Package: openVA (via r-universe)

October 18, 2026

Type Package

Title Automated Method for Verbal Autopsy

Version 1.2.0

Date 2025-09-23

Depends R (>= 3.5.0)

Encoding UTF-8

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

Suggests covr, EAVA, knitr, nbc4va, R.rsp, testthat, vacalibration

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 at 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 7.3.2

VignetteBuilder R.rsp, knitr

NeedsCompilation no

Config/build/clean-inst-doc FALSE

Author Zehang Richard Li [aut, cre], Jason Thomas [aut], Tyler H. McCormick [aut], Samuel J. Clark [aut]

Maintainer Zehang Richard Li <lizehang@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2025-09-24 05:33:53 UTC

RemoteUrl https://github.com/cran/openVA

RemoteRef HEAD

RemoteSha 41440b5b438a8cb2038a616199577c04155efe25

Contents

codeVA . . . 3
ConvertData . . . 5
ConvertData.phmrc . . . 7
csmf_eava . . . 8
DataEAVA . . . 9
getCCC . . . 9
getCSMF . . . 10
getCSMF_accuracy . . . 11
getIndivProb . . . 12
getPHMRC_url . . . 13
getTopCOD . . . 13
grouping_eava_child . . . 15
grouping_eava_neonate . . . 16
interVA_train . . . 16
NeonatesVA5 . . . 18
openVA_status . . . 18
openVA_update . . . 19
plot.eava . . . 19
plot.vacalibration . . . 20
plotVA . . . 22
prepCalibration . . . 23
print.eava . . . 24
print.eava_summary . . . 25
print.vacalibration . . . 25
print.vacalibration_summary . . . 26
RandomVA6 . . . 27
stackplotVA . . . 27
summary.eava . . . 29
summary.vacalibration . . . 30

Index **32**

codeVA *Running automated method on VA data*

Description

Running automated method on VA data

Usage

```
codeVA(
  data,
  data.type = c("WHO2012", "WHO2016", "PHMRC", "EAVA", "customize")[2],
  data.train = NULL,
  causes.train = NULL,
  causes.table = NULL,
  model = c("InSilicoVA", "InterVA", "Tariff", "NBC", "EAVA")[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],
  age_group = c("neonate", "child")[1],
  ...
)
```

Arguments

`data`	Input VA data, see `data.type` below for more information about the format.
`data.type`	There are five data input types currently supported by `codeVA` function as below. • `WHO2012`: InterVA-4 input format using WHO 2012 questionnaire. For example see `data(RandomVA1)`. The first column should be death ID. • `WHO2016`: InterVA-5 input format using WHO 2016 questionnaire. For example see `data(RandomVA5)`. The first column should be death ID. • `PHMRC`: PHMRC data format. The raw PHMRC long format data will be processed internally following the steps described in McCormick et al. (2016). For example see `ConvertData.phmrc` • `EAVA`: EAVA data format using WHO 2016 questionnaire, as produced by [EAVA::odk2EAVA()]. • `customized`: Any dichotomized dataset with "Y" denote "presence", "" denote "absence", and "." denote "missing". The first column should be death ID.
`data.train`	Training data with the same columns as `data`, except for an additional column specifying cause-of-death label. It is not used if `data.type` is "WHO" and

	model is "InterVA" or "InSilicoVA". The first column also has to be death ID for "WHO" and "customized" types.
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.
model	Currently supports five models: "InSilicoVA", "InterVA", "Tariff", "NBC", and "EAVA".
Nchain	Parameter specific to "InSilicoVA" model. Currently not used.
Nsim	Parameter specific to "InSilicoVA" model. Number of iterations to run the sampler.
version	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.
HIV	Parameter specific to "InterVA" model. HIV prevalence level, can take values "h" (high), "l" (low), and "v" (very low).
Malaria	HIV Parameter specific to "InterVA" model. Malaria prevalence level, can take values "h" (high), "l" (low), and "v" (very low).
phmrc.type	Which PHMRC data format is used. Currently supports only "adult" and "child", "neonate" will be supported in the next release.
convert.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. • quantile: the rankings of the P(S|C) are obtained by matching the same quantile distributions in the default InterVA P(S|C) • fixed: P(S|C) are matched to the closest values in the default InterVA P(S|C) table. • empirical: no ranking is calculated, but use the empirical conditional probabilities directly, which will force updateCondProb to be FALSE for InSilicoVA algorithm.
age_group	Parameter specific to "EAVA" model, which identifies the age group of the input VA data. Possible values are "neonate" or "child".
...	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.*

Henry D. Kalter, Abdoulaye-Mamadou Roubanatou, Alain Koffi, and Robert E. Black. (2015). *Direct estimates of national neonatal and child cause-specific mortality proportions in Niger by expert algorithm and physician-coded analysis of verbal autopsy interviews. Journal of Global Health 5(1):010415.*

See Also

`insilico` in package **InSilicoVA**, `InterVA` in package **InterVA4**, `InterVA5` in package **InterVA5**, `interVA_train`, `tariff` in package **Tariff**, nbc function in package **nbc4va**, and `codEAVA` function in package **EAVA**.

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)
```

`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.

See Also

Other data conversion: ConvertData.phmrc()

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.*

See Also

Other data conversion: `ConvertData()`

Examples

```
## Not run:
# Starting from Jan 2024, PHMRC data requires registration at the GHDx website
# to doload. The following commands assume the user has download the file for
# PHMRC VA adult data from the website after logging in.

# For more details on the download process, see ?getPHMRC_url.

raw <- read.csv("IHME_PHMRC_VA_DATA_ADULT_Y2013M09D11_0.csv", 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("IHME_PHMRC_VA_DATA_ADULT_Y2013M09D11_0.csv", 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])

## End(Not run)
```

`csmf_eava` *Calculate CSMF from EAVA::codEAVA output*

Description

Calculate CSMF from EAVA::codEAVA output

Usage

```
csmf_eava(x)
```

Arguments

x	output from EAVA::codEAVA

Value

Named vector

DataEAVA *1,736 records of Sample Input*

Description

This is a dataset of 1,736 arbitrary (neonate and child) deaths in the acceptable format of EAVA. It is meant to be comparable to the RandomVA6 examples data set.

Format

1,736 arbitrary input records.

Examples

```
data(DataEAVA)
```

---

getCCC *Calculate Overall chance-corrected concordance (CCC)*

Description

Denote the cause-specific accuracy for the j-th cause to be (# of deaths correctly assigned to cause j) / (# of death due to cause j). For causes 1, 2, ..., C, the cause-specific CCC is computed for the j-th cause is defined to be (j-th cause-specific accuracy - 1 / C) / (1 - 1 / C) and the overall CCC is the average of each cause-specific CCC.

Usage

```
getCCC(cod, truth, C = NULL)
```

Arguments

cod	a data frame of estimated cause of death. The first column is the ID and the second column is the estimated cause.
truth	a data frame of true causes of death. The first column is the ID and the second column is the estimated cause.
C	the number of possible causes to assign. If unspecified, the number of unique causes in cod and truth will be used.

See Also

Other output extraction: getCSMF(), getCSMF_accuracy(), getIndivProb(), getTopCOD()

Examples

```
est <- data.frame(ID = c(1, 2, 3), cod = c("C1", "C2", "C1"))
truth <- data.frame(ID = c(1, 2, 3), cod = c("C1", "C3", "C3"))
# If there are only three causes
getCCC(est, truth)
# If there are 20 causes that can be assigned
getCCC(est, truth, C = 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.

See Also

Other output extraction: getCCC(), getCSMF_accuracy(), getIndivProb(), getTopCOD()

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 *Calculate CSMF accuracy*

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. If undetermined cause is present, it will be removed and the rest of the CSMF will be re-normalized to sum to 1.

Value

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

See Also

Other output extraction: getCCC(), getCSMF(), getIndivProb(), getTopCOD()

Examples

```
csmf1 <- c(0.2, 0.3, 0.5)
csmf0 <- c(0.3, 0.3, 0.4)
names(csmf0) <- names(csmf1) <- c("c1", "c2", "c3")
getCSMF_accuracy(csmf1, csmf0)
getCSMF_accuracy(csmf1, rev(csmf0))
```

`getIndivProb` *Extract individual distribution of cause of death*

Description

Extract individual distribution of cause of death

Usage

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

Arguments

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

Value

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

See Also

Other output extraction: `getCCC()`, `getCSMF()`, `getCSMF_accuracy()`, `getTopCOD()`

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 *Get the URL to the PHMRC dataset*

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

```
getPHMRC_url("adult")
```

getTopCOD *Extract the most likely cause(s) of death*

Description

Extract the most likely cause(s) of death

Usage

```
getTopCOD(x, interVA.rule = TRUE, n = 1, include.prob = FALSE)
```

Arguments

x	a fitted object from codeVA.
interVA.rule	Logical indicator for interVA object only. If TRUE and (the parameter) n <= 3, then the InterVA thresholds are used to determine the top causes.
n	Number of top causes to include (if n > 3, then the parameter interVA.rule is treated as FALSE).
include.prob	Logical indicator for including the probabilities (for insilico) or indicator of how likely the cause is (for interVA) in the results

Value

a data frame of ID, most likely cause assignment(s), and corresponding probability (for `insilico`) or indicator of how likely the cause is (for `interVA`)

See Also

Other output extraction: `getCCC()`, `getCSMF()`, `getCSMF_accuracy()`, `getIndivProb()`

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)
getTopCOD(fit)
## Not run:
library(openVA)

# InterVA4 Example
data(SampleInput)
fit_interva <- codeVA(SampleInput, data.type = "WHO2012", model = "InterVA",
                      version = "4.03", HIV = "l", Malaria = "l", write = FALSE)
getTopCOD(fit_interva, n = 1)
getTopCOD(fit_interva, n = 3)
getTopCOD(fit_interva, n = 3, include.prob = TRUE)
getTopCOD(fit_interva, interVA.rule = FALSE, n = 3)
getTopCOD(fit_interva, n = 5)
getTopCOD(fit_interva, n = 5, include.prob = TRUE)

# InterVA5 & Example
data(RandomVA5)
fit_interva5 <- codeVA(RandomVA5[1:50,], data.type = "WHO2016", model = "InterVA",
                       version = "5", HIV = "l", Malaria = "l", write = FALSE)
getTopCOD(fit_interva5, n = 1)
getTopCOD(fit_interva5, n = 3)
getTopCOD(fit_interva5, n = 3, include.prob = TRUE)
getTopCOD(fit_interva5, interVA.rule = FALSE, n = 3)
getTopCOD(fit_interva5, n = 5)
getTopCOD(fit_interva5, n = 5, include.prob = TRUE)

# InSilicoVA Example
data(RandomVA5)
fit_insilico <- codeVA(RandomVA5[1:100,], data.type = "WHO2016",
                       auto.length = FALSE)
getTopCOD(fit_insilico, n = 1)
getTopCOD(fit_insilico, n = 3)
getTopCOD(fit_insilico, n = 3, include.prob = TRUE)


# Tariff Example (only top cause is returned)
data(RandomVA3)
test <- RandomVA3[1:200, ]
```

```
train <- RandomVA3[201:400, ]
allcauses <- unique(train$cause)
fit_tariff <- tariff(causes.train = "cause", symps.train = train,
                     symps.test = test, causes.table = allcauses)
getTopCOD(fit_tariff, n = 1)

# NBC Example
library(nbc4va)
data(nbc4vaData)
train <- nbc4vaData[1:50, ]
test <- nbc4vaData[51:100, ]
fit_nbc <- nbc(train, test, known=TRUE)
getTopCOD(fit_nbc, n = 1)
getTopCOD(fit_nbc, n = 3)
getTopCOD(fit_nbc, n = 3, include.prob = TRUE)

## End(Not run)
```

`grouping_eava_child` *Mapping for child causes of death to the list used by the EAVA algorithm.*

Description

Intended for use when comparing assigned causes of death between the EAVA algorithm and InSilicoVA/InterVA5.

Format

Data Frame

Details

#'

Examples

```
data(grouping_eava_child)
```

grouping_eava_neonate *Mapping for neonate causes of death to the list used by the EAVA algorithm.*

Description

Intended for use when comparing assigned causes of death between the EAVA algorithm and InSilicoVA/InterVA5.

Format

Data Frame

Details

#'

Examples

```
data(grouping_eava_neonate)
```

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. • quantile: the rankings of the P(S|C) are obtained by matching the same quantile distributions in the default InterVA P(S|C) • fixed: P(S|C) are matched to the closest values in the default InterVA P(S|C) 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")
```

NeonatesVA5 *200 records of Sample Input*

Description

This is a dataset consisting of 200 arbitrary sample input neonatal deaths in the acceptable format of InterVA5.

Format

200 arbitrary input records.

Examples

```
data(NeonatesVA5)
```

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()
```

See Also

Other package status: `openVA_update()`

Examples

```
## Not run:
openVA_status()

## End(Not run)
```

openVA_update *Update openVA packages*

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()
```

See Also

Other package status: `openVA_status()`

Examples

```
## Not run:
openVA_update()

## End(Not run)
```

plot.eava *Create CSMF plot for EAVA::codEAVA output*

Description

Create CSMF plot for EAVA::codEAVA output

Usage

```
## S3 method for class 'eava'
plot(
  x,
  top = 10,
  title = "Top CSMF Distribution",
  type = "bar",
  return.barplot = FALSE,
  ...
)
```

Arguments

x	Output from EAVA::codEAVA
top	The number of top causes to include in plot. This is exceeded if there are ties.
title	Title for CSMF plot
type	An indicator of the type of chart to plot. "pie" for pie chart; "bar" for bar chart.
return.barplot	a logical indicating if the (barplot) ggplot() object should be returned (instead of printed). Default value is FALSE.
...	Not used.

Value

A barplot if return.barplot is TRUE; otherwise, nothing is returned.

---

plot.vacalibration *Plot CSMF from a vacalibration object*

Description

Plot CSMF from a vacalibration object

Usage

```
## S3 method for class 'vacalibration'
plot(
  x,
  type = c("errorbar", "bar", "compare")[1],
  algorithm = NULL,
  uncalibrated = FALSE,
  top = 10,
  title = "Top CSMF Distribution",
  xlab = "Causes",
  ylab = "CSMF",
  horiz = TRUE,
  angle = 60,
  fill = "lightblue",
  fill_uncalibrated = "lightpink",
  err_width = 0.4,
  err_size = 0.6,
  point_size = 2,
  border = "black",
  bw = TRUE,
  plot_it = TRUE,
  ...
)
```

Arguments

x	Fitted "vacalibration" objects
type	An indicator of the type of chart to plot. "errorbar" for line plots of only the error bars on single population; "bar" for bar chart with error bars on single population; "compare" for line charts on multiple calibrated algorithms.
algorithm	Name or vector of names of algorithm(s) which limits the output to those specific results
uncalibrated	Logical (TRUE/FALSE) indicator for including the uncalibrated CSMF in the plots (not valid with type is set to "compare").
top	The number of top causes (in the calibrated CSMF) to plot. If results from multiple algorithms are included in the fitted "vacalibration" object and type is set to "compare", it will plot the union of the top causes in all algorithms.
title	Title of the plot.
xlab	Labels for the causes.
ylab	Labels for the CSMF values.
horiz	Logical indicator indicating if the bars are plotted horizontally.
angle	Angle of rotation for the texts on x axis when horiz is set to FALSE
fill	The color to fill the bars when type is set to "bar".
fill_uncalibrated	The color to fill the bars for the uncalibrated CSMFs when type is set to "bar".
err_width	Size of the error bars.
err_size	Thickness of the error bar lines.
point_size	Size of the points.
border	The color to color the borders of bars when type is set to "bar".
bw	Logical indicator for setting the theme of the plots to be black and white.
plot_it	Logical (TRUE/FALSE) indicating if the first plot should be rendered.
...	Not used.

Value

A list of plots for each algorithms included in the fitted "vacalibration" object.

Examples

```
## Not run:
data(NeonatesVA5)
fit_insilico <- codeVA(NeonatesVA5)
insilico_prep <- prepCalibration(fit_insilico)
calib_insilico = vacalibration::vacalibration(va_data = insilico_prep,
                                              age_group = "neonate",
                                              country = "Mozambique")

fit_interva <- codeVA(NeonatesVA5, model = "InterVA", version = "5", write = FALSE)
interva_prep <- prepCalibration(fit_interva)
```

```
calib_interva = vacalibration::vacalibration(va_data = interva_prep,
                                             age_group = "neonate",
                                             country = "Mozambique")

two_fits <- prepCalibration(fit_insilico, fit_interva)
calib_ensemble = vacalibration::vacalibration(va_data = two_fits,
                                              age_group = "neonate",
                                              country = "Mozambique")
plot(calib_ensemble)

## End(Not run)
```

---

plotVA *Plot top CSMF for a fitted model*

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 in package **InSilicoVA**, CSMF in package **InterVA4**, CSMF5 in package **InterVA5**, plot.tariff in package **Tariff**.

Other visualization: stackplotVA()

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)
```

`prepCalibration` *Convert the assigned causes to the format expected by vacalibration::vacalibration().*

Description

This function prepares the output from codeVA (using the InSilicoVA or InterVA5 algorithms) and/or EAVA for the vacalibration function. The output is a named list, with each element consisting of a data frame with columns for the death ID ("ID") and the assigned CoD ("cause"). The name of each element is the algorithm used to assign the CoD – "insilicova", "interva", or "eava". If the

Usage

```
prepCalibration(fit, ...)
```

Arguments

`fit`	a fitted object from `codeVA` using either InSilicoVA, InterVA5, or output from `EAVA::codEAVA`. The EAVA algorithm is assumed if the the argument is a Data Frame (with columns "ID" and "cause").
`...`	more fitted objects from `codeVA` using either InSilicoVA or InterVA5, or from `EAVA::codEAVA`.

Value

a list with each element being a Data Frames that contains variables "ID" and "cause", and the element's name (in the list) is the algorithm name used to assign causes of death (either "insilicova", "interva", or "eava")

Examples

```
## Not run:
data(NeonatesVA5)
fit_insilico <- codeVA(NeonatesVA5)
insilico_prep <- prepCalibration(fit_insilico)
calib_insilico = vacalibration::vacalibration(va_data = insilico_prep,
                                              age_group = "neonate",
                                              country = "Mozambique")
```

```
summary(calib_insilico)

fit_interva <- codeVA(NeonatesVA5, model = "InterVA", version = "5", write = FALSE)
interva_prep <- prepCalibration(fit_interva)
calib_interva = vacalibration::vacalibration(va_data = interva_prep,
                                             age_group = "neonate",
                                             country = "Mozambique")
summary(calib_interva)

two_fits <- prepCalibration(fit_insilico, fit_interva)
calib_ensemble = vacalibration::vacalibration(va_data = two_fits,
                                              age_group = "neonate",
                                              country = "Mozambique")
summary(calib_ensemble)


## End(Not run)
```

`print.eava` *Print method for "eava" class.*

Description

Print method for "eava" class.

Usage

```
## S3 method for class 'eava'
print(x, ...)
```

Arguments

`x`	eava object.
`...`	not used

Examples

```
## Not run:
data(DataEAVA)
eava_results <- codeVA(DataEAVA, data.type = "EAVA", model = "EAVA")
eava_results

## End(Not run)
```

print.eava_summary *Print method for summarizing results from EAVA algorithm.*

Description

This function prints a summary message of the results along with the top cause-specific mortality fractions (CSMFs).

Usage

```
## S3 method for class 'eava_summary'
print(x, ...)
```

Arguments

x	eava object
...	not used

print.vacalibration *Print method for vacalibration model fits*

Description

This function is the print method for class vacalibration

Usage

```
## S3 method for class 'vacalibration'
print(x, ...)
```

Arguments

x	vacalibration object
...	not used

Examples

```
## Not run:
data(NeonatesVA5)
fit_insilico <- codeVA(NeonatesVA5, auto.length = FALSE)
insilico_prep <- prepCalibration(fit_insilico)
calib_insilico = vacalibration::vacalibration(va_data = insilico_prep,
                                              age_group = "neonate",
                                              country = "Mozambique",
                                              plot_it = FALSE)
calib_insilico

## End(Not run)
```

print.vacalibration_summary *Print method for summarizing vacalibration results*

Description

This function prints a summary message of the results along with the top cause-specific mortality fractions (CSMFs).

Usage

```
## S3 method for class 'vacalibration_summary'
print(x, top, rnd, algorithm, ...)
```

Arguments

x	vacalibration object
top	number of top CSMF to show
rnd	number of decimal places to round the CSMF
algorithm	a name or vector of names of algorithm(s) which limits the output to those specific results
...	not used

Examples

```
## Not run:
data(NeonatesVA5)
fit_insilico <- codeVA(NeonatesVA5)
insilico_prep <- prepCalibration(fit_insilico)
calib_insilico = vacalibration::vacalibration(va_data = insilico_prep,
                                              age_group = "neonate",
                                              country = "Mozambique")
summary(calib_insilico)

fit_interva <- codeVA(NeonatesVA5, model = "InterVA", version = "5", write = FALSE)
interva_prep <- prepCalibration(fit_interva)
calib_interva = vacalibration::vacalibration(va_data = interva_prep,
                                             age_group = "neonate",
                                             country = "Mozambique")
summary(calib_interva, top = 3, rnd = 2)

two_fits <- prepCalibration(fit_insilico, fit_interva)
calib_ensemble = vacalibration::vacalibration(va_data = two_fits,
                                              age_group = "neonate",
                                              country = "Mozambique")
summary(calib_ensemble, algorithm = "ensemble")
summary(calib_ensemble, algorithm = c("ensemble", "insilicova"))
```

```
## End(Not run)
```

RandomVA6 *1,736 records of Sample Input*

Description

This is a dataset consisting of 1,736 (neonate and child) deaths in the acceptable format of InSilicoVA. It is meant to be comparable to the DataEAVA examples data set.

Format

1,736 arbitrary input records.

Examples

```
data(RandomVA6)
```

---

stackplotVA *plot grouped CSMF from a "insilico" object*

Description

Produce bar plot of the CSMFs for a fitted object in broader groups. This function extends the stackplot() function in the InSilicoVA package to allow for the same visualization for results from InterVA, NBC, and Tariff algorithms.

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

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

Author(s)

Zehang Li, Tyler McCormick, Sam Clark

Maintainer: Zehang Li <lizehang@uw.edu>

See Also

Other visualization: plotVA()

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")
```

`summary.eava` *Summary of results obtained by fitting the EAVA algorithm.*

Description

This function prints a summary message of the results along with the top cause-specific mortality fractions (CSMFs).

Usage

```
## S3 method for class 'eava'
summary(object, top = 5, rnd = 4, ...)
```

Arguments

`object`	eava object
`top`	number of top CSMF to show
`rnd`	number of decimal places to round the CSMF
`...`	not used

Value

id	all IDs of the deaths
cause	assigned cause for individual of death
N	number of deaths
age_group	age group that the deaths belong to (either child or neonate)
csmf.ordered	cause-specific mortality fractions in decreasing order

Examples

```
## Not run:
data(DataEAVA)
eava_results <- codeVA(DataEAVA, data.type = "EAVA", model = "EAVA")
eava_summary <- summary(eava_results)
eava_summary

## End(Not run)
```

`summary.vacalibration` *Summary of results obtained by vacalibration*

Description

This function prints a summary message of the results along with the top cause-specific mortality fractions (CSMFs).

Usage

```
## S3 method for class 'vacalibration'
summary(object, top = 5, rnd = 4, algorithm = NULL, ...)
```

Arguments

object	`vacalibration` object
top	number of top CSMF to show
rnd	number of decimal places to round the CSMF
algorithm	a name or vector of names of algorithm(s) which limits the output to those specific results
...	not used

Examples

```
## Not run:
data(NeonatesVA5)
fit_insilico <- codeVA(NeonatesVA5, auto.length = FALSE)
insilico_prep <- prepCalibration(fit_insilico)
calib_insilico = vacalibration::vacalibration(va_data = insilico_prep,
                                              age_group = "neonate",
                                              country = "Mozambique",
                                              plot_it = FALSE)
calib_insilico_summ <- summary(calib_insilico)
names(calib_insilico_summ)
calib_insilico_summ

fit_interva <- codeVA(NeonatesVA5, model = "InterVA", version = "5", write = FALSE)
interva_prep <- prepCalibration(fit_interva)
calib_interva = vacalibration::vacalibration(va_data = interva_prep,
                                             age_group = "neonate",
                                             country = "Mozambique",
                                             plot_it = FALSE)
summary(calib_interva, top = 3)

two_fits <- prepCalibration(fit_insilico, fit_interva)
calib_ensemble = vacalibration::vacalibration(va_data = two_fits,
                                              age_group = "neonate",
                                              country = "Mozambique",
                                              plot_it = FALSE)
summary(calib_ensemble)

## End(Not run)
```

Index

∗ **EAVA**
codeVA, 3
∗ **InSilicoVA**
codeVA, 3
∗ **InterVA4**
codeVA, 3
interVA_train, 16
∗ **NBC4VA**
codeVA, 3
∗ **Tariff**
codeVA, 3
∗ **VA-Calibration**
codeVA, 3
∗ **data conversion**
ConvertData, 5
ConvertData.phmrc, 7
∗ **datasets**
DataEAVA, 9
grouping_eava_child, 15
grouping_eava_neonate, 16
NeonatesVA5, 18
RandomVA6, 27
∗ **output extraction**
getCCC, 9
getCSMF, 10
getCSMF_accuracy, 11
getIndivProb, 12
getTopCOD, 13
∗ **package status**
openVA_status, 18
openVA_update, 19
∗ **visualization**
plotVA, 22
stackplotVA, 27

codeVA, 3, *22*
ConvertData, 5, *8*
ConvertData.phmrc, *3*, *6*, 7
CSMF, *22*
CSMF5, *22*

csmf_eava, 8

DataEAVA, 9

getCCC, 9, *11, 12*, *14*
getCSMF, 10, *10–12*, *14*
getCSMF_accuracy, *10*, 11, *11, 12*, *14*
getIndivProb, *10, 11*, 12, *14*
getPHMRC_url, 13
getTopCOD, *10–12*, 13
grouping_eava_child, 15
grouping_eava_neonate, 16

insilico, *4, 5*
InterVA, *4, 5*
InterVA5, *5*
interVA_train, *4, 5*, 16

NeonatesVA5, 18

openVA_status, 18, *19*
openVA_update, *18*, 19

plot.eava, 19
plot.insilico, *22*
plot.tariff, *22*
plot.vacalibration, 20
plotVA, 22, *28*
prepCalibration, 23
print.eava, 24
print.eava_summary, 25
print.vacalibration, 25
print.vacalibration_summary, 26

RandomVA6, 27

stackplotVA, *22*, 27
summary.eava, 29
summary.vacalibration, 30

tariff, *4, 5*