Homework 4

Akshay Kumar Varanasi (av32826)

This homework is due on Feb. 19, 2019 at 4:00pm. Please submit as a PDF file on Canvas.

Problem 1: (4 pts) The following two data tables contain information about how many male and female passengers traveling on the Titanic in different classes survived or died. The data-frame `survived` contains information about passengers that survived, and the data-frame `deceased` contains information about passendgers that died. Using the `dplyr` and `tidyr` packages, make these data-frames tidy and then combine them into a single data-frame. Make sure that your final data-frame has a `survival_status` column indicating which data-frame the observations originally came from. HINT: You can use the `bind_rows` function to add rows from one data-frame onto another as long as both data-frames have identical column names.

```
survived <- read.table(text = "
class  male female
1st    57    140
2nd    14     80
3rd    75     76
crew  192     20
", head = T)

deceased <- read.table(text = "
class  male female
1st   118      4
2nd   154     13
3rd   387     89
crew  670      3
", head = T)


survived <- mutate(survived,survival_status = "survived" )
survived_gather<-gather(survived,Sex,Num_passengers,male:female)
head(survived_gather)
```

```
##   class survival_status    Sex Num_passengers
## 1   1st        survived   male             57
## 2   2nd        survived   male             14
## 3   3rd        survived   male             75
## 4  crew        survived   male            192
## 5   1st        survived female            140
## 6   2nd        survived female             80
```

```
#tail(survive_gather)


deceased <- mutate(deceased,survival_status = "deceased" )
deceased_gather<-gather(deceased,Sex,Num_passengers,male:female)
head(deceased_gather)
```

```
##   class survival_status    Sex Num_passengers
## 1   1st        deceased   male            118
## 2   2nd        deceased   male            154
## 3   3rd        deceased   male            387
## 4  crew        deceased   male            670
## 5   1st        deceased female              4
## 6   2nd        deceased female             13
```

```
#tail(deceased_gather)

passengers <- bind_rows(survived_gather, deceased_gather)
head(passengers)
```

```
##   class survival_status    Sex Num_passengers
## 1   1st        survived   male             57
## 2   2nd        survived   male             14
## 3   3rd        survived   male             75
## 4  crew        survived   male            192
## 5   1st        survived female            140
## 6   2nd        survived female             80
```

```
tail(passengers)
```

```
##    class survival_status    Sex Num_passengers
## 11   3rd        deceased   male            387
## 12  crew        deceased   male            670
## 13   1st        deceased female              4
## 14   2nd        deceased female             13
## 15   3rd        deceased female             89
## 16  crew        deceased female              3
```

Using the data-frame you created above, compute the total number of passengers that survived and that did not survive.

```
passengers %>%group_by(survival_status) %>% summarize(Total_num_passengers = su
m(Num_passengers))
```

```
## # A tibble: 2 x 2
##   survival_status Total_num_passengers
##   <chr>                          <int>
## 1 deceased                        1438
## 2 survived                         654
```

Total number of passengers survived are 654 and deceased are 1438.

Problem 2: (3 pts) The `chickwts` dataset contains information on the weight of chicks after being fed different feed supplements. The different feed supplements are labeled casein, horsebean, linseed, meatmeal, soybean, and sunflower in the `feed` column. I have created a new data-frame (`feed_names`), that contains the abbreviated names of different feed supplements. Using one of the `dplyr` join functions, combine the two data-frames so that there is an additional `feed_abbr` column and all of the original columns and rows in chickwts are retained. Which join function is most appropriate to use and why?

```
head(chickwts)
```

```
##   weight      feed
## 1    179 horsebean
## 2    160 horsebean
## 3    136 horsebean
## 4    227 horsebean
## 5    217 horsebean
## 6    168 horsebean
```

```
feed_names <- read.table(text = "
feed feed_abbr
casein cs
whey wh
linseed  ls
meatmeal mm
fishmeal fm
soybean sb
sunflower sf
corn co
wheatbran wb
", head = T)

chickwts_added <- left_join(chickwts, feed_names)
```

```
## Joining, by = "feed"
```

```
## Warning: Column `feed` joining factors with different levels, coercing to
## character vector
```

```
chickwts_added
```

```
##    weight      feed feed_abbr
## 1     179 horsebean      <NA>
## 2     160 horsebean      <NA>
## 3     136 horsebean      <NA>
## 4     227 horsebean      <NA>
## 5     217 horsebean      <NA>
## 6     168 horsebean      <NA>
## 7     108 horsebean      <NA>
## 8     124 horsebean      <NA>
## 9     143 horsebean      <NA>
## 10    140 horsebean      <NA>
## 11    309   linseed        ls
## 12    229   linseed        ls
## 13    181   linseed        ls
## 14    141   linseed        ls
## 15    260   linseed        ls
## 16    203   linseed        ls
## 17    148   linseed        ls
## 18    169   linseed        ls
## 19    213   linseed        ls
## 20    257   linseed        ls
## 21    244   linseed        ls
## 22    271   linseed        ls
## 23    243   soybean        sb
## 24    230   soybean        sb
## 25    248   soybean        sb
## 26    327   soybean        sb
## 27    329   soybean        sb
## 28    250   soybean        sb
## 29    193   soybean        sb
## 30    271   soybean        sb
## 31    316   soybean        sb
## 32    267   soybean        sb
## 33    199   soybean        sb
## 34    171   soybean        sb
## 35    158   soybean        sb
## 36    248   soybean        sb
## 37    423 sunflower        sf
## 38    340 sunflower        sf
## 39    392 sunflower        sf
## 40    339 sunflower        sf
## 41    341 sunflower        sf
## 42    226 sunflower        sf
## 43    320 sunflower        sf
## 44    295 sunflower        sf
## 45    334 sunflower        sf
```

```
## 46    322 sunflower        sf
## 47    297 sunflower        sf
## 48    318 sunflower        sf
## 49    325  meatmeal        mm
## 50    257  meatmeal        mm
## 51    303  meatmeal        mm
## 52    315  meatmeal        mm
## 53    380  meatmeal        mm
## 54    153  meatmeal        mm
## 55    263  meatmeal        mm
## 56    242  meatmeal        mm
## 57    206  meatmeal        mm
## 58    344  meatmeal        mm
## 59    258  meatmeal        mm
## 60    368    casein        cs
## 61    390    casein        cs
## 62    379    casein        cs
## 63    260    casein        cs
## 64    404    casein        cs
## 65    318    casein        cs
## 66    352    casein        cs
## 67    359    casein        cs
## 68    216    casein        cs
## 69    222    casein        cs
## 70    283    casein        cs
## 71    332    casein        cs
```

Left join is the most appropriate join function as we want all the rows and columns from chickwts and not necessarily from feed_names as we only want feed abbr column from feed_names. By making chickwts left table we retain all the rows and columns of it by using left join.

Problem 3: (3 pts) Recall the `flights` dataset from lab 3 worksheet. Ask a **conceptual** question about the `flights` dataset. Your question should not repeat the questions from class materials. Describe in 1-2 sentences how you would answer this question with an analysis or a graph.

```
library(nycflights13)
head(flights)
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>
## 1  2013     1     1      517            515         2      830
## 2  2013     1     1      533            529         4      850
## 3  2013     1     1      542            540         2      923
## 4  2013     1     1      544            545        -1     1004
## 5  2013     1     1      554            600        -6      812
## 6  2013     1     1      554            558        -4      740
## # … with 12 more variables: sched_arr_time <int>, arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>
```

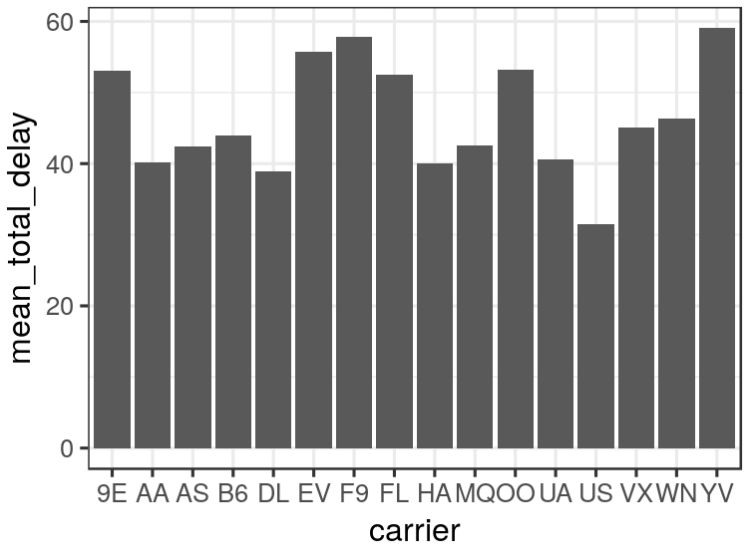
Conceptual Question: Which carrier is the best and which one is the worst?

Ans: One way to go about this question is to see if the flights are running on time, for that calculate the total delay by adding both arrival and departure delays. (Question arises whether we should take absolute value of delay or not). Then we need to group by carriers and finding its mean. The carrier which has lowest total mean delay is the best carrier in terms of running on time. In this case, I am taking absolute value because even though it is arriving or departing early that is not good.

```
flights %>%
  mutate(delay = abs(arr_delay) + abs(dep_delay)) ->flights

flights %>%
  group_by(carrier) %>%
  summarize(mean_total_delay = mean(delay, na.rm = TRUE)) -> flights_delay
#%>% arrange(mean_total_delay) # Either we can use this or plot to find out

flights_delay
```

```
## # A tibble: 16 x 2
##    carrier mean_total_delay
##    <chr>              <dbl>
##  1 9E                  53.1
##  2 AA                  40.2
##  3 AS                  42.4
##  4 B6                  43.9
##  5 DL                  38.9
##  6 EV                  55.7
##  7 F9                  57.8
##  8 FL                  52.5
##  9 HA                  40.0
## 10 MQ                  42.6
## 11 OO                  53.3
## 12 UA                  40.6
## 13 US                  31.4
## 14 VX                  45.1
## 15 WN                  46.3
## 16 YV                  59.1
```

```
 # Not a good way to visualize as x axis is discrete and categorical so this pl
ot cannot be used
#ggplot(flights_delay, aes(x = carrier , y = mean_total_delay)) + geom_point()

# Better way to visualize is bar plot as
ggplot(flights_delay, aes(x = carrier , y = mean_total_delay)) +
       geom_bar(stat="identity")
```



From the above plot we can clearly see that “US” carrier is the best in terms of punctuality as the mean delay is low. But for knowing which one is the worst, it is better to do a t-test between “YV” and “F9” as

their means are very close.

```
yv = flights$delay[flights$carrier == 'YV']
f9 = flights$delay[flights$carrier == 'F9']
t.test(yv,f9)
```

```
##
##  Welch Two Sample t-test
##
## data:  yv and f9
## t = 0.23106, df = 1223, p-value = 0.8173
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -9.906479 12.551390
## sample estimates:
## mean of x mean of y
##  59.09191  57.76946
```

We know from t-test that their means are not same and “YV” carrier is worst in terms of punctuality. We can also find similar results using box-plot.

Another criteria for the best or worst carrier could be expected travel time vs actual travel time, seeing which carrier generally reaches faster than expected time.