Package: regclass (via r-universe)

September 1, 2024

Type Package

Title Tools for an Introductory Class in Regression and Modeling

Version 1.6

Date 2020-2-19

Author Adam Petrie

Maintainer Adam Petrie <apetrie@utk.edu>

Depends R (>= 3.6), bestglm, leaps, VGAM, rpart, randomForest

Imports rpart.plot

Description Contains basic tools for visualizing, interpreting, and
building regression models. It has been designed for use with
the book Introduction to Regression and Modeling with R by Adam
Petrie, Cognella Publishers, ISBN: 978-1-63189-250-9
<https://titles.cognella.com/
introduction-to-regression-and-modeling-with-r-9781631892509>.

License GPL (>= 2)

NeedsCompilation no

Date/Publication 2020-02-21 18:00:07 UTC

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

RemoteUrl https://github.com/cran/regclass

RemoteRef HEAD

RemoteSha 548f2ed1dc66fbf1a58b86f9ada26f9da05483c4

Contents

ACCOUNT *Predicting whether a customer will open a new kind of account*

Description

Customers were marketed a new type of account at a bank. It is desired to model what factors seemed to be associated with the probability of opening the account to tune marketing strategy.

Usage

```
data("ACCOUNT")
```

Format

A data frame with 24242 observations on the following 8 variables.

`Purchase` a factor with levels `No` `Yes`

`Tenure` a numeric vector, the number of years the customer has been with the bank

`CheckingBalance` a numeric vector, amount currently held in checking (may be negative if overdrafted)

`SavingBalance` a numeric vector, amount currently held in savings (0 or larger)

`Income` a numeric vector, yearly income in thousands of dollars

`Homeowner` a factor with levels `No` `Yes`

`Age` a numeric vector

`Area.Classification` a factor with levels `R` `S` `U` for rural, suburban, or urban

Details

Who is more likely to open a new type of account that a bank wants to try to sell its customers? Try logistic regression or partition models to see if you can develop a model that accurately classifies purchasers vs. non-purchasers. Or, try to develop a model that does well in promoting to nearly all customers who would buy the account.

`all_correlations` *Pairwise correlations between quantitative variables*

Description

This function gives a list of all pairwise correlations between quantitative variables in a dataframe. Alternatively, it can provide all pairwise correlations with just a particular variable.

Usage

```
all_correlations(X,type="pearson",interest=NA,sorted="none")
```

Arguments

`X`	A data frame
`type`	Either `pearson`, `spearman`, or `both`. If `pearson`, the Pearson correlations are returned. If `spearman`, the Spearman's rank correlations are returned.
`interest`	If specified, returns only pairwise correlations with this variable. Argument should be in quotes and must give the exact name of the column of the variable of interest.

sorted — Either none, strength, significance, or magnitude. If strength, sorts the list from most negative correlation to most positive (remember, correlations are stronger the farther they are from 0 (positive or negative). If significance, sorts the list by p-value. If none, no sorting takes place. Note: if both is requested, no sorting takes place and an error message is output.

Details

This function filters out any non-numerical variables in the data frame and provides correlations only between quantitative variables. It is useful for quickly glancing at the size of the correlations between many pairs of variables or all correlations with a particular variable. Further analysis should be done on pairs of interest using associate.

Note: if Spearmans' rank correlations are computed, warnings message result indicating that the exact p-value cannot be computed with ties. Running associate will give you an approximate p-value using the permutation procedure.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

cor, associate

Examples

```
#all pairwise (Pearson) correlations between all quantitative variables
data(STUDENT)
all_correlations(STUDENT)
#Spearman correlations between all quantitative variables and CollegeGPA, sorted by pvalue.
#Gives warnings due to ties
all_correlations(STUDENT,interest="CollegeGPA",type="spearman",sorted="significance")
```

APPLIANCE *Appliance shipments*

Description

Appliance shipments from 1960 to 1985

Usage

```
data("APPLIANCE")
```

Format

A data frame with 26 observations on the following 7 variables.

`Year` a numeric vector

`Dishwasher` a numeric vector, Factory shipments (domestic) of dishwashers (thousands)

`Disposal` a numeric vector, Factory shipments (domestic) of disposers (thousands)

`Refrigerator` a numeric vector, Factory shipments (domestic) of refrigerators (thousands)

`Washer` a numeric vector, Factory shipments (domestic) of washing machines (thousands)

`DurableGoodsExp` a numeric vector, Durable goods expenditures (billions of 1972 dollars)

`PrivateResInvest` a numeric vector, Private residential investment (billions of 1972 dollars)

Details

From the (former) Data and Story library.

The file gives unit shipments of dishwashers, disposers, refrigerators, and washers in the United States from 1960 to 1985. This and other data are published currently in the Department of Commerce's Survey of Current Business, and are summarized from time to time in their publication, Business Statistics. Also included in the file are durable goods expenditures and private residential investment in the United States.

`associate` *Association Analysis*

Description

This function takes two quantities and computes relevent numerical measures of association. The p-values of the associations are estimated via permutation tests. Plots for diagnostics are provided as well, with optional arguments that allow for classic tests.

Usage

```
associate(formula, data, permutations = 500, seed=NA, plot = TRUE, classic = FALSE,
  cex.leg=0.7, n.levels=NA,prompt=TRUE,color=TRUE,...)
```

Arguments

`formula`	A standard R formula written as y~x, where y is the name of the variable playing the role of y and x is the name of the variable playing the role of x.
`data`	An optional argument giving the name of the data frame that contains x and y. If not specified, the function will use existing definitions in the parent environment.
`permutations`	The number of permutations for Monte Carlo estimation of the p-value. If 0, function defaults to reporting classic results.
`seed`	An optional argument specifying the random number seed for permutations.
`plot`	`TRUE` or `FALSE`. Indicates whether the relevent plots are displayed.

`classic`	`TRUE` or `FALSE`. Indicates whether p-values should (also) be found using classic approximations.
`cex.leg`	Scale factor for the size of legends in plots. Larger values make legends bigger.
`n.levels`	An optional argument of interest only when y is categorical and x is quantitative. It specifies the number of levels when converting x to a categorical variable during the analysis. Each level will have the same number of cases. If this does not work out evenly, some levels are randomly picked to have one more case than the others. If unspecified, the default is to pick the number of levels so that there are 10 cases per level or a maximum of 6 levels (whichever is smaller).
`prompt`	`TRUE` or `FALSE`. If `FALSE`, function proceeds without prompting user when the number of observations or number of permutation is large (5000 threshold for each for a prompt). Usually only run with `FALSE` for documentation purposes.
`color`	`TRUE` or `FALSE`. Mostly used for mosaic plots. If `FALSE`, plots are presented in greyscale. If `TRUE`, an intelligent color scheme is chosen to shade the plot.
`...`	Additional arguments related to plotting, e.g., pch, lty, lwd

Details

This function uses Monte Carlo simulation (permutation procedure) to approximate the p-value of an association. Only complete cases are considered in the analysis.

Valid formulas may include functions of the variable, e.g. y^2, log10(x), or more complicated functions like I(x1/(x2+x3)). In the latter case, I() must surround the function of interest to be computed correctly.

When both x and y are quantitative variables, an analysis of Pearson's correlation and Spearman's rank correlation is provided. Scatterplots and histograms of the variables are provided. If `classic` is `TRUE`, the QQ-plots of the variables are provided along with tests of assumptions.

When x is categorical and y is quantitative, the averages (as well as mean ranks and medians) of y are compared between levels of x. The "discrepancy" is the F statistic for averages, Kruskal-Wallis statistic for mean ranks, and the chi-squared statistic for the median test. Side-by-side boxplots are also provided. If `classic` is `TRUE`, the QQ-plots of the distribution of y for each level of x are provided.

When x is quantitative and y is categorical, x is converted to a categorical variable with `n.levels` levels with equal numbers of cases. A chi-squared test is performed for the association. The classic approach assumes a multinomial logistic regression to check significance. A mosaic plot showing the distribution of y for each induced level of x is provided as well as a probability "curve". If `classic` is `TRUE`, the multinomial logistic curves for each level are provided versus x..

When both x and y are categorical, a chi-squared test is performed. The contingency table, table of expected counts, and conditional distributions are also reported along with a mosaic plot.

If the permutation procedure is used, the sampling distribution of the measure of association is displayed over the requested amount of permutations along with the observed value on the actual data (except when y is categorical with x quantitative).

If classic results are desired, then plots and tests to check assumptions are supplied. `white.test` from package bstats (version 1.1-11-5) and `mshapiro.test` from package mvnormtest (version 0.1-9) are built into the function to avoid directly referencing the libraries (which sometimes causes problems).

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

lm, glm, anova, cor, chisq.test, vglm

Examples

```
 #Two quantitative variables
 data(SALARY)
associate(Salary~Education,data=SALARY,permutations=1000)

#y is quantitative while x is categorical
data(SURVEY11)
associate(X07.GPA~X40.FavAlcohol,data=SURVEY11,permutations=0,classic=TRUE)

#y is categorical while x is quantitative
data(WINE)
associate(Quality~alcohol,data=WINE,classic=TRUE,n.levels=5)

 #Two categorical variables (many cases, turns off prompt asking for user input)
 data(ACCOUNT)
 set.seed(320)
 #Work with a smaller subset
 SUBSET <- ACCOUNT[sample(nrow(ACCOUNT),1000),]
associate(Purchase~Area.Classification,data=SUBSET,classic=TRUE,prompt=FALSE)
```

ATTRACTF *Attractiveness Score (female)*

Description

The average attractiveness scores of 70 females along with physical attributes

Usage

```
data("ATTRACTF")
```

Format

A data frame with 70 observations on the following 21 variables.

`Score` a numeric vector giving the average attractivness score compiled after 100 student ratings

`Actual.Sexuality` a factor with levels `Gay` `Straight` indicating the self-reported sexuality of the person in the picture

`ApparentRace` a factor with levels `black` `other` `white` indicating the consensus regarding the apparent race of the person

`Chin` a factor with levels `pointed` `rounded` indicating the consensus regarding the shape of the person's chin

`Cleavage` a factor with levels `no` `yes` indicating the consensus regarding whether the pictured woman was prominently displaying cleavage

`ClothingStyle` a factor with levels `conservative` `revealing` indicating the consensus regarding how the women was dressed

`FaceSymmetryScore` a numeric vector indicating the number of people (out of 2) who agreed the woman's case was symmetric

`FashionScore` a numeric vector indicating the number of people (out of 4) who agreed the woman was fashionable

`FitnessScore` a numeric vector indicating the number of people (out of 4) who agreed the woman was physically fit

`GayScore` a numeric vector indicating the number of people (out of 16) who agreed the woman was a lesbian

`Glasses` a factor with levels `Glasses` `No Glasses`

`GroomedScore` a numeric vector indicating the number of people (out of 4) who agreed the woman made a noticeable effort to look nice

`HairColor` a factor with levels `dark` `light` indicating the consensus regarding the woman's hair color

`HairstyleUniquess` a numeric vector indicating the number of people (out of 2) who agreed the woman had an unconventional haircut

`HappinessRating` a numeric vector indicating the number of people (out of 2) who agreed the woman looked happy in her photo

`LookingAtCamera` a factor with levels `no` `yes`

`MakeupScore` a numeric vector indicating the number of people (out of 5) who agreed the woman was wearing a noticeable amount of makeup

`NoseOddScore` a numeric vector indicating the number of people (out of 3) who agreed the woman had an unusually shaped nose

`Selfie` a factor with levels `no` `yes`

`SkinClearScore` a numeric vector indicating the number of people (out of 2) who agreed the woman's complexion was clear.

`Smile` a factor with levels `no` `yes`

Details

Students were asked to rate on a scale of 1 (very unattractive) to 5 (very attractive) the attractiveness of 70 college-aged women who had posted their photos on a dating website. Of the nearly 100 respondents, most were straight males. `Score` represents the average of these ratings.

In a separate survey, students (of both genders) were asked to rate characteristics of the woman by answering the questions: what is her race, is she displaying her cleavage prominently, is she a lesbian, is she physically fit, etc. The variables ending “Score" represent the number of students who answered Yes to the question. Other variables (such as `Selfie`, `Smile`) represent the consensus among the students. The only attribute taken from the woman’s profile was `Actual.Sexuality`.

Source

Students in BAS 320 at the University of Tennessee from 2013-2015.

`ATTRACTM` *Attractiveness Score (male)*

Description

The average attractiveness scores of 70 males along with physical attributes

Usage

```
data("ATTRACTM")
```

Format

A data frame with 70 observations on the following 23 variables.

`Score` a numeric vector giving the average attractivness score compiled after 60 student ratings

`Actual.Sexuality` a factor with levels `Gay` `Straight` indicating the self-reported sexuality of the person in the picture

`ApparentRace` a factor with levels `black` `other` `white` indicating the consensus regarding the apparent race of the person

`Chin` a factor with levels `pointed` `rounded` indicating the consensus regarding the shape of the person’s chin

`ClothingStyle` a factor with levels `conservative` `revealing` indicating the consensus regarding how the man was dressed

`FaceSymmetryScore` a numeric vector indicating the number of people (out of 7) who agreed the woman’s case was symmetric

`FacialHair` a factor with levels `no` `yes` indicating the consensus regarding whether the man appeared to maintain facial hair

`FashionScore` a numeric vector indicating the number of people (out of 7) who agreed the woman was fashionable

`FitnessScore` a numeric vector indicating the number of people (out of 8) who agreed the woman was physically fit

`GayScore` a numeric vector indicating the number of people (out of 16) who agreed the man was gay

`Glasses` a factor with levels `no` `yes`

`GroomedScore` a numeric vector indicating the number of people (out of 6) who agreed the woman made a noticeable effort to look nice

`HairColor` a factor with levels `dark` `light` `unseen` indicating the consensus regarding the man's hair color

`HairstyleUniquess` a numeric vector indicating the number of people (out of 4) who agreed the woman had an unconventional haircut

`HappinessRating` a numeric vector indicating the number of people (out of 6) who agreed the man looked happy in her photo

`Hat` a factor with levels `no` `yes`

`LookingAtCamera` a factor with levels `no` `yes`

`NoseOddScore` a numeric vector indicating the number of people (out of 3) who agreed the woman had an unusually shaped nose

`Piercings` a factor with levels `no` `yes` indicating whether the man had visible piercings

`Selfie` a factor with levels `no` `yes`

`SkinClearScore` a numeric vector indicating the number of people (out of 2) who agreed the woman's complexion was clear.

`Smile` a factor with levels `no` `yes`

`Tattoo` a factor with levels `no` `yes`

Details

Students were asked to rate on a scale of 1 (very unattractive) to 5 (very attractive) the attractiveness of 70 college-aged men who had posted their photos on a dating website. Of the nearly 60 respondents, most were straight females. `Score` represents the average of these ratings.

In a separate survey, students (of both genders) were asked to rate characteristics of the man by answering the questions: what is his race, how symmetric does his face look, is he gay, is he physically fit, etc. The variables ending “Score" represent the number of students who answered Yes to the question. Other variables (such as `Hat`, `Smile`) represent the consensus among the students. The only attribute taken from the man's profile was `Actual.Sexuality`.

Source

Students in BAS 320 at the University of Tennessee from 2013-2015.

`AUTO` *AUTO dataset*

Description

Characteristics of cars from 1991

Usage

```
data("AUTO")
```

Format

A data frame with 82 observations on the following 5 variables.

`CabVolume` a numeric vector, cubic feet of cab space

`Horsepower` a numeric vector, engine horsepower

`FuelEfficiency` a numeric vector, average miles per gallon

`TopSpeed` a numeric vector, miles per hour

`Weight` a numeric vector, in units of 100 lbs

Details

Although this is a popular dataset, there is some question as to the units of the fuel efficiency. The source claims it to be in miles per gallon, but the numbers reported seem unrealistic. However, the units do not appear to be in km/gallon or km/L.

Source

Data provided by the U.S. Environmental Protection Agency and obtained from the (former) Data and Story library

References

R.M. Heavenrich, J.D. Murrell, and K.H. Hellman, Light Duty Automotive Technology and Fuel Economy Trends Through 1991, U.S. Environmental Protection Agency, 1991 (EPA/AA/CTAB/91-02)

`BODYFAT` *BODYFAT data*

Description

Popular Bodyfat dataset

Usage

```
data("BODYFAT")
```

Format

A data frame with 252 observations on the following 14 variables.

`BodyFat` a numeric vector indicating the percentage body fat 0-100

`Age` a numeric vector, yrs

`Weight` a numeric vector, lbs

`Height` a numeric vector, inches

`Neck` a numeric vector

`Chest` a numeric vector

`Abdomen` a numeric vector

`Hip` a numeric vector

`Thigh` a numeric vector

`Knee` a numeric vector

`Ankle` a numeric vector

`Biceps` a numeric vector

`Forearm` a numeric vector

`Wrist` a numeric vector

Details

Bodyfat can be accurately measured by the hydrostatic technique, where someone is submereged in a tank of water. It would be useful to be able to predict body fat from measurements that are simpler to obtain. Unless otherwise specified, all physical measurements are in centimeters.

Source

This is a modified version of the data available in “Fitting Percentage of Body Fat to Simple Body Measurements" as appearing in Journal of Statistics Education v4 n1 (1996). http://www.amstat.org/publications/jse/v4n1/datasets.johnson.html

BODYFAT2 *Secondary BODYFAT dataset*

Description

Bodyfat dataset illustrating quirks of statistical significance

Usage

```
data("BODYFAT2")
```

Format

A data frame with 20 observations on the following 4 variables.

`Triceps` a numeric vector, cm

`Thigh` a numeric vector, cm

`Midarm` a numeric vector, cm

`BodyFat` a numeric vector, 0-100 representing percent

Details

The physical measurements are circumferences of body parts of 25-34 year-old healthy females.

Source

This is a classic dataset found in many textbooks and in many places online. The original source may be Neter, Kutner, Nachtsheim, Wasserman, 1997, p. 261: Applied Statistical Models (4th Edition).

build_model *Variable selection for descriptive or predictive linear and logistic regression models*

Description

This function uses `bestglm` to consider an extensive array of models and makes recommendations on what set of variables is appropriate for the final model. Model hierarchy is not preserved. Interactions and multi-level categorical variables are allowed.

Usage

```
build_model(form,data,type="predictive",Kfold=5,repeats=10,
prompt=TRUE,seed=NA,holdout=NA,...)
```

Arguments

`form`	A model formula giving the most complex model to consider (often predicting y from all variables `y~.` or all variables including two-way interactions `y~.^2`)
`data`	Name of the data frame that contain all variables specifed by `form`
`type`	Either "predictive" or "descriptive". If `predictive`, the procedure estimates the generalization error of candidate models via repeated K-fold cross-validation. If `descriptive`, the procedure calculates the AICs of models.
`Kfold`	The number of folds for repeated K-fold cross-validation for predictive model building
`repeats`	The number of repeats for repeated K-fold cross-validation for predictive model building
`seed`	If specified, the random number seed used to initialize the repeated K-fold cross-validation procedure so that results can be reproduced.
`prompt`	If `FALSE`, the procedure will not output a warning to the user if fitting the candidate set will take "long". Usually only run with `FALSE` for documentation purposes.
`holdout`	A optional dataframe to serve as a holdout sample. The generalization error on the holdout sample will be calculated and displayed for the best model at each number of predictors.
`...`	Additional arguments to `bestglm`. This allows the procedure to do a search rather than exhaustive enumeration or allows tweaking of the number of reported models or maximum number of independent variables (`nvmax`), etc. See `bestglm` and `regsubsets`.

Details

This procedure takes the formula specified by `form` and the original dataframe and simply converts it into a form that `bestglm` (which normally cannot do cross-validation when categorical variables are involved) can use by adding in columns to represent interactions and categorical variables.

One the dataframe has been generated, a warning is given to the user if the procedure may take too long (many rows or many potential predictors), and then `bestglm` is run. A plot and table of models' performances is given, as well as a recommendation for a final set of variables (model with the lowest AIC/estimated generalization error, or a simpler model that is more or less equivalent).

The command returns a list with `bestformula` (the formula of the model with the lowest AIC or the model chosen by the one standard deviation rule), `bestmodel` (the fitted model that had the lowest AIC or the one chosen by the one standard deviation rule), `predictors` (a list giving the predictors that appeared in the best model with 1 predictor, with 2 predictors, etc).

If a descriptive model is sought, the last component of the returned list is `AICtable` (a data frame containing the number of predictors and the AIC of the best model with that number of predictors; a * denotes the model with the lowest AIC while a + denotes the simplest model whose AIC is within 2 of the lowest).

If a predictive model is sought, the last component of the returned list is `CVtable` (a data frame containing the number of predictors and the estimated generalization error of the best model with that number of predictors along with the SD from repeated K-fold cross validation; a * denotes the model with the lowest error while the + denotes the model selected with the one standard deviation rule). Note that the generalization error in the second column of this table is the squared error if the response is quantitative and is another measure of error (not the misclassification rate) if the response is categorical. Additional columns are provided to give the root mean squared error or misclassification rate.

Note: `bestmodel` is the one selected by the one standard deviation rule or the simplest one whose AIC is no more than 2 above the model with the lowest AIC. Because the procedure does not respect model hierarchy and can include interactions, the formula returned may not be immediately useable if it involves a categorical variable since the variable returned is how R names indicator variables. You may have to manually fit the model based on the selected predictors.

If `HOLDOUT` is given a plot of the error on the holdout sample versus the number of predictors (for the best model at that number of predictors) is provided along with the estimated generalization error from the training set. This can be used to see if the models generalize well, but is in general not used to tune which model is selected.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling with R

See Also

`bestglm`, `regsubsets`, `see.models`, `generalization.error`.

Examples

```
#Descriptive model.  Note: Tip and Bill should not be used simultaneously as
#predictors of TipPercentage, so leave Tip out since it's not known ahead of time
data(TIPS)
MODELS <- build_model(TipPercentage~.-Tip,data=TIPS,type="descriptive")
MODELS$AICtable
MODELS$predictors[[1]] #Variable in best model with a single predictors
MODELS$predictors[[2]] #Variables in best model with two predictors
summary(MODELS$bestmodel) #Summary of best model, in this case with two predictors

#Another descriptive model (large dataset so changing prompt=FALSE for documentation)
data(PURCHASE)
set.seed(320)
#Take a subset of full dataframe for quick illustration
SUBSET <- PURCHASE[sample(nrow(PURCHASE),500),]
MODELS <- build_model(Purchase~.,data=SUBSET,type="descriptive",prompt=FALSE)
MODELS$AICtable  #Model with 1 or 2 variables look pretty good
#Predict whether a purchase is made by # of previous visits and distance to store
MODELS$predictors[[2]]

#Predictive model.
data(SALARY)
set.seed(2010)
```

```
train.rows <- sample(nrow(SALARY),0.7*nrow(SALARY),replace=TRUE)
TRAIN <- SALARY[train.rows,]
HOLDOUT <- SALARY[-train.rows,]
MODELS <- build_model(Salary~.^2,data=TRAIN,holdout=HOLDOUT)
summary(MODELS$bestmodel)
M <- lm(Salary~Gender+Education:Months,data=TRAIN)
generalization_error(M,HOLDOUT)

#Predictive model for WINE data, takes a while.  Misclassification rate on holdout sample is 18%.
data(WINE)
set.seed(2010)
train.rows <- sample(nrow(WINE),0.7*nrow(WINE),replace=TRUE)
TRAIN <- WINE[train.rows,]
HOLDOUT <- WINE[-train.rows,]
## Not run: MODELS <- build_model(Quality~.,data=TRAIN,seed=1919,holdout=HOLDOUT)
## Not run: MODELS$CVtable
```

`build_tree` *Exploratory building of partition models*

Description

A tool to choose the "correct" complexity parameter of a tree

Usage

```
build_tree(form, data, minbucket = 5, seed=NA, holdout, mincp=0)
```

Arguments

`form`	A formula describing the tree to be built
`data`	Data frame containing the variables to build the tree
`minbucket`	The minimum number of cases allowed in any leaf in the tree
`seed`	If given, specifies the random number seed so the crossvalidation error can be reproduced.
`holdout`	If given, the error on the holdout sample is calculated and given in the cp table.
`mincp`	The cp parameter to which the tree will be grown. By default it is 0 (recommended), but it can be changed for large datasets. A value of 0.0001 is likely reasonable.

Details

This command combines the action of building a tree to its maximum possible extent using `rpart` and looking at the results using `getcp`. A plot of the estimated relative generalization error (as determined by 10-fold cross validation) versus the number of splits is provided. In addition, the complexity parameter table giving the cp of the tree with the lowest error (and of the simplest tree with an error within one standard deviation of the lowest error) is reported.

If `holdout` is given, the RMSE/misclassification rate on the training and holdout samples are provided in the cp table.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`rpart`, `getcp`

Examples

```
data(JUNK)
build_tree(Junk~.,data=JUNK,seed=1337)
data(CENSUS)
build_tree(ResponseRate~.,data=CENSUS,seed=2017,mincp=0.001)
data(OFFENSE)
build_tree(Win~.,data=OFFENSE[1:200,],seed=2029,holdout=OFFENSE[201:352,])
```

`BULLDOZER` *BULLDOZER data*

Description

Predicting the sales price of a bulldozer at auction

Usage

```
data("BULLDOZER")
```

Format

A data frame with 924 observations on the following 6 variables.

`SalePrice` a numeric vector

`YearsAgo` a numeric vector, the number of years ago (before present) that the sale occurred

`YearMade` a numeric vector, year of manufacture of machine

`Usage` a numeric vector, hours of usage at time of sale

`Blade` a numeric vector, width of the bulldozer blade (feet)

`Tire` a numeric vector, size of primary tires

Details

The goal is to predict the sale price of a particular piece of heavy equiment at auction based on its usage, equipment type, and configuration. The data represents a heavily modified version of competition data found on kaggle.com. See original source for actual dataset

References

https://www.kaggle.com/c/bluebook-for-bulldozers

`BULLDOZER2` *Modified BULLDOZER data*

Description

The BULLDOZER dataset but with the year the dozer was made as a categorical variable

Usage

```
data("BULLDOZER2")
```

Format

A data frame with 924 observations on the following 6 variables.

`Price` a numeric vector

`YearsAgo` a numeric vector

`Usage` a numeric vector

`Tire` a numeric vector

`Decade` a factor with levels `1960s` and `1970s` `1980s` `1990s` `2000s`

`BladeSize` a numeric vector

Details

This is the `BULLDOZER` data except here `YearMade` has been coded into a four level categorical varaible called `Decade`

CALLS *CALLS dataset*

Description

Summary of students' cell phone providers and relative frequency of dropped calls

Usage

```
data("CALLS")
```

Format

A data frame with 579 observations on the following 2 variables.

`Provider` a factor with levels `ATT` `Sprint` `USCellular` `Verizon`

`DropCallFreq` a factor with levels `Occasionally` `Often` `Rarely`

Details

Data is self-reported by students. The dropped call frequency is based on individuals' perceptions and not any independent quantititatve measure. The data is a subset of `SURVEY09`.

Source

Student survey from STAT 201, University of Tennessee Knoxville, Fall 2009

CENSUS *CENSUS data*

Description

Information from the 2010 US Census

Usage

```
data("CENSUS")
```

Format

A data frame with 3534 observations on the following 39 variables.

- `ResponseRate` a numeric vector, 0-100 representing the percentage of households in a block group that mailed in the form
- `Area` a numeric vector, land area in square miles
- `Urban` a numeric vector, percentage of block group in Urbanized area (50000 or greater)
- `Suburban` a numeric vector, percentage of block group in an Urban Cluster area (2500 to 49999)
- `Rural` a numeric vector, percentage of block group in an Urban Cluster area (2500 to 49999)
- `Male` a numeric vector, percentage of males
- `AgeLess5` a numeric vector, percentage of individuals aged less than 5 years old
- `Age5to17` a numeric vector
- `Age18to24` a numeric vector
- `Age25to44` a numeric vector
- `Age45to64` a numeric vector
- `Age65plus` a numeric vector
- `Hispanics` a numeric vector, percentage of individuals who identify as Hispanic
- `Whites` a numeric vector, percentage of individuals who identify as white (alone)
- `Blacks` a numeric vector
- `NativeAmericans` a numeric vector
- `Asians` a numeric vector
- `Hawaiians` a numeric vector
- `Other` a numeric vector, percentage of individuals who identify as another ethnicity
- `RelatedHH` a numeric vector, percentage of households where at least 2 members are related by birth, marriage, or adoption; same-sex couple households with no relatives of the householder present are not included
- `MarriedHH` a numeric vector, percentage of households in which the householder and his or her spouse are listed as members of the same household; does not include same-sex married couples
- `NoSpouseHH` a numeric vector, percentage of households with no spousal relationship present
- `FemaleHH` a numeric vector, percentage of households with a female householder and no husband of householder present
- `AloneHH` a numeric vector, percentage of households where householder is living alone
- `WithKidHH` a numeric vector, percentage of households which have at least one person under the age of 18
- `MedianHHIncomeBlock` a numeric vector, median income of households in the block group (from American Community Survey)
- `MedianHHIncomeCity` a numeric vector, median income of households in the tract
- `OccupiedUnits` a numeric vector, percentage of housing units that are occupied
- `RentingHH` a numeric vector, percentage of housing units occupied by renters

HomeownerHH a numeric vector, percentage of housing units occupied by the owner

MobileHomeUnits a numeric vector, percentage of housing units that are mobile homes (from American Community Survey)

CrowdedUnits a numeric vector, percentage of housing units with more than 1 person per room on average

NoPhoneUnits a numeric vector, percentage of housing units without a landline

NoPlumbingUnits a numeric vector, percentage of housing units without active plumbing

NewUnits a numeric vector, percentage of housing units constructed in 2010 or later

Population a numeric vector, number of people in the block group

NumHH a numeric vector, number of households in the block group

NumUnits a numeric vector, number of housing units in the block group

logMedianHouseValue a numeric vector, the logarithm of the median home value in the block group

Details

The goal is to predict ResponseRate from the other predictors. ResponseRate is the percentage of households in a block group that mailed in the census forms. A block group is on average about 40 blocks, each typically bounded by streets, roads, or water. The number of block groups per county in the US is typically between about 5 and 165 with a median of about 20.

References

See https://www2.census.gov/programs-surveys/research/guidance/planning-databases/2014/pdb-block-2014-11-20a.pdf for variable definitions.

CENSUSMLR *Subset of CENSUS data*

Description

A portion of the CENSUS dataset used for illustration

Usage

```
data("CENSUSMLR")
```

Format

A data frame with 1000 observations on the following 7 variables.

Response a numeric vector, percentage 0-100 of household that mailed in the census form

Population a numeric vector, the number of people living in the census block based on 2010 census

`ACSPopulation` a numeric vector, the number of people living in the census block based on 2010 census

`Rural` a numeric vector, the number of people living in a rural area (in that census block)

`Males` a numeric vector, the number of males living in the census block

`Elderly` a numeric vector, the number of people aged 65+ living in the census block

`Hispanic` a numeric vector, the number of people who self-identify as Hispanic in the census block

Details

See `CENSUS` data for more information.

`CHARITY` *CHARITY dataset*

Description

Charity data (adapted from a small section of a charity's donor database)

Usage

```
data("CHARITY")
```

Format

A data frame with 15283 observations on the following 11 variables.

`Donate` a factor with levels `Donate` `No`

`Homeowner` a factor with levels `No` `Yes`

`Gender` a factor with levels `F` `M`

`UnlistedPhone` a factor with levels `No` `Yes`

`ResponseProportion` a numeric vector giving the fraction of solications that resulted in a donation

`NumResponses` a numeric vector giving the number of past donations

`CardResponseCount` a numeric vector giving the number of past solicitations

`MonthsSinceLastResponse` a numeric vector giving the number of months since last response to solicitation (which may have been declining to give)

`LastGiftAmount` a numeric vector giving the amount of the last donation

`MonthSinceLastGift` a numeric vector giving the number of months since last donation

`LogIncome` a numeric vector giving the logarithm of a scaled and normalized yearly income

Details

This dataset is adapted from a real-world database of donors to a charity.

Source

Unknown

`check_regression` *Linear and Logistic Regression diagnostics*

Description

If the model is a linear regression, obtain tests of linearity, equal spread, and Normality as well as relevant plots (residuals vs. fitted values, histogram of residuals, QQ plot of residuals, and predictor vs. residuals plots). If the model is a logistic regression model, a goodness of fit test is given.

Usage

```
check_regression(M,extra=FALSE,tests=TRUE,simulations=500,n.cats=10,seed=NA,prompt=TRUE)
```

Arguments

`M`	A regression model fitted with either `lm` or `glm`
`extra`	If `TRUE`, allows user to generate the predictor vs. residual plots for linear regression models.
`tests`	If `TRUE`, performs statistical tests of assumptions. If `FALSE`, only visual diagnostics are provided.
`simulations`	The number of artificial samples to generate for estimating the p-value of the goodness of fit test for logistic regression models. These artificial samples are generated assuming the fitted logistic regression is correct.
`n.cats`	Number of (roughly) equal sized categories for the Hosmer-Lemeshow goodness of fit test for logistic regression models
`seed`	If specified, sets the random number seed before generation of artificial samples in the goodness of fit tests for logistic regression models.
`prompt`	For documentation only, if `FALSE`, skips prompting user for extra plots

Details

This function provides standard visual and statistical diagnostics for regression models.

For linear regression, tests of linearity, equal spread, and Normality are performed and residuals plots are generated.

The test for linearity (a goodness of fit test) is an F-test. A simple linear regression model predicting y from x is fit and compared to a model treating each value of the predictor as some level of a categorical variable. If this more sophisticated model does not offer a significant improvement in the sum of squared errors, the linearity assumption in that predictor is reasonable. If the p-value is larger 0.05, then statistically we can consider the relationship to be linear. If the p-value is smaller than 0.05, check the residuals plot and the predictor vs residuals plots for signs of obvious curvature (the test can be overly sensitive to inconsequential violations for larger sample sizes). The test can only be run if are two or more individuals that have a common value of x. A test of the model as a whole is run similarly if at least two individuals have identical combinations of all predictor variables.

Note: if categorical variables, interactions, polynomial terms, etc., are present in the model, the test for linearity is conducted for each term even when it does not necessarily make sense to do so.

The test for equal spread is the Breusch-Pagan test. If the p-value is larger 0.05, then statistically we can consider the residuals to have equal spread everywhere. If the p-value is smaller than 0.05, check the residuals plot for obvious signs of unequal spread (the test can be overly sensitive to inconsequential violations for larger sample sizes).

The test for Normality is the Shapiro-Wilk test when the sample size is smaller than 5000, or the KS-test for larger sample sizes. If the p-value is larger 0.05, then statistically we can consider the residuals to be Normally distributed. If the p-value is smaller than 0.05, check the histogram and QQ plot of residuals to look for obvious signs of non-Normality (e.g., skewness or outlier). The test can be overly sensitive to inconsequential violations for larger sample sizes.

The first three plots displayed are the residuals plot (residuals vs. fitted values), histogram of residuals, and QQ plot of residuals. The function gives the option of pressing Enter to display additional predictor vs. residual plots if `extra=TRUE`, or to terminate by typing 'q' in the console and pressing Enter. If polynomial or interactions terms are present in the model, a plot is provided for each term. If categorical predictors are present, plots are provided for each indicator variable.

For logistic regression, two goodness of fit tests are offered.

Method 1 is a crude test that assumes the fitted logistic regression is correct, then generates an artifical sample according the predicted probabilities. A chi-squared test is conducted that compares the observed levels to the predicted levels. The test is failed is the p-value is less than 0.05. The test is not sensitive to departures from the logistic curve unless the sample size is very large or the logistic curve is a really bad model.

Method 2 is a Hosmer-Lemeshow type goodness of fit test. The observations are put into 10 groups according to the probability predicted by the logistic regression model. For example, if there were 200 observations, the first group would have the cases with the 20 smallest predicted probabilities, the second group would have the cases with the 20 next smallest probabilities, etc. The number of cases with the level of interest is compared with the expected number given the fitted logistic regression model via a chi-squared test. The test is failed is the p-value is less than 0.05.

Note: for both methods, the p-values of the chi-squared tests are estimate via Monte Carlo simulation instead of any asymptotic results.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`lm`, `glm`, `shapiro.test`, `ks.test`, bptest (in package lmtest). The goodness of fit test for logistic regression is further detailed and implemented in package 'rms' using the commands lrm and residuals.

Examples

```
#Simple linear regression where everything looks good
data(FRIEND)
M <- lm(FriendshipPotential~Attractiveness,data=FRIEND)
check_regression(M)

#Multiple linear regression (prompt is FALSE only for documentation)
data(AUTO)
M <- lm(FuelEfficiency~.,data=AUTO)
check_regression(M,extra=TRUE,prompt=FALSE)


#Multiple linear regression with a categorical predictors and an interaction
data(TIPS)
M <- lm(TipPercentage~Bill*PartySize*Weekday,data=TIPS)
check_regression(M)

#Multiple linear regression with polynomial term (prompt is FALSE only for documentation)
#Note:  in this example only plots are provided
data(BULLDOZER)
M <- lm(SalePrice~.-YearMade+poly(YearMade,2),data=BULLDOZER)
check_regression(M,extra=TRUE,tests=FALSE,prompt=FALSE)

#Simple logistic regression.  Use 8 categories since only 8 unique values of Dose
data(POISON)
M <- glm(Outcome~Dose,data=POISON,family=binomial)
check_regression(M,n.cats=8,seed=892)

#Multiple logistic regression
data(WINE)
M <- glm(Quality~.,data=WINE,family=binomial)
check_regression(M,seed=2010)
```

---

choose_order   *Choosing order of a polynomial model*

Description

This function takes a simple linear regression model and displays the adjusted R^2 and AICc for the original model (order 1) and for polynomial models up to a specified maximum order and plots the fitted models.

Usage

```
choose_order(M,max.order=6,sort=FALSE,loc="topleft",...)
```

Arguments

`M`	A simple linear regression model fitted with lm()
`max.order`	The maximum order of the polynomial model to consider.
`sort`	How to sort the results. If TRUE, "R2", "r2", "r2adj", or "R2adj", sorts from highest to lowest adjusted R^2. If "AIC", "aic", "AICC", "AICc", sorts by AICc.
`loc`	Location of the legend. Can also be "top", "topright", "bottomleft", "bottom", "bottomright", "left", "right", "center"
`...`	Additional arguments to plot(), e.g., pch

Details

The function outputs a table of the order of the polynomial and the according adjusted R^2 and AICc. One strategy for picking the best order is to find the highest value of R^2 adjusted, then to choose the smallest order (simplest model) that has an R^2 adjusted within 0.005. Another strategy is the find the lowest value of AICc, then to choose the smallest order that has an AICc no more than 2 higher.

The scatterplot of the data is provided and the fitted models are displayed as well.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
 data(BULLDOZER)
M <- lm(SalePrice~YearMade,data=BULLDOZER)
 #Unsorted list, messing with plot options to make it look alright
choose_order(M,pch=20,cex=.3)

#Sort by R2adj.  A 10th order polynomial is highest, but this seems overly complex
choose_order(M,max.order=10,sort=TRUE)

#Sort by AICc.  4th order is lowest, but 2nd order is simpler and within 2 of lowest
choose_order(M,max.order=10,sort="aic")
```

CHURN *CHURN dataset*

Description

Churn data (artificial based on claims similar to real world) from the UCI data repository

Usage

```
data("CHURN")
```

Format

A data frame with 5000 observations on the following 18 variables.

`churn` a factor with levels `No` `Yes`

`accountlength` a numeric vector

`internationalplan` a factor with levels `no` `yes`

`voicemailplan` a factor with levels `no` `yes`

`numbervmailmessages` a numeric vector

`totaldayminutes` a numeric vector

`totaldaycalls` a numeric vector

`totaldaycharge` a numeric vector

`totaleveminutes` a numeric vector

`totalevecalls` a numeric vector

`totalevecharge` a numeric vector

`totalnightminutes` a numeric vector

`totalnightcalls` a numeric vector

`totalnightcharge` a numeric vector

`totalintlminutes` a numeric vector

`totalintlcalls` a numeric vector

`totalintlcharge` a numeric vector

`numbercustomerservicecalls` a numeric vector

Details

This dataset is modified from the one stored at the UCI data repository (namely, the area code and phone number have been deleted). This is artificial data similar to what is found in actual customer profiles. Charges are in dollars.

Source

Though originally on the UCI data repository, actual data was obtained via https://www.sgi.com/tech/mlc/db/

combine_rare_levels *Combines rare levels of a categorical variable*

Description

This function takes a categorical variable and combines all levels with frequencies less than a user-specified threshold named `Combined`

Usage

```
combine_rare_levels(x,threshold=20,newname="Combined")
```

Arguments

`x`	a vector of categorical values
`threshold`	levels that appear a total of `threshold` times or fewer will be combined into a new level called `Combined`
`newname`	defaults to `Combined`, but give the option as to what this new level will be called

Details

Returns a list of two objects:

`values` - The recoded values of the categorical variable. All levels which appeared `threshold` times or fewer are now known as `Combined` `combined` - The levels that have been combined together

If, after being combined, the `newname` level has `threshold` or fewer instances, the remaining level that appears least often is combined as well.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
data(EX6.CLICK)
x <- EX6.CLICK[,15]
table(x)

#Combine all levels which appear 700 or fewer times (AA, CC, DD)
y <- combine_rare_levels(x,700)
 table( y$values )

 #Combine all levels which appear 1350 or fewer times.  This forces BB (which
 #occurs 2422 times) into the Combined level since the three levels that appear
```

```
#fewer than 1350 times do not appear more than 1350 times combined
y <- combine_rare_levels(x,1350)
table( y$values )
```

---

confusion_matrix *Confusion matrix for logistic regression models*

Description

This function takes the output of a logistic regression created with `glm` and returns the confusion matrix.

Usage

```
confusion_matrix(M,DATA=NA)
```

Arguments

M	A logistic regression model created with glm
DATA	A data frame on which the confusion matrix will be made. If omitted, the confusion matrix is on the data used in M. If specified, the data frame must have the same column names as the data used to build the model in M.

Details

This function makes classifications on the data used to build a logistic regression model by predicting the "level of interest" (last alphabetically) when the predicted probability exceeds 50%.

Author(s)

Adam Petrie

See Also

glm

Examples

```
#On WINE data as a whole
data(WINE)
M <- glm(Quality~.,data=WINE,family=binomial)
confusion_matrix(M)

#Calculate generalization error using training/holdout
set.seed(1010)
train.rows <- sample(nrow(WINE),0.7*nrow(WINE),replace=TRUE)
TRAIN <- WINE[train.rows,]
HOLDOUT <- WINE[-train.rows,]
```

```
 M <- glm(Quality~.,data=TRAIN,family=binomial)
confusion_matrix(M,HOLDOUT)


#Predicting donation
#Model predicting from recent average gift amount is significant, but its
#classifications are the same as the naive model (majority rules)
data(DONOR)
M.naive <- glm(Donate~1,data=DONOR,family=binomial)
confusion_matrix(M.naive)
M <- glm(Donate~RECENT_AVG_GIFT_AMT,data=DONOR,family=binomial)
confusion_matrix(M)
```

`cor_demo` *Correlation demo*

Description

This function shows the correlation and coefficient of determination as user interactively adds datapoints. Useful for seeing what different values of correlation look like and seeing the effect of outliers.

Usage

```
cor_demo(cex.leg=0.8)
```

Arguments

`cex.leg`	A number specifying the magnification of legends inside the plot. Smaller numbers mean smaller font.

Details

This function allows the user to generate data by click on a plot. Once two points are added, the correlation (r) and coefficient of determination (r^2) are displayed. When an additional point is added, these values are updated in the upper left with previous values being displayed in the upper right. The effect of outliers on the correlation and coefficient of determination can easily be illustrated. Pressing the red UNDO button on the plot will allow you to take away recently added points for further exploration.

Note: To end the demo, you MUST click on the red box labeled "End" (or press Escape, which will return an error)

Author(s)

Adam Petrie

`cor_matrix` *Correlation Matrix*

Description

This function produces the matrix of correlations between all quantitative variables in a dataframe.

Usage

```
cor_matrix(X,type="pearson")
```

Arguments

`X`	A data frame
`type`	Either `pearson` or `spearman`. If `pearson`, the Pearson correlations are returned. If `spearman`, the Spearman's rank correlations are returned.

Details

This function filters out any non-numerical variables and provides correlations only between quantitative variables. Best for datasets with only a few variables. The correlation matrix is returned (with class `matrix`).

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`cor`

Examples

```
 data(TIPS)
cor_matrix(TIPS)
data(AUTO)
cor_matrix(AUTO,type="spearman")
```

CUSTCHURN *CUSTCHURN dataset*

Description

Customer database describing customer churn (adapted from a former case study)

Usage

```
data("CUSTCHURN")
```

Format

A data frame with 500 observations on the following 11 variables.

`Duration` a numeric vector giving the days that the company was considered a customer. Note: censored at 730 days, which is the value for someone who is currently a customer (not churned)

`Churn` a factor with levels `N` `Y` giving whether the customer has churned or not

`RetentionCost` a numeric vector giving the average amount of money spent per year to retain the individual or company as a customer

`EBiz` a factor with levels `No` `Yes` giving whether the customer was an e-business or not

`CompanyRevenue` a numeric vector giving the company's revenue

`CompanyEmployees` a numeric vector giving the number of employees working for the company

`Categories` a numeric vector giving the number of product categories from which customer made a purchase of their lifetime

`NumPurchases` a numeric vector giving the total amount of purchases over the customer's lifetime

Details

Each row corresponds to a customer of a Fortune 500 company. These customers are businesses, which may or may not exclusively be an e-business. Whether a customer is still a customer (or has churned) after 730 days is recorded.

Source

Unknown

CUSTLOYALTY *CUSTLOYALTY dataset*

Description

Customer database describing customer value (adapted from a former case study) and whether they have a loyalty card

Usage

```
data("CUSTLOYALTY")
```

Format

A data frame with 500 observations on the following 9 variables.

`Gender` a factor with levels `Female` `Male` giving the customer's gender

`Married` a factor with levels `Married` `Single` giving the customer's marital status

`Income` a factor with levels `f0t30`, `f30t45`, `f45t60`, `f60t75`, `f75t90`, `f90toINF` giving the approximate yearly income of the customer. The first level corresponds to 30K or less, the second level corresponds to 30K to 45K, and the last level corresponds to 90K or above

`FirstPurchase` a numeric vector giving the amount of the customer's first purchase amount

`LoyaltyCard` a factor with levels `No` `Yes` that gives whether the customer has a loyalty card for the store

`WalletShare` a numeric vector giving the percentage from 0 to 100 of similar products that the customer makes at this store. A value of 100 means the customer uses this store exclusively for such purchases.

`CustomerLV` a numeric vector giving the lifetime value of the customer and reflects the amount spent acquiring and retaining the customer along with the revenue brought in by the customer

`TotTransactions` a numeric vector giving the total number of consecutive months the customer has made a transaction in the last year

`LastTransaction` a numeric vector giving the total amount of months since the customers last transaction

Details

Each row corresponds to a customer of a local chain. Does having a loyalty card increase the customer's value?

Source

Unknown

CUSTREACQUIRE *CUSTREACQUIRE dataset*

Description

Customer reacquisition

Usage

```
data("CUSTREACQUIRE")
```

Format

A data frame with 500 observations on the following 9 variables.

`Reacquire` a factor with levels `No` `Yes` indicating whether a customer who has previously churned was reacquired

`Lifetime2` a numeric vector giving the days that the company was considered a customer

`Value2` a numeric vector giving the lifetime value of the customer (related to the amount of money spent on reacquisition and the revenue brought in by the customer; can be negative)

`Lifetime1` a numeric vector giving the days that the company was considered a customer before churning the first time

`OfferAmount` a numeric vector giving the money equivalent of a special offer given to the former customer in an attempt to reacquire

`Lapse` a numeric vector giving the number of days between the customer churning and the time of the offer

`PriceChange` a numeric vector giving the percentage by which the typical product purchased by the customer has changed from the time they churned to the time the special offer was sent

`Gender` a factor with levels `Female` `Male` giving the gender of the customer

`Age` a numeric vector giving the age of the customer

Details

A company kept records of its success in reacquiring customers that had previously churned. Data is based on a previous case study.

Source

Unknown

CUSTVALUE *CUSTVALUE dataset*

Description

Customer database describing customer value (adapted from a former case study)

Usage

```
data("CUSTVALUE")
```

Format

A data frame with 500 observations on the following 11 variables.

`Acquired` a factor with levels `No` `Yes` indicating whether a potential customer was acquired

`Duration` a numeric vector giving the days that the company was considered a customer

`LifetimeValue` a numeric vector giving the lifetime value of the customer (related to the amount of money spent on acquisition and the revenue brought in by the customer; can be negative)

`AcquisitionCost` a numeric vector giving the amount of money spent attempting to acquire as a customer

`RetentionCost` a numeric vector giving the average amount of money spent per year to retain the individual or company as a customer

`NumPurchases` a numeric vector giving the total amount of purchases over the customer's lifetime

`Categories` a numeric vector giving the number of product categories from which customer made a purchase of their lifetime

`WalletShare` a numeric vector giving the percentage of purchases of similar products the customer makes with this company; a few values exceed 100 for some reason

`EBiz` a factor with levels `No` `Yes` giving whether the customer was an e-business or not

`CompanyRevenue` a numeric vector giving the company's revenue

`CompanyEmployees` a numeric vector giving the number of employees working for the company

Details

Each row corresponds to a (potential) customer of a Fortune 500 company. These customers are businesses, which may or may not exclusively an e-business.

Source

Unknown

DIET — *DIET data*

Description

The weight of a person over time who is dieting and exercising

Usage

```
data("DIET")
```

Format

A data frame with 35 observations on the following 2 variables.

`Weight` a numeric vector, lbs

`Day` a numeric vector, the number of days after the diet started

Details

This data was collected by the author and consists of his weight measured first thing in the morning over the course of amount a month. The scale round to the nearest 0.2 lbs.

DONOR — *DONOR dataset*

Description

Adapted from the KDD-CUP-98 data set concerning data regarding donations made to a national veterans organization.

Usage

```
data("DONOR")
```

Format

A data frame with 19372 observations on the following 50 variables.

`Donate` a factor with levels `No` `Yes`

`Donation.Amount` a numeric vector

`ID` a numeric vector

`MONTHS_SINCE_ORIGIN` a numeric vector, number of months donor has been in the database

`DONOR_AGE` a numeric vector

`IN_HOUSE` a numeric vector, 1 if person has donated to the charity's “In House" program

URBANICITY a factor with levels ? C R S T U

SES a factor with levels ? 1 2 3 4, one of five possible codes indicating socioeconomic status

CLUSTER_CODE a factor with levels . 01 02 ... 53, one of 54 possible cluster codes, which are unique in terms of socioeconomic status, urbanicity, ethnicity, and other demographic characteristics

HOME_OWNER a factor with levels H U

DONOR_GENDER a factor with levels A F M U

INCOME_GROUP a numeric vector, but in reality one of 7 possible income groups inferred from demographics

PUBLISHED_PHONE a numeric vector, listed (1) vs not listed (0)

OVERLAY_SOURCE a factor with levels B M N P, source from which the donor was match; B is both sources and N is neither

MOR_HIT_RATE a numeric vector, number of known times donor has responded to a mailed solicitation from a group other than the charity

WEALTH_RATING a numeric vector, but in reality one of 10 groups based on demographics

MEDIAN_HOME_VALUE a numeric vector, inferred from other variables

MEDIAN_HOUSEHOLD_INCOME a numeric vector, inferred from other variables

PCT_OWNER_OCCUPIED a numeric vector, percent of owner-occupied housing near where person lives

PER_CAPITA_INCOME a numeric vector, of neighborhood in which person lives

PCT_ATTRIBUTE1 a numeric vector, percent of residents in person's neighborhood that are male and active military

PCT_ATTRIBUTE2 a numeric vector, percent of residents in person's neighborhood that are male and veterans

PCT_ATTRIBUTE3 a numeric vector, percent of residents in person's neighborhood that are Vietnam veterans

PCT_ATTRIBUTE4 a numeric vector, percent of residents in person's neighborhood that are WW2 veterans

PEP_STAR a numeric vector, 1 if has achieved STAR donor status and 0 otherwise

RECENT_STAR_STATUS a numeric vector, 1 if achieved STAR within last 4 years

RECENCY_STATUS_96NK a factor with levels A (active) E (inactive) F (first time) L (lapsing)N (new) S (star donor) as of 1996.

FREQUENCY_STATUS_97NK a numeric vector indicating number of times donated in last period (but period is determined by RECENCY STATUS 96NK)

RECENT_RESPONSE_PROP a numeric vector, proportion of responses to the individual to the number of (card or other) solicitations from the charitable organization since four years ago

RECENT_AVG_GIFT_AMT a numeric vector, average donation from the individual to the charitable organization since four years ago

RECENT_CARD_RESPONSE_PROP a numeric vector, number of times the individual has responded to a card solicitation from the charitable organization since four years ago

RECENT_AVG_CARD_GIFT_AMT a numeric vector, average donation from the individual in response to a card solicitation from the charitable organization since four years ago

RECENT_RESPONSE_COUNT a numeric vector, number of times the individual has responded to a promotion (card or other) from the charitable organization since four years ago

RECENT_CARD_RESPONSE_COUNT a numeric vector, number of times the individual has responded to a card solicitation from the charitable organization since four years ago

MONTHS_SINCE_LAST_PROM_RESP a numeric vector, number of months since the individual has responded to a promotion by the charitable organization

LIFETIME_CARD_PROM a numeric vector, total number of card promotions sent to the individual by the charitable organization

LIFETIME_PROM a numeric vector, total number of promotions sent to the individual by the charitable organization

LIFETIME_GIFT_AMOUNT a numeric vector, total lifetime donation amount from the individual to the charitable organization

LIFETIME_GIFT_COUNT a numeric vector, total number of donations from the individual to the charitable organization

LIFETIME_AVG_GIFT_AMT a numeric vector, lifetime average donation from the individual to the charitable organization

LIFETIME_GIFT_RANGE a numeric vector, difference between maximum and minimum donation amounts from the individual

LIFETIME_MAX_GIFT_AMT a numeric vector

LIFETIME_MIN_GIFT_AMT a numeric vector

LAST_GIFT_AMT a numeric vector

CARD_PROM_12 a numeric vector, number of card promotions sent to the individual by the charitable organization in the last 12 months

NUMBER_PROM_12 a numeric vector, number of promotions (card or other) sent to the individual by the charitable organization in the last 12 months

MONTHS_SINCE_LAST_GIFT a numeric vector

MONTHS_SINCE_FIRST_GIFT a numeric vector

FILE_AVG_GIFT a numeric vector, same as LIFETIME_AVG_GIFT_AMT

FILE_CARD_GIFT a numeric vector, lifetime average donation from the individual in response to all card solicitations from the charitable organization

Details

Originally, this data was used with the 1998 KDD competition (https://kdd.ics.uci.edu/databases/kddcup98/kddcup98.html). This particular version has been adapted from the version available in SAS Enterprise Miner (http://support.sas.com/documentation/cdl/en/emgsj/61207/PDF/default/emgsj.pdf Appendix 2 for descriptions of variable names). One goal is to determine whether a past donor donated in response to the 97NK mail solicitation and (if so), how much, based on age, gender, most recent donation amount, total gift amount, etc.

EDUCATION *EDUCATION data*

Description

Data on the College GPAs of students in an introductory statistics class

Usage

```
data("EDUCATION")
```

Format

A data frame with 607 observations on the following 18 variables.

- `CollegeGPA` a numeric vector
- `Gender` a factor with levels `Female` `Male`
- `HSGPA` a numeric vector, can range up to 5 if the high school allowed it
- `ACT` a numeric vector, ACT score
- `APHours` a numeric vector, number of AP hours student took in HS
- `JobHours` a numeric vector, number of hours student currently works on average
- `School` a factor with levels `Private` `Public`, type of HS
- `LanguagesSpoken` a numeric vector
- `HSHonorsClasses` a numeric vector, number of honors classes taken in HS
- `SmokeInHS` a factor with levels `No` `Yes`
- `PayCollegeNoLoans` a factor with levels `No` `Yes`, can the student and his/her family pay for the University of Tennessee without taking out loans?
- `ClubsInHS` a numeric vector, number of clubs belonged to in HS
- `JobInHS` a factor with levels `No` `Yes`, whether the student maintained a job at some point while in HS
- `Churchgoer` a factor with levels `No` `Yes`, answer to the question Do you regularly attend chruch?
- `Height` a numeric vector (inches)
- `Weight` a numeric vector (lbs)
- `Family` what position they are in the family, a factor with levels `Middle Child` `Oldest Child` `Only Child` `Youngest Child`
- `Pet` favorite pet, a factor with levels `Both` `Cat` `Dog` `Neither`

Details

Responses are from students in an introductory statistics class at the University of Tennessee in 2010. One goal to try to predict someone's college GPA from some of the students' characteristics. What information about a high school student could a college admission's counselor use to anticipate that student's performance in college?

EX2.CENSUS *CENSUS data for Exercise 5 in Chapter 2*

Description

CENSUS data for Exercise 5 in Chapter 2

Usage

```
data("EX2.CENSUS")
```

Format

A data frame with 3534 observations on the following 41 variables.

`ResponseRate` a numeric vector

`Area` a numeric vector

`Urban` a numeric vector

`Suburban` a numeric vector

`Rural` a numeric vector

`Male` a numeric vector

`Female` a numeric vector

`AgeLess5` a numeric vector

`Age5to17` a numeric vector

`Age18to24` a numeric vector

`Age25to44` a numeric vector

`Age45to64` a numeric vector

`Age65plus` a numeric vector

`Hispanics` a numeric vector

`Whites` a numeric vector

`Blacks` a numeric vector

`NativeAmericans` a numeric vector

`Asians` a numeric vector

`Hawaiians` a numeric vector

`Other` a numeric vector

`RelatedHH` a numeric vector

`MarriedHH` a numeric vector

`NoSpouseHH` a numeric vector

`FemaleHH` a numeric vector

`AloneHH` a numeric vector

`WithKidHH` a numeric vector

`MedianHHIncomeBlock` a numeric vector

`MedianHHIncomeCity` a numeric vector

`OccupiedUnits` a numeric vector

`VacantUnits` a numeric vector

`RentingHH` a numeric vector

`HomeownerHH` a numeric vector

`MobileHomeUnits` a numeric vector

`CrowdedUnits` a numeric vector

`NoPhoneUnits` a numeric vector

`NoPlumbingUnits` a numeric vector

`NewUnits` a numeric vector

`Population` a numeric vector

`NumHH` a numeric vector

`NumUnits` a numeric vector

`logMedianHouseValue` a numeric vector

Details

See `CENSUS` for variable descriptions (this data is nearly identical). The goal is to predict `ResponseRate` from the other predictors. `ResponseRate` is the percentage of households in a block group that mailed in the census forms. A block group is on average about 40 blocks, each typically bounded by streets, roads, or water. The number of block groups per county in the US is typically between about 5 and 165 with a median of about 20.

`EX2.TIPS` *TIPS data for Exercise 6 in Chapter 2*

Description

TIPS data for Exercise 6 in Chapter 2

Usage

```
data("EX2.TIPS")
```

Format

A data frame with 244 observations on the following 8 variables.

`Tip.Percentage` a numeric vector

`Bill_in_USD` a numeric vector

`Tip_in_USD` a numeric vector

`Gender` a factor with levels `Female` `Male`

`Smoker` a factor with levels `No` `Yes`

`Weekday` a factor with levels `Friday` `Saturday` `Sunday` `Thursday`

`Day_Night` a factor with levels `Day` `Night`

`Size_of_Party` a numeric vector

Details

See `TIPS` for more details. This is the same dataset except that the names of the variables are different.

`EX3.ABALONE` *ABALONE dataset for Exercise D in Chapter 3*

Description

ABALONE dataset for Exercise D in Chapter 3

Usage

```
data("EX3.ABALONE")
```

Format

A data frame with 1528 observations on the following 7 variables.

`Length` a numeric vector

`Diameter` a numeric vector

`Height` a numeric vector

`Whole.Weight` a numeric vector

`Meat.Weight` a numeric vector

`Shell.Weight` a numeric vector

`Rings` a numeric vector

Details

Abalone are sea creatures that are considered a delicacy and have very pretty iridescent shells. See https://en.wikipedia.org/wiki/Abalone. Predicting the age of the abalone from physical measurements could be useful for harvesting purposes. Dimensions are in mm and weights are in grams. `Rings` is an indicator of the age of the abalone (Age is about 1.5 plus the number of rings).

Source

Data is adapted from the abalone dataset on UCI Data Repository https://archive.ics.uci.edu/ml/datasets/Abalone. Only the male abalone are represented in this dataset.

References

See page on UCI for full details of owner and donor of this data.

EX3.BODYFAT *Bodyfat data for Exercise F in Chapter 3*

Description

Bodyfat data for Exercise F in Chapter 3

Usage

```
data("EX3.BODYFAT")
```

Format

A data frame with 20 observations on the following 4 variables.

`Triceps` a numeric vector

`Thigh` a numeric vector

`Midarm` a numeric vector

`Fat` a numeric vector

Details

Same data as `BODYFAT2`, which you can see for more details.

EX3.HOUSING *Housing data for Exercise E in Chapter 3*

Description

Housing data for Exercise E in Chapter 3

Usage

```
data("EX3.HOUSING")
```

Format

A data frame with 522 observations on the following 2 variables.

`AREA` a numeric vector, square area of house

`PRICE` a numeric vector, selling price

Details

Selling prices of houses (perhaps in the Boston area in Massachusettes).

Source

Original source unknown, but it appears in many places around the internet, e.g., `public.iastate.edu/~pdixon/stat500/c`

`EX3.NFL` *NFL data for Exercise A in Chapter 3*

Description

NFL data for Exercise A in Chapter 3

Usage

```
data("EX3.NFL")
```

Format

A data frame with 352 observations on the following 137 variables.

`Year` a numeric vector

`Team` a factor with levels `Arizona` `Atlanta` `Baltimore` `Buffalo` `Carolina` `Chicago` `Cincinnati` `Cleveland` `Dallas` `Denver` `Detroit` `GreenBay` `Houston` `Indianapolis` `Jacksonville` `KansasCity` `Miami` `Minnesota` `NewEngland` `NewOrleans` `NYGiants` `NYJets` `Oakland` `Philadelphia` `Pittsburgh` `SanDiego` `SanFrancisco` `Seattle` `St.Louis` `TampaBay` `Tennessee` `Washington`

`Next.Years.Wins` a numeric vector

`Wins` a numeric vector

`X1.Off.Tot.Yds` a numeric vector

`X2.Off.Tot.Plays` a numeric vector

`X3.Off.Tot.Yds.per.Ply` a numeric vector

`X4.Off.Tot.1st.Dwns` a numeric vector

`X5.Off.Pass.1st.Dwns` a numeric vector

`X6.Off.Rush.1st.Dwns` a numeric vector

`X7.Off.Tot.Turnovers` a numeric vector

`X8.Off.Fumbles.Lost` a numeric vector

`X9.Off.1st.Dwns.by.Penalty` a numeric vector

`X10.Off.Pass.Comp` a numeric vector

`X11.Off.Pass.Comp.` a numeric vector

`X12.Off.Pass.Yds` a numeric vector

`X13.Off.Pass.Tds` a numeric vector

`X14.Off.Pass.INTs` a numeric vector

`X15.Off.Pass.INT.` a numeric vector

`X16.Off.Pass.Longest` a numeric vector

`X17.Off.Pass.Yds.per.Att` a numeric vector

`X18.Off.Pass.Adj.Yds.per.Att` a numeric vector

`X19.Off.Pass.Yds.per.Comp` a numeric vector

`X20.Off.Pass.Yds.per.Game` a numeric vector

`X21.Off.Passer.Rating` a numeric vector

`X22.Off.Pass.Sacks.Alwd` a numeric vector

`X23.Off.Pass.Sack.Yds` a numeric vector

`X24.Off.Pass.Net.Yds.per.Att` a numeric vector

`X25.Off.Pass.Adj.Net.Yds.per.Att` a numeric vector

`X26.Off.Pass.Sack.` a numeric vector

`X27.Off.Game.Winning.Drives` a numeric vector

`X28.Off.Rush.Yds` a numeric vector

`X29.Off.Rush.Tds` a numeric vector

`X30.Off.Rush.Longest` a numeric vector

`X31.Off.Rush.Yds.per.Att` a numeric vector

`X32.Off.Rush.Yds.per.Game` a numeric vector

`X33.Off.Fumbles` a numeric vector

`X34.Off.Punt.Returns` a numeric vector

`X35.Off.PR.Yds` a numeric vector

`X36.Off.PR.Tds` a numeric vector

`X37.Off.PR.Longest` a numeric vector

`X38.Off.PR.Yds.per.Att` a numeric vector

`X39.Off.Kick.Returns` a numeric vector

`X40.Off.KR.Yds` a numeric vector

`X41.Off.KR.Tds` a numeric vector

`X42.Off.KR.Longest` a numeric vector

`X43.Off.KR.Yds.per.Att` a numeric vector

`X44.Off.All.Purpose.Yds` a numeric vector

`X45.X1.19.yd.FG.Att` a numeric vector

`X46.X1.19.yd.FG.Made` a numeric vector
`X47.X20.29.yd.FG.Att` a numeric vector
`X48.X20.29.yd.FG.Made` a numeric vector
`X49.X1.29.yd.FG.` a numeric vector
`X50.X30.39.yd.FG.Att` a numeric vector
`X51.X30.39.yd.FG.Made` a numeric vector
`X52.X30.39.yd.FG.` a numeric vector
`X53.X40.49.yd.FG.Att` a numeric vector
`X54.X40.49.yd.FG.Made` a numeric vector
`X55.X50yd.FG.Att` a numeric vector
`X56.X50yd.FG.Made` a numeric vector
`X57.X40yd.FG.` a numeric vector
`X58.Total.FG.Att` a numeric vector
`X59.Off.Tot.FG.Made` a numeric vector
`X60.Off.Tot.FG.` a numeric vector
`X61.Off.XP.Att` a numeric vector
`X62.Off.XP.Made` a numeric vector
`X63.Off.XP.` a numeric vector
`X64.Off.Times.Punted` a numeric vector
`X65.Off.Punt.Yards` a numeric vector
`X66.Off.Longest.Punt` a numeric vector
`X67.Off.Times.Had.Punt.Blocked` a numeric vector
`X68.Off.Yards.Per.Punt` a numeric vector
`X69.Fmbl.Tds` a numeric vector
`X70.Def.INT.Tds.Scored` a numeric vector
`X71.Blocked.Kick.or.Missed.FG.Ret.Tds` a numeric vector
`X72.Total.Tds.Scored` a numeric vector
`X73.Off.2pt.Conv.Made` a numeric vector
`X74.Def.Safeties.Scored` a numeric vector
`X75.Def.Tot.Yds.Alwd` a numeric vector
`X76.Def.Tot.Plays.Alwd` a numeric vector
`X77.Def.Tot.Yds.per.Play.Alwd` a numeric vector
`X78.Def.Tot.1st.Dwns.Alwd` a numeric vector
`X79.Def.Pass.1st.Dwns.Alwd` a numeric vector
`X80.Def.Rush.1st.Dwns.Alwd` a numeric vector
`X81.Def.Turnovers.Created` a numeric vector
`X82.Def.Fumbles.Recovered` a numeric vector

X83.Def.1st.Dwns.Alwd.by.Penalty a numeric vector

X84.Def.Pass.Comp.Alwd a numeric vector

X85.Def.Pass.Att.Alwd a numeric vector

X86.Def.Pass.Comp..Alwd a numeric vector

X87.Def.Pass.Yds.Alwd a numeric vector

X88.Def.Pass.Tds.Alwd a numeric vector

X89.Def.Pass.TDAlwd a numeric vector

X90.Def.Pass.INTs a numeric vector

X91.Def.Pass.INT. a numeric vector

X92.Def.Pass.Yds.per.Att.Alwd a numeric vector

X93.Def.Pass.Adj.Yds.per.Att.Alwd a numeric vector

X94.Def.Pass.Yds.per.Comp.Alwd a numeric vector

X95.Def.Pass.Yds.per.Game.Alwd a numeric vector

X96.Def.Passer.Rating.Alwd a numeric vector

X97.Def.Pass.Sacks a numeric vector

X98.Def.Pass.Sack.Yds a numeric vector

X99.Def.Pass.Net.Yds.per.Att.Alwd a numeric vector

X100.Def.Pass.Adj.Net.Yds.per.Att.Alwd a numeric vector

X101.Def.Pass.Sack. a numeric vector

X102.Def.Rush.Yds.Alwd a numeric vector

X103.Def.Rush.Tds.Alwd a numeric vector

X104.Def.Rush.Yds.per.Att.Alwd a numeric vector

X105.Def.Rush.Yds.per.Game.Alwd a numeric vector

X106.Def.Punt.Returns.Alwd a numeric vector

X107.Def.PR.Tds.Alwd a numeric vector

X108.Def.Kick.Returns.Alwd a numeric vector

X109.Def.KR.Yds.Alwd a numeric vector

X110.Def.KR.Tds.Alwd a numeric vector

X111.Def.KR.Yds.per.Att.Alwd a numeric vector

X112.Def.Tot.FG.Att.Alwd a numeric vector

X113.Def.Tot.FG.Made.Alwd a numeric vector

X114.Def.Tot.FG..Alwd a numeric vector

X115.Def.XP.Att.Alwd a numeric vector

X116.Def.XP.Made.Alwd a numeric vector

X117.Def.XP..Alwd a numeric vector

X118.Def.Punts.Alwd a numeric vector

X119.Def.Punt.Yds.Alwd a numeric vector

`X120.Def.Punt.Yds.per.Att.Alwd` a numeric vector

`X121.Def.2pt.Conv.Alwd` a numeric vector

`X122.Off.Safeties` a numeric vector

`X123.Off.Rush.Success.Rate` a numeric vector

`X124.Head.Coach.Disturbance.` a factor with levels `No` `Yes`

`X125.QB.Disturbance` a factor with levels `No` `Yes`

`X126.RB.Disturbance` a factor with levels ? `No` `Yes`

`X127.Off.Run.Pass.Ratio` a numeric vector

`X128.Off.Pass.Ply.` a numeric vector

`X129.Off.Run.Ply.` a numeric vector

`X130.Off.Yds.Pt` a numeric vector

`X131.Def.Yds.Pt` a numeric vector

`X132.Off.Pass.Drop.rate` a numeric vector

`X133.Def.Pass.Drop.Rate` a numeric vector

Details

See `NFL` for more details. This dataset is actually a more complete version of `NFL` and contains additional variables such as the year, team, next year's wins of the team, etc., and could be used in place of the `NFL` data

---

`EX4.BIKE` *Bike data for Exercise 1 in Chapter 4*

Description

Bike data for Exercise 1 in Chapter 4

Usage

```
data("EX4.BIKE")
```

Format

A data frame with 414 observations on the following 5 variables.

`Demand` a numeric vector, total number of rental bikes

`AvgTemp` a numeric vector, average temperature of the day

`EffectiveAvgTemp` a numeric vector, average temperature it feels like (taking into account dew-point) for the day

`AvgHumidity` a numeric vector, average humidity for the day

`AvgWindspeed` a numeric vector, average wind speed for the day

Details

Adapted from the bike sharing dataset on the UCI data repository http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset. This concerns the demand for rental bikes in the DC area.

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return back has become automatic. Through these systems, user is able to easily rent a bike from a particular position and return back at another position. Currently, there are about over 500 bike-sharing programs around the world which is composed of over 500 thousands bicycles. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues.

Apart from interesting real world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be detected via monitoring these data.

References

Fanaee-T, Hadi, and Gama, Joao, Event labeling combining ensemble detectors and background knowledge, Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.

`EX4.STOCKPREDICT` *Stock data for Exercise 2 in Chapter 4 (prediction set)*

Description

Stock data for Exercise 2 in Chapter 4 (prediction set)

Usage

```
data("EX4.STOCKPREDICT")
```

Format

A data frame with 5 observations on the following 40 variables.

`AAPLlag2` a numeric vector

`AXPlag2` a numeric vector

`BAlag2` a numeric vector

`BAClag2` a numeric vector

`CATlag2` a numeric vector

`CSCOlag2` a numeric vector

`CVXlag2` a numeric vector

`DDlag2` a numeric vector

DISlag2 a numeric vector
GElag2 a numeric vector
HDlag2 a numeric vector
HPQlag2 a numeric vector
IBMlag2 a numeric vector
INTClag2 a numeric vector
JNJlag2 a numeric vector
JPMlag2 a numeric vector
KOlag2 a numeric vector
MCDlag2 a numeric vector
MMMlag2 a numeric vector
MRKlag2 a numeric vector
MSFTlag2 a numeric vector
PFElag2 a numeric vector
PGlag2 a numeric vector
Tlag2 a numeric vector
TRVlag2 a numeric vector
UNHlag2 a numeric vector
VZlag2 a numeric vector
WMTlag2 a numeric vector
XOMlag2 a numeric vector
Australialag2 a numeric vector
Copperlag2 a numeric vector
DollarIndexlag2 a numeric vector
Europelag2 a numeric vector
Exchangelag2 a numeric vector
GlobalDowlag2 a numeric vector
HongKonglag2 a numeric vector
Indialag2 a numeric vector
Japanlag2 a numeric vector
Oillag2 a numeric vector
Shanghailag2 a numeric vector

Details

The data frame for which you are to predict the closing price of Alcoa stock based on the model built using EX4.STOCKS. The actual closing prices are not given.

EX4.STOCKS *Stock data for Exercise 2 in Chapter 4*

Description

Stock data for Exercise 2 in Chapter 4

Usage

```
data("EX4.STOCKS")
```

Format

A data frame with 216 observations on the following 41 variables.

`AA` a numeric vector

`AAPLlag2` a numeric vector

`AXPlag2` a numeric vector

`BAlag2` a numeric vector

`BAClag2` a numeric vector

`CATlag2` a numeric vector

`CSCOlag2` a numeric vector

`CVXlag2` a numeric vector

`DDlag2` a numeric vector

`DISlag2` a numeric vector

`GElag2` a numeric vector

`HDlag2` a numeric vector

`HPQlag2` a numeric vector

`IBMlag2` a numeric vector

`INTClag2` a numeric vector

`JNJlag2` a numeric vector

`JPMlag2` a numeric vector

`KOlag2` a numeric vector

`MCDlag2` a numeric vector

`MMMlag2` a numeric vector

`MRKlag2` a numeric vector

`MSFTlag2` a numeric vector

`PFElag2` a numeric vector

`PGlag2` a numeric vector

`Tlag2` a numeric vector

TRVlag2 a numeric vector

UNHlag2 a numeric vector

VZlag2 a numeric vector

WMTlag2 a numeric vector

XOMlag2 a numeric vector

Australialag2 a numeric vector

Copperlag2 a numeric vector

DollarIndexlag2 a numeric vector

Europelag2 a numeric vector

Exchangelag2 a numeric vector

GlobalDowlag2 a numeric vector

HongKonglag2 a numeric vector

Indialag2 a numeric vector

Japanlag2 a numeric vector

Oillag2 a numeric vector

Shanghailag2 a numeric vector

Details

The goal is to predict the closing price of Alcoa stock (AA) from the closing prices of other stocks and commodities two days prior (IMBlag2, HongKonglag2, etc.). If this were possible, and if the association between the prices continued into the future, it would be possible to use this information to make smart trades.

Source

Compiled from various sources on the internet, e.g., Yahoo historical prices.

EX5.BIKE *BIKE dataset for Exercise 4 Chapter 5*

Description

BIKE dataset for Exercise 4 Chapter 5

Usage

```
data("EX5.BIKE")
```

Format

A data frame with 413 observations on the following 9 variables.

`Demand` a numeric vector

`Day` a factor with levels `Friday` `Monday` `Saturday` `Sunday` `Thursday` `Tuesday` `Wednesday`

`Workingday` a factor with levels `no` `yes`

`Holiday` a factor with levels `no` `yes`

`Weather` a factor with levels `No rain` `Rain`

`AvgTemp` a numeric vector

`EffectiveAvgTemp` a numeric vector

`AvgHumidity` a numeric vector

`AvgWindspeed` a numeric vector

Details

Adapted from the bike sharing dataset on the UCI data repository http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset. This concerns the demand for rental bikes in the DC area. This is an expanded version of `EX4.BIKE` with more variables and without the row containing bad data.

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return back has become automatic. Through these systems, user is able to easily rent a bike from a particular position and return back at another position. Currently, there are about over 500 bike-sharing programs around the world which is composed of over 500 thousands bicycles. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues.

Apart from interesting real world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be detected via monitoring these data.

References

Fanaee-T, Hadi, and Gama, Joao, Event labeling combining ensemble detectors and background knowledge, Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.

EX5.DONOR *DONOR dataset for Exercise 4 in Chapter 5*

Description

DONOR dataset for Exercise 4 in Chapter 5

Usage

```
data("EX5.DONOR")
```

Format

A data frame with 8132 observations on the following 18 variables.

`Donate` a factor with levels `No` `Yes`

`LastAmount` a numeric vector

`AccountAge` a numeric vector

`Age` a numeric vector

`Setting` a factor with levels `Rural` `Suburban` `Urban`

`Homeowner` a factor with levels `No` `Yes`

`Gender` a factor with levels `Female` `Male` `Unknown`

`Phone` a factor with levels `Listed` `Unlisted`

`Source` a factor with levels `B` `M` `N` `P`, source from which the donor was match; B is both sources and N is neither

`MedianHomeValue` a numeric vector

`MedianIncome` a numeric vector

`PercentOwnerOccupied` a numeric vector, of the neighborhood in which donor lives

`Recent` a factor with levels `No` `Yes`

`RecentResponsePercent` a numeric vector

`RecentAvgAmount` a numeric vector

`MonthsSinceLastGift` a numeric vector

`TotalAmount` a numeric vector

`TotalDonations` a numeric vector

Details

See `DONOR` for details. This data is a subset, though attributes have been renamed.

EX6.CLICK *CLICK data for Exercise 2 in Chapter 6*

Description

CLICK data for Exercise 2 in Chapter 6

Usage

```
data("EX6.CLICK")
```

Format

A data frame with 13594 observations on the following 15 variables.

`Click` a factor with levels `No` `Yes`

`BannerPosition` a factor with levels `Pos1` `Pos2`, location of ad

`SiteID` a factor with levels `S1` `S2` `S3` `S4` `S5` `S6` `S7` `S8`

`SiteDomain` a factor with levels `SD1` `SD2` `SD3` `SD4` `SD5` `SD6` `SD7` `SD8`

`SiteCategory` a factor with levels `SCat1` `SCat2` `SCat3` `SCat4` `SCat5`

`AppDomain` a factor with levels `AD1` `AD2` `AD3`

`AppCategory` a factor with levels `AC1` `AC2`

`DeviceModel` a factor with levels `D1` `D10` `D11` `D12` `D13` `D14` `D15` `D16` `D17` `D18` `D2` `D3` `D4` `D5` `D6` `D7` `D8` `D9`

`x1` a numeric vector

`x2` a factor with levels `A` `B` `C` `D` `E` `F` `G` `H` `I` `J` `K` `L` `M` `N` `O` `P` `Q` `R`

`x3` a factor with levels `a` `b` `c` `d` `e` `f`

`x4` a factor with levels `val1` `val2` `val3`

`x5` a factor with levels `type1` `type2` `type3` `type4`

`x6` a factor with levels `class1` `class2` `class3` `class4`

`x7` a factor with levels `AA` `BB` `CC` `DD` `EE`

Details

Inspired from a competition to predict the click-thru rates of ads displayed on mobile devices `https://www.kaggle.com/c/avazu-ctr-prediction`. Does the click-thru rate vary based on where the ad placed, what kind of site and device is used to view the ad, something else? All variables are anonymized.

EX6.DONOR *DONOR dataset for Exercise 1 in Chapter 6*

Description

DONOR dataset for Exercise 1 in Chapter 6

Usage

```
data("EX6.DONOR")
```

Format

A data frame with 8132 observations on the following 18 variables.

`Donate` a factor with levels `No` `Yes`

`LastAmount` a numeric vector

`AccountAge` a numeric vector

`Age` a numeric vector

`Setting` a factor with levels `Rural` `Suburban` `Urban`

`Homeowner` a factor with levels `No` `Yes`

`Gender` a factor with levels `Female` `Male` `Unknown`

`Phone` a factor with levels `Listed` `Unlisted`

`Source` a factor with levels `B` `M` `N` `P`

`MedianHomeValue` a numeric vector

`MedianIncome` a numeric vector

`PercentOwnerOccupied` a numeric vector

`Recent` a factor with levels `No` `Yes`

`RecentResponsePercent` a numeric vector

`RecentAvgAmount` a numeric vector

`MonthsSinceLastGift` a numeric vector

`TotalAmount` a numeric vector

`TotalDonations` a numeric vector

Details

Identical to `EX5.DONOR`, so see that for details

EX6.WINE *WINE data for Exercise 3 Chapter 6*

Description

WINE data for Exercise 3 Chapter 6

Usage

```
data("EX6.WINE")
```

Format

A data frame with 2700 observations on the following 12 variables.

`Quality` a factor with levels `High` `Low`

`fixed.acidity` a numeric vector

`volatile.acidity` a numeric vector

`citric.acid` a numeric vector

`residual.sugar` a numeric vector

`free.sulfur.dioxide` a numeric vector

`total.sulfur.dioxide` a numeric vector

`density` a numeric vector

`pH` a numeric vector

`sulphates` a numeric vector

`alcohol` a numeric vector

`chlorides` a factor with levels `Little` `Lots`

Details

Adapted from the wine quality dataset at the UCI data repository. In this case, the original quality metric has been recoded from a score between 0 and 10 to either `High` or `Low`, and the `chlorides` is treated here as a categorical variable instead of a quantitative variable.

Source

https://archive.ics.uci.edu/ml/datasets/Wine+Quality

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

`EX7.BIKE` *BIKE dataset for Exercise 1 Chapters 7 and 8*

Description

BIKE dataset for Exercise 1 Chapters 7 and 8

Usage

```
data("EX7.BIKE")
```

Format

A data frame with 410 observations on the following 9 variables.

`Demand` a numeric vector

`Day` a factor with levels `Friday` `Monday` `Saturday` `Sunday` `Thursday` `Tuesday` `Wednesday`

`Workingday` a factor with levels `no` `yes`

`Holiday` a factor with levels `no` `yes`

`Weather` a factor with levels `No rain` `Rain`

`AvgTemp` a numeric vector

`EffectiveAvgTemp` a numeric vector

`AvgHumidity` a numeric vector

`AvgWindspeed` a numeric vector

Details

Identical to `EX5.BIKE` except with three additional rows deleted. See that dataset for details.

`EX7.CATALOG` *CATALOG data for Exercise 2 in Chapters 7 and 8*

Description

CATALOG data for Exercise 2 in Chapters 7 and 8

Usage

```
data("EX7.CATALOG")
```

Format

A data frame with 4000 observations on the following 7 variables.

`Buy` a factor with levels `No` `Yes`, whether customer made a purchase through the catalog next quarter

`QuartersWithPurchase` a numeric vector, number of quarters where customer made a purchase through the catalog

`PercentQuartersWithPurchase` a numeric vector, percentage of quarters where customer made a purchase through the catalog

`CatalogsReceived` a numeric vector, total number of catalogs customer has received

`DaysSinceLastPurchase` a numeric vector, number of days since customer placed his or her last order

`AvgOrderSize` a numeric vector, the typical number of items per order when customers buys through the catalog

`LifetimeOrder` a numeric vector, the number of orders the customer has placed through the catalog

Details

The original source of this data is lost, but it is likely adapted from real data.

`EX9.BIRTHWEIGHT` *Birthweight dataset for Exercise 1 in Chapter 9*

Description

Birthweight dataset for Exercise 1 in Chapter 9

Usage

```
data("EX9.BIRTHWEIGHT")
```

Format

A data frame with 553 observations on the following 13 variables.

`Birthweight` a numeric vector, grams

`Gestation` a numeric vector, weeks

`MotherRace` a factor with levels `Asian` `Black` `Mexican` `Mixed` `White`, self-reported

`MotherAge` a numeric vector, self-reported

`MotherEducation` a factor with levels `below HS` `College` `HS`, self-reported

`MotherHeight` a numeric vector, inches

`MotherWeight` a numeric vector, pounds

`FatherRace` a factor with levels `Asian` `Black` `Mexican` `Mixed` `White`, self-reported

`FatherAge` a numeric vector, self-reported

`Father_Education` a factor with levels `below HS College HS`, self-reported

`FatherHeight` a numeric vector, inches

`FatherWeight` a numeric vector, pounds

`Smoking` a factor with levels `never now`, self-reported

Details

An examination of birthweights and their link to gestation, mother and father characteristics, and whether the mother smoked during pregnancy.

Source

Adapted from a subset of a study from Nolan and Speed (2000) consisting of male, single births which survived for at least 28 days. Some rows that contained bad data have been omitted. http://had.co.nz/stat645/week-05/birthweight.txt

---

EX9.NFL *NFL data for Exercise 2 Chapter 9*

Description

NFL data for Exercise 2 Chapter 9

Usage

```
data("EX9.NFL")
```

Format

A data frame with 352 observations on the following 26 variables.

`Wins` a numeric vector

`X1.OffTotPlays` a numeric vector

`X2.OffTotYdsperPly` a numeric vector

`X3.OffPass1stDwns` a numeric vector

`X4.OffRush1stDwns` a numeric vector

`X5.OffFumblesLost` a numeric vector

`X6.OffPassComp` a numeric vector

`X7.OffPassINT` a numeric vector

`X8.OffPassLongest` a numeric vector

`X9.OffPassYdsperAtt` a numeric vector

`X10.OffPassYdsperComp` a numeric vector

`X11.OffPassSackYds` a numeric vector
`X12.OffPassSack` a numeric vector
`X13.OffRushLongest` a numeric vector
`X14.OffRushYdsperAtt` a numeric vector
`X15.OffRushYdsperGame` a numeric vector
`X16.OffFumbles` a numeric vector
`X17.1to29ydFG` a numeric vector
`X18.30to39ydFG` a numeric vector
`X19.40.ydFG` a numeric vector
`X20.TotalFGAtt` a numeric vector
`X21.OffTimesPunted` a numeric vector
`X22.OffTimesHadPuntBlocked` a numeric vector
`X23.OffYardsPerPunt` a numeric vector
`X24.Off2ptConvMade` a numeric vector
`X25.OffSafeties` a numeric vector

Details

A subset of the `NFL` data (see entry for details) containing statistics on the offense.

`EX9.STORE` *Data for Exercise 3 Chapter 9*

Description

Data for Exercise 3 Chapter 9

Usage

```
data("EX9.STORE")
```

Format

A data frame with 1500 observations on the following 68 variables.

`Store1` a factor with levels `Buy` `No`
`Store2` a factor with levels `Buy` `No`
`Store3` a factor with levels `Buy` `No`
`Store4` a factor with levels `Buy` `No`
`Store5` a factor with levels `Buy` `No`
`Store6` a factor with levels `Buy` `No`
`Store7` a factor with levels `Buy` `No`

Store8 a factor with levels Buy No

Store9 a factor with levels Buy No

Store10 a factor with levels Buy No

Store11 a factor with levels Buy No

Store12 a factor with levels Buy No

Store13 a factor with levels Buy No

Store14 a factor with levels Buy No

Store15 a factor with levels Buy No

Store16 a factor with levels Buy No

Store17 a factor with levels Buy No

Store18 a factor with levels Buy No

Store19 a factor with levels Buy No

Store20 a factor with levels Buy No

Store21 a factor with levels Buy No

Store22 a factor with levels Buy No

Store23 a factor with levels Buy No

Store24 a factor with levels Buy No

Store25 a factor with levels Buy No

Store26 a factor with levels Buy No

Store27 a factor with levels Buy No

Store28 a factor with levels Buy No

Store29 a factor with levels Buy No

Store30 a factor with levels Buy No

Store31 a factor with levels Buy No

Store32 a factor with levels Buy No

Store33 a factor with levels Buy No

Store34 a factor with levels Buy No

Store35 a factor with levels Buy No

Store36 a factor with levels Buy No

Store37 a factor with levels Buy No

Store38 a factor with levels Buy No

Store39 a factor with levels Buy No

Store40 a factor with levels Buy No

Store41 a factor with levels Buy No

Store42 a factor with levels Buy No

Store43 a factor with levels Buy No

Store44 a factor with levels Buy No

Store45 a factor with levels Buy No

Store46 a factor with levels Buy No

Store47 a factor with levels Buy No

Store48 a factor with levels Buy No

Store49 a factor with levels Buy No

Store50 a factor with levels Buy No

Store51 a factor with levels Buy No

Store52 a factor with levels Buy No

Store53 a factor with levels Buy No

Store54 a factor with levels Buy No

Store55 a factor with levels Buy No

Store56 a factor with levels Buy No

Store57 a factor with levels Buy No

Store58 a factor with levels Buy No

Store59 a factor with levels Buy No

Store60 a factor with levels Buy No

Store61 a factor with levels Buy No

Store62 a factor with levels Buy No

Store63 a factor with levels Buy No

Store64 a factor with levels Buy No

Store65 a factor with levels Buy No

Store66 a factor with levels Buy No

Store67 a factor with levels Buy No

Store68 a factor with levels Buy No

Details

The data consists of a random sample of 1500 credit card customers and their shopping habits regarding 68 different stores (whether they did or did not make a purchase in the last 90 days). Shoppers don't pick and choose places to shop at random, so it is interesting to study which stores appear together in a customers' history.

Source

Consultation with an anonymous client. Stores have been anonymized to protect the source.

extrapolation_check *A crude check for extrapolation*

Description

This function computes the Mahalanobis distance of points as a check for potential extrapolation.

Usage

```
extrapolation_check(M,newdata)
```

Arguments

`M`	A fitted model that uses only quantitative variables
`newdata`	Data frame (that has the exact same columns as predictors used to fit the model `M`) whose Mahalanobis distances are to be calculated.

Details

This function computes the shape of the predictor data cloud and calculates the distances of points from the center (with respect to the shape of the data cloud). Extrapolation occurs at a combination of predictors that is far from combinations used to build the model. An observation with a large Mahalanobis distance MAY be far from the observations used to build the model and thus MAY require extrapolation.

Note: analysis assumes the predictor data cloud is roughly elliptical (this may not be a good assumptions).

The function reports the percentiles of the Mahalanobis distances of the points in `newdata`. Percentiles are the fraction of observations used in model that are CLOSER to the center than the point(s) in question. Large values of these percentages indicate a greater risk for extrapolation. If `Percentile` is about 99 you may be extrapolating.

The method is sensitive to outliers clusters of outliers and gives only a crude idea of the potential for extrapolation.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`mahalanobis`

Examples

```
data(SALARY)
M <- lm(Salary~Education*Experience+Months,data=SALARY)
newdata <- data.frame(Education=c(0,5,10),Experience=c(15,15,15),Months=c(0,0,0))
extrapolation_check(M,newdata)
#Individuals 1 and 3 are rather unusual (though not terribly) while individual 2 is typical.
```

`find_transformations` *Transformations for simple linear regression*

Description

This function takes a simple linear regression model and finds the transformation of x and y that results in the highest R2

Usage

```
find_transformations(M,powers=seq(from=-3,to=3,by=.25),threshold=0.02,...)
```

Arguments

`M`	A simple linear regression model fitted with `lm`
`powers`	A sequence of powers to try for x and y. By default this ranges from -3 to 3 in steps of 0.25. If 0 is a valid power, then the logarithm is used instead.
`threshold`	Report all models that have an R2 that is within `threshold` of the model with the highest R2
`...`	Additional arguments to `plot` such as `pch` and `cex`.

Details

The relationship between y and x may not be linear. However, some transformation of y may have a linear relationship with some transformation of x. This function considers simple linear regression with x and y raised to powers between -3 and 3 (in 0.25 increments) by default. The function outputs a list of the top models as gauged by R^2 (all models within 0.02 of the highest R^2). Note: there is no guarantee that these "best" transformations are actually good, since a large R^2 can be produced by outliers created during transformations. A plot of the transformation is also provided.

It is exceedingly rare that the "best" transformation is raising x and y to the 1 power (i.e., the original variables). Transformations are typically used only when there are issues in the residuals plots, highly skewed variables, or physical/logical justifications.

Note: if a variable has 0s or negative numbers, only integer transformations are considered.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
 #Straightforward example
 data(BULLDOZER)
M <- lm(SalePrice~YearMade,data=BULLDOZER)
find_transformations(M,pch=20,cex=0.3)

 #Results are very misleading since selected models have high R2 due to outliers
 data(MOVIE)
 M <- lm(Total~Weekend,data=MOVIE)
find_transformations(M,powers=seq(-2,2,by=0.5),threshold=0.05)
```

FRIEND *Friendship Potential vs. Attractiveness Ratings*

Description

Examining the relationship between how likely someone would be friends with a person based on that person's level of attractiveness

Usage

```
data("FRIEND")
```

Format

A data frame with 54 observations on the following 2 variables.

`Attractiveness` a numeric vector - the average scores (1-5) from about 80 male students who rated the attractiveness of the women in each picture

`FriendshipPotential` a numeric vector - the average scores (1-5) from about 30 female students who rated how likely they would be to be friends with the pictured woman

Details

The data contain information on 54 pictures of women posted on the (now defunct/renamed) site hotornot.com. The women in two classes of introductory statistics at the University of Tennessee rated how likely they would be friends with the pictured women (on a scale of 1-5, 1 being very unlikely and 5 being very likely). The men in three (different) classes of introductory statistics gave an attractiveness score to each woman (on a scale of 1-5, 1 being very unattractive and 5 being very attractive). The numbers presented are the averages over all student ratings.

Source

Surveys administered to introductory statistics students at the University of Tennessee from 2008-2010.

FUMBLES *Wins vs. Fumbles of an NFL team*

Description

Wins vs. Fumbles of an NFL team

Usage

```
data("FUMBLES")
```

Format

A data frame with 352 observations on the following 2 variables.

`Wins` a numeric vector, number of wins (0-16) of an NFL team over the course of a season

`FumblesLost` a numeric vector, the number of fumbles lost by that team over the course of a season

Details

This is a subset of the `NFL` data. Data is from the 2002-2012 seasons.

Source

Collected by an undergraduate student from available web data in 2013.

generalization_error *Calculating the generalization error of a model on a set of data*

Description

This function takes a linear regression from `lm`, logistic regression from `glm`, partition model from `rpart`, or random forest from `randomForest` and calculates the generalization error on a dataframe.

Usage

```
generalization_error(MODEL,HOLDOUT,Kfold=FALSE,K=5,R=10,seed=NA)
```

Arguments

MODEL	A linear regression model created using lm, a logistic regression model created using glm, a partition model created using rpart, or a random forest created using randomForest.
HOLDOUT	A dataset for which the generalization error will be calculated. If not given, the error on the data used to build the model (MODEL) is used.
Kfold	If TRUE, function will estimate the generalization error of MODEL using repeated K-fold cross validation (regression models only)
K	The number of folds used in repeated K-fold cross-validation for the estimation of the generalization error for the model MODEL. It is useful to compare this number to the actual generalization error on HOLDOUT.
R	The number of repeats used in repeated K-fold cross-validation.
seed	an optional argument priming the random number seed for estimating the generalization error

Details

This function calculates the error on MODEL, its estimated generalization error from repeated K-fold cross-validation (for regression models only), and the actual generalization error on HOLDOUT. If the response is quantitative, the RMSE is reported. If the response is categorical, the confusion matrices and misclassification rates are returned.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
#Education analytics
data(STUDENT)
set.seed(1010)
train.rows <- sample(1:nrow(STUDENT),0.7*nrow(STUDENT))
TRAIN <- STUDENT[train.rows,]
HOLDOUT <- STUDENT[-train.rows,]
M <- lm(CollegeGPA~.,data=TRAIN)
#Also estimate the generalization error of the model
generalization_error(M,HOLDOUT,Kfold=TRUE,seed=5020)
#Try partition and randomforest, though they do not perform as well as regression here
TREE <- rpart(CollegeGPA~.,data=TRAIN)
FOREST <- randomForest(CollegeGPA~.,data=TRAIN)
generalization_error(TREE,HOLDOUT)
generalization_error(FOREST,HOLDOUT)

#Wine
data(WINE)
```

```
set.seed(2020)
train.rows <- sample(1:nrow(WINE),0.7*nrow(WINE))
TRAIN <- WINE[train.rows,]
HOLDOUT <- WINE[-train.rows,]
M <- glm(Quality~.^2,data=TRAIN,family=binomial)
generalization_error(M,HOLDOUT)
#Random forest predicts best on the holdout sample
TREE <- rpart(Quality~.,data=TRAIN)
FOREST <- randomForest(Quality~.,data=TRAIN)
generalization_error(TREE,HOLDOUT)
generalization_error(FOREST,HOLDOUT)
```

---

getcp *Complexity Parameter table for partition models*

Description

A simple function to take the output of a partition model created with `rpart` and return information abouthe complexity parameter and performance of varies models.

Usage

```
getcp(TREE)
```

Arguments

TREE	An object of class `rpart`. This is created by making a partition model using `rpart`.

Details

This function prints out a table of the complexity parameter, number of splits, relative error, cross validation error, and standard deviation of cross validation error for a partition model. It adds helpful advice for what the value of CP is for the tree that had the lowest cross validation error and also the value of CP for the simplest tree with a cross validation error at most 1 standard deviation above the lowest.

Further, a plot is made of the estimated generalization error (`xerror`) versus the number of splits to illustrate when the tree stops improving. Vertical lines are draw at the number of splits corresponding to the lowest estimated generalization error to the tree selected by the one standard deviation rule.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

rpart

Examples

```
 data(JUNK)
TREE <- rpart(Junk~.,data=JUNK,control=rpart.control(cp=0,xval=10,minbucket=5))
getcp(TREE)
```

`influence_plot` *Influence plot for regression diganostics*

Description

This function plots the leverage vs. deleted studentized residuals for a regression model, highlighting points that are influent based on these two factors as well as Cook's distance

Usage

```
influence_plot(M,large.cook,cooks=FALSE)
```

Arguments

`M`	A linear regression model fitted with lm()
`large.cook`	The threshold for a "large" Cook's distance. If not specified, a default of 4/n is used.
`cooks`	`TRUE` or `FALSE` (default) regarding whether to return the row numbers of observations with unusually large Cooks distances

Details

A point is influential if its addition to the data changes the regression substantially. One way of measuring influence is by looking at the point's leverage (distance from the center of the predictor's datacloud with respect to it shape) and deleted studentized residual (relative size of the residual with respect to a regression made without that point). Points with leverages larger than 2(k+1)/n (where k is the number of predictors) and deleted studentized residuals larger than 2 in magnitude are considered influential.

Influence can also be measured by Cook's distance, which essentially combines the above two measures. This function considers the Cook's distances to be large when it exceeds 4/n, but the user can specify another cutoff.

The radius of a point is proportional to the square root of the Cook's distance. Influential points according to leverage/residual criteria have an X through them while influential points according to Cook's distance are bolded.

The function returns the row numbers of influential observations.

Value

A list with the row numbers of influential points according to Cook's distance (`$Cooks`) and according to leverage/residual criteria (`$Leverage`).

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`cooks.distance`, `hatvalues`, `rstudent`

Examples

```
 data(TIPS)
 M <- lm(TipPercentage~.-Tip,data=TIPS)
influence_plot(M)
```

JUNK *Junk-mail dataset*

Description

Building a junk mail classifier based on word and character frequencies

Usage

```
data("JUNK")
```

Format

A data frame with 4601 observations on the following 58 variables.

`Junk` a factor with levels `Junk` `Safe`

`make` a numeric vector, the percentage (0-100) of words in the email that are the word `make`

`address` a numeric vector

`all` a numeric vector

`X3d` a numeric vector, the percentage (0-100) of words in the email that are the word `3d`

`our` a numeric vector

`over` a numeric vector

`remove` a numeric vector

`internet` a numeric vector

`order` a numeric vector

`mail` a numeric vector

`receive` a numeric vector

`will` a numeric vector

`people` a numeric vector

`report` a numeric vector

`addresses` a numeric vector

`free` a numeric vector

`business` a numeric vector

`email` a numeric vector

`you` a numeric vector

`credit` a numeric vector

`your` a numeric vector

`font` a numeric vector

`X000` a numeric vector, the percentage (0-100) of words in the email that are the word `000`

`money` a numeric vector

`hp` a numeric vector

`hpl` a numeric vector

`george` a numeric vector

`X650` a numeric vector

`lab` a numeric vector

`labs` a numeric vector

`telnet` a numeric vector

`X857` a numeric vector

`data` a numeric vector

`X415` a numeric vector

`X85` a numeric vector

`technology` a numeric vector

`X1999` a numeric vector

`parts` a numeric vector

`pm` a numeric vector

`direct` a numeric vector

`cs` a numeric vector

`meeting` a numeric vector

`original` a numeric vector

`project` a numeric vector

`re` a numeric vector

`edu` a numeric vector

`table` a numeric vector

`conference` a numeric vector

`semicolon` a numeric vector, the percentage (0-100) of characters in the email that are semicolons

`parenthesis` a numeric vector

`bracket` a numeric vector

`exclamation` a numeric vector

`dollarsign` a numeric vector

`hashtag` a numeric vector

`capital_run_length_average` a numeric vector, average length of uninterrupted sequence of capital letters

`capital_run_length_longest` a numeric vector, length of longest uninterrupted sequence of capital letters

`capital_run_length_total` a numeric vector, total number of capital letters in the email

Details

The collection of junk emails came from the postmaster and individuals who classified the email as junk. The collection of safe emails were from work and personal emails. Note that most of the variables are percents and can vary from 0-100, though most values are much less than 1 (1%).

Source

Adapted from the Spambase Data Set at the UCI data repository https://archive.ics.uci.edu/ml/datasets/Spambase. Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt; Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304. Donor: George Forman (gforman at nospam hpl.hp.com)

`LARGEFLYER` *Interest in frequent flier program (large version)*

Description

Interest in frequent flier program (artificial)

Usage

```
data("LARGEFLYER")
```

Format

A data frame with 100000 observations on the following 2 variables.

`Gender` a factor with levels `Female` `Male`

`Interest` a factor with levels `No` `Yes`

Details

This artificial datasets tabulates the interest in a new frequent flyer program based on gender. It illustrates that a statistically significant association may have absolutely no practical significance.

`LAUNCH` *New product launch data*

Description

The profit of newly released products over the first few months of their release

Usage

```
data("LAUNCH")
```

Format

A data frame with 652 observations on the following 420 variables.

`Profit` an anonymized numeric vector, the profit from the product over the first few months of release

`x1` an anonymized numeric vector

`x2` an anonymized numeric vector

`x3` an anonymized numeric vector

`x4` an anonymized numeric vector

`x5` an anonymized numeric vector

`x6` an anonymized numeric vector

`x7` an anonymized numeric vector

`x8` an anonymized numeric vector

`x9` an anonymized numeric vector

`x10` an anonymized numeric vector

`x11` an anonymized numeric vector

`x12` an anonymized numeric vector

`x13` an anonymized numeric vector

`x14` an anonymized numeric vector

`x15` an anonymized numeric vector

x16 an anonymized numeric vector
x17 an anonymized numeric vector
x18 an anonymized numeric vector
x19 an anonymized numeric vector
x20 an anonymized numeric vector
x21 an anonymized numeric vector
x22 an anonymized numeric vector
x23 an anonymized numeric vector
x24 an anonymized numeric vector
x25 an anonymized numeric vector
x26 an anonymized numeric vector
x27 an anonymized numeric vector
x28 an anonymized numeric vector
x29 an anonymized numeric vector
x30 an anonymized numeric vector
x31 an anonymized numeric vector
x32 an anonymized numeric vector
x33 an anonymized numeric vector
x34 an anonymized numeric vector
x35 an anonymized numeric vector
x36 an anonymized numeric vector
x37 an anonymized numeric vector
x38 an anonymized numeric vector
x39 an anonymized numeric vector
x40 an anonymized numeric vector
x41 an anonymized numeric vector
x42 an anonymized numeric vector
x43 an anonymized numeric vector
x44 an anonymized numeric vector
x45 an anonymized numeric vector
x46 an anonymized numeric vector
x47 an anonymized numeric vector
x48 an anonymized numeric vector
x49 an anonymized numeric vector
x50 an anonymized numeric vector
x51 an anonymized numeric vector
x52 an anonymized numeric vector

`x53` an anonymized numeric vector
`x54` an anonymized numeric vector
`x55` an anonymized numeric vector
`x56` an anonymized numeric vector
`x57` an anonymized numeric vector
`x58` an anonymized numeric vector
`x59` an anonymized numeric vector
`x60` an anonymized numeric vector
`x61` an anonymized numeric vector
`x62` an anonymized numeric vector
`x63` an anonymized numeric vector
`x64` an anonymized numeric vector
`x65` an anonymized numeric vector
`x66` an anonymized numeric vector
`x67` an anonymized numeric vector
`x68` an anonymized numeric vector
`x69` an anonymized numeric vector
`x70` an anonymized numeric vector
`x71` an anonymized numeric vector
`x72` an anonymized numeric vector
`x73` an anonymized numeric vector
`x74` an anonymized numeric vector
`x75` an anonymized numeric vector
`x76` an anonymized numeric vector
`x77` an anonymized numeric vector
`x78` an anonymized numeric vector
`x79` an anonymized numeric vector
`x80` an anonymized numeric vector
`x81` an anonymized numeric vector
`x82` an anonymized numeric vector
`x83` an anonymized numeric vector
`x84` an anonymized numeric vector
`x85` an anonymized numeric vector
`x86` an anonymized numeric vector
`x87` an anonymized numeric vector
`x88` an anonymized numeric vector
`x89` an anonymized numeric vector

x90 an anonymized numeric vector
x91 an anonymized numeric vector
x92 an anonymized numeric vector
x93 an anonymized numeric vector
x94 an anonymized numeric vector
x95 an anonymized numeric vector
x96 an anonymized numeric vector
x97 an anonymized numeric vector
x98 an anonymized numeric vector
x99 an anonymized numeric vector
x100 an anonymized numeric vector
x101 an anonymized numeric vector
x102 an anonymized numeric vector
x103 an anonymized numeric vector
x104 an anonymized numeric vector
x105 an anonymized numeric vector
x106 an anonymized numeric vector
x107 an anonymized numeric vector
x108 an anonymized numeric vector
x109 an anonymized numeric vector
x110 an anonymized numeric vector
x111 an anonymized numeric vector
x112 an anonymized numeric vector
x113 an anonymized numeric vector
x114 an anonymized numeric vector
x115 an anonymized numeric vector
x116 an anonymized numeric vector
x117 an anonymized numeric vector
x118 an anonymized numeric vector
x119 an anonymized numeric vector
x120 an anonymized numeric vector
x121 an anonymized numeric vector
x122 an anonymized numeric vector
x123 an anonymized numeric vector
x124 an anonymized numeric vector
x125 an anonymized numeric vector
x126 an anonymized numeric vector

x127 an anonymized numeric vector
x128 an anonymized numeric vector
x129 an anonymized numeric vector
x130 an anonymized numeric vector
x131 an anonymized numeric vector
x132 an anonymized numeric vector
x133 an anonymized numeric vector
x134 an anonymized numeric vector
x135 an anonymized numeric vector
x136 an anonymized numeric vector
x137 an anonymized numeric vector
x138 an anonymized numeric vector
x139 an anonymized numeric vector
x140 an anonymized numeric vector
x141 an anonymized numeric vector
x142 an anonymized numeric vector
x143 an anonymized numeric vector
x144 an anonymized numeric vector
x145 an anonymized numeric vector
x146 an anonymized numeric vector
x147 an anonymized numeric vector
x148 an anonymized numeric vector
x149 an anonymized numeric vector
x150 an anonymized numeric vector
x151 an anonymized numeric vector
x152 an anonymized numeric vector
x153 an anonymized numeric vector
x154 an anonymized numeric vector
x155 an anonymized numeric vector
x156 an anonymized numeric vector
x157 an anonymized numeric vector
x158 an anonymized numeric vector
x159 an anonymized numeric vector
x160 an anonymized numeric vector
x161 an anonymized numeric vector
x162 an anonymized numeric vector
x163 an anonymized numeric vector

x164 an anonymized numeric vector
x165 an anonymized numeric vector
x166 an anonymized numeric vector
x167 an anonymized numeric vector
x168 an anonymized numeric vector
x169 an anonymized numeric vector
x170 an anonymized numeric vector
x171 an anonymized numeric vector
x172 an anonymized numeric vector
x173 an anonymized numeric vector
x174 an anonymized numeric vector
x175 an anonymized numeric vector
x176 an anonymized numeric vector
x177 an anonymized numeric vector
x178 an anonymized numeric vector
x179 an anonymized numeric vector
x180 an anonymized numeric vector
x181 an anonymized numeric vector
x182 an anonymized numeric vector
x183 an anonymized numeric vector
x184 an anonymized numeric vector
x185 an anonymized numeric vector
x186 an anonymized numeric vector
x187 an anonymized numeric vector
x188 an anonymized numeric vector
x189 an anonymized numeric vector
x190 an anonymized numeric vector
x191 an anonymized numeric vector
x192 an anonymized numeric vector
x193 an anonymized numeric vector
x194 an anonymized numeric vector
x195 an anonymized numeric vector
x196 an anonymized numeric vector
x197 an anonymized numeric vector
x198 an anonymized numeric vector
x199 an anonymized numeric vector
x200 an anonymized numeric vector

x201 an anonymized numeric vector
x202 an anonymized numeric vector
x203 an anonymized numeric vector
x204 an anonymized numeric vector
x205 an anonymized numeric vector
x206 an anonymized numeric vector
x207 an anonymized numeric vector
x208 an anonymized numeric vector
x209 an anonymized numeric vector
x210 an anonymized numeric vector
x211 an anonymized numeric vector
x212 an anonymized numeric vector
x213 an anonymized numeric vector
x214 an anonymized numeric vector
x215 an anonymized numeric vector
x216 an anonymized numeric vector
x217 an anonymized numeric vector
x218 an anonymized numeric vector
x219 an anonymized numeric vector
x220 an anonymized numeric vector
x221 an anonymized numeric vector
x222 an anonymized numeric vector
x223 an anonymized numeric vector
x224 an anonymized numeric vector
x225 an anonymized numeric vector
x226 an anonymized numeric vector
x227 an anonymized numeric vector
x228 an anonymized numeric vector
x229 an anonymized numeric vector
x230 an anonymized numeric vector
x231 an anonymized numeric vector
x232 an anonymized numeric vector
x233 an anonymized numeric vector
x234 an anonymized numeric vector
x235 an anonymized numeric vector
x236 an anonymized numeric vector
x237 an anonymized numeric vector

`x238` an anonymized numeric vector
`x239` an anonymized numeric vector
`x240` an anonymized numeric vector
`x241` an anonymized numeric vector
`x242` an anonymized numeric vector
`x243` an anonymized numeric vector
`x244` an anonymized numeric vector
`x245` an anonymized numeric vector
`x246` an anonymized numeric vector
`x247` an anonymized numeric vector
`x248` an anonymized numeric vector
`x249` an anonymized numeric vector
`x250` an anonymized numeric vector
`x251` an anonymized numeric vector
`x252` an anonymized numeric vector
`x253` an anonymized numeric vector
`x254` an anonymized numeric vector
`x255` an anonymized numeric vector
`x256` an anonymized numeric vector
`x257` an anonymized numeric vector
`x258` an anonymized numeric vector
`x259` an anonymized numeric vector
`x260` an anonymized numeric vector
`x261` an anonymized numeric vector
`x262` an anonymized numeric vector
`x263` an anonymized numeric vector
`x264` an anonymized numeric vector
`x265` an anonymized numeric vector
`x266` an anonymized numeric vector
`x267` an anonymized numeric vector
`x268` an anonymized numeric vector
`x269` an anonymized numeric vector
`x270` an anonymized numeric vector
`x271` an anonymized numeric vector
`x272` an anonymized numeric vector
`x273` an anonymized numeric vector
`x274` an anonymized numeric vector

x275 an anonymized numeric vector
x276 an anonymized numeric vector
x277 an anonymized numeric vector
x278 an anonymized numeric vector
x279 an anonymized numeric vector
x280 an anonymized numeric vector
x281 an anonymized numeric vector
x282 an anonymized numeric vector
x283 an anonymized numeric vector
x284 an anonymized numeric vector
x285 an anonymized numeric vector
x286 an anonymized numeric vector
x287 an anonymized numeric vector
x288 an anonymized numeric vector
x289 an anonymized numeric vector
x290 an anonymized numeric vector
x291 an anonymized numeric vector
x292 an anonymized numeric vector
x293 an anonymized numeric vector
x294 an anonymized numeric vector
x295 an anonymized numeric vector
x296 an anonymized numeric vector
x297 an anonymized numeric vector
x298 an anonymized numeric vector
x299 an anonymized numeric vector
x300 an anonymized numeric vector
x301 an anonymized numeric vector
x302 an anonymized numeric vector
x303 an anonymized numeric vector
x304 an anonymized numeric vector
x305 an anonymized numeric vector
x306 an anonymized numeric vector
x307 an anonymized numeric vector
x308 an anonymized numeric vector
x309 an anonymized numeric vector
x310 an anonymized numeric vector
x311 an anonymized numeric vector

x312 an anonymized numeric vector
x313 an anonymized numeric vector
x314 an anonymized numeric vector
x315 an anonymized numeric vector
x316 an anonymized numeric vector
x317 an anonymized numeric vector
x318 an anonymized numeric vector
x319 an anonymized numeric vector
x320 an anonymized numeric vector
x321 an anonymized numeric vector
x322 an anonymized numeric vector
x323 an anonymized numeric vector
x324 an anonymized numeric vector
x325 an anonymized numeric vector
x326 an anonymized numeric vector
x327 an anonymized numeric vector
x328 an anonymized numeric vector
x329 an anonymized numeric vector
x330 an anonymized numeric vector
x331 an anonymized numeric vector
x332 an anonymized numeric vector
x333 an anonymized numeric vector
x334 an anonymized numeric vector
x335 an anonymized numeric vector
x336 an anonymized numeric vector
x337 an anonymized numeric vector
x338 an anonymized numeric vector
x339 an anonymized numeric vector
x340 an anonymized numeric vector
x341 an anonymized numeric vector
x342 an anonymized numeric vector
x343 an anonymized numeric vector
x344 an anonymized numeric vector
x345 an anonymized numeric vector
x346 an anonymized numeric vector
x347 an anonymized numeric vector
x348 an anonymized numeric vector

x349 an anonymized numeric vector
x350 an anonymized numeric vector
x351 an anonymized numeric vector
x352 an anonymized numeric vector
x353 an anonymized numeric vector
x354 an anonymized numeric vector
x355 an anonymized numeric vector
x356 an anonymized numeric vector
x357 an anonymized numeric vector
x358 an anonymized numeric vector
x359 an anonymized numeric vector
x360 an anonymized numeric vector
x361 an anonymized numeric vector
x362 an anonymized numeric vector
x363 an anonymized numeric vector
x364 an anonymized numeric vector
x365 an anonymized numeric vector
x366 an anonymized numeric vector
x367 an anonymized numeric vector
x368 an anonymized numeric vector
x369 an anonymized numeric vector
x370 an anonymized numeric vector
x371 an anonymized numeric vector
x372 an anonymized numeric vector
x373 an anonymized numeric vector
x374 an anonymized numeric vector
x375 an anonymized numeric vector
x376 an anonymized numeric vector
x377 an anonymized numeric vector
x378 an anonymized numeric vector
x379 an anonymized numeric vector
x380 an anonymized numeric vector
x381 an anonymized numeric vector
x382 an anonymized numeric vector
x383 an anonymized numeric vector
x384 an anonymized numeric vector
x385 an anonymized numeric vector

`x386` an anonymized numeric vector
`x387` an anonymized numeric vector
`x388` an anonymized numeric vector
`x389` an anonymized numeric vector
`x390` an anonymized numeric vector
`x391` an anonymized numeric vector
`x392` an anonymized numeric vector
`x393` an anonymized numeric vector
`x394` an anonymized numeric vector
`x395` an anonymized numeric vector
`x396` an anonymized numeric vector
`x397` an anonymized numeric vector
`x398` an anonymized numeric vector
`x399` an anonymized numeric vector
`x400` an anonymized numeric vector
`x401` an anonymized numeric vector
`x402` an anonymized numeric vector
`x403` an anonymized numeric vector
`x404` an anonymized numeric vector
`x405` an anonymized numeric vector
`x406` an anonymized numeric vector
`x407` an anonymized numeric vector
`x408` an anonymized numeric vector
`x409` an anonymized numeric vector
`x410` an anonymized numeric vector
`x411` an anonymized numeric vector
`x412` an anonymized numeric vector
`x413` an anonymized numeric vector
`x414` an anonymized numeric vector
`x415` an anonymized numeric vector
`x416` an anonymized numeric vector
`x417` an anonymized numeric vector
`x418` an anonymized numeric vector
`x419` an anonymized numeric vector

Details

This example is inspired by the Online Product Sales competition on kaggle.com. The goal is to isolate the minimum number predictors required for accurately predicting `Profit`. Since the data is based on an actual case, all predictors are anonymized (some were originally categorical but are treated as numerical for the example).

Source

Inspired by https://www.kaggle.com/c/online-sales

---

mode_factor *Find the mode of a categorical variable*

Description

This function finds the mode of a categorical variable

Usage

```
mode_factor(x)
```

Arguments

x	a factor

Details

The mode is the most frequently occuring level of a categorical variable. This function returns the mode of a categorical variable. If there is a tie for the most frequent level, it returns all modes.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
data(EX6.CLICK)
 mode_factor(EX6.CLICK$DeviceModel)

 #To see how often it appears try sorting a table
 sort( table(EX6.CLICK$DeviceModel),decreasing=TRUE )

 x <- c( rep(letters[1:4],5), "e", "f" )  #multimodel
 mode_factor(x)
```

`mosaic` *Mosaic plot*

Description

Provides a mosaic plot to visualize the association between two categorical variables

Usage

```
mosaic(formula,data,color=TRUE,labelat=c(),xlab=c(),ylab=c(),
                        magnification=1,equal=FALSE,inside=FALSE,ordered=FALSE)
```

Arguments

`formula`	A standard R formula written as y~x, where y is the name of the variable playing the role of y and x is the name of the variable playing the role of x.
`data`	An optional argument giving the name of the data frame that contains x and y. If not specified, the function will use existing definitions in the parent environment.
`color`	`TRUE` or `FALSE`. If `FALSE`, plots are presented in greyscale. If `TRUE`, an intelligent color scheme is chosen to shade the plot.
`labelat`	a vector of factor levels of `x` to be labeled (in the case that you want only certain levels to be labeled)
`xlab`	Label of horizontal axis if you want something different that the name of the `x` variable
`ylab`	Label of vertical axis if you want something different that the name of the `y` variable
`magnification`	Magnification of the labels of the `x` variable. A number smaller than 1 shrinks everything. A number larger than 1 makes everything larger
`equal`	If `FALSE`, the bar widths are proportional to the frequency of the corresponding level. If `TRUE`, the bar widths are all equal (useful if there are many levels or some are extremely rare).
`inside`	If `FALSE`, labels are beneath the bars. If `TRUE`, labels are placed inside the bars and rotated (useful if the levels have long names)
`ordered`	If `FALSE`, bars are in alphabetical order. If `TRUE`, the ordering of the bars reflects the ordering of the factor levels.

Details

This function shows a mosaic plot to visualize the conditional distributions of `y` for each level of `x`, along with the marginal distribution of `y` to the right of the plot. The widths of the segmented bar charts are proportional to the frequency of each level of `x`. These plots are the same that appear using `associate`.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

associate

Examples

```
 data(ACCOUNT)
mosaic(Area.Classification~Purchase,data=ACCOUNT,color=TRUE)

data(EX6.CLICK)
#Default presentation:  not very useful
mosaic(Click~DeviceModel,data=EX6.CLICK)
#Better presentation
mosaic(Click~DeviceModel,data=EX6.CLICK,equal=TRUE,inside=TRUE,magnification=0.8)
```

MOVIE *Movie grosses*

Description

Movie grosses from the late 1990s

Usage

```
data("MOVIE")
```

Format

A data frame with 309 observations on the following 3 variables.

`Movie` a factor giving the name of the movie

`Weekend` a numeric vector, the opening weekend gross (millions of dollars)

`Total` a numeric vector, the total US gross (millions of dollars)

Details

The goal is to predict the total gross of a movie based on its opening weekend gross.

Source

Compiled via information provided on https://www.imdb.com/

`NFL` *NFL database*

Description

Statistics for NFL teams from the 2002-2012 seasons

Usage

```
data("NFL")
```

Format

A data frame with 352 observations on the following 113 variables.

`X4.Wins` a numeric vector, number of wins (0-16) of an NFL team for the season

`X5.OffTotPlays` a numeric vector, number of total plays made on offense for the season

`X6.OffTotYdsperPly` a numeric vector

`X7.OffTot1stDwns` a numeric vector

`X8.OffPass1stDwns` a numeric vector

`X9.OffRush1stDwns` a numeric vector

`X10.OffFumblesLost` a numeric vector

`X11.OffPassComp` a numeric vector

`X12.OffPassComp` a numeric vector

`X13.OffPassYds` a numeric vector

`X14.OffPassTds` a numeric vector

`X15.OffPassTD` a numeric vector

`X16.OffPassINTs` a numeric vector

`X17.OffPassINT` a numeric vector

`X18.OffPassLongest` a numeric vector

`X19.OffPassYdsperAtt` a numeric vector

`X20.OffPassAdjYdsperAtt` a numeric vector

`X21.OffPassYdsperComp` a numeric vector

`X22.OffPasserRating` a numeric vector

`X23.OffPassSacksAlwd` a numeric vector

`X24.OffPassSackYds` a numeric vector

`X25.OffPassNetYdsperAtt` a numeric vector

`X26.OffPassAdjNetYdsperAtt` a numeric vector

`X27.OffPassSack` a numeric vector

`X28.OffRushYds` a numeric vector

X29.OffRushTds a numeric vector
X30.OffRushLongest a numeric vector
X31.OffRushYdsperAtt a numeric vector
X32.OffFumbles a numeric vector
X33.OffPuntReturns a numeric vector
X34.OffPRYds a numeric vector
X35.OffPRTds a numeric vector
X36.OffPRLongest a numeric vector
X37.OffPRYdsperAtt a numeric vector
X38.OffKRTds a numeric vector
X39.OffKRLongest a numeric vector
X40.OffKRYdsperAtt a numeric vector
X41.OffAllPurposeYds a numeric vector
X42.1to19ydFGAtt a numeric vector
X43.1to19ydFGMade a numeric vector
X44.20to29ydFGAtt a numeric vector
X45.20to29ydFGMade a numeric vector
X46.1to29ydFG a numeric vector
X47.30to39ydFGAtt a numeric vector
X48.30to39ydFGMade a numeric vector
X49.30to39ydFG a numeric vector
X50.40to49ydFGAtt a numeric vector
X51.40to49ydFGMade a numeric vector
X52.50ydFGAtt a numeric vector
X53.50ydFGAtt a numeric vector
X54.40ydFG a numeric vector
X55.OffTotFG a numeric vector
X56.OffXP a numeric vector
X57.OffTimesPunted a numeric vector
X58.OffPuntYards a numeric vector
X59.OffLongestPunt a numeric vector
X60.OffTimesHadPuntBlocked a numeric vector
X61.OffYardsPerPunt a numeric vector
X62.FmblTds a numeric vector
X63.DefINTTdsScored a numeric vector
X64.BlockedKickorMissedFGRetTds a numeric vector
X65.Off2ptConvMade a numeric vector

X66.DefSafetiesScored a numeric vector

X67.DefTotYdsAlwd a numeric vector

X68.DefTotPlaysAlwd a numeric vector

X69.DefTotYdsperPlayAlwd a numeric vector

X70.DefTot1stDwnsAlwd a numeric vector

X71.DefPass1stDwnsAlwd a numeric vector

X72.DefRush1stDwnsAlwd a numeric vector

X73.DefFumblesRecovered a numeric vector

X74.DefPassCompAlwd a numeric vector

X75.DefPassAttAlwd a numeric vector

X76.DefPassCompAlwd a numeric vector

X77.DefPassYdsAlwd a numeric vector

X78.DefPassTdsAlwd a numeric vector

X79.DefPassTDAlwd a numeric vector

X80.DefPassINTs a numeric vector

X81.DefPassINT a numeric vector

X82.DefPassYdsperAttAlwd a numeric vector

X83.DefPassAdjYdsperAttAlwd a numeric vector

X84.DefPassYdsperCompAlwd a numeric vector

X85.DefPasserRatingAlwd a numeric vector

X86.DefPassSacks a numeric vector

X87.DefPassSackYds a numeric vector

X88.DefPassNetYdsperAttAlwd a numeric vector

X89.DefPassAdjNetYdsperAttAlwd a numeric vector

X90.DefPassSack a numeric vector

X91.DefRushYdsAlwd a numeric vector

X92.DefRushTdsAlwd a numeric vector

X93.DefRushYdsperAttAlwd a numeric vector

X94.DefPuntReturnsAlwd a numeric vector

X95.DefPRTdsAlwd a numeric vector

X96.DefKickReturnsAlwd a numeric vector

X97.DefKRTdsAlwd a numeric vector

X98.DefKRYdsperAttAlwd a numeric vector

X99.DefTotFGAttAlwd a numeric vector

X100.DefTotFGAlwd a numeric vector

X101.DefXPAlwd a numeric vector

X102.DefPuntsAlwd a numeric vector

`X103.DefPuntYdsAlwd` a numeric vector

`X104.DefPuntYdsperAttAlwd` a numeric vector

`X105.Def2ptConvAlwd` a numeric vector

`X106.OffSafeties` a numeric vector

`X107.OffRushSuccessRate` a numeric vector

`X108.OffRunPassRatio` a numeric vector

`X109.OffRunPly` a numeric vector

`X110.OffYdsPt` a numeric vector

`X111.DefYdsPt` a numeric vector

`X112.HeadCoachDisturbance` a factor with levels `No` `Yes`, whether the head coached changed between this season and the last

`X113.QBDisturbance` a factor with levels `No` `Yes`, whether the quarterback changed between this season and the last

`X114.RBDisturbance` a factor with levels `?` `No` `Yes`, whether the runningback changed between this seasons and the last

`X115.OffPassDropRate` a numeric vector

`X116.DefPassDropRate` a numeric vector

Details

Data was collected from many sources on the internet by a student for use in an independent study in the spring of 2013. Abbreviations for predictor variables typically follow the full name in prior variables, e.g., KR = kick returns, PR = punt returns, XP = extra point. Data is organized by year, so rows 1-32 rows are from 2002, rows 33-64 are from 2003, etc.

Source

Contact the originator Weller Ross (jwellerross@gmail.com) for further details.

`OFFENSE` *Some offensive statistics from* `NFL` *dataset*

Description

A subset of the `NFL` dataset contain some statistics of teams on offense

Usage

```
data("OFFENSE")
```

Format

A data frame with 352 observations on the following 10 variables.

`Win` a numeric vector, number of wins of team over the season (0-16)

`FirstDowns` a numeric vector, number of first downs made over the season

`PassingYards` a numeric vector, number of passing yards over the season

`Interceptions` a numeric vector, number of times ball was intercepted on offense

`RushingYards` a numeric vector, number of rushing yards over the season

`Fumbles` a numeric vector, number of fumbles made on offense

`X1to19FGAttempts` a numeric vector, number of field goal attempts made from 1-19 yards

`X20to29FGAttempts` a numeric vector, number of field goal attemps made from 20-29 yards

`X30to39FGAttempts` a numeric vector

`X40to50FGAttempts` a numeric vector

Details

A small subset of the `NFL` dataset contain select statistics. Seasons are from 2002-2012

`outlier_demo` *Interactive demonstration of the effect of an outlier on a regression*

Description

This function shows regression lines on user-defined data before and after adding an additional point.

Usage

```
outlier_demo(cex.leg=0.8)
```

Arguments

`cex.leg`	A number specifying the magnification of legends inside the plot. Smaller numbers mean smaller font.

Details

This function allows the user to generate data by click on a plot. Once two points are added, the least squares regression line is draw. When an additional point is added, the regression line updates while also showing the line without that point. The effect of outliers on a regression line can easily be illustrated. Pressing the red UNDO button on the plot will allow you to take away recently added points for further exploration.

Note: To end the demo, you MUST click on the red box labeled "End" (or press Escape, which will return an error)

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

`overfit_demo` *Demonstration of overfitting*

Description

This function gives a demonstration of how overfitting occurs on a user-inputted dataset by showing the estimated generalization error as additional variables are added to the regression model (up to all two-way interactions).

Usage

```
overfit_demo(DF,y=NA,seed=NA,aic=TRUE)
```

Arguments

`DF`	The data frame where demonstration will occur.
`y`	The response variable (in quotes)
`seed`	Optional argument setting the random number seed if results need to be reproduced
`aic`	logical, if `FALSE` the demo will show the RMSE on the training sample instead of the AIC.

Details

This function splits `DF` in half to obtain training and holdout samples. Regression models are constructed using a forward selection procedure (adding the variable that decreases the AIC the most on the training set), starting at the naive model and terminating at the full model with all two-way interactions.

The generalization error of each model is computed on the holdout sample. The AIC (or RMSE on the training) and generalization errors are plotted versus the number of variables in the model to illustrate overfitting. Typically, the generalization error decreases at first as useful variables are added to the model, then the generalization error increases after the new variables added start to fit the quirks present only in the training data. When this happens, the model is said to be overfit.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
#Overfitting occurs after about 10 predictors (AIC begins to increase after 12/13)
data(BODYFAT)
overfit_demo(BODYFAT,y="BodyFat",seed=1010)

#Overfitting occurs after about 5 predictors
data(OFFENSE)
overfit_demo(OFFENSE,y="Win",seed=1997,aic=FALSE)
```

PIMA *Pima Diabetes dataset*

Description

Diabetes among women aged 21+ with Pima heritage

Usage

```
data("PIMA")
```

Format

A data frame with 392 observations on the following 8 variables.

`Pregnant` a numeric vector, number of times the woman has been pregnant

`Glucose` a numeric vector, plasma glucose concentration

`BloodPressure` a numeric vector, diastolic blood pressure in mm Hg

`BodyFat` a numeric vector, a measurement of the triceps skinfold thickness which is an indicator of body fat percentage

`Insulin` a numeric vector, 2-hour serum insulin

`BMI` a numeric vector, body mass index

`Age` a numeric vector, years

`Diabetes` a factor with levels `No` `Yes`

Details

Data on 768 women belonging to the Pima tribe. The purpose is to study the associations between having diabetes and various physiological characteristics. Although there are surely other factors (including genetic) that influence the chance of having diabetes, the hope is that by having women who are genetically similar (all from the Pima tribe), that these other factors are naturally accounted for.

Source

Adapted from the UCI data repository https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes. A variable measuring the “diabetes pedigree function" has been omitted.

`POISON` *Cockroach poisoning data*

Description

Dosages and mortality of cockroaches

Usage

```
data("POISON")
```

Format

A data frame with 481 observations on the following 2 variables.

`Dose` a numeric vector indicated the dosage of the poison administered to the cockroach

`Outcome` a factor with levels `Die` `Live`

Details

Artificial data illustrating a dose-reponse curve. The probability of dying is well-modeled by a logistic regression model.

`possible_regressions` *Illustrating how a simple linear/logistic regression could have turned out via permutations*

Description

This function gives a demonstration of what simple linear or logistic regression lines could have looked like "by chance" if x and y were unrelated. A scatterplot and fitted regression line is displayed along with the regression lines produced when x and y are unrelated via the permutation procedure. The sum of squared error reductions for all lines (for linear regressions) are also displayed for an informal assessement of significance.

Usage

```
possible_regressions(M,permutations=100,sse=TRUE,reduction=TRUE)
```

Arguments

M	A simple linear regression model from lm
permutations	The number of artificial samples generated with the permutation procedure to consider (each will have y and x be independent by design).
sse	Optional argument to either show or hide the histogram of sum of squared errors of the regression lines.
reduction	Optional argument that, if sse is TRUE, shows the reduction in the sum of squared errors or the raw sum of squared errors of the regressions themselves.

Details

This function gives a scatterplot and fitted regression line for M in red for a linear regression, or the fitted logistic curve (in black) for logistic regression. Then, via the permutation procedure, it generates permutations, artificial samples where the observed values of x and y are paired up at random, ensuring that no relationship exists between them. A regression is fit on this permutation sample, and the regression line is drawn in grey to illustrate how it may look "by chance" when x and y are unrelated.

If requested, a histogram of the sum of squared error reductions of each of the regressions on the permutation datasets (and the original regression in red) is displayed to allow for an informal assessement of the statistical significance of the regression.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
#A weak but statistically significant relationship
data(TIPS)
M <- lm(TipPercentage~Bill,data=TIPS)
possible_regressions(M)

#A very strong relationship
data(SURVEY10)
M <- lm(PercMoreIntelligentThan~PercMoreAttractiveThan,data=SURVEY10)
possible_regressions(M,permutations=1000)

#Show raw SSE instead of reductions
M <- lm(TipPercentage~PartySize,data=TIPS)
possible_regressions(M,reduction=FALSE)
```

PRODUCT *Sales of a product one quarter after release*

Description

Sales of a product two quarters after release

Usage

```
data("PRODUCT")
```

Format

A data frame with 2768 observations on the following 4 variables.

`Outcome` a factor with levels `fail` `success` indicating whether the product was deemed a success or failure

`Category` a factor with levels `A` `B` `C` `D`, the type of item (e.g., kitchen, toys, consumables)

`Trend` a factor with levels `down` `up`, indicating whether the sales over the first 13 weeks had an upward trend or downward trend according to a simple linear regression

`SoldWeek13` a numeric vector, the number of items sold 13 weeks after release

Details

Inspired by the dunnhumby hackathon hosted at https://www.kaggle.com/c/hack-reduce-dunnhumby-hackathon. The goal is to predict whether a product will be a success or failure half a year after its release based on its characteristics and performance during the first quarter after its release.

Source

Adapted from https://www.kaggle.com/c/hack-reduce-dunnhumby-hackathon

PURCHASE *PURCHASE data*

Description

Purchase habits of customers

Usage

```
data("PURCHASE")
```

Format

A data frame with 27723 observations on the following 6 variables.

`Purchase` a factor with levels `Buy` `No`, whether the customer made a purchase in the following 30 days

`Visits` a numeric vector, number of visits customer has made to the chain in last 90 days

`Spent` a numeric vector, amount of money customer has spent at the chain the last 90 days

`PercentClose` a numeric vector, the percentage of customers' purchases that occur within 5 miles of their home

`Closest` a numeric vector, the distance between the customer's home and the nearest store in the chain

`CloseStores` a numeric vector, the number of stores in the chain within 5 miles of the customer's home

Details

A nation-wide chain is curious as to whether it can predict whether a former customer will make a purchase at one of its stores in the next 30 days based on the customer's spending habits. Some variables are known by the chain (e.g., `Visits`) and some are available to purchase from credit card companies (e.g., `PercentClose`). Is purchasing additional information about the customer worth it?

Source

Adapted from real data on the condition that neither the name of the chain nor other parties be disclosed.

`qq` *QQ plot*

Description

A QQ plot designed with statistics students in mind

Usage

```
qq(x,ax=NA,leg=NA,cex.leg=0.8)
```

Arguments

`x`	A vector of data
`ax`	The name you want to call `x` for the x-axis (if omitted, defaults to what was passed as the first argument). Useful if the variable is a column in a dataframe.
`leg`	Optional argument that places a legend in the top left of the plot with the text given by `leg`
`cex.leg`	Optional argument that gives the magnification of the text in the legend

Details

This function gives a "QQ plot" that is more easily interpreted than the standard QQ plot. Instead of plotting quantiles, it plots the observed values of `x` versus the values expected had `x` come from a Normal distribution.

The distribution can be considered approximately Normal if the points stay within the upper/lower dashed red lines (with the possible exception at the far left/right) and if there is no overall global curvature.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
#Distribution does not resemble a Normal
data(TIPS)
qq(TIPS$Bill,ax="Bill")

#Distribution resembles aNormal
data(ATTRACTF)
qq(ATTRACTF$Score,ax="Attractiveness Score")
```

`SALARY` *Harris Bank Salary data*

Description

Harris Bank Salary data

Usage

```
data("SALARY")
```

Format

A data frame with 93 observations on the following 5 variables.

`Salary` a numeric vector, starting monthly salary in dollars

`Education` a numeric vector, years of schooling at the time of hire

`Experience` a numeric vector, number of years of previous work experience

`Months` a numeric vector, number of months after Jan 1 1969 that the individual was hired

`Gender` a factor with levels `Female` `Male`

Details

Real data used in a court lawsuit. 93 randomly selected employees of Harris Bank Chicago from 1977. Values in this data have been scaled from the original values (e.g., `Experience` in years instead of months, `Education` starts at 0 instead of 8, etc.)

Source

Adapted from the case study at http://www.stat.ualberta.ca/statslabs/casestudies/sexdiscrimination.htm

---

`see_interactions` *Examining pairwise interactions between quantitative variables for a fitted regression model*

Description

Plots all pairwise interactions present in a regression model to allow for an informal assessment of their strength. When both variables are quantitative, the implicit regression lines of y vs. x1 for a small, the median, and a large value of x2 are provided (and vice versa). If one of the variables is categorical, the implicit regression lines of y vs. x as displayed for each level of the categorical variable.

Usage

```
see_interactions(M,pos="bottomright",many=FALSE,level=0.95,...)
```

Arguments

`M`	A fitted linear regression model with interactions between quantitative variables.
`pos`	Where to put the legend, one of "topleft", "top", "topright", "left","center","right","bottomleft","bottom","
`many`	If `TRUE`, will give one pair of interaction plots per page and prompt the user to go to the next set (useful if 3+ interactions). If `FALSE`, tries to put all pairs on one plot (recommended when 1 or 2 interactions in model).
`level`	Defines what makes a "small" and "large" value of x1 and x2. By default `level` is 0.95 so that a large value is the 95th percentile and a small value is the 5th percentile.
`...`	Additional arguments to `legend`, namely `cex` to make them smaller.

Details

When determining the implicit regression lines, all variables not involved in the interaction are assumed to be equal 0 (if quantitative) or equal to the level that comes first alphabetically (if categorical). Tickmarks on the y axis are thus irrelevant and are not displayed.

The plots allow an informal assessment of the presence of an interaction between the variables x1 and x2 in the model, after accounting for the other predictors. If the implicit regression lines are nearly parallel, then the interaction is weak if it exists at all. If the implicit regression lines have noticeably different slopes, then the interaction is strong.

When an interaction is present, then the strength of the relationship between y and x1 depends on the value of x2. In other words, the difference in the average value of y between two individuals who differ in x1 by 1 unit depends on their (common) value of x2 (sometimes the expected difference is large; sometimes it is small).

If one of the variables in the interaction is cateogorical, the presence of an interaction implies that the strength of the relationship between y and x is different between levels of the categorical variable. In other words, sometimes the difference in the expected value of y between an individual with level A and an individual with level B is large and sometimes it is small (and this depends on the common value of x of the individuals we are comparing).

The command `visualize.model` gives a better representation when only two predictors are in the model.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`visualize.model`

Examples

```
 data(SALARY)
M <- lm(Salary~.^2,data=SALARY)
#see_interactions(M,many=TRUE)  #not run since it requires user input

data(STUDENT)
M <- lm(CollegeGPA~(Gender+HSGPA+Family)^2+HSGPA*ACT,data=STUDENT)
see_interactions(M,cex=0.6)
```

`see_models` *Examining model AICs from the "all possible" regressions procedure using regsubsets*

Description

This function takes the output of `regsubsets` and prints out a table of the top performing models based on AIC criteria.

Usage

```
see_models(ALLMODELS,report=0,aicc=FALSE,reltomin=FALSE)
```

Arguments

ALLMODELS	An object of class regsubsets created from regsubsets in package leaps.
report	An optional argument specifying the number of top models to print out. If left at a default of 0, the function reports all models whose AICs are within 4 of the lowest overall AIC.
aicc	Either TRUE or FALSE. If TRUE, the AICc of a model is reported instead of the AIC.
reltomin	Either TRUE or FALSE, specifying whether the actual value of the AIC is reported (FALSE) or if AICs should be reported relative to the smallest overall AIC (TRUE)

Details

This function uses the summary function applied to the output of regsubsets. The AIC is calculated to be the one obtained via extractAIC to allow for easy comparison with build.model and step.

Although the model with the lowest AIC is typically chosen when making a descriptive model, models with AICs within 2 are essentially functionally equivalent. Any model with an AIC within 2 of the smallest is a reasonable choice since there is no statistical reason to prefer one over the other. The function returns a data frame of the AIC (or AICc), the number of variables, and the predictors in the "best" models.

Recall that the function regsubsets by default considers up to 8 predictors and does not preserve model hierarchy. Interactions may appear without both component terms. Further, only a subset of the indicator variables used to represent a categorical variable may appear.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

regsubsets, extractAIC

Examples

```
 data(SALARY)
ALL <- regsubsets(Salary~.^2,data=SALARY,method="exhaustive",nbest=4)
see_models(ALL)

#By default, regsubsets considers up to 8 predictors, here it looks at up to 15
data(ATTRACTF)
ALL <- regsubsets(Score~.,data=ATTRACTF,nvmax=15,nbest=1)
```

```
see_models(ALL,aicc=TRUE,report=5)
```

`segmented_barchart` *Segmented barchart*

Description

Produces a segmented barchart of the input variable, forcing it to be categorical if necessary

Usage

```
segmented_barchart(x)
```

Arguments

x	A vector. If numerical, it is treated as categorical variable in the form of a factor

Details

Standard segmented barchart. Shaded areas are labeled with the levels they represent, and the percentage of cases with that level is labeled on the axis to the right.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
 data(STUDENT)
segmented_barchart(STUDENT$Family)  #Categorical variable
data(TIPS)
segmented_barchart(TIPS$PartySize)  #Numerical variable treated as categorical
```

SMALLFLYER *Interest in a frequent flier program (small version)*

Description

Interest in a frequent flier program (artificial)

Usage

```
data("SMALLFLYER")
```

Format

A data frame with 100 observations on the following 2 variables.

`Gender` a factor with levels `Female` `Male`

`Interest` a factor with levels `No` `Yes`

Details

This artificial datasets tabulates the interest in a new frequent flyer program based on gender. A larger version of the same data is in `LARGEFLYER`.

SOLD26 *Predicting future sales*

Description

Predicting future sales based on sales data in first quarter after release

Usage

```
data("SOLD26")
```

Format

A data frame with 2768 observations on the following 16 variables.

`SoldWeek26` a numeric vector, the number of items sold 26 weeks after release and the quantity to predict

`StoresSelling1` a numeric vector, the number of stores selling the item 1 week after release

`StoresSelling3` a numeric vector

`StoresSelling5` a numeric vector

`StoresSelling7` a numeric vector

`StoresSelling9` a numeric vector

`StoresSelling11` a numeric vector

`StoresSelling13` a numeric vector

`StoresSelling26` a numeric vector, the planned number of stores selling the item 26 weeks after release

`Sold1` a numeric vector, the number of items sold 1 week after release

`Sold3` a numeric vector

`Sold5` a numeric vector

`Sold7` a numeric vector

`Sold9` a numeric vector

`Sold11` a numeric vector

`Sold13` a numeric vector, the number of items sold 13 weeks after release

Details

Inspired by the dunnhumby hackathon hosted at https://www.kaggle.com/c/hack-reduce-dunnhumby-hackathon. The goal is to predict the number of items sold 26 weeks after released based on the characteristics of its sales during the first 13 weeks after release (along with information about how many stores are planning to sell the product 26 weeks after release).

Source

Adapted from https://www.kaggle.com/c/hack-reduce-dunnhumby-hackathon

---

`SPEED` *Speed vs. Fuel Efficiency*

Description

Speed vs. Fuel Efficiency

Usage

```
data("SPEED")
```

Format

A data frame with 40 observations on the following 2 variables.

`AverageSpeed` a numeric vector describing the average speed that the vehicle was driven

`FuelEfficiency` a numeric vector describing the measured fuel efficiency

Details

The relationship between fuel efficiency and speed is non-monotonic.

Source

Artificial

STUDENT *STUDENT data*

Description

Data on the College GPAs of students in an introductory statistics class

Usage

```
data("STUDENT")
```

Format

A data frame with 607 observations on the following 19 variables.

`CollegeGPA` a numeric vector

`Gender` a factor with levels `Female` `Male`

`HSGPA` a numeric vector, can range up to 5 if the high school allowed it

`ACT` a numeric vector, ACT score

`APHours` a numeric vector, number of AP hours student took in HS

`JobHours` a numeric vector, number of hours student currently works on average

`School` a factor with levels `Private` `Public`, type of HS

`Languages` a numeric vector

`Honors` a numeric vector, number of honors classes taken in HS

`Smoker` a factor with levels `No` `Yes`

`AffordCollege` a factor with levels `No` `Yes`, can the student and his/her family pay for the University of Tennessee without taking out loans?

`HSClubs` a numeric vector, number of clubs belonged to in HS

`HSJob` a factor with levels `No` `Yes`, whether the student maintained a job at some point while in HS

`Churchgoer` a factor with levels `No` `Yes`, answer to the question Do you regularly attend chruch?

`Height` a numeric vector (inches)

`Weight` a numeric vector (lbs)

`Class` a factor with levels `Junior` `Senior` `Sophomore`

`Family` what position they are in the family, a factor with levels `Middle Child` `Oldest Child` `Only Child` `Youngest Child`

`Pet` favorite pet, a factor with levels `Both` `Cat` `Dog` `Neither`

Details

Same data as `EDUCATION` with the addition of the `Class` variable and with slighly different names for variables.

Source

Responses are from students in an introductory statistics class at the University of Tennessee in 2010.

`suggest_levels` *Combining levels of a categorical variable*

Description

This function determines levels that are similar to each other either in terms of their average value of some quantitative variable or the percentages of each level of a two-level categorical variable. Use it to get a rough idea of what levels are "about the same" with regard to some variable.

Usage

```
suggest_levels(formula,data,maxlevels=NA,target=NA,recode=FALSE,plot=TRUE,...)
```

Arguments

`formula`	A standard R formula written as y~x. Here, x is the variable whose levels you wish to combine, and y is the quantitative or two-level categorical variable.
`data`	An optional argument giving the name of the data frame that contains x and y. If not specified, the function will use existing definitions in the parent environment.
`maxlevels`	The maximum number of combined levels to consider (cannot exceed 26).
`target`	The number of resulting levels into which the levels of x will be combined. Will default to the suggested value of the fewest number whose resulting BIC is no more than 4 above the lowest BIC of any combination.
`recode`	`TRUE` or `FALSE`. If `TRUE`, the function outputs a conversion table as well as the new level identities
`plot`	`TRUE` or `FALSE`. If `TRUE`, a plot is provided which shows the distribution of `y` for each level of `x` and lines showing which levels are grouped together.
`...`	Additional arguments used to make the plot. Typically this will be `equal=TRUE` and `inside=TRUE` to be passed to `mosaic`.

Details

This function calculates the average value (or percentage of each level) of y for each level of x. It then builds a partition model taking y to be this average value (or percentage) with x being the predictor variable. The first split yields the "best" scheme for combining levels of x into 2 values. The second split yields the "best" scheme for combining levels of x into 3 values, etc.

The argument `maxlevels` specifies the maximum numbers of levels in the combination scheme. By default, it will use the number of levels of x (ie, no combination). Setting this to a lower number saves time, since most likely a small number of combined levels is desired. This is useful for seeing how different combination schemes compare.

The argument `target` will force the algorithm to producing exactly this number of combined levels. This is useful once you have determined how many levels of x you want.

If `recode` is `FALSE`, a table showing the combined levels along with the "BIC" of the combination scheme (lower is better, but a difference of around 4 or less is negligible). The suggested combination will be the fewer number of levels which has as BIC no more than 4 above the scheme that gave the lowest BIC.

If `recode` is `TRUE`, a list of three elements is produced. `$Conversion1` gives a table of the Old and New levels alphabetized by Old while `$Conversion2` gives a table of the Old and New levels alphabized by New. `$newlevels` gives a factor of the cases levels under the new combination scheme. If `target` is not set, it will use the suggested number of levels.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

Examples

```
 data(DONOR)

 #Can levels of URBANICITY be treated the same with regards to probability of donation?
 #Analysis suggests yes (all levels in one)
 suggest_levels(Donate~URBANICITY,data=DONOR)

 #Can levels of URBANICITY be treated the same with regards to donation amount?
 #Analysis suggests yes, but perhaps there are four "effective levels"

 suggest_levels(Donation.Amount~URBANICITY,data=DONOR)
 SL <- suggest_levels(Donation.Amount~URBANICITY,data=DONOR,target=4,recode=TRUE)
SL$Conversion

#Add a column to the DONOR dataframe that contains these new cluster identities
 DONOR$newCLUSTER_CODE <- SL$newlevels
```

`summarize_tree` *Useful summaries of partition models from rpart*

Description

Reports the RMSE, AIC, and variable importances for a partition model or the variable importances from a random forest.

Usage

```
summarize_tree(TREE)
```

Arguments

TREE	A partition model created with `rpart` or a random forest from `randomForest`

Details

Extracts the RMSE and AIC of a partition model and the variable importances of partition models or random forests.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

`rpart`, `randomForest`

Examples

```
 data(WINE)
TREE <- rpart(Quality~.,data=WINE,control=rpart.control(cp=0.01,xval=10,minbucket=5))
summarize_tree(TREE)
RF <- randomForest(Quality~.,data=WINE,ntree=50)
summarize_tree(RF)

data(NFL)
TREE <- rpart(X4.Wins~.,data=NFL,control=rpart.control(cp=0.002,xval=10,minbucket=5))
summarize_tree(TREE)
RF <- randomForest(X4.Wins~.,data=NFL,ntree=50)
summarize_tree(RF)
```

SURVEY09 *Student survey 2009*

Description

Characteristics of students in an introductory statistics class at the University of Tennessee in 2009

Usage

```
data("SURVEY09")
```

Format

A data frame with 579 observations on the following 47 variables.

`X01.ID` a numeric vector

`X02.Gender` a factor with levels `Female` `Male`

`X03.Weight` a numeric vector, estimated weight

`X04.DesiredWeight` a numeric vector

`X05.Class` a factor with levels `Freshman` `Junior` `Senior` `Sophmore`

`X06.BornInTN` a factor with levels `No` `Yes`

`X07.Greek` a factor with levels `No` `Yes`, if the student belongs to a fraternity/sorority

`X08.UTFirstChoice` a factor with levels `No` `Yes`

`X09.Churchgoer` a factor with levels `No` `Yes`, does student attend a religious service once a week

`X10.ParentsMarried` a factor with levels `No` `Yes`

`X11.GPA` a numeric vector

`X12.SittingLocation` a factor with levels `Back` `Front` `Middle` `Varies`

`X13.WeeklyHoursStudied` a numeric vector

`X14.Scholarship` a factor with levels `No` `Yes`

`X15.FacebookFriends` a numeric vector

`X16.AgeFirstKiss` a numeric vector, age at which student had their first romantic kiss

`X17.CarYear` a numeric vector

`X18.DaysPerWeekAlcohol` a numeric vector, how many days a week student typically drinks

`X19.NumDrinksParty` a numeric vector, how many drinks student typically has when he or she goes to a party

`X20.CellProvider` a factor with levels `ATT` `Sprint` `USCellar` `Verizon`

`X21.FreqDroppedCalls` a factor with levels `Occasionally` `Often` `Rarely`

`X22.MarriedAt` a numeric vector, age by which student hopes to be married

`X23.KidsBy` a numeric vector, age by which students hopes to have kids

`X24.Computer` a factor with levels `Mac` `Windows`

`X25.FastestDrivingSpeed` a numeric vector

`X26.BusinessMajor` a factor with levels `No` `Yes`

`X27.Major` a factor with levels `Business` `NonBusiness`

`X28.TxtsPerDay` a numeric vector

`X29.FootballGames` a numeric vector, games student hopes to attend

`X30.HoursWorkOut` a numeric vector, per week

`X31.MilesToSchool` a numeric vector, each day

`X32.MoneyInBank` a numeric vector

`X33.MoneyOnHaircut` a numeric vector

`X34.PercentTuitionYouPay` a numeric vector

`X35.SongsDownloaded` a numeric vector, songs typically downloaded (legally/illegally) a month

`X36.ParentCollegeGraduate` a factor with levels `No` `Yes`

`X37.HoursSleepPerNight` a numeric vector

`X38.Last2DigitsPhone` a numeric vector

`X39.NumClassesMissed` a numeric vector

`X40.BooksReadThisYear` a numeric vector

`X41.UseChopsticks` a factor with levels `No` `Yes`

`X42.YourAttractiveness` a numeric vector, 1 (unattractive) to 5 (very attractive)

`X43.Obama` a factor with levels `No` `NotVote` `Yes`

`X44.HoursWorkedPerWeek` a numeric vector, at a job outside of a school

`X45.MoviesInTheater` a numeric vector, number watched in theater this year

`X46.KnowSomeoneH1N1` a factor with levels `No` `Yes`

`X47.ReadBeacon` a factor with levels `No` `Yes`, the school newspaper

Details

Students answered 47 questions to generate data for a project in an introductory statistics class at the University of Tennessee in the Fall of 2009. The responses here have only had minimal cleaning (negative numbers omitted) so some data is bad (e.g., a weight of 16). The questions were:

Stat 201 Fall 2009 Survey Questions 1. What section are you in? 2. Gender [Male, Female] 3. Your weight (in pounds) [0 to 500] 4. What is your desired weight (in pounds)? [0 to 1000] 5. What year are you? [Freshman, Sophomore, Junior, Senior, Other] 6. Were you born in Tennessee? [Yes, No] 7. Are you a member of a Greek social society (i.e., a Fraternity/Sorority? [Yes, No] 8. Was UT your first choice? [Yes, No] 9. Do you usually attend a religious service once a week? [Yes, No] 10. Are your parents married? [Yes, No] 11. Thus far, what is your GPA (look up on CPO if you need to)? [0 to 4] 12. Given a choice, where do you like to sit in class? [The front row, Near the front, Around the middle, Near the back, The back row, Somewhere different all the time] 13. On average, how many hours per day do you study/do homework? [0 to 24] 14. Do you receive one or more scholarships? [Yes, No] 15. How many Facebook friends do you have? Type -1 if you dont use Facebook. [-1 to 5000] 16. How old were you when you had your first romantic kiss? Type -1 if it has not happened yet. [-1 to 100] 17. What is the year of the car you drive most often? Type a four digit number. Enter 1908 if you never drive a car. [1908 to 2011] 18. On average, how many days per week do you consume one or more alcoholic beverage? Type -1 if you never drink alcoholic beverages. [-1 to 7] 19. On average, how many alcoholic drinks do you have when you party? Type -1 if you never drink alcoholic beverages. [-1 to 100] 20. Which cell phone provider do you use (the most, if you have multiple services)? [ATT (Cingular), Cricket, Sprint, T-Mobile, U.S. Cellular, Verizon, Other, I dont use a cell phone] 21. How often do you have dropped calls? [Never, Rarely, Sometimes, Often, Constantly] 22. What is the age at which you hope to be married? Type -1 if you are already married and type -2 if you never want to get married. [-2 to 100] 23. What is the age at which you hope to have your first child? Type -1 if you already have one or more children, type -2 if you never want to have children. [-2 to 100] 24. What type of computer do you use most often? [PC running Windows, PC running linux, Mac running Mac OS, Mac running linux, Mac running Windows, Other, I dont understand the choices above] 25. What is the fastest speed (in miles per hour) you have ever achieved while driving a car? [0 to 300] 26. Do you plan on going into the Business School? [Yes, No] 27. What is your desired (or actual) major? [Accounting, Economics, Finance, Logistics, Marketing, Statistics, Other] 28. How many text messages do you typically send on any given day? Type -1 if you never send text messages. [-1 to 1000] 29. How many UT football games do you hope to attend this year? (Include games already attended this year. Do not include scrimmages.) [0 to 14] 30. How many hours a week do you work out/play sports/exercise, etc.? [0 to 168] 31. How many miles do you drive to school on a typical day? [0 to 500] 32. How much money do you have in your bank account? Type -999 if you think its none of our business. [-999 to 10000000] 33. How much do you typically spend on a hair cut? [0 to 1000] 34. What percent of tuition are you personally responsible for? Type a number between 0 and 100. [0 to 100] 35. Typically, how many songs do you download a month (both legally and/or illegally)? [0 to 10000] 36. Did at least one of your parents graduate from college? [Yes, No] 37. On average, how many hours do you sleep a night? [0 to 24] 38. What are the last two digits of your phone number? (Type 0 for 00, 1 for 01, 2 for 02, etc.) [0 to 99] 39. Approximately how many classes have you missed/skipped so far this semester? (For all your courses, including absences for legitimate excuses) [0 to 150] 40. How many books (other than textbooks) have you read so far this year? [0 to 1000] 41. Are you proficient with a pair of chopsticks? [Yes, No] 42. How would you rate your attractiveness on a scale of 1 to 5, with 5 being the most attractive? [1 to 5] 43. Did you vote for Barack Obama in last Novembers election? [Yes, No I voted for someone else, No I didnt vote at all] 44. On average, how many hours do you work at a job per week? [0 to 168] 45. How many movies have you watched in theaters this year? [0 to 1000] 46. Do you personally know someone who has come down with H1N1 virus? [Yes, No] 47. Do you read the Daily Beacon on a regular basis? [Yes, No]

`SURVEY10` *Student survey 2010*

Description

Characteristics of students in an introductory statistics class at the University of Tennessee in 2010

Usage

```
data("SURVEY10")
```

Format

A data frame with 699 observations on the following 20 variables.

`Gender` a factor with levels `Female` `Male`

`Height` a numeric vector

`Weight` a numeric vector

`DesiredWeight` a numeric vector

`GPA` a numeric vector

`TxtPerDay` a numeric vector

`MinPerDayFaceBook` a numeric vector

`NumTattoos` a numeric vector

`NumBodyPiercings` a numeric vector

`Handedness` a factor with levels `Ambidextrous` `Left` `Right`

`WeeklyHrsVideoGame` a numeric vector

`DistanceMovedToSchool` a numeric vector

`PercentDateable` a numeric vector

`NumPhoneContacts` a numeric vector

`PercMoreAttractiveThan` a numeric vector

`PercMoreIntelligentThan` a numeric vector

`PercMoreAthleticThan` a numeric vector

`PercFunnierThan` a numeric vector

`SigificantOther` a factor with levels `No` `Yes`

`OwnAttractiveness` a numeric vector

Details

Students answered 50 questions to generate data for a project in an introductory statistics class at the University of Tennessee in the Fall of 2010. The data here represent a selection of the questions. The responses have been somewhat cleaned (unlike `SURVEY09`) where obviously bogus responses have been omitted, but there may still be issue.

The selected questions were:

`Gender` Gender [Male, Female] `Height` Your height (in inches) [48 to 96] `Weight` Your weight (in pounds) [0 to 500] `DesiredWeight` What is your desired weight (in pounds)? [0 to 1000] `GPA` Thus far, what is your GPA (look up on CPO if you need to)? [0 to 4] `TxtPerDay` How many text messages do you typically send on any given day? Type 0 if you never send text messages. [0 to 1000] `MinPerDayFaceBook` On average, how many minutes per day do you spend on internet social networks (such as Facebook, MySpace, Twitter, LinkedIn, etc.)? [0 to 1440] `NumTattoos` How many tattoos do you have? [0 to 100] `NumBodyPiercings` How many body piercings do you have (do not include piercings you have let heal up and are gone)? Count each piercing separately (i.e., pierced ears counts as 2 piercings). [0 to 100] `Handedness` Are you right-handed, left-handed, or ambidextrous? [Right-Handed, Left- Handed, Ambidextrous] `WeeklyHrsVideoGame` About how many hours a week do you play video games? This includes console games like Wii, Playstation, Xbox, as well as gaming apps for your phone, online games in Facebook, general computer games, etc. [0 to 168] `DistanceMovedToSchool` Go to maps.google.com or another website that provides maps. Get directions from your home address (the house/apartment/etc. you most recently lived in before coming to college) and the zip code 37996. How many miles does it say the trip is? Type the smallest number if offered multiple routes. Type 0 if you are unable to get driving directions for any reason. [0 to 5000] `PercentDateable` What percentage of people around your age in your preferred gender do you consider dateable? [0 to 100] `NumPhoneContacts` How many contacts do you have in your cell phone? Answer 0 if you don't use a cell phone, or have no contacts in your cell phone. [0 to 1000] `PercMoreAttractiveThan` What percentage of people at UT of your own gender and class level do you think you are more attractive than? [0 to 100] `PercMoreIntelligentThan` What percentage of people at UT of your own gender and class level do you think you are more intelligent than? [0 to 100] `PercMoreAthleticThan` What percentage of people at UT of your own gender and class level do you think you are more athletic than? [0 to 100] `PercFunnierThan` What percentage of people at UT of your own gender and class level do you think you are funnier than? [0 to 100]

SigificantOther Do you have a significant other? [Yes, No] OwnAttractiveness On a scale of 1-100, with 100 being the most attractive, rate your own attractiveness. [1 to 100]

SURVEY11 *Student survey 2011*

Description

Characteristics of students in an introductory statistics class at the University of Tennessee in 2011

Usage

```
data("SURVEY11")
```

Format

A data frame with 628 observations on the following 51 variables.

- `X01.ID` a numeric vector
- `X02.Gender` a factor with levels `F` `M`
- `X03.Height` a numeric vector
- `X04.Weight` a numeric vector
- `X05.SatisfiedWithWeight` a factor with levels `No I Wish I Weighed Less` `No I Wish I Weighed More` `Yes`
- `X06.Class` a factor with levels `Freshman` `Junior` `Senior` `Sophomore`
- `X07.GPA` a numeric vector
- `X08.Greek` a factor with levels `No` `Yes`
- `X09.PoliticalBeliefs` a factor with levels `Conservative` `Liberal` `Mix`
- `X10.BornInTN` a factor with levels `No` `Yes`
- `X11.HairColor` a factor with levels `Black` `Blonde` `Brown` `Red`
- `X12.GrowUpInUS` a factor with levels `No` `Yes`
- `X13.NumberHousemates` a numeric vector
- `X14.FacebookFriends` a numeric vector
- `X15.NumPeopleTalkToOnPhone` a numeric vector
- `X16.MinutesTalkOnPhone` a numeric vector
- `X17.PeopleSendTextsTo` a numeric vector
- `X18.NumSentTexts` a numeric vector
- `X19.Computer` a factor with levels `Mac` `PC`
- `X20.Churchgoer` a factor with levels `No` `Yes`
- `X21.HoursAtJob` a numeric vector
- `X22.FastestCarSpeed` a numeric vector

X23.NumTimesBrushTeeth a numeric vector

X24.SleepPerNight a numeric vector

X25.MinutesExercisingDay a numeric vector

X26.BooksReadMonth a numeric vector

X27.ShowerLength a numeric vector

X28.PercentRecordedTV a numeric vector

X29.MostMilesRunOneDay a numeric vector

X30.MorningPerson a factor with levels No Yes

X31.PercentStudentsDateable a numeric vector

X32.PercentYouAreMoreAttractive a numeric vector

X33.PercentYouAreSmarter a numeric vector

X34.RelationshipStatus a factor with levels Complicated Dating Married Single

X35.AgeFirstKiss a numeric vector

X36.WeaponAttractMate a factor with levels Humor Intelligence Looks Other

X37.NumSignificantOthers a numeric vector

X38.WeeksLongestRelationship a numeric vector

X39.NumDrinksWeek a numeric vector

X40.FavAlcohol a factor with levels Beer Liquor None Wine

X41.SpeedingTickets a numeric vector

X42.Smoker a factor with levels No Yes

X43.IllegalDrugs a factor with levels No Yes

X44.DefendantInCourt a factor with levels No Yes

X45.NightInJail a factor with levels No Yes

X46.BrokenBone a factor with levels No Yes

X47.CentsCarrying a numeric vector

X48.SawLastHarryPotter a factor with levels No Yes

X49.NumHarryPotterRead a numeric vector

X50.HoursContinuouslyAwake a numeric vector

X51.NumCountriesVisited a numeric vector

Details

Students answered 51 questions to generate data for a project in an introductory statistics class at the University of Tennessee in the Fall of 2011. The responses have been minimally modified or cleaned. The questions were:

1. What section are you in? (To be viewed only by the Stat 201 coordinator, and removed prior to distributing the data.) 2. What is your gender? [M,F] 3. What is your height (in inches)? [0,100] 4. What is your weight (in pounds)? [0,1000] 5. Are you satisfied with your current weight? [Yes, No I wish I weighed less, No I wish I weighed more] 6. What is your class level? [Freshman, Sophomore, Junior, Senior, 5+ year senior, Non-traditional] 7. What is your current GPA? [0,4] 8. Are you a

member of a fraternity/sorority? [Yes, No] 9. Overall, do you consider your social/political beliefs to be: [more liberal, more conservative, a mix of liberal and conservative views] 10. Were you born in Tennessee? [Yes, No] 11. What is your natural hair color? [Black, Brown, Red, Blond, Gray] ##There was a database error requiring Blond and Gray to be combined into one category. 12. Did you grow up in the US? [Yes, No, Some time in the US but a significant time in another country] 13. How many people share your current residence? Count yourself, so if you live alone, answer 1. Also, if you live in a dorm, count yourself plus just your roommates/suitemates. [1, 1000] 14. How many Facebook friends do you currently have? (To see how many friends you have in Facebook, open a new tab or browser window and log in to Facebook, click the down arrow next to Account, select Edit Friends, and on the left of your screen your friends count is in parentheses.) [0,10000] 15. How many people do you talk to on the phone in a typical day? [0,1000] 16. How many MINUTES a day do you typically spend on the phone talking to people? [0,1440] 17. How many different people do you typically send text messages to on a typical day? [0,1000] 18. How many total texts do you think you send to people on a typical day? [0,5000] 19. What type of computer do you use the most? [Mac, PC, Linux] 20. Do you currently attend religious services at least once a month? [Yes, No] 21. About how many HOURS PER WEEK do you work at a job? [0,168] 22. What is the fastest speed you have achieved while driving a car (in miles per hour)? [0, 500] 23. How many times per day do you typically brush your teeth? [0, 100] 24. On a typical school night, how many HOURS do you sleep? [0, 24] 25. How many MINUTES PER DAY do you typically engage in physical activity (e.g., walking to and from class, working out at the gym, sports practice, etc.)? [0, 1440] 26. How many books have you read from cover to cover over the last month for pleasure? [0, 1000] 27. How many MINUTES do you typically spend when you take a shower? [0, 1440] 28. Advertisers are concerned that people are "fast forwarding" past their TV commercials, because more and more people are recording broadcast television and watching it later (for example, on a DVR). Approximately what percent of the TV that you watch (that HAS commercials in it) is something you recorded, and therefore you can "fast forward" past the commercials? [0, 100] 29. What is the longest that you've ever walked/run/hiked in a single day (in MILES)? [0,189] 30. Do you consider yourself a "morning person"? [Yes, No] 31. What percentage of UT students in your preferred gender do you think are dateable? [0, 100] 32. What percentage of UT students do you think you are more attractive than? [0, 100] 33. What percentage of UT students do you think you are more intelligent than? [0, 100] 34. What is your relationship status? [Single, Casually dating one or more people, Dating someone regularly, Engaged, Married, It's complicated] 35. How old were you when you had your first romantic kiss? (Enter 0 if this has not yet happened.) [0, 99] 36. Which of the following would you consider to be your main weapon for attracting a potential mate? [Looks, Intelligence, Sense of Humor, Other] 37. How many boyfriends/girlfriends have you had? (We'll leave it up to you as to what constitutes a boyfriend or girlfriend.) [0, 1000] 38. What is the longest amount of time (in WEEKS) that you have been in a relationship with a significant other? (A shortcut: take the number of months and multiply by 4, or the number of years and multiply by 52.) [0, 4000] 39. How many alcoholic beverages do you typically consume PER WEEK? (consider 1 alcoholic beverage a 12 oz. beer, a 4 oz. glass of wine, a 1 oz. shot of liquor, etc.) [0, 200] 40. What is your favorite kind of alcoholic beverage? [I don't drink alcoholic beverages, Beer, Wine, Whiskey, Vodka, Gin, Tequila, Rum, Other] 41. How may speeding tickets have you received? [0, 500] 42. Do you consider yourself a "smoker"? [Yes, No] 43. Have you ever used an illegal/controlled substance? (Exclude alcohol/cigarettes consumed when underaged.) [Yes, No] 44. Have you ever appeared before a judge/jury as a defendant? (Exclude speeding or parking tickets.) [Yes, No] 45. Have you ever spent the night in a jail cell? [Yes, No] 46. Have you ever broken a bone that required surgery or a cast (or both)? [Yes, No] 47. Check your pockets and/or purse and report how much money in coins (in CENTS) that you currently are carrying. For example, if you have one quarter and one penny, type 26, not 0.26. [0, 1000] 48. Have you seen the latest Harry Potter movie that came out in July 2011? [Yes, No] 49. How many of the seven Harry Potter books have you completely read? [0, 7] 50. Estimate the longest amount of time (in HOURS) that you have continuously stayed awake. [0, 450] 51. How many countries have you ever stepped foot in outside an airport (include the US in your count)? [1, 196]

`TIPS` *TIPS dataset*

Description

One waiter recorded information about each tip he received over a period of a few months working in one restaurant. He collected several variables:

Usage

```
data("TIPS")
```

Format

A data frame with 244 observations on the following 8 variables.

`TipPercentage` a numeric vector, the tip written as a percentage (0-100) of the total bill

`Bill` a numeric vector, the bill amount (dollars)

`Tip` a numeric vector, the tip amount (dollars)

`Gender` a factor with levels `Female` `Male`, gender of the payer of the bill

`Smoker` a factor with levels `No` `Yes`, whether the party included smokers

`Weekday` a factor with levels `Friday` `Saturday` `Sunday` `Thursday`, day of the week

`Time` a factor with levels `Day` `Night`, rough time of day

`PartySize` a numeric vector, number of people in party

Source

This is the `Tips` dataset in package `reshape`, modified to include the tip percentage.

VIF *Variance Inflation Factor*

Description

Calculates the variation inflation factors of all predictors in regression models

Usage

```
VIF(mod)
```

Arguments

mod	A linear or logistic regression model

Details

This function is a simple port of `vif` from the `car` package. The VIF of a predictor is a measure for how easily it is predicted from a linear regression using the other predictors. Taking the square root of the VIF tells you how much larger the standard error of the estimated coefficient is respect to the case when that predictor is independent of the other predictors.

A general guideline is that a VIF larger than 5 or 10 is large, indicating that the model has problems estimating the coefficient. However, this in general does not degrade the quality of predictions. If the VIF is larger than 1/(1-R2), where R2 is the Multiple R-squared of the regression, then that predictor is more related to the other predictors than it is to the response.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling with R

Examples

```
#A case where the VIFs are small
data(SALARY)
M <- lm(Salary~.,data=SALARY)
VIF(M)

 #A case where (some of) the VIFs are large
 data(BODYFAT)
 M <- lm(BodyFat~.,data=BODYFAT)
 VIF(M)
```

visualize_model *Visualizations of one or two variable linear or logistic regressions or of partitions models*

Description

Provides useful plots to illustrate the inner-workings of regression models with one or two predictors or a partition model with not too many branches.

Usage

```
visualize_model(M,loc="topleft",level=0.95,cex.leg=0.7,midline=TRUE,...)
```

Arguments

`M`	A linear or logistic regression model with one or two predictors (not all categorical) produced by `lm` or `glm`, respectively, or a partition model produced by `rpart`. It is ok to pass an object made with `train` from the `caret` package if method `lm` or `glm` is used.
`loc`	The location for the legend, if one is to be displayed. Can also be "top", "topright", "left", "center", "right", "bottomleft", "bottom", or "bottomright".
`level`	The level of confidence for confidence and prediction intervals for the case of simple linear regression.
`cex.leg`	Magnification factor for text in legends. Smaller numbers indicate smaller text. Default is 0.7.
`midline`	logical, either `TRUE` (draw a dotted line at p=0.5 for logistic regression) or `FALSE` (do not draw line)
`...`	Additional arguments to `plot`. This is typically only used for logistic regression models where `xlim` is to be specified to see the entirety of the curve instead of using the default range.

Details

If `M` is a simple linear regression model, this provides a scatter plot, fitted line, and confidence/prediction intervals.

If `M` is a simple logistic regression model, this provides the fitted logistic curve.

If `M` is a regression with two quantitative predictors, this provides the implicit regression lines when one of the variables equals its 5th (small), 50th (median), and 95th (large) percentiles. The model may have interaction terms. In this case, the p-value of the interaction is output. The definition of small and large can be changed with the `level` argument.

If `M` is a regression with a quantitative predictor and a categorical predictor (with or without interactions), this provides the implicit regression lines for each level of the categorical predictor. The p-value of the effect test is displayed if an interaction is in the model.

If `M` is a partition model from `rpart`, this shows the tree.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

rpart, lm, glm

Examples

```
data(SALARY)
#Simple linear regression with 90% confidence and prediction intervals
M <- lm(Salary~Education,data=SALARY)
visualize_model(M,level=0.90,loc="bottomright")

#Multiple linear regression with two quantitative predictors (no interaction)
M <- lm(Salary~Education+Experience,data=SALARY)
visualize_model(M)

#Multiple linear regression with two quantitative predictors (with interaction)
#Take small and large to be the 25th and 75th percentiles
M <- lm(Salary~Education*Experience,data=SALARY)
visualize_model(M,level=0.75)

#Multiple linear regression with one categorical and one quantitative predictor
M <- lm(Salary~Education*Gender,data=SALARY)
visualize_model(M)

data(WINE)
#Simple logistic regression with expanded x limits
M <- glm(Quality~alcohol,data=WINE,family=binomial)
visualize_model(M,xlim=c(0,20))

#Multiple logistic regression with two quantitative predictors
M <- glm(Quality~alcohol*sulphates,data=WINE,family=binomial)
visualize_model(M,loc="left",midline=FALSE)

data(TIPS)
#Multiple logistic regression with one categorical and one quantitative predictor
#expanded x-limits to see more of the curve
M <- glm(Smoker~PartySize*Weekday,data=TIPS,family=binomial)
visualize_model(M,loc="topright",xlim=c(-5,15))

#Partition model predicting a quantitative response
TREE <- rpart(Salary~.,data=SALARY)
visualize_model(TREE)

#Partition model predicting a categorical response
TREE <- rpart(Quality~.,data=WINE)
```

```
visualize_model(TREE)
```

---

visualize_relationship *Visualizing the relationship between y and x in a partition model*

Description

Attempts to show how the relationship between y and x is being modeled in a partition or random forest model

Usage

```
visualize_relationship(TREE,interest,on,smooth=TRUE,marginal=TRUE,nplots=5,
  seed=NA,pos="topright",...)
```

Arguments

TREE	A partition or random forest model (though it works with many regression models as well)
interest	The name of the predictor variable for which the plot of y vs. x is to be made.
on	A dataframe giving the values of the other predictor variables for which the relationship is to be visualized. Typically this is the dataframe on which the partition model was built.
smooth	If `TRUE`, the relationship is plotted using a `loess` to smooth out the relationship
marginal	If `TRUE`, the modeled value of y at a particular value of x is the average of the predicted values of y over all rows which have that common value of x. If `FALSE`, then `nplots` rows from `on` will be selected and all other predictors will be fixed, showing the relationship between y and x for that particular set of characteristics.
nplots	The number of rows of `on` for which the relationship is plotted (if `marginal` is set to `FALSE`)
seed	the seed for the random number seed if reproducibility is required
pos	the location of the legend
...	additional arguments past to `plot`, namely `xlim` and `ylim`

Details

The function shows a scatterplot of y vs. x in the `on` dataframe, then shows how `TREE` is modeling the relationship between y and x with predicted values of y for each row in the data and also a curve illustrating the relationship. It is useful for seeing what the relationship between y and x as modeled by `TREE` "looks like", both as a whole and for particular combinations of other variables. If `marginal` is `FALSE`, then differences in the curves indicate the presence of some interaction between x and another variable.

Author(s)

Adam Petrie

References

Introduction to Regression and Modeling

See Also

loess, lm, glm

Examples

```
data(SALARY)
FOREST <- randomForest(Salary~.,data=SALARY)
visualize_relationship(FOREST,interest="Experience",on=SALARY)
visualize_relationship(FOREST,interest="Months",on=SALARY,xlim=c(1,15),ylim=c(2500,4500))

data(WINE)
TREE <- rpart(Quality~.,data=WINE)
visualize_relationship(TREE,interest="alcohol",on=WINE,smooth=FALSE)
visualize_relationship(TREE,interest="alcohol",on=WINE,marginal=FALSE,nplots=7,smooth=FALSE)
```

WINE *WINE data*

Description

Predicting the quality of wine based on its chemical characteristics

Usage

```
data("WINE")
```

Format

A data frame with 2700 observations on the following 12 variables.

`Quality` a factor with levels `high` `low`

`fixed.acidity` a numeric vector

`volatile.acidity` a numeric vector

`citric.acid` a numeric vector

`residual.sugar` a numeric vector

`chlorides` a numeric vector

`free.sulfur.dioxide` a numeric vector

`total.sulfur.dioxide` a numeric vector

`density` a numeric vector

`pH` a numeric vector

`sulphates` a numeric vector

`alcohol` a numeric vector

Details

This is the famous wine dataset from the UCI data repository https://archive.ics.uci.edu/ml/datasets/Wine+Quality with some modifications. Namely, the quality in the original data was a score between 0 and 10. These has been coded as either high or low. See description on UCI for description of variables.

References

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

Index