Package 'gbts'

October 13, 2022

Type Package

Title Hyperparameter Search for Gradient Boosted Trees

Version 1.2.0

Date 2017-02-26

Author Waley W. J. Liang

Maintainer Waley W. J. Liang <wliang10@gmail.com>

Description An implementation of hyperparameter optimization for Gradient Boosted Trees on binary classification and regression problems. The current version provides two optimization methods: Bayesian optimization and random search. Instead of giving the single best model, the final output is an ensemble of Gradient Boosted Trees constructed via the method of ensemble selection.

License GPL (>= 2) | file LICENSE

LazyData true

Imports doParallel, doRNG, foreach, gbm, earth

Depends R (>= 3.3.0)

Suggests testthat

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-02-27 08:41:32

R topics documented:

boston_housing *Boston housing data*

Description

This dataset concerns the values of 506 houses in suburbs of Boston.

Usage

```
boston_housing
```

Format

A list of 4 components:

train A `data.frame` of the training dataset which contains 354 rows and 14 columns.

test A `data.frame` of the test dataset which contains 152 rows and 14 columns.

target_idx A column index of the target (response) varible.

pred_idx A set of column indicies of the predictors.

Source

https://archive.ics.uci.edu/ml/datasets/Housing

comperf *Compute model performance*

Description

This function computes model performance given a vector of response values and a vector of predictions.

Usage

```
comperf(y, yhat, w = rep(1, length(y)), pfmc = NULL, cdfx = "fpr",
  cdfy = "tpr", dspt = 0.5)
```

Arguments

y	a vector of numeric response values.
yhat	a vector of model predictions.
w	an optional vector of observation weights.
pfmc	a character of the performance metric. For binary classification, pfmc accepts: • "acc": accuracy • "dev": deviance • "ks": Kolmogorov-Smirnov (KS) statistic • "auc": area under the ROC curve. Use the cdfx and cdfy arguments to specify the cumulative distributions for the x-axis and y-axis of the ROC curve, respectively. The default ROC curve is given by true positive rate (y-axis) vs. false positive rate (x-axis). • "roc": rate on the y-axis of the ROC curve at a particular decision point (threshold) on the x-axis specified by the dspt argument. For example, if the desired performance metric is true positive rate at the 5% false positive rate, specify pfmc="roc", cdfx="fpr", cdfy="tpr", and dspt=0.05. For regression, pfmc accepts: • "mse": mean squared error • "mae": mean absolute error • "rsq": r-squared (coefficient of determination)
cdfx	a character of the cumulative distribution for the x-axis. Supported values are • "fpr": false positive rate • "fnr": false negative rate • "rpp": rate of positive prediction
cdfy	a character of the cumulative distribution for the y-axis. Supported values are • "tpr": true positive rate • "tnr": true negative rate
dspt	a decision point (threshold) in [0, 1] for binary classification. If pfmc="acc", instances with probabilities <= dspt are predicted as negative, and those with probabilities > dspt are predicted as positive. If pfmc="roc", dspt is a threhold on the x-axis of the ROC curve such that the corresponding value on the y-axis is used as the performance metric. For example, if the desired performance metric is the true positive rate at the 5% false positive rate, specify pfmc="roc", cdfx="fpr", cdfy="tpr", and dspt=0.05.

Value

A single or a vector of numeric values of model performance, or a list of two components x and y representing the ROC curve.

Author(s)

Waley W. J. Liang <<wliang10@gmail.com>>

See Also

gbts, predict.gbts

Examples

```
y = c(0, 1, 0, 1, 1, 1)
yhat = c(0.5, 0.9, 0.2, 0.7, 0.6,  0.4)
comperf(y, yhat, pfmc = "auc")
# 0.875

y = 1:10
yhat = c(1:5 - 0.1, 6:10 + 0.1)
comperf(y, yhat, pfmc = "mse")
# 0.01
```

---

gbts *Hyperparameter Search for Gradient Boosted Trees*

Description

This package implements hyperparameter optimization for Gradient Boosted Trees (GBT) on binary classification and regression problems. The current version provides two optimization methods:

- Bayesian optimization:
  1. A predictive model is built to capture the relationship between GBT hyperparameters and the resulting predictive performance.
  2. Select the best hyperparameter setting (determined by a pre-specified criterion) to try in the next iteration.
  3. Train GBT on the selected hyperparameter setting and compute validation performance.
  4. Update the predictive model with the new validation performance. Go back to step 2 and repeat.
- Random search: each GBT is built with a randomly selected hyperparameter setting.

Instead of returning a single GBT in the final output, an ensemble of GBTs is produced via the method of ensemble selection. It selects GBTs with replacement from a library into the ensemble, and returns the ensemble with best validation performance. Model library and validation performance are obtained from the hyperparameter search described above, by building GBTs with different hyperparameter settings on the training dataset and obtaining their performances on the validation dataset, based on cross-validation (CV). Since selection from the library is done with replacement, each GBT may be selected more than once into the ensemble. This function returns an ensemble that contains only the unique GBTs with model weights calculated as the number of model duplicates divided by the ensemble size. Each unique GBT in the ensemble is re-trained on the full training data. Prediction is computed as the weighted average of predictions from the re-trained GBTs.

Usage

```
gbts(x, y, w = rep(1, nrow(x)), nitr = 200, nlhs = floor(nitr/2),
  nprd = 5000, kfld = 10, srch = c("bayes", "random"), nbst = 100,
  ensz = 100, nwrk = 2, rpkg = c("gbm"), pfmc = c("acc", "dev", "ks",
  "auc", "roc", "mse", "rsq", "mae"), cdfx = "fpr", cdfy = "tpr",
  dspt = 0.5, lower = c(2, 10, 0.1, 0.1, 0.01, 50, 1), upper = c(10, 200,
  1, 1, 0.1, 1000, 10), quiet = FALSE)
```

Arguments

x	a data.frame of predictors. Categorical predictors represented as factors are accepted.
y	a vector of response values. For binary classification, y must contain values of 0 and 1. It is unnecessary to convert y to a factor variable. For regression, y must contain at least two unique values.
w	an optional vector of observation weights.
nitr	an integer of the number of hyperparameter settings that are sampled. For Bayesian optimization, nitr must be larger than nlhs.
nlhs	an integer of the number of Latin Hypercube samples (each sample is a hyperparameter setting) used to initialize the predictive model of GBT performance. This is used for Bayesian optimization only. After initialization, sequential search continues for nitr-nlhs iterations.
nprd	an integer of the number of hyperparameter settings at which GBT performance is estimated using the predictive model and the best is selected to train GBT in the next iteration.
kfld	an integer of the number of folds for cross-validation.
srch	a character of the search method such that srch="bayes" performs Bayesian optimization (default), and srch="random" performs random search.
nbst	an integer of the number of bootstrap samples to construct the predictive model of GBT performance.
ensz	an integer value of the ensemble size - number of GBTs selected into the ensemble. Since ensemble selection is done with replacement, the number of unique GBTs may be less than ensz, but the sum of model weights always equals ensz.
nwrk	an integer of the number of computing workers to use on a single machine.
rpkg	a character indicating which R package implementation of GBT to use. Currently, only the gbm R package is supported.
pfmc	a character of the performance metric used as the optimization objective. For binary classification, pfmc accepts: • "acc": accuracy • "dev": deviance • "ks": Kolmogorov-Smirnov (KS) statistic • "auc": area under the ROC curve. Use the cdfx and cdfy arguments to specify the cumulative distributions for the x-axis and y-axis of the ROC curve, respectively. The default ROC curve is given by true positive rate (y-axis) vs. false positive rate (x-axis).

- "roc": rate on the y-axis of the ROC curve at a particular decision point (threshold) on the x-axis specified by the dspt argument. For example, if the desired performance metric is true positive rate at the 5% false positive rate, specify pfmc="roc", cdfx="fpr", cdfy="tpr", and dspt=0.05.

For regression, pfmc accepts:

- "mse": mean squared error
- "mae": mean absolute error
- "rsq": r-squared (coefficient of determination)

cdfx — a character of the cumulative distribution for the x-axis. Supported values are

- "fpr": false positive rate
- "fnr": false negative rate
- "rpp": rate of positive prediction

cdfy — a character of the cumulative distribution for the y-axis. Supported values are

- "tpr": true positive rate
- "tnr": true negative rate

dspt — a decision point (threshold) in [0, 1] for binary classification. If pfmc="acc", instances with probabilities <= dspt are predicted as negative, and those with probabilities > dspt are predicted as positive. If pfmc="roc", dspt is a threhold on the x-axis of the ROC curve such that the corresponding value on the y-axis is used as the performance metric. For example, if the desired performance metric is the true positive rate at the 5% false positive rate, specify pfmc="roc", cdfx="fpr", cdfy="tpr", and dspt=0.05.

lower — a numeric vector containing the minimum values of hyperparameters in the following order:

- maximum tree depth
- leaf node size
- bag fraction
- fraction of predictors to try for each split
- shrinkage
- number of trees
- scale of weights for positive cases (for binary classification only)

upper — a numeric vector containing the maximum values of hyperparameters in the order above.

quiet — a logical of TRUE turns off the display of optimization progress in the console.

Value

A list of information with the following components:

- model: an ensemble (list) of GBT model(s).
- model_weight: a vector of model weights whose sum equals ensz.
- best_idx: an integer of the iteration index for the best validation performance.
- pred_val: a matrix of cross-validation predictions where nrow(pred_val) = nrow(x) and ncol(pred_val) = nitr.

- perf_val: a vector of cross-validation performance measures.
- param: a data.frame of hyperparameter settings visited. Each row of the data.frame is a single hyperparameter setting.
- objective: a character of the objective function used.
- time a list of times:
  - pproc_time a numeric value of preprocessing time in minutes.
  - binit_time a numeric value of initialization time in minutes for Bayesian optimization.
  - bsrch_time a numeric value of search time in minutes for Bayesian optimization.
  - rsrch_time a numeric value of random search time in minutes.
  - enslt_time a numeric value of ensemble selection in minutes.
  - refit_time a numeric value of refitting (on the full training data) time in minutes.
  - total_time a numeric value of the total time in minutes.
- ...: input arguments (excluding x, y, and w).

Author(s)

Waley W. J. Liang <<wliang10@gmail.com>>

References

Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. 2004. Ensemble selection from libraries of models. In Proceedings of the 21st international conference on Machine learning (ICML'04). http://www.cs.cornell.edu/~alexn/papers/shotgun.icml04.revised.rev2.pdf

See Also

predict.gbts, comperf

Examples

```
## Not run:
# Binary classification

# Load German credit data
data(german_credit)
train <- german_credit$train
test <- german_credit$test
target_idx <- german_credit$target_idx
pred_idx <- german_credit$pred_idx

# Train a GBT model with optimization on AUC
model <- gbts(train[, pred_idx], train[, target_idx], nitr = 200, pfmc = "auc")

# Predict on test data
yhat_test <- predict(model, test[, pred_idx])

# Compute AUC on test data
```

```
comperf(test[, target_idx], yhat_test, pfmc = "auc")


# Regression

# Load Boston housing data
data(boston_housing)
train <- boston_housing$train
test <- boston_housing$test
target_idx <- boston_housing$target_idx
pred_idx <- boston_housing$pred_idx

# Train a GBT model with optimization on MSE
model <- gbts(train[, pred_idx], train[, target_idx], nitr = 200, pfmc = "mse")

# Predict on test data
yhat_test <- predict(model, test[, pred_idx])

# Compute MSE on test data
comperf(test[, target_idx], yhat_test, pfmc = "mse")

## End(Not run)
```

`german_credit` *German credit data*

Description

This dataset classifies 1,000 people described by a set of attributes as good or bad credit risks.

Usage

```
german_credit
```

Format

A list of 4 components:

train A `data.frame` of the training dataset which contains 700 rows and 21 columns.

test A `data.frame` of the test dataset which contains 300 rows and 21 columns.

target_idx A column index of the target (response) varible.

pred_idx A set of column indicies of the predictors.

Source

https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)

predict.gbts *Predict method for ensemble of Gradient Boosted Trees*

Description

This function generates predictions by weighted averaging the predictions from each model in the ensemble returned from gbts. Weighted average is computed on the log-odds scale for binary classification.

Usage

```
## S3 method for class 'gbts'
predict(object, x, nwrk = 2, ...)
```

Arguments

object	a model object returned from gbts.
x	a data.frame of predictors. It must follow the same format as the training dataset on which the model object is developed.
nwrk	an integer of the number of computing workers to be used. If nwrk is less than the number of available cores on the machine, it uses all available cores.
...	further arguments passed to or from other methods.

Value

A numeric vector of predictions. In the case of binary classification, predictions are probabilities.

Author(s)

Waley W. J. Liang <<wliang10@gmail.com>>

See Also

gbts, comperf

Index