helperFunctions.R

maximiliandeichmann

Fri Jun 29 14:58:04 2018

#import libraries  
library(diverse)

## Loading required package: proxy

## Warning: package 'proxy' was built under R version 3.4.4

##   
## Attaching package: 'proxy'

## The following objects are masked from 'package:stats':  
##   
## as.dist, dist

## The following object is masked from 'package:base':  
##   
## as.matrix

## Loading required package: reshape2

## Loading required package: foreign

## Warning: package 'foreign' was built under R version 3.4.4

# calculate and append hhi for deals  
hhi <- function(df,variable,HHIName) {  
   
 environment(hhi)  
   
 #check variables  
 if (missing(df))  
 stop("Need to specify df.")  
 if (missing(variable))  
 stop("Need to specify variable.")  
 if (missing(HHIName))  
 stop("Need to specify HHIName.")  
 if(!is.data.frame(df))  
 stop("df must be data frame")  
 if(!is.character(variable))  
 stop("variable must be a string")  
 if(!is.character(HHIName))  
 stop("HHIName must be a string")  
   
 #get number of funds  
 maxFundNr = max(df$Investor\_fund\_ID)  
 funds <- 1:maxFundNr  
   
 #loop through funds  
 for(a in seq(from=1, to=maxFundNr, by=1)) {  
   
 #get df with fund only  
 subdf <- subset(df, Investor\_fund\_ID == a)  
   
 #sort by date  
 out <- subdf[order(as.Date(subdf$Deal\_Date)),]  
   
 #get all data of selected variable  
 data <- out[[variable]]  
   
 #get unique data of selected variable  
 uniqueData <- unique(data)  
   
 #create empty matrix to arrange data  
 hhiMatrix <- matrix(0L, nrow = length(uniqueData), ncol = length(data))  
   
 #loop through data  
 for (i in seq(from=1,to=length(data))) {  
   
 #check which unique data, selected data fits  
 index <- match(data[i],uniqueData)  
   
 if(i > 1) {  
   
 #go through the current column  
 for (x in seq(from=1, to=length(uniqueData))) {  
   
 #copy values from last column  
 hhiMatrix[x,i] <- hhiMatrix[x,i-1]  
 }   
 }  
   
 #increase value in current column for indexed value  
 hhiMatrix[index,i] <- hhiMatrix[index,i] + 1  
 }  
  
 #calculate HHI for the entire matrix  
 hhiReturn <- diversity(hhiMatrix, type='hh', category\_row=TRUE)  
   
 # 1 - HHI to have diversification = 1 and specialisation = 0  
 for (i in 1:nrow(hhiReturn)) {  
 hhiReturn$HHI[i] <- 1 - hhiReturn$HHI[i]  
 }  
   
 hhiReturn <- hhiReturn[order(as.numeric(rownames(hhiReturn))),,drop=FALSE]  
 assign('hhiReturn', hhiReturn, pos=.GlobalEnv)  
  
 #manage output   
 names(hhiReturn) <- c(HHIName)  
 if (a == 1) {  
 output <- cbind(out,hhiReturn)  
 } else {  
 output <- rbind(output,cbind(out,hhiReturn))  
 }  
 }  
 return(output)  
}  
  
# crate dataframe on fund level  
fundData <- function(dealdf) {  
   
 #get number of funds  
 col.names = c("Fund\_ID", "Fund\_IRR", "Fund\_SD", "Investments", "Total\_Investments", "GeoHHI","StageHHI","PIGHHI","PICHHI","PISHHI")  
 colClasses = c("integer", "integer", "double", "double", "double", "double", "double", "double", "double", "double")  
   
 funddf <- read.table(text = "", colClasses = colClasses, col.names = col.names)  
   
 maxFundNr = max(dealdf$Investor\_fund\_ID)  
 funds <- 1:maxFundNr  
   
 #loop through funds  
 for(a in seq(from=1, to=maxFundNr, by=1)) {  
   
 #get df with fund only  
 subdf <- subset(dealdf, Investor\_fund\_ID == a)  
   
 #sort by date  
 out <- subdf[order(as.Date(subdf$Deal\_Date)),]  
   
 # weighted average of irr to get fund irr  
 wa <- weighted.mean(out$Gross\_IRR, out$Deal\_Size)  
 sd <- sd(out$Gross\_IRR)  
   
 # create row  
 funddf[nrow(funddf) + 1,] = list(out$Investor\_fund\_ID[nrow(out)],  
 wa,  
 sd,  
 nrow(out),  
 sum(out$Deal\_Size),  
 out$GeoHHI[nrow(out)],  
 out$StageHHI[nrow(out)],  
 out$PIGHHI[nrow(out)],  
 out$PICHHI[nrow(out)],  
 out$PISHHI[nrow(out)])  
 }  
 return(funddf)  
}  
  
  
  
  
  
  
hhiTimeSeries <- function(df, variables) {  
   
 colClasses = c("Date", "double", "double")  
 col.names = c("Date", "Deal\_ID", "Investor\_fund\_ID")  
   
 for (i in 1:length(variables)) {  
 col.names[i+3] <- variables[i]  
 colClasses[i+3] <- "double"  
 }  
   
 timeSeriesdf <- read.table(text = "",  
 colClasses = colClasses,  
 col.names = col.names)  
  
}