Package ‘studentlife’

May 31, 2019

URL https://github.com/Frycast/studentlife

BugReports https://github.com/Frycast/studentlife/issues

Type Package

Title Tidy Handling and Navigation of the Student-Life Dataset

Version 1.0.0

Description Download, navigate and analyse the Student-Life dataset.
The Student-Life dataset contains passive and automatic sensing data from the phones of a class of 48 Dartmouth college students. It was collected over a 10 week term. Additionally, the dataset contains ecological momentary assessment results along with pre-study and post-study mental health surveys. The intended use is to assess mental health, academic performance and behavioral trends. The raw dataset and additional information is available at <https://studentlife.cs.dartmouth.edu/>.

Depends R (>= 3.4.0)

License GPL-3

Encoding UTF-8

LazyData true

Imports purrr (>= 0.3.2), readr (>= 1.3.1), tidyr (>= 0.8.3), dplyr (>= 0.8.0.1), jsonlite (>= 1.6), tibble (>= 2.0.1), R.utils (>= 2.8.0), skimr (>= 1.0.5), visdat (>= 0.5.3), ggplot2 (>= 3.1.1), crayon (>= 1.3.4)

RoxygenNote 6.1.1

Suggests testthat

NeedsCompilation no

Author Daniel Fryer [aut, cre] (<https://orcid.org/0000-0001-6032-0522>)

Maintainer Daniel Fryer <d.fryer@latrobe.edu.au>

Repository CRAN

Date/Publication 2019-05-31 16:20:10 UTC
add_block_labels

Description

Classify observations from an SL_tibble into block labels using available date-time information. See more information about "blocks" under the details section.

Usage

add_block_labels(tab, type = c("hour", "epoch", "day", "week", "weekday", "month", "date"), interval = "start", warning = TRUE, start_date = getOption("SL_start"), epoch_levels = getOption("SL_epoch_levels"), epoch_ubs = getOption("SL_epoch_ubs"))

Arguments

- tab: An SL_tibble as returned by the function load_SL_tibble.
- type: A character vector of block label types to include. Can be one or more of "epoch", "day", "week", "weekday", "month" and "date". Any block label types that are not inferrable from the available date-time data are ignored.
interval A character string that decides how block membership is decided when student is of class interval_sl_tibble. Can be either "start" (use start_timestamp), "end" (use end_timestamp) or "middle" (use the midpoint between start_timestamp and end_timestamp).

warning Logical. If TRUE then a warning is produced whenever a block label type is not inferable from the available date-time data.

start_date Date. The date that the StudentLife study started.

epoch_levels A character vector of epoch levels.

epoch_ubs An integer vector that defines the hour that is the upper boundary of each epoch.

Details

Block label types can be one or more of "epoch" (giving labels morning, evening, afternoon and night), "day" (giving number of days since the start_date of the StudentLife study). "week" (giving integer number of weeks since the first week of the StudentLife study, rounded down), "week-day" (giving the day of the week), "month" (giving integer number of months since the start of the StudentLife study, rounded down) and "date".

Examples

d <- tempdir()
donload_studentlife(location = d, url = "testdata")

    tab <- load_SL_tibble(
        loc = d, schema = "sensing", table = "activity", csv_nrows = 10)

b_tab <- add_block_labels(tab)
b_tab

download_studentlife
download_studentlife

Description

Download the entire StudentLife dataset or a smaller sample dataset for testing.

Usage

download_studentlife(url = "dartmouth", location = ".", unzip = TRUE, untar = TRUE)
get_EMA_questions

Arguments

url A character string. Either "dartmouth" for the Dartmouth URL, or "testdata" for a small sample dataset. Otherwise or a full URL of your choice can be specified leading to the StudentLife dataset as a .tar.gz file.

location The destination path. If the path does not exist it is created with `dir.create`

unzip Logical. If TRUE then the dataset will be unzipped with `bunzip2`. Leave as default unless you plan to do it manually.

untar Logical. If TRUE then the dataset will be untarred with `untar`. Leave as default unless you plan to do it manually.

Details

If url = "dartmouth" then data will be downloaded from <https://studentlife.cs.dartmouth.edu/dataset/dataset.tar.bz2>
If url = "testdata" then data will be downloaded from the test data at the studentlife GitHub repository <https://github.com/frycast/studentlife>

Examples

```r
# download
download_studentlife(location = d, url = "testdata")

# with menu
load_SL_tibble(location = d)

# without menu
SL_tables
load_SL_tibble(schema = "EMA", table = "PAM", location = d)
```

get_EMA_questions

Description

Get the EMA questions from a StudentLife tibble whose schema is "EMA".

Usage

`get_EMA_questions(x)`

Arguments

x A StudentLife tibble whose schema is EMA, as output by the function `load_SL_tibble`. 
get_schema

Value
The EMA_questions attribute of x

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

    tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)
    # Returns "PAM"
    get_EMA_questions(tab_PAM)

Description
Retrieve the schema name from a StudentLife tibble

Usage
get_schema(x)

Arguments

  x
An object of class StudentLife tibble (SL_tbl), as produced by the function
load_SL_tibble.

Value
A character string indicating the schema name

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

    tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)
    # Returns "EMA"
    get_schema(tab_PAM)
get_table

Description
Retrieve the table name from a StudentLife tibble

Usage
get_table(x)

Arguments
x An object of class StudentLife tibble (SL_tbl), as produced by the function load_SL_tibble.

Value
A character string indicating the table name

Examples
```
d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

# Returns "PAM"
get_table(tab_PAM)
```

is_dateless_SL_tibble

Description
Confirm that an object is a dateless StudentLife tibble

Usage
is_dateless_SL_tibble(x)

Arguments
x Any object
is_dateonly_SL_tibble

Value
Logical

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

   tab_S <- load_SL_tibble(
   schema = "survey", table = "BigFive", location = d)

   # Returns TRUE
   is_dateless_SL_tibble(tab_S)


is_dateonly_SL_tibble  is_dateonly_SL_tibble

Description
Confirm that an object is a date-only StudentLife tibble

Usage
is_dateonly_SL_tibble(x)

Arguments
x          Any object

Value
Logical

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

   tab_DL <- load_SL_tibble(
   schema = "education", table = "deadlines", location = d)

   # Returns TRUE
   is_dateonly_SL_tibble(tab_DL)
is_interval_SL_tibble

**Description**

Confirm that an object is an interval StudentLife tibble

**Usage**

`is_interval_SL_tibble(x)`

**Arguments**

- `x`  
  Any object

**Value**

Logical

**Examples**

```r
d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_con <- load_SL_tibble(
  schema = "sensing", table = "conversation", location = d, csv_nrow = 10)

# Returns TRUE
is_interval_SL_tibble(tab_con)
```

---

is_reg_SL_tibble

**Description**

Confirm that an object is a regularised StudentLife tibble

**Usage**

`is_reg_SL_tibble(x)`

**Arguments**

- `x`  
  Any object
is_SL_tibble

Value
Logical

Examples

```r
d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

reg_PAM <- regularise_time(
  tab_PAM, blocks = c("day", "epoch"), m = mean(picture_idx, na.rm = TRUE))

# Returns TRUE
is_reg_SL_tibble(reg_PAM)
```

---

### is_SL_tibble

<table>
<thead>
<tr>
<th>is_SL_tibble</th>
<th>is_SL_tibble</th>
</tr>
</thead>
</table>

### Description

Confirm that an object is a StudentLife tibble

### Usage

```r
is_SL_tibble(x)
```

### Arguments

- `x` Any object

### Value

Logical

### Examples

```r
d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

# Returns TRUE
is_SL_tibble(tab_PAM)
```
is_timestamp_SL_tibble

Description

Confirm that an object is a timestamped StudentLife tibble

Usage

is_timestamp_SL_tibble(x)

Arguments

x Any object

Value

Logical

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

# Returns TRUE
is_timestamp_SL_tibble(tab_PAM)

load_SL_tibble

Description

Import a chosen StudentLife table as a tibble. Leave schema and table unspecified to choose interactively via a menu.

Usage

load_SL_tibble(schema, table, location = ".", time_options = c("interval", "timestamp", "dateonly", "dateless"), vars, csv_nrows, datafolder = "dataset", uid_range = getOption("SL_uids"))
Arguments

schema  A character string. The menu 1 choice. Leave blank to choose interactively.
table   A character string. The menu 2 choice. Leave blank to choose interactively.
location The path to a copy of the StudentLife dataset.
time_options A character vector specifying which table types (out of "interval", "timestamp", "dateonly" and "dateless") to include in the menu. This allows you to restrict menu options according to the amount of date-time information present in the data. The default includes all data. Note this parameter only has an effect when used with the interactive menu.
vars    Character vector of variable names to import for all students. Leave blank and this will be chosen interactively if necessary. If vars contains "timestamp" then effort will be made to convert "timestamp" to appropriate variable name(s) for the target table.
csv_nrows An integer specifying the number of rows to read per student if the target is a csv. The largest files in StudentLife are csv files, so this allows code testing with less overhead.
datafolder Specifies the subfolder of location that contains the relevant data. This should normally be left as the default.
uid_range An integer vector. The range of uids in the StudentLife study.

Value

An object of class SL_tibble is returned. These inherit properties from class tibble and class data.frame. Depending on the date-time information available, the object may also be a timestamp_SL_tibble, interval_SL_tibble or dateonly_SL_tibble (which are all subclasses of SL_tibble).

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

## With menu
load_SL_tibble(location = d)

## Without menu
SL_tables
PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

## Load less data for testing with less overhead
act <- load_SL_tibble(schema = "sensing", table = "activity", location = d, csv_nrows = 10)

## Browse all tables with timestamps (non-interval)
load_SL_tibble(location = d, time_options = "timestamp")
## Browse all tables with intervals
load_SL_tibble(location = d, time_options = "interval")

## Browse all dateless tables
load_SL_tibble(location = d, time_options = "dateless")

---

**regularise_time**

---

### Description

Transform an SL_tibble (as produced by `load_SL_tibble`) in such a way that the observations are aggregated in equal length intervals called 'blocks' (for more information on blocks see `add_block_labels`).

### Usage

```r
ger.Regularise_time(tab, ..., blocks = c("epoch", "day"), add_NAs = TRUE, study_duration = getOption("SL_duration"), start_date = getOption("SL_start"), epoch_levels = getOption("SL_epoch_levels"), epoch_ubs = getOption("SL_epoch_ubs"), uid_range = getOption("SL_uids"), date_range = seq(from = start_date, by = 1, length.out = study_duration))
```

### Arguments

- **tab**
  - An SL_tibble as returned by the function `load_SL_tibble`. The SL_tibble must have some date-time information.

- **...**
  - Arguments passed to `summarise`, used to aggregate values when multiple observations are encountered in a block. Any columns not specified here or under `blocks` will be dropped.

- **blocks**
  - A character vector naming one or more of the block options "hour", "epoch", "day", "week", "weekday", "month" or "date". If not present as column names in `tab`, an attempt will be made to infer the blocks from existing time information with `add_block_labels`. The returned data.frame will have one observation (possibly NA) for each block.

- **add_NAs**
  - A logical. If TRUE then NAs will be introduced to fill missing blocks.

- **study_duration**
  - Integer. The duration of the StudentLife study in days. This parameter does nothing if `limit_date_range` is TRUE.

- **start_date**
  - Date. The date that the StudentLife study started.

- **epoch_levels**
  - A character vector of epoch labels.

- **epoch_ubs**
  - An integer vector that defines the hour that is the upper boundary of each epoch.

- **uid_range**
  - An integer vector. The range of uids in the StudentLife study.

- **date_range**
  - A vector of dates to be used if `limit_date_range` is FALSE.
Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab <- load_SL_tibble(
  loc = d, schema = "sensing", table = "activity", csv_nrows = 10)

r_tab <- regularise_time(
  tab, blocks = c("day","weekday"),
  act_inf = max(activity_inference), add_NAs = FALSE)

r_tab

response_hour_hist

Description

This function produces a histogram that visualizes the frequencies of observations within hourly
blocks, or blocks of multiple hours.

Usage

response_hour_hist(tab, break_hours = 10, xlab = "Hours into study",
  main = paste0("Distribution of ", attr(tab, "table"),
    " response times"), ...)

Arguments

tab A StudentLife tibble with time information, (i.e., and object of class timestamp_SL_tbl
  or interval_SL_tbl) as can be returned by the function load_SL_tibble.

break_hours Specify the width in hours of each histogram bin.

xlab Argument passed to hist.

main Argument passed to hist.

... Arguments passed to hist.

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

response_hour_hist(tab_PAM)
List of all the tables available in the StudentLife dataset.

Description

This command returns a 5 element list. Each of the five elements are given names corresponding to
the schema names of the studentlife data set. Each element is a vector of strings, where each string
 corresponds to the name of a table within the respective schema.

Usage

SL_tables

Format

An object of class list of length 5.

Source

https://studentlife.cs.dartmouth.edu/

Tidy Handling and Navigation of the Student-Life Dataset

Description

Download, navigate and analyse the Student-Life dataset. The Student-Life dataset contains passive
and automatic sensing data from the phones of a class of 48 de-identified Dartmouth college stu-
dents. It was collected over a 10 week term. Additionally, the dataset contains Ecological Momen-
tary Assessment results along with pre- and post-study mental health surveys, such as the PHQ-9.
The intended use is to assess mental health, academic performance and behavioral trends. The raw
dataset and additional information is available at <https://studentlife.cs.dartmouth.edu/>.

Details

Details on the Student-Life dataset as well as the dataset itself are available at https://studentlife.cs.dartmouth.edu/.

Update

Current updates are available through URL: https://github.com/frycast/studentlife

BugReports

https://github.com/frycast/studentlife/issues

Author(s)

Daniel Fryer <d.fryer@latrobe.edu.au>
Description

Produce a visualisation of the number of missing values among each student in a regularised SL_tbl.

Usage

```r
vis_NAs(tab, response, main = paste0("Missing values by student (", attr(tab, "table"), ")", show_perc_col = FALSE, ...)
```

Arguments

- `tab` A regularised StudentLife tibble (i.e., an object of class `reg_SL_tbl`) as produced by the function `regularise_time`.
- `response` A character string naming one of the columns in `tab` that is not in `attr(tab, "blocks")`. If missing then this defaults to the first such column name.
- `main` The plot title, passed to `ggtitle`.
- `show_perc_col` Logical passed to `vis_miss`. TRUE adds in the percentage of missing data in each column into the x axis.
- ... Arguments passed to `vis_miss`.

Value

A `ggplot` object.

Examples

```r
d <- tempdir()
download_studentlife(location = d, url = "testdata")

tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

reg_PAM <- regularise_time(
  tab_PAM, blocks = c("day", "epoch"), m = mean(picture_idx, na.rm = TRUE))

vis_NAs(reg_PAM, response = "m")
```
Description

Produce an ordered bar plot of the total number of responses for each student in a regularised SL_tbl.

Usage

vis_response_counts(tab, response,
  main = paste0("Total responses by student (", attr(tab, "table"), ")"),
  xlab = "Student UID", ylab = "Response count", ...)

Arguments

  tab                  A regularised StudentLife tibble (i.e., an object of class reg_SL_tbl) as pro-
                       duced by the function regularise_time.
  response             A character string naming one of the columns in tab that is not in attr(tab, "blocks").
                        If missing then this defaults to the first such column name.
  main                 The plot title, passed to barplot.
  xlab                 The x axis label, passed to barplot.
  ylab                 The y axis label, passed to barplot.
  ...                  Arguments passed to barplot.

Examples

d <- tempdir()
download_studentlife(location = d, url = "testdata")

  tab_PAM <- load_SL_tibble(schema = "EMA", table = "PAM", location = d)

  reg_PAM <- regularise_time(
    tab_PAM, blocks = c("day", "epoch"), m = mean(picture_idx, na.rm = TRUE))

  vis_response_counts(reg_PAM, response = "m")
Index

*Topic datasets
  SL_tables, 14
add_block_labels, 2, 12

barplot, 16
bunzip2, 4
data.frame, 11
dir.create, 4
download_studentlife, 3

get_EMA_questions, 4
get_schema, 5
get_table, 6
ggtitle, 15

hist, 13

is_dateless_SL_tibble, 6
is_dateonly_SL_tibble, 7
is_interval_SL_tibble, 8
is_reg_SL_tibble, 8
is_SL_tibble, 9
is_timestamp_SL_tibble, 10

load_SL_tibble, 2, 4–6, 10, 12, 13
regularise_time, 12, 15, 16
response_hour_hist, 13

SL_tables, 14
studentlife, 14
studentlife-package (studentlife), 14
summarise, 12

+tibble, 11
untar, 4

vis_miss, 15
vis_NAs, 15
vis_response_counts, 16

17