Post 2

Arleen Wang

12/2/2017

How to use R to analyze stock market

Use R packages quantmod and plotly to anaylze stock price of Tesla

Introduction

Until now, we have learned serveral ways to draw graphs to anaylze the nba data. R is a practical language we could use to anaylze the things happened in the real life. For this post, I choose to use R packages quantmod and plotly to analyze the stock price of Tesla. In addition, I practice what we learned about basic graphics such as plot. The reason why I chosed to anaylze stock price is I want to become a financial analyst one day, and I want to practice what we learned in class to real life.

1,Getting TESLA stock data

There are two ways of getting TESLA stock data. The first way we can do is to use what we've learned in class about readr package to read the data that we could download "TSLA-2.csv" file from the yahoo finance website. Another way we could do is to use quantmod package to import the data straightly from Yahoo Finance.

```
#use readr to get data
library(readr)
dat_tesla <- read_csv('/Users/zilingwang/Documents/stat133/stat133-hws-fall17/post2/TSLA.csv',col_types=cols(
  Date = col_date(format = ""),
  Open = col_double(),
  High = col_double(),
  Low = col_double(),
  Close = col_double(),
  `Adj Close` = col_double(),
  Volume = col_integer()
))
dat_tesla
```

```
## Warning in format.POSIXlt(as.POSIXlt(x), ...): unknown timezone 'zone/tz/
## 2017c.1.0/zoneinfo/America/Los_Angeles'
```

```
## # A tibble: 20 x 7
##          Date   Open   High    Low  Close `Adj Close`   Volume
##        <date>  <dbl>  <dbl>  <dbl>  <dbl>       <dbl>    <int>
##  1 2017-11-03 299.50 306.25 295.13 306.09      306.09  8894000
##  2 2017-11-06 307.00 307.50 299.01 302.78      302.78  6486000
##  3 2017-11-07 301.02 306.50 300.03 306.05      306.05  5294300
##  4 2017-11-08 305.50 306.89 301.30 304.39      304.39  4725300
##  5 2017-11-09 302.50 304.46 296.30 302.99      302.99  5447100
##  6 2017-11-10 302.50 308.36 301.85 302.99      302.99  4625400
##  7 2017-11-13 300.13 316.80 299.11 315.40      315.40  7584900
##  8 2017-11-14 315.00 316.35 306.90 308.70      308.70  5676100
##  9 2017-11-15 306.01 312.49 301.50 311.30      311.30  5978700
## 10 2017-11-16 313.99 318.14 311.30 312.50      312.50  5822100
## 11 2017-11-17 325.67 326.67 313.15 315.05      315.05 13735100
## 12 2017-11-20 313.79 315.50 304.75 308.74      308.74  8247700
## 13 2017-11-21 310.86 318.23 308.71 317.81      317.81  7261300
## 14 2017-11-22 316.77 317.42 311.84 312.60      312.60  4917600
## 15 2017-11-24 313.79 316.41 311.00 315.55      315.55  3244100
## 16 2017-11-27 313.25 317.34 309.51 316.81      316.81  4555900
## 17 2017-11-28 316.36 320.00 313.92 317.55      317.55  4949500
## 18 2017-11-29 317.30 318.00 301.23 307.54      307.54  8767400
## 19 2017-11-30 308.56 310.70 304.54 308.85      308.85  4351600
## 20 2017-12-01 305.44 310.32 305.05 306.53      306.53  4271100
```

```
#use quantomod to get data
library(quantmod)
```

```
## Warning: package 'quantmod' was built under R version 3.4.2
```

```
## 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("TSLA", src = "yahoo")
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be 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 details.
```

```
##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).
```

```
## [1] "TSLA"
```

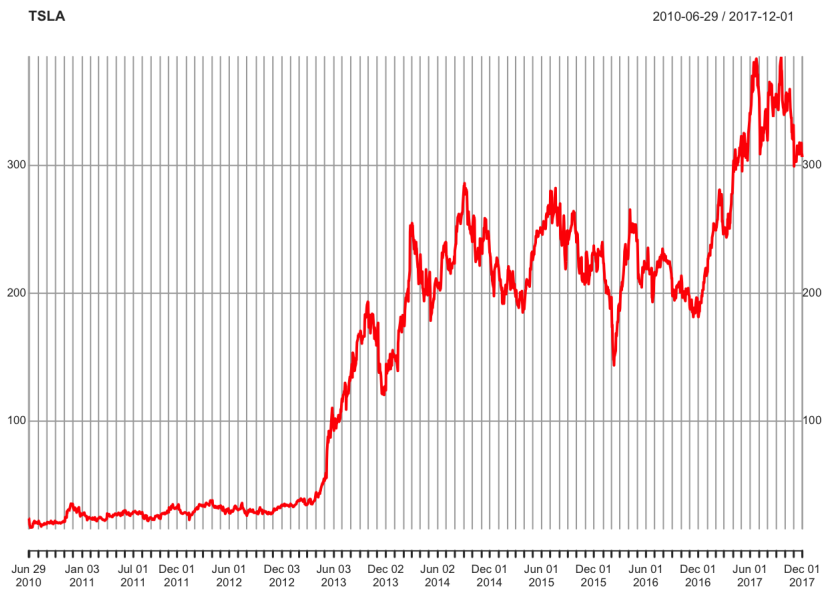
```
tail(TSLA)
```

```
##            TSLA.Open TSLA.High TSLA.Low TSLA.Close TSLA.Volume
## 2017-11-24    313.79    316.41   311.00     315.55     3244100
## 2017-11-27    313.25    317.34   309.51     316.81     4555900
## 2017-11-28    316.36    320.00   313.92     317.55     4949500
## 2017-11-29    317.30    318.00   301.23     307.54     8767400
## 2017-11-30    308.56    310.70   304.54     308.85     4351600
## 2017-12-01    305.44    310.32   305.05     306.53     4271100
##            TSLA.Adjusted
## 2017-11-24        315.55
## 2017-11-27        316.81
## 2017-11-28        317.55
## 2017-11-29        307.54
## 2017-11-30        308.85
## 2017-12-01        306.53
```

2,Visualizing Stock Data

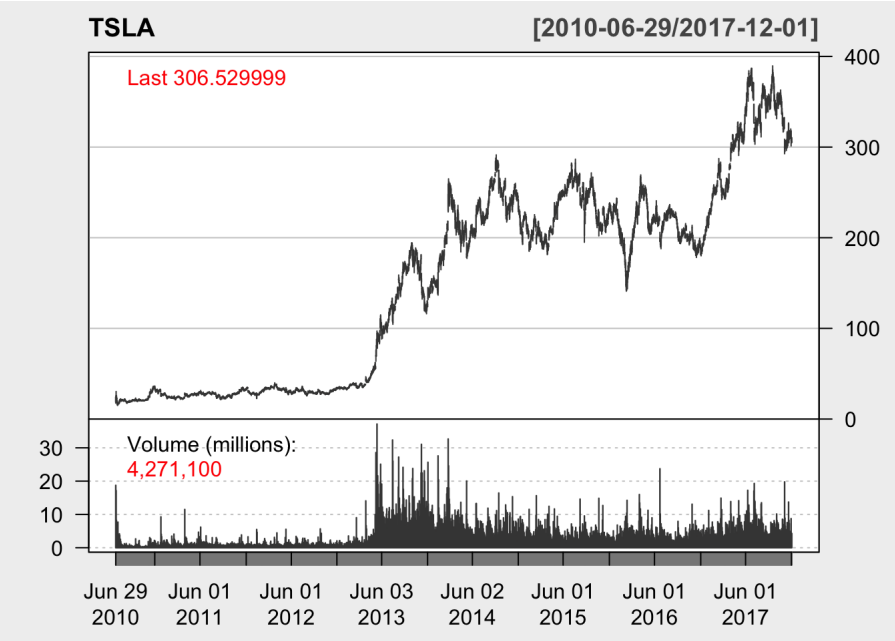
1)use base R plotting to visualize the series.

```
#use base R plotting to visualize the series.
plot(TSLA[, "TSLA.Close"], main = "TSLA",col="red")
```



2)The graph above we got are not accurate since there are at least four variables involved for each date (open, high, low, and close), and we would like to have some visual way to see all four variables that does not require plotting four separate lines.

```
candleChart(TSLA, up.col = "red", dn.col = "green", theme = "white")
```



3)There is a one of famous chart in financial market called Japanese candlestick plot and it was named because it was first created by 18th century Japanese rice traders. Use the function plot_ly() from quantmod to create such a chart. And this chart is much useful, since we could get the data from the chart based on date. Take October 3rd as an exmaple, we could immediately know that the price for open, close, high, and low are: 335.9, 348.55,331.28, and 348.14.

```
library(plotly)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

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

```
getSymbols("TSLA", src = "yahoo")
```

```
## [1] "TSLA"
```

```
# basic example of ohlc charts
df <- data.frame(Date=index(TSLA),coredata(TSLA))
df <- tail(df, 30)

p <- df %>%
  plot_ly(x = ~Date, type="candlestick",
          open = ~TSLA.Open, close = ~TSLA.Close,
          high = ~TSLA.High, low = ~TSLA.Low) %>%
  layout(title = "Basic Candlestick Chart")
p
```

4)Graph about stock split There is no split history in TESLA. If there exits a stock split there should be a sharp jump or a break down point.

```
df <- data.frame(Date=index(TSLA),coredata(TSLA))

# annotation
a <- list(text = "Stock Split",
          x = '2017-09-06',
          y = 1.02,
          xref = 'x',
          yref = 'paper',
          xanchor = 'left',
          showarrow = FALSE
)

# use shapes to create a line
l <- list(type = line,
          x0 = '2017-09-06',
          x1 = '2017-09-06',
          y0 = 0,
          y1 = 1,
          xref = 'x',
          yref = 'paper',
          line = list(color = 'black',
                      width = 0.5)
)

p <- df %>%
  plot_ly(x = ~Date, type="candlestick",
          open = ~TSLA.Open, close = ~TSLA.Close,
          high = ~TSLA.High, low = ~TSLA.Low) %>%
  layout(title = "TESLA Stock",
         annotations = a,
         shapes = l)
p
```

5)Add a trace to the candlestick chart

```
library(plotly)
library(quantmod)

getSymbols("TSLA",src='yahoo')
```

```
## [1] "TSLA"
```

```
df <- data.frame(Date=index(TSLA),coredata(TSLA))
df <- tail(df, 365)

p <- df %>%
  plot_ly(x = ~Date, type="candlestick",
          open = ~TSLA.Open, close = ~TSLA.Close,
          high = ~TSLA.High, low = ~TSLA.Low) %>%
  add_lines(y = ~TSLA.Open, line = list(color = 'black', width = 0.75)) %>%
  layout(showlegend = FALSE)
p
```

```
## Warning: 'scatter' objects don't have these attributes: 'open', 'close', 'high', 'low'
## Valid attributes include:
## 'type', 'visible', 'showlegend', 'legendgroup', 'opacity', 'name', 'uid', 'ids', 'customdata', 'hoverinfo', 'ho
verlabel', 'stream', 'x', 'x0', 'dx', 'y', 'y0', 'dy', 'text', 'hovertext', 'mode', 'hoveron', 'line', 'connectgap
s', 'cliponaxis', 'fill', 'fillcolor', 'marker', 'textposition', 'textfont', 'r', 't', 'error_y', 'error_x', 'xaxi
s', 'yaxis', 'xcalendar', 'ycalendar', 'idssrc', 'customdatasrc', 'hoverinfosrc', 'xsrc', 'ysrc', 'textsrc', 'hove
rtextsrc', 'textpositionsrc', 'rsrc', 'tsrc', 'key', 'set', 'frame', 'transforms', '_isNestedKey', '_isSimpleKey',
'_isGraticule'
```

Conclusion

For this post, I first talked about how to import data then I displayed five useful graph to anyalze the TESLA stock history. And I learned a lot about quantmod and plotly packages.

References

1. `https://finance.yahoo.com/quote/TSLA/history/`
2. `https://ntguardian.wordpress.com/2017/03/27/introduction-stock-market-data-r-1/`
3. `https://plot.ly/r/candlestick-charts/`
4. `https://www.msn.com/en-us/money/stockdetails?symbol=US:TSLA`
5. `http://www.investopedia.com/articles/technical/02/121702.asp`
6. `http://www.investopedia.com/terms/s/stocksplit.asp`
7. `https://www.splithistory.com/tsla/`