Package: gamclass (via r-universe)

October 18, 2026

Type Package

Title Functions and Data for a Course on Modern Regression and Classification

Version 0.62.6

Date 2025-07-24

Maintainer John Maindonald <john@statsresearch.co.nz>

URL https://github.com/jhmaindonald/gamclass

LazyData true

Depends R (>= 3.5.0)

Suggests sp, kernlab, mlbench, car, mgcv, DAAG, MASS, knitr,prettydoc,rmarkdown,bookdown

Imports rpart, randomForest, lattice, latticeExtra, methods

VignetteBuilder knitr,rmarkdown,bookdown

Description Functions and data are provided that support a course that emphasizes statistical issues of inference and generalizability. The functions are designed to make it straightforward to illustrate the use of cross-validation, the training/test approach, simulation, and model-based estimates of accuracy. Methods considered are Generalized Additive Modeling, Linear and Quadratic Discriminant Analysis, Tree-based methods, and Random Forests.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Config/pak/sysreqs libjpeg-dev libpng-dev

Repository https://jhmaindonald.r-universe.dev

Date/Publication 2025-07-24 21:18:49 UTC

RemoteUrl https://github.com/jhmaindonald/gamclass

RemoteRef HEAD

RemoteSha 31bf21d41f63180b6b7bbaa373c66a05261c1b7d

Contents

gamclass-package . . . 2
addhlines . . . 3
airAccs . . . 4
bomregions2018 . . . 5
bronchitis . . . 7
bssBYcut . . . 8
compareModels . . . 10
confusion . . . 11
coralPval . . . 13
cvalues . . . 14
CVcluster . . . 14
CVgam . . . 16
eventCounts . . . 17
FARS . . . 18
fars2007 . . . 20
frontDeaths . . . 21
gamRF . . . 22
german . . . 24
greatLakesM . . . 25
ldaErr . . . 26
loti . . . 27
plotFars . . . 28
relDeaths . . . 29
RFcluster . . . 30
rfErr . . . 31
rpartErr . . . 32
simreg . . . 33
tabFarsDead . . . 34

Index **36**

`gamclass-package` *Functions and Data for a Course in Modern Regression*

Description

For purposes of this package, modern regression extends to include classification and multivariate exploration. A strong focus is on methods described in Wood (2017) <doi:10.1201/9781315370279>

Details

Functions are mostly designed to facilitate a variety of cross-validation and bootstrap calculations.

Author(s)

John Maindonald

Maintainer: jhmaindonald@gmail.com

References

Venables, W N, & Ripley, B D (2013). Modern applied statistics with S-PLUS. Springer Science & Business Media.

Wood, S N (2017) Generalized Additive Models: An Introduction with R (2nd edition). Chapman and Hall/CRC.

https://github.com/jhmaindonald/gamclass

`addhlines` *Add horizontal lines to plot.*

Description

This is designed for adding horizontal lines that show predicted values to a plot of observed values versus x-values, in `rpart` regression. Where predicted values change between two successive x-values lines are extended to the midway point. This reflects the way that `predict.rpart` handles predictions for new data.

Usage

```
addhlines(x, y, ...)
```

Arguments

`x`	Vector of predictor variable values.
`y`	Vector of predicted values.
`...`	Additional graphics parameters, for passing through to the `lines()` function.

Value

Lines are added to the current graph.

Author(s)

John Maindonald

Examples

```
x <- c(34, 18, 45, 18, 27, 24, 34, 20, 24, 28, 21, 18)
y <- c(14, 11, 12, 9, 4, 11, 6, 9, 4, 10, 9, 2)
hat <- c(10.5, 7.75, 10.5, 7.75, 7, 7, 10.5, 7.75, 7, 10.5, 7, 7.75)
plot(x, y)
addhlines(x, hat, lwd=2, col="gray")

## The function is currently defined as
function(x,y, ...){
  ordx <- order(x)
  xo <- x[ordx]
```

```
  yo <- y[ordx]
  breaks <- diff(yo)!=0
  xh <- c(xo[1],0.5*(xo[c(FALSE,breaks)]+xo[c(breaks, FALSE)]))
  yh <- yo[c(TRUE, breaks)]
  y3 <- x3 <- numeric(3*length(xh)-1)
  loc1 <- seq(from=1, to=length(x3), by=3)
  x3[loc1] <- xh
  x3[loc1+1]<- c(xh[-1], max(x))
  x3[loc1[-length(loc1)]+2] <- NA
  y3[loc1[-length(loc1)]+2] <- NA
  y3[loc1] <- yh
  y3[loc1+1] <- yh
  lines(x3,y3, ...)
}
```

`airAccs` *Aircraft Crash data*

Description

Aircraft Crash Data

Usage

```
data(airAccs)
```

Format

A data frame with 5666 observations on the following 7 variables.

`Date` Date of Accident

`location` Location of accident

`operator` Aircraft operator

`planeType` Aircraft type

`Dead` Number of deaths

`Aboard` Number aboard

`Ground` Deaths on ground

Details

For details of inclusion criteria, see https://www.planecrashinfo.com/database.htm

Source

https://www.planecrashinfo.com/database.htm

References

https://www.planecrashinfo.com/reference.htm

Examples

```
data(airAccs)
str(airAccs)
```

---

bomregions2018 *Australian and Related Historical Annual Climate Data, by Region*

Description

Australian regional temperature data, Australian regional rainfall data, and Annual SOI, are given for the years 1900-2018. The regional rainfall and temperature data are area-weighted averages for the respective regions. The Southern Oscillation Index (SOI) is the difference in barometric pressure at sea level between Tahiti and Darwin. Data through to 2021, including also the Indian Ocean Dipole, is available in the file `DAAG::bomregions2021` .

Usage

```
data("bomregions2018")
```

Format

This data frame contains the following columns:

Year Year

seAVt Southeastern region average temperature (degrees C)

southAVt Southern temperature

eastAVt Eastern temperature

northAVt Northern temperature

swAVt Southwestern temperature

qldAVt temperature

nswAVt temperature

ntAVt temperature

saAVt temperature

tasAVt temperature

vicAVt temperature

waAVt temperature

mdbAVt Murray-Darling basin temperature

ausAVt Australian average temperature, area-weighted mean

seRain Southeast Australian annual rainfall (mm)

southRain Southern rainfall

eastRain Eastern rainfall

northRain Northern rainfall

swRain Southwest rainfall

qldRain Queensland rainfall

nswRain NSW rainfall

ntRain Northern Territory rainfall

saRain South Australian rainfall

tasRain Tasmanian rainfall

vicRain Victorian rainfall

waRain West Australian rainfall

mdbRain Murray-Darling basin rainfall

ausRain Australian average rainfall, area weighted

SOI Annual average Southern Oscillation Index

sunspot Annual average sunspot counts

co2mlo Moana Loa CO2 concentrations, from 1959

co2law Moana Loa CO2 concentrations, 1900 to 1978

CO2 CO2 concentrations, composite series

avDMI Annual average Dipole Mode Index, for the Indian Ocean Dipole

Source

Australian Bureau of Meteorology web pages:

http://www.bom.gov.au/climate/change/index.shtml

The SOI data are from http://www.bom.gov.au/climate/enso/#tabs=SOI.

The CO2 series `co2law`, for Law Dome ice core data. is from https://data.ess-dive.lbl.gov/portals/CDIAC.

The CO2 series `co2mlo` is from Dr. Pieter Tans, NOAA/ESRL (https://gml.noaa.gov/ccgg/trends/)

The series `CO2` is a composite series, obtained by adding 0.46 to he Law data for 1900 to 1958, then following this with the Moana Loa data that is avaiable from 1959. The addition of 0.46 is designed so that the averages from the two series agree for the period 1959 to 1968

Sunspot data is from http://www.sidc.be/silso/datafiles

References

D.M. Etheridge, L.P. Steele, R.L. Langenfelds, R.J. Francey, J.-M. Barnola and V.I. Morgan, 1998, *Historical CO2 records from the Law Dome DE08, DE08-2, and DSS ice cores*, in Trends: A Compendium of Data on Global Change, on line at Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Oak Ridge, Tenn., U.S.A.

Lavery, B., Joung, G. and Nicholls, N. 1997. An extended high-quality historical rainfall dataset for Australia. Australian Meteorological Magazine, 46, 27-38.

Nicholls, N., Lavery, B., Frederiksen, C.\ and Drosdowsky, W. 1996. Recent apparent changes in relationships between the El Nino – southern oscillation and Australian rainfall and temperature. Geophysical Research Letters 23: 3357-3360.

SIDC-team, World Data Center for the Sunspot Index, Royal Observatory of Belgium, Monthly Report on the International Sunspot Number, online catalogue of the sunspot index: http://www.sidc.be/silso/datafiles, 1900-2011

Examples

```
plot(ts(bomregions2018[, c("mdbRain","SOI")], start=1900),
     panel=function(y,...)panel.smooth(bomregions2018$Year, y,...))
avrain <- bomregions2018[,"mdbRain"]
xbomsoi <- with(bomregions2018, data.frame(Year=Year, SOI=SOI,
                cuberootRain=avrain^0.33))
xbomsoi$trendSOI <- lowess(xbomsoi$SOI, f=0.1)$y
xbomsoi$trendRain <- lowess(xbomsoi$cuberootRain, f=0.1)$y
xbomsoi$detrendRain <-
  with(xbomsoi, cuberootRain - trendRain + mean(trendRain))
xbomsoi$detrendSOI <-
  with(xbomsoi, SOI - trendSOI + mean(trendSOI))
## Plot time series avrain and SOI: ts object xbomsoi
plot(ts(xbomsoi[, c("cuberootRain","SOI")], start=1900),
     panel=function(y,...)panel.smooth(xbomsoi$Year, y,...),
     xlab = "Year", main="", ylim=list(c(250, 800),c(-20,25)))
par(mfrow=c(1,2))
rainpos <- pretty(xbomsoi$cuberootRain^3, 6)
plot(cuberootRain ~ SOI, data = xbomsoi,
     ylab = "Rainfall (cube root scale)", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
mtext(side = 3, line = 0.8, "A", adj = -0.025)
with(xbomsoi, lines(lowess(cuberootRain ~ SOI, f=0.75)))
plot(detrendRain ~ detrendSOI, data = xbomsoi,
     xlab="Detrended SOI", ylab = "Detrended rainfall", yaxt="n")
axis(2, at = rainpos^0.33, labels=paste(rainpos))
with(xbomsoi, lines(lowess(detrendRain ~ detrendSOI, f=0.75)))
mtext(side = 3, line = 0.8, "B", adj = -0.025)
par(mfrow=c(1,1))
```

`bronchitis` *Chronic bronchitis in a sample of men in Cardiff*

Description

The data consist of observations on three variables for each of 212 men in a sample of Cardiff enumeration districts.

Usage

```
bronchitis
```

Format

A data.frame of 212 obs of 3 variables:

`cig` numeric, the number of cigarettes per day

`poll` numeric, the smoke level in the locality

`r` integer, 1= respondent suffered from chronic bronchitis

`rfac` factor, with levels `abs (r=0)`, and `abs (r=0)`

Note

See p.224 in SMIR

Source

This copy of the dataset was copied from version 0.02 of the **SMIR** package, which in turn obtained it from Jones (1975).

References

Jones, K. (1975), *A geographical contribution to the aetiology of chronic bronchitis*, Unpublished BSc dissertation, University of Southampton. Published in Wrigley, N. (1976). *Introduction to the use of logit models in geography*, Geo.Abstracts Ltd, CATMOG 10, University of East Anglia, Norwich.

Murray Aitkin, Brian Francis, John Hinde and Ross Darnell (2009). SMIR: Companion to Statistical Modelling in R (SMIR). Oxford University Press.

Examples

```
data(bronchit)
```

---

bssBYcut *Between group SS for* `y`*, for all possible splits on values of* `x`

Description

Each point of separation between successve values of `x` is used in turn to create two groups of observations. The between group sum of squares for `y` is calculated for each such split.

Usage

```
bssBYcut(x, y, data)
```

Arguments

x	Variable (numeric) used to define splits. Observations with x values less than the cut point go into the first group, while those with values >= the cut point go into the second group.
y	Variable for which BSS values are to be calculated.
data	Data frame with columns x and y.

Value

Data frame with columns:

xOrd	Cut points for splits.
comp2	Between groups sum of squares

Author(s)

J H Maindonald

Examples

```
xy <- bssBYcut(weight, height, women)
with(xy, xy[which.max(bss), ])

## The function is currently defined as
function (x, y, data)
{
    xnam <- deparse(substitute(x))
    ynam <- deparse(substitute(y))
    xv <- data[, xnam]
    yv <- data[, ynam]
    sumss <- function(x, y, cut) {
        av <- mean(y)
        left <- x < cut
        sum(left) * (mean(y[left]) - av)^2 + sum(!left) * (mean(y[!left]) -
            av)^2
    }
    xOrd <- unique(sort(xv))[-1]
    bss <- numeric(length(xOrd))
    for (i in 1:length(xOrd)) {
        bss[i] <- sumss(xv, yv, xOrd[i])
    }
    list(xOrd = xOrd, bss = bss)
  }
```

compareModels *Compare accuracy of alternative classification methods*

Description

Compare, between models, probabilities that the models assign to membership in the correct group or class. Probabilites should be estimated from cross-validation or from bootstrap out-of-bag data or preferably for test data that are completely separate from the data used to dervive the model.

Usage

```
compareModels(groups, estprobs = list(lda = NULL, rf = NULL),
            gpnames = NULL, robust = TRUE, print = TRUE)
```

Arguments

groups	Factor that specifies the groups
estprobs	List whose elements (with names that identify the models) are matrices that give for each observation (row) estimated probabilities of membership for each of the groups (columns).
gpnames	Character: names for groups, if different from levels(groups)
robust	Logical, TRUE or FALSE
print	Logical. Should results be printed?

Details

The estimated probabilities are compared directly, under normal distribution assumptions. An effect is fitted for each observation, plus an effect for the method. Comparison on a logit scale may sometimes be preferable. An option to allow this is scheduled for incorporation in a later version.

Value

modelAVS	Average accuracies for models
modelSE	Approximate average SE for comparing models
gpAVS	Average accuracies for groups
gpSE	Approximate average SE for comparing groups
obsEff	Effects assigned to individual observations

Note

The analysis estimates effects due to model and group (gp), after accounting for differences between observations.

Author(s)

John Maindonald

Examples

```
library(MASS)
library(DAAG)
library(randomForest)
ldahat <- lda(species ~ length+breadth, data=cuckoos, CV=TRUE)$posterior
qdahat <- qda(species ~ length+breadth, data=cuckoos, CV=TRUE)$posterior
rfhat <- predict(randomForest(species ~ length+breadth, data=cuckoos),
                 type="prob")
compareModels(groups=cuckoos$species, estprobs=list(lda=ldahat,
              qda=qdahat, rf=rfhat), robust=FALSE)
```

`confusion` *Given actual and predicted group assignments, give the confusion matrix*

Description

Given actual and predicted group assignments, give the confusion matrix

Usage

```
confusion(actual, predicted, gpnames = NULL, rowcol=c("actual", "predicted"),
printit = c("overall","confusion"), prior = NULL, digits=3)
```

Arguments

`actual`	Actual (prior) group assigments
`predicted`	Predicted group assigments.
`gpnames`	Names for groups, if different from `levels(actual)`
`rowcol`	For predicted categories to appear as rows, specify `rowcol="predicted"`
`printit`	Character vector. Print `"overall"`, or `"confusion"` matrix, or both.
`prior`	Prior probabilities for groups, if different from the relative group frequencies
`digits`	Number of decimal digits to display in printed output

Details

Predicted group assignments should be estimated from cross-validation or from bootstrap out-of-bag data. Better still, work with assignments for test data that are completely separate from the data used to dervive the model.

Value

A list with elements overall (overall accuracy), confusion (confusion matrix) and prior (prior used for calculation of overall accuracy)

Author(s)

John H Maindonald

References

Maindonald and Braun: 'Data Analysis and Graphics Using R', 3rd edition 2010, Section 12.2.2

Examples

```
library(MASS)
library(DAAG)
cl <- lda(species ~ length+breadth, data=cuckoos, CV=TRUE)$class
confusion(cl, cuckoos$species)

## The function is currently defined as
function (actual, predicted, gpnames = NULL,
            rowcol = c("actual", "predicted"),
            printit = c("overall","confusion"),
            prior = NULL, digits = 3)
{
  if (is.null(gpnames))
    gpnames <- levels(actual)
  if (is.logical(printit)){
    if(printit)printit <- c("overall","confusion")
    else printit <- ""
  }
  tab <- table(actual, predicted)
  acctab <- t(apply(tab, 1, function(x) x/sum(x)))
  dimnames(acctab) <- list(Actual = gpnames, `Predicted (cv)` = gpnames)
  if (is.null(prior)) {
    relnum <- table(actual)
    prior <- relnum/sum(relnum)
    acc <- sum(tab[row(tab) == col(tab)])/sum(tab)
  }
  else {
    acc <- sum(prior * diag(acctab))
  }
  names(prior) <- gpnames
  if ("overall"%in%printit) {
    cat("Overall accuracy =", round(acc, digits), "\n")
    if(is.null(prior)){
      cat("This assumes the following prior frequencies:",
          "\n")
      print(round(prior, digits))
    }
  }
  if ("confusion"%in%printit) {
    cat("\nConfusion matrix", "\n")
    print(round(acctab, digits))
  }
  invisible(list(overall=acc, confusion=acctab, prior=prior))
}
```

coralPval *P-values from biological expression array data*

Description

P-values were calculated for each of 3072 genes, for data that compared expression values between post-settlement coral larvae and pre-settlement coral larvae.

Usage

```
data("coralPval")
```

Format

The format is: num [1:3072, 1] 8.60e-01 3.35e-08 3.96e-01 2.79e-01 6.36e-01 ...

Details

t-statistics, and hence p-values, were derived from five replicate two-colour micro-array slides. Details are in a vignette that accompanies the **DAAGbio** package.

Source

See the `?DAAGbio::coralRG`

References

Grasso, L. C.; Maindonald, J.; Rudd, S.; Hayward, D. C.; Saint, R.; Miller, D. J.; and Ball, E. E., 2008. Microarray analysis identifies candidate genes for key roles in coral development. BMC Genomics, 9:540.

Examples

```
## From p-values, calculate Benjamini-Hochberg false discrimination rates
fdr <- p.adjust(gamclass::coralPval, method='BH')
## Number of genes identified as differentially expressed for FDR = 0.01
sum(fdr<=0.01)
```

cvalues *Historical speed of light measurements*

Description

Measurements made beteween 1675 and 1972

Usage

```
cvalues
```

Format

A data frame with 9 observations on the following 3 variables.

`Year` Year of measurement

`speed` estimated speed in meters per second

`error` measurement error, as estimated by experimenter(s)

Source

https://en.wikipedia.org/wiki/Speed_of_light accessed 2011/12/22

Examples

```
data(cvalues)
```

CVcluster *Cross-validation estimate of predictive accuracy for clustered data*

Description

This function adapts cross-validation to work with clustered categorical outcome data. For example, there may be multiple observations on individuals (clusters). It requires a fitting function that accepts a model formula.

Usage

```
CVcluster(formula, id, data, na.action=na.omit, nfold = 15, FUN = MASS::lda,
            predictFUN=function(x, newdata, ...)predict(x, newdata, ...)$class,
              printit = TRUE, cvparts = NULL, seed = 29)
```

Arguments

formula	Model formula
id	numeric, identifies clusters
data	data frame that supplies the data
na.action	na.fail (default) or na.omit
nfold	Number of cross-validation folds
FUN	function that fits the model
predictFUN	function that gives predicted values
printit	Should summary information be printed?
cvparts	Use, if required, to specify the precise folds used for the cross-validation. The comparison between different models will be more accurate if the same folds are used.
seed	Set seed, if required, so that results are exactly reproducible

Value

class	Predicted values from cross-validation
CVaccuracy	Cross-validation estimate of accuracy
confusion	Confusion matrix

Author(s)

John Maindonald

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Examples

```
if(requireNamespace('mlbench')&requireNamespace('MASS')){
data('Vowel',package='mlbench')
acc <- CVcluster(formula=Class ~., id = V1, data = Vowel, nfold = 15, FUN = MASS::lda,
             predictFUN=function(x, newdata, ...)predict(x, newdata, ...)$class,
                    printit = TRUE, cvparts = NULL, seed = 29)
}
```

CVgam *Cross-validation estimate of accuracy from GAM model fit*

Description

The cross-validation estimate of accuracy is sufficiently independent of the available model fitting criteria (including Generalized Cross-validation) that it provides a useful check on the extent of downward bias in the estimated standard error of residual.

Usage

```
CVgam(formula, data, nfold = 10, debug.level = 0, method = "GCV.Cp",
              printit = TRUE, cvparts = NULL, gamma = 1, seed = 29)
```

Arguments

formula	Model formula, for passing to gam.
data	data frame that supplies the data
nfold	Number of cross-validation folds
debug.level	See mgcv for details
method	Fit method for GAM model. See mgcv for details
printit	Should summary information be printed?
cvparts	Use, if required, to specify the precise folds used for the cross-validation. The comparison between different models will be more accurate if the same folds are used.
gamma	See gam for details.
seed	Set seed, if required, so that results are exactly reproducible

Value

fitted	fitted values
resid	residuals
cvscale	scale parameter from cross-validation
scale.gam	scale parameter from function gam

The scale parameter from cross-validation is the error mean square)

Author(s)

John Maindonald

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Examples

```
if(require(sp)){
library(mgcv)
data(meuse)
meuse$ffreq <- factor(meuse$ffreq)
CVgam(formula=log(zinc)~s(elev) + s(dist) + ffreq + soil,
              data = meuse, nfold = 10, debug.level = 0, method = "GCV.Cp",
              printit = TRUE, cvparts = NULL, gamma = 1, seed = 29)
}
```

`eventCounts` *Tabulate vector of dates by specified time event*

Description

For example, dates may be dates of plane crashes. For purposes of analysis, this function tabulates number of crash events per event of time, for each successive specified event.

Usage

```
eventCounts(data, dateCol="Date", from = NULL, to = NULL,
          by = "1 month", categoryCol=NULL, takeOnly=NULL, prefix="n_")
```

Arguments

`data`	Data frame that should include any columns whose names appear in other function arguments.
`dateCol`	Name of column that holds vector of dates
`from`	Starting date. If `NULL` set to first date given. If supplied, any rows earlier than `from` will be omitted. Similarly, rows later than any supplied date `to` will be omitted.
`to`	Final date, for which numbers of events are to be tallied. If `NULL` set to final date given.
`by`	Time event to be used; e.g. "1 day", or "1 week", or "4 weeks", or "1 month", or "1 quarter", or "1 year", or "10 years".
`categoryCol`	If not `NULL` create one column of counts for each level (or if not a factor, unique value).
`takeOnly`	If not `NULL`, a charater string that when deparsed and executed will return a vector of logicals.
`prefix`	If `categoryCol` is not `NULL`, a prefix for the names of the columns of counts. Otherwise (`categoryCol=NULL`) a name for the column of counts.

Value

A data frame, with columns `Date` (the first day of the event for which events are given), and other column(s) that hols counts of events.

Author(s)

John Maindonald

See Also

cut

Examples

```
crashDate <- as.Date(c("1908-09-17","1912-07-12","1913-08-06",
                       "1913-09-09","1913-10-17"))
df <- data.frame(date=crashDate)
byYears <- eventCounts(data=df, dateCol="date",
                       from=as.Date("1908-01-01"),
                       by="1 year")
```

FARS *US fatal road accident data for automobiles, 1998 to 2010*

Description

Data are from the US FARS (Fatality Analysis Recording System) archive that is intended to include every accident in which there was at least one fatality. Data are limited to vehicles where the front seat passenger seat was occupied. Values are given for selected variables only.

Usage

```
FARS
```

Format

A data frame with 134332 observations on the following 18 variables.

`caseid` a character vector. "state:casenum:vnum"

`state` a numeric vector. See the FARS website for details

`age` a numeric vector; 998=not reported; 999=not known. Cases with age < 16 have been omitted

`airbag` a numeric vector

`injury` a numeric vector; 4 indicates death. Blanks, unknown, and "Died prior to accident" have been omitted

`Restraint` a numeric vector

`sex` 1=male, 2=female, 9=unknown

`inimpact` a numeric vector; direction of initial impact. Categories 1 to 12 describe clock positions, so that 1,11, and 12 relate to near frontal impacts; 0 is not a collision; 13: top; 14: undercarriage. 18, introduced in 2005 has been omitted, as have 404 values in additional categories for 2010. 99 denotes a missing value.

`modelyr` a numeric vector

`airbagAvail` a factor with levels `no` `yes` `NA-code`

`airbagDeploy` a factor with levels `no` `yes` `NA-code`

`D_injury` a numeric vector

`D_airbagAvail` a factor with levels `no` `yes` `NA-code`

`D_airbagDeploy` a factor with levels `no` `yes` `NA-code`

`D_Restraint` a factor with levels `no` `yes` `NA-code`

`year` year of accident

Details

Data is for automabiles where the right passenger seat was occupied, with one observation for each such passenger. Observations for vehicles where the most harmful event was a fire or explosion or immersion or gas inhalation, or where someone fell or jumped from the vehicle, are omitted. Data are limited to vehicle body types 1 to 19,48,49,61, or 62. This excludes large trucks, pickup trucks, vans and buses. The 2009 and 2010 data does not include information on whether airbags were installed.

Note

The papers given as references demonstrate the use of Fatal Accident Recording System data to assess the effectiveness of airbags (even differences between different types of airbags) and seatbelts. Useful results can be obtained by matching driver mortality, with and without airbags, to mortality rates for right front seat passengers in cars without passenger airbags.

Source

https://www-fars.nhtsa.dot.gov/Main/index.aspx

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Olson CM, Cummings P, Rivara FP. 2006. Association of first- and second-generation air bags with front occupant death in car crashes: a matched cohort study. Am J Epidemiol 164:161-169

Cummings, P; McKnight, B, 2010. Accounting for vehicle, crash, and occupant characteristics in traffic crash studies. Injury Prevention 16: 363-366

Braver, ER; Shardell, M; Teoh, ER, 2010. *How have changes in air bag designs affected frontal crash mortality?* Ann Epidemiol 20:499-510.

Examples

```
data(FARS)
```

`fars2007` *US Fatal Road Accident Data, 2007 and 2008*

Description

Data are included on variables that may be relevant to assessing airbag and seatbelt effectiveness in preventing fatal injury.

Usage

```
fars2007
fars2008
```

Format

A data frame with 24179 observations on the following 24 variables.

`state` a numeric vector

`casenum` a numeric vector

`vnum` a numeric vector

`pnum` a numeric vector

`lightcond` a numeric vector

`numfatal` a numeric vector

`age` a numeric vector

`airbag` a numeric vector

`injury` a numeric vector

`ptype` a numeric vector

`restraint` a numeric vector

`seatpos` a numeric vector

`sex` a numeric vector

`body` a numeric vector

`inimpact` A numeric vector; numbers 1 to 12 give clockface directions of initial impact. Values in these datasets are limited to 11, 12 and 1; i.e., near frontal impact

`mhevent` a numeric vector

`numoccs` a numeric vector

`travspd` a numeric vector

`modelyr` a numeric vector

Details

Data is for automabiles where a passenger seat was occupied, with one observation for each such passenger.

Source

https://www-fars.nhtsa.dot.gov/Main/index.aspx

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Olson CM, Cummings P, Rivara FP. 2006. Association of first- and second-generation air bags with front occupant death in car crashes: a matched cohort study. Am J Epidemiol 164:161-169

Cummings, P; McKnight, B, 2010. Accounting for vehicle, crash, and occupant characteristics in traffic crash studies. Injury Prevention 16: 363-366

Braver, ER; Shardell, M; Teoh, ER, 2010. *How have changes in air bag designs affected frontal crash mortality?* Ann Epidemiol 20:499-510.

Examples

```
data(fars2007)
str(fars2007)
```

---

frontDeaths *Safety Device effectiveness Measures, by Year*

Description

Safety devices may be airbags or seatbelts. For airbags, alternatives are to use 'airbag installed' or 'airbag deployed' as the criterion. Ratio of driver deaths to passenger deaths are calculated for driver with device and for driver without device, in both cases for passenger without device.

Usage

```
data("frontDeaths")
```

Format

The format is: List of 3 $ airbagAvail : num [1:13, 1:2, 1:4] 1068 1120 1089 1033 940 ... ..- attr(*, "dimnames")=List of 3 .. ..$ years : chr [1:13] "1998" "1999" "2000" "2001" ... .$ D_airbagAvail: chr [1:2] "no" "yes" .. ..$ injury : chr [1:4] "P_injury" "D_injury" "tot" "prop" $ airbagDeploy: num [1:13, 1:2, 1:4] 1133 1226 1196 1151 1091 ... ..- attr(*, "dimnames")=List of 3 .. ..$ years : chr [1:13] "1998" "1999" "2000" "2001" ... .$ D_airbagAvail: chr [1:2] "no" "yes" .. ..$ injury : chr [1:4] "P_injury" "D_injury" "tot" "prop" $ restraint : num [1:13, 1:2, 1:4] 780 783 735 714 741 645 634 561 558 494 ... ..- attr(*, "dimnames")=List of 3 .. ..$ years : chr [1:13] "1998" "1999" "2000" "2001" ... .$ D_airbagAvail: chr [1:2] "no" "yes" .. ..$ injury : chr [1:4] "P_injury" "D_injury" "tot" "prop"

Source

See FARS

Examples

```
data(frontDeaths)
## maybe str(frontDeaths) ; plot(frontDeaths) ...
```

---

gamRF *Random forest fit to residuals from GAM model*

Description

Fit model using gam, then use random forest regression with residuals. Check performance of this hybrid model for predictions to newdata, if supplied.

Usage

```
gamRF(formlist, yvar, data, newdata = NULL, rfVars, method = "GCV.Cp",
    printit = TRUE, seed = NULL)
```

Arguments

formlist	List of rght hand sides of formulae for GAM models.
yvar	Character string holding y-variable name.
data	Data
newdata	Optionally, supply test data.
rfVars	Names of explanatory variables for the randomForest model.
method	Smoothing parameter estimation method for use of gam.
printit	Should a summary of results (error rates) be printed?
seed	Set a seed to make result repeatable.

Value

A vector of test data accuracies for the hybrid models (one for each element of formlist), plus test error mean square and OOB error mean square for the use of randomForest.

Note

The best results are typically obtained when a relatively low degree of freedom GAM model is used. It seems advisable to use those variables for the GAM fit that seem likely to be similar in their effect irrespective of geographic location.

Author(s)

John Maindonald

References

J. Li, A. D. Heap, A. Potter and J. J. Daniell. 2011. Application of Machine Learning Methods to Spatial Interpolation of Environmental Variables. Environmental Modelling and Software 26: 1647-1656. DOI: 10.1016/j.envsoft.2011.07.004.

See Also

CVgam

Examples

```
if(length(find.package("sp", quiet=TRUE))>0){
data("meuse", package="sp")
meuse <- within(meuse, {levels(soil) <- c("1","2","2")
                        ffreq <- as.numeric(ffreq)
                        loglead <- log(lead)}
)
form <- ~ dist + elev + ffreq + soil
rfVars <- c("dist", "elev", "soil", "ffreq", "x", "y")
## Select 90 out of 155 rows
sub <- sample(1:nrow(meuse), 90)
meuseOut <- meuse[-sub,]
meuseIn <- meuse[sub,]
gamRF(formlist=list("lm"=form), yvar="loglead", rfVars=rfVars,
                    data=meuseIn, newdata=meuseOut)
}

## The function is currently defined as
function (formlist, yvar, data, newdata = NULL, rfVars, method = "GCV.Cp",
    printit = TRUE, seed = NULL)
{   if(!is.null(seed))set.seed(seed)
    errRate <- numeric(length(formlist)+2)
    names(errRate) <- c(names(formlist), "rfTest", "rfOOB")
    ytrain <- data[, yvar]
    xtrain <- data[, rfVars]
    xtest <- newdata[, rfVars]
    ytest = newdata[, yvar]
    res.rf <- randomForest(x = xtrain, y = ytrain,
                           xtest=xtest,
                           ytest=ytest)
    errRate["rfOOB"] <- mean(res.rf$mse)
    errRate["rfTest"] <- mean(res.rf$test$mse)
    GAMhat <- numeric(nrow(data))
    for(nam in names(formlist)){
      form <- as.formula(paste(c(yvar, paste(formlist[[nam]])), collapse=" "))
      train.gam <- gam(form, data = data, method = method)
      res <- resid(train.gam)
      cvGAMms <- sum(res^2)/length(res)
      if (!all(rfVars %in% names(newdata))) {
        missNam <- rfVars[!(rfVars %in% names(newdata))]
        stop(paste("The following were not found in 'newdata':",
                   paste(missNam, collapse = ", ")))
```

```
        }
        GAMtesthat <- predict(train.gam, newdata = newdata)
        GAMtestres <- ytest - GAMtesthat
        Gres.rf <- randomForest(x = xtrain, y = res, xtest = xtest,
                                ytest = GAMtestres)
        errRate[nam] <- mean(Gres.rf$test$mse)
      }
      if (printit)
          print(round(errRate, 4))
      invisible(errRate)
  }
```

---

german *German credit scoring data*

Description

See website for details of data attributes

Usage

```
german
```

Format

A data frame with 1000 observations on the following 21 variables.

`V1` a factor with levels `A11` `A12` `A13` `A14`

`V2` a numeric vector

`V3` a factor with levels `A30` `A31` `A32` `A33` `A34`

`V4` a factor with levels `A40` `A41` `A410` `A42` `A43` `A44` `A45` `A46` `A48` `A49`

`V5` a numeric vector

`V6` a factor with levels `A61` `A62` `A63` `A64` `A65`

`V7` a factor with levels `A71` `A72` `A73` `A74` `A75`

`V8` a numeric vector

`V9` a factor with levels `A91` `A92` `A93` `A94`

`V10` a factor with levels `A101` `A102` `A103`

`V11` a numeric vector

`V12` a factor with levels `A121` `A122` `A123` `A124`

`V13` a numeric vector

`V14` a factor with levels `A141` `A142` `A143`

`V15` a factor with levels `A151` `A152` `A153`

`V16` a numeric vector

`V17` a factor with levels `A171` `A172` `A173` `A174`

`V18` a factor with levels good bad

`V19` a factor with levels `A191` `A192`

`V20` a factor with levels `A201` `A202`

`V21` a numeric vector

Details

700 good and 300 bad credits with 20 predictor variables. Data from 1973 to 1975. Stratified sample from actual credits with bad credits heavily oversampled. A cost matrix can be used.

Source

http://archive.ics.uci.edu/datasets

References

Grömping, U. (2019). South German Credit Data: Correcting a Widely Used Data Set. Report 4/2019, Reports in Mathematics, Physics and Chemistry, Department II, Beuth University of Applied Sciences Berlin.

Examples

```
data(german)
```

---

greatLakesM *Monthly Great Lake heights: 1918 - 2019*

Description

Heights, in meters, are for the lakes Erie, Michigan/Huron, Ontario and St Clair

Usage

```
data(greatLakesM)
```

Format

The format is: 'data.frame': 1212 obs. of 7 variables: $ month : Factor w/ 12 levels "apr","aug","dec",..: 5 4 8 1 9 7 6 2 12 11 ... $ year : int 1918 1918 1918 1918 1918 1918 1918 1918 1918 1918 ... $ Superior : num 183 183 183 183 183 ... $ Michigan.Huron: num 177 177 177 177 177 ... $ St..Clair : num 175 175 175 175 175 ... $ Erie : num 174 174 174 174 174 ... $ Ontario : num 74.7 74.7 74.9 75.1 75.1 ...

Details

For more details, go to https://www.glerl.noaa.gov/data/#archives.

Source

The url for the original source is no longer valid. See https://www.glerl.noaa.gov/data/#archives for data for individual lakes.

Examples

```
data(greatLakesM)
mErie <- ts(greatLakesM[,'Erie'], start=1918, frequency=12)
greatLakes <- aggregate(greatLakesM[,-(1:2)], by=list(greatLakesM$year),
                        FUN=mean)
names(greatLakes)[1] <- 'year'
## maybe str(greatLakesM)
```

---

ldaErr *Calculate Error Rates for Linear Discriminant Model*

Description

Given an lda model object, calculate training set error, leave-one-out cross-validation error, and test set error.

Usage

```
ldaErr(train.lda, train, test, group = "type")
```

Arguments

train.lda	Fitted lda model object.
train	Training set data frame.
test	Test set data frame.
group	Factor that identifies groups in training data.

Value

Vector that holds leave-one-out, training, and test error rates

Examples

```
## Not run:
data(spam, package='kernlab')
spam[,-58] <- scale(spam[,-58])
nr <- sample(1:nrow(spam))
spam01 <- spam[nr[1:3601],]     ## Use for training,
spam2 <- spam[nr[3602:4601],]   ## Test
spam01.lda <- lda(type~., data=spam01)
ldaRates <- ldaErr(train.lda=spam01.lda, train=spam01, test=spam2, group="type")

## End(Not run)
```

loti *Global temperature anomalies*

Description

GISS (Goddard Institute for Space Studies) Land-Ocean Temperature Index (LOTI) data for the years 1880 to 2019, giving anomalies in 0.01 degrees Celsius, from the 1951 - 1980 average.

Usage

```
loti
```

Format

A data frame with 140 observations on the following 19 variables.

`Year` a numeric vector

`Jan` a numeric vector

`Feb` a numeric vector

`Mar` a numeric vector

`Apr` a numeric vector

`May` a numeric vector

`Jun` a numeric vector

`Jul` a numeric vector

`Aug` a numeric vector

`Sep` a numeric vector

`Oct` a numeric vector

`Nov` a numeric vector

`Dec` a numeric vector

`JtoD` Jan-Dec averages

`D.N` Dec-Nov averages

`DJF` Dec-Jan-Feb averages

`MAM` Mar-Apr-May

`JJA` Jun-Jul-Aug

`SON` Sept-Oct-Nov

`JtoD2011` January to December average, from data accessed in 2011

Source

Data are the Combined Land-Surface Air and Sea-Surface Water Temperature Anomalies (Land-Ocean Temperature Index, LOTI), in 0.01 degrees Celsius, from https://data.giss.nasa.gov/gistemp/tabledata_v4/GLB.Ts+dSST.txt Data in the column `JtoD2011` was accessed 2011-09-06.

Also available is a CSV file, with anomalies in degrees Celsius.

References

GISTEMP Team, 2020: GISS Surface Temperature Analysis (GISTEMP), version 4. NASA Goddard Institute for Space Studies. Dataset accessed 2020-11-13 at https://data.giss.nasa.gov/gistemp/.

Examples

```
data(loti)
plot(JtoD ~ Year, data=loti)
## Add 11 point moving average
ma11 <- filter(loti$JtoD, rep(1,11)/11, sides=2)
lines(loti$Year, ma11)
```

`plotFars` *Plot Protection Device Effectiveness Measure Against Year*

Description

Devices may be airbags or seatbelts. For airbags, alternatives are to use "airbag installed" or "airbag deployed" as the criterion. The plot shows, for each of the specified features, the ratio of driver death rate (or other outcome, e.g., death or injury) with feature, to rate without feature, in both cases for passenger without feature.

Usage

```
plotFars(tabDeaths=gamclass::frontDeaths,
          statistics = c("airbagAvail", "airbagDeploy", "restraint"))
```

Arguments

`tabDeaths`	List, containing (as a minimum) three-dimensional arrays with the names specified in the argument `statistics`, such as is returned by the function `tabFarsDead`
`statistics`	Vector of character: names of the sublists, which contain information on the deathrates

Details

The name `injury` is used, with `frontDeaths` or `sideDeaths` or `rearDeaths` or `otherDeaths` as the first argument, to refer to deaths. The function `tabFarsDeaths` allows the option of returning an object, suitable for using as first argument, that treats `injury` as death or serious injury.

Value

A graphics object is returned

Note

Note that the "airbag deployed" statistic is not a useful measure of airbag effectiveness. At its most effective, the airbag will deploy only when the accident is sufficiently serious that deployment will reduce the risk of serious injury and/or accident. The with/without deployment comparison compares, in part, serious accidents with less serious accidents.

Author(s)

John Maindonald

---

relDeaths *Yearly Driver deaths, as Fraction of Deaths for All Years*

Description

The four list elements are for four positions of initial impact. Each list element is a 13 by 3 `years` by "safety device" matrix that gives the proportion, for that device in year, of the total over *years*

Usage

```
data("relDeaths")
```

Format

The format is: List of 4 $ front: num [1:13, 1:3] 0.559 0.548 0.544 0.577 0.574 ... ..- attr(*, "dimnames")=List of 2 .. ..$: chr [1:13] "1998" "1999" "2000" "2001" ... .$: chr [1:3] "airbagAvail" "airbagDeploy" "restraint" $ side : num [1:13, 1:3] 0.36 0.366 0.367 0.35 0.348 ... ..- attr(*, "dimnames")=List of 2 .. ..$: chr [1:13] "1998" "1999" "2000" "2001" ... .$: chr [1:3] "airbagAvail" "airbagDeploy" "restraint" $ rear : num [1:13, 1:3] 0.0507 0.0558 0.0575 0.0498 0.0522 ... ..- attr(*, "dimnames")=List of 2 .. ..$: chr [1:13] "1998" "1999" "2000" "2001" ... .$: chr [1:3] "airbagAvail" "airbagDeploy" "restraint" $ other: num [1:13, 1:3] 0.0312 0.0304 0.0313 0.0237 0.0254 ... ..- attr(*, "dimnames")=List of 2 .. ..$: chr [1:13] "1998" "1999" "2000" "2001" ... .$: chr [1:3] "airbagAvail" "airbagDeploy" "restraint"

Examples

```
data(relDeaths)
## maybe str(relDeaths) ; plot(relDeaths) ...
```

RFcluster *Random forests estimate of predictive accuracy for clustered data*

Description

This function adapts random forests to work (albeit clumsily and inefficiently) with clustered categorical outcome data. For example, there may be multiple observations on individuals (clusters). Predictions are made fof the OOB (out of bag) clusters

Usage

```
RFcluster(formula, id, data, nfold = 15,
              ntree=500, progress=TRUE, printit = TRUE, seed = 29)
```

Arguments

formula	Model formula
id	numeric, identifies clusters
data	data frame that supplies the data
nfold	numeric, number of folds
ntree	numeric, number of trees (number of bootstrap samples)
progress	Print information on progress of calculations
printit	Print summary information on accuracy
seed	Set seed, if required, so that results are exactly reproducible

Details

Bootstrap samples are taken of observations in the in-bag clusters. Predictions are made for all observations in the OOB clusters.

Value

class	Predicted values from cross-validation
OOBaccuracy	Cross-validation estimate of accuracy
confusion	Confusion matrix

Author(s)

John Maindonald

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Examples

```
## Not run:
library(mlbench)
library(randomForest)
data(Vowel)
RFcluster(formula=Class ~., id = V1, data = Vowel, nfold = 15,
              ntree=500, progress=TRUE, printit = TRUE, seed = 29)

## End(Not run)
```

---

rfErr *Calculate Error Rates for randomForest model*

Description

Given an randomForest model object, calculate training set error, out-of-bag (OOB) error, and test set error.

Usage

```
rfErr(train.rf, train, test, group = "type")
```

Arguments

train.rf	Fitted randomForest model object.
train	Training set data frame.
test	Test set data frame.
group	Factor that identifies groups

Value

Vector that holds training set error, out-of-bag (OOB) error, and test set error rates.

Examples

```
## Not run:
data(spam, package='kernlab')
spam[,-58] <- scale(spam[,-58])
nr <- sample(1:nrow(spam))
spam01 <- spam[nr[1:3601],]     ## Use for training,
spam2 <- spam[nr[3602:4601],]   ## Test
spam01.rf <- randomForest(type ~ ., data=spam01)
rfRates <- rfErr(train.rf=spam01.rf, train=spam01, test=spam2,
                 group='type')

## End(Not run)
```

rpartErr *Calculate Error Rates for rpart model*

Description

Given an rpart model object, calculate training set error, 10-fold cross-validation error, and test set error.

Usage

```
rpartErr(train.rp, train, test, group = "type")
```

Arguments

train.rp	Fitted lda model object.
train	Training set data frame.
test	Test set data frame.
group	Factor that identifies groups

Value

Vector that holds training set error, 10-fold cross-validation error, and test set error rates.

Examples

```
## Not run:
data(spam, package='kernlab')
spam[,-58] <- scale(spam[,-58])
nr <- sample(1:nrow(spam))
spam01 <- spam[nr[1:3601],]     ## Use for training,
## if holdout not needed
spam2 <- spam[nr[3602:4601],]   ## Test
spam01.rp <- rpart(type~., data=spam01, cp=0.0001)
rpRates <- rpartErr(train.rp=spam01.rp, train=spam01, test=spam2,
                    group='type')

## End(Not run)
```

simreg *Simulate (repeated) regression calculations*

Description

Derive parameter estimates and standard errors by simulation, or by bootstrap resampling.

Usage

```
simreg(formula, data, nsim = 1000)
bootreg(formula, data, nboot = 1000)
```

Arguments

formula	Model formula
data	Data frame from which names in formula can be taken
nsim	Number of repeats of the simulation (simreg)
nboot	Number of bootstrap resamples (bootreg)

Value

Matrix of coefficients from repeated simulations, or from bootstrap resamples. For simreg there is one row for each repeat of the simulation. For bootreg there is one row for each resample.

Note

Note that bootreg uses the simplest possible form of bootstrap. For any except very large datasets, standard errors may be substantial under-estimates

Author(s)

John Maindonald

References

https://maths-people.anu.edu.au/~johnm/nzsr/taws.html

Examples

```
xy <- data.frame(x=rnorm(100), y=rnorm(100))
simcoef <- simreg(formula = y~x, data = xy, nsim = 100)
bootcoef <- bootreg(formula = y~x, data = xy, nboot = 100)
```

`tabFarsDead` *Extract ratio of ratios estimate of safety device effectiveness, from the* `Fars` *dataset.*

Description

Safety devices may be airbags or seatbelts. For airbags, alternatives are to use 'airbag installed' or 'airbag deployed' as the criterion. Ratio of driver deaths to passenger deaths are calculated for driver with device and for driver without device, in both cases for passenger without device, and the ratio of these ratios calculated.

Usage

```
tabFarsDead(dset=gamclass::FARS, fatal = 4,
            restrict=expression(age>=16&age<998&inimpact%in%c(11,12,1)),
            statistics = c("airbagAvail", "airbagDeploy", "Restraint"))
```

Arguments

`dset`	data frame containing data
`fatal`	numeric: 4 for fatal injury, or `c(3,4)` for incapacitating or fatal injury
`statistics`	Vector of character: ratio of rates variables that will be tabulated
`restrict`	Expression restricting values as specified

Details

Note that the 'airbag deployed' statistic is not a useful measure of airbag effectiveness. At its most effective, the airbag will deploy only when the accident is sufficiently serious that deployment will reduce the risk of serious injury and/or accident. The with/without deployment comparison compares, in part, serious accidents with less serious accidents.

Value

A list with elements

`airbagAvail`	a multiway table with margins `yrs`, `airbagAvail`, and a third margin with levels `P_injury`, `D_injury`, `tot`, and `prop`
`airbagDeploy`	a multiway table with margins `yrs`, `airbagDeploy`, and a third margin with levels `P_injury`, `D_injury`, `tot`, and `prop`
`Restraint`	a multiway table with margins `yrs`, `Restraint`, and a third margin `injury` with levels `P_injury`, `D_injury`, `tot`, and `prop`

Author(s)

John Maindonald

Examples

```
tabDeaths <- tabFarsDead()
```

Index

∗ **chron**
  eventCounts, 17
∗ **datasets**
  airAccs, 4
  bomregions2018, 5
  bronchitis, 7
  coralPval, 13
  cvalues, 14
  FARS, 18
  fars2007, 20
  frontDeaths, 21
  german, 24
  greatLakesM, 25
  loti, 27
  relDeaths, 29
∗ **graphics**
  addhlines, 3
∗ **hplot**
  plotFars, 28
∗ **manip**
  bssBYcut, 8
  eventCounts, 17
  tabFarsDead, 34
∗ **models**
  CVcluster, 14
  CVgam, 16
  gamRF, 22
  RFcluster, 30
  simreg, 33
∗ **multivariate**
  compareModels, 10
  confusion, 11
∗ **package**
  gamclass-package, 2
∗ **regression**
  CVcluster, 14
  CVgam, 16
  gamRF, 22
  RFcluster, 30
  simreg, 33
∗ **statistics**
  compareModels, 10
  confusion, 11

addhlines, 3
airAccs, 4

bomregions2018, 5
bootreg *(*simreg*)*, 33
bronchitis, 7
bssBYcut, 8

compareModels, 10
confusion, 11
coralPval, 13
cut, *18*
cvalues, 14
CVcluster, 14
CVgam, 16, *23*

eventCounts, 17

FARS, 18, *21*
fars2007, 20
fars2008 *(*fars2007*)*, 20
frontDeaths, 21

gam, *16*, *22*
gamclass *(*gamclass-package*)*, 2
gamclass-package, 2
gamRF, 22
german, 24
greatLakesM, 25

ldaErr, 26
loti, 27

mgcv, *16*

otherDeaths *(*frontDeaths*)*, 21

plotFars, 28
predict.rpart, *3*

randomForest, *22*
rearDeaths *(*frontDeaths*)*, 21
relDeaths, 29
RFcluster, 30
rfErr, 31
rpartErr, 32

sideDeaths *(*frontDeaths*)*, 21
simreg, 33

tabFarsDead, *28*, 34