L19. Time Series Data

Sim, Min Kyu, Ph.D., mksim@seoultech.ac.kr

SEOULTECH 서울과학기술대학교 데이터사이언스학과

xts - 시계열 데이터를 위한 data.frame

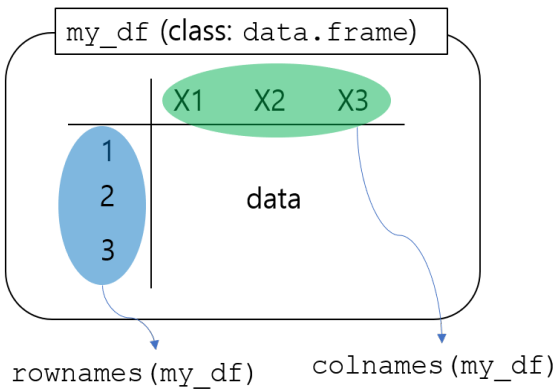
dygraph - xts 객체에 적합한 시각화 도구

lubridate - 날짜, 시각등의 Data Type을 다루는 패키지

xts - 시계열 데이터를 위한 data.frame

시계열 데이터

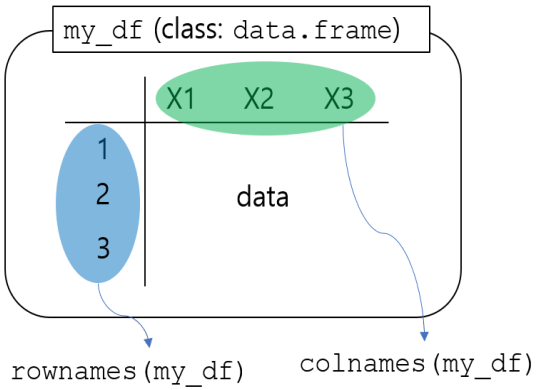
- 시간에 따라서 값이 변하는 데이터
- 1개의 column이 시간에 대한 정보를 담고 있는 data.frame을 생각할 수 있음

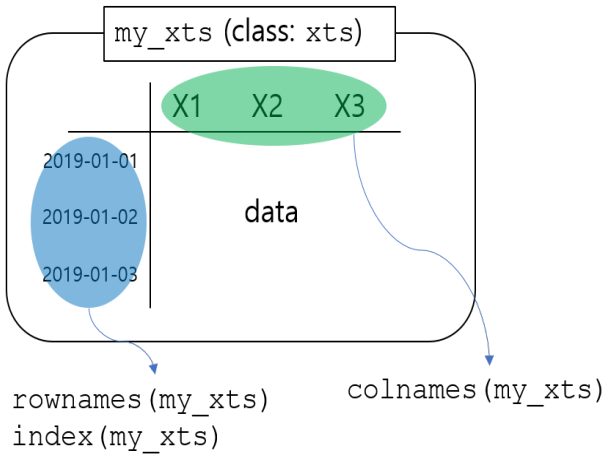


- X1 변수가 시간 정보를 담는다고 가정한다면, X1 변수는 X2, X3가 관찰된 시간을 의미하기 때문에 다른 변수와 성격이 다르다.
- (X1 변수는 다른 변수들의 **time index**에 해당하는 변수이다.)
- **각 변수들이 유사한 성격을 가지게 하기 위해서 rownames가 시간 정보를 담게 하는 것이 더 바람직하다.**

xts - time-series version of data.frame

- 자료구조 xts는 시계열 자료에 특화된 자료구조이다.
- 데이터프레임과 유사하면서 rownames가 시간 정보를 담게한다.





- 시간을 이용해서 간단하게 indexing을 할 수 있다.
- ex) `my_xts$X2["2019-01-02"]` vs `my_df$X2[my_df$X1=="2019-01-02"]`
- data.frame과의 변환이 간단하다.

마이크로소프트 주가

```
library(Quandl) # `quantmod` is another library for financial data
Quandl.api_key("SD27xu59qZmj-YCnxwDm")
MSFT = Quandl("WIKI/MSFT")
```

```
class(MSFT)
```

```
## [1] "data.frame"
```

```
head(MSFT,3)
```

```
##         Date  Open   High   Low Close   Volume Ex-Dividend Split Ratio
## 1 2018-03-27 94.94 95.139 88.51 89.47 53704562           0           1
## 2 2018-03-26 90.61 94.000 90.40 93.78 55031149           0           1
## 3 2018-03-23 89.50 90.460 87.08 87.18 42159397           0           1
##   Adj. Open Adj. High Adj. Low Adj. Close Adj. Volume
## 1     94.94    95.139    88.51      89.47    53704562
## 2     90.61    94.000    90.40      93.78    55031149
## 3     89.50    90.460    87.08      87.18    42159397
```

- 날짜, 종가, 거래량만을 선택

```
colnames(MSFT)
```

```
##  [1] "Date"        "Open"        "High"        "Low"         "Close"
##  [6] "Volume"      "Ex-Dividend" "Split Ratio" "Adj. Open"   "Adj. High"
## [11] "Adj. Low"    "Adj. Close"  "Adj. Volume"
```

```
MSFT <- MSFT %>% select(Date, Price=`Adj. Close`, Volume=`Adj. Volume`)
head(MSFT, 3)
```

```
##         Date Price   Volume
## 1 2018-03-27 89.47 53704562
## 2 2018-03-26 93.78 55031149
## 3 2018-03-23 87.18 42159397
```

```
sapply(MSFT, class)
```

```
##      Date     Price    Volume
##    "Date" "numeric" "numeric"
```

- xts 객체는 xts()라는 함수를 이용해서 생성된다.
- (마치 data.frame() 객체는 data.frame()으로 생성하듯이)
- 위의 data.frame 객체를 xts 객체로 바꾸는 xts() 함수에는 어떤 argument가 들어가야 할까?

xts 객체의 생성

```
library(xts)
MSFT_xts <- xts(x = MSFT[,-1], order.by = MSFT[,1])
```

1. x = MSFT[,-1]: MSFT 객체의 Date 컬럼을 제외한 main body를 넣어준다.
2. order.by = MSFT[,1]: xts 객체는 시간으로 indexing 한다. 그러므로 시간에 해당하는 정보를 입력해준다.

```
head(MSFT_xts)
```

```
##                  Price     Volume
## 1986-03-13 0.06471998 1031788800
## 1986-03-14 0.06703141  308160000
## 1986-03-17 0.06818712  133171200
## 1986-03-18 0.06645355   67766400
## 1986-03-19 0.06529784   47894400
## 1986-03-20 0.06356427   58435200
```

data.frame vs xts

```
dim(MSFT)
```

```
## [1] 8076    3
```

```
head(MSFT, 3)
```

```
##         Date Price   Volume
## 1 2018-03-27 89.47 53704562
## 2 2018-03-26 93.78 55031149
## 3 2018-03-23 87.18 42159397
```

```
MSFT[MSFT$Date=="2018-03-20",] # subsetting
```

```
##         Date Price   Volume
## 6 2018-03-20 93.13 21787780
```

```
dim(MSFT_xts)
```

```
## [1] 8076    2
```

```
head(MSFT_xts, 3)
```

```
##                  Price     Volume
## 1986-03-13 0.06471998 1031788800
## 1986-03-14 0.06703141  308160000
## 1986-03-17 0.06818712  133171200
```

```
MSFT_xts["2018-03-20",] # subsetting
```

```
##             Price   Volume
## 2018-03-20 93.13 21787780
```

- xts 객체가 되면서 dimension이 줄었지만, 모든 정보를 포함하고 있다.
- data.frame의 subsetting에서 행번호인 '6'은 아무 의미없다.
- xts 객체에서는 subsetting이 더 직관적이다.
- 시계열 자료는 xts 객체로 변환하여 다루는 것이 좋다.

R vs Python

	R	Python
데이터 프레임	`data.frame`	`Pandas.DataFrame`
시계열 데이터	`xts`	`Pandas.Series`
그래프	`ggplot2, dygraph`	`matplotlib`

- Python의 Pandas 라이브러리는 R의 데이터 프레임에서 영감을 받았다고 한다.
- `data.frame`과 `xts`의 관계는 `Pandas.DataFrame`과 `Pandas.Series`의 관계와 유사하다.

xts → data.frame

- xts 객체를 data.frame 객체로 바꾸는 것도 당연히 가능하다.

```
MSFT_df <- data.frame(Date = index(MSFT_xts), MSFT_xts)
rownames(MSFT_df) <- NULL
class(MSFT_df)
```

```
## [1] "data.frame"
```

```
head(MSFT_df, 3)
```

```
##         Date      Price     Volume
## 1 1986-03-13 0.06471998 1031788800
## 2 1986-03-14 0.06703141  308160000
## 3 1986-03-17 0.06818712  133171200
```

dygraph - xts 객체에 적합한 시각화 도구

```
library(dygraphs)
dygraph(MSFT_xts$Price)
```

- 시계열 그래프는 x축이 시간으로 항상 고정되어 있기에 자유도가 낮다.
- 그래서 dygraph() 함수의 문법은 매우 쉽다.
- dygraphs for R: https://rstudio.github.io/dygraphs/

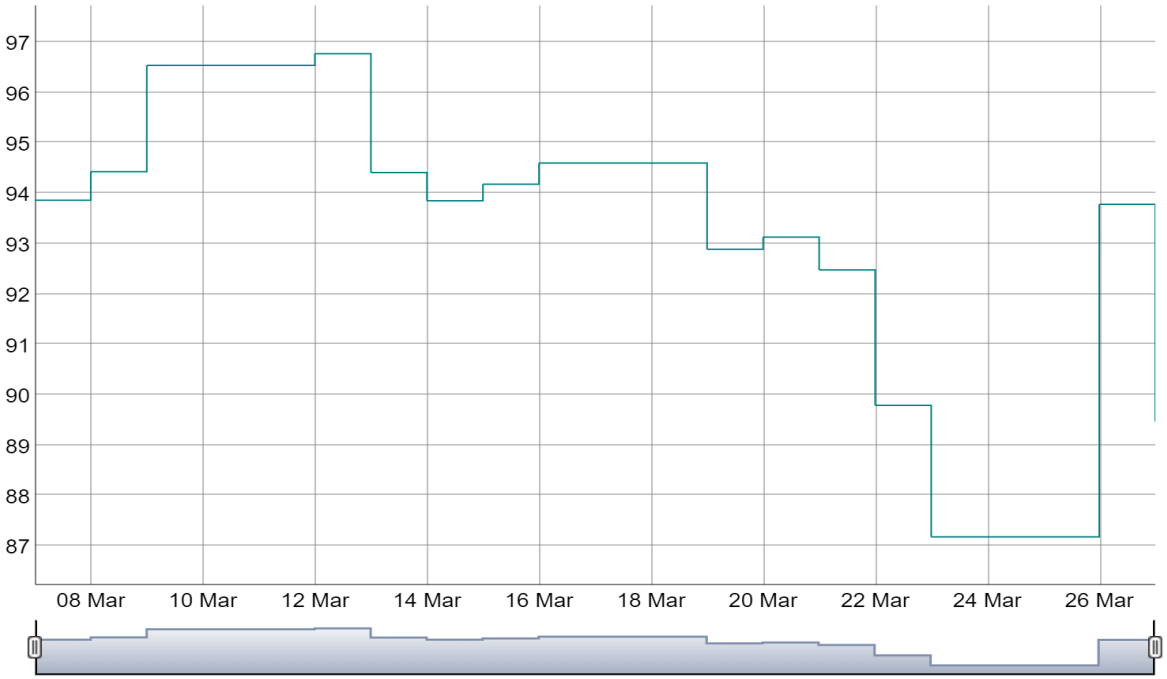
More features of dygraph

Range Selector

```
dygraph(MSFT_xts$Price) %>%
  dyRangeSelector(dateWindow = c("2015-01-01", "2017-12-31"))
```

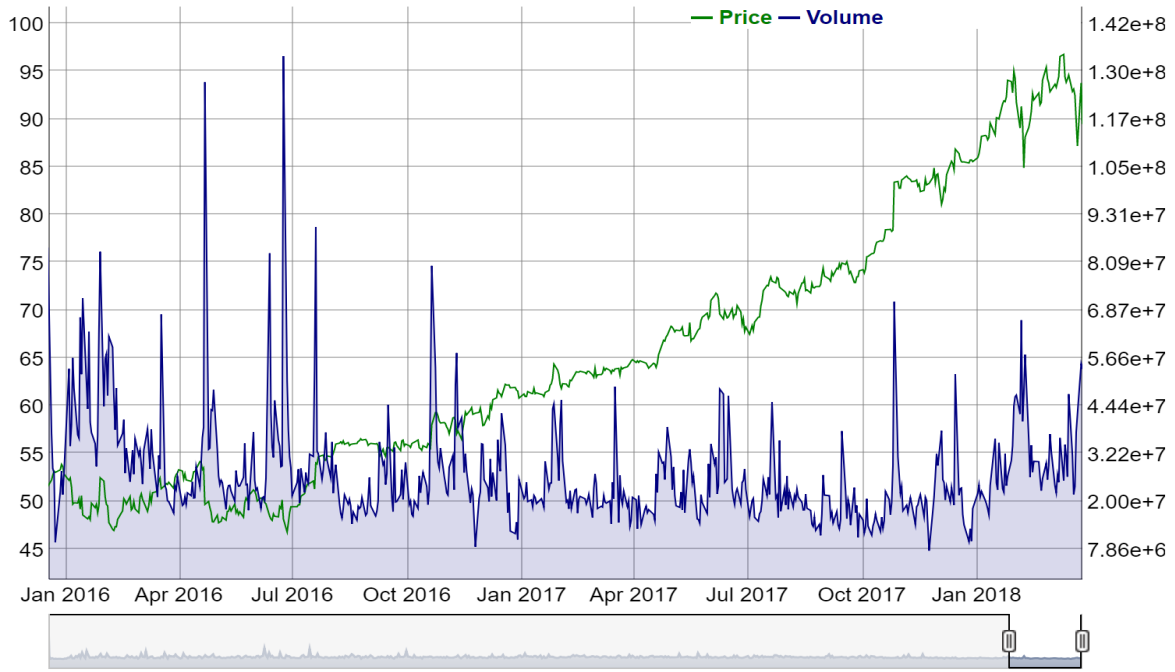
Step Plots

```
dygraph(MSFT_xts$Price %>% tail(15)) %>%
  dyOptions(stepPlot = TRUE) %>%
  dyRangeSelector()
```



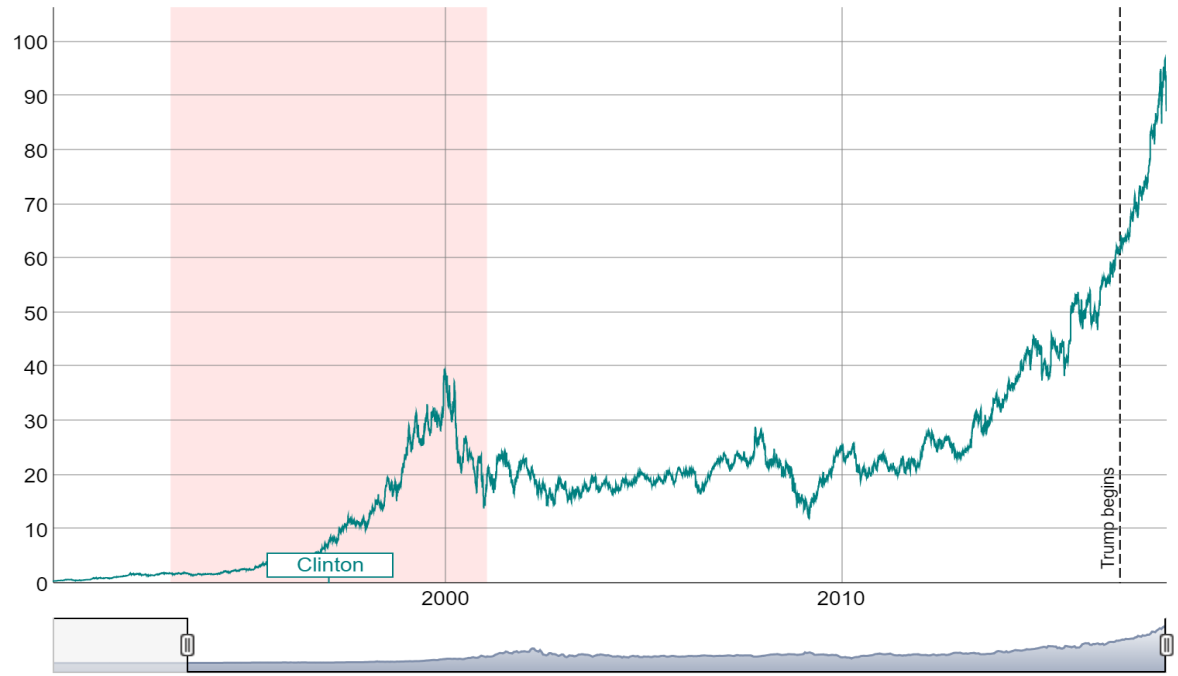
Second Y Axis & fillGraph

```
dygraph(MSFT_xts) %>%
  dySeries("Volume", axis = "y2", fillGraph = TRUE) %>%
  dyRangeSelector()
```



Shading & Annotation & Event

```
dygraph(MSFT_xts$Price) %>%
  dyShading(from = "1993-01-20", to = "2001-01-19", color = "#FFE6E6") %>%
  dyAnnotation("1997-01-20", text = "Clinton", attachAtBottom = TRUE, width = 80) %>%
  dyEvent("2017-01-20", "Trump begins", labelLoc = "bottom") %>%
  dyRangeSelector()
```



Discussion

- dygraph()의 구현
  - 구현이 매우 쉬움.
  - 추가 기능도 쉽게 구현 가능
  - 시계열 자료의 특성에 맞게 구현
- dygraph 객체 - htmlwidget
  - docx나 pdf로 deliver 하면 기능이 제한적
  - → html, flexdashboard, shiny, xaringan 등으로 deliver
- dygraphs for R: https://rstudio.github.io/dygraphs/

lubridate - 날짜, 시각등의 Data Type을 다루는 패키지

Motivation

- '2018-03-26'의 한달전은 언제일까?
- str_sub()로 '03'을 추출하여 numeric으로 바꾸어 2를 만들고 str_pad()로 0을 붙여서 해결한다?
- 그렇다면 '2018-01-15'의 한달전을 계산할 때에는?
- **적절한 패키지를 사용하는 것이 매우 중요하다.**

```
library(lubridate)
my_date <- as.Date("2018-03-26")
my_date + days(5)
```

```
## [1] "2018-03-31"
```

```
my_date - months(1)
```

```
## [1] "2018-02-26"
```

```
my_date + years(3)
```

```
## [1] "2021-03-26"
```

이번달 1일

```
floor_date(my_date, "month")
```

```
## [1] "2018-03-01"
```

저번달 말일

```
floor_date(my_date, "month") - days(1)
```

```
## [1] "2018-02-28"
```

전년 동월 말일

```
ceiling_date(my_date-years(1), "month") - days(1)
```

```
## [1] "2017-03-31"
```

lubridate를 활용한 조회

my_date의 주가

```
MSFT_xts[my_date,]
```

```
##            Price   Volume
## 2018-03-26 93.78 55031149
```

1개월전 주가

```
MSFT_xts[my_date-months(1),]
```

```
##            Price   Volume
## 2018-02-26 95.42 29760276
```

1년전 주가 ???

```
MSFT_xts[my_date-years(1),]
```

```
##      Price Volume
```

1년전 요일 확인

```
weekdays(my_date-years(1))
```

```
## [1] "일요일"
```

어떻게 확인??

- 시계열 데이터에서 흔히 비영업일, 비거래일에 대한 기록이 누락된다.
- tidyverse의 fill()과 같은 함수를 통해 365일로 만드는 방법 → 비영업일과 영업일을 구분하지 않으므로 또 다른 문제를 만든다.
- 데이터 마다 다르지만, 주가등 '직전 관찰값'을 이용하는 경우에는 아래의 접근법을 사용해야 한다.

직전 관찰값 조회 알고리즘

Issue again

```
MSFT_xts[my_date-years(1),]
```

```
##      Price Volume
```

Step 1. 해당 시점에서 가용한 데이터

```
available <- which(index(MSFT_xts) <= my_date-years(1))
tail(available)
```

```
## [1] 7819 7820 7821 7822 7823 7824
```

Step 2. 그 중에 가장 마지막 관찰값

```
MSFT_xts[max(available),]
```

```
##               Price   Volume
## 2017-03-24 63.95089 22617105
```

한줄로 정리하면

```
MSFT_xts[max(which(index(MSFT_xts) <= my_date-years(1))),]
```

```
##               Price   Volume
## 2017-03-24 63.95089 22617105
```

"勿輕小事 - 호랑이는 토끼를 사냥할 때에도 최선을 다한다."