Package ‘wkNNMI’

January 31, 2020

Type Package
Title A Mutual Information-Weighted k-NN Imputation Algorithm
Version 1.0.0
Date 2020-01-20
Description Implementation of an adaptive weighted k-nearest neighbours (wk-NN) imputation algorithm for clinical register data developed to explicitly handle missing values of continuous/ordinal/categorical and static/dynamic features conjointly. For each subject with missing data to be imputed, the method creates a feature vector constituted by the information collected over his/her first 'window_size' time units of visits. This vector is used as sample in a k-nearest neighbours procedure, in order to select, among the other patients, the ones with the most similar temporal evolution of the disease over time. An ad hoc similarity metric was implemented for the sample comparison, capable of handling the different nature of the data, the presence of multiple missing values and include the cross-information among features.

License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.0.2
Imports infotheo, foreach
NeedsCompilation no
Author Sebastian Daberdaku [aut, cre],
       Erica Tavazzi [aut],
       Systems Biology and Bioinformatics Group http://sysbiobig.dei.unipd.it/
       [cph]
Maintainer Sebastian Daberdaku <sebastian.daberdaku@unipd.it>
Repository CRAN
Date/Publication 2020-01-31 14:20:02 UTC

R topics documented:

   impute.subject .................................................. 2
The function performs k-Nearest Neighbours imputation weighted with Mutual Information between features.

Description

This function implements an adaptive weighted k-nearest neighbours (wk-NN) imputation algorithm for clinical register data developed to explicitly handle missing values of continuous/ordinal/categorical and static/dynamic features conjointly. For each subject with missing data to be imputed, the method creates a feature vector constituted by the information collected over his/her first \( \text{window}_\text{size} \) time units of visits. This vector is used as sample in a k-nearest neighbours procedure, in order to select, among the other patients, the ones with the most similar temporal evolution of the disease over time. An \( \text{ad hoc} \) similarity metric was implemented for the sample comparison, capable of handling the different nature of the data, the presence of multiple missing values and include the cross-information among features.

Usage

```r
impute.subject(
  subject.to.impute,
  candidates,
  method = "wknn.MI",
  window_size = 3,
  t.thresh = 1,
  cont.imp.type = "w.mean",
  ord.imp.type = "w.mean",
  static.features = NULL,
  dynamic.features = NULL,
  continuous.features = NULL,
  categorical.features = NULL,
  ordinal.features = NULL,
  time.feature,
  sub.id.feature,
  make.unique.separator = ".",
  K
)
```

Arguments

- `subject.to.impute` data frame containing the visits of the subjects with missing values to be imputed.
candidates data frame containing all the visits to be used as candidates for the imputation.

method imputation type, to be chosen between "wknn.MI", "wknn.simple" or "knn.random". Defaults to "wknn.MI".

window_size size of the time window to be imputed. Defaults to 3 (months).

t.thresh time threshold parameter. Defaults to 1 (months).

cont.imp.type imputation type for the continuous features, to be chosen between "mean", "w.mean" (weighted mean), "median" or "mode". Defaults to "w.mean".

ord.imp.type imputation type for the ordinal features, to be chosen between "mean", "w.mean" (weighted mean), "median" or "mode". Defaults to "w.mean".

static.features list of the static feature names.

dynamic.features list of the dynamic feature names.

continuous.features list of the continuous feature names.

categorical.features list of the categorical feature names.

ordinal.features list of the ordinal feature names.

time.feature name of the time feature

sub.id.feature name of the subject ID feature

make.unique.separator symbol to be used for the make unique function (must not be present in the feature names). Defaults to ".".

K number of neighbours to use. Defaults to 15.

Value

the imputed data.frame

Author(s)

Sebastian Daberduka

Examples

#' This example shows how a user can use the impute.subject() function to impute
#' the visits of a single patient by using the data from another clinical
#' register.

data(patient.data)
data(new.patient)

#' The user must define which features are static/dynamic and
#' continuous/categorical/ordinal.

static.features = c(
  "sex",
  "bmi_premorbid",
...
"bmi_diagnosis",
"fvc_diagnosis",
"familiality",
"genetics",
"ftd",
"onset_site",
"onset_age"
)

dynamic.features = c(
  "niv",
  "peg",
  "alsfrs_1",
  "alsfrs_2",
  "alsfrs_3",
  "alsfrs_4",
  "alsfrs_5",
  "alsfrs_6",
  "alsfrs_7",
  "alsfrs_8",
  "alsfrs_9",
  "alsfrs_10",
  "alsfrs_11",
  "alsfrs_12"
)

continuous.features = c("bmi_premorbid",
  "bmi_diagnosis",
  "fvc_diagnosis",
  "onset_age")

categorical.features = c("sex",
  "familiality",
  "genetics",
  "ftd",
  "onset_site",
  "niv",
  "peg")

ordinal.features = c(
  "alsfrs_1",
  "alsfrs_2",
  "alsfrs_3",
  "alsfrs_4",
  "alsfrs_5",
  "alsfrs_6",
  "alsfrs_7",
  "alsfrs_8",
  "alsfrs_9",
  "alsfrs_10",
  "alsfrs_11",
  "alsfrs_12"
)

#' In what follows, the impute.subject() function is used to impute the missing
#' values in the visits of a new patient in a 3 months wide time window.
#' Please note that missing values in the visits outside of this window will not
impute.wknn

The function performs k-Nearest Neighbours imputation weighted with Mutual Information between features.

Description

This function implements an adaptive weighted k-nearest neighbours (wk-NN) imputation algorithm for clinical register data developed to explicitly handle missing values of continuous/ordinal/categorical and static/dynamic features conjointly. For each subject with missing data to be imputed, the method creates a feature vector constituted by the information collected over his/her first *window_size* time units of visits. This vector is used as sample in a k-nearest neighbours procedure, in order to select, among the other patients, the ones with the most similar temporal evolution of the disease over time. An *ad hoc* similarity metric was implemented for the sample comparison, capable of handling the different nature of the data, the presence of multiple missing values and include the cross-information among features.

Usage

impute.wknn(
  dataset.to.impute,
  window_size = 3,
  t.thresh = 1,
  imputation.method = "wknn.MI",
  cont.imp.type = "w.mean",
  ord.imp.type = "w.mean",
  static.features,
  dynamic.features,
  continuous.features,
categorical.features,
ordinal.features,
time.feature,
sub.id.feature,
make.unique.separator = ".",
K = 15,
parallel = FALSE
)

Arguments

dataset.to.impute

data frame containing missing values.

window_size

size of the time window to be imputed. Defaults to 3 (months).

t.thresh

time threshold parameter. Defaults to 1 (months).

imputation.method

imputation type, to be chosen between "wknn.MI", "wknn.simple" or "knn.random". Defaults to "wknn.MI".

cont.imp.type

imputation type for the continuous features, to be chosen between "mean", "w.mean" (weighted mean), "median" or "mode". Defaults to "w.mean".

ord.imp.type

imputation type for the ordinal features, to be chosen between "mean", "w.mean" (weighted mean), "median" or "mode". Defaults to "w.mean".

static.features

list of the static feature names.

dynamic.features

list of the dynamic feature names.

continuous.features

list of the continuous feature names.

categorical.features

list of the categorical feature names.

ordinal.features

list of the ordinal feature names.

time.feature

name of the time feature

sub.id.feature

name of the subject ID feature

make.unique.separator

symbol to be used for the make unique function (must not be present in the feature names). Defaults to ".".

K

number of neighbours to use. Defaults to 15.

parallel

if TRUE, the iterations are performed in parallel. An appropriate parallel backed must be registered before hand, such as *doMC* or *doSNOW*. Defaults to FALSE.

Value

the imputed data.frame
impute.wknn

Author(s)

Sebastian Daberdaku

Examples

#' This example shows how a user can use the impute.wknn() function to impute an
#' instance of a clinical register composed of static and dynamic, mixed-type
#' clinical data.

data(patient.data)
#' The user must define which features are static/dynamic and
#' continuous/categorical/ordinal.
static.features = c(
  "sex",
  "bmi_premorbid",
  "bmi_diagnosis",
  "fvc_diagnosis",
  "familiality",
  "genetics",
  "ftd",
  "onset_site",
  "onset_age"
)
dynamic.features = c(
  "niv",
  "peg",
  "alsfrs_1",
  "alsfrs_2",
  "alsfrs_3",
  "alsfrs_4",
  "alsfrs_5",
  "alsfrs_6",
  "alsfrs_7",
  "alsfrs_8",
  "alsfrs_9",
  "alsfrs_10",
  "alsfrs_11",
  "alsfrs_12"
)
continuous.features = c("bmi_premorbid",
  "bmi_diagnosis",
  "fvc_diagnosis",
  "onset_age")
categorical.features = c("sex",
  "familiality",
  "genetics",
  "ftd",
  "onset_site",
  "niv",
  "peg")
ordinal.features = c(
  "alsfrs_1",

"alsfrs_2",
"alsfrs_3",
"alsfrs_4",
"alsfrs_5",
"alsfrs_6",
"alsfrs_7",
"alsfrs_8",
"alsfrs_9",
"alsfrs_10",
"alsfrs_11",
"alsfrs_12"
)

# In what follows, the impute.wknn() function is used to impute the missing
# values in the patient.data dataset in a 3 months wide time window.
# Please note that missing values in the visits outside of this window will not
# be imputed.
imputed.patient.data <-
impute.wknn(
    dataset.to.impute = patient.data,
    # dataset to impute
    window_size = 3,
    # how many months of patient data to impute
    K = 5,
    # number of neighbours to consider for the imputation
    static.features = static.features,
    dynamic.features = dynamic.features,
    continuous.features = continuous.features,
    categorical.features = categorical.features,
    ordinal.features = ordinal.features,
    time.feature = "visit_time",
    # the time feature
    sub.id.feature = "subID",
    parallel = FALSE
)

---

**new.patient**

*Example dataset containing 2 visits of a hypothetical patient with amyotrophic lateral sclerosis (ALS).*

**Description**

Example dataset containing 2 visits of a hypothetical patient with amyotrophic lateral sclerosis (ALS).

**Usage**

`data(new.patient)`
A data frame with 2 rows and 25 variables:

**subID**  patient’s ID  
**sex**  patient’s sex  
**bmi_premorbid**  premorbid body mass index  
**bmi_diagnosis**  body mass index at disease diagnosis  
**fvc_diagnosis**  forced vital capacity at disease diagnosis (a measure of respiratory functionality)  
**familiality**  familiality of ALS  
**genetics**  the result of a genetic screening over the most common ALS-associated genes  
**ftd**  presence of frontotemporal dementia  
**onset_site**  site of disease onset (limb/bulbar)  
**onset_age**  age at disease onset  
**visit_time**  month in which the current visit took place; the months start from 0  
**niv**  the presence/absence up to the current visit of non-invasive ventilation  
**peg**  the presence/absence up to the current visit of percutaneous endoscopic gastrostomy  
**alsfrs_1**  Item 1 (SPEECH) of the revised ALS Functional Rating Scale (ALSFRS-R): a 12-item questionnaire rated on a 0–4 point scale evaluating the observable functional status and change for patients with ALS over time  
**alsfrs_2**  Item 2 (SALIVATION) of the ALSFRS-R  
**alsfrs_3**  Item 3 (SWALLOWING) of the ALSFRS-R  
**alsfrs_4**  Item 4 (HANDWRITING) of the ALSFRS-R  
**alsfrs_5**  Item 5 (CUTTING FOOD AND HANDLING UTENSILS) of the ALSFRS-R  
**alsfrs_6**  Item 6 (DRESSING AND HYGIENE) of the ALSFRS-R  
**alsfrs_7**  Item 7 (TURNING IN BED AND ADJUSTING BED CLOTHES) of the ALSFRS-R  
**alsfrs_8**  Item 8 (WALKING) of the ALSFRS-R  
**alsfrs_9**  Item 9 (CLIMBING STAIRS) of the ALSFRS-R  
**alsfrs_10**  Item 10 (DYSPNEA) of the ALSFRS-R  
**alsfrs_11**  Item 11 (ORTHOPNEA) of the ALSFRS-R  
**alsfrs_12**  Item 12 (RESPIRATORY INSUFFICIENCY) of the ALSFRS-R
Example dataset containing 89 visits of 11 hypothetical patients with amyotrophic lateral sclerosis (ALS).
Description

This package implements an adaptive weighted k-nearest neighbours (wk-NN) imputation algorithm for clinical register data developed to explicitly handle missing values of continuous/ordinal/categorical and static/dynamic features conjointly. For each subject with missing data to be imputed, the method creates a feature vector constituted by the information collected over his/her first *window_size* time units of visits. This vector is used as sample in a k-nearest neighbours procedure, in order to select, among the other patients, the ones with the most similar temporal evolution of the disease over time. An *ad hoc* similarity metric was implemented for the sample comparison, capable of handling the different nature of the data, the presence of multiple missing values and include the cross-information among features.

Details

The wkNNMI package mainly serves as container for the two functions that implement the imputation algorithm impute.subject() and impute.wknn(), and for the example datasets patient.data and new.patient.
Index

*Topic datasets
  new.patient, 8
  patient.data, 10

impute.subject, 2
impute.wknn, 5

new.patient, 8
patient.data, 10
wkNNMI, 11