R Notebook: SOC 5800 Team Project

Malcolm S. Townes, Jacob Eikenberry, Daniel Ferris, Nicholas Sokolis

(November 22, 2019)

## Introduction

This is an R Notebook for a study that examines how formerly incarcerated individuals who successfully reintegrated cope with barriers to re-entry. This study was conducted as a team project for the SOC 5800 Survey Design and Sampling course during the Fall 2019 semester at Saint Louis University.

## Project Set Up

The following code chunk enables the R Notebook to integrate seamlessly with the project organization format. This is normally included in the R Notebook to simplify file calls and enable file portability but it has been causing an error. To work around this problem, I’ve embedded the here() function where I enter a file path when necessary.

knitr::opts\_knit$set(root.dir = here::here())

## Load Dependencies

The following code chunk loads package dependencies required to perform the necessary tasks. Basic tasks include importing, reading, wrangling, and cleaning data; selecting a subset of the data; checking for unique observations; analyzing missing data; and performing various types of regression analyses.

library(tidyverse) # loads the basic R packages  
library(here) # enables file portability  
library(readr) # functions for reading data  
library(dplyr) # functions for data wrangling  
library(janitor) # functions for data cleaning  
library(naniar) # functions for analyzing missing data  
library(expss) # functions for calculating on values  
library(ggplot2) # functions for data visualizations  
library(boot) # functions for regression analysis  
library(ordinal) # functions for regression models for ordinal data  
library(MASS) # functions for ordered logistic or probit regression  
library(broom) # functions for tidying ordinal logistic regression models  
library(gvlma) # functions for global validation of linear model assumptions  
library(lmtest) # functions for testing linear regression models  
library(ltm) # functions for latent trait models under Item Response Theory  
library(leaps) # functions for regression subset selection  
library(car) # companion to applied regression  
library(aod) # functions to analyze overdispersed data counts and proportions  
library(pscl) # contains function for pseudo R2 measures for logistic regression  
library(ResourceSelection) # contains function for Hosmer-Lemeshow goodness of fit test  
library(psy) # functions for various procedures used in psychometry  
library(psych) #Procedures for Psychological, Psychometric, and Personality Research

## Load Raw Data

The following code chunk imports the raw data from the csv file.

dataRaw <- read.csv(here("Data","DataRaw","SOC5800\_Data\_NumericValues\_Raw\_CSV.csv"),   
 sep = ",", header = TRUE, fill = TRUE, dec = ".")

## Rename Variables

The following code chunk performs several actions to clean the raw data. It first renames the variables with descriptive camel case names. It then removes unused variables.

dataRaw %>%  
 rename(startDate = StartDate,  
 endDate = EndDate,  
 status = Status,  
 IPaddress = IPAddress,  
 progress = Progress,  
 sessionDuration = Duration..in.seconds.,  
 surveyCompleted = Finished,  
 surveyDate = RecordedDate,  
 responseID = ResponseId,  
 locationLat = LocationLatitude,  
 locationLong = LocationLongitude,  
 consent = Q43,  
 browserName = Q10\_Browser,  
 browserVersion = Q10\_Version,  
 opSyst = Q10\_Operating.System,  
 screenRes = Q10\_Resolution,  
 currentlyIncarcerated = Q1,  
 currentlyDetained = Q2,  
 residentialTreatment = Q3,  
 involCommitment = Q4,  
 gender = Q5,  
 genderSelfDescribe = Q5\_3\_TEXT,  
 transgender = Q7,  
 ethnicityRace = Q6,  
 primaryEthnicityRace = Q6\_10\_TEXT,  
 typeHometown = Q8,  
 religiousAffiliation = Q9,  
 ReligiousOther = Q9\_12\_TEXT,  
 ageNow = Q11,  
 ageRelease = Q13,  
 incarcerationYears = Q14.1\_1,  
 incarcerationMonths = Q14.2\_1,  
 educationLevels = Q15,  
 whenHighestEd = Q16,  
 relationshipStatus = Q17,  
 householdSize = Q18,  
 financialSupportInitial = Q19,  
 financialSupportGovt = Q19\_4\_TEXT,  
 financialSupportNonGovt = Q19\_5\_TEXT,  
 finanicalSupportOther = Q19\_6\_TEXT,  
 selfEmployment = Q20,  
 incomeInitial = Q21,  
 incomeLastYr = Q22,  
 financialStatus = Q23,  
 savings = Q24,  
 pssQ1 = Q41\_1,  
 pssQ2 = Q41\_2,  
 pssQ3 = Q41\_3,  
 pssQ4 = Q41\_4,  
 pssQ5 = Q41\_5,  
 pssQ6 = Q41\_6,  
 pssQ7 = Q41\_7,  
 pssQ8 = Q41\_8,  
 pssQ9 = Q41\_9,  
 pssQ10 = Q41\_10,  
 pssQ11 = Q41\_11,  
 pssQ12 = Q41\_12,  
 supportResidence = Q30\_1,  
 supportJob = Q30\_2,  
 supportAddiction = Q30\_3,  
 supportTransportation = Q30\_4,  
 supportFinancial = Q30\_5,  
 friendResidence = Q31\_1,  
 friendJob = Q31\_2,  
 friendAddiction = Q31\_3,  
 friendTransportation = Q31\_4,  
 friendFinancial = Q31\_5,  
 programsUsed = Q32,  
 importancePublicTrans = Q33\_1,  
 importanceHousing = Q33\_2,  
 importanceSNAP = Q33\_3,  
 importanceWIC = Q33\_4,  
 importanceTANF = Q33\_5,  
 importanceEITC = Q33\_6,  
 importanceJobTraining = Q33\_7,  
 importanceMedicare = Q33\_8,  
 importanceEducAid = Q33\_9,  
 aceQ1 = Q34\_1,  
 aceQ2 = Q34\_2,  
 aceQ3 = Q34\_3,  
 aceQ4 = Q34\_4,  
 aceQ5 = Q34\_5,  
 aceQ6 = Q34\_6,  
 aceQ7 = Q34\_7,  
 aceQ8 = Q34\_8,  
 aceQ9 = Q34\_9,  
 aceQ10 = Q34\_10,  
 helpDuring = Q35,  
 helpDuringType = Q33,  
 helpDuringOther = Q33\_8\_TEXT,  
 helpAfter = Q36,  
 helpAfterType = Q44,  
 helpAfterOther = Q44\_8\_TEXT,  
 gritQ1 = Q35\_1,  
 gritQ2 = Q35\_2,  
 gritQ3 = Q35\_3,  
 gritQ4 = Q35\_4,  
 gritQ5 = Q35\_5,  
 gritQ6 = Q35\_6,  
 gritQ7 = Q35\_7,  
 gritQ8 = Q35\_8,  
 stayInitial = Q36.1,  
 stayInitialOther = Q36\_6\_TEXT,  
 foodSecurityQ1 = Q37\_1,  
 foodSecurityQ2 = Q37\_2,  
 mostHelpful = Q38,  
 greatestObstacle = Q39,  
 helpKind = Q40,  
 justiceInteraction = Q41  
 ) -> dataRenamed  
  
dataRenamed %>%  
 dplyr::select (-c(status, progress, RecipientLastName, RecipientFirstName, RecipientEmail, ExternalReference)) -> dataAugmented

## Change Data Type

The following code chunk changes the data type for certain variables.

dataAugmented$pssQ1 <- as.numeric(dataAugmented$pssQ1)  
dataAugmented$pssQ2 <- as.numeric(dataAugmented$pssQ2)  
dataAugmented$pssQ3 <- as.numeric(dataAugmented$pssQ3)  
dataAugmented$pssQ4 <- as.numeric(dataAugmented$pssQ4)  
dataAugmented$pssQ5 <- as.numeric(dataAugmented$pssQ5)  
dataAugmented$pssQ6 <- as.numeric(dataAugmented$pssQ6)  
dataAugmented$pssQ7 <- as.numeric(dataAugmented$pssQ7)  
dataAugmented$pssQ8 <- as.numeric(dataAugmented$pssQ8)  
dataAugmented$pssQ9 <- as.numeric(dataAugmented$pssQ9)  
dataAugmented$pssQ10 <- as.numeric(dataAugmented$pssQ10)  
dataAugmented$pssQ11 <- as.numeric(dataAugmented$pssQ11)  
dataAugmented$pssQ12 <- as.numeric(dataAugmented$pssQ12)  
dataAugmented$aceQ1 <- as.numeric(dataAugmented$aceQ1)  
dataAugmented$aceQ2 <- as.numeric(dataAugmented$aceQ2)  
dataAugmented$aceQ3 <- as.numeric(dataAugmented$aceQ3)  
dataAugmented$aceQ4 <- as.numeric(dataAugmented$aceQ4)  
dataAugmented$aceQ5 <- as.numeric(dataAugmented$aceQ5)  
dataAugmented$aceQ6 <- as.numeric(dataAugmented$aceQ6)  
dataAugmented$aceQ7 <- as.numeric(dataAugmented$aceQ7)  
dataAugmented$aceQ8 <- as.numeric(dataAugmented$aceQ8)  
dataAugmented$aceQ9 <- as.numeric(dataAugmented$aceQ9)  
dataAugmented$aceQ10 <- as.numeric(dataAugmented$aceQ10)  
dataAugmented$gritQ1 <- as.numeric(dataAugmented$gritQ1)  
dataAugmented$gritQ2 <- as.numeric(dataAugmented$gritQ2)  
dataAugmented$gritQ3 <- as.numeric(dataAugmented$gritQ3)  
dataAugmented$gritQ4 <- as.numeric(dataAugmented$gritQ4)  
dataAugmented$gritQ5 <- as.numeric(dataAugmented$gritQ5)  
dataAugmented$gritQ6 <- as.numeric(dataAugmented$gritQ6)  
dataAugmented$gritQ7 <- as.numeric(dataAugmented$gritQ7)  
dataAugmented$gritQ8 <- as.numeric(dataAugmented$gritQ8)

## Adjust Value Assignments

The following code chunk adjusts the values assigned to fixed pre-determined response choices for select variables and recodes blanks with NA in the dataset.

dataAugmented$ageNow <- (as.numeric(dataAugmented$ageNow)+11)  
dataAugmented$ageRelease <- (as.numeric(dataAugmented$ageRelease)+11)  
dataAugmented[dataAugmented==""] <- NA

## Create Variables

The following code chunk creates additional variables needed for the study.

dataAugmented %>%  
 mutate (mspssScore = case\_when(  
 !is.na(pssQ1)|!is.na(pssQ2)|!is.na(pssQ3)|!is.na(pssQ4)|!is.na(pssQ5)|!is.na(pssQ6)|!is.na(pssQ7)|  
 !is.na(pssQ8)|!is.na(pssQ9)|!is.na(pssQ10)|!is.na(pssQ11)|!is.na(pssQ12) ~  
 (pssQ1+pssQ2+pssQ3+pssQ4+pssQ5+pssQ6+pssQ7+pssQ8+pssQ9+pssQ10+pssQ11+pssQ12)/12  
 ))%>%  
 mutate (aceScore = case\_when (  
 !is.na(aceQ1)|!is.na(aceQ2)|!is.na(aceQ3)|!is.na(aceQ4)|!is.na(aceQ5)|!is.na(aceQ6)|!is.na(aceQ7)|  
 !is.na(aceQ8)|!is.na(aceQ9)|!is.na(aceQ10) ~   
 (aceQ1+aceQ1+aceQ2+aceQ3+aceQ4+aceQ5+aceQ6+aceQ7+aceQ8+aceQ9+aceQ10)  
 )) %>%  
 mutate (gritQ1R = 6-gritQ1, gritQ3R = 6-gritQ3, gritQ5R = 6-gritQ5, gritQ6R = 6-gritQ6) %>%  
 mutate (gritScore = case\_when(  
 !is.na(gritQ1R)|!is.na(gritQ2)|!is.na(gritQ2)|!is.na(gritQ2)|!is.na(gritQ2)|  
 !is.na(gritQ2)|!is.na(gritQ2)|!is.na(gritQ2) ~   
 ((gritQ1R+gritQ2+gritQ3R+gritQ4+gritQ5R+gritQ6R+gritQ7+gritQ8)/8)  
 ))%>%  
 mutate (incarcerationTime = (incarcerationYears\*12)+incarcerationMonths) %>%  
 mutate (povertyRatio = if(as.numeric(incomeLastYr) <= 6) {  
 (8070+(as.numeric(incomeLastYr)\*4420))/(8070+(as.numeric(householdSize)\*4420))  
 } else {  
 if(as.numeric(incomeLastYr) == 7) {  
 (34591/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 8) {  
 (50001/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 9) {  
 (75001/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 10) {  
 (100001/(8070+(as.numeric(householdSize\*4420))))  
 } else {  
 (150000/(8070+(as.numeric(householdSize)\*4420)))  
 }  
 }  
 }  
 }  
 }  
 ) -> dataAugmented

## Select Cases

The following code chunk removes cases where the respondent was part of a protected population or did not provide informed consent.

dataAugmented <- dataAugmented[-c(1,2),]  
dataAugmented %>%  
 subset(currentlyIncarcerated=2) %>%  
 subset(currentlyDetained=2) %>%  
 subset(residentialTreatment=2) %>%  
 subset(involCommitment=2) -> dataClean

## Evaluate Missing Data

The following code chunk evaluates missing data.

miss\_var\_summary(dataClean, order = TRUE)

## # A tibble: 118 x 3  
## variable n\_miss pct\_miss  
## <chr> <int> <dbl>  
## 1 genderSelfDescribe 59 100   
## 2 incarcerationTime 59 100   
## 3 primaryEthnicityRace 56 94.9  
## 4 financialSupportNonGovt 54 91.5  
## 5 selfEmployment 54 91.5  
## 6 helpAfterOther 53 89.8  
## 7 finanicalSupportOther 51 86.4  
## 8 helpDuringOther 51 86.4  
## 9 ReligiousOther 49 83.1  
## 10 stayInitialOther 47 79.7  
## # ... with 108 more rows

miss\_case\_summary(dataClean, order = TRUE)

## # A tibble: 59 x 3  
## case n\_miss pct\_miss  
## <int> <int> <dbl>  
## 1 49 61 51.7  
## 2 50 61 51.7  
## 3 53 48 40.7  
## 4 56 46 39.0  
## 5 51 44 37.3  
## 6 58 23 19.5  
## 7 16 22 18.6  
## 8 57 20 16.9  
## 9 52 19 16.1  
## 10 55 18 15.3  
## # ... with 49 more rows

## Calculate Descriptive Statistics

The following code calculates descriptive statistics for select variables of interest.

print("Number of cases")

## [1] "Number of cases"

nrow(dataClean)

## [1] 59

print("gender; 1 - male, 2 - female, 3 - other, 4 - undisclosed")

## [1] "gender; 1 - male, 2 - female, 3 - other, 4 - undisclosed"

table(as.numeric(dataClean$gender))

##   
## 3 4   
## 31 26

print("ageNow")

## [1] "ageNow"

describe(as.numeric(dataClean$ageNow), na.rm = TRUE)

## vars n mean sd median trimmed mad min max range skew kurtosis se  
## X1 1 59 27.19 7.61 28 27.24 8.9 12 41 29 -0.01 -0.87 0.99

print("ageRelease")

## [1] "ageRelease"

describe(as.numeric(dataClean$ageRelease), na.rm = TRUE)

## vars n mean sd median trimmed mad min max range skew kurtosis se  
## X1 1 59 26.41 9.5 26 26.22 10.38 12 43 31 0.17 -1.14 1.24

print("selfEmployment; 1 - 0%, 2 - 1-25%, 3 - 26-50%, 4 - 51-75%, 5 - 75-99%, 6 - 100%")

## [1] "selfEmployment; 1 - 0%, 2 - 1-25%, 3 - 26-50%, 4 - 51-75%, 5 - 75-99%, 6 - 100%"

table(as.numeric(dataClean$selfEmployment))

##   
## 3 4 5 6   
## 1 1 1 2

## Compute Scale Reliabilities

The following code chunk evaluates the reliability for the MSPSS, ACE, and grit scales by calculating the Cronbach’s alpha for each variable.

reliabilityMSPSS <- cronbach (subset(dataClean, select=c(pssQ1, pssQ2, pssQ3, pssQ4, pssQ5, pssQ6, pssQ7, pssQ8, pssQ9, pssQ10, pssQ11, pssQ12)))  
print("Reliability for MSPSS")

## [1] "Reliability for MSPSS"

reliabilityMSPSS

## $sample.size  
## [1] 59  
##   
## $number.of.items  
## [1] 12  
##   
## $alpha  
## [1] 0.9587711

reliabilityACE <- cronbach (subset(dataClean, select=c(aceQ1, aceQ2, aceQ3, aceQ4, aceQ5, aceQ6, aceQ7, aceQ8, aceQ9, aceQ10)))  
print("Reliability for ACE Scale")

## [1] "Reliability for ACE Scale"

reliabilityACE

## $sample.size  
## [1] 59  
##   
## $number.of.items  
## [1] 10  
##   
## $alpha  
## [1] 0.9634455

reliabilityGrit <- cronbach (subset(dataClean, select=c(gritQ1, gritQ2, gritQ3, gritQ4, gritQ5, gritQ6, gritQ7, gritQ8)))  
print("Reliability for Short Grit Scale")

## [1] "Reliability for Short Grit Scale"

reliabilityGrit

## $sample.size  
## [1] 59  
##   
## $number.of.items  
## [1] 8  
##   
## $alpha  
## [1] 0.906496

## Perform Comparison of Means of Grit

The following code chunk performs calculations for comparison of means of the primary variables of interest.

# Comparison of means for `gritScore` grouped by various operationalizations of success using ANOVA  
# Coding for `gritscore`: values range from 1 (low grit) to 8 (high grit)  
# Null hypothesis: the means of the different groups are the same  
# Alternative hypothesis: the sample mean of at least one group is not equal to the others  
  
  
# Comparison of `gritScore` for cases grouped by `incomeLastYr` using ANOVA  
# Coding for `incomeLastYr` (Qualitrics Q22): ordinal ranging 1 - 10  
groupIncome <- group\_by(dataClean, incomeLastYr)  
  
groupIncome %>%  
 summarise(  
 count = n(),  
 mean = mean(gritScore, na.rm = TRUE),  
 sd = sd(gritScore, na.rm = TRUE)  
 )

## # A tibble: 12 x 4  
## incomeLastYr count mean sd  
## <fct> <int> <dbl> <dbl>  
## 1 1 13 3.16 0.596  
## 2 10 4 3.38 0.941  
## 3 11 2 2.12 0.884  
## 4 2 1 3 NaN   
## 5 3 3 3.38 0.760  
## 6 4 7 2.68 0.710  
## 7 5 3 2.75 0.331  
## 8 6 4 3.78 0.413  
## 9 7 8 2.86 0.398  
## 10 8 7 3.23 0.659  
## 11 9 3 3.67 0.191  
## 12 <NA> 4 3 0

aovGroupIncome <- aov(gritScore ~ as.numeric(incomeLastYr), data = groupIncome)  
summary(aovGroupIncome)

## Df Sum Sq Mean Sq F value Pr(>F)  
## as.numeric(incomeLastYr) 1 0.083 0.0829 0.185 0.669  
## Residuals 53 23.826 0.4495   
## 4 observations deleted due to missingness

# Comparison of `gritScore` for cases grouped by `povertyLevel` using ANOVA  
# Coding for `povertyLevel` (created variable): 1 - above, 0 - below  
dataClean %>%  
 mutate(povertyLevel = if (povertyRatio >1) {1} else {0})-> dataClean  
  
dataClean[dataClean==""] <- NA  
  
groupPoverty <- group\_by(dataClean, povertyLevel)  
  
groupPoverty %>%  
 summarise(  
 count = n(),  
 mean = mean(gritScore, na.rm = TRUE),  
 sd = sd(gritScore, na.rm = TRUE)  
 )

## # A tibble: 1 x 4  
## povertyLevel count mean sd  
## <dbl> <int> <dbl> <dbl>  
## 1 1 59 3.10 0.643

aovGroupPoverty <- aov(gritScore ~ as.numeric(povertyLevel), data = groupPoverty)  
summary(aovGroupPoverty)

## Df Sum Sq Mean Sq F value Pr(>F)  
## Residuals 58 23.95 0.4129

# Comparison of `gritScore` for cases grouped by `financialStatus` using ANOVA  
# Coding for `financialStatus` (Qualitrics Q23): 1 - not able, 2 - just able, 3 - comfortable no savings, 4 - comfortable and saving  
groupFinStat <- group\_by(dataClean, financialStatus)  
  
groupFinStat %>%  
 summarise(  
 count = n(),  
 mean = mean(gritScore, na.rm = TRUE),  
 sd = sd(gritScore, na.rm = TRUE)  
 )

## # A tibble: 5 x 4  
## financialStatus count mean sd  
## <fct> <int> <dbl> <dbl>  
## 1 1 12 3.07 0.602  
## 2 2 19 2.91 0.723  
## 3 3 15 3 0.582  
## 4 4 7 3.71 0.519  
## 5 <NA> 6 3.23 0.357

aovGroupFinStat <- aov(gritScore ~ as.numeric(financialStatus), data = groupFinStat)  
summary(aovGroupFinStat)

## Df Sum Sq Mean Sq F value Pr(>F)   
## as.numeric(financialStatus) 1 1.216 1.2159 2.822 0.0991 .  
## Residuals 51 21.975 0.4309   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## 6 observations deleted due to missingness

# Comparison of `gritScore` for cases grouped by `savings` using ANOVA  
# Coding for `savings` (Qualitrics Q24): 1 - yes, 2 - no  
groupSavings <- group\_by(dataClean, savings)  
  
groupSavings %>%  
 summarise(  
 count = n(),  
 mean = mean(gritScore, na.rm = TRUE),  
 sd = sd(gritScore, na.rm = TRUE)  
 )

## # A tibble: 3 x 4  
## savings count mean sd  
## <fct> <int> <dbl> <dbl>  
## 1 1 15 3.55 0.551  
## 2 2 40 2.93 0.631  
## 3 <NA> 4 3 0

aovGroupSavings <- aov(gritScore ~ as.numeric(savings), data = groupSavings)  
summary(aovGroupSavings)

## Df Sum Sq Mean Sq F value Pr(>F)   
## as.numeric(savings) 1 4.134 4.134 11.08 0.00159 \*\*  
## Residuals 53 19.775 0.373   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## 4 observations deleted due to missingness

## Perform Correlational Analysis

The following code chunk performs correlational analysis for the primary variables of interest.

dataClean %>%  
 dplyr::select(incomeLastYr, povertyRatio, financialStatus, savings, ageNow, ageRelease, mspssScore, aceScore, gritScore) -> correlationDataset   
col\_names <- names(correlationDataset)  
correlationDataset[,col\_names] <- lapply(correlationDataset[,col\_names], as.numeric)  
correlation <- cor(correlationDataset, use="pairwise.complete.obs")  
correlation

## incomeLastYr povertyRatio financialStatus savings  
## incomeLastYr 1.00000000 0.07878926 0.17463014 -0.03251942  
## povertyRatio 0.07878926 1.00000000 -0.12702067 -0.06810200  
## financialStatus 0.17463014 -0.12702067 1.00000000 -0.46516980  
## savings -0.03251942 -0.06810200 -0.46516980 1.00000000  
## ageNow -0.04018701 0.15261350 -0.27436238 0.05829047  
## ageRelease 0.05583711 0.18136159 -0.09029268 -0.04001158  
## mspssScore 0.05025207 0.15372281 0.03607995 -0.09957479  
## aceScore -0.04708187 0.09382109 0.18032901 -0.13713413  
## gritScore 0.05890139 0.06701438 0.22898025 -0.41584262  
## ageNow ageRelease mspssScore aceScore  
## incomeLastYr -0.04018701 0.05583711 0.05025207 -0.04708187  
## povertyRatio 0.15261350 0.18136159 0.15372281 0.09382109  
## financialStatus -0.27436238 -0.09029268 0.03607995 0.18032901  
## savings 0.05829047 -0.04001158 -0.09957479 -0.13713413  
## ageNow 1.00000000 0.35681341 0.29043171 0.34336352  
## ageRelease 0.35681341 1.00000000 0.20048879 0.32248113  
## mspssScore 0.29043171 0.20048879 1.00000000 0.72501825  
## aceScore 0.34336352 0.32248113 0.72501825 1.00000000  
## gritScore 0.08405773 0.08637535 0.07773727 0.07753448  
## gritScore  
## incomeLastYr 0.05890139  
## povertyRatio 0.06701438  
## financialStatus 0.22898025  
## savings -0.41584262  
## ageNow 0.08405773  
## ageRelease 0.08637535  
## mspssScore 0.07773727  
## aceScore 0.07753448  
## gritScore 1.00000000

## Perform Regression Analysis

The following code chunk performs a multiple regression using select variables of interest.

# Multiple regression using `incomeLastYr` as dependent variable  
regressionIncome <- lm(as.numeric(incomeLastYr) ~ as.numeric(ageNow) + as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
 data = dataClean, na.action = na.omit)  
summary(regressionIncome)

##   
## Call:  
## lm(formula = as.numeric(incomeLastYr) ~ as.numeric(ageNow) +   
## as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
## data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -5.2974 -3.8663 0.7448 3.1391 5.0483   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 8.59860 4.97077 1.730 0.090 .  
## as.numeric(ageNow) -0.03346 0.07304 -0.458 0.649   
## as.numeric(ageRelease) 0.03190 0.05683 0.561 0.577   
## mspssScore 0.20104 0.33899 0.593 0.556   
## aceScore -0.07961 0.12416 -0.641 0.524   
## gritScore 0.31846 0.76462 0.416 0.679   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.71 on 49 degrees of freedom  
## (4 observations deleted due to missingness)  
## Multiple R-squared: 0.02039, Adjusted R-squared: -0.07957   
## F-statistic: 0.204 on 5 and 49 DF, p-value: 0.9593

# Multiple regressiong using `povertyRatio` as dependent variable  
regressionPoverty <- lm(as.numeric(povertyRatio) ~ as.numeric(ageNow) + as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
 data = dataClean, na.action = na.omit)  
summary(regressionPoverty)

##   
## Call:  
## lm(formula = as.numeric(povertyRatio) ~ as.numeric(ageNow) +   
## as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
## data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.98140 -0.54312 -0.07029 0.63098 2.31155   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.754583 0.930125 4.037 0.000208 \*\*\*  
## as.numeric(ageNow) 0.010452 0.019030 0.549 0.585540   
## as.numeric(ageRelease) 0.014564 0.015233 0.956 0.344124   
## mspssScore 0.075074 0.087971 0.853 0.397961   
## aceScore -0.007429 0.024228 -0.307 0.760552   
## gritScore 0.029931 0.199807 0.150 0.881593   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9022 on 45 degrees of freedom  
## (8 observations deleted due to missingness)  
## Multiple R-squared: 0.0611, Adjusted R-squared: -0.04323   
## F-statistic: 0.5856 on 5 and 45 DF, p-value: 0.7108

# Multiple regression using `financialStatus` as dependent variable  
regressionFinStat <- lm(as.numeric(financialStatus) ~ as.numeric(ageNow) + as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
 data = dataClean, na.action = na.omit)  
summary(regressionFinStat)

##   
## Call:  
## lm(formula = as.numeric(financialStatus) ~ as.numeric(ageNow) +   
## as.numeric(ageRelease) + mspssScore + aceScore + gritScore,   
## data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.84675 -0.61311 0.05989 0.63595 1.81137   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.97310 1.26608 2.348 0.0231 \*  
## as.numeric(ageNow) -0.03750 0.01841 -2.037 0.0473 \*  
## as.numeric(ageRelease) -0.00679 0.01445 -0.470 0.6406   
## mspssScore -0.03315 0.08794 -0.377 0.7079   
## aceScore 0.04494 0.03175 1.416 0.1635   
## gritScore 0.34415 0.19476 1.767 0.0837 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9336 on 47 degrees of freedom  
## (6 observations deleted due to missingness)  
## Multiple R-squared: 0.1732, Adjusted R-squared: 0.08522   
## F-statistic: 1.969 on 5 and 47 DF, p-value: 0.1008

# Binary logistic regression using `savings` as dependent variable  
# Prepare data  
# Recode `savings` with 1 - yes, 0 - no  
dataClean %>%  
 dplyr::select(incomeLastYr, povertyRatio, povertyLevel, financialStatus, savings, ageNow, ageRelease, mspssScore, aceScore, gritScore) -> dataBinary  
dataBinary$savings <- ifelse(dataBinary$savings==2,0,ifelse(dataBinary$savings==1,1,dataBinary$savings))  
  
# Remove cases with missing data in the `savings` variable  
dataBinary %>%  
 drop\_na(savings) -> dataBinary  
  
logitSavings <- glm(savings ~ ageNow + ageRelease + mspssScore + aceScore + gritScore,   
 data = dataBinary, family = binomial, na.action = na.omit)  
summary(logitSavings)

##   
## Call:  
## glm(formula = savings ~ ageNow + ageRelease + mspssScore + aceScore +   
## gritScore, family = binomial, data = dataBinary, na.action = na.omit)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.2968 -0.6943 -0.4560 0.7516 2.2410   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.167e+01 5.876e+00 -1.985 0.04710 \*   
## ageNow -5.180e-02 5.235e-02 -0.989 0.32243   
## ageRelease -4.366e-04 3.741e-02 -0.012 0.99069   
## mspssScore 2.227e-01 2.620e-01 0.850 0.39531   
## aceScore 1.052e-01 1.379e-01 0.763 0.44548   
## gritScore 2.004e+00 7.271e-01 2.756 0.00585 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 64.455 on 54 degrees of freedom  
## Residual deviance: 51.273 on 49 degrees of freedom  
## AIC: 63.273  
##   
## Number of Fisher Scoring iterations: 5

# Raise e to the coefficients  
print(exp(coef(logitSavings)))

## (Intercept) ageNow ageRelease mspssScore aceScore   
## 8.581692e-06 9.495227e-01 9.995635e-01 1.249473e+00 1.110966e+00   
## gritScore   
## 7.417969e+00

# Obtain various pseudo R-squared measures  
print(pR2(logitSavings))

## llh llhNull G2 McFadden r2ML r2CU   
## -25.6367498 -32.2273940 13.1812884 0.2045044 0.2131045 0.3087469

# Confidence intervals for the coefficients  
print(exp(confint(logitSavings, level = 0.95)))

## 2.5 % 97.5 %  
## (Intercept) 2.106879e-11 0.09929411  
## ageNow 8.506297e-01 1.04978163  
## ageRelease 9.264724e-01 1.07621617  
## mspssScore 7.570155e-01 2.15611806  
## aceScore 8.823115e-01 1.48264118  
## gritScore 2.102437e+00 37.83559357

# Hosmer-Lemeshow Goodness of Fit Test  
# Null hypothesis: the model is a good fit for the data  
# Alternative hypothesis: the model is NOT a good fit for the data  
HosLemLogitSavings <- hoslem.test(dataBinary$savings,   
 fitted(logitSavings), g=10)  
print(HosLemLogitSavings)

##   
## Hosmer and Lemeshow goodness of fit (GOF) test  
##   
## data: dataBinary$savings, fitted(logitSavings)  
## X-squared = 10.872, df = 8, p-value = 0.2091

print(cbind(HosLemLogitSavings$expected, HosLemLogitSavings$observed))

## yhat0 yhat1 y0 y1  
## [0.00249,0.0235] 5.915703 0.08429675 6 0  
## (0.0235,0.0765] 4.833790 0.16621011 5 0  
## (0.0765,0.14] 5.345701 0.65429858 4 2  
## (0.14,0.172] 4.194028 0.80597216 4 1  
## (0.172,0.195] 4.891586 1.10841433 6 0  
## (0.195,0.307] 3.771003 1.22899738 4 1  
## (0.307,0.41] 3.208509 1.79149121 4 1  
## (0.41,0.483] 3.314579 2.68542106 1 5  
## (0.483,0.533] 2.468221 2.53177870 4 1  
## (0.533,0.789] 2.056880 3.94311978 2 4

## Save Data

The following code chunk saves the cleaned data used for the analysis.

write.csv(dataClean, here("Data","DataClean","SOC5800\_Data\_NumericValues\_Clean\_CSV.csv"), append = FALSE)