

Package ‘modelc’

July 23, 2025

Title A Linear Model to 'SQL' Compiler

Version 1.0.0.0

Description This is a cross-platform linear model to 'SQL' compiler. It generates 'SQL' from linear and generalized linear models. Its interface consists of a single function, `modelc()`, which takes the output of `lm()` or `glm()` functions (or any object which has the same signature) and outputs a 'SQL' character vector representing the predictions on the scale of the response variable as described in Dunn & Smith (2018) <[doi:10.1007/978-1-4419-0118-7](https://doi.org/10.1007/978-1-4419-0118-7)> and originating in Nelder & Wedderburn (1972) <[doi:10.2307/2344614](https://doi.org/10.2307/2344614)>. The resultant 'SQL' can be included in a 'SELECT' statement and returns output similar to that of the `glm.predict()` or `lm.predict()` predictions, assuming numeric types are represented in the database using sufficient precision. Currently log and identity link functions are supported.

License MIT + file LICENSE

URL <https://github.com/sparkfish/modelc>

BugReports <https://github.com/sparkfish/modelc/issues>

Encoding UTF-8

LazyData true

Suggests testthat (>= 2.1.0)

RoxygenNote 7.1.0

NeedsCompilation no

Author Sparkfish Analytics [cph],
Hugo Saavedra [aut, cre]

Maintainer Hugo Saavedra <analytics+hugo@sparkfish.com>

Repository CRAN

Date/Publication 2020-06-28 10:00:05 UTC

Contents

<code>apply_linkinverse</code>	2
<code>build_additive_term</code>	3

build_factor_case_statements	3
build_interaction_term	4
build_intercept	4
build_product	5
extract_level	5
extract_parameters	6
extract_parameter_coefficient	6
get_factor_name	7
has_parameter	7
is_factor	8
is_interaction	8
is_intercept	9
modelc	9
Index	11

apply_linkinverse	<i>Wrap the model SQL in the appropriate link function inverse to return scaled predictions</i>
-------------------	---

Description

Wrap the model SQL in the appropriate link function inverse to return scaled predictions

Usage

```
apply_linkinverse(model, sql)
```

Arguments

- model A list with the same signature as the output of lm or glm
- sql A character string representing the SQL to be wrapped in the link inverse

Value

A character string representing a SQL model formula

build_additive_term	<i>Get SQL representing a continuous term in the model with no interactions</i>
---------------------	---

Description

Get SQL representing a continuous term in the model with no interactions

Usage

```
build_additive_term(model, additive_term, first = FALSE)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
additive_term	A parameter name.
first	A logical flag signaling whether the term is the first term in the formula

Value

A SQL character string representing an additive term

build_factor_case_statements	<i>Build SQL CASE statements representing the factors in the model</i>
------------------------------	--

Description

Build SQL CASE statements representing the factors in the model

Usage

```
build_factor_case_statements(model, first = FALSE)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
first	A logical flag signaling whether the term is the first term in the formula

Value

A character string representing a SQL CASE statement

```
build_interaction_term
```

Build a SQL interaction term

Description

Build a SQL interaction term

Usage

```
build_interaction_term(model, interaction_term, first = FALSE)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
interaction_term	The raw interaction term (a character string) from the R model
first	A logical flag signaling whether the term is the first term in the formula

Value

A character string representing a SQL interaction term

build_intercept	<i>Get SQL representing the intercept term given the R model and parameter name</i>
-----------------	---

Description

Get SQL representing the intercept term given the R model and parameter name

Usage

```
build_intercept(model, parameter, first = FALSE)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
parameter	A parameter name.
first	A logical flag signaling whether the term is the first term in the formula

Value

A SQL character string representing the intercept term in the model

build_product	<i>Build a SQL product</i>
---------------	----------------------------

Description

Build a SQL product

Usage

```
build_product(lhs, rhs)
```

Arguments

lhs	A character string representing the left hand side of the multiplication
rhs	A character string representing the right hand side of the multiplication

Value

A character string representing a valid SQL product term

extract_level	<i>Extract the level from the factor name</i>
---------------	---

Description

Extract the level from the factor name

Usage

```
extract_level(parameter, factor)
```

Arguments

parameter	A parameter name
factor	A factor term

Value

A SQL string literal representing the factor level

extract_parameters	<i>Extract parameters from a linear model</i>
--------------------	---

Description

Extract parameters from a linear model

Usage

```
extract_parameters(model)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
-------	---

Value

A character vector of terms from a linear model

extract_parameter_coefficient	<i>Extract the coefficient of a model parameter</i>
-------------------------------	---

Description

Extract the coefficient of a model parameter

Usage

```
extract_parameter_coefficient(model, parameter)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
parameter	A character string corresponding to a model predictor

Value

A double corresponding to the coefficient, or 0 if the coefficient is missing

get_factor_name	<i>Extract the factor name from an R model</i>
-----------------	--

Description

Extract the factor name from an R model

Usage

```
get_factor_name(parameter, model)
```

Arguments

parameter	A parameter name.
model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>

Value

A character string representing the factor name

has_parameter	<i>Check if an R model contains a coefficient</i>
---------------	---

Description

Check if an R model contains a coefficient

Usage

```
has_parameter(model, parameter)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
parameter	A parameter name

Value

A logical representing whether a coefficient is present in the model

is_factor	<i>Detect if the given model term is a factor</i>
-----------	---

Description

Detect if the given model term is a factor

Usage

```
is_factor(parameter, model)
```

Arguments

parameter	A parameter name.
model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>

Value

A logical representing whether or not the term is a factor

is_interaction	<i>Detect if the given model term is an interaction</i>
----------------	---

Description

Detect if the given model term is an interaction

Usage

```
is_interaction(parameter)
```

Arguments

parameter	A parameter name.
-----------	-------------------

Value

A logical representing whether or not the term is an interaction

is_intercept	<i>Check if the given parameter is the intercept</i>
--------------	--

Description

Check if the given parameter is the intercept

Usage

```
is_intercept(parameter)
```

Arguments

parameter	A parameter name.
-----------	-------------------

Value

A logical representing whether the given parameter is the intercept

modelc	<i>Compile an R model to a valid TSQL formula</i>
--------	---

Description

Compile an R model to a valid TSQL formula

Usage

```
modelc(model, modify_scipen = TRUE)
```

Arguments

model	A list with the same signature as the output of <code>lm</code> or <code>glm</code>
modify_scipen	A boolean indicating whether to modify the "scipen" option to avoid generating invalid SQL

Value

A character string representing a SQL model formula

Examples

```
a <- 1:10
b <- 2*1:10
c <- as.factor(a)
df <- data.frame(a, b, c)
formula = b ~ a + c

# A vanilla linear model
linear_model <- lm(formula, data = df)
modelc::modelc(linear_model)

# A generalized linear model with gamma family distribution and log link function
gamma_loglink_model <- glm(formula, data = df, family=Gamma(link="log"))
modelc::modelc(gamma_loglink_model)

# A generalized linear model with gamma family distribution and identity link function
gamma_idlink_model <- glm(formula, data = df, family=Gamma(link="identity"))
modelc::modelc(gamma_idlink_model)
```

Index

`apply_linkinverse`, [2](#)

`build_additive_term`, [3](#)
`build_factor_case_statements`, [3](#)
`build_interaction_term`, [4](#)
`build_intercept`, [4](#)
`build_product`, [5](#)

`extract_level`, [5](#)
`extract_parameter_coefficient`, [6](#)
`extract_parameters`, [6](#)

`get_factor_name`, [7](#)

`has_parameter`, [7](#)

`is_factor`, [8](#)
`is_interaction`, [8](#)
`is_intercept`, [9](#)

`modelc`, [9](#)