Problem Set 2: Data Wrangling

Wade Lindmark

```r
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
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----------------------- tidyverse 2.0.0 --
## v forcats   1.0.0      v readr     2.1.5
## v ggplot2   3.5.1      v stringr   1.5.1
## v lubridate 1.9.3      v tibble    3.2.1
## v purrr     1.0.2      v tidyr     1.3.1

## -- Conflicts --------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(readr)
library (knitr)
```

Background

Political advertising has traditionally been focused on the medium of television, but in recent cycles, online advertising has become much more popular. In this problem set, you will explore a data set that has information on Facebook ad spending and impressions by candidates in the 2018 election cycle in the United States. The variables in this data are described below.

Name	Description
cand_id	unique identifier code for candidate
cand_name	full name of the candidate

Name	Description
cand_name_last	last name of the candidate
party	party affiliation of the candidate (R = Republican, D = Democrat)
office	office being sought by candidate
state	state in which the candidate is running
incumbency	incumbency status of candidate (incumbent, challenger, or open seat)
spend	estimated total spending on Facebook ads by candidate
impressions	estimated total impressions of Facebook ads
ad_tone_attack	proportion of FB ads that mention candidate's opponent only
ad_tone_promote	proportion of FB ads that mention candidate only
ad_tone_contrast	proportion of FB ads that mention candidate and candidate's opponent

Question 1 (8 points)

Load the data using the `read_csv` function and save it as `fb_ads` (using this will automatically make `fb_ads` a tibble). In the text, describe how many candidates there are in the data set.

Use `dplyr` functions to create a table with the number of candidates in each type of incumbency status in the data set. Save this table output as `incumbency_table` (for the autograder). Use the function `knitr::kable()` on this table to have a nicely formatted table produced in the knitted output.

Rubric: 2pt for loading the data (autograder); 1pt for describing the number of candidates (PDF); 3pts for creating the table (autograder); 2pt for using `kable()` to nicely format the output (PDF)

Answer 1

```
fb_ads <-
  read_csv("data/fb_ads.csv")
```

```
## Rows: 7014 Columns: 12
## -- Column specification ----------------------------------------------------------------
## Delimiter: ","
## chr (7): cand_id, cand_name, cand_name_last, party, office, state, incumbency
## dbl (5): spend, impressions, ad_tone_attack, ad_tone_promote, ad_tone_contrast
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
incumbency_table <-
  fb_ads |> count(incumbency, sort = TRUE)

knitr::kable(incumbency_table)
```

incumbency	n
Challenger	2510
Open Seat	2482
Incumbent	2022

```
incumbency_table
```

```
## # A tibble: 3 x 2
##   incumbency      n
##   <chr>       <int>
## 1 Challenger   2510
## 2 Open Seat    2482
## 3 Incumbent    2022
```

There are 7,014 rows in this data set, meaning that there are 7,014 candidates in total. Comprising this total are 2,510 challengers, 2,482 open seats, and 2,022 incumbents seeking reelection.

Question 2 (7 points)

Filter the data to just US House and US Senate races and use this to create a tibble called `party_incumbent_promote` that has 6 rows that summarizes the average of `ad_tone_promote` for each combination of `party` and `incumbency`. Call the variable summarizing the promote variable as `promote_prop` and be sure to remove any missing values when computing the averages.

Use `knitr::kable()` to produce a nicely formatted table. In this call, set the `digits` arguments to 3 and use the `col.names` argument to pass a nicer set of names. You can use the following as a template:

In the writeup, describe which type of candidate sponsored the most promoting ads on average.

Rubric: 3pts for creating `party_incumbent_promote` correctly (autograder); 2pt for a nicely formatted table (PDF); 1pt for changing the column names of the output table (PDF); 1pt for correctly identifying the type of candidate with highest average (PDF)

Answer 2

```
party_incumbent_promote <- fb_ads |>
  filter(office %in% c("US Senate", "US House")) |>
  group_by(party, incumbency) |>
  summarize(promote_prop = mean(ad_tone_promote, na.rm = TRUE))
```

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

```
party_incumbent_promote
```

```
## # A tibble: 6 x 3
## # Groups:   party [2]
##    party incumbency promote_prop
##    <chr> <chr>             <dbl>
## 1 D      Challenger        0.833
## 2 D      Incumbent         0.854
## 3 D      Open Seat         0.845
## 4 R      Challenger        0.813
## 5 R      Incumbent         0.792
## 6 R      Open Seat         0.828
```

```
knitr::kable(party_incumbent_promote, digits = 3,
             col.names = c("Party", "Incumbent Status",
                           "Self-Promotion Ad Proportion"))
```

Party	Incumbent Status	Self-Promotion Ad Proportion
D	Challenger	0.833
D	Incumbent	0.854
D	Open Seat	0.845
R	Challenger	0.813
R	Incumbent	0.792
R	Open Seat	0.828

The answer with the highest average of self-promotion ad proportion is among Democratic incumbent candidates, with a proportion of .8544, higher than all other proportions in the table.

Question 3 (7 points)

Create a new variable called `impressions_millions` that is the total Facebook ad impressions in millions (as opposed to single impressions). Make sure to save the resulting dataset back as `fb_ads`.

Create a histogram of this variable for just the US House races. Save the ggplot output as `plot_q3` and also print it to produce a plot in the output. In the text, describe the shape of the histogram and tell the reader if most of the House candidates had more than 10 million ads impressions on Facebook.

Rubric: 2pt for creating the new variable (autograder); 3pts for creating the histogram object (autograder); 2pts for answering the question about the histogram (PDF)

Answer 3

```r
fb_ads <- fb_ads |>
  mutate(impressions_millions = impressions / 1000000)

fb_ads_2 <- fb_ads |>
  filter(office == "US House")

plot_q3 <- ggplot(data = fb_ads_2, aes(x = impressions_millions)) +
  geom_histogram(fill = "darkorchid3") +
  labs(title = "Distribution of Facebook Ad Impressions for US House Candidates",
       x = "Impressions (in millions)", y = "Number of Candidates")

plot_q3
```
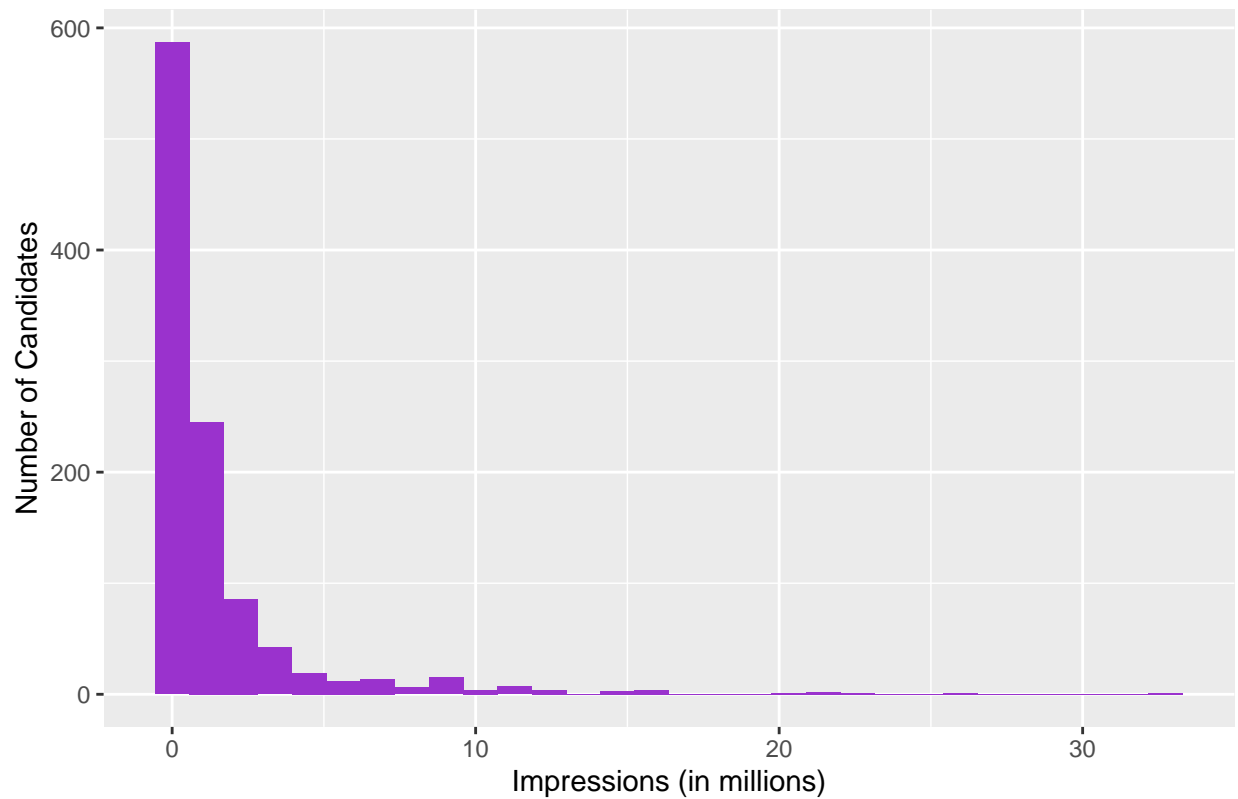
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Distribution of Facebook Ad Impressions for US House Candidates

Histogram Answer: The histogram is skewed to the right, meaning that most U.S. House candidates had **FAR FEWER** than 10 million impressions on Facebook, as the majority of the data is between 0 and 5 million impressions.

Question 4 (13 points)

Let's now recreate the following plot that shows the top 15 House candidates in terms of Facebook ad impressions.

You should save the `ggplot` output as `fb_top_plot`. You should also write `fb_top_plot` on its own line in the chunk to produce the actual plot. The key features of this graph that you should replicate for the autograder are:

- The barplot should have candidate last names on the y-axis and the `impressions_millions` variable from question 3 on the x-axis.
- The data feeding into the `ggplot` call should only have US House candidates and only the candidates with the highest 15 `impressions_millions` values.
- The y-axis should be ordered in ascending values of `impressions_millions` so that the lowest values are at the bottom. You may want to manipulate `cand_name_last` to achieve this.
- The fill color of the bar plot should be mapped to the `party` variable (but not globally!).

You do not need to exactly match the labels, but you should have informative labels. The color does not need to match, but if you want to change the fill colors, you can use the `scale_fill_manual(values = c(R = "red", D = "blue"))` function (where you can change the red and blue to whatever you want).
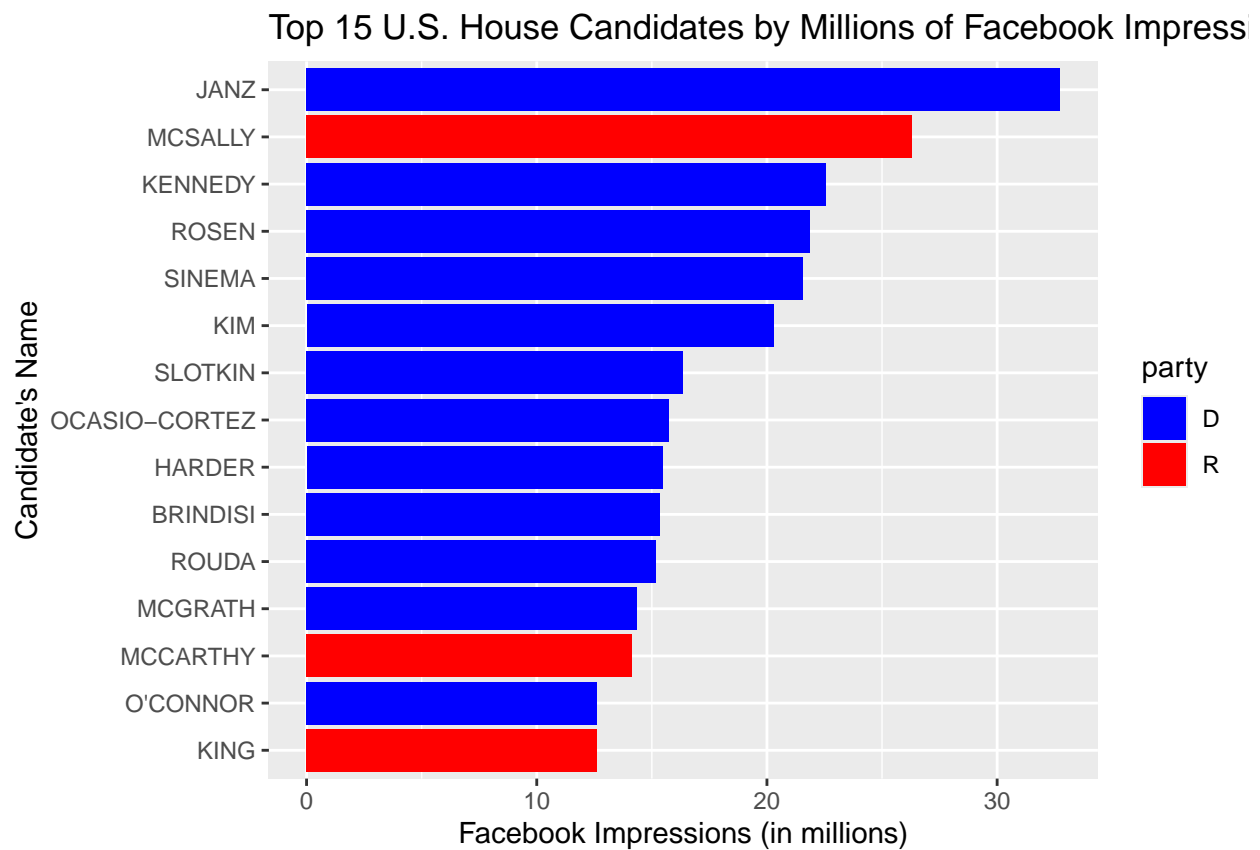
Rubric: 3pts for correct axes (autograder); 3pts for correct data fed into `ggplot` (autograder); 3pts for the correct ordering of the y-axis (PDF); 3pts for fill being mapped to `party` (autograder); 1pt for plot being in knitted output and having informative labels (PDF).

Answer 4

```
fb_top_plot <- fb_ads |>
    filter(office == "US House") |>
  arrange(desc(impressions_millions)) |>
  slice_max(impressions_millions / 1000000, n = 15)
```

```
fb_top_plot <- ggplot(fb_top_plot, mapping = aes(x = impressions_millions,
                                                 y = fct_reorder(cand_name_last,
                                                                 impressions_millions))) +
  geom_col(aes(fill = party)) +
  scale_fill_manual(values = c(R = "red", D = "blue")) +
  labs(title = "Top 15 U.S. House Candidates by Millions of Facebook Impressions",
       x = "Facebook Impressions (in millions)",
       y = "Candidate's Name")

fb_top_plot
```


Top 15 U.S. House Candidates by Millions of Facebook Impressions

##fin