Package: TSdeeplearning (via r-universe)

November 3, 2024

Type Package

Title Deep Learning Model for Time Series Forecasting

Version 0.1.0

Maintainer Ronit Jaiswal <ronitjaiswal2912@gmail.com>

Description RNNs are preferred for sequential data like time series, speech, text, etc. but when dealing with long range dependencies, vanishing gradient problems account for their poor performance. LSTM and GRU are effective solutions which are nothing but RNN networks with the abilities of learning both short-term and long-term dependencies. Their structural makeup enables them to remember information for a long period without any difficulty. LSTM consists of one cell state and three gates, namely, forget gate, input gate and output gate whereas GRU comprises only two gates, namely, reset gate and update gate. This package consists of three different functions for the application of RNN, LSTM and GRU to any time series data for its forecasting. For method details see Jaiswal, R. et al. (2022). <doi:10.1007/s00521-021-06621-3>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Imports keras, tensorflow, reticulate, tsutils, BiocGenerics, utils, graphics, magrittr

Depends R (>= 2.10)

NeedsCompilation no

Author Ronit Jaiswal [aut, cre], Girish Kumar Jha [aut, ths, ctb], Rajeev Ranjan Kumar [aut, ctb], Kapil Choudhary [aut, ctb]

Date/Publication 2022-09-09 07:33:00 UTC

Repository https://ronit10976.r-universe.dev

RemoteUrl https://github.com/cran/TSdeeplearning

RemoteRef HEAD

RemoteSha 3e2393ee916f87bb824c0ceaeab8dbecf479f821

Contents

`Data_Maize` *Monthly International Maize Price Data*

Description

Monthly international Maize price (Dollor per million ton) from January 2010 to June 2020.

Usage

```
data("Data_Maize")
```

Format

A time series data with 126 observations.

`price` a time series

Details

Dataset contains 126 observations of monthly international Maize price (Dollor per million ton). It is obtained from World Bank "Pink sheet".

Source

https://www.worldbank.org/en/research/commodity-markets

References

https://www.worldbank.org/en/research/commodity-markets

Examples

```
data(Data_Maize)
```

GRU_ts *Gated Recurrent Unit Model*

Description

The GRU function computes forecasted value with different forecasting evaluation criteria for gated recurrent unit model.

Usage

```
GRU_ts(xt, xtlag = 4, uGRU = 2, Drate = 0, nEpochs = 10,
Loss = "mse", AccMetrics = "mae",ActFn = "tanh",
Split = 0.8, Valid = 0.1)
```

Arguments

`xt`	Input univariate time series (ts) data.
`xtlag`	Lag of time series data.
`uGRU`	Number of unit in GRU layer.
`Drate`	Dropout rate.
`nEpochs`	Number of epochs.
`Loss`	Loss function.
`AccMetrics`	Metrics.
`ActFn`	Activation function.
`Split`	Index of the split point and separates the data into the training and testing datasets.
`Valid`	Validation set.

Details

The gated recurrent unit (GRU) was introduced by Cho et al.(2014). A GRU is part of a specific model of recurrent neural network that intends to use connections through a sequence of nodes to perform machine learning tasks associated with memory and clustering. Its internal structure is simpler and, therefore, it is also easier to train, as less calculation is required to upgrade the internal states. The update port controls the extent to which the state information from the previous moment is retained in the current state, while the reset port determines whether the current state should be combined with the previous information. Gated recurrent units help to adjust neural network input weights to solve the vanishing gradient problem that is a common issue with recurrent neural networks.

Value

`TrainFittedValue`	Training Fitted value for given time series data.
`TestPredictedValue`	Final forecasted value of the GRU model.
`fcast_criteria`	Different Forecasting evaluation criteria for GRU model.

References

Cho, K., Van Merriënboer, B., Bahdanau, D. and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

See Also

LSTM, RNN

Examples

```
data("Data_Maize")
GRU_ts(Data_Maize)
```

`LST_ts` *Long- Short Term Memory Model*

Description

The LSTM function computes forecasted value with different forecasting evaluation criteria for long- short term memory model.

Usage

```
LSTM_ts(xt, xtlag = 4, uLSTM = 2, Drate = 0, nEpochs = 10,
Loss = "mse", AccMetrics = "mae",ActFn = "tanh",
Split = 0.8, Valid = 0.1)
```

Arguments

`xt`	Input univariate time series (ts) data.
`xtlag`	Lag of time series data.
`uLSTM`	Number of unit in LSTM layer.
`Drate`	Dropout rate.
`nEpochs`	Number of epochs.
`Loss`	Loss function.
`AccMetrics`	Metrics.
`ActFn`	Activation function.
`Split`	Index of the split point and separates the data into the training and testing datasets.
`Valid`	Validation set.

Details

Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) based RNN is designed to overcome the vanishing gradients problem while dealing with long term dependencies. In contrast to standard RNN, LSTM has this peculiar and unique inbuilt ability by maintaining a memory cell to determine which unimportant features should be forgotten and which important features should be remembered during the learning process (Jaiswal et al., 2022). An LSTM model analyses and captures both short-term and long-term temporal dependencies of a complex time series effectively due to its architecture of recurrent neural network and the memory function used in the hidden nodes.

Value

`TrainFittedValue`
: Training Fitted value for given time series data.

`TestPredictedValue`
: Final forecasted value of the LSTM model.

`fcast_criteria`
: Different Forecasting evaluation criteria for LSTM model.

References

Cho, K., Van Merriënboer, B., Bahdanau, D. and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

See Also

GRU, RNN

Examples

```
data("Data_Maize")
LSTM_ts(Data_Maize)
```

`RNN_ts` *Recurrent neural network Model*

Description

The RNN function computes forecasted value with different forecasting evaluation criteria for recurrent neural network model.

Usage

```
RNN_ts(xt, xtlag = 4, uRNN = 2, Drate = 0, nEpochs = 10,
Loss = "mse", AccMetrics = "mae",ActFn = "tanh",
Split = 0.8, Valid = 0.1)
```

Arguments

`xt`	Input univariate time series (ts) data.
`xtlag`	Lag of time series data.
`uRNN`	Number of unit in RNN layer.
`Drate`	Dropout rate.
`nEpochs`	Number of epochs.
`Loss`	Loss function.
`AccMetrics`	Metrics.
`ActFn`	Activation function.
`Split`	Index of the split point and separates the data into the training and testing datasets.
`Valid`	Validation set.

Details

Recurrent neural networks (RNNs) (Rumelhart 1986) add the explicit handling of order between observations when learning a mapping function from inputs to outputs. RNNs actually process single elements of any input sequence at a particular time, and maintain a 'state vector' in their hidden units. Nevertheless, when the interval of data dependencies increases, the standard RNNs tend to suffer increasingly heavily from the problem of either vanishing gradient or exploding gradient (Bengio et al. 1994; Lin et al. 1996).

Value

`TrainFittedValue`
: Training Fitted value for given time series data.

`TestPredictedValue`
: Final forecasted value of the RNN model.

`fcast_criteria`
: Different Forecasting evaluation criteria for RNN model.

References

Bengio et al. 1994; Lin Sagheer A, Kotb M (2019) Time series forecasting of petroleum production using deep LSTM recurrent networks. Neurocomputing 323: 203–213.

Rumelhart DE (1986) Learning internal representations by error propagation. In: Parallel distributed processing: Explorations in the microstructure of cognition. pp 318–362.

Jha, G. K. and Sinha, K. (2014). Time-delay neural networks for time series prediction: An application to the monthly wholesale price of oilseeds in India. Neural Computing and Applications, 24(3–4), 563–571. Jaiswal, R., Jha, G. K., Kumar, R. R. and Choudhary, K. (2022). Deep long short-term memory based model for agricultural price forecasting. Neural Computing and Applications, 34(6), 4661–4676.

See Also

LSTM, GRU

Examples

```
data("Data_Maize")
RNN_ts(Data_Maize)
```

Index