Package 'MCPanel'

October 30, 2017

Type Package

Title Matrix Completion algorithms for Causal Panel Data

Version 0.1.0

Author Khashayar Khosravi

Maintainer Khashayar Khosravi <khosravi@stanford.edu>

Description Given any matrix which some of its entries are missing, it is important to recover (estimate) missing entries. Inference questions usually can be only answered once this completion is finished. This package provides tools and functions to accomplish this task effectively.

Encoding UTF-8

LazyData true

Imports Rcpp (>= 0.12.13)

LinkingTo Rcpp

License GPL-2

R topics documented:

`mcnnm-package` *This package performs matrix completion via nuclear norm minimization.*

Description

Given any matrix which some of its entries are missing, it is important to recover (estimate) missing entries. Inference questions usually can be only answered once this completion is finished. In matrix completion literature, one idea for filling the missing entries is estimating the matrix with a low-rank matrix which is well-fitted to observed entries. One can formulate this objective as a convex optimization problem in which a weighted combination of "error" and "rank" are minimized.

Details

The idea is minimizing the weighted combination of average squared error of the estimated matrix (on observed entries) and nuclear norm of the estimated matrix. The optimum weight can be found in a data-driven fashion using cross-validation. For fitting models for different values of weight, we use warm-start method (similar to LASSO) which provides faster convergence. Three functions are available in this version:

mcnnm_cv(M, mask, to_estimate_u = 0, to_estimate_v = 0, num_lam = 100, niter = 1000, rel_tol = 1e-5, cv_ratio = 0.8, num_folds = 5, is_quiet = 1):

This function takes the observed matrix M and the binary mask which its (i,t) element is one if entry (i,t) is observed and is zero otherwise. Rest of arguments are optional:

to_estimate_u : if this boolean parameter is equal to zero, a vector of row-wise fixed effects would also be estimated in the model, i.e, u1^T. Default value is zero.

to_estimate_v : similar to to_estimate_u, but for columns, i.e. 1v^T. Default is again zero.

num_lam : number of weight values on the cross-validation path. These values are logarithmically spaced where the largest lambda causes the estimated matrix to be zero (in all folds). Default value is 100.

niter : number of iterations in coordinate descent algorithm. Default value is set to 1000, but usually convergence is achieved with a few iterations, due to warm-start.

rel_tol : this is the convergence criteria; if the relative improvement in objective function (new_obj_val-obj_val)/obj_val becomes smaller than rel_tol, the convergence is achieved and coordinate descent terminates. The default value is 1e-5.

cv_ratio : this is the ratio of training entries divided by total number of observed entries. The rest of samples are allocated to validation set, which helps in choosing model via cross-validation. Default allocation is 80/20 (training/validation)

num_folds : number of cross-validation folds. Note that this is not a num_folds-fold cross-validation, but rather, for each fold training and validation sets are chosen randomly (and independent of previous folds) with the probability given by cv-ratio. The default value is 5.

is_quiet : boolean parameter which indicates whether to print status of convergence and other outputs. Default value is TRUE(no output is printed).

mcnnm(M, mask, lambda_L, to_estimate_u = 0, to_estimate_v = 0, niter = 1000, rel_tol = 1e-5, is_quiet = 1);

This function takes M, mask, and a vector of lambda_L values (which is the weight parameter) as inputs, and finds minimizers of the objective value (for different values of lambda_L). Parameters are similar to previous function. It is very important to pass weights in decreasing order as warm-start then would help with faster convergence.

mcnnm_fit(M , mask, lambda_L, L_init = matrix(0,nrow(M),ncol(M)), u_init = matrix(0,nrow(M),1), v_init=matrix(0,ncol(M),1), to_estimate_u = 0, to_estimate_v = 0, niter = 1000, rel_tol = 1e-5, is_quiet = 1)

This function takes M, mask, and also a single lambda_L as input and finds a minimizer of the objective value using coordinate descent. Note that the initial starting values for L, u, and v (in case fixed effects exist) can be also passed to the function. Default values are chosen as matrices and vectors of all zero.

Author(s)

Khashayar Khosravi

Maintainer: Khashayar Khosravi <khosravi@stanford.edu>

References

This optional section can contain literature or other references for background information.

See Also

Optional links to other man pages

Examples

```
  ## Not run:
    mcnnm_cv(M,mask)

## End(Not run)
```

Index