R Survey Cheat Sheet

David J. Barney

1/26/2019

Reading & Exploring Data

To read in your data:

```
# Comma delimited format
dat <- read.csv("~/your/file/path")
# Tab delimited format
dat <- read.table("~/your/file/path")
# Stata (.dta) format
library(haven)
dat <- read_dta("~/your/file/path")
```

To view observations or summaries of a variable:

```
# Print the top few observations of a variable
head(dataframe$vector)
# Print the bottom few observations of a variable
tail(dataframe$vector)
# Print all observations of a variable
print(dataframe$vector)
# Access a summary of a variable or model
summary(dataframe$vector)
summary(model)
```

To learn the structure or format of your data:

```
# Access the type / storage mode of the data
typeof(dataframe$vector)
# Access the structure of the data
str(dataframe)
str(dataframe$vector)
# Access the length of a vector (e.g. the number of observations)
length(dataframe$vector)
# Access the attributes and metadata of an object
attributes(dataframe)
attributes(dataframe$vector)
```

All of the examples in this guide will use the 2016 CCES loaded in `.dta` format. To replicate, simply update the relative filepath below to the location of your local version of the dataset.

```
library(haven)
cces16 <- read_dta("../../../Data/2016 CCES/CCES16_Common_OUTPUT_Feb2018_VV.dta")
```

Manipulating Data

Recoding Data

To recode values in base R:

```
#Create a new vector to work with
cces16$ban_ar <- cces16$CC16_330d
#Attitudes toward gun control
#Call all values for "oppose" and replace with zero
cces16$ban_ar[cces16$ban_ar==2] <- 0
#Call all values for skipped / not asked and replace with missing
cces16$ban_ar[cces16$ban_ar==8] <- NA
cces16$ban_ar[cces16$ban_ar==9] <- NA
```

To collapse categories of an ordinal variable in base R:

```
#Create a new vector to work with
cces16$pid <- cces16$pid7
#Recode values to missing
cces16$pid[cces16$pid==98] <- NA
cces16$pid[cces16$pid==99] <- NA
cces16$pid[cces16$pid==8] <- NA
#Collapse the categories
cces16$pid[cces16$pid==2] <- 1
cces16$pid[cces16$pid==3] <- 1
cces16$pid[cces16$pid==4] <- 2
cces16$pid[cces16$pid==5] <- 3
cces16$pid[cces16$pid==6] <- 3
cces16$pid[cces16$pid==7] <- 3
```

To cut a continuous variable into an ordinal one in base R:

```
# Create the age variable
cces16$age <- 2016 - cces16$birthyr

cces16$agecats <- cut(cces16$age,
                      breaks=c(-Inf, 35, 50, Inf),
                      labels=c("35 and Under","36 to 50","Over 50"))
```

To flip the direction of coding in base R:

```
#Flip the coding
cces16$pid_reverse <- 4 - cces16$pid
```

To apply labels to factor levels of a recoded variable:

```
cces16$pid_reverse <- factor(cces16$pid_reverse,
                             levels = c(1,2,3),
                             labels = c("Republican", "Independent",
                             "Democrat"))
```

Merging Data

Again, to replicate, update the filepath below to point to your local copy of the supplementary dataset for merging. To merge data that have one common vector with the same name:

```r
#Load the supplemental data
cces16s <-  read.csv("../../../Data/2016 CCES Supplementary/CC16_Candidates_By_Race.csv")
#Merge the primary and supplemental data by respondent state
cces16c <- merge(cces16, cces16s)
```

To merge data that have a common vector with different names:

```r
#Merge by state with different column names
rm(cces16c)
cces16c <- merge(cces16, cces16s,
                 by.x = "inputstate",
                 by.y = "inputstate")
```

#Descriptive Statistics ## Summary Statistics To summarize a variable:

```r
summary(cces16$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   33.00   49.00   47.88   61.00   99.00
```

To call specific summary statistics:

```r
#Mean
mean(cces16$age)
```

```
## [1] 47.88014
```

```r
#Standard deviation
sd(cces16$age)
```

```
## [1] 16.83279
```

```r
#Minimum
min(cces16$age)
```

```
## [1] 18
```

```r
#Maximum
max(cces16$age)
```

```
## [1] 99
```

```r
#Range
range(cces16$age)
```

```
## [1] 18 99
```

```r
#Quantiles
quantile(cces16$age)
```

```
## <Labelled double>: Birth year
##   0%  25%  50%  75% 100%
##   18   33   49   61   99
##
## Labels:
##  value     label
##   9998   Skipped
##   9999 Not Asked
```

Tabulations & Cross Tabulations

To tabulate a variable:

```
# Tabulate PID
prop.table(table(cces16$pid_reverse))
```

```
##
##  Republican Independent    Democrat
##   0.3336641   0.1679444   0.4983915
```

To cross-tabulate two variables:

```
# Tabulate PID by age categories
pidxage <- table(cces16$pid_reverse, cces16$agecats)
pidxage
```

```
##
##               35 and Under 36 to 50 Over 50
##   Republican          4799     4470   11578
##   Independent         3137     2660    4696
##   Democrat           10427     7059   13653
```

```
prop.table(pidxage,2)
```

```
##
##               35 and Under  36 to 50   Over 50
##   Republican     0.2613407 0.3150328 0.3868747
##   Independent    0.1708327 0.1874692 0.1569152
##   Democrat       0.5678266 0.4974981 0.4562101
```

Summary Statistics by Group

To summarize variables by group in base R:

```
#Create a dataframe of the variables of interest
subgroup_vars <- c("age","pid")
subgroup_matrix <- as.matrix(cces16[subgroup_vars])
subgroup_df <- as.data.frame(subgroup_matrix)
aggregate(subgroup_df$age, list(subgroup_df$pid), mean)
```

```
##   Group.1        x
## 1       1 46.58152
## 2       2 46.97732
## 3       3 51.23725
```

To summarize variables by group with `dplyr`:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
```

```
##     intersect, setdiff, setequal, union
```

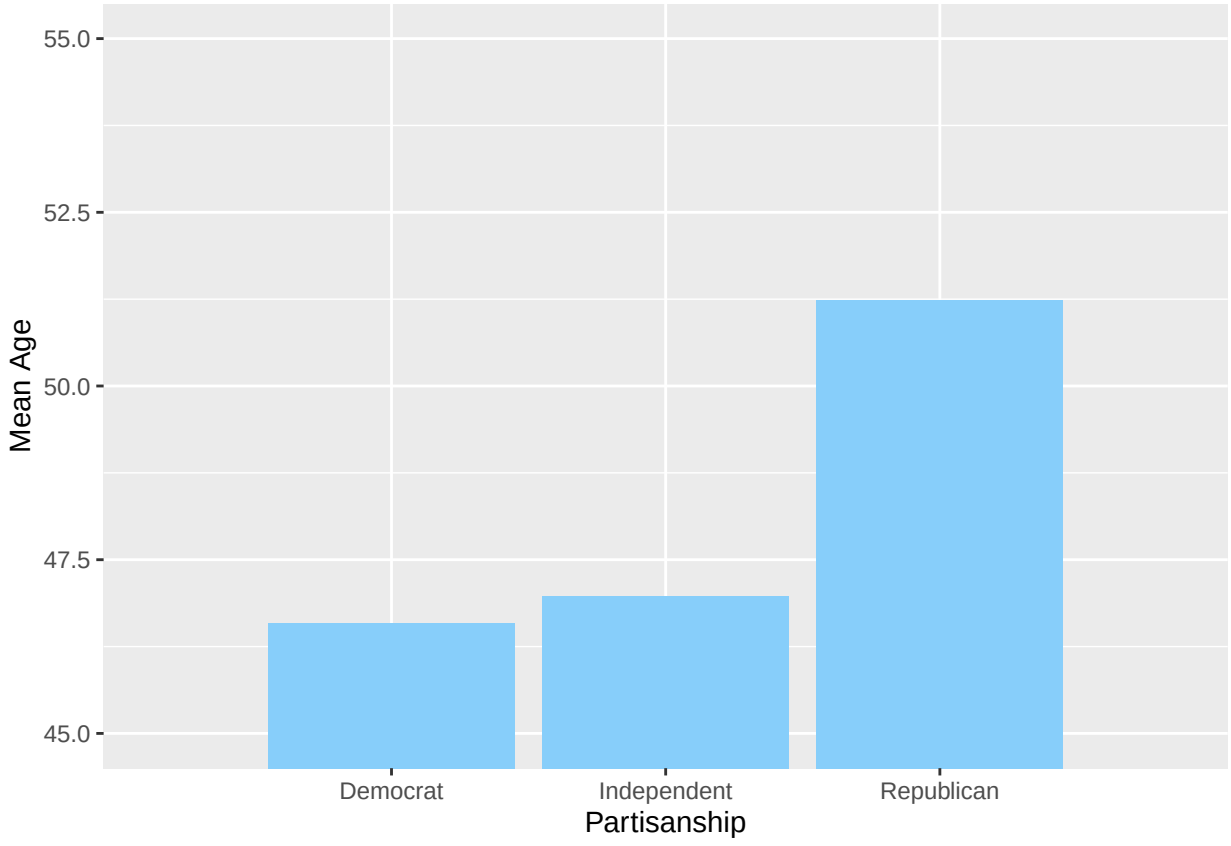
```
subgroup_means <- subgroup_df %>%
  group_by(pid) %>%
  summarise(mean = mean(age))
subgroup_means
```

```
## # A tibble: 4 x 2
##     pid  mean
##   <dbl> <dbl>
## 1     1  46.6
## 2     2  47.0
## 3     3  51.2
## 4    NA  38.4
```

To visualize subgroup means with `ggplot`:

```
library(ggplot2)
pid_labels <- c("Democrat", "Independent", "Republican")
sg_bp <- ggplot(subgroup_means, aes(y=mean, x=pid)) +
  geom_bar(fill="lightskyblue",stat="identity") +
  #xlab("Partisanship") +
  ylab("Mean Age") +
  scale_x_discrete(name = "Partisanship",
                   limits=pid_labels)
sg_bp <- sg_bp + coord_cartesian(ylim=c(45,55))
sg_bp
```

```
## Warning: Removed 1 rows containing missing values (position_stack).
```



Distributions

To plot a histogram of a variable's distribution using `ggplot2`:

```
#Histogram for age
library(ggplot2)
age_matrix <- as.matrix(cces16$age)
age_df <- as.data.frame(age_matrix)
ggplot(data = age_df, aes(x=age_df$V1)) +
  geom_histogram(binwidth = 1) +
  xlab("Age of Respondent") +
  ylab("Frequency")
```


```
#Density plot
ggplot(data = age_df, aes(x=age_df$V1)) +
  geom_density(fill="lightblue") +
  xlab("Age of Respondent") +
  ylab("Density")
```

Modeling

OLS Regression

```
# Prepare Obama approval as the DV
cces16$oa <- cces16$CC16_320a
cces16$oa[cces16$oa=="5"] <- NA
cces16$oa[cces16$oa=="8"] <- NA
cces16$oa <- 5 - cces16$oa
# Fit the OLS model
olsfit <- lm(oa ~ pid + gender + age,
             data = cces16)
# Print the model summary
summary(olsfit)
```

```
##
## Call:
## lm(formula = oa ~ pid + gender + age, data = cces16)
##
## Residuals:
## <Labelled double>: Approve of Job - Obama
##      Min       1Q   Median       3Q      Max
## -2.58510 -0.41900 -0.21139  0.61911  2.95471
##
```

```
## Labels:
##  value               label
##      1    Strongly approve
##      2    Somewhat approve
##      3 Somewhat disapprove
##      4 Strongly disapprove
##      5            Not sure
##      8             Skipped
##      9           Not Asked
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  4.6247560  0.0164018  281.967  < 2e-16 ***
## pid         -1.0051598  0.0038097 -263.844  < 2e-16 ***
## gender       0.0450371  0.0068463    6.578  4.8e-11 ***
## age         -0.0069207  0.0002041  -33.903  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8351 on 60142 degrees of freedom
##   (4454 observations deleted due to missingness)
## Multiple R-squared:  0.5542, Adjusted R-squared:  0.5542
## F-statistic: 2.493e+04 on 3 and 60142 DF,  p-value: < 2.2e-16
```

Logistic Regression

```
## Logistic Regression
# Prepare preference for AR ban as DV
cces16$ban_ar <- cces16$CC16_330d
cces16$ban_ar[cces16$ban_ar==2] <- 0
cces16$ban_ar[cces16$ban_ar==8] <- NA
cces16$ban_ar[cces16$ban_ar==9] <- NA
# Fit the logit model
lfit <- glm(ban_ar ~ pid + agecats + gender,
           data=cces16, family = binomial())
# Print the model summary
summary(lfit)
```

```
##
## Call:
## glm(formula = ban_ar ~ pid + agecats + gender, family = binomial(),
##     data = cces16)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2356  -0.9485   0.4895   0.8301   1.7229
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      1.08299    0.03785   28.61   <2e-16 ***
## pid             -1.05779    0.01104  -95.85   <2e-16 ***
## agecats36 to 50  0.30957    0.02638   11.74   <2e-16 ***
## agecatsOver 50   0.66148    0.02279   29.03   <2e-16 ***
```

```
## gender           0.86323    0.01924   44.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 79067  on 62049  degrees of freedom
## Residual deviance: 66063  on 62045  degrees of freedom
##   (2550 observations deleted due to missingness)
## AIC: 66073
##
## Number of Fisher Scoring iterations: 4
```

Ordinal Logistic Regression

```
## Ordinal Logistic Regression
# Convert Obama approval to a factor
cces16$oaf <- factor(cces16$oa)
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
# Fit the ordinal logit model
olfit <- polr(oaf ~ pid + gender + age,
              data = cces16, Hess = TRUE)
# Print the model summary
summary(olfit)
```

```
## Call:
## polr(formula = oaf ~ pid + gender + age, data = cces16, Hess = TRUE)
##
## Coefficients:
##           Value Std. Error  t value
## pid    -2.01316  0.0121435 -165.781
## gender  0.14036  0.0167348    8.387
## age    -0.01447  0.0005025  -28.794
##
## Intercepts:
##     Value     Std. Error t value
## 1|2   -5.2717    0.0459  -114.8740
## 2|3   -4.3865    0.0436  -100.6132
## 3|4   -2.6727    0.0406   -65.7525
##
## Residual Deviance: 116936.77
## AIC: 116948.77
## (4454 observations deleted due to missingness)
```

Multinomial Logistic Regression

```
## Multinomial Logistic Regression
## Create a 2012 vote choice factor variable
cces16$vc <- cces16$CC16_326
cces16$vc[cces16$vc=="4"] <- NA
cces16$vc[cces16$vc=="5"] <- NA
cces16$vc <- factor(cces16$vc,
                    levels = c(1,2,3),
                    labels = c("Obama", "Romney","Other"))

library(nnet)
# Fit the multinomial logit model
mlfit <- multinom(vc ~ pid + agecats + gender,
                  data=cces16)
```

```
## # weights:  18 (10 variable)
## initial  value 51855.598637
## iter  10 value 23914.700591
## iter  20 value 18733.374011
## final  value 18733.298819
## converged
```

```
# weights:  27 (16 variable)
```

```
# Print the model summary
summary(mlfit)
```

```
## Call:
## multinom(formula = vc ~ pid + agecats + gender, data = cces16)
##
## Coefficients:
##        (Intercept)      pid agecats36 to 50 agecatsOver 50     gender
## Romney   -6.432394 2.776238       0.5026041      1.0392971 -0.1359982
## Other    -4.064238 1.497458       0.1057551     -0.2319235 -0.7827882
##
## Std. Errors:
##        (Intercept)        pid agecats36 to 50 agecatsOver 50     gender
## Romney  0.08088099 0.02171768      0.05035074     0.04370033 0.03297672
## Other   0.11463718 0.03257481      0.07268663     0.06687796 0.05765850
##
## Residual Deviance: 37466.6
## AIC: 37486.6
```

Weighting

All examples in this section require the **survey** package:

```
library(survey)
```

```
## Loading required package: grid
```

```
## Loading required package: Matrix
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survey'
## The following object is masked from 'package:graphics':
##
##     dotchart
```

Weighted Tabulations

```
## WITHOUT WEIGHTING
# Tabulate PID
prop.table(table(cces16$pid_reverse))
```

```
##
##  Republican Independent    Democrat
##   0.3336641   0.1679444   0.4983915
```

```
## WITHOUT WEIGHTING
# Tabulate PID by age categories
pidxage <- table(cces16$pid_reverse, cces16$agecats)
pidxage
```

```
##
##               35 and Under 36 to 50 Over 50
##   Republican          4799     4470   11578
##   Independent         3137     2660    4696
##   Democrat           10427     7059   13653
```

```
prop.table(pidxage,2)
```

```
##
##               35 and Under  36 to 50   Over 50
##   Republican     0.2613407 0.3150328 0.3868747
##   Independent    0.1708327 0.1874692 0.1569152
##   Democrat       0.5678266 0.4974981 0.4562101
```

```
## WITH WEIGHTING
# Create a survey design dataframe
svy.cces16 <- svydesign(ids = ~1,
                        data = cces16,
                        weights = cces16$commonweight_vv)
# Weighted tabulation of PID
prop.table(svytable(~cces16$pid_reverse, design = svy.cces16))
```

```
## cces16$pid_reverse
##  Republican Independent    Democrat
##   0.3732446   0.1539131   0.4728423
```

```
## WITH WEIGHTING
# Weighted tabulation of PID by age categories
prop.table(svytable(~cces16$pid_reverse+cces16$agecats,
                    design = svy.cces16),2)
```

```
##                    cces16$agecats
## cces16$pid_reverse 35 and Under  36 to 50   Over 50
##        Republican     0.3103759 0.3544857 0.4244888
##        Independent    0.1614909 0.1740768 0.1395962
```

```
##        Democrat       0.5281332 0.4714375 0.4359151
```

Weighted Models

```
## WITHOUT WEIGHTING
# Run a logit regression for attitudes toward gun control
fit <- glm(ban_ar ~ pid_reverse + agecats + gender,
           data=cces16, family = binomial())
summary(fit)
```

```
##
## Call:
## glm(formula = ban_ar ~ pid_reverse + agecats + gender, family = binomial(),
##     data = cces16)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2748  -0.9804   0.4639   0.7933   1.6830
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -1.99725    0.03790  -52.69   <2e-16 ***
## pid_reverseIndependent   0.62226    0.02510   24.79   <2e-16 ***
## pid_reverseDemocrat      2.13318    0.02241   95.20   <2e-16 ***
## agecats36 to 50          0.32115    0.02651   12.11   <2e-16 ***
## agecatsOver 50           0.65539    0.02286   28.67   <2e-16 ***
## gender                   0.85891    0.01927   44.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 79067  on 62049  degrees of freedom
## Residual deviance: 65697  on 62044  degrees of freedom
##   (2550 observations deleted due to missingness)
## AIC: 65709
##
## Number of Fisher Scoring iterations: 4
```

```
## WITH WEIGHTING
wfit <- svyglm(ban_ar ~ pid_reverse + agecats + gender,
               design=svy.cces16, family = binomial())
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
summary(wfit)
```

```
##
## Call:
## svyglm(formula = ban_ar ~ pid_reverse + agecats + gender, design = svy.cces16,
##     family = binomial())
##
## Survey design:
## svydesign(ids = ~1, data = cces16, weights = cces16$commonweight_vv)
```

```
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -2.10613    0.05543 -37.999   <2e-16 ***
## pid_reverseIndependent   0.59609    0.03475  17.154   <2e-16 ***
## pid_reverseDemocrat      2.09675    0.03212  65.285   <2e-16 ***
## agecats36 to 50          0.30622    0.03701   8.274   <2e-16 ***
## agecatsOver 50           0.66999    0.03259  20.557   <2e-16 ***
## gender                   0.89915    0.02683  33.509   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 0.9962514)
##
## Number of Fisher Scoring iterations: 4
```

Creating Post-Stratification Weights

To start, we need to create dummies for the variables to use in the weighting process.

```
cces16$female <- NA
cces16$female[cces16$gender==2] = 1
cces16$female[cces16$gender!=2] = 0

cces16$white <- NA
cces16$white[cces16$race==1] = 1
cces16$white[cces16$race!=1] = 0

cces16$black <- NA
cces16$black[cces16$race==2] = 1
cces16$black[cces16$race!=2] = 0

cces16$hispanic <- NA
cces16$hispanic[cces16$race==3] = 1
cces16$hispanic[cces16$race!=3] = 0

cces16$age18_29 <- NA
cces16$age18_29[cces16$age >= 18 & cces16$age <= 29] = 1
cces16$age18_29[cces16$age < 18 | cces16$age > 29] = 0

cces16$age65_over <- NA
cces16$age65_over[cces16$age >= 65] = 1
cces16$age65_over[cces16$age < 65] = 0
```

Next, we'll initialize an unweighted survey design object that will be raked over.

```
svy_unweighted <- svydesign(ids = ~1, data = cces16)
```

```
## Warning in svydesign.default(ids = ~1, data = cces16): No weights or
## probabilities supplied, assuming equal probability
```

Now we define our target marginal distributions of variables for weighting. The margins are defined as a numeric vector of length 2 representing the proportion of zeroes to ones in the population. Note that these margins are arbitrary proportions from another tutorial dataset; you will want to use census data or other population-level data sources to define the margins.

```
female_margins <- data.frame(female = c("0", "1"),
                             Freq = nrow(cces16) * c(0.47, 0.53))
white_margins <- data.frame(white = c("0", "1"),
                            Freq = nrow(cces16) * c(0.12, 0.88))
black_margins <- data.frame(black = c("0", "1"),
                            Freq = nrow(cces16) * c(0.96, 0.04))
hispanic_margins <- data.frame(hispanic = c("0", "1"),
                               Freq = nrow(cces16) * c(0.95, 0.05))
age18_margins <- data.frame(age18_29 = c("0", "1"),
                            Freq = nrow(cces16) * c(0.93, 0.07))
age65_margins <- data.frame(age65_over = c("0", "1"),
                            Freq = nrow(cces16) * c(0.70, 0.30))
```

Now we can rake a set of post-stratification weights by passing the sample margins as a list of model formulae, and the population margins as a list of dataframes.

```
svy_weighted <- rake(design = svy_unweighted,
                     sample.margins = list(~female, ~white, ~black, ~hispanic, ~age18_29, ~age65_over),
                     population.margins = list(female_margins, white_margins, black_margins,
                                               hispanic_margins, age18_margins, age65_margins),
                     control = list(maxit = 25))
```

To briefly examine the weights we estimated:

```
summary(weights(svy_weighted))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1514  0.5247  1.1802  1.0000  1.1871  1.8653
```

If we estimate weights that are too extreme, we can trim them as follows:

```
svy_weighted <- trimWeights(svy_weighted, lower=.4, upper=8, strict=TRUE)
```

Item Scaling

To be completed: alpha index, factor analysis, IRT models

Matching & Balancing

To be completed: propensity score matching, coarsened exact matching, entropy balancing