Package 'ODA'

May 4, 2023

Maintainer Nathaniel Rhodes <nrhode@midwestern.edu>

License GPL-3

Title a package and interface for the MegaODA and CTA software suites

Version 2.0.0

Authors Nathaniel Rhodes [aut, cre] and Paul Yarnold [ctb,cph]

Description This package contains the functions needed to run and evaluate the output from MegaODA software. For any statistical hypothesis this non-parametric statistically-motivated machine-learning algorithm explicitly obtains the model which maximizes the (weighted) predictive accuracy for the sample. Many model validation methods are available. Users are encouraged to read about the ODA paradigm in Maximizing Predictive Accuracy (Paul Yarnold and Robert Soltysik, 2016), or on the ODA website. This package was developed by Nathaniel J. Rhodes to interface with ODA to assist the user in developing, evaluating, and validating maximum-accuracy ODA models.

URL https://odajournal.com/

Date 2023-05-03

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Depends R (>= 4.1.0), utils, stats

Suggests knitr,
rmarkdown,
testthat (>= 3.0.0)

VignetteBuilder knitr

Imports epitools,
dplyr,
filesstrings,
statmod,
stringr,
benchmarkme,
tidyr,

Config/testthat/edition 3

R topics documented:

`.onAttach` *.onAttach start message*

Description

.onAttach start message

Usage

```
.onAttach(libname, pkgname)
```

Arguments

`libname`	defunct
`pkgname`	defunct

Value

invisible()

`.onLoad` *.onLoad getOption package settings*

Description

.onLoad getOption package settings

Usage

```
.onLoad(libname, pkgname)
```

Arguments

libname	defunct
pkgname	defunct

Value

invisible()

`CTAparse` *Parse CTA output files.*

Description

Parses CTA model details, model predictions, and loads objects.

Usage

```
CTAparse(run = "", mod = "", type = "", weight = "", ...)
```

Arguments

run	The numeric value of the folder where the model output is saved
mod	The numeric value of the model number. Additional models can be specified in '...'
type	A character string containing one of the following tree model solution types: 'Unpruned','Pruned','Enumerated'. The default value is 'Enumerated'. See CTArun
weight	A logical value specifying whether the models being processed by CTAparse are weighted tree models.
...	Additional model numbers specified as a list

Details

When executed, CTAparse will return model performance metrics and data loaded within the global environment, which can be further evaluated.

The working directory must be directed toward the project, the input files must be located within the input folder, and model output files must be located within the output folder.

For each CTA model, CTAparse will return the **Effect Strength for Sensitivity** or ESS.

In binary classification problems the following relationships are defined:

$$PAC = \frac{(Sensitivity + Specificity)}{2} \times 100$$

$$ESS = \frac{(PAC - 50)}{(100 - 50)} \times 100$$

$$MPV = \frac{(PPV + NPV)}{2} \times 100$$

$$ESP = \frac{(MPV - 50)}{(100 - 50)} \times 100$$

Unlike mean PAC, ESS is normed against chance (Yarnold *et al*. 2005 and 2016) providing an intuitive scaling of model accuracy within the ODA paradigm.

The significance values reported are from 1. Monte Carlo simulations for all observations and 2. Fisher's Exact tests for LOO analysis.

The user is referred to the ODA User guide `ODAmanual` and is encouraged to review the data contained within the MODEL.OUT file for more information.

Value

The following objects with the run number appended are returned for CTA models:

`cta.denom`	Data frame containing the denominator values for the models contained within each output file in the `run` folder.
`cta.list`	Data frame containing the confusion matrix for each CTA model contained in the output file. This output can be used to evaluate the reproducibility of the model results using a Novometric bootstrap analysis, see `NOVOboot`.
`cta.model`	Data frame containing the parsed model output and significance for CTA models.
`cta.sum`	Data frame containing the overall accuracy, sensitivity, specificity, positive predictive value, and negative predictive value for each CTA model.

Author(s)

Nathaniel J. Rhodes

References

Yarnold P.R. and Soltysik R.C. (2005). *Optimal data analysis: Guidebook with software for Windows*. APA Books.

Yarnold, P.R. and Soltysik, R.C. (2016). *Maximizing Predictive Accuracy*. ODA Books. DOI: 10.13140/RG.2.1.1368.3286.

See Also

`ODAmanual` `ODAclean` `ODAload` `CTArun` `NOVOboot`

Examples

```
# Not run:
# CTAparse(run=1, mod=1, type="Enumerated",weight=F)
# CTAparse(1,1)
```

CTArun *Execute an CTA model run.*

Description

Creates and processes files needed for CTA.exe program

Usage

```
CTArun(
  run = "",
  path = getwd(),
  data = "data.txt",
  out = "model1.txt",
  vstart = "",
  vend = "",
  class = "",
  attribute = "",
  categorical = "",
  include = "",
  exclude = "",
  direction = "",
  force = F,
  forcenode = "",
  nodevar = "",
  miss = "",
  weight = "",
  usefisher = F,
  mciter = "",
  cutoff = "",
  stop = "",
  nopriors = F,
  mindenom = "",
  maxlevel = "",
  prune = "",
  skipnode = "",
  enumerate = "",
  loo = "",
  overwrite = FALSE
)
```

Arguments

run	A numerical value specifying the run folder containing the data
path	The working directory of the project stored as path
data	A character name specifying the data.txt file in the model folder
out	A character name specifying the output file with "model.out" as the default
vstart	A character name specifying the start variable, see key object from ODAload
vend	A character name specifying the end variable, see key object from ODAload

class	A character name specifying the class variable, see key object from ODAload
attribute	A character name specifying the attributes, see key object from ODAload
categorical	An optional character name specifying categorical attributes, see key object from ODAload
include	An optional character name specifying the variable and a value of observations that are included e.g., "v2=2" or "v3>=50"
exclude	An optional character name specifying the variable and a value that are excluded
direction	An optional character name and direction e.g., v2 < 1 or v2 LT 1 specifying a directional hypothesis for the class variable
force	A logical specifying whether CTA should insert nodevar attribute at forcenode node in the solution tree
forcenode	A numeric specifying the node to insert nodevar if force = T
nodevar	A character string specifiying the variable to be inserted at node forcenode if force = T
miss	A numeric value specifying a missing or NA value in the data with a default value set at -9, see ODAclean
weight	An optional character name of a variable containing a positive, non zero, weight value; cannot be the same as variables declared as class attribute or categorical attributes
usefisher	A logical specifying that all p values for categorical attributes will be determined using Fisher's Exact test rather than Monte Carlo
mciter	A numeric value specifying the number of Monte Carlo iterations used to estimate the type I error rate for a given attribute within the solution tree
cutoff	A cutoff value for the Type I error rate as an additional argument to mciter and stop
stop	A confidence level in percent which will stop the processing of a given attribute after the number of Monte Carlo iterations specified in mciter are completed and the Type I error cutoff is exceeded
nopriors	A logical specifying whether the ODA criterion will be weighted by the reciprocal of sample class membership. The default of leaving 'PRIORS' undeclared is equivalent to nopriors = F
mindenom	A numeric value that specifies that only those attributes which yield a solution tree node denominator of size 'MINDENOM' or larger will be allowed in the solution tree
maxlevel	A numeric value that specifies the deepest level of the solution tree
prune	A numeric value specifying the upper bound Type I error rate with which to prune the solution tree or. When nopriors = T, then prune should be specified as 'NOPRIORS'. The default value is 0.05
skipnode	A numeric value specifying the node number to skip.
enumerate	A logical specifying whether the top three attribute or a limited character string. 'ROOT' specifies that only the top node will have all attributes evaluated. 'MINOBS' followed by a numeric value e.g., 'MINOBS 1' which will only allow solution trees with at least MINOBS.
loo	A character string or numeric value specifying the upper bound Type 1 error rate required in jackknife i.e., leave-one-out or LOO cross generalizability analysis for every attribute in the tree. A specification of 'STABLE', the default value, requires that attributes have the same ESS in training and in LOO in order to be included in the solution tree

`overwrite` An optional logical value specifying whether previous files in the model folder should be overwritten with `overwrite = FALSE` as the default

Details

The working directory of the project is stored as `path`. All files needed for the model run must be located in the appropriate model run folder. See `ODAclean`.

`CTArun` will produce a command file with a .pgm extension, a model file with a .out extension, and a batch file with .bat extension. The .bat file must be executed within the model directory. Model output files can be parsed using `CTAparse`.

Value

Nothing is returned. Three files are created in the model folder:

`MODEL.out` The model file that contains the commands from CTA syntax and analysis results, see `ODAmanual`. This file is required for `CTAparse`.

`cta.pgm` The command file that contains all of the commands for CTA passed from R based on user input to `CTArun`.

`ctarun.bat` The batch file that must be executed to begin the model run.

Author(s)

Nathaniel J. Rhodes

References

Yarnold P.R. and Soltysik R.C. (2005). *Optimal data analysis: Guidebook with software for Windows*. APA Books.

Yarnold, P.R. and Soltysik, R.C. (2016). *Maximizing Predictive Accuracy*. ODA Books. DOI: 10.13140/RG.2.1.1368.3286

See Also

`ODAmanual` `ODAclean` `ODAload`

Examples

```
# Not run:
# CTArun(run=1,vstart="v1", vend="v201", class="v46 to v201", exclude ="v3=1", attr="v2 v4 to v44", mindenom =
```

`NOVOboot` *Perform novometric bootstrap analysis using a classification model*

Description

Perform novometric bootstrap analysis using a classification model

Usage

```
NOVOboot(
  data = "",
  run = "",
  predictor = "",
  outcome = "",
  nboot = "",
  seed = "",
  alternative = ""
)
```

Arguments

data	The name of a valid object created by ODAparse that begins with oda.list.x where x is the run.
run	The run number of the ODA model created using ODArun and parsed using ODAparse.
predictor	The model number within the ODArun which is read using ODAparse. This is the model that is compared to chance.
outcome	The outcome number, when more than one outcome was evaluated. For example, if three outcomes were evaluated simultaneously and the impact of outcome one was of interest, then enter 1.
nboot	The number of bootstrap replicates. Both model and chance will be evaluated using this number of replicates using a 50% resampling with replacement. The default value is 25,000 replicates.
seed	The seed number passed to set.seed that serves as the origin of the pseudorandom numbers generated for the bootstrap resampling. The default seed number is the current system time.
alternative	Specification of a two-tailed test as two.sided is assumed but alternative specifications are greater or or less

Details

The first axiom of novometric theory states that, for a random statistical sample "S" consisting of a class variable, one or more attributes, and a weight, the corresponding exact discrete confidence intervals, CIs, for model and for chance do NOT overlap. That is, a signficant model exists. If the CIs overlap, the effect is not statistically significant. However, if the CIs do not overlap, then the effect strength of the model is statistically significant at the confidence level selected by the user.

NOVOboot reports the Effect Strength for Sensitivity (ESS), which is the mean Percent Accuracy in Classification (mean PAC) corrected for chance, in quantiles from 0% to 100% in the first slot of the sum.boot.x. For a 2 x 2 matrix, mean PAC is equivalent to the ROC area. NOVOboot also reports the distribution of mean PAC in quantiles from 0% to 100% in the second slot of sum.boot.x. Because a binary classification matrix can also expressed as an odds ratio (OR) or as a risk ratio (RR), NOVOboot also reports estimates of the OR and RR for the model and chance effect distributions in the third and fourth slots of the sum.boot.x object, respectively. Likewise, one may wish to consider the corresponding distribution of p values from Fisher's Exact tests conducted for each bootstrap replicate. These distributions for both model and chance are reported in the fifth slot of the sum.boot.x object.

The results of this non-parametric bootstrap analysis provide a mechanism to evaluate the precision of point estimate of an effect provided by an ODA model. The same methodology can be applied to

any classifcation model wherein a two class and two attribute confusion matrix can be formulated. It is axiomatic that the exact discrete confidence intervals for model and chance define the boundry between "signal" and "noise". For effects wherein the distributions of model and chance overlap, it is concluded that the effect was not significantly different vs. the distribution of chance given the data.

Value

An array of percentiles from 0% to 100% capturing the resampled model and chance performance metrics. The simulated bootstraps are stored within the object `novo.boot.x` which is appended with the current `run` number `x`. The model and chance quantile summary is stored within the object `sum.boot.x` which is appended with the current `run` number `x`.

References

Yarnold, P.R. (2020). Reformulating the First Axiom of Novometric Theory: Assessing Minimum Sample Size in Experimental Desig *Optimal Data Analysis* **9**, 7-8. https://odajournal.files.wordpress.com/2020/01/v9a2.pdf

Yarnold, P.R. and Soltysik, R.C. (2016). *Maximizing Predictive Accuracy*. ODA Books. DOI: 10.13140/RG.2.1.1368.3286

See Also

ODArun ODAparse set.seed

Examples

```
## Not run
## NOVOboot(data=oda.list.1,run=1,predictor=1,outcome=1,seed=1234)

## Example of a moderate effect size (ESS)
ess <- ((((14/14)+(20/64))/2)-0.5)/.5    # 31.25% ESS, a moderate effect
data <- matrix(c(20,0,44,14),ncol=2,nrow=2,dimnames=list(c(0,1),c(0,1)))
data.raw <- epitools::expand.table(data)
data.tab <- list(table(cbind(data.raw[1],data.raw[2]),dnn=c("v1","x")))
oda.list.1 <- list()                  # supply list formatted for NOVOboot
oda.list.1[[1]] <- do.call("list",data.tab)  # supply data for NOVOboot

#NOVOboot(data=oda.list.1,run=1,predictor=1,outcome=1,seed=1234, nboot=25000)

# boot.1 <- novo.boot.1 # added after NOVOboot() run

# print(sum.boot.1[[1]]) # displays the quantiles of Model vs. Chance ESS

# boot.list <- setNames(data.frame(boot.1$ess.model,boot.1$ess.chance),
#c("Model","Chance"))

# df <- stack(boot.list,select=c("Model","Chance"))

# library(ggplot2)
# ggplot(df,aes(x=values)) +
# geom_histogram(data=subset(df, ind== 'Chance'),
fill="skyblue",colour="black",binwidth = 2) +
# geom_histogram(data=subset(df, ind== 'Model'),
fill="pink", colour="black",binwidth=2) +
```

```
# xlab("Effect strength for sensitivity (ESS %)") +
# ylab("Frequency of bootstrap replicates") +
# geom_vline(aes(xintercept =
# quantile(boot.1$ess.model,probs=0.025, na.rm=T), color="LB"),
# linetype="dashed") +
# geom_vline(aes(xintercept =
# quantile(boot.1$ess.chance,probs=0.975, na.rm=T), color="UB"),
# linetype="dashed") +
# labs(title = "Novoboot plot",
# subtitle = "ESS for Chance (Blue) and for Model (Red) (n=25000)") +
# scale_color_manual(name = "95% PI", values = c(UB = "Blue", LB = "Red")) +
# theme_bw()
```

`ODAclean` *Read and clean a data.csv input file and transform variables for use in* `ODArun` *or* `CTArun`

Description

A valid .csv file is imported, cleaned, and moved to a specified output folder. Data frame objects called `key` and `data` are loaded in the environment. The .csv file should be one level above the output folder.

Usage

```
ODAclean(
  data = "",
  output = "",
  type = "",
  miss = "",
  ipsative = "",
  normative = "",
  id = "",
  overwrite = FALSE
)
```

Arguments

`data`	The character name of the .csv file to be loaded and cleaned. The current working directory must be set to the Project folder.
`output`	An integer that specifies of the folder into which the cleaned data files will be written. If the directory does not exist, it will be created. If it does exist, the user will be asked whether the file contents should be overwritten.
`type`	A character string which should contain either "ODA" or "CTA". Default value is "ODA".
`miss`	A numeric value e.g., -9 that is imputed for all `NA` values in the imported data frame where missing or `NA` values exist.
`ipsative`	A character vector of variable names `x` in the data which will be standardized within an `id` i.e., $x - mean(x)/sd(x)$. An `id` variable must be supplied.

normative	A character vector of variable names x in the cleaned data which will be standardized across columns $x - mean(x)/sd(x)$.
id	A character vector that represents repeated measures. Observations within an id are standardized to the mean of the id number. More than 1 observation per subject is required.
overwrite	Logical value specifying whether files in output directory should be overwritten. Can be overridden by specifying TRUE or FALSE.

Value

Cleaned data is moved to the model type folder as both .txt and .csv files.

data.txt	A cleaned data file is moved to the output directory. Row and column names are removed from the .txt file. If specified, missing value replacements and standardized variables are also written to this file.
data.csv	The .csv file is also moved to the output directory.

Author(s)

Nathaniel J. Rhodes

Examples

```
# Not Run
# ODAclean(data="data.csv",type='ODA',output=1, miss=-9)
```

ODAload *Load data files and variable key for ODA or CTA*

Description

Loads the data file from the specified model folder and generates a variable key for use with ODArun and CTArun.

Usage

```
ODAload(run = "", type = "", ...)
```

Arguments

run	The numerical value of the folder number containing the data specified by the run number. This number will also be used to name objects uniquely by appending run to the object e.g. data.1 for Run 1. At least one value for this parameter must be supplied by the user as no defaults are supplied.
type	A character string which should contain either "ODA" or "CTA". Default value is "ODA".
...	Additional data files to load and review, if desired.

Details

The current working directory is stored as the path and the files to be loaded must be located in the Run folder.

Value

The following objects are loaded into R

`data`	Data frame of data.csv from specified `run` folder.
`key`	Data frame containing 2 columns: the variable names from the `data` and the coded variable names for use in with `ODArun` and `CTArun`.

Author(s)

Nathaniel J. Rhodes

Examples

```
# Not run:
# ODAload(1,type="CTA")
```

`ODAmanual` *Open user and function manuals.*

Description

Opens the ODA User Manual and function libraries

Usage

```
ODAmanual()
```

Details

Help for using ODA package

`ODAparse` *Parse ODA output files.*

Description

Parses ODA model details, model predictions, and loads objects.

Usage

```
ODAparse(run = "", ...)
```

Arguments

`run`	The numeric value of the folder where the model output is saved.
`...`	Additional run numbers specified as a list

Details

When executed, `ODAparse` will return model performance metrics and data loaded within the global environment, which can be further evaluated.

The working directory must be directed toward the project, the input files must be located within the input folder, and model output files must be located within the output folder.

For each ODA model, `ODAparse` will return the **Effect Strength for Sensitivity** or ESS and the **Effect Strength for Predictive Value** or ESP. Mean **Percent Accuracy in Classification** (PAC) and **Mean Predicitve Value** (MPV) are reported, as these are a common metrics for predictive model performance.

In binary classification problems the following relationships are defined:

$$PAC = \frac{(Sensitivity + Specificity)}{2} \times 100$$

$$ESS = \frac{(PAC - 50)}{(100 - 50)} \times 100$$

$$MPV = \frac{(PPV + NPV)}{2} \times 100$$

$$ESP = \frac{(MPV - 50)}{(100 - 50)} \times 100$$

Unlike mean PAC and MPV, both ESS and ESP are normed against chance (Yarnold *et al.* 2005 and 2016) providing an intuitive scaling of model accuracy within the ODA paradigm.

The significance values reported are from 1. Monte Carlo simulations for all observations and 2. Fisher's Exact tests for LOO analysis. Depending on whether a GEN, CAT, or ordered ODA model is detected, and if LOO jackknife is performed, the exact `oda.model` output will vary.

The user is referred to the ODA User guide `ODAmanual` and is encouraged to review the data contained within the MODEL.OUT file for more information.

`ODAparse` is not compatible with models where mcarlo = F see `ODArun`

Users may receive the unexpected warning: "Warning 7: WEIGHTed CATEGORICAL LOO not available. LOO switched to OFF." If this warning does not reflect specification of a categorical weighted model, it is possible that the attribute does not vary within one or more classes. Selection of a different attribute / predictor is suggested.

Value

The following objects with the run number appended are returned for ODA models:

`oda.data`	Data frame based on the data.csv file from specified `run` folder.
`oda.key`	Data frame containing 2 columns: the variable names from the `oda.data` and an ODA friendly alias e.g., v1 v2 for each attribute variable included in the ODA model.
`oda.list`	Data frame containing the confusion matrix for each ODA model contained in the model.out file. This output can be used to evaluate the reproducibility of the model results using a Novometric bootstrap analysis, see `NOVOboot`.

oda.model	Data frame containing the parsed model output and the ESS, ESP, and significance for ODA models.
oda.perf	Data frame containing the overall accuracy, sensitivity, specificity, positive predictive value, and negative predictive value for each attribute included in the ODA model. Univariate OR and 95% CI are also presented. The Haldane-Anscombe-Gart correction is made observed cell counts of zero and a warning is displayed.
oda.stats	Data frame containing the classification summary for each ODA model contained in the model.out file.

Author(s)

Nathaniel J. Rhodes

References

Yarnold P.R. and Soltysik R.C. (2005). *Optimal data analysis: Guidebook with software for Windows*. APA Books.

Yarnold, P.R. and Soltysik, R.C. (2016). *Maximizing Predictive Accuracy*. ODA Books. DOI: 10.13140/RG.2.1.1368.3286.

See Also

ODAmanual ODAclean ODAload ODArun NOVOboot

Examples

```
# Not run:
# ODAparse(1)
```

ODApower *Estimate power for an ODA model.*

Description

Statistical power is estimated for a unit weighted binary application with balanced samples in each of 2 groups

Usage

```
ODApower(n1, n2, p1, p2, comp, alpha, nsim)
```

Arguments

n1	A numeric vector that contains the number of subjects in group 1
n2	A numeric vector that contains the number of subjects in group 2
p1	A numeric value that is the proportion of group 1 with the outcome
p2	A numeric value that is the proportion of group 2 with the outcome

comp	An integer value specifying the number of experiment wise comparions for a Sidak type adjustment to alpha
alpha	Numeric value specifying the a priori level of signifanced assumed
nsim	An integer value specifying the number of Fisher's Exact Tests to simulate.

Details

A default of 10,000 Monte Carlo Fisher's Exact tests are simulated and compared to alpha to estimate power.

The resulting power estimate represents a "worst case scenario" for the lowest level of measurement accuracy, i.e., a two class and two attribute problem, see Rhodes 2020.

For unit weighted applications, a Fisher's Exact test is isomorphic to the power of an ODA model, see Yarnold *et al.* 2005.

The *a priori* signifiance level alpha is adjusted based on number of comparisons (comp) as follows: $alpha_{adjusted} = 1 - (1 - alpha)^{1/comp}$.

Value

An array of power estimates with nrows of length n1 and ncol of length comp

Author(s)

Nathaniel J. Rhodes

References

Rhodes N.J. (2020). Statistical Power Analysis in ODA, CTA and Novometrics. *Optimal Data Analysis* **9**, 21-25. https://odajournal.files.wordpress.com/2020/02/v9a5.pdf

Yarnold P.R. and Soltysik R.C. (2005). *Optimal data analysis: Guidebook with software for Windows*. APA Books.

See Also

fisher.test power.fisher.test

Examples

```
n1 <- seq(15,50,5)
n2 <- seq(15,50,5)
p1 <- 0.74
p2 <- 0.26
alpha <- 0.05
comp <- 1
nsim <- 100
#Power for an analysis with an ESS of 48% (a moderate effect)
ess <- 100*(((0.74+0.74)/2)-0.5)/0.5
ODApower(n1=n1,n2=n2,comp=comp,p1=p1,p2=p2,alpha=alpha,nsim=nsim)
```

ODArun *Execute an ODA model run.*

Description

Creates and processes files needed for MegaODA.exe program

Usage

```
ODArun(
  run = "",
  path = getwd(),
  data = "data.txt",
  out = "model.out",
  hold = "",
  vstart = "",
  vend = "",
  class = "",
  attribute = "",
  categorical = "",
  include = "",
  exclude = "",
  direction = "",
  degen = "",
  gen = "",
  primary = "",
  secondary = "",
  nopriors = F,
  miss = "",
  weight = "",
  mcarlo = T,
  iter = "25000",
  target = "",
  sidak = "",
  stop = "",
  adjust = F,
  setseed = "",
  loooff = F,
  overwrite = FALSE
)
```

Arguments

run	A numerical value specifying the run folder containing the data
path	The working directory of the project stored as path
data	A character name specifying the data.txt file in the model folder
out	A character name specifying the output file with "model.out" as the default
hold	A character name specifying the holdout data file, omitted by default
vstart	A character name specifying the start variable, see key object from ODAload

vend	A character name specifying the end variable, see key object from ODAload
class	A character name specifying the class variable, see key object from ODAload
attribute	A character name specifying the attributes, see key object from ODAload
categorical	An optional character name specifying categorical attributes, see key object from ODAload
include	An optional character name specifying the variable and a value of observations that are included e.g., "v2=2" or "v3>=50"
exclude	An optional character name specifying the variable and a value that are excluded
direction	An optional character name and direction e.g., v2 < 1 or v2 LT 1 specifying a directional hypothesis for the class variable
degen	An optional character name specifying attributes for which degenerate solutions are allowed, off by default
gen	An optional character name specifying the variable whose values denote groups for a multisample generalizability analysis, off by default
primary	An optional character vector specifying the primary criterion for choosing among optimal solutions, see Details
secondary	An optional character vector specifying the secondary criterion for choosing among optimal solutions, see Details
nopriors	An optional logical value specifying whether the ODA criterion is weighted by the reciprocal of class membership, with nopriors = FALSE as the default
miss	A numeric value specifying a missing or NA value in the data with a default value set at -9, see ODAclean
weight	An optional character name of a variable containing a positive, non zero, weight value; cannot be the same as variables declared as class attribute categorical or gen
mcarlo	A logical value specifying whether Monte Carlo analysis should be used to estimate Type I error with mcarlo = TRUE as the default
iter	An integer value specifying the maximum number of Monte Carlo iteratations to be reached before halting and must be specified if mcarlo = TRUE
target	An optional numerical value specifying the target level of alpha < target to be reached before halting and must be specified if sidak or stop are utilized
sidak	An optional integer value specifying an adjustment to target based on the number of experiment wise comparisons and must be combined with target
stop	An optional numerical value specifying the confidence level that Type I error is less than target to be reached before halting and must be combined with target
adjust	An optional logical value specifying whether tied Monte Carlo iterations are to be split in half with adjust = FALSE as the default
setseed	An optional integer value specifying a seed number for Monte Carlo analysis with the current system time as the default
loooff	An optional logical value specifying whether leave one out or LOO analysis should be turned off with loooff = FALSE as the default
overwrite	An optional logical value specifying whether previous files in the model folder should be overwritten with overwrite = FALSE as the default

Details

The working directory of the project is stored as path. All files needed for the model run must be located in the appropriate model run folder. See ODAclean.

ODArun will produce a command file with a .pgm extension, a model file with a .out extension, and a batch file with .bat extension. The .bat file must be executed within the model directory. Model output files can be parsed using ODAparse.

The ODA algorithm explicitly maximizes the classification accuracy which is achieved for the training sample see Yarnold, 2005.

The use of Optimal in Optimal Data Analysis means that an ODA model achieves the theoretically maximum possible level of accuracy in any given application. For more information see ODAmanual.

ODA utilizes primary and secondary criteria for selecting among multiple optimal solutions. By default, when not specified and when priorsoff = FALSE primary is set to maxsens. By default, when not specified secondary is set to samplerep.

When gen is not active, other options include: maxsens meansens samplerep balanced distance random sens(attribute)

When gen is active, other options include: genmean and gensens(attribute)

There are several disallowed run specifications. Error checking is automatically performed on the user inputs. However, if the to keyword is used with a range of variables, it is suggested that the user confirm that the desired analysis was performed as some error checking is not available. The following cannot be combined in ODArun: weight cannot be both declared and listed as any of the following class attribute gen. Likewise, gen cannot be both declared and listed as any of the following attribute categorical class weight.

Value

Nothing is returned. Three files are created in the model folder:

MODEL.out	The model file that contains the commands from MEGAODA syntax and analysis results, see ODAmanual This file is required for ODAparse.
oda.pgm	The command file that contains all of the commands for MEGAODA passed from R based on user input to ODArun.
odarun.bat	The batch file that must be executed to begin the model run.

Author(s)

Nathaniel J. Rhodes

References

Yarnold P.R. and Soltysik R.C. (2005). *Optimal data analysis: Guidebook with software for Windows*. APA Books.

Yarnold, P.R. and Soltysik, R.C. (2016). *Maximizing Predictive Accuracy*. ODA Books. DOI: 10.13140/RG.2.1.1368.3286

See Also

ODAmanual ODAclean ODAload

Examples

```
# Not run:
# ODArun(run=1, vstart="v1",vend="v45",class="v45",attribute="v1 to v44", categorical="v1 v3")
```

`ODAsummary` *Merges results of ODAparse into printable report.*

Description

Transforms objects created by `ODAparse` and loads results.

Usage

```
ODAsummary(run = "", ...)
```

Arguments

`run`	The numerical value of the run folder in which the data file is stored
`...`	Additional run numbers specified as a list

Details

When run, `ODAsummary` will merge the ODA model summary and model performance metrics that are stored in the global environment.

The `run` number should be an existing run file within the current project tree. To function properly, `ODAparse` must first be completed for each run for which a summary report is being requested.

For more information on the objects generated by `ODAparse` see documentation for `ODAparse`. Specifically, `ODAsummary` will merge the object "oda.model.X" and "oda.perf.X" where X is the `run` number specified as an argument to `ODAparse` and `ODAsummary`.

The "oda.summary.X" object also includes several new variables each of which indicate whether the model results are significant in training "mcpsig" or LOO "loosig" or LOO-stable "loostab" indicating the ESS in training is the same as the ESS in LOO.

For models where loooff=T, "loostab" and "loosig" are not reported, see `ODArun` for more details on LOO specifications.

For GEN models, the report is limited to the overall model. Users should review the `ODAparse` outputs and the "oda.model.X" summary object to evaluate the results for each GEN sample / subgroup.

Value

The following objects are returned for ODA models:

`oda.summary.X`	Merged Object containing "oda.model.X" and "oda.perf.X" where X is specified by the `run` argument.

Author(s)

Nathaniel J. Rhodes

See Also

ODAmanual ODAparse ODArun

Examples

```
#' # Not run:
# ODAsummary(1)
# print(oda.summary.1)
```

ODAtree *Generate a folder tree for an ODA project*

Description

Establishes a subdirectory for an ODA project within a given working directory.

Usage

```
ODAtree(project = "NewProject", folder = getwd())
```

Arguments

project	A character string of a new project, e.g. "New Project"
folder	The full path to the root folder for the new project. Default is the current working directory.

Value

A new folder named as project containing the following subfolders:

CTA	The folder to contain the CTArun outputs and data files for the project.
ODA	The folder to contain the ODArun outputs and data files for the project.
Program	The folder containing the executable programs used for ODArun and CTArun analyses.
Rscript	The folder for the Rscript file with a skeleton Rscript containing command syntax for the project.

Examples

```
##Not run
##ODAtree("NewProject")
```

Index