Package ‘fscaret’

May 8, 2018

Type Package
Title Automated Feature Selection from 'caret'
Version 0.9.4.4
Date 2018-05-08
Depends R (>= 3.1.0), caret, gsubfn, hmeasure, utils, parallel
Suggests ada, arm, Boruta, bst, C50, car, caTools, class, Cubist,
e1071, earth (>= 2.2-3), elasticnet, ellipse, etree,
extraTrees, fastICA, foba, gam, gbm (>= 2.1), glmnet (>= 1.8),
hda, HDclassif, Hmisc, ipred, kernlab, kknn, klaR, kohonen,
KRLS, lars, leaps, LogicReg, MASS, mboost, mda, mgcv, mlbench,
nnet, nodeHarvest, obliqueRF, pamr, partDSA, party
(>= 0.9-99992), penalized, penalizedLDA, pls, pROC, proxy,
qrnn, quantregForest, randomForest, RANN, relaxo, rFerns, rocc,
rpart, rrcov, RRF, rrlda, RSNNS, RWeka (>= 0.4-1), sda,
sparseLDA (>= 0.1-1), spls, stepPlr, superpc
Maintainer Jakub Szlek <j.szlek@uj.edu.pl>
License GPL-2 | GPL-3
Description Automated feature selection using variety of models
provided by 'caret' package.
This work was funded by Poland-Singapore bilateral cooperation
project no 2/3/POL-SIN/2012.
NeedsCompilation no
Repository CRAN
Date/Publication 2018-05-08 08:15:17 UTC
Author Jakub Szlek [aut, cre],
Aleksander Mendyk [ctb]

R topics documented:

fscaret-package ....................................................... 2
classVarImp .......................................................... 3
dataPreprocess ....................................................... 4
Automated feature selection caret (fscaret)

Description

This package provides fast and automated feature selection based on caret package modeling methods. The main advantage of this extension is that it requires minimum user involvement. Also, the variety of used methods in combination with the scaling according to RMSE or MSE obtained from models benefits the user. The idea is based on the assumption that the variety of models will balance the roughness of calculations (default model settings are applied). On Windows OS the time limiting function is off, multicore functionality is enabled via parLapply() function of package 'parallel'. Acknowledgments:

This work was funded by Poland-Singapore bilateral cooperation project no 2/3/POL-SIN/2012

Details

Package: fscaret
Type: Package
Version: 0.9.4.2
Date: 2017-12-07
License: GPL-2 | GPL-3

Author(s)

Jakub Szlek <j.szlek@uj.edu.pl> Contributions from Aleksander Mendyk, also stackoverflow and r-help@r-project.org mailing list community.
Maintainer: Jakub Szlek <j.szlek@uj.edu.pl>.
References


See Also
train, trainControl, rfeControl by Max Kuhn <Max.Kuhn at pfizer.com> and predict base utilities

classVarImp  classVarImp

description

The function uses the caret package advantage to perform fitting of numerous classification models.

Usage

classVarImp(model, xTrain, yTrain, xTest,
   fitControl, myTimeLimit, no.cores,
   lk_col, suppress.output)

Arguments

model  Chosed models as called from function fscaret(), argument Used.funcClassPred.
xTrain  Training data set, data frame of input vector
yTrain  Training data set, vector of observed outputs, must be in binary form 0/1.
xTest  Testing data set, data frame of input vector
fitControl  Fitting controls passed to caret function
myTimeLimit  Time limit in seconds for single model fitting
no.cores  Number of used cores for calculations
lk_col  Number of columns for whole data set (inputs + output)
supress.output  If TRUE output of models are supressed.

Author(s)

Jakub Szlek and Aleksander Mendyk

References

Description

The functionality is realized in two main steps:

1. Check for near zero variance predictors and flag as near zero if:
   (a) the percentage of unique values is less than 20
   (b) the ratio of the most frequent to the second most frequent value is greater than 20,
2. Check for susceptibility to multicollinearity
   (a) Calculate correlation matrix
   (b) Find variables with correlation 0.9 or more and delete them

Usage

dataPreprocess(trainMatryca_nr, testMatryca_nr, labelsFrame, lk_col, lk_row, with.labels)

Arguments

trainMatryca_nr  Input training data matrix
testMatryca_nr    Input testing data matrix
labelsFrame      Transposed data frame of column names
lk_col            Number of columns
lk_row            Number of rows
with.labels       If with.labels=TRUE, additional data frame with preprocessed inputs corresponding to original data set column numbers as output is generated

Author(s)

Jakub Szlek and Aleksander Mendyk

References


Examples

library(fscaret)

# Create data sets and labels data frame
trainMatrix <- matrix(rnorm(150*120,mean=10,sd=1), 150, 120)
# Adding some near-zero variance attributes

temp1 <- matrix(rnorm(150,0.0001,0.0005), 150, 12)

# Adding some highly correlated attributes

sampleColIndex <- sample(ncol(trainMatrix), size=10)
temp2 <- matrix(trainMatrix[,sampleColIndex]*2, 150, 10)

# Output variable

output <- matrix(rnorm(150,mean=10,sd=1), 150, 1)

trainMatrix <- cbind(trainMatrix,temp1,temp2,output)

colnames(trainMatrix) <- paste("X",c(1:ncol(trainMatrix)),sep="")

# Subset test data set

testMatrix <- trainMatrix[sample(round(0.1*nrow(trainMatrix))],]

labelsDF <- data.frame("Labels"=paste("X",c(1:(ncol(trainMatrix)-1)),sep=""))

lk_col <- ncol(trainMatrix)
lk_row <- nrow(trainMatrix)

with.labels = TRUE

testRes <- dataPreprocess(trainMatrix, testMatrix, 
labelsDF, lk_col, lk_row, with.labels)

summary(testRes)

# Selected attributes after data set preprocessing

testRes$labelsDF

# Training and testing data sets after preprocessing

testRes$trainMatrixca

testRes$testMatrixca

dataset.test

Example testing data set

Description

The data set after preprocessing, which resulted in 29 inputs. Original data set was obtained in literature survey with 298 inputs. Input: chemical descriptors and characteristics of 8 PLGA mi-
croparicles formulation. Output: mean particle size of PLGA microparticles Number of attributes 29, single output.

Usage
data(dataset.test)

Format
data.frame

Details
Literature survey yielded 68 formulations of PLGA microspheres with protein as active pharma-
aceutical ingredient. In vitro release profiles as well as formulation characteristics and composition
were derived from articles. Chemical descriptors were obtained using Marvin ChemAxon software
(cxcalc plugin). The final data base consisted of 298 inputs and single output mean particle size.

Source
1. Kang F, Singh J. Effect of additives on the release of a model protein from PLGA micro-
spheres. AAPS PharmSciTech 2001(2)4, 1-7
2. Zhou XL et al. Pharmacokinetic and pharmacodynamic profiles of recombinant human erythropoietin-
loaded poly(lactic-co-glycolic acid) microspheres in rats. ActaPharmSinica 2012(33), 137-
144
3. Dongmei F et al. Mesoporous Silicon-PLGA Composite Microspheres for the Double Con-
trolled Release of Biomolecules for Orthopedic Tissue Engineering. Adv Funct Mater 2012(22),
282-293.
5. Blanco D et al. Protein encapsulation and release from poly(lactide-co-glycolide) micro-
spheres: effect of the protein and polymer properties and of the co-encapsulation of surfac-
6. Morita T et al. Applicability of various amphiphilic polymers to the modification of protein re-
lease kinetics from biodegradable reservoir-type microspheres. Eur J Pharm Biopharm. 2001,
51, 45-53.
7. Mok H et al. Water free microencapsulation of proteins within PLGA microparticles by spray
drying using PEG assisted protein solubilization technique in organic solvent. Eur J Pharm
Biopharm. 2008, 70, 137-144.
2009, 37, 477-485.
body response in mice immunized by subcutaneous and intranasal routes. Eur J Pharm Sci.
2011 44, 200-206

**dataset.train**

**Example training data set**

**Description**

The data set after preprocessing, which resulted in 29 inputs. Original data set was obtained in literature survey with 298 inputs. Input: chemical descriptors and characteristics of 8 PLGA microparticles formulation. Output: mean particle size of PLGA microparticles Number of attributes 29, single output.
Usage

data(dataset.train)

Format

data.frame

Details

Literature survey yielded 68 formulations of PLGA microspheres with protein as active pharmacutical ingredient. In vitro release profiles as well as formulation characteristics and composition were derived from articles. Chemical descriptors were obtained using Marvin ChemAxon software (cxcalc plugin). The final data base consisted of 298 inputs and single output mean particle size.

Source


---

**fscaret**

**feature selection caret**

**Description**

Main function for fast feature selection. It utilizes other functions as regPredImp or impCalc to obtain results in a list of data frames.
Usage

fscaret(trainDF, testDF, installReqPckg = FALSE, preprocessData = FALSE, with.labels = TRUE, classPred = FALSE, regPred = TRUE, skel_outfile = NULL, impCalcMet = "RMSE&MSE", myTimeLimit = 24 * 60 * 60, Used.funcRegPred = NULL, Used.funcClassPred = NULL, no.cores = NULL, method = "boot", returnResamp = "all", missData=NULL, supress.output=FALSE, saveModel=FALSE, lvlScale=FALSE, ...)

Arguments

trainDF: Data frame of training data set, MISO (multiple input single output) type

testDF: Data frame of testing data set, MISO (multiple input single output) type

installReqPckg: If TRUE prior to calculations it installs all required packages, please be advised to be logged as root (admin) user

preprocessData: If TRUE data preprocessing is performed prior to modeling

with.labels: If TRUE header of the input files are read

classPred: If TRUE classification models are applied. Please be advised that importance is scaled according to F-measure regardless impCalcMet settings.

regPred: If TRUE regression models are applied

skel_outfile: Skeleton output file, e.g. skel_outfile=c("_myoutput_"))

impCalcMet: Variable importance calculation scaling according to RMSE and MSE, for both please enter impCalcMet="RMSE&MSE"

myTimeLimit: Time limit in seconds for single model development

Used.funcRegPred: Vector of regression models to be used, for all available models please enter Used.funcRegPred="all"

Used.funcClassPred: Vector of classification models to be used, for all available models please enter Used.funcClassPred="all"

no.cores: Number of cores to be used for modeling, if NULL all available cores are used, should be numeric type or NULL

method: Method passed to fitControl of caret package

returnResamp: Returned resampling method passed to fitControl of caret package

missData: Handling of missing data values. Possible values: "delRow" - delete observations with missing values, "delCol" - delete attributes with missing values, "meanCol" - replace missing values with column mean.

supress.output: If TRUE output of modeling phase by caret functions are supressed. Only info which model is currently calculated and resulting variable importance.

saveModel: Logical value [TRUE/FALSE] if trained model should be embedded in final model.

lvlScale: Logical value [TRUE/FALSE] if additional scaling should be applied. For more information please refer to impCalc().

...: Additional arguments, preferably passed to fitControl of caret package
Value

- $ModelPred: List of outputs from caret model fitting
- $VarImp: Data frames of variable importance and corresponding trained models
- $PPLabels: Data frame of resulting preprocessed data set with original input numbers and names
- $PTrainDF: Training data set after preprocessing
- $PTestDF: Testing data set after preprocessing
- $VarImp$model: Trained models

Note

Be advised when using fscaret function as it requires hard disk operations for saving fitted models and data frames. Files are written in R temp session folder, for more details see tempdir(), getwd() and setwd()

Author(s)

Jakub Szlek and Aleksander Mendyk

References


Examples

```r
if(!(Sys.info()['sysname']!="SunOS"){

library(fscaret)

# Load data sets
data(dataset.train)
data(dataset.test)

requiredPackages <- c("R.utils", "gsubfn", "ipred", "caret", "parallel", "MASS")

if(.Platform$OS.type="windows"){

myFirstRES <- fscaret(dataset.train, dataset.test, installReqPckg=FALSE, preprocessData=FALSE, with.labels=TRUE, classPred=FALSE, regPred=TRUE, skel_outfile=NULL, impCalcMet="RMSE&RMSE", myTimeLimit=4, Used.funcRegPred=c("lm"), Used.funcClassPred=NULL, no.cores=1, method="boot", returnResamp="all", suppress.output=TRUE, saveModel=FALSE)

} else {

}
```
myCores <- 2

myFirstRES <- fscaret(dataset.train, dataset.test, installReqPckg=FALSE, preprocessData=FALSE, with.labels=TRUE, classPred=FALSE, regPred=TRUE, skel_outfile=NULL, impCalcMet="RMSE\&MSE", myTimeLimit=4, used.funcRegPred=c("lm","ppr"), used.funcClassPred=NULL, no.cores=myCores, method="boot", returnResamp="all", supress.output=TRUE, saveModel=FALSE)

# Results
myFirstRES

funcClass.all

Classification methods used.

Description
Vector of all classification methods used in solving problems by caret

Usage
data(funcClassPred)

Format
vector

funcReg.all

All regression methods used

Description
Vector of all regression methods used in solving problems by caret

Usage
data(funcRegPred)

Format
vector
impCalc function is designed to scale variable importance according to MSE and RMSE calculations. It also stores the raw MSE, RMSE, F-measure and developed models if saveModel=TRUE. impCalc is low-level function, it shouldn’t be used alone unless user has trained models from caret package stored in RData files.

Usage

impCalc(skel_outfile, xTest, yTest, lk_col, labelsFrame, with.labels, regPred, classPred, saveModel, lvlScale)

Arguments

- **skel_outfile**: Skeleton name of output file
- **xTest**: Input vector of testing data set
- **yTest**: Output vector of testing data set
- **lk_col**: Number of columns of whole data set
- **labelsFrame**: Labels to sort variable importance
- **with.labels**: Pass with.labels argument. It is advised to ALWAYS use labels as in some cases VarImp returns importance in descending values. If you insist turning with.labels FALSE, then make sure data base contains pure data and you read it (read.csv) to data.frame with option header=FALSE.
- **regPred**: Indicating if regression predictions are computed. Logical value [TRUE/FALSE]. If regPred is set TRUE, then classPred should be set FALSE.
- **classPred**: Indicating if classification predictions are computed. Possible values TRUE/FALSE. If classPred is set TRUE, then regPred should be set FALSE. Please be advised that importance is scaled according to F-measure.
- **saveModel**: Logical value [TRUE/FALSE] if trained model should be embedded in final model.
- **lvlScale**: Indicating if use additional scaling. The option is especially usefull when large number of features are getting NA’s or are not included in feature ranking. It levels the scores of the features taking the overall number of features. Default value is FALSE. Logical value [TRUE/FALSE].

Details

impCalc function lists RData files in working directory assuming there are only models derived by caret. In a loop function loads models and tries to get the variable importance.
Author(s)

Jakub Szlek and Aleksander Mendyk

Examples

```r
## Not run:
#
# Hashed to comply with new CRAN check
#
library(fscaret)

# Load dataset
data(dataset.train)
data(dataset.test)

# Make objects
trainDF <- dataset.train
testDF <- dataset.test
model <- c("lm","Cubist")
fitControl <- trainControl(method = "boot", returnResamp = "all")
myTimeLimit <- 5
no.cores <- 2
suppress.output <- TRUE
skel_outfile <- paste("_default_",sep="")
mySystem <- .Platform$OS.type
with.labels <- TRUE
redPred <- TRUE
classPred <- FALSE
saveModel <- FALSE
lvlScale <- FALSE

if(mySystem=="windows"){
  no.cores <- 1
}

# Scan dimensions of trainDF [lk_row x lk_col]
lk_col = ncol(trainDF)
lk_row = nrow(trainDF)

# Read labels of trainDF
labelsFrame <- as.data.frame(colnames(trainDF))
labelsFrame <- cbind(1:ncol(trainDF),labelsFrame)
# Create a train data set matrix
trainMatryca_nr <- matrix(data=NA,nrow=lk_row,ncol=lk_col)

row=0
col=0

for(col in 1:(lk_col)) {
  for(row in 1:(lk_row)) {
    trainMatryca_nr[row,col] <- (as.numeric(trainDF[row,col])
```
imputeMean

# Pointing standard data set train
xTrain <- data.frame(trainMatryca_nr[, -lk_col])
yTrain <- as.vector(trainMatryca_nr[, lk_col])

#--------Scan dimensions of trainDataframe1 [lk_row x lk_col]
lk_col_test = ncol(testDF)
lk_row_test = nrow(testDF)

testMatryca_nr <- matrix(data=NA, nrow=lk_row_test, ncol=lk_col_test)

row=0
col=0

for (col in 1:(lk_col_test)) {
  for (row in 1:(lk_row_test)) {
    testMatryca_nr[row, col] <- as.numeric(testDF[row, col])
  }
}

# Pointing standard data set test
xTest <- data.frame(testMatryca_nr[, -lk_col])
yTest <- as.vector(testMatryca_nr[, lk_col])

# Calling low-level function to create models to calculate on
myVarImp <- regVarImp(model, xTrain, yTrain, xTest, fitControl, myTimeLimit, no.cores, lk_col, suppress.output, mySystem)

myImpCalc <- impCalc(skel_outfile, xTest, yTest, lk_col, labelsFrame, with.labels, redPred, classPred, saveModel, lvlScale)

## End(Not run)

---

imputeMean

**Description**

Secondary function imputes the mean to columns with NA data.

**Usage**

```r
impute.mean(x)
```
Arguments

x  

a vector to calculate mean

Author(s)

Jakub Szlek and Aleksander Mendyk

Examples

library(fscaret)

# Make sample matrix
testData <- matrix(data=rep(1:5), ncol=10, nrow=15)

# Replace random values with NA's
n <- 15
replace <- TRUE
set.seed(1)

rand.sample <- sample(length(testData), n, replace=replace)
testData[rand.sample] <- NA

# Print out input matrix
testData

# Record cols with missing values
missing.colsTestMatrix <- which(colSums(is.na(testData))>0)

for(i in 1:length(missing.colsTestMatrix)){
  rowToReplace <- missing.colsTestMatrix[i]
testData[,rowToReplace] <- impute.mean(testData[,rowToReplace])
}

# Print out matrix with replaced NA's by column mean
testData

installPckg

installPckg

Description

Function installs the packages that are listed in data(requiredPackages). The function is called within fscaret function. If argument "installReqPckg = TRUE" the function installs required packages.
**MSE**

**Usage**

`installPckg(requiredPackages)`

**Arguments**

- `requiredPackages`: Vector of packages to be installed

**Details**

Be advised setting "installReqPckg = TRUE" installs packages in your home directory (.R). To install packages for all users please login as root (admin).

**Author(s)**

Jakub Szlek and Aleksander Mendyk

---

**Description**

Function calculates mean squared error as predicted vs. observed

**Usage**

`MSE(vect1, vect2, rows_no)`

**Arguments**

- `vect1`: Numeric vector of predicted values
- `vect2`: Numeric vector of observed values
- `rows_no`: Number of observations

**Author(s)**

Jakub Szlek and Aleksander Mendyk
Description

The function uses the caret package advantage to perform fitting of numerous regression models.

Usage

```r
regVarImp(model, xTrain, yTrain, xTest,
  fitControl, myTimeLimit, no.cores,
  lk_col, supress.output)
```

Arguments

- `model`: Chosed models as called from function fscaret(), argument Used.funcRegPred.
- `xTrain`: Training data set, data frame of input vector
- `yTrain`: Training data set, vector of observed outputs
- `xTest`: Testing data set, data frame of input vector
- `fitControl`: Fitting controls passed to caret function
- `myTimeLimit`: Time limit in seconds for single model fitting
- `no.cores`: Number of used cores for calculations
- `lk_col`: Number of columns for whole data set (inputs + output)
- `supress.output`: If TRUE output of models are supressed.

Author(s)

Jakub Szlek and Aleksander Mendyk

References

**Description**

Character vector of names of required packages to fully take advantage of fscaret

**Usage**

data(requiredPackages)

**Format**

vector

**Examples**

data(requiredPackages)

---

**RMSE**

Function calculates root mean squared error.

**Usage**

RMSE(vect1, vect2, rows_no)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vect1</td>
<td>Numeric vector of predicted values</td>
</tr>
<tr>
<td>vect2</td>
<td>Numeric vector of observed values</td>
</tr>
<tr>
<td>rows_no</td>
<td>Number of observations</td>
</tr>
</tbody>
</table>

**Author(s)**

Aleksander Mendyk
Description

This function limits elapsed time spent on single model development. It uses low-level functions of parallel package and sets the fork process with time limit. If the result is not returned within set time, it kills fork. Function shouldn’t be called from R console. The function is not used under Windows OS. Only Unix-like systems have fork functionality.

Usage

timeout(..., seconds)

Arguments

... Expression to be time limited
seconds Number of seconds

Author(s)

Original code by Jeroen Ooms <jeroen.ooms at stat.ucla.edu> of OpenCPU package. Modifications by Jakub Szlek and Aleksander Mendyk.
Index

*Topic arith
  MSE, 17
  RMSE, 19

*Topic array
  fscaret, 9

*Topic datasets
  dataset.test, 5
  dataset.train, 7
  funcClass.all, 12
  funcReg.all, 12
  requiredPackages, 19

*Topic data
  installPckg, 16

*Topic design
  impCalc, 13

*Topic error
  timeout, 20

*Topic iteration
  fscaret, 9

*Topic logic
  imputeMean, 15

*Topic math
  imputeMean, 15
  MSE, 17
  RMSE, 19

*Topic methods
  fscaret, 9
  timeout, 20

*Topic models
  classVarImp, 3
  impCalc, 13
  regVarImp, 18

*Topic optimize
  fscaret, 9

*Topic package
  fscaret-package, 2
  installPckg, 16

*Topic robust
  classVarImp, 3
  dataPreprocess, 4
  regVarImp, 18

*Topic univar
  dataPreprocess, 4
  classVarImp, 3
  dataPreprocess, 4
  dataset.test, 5
  dataset.train, 7
  fscaret, 9
  fscaret-package, 2
  funcClass.all, 12
  funcClassPred (funcClass.all), 12
  funcReg.all, 12
  funcRegPred (funcReg.all), 12
  impCalc, 13
  impute.mean (imputeMean), 15
  imputeMean, 15
  installPckg, 16
  MSE, 17
  predict, 3
  regVarImp, 18
  requiredPackages, 19
  rfeControl, 3
  RMSE, 19
  timeout, 20
  train, 3
  trainControl, 3