

Week-4: Code-along

Janelle Tan

2023-09-03

II. Code to edit and execute using the Code-along.Rmd file

A. Data Wrangling

1. Loading packages (Slide #16)

```
# Load package tidyverse  
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(readr)
```

2. Loading data-set (Slide #16)

```
# Read data from the hotels.csv file and assign it to a variable named, "hotels"  
hotels <- read_csv("NM2207 W4 hotels.csv")
```

```
## Rows: 119390 Columns: 32
## — Column specification —————
## Delimiter: ","
## chr  (13): hotel, arrival_date_month, meal, country, market_segment, distrib...
## dbl  (18): is_canceled, lead_time, arrival_date_year, arrival_date_week_numb...
## date  (1): reservation_status_date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

3. List names of the variables in the data-set (Slide #19)

```
# Enter code here
names(hotels)
```

```
## [1] "hotel" "is_canceled"
## [3] "lead_time" "arrival_date_year"
## [5] "arrival_date_month" "arrival_date_week_number"
## [7] "arrival_date_day_of_month" "stays_in_weekend_nights"
## [9] "stays_in_week_nights" "adults"
## [11] "children" "babies"
## [13] "meal" "country"
## [15] "market_segment" "distribution_channel"
## [17] "is_repeated_guest" "previous_cancellations"
## [19] "previous_bookings_not_canceled" "reserved_room_type"
## [21] "assigned_room_type" "booking_changes"
## [23] "deposit_type" "agent"
## [25] "company" "days_in_waiting_list"
## [27] "customer_type" "adr"
## [29] "required_car_parking_spaces" "total_of_special_requests"
## [31] "reservation_status" "reservation_status_date"
```

4. Glimpse of contents of the data-set (Slide #20)

```
# Enter code here
glimpse(hotels)
```

```
## Rows: 119,390
## Columns: 32
## $ hotel                <chr> "Resort Hotel", "Resort Hotel", "Resort...
## $ is_canceled          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, ...
## $ lead_time            <dbl> 342, 737, 7, 13, 14, 14, 0, 9, 85, 75, ...
## $ arrival_date_year    <dbl> 2015, 2015, 2015, 2015, 2015, 2015, 201...
## $ arrival_date_month   <chr> "July", "July", "July", "July", "July",...
## $ arrival_date_week_number <dbl> 27, 27, 27, 27, 27, 27, 27, 27, 27, 27,...
## $ arrival_date_day_of_month <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ stays_in_weekend_nights <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ stays_in_week_nights <dbl> 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 4, 4, 4, ...
## $ adults               <dbl> 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, ...
## $ children             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ babies               <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ meal                 <chr> "BB", "BB", "BB", "BB", "BB", "BB", "BB", "BB...
## $ country              <chr> "PRT", "PRT", "GBR", "GBR", "GBR", "GBR...
## $ market_segment      <chr> "Direct", "Direct", "Direct", "Corporat...
## $ distribution_channel <chr> "Direct", "Direct", "Direct", "Corporat...
## $ is_repeated_guest    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ previous_cancellations <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ previous_bookings_not_canceled <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ reserved_room_type   <chr> "C", "C", "A", "A", "A", "A", "C", "C",...
## $ assigned_room_type   <chr> "C", "C", "C", "A", "A", "A", "C", "C",...
## $ booking_changes      <dbl> 3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ deposit_type         <chr> "No Deposit", "No Deposit", "No Deposit...
## $ agent                <chr> "NULL", "NULL", "NULL", "304", "240", "...
## $ company              <chr> "NULL", "NULL", "NULL", "NULL", "NULL",...
## $ days_in_waiting_list <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ customer_type        <chr> "Transient", "Transient", "Transient", ...
## $ adr                  <dbl> 0.00, 0.00, 75.00, 75.00, 98.00, 98.00,...
## $ required_car_parking_spaces <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ total_of_special_requests <dbl> 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 3, ...
## $ reservation_status   <chr> "Check-Out", "Check-Out", "Check-Out", ...
## $ reservation_status_date <date> 2015-07-01, 2015-07-01, 2015-07-02, 20...
```

B. Choosing rows or columns

5. Select a single column (Slide #24)

```
# Enter code here
select(
  hotels, # name of data set
  lead_time # name of column/variable
)
```

```
## # A tibble: 119,390 × 1
##   lead_time
##   <dbl>
## 1      342
## 2      737
## 3         7
## 4        13
## 5        14
## 6        14
## 7         0
## 8         9
## 9        85
## 10       75
## # i 119,380 more rows
```

Remember: dplyr functions always expect a dataframe and the output is also a dataframe

6. Select multiple columns (Slide #25)

Enter code here

```
select(hotels, lead_time, agent, market_segment)
```

```
## # A tibble: 119,390 × 3
##   lead_time agent market_segment
##   <dbl> <chr> <chr>
## 1      342 NULL Direct
## 2      737 NULL Direct
## 3         7 NULL Direct
## 4        13 304 Corporate
## 5        14 240 Online TA
## 6        14 240 Online TA
## 7         0 NULL Direct
## 8         9 303 Direct
## 9        85 240 Online TA
## 10       75 15 Offline TA/TO
## # i 119,380 more rows
```

Note: you can use -c("column_names") to remove specific columns

eg `hotels <- hotels %>%
select(-lead_time, -agent, -market_segment)`

used - operator to remove columns

7. Arrange entries of a column (Slide #28)

Enter code here

```
arrange(hotels, lead_time )
```

Default: arrange in INCREASING order

Re-arranging applies to the entire row and not just the entries of the column

```
## # A tibble: 119,390 × 32
##   hotel          is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>          <dbl>      <dbl>          <dbl> <chr>
## 1 Resort Hotel      0        0            2015 July
## 2 Resort Hotel      0        0            2015 July
## 3 Resort Hotel      0        0            2015 July
## 4 Resort Hotel      0        0            2015 July
## 5 Resort Hotel      0        0            2015 July
## 6 Resort Hotel      0        0            2015 July
## 7 Resort Hotel      0        0            2015 July
## 8 Resort Hotel      0        0            2015 July
## 9 Resort Hotel      0        0            2015 July
## 10 Resort Hotel     0        0            2015 July
## # i 119,380 more rows
## # i 27 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>, ...
```

8. Arrange entries of a column in the descending order (Slide #30)

```
# Enter code here
arrange(hotels, desc(lead_time))
```

```
## # A tibble: 119,390 × 32
##   hotel          is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>          <dbl>      <dbl>          <dbl> <chr>
## 1 Resort Hotel      0       737            2015 July
## 2 Resort Hotel      0       709            2016 February
## 3 City Hotel        1       629            2017 March
## 4 City Hotel        1       629            2017 March
## 5 City Hotel        1       629            2017 March
## 6 City Hotel        1       629            2017 March
## 7 City Hotel        1       629            2017 March
## 8 City Hotel        1       629            2017 March
## 9 City Hotel        1       629            2017 March
## 10 City Hotel       1       629            2017 March
## # i 119,380 more rows
## # i 27 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>, ...
```

9. Select columns and arrange the entries of a column (Slide

#31)

Enter code here

```
arrange(  
  select(hotels, lead_time),  
  desc(lead_time)  
)
```

1. verb
 2. 1st argument of dataframe
 3. 2nd argument; decreasing order
- note: nested function

```
## # A tibble: 119,390 × 1  
##   lead_time  
##   <dbl>  
## 1      737  
## 2      709  
## 3      629  
## 4      629  
## 5      629  
## 6      629  
## 7      629  
## 8      629  
## 9      629  
## 10     629  
## # i 119,380 more rows
```

10. Select columns and arrange the entries of a column using the pipe operator (Slide #37)

Enter code here

```
hotels %>%  
  select(lead_time) %>%  
  arrange(desc(lead_time))
```

```
## # A tibble: 119,390 × 1  
##   lead_time  
##   <dbl>  
## 1      737  
## 2      709  
## 3      629  
## 4      629  
## 5      629  
## 6      629  
## 7      629  
## 8      629  
## 9      629  
## 10     629  
## # i 119,380 more rows
```

11. Pick rows matching a condition (Slide #44)

```
# Enter code here
hotels %>%
  filter(children >= 1) %>%
  select(hotel, children)
```

```
## # A tibble: 8,590 × 2
##   hotel      children
##   <chr>         <dbl>
## 1 Resort Hotel      1
## 2 Resort Hotel      2
## 3 Resort Hotel      2
## 4 Resort Hotel      2
## 5 Resort Hotel      1
## 6 Resort Hotel      1
## 7 Resort Hotel      2
## 8 Resort Hotel      2
## 9 Resort Hotel      1
## 10 Resort Hotel     2
## # i 8,580 more rows
```

12. Pick rows matching multiple conditions (Slide #46)

```
# Enter code here
hotels %>%
  filter(children >= 1, hotel == "City Hotel") %>%
  select(hotel, children)
```

```
## # A tibble: 5,106 × 2
##   hotel      children
##   <chr>         <dbl>
## 1 City Hotel      1
## 2 City Hotel      2
## 3 City Hotel      1
## 4 City Hotel      1
## 5 City Hotel      1
## 6 City Hotel      1
## 7 City Hotel      1
## 8 City Hotel      1
## 9 City Hotel      1
## 10 City Hotel     1
## # i 5,096 more rows
```

13. Non-conditional selection of rows: sequence of indices (Slide #49)

```
# Enter code here
hotels %>% slice(1:5)
```

```
## # A tibble: 5 × 32
##   hotel          is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>          <dbl>      <dbl>          <dbl> <chr>
## 1 Resort Hotel      0        342            2015 July
## 2 Resort Hotel      0        737            2015 July
## 3 Resort Hotel      0         7            2015 July
## 4 Resort Hotel      0        13            2015 July
## 5 Resort Hotel      0        14            2015 July
## # i 27 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>,
## #   reserved_room_type <chr>, assigned_room_type <chr>, ...
```

14. Non-conditional selection of rows: non-consecutive/specific indices (Slide #50)

```
# Enter code here
hotels %>%
  slice(1,3,5)
```

```
## # A tibble: 3 × 32
##   hotel          is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>          <dbl>      <dbl>          <dbl> <chr>
## 1 Resort Hotel      0        342            2015 July
## 2 Resort Hotel      0         7            2015 July
## 3 Resort Hotel      0        14            2015 July
## # i 27 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>,
## #   reserved_room_type <chr>, assigned_room_type <chr>, ...
```

15. Pick unique rows using distinct() (Slide #52)

```
# Enter code here
hotels %>% distinct(hotel)
```

```
## # A tibble: 2 × 1
##   hotel
##   <chr>
## 1 Resort Hotel
## 2 City Hotel
```


C. Creating new columns

16. Creating a single column with mutate() (Slide #56)

```
# Enter code here
hotels %>%
  mutate(little_ones = children + babies) %>%
  select(hotel, little_ones, children, babies)
```

```
## # A tibble: 119,390 × 4
##   hotel          little_ones children babies
##   <chr>          <dbl>     <dbl> <dbl>
## 1 Resort Hotel      0         0      0
## 2 Resort Hotel      0         0      0
## 3 Resort Hotel      0         0      0
## 4 Resort Hotel      0         0      0
## 5 Resort Hotel      0         0      0
## 6 Resort Hotel      0         0      0
## 7 Resort Hotel      0         0      0
## 8 Resort Hotel      0         0      0
## 9 Resort Hotel      0         0      0
## 10 Resort Hotel     0         0      0
## # i 119,380 more rows
```

17. Creating multiple columns with mutate() (Slide #58)

```
# Enter code here
hotels %>%
  mutate(little_ones = children + babies,
         average_little_ones = mean(little_ones)) %>%
  select(hotel, little_ones, children, babies, average_little_ones)
```

The columns are separated by ,

The newer columns can even use a newly created column(s)

```
## # A tibble: 119,390 × 5
##   hotel          little_ones children babies average_little_ones
##   <chr>          <dbl>     <dbl> <dbl>          <dbl>
## 1 Resort Hotel      0         0      0             NA
## 2 Resort Hotel      0         0      0             NA
## 3 Resort Hotel      0         0      0             NA
## 4 Resort Hotel      0         0      0             NA
## 5 Resort Hotel      0         0      0             NA
## 6 Resort Hotel      0         0      0             NA
## 7 Resort Hotel      0         0      0             NA
## 8 Resort Hotel      0         0      0             NA
## 9 Resort Hotel      0         0      0             NA
## 10 Resort Hotel     0         0      0             NA
## # i 119,380 more rows
```

NA here cld be due the presence of NA is little_ones column

soln: use is.na function to check for NA values in little_ones column

D. More operations with examples

18. count() to get frequencies (Slide #60)

Enter code here

```
hotels %>%
```

```
  count(market_segment)
```

counts no. of entries in specified column

```
## # A tibble: 8 × 2
##   market_segment      n
##   <chr>          <int>
## 1 Aviation        237
## 2 Complementary    743
## 3 Corporate       5295
## 4 Direct         12606
## 5 Groups         19811
## 6 Offline TA/TO   24219
## 7 Online TA       56477
## 8 Undefined         2
```

Note: count() auto sorts into alphabetical order since entries are of type chr

19. count() to get frequencies with sorting of count (Slide #61)

Enter code here

```
hotels %>%
```

```
  count(market_segment, sort = TRUE)
```

to arrange in descending order

sort = FALSE -> no sorting in any order, will be as per the data set

```
## # A tibble: 8 × 2
##   market_segment      n
##   <chr>          <int>
## 1 Online TA       56477
## 2 Offline TA/TO   24219
## 3 Groups         19811
## 4 Direct         12606
## 5 Corporate       5295
## 6 Complementary    743
## 7 Aviation        237
## 8 Undefined         2
```

20. count() multiple variables (Slide #62)

Enter code here

```
hotels %>%
```

```
  count(hotel, market_segment)
```

```
## # A tibble: 14 × 3
##   hotel      market_segment    n
##   <chr>      <chr>          <int>
## 1 City Hotel Aviation         237
## 2 City Hotel Complementary    542
## 3 City Hotel Corporate      2986
## 4 City Hotel Direct         6093
## 5 City Hotel Groups       13975
## 6 City Hotel Offline TA/TO 16747
## 7 City Hotel Online TA     38748
## 8 City Hotel Undefined         2
## 9 Resort Hotel Complementary    201
## 10 Resort Hotel Corporate     2309
## 11 Resort Hotel Direct        6513
## 12 Resort Hotel Groups        5836
## 13 Resort Hotel Offline TA/TO  7472
## 14 Resort Hotel Online TA    17729
```

21. summarise() for summary statistics (Slide #63)

```
# Enter code here
hotels %>%
  summarise(mean_adr = mean(adr))
```

```
## # A tibble: 1 × 1
##   mean_adr
##   <dbl>
## 1    102.
```

22. summarise() by using group_by to find mean (Slide #64)

```
# Enter code here
hotels %>%
  group_by(hotel) %>%
  summarise(mean_adr = mean(adr))
```

```
## # A tibble: 2 × 2
##   hotel      mean_adr
##   <chr>      <dbl>
## 1 City Hotel    105.
## 2 Resort Hotel  95.0
```

23. summarise() by using group_by to get count (Slide #65)

```
# Enter code here

# ?n to see what n() function does

hotels %>%
  group_by(hotel) %>%
  summarise(count = n())
```

```
## # A tibble: 2 × 2
##   hotel      count
##   <chr>      <int>
## 1 City Hotel  79330
## 2 Resort Hotel 40060
```

```
# hotels %>% count(hotel) gvs the same output slide 66
```

24. summarise() for multiple summary statistics (Slide #67)

```
# Enter code here
hotels %>%
  summarise(
    min_adr = min(adr),
    mean_adr = mean(adr),
    median_adr = median(adr),
    max_adr = max(adr)
  )
```

```
## # A tibble: 1 × 4
##   min_adr mean_adr median_adr max_adr
##   <dbl>   <dbl>     <dbl>   <dbl>
## 1   -6.38    102.      94.6    5400
```

25. select(), slice() and arrange() (Slide #68)

```
# Enter code here
hotels %>%
  select(hotel, lead_time) %>%
  slice(1:5) %>%
  arrange(lead_time) arrange default in ascending order
```

```
## # A tibble: 5 × 2
##   hotel      lead_time
##   <chr>      <dbl>
## 1 Resort Hotel      7
## 2 Resort Hotel     13
## 3 Resort Hotel     14
## 4 Resort Hotel    342
## 5 Resort Hotel    737
```

26. select(), arrange() and slice() (Slide #69)

```
# Enter code here
hotels %>%
  select(hotel, lead_time) %>%
  arrange(lead_time) %>%
  slice(1:5)
```

```
## # A tibble: 5 × 2
##   hotel      lead_time
##   <chr>      <dbl>
## 1 Resort Hotel      0
## 2 Resort Hotel      0
## 3 Resort Hotel      0
## 4 Resort Hotel      0
## 5 Resort Hotel      0
```

#order of piping matters! compare output w before

27. filter() to select rows based on conditions (Slide #73)

```
# Enter code here
hotels %>%
  filter(
    adults == 0,
    children >= 1
  ) %>%
  select(adults, babies, children)
```

```
## # A tibble: 223 × 3
##   adults babies children
##   <dbl> <dbl>   <dbl>
## 1      0      0       3
## 2      0      0       2
## 3      0      0       2
## 4      0      0       2
## 5      0      0       2
## 6      0      0       3
## 7      0      1       2
## 8      0      0       2
## 9      0      0       2
## 10     0      0       2
## # i 213 more rows
```

28. filter() to select rows based on complicated conditions (Slide

#74)

```
# Enter code here
hotels %>%
  filter( adults == 1,
  children >= 1 | babies >=1) %>% # / means OR
  select(adults, babies, children)
```

```
## # A tibble: 450 × 3
##   adults babies children
##   <dbl> <dbl>   <dbl>
## 1     1     0       2
## 2     1     0       2
## 3     1     0       1
## 4     1     1       0
## 5     1     0       1
## 6     1     0       1
## 7     1     0       2
## 8     1     0       2
## 9     1     0       1
## 10    1     0       1
## # i 440 more rows
```

29. count() and arrange() (Slide #76)

```
# Enter code here
hotels %>%
  count(market_segment) %>%
  arrange(desc(n))
```

```
## # A tibble: 8 × 2
##   market_segment      n
##   <chr>          <int>
## 1 Online TA      56477
## 2 Offline TA/TO 24219
## 3 Groups        19811
## 4 Direct        12606
## 5 Corporate      5295
## 6 Complementary   743
## 7 Aviation       237
## 8 Undefined        2
```

30. mutate(), select() and arrange() (Slide #77)

```
# Enter code here
hotels %>%
  mutate(little_ones = children + babies) %>% # <---
  select(children, babies, little_ones) %>%
  arrange(desc(little_ones))
```

```
## # A tibble: 119,390 × 3
##   children babies little_ones
##   <dbl>   <dbl>   <dbl>
## 1      10      0         10
## 2       0     10         10
## 3       0      9          9
## 4       2      1          3
## 5       2      1          3
## 6       2      1          3
## 7       3      0          3
## 8       2      1          3
## 9       2      1          3
## 10      3      0          3
## # i 119,380 more rows
```

31. mutate(), filter() and select() (Slide #78)

```
# Enter code here
hotels %>%
  mutate(little_ones = children + babies) %>%
  filter(
    little_ones >= 1,
    hotel == "Resort Hotel"
  ) %>%
  select(hotel, little_ones)
```

```
## # A tibble: 3,929 × 2
##   hotel          little_ones
##   <chr>          <dbl>
## 1 Resort Hotel      1
## 2 Resort Hotel      2
## 3 Resort Hotel      2
## 4 Resort Hotel      2
## 5 Resort Hotel      1
## 6 Resort Hotel      1
## 7 Resort Hotel      2
## 8 Resort Hotel      2
## 9 Resort Hotel      1
## 10 Resort Hotel     1
## # i 3,919 more rows
```

```
hotels %>%
  mutate(little_ones = children + babies) %>%
  filter(
    little_ones >= 1,
    hotel == "City Hotel"
  ) %>%
  select(hotel, little_ones)
```

```
## # A tibble: 5,403 × 2
##   hotel      little_ones
##   <chr>      <dbl>
## 1 City Hotel      1
## 2 City Hotel      1
## 3 City Hotel      2
## 4 City Hotel      1
## 5 City Hotel      1
## 6 City Hotel      1
## 7 City Hotel      1
## 8 City Hotel      1
## 9 City Hotel      1
## 10 City Hotel     1
## # i 5,393 more rows
```