Package 'RTransProb'

April 11, 2017

Type Package

Title Analyze and Forecast Credit Migrations

Version 0.1.0

Date 2017-03-17

Author Ab NDiaye

Maintainer Ab NDiaye <pabdndiaye@gmail.com>

Description A set of functions used to automate commonly used methods in credit risk. This includes multiple methods for bootstrapping default rates and forecasting/stress testing credit exposures migrations, via Econometrics and Machine Learning algorithms.

License GPL-2

LazyData TRUE

RoxygenNote 6.0.1

Depends R (>= 2.2.0)

Imports chron, e1071, expm, matrixStats, nnet, pracma, zoo

NeedsCompilation no

Repository CRAN

Date/Publication 2017-04-11 12:00:27 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 transition probabilities using a bootstrapping procedure for cohort method.

Usage

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

Arguments

transMatrix	list of 12 average monthly credit transitions.
initCount	list of 12 average monthly initial counts (start vector).
nMonths	number of months in the horizon
nStates	number of rating categories.
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

```
sim <- 1000                                  #number of simulation
nMonths <- 12                                #number of month in horizon (default=12)
nStates <- 8                                 #number of ratings categories
initCount <- c(5,10,15,20,20,20,15,5)        #start vector (vector of initial counts)
## Not run:
tolerance <-cohort.CI(transMat,initCount,nMonths,nStates,sim)

## End(Not run)
```

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

Description

Calculate *Through-the-Cycle* transition rates using raw transition counts and starting obligor counts using the *cohort method*.

Usage

```
cohort.TTC(transCount, initCount, gYear, gHorizon, transtype)
```

Arguments

`transCount`	monthly "raw" data transitions
`initCount`	monthly "raw" data counts
`gYear`	number of relevant years
`gHorizon`	number of months in the horizon

`transtype` averaging method. The valid values

- *averageDataTransAnnual* - "Raw" Data Average
- *averageMonthlyTransAnnual* - Average of Monthly Transitions
- *transAnnual* - Average of Annual Transitions
- *averageMonthlyTransAnnualGM* - Geometric Average of Monthly Transitions

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/years using the a single year count to preserve dynamics, then average transitions and firms counts/years separately
2. Estimate the single-year quantities (estimate with monthly transition matrices), then average across years
3. Average annual transition matrices

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

Value

One of the output options corresponding to the 'transtype'value selected as input.

`averageDataTransAnnual`
: *"Raw" Data Average* - weighted average of separately scaled transition and obligor counts.

`averageMonthlyTransAnnual`
: *Average Monthly Transitions* - compute monthly transition matrices then average over years, e.g., average January matrices, then February matrices,...

`transAnnual` *Average Annual Transitions* - compute annual transition matrices then average over years

`averageMonthlyTransAnnualGM`
: *Geometric Average Monthly Transitions* - similar to Average Monthly Transitions except the geometric average of the monthly transition matrices, on an element-byelement basis, is used

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
## Not run:
#Get a sequence of years from "2000-01-01" to "2005-01-01"
TotalDateRange <- seq(as.Date("2000-01-01"), as.Date("2005-01-01"), "mon")
TotalDateRange_Yr <- seq(as.Date("2000-01-01"), as.Date("2005-01-01"), "years")

#Set parameters
snapshots <- 12    #monthly transition matrices
interval <- 1/12    #1 month transition matrices

#define list to hold initial counts and transition counts
lstCnt <-rep(list(list()), length(TotalDateRange_Yr))
lstInit <-rep(list(list()), length(TotalDateRange_Yr))

#initialize counters
nn <- 1
k <- 1
kk <- 1

#Create transition and initial counts (start vector) used as inputs to the function
for (l in 1:(length(TotalDateRange)-1)){

  istartDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l],format = "%Y-%m-%d")))
  iendDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l+1],format = "%Y-%m-%d")))
  DateRange        <- as.Date(matlabToPOSIX(cfdates(istartDate,iendDate,snapshots)))

  for(i in 1:(length(DateRange)-1)){

    sDate  <- DateRange[i]
    eDate    <- DateRange[i+1]

    Sample1<-TransitionProb(data,sDate, eDate, 'cohort', snapshots, interval)

    lstCnt[[k]][[kk]] <- Sample1$sampleTotals$totalsMat
    lstInit[[k]][[kk]]  <- Sample1$sampleTotals$totalsVec
    #print(lstCnt)

    if(kk>=snapshots){
      kk <- 1
      k <- k+1
    } else {
      kk <- kk+1
    }

  }

}



#average of monthly transition matrices using the monthly initial portfolio
```

```
#counts  (start vector) and transition counts.

#remove empty elements from the lists (lstCnt and lstInit)
lstInit <- lstInit[lapply(lstInit,length)>0]
lstCnt <- lstCnt[lapply(lstCnt,length)>0]

transtype <- "AverageMonthlyTransitions"   #averaging method
gYear <- length(lapply(lstCnt,length))     #Add Error Checking for this
gHorizon <- snapshots                     #number of months in the horizon (default= 12)

transMatTest <- cohort.TTC(lstCnt,lstInit,gYear,gHorizon,transtype)

## End(Not run)
```

`data` *data*

Description

data

`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

```
portWgts <-  c(100, 100, 100, 100, 100, 100, 100, 100)
sim <- 100
nHorizon <- 1.0
## Not run:
t<-duration.CI(AverageGeneratorMatrices,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,lstInit,gYear,snapshots,nStates,transtype,TotalDateRange)
```

Arguments

`lstCnt`	annual off-diagonal transition counts (matrix)
`lstInit`	Initial counts for each state (start vector)
`gYear`	count of annual date vector.
`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.
`nStates`	number of rating categories.
`transtype`	averaging method. The valid values • *transRaw* • *genAverage* • *transAverageGen* • *transAverage* • *individualAnnualTrans*
`TotalDateRange`	annual date vector.

Details

Given data representing x years of monthly off-diagonal transition counts, this function combines those data to obtain average monthly counts, 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,

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

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

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 *(annual generator matrices)*, then average across years
3. Average annual transition matrices

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

Value

`transRaw`	transition matrix created from generator matrix created using the average annual transition counts and average annual.
`genAverage`	average annual generator matrix.
`transAverageGen`	transition matrix from average annual generator matrix.
`transAnnual`	individual annual transition matrices.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
TotalDateRange <- seq(as.Date("2000-01-01"), as.Date("2002-01-01"), "years")

snapshots <- 1    #This uses a 1 year transition matrices
interval <- 1     #This gives a 1 year transition matrix

lstCnt <-rep(list(list()), length(TotalDateRange)-1)
lstInit <-rep(list(list()), length(TotalDateRange)-1)
kk <- 1
k <- 1
```

```
#Create transition and inital countsused as inputs to the function
for (l in 1:(length(TotalDateRange)-1)){

  istartDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l],format = "%Y-%m-%d")))
  iendDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l+1],format = "%Y-%m-%d")))
  DateRange        <- as.Date(matlabToPOSIX(cfdates(istartDate,iendDate,snapshots)))

  for(i in 1:(length(DateRange)-1)){

    sDate  <- DateRange[i]       # i.e "3/31/1990"
    eDate    <- DateRange[i+1]     # i.e "6/30/1990"

    t<-TransitionProb(data,sDate, eDate, 'duration', snapshots, interval)

    lstCnt[[k]][[kk]] <- t$sampleTotals$totalsMat
    lstInit[[k]][[kk]]  <- t$sampleTotals$totalsVec

    if(kk>=snapshots){
      kk <- 1
      k <- k+1
    } else {
      kk <- kk+1
    }

  }

}



transtype <- "genAverage"
gYear <- utils::head(TotalDateRange,-1)
nStates <- nrow(as.data.frame(lstInit[[1]][1]))

AverageGeneratorMatrices<-duration.TTC(lstCnt,lstInit,gYear,snapshots,nStates,transtype,
TotalDateRange)
```

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

Description

This function reshapes time series vector of transition counts into a 'Wide' Data Format. 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

```
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)
```

`histData` *histData*

Description

histData

`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_mnl *predData_mnl*

Description

predData_mnl

predData_svm *predData_svm*

Description

predData_svm

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

```
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)
```

`transForecast` *Forecast - using Credit Cycle*

Description

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

Usage

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

Arguments

`genMat`	Generator Matrix
`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.


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', snapshots, interval)
Example9.1 <- Example9$genMat
creditIndex <- -0.25


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

`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(transData, histData, predData, startDate,
                        endDate, ref, depVar, indVars, ratingCat, wgt)
```

Arguments

transData	dataframe containing date, beginning ratings, ending ratings, and transition counts.
histData	historical macroeconomic,financial and non-financial data.
predData	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.
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

```
#(1) Import Data
startDate <-"2000-01-01"
endDate   <-"2002-01-01"
TotalDateRange <- seq(as.Date(startDate), as.Date(endDate), "years")


#(2) Create quarterly transition matrices for the entire date range selected using the normal
#   procedures

#Set parameters
```

```
snapshots <- 4   #4,#12    #monthly transition matrices
interval <-  1   #1/12    #1 month transition matrices

#define list to hold initial counts and transition counts
lstCnt <-rep(list(list()), length(TotalDateRange)-1)
lstInit <-rep(list(list()), length(TotalDateRange)-1)

#initialize counters
n <- 1
k <- 1
for (l in 1:(length(TotalDateRange)-1)){

  istartDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l],format = "%Y-%m-%d")))
  iendDate = POSIXTomatlab(as.POSIXlt(as.Date(TotalDateRange[l+1],format = "%Y-%m-%d")))
  DateRange        <- as.Date(matlabToPOSIX(cfdates(istartDate,iendDate,snapshots)))

  for(i in 1:(length(DateRange)-1)){

    sDate  <- as.Date(DateRange[i])
    eDate    <- as.Date(DateRange[i+1])

    Example1<-TransitionProb(data,sDate, eDate, 'cohort', snapshots, interval)

   lstCnt[[k]][[n]] <- Example1$sampleTotals$totalsMat     #list of monthly transition counts
   lstInit[[k]][[n]]  <- Example1$sampleTotals$totalsVec   #list of monthly initial counts
    n <- n+1

    if(n>snapshots){
      n <- 1
      k <- k+1
    }

  }

}

#(3) extract the aggregate transition counts by date and transition type (i.e, AAA to AA,
#   AAA to A, AAA to BBB, etc...)
ratingCat <- c("A","B", "C", "D", "E", "F", "G", "N")

df <- VecOfTransData(lstCnt,ratingCat,startDate,endDate,snapshots)
df <- subset(df, df[["start_Rating"]] !="N")   #Notes: remove any record having a default
                                                #rating in the 'start_Rating'
#(4) Construct parameters for MNL model
startDate <- as.Date("2000-01-01")
endDate <- as.Date("2005-01-01")

ref <- "A"   #"AAA"
depVar <- c("end_rating")
indVars <-c("Macro1", "Financial1","Industry1")
wgt <-  "mCount"
ratingCat <- c("A","B", "C", "D", "E", "F", "G")  #NOTE: 'N' (Default Rating) is excluded
                                                   #(VBA book count data)
```

```
#(5) run mnl model
## Not run:
mnl_T<-transForecast_mnl(df, histData, predData_mnl, startDate, endDate, ref, depVar,
                         indVars, ratingCat, wgt)

## End(Not run)
```

`transForecast_svm` *Forecast - using Support Vector Machines*

Description

This model implements a forecasting method using Support Vector Machines.

Usage

```
transForecast_svm(transData, histData, predData_svm, startDate, endDate,
                  depVar, indVars,  ratingCat, pct, tuning, kernelType, cost,
                  cost.weights, gamma, gamma.weights)
```

Arguments

`transData`	aggregate transition matrix data.
`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.
`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.

Author(s)

Abdoulaye (Ab) N'Diaye

Examples

```
startDate <- as.Date("1990-01-01")
endDate <- as.Date("2005-01-01")
depVar <- c("end_rating")
indVars <-c("Macro1", "Financial1","Industry1")
pct <- 0.8
wgt <-  "mCount"
ratingCat <- c("A","B", "C", "D", "E", "F", "G", "N")
lstCategoricalVars <- c("end_rating")
tuning <- "TRUE"
cost <- 100
gamma <- .1
cost.weights <- c(0.1, 10, 100)
gamma.weights <- c(0.01, 0.25, 0.5, 1)
kernelType <- "radial"

## Not run:
transData <- expandTransData(df,wgt)

## End(Not run)
## Not run:
svm_T<-transForecast_svm(transData, histData, predData_svm,  startDate, endDate,
       depVar,indVars,  ratingCat, pct, tuning, kernelType,cost, cost.weights,
       gamma, gamma.weights)

## 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(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*.

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 $\Delta t = 1year$ 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

```
#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(data,startDate,endDate,'cohort', snapshots, interval)

## Not run:
#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(data,startDate, endDate,'duration', snapshots, interval)
```

```
## End(Not run)

#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(data,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(data,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(data,startDate, endDate,'cohort', snapshots, interval)

## Not run:
#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(data,startDate, endDate,'duration', snapshots, interval)

## End(Not run)

#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(data,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(data,startDate, endDate,'cohort', snapshots, interval)
```

`VecOfTransData` *Vector of Transition Counts*

Description

This function reshapes transition matrices of counts into a time series vector 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