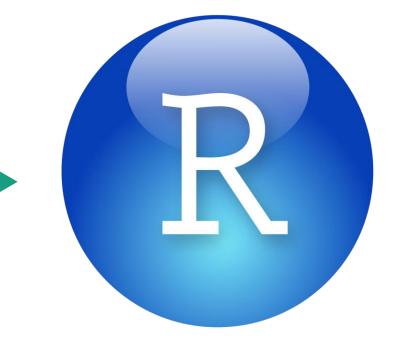
毒性学のためのR分析

3. データの前処理(2)



主題

dplyrパッケージでデータを加工する方法

なぜdplyrを使うのか

速い!

プログラミングが直観的!

Rの基本文法より簡単!

パッケージのインストール方法

Install.packages("~~")を使う

install.packages('dplyr')

パッケージのインストール方法

インストールしたパッケージはlibrary(~~)でロードする

```
> library(dplyr)
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
Warning message:
package 'dplyr' was built under R version 4.0.5
```

dplyrの二つの表記方法

一般的な表記方法

```
filter(df, Average >= 95)
A tibble: 6 x 5
Student Midterm Final Average Grade
<chr>
           <db1> <db1>
                          <db1> <chr>
A005
             100
                   100
                          100
                                A+
A006
             100
                     96
                           98
                                A+
A022
              99
                    92
                           95.5 A+
A063
              97
                     95
                           96
                                A+
A092
                   100
                           96
                                A+
A099
             100
                   100
                          100
                                A+
```

チェーニング (Chaining)

コードの簡略化ができるため、おすすめ!

```
df %>% filter(Average >= 95)
  A tibble: 6 x 5
  Student Midterm Final Average Grade
  <chr>
             <db1> <db1>
                            <db1> <chr>
1 A005
               100
                     100
                           100
                                  A+
2 A006
               100
                      96
                             98
 A022
                             95.5 A+
                      92
 A063
                97
                      95
                             96
                                  A+
 A092
                     100
                             96
                                  A+
6 A099
               100
                     100
                            100
                                  A+
```

先ずこれだけ覚えておけばOK!

- filter
- select
- arrange
- rename
- distinct
- mutate

dplyr関数1:filter()

条件に合う資料だけ選別することができる

条件文(>, <, ==,!=,>=, <=, &, |) を使って指定する</p>

dplyr関数1:filter()

例)Midtermが80点以上で、成績がAのデータを選別

```
> filter(df, Midterm >= 80 & Grade == 'A')
 A tibble: 10 x 5
  Student Midterm Final Average Grade
  <chr> <db1> <db1> <db1> <db1> <chr>
1 A001
             100
                   85 92.5 A
                   88 91.5 A
2 A011
              95
3 A012
              92
                   90 91 A
4 A014
              94
                   92 93 A
5 A039
              96
                   93 94.5 A
6 A040
              82
                  100
                         91 A
7 A068
              93
                   94
                         93.5 A
8 A077
              95
                   85
                         90
                             Α
9 A086
                   94
              94
                         94
                             Α
10 A098
                   86
                         90
              94
                              A
```

dplyr関数1:filter()

例)Midtermが80点以上で、成績がAのデータを選別

```
df \frac{\%}{\%} filter(Midterm >= 80 \& Grade == 'A')
 A tibble: 10 x 5
  Student Midterm Final Average Grade
  <chr> <db1> <db1> <db1> <db1> <chr>
1 A001
            100
                  85 92.5 A
2 A011 95 88 91.5 A
3 A012 92 90
                       91 A
4 A014
       94 92
                       93 A
5 A039
       96 93
                       94.5 A
6 A040
             82
                 100
                       91 A
7 A068
             93
                       93.5 A
                  94
8 A077
             95 85
                       90 A
9 A086
                  94
                       94 A
             94
10 A098
                  86
                       90
             94
                           Α
```

dplyr関数2:arrange()

■ データを基準に合わせて昇順、降順に整列する

降順の場合、desc(~)を使う

dplyr関数2:arrange()

例)Averageで降順整列

```
> arrange(df, desc(Average))
 A tibble: 100 x 5
   Student Midterm Final Average Grade
             <db1> <db1> <db1> <chr>
   <chr>
 1 A005
              100
                    100
                          100
                                A+
 2 A099
              100
                    100
                          100 A+
 3 A006
              100 96
                           98 A+
 4 A063
               97
                     95
                           96 A+
               92
                           96 A+
 5 A092
                    100
 6 A022
               99
                     92
                           95.5 A+
 7 A039
               96
                     93
                           94.5 A
                           94
 8 A086
               94
                     94
 9 A068
                           93.5 A
               93
                     94
               94
                     92
10 A014
                           93
                                Α
   .. with 90 more rows
  i Use `print(n = ...)` to see more rows
```

dplyr関数2:arrange()

例)Averageで降順整列

```
df %>% arrange(desc(Average))
 A tibble: 100 x 5
  Student Midterm Final Average Grade
            <db1> <db1> <db1> <chr>
   <chr>
 1 A005
              100
                    100
                          100
                                A+
 2 A099
              100
                    100
                          100 A+
 3 A006
              100
                     96
                           98 A+
               97
                     95
                           96 A+
 4 A063
 5 A092
               92
                    100
                           96 A+
 6 A022
               99
                     92
                           95.5 A+
 7 A039
               96
                     93
                           94.5 A
 8 A086
               94
                     94
                           94
 9 A068
               93
                     94
                           93.5 A
10 A014
               94
                     92
                           93
                                Α
  .. with 90 more rows
   Use `print(n = ...)` to see more rows
```

dplyr関数3:select()

選択したい変数だけ選択することができる

条件文を使って選択することもできる

dplyr関数3:select()

例)StudentとGradeの変数のみ選別

```
> select(df, Student, Grade)
# A tibble: 100 x 2
   Student Grade
   <chr> <chr>
1 A001 A
 2 A002
 3 A003
 4 A004
 5 A005
 6 A006
        A+
 7 A007
 8 A008
 9 A009
10 A010
  ... with 90 more rows
  i Use `print(n = ...)` to see more rows
```

dplyr関数3:select()

例)StudentとGradeの変数のみ選別

```
> df %>% select(Student, Grade)
# A tibble: 100 x 2
   Student Grade
   <chr> <chr>
1 A001 A
 2 A002
 3 A003
 4 A004
 5 A005
 6 A006
        A+
 7 A007
 8 A008
 9 A009
10 A010
  ... with 90 more rows
  i Use `print(n = ...)` to see more rows
```

dplyr関数4:rename()

変数の名称を変更する

dplyr関数4:rename()

例)StudentをIDに変更する

```
> df2 <- df
  rename(df2, ID = Student)
  A tibble: 100 x 5
         Midterm Final Average Grade
   <chr>
           <db1> <db1> <db1> <chr>
 1 A001
             100
                          92.5 A
                    85
 2 A002
                          83
              72
                    94
                          27.5 F
 3 A003
              35
                    20
                          72 C
 4 A004
              56
                    88
 5 A005
             100
                   100
                         100 A+
                          98
 6 A006
            100
                    96
                               A+
 7 A007
              83
                    77
                          80
                               В
 8 A008
                          46.5 F
              50
                    43
 9 A009
              27
                    89
                          58
10 A010
                           4
  ... with 90 more rows
   Use \hat{print}(n = ...) to see more rows
```

dplyr関数4:rename()

例)StudentをIDに変更する

```
> df2 <- df
 df2 %>% rename(ID = Student)
  A tibble: 100 x 5
        Midterm Final Average Grade
  ID
          <db1> <db1> <db1> <chr>
  <chr>
 1 A001
           100
                  85
                     92.5 A
 2 A002
            72 94
                       83 B
                     27.5 F
 3 A003
            35 20
 4 A004
        56
                 88
                       72 C
 5 A005
        100
                 100
                      100 A+
 6 A006
                       98 A+
        100
                  96
            83 77
 7 A007
                       80
 8 A008
            50 43
                       46.5 F
 9 A009
            27
                  89
                       58
10 A010
                        4 F
  ... with 90 more rows
   Use `print(n = ...)` to see more rows
```

dplyr関数5:distinct()

■ 重複していないデータを出力する

dplyr関数5:distinct()

例)Gradeで重複がないデータを出力する

```
> distinct(df, Grade)
# A tibble: 9 x 1
  Grade
 <chr>
3 F
5 A+
6 C+
7 D
8 D+
9 B+
```

dplyr関数5:distinct()

例)Gradeで重複がないデータを出力する

```
> df %>% distinct(Grade)
 A tibble: 9 x 1
  Grade
  <chr>
 Α
2 B
6 C+
 D
8 D+
```

dplyr関数6: mutate()

新しいカラムを追加する

dplyr関数6: mutate()

例)Averageが98以上はScholarshipを標記

```
> df3 <- df
> mutate(df3, Scholarship = ifelse(Average >= 98, 'Yes', ' '))
# A tibble: 100 x 6
  Student Midterm Final Average Grade Scholarship
  <chr>
            <db1> <db1> <db1> <chr>
                                    <chr>
1 A001
             100
                    85 92.5 A
2 A002 72
                    94 83
3 A003
              35
                    20 27.5 F
              56
4 A004
                    88
                        72
                                    "Yes"
5 A005
             100
                   100
                         100
                             A+
6 A006
                                     "Yes"
             100
                    96
                          98
                             A+
7 A007
              83
                    77
                          80
                        46.5 F
8 A008
              50
                    43
9 A009
              27
                    89
                          58
                           4
10 A010
                     8
  ... with 90 more rows
   Use print(n = ...) to see more rows
```

dplyr関数6: mutate()

例)Averageが98以上はScholarshipを標記

```
df3 <- df
 df3 %>% mutate(Scholarship = ifelse(Average >= 98, 'Yes', ' '))
 A tibble: 100 x 6
  Student Midterm Final Average Grade Scholarship
            <db1> <db1> <db1> <chr>
  <chr>
                                      <chr>>
1 A001
              100
                           92.5 A
                     85
2 A002
               72 94
                           83
                  20
               35
3 A003
                           27.5 F
               56
4 A004
                     88
                           72 C
                                      "Yes"
5 A005
              100
                    100
                         100 A+
6 A006
              100
                     96
                           98
                                      "Yes"
                               A+
7 A007
                   77
                           80
               83
               50
                     43
                           46.5 F
8 A008
9 A009
               27
                     89
                           58
10 A010
                0
                      8
                            4
                                F
  .. with 90 more rows
  i Use `print(n = ...)` to see more rows
```

練習問題

1. 成績をAverage基準で降順整列した後、Averageが98以上の学生はYesと表記するScholarship列を作成し、Student列の名称はIDに修正しなさい。また、最終的にはScholarshipがYesな学生のみ示すこと。