

# Package ‘mpersonalized’

March 17, 2018

**Type** Package

**Title** What the Package Does (Title Case)

**Version** 0.1.0

**Author** Chensheng Kuang

**Maintainer** Chensheng Kuang <ckuang@wisc.edu>

**Description** More about what it does (maybe more than one line)  
Use four spaces when indenting paragraphs within the Description.

**License** What license is it under?

**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 . . . . .	6
plot.mp_cv . . . . .	7
predict.mp . . . . .	8
predict.mp_cv . . . . .	8
simulated_dataset . . . . .	9
<b>Index</b>	<b>10</b>

---

admm_optim	<i>ADMM Algorithm for Meta-analysis/Multiple Outcomes Personalized Medicine</i>
------------	---

---

**Usage**

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

---

lambda_estimate	<i>Default Penalty Parameter Sequence if Not Given</i>
-----------------	--

---

**Usage**

```
lambda_estimate(modelXlist, modelYlist, penalty, unique_rule, alpha,
  num_lambda1, num_lambda2, num_unique_rule_lambda)
```

**Arguments**

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

**Value**

estimated lambda for required penalty if not provided

---

mpersonalized	<i>Meta-analysis/Multiple Outcomes for Personalized Medicine</i>
---------------	--

---

**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,
  unique_rule_lambda = NULL, num_lambda1 = ifelse(!is.null(lambda1),
  length(lambda1), 10), num_lambda2 = ifelse(!is.null(lambda2),
  length(lambda2), 10),
  num_unique_rule_lambda = ifelse(!is.null(unique_rule_lambda),
  length(unique_rule_lambda), 50), alpha = NULL, unique_rule = FALSE,
  admm_control = NULL, contrast_builder_control = NULL)
```

**Arguments**

problem	a character specify whether the user want to solve "meta-analysis" or "multiple outcomes" problem. For "meta-analysis" problem, the user should supply Xlist, Ylist, Trtlist and Plist. For "multiple outcomes" problem, the user should supply X, Ylist, Trt and P.
X	the covariate matrix that should be supplied when the problem is "multiple outcomes" with rows indicating subjects and columns indicating covariates.
Trt	the treatment vector that should be supplied when the problem is "multiple outcomes". It should be coded as 0 or 1.
P	the propensity score vector when the problem is "multiple outcomes". If not supplied, then study is treated as randomized trial and the propensity score is estimated as the proportion of 1's in Trt.
Xlist	a list object with $k$ th element denoting the covariate matrix of study $k$ . This should be supplied when the problem is "meta-analysis".
Ylist	When the problem is "meta-analysis", Ylist should be a list object with $k$ th element denoting the response vector of study $k$ . When the problem is "multiple outcomes", Ylist should be a list object with $k$ th element denoting the $k$ th outcome.
Trtlist	a list object with $k$ th element denoting the treatment vector of study $k$ (coded as 0 or 1). This should be supplied when the problem is "meta-analysis".
Plist	a list object with $k$ th 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 Trt.
penalty	For different rules, the penalty could be "none", "lasso", "GL", "SGL", "fused", "lasso+fused", "GL+fused", "SGL+fused". For unique rule, the penalty could be "none" or "lasso".
lambda1	lambda1 supplied in the framework when different rules are used.
lambda2	lambda2 supplied in the framework when different rules are used.
unique_rule_lambda	$\lambda$ when unique rule is used.
num_lambda1	length of the lambda1 sequence and default to be 10 if lambda1 is not provided
num_lambda2	length of the lambda2 sequence and default to be 10 if lambda2 is not provided
num_unique_rule_lambda	length of the unique_rule_lambda sequence and default to be 50 if unique_rule_lambda is not provided
alpha	alpha in the framework when different rules are used.
unique_rule	a logical value, whether a unique treatment rule is required
admm_control	a list of parameters which 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 control the contrast building process. 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".
typlelist	a list object with $k$ th element denoting the type of response corresponding to the $k$ th element in the list Ylist. Each element should be "continuous" or "binary".

## Details

Assume the total number of studies is  $K$ . This function is aimed to solve meta-analysis/multiple outcomes problems for personalized medicine based on the following framework:

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

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

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

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

If we would like a unique rule to be obtained, we let  $g_1 = \dots = g_K$  and solve the following question instead

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

If we want different rules, 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.

If we want unique rule,

- If  $\lambda \neq 0$ , the penalty is "lasso".
- If  $\lambda = 0$ , there is no penalty.

## Value

an S3 object of class "mp", which contains the information of the fitted model. It could be supplied to the predict function

---

mpersonalized_cv	<i>Meta-analysis/Multiple Outcomes for Personalized Medicine with Cross Validation</i>
------------------	--

---

## 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,
  unique_rule_lambda = NULL, num_lambda1 = ifelse(!is.null(lambda1),
  length(lambda1), 10), num_lambda2 = ifelse(!is.null(lambda2),
  length(lambda2), 10),
  num_unique_rule_lambda = ifelse(!is.null(unique_rule_lambda),
  length(unique_rule_lambda), 50), alpha = NULL, unique_rule = FALSE,
  cv_folds = 5, admm_control = NULL, contrast_builder_control = NULL)
```

## Arguments

problem	a character specify whether you want to solve "meta-analysis" or "multiple outcomes" problem. For "meta-analysis" problem, the user should supply Xlist, Ylist, Trtlist and Plist. For "multiple outcomes" problem, the user should supply X, Ylist, Trt and P.
X	the covariate matrix that should be supplied when the problem is "multiple outcomes" with rows indicating subjects and columns indicating covariates.
Trt	the treatment vector that should be supplied when the problem is "multiple outcomes". It should be coded as 0 or 1.
P	the propensity score vector when the problem is "multiple outcomes". If not supplied, then study is treated as randomized trial and the propensity score is estimated as the proportion of 1's in Trt.
Xlist	a list object with $k$ th element denoting the covariate matrix of study $k$ . This should be supplied when the problem is "meta-analysis".
Ylist	When the problem is "meta-analysis", Ylist should be a list object with $k$ th element denoting the response vector of study $k$ . When the problem is "multiple outcomes", Ylist should be a list object with $k$ th element denoting the $k$ th outcome.
Trtlist	a list object with $k$ th element denoting the treatment vector of study $k$ (coded as 0 or 1). This should be supplied when the problem is "meta-analysis".
Plist	a list object with $k$ th 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 Trt.
penalty	For different rules, the penalty could be "none", "lasso", "GL", "SGL", "fused", "lasso+fused", "GL+fused", "SGL+fused". For unique rule, the penalty could be "none" or "lasso".
lambda1	lambda1 supplied in the framework when different rules are used.
lambda2	lambda2 supplied in the framework when different rules are used.

unique_rule_lambda	$\lambda$ when unique rule is used.
num_lambda1	length of the lambda1 sequence and default to be 10 if lambda1 is not provided
num_lambda2	length of the lambda2 sequence and default to be 10 if lambda2 is not provided
num_unique_rule_lambda	length of the unique_rule_lambda sequence and default to be 50 if unique_rule_lambda is not provided
alpha	alpha in the framework when different rules are used.
unique_rule	a logical value, whether a unique treatment rule is required
cv_folds	number of folds needed for cross-validation, default is 5
admm_control	a list of parameters which 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 control the contrast building process. 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".
typlelist	a list object with $k$ th element denoting the type of response corresponding to the $k$ th element in the list Ylist. Each element should be "continuous" or "binary".

## Details

This function implements mpersonalized but use cross validation for the tuning of penalty parameter. The optimal penalty parameter is selected by minimizing

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

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 best fitted lambda. It can be supplied to the predict function.

---

plot.mp

*Plot for a 'mp' Class Object.*

---

## Usage

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

**Arguments**

mp	the 'mp' class object returned by mpersonalized function
ind1	the index of the lambda1 if different rules are used
ind2	the index of the lambda2 if different rules are used
unique_ind	the index of the unique_rule_lambda if an unique 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 specifying the index of the penalty parameters, we can obtain the plots of the corresponding treatment recommendation rule.

**Value**

a list object with  $k$ th element denoting the plot of study  $k$

---

plot.mp_cv	<i>Plot for a 'mp' Class Object.</i>
------------	--------------------------------------

---

**Usage**

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

**Arguments**

mp_cv	the 'mp_cv' class object returned by mpersonalized_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  $k$ th element denoting the plot of study  $k$

---

predict.mp	<i>Predict for "mp" object</i>
------------	--------------------------------

---

### 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 recommended 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 $g_1, \dots, g_K$

---

predict.mp_cv	<i>Predict for "mp_cv" object</i>
---------------	-----------------------------------

---

### 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 recommended 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 $g_1, \dots, g_K$

---

simulated_dataset	<i>Simulated Dataset Generator</i>
-------------------	------------------------------------

---

**Description**

This function could generate a simulated dataset for the test and usage of this package.

**Usage**

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

**Arguments**

n	sample size
sim_seed	the seed supplied to the generation
problem	a character specify whether to generate simulated data set for "meta-analysis" or "multiple outcomes" problem.

**Value**

A simulated dataset.

Xlist	a list object with $k$ th element denoting the covariate matrix of study k
Ylist	a list object with $k$ th element denoting the response vector of study k
Trtlist	a list object with $k$ th element denoting the treatment vector of study k (coded as 0 or 1)

# Index

`admm_optim`, [2](#)

`lambda_estimate`, [2](#)

`mpersonalized`, [2](#)

`mpersonalized_cv`, [5](#)

`plot.mp`, [6](#)

`plot.mp_cv`, [7](#)

`predict.mp`, [8](#)

`predict.mp_cv`, [8](#)

`simulated_dataset`, [9](#)