시각화

박찬영

2024-08-19

tidyverse와 nycflights13 library를 사용합니다.

```r
fl=flights
head(fl)
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## 4  2013     1     1      544            545        -1     1004           1022
## 5  2013     1     1      554            600        -6      812            837
## 6  2013     1     1      554            558        -4      740            728
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

# 데이터 필터링

```r
filter(fl, month==1, day==1) #1월 1일 데이터만 남기기
```

```
## # A tibble: 842 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 832 more rows
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
x=c(NA, 1, NA) #NA는 결측치입니다.
is.na(x) #결측치를 확인하는 함수
```

```
## [1]  TRUE FALSE  TRUE
```

```
df = tibble(x=c(1, NA, 3))
filter(df, x>1)
```

```
## # A tibble: 1 x 1
##       x
##   <dbl>
## 1     3
```

```
filter(df, is.na(x) | x>1) #is.na 사용법
```

```
## # A tibble: 2 x 1
##       x
##   <dbl>
## 1    NA
## 2     3
```

어찌보면 filter 함수는 bool 벡터값을 이용하는 것 같다.

다음은 데이터 정렬이다

```
arrange(fl, year, month, day) #우선순위 따라 기본은 오름차순
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 336,766 more rows
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
tmp= arrange(fl, desc(arr_delay)) #내림차순 하는법

arrange(df, x) #결측치는 항상 마지막
```

```
## # A tibble: 3 x 1
##       x
##   <dbl>
## 1     1
## 2     3
## 3    NA
```

열을 골라보자

```
select(fl, dep_delay, arr_delay) #원하는 열을 고르기
```

```
## # A tibble: 336,776 x 2
##    dep_delay arr_delay
##        <dbl>     <dbl>
##  1         2        11
##  2         4        20
##  3         2        33
##  4        -1       -18
##  5        -6       -25
##  6        -4        12
##  7        -5        19
##  8        -3       -14
##  9        -3        -8
## 10        -2         8
## # i 336,766 more rows
```

```
select(fl, dep_time:arr_delay) #주루룩 고르기는 : 사용
```

```
## # A tibble: 336,776 x 6
##    dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
##       <int>          <int>     <dbl>    <int>          <int>     <dbl>
##  1      517            515         2      830            819        11
##  2      533            529         4      850            830        20
##  3      542            540         2      923            850        33
##  4      544            545        -1     1004           1022       -18
##  5      554            600        -6      812            837       -25
##  6      554            558        -4      740            728        12
##  7      555            600        -5      913            854        19
##  8      557            600        -3      709            723       -14
##  9      557            600        -3      838            846        -8
## 10      558            600        -2      753            745         8
## # i 336,766 more rows
```

```
select(fl, -(dep_time)) # - 달면 개 빼고
```

```
## # A tibble: 336,776 x 18
##     year month   day sched_dep_time dep_delay arr_time sched_arr_time arr_delay
##    <int> <int> <int>          <int>     <dbl>    <int>          <int>     <dbl>
##  1  2013     1     1            515         2      830            819        11
##  2  2013     1     1            529         4      850            830        20
##  3  2013     1     1            540         2      923            850        33
##  4  2013     1     1            545        -1     1004           1022       -18
##  5  2013     1     1            600        -6      812            837       -25
##  6  2013     1     1            558        -4      740            728        12
##  7  2013     1     1            600        -5      913            854        19
##  8  2013     1     1            600        -3      709            723       -14
##  9  2013     1     1            600        -3      838            846        -8
## 10  2013     1     1            600        -2      753            745         8
## # i 336,766 more rows
## # i 10 more variables: carrier <chr>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>
```

```
select(fl, c(1,3,4,5)) #벡터로 직관적으로 구할 수 있음.
```

```
## # A tibble: 336,776 x 4
##     year   day dep_time sched_dep_time
##    <int> <int>    <int>          <int>
##  1  2013     1      517            515
##  2  2013     1      533            529
##  3  2013     1      542            540
##  4  2013     1      544            545
##  5  2013     1      554            600
##  6  2013     1      554            558
##  7  2013     1      555            600
##  8  2013     1      557            600
##  9  2013     1      557            600
## 10  2013     1      558            600
## # i 336,766 more rows
```

```
select(fl, time_hour, everything()) #순서 체인지 같은것도 가능
```

```
## # A tibble: 336,776 x 19
##    time_hour            year month   day dep_time sched_dep_time dep_delay
##    <dttm>              <int> <int> <int>    <int>          <int>     <dbl>
##  1 2013-01-01 05:00:00  2013     1     1      517            515         2
##  2 2013-01-01 05:00:00  2013     1     1      533            529         4
##  3 2013-01-01 05:00:00  2013     1     1      542            540         2
##  4 2013-01-01 05:00:00  2013     1     1      544            545        -1
##  5 2013-01-01 06:00:00  2013     1     1      554            600        -6
##  6 2013-01-01 05:00:00  2013     1     1      554            558        -4
##  7 2013-01-01 06:00:00  2013     1     1      555            600        -5
##  8 2013-01-01 06:00:00  2013     1     1      557            600        -3
##  9 2013-01-01 06:00:00  2013     1     1      557            600        -3
## 10 2013-01-01 06:00:00  2013     1     1      558            600        -2
## # i 336,766 more rows
## # i 12 more variables: arr_time <int>, sched_arr_time <int>, arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>
```

```
fl_sml = select(fl, year:day, ends_with("delay"), distance, air_time) #응용

rename(fl, dt=dep_time) #이름바꾸기 A로 B를 바꾼다의 문법
```

```
## # A tibble: 336,776 x 19
##     year month   day    dt sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int> <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1   517            515         2      830            819
##  2  2013     1     1   533            529         4      850            830
##  3  2013     1     1   542            540         2      923            850
##  4  2013     1     1   544            545        -1     1004           1022
##  5  2013     1     1   554            600        -6      812            837
##  6  2013     1     1   554            558        -4      740            728
##  7  2013     1     1   555          600        -5      913            854
##  8  2013     1     1   557          600        -3      709            723
##  9  2013     1     1   557          600        -3      838            846
## 10  2013     1     1   558          600        -2      753            745
## # i 336,766 more rows
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

데이터의 추가

```
mutate(fl_sml, gain=arr_delay - dep_delay, speed = distance/air_time*60)
```

```
## # A tibble: 336,776 x 9
##     year month   day dep_delay arr_delay distance air_time  gain speed
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl> <dbl> <dbl>
##  1  2013     1     1         2        11     1400      227     9  370.
##  2  2013     1     1         4        20     1416      227    16  374.
##  3  2013     1     1         2        33     1089      160    31  408.
##  4  2013     1     1        -1       -18     1576      183   -17  517.
##  5  2013     1     1        -6       -25      762      116   -19  394.
##  6  2013     1     1        -4        12      719      150    16  288.
##  7  2013     1     1        -5        19     1065      158    24  404.
##  8  2013     1     1        -3       -14      229       53   -11  259.
##  9  2013     1     1        -3        -8      944      140    -5  405.
## 10  2013     1     1        -2         8      733      138    10  319.
## # i 336,766 more rows
```

```
mutate(fl_sml, gain=arr_delay - dep_delay, hours=air_time/60, gain_per_hour = gain/hours) #방금 만든거를
```

```
## # A tibble: 336,776 x 10
##     year month   day dep_delay arr_delay distance air_time  gain hours
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl> <dbl> <dbl>
##  1  2013     1     1         2        11     1400      227     9 3.78
##  2  2013     1     1         4        20     1416      227    16 3.78
##  3  2013     1     1         2        33     1089      160    31 2.67
##  4  2013     1     1        -1       -18     1576      183   -17 3.05
##  5  2013     1     1        -6       -25      762      116   -19 1.93
##  6  2013     1     1        -4        12      719      150    16 2.5
##  7  2013     1     1        -5        19     1065      158    24 2.63
##  8  2013     1     1        -3       -14      229       53   -11 0.883
##  9  2013     1     1        -3        -8      944      140    -5 2.33
## 10  2013     1     1        -2         8      733      138    10 2.3
## # i 336,766 more rows
## # i 1 more variable: gain_per_hour <dbl>
```

```
transmute(fl_sml, gain=arr_delay - dep_delay, hours=air_time/60, gain_per_hour = gain/hours) #새거만 남
```

```
## # A tibble: 336,776 x 3
##     gain hours gain_per_hour
##    <dbl> <dbl>         <dbl>
##  1     9 3.78           2.38
##  2    16 3.78           4.23
##  3    31 2.67          11.6
##  4   -17 3.05          -5.57
##  5   -19 1.93          -9.83
##  6    16 2.5            6.4
##  7    24 2.63           9.11
##  8   -11 0.883        -12.5
##  9    -5 2.33          -2.14
## 10    10 2.3            4.35
## # i 336,766 more rows
```

데이터 요약

```
summarise(fl, delay=mean(dep_delay, na.rm=TRUE), maxd=max(dep_delay, na.rm=TRUE), mind=min(dep_delay, na
```

```
## # A tibble: 1 x 3
##   delay  maxd  mind
##   <dbl> <dbl> <dbl>
## 1  12.6  1301   -43
```

```
#group_by는 같은 값끼리 데이터프레임열을 만들어주는데 같이 쓰기 좋음

a=fl %>%
group_by(year, month, day) %>% #연 월 일 별로 다 데이터프레임을 쪼갬, 순서가 중요함
summarise(delay=mean(dep_delay, na.rm=TRUE)) #각 데이터프레임에서 평균을 냄
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
not_cancelled <- fl %>% filter(!is.na(dep_delay), !is.na(arr_delay))
not_cancelled %>%
      group_by(year, month, day) %>%
```

```r
      summarise(
        first_dep = min(dep_time),
        last_dep = max(dep_time)
      ) #예제
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 365 x 5
## # Groups:   year, month [12]
##     year month   day first_dep last_dep
##    <int> <int> <int>     <int>    <int>
##  1  2013     1     1       517     2356
##  2  2013     1     2        42     2354
##  3  2013     1     3        32     2349
##  4  2013     1     4        25     2358
##  5  2013     1     5        14     2357
##  6  2013     1     6        16     2355
##  7  2013     1     7        49     2359
##  8  2013     1     8       454     2351
##  9  2013     1     9         2     2252
## 10  2013     1    10         3     2320
## # i 355 more rows
```

```r
not_cancelled %>% group_by(year, month, day) %>%
    summarise(hour_perc=length(arr_delay[arr_delay >60])/length(arr_delay)) #개못함
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day hour_perc
##    <int> <int> <int>     <dbl>
##  1  2013     1     1    0.0722
##  2  2013     1     2    0.0851
##  3  2013     1     3    0.0567
##  4  2013     1     4    0.0396
##  5  2013     1     5    0.0349
##  6  2013     1     6    0.0470
##  7  2013     1     7    0.0333
##  8  2013     1     8    0.0213
##  9  2013     1     9    0.0202
## 10  2013     1    10    0.0183
## # i 355 more rows
```

```
not_cancelled %>% group_by(year, month, day) %>%
    summarise(hour_perc=mean(arr_delay>60)) #개천재 벡터와 부울 변수를 존나 잘씀
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day hour_perc
##    <int> <int> <int>     <dbl>
##  1  2013     1     1    0.0722
##  2  2013     1     2    0.0851
##  3  2013     1     3    0.0567
##  4  2013     1     4    0.0396
##  5  2013     1     5    0.0349
##  6  2013     1     6    0.0470
##  7  2013     1     7    0.0333
##  8  2013     1     8    0.0213
##  9  2013     1     9    0.0202
## 10  2013     1    10    0.0183
## # i 355 more rows
```

```
fl_sml %>%
      group_by(year, month, day) %>%
      filter(rank(desc(arr_delay)) < 10) #이런것도 가능 그룹바이 굿굿
```

```
## # A tibble: 3,306 x 7
## # Groups:   year, month, day [365]
##     year month   day dep_delay arr_delay distance air_time
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl>
##  1  2013     1     1       853       851      184       41
##  2  2013     1     1       290       338     1134      213
##  3  2013     1     1       260       263      266       46
##  4  2013     1     1       157       174      213       60
##  5  2013     1     1       216       222      708      121
##  6  2013     1     1       255       250      589      115
##  7  2013     1     1       285       246     1085      146
##  8  2013     1     1       192       191      199       44
##  9  2013     1     1       379       456     1092      222
## 10  2013     1     2       224       207      550       94
## # i 3,296 more rows
```

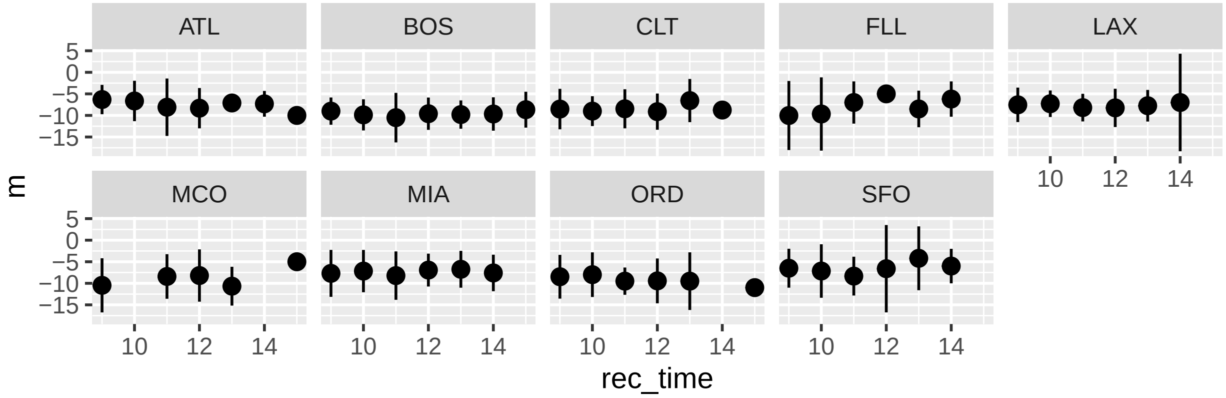
```
pop = not_cancelled %>% group_by(dest) %>% filter(n()>10000)

summer= pop %>% ungroup() %>%
    select(year:day, dep_time, sched_dep_time, dep_delay, dest) %>%
    filter(month<9, month>5, dep_time>=900, dep_time<=1500)

rs=summer %>% group_by(dest, year, month, day) %>%
    arrange(dep_delay) %>%
    summarise(min_delay = first(dep_delay), rec_time=first(sched_dep_time%/%100))
```

```
## `summarise()` has grouped output by 'dest', 'year', 'month'. You can override
## using the `.groups` argument.
```

```
rs %>% group_by(dest, rec_time) %>% summarise(m=mean(min_delay), sd= sd(min_delay),
                                low=m-2*sd, high=m+2*sd ) %>%
ggplot(aes(x=rec_time, y=m, ymin=low, ymax=high)) +
    geom_pointrange() +
    theme(aspect.ratio = 1/2) +
    facet_wrap(~dest, nrow=2)
```

```
## `summarise()` has grouped output by 'dest'. You can override using the
## `.groups` argument.
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_segment()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_segment()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_segment()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_segment()`).
```



```
rs %>% group_by(dest, rec_time) %>% summarise(mean_delay=mean(min_delay)) %>%
    arrange(mean_delay) %>% summarise(rec_timee=first(rec_time), mean_delay=first(mean_delay))
```

```
## `summarise()` has grouped output by 'dest'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 9 x 3
##   dest  rec_timee mean_delay
##   <chr>     <dbl>      <dbl>
## 1 ATL          15     -10
## 2 BOS          11     -10.5
## 3 CLT          12      -9.12
## 4 FLL           9     -10.0
## 5 LAX          12      -8.25
## 6 MCO          13     -10.7
## 7 MIA          11      -8.21
## 8 ORD          15     -11
## 9 SFO          11      -8.33
```

```
#오늘의 결론 summarise() 돼서 짜바리된 그룹은 사라진다. 즉 1개짜리 그룹은 그룹 취급을 안받는다.
```