Practical Data Science: Analyzing Stock Market Data with R

Downloading Stock Market Data from Google

Here we'll learn how to download free daily stock market data from Google. In the video you will seem me use Yahoo, but as of May 2017, that service doesn't work anymore. This is a simple but a critical lecture as we'll be relying on this data throughout the entire course.

In this lecture we'll look at:

- Installing quantmod
- Downloading market data
- Downloading multiple symbols at once

Installing Quantmod

The first thing we're going to do is install the quantmod package if you haven't already done so:

```
#install.packages('quantmod')
```

You can find more details about this unique package on Cran (https://cran.r-project.org/web/packages/quantmod/index.html).

It is described as ***Specify, build, trade, and analyze quantitative financial trading strategies*** on its package description. But two of its greatest features, in my opinion, aren't mentioned: free stock data and great visualization! Let's take a look. Quantmod can access a variety of data sources from databases to Google (as of May 2017 - Yahoo downloader doesn't work) It offers a lot of daily stock market data but you have to know the correct symbol. You can look them up on Google's Finance (https://www.google.com/finance) page (not all symbols work - some indexes don't, so use ETFs as proxies). For more details on additional data sources check out the Qunatmod Examples (http://www.quantmod.com/examples/intro/) page.

Getting Some SPY Data

To download 10 years worth of daily data for symbol SPY we simply call the getSymbols functions (don't forget to load the quantmod library first):

```
library(quantmod)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: TTR
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
getSymbols('SPY', src='google')
```

```
##     As of 0.4-0, 'getSymbols' uses env=parent.frame() and
##  auto.assign=TRUE by default.
##
##  This  behavior  will be  phased out in 0.5-0  when the call  will
##  default to use auto.assign=FALSE. getOption("getSymbols.env") and
##  getOptions("getSymbols.auto.assign") are now checked for alternate defaults
##
##  This message is shown once per session and may be disabled by setting
##  options("getSymbols.warning4.0"=FALSE). See ?getSymbols for more details.
```

```
## [1] "SPY"
```

That's it! You've just accessed 10 years worth of SPY stock market history. The object holding all this data is named after the symbol itself. Let's take a peek at the data by looking at the head and tail of SPY

```
head(SPY)
```

```
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume
## 2007-01-03   142.25   142.86  140.57    141.37   89183100
## 2007-01-04   141.23   142.05  140.61    141.67   66119200
## 2007-01-05   141.33   141.40  140.38    140.54   68523700
## 2007-01-08   140.82   141.41  140.25    141.19   66209700
## 2007-01-09   141.31   141.60  140.40    141.07   69505500
## 2007-01-10   140.58   141.57  140.30    141.54   67530600
```

```
tail(SPY)
```

```
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume
## 2017-05-12   239.09   239.43  238.67    238.98   53912730
## 2017-05-15   239.47   240.44  239.45    240.30   61918937
## 2017-05-16   240.64   240.67  239.63    240.08   51241791
## 2017-05-17   238.10   238.64  235.75    235.82  172174107
## 2017-05-18   235.73   237.75  235.43    236.77  107047656
## 2017-05-19   237.33   239.08  237.27    238.31  115011373
```

Even though the class type of this object is listed as both xts and zoo (time-series objects)

`getSymbols` returns xts and zoo classes:

```
class(SPY)
```

```
## [1] "xts" "zoo"
```

Casting it as a data frame is easy and will allow for much easier manipulation (a lot more about this later on). Here we generalize:

```
SPY <- data.frame(SPY)
summary(SPY)
```

```
##     SPY.Open         SPY.High        SPY.Low        SPY.Close
##  Min.   : 67.95   Min.   : 70.0   Min.   : 67.1   Min.   : 68.11
##  1st Qu.:124.85   1st Qu.:125.7   1st Qu.:123.7   1st Qu.:124.33
##  Median :145.12   Median :145.9   Median :144.1   Median :144.69
##  Mean   :155.02   Mean   :155.9   Mean   :154.0   Mean   :154.80
##  3rd Qu.:195.94   3rd Qu.:196.9   3rd Qu.:195.0   3rd Qu.:195.73
##  Max.   :240.64   Max.   :240.7   Max.   :239.6   Max.   :240.30
##  NA's   :19       NA's   :19      NA's   :19
##    SPY.Volume
##  Min.   :        0
##  1st Qu.: 53009073
##  Median : 89243750
##  Mean   :113965001
##  3rd Qu.:149734401
##  Max.   :814180365
##
```

Downloading Multiple Symbols at Once

We do the same thing as early except we pass getSymbols a vector of stock symbols:

```
basket_symbols <- c('YELP', 'AMZN', 'AAPL')
getSymbols(basket_symbols, src='google')
```

```
## [1] "YELP" "AMZN" "AAPL"
```

```
head(YELP)
```

```
##            YELP.Open YELP.High YELP.Low YELP.Close YELP.Volume
## 2012-03-02     22.01     26.00    22.00       24.58    17507620
## 2012-03-05     24.85     24.86    20.90       20.99     2994852
## 2012-03-06     19.83     20.50    19.36       20.50     1154290
## 2012-03-07     20.45     20.63    19.95       20.25      445454
## 2012-03-08     20.30     20.39    19.96       20.00      436871
## 2012-03-09     20.00     20.10    19.50       19.80      668201
```

```
head(AMZN)
```

```
##            AMZN.Open AMZN.High AMZN.Low AMZN.Close AMZN.Volume
## 2007-01-03     38.68     39.06    38.05       38.70    12441282
## 2007-01-04     38.59     39.14    38.26       38.90     6328615
## 2007-01-05     38.72     38.79    37.60       38.37     6623820
## 2007-01-08     38.22     38.31    37.17       37.50     6784631
## 2007-01-09     37.60     38.06    37.34       37.78     5703051
## 2007-01-10     37.49     37.70    37.07       37.15     6531890
```

```
head(AAPL)
```

```
##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume
## 2007-01-03     12.33     12.37    11.70       11.97   311433248
## 2007-01-04     12.01     12.28    11.97       12.24   214031636
## 2007-01-05     12.25     12.31    12.06       12.15   208817119
## 2007-01-08     12.28     12.36    12.18       12.21   199431337
## 2007-01-09     12.35     13.28    12.16       13.22   838036682
## 2007-01-10     13.54     13.97    13.35       13.86   739605951
```

This returns three separate objects each with its symbol's worth of data. To combine them into one data set (something we'll need to later on when we work with baskets):

```
basket <- data.frame(as.xts(merge(YELP, AMZN, AAPL)))
head(basket,2)
```

```
##            YELP.Open YELP.High YELP.Low YELP.Close YELP.Volume AMZN.Open
## 2007-01-03        NA        NA       NA         NA          NA     38.68
## 2007-01-04        NA        NA       NA         NA          NA     38.59
##            AMZN.High AMZN.Low AMZN.Close AMZN.Volume AAPL.Open AAPL.High
## 2007-01-03     39.06    38.05       38.7    12441282     12.33     12.37
## 2007-01-04     39.14    38.26       38.9     6328615     12.01     12.28
##            AAPL.Low AAPL.Close AAPL.Volume
## 2007-01-03    11.70       11.97   311433248
## 2007-01-04    11.97       12.24   214031636
```

```
tail(basket,2)
```

```
##            YELP.Open YELP.High YELP.Low YELP.Close YELP.Volume AMZN.Open
## 2017-05-18     27.41     28.12    27.11      27.99     3265230    944.80
## 2017-05-19     28.25     28.26    27.83      27.84     2731369    962.84
##            AMZN.High AMZN.Low AMZN.Close AMZN.Volume AAPL.Open AAPL.High
## 2017-05-18    962.75   944.76     958.49     3939347    151.27    153.34
## 2017-05-19    968.92   959.72     959.84     3972089    153.38    153.98
##            AAPL.Low AAPL.Close AAPL.Volume
## 2017-05-18   151.13     152.54    33568215
## 2017-05-19   152.63     153.06    26960788
```

We cast using as.xts to merge by time all our columns into one data frame. As you can see, YELP wasn't listed in 2007 so as.xts added NAs.

We'll revisit a lot of these functions throughout the class.