Package 'MLmetrics'

May 13, 2016

Type Package

Title Machine Learning Evaluation Metrics

Version 1.1.1

Description A collection of evaluation metrics, including loss, score and utility functions, that measure regression, classification and ranking performance.

URL http://github.com/yanyachen/MLmetrics

BugReports http://github.com/yanyachen/MLmetrics/issues

Depends R (>= 2.10)

Imports stats, utils, ROCR

Suggests e1071

License GPL-2

LazyData true

RoxygenNote 5.0.1

NeedsCompilation no

Author Yachen Yan [aut, cre]

Maintainer Yachen Yan <yanyachen21@gmail.com>

Repository CRAN

Date/Publication 2016-05-13 23:57:26

R topics documented:

`Accuracy` *Accuracy*

Description

Compute the accuracy classification score.

Usage

```
Accuracy(y_pred, y_true)
```

Arguments

`y_pred`	Predicted labels vector, as returned by a classifier
`y_true`	Ground truth (correct) 0-1 labels vector

Value

Accuracy

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
Accuracy(y_pred = pred, y_true = mtcars$vs)
```

Area_Under_Curve *Calculate the Area Under the Curve*

Description

Calculate the area under the curve.

Usage

```
Area_Under_Curve(x, y, method = c("trapezoid", "step", "spline"),
  na.rm = FALSE)
```

Arguments

x	the x-points of the curve
y	the y-points of the curve
method	can be "trapezoid" (default), "step" or "spline"
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds

Value

Area Under the Curve (AUC)

Examples

```
x <- seq(0, pi, length.out = 200)
plot(x = x, y = sin(x), type = "l")
Area_Under_Curve(x = x, y = sin(x), method = "trapezoid", na.rm = TRUE)
```

AUC *Area Under the Receiver Operating Characteristic Curve (ROC AUC)*

Description

Compute the Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

Usage

```
AUC(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Area Under the ROC Curve (ROC AUC)

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
AUC(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

ConfusionMatrix *Confusion Matrix*

Description

Compute confusion matrix to evaluate the accuracy of a classification.

Usage

```
ConfusionMatrix(y_pred, y_true)
```

Arguments

y_pred	Predicted labels vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

a table of Confusion Matrix

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
ConfusionMatrix(y_pred = pred, y_true = mtcars$vs)
```

F1_Score *F1 Score*

Description

Compute the F1 Score.

Usage

```
F1_Score(y_true, y_pred, positive = NULL)
```

Arguments

y_true	Ground truth (correct) 0-1 labels vector
y_pred	Predicted labels vector, as returned by a classifier
positive	An optional character string for the factor level that corresponds to a "positive" result

Value

F1 Score

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
F1_Score(y_pred = pred, y_true = mtcars$vs, positive = "0")
F1_Score(y_pred = pred, y_true = mtcars$vs, positive = "1")
```

FBeta_Score *F-Beta Score*

Description

Compute the F-Beta Score

Usage

```
FBeta_Score(y_true, y_pred, positive = NULL, beta = 1)
```

Arguments

y_true	Ground truth (correct) 0-1 labels vector
y_pred	Predicted labels vector, as returned by a classifier
positive	An optional character string for the factor level that corresponds to a "positive" result
beta	Weight of precision in harmonic mean

Value

F-Beta Score

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
FBeta_Score(y_pred = pred, y_true = mtcars$vs, positive = "0", beta = 2)
FBeta_Score(y_pred = pred, y_true = mtcars$vs, positive = "1", beta = 2)
```

GainAUC *Area Under the Gain Chart*

Description

Compute the Area Under the Gain Chart from prediction scores.

Usage

```
GainAUC(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Area Under the Gain Chart

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
GainAUC(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

Gini *Gini Coefficient*

Description

Compute the Gini Coefficient.

Usage

```
Gini(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Gini Coefficient

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
Gini(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

KS_Stat *Kolmogorov-Smirnov Statistic*

Description

Compute the Kolmogorov-Smirnov statistic.

Usage

```
KS_Stat(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Kolmogorov-Smirnov statistic

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
KS_Stat(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

LiftAUC *Area Under the Lift Chart*

Description

Compute the Area Under the Lift Chart from prediction scores.

Usage

```
LiftAUC(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Area Under the Lift Chart

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
LiftAUC(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

LogLoss *Log loss / Cross-Entropy Loss*

Description

Compute the log loss/cross-entropy loss.

Usage

```
LogLoss(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Log loss/Cross-Entropy Loss

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
LogLoss(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

MAE *Mean Absolute Error Loss*

Description

Compute the mean absolute error regression loss.

Usage

```
MAE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Mean Absolute Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
MAE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

MAPE *Mean Absolute Percentage Error Loss*

Description

Compute the mean absolute percentage error regression loss.

Usage

```
MAPE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Mean Absolute Percentage Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
MAPE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

MedianAE *Median Absolute Error Loss*

Description

Compute the median absolute error regression loss.

Usage

```
MedianAE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Median Absolute Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
MedianAE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

MedianAPE *Median Absolute Percentage Error Loss*

Description

Compute the Median absolute percentage error regression loss.

Usage

```
MedianAPE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Median Absolute Percentage Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
MedianAPE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

MLmetrics *MLmetrics: Machine Learning Evaluation Metrics*

Description

A collection of evaluation metrics, including loss, score and utility functions, that measure regression and classification performance.

MSE *Mean Square Error Loss*

Description

Compute the mean squared error regression loss.

Usage

```
MSE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Mean Square Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
MSE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

MultiLogLoss *Multi Class Log Loss*

Description

Compute the multi class log loss.

Usage

```
MultiLogLoss(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities matrix, as returned by a classifier
y_true	Ground truth (correct) labels vector or a matrix of correct labels indicating by 0-1, same format as probabilities matrix

Value

Multi Class Log Loss

Examples

```
data(iris)
svm.model <- e1071::svm(Species~., data = iris, probability = TRUE)
pred <- predict(svm.model, iris, probability = TRUE)
MultiLogLoss(y_true = iris$Species, y_pred = attr(pred, "probabilities"))
```

NormalizedGini *Normalized Gini Coefficient*

Description

Compute the Normalized Gini Coefficient.

Usage

```
NormalizedGini(y_pred, y_true)
```

Arguments

y_pred	Predicted labels vector, as returned by a model
y_true	Ground truth (correct) labels vector

Value

Normalized Gini Coefficient

Examples

```
d_AD <- data.frame(treatment = gl(3,3), outcome = gl(3,1,9),
                   counts = c(18,17,15,20,10,20,25,13,12))
glm_poisson <- glm(counts ~ outcome + treatment,
                   family = poisson(link = "log"), data = d_AD)
NormalizedGini(y_pred = glm_poisson$fitted.values, y_true = d_AD$counts)
```

Poisson_LogLoss *Poisson Log loss*

Description

Compute the log loss/cross-entropy loss.

Usage

```
Poisson_LogLoss(y_pred, y_true)
```

Arguments

y_pred	Predicted labels vector, as returned by a model
y_true	Ground truth (correct) labels vector

Value

Log loss/Cross-Entropy Loss

Examples

```
d_AD <- data.frame(treatment = gl(3,3), outcome = gl(3,1,9),
                   counts = c(18,17,15,20,10,20,25,13,12))
glm_poisson <- glm(counts ~ outcome + treatment,
                   family = poisson(link = "log"), data = d_AD)
Poisson_LogLoss(y_pred = glm_poisson$fitted.values, y_true = d_AD$counts)
```

PRAUC *Area Under the Precision-Recall Curve (PR AUC)*

Description

Compute the Area Under the Precision-Recall Curve (PR AUC) from prediction scores.

Usage

```
PRAUC(y_pred, y_true)
```

Arguments

y_pred	Predicted probabilities vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Area Under the PR Curve (PR AUC)

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
PRAUC(y_pred = logreg$fitted.values, y_true = mtcars$vs)
```

Precision *Precision*

Description

Compute the precision score.

Usage

```
Precision(y_true, y_pred, positive = NULL)
```

Arguments

y_true	Ground truth (correct) 0-1 labels vector
y_pred	Predicted labels vector, as returned by a classifier
positive	An optional character string for the factor level that corresponds to a "positive" result

Value

Precision

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
Precision(y_pred = pred, y_true = mtcars$vs, positive = "0")
Precision(y_pred = pred, y_true = mtcars$vs, positive = "1")
```

R2_Score *R-Squared (Coefficient of Determination) Regression Score*

Description

Compute the R-Squared (Coefficient of Determination) Regression Score.

Usage

```
R2_Score(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

R^2 Score

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
R2_Score(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

RAE *Relative Absolute Error Loss*

Description

Compute the relative absolute error regression loss.

Usage

```
RAE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Relative Absolute Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
RAE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

`Recall` *Recall*

Description

Compute the recall score.

Usage

```
Recall(y_true, y_pred, positive = NULL)
```

Arguments

`y_true`	Ground truth (correct) 0-1 labels vector
`y_pred`	Predicted labels vector, as returned by a classifier
`positive`	An optional character string for the factor level that corresponds to a "positive" result

Value

Recall

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
Recall(y_pred = pred, y_true = mtcars$vs, positive = "0")
Recall(y_pred = pred, y_true = mtcars$vs, positive = "1")
```

RMSE *Root Mean Square Error Loss*

Description

Compute the root mean squared error regression loss.

Usage

```
RMSE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Root Mean Square Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
RMSE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

RMSLE *Root Mean Squared Logarithmic Error Loss*

Description

Compute the root mean squared logarithmic error regression loss.

Usage

```
RMSLE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Root Mean Squared Logarithmic Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
RMSLE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

RMSPE *Root Mean Square Percentage Error Loss*

Description

Compute the root mean squared percentage error regression loss.

Usage

```
RMSPE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Root Mean Squared Percentage Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
RMSPE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

RRSE *Root Relative Squared Error Loss*

Description

Compute the root relative squared error regression loss.

Usage

```
RRSE(y_pred, y_true)
```

Arguments

y_pred	Estimated target values vector
y_true	Ground truth (correct) target values vector

Value

Root Relative Squared Error Loss

Examples

```
data(cars)
reg <- lm(log(dist) ~ log(speed), data = cars)
RRSE(y_pred = exp(reg$fitted.values), y_true = cars$dist)
```

`Sensitivity` *Sensitivity*

Description

Compute the sensitivity score.

Usage

```
Sensitivity(y_true, y_pred, positive = NULL)
```

Arguments

`y_true`	Ground truth (correct) 0-1 labels vector
`y_pred`	Predicted labels vector, as returned by a classifier
`positive`	An optional character string for the factor level that corresponds to a "positive" result

Value

Sensitivity

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
Sensitivity(y_pred = pred, y_true = mtcars$vs, positive = "0")
Sensitivity(y_pred = pred, y_true = mtcars$vs, positive = "1")
```

Specificity *Specificity*

Description

Compute the specificity score.

Usage

```
Specificity(y_true, y_pred, positive = NULL)
```

Arguments

y_true	Ground truth (correct) 0-1 labels vector
y_pred	Predicted labels vector, as returned by a classifier
positive	An optional character string for the factor level that corresponds to a "positive" result

Value

Specificity

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
Specificity(y_pred = pred, y_true = mtcars$vs, positive = "0")
Specificity(y_pred = pred, y_true = mtcars$vs, positive = "1")
```

ZeroOneLoss *Normalized Zero-One Loss (Classification Error Loss)*

Description

Compute the normalized zero-one classification loss.

Usage

```
ZeroOneLoss(y_pred, y_true)
```

Arguments

y_pred	Predicted labels vector, as returned by a classifier
y_true	Ground truth (correct) 0-1 labels vector

Value

Zero-One Loss

Examples

```
data(cars)
logreg <- glm(formula = vs ~ hp + wt,
              family = binomial(link = "logit"), data = mtcars)
pred <- ifelse(logreg$fitted.values < 0.5, 0, 1)
ZeroOneLoss(y_pred = pred, y_true = mtcars$vs)
```

Index