EDAV Fall 2019 Problem Set 2

Mrugank Akarte (mma2247) & Swapnav Deka (sd3344)

Read *Graphical Data Analysis with R*, Ch. 4, 5

Grading is based both on your graphs and verbal explanations. Follow all best practices as discussed in class. Data manipulation should not be hard coded. That is, your scripts should be written to work for new data.

1. useR2016! survey

[18 points]

Data: `useR2016` dataset in the **forwards** package (available on CRAN)

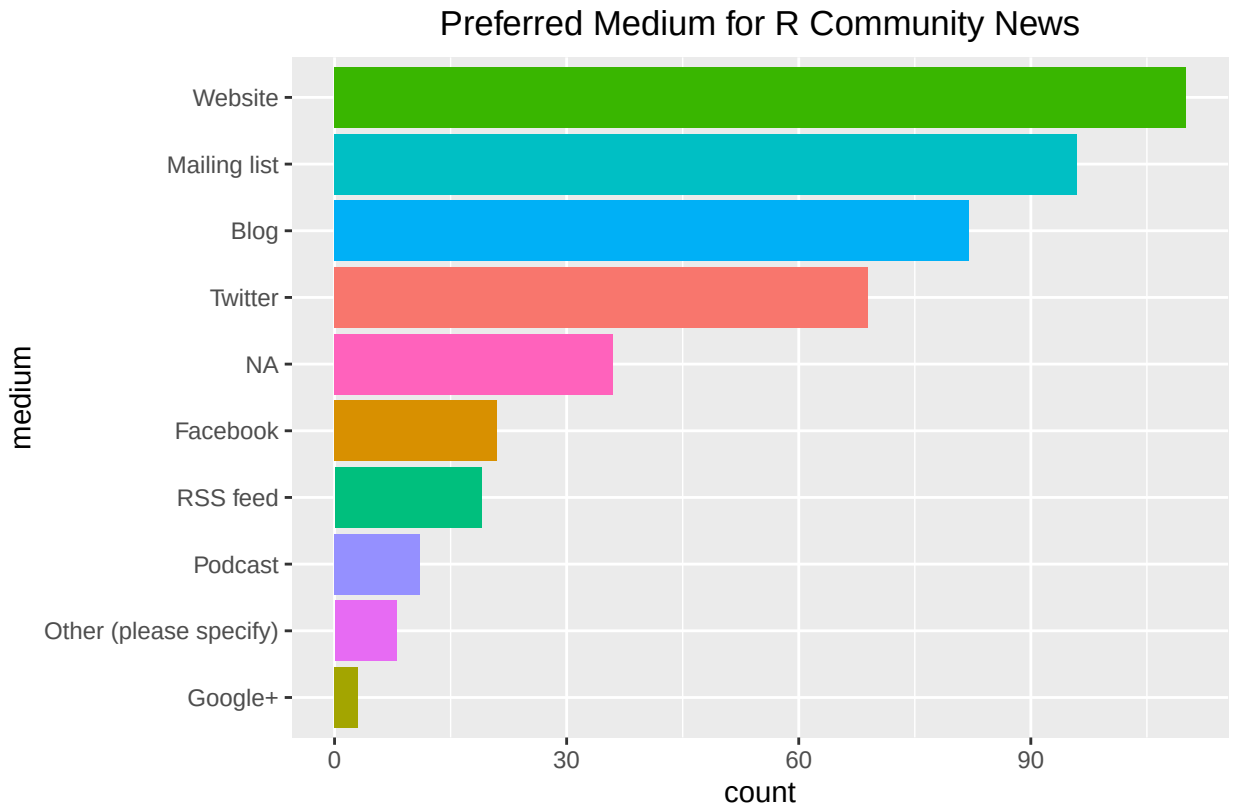
For parts (a) and (b):

- Do not toss NAs.
- Do some research to find the wording of the questions asked as relevant and include them in the titles of your graphs.
- Include the dataset name, package name, and link to the question wording source in the graph caption.

(a) Create a horizontal bar chart of the responses to Q20.

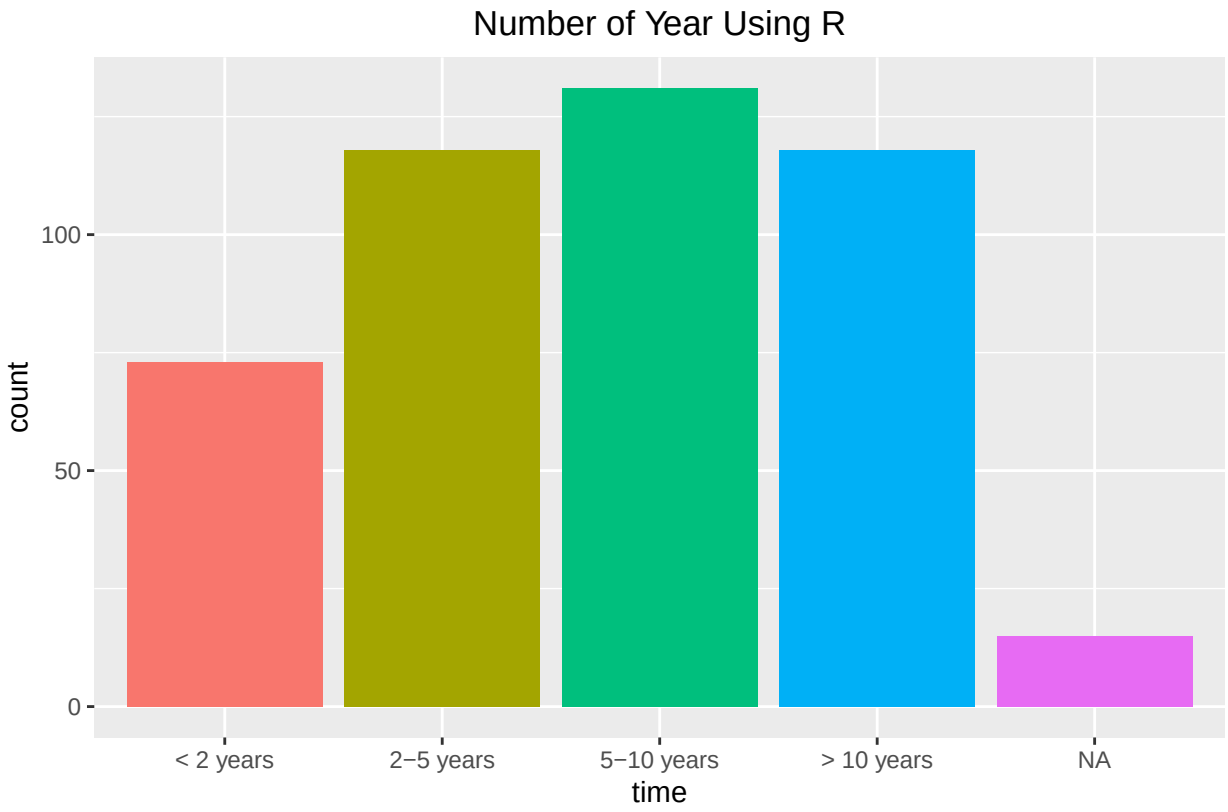
```
library(forwards)
library(tidyverse)
library(dplyr)
library(stringi)
library(forcats)
library(ggplot2)


useR2016 %>%
  mutate(social_media = fct_explicit_na(Q20, na_level = "NA")) %>%
  ggplot() +
  geom_bar(
    mapping = aes(fct_rev(fct_infreq(social_media)), fill = social_media),
    show.legend = F) +
  coord_flip() +
  labs(title = 'Preferred Medium for R Community News',
       caption = 'Dataset: useR2016 | Package: forwards | source: https://cran.r-project.org/web/packag
       x = "medium",
       y = 'count') +
  theme(plot.title = element_text(hjust = 0.5))
```



(b) Create a vertical bar chart of the responses to Q11.

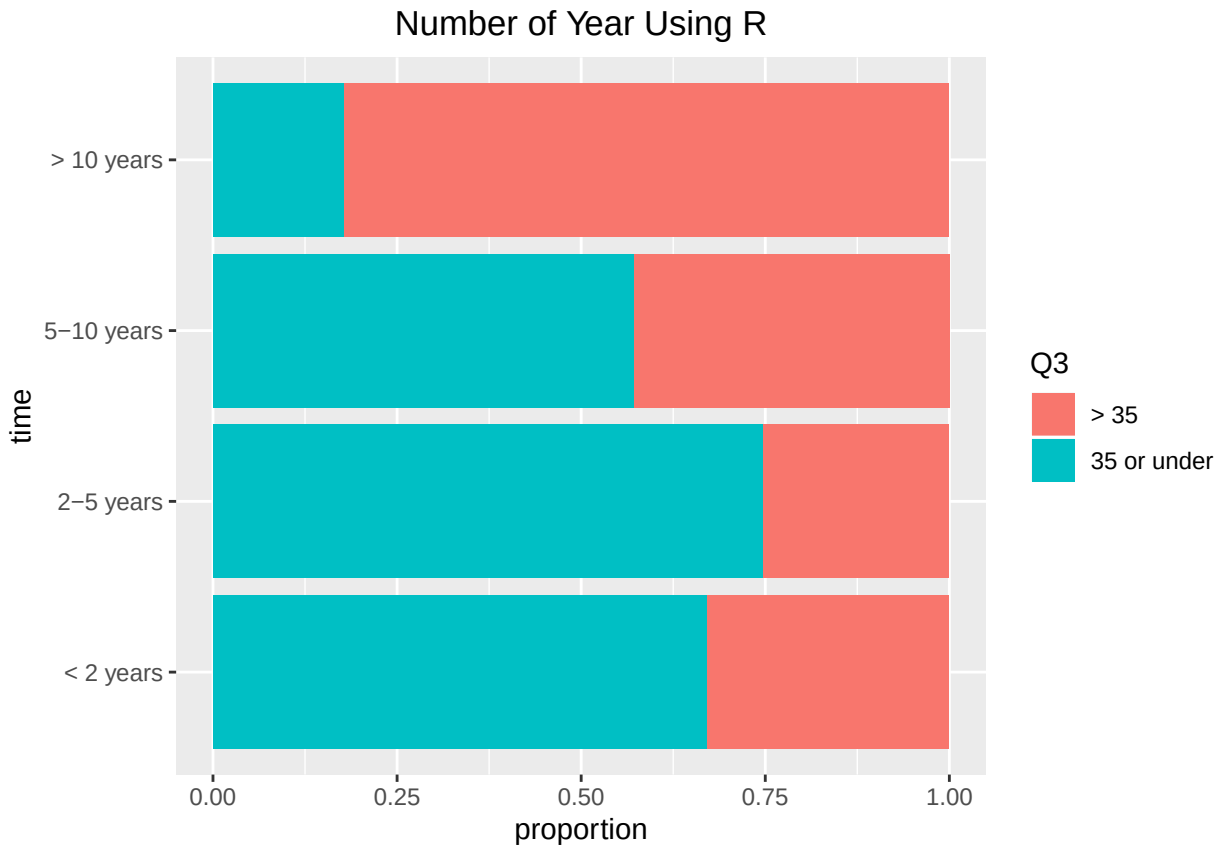
```
useR2016 %>%
  mutate(social_media = fct_explicit_na(Q11, na_level = "NA")) %>%
  ggplot() +
  geom_bar(mapping = aes(social_media, fill = social_media), show.legend = F) +
  labs(x = 'time', y = 'count',
       title = "Number of Year Using R",
       caption = 'Dataset: useR2016 | Package: forwards | source: https://cran.r-project.org/web/packag
  theme(plot.title = element_text(hjust = 0.5))
```



Dataset: useR2016 | Package: forwards | source: https://cran.r−project.org/web/packages/forwards/forwards.pdf

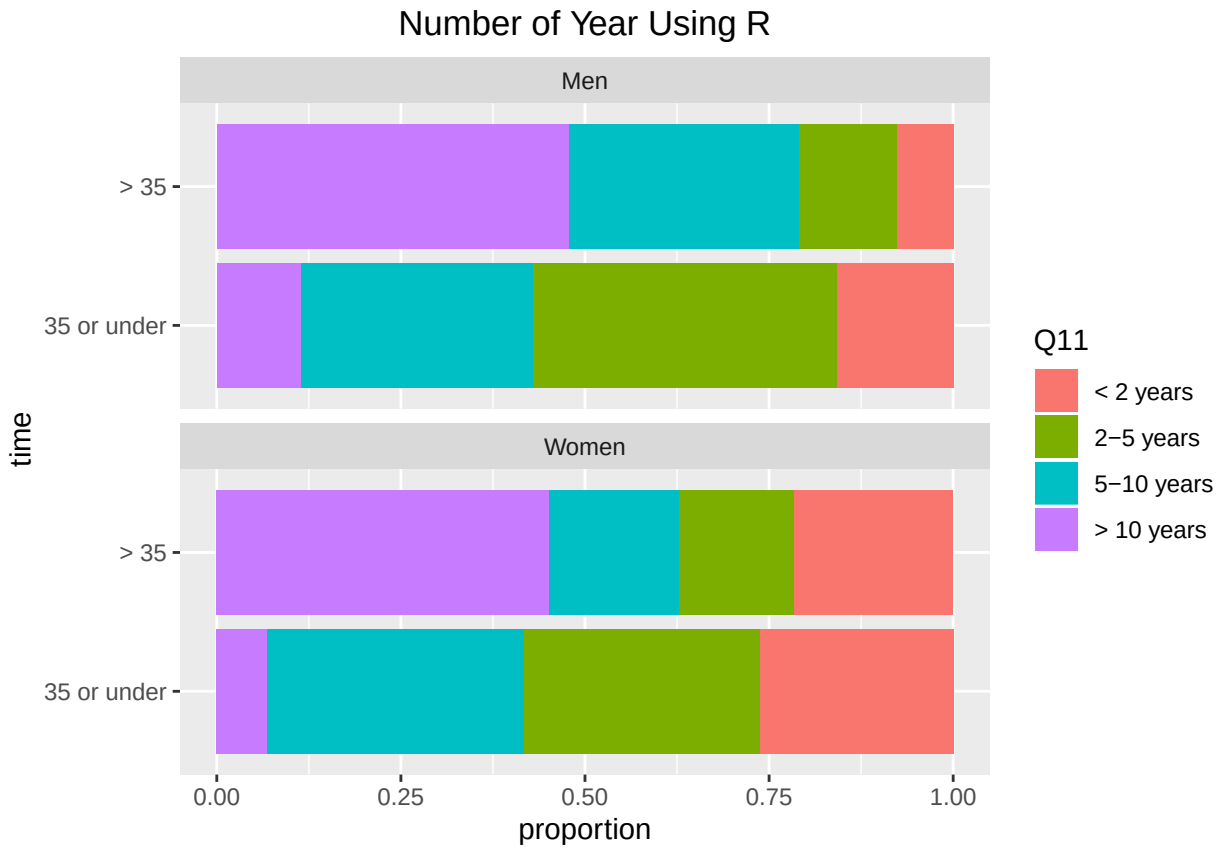
(c) Create a horizontal stacked bar chart showing the proportion of respondents for each level of Q11 who are over 35 vs. 35 or under. Use a descriptive title.

```
useR2016 %>%
  remove_missing(vars = c('Q3', 'Q11'), na.rm = T) %>%
  ggplot() +
  geom_bar(mapping = aes(x = Q11, fill = Q3), position='fill') +
  coord_flip() +
  labs(x = 'time', y = 'proportion', title = "Number of Year Using R") +
  theme(plot.title = element_text(hjust = 0.5))
```



(d) Create a horizontal stacked bar chart showing the proportional breakdown of Q11 for each level of Q3, faceted on Q2. Use a descriptive title.

```
useR2016 %>%
  remove_missing(vars = c('Q2', 'Q3', 'Q11'), na.rm = T) %>%
  ggplot() +
  geom_bar(mapping = aes(x = fct_rev(Q3), fill = Q11), position='fill') +
  coord_flip() +
  facet_wrap(~Q2, dir = 'v') +
  labs(x = 'time', y = 'proportion', title = "Number of Year Using R") +
  theme(plot.title = element_text(hjust = 0.5))
```



(e) For the next part, we will need to be able to add line breaks (\n) to long tick mark labels. Write a function that takes a character string and a desired approximate line length in number of characters and substitutes a line break for the first space after every multiple of the specified line length.

```
wrap_string <- function(string, len){
  splt <- strsplit(string, " ")
  new <- vector(mode="character", length=1)
  vec_len <- 0
  for (elem in splt[[1]]){
    if (vec_len > len){
      new <- paste(new, '\n')
      new<- paste0(new, elem, sep=' ')
      vec_len = 0

    }
    else
      new <- paste0(new, elem, sep=' ')
      vec_len <- vec_len + stri_length(elem) + 1

  }
  return(new)

}

cat(wrap_string('I like eating pies and donuts because I am fat.', 10))
```

```
## I like eating
## pies and donuts
```

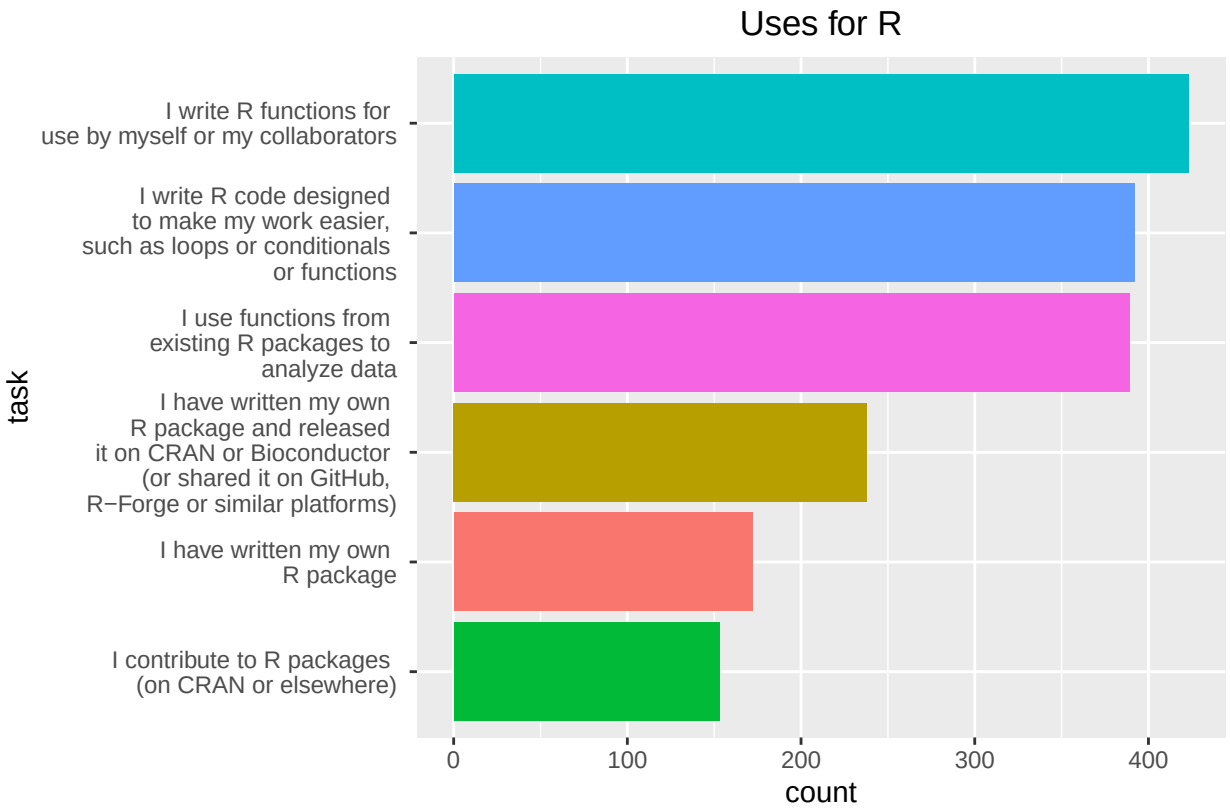
```
## because I am
## fat.
```

(f) Create a horizontal bar chart that shows the percentage of positive responses for `Q13 - Q13_F`. Use your function from part (e) to add line breaks to the responses. Your graph should have one bar each for `Q13 - Q13_F`.

```
dat <- useR2016[7:12] %>%
  gather(var, response) %>%
  count(response) %>%
  na.omit()

labels_wrap = c()
for (name in dat$response){
  var <- wrap_string(name, 20)
  labels_wrap <- c(labels_wrap, var)
}

dat %>%
  ggplot() +
  geom_bar(mapping = aes(x=reorder(response, n), y=n, fill=response),
           stat='identity') +
  coord_flip() +
  labs(x = 'task', y = 'count',
       title = "Uses for R",
       caption = 'Dataset: useR2016 | Package: forwards | source: https://cran.r-project.org/web/packag
  theme(plot.title = element_text(hjust = 0.5),
        axis.text.y = element_text(angle = 0),
        legend.position = 'none'
          ) +
  scale_x_discrete(labels = labels_wrap)
```



Dataset: useR2016 | Package: forwards | source: https://cran.r−project.org/web/packages/forwards/forwards.pdf

2. Rotten Tomatoes

[18 points]

To get the data for this problem, we'll use the **robotstxt** package to check that it's ok to scrape data from Rotten Tomatoes and then use the **rvest** package to get data from the web site.

(a) Use the `paths_allowed()` function from **robotstxt** to make sure it's ok to scrape https://www.rottentomatoes.com/browse/box-office/. Then use **rvest** functions to find relative links to individual movies listed on this page. Finally, paste the base URL to each to create a character vector of URLs.

Display the first six lines of the vector.

```
library(rvest)
library(robotstxt)

url <- 'https://www.rottentomatoes.com/browse/box-office/'
paths_allowed(paths=url)
```

```
## [1] TRUE
```

```
url <- html_attr(html_nodes(read_html(url), ".left a"), "href")

movies <- paste0('https://www.rottentomatoes.com', url)

head(movies)
```

```
## [1] "https://www.rottentomatoes.com/m/abominable/"
## [2] "https://www.rottentomatoes.com/m/downton_abbey/"
```

```
## [3] "https://www.rottentomatoes.com/m/hustlers_2019/"
## [4] "https://www.rottentomatoes.com/m/it_chapter_two/"
## [5] "https://www.rottentomatoes.com/m/ad_astra/"
## [6] "https://www.rottentomatoes.com/m/rambo_last_blood/"
```

(b) Write a function to read the content of one page and pull out the title, tomatometer score and audience score of the film. Then iterate over the vector of all movies using `do.call() / rbind() / lapply()` or `dplyr::bind_rows() / purrr::map()` to create a three column data frame (or tibble).

Display the first six lines of your data frame.

(Results will vary depending on when you pull the data.)

For help, see this SO post: https://stackoverflow.com/questions/36709184/build-data-frame-from-multiple-rvest-elements

Write your data to file so you don't need to scrape the site each time you need to access it.

```
score_meter_function <- function(links){
  movie_meter = data.frame(movie = character(),
                           tomato_meter = character(),
                           audience_score = character())

  for(link in links){
    movie <- read_html(x = link)
    movie_title <- html_text(html_nodes(movie, ".mop-ratings-wrap__title--top"))
    score <- html_text(html_nodes(movie, ".mop-ratings-wrap__percentage"))
    score <-unlist(stringr::str_extract_all(string = score, pattern = '\\d+'))
    if(is.null(score)){score = c(NA, NA)}
    df <- data.frame(movie = movie_title,
                     tomato_meter = score[1],
                     audience_score = score[2])
    movie_meter <- rbind(movie_meter, df)
  }

  return(movie_meter)
}
movie_meter <- score_meter_function(movies)
head(movie_meter, 6)
```

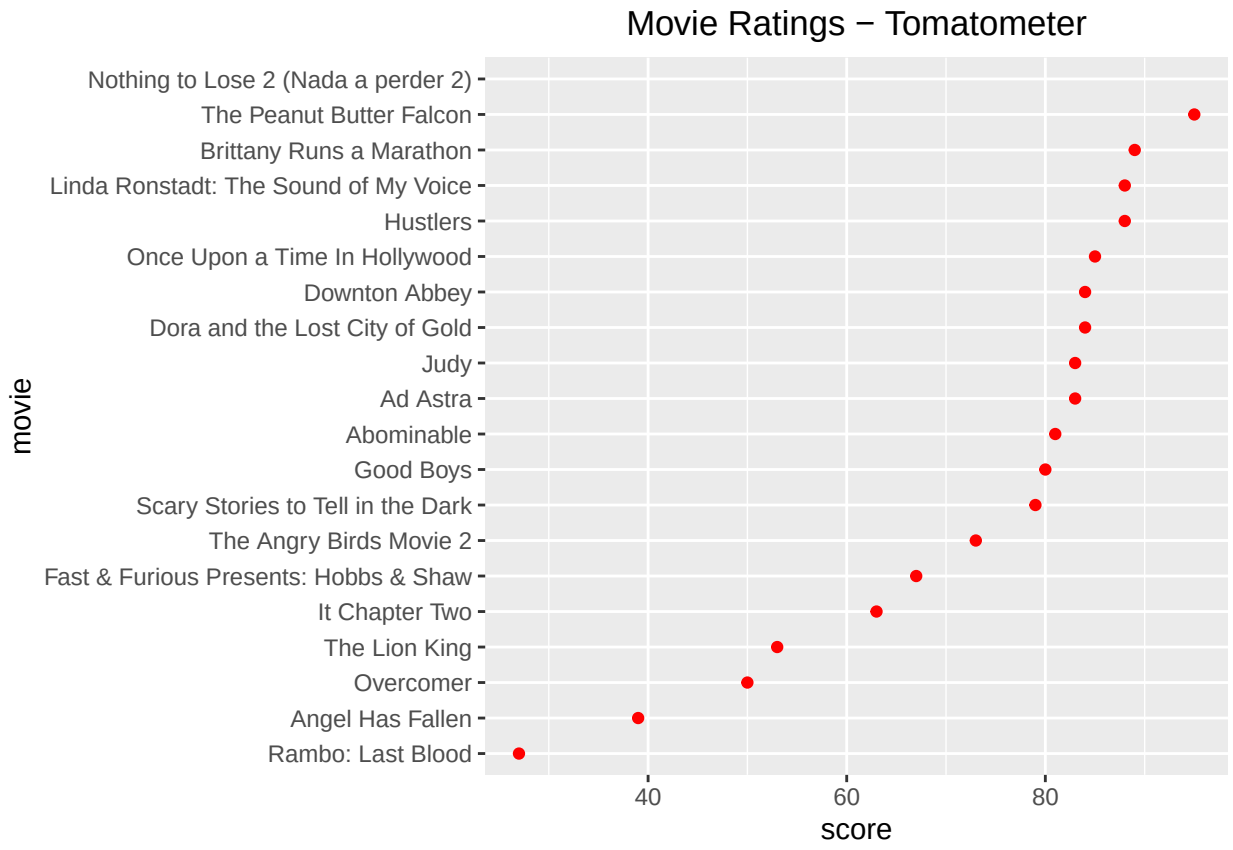
```
##               movie tomato_meter audience_score
## 1        Abominable           81             96
## 2     Downton Abbey           84             95
## 3          Hustlers           88             66
## 4    It Chapter Two           63             78
## 5          Ad Astra           83             40
## 6 Rambo: Last Blood           27             83
```

```
write.csv(movie_meter, 'movie_meter.csv', row.names = F)
movie_meter <- read.csv('movie_meter.csv')
```

(c) Create a Cleveland dot plot of tomatometer scores.

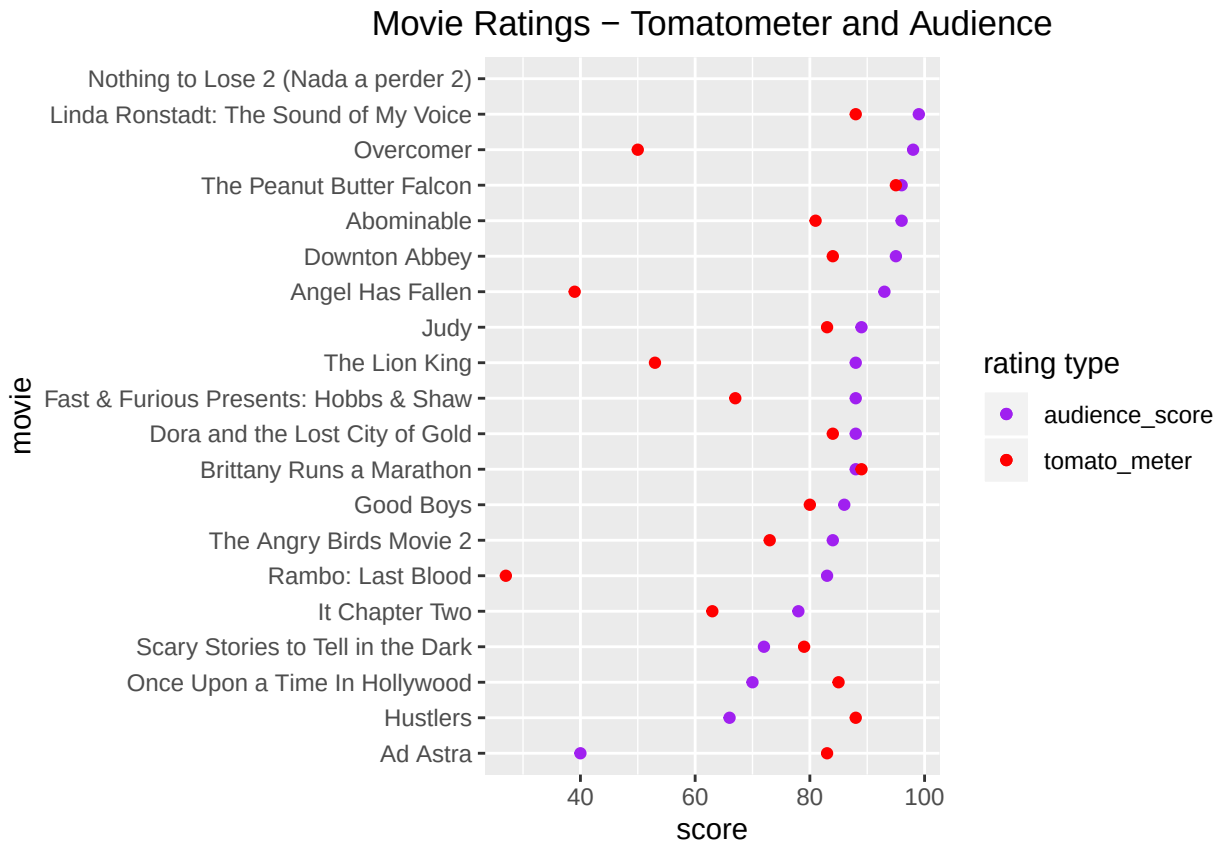
```
ggplot(movie_meter) +
  geom_point(
    aes(y = reorder(movie, tomato_meter), x = tomato_meter),
    color = 'red') +
  labs(
    title = 'Movie Ratings - Tomatometer',
```

```
    x = 'score',
    y = 'movie') +
  theme(plot.title = element_text(hjust = 0.5))
```



(d) Create a Cleveland dot plot of tomatometer *and* audience scores on the same graph, one color for each. Sort by audience score.

```
ggplot(movie_meter) +
  geom_point(aes(y = reorder(movie, audience_score), x = audience_score, color = "audience_score")) +
  geom_point(aes(y = movie, x = tomato_meter, color = "tomato_meter")) +
  scale_color_manual(name = "rating type",
                     values = c("audience_score" = "purple",
                                "tomato_meter" = "red")) +
  labs(title = 'Movie Ratings - Tomatometer and Audience',
    x = 'score',
    y = 'movie') +
  theme(plot.title = element_text(hjust = 0.5))
```

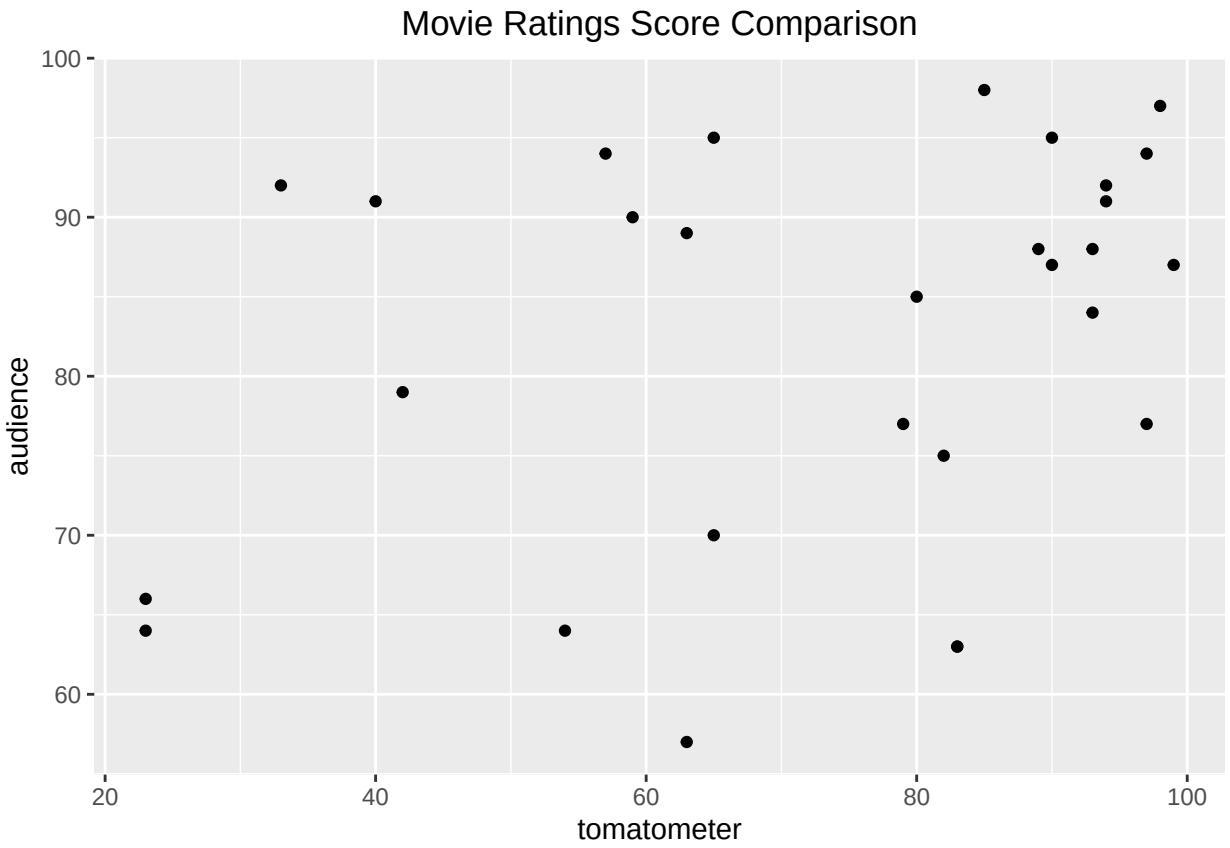


(e) Run your code again for the weekend of July 5 - July 7, 2019. Use **plotly** to create a scatterplot of audience score vs. tomatometer score with the ability to hover over the point to see the film title.

```
html2 <- read_html('https://www.rottentomatoes.com/browse/box-office/?rank_id=11&country=us')
url2 <- html_attr(html_nodes(html2, ".left a"), "href")
base_url <- 'https://www.rottentomatoes.com'

vec2 <- paste0(base_url, url2)
movie_meter2 <- score_meter_function(vec2)
write.csv(movie_meter2, 'movie_meter2.csv', row.names = F)

movie_meter2 <- read.csv('movie_meter2.csv')
library(plotly)
(g <- ggplot(movie_meter2) +
  geom_point(aes(tomato_meter, audience_score), show.legend = F) +
  labs(title = 'Movie Ratings Score Comparison',
    x = 'tomatometer',
    y = 'audience') +
  theme(plot.title = element_text(hjust = 0.5)))
```




```
##Uncomment to render interactive plot
#(plt <- ggplotly(g, tooltip = "movie"))
#htmlwidgets::saveWidget(as_widget(plt),'Q3E.html')
```

3. Weather

[14 points]

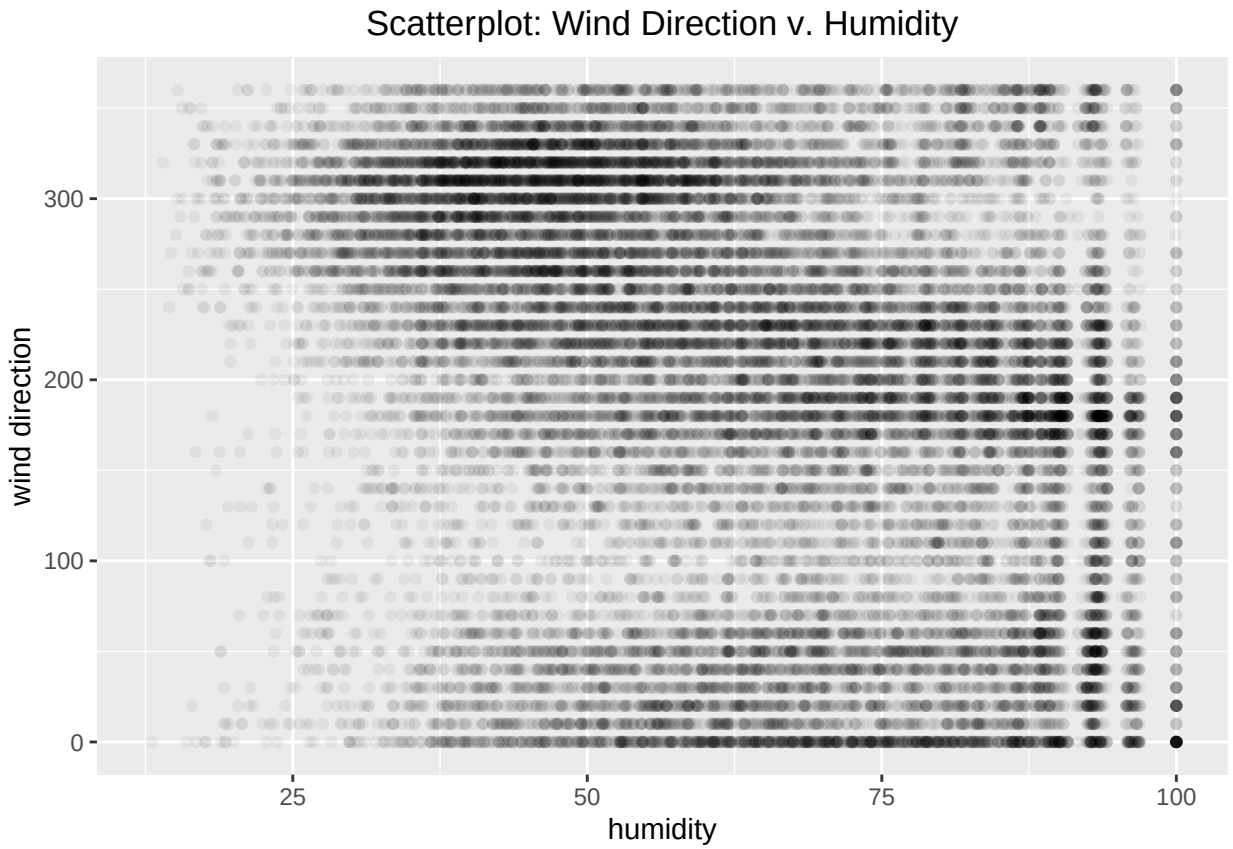
Data: `weather` dataset in **nycflights13** package (available on CRAN)

```
library(nycflights13)
library(hexbin)
```

For parts (a) - (d) draw four plots of `wind_dir` vs. `humid` as indicated. For all, adjust parameters to the levels that provide the best views of the data.

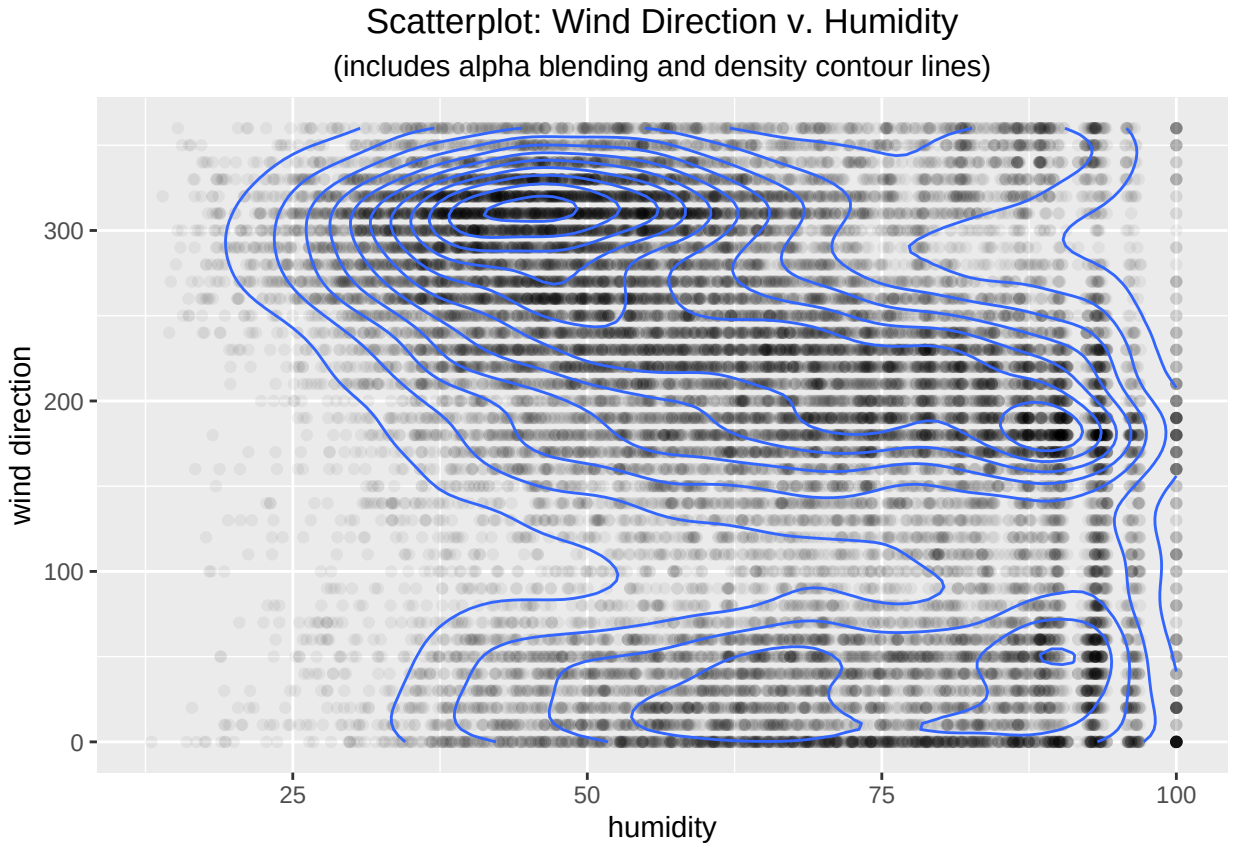
(a) Points with alpha blending

```
ggplot(weather, aes(humid, wind_dir)) +
  geom_point(alpha = 0.05) +
  labs(title = 'Scatterplot: Wind Direction v. Humidity',
    x = 'humidity',
    y = 'wind direction') +
  theme(plot.title = element_text(hjust = 0.5))
```



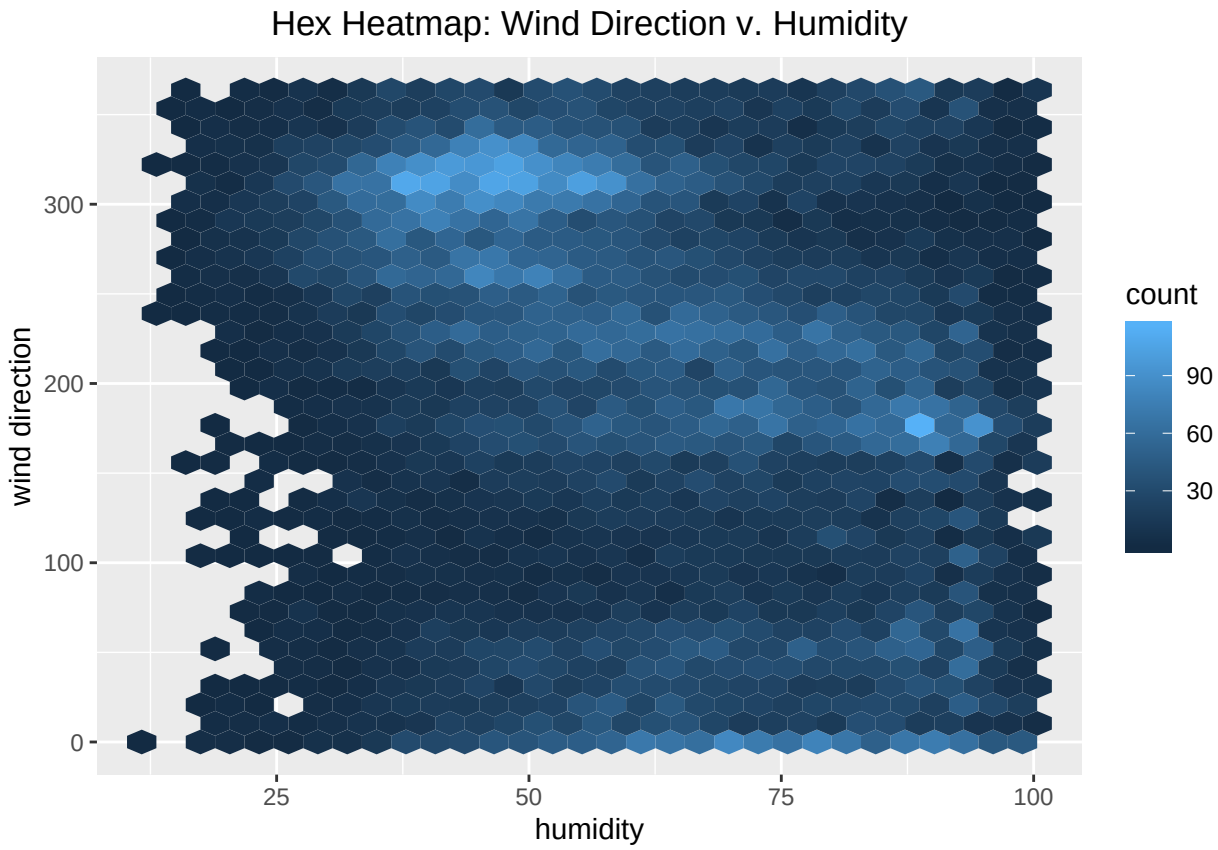
(b) Points with alpha blending + density estimate contour lines

```
ggplot(weather, aes(humid, wind_dir)) +
  geom_point(alpha = 0.05) +
  geom_density_2d(aes(x = humid, y = wind_dir)) +
  labs(title = 'Scatterplot: Wind Direction v. Humidity',
       subtitle = '(includes alpha blending and density contour lines)',
    x = 'humidity',
    y = 'wind direction') +
  theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5))
```



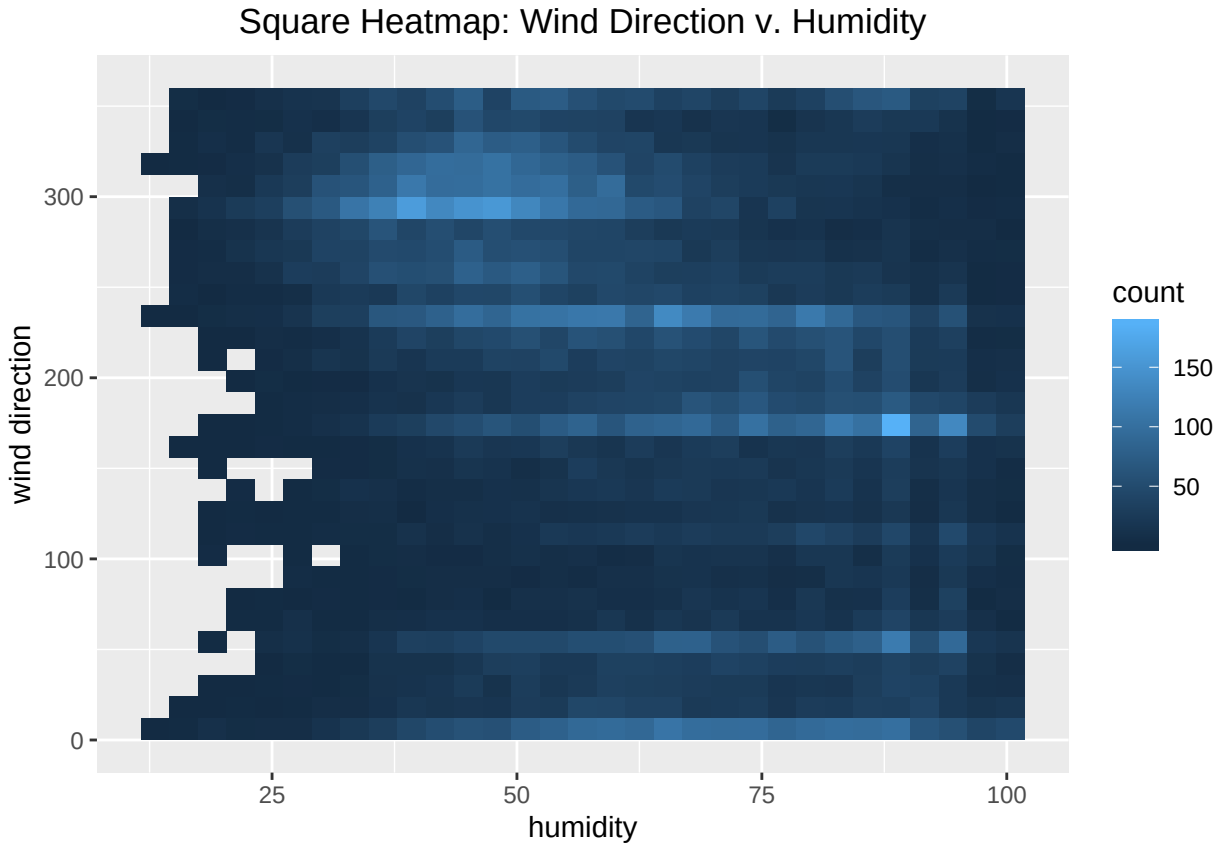
(c) Hexagonal heatmap of bin counts

```
ggplot(weather, aes(humid, wind_dir)) +
  geom_hex() +
  scale_fill_continuous() +
  labs(title = 'Hex Heatmap: Wind Direction v. Humidity',
    x = 'humidity',
    y = 'wind direction') +
  theme(plot.title = element_text(hjust = 0.5))
```



(d) Square heatmap of bin counts

```
ggplot(weather, aes(humid, wind_dir)) +
  geom_bin2d() +
  scale_fill_continuous() +
  labs(title = 'Square Heatmap: Wind Direction v. Humidity',
    x = 'humidity',
    y = 'wind direction') +
  theme(plot.title = element_text(hjust = 0.5))
```



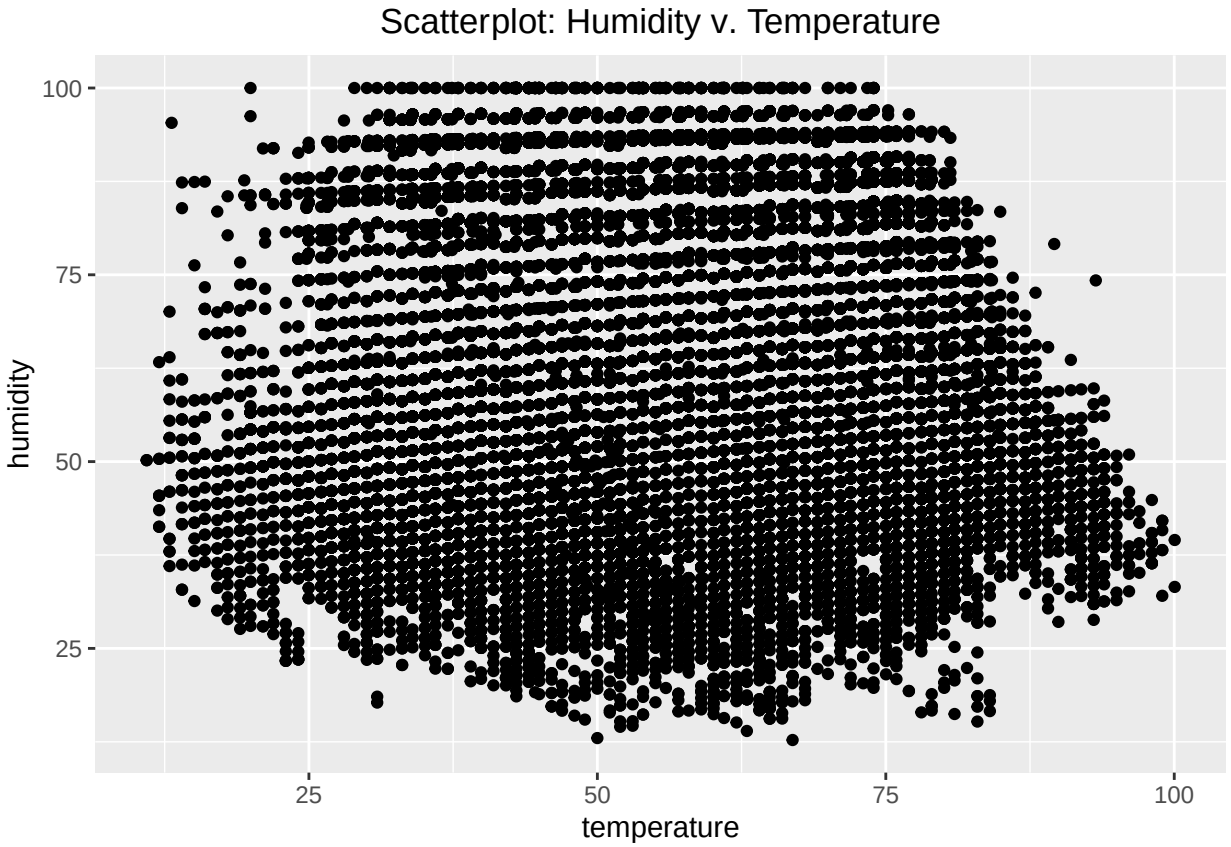
(e) Describe noteworthy features of the data, using the "Movie ratings" example on page 82 (last page of Section 5.3) as a guide.

- Two strong and one weak cluster can be observed in the data (Wind_dir vs Humid).
- Majority of points lie around a point 50% humid and wind speed of 300
- Second cluster can be seen arounf point with around 90% humid and wind speed of 175.
- The third weak cluster lies arounf bottom half of the graph.
- There are minimum point with low humidity and low wind speeds.

The horizontal line pattern in the scatter plot with clear spaces in between each stripe shows that the wind direction data are seemingly discrete in measure. This can be verified by looking at the table and observing that they are exclusively in multiples of 10. Although there is no strong correlation between the two variables, you can observe a slight trend for higher wind directions. It seems that past a certain point (around 150-200), there could be a very very weak negative correlation between the two variables.

(f) Draw a scatterplot of `humid` vs. `temp`. Why does the plot have diagonal lines?

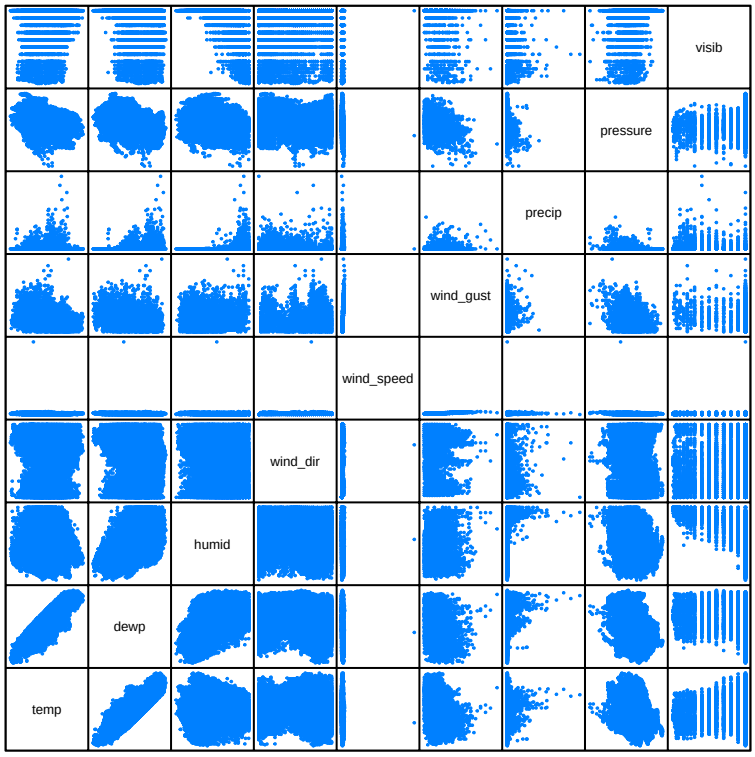
```
ggplot(weather) +
  geom_point(aes(temp, humid)) +
  labs(title = 'Scatterplot: Humidity v. Temperature',
    x = 'temperature',
    y = 'humidity') +
  theme(plot.title = element_text(hjust = 0.5))
```



The diagonal line patterns seem to show that the humidity is a function of the temperature variable. This would explain the systematic nature of the pattern. A possible explanation for this could be that the humidity variable is not actually measuring absolute humidity, but rather relative humidity. This would make sense, because relative humidity is a measure relative to the temperature variable.

(g) Draw a scatterplot matrix of the continuous variables in the `weather` dataset. Which pairs of variables are strongly positively associated and which are strongly negatively associated?

```
library(lattice)
splom(select_if(weather, is.double)[1:9], cex = .1, pscales = 0, varname.cex = .4)
```

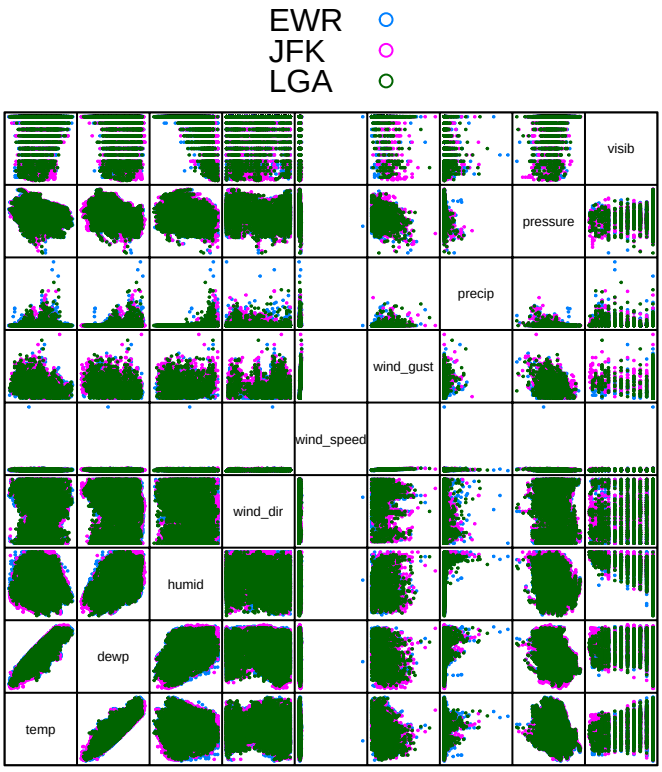


Scatter Plot Matrix

From the looks of the matrix, the only strong correlation seems to be between dewpoint and temperature, which exhibits a strong positive correlation. Humidity and dewpoint could potentially be a very weak positive correlation, but the rest of the pairwise correlations are not informative.

(h) Color the points by `origin`. Do any new patterns emerge?

```
splom(select_if(weather, is.double)[1:9], cex = .1, pscales = 0, varname.cex = .4, groups=weather$origi
```



Scatter Plot Matrix

In the majority of the scatterplots, the EWR originations seem to by the least densely distributed, often appearing at the outskits of the plots. The same is true about JFK originations, but to a slightly lesser extent. La Guardia originations seem to be the most common and most densely distributed in every plot, and also tightly concentrated compared to its neighboring airports.