Introduction to dplyr

Bytelow Dynamics

10/1/2017

Why dplyr?

- dplyr makes data manipulation process easy
- limited number of 'verb' functions for data manipulation tasks
- efficient backends for fast processing

dplyr functions

- filter: select observtions based on their values
- arrange: reorder rows
- select: select variables based on their names
- mutate: add new variables which are functions of existing variables
- summarise: collapse many values to a single value
- group_by: break down a dataset into specified groups of rows

How do they work?

1. First, pick a data frame
2. Then apply a function, describing what to do with the data frame
3. The result is a new data frame

Pipe

The pipe operator %>% is used to make codes more readable

Let's try these functions!

First, read the data file

```
scores <- read.csv('scores.csv', stringsAsFactors = FALSE )
```

Check the dataset

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

filter

Used to filter rows.

Example: Filter rows with Lab.2 score = 8

```
scores %>%
  filter(type == "Lab.2", score == 8) %>% head()
```

```
##   X.1  X Last.Name First.Name  type score
## 1  28  1         A          T Lab.2     8
## 2  35  8         C          T Lab.2     8
## 3  39 12         E          N Lab.2     8
## 4  41 14         G          R Lab.2     8
## 5  43 16         J          S Lab.2     8
## 6  46 19         M          H Lab.2     8
```

Exercise: Filter rows with either Lab.1 or Lab.14 scores > 5

```
scores %>%
  filter(type == "Lab.1" | type == "Lab.14", score > 5) %>% head()
```

```
##   X.1 X Last.Name First.Name  type score
## 1   3 3         B          J Lab.1    10
## 2   4 4         B          B Lab.1    10
## 3   5 5         C          M Lab.1    10
## 4   6 6         C          S Lab.1    10
## 5   7 7         C          L Lab.1    10
## 6   8 8         C          T Lab.1    10
```

With base R

```
scores[(scores$type == "Lab.1" | scores$type == "Lab.14") & (scores$score > 5),]
```

```
##     X.1  X Last.Name First.Name   type score
## 3     3  3         B          J  Lab.1    10
## 4     4  4         B          B  Lab.1    10
## 5     5  5         C          M  Lab.1    10
## 6     6  6         C          S  Lab.1    10
## 7     7  7         C          L  Lab.1    10
## 8     8  8         C          T  Lab.1    10
## 9     9  9         D          R  Lab.1    10
## 10   10 10         D          C  Lab.1    10
## 11   11 11         E          L  Lab.1    10
## 12   12 12         E          N  Lab.1    10
## 13   13 13         F          J  Lab.1    10
## 16   16 16         J          S  Lab.1    10
## 17   17 17         L          M  Lab.1    10
## 18   18 18         M          C  Lab.1    10
## 19   19 19         M          H  Lab.1    10
## 20   20 20         P          N  Lab.1    10
## 21   21 21         P          A  Lab.1     9
## 22   22 22         P          C  Lab.1    10
## 23   23 23         P          B  Lab.1    10
## 24   24 24         R          V  Lab.1    10
## 25   25 25         T          A  Lab.1    10
```

```
## 26   26 26         T          G  Lab.1    10
## 27   27 27         V          A  Lab.1     8
## 352 352  1         A          T Lab.14    10
## 354 354  3         B          J Lab.14    10
## 361 361 10         D          C Lab.14    10
## 365 365 14         G          R Lab.14    10
## 366 366 15         J          J Lab.14    10
## 368 368 17         L          M Lab.14    10
## 371 371 20         P          N Lab.14    10
## 374 374 23         P          B Lab.14    10
```

arrange

Used to reorder/sort data by columns

Example: Reorder the data by score in descending order.

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
scores %>% arrange(desc(score)) %>% head()
```

```
##   X.1  X Last.Name First.Name       type score
## 1 532 19         M          H    Midterm    89
## 2 521  8         C          T    Midterm    87
## 3 624  3         B          J Final.exam    87
## 4 640 19         M          H Final.exam    87
## 5 531 18         M          C    Midterm    85
## 6 537 24         R          V    Midterm    85
```

Exercise: Reorder the data by last name and first name (2 min)

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
scores %>% arrange(Last.Name, First.Name) %>% head()
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2  28 1         A          T Lab.2     8
```

```
## 3  55 1         A          T Lab.3    10
## 4  82 1         A          T Lab.4     0
## 5 109 1         A          T Lab.5     0
## 6 136 1         A          T Lab.6     5
```

select

Used to select columns. Useful when you work with a large dataset.

functions for select

- contains("abc")
- ends_with("abc")
- starts_with("abc")
- matches("abc")
- one_of(c("abc","def",...))
- num_range("abc", c(1,2,3,...))

Example: Select First name and score columns

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
scores %>%
  select(First.Name, score) %>% head()
```

```
##   First.Name score
## 1          T     0
## 2          M     0
## 3          J    10
## 4          B    10
## 5          M    10
## 6          S    10
```

Exercise: Delete columns Last.Name and First.Name. Use one of the select functions from above. (3 min)

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
scores %>%
  select(-(ends_with("Name"))) %>% head()
```

```
##   X.1 X  type score
## 1   1 1 Lab.1     0
## 2   2 2 Lab.1     0
## 3   3 3 Lab.1    10
## 4   4 4 Lab.1    10
## 5   5 5 Lab.1    10
## 6   6 6 Lab.1    10
```

mutate

Adds a new column to the end of the dataset

Example: Filter rows with scores on a 100 scale (Midterm and Final) and add a column that contains scores converted into a 10 scale (0-10).

```
tail(scores)
```

```
##     X.1  X Last.Name First.Name       type score
## 643 643 22         P          C Final.exam    74
## 644 644 23         P          B Final.exam     0
## 645 645 24         R          V Final.exam    82
## 646 646 25         T          A Final.exam    64
## 647 647 26         T          G Final.exam    41
## 648 648 27         V          A Final.exam    32
```

```
exams <- scores %>%
  filter(type == "Midterm" | type == "Final.exam") %>%
  mutate(score_norm = score / 10)
head(exams)
```

```
##   X.1 X Last.Name First.Name    type score score_norm
## 1 514 1         A          T Midterm    40        4.0
## 2 515 2         B          M Midterm     0        0.0
## 3 516 3         B          J Midterm    72        7.2
## 4 517 4         B          B Midterm    68        6.8
## 5 518 5         C          M Midterm    81        8.1
## 6 519 6         C          S Midterm    48        4.8
```

We are making a new data frame so we can use it later.

Exercise: Make a new data frame that has filtered rows with scores on a 10 scale and a new column that contains scores converted into a 100 scale. (3 min)

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
lab_hw_quiz <- scores %>%
  filter(!type == "Midterm" & !type == "Final.exam" & type != "Extra.Credit") %>%
  mutate(score_norm = score * 10)
head(lab_hw_quiz)
```

```
##   X.1 X Last.Name First.Name  type score score_norm
## 1   1 1         A          T Lab.1     0          0
## 2   2 2         B          M Lab.1     0          0
## 3   3 3         B          J Lab.1    10        100
## 4   4 4         B          B Lab.1    10        100
## 5   5 5         C          M Lab.1    10        100
## 6   6 6         C          S Lab.1    10        100
```

summarise

Collapses data into a single row by summarising.

Example: Summarise the exams data and calculate mean, min, and max scores

```
head(exams)
```

```
##   X.1 X Last.Name First.Name    type score score_norm
## 1 514 1         A          T Midterm    40        4.0
## 2 515 2         B          M Midterm     0        0.0
## 3 516 3         B          J Midterm    72        7.2
## 4 517 4         B          B Midterm    68        6.8
## 5 518 5         C          M Midterm    81        8.1
## 6 519 6         C          S Midterm    48        4.8
```

```
exams %>%
  summarise(mean = mean(score), min = min(score), max = max(score)) %>% head()
```

```
##       mean min max
## 1 61.40741   0  89
```

Excercise: Summarise the lab_hw_quiz data and calculate mean, min, and max of the normalized scores on a 100 scale (2 min)

```
head(lab_hw_quiz)
```

```
##   X.1 X Last.Name First.Name  type score score_norm
## 1   1 1         A          T Lab.1     0          0
## 2   2 2         B          M Lab.1     0          0
## 3   3 3         B          J Lab.1    10        100
## 4   4 4         B          B Lab.1    10        100
## 5   5 5         C          M Lab.1    10        100
## 6   6 6         C          S Lab.1    10        100
```

```
lab_hw_quiz %>%
  summarise(mean = mean(score_norm), min = min(score_norm), max = max(score_norm)) %>% head()
```

```
##       mean min max
```

```
## 1 70.07407   0 100
```

group_by

Breaks down a data frame into subsets. This does not change the data frame visually. When used in conjunction with another function, it is applied to each subset.

Example: Summarise and calculate mean exam score for each student

```
head(exams)
```

```
##   X.1 X Last.Name First.Name    type score score_norm
## 1 514 1         A          T Midterm    40        4.0
## 2 515 2         B          M Midterm     0        0.0
## 3 516 3         B          J Midterm    72        7.2
## 4 517 4         B          B Midterm    68        6.8
## 5 518 5         C          M Midterm    81        8.1
## 6 519 6         C          S Midterm    48        4.8
```

---

1. Group by student names
2. Summarise to calculate mean

```
exams %>%
  group_by(Last.Name, First.Name) %>%
  summarise(mean = mean(score)) %>% head()
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name  mean
##       <chr>      <chr> <dbl>
## 1         A          T  39.5
## 2         B          B  68.5
## 3         B          J  79.5
## 4         B          M   0.0
## 5         C          L  70.5
## 6         C          M  74.0
```

Exercise: Use lab_hw_quiz data to summarise and calculate mean score for each student (2 min)

```
head(lab_hw_quiz)
```

```
##   X.1 X Last.Name First.Name  type score score_norm
## 1   1 1         A          T Lab.1     0          0
## 2   2 2         B          M Lab.1     0          0
## 3   3 3         B          J Lab.1    10        100
## 4   4 4         B          B Lab.1    10        100
## 5   5 5         C          M Lab.1    10        100
## 6   6 6         C          S Lab.1    10        100
```

---

1. Group by student names
2. Summarise to calculate mean

```
lab_hw_quiz %>%
  group_by(Last.Name, First.Name) %>%
  summarise(mean = mean(score)) %>% head()
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name     mean
##       <chr>      <chr>    <dbl>
## 1         A          T 3.785714
## 2         B          B 8.180952
## 3         B          J 8.895238
## 4         B          M 0.000000
## 5         C          L 7.714286
## 6         C          M 7.095238
```

Example: Find the students who got best score for each type

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

---

1. Group by type
2. Filter rows with max score for each type

```
scores %>%
  group_by(type) %>%
  filter(score == max(score)) %>% head()
```

```
## # A tibble: 6 x 6
## # Groups:   type [1]
##     X.1     X Last.Name First.Name  type score
##   <int> <int>     <chr>      <chr> <chr> <dbl>
## 1     3     3         B          J Lab.1    10
## 2     4     4         B          B Lab.1    10
## 3     5     5         C          M Lab.1    10
## 4     6     6         C          S Lab.1    10
## 5     7     7         C          L Lab.1    10
## 6     8     8         C          T Lab.1    10
```

Exercise: Add a column that contains the difference from the mean score for each type

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

---

1. Group by type
2. Mutate to add a column that has the difference

```
scores %>%
  group_by(type) %>%
  mutate(diff = score - mean(score)) %>% head()
```

```
## # A tibble: 6 x 7
## # Groups:   type [1]
##     X.1     X Last.Name First.Name  type score      diff
##   <int> <int>     <chr>      <chr> <chr> <dbl>     <dbl>
## 1     1     1         A          T Lab.1     0 -8.407407
## 2     2     2         B          M Lab.1     0 -8.407407
## 3     3     3         B          J Lab.1    10  1.592593
## 4     4     4         B          B Lab.1    10  1.592593
## 5     5     5         C          M Lab.1    10  1.592593
## 6     6     6         C          S Lab.1    10  1.592593
```