# Package ‘places’

October 14, 2022

**Type** Package  
**Title** Clusters GPS Data into Places  
**Version** 0.1.1  
**Description** Clusters GPS coordinates into places (i.e., meaningful stops). Additionally, categorizes places into types (e.g., home, cafe, gym). Places are identified as home using a rules-based algorithm defining home as the stop occurring most frequently during the night. Other places (e.g., cafe, gym) are identified using the Google Maps API.  
**License** MIT + file LICENSE  
**Encoding** UTF-8  
**LazyData** true  
**RoxygenNote** 7.1.1  
**Imports** dplyr, tidyr, sp, rgdal, geosphere, data.table, rlang, googleway, stringr, stats, hms  
**Suggests** R.rsp  
**VignetteBuilder** R.rsp  
**Depends** R (>= 2.10)  
**Language** en-US  
**NeedsCompilation** no  
**Author** Katherine Roehrick [aut, cre]  
**Maintainer** Katherine Roehrick <kr.gitcode@gmail.com>  
**Repository** CRAN  
**Date/Publication** 2021-04-08 07:50:02 UTC

## R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_clusters</td>
<td>2</td>
</tr>
<tr>
<td>get_home</td>
<td>3</td>
</tr>
<tr>
<td>get_places</td>
<td>5</td>
</tr>
<tr>
<td>places_gps</td>
<td>6</td>
</tr>
<tr>
<td>Index</td>
<td>7</td>
</tr>
</tbody>
</table>
**Description**

Use `get_clusters()` to cluster a dataframe of GPS coordinates into places.

**Usage**

```r
get_clusters(
  df,
  max.accu = 165,
  max.speed = 2.6,
  min.time = 3,
  max.time = 15,
  max.distance = 150,
  var.segment = NULL
)
```

**Arguments**

- **df**: A dataframe of GPS coordinates as described below.
- **max.accu**: An integer in meters. This number means there’s a 68% probability that the true location is within this radius. The default is 165 m. Any GPS rows with an accuracy higher than this will be dropped.
- **max.speed**: An integer in meters/sec. It is the threshold value that distinguishes a row as Static or Moving. The default is 2.6 meters/sec.
- **min.time**: An integer in minutes. It is the minimum amount of time between two points for the pair to be considered a stationary cluster. The defaults is 3 minutes.
- **max.time**: An integer in minutes. It is the maximum amount of time between two points for the pair to be considered a stationary cluster. The defaults is 15 minutes.
- **max.distance**: An integer in meters. It is the maximum distance in meters between two points for the pair to be labelled a cluster. The defaults is 150 m.
- **var.segment**: If this variable is NOT set, clusters will be created based on the participant’s entire dataset. If this variable is set, clusters will be segmented on the variable. A list can be provided.

**Value**

A list containing two named objects. **PLACES** is a dataframe of named clusters and latitude and longitude coordinates for each named cluster that was computed as a weighted average of the original GPS datapoints found within the cluster. The **PLACES** dataframe identifies moving clusters as 999999 **CLUSTERS** is a list of dataframes for each participant that contain the named clusters and coordinates for each original GPS datapoint. Unlike the **PLACES** dataframe, the **CLUSTERS** list labels "moving" clusters as NA.
Dataframe Requirements

The dataframe needs to have the following named columns:

- user_id = participant id
- lat = latitude coordinates
- lon = longitude coordinates
- start_time = time of GPS coordinates as POSIXct

The dataframe may - but does need to - have the following named columns:

- tz Olson_id = local timezone (only needed if running "get_home")
- accu = GPS accuracy. This number means there’s a 68% probability that the true location is within this radius. If this is not available, an accu column will be created and set to 0 so all rows are kept.
- speed = Speed in meters/sec at which the phone sensing data indicates an individual was moving. If this is not available, speed will be calculated as distance / time between two coordinates.

See Also

generate home to predict which cluster is an individual’s home
get_places to label each cluster’s place type as identified by Google Places API

Examples

```r
## Prepare the dataset "places_gps" and run get_clusters()
## Not run:
places_gps$time_local <- as.POSIXct(strptime(places_gps$time_local, "%m/%d/%y %H:%M"), tz="UTC")
colnames(places_gps)[c(2,4)] <- c("start_time", "lon")
clusters <- get_clusters(places_gps)
## End(Not run)
```

get_home

Predict which cluster is an individual’s home.

Description

Predict which cluster is an individual’s home.
Usage

get_home(  
  df1,
  df2,
  home.start = "00:00:00",
  home.end = "06:00:00",
  filt = TRUE,
  max.distance = 150
)

Arguments

df1 A dataframe of GPS coordinates as described below.
df2 A dataframe with named clusters (most likely the dataframe that is returned after running `reduce_multi` OR the places dataframe that is returned after running `get_clusters`).
home.start A character vector HH:MM:SS which represents the start time that most individuals will be asleep by.
home.end A character vector HH:MM:SS which represent the start time that most individual may start to wake up by.
filt A logical T or F if the GPS data should be filtered between home.start and home.end. The default is T.
max.distance An integer in meters. It is the maximum distance in meters a cluster can be from the home location to be labelled as "home". The defaults is 150 m.

Value

Returns a list of dataframes. COUNT is a dataframe that count how many times an individual was at a clusters HOME is a dataframe with clusters labelled as "Home", "Other", "In Transit"

Dataframe Requirements

The dataframe needs to have the following named columns:

- user_id = participant id
- lat = latitude coordinates
- lon = longitude coordinates
- start_time = GPS coordinates as POSIXct. Assumes POSIXct variable has been created using UTC timezone.
- tz_olson_id = local timezone (e.g., EST, America/New_York) as character vector.

See Also

- `get_clusters` to cluster GPS coordinates into places.
- `get_places` to label each cluster's place type as identified by Google Places API
Examples

```r
## Assume you have run get_clusters() on the dataset "places_gps"
## Not run:

home <- get_home(places_gps, clusters[[1]], home.start = "21:30:00", home.end = "09:30:00")
## End(Not run)
```

declare get_places

**get_places**

Label each cluster’s place type using Google Places API

**Description**

Use get_places() to return the closest place type identified by Google Places API.

**Usage**

```r
get_places(df, key = NULL, radius = 50)
```

**Arguments**

- `df` A dataframe of GPS coordinates as described below
- `key` A character vector with a Google API key. The default is NULL and must be set by the user.
- `radius` The maximum radius the Google API should search within for nearby locations. The default is 50m.

**Value**

A dataframe with clusters labelled with specific place types (defined by Google) and general categories (defined by package creator)

**Dataframe Requirements**

The dataframe needs to have the following named columns:

- lat.centroid.final = latitude coordinates
- lon.centroid.final = longitude coordinates

**See Also**

- get_clusters to cluster GPS coordinates into places.
- get_home to predict which cluster is an individual’s home
Examples

```r
## Assume you have run get_clusters() and get_home() on the dataset "places_gps"
## Not run:

## Please add your API key from Google - please be aware that this service may cost money.
key <- SET_KEY
labelled <- get_places(home[[2]], key)
## End(Not run)
```

---

**places_gps**

*GPS coordinates.*

Description

A dataset containing the GPS coordinates and other attributes for 1 hypothetical person.

Usage

```r
places_gps
```

Format

A data frame with 309 rows and 7 variables:

- `user_id` unique identifier for each participant
- `time_local` datetime of GPS coordinates
- `lat` latitude
- `lng` longitude
- `ema` survey report id
- `Response.Time` datetime of survey report
- `tz_olson_id` timezone label ...
Index

* datasets
  places_gps, 6

get_clusters, 2, 4, 5
get_home, 3, 3, 5
get_places, 3, 4, 5
places_gps, 6