# Data 607 Week 5 Assignment

Jean Jimenez

2023-09-27

## Week 5 Assignment

### Data Cleanup Work

First, I imported the necessary packages to use for this assignment. I had created and saved an 'airlines.csv' in the current working directory with the data from the homework. Then, I imported the data set of airlines data. For sake of reproducibility, the csv was uploaded to github and this rmd will get the csv from there.

After importing the data, I took a look at it. I wanted to separate the data by airlines then by status (delayed vs. on time). To do this, first I replaced empty strings with NA. Then, I piped my airlines data through the fill function, which is from tidyr. The fill value filled in the missing values in the X column (airline name). I specified the direction down so that the values are filled going down. Afterwards, I passed that through rename(). rename from dplyr changed the name of column X to Airline and column X.1 to Status. Finally, I passed my data through filter function of dplyr to take out rows where status is empty.

```
library(readr)
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(tidyr)

airlines_dat=read.csv(url('https://raw.githubusercontent.com/sleepysloth12/data607_wk5/main/airlines.csv

airlines_dat
```

```
##         X     X.1 Los.Angeles Phoenix San.Diego San.Francisco Seattle
## 1  ALASKA on time         497     221       212           503    1841
## 2         delayed          62      12        20           102     305
## 3                          NA      NA        NA            NA      NA
## 4 AM WEST on time         694    4840       383           320     201
## 5         delayed         117     415        65           129      61
```

```r
airlines_dat$X[airlines_dat$X==""]=NA

clean_data= airlines_dat %>%

  fill(X, .direction='down') %>%

  rename(Airline = X, Status = X.1) %>%

  filter(Status != "")

clean_data
```

```
##   Airline  Status Los.Angeles Phoenix San.Diego San.Francisco Seattle
## 1  ALASKA on time         497     221       212           503    1841
## 2  ALASKA delayed          62      12        20           102     305
## 3 AM WEST on time         694    4840       383           320     201
## 4 AM WEST delayed         117     415        65           129      61
```

There are too many cities. I just made one city column where I placed all of the city's name. To do this, I piped my clean data above through the gather function from tidyr. The gather function converts from wide to long format. It takes a new key name and makes that into a column. In this case, it creates a city column and a count column (to consolidate the counts in one column). By writing -Airlines and -Status, those two columns are left as is.

```r
clean_data = clean_data %>%

  gather(key = "City", value = "Count", -Airline, -Status)

clean_data
```

```
##    Airline  Status          City Count
## 1   ALASKA on time   Los.Angeles   497
## 2   ALASKA delayed   Los.Angeles    62
## 3  AM WEST on time   Los.Angeles   694
## 4  AM WEST delayed   Los.Angeles   117
## 5   ALASKA on time       Phoenix   221
## 6   ALASKA delayed       Phoenix    12
## 7  AM WEST on time       Phoenix  4840
## 8  AM WEST delayed       Phoenix   415
## 9   ALASKA on time     San.Diego   212
## 10  ALASKA delayed     San.Diego    20
## 11 AM WEST on time     San.Diego   383
## 12 AM WEST delayed     San.Diego    65
## 13  ALASKA on time San.Francisco   503
## 14  ALASKA delayed San.Francisco   102
## 15 AM WEST on time San.Francisco   320
## 16 AM WEST delayed San.Francisco   129
## 17  ALASKA on time       Seattle  1841
## 18  ALASKA delayed       Seattle   305
## 19 AM WEST on time       Seattle   201
## 20 AM WEST delayed       Seattle    61
```

Finally, the last step I took in cleaning this data was removing empty columns. I did this by using the filter function again.

```
clean_data = clean_data %>%
  filter(!is.na(Count))

clean_data
```

```
##    Airline  Status          City Count
## 1   ALASKA on time   Los.Angeles   497
## 2   ALASKA delayed   Los.Angeles    62
## 3  AM WEST on time   Los.Angeles   694
## 4  AM WEST delayed   Los.Angeles   117
## 5   ALASKA on time       Phoenix   221
## 6   ALASKA delayed       Phoenix    12
## 7  AM WEST on time       Phoenix  4840
## 8  AM WEST delayed       Phoenix   415
## 9   ALASKA on time     San.Diego   212
## 10  ALASKA delayed     San.Diego    20
## 11 AM WEST on time     San.Diego   383
## 12 AM WEST delayed     San.Diego    65
## 13  ALASKA on time San.Francisco   503
## 14  ALASKA delayed San.Francisco   102
## 15 AM WEST on time San.Francisco   320
## 16 AM WEST delayed San.Francisco   129
## 17  ALASKA on time       Seattle  1841
## 18  ALASKA delayed       Seattle   305
## 19 AM WEST on time       Seattle   201
## 20 AM WEST delayed       Seattle    61
```

## Analysis

For the assignment, we were then asked to perform an analysis on our clean data set and compare the arrival delays between the two airlines.

To do this, used dplyr to get the summary statistics of the data we care about. First, I piped my clean data through filter() to only get the points where Status == delayed. Then, I passed it through group_by which grouped my data by Airline first, then City.

```
summary_stats = clean_data %>%

  filter(Status == "delayed") %>%

  group_by(Airline, City)

summary_stats
```

```
## # A tibble: 10 x 4
## # Groups:   Airline, City [10]
##    Airline Status  City          Count
##    <chr>   <chr>   <chr>         <int>
##  1 ALASKA  delayed Los.Angeles      62
##  2 AM WEST delayed Los.Angeles     117
```

```
## 3 ALASKA  delayed Phoenix          12
## 4 AM WEST delayed Phoenix         415
## 5 ALASKA  delayed San.Diego        20
## 6 AM WEST delayed San.Diego        65
## 7 ALASKA  delayed San.Francisco   102
## 8 AM WEST delayed San.Francisco   129
## 9 ALASKA  delayed Seattle         305
## 10 AM WEST delayed Seattle         61
```

After returning the count of each and comparing the cities and airlines, ALASKA airlines experience less delays than AM WEST. Specifically, ALASKA experienced less delays than AM WEST in 4 out of the 5 airports included in this data set.

ALASKA experiences more delays in Seattle when compared to AM West. If you want to go to Seattle and not experience delays, go with AM WEST.

The most noticeable difference in delays occurred at Phoenix Airport, where AM WEST experienced 415 delays while ALASKA only experienced 12. If you are traveling to Phoenix and don't want to be delayed, use ALASKA.

The least noticeable difference in delays occurred at San Francisco Airport. AM WEST had 129 delays and ALASKA had 102. It seems like San Francisco is just a busy airport.

Now, I want to get the proportion of delayed flights per city per airport. To do this, first I piped my data to get the total count of flights per airline per city. Then, I piped my clean_data again to get all the delay numbers per airline per city. Then, I conducted an inner join to add both together. Now we have a column for number of delays and a column for total number of flights by airline then city. Afterwards, I calculated the ratio of delay flights and added that as a column. Then, I converted those ratios to percent. Finally, I displayed a table of the percent of delayed Flights by city and then airline.

```
total_flights = clean_data %>%
  group_by(Airline, City) %>%
  summarize(Total_Count = sum(Count))
```

```
## `summarise()` has grouped output by 'Airline'. You can override using the
## `.groups` argument.
```

```
delayed_flights = clean_data %>%
  filter(Status == "delayed") %>%
  group_by(Airline, City) %>%
  summarize(Delayed_Count = sum(Count))
```

```
## `summarise()` has grouped output by 'Airline'. You can override using the
## `.groups` argument.
```

```
delayed_and_total = inner_join(total_flights, delayed_flights, by = c("Airline", "City"))

prop_delays=delayed_and_total %>%
  mutate(Delayed_Ratio = Delayed_Count / Total_Count)

prop_delays= prop_delays %>%
  select(Airline, City, Delayed_Ratio)%>%
  arrange(City)
```

```
prop_delays= prop_delays %>%
  mutate(Delayed_Percent=round(Delayed_Ratio*100, 1))

library(knitr)
prop_delays %>%
  select(Airline, City, Delayed_Percent) %>%
  kable(caption= "Percent of Delayed Flights by City and Airline")
```

Table 1: Percent of Delayed Flights by City and Airline

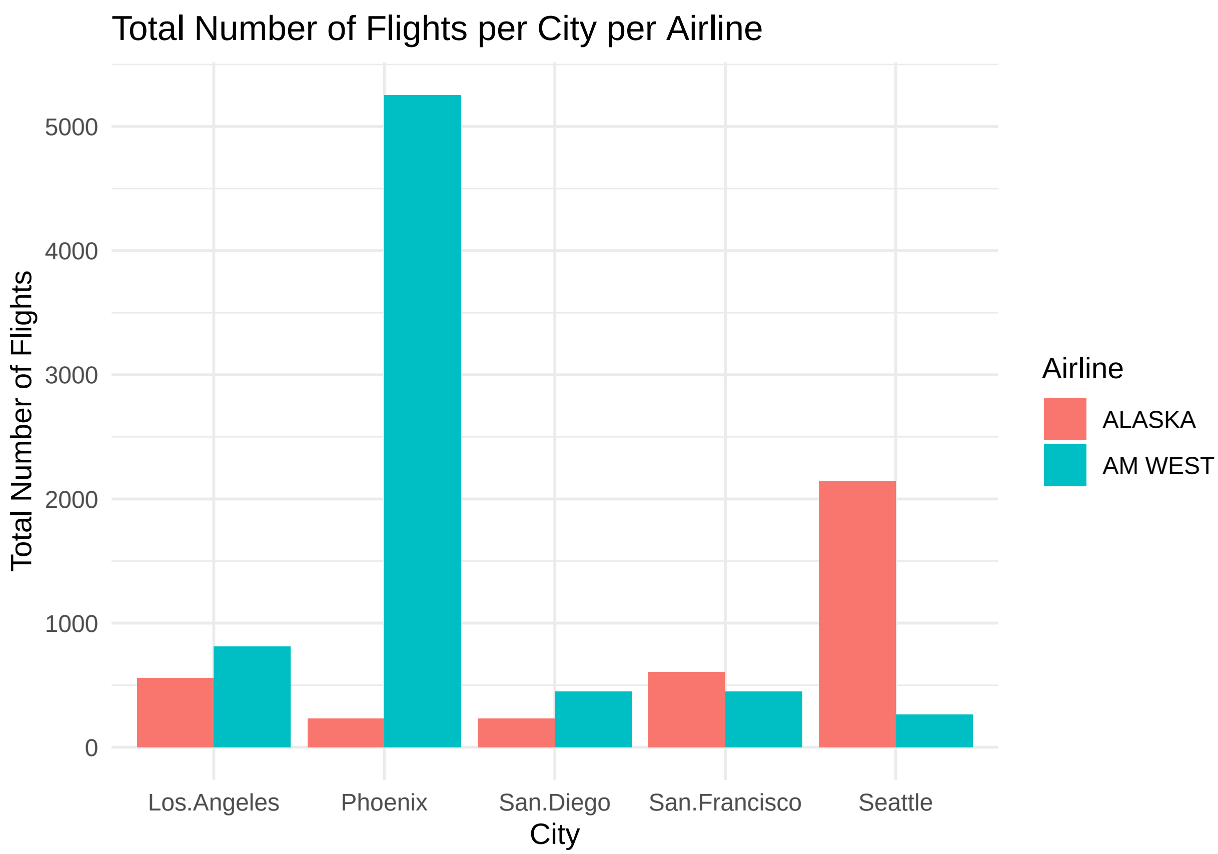
| Airline | City | Delayed_Percent |
|---|---|---:|
| ALASKA | Los.Angeles | 11.1 |
| AM WEST | Los.Angeles | 14.4 |
| ALASKA | Phoenix | 5.2 |
| AM WEST | Phoenix | 7.9 |
| ALASKA | San.Diego | 8.6 |
| AM WEST | San.Diego | 14.5 |
| ALASKA | San.Francisco | 16.9 |
| AM WEST | San.Francisco | 28.7 |
| ALASKA | Seattle | 14.2 |
| AM WEST | Seattle | 23.3 |

If we look at the ratio of delays to total flights per city per airline, we can see that the ratio of delayed flights is somewhat similar between the two airlines. 8% of ALASKA flights are delayed to San Diego while 14.5% of AM WEST flights are delayed to the same city. Likewise, in San Francisco, 16.9% of ALASKA flights were delayed and 28.7% of AM WEST flights were delayed to there.

Finally, I just wanted to include a graph so I plotted the total number of flights per by airline city using ggplot.

```
library(ggplot2)

ggplot(total_flights, aes(x=City, y=Total_Count, fill=Airline)) +
  geom_bar(stat="identity", position="dodge") +
  labs(title="Total Number of Flights per City per Airline",
       x="City",
       y="Total Number of Flights",
       fill="Airline") +
  theme_minimal()
```




As you can see from the visualization, Phoenix airport experiences WAY more flights from AM WEST then ALASKA. This might explain their slightly higher delay rate, as more flights mean more possibility for delays somewhere. Similarly, there is more ALASKA flights to Seattle when compared to AM WEST. Here though, ALASKA still experienced less delays.

## Conclusion

In conclusion, the Tidyr and dplyr packages are really useful when it comes to cleaning and tidying data that is in a complex/weird format. These packages make the job easier because instead of hardcoding some of the functions already included in the package you can just focus and spend more time on analyzing the data. In this homework assignment, I made a wide CSV with airlines data. Then, I cleaned up that wide CSV into a format that is usable by R. Finally, I used my clean data set to compare the delays between the two airlines. After comparing the delays between the two airlines, I calculated the proportion and then percent of delays organized by city then airlines. I then graphed that on a table. Afterwards, I created a plot that shows the density of total flights going to each city per airline.