Package 'stepcount'

July 23, 2025

Title Estimate Step Counts from 'Accelerometry' Data

Version 0.3.2

Description Interfaces the 'stepcount' Python module <https://github.com/OxWearables/stepcount> to estimate step counts and other activities from 'accelerometry' data.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports assertthat, curl, lubridate, magrittr, readr, reticulate

Suggests tidyr, dplyr, ggplot2, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author John Muschelli [aut, cre] (ORCID: <https://orcid.org/0000-0001-6469-1750>)

Maintainer John Muschelli <muschellij2@gmail.com>

Repository CRAN

Date/Publication 2024-10-02 17:00:02 UTC

Contents

conda_create_walking_env *Create Conda Environment for Walking*

Description

Create Conda Environment for Walking

Usage

```
conda_create_walking_env(envname = "stepcount", ...)
```

Arguments

envname	environment name
...	additional arguments to pass to reticulate::conda_create()

Value

Output of reticulate::conda_create

install_stepcount *Install the* stepcount *Python Module*

Description

Install the stepcount Python Module

Usage

```
install_stepcount(packages = "stepcount", ...)

have_stepcount()

stepcount_check()

stepcount_version()
```

Arguments

packages	packages to install. If stepcount is not included, it will be added. This package is known to work with stepcount==3.2.4
...	Additional arguments to pass to reticulate::py_install(), other than pip (pip = TRUE enforced)

Value

Output of reticulate::py_install

Examples

```
if (have_stepcount()) {
   stepcount_version()
}
```

`sc_load_model` *Load Stepcount Model*

Description

Load Stepcount Model

Usage

```
sc_load_model(
  model_type = c("ssl", "rf"),
  model_path = NULL,
  check_md5 = TRUE,
  force_download = FALSE,
  as_python = TRUE
)

sc_model_filename(model_type = c("ssl", "rf"))

sc_download_model(
  model_path,
  model_type = c("ssl", "rf"),
  check_md5 = TRUE,
  ...
)
```

Arguments

`model_type`	type of the model: either random forest (rf) or Self-Supervised Learning model (ssl)
`model_path`	the file path to the model. If on disk, this can be re-used and not re-downloaded. If `NULL`, will download to the temporary directory
`check_md5`	Do a MD5 checksum on the file
`force_download`	force a download of the model, even if the file exists
`as_python`	Keep model object as a python object
`...`	for `sc_download_model`, additional arguments to pass to `curl::curl_download()`

Value

A model from Python. `sc_download_model` returns a model file path.

`sc_model_params` *Run Stepcount Model on Data*

Description

Run Stepcount Model on Data

Usage

```
sc_model_params(model_type, pytorch_device)

stepcount(
  file,
  sample_rate = NULL,
  model_type = c("ssl", "rf"),
  model_path = NULL,
  pytorch_device = c("cpu", "cuda:0"),
  verbose = TRUE,
  keep_data = FALSE
)

stepcount_with_model(
  file,
  model_type = c("ssl", "rf"),
  model,
  sample_rate = NULL,
  pytorch_device = c("cpu", "cuda:0"),
  verbose = TRUE,
  keep_data = FALSE
)
```

Arguments

`model_type`	type of the model: either random forest (rf) or Self-Supervised Learning model (ssl)
`pytorch_device`	device to use for prediction for PyTorch.
`file`	accelerometry file to process, including CSV, CWA, GT3X, and `GENEActiv` bin files
`sample_rate`	the sample rate of the data. Set to `NULL` for `stepcount` to try to guess this
`model_path`	the file path to the model. If on disk, this can be re-used and not re-downloaded. If `NULL`, will download to the temporary directory
`verbose`	print diagnostic messages
`keep_data`	should the data used in the prediction be in the output?
`model`	A model object loaded from `sc_load_model`, but `as_python` must be `TRUE`

Value

A list of the results (data.frame), summary of the results, adjusted summary of the results, and information about the data.

Examples

```
file = system.file("extdata/P30_wrist100.csv.gz", package = "stepcount")
if (stepcount_check()) {
  out = stepcount(file = file)
  st = out$step_times
}
## Not run:
  file = system.file("extdata/P30_wrist100.csv.gz", package = "stepcount")
  df = readr::read_csv(file)
  if (stepcount_check()) {
    out = stepcount(file = df)
    st = out$step_times
  }
  if (requireNamespace("ggplot2", quietly = TRUE) &&
      requireNamespace("tidyr", quietly = TRUE) &&
      requireNamespace("dplyr", quietly = TRUE)) {
    dat = df[10000:12000,] %>%
      dplyr::select(-annotation) %>%
      tidyr::gather(axis, value, -time)
    st = st %>%
      dplyr::mutate(time = lubridate::as_datetime(time)) %>%
      dplyr::as_tibble()
    st = st %>%
      dplyr::filter(time >= min(dat$time) & time <= max(dat$time))
    dat %>%
      ggplot2::ggplot(ggplot2::aes(x = time, y = value, colour = axis)) +
      ggplot2::geom_line() +
      ggplot2::geom_vline(data = st, ggplot2::aes(xintercept = time))
  }


## End(Not run)
```

---

sc_read *Read a Data Set for* stepcount

Description

Read a Data Set for stepcount

Usage

```
sc_read(
  file,
```

```
  sample_rate = NULL,
  resample_hz = "uniform",
  verbose = TRUE,
  keep_pandas = FALSE
)
```

Arguments

file	path to the file for reading
sample_rate	the sample rate of the data. Set to NULL for stepcount to try to guess this
resample_hz	Target frequency (Hz) to resample the signal. If "uniform", use the implied frequency (use this option to fix any device sampling errors). Pass NULL to disable. Defaults to "uniform".
verbose	print diagnostic messages
keep_pandas	do not convert the data to a data.frame and keep as a pandas data.frame

Value

A list of the data and information about the data

Note

The data P30_wrist100 is from https://ora.ox.ac.uk/objects/uuid:19d3cb34-e2b3-4177-91b6-1bad0e0163e7, where we took the first 180,000 rows, the first 30 minutes of data from that participant as an example.

Examples

```
file = system.file("extdata/P30_wrist100.csv.gz", package = "stepcount")
if (stepcount_check()) {
  out = sc_read(file)
}
## Not run:
  file = system.file("extdata/P30_wrist100.csv.gz", package = "stepcount")
  if (stepcount_check()) {
    out = sc_read(file, sample_rate = 100L)
  }

## End(Not run)
```

---

sc_rename_data *Rename data for Stepcount*

Description

Rename data for Stepcount

Usage

```
sc_rename_data(data)

sc_write_csv(data, path = tempfile(fileext = ".csv"))
```

Arguments

data	a data.frame of raw accelerometry
path	path to the CSV output file

Value

A data.frame of renamed columns

---

use_stepcount_condaenv

Use Conda Environment for stepcount

Description

Use Conda Environment for stepcount

Usage

```
use_stepcount_condaenv(envname = "stepcount", ...)

conda_create_stepcount(envname = "stepcount", ..., python_version = "3.9")

unset_reticulate_python()

have_stepcount_condaenv()
```

Arguments

envname	environment name for the conda environment
...	additional arguments to pass to reticulate::use_condaenv() other than condaenv.
python_version	version of Python to use for environment

Value

Nothing

Index