[R] R 데이터 가공을 위한 기본적인 함수 — 나무늘보의 개발 블로그

노트북:	blog
만든 날짜:	2020-10-02 오후 10:09
URL:	https://continuous-development.tistory.com/39?category=793392

R

[R] R 데이터 가공을 위한 기본적인 함수

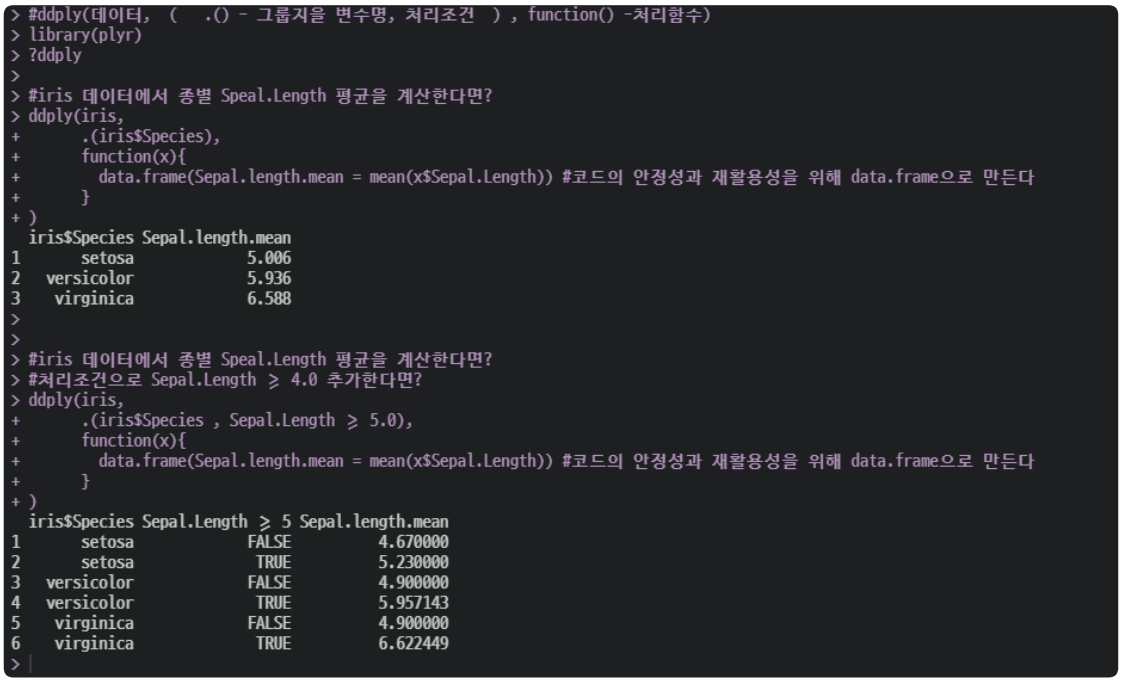
2020. 7. 27. 20:32 수정 삭제 공개

#ddply

- 데이터 프레임(d)을 입력으로 받아 데이터 프레임(d)을 내보내는 함수

-구문

ddply(데이터, .() - 그룹지을 변수명, 처리조건 , function() -처리함수)



```
> #ddply(데이터,  (   .() - 그룹지을 변수명, 처리조건  ) , function() -처리함수)
> library(plyr)
> ?ddply
>
> #iris 데이터에서 종별 Speal.Length 평균을 계산한다면?
> ddply(iris,
+       .(iris$Species),
+       function(x){
+         data.frame(Sepal.length.mean = mean(x$Sepal.Length)) #코드의 안정성과 재활용성을 위해 data.frame으로 만든다
+       }
+ )
  iris$Species Sepal.length.mean
1       setosa            5.006
2   versicolor            5.936
3    virginica            6.588
>
>
> #iris 데이터에서 종별 Speal.Length 평균을 계산한다면?
> #처리조건으로 Sepal.Length ≥ 4.0 추가한다면?
> ddply(iris,
+       .(iris$Species , Sepal.Length ≥ 5.0),
+       function(x){
+         data.frame(Sepal.length.mean = mean(x$Sepal.Length)) #코드의 안정성과 재활용성을 위해 data.frame으로 만든다
+       }
+ )
  iris$Species Sepal.Length ≥ 5 Sepal.length.mean
1       setosa            FALSE          4.670000
2       setosa             TRUE          5.230000
3   versicolor            FALSE          4.900000
4   versicolor             TRUE          5.957143
5    virginica            FALSE          4.900000
6    virginica             TRUE          6.622449
>
```

#rehape 패키지

#melt

- melt 는 식별자id, 측정 변수variable, 측정치value 형태로 데이터를 재구성하는 함수이다
 즉 가로로 된 데이터를 세로로 만든다.

```
-구문

melt(데이터를 구분하는 식별자, 측정대상 변수 , 측정치)
```

```
> # reshape 패키지
> # 변환
> # melt 가로로 된걸 세로로 길게 만든다.
> # cast(dcast, acast) - 동일한 결과를 리턴하는데 array 또는 data.frame으로 만드는 함수
>
>
> ?melt
> # melt(데이터를 구분하는 식별자, 측정대상 변수 , 측정치)
> data(french_fries)
> head(french_fries)
   time treatment subject rep potato buttery grassy rancid painty
61    1         1       3   1    2.9     0.0    0.0    0.0    5.5
25    1         1       3   2   14.0     0.0    0.0    1.1    0.0
62    1         1      10   1   11.0     6.4    0.0    0.0    0.0
26    1         1      10   2    9.9     5.9    2.9    2.2    0.0
63    1         1      15   1    1.2     0.1    0.0    1.1    5.1
27    1         1      15   2    8.8     3.0    3.6    1.5    2.3
> str(french_fries)
'data.frame':   696 obs. of  9 variables:
 $ time     : Factor w/ 10 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ treatment: Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
 $ subject  : Factor w/ 12 levels "3","10","15",..: 1 1 2 2 3 3 4 4 5 5 ...
 $ rep      : num  1 2 1 2 1 2 1 2 1 2 ...
 $ potato   : num  2.9 14 11 9.9 1.2 8.8 9 8.2 7 13 ...
 $ buttery  : num  0 0 6.4 5.9 0.1 3 2.6 4.4 3.2 0 ...
 $ grassy   : num  0 0 0 2.9 0 3.6 0.4 0.3 0 3.1 ...
 $ rancid   : num  0 1.1 0 2.2 1.1 1.5 0.1 1.4 4.9 4.3 ...
 $ painty   : num  5.5 0 0 0 5.1 2.3 0.2 4 3.2 10.3 ...
>
>
> head(french_fries)
   time treatment subject rep potato buttery grassy rancid painty
61    1         1       3   1    2.9     0.0    0.0    0.0    5.5
25    1         1       3   2   14.0     0.0    0.0    1.1    0.0
62    1         1      10   1   11.0     6.4    0.0    0.0    0.0
26    1         1      10   2    9.9     5.9    2.9    2.2    0.0
63    1         1      15   1    1.2     0.1    0.0    1.1    5.1
27    1         1      15   2    8.8     3.0    3.6    1.5    2.3
> fries_melt <- melt(id=1:4, french_fries)
> head(fries_melt)
  time treatment subject rep variable value
1    1         1       3   1   potato   2.9
2    1         1       3   2   potato  14.0
3    1         1      10   1   potato  11.0
4    1         1      10   2   potato   9.9
5    1         1      15   1   potato   1.2
6    1         1      15   2   potato   8.8
```

#cast

- 동일한 결과를 리턴하는데 data.frame으로 만드는 함수
- dcast( )는 결과로 데이터 프레임을 반환하며, acast( )는 벡터, 행렬, 배열을 반환

-구문

dcast(데이터 , 컬럼+컬럼+컬럼+~... <-나머지 컬럼포함)

```
> #cast() - 동일한 결과를 리턴하는데 data.frame으로 만드는 함수
> fries_d ← dcast(fries_melt, time + treatment + subject + rep ~ ...)
> head(fries_d)
  time treatment subject rep potato buttery grassy rancid painty
1    1         1       3   1    2.9     0.0    0.0    0.0    5.5
2    1         1       3   2   14.0     0.0    0.0    1.1    0.0
3    1         1      10   1   11.0     6.4    0.0    0.0    0.0
4    1         1      10   2    9.9     5.9    2.9    2.2    0.0
5    1         1      15   1    1.2     0.1    0.0    1.1    5.1
6    1         1      15   2    8.8     3.0    3.6    1.5    2.3
>
```

#data.table 패키지 - 데이터 테이블을 원하는 식으로 출력하는 함수

#data.table(행의 정보 , 가져올 속성 값) 패키지

```
> iris_table <-data.table(iris)
> iris_table
     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
  1:          5.1         3.5          1.4         0.2    setosa
  2:          4.9         3.0          1.4         0.2    setosa
  3:          4.7         3.2          1.3         0.2    setosa
  4:          4.6         3.1          1.5         0.2    setosa
  5:          5.0         3.6          1.4         0.2    setosa
 ---
146:          6.7         3.0          5.2         2.3 virginica
147:          6.3         2.5          5.0         1.9 virginica
148:          6.5         3.0          5.2         2.0 virginica
149:          6.2         3.4          5.4         2.3 virginica
150:          5.9         3.0          5.1         1.8 virginica
> iris_table[1,]
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1:          5.1         3.5          1.4         0.2  setosa
> # 임의의 2개의 피쳐만 출력한다면?
> iris_table[1,c(2,3)]
   Sepal.Width Petal.Length
1:         3.5          1.4
> iris_table[1,c(Sepal.Length,Sepal.Width)]
[1] 5.1 3.5
> iris_table[1,list(Sepal.Length,Sepal.Width)]
   Sepal.Length Sepal.Width
1:          5.1         3.5
> #iris 데이터에서 Sepal.Length 의 평균값을 종별로 구한다면?
> iris_table[ , mean(Sepal.Length),Species]
      Species    V1
1:     setosa 5.006
2: versicolor 5.936
3:  virginica 6.588
>
```

#readxl - 외부 파일 읽어 들이는 패키지

```r
# read.table
# option
# txt
# -header - 헤더를 가져온다.
# -skip - 내가 원하는 만큼 스킵한다.
# -nrow - 몇개의 행만카운트하겠다.
# -sep - 지정된 데이터 구분자를 사용하겠다. (, tab 같은거)
# -col.names = c(컬럼이름 명시)
# read.table - txt 파일 가져오기
# read_excel - excel 파일 가져오기

txt_data_sample <- read.table(file.choose(), header =  T) #txt 파일의 헤더를 가져와서 넣어준다.
txt_data_sample

txt_data_sample02 <- read.table(file.choose(), header =  T, sep =  ",") #,를 구분자로 사용한다.
txt_data_sample02

col.names <- c("ID","SEX","AGE","AREA") ## 컬럼명을 지정해준다.
txt_data_sample03 <- read.table(file.choose(), header =  T, sep =  ",", col.names = col.names) #,를 구분자로 사용한다.
txt_data_sample03


# service_data_excel_sample.xlsc 읽어보자 (read_excel로 엑셀 데이터 가져오기)
service_data_excel_sample <- read_excel(file.choose())
service_data_excel_sample
str(service_data_excel_sample)

```

같은 출력 다른 구문

```r
# 성별에 따른 17_AMT 평균이용 금액을 확인하고 싶다면
library(dplyr)

#1-1
ddply(service_data_excel_sample,
      .(service_data_excel_sample$SEX),
      function(x){
        data.frame(AMT17.mean = mean(x$AMT17)) #코드의 안정성과 재활용성을 위해 data.frame으로 만든다
      }
)
#1-2
service_data_excel_sample %>%
  group_by((SEX)) %>%
    summarise("17_AMT"=mean(AMT17))

#1-3
sapply(split(service_data_excel_sample$AMT17,service_data_excel_sample$SEX),
       mean,
       na.rm=TRUE)

#2-1
#지역에 따른 y17_cnt 이용건수의 합을 확인하고 싶다면?
ddply(service_data_excel_sample,
      .(service_data_excel_sample$AREA),
      function(x){
        data.frame(y17_cnt.sum = sum(x$Y17_CNT)) #코드의 안정성과 재활용성을 위해 data.frame으로 만든다
      }
)
#2-2
service_data_excel_sample %>%
  group_by((AREA)) %>%
  summarise(y17_cnt_sum = sum(Y17_CNT))

#2-3
sapply(split(service_data_excel_sample$Y17_CNT,service_data_excel_sample$AREA),
       sum,
       na.rm=TRUE)
```

#bind_rows

- 셀을 기준으로 결합한다. 세로 결합

-구문

#bind_rows(value1,value2)

```
> male_hist
# A tibble: 4 x 8
     ID SEX     AGE AREA   AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>  <dbl>   <dbl>  <dbl>   <dbl>
1     2 M        40 경기  450000      25 700000      30
2     4 M        50 서울  400000       8 125000       3
3     5 M        27 서울  845000      30 760000      28
4     9 M        20 인천  930000       4 250000       2
> female_hist
# A tibble: 6 x 8
     ID SEX     AGE AREA    AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>  <dbl>   <dbl>
1     1 F        50 서울  1300000      50 100000      40
2     3 F        28 제주   275000      10  50000       5
3     6 F        23 서울    42900       1 300000       6
4     7 F        56 경기   150000       2 130000       2
5     8 F        47 서울   570000      10 400000       7
6    10 F        38 경기   520000      17 550000      16
> # 세로결합
> # 변수명 기준으로 결합
> # bind_rows()
>
> m_f_bind_join ← bind_rows(male_hist,female_hist)
> m_f_bind_join
# A tibble: 10 x 8
     ID SEX     AGE AREA    AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>  <dbl>   <dbl>
 1     2 M        40 경기   450000      25 700000      30
 2     4 M        50 서울   400000       8 125000       3
 3     5 M        27 서울   845000      30 760000      28
 4     9 M        20 인천   930000       4 250000       2
 5     1 F        50 서울  1300000      50 100000      40
 6     3 F        28 제주   275000      10  50000       5
 7     6 F        23 서울    42900       1 300000       6
 8     7 F        56 경기   150000       2 130000       2
 9     8 F        47 서울   570000      10 400000       7
10    10 F        38 경기   520000      17 550000      16
```

join 종류

데이터를 가로로 결합한다.

left_join : 지정한 변수와 데이터세트1을 기준으로 데이터 세트 2에 있는 나머지 변수 결합 (차집합)

inner_join: 데이터 세트 1과 데이터 세트 2에서 기준으로 지정한 변수 값이 동일할 때만 결합된다. (교집합)

full_join : 전체를 결합 (합집합)

```
> # 가로결합
> # left_join : 지정한 변수와 데이터세트1을 기준으로 데이터세트2에 있는 나머지 변수결합
> # inner_join: 데이터 세트 1과 데이터 세트 2에서 기준으로 지정한 변수값이 동일 할 때만 결합된다.
> # full_join : 전체를 결합
>
> #service_data_jeju_y17_history.xlsx
> #service_data_jeju_y16_history.xlsx
> jeju_y17
# A tibble: 8 x 6
     ID SEX     AGE AREA    AMT17 Y17_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>
1     1 F        50 서울  1300000      50
2     2 M        40 경기   450000      25
3     4 M        50 서울   400000       8
4     5 M        27 서울   845000      30
5     7 F        56 경기   150000       2
6     8 F        47 서울   570000      10
7     9 M        20 인천   930000       4
8    10 F        38 경기   520000      17
> jeju_y16
# A tibble: 9 x 3
     ID  AMT16 Y16_CNT
  <dbl>  <dbl>   <dbl>
1     1 100000      40
2     2 700000      30
3     3  50000       5
4     4 125000       3
5     5 760000      28
6     6 300000       6
7     7 130000       2
8     8 400000       7
9    10 550000      16
> # ID 를 기준으로 jeju_y17_history 데이터 세트를 기준으로 결합
>
```

```
> # ID 를 기준으로 jeju_y17_history 데이터 세트를 기준으로 결합
>
> #차집합
> bind_left ← left_join(jeju_y17,jeju_y16,by="ID")
> bind_left
# A tibble: 8 x 8
     ID SEX     AGE AREA    AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>  <dbl>   <dbl>
1     1 F        50 서울  1300000      50 100000      40
2     2 M        40 경기   450000      25 700000      30
3     4 M        50 서울   400000       8 125000       3
4     5 M        27 서울   845000      30 760000      28
5     7 F        56 경기   150000       2 130000       2
6     8 F        47 서울   570000      10 400000       7
7     9 M        20 인천   930000       4     NA      NA
8    10 F        38 경기   520000      17 550000      16
>
> #교집합
> bind_inner ← inner_join(jeju_y17,jeju_y16,by="ID")
> bind_inner
# A tibble: 7 x 8
     ID SEX     AGE AREA    AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>  <dbl>   <dbl>
1     1 F        50 서울  1300000      50 100000      40
2     2 M        40 경기   450000      25 700000      30
3     4 M        50 서울   400000       8 125000       3
4     5 M        27 서울   845000      30 760000      28
5     7 F        56 경기   150000       2 130000       2
6     8 F        47 서울   570000      10 400000       7
7    10 F        38 경기   520000      17 550000      16
>
> #합집합
> bind_full ← full_join(jeju_y17,jeju_y16,by="ID")
> bind_full
# A tibble: 10 x 8
     ID SEX     AGE AREA    AMT17 Y17_CNT  AMT16 Y16_CNT
  <dbl> <chr> <dbl> <chr>   <dbl>   <dbl>  <dbl>   <dbl>
 1     1 F        50 서울  1300000      50 100000      40
 2     2 M        40 경기   450000      25 700000      30
 3     4 M        50 서울   400000       8 125000       3
 4     5 M        27 서울   845000      30 760000      28
 5     7 F        56 경기   150000       2 130000       2
 6     8 F        47 서울   570000      10 400000       7
 7     9 M        20 인천   930000       4     NA      NA
 8    10 F        38 경기   520000      17 550000      16
 9     3 NA       NA NA         NA      NA  50000       5
10     6 NA       NA NA         NA      NA 300000       6
>
```

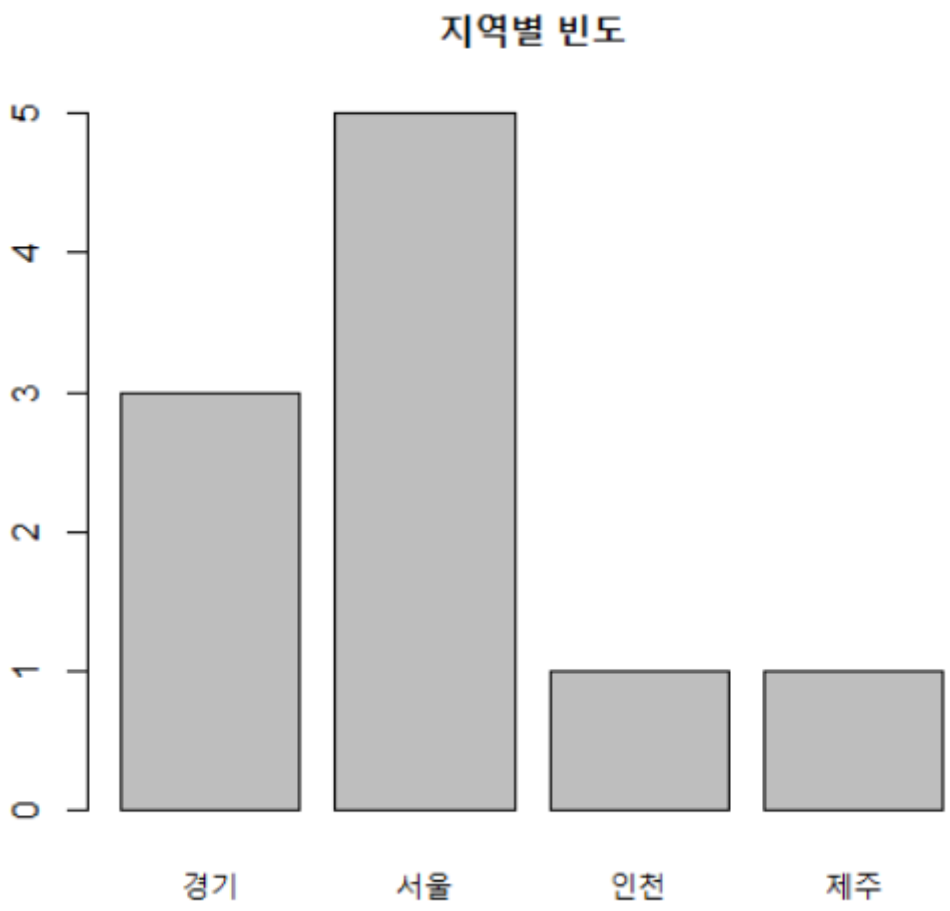
#descr::freq()

- 빈도수를 체크하는 함수

-구문

freq(데이터,plot =T (그래프를 보이게끔한다) , main ="제목")

```
> #descr:::freq() - 빈도수를 체크한다.
> freqArea ← freq(sample_excel$AREA, plot = T)
> freqArea
sample_excel$AREA
      Frequency Percent
경기          3      30
서울          5      50
인천          1      10
제주          1      10
Total        10     100
>
> freqArea ← freq(sample_excel$AREA, plot = T,main='지역별 빈도')
> freqArea
sample_excel$AREA
      Frequency Percent
경기          3      30
서울          5      50
인천          1      10
제주          1      10
Total        10     100
>
> #성별에 따른 빈도분석을 하세요
> freqSex ← freq(sample_excel$SEX, plot = T,main='지역별 빈도')
> freqSex
sample_excel$SEX
      Frequency Percent
F             6      60
M             4      40
Total        10     100
>
```



'**R**' 카테고리의 다른 글□

[R] R 에서 사용되는 기본적인 시각화 그래프-2□

[R] R 에서 사용되는 기본적인 시각화 그래프□

[R] R 데이터 가공을 위한 기본적인 함수□

[R] R 사용자 정의 함수(FUNCTION)와 데이터 전처리를 위한 기본적인 함수□

[R] R로 만드는 제어문 (if, else if, for)과 예제□

[R] R에서 사용되는 Data.frame 과 Factor 에 사용되는 다양한 함수□

bind_rows 함수 ddply 함수 freq함수 melt 함수 R join 종류 readxl 함수

꾸까꾸

혼자 끄적끄적하는 블로그 입니다.