Analyzing YRBS Data Using R's Survey Package: A Comprehensive Guide

Introduction

This guide will walk you through analyzing Youth Risk Behavior Survey (YRBS) data using R, with a particular focus on the `survey` package. If you're transitioning from SAS to R or are new to working with complex survey data, this guide will help you understand both the "how" and "why" of each analytical step.

Prerequisites

Before beginning your analysis, you'll need to install and load several key R packages:

Why These Packages?

- `tidyverse`: Provides essential data manipulation tools (similar to PROC SQL in SAS)
- `vroom`: Enables fast loading of large CSV files
- `survey`: Handles complex survey designs (analogous to PROC SURVEYFREQ in SAS)
- `here`: Manages file paths consistently across different computers
- `data.table`: Offers efficient data manipulation for large datasets
- `questionr`: Allows straightforward odds ratio analysis

Data Loading and Initial Processing

Loading YRBS Data

```r
YRBSdata <- vroom("yrbs2015_2021pooled_csv.csv")
```

```
## Rows: 816894 Columns: 308
## -- Column specification -----------------------------------------------------------------
## Delimiter: ","
## chr (208): location, age, sex, raceeth, grade, helmbike, seatbelt, dripass, weap30dy, weapschl, skip
## dbl  (29): year, height, weigh, bmipct, weight, psu, stratum, bmi, survyear, record, concentrating,
## lgl  (71): gun, fighurt, parviol, ma07ipv, ma05ipv, smodaily, ma5quit, chewschl, macontcp, exerwght,
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Variable Selection

Instead of loading the entire dataset, we select only the variables needed for analysis:

```r
YRBSdata <- YRBSdata %>%
  dplyr::select(year, location, stratum, REGION, psu, weight,
                ADJWEIGHT15_21, sexid_v2, genderid, genderexp,
                year, age, race7, sex, grade, wghtperc, exerwght,
                diet, madiet, fast, padiet, dietpill, purge,
                nofood, bmi, REGION)
```

Why Select Variables?

1. Improves computational efficiency
2. Reduces memory usage
3. Makes the dataset more manageable
4. Helps focus the analysis on relevant variables

Data Preparation

Converting Categorical Variables to Factors

```
YRBSdata <- YRBSdata %>%
  mutate_at(c('genderexp', 'genderid', 'race7', 'REGION',
              'sexid_v2', 'age', 'sex', 'grade'), as.factor)
```

Why Convert to Factors?

- Ensures R treats these variables as categorical rather than continuous
- Enables proper statistical analysis
- Prevents errors in survey calculations
- Similar to FORMAT statements in SAS

Remove Weight Blanks

The survey design will not properly run if we have blank survey weights. This is because the assigned weight variable is used to account for population density across jurisdictions.

```
YRBSdata <- YRBSdata %>%
  filter(complete.cases(YRBSdata$ADJWEIGHT15_21))
```

Creating Separate Datasets for Different Analyses

A critical step in YRBS analysis is creating two separate datasets:

```
YRBSdata_tables <- YRBSdata
```

Why Create Two Datasets?

1. **YRBSdata (for Logistic Regression)**
   - Contains binary (0/1) coded variables needed for logistic regression
   - Used for calculating odds ratios
   - Variables like `genderexp` are coded as 1/0 for statistical modeling
   - Similar to creating dummy variables in SAS
2. **YRBSdata_tables (for Frequency Tables)**
   - Maintains original categorical labels
   - Used for creating readable frequency tables and cross-tabulations
   - Keeps text values (e.g., "SOMEWHAT MASC" instead of 1)

- Makes output tables more interpretable

This separation allows us to:

- Maintain data integrity for different types of analyses
- Avoid repeatedly recoding variables
- Generate both statistical results and human-readable tables
- Prevent confusion between coded and uncoded variables

Preparing Binary Variables (YRBSdata only)

For logistic regression, convert categorical variables to binary (0/1):

```
# Example transformations
YRBSdata$genderexp <- ifelse(YRBSdata$genderexp %in%
                             c("SOMEWHAT MASC", "MOSTLY MASC", "VERY MASC"), 1, 0)
YRBSdata$purge <- ifelse(YRBSdata$purge == "YES", 1, 0)
```

Survey Design Implementation

Creating Separate Survey Design Objects

Because we have two differently formatted datasets, we need two survey design objects:

```
# For regression analysis
yrbsdes <- svydesign(id = ~psu,
                     weights = ~ADJWEIGHT15_21,
                     strata = ~interaction(year, location, stratum, drop = TRUE),
                     data = YRBSdata,
                     nest = TRUE)

# For frequency tables
yrbsdes_tables <- svydesign(id = ~psu,
                            weights = ~ADJWEIGHT15_21,
                            strata = ~interaction(year, location, stratum, drop = TRUE),
                            data = YRBSdata_tables,
                            nest = TRUE)

# Handle single-PSU strata
options(survey.lonely.psu="adjust")
```

Why Create Two Survey Designs?

1. **yrbsdes (for Regression)**
   - Uses the binary-coded dataset
   - Optimized for statistical modeling
   - Produces odds ratios and other statistical measures
2. **yrbsdes_tables (for Tables)**

- Uses the categorical-labeled dataset
- Creates readable frequency tables
- Maintains original variable labels for clear reporting

Important Notes:

1. Complete all data preparation BEFORE creating survey designs
2. The survey design objects "lock in" your data configuration
3. Set `options(survey.lonely.psu="adjust")` to handle single-PSU strata

Analysis Techniques

Creating Weighted Frequency Tables

Using `yrbsdes_tables` for human-readable output:

```
# Example: Demographics by gender expression
prop.table(svytable(~genderexp + race7, yrbsdes_tables), margin=1) %>%
  multiply_by(100) %>%
  round(digits=1)
```

```
##                        race7
## genderexp               American Indian/Alaskan Native Asian Black or African American Hispanic/Latin
##   EQUALLY FEM AND MASC                             1.1   9.8                      15.4            35.
##   MOSTLY FEM                                       0.6   7.5                      13.6            28.
##   MOSTLY MASC                                      0.6   9.0                       9.9            27.
##   SOMEWHAT FEM                                     0.9  11.0                      13.6            33.
##   SOMEWHAT MASC                                    1.2  10.4                      11.5            28.
##   VERY FEM                                         0.7   5.0                      17.1            35.
##   VERY MASC                                        0.9   6.2                      21.6            28.
##                        race7
## genderexp               Multiple Races (Non-His) Native Hawaiian/Other PI White
##   EQUALLY FEM AND MASC                       3.5                      0.9  34.0
##   MOSTLY FEM                                 4.0                      0.4  45.7
##   MOSTLY MASC                                3.8                      0.5  48.6
##   SOMEWHAT FEM                               4.6                      0.5  36.0
##   SOMEWHAT MASC                              3.7                      0.5  44.1
##   VERY FEM                                   2.7                      0.6  38.0
##   VERY MASC                                  3.1                      0.5  38.9
```

Running Logistic Regression

Using `yrbsdes` for regression models:

```
# Example: Analyzing purging behavior
logit_model <- svyglm(purge ~ genderexp,
                      family = 'binomial',
                      design = yrbsdes,
                      na.action = na.omit,
                      rescale = TRUE)
```

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

```r
# Get odds ratios
odds.ratio(logit_model, level = 0.95)
```

```
##                   OR    2.5 % 97.5 %       p
## (Intercept) 0.057658 0.048749 0.0682 < 2e-16 ***
## genderexp   0.545402 0.353371 0.8418 0.00644 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Best Practices and Tips

1. **Data Management**
   - Always check for missing values before analysis
   - Document variable recoding decisions
   - Keep original and analysis datasets separate
2. **Survey Design**
   - Create separate design objects for different analyses
   - Always specify the correct weight variable
   - Handle lonely PSUs by entering `options(survey.lonely.psu="adjust")` into the console
3. **Analysis**
   - Use weighted analyses for all estimates
   - Check assumptions before running models
   - Document all analytical decisions

Common Troubleshooting

1. **Missing Values**
   - Use `na.omit()` or `filter(!is.na())` before analysis
   - Check codebook for expected missing value codes
2. **Survey Design Errors**
   - Ensure all strata have multiple PSUs
   - Verify weight variable is numeric and non-missing
   - Check for proper factor levels in categorical variables
3. **Results Interpretation**
   - Always use weighted percentages for population estimates
   - Report confidence intervals with point estimates
   - Document any subpopulation analyses

Additional Resources

- R Survey Package Documentation
- CDC YRBS Data User's Guide