R Notebook: SOC 5800 Team Project

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

(December 02, 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(generalhoslem) # Hosmer-Lemeshow test for binary and multinomial logistic models  
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 %>%  
 dplyr::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,  
 ethnicityRaceOther = 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,  
 managingFinancially = 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,  
 issFamilyQ1 = Q30\_1,  
 issFamilyQ2 = Q30\_2,  
 issFamilyQ3 = Q30\_3,  
 issFamilyQ4 = Q30\_4,  
 issFamilyQ5 = Q30\_5,  
 issFriendQ1 = Q31\_1,  
 issFriendQ2 = Q31\_2,  
 issFriendQ3 = Q31\_3,  
 issFriendQ4 = Q31\_4,  
 issFriendQ5 = 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$issFamilyQ1 <- as.numeric(dataAugmented$issFamilyQ1)  
dataAugmented$issFamilyQ2 <- as.numeric(dataAugmented$issFamilyQ2)  
dataAugmented$issFamilyQ3 <- as.numeric(dataAugmented$issFamilyQ3)  
dataAugmented$issFamilyQ4 <- as.numeric(dataAugmented$issFamilyQ4)  
dataAugmented$issFamilyQ5 <- as.numeric(dataAugmented$issFamilyQ5)  
dataAugmented$issFriendQ1 <- as.numeric(dataAugmented$issFriendQ1)  
dataAugmented$issFriendQ2 <- as.numeric(dataAugmented$issFriendQ2)  
dataAugmented$issFriendQ3 <- as.numeric(dataAugmented$issFriendQ3)  
dataAugmented$issFriendQ4 <- as.numeric(dataAugmented$issFriendQ4)  
dataAugmented$issFriendQ5 <- as.numeric(dataAugmented$issFriendQ5)  
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)  
  
# Recode variables with `yes` and `no` response options as 1 and 0 respectively  
dataAugmented %>%  
 mutate(savings = ifelse (savings == 1,1,0)) %>%  
 mutate(helpDuring = ifelse(helpDuring == 1,1,0)) %>%  
 mutate(helpAfter = ifelse(helpAfter == 1,1,0)) -> dataAugmented  
# mutate(savings = case\_when(  
# as.numeric(savings) == 1 ~ 1,  
# as.numeric(savings) == 2 ~ 0,  
# TRUE ~ savings)  
# )%>%  
# mutate(helpDuring = case\_when(  
# as.numeric(helpDuring) == 1 ~ 1,  
# as.numeric(helpDuring) == 2 ~ 0,  
# TRUE ~ helpDuring)  
# )%>%  
# mutate(helpAfter = case\_when(  
# as.numeric(helpDuring) == 1 ~ 1,  
# as.numeric(helpDuring) == 2 ~ 0,  
# TRUE ~ helpAfter)  
# )-> dataAugmented  
  
# Recode instances of missing data with `NA`  
dataAugmented[dataAugmented==""] <- NA

## Create Variables

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

dataAugmented %>%  
 # Create variable to consolidate `ethnicityRace` classifications  
 # Coding: 1 - black, 2 - white, 3 - other  
 mutate(ethnicityRaceGrouped = case\_when(  
 ethnicityRace == 4 ~ 1,  
 ethnicityRace == 9 ~ 2,  
 TRUE ~ 3)  
 ) %>%  
   
 # Create variable to consolidate `religiousAffiliation`  
 # Coding: 1 - religious or spiritual, 2 - not religious or spiritual, 3 - prefer not to say  
 mutate(religiousAffilGrouped = case\_when(  
 between (religiousAffiliation, 3, 12) ~ 1,  
 between (religiousAffiliation, 1, 2) ~ 2,  
 religiousAffiliation == 13 ~ 3)  
 ) %>%  
   
 # Create variable to consolidate `typeHometown`  
 # Coding: 1 - less than or equal to 50,000; 2 - greater than 50,000  
 mutate(typeHometownGrouped = case\_when(  
 between (typeHometown, 1, 2) ~ 1,  
 typeHometown == 3 ~ 2)  
 ) %>%  
   
 # Calculate score for multidimensional scale for percieved social support  
 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)  
 )%>%  
   
 # Calculate score for family instrumental social support  
 mutate (issFamilyScore = case\_when(  
 !is.na(issFamilyQ1) & !is.na(issFamilyQ2) & !is.na(issFamilyQ3) & !is.na(issFamilyQ4) & !is.na(issFamilyQ5)  
 ~ (issFamilyQ1+issFamilyQ2+issFamilyQ3+issFamilyQ4+issFamilyQ5)/5)  
 )%>%  
   
 # Calculate score for friend instrumental social support  
 mutate (issFriendScore = case\_when(  
 !is.na(issFriendQ1) & !is.na(issFriendQ2) & !is.na(issFriendQ3) & !is.na(issFriendQ4) & !is.na(issFriendQ5)  
 ~ (issFriendQ1+issFriendQ2+issFriendQ3+issFriendQ4+issFriendQ5)/5)  
 )%>%  
   
 # Calculate score for total instrumental social support  
 mutate(issScore = case\_when(  
 !is.na(issFriendScore) & !is.na(issFamilyScore) ~ (issFamilyScore+issFriendScore)/2)  
 )%>%  
   
 # Calculate score for adverse childhood experience  
 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))  
 ) %>%  
   
 # Calculate socre for grit  
 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(gritQ3R) & !is.na(gritQ4) & !is.na(gritQ5R) &  
 !is.na(gritQ6R) & !is.na(gritQ7) & !is.na(gritQ8) ~   
 ((gritQ1R+gritQ2+gritQ3R+gritQ4+gritQ5R+gritQ6R+gritQ7+gritQ8)/8)  
 ))%>%  
   
 # Calculate total incarceration time in months  
 mutate (incarcerationTime = ((incarcerationYears-1)\*12)+(incarcerationMonths-1)) %>%  
   
 # Calculate total time released  
 mutate (timeReleased = ageNow - ageRelease) %>%  
   
 # Impute 1 (coding for 0%) for `NA` in `selfEmployment` variable  
 mutate (selfEmployment = case\_when(  
 is.na(selfEmployment) ~ 1)  
 )%>%  
   
 # Calculate interval variable for poverty ratio  
 mutate (povertyRatio = if(as.numeric(incomeLastYr) <= 6) {  
 (12490+((as.numeric(incomeLastYr)-1))\*4420)/(8070+(as.numeric(householdSize)\*4420))  
 } else {  
 if(as.numeric(incomeLastYr) == 7) {  
 (59590/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 8) {  
 (62500/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 9) {  
 (87500/(8070+(as.numeric(householdSize)\*4420)))  
 } else {  
 if(as.numeric(incomeLastYr) == 10) {  
 (125000/(8070+(as.numeric(householdSize\*4420))))  
 } else {  
 (150000/(8070+(as.numeric(householdSize)\*4420)))  
 }  
 }  
 }  
 }  
 }  
 ) %>%  
   
 # Calculate change in income  
 mutate(incomeChange = incomeLastYr - incomeInitial) %>%  
   
 # Determine highest education level  
 mutate(educationHighest = sapply(strsplit(as.character(educationLevels), split = ","),   
 function(x) max(as.numeric(x)))) %>%  
   
 # Calculate binary variable indicating food insecurity  
 mutate(foodInsecurity = case\_when(  
 foodSecurityQ1 == 2|foodSecurityQ1 == 3|foodSecurityQ2 == 2|foodSecurityQ2 == 3 ~ 1,  
 foodSecurityQ1 == 1 & foodSecurityQ2 == 1 ~ 0)  
 )%>%  
   
 # Calculate binary variable indicating use of any social services  
 mutate (programsUsedBinary = case\_when(  
 is.na(programsUsed) ~ 0,  
 !is.na(programsUsed) ~ 1)  
 ) %>%  
   
 # Determine number of public and social services used after release  
 mutate(countServicesAfter = count.fields(textConnection(as.character(programsUsed)), sep = ",")) %>%  
   
 # Determine number of social services used during incarceration  
 mutate(countHelpDuring = count.fields(textConnection(as.character(helpDuringType)), sep = ",")) %>%  
   
 # Determine number of social services used after release  
 mutate(countHelpAfter = count.fields(textConnection(as.character(helpAfterType)), sep = ",")) -> dataAugmented

## Select Cases

The following code chunk removes cases where the respondent was part of a protected population or did not provide informed consent. The original data for the variables was not recoded. It then removes duplicate cases.

dataAugmented %>%  
 subset(currentlyIncarcerated=2) %>%  
 subset(currentlyDetained=2) %>%  
 subset(residentialTreatment=2) %>%  
 subset(involCommitment=2) %>%  
 dplyr::distinct(.,.keep\_all = TRUE) -> dataClean  
  
# Case 2 (responseID R\_3ea7BOzjDjNiJYr) and Case 15 (responseID R\_1ouyBTWoxJv4FkS) are the same. Remove case 2 for the final analysis.  
dataClean <- dataClean[-c(2),]

## Evaluate Missing Data

The following code chunk evaluates missing data.

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

## # A tibble: 132 x 3  
## variable n\_miss pct\_miss  
## <chr> <int> <dbl>  
## 1 genderSelfDescribe 58 100   
## 2 ethnicityRaceOther 55 94.8  
## 3 financialSupportNonGovt 53 91.4  
## 4 helpAfterOther 52 89.7  
## 5 helpDuringOther 51 87.9  
## 6 finanicalSupportOther 50 86.2  
## 7 ReligiousOther 48 82.8  
## 8 stayInitialOther 46 79.3  
## 9 financialSupportGovt 39 67.2  
## 10 helpDuringType 33 56.9  
## # ... with 122 more rows

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

## # A tibble: 58 x 3  
## case n\_miss pct\_miss  
## <int> <int> <dbl>  
## 1 48 108 81.8  
## 2 49 108 81.8  
## 3 52 89 67.4  
## 4 55 88 66.7  
## 5 50 83 62.9  
## 6 57 35 26.5  
## 7 56 22 16.7  
## 8 15 21 15.9  
## 9 51 19 14.4  
## 10 54 17 12.9  
## # ... with 48 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] 58

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

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

print("values")

## [1] "values"

table(as.numeric(dataClean$gender))

##   
## 1 2   
## 31 25

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$gender)), margin = NULL)

##   
## 1 2   
## 0.5535714 0.4464286

print("ethnicityRace; 1 - East Asian, 2 - Central Asian, 3 - Southern Asian, 4 - Black or African-American,  
 5 - Hispanic or Latino, 6 - Middle Eastern or North African, 7 - Native American or Alaska,  
 8 - Native Hawaiian or Pacific Islander, 9 - White or European, 10 - Other")

## [1] "ethnicityRace; 1 - East Asian, 2 - Central Asian, 3 - Southern Asian, 4 - Black or African-American,\n 5 - Hispanic or Latino, 6 - Middle Eastern or North African, 7 - Native American or Alaska,\n 8 - Native Hawaiian or Pacific Islander, 9 - White or European, 10 - Other"

print("values")

## [1] "values"

table(as.numeric(dataClean$ethnicityRace))

##   
## 3 4 5 7 9 10   
## 1 13 4 2 33 3

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$ethnicityRace)), margin = NULL)

##   
## 3 4 5 7 9 10   
## 0.01785714 0.23214286 0.07142857 0.03571429 0.58928571 0.05357143

print("religiousAffiliation; 1 - Atheism, 2 - Agnoticism, 3 - Buddhist, 4 - Eastern Orthodox, 5 - Hindu,  
 6 - Islam, 7 - Jewish, 8 - Latter Day Saints, 9 - Non-denominational, 10 - Protestant,  
 11 - Roman Catholic, 12 - Other, 13 - prefer not to answer")

## [1] "religiousAffiliation; 1 - Atheism, 2 - Agnoticism, 3 - Buddhist, 4 - Eastern Orthodox, 5 - Hindu,\n 6 - Islam, 7 - Jewish, 8 - Latter Day Saints, 9 - Non-denominational, 10 - Protestant,\n 11 - Roman Catholic, 12 - Other, 13 - prefer not to answer"

print("values")

## [1] "values"

table(as.numeric(dataClean$religousAffiliation))

## < table of extent 0 >

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$religiousAffiliation)), margin = NULL)

##   
## 1 2 7 8 9 10   
## 0.03571429 0.12500000 0.03571429 0.01785714 0.25000000 0.08928571   
## 11 12 13   
## 0.10714286 0.23214286 0.10714286

print("educationLevels; 1 - high schoor or GED, 2 - trade school, 3 - some college, 4 - associate,  
 5 - bachelor, 6 - master, 7 - doctorate")

## [1] "educationLevels; 1 - high schoor or GED, 2 - trade school, 3 - some college, 4 - associate,\n 5 - bachelor, 6 - master, 7 - doctorate"

print("values")

## [1] "values"

table(as.numeric(dataClean$educationLevels))

##   
## 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17   
## 1 1 1 1 2 2 1 2 1 1 12 2 1 12 9 7

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$educationLevels)), margin = NULL)

##   
## 2 3 4 5 6 7   
## 0.01785714 0.01785714 0.01785714 0.01785714 0.03571429 0.03571429   
## 8 9 10 11 12 13   
## 0.01785714 0.03571429 0.01785714 0.01785714 0.21428571 0.03571429   
## 14 15 16 17   
## 0.01785714 0.21428571 0.16071429 0.12500000

print("whenHighestEd; 1 - before, 2 - during, 3 - after")

## [1] "whenHighestEd; 1 - before, 2 - during, 3 - after"

print("values")

## [1] "values"

table(as.numeric(dataClean$whenHighestEd))

##   
## 1 2 3   
## 10 8 38

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$whenHighestEd)), margin = NULL)

##   
## 1 2 3   
## 0.1785714 0.1428571 0.6785714

print("typeHometown; 1 - rural, 2 - suburban, 3 - urban")

## [1] "typeHometown; 1 - rural, 2 - suburban, 3 - urban"

print("values")

## [1] "values"

table(as.numeric(dataClean$typeHometown))

##   
## 1 2 3   
## 1 20 34

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$typeHometown)), margin = NULL)

##   
## 1 2 3   
## 0.01818182 0.36363636 0.61818182

print("ageNow; ratio data (years)")

## [1] "ageNow; ratio data (years)"

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

## vars n mean sd median trimmed mad min max range skew kurtosis se  
## X1 1 56 43.55 9.13 44 43.07 8.9 25 70 45 0.66 0.7 1.22

print("ageRelease; ratio data (years)")

## [1] "ageRelease; ratio data (years)"

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

## vars n mean sd median trimmed mad min max range skew kurtosis  
## X1 1 53 33.43 10.3 33 32.81 10.38 15 63 48 0.47 -0.16  
## se  
## X1 1.42

print("incarcerationTime; ratio data (months)")

## [1] "incarcerationTime; ratio data (months)"

summary(as.numeric(dataClean$incarcerationTime))

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 10.00 24.00 57.00 90.52 111.00 367.00 25

print("incomeInitial; ordinal data")

## [1] "incomeInitial; ordinal data"

print("values")

## [1] "values"

table(as.numeric(dataClean$incomeInitial))

##   
## 1 2 3 4 5 6 7 8 10 11 12   
## 22 8 7 2 3 3 3 2 1 1 2

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$incomeInitial)), margin = NULL)

##   
## 1 2 3 4 5 6   
## 0.40740741 0.14814815 0.12962963 0.03703704 0.05555556 0.05555556   
## 7 8 10 11 12   
## 0.05555556 0.03703704 0.01851852 0.01851852 0.03703704

print("incomeLastYr; ordinal data")

## [1] "incomeLastYr; ordinal data"

print("values")

## [1] "values"

table(as.numeric(dataClean$incomeLastYr))

##   
## 1 2 3 4 5 6 7 8 9 10 11   
## 12 1 3 7 3 4 8 7 3 4 2

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$incomeLastYr)), margin = NULL)

##   
## 1 2 3 4 5 6   
## 0.22222222 0.01851852 0.05555556 0.12962963 0.05555556 0.07407407   
## 7 8 9 10 11   
## 0.14814815 0.12962963 0.05555556 0.07407407 0.03703704

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%"

print("values")

## [1] "values"

table(as.numeric(dataClean$selfEmployment))

##   
## 1   
## 53

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$selfEmployment)), margin = NULL)

##   
## 1   
## 1

print("foodInsecurity; 1 - yes, 0 - no")

## [1] "foodInsecurity; 1 - yes, 0 - no"

print("values")

## [1] "values"

table(as.numeric(dataClean$foodInsecurity))

##   
## 0 1   
## 31 20

print("proportions")

## [1] "proportions"

prop.table(table(as.numeric(dataClean$foodInsecurity)), margin = NULL)

##   
## 0 1   
## 0.6078431 0.3921569

# count the number of responses to `helpKind` (Qualtrics open-ended question 40)  
print("Number of respondents that answered Q40")

## [1] "Number of respondents that answered Q40"

sum(!is.na(dataClean$helpKind))

## [1] 49

## 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] 51  
##   
## $number.of.items  
## [1] 12  
##   
## $alpha  
## [1] 0.9227266

reliabilityISS <- cronbach (subset(dataClean, select=c(issFamilyQ1, issFamilyQ2, issFamilyQ3, issFamilyQ4, issFamilyQ5,   
 issFriendQ1, issFriendQ2, issFriendQ3, issFriendQ4, issFriendQ5)))  
print("Reliability for Total Instrumental Social Support")

## [1] "Reliability for Total Instrumental Social Support"

reliabilityISS

## $sample.size  
## [1] 51  
##   
## $number.of.items  
## [1] 10  
##   
## $alpha  
## [1] 0.8777059

reliabilityISSfamily <- cronbach (subset(dataClean, select=c(issFamilyQ1, issFamilyQ2, issFamilyQ3, issFamilyQ4, issFamilyQ5)))  
print("Reliability for Family Instrumental Social Support")

## [1] "Reliability for Family Instrumental Social Support"

reliabilityISSfamily

## $sample.size  
## [1] 52  
##   
## $number.of.items  
## [1] 5  
##   
## $alpha  
## [1] 0.828499

reliabilityISSfriend <- cronbach (subset(dataClean, select=c(issFriendQ1, issFriendQ2, issFriendQ3, issFriendQ4, issFriendQ5)))  
print("Reliability for Family Instrumental Social Support")

## [1] "Reliability for Family Instrumental Social Support"

reliabilityISSfriend

## $sample.size  
## [1] 51  
##   
## $number.of.items  
## [1] 5  
##   
## $alpha  
## [1] 0.9320187

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] 50  
##   
## $number.of.items  
## [1] 10  
##   
## $alpha  
## [1] 0.6862534

reliabilityGrit <- cronbach (subset(dataClean, select=c(gritQ1R, gritQ2, gritQ3R, gritQ4, gritQ5R, gritQ6R, gritQ7, gritQ8)))  
print("Reliability for Short Grit Scale")

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

reliabilityGrit

## $sample.size  
## [1] 52  
##   
## $number.of.items  
## [1] 8  
##   
## $alpha  
## [1] 0.7346079

reliabilityHungerVitalSigns <- cronbach (subset(dataClean, select=c(foodSecurityQ1,foodSecurityQ2)))  
print("Reliability for Hunger Vital Signs")

## [1] "Reliability for Hunger Vital Signs"

reliabilityHungerVitalSigns

## $sample.size  
## [1] 50  
##   
## $number.of.items  
## [1] 2  
##   
## $alpha  
## [1] 0.9457329

## 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  
## <int> <int> <dbl> <dbl>  
## 1 1 12 3.71 0.544  
## 2 2 1 3.62 NaN   
## 3 3 3 4 0.760  
## 4 4 7 3.30 0.710  
## 5 5 3 3.38 0.331  
## 6 6 4 4.41 0.413  
## 7 7 8 3.46 0.425  
## 8 8 7 3.90 0.713  
## 9 9 3 4.29 0.191  
## 10 10 4 4 0.941  
## 11 11 2 2.75 0.884  
## 12 NA 4 NaN NaN

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

## Df Sum Sq Mean Sq F value Pr(>F)  
## as.numeric(incomeLastYr) 1 0.055 0.0552 0.121 0.729  
## Residuals 50 22.774 0.4555   
## 6 observations deleted due to missingness

# Comparison of `gritScore` for cases grouped by `povertyLevel` using ANOVA  
# Coding for `povertyLevel` (created variable): 1 - above, 0 - at or below  
dataClean %>%  
 mutate(povertyLevel = if (povertyRatio >1) {1} else {0})-> dataClean  
  
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 0 58 3.71 0.669

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

## Df Sum Sq Mean Sq F value Pr(>F)  
## Residuals 51 22.83 0.4476   
## 6 observations deleted due to missingness

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

## # A tibble: 5 x 4  
## managingFinancially count mean sd  
## <int> <int> <dbl> <dbl>  
## 1 1 11 3.6 0.555   
## 2 2 19 3.54 0.723   
## 3 3 15 3.62 0.604   
## 4 4 7 4.34 0.519   
## 5 NA 6 4.31 0.0884

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

## Df Sum Sq Mean Sq F value Pr(>F)   
## as.numeric(managingFinancially) 1 1.829 1.8292 4.338 0.0426 \*  
## Residuals 48 20.241 0.4217   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## 8 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  
## <dbl> <int> <dbl> <dbl>  
## 1 0 39 3.52 0.625  
## 2 1 15 4.18 0.551  
## 3 NA 4 NaN NaN

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

## Df Sum Sq Mean Sq F value Pr(>F)   
## as.numeric(savings) 1 4.528 4.528 12.37 0.000938 \*\*\*  
## Residuals 50 18.301 0.366   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## 6 observations deleted due to missingness

## Comparison of Means for Categorical Variables

The following code chunk performs comparison of means for select categorical variables of interest.

# Null hypothesis: The two variables are independent  
# Alternative hypothesis: The two variables relate to each other  
  
print("`typeHometown` and `foodInsecurity`")

## [1] "`typeHometown` and `foodInsecurity`"

ChiSqr01 <- chisq.test(dataClean$typeHometown, dataClean$foodInsecurity)  
ChiSqr01

##   
## Pearson's Chi-squared test  
##   
## data: dataClean$typeHometown and dataClean$foodInsecurity  
## X-squared = 0.62557, df = 2, p-value = 0.7314

print("`whenHighestEd` and `incomeLastYr`")

## [1] "`whenHighestEd` and `incomeLastYr`"

ChiSqr02 <- chisq.test(dataClean$whenHighestEd, dataClean$incomeLastYr)  
ChiSqr02

##   
## Pearson's Chi-squared test  
##   
## data: dataClean$whenHighestEd and dataClean$incomeLastYr  
## X-squared = 11.013, df = 20, p-value = 0.9459

## Perform Correlational Analysis

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

dataClean %>%  
 dplyr::select(incomeLastYr, incomeChange, povertyRatio, managingFinancially, savings, programsUsedBinary,  
 countServicesAfter, countHelpDuring, countHelpAfter, educationHighest, ageNow, ageRelease,  
 incarcerationTime, timeReleased, mspssScore, issScore, 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 incomeChange povertyRatio  
## incomeLastYr 1.00000000 0.62389624 0.90245165  
## incomeChange 0.62389624 1.00000000 0.57020204  
## povertyRatio 0.90245165 0.57020204 1.00000000  
## managingFinancially 0.38432546 0.22065969 0.22013358  
## savings 0.16043956 0.02982528 0.09488104  
## programsUsedBinary -0.22587698 0.05293235 -0.11824611  
## countServicesAfter -0.14793655 0.13256880 -0.17969601  
## countHelpDuring -0.13709730 0.05417094 -0.06458020  
## countHelpAfter -0.03217192 0.13784267 -0.05375359  
## educationHighest 0.31339939 0.30114605 0.42784960  
## ageNow -0.09861404 -0.06292118 0.04690360  
## ageRelease -0.23786387 -0.35233526 -0.07209688  
## incarcerationTime -0.09891458 -0.12556572 -0.06405771  
## timeReleased 0.25985930 0.53211785 0.19805433  
## mspssScore 0.05464628 0.02768492 0.02613781  
## issScore -0.14899162 -0.27682815 -0.15198710  
## aceScore 0.12692185 -0.14102817 0.18582176  
## gritScore 0.04919301 0.04883117 0.05971638  
## managingFinancially savings programsUsedBinary  
## incomeLastYr 0.384325456 0.16043956 -0.225876976  
## incomeChange 0.220659690 0.02982528 0.052932352  
## povertyRatio 0.220133577 0.09488104 -0.118246108  
## managingFinancially 1.000000000 0.45930620 0.176565918  
## savings 0.459306203 1.00000000 0.121626064  
## programsUsedBinary 0.176565918 0.12162606 1.000000000  
## countServicesAfter 0.008843988 -0.12089088 0.399697758  
## countHelpDuring -0.064414425 0.08368277 0.190637884  
## countHelpAfter 0.015207940 -0.01324726 0.129766581  
## educationHighest -0.198321714 0.01026097 0.013304718  
## ageNow -0.236078363 -0.07361837 0.062939420  
## ageRelease -0.271761017 -0.04780903 -0.008848668  
## incarcerationTime -0.111169386 0.27761285 0.018434171  
## timeReleased 0.102271985 -0.03908564 0.116335859  
## mspssScore -0.095002081 0.06512421 0.104654047  
## issScore -0.015165964 0.01204938 0.079196581  
## aceScore 0.055259618 0.11663711 0.048335010  
## gritScore 0.287890613 0.44536244 0.335290026  
## countServicesAfter countHelpDuring countHelpAfter  
## incomeLastYr -0.147936545 -0.137097300 -0.03217192  
## incomeChange 0.132568796 0.054170940 0.13784267  
## povertyRatio -0.179696014 -0.064580199 -0.05375359  
## managingFinancially 0.008843988 -0.064414425 0.01520794  
## savings -0.120890877 0.083682767 -0.01324726  
## programsUsedBinary 0.399697758 0.190637884 0.12976658  
## countServicesAfter 1.000000000 0.080303883 0.28444263  
## countHelpDuring 0.080303883 1.000000000 0.31637028  
## countHelpAfter 0.284442629 0.316370283 1.00000000  
## educationHighest -0.027496294 0.069669772 0.12269556  
## ageNow 0.028600170 -0.070744839 0.12001643  
## ageRelease -0.097231840 0.008013068 0.06595764  
## incarcerationTime -0.176209712 0.321056006 0.32682528  
## timeReleased 0.188573528 -0.120538742 0.06289637  
## mspssScore -0.207464572 -0.183472721 -0.07934320  
## issScore -0.373947787 -0.110734804 -0.03954003  
## aceScore -0.094632927 -0.066643726 -0.28117239  
## gritScore -0.094451613 -0.117754040 -0.08905944  
## educationHighest ageNow ageRelease  
## incomeLastYr 0.31339939 -0.09861404 -0.237863865  
## incomeChange 0.30114605 -0.06292118 -0.352335259  
## povertyRatio 0.42784960 0.04690360 -0.072096878  
## managingFinancially -0.19832171 -0.23607836 -0.271761017  
## savings 0.01026097 -0.07361837 -0.047809031  
## programsUsedBinary 0.01330472 0.06293942 -0.008848668  
## countServicesAfter -0.02749629 0.02860017 -0.097231840  
## countHelpDuring 0.06966977 -0.07074484 0.008013068  
## countHelpAfter 0.12269556 0.12001643 0.065957645  
## educationHighest 1.00000000 0.27669591 0.063456462  
## ageNow 0.27669591 1.00000000 0.819674978  
## ageRelease 0.06345646 0.81967498 1.000000000  
## incarcerationTime -0.09093208 0.16732816 0.381308349  
## timeReleased 0.33930490 0.13140627 -0.460151225  
## mspssScore 0.05963715 0.13361740 0.278684650  
## issScore -0.16398329 0.23195723 0.474736479  
## aceScore 0.03506102 0.21313743 0.277311135  
## gritScore 0.16157098 0.07864486 -0.036740293  
## incarcerationTime timeReleased mspssScore issScore  
## incomeLastYr -0.09891458 0.25985930 0.05464628 -0.14899162  
## incomeChange -0.12556572 0.53211785 0.02768492 -0.27682815  
## povertyRatio -0.06405771 0.19805433 0.02613781 -0.15198710  
## managingFinancially -0.11116939 0.10227198 -0.09500208 -0.01516596  
## savings 0.27761285 -0.03908564 0.06512421 0.01204938  
## programsUsedBinary 0.01843417 0.11633586 0.10465405 0.07919658  
## countServicesAfter -0.17620971 0.18857353 -0.20746457 -0.37394779  
## countHelpDuring 0.32105601 -0.12053874 -0.18347272 -0.11073480  
## countHelpAfter 0.32682528 0.06289637 -0.07934320 -0.03954003  
## educationHighest -0.09093208 0.33930490 0.05963715 -0.16398329  
## ageNow 0.16732816 0.13140627 0.13361740 0.23195723  
## ageRelease 0.38130835 -0.46015122 0.27868465 0.47473648  
## incarcerationTime 1.00000000 -0.42252056 -0.05209584 0.26972259  
## timeReleased -0.42252056 1.00000000 -0.26202389 -0.44993734  
## mspssScore -0.05209584 -0.26202389 1.00000000 0.76878108  
## issScore 0.26972259 -0.44993734 0.76878108 1.00000000  
## aceScore 0.09371584 -0.11032828 0.14390670 0.25948482  
## gritScore 0.19083282 0.20551186 0.11307249 0.17331852  
## aceScore gritScore  
## incomeLastYr 0.12692185 0.04919301  
## incomeChange -0.14102817 0.04883117  
## povertyRatio 0.18582176 0.05971638  
## managingFinancially 0.05525962 0.28789061  
## savings 0.11663711 0.44536244  
## programsUsedBinary 0.04833501 0.33529003  
## countServicesAfter -0.09463293 -0.09445161  
## countHelpDuring -0.06664373 -0.11775404  
## countHelpAfter -0.28117239 -0.08905944  
## educationHighest 0.03506102 0.16157098  
## ageNow 0.21313743 0.07864486  
## ageRelease 0.27731114 -0.03674029  
## incarcerationTime 0.09371584 0.19083282  
## timeReleased -0.11032828 0.20551186  
## mspssScore 0.14390670 0.11307249  
## issScore 0.25948482 0.17331852  
## aceScore 1.00000000 0.11183451  
## gritScore 0.11183451 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(programsUsedBinary) + as.numeric(educationHighest) +  
 as.numeric(ageRelease) + as.numeric(timeReleased) + mspssScore + issScore + aceScore + gritScore +   
 gender + as.factor(ethnicityRace) + as.factor(typeHometown),   
 data = dataClean, na.action = na.omit)  
summary(regressionIncome)

##   
## Call:  
## lm(formula = as.numeric(incomeLastYr) ~ as.numeric(programsUsedBinary) +   
## as.numeric(educationHighest) + as.numeric(ageRelease) + as.numeric(timeReleased) +   
## mspssScore + issScore + aceScore + gritScore + gender + as.factor(ethnicityRace) +   
## as.factor(typeHometown), data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.8775 -1.1671 -0.0113 1.1707 4.5163   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 8.12540 6.48943 1.252 0.22168   
## as.numeric(programsUsedBinary) -9.73798 3.08972 -3.152 0.00406 \*\*  
## as.numeric(educationHighest) 0.65963 0.54011 1.221 0.23294   
## as.numeric(ageRelease) -0.04157 0.07169 -0.580 0.56699   
## as.numeric(timeReleased) 0.05839 0.11419 0.511 0.61342   
## mspssScore 0.48666 0.52893 0.920 0.36599   
## issScore -1.01896 1.30066 -0.783 0.44046   
## aceScore 0.14821 0.19029 0.779 0.44309   
## gritScore 0.82624 0.71421 1.157 0.25785   
## gender -1.25095 1.09192 -1.146 0.26238   
## as.factor(ethnicityRace)4 0.18234 3.18524 0.057 0.95479   
## as.factor(ethnicityRace)5 2.85790 3.54928 0.805 0.42800   
## as.factor(ethnicityRace)7 4.43753 3.61743 1.227 0.23093   
## as.factor(ethnicityRace)9 1.00781 3.19963 0.315 0.75529   
## as.factor(ethnicityRace)10 3.87931 3.93188 0.987 0.33292   
## as.factor(typeHometown)3 -1.25009 1.09028 -1.147 0.26200   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.646 on 26 degrees of freedom  
## (16 observations deleted due to missingness)  
## Multiple R-squared: 0.5377, Adjusted R-squared: 0.2709   
## F-statistic: 2.016 on 15 and 26 DF, p-value: 0.05661

# Multiple regression using `incomeRelease` as dependent variable  
regressionIncomeInitial <- lm(as.numeric(incomeInitial) ~ as.numeric(programsUsedBinary) + as.numeric(educationHighest) +  
 as.numeric(ageRelease) + as.numeric(timeReleased) + mspssScore + issScore + aceScore + gritScore +   
 gender + as.factor(ethnicityRace) + as.factor(typeHometown),   
 data = dataClean, na.action = na.omit)  
summary(regressionIncomeInitial)

##   
## Call:  
## lm(formula = as.numeric(incomeInitial) ~ as.numeric(programsUsedBinary) +   
## as.numeric(educationHighest) + as.numeric(ageRelease) + as.numeric(timeReleased) +   
## mspssScore + issScore + aceScore + gritScore + gender + as.factor(ethnicityRace) +   
## as.factor(typeHometown), data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.1480 -1.7275 -0.1354 1.0560 5.6684   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.10622 6.35043 0.174 0.86306   
## as.numeric(programsUsedBinary) -9.96544 3.02354 -3.296 0.00284 \*\*  
## as.numeric(educationHighest) 0.90063 0.52854 1.704 0.10031   
## as.numeric(ageRelease) -0.07948 0.07016 -1.133 0.26759   
## as.numeric(timeReleased) -0.21308 0.11174 -1.907 0.06764 .   
## mspssScore -0.71207 0.51760 -1.376 0.18065   
## issScore 1.71404 1.27280 1.347 0.18971   
## aceScore 0.45570 0.18621 2.447 0.02146 \*   
## gritScore 0.91985 0.69892 1.316 0.19962   
## gender 2.22037 1.06853 2.078 0.04773 \*   
## as.factor(ethnicityRace)4 -3.07278 3.11701 -0.986 0.33331   
## as.factor(ethnicityRace)5 -2.13824 3.47326 -0.616 0.54349   
## as.factor(ethnicityRace)7 -3.01353 3.53995 -0.851 0.40238   
## as.factor(ethnicityRace)9 -3.16646 3.13109 -1.011 0.32119   
## as.factor(ethnicityRace)10 1.44061 3.84766 0.374 0.71113   
## as.factor(typeHometown)3 -1.13102 1.06693 -1.060 0.29886   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.589 on 26 degrees of freedom  
## (16 observations deleted due to missingness)  
## Multiple R-squared: 0.5697, Adjusted R-squared: 0.3215   
## F-statistic: 2.295 on 15 and 26 DF, p-value: 0.03053

# Multiple regressiong using `povertyRatio` as dependent variable and original independent variables  
regressionPoverty <- lm(as.numeric(povertyRatio) ~ as.numeric(programsUsedBinary) + as.numeric(ageRelease) +   
 as.numeric(timeReleased) + mspssScore + issScore + aceScore + gritScore +   
 gender + as.factor(ethnicityRace) + as.factor(typeHometown),   
 data = dataClean, na.action = na.omit)  
summary(regressionPoverty)

##   
## Call:  
## lm(formula = as.numeric(povertyRatio) ~ as.numeric(programsUsedBinary) +   
## as.numeric(ageRelease) + as.numeric(timeReleased) + mspssScore +   
## issScore + aceScore + gritScore + gender + as.factor(ethnicityRace) +   
## as.factor(typeHometown), data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.56012 -0.19372 -0.05032 0.07849 0.95970   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.944571 0.850680 1.110 0.2794   
## as.numeric(programsUsedBinary) -0.974995 0.493850 -1.974 0.0616 .  
## as.numeric(ageRelease) 0.008536 0.011361 0.751 0.4608   
## as.numeric(timeReleased) 0.003337 0.017187 0.194 0.8479   
## mspssScore 0.194719 0.086430 2.253 0.0351 \*  
## issScore -0.478596 0.216048 -2.215 0.0379 \*  
## aceScore 0.013328 0.032891 0.405 0.6894   
## gritScore 0.279716 0.130570 2.142 0.0440 \*  
## gender -0.329910 0.179168 -1.841 0.0797 .  
## as.factor(ethnicityRace)4 -0.215592 0.480905 -0.448 0.6585   
## as.factor(ethnicityRace)5 0.100089 0.548937 0.182 0.8571   
## as.factor(ethnicityRace)7 0.904504 0.590565 1.532 0.1406   
## as.factor(ethnicityRace)9 0.043267 0.470499 0.092 0.9276   
## as.factor(ethnicityRace)10 -0.122368 0.551673 -0.222 0.8266   
## as.factor(typeHometown)3 0.159810 0.189103 0.845 0.4076   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4014 on 21 degrees of freedom  
## (22 observations deleted due to missingness)  
## Multiple R-squared: 0.5009, Adjusted R-squared: 0.1682   
## F-statistic: 1.506 on 14 and 21 DF, p-value: 0.1927

# Multiple regressiong using `povertyRatio` as dependent variable  
regressionPoverty <- lm(as.numeric(povertyRatio) ~ as.numeric(programsUsedBinary) + as.numeric(educationHighest) +  
 as.numeric(ageRelease) + as.numeric(timeReleased) + mspssScore + issScore + aceScore + gritScore +   
 gender + as.factor(ethnicityRace) + as.factor(typeHometown),   
 data = dataClean, na.action = na.omit)  
summary(regressionPoverty)

##   
## Call:  
## lm(formula = as.numeric(povertyRatio) ~ as.numeric(programsUsedBinary) +   
## as.numeric(educationHighest) + as.numeric(ageRelease) + as.numeric(timeReleased) +   
## mspssScore + issScore + aceScore + gritScore + gender + as.factor(ethnicityRace) +   
## as.factor(typeHometown), data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.45426 -0.20233 -0.02155 0.08008 0.91427   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 0.405362 1.065902 0.380 0.708  
## as.numeric(programsUsedBinary) -0.856495 0.516347 -1.659 0.113  
## as.numeric(educationHighest) 0.082544 0.097155 0.850 0.406  
## as.numeric(ageRelease) 0.006677 0.011644 0.573 0.573  
## as.numeric(timeReleased) 0.000190 0.017694 0.011 0.992  
## mspssScore 0.142750 0.106357 1.342 0.195  
## issScore -0.369032 0.252849 -1.459 0.160  
## aceScore 0.011504 0.033181 0.347 0.732  
## gritScore 0.224890 0.146430 1.536 0.140  
## gender -0.264601 0.196064 -1.350 0.192  
## as.factor(ethnicityRace)4 -0.032411 0.529961 -0.061 0.952  
## as.factor(ethnicityRace)5 0.384826 0.646291 0.595 0.558  
## as.factor(ethnicityRace)7 0.945278 0.596449 1.585 0.129  
## as.factor(ethnicityRace)9 0.235131 0.524725 0.448 0.659  
## as.factor(ethnicityRace)10 0.195994 0.669952 0.293 0.773  
## as.factor(typeHometown)3 0.145243 0.191138 0.760 0.456  
##   
## Residual standard error: 0.4041 on 20 degrees of freedom  
## (22 observations deleted due to missingness)  
## Multiple R-squared: 0.5183, Adjusted R-squared: 0.1571   
## F-statistic: 1.435 on 15 and 20 DF, p-value: 0.2224

# Multiple regression using `managingFinancially` as dependent variable  
regressionFinStat <- lm(as.numeric(managingFinancially) ~ as.numeric(programsUsedBinary) + as.numeric(educationHighest) +  
 as.numeric(ageRelease) + as.numeric(timeReleased) + mspssScore + issScore + aceScore + gritScore +   
 gender + as.factor(ethnicityRace) + as.factor(typeHometown),   
 data = dataClean, na.action = na.omit)  
summary(regressionFinStat)

##   
## Call:  
## lm(formula = as.numeric(managingFinancially) ~ as.numeric(programsUsedBinary) +   
## as.numeric(educationHighest) + as.numeric(ageRelease) + as.numeric(timeReleased) +   
## mspssScore + issScore + aceScore + gritScore + gender + as.factor(ethnicityRace) +   
## as.factor(typeHometown), data = dataClean, na.action = na.omit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.3881 -0.5929 0.0000 0.5192 1.6202   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.82805 2.43535 1.161 0.2565   
## as.numeric(programsUsedBinary) -0.88628 1.15975 -0.764 0.4519   
## as.numeric(educationHighest) -0.16886 0.20282 -0.833 0.4130   
## as.numeric(ageRelease) -0.04486 0.02707 -1.657 0.1100   
## as.numeric(timeReleased) -0.02620 0.04304 -0.609 0.5482   
## mspssScore -0.06292 0.19857 -0.317 0.7540   
## issScore 0.02594 0.48992 0.053 0.9582   
## aceScore 0.07418 0.07346 1.010 0.3223   
## gritScore 0.50799 0.26889 1.889 0.0705 .  
## gender -0.40556 0.41052 -0.988 0.3327   
## as.factor(ethnicityRace)4 0.46327 1.19571 0.387 0.7017   
## as.factor(ethnicityRace)5 0.86675 1.33234 0.651 0.5213   
## as.factor(ethnicityRace)7 1.37121 1.35766 1.010 0.3222   
## as.factor(ethnicityRace)9 0.75639 1.20100 0.630 0.5345   
## as.factor(ethnicityRace)10 1.84940 1.47664 1.252 0.2220   
## as.factor(typeHometown)3 -0.15432 0.41779 -0.369 0.7150   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9929 on 25 degrees of freedom  
## (17 observations deleted due to missingness)  
## Multiple R-squared: 0.4252, Adjusted R-squared: 0.08039   
## F-statistic: 1.233 on 15 and 25 DF, p-value: 0.312

# Binary logistic regression using `savings` as dependent variable  
# Select variables  
dataClean %>%  
 dplyr::select(incomeLastYr, povertyRatio, povertyLevel, managingFinancially, savings, programsUsedBinary, educationHighest,   
 ageRelease, timeReleased, gender, mspssScore, issScore, aceScore, gritScore) -> dataBinary  
  
# Remove cases with missing data  
dataBinary %>%  
 drop\_na() -> dataBinary  
  
logitSavings <- glm(savings ~ programsUsedBinary + managingFinancially + educationHighest + ageRelease + timeReleased +   
 mspssScore + issScore + aceScore + gritScore + gender,   
 data = dataBinary, family = binomial, na.action = na.omit)  
summary(logitSavings)

##   
## Call:  
## glm(formula = savings ~ programsUsedBinary + managingFinancially +   
## educationHighest + ageRelease + timeReleased + mspssScore +   
## issScore + aceScore + gritScore + gender, family = binomial,   
## data = dataBinary, na.action = na.omit)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.57792 -0.70726 -0.21377 0.03876 2.06300   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -21.14569 2399.55174 -0.009 0.9930   
## programsUsedBinary 10.09311 2399.54666 0.004 0.9966   
## managingFinancially 1.10771 0.74700 1.483 0.1381   
## educationHighest -0.07757 0.61429 -0.126 0.8995   
## ageRelease -0.07614 0.10066 -0.756 0.4494   
## timeReleased -0.18466 0.14547 -1.269 0.2043   
## mspssScore 0.53322 0.61093 0.873 0.3828   
## issScore -1.00839 1.51181 -0.667 0.5048   
## aceScore 0.10190 0.23473 0.434 0.6642   
## gritScore 2.83896 1.30992 2.167 0.0302 \*  
## gender -0.36125 1.23048 -0.294 0.7691   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 40.488 on 35 degrees of freedom  
## Residual deviance: 24.717 on 25 degrees of freedom  
## AIC: 46.717  
##   
## Number of Fisher Scoring iterations: 15

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

## (Intercept) programsUsedBinary managingFinancially   
## 6.554558e-10 2.417591e+04 3.027406e+00   
## educationHighest ageRelease timeReleased   
## 9.253636e-01 9.266829e-01 8.313855e-01   
## mspssScore issScore aceScore   
## 1.704419e+00 3.648058e-01 1.107277e+00   
## gritScore gender   
## 1.709798e+01 6.968024e-01

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

## llh llhNull G2 McFadden r2ML r2CU   
## -12.3585309 -20.2440652 15.7710685 0.3895233 0.3547292 0.5253376

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

## 2.5 % 97.5 %  
## (Intercept) NA 2.313959e+72  
## programsUsedBinary 6.829248e-78 NA  
## managingFinancially 8.331901e-01 1.774099e+01  
## educationHighest 2.637166e-01 3.296698e+00  
## ageRelease 7.014135e-01 1.085638e+00  
## timeReleased 5.632648e-01 1.042734e+00  
## mspssScore 5.414062e-01 6.728663e+00  
## issScore 1.345302e-02 7.175855e+00  
## aceScore 6.939336e-01 1.850831e+00  
## gritScore 2.116890e+00 5.712075e+02  
## gender 4.250820e-02 6.962757e+00

# 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 = 4.5237, df = 8, p-value = 0.8071

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

## yhat0 yhat1 y0 y1  
## [6.39e-08,0.0151] 3.9813447 0.01865529 4 0  
## (0.0151,0.022] 3.9242434 0.07575658 4 0  
## (0.022,0.0266] 2.9308552 0.06914485 3 0  
## (0.0266,0.0589] 3.8495395 0.15046053 4 0  
## (0.0589,0.169] 2.7325412 0.26745881 2 1  
## (0.169,0.262] 3.0601916 0.93980839 4 0  
## (0.262,0.323] 2.1215166 0.87848338 2 1  
## (0.323,0.412] 2.5321044 1.46789564 2 2  
## (0.412,0.688] 1.2782600 1.72174001 1 2  
## (0.688,0.988] 0.5894034 3.41059660 1 3

## 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)