

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

노트북: [TIL-MY]

만든 날짜: 2020-07-25 오후 5:59

URL: <https://continuous-development.tistory.com/38>

# 나무늘보의 개발 블로그

홈

태그

```
> #함수 정의
> newSumFunc <- function(x,y){
+   result <- x+y
+   return (result)
+ }
카테고리 없음
[R] R 사용자 정의 함수(FUNCTION)와
데이터 전처리를 위한 기본적인 함수
> resultSum <- newSumFunc(5,4)
> resultSum
[1] 9
> |
```

## #함수

### - FUNCTION?

분류 전체보기 

Python

Database

ASP.NET

Algorithm

Deep learning

**function**이란, 영어 뜻 그대로 **사용자 정의 함수**를 정의하는 함수를 말한다. 사용자가 원하는 형식의 함수를 만들어 반복적으로 사용 할 수 있다.

기본 형태는 다음과 같다.

```
Func(함수 이름) <- function(매개변수){  
}
```

## #함수정의

newSumFunc 라는 함수를 사용자가 정의해 준다. 매개변수를 x와 y를 받고 이 매개변수를 더해서 result에 저장을 한 후 반환해주는 함수 형태이다.

```
> #함수 정의  
> newSumFunc <- function(x,y){  
+   result <- x+y  
+   return (result)  
+ }  
>  
> resultSum <- newSumFunc(5,4)  
> resultSum  
[1] 9  
> |
```

## #가변함수

AWS

ETC..

R 

공지사항

글 보실 때 주의사항

: 최근글 : 인기글

[R] R  
사...



2020.07.25

[R] R  
로 ...



2020.07.24

[R] R  
에...



2020.07.24

[R] R  
에 ...



2020.07.23

[R] R  
에 ...



2020.07.23

최근댓글

태그

R 배열, 행렬,

R IF, substr,

Oracle SQL, SQL,

아래와 같은경우 function(...) 으로 할경우 변수를 원하는 만큼 넣을 수 있다.

```
157
158 #가변함수 - 매개변수의 개수가 가변적인 함수
159 varFunc <- function(...){
160   args <- list(...) #몇개가 들어올지 몰라서 리스트로 받는다.
161   result <- 0
162   for(idx in args){
163     result <- result + idx
164   }
165   return (result)
166 }
167
168 varFunc(1)
169 varFunc(1,2)
170 varFunc(1,2,3,4)
171 |
172
```

171:1 (Top Level) ↕

Console Terminal x Jobs x

~/ ➡

```
> varFunc(1)
[1] 1
> varFunc(1,2)
[1] 3
> varFunc(1,2,3,4)
[1] 10
> |
```

**#파생변수 - 기존의 변수들에서 새로운 변수를 추가해 만들어준다.**

만들고 싶은 컬럼명에 값을 넣어준다.

stock\$diff <- 라는 컬럼명을 만들고 값을 넣어준다.

사용법, do.call,  
sample 함수,  
array 함수,  
인스턴스,  
R 정규표현식 사용법,  
cbind,  
rownames, 설정,  
rbind함수,  
arrange 함수,  
R제어문, Oracle,  
cbind함수,  
matrix함수,  
날짜함수,  
R 정규표현식,  
DDL, R FOR,  
AWS,  
테이블 생성,  
unlist함수,  
colnames, rbind

전체 방문자

**114**

Today : 10

Yesterday : 3

```

> stock <- read.csv(file.choose())
> str(stock)
'data.frame': 247 obs. of 6 variables:
 $ Date : chr "30-Oct-15" "29-Oct-15" "28-Oct-15" "27-Oct-15" ...
 $ Open : int 1345000 1330000 1294000 1282000 1298000 1300000 1280000 1265000 1260000 1257000 ...
 $ High : int 1390000 1392000 1308000 1299000 1298000 1300000 1295000 1282000 1273000 1265000 ...
 $ Low : int 1341000 1324000 1291000 1281000 1272000 1278000 1269000 1259000 1256000 1249000 ...
 $ Close: int 1372000 1325000 1308000 1298000 1292000 1289000 1280000 1270000 1266000 1256000 ...
 $ Volume: int 498776 622336 257374 131144 151996 252105 229326 138655 137788 116290 ...
> head(stock)
      Date      Open      High      Low      Close Volume
1 30-Oct-15 1345000 1390000 1341000 1372000 498776
2 29-Oct-15 1330000 1392000 1324000 1325000 622336
3 28-Oct-15 1294000 1308000 1291000 1308000 257374
4 27-Oct-15 1282000 1299000 1281000 1298000 131144
5 26-Oct-15 1298000 1298000 1272000 1292000 151996
6 23-Oct-15 1300000 1300000 1278000 1289000 252105
> stock$diff <- stock$High - stock$Low
> head(stock)
      Date      Open      High      Low      Close Volume diff
1 30-Oct-15 1345000 1390000 1341000 1372000 498776 49000
2 29-Oct-15 1330000 1392000 1324000 1325000 622336 68000
3 28-Oct-15 1294000 1308000 1291000 1308000 257374 17000
4 27-Oct-15 1282000 1299000 1281000 1298000 131144 18000
5 26-Oct-15 1298000 1298000 1272000 1292000 151996 26000
6 23-Oct-15 1300000 1300000 1278000 1289000 252105 22000
>

```

## 예제

```

> row <- nrow(stock)
> rows <- 1:row
> diff_result = ""
>
> for(idx in rows){
+   if(stock$diff[idx] > mean(stock$diff)){
+     diff_result[idx] <- "mean over"
+   }else{
+     diff_result[idx] <- "mean under"
+   }
+ }
> stock$diff_result <- diff_result
> head(stock)
      Date      Open      High      Low      Close Volume diff diff_result
1 30-Oct-15 1345000 1390000 1341000 1372000 498776 49000 mean over
2 29-Oct-15 1330000 1392000 1324000 1325000 622336 68000 mean over
3 28-Oct-15 1294000 1308000 1291000 1308000 257374 17000 mean under
4 27-Oct-15 1282000 1299000 1281000 1298000 131144 18000 mean under
5 26-Oct-15 1298000 1298000 1272000 1292000 151996 26000 mean under
6 23-Oct-15 1300000 1300000 1278000 1289000 252105 22000 mean under
>

```

**#while(논리값) {} - 논리값이 True가 될때 까지 계속 반복한다.**

```
while(){
```

```
}
```

```
while(조건식){  
원하는 로직  
}
```

```
> idx ← 1  
> while(idx ≤ 10){  
+   print(idx)  
+   idx=idx+1  
+ }  
[1] 1  
[1] 2  
[1] 3  
[1] 4  
[1] 5  
[1] 6  
[1] 7  
[1] 8  
[1] 9  
[1] 10  
> |
```

```
[1] 10
> #1~100의 사이에 5의 배수만 출력하고 싶다면?
> idx <-1
> while( idx <= 100){
+   if(idx %% 5 ==0){
+     print(idx)
+   }
+   idx <- idx + 1
+ }
[1] 5
[1] 10
[1] 15
[1] 20
[1] 25
[1] 30
[1] 35
[1] 40
[1] 45
[1] 50
[1] 55
[1] 60
[1] 65
[1] 70
[1] 75
[1] 80
[1] 85
[1] 90
[1] 95
[1] 100
```

**#next** 해당 명령을 만나면 이 위치에  
서 더 진행되지 않고 반복문으로 보낸  
다.

**#break** 문을 만나면 해당 루프를 빠  
져나온다.

```

> ##next(continus) , break
> #홀수만 출력
> idx <- 0
> while(idx <= 10){
+   idx <- idx + 1
+   if(idx%%2!=0){
+     print(idx)
+   }
+ }
[1] 1
[1] 3
[1] 5
[1] 7
[1] 9
[1] 11
> #짝수만 출력
> idx <- 0
> while(idx <= 10){
+   idx <- idx + 1
+   if(idx%%2!=0){
+     next
+   }
+   print(idx)
+ }
[1] 2
[1] 4
[1] 6
[1] 8
[1] 10
> #break
> idx <- 0
> while(idx <= 10){
+   idx <- idx + 1
+   if(idx%%2!=0){
+     break
+   }
+   print(idx)
+ }
> |

```

## #NA 처리(na.rm=T , na.omit , na,pass, na.fail)

```

> # NA 처리
> NA & T
[1] NA
>
> #문제를 해결하기 위해서
> sum(c(1,2,3,NA))
[1] NA
> sum(c(1,2,3,NA), na.rm=T)
[1] 6
> mean(c(1,2,3,NA), na.rm=T)
[1] 2
>
> # package::caret
> # na.omit(), na.pass(), na.fail()
>
> na.omit(c(1,2,3,NA)) #결측치를 제거하고 반환한다.
[1] 1 2 3
attr(,"na.action")
[1] 4
attr(,"class")
[1] "omit"
> na.pass(c(1,2,3,NA)) #모든걸 출력한다.
[1] 1 2 3 NA
> na.fail(c(1,2,3,NA)) #결측치가 있으면 NA를 반환한다.
Error in na.fail.default(c(1, 2, 3, NA)) : 객체안에 결측값들이 있습니다
> |

```

---

**#merge(value1,value2) -데이터의  
순서가 다르더라도 키를 기준으로  
bind 해준다.**

```
> x <- data.frame(name = c("임정섭", "임은결", "임재원"),  
+               math = c(100,60,95))  
> y <- data.frame(name = c("임정섭", "임은결", "임재원"),  
+               math = c(100,60,95))  
> ?cbind  
> cbind(x,y)  
  name math name math  
1 임정섭 100 임정섭 100  
2 임은결  60 임은결  60  
3 임재원  95 임재원  95  
> ?merge  
> merge(x,y)  
  name math  
1 임은결  60  
2 임재원  95  
3 임정섭 100
```

**##doBy pacakage**

**#summary(value) - 요약해서 나타내  
는 함수**



```

> # doBy package - 특정 값에 따라 데이터를 처리하는 유용한 함수들
> # summaryBy(), orderBy(), splitBy()
>
> summary(iris)
  Sepal.Length   Sepal.Width   Petal.Length   Petal.Width   Species
Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa   :50
1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
>
> # 자료의 분포 quantile()
> quantile(iris$Sepal.Length)
 0%  25%  50%  75% 100%
4.3  5.1  5.8  6.4  7.9
> quantile(iris$Sepal.Length,seq(0,1,by=0.1))
 0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
4.30 4.80 5.00 5.27 5.60 5.80 6.10 6.30 6.52 6.90 7.90

```

**#summaryBy() - 평균값을 볼수 있다.**

```

> # summaryBy() - 평균값을 볼수 있다.
> #원하는 컬럼의 값을 특정 조건에 따라 요약하는 목적
> ?summaryBy
>
> summaryBy(Sepal.Length ~ Species, iris) # Species 기준으로 Sepal.Length의 평균을 구해준다
  Species Sepal.Length.mean
1   setosa           5.006
2 versicolor           5.936
3  virginica           6.588

```

**#orderby() - 원하는 기준에 따라 정렬할 수 있다.**

```

> # orderBy() - 원하는 기준에 따라 정렬할 수 있다.
> # 정렬
>
> orderBy(~Species , iris)
  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
6           5.4         3.9         1.7         0.4   setosa
7           4.6         3.4         1.4         0.3   setosa
8           5.0         3.4         1.5         0.2   setosa
9           4.4         2.9         1.4         0.2   setosa
10          4.9         3.1         1.5         0.1   setosa
11          5.4         2.7         1.5         0.2   setosa

```

## #order() - 주어진 값을 정렬했을때의 색인 순서대로 반환

```
> # order():주어진 값을 정렬했을때의 색인 순서대로 반환
> order(iris$Sepal.Width)
[1] 61 63 69 120 42 54 88 94 58 81 82 78 73 98 99 107 109 114 147 80 91 83 119 135 60 60 83 84 95 102 112 124 143 55 56
[36] 72 74 77 100 115 122 123 127 129 131 133 134 9 59 64 65 75 79 97 98 104 108 2 13 14 26 39 46 62 67 76 78 85 89 92
[71] 96 103 105 106 113 117 120 130 136 139 146 148 150 4 10 31 35 53 66 87 138 140 141 142 3 30 36 43 48 51 52 71 111 116 121
[106] 126 144 24 50 57 101 123 145 7 8 12 21 25 27 29 32 40 86 137 149 1 18 28 37 41 44 5 23 38 110 11 22 49 19 20
[141] 45 47 118 132 6 17 15 33 34 16
> iris[order(iris$Sepal.Width),]
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
61          5.0         2.0         3.5          1.0 versicolor
63          6.0         2.2         4.0          1.0 versicolor
69          6.2         2.2         4.5          1.5 versicolor
120         6.0         2.2         5.0          1.5 virginica
```

## #sample(value,개수,[replace =T]) - 샘플에 따라서 추출하는 함수 (replace 옵션에 따라 복원 비복원이 나뉜다.)

```
> #sample()
> #복원추출 - 값을 추출 했을때 다시 복원하고 추출하는 방식
> #비복원추출 - 값을 추출했을때 복원하지않고 추출 된 상태에서 추출하는 방식
> ?sample
>
> sample(1:10, 5) #비복원 추출
[1] 3 5 4 9 6
> sample(1:10, 5,replace=T) #복원 추출
[1] 7 6 9 9 3
```

## #sampleBy(기준, 추출할 샘플의 비율, 데이터) - 데이터를 그룹으로 묶은 후 각 그룹에서 샘플을 추출하는 함수

```

261
262 #sampleBy(기준, 추출할 샘플의 비율, 데이터) - 데이터를 그룹으로 묶은 후 각 그룹에서 샘플을 추출하는 함수
263 ?sampleBy()
264 train <- sampleBy( ~Species ,frac =0.2 , data = iris)
265 train
266
267 test <- sampleBy( ~Species ,frac =0.8 , data = iris)
268 test
269

```

**#split(data,분류기준) - 데이터를 특정 조건에 맞춰서 나눈다.**

```

> # split(data , 분류기준 )- 데이터를 특정 조건에 맞춰서 나눈다.
> # 반환값 list
> # lapply, sapply
> ?split
>
> #iris를 Species 기준으로 나뉘라
> split(iris, iris$Species)
$setosa
  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
6           5.4         3.9          1.7          0.4  setosa
7           4.6         3.4          1.4          0.3  setosa
8           5.0         3.4          1.5          0.2  setosa
9           4.4         2.9          1.4          0.2  setosa

```

**#filter(data , 조건) : 조건에 따라 행을 추출**

```

> # filter(data, 조건) : 조건에 따라 행을 추출
> ?dplyr::filter()
> #1월1일 데이터를 추출
> filter(tbl_df(hflights), Month == 1 & DayofMonth == 1)
# A tibble: 552 x 21
  Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum TailNum ActualElapsedTime AirTime
<int> <int> <int> <int> <int> <int> <chr> <int> <chr> <int> <int>
1 2011 1 1 6 1400 1500 AA 428 N576AA 60 40
2 2011 1 1 6 728 840 AA 460 N520AA 72 41
3 2011 1 1 6 1631 1736 AA 1121 N4MVAA 65 37
4 2011 1 1 6 1756 2112 AA 1294 N306AA 136 113
5 2011 1 1 6 1012 1347 AA 1700 N30AAA 155 117
6 2011 1 1 6 1211 1325 AA 1020 N593AA 74 39
7 2011 1 1 6 357 906 AA 1994 N388AA 129 113
8 2011 1 1 6 1024 2106 AS 731 N614AS 282 255
9 2011 1 1 6 654 1124 B6 620 N324JB 210 181
10 2011 1 1 6 1639 2110 B6 622 N324JB 211 188
# ... with 542 more rows, and 10 more variables: ArrDelay <int>, DepDelay <int>, Origin <chr>, Dest <chr>,
# Distance <int>, TaxiIn <int>, TaxiOut <int>, Cancelled <int>, CancellationCode <chr>, Diverted <int>

```

## #arrange(data,정렬할 변수명) -정렬 함수 (기본 오름차순)

```

> #arrange() -정렬함수
> ?dplyr::arrange
> #데이터를 ArrDelay,Month,Year 순으로 정렬
> arrange(hflights, ArrDelay,Month,Year)
  Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum TailNum ActualElapsedTime AirTime
1 2011 7 3 7 1914 2039 XE 2804 N12157 85 66
2 2011 12 25 7 741 926 00 4591 N814SK 165 147
3 2011 8 21 7 935 1039 00 2001 N767SK 184 171
4 2011 8 31 3 934 1039 00 2040 N783SK 185 172
5 2011 8 26 5 2107 2205 00 2003 N713SK 178 163
6 2011 12 24 6 2129 2337 CO 1552 N37437 248 234
7 2011 8 28 7 2059 2206 00 2003 N783SK 187 171
8 2011 8 29 1 935 1041 00 2040 N767SK 186 169
9 2011 8 18 4 939 1043 00 2001 N783SK 184 172
10 2011 12 24 6 2117 2258 CO 1712 N74856 221 200
11 2011 12 24 6 1431 1613 CO 1737 N73860 222 204
12 2011 8 27 6 938 1040 00 2001 N744SK 182 162
13 2011 1 30 7 620 812 00 4461 N804SK 172 156

```

```

> #내림차순 정렬
> arrange(hflights, desc(Month))
  Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum TailNum ActualElapsedTime AirTime
1 2011 12 15 4 2113 2217 AA 426 N433AA 64 44
2 2011 12 16 5 2004 2128 AA 426 N588AA 84 39
3 2011 12 18 7 2007 2113 AA 426 N43HAA 66 46
4 2011 12 19 1 2108 2223 AA 426 N4YDAA 75 54
5 2011 12 20 2 2008 2107 AA 426 N434AA 59 41
6 2011 12 21 3 2025 2124 AA 426 N589AA 59 43
7 2011 12 22 4 2021 2118 AA 426 N4YCAA 57 39
8 2011 12 23 5 2015 2110 AA 426 N510AA 63 36
9 2011 12 26 1 2013 2118 AA 426 N4YLAA 65 43
10 2011 12 27 2 2007 2123 AA 426 N4YTAA 76 53

```

## #select(data,조건) - 열에 대한 추출

```

351
352 #select() - 열에 대한 추출
353 #mutate() - 열 추가
354 #select() , [3:4] , '-' 지정한 열 제외
355 select(hflights, Year, Month, DayofMonth)
356 select(hflights, Year:DayofMonth)
357 hflights
358 #Year부터 DayOfWeek 를 제외한 열을 추출
359 select(hflights, -c(Year:DayOfWeek))
360 select(hflights, -(Year:DayOfWeek))
361

```


## #mutate() - 열 추가

```

363 # ArrDelay - DepDelay → gain
364 # gain / (AirTime/60) → gain_per_hour
365 |
366 flightDF <- hflights
367 mutate(flightDF,
368        gain = ArrDelay - DepDelay,
369        gain_per_hour = gain / (AirTime/60))
370

```

365:1 (Top Level) : R Script

Console Terminal < Jobs < 

```

~/ #
6353 2011 1 11 2 717 820 AA 460 N574AA 63 44
6354 2011 1 12 3 714 814 AA 460 N580AA 60 42
ArrDelay DepDelay Origin Dest Distance TaxiIn TaxiOut Cancelled CancellationCode Diverted gain gain_per_hour
5424 -10 0 IAH DFW 224 7 13 0 0 0 -10 -15.000000
5425 -9 1 IAH DFW 224 6 9 0 0 -10 -13.333333
5426 -8 0 IAH DFW 224 5 17 0 0 0 0.000000
5427 3 3 IAH DFW 224 9 22 0 0 0 0.000000
5428 -3 5 IAH DFW 224 9 9 0 0 -8 -10.909091
5429 -7 -1 IAH DFW 224 6 13 0 0 -6 -8.000000
5430 -1 -1 IAH DFW 224 12 15 0 0 0 0.000000
5431 -16 -5 IAH DFW 224 7 12 0 0 -11 -16.500000
5432 44 43 IAH DFW 224 8 22 0 0 1 1.463415
5433 43 43 IAH DFW 224 6 19 0 0 0 0.000000

```

## # transform() - 열 추가

```

370
371 #transform() - 열 추가 / 파생변수를 사용하지 못한다.
372
373 transform(flightDF,
374          gain = ArrDelay - DepDelay,
375          gain_per_hour = gain / (AirTime/60))
376
377

```

**# summarise() - 기초 통계량(mean, sd(표준편차) , var, median(중위값))을 구할 수 있다.**

```
> # summarise() - 기초 통계량(mean, sd(표준편차) , var, median(중위값))을 구할 수 있다.
> # 데이터 프레임 형식으로 반환
>
> ?dplyr::summarise
>
> #출발지연시간 평균 및 합계 계산을 한다면??
>
> sum(is.na(flightDF$ArrDelay))
[1] 3622
> mean(is.na(flightDF$ArrDelay))
[1] 0.01592116
>
> summarise(flightDF,
+           sum = sum(is.na(flightDF$ArrDelay),na.rm = T),
+           mean = mean(is.na(flightDF$ArrDelay)),na.rm = T)
   sum    mean na.rm
1 3622 0.01592116 TRUE
>
>
> summarise(flightDF,
+           mean = mean(DepDelay,na.rm=T),
+           sum = sum(DepDelay,na.rm =T))
   mean    sum
1 9.444951 2121251
> |
```

**#예제**

```

397 # hflights 데이터셋에서 비행편수 20편 이상, 평균비행거리 2,000마일 이상인 항공사별 평균 연착시간을 계산한다!
398
399 #1
400 data(hflights)
401 planes <- group_by(hflights, TailNum)
402 head(planes)
403 delay <- dplyr::summarise(planes,
404   count = n(),
405   dist = mean(Distance, na.rm = T),
406   delay = mean(ArrDelay, na.rm = T) )
407
408 delay
409 delay <- filter(delay, count >= 20 , dist >= 2000)
410 delay
411
412 #2
413 filter(dplyr::summarise(group_by(hflights, TailNum),
414   count = n(),
415   dist = mean(Distance, na.rm = T),
416   delay = mean(ArrDelay, na.rm = T) ), count >= 20 , dist >= 2000)
417
418
419
420 library(ggplot2)
4106 (Top Level) :

```

```

console Terminal Jobs
hflights 20 planes 13.7
N77866 85 3579. 13.7

```

```

#2
filter(dplyr::summarise(group_by(hflights, TailNum),
  count = n(),
  dist = mean(Distance, na.rm = T),
  delay = mean(ArrDelay, na.rm = T) ), count >= 20 , dist >= 2000)
summarise(): ungrouping output (override with '.groups' argument)
A tibble: 6 x 4
  TailNum count dist delay
<dbl> <dbl> <dbl> <dbl>
N658956 47 2038. 3.21
N659863 81 3619. 22.2
N76862 43 2033. 15.6
N76864 77 3579. 11.7
N76865 90 3607. 2.89
N77866 85 3579. 13.7

```

#chain()- 다른 구문을 연결해준다 ( %>% )

```

434 # chain() 함수
435 # %>%
436
437 #평균 출발지연 시간이 30분 이상인 데이터 출력
438 chain01 <- group_by(hflights, Year, Month, DayofMonth)
439 chain02 <- select(chain01, Year:DayofMonth, ArrDelay, DepDelay)
440 chain03 <- summarise(chain02,
441   arrival = mean(ArrDelay, na.rm =T),
442   depart = mean(DepDelay, na.rm =T))
443
444 result <- filter(chain03, arrival >= 30 | depart >= 30)
445
446 # 위의 구문을 chain 으로 바꿀시
447 hflights %>%
448   group_by(Year, Month, DayofMonth) %>%
449   (Year:DayofMonth, ArrDelay, DepDelay) %>%
450   summarise(
451     arrival = mean(ArrDelay, na.rm =T),
452     depart = mean(DepDelay, na.rm =T)) %>%
453   filter( arrival >= 30 | depart >= 30)
454
455

```

## # `adply()` - a(배열)을 받아 d(데이터 프레임)으로 반환하는 함수

```
456 # adply() - 이 세가지를 한번에 쓰는 함수
457 # 데이터 분할(split)
458 # apply
459 # combine
460 ?adply
461
462 iris
463
464 # Sepal.Length 가 5.0 이상이고 Species가 setosa인지 여부를 확인한 다음 그 결과를 새로운 컬럼 v1에 기록한다면?
465 Sepal.Length >= 5.0 & Species == setosa
466
467 #row 값을 쓰기 때문에 이식테로하면 컬럼명이 v1으로 온다
468 adply(iris,1, function(row){row$Sepal.Length >= 5.0 & row$Species == "setosa" })
469
470 #이렇게 해야 컬럼명이 divSetosa으로 받아진다.
471 adply(iris,1, function(row){ data.frame( divSetosa = c(row$Sepal.Length >= 5.0 & row$Species == "setosa") )})
472
```

458.8 (Top Level) :

Console Terminal Jobs

```
~/R
> #이렇게 해야 컬럼명이 divSetosa으로 받아진다.
> adply(iris,1, function(row){ data.frame( divSetosa = c(row$Sepal.Length >= 5.0 & row$Species == "setosa") )})
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species divSetosa
1         5.1         3.5         1.4         0.2   setosa      TRUE
2         4.9         3.0         1.4         0.2   setosa     FALSE
3         4.7         3.2         1.3         0.2   setosa     FALSE
4         4.6         3.1         1.5         0.2   setosa     FALSE
5         5.0         3.6         1.4         0.2   setosa      TRUE
6         5.4         3.9         1.7         0.4   setosa      TRUE
7         4.6         3.4         1.4         0.3   setosa     FALSE
8         5.0         3.4         1.5         0.2   setosa      TRUE
9         4.4         2.9         1.4         0.2   setosa     FALSE
10        4.9         3.1         1.5         0.1   setosa     FALSE
11        5.4         3.7         1.5         0.2   setosa      TRUE
12        4.8         3.4         1.6         0.2   setosa     FALSE
13        4.8         3.0         1.4         0.1   setosa     FALSE
```

### 태그

arrange 함수

Chain

filter 함수

function

sample 함수

select 함수

결측치 처리

사용자 정의 함수

### 댓글 0



TEL. 02.1234.5678 / 경기 성남시 분당구 판교역로

© Kakao Corp.

