Package ‘swag’

Type Package
Title Sparse Wrapper Algorithm
Version 0.1.0
Maintainer Samuel Orso <Samuel.Orso@unige.ch>
Description An algorithm that trains a meta-learning procedure that combines screening and wrapper methods to find a set of extremely low-dimensional attribute combinations. This package works on top of the 'caret' package and proceeds in a forward-step manner. More specifically, it builds and tests learners starting from very few attributes until it includes a maximal number of attributes by increasing the number of attributes at each step. Hence, for each fixed number of attributes, the algorithm tests various (randomly selected) learners and picks those with the best performance in terms of training error. Throughout, the algorithm uses the information coming from the best learners at the previous step to build and test learners in the following step. In the end, it outputs a set of strong low-dimensional learners.
License GPL (>= 2)
Encoding UTF-8
LazyData false
Depends R (>= 4.0.0)
Imports caret, Rdpack (>= 0.7), stats
Suggests doParallel, e1071, foreach, ggplot2, glmnet, grDevices, iterators, kernlab, knitr, lattice, methods, mlbench, ModelMetrics, nlme, parallel, plyr, pROC, randomForest, recipes, remotes, reshape2, stats4, rmarkdown, utils, withr
RdMacros Rdpack
RoxygenNote 7.1.1
VignetteBuilder knitr
URL https://github.com/SMAC-Group/SWAG-R-Package/
BugReports https://github.com/SMAC-Group/SWAG-R-Package/issues/
NeedsCompilation no
predict.swag

Author  Samuel Orso [aut, cre],
        Gaetan Bakalli [aut],
        Cesare Miglioli [aut],
        Stephane Guerrier [ctb],
        Roberto Molinari [ctb]

Repository  CRAN
Date/Publication  2020-11-10 15:30:02 UTC

R topics documented:

- `predict.swag` .................................................. 2
- `swag` .................................................................. 3
- `swagControl` ...................................................... 4

Index  6

-----------------------------
predict.swag  Predict method for SWAG
-----------------------------

Description

Gives predictions for different `train` learners obtained by `swag`.

Usage

```r
## S3 method for class 'swag'
predict(
  object,
  newdata = NULL,
  type = c("best", "cv_performance", "attribute"),
  cv_performance = NULL,
  attribute = NULL,
  ...
)
```

Arguments

- `object`  An object of class `swag`.
- `newdata`  an optional set of data to predict on. If `NULL` the original training data are used.
- `type`  type of prediction required. The default is "best", it takes the best model (with lowest CV errors). The option "cv_performance" (which requires `cv_performance`) allows to set a level of CV errors under which models are predicted. The option "attribute" (which requires `attribute`) allows to specify an attribute at which models are predicted.
- `cv_performance`  a level of CV errors (between 0 and 1) combines with type "cv_performance".
- `attribute`  an attribute combines with type "attribute".
- `...`  Not used for the moment.
Details
Currently the different train learners are trained (again) to make the predictions.

Value
Predictions.

Author(s)
Gaetan Bakalli, Samuel Orso and Cesare Miglioli

---

**swag**

*Spare Wrapper AlGorithm (swag)*

---

Description
swag is used to trains a meta-learning procedure that combines screening and wrapper methods to find a set of extremely low-dimensional attribute combinations. swag works on top of the caret package and proceeds in a forward-step manner.

Usage

```r
swag(
  x,
  y,
  control = swagControl(),
  auto_control = TRUE,
  caret_args_dyn = NULL,
  ...
)
```

Arguments

- **x**: A matrix or data.frame of attributes
- **y**: A vector of binary response variable.
- **control**: see swagControl
- **auto_control**: A boolean, whether some control parameters are adjusted depending on x and y (see swagControl).
- **caret_args_dyn**: If not null, a function that can modify arguments for train dynamically (see the details).
- **...**: Arguments to be passed to train functions (see the details).
Details

Currently we expect the user to replace ... with the arguments one would use for \texttt{train}. This requires to know how to use \texttt{train} function. If ... is left unspecified, default values of \texttt{train} are used. But this might lead to unexpected results.

The function \texttt{caret\_args\_dyn} is expected to take as a first argument a \texttt{list} with all arguments for \texttt{train} and as a second argument the number of attributes (see examples in the vignette).

More specifically, \texttt{swag} builds and tests learners starting from very few attributes until it includes a maximal number of attributes by increasing the number of attributes at each step. Hence, for each fixed number of attributes, the algorithm tests various (randomly selected) learners and picks those with the best performance in terms of training error. Throughout, the algorithm uses the information coming from the best learners at the previous step to build and test learners in the following step. In the end, it outputs a set of strong low-dimensional learners. See Molinari et al. (2020) for more details.

Value

\texttt{swag} returns an object of class "swag". It is a \texttt{list} with the following components:

\begin{verbatim}
x same as x input
y same as y input
control the control used (see \texttt{swagControl})
CVs a list containing cross-validation errors from all trained models
VarMat a list containing information about which models are trained
cv_alpha a vector of size pmax containing the cross-validation error at alpha (see \texttt{swagControl})
IDs a list containing information about trained model that performs better than corresponding cv_alpha error
args\_caret arguments used for \texttt{train}
args\_caret\_dyn same as args\_caret\_dyn input
\end{verbatim}

Author(s)

Gaetan Bakalli, Samuel Orso and Cesare Miglioli

References

Usage

swagControl(pmax = 3, m = 100, alpha = 0.05, seed = 163L, verbose = FALSE)

Arguments

- **pmax**: A integer representing the maximum number of attributes per learner.
- **m**: A integer representing the maximum number of learners per dimension explored.
- **alpha**: A double representing the proportion of screening.
- **seed**: An integer that controls the reproducibility.
- **verbose**: A boolean for printing current progress of the algorithm.

See Also

swag
Index

predict.swag, 2
swag, 2, 3, 5
swagControl, 3, 4, 4
train, 2–4