Package: EBMAforecast (via r-universe)

October 16, 2024

Type Package

Title Estimate Ensemble Bayesian Model Averaging Forecasts using Gibbs Sampling or EM-Algorithms

Version 1.0.32

Date 2024-03-19

URL https://github.com/fhollenbach/EBMA/

Description Create forecasts from multiple predictions using ensemble Bayesian model averaging (EBMA). EBMA models can be estimated using an expectation maximization (EM) algorithm or as fully Bayesian models via Gibbs sampling. The methods in this package are Montgomery, Hollenbach, and Ward (2015) <doi:10.1016/j.ijforecast.2014.08.001> and Montgomery, Hollenbach, and Ward (2012) <doi:10.1093/pan/mps002>.

License GPL (>= 2)

Imports Rcpp (>= 1.0.2), plyr, graphics, separationplot, Hmisc, abind, gtools, methods, glue

LinkingTo Rcpp

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Collate 'EBMAforecast-package.R' 'forecastData.R' 'EBMApredict.R' 'RcppExports.R' 'calibrateEnsemble.R' 'compareModels.R' 'document-data.R' 'fitEnsembleLogit.R' 'fitEnsembleNormal.R' 'global.R' 'makeForecastData.R' 'predictLogit.R' 'predictNormal.R' 'summary_plot.R' 'print_show.R' 'utility-functions.R'

Repository https://fhollenbach.r-universe.dev

RemoteUrl https://github.com/fhollenbach/ebma

RemoteRef HEAD

RemoteSha 04118be6f8074258ed44be04e50e2b9a3c2eaa18

Contents

calibrateEnsemble 2
calibrationSample 6
CompareModels-class 7
EBMApredict 9
ForecastData-class 10
makeForecastData 11
presidentialForecast 13
print,ForecastData-method 14
setPredCalibration<- 15
SummaryForecastData-class 17

Index **21**

`calibrateEnsemble` *Calibrate an ensemble Bayesian Model Averaging model*

Description

This function calibrates an EBMA model based on out-of-sample performance in the calibration period. Given a dependent variable and calibration-sample predictions from multiple component forecast models in the `ForecastData`, the `calibrateEnsemble` function fits an ensemble BMA mixture model. The weights assigned to each model are derived from the individual model's performance in the calibration period. Missing observations are allowed in the calibration period, however models with missing observations are penalized. When missing observations are prevalent in the calibration set, the EM algorithm or gibbs sampler is adjusted and model paprameters are estimated by maximizing a renormalized partial expected complete-data log-likelihood (Fraley et al. 2010).

Usage

```
calibrateEnsemble(
  .forecastData = new("ForecastData"),
  exp = 1,
  tol = sqrt(.Machine$double.eps),
  maxIter = 1e+06,
  model = "logit",
  method = "EM",
  predType = "posteriorMedian",
  useModelParams = TRUE,
 W = rep(1/dim(.forecastData@predCalibration)[2], dim(.forecastData@predCalibration)[2]),
  const = 0,
  modelPriors = rep(1, dim(.forecastData@predCalibration)[2]),
  iterations = 40000,
  burns = 20000,
  thinning = 20,
  ...
```

```
)

## S4 method for signature 'ForecastData'
calibrateEnsemble(
  .forecastData = new("ForecastData"),
  exp = 1,
  tol = sqrt(.Machine$double.eps),
  maxIter = 1e+06,
  model = "logit",
  method = "EM",
  predType = "posteriorMean",
  useModelParams = TRUE,
 W = rep(1/dim(.forecastData@predCalibration)[2], dim(.forecastData@predCalibration)[2]),
  const = 0,
  modelPriors = rep(1, dim(.forecastData@predCalibration)[2]),
  iterations = 40000,
  burns = 20000,
  thinning = 20,
  ...
)
```

Arguments

`.forecastData`	An object of class 'ForecastData' that will be used to calibrate the model.
`exp`	The exponential shrinkage term. Forecasts are raised to the (1/exp) power on the logit scale for the purposes of bias reduction. The default value is `exp=3`.
`tol`	Tolerance for improvements in the log-likelihood before the EM algorithm will stop optimization. The default is `tol= 0.01`, which is somewhat high. Researchers may wish to reduce this by an order of magnitude for final model estimation.
`maxIter`	The maximum number of iterations the EM algorithm will run before stopping automatically. The default is `maxIter=10000`.
`model`	The model type that should be used given the type of data that is being predicted (i.e., normal, binary, etc.).
`method`	The estimation method used. It takes either an `EM` or `gibbs` as an argument.
`predType`	The prediction type used for the gibbs sampling EBMA model, user can choose either `posteriorMedian` or `posteriorMean` (default). Model performance statistics are based on the posterior median or mean forecast. Note that the posterior median forecast is not equal to the forecast based on the median posterior weight. EM predictions based on mean.
`useModelParams`	If "TRUE" individual model predictions are transformed based on logit models. If "FALSE" all models' parameters will be set to 0 and 1.
`W`	A vector or matrix of initial model weights. If unspecified, each model will receive weight equal to 1/number of Models
`const`	User provided "wisdom of crowds" parameter, serves as minimum model weight for all models. Default = 0. Only used in model estimated using EM.

`modelPriors`	User provided vector of Dirichlet prior for each of the models. Only used in normal model estimated with gibbs sampling. Default prior is 1 for each model.
`iterations`	The number of iterations for the Bayesian model. Default = 40000.
`burns`	The burn in for the Gibbs sampler. Default = 20000.
`thinning`	How much the Gibbs sampler is thinned. Default = 20.
`...`	Not implemented

Value

Returns a data of class 'FDatFitLogit' or FDatFitNormal, a subclass of 'ForecastData', with the following slots

`predCalibration`	A matrix containing the predictions of all component models and the EBMA model for all observations in the calibration period. Under gibbs sampling, the EBMA prediction is either the median or mean of the posterior predictive distribution, depending on the `predType` setting.
`predTest`	A matrix containing the predictions of all component models and the EBMA model for all observations in the test period. Under gibbs sampling, the EBMA prediction is either the median or mean of the posterior predictive distribution, depending on the `predType` setting.
`outcomeCalibration`	A vector containing the true values of the dependent variable for all observations in the calibration period.
`outcomeTest`	An optional vector containing the true values of the dependent variable for all observations in the test period.
`modelNames`	A character vector containing the names of all component models. If no model names are specified, names will be assigned automatically.
`modelWeights`	A vector containing model weights assigned to each model. When the gibbs sampler is used, this slot contains either the median or mean of the posterior weights, depending on the `predType` setting.
`modelParams`	The parameters for the individual logit models that transform the component models.
`useModelParams`	Indicator whether model parameters for transformation were estimated or not.
`logLik`	The final log-likelihood for the calibrated EBMA model. Empty for estimations using the gibbs sampler.
`exp`	The exponential shrinkage term.
`tol`	Tolerance for improvements in the log-likelihood before the EM algorithm will stop optimization.
`maxIter`	The maximum number of iterations the EM algorithm will run before stopping automatically.
`method`	The estimation method used.
`iter`	Number of iterations run in the EM algorithm. Empty for estimations using the gibbs sampler.

`call` The actual call used to create the object.

`posteriorWeights`
A matrix of the full posterior model weights from model calibration. Rows are the observations in the calibration period, columns are the saved iterations of the gibbs sampler. Empty for EM estimations.

`posteriorPredCalibration`
A matrix of the posterior predictive distribution for observations in the calibration period, based on the full posterior of model weights. Empty for EM estimations.

`posteriorPredTest`
A matrix of the posterior predictive distribution for observations in the test period, based on the full posterior of model weights. Empty for EM estimations.

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Raftery, A. E., T. Gneiting, F. Balabdaoui and M. Polakowski. (2005). Using Bayesian Model Averaging to calibrate forecast ensembles. *Monthly Weather Review*. **133**:1155–1174.

Sloughter, J. M., A. E. Raftery, T. Gneiting and C. Fraley. (2007). Probabilistic quantitative precipitation forecasting using Bayesian model averaging. *Monthly Weather Review*. **135**:3209–3220.

Fraley, C., A. E. Raftery, T. Gneiting. (2010). Calibrating Multi-Model Forecast Ensembles with Exchangeable and Missing Members using Bayesian Model Averaging. *Monthly Weather Review*. **138**:190–202.

Sloughter, J. M., T. Gneiting and A. E. Raftery. (2010). Probabilistic wind speed forecasting using ensembles and Bayesian model averaging. *Journal of the American Statistical Association*. **105**:25–35.

Fraley, C., A. E. Raftery, and T. Gneiting. (2010). Calibrating multimodel forecast ensembles with exchangeable and missing members using Bayesian model averaging. *Monthly Weather Review*. **138**:190–202.

Examples

```
## Not run:
data(calibrationSample)

data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))
initW <- rep(1/3,3)
```

```
this.ensemble.em <- calibrateEnsemble(this.ForecastData, model="logit", tol=0.001)

this.ensemble.gibbs <- calibrateEnsemble(this.ForecastData, model="logit", method = "gibbs")

## End(Not run)
```

`calibrationSample` *Sample data Insurgency Predictions*

Description

This includes the data for the predictions of insurgencies in 29 countries for 2010.

Usage

```
calibrationSample

testSample
```

Format

An object of class `matrix` (inherits from `array`) with 696 rows and 4 columns.

An object of class `matrix` (inherits from `array`) with 348 rows and 4 columns.

Details

The predictions included in the dataset are:

- `LMER` Predictions from a generalized linear mixed effects model using a logistic link function and including a randomeffects term for lagged GDP per capita and the lagged number of conflictual events involving the United States in the country of interest.
- `SAE` Predictions from a one model developed as part of the ICEWS project and was designed by Strategic Analysis Enterprises.
- `GLM` Predictions from a crude logistic model that includes only population size, GDP growth (both lagged 3 months), the number of minority groups at risk in the country, and a measure of anocracy supplied in the Polity IV data set.

More detail about each model can be found in Mongomery et al. (2012)

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Examples

```
## Not run:
data(calibrationSample)
data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))
initW <- rep(1/3,3)

this.ensemble.em <- calibrateEnsemble(this.ForecastData, model="logit", tol=0.001)

this.ensemble.gibbs <- calibrateEnsemble(this.ForecastData, model="logit", method = "gibbs")

## End(Not run)
```

`CompareModels-class` *Function for comparing multiple models based on predictive performance*

Description

This function produces statistics to compare the predictive performance of the different models component models, as well as for the EBMA model itself, for either the calibration or the test period. It currently calculates the area under the ROC (`auc`), the `brier` score, the percent of observations predicted correctly (`percCorrect`), as well as the proportional reduction in error compared to some baseline model (`pre`) for binary models. For models with normally distributed outcomes the `CompareModels` function can be used to calculate the root mean squared error (`rmse`) as well as the mean absolute error (`mae`).

Usage

```
compareModels(
  .forecastData,
  .period = "calibration",
  .fitStatistics = c("brier", "auc", "perCorrect", "pre"),
  .threshold = 0.5,
  .baseModel = 0,
  ...
)

## S4 method for signature 'ForecastData'
compareModels(.forecastData, .period, .fitStatistics, .threshold, .baseModel)
```

Arguments

.forecastData	An object of class 'ForecastData'.
.period	Can take value of "calibration" or "test" and indicates the period for which the test statistics should be calculated.
.fitStatistics	A vector naming statistics that should be calculated. Possible values include "auc", "brier", "percCorrect", "pre" for logit models and "mae","rsme" for normal models.
.threshold	The threshold used to calculate when a "positive" prediction is made by the model for binary dependent variables.
.baseModel	Vector containing predictions used to calculate proportional reduction of error ("pre").
...	Not implemented

Value

A data object of the class 'CompareModels' with the following slots:

fitStatistics	The output of the fit statistics for each model.
period	The period, "calibration" or "test", for which the statistics were calculated.
threshold	The threshold used to calculate when a "positive" prediction is made by the model.
baseModel	Vector containing predictions used to calculate proportional reduction of error ("pre").

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

See Also

ensembleBMA, other functions

Examples

```
## Not run: data(calibrationSample)

data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))
```

```
this.ensemble <- calibrateEnsemble(this.ForecastData, model="logit", tol=0.001, exp=3)

compareModels(this.ensemble,"calibration")

compareModels(this.ensemble,"test")

## End(Not run)
```

EBMApredict *EBMApredict*

Description

Function allows users to create new predictions given an already estimated EBMA model This function produces predictions based on EBMA model weights and component model predictions.

Usage

```
EBMApredict(EBMAmodel, Predictions, Outcome = NULL, ...)

## S4 method for signature 'ForecastData'
EBMApredict(EBMAmodel, Predictions, Outcome = NULL, ...)
```

Arguments

EBMAmodel	An estimated EBMA model object
Predictions	A matrix with a column for each component model's predictions.
Outcome	An optional vector containing the true values of the dependent variable for all observations in the test period.
...	Not implemented

Value

Returns a data of class 'FDatFitLogit' or FDatFitNormal, a subclass of 'ForecastData', with the following slots:

predTest	A matrix containing the predictions of all component models and the EBMA model for all observations in the test period.
period	The period, "calibration" or "test", for which the statistics were calculated.
outcomeTest	An optional vector containing the true values of the dependent variable for all observations in the test period.
modelNames	A character vector containing the names of all component models. If no model names are specified, names will be assigned automatically.
modelWeights	A vector containing the model weights assigned to each model.

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2015). Calibrating ensemble forecasting models with sparse data in the social sciences. *International Journal of Forecasting*. 31(3): 930-942.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

`ForecastData-class` *An ensemble forecasting data object*

Description

Objects of class `ForecastData` are used in the `calibrateEnsemble` function. Datasets should be converted into an object of class `ForecastData` using the `makeForecastData` function. Individual slots of the `ForecastData` object can be accessed and changed using the `get` and `set` functions respectively. Missing observations in the prediction calibration set are allowed.

Details

@slot predCalibration An array containing the predictions of all component models for the observations in the calibration period. @slot predTest An array containing the predictions of all component models for all the observations in the test period. @slot outcomeCalibration A vector containing the true values of the dependent variable for all observations in the calibration period. @slot outcomeTest A vector containing the true values of the dependent variable for all observations in the test period. @slot modelNames A character vector containing the names of all component models.

Examples

```
## Not run:
data(calibrationSample)
data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))

### to acces individual slots in the ForecastData object
getPredCalibration(this.ForecastData)
getOutcomeCalibration(this.ForecastData)
getPredTest(this.ForecastData)
getOutcomeTest(this.ForecastData)
getModelNames(this.ForecastData)
```

```
### to assign individual slots, use set functions

setPredCalibration(this.ForecastData)<-calibrationSample[,c("LMER", "SAE", "GLM")]
setOutcomeCalibration(this.ForecastData)<-calibrationSample[,"Insurgency"]
setPredTest(this.ForecastData)<-testSample[,c("LMER", "SAE", "GLM")]
setOutcomeTest(this.ForecastData)<-testSample[,"Insurgency"]
setModelNames(this.ForecastData)<-c("LMER", "SAE", "GLM")

## End(Not run)
```

`makeForecastData` *Build a ensemble forecasting data object*

Description

This function uses the component model forecasts and dependent variable observations provided by the user to create an object of class `ForecastData`, which can then be used to calibrate and fit the ensemble. Individual slots of the `ForecastData` object can be accessed and changed using the `get` and `set` functions respectively. Missing predictions are allowed in the calibration set.

Usage

```
makeForecastData(
  .predCalibration = array(NA, dim = c(0, 0, 0)),
  .predTest = array(NA, dim = c(0, 0, 0)),
  .outcomeCalibration = numeric(),
  .outcomeTest = numeric(),
  .modelNames = character(),
  ...
)

## S4 method for signature 'ANY'
makeForecastData(
  .predCalibration,
  .predTest,
  .outcomeCalibration,
  .outcomeTest,
  .modelNames
)
```

Arguments

`.predCalibration`
: A matrix with the number of rows being the number of observations in the calibration period and a column with calibration period predictions for each model.

`.predTest` A vector with the number of rows being the number of observations in the test period and a column with test period predictions for each model.

`.outcomeCalibration` A vector with the true values of the dependent variable for each observation in the calibration period.

`.outcomeTest` A vector with the true values of the dependent variable for each observation in the test period.

`.modelNames` A vector of length p with the names of the component models.

`...` Additional arguments not implemented

Value

A data object of the class 'ForecastData' with the following slots:

`predCalibration` An array containing the predictions of all component models for all observations in the calibration period.

`predTest` An array containing the predictions of all component models for all observations in the test period.

`outcomeCalibration` A vector containing the true values of the dependent variable for all observations in the calibration period.

`outcomeTest` A vector containing the true values of the dependent variable for all observations in the test period.

`modelNames` A character vector containing the names of all component models. If no model names are specified, names will be assigned automatically.

Examples

```
## Not run:
data(calibrationSample)
data(testSample)
this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))

### to acces individual slots in the ForecastData object
getPredCalibration(this.ForecastData)
getOutcomeCalibration(this.ForecastData)
getPredTest(this.ForecastData)
getOutcomeTest(this.ForecastData)
getModelNames(this.ForecastData)

### to assign individual slots, use set functions

setPredCalibration(this.ForecastData)<-calibrationSample[,c("LMER", "SAE", "GLM")]
setOutcomeCalibration(this.ForecastData)<-calibrationSample[,"Insurgency"]
setPredTest(this.ForecastData)<-testSample[,c("LMER", "SAE", "GLM")]
setOutcomeTest(this.ForecastData)<-testSample[,"Insurgency"]
```

```
setModelNames(this.ForecastData)<-c("LMER", "SAE", "GLM")

## End(Not run)
```

`presidentialForecast` *Sample data Presidential Election*

Description

This includes the data for the presidential election forecasting example in Montgomery, Hollenbach and Ward (2012). The data ranges from 1952 to 2008 and includes predictions for the six different component models included in the Ensemble model. Users may split the sample into calibration and test sample.

Usage

```
presidentialForecast
```

Format

An object of class `data.frame` with 15 rows and 7 columns.

Details

The variables included in the dataset are:

- `Campbell` Predictions of Campbell's "Trial-Heat and Economy Model" (Campbell 2008).
- `Abramowitz` Predictions of Abramowitz's "Time for Change Model" (Abramowitz 2008).
- `Hibbs` Predictions for the "Bread and Peace Model" created by Douglas Hibbs (2008).
- `Fair` Forecasts from Fair's presidential vote share model (2010).
- `Lewis-Beck/Tien` Predictions from the "Jobs Model Forecast" by Michael Lewis-Beck and Charles Tien (2008).
- `EWT2C2` Predictions from the model in Column 2 in Table 2 by Erickson and Wlezien (2008).
- `Actual` The true values of the dependent variable, i.e. the incumbent-party voteshare in each presidential election in the sample.

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2015). Calibrating ensemble forecasting models with sparse data in the social sciences. *International Journal of Forecasting*. **31**:930–942.

Campbell, James E. 2008. The trial-heat forecast of the 2008 presidential vote: Performance and value considerations in an open-seat election. *PS: Political Science & Politics* **41**:697-701.

Hibbs, Douglas A. 2000. Bread and peace voting in U.S presidential elections. *Public Choice* **104**:149-180.

Fair, Ray C. 2010. Presidential and Congressional vote-share equations: November 2010 update. Working Paper. Yale University.

Lewis-Beck, Michael S. and Charles Tien. 2008. The job of president and the jobs model forecast: Obama for '08? *PS: Political Science & Politics* **41**:687-690.

Erikson, Robert S. and Christopher Wlezien. 2008. Leading economic indicators, the polls, and the presidential vote. *PS: Political Science & Politics* **41**:703-707.

`print,ForecastData-method`

Print and Show methods for forecast data

Description

Functions to print and show the contents of a data object of the class 'ForecastData' or 'SummaryForecastData'.

Usage

```
## S4 method for signature 'ForecastData'
print(x, digits = 3, ...)

## S4 method for signature 'ForecastData'
show(object)

## S4 method for signature 'SummaryForecastData'
print(x, digits = 3, ...)

## S4 method for signature 'SummaryForecastData'
show(object)

## S4 method for signature 'ForecastData'
print(x, digits = 3, ...)

## S4 method for signature 'ForecastData'
show(object)
```

Arguments

`x`	An object of the class 'ForecastData' or 'SummaryForecastData'.
`digits`	An integer specifying the number of significant digits to print. The default is 3.
`...`	Not implemented
`object`	An object of the class 'ForecastData' or 'SummaryForecastData'.

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2015). Calibrating ensemble forecasting models with sparse data in the social sciences. *International Journal of Forecasting*. **31**: 930-942.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Examples

```
## Not run:  data(calibrationSample)

data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))

this.ensemble <- calibrateEnsemble(this.ForecastData, model="logit", tol=0.001,exp=3)

summary.object <- summary(this.ensemble, period="calibration")
print(summary.object)
show(summary.object)

## End(Not run)
```

---

setPredCalibration<- *"Set" functions*

Description

To assign individual slots, use set functions

Usage

```
setPredCalibration(object) <- value

## S4 replacement method for signature 'ForecastData'
setPredCalibration(object) <- value

setPredTest(object) <- value

## S4 replacement method for signature 'ForecastData'
setPredTest(object) <- value
```

```
setOutcomeCalibration(object) <- value

## S4 replacement method for signature 'ForecastData'
setOutcomeCalibration(object) <- value

setOutcomeTest(object) <- value

## S4 replacement method for signature 'ForecastData'
setOutcomeTest(object) <- value

setModelNames(object) <- value

## S4 replacement method for signature 'ForecastData'
setModelNames(object) <- value
```

Arguments

object	The object to which values are assigned.
value	Values to be assigned.

Value

A data object of the class 'ForecastData' with the following slots:

predCalibration
: An array containing the predictions of all component models for all observations in the calibration period.

predTest
: An array containing the predictions of all component models for all observations in the test period.

outcomeCalibration
: A vector containing the true values of the dependent variable for all observations in the calibration period.

outcomeTest
: A vector containing the true values of the dependent variable for all observations in the test period.

modelNames
: A character vector containing the names of all component models. If no model names are specified, names will be assigned automatically.

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2015). Calibrating ensemble forecasting models with sparse data in the social sciences. *International Journal of Forecasting*. **31**:930–942.#'

Examples

```
## Not run:
data(calibrationSample)
data(testSample)
setPredCalibration(this.ForecastData)<-calibrationSample[,c("LMER", "SAE", "GLM")]
setOutcomeCalibration(this.ForecastData)<-calibrationSample[,"Insurgency"]
setPredTest(this.ForecastData)<-testSample[,c("LMER", "SAE", "GLM")]
setOutcomeTest(this.ForecastData)<-testSample[,"Insurgency"]
setModelNames(this.ForecastData)<-c("LMER", "SAE", "GLM")

## End(Not run)
```

SummaryForecastData-class

Summarize and Plot Ensemble models

Description

These functions summarize and plot Ensemble models that have been fit previously by the user.

Usage

```
## S4 method for signature 'FDatFitLogit'
summary(
  object,
  period = "calibration",
  fitStatistics = c("brier", "auc", "perCorrect", "pre"),
  threshold = 0.5,
  baseModel = 0,
  showCoefs = TRUE,
  ...
)

## S4 method for signature 'FDatFitNormal'
summary(
  object,
  period = "calibration",
  fitStatistics = c("rmse", "mae"),
  threshold = 0.5,
  baseModel = 0,
  showCoefs = TRUE,
  ...
```

```
)

## S4 method for signature 'FDatFitLogit'
plot(
  x,
  period = "calibration",
  subset = 1,
  mainLabel = "",
  xLab = "",
  yLab = "",
  cols = 1,
  ...
)

## S4 method for signature 'FDatFitNormal'
plot(
  x,
  period = "calibration",
  subset = 1,
  mainLabel = paste("Observation", subset),
  xLab = "Outcome",
  yLab = "Posterior Probability",
  cols = 2:(length(x@modelNames) + 1),
  ...
)
```

Arguments

object	An object of the subclass "FDatFitLogit" or "FDatFitNormal".
period	The period for which the summary should be provided or plot produced, either "calibration" or "test". The default is "calibration".
fitStatistics	A vector naming statistics that should be calculated. Possible values for objects in the "FDatFitLogit" subclass include "auc", "brier", "percCorrect", "pre". Possible values for objects in the "FDatFitNormal" subclass include "rmse" and "mae." The default is for all statistics to be calculated. Additional metrics will be made available in a future release of this package.
threshold	The threshold used to calculate when a "positive" prediction is made for a model. The default is 0.5. Not used for objects of the "FDatFitNormal" subclass.
baseModel	A vector containing predictions used to calculate proportional reduction of error ("pre"). The default is 0. Not used for objects of the "FDatFitNormal" subclass.
showCoefs	A logical indicating whether model coefficients from the ensemble should be shown. The default is TRUE.
...	Not implemented
x	An object of class "FDatFitLogit" or "FDatFitNormal"
subset	The row names or numbers for the observations the user wishes to plot. The default is the first row. Only implemented for the subclass "FDatFitNormal".

mainLabel	A vector strings to appear at the top of each predictive posterior plot. Only implemented for the subclass "FDatFitNormal"
xLab	The label for the x-axis. Only implemented for the subclass "FDatFitNormal"
yLab	The label for the y-axis. Only implemented for the subclass "FDatFitNormal"
cols	A vector containing the color for plotting the predictive PDF of each component model forecast. The default is a unique color for each PDF. Only implemented for the subclass "FDatFitNormal"

Value

Either a plot or a data object of the class 'SummaryForecastData'. The data object has the following slots:

summaryData	Under the default, the function produces a matrix containing one row for each model plus one row for the EBMA forecast. The first column is always the model weights assigned to the component models. The second and third columns display the model parameters for the transformation of the component models. The remaining columns are the requested fit statistics for all models, as calculated by the copareModels function. If showCoefs=FALSE, then the model parameters will not be shown.

Author(s)

Michael D. Ward <<michael.d.ward@duke.edu>> and Jacob M. Montgomery <<jacob.montgomery@wustl.edu>> and Florian M. Hollenbach <<florian.hollenbach@tamu.edu>>

References

Raftery, A. E., T. Gneiting, F. Balabdaoui and M. Polakowski. (2005). Using Bayesian Model Averaging to calibrate forecast ensembles. *Monthly Weather Review*. **133**:1155–1174.

Greenhill, B., M.D. Ward, A. Sacks. (2011). The Separation Plot: A New Visual Method For Evaluating the Fit of Binary Data. *American Journal of Political Science*.**55**: 991–1002.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2012). Improving Predictions Using Ensemble Bayesian Model Averaging. *Political Analysis*. **20**: 271-291.

Montgomery, Jacob M., Florian M. Hollenbach and Michael D. Ward. (2015). Calibrating ensemble forecasting models with sparse data in the social sciences. *International Journal of Forecasting*. **31**:930–942.

Examples

```
## Not run:  data(calibrationSample)
data(testSample)

this.ForecastData <- makeForecastData(.predCalibration=calibrationSample[,c("LMER", "SAE", "GLM")],
.outcomeCalibration=calibrationSample[,"Insurgency"],.predTest=testSample[,c("LMER", "SAE", "GLM")],
.outcomeTest=testSample[,"Insurgency"], .modelNames=c("LMER", "SAE", "GLM"))

this.ensemble <- calibrateEnsemble(this.ForecastData, model="logit", tol=0.001,exp=3)
```

```
summary(this.ensemble, period="calibration")

summary(this.ensemble, period="test",showCoefs=FALSE)

## End(Not run)
```

Index

∗ **datasets**
calibrationSample, 6
presidentialForecast, 13

calibrateEnsemble, 2
calibrateEnsemble,ForecastData-method *(calibrateEnsemble)*, 2
calibrationSample, 6
compareModels *(CompareModels-class)*, 7
compareModels,ForecastData-method *(CompareModels-class)*, 7
CompareModels-class, 7

EBMApredict, 9
EBMApredict,ForecastData-method *(EBMApredict)*, 9

ForecastData-class, 10

makeForecastData, 11
makeForecastData,ANY-method *(makeForecastData)*, 11

plot,FDatFitLogit-method *(SummaryForecastData-class)*, 17
plot,FDatFitNormal-method *(SummaryForecastData-class)*, 17
presidentialForecast, 13
print,ForecastData-method, 14
print,SummaryForecastData-method *(print,ForecastData-method)*, 14

setModelNames<- *(setPredCalibration<-)*, 15
setModelNames<-,ForecastData-method *(setPredCalibration<-)*, 15
setOutcomeCalibration<- *(setPredCalibration<-)*, 15
setOutcomeCalibration<-,ForecastData-method *(setPredCalibration<-)*, 15
setOutcomeTest<- *(setPredCalibration<-)*, 15
setOutcomeTest<-,ForecastData-method *(setPredCalibration<-)*, 15
setPredCalibration<-, 15
setPredCalibration<-,ForecastData-method *(setPredCalibration<-)*, 15
setPredTest<- *(setPredCalibration<-)*, 15
setPredTest<-,ForecastData-method *(setPredCalibration<-)*, 15
show,ForecastData-method *(print,ForecastData-method)*, 14
show,SummaryForecastData-method *(print,ForecastData-method)*, 14
summary,FDatFitLogit-method *(SummaryForecastData-class)*, 17
summary,FDatFitNormal-method *(SummaryForecastData-class)*, 17
SummaryForecastData-class, 17

testSample *(calibrationSample)*, 6