# Package 'LDNN'

October 12, 2022

**Type** Package

**Title** Longitudinal Data Neural Network

**Version** 1.10

**Date** 2020-11-25

**Author** Vasileios Karapoulios

**Maintainer** Vasileios Karapoulios <billkarap123@gmail.com>

**Description** This is a Neural Network regression model implementation using 'Keras', consisting of 10 Long Short-Term Memory layers that are fully connected along with the rest of the inputs.

**License** GNU General Public License

**RoxygenNote** 7.1.1

**LazyData** true

**Encoding** UTF-8

**Imports** devtools, reticulate, tensorflow

**Depends** keras

**Suggests** testthat,knitr,rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-12-02 10:30:10 UTC

## R topics documented:

---

create_model            *Pre-defined Neural Network for Longitudinal Data*

---

### Description

Create the Neural Network model (Keras).

### Usage

```
create_model(
  rnn_inputs,
  recurrent_droppout,
  inputs,
  layer_dropout,
  n_nodes_hidden_layers,
  loss_function,
  opt,
  metric
)
```

### Arguments

| | |
|---|---|
| `rnn_inputs` | The number of inputs (integers) per each LSTM (vector of length 10). |
| `recurrent_droppout` | |
| | The dropout to be applied in the LSTMs (between 0 and 1). |
| `inputs` | The number of inputs (integer) to be concatenated with the output of the LSTMs. |
| `layer_dropout` | The dropout to be applied between the hidden layers (between 0 and 1). |
| `n_nodes_hidden_layers` | |
| | The number of nodes in the hidden layers (2 in total). |
| `loss_function` | The loss function to be used. |
| `opt` | The optimizer to be used. |
| `metric` | The metric to be used. |

### Value

The model object built in Keras.

### Examples

```
inp = c(20,24,24,24,16,16,16,16,16,15)
rec_drop = rep(0.1,10)
l_drop = c(0.1,0.1)
## Not run:
create_model(inp,rec_drop,232,c(0.1,0.1),l_drop,'mean_squared_error','adam','mean_absolute_error')

## End(Not run)
```

```
# The functions require to have python installed
# As well as tensorflow, keras and reticulate package.
```

---

evaluate_model            *Evaluate the pre-defined Neural Network for Longitudinal Data*

---

### Description

Evaluate the fitted Neural Network model (Keras).

### Usage

```
evaluate_model(
  model,
  X1_test,
  X2_test,
  X3_test,
  X4_test,
  X5_test,
  X6_test,
  X7_test,
  X8_test,
  X9_test,
  X10_test,
  Xif_test,
  y_test,
  bsize
)
```

### Arguments

| | |
|---|---|
| model | The fitted model object produced by create_model(). |
| X1_test | Features as inputs of 1st LSTM. |
| X2_test | Features as inputs of 2nd LSTM. |
| X3_test | Features as inputs of 3rd LSTM. |
| X4_test | Features as inputs of 4th LSTM. |
| X5_test | Features as inputs of 5th LSTM. |
| X6_test | Features as inputs of 6th LSTM. |
| X7_test | Features as inputs of 7th LSTM. |
| X8_test | Features as inputs of 8th LSTM. |
| X9_test | Features as inputs of 9th LSTM. |
| X10_test | Features as inputs of 10th LSTM. |
| Xif_test | The features to be concatenated with the outputs of the LSTMs. |
| y_test | The target variable. |
| bsize | The batch size. |

## Value

The evaluation results.

## Examples

```
X1_test <- matrix(runif(500*20), nrow=500, ncol=20)
X2_test <- matrix(runif(500*24), nrow=500, ncol=24)
X3_test <- matrix(runif(500*24), nrow=500, ncol=24)
X4_test <- matrix(runif(500*24), nrow=500, ncol=24)
X5_test <- matrix(runif(500*16), nrow=500, ncol=16)
X6_test <- matrix(runif(500*16), nrow=500, ncol=16)
X7_test <- matrix(runif(500*16), nrow=500, ncol=16)
X8_test <- matrix(runif(500*16), nrow=500, ncol=16)
X9_test <- matrix(runif(500*16), nrow=500, ncol=16)
X10_test <- matrix(runif(500*15), nrow=500, ncol=15)
Xif_test <- matrix(runif(500*232), nrow=500, ncol=232)
y_test <- matrix(runif(500), nrow=500, ncol=1)
## Not run:
evaluate_model(fitted_model,X1_test,X2_test,X3_test,X4_test,X5_test,X6_test,
X7_test,X8_test,X9_test,X10_test,Xif_test,y_test,32)

## End(Not run)
# The functions require to have python installed
# As well as tensorflow, keras and reticulate package.
```

---

  fit_model                               *Fit the pre-defined Neural Network for Longitudinal Data*

---

## Description

Fit the created Neural Network model (Keras).

## Usage

```
fit_model(
  model,
  ver,
  n_epoch,
  bsize,
  X1,
  X2,
  X3,
  X4,
  X5,
  X6,
  X7,
  X8,
  X9,
```

```
   X10,
   Xif,
   y
)
```

## Arguments

| | |
|---|---|
| model | The model object produced by create_model(). |
| ver | ver=0 to show nothing, ver=1 to show animated progress bar, ver=2 to just mention the number of epoch during training. |
| n_epoch | The number of epochs to train the model. |
| bsize | The batch size. |
| X1 | Features as inputs of 1st LSTM. |
| X2 | Features as inputs of 2nd LSTM. |
| X3 | Features as inputs of 3rd LSTM. |
| X4 | Features as inputs of 4th LSTM. |
| X5 | Features as inputs of 5th LSTM. |
| X6 | Features as inputs of 6th LSTM. |
| X7 | Features as inputs of 7th LSTM. |
| X8 | Features as inputs of 8th LSTM. |
| X9 | Features as inputs of 9th LSTM. |
| X10 | Features as inputs of 10th LSTM. |
| Xif | The features to be concatenated with the outputs of the LSTMs. |
| y | The target variable. |

## Value

The fitted model.

## Examples

```
X1 <- matrix(runif(500*20), nrow=500, ncol=20)
X2 <- matrix(runif(500*24), nrow=500, ncol=24)
X3 <- matrix(runif(500*24), nrow=500, ncol=24)
X4 <- matrix(runif(500*24), nrow=500, ncol=24)
X5 <- matrix(runif(500*16), nrow=500, ncol=16)
X6 <- matrix(runif(500*16), nrow=500, ncol=16)
X7 <- matrix(runif(500*16), nrow=500, ncol=16)
X8 <- matrix(runif(500*16), nrow=500, ncol=16)
X9 <- matrix(runif(500*16), nrow=500, ncol=16)
X10 <- matrix(runif(500*15), nrow=500, ncol=15)
Xif <- matrix(runif(500*232), nrow=500, ncol=232)
y <- matrix(runif(500), nrow=500, ncol=1)
## Not run:
fitted_model = fit_model(model,0,1,32,X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,Xif,y)
```

```
## End(Not run)
# The functions require to have python installed
# As well as tensorflow, keras and reticulate package.
```

# Index