

M44-subway (`tidyverse` + `ggmap`)

LearningSpoonsR

2018-07-19

I. 불러오기

```
raw <- read.csv("raw_subway.csv", stringsAsFactors = FALSE)
```

II. 전처리

UD는 승차와 하차를 구분하는 변수의 이름입니다.

```
colnames(raw) <-  
  c("Date", "Line", "Stn", "UD",  
    paste0("V", 0:23))
```

Stn에서 역이름을 나타내는 Name변수와 Code변수로 분리합니다.

```
raw$name <-  
  substr(raw$Stn,  
         start = 1,  
         stop = nchar(raw$Stn)-5)  
raw$Code <-  
  substr(raw$Stn,  
         start = nchar(raw$Stn)-3,  
         stop = nchar(raw$Stn)-1) %>%  
  as.numeric()
```

- 저는 항상 raw 데이터셋에서 시작하여 tidy한 `data.frame`을 만들면 그것에 이름을 `dataset`라고 붙여줍니다.
- 여러분도 각자의 규칙과 convention을 만들어 보세요.

1. Line변수를 제거합니다. (Code의 맨 앞자리 숫자가 Line과 같으므로 Line은 필요없다 판단해서 제거)
2. Stn변수는 이미 Name변수와 Code변수를 만들어 냈으므로 제거합니다.
3. 오전 1시- 오전 4시까지의 관찰값을 제외합니다.

```
raw <- raw %>%
  select(-c(Line, Stn)) %>% # 1) remove `Line` and `Stn`
  select(-c(V1, V2, V3))      # 2) remove `V1`, `V2`, `V3`
```

3. 4시-23시 사이에 빈 값이 있는 경우에 0으로 채웁니다.

```
# 3) fill NA with zero for `V0`, `V4-V23`
for (V in c("V0", paste0("V", 4:23))) {
  x <- raw[,V]
  x[is.na(x)] <- 0
  raw[,V] <- x
}
```

- 4. `tidyverse`의 `gather` 명령을 이용해서 `raw`를 `tidy`하게 바꿉니다! (`M51-tidyr` 참조)
- 5. `UD`로 `depArr`라는 변수를 만들어서 승차와 하차를 구분합니다.
- 6. 시간을 숫자로 바꿉니다.
- 7. `raw`가 잘 정리되어 이제 `dataset`이라는 명예로운 이름을 붙여줍니다. (저의 convention으로는 이것은 전처리 과정이 끝난 것을 의미합니다.)

```
# 4) `gather` in `table4a` in `M51`
raw <- raw %>%
  gather(colnames(raw)[3:23], key = "hour", value = "pax")
# 5) define `depArr` from `UD`, and kill `UD`
raw <- raw %>%
  mutate(depArr = ifelse(UD == " ", "dep", "arr")) %>%
  select(-UD)
# 6) make hour to number
raw$hour <- substr(raw$hour, 2, nchar(raw$hour)) %>% as.numeric
# 7) rearrange `raw` and now we have tidy `dataset`
dataset <- raw %>%
  select(Date, Code, Name, hour, depArr, pax) %>%
  arrange(Date, Code, Name, hour, depArr, pax)
```

- 데이터셋의 크기 관계로 수업 시간에 적합할 수준인 일별로 집계합니다.
- raw에서 dataset으로 만드는 과정의 프로그램 실행은 오래걸리는 경우가 많습니다.
- 그러므로 dataset을 잘 저장해두고 이후의 분석 코드에서는 dataset을 불러오면서 분석을 시행하는 것이 좋습니다.

```

dataset <- dataset %>%
  group_by(Date, Code, Name, depArr) %>% summarise(dayPax = sum(pax))
head(dataset)

## # A tibble: 6 x 5
## # Groups:   Date, Code, Name [3]
##   Date      Code Name    depArr dayPax
##   <chr>     <dbl> <chr>   <chr>   <dbl>
## 1 2014-01-01 150   arr     37200
## 2 2014-01-01 150   dep     44723
## 3 2014-01-01 151   arr     10199
## 4 2014-01-01 151   dep     12276
## 5 2014-01-01 152   arr     20284
## 6 2014-01-01 152   dep     26045
write.csv(dataset, "dataset_subway_daily.csv")

# dataset <- read.csv("dataset_subway_daily.csv", stringsAsFactors = FALSE)
dataset$Date   <- dataset$Date %>% as.Date()
dataset$dow    <- dataset$Date %>% weekdays() %>% as.factor()
dataset$depArr <- dataset$depArr %>% as.factor()

```

III. 간단한 분석

복습의 의미로 아래의 Task를 해보겠습니다.

Task 1. (dplyr) 2호선 역중에서 2014년 1년간 가장 승하차인원의 합이 많았던 역과 적었던 역 10개를 각각 찾아주세요.

Task 2. (ggplot2) 강남역의 요일별 이용승객수에 대해서 boxplot을 그려주세요.

Task 3. (ggplot2) 강남역의 요일별 이용승객수에 대해서 piechart를 그려주세요.

Task 4. (dygraph) 양재역의 승차인원과 하차인원에 대해서 각각 시계열 차트를 그려주세요. 신분당선의 강남-신사 구간이 2022년 1월에 연장 개통된다고 합니다.

III - Task 1

Task 1. (dplyr) 2호선 역중에서 2014년 1년간 가장 승하차인원의 합이 많았던 역과 적었던 역 10개를 각각 찾아주세요.

```
# 1)
task1_1 <- dataset %>%
  filter(substr(Code, 1, 1)==2) %>%
  group_by(Code, Name) %>% summarise(dayPaxTotal = sum(dayPax)) %>%
  arrange(desc(dayPaxTotal)) %>% head(10)
task1_1

## # A tibble: 10 x 3
## # Groups:   Code [10]
##   Code     Name    dayPaxTotal
##   <dbl>   <chr>      <dbl>
## 1 222      222      77844393
## 2 216      216      56624694
## 3 239      239      54640893
## 4 230      230      52967626
## 5 234      234      49067625
## 6 232      232      46642462
## 7 219      219      44429237
## 8 240      240      39945742
## 9 202      202      39481481
## 10 228     228      39308383
```

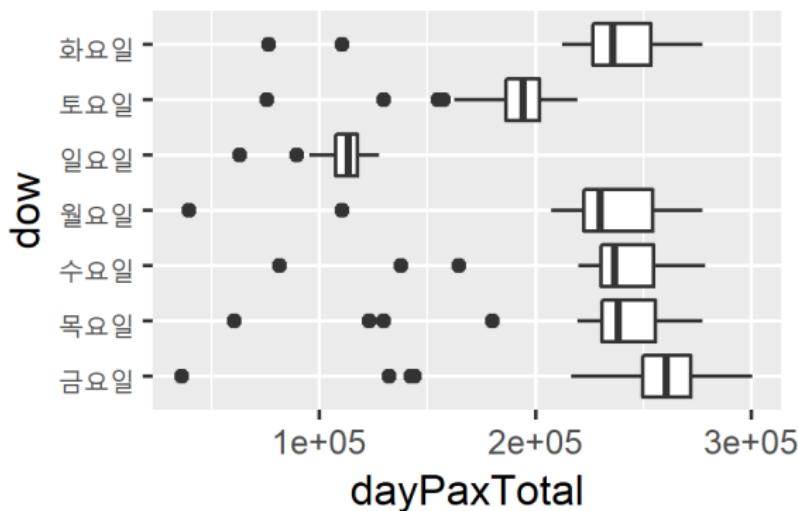
```
# 2)
task1_2 <- dataset %>%
  filter(substr(Code, 1, 1)==2) %>%
  group_by(Code, Name) %>% summarise(dayPaxTotal = sum(dayPax)) %>%
  arrange(desc(dayPaxTotal)) %>% tail(10)
task1_2

## # A tibble: 10 x 3
## # Groups:   Code [10]
##   Code Name     dayPaxTotal
##   <dbl> <chr>      <dbl>
## 1 243  9004863
## 2 249  8160885
## 3 207  7335310
## 4 242  6236475
## 5 248  5639797
## 6 246  2971661
## 7 244  1892476
## 8 250  1702836
## 9 245  1066912
## 10 247  736030
```

III - Task 2

2. (ggplot2) 강남역의 요일별 이용승객수에 대해서 boxplot을 그려주세요.

```
dataset2 <- dataset %>%
  filter(Code == 222) %>%
  group_by(Date, dow) %>% summarise(dayPaxTotal = sum(dayPax))
library(ggplot2)
a <- ggplot(dataset2) + geom_boxplot(aes(x = dow, y = dayPaxTotal)) +
  coord_flip()
ggsave(filename="task2.png", plot=a)
```



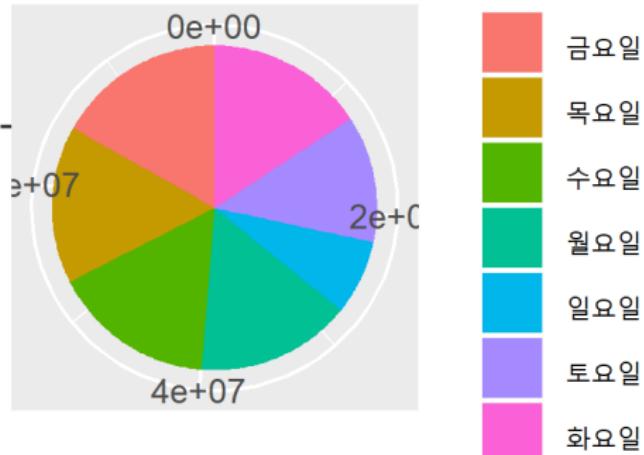
III - Task 3

3. (ggplot2) 강남역의 요일별 이용승객수에 대해서 piechart를 그려주세요.

```
a <- ggplot(dataset2, aes(x = "", y = dayPaxTotal, fill = factor(dow))) +
  geom_bar(width = 1, stat = "identity") +
  theme(axis.line = element_blank(),
        plot.title = element_text(hjust = 0.5)) +
  labs(fill = "dow", x = NULL, y = NULL, title = "Pie Chart of Station 222") +
  coord_polar(theta = "y", start = 0)
ggsave(filename="task3.png", plot=a)

## Saving 3 x 2 in image
```

Pie Chart of Station 222_{dow}



III - Task 4

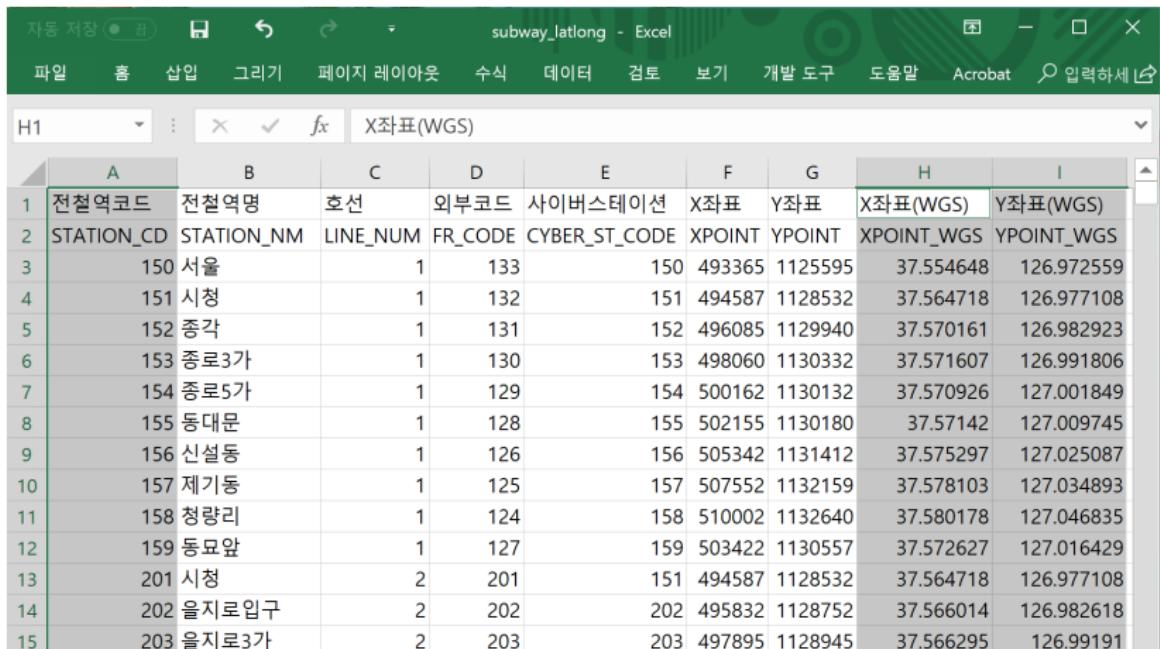
4. (dygraph) 양재역의 승차인원과 하차인원에 대해서 각각 시계열 차트를 그려주세요.
신분당선의 강남-신사 구간이 2022년 1월에 연장 개통된다고 합니다.

```
stn <- dataset %>%
  filter(Name == " ") %>%
  select(Date, Code, Name, depArr, dayPax)
library(xts)
library(dygraphs)
setLang("kr")
dep_stn <- stn %>% filter(depArr=="dep")
arr_stn <- stn %>% filter(depArr=="arr")
time_index <- stn %>% filter(depArr=="arr")
time_index <- time_index$Date
stn_xts <- xts(
  cbind(dep_stn$dayPax, arr_stn$dayPax),
  order.by = time_index)
colnames(stn_xts) <- c("dep_stn", "arr_stn")
dygraph(stn_xts) %>% dyRangeSelector()
```

```
head(stn_xts)
##          dep_stn arr_stn
## 2014-01-01 13052 13601
## 2014-01-02 46292 51013
## 2014-01-03 50812 54119
## 2014-01-04 30131 31015
## 2014-01-05 20097 21374
## 2014-01-06 49850 54408
```


IV. 지도 표시를 위한 준비

1. 위치 정보 불러오기



The screenshot shows an Excel spreadsheet titled "subway_latlong - Excel". The data is organized into columns: A (Station Code), B (Station Name), C (Line Number), D (Outer Code), E (Cyber Station Code), F (X Coordinate), G (Y Coordinate), H (X Coordinate in WGS84), and I (Y Coordinate in WGS84). The data includes 15 rows of subway station information.

	A	B	C	D	E	F	G	H	I
1	전철역코드	전철역명	호선	외부코드	사이버스테이션	X좌표	Y좌표	X좌표(WGS)	Y좌표(WGS)
2	STATION_CD	STATION_NM	LINE_NUM	FR_CODE	CYBER_ST_CODE	XPOINT	YPOINT	XPOINT_WGS	YPOINT_WGS
3	150	서울	1	133		150	493365	1125595	37.554648
4	151	시청	1	132		151	494587	1128532	37.564718
5	152	종각	1	131		152	496085	1129940	37.570161
6	153	종로3가	1	130		153	498060	1130332	37.571607
7	154	종로5가	1	129		154	500162	1130132	37.570926
8	155	동대문	1	128		155	502155	1130180	37.57142
9	156	신설동	1	126		156	505342	1131412	37.575297
10	157	제기동	1	125		157	507552	1132159	37.578103
11	158	청량리	1	124		158	510002	1132640	37.580178
12	159	동묘앞	1	127		159	503422	1130557	37.572627
13	201	시청	2	201		151	494587	1128532	37.564718
14	202	을지로입구	2	202		202	495832	1128752	37.566014
15	203	을지로3가	2	203		203	497895	1128945	37.566295

```
lat_lon <- read.csv("subway_latlong.csv", header = TRUE, skip = 1) %>%
  select(STATION_CD, STATION_NM, XPOINT_WGS, YPOINT_WGS)
```

```
head(lat_lon)

##   STATION_CD STATION_NM XPOINT_WGS YPOINT_WGS
## 1          330      37.49341  127.0141
## 2          331      37.48501  127.0162
## 3          332      37.48415  127.0346
## 4          333      37.48695  127.0468
## 5          334      37.49086  127.0554
## 6          335      37.49461  127.0636
```

2. 지도 불러오기 (서울)

```
library(ggmap)

## Google Maps API Terms of Service: http://developers.google.com/maps/terms.
## Please cite ggmap if you use it: see citation("ggmap") for details.

seoul_map <- get_map(
  location= c(lon= 126.9726, lat= 37.55465),
  zoom= 11,
  maptype= "roadmap")

## Source : https://maps.googleapis.com/maps/api/staticmap?center=37.55465,126.9726&zoom=11
seoul_map <- ggmap(seoul_map)
```

```
print(seoul_map)
```



V. 지도에 표시

승객많은 역 10개 (task1_1) 와 위치 정보(lat_lon)과를 결합하기 (참조: M51 join 부분)

```
map10 <-  
  left_join(x = task1_1, y = lat_lon, by = c("Code"="STATION_CD")) %>%  
  select(-STATION_NM)  
head(map10)  
  
## # A tibble: 6 x 5  
## # Groups:   Code [6]  
##   Code Name      dayPaxTotal XPOINT_WGS YPOINT_WGS  
##   <dbl> <chr>      <dbl>       <dbl>       <dbl>  
## 1    222          77844393     37.5       127.  
## 2    216          56624694     37.5       127.  
## 3    239          54640893     37.6       127.  
## 4    230          52967626     37.5       127.  
## 5    234          49067625     37.5       127.  
## 6    232          46642462     37.5       127.
```

지도위에 Bubble Chart

```
a <- seoul_map +
  # https://ggplot2.tidyverse.org/reference/geom_point.html
  geom_point(data = map10,
             aes(x = YPOINT_WGS, y = XPOINT_WGS,
                  size = dayPaxTotal/10000),
             color = "blue", alpha = 0.8) +
  # https://ggplot2.tidyverse.org/reference/geom_text.html
  geom_text(data = map10,
            aes(x = YPOINT_WGS, y = XPOINT_WGS,
                 label = paste(Name, "Stn"),
                 fontface = "italic",
                 vjust = 1.5),
            size = 2) +
  labs(title = "Top 10 Station for Dep & Arr in 2014 (Green line)")
ggsave(filename="map.png", plot=a)

## Saving 10 x 7 in image
```

Top 10 Station for Dep & Arr in 2014 (Green line)

