Package 'figs'

November 21, 2022

Type Package

Title Fast Interpretable Greedy-Tree Sums

Version 0.8

Author Haoxue Wang, Yan Shuo Tan, Chandan Singh, Keyan Nasseri, Abhineet Agarwal, Bin Yu

Maintainer Haoxue Wang <hw613@cam.ac.uk>

Description

Fast Interpretable Greedy-Tree Sums (FIGS) is an algorithm for fitting concise rule-based models. Specifically, FIGS generalizes CART to simultaneously grow a flexible number of trees in a summation.
The total number of splits across all the trees can be restricted by a pre-specified threshold, keeping the model interpretable.
Experiments across real-world datasets show that FIGS achieves state-of-the-art prediction performance when restricted to just a few splits (e.g. less than 20).
'FIGS' is first defined in Tan et al. (2022) <https://arxiv.org/abs/2201.11931>

Imports rpart, fastshap

License MIT + file LICENSE

URL https://github.com/wanghaoxue0/figs

BugReports https://github.com/wanghaoxue0/figs/issues

Encoding UTF-8

Depends R (>= 3.5.0)

Date 2022-11-01

RoxygenNote 7.2.2

R topics documented:

`classifier_split` *split the node for classification*

Description

make the split based on figs classification rule

Usage

```
classifier_split(
  X,
  y,
  idxs,
  tree_num = 0,
  sample_weight = NULL,
  max_features = NULL
)
```

Arguments

`X`	the design matrix
`y`	the response vector
`idxs`	the index of samples used for the split(array of bool values. If TRUE, it is used for the split)
`tree_num`	initial tree_num (default=0)
`sample_weight`	allocate weight for each individual sampe, array-like of shape n (default=None). If None, then samples are equally weighted.
`max_features`	the number of features to consider when looking for the best split

Value

call: the input setting

node_split: the design matrix for beta after variable splitting node_split$idxs : the index of samples in the original node node_split$value: the value for the split node node_split$impurity: the impurity in the original node node_split$tree_num: the tree number

node_left: the design matrix for gamma after variable splitting node_split$idxs : the index of samples in the left node node_split$value: the value for the left node after the split node_split$impurity: the impurity in the left node node_split$tree_num: the tree number which the node belongs to

node_right: the response vector after variable splitting. node_split$idxs : the index of samples in the right node node_split$value: the value for the right node after the split node_split$impurity: the impurity in the right node node_split$tree_num: the tree number which the node belongs to

n: the number of samples in the original node

feature: the feature selected to do the split

threshold: the threshold to make the split

impurity_reduction: the impurity decreased by the split if the decrease exists

figs.classifier *figs fit for classification*

Description

fitting function for fast interpretable greedy-tree sums on classification

Usage

```
figs.classifier(
  X,
  y,
  max_rules = 12,
  feature_names = NULL,
  sample_weight = NULL,
  min_impurity_decrease = 0,
  verbose = TRUE,
  max_features = NULL
)
```

Arguments

X	the design matrix
y	the response vector
max_rules	max total number of rules across all trees
feature_names	the name of features in the data
sample_weight	allocate weight for each individual sampe, array-like of shape n (default=None). If None, then samples are equally weighted.
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value.
verbose	bool value, if TRUE print the split feature and threshold
max_features	the number of features to consider when looking for the best split

Value

call: the input setting

node_split: the design matrix for beta after variable splitting node_split$idxs : the index of samples in the original node node_split$value: the value for the split node node_split$impurity: the impurity in the original node node_split$tree_num: the tree number

node_left: the design matrix for gamma after variable splitting node_split$idxs : the index of samples in the left node node_split$value: the value for the left node after the split node_split$impurity: the impurity in the left node node_split$tree_num: the tree number which the node belongs to

node_right: the response vector after variable splitting. node_split$idxs : the index of samples in the right node node_split$value: the value for the right node after the split node_split$impurity: the impurity in the right node node_split$tree_num: the tree number which the node belongs to

trees: tree struction based on the result

n: the number of samples in the original node

feature: the feature selected to do the split

threshold: the threshold to make the split

impurity_reduction: the impurity decreased by the split if the decrease exists

---

`figs.regressor` *figs fit for regression*

Description

fitting function for fast interpretable greedy-tree sums on regression

Usage

```
figs.regressor(
  X,
  y,
  max_rules = 12,
  feature_names = NULL,
  verbose = TRUE,
  sample_weight = NULL,
  min_impurity_decrease = 0,
  max_features = NULL
)
```

Arguments

`X`	the design matrix
`y`	the response vector
`max_rules`	max total number of rules across all trees
`feature_names`	the name of features in the data
`verbose`	bool value, if TRUE print the split feature and threshold
`sample_weight`	allocate weight for each individual sampe, array-like of shape n (default=None). If None, then samples are equally weighted.
`min_impurity_decrease`	A node will be split if this split induces a decrease of the impurity greater than or equal to this value.
`max_features`	the number of features to consider when looking for the best split

Value

call: the input setting

node_split: the design matrix for beta after variable splitting node_split$idxs : the index of samples in the original node node_split$value: the value for the split node node_split$impurity: the impurity in the original node node_split$tree_num: the tree number

node_left: the design matrix for gamma after variable splitting node_split$idxs : the index of samples in the left node node_split$value: the value for the left node after the split node_split$impurity: the impurity in the left node node_split$tree_num: the tree number which the node belongs to

node_right: the response vector after variable splitting. node_split$idxs : the index of samples in the right node node_split$value: the value for the right node after the split node_split$impurity: the impurity in the right node node_split$tree_num: the tree number which the node belongs to

trees: tree struction based on the result

n: the number of samples in the original node

feature: the feature selected to do the split

threshold: the threshold to make the split

impurity_reduction: the impurity decreased by the split if the decrease exists

---

`figscv` *cross validation for FIGS algoritms*

Description

choose the parameter by cross validation

Usage

```
figscv(X = X, Y = Y)
```

`plot` *plot the FIGS tree result*

Description

plot the tree based on the tree struction

Usage

```
plot(X, y)
```

`regressor_split` *split the node for regression*

Description

make the split based on figs regression rule

Usage

```
regressor_split(
  X,
  y,
  idxs,
  tree_num = 0,
  sample_weight = NULL,
  max_features = NULL
)
```

Arguments

X	the design matrix
y	the response vector
idxs	the index of samples used for the split(array of bool values. If TRUE, it is used for the split)
tree_num	initial tree_num (default=0)
sample_weight	allocate weight for each individual sampe, array-like of shape n (default=None). If None, then samples are equally weighted.
max_features	the number of features to consider when looking for the best split

Value

call: the input setting

node_split: the design matrix for beta after variable splitting node_split$idxs : the index of samples in the original node node_split$value: the value for the split node node_split$impurity: the impurity in the original node node_split$tree_num: the tree number

node_left: the design matrix for gamma after variable splitting node_split$idxs : the index of samples in the left node node_split$value: the value for the left node after the split node_split$impurity: the impurity in the left node node_split$tree_num: the tree number which the node belongs to

node_right: the response vector after variable splitting. node_split$idxs : the index of samples in the right node node_split$value: the value for the right node after the split node_split$impurity: the impurity in the right node node_split$tree_num: the tree number which the node belongs to

n: the number of samples in the original node

feature: the feature selected to do the split

threshold: the threshold to make the split

impurity_reduction: the impurity decreased by the split if the decrease exists

---

simulation *simulation for FIGS algoritms*

Description

simulation for R function in FIGS packages

Usage

```
simulation(X = NULL)
```

Index