전처리_지하철_박정희_0516.R

LSG

2021-05-16

```
# 1. 데이터 읽기 및 전처리
# 1-1. subway.csv(main), subway_latlong.csv(sub) 읽어와서 구조 확인
library(dplyr); library(lubridate); library(data.table); library(ggplot2)
```

```
##
## 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
```

```
##
## Attaching package: 'lubridate'
```

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

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(reshape2); library(plyr); library(funModeling)
```

```
##
## Attaching package: 'reshape2'
## The following objects are masked from 'package:data.table':
## 
##     dcast, melt
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
## 
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
## 
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## Warning: package 'funModeling' was built under R version 4.0.5
```

```
## Loading required package: Hmisc
```

```
## Warning: package 'Hmisc' was built under R version 4.0.4
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
## 
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:plyr':
## 
##     is.discrete, summarize
```

```
## The following objects are masked from 'package:dplyr':
## 
##     src, summarize
```

```
## The following objects are masked from 'package:base':
## 
##     format.pval, units
## funModeling v.1.9.4 :)
## Examples and tutorials at livebook.datascienceheroes.com
##  / Now in Spanish: librovivodecienciadedatos.ai
```

```
rm(list=ls())
setwd("D:/ADP실기/5월빅분기실기/전처리_지하철_0516")
subway <- fread("subway.csv", stringsAsFactors = TRUE, encoding="UTF-8")

# 1-2. income_date를 date타입으로 변경하고 다른 컬럼들 타입 정리
subway$station <- subway$station %>%  as.factor()
subway$income_date <-  subway$income_date %>%  ymd()
tmp <- subset(subway, select = on_tot:off_24) %>% mutate_if(is.factor, as.numeric)
subway <- bind_cols(subway[ , 1:3], tmp)

# 1-3. 7월까지 밖에 없는 income_date 2014년도 데이터 제외하기
subway$income_date %>% range()
```

```
## [1] "2010-01-01" "2014-07-31"
```

```
subway <- subway %>%  filter(income_date < "2014-01-1")
subway$income_date %>% range()
```

```
## [1] "2010-01-01" "2013-12-31"
```

```
# 1-4. 다른 호선의 같은 역명(stat_name뒤에 괄호)을 하나의 역명으로 처리 ex) 천호(5),천호(8) -> 천호
subway$stat_name <-  subway$stat_name %>%  as.character()
subway$stat_name %>% tail()
```

```
## [1] "모란(8)" "모란(8)" "모란(8)" "모란(8)" "모란(8)" "모란(8)"
```

```
gsub("[:(:][0-9][:):]", "", "모란(8)")
```

```
## [1] "모란"
```

```
subway$stat_name <-  gsub("[:(:][0-9][:):]", "", subway$stat_name)
subway$stat_name <- subway$stat_name %>%  as.factor()

# 1-5. income_date에서 추출한 연,월 컬럼 추가

subway$year  <- subway$income_date %>% year()
subway$month <- subway$income_date %>% month()

# 2. 탑승객 상위 5위 역 구하고 해당 탑승객수 출력 및 호선 정보 출력
# 2-1. stat_name 기준으로 탑승객(on_tot) 상위 5개 출력
aggregate(on_tot ~ stat_name, subway, sum)  %>%  arrange(desc(on_tot)) %>%  head(5)
```

```
##        stat_name   on_tot
## 1           천호 62506080
## 2 가산디지털단지 51204299
## 3         광화문 47791232
## 4           화곡 44025075
## 5         까치산 42827345
```

```
# 2-2. stat_name 기준으로 left join으로 sub파일 내 STATION_NM과 조인해서 역 호선 정보(LINE_NUM) 출력
subway_latlong <- fread("subway_latlong.csv", stringsAsFactors = TRUE)

subway %>%  dim() # 220110     47
```

```
## [1] 220110     47
```

```
subway_latlong %>%  dim() #  539   9
```

```
## [1] 539   9
```

```
subway_merge <- merge(subway, subway_latlong, by.x = "stat_name" , by.y = "STATION_NM", all.x=TRUE)
subway_merge  %>% dim() # 220110     55
```

```
## [1] 220110     55
```

```
subway_merge %>%  colnames()
```

```
##  [1] "stat_name"     "station"       "income_date"   "on_tot"       
##  [5] "on_05"         "on_06"         "on_07"         "on_08"        
##  [9] "on_09"         "on_10"         "on_11"         "on_12"        
## [13] "on_13"         "on_14"         "on_15"         "on_16"        
## [17] "on_17"         "on_18"         "on_19"         "on_20"        
## [21] "on_21"         "on_22"         "on_23"         "on_24"        
## [25] "off_tot"       "off_05"        "off_06"        "off_07"       
## [29] "off_08"        "off_09"        "off_10"        "off_11"       
## [33] "off_12"        "off_13"        "off_14"        "off_15"       
## [37] "off_16"        "off_17"        "off_18"        "off_19"       
## [41] "off_20"        "off_21"        "off_22"        "off_23"       
## [45] "off_24"        "year"          "month"         "STATION_CD"   
## [49] "LINE_NUM"      "FR_CODE"       "CYBER_ST_CODE" "XPOINT"       
## [53] "YPOINT"        "XPOINT_WGS"    "YPOINT_WGS"
```

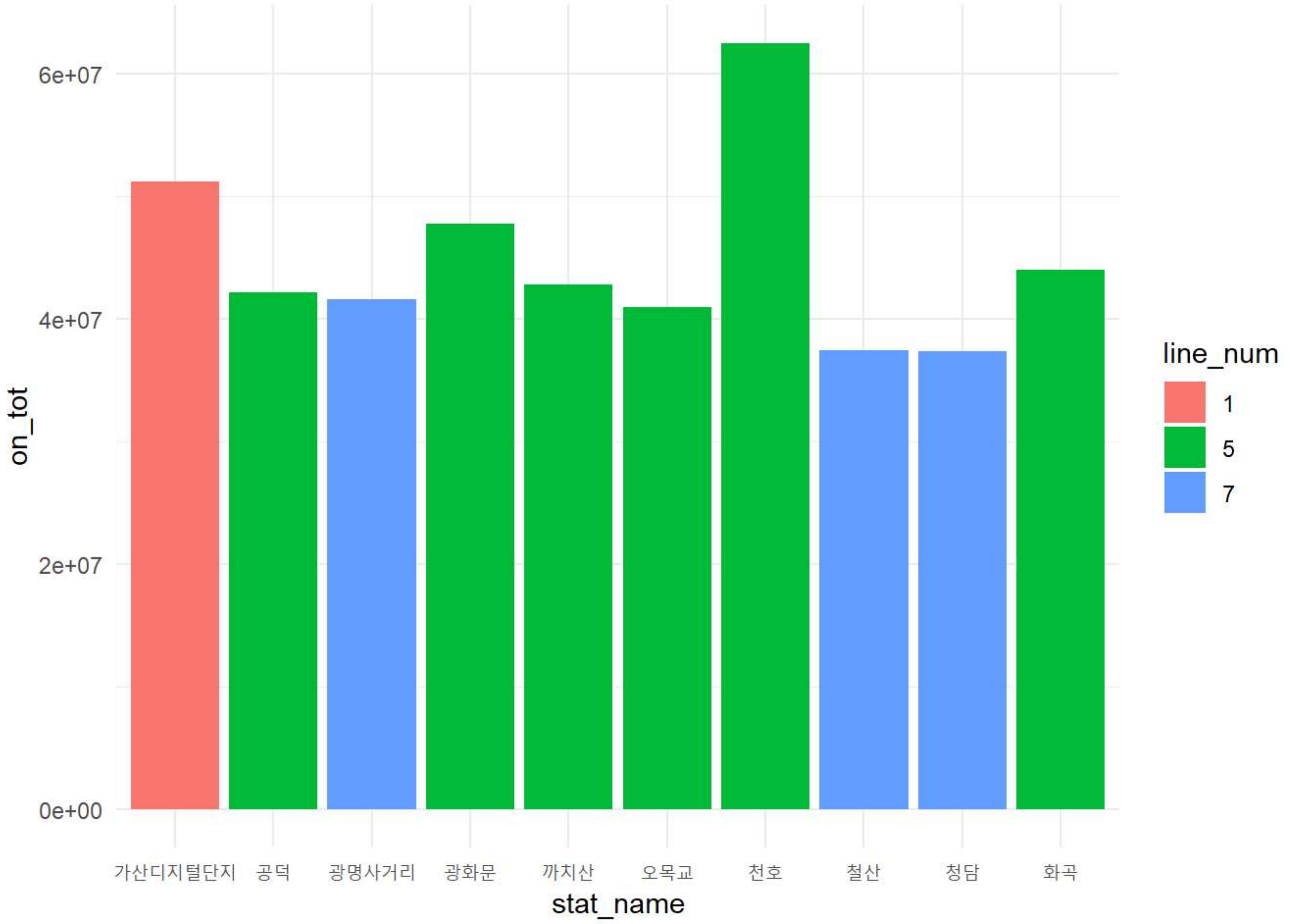
```
# 2-3. 노선별로 정렬하기(LINE_NUM)
subway_merge <- subway_merge %>% janitor::clean_names()
subway_merge %>%  group_by(line_num) %>% arrange(line_num) %>% head()
```

```
## # A tibble: 6 x 55
## # Groups:   line_num [1]
##   stat_name station income_date on_tot on_05 on_06 on_07 on_08 on_09 on_10 on_11
##   <fct>     <fct>   <date>       <int> <dbl> <int> <int> <int> <int> <int> <int>
## 1 가산디지털단지~ 2748    2010-01-01    7499   680   131   132   182   213   188   191
## 2 가산디지털단지~ 2748    2010-01-02   13201   891    85   168   287   264   233   306
## 3 가산디지털단지~ 2748    2010-01-03   10238   780   118   135   220   236   236   231
## 4 가산디지털단지~ 2748    2010-01-04   51184    68   307   708   967   749   764   858
## 5 가산디지털단지~ 2748    2010-01-05   47852    61   341   825   915   752   797   774
## 6 가산디지털단지~ 2748    2010-01-06   44481    82   287   896   853   700   759   743
## # ... with 44 more variables: on_12 <int>, on_13 <int>, on_14 <int>,
## #   on_15 <int>, on_16 <int>, on_17 <int>, on_18 <int>, on_19 <dbl>,
## #   on_20 <dbl>, on_21 <dbl>, on_22 <dbl>, on_23 <dbl>, on_24 <dbl>,
## #   off_tot <int>, off_05 <dbl>, off_06 <int>, off_07 <dbl>, off_08 <int>,
## #   off_09 <int>, off_10 <int>, off_11 <int>, off_12 <int>, off_13 <int>,
## #   off_14 <int>, off_15 <int>, off_16 <int>, off_17 <int>, off_18 <int>,
## #   off_19 <dbl>, off_20 <dbl>, off_21 <dbl>, off_22 <dbl>, off_23 <dbl>,
## #   off_24 <dbl>, year <int>, month <int>, station_cd <int>, line_num <fct>,
## #   fr_code <fct>, cyber_st_code <int>, xpoint <int>, ypoint <int>,
## #   xpoint_wgs <dbl>, ypoint_wgs <dbl>
```

```
# 2-4. 역이름 factor타입으로 변경
subway_merge$stat_name %>%  class()
```

```
## [1] "factor"
```

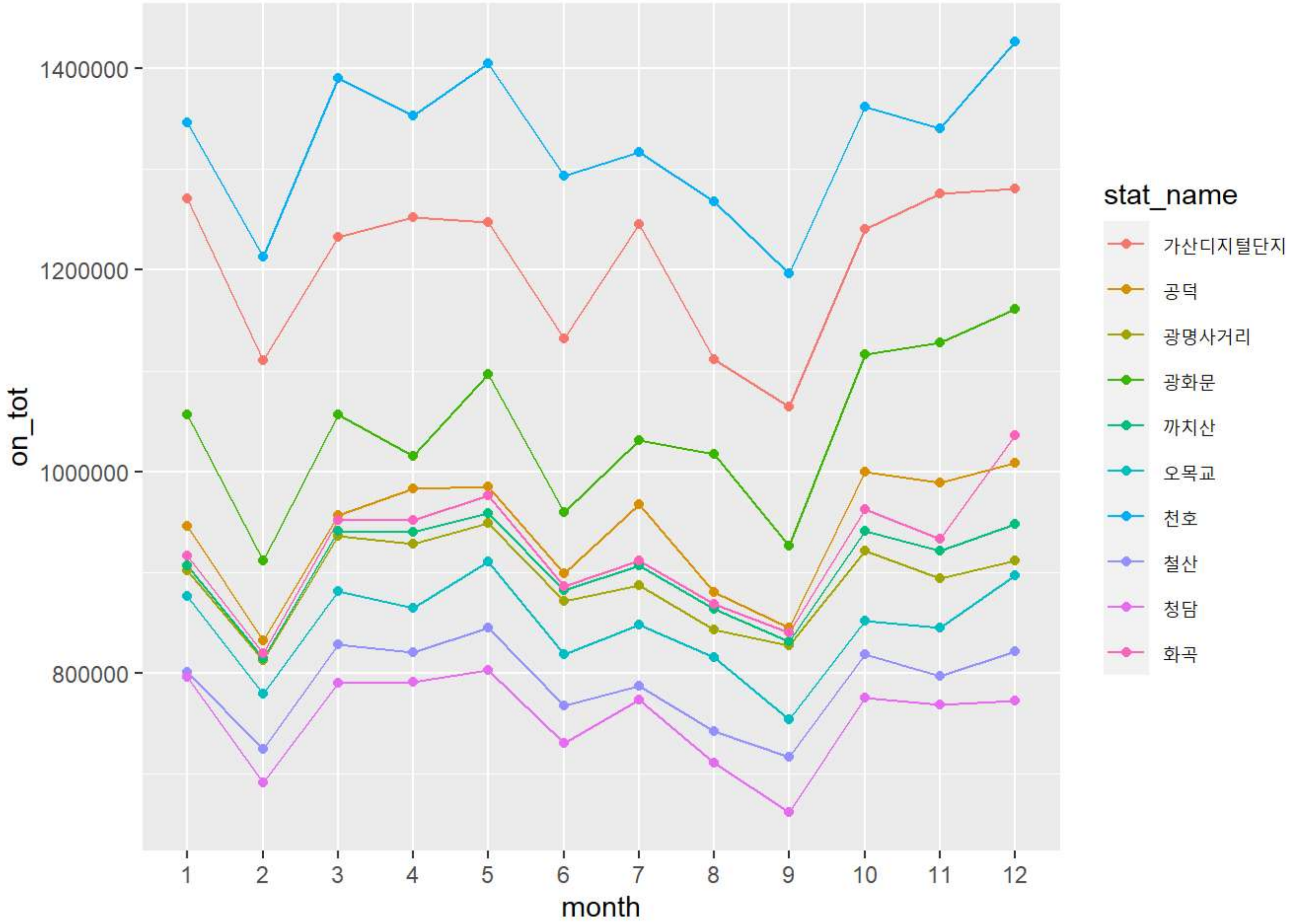
```
# 3. 탑승객 수 상위 10개 역 구하고, x축:역(stat_name)별 y축:탑승객 수(on_tot) 막대그래프 그리기
(노선별로는 색으로 구분)

top10 <- aggregate(on_tot ~ stat_name + line_num, subway_merge, sum) %>%
  arrange(desc(on_tot)) %>% head(10)
ggplot(top10) + aes(x = stat_name, y = on_tot, fill = line_num) +
    geom_bar(stat="identity") + scale_fill_hue() + theme_minimal()
```



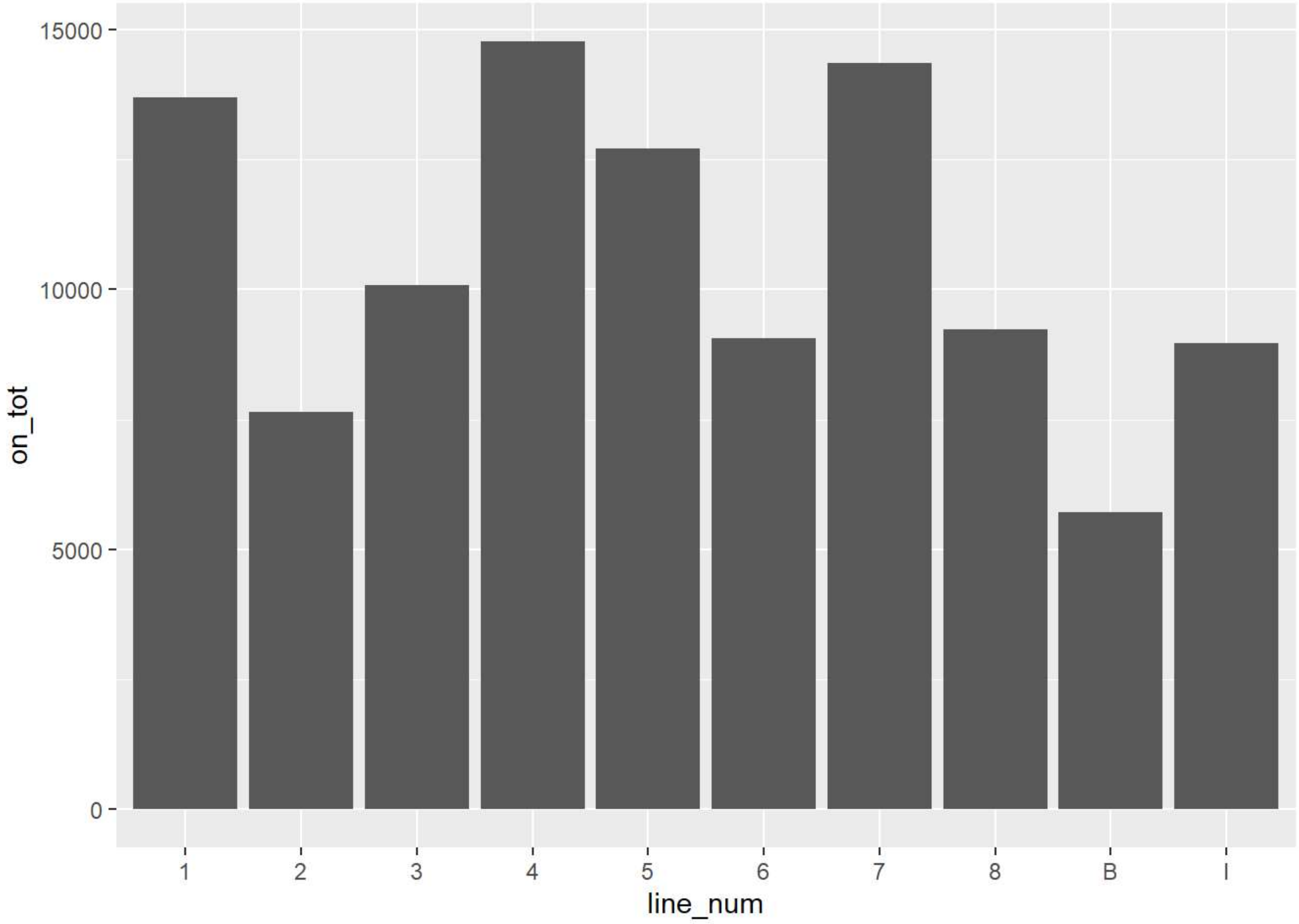
```
# 4. 탑승객 상위 10개역의 2013년도 월별 역별 승객 수 구하고 추이도 그래프 그리기. (x=월(month),
y=탑승객(on_tot), line=역(stat_name) )

tmp <- filter(subway_merge, stat_name %in% top10$stat_name ) %>%  filter(year=="2013")
tmp <- aggregate(on_tot ~ stat_name + month, tmp, sum)
tmp %>%  head()
```

```
##        stat_name month  on_tot
## 1 가산디지털단지     1 1271206
## 2           공덕     1  945549
## 3     광명사거리     1  902068
## 4         광화문     1 1056897
## 5         까치산     1  906514
## 6         오목교     1  876151
```

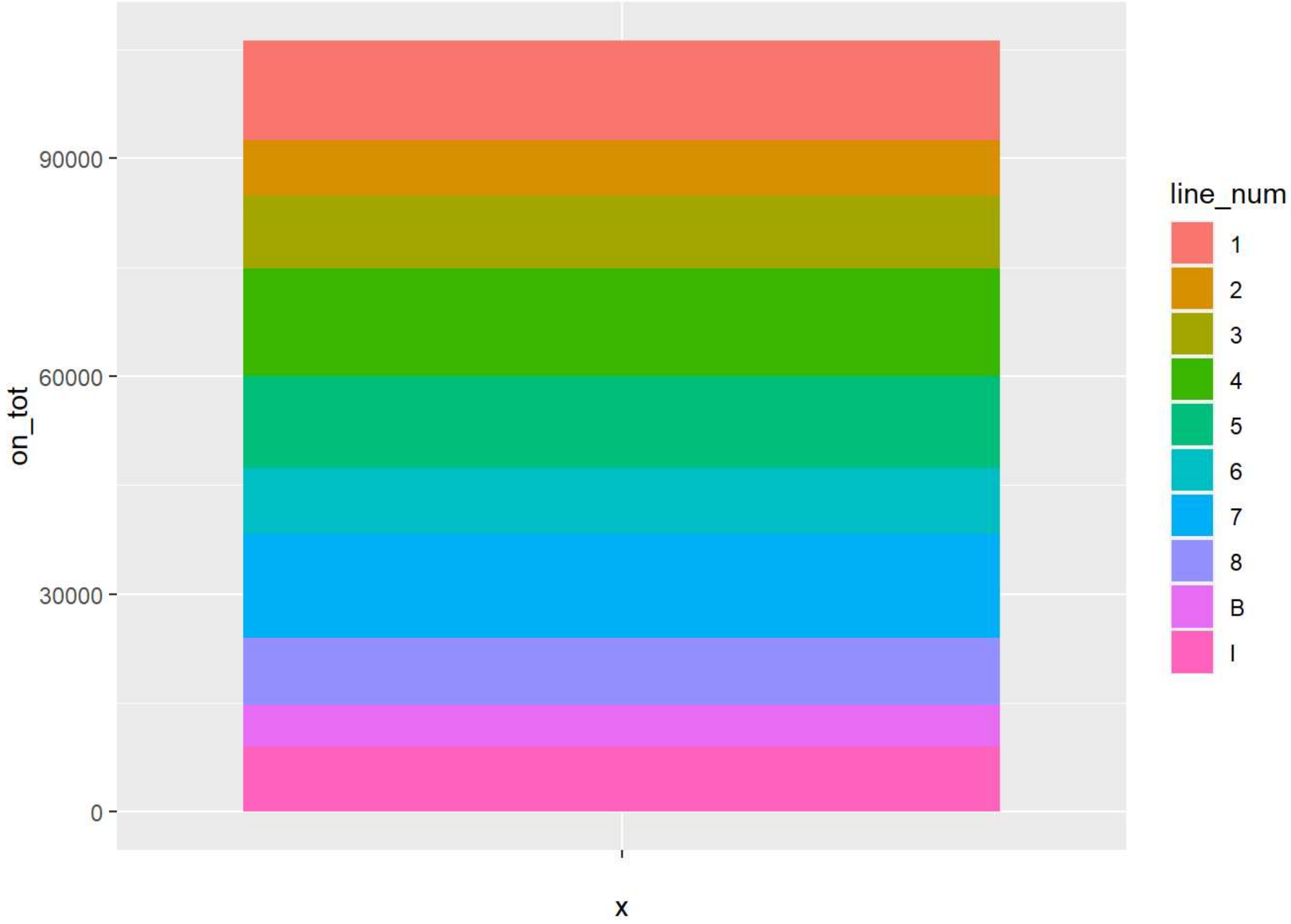
```
tmp$month  <-  tmp$month  %>%  as.factor()

ggplot(tmp) + aes(x=month,y=on_tot, colour=stat_name, group=stat_name) +
  geom_line() + geom_point()
```



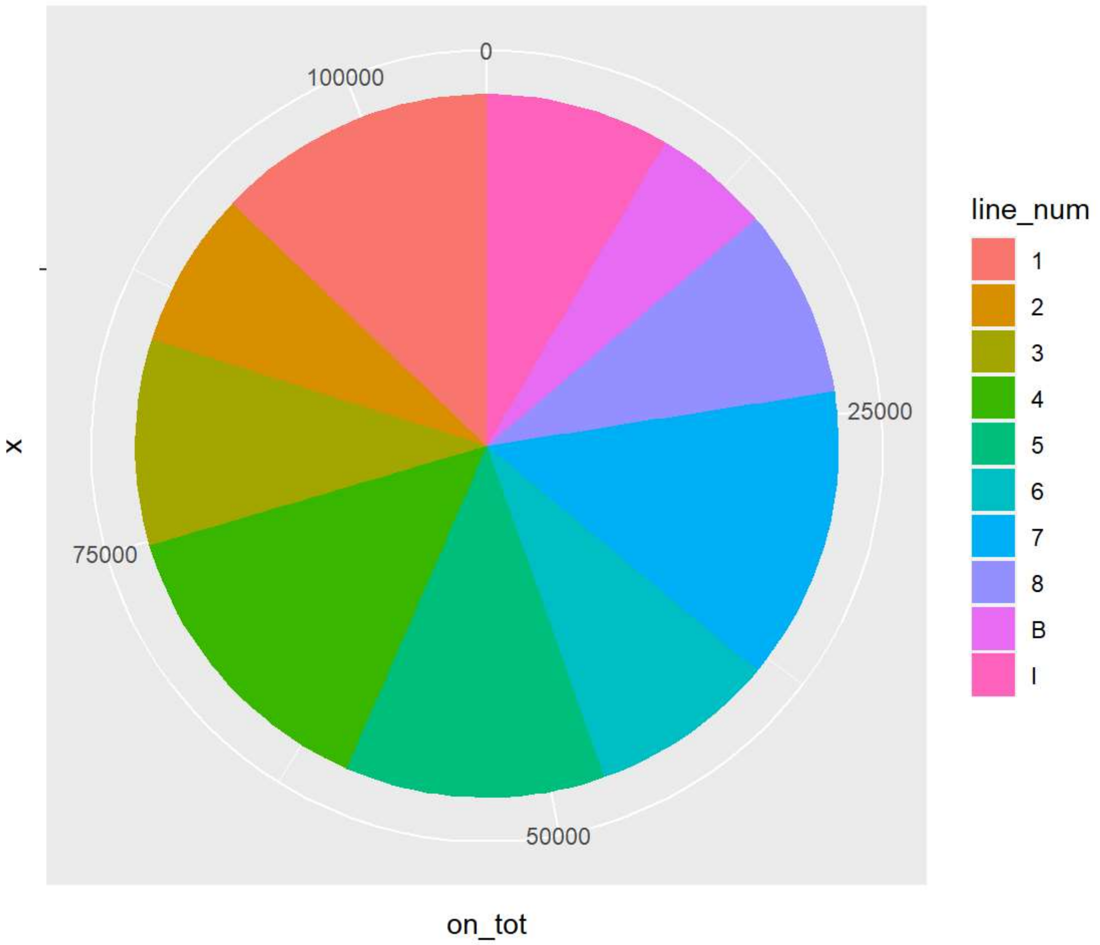
```
#
# 5. 노선별 평균 지하철 탑승객 수 구하고 파이차트 그리기
#
tmp2 <- aggregate(on_tot ~ line_num, subway_merge, mean)

ggplot(tmp2) + aes(x=line_num, y=on_tot) + geom_bar(stat = "identity")
```



```
ggplot(tmp2) + aes(x="", y=on_tot, fill=line_num) + geom_bar(stat = "identity")
```



```
ggplot(tmp2) + aes(x="", y=on_tot, fill=line_num) + geom_bar(stat = "identity") + coord_polar(
"y", start = 0 )
```



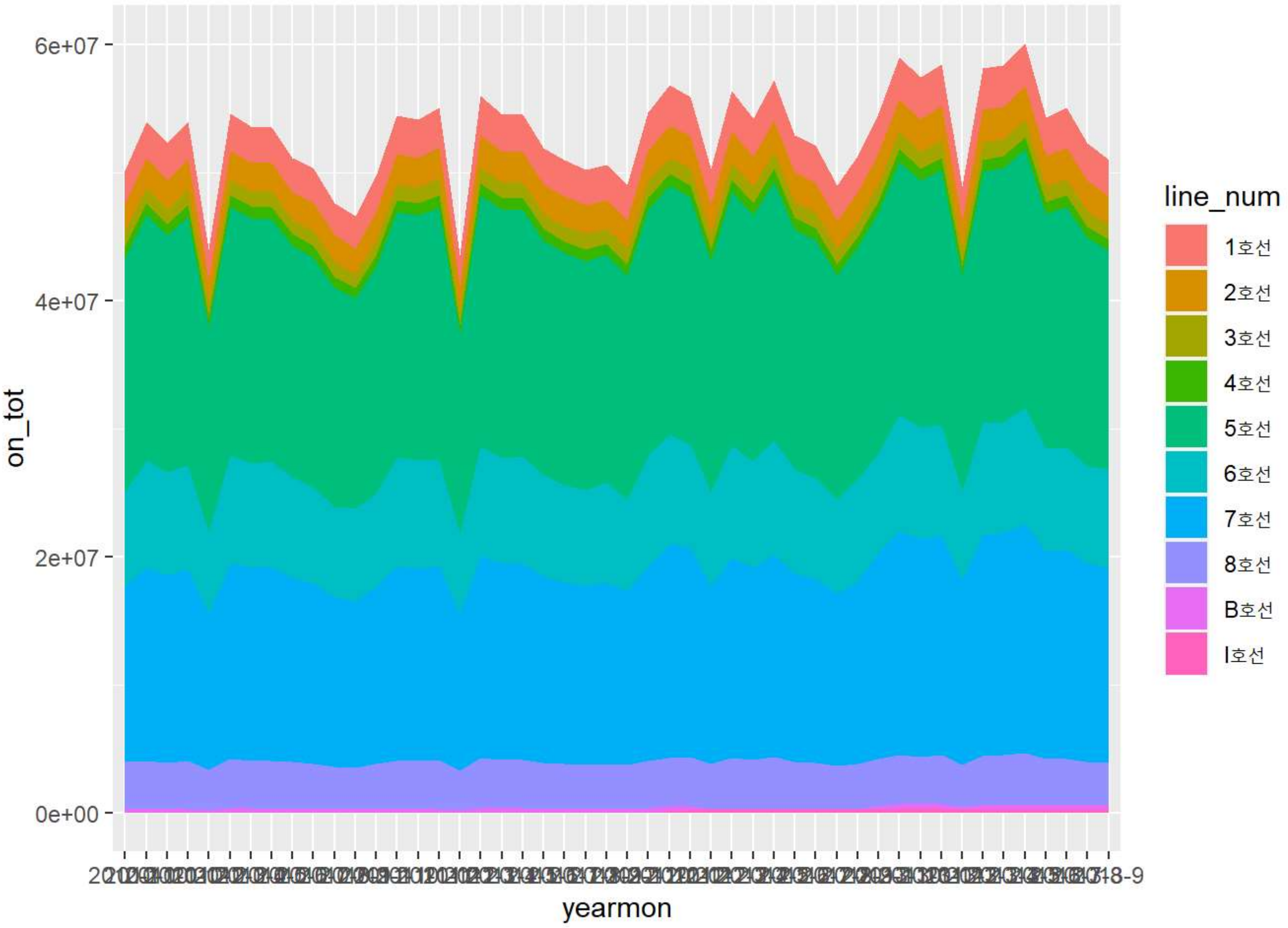
```
# 6. 노선별 누적 승객 수의 상대 비교하고 영역차트 그리기 (x축 YYYY-MM 년월, y축 누적승객수, fil
l=호선)
#
subway_merge$line_num <- paste0(subway_merge$line_num, "호선")
subway_merge$line_num <-  subway_merge$line_num %>%  as.factor()

subway_merge <- mutate(subway_merge, yearmon = paste(year,month, sep="-" ))

subway_merge %>%  is.na() %>%  sum()
```

```
## [1] 10227
```

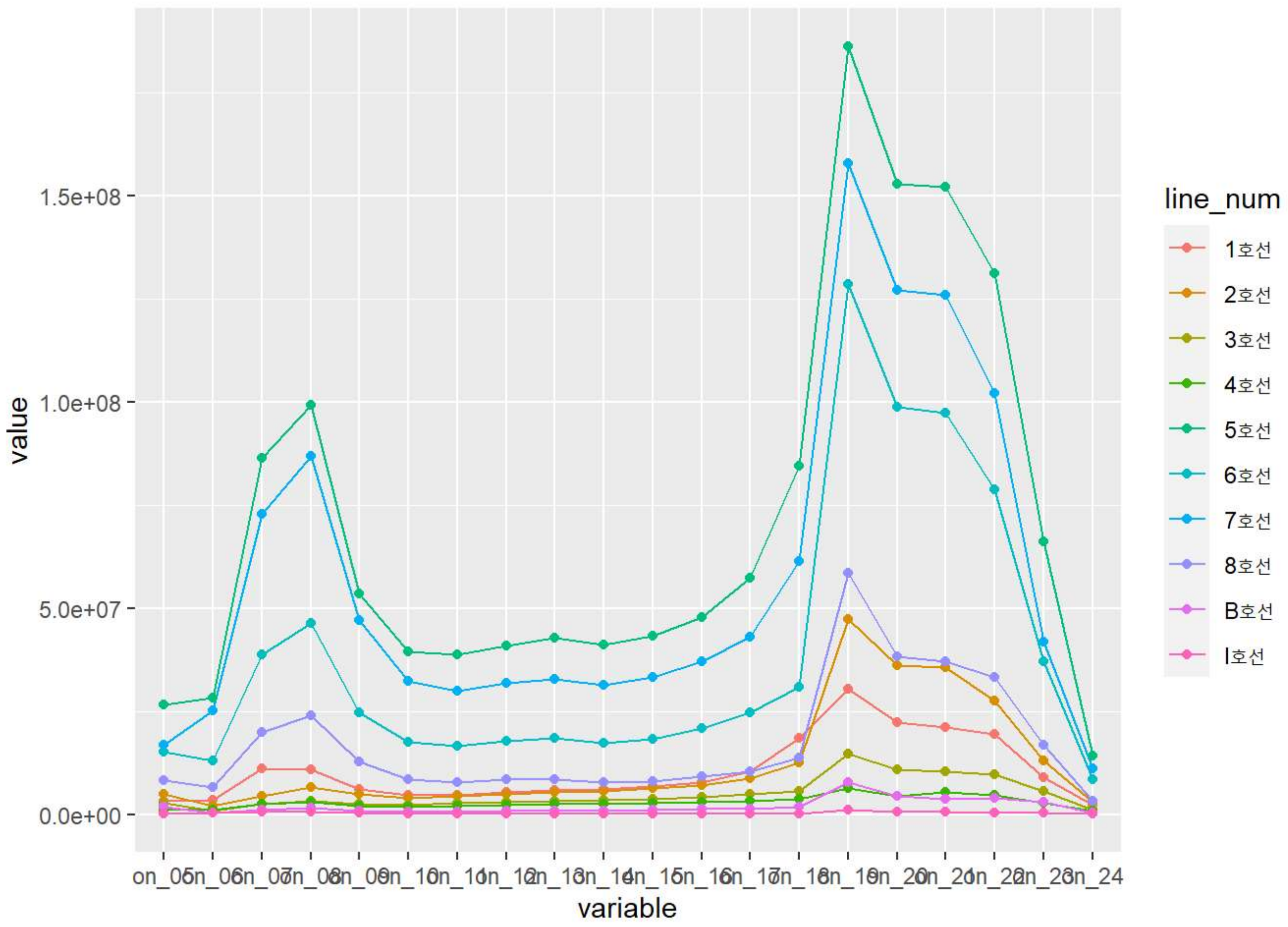
```
subway_merge <- subway_merge[complete.cases(subway_merge),]
subway_merge %>%  is.na() %>%  sum()
```

```
## [1] 0
```

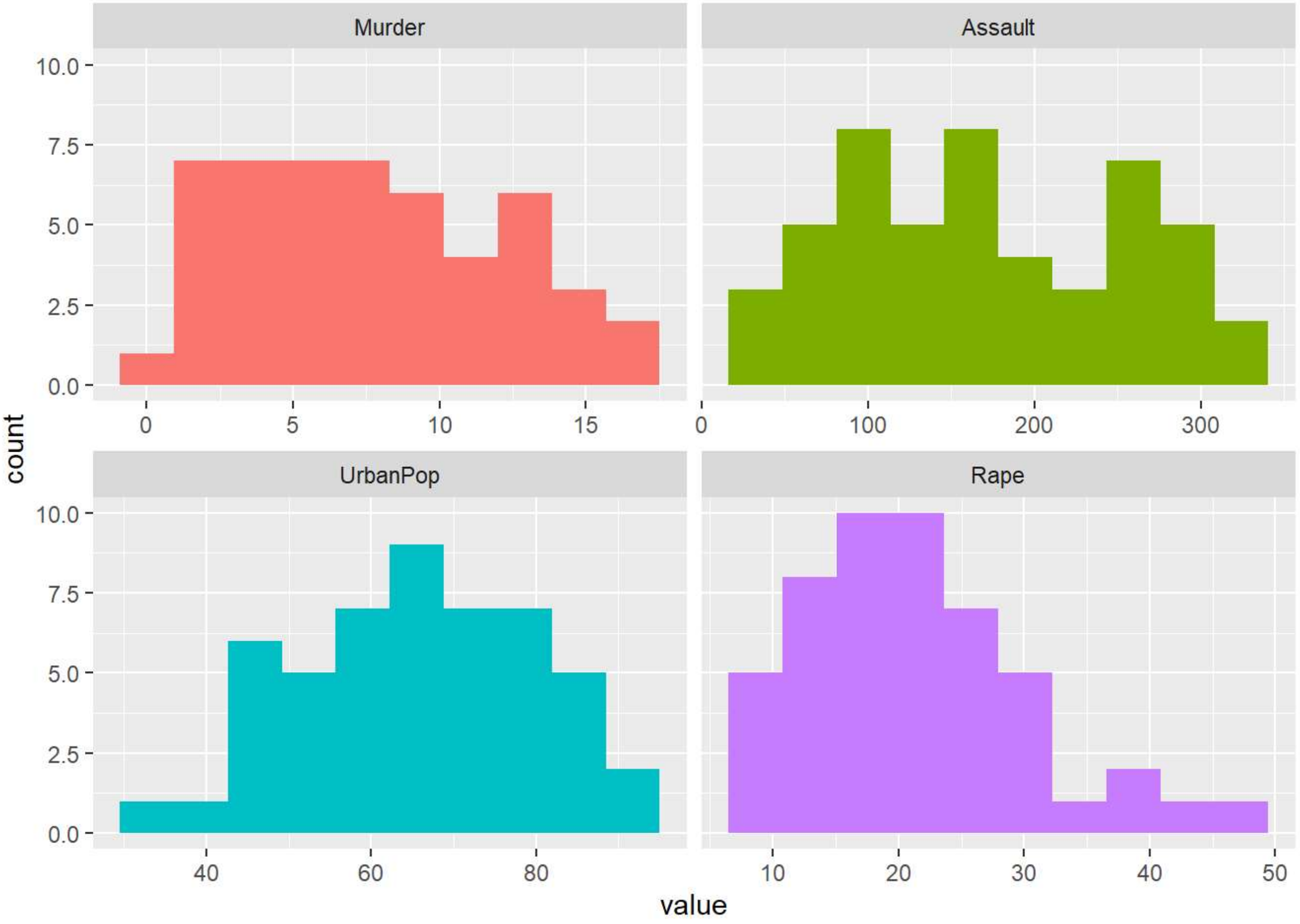
```
aggregate(on_tot ~ line_num + yearmon, subway_merge, sum) %>%
  ggplot(.) + aes(x=yearmon, y = on_tot, fill = line_num, group = line_num) + geom_area()
```



```
# 7. 시간대별 호선별 평균 탑승객(on_HH컬럼) 수의 상대 비교하고 추이도 그래프 그리기 (x축:탑승시간대(00~24), y축:탑승객수, group=호선)
# (**시간대on_HH컬럼 pivot 필요)
subway_merge %>% select(line_num, on_05:on_24) %>%  melt(., id.vars = "line_num") %>%
  dcast(., line_num ~ variable, sum) %>%  melt(., id.vars = "line_num") %>%
  ggplot(.) + aes(x = variable, y= value, color = line_num, group = line_num ) + geom_line() + geom_point()
```



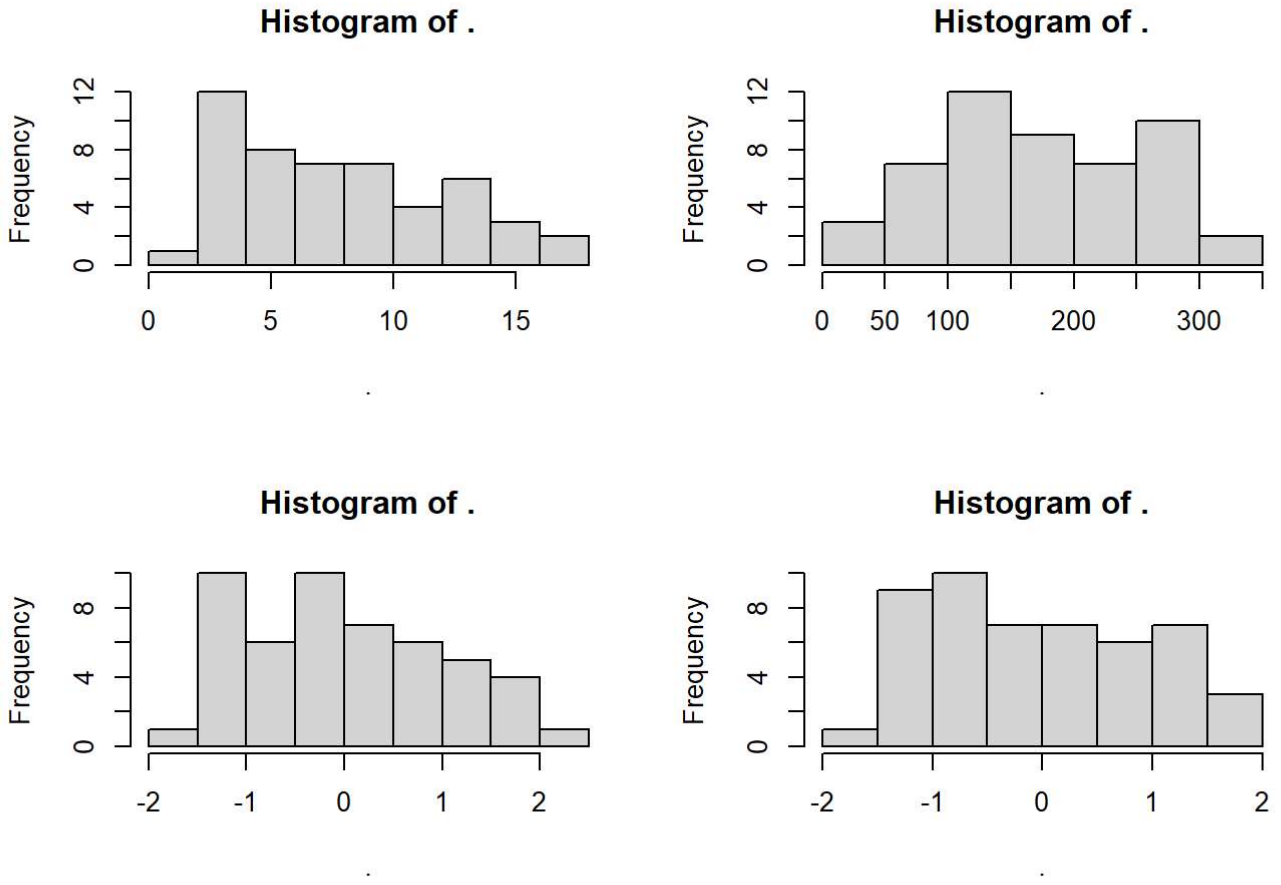
```
# 8. USArrests.csv 파일 읽어서 Feature Scaling
# 8-1. Murder, Assault 변수를 z 표준화 후 히스토그램 그리기
USArrests <- fread("USArrests.csv", stringsAsFactors = TRUE)

USArrests %>% plot_num()
```



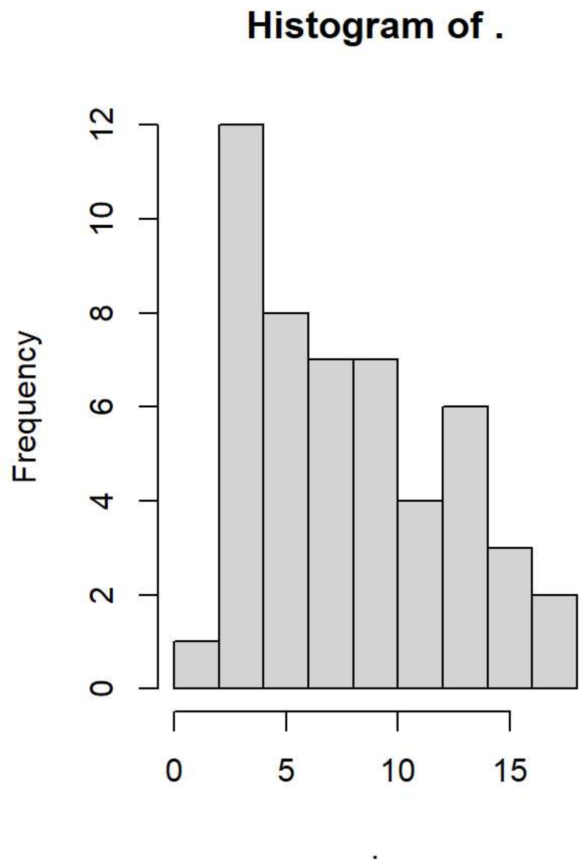
```
USArrests %>%  str()
```

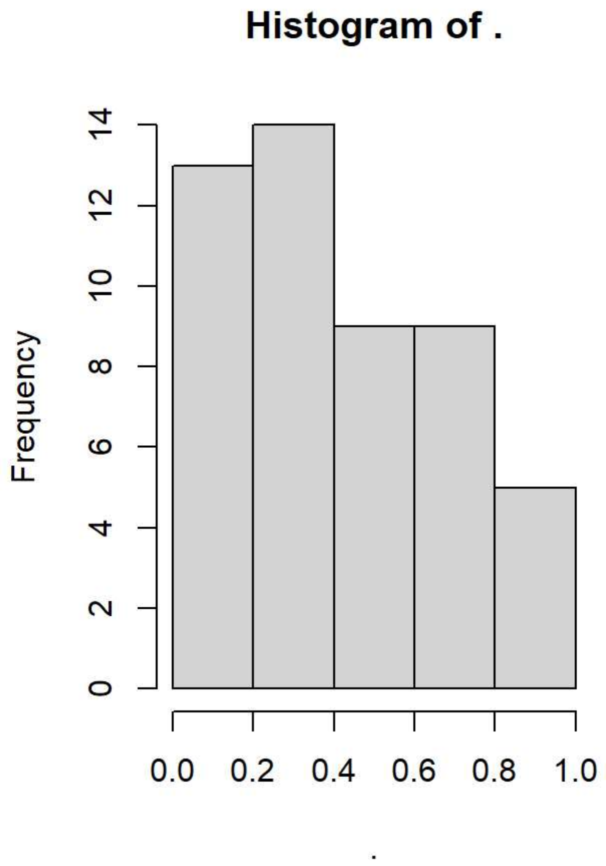
```
## Classes 'data.table' and 'data.frame':   50 obs. of  5 variables:
##  $ V1      : Factor w/ 50 levels "Alabama","Alaska",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
USArrests2 <- USArrests
USArrests2$Murder  <-  scale(USArrests2$Murder, center=TRUE, scale=TRUE)
USArrests2$Assault <-  scale(USArrests2$Assault, center=TRUE, scale=TRUE)

par(mfrow=c(2,2))
USArrests$Murder   %>%  hist()
USArrests$Assault  %>%  hist()
USArrests2$Murder  %>%  hist()
USArrests2$Assault %>%  hist()
```



```
# 8-2. Murder 변수를 min max 정규화 후 히스토그램 그리기
USArrests3 <- USArrests
normalize <- function(x){
  return( (x-min(x))/(max(x)-min(x)))
}
USArrests3$Murder <-  normalize(USArrests3$Murder)
par(mfrow=c(1,2))
USArrests$Murder   %>%  hist()
USArrests3$Murder  %>%  hist()
```


Histogram of .
Frequency
0
2
4
6
8
10
12
0
5
10
15
.


Histogram of .
Frequency
0
2
4
6
8
10
12
14
0.0
0.2
0.4
0.6
0.8
1.0
.