Package ‘rmweather’

June 15, 2020

Type Package

Title Tools to Conduct Meteorological Normalisation on Air Quality Data

Version 0.1.51

Date 2020-06-11

Maintainer Stuart K. Grange <stuart.grange@york.ac.uk>

Description An integrated set of tools to allow data users to conduct meteorological normalisation on air quality data. This meteorological normalisation technique uses predictive random forest models to remove variation of pollutant concentrations so trends and interventions can be explored in a robust way. For examples, see Grange et al. (2018) <doi:10.5194/acp-18-6223-2018> and Grange and Carslaw (2019) <doi:10.1016/j.scitotenv.2018.10.344>.

URL https://github.com/skgrange/rmweather

BugReports https://github.com/skgrange/rmweather/issues

License GPL-3 | file LICENSE

ByteCompile true

Depends R (>= 3.2.0)

Imports dplyr, ggplot2, lubridate, magrittr, pdp, purrr, ranger, stringr, strucchange, tibble, viridis

Suggests testthat, openair

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Author Stuart K. Grange [cre, aut] (<https://orcid.org/0000-0003-4093-3596>)

Repository CRAN

Date/Publication 2020-06-15 06:50:03 UTC
Example observational data for the rmweather package.

Description

These example data are daily means of NO2 and NOx observations at London Marylebone Road. The accompanying surface meteorological data are from London Heathrow, a major airport located 23 km west of Central London.

Usage

data_london

Format

Tibble with 15676 observations and 11 variables. The variables are: date, date_end, site, site_name, value, air_temp, atmospheric_pressure, rh, wd, and ws. The dates are in POSIXct format, the site variables are characters and all other variables are numeric.
Details

The NO2 and NOx observations are sourced from the European Commission Air Quality e-Reporting repository which can be freely shared with acknowledgement of the source. The meteorological data are sourced from the Integrated Surface Data (ISD) database which cannot be redistributed for commercial purposes and are bound to the WMO Resolution 40 Policy.

Author(s)

Stuart K. Grange

Examples

```r
# Load rmweather's example data and check
head(data_london)
```

---

data_london_normalised

*Example of meteorologically normalised data for the rmweather package.*

Description

These example data are derived from the observational data included in rmweather and represent meteorologically normalised NO2 concentrations at London Marylebone Road, aggregated to monthly resolution.

Usage

```
data_london_normalised
```

Format

Tibble with 258 observations and 5 variables. The variables are: date, date_end, site, site_name, and value_predict. The dates are in POSIXct format, the site variables are characters and value_predict is numeric.

Author(s)

Stuart K. Grange

See Also

```
data_london
```
Examples

# Load rmweather's meteorologically normalised example data and check
head(data_london_normalised)

---

model_london

Example **ranger** random forest model for the **rmweather** package.

Description

This example object was created from the observational data included in **rmweather** and is a random forest model returned by **rmw_train_model**. This forest is only made from one tree to keep the file size small and is only used for the package’s examples.

Usage

model_london

Format

A ranger object, a named list with 16 elements.

Author(s)

Stuart K. Grange

See Also

data_london, rmw_train_model

Examples

# Load rmweather's ranger model example data and see what elements it contains
names(model_london)

# Print ranger object
print(model_london)
**rmw_clip**

Function to "clip" the edges of a normalised time series after being produced with `rmw_normalise`.

**Description**

`rmw_clip` helps if the random forest model behaves strangely at the beginning and end of the time series during prediction.

**Usage**

```
rmw_clip(df, seconds = 31536000/2)
```

**Arguments**

- `df` Data frame from `rmw_normalise`.
- `seconds` Number of seconds to clip from start and end of time-series. The default is half a year.

**Value**

Data frame.

**Author(s)**

Stuart K. Grange

**See Also**

`rmw_normalise`, `rmw_plot_normalised`

**Examples**

```
# Clip the edges of a normalised time series, default is half a year
data_normalised_clipped <- rmw_clip(data_london_normalised)
```
**rmw_do_all**

*Function to train a random forest model to predict (usually) pollutant concentrations using meteorological and time variables and then immediately normalise a variable for “average” meteorological conditions.*

---

**Description**

rmw_do_all is a user-level function to conduct the meteorological normalisation process in one step.

**Usage**

```
rmw_do_all(
  df, variables, variables_sample = NA, n_trees = 300,
  min_node_size = 5, mtry = NULL, keep_inbag = TRUE,
  n_samples = 300, replace = TRUE, se = FALSE,
  aggregate = TRUE, n_cores = NA, verbose = FALSE
)
```

**Arguments**

- `df` Input data frame after preparation with `rmw_prepare_data`. `df` has a number of constraints which will be checked for before modelling.
- `variables` Independent/explanatory variables used to predict "value".
- `variables_sample` Variables to use for the normalisation step. If not used, the default of all variables used for training the model with the exception of `date_unix`, the trend term (see `rmw_normalise`).
- `n_trees` Number of trees to grow to make up the forest.
- `min_node_size` Minimal node size.
- `mtry` Number of variables to possibly split at in each node. Default is the (rounded down) square root of the number variables.
- `keep_inbag` Should in-bag data be kept in the **ranger** model object? This needs to be TRUE if standard errors are to be calculated when predicting with the model.
- `n_samples` Number of times to sample `df` and then predict?
replace Should variables be sampled with replacement?

se Should the standard error of the predictions be calculated too? The standard error method is the “infinitesimal jackknife for bagging” and will slow down the predictions significantly.

aggregate Should all the n_samples predictions be aggregated?

n_cores Number of CPU cores to use for the model calculation. Default is system’s total minus one.

verbose Should the function give messages?

Value
Named list.

Author(s)
Stuart K. Grange

See Also
rmw_prepare_data, rmw_train_model, rmw_normalise

Examples

# Load package
library(dplyr)

# Keep things reproducible
set.seed(123)

# Prepare example data
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()

# Use the example data to conduct the steps needed for meteorological # normalisation
list_normalised <- rmw_do_all(
  df = data_london_prepared,
  variables = c("ws", "wd", "air_temp", "rh", "date_unix", "day_julian", "weekday", "hour"),
  n_trees = 300,
  n_samples = 300
)
**rmw_find_breakpoints**  
*Function to detect breakpoints in a data frame using a linear regression based approach.*

**Description**

`rmw_find_breakpoints` will generally be applied to a data frame after `rmw_normalise`. `rmw_find_breakpoints` is rather slow.

**Usage**

```r
rmw_find_breakpoints(df, h = 0.15, n = NULL)
```

**Arguments**

- **df**  
  Tibble from `rmw_normalise` to detect breakpoints in.
- **h**  
  Minimal segment size either given as fraction relative to the sample size or as an integer giving the minimal number of observations in each segment.
- **n**  
  Number of breaks to detect. Default is maximum number allowed by `h`.

**Value**

Tibble with a date variable indicating where the breakpoints are.

**Author(s)**

Stuart K. Grange

**Examples**

```r
# Test for breakpoints in an example normalised time series
data_breakpoints <- rmw_find_breakpoints(data_london_normalised)
```

---

**rmw_model_statistics**  
*Functions to extract model statistics from a model calculated with `rmw_calculate_model`.*

**Description**

Functions to extract model statistics from a model calculated with `rmw_calculate_model`. 
**Usage**

```r
rmw_model_statistics(model)
rmw_model_importance(model, date_unix = TRUE)
```

**Arguments**

- `model`: A ranger model object from `rmw_calculate_model`.
- `date_unix`: Should the `date_unix` variable be included in the return?

**Details**

The variable importances are defined as "the permutation importance differences of predictions errors". This measure is unit-less and the values are not useful when comparing among data sets.

**Value**

Tibble.

**Author(s)**

Stuart K. Grange

**Examples**

```r
# Extract statistics from the example random forest model
rmw_model_statistics(model_london)

# Extract importances from a model object
rmw_model_importance(model_london)
```

---

**Description**

Function to normalise a variable for "average" meteorological conditions.
Usage

```r
rmw_normalise(
  model,
  df,
  variables = NA,
  n_samples = 300,
  replace = TRUE,
  se = FALSE,
  aggregate = TRUE,
  n_cores = NA,
  verbose = FALSE
)
```

Arguments

- **model**: A ranger model object from `rmw_train_model`.
- **df**: Input data used to calculate model using `rmw_prepare_data`.
- **variables**: Variables to randomly sample. Default is all variables used for training the model with the exception of `date_unix`, the trend term.
- **n_samples**: Number of times to sample `df` and then predict?
- **replace**: Should `variables` be sampled with replacement?
- **se**: Should the standard error of the predictions be calculated too? The standard error method is the “infinitesimal jackknife for bagging” and will slow down the predictions significantly.
- **aggregate**: Should all the `n_samples` predictions be aggregated?
- **n_cores**: Number of CPU cores to use for the model predictions. Default is system’s total minus one.
- **verbose**: Should the function give messages?

Value

Tibble.

Author(s)

Stuart K. Grange

See Also

- `rmw_prepare_data`, `rmw_train_model`

Examples

```r
# Load package
```
library(dplyr)

# Keep things reproducible
set.seed(123)

# Prepare example data
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()

# Normalise the example no2 data
data_normalised <- rmw_normalise(
  model_london,
  df = data_london_prepared,
  n_samples = 300,
  verbose = TRUE
)

---

rmw_partial_dependencies

*Function to calculate partial dependencies after training with* *rmweather.*

**Description**

*rmw_plot_partial_dependencies* is rather slow.

**Usage**

```
rmw_partial_dependencies(model, df, variable, n_cores = NA, verbose = FALSE)
```

**Arguments**

- `model`: A ranger model object from *rmw_train_model*.
- `df`: Input data frame after preparation with *rmw_prepare_data*.
- `variable`: Vector of variables to calculate partial dependencies for.
- `n_cores`: Number of CPU cores to use for the model calculation. Default is system’s total minus one.
- `verbose`: Should the function give messages?

**Value**

Tibble.
Author(s)

Stuart K. Grange

Examples

```r
# Load packages
library(dplyr)
# Ranger package needs to be loaded
library(ranger)

# Prepare example data
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()

# Calculate partial dependencies for wind speed
data_partial <- rmw_partial_dependencies(
  model = model_london,
  df = data_london_prepared,
  variable = "ws",
  verbose = TRUE
)

# Calculate partial dependencies for all independent variables used in model
data_partial <- rmw_partial_dependencies(
  model = model_london,
  df = data_london_prepared,
  variable = NA,
  verbose = TRUE
)
```

---

**rmw_plot_importance**

Function to plot random forest variable importances after training by `rmw_train_model`.

**Description**

Function to plot random forest variable importances after training by `rmw_train_model`.

**Usage**

```r
rmw_plot_importance(df, colour = "black")
```
Function to plot the meteorologically normalised time series after `rmw_normalise`.

**Description**

If the input data contains a standard error variable named "se", this will be plotted as a ribbon (+ and -) around the mean.

**Usage**

```r
rmw_plot_normalised(df, colour = "#6B18EF")
```

**Arguments**

- `df` Tibble created by `rmw_normalise`.
- `colour` Colour for line geometry.

**Value**

`ggplot2` plot with a line and ribbon geometries.

**Author(s)**

Stuart K. Grange

**Examples**

```r
# Plot normalised example data
rmw_plot_normalised(data_london_normalised)
```
**rmw_plot_partial_dependencies**

*Function to plot partial dependencies after calculation by \texttt{rmw_partial_dependencies}.*

**Description**

Function to plot partial dependencies after calculation by \texttt{rmw_partial_dependencies}.

**Usage**

\texttt{rmw_plot_partial_dependencies(df)}

**Arguments**

- \texttt{df} Tibble created by \texttt{rmw_partial_dependencies}.

**Value**

\texttt{ggplot2} plot with a point geometry.

**Author(s)**

Stuart K. Grange

---

**rmw_plot_test_prediction**

*Function to plot the test set and predicted set after \texttt{rmw_predict_the_test_set}.*

**Description**

Function to plot the test set and predicted set after \texttt{rmw_predict_the_test_set}.

**Usage**

\texttt{rmw_plot_test_prediction(df, bins = 30, coord_equal = TRUE)}

**Arguments**

- \texttt{df} Tibble created by \texttt{rmw_predict_the_test_set}.
- \texttt{bins} Numeric vector giving number of bins in both vertical and horizontal directions.
- \texttt{coord_equal} Should axes be forced to be equal?
**Value**

ggplot2 plot with a hex geometry.

**Author(s)**

Stuart K. Grange

---

**rmw_predict**  
*Function to predict using a *ranger* random forest.*

---

**Description**

Function to predict using a *ranger* random forest.

**Usage**

```r
rmw_predict(model, df = NA, se = FALSE, n_cores = NULL, verbose = FALSE)
```

**Arguments**

- `model`  
  A *ranger* model object from `rmw_train_model`.
- `df`  
  Input data to be used for predictions.
- `se`  
  If `df` is supplied, should the standard error of the prediction be calculated too? The standard error method is the “infinitesimal jackknife for bagging” and will slow down the predictions significantly.
- `n_cores`  
  Number of CPU cores to use for the model predictions.
- `verbose`  
  Should the function give messages?

**Value**

Numeric vector or a named list containing two numeric vectors.

**Author(s)**

Stuart K. Grange

**Examples**

```r
# Load package
library(dplyr)

# Prepare example data
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()
```
# Make a prediction with the examples
vector_prediction <- rmw_predict(
  model_london,
  df = data_london_prepared
)

# Make a prediction with standard errors too
list_prediction <- rmw_predict(
  model_london,
  df = data_london_prepared,
  se = TRUE
)

---

rmw_predict_the_test_set

*Functions to use a model to predict the observations within a test set after rmw_calculate_model.*

---

**Description**

*rmw_predict_the_test_set* uses data withheld from the training of the model and therefore can be used for investigating overfitting.

**Usage**

`rmw_predict_the_test_set(model, df)`

**Arguments**

- `model` A ranger model object from `rmw_calculate_model`.
- `df` Input data used to calculate model.

**Value**

Tibble.

**Author(s)**

Stuart K. Grange

**Examples**

```r
# Load package
library(dplyr)

# Prepare example data
```
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()

# Use the test set for prediction
rmw_predict_the_test_set(
  model_london,
  df = data_london_prepared
)

# Predict, then produce a hex plot of the predictions
rmw_predict_the_test_set(
  model_london,
  df = data_london_prepared
) %>%
  rmw_plot_test_prediction()

---

### rmw_prepare_data

Function to prepare a data frame for modelling with rmweather.

#### Description

rmw_prepare_data will test and prepare a data frame for further use with rmweather.

#### Usage

```
rmw_prepare_data(
  df, 
  value = "value",
  na.rm = FALSE, 
  replace = FALSE, 
  fraction = 0.8 
)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>Input data frame. Generally a time series of air quality data with pollutant concentrations and meteorological variables.</td>
</tr>
<tr>
<td>value</td>
<td>Name of the dependent variable. Usually a pollutant, for example, &quot;no2&quot; or &quot;pm10&quot;.</td>
</tr>
<tr>
<td>na.rm</td>
<td>Should missing values (NA) be removed from value?</td>
</tr>
<tr>
<td>replace</td>
<td>When adding the date variables to the set, should they replace the versions already contained in the data frame if they exist?</td>
</tr>
<tr>
<td>fraction</td>
<td>Fraction of the observations to make up the training set. Default is 0.8, 80%.</td>
</tr>
</tbody>
</table>
Details

`rmw_prepare_data` will check if a date variable is present and is of the correct data type, impute missing numeric and categorical values, randomly split the input into training and testing sets, and rename the dependent variable to “value”. The date variable will also be used to calculate new variables such as `date_unix`, `day_julian`, `weekday`, and `hour` which can be used as independent variables. These attributes are needed for other `rmweather` functions to operate.

Use `set.seed` in an R session to keep results reproducible.

Value

Tibble, the input data transformed ready for modelling with `rmweather`.

Author(s)

Stuart K. Grange

See Also

`set.seed`, `rmw_train_model`, `rmw_normalise`

Examples

```r
# Load package
library(dplyr)

# Keep things reproducible
set.seed(123)

# Prepare example data for modelling, only use no2 data here
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>%
  rmw_prepare_data()
```

---

**rmw_train_model**

*Function to train a random forest model to predict (usually) pollutant concentrations using meteorological and time variables.*

Description

Function to train a random forest model to predict (usually) pollutant concentrations using meteorological and time variables.
Usage

```r
rmw_train_model(
  df,
  variables,
  n_trees = 300,
  mtry = NULL,
  min_node_size = 5,
  keep_inbag = TRUE,
  n_cores = NA,
  verbose = FALSE
)
```

Arguments

- **df**: Input tibble after preparation with `rmw_prepare_data`. `df` has a number of constraints which will be checked for before modelling.
- **variables**: Independent/explanatory variables used to predict "value".
- **n_trees**: Number of trees to grow to make up the forest.
- **mtry**: Number of variables to possibly split at in each node. Default is the (rounded down) square root of the number variables.
- **min_node_size**: Minimal node size.
- **keep_inbag**: Should in-bag data be kept in the `ranger` model object? This needs to be `TRUE` if standard errors are to be calculated when predicting with the model.
- **n_cores**: Number of CPU cores to use for the model calculation. Default is system’s total minus one.
- **verbose**: Should the function give messages?

Value

A `ranger` model object, a named list.

Author(s)

Stuart K. Grange

See Also

`rmw_prepare_data`, `rmw_normalise`

Examples

```r
# Load package
library(dplyr)
```
# Keep things reproducible
set.seed(123)

# Prepare example data
data_london_prepared <- data_london %>%
  filter(variable == "no2") %>
  rmw_prepare_data()

# Calculate a model using common meteorological and time variables
model <- rmw_train_model(
data_london_prepared,
  variables = c(
    "ws", "wd", "air_temp", "rh", "date_unix", "day_julian", "weekday", "hour"
  ),
  n_trees = 300
)

---

system_cpu_core_count  Function to return the system’s number of CPU cores.

---

**Description**

Function to return the system’s number of CPU cores.

**Usage**

```
system_cpu_core_count(logical_cores = TRUE)
```

**Arguments**

- `logical_cores`  Should logical cores be included in the core count?

**Author(s)**

Stuart K. Grange
wday_monday

---

wday_monday Function to get weekday number from a date where 1 is Monday and 7 is Sunday.

Description

Function to get weekday number from a date where 1 is Monday and 7 is Sunday.

Usage

wday_monday(x, as.factor = FALSE)

Arguments

- **x** Date vector.
- **as.factor** Should the return be a factor?

Value

Numeric vector.

Author(s)

Stuart K. Grange

%% Pseudo-function to re-export magrittr's pipe.

---

Description

Pseudo-function to re-export magrittr's pipe.
Index

*Topic datasets
  data_london, 2
  data_london_normalised, 3
  model_london, 4
  %>%, 21

  data_london, 2, 3, 4
  data_london_normalised, 3
  model_london, 4
  rmw_clip, 5
  rmw_do_all, 6
  rmw_find_breakpoints, 8
  rmw_model_importance, 13
  rmw_model_importance
    (rmw_model_statistics), 8
  rmw_model_statistics, 8
  rmw_normalise, 5–8, 9, 13, 18, 19
  rmw_partial_dependencies, 11, 14
  rmw_plot_importance, 12
  rmw_plot_normalised, 5, 13
  rmw_plot_partial_dependencies, 14
  rmw_plot_test_prediction, 14
  rmw_predict, 15
  rmw_predict_the_test_set, 14, 16
  rmw_prepare_data, 6, 7, 10, 11, 17, 19
  rmw_train_model, 4, 7, 10–13, 18, 18

  set.seed, 18
  system_cpu_core_count, 20

  wday_monday, 21