Package 'learNN'

October 13, 2022

Title Examples of Neural Networks

Version 0.2.0

Description Implementations of several basic neural network concepts in R, as based on posts on \url{http://qua.st/}.

Depends R (>= 2.10.0)

License GPL-3

Suggests testthat

NeedsCompilation no

Author Bastiaan Quast [aut, cre]

Maintainer Bastiaan Quast <bquast@gmail.com>

Repository CRAN

Date/Publication 2015-09-29 17:33:17

R topics documented:

`learnn` *Learn Neural Networks*

Description

Learn Neural Networks

learn_bp *Learn Back Propagation*

Description

Learn Back Propagation

Usage

```
learn_bp(X, y)
```

Arguments

X	input data
y	output data

References

http://qua.st/handcoding-neural-network/

http://qua.st/handcoding-neural-network/ http://iamtrask.github.io/2015/07/12/basic-python-network/

Examples

```
# create data
X = matrix(c(0,0,1,
             0,1,1,
             1,0,1,
             1,1,1), nrow=4, byrow=TRUE)

y = matrix(c(0,
             1,
             1,
             0),
             nrow=4)

# run full function
learn_bp(X, y)
```

learn_bp11 *Learn Back Propagation in 11 lines*

Description

Learn Back Propagation in 11 lines

Usage

```
learn_bp11(X, y)
```

Arguments

X	input data
y	output data

References

http://qua.st/handcoding-neural-network/ http://iamtrask.github.io/2015/07/12/basic-python-network/

See Also

learn_bp

Examples

```
# construct new data
X = matrix(c(0,0,1,0,1,1,1,0,1,1,1,1), nrow=4, byrow=TRUE)
y = matrix(c(0,1,1,0),nrow=4)

# run 11 lines function
learn_bp11(X, y)

# view output
syn0
syn1
```

---

learn_do *Learn Dropout*

Description

Learn Dropout

Usage

```
learn_do(X, y, alpha, hidden_dim, dropout_percent, do_dropout = TRUE)
```

Arguments

X	input data
y	output data
alpha	proportion of gradient descent
hidden_dim	dimension of the hidden layer
dropout_percent	percentage to be used for the dropout
do_dropout	should dropout be used

References

http://qua.st/handcoding-dropout/ http://iamtrask.github.io/2015/07/28/dropout/

Examples

```
# construct data
X = matrix(c(0,0,1,0,1,1,1,0,1,1,1,1), nrow=4, byrow=TRUE)
y = matrix(c(0,1,1,0),nrow=4)

# set hyperparameters
alpha = 0.5
hidden_dim = 4
dropout_percent = 0.2
do_dropout = TRUE

# run 11 lines function
learn_do(X, y, alpha, hidden_dim, dropout_percent, TRUE)

# view output
synapse_0
synapse_1
```

`learn_do15` *Learn Dropout in 15 lines*

Description

Learn Dropout in 15 lines

Usage

```
learn_do15(X, y, alpha, hidden_dim, dropout_percent, do_dropout = TRUE)
```

Arguments

`X`	input data
`y`	output data
`alpha`	proportion of gradient descent
`hidden_dim`	dimension of the hidden layer
`dropout_percent`	percentage to be used for the dropout
`do_dropout`	should dropout be used

References

http://qua.st/handcoding-dropout/ http://iamtrask.github.io/2015/07/28/dropout/

See Also

learn_do

Examples

```
# construct data
X = matrix(c(0,0,1,0,1,1,1,0,1,1,1,1), nrow=4, byrow=TRUE)
y = matrix(c(0,1,1,0),nrow=4)

# set hyperparameters
alpha = 0.5
hidden_dim = 4
dropout_percent = 0.2
do_dropout = TRUE

# run 11 lines function
learn_do15(X, y, alpha, hidden_dim, dropout_percent, TRUE)

# view output
synapse_0
synapse_1
```

---

learn_gd *Learn Gradient Descent*

Description

Learn Gradient Descent

Usage

```
learn_gd(X, y, alpha, hiddenSize)
```

Arguments

X	input data
y	output data
alpha	fraction of gradient descent
hiddenSize	size of the hidden layer

References

http://qua.st/handcoding-gradient-descent/ http://iamtrask.github.io/2015/07/27/python-network-part2/

Examples

```
# input dataset
X = matrix(c(0,0,1,
             0,1,1,
             1,0,1,
             1,1,1), nrow=4, byrow=TRUE)

# output dataset
y = matrix(c(0,
             1,
             1,
             0), nrow=4)

# set parameters
alpha = 0.1
hiddenSize = 32
# also try using:
# alphas = c(0.001,0.01,0.1,1,10,100,1000)
# for (alpha in alphas) {
#   print(paste("Training With Alpha", alpha))
#   learn_gd(X, y, alpha, hiddenSize)         }

# run gradient descent function
learn_gd(X, y, alpha, hiddenSize)
```

`learn_gd13` *Learn Gradient Descent in 13 lines*

Description

Learn Gradient Descent in 13 lines

Usage

```
learn_gd13(X, y, alpha, hidden_dim)
```

Arguments

`X`	input data
`y`	output data
`alpha`	alpha to be used
`hidden_dim`	dimension of the hidden layer

References

http://qua.st/handcoding-gradient-descent/ http://iamtrask.github.io/2015/07/27/python-network-part2/

See Also

learn_gd

Examples

```
# create new data
alpha = 0.5
hidden_dim = 4
X = matrix(c(0,0,1,0,1,1,1,0,1,1,1,1), nrow=4, byrow=TRUE)
y = matrix(c(0,1,1,0),nrow=4)

# run 13 lines function
learn_gd13(X, y, alpha, hidden_dim)
```

Index