Package ‘insurancerating’

April 11, 2020

Type Package

Title Analytic Insurance Rating Techniques

Version 0.6.0

VignetteBuilder knitr

Maintainer Martin Haringa <mtharinga@gmail.com>

Description Methods for insurance rating. It provides a data driven strategy for the construction of insurance tariff classes. This strategy is based on the work by Antonio and Valdez (2012) <doi:10.1007/s10182-011-0152-7>. insurancerating also provides recipes on how to easily perform univariate analyses on an insurance portfolio. In addition it adds functionality to include reference categories in the levels of the coefficients in the output of a generalized linear regression analysis.

License GPL (>= 2)


Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports classInt, data.table, dplyr, evtree, ggplot2, lubridate, magrittr, mgcv, patchwork, stringr, tidyr

Depends R (>= 3.3)

Suggests knitr, rmarkdown, scales, testthat

NeedsCompilation no

Author Martin Haringa [aut, cre]

Repository CRAN

Date/Publication 2020-04-10 23:20:08 UTC
R topics documented:

- autoplot.constructtariffclasses ........................................... 2
- autoplot.fitgam ................................................................. 4
- autoplot.riskfactor ............................................................ 5
- autoplot.univariate ............................................................ 6
- biggest_reference .............................................................. 8
- construct_tariff_classes .................................................... 9
- fisher ............................................................................. 10
- fit_gam ............................................................................ 11
- MTPL ............................................................................. 13
- MTPL2 ........................................................................... 14
- period_to_months ............................................................. 14
- rating_factors ................................................................. 15
- rating_factors1 ............................................................... 16
- univariate ..................................................................... 17
- univariate_all ............................................................... 18

Index 20

autoplot.constructtariffclasses

Automatically create a ggplot for objects obtained from construct_tariff_classes()

Description

Takes an object produced by construct_tariff_classes(), and plots the fitted GAM. In addition the constructed tariff classes are shown.

Usage

```r
## S3 method for class 'constructtariffclasses'
autoplot(  
  object,
  conf_int = FALSE,
  color_gam = "steelblue",
  show_observations = FALSE,
  color_splits = "grey50",
  size_points = 1,
  color_points = "black",
  rotate_labels = FALSE,
  remove_outliers = NULL,
  ...
)
```

**Arguments**

- **object**
  - constructtariffclasses object produced by `construct_tariff_classes`

- **conf_int**
  - determines whether 95% confidence intervals will be plotted. The default is `conf_int = FALSE`

- **color_gam**
  - a color can be specified either by name (e.g.: "red") or by hexadecimal code (e.g.: "#FF1234") (default is "steelblue")

- **show_observations**
  - add observed frequency/severity points for each level of the variable for which tariff classes are constructed

- **color_splits**
  - change the color of the splits in the graph ("grey50" is default)

- **size_points**
  - size for points (1 is default)

- **color_points**
  - change the color of the points in the graph ("black" is default)

- **rotate_labels**
  - rotate x-labels 45 degrees (this might be helpful for overlapping x-labels)

- **remove_outliers**
  - do not show observations above this number in the plot. This might be helpful for outliers.

- **...**
  - other plotting parameters to affect the plot

**Value**

- a ggplot object

**Author(s)**

- Martin Haringa

**Examples**

```r
## Not run:
library(ggplot2)
library(dplyr)
fit_gam(MTPL, nclaims = nclaims, x = age_policyholder, exposure = exposure) %>%
  construct_tariff_classes(.) %>%
  autoplot(., show_observations = TRUE)
## End(Not run)
```

```r
## Not run:
library(ggplot2)
library(dplyr)
fit_gam(MTPL, nclaims = nclaims, x = age_policyholder, exposure = exposure) %>%
  construct_tariff_classes(.) %>%
  autoplot(., show_observations = TRUE)
## End(Not run)
```
Description

Takes an object produced by fit_gam(), and plots the fitted GAM.

Usage

```r
## S3 method for class 'fitgam'
autoplot(
  object,
  conf_int = FALSE,
  color_gam = "steelblue",
  show_observations = FALSE,
  x_stepsize = NULL,
  size_points = 1,
  color_points = "black",
  rotate_labels = FALSE,
  remove_outliers = NULL,
  ...
)
```

Arguments

- **object**: fitgam object produced by fit_gam
- **conf_int**: determines whether 95% confidence intervals will be plotted. The default is conf_int = FALSE
- **color_gam**: a color can be specified either by name (e.g.: "red") or by hexadecimal code (e.g.: "#FF1234") (default is "steelblue")
- **show_observations**: add observed frequency/severity points for each level of the variable for which tariff classes are constructed
- **x_stepsize**: set step size for labels horizontal axis
- **size_points**: size for points (1 is default)
- **color_points**: change the color of the points in the graph ("black" is default)
- **rotate_labels**: rotate x-labels 45 degrees (this might be helpful for overlapping x-labels)
- **remove_outliers**: do not show observations above this number in the plot. This might be helpful for outliers.
- **...**: other plotting parameters to affect the plot

Value

a ggplot object
Author(s)

Martin Haringa

Examples

```r
## Not run:
library(ggplot2)
library(dplyr)
fit_gam(MTPL, nclaims = nclaims, x = age_policyholder, exposure = exposure) %>%
  autoplot(., show_observations = TRUE)
## End(Not run)
```


textbox{ autoplot.riskfactor Automatically create a ggplot for objects obtained from rating_factors() }

Description

Takes an object produced by univariate(), and plots the available input.

Usage

```r
## S3 method for class 'riskfactor'
autoplot(
  object,
  risk_factors = NULL,
  ncol = 1,
  labels = TRUE,
  dec.mark = ",",
  ylab = "rate",
  color = "dodgerblue",
  color_bg = "#E7B800",
  linetype = FALSE,
  ...
)
```

Arguments

object riskfactor object produced by rating_factors()

risk_factors character vector to define which factors are included. Defaults to all risk factors.

ncol number of columns in output (default is 1)

labels show labels with the exposure (default is TRUE)

dec.mark control the format of the decimal point, as well as the mark between intervals before the decimal point, choose either "," (default) or "."
autoplot.univariate

The autoplot.univariate function takes an object produced by `univariate()` and plots the available input. It automatically creates a ggplot for objects obtained from `univariate()`. The function accepts various parameters to modify the plot:

- `ylab`: modify label for the y-axis
- `color`: change the color of the points and line ("dodgerblue" is default)
- `color_bg`: change the color of the histogram ("#E7B800" is default)
- `linetype`: use different linetypes (default is FALSE)
- `...`: other plotting parameters to affect the plot

Value

- a ggplot2 object

Examples

```r
library(dplyr)
df <- MTP2 %>%
    mutate_at(vars(area), as.factor) %>%
    mutate_at(vars(area), ~biggest_reference(., exposure))

mod1 <- glm(nclaims ~ area + premium, offset = log(exposure), family = poisson(), data = df)
mod2 <- glm(nclaims ~ area, offset = log(exposure), family = poisson(), data = df)

x <- rating_factors(mod1, mod2, model_data = df, exposure = exposure)

autoplot(x)
```

Description

Takes an object produced by `univariate()`, and plots the available input.

Usage

```r
## S3 method for class 'univariate'
autoplot(
  object,
  show_plots = 1:9,
  ncol = 1,
  background = TRUE,
  labels = TRUE,
  sort = FALSE,
  sort_manual = NULL,
  dec.mark = ",",
  color = "dodgerblue",
  color_bg = "#E7B800",
  label_width = 10,
  coord_flip = FALSE,
  ...
)
```
Arguments

object  univariate object produced by univariate()
show_plots  numeric vector of plots to be shown (default is c(1,2,3,4,5,6,7,8,9)), there are nine available plots:
  • 1. frequency (i.e. number of claims / exposure)
  • 2. average severity (i.e. severity / number of claims)
  • 3. risk premium (i.e. severity / exposure)
  • 4. loss ratio (i.e. severity / premium)
  • 5. average premium (i.e. premium / exposure)
  • 6. exposure
  • 7. severity
  • 8. nclaims
  • 9. premium
ncol  number of columns in output (default is 1)
background  show exposure as a background histogram (default is TRUE)
labels  show labels with the exposure (default is TRUE)
sort  sort (or order) risk factor into descending order by exposure (default is FALSE)
sort_manual  sort (or order) risk factor into own ordering; should be a character vector (default is NULL)
dec.mark  control the format of the decimal point, as well as the mark between intervals before the decimal point, choose either ",," (default) or "."
color  change the color of the points and line ("dodgerblue" is default)
color_bg  change the color of the histogram ("#E7B800" is default)
label_width  width of labels on the x-axis (10 is default)
coord_flip  flip cartesian coordinates so that horizontal becomes vertical, and vertical, horizontal (default is FALSE)
...  other plotting parameters to affect the plot

Value

a ggplot2 object

Examples

library(ggplot2)
x <- univariate(MTPL2, x = area, severity = amount, nclaims = nclaims, exposure = exposure)
autoplot(x)
autoplot(x, show_plots = c(6,1), background = FALSE, sort = TRUE)
biggest_reference  
Set reference group to the group with largest exposure

Description
This function specifies the first level of a factor to the level with the largest exposure. Levels of factors are sorted using an alphabetic ordering. If the factor is used in a regression context, then the first level will be the reference. For insurance applications it is common to specify the reference level to the level with the largest exposure.

Usage
biggest_reference(x, weight)

Arguments
- x: an unordered factor
- weight: a vector containing weights (e.g. exposure). Should be numeric.

Value
a factor of the same length as x

Author(s)
Martin Haringa

References

Examples
```r
## Not run:
library(dplyr)
df <- chickwts %>%
mutate_if(is.character, as.factor) %>%
mutate_if(is.factor, list(~biggest_reference(., weight)))

## End(Not run)
```
**construct_tariff_classes**

*Construct insurance tariff classes*

---

**Description**

Constructs insurance tariff classes to fitgam objects produced by `fit_gam`. The goal is to bin the continuous risk factors such that categorical risk factors result which capture the effect of the covariate on the response in an accurate way, while being easy to use in a generalized linear model (GLM).

**Usage**

```r
calculate_tariff_classes(
  object,
  alpha = 0,
  niterations = 10000,
  ntrees = 200,
  seed = 1
)
```

**Arguments**

- **object**: fitgam object produced by `fit_gam`
- **alpha**: complexity parameter. The complexity parameter (alpha) is used to control the number of tariff classes. Higher values for alpha render less tariff classes. (alpha = 0 is default).
- **niterations**: in case the run does not converge, it terminates after a specified number of iterations defined by niterations.
- **ntrees**: the number of trees in the population.
- **seed**: an numeric seed to initialize the random number generator (for reproducibility).

**Details**

Evolutionary trees are used as a technique to bin the fitgam object produced by `fit_gam` into risk homogeneous categories. This method is based on the work by Henckaerts et al. (2018). See Grubinger et al. (2014) for more details on the various parameters that control aspects of the evtree fit.

**Value**

A list with components

- **prediction**: data frame with predicted values
- **x**: name of continuous risk factor for which tariff classes are constructed
- **model**: either 'frequency', 'severity' or 'burning'
The function provides an interface to finding class intervals for continuous numerical variables, for example for choosing colours for plotting maps.

Usage

fisher(vec, n = 7, diglab = 2)
Arguments

vec  
a continuous numerical variable
n  
number of classes required (n = 7 is default)
diglab  
number of digits (n = 2 is default)

Details

The "fisher" style uses the algorithm proposed by W. D. Fisher (1958) and discussed by Slocum et al. (2005) as the Fisher-Jenks algorithm. This function is adopted from the classInt package.

Value

Vector with clustering

Author(s)

Martin Haringa

References


Description

Fits a generalized additive model (GAM) to continuous risk factors in one of the following three types of models: the number of reported claims (claim frequency), the severity of reported claims (claim severity) or the burning cost (i.e. risk premium or pure premium).

Usage

```r
fit_gam(  
data,  
nclaims,  
x,  
exposure,  
amount = NULL,  
pure_premium = NULL,  
model = "frequency",  
round_x = NULL  
)
```
Arguments

data: data.frame of an insurance portfolio
nclaims: column in data with number of claims
x: column in data with continuous risk factor
exposure: column in data with exposure
amount: column in data with claim amount
pure_premium: column in data with pure premium
model: choose either 'frequency', 'severity' or 'burning' (model = 'frequency' is default). See details section.
round_x: round elements in column x to multiple of round_x. This gives a speed enhancement for data containing many levels for x.

Details

The 'frequency' specification uses a Poisson GAM for fitting the number of claims. The logarithm of the exposure is included as an offset, such that the expected number of claims is proportional to the exposure.

The 'severity' specification uses a lognormal GAM for fitting the average cost of a claim. The average cost of a claim is defined as the ratio of the claim amount and the number of claims. The number of claims is included as a weight.

The 'burning' specification uses a lognormal GAM for fitting the pure premium of a claim. The pure premium is obtained by multiplying the estimated frequency and the estimated severity of claims. The word burning cost is used here as equivalent of risk premium and pure premium.

Value

A list with components

prediction: data frame with predicted values
x: name of continuous risk factor
model: either 'frequency', 'severity' or 'burning'
data: data frame with predicted values and observed values
x_obs: observations for continuous risk factor

Author(s)

Martin Haringa

References


Examples

```r
fit_gam(MTPL, nclaims = nclaims, x = age_policyholder, exposure = exposure)
```

---

| MTPL | Ages of 32,731 policyholders in a Motor Third Party Liability (MTPL) portfolio. |

Description

A dataset containing the age, number of claims, and exposure of almost 33,000 policyholders

Usage

MTPL

Format

A data frame with 32,731 rows and 4 variables:

- `age_policyholder` age of policyholder, in years.
- `nclaims` number of claims.
- `exposure` exposure, for example, if a vehicle is insured as of July 1 for a certain year, then during that year, this would represent an exposure of 0.5 to the insurance company.
- `amount` claim amount in Euros.

Author(s)

Martin Haringa

Source

The data is derived from the portfolio of a large Dutch motor insurance company.
MTPL2

Characteristics of 3,000 policyholders in a Motor Third Party Liability (MTPL) portfolio.

Description
A dataset containing the area, number of claims, exposure, claim amount, exposure, and premium of 3,000 policyholders

Usage
MTPL2

Format
A data frame with 3,000 rows and 6 variables:

- customer_id: customer id
- area: region where customer lives
- nclaims: number of claims
- amount: claim amount (severity)
- exposure: exposure
- premium: earned premium

Author(s)
Martin Haringa

Source
The data is derived from the portfolio of a large Dutch motor insurance company.

period_to_months

Split period to months

Description
The function splits rows with a time period longer than one month to multiple rows with a time period of exactly one month each. Values in numeric columns (e.g., exposure or premium) are divided over the months proportionately.

Usage
period_to_months(df, begin, end, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>data.frame</td>
</tr>
<tr>
<td>begin</td>
<td>column in df with begin dates</td>
</tr>
<tr>
<td>end</td>
<td>column in df with end dates</td>
</tr>
<tr>
<td>...</td>
<td>numeric columns in df to split</td>
</tr>
</tbody>
</table>

Details

In insurance portfolios it is common that rows relate to periods longer than one month. This is for example problematic in case exposures per month are desired.

Since insurance premiums are constant over the months, and do not depend on the number of days per month, the function assumes that each month has the same number of days (i.e. 30).

Value
data.frame with same columns as in df, and one extra column called id

Author(s)

Martin Haringa

Examples

```r
library(lubridate)
portfolio <- data.frame(
  begin1 = ymd(c("2014-01-01", "2014-01-01")),
  end = ymd(c("2014-03-14", "2014-05-10")),
  termination = ymd(c("2014-03-14", "2014-05-10")),
  exposure = c(0.2025, 0.3583),
  premium = c(125, 150))
period_to_months(portfolio, begin1, end, premium, exposure)
```

rating_factors

Include reference group in regression output

Description

This extracts coefficients in terms of the original levels of the coefficients rather than the coded variables.

Usage

```r
rating_factors(..., model_data = NULL, exposure = NULL, exponentiate = TRUE)
```
Arguments

... glm object(s) produced by glm()
model_data data.frame used to create glm object(s)
exposure column in model_data with exposure
exponentiate Logical indicating whether or not to exponentiate the coefficient estimates. Defaults to TRUE.

Details

A fitted linear model has coefficients for the contrasts of the factor terms, usually one less in number than the number of levels. This function re-expresses the coefficients in the original coding. This function is adopted from dummy.coef(). Our adoption prints a data.frame as output.

Value

data.frame

Author(s)

Martin Haringa

Examples

library(dplyr)
df <- MTPL2 %>%
  mutate_at(vars(area), as.factor) %>%
  mutate_at(vars(area), ~biggest_reference(., exposure))

mod1 <- glm(nclaims ~ area + premium, offset = log(exposure), family = poisson(), data = df)
mod2 <- glm(nclaims ~ area, offset = log(exposure), family = poisson(), data = df)

rating_factors(mod1, mod2, model_data = df, exposure = exposure)

Description

This extracts coefficients in terms of the original levels of the coefficients rather than the coded variables.
Usage

rating_factors1(
  model,
  model_data = NULL,
  exposure = NULL,
  colname = "estimate",
  exponentiate = TRUE
)

Arguments

model glm object produced by glm()
model_data data.frame used to create glm object
exposure column in model_data with exposure
colname name of column with estimates. Defaults to "estimate".
exponentiate Logical indicating whether or not to exponentiate the coefficient estimates. Defaults to TRUE.

univariate Univariate analysis for discrete risk factors

Description

Univariate analysis for discrete risk factors in an insurance portfolio. The following summary statistics are calculated:

- frequency (i.e. number of claims / exposure)
- average severity (i.e. severity / number of claims)
- risk premium (i.e. severity / exposure)
- loss ratio (i.e. severity / premium)
- average premium (i.e. premium / exposure)

If input arguments are not specified, the summary statistics related to these arguments are ignored.

Usage

univariate(
  df,
  x,
  severity = NULL,
  nclaims = NULL,
  exposure = NULL,
  premium = NULL
)
Arguments

- **df**: data.frame with insurance portfolio
- **x**: column in df with risk factor
- **severity**: column in df with severity (default is NULL)
- **nclaims**: column in df with number of claims (default is NULL)
- **exposure**: column in df with exposure (default is NULL)
- **premium**: column in df with premium (default is NULL)

Value

An list of class `univ_all` with components

- **df**: data frame
- **xvar**: name of column in df with risk factor
- **severity**: name of column in df with severity
- **nclaims**: name of column in df with number of claims
- **exposure**: name of column in df with exposure
- **premium**: name of column in df with premium

Examples

```r
univariate(MTPL2, x = area, severity = amount, nclaims = nclaims,
           exposure = exposure, premium = premium)
```

# The summary statistics related to premium are not calculated
univariate(MTPL2, x = area, severity = amount, nclaims = nclaims, exposure = exposure)

---

### `univariate_all`

**Univariate analysis for discrete risk factors**

Description

Univariate analysis for discrete risk factors in an insurance portfolio. The following summary statistics are calculated:

- frequency (i.e. number of claims / exposure)
- average severity (i.e. severity / number of claims)
- risk premium (i.e. severity / exposure)
- loss ratio (i.e. severity / premium)
- average premium (i.e. premium / exposure)

If input arguments are not specified, the summary statistics related to these arguments are ignored.
Usage

univariate_all(
  df,
  x,
  severity = NULL,
  nclaims = NULL,
  exposure = NULL,
  premium = NULL
)

Arguments

df  data.frame with insurance portfolio
x   column in df with risk factor
severity   column in df with severity (default is NULL)
nclaims   column in df with number of claims (default is NULL)
exposure   column in df with exposure (default is NULL)
premium   column in df with premium (default is NULL)

Value

An list of class univ_all with components

df  data frame
xvar  name of column in df with risk factor
severity  name of column in df with severity
nclaims  name of column in df with number of claims
exposure  name of column in df with exposure
premium  name of column in df with premium
Index

*Topic **datasets**
  - MTPL, 13
  - MTPL2, 14

autoplot.constructtariffclasses, 2
autoplot.fitgam, 4
autoplot.riskfactor, 5
autoplot.univariate, 6

biggest_reference, 8
construct_tariff_classes, 9

fisher, 10
fit_gam, 11

MTPL, 13
MTPL2, 14

period_to_months, 14

rating_factors, 15
rating_factors1, 16

univariate, 17
univariate_all, 18