Package 'chemmodlab'

May 2, 2022

Type Package

Title A Cheminformatics Modeling Laboratory for Fitting and Assessing Machine Learning Models

Version 2.0.0

Date 2022-05-01

Description Contains a set of methods for fitting models and methods for validating the resulting models. The statistical methodologies comprise a comprehensive collection of approaches whose validity and utility have been accepted by experts in the Cheminformatics field. As promising new methodologies emerge from the statistical and data-mining communities, they will be incorporated into the laboratory. These methods are aimed at discovering quantitative structure-activity relationships (QSARs). However, the user can directly input their own choices of descriptors and responses, so the capability for comparing models is effectively unlimited.

Depends R (>= 3.6.0)

License GPL-3

LazyData TRUE

URL https://github.com/jrash/ChemModLab

BugReports https://github.com/jrash/ChemModLab/issues

Imports KernSmooth, MSQC, class (>= 7.3.14), e1071 (>= 1.6.7), elasticnet (>= 1.1), lars(>= 1.2), MASS(>= 7.3.45), nnet(>= 7.3.12), pROC(>= 1.8), randomForest(>= 4.6.12), rpart(>= 4.1.10), tree (>= 1.0.37), pls (>= 2.5.0), caret (>= 6.0-71), stats, graphics, grDevices, utils, methods

RoxygenNote 7.1.1

NeedsCompilation no

Author Jacqueline Hughes-Oliver [aut], Jeremy Ash [aut, cre], Atina Brooks [aut]

Maintainer Jeremy Ash <jrash@ncsu.edu>

Suggests knitr, rmarkdown, testthat, vdiffr (>= 0.3.0)

Encoding UTF-8
Repository CRAN
Date/Publication 2022-05-01 23:30:02 UTC

R topics documented:

`aid364` *Cytotoxicity assay using the Jurkat human T-Cell line*

Description

These data are from a cytotoxicity assay conducted by the Scripps Research Institute Molecular Screening Center. There are 500 compounds assessed for toxicity using the the Jurkat human T-Cell line. 50 of these compounds were active (toxic). Visit https://pubchem.ncbi.nlm.nih.gov/bioassay/364 for more details.

Usage

```
aid364
```

Format

Data frame with 500 rows and 173 columns. The first column contains the compound ids. The second contains the outcome of the assay (a binary variable, indicating active/inactive). The next columns are chemical descriptor columns. Two descriptor sets are present. Both of these sets were computed using the software, PowerMV - see Liu et al. (2005) for more information. The first set of 24 continuous descriptors are a modification of the Burden number descriptors (Burden, 1989). The second set contains 147 binary descriptors, indicating the presence/absence of "pharmacophore" features, described in more detail in Liu et al. (2005).

Source

https://pubchem.ncbi.nlm.nih.gov/bioassay/364

References

Burden, F. R. (1989). Molecular identification number for substructure searches. Journal of Chemical Information and Computer Sciences, 29(3), 225-227.

Liu, K., Feng, J., & Young, S. S. (2005). PowerMV: a software environment for molecular viewing, descriptor generation, data analysis and hit evaluation. Journal of chemical information and modeling, 45(2), 515-522.

`ApplicabilityDomain` *Compute applicability domain for a chemmodlab model*

Description

`ApplicabilityDomain` evaluates the applicability domain for a chemmodlab model using a Hotteling T2 control chart.

Usage

```
ApplicabilityDomain(traindata, testdata, pvalue = 0.01, desname = NULL)
```

Arguments

`traindata`	training data
`testdata`	test data
`pvalue`	significance level for control limit threshold
`desname`	descriptor set name

`chemmodlab` *Constructor for the chemmodlab object*

Description

Constructor for the chemmodlab object

Usage

```
chemmodlab(
  all.preds,
  all.probs,
  model.acc,
  classify,
  responses,
  data,
  params,
  des.names,
  models,
  nsplits
)
```

Arguments

`all.preds`	a list of lists of dataframes. The elements of the outer list correspond to each split performed by `ModelTrain`. The elements of the inner list correspond to each descriptor set. For each descriptor set and CV split combination, the output is a dataframe containing all model predictions. The first column of each data frame contains the true value of the response. The remaining columns correspond to the models fit to the data.
`all.probs`	a list of lists of dataframes. Constructed only if there is a binary response. The structure is the same as `all.preds`, except that predictions are replaced by predicted probabilities of a response value of one. Predicted probabilities are only reported for classification models (see `ModelTrain`)
`model.acc`	a list of lists of model accuracy measures. The elements of the outer list correspond each split performed by `ModelTrain`. The elements of the inner list correspond to each descriptor set. For each descriptor set and CV split combination model accuracy measures for each model fit to the data. Regression models are assessed with Pearson's r and $RMSE$ Classification models are assessed with contingency tables.
`classify`	a logical. Was classification models used for binary response?
`responses`	a numeric vector. The true value of the response.
`data`	a list of numeric matrices. Each matrix is a descriptor set used as model input. The first column of each matrix is the response vector, and the remaining columns are descriptors.
`params`	a list of dataframes as made by `MakeModelDefaults`. Each dataframe contains the parameters to be set for a particular model.
`des.names`	a character vector specifying the descriptor set names. NA if unspecified.
`models`	a character vector specifying the models fit to the data.
`nsplits`	number of CV splits performed.

Author(s)

Jacqueline Hughes-Oliver, Jeremy Ash

See Also

`chemmodlab`, `plot.chemmodlab`, `CombineSplits`,

`CombineSplits` *ANOVA and multiple comparisons for chemmodlab objects*

Description

`CombineSplits` evaluates a specified performance measure across all splits created by `ModelTrain` and conducts statistical tests to determine the best performing descriptor set and model (D-M) combinations. `Performance` can evaluate many performance measures across all splits created by `ModelTrain`, then outputs a data frame for each D-M combination.

Usage

```
CombineSplits(cml.result, metric = "enhancement", m = NA, thresh = 0.5)

Performance(cml.result, metrics = "enhancement", m = NA, thresh = 0.5)
```

Arguments

cml.result	an object of class chemmodlab.
metric	the model performance measure to use. This should be one of error rate, enhancement, R2, rho, auc, sensitivity, specificity, ppv, fmeasure.
m	the number of tests to use for binary model performance measures (see Details). If m is not specified, enhancement uses floor(min(300,n/4)), where n is the number of observations. By default, all other binary performance measures are computed using all observations.
thresh	if the predicted probability that a binary response is 1 is above this threshold, an observation is classified as 1. Used to compute error rate, sensitivity, specificity, ppv, and fmeasure.
metrics	a character vector containing a subset of the performance measures above. Performance can compute several measures.

Details

CombineSplits quantifies how sensitive performance measures are to fold assignments (assignments to training and test sets). Intuitively, this assesses how much a performance measure may change if a slightly different data set is used.

ModelTrain is a designed study in that 'experimental' conditions are defined according to two factors: method (D-M combination) and split (fold assignment). The factor "split" is a blocking factor, and factor "method" is of primary interest. The design of this experiment is amenable to an analysis of variance to identify significant differences between performance measures according to factors and levels. CombineSplits outputs such an analysis of variance decomposition.

The multiple comparisons similarity (MCS) plot shows the results for tests for signficance in all pairwise differences of D-M mean performance measures. Because there can be many estimated mean performance measures for a dataset, care must be taken to adjust for multiple testing, and we do this using the Tukey-Kramer multiple comparison procedure (see Tukey (1953) and Kramer (1956)). If you are having trouble viewing all the components of the plot, make the plotting window larger.

By default, CombineSplits uses initial enhancement proposed by Kearsley et al. (1996) to assess model performance. Enhancement at m tests is the hit rate at m tests (accumulated actives at m tests divided by m) divided by the proportion of actives in the entire collection. It is a relative measure of hit rate improvement offered by the new method beyond what can be expected under random selection, and values much larger than one are desired. Initial enhancement is typically taken to be enhancement at m=300 tests.

Root mean squared error (RMSE), despite its popularity in statistics, may be inappropriate for continuous chemical assay responses because it assumes losses are equal for both under-predicting and over-predicting biological activity. A suitable alternative may be initial enhancement. Other options are the coeffcient of determination (R2) and Spearman's rho.

For binary chemical assay responses, alternatives to misclassification rate (`error rate`) (which may be inappropriate because it assigns equal weights to false positives and false negatives) include `sensitivity`, `specificity`, area under the receiver operating characteristic curve (`auc`), positive predictive value, also known as precision (`ppv`), F1 measure (`fmeasure`), and initial `enhancement`.

Functions

- `Performance`: outputs a data frame with performance measures for each D-M combination.

Author(s)

Jacqueline Hughes-Oliver, Jeremy Ash, Atina Brooks

References

Kearsley, S.K., Sallamack, S., Fluder, E.M., Andose, J.D., Mosley, R.T., and Sheridan, R.P. (1996). Chemical similarity using physiochemical property descriptors, J. Chem. Inf. Comput. Sci. 36, 118-127.

Kramer, C. Y. (1956). Extension of multiple range tests to group means with unequal numbers of replications. Biometrics 12, 307-310.

Tukey, J. W. (1953). The problem of multiple comparisons. Unpublished manuscript. In The Collected Works of John W. Tukey VIII. Multiple Comparisons: 1948-1983, Chapman and Hall, New York.

See Also

chemmodlab, ModelTrain

Examples

```
## Not run:
# A data set with  binary response and multiple descriptor sets
data(aid364)

cml <- ModelTrain(aid364, ids = TRUE, xcol.lengths = c(24, 147),
                  des.names = c("BurdenNumbers", "Pharmacophores"))
CombineSplits(cml)

## End(Not run)

# A continuous response
cml <- ModelTrain(USArrests, nsplits = 2, nfolds = 2,
                  models = c("KNN", "Lasso", "Tree"))
CombineSplits(cml)
```

HitEnrich *Plot hit enrichment curves from observed scores and activities*

Description

Plot hit enrichment curves based on scores from multiple algorithms. Actual activities are required. Additionally plot the ideal hit enrichment curve that would result under perfect scoring, and the hit enrichment curve that would result under random scoring. Optionally, simultaneous confidence bands may also be requested.

Usage

```
HitEnrich(
  S.df,
  labels = NULL,
  y,
  x.max = NULL,
  log = TRUE,
  title = "",
  conf = FALSE,
  conf.level = 0.95,
  method = "sup-t",
  plus = TRUE,
  band.frac = NULL
)
```

Arguments

S.df	Data frame where variables are numeric scores from different algorithms. Rows represent unique compounds.
labels	Character vector of labels for the different algorithms in S.df. If missing, variable names in S.df will be used.
y	Numeric vector of activity values. Activity values must be either 0 (inactive/undesirable) or 1 (active/desirable); no other values are accepted. Compounds are assumed to be in the same order as in S.df.
x.max	Integer, the maximum number of tests allowed on the x axis.
log	Logical. TRUE plots the x axis on a log scale.
title	Character string
conf	Logical. TRUE plots (simultaneous) confidence bands for all hit enrichment curves.
conf.level	Numeric, confidence coefficient
method	Character indicates the method used to obtain confidence bands. The default is sup-t but other options (not recommended) are "theta-proj" and "bonf".
plus	Logical. TRUE uses plus-adjusted version of method.

band.frac — Numeric vector of fractions tested to be used in obtaining confidence bands. Vector should be no longer than y, and should have at least 20 entries. Entries should be in (0,1]. It is recommended that entries be consistent with between 1 and x.max tests.

Details

By default, x.max is length(y), so that hit enrichment curves are obtained for all observable fractions, i.e., fractions of (1:length(y))/length(y). By default, confidence bands are evaluated based on a smaller grid of 40 fractions. This smaller grid is evenly spaced on either the original grid of (1:length(y))/length(y), or the log scale of the original grid.

HitEnrichDiff *Plot differences between hit enrichment curves*

Description

Plot differences between hit enrichment curves based on scores from multiple algorithms. Actual activities are required. Additionally plot simultaneous confidence bands for these differences. Plots may be used to determine if one algorithm is "better" than another algorithm.

Usage

```
HitEnrichDiff(
  S.df,
  labels = NULL,
  y,
  x.max = NULL,
  log = TRUE,
  title = "",
  conf.level = 0.95,
  method = "sup-t",
  plus = TRUE,
  band.frac = NULL,
  yrange = NULL
)
```

Arguments

S.df — Data frame where variables are numeric scores from at least 2 different algorithms. Rows represent unique compounds.

labels — Character vector of labels for the different algorithms in S.df. If missing, variable names in S.df will be used.

y — Numeric vector of activity values. Activity values must be either 0 (inactive/undesirable) or 1 (active/desirable); no other values are accepted. Compounds are assumed to be in the same order as in S.df.

x.max — Integer, the maximum number of tests allowed on the x axis.

log	Logical. TRUE plots the x axis on a log scale.
title	Character string
conf.level	Numeric, confidence coefficient
method	Character indicates the method used to obtain confidence bands. The default is sup-t but other options (not recommended) are "theta-proj" and "bonf".
plus	Logical. TRUE uses plus-adjusted version of method.
band.frac	Numeric vector of fractions tested to be used in obtaining confidence bands. Vector should be no longer than y, and should have at least 20 entries. Entries should be in (0,1]. It is recommended that entries be consistent with between 1 and x.max tests.
yrange	Numeric vector of length 2. The desired range for the y axis.

Details

By default, x.max is length(y), so that hit enrichment curves are obtained for all observable fractions, i.e., fractions of (1:length(y))/length(y). By default, confidence bands are evaluated based on a smaller grid of 40 fractions. This smaller grid is evenly spaced on either the original grid of (1:length(y))/length(y), or the log scale of the original grid.

MakeModelDefaults *Model parameters for ModelTrain*

Description

Makes a list containing the default parameters for all models implemented in ModelTrain.

Usage

```
MakeModelDefaults(n, p, classify, nfolds)
```

Arguments

n	The number of observations in the data.
p	The number of descriptors in the data.
classify	A logical. Will classification models be used? (is the response binary?) If false, regression models will be assumed.
nfolds	The number of folds used for k-fold cross validation.

Details

Sensible default values are selected for each tunable model parameter, however users may set any parameter manually by generating a list with this function and assigning the parameters.

See https://pages.github.ncsu.edu/jrash/chemmodlab/ for more information about the models available (including model default parameters).

Value

A list whose elements are dataframes containing the default parameter values for models implemented in ModelTrain.

Author(s)

Jeremy Ash

See Also

ModelTrain, chemmodlab

Examples

```
params <- MakeModelDefaults(n = nrow(USArrests),
 p = ncol(USArrests[, -1]), classify = TRUE, nfolds = 10)
params$Forest$mtry <- ncol(USArrests[, -1])-1
params

cml <- ModelTrain(USArrests, models = "RF", nsplits = 3,
 user.params = params)
```

ModelTrain *Fit predictive models to sets of descriptors.*

Description

ModelTrain is a generic S3 function that fits a series of classification or regression models to sets of descriptors and computes cross-validated measures of model performance.

Usage

```
ModelTrain(...)

## Default S3 method:
ModelTrain(
  x,
  y,
  nfolds = 10,
  nsplits = 3,
  seed.in = NA,
  des.names = NA,
 models = c("NNet", "PLS", "LAR", "Lasso", "PLSLDA", "Tree", "SVM", "KNN", "RF"),
  user.params = NULL,
  verbose = FALSE,
  ...
```

```
)

## S3 method for class 'data.frame'
ModelTrain(
  d,
  ids = FALSE,
  xcol.lengths = ifelse(ids, length(d) - 2, length(d) - 1),
  xcols = NA,
  nfolds = 10,
  nsplits = 3,
  seed.in = NA,
  des.names = NA,
 models = c("NNet", "PLS", "LAR", "Lasso", "PLSLDA", "Tree", "SVM", "KNN", "RF"),
  user.params = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`...`	Additional parameters.
`x`	a list of numeric descriptor set matrices. At the moment, only binary and continuous descriptors are supported. Binary descriptors should be numeric (0 or 1).
`y`	a numeric vector containing the binary or continuous response.
`nfolds`	the number of folds to use for each cross validation split.
`nsplits`	the number of splits to use for repeated cross validation.
`seed.in`	a numeric vector with length equal to `nsplits`. The seeds are used to randomly assign folds to observations for each repeated cross-validation split. If `NA`, the first seed will be 11111, the second will be 22222, and so on.
`des.names`	a character vector specifying the names for each descriptor set. The length of the vector must match the number of descriptor sets. If `NA`, each descriptor set will be named "Descriptor Set i", where i is the number of the descriptor set.
`models`	a character vector specifying the regression or classification models to use. The strings must match models implemented in ‘chemmodlab‘ (see Details).
`user.params`	a list of data frames where each data frame contains the parameter values for a model. The list should have the format of the list constructed by `MakeModelDefaults`. One can construct a list of parameters using `MakeModelDefaults` and then modify the parameters.
`verbose`	verbose mode or not?
`d`	a data frame containing an (optional) ID column, a response column, and descriptor columns. The columns should be provide in this order.
`ids`	a logical. Is an ID column provided?
`xcol.lengths`	a vector of integers. It is assumed that the columns in `d` are grouped by descriptor set. The integers specify the number of descriptors in each descriptor set.

They should be ordered as the descriptor sets are ordered in d. Users can specify multiple descriptor sets. By default there is one descriptor set, namely all columns in d except the response column and the optional ID column. Specify xcol.lengths or xcols, but not both.

xcols — A list of integer vectors. Each vector contains column indices of data where a set of descriptor variables is located. Users can specify multiple descriptor sets. Specify xcol.lengths or xcols, but not both.

Details

Multiple descriptor sets can be specified by the user. For each descriptor set, repeated k-fold cross validation is performed for the specified regression and/or classification models.

Not all modeling strategies will be appropriate for all response types. For example, partial least squares linear discriminant analysis ("PLSLDA") is not directly appropriate for continuous response assays such as percent inhibition, but it can be applied once a threshold value for percent inhibition is used to create a binary (active/inactive) response.

See https://jrash.github.io/chemmodlab/ for more information about the models available (including model default parameters). The default value for argument models includes only some of the possible values.

Sensible default values are selected for each tunable model parameter, however users may set any parameter manually using MakeModelDefaults and user.params.

ModelTrain predictions are based on k-fold cross-validation, where the dataset is randomly divided into k parts, each containing approximately equal numbers of compounds. Treating one of these parts as a "test set" the remaining k-1 parts are combined together as a "training set" and used to build a model from the desired modeling technique and descriptor set. This model is then applied to the "test set" to obtain predictions. The process is repeated, holding out each of the k parts in turn. One advantage of k-fold cross-validation is reduction in bias from using the same data to both build and assess a model. Another advantage is the increased precision of error estimation offered by k-fold cross validation over a one-time split.

Recognizing that the definition of folds in k-fold cross validation may have an impact on the observed performance measures, all models are built using the same definition of folds. This process is repeated to obtain multiple separate k-fold cross validation runs resulting in multiple separate definitions of folds. The number of these "splits" is specified by nsplits.

Observed performance measures are assessed across all splits using CombineSplits. This function assesses how sensitive performance measures are to fold assignments, or changes to the training and test sets. Statistical tests are used to determine the best performing model and descriptor set combination.

Value

A list is returned of class chemmodlab containing:

all.preds — a list of lists of data frames. The elements of the outer list correspond to each CV split performed by ModelTrain. The elements of the inner list correspond to each descriptor set. For each descriptor set and CV split combination, the output is a dataframe containing all model predictions. The first column of each data frame contains the true value of the response. The remaining columns contain the predictions for each model.

`all.probs` a list of lists of data frames. Constructed only if there is a binary response. The structure is the same as `all.preds`, except that predictions are replaced by "predicted probabilities" (i.e. estimated probabilities of a response value of one). Predicted probabilities are only reported for classification models.

`model.acc` a list of lists of model accuracy measures. The elements of the outer list correspond to each CV split performed by `ModelTrain`. The elements of the inner list correspond to each descriptor set. For each descriptor set and CV split combination, a limited collection of performance measures are given for each model fit to the data. Regression models are assessed with Pearson's r and $RMSE$. Classification models are assessed with contingency tables. For additional model performance measures, see `Performance`

.

`classify` a logical. Were classification models used for binary response?

`responses` a numeric vector. The observed value of the response.

`data` a list of numeric matrices. Each matrix is a descriptor set used as model input.

`params` a list of data frames as made by `MakeModelDefaults`. Each data frame contains the parameters to be set for a particular model.

`des.names` a character vector specifying the descriptor set names. NA if unspecified.

`models` a character vector specifying the models fit to the data.

`nsplits` number of CV splits performed.

Methods (by class)

- `default`: Default S3 method
- `data.frame`: S3 method for class 'data.frame'

Author(s)

Jacqueline Hughes-Oliver, Jeremy Ash, Atina Brooks

See Also

`chemmodlab`, `plot.chemmodlab`, `CombineSplits`,

Examples

```
## Not run:
# A data set with  binary response and multiple descriptor sets
data(aid364)

cml <- ModelTrain(aid364, ids = TRUE, xcol.lengths = c(24, 147),
                  des.names = c("BurdenNumbers", "Pharmacophores"))
cml

## End(Not run)
```

```
# A continuous response
cml <- ModelTrain(USArrests, nsplits = 2, nfolds = 2,
                  models = c("KNN", "Lasso", "Tree"))
cml
```

PerfCurveBands *Construct a confidence band for a recall or precision curve*

Description

PerfCurveBands takes a pair of score and activity vectors as input. A performance curve and confidence band is created for the selected testing fractions.

Usage

```
PerfCurveBands(
  S,
  X,
  r,
  metric = "rec",
  type = "band",
  method = "sup-t",
  plus = T,
  conf.level = 0.95,
  boot.rep = 100,
  mc.rep = 1e+05,
  myseed = 111,
  h = NULL
)
```

Arguments

S	a vector of scores.
X	a vector of activities.
r	a vector of testing fractions.
metric	the performance curve to use. Options are recall ("rec") and precision ("prec").
type	specifies whether a point-wise confidence interval ("pointwise") or a confidence band ("band") should be constructed.
method	the method to use. Point-wise confidence interval options are "binomial", "JZ", "bootstrap". Confidence band options are "sup-t", "theta-proj".
plus	should plus correction be used or not?
conf.level	the confidence level for the bands.
boot.rep	the number of replicates to use for the bootstrap method.

mc.rep	the number of Monte Carlo replicates to use for the sup-t method.
myseed	the random seed.
h	the bandwidth for the local regression estimator of Lambda. If NULL, uses the default plugin estimator.

`PerfCurveTest` *Perform a hypothesis test for the difference between two performance curves.*

Description

`PerfCurveTest` takes score vectors for two scoring algorithms and an activity vector. A performance curve is created for the two scoring algorithms and hypothesis tests are performed at the selected testing fractions.

Usage

```
PerfCurveTest(
  S1,
  S2,
  X,
  r,
  metric = "rec",
  method = "EmProc",
  type = "pointwise",
  plus = T,
  pool = F,
  alpha = 0.05,
  h = NULL,
  seed = 111,
  mc.rep = 1e+05
)
```

Arguments

S1	a vector of scores for scoring algorithm 1.
S2	a vector of scores for scoring algorithm 2.
X	a vector of activities.
r	a vector of testing fractions.
metric	the performance curve to use. Options are recall ("rec") and precision ("prec").
method	the method to use. Recall options are c("EmProc", "binomial", "JZ ind", "mcnemar", "binomial ind"). Precision options are c("EmProc", "binomial", "JZ ind", "stouffer", "binomial ind").
type	specifies whether a point-wise confidence interval ("pointwise") or a confidence band ("band") should be constructed.

plus	should plus correction be applied to the confidence intervals?
pool	use pooling for hypothesis tests? Only relevant to "EmProc".
alpha	the significance level.
h	the bandwidth for the local regression estimator of Lambda. If NULL, uses the default plugin estimator.
seed	the random seed.
mc.rep	the number of Monte Carlo replicates to use for the sup-t method.

---

plot.chemmodlab *Plot method for the chemmodlab class.*

Description

plot.chemmodlab takes a chemmodlab object output by the ModelTrain function and creates a series of accumulation curve plots for assesing model and descriptor set performance.

Usage

```
## S3 method for class 'chemmodlab'
plot(
  x,
  max.select = NA,
  splits = 1:x$nsplits,
  meths = x$models,
  series = "both",
  ...
)
```

Arguments

x	an object of class chemmodlab.
max.select	the maximum number of tests to plot for the accumulation curve. If max.select is not specified, use floor(min(300,n/4)), where n is the number of compounds.
splits	a numeric vector containing the indices of the splits to use to construct accumulation curves. Default is to use all splits. NA means the first series of plots are not generated. See Details.
meths	a character vector with statistical methods implemented in chemmodlab. The statistical methods to use for the second series of plots. This argument can take the same values as argument models in function ModelTrain. See Details.
series	a character vector. Which series of plots to construct. Can be one of "descriptors", "methods", "both".
...	other parameters to be passed through to plotting functions.

Details

For a binary response, the accumulation curve plots the number of assay hits identified as a function of the number of tests conducted, where testing order is determined by the predicted probability of a response being positive obtained from k-fold cross validation. Given a particular compound collection, larger accumulations are preferable.

The accumulation curve has also been extended to continuous responses. Assuming large positive values of a continuous response y are preferable, chemmodlab accumulates y so that $\sum y_i$ is the sum of the y over the first n tests. This extension includes the binary-response accumulation curve as a special case.

By default, we display accumulation curves up to 300 tests, not for the entire collection, to focus on the goal of finding actives as early as possible.

There are two main series of plots generated:

Methods plot series

There is one plot per CV split and descriptor set combination. The accumulation curves for each modeling method is compared.

Descriptors plot series

There is one plot per CV split and model fit. The accumulation curves for each descriptor set is compared.

Author(s)

Jacqueline Hughes-Oliver, Jeremy Ash, Atina Brooks

References

Modified from code originally written by William J. Welch 2001-2002

See Also

chemmodlab, ModelTrain

Examples

```
## Not run:
# A data set with  binary response and multiple descriptor sets
data(aid364)

cml <- ModelTrain(aid364, ids = TRUE, xcol.lengths = c(24, 147),
                  des.names = c("BurdenNumbers", "Pharmacophores"))
plot(cml)

## End(Not run)

# A continuous response
cml <- ModelTrain(USArrests, nsplits = 2, nfolds = 2,
```

```
                    models = c("KNN", "Lasso", "Tree"))
plot(cml)
```

`pparg` *Docking scores for 3212 ligands for target PPARg*

Description

A dataset containing docking scores for the protein regulating gene peroxisome proliferator-activated receptor gamma (PPARg). 3212 ligands are scored. Scores are provided for three docking methods: Surflex-dock, ICM, and Vina. Scores are also provided for two consensus methods: the minimum rank consensus of Surflex-dock and ICM, and the maximum z-score consensus of Surflex-dock and ICM. Docking scores have been rescaled so that larger values suggest active ligands.

Usage

```
pparg
```

Format

A data frame with 3212 rows and 15 variables:

surf_id unique ligand identifier

surf_scores score from Surflex-dock

surf_actives activity label of ligand: 1 means active, 0 means not active

icm_id unique ligand identifier

icm_scores score from ICM

icm_actives activity label of ligand: 1 means active, 0 means not active

vina_id unique ligand identifier

vina_scores score from Vina

vina_actives activity label of ligand: 1 means active, 0 means not active

minr_id unique ligand identifier

minr_scores score from the minimum rank consensus of Surflex-dock and ICM

minr_actives activity label of ligand: 1 means active, 0 means not active

maxz_id unique ligand identifier

maxz_scores score from the maximum z-score consensus of Surflex-dock and ICM

maxz_actives activity label of ligand: 1 means active, 0 means not active

Source

http://stats.drugdesign.fr

Index