

Package ‘MantaID’

October 18, 2022

Title A Machine-Learning Based Tool to Automate the Identification of Biological Database IDs

Version 1.0.2

Description The number of biological databases is growing rapidly, but different databases use different IDs to refer to the same biological entity. The inconsistency in IDs impedes the integration of various types of biological data. To resolve the problem, we developed 'MantaID', a data-driven, machine-learning based approach that automates identifying IDs on a large scale. The 'MantaID' model's prediction accuracy was proven to be 99%, and it correctly and effectively predicted 100,000 ID entries within two minutes. 'MantaID' supports the discovery and exploitation of ID patterns from large quantities of databases. (e.g., up to 542 biological databases). An easy-to-use freely available open-source software R package, a user-friendly web application, and APIs were also developed for 'MantaID' to improve applicability. To our knowledge, 'MantaID' is the first tool that enables an automatic, quick, accurate, and comprehensive identification of large quantities of IDs, and can therefore be used as a starting point to facilitate the complex assimilation and aggregation of biological data across diverse databases.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

Suggests

Depends R (>= 4.2.0)

Imports biomaRt, caret, data.table, dplyr, ggplot2, keras, magrittr, mlr3, purrr, reshape2, scutr, stringr, tibble, tidyr, tidyselect, mlr3tuning, paradox, RColorBrewer, ggcorrplot

LazyData true

NeedsCompilation no

Author Zeng Zhengpeng [aut, cre, ctb]
(<<https://orcid.org/0000-0001-7919-209X>>),
Mao Longfei [aut] (<<https://orcid.org/0000-0003-0759-0501>>),
Yu Feng [aut] (<<https://orcid.org/0000-0002-5221-281X>>)

Maintainer Zeng Zhengpeng <molaison@foxmail.com>

Repository CRAN

Date/Publication 2022-10-18 07:45:34 UTC

R topics documented:

mi	2
mi_balance_data	3
mi_clean_data	4
mi_data_attributes	5
mi_data_procID	5
mi_data_rawID	6
mi_get_confusion	6
mi_get_ID	7
mi_get_ID_attr	7
mi_get_miss	8
mi_get_padlen	8
mi_plot_cor	9
mi_plot_heatmap	9
mi_predict_new	10
mi_run_bmr	10
mi_split_col	11
mi_split_str	11
mi_to_numer	12
mi_train_BP	13
mi_train_rg	13
mi_train_rp	14
mi_train_xgb	15
mi_tune_rg	15
mi_tune_rp	16
mi_tune_xgb	16
Index	18

mi	<i>A wrapper function that executes MantaID workflow.</i>
----	---

Description

A wrapper function that executes MantaID workflow.

Usage

```
mi(
  mi_data_allID,
  cores = NULL,
  levels = c("*", 0:9, letters, LETTERS, "_", ".", "-", " ", "/", "\\", ":"),
  ratio = 0.3,
  para_blc = FALSE,
  model_path = NULL,
  batch_size = 128,
  epochs = 64,
```

```

    validation_split = 0.3,
    graph_path = NULL
)

```

Arguments

mi_data_allID	IDs raw data.
cores	The number of cores used when balancing data.
levels	The vector that includes all the single character occurred in IDs.
ratio	The ratio of test set.
para_blc	A logical value whether using parallel computing when balancing data.
model_path	The path to save models.
batch_size	The batch size of deep learning model fitting.
epochs	The epochs of deep learning model fitting.
validation_split	The validation ratio of deep learning model fitting.
graph_path	The path to save graphs.

Value

The list of models and graphs.

mi_balance_data	<i>Data balance. Most classes adopt random undersampling, while a few classes adopt smote method to oversample to obtain relatively balanced data;</i>
-----------------	--

Description

Data balance. Most classes adopt random undersampling, while a few classes adopt smote method to oversample to obtain relatively balanced data;

Usage

```
mi_balance_data(data, ratio = 0.3, parallel = FALSE)
```

Arguments

data	A data frame. Except class column, all are numeric types.
ratio	Numeric between 0 and 1. The percent of test set split from data.
parallel	Logical.

Value

A list contain train set and test set.

Examples

```
library(dplyr)
data = rename(iris, class =Species)
mi_balance_data(data)
```

mi_clean_data	<i>Reshape data and delete meaningless rows.</i>
---------------	--

Description

Reshape data and delete meaningless rows.

Usage

```
mi_clean_data(data, cols = everything(), placeholder = c("-"))
```

Arguments

data	A dataframe or tibble or data.table or matrix. Names of the column will be regard as the class of ID included in column.
cols	Character vectors. Columns of data that contain the IDs
placeholder	Character vectors. IDs included in placeholder will be omitted.

Value

A tibble with two columns("ID" and "class")

Examples

```
data <- tibble::tibble(
  "class1" = c("A", "B", "C", "D"),
  "class2" = c("E", "F", "G", "H"),
  "class3" = c("L", "M", "-", "O")
)
mi_clean_data(data)
```

mi_data_attributes *ID-related datasets in biomart.*

Description

ID-related datasets in biomart.

Usage

mi_data_attributes

Format

A dataframe with 65 variables and 3 variables.

name The name of dataset.

description Description of dataset.

page collection of attributes.

mi_data_procID *Processed ID data.*

Description

Processed ID data.

Usage

mi_data_procID

Format

A tibble dataframe with 5000 rows and 21 variables.

pos1 to pos20 Splited ID.

class The databases that ID belongs to.

mi_data_rawID	<i>ID dataset for testing.</i>
---------------	--------------------------------

Description

ID dataset for testing.

Usage

```
mi_data_rawID
```

Format

A tibble with 5000 rows and 2 variables.

ID A identifier character.

class The database the ID belongs to.

mi_get_confusion	<i>Compute the confusion matrix for the predict result.</i>
------------------	---

Description

Compute the confusion matrix for the predict result.

Usage

```
mi_get_confusion(result_list, ifnet = FALSE)
```

Arguments

result_list A list return from model training functions.

ifnet Logical. Whether the data is obtained by a deep learning model.

Value

A confusionMatrix object.

mi_get_ID	<i>Get ID data from Biomart database use attributes.</i>
-----------	--

Description

Get ID data from Biomart database use attributes.

Usage

```
mi_get_ID(  
  attributes,  
  biomart = "genes",  
  dataset = "hsapiens_gene_ensembl",  
  mirror = "asia"  
)
```

Arguments

attributes	A dataframe. The information we want to retrieve. Use <code>mi_get_ID_attr</code> to have a try.
biomart	BioMart database name you want to connect to. Use <code>biomaRt::listEnsembl</code> to retrieve the possible database names.
dataset	Datasets of the selected BioMart database.
mirror	Specify an Ensembl mirror to connect to.

Value

A tibble dataframe.

mi_get_ID_attr	<i>Get ID attributes from Biomart database.</i>
----------------	---

Description

Get ID attributes from Biomart database.

Usage

```
mi_get_ID_attr(  
  biomart = "genes",  
  dataset = "hsapiens_gene_ensembl",  
  mirror = "asia"  
)
```

Arguments

biomart	BioMart database name you want to connect to. Use <code>biomaRt::listEnsembl</code> to retrieve the possible database names.
dataset	Datasets of the selected BioMart database.
mirror	Specify an Ensembl mirror to connect to.

Value

A dataframe.

mi_get_miss	<i>Observe the distribution of the false response of test set.</i>
-------------	--

Description

Observe the distribution of the false response of test set.

Usage

```
mi_get_miss(predict)
```

Arguments

predict	A R6 class <code>PredictionClassif</code> .
---------	---

Value

A tibble data frame that records the number of wrong predictions for each category ID;

mi_get_padlen	<i>Get max length of ID data.</i>
---------------	-----------------------------------

Description

Get max length of ID data.

Usage

```
mi_get_padlen(data)
```

Arguments

data	A dataframe.
------	--------------

Value

A int.

Examples

```
data(mi_data_rawID)
mi_get_padlen(mi_data_rawID)
```

mi_plot_cor	<i>Plot correlation heatmap.</i>
-------------	----------------------------------

Description

Plot correlation heatmap.

Usage

```
mi_plot_cor(data, cls = "class")
```

Arguments

- data Data frame that including IDs' position features.
- cls The name of the class column.

Value

A heatmap.

Examples

```
data(mi_data_procID)
data_num <- mi_to_numer(mi_data_procID)
mi_plot_cor(data_num)
```

mi_plot_heatmap	<i>Plot heatmap for result confusion matrix.</i>
-----------------	--

Description

Plot heatmap for result confusion matrix.

Usage

```
mi_plot_heatmap(table, name = NULL, filepath = NULL)
```

Arguments

table	A table.
name	Model names.
filepath	File path the plot to save.Default NULL.

Value

A ggplot object.

mi_predict_new	<i>Predict new data with trained learner.</i>
----------------	---

Description

Predict new data with trained learner.

Usage

```
mi_predict_new(data, learner)
```

Arguments

data	A dataframe.
learner	A R6 class object.

Value

A data frame that contains features and 'predict' class.

mi_run_bmr	<i>Compare classification models with small samples.</i>
------------	--

Description

Compare classification models with small samples.

Usage

```
mi_run_bmr(data, row_num = 1000, resamplings = rsmps("cv", folds = 10))
```

Arguments

data	A tibble.All are numeric except the first column is a factor.
row_num	Number of samples used.
resamplings	R6/Resampling.Resampling method.

Value

A list of R6 class of benchmark results and scores of test set. examples `data(mi_data_procID)`
`mi_run_bmr(mi_data_procID)`

mi_split_col	<i>Cut the string of ID column character by character and divide it into multiple columns.</i>
--------------	--

Description

Cut the string of ID column character by character and divide it into multiple columns.

Usage

```
mi_split_col(data, cores = NULL, pad_len = 10)
```

Arguments

data	Dataframe(tibble) to be split.
cores	Int.The num of cores to allocate for computing.
pad_len	The length of longest id, i.e. the maxlength.

Value

A tibble with pad_len+1 column.

mi_split_str	<i>Split the string into individual characters and complete the character vector to the maximum length.</i>
--------------	---

Description

Split the string into individual characters and complete the character vector to the maximum length.

Usage

```
mi_split_str(str, pad_len)
```

Arguments

str	The string to be splited.
pad_len	The length of longest id, i.e. the maxlength.

Value

Splited character vector.

Examples

```
string_test = "Good Job"
length = 15
mi_split_str(string_test,length)
```

mi_to_numer

Convert data to numeric, and for ID column convert with fixed levels.

Description

Convert data to numeric, and for ID column convert with fixed levels.

Usage

```
mi_to_numer(
  data,
  levels = c("*", 0:9, letters, LETTERS, "_", ".", "-", " ", "/", "\\", ":")
)
```

Arguments

data A tibble with n position column(pos1,pos2,...) and class column.
levels Characters accommodated in IDs.

Value

A numeric data frame with numerical or factor type columns.

Examples

```
data(mi_data_procID)
mi_to_numer(mi_data_procID)
```

mi_train_BP	<i>Train a three layers neural network model.</i>
-------------	---

Description

Train a three layers neural network model.

Usage

```
mi_train_BP(
  train,
  test,
  cls = "class",
  path2save = NULL,
  batch_size = 128,
  epochs = 64,
  validation_split = 0.3
)
```

Arguments

train	A dataframe with class column as label.
test	A dataframe with class column as label.
cls	A character. The name of the label column.
path2save	The folder path to store the model and train history.
batch_size	Integer or NULL. Number of samples per gradient update.
epochs	Number of epochs to train the model.
validation_split	Float between 0 and 1. Fraction of the training data to be used as validation data.

Value

A list object that contains the prediction confusion matrix and the model object.

mi_train_rg	<i>Random Forest Model Training.</i>
-------------	--------------------------------------

Description

Random Forest Model Training.

Usage

```
mi_train_rg(train, test, measure = msr("classif.acc"), instance = NULL)
```

Arguments

train	A dataframe.
test	A dataframe.
measure	Model evaluation method.
instance	A tuner.

Value

A list of learner for predict and predict result of test set.

mi_train_rp	<i>Classification tree model training.</i>
-------------	--

Description

Classification tree model training.

Usage

```
mi_train_rp(train, test, measure = msr("classif.acc"), instance = NULL)
```

Arguments

train	A dataframe.
test	A dataframe.
measure	Model evaluation method. Use mlr_measures and msr() to view and choose metrics.
instance	A tuner.

Value

A list of learner for predict and predict result of test set.

mi_train_xgb	<i>Xgboost model training</i>
--------------	-------------------------------

Description

Xgboost model training

Usage

```
mi_train_xgb(train, test, measure = msr("classif.acc"), instance = NULL)
```

Arguments

train	A dataframe.
test	A dataframe.
measure	Model evaluation method.
instance	A tuner.

Value

A list of learner for predict and predict result of test set.

mi_tune_rg	<i>Tune Random Forest model by hyperband.</i>
------------	---

Description

Tune Random Forest model by hyperband.

Usage

```
mi_tune_rg(
  data,
  resampling = rsmpl("cv", folds = 5),
  measure = msr("classif.acc"),
  eta = 3
)
```

Arguments

data	A tibble. All are numeric except the first column is a factor.
resampling	R6/Resampling.
measure	Model evaluation method. Use mlr_measures and msr() to view and choose metrics.
eta	The percent parameter configurations discarded.

Value

A list of tuning instance and stage plot.

mi_tune_rp	<i>Tune decision tree model by hyperband.</i>
------------	---

Description

Tune decision tree model by hyperband.

Usage

```
mi_tune_rp(
  data,
  resampling = rsmp("bootstrap", ratio = 0.8, repeats = 5),
  measure = msr("classif.acc"),
  eta = 3
)
```

Arguments

data	A tibble. All are numeric except the first column is a factor.
resampling	R6/Resampling.
measure	Model evaluation method. Use mlr_measures and msr() to view and choose metrics.
eta	The percent parameter configurations discarded.

Value

A list of tuning instance and stage plot.

mi_tune_xgb	<i>Tune Xgboost model by hyperband.</i>
-------------	---

Description

Tune Xgboost model by hyperband.

Usage

```
mi_tune_xgb(
  data,
  resampling = rsmp("cv", folds = 5),
  measure = msr("classif.acc"),
  eta = 3
)
```


Arguments

<code>data</code>	A tibble. All are numeric except the first column is a factor.
<code>resampling</code>	R6/Resampling.
<code>measure</code>	Model evaluation method. Use <code>mlr_measures</code> and <code>msr()</code> to view and choose metrics.
<code>eta</code>	The percent parameter configurations discarded.

Value

A list of tuning instance and stage plot.

Index

* datasets

- mi_data_attributes, 5
- mi_data_procID, 5
- mi_data_rawID, 6

- mi, 2
- mi_balance_data, 3
- mi_clean_data, 4
- mi_data_attributes, 5
- mi_data_procID, 5
- mi_data_rawID, 6
- mi_get_confusion, 6
- mi_get_ID, 7
- mi_get_ID_attr, 7
- mi_get_miss, 8
- mi_get_padlen, 8
- mi_plot_cor, 9
- mi_plot_heatmap, 9
- mi_predict_new, 10
- mi_run_bmr, 10
- mi_split_col, 11
- mi_split_str, 11
- mi_to_numer, 12
- mi_train_BP, 13
- mi_train_rg, 13
- mi_train_rp, 14
- mi_train_xgb, 15
- mi_tune_rg, 15
- mi_tune_rp, 16
- mi_tune_xgb, 16