

# Package: regressorR (via r-universe)

October 31, 2024

**Title** Regression Data Analysis System

**Type** Package

**Version** 3.0.2

**Depends** R (>= 4.1)

**Imports** DT (>= 0.27), pls (>= 2.8-1), dplyr (>= 1.1.0), shiny (>= 1.7.4), golem (>= 0.3.5), rlang (>= 1.0.6), glmnet (>= 4.1-6), loadeR (>= 1.1.3), shinyjs (>= 2.1.0), traineR (>= 2.0.4), shinyAce (>= 0.4.2), echarts4r (>= 0.4.4), htmltools (>= 0.5.4), rpart.plot (>= 3.1.1), shinydashboard (>= 0.7.2), shinycustomloader (>= 0.9.0), shinydashboardPlus (>= 2.0.3)

**Description** Perform a supervised data analysis on a database through a 'shiny' graphical interface. It includes methods such as linear regression, penalized regression, k-nearest neighbors, decision trees, ada boosting, extreme gradient boosting, random forest, neural networks, deep learning and support vector machines.

**License** GPL (>=2)

**Encoding** UTF-8

**URL** <https://promidat.website/>, <https://github.com/PROMiDAT/regressorR>

**BugReports** <https://github.com/PROMiDAT/predictor/issues>

**RoxygenNote** 7.1.2

**Repository** <https://promidat.r-universe.dev>

**RemoteUrl** <https://github.com/PROMiDAT/regressorR>

**RemoteRef** HEAD

**RemoteSha** a111c08c36abd3a59b22da18769bb9cb68cdea0f

## Contents

app_server . . . . .	2
as_string_c . . . . .	3
boosting_importance_plot . . . . .	3

calibrate_boosting . . . . .	4
coef_lambda . . . . .	4
datos.disyuntivos . . . . .	5
disp_models . . . . .	5
dt_plot . . . . .	6
exe . . . . .	6
extract_code . . . . .	7
e_coeff_landa . . . . .	7
e_JS . . . . .	8
e_posib_lambda . . . . .	9
general_indices . . . . .	9
importance_plot_rf . . . . .	10
nn_plot . . . . .	11
pairs_power . . . . .	11
plot_pred_rd . . . . .	11
plot_real_prediction . . . . .	12
plot_RMSE . . . . .	13
plot_var_pred_rd . . . . .	13
rd_model . . . . .	14
rd_prediction . . . . .	15
rd_type . . . . .	15
rlr_model . . . . .	16
rlr_prediction . . . . .	16
rlr_type . . . . .	17
rl_coeff . . . . .	17
run_app . . . . .	18
summary_indices . . . . .	18
<b>Index</b>	<b>19</b>

---

app\_server

*The application server-side*


---

## Description

The application server-side

## Usage

```
app_server(input, output, session)
```

## Arguments

input, output, session

Internal parameters for shiny. DO NOT REMOVE.

---

`as_string_c`*as\_string\_c*

---

**Description**

creates a string representative of a vector

**Usage**

```
as_string_c(vect, quote = TRUE)
```

**Arguments**

`vect` a vector with values

`quote` a logical value. If TRUE, the values on the vector will be surrounded by quotes.

**Examples**

```
as_string_c(c("A", "B", "C"))
as_string_c(c(5, 6, 7))
as_string_c(c(5, 6, 7), quote = FALSE)
as_string_c(iris$Species)
```

---

`boosting_importance_plot`*boosting\_importance\_plot*

---

**Description**

generates the graph of variable importance.

**Usage**

```
boosting_importance_plot(
  model,
  titles = c("Importancia de Variables segun Influencia Relativa",
            "Influencia Relativa", "Variable")
)
```

**Arguments**

`model` boosting model(gbm).

`titles` Labels on the chart

---

calibrate_boosting	<i>calibrate_boosting</i>
--------------------	---------------------------

---

**Description**

helps to get the maximum of n.minobsinnode and bag.fraction values with which no error is generated in the model.

**Usage**

```
calibrate_boosting(data)
```

**Arguments**

data	the name of the learning data.
------	--------------------------------

**See Also**

[gbm](#)

**Examples**

```
calibrate_boosting(iris)
```

---

coef_lambda	<i>coef_lambda</i>
-------------	--------------------

---

**Description**

get penalized regression coefficients.

**Usage**

```
coef_lambda(data, variable.pred, model, log.lambda = NULL)
```

**Arguments**

data	dataframe
variable.pred	the name of the variable to be predicted.
model	a penalized regression model(cv.glmnet).
log.lambda	numerical. Logarithm of lambda in case you don't want to use the optimal lambda.

---

datos.disyuntivos      *Create disjunctive columns to a data.frame.*

---

**Description**

Create disjunctive columns to a data.frame.

**Usage**

```
datos.disyuntivos(data, var)
```

**Arguments**

data	a data.frame object.
var	the column name to apply disjunctive code.

**Value**

data.frame

**Author(s)**

Diego Jimenez <diego.jimenez@promidat.com>

**Examples**

```
datos.disyuntivos(iris, "Species")
```

---

disp\_models      *disp\_models*

---

**Description**

this function generates the call code of the scatter function.

**Usage**

```
disp_models(prediction, model_name, var_pred)
```

**Arguments**

prediction	the name of the prediction object.
model_name	the name of the model.
var_pred	the name of the variable to be predicted.

**Examples**

```
disp_models("prediction.knn", "KNN", "Species")
```

---

dt_plot	<i>dt_plot</i>
---------	----------------

---

**Description**

makes the graph of the tree.

**Usage**

```
dt_plot(model)
```

**Arguments**

model            a decision trees model(rpart).

---

exe	<i>exe</i>
-----	------------

---

**Description**

concat and execute a text in R.

**Usage**

```
exe(..., envir = parent.frame())
```

**Arguments**

...            one or more texts to be concatenated and executed.  
 envir        the environment in which expr is to be evaluated.

**Value**

the result of the execute.

**Examples**

```
exe("5+5")
exe("5", "+", "5")
exe("plot(iris$Species)")
```

---

extract_code	<i>extract_code</i>
--------------	---------------------

---

**Description**

gets the code of a function in text form.

**Usage**

```
extract_code(funcion, envir = parent.frame())
```

**Arguments**

funcion	the name of the function to be extracted.
envir	the environment in which expr is to be evaluated.

**Examples**

```
extract_code("cat")
extract_code("plot")

parse(text = extract_code("plot"))
```

---

e_coeff_landa	<i>e_coeff_landa</i>
---------------	----------------------

---

**Description**

Graph the coefficients and lambdas of a cv.glmnet model

**Usage**

```
e_coeff_landa(
  cv.glm,
  log.lambda = NULL,
  titles = c("Coeficientes", "Seleccionado", "Automatico")
)
```

**Arguments**

cv.glm	a cv.glmnet model.
log.lambda	number that specifies the logarithm of the selected lambda
titles	labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

**See Also**

[cv.glmnet](#)

---

e\_JS

*Eval character vectors to JS code*

---

**Description**

Eval character vectors to JS code

**Usage**

```
e_JS(...)
```

**Arguments**

... character vectors to evaluate

**Author(s)**

Joseline Quiros <joseline.quiros@promidat.com>

**Examples**

```
e_JS('5 * 3')
```



---

e_posib_lambda	<i>e_posib_lambda</i>
----------------	-----------------------

---

**Description**

Graph a cv.glmnet model

**Usage**

```
e_posib_lambda(  
  cv.glm,  
  log.lambda = NULL,  
  titles = c("Error Cuadratico Medio", "Curva Inferior", "Curva Superior",  
            "Seleccionado", "Automatico", "Coeficientes Distintos de Cero")  
)
```

**Arguments**

cv.glm	a cv.glmnet model.
log.lambda	number that specifies the logarithm of the selected lambda
titles	labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

**See Also**

[cv.glmnet](#)

---

general_indices	<i>general_indices</i>
-----------------	------------------------

---

**Description**

calculates indices to measure accuracy of a model.

**Usage**

```
general_indices(real, prediccion)
```

**Arguments**

real                the real values in training-testing.  
prediccion        the prediction values in training-testing.

**Value**

a list with the Correlation, Relative Error, Mean Absolute Error and Root Mean Square Error.

**Examples**

```
real <- rnorm(45)
prediction <- rnorm(45)
model <- "KNN"
general_indices(real, prediction)
```

---

importance\_plot\_rf     *importance\_plot\_rf*

---

**Description**

graphs the importance of variables for the random forest model according to the percentage increase in mean square error.

**Usage**

```
importance_plot_rf(
  model.rf,
  titles = c("Importancia de Variables Segun el Porcentaje de Incremento del MSE",
            "Aumento porcentual del error cuadratico medio", "Variable")
)
```

**Arguments**

model.rf            a random forest model.  
titles               labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

**See Also**

[randomForest](#)

---

nn_plot	<i>nn_plot</i>
---------	----------------

---

**Description**

graph of the neural network.

**Usage**

```
nn_plot(model)
```

**Arguments**

model	a neural network model(neuralnet)
-------	-----------------------------------

---

pairs_power	<i>pairs_power</i>
-------------	--------------------

---

**Description**

Generate a pair chart

**Usage**

```
pairs_power(data, decimals = 2)
```

**Arguments**

data	A DataFrame
decimals	Number of numbers after the decimal point.

---

plot_pred_rd	<i>plot_pred_rd</i>
--------------	---------------------

---

**Description**

graph of variance explained in the predictors according to components used.

**Usage**

```
plot_pred_rd(
  model,
  n.comp,
  titles = c("Varianza Explicada en Predictores", "Numero de Componentes",
            "Porcentaje de Varianza Explicada")
)
```

**Arguments**

model	a dimension reduction model.
n.comp	the optimum number of components.
titles	labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

---

`plot_real_prediction` *plot\_real\_prediction*

---

**Description**

scatter plot between the actual value of the variable to be predicted and the prediction of the model.

**Usage**

```
plot_real_prediction(
  real,
  prediction,
  model = "",
  titles = c("Predicciones vs Valores Reales", "Valor Real", "Prediccion")
)
```

**Arguments**

real	the real values in training-testing.
prediction	the prediction values in training-testing.
model	the name of the model of the scatter plot.
titles	Labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

---

`plot_RMSE`*plot\_RMSE*

---

**Description**

graph the root mean square error of cross validation according to components used.

**Usage**

```
plot_RMSE(  
  model,  
  n.comp,  
  titles = c("RMSE Segun Numero de Componentes", "Numero de Componente", "RMSE")  
)
```

**Arguments**

<code>model</code>	a dimension reduction model.
<code>n.comp</code>	the optimum number of components.
<code>titles</code>	labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

---

`plot_var_pred_rd`*plot\_var\_pred\_rd*

---

**Description**

graph of the variance explained in the variable to predict according to the components used.

**Usage**

```
plot_var_pred_rd(  
  model,  
  n.comp,  
  titles = c("Varianza Explicada en Variable a Predecir", "Numero de Componente",  
            "Porcentaje de Varianza Explicada")  
)
```

**Arguments**

<code>model</code>	a dimension reduction model.
<code>n.comp</code>	the optimum number of components.
<code>titles</code>	labels on the chart

**Value**

echarts4r plot

**Author(s)**

Ariel Arroyo <luis.ariel.arroyo@promidat.com>

---

<code>rd_model</code>	<i>rd_model</i>
-----------------------	-----------------

---

**Description**

generates a dimension reduction model.

**Usage**

```
rd_model(data, variable.pred, mode = 0, scale = TRUE)
```

**Arguments**

<code>data</code>	dataframe
<code>variable.pred</code>	the name of the variable to be predicted.
<code>mode</code>	the method of dimension reduction is defined as mode=1 is the MCP, and mode=0 the ACP.
<code>scale</code>	the scale parameter of the model.

**See Also**

[pca](#), [pls](#)

---

rd_prediction	<i>rd_prediction</i>
---------------	----------------------

---

**Description**

generates the prediction of a dimension reduction model.

**Usage**

```
rd_prediction(model, test.data, ncomp = NULL)
```

**Arguments**

model	dimension reduction model(pcr/plsr).
test.data	dataframe.
ncomp	a numerical value in case you don't want to use the optimum number of components.

---

rd_type	<i>rd_type</i>
---------	----------------

---

**Description**

returns the name of the method of dimension reduction.

**Usage**

```
rd_type(mode.rd = 0)
```

**Arguments**

mode.rd	the method of dimension reduction is defined as mode=1 is the MCP, and mode=0 the ACP.
---------	--

**See Also**

[pcr](#), [pls](#)

**Examples**

```
rd_type(1)  
rd_type(0)
```

---

`rlr_model`*rlr\_model*

---

**Description**

generates a penalized regression model.

**Usage**

```
rlr_model(data, variable.pred, alpha = 0, standardize = TRUE)
```

**Arguments**

<code>data</code>	dataframe
<code>variable.pred</code>	the name of the variable to be predicted.
<code>alpha</code>	the alpha parameter of the model.
<code>standardize</code>	the standardize parameter of the model.

**See Also**

[glmnet](#), [cv.glmnet](#)

---

`rlr_prediction`*rlr\_prediction*

---

**Description**

generates the prediction of the penalized regression model.

**Usage**

```
rlr_prediction(model, test.data, variable.pred, log.lambda = NULL)
```

**Arguments**

<code>model</code>	a penalized regression model( <a href="#">cv.glmnet</a> ).
<code>test.data</code>	dataframe.
<code>variable.pred</code>	the name of the variable to be predicted.
<code>log.lambda</code>	numerical. Logarithm of lambda in case you don't want to use the optimal lambda.



---

rlr_type	<i>rlr_type</i>
----------	-----------------

---

**Description**

returns the name of the penalty according to the alpha.

**Usage**

```
rlr_type(alpha_rlr = 0)
```

**Arguments**

alpha\_rlr      the penalty is defined as alpha=1 is the lasso penalty, and alpha=0 the ridge penalty.

**See Also**

[glmnet](#)

**Examples**

```
rlr_type(1)  
rlr_type(0)
```

---

r1_coeff	<i>rl_coeff</i>
----------	-----------------

---

**Description**

get the information of the coefficients of the linear regression model

**Usage**

```
r1_coeff(modelo)
```

**Arguments**

modelo          linear regression model

---

run_app	<i>Run the Shiny Application</i>
---------	----------------------------------

---

**Description**

Run the Shiny Application

**Usage**

```
run_app(...)
```

**Arguments**

... A series of options to be used inside the app.

---

summary_indices	<i>summary_indices</i>
-----------------	------------------------

---

**Description**

summarizes a variable by returning the minimum, first quartile, third quartile and maximum value.

**Usage**

```
summary_indices(data)
```

**Arguments**

data a numeric vector.

**Examples**

```
summary_indices(iris$Sepal.Length)
```

# Index

app\_server, 2  
as\_string\_c, 3

boosting\_importance\_plot, 3

calibrate\_boosting, 4  
coef\_lambda, 4  
cv.glmnet, 8, 9, 16

datos.disyuntivos, 5  
disp\_models, 5  
dt\_plot, 6

e\_coef\_lambda, 7  
e\_JS, 8  
e\_posib\_lambda, 9  
exe, 6  
extract\_code, 7

gbm, 4  
general\_indices, 9  
glmnet, 16, 17

importance\_plot\_rf, 10

nn\_plot, 11

pairs\_power, 11  
pcr, 14, 15  
plot\_pred\_rd, 11  
plot\_real\_prediction, 12  
plot\_RMSE, 13  
plot\_var\_pred\_rd, 13  
plsr, 14, 15

randomForest, 10  
rd\_model, 14  
rd\_prediction, 15  
rd\_type, 15  
rl\_coef, 17  
rlr\_model, 16  
rlr\_prediction, 16  
rlr\_type, 17  
run\_app, 18  
summary\_indices, 18