Chapter 9
Using EdSurvey to Analyse PIAAC Data

Paul Bailey, Michael Lee, Trang Nguyen, and Ting Zhang

Abstract This chapter describes the use of the R package EdSurvey and its use in analysing PIAAC data. The package allows users to download public use PIAAC data, explore the codebooks, explore data, read in and edit relevant variables, and run analyses such as regression, logistic regression, and gap analysis.

9.1 Introduction

The EdSurvey package is a collection of functions for use in the R programming language R Core Team (2019) to help users easily work with data from the National Center for Education Statistics (NCES) and international large-scale assessments. Developed by the American Institutes for Research and commissioned by the NCES, this package manages the entire process of analyses of Programme for the International Assessment of Adult Competencies (PIAAC) data: downloading, searching the codebook and other metadata, conducting exploratory data analysis, cleaning and manipulating the data, extracting variables of interest, and finally data

This publication was prepared for NCES under Contract No. ED-IES-12-D-0002 with the American Institutes for Research. Mention of trade names, commercial products, or organisations does not imply endorsement by the US government.

P. Bailey (✉) · M. Lee · T. Zhang
American Institutes for Research, Washington, DC, USA
e-mail: pbailey@air.org; mlee@air.org; tzhang@air.org

T. Nguyen
Tamr Inc., Cambridge, MA, USA

D. B. Maehler, B. Rammstedt (eds.), *Large-Scale Cognitive Assessment*, Methodology of Educational Measurement and Assessment,
https://doi.org/10.1007/978-3-030-47515-4_9

analysis. This chapter describes the use of EdSurvey for each activity, with a focus on PIAAC data.[1,2]

Because of the scope and complexity of data from large-scale assessment programmes, such as PIAAC, the analysis of their data requires proper statistical methods—namely, the use of weights and plausible values. The EdSurvey package gives users intuitive one-line functions to perform analyses that account for these methods.

Given the size of large-scale data and the constraint of limited computer memory, the EdSurvey package is designed to minimise memory usage. Users with computers that have insufficient memory to read in entire datasets—the OECD Cycle 1 data are over a gigabyte once read in to R—can still perform analyses without having to write special code to limit the dataset. This is all addressed directly in the EdSurvey package—behind the scenes and without any additional intervention by the user—allowing researchers to more efficiently explore and analyse variables of interest.

The results of analyses on this saved data connection can then be stored or further manipulated. Alternatively, the getData function reads in selected variables of interest to generate an R data.frame. Individuals familiar with R programming might prefer to clean and explore their data using supplementary packages, which EdSurvey supports. These data.frames can then be used with all EdSurvey analytical functions.

The next section shows how to load EdSurvey and download and read in PIAAC data. The third section describes how you can see survey attributes in EdSurvey. The fourth deals with exploring PIAAC data. The fifth section describes data manipulation. The sixth section describes data analysis. The final section explains how to stay current with new developments in EdSurvey.

9.2 Getting Started

R is an open-source software and can be downloaded free of charge from www.r-project.org/ R Core Team (2019). The Comprehensive R Archive Network (CRAN) stores extensions to the base R functionality and can be used to install EdSurvey using the command

[1] EdSurvey 2.4 also can work with public and/or restricted use datasets from ECLS:K, ICCS, ICILS, NAEP, PIRLS, ePIRLS, PISA, TALIS, TIMSS, and TIMSS advanced; more datasets are added with each release.

[2] EdSurvey uses a variety of other packages; for a complete list, see https://CRAN.R-project.org/package=EdSurvey.

```
> install.packages('EdSurvey')
```

Having downloaded the `EdSurvey` package from CRAN, it must be loaded in every session with the command

```
> library('EdSurvey')
```

Then the user can download the OECD 2012 files with

```
> downloadPIAAC('~/')
```

When `downloadPIAAC` is run, the data are stored in a folder in the directory that the user specifies, here an operating system-defined folder called `'~/'`. On all machines this is the user's home folder. After the download is complete, users can manually change the folder structure. This chapter will assume that the download call used the folder `'~/'`, and the data were not subsequently moved from that folder. Within the target folder, the user specified (here `'~/'`) the data will be stored in a subfolder named '`PIAAC`'. All data for participating countries in Cycle 1 will be stored in the subdirectory '`PIAAC/Cycle 1`'. At the time of writing, only Cycle 1 is available for download.

One also can manually download desirable PIAAC data from the Organisation for Economic Co-operation and Development (OECD) webpage[3], including the 2012/2014 data, or acquire a data licence and access the restricted-use data files. When downloading manually, note that the PIAAC read-in function, `readPIAAC`, requires both the `.csv` files with the data and a codebook spreadsheet (`.xlsx` file) to be in the same folder.

The next step in running analysis is reading in the data. For PIAAC data, this is accomplished with the `readPIAAC` function, which creates an `edsurvey.data.frame` that stores information about the specific data files processed. This includes the location on disk, the file format and layout of those files, and the metadata that will allow `EdSurvey` to analyse the data. A PIAAC `edsurvey.data.frame` includes information for all variables at the individual level and any household-level variables.

Upon the first read-in, the `EdSurvey` package caches existing data as a flat text file; for all future sessions, this flat file stores the variables needed for any analysis. The PIAAC Cycle 1 data can be read-in by pointing to the pathway in the PIAAC Cycle 1 data folder and defining the country of interest. By setting `countries = c('ITA')` in a call to `readPIAAC`, an `edsurvey.data.frame` containing Cycle 1 data for Italy is created as the object `ita`:

[3]https://www.oecd.org/skills/piaac/data/

```
> ita <- readPIAAC('~/PIAAC/Cycle 1/', countries='ITA')

Found cached data for country code "ita".
```

The function uses the three-digit International Organization for Standardization country code to select countries to import (here, '`ITA`)'. Section 9.6.3 describes how to read in and analyse data from multiple countries at once. For now, other countries can be read in and analysed separately by repeating the above command with the code of another country, such as the Netherlands:

```
> nld <- readPIAAC('~/PIAAC/Cycle 1/', countries='NLD')

Found cached data for country code "nld".
```

9.3 Survey Design Attributes

When analysing data with `EdSurvey`, the package automatically accounts for the plausible values of scores as well as the sample survey design when conducting data analyses by storing metadata in the `edsurvey.data.frame`. There are four important survey design attributes that have a great influence on the output of later analysis: plausible values, weights, omitted levels, and achievement levels. This section describes these metadata elements and how users can display them.

PIAAC Cycle 1 data have ten plausible values for each domain (numeracy, literacy, and problem solving), as shown in the output of `showPlausibleValues` function. The `showPlausibleValues` function not only tells users about the PIAAC domain of skills this round of survey questionnaires contains but also shows the plausible value domain names representing their corresponding domain/subject scale as used in `EdSurvey` analytical functions.

```
> showPlausibleValues(ita, verbose=TRUE)

There are 3 subject scale(s) or subscale(s) in this
  edsurvey.data.frame:
'lit' subject scale or subscale with 10 plausible values
 (the default).
  The plausible value variables are: 'pvlit1', 'pvlit2',
  'pvlit3', 'pvlit4', 'pvlit5', 'pvlit6', 'pvlit7',
  'pvlit8', 'pvlit9', and 'pvlit10'

'num' subject scale or subscale with 10 plausible values.
```

(continued)

```
  The plausible value variables are: 'pvnum1', 'pvnum2',
  'pvnum3', 'pvnum4', 'pvnum5', 'pvnum6', 'pvnum7',
  'pvnum8', 'pvnum9', and 'pvnum10'

'psl' subject scale or subscale with 10 plausible values.
  The plausible value variables are: 'pvpsl1', 'pvpsl2',
  'pvpsl3', 'pvpsl4', 'pvpsl5', 'pvpsl6', 'pvpsl7',
  'pvpsl8', 'pvpsl9', and 'pvpsl10'
```

For example, the ten variables named `pvlit1` to `pvlit10` store an individual set of plausible values for the literacy scale score domain. These ten variables can simply be referred to by the name `lit`, and `EdSurvey` functions will correctly account for the plausible values in both estimation and variance estimation.

The PIAAC sample is a probability sample that was a single stage sample in some countries but a multistage sample in other countries Mohadjer et al. (2016). In addition, because of oversampling and nonresponse, the weights are informative. Users can print the available weights with the `showWeights` function

```
> showWeights(ita)

There is 1 full sample weight in this edsurvey.data.
  frame:
 'spfwt0' with 80 JK replicate weights (the default).
```

Similar to other PIAAC Cycle 1 countries, only one full sample weight (`spfwt0`) is available for Italy data, and the `showWeights` function displays it along with 80 replicate weights associated with it. Because it is the default and exclusive full sample weight, it is not necessary to specify the weight in `EdSurvey` analytical functions; `spfwt0` will be used by default. In addition, the jackknife replicates associated with `spfwt0` will be used by the variance estimation procedures without the user having to further specify anything.

By default, `EdSurvey` will show results from the analyses after listwise deletion of respondents with any special values, which are referred as 'omitted levels' in `EdSurvey`. For any data, the omitted levels can be seen with the `omittedLevels` command

```
> getAttributes(ita,'omittedLevels')
```

(continued)

```
 [1] "(Missing)"                    "DON'T KNOW"
 [3] "NOT STATED OR INFERRED"       "VALID SKIP"
 [5] "REFUSED"                      "DON'T KNOW/REFUSED"
 [7] "NO RESPONSE"                  "NOT REACHED/NOT
                                      ATTEMPTED"
 [9] "ALL ZERO RESPONSE"            NA
```

Users wishing to include these levels in their analysis can do so, usually, by recoding them or setting `omittedLevels=TRUE`. More information is available in the help documentation for each respective function.

To see all this information at once, the user can simply 'show' the data by typing the name of the `edsurvey.data.frame` object (i.e. `ita`) in the console

```
> ita

edsurvey.data.frame for Round 1 PIAAC (Numeracy,
Literacy, and Problem Solving) in Italy
Dimensions: 4621 rows and 1328 columns.

There is 1 full sample weight in this edsurvey.data.
frame:
  'spfwt0' with 80 JK replicate weights (the default).


There are 3 subject scale(s) or subscale(s) in this
  edsurvey.data.frame:
'lit' subject scale or subscale with 10 plausible values
 (the default).

'num'subject scale or subscale with 10 plausible values.

'psl'subject scale or subscale with 10 plausible values.


Omitted Levels:'(Missing)','DON'T KNOW','NOT STATED OR
               INFERRED','VALID SKIP','REFUSED','DON'T
               KNOW/REFUSED','NO RESPONSE','NOT REACHED/
               NOT ATTEMPTED','ALL ZERO RESPONSE', and
               'NA'
Achievement Levels:
Numeracy:
Proficiency Level 1: 176.00
Proficiency Level 2: 226.00
```

(continued)

```
Proficiency Level 3: 276.00
Proficiency Level 4: 326.00
Proficiency Level 5: 376.00
Achievement Levels:
Literacy:
Proficiency Level 1: 176.00
Proficiency Level 2: 226.00
Proficiency Level 3: 276.00
Proficiency Level 4: 326.00
Proficiency Level 5: 376.00
Achievement Levels:
Problem Solving:
Proficiency Level 1: 241.00
Proficiency Level 2: 291.00
Proficiency Level 3: 341.00
```

9.4 Exploring PIAAC Data

Once the desired data have been read in, `EdSurvey` provides data exploration functions that users can use in combination with PIAAC codebooks and technical documents in preparation for analysis.

It is worth mentioning that many of the basic functions that work on a `data.frame`, such as `dim`, `nrow`, `ncol`, and `$`, also work on an `edsurvey.data.frame` and can be used for exploration. Editing data is not similar to a `data.frame` and is covered in Sect. 9.5.2.

To view the codebook, the user can use the `showCodebook` function. The output will be long, given the number of columns in the PIAAC data; use the function `View` to display it in spreadsheet format

```
> View(showCodebook(ita))
```

Even with spreadsheet formatting, the codebook can be somewhat daunting to browse. The `searchSDF` function allows the user to search the codebook variable names and labels

```
> searchSDF('income', data=ita)

  variableName
1     d_q18a_t
```

(continued)

```
2 monthlyincpr
3  yearlyincpr
                                                      Labels
1 ANNUAL NET INCOME BEFORE TAXES AND DEDUCTIONS
   (TREND-IALS/ALL)
2              MONTHLY INCOME PERCENTILE RANK CATEGORY
                 (DERIVED)
3               YEARLY INCOME PERCENTILE RANK CATEGORY
                 (DERIVED)
```

Notice that the search is not case sensitive and uses regular expressions. The search can be refined by adding additional terms in a vector, using the `c` function; this refines the search to just those rows where all the strings named are present. This search refines the previous results to a single variable

```
> searchSDF(c('income','annual'), data=ita)

  variableName
1     d_q18a_t
                                                      Labels
1 ANNUAL NET INCOME BEFORE TAXES AND DEDUCTIONS
   (TREND-IALS/ALL)
```

Sometimes knowing the variable name and label is insufficient, and knowing the levels helps. Users can show these levels by setting the `levels` argument to `TRUE`

```
> searchSDF(c('income','annual'), data=ita, levels=TRUE)

Variable: d_q18a_t
Label: ANNUAL NET INCOME BEFORE TAXES AND DEDUCTIONS
        (TREND-IALS/ALL)
Levels (Lowest level first):
     0. NO INCOME
     1. LOWEST QUINTILE
     2. NEXT LOWEST QUINTILE
     3. MID-LEVEL QUINTILE
     4. NEXT TO HIGHEST QUINTILE
     5. HIGHEST QUINTILE
     6. VALID SKIP
     7. DON'T KNOW
     8. REFUSED
     9. NOT STATED OR INFERRED
```

To get an initial insight into a variable's response frequencies, population estimated response frequencies, and response percentages, use the `summary2` function. The function prints out weighted summary statistics using the default weight variable, which is automatically picked up in `readPIAAC` function. The summary statistics for the variable `'d_q18a_t'` are shown in Table 9.1

```
> summary2(ita, 'd_q18a_t')
```

Note that `EdSurvey` will show variables that OECD includes in the data, some of which will be entirely missing; `summary2` will show this. An example of this is the `d_q18a_t` variable in Canada.

Similarly, `summary2` can show summary statistics for continuous variables. The following example code shows the summary statistics for the set of plausible values for the literature domain (`'lit'`), as shown in Table 9.2

Table 9.1 Results from `summary2(ita, 'd_q18a_t')`

d_q18a_t	N	Weighted N	Weighted percent	Weighted percent SE
(Missing)	2350	21896886.00	55.62	0.82
NO INCOME	43	345319.76	0.88	0.14
LOWEST QUINTILE	418	3428919.30	8.71	0.47
NEXT LOWEST QUINTILE	415	3414626.97	8.67	0.51
MID-LEVEL QUINTILE	423	3457583.24	8.78	0.48
NEXT TO HIGHEST QUINTILE	468	3378711.90	8.58	0.47
HIGHEST QUINTILE	504	3447782.84	8.76	0.39

Note. Estimates are weighted using weight variable `spfwt0`

Table 9.2 Results from `summary2(ita, 'lit')`

d_q18a_t	N	Weighted N	Weighted percent	Weighted percent SE
(Missing)	2350	21896886.00	55.62	0.82
NO INCOME	43	345319.76	0.88	0.14
LOWEST QUINTILE	418	3428919.30	8.71	0.47
NEXT LOWEST QUINTILE	415	3414626.97	8.67	0.51
MID-LEVEL QUINTILE	423	3457583.24	8.78	0.48
NEXT TO HIGHEST QUINTILE	468	3378711.90	8.58	0.47
HIGHEST QUINTILE	504	3447782.84	8.76	0.39

Note. Estimates are weighted using weight variable `spfwt0`

```
> summary2(ita, 'lit')
```

Another powerful exploratory function in the package is `edsurveyTable`. This function allows users to run weighted cross-tab analyses for any number of categorical variables along with or without an outcome (or continuous) variable.

The following example shows how to create a cross-tab table of employment status (`c_d05`) by age groups in 10-year intervals (`ageg10lfs`) on literacy outcome

```
> edsurveyTable(lit ~ ageg10lfs, data = ita)

Formula: lit ~ ageg10lfs

Plausible values: 10
jrrIMax: 1
Weight variable: 'spfwt0'
Variance method: jackknife
JK replicates: 80
full data n: 4621
n used: 4589


Summary Table:
  ageg10lfs    N   WTD_N      PCT   SE(PCT)     MEAN SE(MEAN)
 24 OR LESS  524 5649536 14.44420 0.1710222 260.8013 2.689490
      25-34  784 7359208 18.81533 0.3123164 260.2447 2.334559
      35-44 1229 9524266 24.35075 0.3821840 252.7739 1.817189
      45-54 1021 8554035 21.87015 0.3640822 248.7787 1.817378
    55 PLUS 1031 8025778 20.51956 0.2523894 233.3650 2.260212
```

Similar to `summary2`, the `edsurveyTable` function returns the weighted percentage (`PCT`) and conditional means (`MEAN`) of a selected outcome variable—in this case the literacy score.

The results also can be broken down by multiple variables by using a plus (+) between variables. For example, we add `c_d05`, the current employment status, in the equation.

```
> edsurveyTable(lit ~ ageg10lfs + c_d05, data = ita)
    # output not shown
```

Finally, the correlation function can help users explore associations between variables. The function `cor.sdf` allows for Pearson (for bivariate normal variables), Spearman (for two continuous variables), polyserial (for one continuous and one discrete variable), and polychoric (for two discrete variables) correlations.[4]

```
> cor.sdf('lit','d_q18a_t',data=ita,method='polyserial')

Method: polyserial
full data n: 4621
n used: 2271

Correlation: 0.1973387

Correlation Levels:
  Levels for Variable 'd_q18a_t' (Lowest level first):
    1. NO INCOME
    2. LOWEST QUINTILE
    3. NEXT LOWEST QUINTILE
    4. MID-LEVEL QUINTILE
    5. NEXT TO HIGHEST QUINTILE
    6. HIGHEST QUINTILE
```

These results show a polyserial correlation between literacy and income quintile as .20 (after rounding), with weight `spfwt0` applied by default. Because a correlation analysis assumes that the discrete outcome is ordered, the levels of the discrete variable `d_q18a_t` are shown to allow users to check that it moves in one direction; here, increasing from 1 to 6.

9.5 Accessing and Manipulating PIAAC Data

Typically, before performing an analysis, users edit data consistent with their research goals. This can happen in one of two ways in the `EdSurvey` package:

1. Clean and analyse data within the `EdSurvey` package functions,
2. Use `getData` to extract a `data.frame` to clean and edit with any `R` tool, and then use `rebindAttributes` to use `EdSurvey` functions to analyse the data.

This section describes these two ways of preparing data for an analysis for use in the `EdSurvey` package (see fig. 9.1 for an overview).

[4]For more details on the correlations and their computation, see `vignette('wCorrFormulas',package='wCorr')`.

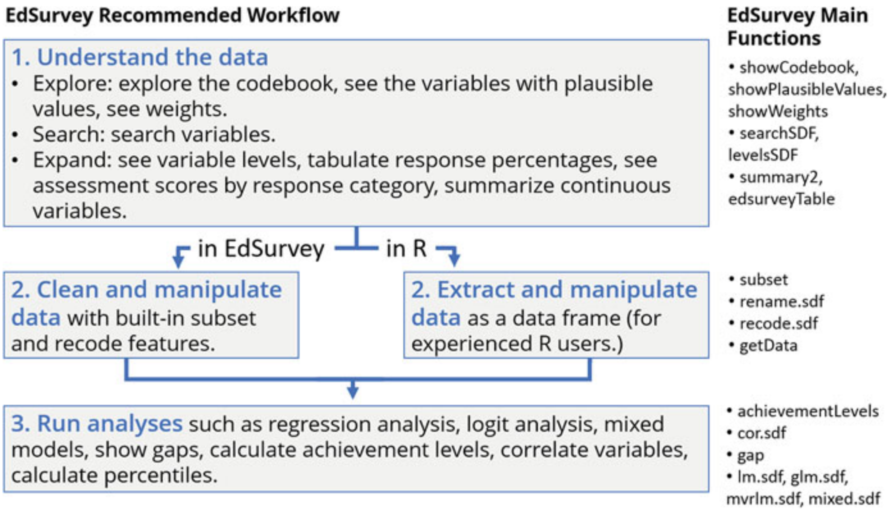

Fig. 9.1 EdSurvey workflow and functions

9.5.1 Cleaning Data in `EdSurvey`

`EdSurvey` provides three data manipulation functions: `subset`, `recode`, and `rename`.

The subset function limits the rows that are used in an analysis to those that meet a condition. For example, to return the summary statistics for the literacy variable, restricting the population of interest to Italian males, one could use `subset`. Note the level label (e.g. the ‘`MALE`’ in the following code) needs to be consistent with the label that is in the data, which can be revealed through a call such as `table(ita$gender_r)`.

```
> itaM <- subset(ita, gender_r %in% 'MALE')
> summary2(itaM, 'lit')

Estimates are weighted using weight variable 'spfwt0'
  Variable    N Weighted N     Min.  1st Qu.   Median     Mean
1      lit 2235   19679710 88.20746 219.5522 251.8223 250.3554
   3rd Qu.     Max.       SD NA's Zero-weights
1 283.9397 399.2344 46.42543   15            0
```

The `recode` function allows us to change the labels or condense on a discrete variable. For example, the user may want to generate conditional means of the employment status variable (`c_d05`), wherein those individuals who are (a) ‘`UNEMPLOYED`’ or (b) ‘`OUT OF THE LABOUR FORCE`’ are condensed to one level to compare to the subgroup of individuals employed. This leaves a level ('NOT KNOWN') that is then removed with subset

```
> itaRecode <- recode.sdf(ita, recode=
+                list(c_d05=
+                 list(from=c('OUT OF THE LABOUR FORCE',
+                              'UNEMPLOYED'),
+                       to=c('NOT EMPLOYED'))))
> itaRecode <- subset(itaRecode, !c_d05
                 %in% c('NOT KNOWN'))
> edsurveyTable(lit ~ c_d05, data=itaRecode)

Formula: lit ~ c_d05

Plausible values: 10
jrrIMax: 1
Weight variable: 'spfwt0'
Variance method: jackknife
JK replicates: 80
full data n: 4621
n used: 4587


Summary Table:
        c_d05    N    WTD_N       PCT    SE(PCT)      MEAN
     EMPLOYED 2869 21957948 56.19657 0.06896769 254.4060
 NOT EMPLOYED 1718 17115519 43.80343 0.06896769 245.5068

                                                  SE(MEAN)
                                                  1.468391
                                                  1.521626
```

Finally, rename allows the user to adjust a variable's name.

```
> itaRecode <- rename.sdf(itaRecode, oldnames='c_d05',
   newnames='emp')
> edsurveyTable(lit ~ emp, data=itaRecode)

Formula: lit ~ emp

Plausible values: 10
jrrIMax: 1
Weight variable: 'spfwt0'
Variance method: jackknife
JK replicates: 80
```

(continued)

```
full data n: 4621
n used: 4587


Summary Table:
          emp    N    WTD_N       PCT    SE(PCT)      MEAN
     EMPLOYED 2869 21957948 56.19657 0.06896769 254.4060
 NOT EMPLOYED 1718 17115519 43.80343 0.06896769 245.5068

                                                   SE(MEAN)
                                                   1.468391
                                                   1.521626
```

9.5.2 Using getData

Users may want to perform extensive recoding of variables but have preferred methods of recoding using specific R packages. The `getData` function allows users to select variables to read into memory, extract, and then edit freely. The `rebindAttributes` function allows the final `data.frame` to be used with `EdSurvey` analysis functions.

```
> itaRaw <- getData(data=ita,
+                   varnames=c('lit', 'spfwt0',
                               'gender_r', 'c_d05'))
```

In this example, `getData` extracts the following:

- two variables: `gender_r` and `c_d05`
- ten plausible values associated with `lit`
- the weight for this data frame: `spfwt0`

Some important things to note:

1. `addAttributes` is set to the default value of `FALSE`. Setting `addAttributes = TRUE` is one method in which the resultant data object (`itaRaw`) can be passed to other `EdSurvey` package functions.
2. All the jackknife replicate weights are returned automatically (`spfwt1` to `spfwt80`).
3. `omittedLevels` is set to `TRUE`, the default, so that variables with special values (such as multiple entries or NAs) are removed by `getData`. This setting removes these values from factors that are not typically included in regression

analysis and cross-tabulation. Alternatively, this can be set to `FALSE` to be manipulated by the user.

The `itaRaw` data object is a class `data.frame`, which allows it to be manipulated with any supplementary R function. For instance, the `head` function shows us a preview of our data, focusing on Columns 1 through 15, revealing the requested variables and the first few rows of the resulting data

```
> head(x = itaRaw[,1:15])

  gender_r                      c_d05   pvlit1   pvlit2   pvlit3
1     MALE                   EMPLOYED 239.8982 258.2188 261.3314
2   FEMALE                   EMPLOYED 261.4386 246.9221 276.6944
3     MALE                   EMPLOYED 310.1177 328.5708 308.8707
4   FEMALE                   EMPLOYED 280.5043 255.7476 261.8692
5     MALE                   EMPLOYED 288.1527 307.2000 298.3016
6   FEMALE OUT OF THE LABOUR FORCE 223.8645 216.0648 243.9239
    pvlit4   pvlit5   pvlit6   pvlit7   pvlit8   pvlit9  pvlit10
1 271.8589 255.7649 243.9113 262.1387 249.3910 276.2055 244.6589
2 258.2071 246.7529 245.5175 257.0885 264.5383 254.7749 252.8056
3 311.5167 296.3410 306.3655 309.7482 308.1918 304.6406 307.8876
4 248.4239 270.5346 279.4498 294.2028 289.6540 259.8313 272.2326
5 338.3870 303.7172 297.3620 300.9883 300.2252 316.3354 328.8312
6 283.3290 167.0126 252.9510 228.5226 280.0687 207.0705 242.5360
     spfwt1    spfwt2    spfwt3
1  2076.916  2151.808  2139.313
2 11421.905 11409.298 11372.425
3 11125.408 11378.000 11020.750
4  2165.858  2177.041  2179.606
5  4415.642  4409.966  4398.984
6  8739.920  8692.451  8708.170
```

To replicate the data manipulation from Sect. 9.5.1, `gsub`, a base R function that uses pattern matching to replace values in a variable, recodes the values in the variable `c_d05`. The base function `subset` then removes the level '`NOT KNOWN`'.

```
> itaRaw$c_d05 <- gsub(pattern = 'OUT OF THE LABOUR
                       FORCE|UNEMPLOYED',
+                      replacement = 'not employed',
+                      x = itaRaw$c_d05)
> itaRaw <- subset(itaRaw, !c_d05 %in% 'NOT KNOWN')
```

The `rebindAttributes` function allows us to reassign survey attributes so that `EdSurvey` package functions are accessible. Simply call the manipulated data frame and the `edsurvey.data.frame` containing the requisite attributes

```
> itaRawRebinded <- rebindAttributes(itaRaw, ita)
```

Now we can apply EdSurvey functions, for example,

```
> edsurveyTable(lit ~ c_d05, data=itaRawRebinded)
Formula: lit ~ c_d05

Plausible values: 10
jrrIMax: 1
Weight variable: 'spfwt0'
Variance method: jackknife
JK replicates: 80
full data n: 4621
n used: 4587


Summary Table:
         c_d05    N    WTD_N       PCT    SE(PCT)      MEAN
      EMPLOYED 2869 21957948 56.19657 0.06896769 254.4060
  not employed 1718 17115519 43.80343 0.06896769 245.5068

                                                   SE(MEAN)
                                                   1.468391
                                                   1.521626
```

9.6 Data Analysis

9.6.1 Regression

Regression is a well-known and frequently used tool that EdSurvey provides in the lm.sdf function. Regression equations are typically written as

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i \tag{9.1}$$

where y_i is the outcome for individual i, α is an intercept, x_{ki} is the level of the kth explanatory (exogenous) variable, β_k is the kth regression coefficient, and ϵ_i is the regression residual for individual i.

As an example, the outcome is the literacy score (lit), which is described as a function of income quintile (d_q18a_t) and age (age_r). See results in Table 9.3.

Table 9.3 Results from `summary(lm1)`

	coef	se	t	dof	Pr(> |t|)
(Intercept)	282.65	11.09	25.50	34.86	0.00
d_q18a_tLOWEST QUINTILE	−17.23	10.11	−1.70	20.83	0.10
d_q18a_tNEXT LOWEST QUINTILE	−10.86	10.42	−1.04	28.10	0.31
d_q18a_tMID-LEVEL QUINTILE	1.46	9.79	0.15	24.35	0.88
d_q18a_tNEXT TO HIGHEST QUINTILE	6.16	10.16	0.61	26.19	0.55
d_q18a_tHIGHEST QUINTILE	13.47	9.73	1.38	25.00	0.18
age_r	−0.65	0.13	−5.13	71.39	0.00

```
> lm1 <- lm.sdf(lit ~ d_q18a_t + age_r, data=ita)
> summary(lm1)
```

In `R`, the formula for this regression equation is written as `y ~x1 + x2`. Note that there is no need to generate dummy codes for discrete variables like `d_q18a_t`.

The typical outcome contains a header similar to `edsurveyTable`, which is not shown for brevity. To explore the unprinted attributes, print `summary(lm1)` in the console.

`EdSurvey` calculates the regression coefficients by running one weighted regression per plausible value:

$$\hat{\beta}_k = \frac{1}{P}\sum_{p=1}^{P}\beta_k^{(p)} \tag{9.2}$$

where there are P plausible values, each indexed with a p, and the superscript (p) indicates the pth plausible value was used.

Variance estimation is complicated because of the presence of the plausible values and because many countries used a multistage, geography-based, sampling technique to form the PIAAC sample. Because of the geographic proximity between respondents, there is a correlation between respondents' scores within a sampled group, relative to two randomly selected individuals. The variance estimator `EdSurvey` uses accounts for both of these using the variance estimator

$$V = V_I + V_S \tag{9.3}$$

where V is the total variance of an estimator, V_I is the imputation variance—accounting for the plausible values—and V_S is the sampling variance, accounting for the covariance between geographically clustered individuals. V_I is estimated according to Rubin's rule (Rubin 1987)

$$V_I = \frac{M}{M+1} \sum_{p=1}^{P} \left(\beta_k^{(p)} - \beta_k \right) \tag{9.4}$$

where β_k is averaged across the plausible values (Eq. 9.2). Then the sampling variance frequently uses the jackknife variance estimator and can be estimated with each plausible value as

$$V_S^{(p)} = \sum_{j=1}^{J} \left(\beta_{kj}^{(p)} - \beta_k \right) \tag{9.5}$$

where $\beta_{kj}^{(p)}$ is the estimate of the regressor estimated with the jth replicate weights, with the pth plausible value. In `EdSurvey`, the `jrrIMax` argument sets the number of plausible values used; any number is valid, but lower numbers are faster.

$$V_S = \frac{1}{\text{jrrIMax}} \sum_{p=1}^{\text{jrrIMax}} V_S^{(p)} \tag{9.6}$$

As a convenience, `EdSurvey` sets values larger than the number of plausible values equal to the number of plausible values, so using `jrrIMax=Inf` uses all plausible values.

The `EdSurvey` package also can use a Taylor series variance estimator—available by adding the argument `varMethod='Taylor'` (Binder 1983). More details regarding variance estimation can be found in the EdSurvey Statistics vignette.

Although most of the model details are returned in the regression output, a few additional elements are available to inform interpretation of the results. First, there is a head block that describes the weight used (`spfwt0`), the variance method (`jackknife`), the number of jackknife replicates (80), the full data n-size (4,621), and the n-size for this regression (2,271). The latter n-size includes the extent of listwise deletion.

The coefficients block has many typically displayed statistics, including the degrees of freedom (`dof`) by coefficient. This is calculated using the Welch-Satterthwaite equation (Satterthwaite 1946). For the kth coefficient, the notation of (Wikipedia Contributors 2019), $k_i = 1$ and $s_i = \beta_{kj} - \beta_k$, indicates the difference between the estimated value for the jth jackknife replicate weight and the value estimated with the full sample weights (β_k). Because this statistic varies by coefficient, so do the degrees of freedom. `EdSurvey` applies the Rust and Johnson modification to the Welch-Satterthwaite equation that multiplies the Welch-Satterthwaite degrees of freedom by a factor of $3.16 - \frac{2.77}{J^{1/2}}$, where J is the number of jackknife replicates (Rust and Johnson 1992).

9.6.2 Binomial Regression

When a regression's dependent variable (outcome) is binary—consisting of 1s and 0s or true and false—the regression is a binomial regression. `EdSurvey` allows for two such regressions: logistic regression and probit regression. The corresponding functions for these methods are `logit.sdf` and `probit.sdf`. This section focuses on `logit.sdf`, but most components also apply to `probit.sdf`.

An example of a binomial regression is to look at the outcome of income percentile being in the mid-quintile or higher as described by mother's education (`j_q06b`) and own age (`age_r`). The user may first wish to inspect `j_q06b` (results in Table 9.4).[5]

```
> summary2(ita,'j_q06b')
```

When a regression is run, `EdSurvey` will exclude the values other than '`ISCED 1, 2, AND 3C SHORT`', '`ISCED 3 (EXCLUDING 3C SHORT) AND 4`', and '`ISCED 5 AND 6`'; the first of these levels will be the omitted group and treated as the reference.

For binomial regression, we recommend explicitly dichotomising the dependent variable in the `logit.sdf` call so that the desired level has the 'high state' associated with positive regressors—this is done with the `I(·)` function. Here, the function makes the dependent variable a 1 when the condition is `TRUE` and a 0 when the condition is `FALSE`; the results are shown in Table 9.5.

Table 9.4 Results from `summary2(ita,'j_q06b')`

j_q06b	N	Weighted N	Weighted percent	Weighted percent SE
(Missing)	2	16688.34	0.04	0.04
ISCED 1, 2, AND 3C SHORT	3639	31437133.66	79.85	0.66
ISCED 3 (EXCL 3C SHORT) AND 4	758	6057515.46	15.39	0.57
ISCED 5 AND 6	176	1471224.40	3.74	0.32
DON'T KNOW	10	107909.31	0.27	0.09
REFUSED	3	24560.83	0.06	0.03
NOT STATED OR INFERRED	33	254798.01	0.65	0.16

[5]In the tables the level 'ISCED 3 (EXCLUDING 3C SHORT) AND 4' is sometimes shortened to 'ISCED 3 (EXCL 3C SHORT) AND 4'.

Table 9.5 Results from `summary(logit1)`

	coef	se	t	dof	Pr(> |t|)
(Intercept)	−1.25	0.24	−5.20	73.08	0.00
j_q06bISCED 3 (EXCL 3C SHORT) AND 4	0.62	0.14	4.59	77.55	0.00
j_q06bISCED 5 AND 6	0.07	0.25	0.28	67.79	0.78
age_r	0.04	0.01	6.87	87.51	0.00

Table 9.6 Results from `oddsRatio(logit1)`

	OR	2.5%	97.5%
(Intercept)	0.29	0.15	0.42
j_q06bISCED 3 (EXCLUDING 3C SHORT) AND 4	1.86	1.37	2.35
j_q06bISCED 5 AND 6	1.07	0.54	1.60
age_r	1.04	1.03	1.05

```
> logit1 <- logit.sdf(I(d_q18a_t %in% c
                         ('MID-LEVEL QUINTILE',
+                         'NEXT TO HIGHEST QUINTILE',
+                         'HIGHEST QUINTILE')) ~
+                      j_q06b + age_r, data=ita)
> summary(logit1)
```

This regression shows that there is a larger contrast between individuals with mother's highest education in ‘`ISCED 3 (EXCLUDING 3C SHORT) AND 4`’ and the reference group (‘`ISCED 1, 2, AND 3C SHORT`’) at 0.62 than there is between ‘`ISCED 5 and 6`’) and the reference group at 0.07, with the former coefficient being statistically significant and the latter not. Some researchers appreciate the odds ratios when interpreting regression results. The `oddsRatio` function can show these, along with their confidence intervals. The results are shown in Table 9.6.

```
> oddsRatio(logit1)
```

The `oddsRatio` function works only for results from the `logit.sdf` function—not `probit.sdf` results—because only logistic regression has invariant odds ratios.

Although the t-test statistic in logistic regression output is a good test for an individual regressor (such as `age_r`), a Wald test is needed to conduct joint hypothesis testing. Typically, it is possible to use the Akaike information criterion (AIC) (Akaike 1974) or a likelihood-ratio test. However, the likelihood shown in the results is actually a pseudo-likelihood, or a population estimate likelihood for the model. Because the entire population was not sampled, deviance-based tests—such as those shown in McCullagh and Nelder (1989)—cannot be used. Although it would be possible to use Lumley and Scott (2015) to form an AIC comparison, that does not account for plausible values.[6]

For example, it would be reasonable to ask if the j_j06b variable is jointly significant. To test this, we can use a Wald test

```
> waldTest(model=logit1, coef='j_q06b')

Wald test:
----------
H0:
j_q06bISCED 3 (EXCLUDING 3C SHORT) AND 4 = 0
j_q06bISCED 5 AND 6 = 0

Chi-square test:
X2 = 21.1, df = 2, P(> X2) = 2.6e-05

F test:
W = 10.4, df1 = 2, df2 = 79, P(> W) = 9.6e-05
```

This is a test of both coefficients in j_q06b being zero. Two test results are shown: the chi-square test and the F-test. In the case of a well-known sample design, it probably makes more sense to use the F-test (Korn and Graubard 1990).

9.6.3 Gap Analysis

A gap analysis compares the levels of two groups and tests if they are different. The gap function supports testing gaps in mean scores, survey responses, score percentiles, and achievement levels. In this section, we discuss gaps in mean scores.

The simplest gap is within a single survey on a score and requires a selection of two groups. In the following example, we compare literacy scores of the self-employed and those who are employees

[6] The use of plausible values is allowed by logit.sdf and probit.sdf. An example of an outcome with plausible values would be a comparison of literature scores above the user-specified cutoff.

```
> gap(variable='lit', data=ita, groupA= d_q04 %in%
                                   'SELF-EMPLOYED',
+                                 groupB= d_q04
                                   %in% 'EMPLOYEE')

Call: gap(variable = "lit", data = ita, groupA = d_q04
                                   %in% "SELF-EMPLOYED",
    groupB = d_q04 %in% "EMPLOYEE")

Labels:
 group                 definition nFullData nUsed
     A d_q04 %in% "SELF-EMPLOYED"      4621   637
     B      d_q04 %in% "EMPLOYEE"      4621  2165

Percentage:
     pctA    pctAse     pctB    pctBse    diffAB
 23.05259 0.8760763 76.94741 0.8760763 -53.89482

                                   covAB diffABse
                              -0.7675097 1.752153

 diffABpValue    dofAB
            0 87.26671

Results:
 estimateA estimateAse estimateB estimateBse   diffAB
  256.6286    2.483797  253.5839    1.567581 3.044695
                                                 covAB
                                             0.9716681

 diffABse diffABpValue    dofAB
 2.585192     0.243015 67.82052
```

The gap output contains three blocks: labels, percentage, and results.

In the first block, ‘labels’, the definition of the groups `A` and `B` is shown, along with a reminder of the full data *n* count (`nFullData`) and the *n* count of the number of individuals who are in the two subgroups with valid scores (`nUsed`).

The second block, ‘percentage’, shows the percentage of individuals who fall into each category, with omitted levels removed. In the preceding example, the estimated percentage of Italians who are self-employed (in Group A) is shown in the `pctA` column, and the percentage of employees (in Group B) is shown in the `pctB` column. In this case, the only nonomitted levels are ‘`SELF-EMPLOYED`’ and ‘`EMPLOYEE`’, so they add up to 100%. The other columns listed in the ‘percentage’ block regard uncertainty in those percentages and tests determining whether the two percentages are equal.

The third block, 'results', shows the estimated average literacy score for Italians who are self-employed (Group A) in column `estimateA` and the estimated average literacy score of Italians who are employees in column `estimateB`. The `diffAB` column shows that the estimated difference between these two statistics is 3.04 literacy scale score points, whereas the `diffABse` column shows that the estimate has a standard error of 2.59 scale score points. A t-test for the difference being zero has a p-value of 0.24 is shown in column `difABpValue`.

Some software does not calculate a covariance between groups when the groups consist of distinct individuals. When survey collection was administered in such a way that respondents have more in common than randomly selected individuals—as in the Italian PIAAC sample—this is not consistent with the survey design. When there is no covariance between two units in the same variance estimation strata—as in the case of countries that use one-stage sampling—there is little harm in estimating the covariance, because it will be close to zero.

The gap output information listed is not exhaustive; similar to other `EdSurvey` functions, the user can see the list of output variables using the `?` function and typing the function of interest.

```
> ?gap # output not shown
```

The 'Value' section describes all columns contained in gap outputs.

Another type of gap compares results across samples. For example, the male/female gap in literacy scores can be compared between Italy and the Netherlands by forming an `edsurvey.data.frame.list` and running `gap` with that combined data.

```
> # form the edsurvey.data.frame.list
> ita_nld <- edsurvey.data.frame.list(datalist=list(ita, nld))
> # run the gap
> gap(variable='lit', data=ita_nld, groupA= gender_r %in% 'MALE',
+                                    groupB= gender_r %in% 'FEMALE')

gapList
Call: gap(variable = "lit", data = ita_nld, groupA = gender_r %in%
    "MALE", groupB = gender_r %in% "FEMALE")

Labels:
 group            definition
     A   gender_r %in% "MALE"
     B gender_r %in% "FEMALE"

Percentage:
     country     pctA     pctAse     pctB     pctBse      diffAB
       Italy 50.00314 0.05349453 49.99686 0.05349453 0.006289097
 Netherlands 50.20262 0.12935306 49.79738 0.12935306 0.405249502
```

(continued)

```
        covAB  diffABse diffABpValue    dofAB      diffAA covAA
  -0.002861664 0.1069891    0.9536079 24.20301         NA    NA
  -0.016732214 0.2587061    0.1225427 59.55281 -0.1994802     0
   diffAAse diffAApValue    dofAA    diffBB covBB  diffBBse
        NA           NA       NA        NA    NA        NA
  0.1399781    0.1582179 76.18208 0.1994802     0 0.1399781
  diffBBpValue    dofBB   diffABAB covABAB diffABABse diffABABpValue
           NA       NA          NA      NA         NA             NA
     0.1582179 76.18208 -0.3989604       0  0.2799563      0.1582179
   dofABAB
       NA
  76.18208

 Results:
      country estimateA estimateAse estimateB estimateBse    diffAB
        Italy  250.3554    1.488650  250.6100    1.325433 -0.254644
  Netherlands  287.0560    1.066479  280.9205    1.023297  6.135510
        covAB diffABse diffABpValue    dofAB    diffAA covAA diffAAse
   0.44350144 1.756658 0.8851353824 74.31867        NA    NA       NA
  -0.06822208 1.523469 0.0001594966 60.61344 -36.70064     0 1.831244
  diffAApValue    dofAA    diffBB covBB diffBBse diffBBpValue   dofBB
           NA       NA        NA    NA       NA           NA       NA
            0 161.3324 -30.31049     0 1.674488            0 127.6201
   diffABAB covABAB diffABABse diffABABpValue  dofABAB sameSurvey
        NA      NA         NA             NA       NA         NA
  -6.390154       0   2.325254    0.006814802 134.7154      FALSE
```

This output contains the same three blocks and columns as in the previous gap analysis. Several additional columns have been added, focusing on the contrasts between Italy and the Netherlands. The results block columns labelled with an `AA`, such as `diffAA`, compare Italian males to Dutch males. The columns labelled with a `BB`, such as `diffBB`, compare Italian females to Dutch females. Here the `diffAA` column has a value of −36.7, indicating that Italian males have an average scale score 36.7 points less than Dutch males. The column `diffAAse` has a value of 1.83, indicating that the standard error of that difference is 1.83. The two samples were collected separately, so there is no covariance in these estimates, and the `covAA` column is zero.

It also is possible to compare the male/female gap in literacy scores within and across countries. Looking at the `diffAB` column, the gap is −0.25 in Italy and 6.13 in the Netherlands, indicating that females outscore males in Italy, but males outscore females in the Netherlands. The `diffABAB` column shows that the difference in the gaps is −6.39, with a standard error (taken from `diffABABse`) of 2.32, and an associated *p*-value of 0.007, taken from `diffABABpValue`.

Table 9.7 Results from `percentile(variable = 'lit', percentiles = c(10, 25, 50, 75, 90), data = ita)`

Percentile	Estimate	se	df	confInt.ci_lower	confInt.ci_upper
10.00	192.37	2.28	22.30	187.22	196.75
25.00	221.86	1.46	11.08	217.99	225.34
50.00	252.44	1.32	16.07	249.82	255.25
75.00	282.17	1.17	14.62	279.63	284.77
90.00	306.16	1.22	22.55	303.28	309.42

9.6.4 Percentile Analysis

Discussions presented so far have focused on the mean and other measures of centrality. This section describes the `percentile` function, which calculates statistics regarding the distribution of continuous variables—namely, the percentiles of a numeric variable in the range 0 to 100 for a survey dataset. For example, to compare the PIAAC index of reading skills at home ('`lit`') at the 10th, 25th, 50th, 75th, and 90th percentile, include these as integers in the `percentiles` argument; the results are shown in Table 9.7.

```
> percentile(variable = 'lit',
+            percentiles = c(10, 25, 50, 75, 90),
+            data = ita)
```

If researchers are interested in a comparison of percentile distributions between males and females, the `subset` function can be used together with the `percentile` function. Alternatively, EdSurvey's `gap` function, covered in Sect. 9.6.3, can calculate distributions in percentiles. The results of the percentile by gender are shown in Table 9.8.

```
> percentile(variable = 'lit',
+            percentiles = c(25, 50, 75),
+            data = subset(ita, gender_r %in% 'MALE'))
> percentile(variable = 'lit',
+            percentiles = c(25, 50, 75),
+            data = subset(ita, gender_r %in% 'FEMALE'))
```

Table 9.8 Results from `percentile` by `gender_r`

gender_r	Percentile	Estimate	se	df	confInt.ci_lower	confInt.ci_upper
MALE	25.00	219.55	2.94	10.90	214.76	224.24
MALE	50.00	251.82	1.85	17.52	247.98	256.11
MALE	75.00	283.94	2.08	18.42	279.94	287.91
FEMALE	25.00	223.70	2.16	22.93	219.49	227.81
FEMALE	50.00	252.90	0.97	15.79	249.85	256.02
FEMALE	75.00	280.59	1.33	12.13	277.46	284.04

9.6.5 Proficiency Level Analysis

Scale score averages and distributions have the advantage of being numeric expressions of respondent ability; they also have the disadvantage of being essentially impossible to interpret or compare to an external benchmark. Proficiency levels, developed by experts to compare scores with performance criteria, provide an external benchmark against which scale scores can be compared (PIAAC Numeracy Expert Group 2009).

In `EdSurvey`, users can see the proficiency level cutpoints with the `showCutPoints` function:

```
> showCutPoints(ita)

Achievement Levels:
  Numeracy:  176, 226, 276, 326, 376
  Literacy:  176, 226, 276, 326, 376
  Problem Solving:  241, 291, 341
```

The `achievementLevels` function applies appropriate weights and the variance estimation method for each `edsurvey.data.frame`, with several arguments for customising the aggregation and output of the analysis results.[7] Namely, by using these optional arguments, users can

– choose to generate the percentage of individuals performing at each proficiency level (**discrete**) or at or above each proficiency level (**cumulative**),

[7]The terms *proficiency levels*, *benchmarks*, or *achievement levels* are all operationalised in the same way: individuals above a cutpoint are regarded as having met that level of proficiency or benchmark or have that achievement. `EdSurvey` calls all these *achievement levels* in the function names, cutpoints, and documentation. But the difference is entirely semantic and so can be ignored.

Table 9.9 Results from `achievementLevels(c('lit', 'gender_r'), data=ita, aggregateBy = 'gender_r', returnDiscrete = FALSE, returnCumulative = TRUE)`

Level	gender_r	N	wtdN	Percent	StandardError
Below PL 1	MALE	107.00	1178474.99	6.03	0.86
At or Above PL 1	MALE	2113.00	18379167.00	93.97	0.86
At or Above PL 2	MALE	1651.80	13848243.51	70.81	1.51
At or Above PL 3	MALE	756.00	6060156.78	30.99	1.50
At or Above PL 4	MALE	101.40	796244.02	4.07	0.55
At PL 5	MALE	2.70	14647.88	0.07	0.08
Below PL 1	FEMALE	111.90	995395.47	5.09	0.74
At or Above PL 1	FEMALE	2257.10	18559786.68	94.91	0.74
At or Above PL 2	FEMALE	1794.10	14366053.72	73.46	1.39
At or Above PL 3	FEMALE	761.40	5622973.69	28.75	1.39
At or Above PL 4	FEMALE	76.70	510122.91	2.61	0.45
At PL 5	FEMALE	1.50	7064.90	0.04	0.05

– calculate the percentage distribution of individuals by proficiency level (`discrete` or `cumulative`) and selected characteristics (specified in `aggregateBy`), and
– compute the percentage distribution of individuals by selected characteristics within a specific proficiency level.

The `achievementLevels` function also can produce statistics by both discrete and cumulative proficiency levels. By default, the `achievementLevels` function produces the results only for discrete proficiency levels. Setting the `returnCumulative` argument to `TRUE` generates results by both discrete and cumulative proficiency levels.

The `achievementLevels` function can calculate the overall cumulative proficiency level analysis of the literacy. These results are shown in Table 9.9, where the term 'Performance Level' has been replaced by 'PL' for brevity.

```
> achievementLevels(c('lit', 'gender_r'),
+                   data=ita,
+                   aggregateBy='gender_r',
+                   returnDiscrete=FALSE,
+                   returnCumulative=TRUE)
```

This call requests that the Italian literacy proficiency levels can be broken down by the `gender_r` variable—the `aggregateBy` argument is set to '`gender_r`' and therefore the `Percent` column sums to 100 within each gender. The results show that 31% of Italian males are at or above Proficiency Level 3, whereas 28.8%

of Italian females are at or above Proficiency Level 3. Note that proficiency levels are useful only if considered in the context of the descriptor, which is available from NCES at https://nces.ed.gov/surveys/piaac/litproficiencylevel.asp.

The advantage of cumulative proficiency levels is that increases are always unambiguously good. Conversely, discrete proficiency levels can change because individuals moved between levels, making their interpretation ambiguous, although increases in the highest and lowest proficiency levels are always unambiguously good (highest) or bad (lowest).

9.7 Expansion

The `EdSurvey` package continues to be developed, and new features are added in each subsequent release. To learn about current features, visit the EdSurvey webpage to see the latest version and most recent documentation.[8] The webpage also has many user guides and a complete explanation of the methodology involved in `EdSurvey`.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*(6), 716–723.

Binder, D. A. (1983). On the variances of asymptotically normal estimators from complex surveys. *International Statistical Review, 51*(3), 279–292, doi:10.2307/1402588.

Korn, E. L., & Graubard, B. I. (1990). Simultaneous testing of regression coefficients with complex survey data: Use of Bonferroni t statistics. *The American Statistician, 44*(4):270–276, doi:10.1080/00031305.1990.10475737.

Lumley, T., & Scott, A. (2015). AIC and BIC for modeling with complex survey data. *Journal of Survey Statistics and Methodology, 3*(1), 1–18, doi:10.1093/jssam/smu021.

McCullagh, P., & Nelder, J. (1989). *Generalized linear models* (Chapman and Hall/CRC Monographs on statistics and applied probability series, 2nd ed.) Boca Raton: Chapman & Hall.

Mohadjer, L., Krenzke, T., Van de Kerckhove, W., & Li, L. (2016). Sampling design. In I. Kirsch, & W. Thorn (Eds.), *Survey of adult skills technical report* (2nd ed., chapter 14, pp. 14-1–14-36). Paris: OECD.

PIAAC Numeracy Expert Group. (2009). *PIAAC numeracy: A conceptual framework*. Technical Report 35. Paris: OECD Publishing.

R Core Team. (2019). *R: A language and environment for statistical computing*. Vienna: R Foundation for Statistical Computing.

Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. Hoboken: Wiley.

Rust, K., & Johnson, E. (1992). Sampling and weighting in the national assessment. *Journal of Educational and Behavioral Statistics, 17*(2), 111–129, doi:10.3102/10769986017002111.

[8] https://www.air.org/project/nces-data-r-project-edsurvey

Satterthwaite, F. E. (1946). An approximate distribution of estimates of variance components. *Biometrics Bulletin, 2*(6), 110–114.
Wikipedia Contributors. (2019). *Welch-Satterthwaite equation—Wikipedia, the free encyclopedia*. [Online; Accessed 24 Feb 2019]