Package ‘CustomerScoringMetrics’

April 6, 2018

Type Package
Title Evaluation Metrics for Customer Scoring Models Depending on Binary Classifiers
Version 1.0.0
Author Koen W. De Bock
Maintainer Koen W. De Bock <kdebock@audencia.com>
Description Functions for evaluating and visualizing predictive model performance (specifically: binary classifiers) in the field of customer scoring. These metrics include lift, lift index, gain percentage, top-decile lift, F1-score, expected misclassification cost and absolute misclassification cost. See Berry & Linoff (2004, ISBN:0-471-47064-3), Witten and Frank (2005, 0-12-088407-0) and Blattberg, Kim & Neslin (2008, ISBN:978–0–387–72578–9) for details. Visualization functions are included for lift charts and gain percentage charts. All metrics that require class predictions offer the possibility to dynamically determine cutoff values for transforming real-valued probability predictions into class predictions.
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 6.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2018-04-06 10:39:01 UTC

R topics documented:

checkDepVector .......................................................... 2
cnfMatrixMetrics ......................................................... 2
cumGainsChart .......................................................... 4
cumGainsTable .......................................................... 5
cutoffSensitivityPlot .................................................... 6
dynAccuracy .............................................................. 7
dynConfMatrix .......................................................... 8
expMisclassCost ......................................................... 10
checkDepVector  

Perform check on the true class label vector

Description

Perform check on the true class label vector.

Usage

checkDepVector(depTest)

Arguments

depTest    Vector with true data labels (outcome values)

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

Examples

```r
## Load response modeling predictions
data("response")
## Apply checkDepVector checking function
checkDepVector(response$test[,1])
```

confMatrixMetrics  

Obtain several metrics based on the confusion matrix

Description

Calculates a range of metrics based upon the confusion matrix: accuracy, true positive rate (TPR; sensitivity or recall), true negative rate (specificity), false positive rate (FPR), false negative rate (FPR), F1-score, with the optional ability to dynamically determine an incidence-based cutoff value using validation sample predictions.
Usage

confMatrixMetrics(predTest, depTest, cutoff = 0.5, dyn.cutoff = FALSE, predVal = NULL, depVal = NULL)

Arguments

predTest Vector with predictions (real-valued or discrete)
depTest Vector with real class labels
cutoff Threshold for converting real-valued predictions into class predictions. Default 0.5.
dyn.cutoff Logical indicator to enable dynamic threshold determination using validation sample predictions. In this case, the function determines, using validation data, the incidence (occurrence percentage of the customer behavior or characteristic of interest) and chooses a cutoff value so that the number of predicted positives is equal to the number of true positives. If TRUE, then the value for the cutoff parameter is ignored.
predVal Vector with predictions (real-valued or discrete). Only used if dyn.cutoff is TRUE.
depVal Optional vector with true class labels for validation data. Only used if dyn.cutoff is TRUE.

Value

A list with the following items:

accuracy accuracy value
truePositiveRate TPR or true positive rate
trueNegativeRate TNR or true negative rate
falsePositiveRate FPR or false positive rate
falseNegativeRate FNR or false negative rate
F1Score F1-score
cutoff the threshold value used to convert real-valued predictions to class predictions

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

References

cumGainsChart

See Also
dynConfMatrix, dynAccuracy

Examples

```r
## Load response modeling data set
data("response")
## Apply confMatrixMetrics function to obtain confusion matrix-based performance metrics
## achieved on the test sample. Use validation sample predictions to dynamically
## determine a cutoff value.
cmm <- confMatrixMetrics(response$test[,2], response$test[,1], dyn.cutoff=TRUE,
                        predVal=response$val[,2], depVal=response$val[,1])
## Retrieve F1-score
print(cmm$F1Score)
```

---

cumGainsChart  

*Plot a cumulative gains chart*

Description

Visualize gain through a cumulative gains chart.

Usage

```r
cumGainsChart(predTest, depTest, resolution = 1/10)
```

Arguments

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with true class labels
- `resolution`: Value for the determination of percentile intervals. Default 1/10 (10%).

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

References


See Also
topDecileLift, liftIndex, liftChart
Examples

```r
# Load response modeling predictions
data("response")
# Apply cumGainschart function to visualize cumulative gains of a customer response model
cumGainsChart(response$test[,2],response$test[,1])
```

---

`cumGainsTable`  
*Calculates cumulative gains table*

Description

Calculates a cumulative gains (cumulative lift) table, showing for different percentiles of predicted scores the percentage of customers with the behavior or characteristic of interest is reached.

Usage

`cumGainsTable(predTest, depTest, resolution = 1/10)`

Arguments

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with true class labels
- `resolution`: Value for the determination of percentile intervals. Default 1/10 (10%).

Value

A gain percentage table.

Author(s)

Koen W. De Bock, `<kdebock@audencia.com>`

References


See Also

- `topDecileLift`
- `liftIndex`
- `liftChart`
Examples

```r
## Load response modeling predictions
data("response")
## Apply cumGainsTable function to obtain cumulative gains table for test sample results
## and print results
cgt <- cumGainsTable(response$test[, 2], response$test[, 1])
print(cgt)
```

cutoffSensitivityPlot  Plot a sensitivity plot for cutoff values

Description

Visualize the sensitivity of a chosen metric to the choice of the threshold (cutoff) value used to transform continuous predictions into class predictions.

Usage

```r
cutoffSensitivityPlot(predTest, depTest, metric = c("accuracy", "expMisclassCost", "misclassCost"), costType = c("costRatio", "costMatrix", "costVector"), costs = NULL, resolution = 1/10)
```

Arguments

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with true class labels
- `metric`: Which metric to assess. Should be one of the following values: "accuracy", "misclassCost" or "expMisclassCost".
- `costType`: An argument that specifies how the cost information is provided. This should be either "costRatio" or "costMatrix" when metric equals "expMisclassCost"; or "costRatio", "costVector" or "costMatrix" when metric equals "MisclassCost". In the former case, a single value is provided which reflects the cost ratio (the ratio of the cost associated with a false negative to the cost associated with a false positive). In the latter case, a full (4x4) misclassification cost matrix should be provided in the form `rbind(c(0, 3), c(15, 0))` where in this example 3 is the cost for a false positive, and 15 the cost for a false negative case.
- `costs`: see `costType`
- `resolution`: Value for the determination of percentile intervals. Default 1/10 (10%).

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

See Also

dynAccuracy, misclassCost, expMisclassCost
Examples

```r
## Load response modeling predictions
data("response")
## Apply cutoffSensitivityPlot function to visualize how the cutoff value influences
## accuracy.
cutoffSensitivityPlot(response$test[,2],response$test[,1],metric="accuracy")
## Same exercise, but in function of misclassification costs
costs <- runif(nrow(response$test), 1, 50)
cutoffSensitivityPlot(response$test[,2],response$test[,1],metric="misclassCost",
costType="costVector",costs=costs, resolution=1/10)
```

dynAccuracy  Calculate accuracy

Description

Calculates accuracy (percentage correctly classified instances) for real-valued classifier predictions, with the optional ability to dynamically determine an incidence-based cutoff value using validation sample predictions.

Usage

dynAccuracy(predTest, depTest, dyn.cutoff = FALSE, cutoff = 0.5,
            predVal = NULL, depVal = NULL)

Arguments

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with real class labels
- `dyn.cutoff`: Logical indicator to enable dynamic threshold determination using validation sample predictions. In this case, the function determines, using validation data, the indicidence (occurrence percentage of the customer behavior or characteristic of interest) and chooses a cutoff value so that the number of predicted positives is equal to the number of true positives. If TRUE, then the value for the cutoff parameter is ignored.
- `cutoff`: Threshold for converting real-valued predictions into class predictions. Default 0.5.
- `predVal`: Vector with predictions (real-valued or discrete). Only used if `dyn.cutoff` is TRUE.
- `depVal`: Optional vector with true class labels for validation data. Only used if `dyn.cutoff` is TRUE.
**Value**

Accuracy value

- **accuracy**  
  accuracy value

- **cutoff**  
  the threshold value used to convert real-valued predictions to class predictions

**Author(s)**

Koen W. De Bock, <kdebock@audencia.com>

**See Also**

- `dynconfmatrix`
- `confmatrixmetrics`

**Examples**

```r
# Load response modeling data set
data("response")
# Apply dynAccuracy function to obtain the accuracy that is achieved on the test sample.
# Use validation sample predictions to dynamically determine a cutoff value.
acc <- dynAccuracy(response$test[,2], response$test[,1], dyn.cutoff = TRUE, predVal = response$val[,2], depVal = response$val[,1])
print(acc)
```

---

**dynConfMatrix**  
*Calculate a confusion matrix*

**Description**

Calculates a confusion matrix for real-valued classifier predictions, with the optional ability to dynamically determine an incidence-based cutoff value using validation sample predictions.

**Usage**

```r
dynConfMatrix(predTest, depTest, cutoff = 0.5, dyn.cutoff = FALSE, 
              predVal = NULL, depVal = NULL, returnClassPreds = FALSE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>predTest</code></td>
<td>Vector with predictions (real-valued or discrete)</td>
</tr>
<tr>
<td><code>depTest</code></td>
<td>Vector with real class labels</td>
</tr>
<tr>
<td><code>cutoff</code></td>
<td>Threshold for converting real-valued predictions into class predictions. Default 0.5.</td>
</tr>
</tbody>
</table>
dynConfMatrix

**dyn.cutoff** Logical indicator to enable dynamic threshold determination using validation sample predictions. In this case, the function determines, using validation data, the incidence (occurrence percentage of the customer behavior or characteristic of interest) and chooses a cutoff value so that the number of predicted positives is equal to the number of true positives. If TRUE, then the value for the cutoff parameter is ignored.

**predVal** Vector with predictions (real-valued or discrete). Only used if `dyn.cutoff` is TRUE.

**depVal** Optional vector with true class labels for validation data. Only used if `dyn.cutoff` is TRUE.

**returnClassPreds** Boolean value: should class predictions (using cutoff) be returned?

**Value**

A list with two elements:

- **confMatrix** a confusion matrix
- **cutoff** the threshold value used to convert real-valued predictions to class predictions
- **classPreds** class predictions, if requested using `returnClassPreds`

**Author(s)**

Koen W. De Bock, <kdebock@audencia.com>

**References**


**See Also**

dynAccuracy, confMatrixMetrics

**Examples**

```r
## Load response modeling data set
data("response")
## Apply dynConfMatrix function to obtain a confusion matrix. Use validation sample
## predictions to dynamically determine an incidence-based cutoff value.
cm <- dynConfMatrix(response$test[,2],response$test[,1],dyn.cutoff=TRUE,
predVal=response$val[,2],depVal=response$val[,1])
print(cm)
```
expMisclassCost  

*Calculate expected misclassification cost*

**Description**

Calculates the expected misclassification cost value for a set of predictions.

**Usage**

```r
expMisclassCost(predTest, depTest, costType = c("costRatio", "costMatrix"),
    costs = NULL, cutoff = 0.5, dyn.cutoff = FALSE, predVal = NULL,
    depVal = NULL)
```

**Arguments**

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with real class labels
- `costType`: An argument that specifies how the cost information is provided. This should be either "costRatio" or "costMatrix". In the former case, a single value is provided which reflects the cost ratio (the ratio of the cost associated with a false negative to the cost associated with a false positive). In the latter case, a full (4x4) misclassification cost matrix should be provided in the form `rbind(c(0,3), c(15,0))` where in this example 3 is the cost for a false positive, and 15 the cost for a false negative case.
- `costs`: see `costType`
- `cutoff`: Threshold for converting real-valued predictions into class predictions. Default 0.5.
- `dyn.cutoff`: Logical indicator to enable dynamic threshold determination using validation sample predictions. In this case, the function determines, using validation data, the indidicence (occurrence percentage of the customer behavior or characteristic of interest) and chooses a cutoff value so that the number of predicted positives is equal to the number of true positives. If TRUE, then the value for the cutoff parameter is ignored.
- `predVal`: Vector with predictions (real-valued or discrete). Only used if `dyn.cutoff` is TRUE.
- `depVal`: Optional vector with true class labels for validation data. Only used if `dyn.cutoff` is TRUE.

**Value**

A list with

- **EMC**: expected misclassification cost value
- **cutoff**: the threshold value used to convert real-valued predictions to class predictions
Author(s)
Koen W. De Bock, <kdebock@audencia.com>

See Also
dynConfMatrix, misclassCost

Examples
## Load response modeling data set
data("response")
## Apply expMisclassCost function to obtain the misclassification cost for the
## predictions for test sample. Assume a cost ratio of 5.
emc<-expMisclassCost(response$test[,2],response$test[,1],costType="costRatio", costs=5)
print(emc$EMC)

liftchart

Generate a lift chart

Description
Visualize lift through a lift chart.

Usage
liftChart(predTest, depTest, resolution = 1/10)

Arguments
predTest Vector with predictions (real-valued or discrete)
depTest Vector with true class labels
resolution Value for the determination of percentile intervals. Default 1/10 (10%).

Author(s)
Koen W. De Bock, <kdebock@audencia.com>

References

See Also
topDecileLift, liftIndex, liftChart
Examples

```r
## Load response modeling predictions
data("response")
## Apply liftChart function to visualize lift table results
liftChart(response$test[,2], response$test[,1])
```

---

**liftIndex**

*Calculate lift index*

Description

Calculates lift index metric.

Usage

`liftIndex(predTest, depTest)`

Arguments

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with true class labels

Value

Lift index value

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

References


See Also

`liftTable`, `topDecileLift`, `liftChart`

Examples

```r
## Load response modeling predictions
data("response")
## Calculate lift index for test sample results
li<-liftIndex(response$test[,2], response$test[,1])
print(li)
```
Calculate lift table

Description

Calculates a lift table, showing for different percentiles of predicted scores how much more the characteristic or action of interest occurs than for the overall sample.

Usage

liftTable(predTest, depTest, resolution = 1/10)

Arguments

- predTest: Vector with predictions (real-valued or discrete)
- depTest: Vector with true class labels
- resolution: Value for the determination of percentile intervals. Default 1/10 (10%).

Value

A lift table.

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

References


See Also

topDecileLift, liftIndex, liftChart

Examples

```r
## Load response modeling predictions
data("response")
## Apply liftTable function to obtain lift table for test sample results and print
## results
lt<-liftTable(response$test[,2],response$test[,1])
print(lt)
```
misclassCost  

*Calculate misclassification cost*

**Description**

Calculates the absolute misclassification cost value for a set of predictions.

**Usage**

```r
misclassCost(predTest, depTest, costType = c("costRatio", "costMatrix", "costVector"), costs = NULL, cutoff = 0.5, dyn.cutoff = FALSE, predVal = NULL, depVal = NULL)
```

**Arguments**

- `predTest`: Vector with predictions (real-valued or discrete)
- `depTest`: Vector with real class labels
- `costType`: An argument that specifies how the cost information is provided. This should be either "costRatio" or "costMatrix". In the former case, a single value is provided which reflects the cost ratio (the ratio of the cost associated with a false negative to the cost associated with a false positive). In the latter case, a full (4x4) misclassification cost matrix should be provided in the form `rbind(c(0,3),c(15,0))` where in this example 3 is the cost for a false positive, and 15 the cost for a false negative case.
- `costs`: see `costType`
- `cutoff`: Threshold for converting real-valued predictions into class predictions. Default 0.5.
- `dyn.cutoff`: Logical indicator to enable dynamic threshold determination using validation sample predictions. In this case, the function determines, using validation data, the incidence (occurrence percentage of the customer behavior or characteristic of interest) and chooses a cutoff value so that the number of predicted positives is equal to the number of true positives. If TRUE, then the value for the cutoff parameter is ignored.
- `predVal`: Vector with predictions (real-valued or discrete). Only used if `dyn.cutoff` is TRUE.
- `depVal`: Optional vector with true class labels for validation data. Only used if `dyn.cutoff` is TRUE.

**Value**

A list with the following elements:

- `misclassCost`: Total misclassification cost value
- `cutoff`: the threshold value used to convert real-valued predictions to class predictions
response

Author(s)
Koen W. De Bock, <kdebock@audencia.com>

References

See Also
dynConfMatrix.expMisclassCost.dynAccuracy

Examples
```r
## Load response modeling data set
data("response")
## Generate cost vector
costs <- runif(nrow(response$test), 1, 100)
## Apply misclassCost function to obtain the misclassification cost for the
## predictions for test sample. Assume a cost ratio of 5.
emc <- misclassCost(response$test[,2], response$test[,1], costType="costVector", costs=costs)
print(emc$EMC)
```

---

response

Description
Predicted customer response probabilities and true responses for a customer scoring model. Includes results for two data samples: a test sample (response$test) and a validation sample (response$val).

Usage
data(response)

Format
A list with two elements: response$test and response$val, both are data frames with data for 2 variables: preds and dep.

Author(s)
Authors: Koen W. De Bock Maintainer: <kdebock@audencia.com>
Examples

```r
# Load data
data(response)
# Calculate incidence in test sample
print(sum(response$test[,1]=="cl1")/nrow(response$test))
```

Description

Calculates top-decile lift, a metric that expresses how the incidence in the 10% customers with the highest model predictions compares to the overall sample incidence. A top-decile lift of 1 is expected for a random model. A top-decile lift of 3 indicates that in the 10% highest predictions, 3 times more positive cases are identified by the model than would be expected for a random selection of instances. The upper boundary of the metric depends on the sample incidence and is given by 100% / Incidence %. E.g. when the incidence is 10%, top-decile lift can be no higher than 10.

Usage

```r
topDecileLift(predTest, depTest)
```

Arguments

- `predTest` Vector with predictions (real-valued or discrete)
- `depTest` Vector with true class labels

Value

Top-decile lift value

Author(s)

Koen W. De Bock, <kdebock@audencia.com>

References


See Also

`liftTable`, `liftIndex`, `liftChart`
Examples

```r
## Load response modeling predictions
data("response")
## Calculate top-decile lift for test sample results
tdl <- topDecileLift(response$test[,2], response$test[,1])
print(tdl)
```
Index

*Topic datasets
  response, 15

checkDepVector, 2
confMatrixMetrics, 2, 8, 9
cumGainsChart, 4
cumGainsTable, 5
cutoffSensitivityPlot, 6
dynAccuracy, 4, 6, 7, 9, 15
dynConfMatrix, 4, 8, 8, 11, 15
expMisclassCost, 6, 10, 15
liftChart, 4, 5, 11, 11, 12, 13, 16
liftIndex, 4, 5, 11, 12, 13, 16
liftTable, 12, 13, 16
misclassCost, 6, 11, 14
response, 15
topDecileLift, 4, 5, 11–13, 16