

Package ‘openVA’

February 3, 2021

Type Package

Title Automated Method for Verbal Autopsy

Version 1.0.12

Date 2021-02-02

Author Zehang Richard Li, Tyler McCormick, Sam Clark

Maintainer Zehang Richard Li <lizehang@gmail.com>

Depends R (>= 3.1)

Imports InterVA5 (>= 1.0.1), InSilicoVA (>= 1.1.3), InterVA4 (>= 1.7.3), Tariff (>= 1.0.1), ggplot2, crayon, cli, methods

Suggests nbc4va, testthat

Description Implements multiple existing open-source algorithms for coding cause of death from verbal autopsies. The methods implemented include 'InterVA4' by Byass et al (2012) <doi:10.3402/gha.v5i0.19281>, 'InterVA5' by Byass et al (2019) <doi:10.1186/s12916-019-1333-6>, 'InSilicoVA' by McCormick et al (2016) <doi:10.1080/01621459.2016.1152191>, 'NBC' by Miasnikof et al (2015) <doi:10.1186/s12916-015-0521-2>, and a replication of 'Tariff' method by James et al (2011) <doi:10.1186/1478-7954-9-31> and Serina, et al. (2015) <doi:10.1186/s12916-015-0527-9>. It also provides tools for data manipulation tasks commonly used in Verbal Autopsy analysis and implements easy graphical visualization of individual and population level statistics. The 'NBC' method is implemented by the 'nbc4va' package that can be installed from <<https://github.com/rrwen/nbc4va>>. Note that this package was not developed by authors affiliated with the Institute for Health Metrics and Evaluation and thus unintentional discrepancies may exist in the implementation of the 'Tariff' method.

License GPL-2

URL <https://github.com/verbal-autopsy-software/openVA>

BugReports <https://github.com/verbal-autopsy-software/openVA/issues>

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-02-03 16:40:08 UTC

R topics documented:

codeVA	2
ConvertData	5
ConvertData.phmrc	6
getCSMF	7
getCSMF_accuracy	8
getIndivProb	9
getPHMRC_url	10
getTopCOD	10
interVA.train	11
openVA_status	12
openVA_update	13
plotVA	13
stackplotVA	14
Index	17

codeVA

Running automated method on VA data

Description

Running automated method on VA data

Usage

```
codeVA(
  data,
  data.type = c("WHO2012", "WHO2016", "PHMRC", "customize")[1],
  data.train = NULL,
  causes.train = NULL,
  causes.table = NULL,
  model = c("InSilicoVA", "InterVA", "Tariff", "NBC")[1],
  Nchain = 1,
  Nsim = 10000,
  version = c("4.02", "4.03", "5")[2],
  HIV = "h",
  Malaria = "h",
  phmrc.type = c("adult", "child", "neonate")[1],
  convert.type = c("quantile", "fixed", "empirical")[1],
  ...
)
```

Arguments

<code>data</code>	Input VA data, see <code>data.type</code> below for more information about the format.
<code>data.type</code>	There are four data input types currently supported by <code>codeVA</code> function as below. <ul style="list-style-type: none"> • WHO2012: InterVA-4 input format using WHO 2012 questionnaire. For example see <code>data(RandomVA1)</code>. The first column should be death ID. • WHO2016: InterVA-5 input format using WHO 2016 questionnaire. For example see <code>data(RandomVA5)</code>. The first column should be death ID. • PHMRC: PHMRC data format. For example see ConvertData.phmrc • customized: Any dichotomized dataset with “Y” denote “presence”, “” denote “absence”, and “.” denote “missing”. The first column should be death ID.
<code>data.train</code>	Training data with the same columns as <code>data</code> , except for an additional column specifying cause-of-death label. It is not used if <code>data.type</code> is “WHO” and <code>model</code> is “InterVA” or “InSilicoVA”. The first column also has to be death ID for “WHO” and “customized” types.
<code>causes.train</code>	the column name of the cause-of-death assignment label in training data.
<code>causes.table</code>	list of causes to consider in the training data. Default to be NULL, which uses all the causes present in the training data.
<code>model</code>	Currently supports four models: “InSilicoVA”, “InterVA”, “Tariff”, and “NBC”.
<code>Nchain</code>	Parameter specific to “InSilicoVA” model. Currently not used.
<code>Nsim</code>	Parameter specific to “InSilicoVA” model. Number of iterations to run the sampler.
<code>version</code>	Parameter specific to “InterVA” model. Currently supports “4.02”, “4.03”, and “5”. For InterVA-4, “4.03” is strongly recommended as it fixes several major bugs in “4.02” version. “4.02” is only included for backward compatibility. “5” version implements the InterVA-5 model, which requires different data input format.
<code>HIV</code>	Parameter specific to “InterVA” model. HIV prevalence level, can take values “h” (high), “l” (low), and “v” (very low).
<code>Malaria</code>	HIV Parameter specific to “InterVA” model. Malaria prevalence level, can take values “h” (high), “l” (low), and “v” (very low).
<code>phmrc.type</code>	Which PHMRC data format is used. Currently supports only “adult” and “child”, “neonate” will be supported in the next release.
<code>convert.type</code>	type of data conversion when calculating conditional probability (probability of each symptom given each cause of death) for InterVA and InSilicoVA models. Both “quantile” and “fixed” usually give similar results empirically. <ul style="list-style-type: none"> • <code>quantile</code>: the rankings of the P(SIC) are obtained by matching the same quantile distributions in the default InterVA P(SIC) • <code>fixed</code>: P(SIC) are matched to the closest values in the default InterVA P(SIC) table. • <code>empirical</code>: no ranking is calculated, but use the empirical conditional probabilities directly, which will force <code>updateCondProb</code> to be FALSE for InSilicoVA algorithm.

... other arguments passed to [insilico](#), [InterVA](#), [interVA.train](#), [tariff](#), and [nbc](#) function in the `nbc4va` package. See respective package documents for details.

Value

a fitted object

References

Tyler H. McCormick, Zehang R. Li, Clara Calvert, Amelia C. Crampin, Kathleen Kahn and Samuel J. Clark (2016) *Probabilistic cause-of-death assignment using verbal autopsies*. <https://arxiv.org/abs/1411.3042>, *Journal of the American Statistical Association*

James, S. L., Flaxman, A. D., Murray, C. J., & Population Health Metrics Research Consortium. (2011). *Performance of the Tariff Method: validation of a simple additive algorithm for analysis of verbal autopsies*. *Population Health Metrics*, 9(1), 1-16.

Zehang R. Li, Tyler H. McCormick, Samuel J. Clark (2014) *InterVA4: An R package to analyze verbal autopsy data*. *Center for Statistics and the Social Sciences Working Paper, No.146*

<http://www.interva.net/>

Miasnikof P, Giannakeas V, Gomes M, Aleksandrowicz L, Shestopaloff AY, Alam D, Tollman S, Samarikhalaj, Jha P. *Naive Bayes classifiers for verbal autopsies: comparison to physician-based classification for 21,000 child and adult deaths*. *BMC Medicine*. 2015;13:286.

See Also

[insilico](#), [InterVA](#), [interVA.train](#), [tariff](#), and [nbc](#) function in the `nbc4va` package.

Examples

```
data(RandomVA3)
test <- RandomVA3[1:200, ]
train <- RandomVA3[201:400, ]
fit1 <- codeVA(data = test, data.type = "customize", model = "InSilicoVA",
              data.train = train, causes.train = "cause",
              Nsim=1000, auto.length = FALSE)

fit2 <- codeVA(data = test, data.type = "customize", model = "InterVA",
              data.train = train, causes.train = "cause", write=FALSE,
              version = "4.02", HIV = "h", Malaria = "1")

fit3 <- codeVA(data = test, data.type = "customize", model = "Tariff",
              data.train = train, causes.train = "cause",
              nboot.sig = 100)
```

 ConvertData

Converting Input data with different coding scheme to standard format

Description

Converting Input data with different coding scheme to standard format

Usage

```
ConvertData(
  input,
  yesLabel = NULL,
  noLabel = NULL,
  missLabel = NULL,
  data.type = c("WHO2012", "WHO2016")[1]
)
```

Arguments

input	matrix input, the first column is ID, the rest of the columns each represent one symptom
yesLabel	The value(s) coding "Yes" in the input matrix.
noLabel	The value(s) coding "No" in the input matrix.
missLabel	The value(s) coding "Missing" in the input matrix.
data.type	The coding scheme of the output. This can be either "WHO2012" or "WHO2016".

Value

a data frame coded as follows. For WHO2012 scheme: "Y" for yes, "" for No, and "." for missing. For WHO2016 scheme: "y" for yes, "n" for No, and "-" for missing.

Examples

```
# make up a fake 2 by 3 dataset with 2 deaths and 3 symptoms
id <- c("d1", "d2")
x <- matrix(c("Yes", "No", "Don't know",
  "Yes", "Refused to answer", "No"),
  byrow = TRUE, nrow = 2, ncol = 3)
x <- cbind(id, x)
colnames(x) <- c("ID", "S1", "S2", "S3")
# see possible raw data (or existing data created for other purpose)
x
new <- ConvertData(x, yesLabel = "Yes", noLabel = "No",
  missLabel = c("Don't know", "Refused to answer"))
new
```

ConvertData.phmrc *Convert standard PHMRC data into binary indicator format*

Description

The PHMRC data and the description of the format could be found at <https://ghdx.healthdata.org/record/ihme-data/population-health-metrics-research-consortium-gold-standard-verbal-autopsy-da>. This function convert the symptoms into binary indicators of three levels: Yes, No, and Missing. The health care experience (HCE) and free-text columns, i.e., columns named "word_****", are not considered in the current version of data conversion.

Usage

```
ConvertData.phmrc(
  input,
  input.test = NULL,
  cause = NULL,
  phmrc.type = c("adult", "child", "neonate")[1],
  cutoff = c("default", "adapt")[1],
  ...
)
```

Arguments

input	standard PHMRC data format
input.test	standard PHMRC data format to be transformed in the same way as input
cause	the column name for the cause-of-death variable to use. For example, "va34", "va46", or "va55". It is used if adaptive cut-offs are to be calculated for continuous variables. See below for details.
phmrc.type	which data input format it is. The three data formats currently available are "adult", "child", and "neonate".
cutoff	This determines how the cut-off values are to be set for continuous variables. "default" sets the cut-off values proposed in the original paper published with the dataset. "adapt" sets the cut-off values using the rules described in the original paper, which calculates the cut-off as being two median absolute deviations above the median of the mean durations across causes. However, we are not able to replicate the default cut-offs following this rule. So we suggest users to use this feature with caution.
...	not used

Value

converted dataset with only ID and binary symptoms. Notice that when applying this function to the raw PHMRC data, the returned ID variable corresponds to the row index of the raw PHMRC data (i.e., cleaned data with ID = 10 correspond to the 10th row of the raw dataset), and does not correspond to the "newid" column in the PHMRC data.

References

James, S. L., Flaxman, A. D., Murray, C. J., & Population Health Metrics Research Consortium. (2011). *Performance of the Tariff Method: validation of a simple additive algorithm for analysis of verbal autopsies*. *Population Health Metrics*, 9(1), 1-16.

Examples

```
# read the raw data files from PHMRC website
# notice reading directly from internet could be time consuming
# so we only read 100 rows here.
# in practice, it is much easier and faster to download the file first,
# and read all at once.
raw <- read.csv(getPHMRC_url("adult"), nrows = 100)
head(raw[, 1:20])
# default way of conversion
clean <- ConvertData.phmrc(raw, phmrc.type = "adult")
head(clean$output[, 1:20])
# using cut-offs calculated from the data (caution)
clean2 <- ConvertData.phmrc(raw, phmrc.type = "adult",
cause = "va55", cutoff = "adapt")
head(clean2$output[, 1:20])

# Now using the first 100 rows of data as training dataset
# And the next 100 as testing dataset
test <- read.csv(getPHMRC_url("adult"), nrows = 200)
test <- test[-(1:100), ]

# For the default transformation it does matter
clean <- ConvertData.phmrc(raw, test, phmrc.type = "adult")
head(clean$output[, 1:20])
head(clean$output.test[, 1:20])
# For adaptive transformation, need to make sure both files use the same cutoff
clean2 <- ConvertData.phmrc(raw, test, phmrc.type = "adult",
cause = "va55", cutoff = "adapt")
head(clean2$output[, 1:20])
head(clean2$output.test[, 1:20])
```

getCSMF

Obtain CSMF from fitted model

Description

Obtain CSMF from fitted model

Usage

```
getCSMF(x, CI = 0.95, interVA.rule = TRUE)
```

Arguments

x	a fitted object from codeVA.
CI	For insilico object only, specifying the credible interval to return. Default value to be 0.95.
interVA.rule	Logical indicator for interVA object only. If TRUE, it means only up to top 3 causes for each death are used to calculate CSMF and the rest are categorized as "undetermined"

Value

a vector or matrix of CSMF for all causes.

Examples

```
## Not run:
library(InSilicoVA)
data(RandomVA1)
# for illustration, only use interVA on 100 deaths
fit <- codeVA(RandomVA1[1:100, ], data.type = "WHO2012", model = "InterVA",
              version = "4.03", HIV = "h", Malaria = "l", write=FALSE)

getCSMF(fit)
library(InterVA5)
data(RandomVA5)
fit <- codeVA(RandomVA5[1:100, ], data.type = "WHO2016", model = "InterVA",
              version = "5", HIV = "h", Malaria = "l", write=FALSE)

getCSMF(fit)

## End(Not run)
```

getCSMF_accuracy	<i>Calculate CSMF accuracy</i>
------------------	--------------------------------

Description

Calculate CSMF accuracy

Usage

```
getCSMF_accuracy(csmf, truth, undet = NULL)
```

Arguments

csmf	a CSMF vector from getCSMF or a InSilicoVA fitted object.
truth	a CSMF vector of the true CSMF.
undet	name of the category denoting undetermined causes. Default to be NULL.

Value

a number (or vector if input is InSilicoVA fitted object) of CSMF accuracy as $1 - \text{sum}(\text{abs}(\text{CSMF} - \text{CSMF_true})) / (2 * (1 - \min(\text{CSMF_true})))$.

Examples

```
csmf1 <- c(0.2, 0.3, 0.5)
csmf0 <- c(0.3, 0.3, 0.4)
acc <- getCSMF_accuracy(csmf1, csmf0)
```

<code>getIndivProb</code>	<i>Extract individual distribution of cause of death</i>
---------------------------	--

Description

Extract individual distribution of cause of death

Usage

```
getIndivProb(x, CI = NULL, ...)
```

Arguments

<code>x</code>	a fitted object from codeVA.
<code>CI</code>	Credible interval for posterior estimates. If CI is set to TRUE, a list is returned instead of a data frame.
<code>...</code>	additional arguments that can be passed to <code>get.indiv</code> from InSilicoVA package.

Value

a data frame of COD distribution for each individual specified by row names.

Examples

```
data(RandomVA1)
# for illustration, only use interVA on 100 deaths
fit <- codeVA(RandomVA1[1:100, ], data.type = "WHO", model = "InterVA",
              version = "4.02", HIV = "h", Malaria = "l", write=FALSE)
probs <- getIndivProb(fit)
```

getPHMRC_url	<i>Get the URL to the PHMRC dataset</i>
--------------	---

Description

Get the URL to the PHMRC dataset

Usage

```
getPHMRC_url(type)
```

Arguments

type adult, child, or neonate

Value

URL of the corresponding dataset

Examples

```
link <- getPHMRC_url("adult")
summary(link)$description
```

getTopCOD	<i>Extract the most likely cause of death</i>
-----------	---

Description

Extract the most likely cause of death

Usage

```
getTopCOD(x, interVA.rule = TRUE)
```

Arguments

x a fitted object from codeVA.
interVA.rule Logical indicator for interVA object only. If TRUE, only the InterVA reported first cause is extracted.

Value

a data frame of ID and most likely cause assignment.

Examples

```

data(RandomVA1)
# for illustration, only use interVA on 100 deaths
fit <- codeVA(RandomVA1[1:100, ], data.type = "WHO", model = "InterVA",
              version = "4.02", HIV = "h", Malaria = "1", write=FALSE)
getTopCOD(fit)

```

interVA.train

*Extended InterVA method for non-standard input***Description**

Extended InterVA method for non-standard input

Usage

```

interVA.train(
  data,
  train,
  causes.train,
  causes.table = NULL,
  thre = 0.95,
  type = c("quantile", "fixed", "empirical")[1],
  prior = c("uniform", "train")[1],
  ...
)

```

Arguments

data	A matrix input, or data read from csv files. Sample input is included as data(RandomVA3).
train	A matrix input, or data read from csv files in the same format as data, but with an additional column specifying cause-of-death. Sample input is included as data(RandomVA3).
causes.train	the column name of the cause-of-death assignment label in training data.
causes.table	list of causes to consider in the training data. Default to be NULL, which uses all the causes present in the training data.
thre	numerical number between 0 and 1. Symptoms with missing rate higher than thre in the training data will be dropped from both training and testing data.
type	type of data conversion when calculating conditional probability (probability of each symptom given each cause of death) for InterVA and InSilicoVA models. Both “quantile” and “fixed” usually give similar results empirically. <ul style="list-style-type: none"> quantile: the rankings of the P(SIC) are obtained by matching the same quantile distributions in the default InterVA P(SIC)

	<ul style="list-style-type: none"> • fixed: P(SIC) are matched to the closest values in the default InterVA P(SIC) table. • empirical: no ranking is calculated, but use the empirical conditional probabilities directly.
prior	The prior distribution of CSMF. “uniform” uses no prior information, i.e., 1/C for all C causes and “train” uses the CSMF in the training data as prior distribution of CSMF.
...	not used

Value

fitted interVA object

References

Tyler H. McCormick, Zehang R. Li, Clara Calvert, Amelia C. Crampin, Kathleen Kahn and Samuel J. Clark (2016) *Probabilistic cause-of-death assignment using verbal autopsies*. <https://arxiv.org/abs/1411.3042>, To appear, *Journal of the American Statistical Association*

Zehang R. Li, Tyler H. McCormick, Samuel J. Clark (2014) *InterVA4: An R package to analyze verbal autopsy data.*, *Center for Statistics and the Social Sciences Working Paper, No.146*

<http://www.interva.net/>

Examples

```
data(RandomVA3)
test <- RandomVA3[1:200, ]
train <- RandomVA3[201:400, ]
out <- interVA.train(data = test, train = train, causes.train = "cause",
                    prior = "train", type = "quantile")
```

openVA_status

Check openVA packages status

Description

This will print the current versions of all openVA packages (and optionally, their dependencies) are up-to-date, and will install after an interactive confirmation.

Usage

```
openVA_status()
```

Examples

```
## Not run:
openVA_status()

## End(Not run)
```

openVA_update	<i>Update openVA packages</i>
---------------	-------------------------------

Description

This will check to see if all openVA packages (and optionally, their dependencies) are up-to-date, and will install after an interactive confirmation.

Usage

```
openVA_update()
```

Examples

```
## Not run:
openVA_update()

## End(Not run)
```

plotVA	<i>Plot top CSMF for a fitted model</i>
--------	---

Description

Plot top CSMF for a fitted model

Usage

```
plotVA(object, top = 10, title = NULL, ...)
```

Arguments

object	a fitted object using codeVA
top	number of top causes to plot
title	title of the plot
...	additional arguments passed to plot.insilico , plot.tariff , CSMF , or plot.nbc function in the nbc4va package.

See Also

[plot.insilico](#), [plot.tariff](#), [CSMF](#)

Examples

```
data(RandomVA3)
test <- RandomVA3[1:200, ]
train <- RandomVA3[201:400, ]
fit1 <- codeVA(data = test, data.type = "customize", model = "InSilicoVA",
              data.train = train, causes.train = "cause",
              Nsim=1000, auto.length = FALSE)

fit2 <- codeVA(data = test, data.type = "customize", model = "InterVA",
              data.train = train, causes.train = "cause",
              version = "4.02", HIV = "h", Malaria = "l")

fit3 <- codeVA(data = test, data.type = "customize", model = "Tariff",
              data.train = train, causes.train = "cause",
              nboot.sig = 100)

plotVA(fit1)
plotVA(fit2)
plotVA(fit3)
```

stackplotVA

plot grouped CSMF from a "insilico" object

Description

Produce bar plot of the CSMFs for a fitted "insilico" object in broader groups.

Usage

```
stackplotVA(
  x,
  grouping = NULL,
  type = c("stack", "dodge")[1],
  group_order = NULL,
  err = TRUE,
  CI = 0.95,
  sample_size_print = FALSE,
  xlab = "",
  ylab = "CSMF",
  ylim = NULL,
  title = "CSMF by broader cause categories",
  horiz = FALSE,
```

```

    angle = 0,
    err_width = 0.4,
    err_size = 0.6,
    border = "black",
    bw = FALSE,
    filter_legend = FALSE,
    ...
)

```

Arguments

<code>x</code>	one or a list of fitted object from codeVA function
<code>grouping</code>	C by 2 matrix of grouping rule. If set to NULL, make it default.
<code>type</code>	type of the plot to make
<code>group_order</code>	list of grouped categories. If set to NULL, make it default.
<code>err</code>	indicator of inclusion of error bars
<code>CI</code>	Level of posterior credible intervals.
<code>sample_size_print</code>	Logical indicator for printing also the sample size for each sub-population labels.
<code>xlab</code>	Labels for the causes.
<code>ylab</code>	Labels for the CSMF values.
<code>ylim</code>	Range of y-axis.
<code>title</code>	Title of the plot.
<code>horiz</code>	Logical indicator indicating if the bars are plotted horizontally.
<code>angle</code>	Angle of rotation for the texts on x axis when <code>horiz</code> is set to FALSE
<code>err_width</code>	Size of the error bars.
<code>err_size</code>	Thickness of the error bar lines.
<code>border</code>	The color for the border of the bars.
<code>bw</code>	Logical indicator for setting the theme of the plots to be black and white.
<code>filter_legend</code>	Logical indicator for including all broad causes in the plot legend (default; FALSE) or filtering to only the broad causes in the data being plotted
<code>...</code>	Not used.

Author(s)

Zehang Li, Tyler McCormick, Sam Clark

Maintainer: Zehang Li <lizehang@uw.edu>

Examples

```
data(RandomVA3)
test <- RandomVA3[1:200, ]
train <- RandomVA3[201:400, ]
fit1 <- codeVA(data = test, data.type = "customize", model = "InSilicoVA",
               data.train = train, causes.train = "cause",
               Nsim=1000, auto.length = FALSE)

fit2 <- codeVA(data = test, data.type = "customize", model = "InterVA",
               data.train = train, causes.train = "cause", write=FALSE,
               version = "4.02", HIV = "h", Malaria = "l")

fit3 <- codeVA(data = test, data.type = "customize", model = "Tariff",
               data.train = train, causes.train = "cause",
               nboot.sig = 100)

data(SampleCategory)
stackplotVA(fit1, grouping = SampleCategory, type ="dodge",
            ylim = c(0, 1), title = "InSilicoVA")
stackplotVA(fit2, grouping = SampleCategory, type = "dodge",
            ylim = c(0, 1), title = "InterVA4.02")
stackplotVA(fit3, grouping = SampleCategory, type = "dodge",
            ylim = c(0, 1), title = "Tariff")
```


Index

- * **InSilicoVA**
 - codeVA, 2
- * **InterVA4**
 - codeVA, 2
 - interVA.train, 11
- * **NBC4VA**
 - codeVA, 2
- * **Tariff**
 - codeVA, 2

codeVA, 2, 13
ConvertData, 5
ConvertData.phmrc, 3, 6
CSMF, 13, 14

getCSMF, 7
getCSMF_accuracy, 8
getIndivProb, 9
getPHMRC_url, 10
getTopCOD, 10

insilico, 4
InterVA, 4
interVA.train, 4, 11

openVA_status, 12
openVA_update, 13

plot.insilico, 13, 14
plot.tariff, 13, 14
plotVA, 13

stackplotVA, 14

tariff, 4