

# Package ‘radiant.model’

July 31, 2016

**Type** Package

**Title** Model estimation and evaluation menu for Radiant. Builds on the radiant.data package

**Version** 0.5.7

**Date** 2016-7-28

**Description** Modeling menu for Radiant.

**Depends** R (>= 3.3.0),  
radiant.data (>= 0.5.8)

**Imports** radiant.basics (>= 0.5.7),  
shiny (>= 0.13.2.9003),  
nnet (>= 7.3.11),  
NeuralNetTools (>= 1.4.0),  
sandwich (>= 2.3.4),  
car (>= 2.1.1),  
ggplot2 (>= 2.0.0),  
gridExtra (>= 2.0.0),  
data.tree (>= 0.1.9),  
yaml (>= 2.1.13),  
stringr (>= 1.0),  
pryr (>= 0.1.2),  
lubridate (>= 1.5.0),  
tidyr (>= 0.4.1),  
dplyr (>= 0.5),  
magrittr (>= 1.5),  
DiagrammeR (>= 0.8.4),  
import (>= 1.1.0),  
methods

**Suggests** testthat (>= 1.0.0),  
covr (>= 1.2.0)

**URL** <https://github.com/radiant-rstats/radiant.model>

**BugReports** <https://github.com/radiant-rstats/radiant.model/issues>

**License** AGPL-3 | file LICENSE

**LazyData** true

**RoxygenNote** 5.0.1

**R topics documented:**

ann . . . . .	3
auc . . . . .	4
catalog . . . . .	5
confint_robust . . . . .	5
confusion . . . . .	6
crs . . . . .	7
direct_marketing . . . . .	7
dtree . . . . .	8
dtree_parser . . . . .	8
dvd . . . . .	9
evalbin . . . . .	9
evalreg . . . . .	10
find_max . . . . .	11
find_min . . . . .	11
houseprices . . . . .	12
ideal . . . . .	12
logistic . . . . .	13
plot.ann . . . . .	14
plot.confusion . . . . .	14
plot.crs . . . . .	15
plot.dtree . . . . .	16
plot.evalbin . . . . .	16
plot.evalreg . . . . .	17
plot.logistic . . . . .	18
plot.model.predict . . . . .	19
plot.regress . . . . .	20
plot.repeater . . . . .	21
plot.simulater . . . . .	21
predict.ann . . . . .	22
predict.logistic . . . . .	23
predict.model . . . . .	24
predict.regress . . . . .	25
print.ann.predict . . . . .	26
print.logistic.predict . . . . .	26
print.model.predict . . . . .	27
print.regress.predict . . . . .	27
radiant.model . . . . .	28
radiant.model-deprecated . . . . .	28
regress . . . . .	28
render.DiagrammeR . . . . .	29
repeater . . . . .	30
scaledf . . . . .	31
sdw . . . . .	31
sensitivity . . . . .	32
sensitivity.dtree . . . . .	32
simulater . . . . .	33
sim_cleaner . . . . .	34
sim_splitter . . . . .	35
sim_summary . . . . .	35
store.model . . . . .	36

store.model.predict . . . . .	36
store_ann . . . . .	37
store_crs . . . . .	37
store_glm . . . . .	38
store_reg . . . . .	38
summary.ann . . . . .	39
summary.crs . . . . .	39
summary.dtree . . . . .	40
summary.evalbin . . . . .	40
summary.evalreg . . . . .	41
summary.logistic . . . . .	42
summary.regress . . . . .	43
summary.repeater . . . . .	44
summary.simulator . . . . .	44
test_specs . . . . .	45
var_check . . . . .	45
<b>Index</b>	<b>47</b>

---

ann	<i>Artificial Neural Networks</i>
-----	-----------------------------------

---

## Description

Artificial Neural Networks

## Usage

```
ann(dataset, rvar, evar, type = "classification", lev = "", size = 1,
     decay = 0.5, wts = "None", seed = NA, check = "standardize",
     data_filter = "")
```

## Arguments

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
rvar	The response variable in the logit (probit) model
evar	Explanatory variables in the model
type	Model type (i.e., "classification" or "regression")
lev	The level in the response variable defined as <code>_success_</code>
size	Number of units (nodes) in the hidden layer
decay	Parameter decay
wts	Weights to use in estimation
seed	Random seed to use as the starting point
check	Optional estimation parameters ("standardize" is the default)
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/ann.html> for an example in Radiant

**Value**

A list with all variables defined in ann as an object of class ann

**See Also**

[summary.ann](#) to summarize results

[plot.ann](#) to plot results

[predict.ann](#) for prediction

**Examples**

```
result <- ann("titanic", "survived", c("pclass","sex"), lev = "Yes")
result <- ann("titanic", "survived", c("pclass","sex"))
result <- ann("diamonds", "price", c("carat","clarity"), type = "regression")
```

---

auc	<i>Area Under the Curve (AUC)</i>
-----	-----------------------------------

---

**Description**

Area Under the Curve (AUC)

**Usage**

```
auc(pred, rvar, lev)
```

**Arguments**

pred	Prediction or predictor
rvar	Response variable
lev	The level in the response variable defined as <code>_success_</code>

**Details**

See <http://radiant-rstats.github.io/docs/model/evalbin.html> for an example in Radiant

**Value**

AUC statistic

**See Also**

[evalbin](#) to calculate results

[summary.evalbin](#) to summarize results

[plot.evalbin](#) to plot results

**Examples**

```
auc(mtcars$mpg, mtcars$vs, 1)
```

---

catalog	<i>Catalog sales for men's and women's apparel</i>
---------	--

---

**Description**

Catalog sales for men's and women's apparel

**Usage**

```
data(catalog)
```

**Format**

A data frame with 200 rows and 5 variables

**Details**

Description provided in attr(catalog,"description")

---

confint_robust	<i>Confidence interval for robust estimators</i>
----------------	--

---

**Description**

Confidence interval for robust estimators

**Usage**

```
confint_robust(object, parm, level = 0.95, vcov = NULL, ...)
```

**Arguments**

object	A fitted model object
parm	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered
level	The confidence level required
vcov	Covariance matrix generated by, e.g., sandwich::vcovHC
...	Additional argument(s) for methods

**Details**

Wrapper for confint.default with robust standard errors. See <http://stackoverflow.com/a/3820125/1974918>

---

 confusion

*Confusion matrix*


---

## Description

Confusion matrix

## Usage

```
confusion(dataset, pred, rvar, lev = "", margin = 1, cost = 1,
  train = "", method = "xtile", data_filter = "", ...)
```

## Arguments

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
pred	Predictions or predictors
rvar	Response variable
lev	The level in the response variable defined as <code>_success_</code>
margin	Margin on each customer purchase
cost	Cost for each connection (e.g., email or mailing)
train	Use data from training ("Training"), validation ("Validation"), both ("Both"), or all data ("All") to evaluate model evalbin
method	Use either <code>ntile</code> or <code>xtile</code> to split the data (default is <code>xtile</code> )
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/evalbin.html> for an example in Radiant

## Value

A list of results

## See Also

[summary.evalbin](#) to summarize results

[plot.evalbin](#) to plot results

## Examples

```
result <- evalbin("titanic", c("age", "fare"), "survived")
```

---

crs	<i>Collaborative Filtering</i>
-----	--------------------------------

---

**Description**

Collaborative Filtering

**Usage**

```
crs(dataset, id, prod, pred, rate, name = "pred", data_filter = "")
```

**Arguments**

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
id	String with name of the variable containing user ids
prod	String with name of the variable with product ids
pred	Products to predict for
rate	String with name of the variable with product ratings
name	Name for the prediction variable
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/crs.html> for an example in Radiant

**Value**

A data.frame with the original data and a new column with predicted ratings

---

direct_marketing	<i>Direct marketing data</i>
------------------	------------------------------

---

**Description**

Direct marketing data

**Usage**

```
data(direct_marketing)
```

**Format**

A data frame with 1,000 rows and 12 variables

**Details**

Description provided in `attr(direct_marketing,"description")`

---

dtree	Create a decision tree
-------	------------------------

---

**Description**

Create a decision tree

**Usage**

```
dtree(y1, opt = "max")
```

**Arguments**

y1	A yaml string or a list (e.g., from <code>yaml::yaml.load_file()</code> )
opt	Find the maximum ("max") or minimum ("min") value for each decision node

**Details**

See <http://radiant-rstats.github.io/docs/model/dtree.html> for an example in Radiant

**Value**

A list with the initial tree and the calculated tree

**See Also**

[summary.dtree](#) to summarize results  
[plot.dtree](#) to plot results

---

dtree_parser	Parse yaml input for dtree to provide (more) useful error messages
--------------	--

---

**Description**

Parse yaml input for dtree to provide (more) useful error messages

**Usage**

```
dtree_parser(y1)
```

**Arguments**

y1	A yaml string
----	---------------

**Details**

See <http://radiant-rstats.github.io/docs/model/dtree.html> for an example in Radiant

**Value**

An updated yaml string or a vector messages to return to the users



**See Also**

[dtree](#) to calculate tree

[summary.dtree](#) to summarize results

[plot.dtree](#) to plot results

---

dvd	<i>Data on DVD sales</i>
-----	--------------------------

---

**Description**

Data on DVD sales

**Usage**

```
data(dvd)
```

**Format**

A data frame with 20,000 rows and 4 variables

**Details**

Binary purchase response to coupon value. Description provided in `attr(dvd,"description")`

---

evalbin	<i>Model evalbin</i>
---------	----------------------

---

**Description**

Model evalbin

**Usage**

```
evalbin(dataset, pred, rvar, lev = "", qnt = 10, margin = 1, cost = 1,
        train = "", method = "xtile", data_filter = "")
```

**Arguments**

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
pred	Predictions or predictors
rvar	Response variable
lev	The level in the response variable defined as <code>_success_</code>
qnt	Number of bins to create
margin	Margin on each customer purchase
cost	Cost for each connection (e.g., email or mailing)
train	Use data from training ("Training"), validation ("Validation"), both ("Both"), or all data ("All") to evaluate model evalbin
method	Use either <code>ntile</code> or <code>xtile</code> to split the data (default is <code>xtile</code> )
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/evalbin.html> for an example in Radiant

**Value**

A list of results

**See Also**

`summary.evalbin` to summarize results

`plot.evalbin` to plot results

**Examples**

```
result <- evalbin("titanic", c("age", "fare"), "survived")
```

---

evalreg	<i>Model evalreg</i>
---------	----------------------

---

**Description**

Model evalreg

**Usage**

```
evalreg(dataset, pred, rvar, train = "", data_filter = "")
```

**Arguments**

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
pred	Predictions or predictors
rvar	Response variable
train	Use data from training ("Training"), validation ("Validation"), both ("Both"), or all data ("All") to evaluate model evalreg
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/evalreg.html> for an example in Radiant

**Value**

A list of results

**See Also**

`summary.evalreg` to summarize results

`plot.evalreg` to plot results

---

find_max	<i>Find maximum value of a vector</i>
----------	---------------------------------------

---

**Description**

Find maximum value of a vector

**Usage**

```
find_max(var, val = "")
```

**Arguments**

var	Variable to find the maximum for
val	Variable to find the value for at the maximum of var

**Value**

Value of val at the maximum of var

---

find_min	<i>Find minimum value of a vector</i>
----------	---------------------------------------

---

**Description**

Find minimum value of a vector

**Usage**

```
find_min(var, val = "")
```

**Arguments**

var	Variable to find the minimum for
val	Variable to find the value for at the maximum of var

**Value**

Value of val at the minimum of var

---

houseprices	<i>Houseprices</i>
-------------	--------------------

---

**Description**

Houseprices

**Usage**

```
data(houseprices)
```

**Format**

A data frame with 128 home sales and 6 variables

**Details**

Description provided in attr(houseprices,"description")

---

ideal	<i>Ideal data for linear regression</i>
-------	---

---

**Description**

Ideal data for linear regression

**Usage**

```
data(ideal)
```

**Format**

A data frame with 1,000 rows and 4 variables

**Details**

Description provided in attr(ideal,"description")

logistic

*Generalized linear models (GLM)***Description**

Generalized linear models (GLM)

**Usage**

```
logistic(dataset, rvar, evar, lev = "", int = "", wts = "None",
  check = "", data_filter = "")
```

**Arguments**

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
rvar	The response variable in the logit (probit) model
evar	Explanatory variables in the model
lev	The level in the response variable defined as <code>_success_</code>
int	Interaction term to include in the model
wts	Weights to use in estimation
check	Optional estimation parameters. "standardize" to output standardized coefficient estimates. "stepwise" to apply step-wise selection of variables
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/logistic.html> for an example in Radiant

**Value**

A list with all variables defined in `logistic` as an object of class `logistic`

**See Also**

`summary.logistic` to summarize the results  
`plot.logistic` to plot the results  
`predict.logistic` to generate predictions  
`plot.model.predict` to plot prediction output

**Examples**

```
result <- logistic("titanic", "survived", c("pclass","sex"), lev = "Yes")
result <- logistic("titanic", "survived", c("pclass","sex"))
```

---

plot.ann	<i>Plot method for the ann function</i>
----------	---

---

### Description

Plot method for the ann function

### Usage

```
## S3 method for class 'ann'  
plot(x, shiny = FALSE, ...)
```

### Arguments

x	Return value from <a href="#">ann</a>
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

### Details

See <http://radiant-rstats.github.io/docs/model/ann.html> for an example in Radiant

### See Also

[ann](#) to generate results  
[summary.ann](#) to summarize results  
[predict.ann](#) for prediction

### Examples

```
result <- ann("titanic", "survived", c("pclass","sex"), lev = "Yes")  
plot(result, plots = c("imp","net"))
```

---

plot.confusion	<i>Plot method for the confusion matrix</i>
----------------	---

---

### Description

Plot method for the confusion matrix

### Usage

```
## S3 method for class 'confusion'  
plot(x, scale_y = FALSE, shiny = FALSE, ...)
```

**Arguments**

x	Return value from <a href="#">evalreg</a>
scale_y	Free scale in faceted plot of the confusion matrix (TRUE or FALSE)
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/evalreg.html> for an example in Radiant

**See Also**

[evalreg](#) to generate results  
[summary.evalreg](#) to summarize results

---

plot.crs	<i>Plot method for the crs function</i>
----------	---

---

**Description**

Plot method for the crs function

**Usage**

```
## S3 method for class 'crs'  
plot(x, shiny = FALSE, ...)
```

**Arguments**

x	Return value from <a href="#">crs</a>
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/crs.html> for an example in Radiant

**See Also**

[crs](#) to generate results  
[summary.crs](#) to summarize results

---

plot.dtree	<i>Plot method for the dtree function</i>
------------	---

---

**Description**

Plot method for the dtree function

**Usage**

```
## S3 method for class 'dtree'
plot(x, symbol = "$", dec = 2, final = FALSE,
     shiny = FALSE, ...)
```

**Arguments**

x	Return value from <a href="#">dtree</a>
symbol	Monetary symbol to use (\$ is the default)
dec	Decimal places to round results to
final	If TRUE plot the decision tree solution, else the initial decision tree
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/dtree.html> for an example in Radiant

**See Also**

[dtree](#) to generate the result  
[summary.dtree](#) to summarize results

---

plot.evalbin	<i>Plot method for the evalbin function</i>
--------------	---

---

**Description**

Plot method for the evalbin function

**Usage**

```
## S3 method for class 'evalbin'
plot(x, plots = c("lift", "gains"), shiny = FALSE, ...)
```

**Arguments**

x	Return value from <a href="#">evalbin</a>
plots	Plots to return
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods



## Details

See <http://radiant-rstats.github.io/docs/model/evalbin.html> for an example in Radiant

## See Also

[evalbin](#) to generate results

[summary.evalbin](#) to summarize results

## Examples

```
evalbin("titanic", "age", "survived") %>% plot
evalbin("titanic", c("age", "fare"), "survived") %>% plot
evalbin("titanic", c("age", "fare"), "survived", method = "xtile") %>% plot
evalbin("titanic", c("age", "fare"), "survived") %>% summary
```

---

plot.evalreg

*Plot method for the evalreg function*

---

## Description

Plot method for the evalreg function

## Usage

```
## S3 method for class 'evalreg'
plot(x, shiny = FALSE, ...)
```

## Arguments

x	Return value from <a href="#">evalreg</a>
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/evalreg.html> for an example in Radiant

## See Also

[evalreg](#) to generate results

[summary.evalreg](#) to summarize results

---

plot.logistic	<i>Plot method for the logistic function</i>
---------------	--

---

## Description

Plot method for the logistic function

## Usage

```
## S3 method for class 'logistic'
plot(x, plots = "", conf_lev = 0.95, intercept = FALSE,
     shiny = FALSE, custom = FALSE, ...)
```

## Arguments

x	Return value from <a href="#">logistic</a>
plots	Plots to produce for the specified GLM model. Use "" to avoid showing any plots (default). "hist" shows histograms of all variables in the model. "scatter" shows scatter plots (or box plots for factors) for the response variable with each explanatory variable. "dashboard" is a series of four plots used to visually evaluate model. "coef" provides a coefficient plot
conf_lev	Confidence level to use for coefficient and odds confidence intervals (.95 is the default)
intercept	Include the intercept in the coefficient plot (TRUE or FALSE). FALSE is the default
shiny	Did the function call originate inside a shiny app
custom	Logical (TRUE, FALSE) to indicate if ggplot object (or list of ggplot objects) should be returned. This option can be used to customize plots (e.g., add a title, change x and y labels, etc.). See examples and <a href="http://docs.ggplot2.org/">http://docs.ggplot2.org/</a> for options.
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/logistic.html> for an example in Radiant

## See Also

[logistic](#) to generate results  
[plot.logistic](#) to plot results  
[predict.logistic](#) to generate predictions  
[plot.model.predict](#) to plot prediction output

## Examples

```
result <- logistic("titanic", "survived", c("pclass","sex"), lev = "Yes")
plot(result, plots = "coef")
```

---

plot.model.predict	<i>Plot method for model.predict functions</i>
--------------------	--

---

## Description

Plot method for model.predict functions

## Usage

```
## S3 method for class 'model.predict'
plot(x, xvar = "", facet_row = ".",
     facet_col = ".", color = "none", conf_lev = 0.95, ...)
```

## Arguments

x	Return value from predict functions (e.g., predict.regress)
xvar	Variable to display along the X-axis of the plot
facet_row	Create vertically arranged subplots for each level of the selected factor variable
facet_col	Create horizontally arranged subplots for each level of the selected factor variable
color	Adds color to a scatter plot to generate a heat map. For a line plot one line is created for each group and each is assigned a different colour
conf_lev	Confidence level to use for prediction intervals (.95 is the default)
...	further arguments passed to or from other methods

## See Also

[predict.regress](#) to generate predictions

[predict.logistic](#) to generate predictions

## Examples

```
regress("diamonds", "price", c("carat","clarity")) %>%
  predict(pred_cmd = "carat = 1:10") %>%
  plot(xvar = "carat")
logistic("titanic", "survived", c("pclass","sex","age"), lev = "Yes") %>%
  predict(pred_cmd="pclass=levels(pclass), sex=levels(sex), age=seq(0,100,20)") %>%
  plot(xvar = "age", color = "sex", facet_col = "pclass")
```

---

plot.regress

*Plot method for the regress function*


---

## Description

Plot method for the regress function

## Usage

```
## S3 method for class 'regress'
plot(x, plots = "", lines = "", conf_lev = 0.95,
     intercept = FALSE, shiny = FALSE, custom = FALSE, ...)
```

## Arguments

x	Return value from <a href="#">regress</a>
plots	Regression plots to produce for the specified regression model. Enter "" to avoid showing any plots (default). "hist" to show histograms of all variables in the model. "correlations" for a visual representation of the correlation matrix selected variables. "scatter" to show scatter plots (or box plots for factors) for the response variable with each explanatory variable. "dashboard" for a series of six plots that can be used to evaluate model fit visually. "resid_pred" to plot the explanatory variables against the model residuals. "coef" for a coefficient plot with adjustable confidence intervals. "leverage" to show leverage plots for each explanatory variable
lines	Optional lines to include in the select plot. "line" to include a line through a scatter plot. "loess" to include a polynomial regression fit line. To include both use c("line","loess")
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
intercept	Include the intercept in the coefficient plot (TRUE, FALSE). FALSE is the default
shiny	Did the function call originate inside a shiny app
custom	Logical (TRUE, FALSE) to indicate if ggplot object (or list of ggplot objects) should be returned. This option can be used to customize plots (e.g., add a title, change x and y labels, etc.). See examples and <a href="http://docs.ggplot2.org/">http://docs.ggplot2.org/</a> for options.
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

## See Also

[regress](#) to generate the results  
[summary.regress](#) to summarize results  
[predict.regress](#) to generate predictions

**Examples**

```

result <- regress("diamonds", "price", c("carat","clarity"))
plot(result, plots = "dashboard", lines = c("line","loess"))
plot(result, plots = "coef", conf_lev = .99, intercept = TRUE)
plot(result, plots = "hist")
plot(result, plots = "scatter", lines = c("line","loess"))
plot(result, plots = "correlations")
plot(result, plots = "resid_pred", lines = "line")

```

---

plot.repeater	<i>Plot repeated simulation</i>
---------------	---------------------------------

---

**Description**

Plot repeated simulation

**Usage**

```

## S3 method for class 'repeater'
plot(x, shiny = FALSE, ...)

```

**Arguments**

x	Return value from <a href="#">repeater</a>
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

---

plot.simulater	<i>Plot method for the simulater function</i>
----------------	---

---

**Description**

Plot method for the simulater function

**Usage**

```

## S3 method for class 'simulater'
plot(x, shiny = FALSE, ...)

```

**Arguments**

x	Return value from <a href="#">simulater</a>
shiny	Did the function call originate inside a shiny app
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/simulater> for an example in Radiant

**See Also**

[simulator](#) to generate the result  
[summary.simulator](#) to summarize results

**Examples**

```
result <- simulator(const = "cost 3", norm = "demand 2000 1000",
                   discrete = "price 5 8 .3 .7",
                   form = "profit = demand * (price - cost)")
plot(result)
```

---

predict.ann	<i>Predict method for the ann function</i>
-------------	--

---

**Description**

Predict method for the ann function

**Usage**

```
## S3 method for class 'ann'
predict(object, pred_data = "", pred_cmd = "",
        conf_lev = 0.95, se = FALSE, dec = 3, ...)
```

**Arguments**

object	Return value from <a href="#">ann</a>
pred_data	Provide the name of a dataframe to generate predictions (e.g., "titanic"). The dataset must contain all columns used in the estimation
pred_cmd	Generate predictions using a command. For example, 'pclass = levels(pclass)' would produce predictions for the different levels of factor 'pclass'. To add another variable use a ',' (e.g., 'pclass = levels(pclass), age = seq(0,100,20)')
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
se	Logical that indicates if prediction standard errors should be calculated (default = FALSE)
dec	Number of decimals to show
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/ann.html> for an example in Radiant

**See Also**

[ann](#) to generate the result  
[summary.ann](#) to summarize results

**Examples**

```
result <- logistic("titanic", "survived", c("pclass","sex"), lev = "Yes")
predict(result, pred_cmd = "pclass = levels(pclass)")
logistic("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_cmd = "sex = c('male','female')")
logistic("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_data = "titanic")
```

---

predict.logistic	<i>Predict method for the logistic function</i>
------------------	---

---

**Description**

Predict method for the logistic function

**Usage**

```
## S3 method for class 'logistic'
predict(object, pred_data = "", pred_cmd = "",
  conf_lev = 0.95, se = FALSE, dec = 3, ...)
```

**Arguments**

object	Return value from <a href="#">logistic</a>
pred_data	Provide the name of a dataframe to generate predictions (e.g., "titanic"). The dataset must contain all columns used in the estimation
pred_cmd	Generate predictions using a command. For example, 'pclass = levels(pclass)' would produce predictions for the different levels of factor 'pclass'. To add another variable use a ',' (e.g., 'pclass = levels(pclass), age = seq(0,100,20)')
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
se	Logical that indicates if prediction standard errors should be calculated (default = FALSE)
dec	Number of decimals to show
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/logistic.html> for an example in Radiant

**See Also**

[logistic](#) to generate the result  
[summary.logistic](#) to summarize results  
[plot.logistic](#) to plot results  
[plot.model.predict](#) to plot prediction output

## Examples

```
result <- logistic("titanic", "survived", c("pclass","sex"), lev = "Yes")
predict(result, pred_cmd = "pclass = levels(pclass)")
logistic("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_cmd = "sex = c('male','female')")
logistic("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_data = "titanic")
```

---

predict.model	<i>Predict method for model functions</i>
---------------	---

---

## Description

Predict method for model functions

## Usage

```
## S3 method for class 'model'
predict(object, pfun, mclass, pred_data = "", pred_cmd = "",
  conf_lev = 0.95, se = FALSE, dec = 3, ...)
```

## Arguments

object	Return value from <a href="#">regress</a>
pfun	Function to use for prediction
mclass	Model class to attach
pred_data	Name of the dataset to use for prediction
pred_cmd	Command used to generate data for prediction
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
se	Logical that indicates if prediction standard errors should be calculated (default = FALSE)
dec	Number of decimals to show
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant



---

predict.regress	<i>Predict method for the regress function</i>
-----------------	--

---

## Description

Predict method for the regress function

## Usage

```
## S3 method for class 'regress'
predict(object, pred_data = "", pred_cmd = "",
        conf_lev = 0.95, se = TRUE, dec = 3, ...)
```

## Arguments

object	Return value from <a href="#">regress</a>
pred_data	Name of the dataset to use for prediction
pred_cmd	Command used to generate data for prediction
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
se	Logical that indicates if prediction standard errors should be calculated (default = FALSE)
dec	Number of decimals to show
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

## See Also

[regress](#) to generate the result  
[summary.regress](#) to summarize results  
[plot.regress](#) to plot results

## Examples

```
result <- regress("diamonds", "price", c("carat","clarity"))
predict(result, pred_cmd = "carat = 1:10")
predict(result, pred_cmd = "clarity = levels(clarity)")
result <- regress("diamonds", "price", c("carat","clarity"), int = c("carat:clarity"))
dpred <- getdata("diamonds") %>% slice(1:10)
predict(result, pred_data = "dpred")
rm(dpred, envir = .GlobalEnv)
```

---

```
print.ann.predict
```

*Print method for predict.ann*

---

### Description

Print method for predict.ann

### Usage

```
## S3 method for class 'ann.predict'
print(x, ..., n = 10)
```

### Arguments

x	Return value from prediction method
...	further arguments passed to or from other methods
n	Number of lines of prediction results to print. Use -1 to print all lines

---

```
print.logistic.predict
```

*Print method for logistic.predict*

---

### Description

Print method for logistic.predict

### Usage

```
## S3 method for class 'logistic.predict'
print(x, ..., n = 10)
```

### Arguments

x	Return value from prediction method
...	further arguments passed to or from other methods
n	Number of lines of prediction results to print. Use -1 to print all lines

---

print.model.predict    *Print method for the model prediction*

---

### Description

Print method for the model prediction

### Usage

```
## S3 method for class 'model.predict'  
print(x, ..., n = 10, header = "", lev = "")
```

### Arguments

x	Return value from prediction method
...	further arguments passed to or from other methods
n	Number of lines of prediction results to print. Use -1 to print all lines
header	Header line
lev	The level in the response variable defined as <code>_success_</code> for classification models

---

print.regress.predict    *Print method for predict.regress*

---

### Description

Print method for predict.regress

### Usage

```
## S3 method for class 'regress.predict'  
print(x, ..., n = 10)
```

### Arguments

x	Return value from prediction method
...	further arguments passed to or from other methods
n	Number of lines of prediction results to print. Use -1 to print all lines

---

radiant.model	<i>radiant.model</i>
---------------	----------------------

---

**Description**

radiant.model  
Launch Radiant in the default browser

**Usage**

radiant.model()

**Details**

See <http://radiant-rstats.github.io/docs> for documentation and tutorials

---

radiant.model-deprecated	<i>Deprecated function(s) in the radiant.model package</i>
--------------------------	--

---

**Description**

These functions are provided for compatibility with previous versions of radiant. They will eventually be removed.

**Usage**

regression(...)

**Arguments**

... Parameters to be passed to the updated functions

**Details**

regression is now a synonym for [regress](#)  
 glm\_reg is now a synonym for [logistic](#)  
 performance is now a synonym for [evalbin](#)

---

regress	<i>Linear regression using OLS</i>
---------	------------------------------------

---

**Description**

Linear regression using OLS

**Usage**

```
regress(dataset, rvar, evar, int = "", check = "", data_filter = "")
```

**Arguments**

dataset	Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant
rvar	The response variable in the regression
evar	Explanatory variables in the regression
int	Interaction terms to include in the model
check	"standardize" to see standardized coefficient estimates. "stepwise" to apply step-wise selection of variables in estimation
data_filter	Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000")

**Details**

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

**Value**

A list of all variables used in the regress function as an object of class `regress`

**See Also**

[summary.regress](#) to summarize results  
[plot.regress](#) to plot results  
[predict.regress](#) to generate predictions

**Examples**

```
result <- regress("diamonds", "price", c("carat","clarity"))
result <- regress("diamonds", "price", c("carat","clarity"), check = "standardize")
```

---

render.DiagrammeR	<i>Method to render DiagrammeR plots</i>
-------------------	--

---

**Description**

Method to render DiagrammeR plots

**Usage**

```
## S3 method for class 'DiagrammeR'
render(object, ...)
```

**Arguments**

object	DiagrammeR plot
...	Additional arguments

---

repeater	<i>Repeat simulation</i>
----------	--------------------------

---

## Description

Repeat simulation

## Usage

```
repeater(nr = 12, vars = "", grid = "", sum_vars = "", byvar = "sim",
  fun = "sum_rm", form = "", seed = "", name = "", sim = "")
```

## Arguments

nr	Number times to repeat the simulation
vars	Variables to use in repeated simulation
grid	Expression to use in grid search for constants
sum_vars	(Numeric) variables to summaries
byvar	Variable(s) to group data by before summarizing
fun	Functions to use for summarizing
form	A string with the formula to apply to the summarized data
seed	To repeat a simulation with the same randomly generated values enter a number into Random seed input box.
name	To save the simulated data for further analysis specify a name in the Sim name input box. You can then investigate the simulated data by choosing the specified name from the Datasets dropdown in any of the other Data tabs.
sim	Return value from the simulator function

## Examples

```
result <- simulator(const = "var_cost 5;fixed_cost 1000;", norm = "E 0 100;",
  discrete = "price 6 8 .3 .7;",
  form = "demand = 1000 - 50*price + E;
  profit = demand*(price-var_cost) - fixed_cost;
  profit_small = profit < 100",
  seed = "1234")
repeater(nr = 12, vars = c("E","price"), sum_vars = "profit",
  byvar = "sim", form = "profit_365 = profit < 36500",
  seed = "1234", sim = result) %>% head
```

---

scaledf	<i>Center or standardize variables in a data frame</i>
---------	--

---

**Description**

Center or standardize variables in a data frame

**Usage**

```
scaledf(dat, center = TRUE, scale = TRUE, sf = 2, wts = NULL,  
        calc = TRUE)
```

**Arguments**

dat	Data frame
center	Center data (TRUE or FALSE)
scale	Scale data (TRUE or FALSE)
sf	Scaling factor (default is 2)
wts	Weights to use (default is NULL for no weights)
calc	Calculate mean and sd or use available attributes

**Value**

Scaled data frame

---

sdw	<i>Standard deviation of weighted sum of variables</i>
-----	--

---

**Description**

Standard deviation of weighted sum of variables

**Usage**

```
sdw(...)
```

**Arguments**

...	A matched number of weights and stocks
-----	--

**Value**

A vector of standard deviation estimates

---

sensitivity	<i>Method to evaluate sensitivity of an analysis</i>
-------------	--

---

**Description**

Method to evaluate sensitivity of an analysis

**Usage**

```
sensitivity(object, ...)
```

**Arguments**

object	Object of relevant class for which to evaluate sensitivity
...	Additional arguments

---

sensitivity.dtree	<i>Evaluate sensitivity of the decision tree</i>
-------------------	--

---

**Description**

Evaluate sensitivity of the decision tree

**Usage**

```
## S3 method for class 'dtree'
sensitivity(object, vars = NULL, decs = NULL,
  shiny = FALSE, ...)
```

**Arguments**

object	Return value from <a href="#">dtree</a>
vars	Variables to include in the sensitivity analysis
decs	Decisions to include in the sensitivity analysis
shiny	Did the function call originate inside a shiny app
...	Additional arguments

**Details**

See <http://radiant-rstats.github.io/docs/model/dtree.html> for an example in Radiant



simulater

*Simulate data for decision analysis***Description**

Simulate data for decision analysis

**Usage**

```
simulater(const = "", lnorm = "", norm = "", unif = "", discrete = "",
  binom = "", sequ = "", grid = "", data = "", form = "", seed = "",
  name = "", nr = 1000, dat = NULL)
```

**Arguments**

const	A string listing the constants to include in the analysis (e.g., "cost = 3; size = 4")
lnorm	A string listing the log-normally distributed random variables to include in the analysis (e.g., "demand 2000 1000" where the first number is the log-mean and the second is the log-standard deviation)
norm	A string listing the normally distributed random variables to include in the analysis (e.g., "demand 2000 1000" where the first number is the mean and the second is the standard deviation)
unif	A string listing the uniformly distributed random variables to include in the analysis (e.g., "demand 0 1" where the first number is the minimum value and the second is the maximum value)
discrete	A string listing the random variables with a discrete distribution to include in the analysis (e.g., "price 5 8 .3 .7" where the first set of numbers are the values and the second set the probabilities)
binom	A string listing the random variables with a binomial distribution to include in the analysis (e.g., "crash 100 .01") where the first number is the number of trials and the second is the probability of success)
sequ	A string listing the start and end for a sequence to include in the analysis (e.g., "trend 1 100 1"). The number of 'steps' is determined by the number of simulations.
grid	A string listing the start, end, and step for a set of sequences to include in the analysis (e.g., "trend 1 100 1"). The number of rows in the expanded will override the number of simulations
data	Name of a dataset to be used in the calculations
form	A string with the formula to evaluate (e.g., "profit = demand * (price - cost)")
seed	To repeat a simulation with the same randomly generated values enter a number into Random seed input box.
name	To save the simulated data for further analysis specify a name in the Sim name input box. You can then investigate the simulated data by choosing the specified name from the Datasets dropdown in any of the other Data tabs.
nr	Number of simulations
dat	Data list from previous simulation. Used by repeater function

**Details**

See <http://radiant-rstats.github.io/docs/model/simulater.html> for an example in Radiant

**Value**

A data.frame with the created variables

**See Also**

`summary.simulater` to summarize results

`plot.simulater` to plot results

**Examples**

```
result <- simulater(const = "cost 3", norm = "demand 2000 1000",
  discrete = "price 5 8 .3 .7",
  form = "profit = demand * (price - cost)")
```

---

sim\_cleaner

*Clean input command string*

---

**Description**

Clean input command string

**Usage**

```
sim_cleaner(x)
```

**Arguments**

x                      Input string

**Value**

Cleaned string

---

sim_splitter	<i>Split input command string</i>
--------------	-----------------------------------

---

**Description**

Split input command string

**Usage**

```
sim_splitter(x, symbol = " ")
```

**Arguments**

x	Input string
symbol	Symbol used to split the command string

**Value**

Split input command string

---

sim_summary	<i>Print simulation summary</i>
-------------	---------------------------------

---

**Description**

Print simulation summary

**Usage**

```
sim_summary(dat, dc = getclass(dat), fun = "", dec = 4)
```

**Arguments**

dat	Simulated data
dc	Variable classes
fun	Summary function to apply
dec	Number of decimals to show

---

store.model	<i>Store residuals from a model</i>
-------------	-------------------------------------

---

### Description

Store residuals from a model

### Usage

```
## S3 method for class 'model'
store(object, ..., name = "residuals")
```

### Arguments

object	Return value from a model function
...	Additional arguments
name	Variable name(s) assigned to predicted values

### Details

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

### Examples

```
regress(diamonds, rvar = "price", evar = c("carat", "cut")) %>%
  store %>% head
```

---

store.model.predict	<i>Store predicted values generated in model functions</i>
---------------------	--

---

### Description

Store predicted values generated in model functions

### Usage

```
## S3 method for class 'model.predict'
store(object, ..., data = attr(object, "pred_data"),
      name = "prediction")
```

### Arguments

object	Return value from model function
...	Additional arguments
data	Data or dataset name (e.g., data = mtcars or data = "mtcars")
name	Variable name(s) assigned to predicted values

**Details**

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

**Examples**

```
regress(diamonds, rvar = "price", evar = c("carat", "cut")) %>%
  predict(diamonds) %>%
  store %>% head
```

---

store_ann	<i>Deprecated function to store predictions from an ANN</i>
-----------	---

---

**Description**

Deprecated function to store predictions from an ANN

**Usage**

```
store_ann(object, data = object$dataset, name = paste0("predict_ann"))
```

**Arguments**

object	Return value from <code>predict.ann</code>
data	Dataset name
name	Variable name assigned to the residuals or predicted values

**Details**

Use `store.model.predict` or `store.model` instead

---

store_crs	<i>Store predicted values generated in the crs function</i>
-----------	---

---

**Description**

Store predicted values generated in the crs function

**Usage**

```
store_crs(pred, data, name = "pred_crs")
```

**Arguments**

pred	Return value from <code>predict.nnet</code>
data	Dataset name
name	Variable name assigned to the predicted values

**Details**

See <http://radiant-rstats.github.io/docs/model/crs.html> for an example in Radiant

---

store_glm	<i>Deprecated function to store logistic regression residuals and predictions</i>
-----------	---

---

**Description**

Deprecated function to store logistic regression residuals and predictions

**Usage**

```
store_glm(object, data = object$dataset, type = "residuals",
          name = paste0(type, "_logit"))
```

**Arguments**

object	Return value from <a href="#">logistic</a> or <a href="#">predict.logistic</a>
data	Dataset name
type	Residuals ("residuals") or predictions ("predictions"). For predictions the dataset name must be provided
name	Variable name assigned to the residuals or predicted values

**Details**

Use [store.model.predict](#) or [store.model](#) instead

---

store_reg	<i>Deprecated function to store regression residuals and predictions</i>
-----------	--

---

**Description**

Deprecated function to store regression residuals and predictions

**Usage**

```
store_reg(object, data = object$dataset, type = "residuals",
          name = paste0(type, "_reg"))
```

**Arguments**

object	Return value from <a href="#">regress</a> or <a href="#">predict.regress</a>
data	Dataset name
type	Residuals ("residuals") or predictions ("predictions"). For predictions the dataset name must be provided
name	Variable name assigned to the residuals or predicted values

**Details**

Use [store.model.predict](#) or [store.model](#) instead

---

summary.ann	<i>Summary method for the ann function</i>
-------------	--

---

**Description**

Summary method for the ann function

**Usage**

```
## S3 method for class 'ann'  
summary(object, ...)
```

**Arguments**

object	Return value from <a href="#">ann</a>
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/ann.html> for an example in Radiant

**See Also**

[ann](#) to generate esults  
[plot.ann](#) to plot results  
[predict.ann](#) for prediction

**Examples**

```
result <- ann("titanic", "survived", "pclass", lev = "Yes")  
summary(result)
```

---

summary.crs	<i>Summary method for Collaborative Filter</i>
-------------	--

---

**Description**

Summary method for Collaborative Filter

**Usage**

```
## S3 method for class 'crs'  
summary(object, ...)
```

**Arguments**

object	Return value from <a href="#">crs</a>
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/crs.html> for an example in Radiant

**See Also**

[crs](#) to generate the results

[plot.crs](#) to plot results

---

summary.dtree	<i>Summary method for the dtree function</i>
---------------	--

---

**Description**

Summary method for the dtree function

**Usage**

```
## S3 method for class 'dtree'
summary(object, ...)
```

**Arguments**

object	Return value from <a href="#">simulator</a>
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/dtree.html> for an example in Radiant

**See Also**

[dtree](#) to generate the results

[plot.dtree](#) to plot results

---

summary.evalbin	<i>Summary method for the evalbin function</i>
-----------------	--

---

**Description**

Summary method for the evalbin function

**Usage**

```
## S3 method for class 'evalbin'
summary(object, prn = TRUE, ...)
```



**Arguments**

object	Return value from <a href="#">evalbin</a>
prn	Print model evalbin results (default is TRUE)
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/evalbin.html> for an example in Radiant

**See Also**

[evalbin](#) to summarize results

[plot.evalbin](#) to plot results

**Examples**

```
evalbin("titanic", "age", "survived") %>% summary
evalbin("titanic", c("age", "fare"), "survived") %>% summary
```

---

summary.evalreg

*Summary method for the evalreg function*

---

**Description**

Summary method for the evalreg function

**Usage**

```
## S3 method for class 'evalreg'
summary(object, ...)
```

**Arguments**

object	Return value from <a href="#">evalreg</a>
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/evalreg.html> for an example in Radiant

**See Also**

[evalreg](#) to summarize results

[plot.evalreg](#) to plot results

summary.logistic

*Summary method for the logistic function***Description**

Summary method for the logistic function

**Usage**

```
## S3 method for class 'logistic'
summary(object, sum_check = "", conf_lev = 0.95,
        test_var = "", dec = 3, ...)
```

**Arguments**

object	Return value from <a href="#">logistic</a>
sum_check	Optional output. "vif" to show multicollinearity diagnostics. "confint" to show coefficient confidence interval estimates. "odds" to show odds ratios and confidence interval estimates.
conf_lev	Confidence level to use for coefficient and odds confidence intervals (.95 is the default)
test_var	Variables to evaluate in model comparison (i.e., a competing models Chi-squared test)
dec	Number of decimals to show
...	further arguments passed to or from other methods

**Details**

See <http://radiant-rstats.github.io/docs/model/logistic.html> for an example in Radiant

**See Also**

[logistic](#) to generate the results  
[plot.logistic](#) to plot the results  
[predict.logistic](#) to generate predictions  
[plot.model.predict](#) to plot prediction output

**Examples**

```
result <- logistic("titanic", "survived", "pclass", lev = "Yes")
summary(result, test_var = "pclass")
res <- logistic("titanic", "survived", c("pclass", "sex"), int="pclass:sex", lev="Yes")
summary(res, sum_check = c("vif", "confint", "odds"))
titanic %>% logistic("survived", c("pclass", "sex", "age"), lev = "Yes") %>% summary("vif")
```

---

summary.regress	<i>Summary method for the regress function</i>
-----------------	--

---

## Description

Summary method for the regress function

## Usage

```
## S3 method for class 'regress'
summary(object, sum_check = "", conf_lev = 0.95,
        test_var = "", dec = 3, ...)
```

## Arguments

object	Return value from <a href="#">regress</a>
sum_check	Optional output. "rsme" to show the root mean squared error and the standard deviation of the residuals. "sumsquares" to show the sum of squares table. "vif" to show multicollinearity diagnostics. "confint" to show coefficient confidence interval estimates.
conf_lev	Confidence level used to estimate confidence intervals (.95 is the default)
test_var	Variables to evaluate in model comparison (i.e., a competing models F-test)
dec	Number of decimals to show
...	further arguments passed to or from other methods

## Details

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

## See Also

[regress](#) to generate the results  
[plot.regress](#) to plot results  
[predict.regress](#) to generate predictions

## Examples

```
result <- regress("diamonds", "price", c("carat","clarity"))
summary(result, sum_check = c("rmse","sumsquares","vif","confint"), test_var = "clarity")
result <- regress("ideal", "y", c("x1","x2"))
summary(result, test_var = "x2")
ideal %>% regress("y", "x1:x3") %>% summary
```

---

summary.repeater	<i>Summarize repeated simulation</i>
------------------	--------------------------------------

---

### Description

Summarize repeated simulation

### Usage

```
## S3 method for class 'repeater'
summary(object, dec = 4, ...)
```

### Arguments

object	Return value from <a href="#">repeater</a>
dec	Number of decimals to show
...	further arguments passed to or from other methods

---

summary.simulater	<i>Summary method for the simulater function</i>
-------------------	--

---

### Description

Summary method for the simulater function

### Usage

```
## S3 method for class 'simulater'
summary(object, dec = 4, ...)
```

### Arguments

object	Return value from <a href="#">simulater</a>
dec	Number of decimals to show
...	further arguments passed to or from other methods

### Details

See <http://radiant-rstats.github.io/docs/model/simulater.html> for an example in Radiant

### See Also

[simulater](#) to generate the results  
[plot.simulater](#) to plot results

### Examples

```
result <- simulater(norm = "demand 2000 1000")
summary(result)
```

---

test_specs	<i>Add interaction terms to list of test variables if needed</i>
------------	--

---

**Description**

Add interaction terms to list of test variables if needed

**Usage**

```
test_specs(test_var, int)
```

**Arguments**

test_var	List of variables to use for testing for regress or logistic
int	Interaction terms specified

**Details**

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

**Value**

A vector of variables names to test

**Examples**

```
test_specs("a", c("a:b", "b:c"))
```

---

var_check	<i>Check if main effects for all interaction effects are included in the model If ':' is used to select a range _evar_ is updated</i>
-----------	---

---

**Description**

Check if main effects for all interaction effects are included in the model If ':' is used to select a range \_evar\_ is updated

**Usage**

```
var_check(ev, cn, intv = "")
```

**Arguments**

ev	List of explanatory variables provided to _regress_ or _logistic_
cn	Column names for all explanatory variables in _dat_
intv	Interaction terms specified

**Details**

See <http://radiant-rstats.github.io/docs/model/regress.html> for an example in Radiant

**Value**

'vars' is a vector of right-hand side variables, possibly with interactions, 'iv' is the list of explanatory variables, and intv are interaction terms

**Examples**

```
var_check("a:d", c("a", "b", "c", "d"))  
var_check(c("a", "b"), c("a", "b"), "a:c")
```

# Index

## \*Topic **datasets**

- catalog, 5
- direct\_marketing, 7
- dvd, 9
- houseprices, 12
- ideal, 12
- ann, 3, 14, 22, 39
- auc, 4
- catalog, 5
- confint\_robust, 5
- confusion, 6
- crs, 7, 15, 39, 40
- direct\_marketing, 7
- dtree, 8, 9, 16, 32, 40
- dtree\_parser, 8
- dvd, 9
- evalbin, 4, 9, 16, 17, 28, 41
- evalreg, 10, 15, 17, 41
- find\_max, 11
- find\_min, 11
- glm\_reg (radiant.model-deprecated), 28
- houseprices, 12
- ideal, 12
- logistic, 13, 18, 23, 28, 38, 42
- performance (radiant.model-deprecated), 28
- plot.ann, 4, 14, 39
- plot.confusion, 14
- plot.crs, 15, 40
- plot.dtree, 8, 9, 16, 40
- plot.evalbin, 4, 6, 10, 16, 41
- plot.evalreg, 10, 17, 41
- plot.logistic, 13, 18, 18, 23, 42
- plot.model.predict, 13, 18, 19, 23, 42
- plot.regress, 20, 25, 29, 43
- plot.repeater, 21
- plot.simulator, 21, 34, 44
- predict.ann, 4, 14, 22, 37, 39
- predict.logistic, 13, 18, 19, 23, 38, 42
- predict.model, 24
- predict.regress, 19, 20, 25, 29, 38, 43
- print.ann.predict, 26
- print.logistic.predict, 26
- print.model.predict, 27
- print.regress.predict, 27
- radiant.model, 28
- radiant.model-deprecated, 28
- radiant.model-deprecated-package (radiant.model-deprecated), 28
- radiant.model-package (radiant.model), 28
- regress, 20, 24, 25, 28, 28, 38, 43
- regression (radiant.model-deprecated), 28
- render.DiagrammeR, 29
- repeater, 21, 30, 44
- scaledf, 31
- sdw, 31
- sensitivity, 32
- sensitivity.dtree, 32
- sim\_cleaner, 34
- sim\_splitter, 35
- sim\_summary, 35
- simulator, 21, 22, 33, 40, 44
- store.model, 36, 37, 38
- store.model.predict, 36, 37, 38
- store\_ann, 37
- store\_crs, 37
- store\_glm, 38
- store\_reg, 38
- summary.ann, 4, 14, 22, 39
- summary.crs, 15, 39
- summary.dtree, 8, 9, 16, 40
- summary.evalbin, 4, 6, 10, 17, 40
- summary.evalreg, 10, 15, 17, 41
- summary.logistic, 13, 23, 42
- summary.regress, 20, 25, 29, 43

`summary.repeater`, [44](#)  
`summary.simulator`, [22](#), [34](#), [44](#)  
`test_specs`, [45](#)  
`var_check`, [45](#)