Package 'mpersonalized'

March 18, 2018

Type Package

Title What the Package Does (Title Case)

Version 0.1.0

Author Chensheng Kuang

Maintainer Chensheng Kuang <ckuang@wisc.edu>

Description Implements a general framework to solve the problem of personalized medicine in meta-analysis and multiple outcomes. This framework allows either separate rules for each study/outcome or a single rule for all the studies/outcomes, depending on the requirement of user. A flexible choice of penalty functions to increase estimation efficiency is also provided.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Imports glmnet, SGL, caret, ggplot2, gridExtra, genlasso, Matrix

Suggests knitr, rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp

R topics documented:

admm_optim 2
lambda_estimate 2
mpersonalized 2
mpersonalized_cv 5
plot.mp 8
plot.mp_cv 8
predict.mp 9
predict.mp_cv 9
simulated_dataset 10

Index **12**

admm_optim *ADMM Algorithm for Meta-analysis/Multiple Outcomes Personalized Medicine*

Usage

```
admm_optim(x, y, p, q, lambda1, lambda2, lambda3, abs.tol = 1e-05,
  rel.tol = 1e-05, maxit = 500L, rho = NULL)
```

lambda_estimate *Default Penalty Parameter Sequence if Not Given*

Usage

```
lambda_estimate(modelXlist, modelYlist, penalty, single_rule, alpha,
  num_lambda1, num_lambda2, num_single_rule_lambda)
```

Arguments

modelXlist	the Xlist in contrast framework after standardization
modelYlist	the Ylist in contrast framework after standardization
penalty	penalty type
single_rule	whether a single treatment rule is required

Value

estimated lambda for required penalty if not provided

mpersonalized *A General Framework to Solve Personalized Medicine in the Settings of Meta-analysis/Multiple Outcomes*

Usage

```
mpersonalized(problem = c("meta-analysis", "multiple outcomes"), X, Trt,
  P = NULL, Xlist, Ylist, Trtlist, Plist = replicate(length(Xlist), NULL,
  simplify = FALSE), typelist = replicate(length(Xlist), "continuous",
  simplify = FALSE), penalty = c("none", "lasso", "GL", "SGL", "fused",
  "lasso+fused", "GL+fused", "SGL+fused"), lambda1 = NULL, lambda2 = NULL,
  single_rule_lambda = NULL, num_lambda1 = ifelse(!is.null(lambda1),
  length(lambda1), 10), num_lambda2 = ifelse(!is.null(lambda2),
  length(lambda2), 10),
  num_single_rule_lambda = ifelse(!is.null(single_rule_lambda),
  length(single_rule_lambda), 50), alpha = NULL, single_rule = FALSE,
  admm_control = NULL, contrast_builder_control = NULL)
```

Arguments

problem	A character string specifiy whether the user want to solve "meta-analysis" or "multiple outcomes" problem. For problem = "meta-analysis", the user should also supply Xlist, Ylist, Trtlist. For problem = "multiple outcomes", the user should supply X, Ylist, Trt.
X	Covariate matrix that should be supplied when problem = "multiple outcomes" with rows indicating subjects and columns indicating covariates.
Trt	Treatment vector that should be supplied when problem = "multiple outcomes", which should be coded as 0 or 1.
P	Propensity score vector when problem = "multiple outcomes". If not supplied, then study is treated as randomzied trial and the propensity score is estimated as the proportion of 1's in Trt for every subject.
Xlist	A list object that should be supplied when problem = "meta-analysis", with kth element denoting the covariate matrix of study k.
Ylist	When problem = "meta-analysis", Ylist should be a list object with kth element denoting the response vector of study k. When problem = "multiple outcomes", Ylist should be a list object with kth element denoting the kth outcome.
Trtlist	A list object that should be supplied when problem = "meta-analysis", with kth element denoting the treatment vector of study k (coded as 0 or 1).
Plist	A list object that should be supplied when problem = "meta-analysis", with $k$the element denoting the propensity score vector of study k. If not supplied, then each study is treated as randomized trial and the corresponding propensity score is estimated as the proportion of 1's in the kth element of Trtlist for all subjects.
penalty	For different rules, the penalty could be "none", "lasso", "GL", "SGL", "fused", "lasso+fused", "GL+fused", "SGL+fused". For single rule, the penalty could be "none" or "lasso". User should always input penalty and then supply corresponding penalty parameters sequence if needed. Default option is "none".
lambda1	λ_1 in the framework of different rules. If not supplied, a default sequence will be computed.
lambda2	λ_2 in the framework of different rules. If not supplied, a default sequence will be computed.
single_rule_lambda	λ_{single} in the framework of single rule.
num_lambda1	If lambda1 is not specified by user, user could still specify the length of the lambda1 sequence. The default length is 10.
num_lambda2	If lambda2 is not specified by user, user could still specify the length of the lambda2 sequence. The default length is 10.
num_single_rule_lambda	If single_rule_lambda is not specified, user could still specify the length of the single_rule_lambda sequence. The default length is 50.
alpha	α in the framework of different rules. If not supplied, a default value will be used depending on penalty.
single_rule	A logical value, whether the single treatment framework is used. Deafult is FALSE.

`admm_control` A list of parameters which user can specify to control the admm algorithm. In `admm_control`, the following parameters can be supplied: `abs.tol`, absolute tolerance; `rel.tol`, relative tolerance; `maxit`, maximum number of iterations; `rho`, Lagrangian parameter.

`contrast_builder_control` A list of parameters which user can specify to control estimation of contrast function. In `contrast_builder_control`, the following parameters could be supplied: `response_model`, this could be "lasso" or "linear"; `contrast_builder_folds`, the number of folds used in cross validation when response_model = "lasso".

`typelist` A list object with kth element denoting the type of outcome corresponding to the kth element in `Ylist`. Each element could be "continuous" or "binary".

Details

Assume the total number of studies/outcomes is K and we denote the contrast estimator for the kth study/outcome as $\hat{C}_k$ and the corresponding recommendation rule as g_k.

If we want different rules for each study/outcome, this function solves meta-analysis/multiple outcomes problems for personalized medicine based on the framework

$$\min_{g_1,\ldots,g_K} \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\frac{|\hat{C}_k(X_i)|}{\sum_{i=1}^{n_k}|\hat{C}_k(X_i)|}\left[1\{\hat{C}_k(X_i)>0\}-g_k(X_i)\right]^2+h(g_1,\ldots,g_K)$$

Here the regularization function h is of the form of a sum of sparse group lasso and fused lasso penalty

$$h=(1-\alpha)\lambda_1\sqrt{q}\sum_{j=1}^{p}\|\boldsymbol{\beta_j}\|_2+\alpha\lambda_1\sum_{j=1}^{p}\|\boldsymbol{\beta_j}\|_1+\lambda_2\sum_{j=1}^{p}\sum_{1\le a<b\le K}|\beta_{ja}-\beta_{jb}|$$

where $\boldsymbol{\beta_j}=(\beta_{j1},\ldots,\beta_{jK})$

By setting $\lambda_1,\lambda_2,\alpha$ differently, different penalties can be obtained.

- If $\lambda_1,\lambda_2\neq 0$ and $\alpha\neq 0$ or 1, the penalty is "SGL+fused".
- If $\lambda_1,\lambda_2\neq 0$ and $\alpha=0$, the penalty is "GL+fused".
- If $\lambda_1,\lambda_2\neq 0$ and $\alpha=1$, the penalty is "lasso+fused".
- If $\lambda_1=0,\lambda_2\neq 0$, the penalty is "fused".
- If $\lambda_1\neq 0,\lambda_2=0$ and $\alpha\neq 0$ or 1, the penalty is "SGL".
- If $\lambda_1\neq 0,\lambda_2=0$ and $\alpha=0$, the penalty is "GL".
- If $\lambda_1\neq 0,\lambda_2=0$ and $\alpha=1$, the penalty is "lasso".
- If $\lambda_1,\lambda_2=0$, there is no penalty.

On the other hand, if we would like to fit a single rule for all studies/outcomes, we let $g_1=\ldots=g_K$ and solve the following problem instead

$$\min_{g} \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\frac{|\hat{C}_k(X_i)|}{\sum_{i=1}^{n_k}|\hat{C}_k(X_i)|}\left[1\{\hat{C}_k(X_i)>0\}-g(X_i)\right]^2+h(g_1,\ldots,g_K)+\lambda_{single}\|\beta\|_1$$

Depending on the value of λ_{single}

- If $\lambda_{single}\neq 0$, the penalty is "lasso".
- If $\lambda_{single}=0$, there is no penalty.

Value

An S3 object of class "mp", which contains the information of the fitted model. It could be supplied to some other functions in mperosnalized package for further analysis or prediction.

`penalty_parameter_sequence`

configuration of the penalty parameters.

`interceptlist`

kth element corresponds to the kth row in `penalty_parameter_sequence`.

`betalist`

kth element corresponds to the kth row in `penalty_parameter_sequence`.

`number_covariates`

`number_studies_or_outcomes`

`problem = "meta-analysis"` or number of outcomes if `problem = "multiple outcomes"`.

Examples

```
set.seed(123)
sim_dat  = simulated_dataset(200, problem = "meta-analysis")
Xlist = sim_dat$Xlist; Ylist = sim_dat$Ylist; Trtlist = sim_dat$Trtlist

# fit different rules with SGL penalty for this meta-analysis problem
mp_mod_diff = mpersonalized(problem = "meta-analysis",
                      Xlist = Xlist, Ylist = Ylist, Trtlist = Trtlist,
                      penalty = "SGL", single_rule = FLASE)

# fir a single rule with lasso penalty
mp_mod_single = mpersonalized(problem = "meta-analysis",
                              Xlist = Xlist, Ylist = Ylist, Trtlist = Trtlist,
                              penalty = "lasso", single_rule = TRUE)
set.seed(NULL)
```

`mpersonalized_cv` *Cross Validation for* `mpersonalized`

Usage

```
mpersonalized_cv(problem = c("meta-analysis", "multiple outcomes"), X, Trt,
  P = NULL, Xlist, Ylist, Trtlist, Plist = replicate(length(Xlist), NULL,
  simplify = FALSE), typelist = replicate(length(Xlist), "continuous",
  simplify = FALSE), penalty = c("lasso", "GL", "SGL", "fused", "lasso+fused",
  "GL+fused", "SGL+fused"), lambda1 = NULL, lambda2 = NULL,
  single_rule_lambda = NULL, num_lambda1 = ifelse(!is.null(lambda1),
  length(lambda1), 10), num_lambda2 = ifelse(!is.null(lambda2),
```

```
  length(lambda2), 10),
  num_single_rule_lambda = ifelse(!is.null(single_rule_lambda),
  length(single_rule_lambda), 50), alpha = NULL, single_rule = FALSE,
  cv_folds = 5, admm_control = NULL, contrast_builder_control = NULL)
```

Arguments

problem	A character string specifiy whether the user want to solve "meta-analysis" or "multiple outcomes" problem. For `problem = "meta-analysis"`, the user should also supply `Xlist`, `Ylist`, `Trtlist`. For `problem = "multiple outcomes"`, the user should supply `X`, `Ylist`, `Trt`.
X	Covariate matrix that should be supplied when `problem = "multiple outcomes"` with rows indicating subjects and columns indicating covariates.
Trt	Treatment vector that should be supplied when `problem = "multiple outcomes"`, which should be coded as 0 or 1.
P	Propensity score vector when `problem = "multiple outcomes"`. If not supplied, then study is treated as randomzied trial and the propensity score is estimated as the proportion of 1's in `Trt` for every subject.
Xlist	A list object that should be supplied when `problem = "meta-analysis"`, with kth element denoting the covariate matrix of study k.
Ylist	When `problem = "meta-analysis"`, `Ylist` should be a list object with kth element denoting the response vector of study k. When `problem = "multiple outcomes"`, `Ylist` should be a list object with kth element denoting the kth outcome.
Trtlist	A list object that should be supplied when `problem = "meta-analysis"`, with kth element denoting the treatment vector of study k (coded as 0 or 1).
Plist	A list object that should be supplied when `problem = "meta-analysis"`, with $k$the element denoting the propensity score vector of study k. If not supplied, then each study is treated as randomized trial and the corresponding propensity score is estimated as the proportion of 1's in the kth element of `Trtlist` for all subjects.
penalty	For different rules, the penalty could be "lasso", "GL", "SGL", "fused", "lasso+fused", "GL+fused", "SGL+fused". For single rule, the penalty could only be "lasso". For `penalty = "none"`, use function `mpersonalized` instead. User should always input `penalty` and then supply correponding penalty parameters sequence if needed.
lambda1	λ_1 in the framework of different rules. If not supplied, a default sequence will be computed.
lambda2	λ_2 in the framework of different rules. If not supplied, a default sequence will be computed.
single_rule_lambda	λ_{single} in the framework of single rule.
num_lambda1	If `lambda1` is not specified by user, user could still specify the length of the `lambda1` sequence. The default length is 10.
num_lambda2	If `lambda2` is not specified by user, user could still specify the length of the `lambda2` sequence. The default length is 10.
num_single_rule_lambda	If `single_rule_lambda` is not specified, user could still specify the length of the `single_rule_lambda` sequence. The default length is 50.

`alpha` α in the framework of different rules. If not supplied, a default value will be used depending on `penalty`.

`single_rule` A logical value, whether the single treatment framework is used. Deafult is `FALSE`.

`cv_folds` Number of folds needed for cross-validation. Default is 5

`admm_control` A list of parameters which user can specify to control the admm algorithm. In `admm_control`, the following parameters can be supplied: `abs.tol`, absolute tolerance; `rel.tol`, relative tolerance; `maxit`, maximum number of iterations; `rho`, Lagrangian parameter.

`contrast_builder_control`
A list of parameters which user can specify to control estimation of contrast function. In `contrast_builder_control`, the following parameters could be supplied: `response_model`, this could be "lasso" or "linear"; `contrast_builder_folds`, the number of folds used in cross validation when response_model = "lasso".

`typelist` A list object with kth element denoting the type of outcome corresponding to the kth element in `Ylist`. Each element could be "continuous" or "binary".

Details

This function implments `mpersonalized` but use cross validatation for the tuning of penalty parameter. The optimal penalty parameter is selected by minimizing

$$\sum_{i=1}^{n_k} \frac{|\hat{C}_k(X_i)|}{\sum_{i=1}^{n_k}|\hat{C}_k(X_i)|}\left[1\{\hat{C}_k(X_i) > 0\} - g_k(X_i)\right]^2$$

in the leave-out fold, where $\hat{C}_k(X_i)$ in the leave-out fold is separately estimated from the training set.

Value

An S3 object of class "mp_cv", which contains the information of the model with the optimal lambda. It can be supplied to some other functions in mperosnalized package for further analysis or prediction.

`penalty_parameter_sequence`

configuration of the penalty parameters.

`opt_penalty_parameter`

`interccept`

\teimbeta The coefficient matrix corresponding to the optimal penalty parameter.

`number_covariates`

`number_studies_or_outcomes`

`problem = "meta-analysis"` or number of outcomes if `problem = "multiple outcomes"`.

plot.mp *Plot for a 'mp' Class Object.*

Usage

```
## S3 method for class 'mp'
plot(mp, ind1, ind2, single_ind)
```

Arguments

mp	the 'mp' class object returned by mpersonalzied function
ind1	the index of the lambda1 if different rules are used
ind2	the index of the lambda2 if different rules are used
single_ind	the index of the single_rule_lambda if an single rule is used

Details

This function plots the results for estimated treatment effects. Depending on the received treatment and recommended treatment, the group means of the outcome are computed and the relations between them are plotted. This plot provides a sanity check of the treatment recommendation rule. By specifiying the index of the penalty parameters, we can obtain the plots of the corresponding treatment recommendation rule.

Value

a list object with kth element denoting the plot of study k

plot.mp_cv *Plot for a 'mp' Class Object.*

Usage

```
## S3 method for class 'mp_cv'
plot(mp_cv)
```

Arguments

mp_cv	the 'mp_cv' class object returned by mpersonalzied_cv function

Details

This function plots the results for estimated treatment effects by using the estimated optimal treatment recommendation rule obtained from corss validation.

Value

a list object with kth element denoting the plot of study k

predict.mp *Predict for "mp" object*

Description

This function predict the benefit scores and optimal treatment for new patients

Usage

```
## S3 method for class 'mp'
predict(mp, newx = NULL, weight = NULL, overall_rec = TRUE)
```

Arguments

mp	the fitted "mp" object returned by "mpersonalized"
newx	the covariate matrix of the new patients. If newx = NULL, then the prediction is made for each data set based on the rule of that study/outcome.
weight	a weight vector for the overall recommendation. If leave as NULL, a equally weighted recommendation will be made.
overall_rec	a logical value. If TRUE, an overall recommendation will be made weighted by the "weight" parameter.

Value

a list of results with each element in the list corresponding to the prediciton by each different penalty parameter. Each element in this list contains:

treatment	recommended treatment for each patient for each study/outcome. If overall_rec = TRUE, the weighted overall recommended treatment will be computed as well. If the overall recommened treatment is equal to 0.5, it means the sum of weight is equal for 0 and 1.
benefit_score	the benefit score computed from $g_1, \ldots, g_K$

predict.mp_cv *Predict for "mp_cv" object*

Description

This function predict the benefit scores and optimal treatment for new patients

Usage

```
## S3 method for class 'mp_cv'
predict(mp_cv, newx = NULL, weight = NULL,
  overall_rec = TRUE)
```

Arguments

mp_cv	the fitted "mp_cv" object returned from "mpersonalized_cv" function
newx	the covariate matrix of the new patients. If newx = NULL, then the prediction is made for each data set based on the rule of that study/outcome.
weight	a weight vector for the overall recommendation. If leave as NULL, a equally weighted recommendation will be made.
overall_rec	a logical value. If TRUE, an overall recommendation will be made weighted by the "weight" parameter.

Value

the prediciton by using the optimal penalty parameter selected by cross validation. It contains:

treatment	recommended treatment for each patient for each study/outcome. If overall_rec = TRUE, the weighted overall recommended treatment will be computed as well. If the overall recommened treatment is equal to 0.5, it means the sum of weight is equal for 0 and 1.
benefit_score	the benefit score computed from $g_1, \ldots, g_K$

---

simulated_dataset *Simulated Dataset Generator*

Description

Generate a simulated dataset, which could be used to demonstrate the features of the mpersonalized package.

Usage

```
simulated_dataset(n, problem = c("meta-analysis", "multiple outcomes"))
```

Arguments

n	Sample size for each study/outcome.
problem	A character string specified what problem the simulated dataset is generated for. problem can be set to "meta-analysis" or "multiple outcomes".

Details

In the simulated dataset, outcomes are generated from the model

$$Y = \delta_0 + \boldsymbol{X}\boldsymbol{\delta} + A(\theta_0 + \boldsymbol{X}\boldsymbol{\theta}) + \epsilon,$$

where $\boldsymbol{X}$ is the baseline covariates and A is the treatment indicator coded as 0,1. For different outcomes or studies, values of δ_0, $\boldsymbol{\delta}$, θ_0 and $\boldsymbol{\theta}$ are also different so as to represent the heterogeneity in real problems.

The number of different studies/outcomes is set to be 6 and total number of candidate covariates is 50. Treatment indicator A is generated with equal probability of 0 or 1.

This function randomly generates the coefficients for each study/outcome and then generates the baseline covariates and error term for each subject. Depending on the value of problem, generation of baseline covariates are slightly different. For problem = "meta-analysis", baseline covariates are generated independently for each study; for problem = "multiple outcomes", baseline covariates are the same across different outcomes.

Value

A list object of the ingredients from the simulated dataset. The elements of this list depends on value of `problem`.

For `problem = "meta-analysis"`,

`Xlist`	a list object with kth element denoting the baseline covariate matrix of kth study
`Ylist`	a list object with kth element denoting the response vector of kth study
`Trtlist`	a list object with kth element denoting the treatment vector of kth study and coded as 0 or 1
`B`	the coefficient matrix containing δ_0, $\boldsymbol{\delta}$, θ_0 and $\boldsymbol{\theta}$

For `problem = "multiple outcomes"`,

`X`	a matrix object denoting the baseline covariate matrix
`Ylist`	a list object with kth element denoting the response vector of kth outcome
`Trt`	a vector denoting the treatment and coded as 0 or 1)
`B`	the coefficient matrix containing δ_0, $\boldsymbol{\delta}$, θ_0 and $\boldsymbol{\theta}$

Examples

```
set.seed(123)
sim_dat  = simulated_dataset(200, problem = "meta-analysis")
str(sim_dat$Xlist)
str(sim_dat$Ylist)
str(sim_dat$Trtlist)
set.seed(NULL)
```

Index

admm_optim, 2

lambda_estimate, 2

mpersonalized, 2
mpersonalized_cv, 5

plot.mp, 8
plot.mp_cv, 8
predict.mp, 9
predict.mp_cv, 9

simulated_dataset, 10