Information-Theretic Metric Learning

Davis, J. V., Kulis, B., Jain, P., Sra, S., & Dhillon, I. S. Information-theoretic metric learning.(ICML 2007)

Implemented by Junjie Hu (jjhu@cse.cuhk.edu.hk)

Sun Jul 6 2014 05:11:37

This is the C++ implementation of ITML. To use this C++ version, you need to install the C++ boost library, which is very popular in std C++ users. You can install it from its website. http://www.boost.org/

Copy all the files to the Linux machine. It is tested on the Ubuntu 12.04.The result is exactly the same as the Matlab version, but much faster. Run the demo:
(1) cmake .
(2) make
(3) ./demo
Then you will find the result in the ./dataset folder.

Contents

Chapter 1

Generate the Constraints

ITML C++ version, implemented by Junjie Hu. CreateDate 2014-07-05

Version

0.1

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 dml_problem Struct Reference

the struct to store the raw data for the Distance Metric Learning

`#include <itml.h>`

Public Attributes

- matrix< float > **X**
- vector< float > **y**
- matrix< float > **C**
- vector< float > **label**

3.1.1 Detailed Description

the struct to store the raw data for the Distance Metric Learning

The documentation for this struct was generated from the following file:

- itml.h

3.2 ITML Class Reference

ITML class containing all the supporting functions.

`#include <itml.h>`

Public Member Functions

- void itml_alg (const matrix< float > &C, const matrix< float > &X, const matrix< float > &M_0, const itml_param &param, matrix< float > &M)

  The main function of ITML.
- void KNN (const vector< float > &y, const matrix< float > &X, const matrix< float > &M, int k, const matrix< float > &Xt, vector< float > &pred_y)
- void ComputeDistanceExtreme (const matrix< float > &X, int lpercent, int upercent, const matrix< float > &M, float &l, float &u)

  This function is to compute the distance extreme value for the similar/dissimilar pairs.
- void ComputeDistanceExtreme (const matrix< float > &X, int lpercent, int upercent, float &l, float &u)

This function is to compute the distance extreme value for the similar/dissimilar pairs.

- void GetConstraints (const vector< float > &y, int contraints_num, float l, float u, matrix< float > &C)

  This function is to generate the similar/dissimilar pairs of constraints.

3.2.1 Detailed Description

ITML class containing all the supporting functions.

3.2.2 Member Function Documentation

3.2.2.1 void ITML::ComputeDistanceExtreme (const matrix< float > & *X*, int *lpercent*, int *upercent*, const matrix< float > & *M*, float & *l*, float & *u*)

This function is to compute the distance extreme value for the similar/dissimilar pairs.

Compute the distance extreme for similar/dissimilar pairs Input:

Parameters

X	nxd matrix, n samples with d dimensions
lpercent	the lower percent of the distance range
upercent	the upper percent of the distance range
M	dxd matrix, the learned matrix that is returned as a reference parameter
l	the lower bound for the constrains
u	the upper bound for the constrains
X	nxd matrix, n samples with d dimensions
lpercent	lower percent of sorted distance
upercent	upper percent of sorted distance
M	dxd matrix, distance matrix Output:

Returns

l lower extreme
u upper extreme

3.2.2.2 void ITML::ComputeDistanceExtreme (const matrix< float > & *X*, int *lpercent*, int *upercent*, float & *l*, float & *u*)

This function is to compute the distance extreme value for the similar/dissimilar pairs.

Parameters

X	nxd matrix, n samples with d dimensions
lpercent	the lower percent of the distance range
upercent	the upper percent of the distance range
l	the lower bound for the constrains
u	the upper bound for the constrains

Input: X: nxd matrix, n samples with d dimensions lpercent: lower percent of sorted distance upercent: upper percent of sorted distance Output: l: lower extreme u: upper extreme

3.2.2.3 void ITML::GetConstraints (const vector< float > & *y*, int *constraints_num*, float *l*, float *u*, matrix< float > & *C*)

This function is to generate the similar/dissimilar pairs of constraints.

Parameters

y	1xn vector, the label
contraints_num	int, the number of comstraints
l	float, the lower bound for the constrains
u	float, the upper bound for the constrains
C	mx4 matrix, 1:x1 2:x2 3:+/-1 similar/disimilar 4:l/u lower/upper bound

Input: y: the class labels of the samples constrains_num: number of the generated constraints l: lower extreme u: upper extreme Output: C: mx4 matrix, 1:x1 2:x2 3:+/-1 similar/disimilar 4:l/u lower/upper bound

Parameters

y	the class labels of the samples
constraints_num	number of the generated constraints
l	lower extreme
u	upper extreme
C	mx4 matrix, 1:x1 2:x2 3:+/-1 similar/disimilar 4:l/u lower/upper bound

3.2.2.4 void ITML::itml_alg (const matrix< float > & *C*, const matrix< float > & *X*, const matrix< float > & *A_0*, const itml_param & *param*, matrix< float > & *A*)

The main function of ITML.

Parameters

C	mx4 matrix, 1:x1 2:x2 3:+/-1 similar/disimilar 4:l/u lower/upper bound
X	nxd matrix, n samples with d dimensions
M_0	dxd matrix, identity matrix
param	itml paramers
M	dxd matrix, the learned matrix that is returned as a reference parameter

Input: C: mx4 matrix, 1:x1 2:x2 3:+/-1 similar/disimilar 4:l/u lower/upper bound X: nxd matrix, n samples with d dimensions A_0: dxd matrix, identity matrix param: itml paramers Output: A: dxd matrix, the learned matrix

3.2.2.5 void ITML::KNN (const vector< float > & *y*, const matrix< float > & *X*, const matrix< float > & *M*, int *k*, const matrix< float > & *Xt*, vector< float > & *pred_y*)

Parameters

y	1xn vector, the label
X	nxd matrix, n samples with d dimensions
M	dxd matrix, the learned matrix that is returned as a reference parameter
k	the number of the nearest neighbors
Xt	mxd matrix, m testing samples with d dimensions
pred_y	1xm vector, the predicted label for testing samples (return)

Input: y: the class labels of the samples X: n1xd matrix, n1 training samples with d dimensions M: dxd matrix, distance matrix k: the k-nearest neighbors Xt: n2xd matrix, n2 testing samples with d dimensions Output: pred_y: 1xn2 vector, predicted labels for the testing samples

The documentation for this class was generated from the following files:

- itml.h
- itml.cpp

3.3 itml_param Struct Reference

the struct storing the parameters of ITML

`#include <itml.h>`

Public Attributes

- float **thresh**
- float **gamma**
- int **max_iters**

3.3.1 Detailed Description

the struct storing the parameters of ITML

The documentation for this struct was generated from the following file:

- itml.h