Package ‘stats19’

March 3, 2020

Title  Work with Open Road Traffic Casualty Data from Great Britain

Version  1.2.0

Description  Tools to help download, process and analyse the UK road collision data collected using the
‘STATS19’ form. The data are provided as ‘CSV’ files with detailed road safety data about the
circumstances of car crashes and other incidents on the roads resulting in
casualties in Great Britain from 1979, the types
(including make and model) of vehicles involved and the consequential casualties. The
statistics relate only to personal casualties on public roads that are reported
to the police, and subsequently recorded, using the 'STATS19' accident reporting form. See
the Department for Transport website
<https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data> for more
information on these data.

Depends  R (>= 3.4.0)

License  GPL-3

URL  https://github.com/ropensci/stats19,
 https://docs.ropensci.org/stats19/

BugReports  https://github.com/ropensci/stats19/issues

Encoding  UTF-8

LazyData  true

Imports  sf, readr

Suggests  curl (>= 3.2), dplyr, ggplot2, knitr, lubridate, markdown,
stringr, testthat (>= 2.1.0), tidyr, pkgdown, kableExtra,
leaflet, geojsonsf, htmltools, tmap, jsonlite, pct, spatstat,
osmdata

VignetteBuilder  knitr

RoxygenNote  7.0.2

Language  en-US

X-schema.org-keywords  stats19, road-safety, transport, car-crashes,
ropensci, data
NeedsCompilation no

Author Robin Lovelace [aut, cre] (<https://orcid.org/0000-0001-5679-6536>),
Malcolm Morgan [aut] (<https://orcid.org/0000-0002-9488-9183>),
Layik Hama [aut] (<https://orcid.org/0000-0003-1912-4890>),
Mark Padgham [aut] (<https://orcid.org/0000-0003-2172-5265>),
David Ranzolin [rev],
Adam Sparks [rev, ctb] (<https://orcid.org/0000-0002-0061-8359>),
Ivo Wengraf [ctb],
RAC Foundation [fnd]

Maintainer Robin Lovelace <rob00x@gmail.com>

Repository CRAN

Date/Publication 2020-03-03 22:40:02 UTC

R topics documented:

accidents_sample .................................................... 3
casualties_sample .................................................... 3
check_input_file .................................................... 4
check_year ............................................................ 4
dl_stats19 ............................................................. 5
file_names ............................................................ 6
find_file_name ....................................................... 6
format_accidents ..................................................... 7
format_casualties .................................................... 8
format_column_names ................................................. 8
format_ppp ........................................................... 9
format_sf ........................................................... 10
format_vehicles .................................................... 10
get_data_directory .................................................. 11
get_MOT .............................................................. 11
get_stats19 ............................................................ 12
get_url ............................................................... 14
locate_files ........................................................ 15
locate_one_file ..................................................... 16
phrase ................................................................. 16
police_boundaries .................................................. 17
read_accidents ...................................................... 17
read_casualties ..................................................... 18
read_vehicles ....................................................... 19
schema_original ..................................................... 20
select_file .......................................................... 20
set_data_directory .................................................. 21
stats19_schema ..................................................... 21
vehicles_sample ..................................................... 21

Index 23
**accidents_sample**  
*Sample of stats19 data (2017 accidents)*

**Description**  
Sample of stats19 data (2017 accidents)

**Format**  
A data frame

**Note**  
These were generated using the script in the data-raw directory (misc.Rmd file).

**Examples**

```r
nrow(accidents_sample_raw)
```

```r
accidents_sample_raw
```

---

**casualties_sample**  
*Sample of stats19 data (2017 casualties)*

**Description**  
Sample of stats19 data (2017 casualties)

**Format**  
A data frame

**Note**  
These were generated using the script in the data-raw directory (misc.Rmd file).

**Examples**

```r
nrow(casualties_sample_raw)
```

```r
casualties_sample_raw
```
check_input_file  Local helper to be reused.

Description
Local helper to be reused.

Usage
check_input_file(filename = NULL, type = NULL, data_dir = NULL, year = NULL)

Arguments
- **filename**: Character string of the filename of the .csv to read, if this is given, type and years determine whether there is a target to read, otherwise disk scan would be needed.
- **type**: The type of file to be downloaded (e.g. 'Accidents', 'Casualties' or 'Vehicles'). Not case sensitive and searches using regular expressions ('acc' will work).
- **data_dir**: Where sets of downloaded data would be found.
- **year**: Single year for which data are to be read

check_year  check and convert year argument

Description
check and convert year argument

Usage
check_year(year)

Arguments
- **year**: Single year for which file is to be downloaded.

Examples
# check_year("2018")  # fails
# check_year(2017)
# check_year(2006)
# check_year(1985)
**dl_stats19**

*Download STATS19 data for a year or range of two years.*

**Description**

Download STATS19 data for a year or range of two years.

**Usage**

```r
dl_stats19(
  year = NULL,
  type = NULL,
  data_dir = get_data_directory(),
  file_name = NULL,
  ask = FALSE,
  silent = FALSE
)
```

**Arguments**

- `year`: Single year for which file is to be downloaded.
- `type`: One of 'Accidents', 'Casualties', 'Vehicles'; defaults to 'Accidents'. Or any variation of to search the file names with such as "ace" or "accid".
- `data_dir`: Parent directory for all downloaded files. Defaults to `tempdir()`.
- `file_name`: The file name (DfT named) to download.
- `ask`: Should you be asked whether or not to download the files? TRUE by default.
- `silent`: Boolean. If FALSE (default value), display useful progress messages on the screen.

**Details**

This function downloads and unzips UK road crash data. It results in unzipped .csv files that are put in the temporary directory specified by `get_data_directory()` or provided `data_dir`.

The file downloaded would be for a specific year (e.g. 2017). It could also be a file containing data for a range of two (e.g. 2005-2014).

The `dl_*` functions can download many MB of data so ensure you have a sufficient internet access and hard disk space.

**See Also**

`get_stats19()`
Examples

```r
dl_stats19(year = 2017) # interactively select files...
# now you can read-in the data
dl_stats19(year = 2009)
dl_stats19(year = 2009, type = "casualties")
dl_stats19(type = "casualties")
dl_stats19(year = 1985)
```

---

**file_names**  
*stats19 file names for easy access*

---

**Description**

URL decoded file names. Currently there are 52 file names released by the DfT (Department for Transport) and the details include how these were obtained and would be kept up to date.

**Format**

A named list

**Note**

These were generated using the script in the data-raw directory (misc.Rmd file).

**Examples**

```r
## Not run:
length(file_names)
file_names$dftRoadSafetyData_Vehicles_2017.zip

## End(Not run)
```

---

**find_file_name**  
*Find file names within stats19::file_names.*

---

**Description**

Currently, there are 52 file names to download/read data from.

**Usage**

```r
find_file_name(years = NULL, type = NULL)
```
format_accidents

Arguments

years          Years for which data are to be found

type           One of 'Accidents', 'Casualties', 'Vehicles'; defaults to 'Accidents', ignores case.

Examples

find_file_name(2016)
find_file_name(2016, type = "Accidents")
find_file_name(1985, type = "Accidents")
find_file_name(type = "cas")
find_file_name(type = "accid")
find_file_name(2006)
find_file_name(2016:2017)

format_accidents  Format STATS19 'accidents' data

Description

Format STATS19 'accidents' data

Usage

format_accidents(x)

Arguments

x              Data frame created with read_accidents()

Details

This is a helper function to format raw STATS19 data

Examples

dl_stats19(year = 2017, type = "accident")
x = read_accidents(year = 2017, format = FALSE)
x[1:3, 1:12]
crashes = format_accidents(x)
crashes[1:3, 1:12]
summary(crashes$datetime)
format_casualties  Format STATS19 casualties

**Description**

Format STATS19 casualties

**Usage**

format_casualties(x)

**Arguments**

- **x**  
  Data frame created with read_casualties()

**Details**

This function formats raw STATS19 data

**Examples**

```r
dl_stats19(year = 2017, type = "casualties")
x = read_casualties(year = 2017)
casualties = format_casualties(x)
```

format_column_names  Format column names of raw STATS19 data

**Description**

This function takes messy column names and returns clean ones that work well with R by default. Names that are all lower case with no R-unfriendly characters such as spaces and - are returned.

**Usage**

format_column_names(column_names)

**Arguments**

- **column_names**  
  Column names to be cleaned

**Value**

Column names cleaned.
Examples

```r
crashes_raw = read_accidents(year = 2017)
column_names = names(crashes_raw)
column_names
format_column_names(column_names = column_names)
```

format_ppp

Convert STATS19 data into ppp (spatstat) format.

Description

This function is a wrapper around `ppp` function and it is used to transform STATS19 data into a ppp format.

Usage

```r
format_ppp(data, window = NULL, ...)
```

Arguments

- `data`: A STATS19 dataframe to be converted into ppp format.
- `window`: A windows of observation, an object of class `owin()`. If `window = NULL` (i.e. the default) then the function creates an approximate bounding box covering the whole UK. It can also be used to filter only the events occurring in a specific region of UK (see the examples of `get_stats19`).
- `...`: Additional parameters that should be passed to `ppp` function. Read the help page of that function for a detailed description of the available parameters.

Value

A ppp object.

See Also

`format_sf` for an analogous function used to convert data into sf format and `ppp` for the original spatstat function.

Examples

```r
if (requireNamespace("spatstat", quietly = TRUE)) {
  x_ppp = format_ppp(accidents_sample)
  spatstat::plot.ppp(spatstat::unmark(x_ppp))
}
```
format_sf | Format convert STATS19 data into spatial (sf) object

Description
Format convert STATS19 data into spatial (sf) object

Usage
format_sf(x, lonlat = FALSE)

Arguments
x Data frame created with read_accidents()
lonlat Should the results be returned in longitude/latitude? By default FALSE, meaning the British National Grid (EPSG code: 27700) is used.

Examples
x_sf = format_sf(accidents_sample)
sf:::plot.sf(x_sf)

format_vehicles | Format STATS19 vehicles data

Description
Format STATS19 vehicles data

Usage
format_vehicles(x)

Arguments
x Data frame created with read_vehicles()

Details
This function formats raw STATS19 data

Examples
dl_stats19(year = 2017, type = "vehicles", ask = FALSE)
x = read_vehicles(year = 2017, format = FALSE)
vehicles = format_vehicles(x)
get_data_directory  Get data download dir

Description
Get data download dir

Usage
get_data_directory()

Examples
# get_data_directory()

get_MOT  Download vehicle data from the DVSA MOT API using VRM.

Description
Download vehicle data from the DVSA MOT API using VRM.

Usage
get_MOT(vrm, apikey)

Arguments
vrm  A list of VRMs as character strings.
apikey  Your API key as a character string.

Details
This function takes a character vector of vehicle registrations (VRMs) and returns vehicle data from MOT records. It returns a data frame of those VRMs which were successfully used with the DVSA MOT API.

Information on the DVSA MOT API is available here: https://dvsa.github.io/mot-history-api-documentation/

The DVSA MOT API requires a registration. The function therefore requires the API key provided by the DVSA. Be aware that the API has usage limits. The function will therefore limit lists with more than 150,000 VRMs.
get_stats19

Examples

vrm = c("1RAC","P1RAC")
apikey = Sys.getenv("MOTKEY")
if(nchar(apikey) > 0) {
  get_MOT(vrm = vrm, apikey = apikey)
}

get_stats19

Description

Download, read and format STATS19 data in one function.

Usage

get_stats19(
  year = NULL,
  type = "accidents",
  data_dir = get_data_directory(),
  file_name = NULL,
  format = TRUE,
  ask = FALSE,
  silent = FALSE,
  output_format = "tibble",
  ...
)

Arguments

year
  Single year for which file is to be downloaded.

type
  One of 'Accidents', 'Casualties', 'Vehicles'; defaults to 'Accidents'. Or any
  variation of to search the file names with such as "ace" or "accid".

data_dir
  Parent directory for all downloaded files. Defaults to tempdir().

file_name
  The file name (DfT named) to download.

format
  Switch to return raw read from file, default is TRUE.

ask
  Should you be asked whether or not to download the files? TRUE by default.

silent
  Boolean. If FALSE (default value), display useful progress messages on the
  screen.

output_format
  A string that specifies the desired output format. The default value is "tibble".
  Other possible values are "data.frame", "sf" and "ppp", that, respectively,
  returns objects of class data.frame, sf::sf and spatstat::ppp. Any other
  string is ignored and a tibble output is returned. See details and examples.

...

Other arguments that should be passed to format_sf() or format_ppp() func-
tions. Read and run the examples.
get_stats19

Details

This function utilizes dl_stats19 and read_* functions and returns a tibble (default), a data.frame, a sf object or a ppp object (according to the output_format parameter). The file downloaded would be for a specific year (e.g. 2017) or multiple years (e.g. c(2017, 2018)). See examples.

As this function uses dl_stats19 function, it can download many MB of data, so ensure you have a sufficient disk space.

If output_format = "data.frame" or output_format = "sf" or output_format = "ppp" then the output data is transformed into a data.frame, sf or ppp object using the as.data.frame() or format_sf() or format_ppp() functions, respectively. See examples.

See Also
dl_stats19()
read_accidents()

Examples

# default tibble output
x = get_stats19(2009)
class(x)
x = get_stats19(2017, silent = TRUE)

# data.frame output
x = get_stats19(2009, silent = TRUE, output_format = "data.frame")
class(x)

# multiple years
get_stats19(c(2017, 2018), silent = TRUE)

# sf output
x_sf = get_stats19(2017, silent = TRUE, output_format = "sf")

# sf output with lonlat coordinates
x_sf = get_stats19(2017, silent = TRUE, output_format = "sf", lonlat = TRUE)
f sf::st_crs(x_sf)

# multiple years
get_stats19(c(2017, 2018), silent = TRUE, output_format = "sf")

if (requireNamesp ace("spatstat", quietly = TRUE)) {
# ppp output
x_ppp = get_stats19(2017, silent = TRUE, output_format = "ppp")
spatstat::plot.ppp(x_ppp, use.marks = FALSE)

# Multiple years
get_stats19(c(2017, 2018), silent = TRUE, output_format = "ppp")

# We can use the window parameter of format_ppp function to filter only the
# events occurred in a specific area. For example we can create a new bbox
# of 5km around the city center of Leeds

leeds_window = spatstat::owin(
xrange = c(425046.1, 435046.1),
yrange = c(428577.2, 438577.2)
)

leeds_ppp = get_stats19(2017, silent = TRUE, output_format = "ppp", window = leeds_window)
spatstat::plot.ppp(leeds_ppp, use.marks = FALSE, clipwin = leeds_window)

# or even more fancy examples where we subset all the events occurred in a
# pre-defined polygon area

# The following example requires osmdata package
# greater_london_sf_polygon = osmdata::getbb(
# "Greater London, UK",
# format_out = "sf_polygon"
# )

# spatstat works only with planar coordinates
# greater_london_sf_polygon = sf::st_transform(greater_london_sf_polygon, 27700)
# then we extract the coordinates and create the window object.
# greater_london_polygon = sf::st_coordinates(greater_london_sf_polygon)[, c(1, 2)]
# greater_london_window = spatstat::owin(poly = greater_london_polygon)

# greater_london_ppp = get_stats19(2017, output_format = "ppp", window = greater_london_window)
# spatstat::plot.ppp(greater_london_ppp, use.marks = FALSE, clipwin = greater_london_window)

---

get_url

**Convert file names to urls**

**Description**

Convert file names to urls

**Usage**

get_url(file_name = "",
        domain = "http://data.dft.gov.uk.s3.amazonaws.com",
        directory = "road-accidents-safety-data")

**Arguments**

file_name: Optional file name to add to the url returned (empty by default)
domain: The domain from where the data will be downloaded
directory: The subdirectory of the url
**locate_files**

**Details**

This function returns urls that allow data to be downloaded from the pages:


Last updated: 22nd Nov 2018. Files available from the s3 url in the default domain argument.

**Examples**

```
# get_url(find_file_name(1985))
```

---

**locate_files**

*Locate a file on disk*

**Description**

Helper function to locate files. Given below params, the function returns 0 or more files found at location/names given.

**Usage**

```
locate_files(
  data_dir = get_data_directory(),
  type = NULL,
  years = NULL,
  quiet = FALSE
)
```

**Arguments**

- **data_dir**  
  Super directory where dataset(s) were first downloaded to.
- **type**  
  One of 'Accidents', 'Casualties', 'Vehicles'; defaults to 'Accidents', ignores case.
- **years**  
  Years for which data are to be found
- **quiet**  
  Print out messages (files found)

**Value**

Character string representing the full path of a single file found, list of directories where data from the Department for Transport (stats19::filenames) have been downloaded, or NULL if no files were found.
locate_one_file  
Pin down a file on disk from four parameters.

Description
Pin down a file on disk from four parameters.

Usage

locate_one_file(
  filename = NULL,
  data_dir = get_data_directory(),
  year = NULL,
  type = NULL
)

Arguments

filename  Character string of the filename of the .csv to read, if this is given, type and years determine whether there is a target to read, otherwise disk scan would be needed.
data_dir  Where sets of downloaded data would be found.
year  Single year for which file is to be found.
type  One of: 'Accidents', 'Casualties', 'Vehicles'; ignores case.

Value
One of: path for one file, a message More than one file found or error if none found.

Examples

locate_one_file()
locate_one_file(filename = "Cas.csv")

phrase  
Generate a phrase for data download purposes

Description
Generate a phrase for data download purposes

Usage

phrase()
**Examples**

```
stats19:::phrase()
```

---

**police_boundaries**

*Police force boundaries in England (2016)*

---

**Description**

This dataset represents the 43 police forces in England and Wales. These are described on the [Wikipedia page](https://en.wikipedia.org/wiki/List_of_police_forces_in_England_%26_Wales) on UK police forces.

**Format**

An sf data frame

**Details**

The geographic boundary data were taken from the UK government’s official geographic data portal. See [http://geoportal.statistics.gov.uk/](http://geoportal.statistics.gov.uk/)

**Note**

These were generated using the script in the `data-raw` directory (`misc.Rmd` file) in the package’s GitHub repo: [github.com/ITSLeeds/stats19](https://github.com/ITSLeeds/stats19).

**Examples**

```
nrow(police_boundaries)
police_boundaries[police_boundaries$pfa16nm == "West Yorkshire", ]
sf:::plot.sf(police_boundaries)
```

---

**read_accidents**

*Read in STATS19 road safety data from .csv files downloaded.*

---

**Description**

Read in STATS19 road safety data from .csv files downloaded.

**Usage**

```
read_accidents(
    year = NULL,
    filename = "",
    data_dir = get_data_directory(),
    format = TRUE,
    silent = FALSE
)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Single year for which data are to be read</td>
</tr>
<tr>
<td>filename</td>
<td>Character string of the filename of the .csv to read, if this is given, type and years determine whether there is a target to read, otherwise disk scan would be needed.</td>
</tr>
<tr>
<td>data_dir</td>
<td>Where sets of downloaded data would be found.</td>
</tr>
<tr>
<td>format</td>
<td>Switch to return raw read from file, default is TRUE.</td>
</tr>
<tr>
<td>silent</td>
<td>Boolean. If FALSE (default value), display useful progress messages on the screen.</td>
</tr>
</tbody>
</table>

Details

This is a wrapper function to access and load stats 19 data in a user-friendly way. The function returns a data frame, in which each record is a reported incident in the STATS19 data.

Examples

```r
dl_stats19(year = 2011, type = "Accidents")
ac = read_accidents(year = 2011)
dl_stats19(year = 2009, type = "Accidents")
ac_2009 = read_accidents(year = 2009)
```

Description

Read in STATS19 road safety data from .csv files downloaded.

Usage

```r
read_casualties(
  year = NULL,
  filename = "",
  data_dir = get_data_directory(),
  format = TRUE
)
```
Arguments

year | Single year for which data are to be read
filename | Character string of the filename of the .csv to read, if this is given, type and years determine whether there is a target to read, otherwise disk scan would be needed.
data_dir | Where sets of downloaded data would be found.
format | Switch to return raw read from file, default is TRUE.

Details

The function returns a data frame, in which each record is a reported casualty in the STATS19 dataset.

Examples

dl_stats19(year = 2017, type = "casualties")
casualties = read_casualties(year = 2017)

Description

Read in stats19 road safety data from .csv files downloaded.

Usage

read_vehicles(
  year = NULL,
  filename = "",
  data_dir = get_data_directory(),
  format = TRUE
)

Arguments

year | Single year for which data are to be read
filename | Character string of the filename of the .csv to read, if this is given, type and years determine whether there is a target to read, otherwise disk scan would be needed.
data_dir | Where sets of downloaded data would be found.
format | Switch to return raw read from file, default is TRUE.
Details

The function returns a data frame, in which each record is a reported vehicle in the STATS19 dataset for the data_dir and filename provided.

Examples

```r
dl_stats19(year = 2009, type = "vehicles")
ve = read_vehicles(year = 2009)
```

---

**schema_original**

*Schema for stats19 data (UKDS)*

---

**Description**

Schema for stats19 data (UKDS)

**Format**

A data frame

---

**select_file**

*Interactively select from options*

---

**Description**

Interactively select from options

**Usage**

```r
select_file(fnames)
```

**Arguments**

- `fnames` File names to select from

**Examples**

```r
# fnames = c("f1", "f2")
# stats19:::select_file(fnames)
```
set_data_directory  

Description

Handy function to manage stats19 package underlying environment variable. If run interactively it makes sure user does not change directory by mistake.

Usage

set_data_directory(data_path)

Arguments

data_path valid existing path to save downloaded files in.

Examples

# set_data_directory("MY_PATH")

stats19_schema  

Description

stats19_schema and stats19_variables contain metadata on stats19 data. stats19_schema is a look-up table matching codes provided in the raw stats19 dataset with character strings.

Note

The schema data can be (re-)generated using the script in the data-raw directory.

vehicles_sample  

Description

Sample of stats19 data (2017 vehicles)

Format

A data frame
Note

These were generated using the script in the data-raw directory (misc.Rmd file).

Examples

nrow(vehicles_sample_raw)
vehicles_sample_raw
Index

*Topic datasets
  accidents_sample, 3
  casualties_sample, 3
  file_names, 6
  police_boundaries, 17
  schema_original, 20
  stats19_schema, 21
  vehicles_sample, 21

accidents_sample, 3
accidents_sample_raw (accidents_sample), 3
as.data.frame(), 13

casualties_sample, 3
casualties_sample_raw (casualties_sample), 3
check_input_file, 4
check_year, 4

data.frame, 12
dl_stats19, 5
dl_stats19(), 13

file_names, 6
file_names_old(file_names), 6
find_file_name, 6
format_accidents, 7
format_casualties, 8
format_column_names, 8
format_ppp, 9
format_ppp(), 12, 13
format_sf, 9, 10
format_sf(), 12, 13
format_vehicles, 10

get_data_directory, 11
get_MOT, 11
get_stats19, 9, 12
get_stats19(), 5
get_url, 14

locate_files, 15
locate_one_file, 16

phrase, 16
police_boundaries, 17
ppp, 9

read_accidents, 17
read_accidents(), 13
read_casualties, 18
read_vehicles, 19

schema_original, 20
select_file, 20
set_data_directory, 21
sf::sf, 12
spatstat::ppp, 12
stats19_schema, 21
stats19_variables(stats19_schema), 21

vehicles_sample, 21
vehicles_sample_raw(vehicles_sample), 21