Package 'ForecastComb'

October 12, 2022

Type Package

Title Forecast Combination Methods

Version 1.3.1

Depends R (>= 3.0.2)

Imports forecast (>= 7.3), ggplot2 (>= 2.1.0), Matrix (>= 1.2-6), mtsdi (>= 0.3.3), psych (>= 1.6.9), quadprog (>= 1.5-5), quantreg (>= 5.29)

Suggests testthat (>= 1.0.2)

Author Christoph E. Weiss, Gernot R. Roetzer, Eran Raviv

Maintainer Christoph E. Weiss <info@ceweiss.com>

Description Provides geometric- and regression-based forecast combination methods under a unified user interface for the packages 'ForecastCombinations' and 'GeomComb'. Additionally, updated tools and convenience functions for data pre-processing are available in order to deal with common problems in forecast combination (missingness, collinearity). For method details see Hsiao C, Wan SK (2014). <doi:10.1016/j.jeconom.2013.11.003>, Hansen BE (2007). <doi:10.1111/j.1468-0262.2007.00785.x>, Elliott G, Gargano A, Timmermann A (2013). <doi:10.1016/j.jeconom.2013.04.017>, and Clemen RT (1989). <doi:10.1016/0169-2070(89)90012-5>.

URL https://github.com/ceweiss/ForecastComb

BugReports https://github.com/ceweiss/ForecastComb/issues

License GPL (>= 2)

LazyData true

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-08-07 13:50:08 UTC

R topics documented:

`auto_combine` *Automated Forecast Combination*

Description

Computes the fit for all the available forecast combination methods on the provided dataset with respect to the loss criterion. Returns the best fit method.

Usage

```
auto_combine(x, criterion = "RMSE", param_list = NULL)
```

Arguments

`x`	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
`criterion`	Specifies loss criterion. Set criterion to either 'RMSE' (default), 'MAE', or 'MAPE'.
`param_list`	Can contain additional parameters for the different combination methods (see example below).

Details

The function auto_combine allows to quickly apply all the different forecast combination methods onto the provided time series data and selects the method with the best fit.

The user can choose from 3 different loss criteria for the best-fit evaluation: root mean square error (criterion='RMSE'), mean absolute error (criterion='MAE'), and mean absolute percentage error (criterion='MAPE').

In case the user does not want to optimize over the parameters of some of the combination methods, auto_combine allows to specify the parameter values for these methods explicitly (see Examples).

The best-fit results are stored in an object of class 'foreccomb_res', for which separate plot and summary functions are provided.

Value

Returns an object of class foreccomb_res that represents the results for the best-fit forecast combination method:

Method	Returns the best-fit forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights obtained by applying the best-fit combination method to the training set.
Fitted	Returns the fitted values of the combination method for the training set.
Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]
```

```
data<-foreccomb(train_o, train_p, test_o, test_p)

# Evaluating all the forecast combination methods and returning the best.
# If necessary, it uses the built-in automated parameter optimisation methods
# for the different methods.
best_combination<-auto_combine(data, criterion = "MAPE")

# Same as above, but now we restrict the parameter ntop_pred for the method comb_EIG3 to be 3.
param_list<-list()
param_list$comb_EIG3$ntop_pred<-3
best_combination_restricted<-auto_combine(data, criterion = "MAPE", param_list = param_list)
```

---

comb_BG *Bates/Granger (1969) Forecast Combination Approach*

Description

Computes forecast combination weights according to the approach by Bates and Granger (1969) and produces forecasts for the test set, if provided.

Usage

```
comb_BG(x)
```

Arguments

x	An object of class `foreccomb`. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

In their seminal paper, Bates and Granger (1969) introduce the idea of combining forecasts. Their approach builds on portfolio diversification theory and uses the diagonal elements of the estimated mean squared prediction error matrix in order to compute combination weights:

$$w_i^{BG} = \frac{\hat{\sigma}^{-2}(i)}{\Sigma_{j=1}^{N} \hat{\sigma}^{-2}(j)}$$

where $\hat{\sigma}^{-2}(i)$ is the estimated mean squared prediction error of the i-th model.

The combined forecast is then obtained by:

$$\hat{y}_t = \mathbf{f}_t{}' \mathbf{w}^{BG}$$

Their approach ignores correlation between forecast models due to difficulties in precisely estimating the covariance matrix.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Bates, J. M., and Granger, C. W. (1969). The Combination of Forecasts. *Journal of the Operational Research Society*, **20(4)**, 451–468.

Timmermann, A. (2006). Forecast Combinations. In: Elliott, G., Granger, C. W. J., and Timmermann, A. (Eds.), *Handbook of Economic Forecasting*, **1**, 135–196.

See Also

`foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_BG(data)
```

comb_CLS *Constrained Least Squares Forecast Combination*

Description

Computes forecast combination weights using constrained least squares (CLS) regression.

Usage

```
comb_CLS(x)
```

Arguments

x	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

The function integrates the constrained least squares (CLS) forecast combination implementation of the *ForecastCombinations* package into ForecastComb. The implementation has improved robustness regarding multicollinearity.

Compared to the ordinary least squares forecast combination method, CLS forecast combination has the additional requirement that the weights, $\mathbf{w}^{CLS} = (w_1, \ldots, w_N)'$, sum up to 1 and that there is no intercept. That is, the combinations of comb_CLS are affine combinations.

This method was first introduced by Granger and Ramanathan (1984). The general appeal of the method is its ease of interpretation (the weights can be interpreted as percentages) and often produces better forecasts than the OLS method when the individual forecasts are highly correlated. A disadvantage is that if one or more individual forecasts are biased, this bias is not corrected through the forecast combination due to the lack of an intercept.

In addition to the version presented by Granger and Ramanathan (1984), this variant of the method adds the restriction that combination weights must be non-negative, which has been found to be almost always outperform unconstrained OLS by Aksu and Gunter (1992) and was combined with the condition of forcing the weights to sum up to one by Nowotarski et al. (2014), who conclude that even though the method provides a suboptimal solution in-sample, it almost always produces better forecasts than unrestricted OLS out-of-sample.

The results are stored in an object of class 'foreccomb_res', for which separate plot and summary functions are provided.

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the best-fit forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.

`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

References

Aksu, C., and Gunter, S. I. (1992). An Empirical Analysis of the Accuracy of SA, OLS, ERLS and NRLS Combination Forecasts. *International Journal of Forecasting*, **8(1)**, 27–43.

Granger, C., and Ramanathan, R. (1984). Improved Methods Of Combining Forecasts. *Journal of Forecasting*, **3(2)**, 197–204.

Nowotarski, J., Raviv, E., Tr\"uck, S., and Weron, R. (2014). An Empirical Comparison of Alternative Schemes for Combining Electricity Spot Price Forecasts. *Energy Economics*, **46**, 395–412.

See Also

`Forecast_comb`, `foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_CLS(data)
```

---

comb_CSR *All Possible Combinations Forecast Averaging*

Description

Combine different forecasts using complete subset regressions. Apart from the simple averaging, weights based on information criteria (AIC, corrected AIC, Hannan Quinn and BIC).

Usage

```
comb_CSR(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

OLS forecast combination is based on

$$obs_t = const + \sum_{i=1}^{p} w_i \widehat{obs}_{it} + e_t,$$

where obs is the observed values and $\widehat{obs}$ is the forecast, one out of the p forecasts available.

The function computes the complete subset regressions. So a matrix of forecasts based on all possible subsets of fhat is returned.

Those forecasts can later be cross-sectionally averaged (averaged over rows) to create a single combined forecast using weights which are based on the information criteria of the different individual regression, rather than a simple average.

Additional weight-vectors which are based on different information criteria are also returned. This is in case the user would like to perform the frequensit version of forecast averaging (see references for more details).

Although the function is geared towards forecast averaging, it can be used in any other application as a generic complete subset regression.

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the used forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights based on the different information criteria.
Fitted	Returns the fitted values for each information criterion.
Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

Author(s)

Eran Raviv and Gernot R. Roetzer

References

Hansen, B. (2008). Least-squares forecast averaging *Journal of Econometrics*, **146(2)**, 342–350.

Kapetanios, G., Labhard V., Price, S. (2008). Forecasting Using Bayesian and Information-Theoretic Model Averaging. *Journal of Business & Economic Statistics*, **26(1)**.

Koenker R. (2005). *Quantile Regression. Cambridge University Press*.

Graham, E., Garganob, A., Timmermann, A. (2013). Complete subset regressions. *Journal of Econometrics*, **177(2)**, 357–373.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, comb_NG, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_CSR(data)
```

---

comb_EIG1 *Standard Eigenvector Forecast Combination*

Description

Computes forecast combination weights according to the standard eigenvector approach by Hsiao and Wan (2014) and produces forecasts for the test set, if provided.

Usage

```
comb_EIG1(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

The standard eigenvector approach retrieves combination weights from the sample estimated mean squared prediction error matrix as follows: Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$, Σ is the (positive definite) mean squared prediction error matrix of $\mathbf{f}_t$ and $\mathbf{e}$ is an $N \times 1$ vector of $(1, \ldots, 1)'$. The N positive eigenvalues are then arranged in increasing order $(\Phi_1 = \Phi_{min}, \Phi_2, \ldots, \Phi_N)$, and $\mathbf{w^j}$ is defined as the eigenvector corresponding to Φ_j. The combination weights $\mathbf{w}^{EIG1} = (w_1, \ldots, w_N)'$ are then chosen corresponding to the minimum of $\left(\frac{\Phi_1}{d_1^2}, \frac{\Phi_2}{d_2^2}, \ldots, \frac{\Phi_N}{d_N^2}\right)$, denoted as $\mathbf{w}^l$, where $d_j = \mathbf{e}'\mathbf{w}^j$, as:

$$\mathbf{w}^{EIG1} = \frac{1}{d_l}\mathbf{w}^l$$

The combined forecast is then obtained by:

$$\hat{y}_t = \mathbf{f}_t{}'\mathbf{w}^{EIG1}$$

The difference to extant methods that minimize the population mean squared prediction error (e.g., Newbold and Granger, 1974) is the normalization function. While previous approaches optimize MSPE under the constraint of $\mathbf{e}'\mathbf{w} = 1$, Hsiao and Wan (2014) show that this is dominated by using $\mathbf{w}'\mathbf{w} = 1$ as constraint in the optimization problem.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

Newbold, P., and Granger, C. W. J. (1974). Experience with Forecasting Univariate Time Series and the Combination of Forecasts. *Journal of the Royal Statistical Society, Series A*, **137(2)**, 131–165.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, comb_NG, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG1(data)
```

---

comb_EIG2 *Bias-Corrected Eigenvector Forecast Combination*

Description

Computes forecast combination weights according to the bias-corrected eigenvector approach by Hsiao and Wan (2014) and produces forecasts for the test set, if provided.

Usage

```
comb_EIG2(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

The bias-corrected eigenvector approach builds on the idea that if one or more of the predictive models yield biased predictions, the accuracy of the standard eigenvector approach can be improved by eliminating the bias. The optimization procedure to obtain combination weights coincides with the standard eigenvector approach, except that it is applied to the centered MSPE matrix after extracting the bias (by subtracting the column means of the MSPE).

The combination weights are calculated as:

$$\mathbf{w}^{EIG2} = \frac{1}{\tilde{d}_l}\tilde{\mathbf{w}}^l$$

where $\tilde{d}_j$ and $\tilde{\mathbf{w}}^j$ are defined analogously to d_j and $\mathbf{w}^j$ in the standard eigenvector approach, with the only difference that they correspond to the spectral decomposition of the centered MSPE matrix rather than the uncentered one.

The combined forecast is then obtained by:

$$\hat{y}_t = a + \mathbf{f}_t{}'\mathbf{w}^{EIG2}$$

where $a = E(y_t) - E(\mathbf{f}_t)'\mathbf{w}^{EIG2}$ is the intercept for bias correction. If the actual series and the forecasts are stationary, the expectations can be approximated by the time series means, i.e. the intercept is obtained by subtracting the weighted sum of column means of the MSPE matrix from the mean of the actual series. Forecast combination methods including intercepts therefore usually require stationarity.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Intercept`	Returns the intercept (bias correction).
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

See Also

`comb_EIG1`, `foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG2(data)
```

`comb_EIG3` *Trimmed Eigenvector Forecast Combination*

Description

Computes forecast combination weights according to the trimmed eigenvector approach by Hsiao and Wan (2014) and produces forecasts for the test set, if provided.

Usage

```
comb_EIG3(x, ntop_pred = NULL, criterion = "RMSE")
```

Arguments

`x`	An object of class `foreccomb`. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
`ntop_pred`	Specifies the number of retained predictors. If `NULL` (default), the inbuilt optimization algorithm selects this number.
`criterion`	If `ntop_pred` is not specified, a selection criterion is required for the optimization algorithm: one of "MAE", "MAPE", or "RMSE". If `ntop_pred` is selected by the user, `criterion` should be set to `NULL` (default).

Details

The underlying methodology of the trimmed eigenvector approach by Hsiao and Wan (2014) is the same as their `standard eigenvector approach`. The only difference is that the trimmed eigenvector approach pre-selects the models that serve as input for the forecast combination, only a subset of the available forecast models is retained, while the models with the worst performance are discarded.

The number of retained forecast models is controlled via `ntop_pred`. The user can choose whether to select this number, or leave the selection to the inbuilt optimization algorithm (in that case `ntop_pred = NULL`). If the optimization algorithm should select the best number of retained models, the user must select the optimization `criterion`: MAE, MAPE, or RMSE. After this trimming step, the weights and the combined forecast are computed in the same way as in the `standard eigenvector approach`.

The trimmed eigenvector approach takes note of the eigenvector approaches' property to treat y and $\mathbf{f}$ symmetrically, which bears the risk that the (non-trimmed) eigenvector approaches' performance could be severely impaired by one or a few models that produce forecasts much worse than the average.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Top_Predictors`	Number of retained predictors.
`Ranking`	Ranking of the predictors that determines which models are removed in the trimming step.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

See Also

`comb_EIG1` `foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]
```

```
## Number of retained models selected by the user:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG3(data, ntop_pred = 2, criterion = NULL)

## Number of retained models selected by algorithm:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG3(data, ntop_pred = NULL, criterion = "RMSE")
```

---

comb_EIG4 *Trimmed Bias-Corrected Eigenvector Forecast Combination*

Description

Computes forecast combination weights according to the trimmed bias-corrected eigenvector approach by Hsiao and Wan (2014) and produces forecasts for the test set, if provided.

Usage

```
comb_EIG4(x, ntop_pred = NULL, criterion = "RMSE")
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
ntop_pred	Specifies the number of retained predictors. If NULL (default), the inbuilt optimization algorithm selects this number.
criterion	If ntop_pred is not specified, a selection criterion is required for the optimization algorithm: one of "MAE", "MAPE", or "RMSE". If ntop_pred is selected by the user, criterion should be set to NULL (default).

Details

The underlying methodology of the trimmed bias-corrected eigenvector approach by Hsiao and Wan (2014) is the same as their bias-corrected eigenvector approach. The only difference is that the bias-corrected trimmed eigenvector approach pre-selects the models that serve as input for the forecast combination, only a subset of the available forecast models is retained, while the models with the worst performance are discarded.

The number of retained forecast models is controlled via ntop_pred. The user can choose whether to select this number, or leave the selection to the inbuilt optimization algorithm (in that case ntop_pred = NULL). If the optimization algorithm should select the best number of retained models, the user must select the optimization criterion: MAE, MAPE, or RMSE. After this trimming step, the weights, the intercept and the combined forecast are computed in the same way as in the bias-corrected eigenvector approach.

The bias-corrected trimmed eigenvector approach combines the strengths of the bias-corrected eigenvector approach and the trimmed eigenvector approach.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Intercept`	Returns the intercept (bias correction).
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Top_Predictors`	Number of retained predictors.
`Ranking`	Ranking of the predictors that determines which models are removed in the trimming step.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

See Also

`comb_EIG2` `comb_EIG3` `foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

## Number of retained models selected by the user:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG4(data, ntop_pred = 2, criterion = NULL)
```

```
## Number of retained models selected by algorithm:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_EIG4(data, ntop_pred = NULL, criterion = "RMSE")
```

---

comb_InvW *Inverse Rank Forecast Combination*

Description

Computes forecast combination weights according to the inverse rank approach by Aiolfi and Timmermann (2006) and produces forecasts for the test set, if provided.

Usage

```
comb_InvW(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

In the inverse rank approach by Aiolfi and Timmermann (2006), the combination weights are inversely proportional to the forecast model's rank, $Rank_i$:

$$w_i^{InvW} = \frac{Rank_i^{-1}}{\Sigma_{j=1}^{N} Rank_j^{-1}}$$

The combined forecast is then obtained by:

$$\hat{y}_t = \mathbf{f}_t{}' \mathbf{w}^{InvW}$$

This is a robust variant of the Bates/Granger (1969) approach and also ignores correlations across forecast errors.

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the used forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights obtained by applying the combination method to the training set.
Fitted	Returns the fitted values of the combination method for the training set.

Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Aiolfi, M., amd Timmermann, A. (2006). Persistence in Forecasting Performance and Conditional Combination Strategies. *Journal of Econometrics*, **135(1)**, 31–53.

Bates, J. M., and Granger, C. W. (1969). The Combination of Forecasts. *Journal of the Operational Research Society*, **20(4)**, 451–468.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, comb_BG, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_InvW(data)
```

---

comb_LAD *Least Absolute Deviation Forecast Combination*

Description

Computes forecast combination weights using least absolute deviation (LAD) regression.

Usage

```
comb_LAD(x)
```

Arguments

x	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

The function integrates the least absolute deviation (LAD) forecast combination implementation of the *ForecastCombinations* package into ForecastComb.

The defining property of comb_LAD is that it does not minimize the squared error loss like comb_OLS and comb_CLS, but the absolute values of the errors. This makes the method more robust to outliers – comb_LAD tends to penalize models, which have high errors for some observations, less harshly than the least squares methods would.

Optimal forecast combinations under general loss functions are discussed by Elliott and Timmermann (2004). The LAD method is described in more detail, and used in an empirical context, by Nowotarksi et al. (2014).

The results are stored in an object of class 'foreccomb_res', for which separate plot and summary functions are provided.

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the best-fit forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights obtained by applying the combination method to the training set.
Intercept	Returns the intercept of the linear regression.
Fitted	Returns the fitted values of the combination method for the training set.
Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

References

Elliott, G., and Timmermann, A. (2004). Optimal Forecast Combinations Under General Loss Functions and Forecast Error Distributions. *Journal of Econometrics*, **122(1)**, 47–79.

Nowotarski, J., Raviv, E., Tr\"uck, S., and Weron, R. (2014). An Empirical Comparison of Alternative Schemes for Combining Electricity Spot Price Forecasts. *Energy Economics*, **46**, 395–412.

See Also

Forecast_comb, foreccomb, plot.foreccomb_res, summary.foreccomb_res, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_LAD(data)
```

---

comb_MED *Median Forecast Combination*

Description

Computes a 'combined forecast' from a pool of individual model forecasts using their median at each point in time.

Usage

```
comb_MED(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$. For each point in time, the median method gives a weight of 1 to the median forecast and a weight of 0 to all other forecasts, the combined forecast is obtained by:

$$\hat{y}_t = median(\mathbf{f}_t)$$

The median method is an appealing simple, rank-based combination method that has been proposed by authors such as Armstrong (1989), McNees (1992), Hendry and Clements (2004), Stock and Watson (2004), and Timmermann (2006). It is more robust to outliers than the simple average approach.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Armstrong, J. S. (1989). Combining Forecasts: The End of the Beginning or the Beginning of the End?. *International Journal of Forecasting*, **5(4)**, 585–588.

Hendry, D. F., and Clements, M. P. (2004). Pooling of Forecasts. *The Econometrics Journal*, **7(1)**, 1–31.

McNees, S. K. (1992). The Uses and Abuses of 'Consensus' Forecasts. *Journal of Forecasting*, **11(8)**, 703–710.

Stock, J. H., and Watson, M. W. (2004). Combination Forecasts of Output Growth in a Seven-Country Data Set. *Journal of Forecasting*, **23(6)**, 405–430.

Timmermann, A. (2006). Forecast Combinations. In: Elliott, G., Granger, C. W. J., and Timmermann, A. (Eds.), *Handbook of Economic Forecasting*, **1**, 135–196.

See Also

`foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `comb_SA`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]
```

```
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_MED(data)
```

---

comb_NG *Newbold/Granger (1974) Forecast Combination*

Description

Computes forecast combination weights according to the approach by Newbold and Granger (1974) and produces forecasts for the test set, if provided.

Usage

```
comb_NG(x)
```

Arguments

x	An object of class `foreccomb`. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

Building on early research by Bates and Granger (1969), the methodology of Newbold and Granger (1974) also extracts the combination weights from the estimated mean squared prediction error matrix.

Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$, Σ is the (positive definite) mean squared prediction error matrix of $\mathbf{f}_t$ and $\mathbf{e}$ is an $N \times 1$ vector of $(1, \ldots, 1)'$.

Their approach is a constrained minimization of the mean squared prediction error using the normalization condition $\mathbf{e}'\mathbf{w} = 1$. This yields the following combination weights:

$$\mathbf{w}^{NG} = \frac{\Sigma^{-1}\mathbf{e}}{\mathbf{e}'\Sigma^{-1}\mathbf{e}}$$

The combined forecast is then obtained by:

$$\hat{y}_t = \mathbf{f}_t{}'\mathbf{w}^{NG}$$

While the method dates back to Newbold and Granger (1974), the variant of the method used here does not impose the prior restriction that Σ is diagonal. This approach, called `VC` in Hsiao and Wan (2014), is a generalization of the original method.

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Bates, J. M., and Granger, C. W. (1969). The Combination of Forecasts. *Journal of the Operational Research Society*, **20(4)**, 451–468.

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

Newbold, P., and Granger, C. W. J. (1974). Experience with Forecasting Univariate Time Series and the Combination of Forecasts. *Journal of the Royal Statistical Society, Series A*, **137(2)**, 131–165.

See Also

`comb_BG`, `comb_EIG1`, `foreccomb`, `plot.foreccomb_res`, `summary.foreccomb_res`, `accuracy`

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_NG(data)
```

comb_OLS *Ordinary Least Squares Forecast Combination*

Description

Computes forecast combination weights using ordinary least squares (OLS) regression.

Usage

```
comb_OLS(x)
```

Arguments

x	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

The function integrates the ordinary least squares (OLS) forecast combination implementation of the *ForecastCombinations* package into ForecastComb.

The OLS combination method (Granger and Ramanathan (1984)) uses ordinary least squares to estimate the weights, $\mathbf{w}^{OLS} = (w_1, \ldots, w_N)'$, as well as an intercept, b, for the combination of the forecasts.

Suppose that there are N not perfectly collinear predictors $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$, then the forecast combination for one data point can be represented as:

$$y_t = b + \sum_{i=1}^{N} w_i f_{it}$$

An appealing feature of the method is its bias correction through the intercept – even if one or more of the individual predictors are biased, the resulting combined forecast is unbiased. A disadvantage of the method is that it places no restriction on the combination weights (i.e., they do not add up to 1 and can be negative), which can make interpretation hard. Another issue, documented in Nowotarski et al. (2014), is the method's unstable behavior when predictors are highly correlated (which is the norm in forecast combination): Minor fluctuations in the sample can cause major shifts of the coefficient vector ('bouncing betas') – often causing poor out-of-sample performance. This issue is addressed by the comb_LAD method that is more robust to outliers.

The results are stored in an object of class 'foreccomb_res', for which separate plot and summary functions are provided.

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the best-fit forecast combination method.

Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights obtained by applying the combination method to the training set.
Intercept	Returns the intercept of the linear regression.
Fitted	Returns the fitted values of the combination method for the training set.
Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

References

Granger, C., and Ramanathan, R. (1984). Improved Methods Of Combining Forecasts. *Journal of Forecasting*, **3(2)**, 197–204.

Nowotarski, J., Raviv, E., Tr\"uck, S., and Weron, R. (2014). An Empirical Comparison of Alternative Schemes for Combining Electricity Spot Price Forecasts. *Energy Economics*, **46**, 395–412.

See Also

Forecast_comb, foreccomb, plot.foreccomb_res, summary.foreccomb_res, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_OLS(data)
```

---

comb_SA *Simple Average Forecast Combination*

Description

Computes forecast combination weights using simple average and produces forecasts for the test set, if provided.

Usage

```
comb_SA(x)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Details

Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$. The simple average gives equal weights to all predictors:

$$\mathbf{w}^{SA} = \frac{1}{N}$$

The combined forecast is then obtained by:

$$\hat{y}_t = \mathbf{f}_t{}'\mathbf{w}^{SA}$$

It is well-documented that simple average is a robust combination method that is hard to beat (e.g., Stock and Watson, 2004; Timmermann, 2006). This is often associated with the importance of parameter estimation error in sophisticated techniques – a problem that simple averaging avoids. However, simple averaging may not be a suitable combination method when some of the predictors are biased (Palm and Zellner, 1992).

Value

Returns an object of class foreccomb_res with the following components:

Method	Returns the used forecast combination method.
Models	Returns the individual input models that were used for the forecast combinations.
Weights	Returns the combination weights obtained by applying the combination method to the training set.
Fitted	Returns the fitted values of the combination method for the training set.
Accuracy_Train	Returns range of summary measures of the forecast accuracy for the training set.
Forecasts_Test	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Palm, F. C., and Zellner, A. (1992). To Combine or not to Combine? Issues of Combining Forecasts. *Journal of Forecasting*, **11(8)**, 687–701.

Stock, J. H., and Watson, M. W. (2004). Combination Forecasts of Output Growth in a Seven-Country Data Set. *Journal of Forecasting*, **23(6)**, 405–430.

Timmermann, A. (2006). Forecast Combinations. In: Elliott, G., Granger, C. W. J., and Timmermann, A. (Eds.), *Handbook of Economic Forecasting*, **1**, 135–196.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
comb_SA(data)
```

---

comb_TA *Trimmed Mean Forecast Combination*

Description

Computes a 'combined forecast' from a pool of individual model forecasts using trimmed mean at each point in time.

Usage

```
comb_TA(x, trim_factor = NULL, criterion = "RMSE")
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
trim_factor	numeric. Must be between 0 (simple average) and 0.5 (median).
criterion	If trim_factor is not specified, an optimization criterion for automated trimming needs to be defined. One of "MAE", "MAPE", or "RMSE" (default).

Details

Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$. For each point in time, the order forecasts are computed:

$$\mathbf{f}_t^{ord} = (f_{(1)t}, \ldots, f_{(N)t})'$$

Using a trim factor λ (i.e., the top/bottom $\lambda\%$ are trimmed) the combined forecast is calculated as:

$$\hat{y}_t = \frac{1}{N(1-2\lambda)} \sum_{i=\lambda N+1}^{(1-\lambda)N} f_{(i)t}$$

The trimmed mean is an interpolation between the simple average and the median. It is an appealing simple, rank-based combination method that is less sensitive to outliers than the simple average approach, and has been proposed by authors such as Armstrong (2001), Stock and Watson (2004), and Jose and Winkler (2008).

This method allows the user to select λ (by specifying `trim_factor`), or to leave the selection to an optimization algorithm – in which case the optimization criterion has to be selected (one of "MAE", "MAPE", or "RMSE").

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Trim Factor`	Returns the trim factor, λ.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Armstrong, J. S. (2001). Combining Forecasts. In: *Armstrong, J. S. (Ed.), Principles of Forecasting. Springer, Boston, MA*, 417–439.

Jose, V. R. R., and Winkler, R. L. (2008). Simple Robust Averages of Forecasts: Some Empirical Results. *International Journal of Forecasting*, **24(1)**, 163–169.

Stock, J. H., and Watson, M. W. (2004). Combination Forecasts of Output Growth in a Seven-Country Data Set. *Journal of Forecasting*, **23(6)**, 405–430.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res, comb_SA, comb_MED, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

## User-selected trim factor:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_TA(data, trim_factor=0.1)

## Algorithm-optimized trim factor:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_TA(data, criterion="RMSE")
```

---

comb_WA *Winsorized Mean Forecast Combination*

Description

Computes a 'combined forecast' from a pool of individual model forecasts using winsorized mean at each point in time.

Usage

```
comb_WA(x, trim_factor = NULL, criterion = "RMSE")
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
trim_factor	numeric. Must be between 0 and 0.5.
criterion	If trim_factor is not specified, an optimization criterion for automated trimming needs to be defined. One of "MAE", "MAPE", or "RMSE" (default).

Details

Suppose y_t is the variable of interest, there are N not perfectly collinear predictors, $\mathbf{f}_t = (f_{1t}, \ldots, f_{Nt})'$. For each point in time, the order forecasts are computed:

$$\mathbf{f}_t^{ord} = (f_{(1)t}, \ldots, f_{(N)t})'$$

Using a trim factor λ (i.e., the top/bottom $\lambda\%$ are winsorized), and setting $K = N\lambda$, the combined forecast is calculated as (Jose and Winkler, 2008):

$$\hat{y}_t = \frac{1}{N}\left[Kf_{(K+1)t} + \sum_{i=K+1}^{N-K} f_{(i)t} + Kf_{(N-K)t}\right]$$

Like the trimmed mean, the winsorized mean is a robust statistic that is less sensitive to outliers than the simple average. It is less extreme about handling outliers than the trimmed mean and preferred by Jose and Winkler (2008) for this reason.

This method allows the user to select λ (by specifying `trim_factor`), or to leave the selection to an optimization algorithm – in which case the optimization criterion has to be selected (one of "MAE", "MAPE", or "RMSE").

Value

Returns an object of class `foreccomb_res` with the following components:

`Method`	Returns the used forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the combination method to the training set.
`Trim Factor`	Returns the trim factor, λ.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.
`Accuracy_Test`	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
`Input_Data`	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

References

Jose, V. R. R., and Winkler, R. L. (2008). Simple Robust Averages of Forecasts: Some Empirical Results. *International Journal of Forecasting*, **24(1)**, 163–169.

See Also

winsor.mean, foreccomb, plot.foreccomb_res, summary.foreccomb_res, comb_SA, comb_TA, accuracy

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

## User-selected trim factor:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_TA(data, trim_factor=0.1)

## Algorithm-optimized trim factor:
data<-foreccomb(train_o, train_p, test_o, test_p)
comb_TA(data, criterion="RMSE")
```

---

cs_dispersion *Compute Cross-Sectional Dispersion*

Description

Computes (time-varying) dispersion measures for the cross section of individual model forecasts that are the input of forecast combination.

Usage

```
cs_dispersion(x, measure = "SD", plot = FALSE)
```

Arguments

x	An object of class foreccomb. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
measure	Cross-sectional dispersion measure, one of: "SD" = standard deviation (default); "IQR" = interquartile range; or "Range" = range.
plot	logical. If TRUE, evolution of cross-sectional forecast dispersion is plotted as ggplot.

Details

The available measures of scale are defined as in Davison (2003). Let $y_{(i)}$ denote the i-th order statistic of the sample, then:

$$Range_t = y_{(n),t} - y_{(1),t}$$

$$IQR_t = y_{(3n/4),t} - y_{(n/4),t}$$

$$SD_t = \sqrt{\frac{1}{n-1}\Sigma_{i=1}^{n}\left(y_{i,t} - \bar{y}_t\right)}$$

Previous research in the forecast combination literature has documented that regression-based combination methods tend to have relative advantage when one or more individual model forecasts are better than the rest, while eigenvector-based methods tend to have relative advantage when individual model forecasts are in the same ball park.

Value

Returns a vector of the evolution of cross-sectional dispersion over the sample period (using the selected dispersion measure)

References

Davison, A. C. (2003). Statistical Models. *Cambridge University Press*.

Hsiao, C., and Wan, S. K. (2014). Is There An Optimal Forecast Combination? *Journal of Econometrics*, **178(2)**, 294–309.

See Also

foreccomb, sd, IQR, range

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
cs_dispersion(data, measure = "IQR")
```

electricity *UK Electricity Supply 2007 - 2017*

Description

The `electricity` dataset contains monthly data on the total UK electricity supply in GWh from January 2007 to March 2017, as well as univariate time series forecasts for this series.

Usage

```
data(electricity)
```

Format

A multivariate time series of 123 observations; monthly, 2007-2017.

This data contains the following columns:

arima (ARIMA 1-month forecasts)

ets (ETS 1-month forecasts)

nnet (Neural Network 1-month forecasts)

dampedt (Damped Trend 1-month forecasts)

dotm (Dynamic Optimized Theta 1-month forecasts)

Actual (Observed values)

Source

International Energy Agency (2017). IEA Monthly Electricity Statistics. Available at http://www.iea.org/statistics/monthlystatistics/monthlyelectricitystatistics/

foreccomb *Format Raw Data for Forecast Combination*

Description

Structures cross-sectional input data (individual model forecasts) for forecast combination. Stores data as S3 class `foreccomb` that serves as input to the forecast combination techniques. Handles missing value imputation (optional) and resolves problems due to perfect collinearity.

Usage

```
foreccomb(observed_vector, prediction_matrix, newobs = NULL,
  newpreds = NULL, byrow = FALSE, na.impute = TRUE, criterion = "RMSE")
```

Arguments

observed_vector	A vector or univariate time series; contains 'actual values' for training set.
prediction_matrix	A matrix or multivariate time series; contains individual model forecasts for training set.
newobs	A vector or univariate time series; contains 'actual values' if a test set is used (optional).
newpreds	A matrix or multivariate time series; contains individual model forecasts if a test set is used (optional). Does not require specification of newobs – in the case in which a forecaster only wants to train the forecast combination method with a training set and apply it to future individual model forecasts, only newpreds is required, not newobs.
byrow	logical. The default (FALSE) assumes that each column of the forecast matrices (prediction_matrix and – if specified – newpreds) contains forecasts from one forecast model; if each row of the matrices contains forecasts from one forecast model, set to TRUE.
na.impute	logical. The default (TRUE) behavior is to impute missing values via the cross-validated spline approach of the mtsdi package. If set to FALSE, forecasts with missing values will be removed. Missing values in the observed data are never imputed.
criterion	One of "RMSE" (default), "MAE", or "MAPE". Is only used if prediction_matrix is not full rank: The algorithm checks which models are causing perfect collinearity and the one with the worst individual accuracy (according to the chosen criterion) is removed.

Details

The function imports the column names of the prediction matrix (if byrow = FALSE, otherwise the row names) as model names; if no column names are specified, generic model names are created.

The missing value imputation algorithm is a modified version of the EM algorithm for imputation that is applicable to time series data - accounting for correlation between the forecasting models and time structure of the series itself. A smooth spline is fitted to each of the time series at each iteration. The degrees of freedom of each spline are chosen by cross-validation.

Forecast combination relies on the lack of perfect collinearity. The test for this condition checks if prediction_matrix is full rank. In the presence of perfect collinearity, the iterative algorithm identifies the subset of forecasting models that are causing linear dependence and removes the one among them that has the lowest accuracy (according to a selected criterion, default is RMSE). This procedure is repeated until the revised prediction matrix is full rank.

Value

Returns an object of class foreccomb.

Author(s)

Christoph E. Weiss, Gernot R. Roetzer

References

Junger, W. L., Ponce de Leon, A., and Santos, N. (2003). Missing Data Imputation in Multivariate Time Series via EM Algorithm. *Cadernos do IME*, **15**, 8–21.

Dempster, A., Laird, N., and Rubin D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, **39(1)**, 1–38.

See Also

mnimput, rankMatrix

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

## Example with a training set only:
foreccomb(train_o, train_p)

## Example with a training set and future individual forecasts:
foreccomb(train_o, train_p, newpreds=test_p)

## Example with a training set and a full test set:
foreccomb(train_o, train_p, test_o, test_p)

## Example with forecast models being stored in rows:
preds_row <- matrix(rnorm(1000, 1), 10, 100)
train_p_row <- preds_row[,1:80]
foreccomb(train_o, train_p_row, byrow = TRUE)

## Example with NA imputation:
train_p_na <- train_p
train_p_na[2,3] <- NA
foreccomb(train_o, train_p_na, na.impute = TRUE)

## Example with perfect collinearity:
train_p[,2] <- 0.8*train_p[,1] + 0.4*train_p[,8]
foreccomb(train_o, train_p, criterion="RMSE")
```

---

foreccomb_res *Result Rbject for the Forecast Combination Methods*

Description

Stores the results and inputs for some combined forecasts.

Usage

```
foreccomb_res(method, modelnames, fitted, accuracy_insample, input_data,
  predict = NULL, intercept = NULL, weights = NULL, pred = NULL,
  accuracy_outsample = NULL, trim_factor = NULL, top_predictors = NULL,
  ranking = NULL)
```

Arguments

method	Name of the method.
modelnames	Names of the models provided by the user.
fitted	The fitted values.
accuracy_insample	Insample accuracy obtained by the method.
input_data	Contains the input data provided by the user.
predict	(optional) Function used to conduct predictions for new forecasts.
intercept	(optional) Intercept of the method, if it requires one.
weights	(optional) Weights of the method, if it requires one.
pred	= (optional) Predictions on a test set.
accuracy_outsample	= (optional) Accuracy on the test set.
trim_factor	= (optional) Trim factor used in some of the methods.
top_predictors	(optional) Number of retained predictors.
ranking	(optional) Ranking of the predictors that determines which models are removed in the trimming step.

Value

Returns an object of class `foreccomb_res`.

Author(s)

Christoph E. Weiss, Gernot R. Roetzer

---

plot.foreccomb_res *Plot results from forecast combination model*

Description

Produces plots for the results of a forecast combination method. Either an actual vs. fitted plot (`which = 1`) or a barplot of the combination weights (`which = 2`).

Usage

```
## S3 method for class 'foreccomb_res'
plot(x, which = 1, ...)
```

Arguments

x	An object of class 'foreccomb_res'.
which	Type of plot: 1 = actual vs. fitted, 2 = combination weights.
...	Other arguments passing to plot.default.

Value

A plot for the foreccomb_res class.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

See Also

foreccomb, summary.foreccomb_res

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
fit <- comb_EIG1(data)
plot(fit)
```

---

predict.foreccomb_res *Prediction function for Forecast Combinations*

Description

prediction method for class 'foreccomb_res'. Uses the previously created forecast combination result to predict the combination for a newly provided prediction dataset.

Usage

```
## S3 method for class 'foreccomb_res'
predict(object, newpreds, newobs = NULL,
  simplify = TRUE, byrow = FALSE, ...)
```

Arguments

object	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
newpreds	A matrix or multivariate time series; contains individual model forecasts if a test set is used (optional). Does not require specification of newobs – in the case in which a forecaster only wants to train the forecast combination method with a training set and apply it to future individual model forecasts, only newpreds is required, not newobs.
newobs	A vector or univariate time series; contains 'actual values' if a test set is used (optional).
simplify	logical. The default (TRUE) returns the predictions separately. If set to (FALSE) the predictions are incorporated into the foreccomb_res object, that is, the object is equal to the one that would have been obtained, if the new prediction set would have been provided when the forecast combination method was trained originally.
byrow	logical. The default (FALSE) assumes that each column of the forecast matrices (prediction_matrix and – if specified – newpreds) contains forecasts from one forecast model; if each row of the matrices contains forecasts from one forecast model, set to TRUE.
...	potential further arguments (require by generic)

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

See Also

foreccomb,

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p)
fit<-comb_BG(data)
predict(fit, test_p)
```

`rolling_combine` *Dynamic Forecast Combination*

Description

Computes the dynamic version of the combined forecast for a method included in the ForecastComb package.

Usage

```
rolling_combine(x, comb_method, criterion = NULL)
```

Arguments

`x`	An object of class 'foreccomb'. Must contrain full training set and test set.
`comb_method`	The combination method that should be used.
`criterion`	Specifies loss criterion. Set criterion to either 'RMSE', 'MAE', or 'MAPE' for the methods `comb_TA`, `comb_WA`, `comb_EIG3`, and `comb_EIG4`, or to 'NULL' (default) for all other methods.

Details

The function `rolling_combine` allows to estimate a dynamic version of the other combination methods of the package in a standardized way, i.e., it allows for time-varying weights. The function builds on the idea of time series cross-validation: Taking the provided training set as starting point, the models are re-estimated at each period of the test set using a revised (increased) training set.

Like univariate dynamic forecasting, the validation approach requires a full test set – including the observed values.

The results are stored in an object of class 'foreccomb_res', for which separate plot and summary functions are provided.

Value

Returns an object of class `foreccomb_res` that represents the results for the best-fit forecast combination method:

`Method`	Returns the best-fit forecast combination method.
`Models`	Returns the individual input models that were used for the forecast combinations.
`Weights`	Returns the combination weights obtained by applying the best-fit combination method to the training set.
`Fitted`	Returns the fitted values of the combination method for the training set.
`Accuracy_Train`	Returns range of summary measures of the forecast accuracy for the training set.
`Forecasts_Test`	Returns forecasts produced by the combination method for the test set. Only returned if input included a forecast matrix for the test set.

Accuracy_Test	Returns range of summary measures of the forecast accuracy for the test set. Only returned if input included a forecast matrix and a vector of actual values for the test set.
Input_Data	Returns the data forwarded to the method.

Author(s)

Christoph E. Weiss

References

Bergmeir, C., Hyndman, R.J., and Koo, B. (2015). A Note on the Validity of Cross-Validation for Evaluating Time Series Prediction. *Monash University, Deparment of Econometrics and Business Statistics*, Working Paper No. 10/15.

Timmermann, A. (2006). Forecast Combinations. *Handbook of Economic Forecasting*, **1**, 135–196.

See Also

foreccomb, plot.foreccomb_res, summary.foreccomb_res,

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)

#Static forecast combination (for example OLS):
static_OLS <- comb_OLS(data)

#Dynamic forecast combination:
dyn_OLS <- rolling_combine(data, "comb_OLS")
```

---

summary.foreccomb_res *Summary of Forecast Combination*

Description

summary method for class 'foreccomb_res'. Includes information about combination method, combination weights assigned to the individual forecast models, as well as an accuracy evaluation of the combined forecast.

Usage

```
## S3 method for class 'foreccomb_res'
summary(object, ...)

## S3 method for class 'foreccomb_res_summary'
print(x, ...)
```

Arguments

object	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.
...	potential further arguments (require by generic)
x	An object of class 'foreccomb'. Contains training set (actual values + matrix of model forecasts) and optionally a test set.

Author(s)

Christoph E. Weiss and Gernot R. Roetzer

See Also

foreccomb, plot.foreccomb_res,

Examples

```
obs <- rnorm(100)
preds <- matrix(rnorm(1000, 1), 100, 10)
train_o<-obs[1:80]
train_p<-preds[1:80,]
test_o<-obs[81:100]
test_p<-preds[81:100,]

data<-foreccomb(train_o, train_p, test_o, test_p)
fit<-comb_BG(data)
summary(fit)
```

Index