

Package ‘codez’

July 17, 2026

Type Package

Title Torch Seq2Seq Encoder-Decoder Model for Time-Feature Analysis

Version 2.0.0

Maintainer Giancarlo Vercellino <giancarlo.vercellino@gmail.com>

Description Proposes Seq2seq Time-Feature Analysis using a torch Encoder-Decoder to project into latent space and a Forward Network to predict the next sequence, with dependency-light local support functions, tidy outputs and baseline backtesting helpers.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 3.6)

URL https://rpubs.com/giancarlo_vercellino/codez

Suggests testthat (>= 3.0.0), torch

Config/testthat/edition 3

NeedsCompilation no

Author Giancarlo Vercellino [aut, cre, cph]

Repository CRAN

Date/Publication 2026-07-17 15:50:21 UTC

Contents

amzn_aapl_fb	2
autoplot	2
codez	3
codez_control	6
codez_forecast_tidy	7
codez_model_methods	7
fit_codez	8

Index	10
--------------	-----------

amzn_aapl_fb	<i>amzn_aapl_fb data set</i>
--------------	------------------------------

Description

A data frame with the close prices for Amazon, Google and Facebook.

Usage

```
amzn_aapl_fb
```

Format

A data frame with 4 columns and 1798 rows.

Source

Yahoo Finance

autoplot	<i>Create a codez plot object</i>
----------	-----------------------------------

Description

Generic for creating dependency-free plot objects. Use `plot()` to draw the returned object with base R graphics.

Usage

```
autoplot(object, ...)
```

Arguments

object	An object with an <code>autoplot()</code> method.
...	Additional arguments passed to methods.

Value

A lightweight codez plot object when called on a supported codez object.

`codez`*codez*

Description

Seq2seq Time-Feature Analysis using an Encoder-Decoder to project into latent space and a Forward Network to predict the next sequence.

Usage

```
codez(  
  df,  
  seq_len = NULL,  
  n_windows = 10,  
  latent = NULL,  
  smoother = FALSE,  
  n_samp = 30,  
  autoencoder_layers_n = NULL,  
  autoencoder_layers_size = NULL,  
  autoencoder_activ = NULL,  
  forward_net_layers_n = NULL,  
  forward_net_layers_size = NULL,  
  forward_net_activ = NULL,  
  forward_net_reg_L1 = NULL,  
  forward_net_reg_L2 = NULL,  
  forward_net_drop = NULL,  
  loss_metric = "mae",  
  autoencoder_optimizer = NULL,  
  forward_net_optimizer = NULL,  
  epochs = 100,  
  patience = 10,  
  holdout = 0.5,  
  verbose = FALSE,  
  ci = 0.8,  
  error_scale = "naive",  
  error_benchmark = "naive",  
  dates = NULL,  
  seed = 42,  
  batch_size = 32,  
  validation_split = 0.1,  
  n_sim = 1000,  
  backend = "torch"  
)
```

Arguments

<code>df</code>	A data frame with time features on columns. They could be numeric variables or categorical, but not both.
-----------------	---

seq_len	Positive integer. Time-step number of the forecasting sequence. Default: NULL (random selection within 2 to max preset boundary).
n_windows	Positive integer. Number of validation windows to test prediction error. Default: 10.
latent	Positive integer. Dimensions of the latent space for encoding-decoding operations. Default: NULL (random selection within preset boundaries)
smoother	Logical. Perform optimal smoothing using standard loess for each time feature. Default: FALSE
n_samp	Positive integer. Number of samples for random search. Default: 30.
autoencoder_layers_n	Positive integer. Number of layers for the encoder-decoder model. Default: NULL (random selection within preset boundaries)
autoencoder_layers_size	Positive integer. Numbers of nodes for the encoder-decoder model. Default: NULL (random selection within preset boundaries)
autoencoder_activ	String. Activation function to be used by the encoder-decoder model. Implemented functions are: "linear", "relu", "leaky_relu", "selu", "elu", "sigmoid", "tanh", "swish", "gelu". Default: NULL (random selection within standard activations)
forward_net_layers_n	Positive integer. Number of layers for the forward net model. Default: NULL (random selection within preset boundaries)
forward_net_layers_size	Positive integer. Numbers of nodes for the forward net model. Default: NULL (random selection within preset boundaries)
forward_net_activ	String. Activation function to be used by the forward net model. Implemented functions are: "linear", "relu", "leaky_relu", "selu", "elu", "sigmoid", "tanh", "swish", "gelu". Default: NULL (random selection within standard activations)
forward_net_reg_L1	Positive numeric between. Weights for L1 regularization. Default: NULL (random selection within preset boundaries).
forward_net_reg_L2	Positive numeric between. Weights for L2 regularization. Default: NULL (random selection within preset boundaries).
forward_net_drop	Positive numeric between 0 and 1. Value for the dropout parameter for each layer of the forward net model (for example, a neural net with 3 layers may have dropout = c(0, 0.5, 0.3)). Default: NULL (random selection within preset boundaries).
loss_metric	String. Loss function for both models. Available metrics: "mse", "mae", "mape". Default: "mae".
autoencoder_optimizer	String. Optimization method for autoencoder. Implemented methods are: "adam", "adadelat", "adagrad", "rmsprop", "sgd", "nadam", "adamax". Default: NULL (random selection within standard optimizers).

<code>forward_net_optimizer</code>	String. Optimization method for forward net. Implemented methods are: "adam", "adadelta", "adagrad", "rmsprop", "sgd", "nadam", "adamax". Default: NULL (random selection within standard optimizers).
<code>epochs</code>	Positive integer. Default: 100.
<code>patience</code>	Positive integer. Waiting time (in epochs) before evaluating the overfit performance. Default: 10.
<code>holdout</code>	Positive numeric between 0 and 1. Holdout sample for validation. Default: 0.5.
<code>verbose</code>	Logical. Default: FALSE.
<code>ci</code>	Positive numeric. Confidence interval. Default: 0.8
<code>error_scale</code>	String. Scale for the scaled error metrics (for continuous variables). Two options: "naive" (average of naive one-step absolute error for the historical series) or "deviation" (standard error of the historical series). Default: "naive".
<code>error_benchmark</code>	String. Benchmark for the relative error metrics (for continuous variables). Two options: "naive" (sequential extension of last value) or "average" (mean value of true sequence). Default: "naive".
<code>dates</code>	Date. Vector with dates for time features.
<code>seed</code>	Positive integer. Random seed. Default: 42.
<code>batch_size</code>	Positive integer. Mini-batch size for torch training. Default: 32.
<code>validation_split</code>	Positive numeric between 0 and 1. Validation share used by the autoencoder. Default: 0.1.
<code>n_sim</code>	Positive integer. Number of simulated paths generated from validation residuals. Default: 1000.
<code>backend</code>	String. Neural backend. Only "torch" is supported in codez 2.0.

Value

This function returns a list including:

- `history`: a table with the sampled models, hyper-parameters, validation errors
- `best_model`: results for the best selected model according to the weighted average rank, including:
 - `predictions`: for continuous variables, min, max, q25, q50, q75, quantiles at selected ci, mean, sd, mode, skewness, kurtosis, IQR to range, risk ratio, upside probability and divergence for each point fo predicted sequences; for factor variables, min, max, q25, q50, q75, quantiles at selected ci, proportions, difformity (deviation of proportions normalized over the maximum possible deviation), entropy, upgrade probability and divergence for each point fo predicted sequences
 - `testing_errors`: testing errors for each time feature for the best selected model (for continuous variables: me, mae, mse, rmsse, mpe, mape, rmae, rrmse, rame, mase, smse, sce, gmrae; for factor variables: czekanowski, tanimoto, cosine, hassebrook, jaccard, dice, canberra, gower, lorentzian, clark)
 - `plots`: standard plots with confidence interval for each time feature
- `time_log`

Author(s)

Maintainer: Giancarlo Vercellino <giancarlo.vercellino@gmail.com> [copyright holder]
 Giancarlo Vercellino <giancarlo.vercellino@gmail.com>

See Also

Useful links:

- https://rpubs.com/giancarlo_vercellino/codez

codez_control

codez training controls

Description

Create reusable training controls for the codez 2.0 torch backend.

Usage

```
codez_control(
  batch_size = 32,
  validation_split = 0.1,
  n_sim = 1000,
  backend = "torch",
  epochs = NULL,
  patience = NULL,
  holdout = NULL
)
```

Arguments

batch_size	Positive integer. Mini-batch size for torch training.
validation_split	Numeric value in [0, 1). Validation share for the autoencoder.
n_sim	Positive integer. Number of simulated paths generated from validation residuals.
backend	String. Neural backend. Only "torch" is supported.
epochs	Optional positive integer overriding codez() epochs.
patience	Optional positive integer overriding early-stopping patience.
holdout	Optional numeric value between 0 and 1 overriding validation holdout.

Value

A list of controls for fit_codez().

codez_forecast_tidy	<i>Extract tidy forecasts from a codez result</i>
---------------------	---

Description

Extract tidy forecasts from legacy `codez()` output or a fitted `codez_model`.

Usage

```
codez_forecast_tidy(result)
```

Arguments

`result` A result returned by `codez()` or a `codez_model`.

Value

A data frame with one row per feature and forecast horizon.

codez_model_methods	<i>codez model methods</i>
---------------------	----------------------------

Description

S3 helpers for fitted codez 2.0 models.

Usage

```
## S3 method for class 'codez_model'
predict(object, type = c("forecast", "raw", "plots", "benchmarks"), ...)
## S3 method for class 'codez_model'
summary(object, ...)
## S3 method for class 'codez_model'
autoplot(object, feature = NULL, ...)
## S3 method for class 'codez_model'
plot(x, ...)
## S3 method for class 'codez_model'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

object	A codez_model.
x	A codez_model.
type	Output type for predict().
feature	Optional feature name or names to plot.
...	Additional arguments reserved for future use.
row.names	Optional row names for as.data.frame().
optional	Optional argument passed through by as.data.frame().

Value

predict() and as.data.frame() return forecast data. summary() returns a summary object. autoplot() returns a lightweight codez plot object and plot() draws it with base R graphics.

fit_codez	<i>Fit a codez 2.0 model</i>
-----------	------------------------------

Description

fit_codez() is the codez 2.0 workflow wrapper. It keeps codez() as the forecasting engine, then attaches tidy forecasts, rolling baseline comparisons, the training call, and S3 methods.

Usage

```
fit_codez(df, ..., dates = NULL, control = codez_control())
```

Arguments

df	A data frame with time features on columns.
...	Arguments passed to codez().
dates	Optional Date vector with one value per row of df.
control	A codez_control() object.

Value

A codez_model object containing the legacy result, tidy forecasts, baseline backtests, controls, call, and backend metadata.

Examples

```
## Not run:
model <- fit_codez(
  amzn_aapl_fb[, -1],
  dates = as.Date(amzn_aapl_fb$Date),
  seq_len = 20,
  n_samp = 3,
  n_windows = 5
)
predict(model)
summary(model)
autoplot(model)

## End(Not run)
```

Index

* datasets

amzn_aapl_fb, [2](#)

amzn_aapl_fb, [2](#)

as.data.frame.codez_model
(codez_model_methods), [7](#)

autoplot, [2](#)

autoplot.codez_model
(codez_model_methods), [7](#)

codez, [3](#)

codez-package (codez), [3](#)

codez_control, [6](#)

codez_forecast_tidy, [7](#)

codez_model_methods, [7](#)

fit_codez, [8](#)

plot.codez_model (codez_model_methods),
[7](#)

predict.codez_model
(codez_model_methods), [7](#)

summary.codez_model
(codez_model_methods), [7](#)