SMI606: Week 3
Relationships between variables

Dr. Calum Webb
Sheffield Methods Institute, the University of Sheffield.
c.j.webb@sheffield.ac.uk (mailto:c.j.webb@sheffield.ac.uk)

Sign In

Learning Objectives

What will I learn? (?panelset=what-will-i-learn%3F#panelset_what-will-i-learn%3F)

How does this week fit into my course? (?panelset=how-does-this-week-fit-into-my-course%3F#panelset_how-does-this-week-fit-into-my-course%3F)

By the end of this week you will:

- Learn how to **use variable types to select appropriate bivariate data visualisations and descriptive statistics**.
- Be able to create and interpret **bivariate bar charts, heatmaps, boxplots, ridgeplots, scatterplots, and hexbin plots** using `R`.
- Be able to calculate and interpret several **bivariate descriptive statistics including contingency tables, Cramer's V, Mean/Median differences, Spearman's ρ, and Pearson's R** in `R`.

Learning Objectives

What will I learn? (?panelset=what-will-i-learn%3F#panelset_what-will-i-learn%3F)

How does this week fit into my course? (?panelset=how-does-this-week-fit-into-my-course%3F#panelset_how-does-this-week-fit-into-my-course%3F)

- Bivariate statistics an be important parts of research studies themselves, especially research studies with *a priori* controls (e.g. randomised controlled trials).
- Bivariate statistics and data visualisations are important for checking assumptions and for communicating findings in research.
- Bivariate statistics can also be used to develop the rationale behind case studies in mixed methods research.

Visualisations for Exploring Relationships (Dependence) Between Variables

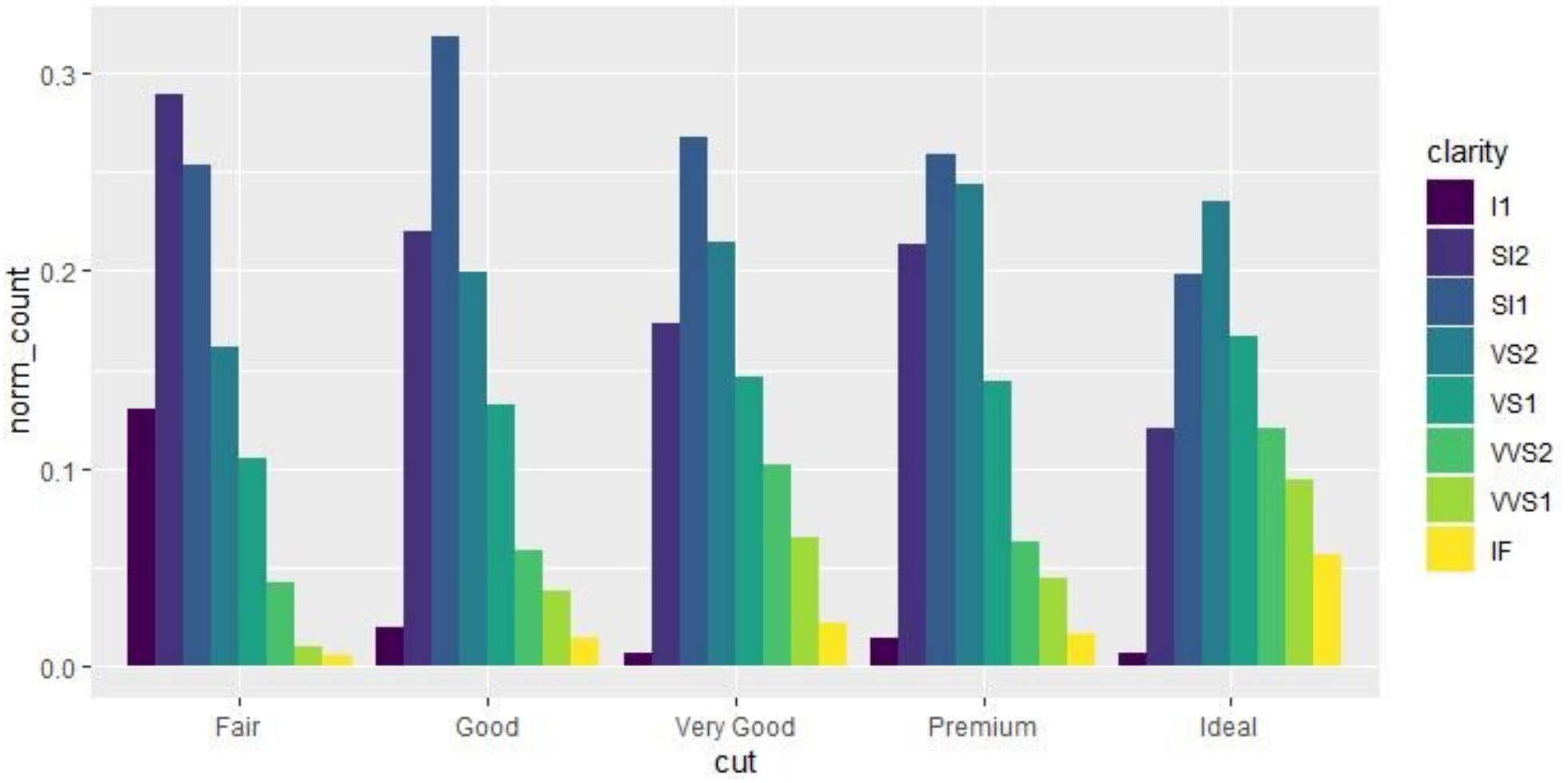
Variable Type	Nominal	Ordinal	Continuous
Nominal	Heatmap/ Bivariate Bar Chart		
Ordinal	Heatmap/ Bivariate Bar Chart	Heatmap/ Bivariate Bar Chart	
Continuous	Boxplot/ Ridgeplot	Boxplot/ Ridgeplot	Scatterplot/ Hex Bin Plot

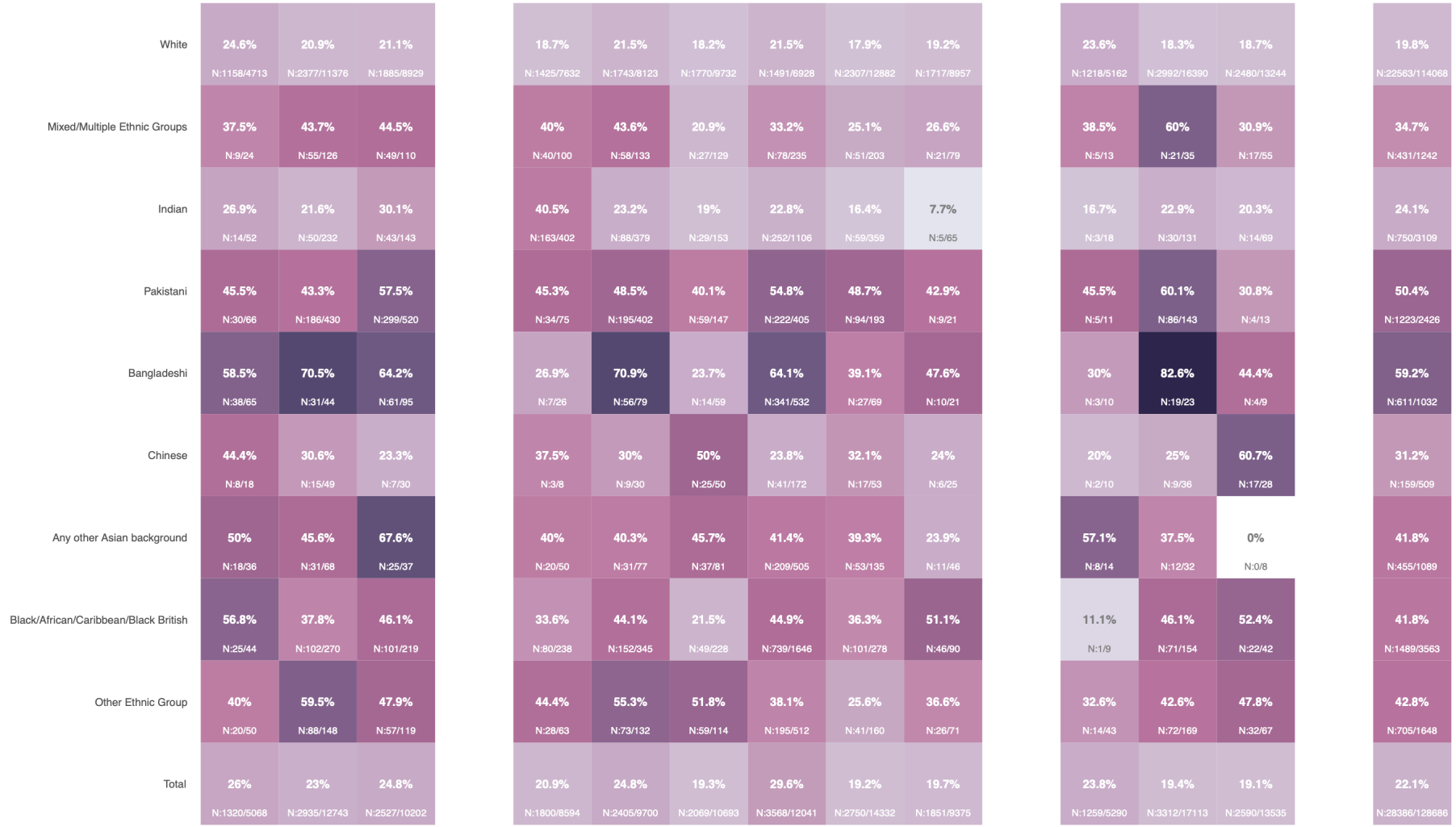
Descriptive Statistics for Describing Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Contingency Table + Cramer's V		
Ordinal	Contingency Table + Cramer's V	Spearman's Rho/ Contingency Table	
Continuous	Mean/ Median Difference	Mean/ Median Difference	Pearson's R or Spearman's Rho

Visualisations for Exploring Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	**Heatmap/ Bivariate Bar Chart**		
Ordinal	**Heatmap/ Bivariate Bar Chart**	**Heatmap/ Bivariate Bar Chart**	
Continuous	Boxplot/ Ridgeplot	Boxplot/ Ridgeplot	Scatterplot/ Hex Bin Plot


0.3
0.2
0.1
0.0
norm_count
Fair
Good
Very Good
Premium
Ideal
cut
clarity
I1
SI2
SI1
VS2
VS1
VVS2
VVS1
IF

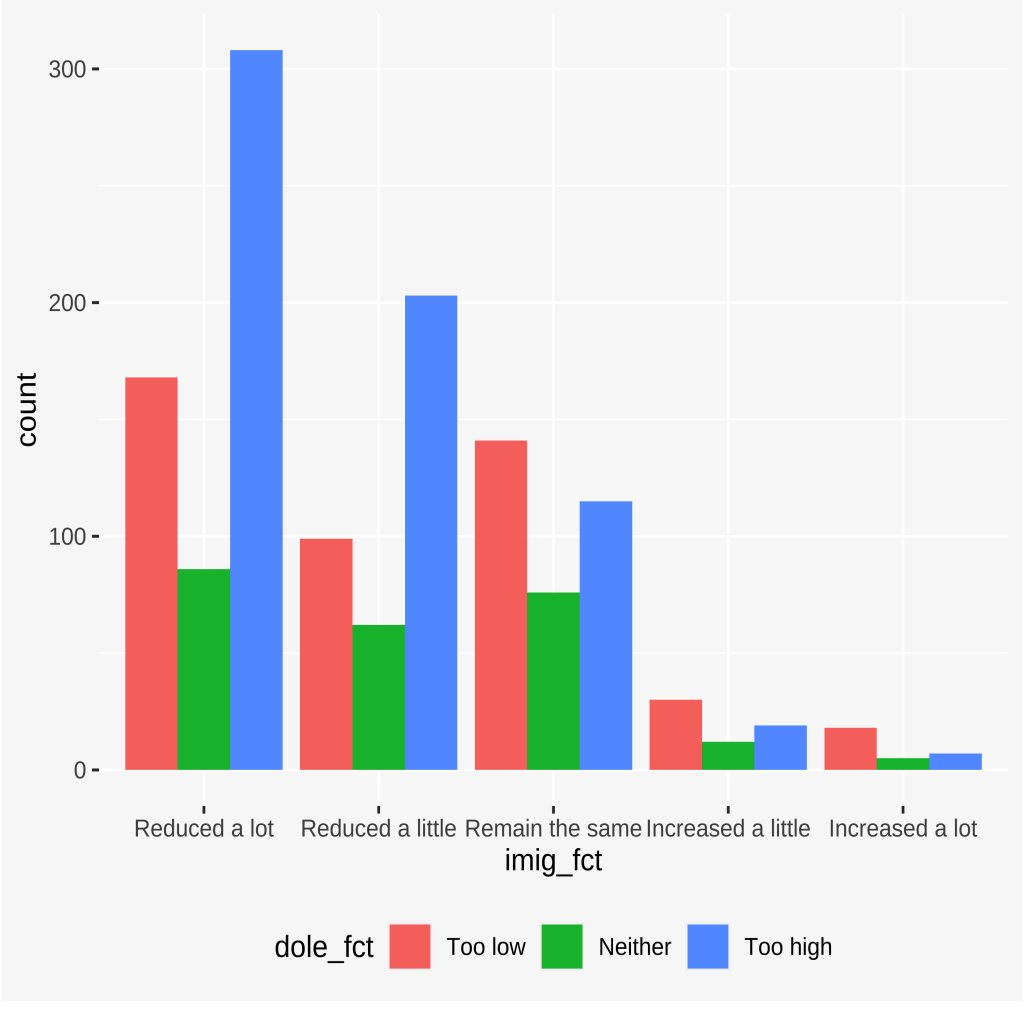


N = Number of children in sample living in poverty / Total number of children in sample.

Is there a relationship between peoples' attitudes towards the adequacy of benefits (welfare) payments and their attitudes towards immigration in Scotland?

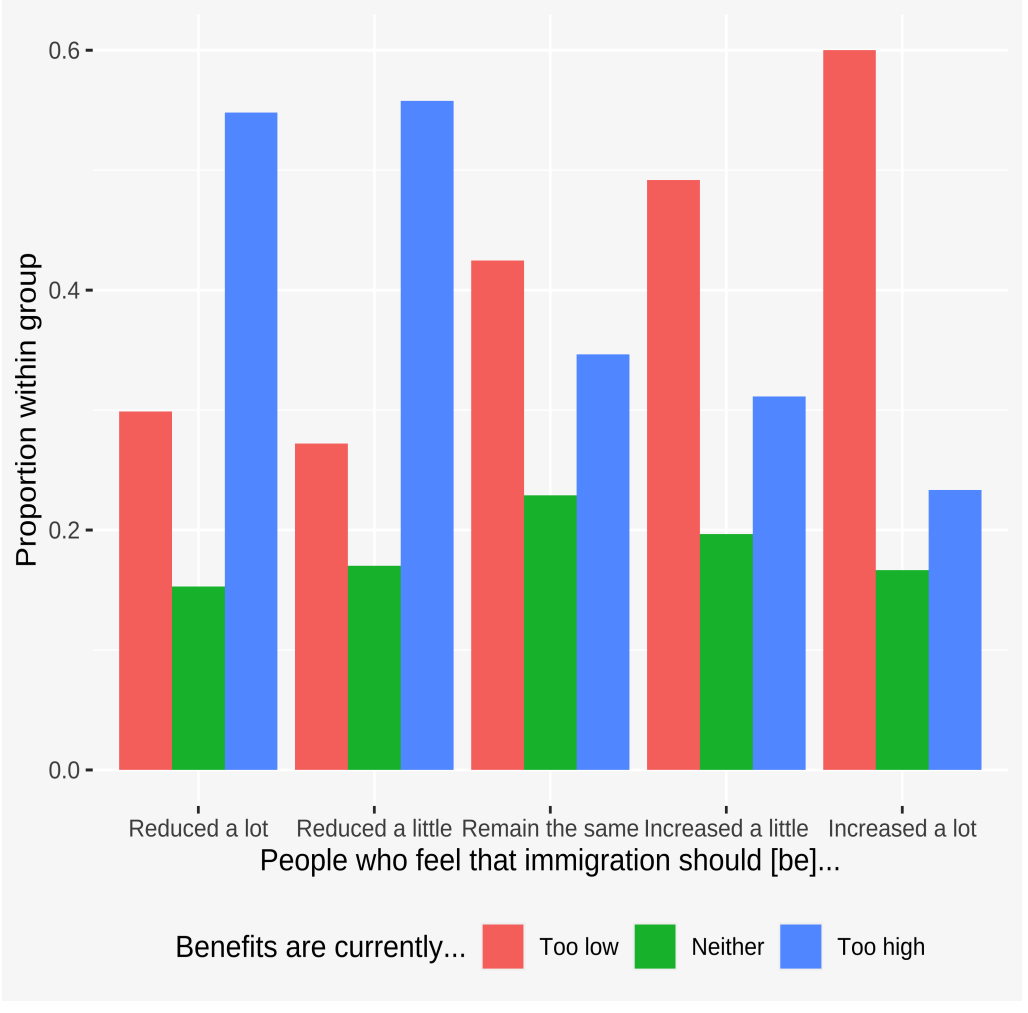
Grouped bar chart

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  # Plot bar chart
  ggplot() +
  geom_bar(aes(x = imig_fct, fill = dole_fct),
           position = "dodge") +
  theme(legend.position = "bottom")

# imig_fct = views on how immigration should change

# dole_fct = views on whether benefits are too high, too low,
or neither
```



Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row") %>%
  pivot_longer(-1) %>%
  mutate(name = factor(name,
                       levels = c("Too low", "Neither", "Too h
igh"))
         ) %>%
  # Plot bar chart
  ggplot() +
  geom_col(aes(x = imig_fct, y = value, fill = name),
           position = "dodge") +
  theme(legend.position = "bottom") +
  ylab("Proportion within group") +
  xlab("People who feel that immigration should [be]...") +
  labs(fill = "Benefits are currently...")
```



Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct)
```

```
##            imig_fct Too low Neither Too high
##       Reduced a lot     168      86      308
##    Reduced a little      99      62      203
##     Remain the same     141      76      115
##  Increased a little      30      12       19
##     Increased a lot      18       5        7
```

Filter out the missing values and then create a **two-way contingency table** using `janitor`, which shows the **frequency** of responses in every combination of categories across the two variables.

Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row")
```

```
##            imig_fct   Too low   Neither  Too high
##       Reduced a lot 0.2989324 0.1530249 0.5480427
##    Reduced a little 0.2719780 0.1703297 0.5576923
##     Remain the same 0.4246988 0.2289157 0.3463855
##  Increased a little 0.4918033 0.1967213 0.3114754
##     Increased a lot 0.6000000 0.1666667 0.2333333
```

Now, we use the `adorn_percentages()` function from the `janitor` package to convert these responses into **proportions within each row**.

For example, the first row would read as: "For all of the survey respondents who said imigration should be reduced a lot, 30.4% said that benefits were too low, 14.5% said that benefits were neither too high nor too low, and 54.9% said that benefits were too high".

The percentage here is based on the **row total**, because we specified "row" in the `adorn_percentages` function.

Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row") %>%
  pivot_longer(-1)
```

To turn this table into something ggplot can plot as a bar chart, we need to do a kind of confusing transformation.

We want all of the categories from our two variables to be in their own columns, with every valid combination, and then all of our proportions to be in their own column. For now, don't worry about what is going on with `pivot_longer`, just know that we can get from a janitor contingency tabyl into a structure we can plot using `pivot_longer(1)`.

```
## # A tibble: 15 × 3
##    imig_fct           name     value
##    <fct>              <chr>    <dbl>
##  1 Reduced a lot      Too low  0.299
##  2 Reduced a lot      Neither  0.153
##  3 Reduced a lot      Too high 0.548
##  4 Reduced a little   Too low  0.272
##  5 Reduced a little   Neither  0.170
##  6 Reduced a little   Too high 0.558
##  7 Remain the same    Too low  0.425
##  8 Remain the same    Neither  0.229
##  9 Remain the same    Too high 0.346
## 10 Increased a little Too low  0.492
## 11 Increased a little Neither  0.197
## 12 Increased a little Too high 0.311
## 13 Increased a lot    Too low  0.6
## 14 Increased a lot    Neither  0.167
## 15 Increased a lot    Too high 0.233
```

Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row") %>%
  pivot_longer(-1) %>%
  mutate(name = factor(name,
     levels = c("Too low", "Neither", "Too high"))
       )
```

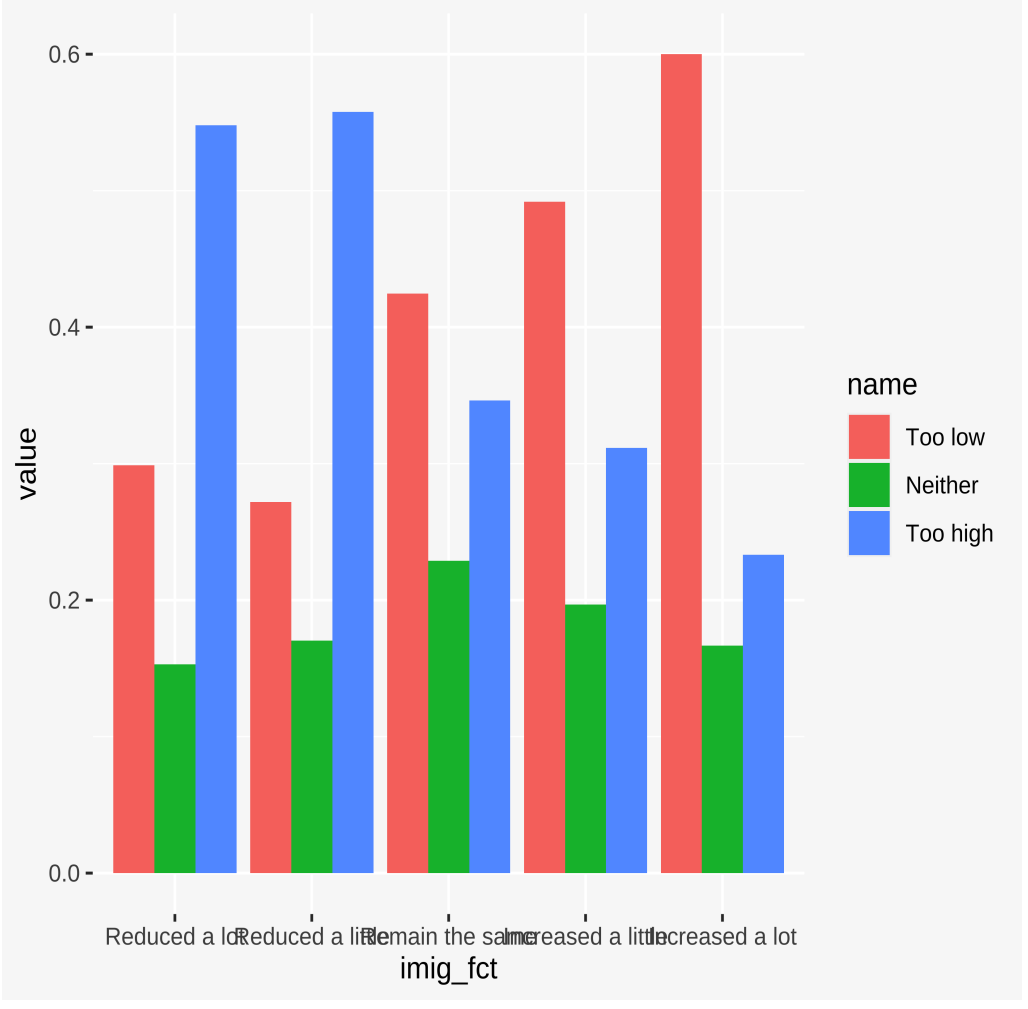
The last thing we do before plotting is reset the benefits column (now called `name`, thanks to `pivot_longer`) back to a factor type with the levels in the correct order ("Too low", "Neither", "Too high"). This will help ggplot arrange them in an intuitive way.

```
## # A tibble: 15 × 3
##    imig_fct           name     value
##    <fct>              <fct>    <dbl>
##  1 Reduced a lot      Too low  0.299
##  2 Reduced a lot      Neither  0.153
##  3 Reduced a lot      Too high 0.548
##  4 Reduced a little   Too low  0.272
##  5 Reduced a little   Neither  0.170
##  6 Reduced a little   Too high 0.558
##  7 Remain the same    Too low  0.425
##  8 Remain the same    Neither  0.229
##  9 Remain the same    Too high 0.346
## 10 Increased a little Too low  0.492
## 11 Increased a little Neither  0.197
## 12 Increased a little Too high 0.311
## 13 Increased a lot    Too low  0.6
## 14 Increased a lot    Neither  0.167
## 15 Increased a lot    Too high 0.233
```

Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row") %>%
  pivot_longer(-1) %>%
  mutate(name = factor(name,
     levels = c("Too low", "Neither", "Too high"))
        ) %>%
  # Plot bar chart
   ggplot() +
   geom_col(aes(x = imig_fct, y = value, fill = name),
            position = "dodge")
```

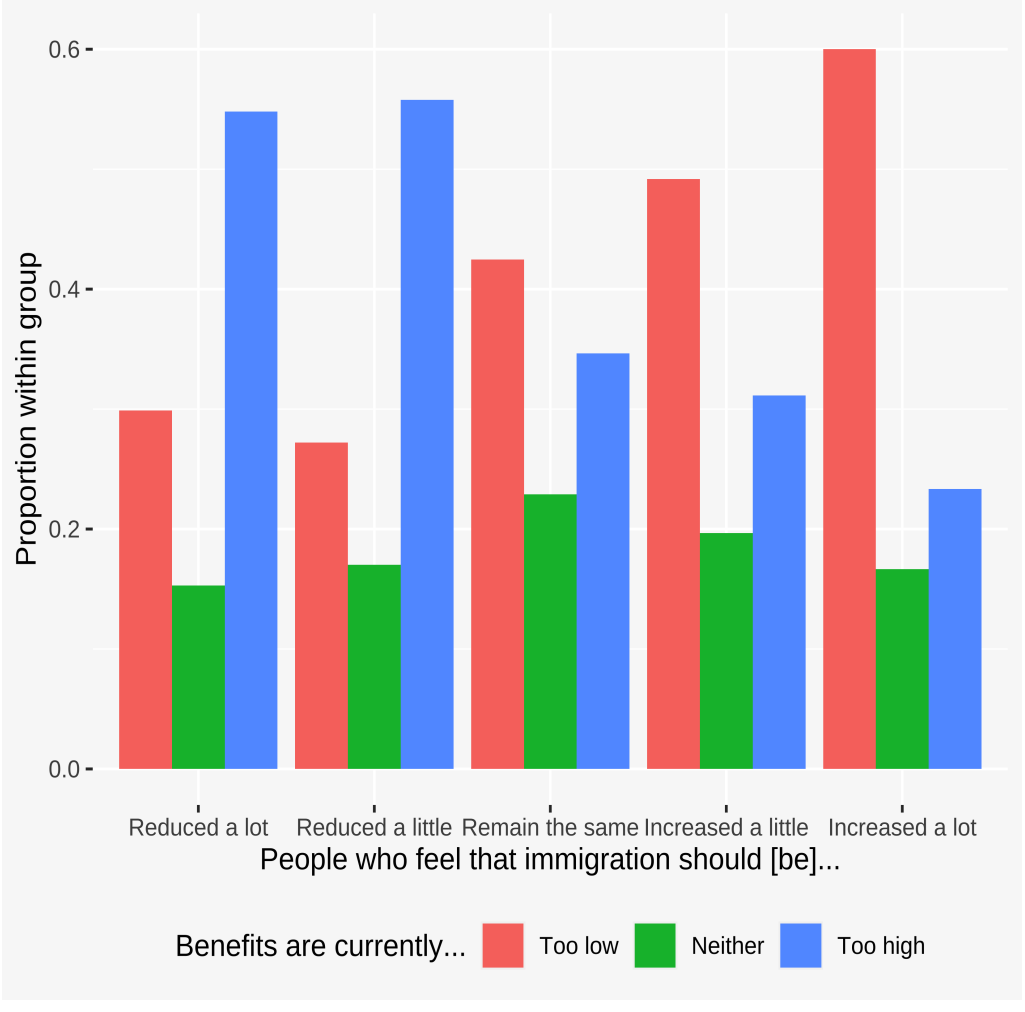
Now we can plot the results. Note that with this kind of data format, where ggplot isn't doing any processing of statistics itself (we've already given it the proportions and are just telling them what they are), we switch to `geom_col` rather than using `geom_bar`.



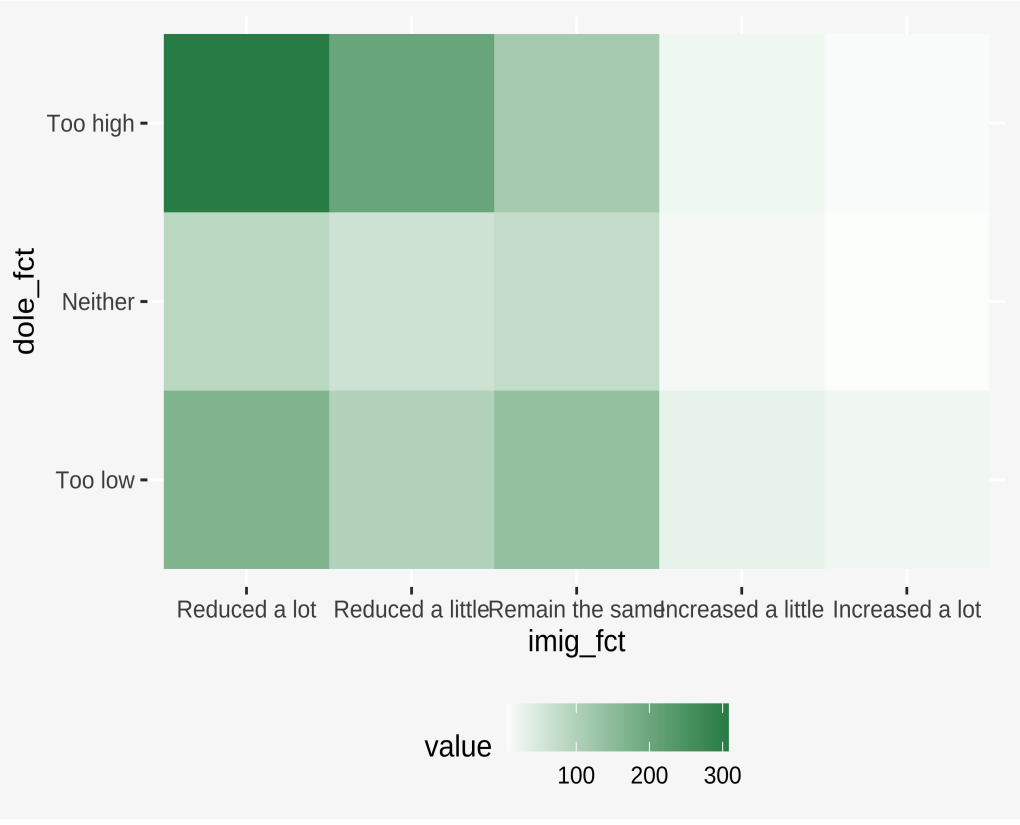
Grouped bar chart (Within Group %)

```
ssa %>%
  # Remove missing
  filter(!is.na(imig_fct) & !is.na(dole_fct)
    ) %>%
  janitor:: tabyl(imig_fct, dole_fct) %>%
  janitor::adorn_percentages("row") %>%
  pivot_longer(-1) %>%
  mutate(name = factor(name,
     levels = c("Too low", "Neither", "Too high"))
        ) %>%
  # Plot bar chart
   ggplot() +
   geom_col(aes(x = imig_fct, y = value, fill = name),
            position = "dodge") +
   theme(legend.position = "bottom") +
   ylab("Proportion within group") +
   xlab("People who feel that immigration should [be]...") +
   labs(fill = "Benefits are currently...")
```

Finally, we can add some custom labels and formatting changes to make our plot look nicer and make it so that anyone can interpret it.



Heatmap




```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>%
  pivot_longer(-1, names_to = "dole_fct") %>%
  mutate(
    dole_fct = factor(dole_fct,
                      levels = c("Too low", "Neither", "Too hi
gh"))
  ) %>%
  ggplot() +
  geom_tile(aes(x = imig_fct, y = dole_fct, fill = value)) +
  theme(legend.position = "bottom") +
  coord_fixed(1) +
  scale_fill_gradient2(low = "white", high = "seagreen")
```

Heatmap

```
##            imig_fct Too low Neither Too high
##       Reduced a lot     168      86      308
##    Reduced a little      99      62      203
##     Remain the same     141      76      115
##  Increased a little      30      12       19
##     Increased a lot      18       5        7
```

```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) # Get crosstab
```

Filter the missing data out of the dataset using `filter()` and then create a contingency table using `janitor::tabyl()`.

Heatmap

```
## # A tibble: 15 × 3
##    imig_fct           dole_fct value
##    <fct>              <chr>    <dbl>
##  1 Reduced a lot      Too low    168
##  2 Reduced a lot      Neither     86
##  3 Reduced a lot      Too high   308
##  4 Reduced a little   Too low     99
##  5 Reduced a little   Neither     62
##  6 Reduced a little   Too high   203
##  7 Remain the same    Too low    141
##  8 Remain the same    Neither     76
##  9 Remain the same    Too high   115
## 10 Increased a little Too low     30
## 11 Increased a little Neither     12
## 12 Increased a little Too high    19
## 13 Increased a lot    Too low     18
## 14 Increased a lot    Neither      5
## 15 Increased a lot    Too high     7
```

```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>% # Get crosstab
  pivot_longer(-1, names_to = "dole_fct") # Create 'long' data
```

Here I've used the `pivot_longer` trick again, but I've also used the optional argument `names_to` to make the names of the columns be placed in a new column called "dole_fct", which makes the plotting a little clearer.

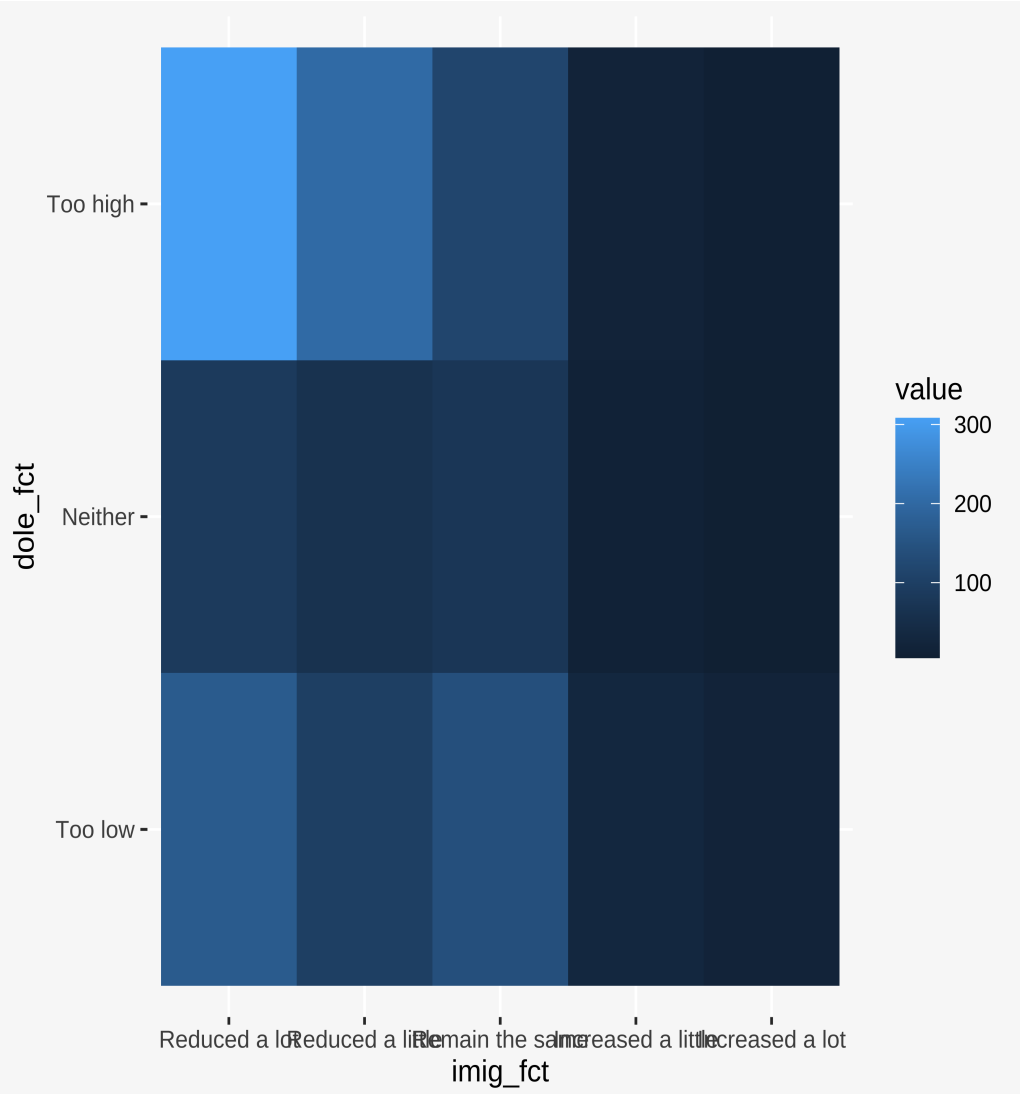
Heatmap

```
## # A tibble: 15 × 3
##    imig_fct           dole_fct value
##    <fct>              <fct>    <dbl>
##  1 Reduced a lot      Too low    168
##  2 Reduced a lot      Neither     86
##  3 Reduced a lot      Too high   308
##  4 Reduced a little   Too low     99
##  5 Reduced a little   Neither     62
##  6 Reduced a little   Too high   203
##  7 Remain the same    Too low    141
##  8 Remain the same    Neither     76
##  9 Remain the same    Too high   115
## 10 Increased a little Too low     30
## 11 Increased a little Neither     12
## 12 Increased a little Too high    19
## 13 Increased a lot    Too low     18
## 14 Increased a lot    Neither      5
## 15 Increased a lot    Too high     7
```

```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>% # Get crosstab
  as_tibble() %>% # Convert to tibble
  pivot_longer(-1, names_to = "dole_fct") %>% # Create 'long'
data
  mutate( # re-factorise dole_fct
    dole_fct = factor(dole_fct, levels = c("Too low", "Neithe
r", "Too high"))
  )
```

Here, I am mutating the `dole_fct` variable to put the levels in the correct order (because it is an ordinal variable).

Heatmap

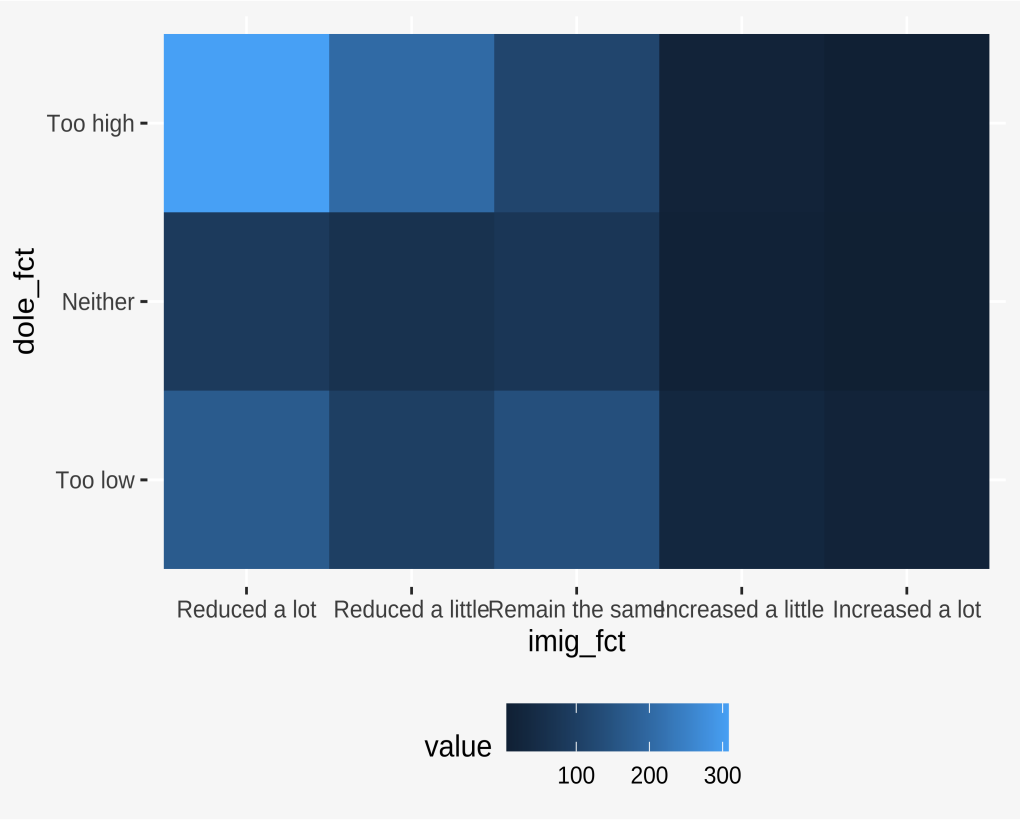



```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>% # Get crosstab
  as_tibble() %>% # Convert to tibble
  pivot_longer(-1, names_to = "dole_fct") %>% # Create 'long'
data
  mutate( # re-factorise dole_fct
    dole_fct = factor(dole_fct, levels = c("Too low", "Neithe
r", "Too high"))
  ) %>%
  ggplot() + # plot data
  geom_tile(aes(x = imig_fct, y = dole_fct, fill = value)) # c
reate heatmap
```

And now we can use `geom_tile` to create a heatmap.

`geom_tile` essentially just draws squares on a grid, using the x and y variables as guides for locations. The `fill` optional argument can be used colour the tiles according to their value (here we are using frequencies, but we could have added the intermediate `adorn_percentages("row")` or `adorn_percentages("col")` function from `janitor` to make these row or column proportions)

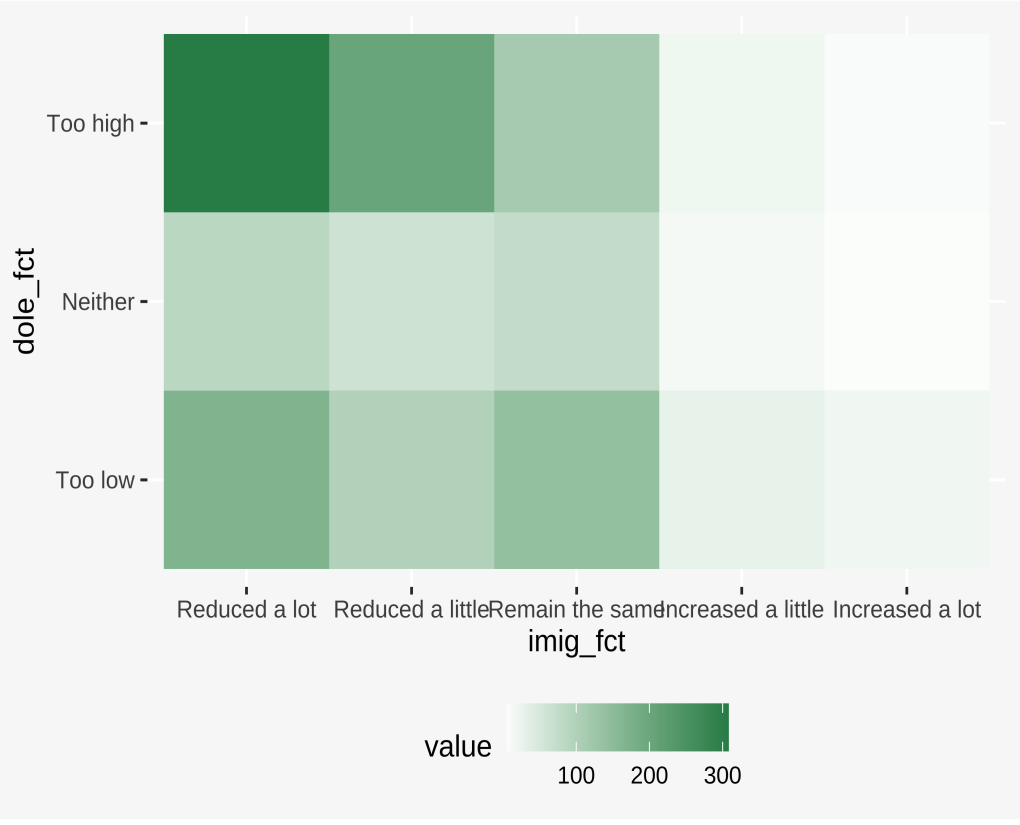
Heatmap




```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>% # Get crosstab
  as_tibble() %>% # Convert to tibble
  pivot_longer(-1, names_to = "dole_fct") %>% # Create 'long'
data
  mutate( # re-factorise dole_fct
    dole_fct = factor(dole_fct, levels = c("Too low", "Neithe
r", "Too high"))
  ) %>%
  ggplot() + # plot data
  geom_tile(aes(x = imig_fct, y = dole_fct, fill = value)) +
# create heatmap
  theme(legend.position = "bottom") + # move legend to bottom
  coord_fixed(1) # force squares
```

We can use the `coord_fixed(1)` function to force a 1 to 1 relationship between the X and Y axis, this will make the tiles display as perfect squares (which looks quite nice!)

Heatmap




```
ssa %>%
  filter(!is.na(imig_fct) & !is.na(dole_fct)) %>% # Remove mis
sing
  janitor::tabyl(imig_fct, dole_fct) %>% # Get crosstab
  as_tibble() %>% # Convert to tibble
  pivot_longer(-1, names_to = "dole_fct") %>% # Create 'long'
data
  mutate( # re-factorise dole_fct
    dole_fct = factor(dole_fct, levels = c("Too low", "Neithe
r", "Too high"))
  ) %>%
  ggplot() + # plot data
  geom_tile(aes(x = imig_fct, y = dole_fct, fill = value)) +
# create heatmap
  theme(legend.position = "bottom") + # move legend to bottom
  coord_fixed(1) + # force squares
  scale_fill_gradient2(low = "white", high = "seagreen") # cha
nge colour scheme
```

Finally, we can use `scale_fill_gradient2()` to set any arbitrary fill colour scale between two colours we choose (here, a white and a green). **R** has a list of pre-named colours (you can see all of these by running `colors()` in your console), or you can use any hex code colour (e.g. #B00100)

How do we effectively summarise and communicate this relationship in a standardised way?

Descriptive Statistics for Describing Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	**Contingency Table + Cramer's V**		
Ordinal	**Contingency Table + Cramer's V**	Spearman's Rho/ Contingency Table	
Continuous	Mean/ Median Difference	Mean/ Median Difference	Pearson's R or Spearman's Rho

Contingency tables ("Crosstabs")

Contingency tables show the **frequency of responses that fall within all combinations of categories within two variables** with mutually exclusive categories.

```
library(janitor)

ssa %>%
  tabyl(imig_fct, dole_fct)
```

```
##            imig_fct Too low Neither Too high NA_
##       Reduced a lot     168      86      308  49
##    Reduced a little      99      62      203  36
##     Remain the same     141      76      115  29
##  Increased a little      30      12       19   5
##     Increased a lot      18       5        7   2
##                <NA>       9      11        9   2
```

Contingency tables ("Crosstabs")

Contingency tables show the **frequency of responses that fall within all combinations of categories within two variables** with mutually exclusive categories.

Most visualisations rely on contingency but assign aesthetic elements (e.g. height of a bar or fill of a square) to illustrate variation.

```
library(janitor)

ssa %>%
  tabyl(imig_fct, dole_fct)
```

```
##            imig_fct Too low Neither Too high NA_
##       Reduced a lot     168      86       308  49
##    Reduced a little      99      62       203  36
##     Remain the same     141      76       115  29
##  Increased a little      30      12        19   5
##     Increased a lot      18       5         7   2
##                <NA>       9      11         9   2
```

Contingency tables ("Crosstabs")

Contingency tables show the **frequency of responses that fall within all combinations of categories within two variables** with mutually exclusive categories.

Most visualisations rely on contingency but assign aesthetic elements (e.g. height of a bar or fill of a square) to illustrate variation.

- Interpreting and communicating dependence between two categorical or ordinal variables using a contingency table **often requires the use of percentages**.
    - These can be calculated based on the rows

```
library(janitor)

ssa %>%
  tabyl(imig_fct, dole_fct) %>%
  adorn_percentages(denominator = "row") %>%
  adorn_totals(where = c("col")) %>%
  adorn_pct_formatting()
```

```
##            imig_fct Too low Neither Too high  NA_  Total
##       Reduced a lot   27.5%   14.1%    50.4% 8.0% 100.0%
##    Reduced a little   24.8%   15.5%    50.7% 9.0% 100.0%
##     Remain the same   39.1%   21.1%    31.9% 8.0% 100.0%
##  Increased a little   45.5%   18.2%    28.8% 7.6% 100.0%
##     Increased a lot   56.2%   15.6%    21.9% 6.2% 100.0%
##                <NA>   29.0%   35.5%    29.0% 6.5% 100.0%
```

Contingency tables ("Crosstabs")

Contingency tables show the **frequency of responses that fall within all combinations of categories within two variables** with mutually exclusive categories.

Most visualisations rely on contingency but assign aesthetic elements (e.g. height of a bar or fill of a square) to illustrate variation.

- Interpreting and communicating dependence between two categorical or ordinal variables using a contingency table **often requires the use of percentages**.
    - These can be calculated based on the rows
    - Or based on columns

```
library(janitor)

ssa %>%
  tabyl(imig_fct, dole_fct) %>%
  adorn_percentages(denominator = "col") %>%
  adorn_totals(where = c("row")) %>%
  adorn_pct_formatting()
```

```
##            imig_fct Too low Neither Too high    NA_
##       Reduced a lot   36.1%   34.1%    46.6%  39.8%
##    Reduced a little   21.3%   24.6%    30.7%  29.3%
##     Remain the same   30.3%   30.2%    17.4%  23.6%
##  Increased a little    6.5%    4.8%     2.9%   4.1%
##     Increased a lot    3.9%    2.0%     1.1%   1.6%
##                <NA>    1.9%    4.4%     1.4%   1.6%
##               Total  100.0%  100.0%   100.0% 100.0%
```

Contingency tables ("Crosstabs")

Contingency tables show the **frequency of responses that fall within all combinations of categories within two variables** with mutually exclusive categories.

Most visualisations rely on contingency but assign aesthetic elements (e.g. height of a bar or fill of a square) to illustrate variation.

- Interpreting and communicating dependence between two categorical or ordinal variables using a contingency table **often requires the use of percentages**.
    - These can be calculated based on the rows
    - Or based on columns
- However, this should **always be clearly labelled** and actual frequency counts should **always be available** to be transparent about small numbers of respondents.

```
library(janitor)

ssa %>%
  tabyl(imig_fct, dole_fct) %>%
  adorn_percentages(denominator = "col") %>%
  adorn_totals(where = c("row")) %>%
  adorn_pct_formatting()
```

```
##            imig_fct Too low Neither Too high    NA_
##       Reduced a lot   36.1%   34.1%    46.6%  39.8%
##    Reduced a little   21.3%   24.6%    30.7%  29.3%
##     Remain the same   30.3%   30.2%    17.4%  23.6%
##  Increased a little    6.5%    4.8%     2.9%   4.1%
##     Increased a lot    3.9%    2.0%     1.1%   1.6%
##                <NA>    1.9%    4.4%     1.4%   1.6%
##               Total  100.0%  100.0%   100.0% 100.0%
```

```
##            imig_fct Too low Neither Too high NA_
##       Reduced a lot     168      86      308  49
##    Reduced a little      99      62      203  36
##     Remain the same     141      76      115  29
##  Increased a little      30      12       19   5
##     Increased a lot      18       5        7   2
##                <NA>       9      11        9   2
```

Cramer's V

Measure of asymmetry/dependence.

- 0 = Total independence
- 1 = Total dependence

Does knowing the value of one of the variables help you know the value of the other variable (dependence), or does it make no difference (independence)?

```
library(rcompanion)

ssa %>%
  tabyl(imig_fct, dole_fct)
```

```
##            imig_fct Too low Neither Too high NA_
##       Reduced a lot     168      86      308  49
##    Reduced a little      99      62      203  36
##     Remain the same     141      76      115  29
##  Increased a little      30      12       19   5
##     Increased a lot      18       5        7   2
##                <NA>       9      11        9   2
```

```
cramerV(x = ssa$dole_fct, y = ssa$imig_fct)
```

```
## Cramer V
##   0.1169
```

Cramer's V

Measure of asymmetry/dependence.

- 0 = Total independence
- 1 = Total dependence

Does knowing the value of one of the variables help you know the value of the other variable (dependence), or does it make no difference (independence)?

```
zero_dep <- matrix(c(50, 50, 50, 50), nrow = 2, ncol = 2)

zero_dep
```

```
##      [,1] [,2]
## [1,]   50   50
## [2,]   50   50
```

```
cramerV(zero_dep)
```

```
## Cramer V
##        0
```

Cramer's V

Measure of asymmetry/dependence.

- 0 = Total independence
- 1 = Total dependence

Does knowing the value of one of the variables help you know the value of the other variable (dependence), or does it make no difference (independence)?

```
total_dep <- matrix(c(0, 100, 100, 0), nrow = 2, ncol = 2)

total_dep
```

```
##      [,1] [,2]
## [1,]    0  100
## [2,]  100    0
```

```
cramerV(total_dep)
```

```
## Cramer V
##        1
```

Cramer's V

Measure of asymmetry/dependence.

- 0 = Total independence
- 1 = Total dependence

Does knowing the value of one of the variables help you know the value of the other variable (dependence), or does it make no difference (independence)?

```
partial_dep <- matrix(c(25, 75, 75, 25), nrow = 2, ncol = 2)

partial_dep
```

```
##      [,1] [,2]
## [1,]   25   75
## [2,]   75   25
```

```
cramerV(partial_dep)
```

```
## Cramer V
##      0.5
```

Cramer's V

df*	Negligible	Weak	Moderate	Strong
1	0 < .10	.10 < .30	.30 < .50	.50 or more
2	0 < .07	.07 < .21	.21 < .35	.35 or more
3	0 < .06	.06 < .17	.17 < .29	.29 or more
4	0 < .05	.05 < .15	.15 < .25	.25 or more
5	0 < .05	.05 < .13	.13 < .22	.22 or more

The degrees of freedom (df) are calculated by subtracting 1 from the smallest number of rows or columns in our crosstab.

For example here we have 6 rows and 4 columns; 4 is smaller than 6; so our "df" is equal to 4 minus 1, which equals 3.

For 3 degrees of freedom a result of 0.1169 could be interpreted as a **weak** association between attitudes towards immigration and benefits policies.

```
ssa %>%
  tabyl(imig_fct, dole_fct)
```

```
##            imig_fct Too low Neither Too high NA_
##       Reduced a lot     168      86       308  49
##    Reduced a little      99      62       203  36
##     Remain the same     141      76       115  29
##  Increased a little      30      12        19   5
##     Increased a lot      18       5         7   2
##                <NA>       9      11         9   2
```

```
cramerV(x = ssa$dole_fct, y = ssa$imig_fct)
```

```
## Cramer V
##   0.1169
```

Descriptive Statistics for Describing Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Contingency Table + Cramer's V		
Ordinal	Contingency Table + Cramer's V	**Spearman's Rho/ Contingency Table**	
Continuous	Mean/ Median Difference	Mean/ Median Difference	Pearson's R or Spearman's Rho

Spearman's rho (ρ): Rank order correlation

```
# Change factors to be ordered
ssa <- ssa %>%
  mutate(
    imig_fct = factor(imig_fct, ordered = TRUE),
    dole_fct = factor(dole_fct, ordered = TRUE)
  )

# Immigration should be...
# 5 Levels: Reduced a lot < Reduced a little < ... < Increased
a lot

# Benefits payments are...
# Levels: Too low < Neither < Too high

cor(x = as.numeric(ssa$imig_fct), # Must be numeric
    y = as.numeric(ssa$dole_fct), # Must be numeric
    use = "complete.obs", # Remove missing values
    method = "spearman" # Use spearman's rho
    )
```

```
## [1] -0.1664359
```

How closely do the orders of the two variables match up when ranked?

- 1 = Ranks increase at exactly the same rate
- 0 = Ranks neither increase or decrease consistently between the two variables
- -1 = As the rank of one variable increases, the other decreases.

Spearman's rho (ρ): Rank order correlation

```
# Change factors to be ordered
ssa <- ssa %>%
  mutate(
    imig_fct = factor(imig_fct, ordered = TRUE),
    dole_fct = factor(dole_fct, ordered = TRUE)
  )

# Immigration should be...
# 5 Levels: Reduced a lot < Reduced a little < ... < Increased
a lot

# Benefits payments are...
# Levels: Too low < Neither < Too high

cor(x = as.numeric(ssa$imig_fct), # Must be numeric
    y = as.numeric(ssa$dole_fct), # Must be numeric
    use = "complete.obs", # Remove missing values
    method = "spearman" # Use spearman's rho
    )
```

```
## [1] -0.1664359
```

How closely do the orders of the two variables match up when ranked?

- 1 = Ranks increase at exactly the same rate
- 0 = Ranks neither increase or decrease consistently between the two variables
- **-1 = As the rank of one variable increases, the other decreases.**

As the rank of immigration attitudes increases (people feel more that it should immigration should be increased and less that it should be reduced), **the rank of attitudes towards benefits tends to decrease** (people feel more that benefits are too low rather than too high).

People who are pro-immigration are also more likely to support greater welfare generosity, though this is only a weak association.

Spearman's rho (ρ): Rank order correlation

```
# Change factors to be ordered
ssa <- ssa %>%
  mutate(
    imig_fct = factor(imig_fct, ordered = TRUE),
    dole_fct = factor(dole_fct, ordered = TRUE)
  )

# Immigration should be...
# 5 Levels: Reduced a lot < Reduced a little < ... < Increased
a lot

# Benefits payments are...
# Levels: Too low < Neither < Too high

cor(x = as.numeric(ssa$imig_fct), # Must be numeric
    y = as.numeric(ssa$dole_fct), # Must be numeric
    use = "complete.obs", # Remove missing values
    method = "spearman" # Use spearman's rho
    )
```

```
## [1] -0.1664359
```

Spearman's Rank Order Correlation is **directional**, Cramer's V is **not**.

Cohen's (1988) commonly used effect sizes for rho:

- 0 to 0.1 or 0 to -0.1: Negligible
- 0.1 to 0.3 or -0.1 to -0.3: Weak
- 0.3 to 0.5 or -0.3 to -0.5: Moderate
- 0.5 to 1 or -0.5 to -1: Strong

However, **context matters most**: if you are looking for things that might affect an outcome **at scale**, or **when assessing a potentially cheap intervention** even a relatively weak effect size is important.

- If the evidence suggested giving students a free notepad and pen at the start of a semester had a weak effect on their grades, would we do it?
- If the evidence suggested giving students £1million at the start of a semester had a strong effect on their grades, would we do it?

Are attitudes towards immigration associated with age?

Visualisations for Exploring Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Heatmap/ Bivariate Bar Chart		
Ordinal	Heatmap/ Bivariate Bar Chart	Heatmap/ Bivariate Bar Chart	
Continuous	**Boxplot/** **Ridgeplot**	**Boxplot/** **Ridgeplot**	Scatterplot/ Hex Bin Plot

Data preparation: Recoding

It's sometimes necessary to "recode" variables. Some reasons why you may want to do this are:

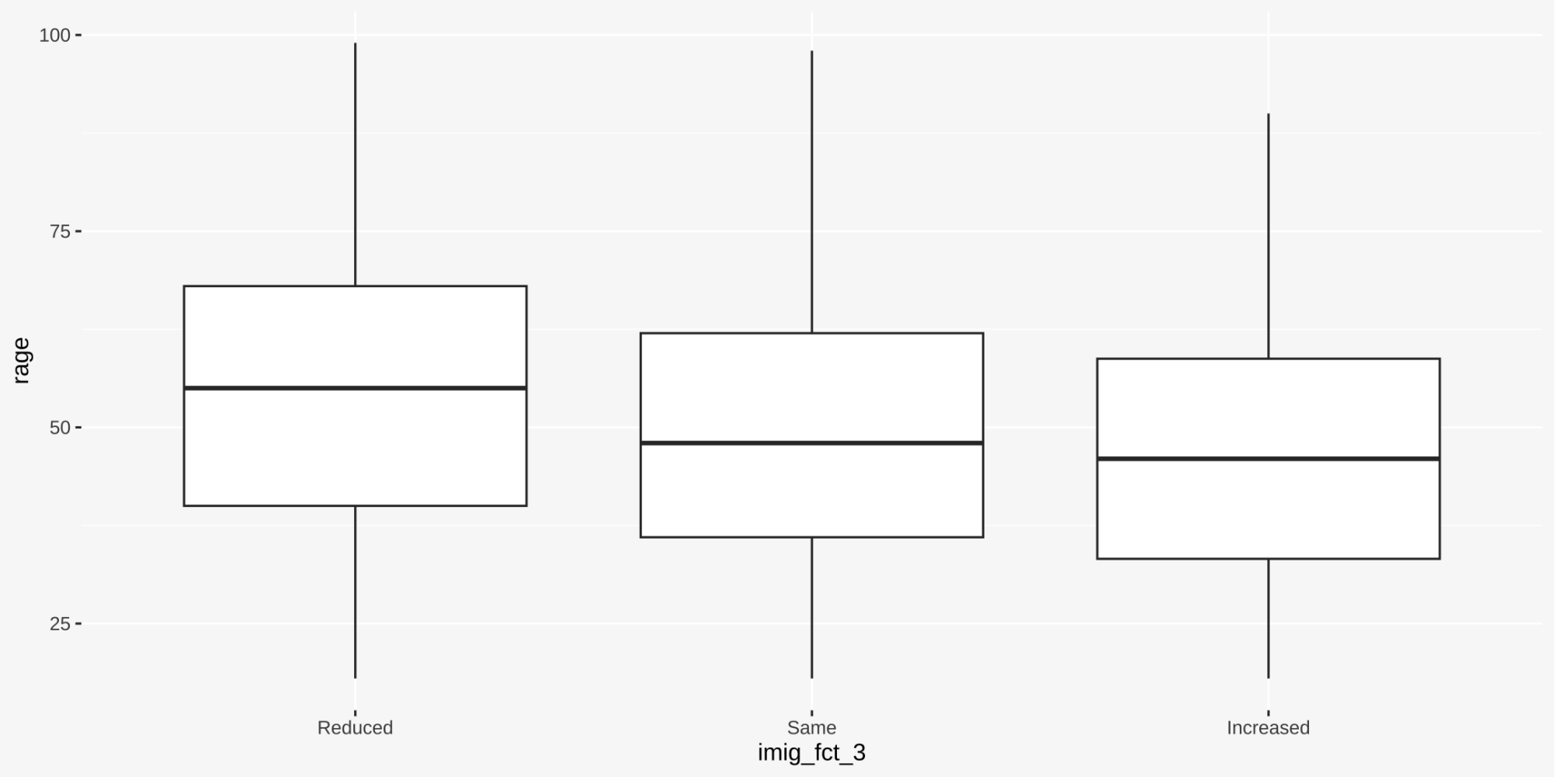
- To combine some related groups within each there are **too small a number of cases to be statistically informative** (generally, fewer than 30 is not very informative for frequentist hypothesis testing)
- To compress certain groups to make a **data visualisation more clear or appealing**, or to use a different type of visualisation.
- **To remove missing data categories** that were placed in the data to be informative but are not needed for your analysis
- Example of ways you might recode a variable:
    - Recoding a 7-point Likert scale to a 5-point scale;
    - Turning a continuous variable like age into an ordinal variable like 'age group';
    - Turning an ordinal variable into a binary variable, e.g. simplifying a five point scale from Strongly Agree -- Strongly Disagree into a simple "Agree / Does not agree" binary;
    - Changing the categorical responses "Refused to answer", "Not applicable", and "Not sure" into `NA`.

Data preparation: Recoding

```
ssa <- ssa %>%
  mutate(
    imig_fct_3 = case_when(imig_fct == "Reduced a lot" ~ "Reduced",
                           imig_fct == "Reduced a little" ~ "Reduced",
                           imig_fct == "Remain the same" ~ "Same",
                           imig_fct == "Increased a little" ~ "Increased",
                           imig_fct == "Increased a lot" ~ "Increased",
                           TRUE ~ NA_character_)
  ) %>%
  mutate(
    imig_fct_3 = factor(imig_fct_3,
                        levels = c("Reduced", "Same", "Increased"),
                        ordered = TRUE)
  )

ssa %>%
  tabyl(imig_fct, imig_fct_3)
```

```
##            imig_fct Reduced Same Increased NA_
##       Reduced a lot     611    0         0   0
##    Reduced a little     400    0         0   0
##     Remain the same       0  361         0   0
##  Increased a little       0    0        66   0
##     Increased a lot       0    0        32   0
##                <NA>       0    0         0  31
```

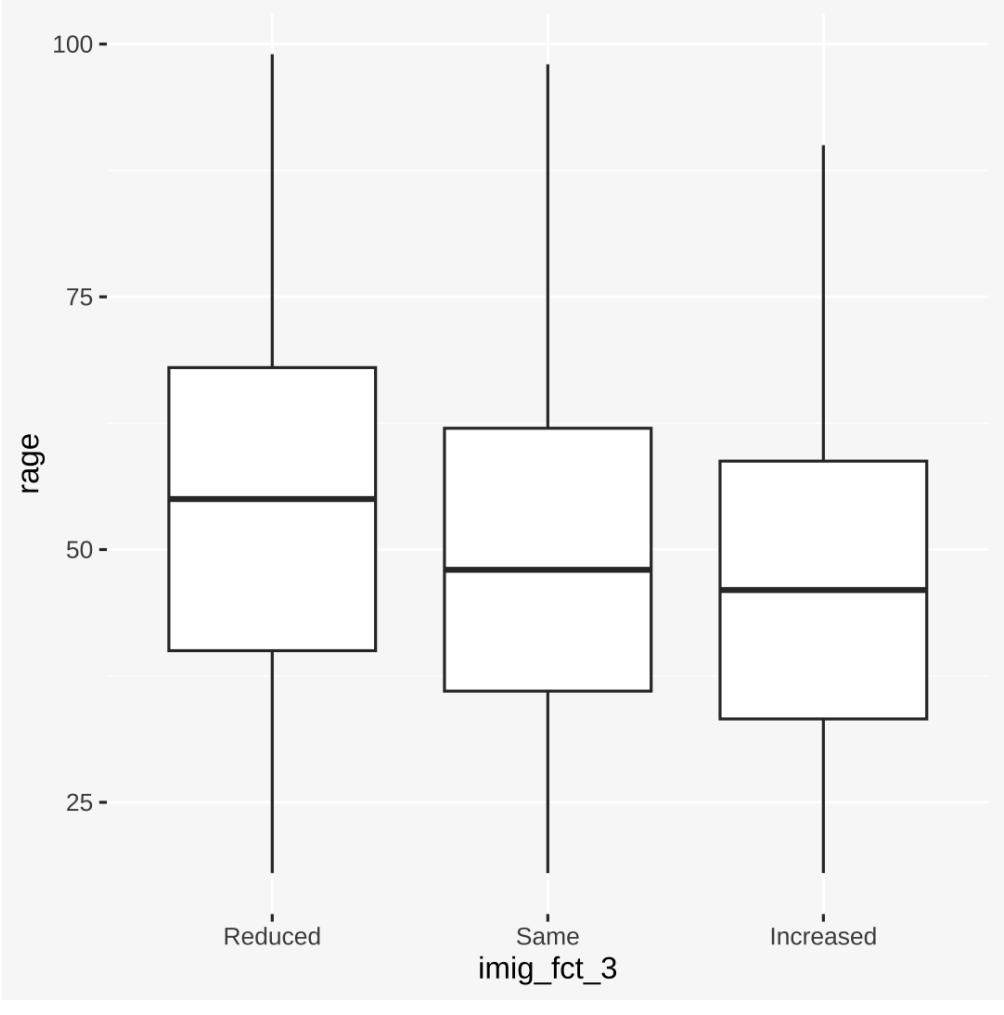
Boxplot



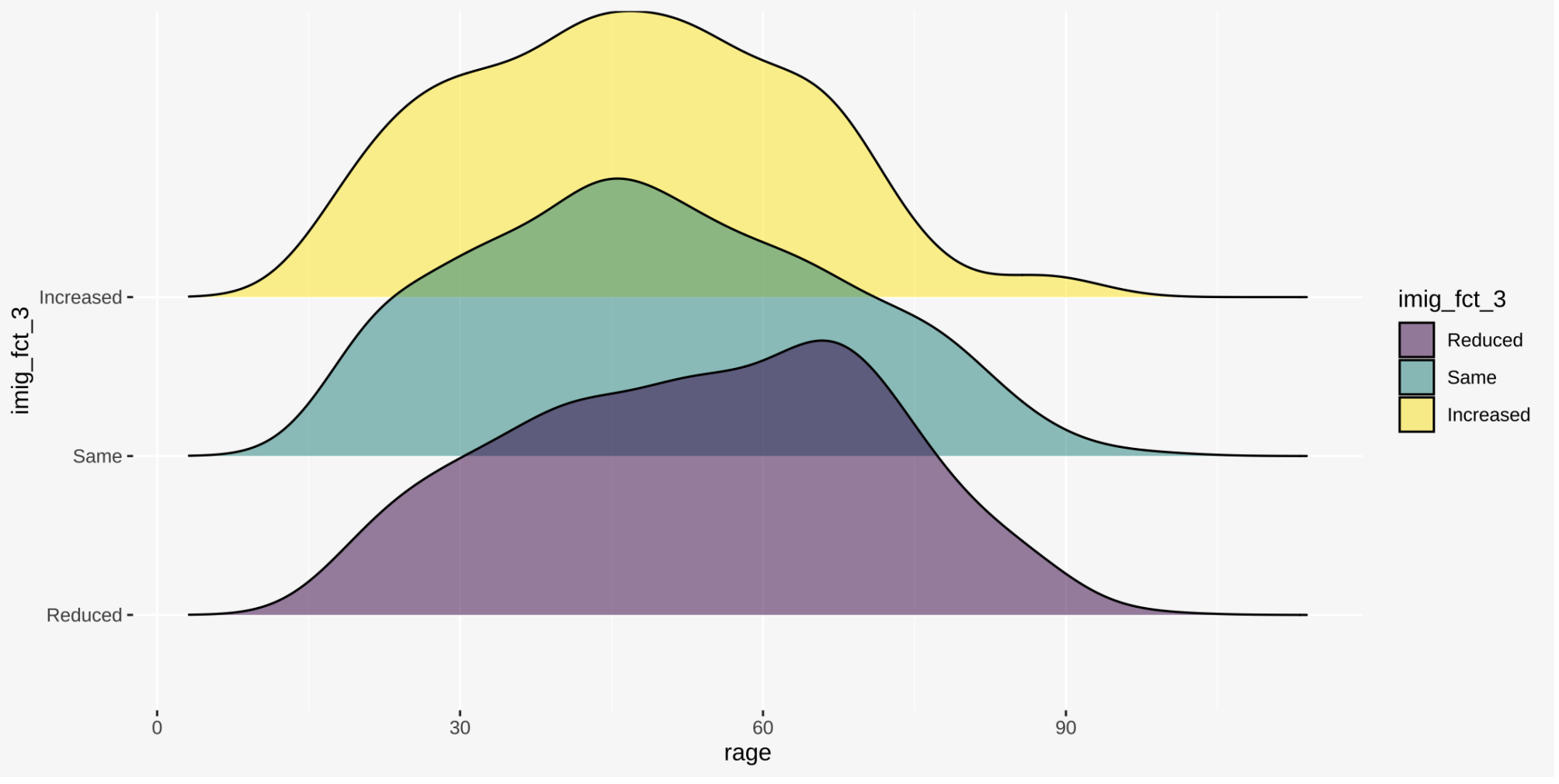
Boxplot

```
ssa %>%
  # Remove missing
  filter(!is.na(rage) & !is.na(imig_fct_3)) %>%
  # Start plotting
  ggplot() +
  geom_boxplot(aes(x = imig_fct_3, y = rage))

# rage = Respondent's Age
```

- Whiskers = Minimum and Maximum
- Box = 25th to 75th percentile (50% of observations)
- Medial = Median



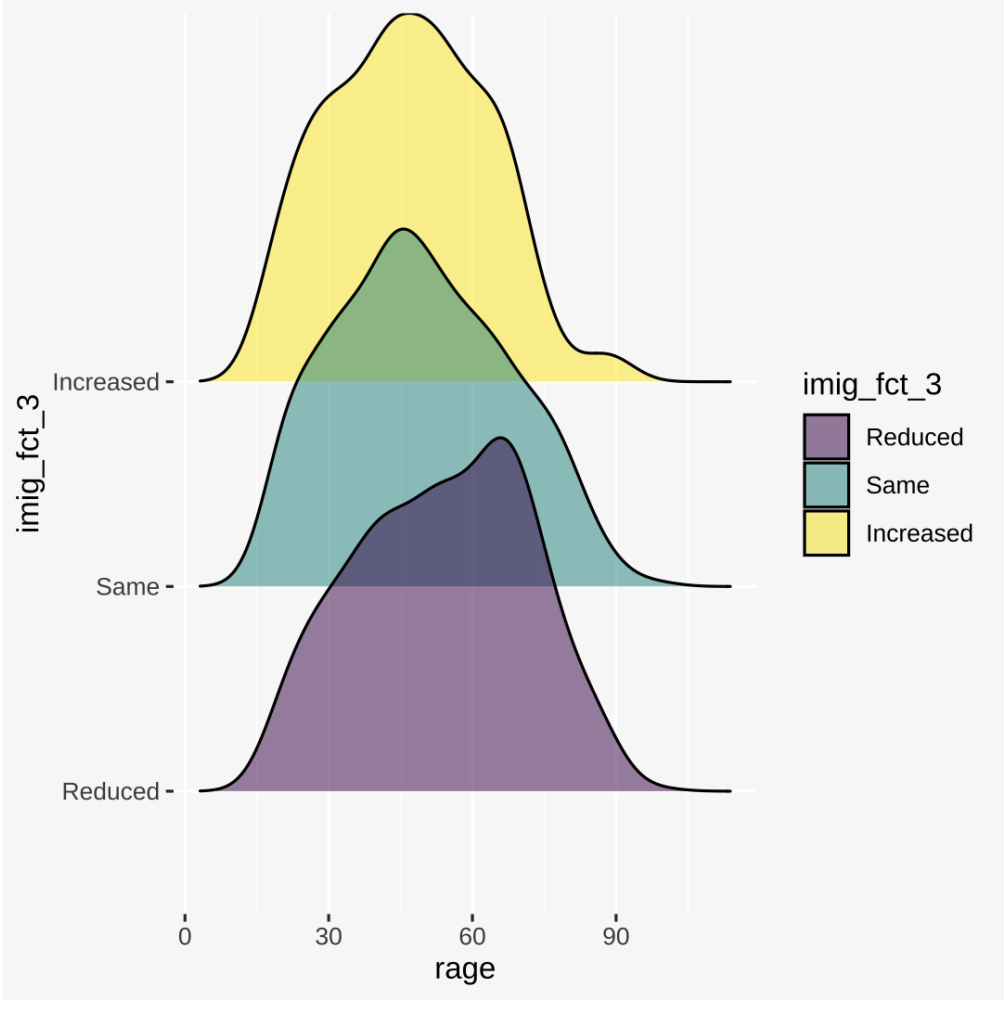
Ridgeplot


imig_fct_3
Increased
Same
Reduced
0
30
60
90
rage
imig_fct_3
Reduced
Same
Increased

Ridgeplot

```
library(ggridges)

ssa %>%
  filter(!is.na(rage) & !is.na(imig_fct_3)) %>%
  ggplot() +
  geom_density_ridges(
    aes(x = rage, y = imig_fct_3, fill = imig_fct_3),
    alpha = 0.5
    )

# rage = Respondent's Age
```

- Shows distribution across the range - makes it easy to assess the distribution across groups (e.g. whether the variable is normally distributed across all groups).



Descriptive Statistics for Describing Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Contingency Table + Cramer's V		
Ordinal	Contingency Table + Cramer's V	Spearman's Rho/ Contingency Table	
Continuous	**Mean/ Median Difference**	**Mean/ Median Difference**	Pearson's R or Spearman's Rho

Mean/Median Differences (and Ranges)

```
ssa %>%
  group_by(imig_fct_3) %>%
  summarise(
    mean_age      = mean(rage, na.rm = TRUE),
    median_age    = median(rage, na.rm = TRUE),
    min_age       = min(rage, na.rm = TRUE),
    percentile_25 = quantile(rage, probs = 0.25),
    percentile_75 = quantile(rage, probs = 0.75),
    max_age       = max(rage, na.rm = TRUE)
  )
```

Age summary statistics by attitudes towards future immigration policy

```
## # A tibble: 4 × 7
##   imig_fct_3 mean_age median_age min_age percentile_25 percentile_75 max_age
##   <ord>         <dbl>      <dbl>   <dbl>         <dbl>         <dbl>   <dbl>
## 1 Reduced        53.9         55      18          40            68        99
## 2 Same           49.4         48      18          36            62        98
## 3 Increased      46.7         46      18          33.2          58.8      90
## 4 <NA>           50.6         52      24          35.5          64        92
```

"The median age of someone who answered immigration should be reduced was 54 years, while the median age if participants who responded that immigration should be increased was 46 years. The interquartile range between those who believed immigration should be Reduced and those who believed it should be increased had some degree of overlap, but those who felt immigration should be reduced had a considerably younger range. The inter-quartile range was between age 39 and age 67 for those who felt immigration should be reduced and from 33 to 58 for those who felt it should be increased."

Mean/Median Differences (and Ranges)

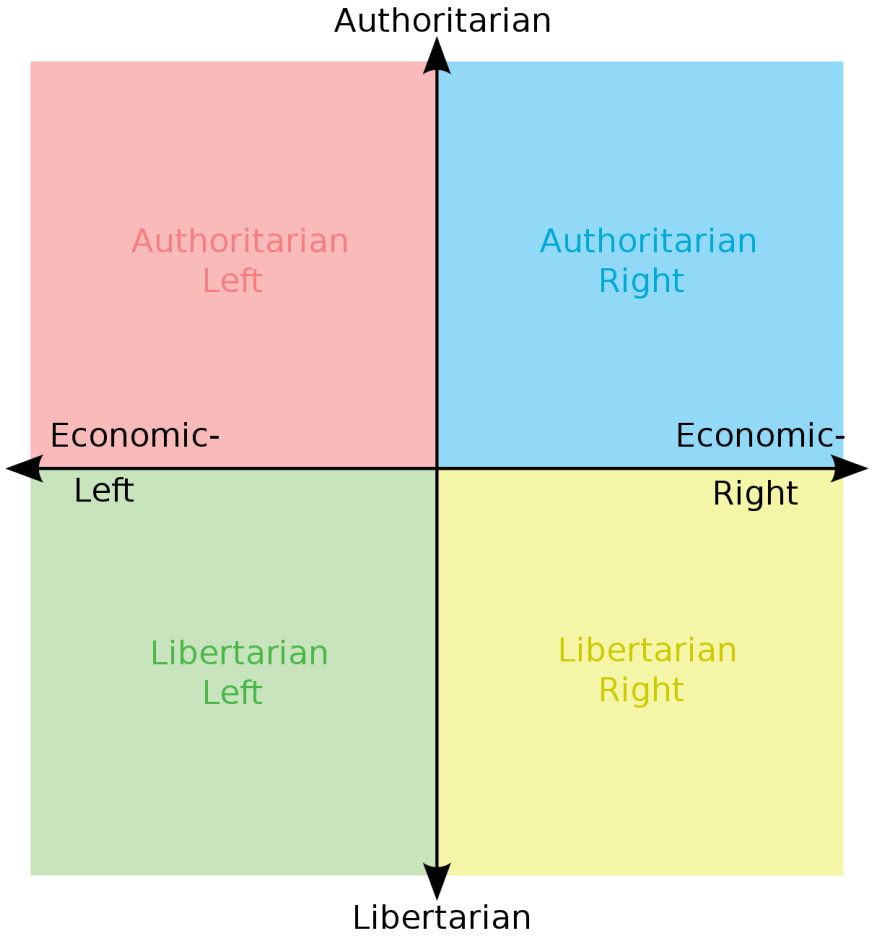
```
ssa %>%
  group_by(dole_fct) %>%
  summarise(
    mean_age      = mean(rage, na.rm = TRUE),
    median_age    = median(rage, na.rm = TRUE),
    min_age       = min(rage, na.rm = TRUE),
    percentile_25 = quantile(rage, probs = 0.25),
    percentile_75 = quantile(rage, probs = 0.75),
    max_age       = max(rage, na.rm = TRUE)
  )
```

Age summary statistics by attitudes towards welfare generousity

```
## # A tibble: 4 × 7
##   dole_fct mean_age median_age min_age percentile_25 percentile_75 max_age
##   <ord>       <dbl>      <dbl>   <dbl>         <dbl>         <dbl>   <dbl>
## 1 Too low      52.4         53      18          41            65        90
## 2 Neither      53.2         53      18          40            66        99
## 3 Too high     51.8         52      18          37            67        98
## 4 <NA>         52.2         51      19          37.5          67.5      90
```

How would you summarise this table in text form?

Is a person's age associated with their position on an Authoritarian-Libertarian scale?


Authoritarian
Authoritarian
Left
Authoritarian
Right
Economic-
Left
Economic-
Right
Libertarian
Left
Libertarian
Right
Libertarian

'Young people today don't have enough respect for traditional British values'

1. Agree strongly
2. Agree
3. Neither agree nor disagree
4. Disagree
5. Disagree strongly
6. (Don't know)
7. (Refusal)

'People who break the law should be given stiffer sentences'

'For some crimes, the death penalty is the most appropriate sentence'

'Schools should teach children to obey authority'

'The law should always be obeyed, even if a particular law is wrong'

'Censorship of films and magazines is necessary to uphold moral standards'

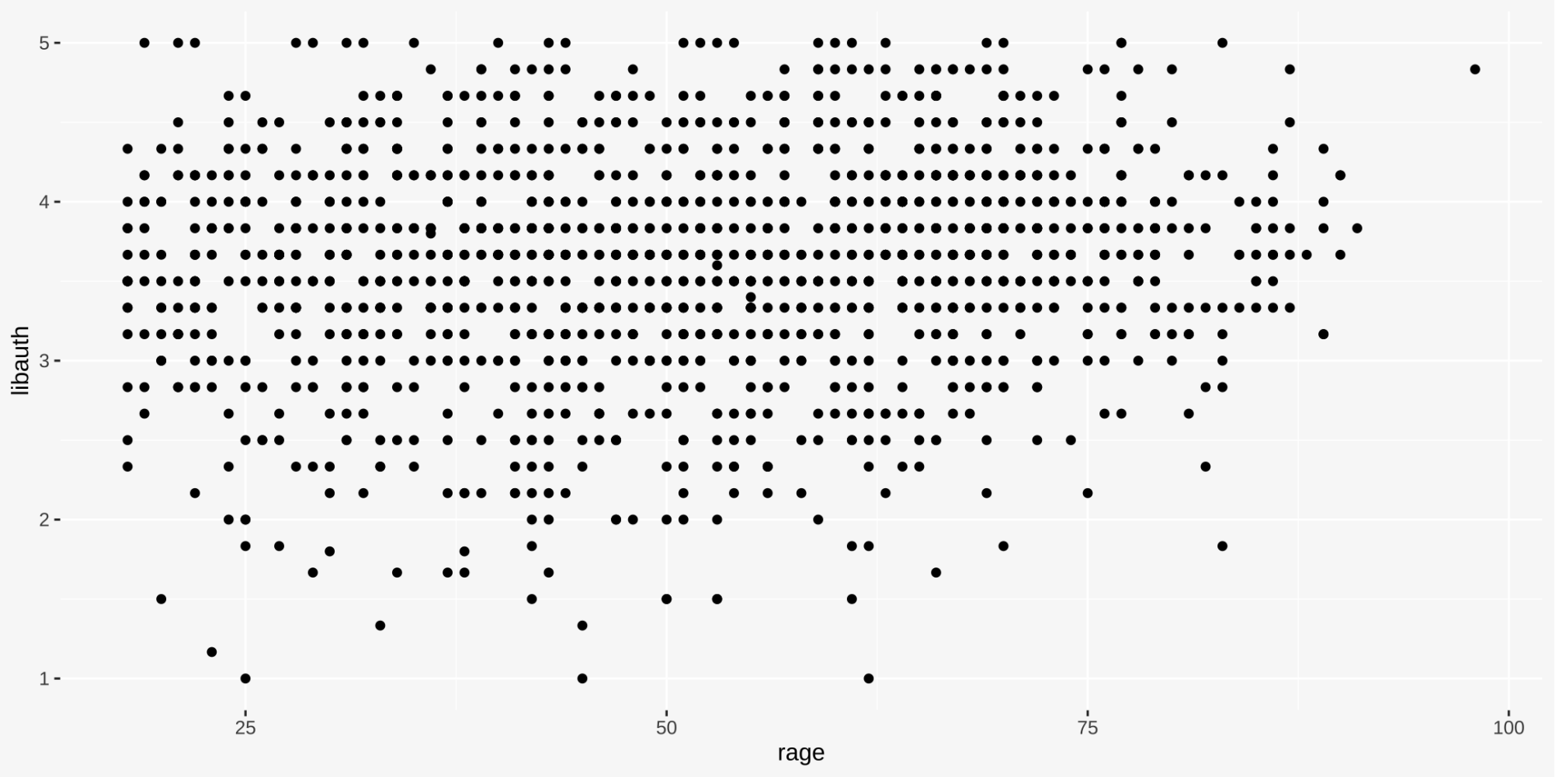
'Gay or lesbian couples should have the right to marry one another if they want to.'

Variable `libauth` is a scale based on responses where the value 1 represents the most libertarian position and 5 the most authoritarian.

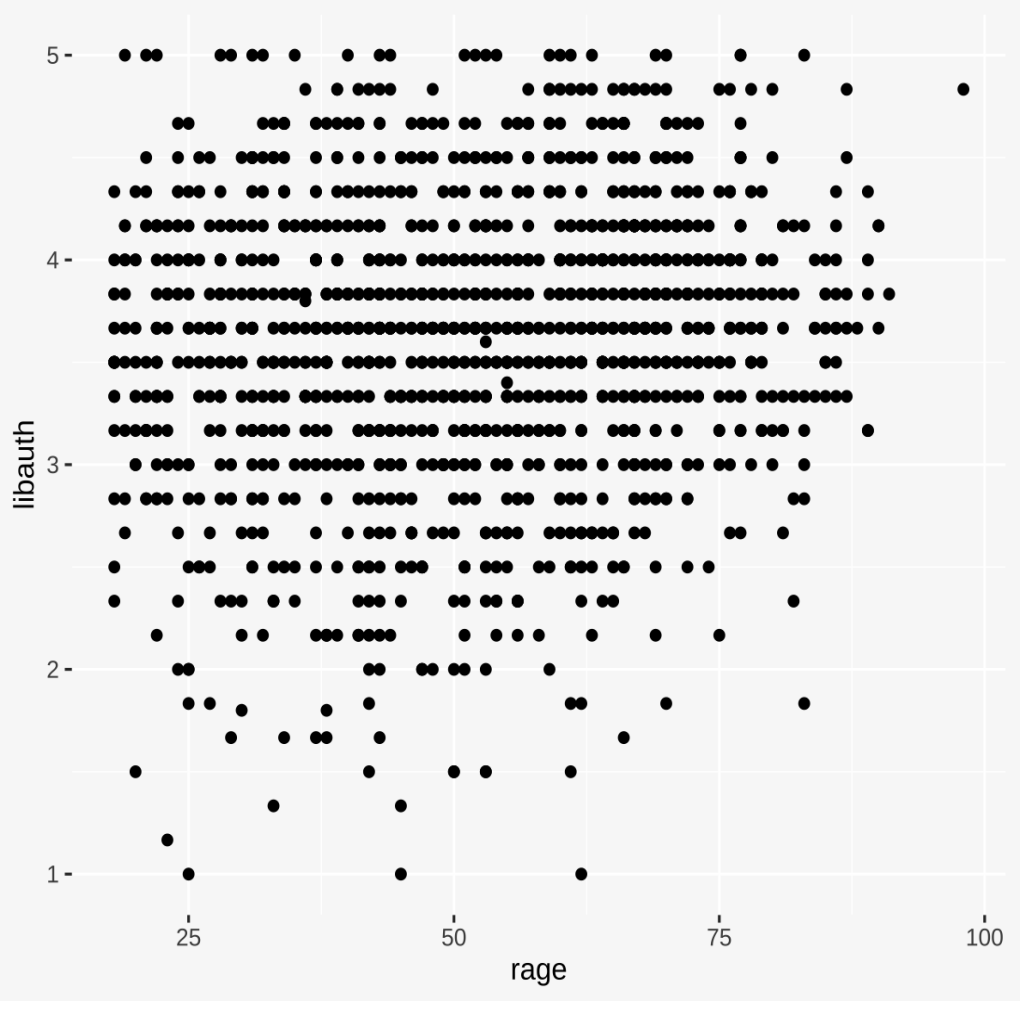
Visualisations for Exploring Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Heatmap/ Bivariate Bar Chart		
Ordinal	Heatmap/ Bivariate Bar Chart	Heatmap/ Bivariate Bar Chart	
Continuous	Boxplot/ Ridgeplot	Boxplot/ Ridgeplot	**Scatterplot/** **Hex Bin Plot**

Scatterplot

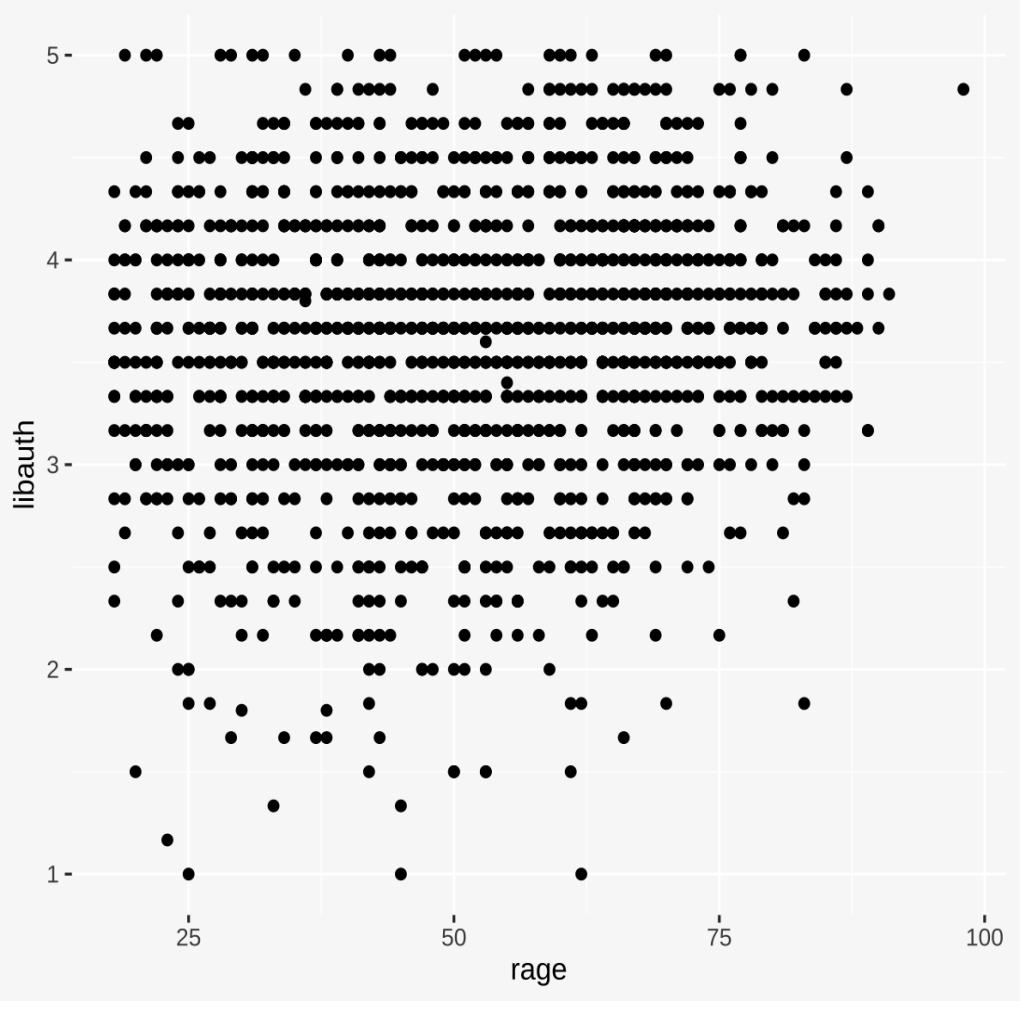


Scatterplot




```
ssa %>%
  ggplot() +
  geom_point(aes(x = rage, y = libauth))
```

Scatterplot




```
ssa %>%
  ggplot() +
  geom_point(aes(x = rage, y = libauth))
```

- Can easily end up with overplotting - especially with discrete variables.

Possible solutions:

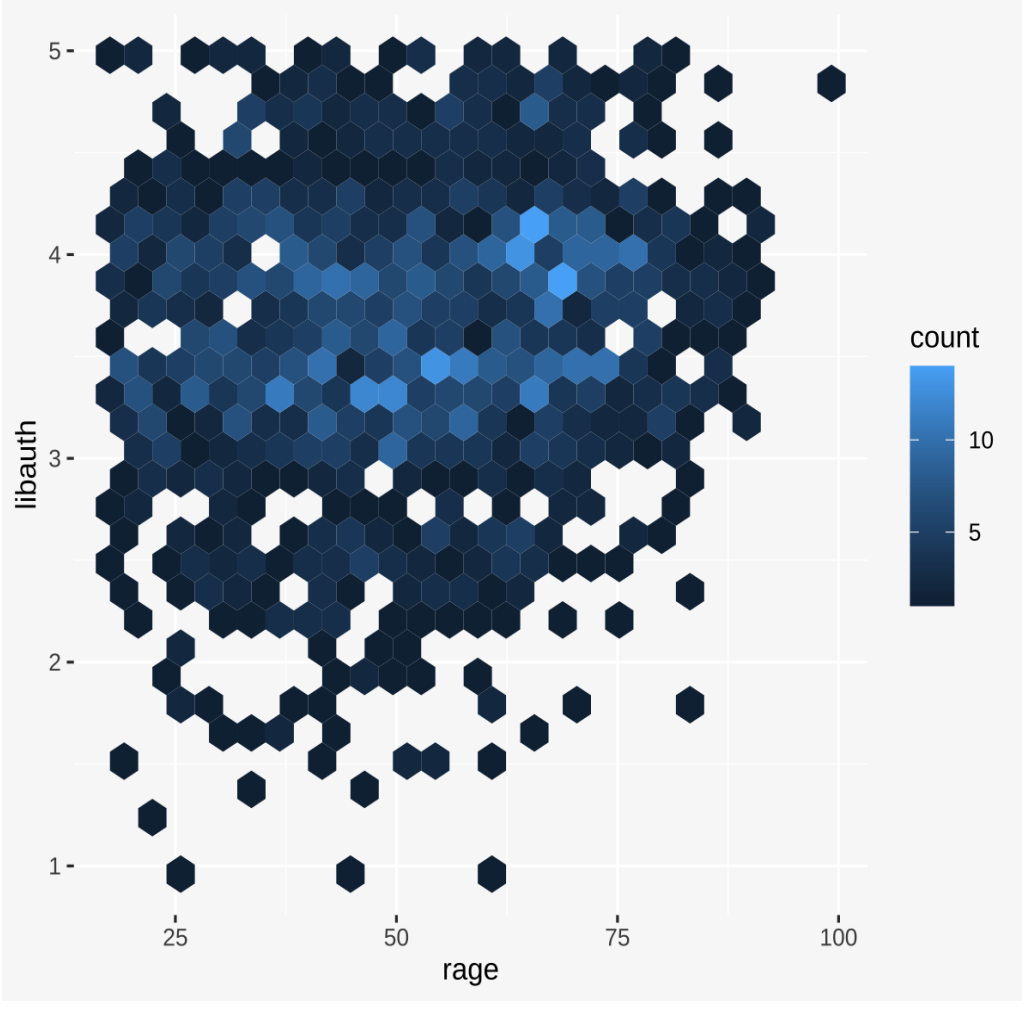
- Add jitter to points (artificially spread them out from their true value) - not great as it manipulates data.
- Make points smaller - useful for overplotting with continuous variables but less useful for discrete
- Add transparency - same as above

Or...

- Use a hexbin plot.

Hexbin plot

```
ssa %>%
  ggplot() +
  geom_hex(aes(x = rage, y = libauth), bins = 25)
```



Descriptive Statistics for Describing Relationships (Dependence) Between Variables

Variable Type	Nominal	Ordinal	Continuous
Nominal	Contingency Table + Cramer's V		
Ordinal	Contingency Table + Cramer's V	Spearman's Rho/ Contingency Table	
Continuous	Mean/ Median Difference	Mean/ Median Difference	**Pearson's R or Spearman's Rho**

Correlation: Pearson's R

```
cor(x = ssa$rage, y = ssa$libauth,
    use = "complete.obs",
    method = "pearson")
```

```
## [1] 0.1236948
```

```
cor(x = ssa$rage, y = ssa$libauth,
    use = "complete.obs",
    method = "spearman")
```

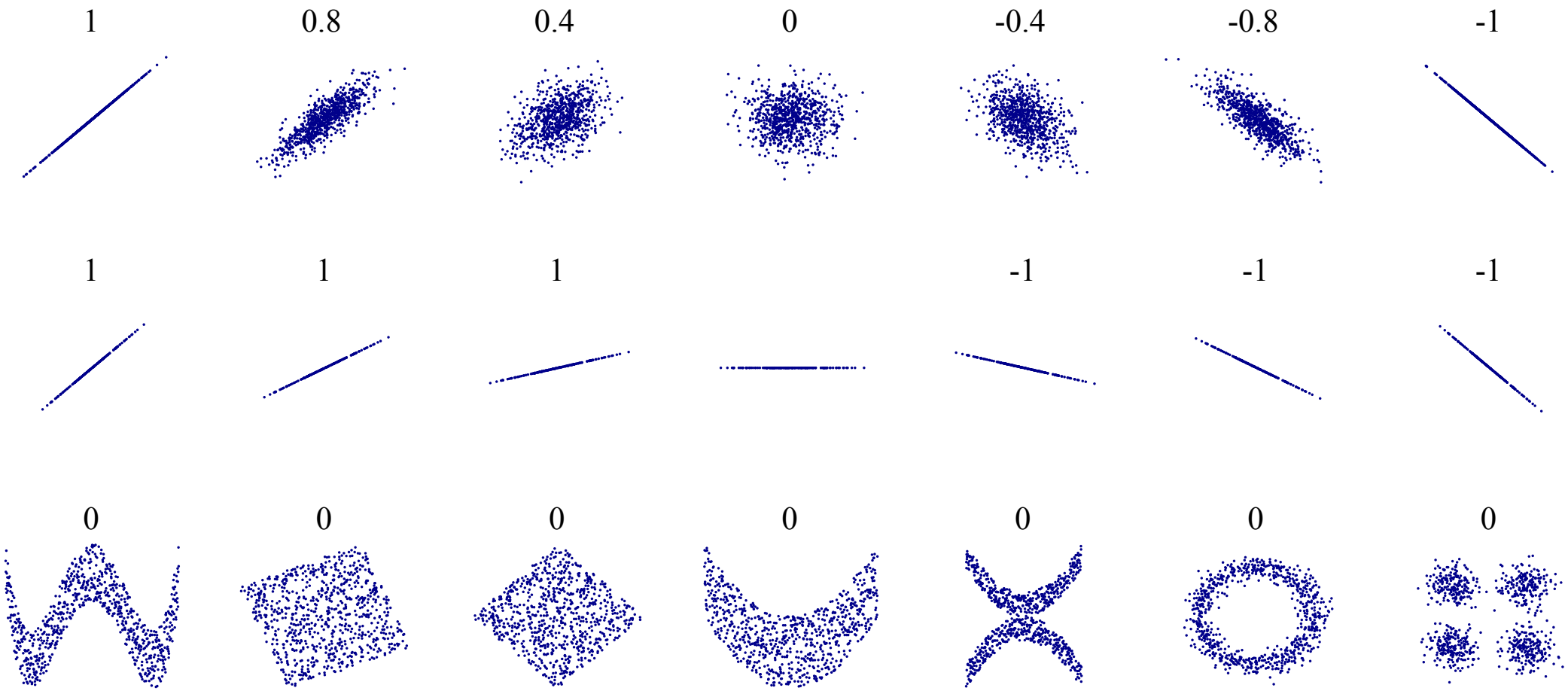
```
## [1] 0.1194041
```

Pearson's correlation coefficient R, is a measure of **how closely associated the variance in one variable is with the variance in another variable**, relative to the maximum possible covariance they could share.

- -1 = Perfect negative correlation - as the value of one variable increases the other decreases.
- 0 = No correlation - changes in the value of one variable are not associated with changes in the other.
- 1 = Perfect positive correlation - as the value of one variable increases the other also increases.

Rules of thumb:

- ±0.01 - 0.19 = Very weak/Negligible correlation
- ±0.20 - 0.39 = Weak correlation
- ±0.40 - 0.59 = Moderate correlation
- ±0.60 - 0.79 = Strong correlation
- ±0.80 - 0.99 = Very strong correlation


1
0.8
0.4
0
-0.4
-0.8
-1
1
1
1
-1
-1
-1
0
0
0
0
0
0
0

Difference between Spearman & Pearson

- Pearson's correlation relies on the use of means and variance (**assuming normal distribution**), and therefore can be sensitive to outliers or non-linearity (relationships that aren't a straight line).
- Spearman's correlation uses the **rank** order association, so outliers will not affect it if the rank order remains the same.

```
x <- c(1, 2, 3, 4, 15)
y <- c(1, 2, 3, 4, 5)

cor(x, y, method = "pearson")
```

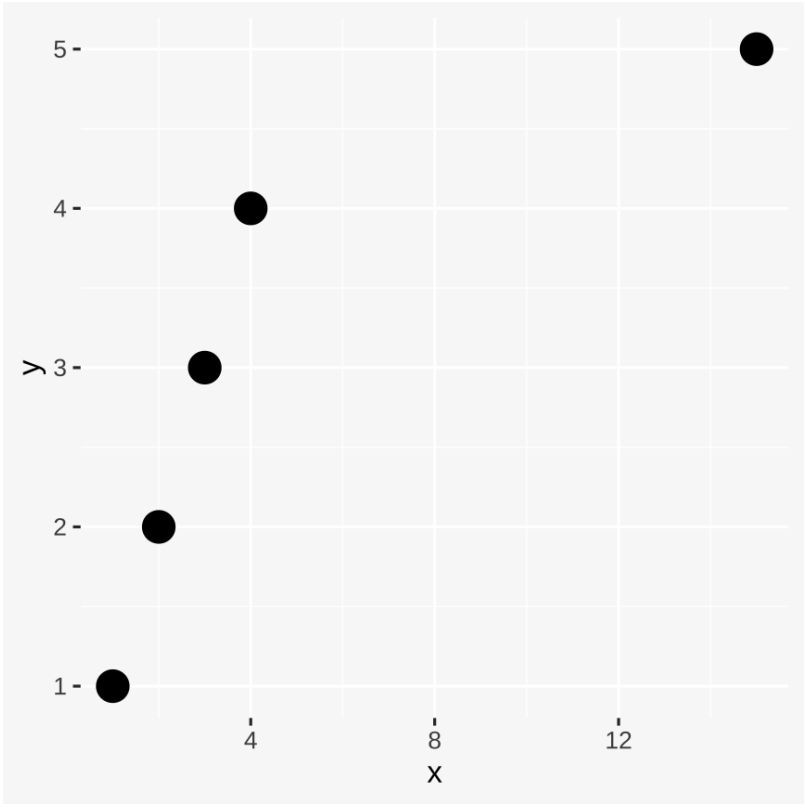
```
## [1] 0.8320503
```

```
cor(x, y, method = "spearman")
```

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

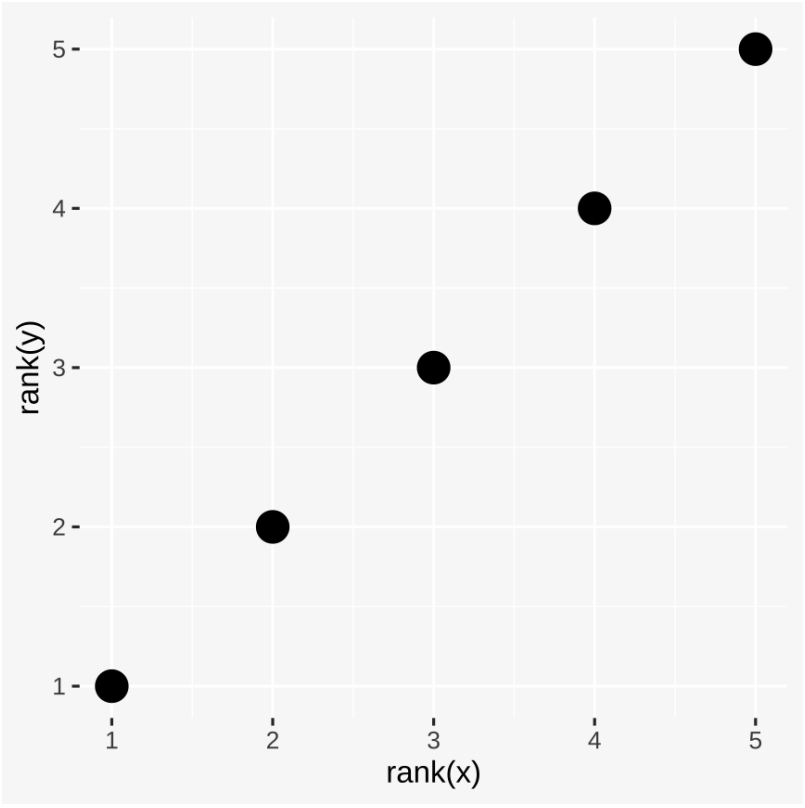
Pearson (raw values) = 0.832

```
ggplot() +
  geom_point(
    aes(x = x, y = y), size = 5
  )
```



Spearman (ranks) = 1

```
ggplot() +
  geom_point(
    aes(x = rank(x), y = rank(y)), size = 5
  )
```



Summary

Variable Type	Nominal	Ordinal	Continuous
Nominal	Contingency Table + Cramer's V		
Ordinal	Contingency Table + Cramer's V	Spearman's Rho/ Contingency Table	
Continuous	Mean/ Median Difference	Mean/ Median Difference	Pearson's R or Spearman's Rho

Summary

Variable Type	Nominal	Ordinal	Continuous
Nominal	Heatmap/ Bivariate Bar Chart		
Ordinal	Heatmap/ Bivariate Bar Chart	Heatmap/ Bivariate Bar Chart	
Continuous	Boxplot/ Ridgeplot	Boxplot/ Ridgeplot	Scatterplot/ Hex Bin Plot

Summary

- There are a large range of ways to visualise and statistically describe relationships between variables in `R`; **appropriate methods can be chosen by considering the type of variables we wish to analyse using what we learned in Week 2.**
- **Visualisations of relationships** between variables can help us get a full and proportionate picture of relationships, whereas **bivariate descriptive statistics** (contingency tables, Spearman's Rho, Cramer's V, and Pearson's R) can give us a standardised, shorthand way of describing what these relationships are.
- Note: I don't expect you to remember all of these different methods: **Use the table and the slides as a cheat sheet**! When viewing the slides on your computer you can press "o" on your keyboard to quickly browse through them all.

R Exercise

- This week, the focus is on **interpreting output** from `R` in a quantitative research context - not on writing `R` code. This is an opportunity to take a bit of a break from code-writing in class, but I would encourage you to experiment on your own or through textbook examples.

- **Download and unzip the Week 3 R Exercise files from the Blackboard page and open the .Rproj and week-3.Rmd files in Rstudio.**