Package ‘SmartMeterAnalytics’

October 12, 2022

Type Package
Title Methods for Smart Meter Data Analysis
Version 1.0.3
Date 2020-08-11
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Imports plyr, futile.logger, FNN, stinepack, zoo
Suggests stringr, knitr, rmarkdown, ROCR, randomForest, caret, dplyr
NeedsCompilation no
Author Konstantin Hopf [aut, cre] (<https://orcid.org/0000-0002-5452-0672>), Andreas Weigert [ctb], Ilya Kozlovskiy [ctb], Thorsten Staake [ctb] (<https://orcid.org/0000-0003-1399-4676>)
Maintainer Konstantin Hopf <konstantin.hopf@uni-bamberg.de>
Repository CRAN
Date/Publication 2020-08-18 09:30:07 UTC

R topics documented:
calc_features15_consumption ........................................... 2
calc_features30_consumption ........................................... 3
calc_features15_consumption

Calculates features from 15-min smart meter data

Description

Calculates features from 15-min smart meter data

Usage

calc_features15_consumption(
  B,
  rowname = NULL,
  featsCoarserGranularity = FALSE,
  replace_NA_with_defaults = TRUE
)

Arguments

- **B**: a vector with length 4*24*7 = 672 measurements in one day in seven days a week
- **rowname**: the row name of the resulting feature vector
- **featsCoarserGranularity**: are the features of finer granularity levels also to be calculated (TRUE/FALSE)
- **replace_NA_with_defaults**: replaces missing (NA) or infinite values that may appear during calculation with default values
Value

a data.frame with the calculated features as columns and a specified rowname, if given

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

References


Examples

# Create a random time series of 15-minute smart meter data (672 measurements per week)
smd <- runif(n=672, min=0, max=2)
# Calculate the smart meter data features
calc_features15_consumption(smd)

```
calc_features30_consumption

Calculates features from 30-min smart meter data

Description

Calculates features from 30-min smart meter data

Usage

calc_features30_consumption(
  B,
  rowname = NULL,
  featsCoarserGranularity = FALSE,
  replace_NA_with_defaults = TRUE
)
```
Arguments

- `B` a vector with length 2*24*7 = 336 measurements in one day in seven days a week
- `rowname` the row name of the resulting feature vector
- `featsCoarserGranularity` are the features of finer granularity levels also to be calculated (TRUE/FALSE)
- `replace_NA_with_defaults` replaces missing (NA) or infinite values that may appear during calculation with default values

Value

a data.frame with the calculated features as columns and a specified rowname, if given

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

References


Examples

# Create a random time series of 30-minute smart meter data (336 measurements per week)
smd <- runif(n=336, min=0, max=2)
# Calculate the smart meter data features
calc_features30_consumption(smd)
calc_features60_consumption

Calculates features from 15-min smart meter data

Description

Calculates features from 15-min smart meter data

Usage

calc_features60_consumption(B, rowname = NULL, replace_NA_with_defaults = TRUE)

Arguments

B
a vector with length 24*7 = 168 measurements in one day in seven days a week

rowname
the row name of the resulting feature vector

replace_NA_with_defaults
replaces missing (NA) or infinite values that may appear during calculation with default values

Value

a data.frame with the calculated features as columns and a specified rowname, if given the row name of the resulting feature vector

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

Examples

# Create a random time series of 60-minute smart meter data (168 measurements per week)
smd <- runif(n=168, min=0, max=2)
# Calculate the smart meter data features
calc_features60_consumption(smd)

calc_featuresco_consumption

Calculates consumption features from weekly consumption only

Description

Calculates consumption features from weekly consumption only
calc_featuresda_consumption

Usage

calc_featuresda_consumption(B, rowname = NULL)

Arguments

B  a vector of any length with measurements
rowname  the row name of the resulting feature vector

Value

a data.frame with the calculated features as columns and a specified rowname, if given

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

References


calc_featuresda_consumption

Calculates consumption features from daily smart meter data

Description

Calculates consumption features from daily smart meter data

Usage

calc_featuresda_consumption(
  B,
  rowname = NULL,
  featsCoarserGranularity = FALSE,
  replace_NA_with_defaults = TRUE
)
**Argument**

- **B**
  - a vector with length 7 measurements
- **rowname**
  - the row name of the resulting feature vector
- **featsCoarserGranularity**
  - are the features of finer granularity levels also to be calculated (TRUE/FALSE)
- **replace_NA_with_defaults**
  - replaces missing (NA) or infinite values that may appear during calculation with default values

**Value**

a data.frame with the calculated features as columns and a specified rowname, if given

**Author(s)**

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

**References**


---

**Description**

The division in HT / NT is done from the input smart meter data

**Usage**

```r
calc_featureshtnt_consumption2(HTCons, NTCons, rowname = NULL, featsCoarserGranularity = FALSE)
```

**Arguments**

- **HTCons**
  - a vector with 7 measurements for HT consumption in one week (beginning with monday)
- **NTCons**
  - a vector with 7 measurements for NT consumption in one week (beginning with monday)
calc_featuresnt_consumption

calculates consumption features from daily (HT / NT) smart meter data

Description
The division in HT / NT is done from the input smart meter data

Usage

```r
calc_featuresnt_consumption(
  B,
  rowname = NULL,
  featsCoarserGranularity = FALSE,
  replace_NA_with_defaults = TRUE
)
```

Arguments

- **B**
  a vector with length 2*24*7 = 336 measurements in one day in seven days a week
- **rowname**
  the row name of the resulting feature vector
- **featsCoarserGranularity**
  are the features of finer granularity levels also to be calculated (T/FALSE)
- **replace_NA_with_defaults**
  an optional boolean argument specifying if missing values will be replaced with standard values (i.e., zero values)

Details
HT consumption is during the time 07:00-22:00

Author(s)
Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

References
calc_features_daily_multipleTS

Author(s)
Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

References

calc_features_daily_multipleTS
Calculates feature from multiple time series data vectors

Description
This function is intended to compute features for daily consumption data from electricity, gas, and water consumption time series data.

Usage
calc_features_daily_multipleTS(
el = NULL,
gas = NULL,
wa = NULL,
rowname = NULL,
cor.useNA = "complete.obs"
)

Arguments
el  electricity consumption
gas gas consumption
wa water consumption
rowname the name of the consumer (e.g., a household ID in a study database)
cor.useNA an optional character string for the cor function, specifying a method for computing covariances in the presence of missing values.

Value
a data frame with feature values as columns, named by `rowname`
calc_features_weather  Calculates features from one environmental time-series variable and smart meter data

Description
Calculates features from one environmental time-series variable and smart meter data

Usage
calc_features_weather(SMD, WEATHER, rowname = NULL)

Arguments
SMD  the load trace for one week (vector with 672 or 336 elements)
WEATHER  weather observations (e.g. temperature) in 30-minute readings (vector with 336 elements)
rowname  the row name of the current data point

Author(s)
Konstantin Hopf <konstantin.hopf@uni-bamberg.de>, Ilya Kozlovskiy

References
encode_p_val_stars

Encode p-values with a star rating according to the Significance code:

Description

'.' for p-value < 0.1, '*' for < 0.05, '**' for < 0.01, ***' for < 0.001

Usage

```r
encode_p_val_stars(pval)
```

Arguments

- `pval`: the p-value

Value

character with the encoding

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

features_all_subsets

Creates a set of all combinations of features

Description

Creates a set of all combinations of features

Usage

```r
features_all_subsets(set)
```

Arguments

- `set`: vector of available features that are premutated

Value

a list of subsets of the input vector

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>, Ilya Kozlovskiy
getDay_ISO8601_week

Retrieves the date of the monday in a ISO8601 week-string

Examples

features_all_subsets(c("A", "B", "C"))

day_ISO8601_week

getDay_ISO8601_week(theweek, day = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")

Arguments

theweek the string with the week name
day the weekday that shall be returned

Details

The function uses format und as.Date internally and can therefore not handle ISO8601 week formats. Therefore, a workaround is implemented that can lead to suspicious behavior in future versions

Value

the date of the weekday in the given week

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>
getDay_US_week

Retrieves the date of the monday in a US week-string (as implemented by R as.Date)

Description

According to date formats defined by ISO 8601: * Single days are written in yyy-mm-dd (y: year, m: month, d: day); e.g., 2016-07-19 * Weeks are written in yyyy-WUww; e.g., 2016-WU29 (typically with the first Sunday of the year as day 1 of week 1)

Usage

getDay_US_week(
  theweek,
  day = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")
)

Arguments

theweek the string with the week name
day the weekday that shall be returned

Value

the date of the weekday in the given week

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

interpolate_missingReadings

Interpolate missing readings

Description

Interpolate missing readings

Usage

interpolate_missingReadings(timeseries, option = "linear", ...)

Arguments

- `timeseries` (vector or Time Series (ts)) object in which missing values shall be replaced.
- `option` Algorithm to be used. Accepts the following input:
  - "linear" - for linear interpolation using `approx`.
  - "spline" - for spline interpolation using `spline`.
  - "stine" - for Stineman interpolation using `stinterp`.
- Additional parameters to be passed through to `approx` or `spline` interpolation functions.

Details

Missing values get replaced by values of an `approx`, `spline` or `stinterp` interpolation.

Value

A vector (vector) or Time Series (ts) object (dependent on given input at parameter `x`).

Author(s)


naInf_omit

Removes the rows with NA or Inf values.

Description

Cleans up a data.frame or matrix which is useful for cases where you need complete datasets.

Usage

`naInf_omit(V)`

Arguments

- `V` A data.frame or matrix which has to be cleaned.

Value

A cleaned version of data.frame or matrix.

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

See Also

`replaceNAsFeatures`, `remove_empty_features`
occupancy_cluster

Determine two clusters of high and low consumption times (e.g., non-occupancy during holidays)

Usage

occupancy_cluster(consumption, n_days_check = 4, sds_between_clusters = 1.5)

Arguments

- consumption: the consumption time series
- n_days_check: number of consecutive days that should be considered as a minimal cluster
- sds_between_clusters: the multiples of standard deviation that must be at least between the cluster centers (decimal number)

Value

list with cluster assignments and the k-Means clustering model

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

prepareFeatureSet

Compiles a list of features from energy consumption data

Description

Returns a vector of feature names that can be calculated by methods in the *SmartMeterAnalytics* package by obtaining the feature set according

Usage

prepareFeatureSet(
    features.granularity = NA,
    features.w_adj = FALSE,
    features.anonymized = FALSE,
    features.categorical = FALSE,
    features.geo = "osm-v1",
    features.temperature = TRUE,
    features.weather = TRUE,
    features.neighborhood = FALSE
)
**Arguments**

*features.granularity*  
Character: The granularity of the input data, either "15-min" (only 15-min features), "30-min" (only 30-minute features), "all_30min_to_week" (all features on daily, weekly, hourly, ..., up to 30-min data), "all_15_week" (all up to 15-min data), "week" (only the consumption of one week as a single feature).

*features.w_adj*  
Boolean: are the features to be weather adjusted with DiD-Class (NOT IMPLEMENTED YET!)

*features.anonymized*  
Boolean: are anonymized geographic features used (NOT IMPLEMENTED YET!)

*features.categorical*  
Boolean: use categorical features additionally (if only numeric features are used)

*features.geo*  
Character: Version of the geographic feature set (either "none", "osm-v1", "osm-v2")

*features.temperature*  
Boolean, if features for the temperature should be included

*features.weather*  
Boolean, if other weather features should be included

*features.neighborhood*  
Boolean, if features for the neighborhood should be included

**Value**

Character vector

**Author(s)**

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

**References**


**remove_empty_features**  
*Removes variables with no necessary information from a data.frame*

**Description**

Removes variable names from a list of variables that contain only, or a large portion of, NA values or have zero bandwidth (if they are numeric) and returns the variable names.

**Usage**

```r
remove_empty_features(
  all.features,
  dataset,
  percentage_NA_allowed = NA,
  bandwidth = (\$.Machine$double.eps^0.5),
  verbose = FALSE
)
```

**Arguments**

- `all.features` a character vector with all column names of `dataset` that should be considered by the function
- `dataset` the dataset as a data.frame
- `percentage_NA_allowed` the percentage of missing values per vector that should be allowed without removing the feature. All features with NA values that are higher than this level are excluded.
- `bandwidth` The length of the interval that values of variable must exceed to be not removed. By default, half of `\$.Machine$double.eps` is used.
- `verbose` boolean if debug messages should be printed when a variable is removed from the list (uses `futile.logger` package)

**Details**

The function checks all given column names for the portion of NA values. If the number of NA of Inf exceeds `percentage_NA_allowed`, the column name is removed from the variable set. Besides, all numeric variables are checked if they have almost zero bandwidth, are removed.

**Value**

a vector of variable names that are not considered as empty

**Author(s)**

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>
replaceNAsFeatures

Replace NA values with a given ones

Description

Taks a data.frame and replaces all NA values with a certain value.

Usage

replaceNAsFeatures(indata, features, replacement = 0)

Arguments

indata a data.frame
features a vector of variable names (must be column names of indata that are to be used for NA-replacement
replacement the alternative value, NA values should be replaced with, zero by default

Value

the modified data.frame with replaced values

Author(s)

Konstantin Hopf <konstantin.hopf@uni-bamberg.de>

See Also

naInf_omit, remove_empty_features
**smote**  

**Synthetic minority oversampling (SMOTE)**

---

**Description**

Performs oversampling by creating new instances.

**Usage**

```r
smote(
  Variables,
  Classes,
  subset_use = NULL,
  k = 5,
  use_nearest = TRUE,
  proportions = 0.9,
  equalise_with_undersampling = FALSE,
  safe = FALSE
)
```

**Arguments**

- **Variables**
  - the `data.frame` of independent variables that should be used to create new instances
- **Classes**
  - the class labels in the prediction problem
- **subset_use**
  - a specific subset only is used for the oversampling. If `NULL`, everything is used.
- **k**
  - the number of neighbours for generation
- **use_nearest**
  - should only the nearest neighbours be used? (very slow)
- **proportions**
  - to which proportion (of the biggest class) should the classes be equalized
- **equalise_with_undersampling**
  - should additional undersampling be performed?
- **safe**
  - should a safe version of SMOTE be used?

**Details**

SMOTE is used to generate synthetic datapoints of a smaller class, for example to overcome the problem of imbalanced classes in classification.

**Value**

- a list containing new independent variables `data.frame` and new class labels

**Author(s)**

Ilya Kozlovskiy, Konstantin Hopf <konstantin.hopf@uni-bamberg.de>
Index

approx, 14

calc_features15_consumption, 2
calc_features30_consumption, 3
calc_features60_consumption, 5
calc_features_daily_multipleTS, 9
calc_features_weather, 10
calc_featuresco_consumption, 5
calc_featuresda_consumption, 6
calc_featureshtnt_consumption2, 7
calc_featuresnt_consumption, 8
cor, 9
data.frame, 14, 17–19
encode_p_val_stars, 11
features_all_subsets, 11
futile.logger, 17
getDay_ISO8601_week, 12
getAddress_US_week, 13
Inf, 14, 17
interpolate_missingReadings, 13
matrix, 14
NA, 14, 17, 18
naInf_omit, 14, 18
NULL, 19
occupancy_cluster, 15
prepareFeatureSet, 15
remove_empty_features, 14, 17, 18
replaceNAsFeatures, 14, 18, 18
smote, 19
spline, 14
stinterp, 14