Package ‘fscaret’

October 6, 2014

Type Package

Title Automated caret feature selection

Version 0.8.6.3

Date 2014-10-5

Depends R (>= 3.0.0), caret, gsubfn, utils, parallel

Suggests ada, arm, Boruta, bst, C50, car, caTools, class, Cubist,e1071, earth (>= 2.2-3), elasticnet, ellipse, evtree,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,neuralnet, 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

Author Jakub Szlek <j.szlek@uj.edu.pl>, acknowledgments to Aleksander Mendyk, contributions from stackoverflow and r-help@r-project.org mailing list community. Acknowledgments: This work was funded by Poland-Singapore bilateral cooperation project no 2/3/POL-SIN/2012.

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

NeedsCompilation no

Repository CRAN

Date/Publication 2014-10-06 13:48:52

R topics documented:

`fscaret-package` *Automated feature selection caret (fscaret)*

Description

This package provide 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 profit 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 also the number of cores used is set to 1.

Details

Package:	fscaret
Type:	Package
Version:	0.8.6.3
Date:	2014-10-5
License:	GPL-2 | GPL-3

Author(s)

Jakub Szlek <j.szlek@uj.edu.pl>, acknowledgments to Aleksander Mendyk, contributions from stackoverflow and r-help@r-project.org mailing list community.
Maintainer: Jakub Szlek <j.szlek@uj.edu.pl>.
Acknowledgments:
This work was funded by Poland-Singapore bilateral cooperation project no 2/3/POL-SIN/2012

References

Kuhn M. (2008) Building Predictive Models in R Using the caret Package *Journal of Statistical Software* **28(5)** `http://www.jstatsoft.org/`.
Szlek J, Paclawski A, Lau R, Jachowicz R, Mendyk A. Heuristic modeling of macromolecule release from PLGA microspheres. *International Journal of Nanomedicine.* **2013:8(1); 4601 - 4611**. `http://www.dovepress.com/international-journal-of-nanomedicine-journal`.

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, supress.output, mySystem)
```

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
`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.
`mySystem`	Called from fscaret() result of function .Platform$OS.type

Author(s)

Jakub Szlek and Aleksander Mendyk

References

Kuhn M. (2008) Building Predictive Models in R Using the caret Package *Journal of Statistical Software* **28(5)** http://www.jstatsoft.org/.

Examples

```
#
# Code block prepared to impement Class variable importance
#
# pre-ALPHA VERSION!
#
# Code block prepared to impement Class variable importance
#
```

`dataPreprocess` *dataPreprocess*

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

Kuhn M. (2008) Building Predictive Models in R Using the caret Package *Journal of Statistical Software* **28(5)** http://www.jstatsoft.org/.

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(runif(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$trainMatryca
testRes$testMatryca
```

dataset.test *Example testing data set*

Description

The data set after preprocessing, which resulted in 29 inptus. Original data set was obtained in literature survey with 298 inputs. Input: chemical descriptors and characteristics of 8 PLGA microparicles 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 pharmaceuticla ingridient. 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 microspheres. 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 Controlled Release of Biomolecules for Orthopedic Tissue Engineering. Adv Funct Mater 2012(22), 282-293.
4. Kim T.H. et al. Pegylated recombinant human epidermal growth factor (rhEGF) for sustained release from biodegradable PLGA microspheres. Biomater 2002,23, 2311-2317.
5. Blanco D et al. Protein encapsulation and release from poly(lactide-co-glycolide) microspheres: effect of the protein and polymer properties and of the co-encapsulation of surfactants. Eur J Pharm Biopharm. 1998, 45, 285-294.

6. Morita T et al. Applicability of various amphiphilic polymers to the modification of protein release 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.
8. Buske J et al. Influence of PEG in PEG-PLGA microspheres on particle properties and protein release. Eur J Pharm Biopharm. 2012, 81, 57-63.
9. Corrigan OI et al. Quantifying drug release from PLGA nanoparticulates. Eur J Pharm Sci. 2009, 37, 477-485.
10. Puras G. et al. Encapsulation of A-beta-(1-15) in PLGA microparticles enhances serum antibody response in mice immunized by subcutaneous and intranasal routes. Eur J Pharm Sci. 2011 44, 200-206
11. Tran VT et al. Protein loaded PLGA PEG PLGA microspheres A tool for cell therapy. Eur J Pharm Sci. 2012, 45, 128-137.
12. Kim HK et al. Microencapsulation of dissociable human growth hormone aggregates within poly(D,L-lactic-co-glycolic acid) microparticles for sustained release. Int J Pharm. 2001, 229, 107-116
13. Han Y et al. Insulin nanoparticle preparation and encapsulation into poly(lactic-co-glycolic acid) microspheres by using an anhydrous system. Int J Pharm. 2009, 378, 159-166
14. Liu Q et al. In vitro and in vivo study of thymosin alpha1 biodegradable in situ forming poly(lactide-co-glycolide) implants. Int J Pharm. 2010, 397, 122-129.
15. He J et al. Stabilization and encapsulation of recombinant human erythropoietin into PLGA microspheres using human serum albumin as a stabilizer. Int J Pharm. 2011, 416, 69-76.
16. Gasper MM et al. Formulation of L-asparaginase-loaded poly(lactide-co-glycolide) nanoparticles: influence of polymer properties on enzyme loading, activity and in vitro release. J Control Release. 1998, 52, 53-62.
17. Kawashima Y et al. Pulmonary delivery of insulin with nebulized DL-lactide/glycolide copolymer (PLGA) nanospheres to prolong hypoglycemic effect. J Control Release. 1999, 62, 279-287.
18. Geng Y et al. Formulating erythropoietin-loaded sustained-release PLGA microspheres without protein aggregation. J Control Release. 2008, 130, 259-265.
19. Ungaro F et al. Insulin-loaded PLGA/cyclodextrin large porous particles with improved aerosolization properties: in vivo deposition and hypoglycaemic activity after delivery to rat lungs. J Control Release. 2009, 135(1), 25-34.
20. Iwata M et al. In vitro and in vivo release properties of brilliant blue and tumour necrosis factor-alpha (TNF-alpha) from poly(D,L-lactic-co-glycolic acid) multiphase microspheres. J Microencapsul. 1999, 16(6), 777-792.
21. Jiang HL et al. Improvement of protein loading and modulation of protein release from poly(lactide-co-glycolide) microspheres by complexation of proteins with polyanions. J Microencapsul. 2004, 21(6), 615-624
22. Pirooznia N et al. Encapsulation of alpha-1 antitrypsin in PLGA nanoparticles: in vitro characterization as an effective aerosol formulation in pulmonary diseases. J Nanobiotechnology. 2012, 10(1), 20-35.

23. Castellanos IJ et al. Effect of cyclodextrins on alpha-chymotrypsin stability and loading in PLGA microspheres upon S/O/W encapsulation. J Pharm Sci. 2006, 95(4), 849-858.
24. Brodbeck KJ et al. Sustained release of human growth hormone from PLGA solution depots. Pharm Res. 2009, 16(12), 1825-1829.

Examples

```
library(fscaret)

data(dataset.test)

dataset.test
```

`dataset.train` *Example training data set*

Description

The data set after preprocessing, which resulted in 29 inptus. Original data set was obtained in literature survey with 298 inputs. Input: chemical descriptors and characteristics of 8 PLGA microparicles 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 pharmaceuticla ingridient. 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 microspheres. 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 Controlled Release of Biomolecules for Orthopedic Tissue Engineering. Adv Funct Mater 2012(22), 282-293.
4. Kim T.H. et al. Pegylated recombinant human epidermal growth factor (rhEGF) for sustained release from biodegradable PLGA microspheres. Biomater 2002,23, 2311-2317.
5. Blanco D et al. Protein encapsulation and release from poly(lactide-co-glycolide) microspheres: effect of the protein and polymer properties and of the co-encapsulation of surfactants. Eur J Pharm Biopharm. 1998, 45, 285-294.
6. Morita T et al. Applicability of various amphiphilic polymers to the modification of protein release 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.
8. Buske J et al. Influence of PEG in PEG-PLGA microspheres on particle properties and protein release. Eur J Pharm Biopharm. 2012, 81, 57-63.
9. Corrigan OI et al. Quantifying drug release from PLGA nanoparticulates. Eur J Pharm Sci. 2009, 37, 477-485.
10. Puras G. et al. Encapsulation of A-beta-(1-15) in PLGA microparticles enhances serum antibody response in mice immunized by subcutaneous and intranasal routes. Eur J Pharm Sci. 2011 44, 200-206
11. Tran VT et al. Protein loaded PLGA PEG PLGA microspheres A tool for cell therapy. Eur J Pharm Sci. 2012, 45, 128-137.
12. Kim HK et al. Microencapsulation of dissociable human growth hormone aggregates within poly(D,L-lactic-co-glycolic acid) microparticles for sustained release. Int J Pharm. 2001, 229, 107-116
13. Han Y et al. Insulin nanoparticle preparation and encapsulation into poly(lactic-co-glycolic acid) microspheres by using an anhydrous system. Int J Pharm. 2009, 378, 159-166
14. Liu Q et al. In vitro and in vivo study of thymosin alpha1 biodegradable in situ forming poly(lactide-co-glycolide) implants. Int J Pharm. 2010, 397, 122-129.
15. He J et al. Stabilization and encapsulation of recombinant human erythropoietin into PLGA microspheres using human serum albumin as a stabilizer. Int J Pharm. 2011, 416, 69-76.
16. Gasper MM et al. Formulation of L-asparaginase-loaded poly(lactide-co-glycolide) nanoparticles: influence of polymer properties on enzyme loading, activity and in vitro release. J Control Release. 1998, 52, 53-62.
17. Kawashima Y et al. Pulmonary delivery of insulin with nebulized DL-lactide/glycolide copolymer (PLGA) nanospheres to prolong hypoglycemic effect. J Control Release. 1999, 62, 279-287.
18. Geng Y et al. Formulating erythropoietin-loaded sustained-release PLGA microspheres without protein aggregation. J Control Release. 2008, 130, 259-265.
19. Ungaro F et al. Insulin-loaded PLGA/cyclodextrin large porous particles with improved aerosolization properties: in vivo deposition and hypoglycaemic activity after delivery to rat lungs. J Control Release. 2009, 135(1), 25-34.

20. Iwata M et al. In vitro and in vivo release properties of brilliant blue and tumour necrosis factor-alpha (TNF-alpha) from poly(D,L-lactic-co-glycolic acid) multiphase microspheres. J Microencapsul. 1999, 16(6), 777-792.

21. Jiang HL et al. Improvement of protein loading and modulation of protein release from poly(lactide-co-glycolide) microspheres by complexation of proteins with polyanions. J Microencapsul. 2004, 21(6), 615-624

22. Pirooznia N et al. Encapsulation of alpha-1 antitrypsin in PLGA nanoparticles: in vitro characterization as an effective aerosol formulation in pulmonary diseases. J Nanobiotechnology. 2012, 10(1), 20-35.

23. Castellanos IJ et al. Effect of cyclodextrins on alpha-chymotrypsin stability and loading in PLGA microspheres upon S/O/W encapsulation. J Pharm Sci. 2006, 95(4), 849-858.

24. Brodbeck KJ et al. Sustained release of human growth hormone from PLGA solution depots. Pharm Res. 2009, 16(12), 1825-1829.

Examples

```
library(fscaret)

data(dataset.train)

dataset.train
```

---

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, ...)
```

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 (for v0.8 it is not available)
`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 v0.8 it is not available)
`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.
`...`	Additional arguments, preferably passed to fitControl of caret package

Value

`$ModelPred`	List of outputs from caret model fitting
`$VarImp`	Data frames of variable importance
`$PPlabels`	Data frame of resulting preprocessed data set with original input numbers and names
`$PPTrainDF`	Training data set after preprocessing
`$PPTestDF`	Testing data set after preprocessing

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

Kuhn M. (2008) Building Predictive Models in R Using the caret Package *Journal of Statistical Software* **28(5)** http://www.jstatsoft.org/.

Examples

```
library(fscaret)

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

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

mySystem <- .Platform$OS.type

if(mySystem=="windows"){

myCores <- 1

} 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=5,
                  Used.funcRegPred=c("lm","pls","pcr"), Used.funcClassPred=NULL,
                  no.cores=myCores, method="boot", returnResamp="all",
                  supress.output=TRUE)

# Training data set after preprocessing
myFirstRES$PPTrainDF

# Testing data set after preprocessing
myFirstRES$PPTestDF


# Model predictions
myFirstRES$ModelPred


# Variable importance after scaling according to RMSE and MSE
myFirstRES$VarImp


# Reduced input vector (data set) after preprocessing
myFirstRES$PPlabels
```

`funcClass.all` *Classification methods used. Warning, in fscaret version 0.8.6.1 methods for variable importence ranking harvesting have not been implemented!*

Description

Vector of all classification methods used in solving problems by caret

Usage

```
data(funcClassPred)
```

Format

vector

Examples

```
# Load library
library(fscaret)

# Load data set
data(funcClassPred)

# Print out object
funcClassPred
```

`funcReg.all` *All regression methods used*

Description

Vector of all regression methods used in solving problems by caret

Usage

```
data(funcRegPred)
```

Format

vector

Examples

```
# Load library
library(fscaret)

# Load data set
data(funcRegPred)

# Print out object
funcRegPred
```

`impCalc` *impCalc*

Description

impCalc function is designed to scale variable importance according to MSE and RMSE calculations. It also stores the raw MSE and RMSE derived from models. impCalc function shouldn't be used alone unless user has trained models from caret package in RData files.

Usage

```
impCalc(skel_outfile, xTest, yTest, lk_col,labelsFrame,with.labels)
```

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.

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

```
#
# Hashed to comply with new CRAN check
#
library(fscaret)

# Read working directory
myWD <- getwd()

# Set working directory to tmp
setwd(tempdir())

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

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


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))

# 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]))
    }
}
```

```
# 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 lower function to create models to calculate on
myVarImp <- regVarImp(model, xTrain, yTrain, xTest,
    fitControl, myTimeLimit, no.cores, lk_col,
    supress.output, mySystem)


myImpCalc <- impCalc(skel_outfile, xTest, yTest, lk_col,labelsFrame,with.labels)
```

---

imputeMean *imputeMean*

Description

Secondary function imputes the mean to columns with NA data.

Usage

```
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.

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

MSE *MSE*

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

Examples

```
## The function is currently defined as
function (vect1, vect2, rows_no)
{
    result = 0
    pred <- 0
    obs <- 0
    for (i in 1:rows_no) {
        result <- result + (vect1[i] - vect2[i])^2
    }
    result <- (result/rows_no)
    return(result)
  }
```

regVarImp *regVarImp*

Description

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

Usage

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

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.
`mySystem`	Called from fscaret() result of function .Platform$OS.type

Author(s)

Jakub Szlek and Aleksander Mendyk

References

Kuhn M. (2008) Building Predictive Models in R Using the caret Package *Journal of Statistical Software* **28(5)** `http://www.jstatsoft.org/`.

Examples

```
# Hashed to comply with new CRAN check
#
# Load library
library(fscaret)

# Read working directory
myWD <- getwd()
```

```
# Set working directory to tmp
setwd(tempdir())

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

# Make objects
trainDF <- dataset.train
testDF <- dataset.test
model <- c("lm","pls","pcr")
fitControl <- trainControl(method = "boot", returnResamp = "all")
myTimeLimit <- 5
no.cores <- 2
supress.output <- TRUE

mySystem <- .Platform$OS.type

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))

# 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]))
    }
}

# 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,ncol=lk_col)

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])


myVarImp <- regVarImp(model, xTrain, yTrain, xTest,
    fitControl, myTimeLimit, no.cores, lk_col,
    supress.output, mySystem)

summary(myVarImp)

print(myVarImp)

# Get back to previous working directory
setwd(myWD)
```

`requiredPackages` *requiredPackages*

Description

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

Usage

```
data(requiredPackages)
```

Format

vector

Examples

```
data(requiredPackages)
```

RMSE *RMSE*

Description

Function calculates root mean squared error.

Usage

```
RMSE(vect1, vect2, rows_no)
```

Arguments

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

Author(s)

Aleksander Mendyk

Examples

```
## The function is currently defined as
function (vect1, vect2, rows_no)
{
    result = 0
    obs <- 0
    pred <- 0
    for (i in 1:rows_no) {
        result <- result + (vect1[i] - vect2[i])^2
    }
    result <- (result/rows_no)^0.5
    return(result)
  }
```

timeout *timeout*

Description

This function limits the cpu time spent on single model fitting. It simply sets the killing process of itself to sleep for chosen number of seconds. Shouldn't be called from R console. The function is not used under Windows OS.

Usage

```
timeout(expr, seconds, my.pid)
```

Arguments

`expr`	Expression to be time limited
`seconds`	Number of seconds
`my.pid`	Process PID

Author(s)

Jakub Szlek and Aleksander Mendyk with contribution to nabble users

Examples

```
## The function is currently defined as
function (expr, seconds, my.pid)
{
    killer.pid <- system(intern = TRUE, paste(" (sleep", seconds,
        " ; kill -9", my.pid, ")>/dev/null&\n echo $!", sep = " "))
    on.exit(system(paste("kill", killer.pid, "> /dev/null 2>&1",
        sep = " ")))
    withCallingHandlers(expr, interrupt = function(...) {
        stop("Timedout", call. = FALSE)
    })
  }
```

Index