Package ‘EWS’

April 7, 2020

Type Package

Title Early Warning System

Version 0.1.0

Description Early Warning Systems (EWS) are a toolbox for policymakers to prevent or attenuate the impact of economic downturns. Modern EWS are based on the econometric framework of Kauppi and Saikkonen (2008) <doi:10.1162/rest.90.4.777>. Specifically, this framework includes four dichotomous models, relying on a logit approach to model the relationship between yield spreads and future recessions, controlling for recession risk factors. These models can be estimated in a univariate or a balanced panel framework as in Candelon, Dumitrescu and Hurlin (2014) <doi:10.1016/j.ijforecast.2014.03.015>. This package provides both methods for estimating these models and a dataset covering 13 OECD countries over a period of 45 years. This package constitutes a useful toolbox (data and functions) for scholars as well as policymakers.

Depends R (>= 2.10)

License GPL-3

Encoding UTF-8

LazyData true

Imports numDeriv

NeedsCompilation no

Author Quentin Lajaunie [cre],
Jean-Baptiste Hasse [aut]

Maintainer Quentin Lajaunie <quentin_lajaunie@hotmail.fr>

Repository CRAN

Date/Publication 2020-04-07 15:00:22 UTC

R topics documented:

data_panel *Historical data for 13 OECD countries*

Description

data_USA contains: - OECD based Recession Indicators for 13 OECD countries from the Peak through the Trough from 1975:03 to 2019:05 - Yield Spread (10Years TB minus 3Months TB) for 13 OECD countries from 1975:03 to 2019:05

List of countries: Australia, Belgium, Canada, France, Germany, Italy, Japan, the Netherlands, New Zealand, Sweden, Switzerland, the United Kinkdom, the United States.

Usage

```
data("data_panel")
```

Format

A data frame with 6903 observations on the following 4 variables.

`country` List of countries.

`Date` Vector of dates.

`YIESPR` historical yield spread for the 13 OECD countries.

`OECD_Recession` Historical binary variable related to historical recessions for the 13 OECD countries.

Source

https://fred.stlouisfed.org/

data_USA *Historical data for the United States*

Description

data_USA contains: - NBER based Recession Indicators for the United States from 1975:03 to 2019:05 - Yield Spread (10Years TB minus 3Months TB) for the United States from 1975:03 to 2019:05

Usage

```
data("data_USA")
```

Format

A data frame with 531 observations on the following 4 variables.

`country` USA.

`Date` Vector of dates.

`YIESPR` Historical yield spread.

`NBER` Historical binary variable related to historical recessions.

Source

https://fred.stlouisfed.org/

`Logistic_Estimation` *Logistic Estimation for Dichotomous Analysis*

Description

This function provides methods for estimating the four dichotomous models as in Kauppi & Saikkonen (2008). Based on a logit approach, models are estimated in a univariate or a balanced panel framework as in Candelon, Dumitrescu and Hurlin (2014). This estimation has been used in recent papers such in Ben Naceur, Candelon and Lajaunie (2019) and Hasse and Lajaunie (2020).

Usage

```
Logistic_Estimation(Dicho_Y, Exp_X, Intercept, Nb_Id, Lag, type_model)
```

Arguments

`Dicho_Y`	Vector of the binary time series.
`Exp_X`	Vector or Matrix of explanatory time series.
`Intercept`	Boolean value: TRUE for an estimation with intercept, and FALSE otherwise.
`Nb_Id`	Number of individuals studied for a panel approach. Nb_Id=1 in the univariate case.
`Lag`	Number of lags used for the estimation.
`type_model`	Model number: 1, 2, 3 or 4.

-> 1 for the static model:

$$P_{t-1}(Y_t) = F(\pi_t) = F(\alpha + \beta' X_t)$$

-> 2 for the dynamic model with lag binary variable:

$$P_{t-1}(Y_t) = F(\pi_t) = F(\alpha + \beta' X_t + \gamma Y_{t-l})$$

-> 3 for the dynamic model with lag index variable:

$$P_{t-1}(Y_t) = F(\pi_t) = F(\alpha + \beta' X_t + \eta \pi_{t-l})$$

-> 4 for the dynamic model with both lag binary variable and lag index variable:

$$P_{t-1}(Y_t) = F(\pi_t) = F(\alpha + \beta' X_t + \eta \pi_{t-l} + \gamma Y_{t-l})$$

Value

A list with:

Estimation	a dataframe containing the coefficients of the logitic estimation, the Standard Error for each coefficient, the Z-score and the associated critical probability
AIC	a numeric vector containing the Akaike information criterion
BIC	a numeric vector containing the Bayesian information criterion
R2	a numeric vector containing the Pseudo R Square
LogLik	a numeric vector containing the Log likelihood value of the estimation
VCM	a numeric matrix of the Variance Covariance of the estimation

Note

For the panel estimation, data must be stacked one after the other for each country or for each individual.

Author(s)

Jean-Baptiste Hasse and Quentin Lajaunie

References

Candelon, Bertrand, Elena-Ivona Dumitrescu, and Christophe Hurlin. "Currency crisis early warning systems: Why they should be dynamic." International Journal of Forecasting 30.4 (2014): 1016-1029.

Hasse, Jean-Baptiste, Lajaunie Quentin. "Does the Yield Curve Signal Recessions? New Evidence from an International Panel Data Analysis." (2020)

Kauppi, Heikki, and Pentti Saikkonen. "Predicting US recessions with dynamic binary response models." The Review of Economics and Statistics 90.4 (2008): 777-791.

Naceur, Sami Ben, Bertrand Candelon, and Quentin Lajaunie. "Taming financial development to reduce crises." Emerging Markets Review 40 (2019): 100618.

Examples

```
# First Example: univariate analysis of the predictive power of the yield spread

# NOT RUN {


# Import data
data("data_USA")

# Data process
Var_Y <- as.vector(data_USA$NBER)
Var_X <- as.vector(data_USA$Spread)
```

```
# Estimate the logit regression
results <- Logistic_Estimation(Dicho_Y = Var_Y, Exp_X = Var_X, Intercept = TRUE,
                      Nb_Id = 1, Lag = 1, type_model = 4)

# print results
results


# }


# Second Example: panel analysis of the predictive power of the yield spread

# NOT RUN {


# Import data
data("data_panel")

# Data process
Var_Y <- as.vector(data_panel$OCDE)
Var_X <- as.vector(data_panel$Spread)

# Estimate the logit regression
results <- Logistic_Estimation(Dicho_Y = Var_Y, Exp_X = Var_X, Intercept = TRUE,
                        Nb_Id = 13, Lag = 1, type_model = 4)

# print results
results


# }
```

Matrix_lag *Matrix Lag - data processing*

Description

Compute a lagged version of a time series, shifting the time base back by a given number of observations defined by the user. The user must enter three parameters for this function: the matrix, the number of lags, and of boolean variable calls 'beginning'. If 'beginning'=TRUE, then the lag will be applied at the beginning of the matrix whereas if 'beginning'=FALSE, then the lag will be applied at the end of the matrix.

Usage

```
Matrix_lag(Matrix_target, Nb_lag, beginning)
```

Arguments

Matrix_target	Initial Matrix
Nb_lag	Number of lag
beginning	Boolean variable. If 'place'=TRUE, the lag is applied at the beginning of the matrix. If 'place'=FALSE, the lag is applied at the end of the matrix.

Value

A numeric Matrix.

Examples

```
# Initialize the following matrix
Matrix_example <- matrix(data=(1:10), nrow=5, ncol=2)

# Use Matrix_lag
new_matrix <- Matrix_lag(Matrix_target = Matrix_example, Nb_lag = 2, beginning = TRUE)

new_matrix

# Results:
#> new_matrix
#     [,1] [,2]
#[1,]    2    7
#[2,]    3    8
#[3,]    4    9
#[4,]    5   10
```

Vector_lag — *Vector lag - data processing*

Description

Compute a lagged version of a time series, shifting the time base back by a given number of observations defined by the user. The user must enter three parameters for this function: the vector, the number of lags, and a boolean variable named 'beginning'. If 'beginning'=TRUE, then the lag will be applied at the beginning of the vector whereas if 'beginning'=FALSE, then the lag will be applied at the end of the vector.

Usage

```
Vector_lag(Vector_target, Nb_lag, beginning)
```

Arguments

`Vector_target`	Initial vector
`Nb_lag`	Number of lag
`beginning`	Boolean variable. If 'beginning'=TRUE, the lag is applied at the beginning of the vector. If 'beginning'=FALSE, the lag is applied at the end of the vector.

Value

A numeric Vector.

Examples

```
# Initialize the following vector
vector_example <- as.vector(1:10)

# Use Vector_lag
new_vector <- Vector_lag(Vector_target = vector_example, Nb_lag = 2, beginning = TRUE)

new_vector
# Results:
#> new_vector
#[1]  3  4  5  6  7  8  9 10
```

Index