# Telling Stories With Data: Visualizing Categorical Data in R

true

2021-12-07

## Contents

# 1 Introduction

Visualizing categorical data presents unique challenges. A common solution is a bar graph, which may often be the best data visualization solution.

However there are also some alternatives to bar graphs.

Below we present some options for bar graphs, and some possible alternative strategies.

Note that the outcomes–which you could think of as a *good* outcome, or a *bad* outcome, are unevenly distributed by group. Therefore, these data represent *inequities* or *disparities*.

## 2 Some Data

We create some simulated data with the `tribble` function. The data are created so that the two groups experience the outcomes unequally.

```
library(tibble) # rowise data frame (tibble) creation

library(tidyr) # data wrangling

mydata <- tribble(
  ~group, ~outcome, ~count,
  "Group A",   "beneficial outcome", 55,
  "Group A",   "undesirable outcome", 40,
  "Group B",   "beneficial outcome", 50,
  "Group B",   "undesirable outcome", 75
)

mydata$group <- factor(mydata$group) # data wrangling

mydata$outcome <- factor(mydata$outcome) # data wrangling

# duplicate the observations by count

mydata <- mydata %>% uncount(count)

pander(table(mydata)) # nice table of data
```

| | beneficial outcome | undesirable outcome |
|---|---|---|
| **Group A** | 55 | 40 |
| **Group B** | 50 | 75 |

## 3 Call The Graphing Library

```
library(ggplot2)
```

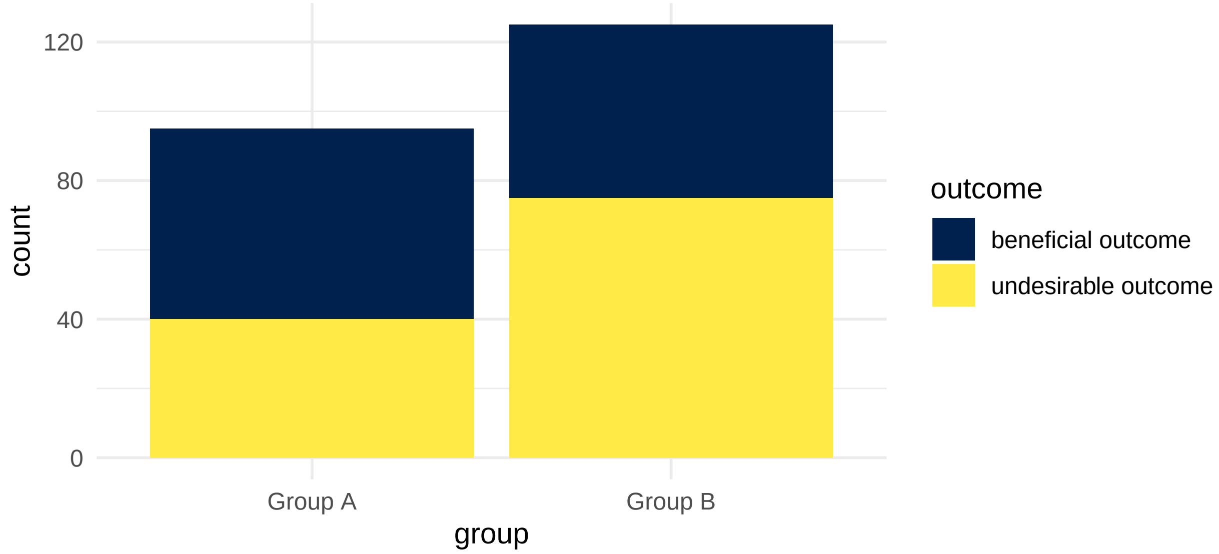
## 4 Bar Graphs

Bar graphs are often the simplest and best option for displaying categorical data. When used with an aesthetically pleasing color scheme, bar graphs can be an effective way of displaying data.

There are several different types of bar graph.

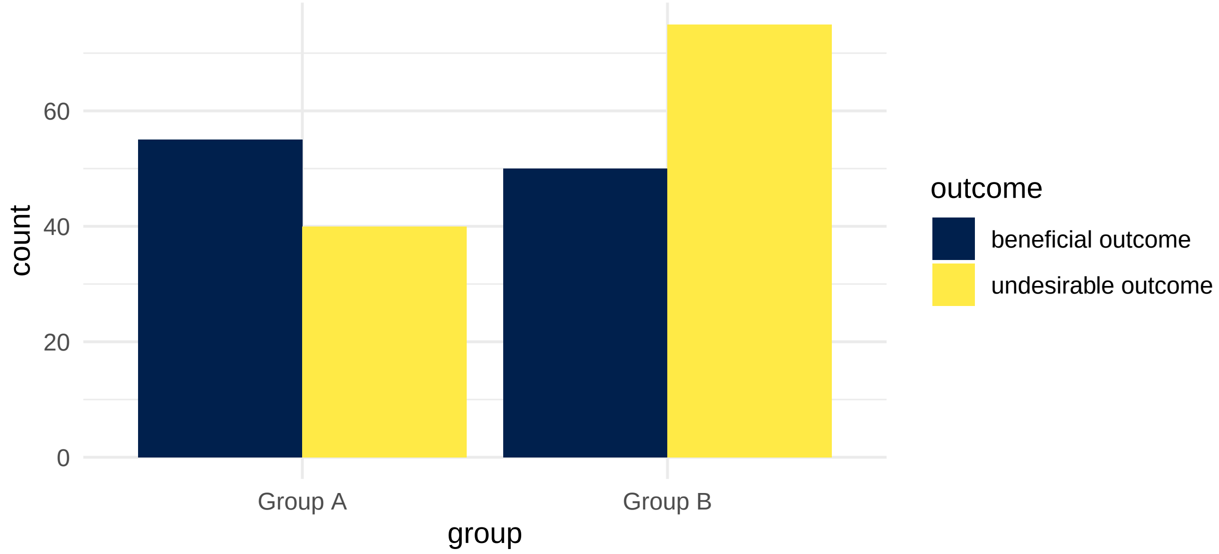
## 4.1 Stacked Bar Graph

```
ggplot(mydata, aes(x = group, # x is group
                   fill = outcome)) + # color fill is outcome
  geom_bar() + # bars
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_minimal() # nice theme
```



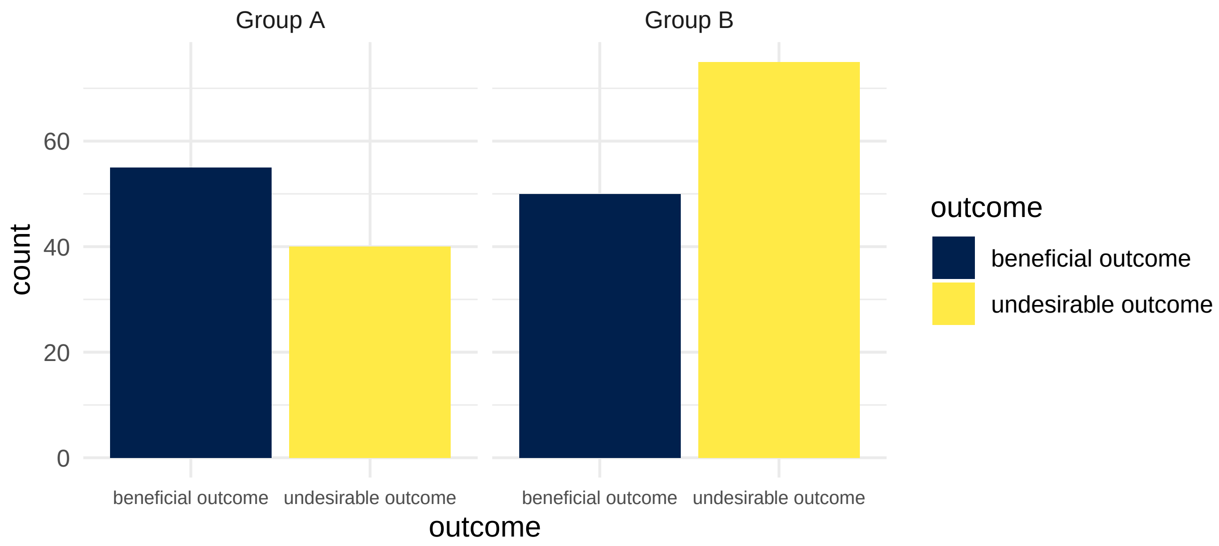
## 4.2 Unstacked Bar Graph

```
ggplot(mydata, aes(x = group, # x is group
                   fill = outcome)) + # color fill is outcome
  geom_bar(position = position_dodge()) + # "dodged" bars
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_minimal() # nice theme
```



### 4.3 Faceted Bar Graph

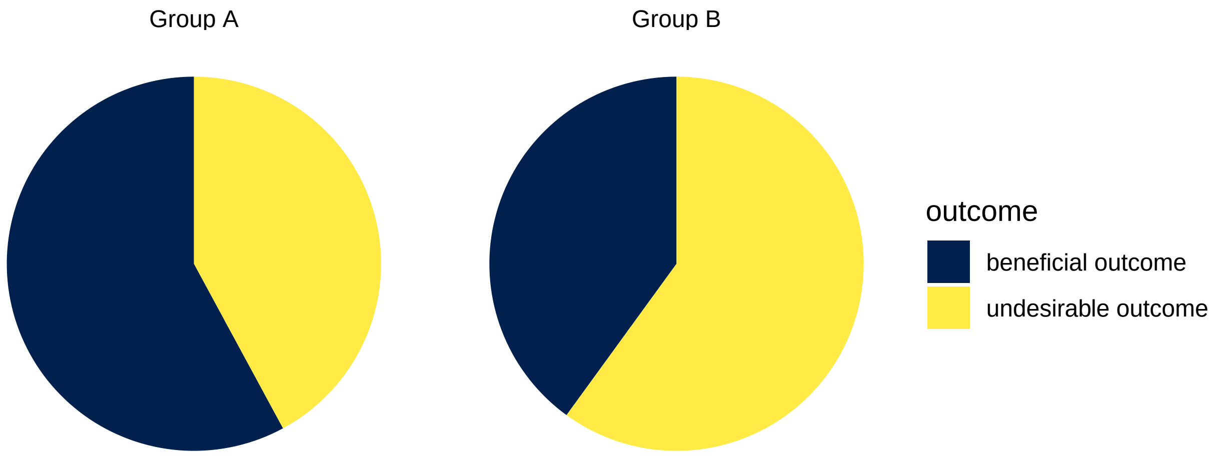
```
ggplot(mydata, aes(x = outcome, # x is outcome
                   fill = outcome)) + # color fill is outcome
  geom_bar() + # bars
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_minimal() + # nice theme
  theme(axis.text.x = element_text(size = rel(.75))) + # smaller x axis text
  facet_wrap(~group) # facet on group
```



# 5 Pie Chart

In `ggplot` terms, pie charts are bar graphs displayed with *polar coordinates*.

```
ggplot(mydata, aes(x = 1, # x is always 1
                   fill = outcome)) + # color fill is outcome
  geom_bar(position = "fill") + # bars
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_void() + # void theme for pie charts
  coord_polar(theta = "y") + # polar coordinates
  facet_wrap(~group) # facet on group
```

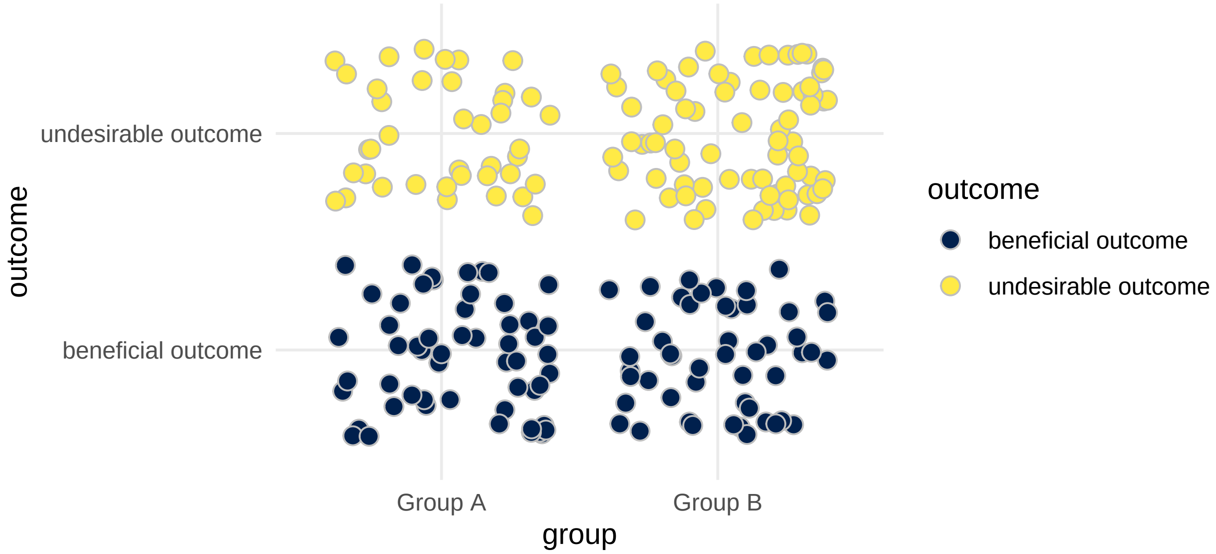


## 6 Jittered Points

*Jittered points* may be a good choice because every point represents an individual in the data set. However, it may be difficult to draw exact conclusions from jittered points.

Jittered points *may* (or may not) benefit from having an outline in a different color to make them more distinct.

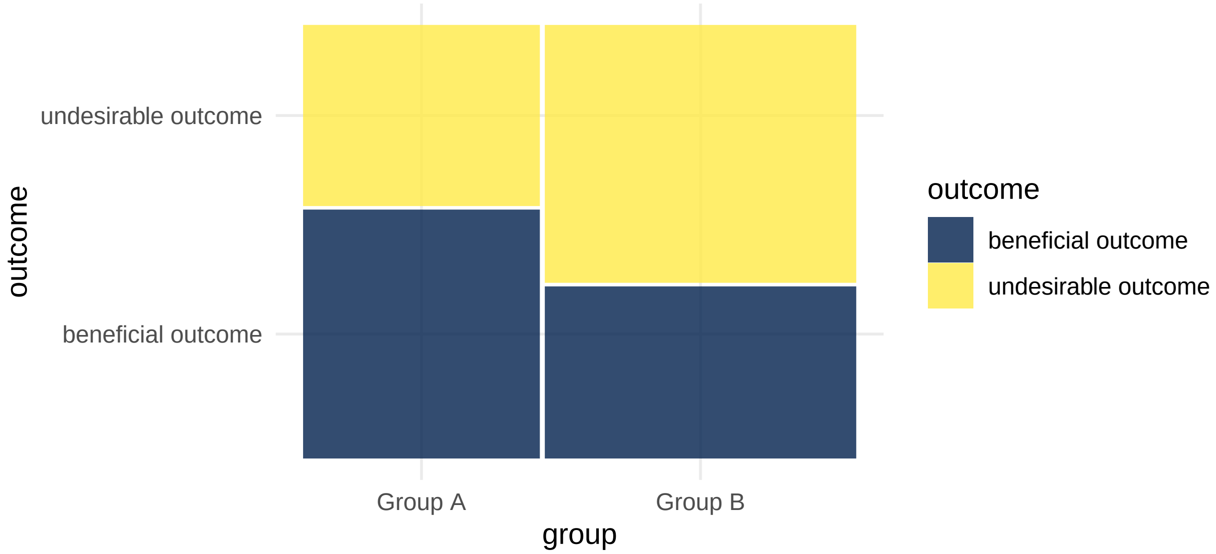
```
ggplot(mydata, aes(x = group, # x is group
                   fill = outcome,
                   y = outcome)) + # color fill is outcome
  geom_jitter(size = 3, # jittered points
              pch = 21, # Point Character 21; filled points
              color = "grey") + # outline color
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_minimal() # nice theme
```



## 7 Mosaic Plot

Mosaic plots are another way to display data. They are especially effective for being clear about the relative membership in different groups, and about the proportion of each group experiencing each outcome.

```
library(ggmosaic) # mosaic plots

ggplot(mydata) +
  geom_mosaic(aes(x = product(group), # "mosaic" geometry
                  fill = outcome)) +
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_minimal()  # nice theme
```



# 8 Waffle Plot

Lastly, waffle plots may be a useful way to display information. Waffle plots are aesthetically appealing. The aesthetic appeal of a waffle plot may, however, obscure the fact that they may not provide the clearest presentation of quantitative information. Waffle plots work best when the sample size is several hundred or fewer.

Waffle plots require some data wrangling.

## 8.1 Call The Libraries

```
library(waffle) # waffle geometry

library(dplyr) # data wrangling
```

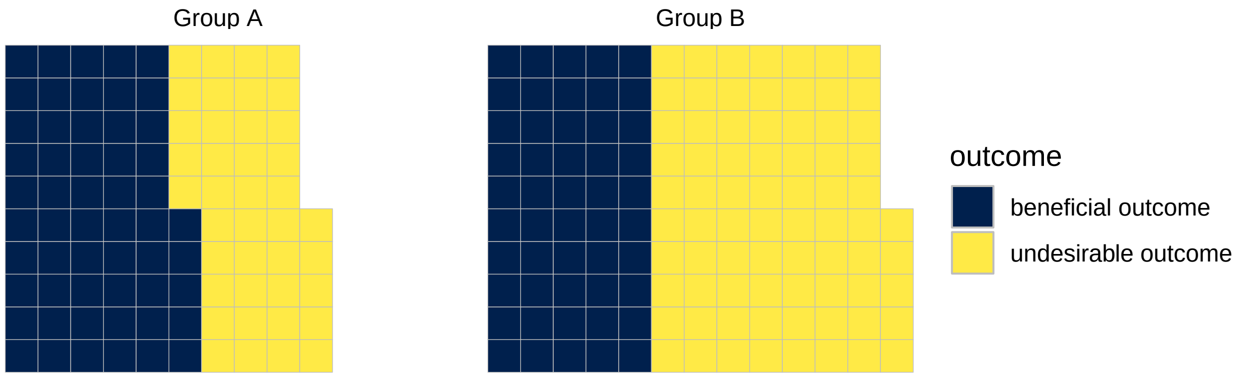
## 8.2 Make A Data Set Of Counts

```
# make a data set of counts

mycounts <- mydata %>%
  group_by(group, outcome) %>% # group by group & outcome
  tally() # count up observations

pander(mycounts) # replay this data
```

| group | outcome | n |
|---|---|---|
| Group A | beneficial outcome | 55 |
| Group A | undesirable outcome | 40 |
| Group B | beneficial outcome | 50 |
| Group B | undesirable outcome | 75 |

## 8.3 Make The Waffle Plot

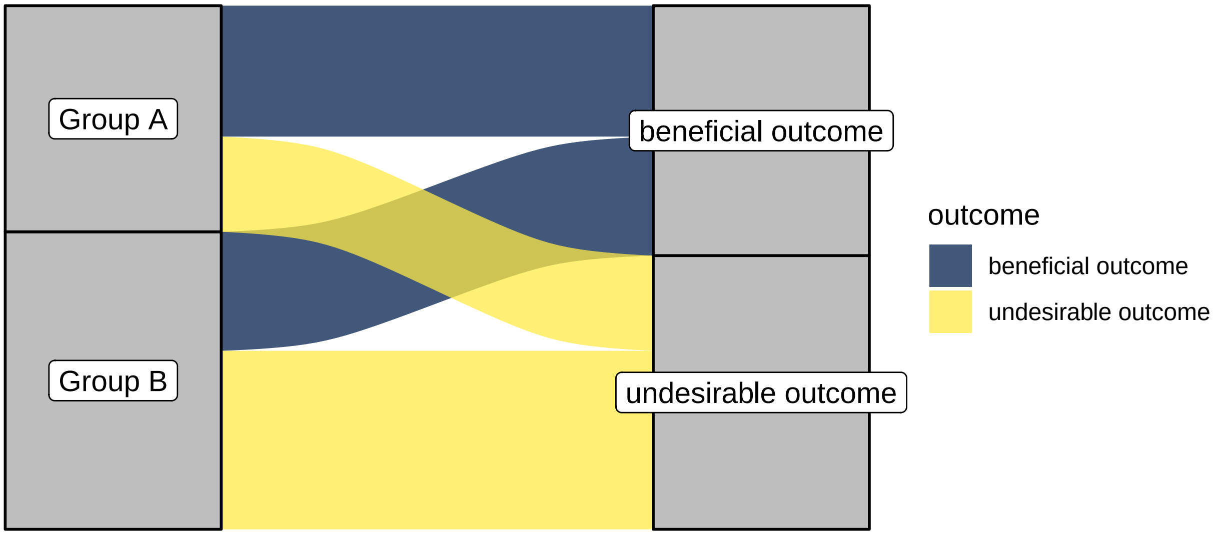
```
# use geom_waffle with this data set of counts

ggplot(mycounts, # use this new data
       aes(fill = outcome, # color fill is outcome
           values = n)) + # values are n
  geom_waffle(color = "grey") + # waffle geometry w/ grey separator
  facet_wrap(~group) + # facet on group
  coord_equal() + # squares!
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_void()  # nice theme
```



# 9 Alluvial Diagram

Lastly, an alluvial diagram may be useful to illustrate a *flow* from one status to another. We will use the data set of `mycounts` that we generated above.

```
library(ggalluvial)

ggplot(mycounts,
       aes(y = n,
           axis1 = group,
           axis2 = outcome)) +
  geom_alluvium(aes(fill = outcome), # alluvia; flows
                alpha = .75) +
  geom_stratum(width = 1/3, # end "strata"
               color = "black", # outline color
               fill = "grey",
               color = "grey") +
  geom_label(stat = "stratum", # textual labels
             aes(label = after_stat(stratum))) +
  scale_fill_viridis_d(option = "cividis") + # beautiful colors
  theme_void()  # nice theme
```


Group A
Group B
beneficial outcome
undesirable outcome
outcome
beneficial outcome
undesirable outcome