Package ‘mmit’

February 21, 2019

Maintainer Toby Dylan Hocking <toby.hocking@r-project.org>

Author Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Version 2017.03.15

License GPL-3

Title Max Margin Interval Trees

Description Fast O(P N log N) algorithm for learning a regression tree
with interval censored output data.

Suggests future.apply, testthat, penaltyLearning

Imports partykit, assertthat

RoxygenNote 6.0.1

R topics documented:

compute_optimal_costs 2
mmif 3
mmif.cv 4
mmif.predict 5
mmit 6
mmit.cv 7
mmit.predict 8
mmit.pruning 9
mse 10
zero_one_loss 10

Index **12**

compute_optimal_costs *compute optimal costs*

Description

Compute vector of optimal prediction and cost.

Usage

```
compute_optimal_costs(target.mat, margin, loss = "hinge")
```

Arguments

target.mat	n x 2 matrix of limits.
margin	numeric scalar, margin size parameter.
loss	character scalar, hinge or square.

Value

data.frame with columns moves (number of times the pointer was moved for each data point, sum of upper and lower limit moves), pred (predicted output value that achieves minimum cost), cost (minimum cost value).

Author(s)

Toby Dylan Hocking, Alexandre Drouin

Examples

```
library(mmit)
target.mat <- rbind(
  c(-1, Inf),
  c(-2, 3),
  c(-Inf, 1))
compute_optimal_costs(target.mat, 0)
compute_optimal_costs(target.mat, 2)
```

mmif *Random Forest of Max Margin Interval Tree*

Description

Learning a random forest of Max Margin Interval Tree.

Usage

```
mmif(target.mat, feature.mat, max_depth = Inf, margin = 0, loss = "hinge",
  min_sample = 1, n_trees = 10,
  n_features = ceiling(ncol(feature.mat)^0.5))
```

Arguments

`target.mat`	The response variable of the model
`feature.mat`	a data frame containing the feature variables in the model.
`max_depth`	The maximum depth of each tree
`margin`	margin hyperparameter
`loss`	The type of loss; (`"hinge"`, `"square"`)
`min_sample`	The minimum number of samples required to partition a leaf in a tree
`n_trees`	The number of trees in the ensemble (forest)
`n_features`	The number of features to be used to train each tree

Value

List of trees containing each tree in the random forest.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)

target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))

colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)
```

```
trees <- mmif(target.mat, feature.mat, margin = 2.0)
```

mmif.cv *Cross-validation for model selection with Random Forests of Max Margin Interval Trees*

Description

Performing grid search to select the best hyperparameters of mmif via cross-validation.

Usage

```
mmif.cv(target.mat, feature.mat, param_grid, n_folds = 3, scorer = NULL)
```

Arguments

`target.mat` The response variable of the model

`feature.mat` A data frame containing the feature variables in the model.

`param_grid` A list with values to try for each hyperparameter (max_depth, margin, min_sample, loss, n_trees, n_features).

`n_folds` The number of folds for k-fold cross-validation

`scorer` The function used to calculate the cross-validation score (e.g., mse, zero_one_loss)

Value

The best score, best model (trained with best parameters), best parameters, and list of all parameter values with cross validation score.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)

target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))

colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)
```

```
param_grid <- NULL
param_grid$max_depth <- c(Inf, 4, 3)
param_grid$margin <- c(2, 3, 5)
param_grid$min_sample <- c(2, 5, 10)
param_grid$loss <- c("hinge")
param_grid$n_trees <- c(10, 20, 30)
param_grid$n_features <- c(ceiling(ncol(feature.mat)**0.5))

result <- mmif.cv(target.mat, feature.mat, param_grid, scorer = mse)
```

`mmif.predict` *Predictions with random forests of Max Margin Interval Trees*

Description

Predictions with random forests of Max Margin Interval Trees

Usage

```
mmif.predict(forest, test_feature.mat = NULL)
```

Arguments

`forest` Ensemble of MMITs

`test_feature.mat` A data frame containing the features of the examples for which predictions must be computed.

Value

Predictions Average output of each tree in the forest

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)

target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))
```

```
colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)

forest <- mmif(target.mat, feature.mat)
pred <- mmif.predict(forest, feature.mat)
```

mmit *The Max Margin Interval Tree*

Description

Learning a regression tree for censored data.

Usage

```
mmit(target.mat, feature.mat, max_depth = Inf, margin = 0, loss = "hinge",
  min_sample = 1)
```

Arguments

`target.mat`	The response variable of the model
`feature.mat`	a data frame containing the feature variables in the model.
`max_depth`	The maximum depth criteia
`margin`	margin paramaters
`loss`	The type of loss; (`"hinge"`, `"square"`)
`min_sample`	The minimum number of sample required

Value

The learned regression tree as an object of class party.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))
```

```
colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)


out <- mmit(target.mat, feature.mat)
```

mmit.cv *The Cross Validation of Max Margin Interval Tree*

Description

Performing grid search to select the best parameters via cross validation on the a regression tree for censored data.

Usage

```
mmit.cv(target.mat, feature.mat, param_grid, n_folds = 3, scorer = NULL,
  pruning = TRUE)
```

Arguments

`target.mat`	The response variable of the model
`feature.mat`	a data frame containing the feature variables in the model.
`param_grid`	the list of paramaters
`n_folds`	The number of folds
`scorer`	The Loss calculation function
`pruning`	Boolean whether pruning is to be done or not.

Value

The list consist of best score, best tree, best parameters and list of all parameter values with cross validation score .

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))

colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)

param_grid <- NULL
param_grid$max_depth <- c(Inf, 4, 3)
param_grid$margin <- c(2, 3, 5)
param_grid$min_sample <- c(2, 5, 10)
param_grid$loss <- c("hinge")

result <- mmit.cv(target.mat, feature.mat, param_grid, scorer = mse)
```

mmit.predict *The Predict Function for Max Margin Interval Tree*

Description

Fits the new data into the MMIT model to give prediction values

Usage

```
mmit.predict(tree, newdata = NULL, perm = NULL)
```

Arguments

tree	The Max Margin Interval Tree obtained from "mmit()"
newdata	an optional data frame containing the testing data which is to be predicted.
perm	an optional character vector of variable names.

Value

The learned regression tree as an object of class party.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))
```

```
colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)

tree <- mmit(target.mat, feature.mat)

pred <- mmit.predict(tree)
```

`mmit.pruning` *The Pruned Max Margin Interval Tree*

Description

Pruning the regression tree for censored data.

Usage

```
mmit.pruning(tree)
```

Arguments

`tree` The fitted tree using "`mmit()`" function

Value

The learned regression tree as an object of class party.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
target.mat <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

feature.mat <- rbind(
  c(1,0,0), c(1,1,0), c(1,2,0),
  c(1,3,0), c(1,4,0), c(1,5,0))

colnames(feature.mat) <- c("a", "b", "c")
feature.mat <- data.frame(feature.mat)


tree <- mmit(target.mat, feature.mat)

pruned_tree <- mmit.pruning(tree)
```

mse *The Mean Square Error*

Description

Metric for mean aquare error calculation.

Usage

```
mse(y_true, y_pred)
```

Arguments

y_true	The actual response variable of the model
y_pred	The predicted response value of the model

Value

A numeric value which signifies the error quantity.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
y_true <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

y_pred <- c(0.5, 2, 0, 1.5, 3.5, 2.5)

out <- mse(y_true, y_pred)
```

zero_one_loss *The Zero One Loss*

Description

Metric for error calculation where the function gives zero value inside the interval else one.

Usage

```
zero_one_loss(y_true, y_pred)
```

Arguments

y_true	The actual response variable of the model
y_pred	The predicted response value of the model

Value

A numeric value which signifies the error quantity.

Author(s)

Toby Dylan Hocking, Alexandre Drouin, Torsten Hothorn, Parismita Das

Examples

```
library(mmit)
y_true <- rbind(
  c(0,1), c(0,1), c(0,1),
  c(2,3), c(2,3), c(2,3))

y_pred <- c(0.5, 2, 0, 1.5, 3.5, 2.5)

out <- zero_one_loss(y_true, y_pred)
```

Index

compute_optimal_costs, 2

mmif, 3
mmif.cv, 4
mmif.predict, 5
mmit, 6
mmit.cv, 7
mmit.predict, 8
mmit.pruning, 9
mse, 10

zero_one_loss, 10