


# Introducción a tidyquant


## Llevando el análisis financiero al tidyverse


Gabriel Cabrera

16 de agosto de 2018

# 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 coleccionar y analizar datos financieros: zoo, xts, quantmod, TTR y PerformanceAnalytics, con la infraestructura **tidy data**<sup>1</sup> del tidyverse, permitiendo la interacción entre ambos.
- 3 Nos permite implementar las funciones de **ggplot2**, para visualizaciones hermosas!!!

---

<sup>1</sup>Para mayor detalle ver el paper “Tidy Data (2014)” de Hadley Wickham.

# Breve resumen de las librerías

- ❶ 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.
- ❷ quantmod o Quantitative Financial Modelling & Trading Framework: Es una librería diseñada para recuperar, manipular y modelar datos cuantitativos financieros.
- ❸ TTR o Technical Trading Rules: Librería que incluye varias funciones para computar análisis técnico.
- ❹ 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

- ➊ Obtener data de diversas fuentes podemos usar `tq_get()`.
- ➋ Transmutar data: `tq_transmute()`.
- ➌ mutar data: `tq_mutate()`.
- ➍ 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 un 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 descargar. Cuando usamos `stock.prices`, se obtienen los datos desde Yahoo Finance. Si quisiera los *financial statements*, escribo `get = "financial"` y los descarga desde **Google Finance**.

# Ahora con quantmod

Podemos hacer lo 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`:

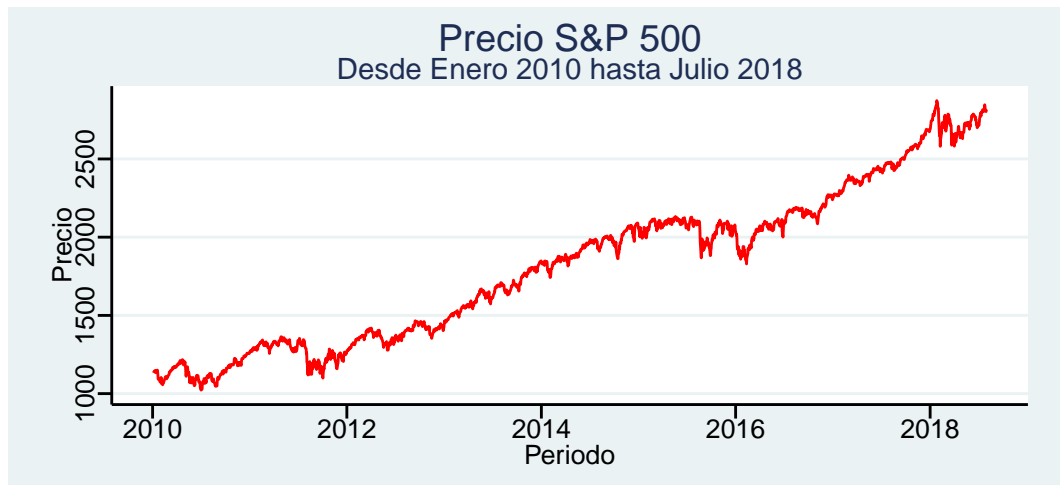
```
g <- ggplot(sp500_precio) + geom_line(aes(date,adjusted), color = "red")
g <- g + labs(title = "Precio S&P 500", subtitle = "Desde Enero 2010 hasta Julio 2018")
g <- g + theme_tq() + scale_color_tq()
g <- g + xlab("Periodo") + ylab("Precio")
g
```

# Precio S&P 500

Desde Enero 2010 hasta Julio 2018



Si extrañan stata ... pueden usar



Para realizarlo deben usar la librería ggthemes.

# Multiples Datos

# Descargando Múltiples Datos

Ahora veremos como podemos descargar más un activo. Usaremos las siguientes empresas; Oracle (ORCL), Intel (IT), Nvidia (NVDA) y Netflix (NFLX)

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

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

# Retornos y Retornos Acumulados

# Calculo Retornos

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



# Calculo Retornos Acumulados

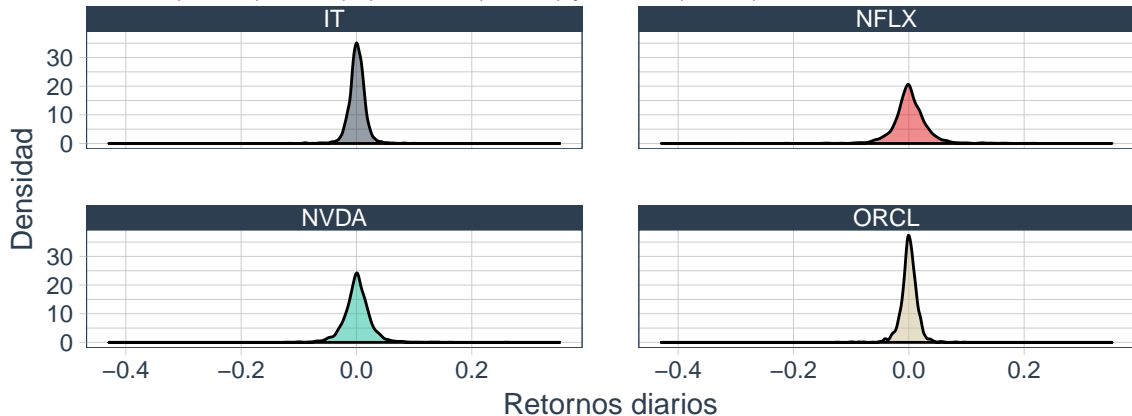
```
retornos_acum_activos <- retornos_activos %>%  
  group_by(symbol) %>%  
  mutate(ret.cum = cumsum(retornos.diarios))
```

# Gráficando Retornos

```
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)



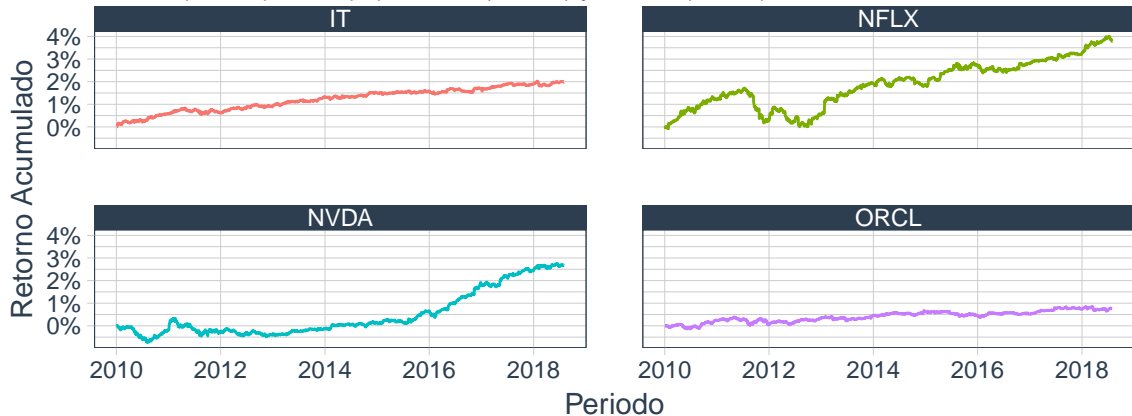
Activos:  IT  NFLX  NVDA  ORCL

# Gráficando Retornos Acumulados

```
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)



Activos: — IT — NFLX — NVDA — ORCL