Package ‘kernelFactory’

September 29, 2015

Type Package

Title Kernel Factory: An Ensemble of Kernel Machines

Version 0.3.0

Date 2015-09-29

Imports randomForest, AUC, genalg, kernlab, stats

Author Michel Ballings, Dirk Van den Poel

Maintainer Michel Ballings <Michel.Ballings@GMail.com>

Description Binary classification based on an ensemble of kernel machines ("Ballings, M. and Van den Poel, D. (2013), Kernel Factory: An Ensemble of Kernel Machines. Expert Systems With Applications, 40(8), 2904-2913"). Kernel factory is an ensemble method where each base classifier (random forest) is fit on the kernel matrix of a subset of the training data.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2015-09-29 17:33:15

R topics documented:

Credit .......................................................... 2
kernelFactory .................................................. 2
kFNews ........................................................ 4
predict.kernelFactory ........................................ 5

Index ......................................................... 7
Credit

Credit approval (Frank and Asuncion, 2010)

Description
Credit contains credit card applications. The dataset has a good mix of continuous and categorical features.

Usage
data(Credit)

Format
A data frame with 653 observations, 15 predictors and a binary criterion variable called Response

Details
All observations with missing values are deleted.

Source

References
The original dataset can be downloaded at http://archive.ics.uci.edu/ml/datasets/Credit+Approval

Examples
data(Credit)
str(Credit)
table(Credit$Response)

kernelFactory

Binary classification with Kernel Factory

Description
kernelFactory implements an ensemble method for kernel machines (Ballings and Van den Poel, 2013).
Usage

kernelFactory(x = NULL, y = NULL, cp = 1, rp = round(log(nrow(x), 10)),
    method = "burn", ntree = 500, filter = 0.01, popSize = rp * cp * 7,
    iters = 80, mutationChance = 1/(rp * cp), elitism = max(1, round((rp * cp) * 0.05)), oversample = TRUE)

Arguments

x       A data frame of predictors (numeric, integer or factor). Categorical variables
        need to be factors. Indicator values should not be too imbalanced because this
        might produce constants in the subsetting process.
y       A factor containing the response vector. Only \{0,1\} is allowed.
cp      The number of column partitions.
rp      The number of row partitions.
method  Can be one of the following: POLynomial kernel function (pol), LINear
        kernel function (lin), Radial Basis kernel Function rbf), random choice
        (random=pol, lin, rbf), burn in choice of best function (burn=pol, lin, rbf
        (burn). Use random or burn if you don’t know in advance which kernel function is best.
ntree   Number of trees in the Random Forest base classifiers.
filter  either NULL (deactivate) or a percentage denoting the minimum class size of
        dummy predictors. This parameter is used to remove near constants. For example if
        nrow(xTRAIN)=100, and filter=0.01 then all dummy predictors with any
        class size equal to 1 will be removed. Set this higher (e.g., 0.05 or 0.10) in case
        of errors.
popSize Population size of the genetic algorithm.
iters   Number of generations of the genetic algorithm.
mutationChance Mutationchance of the genetic algorithm.
elitism Elitism parameter of the genetic algorithm.
oversample Oversample the smallest class. This helps avoid problems related to the subsetting
        procedure (e.g., if rp is too high).

Value

An object of class kernelFactory, which is a list with the following elements:

trn       Training data set.
trnlst    List of training partitions.
rbfstre   List of used kernel functions.
rbfmtrX   List of augmented kernel matrices.
rsltsKF   List of models.
cpr       Number of column partitions.
rpr       Number of row partitions.
cntr      Number of partitions.
kfNews

Weights of the ensemble members.

rmDtrn Vector indicating the numeric (and integer) features.

rngs Ranges of numeric predictors.

constants To exclude from newdata.

Author(s)

Authors: Michel Ballings and Dirk Van den Poel, Maintainer: <michel.ballings@gmail.com>

References


See Also

predict.kernelFactory

Examples

#Credit Approval data available at UCI Machine Learning Repository
data(Credit)
#take subset (for the purpose of a quick example) and train and test
Credit <- Credit[1:100,]
train.ind <- sample(nrow(Credit),round(0.5*nrow(Credit)))

#Train Kernel Factory on training data
kfmodel <- kernelFactory(x=Credit[train.ind,names(Credit)]! = "Response"],
y=Credit[train.ind,"Response"], method=random)

#Deploy Kernel Factory to predict response for test data
#predictedresponse <- predict(kfmodel, newdata=Credit[-train.ind,names(Credit)]! = "Response"})

kFNews

Display the NEWS file

Description

kFNews shows the NEWS file of the kernelFactory package.

Usage

kFNews()

Value

None.
**predict.kernelFactory**

**Author(s)**
Authors: Michel Ballings and Dirk Van den Poel, Maintainer: <Michel.Ballings@GMail.com>

**References**

**See Also**
kernelFactory, predict.kernelFactory

**Examples**
kFNews()

---

**predict.kernelFactory**  Predict method for kernelFactory objects

**Description**
Prediction of new data using kernelFactory.

**Usage**
```r
## S3 method for class 'kernelFactory'
predict(object, newdata = NULL, predict.all = FALSE, ...)
```

**Arguments**
- `object`: An object of class kernelFactory, as created by the function kernelFactory
- `newdata`: A data frame with the same predictors as in the training data.
- `predict.all`: TRUE or FALSE. If TRUE and rp and cp are 1 then the individual predictions of the random forest are returned. If TRUE and any of rp and cp or bigger than 1 then the predictions of all the members are returned.
- `...`: Not used currently.

**Value**
A vector containing the response probabilities.

**Author(s)**
Authors: Michel Ballings and Dirk Van den Poel, Maintainer: <Michel.Ballings@GMail.com>
References


See Also

kernelFactory

Examples

# Credit Approval data available at UCI Machine Learning Repository
data(Credit)
# take subset (for the purpose of a quick example) and train and test
Credit <- Credit[1:100,]
train.ind <- sample(nrow(Credit),round(0.5*nrow(Credit)))

# Train Kernel Factory on training data
kFmodel <- kernelFactory(x=Credit[train.ind, names(Credit)!== "Response"],
y=Credit[train.ind, "Response"], method=random)

# Deploy Kernel Factory to predict response for test data
predictedresponse <- predict(kFmodel, newdata=Credit[-train.ind, names(Credit)!== "Response"])
Index

*Topic classification
   kernelFactory, 2
   predict.kernelFactory, 5
*Topic datasets
   Credit, 2

Credit, 2

kernelFactory, 2, 5, 6
kFNews, 4

predict.kernelFactory, 4, 5, 5