Package

March 26, 2019

Type Package

Title Machine Learning Automation

Version 0.1.0

Author Mohamed Maher Zenhom Abdelrahman

Maintainer Mohamed Maher Zenhom Abdelrahman <m.maher525@gmail.com>

Description

SmartML is a meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. Being meta-learning based, the framework is able to simulate the role of the machine learning expert. In particular, the framework is equipped with a continuously updated knowledge base that stores information about the meta-features of all processed datasets along with the associated performance of the different classifiers and their tuned parameters. Thus, for any new dataset, SmartML automatically extracts its meta features and searches its knowledge base for the best performing algorithm to start its optimization process. In addition, SmartML makes use of the new runs to continuously enrich its knowledge base to improve its performance and robustness for future runs.

License GPL-2

Encoding UTF-8

LazyData true

Imports

devtools,R.utils,stats,httr,mice,RCurl,tictoc,e1071,caret,mlbench,fastICA,RMySQL,BBmisc,rjson,ggplot2,RWeka,farf

Suggests knitr

RoxygenNote 6.1.1

R topics documented:

`autoRLearn` *Run smartML function for automatic Supervised Machine Learning.*

Description

Run the smartML main function for automatic classifier algorithm selection, and hyper-parameter tuning.

Usage

```
autoRLearn(maxTime, directory, testDirectory, classCol = "class",
  selectedFeats = c(), vRatio = 0.1, preProcessF = "N",
  featuresToPreProcess = c(), nComp = NA, nModels = 3, option = 2,
  featureTypes = c(), interp = 0, missingVal = c("NA", "?", " "),
  missingOpr = 0)
```

Arguments

`maxTime`	Float of the maximum time budget for reading dataset, preprocessing, calculating meta-features, Algorithm Selection & hyper-parameter tuning process only in minutes(Excluding Model Interpretability) - This is applicable in case of Option = 2 only.
`directory`	String of the training dataset directory (SmartML accepts file formats arff/(csv with columns headers)).
`testDirectory`	String of the testing dataset directory (SmartML accepts file formats arff/(csv with columns headers)).
`classCol`	String of the name of the class label column in the dataset (default = 'class').
`selectedFeats`	Vector of numbers of features columns to include from the training set and ignore the rest of columns - In case of empty vector, this means to include all features in the dataset file (default = c()).
`vRatio`	Float of the validation set ratio that should be splitted out of the training set for the evaluation process (default = 0.1 –> 10%).
`preProcessF`	string containing the name of the preprocessing algorithm (default = 'N' –> no preprocessing):

- "boxcox" - apply a Box–Cox transform and values must be non-zero and positive in all features,
- "yeo-Johnson" - apply a Yeo-Johnson transform, like a BoxCox, but values can be negative,
- "zv" - remove attributes with a zero variance (all the same value),
- "center" - subtract mean from values,
- "scale" - divide values by standard deviation,
- "standardize" - perform both centering and scaling,
- "normalize" - normalize values,
- "pca" - transform data to the principal components,
- "ica" - transform data to the independent components.

`featuresToPreProcess`	Vector of number of features to perform the feature preprocessing on - In case of empty vector, this means to include all features in the dataset file (default = c()) - This vector should be a subset of `selectedFeats`.
`nComp`	Integer of Number of components needed if either "pca" or "ica" feature preprocessors are needed.
`nModels`	Integer representing the number of classifier algorithms that you want to select based on Meta-Learning and start to tune using Bayesian Optimization (default = 3).
`option`	Integer representing either Classifier Algorithm Selection is needed only = 1 or Algorithm selection with its parameter tuning is required = 2 which is the default value.
`featureTypes`	Vector of either 'numerical' or 'categorical' representing the types of features in the dataset (default = c() –> any factor or character features will be considered as categorical otherwise numerical).
`interp`	Boolean representing if model interpretability (Feature Importance and Interaction) is needed or not (default = 0) This option will take more time budget if set to 1.
`missingVal`	Vector of strings representing the missing values in dataset (default: c('NA', '?', ' ')).
`missingOpr`	Boolean variable represents either delete instances with missing values or apply imputation using "MICE" library which helps you imputing missing values with plausible data values that are drawn from a distribution specifically designed for each missing datapoint- (default = 0 –> delete instances).

Value

List of Results

- "option=1" - Choosen Classifier Algorithms Names `clfs` with their parameters configurations `params` in case of `option=2`,
- "option=2" - Best classifier algorithm name found `clfs` with its parameters configuration `params`, model variable `model`, performance on TestingSet `perf`, and Feature Importance `interpret$featImp` / Interaction `interpret$Interact` plots in case of interpretability `interp` is needed.

Examples

```
## Not run:
autoRLearn(1, 'sampleDatasets/car/train.arff', \
'sampleDatasets/car/test.arff', option = 2, preProcessF = 'normalize')

result <- autoRLearn(10, 'sampleDatasets/shuttle/train.arff', 'sampleDatasets/shuttle/test.arff')

## End(Not run)
```

`datasetReader` *Read Dataset File into Memory.*

Description

Read the file of the training and testing dataset, and perform preprocessing and data cleaning if necessary.

Usage

```
datasetReader(directory, testDirectory, selectedFeats = c(),
  classCol = "class", preProcessF = "N", featuresToPreProcess = c(),
  nComp = NA, missingVal = c("NA", "?", " "), missingOpr = 0)
```

Arguments

`directory`	String of the directory to the file containing the training dataset.
`testDirectory`	String of the directory to the file containing the testing dataset.
`selectedFeats`	Vector of numbers of features columns to include from the training set and ignore the rest of columns - In case of empty vector, this means to include all features in the dataset file (default = c()).
`classCol`	String of the name of the class label column in the dataset (default = 'class').
`preProcessF`	string containing the name of the preprocessing algorithm (default = 'N' –> no preprocessing): • "boxcox" - apply a Box–Cox transform and values must be non-zero and positive in all features, • "yeo-Johnson" - apply a Yeo-Johnson transform, like a BoxCox, but values can be negative, • "zv" - remove attributes with a zero variance (all the same value), • "center" - subtract mean from values, • "scale" - divide values by standard deviation, • "standardize" - perform both centering and scaling, • "normalize" - normalize values, • "pca" - transform data to the principal components, • "ica" - transform data to the independent components.
`featuresToPreProcess`	Vector of number of features to perform the feature preprocessing on - In case of empty vector, this means to include all features in the dataset file (default = c()) - This vector should be a subset of `selectedFeats`.
`nComp`	Integer of Number of components needed if either "pca" or "ica" feature preprocessors are needed.
`missingVal`	Vector of strings representing the missing values in dataset (default: c('NA', '?', ' ')).
`missingOpr`	Boolean variable represents either delete instances with missing values or apply imputation using "MICE" library which helps you imputing missing values with plausible data values that are drawn from a distribution specifically designed for each missing datapoint- (default = 0 –> delete instances).

Value

List of the TrainingSet `Train` and TestingSet `Test`.

Examples

```
## Not run:
dataset <- datasetReader('/Datasets/irisTrain.csv', '/Datasets/irisTest.csv')

## End(Not run)
```

`runClassifier` *Fit a classifier model.*

Description

Run the classifier on a training set and measure performance on a validation set.

Usage

```
runClassifier(trainingSet, validationSet, params, classifierAlgorithm,
  interp = 0)
```

Arguments

`trainingSet`	Dataframe of the training set.
`validationSet`	Dataframe of the validation Set.
`params`	A string of parameter configuration values for the current classifier to be tuned (parameters are separated by #) and can be obtained from `params` out of resulted list after running `autoRLearn` function.
`classifierAlgorithm`	String of the name of classifier algorithm used now. • "svm" - Support Vector Machines from e1071 package, • "naiveBayes" - naiveBayes from e1071 package, • "randomForest" - randomForest from randomForest package, • "lmt" - LMT Weka classifier trees from RWeka package, • "lda" - Linear Discriminant Analysis from MASS package, • "j48" - J48 Weka classifier Trees from RWeka package, • "bagging" - Bagging Classfier from ipred package, • "knn" - K nearest Neighbors from FNN package, • "nnet" - Simple neural net from nnet package, • "fda" - Flexible discriminant Analysis from MDA package, • "C50" - C50 decision tree from C5.0 pacakge, • "rpart" - rpart decision tree from rpart package, • "rda" - regularized discriminant analysis from klaR package, • "plsda" - Partial Least Squares And Sparse Partial Least Squares Discriminant Analysis from caret package, • "glm" - Fitting Generalized Linear Models from stats package, • "deepboost" - deep boost classifier from deepboost package.
`interp`	Boolean representing if interpretability is required or not (Default = 0).

Value

List of performance on validationSet named `perf`, model fitted on trainingSet named `trainingSet`, and interpretability plots named `interact` in case of interp = 1

Examples

```
## Not run:
result1 <- autoRLearn(10, 'sampleDatasets/shuttle/train.arff', 'sampleDatasets/shuttle/test.arff')
dataset <- datasetReader('/Datasets/irisTrain.csv', '/Datasets/irisTest.csv')
result2 <- runClassifier(dataset$Train, dataset$Test, result1$params, result1$clfs)

## End(Not run)
```

Index