# mlqp

## *Release 1.1*

LogCreative

Mar 08, 2022

# CONTENTS

# CHAPTER ONE

# MINMAX MODULE

**class** minmax.**Max**(*operands*)

Bases: *minmax.Operator*

Max module

**forward**(*src*)

Foward prediction for src input

**Parameters** **src** (*list*) – the input list [x,y]

**class** minmax.**Min**(*operands*)

Bases: *minmax.Operator*

Min module

**forward**(*src*)

Foward prediction for src input

**Parameters** **src** (*list*) – the input list [x,y]

**class** minmax.**Operator**(*operands*)

Bases: `object`

Base operator class for Min and Max module.

**forward**(*src*)

Foward prediction for src input

**Parameters** **src** (*list*) – the input list [x,y]

minmax.**divide**(*train_data*, *k=2*)

Divide the data into positive and negative merged 2D data array.

**Parameters**

- **train_data** (*array*) – the data to be divided
- **k** (*int*) – the number of split on positive/negative set

**Returns** the 2D divided data array for computation.

**Return type** array

minmax.**minmax**(*train_data*, *k*, *epochs*, *lr=0.05*, *random_seed=None*, *parallel=True*)

Train minmax network in multiprocessing.

**Parameters**

- **train_data** (*array*) – the training data.
- **k** (*int*) – the number of split

- **epochs** (*int*) – the threshold of training epochs.
- **lr** (*float, optional*) – Learning rate. Defaults to 0.05.
- **random_seed** (*int, optional*) – Random Seed. Defaults to None.
- **parallel** (*bool, optional*) – If uses parallel training. Defaults to True.

**Returns** Target Network, subnets, min nets, elapsed training time among units if parallel or max training time if not parallel.

**Return type** *Max*, array[*Net*], array[*Min*], float

minmax.**trainer**(*train_sub_data*, *epochs*, *lr=0.05*, *random_seed=None*)

Trainer worker

**Parameters**

- **train_sub_data** (*array*) – the input array for training.
- **epochs** (*int*) – the number threshold of epochs.
- **lr** (*float, optional*) – Learning rate. Defaults to 0.05.
- **random_seed** (*int, optional*) – Random Seed. Defaults to None.

**Returns** the trained network.

**Return type** *Net*

CHAPTER

TWO

# MODEL MODULE

**class** model.**Net**(*lr=0.05*, *alpha=0.8*, *random_seed=None*, *hidden_num=10*)

Bases: `object`

MLQP Network

**backward**(*pred*, *target*)

Backward pass, update the parameters. NOTE: should run forward pass first before calling this function.

**Parameters**

- **pred** (*float*) – prediction based on foward pass
- **target** (*float*) – the target label

**forward**(*src*)

Foward pass, return the prediction based on the given data.

**Parameters** **src** (*list*) – the input list of data *[x,y]*

**param_init**()

Init parameters.

model.**cross_validation**(*model*, *split_data*)

Cross validation over split data.

**Parameters**

- **model** (Net) – the instance of Net
- **split_data** (*array*) – the splitted data generated from folds()

**Returns** the mean of training error and validation error among experiments.

**Return type** float, float

model.**folds**(*data*, *k*)

divide data sequencially into k portions

**Parameters**

- **data** (*array*) – the data to be divided
- **k** (*int*) – the number of portions

**Returns** the divided data

**Return type** array

model.**split**(*train_data*, *k*)

split train_data into k folds.

**Parameters**

- **train_data** (*array*) – the training data
- **k** (*int*) – fold number

**Returns** the splited data formatted [train_data, val_data] array.

**Return type** array

`model.`**`step`**(*model*, *data*, *with_grad=True*)

Common step for data on training or testing.

**Parameters**

- **model** (Net) – the instance of Net
- **data** (*array*) – data for training or testing
- **with_grad** (*bool, optional*) – If it needs backward process. Defaults to True.

**Returns** the mse loss of this batch of data

**Return type** loss

`model.`**`test`**(*model*, *test_data*)

Test the model

**Parameters**

- **model** (Net) – the instance of Net
- **test_data** (*arrat*) – the testing set

**Returns** the mse error over test set

**Return type** float

`model.`**`test_step`**(*model*, *test_data*)

Test the model for test_data

**Parameters**

- **model** (Net) – the instance of Net
- **test_data** (*array*) – the testing data

**Returns** the mse error over test_data

**Return type** array

`model.`**`train`**(*model*, *train_data*, *epochs*, *test_data=None*)

Train the model by epochs.

**Parameters**

- **model** (Net) – the instance of Net
- **train_data** (*array*) – the training set
- **epochs** (*int*) – the number of epochs
- **test_data** (*array, optional*) – if assigned, the test error will be tracked but will not go into the training process.

**Returns** trained model

**Return type** *Net*

`model.`**`train_step`**(*model*, *train_data*)

Train the model for one step.

**Parameters**

- **model** (Net) – the instance of Net
- **train_data** (*array*) – the training data

CHAPTER

THREE

# UTIL MODULE

util.**mse**(*pred*, *target*)

util.**read_data**(*filename*)

Reads data from the file and return an array of data formatting: [x y label]

**Parameters** **filename** (*str*) – the path of file

**Returns** the array of the data read from file

**Return type** data

util.**sigmoid**(*x*)

util.**sigmoid_prime**(*x*)

## CHAPTER FOUR

# INDICES AND TABLES

- genindex
- modindex
- search

# PYTHON MODULE INDEX

# INDEX