Likert-Scale Survey Question Analysis Demonstration

Anthony Castellani

This is a demonstration of a basic analysis of the responses to one question of a hypothetical survey. This will show the use of R, as well as R packages `dplyr`, `ggplot2`, `gridExtra`, and `RColorBrewer`.

1. Load the packages.

```
library(dplyr)
library(ggplot2)
library(gridExtra)
library(RColorBrewer)
```

2. Use the `set.seed` function for repeatability.

```
set.seed(1)
```

3. Create a dummy data set. This data set will simulate 1,000 responses to five survey questions, with columns for Likert-scale responses (with a range of possibilities of 1 to 5), and the generic binary demographic attribute. The final product will be an object of `tbl` class (from the package `dplyr`).

```
demographic <- round(runif(n = 1000, min = 0, max = 1))
question.1 <- round(runif(n = 1000, min = 1, max = 5))
question.2 <- round(runif(n = 1000, min = 1, max = 5))
question.3 <- round(runif(n = 1000, min = 1, max = 5))
question.4 <- round(runif(n = 1000, min = 1, max = 5))
question.5 <- round(runif(n = 1000, min = 1, max = 5))

my.df <- data.frame(matrix(data = c(demographic, question.1,
                                    question.2, question.3,
                                    question.4, question.5), ncol = 6))

colnames(my.df) <- c("dem", "Q1", "Q2", "Q3", "Q4", "Q5")

my.tbl <- tbl_df(my.df)
```

- The first few rows of the data set look like this:

```
Source: local data frame [1,000 x 6]

     dem    Q1    Q2    Q3    Q4    Q5
   (dbl) (dbl) (dbl) (dbl) (dbl) (dbl)
1      0     4     4     4     2     2
2      1     1     1     3     5     4
3      1     1     2     2     5     2
4      1     4     3     2     4     5
5      1     2     2     1     3     5
6      1     3     4     2     3     3
7      0     3     4     5     3     1
8      1     2     2     4     5     5
9      1     2     5     5     5     5
10     0     4     5     5     5     2
..   ...   ...   ...   ...   ...   ...
```

4. Using the package `dplyr`, isolate the survey responses to one question, reduce the data to the number of responses per Likert-scale response option, and convert the data to a percentage of total responses.

```
select(my.tbl, Q1) %>%
  count(Q1) %>%
  mutate(n = (n / length(my.df$Q1)) * 100)
```

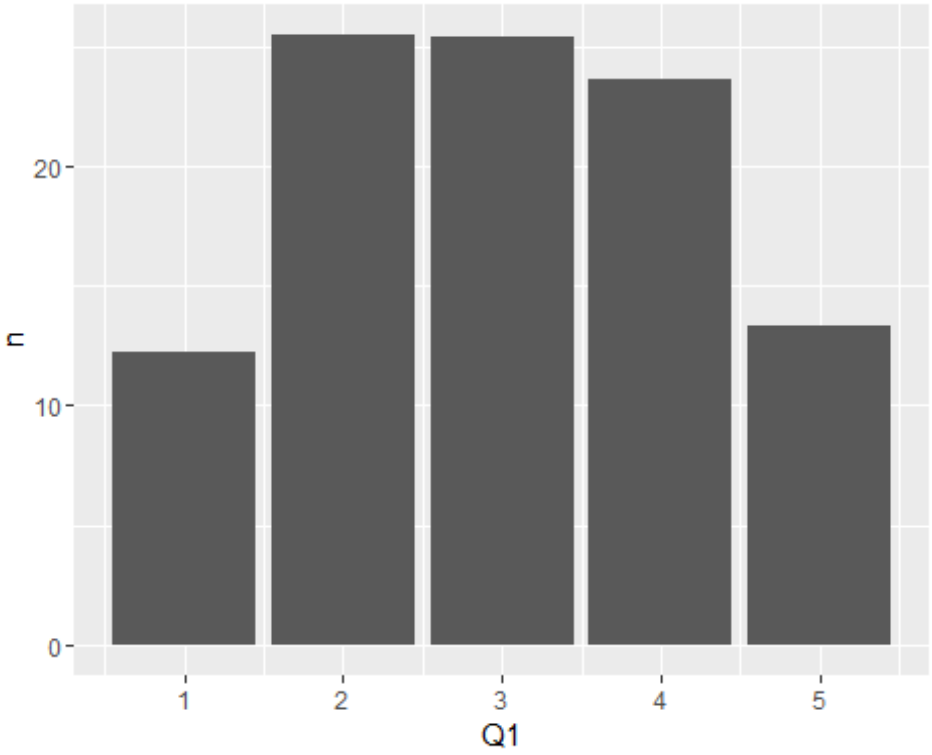
- The result will look like this:

```
Source: local data frame [5 x 2]

     Q1     n
  (dbl) (dbl)
1     1  12.2
2     2  25.5
3     3  25.4
4     4  23.6
5     5  13.3
```

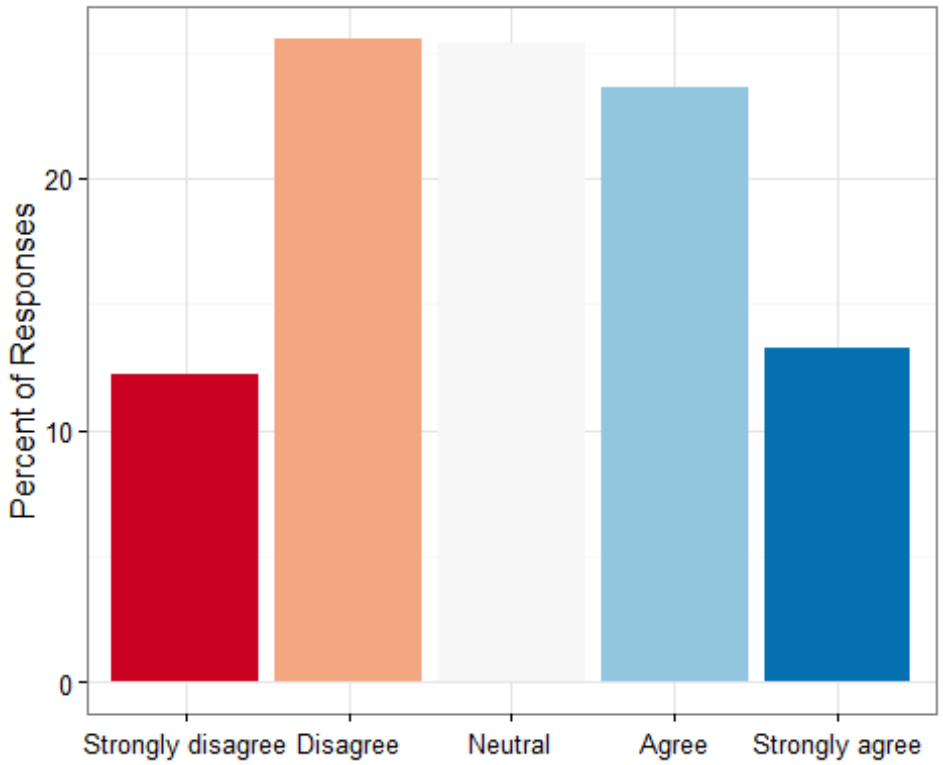
5. This outcome, produced with `dplyr`, dovetails nicely with `ggplot2`. By appending the end of the `dplyr` code with the piping operator (`%>%`), one can continue directly with plotting:

```
select(my.tbl, Q1) %>%
  count(Q1) %>%
  mutate(n = (n / length(my.df$Q1)) * 100) %>%
  ggplot(aes(x = Q1, y = n)) +
  geom_bar(stat = "identity")
```



6. While this does indeed produce a result, it's a little lacking in style. Plus, some of the annotations that carry over with no further modification don't tell much of a story (e.g., What does "n" mean?, etc.). With a few extra lines of code, it's trivial to add some color, modify the background, and give the axes some meaning. By making the assumption that this particular survey question had specific meanings attached to the Likert scale, the x-axis scale can also be meaningfully modified.

```
select(my.tbl, Q1) %>%
  count(Q1) %>%
  mutate(n = (n / length(my.df$Q1)) * 100) %>%
  ggplot(aes(x = Q1, y = n, fill = Q1)) +
  scale_fill_gradientn(colors = brewer.pal(n = 5, name = "RdBu")) +
  guides(fill = "none") +
  geom_bar(stat = "identity") +
  theme_bw() +
  xlab(NULL) +
  ylab("Percent of Responses") +
  scale_x_discrete(breaks = c("1", "2", "3", "4", "5"),
                   limits = c(1:5),
                   labels = c("Strongly disagree",
                              "Disagree",
                              "Neutral",
                              "Agree",
                              "Strongly agree"))
```



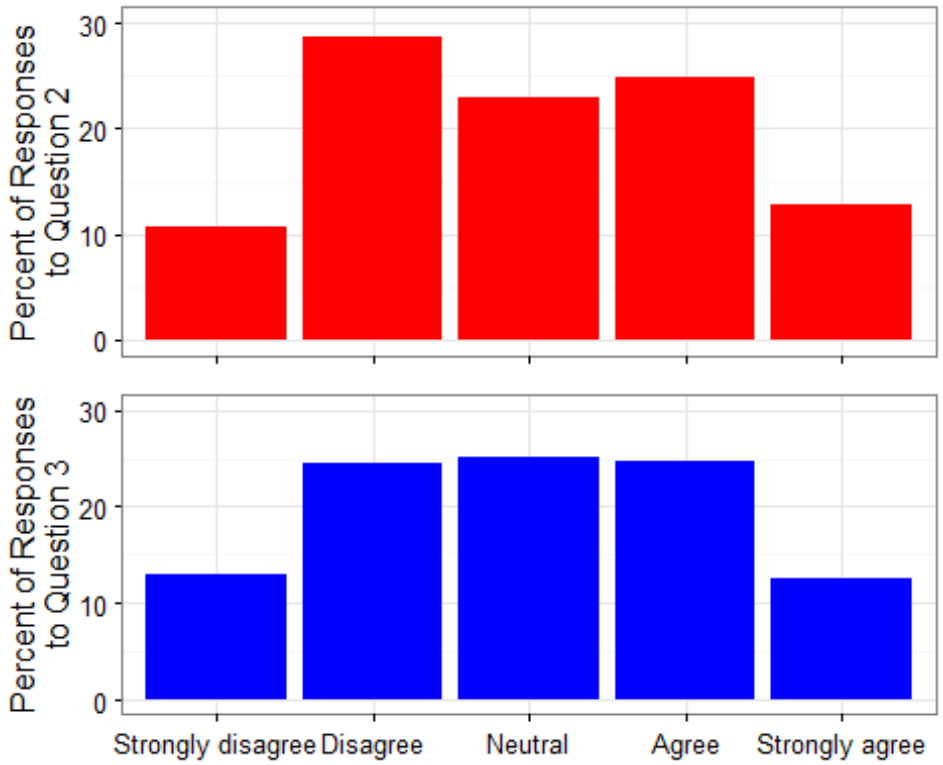
- For part two, let us assume that questions two and three are actually parts one and two of a two-part question. They should really be visualized together. There are a few ways of doing this, including using the faceting feature in `ggplot2`. Just to change things up, though, let's use the `gridExtra` package to arrange two separate visualizations adjacent to each other.

7. Because the function `grid.arrange` will plot two independent graphs, a little extra work is necessary to ensure they both display the same scale. The y-axis could be set manually, but setting it programmatically would be better. To do that, first we will need an object that captures the maximum value in each chart that will be used later.

```
my.max <- ceiling(max(
max(select(my.tbl, Q2) %>%
      count(Q2) %>%
      mutate(n = (n / length(my.df$Q2)) * 100))
,
max(select(my.tbl, Q3) %>%
      count(Q3) %>%
      mutate(n = (n / length(my.df$Q2)) * 100))
)) + 1
```

8. Plotting with `grid.arrange` is similar to the previous efforts, but with two blocks of `dplyr` and `ggplot2` code separated by a comma inside of `grid.arrange()`. In order to show some different color options, assume that the previous color scheme was found to be inappropriate, and so we'll use a simple red and blue scheme here. Aside from that, there are only a few little tweaks needed to do things like remove the x-axis labeling in the upper chart before we arrive at a side-by-side comparison of the two-part question.

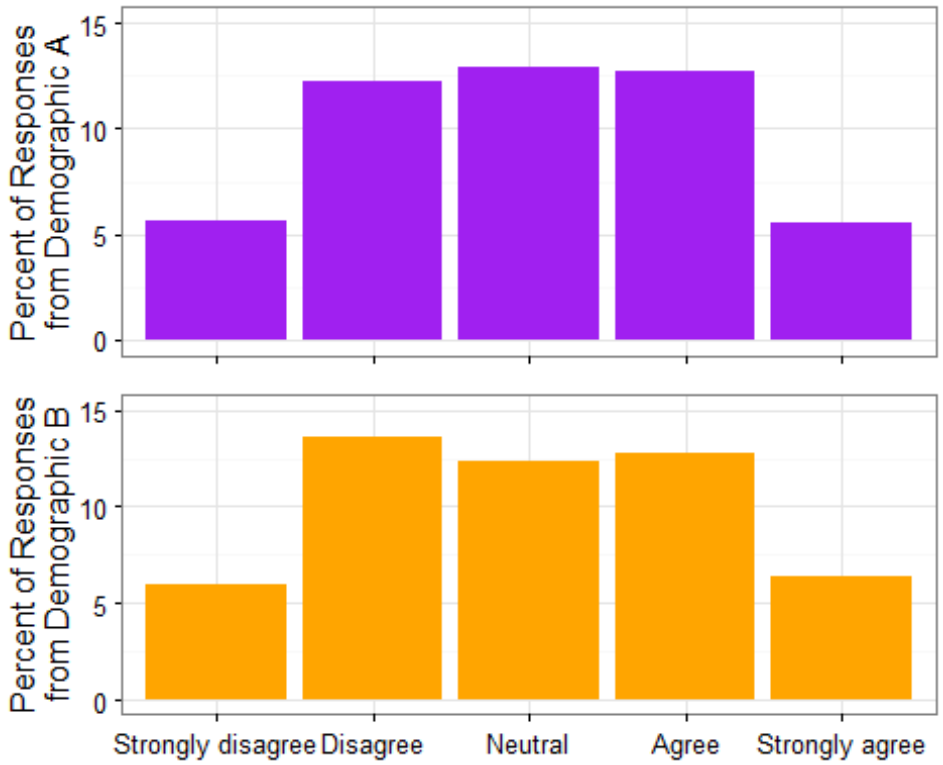
```
grid.arrange(
select(my.tbl, Q2) %>%
  count(Q2) %>%
  mutate(n = (n / length(my.df$Q2)) * 100) %>%
  ggplot(aes(x = Q2, y = n)) +
  geom_bar(stat = "identity", fill = "red") +
  theme_bw() +
  xlab(NULL) +
  ylab("Percent of Responses\nto Question 2") +
  scale_y_continuous(limits = c(0,my.max)) +
  scale_x_discrete(breaks = c("1", "2", "3", "4", "5"),
                   limits = c(1:5),
                   labels = NULL)
,
select(my.tbl, Q3) %>%
  count(Q3) %>%
  mutate(n = (n / length(my.df$Q3)) * 100) %>%
  ggplot(aes(x = Q3, y = n)) +
  geom_bar(stat = "identity", fill = "blue") +
  theme_bw() +
  xlab(NULL) +
  ylab("Percent of Responses\nto Question 3") +
  scale_y_continuous(limits = c(0,my.max)) +
  scale_x_discrete(breaks = c("1", "2", "3", "4", "5"),
                   limits = c(1:5),
                   labels = c("Strongly disagree",
                              "Disagree",
                              "Neutral",
                              "Agree",
                              "Strongly agree"))
)
```



9. For the last part of this demonstration, we will look at the effect that demographics has on one of the questions. Looking at question 4, let's find out the difference in opinion between demographic group A (which is represented by zeros in the demographic column of the data table) and demographic group B (which is represented by ones in the demographic column). This effort will be almost identical to the previous dual-graph visualization. The only difference will be the addition of the `dplyr` `filter` function at the beginning of each block of code inside of `grid.arrange`. Also, the color scheme will be adjusted to make this graph stand out from the previous ones.

```r
my.max2 <- ceiling(max(
  max(filter(my.tbl, dem == 0) %>%
        select(Q4) %>%
        count(Q4) %>%
        mutate(n = (n / length(my.df$Q4)) * 100))
  ,
  max(filter(my.tbl, dem == 1) %>%
        select(Q4) %>%
        count(Q4) %>%
        mutate(n = (n / length(my.df$Q4)) * 100))
)) + 1

grid.arrange(
filter(my.tbl, dem == 0) %>%
  select(Q4) %>%
  count(Q4) %>%
  mutate(n = (n / length(my.df$Q4)) * 100) %>%
  ggplot(aes(x = Q4, y = n)) +
  geom_bar(stat = "identity", fill = "purple") +
  theme_bw() +
  xlab(NULL) +
  ylab("Percent of Responses\nfrom Demographic A") +
  scale_y_continuous(limits = c(0,my.max2)) +
  scale_x_discrete(breaks = c("1", "2", "3", "4", "5"),
                   limits = c(1:5),
                   labels = NULL)
,
filter(my.tbl, dem == 1) %>%
  select(Q4) %>%
  count(Q4) %>%
  mutate(n = (n / length(my.df$Q4)) * 100) %>%
  ggplot(aes(x = Q4, y = n)) +
  geom_bar(stat = "identity", fill = "orange") +
  theme_bw() +
  xlab(NULL) +
  ylab("Percent of Responses\nfrom Demographic B") +
  scale_y_continuous(limits = c(0,my.max2)) +
  scale_x_discrete(breaks = c("1", "2", "3", "4", "5"),
                   limits = c(1:5),
                   labels = c("Strongly disagree",
                              "Disagree",
                              "Neutral",
                              "Agree",
                              "Strongly agree"))
)
```



And thus concludes this demonstration of a simple analysis of Likert-scale survey questions both with and without demographic breakdown.