Primer Capítulo: Retornos y Portafolio

Dereck Amesquita & Brandon Gil - DAEC Consultoría

Agosto - 2022

Resumen

El siguiente texto tiene por objetivo mostrar una guía que permita al lector entender como implementar mediante R, cálculos aplicados al rendimiento de portafolio.

Temas

- Gráficos de precios
- Retornos de portafolio
- Retornos mediante matrices
- Time-Weighted Rate of Return
- Money-Weighted Rate of Return

Primero instalamos las librerias necesarias en este caso la que me llamó más la atención fue la librería quantmod que nos facilitará al momento de hacer el siguiente trabajo. Esta librería nos ayuda bastante a la vosualizacion de datos. Principamente usada para análisis cuantitativo

Librerías necesarias :

```
library("quantmod")
library('dplyr')
library("anytime")
```

Recuerda que si no tienes instaladas estas librerias deberas usar, por ejemplo: install.packages("WDI")

Es importante recordar que cada vez que ya hayamos descargado cualquier librería tenemos que activarla para usarla.

Importación de data

#Para este ejemplo decidimos trabajr con BlackRock, Apple y Tesla que son empresas que sean vuelto populares los últimos meses.

Lo primero que tenemos que hacer es importar la data necesaria, en este caso usaremos Yahoo Finance, de donde extraeremos data histórica desde el 2018-01-01 hasta el 2022-07-30.

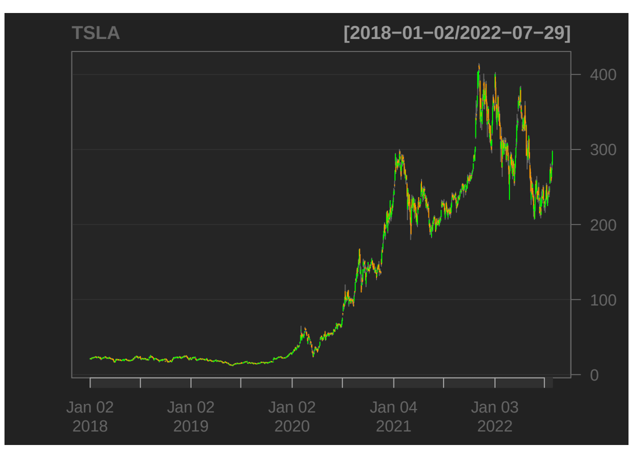
```
getSymbols(c('BLK','AAPL','TSLA'),
           src = "yahoo",
           from = "2018-01-01",
           to = "2022-07-30",
           periodicity = "daily")
```

```
## [1] "BLK"  "AAPL" "TSLA"
```

Con el comando ChartSeries que viene incluido en la librería "quantmod" podemos ver la evolución de las acciones en el mercado.

```
chartSeries(AAPL, TA=NULL)
```


```
chartSeries(TSLA, TA=NULL)
```



```r
chartSeries(BLK, TA=NULL)
```

Trabajando el retorno de los portafolios

Seleccionaremos la última columna de los datos descargados, de esta forma obtenemos el precio de dicha acción, le aplicamos "Delt", el cual calculará la variación porcentual, se cambiará de nombre según cada una.

```r
apple = Delt(AAPL[,6])
names(apple) = "APPLE"
head(apple)
```

```
##                    APPLE
## 2018-01-02            NA
## 2018-01-03 -0.0001742875
## 2018-01-04  0.0046450396
## 2018-01-05  0.0113853590
## 2018-01-08 -0.0037141051
## 2018-01-09 -0.0001147747
```

```r
blk = Delt(BLK[,6])
names(blk) = "BLK"
head(blk)
```

```
##                    BLK
## 2018-01-02          NA
## 2018-01-03 0.010550788
## 2018-01-04 0.013201329
## 2018-01-05 0.008520095
## 2018-01-08 0.007458295
## 2018-01-09 0.008404435
```

```r
tesla = Delt(TSLA[,6])
names(tesla) = "TSLA"
head(tesla)
```

```
##                   TSLA
## 2018-01-02           NA
## 2018-01-03 -0.010233113
## 2018-01-04 -0.008289976
## 2018-01-05  0.006229706
## 2018-01-08  0.062638244
## 2018-01-09 -0.008085402
```

Por usar Delt, el primer valor resulta como NA, puesto que, para calcular variaciones, se necesitan como mínimo dos periodos.

```
retornos = cbind(tesla,blk,apple)
head(retornos)
```

```
##                   TSLA         BLK         APPLE
## 2018-01-02           NA          NA            NA
## 2018-01-03 -0.010233113 0.010550788 -0.0001742875
## 2018-01-04 -0.008289976 0.013201329  0.0046450396
## 2018-01-05  0.006229706 0.008520095  0.0113853590
## 2018-01-08  0.062638244 0.007458295 -0.0037141051
## 2018-01-09 -0.008085402 0.008404435 -0.0001147747
```

```
retornos = retornos[-1,]
head(retornos)
```

```
##                   TSLA          BLK         APPLE
## 2018-01-03 -0.010233113  0.010550788 -0.0001742875
## 2018-01-04 -0.008289976  0.013201329  0.0046450396
## 2018-01-05  0.006229706  0.008520095  0.0113853590
## 2018-01-08  0.062638244  0.007458295 -0.0037141051
## 2018-01-09 -0.008085402  0.008404435 -0.0001147747
## 2018-01-10  0.003326441 -0.004438717 -0.0002294554
```

Procedemos a crear distintos pesos.

```
options(scipen = 999) #Evitamos notación cientifica, pero con 0 obtenemos el default
i.tesla = 50000
i.blk = 30000
i.apple = 20000
i.total = i.tesla + i.blk + i.apple
```

Cálculo de pesos

```
(w.tesla = i.tesla / i.total)
```

```
## [1] 0.5
```

```r
(w.apple = i.apple / i.total)
```

```
## [1] 0.2
```

```r
(w.blk = i.blk / i.total)
```

```
## [1] 0.3
```

Retorno de portafolios

```r
ret.por = 1 + retornos
head(ret.por)
```

```
##                 TSLA       BLK     APPLE
## 2018-01-03 0.9897669 1.0105508 0.9998257
## 2018-01-04 0.9917100 1.0132013 1.0046450
## 2018-01-05 1.0062297 1.0085201 1.0113854
## 2018-01-08 1.0626382 1.0074583 0.9962859
## 2018-01-09 0.9919146 1.0084044 0.9998852
## 2018-01-10 1.0033264 0.9955613 0.9997705
```

Retorno acumulado del portafolio

```r
retor.acum = cumprod(ret.por)
tail(retor.acum) # Esta es una suma directa periodo a periodo
```

```
##                TSLA      BLK    APPLE
## 2022-07-22 12.74030 1.394551 3.751393
## 2022-07-25 12.56201 1.400076 3.723640
## 2022-07-26 12.11400 1.371002 3.690774
## 2022-07-27 12.86089 1.415856 3.817126
## 2022-07-28 13.14541 1.452522 3.830760
## 2022-07-29 13.90587 1.472770 3.956382
```

Restamos el 1 que sumamos anteriormente, y lo haremos en el último periodo. Nos quedamos con el ultimo periodo puesto que este acumula el rendimiento a lo largo de todos los periodos dados.

```r
nrow(retor.acum)
```

```
## [1] 1151
```

```r
(retor.acum2 = retor.acum[nrow(retor.acum)] - 1)
```

```
##                TSLA       BLK    APPLE
## 2022-07-29 12.90587 0.4727699 2.956382
```

Calculando el retorno del portafolio mediante una multiplicación del rendimiento con el peso en el portafolio.

```r
(ret.final <- as.numeric(w.tesla*retor.acum2$TSLA +
                          w.blk * retor.acum2$BLK +
                          w.apple * retor.acum2$APPLE))
```

```
## [1] 7.186044
```

Es decir, en todo este tiempo el rendimiento de nuestro portafolio seria de 718%. Este resultado no considera el riesgo que se tiene y tampoco considera la poca diversificación de portafolio.

Uso de matrices

A traves de una vista matricial podemos ver que el rendimiento del portafolio es:

$$Rp = W^t R$$

Donde W^t es la transpuesta de la matriz de pesos, la cual se transforma para poder ser multiplicada con la matriz de rendimientos.

```r
pesos = c(w.apple, w.blk, w.tesla)
# Creando la matriz de pesos
(mat.pes = matrix(pesos,nrow=1)) #1 es el numero de filas
```

```
##      [,1] [,2] [,3]
## [1,]  0.2  0.3  0.5
```

```r
# Creando la matriz de rendimientos
(mat.ret = matrix(retor.acum2,nrow=3)) #Ahora le decimos que divida en tres filas.
```

```
##            [,1]
## [1,] 12.9058735
## [2,]  0.4727699
## [3,]  2.9563824
```

Si necesitamos convertir una matriz a su transpuesta podemos usar t(matriz).

Ahora vamos a calcular el retorno del portafolio:

```r
(rpmatrix = mat.pes %*% mat.ret)
```

```
##          [,1]
## [1,] 4.201197
```

Time-Weighted Rate of Return

El time-weighted rate of return vendría a ser en pocas palabras, el porcentaje de retorno que se obtiene en un determinado periodo de tiempo corrigiendo y eliminando el sesgo producido por las entradas y salidas de dinero. Se puede entender como la productoria del Holding Period Return

$$TWRR = \prod_{t=1}^{T} (1 + HPR_t)$$

Holding Period Return

El Holding Period Return es basicamente el rendimiento total obtenido por tener ciertos activos o carteras de acciones en un determinado periodo de tiempo.

$$HPR_t = \frac{V_t + C_t - V_{t-1}}{V_{t-1}}$$

Donde, V_t es el valor de mercado del portafolio en el tiempo t. C_t, es el cashflow de dicho periodo.

Introduciendo la data

Ahora usaremos el comando 'anydate' el cual nos permitira introducir valores de fecha en R.

```
fecha = anydate(c("2020/12/31", "2021/03/31","2021/06/30", "2021/07/31",
                  "2021/09/30", "2021/12/31"))
mv = c(2000000, 1950000, 2000000, 2220000, 2400000, 2500000)
cf = c(0, 0, 0, 20000, 400, -5000)

cbind(data.frame(fecha), mv, cf)
```

```
##        fecha      mv    cf
## 1 2020-12-31 2000000     0
## 2 2021-03-31 1950000     0
## 3 2021-06-30 2000000     0
## 4 2021-07-31 2220000 20000
## 5 2021-09-30 2400000   400
## 6 2021-12-31 2500000 -5000
```

Creamos una matriz de ceros con tamaño 6. Utilizamos length(mv), el cual nos dice cuantos elementos tiene el vector mv.

```
(hpr = rep(0, length(mv)))
```

```
## [1] 0 0 0 0 0 0
```

En el siguiente bucle, lo que haremos es recorrer elementos donde calculamos el HPR, desde el elemento 2, puesto que se necesita un valor anterior para ser calculado.

```
for (i in (2 : (length(cf)))) {
  hpr[i] = (mv[i] - mv[i-1] + cf[i]) / mv[i - 1]
}
cbind(data.frame(fecha), mv, cf,hpr)
```

```
##        fecha      mv    cf         hpr
## 1 2020-12-31 2000000     0  0.00000000
## 2 2021-03-31 1950000     0 -0.02500000
## 3 2021-06-30 2000000     0  0.02564103
## 4 2021-07-31 2220000 20000  0.12000000
## 5 2021-09-30 2400000   400  0.08126126
## 6 2021-12-31 2500000 -5000  0.03958333
```

```r
(hpr2 = 1+hpr)
```

```
## [1] 1.000000 0.975000 1.025641 1.120000 1.081261 1.039583
```

```r
# Obteniendo el TWRR mediante dos formas
#Forma 1 Acumulando los productos
(cum.ret = cumprod(hpr2)) # Acumulamos los rendimientos multiplicando cada elemento
```

```
## [1] 1.000000 0.975000 1.000000 1.120000 1.211013 1.258949
```

```r
(cum.ret[length(cf)] -1 ) # Accedemos al ultimo valor y le quitamos 1
```

```
## [1] 0.2589485
```

```r
# Forma 2 Multiplicando
(prod(hpr2))-1 #Multiplicamos todos los elementos y le restamos 1
```

```
## [1] 0.2589485
```

Money-Weighted Rate of Return

Concepto que nos arroja un valor aproximado a una TIR o una IRR (Internal Rate of Return) pero de portafolio. Congela el valor presente neto del portafolio (NPV) (Valor presente menos los costos de inversión) a cero.

$$0 = \sum_{t=0}^{T} \frac{C_t}{(1 + MWRR)^t}$$

Podemos trabajarlo si:

$$a = (1 + MWRR)^{-1}$$

Podemos obtener:

$$0 = C_0 + aC_1 + a^2C_2 + ... + a^nC_n$$

Creamos la función del valor presente:

```r
pv = function(cf, dates, r){
  t = as.numeric((dates - dates[1]) / 365)
  pv_factor = (1 + r)^-t
  pv = cf * pv_factor
  value = sum(pv)
  return(value)
}
```

```r
mwrr = function(cf, dates, guess) {
  delta.x = 0.01
  tol = 0.00000001
  cur.x =guess
  iter = 0
```

```r
  for (i in 1:1000) {
    fx = pv(cf, dates, cur.x)
    cur.x.delta = cur.x - delta.x
    fx.delta = pv(cf, dates, cur.x.delta)
    dx = (fx - fx.delta) / delta.x
    cur.x = cur.x - (fx/dx)
    iter = iter+1
    cat("En la iteración", iter, "MWRR es igual a", cur.x, "\n") #Funciona como Print
    if (abs(fx) < tol) break
  }
}
```

```r
cf = c(-100000, 104000)
dates = anydate(c("2018/12/31", "2019/12/31"))
mwrr(cf, dates, 0.1)
```

```
## En la iteración 1 MWRR es igual a 0.03711538
## En la iteración 2 MWRR es igual a 0.03996426
## En la iteración 3 MWRR es igual a 0.03999966
## En la iteración 4 MWRR es igual a 0.04
## En la iteración 5 MWRR es igual a 0.04
## En la iteración 6 MWRR es igual a 0.04
## En la iteración 7 MWRR es igual a 0.04
## En la iteración 8 MWRR es igual a 0.04
```