Data 606 - Lab 2

Cameron Smith

2020-09-06

```
library(tidyverse)
library(openintro)
data(nycflights)
```

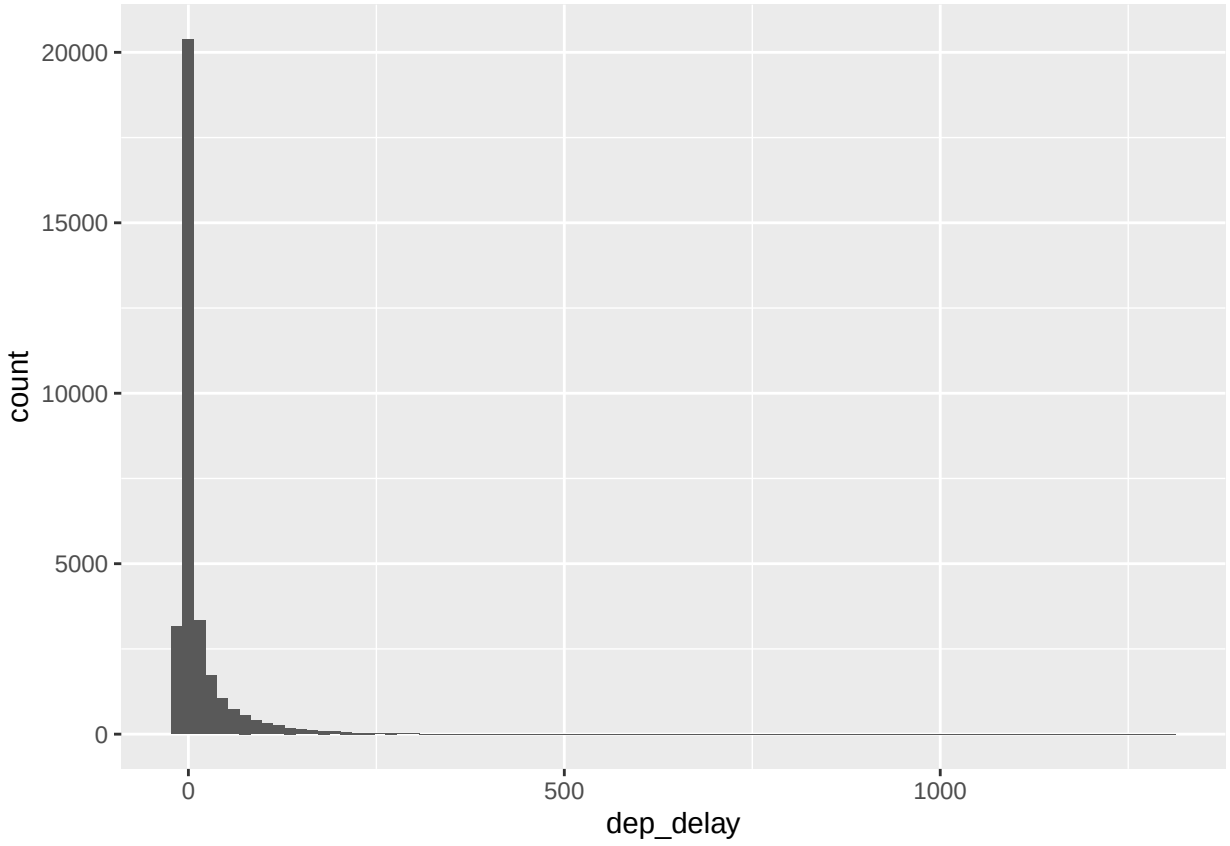
Exercise 1

The three histograms each show a different level of detail / granularity for the data. The most detailed of the options is the one using "binwidth=15", which reveals an interesting insight - the vast majority of flight delays are within the -15 to 15 minute range, with many flights actually arriving early (supported by the negative median of -1). This insight is obscured by the "binwidth=150" option.

```
lax_flights <- nycflights %>%
  filter(dest == "LAX")
lax_flights %>%
  summarise(mean_dd = mean(dep_delay),
            median_dd = median(dep_delay),
            n = n())
```

```
## # A tibble: 1 x 3
##   mean_dd median_dd     n
##     <dbl>     <dbl> <int>
## 1    9.78        -1  1583
```

```
ggplot(data = nycflights, aes(x = dep_delay)) +
  geom_histogram(binwidth=15)
```



Exercise 2

68 flights meet the criteria of (1) headed to SFO; and (2) in February.

```
# Create the data frame based on filtered values and then view key info
sfo_feb_arrivals <- nycflights %>% filter(dest == 'SFO', month == 2)
nrow(sfo_feb_arrivals)
```

```
## [1] 68
```

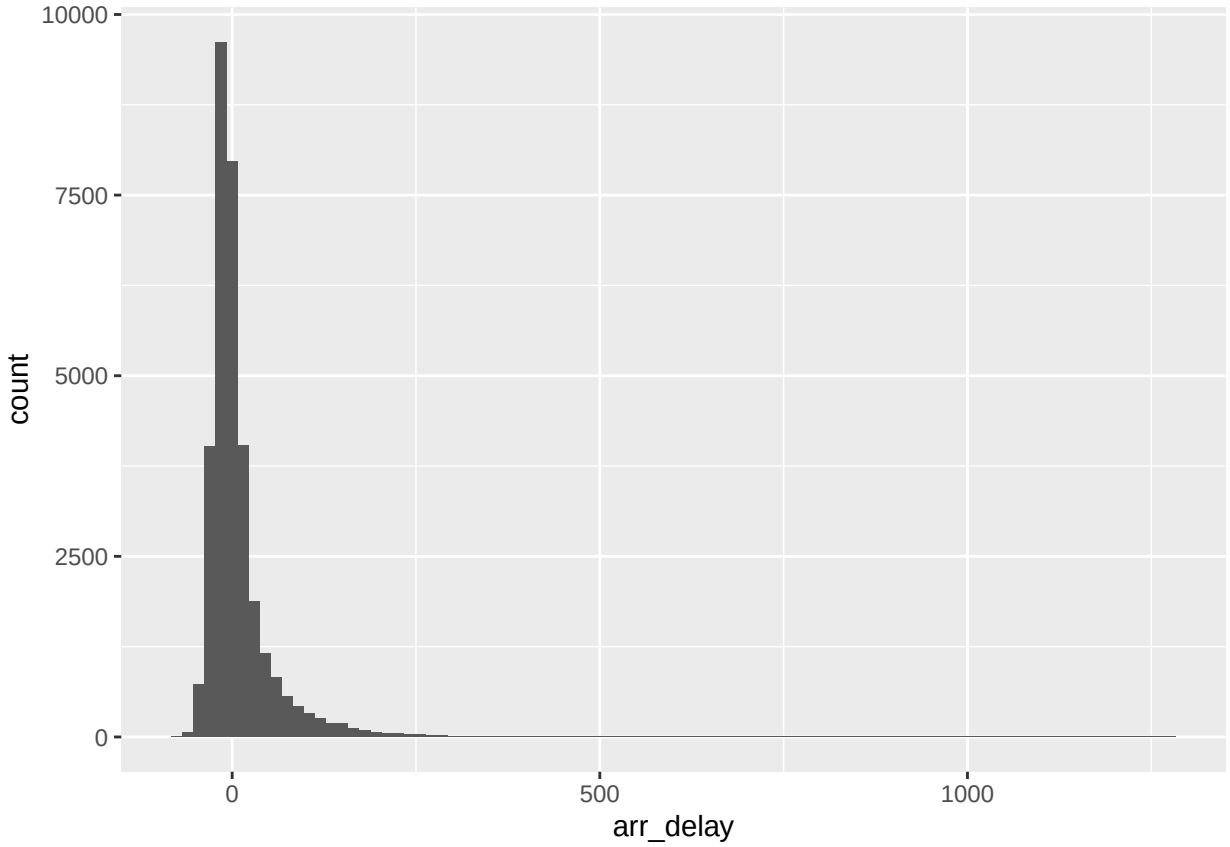
Exercise 3

Using a bincount of 15, the distribution of the arrival delays is bimodial and right skewed (values ranging from -66 to 196), with the majority of delays having negative values, i.e. the majority of arrivals were *early*. Many of the values fall near the mean and median of -4.5 and -11, respectively, and most fall within one standard deviation of the mean which is 36.28.

```
sfo_feb_arrivals %>%
  summarise(mean_sfo = mean(arr_delay),
            median_sfo = median(arr_delay),
            std_sfo = sd(arr_delay),
            min_sfo = min(arr_delay),
            max_sfo = max(arr_delay),
            n = n())
```

```
## # A tibble: 1 x 6
##   mean_sfo median_sfo std_sfo min_sfo max_sfo     n
##      <dbl>      <dbl>   <dbl>   <dbl>   <dbl> <int>
## 1     -4.5        -11    36.3     -66     196    68
```

```
ggplot(data = nycflights, aes(x = arr_delay)) +
  geom_histogram(binwidth=15)
```



Exercise 4

The carrier with the most variable arrival delays to SFO is United Airlines.

```
sfo_feb_arrivals %>%
  group_by(carrier) %>%
  summarise(median_sfo = median(arr_delay),
            iqr_sfo = IQR(arr_delay),
            min(arr_delay),
            max(arr_delay),
            sd(arr_delay),
            n = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 5 x 7
```

```
##   carrier median_sfo iqr_sfo `min(arr_delay)` `max(arr_delay)` `sd(arr_delay)`
##   <chr>        <dbl>   <dbl>            <dbl>            <dbl>           <dbl>
## 1 AA             5      17.5              -26               76            29.5
## 2 B6           -10.5    12.2              -18               11            11.0
## 3 DL           -15      22                -48               48            22.0
## 4 UA           -10      22                -35              196            48.3
## 5 VX           -22.5    21.2              -66               99            40.8
## # ... with 1 more variable: n <int>
```

Exercise 5

The pro of using the lowest mean to plan travel leaving NYC that minimizes potential departure delays is that it is a simple metric which takes into account all data points. The con is that it is affected by extreme observations (i.e. outliers), which in this case we do have. The median on the other hand can be a good metric to use because it is not affected by outliers. This is also the converse however as it is missing the full picture. In this case, using the data below, we can see that the media is the more helpul of the metrics due to the skewed distribution of the data:

```
nycflights %>%
  group_by(month) %>%
  summarise(mean_dd = mean(dep_delay), median_dd = median(dep_delay)) %>%
  arrange(desc(mean_dd))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 12 x 3
##    month mean_dd median_dd
##    <int>   <dbl>     <dbl>
##  1     7   20.8          0
##  2     6   20.4          0
##  3    12   17.4          1
##  4     4   14.6         -2
##  5     3   13.5         -1
##  6     5   13.3         -1
##  7     8   12.6         -1
##  8     2   10.7         -2
##  9     1   10.2         -2
## 10     9    6.87        -3
## 11    11    6.10        -2
## 12    10    5.88        -3
```

Exercise 6

The NYC airport I would fly out of is LGA as it has the highest on time departure rate of the NYC airports.

```
nycflights <- nycflights %>%
  mutate(dep_type = ifelse(dep_delay < 5, "on time", "delayed"))
nycflights %>%
  group_by(origin) %>%
  summarise(ot_dep_rate = sum(dep_type == "on time") / n()) %>%
  arrange(desc(ot_dep_rate))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 2
##   origin ot_dep_rate
##   <chr>        <dbl>
## 1 LGA          0.728
## 2 JFK          0.694
## 3 EWR          0.637
```

Exercise 7

The average speed variable can be added to the dataframe as follows:

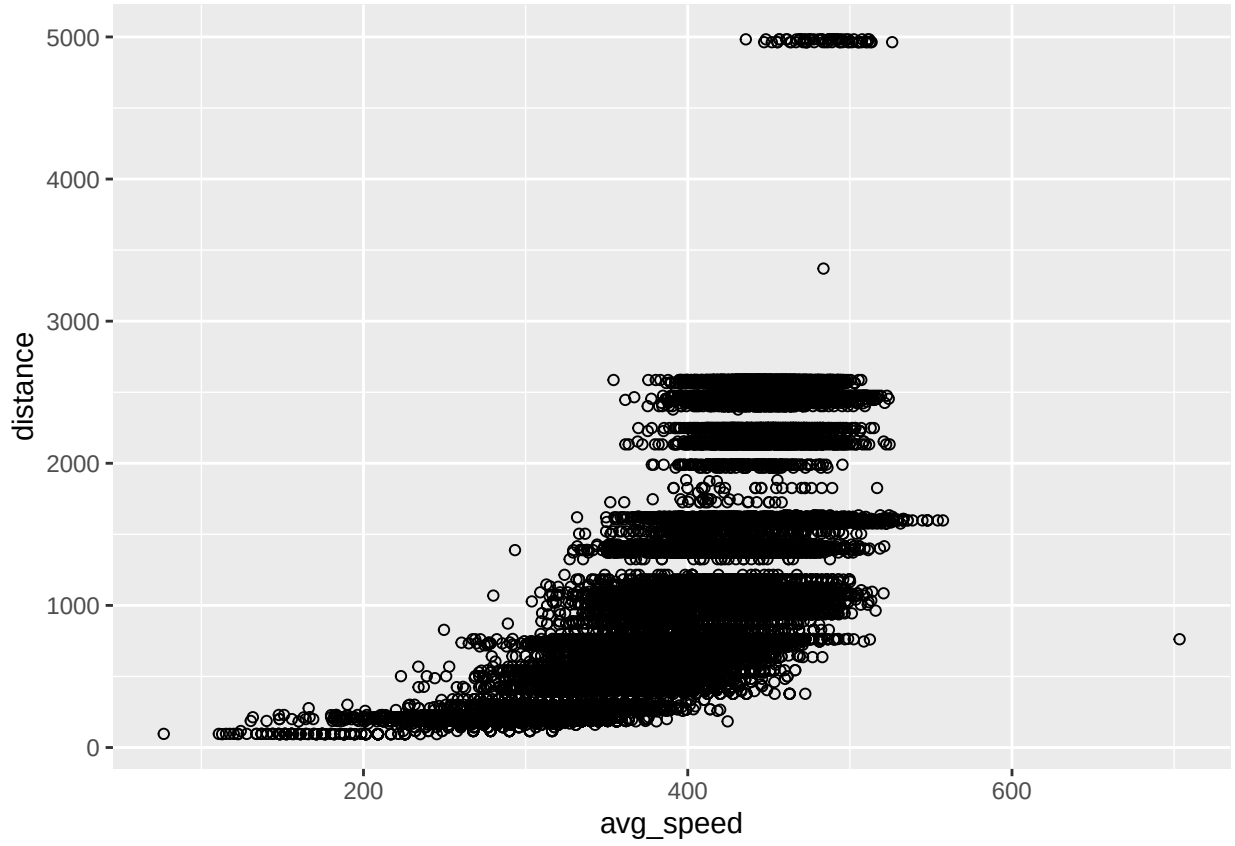
```
nycflights <- nycflights %>%
  mutate(avg_speed = distance / (air_time / 60))
# Verify transformation by viewing first few rows
head(select(nycflights, flight, avg_speed))
```

```
## # A tibble: 6 x 2
##   flight avg_speed
##    <int>     <dbl>
## 1    407      474.
## 2    329      444.
## 3    422      395.
## 4   2391      447.
## 5   3652      355.
## 6    353      319.
```

Exercise 8

There appears to be a positive non-linear relationship between the two variables, with increased distance corresponding to increased speed.

```
ggplot(nycflights, aes(x = avg_speed, y = distance)) +
  geom_point(shape=1)
```

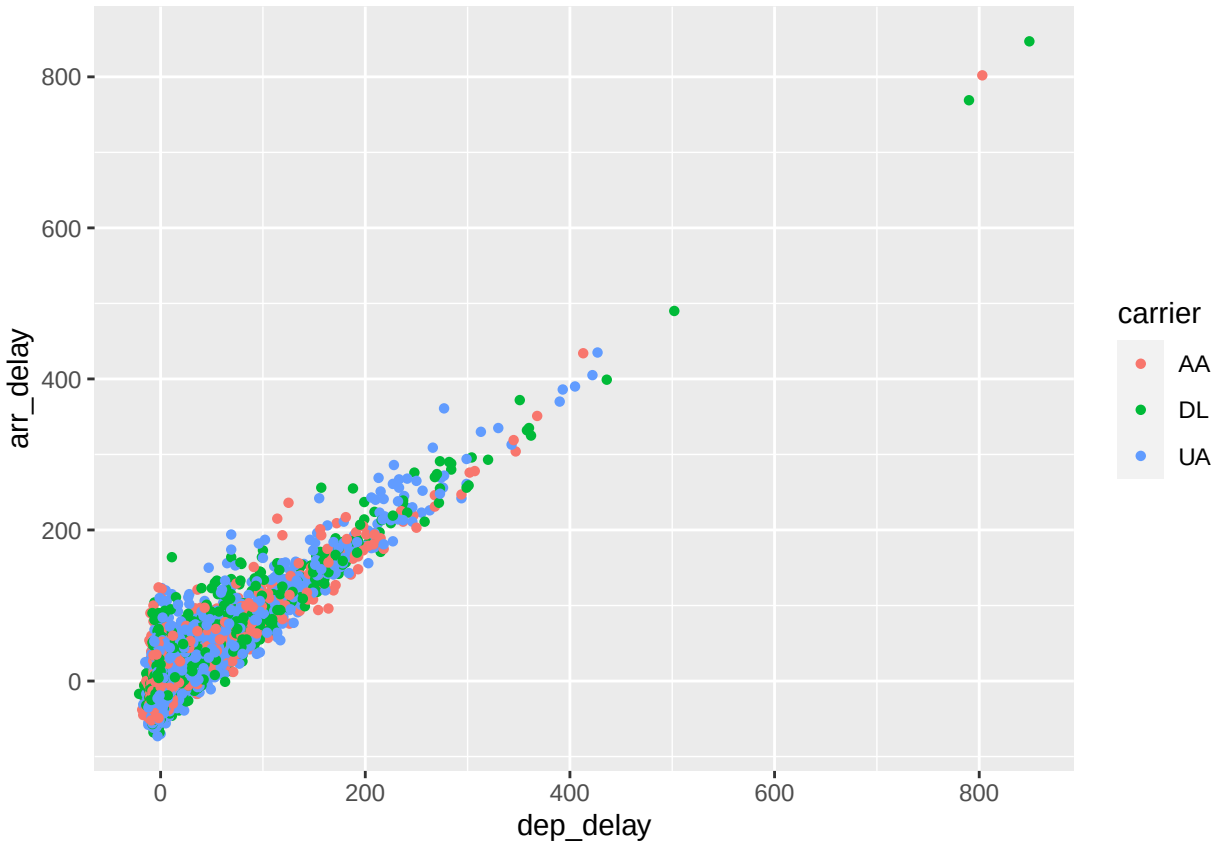


Exercise 9

Based on the below plot it appears that the (rough) cutoff point for departure delays where you can still expect to get to your destination on time is 45 minutes.

```
filtered_airlines <- nycflights %>%
  filter(carrier == "AA" | carrier == "DL" | carrier == "UA")

filtered_airlines %>% ggplot(aes(x = dep_delay, y = arr_delay, color = carrier)) +
  geom_point(shape=16)
```



End of lab report