Testing reportRmd Labels

Install the new version - this is on a separate branch, so that if you want to revert you the development or CRAN versions you can simply delete the reportRmd package folder and re-install.

# if you don't have devtools installed:  
#install("devtools")  
  
# otherwise the variable label version is here:  
devtools::install\_github("biostatsPMH/reportRmd", ref="varLabs")

Skipping install of 'reportRmd' from a github remote, the SHA1 (f313b10a) has not changed since last install.  
 Use `force = TRUE` to force installation

# For the most recent stable version  
# devtools::install\_github("biostatsPMH/reportRmd", ref="development")  
library(reportRmd)

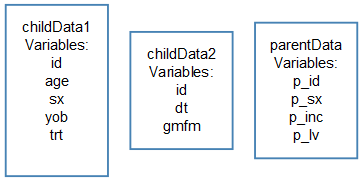
# Basic Idea

I’ve tried not to alter the basic workings of the package.

The labelling system works by:

1. creating a data frame that contains variable/label pairs
2. specifying that this is a variable label data frame using the setVariableLabels function
3. running the rm\_covsum, rm\_uvsum or rm\_mvsum as usual

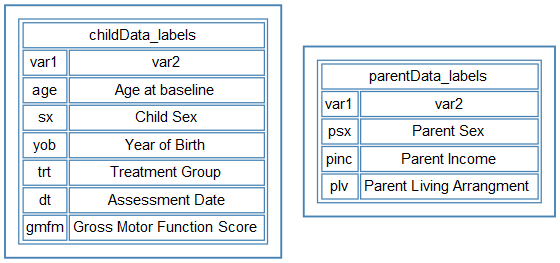
Imagine these are the three data sets your report uses:



You can create a default table of variable/label pairs for the child data, and a separate table specific to the parent data.

We’ll create two: one for the child data, which will be the default and one for the parent data. Each data frame has two columns, one for variable names, the other for labels. The columns names can be whatever you want, here they are simply var1 and var2.

**Step 1: create variable/label look-up tables** – you can specify a default for all data frames, or data frame specific tables



Now we use the setVariablLabels command to tell reportRmd to use the parentData\_labels data frame as labels for the parentData data and the childData\_labels data frame as the default variable labels, for all other data frames.

**Step 2: set the new data frames as label tables using the setVariableLabels command**

setVariableLabels(childData\_labels,parentData=parentData\_labels)

Once we’ve set that, reportRmd will use the variable labels in rm\_covsum,rm\_uvsum and rm\_mvsum unless the argument nicenames is set to FALSE.

**Step 3: run the rm\_ functions as you normally would**

rm\_covsum(data=childData1,  
 maincov = 'trt',  
 covs = c('age','sx','yob'))

|  | Full Sample (n=100) | control (n=64) | intervention (n=36) | p-value |
| --- | --- | --- | --- | --- |
| **Age at baseline** |  |  |  | 0.39 |
| Mean (sd) | 5.9 (1.2) | 6.0 (1.1) | 5.8 (1.3) |  |
| Median (Min,Max) | 5.9 (3.2, 9.7) | 5.9 (3.9, 8.9) | 5.8 (3.2, 9.7) |  |
| **Child Sex** |  |  |  | 0.18 |
| Male | 68 (68) | 40 (62) | 28 (78) |  |
| Female | 32 (32) | 24 (38) | 8 (22) |  |
| **Year of Birth** |  |  |  | 0.54 |
| 2008 | 36 (36) | 23 (36) | 13 (36) |  |
| 2009 | 31 (31) | 20 (31) | 11 (31) |  |
| 2010 | 26 (26) | 18 (28) | 8 (22) |  |
| 2011 | 6 (6) | 2 (3) | 4 (11) |  |
| 2014 | 1 (1) | 1 (2) | 0 (0) |  |

# Worked Example

Get some descriptive stats for the ctDNA data that comes with the package. The nicenames argument is TRUE by default so underscores are replaced by spaces

data(ctDNA)  
rm\_covsum(data=ctDNA,  
 covs=c('cohort','ctdna\_status','size\_change'))

|  | n=270 |
| --- | --- |
| **cohort** |  |
| A | 50 (19) |
| B | 14 (5) |
| C | 18 (7) |
| D | 88 (33) |
| E | 100 (37) |
| **ctdna status** |  |
| Clearance | 137 (51) |
| No clearance, decrease from baseline | 44 (16) |
| No clearance, increase from baseline | 89 (33) |
| **size change** |  |
| Mean (sd) | -29.7 (52.8) |
| Median (Min,Max) | -32.5 (-100.0, 197.1) |
| Missing | 8 |

otherwise, we can keep the exact variable names

rm\_covsum(data=ctDNA,  
 covs=c('cohort','ctdna\_status','size\_change'),  
 nicenames = F)

|  | n=270 |
| --- | --- |
| **cohort** |  |
| A | 50 (19) |
| B | 14 (5) |
| C | 18 (7) |
| D | 88 (33) |
| E | 100 (37) |
| **ctdna\_status** |  |
| Clearance | 137 (51) |
| No clearance, decrease from baseline | 44 (16) |
| No clearance, increase from baseline | 89 (33) |
| **size\_change** |  |
| Mean (sd) | -29.7 (52.8) |
| Median (Min,Max) | -32.5 (-100.0, 197.1) |
| Missing | 8 |

We can define a table with variable labels

ctDNA\_names <- data.frame(var=names(ctDNA),  
 label=c('Patient ID',  
 'Study Cohort',  
 'Change in ctDNA since baseline',  
 'Number of weeks on treatment',  
 'Percentage change in tumour measurement'))

Then we set the variable labels that correspond to the ctDNA data like this (no quotes!):

setVariableLabels(ctDNA=ctDNA\_names)

That gave a message that we haven’t set a table of default variable names, just a table of names specific to the ctDNA data

We can look to see which data frames have variable labels and if there is a default set

getVariableLabels()

Now if we re-run our stats, the variable names are replaced by the labels

rm\_covsum(data=ctDNA,  
 covs=c('cohort','ctdna\_status','size\_change'),  
 nicenames = T)

|  | n=270 |
| --- | --- |
| **Study Cohort** |  |
| A | 50 (19) |
| B | 14 (5) |
| C | 18 (7) |
| D | 88 (33) |
| E | 100 (37) |
| **Change in ctDNA since baseline** |  |
| Clearance | 137 (51) |
| No clearance, decrease from baseline | 44 (16) |
| No clearance, increase from baseline | 89 (33) |
| **Percentage change in tumour measurement** |  |
| Mean (sd) | -29.7 (52.8) |
| Median (Min,Max) | -32.5 (-100.0, 197.1) |
| Missing | 8 |

..unless we want to keep the variable names, and they we just use nicenames=F:

rm\_covsum(data=ctDNA,  
 covs=c('cohort','ctdna\_status','size\_change'),  
 nicenames = F)

|  | n=270 |
| --- | --- |
| **cohort** |  |
| A | 50 (19) |
| B | 14 (5) |
| C | 18 (7) |
| D | 88 (33) |
| E | 100 (37) |
| **ctdna\_status** |  |
| Clearance | 137 (51) |
| No clearance, decrease from baseline | 44 (16) |
| No clearance, increase from baseline | 89 (33) |
| **size\_change** |  |
| Mean (sd) | -29.7 (52.8) |
| Median (Min,Max) | -32.5 (-100.0, 197.1) |
| Missing | 8 |

We can define variable labels for the pembrolizumab data and add these as the default variable labels to use for all data frames without a specific table

varNames <- data.frame(var=names(pembrolizumab),  
 label=c('Patient ID','Age at study entry','Patient Sex','Study Cohort','Target lesion size at baseline','PD L1 percent','log of tumour size','Baseline ctDNA','Did ctDNA increase or decrease from baseline to cycle 3','Objective Response','Clinical Beneficial Response','Overall survival status', 'Overall survival time in months','Progression free survival status','Progression free survival time in months'))  
  
setVariableLabels(varNames) # equivalent to setVariableLabels(default=varNames)

default variable names will be updated.

Now when we check we can see that there is a default variable table as well

getVariableLabels()

data.frame variable.names  
1 ctDNA ctDNA\_names  
2 default varNames

If we subset the pembrolizumab data and run some stats they are output with variable labels - because we have set default labels, we don’t need to set separate labels for the cohort\_A data.

cohort\_A <- pembrolizumab %>%  
 filter(cohort=="A")  
  
rm\_uvsum(data=cohort\_A, response='orr',  
covs=c('age','pdl1','change\_ctdna\_group'))

|  | OR(95%CI) | p-value | N | Event |
| --- | --- | --- | --- | --- |
| **Age at study entry** | 0.87 (0.62, 1.10) | 0.30 | 16 | 13 |
| **PD L1 percent** | 0.88 (0.46, 0.98) | 0.46 | 15 | 12 |
| **change ctdna group** |  | 1.00 | 14 | 11 |
| Decrease from baseline | Reference |  | 6 | 3 |
| Increase from baseline | 8.5e+08 (0e+00, NA) |  | 8 | 8 |

You don’t need to specify labels for all variables. Variables without labels will just be displayed as usual (using nicenames).

If we change the labels:

varNames <- data.frame(var=c('l\_size','pdl1','tmb'),  
 label=c('Target lesion size at baseline','PD L1 percent','log of tumour size'))  
  
setVariableLabels(varNames) # equivalent to setVariableLabels(default=varNames)

default variable names will be updated.

And get some new stats we see labels if we set them and nice variable names otherwise

rm\_covsum(data=cohort\_A,   
 covs=c('change\_ctdna\_group','pdl1','l\_size','tmb'))

|  | n=16 |
| --- | --- |
| **change ctdna group** |  |
| Decrease from baseline | 6 (43) |
| Increase from baseline | 8 (57) |
| Missing | 2 |
| **PD L1 percent** |  |
| Mean (sd) | 32.8 (42.1) |
| Median (Min,Max) | 2 (0, 100) |
| Missing | 1 |
| **Target lesion size at baseline** |  |
| Mean (sd) | 84.4 (39.8) |
| Median (Min,Max) | 90.5 (11.0, 152.0) |
| **log of tumour size** |  |
| Mean (sd) | 0.8 (0.4) |
| Median (Min,Max) | 0.8 (0.1, 1.6) |

# Working With Labelled Packages

## expss

Can can also use the expss package to add labels:

library(expss)

data(mtcars)  
mtcars = apply\_labels(mtcars,  
 mpg = "Miles/(US) gallon",  
 cyl = "Number of cylinders",  
 disp = "Displacement (cu.in.)",  
 hp = "Gross horsepower",  
 drat = "Rear axle ratio",  
 wt = "Weight (1000 lbs)",  
 qsec = "1/4 mile time",  
 vs = "Engine",  
 vs = c("V-engine" = 0,  
 "Straight engine" = 1),  
 am = "Transmission",  
 am = c("Automatic" = 0,  
 "Manual"=1),  
 gear = "Number of forward gears",  
 carb = "Number of carburetors"  
)  
extractLabels(mtcars) # this extracts the labels to a data frame called mtcars\_names and adds it to the list of variable label tables

mtcars\_names created and added to list of variable tables

var lbl  
1 mpg Miles/(US) gallon  
2 cyl Number of cylinders  
3 disp Displacement (cu.in.)  
4 hp Gross horsepower  
5 drat Rear axle ratio  
6 wt Weight (1000 lbs)  
7 qsec 1/4 mile time  
8 vs Engine  
9 am Transmission  
10 gear Number of forward gears  
11 carb Number of carburetors

Check that these have been added to the list of variable labels

getVariableLabels()

data.frame variable.names  
1 ctDNA ctDNA\_names  
2 default varNames  
3 mtcars mtcars\_names

## Haven

If you use haven to load SPSS data with labels you can just pull out the labels and add them as well

library(haven)

Attaching package: 'haven'

The following objects are masked from 'package:expss':  
  
 is.labelled, read\_spss

# this is just to show the command to read spss data  
#df <- read\_sav("some\_data.sav")  
#save(df,file='some\_SPSS\_data.rda')

head(df)

# A tibble: 6 × 5  
 Subject\_ID Age Gender Children TimeDx  
 <dbl> <dbl> <dbl+lbl> <dbl+lbl> <dbl>  
1 1 43 2 [female] 0 [No] 1.92  
2 2 51 2 [female] 1 [Yes] 2.59  
3 3 50 2 [female] 1 [Yes] 1.34  
4 4 72 2 [female] 1 [Yes] 2.51  
5 5 NA NA NA NA   
6 6 46 2 [female] 1 [Yes] 3.50

extractLabels(df,default = T) # this uses the labels from df as the default labels

rm\_covsum(data=df,  
 covs=c('Age','Gender','Children'))

|  | n=302 |
| --- | --- |
| **Age** |  |
| Mean (sd) | 55.7 (11.1) |
| Median (Min,Max) | 55 (30, 90) |
| Missing | 7 |
| **Gender** |  |
| Mean (sd) | 2.0 (0.1) |
| Median (Min,Max) | 2 (1, 2) |
| Missing | 6 |
| **Do you have children?** |  |
| Mean (sd) | 0.7 (0.4) |
| Median (Min,Max) | 1 (0, 1) |
| Missing | 7 |

## sjlabelled

Load in data using the sjlabelled package, extract the names and do some stats:

library(sjlabelled)

data(efc)  
extractLabels(efc)

rm\_covsum(data=efc,  
 covs=c('e17age','e16sex','nur\_pst'))

|  | n=908 |
| --- | --- |
| **elder’ age** |  |
| Mean (sd) | 79.1 (8.1) |
| Median (Min,Max) | 79 (65, 103) |
| Missing | 17 |
| **elder’s gender** |  |
| Mean (sd) | 1.7 (0.5) |
| Median (Min,Max) | 2 (1, 2) |
| Missing | 7 |
| **Care level** |  |
| Mean (sd) | 1.9 (0.8) |
| Median (Min,Max) | 2 (1, 3) |
| Missing | 419 |

To clear all variable labels:

clearVariableLabels()

Check

getVariableLabels()

[1] variable.names  
<0 rows> (or 0-length row.names)