Package ‘scoringTools’

January 10, 2021

Type Package
Title Credit Scoring Tools
Version 0.1.2
Date 2020-12-16
Maintainer Adrien Ehrhardt <adrien.ehrhardt@centraliens-lille.org>
Description Grouping essential tools for credit scoring. These statistical tools may be use-
ful for other use-cases as well but were primarily designed for it. First, there are Reject Infer-
ence methods (Ehrhardt et al. (2017) <arXiv:1903.10855>). Second, we build upon the al-
ready CRAN-available package 'discretization' to automate discretization of continuous features.
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Imports discretization, sqldf, magrittr, dplyr, methods
Suggests MASS, rpart, Rmixmod, mvtnorm, ggplot2, speedglm, knitr,
markdown, plotly, pROC, shiny, testthat, covr
URL https://adimajo.github.io/scoringTools/
BugReports https://github.com/adimajo/scoringTools/issues
VignetteBuilder knitr
Collate 'allClasses.R' 'allMethods.R' 'augmentation.R'
  'check_consistency.R' 'chi2.R' 'chiM.R' 'cut.dataset.R'
  'data.R' 'discretize.cutp.R' 'echi2.R' 'fuzzyAugmentation.R'
  'generate_data.R' 'get_cutp.R' 'mdlp.R' 'methodsDisc.R'
  'modChi2.R' 'model_f.R' 'normalizedGini.R' 'parcelling.R'
  'reclassification.R' 'runDemo.R' 'scoringTools.R' 'topdown.R'
  'twins.R'
NeedsCompilation no
Author Adrien Ehrhardt [aut, cre]
Repository CRAN
Date/Publication 2021-01-10 17:20:02 UTC
R topics documented:

- scoringTools-package ........................................... 2
- augmentation ......................................................... 3
- chi2_iter ............................................................. 4
- chiM_iter ............................................................. 5
- discretization ......................................................... 7
- discretize ............................................................. 8
- echi2_iter ............................................................. 8
- fuzzy_augmentation .................................................. 10
- generate_data ........................................................ 11
- lendingClub .......................................................... 12
- mdlp_iter .............................................................. 13
- modChi2_iter .......................................................... 14
- normalizedGini ....................................................... 16
- parcelling ............................................................. 17
- plot ................................................................. 18
- predict ............................................................... 19
- reclassification ....................................................... 19
- reject_inferred-class ............................................... 21
- runDemo ............................................................. 21
- summary ............................................................... 21
- topdown_iter .......................................................... 22
- twins ................................................................. 24

Index ................................................................. 26

scoringTools-package  Credit Scoring Tools.

Description

Refer to the package’s vignette.

Author(s)

Maintainer: Adrien Ehrhardt <adrien.ehrhardt@centraliens-lille.org>

See Also

Useful links:

- https://adimajo.github.io/scoringTools/
- Report bugs at https://github.com/adimajo/scoringTools/issues
Description

This function performs Reject Inference using the Augmentation technique. Note that this technique is theoretically better than using the financed clients scorecard in the MAR and misspecified model case.

Usage

\texttt{augmentation(xf, xnf, yf)}

Arguments

\begin{itemize}
  \item \texttt{xf} \hspace{1cm} The matrix of financed clients' characteristics to be used in the scorecard.
  \item \texttt{xnf} \hspace{1cm} The matrix of not financed clients' characteristics to be used in the scorecard (must be the same features in the same order as \texttt{xf}!).
  \item \texttt{yf} \hspace{1cm} The matrix of financed clients' labels
\end{itemize}

Details

This function performs the Augmentation method on the data. When provided with labeled observations \((x^f, y)\), it first fits the logistic regression model \(p_\theta\) of \(x^f\) on \(y\), then reweights labeled observations according to their probability of being sampled, i.e. calculates the predicted probabilities of \(p_\theta\) on all observations, defines score-bands and calculates, in each of these score-bands, the probability of having been accepted as the proportion of labeled samples in that score-band. It then refits a logistic regression model \(p_\eta\) on the labeled samples.

Value

List containing the model using financed clients only and the model produced using the Augmentation method.

Author(s)

Adrien Ehrhardt

References


See Also

\texttt{glm}, \texttt{speedglm}
Examples

# We simulate data from financed clients
df <- generate_data(n = 100, d = 2)
xf <- df[, -ncol(df)]
yf <- df$y

# We simulate data from not financed clients (MCAR mechanism)
xnf <- generate_data(n = 100, d = 2)[, -ncol(df)]
augmentation(xf, xnf, yf)

chi2_iter

Wrapper function for the chi2 function from the discretization package.

Description

This function discretizes a training set using the chi2 method and the user-provided parameters and chooses the best discretization scheme among them based on a user-provided criterion and eventually a test set.

Usage

chi2_iter(
    predictors,
    labels,
    test = FALSE,
    validation = FALSE,
    proportions = c(0.3, 0.3),
    criterion = "gini",
    param = list(list(alp = 0.001, del = 0.5))
)

Arguments

predictors The matrix array containing the numeric attributes to discretize.
labels The actual labels of the provided predictors (0/1).
test Boolean : True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
validation Boolean : True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
proportions The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2, 0.2).
chiM_iter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criterion</td>
<td>The criterion (&quot;gini&quot;,&quot;aic&quot;,&quot;bic&quot;) to use to choose the best discretization scheme among the generated ones (default: &quot;gini&quot;). Nota Bene: it is best to use &quot;gini&quot; only when test is set to TRUE and &quot;aic&quot; or &quot;bic&quot; when it is not. When using &quot;aic&quot; or &quot;bic&quot; with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.</td>
</tr>
<tr>
<td>param</td>
<td>List providing the parameters to test (see ?discretization::chi2, default=list(list(alp=0.001, del=0.5))).</td>
</tr>
</tbody>
</table>

Author(s)

Adrien Ehrhardt

References


Examples

```r
# Simulation of a discretized logit model
x <- matrix(runif(300), nrow = 100, ncol = 3)
cuts <- seq(0, 1, length.out = 4)
xd <- apply(x, 2, function(col) as.numeric(cut(col, cuts)))
theta <- t(matrix(c(0, 0, 0, 2, 2, 2, -2, -2, -2), ncol = 3, nrow = 3))
log_odd <- rowSums(t(sapply(seq_along(xd[, 1]), function(row_id) {
  sapply(seq_along(xd[row_id, ]), function(element) theta[xd[row_id, element], element]
}
))
)y <- stats::rbinom(100, 1, 1 / (1 + exp(-log_odd)))
chi2_iter(x, y)
```

Description

This function discretizes a training set using the chiMerge method and the user-provided parameters and chooses the best discretization scheme among them based on a user-provided criterion and eventually a test set.
Usage

chiM_iter(
  predictors,
  labels,
  test = FALSE,
  validation = FALSE,
  proportions = c(0.3, 0.3),
  criterion = "gini",
  param = list(alpha = 0.05)
)

Arguments

predictors  The matrix array containing the numeric attributes to discretize.
labels      The actual labels of the provided predictors (0/1).
test        Boolean : True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
validation  Boolean : True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
proportions The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2, 0.2).
criterion   The criterion ("gini", "aic", "bic") to use to choose the best discretization scheme among the generated ones (default: "gini"). Nota Bene: it is best to use "gini" only when test is set to TRUE and "aic" or "bic" when it is not. When using "aic" or "bic" with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.
param       List providing the parameters to test (see ?discretization::chiM, default=list(alpha = 0.05)).

Details

This function discretizes a dataset containing continuous features $X$ in a supervised way, i.e. knowing observations of a binomial random variable $Y$ which we would like to predict based on the discretization of $X$. To do so, the ChiMerge algorithm starts by putting each unique values of $X$ in a separate value of the “discretized” categorical feature $E$. It then tests if two adjacent values of $E$ are significantly different using the $\chi^2$-test. In the context of Credit Scoring, a logistic regression is fitted between the “discretized” features $E$ and the response feature $Y$. As a consequence, the output of this function is the discretized features $E$, the logistic regression model of $E$ on $Y$ and the parameters used to get this fit.

Author(s)

Adrien Ehrhardt
References


See Also

glm, speedglm, discretization

Examples

# Simulation of a discretized logit model
x <- matrix(runif(300), nrow = 100, ncol = 3)
cuts <- seq(0, 1, length.out = 4)
xd <- apply(x, 2, function(col) as.numeric(cut(col, cuts)))
theta <- t(matrix(c(0, 0, 0, 2, 2, 2, -2, -2, -2), ncol = 3, nrow = 3))
log_odd <- rowSums(t(sapply(seq_along(xd[, 1]), function(row_id) {
   sapply(
      seq_along(xd[row_id, ]),
      function(element) theta[xd[row_id, element], element]
   )
}))

y <- stats::rbinom(100, 1, 1 / (1 + exp(-log_odd)))

chiM_iter(x, y)

discretization

Class discretization

Description

An S4 class to represent a discretization scheme.

Slots

method.name The name of the used discretization method.
parameters The parameters associated with the used method.
best.disc The best discretization scheme found by the method given its parameters.
performance The performance obtained with the method given its parameters.
disc.data The discretized data: test set if test is TRUE; if test is FALSE and validation is TRUE, then it provides the discretized validation set. Otherwise, it provides the discretized training set.
disc.data The continuous data: test set if test is TRUE; if test is FALSE and validation is TRUE, then it provides the discretized validation set. Otherwise, it provides the discretized training set.
discretize  

Generic method "discretize" for discretization objects.

Description
This defines the generic method "discretize" which will discretize a new input dataset given a discretization scheme of S4 class discretization.

Usage

\begin{verbatim}
discretize(object, data)

## S4 method for signature 'discretization'
discretize(object, data)
\end{verbatim}

Arguments

- **object**: the S4 discretization object
- **data**: new data to discretize

Details
This function discretizes a new data set using a previously learnt discretization scheme.

echi2_iter  

Wrapper function for the extended Chi2 function from the discretization package.

Description
This function discretizes a training set using the extended Chi2 method and the user-provided parameters and chooses the best discretization scheme among them based on a user-provided criterion and eventually a test set.

Usage

\begin{verbatim}
echi2_iter(
predictors,
labels,
test = FALSE,
validation = FALSE,
proportions = c(0.3, 0.3),
criterion = "gini",
param = list(alp = 0.5)
)
\end{verbatim}
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predictors</td>
<td>The matrix array containing the numeric attributes to discretize.</td>
</tr>
<tr>
<td>labels</td>
<td>The actual labels of the provided predictors (0/1).</td>
</tr>
<tr>
<td>test</td>
<td>Boolean: True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).</td>
</tr>
<tr>
<td>validation</td>
<td>Boolean: True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).</td>
</tr>
<tr>
<td>proportions</td>
<td>The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2,0.2).</td>
</tr>
<tr>
<td>criterion</td>
<td>The criterion (’gini’,’aic’,’bic’) to use to choose the best discretization scheme among the generated ones (default: ’gini’). Nota Bene: it is best to use ’gini’ only when test is set to TRUE and ’aic’ or ’bic’ when it is not. When using ’aic’ or ’bic’ with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.</td>
</tr>
<tr>
<td>param</td>
<td>List providing the parameters to test (see ?discretization::extendChi2, default=list(alp = 0.5)).</td>
</tr>
</tbody>
</table>

Details

This function discretizes a dataset containing continuous features $X$ in a supervised way, i.e. knowing observations of a binomial random variable $Y$ which we would like to predict based on the discretization of $X$. To do so, the ExtendedChi2 algorithm starts by putting each unique values of $X$ in a separate value of the “discretized” categorical feature $E$. It then tests if two adjacent values of $E$ are significantly different using the $\chi^2$-test. In the context of Credit Scoring, a logistic regression is fitted between the “discretized” features $E$ and the response feature $Y$. As a consequence, the output of this function is the discretized features $E$, the logistic regression model of $E$ on $Y$ and the parameters used to get this fit.

Author(s)

Adrien Ehrhardt

References


See Also

  glm, speedglm, discretization

Examples

```r
# Simulation of a discretized logit model
x <- matrix(runif(300), nrow = 100, ncol = 3)
cuts <- seq(0, 1, length.out = 4)
xd <- apply(x, 2, function(col) as.numeric(cut(col, cuts)))
theta <- t(matrix(c(0, 0, 0, 2, 2, -2, -2, -2), ncol = 3, nrow = 3))
log_odd <- rowSums(t(sapply(seq_along(xd[, 1]), function(row_id) {
  sapply(seq_along(xd[row_id, ]), function(element) theta[xd[row_id, element], element]
})))
y <- stats::rbinom(100, 1, 1 / (1 + exp(-log_odd)))
echi2_iter(x, y)
```

Description

This function performs Reject Inference using the Fuzzy Augmentation technique. Note that this technique has no theoretical foundation and should produce (under the identifiability assumption) the same parameters' estimates than the financed clients scorecard.

Usage

```r
fuzzy_augmentation(xf, xnf, yf)
```

Arguments

- `xf`: The matrix of financed clients' characteristics to be used in the scorecard.
- `xnf`: The matrix of not financed clients' characteristics to be used in the scorecard (must be the same in the same order as `xf`).
- `yf`: The matrix of financed clients' labels

Details

This function performs the Fuzzy Augmentation method on the data. When provided with labeled observations \((x^f, y)\), it first fits the logistic regression model \(p_\theta\) of \(x^f\) on \(y\), then labels the unlabelled samples \(x^u\) with the predicted probabilities of \(p_\theta\), i.e. \(\hat{y}^u = p_\theta(y|x^u)\) then refits a logistic regression model \(p_\eta\) on the whole sample.
Value

List containing the model using financed clients only and the model produced using the Fuzzy Augmentation method.

Author(s)

Adrien Ehrhardt

References


See Also

glm, speedglm

Examples

# We simulate data from financed clients
df <- generate_data(n = 100, d = 2)
xf <- df[, -ncol(df)]
yf <- df$y
# We simulate data from not financed clients (MCAR mechanism)
xnf <- generate_data(n = 100, d = 2)[, -ncol(df)]
fuzzy_augmentation(xf, xnf, yf)
Details

This function generates data from a uniform(0,1) distribution, and generates labels y according to a logistic regression on this data with random -1/1 parameter for each coordinate (MAR well-specified), the square of this data (MAR misspecified), or this data and some additional feature (from U(0,1) as well - MNAR).

Value

Dataframe containing features as x.1..d, labels as y.

Author(s)

Adrien Ehrhardt

References


Examples

# We simulate data from financed clients
generate_data(n = 100, d = 3, type = "MAR well specified")

lendingClub

Description

A dataset containing the information about Lending Club loans available online.

Usage

lendingClub

Format

A data frame with 2167 rows and 16 variables.

Source

https://www.lendingclub.com/
mdlp_iter

Wrapper function for the mdp function from the discretization package.

Description

This function discretizes a training set using the Minimum Description Length Principle method and the user-provided parameters.

Usage

mdlp_iter(
  predictors,
  labels,
  test = FALSE,
  validation = FALSE,
  proportions = c(0.3, 0.3),
  criterion = "gini"
)

Arguments

predictors The matrix array containing the numeric attributes to discretize.
labels The actual labels of the provided predictors (0/1).
test Boolean : True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
validation Boolean : True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
proportions The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2,0.2).
criterion The criterion ("gini",'aic','bic') to use to choose the best discretization scheme among the generated ones (default: 'gini'). Nota Bene: it is best to use 'gini' only when test is set to TRUE and 'aic' or 'bic' when it is not. When using 'aic' or 'bic' with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.

Details

This function discretizes a dataset containing continuous features $X$ in a supervised way, i.e. knowing observations of a binomial random variable $Y$ which we would like to predict based on the
discretization of $X$. To do so, the MdLP algorithm dichotomizes $X$ and puts the subsequent two values in the “discretized” categorical feature $E$. It chooses the cut-off point so as to minimize the resulting entropy and goes on in the subsequent two sub-spaces it just created. In the context of Credit Scoring, a logistic regression is fitted between the “discretized” features $E$ and the response feature $Y$. As a consequence, the output of this function is the discretized features $E$, the logistic regression model of $E$ on $Y$ and the parameters used to get this fit.

Author(s)
Adrien Ehrhardt

References


See Also
glm, speedglm, discretization

Examples
# Simulation of a discretized logit model
x <- matrix(runif(300), nrow = 100, ncol = 3)
cuts <- seq(0, 1, length.out = 4)
xd <- apply(x, 2, function(col) as.numeric(cut(col, cuts)))
theta <- t(matrix(c(0, 0, 0, 2, 2, 2, -2, -2, -2), ncol = 3, nrow = 3))
log_odd <- rowSums(t(sapply(seq_along(xd[, 1]), function(row_id) {
    sapply(seq_along(xd[row_id,]), function(element) theta[xd[row_id, element], element]
    )
}))
y <- stats::rbinom(100, 1, 1 / (1 + exp(-log_odd)))

mdlp_iter(x, y)
modChi2_iter

Description

This function discretizes a training set using the modified Chi2 method and the user-provided parameters and chooses the best discretization scheme among them based on a user-provided criterion and eventually a test set.

Usage

modChi2_iter(
  predictors,
  labels,
  test = FALSE,
  validation = FALSE,
  proportions = c(0.3, 0.3),
  criterion = "gini",
  param = list(alp = 0.5)
)

Arguments

predictors The matrix array containing the numeric attributes to discretize.
labels The actual labels of the provided predictors (0/1).
test Boolean: True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
validation Boolean: True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
proportions The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2,0.2).
criterion The criterion ("gini","aic","bic") to use to choose the best discretization scheme among the generated ones (default: "gini"). Nota Bene: it is best to use 'gini' only when test is set to TRUE and 'aic' or 'bic' when it is not. When using 'aic' or 'bic' with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.
param List providing the parameters to test (see ?discretization::modChi2, default=list(alp = 0.5)).

Details

This function discretizes a dataset containing continuous features $X$ in a supervised way, i.e. knowing observations of a binomial random variable $Y$ which we would like to predict based on the discretization of $X$. To do so, the ModifiedChi2 alorithm starts by putting each unique values of $X$ in a separate value of the “discretized” categorical feature $E$. It then tests if two adjacent values
of \( E \) are significantly different using the \( \chi^2 \)-test. In the context of Credit Scoring, a logistic regression is fitted between the “discretized” features \( E \) and the response feature \( Y \). As a consequence, the output of this function is the discretized features \( E \), the logistic regression model of \( E \) on \( Y \) and the parameters used to get this fit.

Author(s)
Adrien Ehrhardt

References

See Also
glm, speedglm, discretization

normalizedGini

Calculating the normalized Gini index

**Description**
This function calculates the Gini index of a classification rule outputting probabilities. It is a classical metric in the context of Credit Scoring. It is equal to 2 times the AUC (Area Under ROC Curve) minus 1.

**Usage**
normalizedGini(actual, predicted)

**Arguments**
- **actual**: The numeric binary vector of the actual labels observed.
- **predicted**: The vector of the probabilities predicted by the classification rule.

**Examples**
normalizedGini(c(1, 1, 1, 0, 0), c(0.7, 0.9, 0.5, 0.6, 0.3))
Description

This function performs Reject Inference using the Parcelling technique. Note that this technique is theoretically good in the MNAR framework although coefficients must be chosen a priori.

Usage

parcelling(
  xf,
  xnf,
  yf,
  probs = seq(0, 1, 0.25),
  alpha = rep(1, length(probs) - 1)
)

Arguments

xf The matrix of financed clients' characteristics to be used in the scorecard.

xnf The matrix of not financed clients' characteristics to be used in the scorecard (must be the same in the same order as xf!).

yf The matrix of financed clients' labels

probs The sequence of quantiles to use to make scorebands (see the vignette).

alpha The user-defined coefficients to use with Parcelling (see the vignette).

Details

This function performs the Parcelling method on the data. When provided with labeled observations \((x^f, y)\), it first fits the logistic regression model \(p_\theta\) of \(x^f\) on \(y\), then labels the unlabelled samples \(x^u\) with the observed bad rate in user-defined classes of predicted probabilities of \(p_\theta\) reweighted using user-supplied weights, i.e. \(\hat{y}^u = \alpha_k T(k)\) where \(k\) denotes the group (which depends on \(p_\theta\)) and \(T(k)\) the observed bad rate of labeled observations in this group. It then refits a logistic regression model \(p_\eta\) on the whole sample.

Value

List containing the model using financed clients only and the model produced using the Parcelling method.

Author(s)

Adrien Ehrhardt
References


See Also

glm, speedglm

Examples

# We simulate data from financed clients
df <- generate_data(n = 100, d = 2)
xf <- df[, -ncol(df)]
yf <- df$y

# We simulate data from not financed clients (MCAR mechanism)
xn <- generate_data(n = 100, d = 2)[, -ncol(df)]
parcelling(xf, xn, yf)

plot

Different kinds of plots using either plotly (if available) or the standard plot (graphics package).

Description

This function aims at producing useful graphs in the context of credit scoring in order to simplify the validation process of the produced credit score.

Usage

plot(x, y, ...)

plot.discretization(x, type)

## S4 method for signature 'discretization'
plot(x, type)

Arguments

x S4 discretization object.
y (For standard plots only)
... (For standard plots only)
type Type of plot. For now only "ROC" is supported.
**predict**

*Prediction on a raw test set of the best logistic regression model on discretized data.*

**Description**

This function discretizes a user-provided test dataset given a discretization scheme provided by an S4 "discretization" object. It then applies the learnt logistic regression model and outputs its prediction (see `predict.glm`).

**Usage**

```r
predict(object, ...)  
predict.discretization(object, newdata)  
predict.reject_infered(object, newdata, ...)  
## S4 method for signature 'discretization'  
predict(object, newdata)  
## S4 method for signature 'reject_infered'  
predict(object, newdata, ...)  
```

**Arguments**

- `object`  
  The S4 reject_infered object.
- `...`  
  Additional parameters to pass on to base predict.
- `newdata`  
  The test dataframe to discretize and for which we wish to have predictions.

**reclassification**

*Reclassification*

**Description**

This function performs Reject Inference using the Reclassification technique. Note that this technique has no theoretical foundation as it performs a one-step CEM algorithm.

**Usage**

```r
reclassification(xf, xnf, yf, thresh = 0.5)  
```
reclassification

Arguments

- `xf`: The matrix of financed clients’ characteristics to be used in the scorecard.
- `xnf`: The matrix of not financed clients’ characteristics to be used in the scorecard (must be the same in the same order as xf!).
- `yf`: The matrix of financed clients’ labels
- `thresh`: The threshold to use in the Classification step, i.e. the probability above which a not financed client is considered to have a label equal to 1.

Details

This function performs the Reclassification method on the data. When provided with labeled observations \((x^f, y)\), it first fits the logistic regression model \(p_\theta\) of \(x^f\) on \(y\), then considers that unlabeled observations are of the expected class given by the model \(p_\theta\) (this is equivalent to a CEM algorithm). It then refits a logistic regression model \(p_\eta\) on the whole sample.

Value

List containing the model using financed clients only and the model produced using the Reclassification method.

Author(s)

Adrien Ehrhardt

References


See Also

glm, speedglm

Examples

```r
# We simulate data from financed clients
xf <- matrix(runif(100 * 2), nrow = 100, ncol = 2)
theta <- c(2, -2)
log_odd <- apply(xf, 1, function(row) theta %*% row)
yf <- rbinom(100, 1, 1 / (1 + exp(-log_odd)))
# We simulate data from not financed clients (MCAR mechanism)
xnf <- matrix(runif(100 * 2), nrow = 100, ncol = 2)
reclassification(xf, xnf, yf)
```
**reject_infered-class**  
*Class reject_infered*

---

**Description**

An S4 class to represent a reject inference technique.

**Slots**

- `method_name`  The name of the used reject inference method.
- `financed_model`  The logistic regression model on financed clients.
- `acceptance_model`  The acceptance model (if estimated by the given method).
- `infered_model`  The logistic regression model resulting from the reject inference method.

---

**runDemo**

*Launch the Shiny demo app.*

---

**Description**

Launch the Shiny demo app.

**Usage**

```r
runDemo()
```

---

**summary**  
*Summary*

---

**Description**

Summary generic.

**Usage**

```r
summary(object, ...)

summary.discretization(object)
```

```r
## S4 method for signature 'discretization'
summary(object)
```

**Arguments**

- `object`  S4 discretization object.
- `...`  Other parameters to `summary`
topdown_iter  

Wrapper function for the 3 topdown functions from the discretization package.

---

**Description**

This function discretizes a training set using the user provided method(s) among the three topdown methods from the discretization package. Depending on the user providing a test and/or a validation set, the function returns the best discretization for logistic regression.

**Usage**

```r
topdown_iter(
  predictors,  # The matrix array containing the numeric attributes to discretize.
  labels,      # The actual labels of the provided predictors (0/1).
  test = F,    # Boolean : True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
  validation = F,  # Boolean : True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
  proportions = c(0.3, 0.3),  # The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2,0.2).
  criterion = "gini",  # The criterion ("gini","aic","bic") to use to choose the best discretization scheme among the generated ones (default: "gini"). Nota Bene: it is best to use "gini" only when test is set to TRUE and "aic" or "bic" when it is not. When using "aic" or "bic" with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.
  param = list(1, 2, 3)  # List providing the methods to test (from 1, 2 and 3, default: list(1,2,3)).
)
```

**Arguments**

- **predictors**: The matrix array containing the numeric attributes to discretize.
- **labels**: The actual labels of the provided predictors (0/1).
- **test**: Boolean : True if the algorithm should use predictors to construct a test set on which to search for the best discretization scheme using the provided criterion (default: TRUE).
- **validation**: Boolean : True if the algorithm should use predictors to construct a validation set on which to calculate the provided criterion using the best discretization scheme (chosen thanks to the provided criterion on either the test set (if true) or the training set (otherwise)) (default: TRUE).
- **proportions**: The list of the (2) proportions wanted for test and validation set. Only the first is used when there is only one of either test or validation that is set to TRUE. Produces an error when the sum is greater to one. Useless if both test and validation are set to FALSE. Default: list(0.2,0.2).
- **criterion**: The criterion ("gini","aic","bic") to use to choose the best discretization scheme among the generated ones (default: "gini"). Nota Bene: it is best to use "gini" only when test is set to TRUE and "aic" or "bic" when it is not. When using "aic" or "bic" with a test set, the likelihood is returned as there is no need to penalize for generalization purposes.
- **param**: List providing the methods to test (from 1, 2 and 3, default: list(1,2,3)).
Details

This function discretizes a dataset containing continuous features $X$ in a supervised way, i.e. knowing observations of a binomial random variable $Y$ which we would like to predict based on the discretization of $X$. To do so, the Topdown algorithms ... In the context of Credit Scoring, a logistic regression is fitted between the “discretized” features $E$ and the response feature $Y$. As a consequence, the output of this function is the discretized features $E$, the logistic regression model of $E$ on $Y$ and the parameters used to get this fit.

Author(s)

Adrien Ehrhardt

References


See Also

glm, speedglm, discretization

Examples

# Simulation of a discretized logit model
x <- matrix(runif(300), nrow = 100, ncol = 3)
cuts <- seq(0, 1, length.out = 4)
xd <- apply(x, 2, function(col) as.numeric(cut(col, cuts)))
theta <- t(matrix(c(0, 0, 0, 2, 2, 2, -2, -2, -2), ncol = 3, nrow = 3))
log_odd <- rowSums(t(sapply(seq_along(xd[, 1]), function(row_id) {
  sapply(
    seq_along(xd[row_id, ]),
    function(element) theta[xd[row_id, element], element]
  )}))
y <- stats::rbinom(100, 1, 1 / (1 + exp(-log_odd)))

topdown_iter(x, y)
twins

Twins

Description

This function performs Reject Inference using the Twins technique. Note that this technique has no theoretical foundation.

Usage

twins(xf, xnf, yf)

Arguments

xf The matrix of financed clients’ characteristics to be used in the scorecard.

xnf The matrix of not financed clients’ characteristics to be used in the scorecard (must be the same in the same order as xf!).

yf The matrix of financed clients’ labels

Details

This function performs the Twins method on the data. When provided with labeled observations \((x^f, y)\), it first fits the logistic regression model \(p_0\) of \(x^f\) on \(y\), then fits the logistic regression model \(p_\omega\) of \(X\) on the binomial random variable denoting the observation of the data \(Z\). We use predictions of both models on the labeled observations to construct a "meta"-score based on logistic regression which predicted probabilities are used to reweight samples and construct the final score \(p_\eta\).

Value

List containing the model using financed clients only, the model of acceptance and the model produced using the Twins method.

Author(s)

Adrien Ehrhardt

References


See Also

glm, speedglm
Examples

# We simulate data from financed clients
df <- generate_data(n = 100, d = 2)
exf <- df[, -ncol(df)]
yf <- df$y

# We simulate data from not financed clients (MCAR mechanism)
xnf <- generate_data(n = 100, d = 2)[, -ncol(df)]
twins(xf, xnf, yf)
Index

* Chi2
  echi2_iter, 8
  modChi2_iter, 14
* chi2
  chi2_iter, 4
* chiMerge
  chiM_iter, 5
* credit
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* datasets
  lendingClub, 12
* discretization
  chi2_iter, 4
  chiM_iter, 5
  echi2_iter, 8
  mdp_iter, 13
  modChi2_iter, 14
  topdown_iter, 22
* extended
  echi2_iter, 8
* gini index
  normalizedGini, 16
* inference
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* mdp
  mdp_iter, 13
* modified
  modChi2_iter, 14
* réintégration
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* reject
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* scorecard
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* scoring
  augmentation, 3
  fuzzy_augmentation, 10
  generate_data, 11
  parcelling, 17
  reclassification, 19
  twins, 24
* topdown
  topdown_iter, 22
  augmentation, 3
  chi2_iter, 4
  chiM_iter, 5
  discretization, 7
  discretize, 8
  discretize, discretization-method (discretize), 8
  echi2_iter, 8
fuzzy_augmentation, 10
generate_data, 11
lendingClub, 12
mdlp_iter, 13
modChi2_iter, 14
normalizedGini, 16
parcelling, 17
plot, 18
plot.discretization-method (plot), 18
plot.discretization (plot), 18
predict, 19
predict.discretization-method (predict), 19
predict.reject_infered-method (predict), 19
predict.discretization (predict), 19
predict.reject_infered (predict), 19
reclassification, 19
reject_infered-class, 21
runDemo, 21
scoringTools (scoringTools-package), 2
scoringTools-package, 2
summary, 21
summary.discretization-method (summary), 21
summary.discretization (summary), 21
topdown_iter, 22
twins, 24