

# Package ‘DamiaNN’

May 7, 2026

**Type** Package

**Title** Neural Network Numerai

**Version** 1.0.0

**Author** Damian Siniakowicz

**Maintainer** Damian Siniakowicz <DamianSiniakowicz@gmail.com>

**Date** 2016-09-13

**Description** Interactively train neural networks on Numerai, <<https://numer.ai/>>, data. Generate tournament predictions and write them to a CSV.

**Imports** caret, methods, testthat

**License** GPL-3

**LazyData** FALSE

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2016-09-14 18:50:32

## Contents

back_propogation,Neural_Network,numeric,numeric,numeric-method . . . . .	2
forward_propogation,Neural_Network,matrix-method . . . . .	2
Get_Cost,Neural_Network,numeric-method . . . . .	3
Get_LogLoss . . . . .	3
Get_Number_Observations,Neural_Network-method . . . . .	4
initialize,Neural_Network-method . . . . .	4
Neural_Network-class . . . . .	5
Predict,Neural_Network,data.frame-method . . . . .	5
Start . . . . .	6
Train,Neural_Network,data.frame,numeric,numeric,numeric-method . . . . .	6

<b>Index</b>	<b>7</b>
--------------	----------

---

back\_propagation,Neural\_Network,numeric,numeric,numeric-method  
*back\_prop*

---

**Description**

updates connection strengths using results of last forward prop

**Usage**

```
## S4 method for signature 'Neural_Network,numeric,numeric,numeric'
back_propagation(object,
  target, regularization_parameter, learning_rate)
```

**Arguments**

object            is a Neural\_Network  
target            is a numeric vector  
regularization\_parameter  
                  is non-negative number punishes strong connections  
learning\_rate    is a positive number that controls the rate at which connections are adjusted

**Value**

Neural\_Network

---

forward\_propagation,Neural\_Network,matrix-method  
*f\_prop*

---

**Description**

... part of the training program

**Usage**

```
## S4 method for signature 'Neural_Network,matrix'
forward_propagation(object, dataset)
```

**Arguments**

object            is a Neural\_Network  
dataset            is a matrix not containing the target vector

**Value**

Neural\_Network

---

Get\_Cost,Neural\_Network,numeric-method  
*cost*

---

**Description**

get the logarithmic loss for a set of predictions

**Usage**

```
## S4 method for signature 'Neural_Network,numeric'  
Get_Cost(object, target)
```

**Arguments**

object           ... a Neural\_Network that has run forward\_prop at least once  
target           ... a numeric vector ... the target ...

**Value**

Numeric

---

Get\_LogLoss           *log loss*

---

**Description**

get log loss

**Usage**

```
Get_LogLoss(predictions, target)
```

**Arguments**

predictions    is a numeric vector  
target          is a numeric vector

**Value**

Numeric

---

Get\_Number\_Observations,Neural\_Network-method  
*num observs*

---

**Description**

returns the number of observations that the network has processed

**Usage**

```
## S4 method for signature 'Neural_Network'
Get_Number_Observations(object)
```

**Arguments**

object                   ... a Neural Network that has called fprop. ie. that has called train/predict

**Value**

Numeric

---

initialize,Neural\_Network-method  
*init*

---

**Description**

initializes a neural network capable of studying datasets with ncol = to the ncol(sample\_dataset) and making predictions on such datasets

**Usage**

```
## S4 method for signature 'Neural_Network'
initialize(.Object, number_predictors,
  hidden_layer_lengths)
```

**Arguments**

.Object                   ... a Neural\_Network object  
number\_predictors  
                          ... a numeric telling how many predictors there are  
hidden\_layer\_lengths  
                          ... a numeric telling the number of layers and the number of neurons in each layer

**Details**

NN is parametrized by its connection\_strength matrices

**Value**

Neural\_Network

---

Neural\_Network-class *Neural Network implementation*

---

**Description**

Neural Network implementation

---

Predict,Neural\_Network,data.frame-method  
*predict stuff*

---

**Description**

returns predictions

**Usage**

```
## S4 method for signature 'Neural_Network,data.frame'  
Predict(object, dataset)
```

**Arguments**

object : a neural network  
dataset : a dataframe of features and observations

**Value**

Numeric

---

Start	<i>start script</i>
-------	---------------------

---

**Description**

main function that runs the interactive script

**Usage**

Start()

**Details**

takes your numerai training data and trains a neural network to your architectural specifications. provides you with the out of sample error offers to retrain with a new architecture or predict on a numerai tournament dataset. Can then write the predictions to a CSV

---

Train,Neural_Network,data.frame,numeric,numeric,numeric-method	<i>train the NN</i>
--	---------------------

---

**Description**

gets NN parameters that minimize cost on dataset using optimization\_method

**Usage**

```
## S4 method for signature 'Neural_Network,data.frame,numeric,numeric,numeric'
Train(object,
      dataset, regularization_constant, learning_rate, tolerable_error)
```

**Arguments**

object	is a Neural Network
dataset	is a data.frame, the original data frame that includes the target
regularization_constant	is a numeric
learning_rate	is a numeric
tolerable_error	is a numeric, units : log loss

**Value**

Neural\_Network

# Index

back\_propogation,Neural\_Network,numeric,numeric,numeric-method,  
[2](#)

forward\_propogation,Neural\_Network,matrix-method,  
[2](#)

Get\_Cost,Neural\_Network,numeric-method,  
[3](#)

Get\_LogLoss, [3](#)

Get\_Number\_Observations,Neural\_Network-method,  
[4](#)

initialize,Neural\_Network-method, [4](#)

Neural\_Network-class, [5](#)

Predict,Neural\_Network,data.frame-method,  
[5](#)

Start, [6](#)

Train,Neural\_Network,data.frame,numeric,numeric,numeric-method,  
[6](#)