

talk05 练习与作业

目录

0.1 练习和作业说明	1
0.2 Talk05 内容回顾	1
0.3 练习与作业：用户验证	1
0.4 练习与作业 1: dplyr 练习	2

0.1 练习和作业说明

将相关代码填写入以 “{r}” 标志的代码框中，运行并看到正确的结果；

完成后，用工具栏里的”Knit” 按钮生成 PDF 文档；

将 PDF 文档改为：姓名-学号-talk05 作业.pdf，并提交到老师指定的平台/钉群。

0.2 Talk05 内容回顾

- dplyr 、tidyr (超级强大的数据处理) part 1
 - pipe
 - dplyr 几个重要函数

0.3 练习与作业：用户验证

请运行以下命令，验证你的用户名。

如你当前用户名不能体现你的真实姓名，请改为拼音后再运行本作业！

```
Sys.info()[["user"]]
```

```
## [1] "lucas"
```

```
Sys.getenv("HOME")
```

```
## [1] "/Users/lucas"
```

```
getwd(); ## 显示当前工作目录
```

```
## [1] "/Users/lucas/Library/Mobile Documents/com~apple~CloudDocs/~aa学习/大二上/学习/
```

0.4 练习与作业 1: dplyr 练习

0.4.1 使用 mouse.tibble 变量做统计

- 每个染色体（或 scaffold）上每种基因类型的数量、平均长度、最大和最小长度，挑出最长和最短的基因
- 去掉含有 500 以下基因的染色体（或 scaffold），按染色体（或 scaffold）、数量高 -> 低进行排序

挑战题（可选做）：

实现上述目标（即：去掉少于 500 基因的染色体、排序、并统计）时不使用中间变量；

```
# Load the package  
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
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
```

```
## v forcats   1.0.0      v readr     2.1.4
```

```
## v ggplot2   3.4.3      v stringr  1.5.0
```

```
## v lubridate 1.9.2      v tibble   3.2.1
```

```
## v purrr     1.0.2      v tidyr    1.3.0
```

```
## -- Conflicts ----- tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()    masks stats::lag()
```

```
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts
```

```
# read mouse.tibble
```

```
mouse_tibble =
```

```
  read_delim(file = "../data/talk04/mouse_genes_biomart_sep2018.txt",
```

```
             delim = "\t",
```

```
             quote = "",
```

```
             show_col_types = FALSE)
```

```
# Convert the mouse.tibble to tibble
```

```
if(!is.tibble(mouse_tibble))
```

```
  mouse_tibble = as_tibble(mouse_tibble)
```

```
## Warning: `is.tibble()` was deprecated in tibble 2.0.0.
```

```
## i Please use `is_tibble()` instead.
```

```
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was  
## generated.
```

```
# Arrange the data  
gene_summary = mouse_tibble %>%  
  group_by(  
    `Chromosome/scaffold name`,  
    `Transcript type`) %>%  
  summarize(  
    Gene_Count = n(),  
    Avg_Length =  
      mean(`Transcript length (including UTRs and CDS)`),  
    Max_Length =  
      max(`Transcript length (including UTRs and CDS)`),  
    Min_Length =  
      min(`Transcript length (including UTRs and CDS)`),  
    Longest_Gene =  
      `Chromosome/scaffold name`[  
        which.max(  
          `Transcript length (including UTRs and CDS)`)],  
    Shortest_Gene =  
      `Chromosome/scaffold name`[  
        which.min(  
          `Transcript length (including UTRs and CDS)`)]  
  )
```

```
## `summarise()` has grouped output by 'Chromosome/scaffold name'. You can  
## override using the `.groups` argument.
```

```
# Remove the chromosome containing the following 500 genes  
filtered_gene_summary =  
  gene_summary %>%  
  group_by(  
    `Chromosome/scaffold name`  
  )
```

```

`Chromosome/scaffold name`) %>%
filter(
  sum(Gene_Count) > 500)

# Descending order by number of chromosomes and genes
sorted_gene_summary =
  filtered_gene_summary %>%
  arrange(
    `Chromosome/scaffold name`, desc(Gene_Count))

# Print the data
sorted_gene_summary

```

```

## # A tibble: 521 x 8
## # Groups:   Chromosome/scaffold name [21]
##   `Chromosome/scaffold name` `Transcript type` Gene_Count Avg_Length Max_Length
##   <chr>                      <chr>          <int>      <dbl>      <dbl>
## 1 1 protein_coding           3369      2700.      40378
## 2 1 retained_intron          1509      1748.       8483
## 3 1 processed_transc~         738       951.       7640
## 4 1 processed_pseudo~         627       728.       4530
## 5 1 TEC                       486      2241.       8163
## 6 1 nonsense_mediate~         480      1844.      10770
## 7 1 lincRNA                    457      1207.       9720
## 8 1 antisense                   315      1236.       7928
## 9 1 miRNA                      128       98.0        442
## 10 1 snRNA                     105       113.        191
## # i 511 more rows
## # i 3 more variables: Min_Length <dbl>, Longest_Gene <chr>, Shortest_Gene <chr>

```

挑战题:

```
# Load the packages
library(dplyr)
library(tidyverse)

# read mouse.tibble
mouse_tibble2 =
  read_delim(file = "../data/talk04/mouse_genes_biomart_sep2018.txt",
             delim = "\t",
             quote = "",
             show_col_types = FALSE)

# Convert the mouse.tibble to tibble
if(!is.tibble(mouse_tibble2))
  mouse_tibble2 = as_tibble(mouse_tibble2)

# Processing data
sorted_gene_summary2 =
  mouse_tibble2 %>%
  group_by(
    `Chromosome/scaffold name`,
    `Transcript type`) %>%
  summarize(
    Gene_Count = n(),
    Avg_Length =
      mean(`Transcript length (including UTRs and CDS)`),
    Max_Length =
      max(`Transcript length (including UTRs and CDS)`),
    Min_Length =
      min(`Transcript length (including UTRs and CDS)`),
    Longest_Gene =
      `Chromosome/scaffold name`[
        which.max(
          `Transcript length (including UTRs and CDS)`)],
```

```

Shortest_Gene =
  `Chromosome/scaffold name`[
    which.min(
      `Transcript length (including UTRs and CDS)`)]
)%>%
ungroup() %>%
group_by(
  `Chromosome/scaffold name`) %>%
filter(
  sum(Gene_Count) > 500) %>%
# ungroup() %>%
arrange(
  `Chromosome/scaffold name`, desc(Gene_Count))

```

`summarise()` has grouped output by 'Chromosome/scaffold name'. You can
override using the `.groups` argument.

```

# Print the data
sorted_gene_summary2

```

```

## # A tibble: 521 x 8
## # Groups:   Chromosome/scaffold name [21]
##   `Chromosome/scaffold name` `Transcript type` Gene_Count Avg_Length Max_Length
##   <chr>                      <chr>          <int>      <dbl>      <dbl>
## 1 1 protein_coding             3369      2700.      40378
## 2 1 retained_intron            1509      1748.       8483
## 3 1 processed_transc~          738        951.       7640
## 4 1 processed_pseudo~          627        728.       4530
## 5 1 TEC                        486      2241.       8163
## 6 1 nonsense_mediate~          480      1844.     10770
## 7 1 lincRNA                     457      1207.       9720
## 8 1 antisense                   315      1236.       7928
## 9 1 miRNA                       128        98.0        442

```

## 10 1	snRNA	105	113.	191
## # i 511 more rows				
## # i 3 more variables:	Min_Length <dbl>, Longest_Gene <chr>, Shortest_Gene <chr>			

0.4.2 使用 grades2 变量做练习

首先，用下面命令生成 grades2 变量：

```
grades2 <- tibble( "Name" = c("Weihua Chen", "Mm Hu", "John Doe", "Jane Doe",
                             "Warren Buffet", "Elon Musk", "Jack Ma"),
                  "Occupation" = c("Teacher", "Student", "Teacher", "Student",
                                   rep( "Entrepreneur", 3 ) ),
                  "English" = sample( 60:100, 7 ),
                  "ComputerScience" = sample(80:90, 7),
                  "Biology" = sample( 50:100, 7),
                  "Bioinformatics" = sample( 40:90, 7)
                  );
```

然后统计：1. 每个人最差的学科和成绩分别是什么？ 2. 哪个职业的平均成绩最好？ 3. 每个职业的最佳学科分别是什么（按平均分排序）???

```
## 代码写这里，并运行；

# Load the tibble
grades2 =
  tibble(
    "Name" = c("Weihua Chen", "Mm Hu",
               "John Doe", "Jane Doe",
               "Warren Buffet", "Elon Musk",
               "Jack Ma"),
    "Occupation" = c("Teacher", "Student",
                     "Teacher", "Student",
```



```
      rep("Entrepreneur", 3)),
  "English" = sample(60:100, 7),
  "ComputerScience" = sample(80:90, 7),
  "Biology" = sample(50:100, 7),
  "Bioinformatics" = sample(40:90, 7))

# Question 1
# Use dplyr to find everyone's worst subjects and grades
worst_subject_per_person =
  grades2 %>%
  mutate(
    Min_Score = pmin(English,
                     ComputerScience,
                     Biology,
                     Bioinformatics)) %>%
  select(Name, Min_Score)

print(worst_subject_per_person)
```

```
## # A tibble: 7 x 2
##   Name      Min_Score
##   <chr>      <int>
## 1 Weihua Chen      81
## 2 Mm Hu            59
## 3 John Doe         40
## 4 Jane Doe         47
## 5 Warren Buffet    61
## 6 Elon Musk        42
## 7 Jack Ma         66
```

```
# Question 2
# Use dplyr to calculate the average score for each occupation
# Find the occupation with the highest average score
```

```
best_occupation =
  grades2 %>%
  group_by(Occupation) %>%
  summarise(
    Avg_Score =
      mean(English +
            ComputerScience +
            Biology +
            Bioinformatics)/4) %>%
  arrange(desc(Avg_Score)) %>%
  slice(1)

print(best_occupation)
```

```
## # A tibble: 1 x 2
##   Occupation Avg_Score
##   <chr>       <dbl>
## 1 Student      81.2
```

```
# Question 3
# Use dplyr to calculate the average score
# for each subject for each occupation
# Find the subject with the highest average score
best_subject_per_occupation =
  grades2 %>%
  group_by(Occupation) %>%
  summarise(
    Best_Subject =
      which.max(c(mean(English),
                    mean(ComputerScience),
                    mean(Biology),
                    mean(Bioinformatics)))) %>%
  ungroup() %>%
```

```
mutate(  
  Best_Subject =  
    c("English",  
      "ComputerScience",  
      "Biology",  
      "Bioinformatics")[Best_Subject])  
  
print(best_subject_per_occupation)
```

```
## # A tibble: 3 x 2  
##   Occupation Best_Subject  
##   <chr>      <chr>  
## 1 Entrepreneur ComputerScience  
## 2 Student      Biology  
## 3 Teacher      ComputerScience
```

0.4.3 使用 starwars 变量做计算

1. 计算每个人的 BMI;
2. 挑选出肥胖 (BMI ≥ 30) 的人类, 并且只显示其 `name`, `sex` 和 `homeworld`;

```
## 代码写这里, 并运行;  
# Loading the dplyr library  
library(dplyr)  
  
# Calculate BMI for each individual  
starwars_bmi =  
  starwars %>%  
  mutate(  
    BMI = mass / (height/100)^2)
```

```
# Print the results of Q1
```

```
print(starwars_bmi)
```

```
## # A tibble: 87 x 15
```

```
##   name      height  mass hair_color skin_color eye_color birth_year sex  gender
##   <chr>      <int> <dbl> <chr>      <chr>      <chr>      <dbl> <chr> <chr>
## 1 Luke Sk~    172    77 blond      fair        blue        19   male masculi~
## 2 C-3PO      167    75 <NA>      gold        yellow      112  none masculi~
## 3 R2-D2      96    32 <NA>      white, bl~ red        33   none masculi~
## 4 Darth V~   202   136 none      white      yellow     41.9 male masculi~
## 5 Leia Or~   150    49 brown     light      brown       19   fema~ feminin~
## 6 Owen La~   178   120 brown, gr~ light      blue       52   male masculi~
## 7 Beru Wh~   165    75 brown     light      blue       47   fema~ feminin~
## 8 R5-D4      97    32 <NA>      white, red red        NA   none masculi~
## 9 Biggs D~   183    84 black     light      brown       24   male masculi~
## 10 Obi-Wan~  182    77 auburn, w~ fair        blue-gray   57   male masculi~
## # i 77 more rows
## # i 6 more variables: homeworld <chr>, species <chr>, films <list>,
## #   vehicles <list>, starships <list>, BMI <dbl>
```

```
# Pick out obese humans and screen for columns that need to be
```

```
obese_humanoids =
```

```
  starwars_bmi %>%
```

```
  filter(
```

```
    species == "Human"
```

```
    & BMI >= 30) %>%
```

```
  select(name, sex, homeworld)
```

```
# Print the result of Q2
```

```
print(obese_humanoids)
```

```
## # A tibble: 3 x 3
```

```
##   name      sex  homeworld
```

```
##   <chr>           <chr> <chr>
## 1 Darth Vader     male  Tatooine
## 2 Owen Lars       male  Tatooine
## 3 Jek Tono Porkins male  Bestine IV
```

3. 挑选出所有人类;
4. 按 BMI 将他们分为三组, <18, 18~25, >25, 统计每组的人数, 并用 barplot 进行展示; 注意: 展示时三组的按 BMI 从小到大排序;
5. 改变排序方式, 按每组人数从小到大排序;

```
## 代码写这里, 并运行;
# Select all human beings
humans =
  starwars_bmi %>%
  filter(species == "Human")

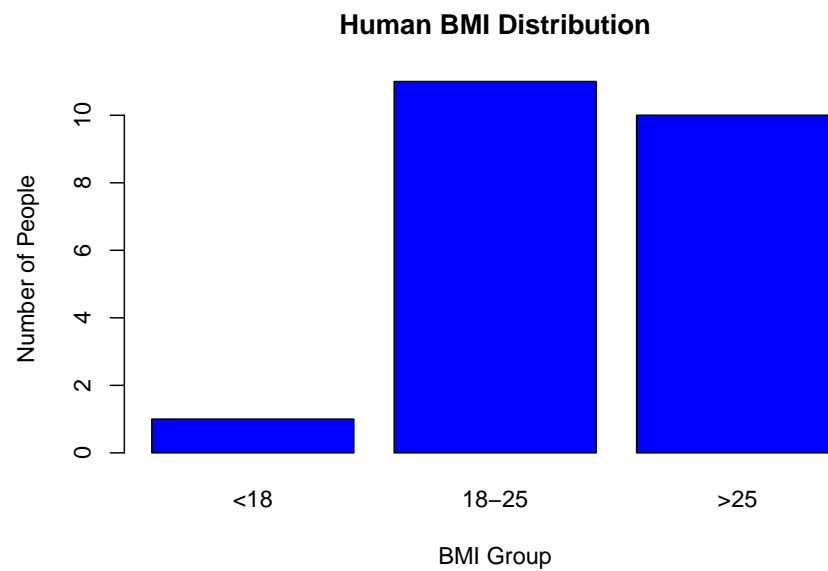
# Create three BMI groupings
humans$BMI_Group =
  cut(humans$BMI,
      breaks =
        c(-Inf, 18, 25, Inf),
      labels =
        c("<18", "18-25", ">25"))

# Counting the number of people in each group
group_counts =
  table(humans$BMI_Group)

# Sort by BMI from smallest to largest
group_counts =
  group_counts[
    order(
      as.numeric(names(group_counts)))]
```

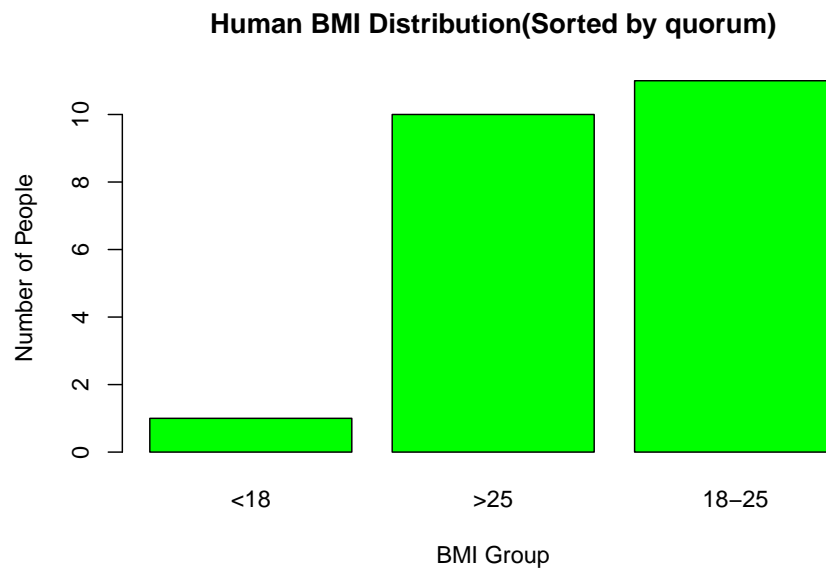
```
## Warning in order(as.numeric(names(group_counts))): NAs introduced by coercion
```

```
# Displayed using barplot  
barplot(  
  group_counts,  
  main = "Human BMI Distribution",  
  xlab = "BMI Group",  
  ylab = "Number of People",  
  col = "blue")
```



```
# Ordered from smallest to largest  
group_counts =  
  group_counts[order(group_counts)]  
  
# Displayed using barplot,  
# sorted by quorum  
# from smallest to largest  
barplot(  
  group_counts,
```

```
main = "Human BMI Distribution(Sorted by quorum)",  
xlab = "BMI Group",  
ylab = "Number of People",  
col = "green")
```



6. 查看 `starwars` 的 `films` 列，它有什么特点？`data.frame` 可以实现类似的功能吗？

答：

这一列中的所有元素都是以列表的形式存储在 `tibble` 中。

`data.frame` 不能够直接存储一些列表在某特定单元格内，但是可以通过其他的方式来实现类似的效果。

- List Column: 使用 `data.frame` 的一列来存储列表 (list) 数据，每个元素表示一个包含电影信息的子列表。这样可以模拟 `films` 列的结构。

```
films_list_colmn =
  data.frame(
    Name =
      c("Luke Skywalker",
        "Darth Vader"),
    Films =
      list(
        list(
          "A New Hope",
          "The Empire Strikes Back"),
        list(
          "A New Hope",
          "The Empire Strikes Back",
          "Return of the Jedi")
      )
  )

print(films_list_colmn)
```

```
##           Name Films..A.New.Hope. Films..The.Empire.Strikes.Back.
## 1 Luke Skywalker          A New Hope          The Empire Strikes Back
## 2   Darth Vader          A New Hope          The Empire Strikes Back
##  Films..A.New.Hope..1 Films..The.Empire.Strikes.Back..1
## 1           A New Hope          The Empire Strikes Back
## 2           A New Hope          The Empire Strikes Back
##  Films..Return.of.the.Jedi.
## 1           Return of the Jedi
## 2           Return of the Jedi
```

- 使用 Tidy Data Structure

```
# 创建一个整洁数据结构示例
films_tidy_data_str =
```



```
data.frame(  
  Name = c(  
    "Luke Skywalker",  
    "Luke Skywalker",  
    "Darth Vader",  
    "Darth Vader"),  
  Film = c(  
    "A New Hope",  
    "The Empire Strikes Back",  
    "A New Hope",  
    "The Empire Strikes Back")  
)  
  
print(films_tidy_data_str)
```

```
##           Name           Film  
## 1 Luke Skywalker      A New Hope  
## 2 Luke Skywalker The Empire Strikes Back  
## 3   Darth Vader      A New Hope  
## 4   Darth Vader The Empire Strikes Back
```

7. 为 `starwars` 增加一列，用于统计每个角色在多少部电影中出现。

```
## 代码写这里，并运行；  
# Adding a new column using the mutate function  
starwars_with_appearances =  
  starwars %>%  
  mutate(  
    Appearances =  
      rowSums(!is.na(. [5:12]))  
  )  
  
# Printing results  
# Including the new Appearances column
```

```
print(starwars_with_appearances)
```

```
## # A tibble: 87 x 15
##   name      height  mass hair_color skin_color eye_color birth_year sex  gender
##   <chr>      <int> <dbl> <chr>      <chr>      <chr>      <dbl> <chr> <chr>
## 1 Luke Sk~    172    77 blond      fair        blue        19   male masculi~
## 2 C-3P0      167    75 <NA>      gold        yellow      112  none masculi~
## 3 R2-D2       96    32 <NA>      white, bl~ red        33   none masculi~
## 4 Darth V~   202   136 none      white       yellow     41.9 male masculi~
## 5 Leia Or~   150    49 brown     light      brown       19   fema~ feminin~
## 6 Owen La~   178   120 brown, gr~ light      blue       52   male masculi~
## 7 Beru Wh~   165    75 brown     light      blue       47   fema~ feminin~
## 8 R5-D4       97    32 <NA>      white, red red        NA   none masculi~
## 9 Biggs D~   183    84 black     light      brown       24   male masculi~
## 10 Obi-Wan~  182    77 auburn, w~ fair        blue-gray   57   male masculi~
## # i 77 more rows
## # i 6 more variables: homeworld <chr>, species <chr>, films <list>,
## #   vehicles <list>, starships <list>, Appearances <dbl>
```

0.4.4 使用 Theoph 变量做练习

注：以下练习请只显示结果的前 6 行；

1. 选取从 Subject 到 Dose 的列；总共有几列？

```
## 代码写这里，并运行；
# Loading the Theoph dataset
data("Theoph")

# Select the columns from Subject to Dose.
selected_columns =
  Theoph[, c("Subject", "Wt", "Dose")]
```

```
# Calculate the total number of columns
num_columns =
  ncol(selected_columns)

# Show first 6 rows
head(selected_columns)
```

```
## Subject Wt Dose
## 1      1 79.6 4.02
## 2      1 79.6 4.02
## 3      1 79.6 4.02
## 4      1 79.6 4.02
## 5      1 79.6 4.02
## 6      1 79.6 4.02
```

```
# Total number of columns displayed
cat("There are ",
    num_columns,
    "columns in total.")
```

```
## There are 3 columns in total.
```

2. 用 `filter` 选取 Dose 大于 5, 且 Time 高于 Time 列平均值的行;

```
## 代码写这里, 并运行;

# Calculate the average of the Time columns
mean_time =
  mean(Theoph$Time)

# Use the filter function to
# select the rows that match the conditions
filtered_data =
```

```
filter(Theoph, Dose > 5,  
       Time > mean_time)  
  
# Display the first 6 rows of results  
head(filtered_data)
```

```
## Subject   Wt Dose  Time conc  
## 1         5 54.6 5.86  7.02 7.09  
## 2         5 54.6 5.86  9.10 5.90  
## 3         5 54.6 5.86 12.00 4.37  
## 4         5 54.6 5.86 24.35 1.57  
## 5        10 58.2 5.50  7.08 8.02  
## 6        10 58.2 5.50  9.38 7.14
```

3. 用 `mutate` 函数产生新列 `trend`, 其值为 `Time` 与 `Time` 列平均值的差;
注意: 请去除可能产生的 `na` 值;

```
## 代码写这里, 并运行;  
# Calculate the average  
# of the Time columns  
mean_time =  
  mean(Theoph$Time,  
       na.rm = TRUE)  
  
# Using the mutate function  
# to create a new column trend  
Theoph =  
  mutate(Theoph,  
         trend = Time - mean_time)  
  
# Removal of possible NA values  
Theoph =  
  na.omit(Theoph)
```

```
# Display the first 6 rows of results  
head(Theoph)
```

```
##   Subject   Wt Dose Time  conc   trend  
## 1         1 79.6 4.02 0.00  0.74 -5.894621  
## 2         1 79.6 4.02 0.25  2.84 -5.644621  
## 3         1 79.6 4.02 0.57  6.57 -5.324621  
## 4         1 79.6 4.02 1.12 10.50 -4.774621  
## 5         1 79.6 4.02 2.02  9.66 -3.874621  
## 6         1 79.6 4.02 3.82  8.58 -2.074621
```

4. 用 `mutate` 函数产生新列 `weight_cat`，其值根据 `Wt` 的取值范围而不同：

- 如果 `Wt > 76.2`，为 ‘Super-middleweight’，否则
- 如果 `Wt > 72.57`，为 ‘Middleweight’，否则
- 如果 `Wt > 66.68`，为 ‘Light-middleweight’
- 其它值，为 ‘Welterweight’

```
# Using the mutate function to create a new column weight_cat  
Theoph =  
  mutate(  
    Theoph,  
    weight_cat = case_when(  
      Wt > 76.2 ~ 'Super-middleweight',  
      Wt > 72.57 ~ 'Middleweight',  
      Wt > 66.68 ~ 'Light-middleweight',  
      TRUE ~ 'Welterweight'  
    )  
  )  
  
# Display the first 6 rows of results  
head(Theoph)
```

##	Subject	Wt	Dose	Time	conc	trend	weight_cat
## 1	1	79.6	4.02	0.00	0.74	-5.894621	Super-middleweight
## 2	1	79.6	4.02	0.25	2.84	-5.644621	Super-middleweight
## 3	1	79.6	4.02	0.57	6.57	-5.324621	Super-middleweight
## 4	1	79.6	4.02	1.12	10.50	-4.774621	Super-middleweight
## 5	1	79.6	4.02	2.02	9.66	-3.874621	Super-middleweight
## 6	1	79.6	4.02	3.82	8.58	-2.074621	Super-middleweight