Package 'LSTbook'

July 21, 2025

Title Data and Software for ``Lessons in Statistical Thinking"

Version 0.6

Description ``Lessons in Statistical Thinking" D.T. Kaplan (2014) <https://dtkaplan.github.io/Lessons-in-statistical-thinking/> is a textbook for a first or second course in statistics that embraces data wrangling, causal reasoning, modeling, statistical adjustment, and simulation. 'LSTbook' supports the student-centered, tidy, pipeline-oriented computing style featured in the book.

Encoding UTF-8

Depends R (>= 3.5.0)

LazyData true

LazyDataCompression xz

Imports rlang, dplyr, ggplot2 (>= 3.4.4), broom, glue, stats, MASS, tibble, stringi

RoxygenNote 7.2.1

Suggests igraph, mosaicData, moderndive, palmerpenguins, stringdist, rmarkdown, knitr, testthat

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/dtkaplan/LSTbook

BugReports https://github.com/dtkaplan/LSTbook/issues

License MIT + file LICENSE

NeedsCompilation no

Author Daniel Kaplan [aut, cre], Randall Pruim [aut]

Maintainer Daniel Kaplan <dtkaplan@gmail.com>

Repository CRAN

Date/Publication 2024-12-07 13:40:03 UTC

Contents

AAUP *1984 salaries in various professional fields*

Description

These data were published in the American Association of University Professsor's journal, *Academe*. There were compiled by Marcia Bellas drawing on data from the 1984 Carnegie survey of faculty, the US National Science Foundation, the National Research Council, and the US Census Bureau. The motivation for the work was to investigate salary "disparities among faculty whose education and experience are comparable and whose duties are broadly similar," in particular those due to sex. Regretably, the data do not include measures of the production of workers in the various fields or the numbers of people employed in each field.

Usage

```
data(AAUP)
```

Format

28 rows, each of which is a professional discipline:

- `subject` name of the discipline
- `ac`: average salary (USD) for academics
- `nonacsal` median salary (USD) for non-academics
- `fem`: fraction of the workforce that is female
- `unemp`: unemployment rate in the discipline
- `nonac`: fraction of the workforce that is non-academic,
- `licensed`: Does work in the profession require a license (from George Cobb's paper)

Source

George Cobb (2011) "Teaching statistics: some important tensions" *Chilean Journal of Statistics* **2**(1):31-62 link

References

M Bellas & BF Reskin (1994) "On comparable worth" *Academe* **80**:83-85

`add_plot_labels` *Convenience function for adding labels to point_plot or others without needing the ggplot2 + pipe.*

Description

Convenience function for adding labels to point_plot or others without needing the ggplot2 + pipe.

Usage

```
add_plot_labels(P, ..., color = NULL)
```

Arguments

`P`	A ggplot2 object, for instance as made with `point_plot()` or `model_plot()`
`color`	Name for color legend (works for `point_plot()`)
`...`	Label items (e.g. `x = "hello"`) as in ggplot2::labs

Value

A ggplot graphics object

Examples

```
mtcars |> point_plot(mpg ~ hp + cyl) |>
  add_plot_labels(x = "The X axis", y = "Vertical", color = "# cylinders")
```

`add_slope_rose` *Add a slope "rose" to a plot.*

Description

To guide a reader in quantifying the slope of components of an x-y graph, a "slope rose" is helpful. Several radiating lines are drawn, each marked with a numerical slope. A suitable choice of slopes is made automatically, based on the x- and y- scale of the plot.

Usage

```
add_slope_rose(
  P,
  x = NULL,
  y = NULL,
  scale = 1/4,
  color = "red",
  keepers = c("both", "pos", "neg")
)

add_violin_ruler(
  P,
  x = NULL,
  y = NULL,
  width = 1/10,
  ticks = seq(0, 1, by = 0.1),
  ...
)
```

Arguments

P	a ggplot2 object made by the ggplot2 or ggformula packages
x	the x-position of the rose. This will be assigned automatically if x isn't specified.
y	the y-position of the rose, just like x.
scale	the size of the rose as a fraction of the plot area covered (default 1/4)
color	text string (e.g. "blue") for the rose
keepers	whether to show "both" positive and negative slopes or just show the "pos" or the "neg"
width	for rulers, the distance between tick marks (in native units, where categories are separated by a distance of 1.)
ticks	Integers, typically 0:5, that label the ticks.
...	additional graphical parameters, e.g. color = "blue"

Details

For the ruler, x gives the position of the root of the ruler, with the rest of the ruler moving off to the left. (For vertically oriented rulers, use a negative width.)

Value

A ggplot graphics object

Note

Use the pipe operator to send a previously made plot to have a rose added. Don't use the {ggplot2} + connector.

Examples

```
mtcars |> point_plot(mpg ~ hp, annot="model") |> add_slope_rose()
mtcars |> point_plot(wt ~ hp) |> add_slope_rose(keepers="pos", color="blue", x=100, scale=.5 )
```

`Anthro_F` *Anthropometric data from college-aged women*

Description

Percentage of body fat, age, weight, height, body mass index and fourteen circumference measurements are given for 184 college women ages 18-25. Body fat was accurately determined by an underwater weighing technique which requires special equipment and training of the individuals conducting the process. Circumference measurements were made to the nearest 0.1 cm using a cloth tape in complete contact with the skin but without compression of the soft tissues. The measurement process, described somewhat incompletely below, is described in greater detail in Slack (1997) who used the standards recommended by Lohman, Roche and Martrell (1988).

Usage

```
data(Anthro_F)
```

Format

A data.frame with one row for each of 184 woman

- Weight (kg)
- Height (m)
- BMI: (Body Mass Index) Weight divided by the square of Height
- Age
- Neck: Minimal circumference perpendicular to the long axis of the neck (cm)
- Chest: Horizontal plane measurement at the sixth rib, at the end of a normal expiration (cm)
- Calf: Horizontal maximal calf measurement (cm)
- Biceps: Measurement with arm extended (cm)
- Hips: Horizontal maximal measurement around buttocks (cm)
- Waist: Horizontal minimal measurement, at the end of a normal expiration (cm)
- Forearm: Maximal measurement perpendicular to long axis (cm)
- PThigh: (Proximal Thigh) Horizontal measurement immediately distal to the gluteal furrow (cm)
- MThigh: (Middle Thigh) Measurement midway between the midpoint of the inguinal crease and the proximal border of the patella (cm)
- DThigh: (Distal Thigh) Measurement proximal to the femoral epicondyles (cm)
- Wrist: Measurement perpendicular to the long axis of the forearm (cm)

- Knee: Measurement at the mid-patellar level, with the knee slightly flexed (cm)
- Elbow: A minimal circumference measurement with the elbow extended (cm)
- Ankle: Minimal circumference measurement perpendicular to the long axis of the calf (cm)
- BFat: Amount of body fat expressed as a percentage of total body weight, using Siri's (1956) method

Source

Roger W. Johnson (2021) "Fitting Percentage of Body Fat to Simple Body Measurements: College Women" *Journal of Statistics and Data Science Education* **29**(3) doi:10.1080/26939169.2021.1971585

`Birdkeepers` *Birdkeeping and Lung Cancer*

Description

A 1972–1981 health survey in The Hague, Netherlands, discovered an association between keeping pet birds and increased risk of lung cancer. To investigate birdkeeping as a risk factor, researchers conducted a *case–control* study of patients in 1985 at four hospitals in The Hague (population 450,000). They identified 49 cases of lung cancer among the patients who were registered with a general practice, who were age 65 or younger and who had resided in the city since 1965. They also selected 98 controls from a population of residents having the same general age structure.

Usage

```
data(Birdkeepers)
```

Format

A data frame with 147 observations on the following 7 variables.

- LC Whether subject has lung cancer
- FM Sex of subject
- SS Socioeconomic status, determined by occupation of the household's principal wage earner
- BK Indicator for birdkeeping (caged birds in the home for more that 6 consecutive months from 5 to 14 years before diagnosis (cases) or examination (control))
- AG Age of subject (in years)
- YR Years of smoking prior to diagnosis or examination
- CD Average rate of smoking (in cigarettes per day)

Details

This dataset is copied and renamed from the Sleuth3 R package, where it is called case2002.

Source

Ramsey, F.L. and Schafer, D.W. (2013). *The Statistical Sleuth: A Course in Methods of Data Analysis (3rd ed)*, Cengage Learning.

References

Holst, P.A., Kromhout, D. and Brand, R. (1988). "For Debate: Pet Birds as an Independent Risk Factor for Lung Cancer" *British Medical Journal* **297**: 13–21.

`Births2022` *Records on births in the US in 2022*

Description

These data come from the Centers for Disease Controls "public use file" recording all 3,699,040 (known) births in the US in 2022. `Births2022` is a random sample of size 20,000 from the comprehensive file

Usage

```
Births2022
```

Format

A data frame with 20,000 observations on the following 38 variables. The unit of observation is a birth.

- `month`: 1-12
- `dow`: Day of week: Sun, Mon, Tues, ...
- `place`: hospital, home, clinic, etc.
- `paternity`: is paternity acknowledged. Y, N, and X. X stands for "not applicable" which is shorthand for the mother is married (consequently the husband is presumed to be the father).
- `meduc`: mother's educational level. <8 is 8th grade or less, HSG+ means high school plus some college but no degree.
- `feduc`: father's education. Same coding as `meduc`.
- `married`: Is the mother married?
- `mage`: mother's age
- `fage`: father's age
- `total_kids`: how many total births to mother including this one.
- `interval`: months since last birth (if applicable).
- `prenatal_start`: Which trimester did the mother start prenatal care?
- `prenatal_visits`: How many total prenatal care visits.
- `mheight`: Mother's height in inches

- wt_pre: Mother's weight in pounds before pregnancy
- wt_delivery: Mother's weight in pounds at delivery
- diabetes_pre: Did the mother have diabetes before pregnancy
- diabetes_gest: Did the mother develop gestational diabetes
- hbp_pre: Did the mother have high blood pressure before pregancy
- hbp_gest: Did the mother develop high blood pressure during pregnancy
- eclampsia: Did the mother develop eclampsia
- induction: Was labor induced?
- augmentation: Was the uterus stimulated to increase frequency, duration, and intensity of contractions.
- anesthesia: Was the mother given anesthesia?
- presentation: Baby's presentation at birth (e.g. cephalic or breech)
- method: method of delivery (vaginal or C-section)
- trial_at_labor: For mother's who delivered by C-section, was there an attempt at labor.
- attendant: MD, DO, midwife, other
- payment: How was the bill paid?
- apgar5, apgar10: APGAR scores (0-10) at five and ten minutes after birth.
- plurality: singletons, twins, triplets, quadruplets (as an integer 1-4)
- sex: Baby's sex
- duration: Duration of gestation, in weeks by "obstretric estimate."
- menses: Last normal menses month: 1-12 (Jan-Dec)
- weight: Baby's weight (in grams)
- living: Baby living at time of birth report
- breastfed: Baby breastfed at time of discharge

Source

US Centers for Disease Control "Natality Public Use File", CDC vital stats online

References

"User Guide to the 2022 Natality Public Use File"

Boston_marathon *Winning times in the Boston Marathon*

Description

The Boston marathon is the oldest continuing marathon in the US.

Usage

```
Boston_marathon
```

Format

A data frame

- year
- name: the winner's name
- country from which the winner registered
- time: the winning time
- sex: female or male
- minutes: the winning time converted to minutes

Source

Boston Athletic Association

Butterfly *World records in the 100 & 200 m butterfly swim*

Description

World records in the 100 & 200 m butterfly swim

Usage

```
data(Butterfly)
```

Format

A data.frame object with one row for each world record and variables

- `time`: the record time in seconds
- `swimmer` the name of the swimmer
- `date` a Date object containing the date the record was made
- `place` string descripting the location
- `sex`: coded as `F` and `M`
- `lengths`: the total distance was divided into lengths of either 25 or 50 meters. `lengths` gives the number of such lengths in the total distance.
- `dist`: the total distance (in meters) of the race

Source

Wikipedia

`Calif_precip` *Annual precipitation in California locations*

Description

These data are from an article in the journal *Geography* that illustrates precipitation modeling.

Usage

```
Calif_precip
```

Format

A data.frame with 30 rows, each a weather station in California

- `station`: the name of the station
- `precip` average annual precipitation in inches
- `altitude` in feet
- `latitude` the station's north-south location (degrees North)
- `distance`: distance in miles from the coast
- `orientation`: related to the rain shadow effect of the mountains. "W" means westward facing (toward the prevailing winds). "L" mean "leeward," that is, facing away from the prevailing winds.

Source

P. J. Taylor (1980) "A Pedagogic Application of Multiple Regression Analysis: Precipitation in California" *Geography* **65** (3) 203-212

Callback *Resume Experiment Data*

Description

Data from an experimental study in which researchers sent the resumes of fictitious job applicants to potential employers. The first names of the fictitious applicants was set randomly to sound either Black or white.

Usage

```
data("Callback")

Names_and_race
```

Format

: `Callback`: A data frame with 4870 rows and 4 variables. Each row is one fictitious applicant

- name: first name of the fictitious job applicant
- sex: sex of applicant (female or male)
- callback: whether the potential employer called back to follow up. (1 = yes, 0 = no) Another data frame, `Names_and_race`: which first names are associated with which race.

An object of class `grouped_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 36 rows and 2 columns.

References

- Imai, Kosuke. 2017. Quantitative Social Science: An Introduction. Princeton University Press. URL from whence these data were added to this package. In QSS, the data are called `resume`.
- Marianne Bertrand and Sendhil Mullainathan (2004) "Are Emily and Greg more employable than Lakisha and Jamal? A field experiment on labor market discrimination." American Economic Review, vol. 94, no. 4, pp. 991–1013. doi: 10.3386/w9873

categorical *Helpers for specifying nodes in simulations*

Description

Helpers for specifying nodes in simulations

Mix two variables together. The output will have the specified R-squared with var1 and variance one.

Evaluate an expression separately for each case

Usage

```
categorical(n = 5, ..., exact = TRUE)

cat2value(variable, ...)

bernoulli(n = 0, logodds = NULL, prob = 0.5, labels = NULL)

mix_with(signal, noise = NULL, R2 = 0.5, var = 1, exact = FALSE)

each(ex)

block_by(block_var, levels = c("treatment", "control"), show_block = FALSE)

random_levels(n, k = NULL, replace = FALSE)
```

Arguments

n	The symbol standing for the number of rows in the data frame to be generated by datasim_run(). Just use n as a symbol; don't assign it a value. (That will be done by datasim_run().)
exact	if TRUE, make R-squared or the target variance exactly as specified.
variable	a categorical variable
logodds	Numerical vector used to generate bernouilli trials. Can be any real number.
prob	An alternative to logodds. Values must be in [0,1].
labels	Character vector: names for categorical levels, also used to replace 0 and 1 in bernouilli()
signal	The part of the mixture that will be correlated with the output.
noise	The rest of the mixture. This will be **uncorrelated** with the output only if you specify it as pure noise.
R2	The target R-squared.
var	The target variance.
ex	an expression potentially involving other variables.
block_var	Which variable to use for blocking
levels	Character vector giving names to the blocking levels
show_block	Logical. If TRUE, put the block number in the output.
k	Number of distinct levels
replace	if TRUE, use resampling on the set of k levels
...	assignments of values to the names in variable

Details

datasim_make() constructs a simulation which can then be run with datasim_run(). Each argument to datasim_make() specifies one node of the simulation using an assignment-like syntax

such as `y <- 3*x + 2 + rnorm(n)`. The datasim helpers documented here are for use on the right-hand side of the specification of a node. They simplify potentially complex operations such as blocking, creation of random categorical methods, translation from categorical to numerical values, etc.

The target R-squared and variance will be achieved only if `exact=TRUE` or the sample size goes to infinity.

Value

A numerical or categorical vector which will be assembled into a data frame by `datasim_run()`

Examples

```
Demo <- datasim_make(
  g <- categorical(n, a=2, b=1, c=0.5),
  x <- cat2value(g, a=-1.7, b=0.1, c=1.2),
  y <- bernoulli(logodds = x, labels=c("no", "yes")),
  z <- random_levels(n, k=4),
  w <- mix_with(x, noise=rnorm(n), R2=0.75, var=1),
  treatment <- block_by(w),
  dice <- each(rnorm(1, sd = abs(w)))
)
```

College_grades *Grades at a small college*

Description

These are the actual grades for 400+ individual students in the courses they took at a small, liberal-arts college in the midwest US. All the students graduated in 2006. Each row corresponds to a single student in a single course. The data have been de-identified by translating the student ID, the instructor ID, and the name of the department. Typically a graduating student has taken about 32 courses. As another form of de-identification, only half of the courses each student, selected randomly, are included. Only courses with 10 or more students enrolled were included.

Usage

```
data(College_grades)
```

Format

A data frame with 6146 Grades for 443 students.

```
-grade The letter grade for the student in this course: A is the highest.
- sessionID An identifier for the course taken. Courses
offered multiple times in one semester or across semesters have individual IDs.
-sid The student ID
```

-dept The department in which the course was offered. 100 is entry-level, 200 sophomore-level, 300 junior-level, 400 senior-level.
-enroll Student enrollment in the course. This includes students who are not part of this sample.
-iid Instructor ID
-gradepoint A translation of the letter grade into a numerical scale. 4 is high. Some letter grades are not counted in a student's gradepoint average. These have \code{NA} for the gradepoint.

Source

The data were helpfully provided by the registrar of the college with the proviso that the de-identification steps outlined above be performed.

---

conf_interval *Summaries of regression models*

Description

The summaries are always in the form of a data frame

- `conf_interval()` — displays coefficients and their confidence intervals
- `R2()` — R-squared of a model together with related measures such as F, adjusted R-squared, the p-value, and degrees of freedom used in calculating the p-value.
- `regression_summary()` – A regression report in data-frame format.
- `anova_summary()` — An ANOVA report in data-frame format. If only one model is passed as an argument, the data frame will have one line per model term. If multiple models are given as arguments, the ANOVA report will show the increments from one model to the next.

Usage

```
conf_interval(model, level = 0.95, show_p = FALSE)

R2(model)

regression_summary(model)

anova_summary(...)
```

Arguments

`model`	A model as produced by `model_train()`, `lm()`, `glm()`, and so on
`level`	Confidence level to use in `conf_interval()` (default: 0.95)
`show_p`	For `conf_interval()`, append the p-value to the report.
`...`	One or more models (for ANOVA)

Details

Many of these are wrappers around `broom::tidy()` used to emphasize to students that the results are a summary in the form of a regression report, similar to the summaries produced by `stats::confint()`, `stats::coef()`, etc.

Value

a data frame

Examples

```
Model <- CRDS |> model_train(FEV ~ age + smoker)
Model |> conf_interval()
Model |> R2()
Model |> anova_summary()
```

CRDS *Smoking and lung function among youths*

Description

Data from the Childhood Respiratory Disease Study collected in the late 1970s to examine the effects of smoking and exposure to second-hand smoke. on pulmonary functions in youths.

Usage

```
data(CRDS)
```

Format

A data.frame with one row for each of 645 youngsters.

- age in years
- FEV (forced expiratory lung volume) in liters
- height in inches
- sex
- smoker whether or not the youngster smokes

Source

Cummiskey, et al. (2020) "Causal Inference in Introductory Statistics Courses" *Journal of Statistics Education* **28**(1) doi:10.1080/10691898.2020.1713936

dag_draw *Draw a DAG*

Description

Make a simple drawing of a Directed Acyclic Graph as constructed by `datasim_make`.

Usage

```
dag_draw(DAG, ..., report_hidden = FALSE)
```

Arguments

`DAG`	The DAG to draw
`report_hidden`	logical. If `TRUE`, show the hidden nodes.
`...`	Additional arguments to plot.igraph()

Details

See the igraph package for more details.

By default, edges are not drawn to hidden nodes, that is, those whose names begin with a dot. To show the hidden nodes, use the argument `show_hidden=TRUE`.

Value

No return value. Called for graphics side-effects.

Examples

```
dag_draw(sim_03)
```

datasim_make *Construct and modify data simulations*

Description

Construct and modify data simulations

Usage

```
datasim_make(...)

datasim_to_igraph(sim, report_hidden = FALSE)

datasim_intervene(sim, ...)
```

Arguments

sim	The data simulation object to be modified.
report_hidden	If TRUE, show the hidden nodes (nodes whose names begin with a dot.)
...	Descriptions of the nodes in the simulation, written in assignment form. See details.

Details

Simulations in LSTbook are first specified by providing the code for each node (which can be written in terms of the values of other nodes). Once constructed, data can be extracted from the simulation using `datasim_run(n)` or the generic `take_sample(n)`.

Each argument defines one node in the simulation. The argument syntax is unusual, using *assignment*. For instance, an argument `y <- 3*x + rnorm(n)` defines a node named `y`. The R code on the RHS of the assignment operator (that is, `3*x + rnorm(n)` in the example) will be used by `datasim_run()` to generate the `y` variable when the simulation is run. Nodes defined by previous arguments can be used in the code for later arguments.

Helper functions such as `mix_with()`, `categorical()`, and several others can be used within the node to perform complex operations.

Remember to use *commas* to separate the arguments in the normal way.

Value

an object of class "datasim". Internally, this is a list of the R assignment expressions used when running the simulation.

Examples

```
Simple_sim <- datasim_make(x <- rnorm(n, sd=2), y <- 3*x + rnorm(n))
Simple_sim |> datasim_run(n = 5)
```

---

datasim_run *Run a datasim simulation, producing a data frame*

Description

Run a datasim simulation, producing a data frame

Usage

```
datasim_run(sim, n = 5, seed = NULL, report_hidden = FALSE)
```

Arguments

sim	A simulation object, as produced by datasim_make().
n	The size of the data sample pulled from the simulation.
seed	An integer random number seed, for reproducibility. (Or, use set.seed() before running sim_run().)
report_hidden	logical. If TRUE, show the values of hidden variables (that is, variables whose name begins with a dot)

Value

a data frame with a column for each node in the datasim object.

Econ_outlook_poll *SIMULATED data from an economic outlook poll*

Description

SIMULATED data from an economic outlook poll

Usage

```
Econ_outlook_poll
```

Format

10,000 rows with three variables: age, income, pessimism

Data from a simple SIMULATION of people's pessimism about the economic state based on age group and income group. Nothing about the real world should be inferred from these data; they are merely to illustrate adjusting for covariates.

Source

The simulation is from "Statistical Modeling: A Fresh Approach" (2/e). Code for it is in the file system.file("SM2-simulations.R", package="LSTbook")

explanatory_vars *Utilities*

Description

Functions for pulling various components from model objects. These work mainly for lm and glm objects. It's a future project to add facilities for other object types.

Usage

```
explanatory_vars(model, ...)

response_var(model, ...)

response_values(model, ...)

formula_from_mod(model, ...)

get_training_data(model, ...)
```

Arguments

model	the model in question
...	(not used)

FARS *Annual summaries concerning motor-vehicle related fatalities in the US#'*

Description

Annual summaries concerning motor-vehicle related fatalities in the US#'

Usage

```
data(FARS)
```

Format

A data.frame object with one row per year from 1994 to 2016

- year: The year covered by the summary
- crashes the number of incidents in that year
- drivers the number of drivers killed in those incidents
- passengers the number of passengers killed in those incidents

- unknown vehicle occupants killed whose status as driver or passenger is unknown
- motorcyclists the number of motorcyclists killed in those incidents
- pedestrians the number of pedestrians killed in those incidentss
- pedalcyclists the number of non-motorized cyclist deaths
- other_nonvehicle the number of other deaths in those incidents
- vehicle_miles the number of miles driven by all vehicles, whether they were involved in an incident or not. (billions of miles)
- population the population of the US (thousands of people)
- registered_vehicles the number of motor vehicles registered in the US (thousands)
- licensed_drivers the number of licenced drivers in the US (thousands)

Source

From the Fatality Analysis Reporting System of the US Department of Transportation (DOT). The data have been put into a tidy form from the untidy version published by the DOT, removing columns calculated from other columns and so on.

Framingham *Data from the Framingham heart study*

Description

When it launched in 1948 the original goal of The Framingham Heart Study (FHS) launched in 1948 with the goal of identifying risk factors for cardiovascular disease. FHS had over 14,000 people from three generations, including the original participants, their children, and their grandchildren. These data represent 4238 Framingham subjects and were published by Kaggle for a machine-learning competition. The goal of the competition was to predict TenYearCHD from the other factors.

Usage

```
data(Framingham)
```

Format

4238 rows, each of which is a FHS subject. There are 16 variables:

- sex
- age of the patient
- education highest level achieved: some HS, HS grad/GED, some college/vocational school, college graduate
- currentSmoker: whether or not the patient is a current smoker
- cigsPerDay: the number of cigarettes that the person smoked on average in one day.
- BPMeds: whether or not the patient was on blood pressure medication

- `prevalentStroke`: whether or not the patient had previously had a stroke
- `prevalentHyp`: whether or not the patient was hypertensive
- `diabetes`: whether or not the patient had diabetes
- `totChol`: total cholesterol level
- `sysBP`: systolic blood pressure
- `diaBP`: diastolic blood pressure
- `BMI`: Body Mass Index
- `heartRate`: heart rate
- `glucose`: glucose level
- `TenYearCHD`: Did the patient develop congestive heart disease during a 10 year follow-up? (1=Yes)

Source

Kaggle and Github repository

References

Description of FHS by the National Heart, Lung, and Blood Institute

---

`Germany1933vote` *Voting patterns in the 1933 German national election*

Description

1933 was the year that Hitler and the Nazi party came to power. The initial basis for this was a national election in which the Nazis secured a substantial fraction of the vote. (Immediately after the election, the Nazis burned the Reichtag (the German parliament) and started repressing their political opposition though a campaign of imprisonment and murder.)

Usage

```
data("Germany1933vote")
```

Format

A data frame with 681 rows and 7 variables. Each row is a German precinct.

- self: share of potential voters who are self-employed
- blue: share of potential voters who are blue-collar workers
- white: share of potential voters who are white-collar workers
- domestic: share of potential voters who are employed domestically
- unemployed: share of potential voters who are un-employed
- nvoter: number of eligible voters (not clear if this include people who didn't vote)
- nazivote: number of votes for the Nazis

References

- Imai, Kosuke. 2017. Quantitative Social Science: An Introduction. Princeton University Press. URL from whence these data were added to this package. In QSS, the data are called nazis.
- G. King, O. Rosen, M. Tanner, A.F. Wagner (2008) "Ordinary economic voting behavior in the extraordinary election of Adolf Hitler." Journal of Economic History, vol. 68, pp. 951–996.#'

`Gilbert` *Data from the trial of serial killer Kristen Gilbert*

Description

Intensive care unit nurse Kristen Gilbert worked for several years in the 1990s at a Veterans Administration Hospital. Her co-workers became suspicious. The co-workers observed that unexpected patient deaths occurred more frequently on her shifts than on other shifts. They also noticed a shortage of supplies of the cardiac stimulant epinephrine, which can be fatal when administered in large enough doses through an IV drip. The hospital investigators went through all the shifts during the years Gilbert worked at the hospital, noting whether Gilbert was on duty during that shift and whether there was a death during the shift.

Usage

```
data(Gilbert)
```

Format

A data frame with one row for each shift at the VA hospital.

- `death` Whether a patient death occurred during the shift.
- `gilbert` Whether nurse Kristen Gilbert was on duty during the shift.
- `time`: the winning time in seconds
- `race` the name of the race. Many races are repeated over successive years.
- `year` the year the race was run
- `name` the name of the winning runner
- `sex`: the runner's sex, coded as `F` and `M`
- `distance`: the total distance of the race in km
- `climb`: the total vertical climb of the race in meters

Details

Only tabular summaries of the shift/death information are public. This data frame was reconstructed from those summaries.

Go_vote *Get out the vote experiment*

Description

An experiment about ways to encourage voting in primary elections. During the 2006 primary election in Michigan, registered voters were randomly assigned to different treatments, each in the form of a postcard mailed to them before the primary. The most high-pressure message("Neighbors") listed the voters neighbors and whether they voted in the previous primary elections. The card promised to send out the same information after the 2006 primary, so that "you and your neighbors will all know who voted and who did not." (From the Gerber et al. reference, below.) In another treatment, "Civic Duty," the postcard said, "On August 8, remember your rights and responsibilities as a citizen. Remember to vote. DO YOUR CIVIC DUTY—VOTE!" Yet another treatment, "Hawthorne" simply told the voter that "YOU ARE BEING STUDIED!" as part of research on why people do or do not vote. There was also a control group that did not receive a postcard.

Usage

```
data(Go_vote)
```

Format

A data frame with 305866 rows and 6 variables:

- sex of the voter (female or male)
- yearofbirth: year of birth of the voter
- primary2004: whether the voter voted in the 2004 primary election (voted, abstained)
- messages: Get-out-the-vote message the voter received (Civic Duty, Control, Neighbors, Hawthorne)
- primary2006: whether the voter turned out in the 2006 primary election (voted, abstained)
- hhsize: household size of the voter

References

- Imai, Kosuke. 2017. Quantitative Social Science: An Introduction. Princeton University Press. URL.
- Alan S. Gerber, Donald P. Green, and Christopher W. Larimer (2008) “Social pressure and voter turnout: Evidence from a large-scale field experiment.” American Political Science Review, vol. 102, no. 1, pp. 33–48. doi: 10.1017/S000305540808009X

Hill_racing *Winning times in Scottish Hill races, 2005-2017*

Description

Winning times in Scottish Hill races, 2005-2017

Usage

```
data(Hill_racing)
```

Format

A data.frame object with one row for each race winner. Most races have both a male and female winner.

"year" "sex" "name" "time" "race" "distance" "climb"

- `time`: the winning time in seconds
- `race` the name of the race. Many races are repeated over successive years.
- `year` the year the race was run
- `name` the name of the winning runner
- `sex`: the runner's sex, coded as `F` and `M`
- `distance`: the total distance of the race in km
- `climb`: the total vertical climb of the race in meters

Source

The data were scraped from the Scottish Hill Racing site.

McClave_Sincich *Data from McClave-Sincich* Statistics *11/e*

Description

These are relatively small data frames used for exercises

Usage

```
Clock_auction

Geography_journals

PGA_index

Dowsing
```

Format

`Clock_auction`: Prices for grandfather clocks sold in auction

- Sales price for the clock
- Age of the clock
- Number of bidders for the clock

`Geography_journals`: Prices for geography journals, c. 2005

- `journal` name of journal
- `cost` for a one-year subscription
- `jif` journal impact factor
- `cites` number of citations of the journal in the past five years
- `rpi` relative price index

`PGA_index`: Driving distance, accuracy, and a derived index from the PGA tour

- `player` name of the player
- `dist` driving distance in yards
- `accuracy` percent of drives that land in the fairway
- `index` an index score for ranking players.

`Dowsing`: Locations identified by dowsers in an experiment

- `trial` just the row number
- `subject` identifying number assigned to the subject
- `pipe` location of the flowing-water pipe along a 10-meter line (decimeters)
- `guess` the dowser's guess of the location of the pipe in that trial (decimeters)

Source

StatCrunch

McCredie_Kurtz *"Big Five" personality ratings for college first-year students*

Description

Abstract from the research paper: Five-factor personality ratings were provided by undergraduate freshmen, their parents, and their college peers as predictors of cumulative GPA upon graduation. Conscientiousness ratings were significant predictors of GPA by all three raters; peer ratings of Conscientiousness were the only significant predictor of GPA when self-, parent-, and peer-ratings of Conscientiousness were examined simultaneously. College major was a moderator of this relationship, with self- and parent-ratings of Conscientiousness correlating more strongly with GPA among Social Science majors and parent-ratings of Conscientiousness correlating less strongly with GPA among Science majors. These findings replicate existing research regarding the validity of informant ratings as predictors of behavioral outcomes such as academic performance, while emphasizing the importance of including multiple informants from various life contexts.

Usage

```
data(McCredie_Kurtz)
```

Format

For simplicity, only the mother's and father's ratings for the student are given. The variable name indicates whose rating and on what scale, e.g. `m_extra` is the mothers rating on the extraversion scale. Other variables are:

- subjid: Unique ID for the student
- age: The student's age when the ratings were collected
- GPA: The student's eventual 4-year grade-point average
- sex: The student's sex
- field: What field the student ended up studying

Details

The five personality factors are:

1. extraversion: sociability
2. neuroticism: sadness or emotional instability
3. openness to experience
4. agreeableness: kindness
5. conscientiousness: thoughtfulness

Source

McCredie_Kurtz_Open_Data.sav comes from https://data.mendeley.com/datasets/rn2bpp6f37/1

References

Morgan N. McCredie and John E. Kurtz (2020) "Prospective prediction of academic performance in college using self- and informant-rated personality traits" *Journal of Research in Personality* **85**

`model_eval` *Evaluate a model on inputs*

Description

Evaluate a model on inputs

Usage

```
model_eval(
  mod,
  data = NULL,
  ...,
  skeleton = FALSE,
  ncont = 3,
  interval = c("prediction", "confidence", "none"),
  level = 0.95,
  type = c("response", "link")
)
```

Arguments

mod	A model as from `model_train()`, `lm()` or `glm()`
data	A data frame of inputs. If missing, the inputs will be assembled from ... or from the training data, or an skeleton will be constructed.
skeleton	Logical flag. If `TRUE`, a skeleton on inputs will be created. See `model_skeleton()`.
ncont	Only relevant to skeleton. The number of levels at which to evaluate continuous variables. See `model_skeleton()`.
interval	One of "prediction" (default), "confidence", or "none".
level	The level at which to construct the interval (default: 0.95)
type	Either "response" (default) or "link". Relevant only to glm models. The format of the `.output`
...	Optional vectors specifying the inputs. See examples.

Value

A data frame. There is one row for each row of the input values (see `data` parameter). The columns include

- the explanatory variables
- `.output` — the output of the model that corresponds to the explanatory value
- the `.lwr` and `.upr` bounds of the prediction or confidence interval
- if training data is used as the input, then it's possible to calculate the residual. This will be called `.resid`.

Examples

```
mod <- mtcars |> model_train(mpg ~ hp + wt)
model_eval(mod, hp=100, wt=c(2,3))
model_eval(mod) # training data
model_eval(mod, skeleton=TRUE)
```

model_eval_fun *Helper functions to evaluate models*

Description

Only used internally in {LSTbook}. These were originally arranged as S3 methods, but now the dispatch is done "by hand" in order to eliminate any exported S3 methods.

Usage

```
model_eval_fun(model, data = NULL, interval = "none", level = 0.95, ...)
```

Arguments

model	A model object of the classes permitted
data	Usually, a data table specifying the inputs to the model. But if not specified, the training data will be used.
interval	One of "none", "confidence", or "prediction". Not all model types support "prediction" or even "confidence".
level	(default 0.95) confidence or prediction level. Must be in [0,1]
...	additional arguments

Value

a data frame

model_family *Check model type against model specification and data*

Description

This can be used to automatically determine a model type or to determine if the specified model type is consistent with the specification/data

Usage

```
model_family(
  .data,
  .tilde,
 family = c("auto", "lm", "linear", "binomial", "poisson", "svm", "gaussian", "rlm")
)
```

Arguments

.data	A data frame or equivalent
.tilde	A model specification as a tilde expression
family	Requested model type, if any.

---

model_plot *Graph a model function*

Description

Every model has an implicit function whose output is the response variable and which has one or more explanatory variables. (Exceptionally, there might be no explanatory variables as in `response ~ 1`.) One of the explanatory variables can be mapped to the horizontal axis; this can be either quantitative or categorical. The remaining explanatory variables will be mapped to color, facet-horizontal, and facet-vertical. For visual clarity, any quantitative variables among these remaining variables must be coerced to categorical, corresponding to a discrete set of colors and a discrete set of facets.

Usage

```
model_plot(
  mod,
  nlevels = 3,
  interval = c("confidence", "prediction", "none"),
  level = 0.95,
  palette = LETTERS[1:8],
  model_ink = 0.7
)
```

Arguments

mod	A model object, made with `model_train()`, `lm()`, or `glm()`
nlevels	Integer. When quantitative variables need to be converted to factors for color or facetting, how many levels in those factors.
interval	The type of interval to draw (default: confidence)
level	The confidence or prediction level for the interval
palette	One of "A" through "F" giving some control for people who don't like or can't see the default palette
model_ink	The density of ink used to draw the model. ("alpha" for those in the know.)

Value

A ggplot graphics object

model_skeleton *Convert a model to a skeleton*

Description

A "skeleton" is a data frame containing "nicely spaced" values for the explanatory variables in a model.

Usage

```
model_skeleton(mod, data = NULL, ncont = 3, nfirstcont = 50)
```

Arguments

mod	A fitted model **or** a tilde expression describing a model structure, e.g. `outcome ~ vara+varb`.
data	a data frame. Relevant only when `mod` is a tilde expression
ncont	minimum number of levels at which to represent continuous variables. (More levels may be added to "prettify" the choices. See `pretty()`.)
nfirstcont	Like `ncont`, but for the first explanatory variable if it is categorical. This variable is mapped to the horizontal axis and so should have many levels to produce a smooth graph. (Default: 50)

Value

a data frame

Examples

```
Model <- CRDS |> model_train(FEV ~ sex + age + height)
Model |> model_skeleton()
```

model_train *train a model, easily*

Description

An interface to several of the most often used model-fitting routines designed to make it easy to construct.

Usage

```
model_train(
  data,
  tilde,
  family = c("auto", "lm", "linear", "binomial", "poisson", "rlm")
)
```

Arguments

data	Data frame to use as training data
tilde	Formula for the model
family	Character string: the family of model to fit, e.g. "lm", "binomial", "poisson", "rlm", ...

Details

Since data may be piped into this function, the training data frame will be called simply "data", the name of the first argument to this function. In order to be able to access the training data in such cases, the training data is assigned to an attribute of the resulting model, "training_data".

Value

An object of class "model_object". This is much the same as an "lm" or "glm" object but with the additional attribute of the training data and a printing method that encourages the use of the regression summary methods `conf_interval()`, `R2()`, or `anova_summary()`

---

model_values *Construct a model and return the model values*

Description

One-stop shopping to fit a model and return the model output on the training data.

Usage

```
model_values(data, tilde, family = c("linear", "prob", "counts"))
```

Arguments

data	A data frame containing the training data. When used with `mutate()`, `data` will hold the model specification, instead of `tilde`.
tilde	A model specification in the form of a tilde expression
family	The type of model architecture: `"linear"`, `"prob"`, or `"counts"`

Details

This is intended to be used ONLY WITHIN `mutate()`

Value

A **vector** (not a data frame) of the model evaluated on the training data. This is intended mainly for use within `mutate()`, so that a general model can be used in the place of simple reduction verbs like `mean()`, `median()`

Examples

```
mtcars |> mutate(mpg_mod = model_values(mpg ~ hp + wt)) |> select(hp, wt, mpg_mod) |> head()
```

Monocacy_river *Data on run-off from the Monocacy river at Jug Bridge, Maryland.*

Description

When it rains, some of the water is absorbed into the ground or quickly evaporates. Some of the water runs off (the "runoff") and is collected by streams and rivers. These data, from a 1964 reference on water resource management, were measured after 25 storms in the basin of the Monocacy River at Jug Bridge, Maryland, US.

Usage

```
Monocacy_river
```

Format

A data.frame with 25 observations on the following 2 variables'

- `precipitation`: amount of rain in inches
- `runoff` runoff in inches

Details

YOU WERER HERE, COPYING from the Rd file in man/

Source

"Probability Concepts in Engineering" A H-S Ang and W H Tang, 2007, John Wiley based on R.K. Linsley and J.B. Franzini (1964) *Water Resources Engineering* McGraw-Hill, p.68

---

mosaic_cull_for_do *Cull objects used with do()*

Description

The `do()` function facilitates easy replication for randomization tests and bootstrapping (among other things). Part of what makes this particularly useful is the ability to cull from the objects produced those elements that are useful for subsequent analysis. `cull_for_do` does this culling. It is generic, and users can add new methods to either change behavior or to handle additional classes of objects.

Usage

```
mosaic_cull_for_do(object, ...)
```

Arguments

object	an object to be culled
...	additional arguments (currently ignored)

Details

When `do(n) * expression` is evaluated, `expression` is evaluated `n` times to produce a list of `n` result objects. `cull_for_do` is then applied to each element of this list to extract from it the information that should be stored. For example, when applied to a object of class `"lm"`, the default `cull_for_do` extracts the coefficients, coefficient of determinism, an the estimate for the variance, etc.

Examples

```
Clock_auction |> model_train(price ~ resample(bidders)) |>
  R2() |> trials(times=10)
```

MPG *Fuel economy measurements on US car models*

Description

Fuel economy measurements on US car models

Usage

```
data(MPG)
```

Format

A data.frame object with one row year for each model or configuration of automobile or light truck sold in the US.

- manufacturer: name of company making the vehicle
- division: name of the company division making the vehicle
- model: vehicle model name
- fuel_year: fuel consumed in 10,000 miles (roughly 1 year.)
- CO2_year: Carbon dioxide produced per year, in kilograms. 10,000 miles of driving is taken to represent a year. Note, Carbon-per-year (without the oxygen) is roughly one-quarter the mass of CO2-per-year.
- hybrid: whether the car is a hybrid
- class: the type of vehicle, e.g. midsize, compact, large, SUV

- vol_passenger: volume for passengers (cubic feet)
- vol_luggage: volume for luggage (cubic feet)
- doors: number of passenger doors
- mpg_city: Estimated fuel consumption in city driving (miles per gallon)
- mpg_hwy: like `mpg_city` but for highway driving
- mpg_comb: like `mpg_city` but for a standard combination of city and highway driving
- EPA_fuel_cost: Annual fuel cost using a standard price for gas and a standard miles per year of driving.
- valves_exhaust: how many exhaust valves per cylinder
- valves_intake: how many air intake valves per cylinder
- CO2city: estimate of carbon-dioxide (grams/mile) production per mile in city driving.
- CO2hwy: like `CO2city` but for highway driving
- CO2combined: like `CO2city` but for a standard mixture of city and highway driving
- hatchback: is there a hatchback rear door
- start_stop: does the vehicle have a system to stop the engine when idling
- cyl_deact: are cylinders in the engine deactivated when power demand warrants
- fuel: the kind of fuel used.
  - G = regular unleaded gasoline,
  - GM = mid-grade recommended,
  - GP = premium unleaded recommended,
  - GPR = premium unleaded required,
  - DU = diesel (ultra low sulfur)
- drive: type of drive, e.g. front-wheel, 4-wheel, ...
- regen: wheels with regenerative breaking (for hybrids)
- n_gears: number of transmission gears
- n_cyl: number of engine cylinders
- displacement: engine displacement (liters)
- transmission: transmission type
  - A = automatic,
  - M = manual,
  - AM = automated manual,
  - AM = automated manual (paddles),
  - CVT = continuously variable,
  - SCV = continuously variable with selection paddles,
  - SA = semi-automatic
- lockup_torque_converter:
- air_aspiration:
- model_year:

Source

Data from the US Environmental Protection Agency (EPA) available at https://www.fueleconomy.gov/feg/download.shtml. The file for 2019 model-year vehicles is https://www.fueleconomy.gov/feg/epadata/19data.zip

---

`Nats` *Short, simple data frames for textbook examples*

Description

These small data frames are for illustration purposes only.

- `Nats` made up demographic and economic data, by country and year
- `Big` a simplified form of the PalmerPenguin data
- `Tiny` an 8-row subset of `Big`

Usage

```
Nats
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 8 rows and 4 columns.

`ntiles` *Create vector based on roughly equally sized groups*

Description

Create vector based on roughly equally sized groups

Usage

```
ntiles(
  x,
  n = 3,
  format = c("rank", "interval", "mean", "median", "center", "left", "right"),
  digits = 3
)
```

Arguments

`x`	a numeric vector
`n`	(approximate) number of quantiles
`format`	a specification of desired output format. One of "center", "interval", "left", "right", "mean", or "median.
`digits`	desired number of digits for labeling of factors.

Details

This is a functional clone of `mosaic::ntiles` in order to avoid the dependency. It should be removed in the future, when there is no need to avoid such dependency, e.g. when `{mosaic}` is available on WASM.

Value

a vector. The type of vector will depend on `format`.

Examples

```
CRDS |> head(20) |> mutate(group = ntiles(height, 3, format="center"))
CRDS |> head(20) |> mutate(group = ntiles(height, 3, format="interval"))
```

Offspring *Relative sizes offspring/parent for many species*

Description

Body mass of adult of numerous vertebrate species and newly hatched or born offspring.

Usage

```
data("Offspring")
```

Format

A data.frame with 3971 rows, each for a different species

- species
- class (phylogenetic class)
- group (phylogenetic group)
- adult Mass of the adult (female) in grams.
- hatchling Mass of the offspring, in grams. This applies as well to species where the offspring is born rather than hatched.

Source

Shai Meiri, "Endothermy, offspring size and evolution of parental provisioning in vertebrates", *Biological Journal of the Linnean Society*, **128**:4, pp. 1052-6 (See Appendix S1.)

Orings *Space Shuttle O-Ring Failures*

Description

On January 27, 1986, the night before the space shuttle *Challenger* exploded, an engineer recommended to the National Aeronautics and Space Administration (NASA) that the shuttle not be launched in the cold weather. The forecasted temperature for the *Challenger* launch was 31 degrees Fahrenheit—the coldest launch ever. After an intense 3-hour telephone conference, officials decided to proceed with the launch. This data frame contains the launch temperatures and the number of O-ring problems in 24 shuttle launches prior to the *Challenger*. (This documentation comes from the `Sleuth3` package, where the dataset is called `ex2223`.)

Usage

```
data(Orings)
```

Format

A data frame with 24 observations on the following 2 variables.

- temp Launch temperatures (in degrees Fahrenheit)
- incidents Numbers of O-ring incidents

Source

Ramsey, F.L. and Schafer, D.W. (2013). *The Statistical Sleuth: A Course in Methods of Data Analysis (3rd ed)*, Cengage Learning.

Penguins *Body measurements on penguins*

Description

This is the `palmerpenguins::penguins` data frame with slightly simplified variable names.

Usage

```
Penguins
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 333 rows and 8 columns.

PIDD *Pima Indians Diabetes Database*

Description

"The population for this study was the Pima Indian population near Phoenix, Arizona. That population has been under continuous study since 1965 by the National Institute of Diabetes and Digestive and Kidney Diseases because of its high incidence rate of diabetes. Each community resident over 5 years of age was asked to undergo a standardized examination every two years, which included an oral glucose tolerance test. Diabetes was diagnosed according to World Health Organization Criteria; that is, if the 2 hour post-load plasma glucose was at least 200 mg/dl (11.1 mmol/l) at any survey examination or if the Indian Health Service Hospital serving the community found a glucose concentration of at least 200 mg/dl during the course of routine medical care." — quoted from the reference below. The data were published by Kaggle for a machine-learning competition whose goal was to develop a prediction function for diabetes.

Usage

```
data(PIDD)
```

Format

768 rows, each of which is a woman 21 years or older. There are 9 variables:

- `age` of the woman
- `pregnancies`: number of previous pregnancies
- `glucose`: glucose level
- `BP`: systolic blood pressure
- `skin_thickness`:
- `insulin`:
- `bmi`: Body mass index
- `pedigree`: "Diabetes Pedigree Function"
- `diabetes`: Did the patient develop diabetes during a 5-year follow-up?

Source

Kaggle

References

Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988) "Using the ADAP learning algorithm to forecast the onset of diabetes mellitus" *Proceedings of the Symposium on Computer Applications and Medical Care*

`point_plot` *One-step data graphics*

Description

`point_plot()` makes it easy to construct an informative basic graph of a data frame. "Making it easy" means that the user only needs to specify two things: 1) the data frame to be used and 2) a tilde expression with the response variable on the left and up to three explanatory variables on the right. The response variable is mapped to the vertical axis while the first explanatory variable defines the horizontal axis. The second explanatory variable (if any) maps to color, the third (if any) defines facets. Quantitative variables used for color or faceting are cut into categorical variables, so color and facets will always be discrete.

Usage

```
point_plot(
  D,
  tilde,
  ...,
  seed = 101,
  annot = c("none", "violin", "model", "bw"),
  jitter = c("default", "none", "all", "x", "y"),
  interval = c("confidence", "none", "prediction"),
  point_ink = 0.5,
  model_ink = 0.4,
  palette = LETTERS[1:8],
  bw = NULL,
  level = 0.95,
  nx = 50,
  model_family = NULL
)
```

Arguments

`D`	a data frame
`tilde`	tilde expression specifying `y ~ x` or `y ~ x + color`
`seed`	(optional) random seed for jittering
`annot`	Statistical annotation (one of "none", "violin", "model", "bw")
`jitter`	Options for turning on jitter: one of `"default"`, `"both"`, `"none"`, `"x"`, `"y"`. By default, By default, categorical variables are jittered.
`interval`	the type of interval: default `"confidence"`. Others: `"none"` or `"prediction"`
`point_ink`	Opacity of ink for the data points
`model_ink`	Opacity of ink for the model annotation
`palette`	Depending on taste and visual capabilities, some people might prefer to alter the color scheme. There are 8 palettes available: `"A"` through `"H"`.

bw	bandwidth for violin plot
level	confidence level to use (0.95)
nx	Number of places to evaluate any x-axis quantitative vars. Default 50. Use higher if graph isn't smooth enough.
model_family	Override the default model type. See model_train()
...	Graphical options for the data points, labels, e.g. size

Details

When an x- or y- variables is categorical, jittering is automatically applied.

Using annot = "model" will annotate the data with the graph of a model — shown as confidence intervals/bands — corresponding to the tilde expression. annot = "violin" will annotate with a violin plot.

If you want to use the same explanatory variable for color and faceting (this might have pedagogical purposes) merely repeat the name of the color variable in the faceting position, e.g. mpg ~ hp + cyl + cyl.

Value

A ggplot graphics object

See Also

add_plot_labels to add labels to the plot (without needing the ggplot2 + pipe)

Examples

```
mosaicData::Galton |> point_plot(height ~ mother + sex + father, annot="model", model_ink=1)
mtcars |> point_plot(mpg ~ wt + cyl)
mtcars |> point_plot(mpg ~ wt + cyl + hp, annot="model")
```

---

print.datasim *Nice printing of some internal objects*

Description

Nice printing of some internal objects

Usage

```
## S3 method for class 'datasim'
print(x, ..., report_hidden = FALSE)
```

Arguments

x	A data simulation as made by datasim_make()
report_hidden	Show the hidden nodes (nodes whose name begins with .)
...	for compatibility with generic print()

print.model_object *A printing method for model objects*

Description

A printing method for model objects

Usage

```
## S3 method for class 'model_object'
print(x, ...)
```

Arguments

x	The object to print
...	Not used, but here for consistency with generic print()

random_terms *Create columns with random numbers for modeling*

Description

For demonstration purposes, add the specified number of random columns to a model matrix. This is intended to be used in modeling functions, e.g. `model_train()`, `lm()`, and so on to explore the extent to which random columns "explain" the response variable.

Usage

```
random_terms(df = 1, rdist = rnorm, args = list(), n, seed = NULL)
```

Arguments

df	How many columns to add
rdist	Function to generate each column's numbers (default: `rnorm`)
args	A list holding the parameters (if any) to be used for the `rdist` argument
n	OPTIONALLY, dictate the number of rows in the output
seed	Integer seed for the random-number generator

Details

`random_terms()` will try to guess a suitable value for `n` based on the calling function.

Examples

```
mtcars |> model_train(mpg ~ wt + random_terms(4)) |> conf_interval()
mtcars |> model_train(mpg ~ wt + random_terms(4)) |> anova_summary()
head(mtcars) |> select(wt, mpg) |> mutate(r = random_terms(3))
```

`Registrar` *Sample from a college registrar's database*

Description

Grade data from students at a liberal arts college. IDs of students, professors, and departments have been dis-identified.

Usage

```
Sessions

Grades

Gradepoint
```

Format

Three data frames

- Sessions: ID for a class session, that is, a course in a semester
  - sessionID: Unique identifier for the session
  - iid: Unique identifier for the instructor
  - enroll: Total enrollment in the session (note: includes students who didn't make it into the sample in `Grades`)
  - dept: Unique identifier for the department
  - level: Instruction evel of the course 100, 200, 300, 400. Roughly: first-year, sophomore, junior, senior
  - sem: The semester in which the session was held.
- Grades: A 50% random sample of student-by-student grades in those Sessions
  - sid: Unique identifier or the student.
  - grade: Letter grade: A, A-, B+ and so on,
  - sessionID: The course session for the grade, as in the Sessions data frame
- Gradepoint: Letter to numerical conversion (per college policy)
  - grade: Letter grade: A, A-, and so on
  - gradepoint: Numerical equivalent

An object of class `data.frame` with 6124 rows and 3 columns.

An object of class `grouped_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 14 rows and 2 columns.

Source

Used with permission by the college's registrar.

Shipping_losses *Shipping losses in 1941 in the Atlantic*

Description

A major theater of action in World War II was the Atlantic ocean. Germany attempted through submarine and aerial attacks to sink shipping supplying Britain through the war. This table summarizes the losses in 1941.

Usage

```
data(Shipping_losses)
```

Format

12 rows, one for each month of 1941

- month
- nships number of ships sunk
- tons gross tonnage of the ships sunk. This includes both the ship and the cargo.
- country whether the ships were British, or belonged to Allied or Neutral countries.

Source

From W.S. Churchill (1952) *The Grand Alliance* a history of the Second World War. Houghton Mifflin Co. Boston. p. 782

sim_objects *Simulations for use in* Lessons in Statistical Thinking

Description

These `datasim` objects are provided.

Usage

```
sim_00

sim_01

sim_02

sim_03

sim_04
```

sim_05

sim_06

sim_07

sim_08

sim_09

sim_10

sim_11

sim_12

sim_flights

sim_medical_observations

sim_prob_21.1

sim_satgpa

sim_school1

sim_school2

sim_vaccine

Format

An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.
An object of class list (inherits from datasim) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

An object of class `list` (inherits from `datasim`) of length 2.

Details

They are defined in the sim_library.R file in `inst/`

`split_tilde` *Evaluate a tilde expression on a data frame*

Description

Evaluate a tilde expression on a data frame

Usage

```
split_tilde(tilde)
```

Arguments

`tilde`	A two-sided tilde expression used for model specification

---

`STAR` *STAR Project Data*

Description

Data from the STAR (Student–Teacher Achievement Ratio) Project, a four-year longitudinal study examining the effect of class size in early grade levels on educational performance and personal development

Usage

```
data("STAR")
```

Format

A data frame with 6325 rows and 6 variables:

- race: black or white
- classtype: kindergarten class type: small, regular, regular with aid
- yearssmall: number of years (0 to 4) in small classes
- hsgrad: high-school graduation (graduated or not). NOTE: There are many NAs
- g4math: total scaled score for the math portion of the fourth-grade standardized test
- g4reading: total scaled score for the reading portion of the fourth-grade standardized test

References

- Imai, Kosuke. 2017. Quantitative Social Science: An Introduction. Princeton University Press. URL from whence these data were added to this package.
- Mosteller, Frederick. 1997. "The Tennessee Study of Class Size in the Early School Grades." Bulletin of the American Academy of Arts and Sciences 50(7): 14-25. doi = 10.2307/3824562

`take_sample` *Samples from various kinds of objects*

Description

A set of methods to generate random samples from data frames and data simulations. For data frames, individual rows are sampled. For vectors, elements are sampled.

Usage

```
take_sample(x, n, replace = FALSE, ...)

## Default S3 method:
take_sample(
  x,
  n = length(x),
  replace = FALSE,
  prob = NULL,
  .by = NULL,
  groups = .by,
  orig.ids = FALSE,
  ...
)

resample(..., replace = TRUE)
```

Arguments

x	The object from which to sample
n	Size of the sample.
replace	Logical flag: whether to sample with replacement. (default: FALSE)
prob	Probabilities to use for sampling, one for each element of x
.by	Variables to use to define groups for sampling, as in {dplyr}. The sample size applies to each group.
groups	Variable indicating blocks to sample within
orig.ids	Logical. If TRUE, append a column named "orig.ids" with the row from the original x that the same came from.
...	Arguments to pass along to specific sample methods.

Details

These are based in spirit on the sample functions in the {mosaic} package, but are redefined here to 1) avoid a dependency on {mosaic} and 2) bring the arguments in line with the .by = features of {dplyr}.

Value

A vector or a data frame depending on the nature of the x argument.

Examples

```
take_sample(sim_03, n=5) # run a simulation
take_sample(Clock_auction, n = 3) # from a data frame
take_sample(1:6, n = 6) # sample from a vector
```

---

trials *Run the left side of the pipeline multiple times.*

Description

Write a pipeline to perform some calculation whose result can be coerced into one line of a data frame. Add trials(times=3) to the end of the pipeline in order to repeat the calculation multiple times. Typically, each trial involves some random component, but another (or an additional) capability is to parameterize the pipeline expression by including some unbound variable in it, e.g. lambda. Then call trials(lambda=c(10,20)) to repeat the calculation for each of the elements of the named parameter.

Usage

```
trials(.ex, times = 1, ...)
```

Arguments

.ex	(Not user-facing.) The left side of the pipeline.
times	The number of times to run the trial.
...	Values for any unbound parameter in the left side of the pipeline. If a vector of length > 1, the trials will be run separately for each element of the vector.

Details

This is intended as a pipeline friendly replacement for `mosaic::do()`.

Value

a dataframe with one row for each trial. (But see the ... argument.)

Examples

```
mean(rnorm(10)) |> trials(times=3)
mean(rnorm(lambda)) |> trials(lambda=c(1, 100, 10000))
mean(rnorm(lambda)) |> trials(times=5, lambda=c(1, 100, 10000))
take_sample(mtcars, n=lambda, replace=TRUE) |> select(mpg, hp) |>
  model_train(mpg ~ resample(hp)) |>
  regression_summary() |> trials(times=3, lambda=c(10, 20, 40)) |>
  filter(term == "resample(hp)")
```

UCB_applicants *Roster of applicants to six major departments at UC Berkeley*

Description

Roster of applicants to six major departments at UC Berkeley

Usage

```
data(UCB_applicants)
```

Format

A data.frame object with 4236 rows, one for each of the applicants to graduate school at UC Berkeley for the Fall 1973 quarter.

- `admit`: Whether the applicant was admitted.
- `gender`: male or female
- `dept`: The graduate department applied to. Rather than identifying the actual departments involved, the data released by Berkeley used letter codes.

Details

In 1973, officials at the University of California Berkeley noticed disturbing trends in graduate admissions rates. The data, with department names redacted, was presented and interpretted in a famous paper in *Science*, Bickel et al. 1975. In that paper, summary tables were presented. `UCB_applicants` was reverse engineered from `datasets::UCBAdmissions` into a data table where the unit of observation is an individual applicant. The origin of `datasets::UCBAdmissions` is not clear; those data are not explicitly provided in Bickel et al.

Source

The `UCBApplicants` summary table in the `datasets` R package.

References

Bickel, P. J., Hammel, E. A., and O'Connell, J. W. (1975). Sex bias in graduate admissions: Data from Berkeley. Science, 187, 398–403.

`US_wildfires` *Monthly tallies of wildfires in the US from 2000 to 2022*

Description

Records for each month of wildfires in the US.

Usage

```
data(AAUP)
```

Format

275 rows, each of which is a month

- `date` The year and month in a format that can be easily plotted
- `area`: total area burned by the wildfires in that month (acres)
- `number`: the number of wildfires in that month
- `month`: for convenience, the month (Jan, Feb, ..., Dec) as an ordered factor.

Source

USGS

Wheat — *Experimental data on the yield of winter wheat*

Description

In the experiment, eight different varieties of winter wheat were planted in each of 7 calendar years (1996-2002). Each genotype was assigned randomly to a `plot` within a `block`.

Usage

```
data("Wheat")
```

Format

A data.frame with 240 rows

- genotype The type of wheat.
- yield of the wheat from this plot
- block Major region of the field
- plot Subdivision of block in which the wheat was planted
- year of the planting and measurement

Source

Andrea Onofri "Repeated measures with perennial crops"

zero_one — *Zero-one transformation for categorical variable*

Description

A convenience function for handling categorical response variables. Ordinarily, ggplot2 maps categorical levels to numerical values 1, 2, ... Such numerical mapping is inappropriate for logistic modeling, where we want the levels to be on a probability scale.

Usage

```
zero_one(x, one)

label_zero_one(P)
```

Arguments

`x`	a categorical variable
`one`	character string specifying the level that gets mapped to 1.
`P`	A ggplot2 object made by `model_plot()` or `point_plot()`

Value

A numerical vector of 0s and 1s.

Examples

```
Birdkeepers |>
  point_plot(zero_one(LC, one="LungCancer") ~ AG + BK, annot = "model")

Birdkeepers |>
  mutate(Condition = zero_one(LC, one = "LungCancer")) |>
  point_plot(Condition ~ AG + BK, annot = "model") |>
  label_zero_one() |>
  add_plot_labels(x="age", color = "Birdkeeper?")
```

Index