서울 지하철 혼잡도 데이터 분석 보고서

박희선

2020년 3월 19일

1. 분석 목적

- 비교적 이용객 수가 높은 지하철 노선의 혼잡도 분석을 통하여 출퇴근 시간대에 더 편하게 갈 수 있는 노선명과 특정 역명 도출
- 출퇴근 시간대에 혼잡도가 높은 노선과 특정 역 분석
- 혼잡도 노선별, 역별 분석을 통한 혼잡도가 비교적 높은 특정 역 도출

2. 분석개요

- 분석 과제 : 지하철 혼잡도 분석
- 분석 데이터
    - 서울 열린데이터 광장(https://data.seoul.go.kr/)의 (https://data.seoul.go.kr/)의) 서울시 지하철 호선별 역별 승하차 인원 정보(2018~2020.02)
    - 서울 열린데이터 광장(https://data.seoul.go.kr/)의 (https://data.seoul.go.kr/)의) 서울시 지하철 호선별 역별 시간대별 승하차 인원 정보(2018~2020.02)
- 분석 도구 : R
- 분석 내용
    - 서울 지하철 시간별, 월별 데이터 분석
    - 서울 지하철 노선별, 역별 데이터 분석

3. 분석 과정

서울 지하철 혼잡도 분석, 즉, 어떤 특정한 역에서 승차했을 때 비교적 무리하게 타지 않아도 되는지를 분석해보았다. 이 분석을 하기 위해서, 서울 열린 데이터 광장에서 '서울시 지하철 호선별 역별 승하차 인원 정보'와 '서울시 지하철 호선별 역별 시간대별 승하차 인원 정보' 데이터들을 가져왔다. 데이터의 양이 방대하므로, 최근 2018년, 2019년, 2020년 2월까지의 데이터를 사용하였다.

```
# 데이터 불러오기
library(readr)

subway_202002_ = read_csv("CARD_SUBWAY_MONTH_202002.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_double(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_202001_ = read.csv("CARD_SUBWAY_MONTH_202001.csv")
subway_201912_ = read.csv("CARD_SUBWAY_MONTH_201912.csv")
subway_201911_ = read.csv("CARD_SUBWAY_MONTH_201911.csv")
subway_201910_ = read_csv("CARD_SUBWAY_MONTH_201910.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201909_ = read_csv("CARD_SUBWAY_MONTH_201909.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201908_ = read_csv("CARD_SUBWAY_MONTH_201908.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201907_ = read_csv("CARD_SUBWAY_MONTH_201907.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201906_ = read_csv("CARD_SUBWAY_MONTH_201906.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201905_ = read_csv("CARD_SUBWAY_MONTH_201905.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201904_ = read_csv("CARD_SUBWAY_MONTH_201904.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201903_ = read_csv("CARD_SUBWAY_MONTH_201903.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201902_ = read_csv("CARD_SUBWAY_MONTH_201902.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201901_ = read_csv("CARD_SUBWAY_MONTH_201901.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201812_ = read_csv("CARD_SUBWAY_MONTH_201812.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201811_ = read_csv("CARD_SUBWAY_MONTH_201811.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201810_ = read_csv("CARD_SUBWAY_MONTH_201810.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
subway_201809_ = read_csv("CARD_SUBWAY_MONTH_201809.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 17459 parsing failures.
## row col  expected    actual                          file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201809.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201809.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201809.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201809.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201809.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201808_ = read_csv("CARD_SUBWAY_MONTH_201808.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 18045 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201808.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201808.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201808.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201808.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201808.csv'
## ... ... ......... ......... ............................
## See problems(...) for more details.
```

```
subway_201807_ = read_csv("CARD_SUBWAY_MONTH_201807.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 18043 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201807.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201807.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201807.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201807.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201807.csv'
## ... ... ......... ......... ............................
## See problems(...) for more details.
```

```
subway_201806_ = read_csv("CARD_SUBWAY_MONTH_201806.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 17469 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201806.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201806.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201806.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201806.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201806.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201805_ = read_csv("CARD_SUBWAY_MONTH_201805.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 18056 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201805.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201805.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201805.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201805.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201805.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201804_ = read_csv("CARD_SUBWAY_MONTH_201804.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 17476 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201804.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201804.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201804.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201804.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201804.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201803_ = read_csv("CARD_SUBWAY_MONTH_201803.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 18024 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201803.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201803.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201803.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201803.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201803.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201802_ = read_csv("CARD_SUBWAY_MONTH_201802.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 16282 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201802.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201802.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201802.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201802.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201802.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_201801_ = read_csv("CARD_SUBWAY_MONTH_201801.csv")
```

```
## Parsed with column specification:
## cols(
##   사용일자 = col_double(),
##   노선명 = col_character(),
##   역ID = col_character(),
##   역명 = col_character(),
##   승차총승객수 = col_double(),
##   하차총승객수 = col_double(),
##   등록일자 = col_double()
## )
```

```
## Warning: 18008 parsing failures.
## row col  expected    actual                           file
##   1  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201801.csv'
##   2  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201801.csv'
##   3  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201801.csv'
##   4  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201801.csv'
##   5  -- 7 columns 8 columns 'CARD_SUBWAY_MONTH_201801.csv'
## ... ... ......... ......... ..............................
## See problems(...) for more details.
```

```
subway_time_ = read.csv("지하철 호선별 역별 시간대별 승하차 인원 정보.csv")
```

```
# 데이터 복사
subway_202002 <- subway_202002_
subway_202001 <- subway_202001_
subway_201912 <- subway_201912_
subway_201911 <- subway_201911_
subway_201910 <- subway_201910_
subway_201909 <- subway_201909_
subway_201908 <- subway_201908_
subway_201907 <- subway_201907_
subway_201906 <- subway_201906_
subway_201905 <- subway_201905_
subway_201904 <- subway_201904_
subway_201903 <- subway_201903_
subway_201902 <- subway_201902_
subway_201901 <- subway_201901_
subway_201812 <- subway_201812_
subway_201811 <- subway_201811_
subway_201810 <- subway_201810_
subway_201809 <- subway_201809_
subway_201808 <- subway_201808_
subway_201807 <- subway_201807_
subway_201806 <- subway_201806_
subway_201805 <- subway_201805_
subway_201804 <- subway_201804_
subway_201803 <- subway_201803_
subway_201802 <- subway_201802_
subway_201801 <- subway_201801_

subway_time <- subway_time_
```

데이터에서 '역ID'와 '등록일자'를 제외하였고, '지하철 호선별 역별 시간대별 승하차 인원 정보'에서 2018년, 2019년, 2020년의 데이터만 추출하였다.

```
# 필요한 열(변수), 행 추출
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
```

```
subway_202002 <- subway_202002 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_202001 <- subway_202001 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201912 <- subway_201912 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201911 <- subway_201911 %>% select(사용일자, 호선명, 역명, 승차총승객수, 하차총승객수)
subway_201910 <- subway_201910 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201909 <- subway_201909 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201908 <- subway_201908 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201907 <- subway_201907 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201906 <- subway_201906 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201905 <- subway_201905 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201904 <- subway_201904 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201903 <- subway_201903 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201902 <- subway_201902 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201901 <- subway_201901 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201812 <- subway_201812 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201811 <- subway_201811 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201810 <- subway_201810 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201809 <- subway_201809 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201808 <- subway_201808 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201807 <- subway_201807 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201806 <- subway_201806 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201805 <- subway_201805 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201804 <- subway_201804 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201803 <- subway_201803 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201802 <- subway_201802 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)
subway_201801 <- subway_201801 %>% select(사용일자, 노선명, 역명, 승차총승객수, 하차총승객수)

subway_time <- subway_time %>%
  filter(사용월 %in% c(202002, 202001, 201912, 201911, 201910, 201909, 201908, 201907, 201906,
201905, 201904, 201903, 201902, 201901, 201812, 201811, 201810, 201809, 201808, 201807, 20180
6, 201805, 201804, 201803, 201802, 201801))
```

```
# 2019년 11월 데이터 변수 변경
subway_201911 <- rename(subway_201911, 노선명 = 호선명)
```

개별 데이터를 쉽게 보기 위해서 '서울시 지하철 호선별 역별 승하차 인원 정보'의 데이터를 합치고 결측치를 확인하였다.

```
# 데이터 합치기
subway_month1 <- bind_rows(subway_201801, subway_201802, subway_201803, subway_201804, subway
_201805, subway_201806, subway_201807, subway_201808, subway_201809, subway_201810, subway_20
1811, subway_201812)

subway_month2 <- bind_rows(subway_201901, subway_201902, subway_201903, subway_201904, subway
_201905, subway_201906, subway_201907, subway_201908)

subway_month3 <- bind_rows(subway_201909, subway_201910, subway_201911, subway_201912, subway
_202001, subway_202002)
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

```
subway_month <- bind_rows(subway_month1, subway_month2, subway_month3)
```

```
# 결측치 확인
table(is.na(subway_month))
```

```
##
##   FALSE
## 2319740
```

```
table(is.na(subway_time))
```

```
##
##  FALSE
## 803088
```

노선이 26개이기 때문에 상대적으로 이용객 수가 많은 15개 노선을 선정하였다.

```
subway_line <- subway_month %>%
  group_by(노선명) %>%
  summarise(승차총승객수평균 = mean(승차총승객수),
                    하차총승객수평균 = mean(하차총승객수))

s1 <- subway_line %>% arrange(승차총승객수평균) %>% head(11)
s1
```

```
## # A tibble: 11 x 3
##    노선명         승차총승객수평균 하차총승객수평균
##    <chr>                    <dbl>            <dbl>
##  1 경춘선                   2041.            1957.
##  2 경강선                   2457.            2331.
##  3 장항선                   2529.            2405.
##  4 우이신설선               3270.            3158.
##  5 수인선                   3454.            3464.
##  6 경의선                   4030.            3908.
##  7 중앙선                   4635.            4495.
##  8 9호선2~3단계             6333.            6186.
##  9 경원선                   7358.            7122.
## 10 9호선2단계               8099.            7878.
## 11 공항철도 1호선           8170.            7509.
```

```
s2 <- subway_line %>% arrange(하차총승객수평균) %>% head(11)
s2
```

```
## # A tibble: 11 x 3
##    노선명         승차총승객수평균 하차총승객수평균
##    <chr>                    <dbl>            <dbl>
##  1 경춘선                   2041.            1957.
##  2 경강선                   2457.            2331.
##  3 장항선                   2529.            2405.
##  4 우이신설선               3270.            3158.
##  5 수인선                   3454.            3464.
##  6 경의선                   4030.            3908.
##  7 중앙선                   4635.            4495.
##  8 9호선2~3단계             6333.            6186.
##  9 경원선                   7358.            7122.
## 10 공항철도 1호선           8170.            7509.
## 11 9호선2단계               8099.            7878.
```

```
s3 <- bind_rows(s1, s2)

table(s3$노선명)
```

```
##
##   9호선2~3단계      9호선2단계          경강선          경원선          경의선
##              2               2               2               2               2
##         경춘선 공항철도 1호선          수인선      우이신설선          장항선
##              2               2               2               2               2
##         중앙선
##              2
```

```
# 필요한 행 추출
subway_month <- subway_month %>%
  filter(노선명 %in% c("과천선", "일산선", "안산선", "분당선", "경인선", "경부선","9호선", "8호
선", "7호선", "6호선", "5호선", "4호선", "3호선", "2호선", "1호선"))
str(subway_month)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 336350 obs. of  5 variables:
##  $ 사용일자    : num  20180101 20180101 20180101 20180101 20180101 ...
##  $ 노선명      : chr  "3호선" "3호선" "3호선" "3호선" ...
##  $ 역명        : chr  "녹번" "홍제" "무악재" "독립문" ...
##  $ 승차총승객수: num  4867 8751 1808 3402 3979 ...
##  $ 하차총승객수: num  4607 8938 2009 3810 3990 ...
```

```
subway_time <- subway_time %>%
  filter(호선명 %in% c("과천선", "일산선", "안산선", "분당선", "경인선", "경부선","9호선", "8호
선", "7호선", "6호선", "5호선", "4호선", "3호선", "2호선", "1호선")) %>%
  select("사용월", "호선명", "지하철역", "X06시.07시.승차인원", "X06시.07시.하차인원", "X07시.08
시.승차인원", "X07시.08시.하차인원", "X08시.09시.승차인원", "X08시.09시.하차인원", "X17시.18시.승
차인원", "X17시.18시.하차인원", "X18시.19시.승차인원", "X18시.19시.하차인원", "X19시.20시.승차인
원", "X19시.20시.하차인원")
str(subway_time)
```

```
## 'data.frame':    11120 obs. of  15 variables:
##  $ 사용월             : int  202002 202002 202002 202002 202002 202002 202002 202002 202002
202002 ...
##  $ 호선명             : Factor w/ 26 levels "1호선","2호선",..: 1 1 1 1 1 24 24 24 24 24
...
##  $ 지하철역           : Factor w/ 563 levels "4.19민주묘지",..: 147 475 474 317 272 129 479
468 167 207 ...
##  $ X06시.07시.승차인원: int  8461 3476 3467 3300 11027 23789 13417 4180 8240 8468 ...
##  $ X06시.07시.하차인원: int  5858 11699 24993 16533 38332 5180 3506 3021 4264 5307 ...
##  $ X07시.08시.승차인원: int  12050 4511 5274 6242 32573 60906 22713 7516 14477 20195 ...
##  $ X07시.08시.하차인원: int  10105 23897 94607 57309 81286 11528 6226 5527 7317 11131 ...
##  $ X08시.09시.승차인원: int  17226 7227 9004 8081 58541 42731 21197 8381 13877 21478 ...
##  $ X08시.09시.하차인원: int  22032 69206 242224 178660 193055 17511 12021 18569 20353 24731
...
##  $ X17시.18시.승차인원: int  22813 72521 113121 60679 119660 18074 12841 17220 13749 18882
...
##  $ X17시.18시.하차인원: int  18518 33092 37647 16297 60448 18862 13990 12494 11193 16829
...
##  $ X18시.19시.승차인원: int  23275 98198 206201 144434 196464 19699 13671 25081 18138 24989
...
##  $ X18시.19시.하차인원: int  19959 38211 48256 17295 70707 26179 20688 16880 15532 22523
...
##  $ X19시.20시.승차인원: int  16262 63162 97266 56175 99142 10410 7254 12165 9818 13845 ...
##  $ X19시.20시.하차인원: int  21960 23269 30281 10215 51008 31440 24007 15058 16246 24737
...
```

데이터 중에서 승하차 인원이 0명이거나 승하차 인원이 유난히 적은 역이 존재하여 이상치로 판단하여 제거하였다. 그 이유는 다음과 같다. - 충무로(3호선) : 3호선과 4호선 개찰구 공동 집계 -> 4호선에 포함 - 신내(6호선) :

신설 역 (2019.12.21 개통) - 연신내(6호선) : 6호선과 3호선 공동 집계 -> 3호선에 포함 - 복정(분당선) : 분당선과 8호선 공동 집계 -> 8호선에 포함 - 지축(일산선) : 일산선과 3호선의 경계 -> 3호선으로 봐도 무방

```
# 이상치 확인 후 NA로 변경
View(subway_month)
subway_time %>% arrange(X06시.07시.하차인원) %>% head()
```

```
##   사용월 호선명 지하철역 X06시.07시.승차인원 X06시.07시.하차인원
## 1 202002 분당선     복정                  0                  0
## 2 202002  6호선     신내                  0                  0
## 3 202002  6호선   연신내                  2                  0
## 4 202002  3호선   충무로                  0                  0
## 5 202001  3호선   충무로                  0                  0
## 6 202001  6호선   연신내                  0                  0
##   X07시.08시.승차인원 X07시.08시.하차인원 X08시.09시.승차인원
## 1                   2                  0                  0
## 2                   0                  0                  0
## 3                   0                  0                  1
## 4                   0                  0                  1
## 5                   1                  0                  1
## 6                   0                  0                  1
##   X08시.09시.하차인원 X17시.18시.승차인원 X17시.18시.하차인원
## 1                   0                  1                  0
## 2                   0                  0                  0
## 3                   0                  5                  0
## 4                   0                  4                  0
## 5                   0                  2                  0
## 6                   0                  0                  0
##   X18시.19시.승차인원 X18시.19시.하차인원 X19시.20시.승차인원
## 1                   1                  0                  2
## 2                   0                  0                  1
## 3                   0                  0                  0
## 4                   6                  0                  5
## 5                   3                  0                  7
## 6                   2                  0                  1
##   X19시.20시.하차인원
## 1                   0
## 2                   0
## 3                   0
## 4                   0
## 5                   0
## 6                   0
```

```
subway_month$하차총승객수 <- ifelse(subway_month$하차총승객수 == 0, NA, subway_month$하차총승객수)
subway_month$승차총승객수 <- ifelse(subway_month$하차총승객수 == 0, NA, subway_month$승차총승객수)

# 이상치 변경을 위한 노선, 역별 평균
subway_mean = subway_month %>%
  group_by(노선명,역명) %>%
  summarise(승차총승객수평균 = mean(승차총승객수, na.rm = T),
                    하차총승객수평균 = mean(하차총승객수, na.rm = T))

# 판단에 의한 이상치 제거
subway_month <- subway_month %>% filter(!is.na(subway_month$승차총승객수))
table(is.na(subway_month))
```

```
##
##   FALSE
## 1674800
```

```
subway_time <- subway_time[!(subway_time$호선명 == "분당선" & subway_time$지하철역 == "복정"),]
subway_time <- subway_time[!(subway_time$호선명 == "3호선" & subway_time$지하철역 == "충무로"),]
subway_time <- subway_time[!(subway_time$호선명 == "6호선" & subway_time$지하철역 == "연신내"),]
subway_time <- subway_time[!(subway_time$호선명 == "6호선" & subway_time$지하철역 == "신내"),]
subway_time <- subway_time[!(subway_time$호선명 == "일산선" & subway_time$지하철역 == "지축"),]
```

데이터들을 각각 노선별, 역별로 요약하였다. 주말도 포함해서 요약한 이유는 주말 아침이나 저녁에 지하철을 타게 되면 평일 출퇴근길에 비할 바는 아니지만, 어느 정도 이용객이 많기 때문이다.

```
# 노선별, 역별로 요약
subway_1 <- subway_month %>%
  group_by(노선명, 역명) %>%
  summarise(승차총승객수평균 = mean(승차총승객수),
                    하차총승객수평균 = mean(하차총승객수))

subway_time <-  subway_time %>%
  group_by(호선명, 지하철역) %>%
  summarise(X06시.07시.승차인원평균 = mean(X06시.07시.승차인원),
            X06시.07시.하차인원평균 = mean(X06시.07시.하차인원),
            X07시.08시.승차인원평균 = mean(X07시.08시.승차인원),
            X07시.08시.하차인원평균 = mean(X07시.08시.하차인원),
            X08시.09시.승차인원평균 = mean(X08시.09시.승차인원),
            X08시.09시.하차인원평균 = mean(X08시.09시.하차인원),
            X17시.18시.승차인원평균 = mean(X17시.18시.승차인원),
            X17시.18시.하차인원평균 = mean(X17시.18시.하차인원),
            X18시.19시.승차인원평균 = mean(X18시.19시.승차인원),
            X18시.19시.하차인원평균 = mean(X18시.19시.하차인원),
            X19시.20시.승차인원평균 = mean(X19시.20시.승차인원),
            X19시.20시.하차인원평균 = mean(X19시.20시.하차인원))
```

승차 수/하차 수를 혼잡도로 기준 삼은 이유는 특정한 역에서 하차하는 사람보다 승차하는 사람이 많으면 승차했을 당시 전철 내의 사람이 더 많아져 혼잡해지기 때문이다. 그래서 승차 수/하차 수가 1 이상이면 비교적 혼잡하다고 보고, 1 미만이면 덜 혼잡하다고 보았다.

```
subway_1 <- subway_1  %>%
  mutate(ratio = 승차총승객수평균/하차총승객수평균)

subway_1 %>% arrange(ratio) %>%
  head(20)
```

```
## # A tibble: 20 x 5
## # Groups:   노선명 [10]
##    노선명 역명                 승차총승객수평균 하차총승객수평균 ratio
##    <chr>  <chr>                           <dbl>            <dbl> <dbl>
##  1 분당선 선릉                            9766.            17429. 0.560
##  2 4호선  서울역                         13189.            22661. 0.582
##  3 9호선  고속터미널                     13658.            21249. 0.643
##  4 3호선  종로3가                         10275.            13438. 0.765
##  5 9호선  김포공항                         9504.            12046. 0.789
##  6 5호선  동대문역사문화공원             2841.             3443. 0.825
##  7 7호선  도봉산                          9364.            11319. 0.827
##  8 4호선  삼각지                          4856.             5785. 0.840
##  9 6호선  태릉입구                        5575.             6578. 0.848
## 10 경부선 가산디지털단지                 15715.            18511. 0.849
## 11 6호선  한강진                          8152.             9578. 0.851
## 12 5호선  왕십리(성동구청)                5000.             5842. 0.856
## 13 분당선 강남구청                        8791.            10228. 0.859
## 14 6호선  상수                           10604.            12272. 0.864
## 15 8호선  잠실(송파구청)                 15287.            17677. 0.865
## 16 8호선  가락시장                        7582.             8737. 0.868
## 17 6호선  역촌                            3993.             4572. 0.873
## 18 2호선  동대문역사문화공원             18406.            21052. 0.874
## 19 2호선  한양대                         10794.            12307. 0.877
## 20 5호선  신길                            3306.             3764. 0.878
```

```
subway_1 %>% arrange(desc(ratio)) %>%
  head(20)
```

```
## # A tibble: 20 x 5
## # Groups:   노선명 [13]
##    노선명 역명                 승차총승객수평균 하차총승객수평균 ratio
##    <chr>  <chr>                        <dbl>           <dbl> <dbl>
##  1 경부선 서울역                      6656.           1313.  5.07
##  2 7호선  장암                       2446.            963.  2.54
##  3 4호선  남태령                     2425.           1166.  2.08
##  4 3호선  교대(법원.검찰청)          14515.           9563.  1.52
##  5 분당선 수원                       9315.           6213.  1.50
##  6 9호선  개화                       6069.           4085.  1.49
##  7 7호선  군자(능동)                14710.          10902.  1.35
##  8 6호선  구산                       7947.           5942.  1.34
##  9 6호선  화랑대(서울여대입구)       12804.          9780.  1.31
## 10 일산선 대곡                       1727.           1323.  1.31
## 11 8호선  모란                       4839.           3713.  1.30
## 12 5호선  신정(은행정)               13845.          10761.  1.29
## 13 9호선  동작(현충원)               2124.           1664.  1.28
## 14 경부선 광명                       2811.           2226.  1.26
## 15 일산선 대화                      15319.          12208.  1.25
## 16 안산선 수리산                     4507.           3618.  1.25
## 17 7호선  부평구청                  11754.           9457.  1.24
## 18 6호선  삼각지                     7007.           5665.  1.24
## 19 과천선 선바위                     9257.           7514.  1.23
## 20 경인선 인천                       4491.           3649.  1.23
```

```
subway_time <- subway_time %>%
  mutate(X06시.07시_ratio = X06시.07시.승차인원평균/X06시.07시.하차인원평균) %>%
  mutate(X07시.08시_ratio = X07시.08시.승차인원평균/X07시.08시.하차인원평균) %>%
  mutate(X08시.09시_ratio = X08시.09시.승차인원평균/X08시.09시.하차인원평균) %>%
  mutate(X17시.18시_ratio = X17시.18시.승차인원평균/X17시.18시.하차인원평균) %>%
  mutate(X18시.19시_ratio = X18시.19시.승차인원평균/X18시.19시.하차인원평균) %>%
  mutate(X19시.20시_ratio = X19시.20시.승차인원평균/X19시.20시.하차인원평균)

subway_2 <- subway_time %>% select(호선명, 지하철역, X06시.07시_ratio, X07시.08시_ratio, X08시.
09시_ratio, X17시.18시_ratio, X18시.19시_ratio, X19시.20시_ratio)
```

4 . 분석 결과

```
library(ggplot2)
library(plotly)
```

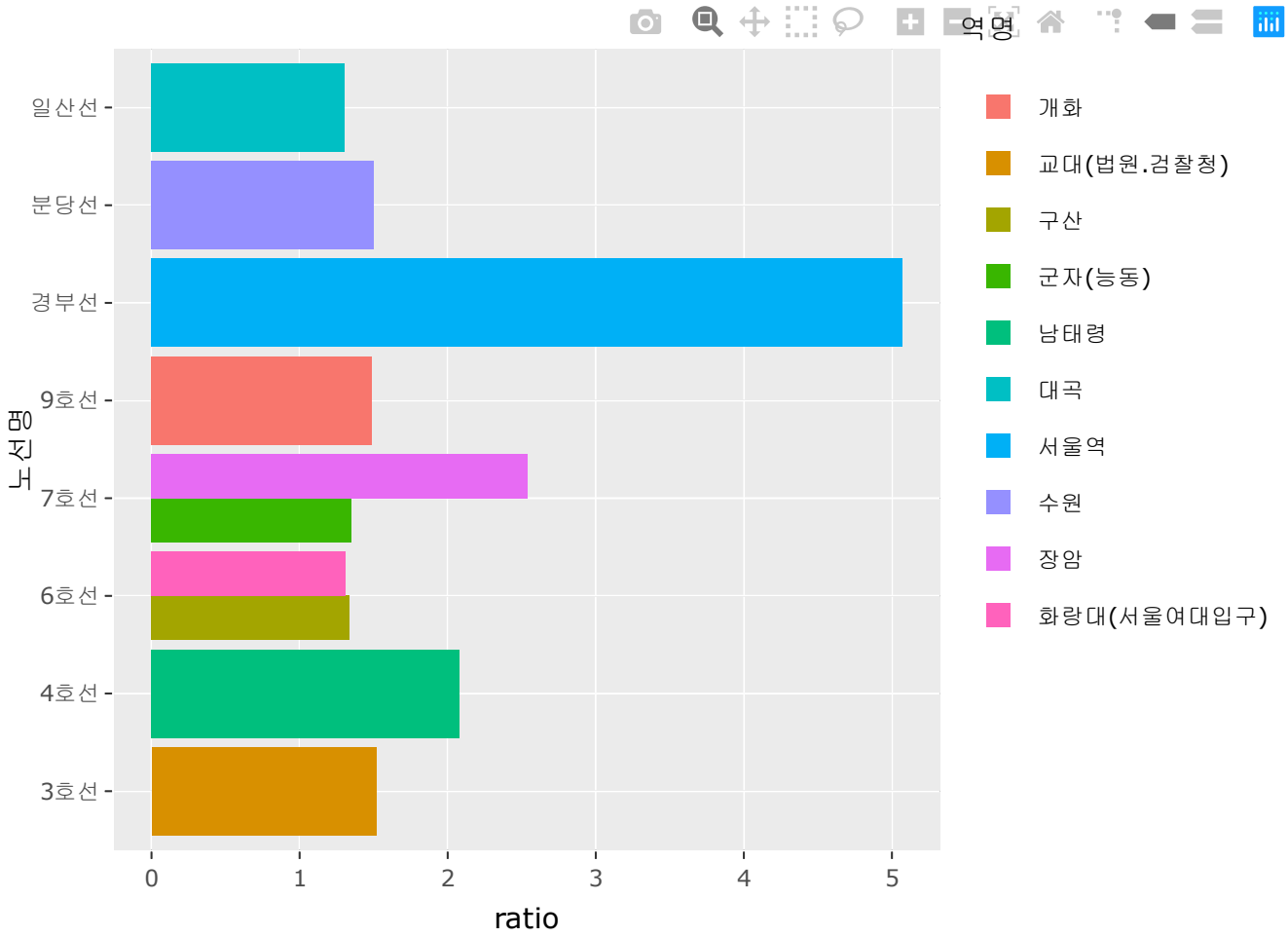
```
##
## Attaching package: 'plotly'
```

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

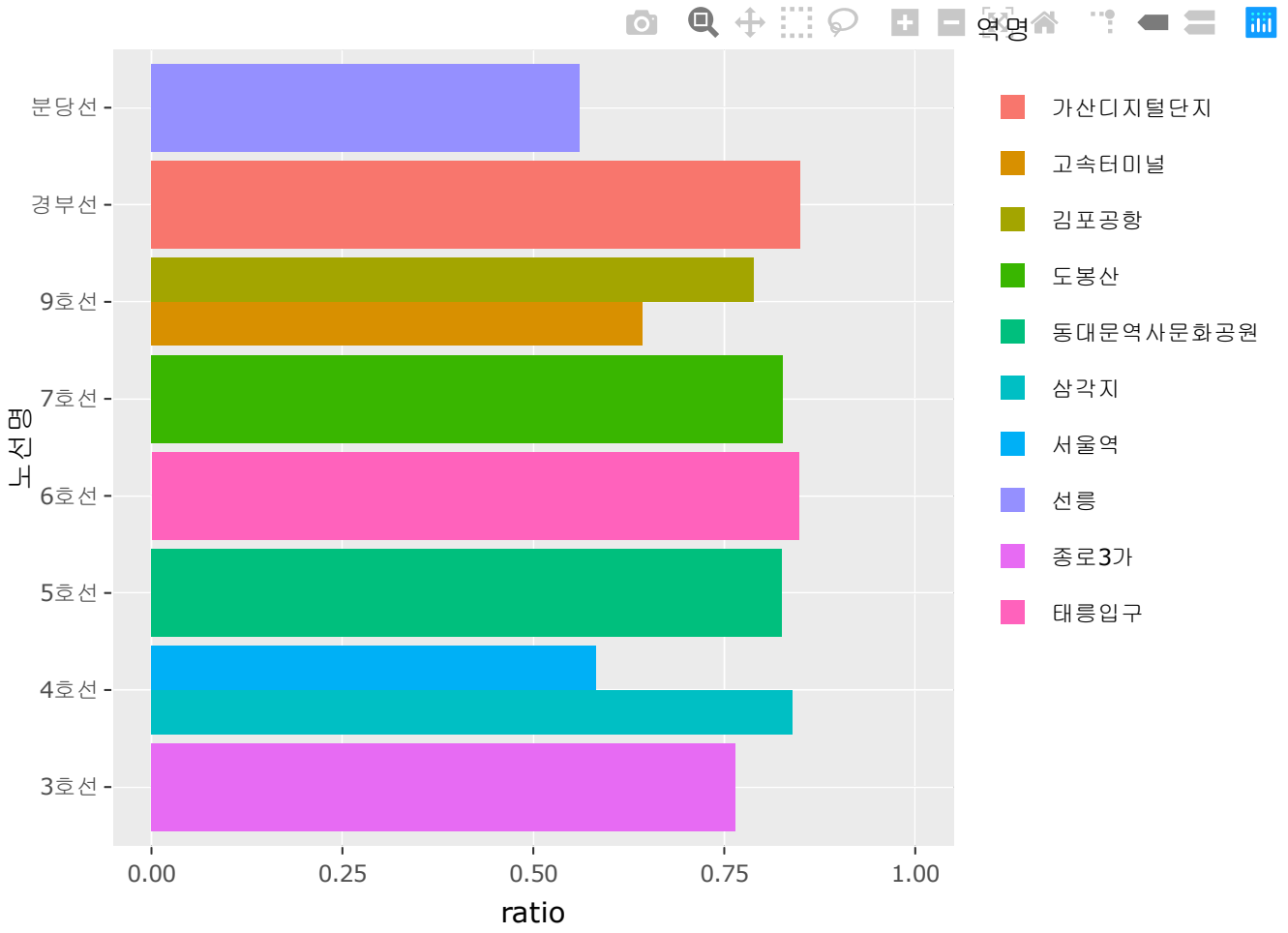
```
## The following object is masked from 'package:stats':
##
##     filter
```

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

```
total_de <- subway_1 %>% arrange(desc(ratio)) %>%
  select(노선명, 역명, ratio) %>%
  head(10)

p13 <- ggplot(data = total_de, aes(x = 노선명, y = ratio, fill = 역명)) + geom_col(position =
"dodge") + coord_flip()
ggplotly(p13)
```

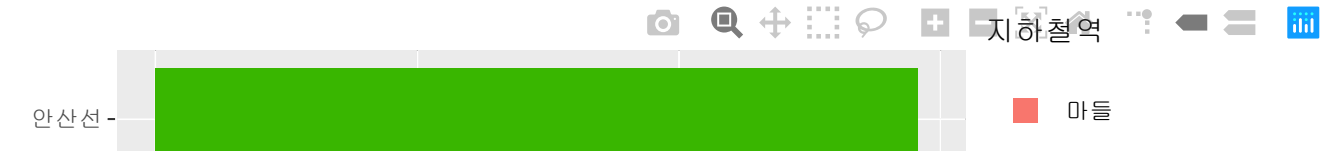


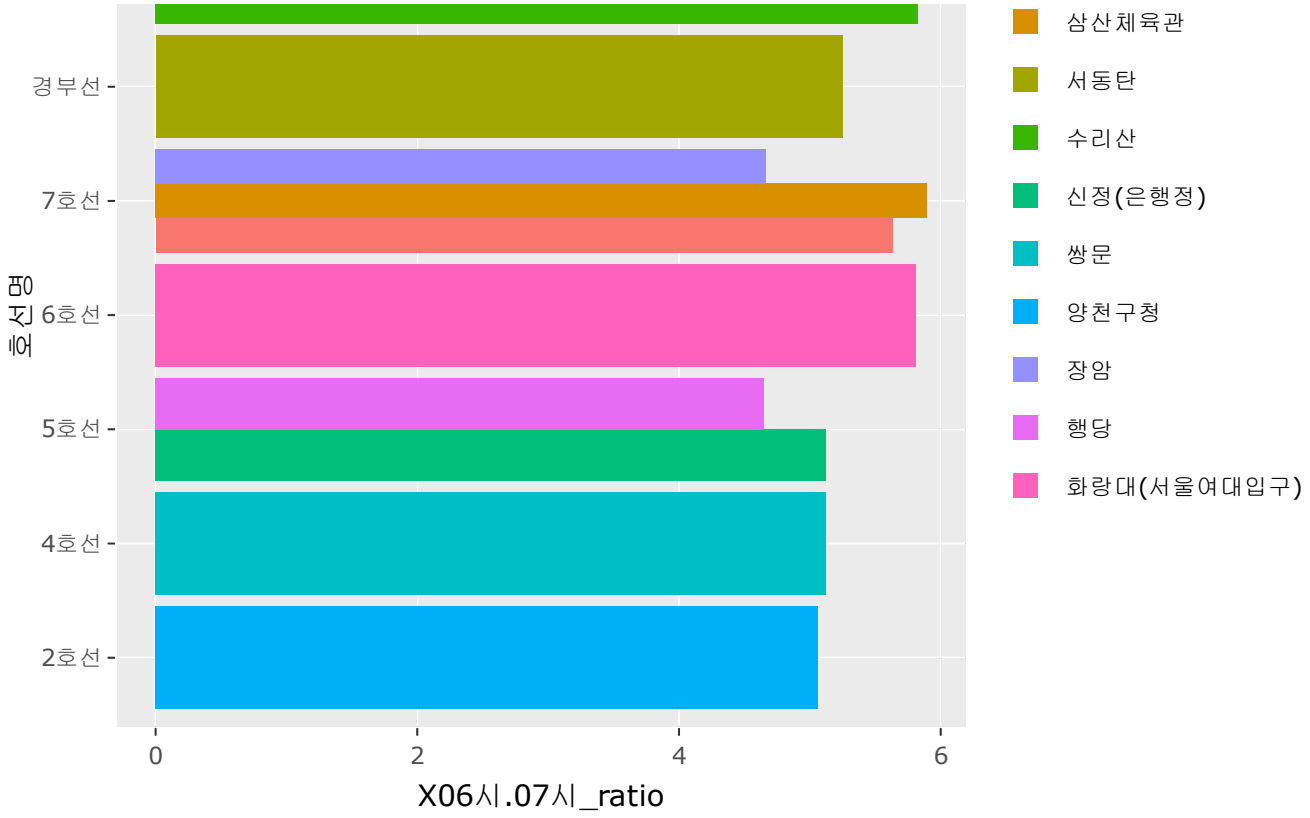
```
total_as <- subway_1 %>% arrange(ratio) %>%
  select(노선명, 역명, ratio) %>%
  head(10)

p14 <- ggplot(data = total_as, aes(x = 노선명, y = ratio, fill = 역명)) + geom_col(position =
"dodge") + coord_flip() + ylim(0, 1)
ggplotly(p14)
```



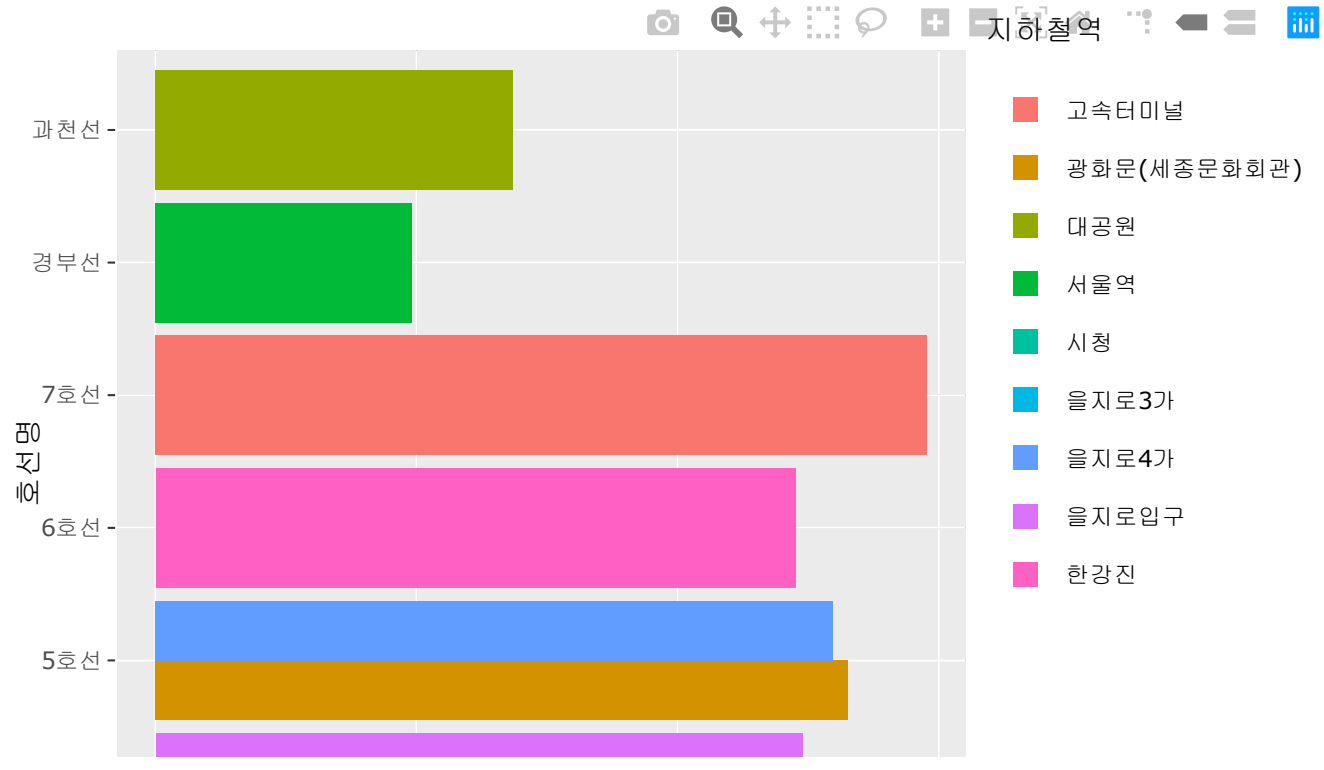
이 그래프들을 통해서 전체적으로 보면 수치가 비슷비슷하지만, 서울역(경부선, 1호선)이 다른 역들에 비해서 수치가 두드러지게 높고, 장암(7호선)과 남태령(4호선)도 높은 편이다. 반면에 선릉(분당선), 서울역(4호선), 고속터미널(9호선)의 수치가 비교적 낮은 것을 알 수 있다.

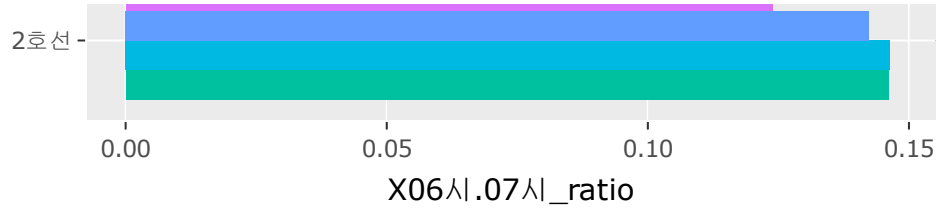
```
sixam_de = subway_2 %>% arrange(desc(X06시.07시_ratio)) %>%
  select(호선명, 지하철역, X06시.07시_ratio) %>%
  head(10)

p1 <- ggplot(data = sixam_de, aes(x = 호선명, y = X06시.07시_ratio, fill = 지하철역)) + geom_co
l(position = "dodge") + coord_flip()
ggplotly(p1)
```





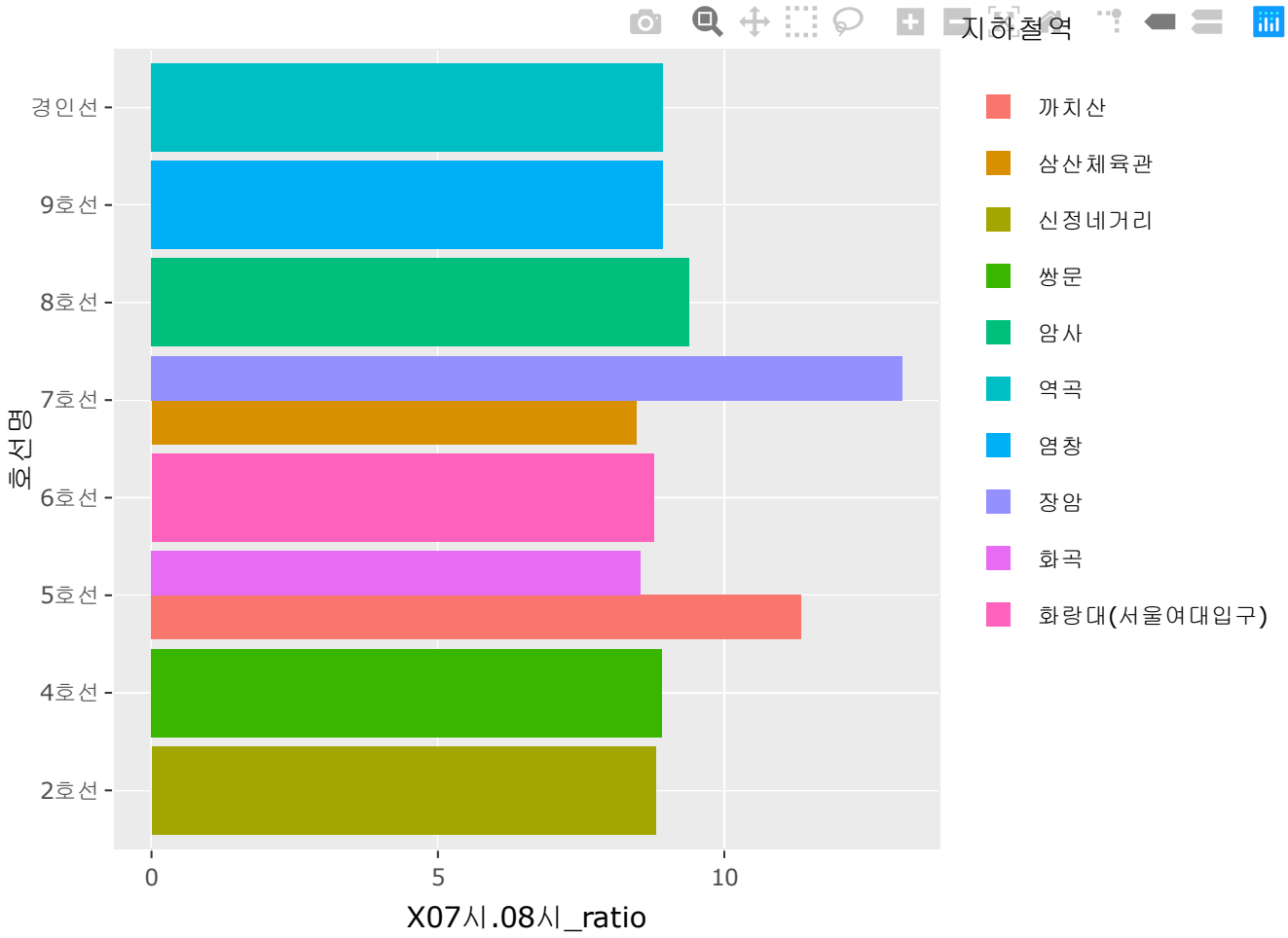

```
sixam_as = subway_2 %>% arrange(X06시.07시_ratio) %>%
  select(호선명, 지하철역, X06시.07시_ratio) %>%
  head(10)

p2 <- ggplot(data = sixam_as, aes(x = 호선명, y = X06시.07시_ratio, fill = 지하철역)) + geom_co
l(position = "dodge") + coord_flip()
ggplotly(p2)
```



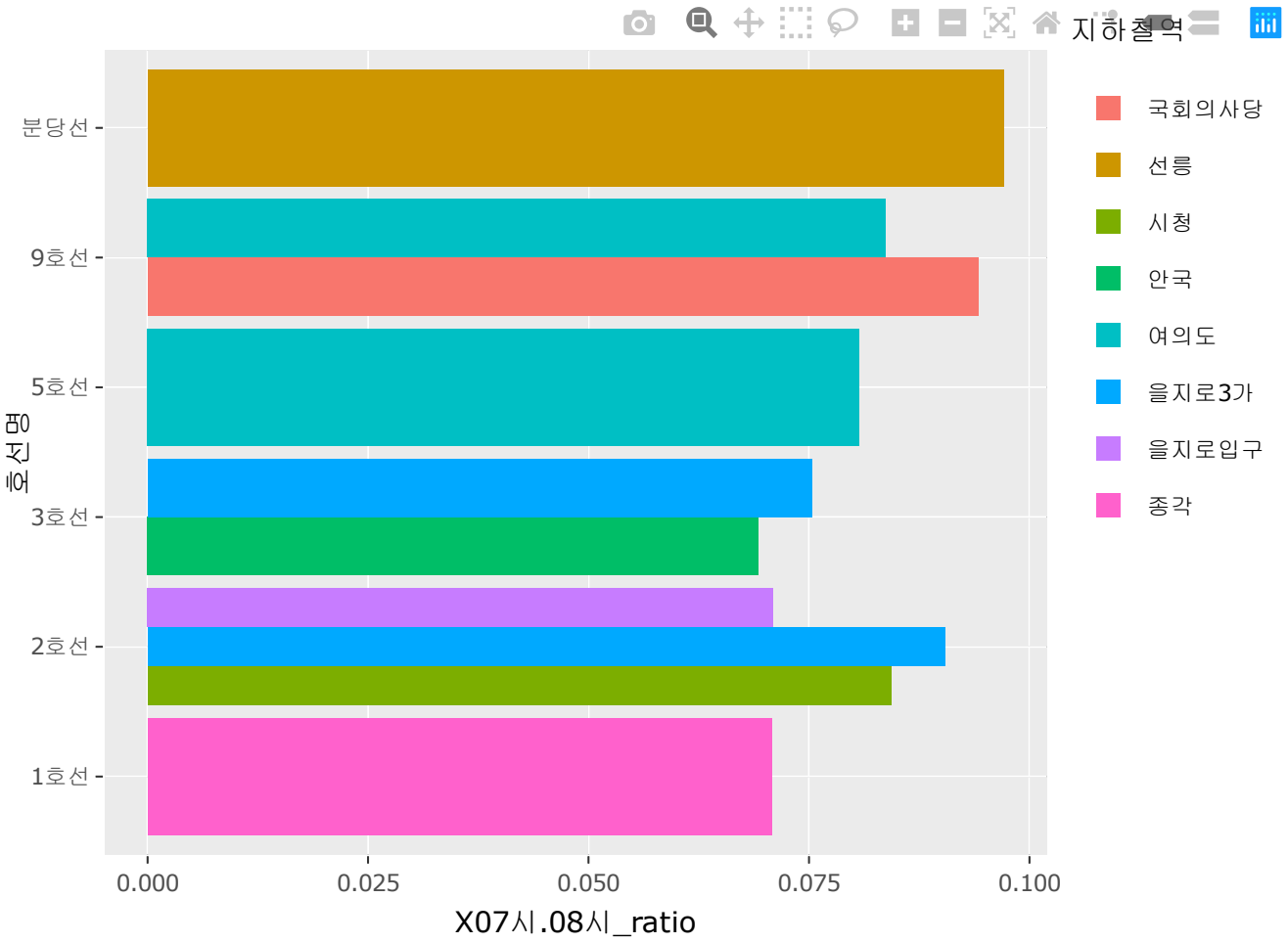


아침 6시에서 7시까지의 시간대의 혼잡도는 삼산체육관(7호선), 화랑대(6호선), 수리산(안산선)이 가장 높은 편이고 경부선(서울역), 과천선(대공원)이 비교적 낮고, 그다음으로 한강진(6호선)이 가장 낮은 편이다.

```
sevenam_de = subway_2 %>% arrange(desc(X07시.08시_ratio)) %>%
  select(호선명, 지하철역, X07시.08시_ratio) %>%
  head(10)

p3 <- ggplot(data = sevenam_de, aes(x = 호선명, y = X07시.08시_ratio, fill = 지하철역)) + geom_
col(position = "dodge") + coord_flip()
ggplotly(p3)
```

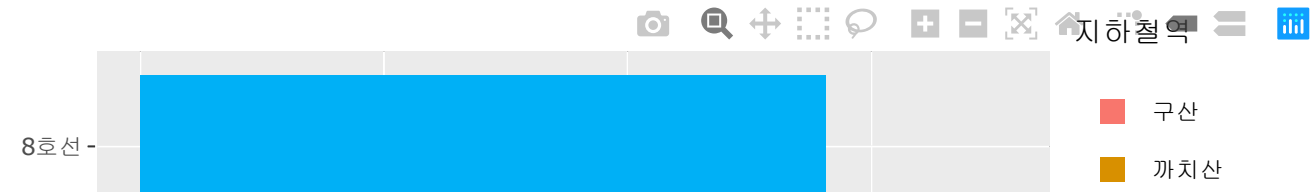


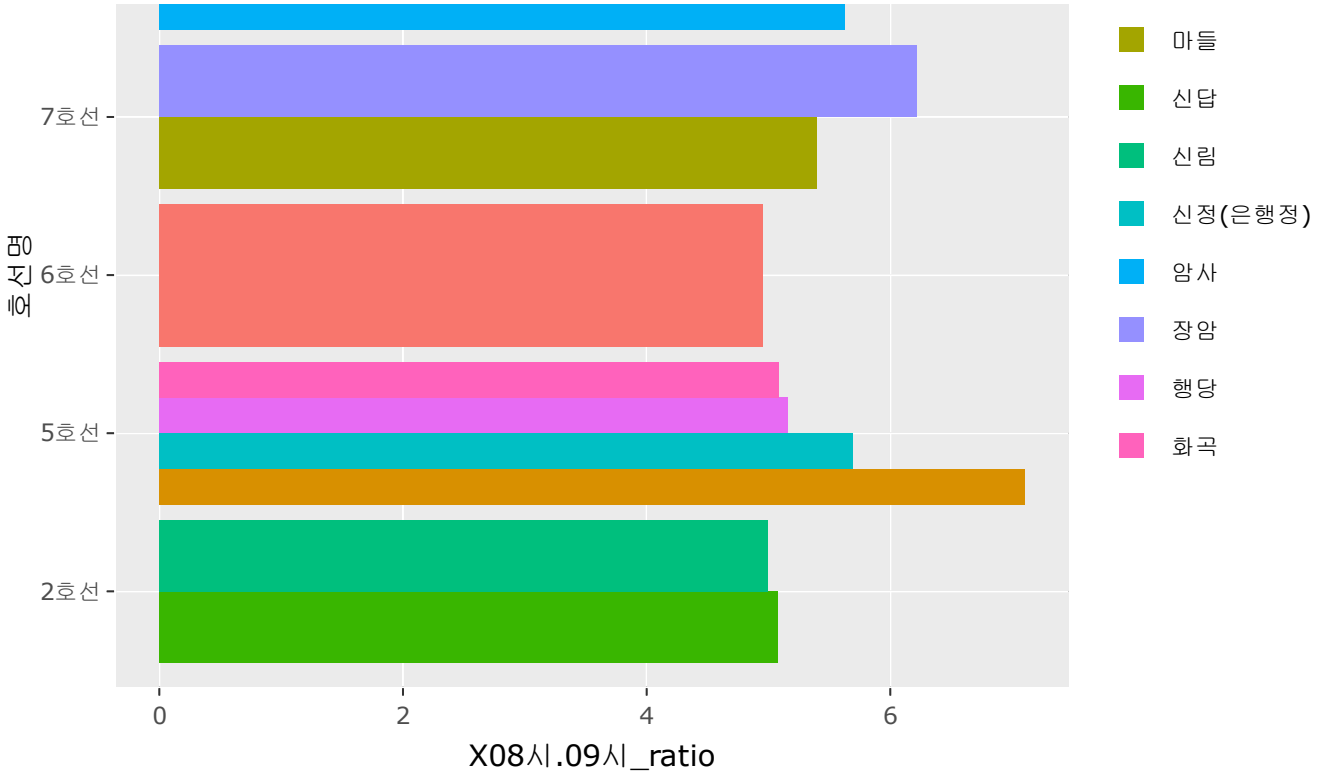
```
sevenam_as = subway_2 %>% arrange(X07시.08시_ratio) %>%
  select(호선명, 지하철역, X07시.08시_ratio) %>%
  head(10)

p4 <- ggplot(data = sevenam_as, aes(x = 호선명, y = X07시.08시_ratio, fill = 지하철역)) + geom_
col(position = "dodge") + coord_flip()
ggplotly(p4)
```



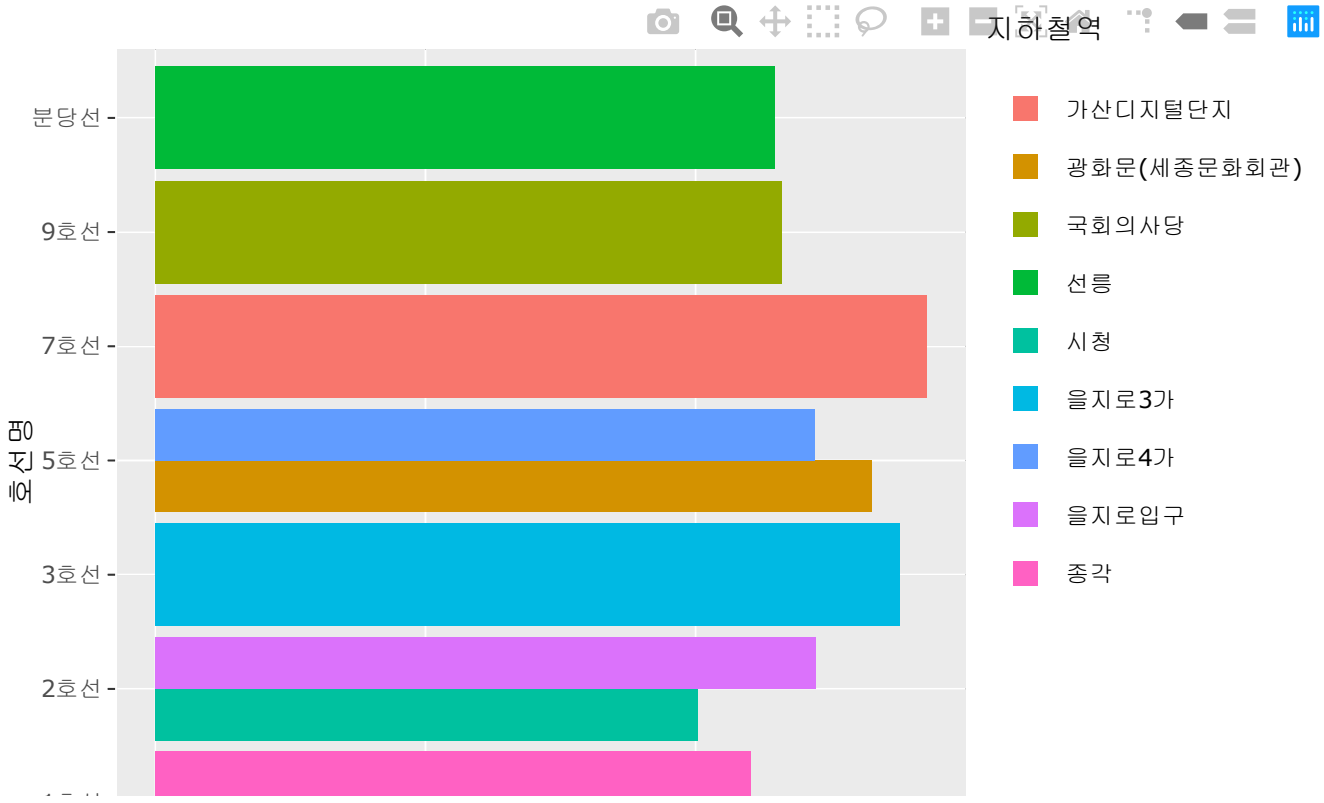
아침 7시에서 8시까지를 보자면, 대체로 다 높지만, 장암(7호선), 까치산(5호선)이 유달리 높다. 반면에 안국(3호선), 을지로입구(2호선), 종각(1호선)이 낮은 편임을 알 수 있다.

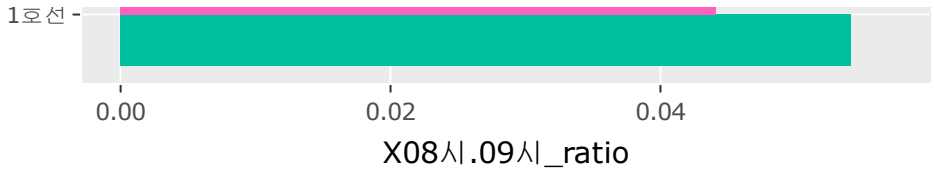
```
eightam_de = subway_2 %>% arrange(desc(X08시.09시_ratio)) %>%
  select(호선명, 지하철역, X08시.09시_ratio) %>%
  head(10)

p5 <- ggplot(data = eightam_de, aes(x = 호선명, y = X08시.09시_ratio, fill = 지하철역)) + geom_
col(position = "dodge") + coord_flip()
ggplotly(p5)
```






```
eightam_as = subway_2 %>% arrange(X08시.09시_ratio) %>%
  select(호선명, 지하철역, X08시.09시_ratio) %>%
  head(10)

p6 <- ggplot(data = eightam_as, aes(x = 호선명, y = X08시.09시_ratio, fill = 지하철역)) + geom_
col(position = "dodge") + coord_flip()
ggplotly(p6)
```





아침 8시부터 9시까지를 보면, 까치산(5호선), 장암(7호선), 신정(5호선)이 높고, 출근시간에 가까워져서 그런지 상하위 10개의 비율이 7~8시 시간대보다 다소 낮다. 시청(2호선), 종각(1호선), 선릉(분당선)의 수치가 낮은 편이다.

2호선은 승하차의 유동성이 가장 높다고 평가되기 때문에 2호선은 출근 시간대의 상/하위에 전부 포함되어있고 특정 역에 집중해서 보자면 2호선이 비교적 수치가 낮고 오히려 다른 호선의 역이 높은 것을 알 수 있다.

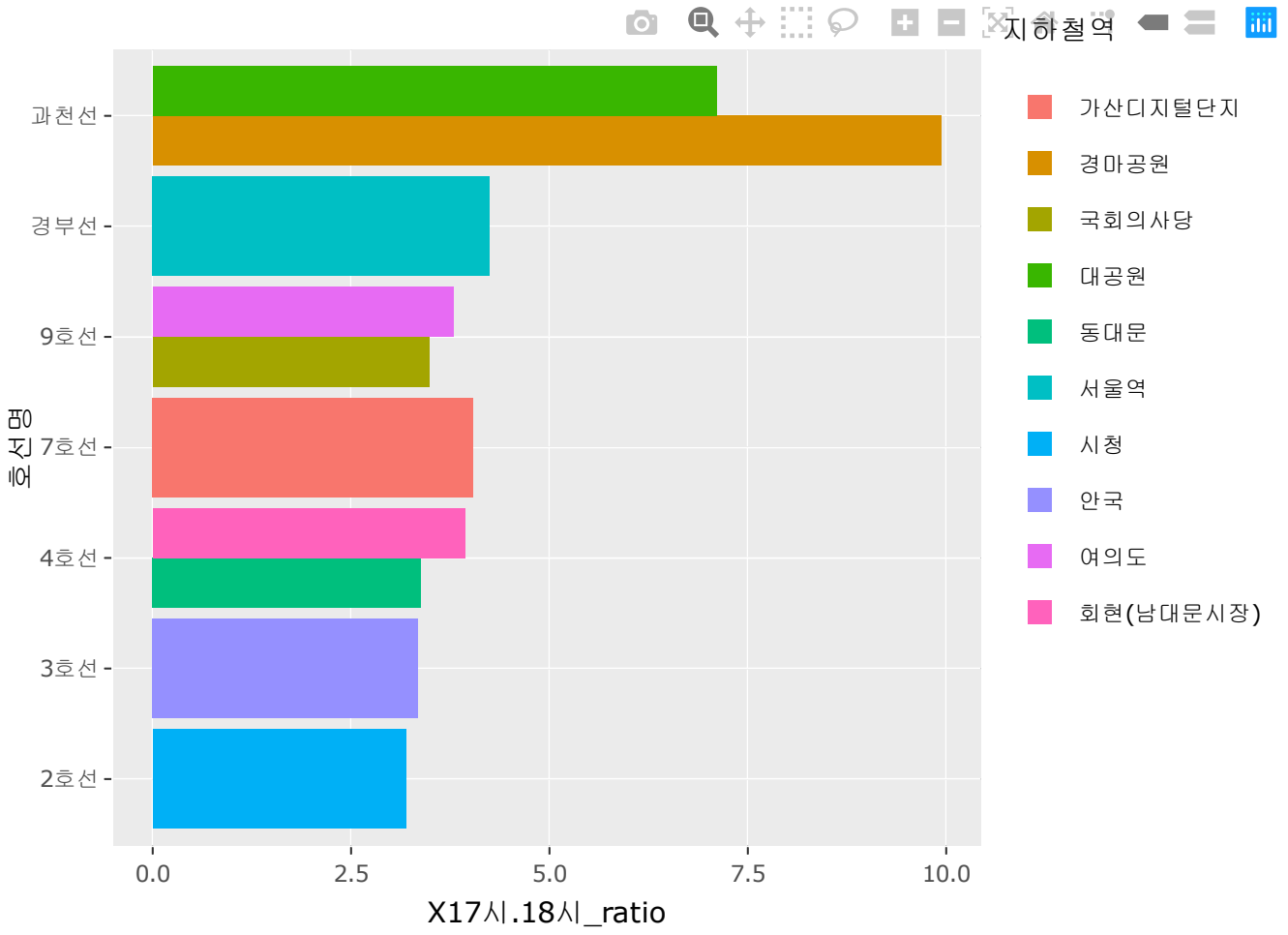
```
gotowork_de <- bind_rows(sixam_de, sevenam_de, eightam_de)
gotowork_as <- bind_rows(sixam_as, sevenam_as, eightam_as)

table(droplevels(gotowork_de$지하철역))
```

```
##
##                 구산                 까치산                   마들
##                    1                      2                      2
##           삼산체육관                 서동탄                 수리산
##                    2                      1                      1
##                 신답                  신림          신정(은행정)
##                    1                      1                      2
##           신정네거리                  쌍문                   암사
##                    1                      2                      2
##              양천구청                  역곡                   염창
##                    1                      1                      1
##                 장암                  행당                   화곡
##                    3                      2                      2
## 화랑대(서울여대입구)
##                    2
```

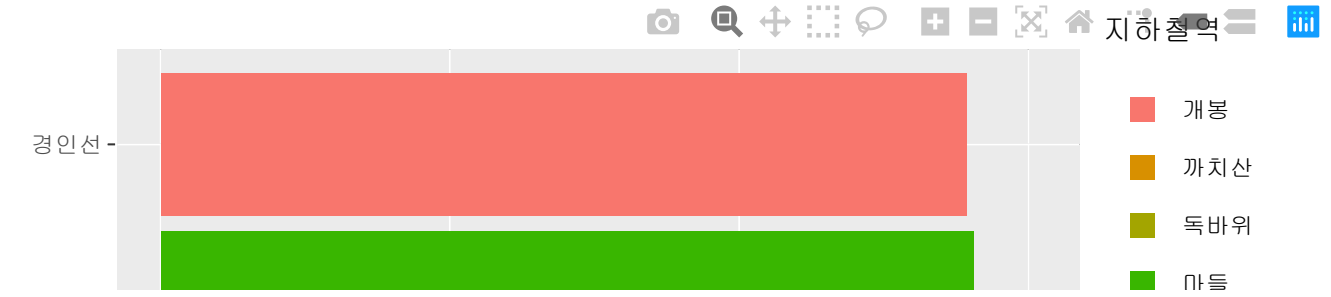
```
table(droplevels(gotowork_as$지하철역))
```

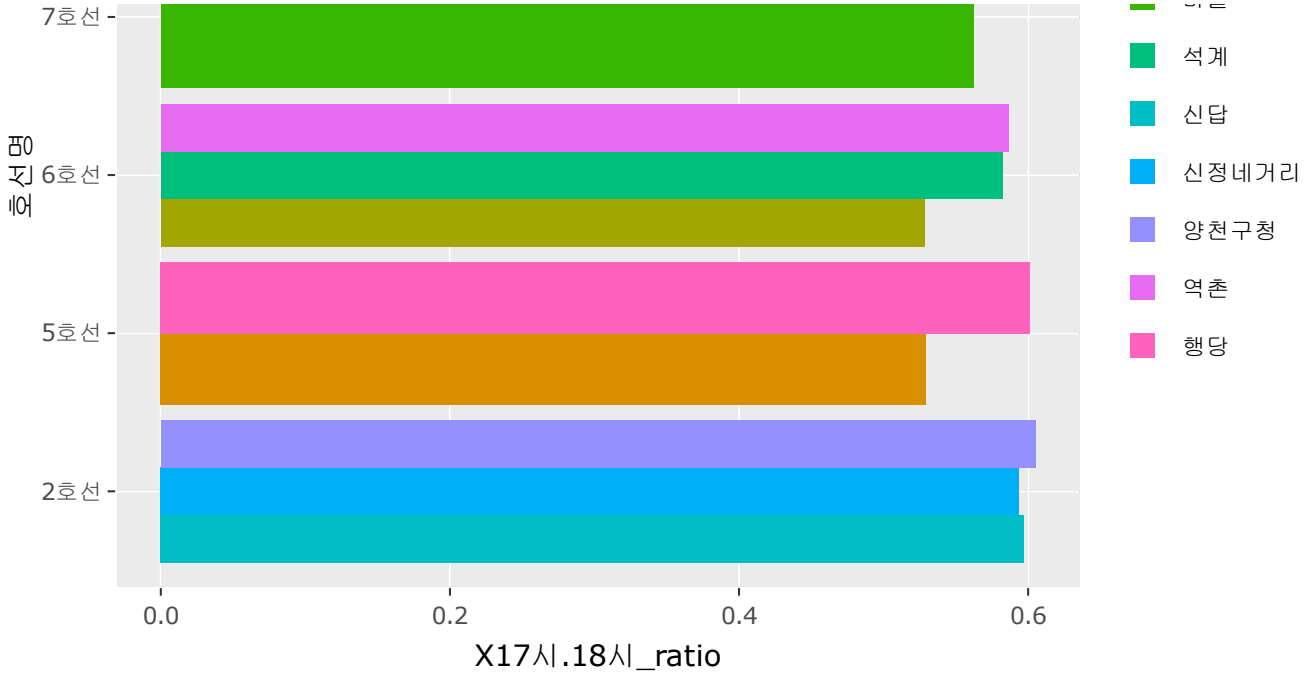
```
##
##       가산디지털단지             고속터미널  광화문(세종문화회관)
##                    1                      1                      2
##           국회의사당                 대공원                 서울역
##                    2                      1                      1
##                 선릉                  시청                   안국
##                    2                      4                      1
##               여의도              을지로3가              을지로4가
##                    2                      4                      3
##           을지로입구                  종각                 한강진
##                    3                      2                      1
```

```
fivepm_de = subway_2 %>% arrange(desc(X17시.18시_ratio)) %>%
  select(호선명, 지하철역, X17시.18시_ratio) %>%
  head(10)

p7 <- ggplot(data = fivepm_de, aes(x = 호선명, y = X17시.18시_ratio, fill = 지하철역)) + geom_c
ol(position = "dodge") + coord_flip()
ggplotly(p7)
```



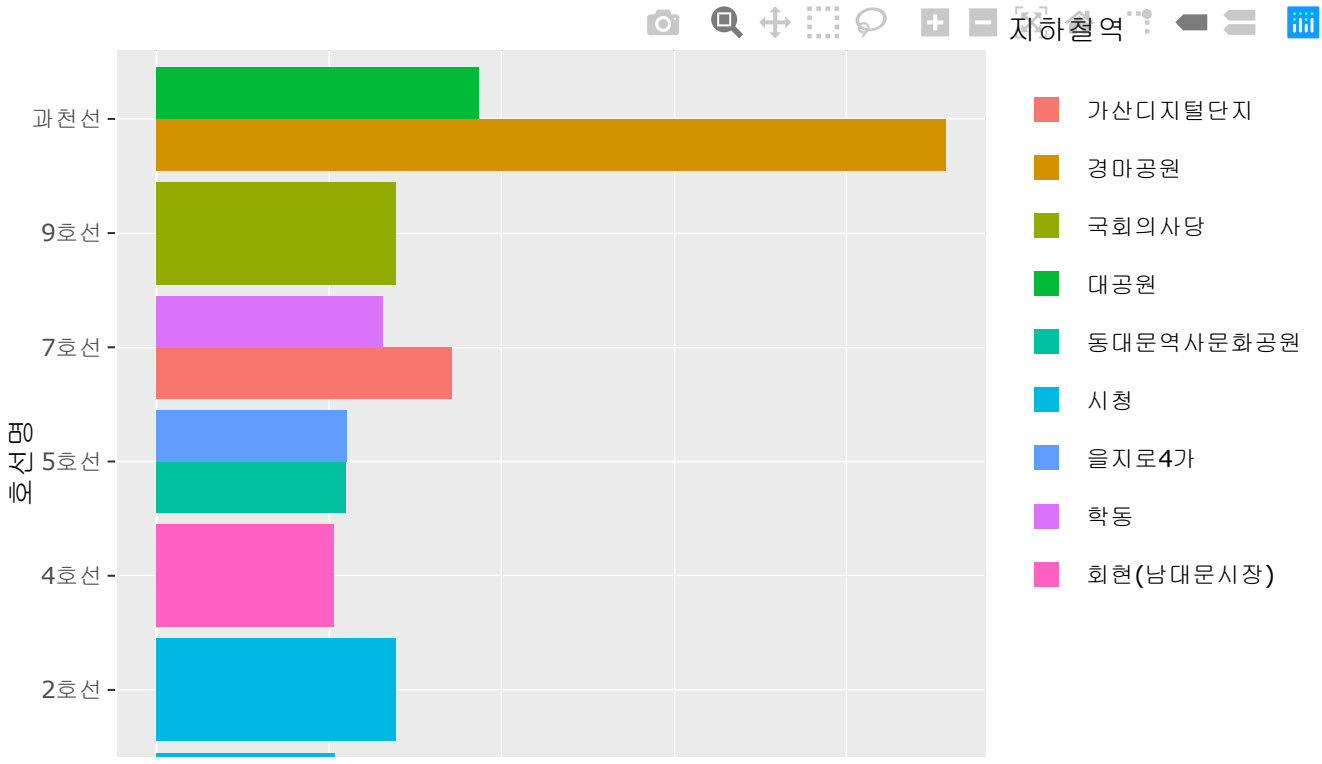

```
fivepm_as = subway_2 %>% arrange(X17시.18시_ratio) %>%
  select(호선명, 지하철역, X17시.18시_ratio) %>%
  head(10)

p8 <- ggplot(data = fivepm_as, aes(x = 호선명, y = X17시.18시_ratio, fill = 지하철역)) + geom_c
ol(position = "dodge") + coord_flip()
ggplotly(p8)
```

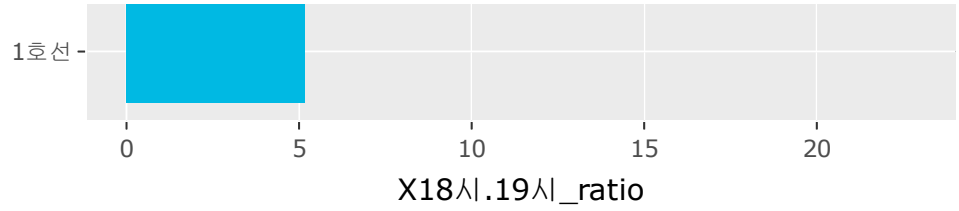




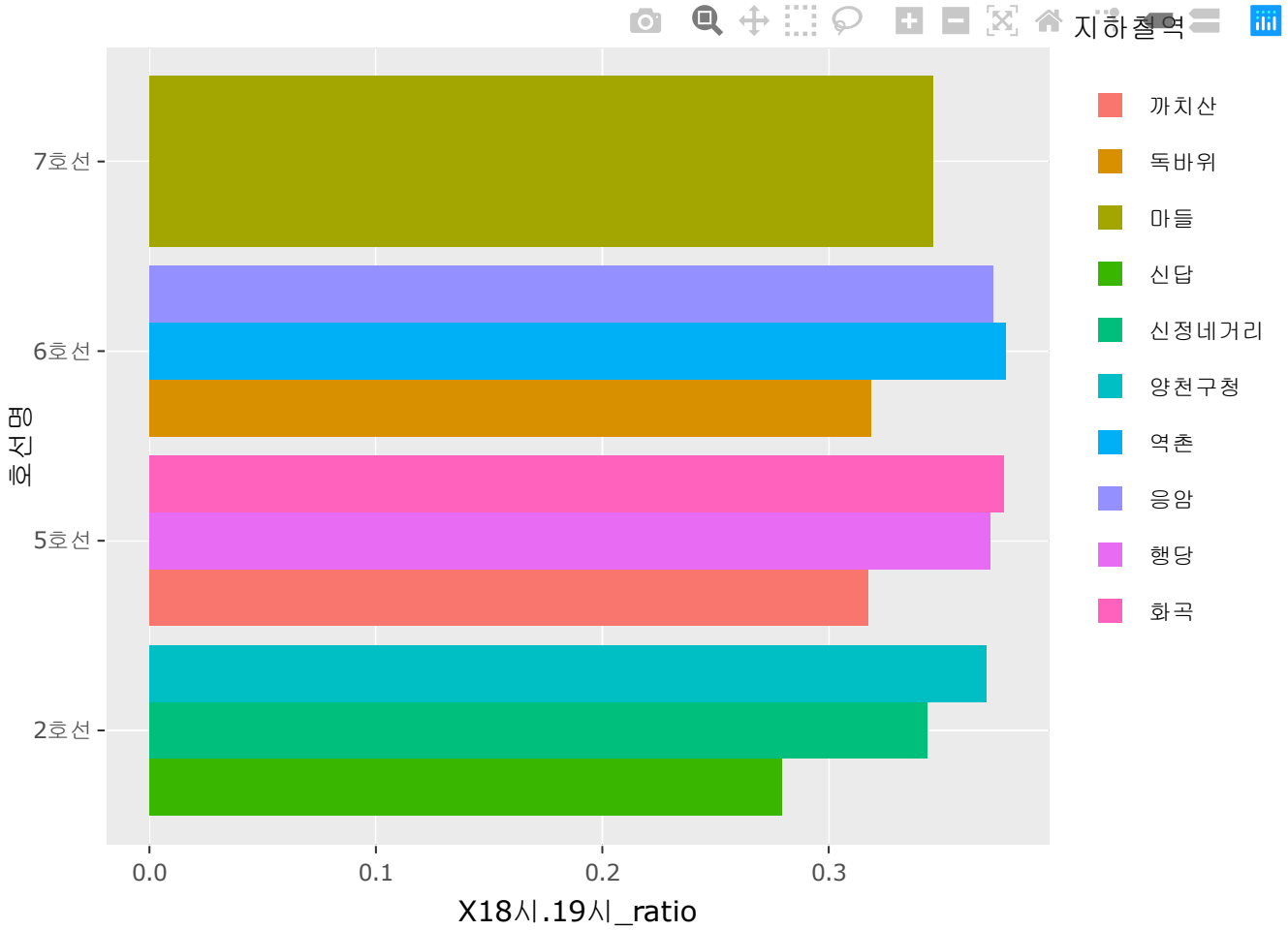
오후 5시에서 6시 사이를 보면, 경마공원, 대공원(과천선)이 높은 걸 알 수 있고, 일찍 퇴근하거나 퇴근 시간에 가까워져 있어서 하위 10개 역의 수치가 다른 시간대에 비해서 높은 편이다. 가장 수치가 낮은 역은, 비슷한 수치이지만, 독바위(6호선), 까치산(5호선), 개봉(경인선)이 비교적 낮은 편이다.

```
sixpm_de = subway_2 %>% arrange(desc(X18시.19시_ratio)) %>%
  select(호선명, 지하철역, X18시.19시_ratio) %>%
  head(10)

p9 <- ggplot(data = sixpm_de, aes(x = 호선명, y = X18시.19시_ratio, fill = 지하철역)) + geom_co
l(position = "dodge") + coord_flip()
ggplotly(p9)
```



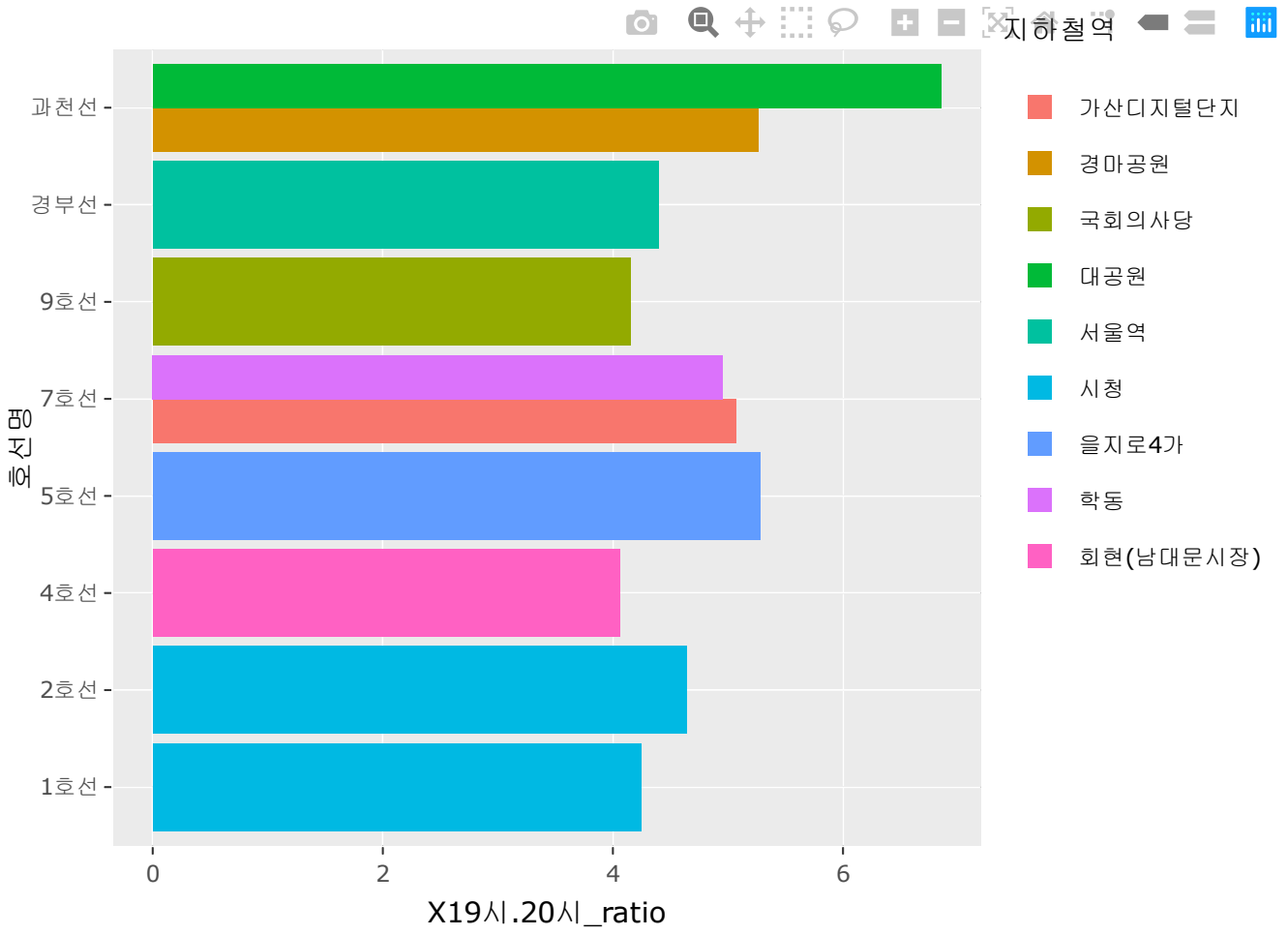



```
sixpm_as = subway_2 %>% arrange(X18시.19시_ratio) %>%
  select(호선명, 지하철역, X18시.19시_ratio) %>%
  head(10)

p10 <- ggplot(data = sixpm_as, aes(x = 호선명, y = X18시.19시_ratio, fill = 지하철역)) + geom_c
ol(position = "dodge") + coord_flip()
ggplotly(p10)
```

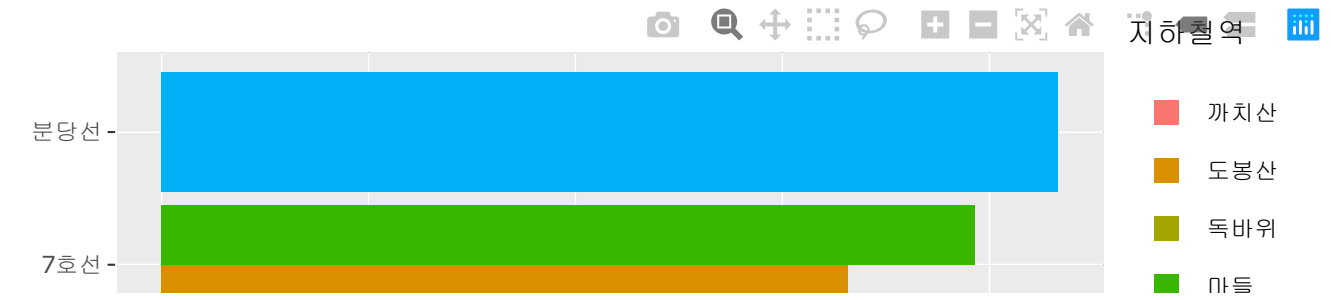


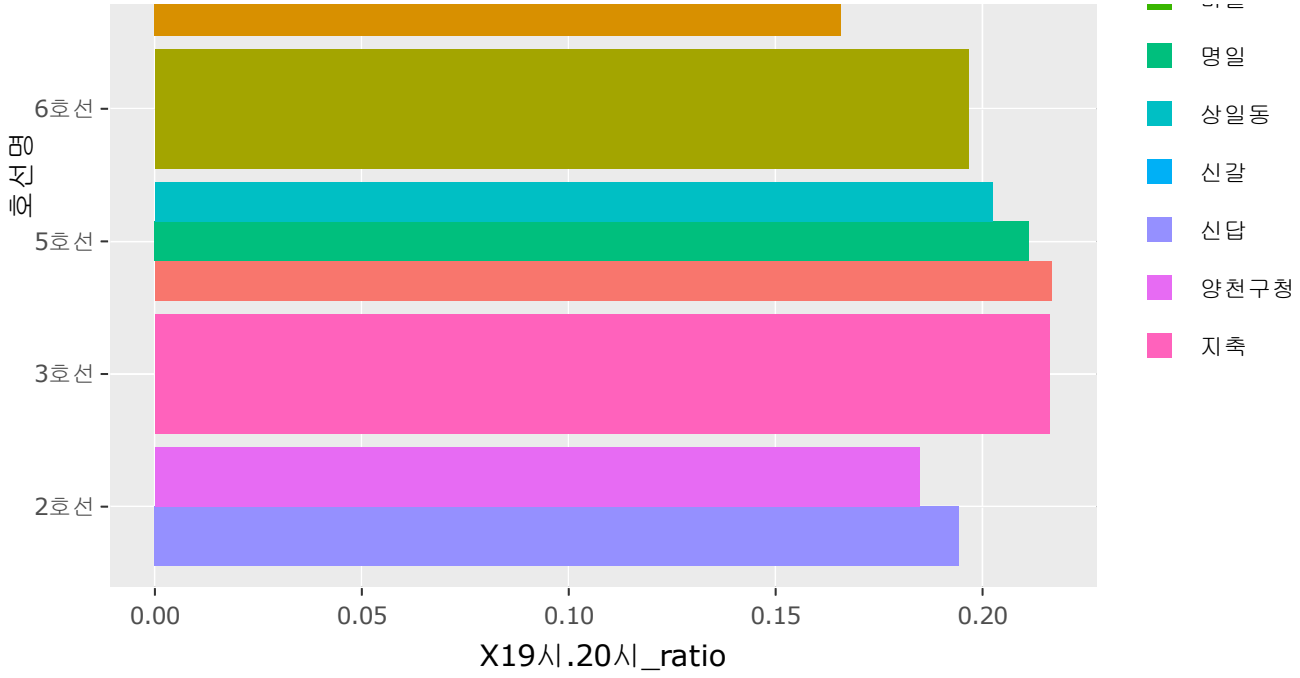
오후 6시에서 7시 사이를 보면, 대체로 높은 수치를 보여주고 있지만, 경마공원(과천선)이 두드러지게 높고 대공원(과천선), 가산디지털단지(7호선), 시청(2호선)이 높은 편이며 신답(2호선), 까치산(5호선), 독바위(6호선)이 비교적 낮은 편이다.

```
sevenpm_de = subway_2 %>% arrange(desc(X19시.20시_ratio)) %>%
  select(호선명, 지하철역, X19시.20시_ratio) %>%
  head(10)

p11 <- ggplot(data = sevenpm_de, aes(x = 호선명, y = X19시.20시_ratio, fill = 지하철역)) + geom
_col(position = "dodge") + coord_flip()
ggplotly(p11)
```




```
sevenpm_as = subway_2 %>% arrange(X19시.20시_ratio) %>%
  select(호선명, 지하철역, X19시.20시_ratio) %>%
  head(10)

p12 <- ggplot(data = sevenpm_as, aes(x = 호선명, y = X19시.20시_ratio, fill = 지하철역)) + geom
_col(position = "dodge") + coord_flip()
ggplotly(p12)
```





오후 7시에서 8시 사이를 보면, 직전 시간대에 비해서는 낮은 수치를 보여주고 있지만, 대공원, 경마공원(과천선)이 높은 수치를 보여주고 있고, 을지로4가(5호선), 가산디지털단지, 학동(7호선)도 높은 수치이다. 퇴근 시간대가 끝나갈 시간이기 때문에 하위 10개의 수치가 다른 시간대에 비해서 낮은 편이며 도봉산(7호선), 양천구청, 신답(2호선)이 그중에서 낮은 편이다.

주말 시간대도 포함했기 때문에 대공원, 경마공원(과천선)과 같은 테마파크의 수치가 높은 것으로 보이며 가산디지털단지(7호선)의 수치는 3개의 시간대에서 꾸준히 상위권을 지키고 있으며 까치산(5호선), 신답(2호선)은 퇴근 시간대에서 낮은 수치로 볼 수 있다.

```
gohome_de <- bind_rows(fivepm_de, sixpm_de, sevenpm_de)
gohome_as <- bind_rows(fivepm_as, sixpm_as, sevenpm_as)

table(droplevels(gohome_de$지하철역))
```

```
##
##      가산디지털단지             경마공원           국회의사당
##                   3                    3                    3
##             대공원              동대문  동대문역사문화공원
##                   3                    1                    1
##             서울역                 시청                 안국
##                   2                    5                    1
##             여의도           을지로4가                 학동
##                   1                    2                    2
##   회현(남대문시장)
##                   3
```

```
table(droplevels(gohome_as$지하철역))
```

```
##
##       개봉      까치산      도봉산      독바위        마들        명일
##          1           3           1           3           3           1
##     상일동        석계        신갈        신답  신정네거리   양천구청
##          1           1           1           3           2           3
##       역촌        응암        지축        행당        화곡
##          2           1           1           2           1
```

5. 분석 결론

전체적으로 본다면 수치가 비슷하지만, 서울역(경부선, 1호선)에서는 하차 수와 비교해서 승차 수가 현저히 많아서, 이곳에서 승차한다면 편하지 않을 것이라고 예상이 된다. 한편, 선릉(분당선), 서울역(4호선), 고속터미널(9호선)은 하차 수가 승차 수보다 많아서 이 곳에서 승차한다면 비교적 편하게 갈 것이라고 볼 수 있다.

출근 시간대 즉, 평일과 주말 오전 6시에서 9시까지의 시간대에 까치산(5호선), 마들(7호선), 삼산체육관(7호선), 신정(5호선), 쌍문(4호선), 암사(8호선), 장암(7호선), 행당(5호선), 화곡(5호선), 화랑대(6호선), 나열된 이 역들에서 승차한다면 사람이 많아 비교적 무리하게 승차할 확률이 높고, 광화문(5호선), 국회의사당(9호선), 선릉(분당선), 시청(2호선), 여의도(9호선), 을지로3가(3호선, 2호선), 을지로4가(5호선), 을지로입구(2호선), 종각(1호선) 나열된 역에서 승차한다면 비교적 편하게 갈 확률이 높다.

퇴근 시간대 즉, 평일과 주말 오후 5시에서 8시까지의 시간대에는 가산디지털단지(7호선), 경마공원, 대공원(과천선, 4호선), 서울역(경부선, 1호선), 시청역(2호선, 1호선), 을지로4가(5호선), 학동(7호선), 회현(4호선), 이렇게 나열된 역에서 승차한다면 비교적 사람이 많아 불편하게 타고 갈 확률이 높고, 까치산(5호선), 독바위(6호선), 마들(7호선), 신답(2호선), 신정네거리(2호선), 양천구청(2호선), 역촌(6호선), 행당(5호선) 이 역들에서 승차한다면 상대적으로 편하게 갈 확률이 높다.