homework_4_kanutala_santosh

Santosh Kanutala

7/1/2018

My Github repository for my assignments can be found at below URL: (https://github.com/santumagic/compscix-415-2assignments.git)

```
library(tidyverse)
library(mdsr)
library(nycflights13)
```

Section 5.6.7: #2, #4 and #6 only. Extra Credit: Do #5

QUESTION 2:

```
# First lets find the not cancelled flights.
not_cancelled <- flights %>%
  filter(!is.na(dep_delay), !is.na(arr_delay))

# given code for not_cancelled %>% count(dest)
not_cancelled %>%
  count(dest)
```

```
## # A tibble: 104 x 2
##    dest      n
##    <chr> <int>
##  1 ABQ     254
##  2 ACK     264
##  3 ALB     418
##  4 ANC       8
##  5 ATL   16837
##  6 AUS    2411
##  7 AVL     261
##  8 BDL     412
##  9 BGR     358
## 10 BHM     269
## # ... with 94 more rows
```

```
# new code for not_cancelled %>% count(dest) by group by and summarise
not_cancelled %>%
  group_by(dest) %>%
  summarise(n = n())
```

```
## # A tibble: 104 x 2
##    dest      n
##    <chr> <int>
##  1 ABQ     254
##  2 ACK     264
##  3 ALB     418
##  4 ANC       8
##  5 ATL   16837
##  6 AUS    2411
##  7 AVL     261
##  8 BDL     412
##  9 BGR     358
## 10 BHM     269
## # ... with 94 more rows
```

```
# given code for not_cancelled %>% count(tailnum, wt = distance)
not_cancelled %>%
  count(tailnum, wt = distance)
```

```
## # A tibble: 4,037 x 2
##    tailnum      n
##    <chr>    <dbl>
##  1 D942DN    3418
##  2 N0EGMQ  239143
##  3 N10156  109664
##  4 N102UW   25722
##  5 N103US   24619
##  6 N104UW   24616
##  7 N10575  139903
##  8 N105UW   23618
##  9 N107US   21677
## 10 N108UW   32070
## # ... with 4,027 more rows
```

```
# new code for not_cancelled %>% count(tailnum, wt = distance) group by and summarise
not_cancelled %>%
  group_by(tailnum) %>%
  summarize(n = sum(distance, na.rm = TRUE))
```

```
## # A tibble: 4,037 x 2
##    tailnum      n
##    <chr>    <dbl>
##  1 D942DN    3418
##  2 N0EGMQ  239143
##  3 N10156  109664
##  4 N102UW   25722
##  5 N103US   24619
##  6 N104UW   24616
##  7 N10575  139903
##  8 N105UW   23618
##  9 N107US   21677
## 10 N108UW   32070
## # ... with 4,027 more rows
```

QUESTION 4:

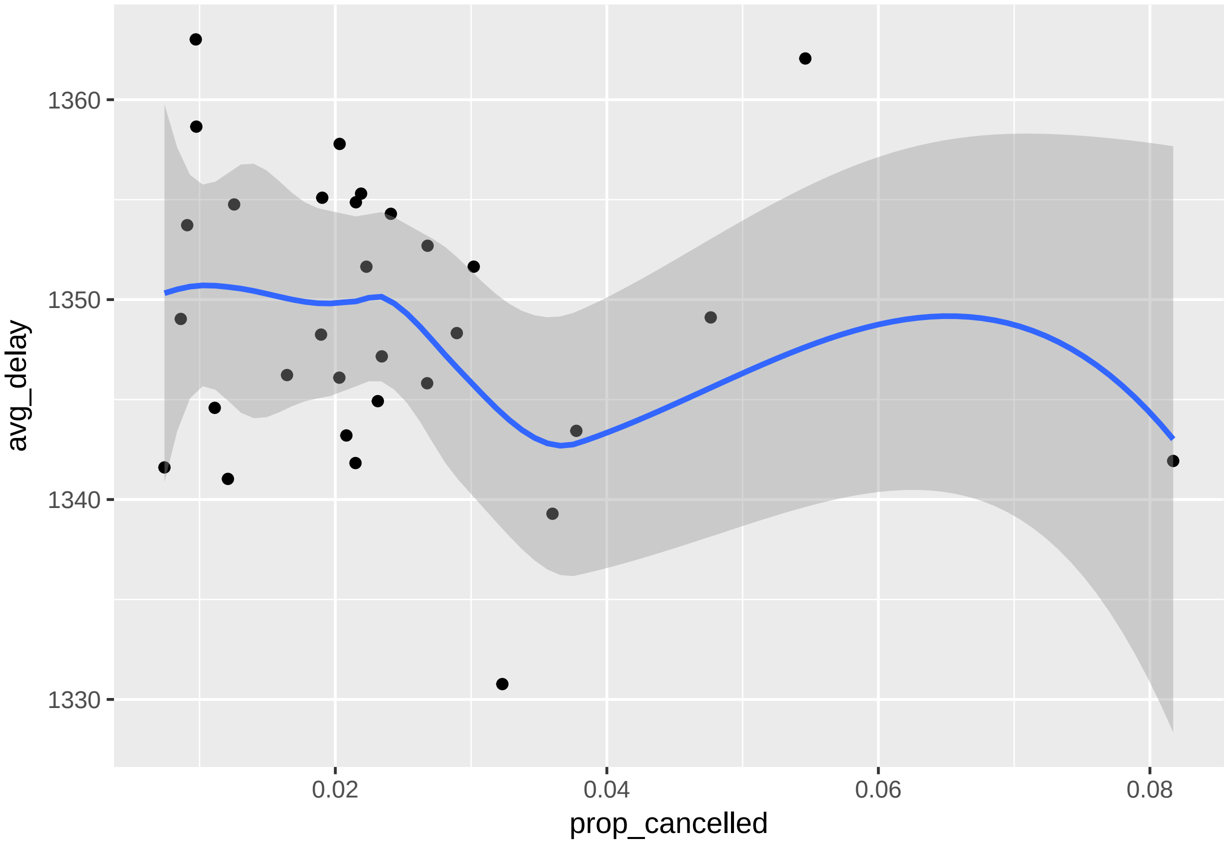
```
# number of cancelled flights per day
(cancelled_flights <- flights %>%
  filter(is.na(dep_time)) %>%
  count(day))
```

```
## # A tibble: 31 x 2
##      day     n
##    <int> <int>
##  1     1   246
##  2     2   250
##  3     3   109
##  4     4    82
##  5     5   226
##  6     6   296
##  7     7   318
##  8     8   921
##  9     9   593
## 10    10   535
## # ... with 21 more rows
```

```r
# proportion of day cancelled flights vs aerage delays
(cancelled_flights <- flights %>%
  group_by(day) %>%
  summarize(prop_cancelled = sum(is.na(dep_time)) / n(),
            avg_delay = mean(dep_time, na.rm = TRUE)))
```

```
## # A tibble: 31 x 3
##      day prop_cancelled avg_delay
##    <int>          <dbl>     <dbl>
##  1     1        0.0223      1352.
##  2     2        0.0231      1345.
##  3     3        0.00972     1363.
##  4     4        0.00741     1342.
##  5     5        0.0208      1343.
##  6     6        0.0268      1346.
##  7     7        0.0289      1348.
##  8     8        0.0817      1342.
##  9     9        0.0546      1362.
## 10    10        0.0477      1349.
## # ... with 21 more rows
```

```r
# plot for the relationship
ggplot(cancelled_flights, aes(x = prop_cancelled, y = avg_delay)) +
  geom_point() +
  geom_smooth()
```



QUESTION 6:

ANSWER 6:

sort argument will sort the elements of count () in the decending order. After the results are extracted, we can use the sort to arrange the values.

Section 10.5: #1, #2, #3 and #6 only.

QUESTION 2:

```r
# mtcars as a data frame
print(mtcars)
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
# mtcars as a tibble
print(as_tibble(mtcars))
```

```
## # A tibble: 32 x 11
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##    * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##    1  21       6  160     110  3.9   2.62  16.5     0     1     4     4
##    2  21       6  160     110  3.9   2.88  17.0     0     1     4     4
##    3  22.8     4  108      93  3.85  2.32  18.6     1     1     4     1
##    4  21.4     6  258     110  3.08  3.22  19.4     1     0     3     1
##    5  18.7     8  360     175  3.15  3.44  17.0     0     0     3     2
##    6  18.1     6  225     105  2.76  3.46  20.2     1     0     3     1
##    7  14.3     8  360     245  3.21  3.57  15.8     0     0     3     4
##    8  24.4     4  147.     62  3.69  3.19  20       1     0     4     2
##    9  22.8     4  141.     95  3.92  3.15  22.9     1     0     4     2
##   10  19.2     6  168.    123  3.92  3.44  18.3     1     0     4     4
## # ... with 22 more rows
```

When we print a dataframe whole records will be printed where as with the tibble only top rows will be printed and
the columns and widths are adjusted to fit the screen resolution.

QUESTION 2:

```
# with a data frame
df <- data_frame(abc = 1, xyz = 'a')
df
```

```
## # A tibble: 1 x 2
##     abc xyz
##   <dbl> <chr>
## 1     1 a
```

```
df$x
```

```
## NULL
```

```
df[, "xyz"]
```

```
## # A tibble: 1 x 1
##   xyz
##   <chr>
## 1 a
```

```
df[, c("abc", "xyz")]
```

```
## # A tibble: 1 x 2
##     abc xyz
##   <dbl> <chr>
## 1     1 a
```

```
# with a tibble
df <- tibble(abc = 1, xyz = "a")
df
```

```
## # A tibble: 1 x 2
##     abc xyz
##   <dbl> <chr>
## 1     1 a
```

```
df$x
```

```
## NULL
```

```
df[, "xyz"]
```

```
## # A tibble: 1 x 1
##   xyz
##   <chr>
## 1 a
```

```
df[, c("abc", "xyz")]
```

```
## # A tibble: 1 x 2
##     abc xyz
##   <dbl> <chr>
## 1     1 a
```

By using $ we can only extract by names. tibbles are much strictier and they alert us when we are trying to pull a column which never existed.

QUESTION 3:

```
var <- "mpg"
```

ANSWER:

We can extract the refernce variable by uisng double square brackets like df[[var]]

QUESTION 6:

```
?tibble
```

```
## Help on topic 'tibble' was found in the following packages:
##
##   Package               Library
##   tibble                /Library/Frameworks/R.framework/Versions/3.5/Resources/library
##   dplyr                 /Library/Frameworks/R.framework/Versions/3.5/Resources/library
##
##
## Using the first match ...
```

ANSWER:

```
getOption("tibble.max_extra_cols")
```

```
## NULL
```

Section 12.3.3: #2, #3 and #4 only.

QUESTION 2:

```
# table4a %>%
  # gather(1999, 2000, key = "year", value = "cases") # commented out as it is failing and need to ask
```

ANSWER:

we need to add the quotes around 1999 & 2000 otherwise 1999 & 2000 are treated as columns in the data frame. Below is the modified code.

```
table4a %>%
  gather('1999', '2000', key = "year", value = "cases")
```

```
## # A tibble: 6 x 3
##   country     year   cases
##   <chr>       <chr>  <int>
## 1 Afghanistan 1999     745
## 2 Brazil      1999   37737
## 3 China       1999  212258
## 4 Afghanistan 2000    2666
## 5 Brazil      2000   80488
## 6 China       2000  213766
```

QUESTION 3:

```
# given code
 people <- tribble(
   ~name,             ~key,    ~value,
  #-----------------|--------|------
  "Phillip Woods",   "age",       45,
  "Phillip Woods",   "height",   186,
  "Phillip Woods",   "age",       50,
  "Jessica Cordero", "age",       37,
  "Jessica Cordero", "height",   156
)

# Spreading the above tibble
# spread(people, key, value) # commented out as it is failing and need to ask instructor
```

ANSWER:

Duplicates are identified. In otherwords Phillip has two age records. In this case we can add another column related to the row identifier like the people_id then the issue will be resolved. Below is the code for the same.

```
people$people_id <- c(1, 1, 2, 1, 1)
people # display the modified tibble
```

```
## # A tibble: 5 x 4
##   name            key    value people_id
##   <chr>           <chr>  <dbl>     <dbl>
## 1 Phillip Woods   age       45         1
## 2 Phillip Woods   height   186         1
## 3 Phillip Woods   age       50         2
## 4 Jessica Cordero age       37         1
## 5 Jessica Cordero height   156         1
```

```
spread(people, key, value) # spread the tibble
```

```
## # A tibble: 3 x 4
##   name            people_id   age height
##   <chr>               <dbl> <dbl>  <dbl>
## 1 Jessica Cordero         1    37    156
## 2 Phillip Woods           1    45    186
## 3 Phillip Woods           2    50     NA
```

QUESTION 4:

```
preg <- tribble(
  ~pregnant, ~male, ~female,
  "yes",     NA,    10,
  "no",      20,    12
)
```

ANSWER:

By lokking at above tibble, we can observe that there is a missing value. So we can use gather() and add na.rm = TRUE to remove the value NA. Below is the gathered tibble with variables, pregnant, gender, count.

```
preg %>%
  gather(key = 'gender', value = 'value', c(2:3), na.rm = TRUE)
```

```
## # A tibble: 3 x 3
##   pregnant gender value
## * <chr>    <chr>  <dbl>
## 1 no       male      20
## 2 yes      female    10
## 3 no       female    12
```