Touvlo Documentation

Benardi Nunes

Oct 21, 2019

Contents

1 Contents **3**
- 1.1 Supervised learning . . . 3
- 1.2 Unsupervised learning . . . 9
- 1.3 Recommender Systems . . . 13
- 1.4 Utils . . . 14

2 Indices and tables **17**

Python Module Index **19**

Index **21**

Welcome to touvlo's documentation!

CHAPTER 1

Contents

1.1 Supervised learning

1.1.1 Linear Regression routines

touvlo.supv.lin_rg.**cost_func**(*X*, *y*, *theta*)
Computes the cost function J for Linear Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns Computed cost.

Return type float

touvlo.supv.lin_rg.**grad**(*X*, *y*, *theta*)
Computes the gradient for Linear Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns Gradient column vector.

Return type numpy.array

touvlo.supv.lin_rg.**h**(*X*, *theta*)
Linear regression hypothesis.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns The projected value for each line of the dataset.

Return type numpy.array

touvlo.supv.lin_rg.**normal_eqn**(*X*, *y*)
Produces optimal theta via normal equation.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.

Raises LinAlgError

Returns Optimized model parameters theta.

Return type numpy.array

touvlo.supv.lin_rg.**predict**(*X*, *theta*)
Computes prediction vector.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns vector with predictions for each input line.

Return type numpy.array

touvlo.supv.lin_rg.**reg_cost_func**(*X*, *y*, *theta*, *_lambda*)
Computes the regularized cost function J for Linear Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Computed cost with regularization.

Return type float

touvlo.supv.lin_rg.**reg_grad**(*X*, *y*, *theta*, *_lambda*)
Computes the regularized gradient for Linear Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Regularized gradient column vector.

Return type numpy.array

1.1.2 Logistic Regression routines

touvlo.supv.lgx_rg.**cost_func**(*X*, *y*, *theta*)
Computes the cost function J for Logistic Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns Computed cost.

Return type float

touvlo.supv.lgx_rg.**grad**(*X*, *y*, *theta*)
Computes the gradient for the parameters theta.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns Gradient column vector.

Return type numpy.array

touvlo.supv.lgx_rg.**h**(*X*, *theta*)
Logistic regression hypothesis.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Raises ValueError

Returns The probability that each entry belong to class 1.

Return type numpy.array

touvlo.supv.lgx_rg.**p**(*x*, *threshold=0.5*)
Predicts whether a probability falls into class 1.

Parameters

- **x** (*obj*) – Probability that example belongs to class 1.
- **threshold** (*float*) – point above which a probability is deemed of class 1.

Returns Binary value to denote class 1 or 0

Return type int

touvlo.supv.lgx_rg.**predict**(*X*, *theta*)
Classifies each entry as class 1 or class 0.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **theta** (*numpy.array*) – Column vector of model's parameters.

Returns Column vector with each entry classification.

Return type numpy.array

touvlo.supv.lgx_rg.**predict_prob**(*X*, *theta*)
Produces the probability that the entries belong to class 1.

Returns Features' dataset plus bias column. theta (numpy.array): Column vector of model's parameters.

Return type X (numpy.array)

Raises ValueError

Returns The probability that each entry belong to class 1.

Return type numpy.array

touvlo.supv.lgx_rg.**reg_cost_func**(*X*, *y*, *theta*, *_lambda*)
Computes the regularized cost function J for Logistic Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Computed cost with regularization.

Return type float

touvlo.supv.lgx_rg.**reg_grad**(*X*, *y*, *theta*, *_lambda*)
Computes the regularized gradient for Logistic Regression.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Regularized gradient column vector.

Return type numpy.array

1.1.3 Classification Neural Network routines

touvlo.supv.nn_clsf.**back_propagation**(*y*, *theta*, *a*, *z*, *num_labels*, *n_hidden_layers=1*)
Applies back propagation to minimize model's loss.

Parameters

- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array (numpy.array)*) – array of model's weight matrices by layer.
- **a** (*numpy.array (numpy.array)*) – array of activation matrices by layer.
- **z** (*numpy.array (numpy.array)*) – array of parameters prior to sigmoid by layer.
- **num_labels** (*int*) – Number of classes in multiclass classification.

- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns array of matrices of 'error values' by layer.

Return type numpy.array(numpy.array)

touvlo.supv.nn_clsf.**cost_function**(*X*, *y*, *theta*, *_lambda*, *num_labels*, *n_hidden_layers=1*)

Computes the cost function J for Neural Network.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **y** (*numpy.array*) – Column vector of expected values.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.
- **num_labels** (*int*) – Number of classes in multiclass classification.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns Computed cost.

Return type float

touvlo.supv.nn_clsf.**feed_forward**(*X*, *theta*, *n_hidden_layers=1*)

Applies forward propagation to calculate model's hypothesis.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns

A 2-tuple consisting of an array of parameters prior to activation by layer and an array of activation matrices by layer.

Return type (numpy.array(numpy.array), numpy.array(numpy.array))

touvlo.supv.nn_clsf.**grad**(*X*, *y*, *nn_params*, *_lambda*, *input_layer_size*, *hidden_layer_size*, *num_labels*, *n_hidden_layers=1*)

Calculates gradient of neural network's parameters.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **y** (*numpy.array*) – Column vector of expected values.
- **nn_params** (*numpy.array*) – Column vector of model's parameters.
- **_lambda** (*float*) – The regularization hyperparameter.
- **input_layer_size** (*int*) – Number of units in the input layer.
- **hidden_layer_size** (*int*) – Number of units in a hidden layer.
- **num_labels** (*int*) – Number of classes in multiclass classification.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns array of gradient values by weight matrix.

Return type numpy.array(numpy.array)

touvlo.supv.nn_clsf.**h**(*X*, *theta*, *n_hidden_layers=1*)
Classification Neural Network hypothesis.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **theta** (*numpy.array*) – Column vector of model's parameters.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns The probability that each entry belong to class 1.

Return type numpy.array

touvlo.supv.nn_clsf.**init_nn_weights**(*input_layer_size*, *hidden_layer_size*, *num_labels*, *n_hidden_layers=1*)
Initialize the weight matrices of a network with random values.

Parameters

- **hidden_layer_size** (*int*) – Number of units in a hidden layer.
- **input_layer_size** (*int*) – Number of units in the input layer.
- **num_labels** (*int*) – Number of classes in multiclass classification.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns array of weight matrices of random values.

Return type numpy.array(numpy.array)

touvlo.supv.nn_clsf.**rand_init_weights**(*L_in*, *L_out*)
Initializes weight matrix with random values.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **L_in** (*int*) – Number of units in previous layer.
- **n_hidden_layers** (*int*) – Number of units in next layer.

Returns Random values' matrix of conforming dimensions.

Return type numpy.array

touvlo.supv.nn_clsf.**unravel_params**(*nn_params*, *input_layer_size*, *hidden_layer_size*, *num_labels*, *n_hidden_layers=1*)
Unravels flattened array into list of weight matrices

Parameters

- **nn_params** (*numpy.array*) – Row vector of model's parameters.
- **input_layer_size** (*int*) – Number of units in the input layer.
- **hidden_layer_size** (*int*) – Number of units in a hidden layer.
- **num_labels** (*int*) – Number of classes in multiclass classification.
- **n_hidden_layers** (*int*) – Number of hidden layers in network.

Returns array with model's weight matrices.

Return type numpy.array(numpy.array)

1.2 Unsupervised learning

1.2.1 PCA

touvlo.unsupv.pca.**pca**(*X*)
Runs Principal Component Analysis on dataset

Parameters **X** (*numpy.array*) – Features' dataset

Returns

A 2-tuple of U, eigenvectors of covariance matrix, and S, eigenvalues (on diagonal) of covariance matrix.

Return type (numpy.array, numpy.array)

touvlo.unsupv.pca.**project_data**(*X*, *U*, *k*)
Computes reduced data representation (projected data)

Parameters

- **X** (*numpy.array*) – Normalized features' dataset
- **U** (*numpy.array*) – eigenvectors of covariance matrix
- **k** (*int*) – Number of features in reduced data representation

Returns Reduced data representation (projection)

Return type numpy.array

touvlo.unsupv.pca.**recover_data**(*Z*, *U*, *k*)
Recovers an approximation of original data using the projected data

Parameters

- **Z** (*numpy.array*) – Reduced data representation (projection)
- **U** (*numpy.array*) – eigenvectors of covariance matrix
- **k** (*int*) – Number of features in reduced data representation

Returns Approximated features' dataset

Return type numpy.array

1.2.2 K-means

touvlo.unsupv.kmeans.**compute_centroids**(*X*, *idx*, *K*)
Computes centroids from the mean of its cluster's members.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **idx** (*numpy.array*) – Column vector of assigned centroids' indices.
- **K** (*int*) – Number of centroids.

Returns Column vector of newly computed centroids

Return type numpy.array

touvlo.unsupv.kmeans.**cost_function**(*X*, *idx*, *centroids*)
Calculates the cost function for K means.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **idx** (*numpy.array*) – Column vector of assigned centroids' indices.

Returns Computed cost

Return type float

touvlo.unsupv.kmeans.**elbow_method**(*X*, *K_values*, *max_iters*, *n_inits*)
Calculates the cost for each given K.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **K_values** (*list(int)*) – List of possible number of centroids.
- **max_iters** (*int*) – Number of times the algorithm will be fitted.
- **n_inits** (*int*) – Number of random initialization.

Returns

A 2-tuple of K_values, a list of possible numbers of centroids, and cost_values, a computed cost for each K.

Return type (list(int), list(float))

touvlo.unsupv.kmeans.**euclidean_dist**(*p*, *q*)
Calculates Euclidean distance between 2 n-dimensional points.

Parameters

- **p** (*numpy.array*) – First n-dimensional point.
- **q** (*numpy.array*) – Second n-dimensional point.

Returns Distance between 2 points.

Return type float

touvlo.unsupv.kmeans.**find_closest_centroids**(*X*, *initial_centroids*)
Assigns to each example the indice of the closest centroid.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **initial_centroids** (*numpy.array*) – List of initialized centroids.

Returns Column vector of assigned centroids' indices.

Return type numpy.array

touvlo.unsupv.kmeans.**init_centroids**(*X*, *K*)
Computes centroids from the mean of its cluster's members.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **idx** (*numpy.array*) – Column vector of assigned centroids' indices.
- **K** (*int*) – Number of centroids.

Returns Column vector of centroids randomly picked from dataset

Return type numpy.array

touvlo.unsupv.kmeans.**run_intensive_kmeans**(*X*, *K*, *max_iters*, *n_inits*)
Applies kmeans using multiple random initializations.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **K** (*int*) – Number of centroids.
- **max_iters** (*int*) – Number of times the algorithm will be fitted.
- **n_inits** (*int*) – Number of random initialization.

Returns

A 2-tuple of centroids, a column vector of centroids, and idx, a column vector of assigned centroids' indices.

Return type (numpy.array, numpy.array)

touvlo.unsupv.kmeans.**run_kmeans**(*X*, *K*, *max_iters*)
Applies kmeans using a single random initialization.

Parameters

- **X** (*numpy.array*) – Features' dataset
- **K** (*int*) – Number of centroids.
- **max_iters** (*int*) – Number of times the algorithm will be fitted.

Returns

A 2-tuple of centroids, a column vector of centroids, and idx, a column vector of assigned centroids' indices.

Return type (numpy.array, numpy.array)

1.2.3 Anomaly Detection

touvlo.unsupv.anmly_detc.**cov_matrix**(*X*, *mu*)
Calculates the covariance matrix for matrix X (m x n).

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **mu** (*numpy.array*) – Mean of each feature/column of.

Returns Covariance matrix (n x n)

Return type int

touvlo.unsupv.anmly_detc.**estimate_multi_gaussian**(*X*)
Estimates parameters for Multivariate Gaussian distribution.

Parameters **X** (*numpy.array*) – Features' dataset.

Returns

A 2-tuple of mu, the mean of each feature/column of X, and sigma, the covariance matrix for X.

Return type (numpy.array, numpy.array)

touvlo.unsupv.anmly_detc.**estimate_uni_gaussian**(*X*)
Estimates parameters for Univariate Gaussian distribution.

Parameters **X** (*numpy.array*) – Features' dataset.

Returns

A 2-tuple of mu, the mean of each feature/column of X, and sigma2, the variance of each feature/column of X.

Return type (numpy.array, numpy.array)

touvlo.unsupv.anmly_detc.**is_anomaly**(*p*, *threshold=0.5*)
Predicts whether a probability falls into class 1 (anomaly).

Parameters

- **p** (*numpy.array*) – Probability that example belongs to class 1 (is anomaly).
- **threshold** (*float*) – point below which an example is considered of class 1.

Returns Binary value to denote class 1 or 0

Return type int

touvlo.unsupv.anmly_detc.**multi_gaussian**(*X*, *mu*, *sigma*)
Estimates probability that examples belong to Multivariate Gaussian.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **mu** (*numpy.array*) – Mean of each feature/column of X.
- **sigma** (*numpy.array*) – Covariance matrix for X.

Returns Probability density function for each example

Return type numpy.array

touvlo.unsupv.anmly_detc.**predict**(*X*, *epsilon*, *gaussian*, ***kwargs*)
Predicts whether examples are anomalies.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **epsilon** (*float*) – point below which an example is considered of class 1.
- **gaussian** (*numpy.array*) – Function that estimates pertinency probability.

Returns Column vector of classification

Return type numpy.array

touvlo.unsupv.anmly_detc.**uni_gaussian**(*X*, *mu*, *sigma2*)
Estimates probability that examples belong to Univariate Gaussian.

Parameters

- **X** (*numpy.array*) – Features' dataset.
- **mu** (*numpy.array*) – Mean of each feature/column of X.
- **sigma2** (*numpy.array*) – Variance of each feature/column of X.

Returns Probability density function for each example

Return type numpy.array

1.3 Recommender Systems

1.3.1 Collaborative Filtering

touvlo.rec_sys.cf.**cost_function**(*X*, *Y*, *R*, *theta*, *_lambda*)
Computes the cost function J for Collaborative Filtering.

Parameters

- **X** (*numpy.array*) – Matrix of product features.
- **Y** (*numpy.array*) – Scores' matrix.
- **R** (*numpy.array*) – Matrix of 0s and 1s (whether there's a rating).
- **theta** (*numpy.array*) – Matrix of user features.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Computed cost.

Return type float

touvlo.rec_sys.cf.**grad**(*params*, *Y*, *R*, *num_users*, *num_products*, *num_features*, *_lambda*)
Calculates gradient of Collaborative Filtering's parameters

Parameters

- **params** (*numpy.array*) – flattened product and user features..
- **Y** (*numpy.array*) – Scores' matrix.
- **R** (*numpy.array*) – Matrix of 0s and 1s (whether there's a rating).
- **num_users** (*int*) – Number of users in this instance.
- **num_products** (*int*) – Number of products in this instance.
- **num_features** (*int*) – Number of features in this instance.
- **_lambda** (*float*) – The regularization hyperparameter.

Returns Flattened gradient of product and user parameters.

Return type numpy.array

touvlo.rec_sys.cf.**unravel_params**(*params*, *num_users*, *num_products*, *num_features*)
Unravels flattened array into features' matrices

Parameters

- **params** (*numpy.array*) – Row vector of coefficients.
- **num_users** (*int*) – Number of users in this instance.
- **num_products** (*int*) – Number of products in this instance.
- **num_features** (*int*) – Number of features in this instance.

Returns A 2-tuple consisting of a matrix of product features and a matrix of user features.

Return type (numpy.array, numpy.array)

1.4 Utils

touvlo.utils.**BGD**(*X*, *y*, *grad*, *initial_theta*, *alpha*, *num_iters*, ***kwargs*)
Performs parameter optimization via Batch Gradient Descent.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **grad** (*numpy.array*) – Routine that generates the partial derivatives given theta.
- **initial_theta** (*numpy.array*) – Initial value for parameters to be optimized.
- **alpha** (*float*) – Learning rate or _step size of the optimization.
- **num_iters** (*int*) – Number of times the optimization will be performed.

Returns Optimized model parameters.

Return type numpy.array

touvlo.utils.**MBGD**(*X*, *y*, *grad*, *initial_theta*, *alpha*, *num_iters*, *b*, ***kwargs*)
Performs parameter optimization via Mini-Batch Gradient Descent.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **grad** (*numpy.array*) – Routine that generates the partial derivatives given theta.
- **initial_theta** (*numpy.array*) – Initial value for parameters to be optimized.
- **alpha** (*float*) – Learning rate or _step size of the optimization.
- **num_iters** (*int*) – Number of times the optimization will be performed.
- **b** (*int*) – Number of examples in mini batch.

Returns Optimized model parameters.

Return type numpy.array

touvlo.utils.**SGD**(*X*, *y*, *grad*, *initial_theta*, *alpha*, *num_iters*, ***kwargs*)
Performs parameter optimization via Stochastic Gradient Descent.

Parameters

- **X** (*numpy.array*) – Features' dataset plus bias column.
- **y** (*numpy.array*) – Column vector of expected values.
- **grad** (*numpy.array*) – Routine that generates the partial derivatives given theta.
- **initial_theta** (*numpy.array*) – Initial value for parameters to be optimized.
- **alpha** (*float*) – Learning rate or _step size of the optimization.
- **num_iters** (*int*) – Number of times the optimization will be performed.

Returns Optimized model parameters.

Return type numpy.array

touvlo.utils.**feature_normalize**(*X*)
Performs Z score normalization in a numeric dataset.

Parameters **X** (*numpy.array*) – Features' dataset plus bias column.

Returns

A 3-tuple of X_norm, normalized features' dataset, mu, mean of each feature, and sigma, standard deviation of each feature.

Return type (numpy.array, numpy.array, numpy.array)

touvlo.utils.**g**(*x*)
This function applies the sigmoid function on a given value.

Parameters **x** (*obj*) – Input value or object containing value .

Returns Sigmoid function at value.

Return type obj

touvlo.utils.**g_grad**(*x*)
This function calculates the sigmoid gradient at a given value.

Parameters **x** (*obj*) – Input value or object containing value .

Returns Sigmoid gradient at value.

Return type obj

touvlo.utils.**mean_normlztn**(*Y*, *R*)
Performs mean normalization in a numeric dataset.

Parameters

- **Y** (*numpy.array*) – Scores' dataset.
- **R** (*numpy.array*) – Dataset of 0s and 1s (whether there's a rating).

Returns

- Y_norm - Normalized scores' dataset (row wise).
- Y_mean - Column vector of calculated means.

Return type

- Y_norm (:py:class: numpy.array)
- Y_mean (:py:class: numpy.array)

touvlo.utils.**numerical_grad**(*J*, *theta*, *err*)
Numerically calculates the gradient of a given cost function.

Parameters

- **J** (*Callable*) – Function handle that computes cost given theta.
- **theta** (*numpy.array*) – Model parameters.
- **err** (*float*) – distance between points where J is evaluated.

Returns Computed numeric gradient.

Return type numpy.array

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

anmly_detc, 11

c

cf, 13

k

kmeans, 9

l

lgx_rg, 5
lin_rg, 3

n

nn_clsf, 6

p

pca, 9

t

touvlo.rec_sys.cf, 13
touvlo.supv.lgx_rg, 5
touvlo.supv.lin_rg, 3
touvlo.supv.nn_clsf, 6
touvlo.unsupv.anmly_detc, 11
touvlo.unsupv.kmeans, 9
touvlo.unsupv.pca, 9
touvlo.utils, 14

u

utils, 14

Index

A

anmly_detc (*module*), 11

B

back_propagation() (*in module touvlo.supv.nn_clsf*), 6
BGD() (*in module touvlo.utils*), 14

C

cf (*module*), 13
compute_centroids() (*in module touvlo.unsupv.kmeans*), 9
cost_func() (*in module touvlo.supv.lgx_rg*), 5
cost_func() (*in module touvlo.supv.lin_rg*), 3
cost_function() (*in module touvlo.rec_sys.cf*), 13
cost_function() (*in module touvlo.supv.nn_clsf*), 7
cost_function() (*in module touvlo.unsupv.kmeans*), 9
cov_matrix() (*in module touvlo.unsupv.anmly_detc*), 11

E

elbow_method() (*in module touvlo.unsupv.kmeans*), 10
estimate_multi_gaussian() (*in module touvlo.unsupv.anmly_detc*), 11
estimate_uni_gaussian() (*in module touvlo.unsupv.anmly_detc*), 11
euclidean_dist() (*in module touvlo.unsupv.kmeans*), 10

F

feature_normalize() (*in module touvlo.utils*), 14
feed_forward() (*in module touvlo.supv.nn_clsf*), 7
find_closest_centroids() (*in module touvlo.unsupv.kmeans*), 10

G

g() (*in module touvlo.utils*), 15
g_grad() (*in module touvlo.utils*), 15
grad() (*in module touvlo.rec_sys.cf*), 13
grad() (*in module touvlo.supv.lgx_rg*), 5
grad() (*in module touvlo.supv.lin_rg*), 3
grad() (*in module touvlo.supv.nn_clsf*), 7

H

h() (*in module touvlo.supv.lgx_rg*), 5
h() (*in module touvlo.supv.lin_rg*), 3
h() (*in module touvlo.supv.nn_clsf*), 7

I

init_centroids() (*in module touvlo.unsupv.kmeans*), 10
init_nn_weights() (*in module touvlo.supv.nn_clsf*), 8
is_anomaly() (*in module touvlo.unsupv.anmly_detc*), 12

K

kmeans (*module*), 9

L

lgx_rg (*module*), 5
lin_rg (*module*), 3

M

MBGD() (*in module touvlo.utils*), 14
mean_normlztn() (*in module touvlo.utils*), 15
multi_gaussian() (*in module touvlo.unsupv.anmly_detc*), 12

N

nn_clsf (*module*), 6
normal_eqn() (*in module touvlo.supv.lin_rg*), 4
numerical_grad() (*in module touvlo.utils*), 15

P

p() (*in module touvlo.supv.lgx_rg*), 5

pca (*module*), 9
pca() (*in module touvlo.unsupv.pca*), 9
predict() (*in module touvlo.supv.lgx_rg*), 5
predict() (*in module touvlo.supv.lin_rg*), 4
predict() (*in module touvlo.unsupv.anmly_detc*), 12
predict_prob() (*in module touvlo.supv.lgx_rg*), 6
project_data() (*in module touvlo.unsupv.pca*), 9

R

rand_init_weights() (*in module touvlo.supv.nn_clsf*), 8
recover_data() (*in module touvlo.unsupv.pca*), 9
reg_cost_func() (*in module touvlo.supv.lgx_rg*), 6
reg_cost_func() (*in module touvlo.supv.lin_rg*), 4
reg_grad() (*in module touvlo.supv.lgx_rg*), 6
reg_grad() (*in module touvlo.supv.lin_rg*), 4
run_intensive_kmeans() (*in module touvlo.unsupv.kmeans*), 10
run_kmeans() (*in module touvlo.unsupv.kmeans*), 11

S

SGD() (*in module touvlo.utils*), 14

T

touvlo.rec_sys.cf (*module*), 13
touvlo.supv.lgx_rg (*module*), 5
touvlo.supv.lin_rg (*module*), 3
touvlo.supv.nn_clsf (*module*), 6
touvlo.unsupv.anmly_detc (*module*), 11
touvlo.unsupv.kmeans (*module*), 9
touvlo.unsupv.pca (*module*), 9
touvlo.utils (*module*), 14

U

uni_gaussian() (*in module touvlo.unsupv.anmly_detc*), 12
unravel_params() (*in module touvlo.rec_sys.cf*), 13
unravel_params() (*in module touvlo.supv.nn_clsf*), 8
utils (*module*), 14