

# Package ‘car2’

June 2, 2015

**Type** Package

**Title** Extends capabilities of 'car' to include companion functions for logistic regression

**Version** 0.1

**Date** 2015-05-26

**Author** Selva Prabhakaran

**Maintainer** Selva Prabhakaran <selva86@gmail.com>

**Description** More about what it does (maybe more than one line)

**License** GPL (>= 2)

**LazyData** TRUE

**LazyLoad** yes

**Depends** car

**Import** ggplot2

## R topics documented:

aROC . . . . .	2
calcConcordance . . . . .	2
confusionMatrix . . . . .	3
kappaCohen . . . . .	4
misClassError . . . . .	5
plotROC . . . . .	6
sensitivity . . . . .	6
somersD . . . . .	7
specificity . . . . .	8
youdensIndex . . . . .	9

<b>Index</b>	<b>10</b>
--------------	-----------

aROC	aROC
<p><b>Description</b></p> <p>Calculate the area under ROC curve statistic for a given logit model.</p> <p><b>Usage</b></p> <pre>aROC(logitMod)</pre> <p><b>Arguments</b></p> <p>logitMod      A logit model</p> <p><b>Details</b></p> <p>For a given logit model, the area under the ROC curve shows how well the model performs at capturing the false events and false non-events. An best case model will have an area of 1. However that would be unrealistic, so the closer the aROC to 1, the better is the model.</p> <p><b>Value</b></p> <p>The area under the ROC curve for a given logit model.</p> <p><b>Author(s)</b></p> <p>Selva Prabhakaran</p> <p><b>Examples</b></p> <pre>accept &lt;- c (1, 0, 1, 0, 1, 1, 0, 0, 0,1, 0, 1, 0, 0, 1) acad   &lt;- c (66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76) sports &lt;- c (2.6,4.6,4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9) rank   &lt;- c (3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1) inputData &lt;- data.frame (accept, acad , sports, rank) # assemble the data frame logitModel &lt;- glm(accept ~ ., family="binomial", data = inputData ) aROC(logitMod=logitModel)</pre>	
calcConcordance	calcConcordance

## Description

Calculate concordance and discordance percentages for a logit model

## Usage

```
calcConcordance(logitMod)
```

**Arguments**

logitMod            A logit model

**Details**

Calculate the percentage of concordant and discordant pairs for a given logit model.

**Value**

a list containing percentage of concordant pairs, percentage discordant pairs, percentage ties and No. of pairs.

**Author(s)**

Selva Prabhakaran

**Examples**

```
accept <- c (1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad   <- c (66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c (2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank   <- c (3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame (accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
calcConcordance(logitModel)
```

---

confusionMatrix	<i>confusionMatrix</i>
-----------------	------------------------

---

**Description**

Calculate the confusion matrix for the fitted values for a logistic regression model.

**Usage**

```
confusionMatrix(logitMod, threshold = 0.5)
```

**Arguments**

logitMod            A logit model

threshold           If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

**Details**

For a given logit model, the confusion matrix showing the count of predicted events and non-events against actual events and non events.

**Value**

For a given logit model, returns the confusion matrix showing the count of predicted events and non-events against actual events and non events.

**Author(s)**

Selva Prabhakaran

**Examples**

```
accept <- c(1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad <- c(66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c(2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank <- c(3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame(accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
confusionMatrix(logitMod=logitModel)
```

kappaCohen

*kappaCohen***Description**

Calculate the Cohen's kappa statistic for a given logit model.

**Usage**

```
kappaCohen(logitMod, threshold = 0.5)
```

**Arguments**

logitMod	A logit model
threshold	If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

**Details**

For a given logit model, Cohen's kappa is calculated. Cohen's kappa is calculated as (probability of agreement - probability of expected) / (1-(probability of expected))

**Value**

The Cohen's kappa of the logit model

**Author(s)**

Selva Prabhakaran

**Examples**

```
accept <- c(1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad <- c(66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c(2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank <- c(3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame(accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
kappaCohen(logitMod=logitModel)
```

---

misClassError	<i>misClassError</i>
---------------	----------------------

---

## Description

Calculate the percentage misclassification error for this logit model's fitted values.

## Usage

```
misClassError(logitMod, threshold = 0.5)
```

## Arguments

logitMod	A logit model
threshold	If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

## Details

For a given logit model, misclassification error is the number of mismatches between the predicted and actuals direction of the binary y variable.

## Value

The misclassification error, which tells what proportion of predicted direction did not match with the actuals.

## Author(s)

Selva Prabhakaran

## Examples

```
accept <- c(1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad <- c(66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c(2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank <- c(3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame(accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
misClassError(logitMod=logitModel)
```

---

plotROC

*plotROC*


---

### Description

Plot the Receiver Operating Characteristics(ROC) Curve based on ggplot2

### Usage

```
plotROC(logitMod)
```

### Arguments

logitMod            A logit model

### Details

For a given logit model, A ROC curve is plotted using the ggplot2 framework along the the area under the curve.

### Value

Plots the ROC curve

### Author(s)

Selva Prabhakaran

### Examples

```
accept <- c (1, 0, 1, 0, 1, 1, 0, 0, 0,1, 0, 1, 0, 0, 1)
acad   <- c (66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c (2.6,4.6,4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank   <- c (3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame (accept, acad , sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData )
plotROC(logitMod=logitModel)
```

---

sensitivity

*sensitivity*


---

### Description

Calculate the sensitivity for a given logit model.

### Usage

```
sensitivity(logitMod, threshold = 0.5)
```

**Arguments**

logitMod	A logit model
threshold	If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

**Details**

For a given logit model, sensitivity is defined as number of observations with the event AND predicted to have the event divided by the number of observations with the event. It can be used as an indicator to gauge how sensitive is your model in detecting the occurrence of events, especially when you are not so concerned about predicting the non-events as true.

**Value**

The sensitivity of the logit model, which is, the number of observations with the event AND predicted to have the event divided by the number of observations with the event.

**Author(s)**

Selva Prabhakaran

**Examples**

```
accept <- c(1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad    <- c(66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports  <- c(2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank    <- c(3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame(accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
sensitivity(logitMod=logitModel)
```

---

somersD

*somersD*


---

**Description**

Calculate the Somers D statistic for a given logit model

**Usage**

```
somersD(logitMod)
```

**Arguments**

logitMod	A logit model
----------	---------------

**Details**

For a given logit model, Somer's D is calculated as the number of concordant pairs less number of discordant pairs divided by total number of pairs.

**Value**

The Somers D statistic, which tells how many more concordant than discordant pairs exist divided by total number of pairs.

**Author(s)**

Selva Prabhakaran

**Examples**

```
accept <- c(1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad <- c(66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c(2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank <- c(3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame(accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
somersD(logitMod=logitModel)
```

---

specificity

*specificity*


---

**Description**

Calculate the specificity for a given logit model.

**Usage**

```
specificity(logitMod, threshold = 0.5)
```

**Arguments**

logitMod	A logit model
threshold	If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

**Details**

For a given logit model, specificity is defined as number of observations without the event AND predicted to not have the event divided by the number of observations without the event. Specificity is particularly useful when you are extra careful not to predict a non event as an event, like in spam detection where you dont want to classify a genuine mail as spam(event) where it may be somewhat ok to occasionally classify a spam as a genuine mail(a non-event).

**Value**

The specificity of the logit model, which is, the number of observations without the event AND predicted to not have the event divided by the nummber of observations without the event.

**Author(s)**

Selva Prabhakaran



**Examples**

```

accept <- c (1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad   <- c (66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c (2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank   <- c (3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame (accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
specificity(logitMod=logitModel)

```

youdensIndex

*youdensIndex***Description**

Calculate the specificity for a given logit model.

**Usage**

```
youdensIndex(logitMod, threshold = 0.5)
```

**Arguments**

logitMod	A logit model
threshold	If predicted value is above the threshold, it will be considered as an event (1), else it will be a non-event (0). Defaults to 0.5.

**Details**

For a given logit model, Youden's index is calculated as sensitivity + specificity - 1

**Value**

The youdensIndex of the logit model, which is calculated as Sensitivity + Specificity - 1

**Author(s)**

Selva Prabhakaran

**Examples**

```

accept <- c (1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1)
acad   <- c (66, 60, 80, 60, 52, 60, 47, 90, 75, 35, 46, 75, 66, 54, 76)
sports <- c (2.6, 4.6, 4.5, 3.3, 3.13, 4, 1.9, 3.5, 1.2, 1.8, 1, 5.1, 3.3, 5.2, 4.9)
rank   <- c (3, 3, 1, 4, 4, 2, 4, 4, 4, 3, 3, 3, 2, 2, 1)
inputData <- data.frame (accept, acad, sports, rank) # assemble the data frame
logitModel <- glm(accept ~ ., family="binomial", data = inputData)
youdensIndex(logitMod=logitModel)

```

# Index

aROC, [2](#)

calcConcordance, [2](#)

confusionMatrix, [3](#)

kappaCohen, [4](#)

misClassError, [5](#)

plotROC, [6](#)

sensitivity, [6](#)

somersD, [7](#)

specificity, [8](#)

youdensIndex, [9](#)