Introducción a tidyquant

Llevando el análisis financiero al tidyverse

Gabriel Cabrera

16 de agosto de 2018 (2018-09-18)

Información de Contacto

gcabrerag@fen.uchile.cl
gcabrerag.rbind.io
@GaboC_g
@GaboCg
Facultad de Economía & Negocios, Universidad de Chile

Introducción a tidyquant

¿Qué es tidyquant?

1. Una "megalibrería" creada por Matt Dancho (`@mdancho84`) y proxima a ser la base del libro "Reproducible Finance with R: Code Flows and Shiny Apps for Portfolio Analysis" de Jonathan K. Regenstein Jr. (`@jkregenstein`)
2. **tidyquant** integra los mejores recursos/librerías para colectar y analizar datos financieros: zoo, xts, quantmod, TTR y PerformanceAnalytic, con la infraestructura **tidy data**[1] del tidyverse, permitiendo la interacción entre ambos.
3. Nos permite implementar las funciones de **ggplot2**, para visualizaciones hermosas!!!

[1] Para mayor detalle ver el paper "Tidy Data (2014)" de Hadley Wickham.

Breve resumen de las librerías

1. xts o eXtensible time series: Es una estructura dato y a la vez una librería para manipular series de tiempo. Detras se encuentra la estructura zoo.
2. quantmod o Quantitative Financial Modelling & Trading Framework: Es una librería diseñada para recuperar, manipular y modelar datos cuantitativos financieros.
3. TTR o Technical Trading Rules: Librería que incluye varias funciones para computar análisis técnico.
4. PerformanceAnalytics: Librería que incluye una colección de funciones econométricas para desempeño y análisis de riesgo. Se necesita los retornos y no los precios.

Pequeñas funciones con mucho poder

1. Obtener data de diversas fuentes podemos usar `tq_get()`.
2. Transmutar data: `tq_transmute()`.
3. mutar data: `tq_mutate()`.
4. Análisis de "performance": `'tq_performance()`.

¿Qué sucede si escribimos `tq_index()`, `tq_index_option` o `tq_exchange()`?

Descargando el S&P 500

Cargamos las librerías con las que vamos a trabajar

```
if(!require("pacman")) install.packages("pacman")
p_load("tidyverse","tidyquant","ggthemes")
```

Imaginemos por una momento que necesitamos obtener los precios del S&P 500 desde Enero del 2010 hasta 31 de Agosto del 2018 con periodicidad diaria:

```
sp500_precio  <- tq_get("^GSPC",
                        get = "stock.prices",
                        from = "2010-01-01",
                        to = "2018-08-01",
                        periodicity = "daily")
```

Tener presente que get es que "tipo" de datos financieros van a decargar. Cuando usamos `stock.prices`, se obtienen los datos desde Yahoo Finance. Si quisiera los *finacial statements*, escribo `get = "financial"` y los descarga desde **Google Finance**.

Ahora con quantmod

Podemos hacer los mismo de muchas formas, por ejemplo, usar directamente `quantmod`

```
sp500_quantmod <- getSymbols("^GSPC",
                             src = "yahoo",
                             from = "2010-01-01",
                             to = "2018-08-01",
                             periodicity = "daily")
```

Lo positivo de usar quantmod es que podemos aprovechar las funciones `chartSeries()`

```
chartSeries(GSPC)
```

Para verlo sin los *volume*

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

Para ver los últimos 3 meses

```
chartSeries(GSPC, subset = "last 3 months")
```

Podemos replicar el objeto creado `sp500_precio` usando:

```
sp500_quantmod <- as.data.frame(GSPC) %>%
                  as.tibble() %>%
                  mutate(date = index(GSPC)) %>%
                  rename("open" = "GSPC.Open", "high" = "GSPC.High",
                         "low"  = "GSPC.Low" , "close"= "GSPC.Close",
                         "volume" = "GSPC.Volume", "adjusted" = "GSPC.Adjusted")
```

¿Qué podemos observar?

Ahora con ggplot2

Continuaremos con el objeto `sp500_precio`, pero lo graficaremos usando ggplot2:

```
g <- ggplot(sp500_precio) + geom_line(aes(date,adjusted), color = "red")
g <- g + labs(title = "Precio S&P 500", subtitle = "Desde Enero 2010 hasta Julio 2018")
g <- g + theme_tq() + scale_color_tq()
g <- g + xlab("Periodo") + ylab("Precio")
g
```


Precio S&P 500
Desde Enero 2010 hasta Julio 2018
Precio
2500
2000
1500
1000
2010
2012
2014
2016
2018
Periodo

Si extrañan stata ... pueden usar

Para realizarlo deben usar la librería `ggthemes`.

Multiples Datos

Descargando Multiples Datos

Ahora veremos como podemos descargar más de un activo. Usaremos las siguientes empresas: Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX). Comenzamos con construir un objeto `tickers` con los nombres.

```
tickers <- c("ORCL", "IT", "NVDA", "NFLX")
```

Luego obtenemos los activos[2].

```
data_activos  <- tq_get(tickers,
                        get = "stock.prices",
                        from = "2010-01-01",
                        to = "2018-08-01",
                        periodicity = "daily")
```

[2] La opción `to` considera la fecha anterior a la solicitada.

Retornos y Retornos Acumulados

Calculo Retornos

La formula para calcular (log) retornos es:

$$r_t = log(1 + R_t) = log(\frac{P_t}{P_{t-1}}) = p_t - p_{t-1}$$

donde $p_t = log(P_t)$ es llamado "log price".

```
retornos_activos <- data_activos %>%
                        group_by(symbol) %>%
                        tq_transmute(select = close,
                                     mutate_fun = periodReturn,
                                     period = "daily",
                                     type = "log",
                                     col_rename = "retornos.diarios")
```

Calculo Retornos Acumulados

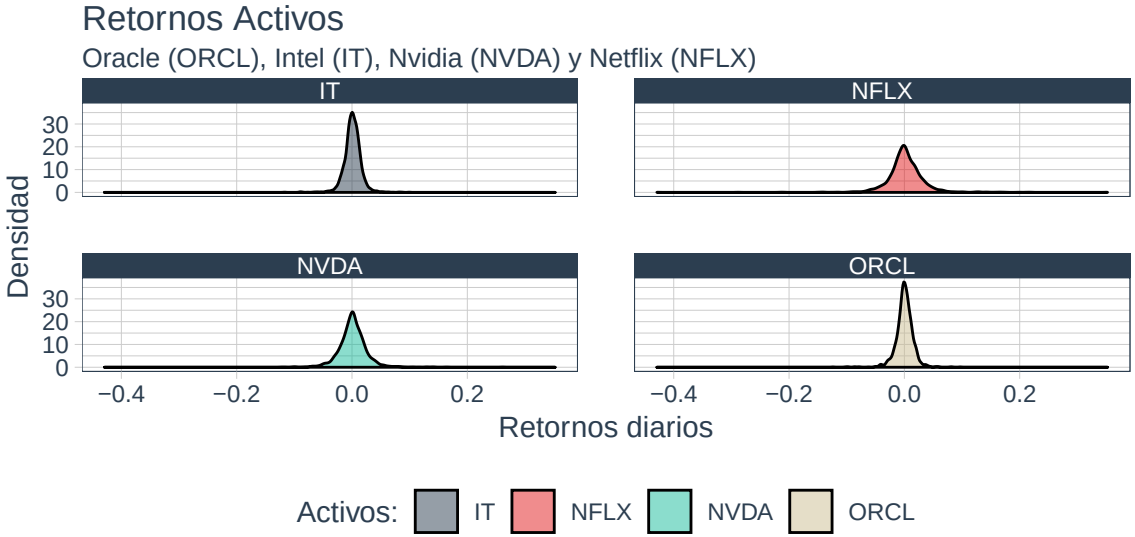
Como el nombre sugiere, el retorno acumulado indica el efecto agregado del precio. Podremos ver con mayor facilidad si se mantienen las perdidas (ganancias) por un periodo dado.

```
retornos_acum_activos <- retornos_activos %>%
                            group_by(symbol) %>%
                            mutate(ret.cum = cumsum(retornos.diarios))
```

Gráficando Retornos

Una manera de graficar los retornos es usar `geom_density()`, que es la versión suavisada de un histograma y se suele usar para datos que son continuos. Intenten con `geom_line()` y vean que sucede.

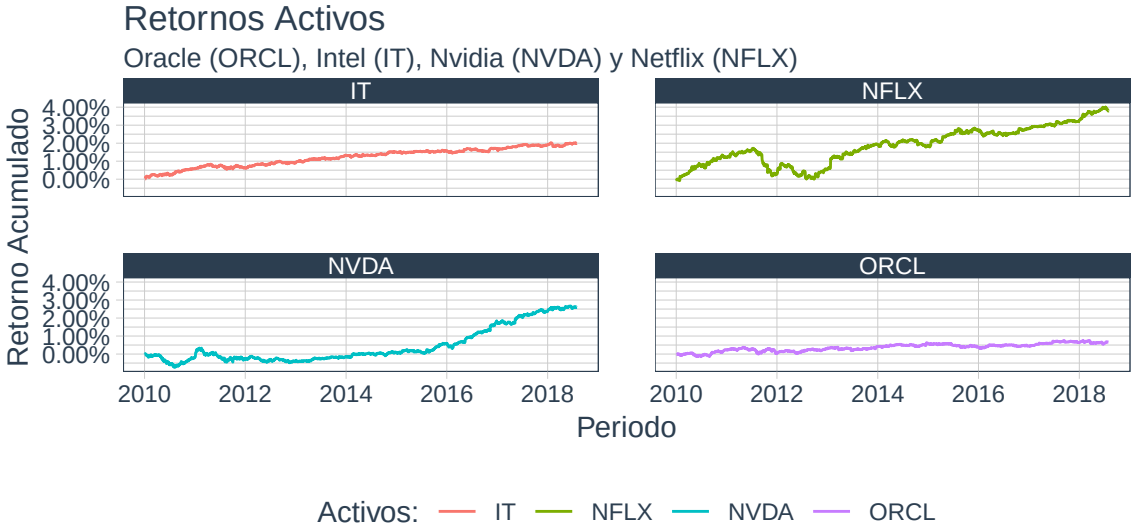
```
retornos_activos %>%
  ggplot(mapping = aes(x = retornos.diarios, fill = symbol))+
  geom_density(alpha = 0.5) +
  labs(title = "Retornos Activos",
       subtitle = "Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX)",
       x = "Retornos diarios", y = "Densidad") +
  theme_tq() +
  scale_fill_tq() +
  facet_wrap(~ symbol, ncol = 2) +
  guides(fill=guide_legend(title="Activos:"))
```


Retornos Activos
Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX)
IT
NFLX
NVDA
ORCL
Densidad
30
20
10
0
−0.4
−0.2
0.0
0.2
Retornos diarios
Activos:
IT
NFLX
NVDA
ORCL

Gráficando Retornos Acumulados

Para graficar los retornos acumulados, se recomienda usar `geom_line()`.

```
retornos_acum_activos %>%
     ggplot(mapping = aes(x = date, y = ret.cum/100, color = symbol)) +
     geom_line() +
     labs(title = "Retornos Activos",
          subtitle = "Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX)",
          x = "Periodo", y = "Retorno Acumulado") +
     theme_tq() +
     scale_fill_tq() +
     facet_wrap(~ symbol, ncol = 2) +
     guides(color = guide_legend(title="Activos:")) +
     scale_y_continuous(labels = scales::percent)
```


Retornos Activos
Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX)
IT
NFLX
NVDA
ORCL
Retorno Acumulado
4.00%
3.00%
2.00%
1.00%
0.00%
2010
2012
2014
2016
2018
Periodo
Activos:
IT
NFLX
NVDA
ORCL

Análisis Técnico

¿Qué es el análisis técnico?

1. El análisis técnico consiste en detectar determinados patrones de comportamiento de los precios en el pasado, con la esperanza de que dicho patrones vuelvan a repetirse y poder así aprovecharnos de ello.
2. Las bases provienen de la **Teoría de Dow**.
3. En esta sesiones veremos los *Bar Chart*, *Candlestick Chart* y las *Bandas de Bollinger*.

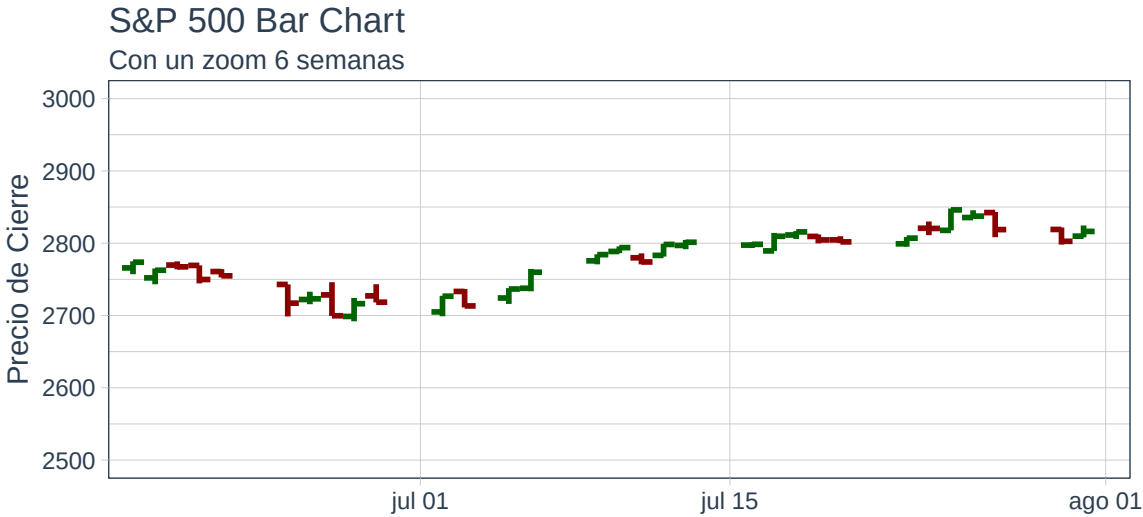
Bar Chart

Este gráfico es llamado OHLC charts, por Open-High-Low-Close. Si el precio de cierre es mayor que el de apertura es un "up day", caso contrario es un "down day". Se recomienda verlo en un periodo del tiempo, en nuestro caso los últimos 6 semanas.

```
end <- as_date("2018-07-31")
```

Luego

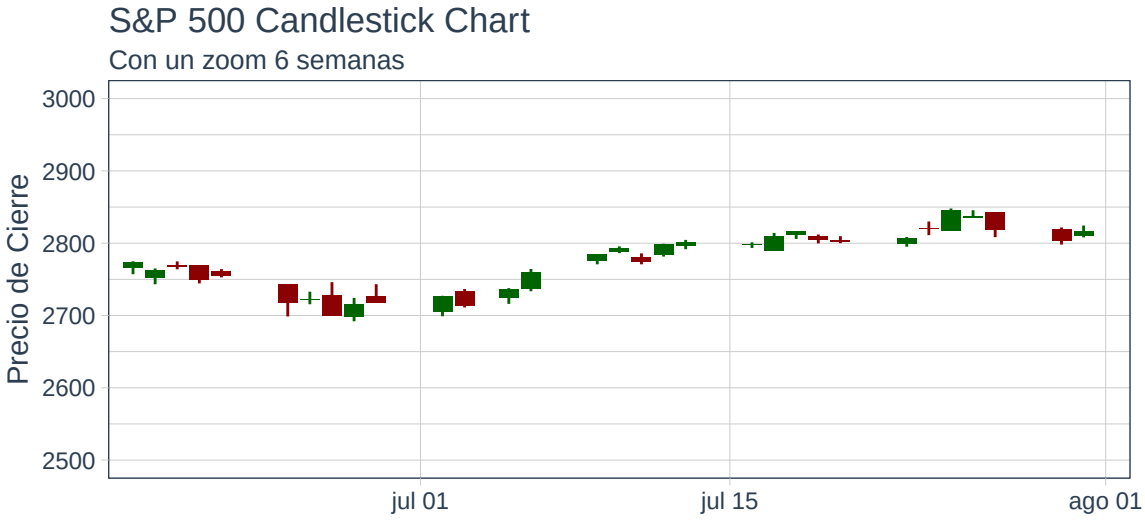
```
sp500_precio %>%
    ggplot(aes(x = date, y = close)) +
    geom_barchart(aes(open = open, high = high, low = low, close = close),
                  color_up = "darkgreen", color_down = "darkred", size = 1) +
    labs(title = "S&P 500 Bar Chart",
         subtitle = "Con un zoom 6 semanas",
         y = "Precio de Cierre", x = "") +
    coord_x_date(xlim = c(end - weeks(6), end),
                 ylim = c(2500, 3000)) +
    theme_tq()
```


S&P 500 Bar Chart
Con un zoom 6 semanas
Precio de Cierre
3000
2900
2800
2700
2600
2500
jul 01
jul 15
ago 01

Candlestick Chart

Un candlestick chart es otra forma de representar OHLC. Lo importante es que cuando es "up day" es **bullish** "down day" es **bearish**.

```
sp500_precio %>%
     ggplot(aes(x = date, y = close)) +
     geom_candlestick(aes(open = open, high = high, low = low, close = close),
                      color_up = "darkgreen", color_down = "darkred",
                      fill_up  = "darkgreen", fill_down  = "darkred") +
     labs(title = "S&P 500 Candlestick Chart",
          subtitle = "Con un zoom 6 semanas",
          y = "Precio de Cierre", x = "") +
     coord_x_date(xlim = c(end - weeks(6), end),
                  ylim = c(2500, 3000)) +
     theme_tq()
```


S&P 500 Candlestick Chart
Con un zoom 6 semanas
Precio de Cierre
3000
2900
2800
2700
2600
2500
jul 01
jul 15
ago 01

Bandas de Bollinger

El análisis Técnico se divide en cuatro tipo: Tendencia (*Trend*), Volatilidad (*Volatility*), Momentum (*Momentum*) y Volumen (*Volume*). Uno de los más utilizados son las bandas de Bollinger, las que son formadas por tres líneas.

- La línea central (*Middle Line*, ML) es una media móvil simple.

$$SMA = \frac{P_M + P_{M-1} + ... + P_{M-(n-1)}}{n} = ML$$

- La línea superior (*Top Line*, TL) es la misma línea central pero desplazada hacia arriba a un número determinado de desviaciones estándares (D).

$$TL = ML + (D * \sigma)$$

- La línea inferior (*Bottom Line*, BL) es la línea central desplazada hacia abajo al mismo número de desviaciones estándares.

$$BL = ML - (D * \sigma)$$

Bandas de Bollinger con quantmod

Realizar las Bandas no es difícil:

```
# Bandas de Bollinger
chartSeries(GSPC, subset = "last 3 months")
addBBands()
```

addBBands tiene las siguientes opciones

```
addBBands(n = 20, sd = 2, ma= "SMA", draw = "bands", on = -1)
```

Donde `n` es el numero de periodo de la media movil, `sd` las desviaciones estandar y `ma` el tipo de media movil.

Otros tipos de Análisis en quantmod

Tendencia:

INDICATOR	TTR NAME	QUANTMOD NAME
Welles Wilder's Directional Movement Indicator	ADX	addADX
Double Exponential Moving Average	DEMA	addDEMA
Exponential Moving Average	EMA	addEMA
Simple Moving Average	SMA	addSMA
Parabolic Stop and Reverse	SAR	addSAR
Exponential Volume Weighted Moving Average	EVWMA	addEVWMA
Moving Average Convergence Divergence	MACD	addMACD
Triple Smoothed Exponential Oscillator	TRIX	addTRIX
Weighted Moving Average	WMA	addWMA
ZLEMA	ZLEMA	addZLEMA

Volatility:

INDICATOR	TTR NAME	QUANTMOD NAME
Average True Range	ATR	addATR
Bollinger Bands	BBands	addBBands
Price Envelope	N/A	addEnvelope

Volume:

INDICATOR	TTR NAME	QUANTMOD NAME
Chaiken Money Flow	CMF	addCMF
Volume	N/A	addVo

Momentum:

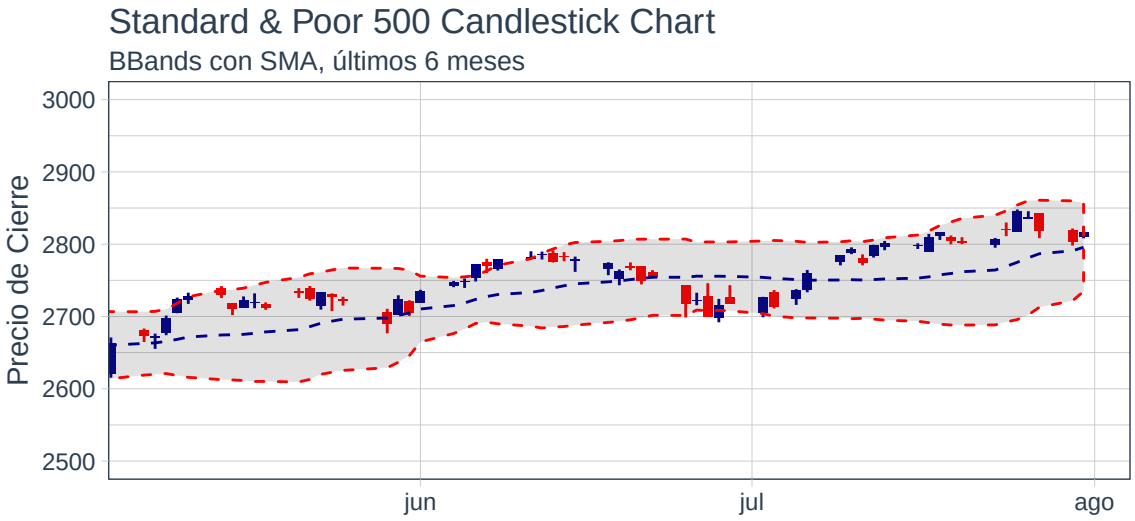
INDICATOR	TTR NAME	QUANTMOD NAME
Commodity Channel Index	CCI	addCCI
Chande Momentum Oscillator	CMO	addCMO
Detrended Price Oscillator	DPO	addDPO
momentum	addMomentum	
Rate of Change	ROC	addROC
Relative Strength Indicator	RSI	addRSI
Stocastic Momentum Index	SMI	addSMI
Williams %R	WPR	addWPR

Bandas de Bollinger con tidyquant

La librería `tidyquant` tiene el layer que puede ser usada en `ggplot2` al igual que: `geom_candlestick()`, `geom_barchart()` y `geom_bbands()`

```
end <- as_date("2018-07-31")
```

```
sp500_precio %>%
  ggplot(aes(x=date, y=close, open = open,
             high = high, low = low, close = close)) +
  geom_candlestick() +
  geom_bbands(ma_fun = SMA, sd = 2, n = 20) +
  labs(title = "Standard & Poor 500 Candlestick Chart",
       subtitle = "BBands con SMA",
       y = "Precio de Cierre", x = "") +
  coord_x_date(xlim = c(end - weeks(12), end),
               ylim = c(2500, 3000)) +
  theme_tq()
```


Standard & Poor 500 Candlestick Chart
BBands con SMA, últimos 6 meses
Precio de Cierre
3000
2900
2800
2700
2600
2500
jun
jul
ago

Usar la librería plotly en finanzas

Como ya saben existe la librería plotly la que nos permite construir gráficos dinamicos. A continuación les presentó lo útil que es para realizar análisis técnico:

1. Ejemplo plotly

Recomendación

Cada librería es un mundo nuevo, si quiere aprender más de plotly y su relación con otros lenguajes de programación visiten:
1. plotly for R

Referencias

Referencias

[1] H. Wickham and G. Grolemund. *R for data science: import, tidy, transform, visualize, and model data*. " O'Reilly Media, Inc.", 2016.
[2] H. Wickham and others. "Tidy data". In: *Journal of Statistical Software* 59.10 (2014), pp. 1-23.

Introducción a tidyquant

Llevando el análisis financiero al tidyverse

Gabriel Cabrera

16 de agosto de 2018 (2018-09-18)