ArjunKoshalFinalProject.R

arjunkoshal

2020-12-17

```
# Step 1. Import all the packages that we will be utilizing in this program.

library(tidyquant)
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
## Loading required package: PerformanceAnalytics
```

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

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

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

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

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

```
## Loading required package: quantmod
```

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

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## == Need to Learn tidyquant? ============================================================
## Business Science offers a 1-hour course - Learning Lab #9: Performance Analysis & Portfolio Optimiza
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)

# Step 2. I decided to import the stock data directly from Yahoo finance, rather
# than creating 6 CSV files. I decided to use the internet to view how this
# would be possible and figured out the tidyquant package allows us to use the
# stock symbol directly from Yahoo finance. Quantmod stores the symbols with their
# own names, however, we can bypass that by setting the warnings to false.

options("getSymbols.warning4.0"=FALSE)
options("getSymbols.yahoo.warning"=FALSE)

# Step 3. Pick which stocks I want to display graphically. I decided to take
# the top 6 stocks from the S&P 500 and display them. I picked the time frame
# 2015 to 2020 as I felt that was the best option. At the bottom, I displayed the
# head of the prices and it was noticeable that only the Apple stock was being
# shown, therefore I had to change it so that we could see every stock,
# not only just the Apple Stock.

tickers = c("AAPL", "MSFT", "AMZN", "FB", "GOOGL", "TSLA")
prices <- tq_get(tickers,
           from = "2015-01-01",
           to = "2020-01-01",
           get = "stock.prices")
head(prices)
```

```
## # A tibble: 6 x 8
##   symbol date        open  high   low close    volume adjusted
##   <chr>  <date>     <dbl> <dbl> <dbl> <dbl>     <dbl>    <dbl>
## 1 AAPL   2015-01-02  27.8  27.9  26.8  27.3 212818400     24.9
## 2 AAPL   2015-01-05  27.1  27.2  26.4  26.6 257142000     24.2
## 3 AAPL   2015-01-06  26.6  26.9  26.2  26.6 263188400     24.2
```

```
## 4 AAPL   2015-01-07  26.8  27.0  26.7  26.9 160423600     24.5
## 5 AAPL   2015-01-08  27.3  28.0  27.2  28.0 237458000     25.5
## 6 AAPL   2015-01-09  28.2  28.3  27.6  28.0 214798000     25.5
```

```
# Step 4. I decided to group the stock data by each different company, as that was
# definitely the clearest way of showing all the data. The slice function lets
# us view the first row of each different stock (or symbol in this case).

prices %>%
  group_by(symbol) %>%
slice(1)
```

```
## # A tibble: 6 x 8
## # Groups:   symbol [6]
##   symbol date        open  high   low close    volume adjusted
##   <chr>  <date>     <dbl> <dbl> <dbl> <dbl>     <dbl>    <dbl>
## 1 AAPL   2015-01-02  27.8  27.9  26.8  27.3 212818400     24.9
## 2 AMZN   2015-01-02 313.  315.  307.  309.    2783200    309.
## 3 FB     2015-01-02  78.6  78.9  77.7  78.4  18177500     78.4
## 4 GOOGL  2015-01-02 533.  536.  528.  530.    1324000    530.
## 5 MSFT   2015-01-02  46.7  47.4  46.5  46.8  27913900     41.5
## 6 TSLA   2015-01-02  44.6  44.7  42.7  43.9  23822000     43.9
```

```
# Step 5.Plot all the stocks in different colors, based on their respective symbol
# I decided to plot based on full year ($Y) as it was the most clear and concise
# method. I needed to use the facet_wrap function as we learned in class to make
# the data fit to their stock prices. Because the Amazon stock was in the thousands
# range, and Microsoft was in the hundreds range, the function facet_wrap allowed
# the y axis to alter among the different visualizations.

prices %>%
  ggplot(aes(x=date,y=adjusted,color=symbol)) +
  geom_line() +
  facet_wrap(~symbol,scales='free_y') +
  theme_classic() +
  labs(x='Date (in Years)',
       y="Adjusted Price ($)",
       title="6 Stocks in the S&P 500",
       subtitle="Chart of the Prices for the Past 5 Years") +
  scale_x_date(date_breaks="year",
               date_labels="%Y") +
  labs(color="Stock Name") +
  scale_color_manual(name="Stock Name",
                     labels=c("Apple",
                              "Amazon",
                              "Facebook",
                              "Google",
                              "Microsoft",
                              "Tesla"),
                     values=c("AAPL"="red",
                              "AMZN"="orange",
                              "FB"="yellow",
                              "GOOGL"="green",
```

```
                                "MSFT"="blue",
                                "TSLA"="purple"))
```