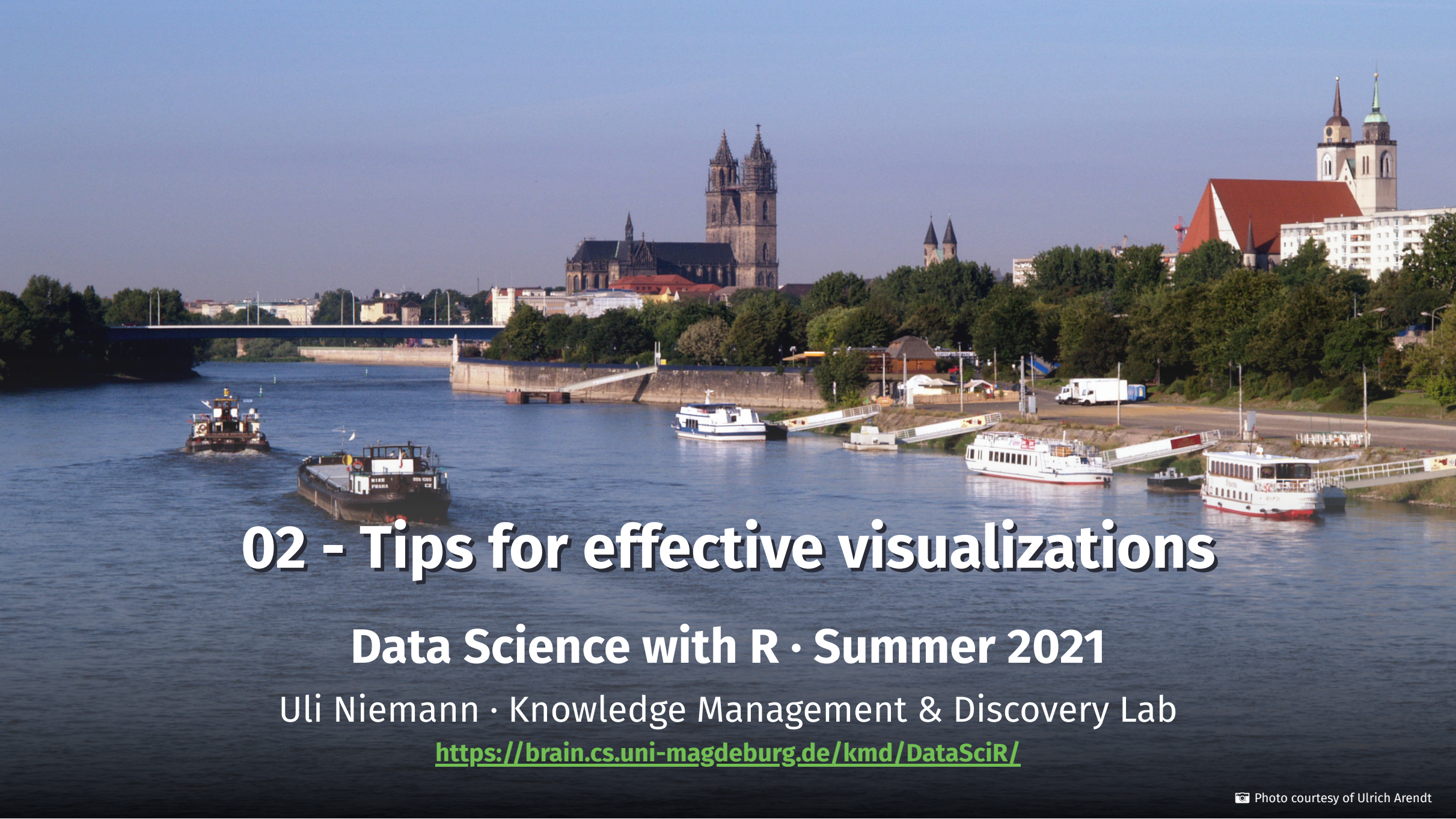


02 - Tips for effective visualizations

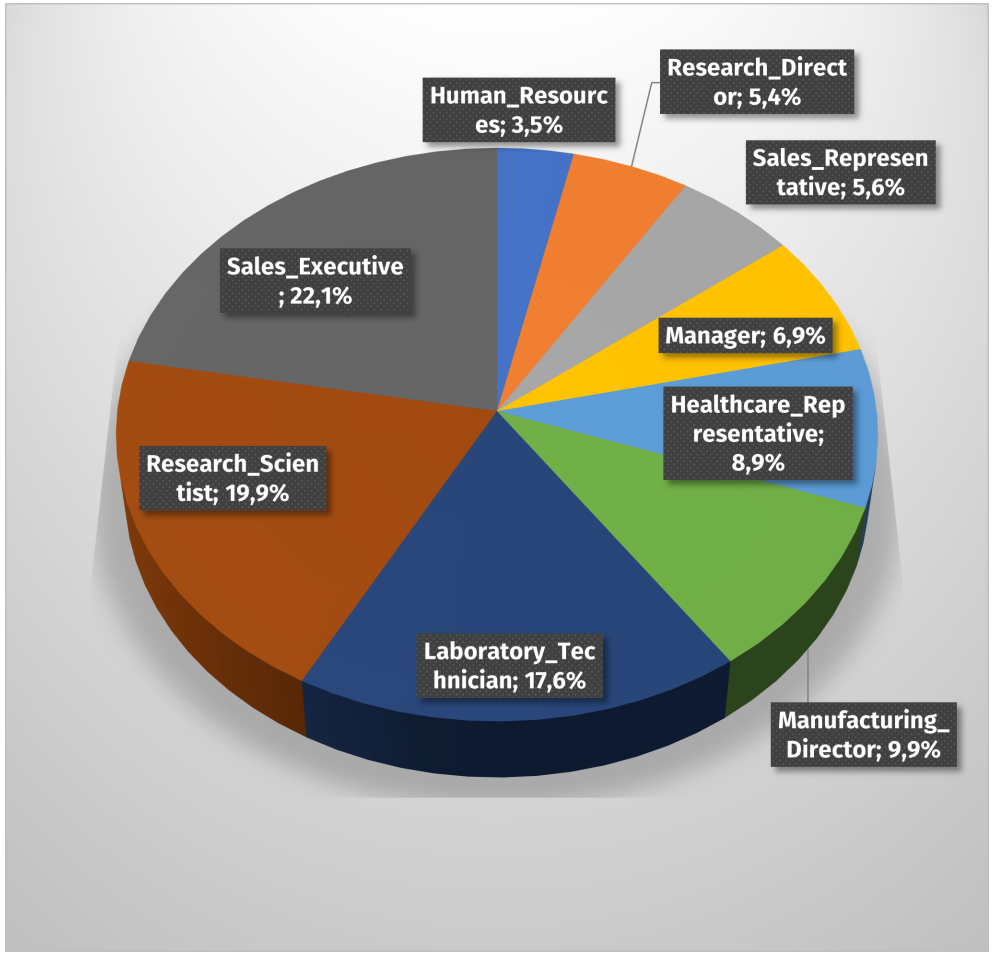
Data Science with R · Summer 2021

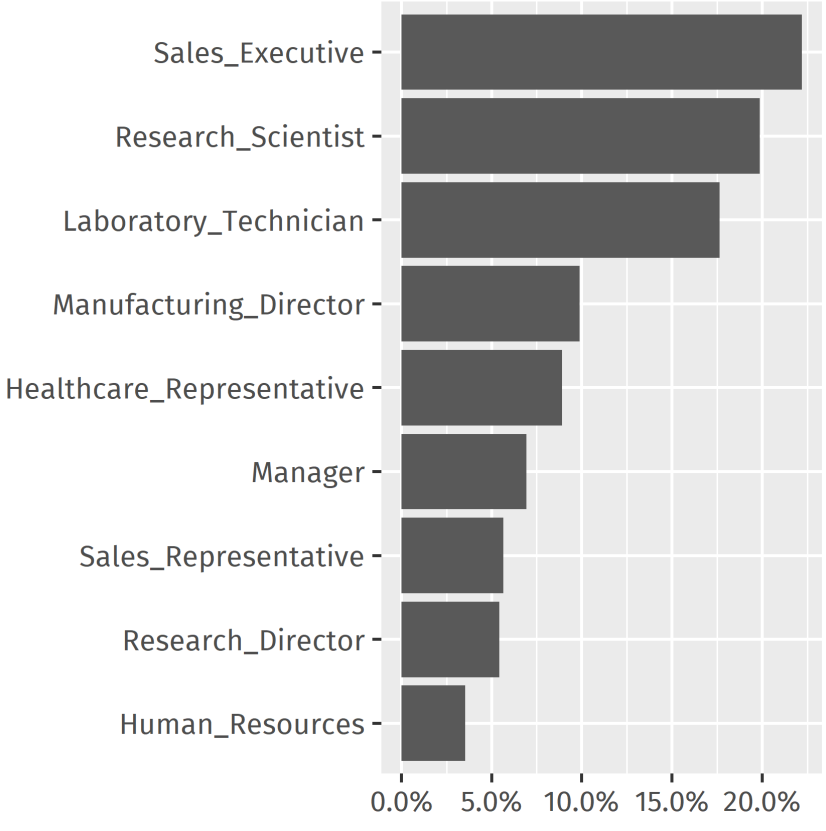
Uli Niemann · Knowledge Management & Discovery Lab

https://brain.cs.uni-magdeburg.de/kmd/DataSciR/

Photo courtesy of Ulrich Arendt

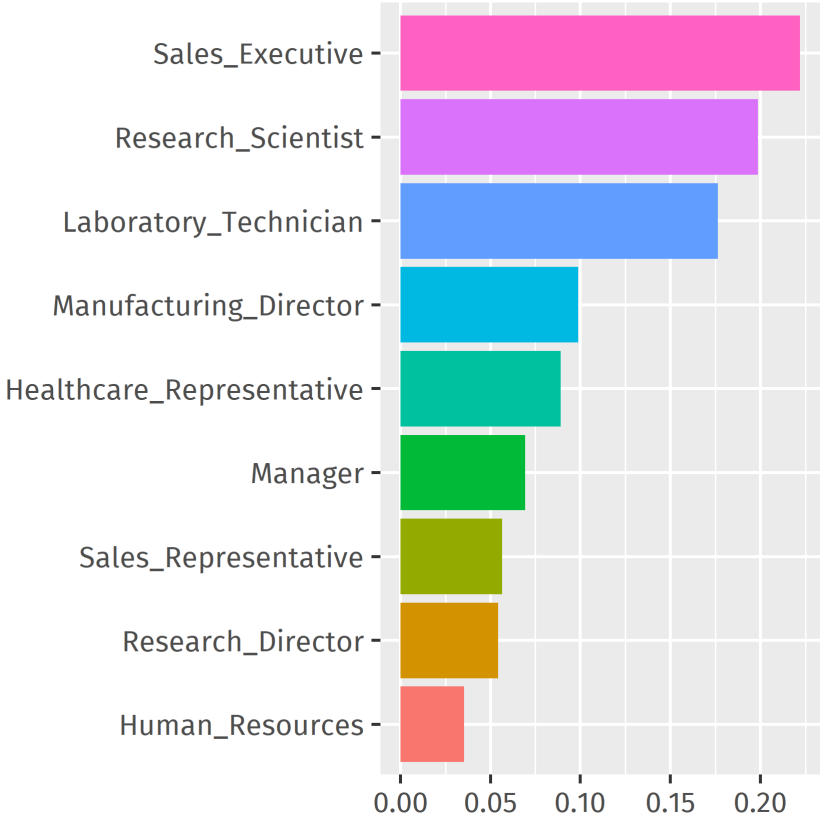
Keep it simple

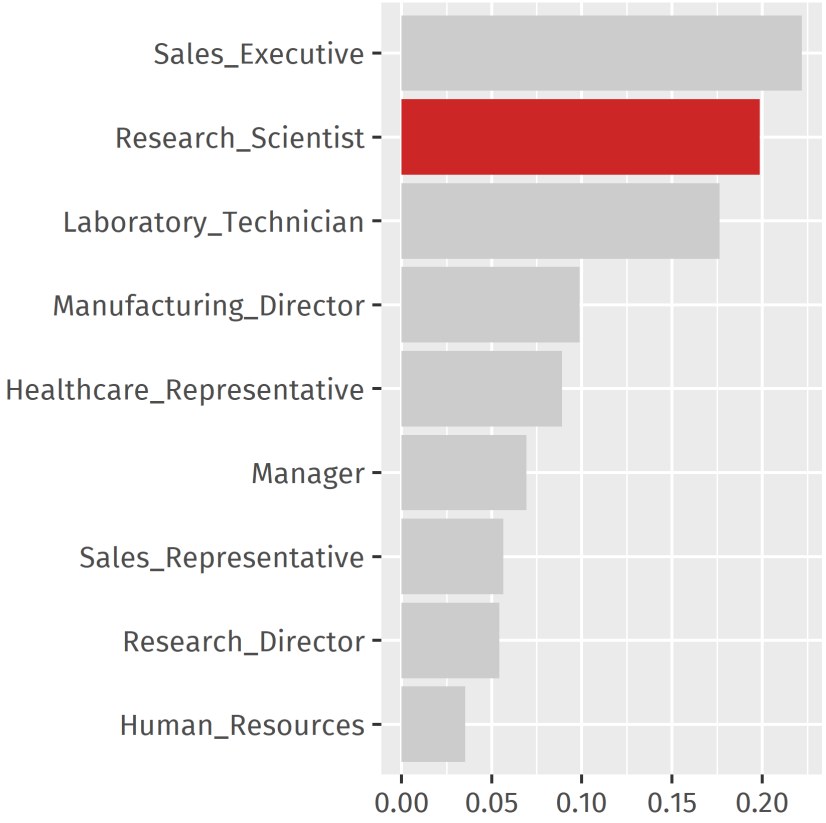




Slides adapted from: [Introduction to Data Science Course 2020 @ Univ. Edinburgh]

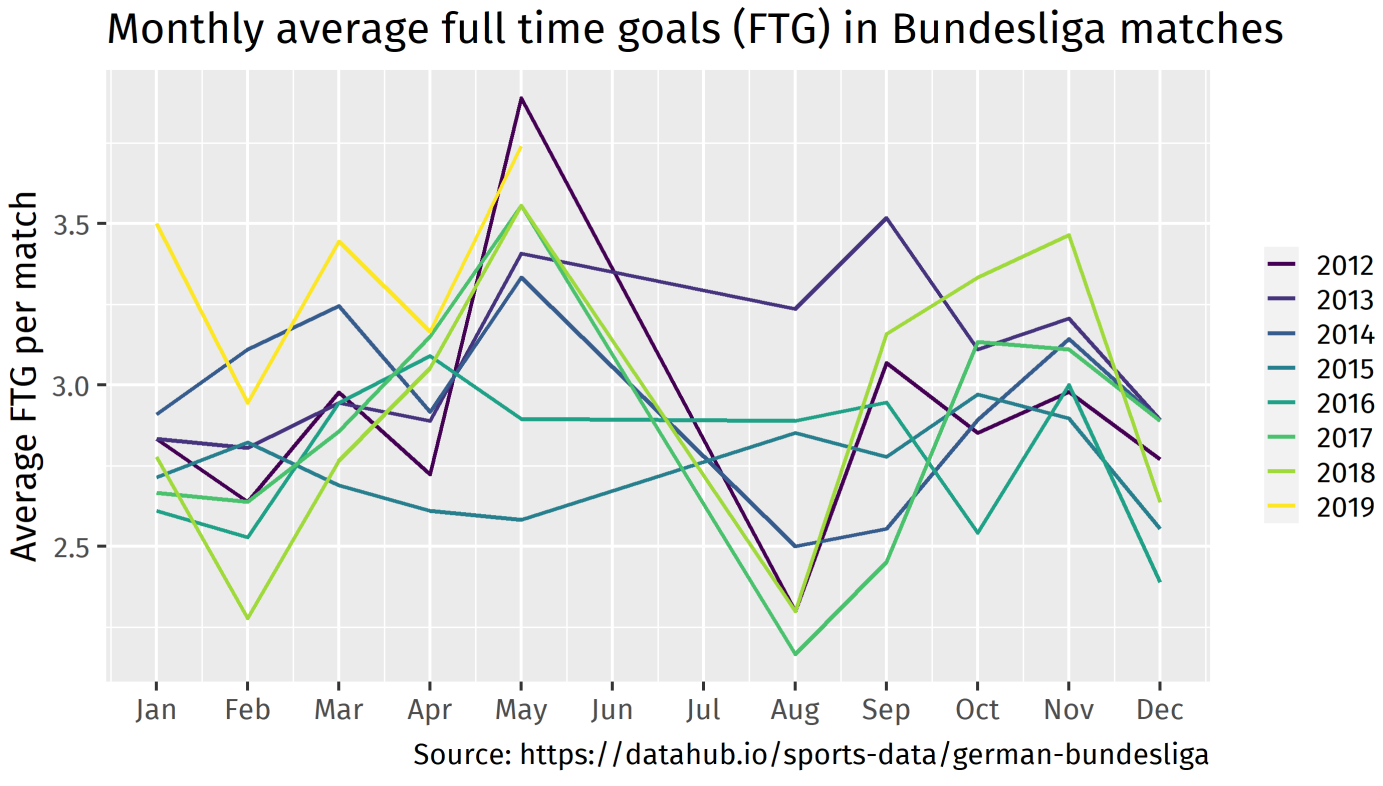
Use color to draw attention





Tell a story

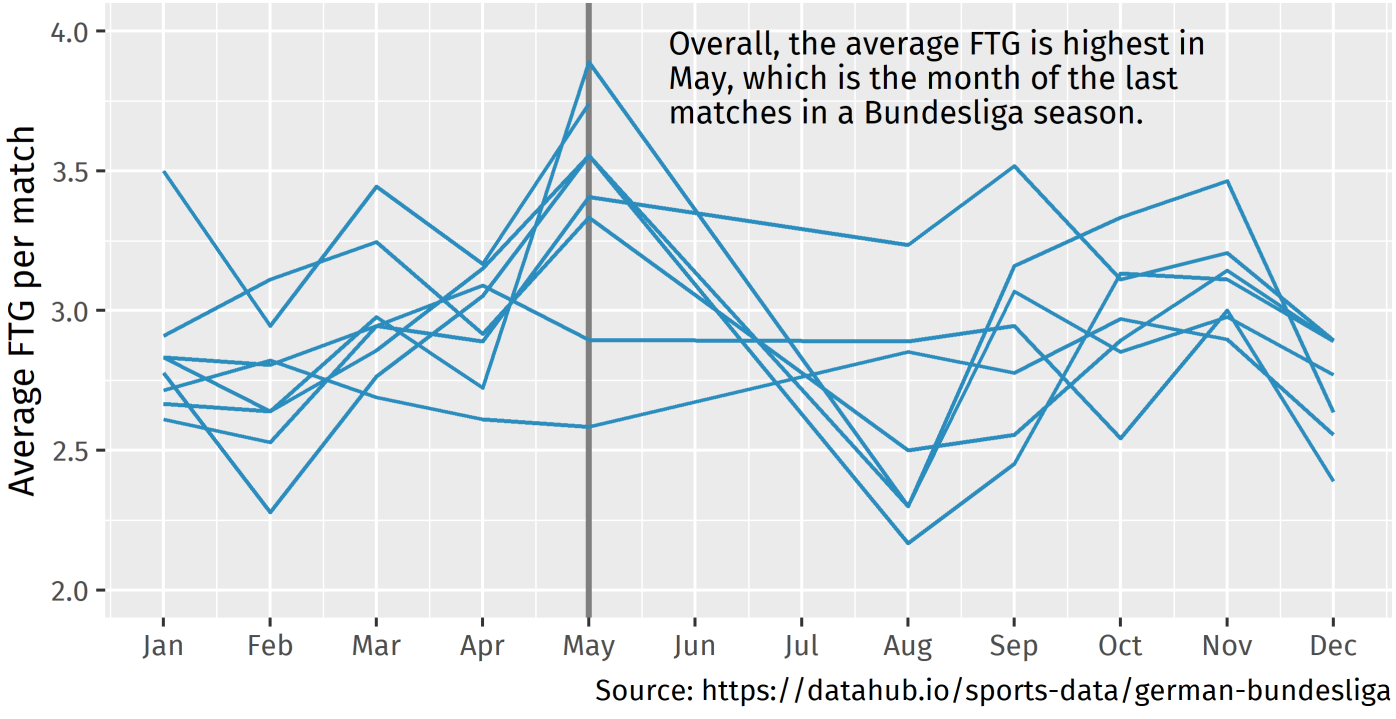
Does the year matter? | Plot annotation



Tell a story

Does the year matter? | Plot annotation

Monthly average full time goals (FTG) in Bundesliga matches (2012 - 2019)

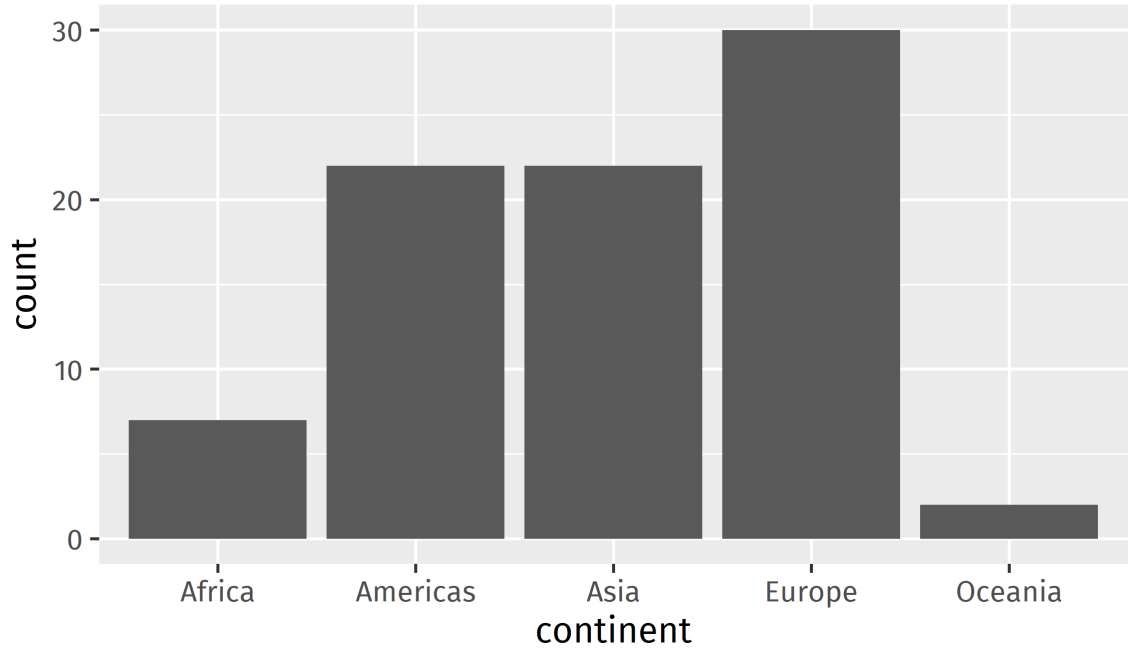


Principles for effective visualizations

- Order matters
- Put long categories on the y-axis
- Keep scales consistent
- Select meaningful colors
- Use meaningful and nonredundant labels

Alphabetical order is rarely ideal

Plot | Code



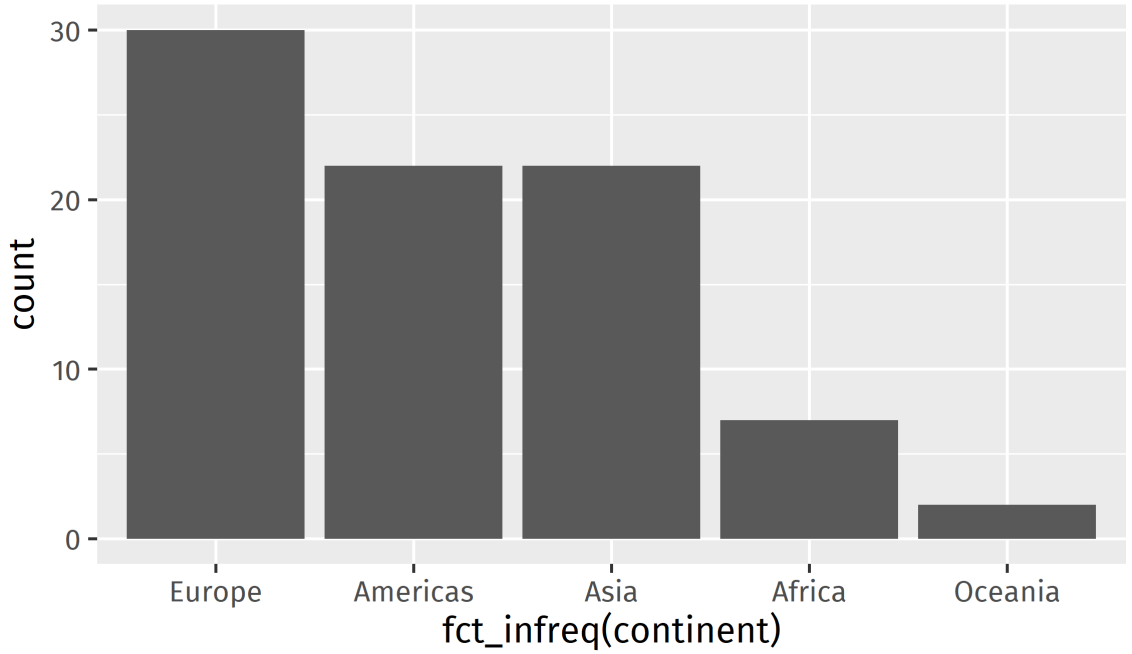
Alphabetical order is rarely ideal

Plot | Code

```
library(gapminder)
l97 <- filter(gapminder, year == 2007, lifeExp > 70)

ggplot(l97, aes(x = continent)) +
  geom_bar()
```

Order by frequency

Plot | Code



Order by frequency

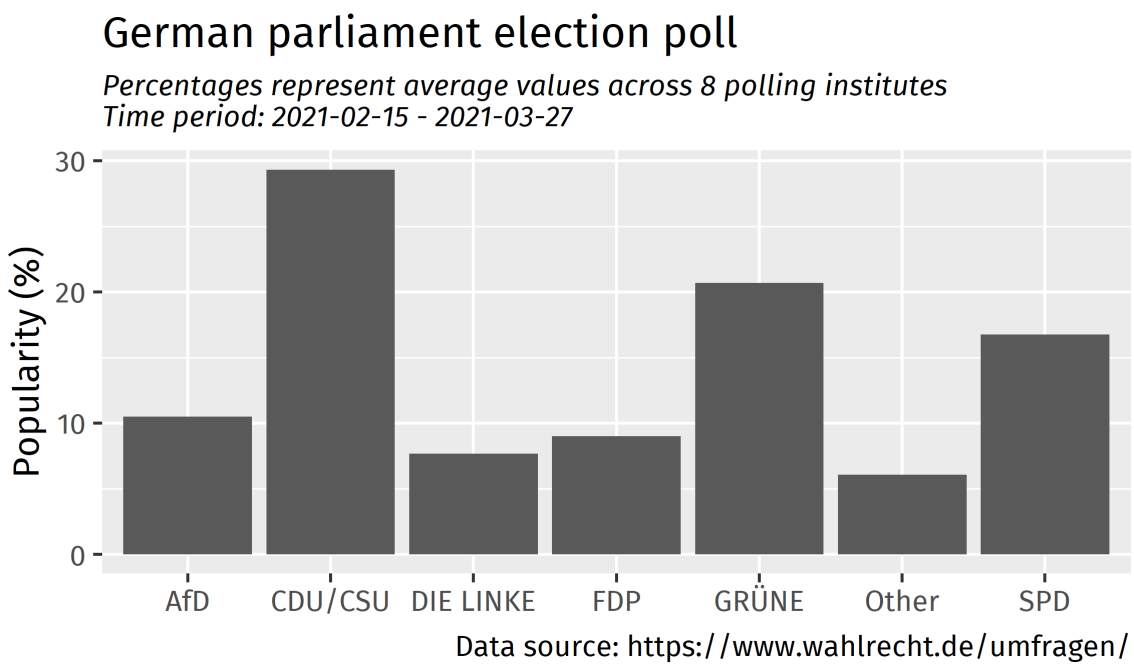
Plot | Code

`fct_infreq()`: Reorder factor levels by frequency.

```
ggplot(l97, aes(x = fct_infreq(continent))) +
  geom_bar()
```

Alphabetical order is rarely ideal

5 Plot | 3 Code to prep data | 4 Code for plot | 1 Poll aggregator website | 2 Code to scrape data



Since we're using `geom_col()` we can't use `fct_infreq()` because every category (i.e. party) appears exactly in one and only one observation.

Alphabetical order is rarely ideal

5 Plot | 3 Code to prep data | 4 Code for plot | 1 Poll aggregator website | 2 Code to scrape data

```
umfrage <- read_rds(
  here::here("data", "umfrage.rds")
)
umfrage
```

```
## # A tibble: 56 x 3
##    party   pollster          popularity
##    <chr>   <chr>                  <dbl>
##  1 CDU/CSU Allensbach              28.5
##  2 CDU/CSU Kantar(Emnid)           25
##  3 CDU/CSU Forsa                   26
##  4 CDU/CSU Forsch'gr.Wahlen        28
##  5 CDU/CSU GMS                     37
##  6 CDU/CSU Infratestdimap          29
##  7 CDU/CSU INSA                    28
##  8 CDU/CSU Yougov                  33
##  9 SPD     Allensbach              18
## 10 SPD     Kantar(Emnid)           17
## # ... with 46 more rows
```

```
(date_range <- attr(umfrage, "date_range"))
```

```
## [1] "2021-02-15" "2021-03-27"
```

```
(date_range_chr <- paste0(
  date_range, collapse = " - "))
```

```
## [1] "2021-02-15 - 2021-03-27"
```

```
umfrage_avg <- umfrage %>%
  group_by(party) %>%
  summarize(popularity = mean(popularity)) %>%
  ungroup()
umfrage_avg
```

```
## # A tibble: 7 x 2
##   party     popularity
##   <chr>          <dbl>
## 1 AfD             10.5
## 2 CDU/CSU         29.3
## 3 DIE LINKE        7.69
## 4 FDP              9
## 5 GRÜNE           20.7
## 6 Other            6.06
## 7 SPD             16.8
```

Alphabetical order is rarely ideal

5 Plot | 3 Code to prep data | **4 Code for plot** | 1 Poll aggregator website | 2 Code to scrape data

```
ggplot(umfrage_avg, aes(x = party, y = popularity)) +
  geom_col() +
  labs(
    x = NULL,
    y = "Popularity (%)",
    title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/"
  ) +
  theme(plot.subtitle = element_text(size = rel(0.8), face = "italic"))
```

Alphabetical order is rarely ideal

5 Plot | 3 Code to prep data | 4 Code for plot | **1 Poll aggregator website** | 2 Code to scrape data

https://www.wahlrecht.de/umfragen/

Sonntagsfrage Bundestagswahl

Wenn am nächsten Sonntag Bundestagswahl wäre …

Institut	Allensbach	Kantar (Emnid)	Forsa	Forsch'gr. Wahlen	GMS	Infratest dimap	INSA	Yougov	Bundes-tagswahl
Veröffentl.	28.01.2021	07.02.2021	02.02.2021	28.01.2021	04.01.2021	04.02.2021	02.02.2021	04.02.2021	24.09.2017
CDU/CSU	37 %	36 %	37 %	37 %	37 %	34 %	36,5 %	36 %	32,9 %
SPD	16 %	16 %	15 %	15 %	16 %	15 %	15 %	15 %	20,5 %
GRÜNE	20 %	19 %	19 %	20 %	18 %	21 %	17 %	18 %	8,9 %
FDP	6,5 %	7 %	6 %	6 %	6 %	8 %	8 %	7 %	10,7 %
DIE LINKE	7,5 %	7 %	8 %	7 %	8 %	6 %	7,5 %	9 %	9,2 %
AfD	9 %	9 %	8 %	9 %	9 %	10 %	11 %	10 %	12,6 %
Sonstige	4 %	6 %	7 %	6 %	6 %	6 %	5 %	5 %	5,0 %
Erhebung	F • 1.080 10.01.–20.01.	T • 1.412 28.01.–03.02.	T • 2.503 26.01.–02.02.	T • 1.371 25.01.–27.01.	T • 1.004 29.12.–04.01.	T • 1.503 01.02.–03.02.	O • 2.044 29.01.–01.02.	O • 1.606 29.01.–01.02.	

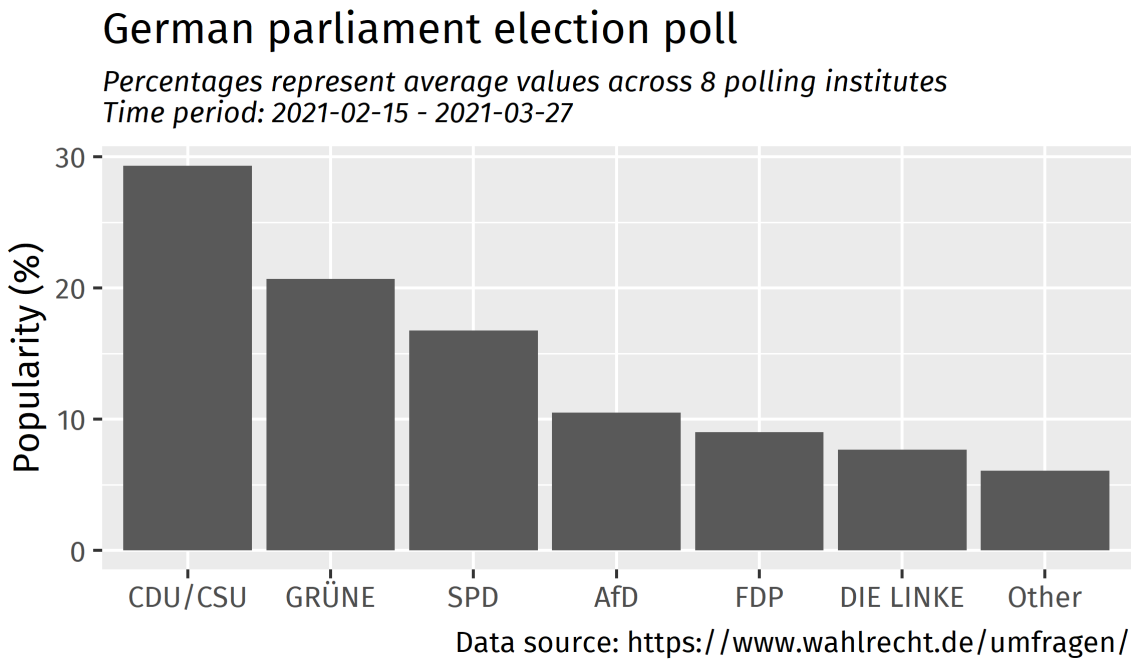
Alphabetical order is rarely ideal

5 Plot | 3 Code to prep data | 4 Code for plot | 1 Poll aggregator website | **2 Code to scrape data**

```
library(rvest)
umfrage <- read_html("https://www.wahlrecht.de/umfragen/") %>%
  html_node(".wilko") %>%
  html_table()
umfrage[names(umfrage) == ""] <- NULL
umfrage[length(umfrage)] <- NULL # "Letzte Bundestagswahl"

date_range <- as.character(range(lubridate::dmy(as.character(umfrage[1, ][-1]))))

umfrage <- umfrage %>%
  filter(Institut %in% c("CDU/CSU", "SPD", "GRÜNE", "FDP", "DIE LINKE", "AfD", "Sonstige")) %>%
  rename(party = Institut) %>%
  pivot_longer(cols = -party, names_to = "pollster", values_to = "popularity") %>%
  mutate(popularity = str_replace(popularity, ",", "\\.")) %>%
  mutate(popularity = str_remove(popularity, " %")) %>%
  mutate(popularity = as.double(popularity)) %>%
  mutate(party = ifelse(party == "Sonstige", "Other", party))

attr(umfrage, "date_range") <- date_range

write_rds(umfrage, here::here("data", "umfrage.rds"))
```

Order by a second variable

Plot | Code



Order by a second variable

Plot | Code

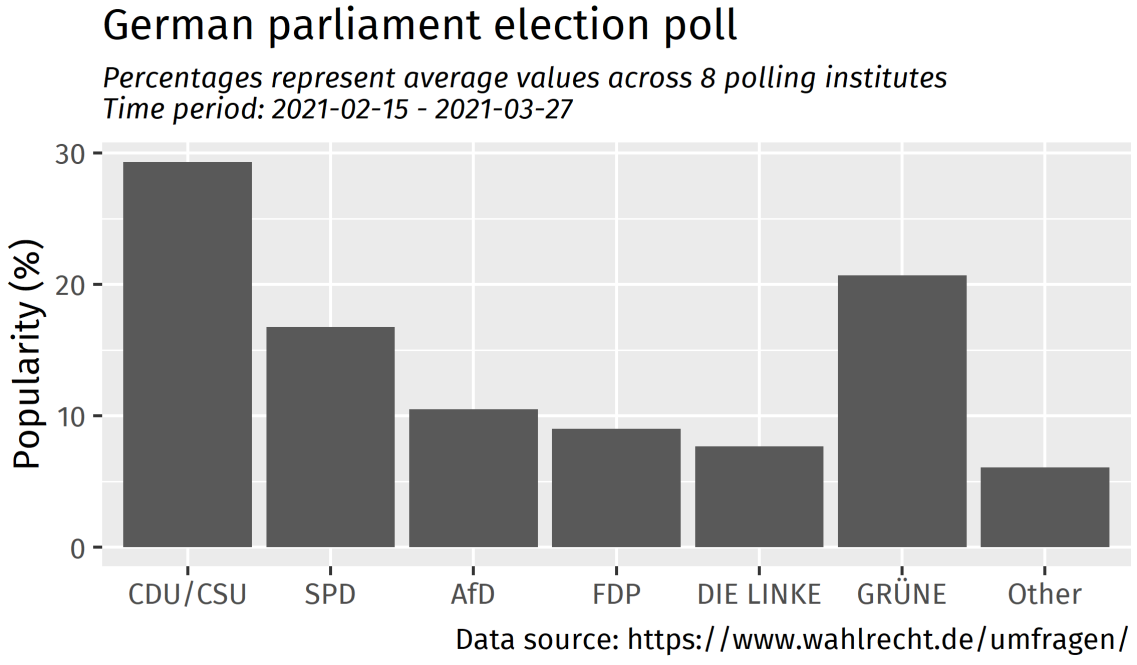
`fct_reorder()`: Reorder factor levels by another numeric variable. Use `-` to sort in descending order.

```
ggplot(
  umfrage_avg,
  aes(
    x = fct_reorder(party, -popularity),
    y = popularity
  )
) +
  geom_col() +
  labs(
    x = NULL,
    y = "Popularity (%)",
    title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/"
  ) +
  theme(plot.subtitle = element_text(size = rel(0.8), face = "italic"))
```

Custom order

Sometimes you see in election polls that the parties are shown in the order of their vote shares in the previous election. For example, in the 2017 elections the SPD received the second most votes, whereas GRÜNE were only sixth.

Plot | Code



Custom order

Sometimes you see in election polls that the parties are shown in the order of their vote shares in the previous election. For example, in the 2017 elections the SPD received the second most votes, whereas GRÜNE were only sixth.

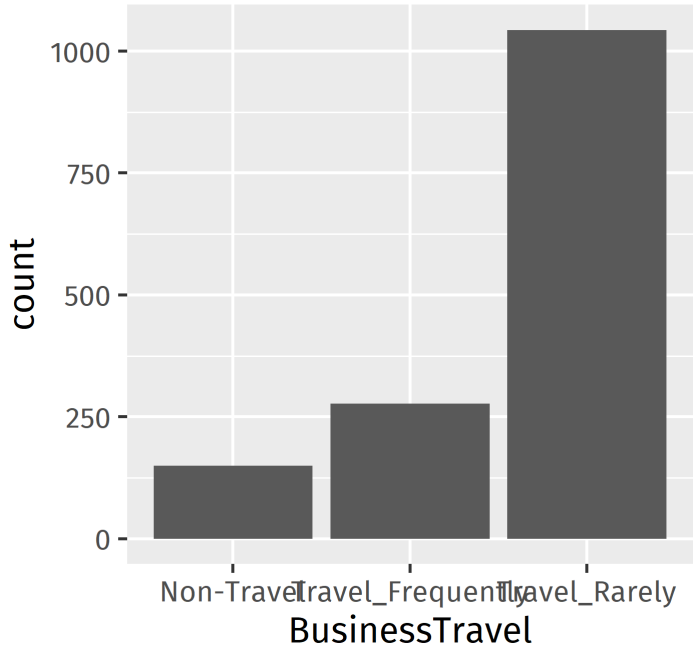
Plot | Code

`fct_relevel()`: Manually reorder factor levels.

```
umfrage_avg <- umfrage_avg %>%
  mutate(
    party = fct_relevel(party,
      "CDU/CSU", "SPD", "AfD", "FDP", "DIE LINKE", "GRÜNE", "Other"
    )
  )
ggplot(umfrage_avg, aes(x = party, y = popularity)) +
  geom_col() +
  labs(
    x = NULL,
    y = "Popularity (%)",
    title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/"
  ) +
  theme(plot.subtitle = element_text(size = rel(0.8), face = "italic"))
```

Factor levels often need to be cleaned up

Plot | Code



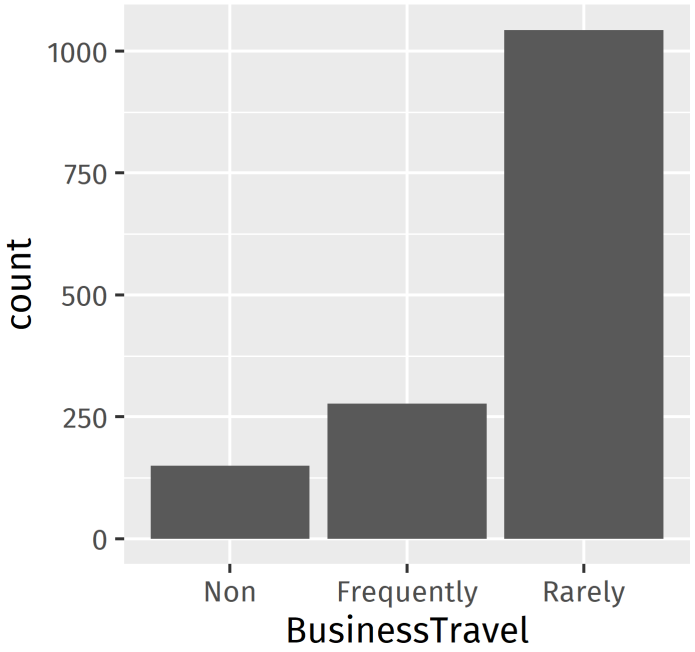
Factor levels often need to be cleaned up

Plot | Code

```
ggplot(attrition, aes(x = BusinessTravel)) +
  geom_bar()
```

Clean up labels

Plot | Code



Clean up labels

Plot | Code

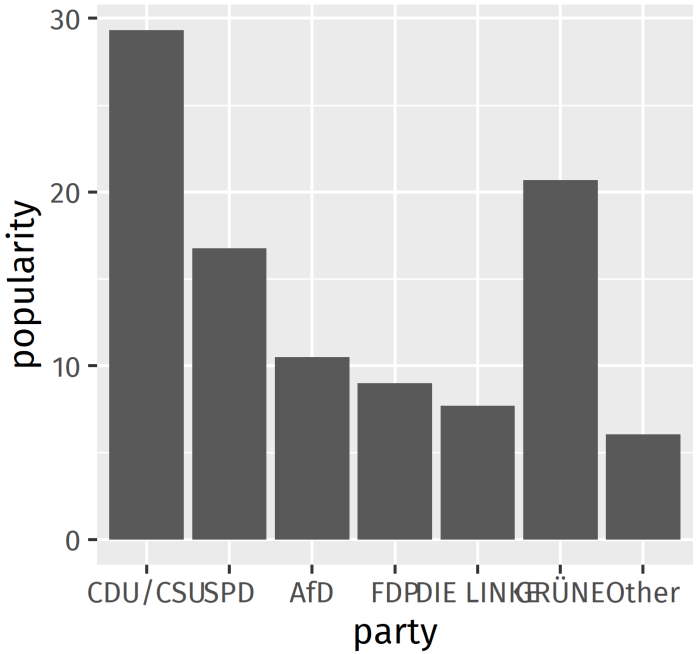
`fct_relevel()`: Manually reorder factor levels.

```
attrition <- attrition %>%
  mutate(
    BusinessTravel = fct_recode(
      BusinessTravel,
      "Frequently" = "Travel_Frequently",
      "Rarely" = "Travel_Rarely",
      "Non" = "Non-Travel"
    )
  )


ggplot(attrition, aes(x = BusinessTravel)) +
  geom_bar()
```

Put long and overlapping categories on the y-axis

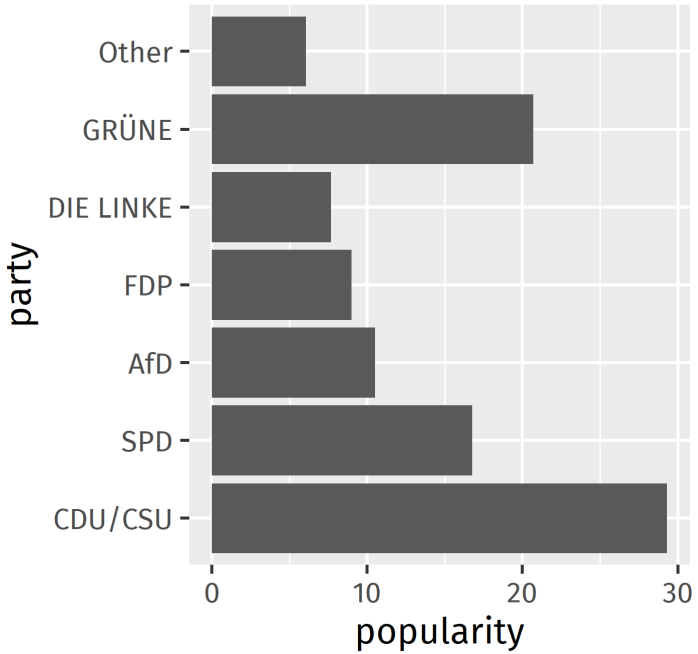
Categories on x-axis:

```
ggplot(
  umfrage_avg,
  aes(x = party, y = popularity)
) +
  geom_col()
```



Categories on y-axis:

```
ggplot(
  umfrage_avg,
  aes(x = popularity, y = party)
) +
  geom_col()
```

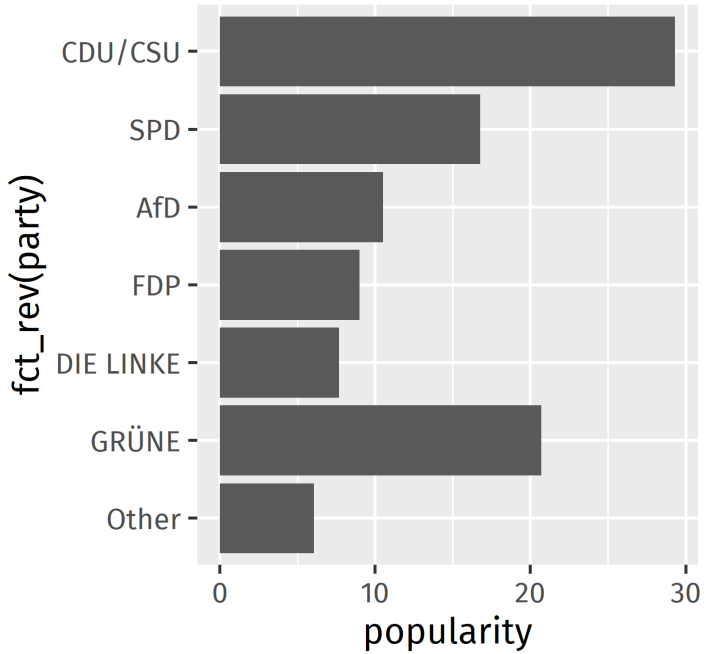


Reverse the order of levels

fct_rev() | Via scale setting

`fct_rev()`: Reverse the order of factor levels

```
ggplot(umfrage_avg, aes(x = popularity,
                        y = fct_rev(party))) +
  geom_col()
```

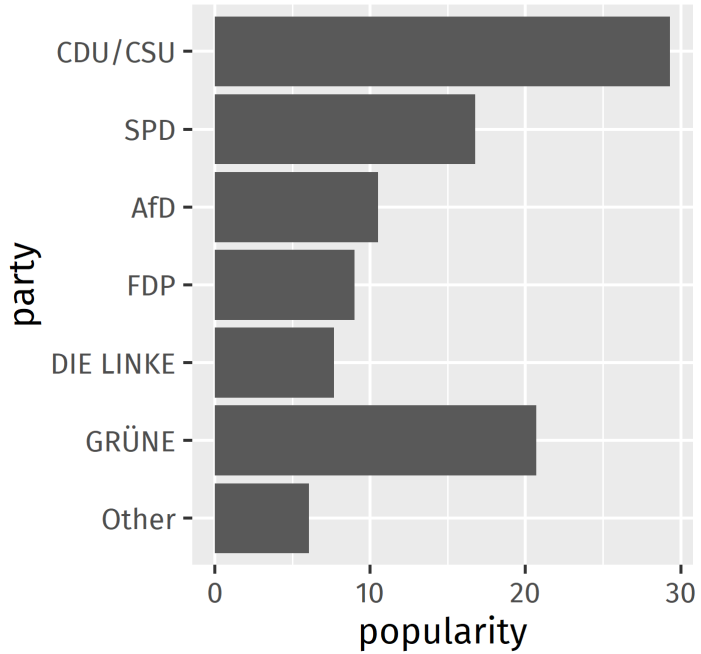


Reverse the order of levels

fct_rev() | Via scale setting

`rev()`: Reverse the order of values (any vector type)

```
ggplot(umfrage_avg, aes(x = popularity, y = party)) +
  geom_col() +
  scale_y_discrete(limits = rev)
```

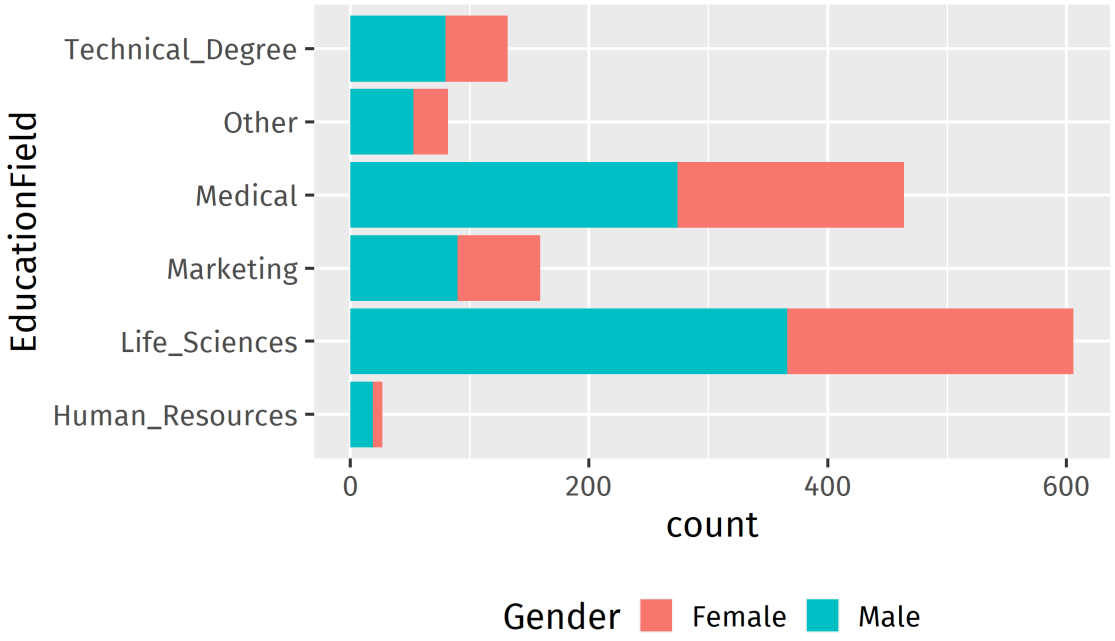


Before plotting, think about the purpose

Example: What is the number and share of women for each education field in the attrition data?

Stacked bars | Filled bars | Dodged bars | Facetted bars

```
ggplot(attrition, aes(y = EducationField, fill = Gender)) +
  geom_bar() +
  theme(legend.position = "bottom")
```

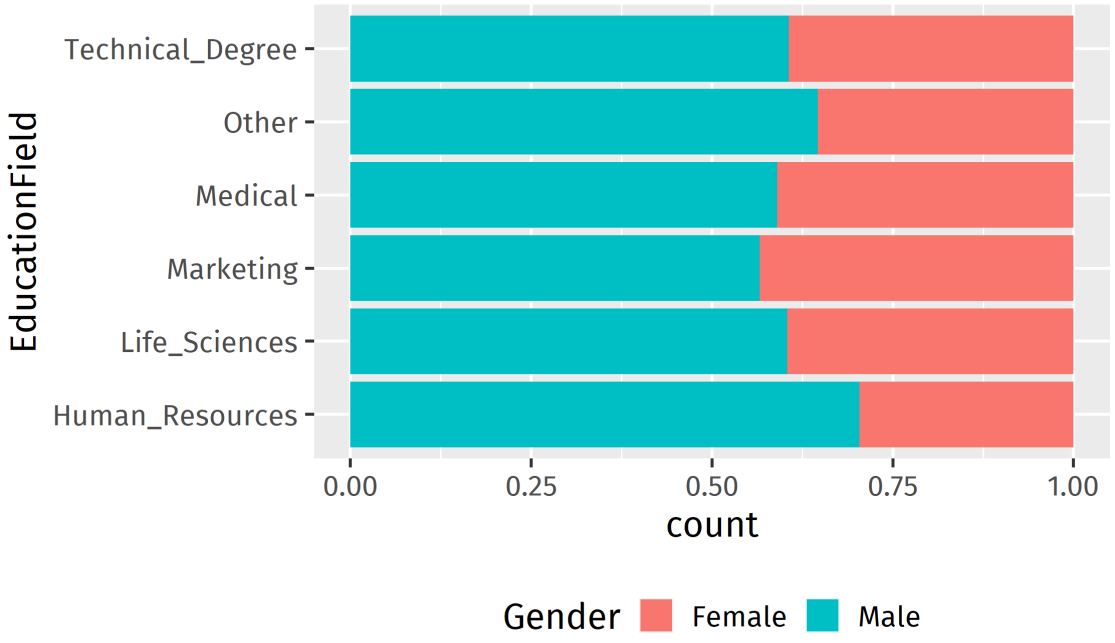


Before plotting, think about the purpose

Example: What is the number and share of women for each education field in the attrition data?

Stacked bars | **Filled bars** | Dodged bars | Facetted bars

```
ggplot(attrition, aes(y = EducationField, fill = Gender)) +
  geom_bar(position = "fill") +
  theme(legend.position = "bottom")
```

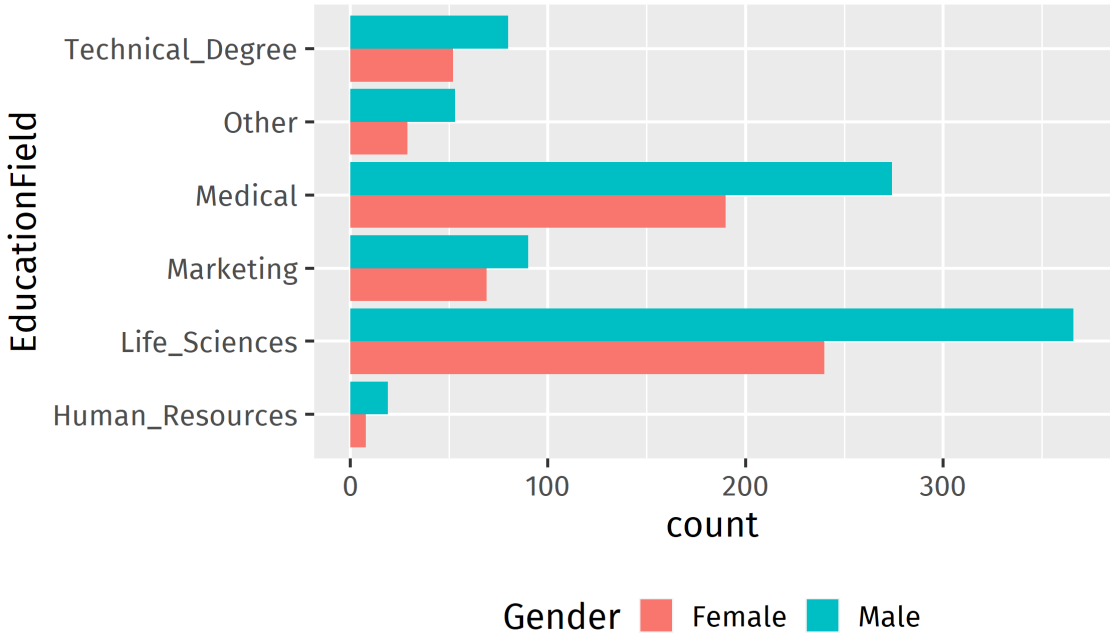


Before plotting, think about the purpose

Example: What is the number and share of women for each education field in the attrition data?

Stacked bars | Filled bars | **Dodged bars** | Facetted bars

```
ggplot(attrition, aes(y = EducationField, fill = Gender)) +
  geom_bar(position = "dodge") +
  theme(legend.position = "bottom")
```



Before plotting, think about the purpose

Example: What is the number and share of women for each education field in the attrition data?

Stacked bars | Filled bars | Dodged bars | **Facetted bars**

```
ggplot(attrition, aes(y = Gender, fill = Gender)) +
  geom_bar() +
  facet_wrap(~ EducationField) +
  theme(legend.position = "bottom")
```

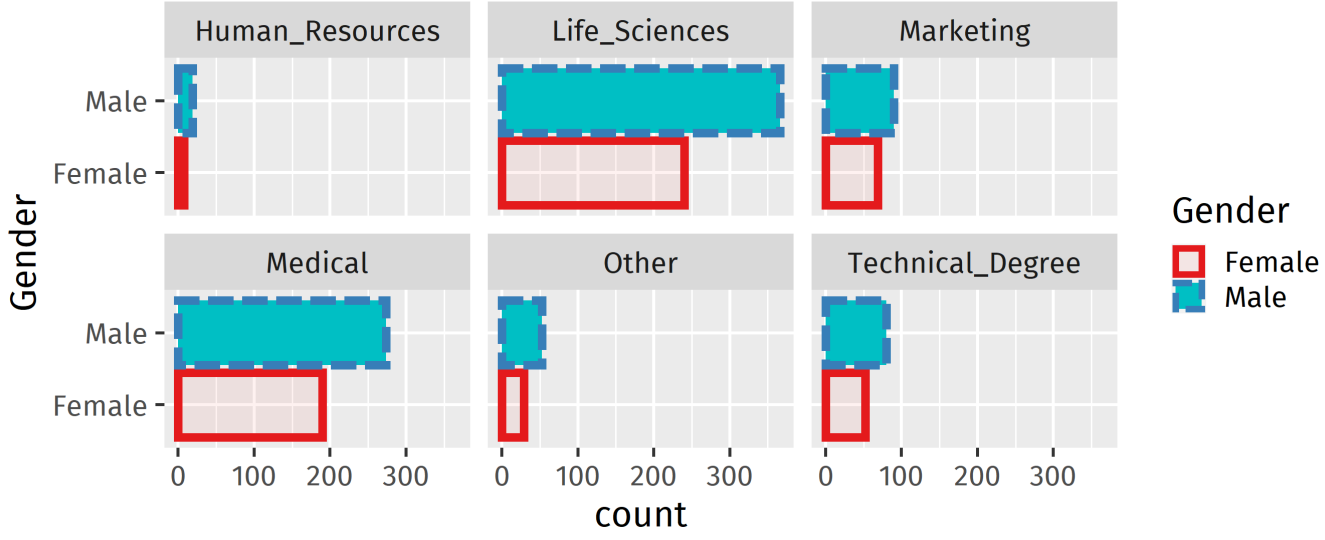
Avoid redundancy

High redundancy | Low redundancy

⃠ DON'T

```
ggplot(attrition,
  aes(y = Gender, fill = Gender, color = Gender, linetype = Gender, alpha = Gender)) +
  geom_bar(size = 2) +
  facet_wrap(~ EducationField) +
  scale_color_brewer(palette = "Set1")
```

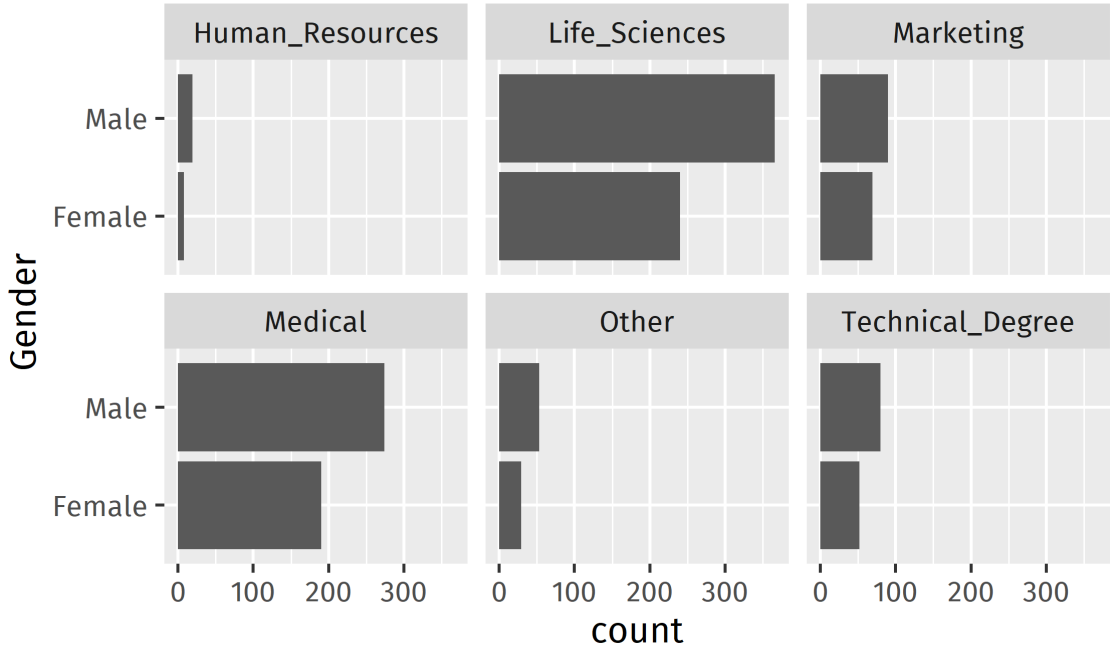
```
## Warning: Using alpha for a discrete variable is not advised.
```



Avoid redundancy

High redundancy | Low redundancy

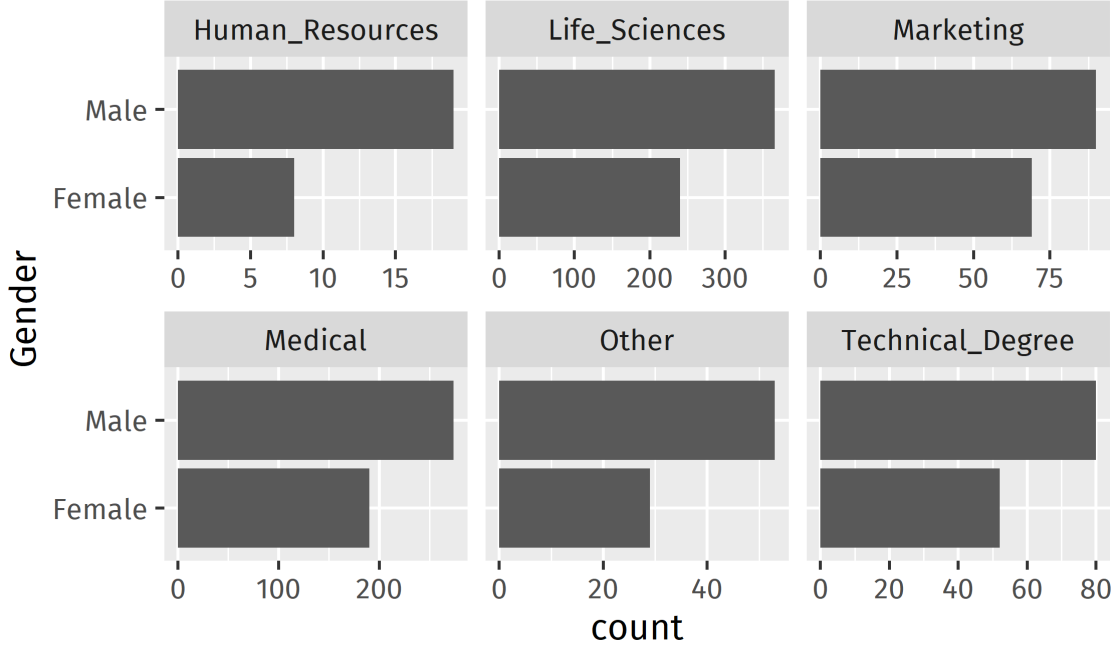
```
ggplot(attrition, aes(y = Gender)) +
  geom_bar() +
  facet_wrap(~ EducationField)
```



Keep scales consistent

⃠ DON'T

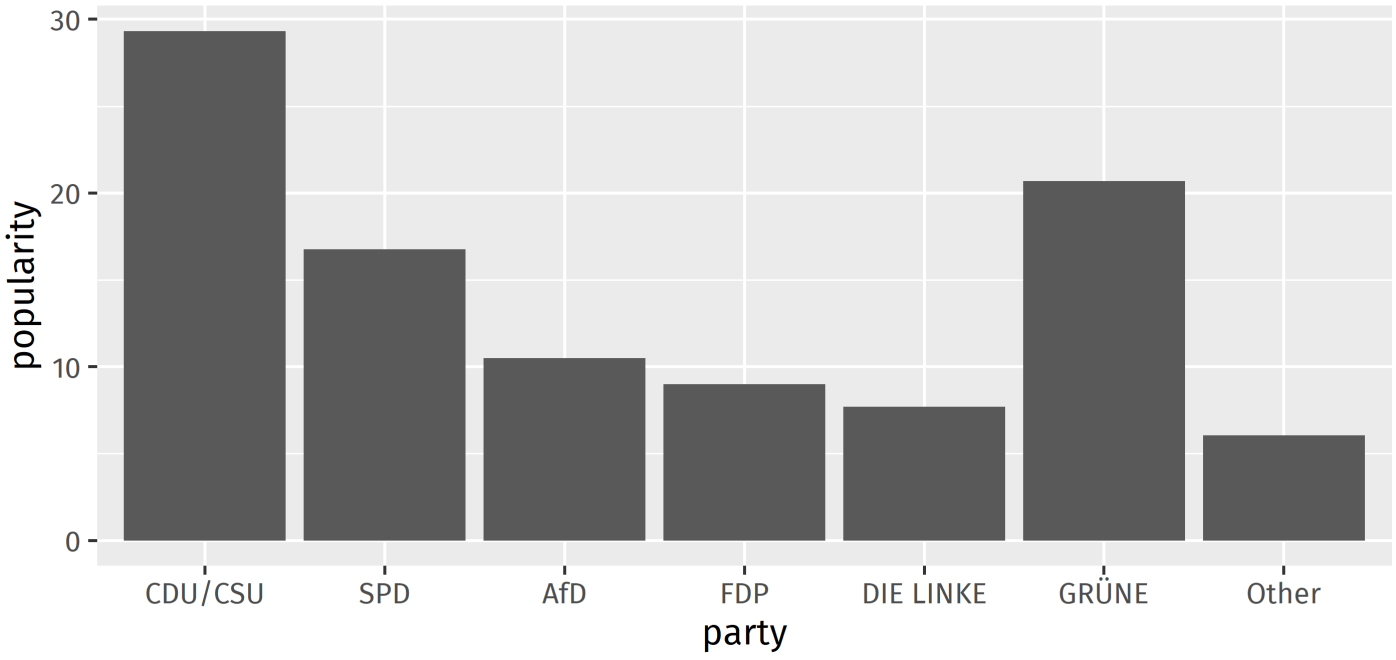
```
ggplot(attrition, aes(y = Gender)) +
  geom_bar() +
  facet_wrap(~ EducationField, scales = "free_x")
```



Use meaningful and nonredundant labels

Without context | With context

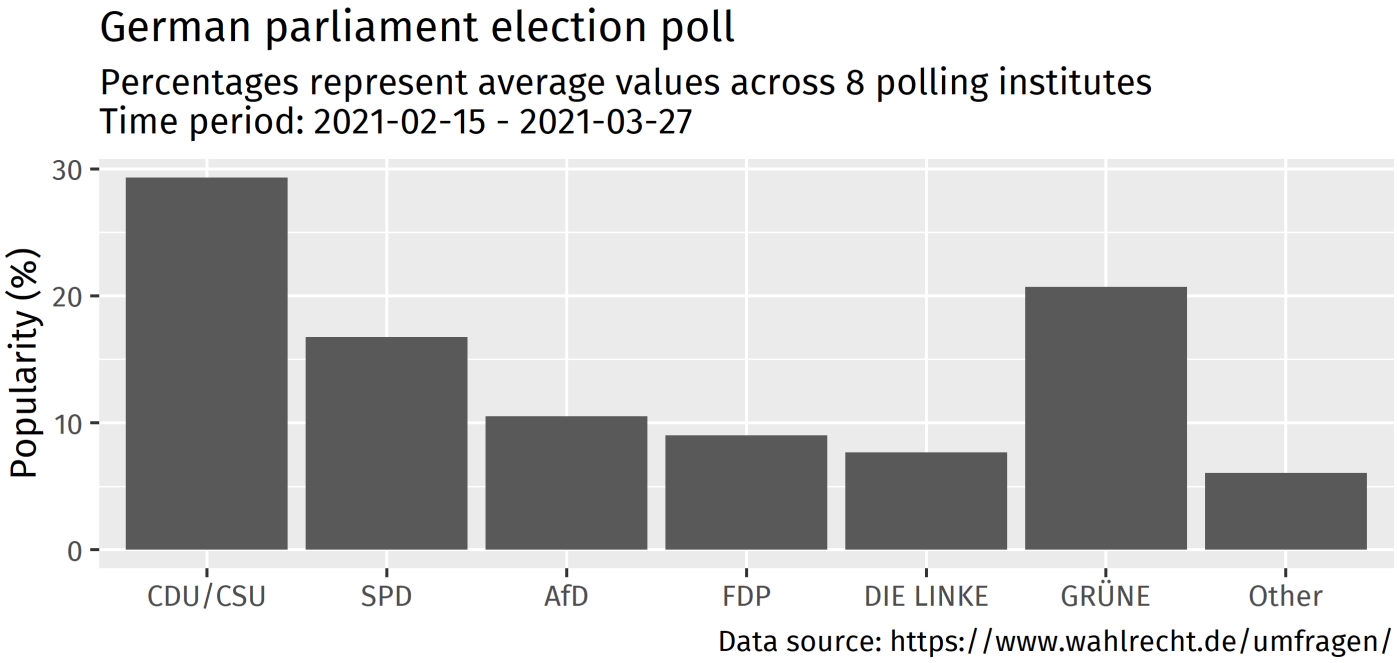
```
ggplot(umfrage_avg, aes(x = party, y = popularity)) +
  geom_col()
```



Use meaningful and nonredundant labels

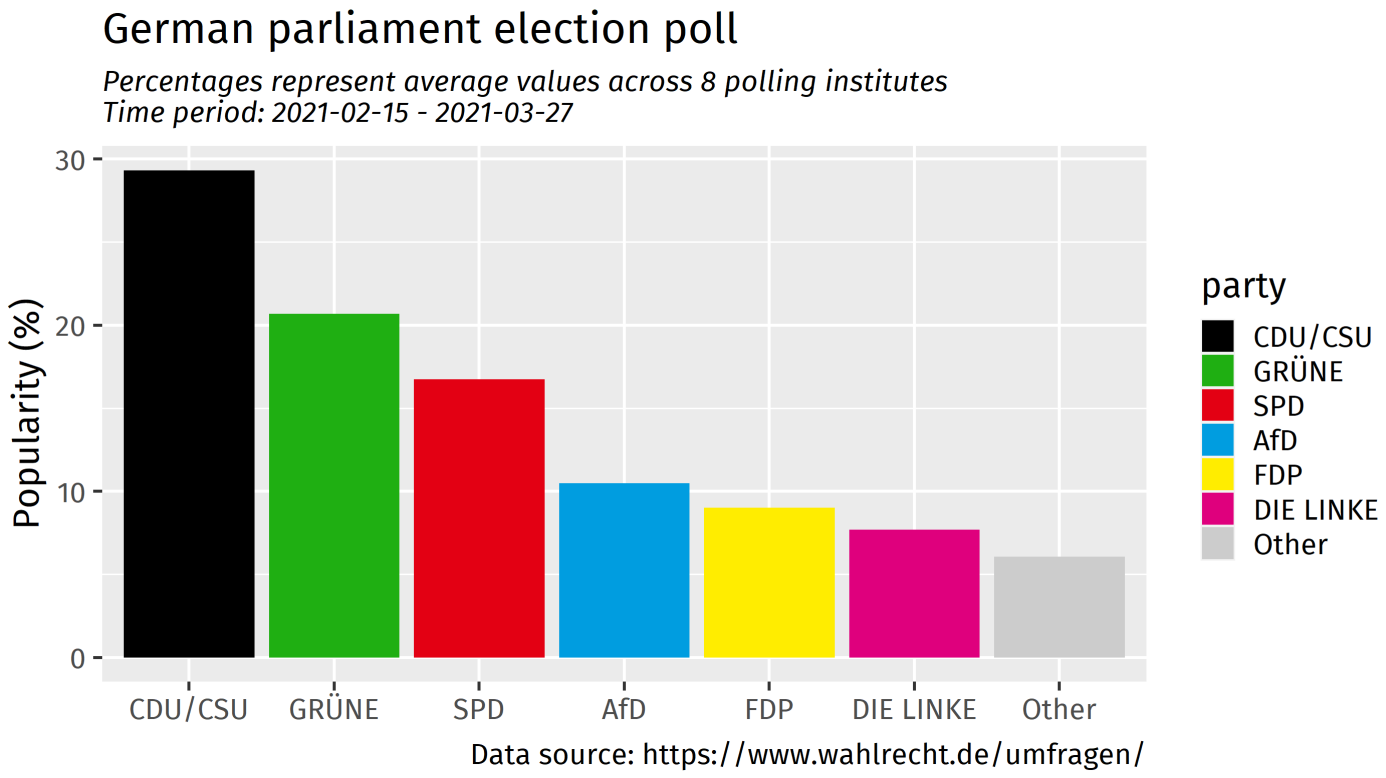
Without context | With context

```
ggplot(umfrage_avg, aes(x = party, y = popularity)) +
  geom_col()+
  labs(x = NULL, y = "Popularity (%)", title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/")
```



Select meaningful colors

Plot | Code



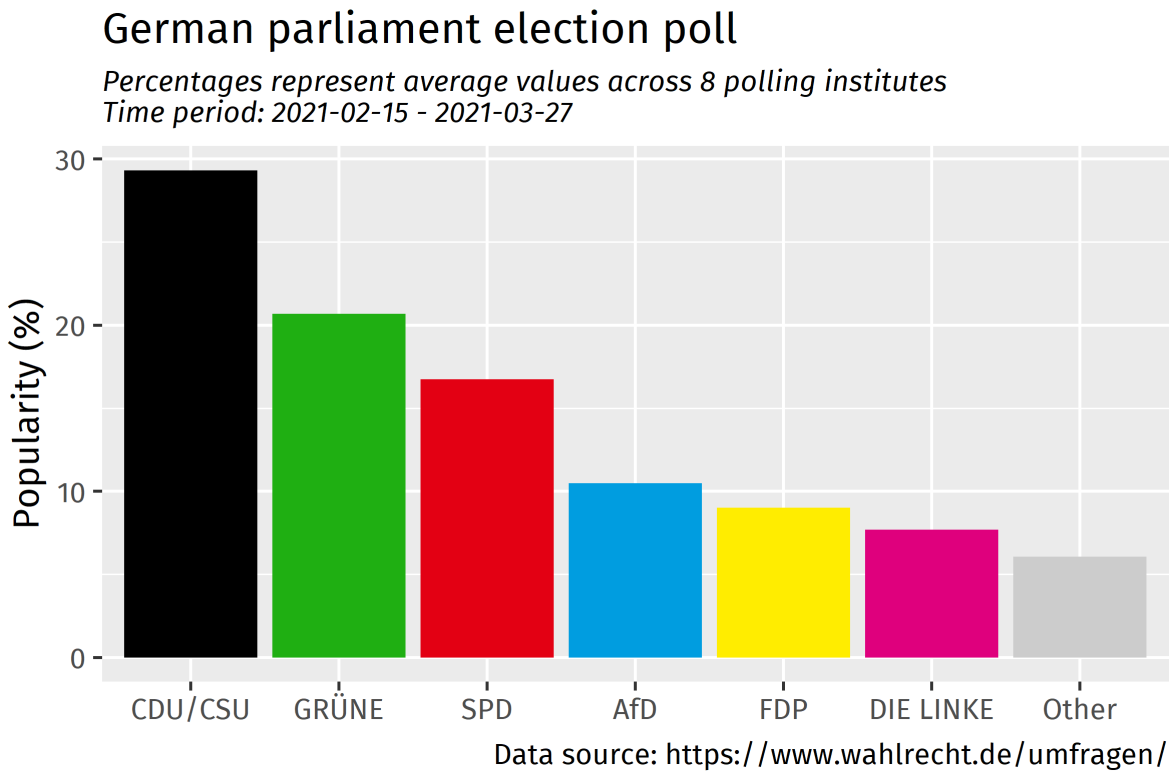
Select meaningful colors

Plot | Code

```
umfrage_avg <- umfrage_avg %>% mutate(party = fct_reorder(party, -popularity))
ggplot(umfrage_avg, aes(x = party, y = popularity, fill = party)) +
  geom_col() +
  labs(x = NULL, y = "Popularity (%)", title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/") +
  theme(plot.subtitle = element_text(size = rel(0.8), face = "italic")) +
  scale_fill_manual(values = c("CDU/CSU" = "#000000", "GRÜNE" = "#1FAF12", "SPD" = "#E30013",
"AfD" = "#009DE0", "DIE LINKE" = "#DF007D", "FDP" = "#FFED00", "Other" = "gray80"))
```

Be selective with redundancy

Plot | Code



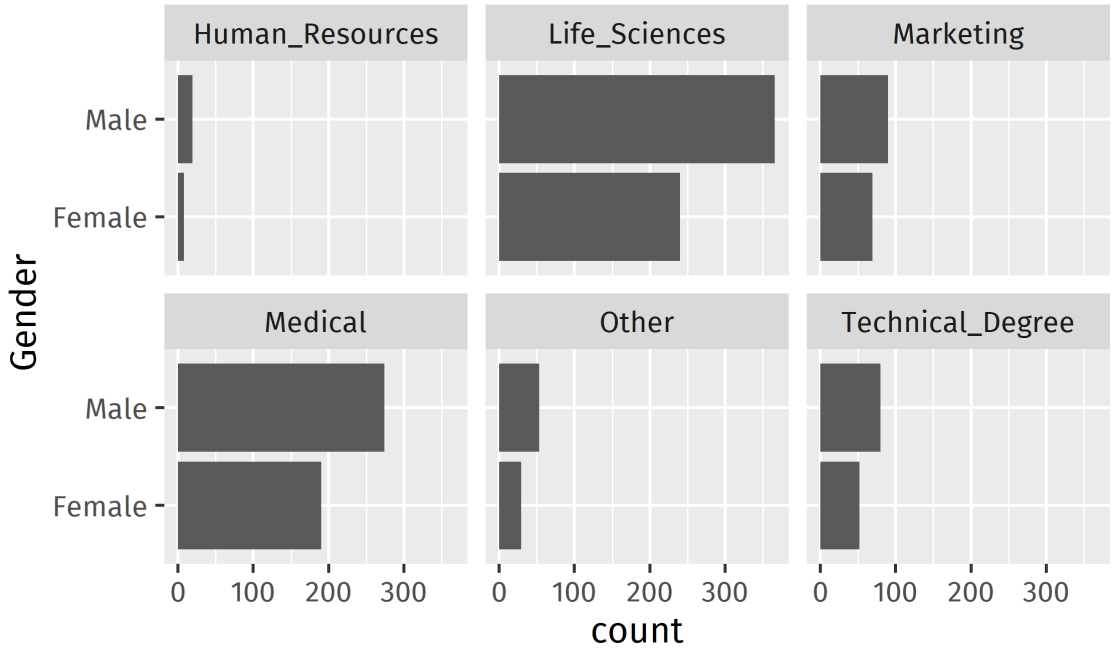
Be selective with redundancy

Plot | Code

```
umfrage_avg <- umfrage_avg %>% mutate(party = fct_reorder(party, -popularity))
ggplot(umfrage_avg, aes(x = party, y = popularity, fill = party)) +
  geom_col() +
  labs(x = NULL, y = "Popularity (%)", title = "German parliament election poll",
    subtitle = glue::glue("Percentages represent average values across 8 polling institutes\nTime period
    caption = "Data source: https://www.wahlrecht.de/umfragen/") +
  theme(plot.subtitle = element_text(size = rel(0.8), face = "italic")) +
  scale_fill_manual(values = c("CDU/CSU" = "#000000", "GRÜNE" = "#1FAF12", "SPD" = "#E30013",
"AfD" = "#009DE0", "DIE LINKE" = "#DF007D", "FDP" = "#FFED00", "Other" = "gray80")) +
  guides(fill = FALSE)
```

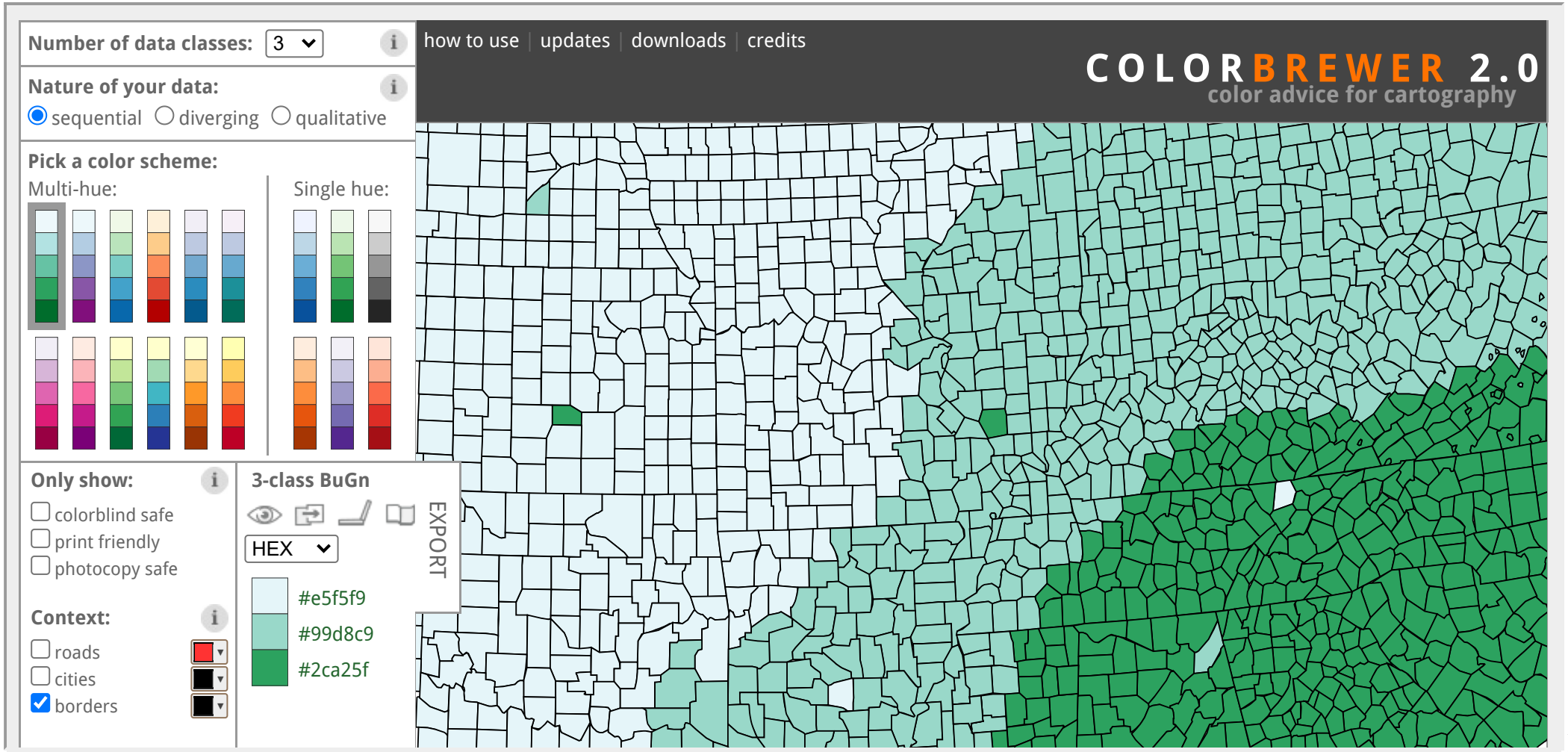
Select meaningful colors

No color | ColorBrewer website | Manual colors | RColorBrewer package | Palette



Select meaningful colors

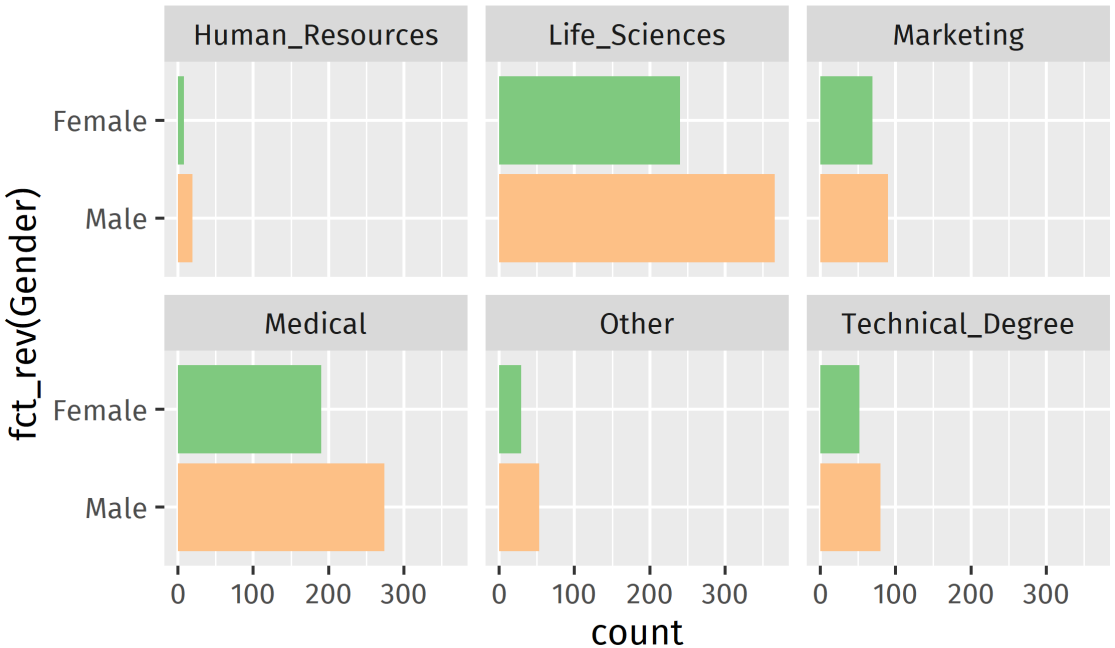
No color | **ColorBrewer website** | Manual colors | RColorBrewer package | Palette



Select meaningful colors

No color | ColorBrewer website | **Manual colors** | RColorBrewer package | Palette

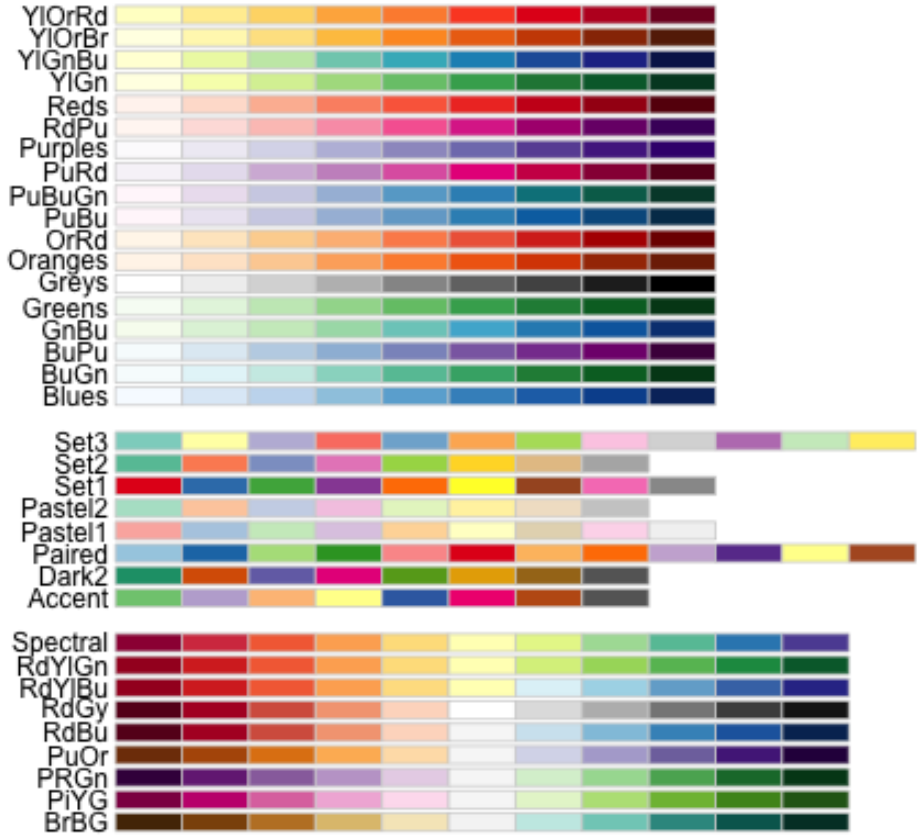
```
ggplot(attrition, aes(y = fct_rev(Gender), fill = Gender)) +
  geom_bar() +
  facet_wrap(~ EducationField) +
  scale_fill_manual(values = c("Female" = "#7fc97f", "Male" = "#fdc086")) +
  guides(fill = FALSE)
```



Select meaningful colors

No color | ColorBrewer website | Manual colors | **RColorBrewer package** | Palette

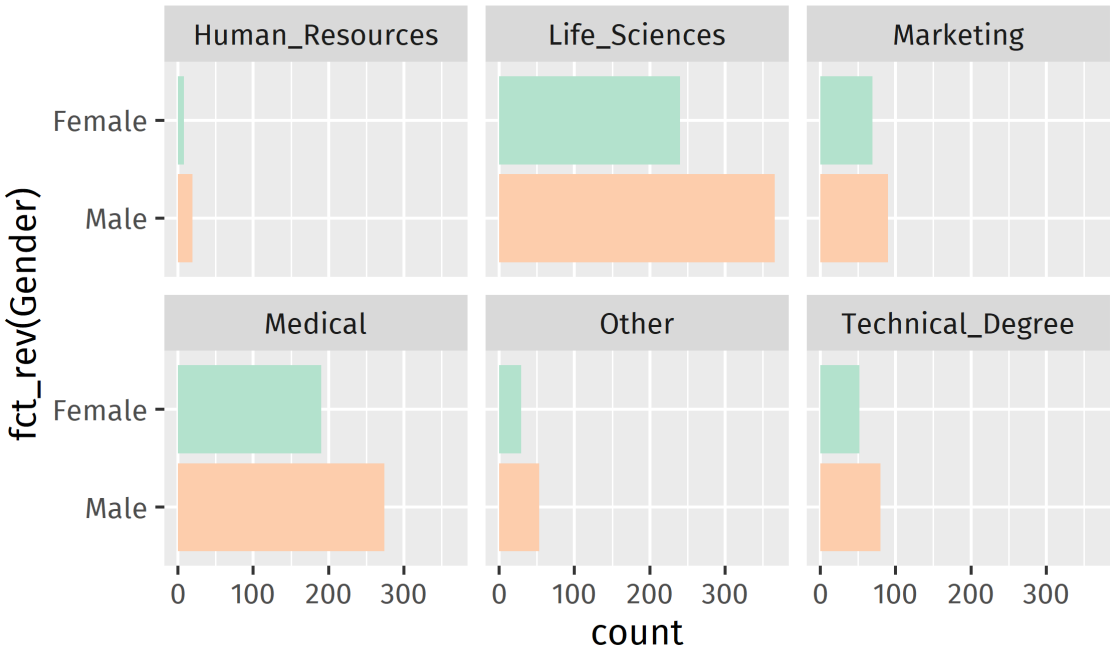
```
RColorBrewer::display.brewer.all()
```



Select meaningful colors

No color | ColorBrewer website | Manual colors | RColorBrewer package | **Palette**

```
ggplot(attrition, aes(y = fct_rev(Gender), fill = Gender)) +
  geom_bar() +
  facet_wrap(~ EducationField) +
  scale_fill_brewer(palette = "Pastel2") +
  guides(fill = FALSE)
```



Session info

```
##  setting  value
##  version  R version 4.0.4 (2021-02-15)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.1252
##  ctype    English_United States.1252
##  tz       Europe/Berlin
##  date     2021-03-29
```

package	version	date	source
dplyr	1.0.5	2021-03-05	CRAN (R 4.0.4)
forcats	0.5.1	2021-01-27	CRAN (R 4.0.3)
gapminder	0.3.0	2017-10-31	CRAN (R 4.0.3)
ggplot2	3.3.3	2020-12-30	CRAN (R 4.0.3)
purrr	0.3.4	2020-04-17	CRAN (R 4.0.2)

package	version	date	source
readr	1.4.0	2020-10-05	CRAN (R 4.0.3)
stringr	1.4.0	2019-02-10	CRAN (R 4.0.2)
tibble	3.1.0	2021-02-25	CRAN (R 4.0.3)
tidyr	1.1.3	2021-03-03	CRAN (R 4.0.4)
tidyverse	1.3.0	2019-11-21	CRAN (R 4.0.2)


Thank you! Questions?
Photo courtesy of Stefan Berger