

Package ‘valleybikeData’

December 28, 2020

Title ValleyBike.org Data Package

Version 0.0.1

Description All currently-available ValleyBike.org point data for 2018-2020, as well as some aggregated datasets.

Encoding UTF-8

LazyData true

Imports data.table,
dplyr,
fasttime,
fuzzyjoin,
janitor,
magrittr,
readr,
R.utils,
parallel,
stringr,
tibble

RoxygenNote 7.1.0

License MIT + file LICENSE

Depends R (>= 2.10)

R topics documented:

aggregate_trips	2
aggregate_users	2
download_files	3
get_full_data	4
get_monthly_dataset	4
import_day	5
import_month	6
monthly	6
stations	8
trips	9
users	9

Index	11
--------------	-----------

aggregate_trips	<i>aggregate_trips</i>
-----------------	------------------------

Description

Aggregate trip data.

Usage

```
aggregate_trips(full_data)
```

Arguments

`full_data` The full trajectory data (as output by `get_full_data`).

Details

Create a one-row-per-trip dataset from the output of `get_full_data`.

Value

A tibble of all available trip data.

Examples

```
## Not run:  
full_data <- get_full_data()  
trips <- aggregate_trips(full_data)  
  
## End(Not run)
```

aggregate_users	<i>aggregate_users</i>
-----------------	------------------------

Description

Aggregate user data.

Usage

```
aggregate_users(trip_data)
```

Arguments

`trip_data` The one-row-per-trip data (as output by `aggregate_trips`).

Details

Create a one-row-per-user dataset from the output of `aggregate_trips`.

Value

A tibble of all available user data.

Examples

```
## Not run:
full_data <- get_full_data()
trips <- aggregate_trips(full_data)
users <- aggregate_users(trips)

## End(Not run)
```

download_files	<i>download_files</i>
----------------	-----------------------

Description

Download raw data files

Usage

```
download_files(path, overwrite = FALSE)
```

Arguments

path	The path where to download the data files.
overwrite	Whether to overwrite the existing files at the given path. Defaults to FALSE.

Details

Download all available .csv.gz raw trajectory data files for the years 2018-2020 into a specified directory.

Examples

```
## Not run:
download_files(path = "~/Desktop/raw-data")

## End(Not run)
```

get_full_data	<i>get_full_data</i>
---------------	----------------------

Description

Get the full trajectory data (raw)

Usage

```
get_full_data()
```

Details

Get all available trajectory data for the years 2018-2020, in raw format.

Value

A 65,975,278 x 6 tibble of all available trajectory data.

Examples

```
## Not run:
full_data <- get_full_data()

## End(Not run)
```

get_monthly_dataset	<i>get_monthly_dataset</i>
---------------------	----------------------------

Description

Get the trajectory dataset for one month using numeric parameters for the month and year.

Usage

```
get_monthly_dataset(month, year)
```

Arguments

month	The month of the year for which to get the trajectory dataset (as an integer or as a string).
year	The year for which to get the trajectory dataset (as an integer or as a string).

Details

Get the trajectory dataset for one month using numeric parameters for the month and year. The month parameter can be any number from 4 to 11 (since ValleyBike doesn't run during the winter months), and the year parameter can be any valid year during which ValleyBike was active (i.e., 2018, 2019, 2020).

Value

The trajectory dataset for that specific month-year combination (as a tibble).

Examples

```
## Not run:
get_monthly_dataset(month = 7, year = 2018)

## End(Not run)
```

import_day	<i>import_day</i>
------------	-------------------

Description

Import trajectory data for one day.

Usage

```
import_day(day, return = c("clean", "anomalous", "all"), future_cutoff = 24)
```

Arguments

day	The day for which to import the data (as a string of the form "YYYY-MM-DD").
return	The type of data to return (one of "clean", "anomalous", "all"). Defaults to "clean".
future_cutoff	The next-day cutoff (in hours) past which observations are categorized as "anomalous", since rides may last past midnight. Defaults to 24.0 hours.

Details

Import trajectory data for a given day. The user can choose to import the raw data, the clean data (i.e. the raw data minus any anomalous observations), or the anomalous data.

Value

A tibble of available trajectory data for that specific day.

Examples

```
## Not run:
june_28_2018 <- import_day("2018-06-28", return = "all")

## End(Not run)
```

import_month	<i>import_month</i>
--------------	---------------------

Description

Import trajectory data for one month.

Usage

```
import_month(month, ...)
```

Arguments

month	The month for which to import the data (as a string of the form "YYYY-MM").
...	Further parameters to pass to import_day (e.g. return or future_cutoff).

Details

Import trajectory data for a specific month. The user can choose to import the raw data, the clean data (i.e. the raw data minus any anomalous observations), or the anomalous data.

Value

A tibble of available trajectory data for that specific month.

Examples

```
## Not run:
june2018 <- import_month("2018-06")

## End(Not run)
```

monthly	<i>Monthly trajectory data</i>
---------	--------------------------------

Description

The monthly datasets contain month-by-month trajectory data for all the months that ValleyBike has been in active operation (normally April-November each year). The point data (latitude, longitude) was collected during every trip, at 5-second intervals.

Usage

june2018
july2018
august2018
september2018
october2018
november2018
april2019
may2019
june2019
july2019
august2019
september2019
october2019
november2019
june2020
july2020
august2020
september2020
october2020

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 36773 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2054773 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 5802790 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 8927495 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 7331158 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2598266 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4681751 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 3379888 rows and 6 columns.
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4875254 rows and 6 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4369828 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4006793 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 3360060 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2223486 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 879494 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 435629 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2641636 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2890749 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2681825 rows and 6 columns.
 An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 361804 rows and 6 columns.

Variables

- `route_id` (character), the trip's unique route id (primary key)
- `user_id` (character), the rider's unique user id
- `bike` (character), unique bike id
- `time` (datetime), the time at which the location was recorded (down to seconds)
- `longitude` (double), the longitude of the bike at that point in time
- `latitude` (double), the latitude of the bike at that point in time

stations

ValleyBike stations (as of 2020)

Description

This dataset contains information on the 54 ValleyBike stations.

Usage

stations

Format

A tibble

Variables

- `serial_num` (integer), the station's serial number (primary key)
- `name` (character), the station's name
- `address` (character), the station's address
- `city` (character), the city in which the station is
- `latitude` (double), the station's latitude
- `longitude` (double), the station's longitude
- `docks` (integer), the number of bike docks at the station
- `display` (character), display name for the station (usually name + city)

trips*ValleyBike trips over 2018-2020*

Description

This data set is an aggregated one-row-per-trip version of the original point-in-time ValleyBike data for the years 2018, 2019, and 2020.

Usage

trips

Format

A tibble

Variables

- route_id (character), the trip's unique route id (primary key)
- user_id (character), the rider's unique user id
- bike (character), unique bike id
- start_time (datetime), the trip's starting date-time (EDT)
- end_time (datetime), the trip's ending date-time (EDT)z
- start_station (character), the trip's starting station
- start_latitude (double), the trip's starting latitude
- start_longitude (double), the trip's starting longitude
- end_station (character), the trip's ending station
- end_latitude (double), the trip's ending latitude
- end_longitude (double), the trip's ending longitude
- duration (double), the trip's duration (in seconds)

users*ValleyBike user statistics over 2018-2019*

Description

This dataset is contains anonymous statistics for ValleyBike users in 2018, 2019, and 2020.

Usage

users

Format

A tibble

Variables

- user_id (character), the user's unique id (primary key)
- trips (integer), the total number of trips taken by the user
- min_trip_duration (double), the user's minimum trip duration
- mean_trip_duration (double), the user's mean trip duration
- median_trip_duration (double), the user's median trip duration
- max_trip_duration (double), the user's maximum trip duration
- first_trip_time (datetime), the datetime of the user's first recorded trip
- last_trip_time (datetime), the datetime of the user's last recorded trip
- top_start_station (character), the station at which the user most frequently starts a trip
- top_start_station_trips (integer), the number of trips starting at the top start station
- top_end_station (character), the station at which the user most frequently ends a trip
- top_end_station_trips (integer), the number of trips ending at the top end station

Index

* datasets

- monthly, [6](#)
- stations, [8](#)
- trips, [9](#)
- users, [9](#)

- aggregate_trips, [2](#)
- aggregate_users, [2](#)
- april2019 (monthly), [6](#)
- august2018 (monthly), [6](#)
- august2019 (monthly), [6](#)
- august2020 (monthly), [6](#)

- download_files, [3](#)

- get_full_data, [4](#)
- get_monthly_dataset, [4](#)

- import_day, [5](#)
- import_month, [6](#)

- july2018 (monthly), [6](#)
- july2019 (monthly), [6](#)
- july2020 (monthly), [6](#)
- june2018 (monthly), [6](#)
- june2019 (monthly), [6](#)
- june2020 (monthly), [6](#)

- may2019 (monthly), [6](#)
- monthly, [6](#)

- november2018 (monthly), [6](#)
- november2019 (monthly), [6](#)

- october2018 (monthly), [6](#)
- october2019 (monthly), [6](#)
- october2020 (monthly), [6](#)

- september2018 (monthly), [6](#)
- september2019 (monthly), [6](#)
- september2020 (monthly), [6](#)
- stations, [8](#)

- trips, [9](#)

- users, [9](#)