# Package 'BESTree'

January 20, 2025

**Type** Package

**Title** Branch-Exclusive Splits Trees

**Version** 0.5.2

**Description** Decision tree algorithm with a major feature added. Allows for users to define an ordering on the partitioning process.
Resulting in Branch-Exclusive Splits Trees (BEST). Cedric Beaulac and Jeffrey S. Rosentahl (2019) <arXiv:1804.10168>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Imports** plyr, compiler, utils, stats

**RoxygenNote** 6.1.1

**Suggests** knitr, rmarkdown, testthat

**Depends** R (>= 2.10)

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Beaulac Cedric [aut, cre]

**Maintainer** Beaulac Cedric <cedric@utstat.toronto.edu>

**Repository** CRAN

**Date/Publication** 2019-08-09 11:00:02 UTC

## Contents

## Acc *Computes the proportion of matching terms in two vectors of the same length. Used to compute the accuracy for prediction on test set.*

### Description

Computes the proportion of matching terms in two vectors of the same length. Used to compute the accuracy for prediction on test set.

### Usage

```
Acc(Vec1, Vec2)
```

### Arguments

| | |
|---|---|
| Vec1 | A vector of labels |
| Vec2 | Another vector of labels |

### Value

Percentage of identical labels (accuracy)

### Examples

```
Vec1 <- c(1,1,2,3,1)
Vec2 <- c(1,2,2,3,1)
Acc(Vec1,Vec2)
```

## BaggedBEST *Performs Bootstrap Aggregating of BEST trees*

### Description

Performs Bootstrap Aggregating of BEST trees

### Usage

```
BaggedBEST(Data, VA, NoT = 50, Size = 50)
```

### Arguments

`Data` A data set (Data Frame): Can take on both numerical and categorical predictors. Last column of the data set must be the Repsonse Variable (Categorical Variables only)

`VA` Variable Availability structure

`NoT` Number of Trees in the bag

`Size` Minimal Number of Observation within a leaf needed for partitionning (default is 50)

### Value

A list of BEST Objects

### Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
NoT <- 10
Fit <- BESTree::BaggedBEST(Data,VA,NoT,Size)
```

---

## `BEST` *Main function of the package. It produces Classification Trees with Branch-Exclusive variables.*

---

### Description

Main function of the package. It produces Classification Trees with Branch-Exclusive variables.

### Usage

```
BEST(Data, Size, VA)
```

### Arguments

`Data` A data set (Data Frame): Can take on both numerical and categorical predictors. Last column of the data set must be the Repsonse Variable (Categorical Variables only)

`Size` Minimal Number of Observation within a leaf needed for partitionning

`VA` Variable Availability structure

### Value

A BEST object with is a list containing the resulting tree, row numbers for each regions and the split points

### Examples

```
n <- 1000
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
Fit <- BESTree::BEST(Data,Size,VA)
```

## `BESTForest` *Generates a random forest of BEST trees*

### Description

Generates a random forest of BEST trees

### Usage

```
BESTForest(Data, VA, NoT = 50, Size = 50)
```

### Arguments

`Data` A data set (Data Frame): Can take on both numerical and categorical predictors. Last column of the data set must be the Repsonse Variable (Categorical Variables only)

`VA` Variable Availability structure

`NoT` Number of Trees in the bag

`Size` Minimal Number of Observation within a leaf needed for partitionning (default is 50)

### Value

A list of BEST Objects (Random Forest)

### Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
NoT <- 10
Fit <- BESTree::BESTForest(Data,VA,NoT,Size)
```

## `Data` *Data generated according to decision tree for simulation purposes*

### Description

Data generated according to decision tree for simulation purposes

### Usage

```
Data
```

### Format

A data frame with 10000 rows and 5 variables:

**X_1** Binary predictor

**X_2** Binary predictor

**X_3** Continuous predictor between 0 and 1

**X_4** Continuous predictor between 0 and 1

**Y** The response variable ...

## `Fit` *Data generated according to decision tree for simulation purposes*

### Description

Data generated according to decision tree for simulation purposes

### Usage

```
Fit
```

### Format

A typical list produced by the BEST function:

**1** Tree structure indicating spliting variables, impurity of the region and split variable

**2** List of splitting values

**3** Observaton numbers in the respective regions ...

## ForgeVA *Quickly build the Available Variable list necessary for BEST This list contains details as to which variables is available for the partitioning. It also contains which variables are gating variables.*

### Description

Quickly build the Available Variable list necessary for BEST This list contains details as to which variables is available for the partitioning. It also contains which variables are gating variables.

### Usage

```
ForgeVA(d, GV, BEV, Thresh = 0.5, Direc = 0)
```

### Arguments

| | |
|---|---|
| `d` | Number of predictors |
| `GV` | Gating variables |
| `BEV` | Branch-Exclusive Variables |
| `Thresh` | Threshold for Gates |
| `Direc` | Direction of Gates ( 1 means add variable if bigger than thresh) |

### Value

The list containing the Variable Availability structure

### Examples

```
#This function can be used to set up the variable availability structure.
#Suppose we want to fit a regular decision tree on a data set containing d predictors
d <- 10
VA <- ForgeVA(d,1,0,0,0)
#Suppose now that predictor x5 is a binary gating variable for x4
#such that x4 is available if x5 = 1
GV <- 5 #The gating variable
BEV <- 4 #The Branch-Exclusive variable
Tresh = 0.5 #Value between 0 and 1
Direc = 1 #X4 is available if X5 is bigger than Tresh
VA <- ForgeVA(d,GV,BEV,Tresh,Direc)
```

## FPredict *Emits prediction from a forest of BEST's*

### Description

Emits prediction from a forest of BEST's

### Usage

```
FPredict(M, LFit)
```

### Arguments

| | |
|---|---|
| M | A matrix of new observations where one row is one observation |
| LFit | A list of BEST Objects (Usually produced by RBEST or BESTForest) |

### Value

A vector of predictions

### Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
NewPoints <- BESTree::Data[(n+1):(n+11),1:d]
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
NoT <- 10
Fit <- BESTree::BaggedBEST(Data,VA,NoT,Size)
Predictions <- BESTree::FPredict(NewPoints,Fit)
```

## MPredict *Classify a set of new observation points*

### Description

Classify a set of new observation points

### Usage

```
MPredict(M, Fit)
```

### Arguments

| | |
|---|---|
| M | A matrix of new observations where one row is one observation |
| Fit | A BEST object |

## Value

The predicted class

## Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
NewPoints <- BESTree::Data[(n+1):(n+11),1:d]
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
Fit <- BESTree::BEST(Data,Size,VA)
Predictions <- BESTree::MPredict(NewPoints,Fit)
```

---

# `Predict` *Classify a new observation point*

---

## Description

Classify a new observation point

## Usage

```
Predict(Point, Fit)
```

## Arguments

| | |
|---|---|
| `Point` | A new observation |
| `Fit` | A BEST object |

## Value

The predicted class

## Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
NewPoint <- BESTree::Data[n+1,]
d <- ncol(Data)-1
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
Fit <- BESTree::BEST(Data,Size,VA)
BESTree::Predict(NewPoint[1:d],Fit)
```

---

`TreePruning` *Uses a Validation Set to select the best trees within the list of pruned trees.*

---

### Description

Uses a Validation Set to select the best trees within the list of pruned trees.

### Usage

```
TreePruning(Fit, VSet)
```

### Arguments

| | |
|---|---|
| `Fit` | A BEST object |
| `VSet` | A Validation Set (Can also be used in CV loop) |

### Value

The shallower trees among trees wiht Highest accuracy. This replaces the first element in the BEST object list.

### Examples

```
nv <- 50
ValData <- BESTree::Data[(1000+1):nv,]
Fit <- BESTree::Fit
Fit[[1]] <- BESTree::TreePruning(Fit,ValData)
```

---

`VI` *Produces a variable important analysis using the mean decrease in node impurity*

---

### Description

Produces a variable important analysis using the mean decrease in node impurity

### Usage

```
VI(Forest)
```

### Arguments

| | |
|---|---|
| `Forest` | A list of BEST Objects (Usually produced by RBEST or BESTForest) |

## Value

A vector of importance (size d)

## Examples

```
n <- 500
Data <- BESTree::Data[1:n,]
d <- ncol(Data)-1
NewPoints <- BESTree::Data[(n+1):(n+11),1:d]
VA <- ForgeVA(d,1,0,0,0)
Size <- 50
NoT <- 10
Fit <- BESTree::BaggedBEST(Data,VA,NoT,Size)
VI <- BESTree::VI(Fit)
```

# Index