Package 'HMC'

March 5, 2024

Title High dimensional mean comparison with projection and cross-fitting

Version 1.0

Date 2024-03-04

Author Tianyu Zhang

Maintainer Tianyu Zhang <tianyuz3@andrew.cmu.edu>

Description

This package provide interpretable High-dimensional Mean Comparison methods (HMC). For example, users can use them to assess the difference of gene expression between two treatment groups. It is not a gene-by-gene comparison. Instead, we focus on the interplay between features and are interested in those are predictive of the group label. The methods are valid frequentist tests and give sparse estimate indicating which features contribute to the test results.

License GPL-2

Imports glmnet, irlba, PMA, MASS

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

NeedsCompilation no

R topics documented:

---

anchored_lasso_testing

Simple plug-in test for two-sample mean comparison.

Description

Simple plug-in test for two-sample mean comparison.

Usage

```
anchored_lasso_testing(
  sample_1,
  sample_2,
  pca_method = "sparse_pca",
  mean_method = "lasso",
  num_latent_factor = 1,
  n_folds = 5
)
```

Arguments

sample_1	Group 1 sample. Each row is a subject and each column corresponds to a feature.
sample_2	Group 2 sample. Each row is a subject and each column corresponds to a feature.
pca_method	Methods used to estimate principle component The default is "sparse_pca", using sparse PCA from package PMA. Other choices are "dense_pca"—the regular PCA; and "hard"— hard-thresholding PCA, which also induces sparsity.
mean_method	Methods used to estimate the mean vector. Default is sample mean "naive". There is also a hard-thresholding sparse estiamtor "hard".
num_latent_factor	
	The principle component that lasso coefficient anchors at. The default is PC1 = 1.
n_folds	Number of splits when performing cross-fitting. The default is 5, if computational time allows, you can try to set it to 10.

Examples

```
sample_size_1 <- sample_size_2 <- 300
true_mean_1 <- matrix(c(rep(1, 10), rep(0, 90)), ncol = 1)
true_mean_2 <- matrix(c(rep(1.5, 10), rep(0, 90)), ncol = 1)

sample_1 <- data.frame(MASS::mvrnorm(sample_size_1,
                                mu = true_mean_1,
                                Sigma = diag(1, 100)))
 sample_2 <- data.frame(MASS::mvrnorm(sample_size_2,
                                mu = true_mean_2,
                                Sigma = diag(1, 100)))
 result <- anchored_lasso_testing(sample_1, sample_2)
 result$test_statistics
##the test statistic. It should follow normal(0,1) when there is no difference between the groups.
 summarize_feature_name(result)
 #summarize which features contribute to discriminant vectors (i.e. logistic lasso)
```

```
extract_pc(result) # extract the estimated discriminant coefficients
```

---

debiased_pc_testing	*Debiased one-step test for two-sample mean comparison. A small p-value tells us not only there is difference in the mean vectors, but can also indicates which principle component the difference aligns with.*

Description

Debiased one-step test for two-sample mean comparison. A small p-value tells us not only there is difference in the mean vectors, but can also indicates which principle component the difference aligns with.

Usage

```
debiased_pc_testing(
  sample_1,
  sample_2 = NULL,
  pca_method = "sparse_pca",
  mean_method = "naive",
  num_latent_factor = 1,
  n_folds = 5
)
```

Arguments

sample_1	Group 1 sample. Each row is a subject and each column corresponds to a feature.
sample_2	Group 2 sample. Each row is a subject and each column corresponds to a feature.
pca_method	Methods used to estimate principle component The default is "sparse_pca", using sparse PCA from package PMA. Other choices are "dense_pca"—the regular PCA; and "hard"— hard-thresholding PCA, which also induces sparsity.
mean_method	Methods used to estimate the mean vector. Default is sample mean "naive". There is also a hard-thresholding sparse estiamtor "hard".
num_latent_factor	
	Number of principle to be estimated/tested. Default is 1.
n_folds	Number of splits when performing cross-fitting. The default is 5, if computational time allows, you can try to set it to 10.

Examples

```
sample_size_1 <- sample_size_2 <- 300

true_mean_1 <- matrix(c(rep(1, 10), rep(0, 90)), ncol = 1)
true_mean_2 <- matrix(c(rep(1.5, 10), rep(0, 90)), ncol = 1)
pc1 <- c(rep(1, 10), rep(0, 90))
pc1 <- pc1/norm(pc1, type = '2')

simulation_covariance <- 10 * pc1 %*% t(pc1)
simulation_covariance <- simulation_covariance + diag(1, 100)
```

```
sample_1 <- data.frame(MASS::mvrnorm(sample_size_1,
                                mu = true_mean_1,
                                Sigma = simulation_covariance))
 sample_2 <- data.frame(MASS::mvrnorm(sample_size_2,
                                mu = true_mean_2,
                                Sigma = simulation_covariance))
 result <- debiased_pc_testing(sample_1, sample_2)
 result$test_statistics
 ##these are test statistics. Each one of them corresponds to one PC.
 summarize_pc_name(result, latent_fator_index = 1) #shows which features contribute to PC1
 extract_pc(result) # extract the estimated leading PCs.
```

---

estimate_nuisance_parameter_lasso

The function for nuisance parameter estimation in anchored_lasso_testing().

Description

The function for nuisance parameter estimation in anchored_lasso_testing().

Usage

```
estimate_nuisance_parameter_lasso(
  nuisance_sample_1,
  nuisance_sample_2,
  pca_method = "sparse_pca",
  mean_method = "lasso",
  num_latent_factor = 1
)
```

Arguments

nuisance_sample_1
                    Group 1 sample. Each row is a subject and each column corresponds to a feature.

nuisance_sample_2
                    Group 2 sample. Each row is a subject and each column corresponds to a feature.

pca_method   Methods used to estimate principle component The default is "sparse_pca", us-
                    ing sparse PCA from package PMA. Other choices are "dense_pca"—the regular
                    PCA; and "hard"— hard-thresholding PCA, which also induces sparsity.

mean_method   Methods used to estimate the discriminant direction. Default is logistic Lasso
                    "lasso".

num_latent_factor
                    The principle component that lasso coefficient anchors at. The default is PC1 =
                    1.

---

estimate_nuisance_pc   *The function for nuisance parameter estimation in simple_pc_testing()*
                              and debiased_pc_testing().

Description

The function for nuisance parameter estimation in simple_pc_testing() and debiased_pc_testing().

Usage

```
estimate_nuisance_pc(
  nuisance_sample_1,
  nuisance_sample_2 = NULL,
  pca_method = "sparse_pca",
  mean_method = "naive",
  num_latent_factor = 1
)
```

Arguments

nuisance_sample_1

        Group 1 sample. Each row is a subject and each column corresponds to a feature.

nuisance_sample_2

        Group 2 sample. Each row is a subject and each column corresponds to a feature.

pca_method   Methods used to estimate principle component The default is "sparse_pca", us-
        ing sparse PCA from package PMA. Other choices are "dense_pca"—the regular
        PCA; and "hard"— hard-thresholding PCA, which also induces sparsity.

mean_method   Methods used to estimate the mean vector. Default is sample mean "naive".
        There is also a hard-thresholding sparse estiamtor "hard".

num_latent_factor

        Number of principle to be estimated/tested. Default is 1.

---

evaluate_influence_function_multi_factor

        Calculate the test statistics on the left-out samples. Called in debi-
        ased_pc_testing().

Description

Calculate the test statistics on the left-out samples. Called in debiased_pc_testing().

Usage

```
evaluate_influence_function_multi_factor(
  cross_fitting_sample_1,
  cross_fitting_sample_2 = NULL,
  nuisance_collection,
  num_latent_factor = 1
)
```

Arguments

cross_fitting_sample_1
:   Group 1 sample. Each row is a subject and each column corresponds to a feature.

cross_fitting_sample_2
:   Group 2 sample. Each row is a subject and each column corresponds to a feature.

nuisance_collection
:   A collection of nuisance quantities estimated using "nuisance" samples. It is the output of estimate_nuisance_pc().

num_latent_factor
:   Number of principle components to be considered.

---

evaluate_pca_lasso_plug_in

Calculate the test statistics on the left-out samples. Called in anchored_lasso_testing().

Description

Calculate the test statistics on the left-out samples. Called in anchored_lasso_testing().

Usage

```
evaluate_pca_lasso_plug_in(
  cross_fitting_sample_1,
  cross_fitting_sample_2,
  nuisance_collection,
  mean_method = "lasso"
)
```

Arguments

cross_fitting_sample_1
:   Group 1 sample. Each row is a subject and each column corresponds to a feature.

cross_fitting_sample_2
:   Group 2 sample. Each row is a subject and each column corresponds to a feature.

nuisance_collection
:   A collection of nuisance quantities estimated using "nuisance" samples. It is the output of estimate_nuisance_pc().

mean_method
:   Methods used to estimate the discriminant direction. Default is logistic Lasso "lasso".

---

evaluate_pca_plug_in   *Calculate the test statistics on the left-out samples. Called in simple_pc_testing().*

Description

Calculate the test statistics on the left-out samples. Called in simple_pc_testing().

Usage

```
evaluate_pca_plug_in(
  cross_fitting_sample_1,
  cross_fitting_sample_2 = NULL,
  nuisance_collection
)
```

Arguments

cross_fitting_sample_1
> Group 1 sample. Each row is a subject and each column corresponds to a feature.

cross_fitting_sample_2
> Group 2 sample. Each row is a subject and each column corresponds to a feature.

nuisance_collection
> A collection of nuisance quantities estimated using "nuisance" samples. It is the output of estimate_nuisance_pc().

---

extract_lasso_coef   *Extract the lasso estimate from the output of anchored_lasso_testing().*

Description

Extract the lasso estimate from the output of anchored_lasso_testing().

Usage

```
extract_lasso_coef(testing_result)
```

Arguments

testing_result   The output/test result list from anchored_lasso_testing().

Value

A list, whose elements are the estimated discriminant directions for each split—the length of the output list is the same as n_folds.

---

extract_pc   *Extract the principle components from the output of sim-*
                                    ple_pc_testing() and debiased_pc_testing().

Description

Extract the principle components from the output of simple_pc_testing() and debiased_pc_testing().

Usage

```
extract_pc(testing_result)
```

Arguments

testing_result   The output/test result list from simple_pc_testing() or debiased_pc_testing().

Value

A list, whose elements are the estimated PC for each split—the length of the output list is the same as n_folds.

---

index_spliter   *Split the sample index into n_folds many groups so that we can per-*
                                    form cross-fitting

Description

Split the sample index into n_folds many groups so that we can perform cross-fitting

Usage

```
index_spliter(array, n_folds = 5)
```

Arguments

array   Sample index. Usually just an array from 1 to the number of samples in one group.

n_folds   Number of splits

---

simple_pc_testing   *Simple plug-in test for two-sample mean comparison.*

Description

Simple plug-in test for two-sample mean comparison.

Usage

```
simple_pc_testing(
  sample_1,
  sample_2 = NULL,
  pca_method = "sparse_pca",
  mean_method = "naive",
  num_latent_factor = 1,
  n_folds = 5
)
```

Arguments

sample_1	Group 1 sample. Each row is a subject and each column corresponds to a feature.
sample_2	Group 2 sample. Each row is a subject and each column corresponds to a feature.
pca_method	Methods used to estimate principle component The default is "sparse_pca", using sparse PCA from package PMA. Other choices are "dense_pca"—the regular PCA; and "hard"— hard-thresholding PCA, which also induces sparsity.
mean_method	Methods used to estimate the mean vector. Default is sample mean "naive". There is also a hard-thresholding sparse estiamtor "hard".
num_latent_factor	
	Number of principle to be estimated/tested. Default is 1.
n_folds	Number of splits when performing cross-fitting. The default is 5, if computational time allows, you can try to set it to 10.

Examples

```
sample_size_1 <- sample_size_2 <- 300

true_mean_1 <- matrix(c(rep(1, 10), rep(0, 90)), ncol = 1)
true_mean_2 <- matrix(c(rep(1.5, 10), rep(0, 90)), ncol = 1)
pc1 <- c(rep(1, 10), rep(0, 90))
pc1 <- pc1/norm(pc1, type = '2')

simulation_covariance <- 10 * pc1 %*% t(pc1)
simulation_covariance <- simulation_covariance + diag(1, 100)

sample_1 <- data.frame(MASS::mvrnorm(sample_size_1,
                              mu = true_mean_1,
                              Sigma = simulation_covariance))
 sample_2 <- data.frame(MASS::mvrnorm(sample_size_2,
                              mu = true_mean_2,
                              Sigma = simulation_covariance))
 result <- simple_pc_testing(sample_1, sample_2)
```

```
result$test_statistics
##these are test statistics. Each one of them corresponds to one PC.
summarize_pc_name(result, latent_fator_index = 1) #shows which features contribute to PC1
extract_pc(result) # extract the estimated leading PCs.
```

---

summarize_feature_name

> *Summarize the features (e.g. genes) that contribute to the test result,*
> *i.e. those features consistently show up in Lasso vectors.*

Description

Summarize the features (e.g. genes) that contribute to the test result, i.e. those features consistently show up in Lasso vectors.

Usage

```
summarize_feature_name(testing_result, method = "majority voting")
```

Arguments

testing_result   The output/test result list from anchored_lasso_testing().

method   How to combine the feature list across different splits. Default is 'majority voting'—features that show up more than 50% of the splits are considered active/useful. It can be 'union'—all the features pooled together; or 'intersection'—only include features showing up in all splits.

Value

A list of names of features (your very original input data need to have column names!) that contribute to the test result. An empty list means there is barely any difference between the two groups.

---

summarize_pc_name   *Summarize the features (e.g. genes) that contribute to the test result,*
                        i.e. those features consistently show up in the sparse principle compo-
                        nents.

Description

Summarize the features (e.g. genes) that contribute to the test result, i.e. those features consistently show up in the sparse principle components.

Usage

```
summarize_pc_name(
  testing_result,
  latent_fator_index = 1,
  method = "majority voting"
)
```

Arguments

testing_result   The output/test result list from simple_pc_testing() or debiased_pc_testing().

latent_fator_index

Which principle component should the algorithm summarize? Default is PC1.

method   How to combine the feature list across different splits. Default is 'majority voting'—features that show up more than 50% of the splits are considered active/useful. It can be 'union'—all the features pooled together; or 'intersection'—only include features showing up in all splits.

Value

A list of names of features (your very original input data need to have column names!) that contribute to the test result.

Index