Package: EnsembleCV (via r-universe)

October 4, 2024

Type Package

Title Extensible Package for Cross-Validation-Based Integration of Base Learners

Version 0.8

Date 2016-09-13

Author Mansour T.A. Sharabiani, Alireza S. Mahani

Maintainer Alireza S. Mahani <alireza.s.mahani@gmail.com>

Description Extends the base classes and methods of EnsembleBase package for cross-validation-based integration of base learners. Default implementation calculates average of repeated CV errors, and selects the base learner / configuration with minimum average error. The package takes advantage of the file method provided in EnsembleBase package for writing estimation objects to disk in order to circumvent RAM bottleneck. Special save and load methods are provided to allow estimation objects to be saved to permanent files on disk, and to be loaded again into temporary files in a later R session. The package can be extended, e.g. by adding variants of the current implementation.

License GPL (>= 2)

Depends EnsembleBase

Imports parallel,methods

NeedsCompilation no

Repository CRAN

Date/Publication 2016-09-13 22:20:51

Contents

ecv.regression 2
ecv.regression.baselearner.control 4
ecv.save 5
plot.ecv.regression 6

Regression.Select.MinAvgErr.Config-class 7
Regression.Select.MinAvgErr.FitObj-class 8

Index **9**

`ecv.regression` *Cross-Validation-Based Integration of Regression Base Learners for Ensemble Learning*

Description

This function uses repeated cross-validation to find the base learner configuration with smallest error. It then trains and returns the chosen model (base learner and configuration), trained on the full data set.

Usage

```
ecv.regression(formula, data
  , baselearner.control = ecv.regression.baselearner.control()
  , integrator.control = ecv.regression.integrator.control()
  , ncores = 1, filemethod = FALSE, print.level = 1
  , preschedule = TRUE
  , schedule.method = c("random", "as.is", "task.length")
  , task.length
)
```

Arguments

`formula`	Formula expressing response variable and covariates.
`data`	Data frame containing the response variable and covariates.
`baselearner.control`	Control structure determining the base learners, their configurations, and data partitioning details. See `ecv.regression.baselearner.control`.
`integrator.control`	Control structure governing integrator behavior. See `ecv.regression.integrator.control`.
`ncores`	Number of cores used for parallel training of base learners.
`filemethod`	Boolean flag indicating whether or not to save estimation objects to disk or not. Using `filemethod=T` reduces RAM pressure.
`print.level`	Controlling verbosity level.
`preschedule`	Boolean flag, indicating whether base learner training jobs must be scheduled statically (`TRUE`) or dynamically (`FALSE`).
`schedule.method`	Method used for scheduling tasks on threads. In "as.is" tasks are assigned to threads in a round-robin fashion for static scheduling. In dynamic scheduling, tasks form a queue without any re-ordering. In "random", tasks are first randomly shuffled, and the rest is similar to "as.is". In "task.length", a heuristic

algorithm is used in static scheduling for assigning tasks to threads to minimize load imbalance, i.e. make total task lengths in threads roughly equal. In dynamic scheduling, tasks are sorted in descending order of expected length to form the task queue.

`task.length` Vector of estimated task lengths, to be used in the "task.length" method of scheduling.

Value

An object of classes `ecv.regression` (if `filemethod==TRUE`, also has class of `ecv.file`), a list with the following elements:

`call` Copy of function call.

`formula` Copy of formula argument in function call.

`instance.list` An object of class `Instance.List`, containing all permutations of base learner configurations and random data partitions generated in the body of the function.

`integrator.config` Copy of configuration object passed to the integrator. Object of class `Regression.Select.MinAvgErr.Co`

`method` Integration method. Currently, only "default" is supported.

`est` A list with these elements: 1) `baselearner.cv.batch`, an object of class `Regression.CV.Batch.FitObj` containing the fit object from CV batch training of base learners; 2) `baselearner.batch`, an object of class `Regression.Batch.FitObj` containing the fit object from batch training of base learners on entire `data`; 3) `integrator`, an object of class `Regression.Select.MinAvgErr.FitObj` containing the fit object returned by the integrator.

`y` Copy of response variable vector.

`pred` Within-sample prediction of the ensemble model.

`filemethod` Copy of passed-in `filemethod` argument.

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

See Also

`ecv.regression.baselearner.control`, `ecv.regression.integrator.control`, `Instance.List`, `Regression.Select.MinAvgErr.Config`, `Regression.CV.Batch.FitObj`, `Regression.Batch.FitObj`, `Regression.Select.MinAvgErr.FitObj`

Examples

```
data(servo)
myformula <- class~motor+screw+pgain+vgain
perc.train <- 0.7
index.train <- sample(1:nrow(servo), size = round(perc.train*nrow(servo)))
data.train <- servo[index.train,]
data.predict <- servo[-index.train,]
## to run longer test using all 5 default regression base learners
```

```
## try: est <- ecv.regression(myformula, data.train, ncores=2)
est <- ecv.regression(myformula, data.train, ncores=2
  , baselearner.control =
      ecv.regression.baselearner.control(baselearners = c("knn")))
newpred <- predict(est, data.predict)
```

`ecv.regression.baselearner.control`
Utility Functions for Configuring Regression Base Learners and Integrator in **EnsembleCV** *Package*

Description

Function `ecv.regression.baselearner.control` sets up the base learners used in the `ecv.regression` call.

Usage

```
ecv.regression.baselearner.control(
  baselearners = c("nnet", "rf", "svm", "gbm", "knn", "penreg")
  , baselearner.configs = make.configs(baselearners, type = "regression")
  , npart = 1, nfold = 5
)
ecv.regression.integrator.control(errfun=rmse.error, method=c("default"))
```

Arguments

`baselearners`	Names of base learners used. Currently, regression options available are Neural Network ("nnet"), Random Forest ("rf"), Support Vector Machine ("svm"), Gradient Boosting Machine ("gbm"), and K-Nearest Neighbors ("knn"), Penalized Rergession ("penreg") and Bayesian Additive Regression Trees ("bart"). The last learner is not included by default, due to significantly longer training time needed by it ("bart") compared to other learners.
`baselearner.configs`	List of base learner configurations. Default is to call `make.configs` from package **EnsembleBase**.
`npart`	Number of partitions to train each base learner configuration in a CV scheme.
`nfold`	Number of folds within each data partition.
`errfun`	Error function used to compare performance of base learner configurations. Default is to use `rmse.error` from package **EnsembleBase**.
`method`	Integrator method. Currently, only option is "default", which uses average error for each base learner configuration across repeated CV runs to chose the best configuration.

Value

Both functions return lists with same element names as function arguments.

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

See Also

make.configs, rmse.error

---

ecv.save *Custom Functions for Disk I/O in* **EnsembleCV** *Package*

Description

These functions can be used whether filemethod flag is set to TRUE or FALSE during the epcreg call. Note that ecv.load 'returns' the estimation object (in contrast to the standard load method).

Usage

```
ecv.save(obj, file)
ecv.load(file)
```

Arguments

obj	Object of classes "ecv.regression" and "ecv.file", usually the output of call to function ecv.regression with filemethod flag set to TRUE.
file	Filepath to where obj must be saved to / loaded from.

Value

Function ecv.load returns the saved obj, with estimation files automatically copied to R temporary directory, and filepaths inside the obj fields updated to point to these new filepaths.

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

See Also

ecv.regression

Examples

```
## Not run:
data(servo)
myformula <- class~motor+screw+pgain+vgain
perc.train <- 0.7
index.train <- sample(1:nrow(servo), size = round(perc.train*nrow(servo)))
data.train <- servo[index.train,]
data.predict <- servo[-index.train,]
```

```
est <- ecv.regression(myformula, data.train, ncores=2, filemethod=TRUE
  , baselearner.control=ecv.regression.baselearner.control(baselearners="knn"))
ecv.save(est, "somefile")
rm(est) # alternatively, exit and re-launch R session
est.loaded <- ecv.load("somefile")
newpred <- predict(est.loaded, data.predict)

# can also be used with filemethod set to FALSE
est <- ecv.regression(myformula, data.train, ncores=2, filemethod=FALSE
  , baselearner.control=ecv.regression.baselearner.control(baselearners="knn"))
ecv.save(est, "somefile")
rm(est) # alternatively, exit and re-launch R session
est.loaded <- ecv.load("somefile")
newpred <- predict(est.loaded, data.predict)

## End(Not run)
```

`plot.ecv.regression` *S3 Methods for class* `"ecv.regression"`

Description

Functions for prediction and plotting of `ecv.regression` objects.

Usage

```
## S3 method for class 'ecv.regression'
predict(object, newdata=NULL, ncores=1, ...)
## S3 method for class 'ecv.regression'
plot(x, ...)
```

Arguments

`object`	Object of class `"ecv.regression"`, typically the output of function `ecv.regression`.
`newdata`	New data frame to make predictions for. If `NULL`, prediction is made for training data.
`ncores`	Number of cores to use for parallel prediction.
`x`	Object of class `"ecv.regression"`, typically the output of function `ecv.regression`.
`...`	Arguments passed to/from other methods.

Value

Function `plot.ecv.regression` creates a plot of base learner CV errors, with one data point per base learner configuration. The horizontal dotted line indicates the CV error corresponding to the chosen base learner configuration. For "default" method, this is the same as the minimum error of points on this plot. Function `predict.ecv.regression` returns a vector of length `nrow(newdata)` (or of length of training data if `newdata==NULL`.)

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

`Regression.Select.MinAvgErr.Config-class`
Class `"Regression.Select.MinAvgErr.Config"`

Description

Configuration class for the "MinAvgErr" specialization of the "Regression.Select.Fit" operation in **EnsembleBase** package. This operation selects the base learner configuration with minimum average error across repeated cross-validation runs.

Objects from the Class

Objects can be created by calls of the form `new("Regression.Select.MinAvgErr.Config", ...)`.

Slots

`instance.list`**:** Object of class `Instance.List`, containing a list of base learners to train.

`errfun`**:** Object of class `"function"`, the error metric to use for ranking base learner performances.

Extends

Class `"Regression.Select.Config"`, directly.

Methods

Regression.Select.Fit `signature(object = "Regression.Select.MinAvgErr.Config")`: ...

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

Regression.Select.MinAvgErr.FitObj-class
Class "Regression.Select.MinAvgErr.FitObj"

Description

Class containing the fit object from the "MinAvgErr" specialization of the "Regression.Select.Fit" operation in **EnsembleBase** package.

Objects from the Class

Objects can be created by calls of the form `new("Regression.Select.MinAvgErr.FitObj", ...)`.

Slots

`config`: Object of class "Regression.Select.Config", containing the configuration supplied to the fit operation.

`est`: Object of class "ANY", containing the estimation object needed for prediction. This is a list with elements `config.opt` (optimal base learner configuration), `error.opt` (error associated with optimal configuration), and `errors` (vector of errors for all base learner configurations).

`pred`: Object of class "RegressionSelectPred", containing the within-sample prediction, in this case the average prediction across all partitions. Note that this prediction is not used in the `ecv.regression` function as the ultimate training-set prediction. Instead, base learners trained on full training set (not CV style) are used for that purpose.

Extends

Class "Regression.Select.FitObj", directly.

Author(s)

Mansour T.A. Sharabiani, Alireza S. Mahani

Index

∗ **classes**
    Regression.Select.MinAvgErr.Config-class, 7
    Regression.Select.MinAvgErr.FitObj-class, 8

ecv.load *(*ecv.save*)*, 5
ecv.regression, 2, *4–6*, *8*
ecv.regression.baselearner.control, *2*, *3*, 4
ecv.regression.integrator.control, *2*, *3*
ecv.regression.integrator.control *(*ecv.regression.baselearner.control*)*, 4
ecv.save, 5

Instance.List, *3*, *7*

make.configs, *4*, *5*

plot.ecv.regression, 6
predict.ecv.regression *(*plot.ecv.regression*)*, 6

Regression.Batch.FitObj, *3*
Regression.CV.Batch.FitObj, *3*
Regression.Select.Config, *7*, *8*
Regression.Select.FitObj, *8*
Regression.Select.MinAvgErr.Config, *3*
Regression.Select.MinAvgErr.Config-class, 7
Regression.Select.MinAvgErr.FitObj, *3*
Regression.Select.MinAvgErr.FitObj-class, 8
RegressionSelectPred, *8*
rmse.error, *4*, *5*