Lab_09

Fantastic Four

2016 Colorado Rockies

Roster

Here is the roster for the 2016 Colorado Rockies:

```
names <- filter(Lahman::Master)
batting <- filter(Lahman::Batting, yearID == 2016 & teamID == "COL")
batting <- left_join(batting,names,"playerID")
batting <- select(batting,playerID,nameFirst,nameLast,birthDate,birthMonth,debut,finalGame,G,AB,R,HR,RBI
roster <- select(batting,nameFirst,nameLast,birthDate,debut,birthCity,birthState,birthCountry)
roster
```

```
##    nameFirst   nameLast  birthDate      debut           birthCity
## 1  Cristhian     Adames 1991-07-26 2014-07-29       Santo Domingo
## 2      Tyler   Anderson 1989-12-30 2016-06-12           Las Vegas
## 3      Nolan    Arenado 1991-04-16 2013-04-28       Newport Beach
## 4    Brandon     Barnes 1986-05-15 2012-08-07              Orange
## 5  Christian    Bergman 1988-05-04 2014-06-09            Glendale
## 6       Chad     Bettis 1989-04-26 2013-08-01             Lubbock
## 7    Charlie   Blackmon 1986-07-01 2011-06-07              Dallas
## 8      Eddie     Butler 1991-03-13 2014-06-06          Chesapeake
## 9       Matt   Carasiti 1991-07-23 2016-08-12         New Britain
## 10   Stephen   Cardullo 1987-08-31 2016-08-26           Hollywood
## 11    Miguel     Castro 1994-12-24 2015-04-06           La Romana
## 12     Tyler   Chatwood 1989-12-16 2011-04-11            Redlands
## 13     David       Dahl 1994-04-01 2016-07-25          Birmingham
## 14     Jorge De La Rosa 1981-04-05 2004-08-14           Monterrey
## 15    Daniel   Descalso 1986-10-19 2010-09-18        Redwood City
## 16    Carlos    Estevez 1992-12-28 2016-04-23       Santo Domingo
## 17     Yohan     Flande 1986-01-27 2014-06-25            El Seibo
## 18    Dustin    Garneau 1987-08-13 2015-08-20            Torrance
## 19  Gonzalez     Germen 1987-09-23 2013-07-12            Guaymate
## 20    Carlos   Gonzalez 1985-10-17 2008-05-30           Maracaibo
## 21       Jon       Gray 1991-11-05 2015-08-04             Shawnee
## 22     Jason      Gurka 1988-01-10 2015-08-29             Houston
## 23     David       Hale 1987-09-27 2013-09-13             Atlanta
## 24      Jeff    Hoffman 1993-01-08 2016-08-20              Latham
## 25      Nick    Hundley 1983-09-08 2008-07-04           Corvallis
## 26        DJ   LeMahieu 1988-07-13 2011-05-30             Visalia
## 27     Boone      Logan 1984-08-13 2006-04-04         San Antonio
## 28    Jordan      Lyles 1990-10-19 2011-05-31          Hartsville
## 29    German    Marquez 1995-02-22 2016-09-08           San Felix
## 30      Jake      McGee 1986-08-06 2010-09-14            San Jose
## 31    Justin     Miller 1987-06-13 2014-04-18         Bakersfield
## 32     Jason      Motte 1982-06-22 2008-09-03          Port Huron
## 33       Tom     Murphy 1991-04-03 2015-09-12         West Monroe
## 34     Scott      Oberg 1990-03-13 2015-04-14           Tewksbury
```

```
## 35     Adam   Ottavino 1985-11-22 2010-05-29             New York
## 36  Gerardo      Parra 1987-05-06 2009-05-13        Santa Barbara
## 37   Jordan  Patterson 1992-02-12 2016-09-08               Mobile
## 38      Ben    Paulsen 1987-10-27 2014-05-22             Plymouth
## 39     Chad     Qualls 1978-08-17 2004-07-22               Lomita
## 40     Ryan     Raburn 1981-04-17 2004-09-12                Tampa
## 41     Mark   Reynolds 1983-08-03 2007-05-16            Pikeville
## 42    Chris      Rusin 1986-10-22 2012-08-21              Detroit
## 43   Trevor      Story 1992-11-15 2016-04-04               Irving
## 44   Raimel      Tapia 1994-02-04 2016-09-02 San Pedro de Macoris
## 45      Pat    Valaika 1992-09-09 2016-09-06             Valencia
## 46     Tony    Wolters 1992-06-09 2016-04-05                Vista
## 47   Rafael       Ynoa 1987-08-07 2014-09-01             Santiago
##           birthState birthCountry
## 1   Distrito Nacional         D.R.
## 2                  NV          USA
## 3                  CA          USA
## 4                  CA          USA
## 5                  CA          USA
## 6                  TX          USA
## 7                  TX          USA
## 8                  VA          USA
## 9                  CT          USA
## 10                 FL          USA
## 11          La Romana         D.R.
## 12                 CA          USA
## 13                 AL          USA
## 14        Nuevo Leon       Mexico
## 15                 CA          USA
## 16  Distrito Nacional         D.R.
## 17           El Seibo         D.R.
## 18                 CA          USA
## 19          La Romana         D.R.
## 20              Zulia    Venezuela
## 21                 OK          USA
## 22                 TX          USA
## 23                 GA          USA
## 24                 NY          USA
## 25                 OR          USA
## 26                 CA          USA
## 27                 TX          USA
## 28                 SC          USA
## 29            Bolivar    Venezuela
## 30                 CA          USA
## 31                 CA          USA
## 32                 MI          USA
## 33                 NY          USA
## 34                 MA          USA
## 35                 NY          USA
## 36              Zulia    Venezuela
## 37                 AL          USA
## 38                 WI          USA
## 39                 CA          USA
## 40                 FL          USA
```

```
## 41                    KY          USA
## 42                    MI          USA
## 43                    TX          USA
## 44 San Pedro de Macoris         D.R.
## 45                    CA          USA
## 46                    CA          USA
## 47             Santiago         D.R.
```

Who came the quickest to the majors?

```r
batting$howLong <- ymd(batting$debut)- ymd(batting$birthDate)
batting$howLong[order(batting$howLong)]
```

```
## Time differences in days
##  [1]  7408  7529  7786  7869  7904  8043  8048  8151  8246  8261  8356
## [12]  8404  8486  8517  8532  8541  8549  8625  8673  8687  8701  8735
## [23]  8763  8805  8863  8928  8954  8975  9066  9107  9152  9163  9424
## [34]  9435  9471  9483  9532  9570  9581  9661  9704  9806  9887 10093
## [45] 10234 10376 10588
```

```r
filter(batting, howLong == 7408)
```

```
##    playerID nameFirst nameLast  birthDate birthMonth      debut  finalGame
## 1 castrmi01    Miguel   Castro 1994-12-24         12 2015-04-06 2016-06-24
##    G AB R HR RBI SB SO birthCity birthState birthCountry   howLong
## 1 19  0 0  0   0  0  0 La Romana  La Romana         D.R. 7408 days
```

From this, we see that Miguel Castro was the quickest of the whole team to make it to the majors in 7408 days.

Which players on the team have five letter names?

```r
fiveletter <- str_subset(batting$nameFirst,"^(.....)$")
fiveletter
```

```
##  [1] "Tyler" "Nolan" "Eddie" "Tyler" "David" "Jorge" "Yohan" "Jason"
##  [9] "David" "Boone" "Jason" "Scott" "Chris"
```

Where are the players are from?

```r
batting$geo<-geocode(batting$birthCity,output="latlona",source="google")
```

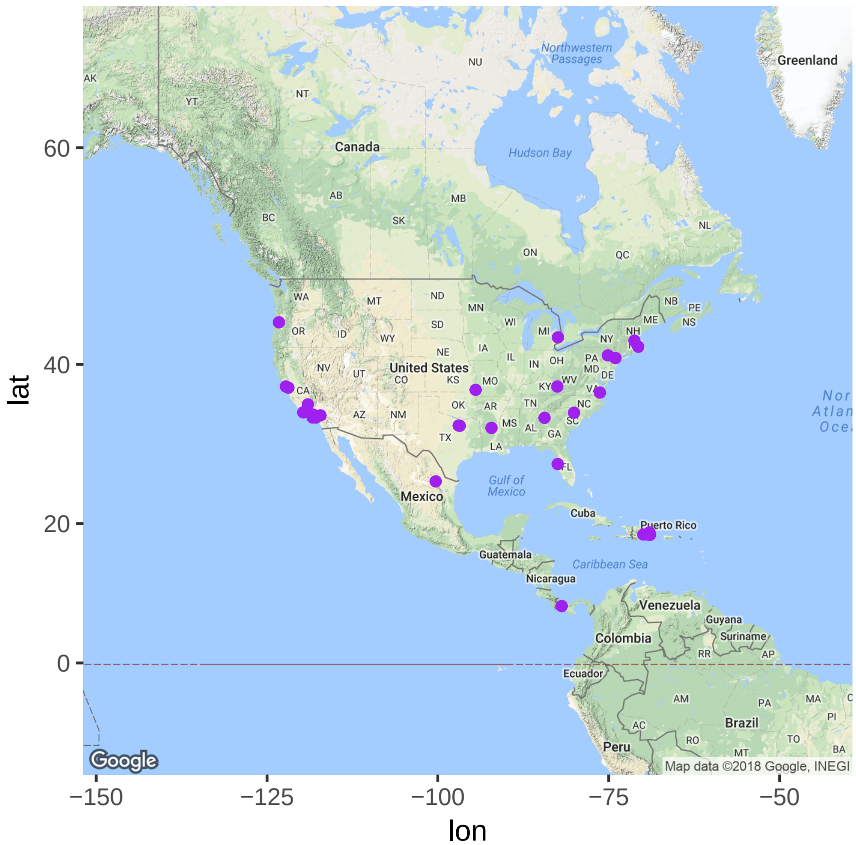
```
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Santo%20Domingo&sens

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Las%20Vegas&sensor=f

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Las
## Vegas"

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Newport%20Beach&sens

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Newport
## Beach"
```

```
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Orange&sensor=false

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Glendale&sensor=fals

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Lubbock&sensor=false

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Lubbock"

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Dallas&sensor=false

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Chesapeake&sensor=fal

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=New%20Britain&sensor=

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "New
## Britain"

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Hollywood&sensor=fals

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=La%20Romana&sensor=
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Redlands&sensor=fals
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Birmingham&sensor=fa

## Warning: geocode failed with status OVER_QUERY_LIMIT, location =
## "Birmingham"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Monterrey&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Redwood%20City&sens
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Santo%20Domingo&sens

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Santo
## Domingo"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=El%20Seibo&sensor=fa
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Torrance&sensor=fals
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Guaymate&sensor=false
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Maracaibo&sensor=fal

## Warning: geocode failed with status OVER_QUERY_LIMIT, location =
## "Maracaibo"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Shawnee&sensor=false
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Houston&sensor=false

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Houston"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Atlanta&sensor=false
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Latham&sensor=false

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Latham"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Corvallis&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Visalia&sensor=false

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Visalia"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=San%20Antonio&senso

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "San
## Antonio"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Hartsville&sensor=fa
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=San%20Felix&sensor=
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=San%20Jose&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Bakersfield&sensor=
```

```
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Port%20Huron&sensor
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=West%20Monroe&senso
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Tewksbury&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=New%20York&sensor=f
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Santa%20Barbara&sens
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Mobile&sensor=false
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Plymouth&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Lomita&sensor=false
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Tampa&sensor=false
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Pikeville&sensor=fa
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Detroit&sensor=fals

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Detroit"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Irving&sensor=false
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=San%20Pedro%20de%20M
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Valencia&sensor=fal
## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Vista&sensor=false

## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "Vista"

## .Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Santiago&sensor=fal
```

```
ggmap(get_map(location = 'USA',zoom = 3)) + geom_point(data=batting$geo,mapping=aes(x=lon,y=lat),color=
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=USA&zoom=3&size=640x640&scale=2&
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=USA&sensor=false

## Warning: Removed 15 rows containing missing values (geom_point).
```



Individual Findings

Lindsay Gettel

Section 13.5.1#3: Get the records for the most common vehicles from the fuel economy data set.

```
## # A tibble: 347 x 4
## # Groups:   make [?]
##    make  model                n years
##    <chr> <chr>            <int> <int>
##  1 Acura Integra             42    16
##  2 Acura Legend              28    10
##  3 Acura MDX 4WD             12    12
##  4 Acura NSX                 28    14
##  5 Acura TSX                 27    11
##  6 Audi  A4                  49    19
##  7 Audi  A4 Avant quattro    49    15
##  8 Audi  A4 quattro          66    19
##  9 Audi  A6                  20    19
## 10 Audi  A6 Avant quattro    12    12
## # ... with 337 more rows
```

```
## # A tibble: 33,442 x 12
##       id make  model  year class trans drive   cyl displ fuel    hwy   cty
##    <int> <chr> <chr> <int> <chr> <chr> <chr> <int> <dbl> <chr> <int> <int>
##  1 27550 AM G~ DJ P~  1984 Spec~ Auto~ 2-Wh~     4  2.50 Regu~    17    18
##  2 28426 AM G~ DJ P~  1984 Spec~ Auto~ 2-Wh~     4  2.50 Regu~    17    18
##  3 27549 AM G~ FJ8c~  1984 Spec~ Auto~ 2-Wh~     6  4.20 Regu~    13    13
##  4 28425 AM G~ FJ8c~  1984 Spec~ Auto~ 2-Wh~     6  4.20 Regu~    13    13
##  5  1032 AM G~ Post~  1985 Spec~ Auto~ Rear~     4  2.50 Regu~    17    16
##  6  1033 AM G~ Post~  1985 Spec~ Auto~ Rear~     6  4.20 Regu~    13    13
##  7  3347 ASC ~ GNX    1987 Mids~ Auto~ Rear~     6  3.80 Prem~    21    14
##  8 13309 Acura 2.2C~  1997 Subc~ Auto~ Fron~     4  2.20 Regu~    26    20
##  9 13310 Acura 2.2C~  1997 Subc~ Manu~ Fron~     4  2.20 Regu~    28    22
## 10 13311 Acura 2.2C~  1997 Subc~ Auto~ Fron~     6  3.00 Regu~    26    18
## # ... with 33,432 more rows
```

The top three cars, based on highest average miles per gallons of highway and city driving, are the Ford Ranger pickup, Honda Insight, and the Toyota Prius.

Section 14.4.5.1#3: Switch the first and last letters for every word in the words data and compare and see which are still common words.

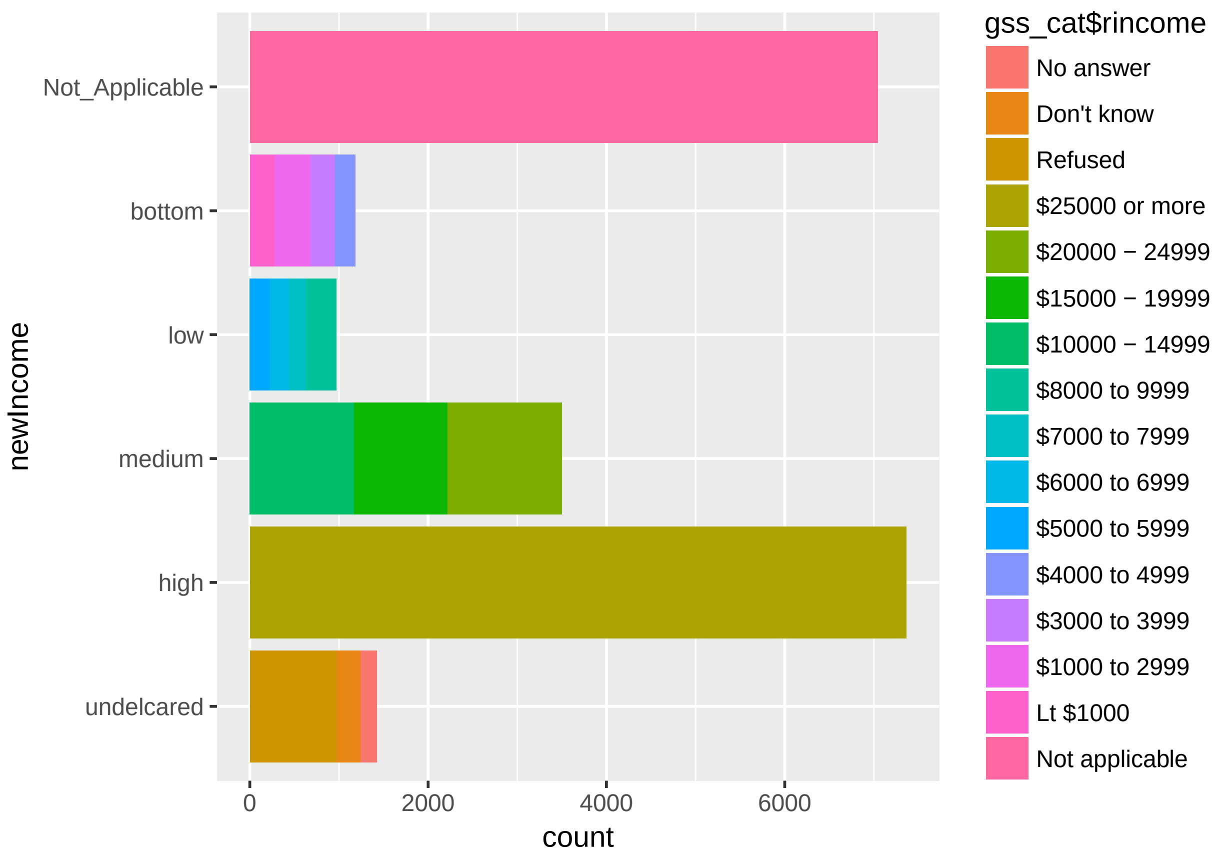
```
##  [1] "a"          "america"    "area"       "dad"        "dead"
##  [6] "depend"     "educate"    "else"       "encourage"  "engine"
## [11] "europe"     "evidence"   "example"    "excuse"     "exercise"
## [16] "expense"    "experience" "eye"        "health"     "high"
## [21] "knock"      "level"      "local"      "nation"     "non"
## [26] "rather"     "refer"      "remember"   "serious"    "stairs"
## [31] "test"       "tonight"    "transport"  "treat"      "trust"
## [36] "window"     "yesterday"
```

Above are the words which remain common words even after switching the first and last letters.

Section 15.3.1#1: Consider the distribution of the reported incomes from the General Social survey.

```
## # A tibble: 16 x 2
```

```
##    f                  n
##    <fct>          <int>
##  1 No answer        183
##  2 Don't know       267
##  3 Refused          975
##  4 $25000 or more  7363
##  5 $20000 - 24999  1283
##  6 $15000 - 19999  1048
##  7 $10000 - 14999  1168
##  8 $8000 to 9999    340
##  9 $7000 to 7999    188
## 10 $6000 to 6999    215
## 11 $5000 to 5999    227
## 12 $4000 to 4999    226
## 13 $3000 to 3999    276
## 14 $1000 to 2999    395
## 15 Lt $1000         286
## 16 Not applicable  7043
```



The graph above demonstrates how many people responded to having specific incomes in the survey.

Section 16.3.4#5: What day of the week is the best day to fly in order to minimize the amount of delays.

```
#16.3.4 Problem 5
make_datetime_100 <- function(year, month, day, time){
  make_datetime(year, month, day, time %/% 100, time %% 100)}
flights_dt <- flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  mutate(
```

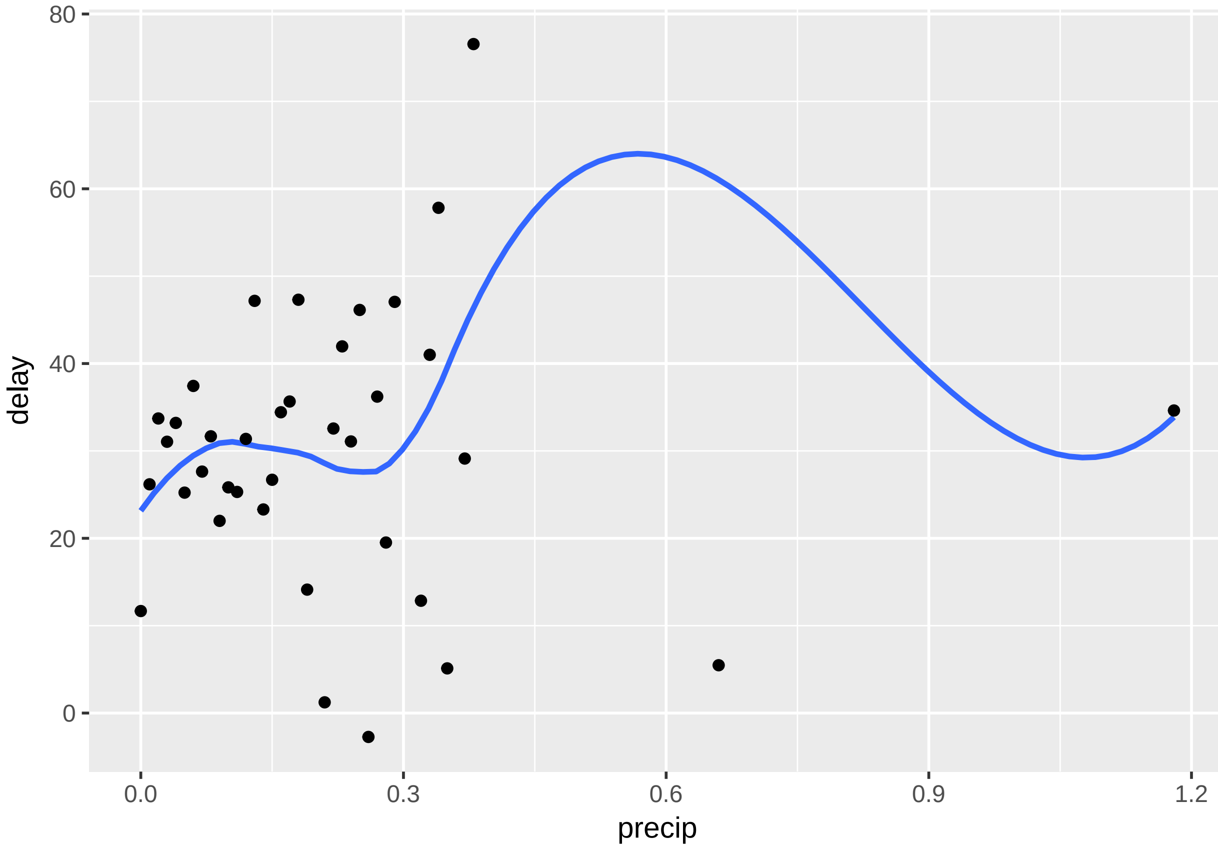
```
    dep_time = make_datetime_100(year, month, day, dep_time),
    arr_time = make_datetime_100(year, month, day, arr_time),
    sched_dep_time = make_datetime_100(year, month, day, sched_dep_time),
    sched_arr_time = make_datetime_100(year, month, day, sched_arr_time)
  ) %>%
  select(origin, dest, ends_with("delay"), ends_with("time"))
day_compare<-flights_dt%>%
  select(dep_delay, arr_delay, sched_dep_time, sched_arr_time)%>%
  mutate(day=wday(sched_dep_time))%>%
  group_by(day)%>%
  summarise(avg_dep_delay=mean(dep_delay), avg_arr_delay=mean(arr_delay, na.rm=TRUE))
print(day_compare)
```

```
## # A tibble: 7 x 3
##     day avg_dep_delay avg_arr_delay
##   <dbl>         <dbl>         <dbl>
## 1  1.00         11.5           4.82
## 2  2.00         14.7           9.65
## 3  3.00         10.6           5.39
## 4  4.00         11.7           7.05
## 5  5.00         16.1          11.7
## 6  6.00         14.7           9.07
## 7  7.00          7.62        - 1.45
```

The best day to fly with minimal delays is Sunday, it has the lowest arrival and the lowest departure delays.

Lexie Marinelli

```
## `geom_smooth()` using method = 'loess'
```

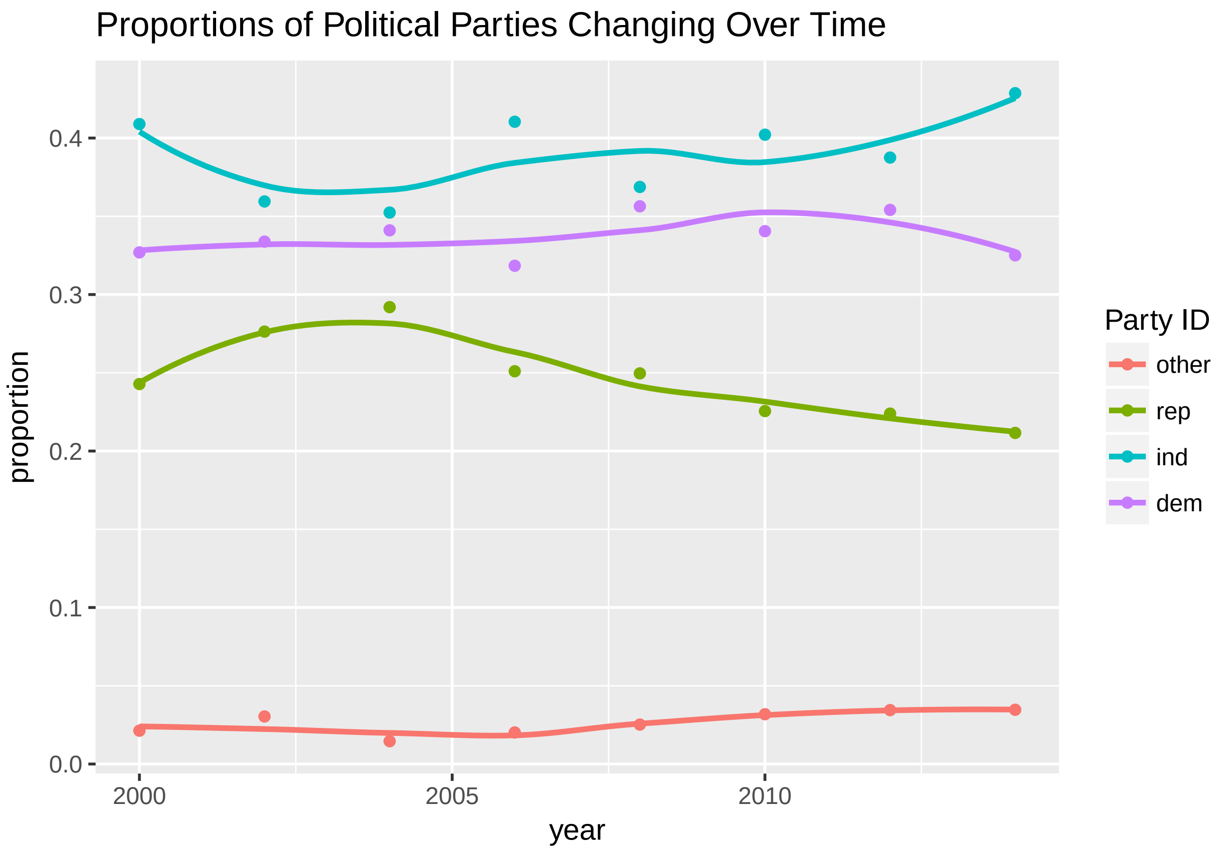


You would think there would be a positive correlation but after 0.2 the trend doesn't show any relation between delay time and the amount it is raining.

```
## # A tibble: 1,904 x 2
## # Groups:   word [1,904]
##    word      n
##    <chr> <int>
##  1 the     751
##  2 a       202
##  3 of      132
##  4 to      123
##  5 and     118
##  6 in       87
##  7 is       81
##  8 was      66
##  9 on       60
## 10 with     51
## # ... with 1,894 more rows
```

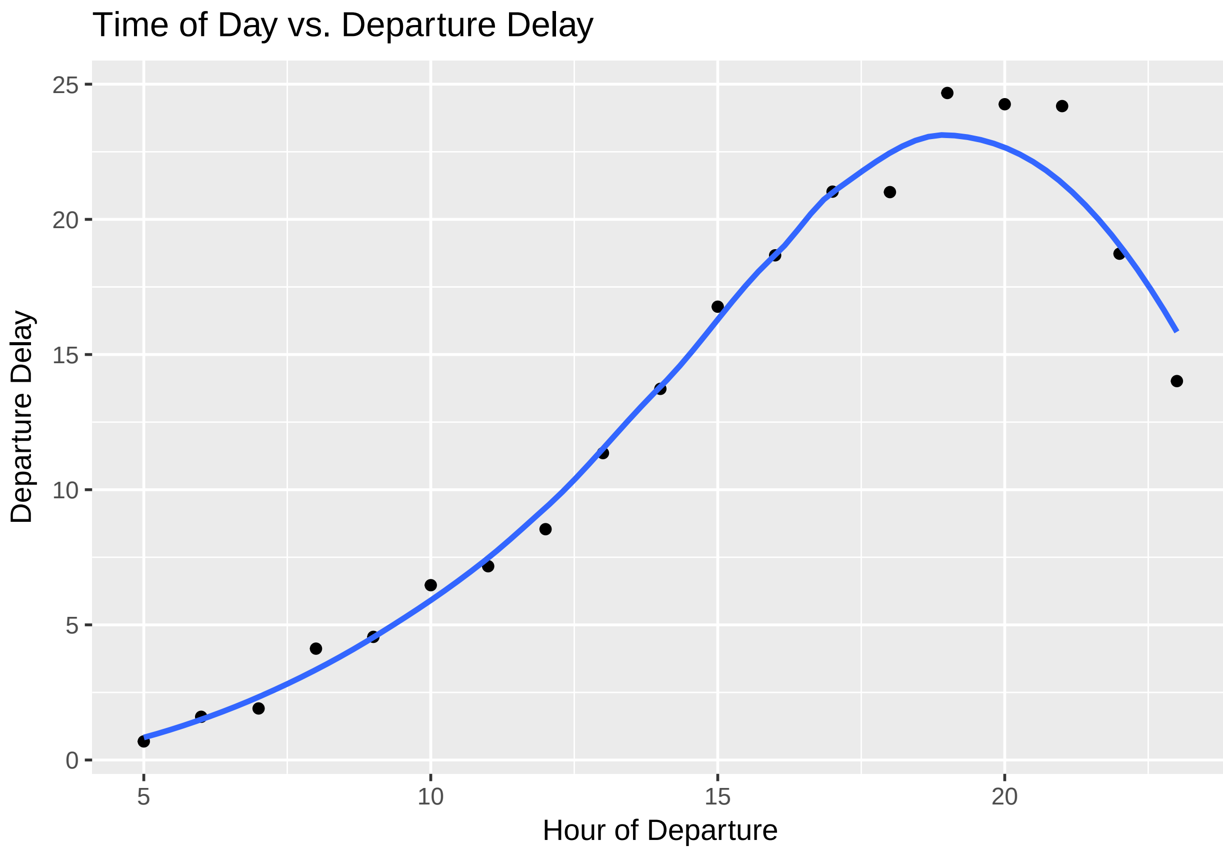
The chart is pretty self explanatory, the top five words used in sentences are: the, a, of, to, & and.

```
## `geom_smooth()` using method = 'loess'
```



This plot shows us that over time the amount of people who consider themselves other have increases slightly, republicans have decreased slightly, democrats have been mostly stable and independent people have oscillated throughout the years.

```
## `geom_smooth()` using method = 'loess'
```



As it gets later in the day, the number of departure delays increase which makes sense because as the morning flights get pushed back, people on the later flights will also be delayed because of what happens earlier in the day.

Scott Baker

13.4.6.2: Add the location of the origin and destination (lat,long) to flights

```
flights <- nycflights13::flights
airports <- nycflights13::airports
left_join(flights,airports, by = c("dest" = "faa"))
```

```
## # A tibble: 336,776 x 26
##     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      2.00      830
##  2  2013     1     1      533            529      4.00      850
##  3  2013     1     1      542            540      2.00      923
##  4  2013     1     1      544            545     -1.00     1004
##  5  2013     1     1      554            600     -6.00      812
##  6  2013     1     1      554            558     -4.00      740
##  7  2013     1     1      555            600     -5.00      913
##  8  2013     1     1      557            600     -3.00      709
##  9  2013     1     1      557            600     -3.00      838
## 10  2013     1     1      558            600     -2.00      753
## # ... with 336,766 more rows, and 19 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>, name <chr>, lat <dbl>, lon <dbl>,
## #   alt <int>, tz <dbl>, dst <chr>, tzone <chr>
```

```
left_join(flights,airports, by = c("origin"= "faa"))
```

```
## # A tibble: 336,776 x 26
##     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      2.00      830
##  2  2013     1     1      533            529      4.00      850
##  3  2013     1     1      542            540      2.00      923
##  4  2013     1     1      544            545     -1.00     1004
##  5  2013     1     1      554            600     -6.00      812
##  6  2013     1     1      554            558     -4.00      740
##  7  2013     1     1      555            600     -5.00      913
##  8  2013     1     1      557            600     -3.00      709
##  9  2013     1     1      557            600     -3.00      838
## 10  2013     1     1      558            600     -2.00      753
## # ... with 336,766 more rows, and 19 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>, name <chr>, lat <dbl>, lon <dbl>,
## #   alt <int>, tz <dbl>, dst <chr>, tzone <chr>
```

```
flights
```

```
## # A tibble: 336,776 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      2.00      830
##  2  2013     1     1      533            529      4.00      850
##  3  2013     1     1      542            540      2.00      923
##  4  2013     1     1      544            545     -1.00     1004
##  5  2013     1     1      554            600     -6.00      812
##  6  2013     1     1      554            558     -4.00      740
##  7  2013     1     1      555            600     -5.00      913
##  8  2013     1     1      557            600     -3.00      709
##  9  2013     1     1      557            600     -3.00      838
## 10  2013     1     1      558            600     -2.00      753
## # ... with 336,766 more rows, and 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>
```

14.3.2.1: Find words that:

Start with y

```
#words <- stringr::words
#str_view_all(words,"^y")
```

Start with x

```
#str_view_all(words,"x$")
```

Are three letters

```
#str_view_all(words,"^(...)$")
```

Have seven letters or more

```
#str_view_all(words,".......")
```

15.5.1.2:

```
#mutate(gss_cat,rincome = fct_collapse(rincome, 'NA' = c("Don't know","Refused","Not applicable","No an
```

16.4.5.3:

```
dates <- c(20150101,20150201,20150301,20150401,20150501,20150601,20150701,20150801,20150901,20151001,20
ymd(dates)
```

```
##  [1] "2015-01-01" "2015-02-01" "2015-03-01" "2015-04-01" "2015-05-01"
##  [6] "2015-06-01" "2015-07-01" "2015-08-01" "2015-09-01" "2015-10-01"
## [11] "2015-11-01" "2015-12-01"
```

```
tday <- today()
tday
```

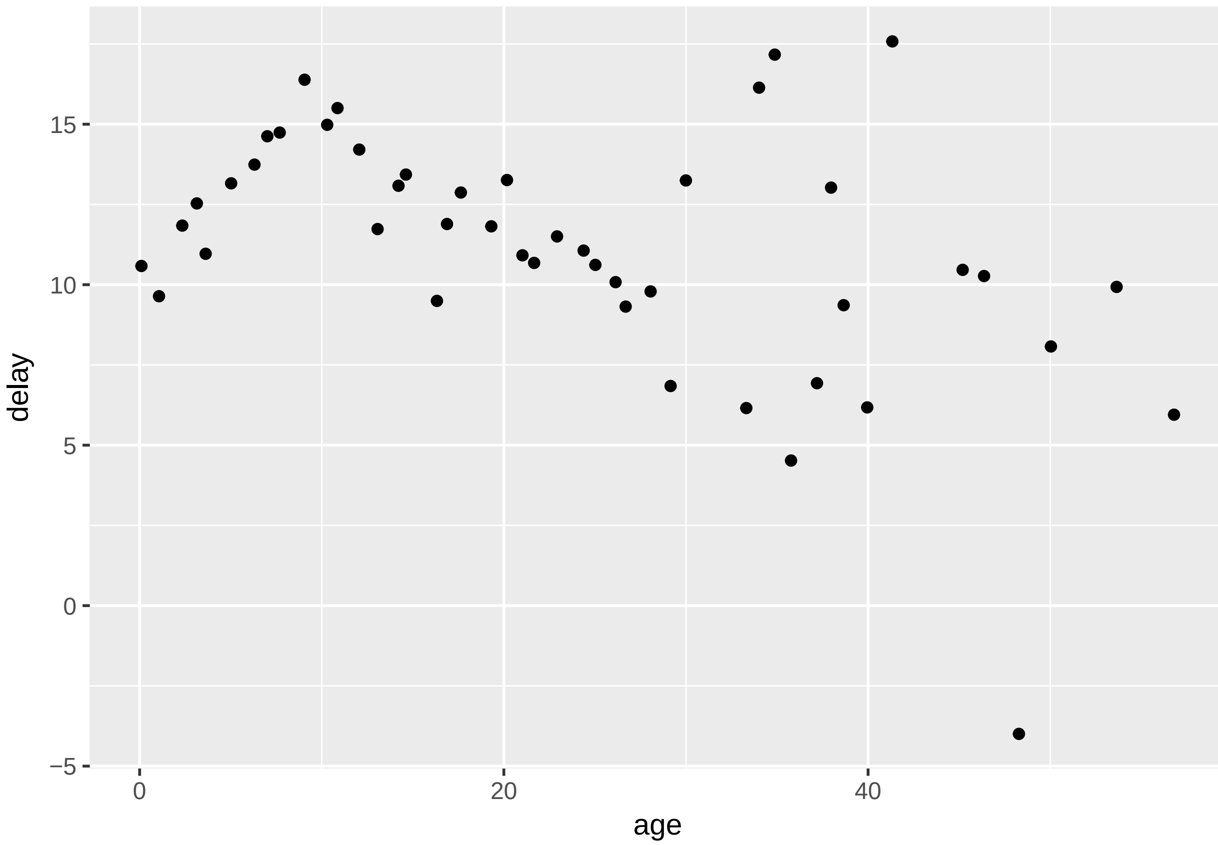
```
## [1] "2018-03-16"
```

```
thisYear <- dates + 30000
ymd(thisYear)
```

```
##  [1] "2018-01-01" "2018-02-01" "2018-03-01" "2018-04-01" "2018-05-01"
##  [6] "2018-06-01" "2018-07-01" "2018-08-01" "2018-09-01" "2018-10-01"
## [11] "2018-11-01" "2018-12-01"
```

Zhenlong Li

```
#13.4.6 #3
plane_ages <-
  planes %>%
  mutate(age = 2013 - year) %>%
  select(tailnum, age)

flights %>%
  inner_join(plane_ages, by = "tailnum") %>%
  group_by(age) %>%
  filter(!is.na(dep_delay)) %>%
  summarise(delay = mean(dep_delay)) %>%
  ggplot(aes(x = age, y = delay)) + geom_jitter()
```



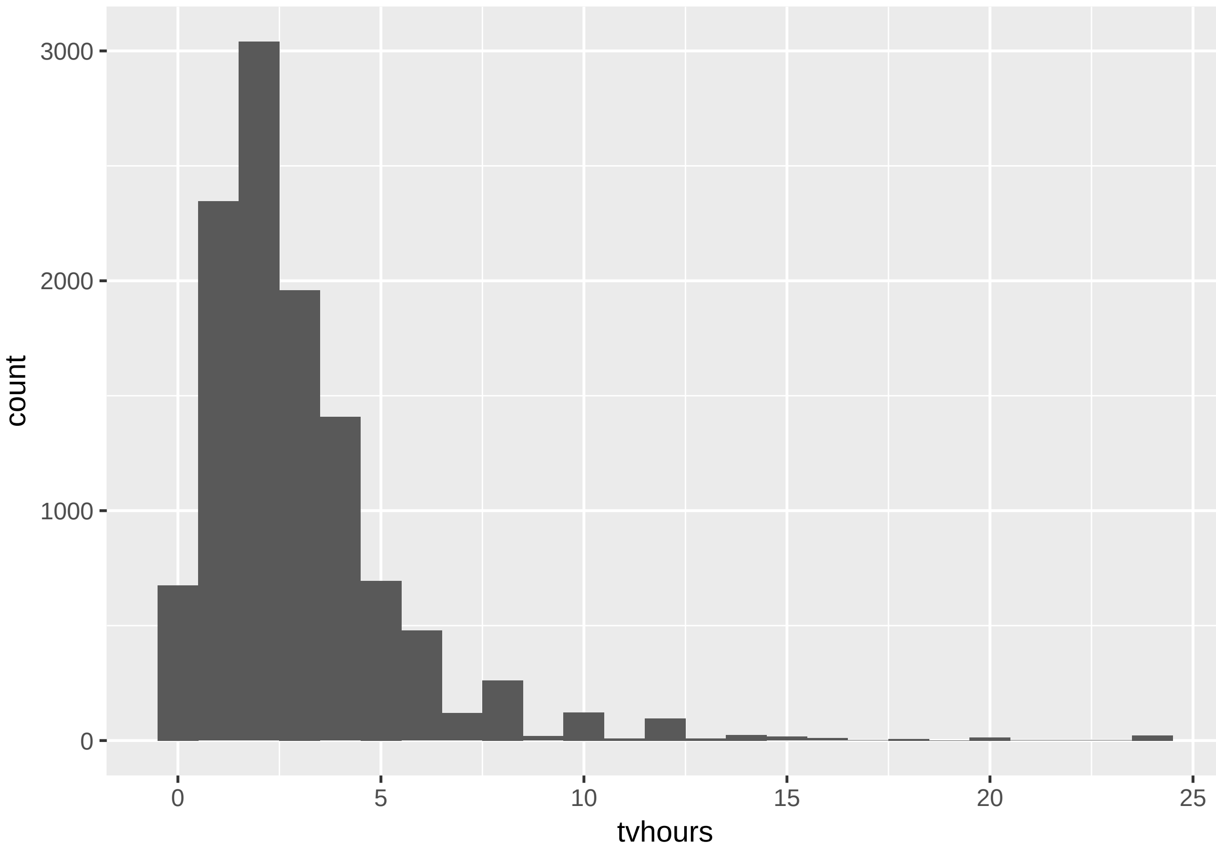
From the plot, we can see that there is no relationship between the age of a plane and its delays.

```
#14.3.3 #5
x <- c("123-456-7890", "1234-5678")
#str_view(x, "\\d\\d\\d-\\d\\d\\d-\\d\\d\\d\\d")
```

```
#15.4.1 #1
summary(gss_cat[["tvhours"]])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##   0.000   1.000   2.000   2.981   4.000  24.000   10146
```

```
gss_cat %>%
  filter(!is.na(tvhours)) %>%
  ggplot(aes(x = tvhours)) +
  geom_histogram(binwidth = 1)
```



I

think the median will be a better summary.

```
#16.2.4 #1
parsing <- ymd(c("2010-10-10", "bananas"))
```

```
## Warning: 1 failed to parse.
```

```
parsing
```

```
## [1] "2010-10-10" NA
```

It will produce an NA and a warning.

Contributions

- Lindsay: Answered four textbook questions, using semi_join, str_subset, fct_collapse, and make_datetime. These were also combined with ggplot, and dplyr functions.
- Lexie: Created individual plots answering 4 textbook questions. For the tidyverse functions, I used: stringr, ggplot2, tibble, and forcats.
- Li: Finished four exercise from book. I used following functions: mutate, select, inner_join, group_by, filter, summarise, summary, str_view, geom_jitter and geom_histogram.
- Scott: Completed four excercises from the book (note: for the 'strings' and 'factors' excercises it would not compile for knit–something about html namespace). Also worked on the Colorado Rockies team section. As usual, organized the git and managed files/knitting.