Package: readspss (via r-universe)

November 16, 2024

Type Package

Title Importing and Exporting SPSS Files

Version 0.17.1

Maintainer Jan Marvin Garbuszus <jan.garbuszus@ruhr-uni-bochum.de>

Description Package to read and write the SPSS file formats.

URL https://github.com/JanMarvin/readspss

BugReports https://github.com/JanMarvin/readspss/issues

License GPL-2 | file LICENSE

LazyData TRUE

Language en-US

Imports Rcpp (>= 0.11.2)

Suggests covr, datasets, foreign, knitr, rmarkdown, roxygen2, testthat

LinkingTo Rcpp, BH

ByteCompile yes

SystemRequirements OpenSSL >= 1.0.2

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Config/pak/sysreqs libssl-dev

Repository https://janmarvin.r-universe.dev

RemoteUrl https://github.com/JanMarvin/readspss

RemoteRef HEAD

RemoteSha 3ba9cc27aba964df7eadbcb51419d0cf307b2396

Contents

`read.por` *read.por*

Description

Function to read a SPSS por file into a data.frame().

Usage

```
read.por(
  file,
  convert.factors = TRUE,
  generate.factors = TRUE,
  encoding = TRUE,
  fromEncoding = NULL,
  use.missings = TRUE,
  debug = FALSE,
  override = FALSE,
  convert.dates = TRUE,
  add.rownames = FALSE
)
```

Arguments

`file`	*string* a por-file to import. can be a file on a computer or an url. in this case the file will be downloaded and read before it is used.
`convert.factors`	*logical* if true numeric or character variables will be converted into a factor in R.
`generate.factors`	*logical* function to convert variables with partial labels into factors. e.g. 1 - low and 5 - high are provided, labels 2, 3 and 4 will be created. especially useful in combination with `use.missings=TRUE`.
`encoding`	*logical* shall values be converted? If true, `read.por()` will try the charcode stored inside the por-file. If this value is 2 or not available, `fromEncoding` can be used to change encoding.

`fromEncoding`	*character* encoding of the imported file. This information is stored inside the por-file, but is currently unused. Still this option can be used to define the initial encoding by hand.
`use.missings`	*logical* should missing values be converted. Defaults to TRUE.
`debug`	*logical* provides additional debug information. Most likely not useful to any user.
`override`	*logical* The filename provided in `file` is checked for the ending por. If the file ending is different, nothing is read. This option can be used to override this behavior.
`convert.dates`	*logical* Should dates be converted on the fly?
`add.rownames`	*logical* If `TRUE`, the first column will be used as rownames. Variable will be dropped afterwards.

Details

SPSS files are widely available, though for R long time only foreign and memisc provided functions to import por-files. Lately haven joined. This package is an approach to offer another alternative, to document the por-format and provide additional options to import the data.

Note

Information to decrypt the por-format was provided by tda www.stat.rub.de/tda.html and pspp www.gnu.org/software/pspp/

See Also

`read.spss`, `memisc`.

`read.sav` *read.sav*

Description

Function to read a SPSS sav file into a data.frame().

Usage

```
read.sav(
  file,
  convert.factors = TRUE,
  generate.factors = TRUE,
  encoding = TRUE,
  fromEncoding = NULL,
  use.missings = TRUE,
  debug = FALSE,
  override = FALSE,
```

```
  convert.dates = TRUE,
  add.rownames = FALSE,
  pass
)
```

Arguments

`file`	*string* a sav-file to import. can be a file on a computer or an url. in this case the file will be downloaded and read before it is used.
`convert.factors`	*logical* if true numeric or character variables will be converted into a factor in R.
`generate.factors`	*logical* function to convert variables with partial labels into factors. e.g. 1 - low and 5 - high are provided, labels 2, 3 and 4 will be created. especially useful in combination with `use.missings=TRUE`.
`encoding`	*logical* shall values be converted? If true, read.sav will try the charcode stored inside the sav-file. If this value is 2 or not available, fromEncoding can be used to change encoding.
`fromEncoding`	*character.* encoding of the imported file. This information is stored inside the sav-file, but is currently unused. Still this option can be used to define the initial encoding by hand.
`use.missings`	*logical* should missing values be converted. Defaults to TRUE.
`debug`	*logical* provides additional debug information. Most likely not useful to any user.
`override`	*logical*. The filename provided in `file` is checked for the ending sav. If the file ending is different, nothing is read. This option can be used to override this behavior.
`convert.dates`	*logical*. Should dates be converted on the fly?
`add.rownames`	*logical.* If `TRUE`, the first column will be used as rownames. Variable will be dropped afterwards.
`pass`	*character*. If encrypted sav should be imported, this is a maximum of ten character encryption key.

Details

SPSS files are widely available, though for R long time only foreign and memisc provided functions to import sav-files. Lately haven joined. This package is an approach to offer another alternative, to document the sav-format and provide additional options to import the data. sav-files are stored most exclusively as numerics only in compression mode are some integers stored as integers. Still they are returned as numerics.

Value

`readspss` returns a data.frame with additional attributes

- *row.names* rownames

- *names* colnames
- *datalabel* datalabel
- *datestamp* datestamp
- *timestamp* timestamp
- *filelabel* filelabel
- *class* data.frame
- *vtype* SPSS type 0 is usually a numeric/integer
- *disppar* matrix of display parameters if available
- *missings* a list containing information about the missing variables. if `use.missings=TRUE` this Information will be used to generate missings.
- *haslabel* list of variables that contain labels
- *longstring* character vector of long strings if any in file
- *longmissing* character vector of missings in longstrings if any
- *longlabel* character vector of long labels
- *cflag* 0 if uncompressed, 1 if compressed
- *endian* 2 or 3 if little endian else 0
- *compression* compression similar to cflag, somehow stored twice in the sav file
- *doc* list containing documentation information if any
- *charcode* encoding string most likely 2 is CP1252
- *encoding* sometimes sav-file contain encoding as a extra string
- *ownEnc* encoding of the R-session
- *doenc* was the file supposed to be encoded?
- *autoenc* was encoding applied to the file?
- *swapit* were the bytes swapped?
- *totals* character string of totals if any
- *dataview* xml file how the data should be printed
- *extraproduct* additional string provided
- *label* list containing label value information
- *varmatrix* a matrix with information how the data is stored
- *var.label* variable labels
- *lmissings* missings table if any in longstrings

Note

Information to decrypt the sav-format was provided by tda www.stat.rub.de/tda.html and pspp www.gnu.org/software/pspp/

See Also

`read.spss`, `memisc` and `read_sav`.

Examples

```
fl <- system.file("extdata", "electric.sav", package = "readspss")
dd <- read.sav(fl)
```

`read.spss` *read.spss*

Description

Function to read a SPSS (z)sav or por file into a data.frame(). This is just a wrapper around read.sav and read.por for convenience.

Usage

```
read.spss(x, ...)
```

Arguments

`x`	file to import
`...`	additional arguments passed to read.sav or read.por please see the documentation for these functions.

See Also

read.sav and read.por

`write.por` *write.por*

Description

Function to write an SPSS por file. Returns an por file that read.por can read as well as SPSS can. Other packages as foreign, memisc and haven might fail (fail reading or return wrong values).

Usage

```
write.por(
  dat,
  filepath,
  label,
  add.rownames = FALSE,
  convert.factors = TRUE,
  toEncoding = "CP1252",
  convert.dates = TRUE,
  tz = "GMT"
)
```

Arguments

dat	*data.frame* a data.frame to export as por-file.
filepath	*string* full path where and how this file should be stored
label	*character* vector of labels. must be of size ncol(dat)
add.rownames	*logical* If TRUE, a new variable rownames will be added to the por-file.
convert.factors	*logical* If TRUE, factors will be converted to SPSS variables with labels. SPSS expects strings to be encoded as Windows-1252, so all levels will be recoded. Character which can not be mapped in Windows-1252 will be saved as hexcode.
toEncoding	*character* encoding used for the por file. SPSS itself claims to have problems with unicode and por files, so "CP1252" is the default.
convert.dates	*logical* should dates be converted to SPSS format
tz	*character* The name of the timezone convert.dates will use.

Details

Strings longer than 255 chars are not provided. File will be stored using "CP1252" encoding.

Value

write.por returns nothing

---

write.sav *write.sav*

Description

Function to write an SPSS sav or zsav file from a data.frame().

Usage

```
write.sav(
  dat,
  filepath,
  label,
  add.rownames = FALSE,
  compress = FALSE,
  convert.dates = TRUE,
  tz = "GMT",
  debug = FALSE,
  is_zsav = FALSE,
  disppar
)
```

Arguments

dat	*data.frame* a data.frame to store as SPSS file.
filepath	*string* full path where and how this file should be stored
label	*character* if any provided this must be a vector of labels. It must be of size ncol(dat)
add.rownames	*logical* If TRUE, a new variable rownames will be added to the sav-file.
compress	*logical* should compression be used. If TRUE some integers will be stored more efficiently. Everything will be stored in chunks of 8 chars. Reduces memory size of sav-file.
convert.dates	*logical* should dates be converted to SPSS format.
tz	*character* The name of the timezone convert.dates will use.
debug	*logical* print debug information.
is_zsav	*logical* explicitly create a zsav file. If the file ending zsav is used, this is selected as default.
disppar	optional display parameter matrix. Needs documentation.

Details

Writing of strings longer than 255 chars is not provided.

Value

write.sav returns nothing

Index