

Package ‘processpredictR’

May 9, 2026

Type Package

Title Process Prediction

Version 0.1.1

Description Means to predict process flow, such as process outcome, next activity, next time, remaining time, and remaining trace. Off-the-shelf predictive models based on the concept of Transformers are provided, as well as multiple way to customize the models. This package is partly based on work described in Zaharah A. Bukhsh, Aaqib Saeed, & Remco M. Dijkman. (2021). ``ProcessTransformer: Predictive Business Process Monitoring with Transformer Network" <[doi:10.48550/arXiv.2104.00721](https://doi.org/10.48550/arXiv.2104.00721)>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports bupaR, edeaR, eventdataR, dplyr, forcats, magrittr, reticulate, tidyr, tidyselect, purrr, stringr, keras, tensorflow, rlang, data.table, mltools, ggplot2, cli, glue, plotly

Config/testthat/edition 3

Depends R (>= 2.10)

Suggests knitr, rmarkdown, lubridate

VignetteBuilder knitr

NeedsCompilation no

Author Ivan Esin [aut],
Gert Janssenswillen [cre],
Hasselt University [cph]

Maintainer Gert Janssenswillen <gert.janssenswillen@uhasselt.be>

Repository CRAN

Date/Publication 2025-09-15 14:00:02 UTC

Contents

confusion_matrix	2
create_model	3
create_vocabulary	3
get_vocabulary	4
max_case_length	4
num_outputs	5
plot.ppred_predictions	5
ppred_examples_df	6
ppred_model	6
ppred_predictions	6
prepare_examples	6
print.ppred_model	7
processpredictR	8
split_train_test	8
stack_layers	9
tokenize	9
vocab_size	10
Index	11

confusion_matrix	<i>Confusion matrix for predictions</i>
------------------	---

Description

Confusion matrix for predictions

Usage

```
confusion_matrix(predictions, ...)
```

Arguments

predictions [ppred_predictions](#): A [data.frame](#) with predicted values returned by `predict.ppred_model()`.
 ... additional arguments.

Value

A [table](#) object that can be used for plotting a confusion matrix using `plot()`.

create_model	<i>Define transformer model</i>
--------------	---------------------------------

Description

Defines the model using the keras functional API.

Usage

```
create_model(x_train, custom = FALSE, ...)
```

Arguments

x_train	data.frame : A processed data.frame from <code>prepare_examples()</code> .
custom	logical (default <code>FALSE</code>): If <code>TRUE</code> , returns a custom model.
...	you can pass additional arguments to <code>keras::keras_model()</code> (ex.: name argument).

Value

A [list](#) containing a Transformer model (returned by `keras::keras_model()`) and some additional useful metrics.

create_vocabulary	<i>Create a vocabulary</i>
-------------------	----------------------------

Description

Creates a vocabulary of activities and outcome labels.

Usage

```
create_vocabulary(processed_df)
```

Arguments

processed_df	A preprocessed object of type ppred_examples_df returned by <code>prepare_examples()</code> .
--------------	---

Value

A [list](#) consisting of:

- "keys_x": [list](#) of activity labels
- "keys_y": [list](#) of outcome labels (none for tasks "next_time" and "remaining_time")

get_vocabulary	<i>Utils</i>
----------------	--------------

Description

Utils

Usage

```
get_vocabulary(examples)
```

Arguments

examples a preprocessed dataset returned by `prepare_examples_dt()`.

max_case_length	<i>Calculate the maximum length of a case / number of activities in the longest trace in an event log</i>
-----------------	---

Description

Calculate the maximum length of a case / number of activities in the longest trace in an event log

Usage

```
max_case_length(processed_df)
```

Arguments

processed_df A processed dataset of class `ppred_examples_df` returned by `prepare_examples()`.

Value

An integer number of the maximum case length (longest trace) in an event log.

Examples

```
library(processpredictR)
library(eventdataR)

df <- prepare_examples(patients)
max_case_length(df)
```

num_outputs	<i>Calculate number of outputs (target variables)</i>
-------------	---

Description

Calculate number of outputs (target variables)

Usage

```
num_outputs(processed_df)
```

Arguments

processed_df A processed dataset of class `ppred_examples_df`.

Value

an integer number of outputs for supplying as an argument to a Transformer model, i.e. number of unique labels for a specific process monitoring task.

Examples

```
library(processpredictR)
library(eventdataR)
df <- prepare_examples(patients)
num_outputs(df)
```

plot.ppred_predictions	<i>Plot Methods</i>
------------------------	---------------------

Description

Visualize metric

Usage

```
## S3 method for class 'ppred_predictions'
plot(x, ...)
```

Arguments

x Data to plot. An object of type `ppred_predictions`.
... Additional variables

Value

A ggplot object, which can be customized further, if deemed necessary.

ppred_examples_df *ppred_examples_df object*

Description

object of type `ppred_examples_df` is a transformed event log returned by `prepare_examples_dt()`.

ppred_model *ppred_model object*

Description

object of type `ppred_model` is a list returned by `processpredictR::create_model()` containing a custom keras functional (transformer) model and some other useful metrics of an event log.

ppred_predictions *ppred_predictions object*

Description

object of type `ppred_predictions` is a data.frame with predicted values returned by `predict.ppred_model()`.

prepare_examples *Convert a dataset of type `log` into a preprocessed format.*

Description

an event log is converted into a tibble where each row contains a cumulative sequence of activities per case. This sequence will eventually be feeded to the Transformer model's token embedding layer.

Usage

```
prepare_examples(
  log,
  task = c("outcome", "next_activity", "next_time", "remaining_time", "remaining_trace"),
  features = NULL,
  ...
)
```

Arguments

log [log](#): Object of class [log](#) or derivatives (grouped_log, eventlog, activitylog, etc.).

task [character](#): a process monitoring task for which to prepare an event log.

features [character](#) (default `NULL`): additional features. Appends attributes (if present) `numeric_features` and/or `categorical_features` to a preprocessed event log.

... additional arguments.

Value

a preprocessed dataset of class `ppred_examples_df`.

Examples

```
library(processpredictR)
library(eventdataR)

prepare_examples(patients, "next_activity")
```

print.ppred_model *Print methods*

Description

Print methods

Usage

```
## S3 method for class 'ppred_model'
print(x, ...)
```

Arguments

x [ppred_model](#): An object of class `ppred_model`.

... Additional Arguments.

Value

prints a Transformer model from a list returned by `create_model()`.

processpredictR *processpredictR*

Description

Means to predict process flow, such as process outcome, next activity, next time, remaining time, and remaining trace. Off-the-shelf predictive models based on the concept of Transformers are provided, as well as multiple way to customize the models. This package is partly based on work described in Zaharah A. Bukhsh, Aaqib Saeed, & Remco M. Dijkman. (2021). "ProcessTransformer: Predictive Business Process Monitoring with Transformer Network" [doi:10.48550/arXiv.2104.00721](https://doi.org/10.48550/arXiv.2104.00721).

Author(s)

Maintainer: Gert Janssenswillen <gert.janssenswillen@uhasselt.be>

Authors:

- Ivan Esin <ivan.esin@student.uhasselt>

Other contributors:

- Hasselt University [copyright holder]

split_train_test *Splits the preprocessed data.frame.*

Description

Returns train- and test dataframes as a list.

Usage

```
split_train_test(processed_df, split = 0.7)
```

Arguments

`processed_df` A preprocessed object of type `ppred_examples_df` returned by `prepare_examples()`.
`split` `numeric` (default 0.7): A train-test split ratio.

Value

A `list` containing the train- and the test set objects.

Examples

```
library(processpredictR)
library(eventdataR)

df <- prepare_examples(patients, "next_activity")
split_train_test(df, split = 0.8)
```

stack_layers	<i>Stacks a keras layer on top of existing model</i>
--------------	--

Description

User friendly interface to add a keras layer on top of existing model.

Usage

```
stack_layers(object, ...)
```

Arguments

object	a list containing a model returned by <code>create_model()</code> .
...	functions for adding layers by using functional keras API. For example, <code>keras::layer_dense(units=32, activation="relu")</code> .

Value

a [list](#) containing an adapted Transformer model.

tokenize	<i>Tokenize features and target of a processed dataset of class ppred_examples_df</i>
----------	---

Description

Tokenize features and target of a processed [ppred_examples_df](#) object to fit the Transformer model.

Usage

```
tokenize(processed_df)
```

Arguments

processed_df	A preprocessed object of type ppred_examples_df returned by <code>prepare_examples()</code> .
--------------	---

Value

A [list](#) of (sequence) tokens and additional numeric or categorical features.

vocab_size	<i>Calculate the vocabulary size, i.e. the sum of number of activities, outcome labels and padding keys</i>
------------	---

Description

Calculate the vocabulary size, i.e. the sum of number of activities, outcome labels and padding keys

Usage

```
vocab_size(processed_df)
```

Arguments

processed_df A processed dataset of class [ppred_examples_df](#) from `prepare_examples()`.

Value

an integer number of vocabulary size to define the Transformer model.

Examples

```
library(processpredictR)
library(eventdataR)
df <- prepare_examples(patients)
vocab_size(df)
```

Index

* **visualization**

plot.ppred_predictions, 5

character, 7

confusion_matrix, 2

create_model, 3

create_vocabulary, 3

data.frame, 2, 3, 8

FALSE, 3

get_vocabulary, 4

list, 3, 8–10

log, 6, 7

logical, 3

max_case_length, 4

NULL, 7

num_outputs, 5

numeric, 8

plot.ppred_predictions, 5

ppred_examples_df, 3–6, 6, 7–10

ppred_model, 6, 6, 7

ppred_predictions, 2, 5, 6, 6

prepare_examples, 6

print.ppred_model, 7

processpredictR, 8

processpredictR-package

(processpredictR), 8

split_train_test, 8

stack_layers, 9

table, 2

tokenize, 9

TRUE, 3

vocab_size, 10