Subway

Jihyun

7/16/2019

날씨에 따른 서울 지하철 2호선 혼잡도 예상

1. 데이터 전처리

```
library(openxlsx)
library(dplyr)
```

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

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

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

지하철 데이터를 읽어옵니다.

2018년

```
# 2018년 데이터
subway_2018_raw <- read.xlsx("subway/subway_2018.xlsx", sheet = 1,startRow =
2, colNames = TRUE)
str(subway_2018_raw)
```

```
## 'data.frame':    200750 obs. of  26 variables:
##  $ date        : num  43101 43101 43101 43101 43101 ...
##  $ line        : chr  "1호선" "1호선" "1호선" "1호선" ...
##  $ station_code: num  150 150 151 151 152 152 153 153 154 154 ...
##  $ station_name: chr  "서울역" "서울역" "시청" "시청" ...
##  $ on_off      : chr  "승차" "하차" "승차" "하차" ...
##  $ 05.~.06     : num  373 205 87 47 604 74 282 48 54 33 ...
##  $ 06.~.07     : num  318 1040 105 294 399 219 211 128 61 83 ...
##  $ 07.~.08     : num  365 872 124 497 191 327 133 165 78 145 ...
##  $ 08.~.09     : num  785 984 197 1017 250 ...
##  $ 09.~.10     : num  1047 1650 291 673 370 ...
##  $ 10.~.11     : num  1576 1743 499 657 439 ...
##  $ 11.~.12     : num  2510 2175 722 820 705 ...
```

```
##  $ 12.~.13     : num  3233 2991 612 958 980 ...
##  $ 13.~.14     : num  3145 2877 580 1009 1153 ...
##  $ 14.~.15     : num  2443 2743 821 877 1392 ...
##  $ 15.~.16     : num  2980 2687 907 830 1710 ...
##  $ 16.~.17     : num  3476 2885 1027 781 2127 ...
##  $ 17.~.18     : num  3891 2845 1102 702 2172 ...
##  $ 18.~.19     : num  3227 2337 1278 552 2171 ...
##  $ 19.~.20     : num  2945 2131 1163 388 1873 ...
##  $ 20.~.21     : num  2382 1669 1032 308 1935 ...
##  $ 21.~.22     : num  3070 1404 975 236 2084 ...
##  $ 22.~.23     : num  1750 868 553 160 1458 ...
##  $ 23.~.24     : num  781 477 214 100 580 152 313 139 85 99 ...
##  $ 00.~.01     : num  96 147 9 39 28 33 16 35 8 28 ...
##  $ sum         : num  40393 34730 12298 10945 22621 ...
```

2018년도의 일별 2호선 승차 데이터를 분리해 냅니다.

```
subway_2018 <- subway_2018_raw %>%
  filter(on_off == '승차' & line == '2호선')
```

num 형태의 date를 date 형태로 변형하고, date 변수를 통해 day를 변수를 만들어 줍니다.

```
subway_2018$date <- convertToDate(subway_2018$date)# 43101 -> 2018-01-01
subway_2018$day <- weekdays(as.Date(subway_2018$date)) #2018-01-01 -> Monday
```

휴일과 아닌날을 구분하기 위해 2018년의 공휴일 리스트를 가져온 후, holiday 변수에 휴일과 아닌날을 구분해 줍니다.

```
holiday_2018 <- c('2018-01-01', '2018-02-15', '2018-02-16', '2018-02-17',
            '2018-03-01', '2018-05-05', '2018-05-22', '2018-06-06',
            '2018-06-13', '2018-05-07', '2018-05-06', '2018-05-01',
            '2018-08-15', '2018-09-23', '2018-09-24', '2018-09-26',
            '2018-09-25', '2018-10-03', '2018-10-09', '2018-12-25')

subway_2018$holiday <- ifelse(subway_2018$day %in% c('Saturday', 'Sunday') |
subway_2018$date %in% as.Date.character(holiday_2018), 'T', 'F')
```

오전 6시부터 10시, 오후 5시부터 9시까지 출퇴근 시간대로 구분하여 각 날의 출퇴근 인원을 rush_user변수에 담았습니다. 휴일에는 출퇴근 인원이 없다고 가정하여, notrush_user에 전체 인원을 넣었습니다.

```
rush_user <- ifelse(subway_2018[,28] == 'T',0,
rowSums(subway_2018[,c(7:10,18:21)]))
notrush_user <- ifelse(subway_2018[,28] == 'T',subway_2018[,26],
subway_2018[,26]-rush_user)
```

구한 rush_usre와 notrush_user를 subway_2018변수와 합쳐줍니다.

```
subway_2018 = cbind(subway_2018, rush_user)
subway_2018 = cbind(subway_2018, notrush_user)
head(subway_2018)
```

```
##         date  line station_code       station_name on_off 05.~.06 06.~.07
## 1 2018-01-01 2호선          201              시청    승차      37      57
## 2 2018-01-01 2호선          202         을지로입구    승차     128     116
## 3 2018-01-01 2호선          203          을지로3가    승차      42      79
## 4 2018-01-01 2호선          204          을지로4가    승차      24      41
## 5 2018-01-01 2호선          205 동대문역사문화공원    승차     123     112
## 6 2018-01-01 2호선          206              신당    승차     140     139
##   07.~.08 08.~.09 09.~.10 10.~.11 11.~.12 12.~.13 13.~.14 14.~.15 15.~.16
## 1      77     106     179     342     478     502     448     568     610
## 2     127     205     373     524     827    1116    1184    1468    1722
## 3      98     124     215     542     454     778     539     538     528
## 4      57      83     151     227     342     283     317     274     271
## 5     146     195     361     413     506     638     772     737     964
## 6     144     253     311     400     460     527     607     629     631
##   16.~.17 17.~.18 18.~.19 19.~.20 20.~.21 21.~.22 22.~.23 23.~.24 00.~.01
## 1     698     798     765     630     633     617     392     176       2
## 2    1798    2139    2478    2001    1862    2196    1804     863      13
## 3     545     619     539     427     367     342     237      98       0
## 4     308     296     247     194     139     126      78      37       3
## 5    1103     984     978     865     808     685     616     446       2
## 6     721     635     496     326     276     251     214     114       1
##     sum    day holiday rush_user notrush_user
## 1  8115 Monday       T         0         8115
## 2 22944 Monday       T         0        22944
## 3  7111 Monday       T         0         7111
## 4  3498 Monday       T         0         3498
## 5 11454 Monday       T         0        11454
## 6  7275 Monday       T         0         7275
```

시간대별로, 지하철 역 별로 나눠진 인원을 일자별로 합쳐줍니다.

```
subway_2018 = subway_2018 %>%
  group_by(date,holiday,day) %>%
  summarise(rush_user_tot= sum(rush_user), notrush_user_tot =
sum(notrush_user))
head(subway_2018)
```

```
## # A tibble: 6 x 5
## # Groups:   date, holiday [6]
##   date       holiday day       rush_user_tot notrush_user_tot
##   <date>     <chr>   <chr>             <dbl>            <dbl>
## 1 2018-01-01 T       Monday                0           704331
## 2 2018-01-02 F       Tuesday          922781           686791
## 3 2018-01-03 F       Wednesday        943062           722416
## 4 2018-01-04 F       Thursday         934506           742750
## 5 2018-01-05 F       Friday           977363           788476
## 6 2018-01-06 T       Saturday              0          1262856
```

앞으로의 분석을 쉽게 하기 위해서 변수명을 수정해 줍니다.

```
subway_2018 <- rename(subway_2018,
                      rush_user = rush_user_tot,
                      notrush_user = notrush_user_tot)
head(subway_2018)
```

```
## # A tibble: 6 x 5
## # Groups:   date, holiday [6]
##   date       holiday day       rush_user notrush_user
##   <date>     <chr>   <chr>         <dbl>        <dbl>
## 1 2018-01-01 T       Monday            0       704331
## 2 2018-01-02 F       Tuesday      922781       686791
## 3 2018-01-03 F       Wednesday    943062       722416
## 4 2018-01-04 F       Thursday     934506       742750
## 5 2018-01-05 F       Friday       977363       788476
## 6 2018-01-06 T       Saturday          0      1262856
```

정리된 2018년 지하철 데이터를 저장해 줍니다.

```
write.csv(subway_2018, file = 'subway_2018_rush.csv', row.names = F)
```

2017년

2017년 데이터는 10월달부터 데이터의 날짜 형식이 달라져서 분리해서 전처리 했습니다.

2017 중간에 날짜 데이터 형식 달라짐

date, total_user, day

subway_2017_raw <- read.xlsx("subway_2017.xlsx", sheet = 1,startRow = 2, colNames = TRUE) tail(subway_2017_raw) subway_2017 = subway_2017_raw %>% filter(on_off == '승차' & line == '2')

View(subway_2017)

2017 1월달~9월달

subway_2017_1to9 = subway_2017[0:13650,]
class(subway_2017_1to9$date) $subway_{2017_1}to9date$ = as.Date.character(subway_2017_1to9$date) View(subway_2017_1to9) head(subway_2017_1to9)

subway_2017_1to9$day = weekdays(as.Date(subway_{2017_1}to9date))$
subway_2017_1to9$date = as.Date.character(subway_{2017_1}to9date)$

2017 10월달~12월달

row <- nrow(subway_2017) subway_2017_10to12 = subway_2017[13651:row,] head(subway_2017_10to12$date)

subway_2017_10to12$date = convertToDate(subway_2017_10to12date) subway_2017_10to12$day = weekdays(as.Date(subway_2017_10to12date))

str(subway_2017_10to12)

두 데이터 합치기

subway_2017 = rbind(subway_2017_1to9, subway_2017_10to12) head(subway_2017) tail(subway_2017) for(i in 6:25){ subway_2017[,i] = as.integer(subway_2017[,i]) }

subway_2017$holiday = ifelse(subway_2017day %in% c('Saturday', 'Sunday') | subway_2017$date %in% as.Date.character(holiday_2017), 'T', 'F')

head(subway_2017) row = nrow(subway_2017)

rush_user = c(1,2) notrush_user= c(1,2)

class(subway_2017$sum) #6시부터 10시, 5시부터 9시 for(i in 1:row){ rush_user[i] = ifelse(subway_2017[i,28] == 'T',0, sum(subway_2017[i,c(7:10,18:21)])) notrush_user[i] = ifelse(subway_2017[i,28] == 'T',subway_2017[i,26], subway_2017[i,26]-rush_user[i]) } subway_2017 = cbind(subway_2017, rush_user) subway_2017 = cbind(subway_2017, notrush_user)

head(subway_2017)

subway_2017 = subway_2017 %>% group_by(date) %>% summarise(rush_user_tot= sum(rush_user), notrush_user_tot = sum(notrush_user))

head(subway_2017) colnames(subway_2017) = c('date','rush_user', 'notrush_user')

subway_2017

subway_2017$day = weekdays(as.Date(subway_2017date)) subway_2017$holiday = ifelse(subway_2017day %in% c('Saturday', 'Sunday') | subway_2017$date %in% as.Date.character(holiday_2017), 'T', 'F')

View(subway_2017) str(subway_2017) write.csv(subway_2017, file = 'subway_2017_rush.csv', row.names = F)

```
library(dplyr)
library(ggplot2)
```

하나로 합쳐보자

지하철 하나로 합치기

s2018 <- read.csv('subway2/subway_2018_rush.csv', header = T) s2017 <- read.csv('subway2/subway_2017_rush.csv', header = T)

subway_rush <- rbind(s2017,s2018) nrow(subway_rush)#730 View(subway_rush)

붐비는 단계 넣기

rush : 6시부터 10시, 6시부터 9시 (총 7 구간), 전체는 20구간

평균 탑승객 수를 넣어야 한다.

평일 : 2호선 rush : 266, notrush : 221

토요일 : 2호선 rush : 266, notrush : 221

일요일, 공휴일 : 2호선 rush : 266, notrush : 221

notholiday <-subway_rush %>% filter(holiday == FALSE) %>% mutate(mean_rush_user = round((rush_user)/266)) %>% mutate(mean_notrush_user = round(((notrush_user)/221)))

ggplot(data = notholiday, aes(x = date, y = mean_rush_user)) + geom_text(aes(label=date), size=4) ggplot(data = notholiday, aes(x = date, y = mean_notrush_user)) + geom_text(aes(label=date), size=4)

saturday <- subway_rush %>% filter(day == 'Saturday' & !(date %in% (holiday_2018)) & !(date %in% (holiday_2017))) %>% mutate(mean_rush_user = round((rush_user)/1)) %>% mutate(mean_notrush_user = round((notrush_user)/440))

ggplot(data = saturday, aes(x = date, y = mean_rush_user)) + geom_text(aes(label=date), size=4) ggplot(data = saturday, aes(x = date, y = mean_notrush_user)) + geom_text(aes(label=date), size=4)

redday <- subway_rush %>% filter(day == 'Sunday' | (date %in% (holiday_2018))| (date %in% (holiday_2017))) %>% mutate(mean_rush_user = round((rush_user)/1)) %>% mutate(mean_notrush_user = round((notrush_user)/389))

ggplot(data = redday, aes(x = date, y = mean_rush_user)) + geom_text(aes(label=date), size=4) ggplot(data = redday, aes(x = date, y = mean_notrush_user)) + geom_text(aes(label=date), size=4)

subway_rush <- rbind(notholiday,saturday,redday) nrow(subway_rush)

계절정보 넣기

봄

subway_rush$season[grepl("-03-", subway_rush$date)] <- 'spring'
subway_rush$season[grepl("-04-", subway_rush$date)] <- 'spring'
subway_rush$season[grepl("-05-", subway_rush$date)] <- 'spring'
여름 subway_rush$season[grepl("-06-", subway_rush$date)] <- 'summer'
subway_rush$season[grepl("-07-", subway_rush$date)] <- 'summer'
subway_rush$season[grepl("-08-", subway_rush$date)] <- 'summer' # 가을
subway_rush$season[grepl("-09-", subway_rush$date)] <- 'fall'
subway_rush$season[grepl("-10-", subway_rush$date)] <- 'fall'
subway_rush$season[grepl("-11-", subway_rush$date)] <- 'fall' # 겨울
subway_rush$season[grepl("-12-", subway_rush$date)] <- 'winter'
subway_rush$season[grepl("-01-", subway_rush$date)] <- 'winter'
subway_rush$season[grepl("-02-", subway_rush$date)] <- 'winter'

날씨정보 추가하기

비 데이터

rain <- read.csv('rain/rain.csv', header = T) View(rain) rain <- rain[,-1]# 지역코드 삭제
rain_simple <- rain[,c(1,4)] # date, 일 강수량만 사용

subway_rush <- merge(x = subway_rush, y = rain_simple, by ='date', all.x = TRUE) # 비 안온날 데이터 넣어주기 subway_rush$rain <- ifelse(is.na(subway_rush$rain), 0, subway_rush$rain)

눈 데이터

snow <- read.csv('snow/snow.csv', header = T) View(snow) snow <- snow[,-1]
subway_rush <- merge(x = subway_rush, y = snow, by ='date', all.x = TRUE)

눈 안온날 데이터 넣어주기

subway_rush$snow < -ifelse(is.na(subway_rush$snow), 0,
subway_rush$snow)subway_rush$newsnow <-
ifelse(is.na(subway_rush$newsnow),0, subway_rush$newsnow)

기온 데이터

temperature <- read.csv('temperature/temperature.csv', header = T) View(temperature)
temperature <- temperature[,-1] subway_rush <- merge(x = subway_rush, y = temperature,
by ='date', all.x = TRUE)

View(subway_rush)

습도

humid <- read.csv('humid/humid.csv', header = T) View(humid) humid <- humid[,-1]
subway_rush <- merge(x = subway_rush, y = humid, by ='date', all.x = TRUE)

바람

wind <- read.csv('wind/wind.csv', header = T) View(wind) wind <- wind[,-1] subway_rush
<- merge(x = subway_rush, y = wind, by ='date', all.x = TRUE)

저장하기

save(subway_rush, file="subway_merge2.RData")