빅데이터 분석모듈 개발하기

R를 이용한 빅데이터 분석

작성자: 이슬이

내용

빅데이터 분석모듈 개발하기

오픈 소스 분석도구 R

R 는 오픈 소스 프로그램으로 통계 및 데이터마이닝 알고리즘을 지원하고, 분석 결과를 직관적으로 이해할 수 있는 시각화 환경 및 소프트웨어를 제공한다. 미국 Bell Lab 에서 개발된 데이터 분석용 객체 지향 언어인 S 를 기반으로 1990 년대에 개발된 R 언어는 1997 년 12 월 GNU 프로젝트로 편입되면서 사용처가 폭발적으로 확산되었다. R 의 가장 큰 특징은 다양한 최신 통계 분석 및 마이닝 기능을 R 플랫폼에서 제공한다는 것이다. 상용 패키지들은 새로운 알고리즘을 적용하기까지 오랜 시간이 걸리지만, R 는 다양한 기능을 지원하는 8,000 여 개에 이르는 패키지가 수시로 업데이트된다. 최신 알고리즘을 이용해서 새로운 시도를 할 수 있다는 장점과 함께 R 언어라 어렵지 않으면서 사용자들이 커뮤니티를 통해 예시를 공유한다는 점이다. 별도의 교육 없이 데이터 분석을 공부하고 활용할 수 있다는 점이 주요 장점이다.

dplyr 패키지에 있는 함수 중 많이 사용하는 종류는 다음과 같다.

dplyr 함수	**기능**
filter()	행 추출
select()	열 추출
arrange()	정렬
mutate()	변수 추가
summarise()	통계치 산출
group_by()	집단 별로 나누기
left_join()	데이터 합치기(열)
bind_rows()	데이터 합치기(행)

dplyr 패키지 함수 이용하여 수행

1) `exam.csv` 파일의 데이터를 데이터프레임 출력
2) math, english, science 변수만 갖는 데이터프레임 출력
3) class 가 1 인 모든 변수를 갖는 데이터프레임 출력
4) math 가 60 점 이상이고 80 점 미만 데이터프레임 출력
5) english 가 60 점 이상이고 80 점 미만 데이터프레임 출력
6) math 가 60 점 이상이고 점수가 높은 순서를 갖는 class, id, math 변수를 갖는 데이터프레임 출력
7) class 로 그룹화 되고 수학점수 평균(mean_math)변수를 갖는 데이터프레임 출력
8) total(math, english, science 의 합) 파생변수를 갖는 데이터프레임 출력
9) mean(math, english, science 의 합의 평균) 파생변수를 갖는 데이터프레임 출력
10) grade(평균의 등급, A, B, C, D, F) 파생변수를 갖는 데이터프레임 출력

1. 먼저 패키지 함수를 설치하고 로드한다.

```
install.packages("dplyr")
library(dplyr)
```

1) `exam.csv` 파일의 데이터를 데이터프레임 출력

```
df_exam <- read.csv("./file/exam.csv")
df_exam
```

```
> df_exam <- read.csv("./file/exam.csv")
> df_exam
   id class math english science
1   1     1   50      98      50
2   2     1   60      97      60
3   3     1   45      86      78
4   4     1   30      98      58
5   5     2   25      80      65
6   6     2   50      89      98
7   7     2   80      90      45
8   8     2   90      78      25
9   9     3   20      98      15
10 10     3   50      98      45
11 11     3   65      65      65
12 12     3   45      85      32
13 13     4   46      98      65
14 14     4   48      87      12
15 15     4   75      56      78
16 16     4   58      98      65
17 17     5   65      68      98
18 18     5   80      78      90
19 19     5   89      68      87
20 20     5   78      83      58
```

2) math, english, science 변수만 갖는 데이터프레임 출력

```
df_math <- df_exam %>% select(math)
df_math

df_english <- df_exam %>% select(math)
df_english

df_science <- df_exam %>% select(math)
df_science
```

```
> df_math    <- df_exam %>% select(math)
> df_math
   math
1    50
2    60
3    45
4    30
5    25
6    50
7    80
8    90
9    20
10   50
11   65
12   45
13   46
14   48
15   75
16   58
17   65
18   80
19   89
20   78
```

```
> df_english <- df_exam %>% select(english)
> df_english
   english
1       98
2       97
3       86
4       98
5       80
6       89
7       90
8       78
9       98
10      98
11      65
12      85
13      98
14      87
15      56
16      98
17      68
18      78
19      68
20      83
```

```
> df_science <- df_exam %>% select(science)
> df_science
   science
1       50
2       60
3       78
4       58
5       65
6       98
7       45
8       25
9       15
10      45
11      65
12      32
13      65
14      12
15      78
16      65
17      98
18      90
19      87
20      58
```

3) class가 1인 모든 변수를 갖는 데이터프레임 출력

* select(everything())은 생략이 가능하다.

```
df_class <- df_exam %>% select(everything()) %>% filter(class == 1)
df_class
```

```
> df_class <- df_exam %>% select(everything()) %>% filter(class == 1)
> df_class
  id class math english science
1  1     1   50      98      50
2  2     1   60      97      60
3  3     1   45      86      78
4  4     1   30      98      58
```

4) math 가 60점 이상이고 80점 미만 데이터프레임 출력

```
df_filter1 <- df_exam %>% select(everything()) %>% filter(math >= 60 & math < 80)
df_filter1
```

```
> df_filter1 <- df_exam %>% select(everything()) %>% filter(math >= 60 & math < 80)
> df_filter1
  id class math english science
1  2     1   60      97      60
2 11     3   65      65      65
3 15     4   75      56      78
4 17     5   65      68      98
5 20     5   78      83      58
```

5) english 가 60점 이상이고 80점 미만 데이터프레임 출력

```
df_filter2 <- df_exam %>% filter(english >= 60 & english < 80)
df_filter2
```

```
> df_filter2 <- df_exam %>% filter(english >= 60 & english < 80)
> df_filter2
  id class math english science
1  8     2   90      78      25
2 11     3   65      65      65
3 17     5   65      68      98
4 18     5   80      78      90
5 19     5   89      68      87
```

6) math 가 60점 이상이고 점수가 높은 순서를 갖는 class, id, math 변수를 갖는 데이터프레임 출력

```
df_math_desc <- df_exam %>% select(everything()) %>% filter(math >= 60) %>% arrange(desc(math))
df_math_desc
```

```
> df_math_desc <- df_exam %>% select(everything()) %>% filter(math >= 60) %>% arrange(desc(math))
> df_math_desc
  id class math english science
1  8     2   90      78      25
2 19     5   89      68      87
3  7     2   80      90      45
4 18     5   80      78      90
5 20     5   78      83      58
6 15     4   75      56      78
7 11     3   65      65      65
8 17     5   65      68      98
9  2     1   60      97      60
```

7) class로 그룹화 되고 수학점수 평균(mean_math)변수를 갖는 데이터프레임 출력

```
df_group1 <- df_exam %>% group_by(class) %>% summarise(mean_math = mean(math))
df_group1
```

```
> df_group1 <- df_exam %>% group_by(class) %>% summarise(mean_math = mean(math))
> df_group1
# A tibble: 5 x 2
  class mean_math
* <int>     <dbl>
1     1      46.2
2     2      61.2
3     3      45
4     4      56.8
5     5      78
```

8) total(math, english, science의 합) 파생변수를 갖는 데이터프레임 출력

```
df_total <- df_exam %>% mutate(total = math + english + science)
df_total
```

```
> df_total <- df_exam %>% mutate(total = math + english + science)
> df_total
   id class math english science total
1   1     1   50      98      50   198
2   2     1   60      97      60   217
3   3     1   45      86      78   209
4   4     1   30      98      58   186
5   5     2   25      80      65   170
6   6     2   50      89      98   237
7   7     2   80      90      45   215
8   8     2   90      78      25   193
9   9     3   20      98      15   133
10 10     3   50      98      45   193
11 11     3   65      65      65   195
12 12     3   45      85      32   162
13 13     4   46      98      65   209
14 14     4   48      87      12   147
15 15     4   75      56      78   209
16 16     4   58      98      65   221
17 17     5   65      68      98   231
18 18     5   80      78      90   248
19 19     5   89      68      87   244
20 20     5   78      83      58   219
```

9) mean(math, english, science의 합의 평균) 파생변수를 갖는 데이터프레임 출력

```
df_mean <- df_exam %>% mutate(total = math + english + science) %>% mutate(mean = total/3)
df_mean
```

```
> df_mean <- df_exam %>% mutate(total = math + english + science) %>% mutate(mean = total/3)
> df_mean
   id class math english science total     mean
1   1     1   50      98      50   198 66.00000
2   2     1   60      97      60   217 72.33333
3   3     1   45      86      78   209 69.66667
4   4     1   30      98      58   186 62.00000
5   5     2   25      80      65   170 56.66667
6   6     2   50      89      98   237 79.00000
7   7     2   80      90      45   215 71.66667
8   8     2   90      78      25   193 64.33333
9   9     3   20      98      15   133 44.33333
10 10     3   50      98      45   193 64.33333
11 11     3   65      65      65   195 65.00000
12 12     3   45      85      32   162 54.00000
13 13     4   46      98      65   209 69.66667
14 14     4   48      87      12   147 49.00000
15 15     4   75      56      78   209 69.66667
16 16     4   58      98      65   221 73.66667
17 17     5   65      68      98   231 77.00000
18 18     5   80      78      90   248 82.66667
19 19     5   89      68      87   244 81.33333
20 20     5   78      83      58   219 73.00000
```

10) grade(평균의 등급, A, B, C, D, F) 파생변수를 갖는 데이터프레임 출력

```
df_grade <- df_exam %>% mutate(total = math + english + science) %>%
                        mutate(mean = total/3) %>%
                        mutate(grade = ifelse(mean >= 90, "A",
                                              ifelse(mean >= 80, "B",
                                              ifelse(mean >= 70, "C",
                                              ifelse(mean >= 60, "D", "F")))))
df_grade
```

```
> df_grade <- df_exam %>% mutate(total = math + english + science) %>%
+              mutate(mean = total/3) %>%
+              mutate(grade = ifelse(mean >= 90, "A",
+                     ifelse(mean >= 80, "B",
+                     ifelse(mean >= 70, "C",
+                     ifelse(mean >= 60, "D", "F")))))
> df_grade
   id class math english science total     mean grade
1   1     1   50      98      50   198 66.00000     D
2   2     1   60      97      60   217 72.33333     C
3   3     1   45      86      78   209 69.66667     D
4   4     1   30      98      58   186 62.00000     D
5   5     2   25      80      65   170 56.66667     F
6   6     2   50      89      98   237 79.00000     C
7   7     2   80      90      45   215 71.66667     C
8   8     2   90      78      25   193 64.33333     D
9   9     3   20      98      15   133 44.33333     F
10 10     3   50      98      45   193 64.33333     D
11 11     3   65      65      65   195 65.00000     D
12 12     3   45      85      32   162 54.00000     F
13 13     4   46      98      65   209 69.66667     D
14 14     4   48      87      12   147 49.00000     F
15 15     4   75      56      78   209 69.66667     D
16 16     4   58      98      65   221 73.66667     C
17 17     5   65      68      98   231 77.00000     C
18 18     5   80      78      90   248 82.66667     B
19 19     5   89      68      87   244 81.33333     B
20 20     5   78      83      58   219 73.00000     C
```