Package ‘tools4uplift’

January 6, 2021

Version 1.0.0
Date 2021-01-04
Title Tools for Uplift Modeling
Author Mouloud Belbahri, Olivier Gandouet, Alejandro Murua, Vahid Partovi Nia
Maintainer Mouloud Belbahri <mouloud.belbahri@gmail.com>
Description Uplift modeling aims at predicting the causal effect of an action such as a marketing campaign on a particular individual. In order to simplify the task for practitioners in uplift modeling, we propose a combination of tools that can be separated into the following ingredients: i) quantization, ii) visualization, iii) variable selection, iv) parameters estimation and, v) model validation. For more details, see <https://dms.umontreal.ca/~murua/research/UpliftRegression.pdf>.
Depends R (>= 3.1.2)
Imports BiasedUrn, dplyr, glmnet, latticeExtra, lhs
Suggests lattice
License GPL-2 | GPL-3
Repository CRAN
Repository/R-Forge/Project tools4uplift
Repository/R-Forge/Revision 65
Repository/R-Forge/DateTimeStamp 2021-01-06 15:58:39
Date/Publication 2021-01-06 16:50:02 UTC
NeedsCompilation no

R topics documented:

barplot.PerformanceUplift ........................................... 2
BestFeatures .............................................................. 3
BinUplift ................................................................. 4
BinUplift2d ............................................................... 6
DualUplift ................................................................. 7
InterUplift ............................................................... 8
LassoPath ............................................................... 9
Description

Barplot of observed uplift with respect to predicted uplift sorted from the highest to the lowest.

Usage

## S3 method for class 'PerformanceUplift'
barplot(height, ...)

Arguments

height a table that must be the output of PerformanceUplift function.
...
additional barplot arguments.

Value

a barplot and the associated Kendall’s uplift rank correlation

Author(s)

Mouloud Belbahri

References


See Also

PerformanceUplift
Examples

library(tools4uplift)
data("SimUplift")

model <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")

# performance of the heat map uplift estimation on the training dataset
perf <- PerformanceUplift(data = model, treat = "treat",
                            outcome = "y", prediction = "Uplift_X1_X2",
                            equal.intervals = TRUE, nb.group = 5)

barplot(perf)

BestFeatures

Qini-based feature selection

Description

Qini-based Uplift Regression in order to select the features that maximize the Qini coefficient.

Usage

BestFeatures(data, treat, outcome, predictors, rank.precision = 2,
              equal.intervals = FALSE, nb.group = 10,
              validation = TRUE, p = 0.3)

Arguments

data : a data frame containing the treatment, the outcome and the predictors.
treat : name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
outcome : name of a binary response (numeric) vector (coded as 0/1).
predictors : a vector of names representing the predictors to consider in the model.
rank.precision : precision for the ranking quantiles to compute the Qini coefficient. Must be 1 or 2. If 1, the ranking quantiles will be rounded to the first decimal. If 2, to the second decimal.
equal.intervals : flag for using equal intervals (with equal number of observations) or the true ranking quantiles which result in an unequal number of observations in each group to compute the Qini coefficient.

nb.group : the number of groups for computing the Qini coefficient if equal.intervals is TRUE - Default is 10.
validation : if TRUE, the best features are selected based on cross-validation - Default is TRUE.
if validation is TRUE, the desired proportion for the validation set. p is a value between 0 and 1 expressed as a decimal, it is set to be proportional to the number of observations per group - Default is 0.3.

Details

The regularization parameter is chosen based on the interaction uplift model that maximizes the Qini coefficient. Using the LASSO penalty, some predictors have coefficients set to zero.

Value

a vector of names representing the selected best features from the penalized logistic regression.

Author(s)

Mouloud Belbahri

References


Examples

```r
library(tools4uplift)
data("SimUplift")
features <- BestFeatures(data = SimUplift, treat = "treat", outcome = "y", predictors = colnames(SimUplift[,3:7]), equal.intervals = TRUE, nb.group = 5, validation = FALSE)
features
```

---

**BinUplift**  
*Univariate quantization*

Description

Univariate optimal partitionning for Uplift Models. The algorithm quantizes a single variable into bins with significantly different observed uplift.

Usage

```r
BinUplift(data, treat, outcome, x, n.split = 10, alpha = 0.05, n.min = 30)
```
**BinUplift**

**Arguments**

- **data**: a data frame containing the treatment, the outcome and the predictor to quantize.
- **treat**: name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
- **outcome**: name of a binary response (numeric) vector (coded as 0/1).
- **x**: name of the explanatory variable to quantize.
- **n.split**: number of splits to test at each node. For continuous explanatory variables only (must be > 0). If n.split = 10, the test will be executed at each decile of the variable.
- **alpha**: significance level of the statistical test (must be between 0 and 1).
- **n.min**: minimum number of observations per child node.

**Value**

- **out.tree**: Descriptive statistics for the different nodes of the tree

**Author(s)**

Mouloud Belbahri

**References**


**See Also**

predict.BinUplift

**Examples**

```r
library(tools4uplift)
data("SimUplift")
binX1 <- BinUplift(data = SimUplift, treat = "treat", outcome = "y", x = "X1",
                  n.split = 100, alpha = 0.01, n.min = 30)
```
BinUplift2d  

Bivariate quantization

Description

A non-parametric heat map representing the observed uplift in rectangles that explore a bivariate dimension space. The function also returns the individual uplift based on the heatmap.

Usage

BinUplift2d(data, var1, var2, treat, outcome, valid = NULL, n.split = 3, n.min = 30, plotit = FALSE, nb.col = 20)

Arguments

data  
a data frame containing uplift models variables.

var1  
x-axis variable name. Represents the first dimension of interest.

var2  
y-axis variable name. Represents the second dimension of interest.

treat  
name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).

outcome  
name of a binary response (numeric) vector (coded as 0/1).

valid  
a validation data frame containing uplift models variables.

n.split  
the number of intervals to consider per explanatory variable. Must be an integer > 1.

n.min  
minimum number of observations per group (treatment and control) within each rectangle. Must be an integer > 0.

plotit  
if TRUE, a heatmap of observed uplift per rectangle is plotted.

nb.col  
number of colors for the heatmap. Default is 20. Must be an integer and should greater than n.split for better visualization.

Value

returns an augmented dataset with Uplift_var1_var2 variable representing a predicted uplift for each observation based on the rectangle it belongs to. The function also plots a heat map of observed uplifts.

Author(s)

Mouloud Belbahri

References

Examples

```r
library(tools4uplift)
data("SimUplift")

heatmap <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")
```

---

**DualUplift**

Two-model estimator

**Description**

Fit the two-model uplift model estimator.

**Usage**

```r
## S3 method for class 'formula'
DualUplift(formula, treat, data, ...)

## Default S3 method:
DualUplift(data, treat, outcome, predictors, ...)
```

**Arguments**

- `data, formula` a data frame containing the treatment, the outcome and the predictors or a formula describing the model to be fitted.
- `treat` name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
- `outcome` name of a binary response (numeric) vector (coded as 0/1).
- `predictors` a vector of names representing the explanatory variables to include in the model.
- `...` additional arguments (other than formula, family, and data) to be passed to glm function for each sub-model.

**Value**

- `model0` Fitted model for control group
- `model1` Fitted model for treatment group

**Author(s)**

Mouloud Belbahri
References


Examples

```r
library(tools4uplift)
data("SimUplift")
fit <- DualUplift(SimUplift, "treat", "y", predictors = colnames(SimUplift[, 3:12]))
print(fit)
summary(fit)
```

InterUplift

Interaction estimator

Description

Fit the interaction uplift model estimator.

Usage

```r
## S3 method for class 'formula'
InterUplift(formula, treat, data, ...)

## Default S3 method:
InterUplift(data, treat, outcome, predictors, input = "all", ...)
```

Arguments

data, formula  a data frame containing the treatment, the outcome and the predictors or a formula describing the model to be fitted.
treat  name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
outcome  name of a binary response (numeric) vector (coded as 0/1).
predictors  a vector of names representing the explanatory variables to include in the model.
input  an option for predictors argument. If "all" (default), the model assumes that the model has to create the interaction of all variables with treat. If "best", the model assumes that the predictors vector is the output of the BestFeatures function.
...  additional arguments (other than formula, family, and data) to be passed to glm function for the interaction model.
LassoPath

Value
an interaction model

Author(s)
Mouloud Belbahri

References
database marketing. ACM SIGKDD Explorations Newsletter, Vol. 4(2), 78-86.

Examples

library(tools4uplift)
data("SimUplift")

fit <- InterUplift(SimUplift, "treat", "y", colnames(SimUplift[, 3:12]))

| LassoPath | LASSO path for the penalized logistic regression |

Description
Fit an interaction uplift model via penalized maximum likelihood. The regularization path is computed for the lasso penalty at a grid of values for the regularization constant.

Usage
LassoPath(data, formula)

Arguments
data a data frame containing the treatment, the outcome and the predictors.
formula an object of class "formula" (or one that can be coerced to that class): a symbolic
description of the model to be fitted.

Value
a dataframe containing the coefficients values and the number of nonzeros coefficients for different values of lambda.

Author(s)
Mouloud Belbahri
References


See Also

BestFeatures, glmnet

Examples

#See glmnet() from library(glmnet) for more information

lines.PerformanceUplift

Qini curve

Description

Curve of the function Qini, the incremental observed uplift with respect to predicted uplift sorted from the highest to the lowest.

Usage

## S3 method for class 'PerformanceUplift'
lines(x, ...)

Arguments

x a table that must be the output of PerformanceUplift function.

... additional plot arguments.

Value

a Qini curve and the associated Qini coefficient

Author(s)

Mouloud Belbahri

References


PerformaceUplift

See Also
PerformanceUplift

Examples

library(tools4uplift)
data("SimUplift")

model1 <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")
perf1 <- PerformanceUplift(data = model1, treat = "treat",
outcome = "y", prediction = "Uplift_X1_X2",
equal.intervals = TRUE, nb.group = 3)

model2 <- BinUplift2d(SimUplift, "X3", "X4", "treat", "y")
perf2 <- PerformanceUplift(data = model2, treat = "treat",
outcome = "y", prediction = "Uplift_X3_X4",
equal.intervals = TRUE, nb.group = 3)

plot(perf1, type='b')
lines(perf2, type='b', col='red')

PerformanceUplift

Performance of an uplift estimator

Description
Table of performance of an uplift model. This table is used in order to visualize the performance of an uplift model and to compute the Qini coefficient.

Usage
PerformanceUplift(data, treat, outcome, prediction, nb.group = 10,
equal.intervals = TRUE, rank.precision = 2)

Arguments

data a data frame containing the response, the treatment and predicted uplift.
treat a binary (numeric) vector representing the treatment assignment (coded as 0/1).
outcome a binary response (numeric) vector (coded as 0/1).
prediction a predicted uplift (numeric) vector to sort the observations from highest to lowest uplift.
 nb.group if equal.intervals is set to true, the number of groups of equal observations in which to partition the data set to show results.
equal.intervals
flag for using equal intervals (with equal number of observations) or the true ranking quantiles which result in an unequal number of observations in each group.

rank.precision
precision for the ranking quantiles. Must be 1 or 2. If 1, the ranking quantiles will be rounded to the first decimal. If 2, to the second decimal.

Value
a table with descriptive statistics related to an uplift model estimator.

Author(s)
Mouloud Belbahri

References

See Also
QiniArea

Examples

```r
library(tools4uplift)
data("SimUplift")
model1 <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")
perf1 <- PerformanceUplift(data = model1, treat = "treat", outcome = "y", prediction = "Uplift_X1_X2", equal.intervals = TRUE, nb.group = 3)
print(perf1)
```

plot.PerformanceUplift

Qini curve

Description
Curve of the function Qini, the incremental observed uplift with respect to predicted uplift sorted from the highest to the lowest.
Usage

```r
## S3 method for class 'PerformanceUplift'
plot(x, ...)
```

Arguments

- `x`: a table that must be the output of `PerformanceUplift` function.
- `...`: additional plot arguments.

Value

- a Qini curve and the associated Qini coefficient

Author(s)

Mouloud Belbahri

References


See Also

- `PerformanceUplift`

Examples

```r
library(tools4uplift)
data("SimUplift")

modell <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")
perf1 <- PerformanceUplift(data = modell, treat = "treat",
outcome = "y", prediction = "Uplift_X1_X2",
equal.intervals = TRUE, nb.group = 3)

plot(perf1, type='b')
```
Description

Predictions from the univariate quantization method, i.e. this function transforms a continuous variable into a categorical one.

Usage

```r
## S3 method for class 'BinUplift'
predict(object, newdata, ...)
```

Arguments

- `object`: an object of class `BinUplift`, as that created by the function `BinUplift`.
- `newdata`: the variable that was quantized in `object`.
- `...`: additional arguments to be passed to `cut` function.

Value

a quantized variable

Author(s)

Mouloud Belbahri

References


See Also

`BinUplift`

Examples

```r
library(tools4uplift)
data("SimUplift")

binX1 <- BinUplift(data = SimUplift, treat = "treat", outcome = "y", x = "X1",
                  n.split = 100, alpha = 0.01, n.min = 30)

quantizedX1 <- predict(binX1, SimUplift$X1)
```
predict.DualUplift

Predictions from a two-model estimator

Description

Predictions from the two-model uplift model estimator with associated model performance.

Usage

## S3 method for class 'DualUplift'
predict(object, newdata, ...)

Arguments

- **object**: an object of class DualUplift, as that created by the function DualUplift.
- **newdata**: a data frame containing the treatment, the outcome and the predictors of observations at which predictions are required.
- **...**: additional arguments to be passed to predict.glm function for each sub-model.

Value

a vector of predicted uplift

Author(s)

Mouloud Belbahri

References


See Also

DualUplift

Examples

```r
library(tools4uplift)
data("SimUplift")
fit <- DualUplift(SimUplift, "treat", "y", predictors = colnames(SimUplift[, 3:12]))
pred <- predict(fit, SimUplift)
```
predict.InterUplift  Predictions from an interaction estimator

Description

Predictions from the interaction uplift model estimator with associated model performance.

Usage

```r
## S3 method for class 'InterUplift'
predict(object, newdata, treat, ...)
```

Arguments

- **object**: an object of class InterUplift, as that created by the function InterUplift.
- **newdata**: a data frame containing the treatment, the outcome and the predictors of observations at which predictions are required.
- **treat**: name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
- **...**: additional arguments to be passed to `predict.glm` function for the interaction model.

Value

a vector of predicted uplift

Author(s)

Mouloud Belbahri

References


See Also

InterUplift
Examples

```r
library(tools4uplift)
data("SimUplift")
fit <- InterUplift(SimUplift, "treat", "y", colnames(SimUplift[, 3:12]))
pred <- predict(fit, SimUplift, "treat")
```

<table>
<thead>
<tr>
<th>QiniArea</th>
<th>Qini coefficient</th>
</tr>
</thead>
</table>

Description

Computes the area under the Qini curve.

Usage

`QiniArea(x, adjusted=FALSE)`

Arguments

- `x`: a table that must be the output of `PerformanceUplift` function.
- `adjusted`: if TRUE, returns the Qini coefficient adjusted by the Kendall’s uplift rank correlation.

Value

the Qini or the adjusted Qini coefficient

Author(s)

Mouloud Belbahri

References


See Also

`PerformanceUplift`
Examples

```r
library(tools4uplift)
data("SimUplift")

model <- BinUplift2d(SimUplift, "X1", "X2", "treat", "y")

# performance of the heat map uplift estimation on the training dataset
perf <- PerformanceUplift(data = model, treat = "treat",
                           outcome = "y", prediction = "Uplift_X1_X2",
                           equal.intervals = TRUE, nb.group = 5)
QiniArea(perf)
```

---

**qLHS**

*Qini-based uplift regression*

---

**Description**

A Qini-based LHS (Latin Hypercube Sampling) uplift model.

**Usage**

```r
qLHS(data, treat, outcome, predictors,
     lhs_points = 50, lhs_range = 1,
     adjusted = TRUE, rank.precision = 2, equal.intervals = FALSE,
     nb.group = 10, validation = TRUE, p = 0.3)
```

**Arguments**

- `data` a data frame containing the treatment, the outcome and the predictors.
- `treat` name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
- `outcome` name of a binary response (numeric) vector (coded as 0/1).
- `predictors` a vector of names representing the predictors to consider in the model.
- `lhs_points` number of LHS points to sample for each regularization constant.
- `lhs_range` a multiplicative scalar that controls the variance of the LHS search - Default is 1, the LHS procedure samples points uniformly with variance equal to the variance of the maximum likelihood estimator.
- `adjusted` if TRUE, the adjusted Qini coefficient is used instead of the Qini coefficient.
- `rank.precision` precision for the ranking quantiles to compute the Qini coefficient. Must be 1 or 2. If 1, the ranking quantiles will be rounded to the first decimal. If 2, to the second decimal.
equal.intervals
flag for using equal intervals (with equal number of observations) or the true ranking quantiles which result in an unequal number of observations in each group to compute the Qini coefficient.

nb.group
the number of groups for computing the Qini coefficient if equal.intervals is TRUE - Default is 10.

validation
if TRUE, the best LHS model is selected based on cross-validation - Default is TRUE.

p
if validation is TRUE, the desired proportion for the validation set. p is a value between 0 and 1 expressed as a decimal, it is set to be proportional to the number of observations per group - Default is 0.3.

Details
The regularization parameter is chosen based on the interaction uplift model that maximizes the Qini coefficient of the LHS search.

Value
the models with LHS coefficients of class InterUplift.

Author(s)
Mouloud Belbahri

References

Examples

```r
library(tools4uplift)
data("SimUplift")
upliftLHS <- qLHS(data = SimUplift, treat = "treat", outcome = "y",
predictors = colnames(SimUplift[,3:7]), lhs_points = 5,
lhs_range = 1, adjusted = TRUE, equal.intervals = TRUE,
nb.group = 5, validation = FALSE)
```
SimUplift

Synthetic data for uplift modeling

Description
The synthetic data contains 20 predictors, a treatment allocation variable and an outcome binary variable. This dataset is used in the package examples.

Usage
data("SimUplift")

Format
A data frame with 1000 observations on the following 22 variables.
y  a binary response vector
treat  a binary treatment allocation vector
X1  a numeric vector
X2  a numeric vector
X3  a numeric vector
X4  a numeric vector
X5  a numeric vector
X6  a numeric vector
X7  a numeric vector
X8  a numeric vector
X9  a numeric vector
X10  a numeric vector
X11  a numeric vector
X12  a numeric vector
X13  a numeric vector
X14  a numeric vector
X15  a numeric vector
X16  a numeric vector
X17  a numeric vector
X18  a numeric vector
X19  a numeric vector
X20  a numeric vector

Examples
data("SimUplift")
SplitUplift

*Split data with respect to uplift distribution*

**Description**

Split a dataset into training and validation subsets with respect to the uplift sample distribution.

**Usage**

```r
SplitUplift(data, p, group)
```

**Arguments**

- `data`: a data frame of interest that contains at least the response and the treatment variables.
- `p`: The desired sample size. `p` is a value between 0 and 1 expressed as a decimal, it is set to be proportional to the number of observations per group.
- `group`: Your grouping variables. Generally, for uplift modelling, this should be a vector of treatment and response variables names, e.g. c("treat", "y").

**Value**

- `train`: a training data frame of `p` percent
- `valid`: a validation data frame of `1-p` percent

**Author(s)**

Mouloud Belbahri

**References**


**Examples**

```r
library(tools4uplift)
data("SimUplift")

split <- SplitUplift(SimUplift, 0.8, c("treat", "y"))
train <- split[[1]]
valid <- split[[2]]
```
Description

Computes the observed uplift per category and creates a barplot.

Usage

UpliftPerCat(data, treat, outcome, x, ...)

Arguments

data a data frame containing the treatment, the outcome and the variable of interest.
treat name of a binary (numeric) vector representing the treatment assignment (coded as 0/1).
outcome name of a binary response (numeric) vector (coded as 0/1).
x name of the explanatory variable of interest.
... extra parameters for the barplot.

Value

returns a barplot representing the uplift per category.

Author(s)

Mouloud Belbahri

Examples

library(tools4uplift)
data("SimUplift")

binX1 <- BinUplift(data = SimUplift, treat = "treat", outcome = "y", x = "X1",
n.split = 100, alpha = 0.01, n.min = 30)

SimUplift$quantizedX1 <- predict(binX1, SimUplift$X1)
UpliftPerCat(data = SimUplift, treat = "treat", outcome = "y",
            x = "quantizedX1", xlab='Quantized X1', ylab='Uplift',
ylim=c(-1,1))
Index

* LASSO
  BestFeatures, 3
  LassoPath, 9
  qLHS, 18
* Qini
  BestFeatures, 3
  lines.PerformanceUplift, 10
  plot.PerformanceUplift, 12
  QiniArea, 17
  qLHS, 18
* barplot
  UpliftPerCat, 22
* datasets
  SimUplift, 20
* heatmap
  BinUplift2d, 6
* lhs
  qLHS, 18
* logistic
  BestFeatures, 3
  qLHS, 18
* performance
  barplot.PerformanceUplift, 2
  lines.PerformanceUplift, 10
  PerformanceUplift, 11
  plot.PerformanceUplift, 12
  QiniArea, 17
* prediction
  predict.DualUplift, 15
  predict.InterUplift, 16
* quantization
  BinUplift, 4
  predict.BinUplift, 14
* sampling
  SplitUplift, 21
* tree
  BinUplift, 4
* uplift
  BinUplift2d, 6