Package ‘dblr’

July 22, 2025

Type Package

Title Discrete Boosting Logistic Regression

Version 0.1.0

Author Nailong Zhang

Maintainer Nailong Zhang <setseed2016@gmail.com>

Description Trains logistic regression model by discretizing continuous variables via gradient boosting approach. The proposed method tries to achieve a tradeoff between interpretation and prediction accuracy for logistic regression by discretizing the continuous variables. The variable binning is accomplished in a supervised fashion. The model trained by this package is still a single logistic regression model, but not a sequence of logistic regression models. The fitted model object returned from the model training consists of two tables. One table is used to give the boundaries of bins for each continuous variable as well as the corresponding coefficients, and the other one is used for discrete variables. This package can also be used for binning continuous variables for other statistical analysis.

License GPL-3

Encoding UTF-8

LazyData true

Imports data.table (>= 1.9.6), xgboost (>= 0.6-4), CatEncoders (>= 0.1.1), Metrics (>= 0.1.1), methods

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-11 17:31:59 UTC

Contents

dblr_train *Discrete Boosting Logistic Regression Training*

Description

dblr_train fits a dblr (discrete boosting logistic regression) model.

Usage

```
dblr_train(train_x, train_y, category_cols = NULL, metric = "auc",
  subsample = 1, eta = 0.1, colsample = 1, cv_nfold = 5,
  cv_nrounds = 1000, cv_early_stops = 25, lambda = 1, alpha = 0,
  scale_pos_weight = 1, verbose = FALSE, seed = 123456L)
```

Arguments

`train_x`	A data.frame of training variables, which can include NA as well
`train_y`	A vector of 0 and 1 to represent labels of training samples
`category_cols`	A vector of column names to indicate which columns are categorical. Default: NULL means all columns are continuous
`metric`	Which metric to use, can be either auc or logloss. Default: auc
`subsample`	Subsample ratio from the trainnig samples in each iteration. Default: 1.0
`eta`	Controls the rate of learning. eta should be between 0 and 1. Default: 0.1
`colsample`	Subsample ratio from all available variables/columns. Default: 1.0
`cv_nfold`	Number of folds used for cross-validation. Default: 5
`cv_nrounds`	Number of iterations used for cross-validation. Default: 1000
`cv_early_stops`	Cross-validation would be stopped if there is no improvement after cv_early_stops iterations. Default: 25
`lambda`	Control L2 regularization term. Default: 1.0
`alpha`	Control L1 regularization term. Default: 0.0
`scale_pos_weight`	Useful when training metric is set to auc for imbalanced training data
`verbose`	Default: FALSE. If TRUE, the cross-validation process would be showed
`seed`	Random seed for the sampling. Default: 123456

Details

As one of the generalized linear models, traditional logistic regression on continuous variables implies that there is a monotonic relation between each predictor and the predicted probability. Bining or discretizing the continuous variables would be helpful when non-monotonic relation exists. In general, it is challenging to find the optimal binning for continuous variables. Too many bins may cause over-fitting and too few bins may not reveal the non-monotinc relation as much as possible. Thus, we propose to use a boosting decision trees to construct a discrete logistic regressions aiming at an automated binning process with good performance. Our algorithm is to construct a sequence of gradient boosting decision trees with at most 1 variable in each tree. Aggregating all decision trees with the same variable would result in the corresponding bins and the coefficients. And by aggregating all trees without variables we would get the intercept.

The model is defined as:

$$Pr(y=1|\boldsymbol{x}_i) = \frac{1}{1+\exp(-\sum_{j=1}^{m} g(\boldsymbol{x}_{i,j}) - b)},$$

where $g(\boldsymbol{x}_{i,j})$ denotes the coefficient of the bin which $\boldsymbol{x}_{i,j}$ falls into and b denotes the intercept. Both coefficients and intercept are consolidated from boosting trees. More specifically,

$$g(\boldsymbol{x}_{i,j}) = \sum_{k=1}^{K} f_k(\boldsymbol{x}_{i,j}) \cdot I(\text{tree } k \text{ splits on variable } j),$$

$$b = \sum_{k=1}^{K} f_k \cdot I(\text{tree } k \text{ does not split on any variable}),$$

where K is the total number of trees and f_k is the output value for tree k. In this package, we use xgboost package to training the underlying gradient boosting trees.

Value

Returns an object of S3 class dblr, which contains two attributes, i.e., continuous_bins and categorical_bins.

Examples

```
# use iris data for example
dat <- iris
# create two categorical variables
dat$Petal.Width <- as.factor((iris$Petal.Width<=0.2)*1+(iris$Petal.Width>1.0)*2)
dat$Sepal.Length <- (iris$Sepal.Length<=3.0)*2+(iris$Sepal.Length>6.0)*1.25
# create the response variable
dat$Species <- as.numeric(dat$Species=='versicolor')
set.seed(123)
# random sampling
index <- sample(1:150,100,replace = FALSE)
# train the dblr model using the training data
dblr_fit <- dblr_train(train_x=dat[index,c(1:4)],
train_y=dat[index,5],category_cols = c('Petal.Width','Sepal.Length'),
metric = 'logloss',subsample = 0.5,eta = 0.05,colsample = 1.0,
lambda = 1.0,cv_early_stops = 10,verbose=FALSE)
# make predictions on testing data
pred_dblr <- predict(dblr_fit,newdata = dat[-index,],type = 'response')
dblr_auc <- Metrics::auc(actual = dat[-index,'Species'],predicted = pred_dblr)
dblr_logloss <- Metrics::logLoss(actual = dat[-index,'Species'],predicted = pred_dblr)
cat('test auc for dblr model:',dblr_auc,'\n')
cat('test logloss for dblr model:',dblr_logloss,'\n')
glm_fit <- glm(data=dat[index,],formula =Species~. ,family = binomial)
```

```
pred_glm <- predict(glm_fit,newdata = dat[-index,],type='response')
glm_auc <- Metrics::auc(actual = dat[-index,'Species'],predicted = pred_glm)
glm_logloss <- Metrics::logLoss(actual = dat[-index,'Species'],predicted = pred_glm)
cat('test auc for glm model:',glm_auc,'\n')
cat('test logloss for glm model:',glm_logloss,'\n')
```

---

predict.dblr *Discrete Boosting Logistic Regression Prediction*

Description

predict.dblr makes predictions on new data set given the fitted dblr model object.

Usage

```
## S3 method for class 'dblr'
predict(object, newdata, type = "response", ...)
```

Arguments

object	A fitted dblr model object, which should be returned by calling dblr_train function
newdata	A data.frame contains the samples to predict
type	Control the output of prediction. Default: 'response' means probability; 'Link' would produce the linear part; 'mapped' would produce a data.frame filling with the coefficients of the model
...	further arguments passed to or from other methods

Value

Returns a vector of prediction or a data.frame

Index