Annex2\_2

Alejandro Vaca

3/3/2018

In the first chunk, the whole data mining and transformation process is performed, using the same code as in the previous part of Annex 2. However, as the directory must be changed. This is because in Annex2, as seen in previous code, a generic function to read all csv is used. In this section, we need to read also, but separately, all data from Google Trends. This means that if we do not choose a new directory with all the cryptocurrencies’ csvs plus the google trends csvs, our generic function would read the latter also; therefore the data for the beginning of Annex 1 would not work. For this reason, the first thing to do for running this Rmd is to choose as a directory (in my computer): ~Documents/TFG/google\_interest\_portfolio\_selection. For the first part of the Annex 1 (the Rmd previous to this), the directory: ~Documents/TFG/portfolio\_analysis\_tfg should be chosen.

library(readr)  
btc <- read\_csv('btc.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

dash <- read\_csv('dash.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

eth <- read\_csv('eth.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

iota <- read\_csv('iot.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

lsk <- read\_csv('lsk.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

maid <- read\_csv('maid.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_integer(),  
## volumeto = col\_double()  
## )

neo <- read\_csv('neo.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

xem <- read\_csv('xem.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

xmr <- read\_csv('xmr.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

xrp <- read\_csv('xrp.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

zec <- read\_csv('zec.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

btc = data.frame(btc)  
dash = data.frame(dash)  
eth = data.frame(eth)  
iota = data.frame(iota)  
lsk = data.frame(lsk)  
maid = data.frame(maid)  
neo = data.frame(neo)  
xem = data.frame(xem)  
xmr = data.frame(xmr)  
xrp = data.frame(xrp)  
zec = data.frame(zec)  
  
  
btc = btc[ , 2:5]  
dash = dash[ , 2:5]  
eth = eth[ , 2:5]  
iota = iota[ , 2:5]  
lsk = lsk[ , 2:5]  
neo = neo[ , 2:5]  
xem = xem[ , 2:5]  
xmr = xmr[ , 2:5]  
xrp = xrp[ , 2:5]  
zec = zec[ , 2:5]  
maid = maid[ , 2:5]  
  
  
btc$volatility = btc[,3] - btc[,4]  
dash$volatility = dash[,3] - dash[,4]  
#we cannot use eos because there is no data about its volatility (high -low)  
eth$volatility = eth[,3] - eth[,4]  
iota$volatility = iota[,3] - iota[,4]  
lsk$volatility = lsk[,3] - lsk[,4]  
neo$volatility = neo[,3] - neo[,4]  
xem$volatility = xem[,3] - xem[,4]  
xmr$volatility = xmr[,3] - xmr[,4]  
xrp$volatility = xrp[,3] - xrp[,4]  
zec$volatility = zec[,3] - zec[,4]  
maid$volatility = maid[,3] - maid[,4]  
  
  
n = 196  
  
btc = btc[(nrow(btc)-n):nrow(btc),]  
dash = dash[(nrow(dash)-n):nrow(dash),]  
eth = eth[(nrow(eth)-n):nrow(eth),]  
iota = iota[(nrow(iota)-n):nrow(iota),]  
lsk = lsk[(nrow(lsk)-n):nrow(lsk),]  
xem = xem[(nrow(xem)-n):nrow(xem),]  
xmr = xmr[(nrow(xmr)-n):nrow(xmr),]  
xrp = xrp[(nrow(xrp)-n):nrow(xrp),]  
zec = zec[(nrow(zec)-n):nrow(zec),]  
maid = maid[(nrow(maid)-n):nrow(maid),]  
  
btc$ret = c(NA, diff(log(btc$close)))  
dash$ret = c(NA, diff(log(dash$close)))  
eth$ret = c(NA, diff(log(eth$close)))  
iota$ret = c(NA, diff(log(iota$close)))  
lsk$ret = c(NA, diff(log(lsk$close)))  
neo$ret = c(NA, diff(log(neo$close)))  
xem$ret = c(NA, diff(log(xem$close)))  
xmr$ret = c(NA, diff(log(xmr$close)))  
xrp$ret = c(NA, diff(log(xrp$close)))  
zec$ret = c(NA, diff(log(zec$close)))  
maid$ret = c(NA, diff(log(maid$close)))  
  
btc = btc[2:nrow(btc), ]  
eth = eth[2:nrow(eth), ]  
dash = dash[2:197, ]  
iota = iota[2:197, ]  
lsk = lsk[2:197, ]  
neo = neo[2:197, ]  
xem = xem[2:197, ]  
xmr = xmr[2:197, ]  
xrp = xrp[2:197, ]  
zec = zec[2:197, ]  
maid = maid[2:197, ]

#the next step is to create tables for each category of cryptocurrencies, and put some variables   
  
#for that we need to read the variables again, one by one, as done in the previous code.  
  
  
total\_perc\_ret\_btc = (btc$close[nrow(btc)] - btc$close[1])/btc$close[1]  
print(total\_perc\_ret\_btc)

## [1] 2.340743

total\_perc\_ret\_eth = (eth$close[nrow(eth)] - eth$close[1])/eth$close[1]  
print(total\_perc\_ret\_eth)

## [1] 2.655352

mean\_daily\_vol\_btc = mean(btc$volatility)  
print(mean\_daily\_vol\_btc)

## [1] 863.044

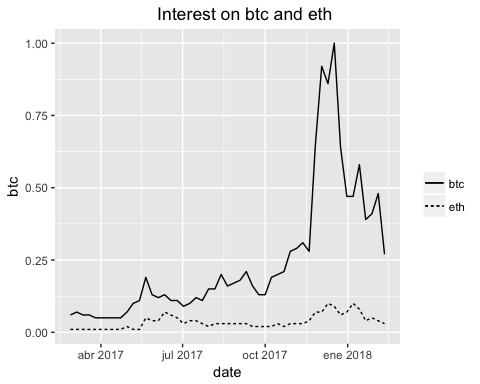
mean\_daily\_vol\_eth = mean(eth$volatility)  
print(mean\_daily\_vol\_eth) #much lower volatility

## [1] 59.27429

mean\_price\_btc = mean(btc$close)  
mean\_vol\_to\_mean\_price\_btc = mean\_daily\_vol\_btc/mean\_price\_btc  
  
mean\_price\_eth = mean(eth$close)  
mean\_vol\_to\_mean\_price\_eth = mean\_daily\_vol\_eth/mean\_price\_eth  
  
mean\_daily\_ret\_btc = mean(btc$ret)  
mean\_daily\_ret\_eth = mean(eth$ret)  
  
mean\_daily\_perc\_vol\_btc = mean(btc$volatility) / mean(btc$close)  
mean\_daily\_perc\_vol\_eth = mean(eth$volatility) / mean(eth$close)  
  
  
#we now need to take into account the google interest towards the cryptocurrencies  
  
library(readr)  
interest = read\_csv('eth\_btc\_google.csv')

## Parsed with column specification:  
## cols(  
## date = col\_date(format = ""),  
## btc = col\_integer(),  
## eth = col\_integer()  
## )

interest = data.frame(interest)  
interest$btc = interest$btc/100  
interest$eth = interest$eth/100  
interest$date = as.Date(interest$date)  
  
library(ggplot2)  
  
btc\_eth\_int <- ggplot(data = interest, aes(x = date)) +  
 geom\_line(aes(y = btc, linetype = "btc")) +  
 geom\_line(aes(y = eth, linetype = "eth")) +  
 ggtitle(' Interest on btc and eth ') +  
 scale\_linetype\_discrete(name ="")  
btc\_eth\_int



returns = cbind(btc$ret, eth$ret, dash$ret, iota$ret, lsk$ret, neo$ret, xem$ret, xmr$ret, xrp$ret, zec$ret, maid$ret)  
returns = data.frame(returns)  
colnames(returns) = c("btc", "eth", "dash", "iota", "lsk", "neo", "xem", "xmr", "xrp", "zec", "maid")

avg\_interest\_btc = mean(interest$btc)  
avg\_interest\_eth = mean(interest$eth)  
btc2 = read\_csv('btc.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

nrow(btc2) #this number of observations is going to be used for taking the inverse of the time, so that we give more weight to currencies that have had good performance but are not still in the maturity stage, not even close (btc is also not in the maturity stage, but we expect ethereum to have more relevance in the coming years, as in less time it has done as much as bitcoin for the blockchain community, if not more)

## [1] 2773

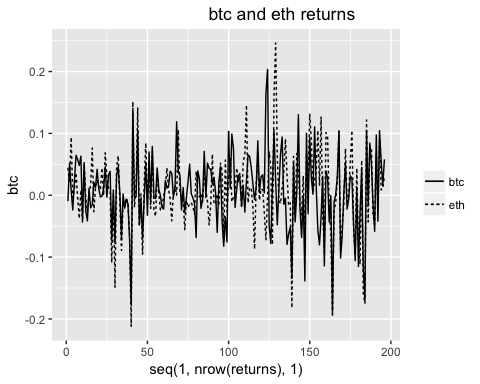
inv\_time\_btc = 1 / (nrow(btc2) / 365) #0.1316264  
  
eth2 = read\_csv('eth.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

nrow(eth2)

## [1] 926

inv\_time\_eth = 1 / (nrow(eth2) / 365) #0.3941685  
  
#let's visualize the movement of the returns of both series  
ggplot(data = returns, aes(x = seq(1, nrow(returns), 1))) +  
 geom\_line(aes(y = btc, linetype = "btc")) +  
 geom\_line(aes(y = eth, linetype = "eth")) +  
 ggtitle(' btc and eth returns') +  
 scale\_linetype\_discrete(name = "")



#this chunk contains all the rest of calculations for including in the excel spreadsheet (where the final computation of portfolio weights is carried out)  
  
#anonymity cryptos  
total\_perc\_ret\_xmr = (xmr$close[nrow(xmr)] - xmr$close[1])/xmr$close[1]  
total\_perc\_ret\_zec = (zec$close[nrow(zec)] - zec$close[1])/zec$close[1]  
total\_perc\_ret\_dash = (dash$close[nrow(dash)] - dash$close[1])/dash$close[1]  
  
mean\_ret\_xmr = mean(returns$xmr)  
mean\_ret\_zec = mean(returns$zec)  
mean\_ret\_dash = mean(returns$dash)  
  
  
anonim\_interest = read\_csv('anonimity\_cryptos\_interest.csv')

## Parsed with column specification:  
## cols(  
## date = col\_date(format = ""),  
## dash = col\_integer(),  
## zec = col\_integer(),  
## xmr = col\_integer()  
## )

anonim\_interest = data.frame(anonim\_interest)  
anonim\_interest$dash = anonim\_interest$dash/100  
anonim\_interest$zec = anonim\_interest$zec/100  
anonim\_interest$xmr = anonim\_interest$xmr/100  
  
mean\_int\_dash = mean(anonim\_interest$dash)  
mean\_int\_zec = mean(anonim\_interest$zec)  
mean\_int\_xmr = mean(anonim\_interest$xmr)  
  
mean\_vol\_dash = mean(dash$volatility)  
mean\_vol\_zec = mean(zec$volatility)  
mean\_vol\_xmr = mean(xmr$volatility)  
  
mean\_perc\_vol\_dash = mean\_vol\_dash / mean(dash$close)  
mean\_perc\_vol\_zec = mean\_vol\_zec / mean(zec$close)  
mean\_perc\_vol\_xmr = mean\_vol\_xmr / mean(xmr$close)  
  
xmr2 = read\_csv('xmr.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

nrow(xmr2)

## [1] 1116

inv\_time\_xmr = 1 / (nrow(xmr2) / 365)  
  
zec2 = read\_csv('zec.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

inv\_time\_zec = 1 / (nrow(zec2) / 365)  
  
dash2 = read\_csv('dash.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

inv\_time\_dash = 1 / (nrow(dash2) / 365)  
  
  
xem2 = read\_csv('xem.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

tot\_perc\_ret\_xem = (xem$close[nrow(xem)] - xem$close[1]) / xem$close[1]  
tot\_perc\_ret\_xem

## [1] 1.402948

xrp2 = read\_csv('xrp.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

tot\_perc\_ret\_xrp = (xrp$close[nrow(xrp)] - xrp$close[1] ) / xrp$close[1]  
tot\_perc\_ret\_xrp

## [1] 5.437292

tot\_perc\_ret\_neo = (neo$close[nrow(neo)] - neo$close[1]) / neo$close[1]  
tot\_perc\_ret\_neo

## [1] 7.43707

mean\_daily\_ret\_xem = mean(xem$ret)  
mean\_daily\_ret\_xem

## [1] 0.004787476

mean\_daily\_ret\_xrp = mean(xrp$ret)  
mean\_daily\_ret\_xrp

## [1] 0.009371945

mean\_daily\_ret\_neo = mean(neo$ret)  
mean\_daily\_ret\_neo

## [1] 0.01150895

platform\_int = read\_csv('platform\_interest.csv')

## Parsed with column specification:  
## cols(  
## date = col\_date(format = ""),  
## xem = col\_integer(),  
## xrp = col\_character(),  
## neo = col\_integer()  
## )

platform\_int = data.frame(platform\_int)  
platform\_int$xrp[1:3] = c(0,0,0) #we clean the data to have only numbers.   
platform\_int$xem = platform\_int$xem/100  
platform\_int$xrp = as.numeric(platform\_int$xrp)  
#platform\_int = gsub(pattern = '<1', x = platform\_int, replacement = 0.5)  
platform\_int$xrp = platform\_int$xrp/100  
platform\_int$neo = platform\_int$neo/100  
  
mean\_vol\_xem = mean(xem$volatility)  
mean\_vol\_xrp = mean(xrp$volatility)  
mean\_vol\_neo = mean(neo$volatility)  
mean\_vol\_xem

## [1] 0.09538571

mean\_vol\_xrp

## [1] 0.1119

mean\_vol\_neo

## [1] 10.43291

mean\_perc\_vol\_xem = mean\_vol\_xem/mean(xem$close)  
mean\_perc\_vol\_xrp = mean\_vol\_xrp/mean(xrp$close)  
mean\_perc\_vol\_neo = mean\_vol\_neo/mean(neo$close)  
  
inv\_time\_xem = 1/(nrow(xem2)/365)  
inv\_time\_xrp = 1/(nrow(xrp2)/365)  
  
neo2 = read\_csv('neo.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

inv\_time\_neo = 1/(nrow(neo2)/365)  
  
  
#now let's do the protocol currencies analysis  
tot\_perc\_ret\_iota = (iota$close[nrow(iota)] - iota$close[1]) / iota$close[1]  
tot\_perc\_ret\_lsk = (lsk$close[nrow(lsk)] - lsk$close[1]) / lsk$close[1]  
tot\_perc\_ret\_maid = (maid$close[nrow(maid)] - maid$close[1]) / maid$close[1]  
  
mean\_daily\_ret\_iota = mean(iota$ret)  
mean\_daily\_ret\_iota

## [1] 0.008573417

mean\_daily\_ret\_lsk = mean(lsk$ret)  
mean\_daily\_ret\_lsk

## [1] 0.01424316

mean\_daily\_ret\_maid = mean(maid$ret)  
mean\_daily\_ret\_maid

## [1] 0.001079182

#now let us check the google interest in each of the protocol currencies  
  
protocol\_int = read\_csv('protocol\_interest.csv')

## Parsed with column specification:  
## cols(  
## date = col\_date(format = ""),  
## iota = col\_integer(),  
## lsk = col\_character(),  
## maid = col\_character()  
## )

protocol\_int$lsk = ifelse(protocol\_int$lsk == "<1", 0.5, protocol\_int$lsk) #we transform all values <1 to 0.5, as the expected mean between 0 and 1 is supposed to be this if the range behaves normally (bell-shaped).  
protocol\_int$maid = ifelse(protocol\_int$maid == "<1", 0.5, protocol\_int$maid)  
  
protocol\_int$iota = as.numeric(protocol\_int$iota)  
protocol\_int$lsk = as.numeric(protocol\_int$lsk)  
protocol\_int$maid = as.numeric(protocol\_int$maid)  
  
protocol\_int$iota = protocol\_int$iota/100  
protocol\_int$lsk = protocol\_int$lsk/100  
protocol\_int$maid = protocol\_int$maid / 100  
  
avg\_int\_iota = mean(protocol\_int$iota)  
avg\_int\_lsk = mean(protocol\_int$lsk)  
avg\_int\_maid = mean(protocol\_int$maid)  
  
mean\_vol\_iota = mean(iota$volatility)  
mean\_vol\_lsk = mean(lsk$volatility)  
mean\_vol\_maid = mean(maid$volatility)  
  
mean\_vol\_iota

## [1] 0.3354153

mean\_vol\_lsk

## [1] 2.117857

mean\_vol\_maid

## [1] 0.09970969

perc\_vol\_iota = mean\_vol\_iota / mean(iota$close)  
perc\_vol\_lsk = mean\_vol\_lsk / mean(lsk$close)  
perc\_vol\_maid = mean\_vol\_maid / mean(maid$close)  
  
perc\_vol\_iota

## [1] 0.2063972

perc\_vol\_lsk

## [1] 0.1786537

perc\_vol\_maid

## [1] 0.1829106

iota2 = read\_csv('iot.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

inv\_time\_iota = 1 / (nrow(iota2) / 365)  
  
lsk2 = read\_csv('lsk.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_double(),  
## volumeto = col\_double()  
## )

inv\_time\_lsk = 1 / (nrow(lsk2) / 365)  
  
maid2 = read\_csv('maid.csv')

## Parsed with column specification:  
## cols(  
## time = col\_double(),  
## timeDate = col\_date(format = ""),  
## close = col\_double(),  
## high = col\_double(),  
## low = col\_double(),  
## open = col\_double(),  
## volumefrom = col\_integer(),  
## volumeto = col\_double()  
## )

inv\_time\_maid = 1 / (nrow(maid) / 365)  
  
inv\_time\_iota

## [1] 1.46

inv\_time\_lsk

## [1] 1.01108

inv\_time\_maid

## [1] 1.862245