
DBM Documentation

Release 1.2.1

Simon Lee

Mar 09, 2017

CONTENTS:

1	Classes	3
1.1	Matrix	3
1.2	Data Set	4
1.3	Parameters	5
1.4	Delta Boosting Machines	6
1.5	Delta Boosting Machines with Automatic BL Selection	8
2	Functions	11
	Index	13

This is the document of the Python APIs of Delta Boosting Machine. Classes and functions are listed and described.

CLASSES

Matrix

class dbm_py.interface.**Matrix** (*height=None, width=None, val=None, file_name=None, sep=None, mat=None*)

__init__ (*height=None, width=None, val=None, file_name=None, sep=None, mat=None*)

This is the class of Matrix used in DBM. To feed the training and prediction data to DBM, they should be converted to Matrix first of all. The Matrix interface provides four ways of initialization, i.e. initialization with random values in $[-1, 1]$, initialization with a user-provided value, initialization from a file and initialization with a Float_Matrix object. One may also initialize a matrix with any values and then use the method `from_np2darray` to transfer the values from a numpy array of the same shape to it.

Initialization with random values in $[-1, 1]$:param height: height of the matrix :param width: width of the matrix

Initialization with a user-provided value :param val: a particular value for initialization

Initialization from a file :param file_name: file name of the file where data comes from :param sep: separator used in the file

Initialization with a Float_Matrix object :param mat: a Float_Matrix object

Note: 1. When initializing from a file, the format should be correct. One may first of all save a matrix to a file and look at the file and see how it looks like. 2. Avoid directly using Float_Matrix. 3. Converting tools `np2darray_to_float_matrix` and `float_matrix_to_np2darray` are provided.

assign (*i, j, val*)

Assign a value to a particular element.

Parameters

- **i** – height of the element
- **j** – width of the element
- **val** – value to be assigned

clear ()

Set all elements to 0.

from_np2darray (*source*)

Assign the data stored in a two-dimensional numpy array to this matrix.

Parameters **source** – a two-dimensional numpy array of the same shape as this matrix

get (*i, j*)

Access to a particular element in the matrix.

Parameters

- **i** – height of the element
- **j** – width of the element

Returns the element

Note: i and j should be in the correct ranges

save (*file_name*, *sep*='\\')

Save the data stored in it to a file.

Parameters

- **file_name** – a string
- **sep** – a character

shape ()

Return a list containing the shape of the matrix.

Returns [matrix height, matrix width]

show ()

Print to screen the data stored in the matrix.

to_np2darray ()

Assign the data stored in this matrix to a two-dimensional numpy array and return it.

Returns a two-dimensional numpy array of the same shape as this matrix

Data Set

class dbm_py.interface.**Data_set** (*data_x*, *data_y*, *portion_for_validating*, *random_seed*=-1)

__init__ (*data_x*, *data_y*, *portion_for_validating*, *random_seed*=-1)

This is the class of Data_set that provides an easy to tool for splitting all data into training and validating parts.

Parameters

- **data_x** – a Matrix object
- **data_y** – a Matrix object
- **portion_for_validating** – percentage of the whole data used for validating
- **random_seed** – optional random seed (random if negative or fixed if non-negative)

get_train_x ()

Return the part of predictors for training.

Returns a Matrix object

get_train_y ()

Return the part of responses for training.

Returns a Matrix object

get_validate_x ()

Return the part of predictors for validating.

Returns a Matrix object

get_validate_y()

Return the part of responses for validating.

Returns a Matrix object

Parameters

Parameter Name	Type	Meaning
dbm_no_bunches_of_learners	int	number of bootstrapped BLs
dbm_no_candidate_feature	int	number of features for each BL (< total number of features)
dbm_portion_train_sample	double	percentage for training each BL
dbm_no_cores	int	number of BL in each bunch (number of cores used)
dbm_loss_function	char	(n)ormal, (b)ernoulli, (p)oisson or (t)weedie
dbm_display_training_progress	bool	whether to display training progress or not
dbm_record_every_tree	bool	whether to record trees in a file or not
dbm_freq_showing_loss_on_test	int	show loss on test after how many bunches of BLs
dbm_shrinkage	double	shrinkage for each BL
dbm_nonoverlapping_training	int	whether to BLs in a bunch use nonoverlapping samples or not
dbm_remove_rows_containing_nans	int	whether to remove rows containing NaNs in training every BL
dbm_min_no_samples_per_bl	int	minimal number of samples for trainin every BL
dbm_portion_for_trees	double	percentage of BLs using trees
dbm_random_seed	int	random seed (random < 0 and fixed >= 0)
dbm_portion_for_lr	double	percentage of BLs using linear regression
dbm_portion_for_s	double	percentage of BLs using splines
dbm_portion_for_k	double	percentage of BLs using k-means
dbm_portion_for_nn	double	should be 0
dbm_portion_for_d	double	percentage of BLs using dominating principal component stairs
dbm_accumulated_portion	double	unused
_shrinkage_for_selected_b		
dbm_portion_shrinkage_for_unselected_bl	double	unused
tweedie_p	double	p of tweedie should in (1, 2)
splines_no_knot	int	number of knots of splines
splines_portion_of_pairs	double	percentage of pairs of perdictors considered
splines_regularization	double	ridge regression penalty
splines_hinge_coefficient	double	coefficient in splines
kmeans_no_centroids	int	number of centroids
kmeans_max_iteration	int	max number of iterations of training
kmeans_tolerance	double	max tolerated error
kmeans_fraction_of_pairs	double	percentage of pairs of predictors considered
nn_no_hidden_neurons	int	number of hidden neurons

Continued on next page

Parameters **params** – a Params object

calibrate_plot (*observation, prediction, resolution, file_name=''*)

This is exactly the same as the one in GBM in R.

Parameters

- **observation** – a Matrix object
- **prediction** – a Matrix object
- **resolution** – a scalar
- **file_name** – save the result if provided

Returns a Matrix object

interact (*data, predictor_ind, total_no_predictor*)

This is exactly the same as the one in GBM in R.

Parameters

- **data** – a Matrix object
- **predictor_ind** – a Matrix object
- **total_no_predictor** – a scalar

Returns a scalar

load (*file_name*)

Load from a file.

Parameters **file_name** – a string

pdp (*data_x, feature_index*)

Calculate the data used in partial dependence plots.

Parameters

- **data_x** – a Matrix object used for calculating
- **feature_index** – the index of the predictor of interest (the No. of the column)

Returns a Matrix object storing the data used in partial dependence plots

predict (*data_x*)

Predict if it has been trained or it has been loaded from a trained model.

Parameters **data_x** – a Matrix object

Returns

save (*file_name*)

Save the DBM after trained.

Parameters **file_name** – a string

save_performance (*file_name*)

Save the training and validating losses.

Parameters **file_name** – a string

ss (*data_x*)

Calculate statistical significance of every predictor.

Parameters **data_x** – a Matrix object used for calculating

Returns a Matrix object storing P-values for every predictor

train (*data_set*)

Train the DBM.

Parameters **data_set** – a Data_set object

train_with_monotonic_constraints (*data_set, monotonic_constraints*)

Train the DBM with monotonic constraints. Negative values indicate a decreasing relationship and positive values indicates an increasing relationship.

Parameters

- **data_set** – a Data_set object
- **monotonic_constraints** – a Matrix object of the dimension $p \times 1$

Delta Boosting Machines with Automatic BL Selection

class dbm_py.interface.**AUTO_DBM** (*params*)

__init__ (*params*)

This is the class of DBM.

Parameters **params** – a Params object

calibrate_plot (*observation, prediction, resolution, file_name*)

This is exactly the same as the one in GBM in R.

Parameters

- **observation** – a Matrix object
- **prediction** – a Matrix object
- **resolution** – a scalar
- **file_name** – save the result if provided

Returns a Matrix object

interact (*data, predictor_ind, total_no_predictor*)

This is exactly the same as the one in GBM in R.

Parameters

- **data** – a Matrix object
- **predictor_ind** – a Matrix object
- **total_no_predictor** – a scalar

Returns a scalar

load (*file_name*)

Load from a file.

Parameters **file_name** – a string

pdp (*data_x, feature_index*)

Calculate the data used in partial dependence plots.

Parameters

- **data_x** – a Matrix object used for calculating

- **feature_index** – the index of the predictor of interest (the No. of the column)

Returns a Matrix object storing the data used in partial dependence plots

predict (*data_x*)

Predict if it has been trained or it has been loaded from a trained model.

Parameters **data_x** – a Matrix object

Returns

save (*file_name*)

Save the DBM after trained.

Parameters **file_name** – a string

save_performance (*file_name*)

Save the training and validating losses.

Parameters **file_name** – a string

ss (*data_x*)

Calculate statistical significance of every predictor.

Parameters **data_x** – a Matrix object used for calculating

Returns a Matrix object storing P-values for every predictor

train (*data_set*)

Train the DBM.

Parameters **data_set** – a Data_set object

train_with_monotonic_constraints (*data_set, monotonic_constraints*)

Train the DBM with monotonic constraints. Negative values indicate a decreasing relationship, positive values indicates an increasing relationship and zero indicates no constraints.

Parameters

- **data_set** – a Data_set object
- **monotonic_constraints** – a Matrix object of the dimension $p \times 1$

FUNCTIONS

`dbm_py.interface.np2darray_to_float_matrix(source)`

Convert a two-dimensional numpy array to a Matrix.

Parameters **source** – a two-dimensional numpy array

Returns a Matrix object of the same shape as the numpy array

`dbm_py.interface.float_matrix_to_np2darray(source)`

Convert a Matrix to a two-dimensional numpy array.

Parameters **source** – a Matrix object

Returns a two-dimensional numpy array of the same shape as the Matrix

`dbm_py.interface.string_to_params(string, sep=' ')`

Directly transfer a string to a Params object.

Parameters

- **string** – a string
- **sep** – a character

Returns a Params object

Symbols

__init__() (dbm_py.interface.AUTO_DBM method), 8
 __init__() (dbm_py.interface.DBM method), 6
 __init__() (dbm_py.interface.Data_set method), 4
 __init__() (dbm_py.interface.Matrix method), 3
 __init__() (dbm_py.interface.Params method), 6

A

assign() (dbm_py.interface.Matrix method), 3
 AUTO_DBM (class in dbm_py.interface), 8

C

calibrate_plot() (dbm_py.interface.AUTO_DBM method), 8
 calibrate_plot() (dbm_py.interface.DBM method), 7
 clear() (dbm_py.interface.Matrix method), 3

D

Data_set (class in dbm_py.interface), 4
 DBM (class in dbm_py.interface), 6

F

float_matrix_to_np2darray() (in module dbm_py.interface), 11
 from_np2darray() (dbm_py.interface.Matrix method), 3

G

get() (dbm_py.interface.Matrix method), 3
 get_train_x() (dbm_py.interface.Data_set method), 4
 get_train_y() (dbm_py.interface.Data_set method), 4
 get_validate_x() (dbm_py.interface.Data_set method), 4
 get_validate_y() (dbm_py.interface.Data_set method), 4

I

interact() (dbm_py.interface.AUTO_DBM method), 8
 interact() (dbm_py.interface.DBM method), 7

L

load() (dbm_py.interface.AUTO_DBM method), 8
 load() (dbm_py.interface.DBM method), 7

M

Matrix (class in dbm_py.interface), 3

N

np2darray_to_float_matrix() (in module dbm_py.interface), 11

P

Params (class in dbm_py.interface), 6
 pdp() (dbm_py.interface.AUTO_DBM method), 8
 pdp() (dbm_py.interface.DBM method), 7
 predict() (dbm_py.interface.AUTO_DBM method), 9
 predict() (dbm_py.interface.DBM method), 7
 print_all() (dbm_py.interface.Params method), 6

S

save() (dbm_py.interface.AUTO_DBM method), 9
 save() (dbm_py.interface.DBM method), 7
 save() (dbm_py.interface.Matrix method), 4
 save_performance() (dbm_py.interface.AUTO_DBM method), 9
 save_performance() (dbm_py.interface.DBM method), 7
 set_params() (dbm_py.interface.Params method), 6
 shape() (dbm_py.interface.Matrix method), 4
 show() (dbm_py.interface.Matrix method), 4
 ss() (dbm_py.interface.AUTO_DBM method), 9
 ss() (dbm_py.interface.DBM method), 7
 string_to_params() (in module dbm_py.interface), 11

T

to_np2darray() (dbm_py.interface.Matrix method), 4
 train() (dbm_py.interface.AUTO_DBM method), 9
 train() (dbm_py.interface.DBM method), 7
 train_with_monotonic_constraints() (dbm_py.interface.AUTO_DBM method), 9
 train_with_monotonic_constraints() (dbm_py.interface.DBM method), 8