Package 'spinifex'

July 14, 2020

Title Manual Tours, Manual Control of Dynamic Projections of Numeric Multivariate Data

Version 0.2.5

Description Generates the path for manual tours ['Cook' & 'Buja' (1997) <doi:10.2307/1390747>]. Tours are generally available in the 'tourr' package ['Wickham' 'et' 'al.' (2011) <doi:10.18637/jss.v040.i02>].
The grand tour is an algorithm that shows all possible projections given sufficient time.
Guided uses projection pursuit to steer the tour towards interesting projections.
The 'spinifex' package implements manual control, where the contribution of a selected variable can be adjusted between -1 to 1, to examine the sensitivity of structure in the data to that variable.
The result is an animation where the variable is toured into and out of the projection completely, which can be rendered using the 'gganimate' and 'plotly' packages.

Depends R (>= 3.4.0)

License CC BY-NC-SA 4.0

URL https://github.com/nspyrison/spinifex/

BugReports https://github.com/nspyrison/spinifex/issues

Imports tourr, ggplot2, gganimate, plotly, shiny

Suggests rmarkdown, RColorBrewer, htmlwidgets, knitr, testthat, covr, dplyr, GGally

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Author Nicholas Spyrison [aut, cre], Dianne Cook [aut, ths]

Maintainer Nicholas Spyrison <spyrison@gmail.com>

Repository CRAN

Date/Publication 2020-07-14 13:10:06 UTC

R topics documented:

`array2df` *Turns a tour path array into a long data frame.*

Description

Typically called by a wrapper function, `play_manual_tour` or `play_tour_path`. Takes the result of `tourr::save_history()` or `manual_tour()` and restructures the data from an array to a long data frame for use in ggplots.

Usage

```
array2df(array, data = NULL, lab = NULL)
```

Arguments

`array`	A (p, d, n_slides) array of a tour, the output of `manual_tour()`.
`data`	Optional, (n, p) dataset to project, consisting of numeric variables.
`lab`	Optional, labels for the reference frame of length 1 or the number of variables used. Defaults to an abbreviation of the variables.

Value

A list containing an array of basis slides (p, d, n_slides) and an array of data slides (n, d, n_slides) if data is present.

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])

rb <- tourr::basis_random(n = ncol(flea_std))
mtour <- manual_tour(basis = rb, manip_var = 4)
array2df(array = mtour, data = flea_std)
```

---

breastcancer *Wisconsin Breast Cancer Database*

Description

Formatted subset of `mlbench::BreastCancer`. See `mlbench` for original data more context.

Usage

```
breastcancer
```

Format

Data frame with 675 observations on 10 variables: a factor Id, 9 numeric variables, and target class.

Details

The objective is to identify each of a number of benign or malignant classes. Samples arrive periodically as Dr. Wolberg reports his clinical cases. The database therefore reflects this chronological grouping of the data. This grouping information appears immediately below, having been removed from the data itself. Each variable except for the first was converted into 11 primitive numerical attributes with values ranging from 0 through 10. There are 16 missing attribute values.

Data frame with 675 observations on 10 variables: a factor Id, 9 numeric variables, and target class:

- Id, Sample code number
- Cl.thickness, Clump thickness
- Cell.size, Uniformity of cell size
- Cell.shape, Uniformity of cell shape
- Marg.adhesion, Marginal adhesion
- Epith.c.size, Single Epthelial cell size
- Bare.nuclei, Bare nuclei
- Bl.cromatin, Bland chromatin
- Normal.nucleoli, Normal Nucleoli

- Class, Class

Reproducing this dataset:

```
requireNamespace("mlbench")
data(BreastCancer)

d <- BreastCancer
d <- d[!duplicated(d), ]
d <- d[complete.cases(d), ]
mat <- as.matrix(d[ , 2:9])
mat <- apply(mat, 2, as.numeric)
breastcancer <- data.frame(Id = d$Id, mat, Class = d$Class)
```

Examples

```
str(breastcancer)
## Not run:
play_manual_tour(data = breastcancer[, 2:9], manip_var = 3, rescale_data = TRUE)

## End(Not run)
```

`col_of` *Return hex color code for a given discrete categorical variable.*

Description

Return hex color code for a given discrete categorical variable.

Usage

```
col_of(class, pallet_name = "Dark2")
```

Arguments

`class`	The discrete categorical variable to return the color of.
`pallet_name`	The name of the `RColorBrewer` pallet to get the colors from. Defaults to "Dark2".

Value

Vector of character hex color code of the passed categorical variable.

Examples

```
col_of(tourr::flea$species)
```

create_manip_space *Create a manipulation space to rotate the manip variable in.*

Description

Typically called by `manual_tour()`. Creates a (p, d) orthonormal matrix, the manipulation space from the given basis right concatenated with a zero vector, with manip_var set to 1.

Usage

```
create_manip_space(basis, manip_var)
```

Arguments

`basis`	A (p, d) orthonormal numeric matrix, the linear combination the original variables contribute to projection frame. Required, no default.
`manip_var`	The number of the variable/column to rotate.

Value

A (p, d + 1) orthonormal matrix, the manipulation space to manipulate the projection in.

Examples

```
flea_std <- tourr::rescale(tourr::flea[,1:6])

rb <- tourr::basis_random(n = ncol(flea_std))
create_manip_space(basis = rb, manip_var = 4)
```

is_orthonormal *Test if a numeric matrix is orthonormal.*

Description

Handles more cases than tourr::is_orthonormal().

Usage

```
is_orthonormal(x, tol = 0.001)
```

Arguments

`x`	Numeric matrix to test the orthonormality of.
`tol`	Tolerance of (the sum of element-wise) floating point differences.

Value

Single logical of the orthonormal matrix of the matrix.

Examples

```
is_orthonormal(tourr::basis_random(n = 6))
is_orthonormal(matrix(1:12, ncol=2), tol = 0.01)
```

`manual_tour` *Produce the series of projection bases to rotate a variable into and out of a projection.*

Description

Typically called by `array2af()`. An array of projections, the radial tour of the `manip_var`, which is rotated from phi's starting position to `phi_max`, to `phi_min`, and back to the start position.

Usage

```
manual_tour(
  basis = NULL,
  manip_var,
  theta = NULL,
  phi_min = 0L,
  phi_max = 0.5 * pi,
  angle = 0.05,
  ...
)
```

Arguments

`basis`	A (p, d) orthonormal numeric matrix. The linear combination the original variables contribute to projection space. Defaults to NULL, generating a random basis.
`manip_var`	Integer column number or string exact column name of the. variable to manipulate. Required, no default.
`theta`	Angle in radians of "in-plane" rotation, on the xy plane of the reference frame. Defaults to theta of the basis for a radial tour.
`phi_min`	Minimum value phi should move to. Phi is angle in radians of the "out-of-plane" rotation, the z-axis of the reference frame. Required, defaults to 0.
`phi_max`	Maximum value phi should move to. Phi is angle in radians of the "out-of-plane" rotation, the z-axis of the reference frame. Required, defaults to pi/2.
`angle`	Target distance (in radians) between steps. Defaults to .05.
`...`	Handles unused arguments that are being also being passed from `play_manual_tour()` to `render_()`.

Value

A (p, d, 4) history_array of the radial tour. The bases set for phi_start, `phi_min`, `phi_max`, and back to phi_start.

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
rb <- tourr::basis_random(n = ncol(flea_std))

manual_tour(basis = rb, manip_var = 4)

manual_tour(basis = rb, manip_var = 6,
            theta = pi / 2, phi_min = pi / 16, phi_max = pi, angle = .1)
```

---

oblique_basis *Return the basis of an oblique frame*

Description

Rotates a basis returning (p, 2) basis describing `oblique_frame()` Used to create an oblique tour by small changes to the rotation.

Usage

```
oblique_basis(basis = NULL, manip_var, theta = NULL, phi = NULL)
```

Arguments

`basis`	A (p, d) orthonormal numeric matrix. The linear combination the original variables contribute to projection space. Defaults to NULL, generating a random basis.
`manip_var`	Number of the column/dimension to rotate.
`theta`	Angle in radians of "in-plane" rotation, on the xy plane of the reference frame. Required, no default. If left NULL, will initialize the radial angle of the `manip_var`.‘
`phi`	Phi is angle in radians of the "out-of-plane" rotation, the z-axis of the reference frame. Required, no default.

Value

(p, 2) matrix of the rotated basis.

Examples

```
rb    <- tourr::basis_random(n = 6)
theta <- runif(1, 0, 2*pi)
phi   <- runif(1, 0, 2*pi)

oblique_basis(basis = rb, manip_var = 4, theta, phi)
```

`oblique_frame` *Plot a single frame of a manual tour*

Description

Projects the specified rotation as a 2D ggplot object. One static frame of manual tour. Useful for providing user-guided interaction.

Usage

```
oblique_frame(
  basis = NULL,
  data = NULL,
  manip_var = NULL,
  theta = 0L,
  phi = 0L,
  lab = NULL,
  rescale_data = FALSE,
  ...
)
```

Arguments

`basis`	A (p, d) dim orthonormal numeric matrix. Defaults to NULL, giving a random basis.
`data`	A (n, p) dataset to project, consisting of numeric variables.
`manip_var`	Number of the variable to rotate.
`theta`	Angle in radians of "in-projection plane" rotation, on the xy plane of the reference frame. Defaults to 0, no rotation.
`phi`	Angle in radians of the "out-of-projection plane" rotation, into the z-direction of the axes. Defaults to 0, no rotation.
`lab`	Optionally, provide a character vector of length p (or 1) to label the variable contributions to the axes, Default NULL, results in a 3 character abbreviation of the variable names.
`rescale_data`	When TRUE scales the data to between 0 and 1. Defaults to FALSE.
`...`	Optionally pass additional arguments to the `render_type` for projection point aesthetics; `geom_point(aes(...))`.

Value

A ggplot object of the rotated projection.

Examples

```
flea_std <- tourr::rescale(tourr::flea[,1:6])
rb       <- tourr::basis_random(n = ncol(flea_std))
theta    <- runif(1, 0, 2*pi)
phi      <- runif(1, 0, 2*pi)

oblique_frame(data = flea_std, basis = rb, manip_var = 4, theta, phi)

oblique_frame(data = flea_std, basis = rb, manip_var = 4,
              theta = 0, phi = 1,
              lab = paste0("MyNm", 3:8), rescale_data = TRUE)
```

---

pan_zoom *Pan (offset) and zoom (scale) a 2 column matrix, dataframe or scaler number.*

Description

Pan (offset) and zoom (scale) a 2 column matrix, dataframe or scaler number.

Usage

```
pan_zoom(x, x_offset = 0L, y_offset = 0L, scale = 1L)
```

Arguments

x	Numeric data object with 2 columns (or scaler) to scale and offset.
x_offset	Numeric value to pan in the x-direction.
y_offset	Numeric value to pan in the x-directionl
scale	Numeric value to scale/zoom the size for x.

Value

Scaled and offset x. A manual variant of set_axes_position().

See Also

set_axes_position for preset choices.

Examples

```
ib <- tourr::basis_init(6, 2)
pan_zoom(x = ib, x_offset = -1, y_offset = 0, scale = 2/3)
```

pch_of *Return shape integers for a given discrete categorical variable.*

Description

Return shape integers for a given discrete categorical variable.

Usage

```
pch_of(class)
```

Arguments

class	The discrete categorical variable to return the shape of.

Value

Vector of integer shape values of the discrete categorical variable.

Examples

```
pch_of(tourr::flea$species)
```

play_manual_tour *Animate a manual tour*

Description

Performs the a manual tour and returns an animation of `render_type`. For use with `tourr::save_history()` tour paths see `play_tour_path()`.

Usage

```
play_manual_tour(
  basis = NULL,
  data,
  manip_var,
  render_type = render_plotly,
  rescale_data = FALSE,
  theta = NULL,
  phi_min = 0L,
  phi_max = 0.5 * pi,
  angle = 0.05,
  ...
)
```

Arguments

basis	A (p, d) orthonormal numeric matrix. The linear combination the original variables contribute to projection space. Defaults to NULL, generating a random basis.
data	(n, p) dataset to project, consisting of numeric variables.
manip_var	Integer column number or string exact column name of the. variable to manipulate. Required, no default.
render_type	Which graphics to render to. Defaults to render_plotly,
rescale_data	When TRUE scales the data to between 0 and 1.
theta	Angle in radians of "in-plane" rotation, on the xy plane of the reference frame. Defaults to theta of the basis for a radial tour.
phi_min	Minimum value phi should move to. Phi is angle in radians of the "out-of-plane" rotation, the z-axis of the reference frame. Required, defaults to 0.
phi_max	Maximum value phi should move to. Phi is angle in radians of the "out-of-plane" rotation, the z-axis of the reference frame. Required, defaults to pi/2.
angle	Target distance (in radians) between steps. Defaults to .05.
...	Optionally pass additional arguments to the render_type for projection point aesthetics; geom_point(aes(...)). OR passes optional arguments to manual_tour,

Value

An animation of a radial tour.

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
rb <- tourr::basis_random(n = ncol(flea_std))
class <- tourr::flea$species

## Not run:
play_manual_tour(basis = rb, data = flea_std, manip_var = 4)

play_manual_tour(basis = rb, data = flea_std, manip_var = 6, theta = .5 * pi,
                 render_type = render_gganimate, col = class, pch = class,
                 axes = "bottomleft", fps = 5)

## End(Not run)
```

---

play_tour_path *Render display of a provided tour path*

Description

Takes the result of tourr::save_history() or manual_tour(), interpolates over the path and renders into a selected render_type.

Usage

```
play_tour_path(
  tour_path,
  data = NULL,
  angle = 0.15,
  render_type = render_plotly,
  rescale_data = FALSE,
  ...
)
```

Arguments

tour_path	The result of tourr::save_history() or manual_tour().
data	Optional, number of columns must match that of tour_path.
angle	Target distance (in radians) between steps. Defaults to .15.
render_type	Graphics to render to. Defaults to render_plotly, alternative use render_gganimate.
rescale_data	When TRUE scales the data to between 0 and 1. Defaults to FALSE.
...	Optionally pass additional arguments to the render_type for projection point aesthetics; geom_point(aes(...)).

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
tpath <- tourr::save_history(flea_std, tour_path = tourr::grand_tour(), max = 3)
class <- tourr::flea$species

## Not run:
play_tour_path(tour_path = tpath, data = flea_std)

play_tour_path(tour_path = tpath, data = flea_std, angle = .25, fps = 4,
  render_type = render_gganimate, color = class, shape = class, axes = "bottomleft")

## End(Not run)
```

---

render_ *Prepate the ggplot object before passing to either animation package.*

Description

Typically called by render_plotly() or render_gganimate(). Takes the result of array2df(), and renders them into a ggplot2 object.

Usage

```
render_(slides, axes = "center", manip_col = "blue", ...)
```

Arguments

slides	The result of `array2df()`, a long df of the projected frames.
axes	Position of the axes: "center", "bottomleft", "off", "left", "right". Defaults to "center".
manip_col	String of the color to highlight the `manip_var` with. Defaults to "blue".
...	Optionally passes arguments to the projection points inside the aesthetics; `geom_point(aes(...))`.

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
rb <- tourr::basis_random(n = ncol(flea_std))
mtour <- manual_tour(basis = rb, manip_var = 4)
sshow <- array2df(array = mtour, data = flea_std)

render_(slides = sshow)

render_(slides = sshow, axes = "bottomleft",
        col = tourr::flea$species, pch = tourr::flea$species, cex = 2, alpha = .5)

render_(slides = sshow, axes = "bottomleft",
        col = tourr::flea$species, pch = tourr::flea$species, cex = 2, alpha = .5)
```

`render_gganimate` *Render the slides as a* gganimate *animation.*

Description

Takes the result of `array2df()` and renders them into a *gganimate* animation.

Usage

```
render_gganimate(
  fps = 3L,
  rewind = FALSE,
  start_pause = 1L,
  end_pause = 3L,
  gif_filename = NULL,
  gif_path = NULL,
  ...
)
```

Arguments

fps	Frames/slides shown per second. Defaults to 3.
rewind	Logical, should the animation play backwards after reaching the end? Default to FALSE.

start_pause	Number of seconds to pause on the first frame for. Defaults to 1.
end_pause	Number of seconds to pause on the last frame for. Defaults to 3.
gif_filename	Optional, saves the animation as a GIF to this string (without folderpath) . Defaults to NULL (no GIF saved). For more control call gganimate::anim_save() on a return object of render_gganimate().
gif_path	Optional, A string of the directory path (without filename) to save a GIF to. Defaults to NULL (current work directory).
...	Optionally passes arguments to the projection points inside the aesthetics; geom_point(aes(...)).

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
flea_class <- tourr::flea$species
rb <- tourr::basis_random(n = ncol(flea_std))
mtour <- manual_tour(basis = rb, manip_var = 4)
sshow <- array2df(array = mtour, data = flea_std)
## Not run:
render_gganimate(slides = sshow)

render_gganimate(slides = sshow, axes = "bottomleft", fps = 2, rewind = TRUE,
  col = flea_class, pch = flea_class, size = 2, alpha = .6)

if(F){
  render_gganimate(slides = sshow, axes = "right", fps = 4, rewind = TRUE,
    col = flea_class, pch = flea_class, size = 2,
    gif_filename = "myRadialTour.gif", gif_path = "./docs")
}

## End(Not run)
```

---

render_plotly *Render the slides as a* plotly *animation.*

Description

Takes the result of array2df() and renders them into a *plotly* animation.

Usage

```
render_plotly(fps = 3L, tooltip = "none", html_filename = NULL, ...)
```

Arguments

fps	Frames/slides shown per second. Defaults to 3.
tooltip	Character vector of aesthetic mappings to show in the plotly hover-over tooltip. Defaults to "none". "all" shows all the aesthetic mappings. The order of variables controls the order they appear. For example, tooltip = c("id", "frame", "x", "y", "category", "color").

html_filename	Optional, saves the plotly object as an HTML widget to this string (without folderpath). Defaults to NULL (not saved). For more control call htmlwidgets::saveWidget() on a return object of render_plotly().
...	Optionally passes arguments to the projection points inside the aesthetics; geom_point(aes(...)).

---

rotate_manip_space *Performs a rotation on the manipulation space of the given manip var.*

Description

A specific R3 rotation of the manipulation space for a 2D tour. Typically called by manual_tour(). The first 2 columns are x and y in the projection plane. The 3rd column extends "in the z-direction" orthogonal to the projection plane.

Usage

```
rotate_manip_space(manip_space, theta, phi)
```

Arguments

manip_space	A (p, d+1) dim matrix (manipulation space) to be rotated.
theta	Angle (radians) of "in-projection-plane" rotation (ie. on xy- of the projection). Typically set by the manip_type argument in proj_data().
phi	Angle (radians) of "out-of-projection-plane" rotation (ie. into the z-direction of the manipulation space. Effectively changes the norm of the manip_var in the projection plane.

Value

A (p, d+1) orthonormal matrix of the rotated (manipulation) space. The first 2 columns are x and y in the projection plane. The 3rd column extends "in the z-direction" orthogonal to the projection plane.

Examples

```
flea_std <- tourr::rescale(tourr::flea[,1:6])
rb  <- tourr::basis_random(n = ncol(flea_std))
msp <- create_manip_space(basis = rb, manip_var = 4)

rotate_manip_space(msp, theta = runif(1, max = 2 * pi),
                   phi = runif(1, max = 2 * pi) )
```

run_app *Runs a shiny app demonstrating manual tours*

Description

Runs a local shiny app that demonstrates manual tour and comparable traditional techniques for static projections of multivariate data sets.

Usage

```
run_app(app_nm = "intro", ...)
```

Arguments

app_nm	name of the shiny app to run. Expects "intro".
...	Other arguments passed into `shiny::runApp()`. Such as display.mode = "showcase".

Value

opens a local shiny app

Examples

```
## Not run:
run_app(app_nm= "intro")
run_app(app_nm= "intro", display.mode = "showcase")

## End(Not run)
```

set_axes_position *Returns the axis scale and position.*

Description

Typically called, by other functions to scale axes.

Usage

```
set_axes_position(x, axes)
```

Arguments

x	Numeric data object to scale and offset.
axes	Position of the axes: "center", "bottomleft" or "off". Defaults to "center".

Value

Scaled and offset ‘x“ typically controling axes placement.

See Also

pan_zoom to set the scale and offset.

Examples

```
rb <- tourr::basis_random(4, 2)
set_axes_position(x = rb, axes = "bottomleft")
```

---

spinifex *spinifex*

Description

spinifex is a package that extends the package tourr. It builds the functionality for manual tours and allows other tours to be rendered by plotly or gganimate. Tours are a class of dynamic linear (orthogonal) projections of numeric multivariate data from p down to d dimensions that are viewed as an animation as p-space is rotated. Manual tours manipulate a selected variable, exploring how they contribute to the sensitivity of the structure in the projection. This is particularly useful after finding an interesting tour, perhaps via a guided tour.

Details

Its main functions are:

- run_app(), running run_app("intro") will open an introductory shiny app demonstrating radial tours.
- play_manual_tour(), performs a manual tour, returning a plotly animate by default.
- play_tour_path(), turns a tour path into an animation, returning a plotly object by default.
- view_basis(), plot a basis set on a reference axis.
- view_manip_space(), plot a manipulation space highlighting the manip var.

GitHub: https://github.com/nspyrison/spinifex

Author(s)

Maintainer: Nicholas Spyrison <spyrison@gmail.com>

Authors:

- Dianne Cook [thesis advisor]

See Also

tourr (package)

theme_spinifex *Aesthetic settings that can be applied to a ggplot object.*

Description

A ggplot2 theme (group of aesthetic settings), that can be added to a ggplot.

Usage

```
theme_spinifex()
```

view_basis *Plot the axes directions of the basis table without data points.*

Description

ggplot2 object of axes contribution inscribed in a unit circle.

Usage

```
view_basis(basis, data = NULL, lab = NULL, axes = "center", ...)
```

Arguments

basis	A (p, d) orthonormal numeric matrix. The linear combination the original variables contribute to projection space. Required, no default.
data	Optional (n, p) data to plot on through the projection basis.
lab	Optional, labels for the reference frame of length 1 or the number of variables used. By default will abbreviate data if available.
axes	Position of the axes: "center", "bottomleft" or "off". Defaults to "center".
...	Optionally passes arguments to the projection points inside the aesthetics; geom_point(aes(...)).

Value

ggplot object of the basis.

Examples

```
rb <- tourr::basis_random(4, 2)
view_basis(basis = rb)

flea_std <- tourr::rescale(tourr::flea[, 1:4])
view_basis(basis = rb, data = flea_std, axes = "bottomleft")

view_basis(basis = rb, data = flea_std, axes = "right",
           col = col_of(tourr::flea[, 7], "Paired"),
           pch = pch_of(tourr::flea[,7]),
           alpha = .7, size = 2)
```

`view_manip_space` *Plot projection frame and return the axes table.*

Description

Uses base graphics to plot the circle with axes representing the projection frame. Returns the corresponding table.

Usage

```
view_manip_space(
  basis,
  manip_var,
  tilt = 1/12 * pi,
  lab = paste0("V", 1:nrow(basis)),
  manip_col = "blue",
  z_col = "red"
)
```

Arguments

`basis`	A (p, d) orthonormal numeric matrix. The linear combination the original variables contribute to projection space. Required, no default.
`manip_var`	Number of the column/dimension to rotate.
`tilt`	angle in radians to rotate the projection plane. Defaults to pi * 5/12.
`lab`	Optional, character vector of `p` length, add name to the axes in the reference frame, typically the variable names.
`manip_col`	String of the color to highlight the `manip_var`.
`z_col`	Color to illustrate the z direction or out of the projection plane.

Value

ggplot object of the basis.

Examples

```
flea_std <- tourr::rescale(tourr::flea[, 1:6])
rb <- tourr::basis_random(ncol(flea_std), 2)

view_manip_space(basis = rb, manip_var = 4)
```

`weather` *Sample dataset of daily weather observations from Canberra airport in Australia.*

Description

A subset from `rattle.data::weather`, instructions to reproduce below.

Usage

```
weather
```

Format

Data frame of 366 observations of 20 variables, one year of daily observations of weather variables at Canberra airport in Australia starting November 2007.

Details

One year of daily weather observations collected from the Canberra airport in Australia was obtained from the Australian Commonwealth Bureau of Meteorology and processed to create this sample dataset for illustrating data mining using R and Rattle.

The data has been processed to provide a target variable RainTomorrow (whether there is rain on the following day - No/Yes) and a risk variable 'RISK_MM" (how much rain recorded in millimeters). Various transformations were performed on the source data. The dataset is quite small and is useful only for repeatable demonstration of various data science operations.

The source dataset is Copyright by the Australian Commonwealth Bureau of Meteorology and is provided as part of the rattle package with permission.

Data frame of 366 observations of 20 variables, one year of daily observations of weather variables at Canberra airport in Australia starting November 2007:

- Date, The date of observation (a Date object).
- MinTemp, The minimum temperature in degrees Celsius.
- MaxTemp, The maximum temperature in degrees Celsius.
- Rainfall, The amount of rainfall recorded for the day in mm.
- Evaporation, The so-called Class A pan evaporation (mm) in the 24 hours to 9am.
- Sunshine, The number of hours of bright sunshine in the day.
- WindGustSpeed, The speed (km/h) of the strongest wind gust in the 24 hours to midnight.
- WindSpeed9am, Wind speed (km/hr) averaged over 10 minutes prior to 9am.
- WindSpeed3pm, Wind speed (km/hr) averaged over 10 minutes prior to 3pm.
- Humid9am, Relative humidity (percent) at 9am.
- Humid3pm, Relative humidity (percent) at 3pm.
- Pressure9am, Atmospheric pressure (hpa) reduced to mean sea level at 9am.

- Pressure3pm, Atmospheric pressure (hpa) reduced to mean sea level at 3pm.
- Cloud9am, Fraction of sky obscured by cloud at 9am. This is measured in "oktas", which are a unit of eighths. It records how many eighths of the sky are obscured by cloud. A 0 measure indicates completely clear sky whilst an 8 indicates that it is completely overcast.
- Cloud3pm, Fraction of sky obscured by cloud (in "oktas": eighths) at 3pm. See Cloud9am for a description of the values.
- Temp9am, Temperature (degrees C) at 9am.
- Temp3pm, Temperature (degrees C) at 3pm.
- RainToday, Integer: 1 if precipitation (mm) in the 24 hours to 9am exceeds 1mm, otherwise 0.
- RISK_MM, The amount of rain. A kind of measure of the "risk".
- RainTomorrow, The target variable. Did it rain tomorrow?

Reproducing this dataset:

```
requireNamespace("rattle.data")
weather <- weather[, c(1,3:7,9,12:24)]
```

Source

The daily observations are available from http://www.bom.gov.au/climate/data. Copyright Commonwealth of Australia 2010, Bureau of Meteorology.

Definitions adapted from http://www.bom.gov.au/climate/dwo/IDCJDW0000.shtml

References

Data source: http://www.bom.gov.au/climate/dwo/ and http://www.bom.gov.au/climate/data.

Examples

```
str(weather)
## Not run:
play_manual_tour(data = weather[, 2:17], manip_var = 5, rescale_data = TRUE)

## End(Not run)
```

---

wine *The wine dataset from the UCI Machine Learning Repository.*

Description

The wine dataset contains the results of a chemical analysis of wines grown in a specific area of Italy. Three types of wine are represented in the 178 samples, with the results of 13 chemical analyses recorded for each sample. The Type variable has been transformed into a categorical variable.

Usage

```
wine
```

Format

data frame of 178 observations of 13 variables, target class Type and 12 numeric variables.

Details

The data contains no missing values and consist of only numeric data, with a three class target variable (Type) for classification.

Data frame of 178 observations of 13 variables, target class Type and 12 numeric variables:

- Type, The type of wine, into one of three classes, 1 (59 obs), 2(71 obs), and 3 (48 obs).
- Alcohol, Alcohol
- Malic, Malic acid
- Ash, Ash
- Alcalinity, Alcalinity of ash
- Magnesium, Magnesium
- Phenols, Total phenols
- Flavanoids, Flavanoids
- Nonflavanoids, Nonflavanoid phenols
- Proanthocyanins, Proanthocyanins
- Color, Color intensity
- Hue, Hue
- Dilution, D280/OD315 of diluted wines
- Proline, Proline

Reproducing this dataset:

```
requireNamespace("rattle.data")
wine
```

Examples

```
str(wine)
## Not run:
play_manual_tour(data = wine[, 2:14], manip_var = 1, rescale_data = TRUE)

## End(Not run)
```

Index