIT1, FVBEANS, FVGREEN, FVORANG, VEGETAB1, EXEROFT1,
EXERHMM1, EXEROFT2,
                EXERHMM2, STRENGTH, BLDSUGAR, FEETCHK2, DOCTDIAB,
CHKHEMO3, FEETCHK, LONGWTCH,
                ASTHMAGE, ASRCHKUP, ASACTLIM, SCNTWRK1, SCNTLWK1,
ADPLEASR, ADDOWN, ADSLEEP,
                ADENERGY, ADEAT1, ADFAIL, ADTHINK, ADMOVE)

#catVars <- cleaningVars %>%
catVars <- cleanedDataset %>%
          select(IDATE, SEQNO, `X_PSU`, StateText, PVTRES, CCLGHOUS, CSTATE,
LANDLINE,
                GENHLTH, HLTHPLN1, PERSDOC2, MEDCOST, CHECKUP1, BPHIGH4,
BPMEDS,
                BLOODCHO, CHOLCHK, TOLDHI2, CVDINFR4, CVDCRHD4, CVDSTRK3,
ASTHMA3,
                ASTHNOW, CHCSCNCR, CHCOCNCR, CHCCOPD1, HAVARTH3,
ADDEPEV2,
                CHCKIDNY, DIABETE3, SEX, MARITAL, EDUCA, RENTHOM1,
NUMHHOL2,
                CPDEMO1, VETERAN3, EMPLOY1, INCOME2, INTERNET, PREGNANT,
QLACTLM2,
                USEEQUIP, BLIND, DECIDE, DIFFWALK, DIFFDRES, DIFFALON,
SMOKE100,
                SMOKDAY2, STOPSMK2, LASTSMK2, USENOW3, EXERANY2, EXRACT11,
EXRACT21,
                LMTJOIN3, ARTHDIS2, ARTHSOCL, JOINPAIN, SEATBELT, FLUSHOT6,
FLSHTMY2, IMFVPLAC,
                PNEUVAC3, HIVTST6, HIVTSTD3, WHRTST10, PDIABTST, PREDIAB1,
INSULIN,
                EYEEXAM, DIABEYE, DIABEDU, CAREGIV1, CRGVREL1, CRGVLNG1,
CRGVHRS1,
                CRGVPRB1, CRGVPERS, CRGVHOUS, CRGVMST2, CRGVEXPT,
VIDFCLT2, VIREDIF3,
                VIPRFVS2, VINOCRE2, VIEYEXM2, VIINSUR2, VICTRCT4, VIGLUMA2,
VIMACDG2,
                CIMEMLOS, CDHOUSE, CDASSIST, CDHELP, CDSOCIAL, CDDISCUS,
WTCHSALT,
                DRADVISE, ASATTACK, ASYMPTOM, ASNOSLEP, ASTHMED3,
ASINHALR, HAREHAB1,
                  STREHAB1, CVDASPRN, ASPUNSAF, RLIVPAIN, RDUCHART, RDUCSTRK,
ARTTODAY,
                  ARTHWGT, ARTHEXER, ARTHEDU, TETANUS, HPVADVC2, HPVADSHT,
SHINGLE2,HADMAM,
                  HOWLONG, HADPAP2, LASTPAP2, HPVTEST, HPLSTTST, HADHYST2,
PROFEXAM,LENGEXAM,
                  BLDSTOOL, LSTBLDS3, HADSIGM3, HADSGCO1, LASTSIG3, PCPSAAD2,
PCPSADI1,
                  PCPSARE1, PSATEST1, PSATIME, PCPSARS1, PCPSADE1, PCDMDECN,
SCNTMNY1,
                  SCNTMEL1, SCNTPAID, SCNTLPAD, SXORIENT, TRNSGNDR, RCSGENDR,
RCSRLTN2, CASTHDX2,
                  CASTHNO2, EMTSUPRT, LSATISFY, MISTMNT, ADANXEV, QSTVER,
QSTLANG, EXACTOT1,
                  EXACTOT2, `X_CHISPNC`, `X_CRACE1`, `X_CPRACE`, `X_DUALUSE`)

#codingVars <- cleaningVars %>%
codingVars <- cleanedDataset %>%
              select(IDATE, SEQNO, `X_PSU`, MSCODE, `X_STSTR`, `X_STRWT`,
`X_RAWRAKE`,
                   `X_WT2RAKE`, `X_CLLCPWT`, `X_DUALCOR`, `X_LLCPWT`)

#calcVars <- cleaningVars %>%
calcVars <- cleanedDataset %>%
             select(IDATE, SEQNO, `X_PSU`, `X_RFHLTH`, `X_HCVU651`, `X_RFHYPE5`,
`X_CHOLCHK`,
                  `X_RFCHOL`, `X_MICHD`, `X_LTASTH1`, `X_CASTHM1`, `X_ASTHMS1`,
`X_DRDXAR1`,
                  `X_PRACE1`, `X_MRACE1`, `X_HISPANC`, `X_RACE`, `X_RACEG21`,
`X_RACEGR3`,
                  `X_RACE_G1`, `X_AGEG5YR`, `X_AGE65YR`, `X_AGE80`, `X_AGE_G`,
HTIN4, HTM4,
                  WTKG3, `X_BMI5`, `X_BMI5CAT`, `X_RFBMI5`, `X_CHLDCNT`,
`X_EDUCAG`, `X_INCOMG`,
                  `X_SMOKER3`, `X_RFSMOK3`, DRNKANY5, DROCDY3_, `X_RFBING5`,
`X_DRNKWEK`,
                  `X_RFDRHV5`, FTJUDA1_, FRUTDA1_, BEANDAY_, GRENDAY_,
ORNGDAY_, VEGEDA1_,
                  `X_MISFRTN`, `X_MISVEGN`, `X_FRTRESP`, `X_VEGRESP`,
`X_FRUTSUM`, `X_VEGESUM`,
                `X_FRTLT1`, `X_VEGLT1`, `X_FRT16`, `X_VEG23`, `X_FRUITEX`,
`X_VEGETEX`,
                `X_TOTINDA`, METVL11_, METVL21_, MAXVO2_, FC60_, ACTIN11_,
ACTIN21_,
                PADUR1_, PADUR2_, PAFREQ1_, PAFREQ2_, `X_MINAC11`, `X_MINAC21`,
STRFREQ_,
                PAMISS1_, PAMIN11_, PAMIN21_, PA1MIN_, PAVIG11_, PAVIG21_,
PA1VIGM_,
                `X_PACAT1`, `X_PAINDX1`, `X_PA150R2`, `X_PA300R2`, `X_PA30021`,
`X_PASTRNG`,
                `X_PAREC1`, `X_PASTAE1`, `X_LMTACT1`, `X_LMTWRK1`, `X_LMTSCL1`,
`X_RFSEAT2`,
                `X_RFSEAT3`, `X_FLSHOT6`, `X_PNEUMO2`, `X_AIDTST3`)


# #exporting clean data set and subsets to csv files
setwd('/Users/rasawillette/rStudio/AdvancedDataAnalysis/project/project')
cwd <- getwd()

fullFN <- 'cleanedFullSet.csv'
#write.csv(cleaningVars, paste(cwd,fullFN,sep='/'), row.names = FALSE)
write.csv(cleanedDataset, paste(cwd,fullFN,sep='/'), row.names = FALSE)



quantFN <- 'quantData.csv'
catFN <- 'categoricalData.csv'
codingFN <- 'codingVariables.csv'
calcFN <- 'calculatedFields.csv'

# write.csv(cleaningVars, paste(cwd,quantFN,sep='/'), row.names = FALSE)
# write.csv(cleaningVars, paste(cwd,catFN,sep='/'), row.names = FALSE)
# write.csv(cleaningVars, paste(cwd,codingFN,sep='/'), row.names = FALSE)
# write.csv(cleaningVars, paste(cwd,calcFN,sep='/'), row.names = FALSE)

write.csv(quantVars, paste(cwd,quantFN,sep='/'), row.names = FALSE)
write.csv(catVars, paste(cwd,catFN,sep='/'), row.names = FALSE)
write.csv(codingVars, paste(cwd,codingFN,sep='/'), row.names = FALSE)
write.csv(calcVars, paste(cwd,calcFN,sep='/'), row.names = FALSE)
```

File 2:

```
# Rasa Willette
# DSC 424
# HW 3 Q 1 and Group Project


# question 1



###############################################
#                Imports                #
###############################################


library(corrr)
library(ggcorrplot)
library(FactoMineR)
library(psych)


# load in the data
# going to use the numerical only data subset for PCA analysis

quantData = read.table("quantData.csv", sep=",", header=T)
summary(quantData)

# cleanedFullSet = read.table("cleanedFullSet.csv", sep=",", header=T)
# summary(cleanedFullSet)

# categoricalData = read.table("categoricalData.csv", sep=",", header=T)
# summary(categoricalData)
#
# calculatedFields = read.table("calculatedFields.csv", sep=",", header=T)
# summary(calculatedFields)
#
# codingVariables = read.table("codingVariables.csv", sep=",", header=T)
# summary(codingVariables)
```

```
#################################################
#               Clean Dataset               #
#################################################


# save numerical data to new dataframe (without date)
quantData_noDate = quantData[,-1]
cor(quantData_noDate)

# PCA
#p = prcomp(quantData_noDate)
#has NA values so it throws an error

D = dim(quantData_noDate)
store_bad_col=c()

#remove any variable with more than 11k NaN's
for (i in 1:D[2]){
  current_variable = quantData_noDate[,i]
  idx = which(is.na(current_variable))
  L=length(idx)
  if (L> 11000){
    store_bad_col = c(store_bad_col,i)
  }
}

store_bad_col
quantData_noDate_Clean = quantData_noDate[,-store_bad_col]
dim(quantData_noDate_Clean)
quantData_noDate_Clean

quantData_noDate_noSequence_Clean = quantData_noDate_Clean[, -c(1:2)]
dim(quantData_noDate_noSequence_Clean)
quantData_noDate_noSequence_Clean

#replace NaNs w/ median
D = dim(quantData_noDate_noSequence_Clean)
quantData_noDate_noSequence_Cleaner = quantData_noDate_noSequence_Clean
```

```
for (i in 1:D[2]){
  current_variable = quantData_noDate_noSequence_Cleaner[,i]
  MEDIAN = median(current_variable,na.rm=TRUE)
  idx = which(is.na(current_variable))
  current_variable[idx] = MEDIAN
  quantData_noDate_noSequence_Cleaner[,i] = current_variable
}

#check to make sure Na's are removed:
idx=which(is.na(quantData_noDate_noSequence_Cleaner))
idx



#################################################
#              Final Dataset               #
#################################################


final_dataset = quantData_noDate_noSequence_Cleaner



#################################################
#                  PCA                    #
#################################################


# try PCA
p = prcomp(final_dataset)
summary(p)
round(p$rotation, 2)

plot(p)
# p$x

# PCA again
p2 = prcomp(final_dataset, scale=T)
summary(p2)
```

```
round(p2$rotation, 2)

plot(p2)
abline(1, 0, col="red")

s = as.data.frame(p2$x)
attach(s)
s[order(PC1), 1:5]
detach(s)

# PCA again
p3 = principal(final_dataset, nf=5)
p3
p3$loadings



#################################################
#               PCA Plots                    #
# Taken from class file: PCA_Plot.R (Week 2) #
#################################################


library(ggplot2)
library(psych)


PCA_Plot = function(pcaData)
{
  library(ggplot2)

  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()

  loadings = data.frame(pcaData$rotation, .names = row.names(pcaData$rotation))
  p + geom_text(data=loadings, mapping=aes(x = PC1, y = PC2, label = .names, colour = .names,
fontface="bold")) +
    coord_fixed(ratio=1) + labs(x = "PC1", y = "PC2")
}
```

```
PCA_Plot_rasa = function(pcaData)
{
  library(ggplot2)

  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()

  loadings = data.frame(pcaData$rotation, .names = row.names(pcaData$rotation))
  # p + geom_text(data=loadings, mapping=aes(x = PC1, y = PC2, label = .names, colour =
.names, fontface="bold")) +
  #   coord_fixed(ratio=1) + labs(x = "PC1", y = "PC2")

  p + geom_point(data=loadings, mapping=aes(x = PC1, y = PC2, colour = .names)) +
    coord_fixed(ratio=1) + labs(x = "PC1", y = "PC2")
}

PCA_Plot_Secondary = function(pcaData)
{
  library(ggplot2)

  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()

  loadings = data.frame(pcaData$rotation, .names = row.names(pcaData$rotation))
  p + geom_text(data=loadings, mapping=aes(x = PC3, y = PC4, label = .names, colour = .names,
fontface="bold")) +
    coord_fixed(ratio=1) + labs(x = "PC3", y = "PC4")
}


PCA_Plot_Secondary_rasa = function(pcaData)
{
  library(ggplot2)

  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()
  loadings = data.frame(pcaData$rotation, .names = row.names(pcaData$rotation))
  # p + geom_text(data=loadings, mapping=aes(x = PC3, y = PC4, label = .names, colour =
.names, fontface="bold")) +
  #   coord_fixed(ratio=1) + labs(x = "PC3", y = "PC4")
  p + geom_point(data=loadings, mapping=aes(x = PC3, y = PC4, colour = .names)) +
    coord_fixed(ratio=1) + labs(x = "PC3", y = "PC4")
}

PCA_Plot_Psyc = function(pcaData)
{
  library(ggplot2)

  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()

  loadings = as.data.frame(unclass(pcaData$loadings))
  s = rep(0, ncol(loadings))
  for (i in 1:ncol(loadings))
  {
    s[i] = 0
    for (j in 1:nrow(loadings))
      s[i] = s[i] + loadings[j, i]^2
    s[i] = sqrt(s[i])
  }

  for (i in 1:ncol(loadings))
    loadings[, i] = loadings[, i] / s[i]

  loadings$.names = row.names(loadings)

  p + geom_text(data=loadings, mapping=aes(x = RC1, y = RC2, label = .names, colour =
.names, fontface="bold")) +
    coord_fixed(ratio=1) + labs(x = "PC1", y = "PC2")
}

PCA_Plot_Psyc_Secondary = function(pcaData)
{
  library(ggplot2)
  theta = seq(0,2*pi,length.out = 100)
  circle = data.frame(x = cos(theta), y = sin(theta))
  p = ggplot(circle,aes(x,y)) + geom_path()

  loadings = as.data.frame(unclass(pcaData$loadings))
  s = rep(0, ncol(loadings))
  for (i in 1:ncol(loadings))
  {
    s[i] = 0
    for (j in 1:nrow(loadings))
      s[i] = s[i] + loadings[j, i]^2
    s[i] = sqrt(s[i])
  }

  for (i in 1:ncol(loadings))
    loadings[, i] = loadings[, i] / s[i]

  loadings$.names = row.names(loadings)

  print(loadings)
  p + geom_text(data=loadings, mapping=aes(x = PC3, y = PC4, label = .names, colour = .names,
fontface="bold")) +
    coord_fixed(ratio=1) + labs(x = "PC3", y = "PC4")
}



################################################
#             My PCA Plots                  #
################################################


PCA_Plot(p)
PCA_Plot(p2)
PCA_Plot_rasa(p2)


PCA_Plot_Secondary(p)
PCA_Plot_Secondary(p2)
```

```
PCA_Plot_Secondary_rasa(p2)


# PCA_Plot_Psyc(p)
# PCA_Plot_Psyc(p2)
# PCA_Plot_Psyc(p3)

# PCA_Plot_Psyc_Secondary(p)
# PCA_Plot_Psyc_Secondary(p2)
# PCA_Plot_Psyc_Secondary(p3)
```

Jessica Bicek's Code:

```
install.packages("tidyverse")
library(tidyverse)
data <- read_csv("Downloads/2015.csv")
names(data)
#Looking at the relationship between feeling supported and chronic illness

#building emotion set
emotion <- data[,211:212]
emotion2 <- emotion
emotion2[is.na(emotion2)] = 0

#building and cleaning chronic data set
chronic <- data[,c(40:43,45:49,51)]
sum(is.na(chronic))
chronic2 <- chronic
chronic2[is.na(chronic2)] = 0


ccaWilks = function(set1, set2, cca)
{
  ev = ((1 - cca$cor^2))
  ev

  n = dim(set1)[1]
  p = length(set1)
  q = length(set2)
  k = min(p, q)
  m = n - 3/2 - (p + q)/2
  m

  w = rev(cumprod(rev(ev)))

  # initialize
  d1 = d2 = f = vector("numeric", k)

  for (i in 1:k)
  {
    s = sqrt((p^2 * q^2 - 4)/(p^2 + q^2 - 5))
    si = 1/s
    d1[i] = p * q
    d2[i] = m * s - p * q/2 + 1
    r = (1 - w[i]^si)/w[i]^si
    f[i] = r * d2[i]/d1[i]
    p = p - 1
    q = q - 1
  }

  pv = pf(f, d1, d2, lower.tail = FALSE)
  dmat = cbind(WilksL = w, F = f, df1 = d1, df2 = d2, p = pv)
}


library(yacca)
c3 = cca(chronic2, emotion2)
summary(c3)

helio.plot(c3, cv=1, x.name="Chronic Illness",
         y.name="QOL")
#CV2
helio.plot(c3, cv=2, x.name="Chronic Illness",
         y.name="QOL")

#Function Names
ls(c3)

# Perform a chi-square test on C2
c3
ls(c3)
c3$chisq
c3$df
summary(c3)
round(pchisq(c3$chisq, c3$df, lower.tail=F), 3)
```

Mahender/ Maahi's Code:

~~~
```{r}
install.packages(c("lavaan", "dplyr", "psych"))
library(lavaan)
library(dplyr)
library(psych)
```

```{r}
# Loading the dataset
health_dataset <- read.csv("2015.csv", header = TRUE)

# Filtering the dataset based on the required columns
health_df_selected <- health_dataset %>%
  select(DIABETE3, X_RFHYPE5, TOLDHI2, X_CHOLCHK, X_BMI5,
SMOKE100,
         CVDSTRK3, X_MICHD, X_TOTINDA, X_FRTLT1, X_VEGLT1,
X_RFDRHV5,
         HLTHPLN1, MEDCOST, GENHLTH, MENTHLTH, PHYSHLTH,
DIFFWALK,
         EDUCA, INCOME2)

# Renaming columns
colnames(health_df_selected) <- c('Diabetes', 'HighBP',
'HighChol', 'CholCheck',
                                  'BMI', 'Smoker', 'Stroke',
'HeartDiseaseorAttack',
                                  'PhysActivity', 'Fruits',
'Veggies', 'HvyAlcoholConsump',
                                  'AnyHealthcare', 'NoDocbcCost',
'GenHlth', 'MentHlth',
                                  'PhysHlth', 'DiffWalk',
'Education', 'Income')
~~~

```
str(health_df_selected)
summary(health_df_selected)
print("Number of missing values per column:")
print(colSums(is.na(health_df_selected)))

for (col in colnames(health_df_selected)) {
  if (is.numeric(health_df_selected[[col]])) {
    health_df_selected[[col]][is.na(health_df_selected[[col]])]
<- mean(health_df_selected[[col]], na.rm = TRUE)
  }
}

head(health_df_selected)

scree(health_df_selected)
```
```{r}
anyNA(health_df_selected)
health_df_selected <- na.omit(health_df_selected)
any(apply(health_df_selected, 2, function(x)
any(is.infinite(x))))
zero_sd_vars <-
names(health_df_selected)[apply(health_df_selected, 2, sd) == 0]
print(zero_sd_vars)
health_df_selected <- health_df_selected[,
!(names(health_df_selected) %in% zero_sd_vars)]

fa_result <- factanal(health_df_selected, factors=3,
rotation="varimax")
print(fa_result)

cfa_model <- '
  # Latent Variables
  Lifestyle =~ PhysActivity + Fruits + Veggies +
HvyAlcoholConsump
```

```
  MedHistory =~ HighBP + HeartDiseaseorAttack + GenHlth +
PhysHlth + DiffWalk
  SocioEconStatus =~ NoDocbcCost + MentHlth + Income
'

fit <- sem(cfa_model, data = health_df_selected)
varTable(fit)

summary(fit, fit.measures=TRUE, standardized=TRUE, rsquare=TRUE)
```
```{r}
#goodness of fit
observed <- lavCor(fit, type="observed")
fitted <- lavCor(fit, type="fitted")


df <- data.frame(Observed = as.vector(observed), Fitted =
as.vector(fitted))
library(ggplot2)

ggplot(df, aes(x = Observed, y = Fitted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red") +
  ggtitle("Goodness-of-Fit Plot") +
  xlab("Observed Covariance") + ylab("Fitted Covariance")
```

Harini’s Code:

```
import pandas as pd
import numpy as np
import seaborn as sns
data = pd.read_csv("2015.csv")

missing_values_count = data.isnull().sum()

# Sorting the columns by the number of missing values in
descending order
sorted_missing_values =
missing_values_count.sort_values(ascending=False)

print(sorted_missing_values)
# Calculate the number of missing values for each column
missing_values_count = data.isnull().sum()

# Select columns with less than 10,000 missing values
selected_columns = missing_values_count[missing_values_count <
10000].index.tolist()

# Extract the selected columns from the dataset
data = data[selected_columns]
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 6))
missing_values_count.plot(kind='bar')
plt.title('Missing Values per Column')
plt.xlabel('Columns')
plt.ylabel('Number of Missing Values')
plt.xticks(rotation=45)
plt.show()
plt.figure(figsize=(10, 5))
sns.histplot(data['GENHLTH'], kde=True)
plt.title('Distribution of GENHLTH')
plt.show()

# Calculate the correlation matrix for the entire dataset
# Filter out only numeric columns
numeric_cols = data.select_dtypes(include=[np.number])

# Calculate the correlation matrix for the numeric columns
correlation_matrix = numeric_cols.corr()

# Extract pairs with strong correlations (both positive and
negative)
# We'll consider strong correlations as those with absolute
values greater than 0.5
strong_correlations =
correlation_matrix.unstack().sort_values(ascending=False,
key=abs)
strong_correlations = strong_correlations[strong_correlations !=
1] # Exclude self-correlations (which are always 1)
strong_correlations = strong_correlations.drop_duplicates() #
Drop duplicate pairs

# Get the top correlated pairs
top_correlated_pairs = strong_correlations.head(10)

top_correlated_pairs
# Calculate the correlations specifically for GENHLTH
```

```
genhlth_correlations =
correlation_matrix['GENHLTH'].sort_values(ascending=False,
key=abs)

# Exclude the self-correlation of GENHLTH with itself
genhlth_correlations = genhlth_correlations[genhlth_correlations
!= 1]

# Get the top correlated variables with GENHLTH
top_genhlth_correlated = genhlth_correlations.head(10)

top_genhlth_correlated

#select a set of variables for further analysis
selected_variables = ['_RFHLTH', 'PHYSHLTH', '_DRDXAR1',
'EMPLOY1', 'EDUCA', 'BPHIGH4', '_EDUCAG', '_MICHD', 'QLACTLM2',
'DIABETE3']
# Extract the selected columns from the dataset
selected_data_analysis = data[selected_variables]

# Check the correlation matrix for the selected variables
selected_correlation_matrix = selected_data_analysis.corr()

selected_correlation_matrix

plt.figure(figsize=(12, 6))
sns.heatmap(selected_correlation_matrix, annot=True,
cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap of Selected Variables')
plt.show()
```

```
selected_columns = ['GENHLTH','_RFHLTH', 'PHYSHLTH', '_DRDXAR1',
'EMPLOY1', 'EDUCA', 'BPHIGH4', '_EDUCAG', '_MICHD', 'QLACTLM2',
'DIABETE3']
selected_data = data[selected_columns]


print(len(selected_data)) # This will print the number of rows
after removing outliers
len(selected_data)
cleaned_data = selected_data.dropna()
len(cleaned_data)
X = cleaned_data[['_RFHLTH', 'PHYSHLTH', '_DRDXAR1', 'EMPLOY1',
'EDUCA', 'BPHIGH4', '_EDUCAG', '_MICHD', 'QLACTLM2',
'DIABETE3']]
y = cleaned_data['GENHLTH']

import statsmodels.api as sm
X = sm.add_constant(X)

import statsmodels.api as sm

def stepwise_selection(X, y, initial_list=[], threshold_in=0.01,
threshold_out=0.05, verbose=True):
included = list(initial_list)
while True:
changed=False
# forward step
excluded = list(set(X.columns)-set(included))
new_pval = pd.Series(index=excluded)
for new_column in excluded:
```

```
model = sm.OLS(y,
sm.add_constant(pd.DataFrame(X[included+[new_column]]))).fit()
new_pval[new_column] = model.pvalues[new_column]
best_pval = new_pval.min()
if best_pval < threshold_in:
best_feature = new_pval.idxmin()
included.append(best_feature)
changed=True
if verbose:
print('Add {:30} with p-value {:.6}'.format(best_feature,
best_pval))

# backward step
model = sm.OLS(y,
sm.add_constant(pd.DataFrame(X[included]))).fit()
pvalues = model.pvalues.iloc[1:]
worst_pval = pvalues.max()
if worst_pval > threshold_out:
changed=True
worst_feature = pvalues.idxmax()
included.remove(worst_feature)
if verbose:
print('Drop {:30} with p-value {:.6}'.format(worst_feature,
worst_pval))
if not changed:
break
return included,model

result,model = stepwise_selection(X, y)

print('resulting features:')
print(result)
```

```
from statsmodels.stats.outliers_influence import
variance_inflation_factor

def check_vif(X):
vif_data = pd.DataFrame()
vif_data["Variable"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i
in range(X.shape[1])]
return vif_data

vif_data = check_vif(X)
print(vif_data)


X = cleaned_data[['_RFHLTH', 'PHYSHLTH', '_DRDXAR1', 'EMPLOY1',
'EDUCA', 'BPHIGH4', '_MICHD', 'QLACTLM2', 'DIABETE3']]
y = cleaned_data['GENHLTH']
X = sm.add_constant(X)
stepwise_selection(X, y, initial_list=[], threshold_in=0.01,
threshold_out=0.05, verbose=True)
vif_data = check_vif(X)
print(vif_data)

new_result,new_model=stepwise_selection(X,y,initial_list=[],
threshold_in=0.01, threshold_out=0.05, verbose=True)
print('resulting features:')
print(result)
print(model.summary())

# %%
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Run regression on training data
model_train = sm.OLS(y_train, sm.add_constant(X_train)).fit()

# Predict on test data
predictions = model_train.predict(sm.add_constant(X_test))

from sklearn.metrics import mean_squared_error

mse = mean_squared_error(y_test, predictions)
print("Mean Squared Error on Test Data:", mse)

print(model.summary())
```