Package ‘RTransProb’

July 17, 2018

Type Package

Title Analyze and Forecast Credit Migrations

Version 0.3.3

Date 2018-07-16

Author Ab NDiaye

Maintainer Ab NDiaye <ab.ndiaye@illumirisk.com>

Description A set of functions used to automate commonly used methods in credit risk to estimate migration (transition) matrices. The package includes multiple methods for bootstrapping default rates and forecasting/stress testing credit exposures migrations, via Econometric and Machine Learning approaches. More information can be found at <https://analyticsrusers.blog>.

LazyData TRUE

RoxygenNote 6.0.1

Depends R (>= 2.2.0)

Imports chron, caret, e1071, expm, MASS, Matrix, matrixStats, neuralnet, nnet, pracma, tcltk, zoo, Rcpp (>= 0.12.11)

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-07-17 17:10:11 UTC

R topics documented:

`cfdates` *Create Date Squence*

Description

This function takes two dates and returns a sequence of dates using the interval.

Usage

```
cfdates(sdates,edates,snapshots)
```

Arguments

`sdates`	start date in Matlab datenum format.
`edates`	end date in Matlab datenum format.
`snapshots`	Integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, 12, 54, and 356. For example, 1 = one snapshot per year

Value

a list containing a date sequence.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
# Convert a date string to Matlab datenum format.
sdates <-  POSIXTomatlab(as.POSIXlt(as.Date("2000-01-01")))
edates <-  POSIXTomatlab(as.POSIXlt(as.Date("2002-01-01")))

cfdates(sdates, edates, 1)
```

---

cohort.CI   *Bootstrapped confidence intervals - Cohort*

Description

estimate confidence intervals for the TTC transition probabilities using a bootstrapping procedure for cohort method.

Usage

```
cohort.CI(transMatrix,initCount,sim)
```

Arguments

transMatrix	list containing average transitions matrices for each time-step.
initCount	list containing average start vector counts for each time-step.
sim	number of simulations.

Details

The general idea of bootstrapping is to use resampling methods to estimate features of the sampling distribution of an estimator, especially in situations where 'asymptotic approximations' may provide poor results. In the case of a *parametric* bootstrap method one samples from the estimated distribution derived using maximum likelihood estimation. In summary,

1. Estimate the distribution from the observed sample using maximum likelihood
2. Draw samples from the estimated distribution
3. Calculate the parameter of interest from each of the samples
4. Construct an empirical distribution for the parameter of interest

5. Select percentiles from the empirical distribution

One can contrast this method with a *nonparametric bootstrap* in which one samples with replacement from the empirical cumulative distribution function of the observed sample. Since there are grades with zero observed default rates, resampling directly from the observed data will not produce meaningful confidence intervals in for credit transition matrices where historically there are a limited number of defaults in higher credit quality buckets.

The parametric bootstrap method modeled here generates 12-month paths for each obligor represented in the portfolio and estimates the 12 monthly transition matrices to get a single observation. Annual paths (histories) are simulated using the estimated monthly transition matrices. A consequence of this approach, is that it is computationally intensive, but once the bootstrapped distributions of the PD values have been completed, it is simple to identify the percentiles of interest for calculation of confidence intervals

Value

Returns the default probabilites values for the *n* ratings at the 2.5, 5, 25, 50, 75, 95, 97.5 percentiles.

Author(s)

Abdoulaye (Ab) N'Diaye

References

Hanson, S. and Schuermann, T. 2005 Confidence Intervals for Probabilities of Default, Federal Reserve Bank of New York

Jafry, Y. and Schuermann, T. 2003 Metrics for Comparing Credit Migration Matrices, Wharton Financial Institutions Working Paper 03-08.

Loffler, G., P. N. Posch. 2007 Credit Risk Modeling Using Excel and VBA. West Sussex, England, Wiley Finance

Examples

```
## Not run:

#Set parameters to generate PIT transition matrices
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
method       <- "cohort"
snapshots <- 4
interval <-  .25
Example<-getPIT(data,startDate, endDate,method, snapshots, interval)

lstInit <- Example$lstInitVec[lapply(Example$lstInitVec,length)>0]
lstCnt <- Example$lstCntMat[lapply(Example$lstCntMat,length)>0]
ExampleTTC <- cohort.TTC(lstCnt,lstInit)

#use $ATMP from the cohort.TTC() as the input into the cohort.CI() function
transMatrix    <- ExampleTTC$ATMP
```

```
initCount      <- ExampleTTC$ACP[[1]][,1]
sim            <- 1000
tolerance_Cohort <-cohort.CI(transMatrix,initCount,sim)



Example 2:
#Set parameters to generate PIT transition matrices
startDate  <- "1997-01-01"
endDate    <- "2002-01-01"
method       <- "cohort"
snapshots <- 12
interval <-  1
Example<-getPIT(data,startDate, endDate,method, snapshots, interval)

lstInit <- Example$lstInitVec[lapply(Example$lstInitVec,length)>0]
lstCnt <- Example$lstCntMat[lapply(Example$lstCntMat,length)>0]
ExampleTTC <- cohort.TTC(lstCnt,lstInit)

#use $ATMP from the cohort.TTC() as the input into the cohort.CI() function
transMatrix    <- ExampleTTC$ATMP
initCount      <- ExampleTTC$ACP[[1]][,1]
sim            <- 1000
tolerance_Cohort <-cohort.CI(transMatrix,initCount,sim)

## End(Not run)
```

`cohort.TTC` *Cohort - Data Weighting and "TTC" Calculation*

Description

Calculate *Through-the-Cycle* transition matrices using the *cohort method* transitions.

Usage

```
cohort.TTC(transCount, initCount)
```

Arguments

`transCount`	transitions counts for each time-step
`initCount`	start vector counts for each time-step

Details

Many credit risk models require a *long-run average* (Through-the-Cycle) PD estimate. This has been interpreted as meaning the data from multiple years should be combined and the method capable of supporting some form of weighting of samples.

The three methods of weighting considered for data generated via the cohort method are:

1. Scale the number of transitions and firm counts using the a single year count to preserve dynamics, then average transitions and firms counts separately
2. Estimate the single-year quantities (estimate with transition matrices for each time-step), then average across years
3. Average annual transition matrices

The Markov property allows for direct weighting as each time-step can be regarded as distinct(independence).

Value

SAT	*Scaled Average Transitions* - compute a TTC transition matrix by first scaling and weighting the counts (start vector counts and transition counts) then calculate the transition matrices for each time-step, and finally averaging over all available time-steps. e.g., average January matrices, then February matrices or average Q1, then Q2 ...then obtain the average of the transition matrices
SAPT	*Scaled Average Periodic Transitions* - compute a TTC transition matrix by weighting the transition percentages for each time-step (calculate the transition matrices for each time-step then weigh the percentages, and finally averaging over all available time-steps. e.g., average January matrices, then February matrices or average Q1, then Q2 ...then obtain the average of the transition matrices
USAT	*Unscaled Average Transitions* - compute a TTC transition matrix by first obtaining unscaled transition matrices for each time-step then averaging over all available time-steps
ATMP	*averageTransMatByPeriod* - returns the weighted the transition percentages for each time-step (calculate the transition matrices for each time-step then weigh the percentages
ATP	*averageTransByPeriod* - returns the scaled transitions for each time-step
ACP	*averageCountByPeriod* - returns the scaled start vector counts for each time-step

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:

#Set parameters
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
method       <- "cohort"
snapshots <- 4
interval <-  .25
Example<-getPIT(data,startDate, endDate,method, snapshots, interval)

lstInit <- Example$lstInitVec[lapply(Example$lstInitVec,length)>0]
lstCnt <- Example$lstCntMat[lapply(Example$lstCntMat,length)>0]
ExampleTTC <- cohort.TTC(lstCnt,lstInit)
```

```
## End(Not run)
```

`data` *data*

Description

data

`dataTM` *dataTM*

Description

dataTM

`duration.CI` *Bootstrapped confidence intervals - Duration*

Description

estimate confidence intervals for the transition probabilities using a bootstrapping procedure for duration method

Usage

```
duration.CI(genMat,portWgts,nHorizon,sim)
```

Arguments

`genMat`	generator matrix
`portWgts`	list containing weights of each rating class
`nHorizon`	horizon
`sim`	number of simulations

Details

The general idea of bootstrapping is to use resampling methods to estimate features of the sampling distribution of an estimator, especially in situations where asymptotic approximations may provide poor results. In the case of a parametric bootstrap method one samples from the estimated distribution derived using maximum likelihood estimation. In summary,

1. Estimate the distribution from the observed sample using maximum likelihood
2. Draw samples from the estimated distribution
3. Calculate the parameter of interest from each of the samples
4. Construct an empirical distribution for the parameter of interest
5. Select percentiles from the empirical distribution

One can contrast this method with a *nonparametric bootstrap* in which one samples with replacement from the empirical cumulative distribution function of the observed sample.

A parametric bootstrapping method is employed for the time-homogeneous continuous-time Markov model. The elements of the infinitesimal generator matrix, provide most of the information one needs to perform the parametric bootstrap. The outline of the bootstrapping is provided below.

For each obligor in a given assigned credit grade:

1. Start by drawing a (sojourn) time from the exponential distribution with parameter, $-\hat{\lambda}_{kk}$
2. If the time is greater than or equal to the time left to horizon then stop
3. If the time is less than the time left to horizon
   - Draw from the multinomial distribution associated with the possible transition states using the vector of probabilities
   - Determine the state to which the obligor moves, for example, i
   - Repeat the process in 1. now using the diagonal element, $-\hat{\lambda}_{ii}$
   - Continue until the sampled time exceeds the time to horizon

Value

Returns the default probabilites values for the n ratings at the 2.5, 5, 25, 50, 75, 95, 97.5 percentiles.

Author(s)

Abdoulaye (Ab) N'Diaye

References

Hanson, S. and Schuermann, T. 2005 Confidence Intervals for Probabilities of Default, Federal Reserve Bank of New York

Jafry, Y. and Schuermann, T. 2003 Metrics for Comparing Credit Migration Matrices, Wharton Financial Institutions Working Paper 03-08.

Loffler, G., P. N. Posch. 2007 Credit Risk Modeling Using Excel and VBA. West Sussex, England, Wiley Finance

Trueck, Stefan, (February 16, 2009) Simulating Dependent Credit Migrations. Available at SSRN: https://ssrn.com/abstract=1344897

Examples

```
## Not run:
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
method       <- "duration"
snapshots <- 4
interval  <-  0
Example1  <-getPIT(data,startDate, endDate,method, snapshots, interval)

lstInit <- Example1$lstInitVec[lapply(Example1$lstInitVec,length)>0]
lstCnt <-  Example1$lstCntMat[lapply(Example1$lstCntMat,length)>0]
ExampleTTC1<-duration.TTC(Example1$lstCntMat,Example1$lstInitVec)


genMat   <- ExampleTTC1$WGM
portWgts <- ExampleTTC1$SWFY[,1]
nHorizon <- length(ExampleTTC1$UUPTM[[1]])
sim      <- 100

tolerance_Duration <- duration.CI(genMat,portWgts,nHorizon,sim)

## End(Not run)
```

---

duration.TTC — *Duration - Data Weighting and "TTC" Calculation*

Description

Calculating *Through-the-Cycle* generator matrix and transition counts using *duration method*

Usage

```
duration.TTC(lstCnt,lstFirmYears)
```

Arguments

`lstCnt`	off-diagonal transition counts (matrix) for each time-step
`lstFirmYears`	firm years each time-step

Details

Given data representing *x* off-diagonal transition counts for each time-step, this function combines those data to obtain average counts for each time-step, in such a way as to preserve the information while implementing a weighting scheme that would allow for the weighting of the historical experiences.

Let $T(m, y)$ and $F(m, y)$ represent the off-diagonal transition matrix and 'firm-years' vector, for month = m and year = y, respectively. Then,

$$T(m, y) = \{T_{ij}(m, y)\}_{i,j=1,...,K}$$

$$F(m, y) = \{F_i(m, y)\}_{i=1,...,K}$$

Many credit risk models require a *long-run average* PD estimate. This has been interpreted as meaning the data from multiple years should be combined and in a method capable of supporting some form of weighting of samples. The three methods of weighting considered for data generated via the *duration method* are:

1. Scale the number of transitions and firm counts/years using the a single year count to preserve dynamics, then average transitions and firms counts/years separately to create a generator matrix.
2. Estimate the single-year quantities *(generator matrices for each time-step)*, then average across years
3. Average transition matrices from each time-step

The Markov property allows for direct weighting as each year can be regarded as distinct.

Value

CLW	*Count Level Weighting* - Construct TTC transition matrix from aggregate scaled and weighted counts data (transitions and 'firm-years').
PTMLW	*Periodic Transition Matrix Level Weighting* - Construct TTC transition matrix using the average of the weighted transition matrices from each time-step (Scaling is performed at the transition matrix level for each time-step).
PGMLW	*Periodic Generator Matrix Level Weighting* - Construct TTC transition matrix using the average of the weighted Generator matices from each time-step (Scaling is performed at the generator matrix level for each time-step).
UUPTM	*Unscaled and UnWeighted Periodic Transition Matrices* - Construction of unscaled and unweighted periodic transition matrices from unscaled and unweighted generator matrices for each time-step .
WGM	*Weighted Generator Matrix* - Average generator matrix from each time-step.
SWT	*Scaled and Weighted Transitions* - aggregate scaled and weighted transitions
SWFY	*Scaled and Weighted Firm Years* - aggregate scaled and weighted firm years

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:

#Set parameters
startDate  <- "2000-01-01"
```

```
endDate    <- "2005-01-01"
method       <- "duration"
snapshots <- 4
interval <-  0
Example1<-getPIT(data,startDate, endDate,method, snapshots, interval)

ExampleTTC1<-duration.TTC(Example1$lstCntMat,Example1$lstInitVec)


## End(Not run)
```

`expandTransData` *Reshape Data to 'Wide' Data Format*

Description

This function reshapes time series vector of transition counts with elements of both 'wide' and 'long' data Formats. Each non-Zero weighted row is expanded to show one row per record.

Usage

```
expandTransData(transData, wgtname)
```

Arguments

`transData`	dataframe containing the time series vector of transition counts.
`wgtname`	weight.

Value

The output a dataframe of transition data in 'wide' format.

Author(s)

Abdoulaye (Ab) N'Diaye

`fromThresholds` *Convert credit quality thresholds to probabilities.*

Description

Use this function to transform credit quality thresholds into transition probabilities.

Usage

```
fromThresholds(thresh)
```

Arguments

`thresh` *m-by-m* matrix of credit quality thresholds. In each row, the first element must be *Inf* and the entries must satisfy the following monotonicity condition:

Value

Returns a *m-by-m* matrix with transition probabilities, in percent.

Author(s)

Abdoulaye (Ab) N'Diaye

References

MathWorld.com (2011). Matlab Central http://www.mathworks.com/matlabcentral/. Mathtools.net http://www.mathtools.net/.

Examples

```
## Not run:
rc <- c("AAA", "AA", "A", "BBB", "BB", "B", "CCC", "D")
t<- matrix(c(Inf,-1.3656,-2.1806,-3.0781,-3.5482,-4.1612,-4.2591,-4.8399,
             Inf, 1.5712,-1.5217,-2.3028,-2.6872,-3.5256,-3.7324,-4.1972,
             Inf, 2.6895, 1.3806,-1.2901,-2.3422,-2.8928,-3.0063,-3.7861,
             Inf, 3.1004, 2.5623, 1.4479,-1.5211,-2.1407,-2.434,-3.2814,
             Inf, 3.4339, 2.6156, 2.4434, 1.4561,-1.4573,-1.9742,-2.4668,
             Inf, 2.5852, 2.5586, 2.4218, 2.268, 1.6737,-1.6194,-2.252,
             Inf, 3.6953, 3.6362, 3.3406, 2.5019, 2.2394, 1.6263,-1.3853,
             Inf, Inf,   Inf,   Inf,   Inf,   Inf,   Inf,   Inf
), 8,8, dimnames = list(rc,rc), byrow=TRUE)


transmatrix <- fromThresholds(t)

## End(Not run)
```

`getPIT` *Estimation of credit transition probabilities*

Description

This function is used to estimate Point-in-Time transition probabilities and counts given historical credit data (a.k.a., credit migration data).

Usage

```
getPIT(data, startDate, endDate, method, snapshots, interval)
```

Arguments

`data`	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
`startDate`	start date of the estimation time window, in string or numeric format. The default start date is the earliest date in 'data'.
`endDate`	end date of the estimation time window, in string or numeric format. The default end date is the latest date in 'data'. The end date cannot be a date before the start date.
`method`	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
`snapshots`	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
`interval`	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated.*

Value

Returns the following objects:

sampleTotals	a list containing the following count components:
`totalsVec`	A vector of size *1-by-nRatings.* For 'duration' calculations, the vector stores the total time spent on *rating i.* For 'cohort' calculations, the vector stores the initial counts (start vector) in *rating i.*
`totalsMat`	A matrix of size *nRatings-by-nRatings.* For 'duration' calculations, the matrix contains the total transitions observed out of *rating i* into *rating j* (all the diagonal elements are zero). For 'cohort' calculations, the matrix contains the total transitions observed from *rating i* to *rating j.*
`algorithm`	A character vector with values 'duration' or 'cohort'.
transMat	Matrix of transition probabilities in percent. The size of the transition matrix is *nRatings-by-nRatings.*
genMat	Generator Matrix. *use only with duration method*

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:
snapshots <- 4     #This uses quarterly snapshots
interval <- .25     #This gives quarterly transition matrix
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example<-getPIT(data,startDate, endDate,'cohort', snapshots, interval)

## End(Not run)
```

histData *histData*

Description

histData

histData.normz *histData.normz*

Description

histData.normz

matlabToPOSIX *Convert a numeric MATLAB datenum to R POSIXt time values*

Description

This function is used to convert a numeric MATLAB datenum

Usage

```
matlabToPOSIX(gTime, timez)
```

Arguments

`gTime`	a MATLAB datenum value
`timez`	time zone, default is "UTC"

Value

R POSIXt time values.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
matlabToPOSIX(734139)
```

`POSIXTomatlab` *Convert between MATLAB datenum and R POSIXt*

Description

This function is used to convert between MATLAB datenum values and R POSIXt time values.

Usage

```
POSIXTomatlab(gTime)
```

Arguments

`gTime`	a POSIXct or POSIXlt date value

Value

MATLAB datenum value.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
POSIXTomatlab(as.POSIXlt(as.Date("2010-01-01")))
```

`predData_ann_Adverse` *predData_ann_Adverse*

Description

predData_ann_Adverse

`predData_ann_Baseline` *predData_ann_Baseline*

Description

predData_ann_Baseline

predData_ann_SeverelyAdverse *predData_ann_SeverelyAdverse*

Description

predData_ann_SeverelyAdverse

predData_lda_Adverse *predData_lda_Adverse*

Description

predData_lda_Adverse

predData_lda_Baseline *predData_lda_Baseline*

Description

predData_lda_Baseline

predData_lda_SeverelyAdverse *predData_lda_SeverelyAdverse*

Description

predData_lda_SeverelyAdverse

predData_mnl_Adverse *predData_mnl_Adverse*

Description

predData_mnl_Adverse

predData_mnl_Baseline *predData_mnl_Baseline*

Description

predData_mnl_Baseline

predData_mnl_NoVariables *predData_mnl_NoVariables*

Description

predData_mnl_NoVariables

predData_mnl_SeverelyAdverse *predData_mnl_SeverelyAdverse*

Description

predData_mnl_SeverelyAdverse

predData_svm_Adverse *predData_svm_Adverse*

Description

predData_svm_Adverse

predData_svm_Baseline *predData_svm_Baseline*

Description

predData_svm_Baseline

predData_svm_SeverelyAdverse *predData_svm_SeverelyAdverse*

Description

predData_svm_SeverelyAdverse

toThresholds *Convert probabilities to credit quality thresholds.*

Description

Use this function to transform transition probabilities into credit quality thresholds.

Usage

```
toThresholds(trans)
```

Arguments

trans	a *m-by-m* matrix with transition probabilities, in percent. Entries cannot be negative and cannot exceed 100, and all rows must sum up to 100.

Value

Returns a *m-by-m* matrix of credit quality thresholds

Author(s)

Abdoulaye (Ab) N'Diaye

References

MathWorld.com (2011). Matlab Central http://www.mathworks.com/matlabcentral/. Mathtools.net http://www.mathtools.net/.

Examples

```
## Not run:
rc <- c("AAA", "AA", "A", "BBB", "BB", "B", "CCC", "D")
t<- matrix(c(91.3969, 7.1423, 1.3566, 0.0848, 0.0178, 0.0006, 0.0010, 0.0001,
              5.8072, 87.7881, 5.3402, 0.7040, 0.3391, 0.0116, 0.0081, 0.0014,
              0.3578, 8.0124, 81.7798, 8.8916, 0.7675, 0.0587, 0.1246, 0.0077,
              0.0966, 0.4232, 6.8627, 86.2059, 4.7967, 0.8681, 0.6951, 0.0516,
              0.0297, 0.4156, 0.2821, 6.5406, 85.4804, 4.8337, 1.7363, 0.6815,
```

```
              0.4866, 0.0389, 0.2467, 0.3945, 3.5428, 90.0229, 4.0516, 1.2161,
              0.0110, 0.0029, 0.0280, 0.5759, 0.6389, 3.9374, 86.5074, 8.2987,
              0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 100.0000
), 8,8, dimnames = list(rc,rc), byrow=TRUE)


thresholds<-toThresholds(t)

## End(Not run)
```

`transForecast` *Forecast - using Credit Cycle*

Description

This model implements the One-Parameter Representation method developed by Forest, Belkin and Suchower.

Usage

```
transForecast(genMat, isGenerator, creditIndex)
```

Arguments

`genMat`	Generator Matrix
`isGenerator`	a variable to determine if the input matrix is a generator matrix or a proper transition matrix. The default value is 'No'.
`creditIndex`	a credit index. The model relies on the assumption that credit migration matrices are driven by a single parameter Z, which depicts the average financial health of corporate institutions (credit index). The degree of 'shift' corresponds to a simple change in the transition probabilities from an average average matrix.

Details

The Vasicek (1987) Single Factor moodel $A_i = \sqrt{\rho_i} Z + \sqrt{1-\rho_i}\epsilon$ presents a framework which Forest, Belkin and Suchower (1998) used to developed the One-Parameter Representation method. In that model, migration behaviors are described standard normal variables instead of transition probabilities without the loss of information. The transition through probabilites are transformed to thresholds where the upper and lower bounds of the threshold values together represent bins. Therefore, when a random variable falls within a particular bin that signifies a transition to the corresponding transition rating bucket.

The advantage of representing transitions probabilities in terms of the threshold framework is that we can now use the standard normal density curve to understand the behavior rating transitions. The area under a standard normal curve between the lower and upper bounds of a thresholds for a particular bin is the transition probability. Therefore in the context of economic conditions, the shifting of curves (to the left or the right) under static thresholds, informs us about the behavior of transitions matrices during benign and stressed periods.

To the extent that we can represent economic conditions with a single variable, we can 'shift' the average transition matrix by this amount to generate a forecast of the transition matrix.

See Forest, Belkin and Suchower (1998) for a more detailed discussion

Value

The output consists of a forecasted (shifted) transition matrix.

Author(s)

Abdoulaye (Ab) N'Diaye

Loffler, G., P. N. Posch. 2007 Credit Risk Modeling Using Excel and VBA. West Sussex, England, Wiley Finance

L. R. Forest, B. Belkin, and S. J. Suchower, 1998 A One-Parameter Representation of Credit Risk and Transition Matrices, CreditMetrics Monitor. Q3

Vasicek, O., 1987 Probability of loss on a loan portfolio. Working paper, KMV.

Examples

```
#Use the function 'TransitionProb' to estimate an annualized transition matrix which will
#then be used along with the appropriate creditIndex to forecast future period migration
#effects.

## Not run:
snapshots <- 4    #This uses quarterly transition matrices
interval <- 1    #This gives a 1 year transition matrix
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example9<-TransitionProb(data,startDate, endDate,'duration', period, snapshots, interval)
Example9.1 <- Example9$genMat
creditIndex <- -0.25


Example10 <- transForecast(Example9.1, isGenerator, creditIndex)

## End(Not run)
```

`transForecast_ann` *Forecast - using Artificial Neural Networks*

Description

This model implements a forecasting method using Artificial Neural Networks.

Usage

```
transForecast_ann(data, histData, predData_ann, startDate, endDate,
                method, interval, snapshots, defind, depVar, indVars,  ratingCat,
                    pct, hiddenlayers,activation,stepMax,rept, calibration)
```

Arguments

`data`	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
`histData`	historical macroeconomic,financial and non-financial data.
`predData_ann`	forecasting data.
`startDate`	start date of the estimation time window, in string or numeric format.
`endDate`	end date of the estimation time window, in string or numeric format.
`method`	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
`interval`	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated*.
`snapshots`	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
`defind`	Default Indicator
`depVar`	dependent variable, as a string.
`indVars`	list containing the independent variables.
`ratingCat`	list containing the unique rating categories
`pct`	percent of data used in training dataset.
`hiddenlayers`	a vector of integers specifying the number of hidden neurons (vertices) in each layer.
`activation`	activation function. strings, 'logistic' and 'tanh' are possible for the logistic function and tangent hyperbolicus
`stepMax`	the maximum steps for the training of the neural network. Reaching this maximum leads to a stop of the neural network's training process.
`rept`	the number of repetitions for the neural network's training.
`calibration`	determines if code uses the caret package to find optimal parameter. 'Yes' and 'No'

Value

The output consists of a forecasted transition matrix using ANN.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:
library(dplyr)
library(plyr)
library(Matrix)


for (i in c(24,25,26)) {
 tic()
 data <- data

 histData <- histData.normz

 predData_ann2 <- predData_ann_Baseline
 predData_ann2 <- subset(
   predData_ann2,
   X == i,
   select = c(Market.Volatility.Index..Level..normz

   )
 )

 indVars   = c("Market.Volatility.Index..Level..normz"

 )


 startDate = "1991-08-16"
 endDate   = "2007-08-16"


 depVar <- c("end_rating")

 pct <- 1
 wgt <-  "mCount"
 ratingCat <- c("A", "B", "C", "D", "E", "F", "G")
 defind    <- "G"


 ratingCat <- as.numeric(factor(
   ratingCat,
   levels = c('A', 'B', 'C', 'D', 'E', 'F', 'G'),
   labels = c(1, 2, 3, 4, 5, 6, 7)
 ))

 defind <- as.numeric(factor(
   defind,
   levels = c('A', 'B', 'C', 'D', 'E', 'F', 'G'),
   labels = c(1, 2, 3, 4, 5, 6, 7)
 ))

 method    = "cohort"
```

```
  snapshots = 1
  interval  = 1
  hiddenlayers = c(1)
  activation = "logistic"
  stepMax = 1e9                #increase to make sure the DNN converges
  calibration = "FALSE"
  rept = 1

  ann_TM <-
    transForecast_ann(
      data,
      histData,
      predData_ann2,
      startDate,
      endDate,
      method,
      interval,
      snapshots,
      defind,
      depVar,
      indVars,
      ratingCat,
      pct,
      hiddenlayers,
      activation,
      stepMax,
      rept,
      calibration
    )
  print(ann_TM)

  toc()
}

## End(Not run)
```

---

transForecast_lda *Forecast - using Linear Discriminant Analysis*

Description

This model implements a forecasting method using Linear Discriminant Analysis.

Usage

```
transForecast_lda(data, histData, predData_lda, startDate, endDate, method,
                      interval, snapshots, defind, depVar, indVars, pct, ratingCat)
```

Arguments

data	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
histData	historical macroeconomic,financial and non-financial data.
predData_lda	forecasting data.
startDate	start date of the estimation time window, in string or numeric format.
endDate	end date of the estimation time window, in string or numeric format.
method	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
interval	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated.*
snapshots	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
defind	Default Indicator
depVar	dependent variable, as a string.
indVars	list containing the independent variables.
pct	percent of data used in training dataset.
ratingCat	list containing the unique rating caetgories

Value

The output consists of a forecasted transition matrix.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:

library(dplyr)
library(plyr)
library(Matrix)


for (i in c(24, 25, 26)) {
  data <- data

  histData <- histData.normz


  predData_lda2 <- predData_lda_Baseline
  predData_lda2 <- subset(
```

```
      predData_lda2,
      X == i,
      select = c(Market.Volatility.Index..Level..normz

      )
    )

    indVars   = c("Market.Volatility.Index..Level..normz"

    )



    startDate = "1991-08-16"
    endDate   = "2007-08-16"


    depVar <- c("end_rating")
    pct <- 1
    wgt <-  "mCount"
    ratingCat <- c("A", "B", "C", "D", "E", "F", "G")
    defind    <- "G"
    method    = "cohort"
    snapshots = 1
    interval  = 1


    lda_TM <-
      transForecast_lda(
        data,
        histData,
        predData_lda2,
        startDate,
        endDate,
        method,
        interval,
        snapshots,
        defind,
        depVar,
        indVars,
        pct,
        ratingCat
      )
    print(lda_TM)

  }


  ## End(Not run)
```

`transForecast_mnl` *Forecast - using Multinomial Logistic Regression*

Description

This model implements a forecasting method using multinomial logistic regression (also known as Softmax Regression in machine learning parlance).

Usage

```
transForecast_mnl(data, histData, predData_mnl, startDate, endDate, method,
                  interval,snapshots, defind,ref, depVar, indVars, ratingCat, wgt)
```

Arguments

`data`	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
`histData`	historical macroeconomic,financial and non-financial data.
`predData_mnl`	forecasting data.
`startDate`	start date of the estimation time window, in string or numeric format.
`endDate`	end date of the estimation time window, in string or numeric format.
`method`	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
`interval`	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated*.
`snapshots`	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
`defind`	Default Indicator
`ref`	base or reference category for the dependent variable.
`depVar`	dependent variable, as a string.
`indVars`	list containing the independent variables
`ratingCat`	list containing the unique rating categories
`wgt`	weights

Details

Multinomial logistic regression is a simple extension of binary logistic regression that allows for more than two categories of the dependent or outcome variable. Whereas, a binary logistic regression model compares one dichotomy, the multinomial logistic regression model compares a number of dichotomies. Like binary logistic regression, multinomial logistic regression uses maximum likelihood estimation to evaluate the probability of categorical membership.

Assume there are 1,2,3 ...K groups in a dataset, and group 1 is the one chosen as the reference category. The logistic model states that the probability of falling into group j given the set of predictor values x is given by the general expression

$$P(y = k|X) = \frac{exp(X\beta_k)}{1 + \sum_{j=2}^{N} exp(X\beta_j)}$$

Value

The output consists of a forecasted transition matrix.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:

library(dplyr)
library(plyr)
library(Matrix)




for (i in c(24, 25, 26)) {
  data <- data

  attach(data)
  data2 <- data[order(ID, Date),]
  detach(data)

  data <- data2
  rm(data2)


  histData <- histData


  predData_mnl2 <- predData_mnl_Baseline
  predData_mnl2 <- subset(
    predData_mnl,
    X == i,
    select = c(Market.Volatility.Index..Level.,
               D_B,
               D_C,
               D_D,
               D_E,
               D_F,
               D_G
    )
  )

  indVars   = c("Market.Volatility.Index..Level."

  )


  startDate = "1991-08-16"
```

```
  endDate   = "2007-08-16"
  method    = "cohort"
  snapshots = 1
  interval  = 1
  ref       = 'A'
  depVar    = c("end_rating")
  ratingCat = c("A", "B", "C", "D", "E", "F", "G", "N")
  defind    = "N"
  wgt       = "mCount"



  transForecast_mnl_out <-
    transForecast_mnl(
      data,
      histData,
      predData_mnl2,
      startDate,
      endDate,
      method,
      interval,
      snapshots,
      defind,
      ref,
      depVar,
      indVars,
      ratingCat,
      wgt
    )
  output <- transForecast_mnl_out$mnl_Predict
  print(output)

}


## End(Not run)
```

`transForecast_svm` *Forecast - using Support Vector Machines*

Description

This model implements a forecasting method using Support Vector Machines.

Usage

```
transForecast_svm(data, histData, predData_svm, startDate, endDate,
                  method, interval, snapshots, defind, depVar, indVars,  ratingCat,
                  pct, tuning, kernelType, cost, cost.weights, gamma, gamma.weights)
```

Arguments

`data`	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
`histData`	historical macroeconomic,financial and non-financial data.
`predData_svm`	forecasting data.
`startDate`	start date of the estimation time window, in string or numeric format.
`endDate`	end date of the estimation time window, in string or numeric format.
`method`	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
`interval`	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated.*
`snapshots`	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
`defind`	Default Indicator
`depVar`	dependent variable, as a string.
`indVars`	list containing the independent variables.
`ratingCat`	list containing the unique rating caetgories
`pct`	percent of data used in training dataset.
`tuning`	perform tuning. If tuning='TRUE' tuning is perform. If tuning='FALSE' tuning is not performed
`kernelType`	the kernel used in training and predicting (see Package e1071 for more detail)
`cost`	cost of constraints violation (default: 1) it is the 'C' constant of the regularization term in the Lagrange formulation.
`cost.weights`	vector containing tuning parameters for cost
`gamma`	parameter needed for all kernels except linear (default: 1/(data dimension))
`gamma.weights`	vector containing tuning parameters for gamma

Value

The output consists of a forecasted transition matrix using SVM.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:
library(dplyr)
library(plyr)
library(Matrix)
```

```
library(tictoc)


for (i in c(24, 25, 26)) {
 print(paste("RUN-",i,sep=""))
 data <- data

 histData <- histData.normz


 predData_svm2 <- predData_svm_Baseline
 predData_svm2 <- subset(
   predData_svm2,
   X == i,
   select = c(Market.Volatility.Index..Level..normz

   )
 )

 indVars   = c("Market.Volatility.Index..Level..normz"

 )


 startDate = "1991-08-16"
 endDate   = "2007-08-16"

 depVar <- c("end_rating")

 pct <- 1
 wgt <-  "mCount"
 ratingCat <- c("A", "B", "C", "D", "E", "F", "G")
 defind    <- "G"
 lstCategoricalVars <- c("end_rating")
 tuning <- "FALSE"
 cost <- 0.01
 gamma <- 0.01
 cost.weights <-  c(0.01, 0.05, 0.1, 0.25, 10, 50, 100)
 gamma.weights <- c(0.01, 0.05, 0.1, 0.25, 10, 50, 100)
 kernelType <- "sigmoid"
 method    = "cohort"
 snapshots = 1
 interval  = 1


 svm_TM <-
   transForecast_svm(
     data,
     histData,
     predData_svm2,
     startDate,
     endDate,
```

```
      method,
      interval,
      snapshots,
      defind,
      depVar,
      indVars,
      ratingCat,
      pct,
      tuning,
      kernelType,
      cost,
      cost.weights,
      gamma,
      gamma.weights
    )
 print(svm_TM)

}

## End(Not run)
```

TransitionProb *Estimation of credit transition probabilities*

Description

This function is used to estimate transition probabilities and counts given historical credit data (a.k.a., credit migration data).

Usage

```
TransitionProb(dataTM, startDate, endDate, method, snapshots, interval)
```

Arguments

dataTM	a table containing historical credit ratings data (i.e., credit migration data). A dataframe of size *nRecords* x 3 where each row contains an ID (column 1), a date (column 2), and a credit rating (column 3); The credit rating is the rating assigned to the corresponding ID on the corresponding date.
startDate	start date of the estimation time window, in string or numeric format. The default start date is the earliest date in 'data'.
endDate	end date of the estimation time window, in string or numeric format. The default end date is the latest date in 'data'. The end date cannot be a date before the start date.
method	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.
snapshots	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.

`interval` the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated.*

Details

The two most commonly used methods to estimate credit transition matrices are the cohort (discrete time) and duration (continuous time) methods.

Cohort Method (Discrete-time Markov Chains) - The method most commlonly used by rating agencies is the cohort method. Let $P_{ij}(\Delta t)$ be the probability of migrating from grade i to j over a specified time period Δt. An estimate of the transition probability of a 1 year horizon where $\Delta t = 1 year$ is thus:

$$P_{ij}(\Delta t) = \frac{N_{ij}}{N_i}$$

where N_i = number of firms in rating category i at the beginning of the horizon, and N_{ij} = the number of firms that migrated to grade j by horizon-end.

It is important to note that any rating change activity which occurs within the period Δt is ignored, thus leading to information loss.

Duration Method (Continuous-time Markov Chains) - A time homogenous continuous-time Markov chain in a sense uses all of the available information and is specified using a (KxK) generator matrix estimated via the maximum likelihood estimator

$$\lambda_{ij} = \frac{N_{ij}(T)}{\int_T^0 Y_i(s)ds}$$

where $Y_i(s)$ is the number of firms in rating class i at time s and $N_{ij}(T)$ is the total number of transitions over the period from i to j, where $i \neq j$.

Value

Returns the following objects:

sampleTotals a list containing the following count components:

`totalsVec` A vector of size *1-by-nRatings.* For 'duration' calculations, the vector stores the total time spent on *rating i.* For 'cohort' calculations, the vector stores the initial counts (start vector) in *rating i.*

`totalsMat` A matrix of size *nRatings-by-nRatings.* For 'duration' calculations, the matrix contains the total transitions observed out of *rating i* into *rating j* (all the diagonal elements are zero). For 'cohort' calculations, the matrix contains the total transitions observed from *rating i* to *rating j.*

`algorithm` A character vector with values 'duration' or 'cohort'.

transMat Matrix of transition probabilities in percent. The size of the transition matrix is *nRatings-by-nRatings.*

genMat Generator Matrix. *use only with duration method*

Author(s)

Abdoulaye (Ab) N'Diaye

References

Jafry, Y. and Schuermann, T. 2003 Metrics for Comparing Credit Migration Matrices, Wharton Financial Institutions Working Paper 03-08.

Lando, D., Skodeberg, T. M. 2002 Analyzing Rating Transitions and Rating Drift with Continuous Observations, Journal of Banking and Finance 26, No. 2-3, 423-444

MathWorld.com (2011). Matlab Central http://www.mathworks.com/matlabcentral/. Mathtools.net http://www.mathtools.net/.

Schuermann, T. and Hanson, S. 2004 Estimating Probabilities of Default, Staff Report No. 190, Federal Reserve Bank of New York,

Examples

```
## Not run:
#Example 1:
#When start date and end date are not specified, the entire dataset is used and the package
#performs TTC calculations. Equally when snapshots and interval are not specified the defaults
#are 1.
snapshots <- 0
interval <- 0
startDate  <- 0
endDate    <- 0
Example1<-TransitionProb(dataTM,startDate,endDate,'cohort', snapshots, interval)


#Example 2:
#using the duration method the time window of interest are specified 2-year period from the
#beginning of 2000 to the beginning of 2002 snapshots and interval are not specified.
snapshots <- 0
interval <- 0
startDate  <- "2000-01-01"
endDate    <- "2002-01-01"
Example2<-TransitionProb(dataTM,startDate, endDate,'duration', snapshots, interval)


#Example 3:
#using the cohort method the time window of interest are specified 5-year period from the
#beginning of 2000 to the beginning of 2005 snapshots and interval are not specified.
snapshots <- 0
interval <- 0
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example3<-TransitionProb(dataTM,startDate, endDate,'cohort', snapshots, interval)


#Example 4:
#assume that the time window of interest is the 5-year period from the beginning of 2000 to
#the beginning of 2005. We want to estimate 1-year transition probabilities using quarterly
#snapshots using cohort method.
snapshots <- 4    #This uses quarterly transition matrices
interval <- 1    #This gives a 1 year transition matrix
```

```
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example4<-TransitionProb(dataTM,startDate, endDate,'cohort', snapshots, interval)


#Example 5:
#assume that the time window of interest is the 5-year period from the beginning of 2000 to
#the beginning of 2005. We want to estimate a 2-year transition probabilities using quarterly
#snapshots using cohort method.
snapshots <- 4     #This uses quarterly transition matrices
interval <- 2     #This gives a 2 years transition matrix
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example5<-TransitionProb(dataTM,startDate, endDate,'cohort', snapshots, interval)


#Example 6:
#assume that the time window of interest is the 2-year period from the beginning of 2000 to
#the beginning of 2005. We want to estimate 1-year transition probabilities using quarterly
#snapshots using duration method.
snapshots <- 4    #This uses quarterly transition matrices
interval <- 1    #This gives a 1 year transition matrix
startDate  <- "2000-01-01"
endDate    <- "2002-01-01"
Example6<-TransitionProb(dataTM,startDate, endDate,'duration', snapshots, interval)


#Example 7:
#assume that the time window of interest is the 5-year period from the beginning of 2000 to
#the beginning of 2005. We want to estimate 1-year transition probabilities using monthly
#snapshots using cohort method.
snapshots <- 12    #This uses monthly transition matrices
interval <- 1    #This gives a 1 year transition matrix
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example7<-TransitionProb(dataTM,startDate, endDate,'cohort', snapshots, interval)


#Example 8:
#assume that the time window of interest is the 5-year period from the beginning of 2000 to
#the beginning of 2005. We want to estimate 1-year transition probabilities using annual
#snapshots using cohort method.
snapshots <- 1    #This uses annual transition matrices
interval <- 1    #This gives a 1 year transition matrix
startDate  <- "2000-01-01"
endDate    <- "2005-01-01"
Example8<-TransitionProb(dataTM,startDate, endDate,'cohort', snapshots, interval)

## End(Not run)
```

`transitionprobbytotals` *estimate of transition probabilities.*

Description

estimation of transition probabilities using a transition counts and start vector.

Usage

```
transitionprobbytotals(idTotCnt,snapshots,interval,method)
```

Arguments

`idTotCnt`	a list structure containing *m-by-m* matrices of transition counts, *1-by-m* vectors start counts, and a string with values 'duration' or 'cohort'.
`snapshots`	integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, or 12. The default value is 1, *i.e., one snapshot per year*. This parameter is only used in the 'cohort' algorithm.
`interval`	the length of the transition interval under consideration, in years. The default value is 1, *i.e., 1-year transition probabilities are estimated.*
`method`	estimation algorithm, in string format. Valid values are 'duration' or 'cohort'.

Value

Returns *m-by-m* matrices of credit transition probabilities

Author(s)

Abdoulaye (Ab) N'Diaye

References

MathWorld.com (2011). Matlab Central `http://www.mathworks.com/matlabcentral/`. Mathtools.net `http://www.mathtools.net/`.

VecOfTransData *Vector of Transition Counts*

Description

This function reshapes transition matrices of counts into a 'wide format', time series of transition counts.

Usage

```
VecOfTransData(lstCnt,ratingCat,startDate,endDate,snapshots)
```

Arguments

lstCnt	quarterly transition count data.
ratingCat	list containing the unique rating caetgories.
startDate	start date of the estimation time window, in string or numeric format. The default start date is the earliest date in 'data'.
endDate	end date of the estimation time window, in string or numeric format. The default end date is the latest date in 'data'. The end date cannot be a date before the start date.
snapshots	Integer indicating the number of credit-rating snapshots per year to be considered for the estimation. Valid values are 1, 4, 12, 54, and 356. For example, 1 = one snapshot per year

Value

The output is a time series vector of transition counts .

Author(s)

Abdoulaye (Ab) N'Diaye

Index