MatrixFactorizationModel

NAME

MatrixFactorizationModel - A data structure used in modeling the output of the frovedis server side matrix factorization using ALS algorithm, at client spark side.

SYNOPSIS

import com.nec.frovedis.mllib.recommendation.MatrixFactorizationModel

Public Member Functions

Double predict (Int uid, Int pid)
`RDD[Rating]` predict (`RDD[(Int, Int)]` usersProducts)
`Array[Rating]` recommendProducts(Int uid, Int num)
`Array[Rating]` recommendUsers(Int pid, Int num)
Unit save(String path)
Unit save(SparkContext sc, String path)
MatrixFactorizationModel MatrixFactorizationModel.load(String path)
MatrixFactorizationModel MatrixFactorizationModel.load(SparkContext sc, String path)
Unit debug_print()
Unit release()

DESCRIPTION

MatrixFactorizationModel models the output of the frovedis matrix factorization using ALS (alternating least square) algorithm, the trainer interface of which aims to optimize an initial model and outputs the same after optimization.

Note that the actual model with user/product features etc. is created at frovedis server side only. Spark side MatrixFactorizationModel contains a unique ID associated with the frovedis server side model, along with some generic information like rank value etc. It simply works like a pointer to the in-memory model at frovedis server.

Any operations, like prediction etc. on a MatrixFactorizationModel makes a request to the frovedis server along with the unique model ID and the actual job is served by the frovedis server. For functions which returns some output, the result is sent back from frovedis server to the spark client.

Pubic Member Function Documentation

Double predict (Int uid, Int pid)

This method can be used on a trained model in order to predict the rating confidence value for the given product id, by the given user id.

"uid" should be in between 1 to M, where M is the number of users in the given data. And "pid" should be in between 0 to N, where N is the number of products in the given data.

`RDD[Rating]` predict (`RDD[(Int, Int)]` usersProducts)

This method can be used to predict the rating confidence values for a given list of pair of some user ids and product ids.

In the list of pairs, "uid" should be in between 1 to M and "pid" should be in between 1 to N, where M is the number of users and N is the number of products in the given data.

It is performed by all the worker nodes in parallel and on success the function returns a `RDD[Rating]` object containing the distributed predicted ratings at worker nodes.

`Array[Rating]` recommendProducts(Int uid, Int num)

This method can be used to recommend given "num" number of products for the user with given user id in sorted order (highest scored products to lowest scored products).

"uid" should be in between 1 to M, where M is the number of users in the given data. On success, it returns an array containing ratings for the recommended products by the given user.

`Array[Rating]` recommendUsers(Int pid, Int num)

This method can be used to recommend given "num" number of users for the product with given product id in sorted order (user with highest scores to user with lowest scores).

"pid" should be in between 1 to N, where N is the number of products in the given data. On success, it returns an array containing ratings for the recommended users the given product.

MatrixFactorizationModel MatrixFactorizationModel.load(String path)

This static function is used to load the target model with data in given filename stored at frovedis server side at specified location (filename with relative/aboslute path) as little-endian binary data. On success, it returns the loaded model.

MatrixFactorizationModel MatrixFactorizationModel.load(SparkContext sc, String path)

This is Spark like static API provided for compatibility with spark code. But the "sc" parameter is simply ignored in this case and internally it calls the above load() method as "MatrixFactorizationModel.load(path)".

Unit save(String path)

This function is used to save the target model with given filename. Note that the target model is saved at frovedis server side at specified location (filename with relative/aboslute path) as little-endian binary data.

Unit save(SparkContext sc, String path)

This is Spark like API provided for compatibility with spark code. But the “sc” parameter is simply ignored in this case and internally it calls the above save() method as “save(path)”.

Unit debug_print()

It prints the contents of the server side model on the server side user terminal. It is mainly useful for debugging purpose.

Unit release()

This function can be used to release the existing in-memory model at frovedis server side.