" />

R 데이터 매니지먼트: tidyverse

김진섭; jinseob2kim

차라투(주)

Executive Summary

tidyverse는 직관적인 코드를 장점으로 원래의 R 문법을 빠르게 대체하고 있다.

- magrittr 패키지의 %>% 연산자로 의식의 흐름대로 코딩한다.
- dplyr 패키지의 select, mutate, filter, group_by, summarize 함수는 %>% 와 찰떡궁합이다.

magrittr: %>%

하나만 기억해야 한다면 %>%

- 단축키 Ctrl + Shift + M (**OS X**: Cmd + Shift + M)
- 의식 흐름대로 코딩 가능.

```
a <- read.csv("https://raw.githubusercontent.com/jinseob2kim/R-skku-biohrs/main/da
library(magrittr)
a %>% head
```

	Sex <chr>	Age <int>	Height <int>	Weight <dbl>
1	M	52	160	63
2	M	67	162	63
3	M	75	163	63
4	F	66	154	61
5	M	52	165	64
6	M	56	166	70

6 rows | 1-5 of 15 columns

a의 head를 보여줘

- head(a) 와 a %>% head 는 동일한 코드.
- 후자가 생각의 흐름을 그대로 반영.

```
head(a)
a %>% head                  ## 동일
a %>% head()                ## 동일
```

%>%: 함수 입력값을 앞으로 빼옴.

첫 입력값은 그냥 빼오면 됨

- f(x, y) = x %>% f(y)

첫 입력값 아니라면 . 으로 흔적 남겨야

```
a %>% head(n = 10)
10 %>% head(a, .)
10 %>% head(a, n = .)
```

실습 1: %>% 써보기

데이터셋 a 에서 **남자만** 뽑고, 1주차 방법과 비교하기.

```
subset(a, Sex == "M")
a %>% subset(Sex == "M")
```

	Sex <chr>	Age <int>	Height <int>	Weight <dbl>
1	M	52	160	63.0
2	M	67	162	63.0
3	M	75	163	63.0
5	M	52	165	64.0
6	M	56	166	70.0
11	M	60	170	75.0
12	M	69	161	67.0
13	M	55	162	67.0
15	M	73	168	66.0
16	M	73	168	64.0

1-10 of 746 rows | 1-5 of 15 columns Previous 1 2 3 4 5 6 75 Next

실습 2: 변수 선택

Sex 변수만 고르기

```
## original
a$Sex                   ## data.frame style
a[, "Sex"]              ## matrix style
a[["Sex"]]              ## list style

## tidyverse style
a %>% .$Sex
a %>% .[, "Sex"]
a %>% .[["Sex"]]
```

여러 함수 같이 쓸 때

a에서 **남자만** 뽑아서 head를 보여줘

```
head(subset(a, Sex == "M"))
a %>% subset(Sex == "M") %>% head
```

	Sex <chr>	Age <int>	Height <int>	Weight <dbl>
1	M	52	160	63
2	M	67	162	63
3	M	75	163	63
5	M	52	165	64
6	M	56	166	70
11	M	60	170	75

6 rows | 1-5 of 15 columns

예: 회귀분석

남자만 뽑아 **회귀분석**을 수행하고 그 **계수와 p-value** 보여주기

```
b <- subset(a, Sex == "M")
model <- glm(DM ~ Age + Weight + BMI, data = b, family = binomial)
summ.model <- summary(model)
summ.model$coefficients
```

	Estimate	Std. Error	z value	Pr(>|z|)
	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-4.366	0.937	-4.662	0.000
Age	0.033	0.008	4.032	0.000
Weight	0.028	0.009	3.190	0.001
BMI	-0.001	0.001	-1.327	0.185

4 rows

중간 결과물인 **b, model, summ.model** 필요

회귀분석 with %>%

```
a %>%
  subset(Sex == "M") %>%
  glm(DM ~ Age + Weight + BMI, data = ., family = binomial) %>%
  summary %>%
  .$coefficients
```

	Estimate	Std. Error	z value	Pr(>|z|)
	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-4.366	0.937	-4.662	0.000
Age	0.033	0.008	4.032	0.000
Weight	0.028	0.009	3.190	0.001
BMI	-0.001	0.001	-1.327	0.185

4 rows

읽기 쉬움, 중간결과물 필요없음

각 줄은 꼭 %>% 로 끝나야 한다

```
a %>% subset(Sex == "M")
  %>% head
```

(X)

```
a %>% subset(Sex == "M") %>%
  head
```

(O)

오늘 강의에서 %>% **만 익숙해지면 성공**

실습 3

50세 이상만 뽑아서, **성별과 흡연력** 에 따른 **모든 변수**들의 **평균, 표준편차**를 구하라. (aggregate)

```
b <- subset(a, Age >= 50)
aggregate(. ~ Sex + Smoking, data = b,
          FUN = function(x){c(mean = mean(x), sd = sd(x))})
```

aggregate 는 범주형 변수 알아서 제외해줌.

%>% : 중간결과물인 **b** 필요없음

```
a %>%
  subset(Age >= 50) %>%
  aggregate(. ~ Sex + Smoking, data = .,
            FUN = function(x){c(mean = mean(x), sd = sd(x))})
```

실습 3: 결과 저장

결과를 **out** 에 저장

```
out <- a %>%
  subset(Age >= 50) %>%
  aggregate(. ~ Sex + Smoking, data = .,
            FUN = function(x){c(mean = mean(x), sd = sd(x))})
```

-> 도 가능하지만 권장안함.

```
a %>%
  subset(Age >= 50) %>%
  aggregate(. ~ Sex + Smoking, data = .,
             FUN = function(x){c(mean = mean(x), sd = sd(x))}) -> out
```

dplyr

데이터 다루는 함수들 모음

- 기본 R 함수보다 깔끔

```
library(dplyr)                                  ## 따로 magrittr 불러올 필요 없음
a %>%
   filter(Age >= 50) %>%
   select(-STRESS_EXIST) %>%          ## 범주형 변수 제외
   group_by(Sex, Smoking) %>%
   summarize_all(list(mean = mean, sd = sd))
```

Sex <chr>	Smoking <int>	Age_mean <dbl>	Height_mean <dbl>	Weight_mean <dbl>
F	0	68.32456	153.4035	58.96096
F	1	64.33333	151.4444	53.11111
M	0	64.62500	166.3380	69.17037
M	1	62.88785	166.4299	68.87336

4 rows | 1-5 of 24 columns

filter: subset 과 비슷

subset 과 거의 동일

```
subset(a, Sex == "M")
a %>% subset(Sex == "M")
filter(a, Sex == "M")
a %>% filter(Sex == "M")
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	52	160	63.0	24.60938
M	67	162	63.0	24.00549
M	75	163	63.0	23.71184
M	52	165	64.0	23.50781
M	56	166	70.0	25.40282
M	60	170	75.0	25.95156
M	69	161	67.0	25.84777
M	55	162	67.0	25.52964
M	73	168	66.0	23.38435
M	73	168	64.0	22.67574

여러 조건일 때 편함

- & 대신 , 가능
- between: 범위 설정

```
a %>% subset(Age >= 50 & Age <= 60)
a %>% filter(Age >= 50, Age <= 60)    # ,
a %>% filter(between(Age, 50, 60))    # between 50 and 60
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	52	160	63.0	24.60938
M	52	165	64.0	23.50781
M	56	166	70.0	25.40282
M	60	170	75.0	25.95156
M	55	162	67.0	25.52964
M	56	167	69.0	24.74094
M	52	174	75.0	24.77210
M	57	168	75.0	26.57313
M	57	168	57.0	20.19558

arrange: 정렬

- order 는 순서만 보여줌. arrage 는 바로 정렬까지.

```
a[order(a$Age), ]
a %>% .[order(.$Age), ]
a %>% arrange(Age)
a %>% arrange_("Age")    ## 문자로 넣을 때
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	21	174	74.0	24.44180
M	35	169	79.0	27.66010
M	36	178	101.0	31.87729
M	36	99	99.0	999.00000
M	37	180	90.0	27.77778
M	38	175	95.0	31.02041
M	38	183	75.0	22.39541
M	38	172	78.0	26.36560
F	38	156	58.0	23.83300
M	40	177	74.0	23.62939

desc 내림차순

```
a[order(a$Age, -a$BMI), ]
a %>% .[order(.$Age, -.$BMI), ]
a %>% arrange(Age, desc(BMI))
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	21	174	74.0	24.44180
M	35	169	79.0	27.66010
M	36	99	99.0	999.00000
M	36	178	101.0	31.87729
M	37	180	90.0	27.77778
M	38	175	95.0	31.02041
M	38	172	78.0	26.36560
F	38	156	58.0	23.83300
M	38	183	75.0	22.39541
M	40	170	78.0	26.98962

1-10 of 1,000 rows | 1-5 of 14 columns Previous **1** 2 3 4 5 6 … 100 Next

select: 변수 선택

```
a[, c("Sex", "Age", "Height")]
a %>% .[, c("Sex", "Age", "Height")]
a %>% select(Sex, Age, Height)
a %>% select_("Sex", "Age", "Height")
```

Sex <chr>	Age <int>	Height <int>
M	52	160
M	67	162
M	75	163
F	66	154
M	52	165
M	56	166
F	67	155
F	65	155
F	68	145
F	45	156

1-10 of 1,000 rows Previous 1 2 3 4 5 6 … 100 Next

여러 표현방법

```
a %>% select(Sex:Height)          ## Sex 부터 Height 사이의 모든 변수
a %>% select("Sex":"Height")
a %>% select(2, 3, 4)
a %>% select(c(2, 3, 4))
a %>% select(2:4)
```

특정 변수 제외

```
a[, -c("Sex", "Age", "Height")]
a %>% .[, -c("Sex", "Age", "Height")]
a %>% select(-Sex, -Age, -Height)
```

Weight <dbl>	BMI <dbl>	DM <int>	HTN <int>	Smoking <int>
63.0	24.60938	0	1	1
63.0	24.00549	1	1	0
63.0	23.71184	1	1	0
61.0	25.72103	0	0	0
64.0	23.50781	0	1	0
70.0	25.40282	1	1	1
56.0	23.30905	0	1	0
57.0	23.72529	1	0	0
55.0	26.15933	0	1	0
56.0	23.01118	0	0	0

1-10 of 1,000 rows | 1-5 of 11 columns Previous **1** 2 3 4 5 6 … 100 Next

여러 표현방법

```
a %>% select(-2, -3, -4)
a %>% select(-(2:4))
a %>% select(-c(2, 3, 4))

a %>% select(-(Sex:Height))
a %>% select(-"Sex", -"Age", -"Height")
a %>% select(-("Sex":"Height"))
```

특정 조건

_date 로 끝나는 변수들만 고르고 싶다면?

```
a[, grep("_date", names(a))]          ## "_date" 포함
a %>% .[, grep("_date", names(.))]
a %>% select(ends_with("date"))       ## "_date" 로 끝남
```

MACCE_date	Death_date
<int>	<int>
1056	1056
270	270
1875	1875
2112	2112
2052	2052
792	792
2171	2171
1210	1210
2437	2437
1078	1078

select 와 함께하는 함수들

start_with("abc"): "abc"로 시작하는 이름

end_with("xyz"): "xyz"로 끝나는 이름

contains("ijk"): "ijk"를 포함하는 이름

one_of(c("a", "b", "c")): 변수명이 a, b, c 중 하나

num_range("x", 1:3): x1, x2, x3

실습 4

남자 만 골라, **Sex:HTN** 사이의 변수들만 뽑고, 나이로 정렬하라.

```
a %>% filter(Sex == "M") %>% select(Sex:HTN) %>% arrange(Age)
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	21	174	74.0	24.44180
M	35	169	79.0	27.66010
M	36	178	101.0	31.87729
M	36	99	99.0	999.00000
M	37	180	90.0	27.77778
M	38	175	95.0	31.02041
M	38	183	75.0	22.39541
M	38	172	78.0	26.36560
M	40	177	74.0	23.62029
M	40	168	66.0	23.38435

1-10 of 746 rows | 1-5 of 7 columns Previous **1** 2 3 4 5 6 … 75 Next

실습 4: 기본 R 스타일

```
a %>% subset(Sex == "M") %>% .[, c("Sex", "Age", "Height", "Weight", "BMI", "DM",
a %>% subset(Sex == "M") %>% .[, 2:8] %>% .[order(.$Age), ]
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	21	174	74.0	24.44180
M	35	169	79.0	27.66010
M	36	178	101.0	31.87729
M	36	99	99.0	999.00000
M	37	180	90.0	27.77778
M	38	175	95.0	31.02041
M	38	183	75.0	22.39541
M	38	172	78.0	26.36560
M	40	177	74.0	23.62029
M	40	168	66.0	23.38435

1-10 of 746 rows | 1-5 of 7 columns Previous **1** 2 3 4 5 6 … 75 Next

mutate: 변수 생성

Old, Overweight 변수 만들기

```
a$old <- as.integer(a$Age >= 65)
a$overweight <- as.integer(a$BMI >= 27)
a %>% mutate(Old = as.integer(Age >= 65), Overweight = as.integer(BMI >= 27))
```

```
a %>% mutate(Old = as.integer(Age >= 65), Overweight = as.integer(BMI >= 27)) %>% |
```

Sex <chr>	Age <int>	Height <int>	Weight <dbl>	BMI <dbl>
M	52	160	63.0	24.60938
M	67	162	63.0	24.00549
M	75	163	63.0	23.71184
F	66	154	61.0	25.72103
M	52	165	64.0	23.50781
M	56	166	70.0	25.40282
F	67	155	56.0	23.30905

transmute: 만든 변수만 보여주기

```
a %>% transmute(Old = as.integer(Age >= 65),
                Overweight = as.integer(BMI >= 27)
                )
```

```
a %>% transmute(Old = as.integer(Age >= 65),
                Overweight = as.integer(BMI >= 27)
                ) %>% paged_table
```

group_by & summarize

그룹으로 나누고, **요약통계량**을 구한다

```
a %>%
   group_by(Sex, Smoking) %>%
   summarize(count = n(),                ## n()는 샘플수
             meanBMI = mean(BMI),
             sdBMI = sd(BMI))
```

Sex	Smoking	count	meanBMI	sdBMI
<chr>	<int>	<int>	<dbl>	<dbl>
F	0	242	32.76081	88.44981
F	1	12	104.36833	281.74522
M	0	493	42.47247	130.59380
M	1	253	32.51974	86.49165

4 rows

summarize_all 모든 변수에 적용

```
a %>%
  filter(Age >= 50) %>%
  group_by(Sex, Smoking) %>%
  summarize_all(mean)
```

Sex <chr>	Smoking <int>	Age <dbl>	Height <dbl>	Weight <dbl>
F	0	68.32456	153.4035	58.96096
F	1	64.33333	151.4444	53.11111
M	0	64.62500	166.3380	69.17037
M	1	62.88785	166.4299	68.87336

4 rows | 1-5 of 14 columns

범주형 변수의 평균은 NA 로 나온다.

summarize_all with 여러 함수

```
a %>%
  filter(Age >= 50) %>%
  select(-STRESS_EXIST) %>%        ## Except categorical variable
  group_by(Sex, Smoking) %>%
  summarize_all(funs(mean = mean, sd = sd))
```

Sex <chr>	Smoking <int>	Age_mean <dbl>	Height_mean <dbl>	Weight_mean <dbl>
F	0	68.32456	153.4035	58.96096
F	1	64.33333	151.4444	53.11111
M	0	64.62500	166.3380	69.17037
M	1	62.88785	166.4299	68.87336

4 rows | 1-5 of 24 columns

실습 5: 실습 3과 비교

50세 이상만 뽑아서, **성별과 흡연력** 에 따른 **모든 변수**들의 **평균, 표준편차**를 구하라.

```
a %>%
  subset(Age >= 50) %>%
  aggregate(. ~ Sex + Smoking, data = .,
            FUN = function(x){c(mean = mean(x), sd = sd(x))})
```

```
a %>%
  filter(Age >= 50) %>%
  select(-Patient_ID, -STRESS_EXIST) %>%       ## Except categorical variable
  group_by(Sex, Smoking) %>%
  summarize_all(funs(mean = mean, sd = sd))
```

미리보기가 없음

Executive Summary

tidyverse는 직관적인 코드를 장점으로 원래의 R 문법을 빠르게 대체하고 있다.

- magrittr 패키지의 %>% 연산자로 의식의 흐름대로 코딩한다.
- dplyr 패키지의 select, mutate, filter, group_by, summarize 함수는 %>% 와 찰떡궁합이다.

END