

# *Graphic Design with ggplot2*

## **Concepts of the {ggplot2} Package Pt. 2:** Facets, Scales, and Coordinate Systems

Cédric Scherer // rstudio::conf // July 2022

# Setup

```
1 library(tidyverse)
2
3 bikes <- readr::read_csv(
4   here::here("data", "london-bikes-custom.csv"),
5   col_types = "Dcffffillllddddc"
6 )
7
8 bikes$season <- forcats::fct_inorder(bikes$season)
9
10 theme_set(theme_light(base_size = 14, base_family = "Roboto Condensed"))
11
12 theme_update(
13   panel.grid.minor = element_blank(),
14   plot.title = element_text(face = "bold"),
15   legend.position = "top",
16   plot.title.position = "plot"
17 )
18
19 invisible(Sys.setlocale("LC_TIME", "C"))
```

# Facets

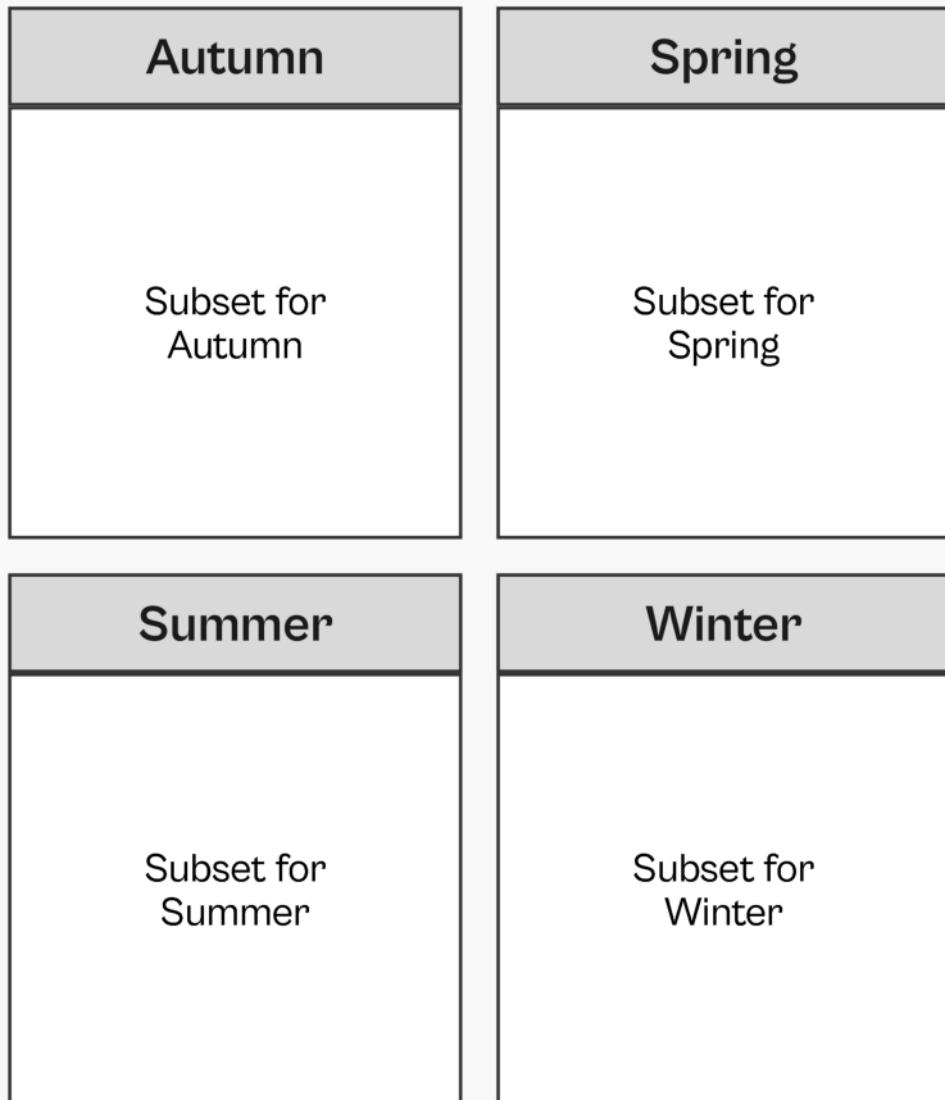
# Facets

= split variables to multiple panels

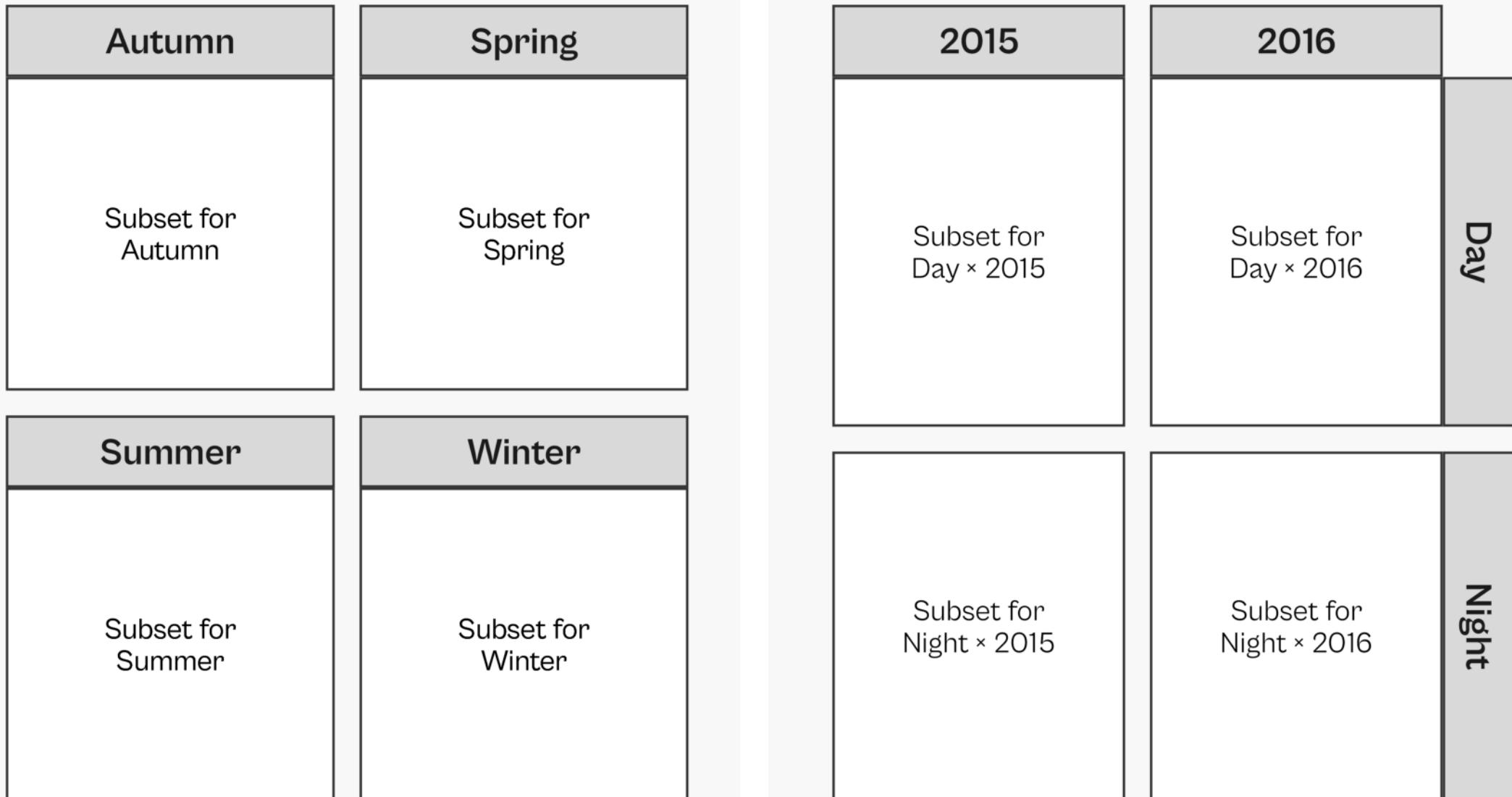
Facets are also known as:

- small multiples
- trellis graphs
- lattice plots
- conditioning

# **facet\_wrap()**

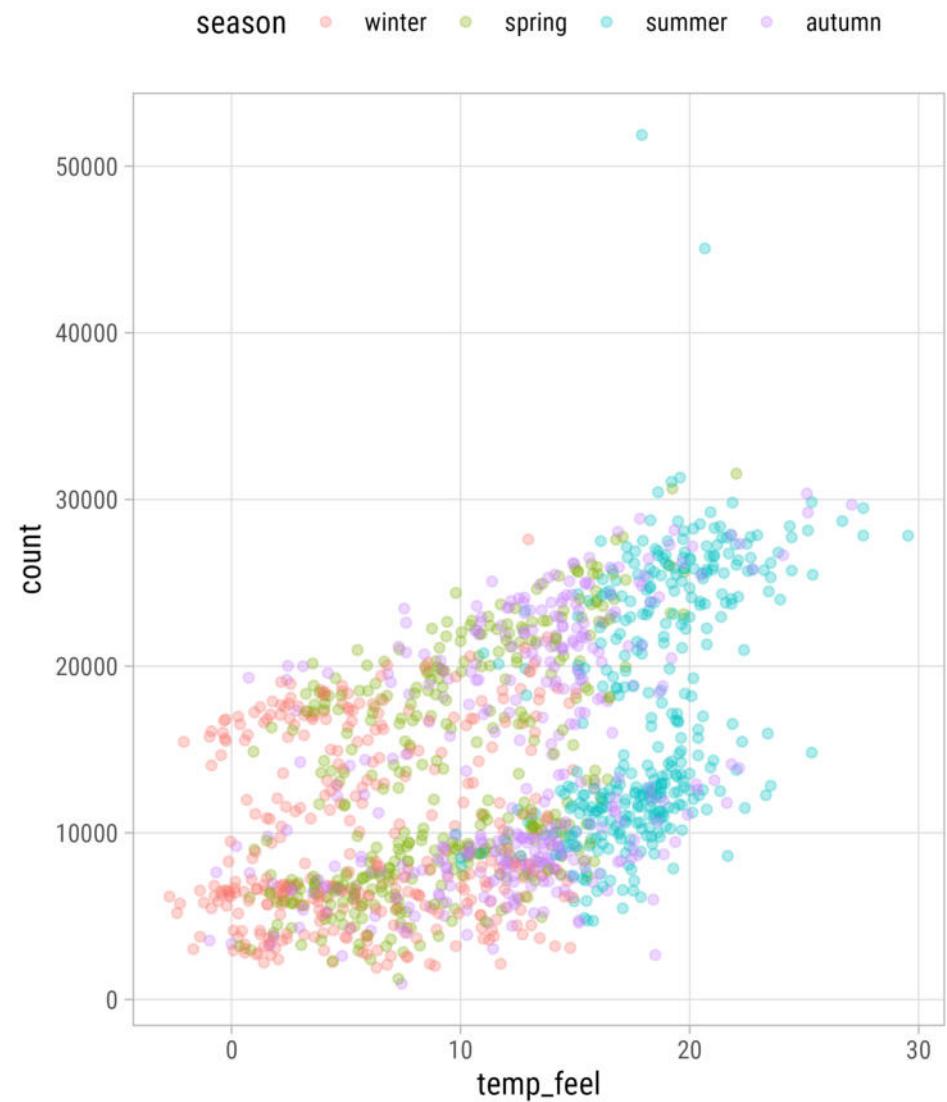


# **facet\_grid()**



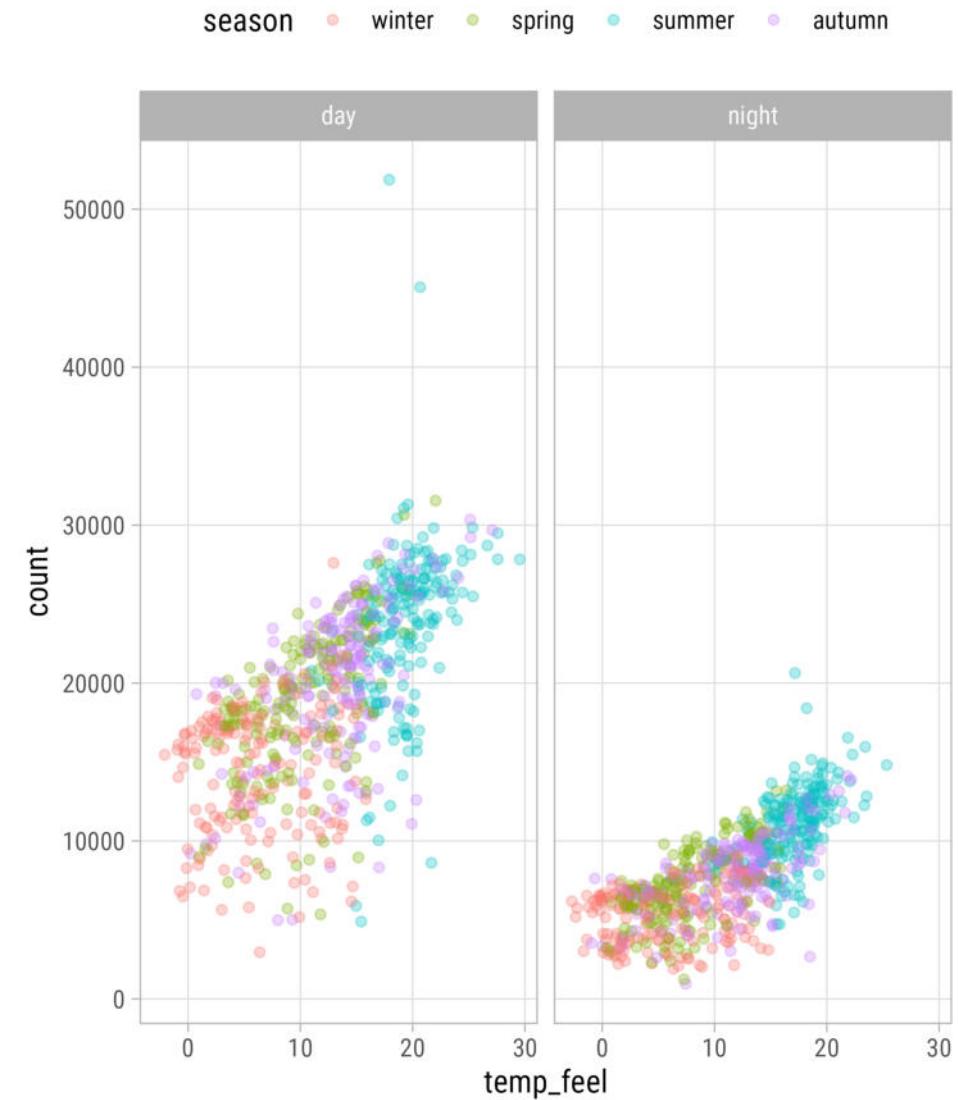
# Setup

```
1 g <-  
2   ggplot(  
3     bikes,  
4     aes(x = temp_feel, y = count,  
5           color = season))  
6   ) +  
7   geom_point(  
8     alpha = .3,  
9     guide = "none"  
10 )  
11  
12 g
```



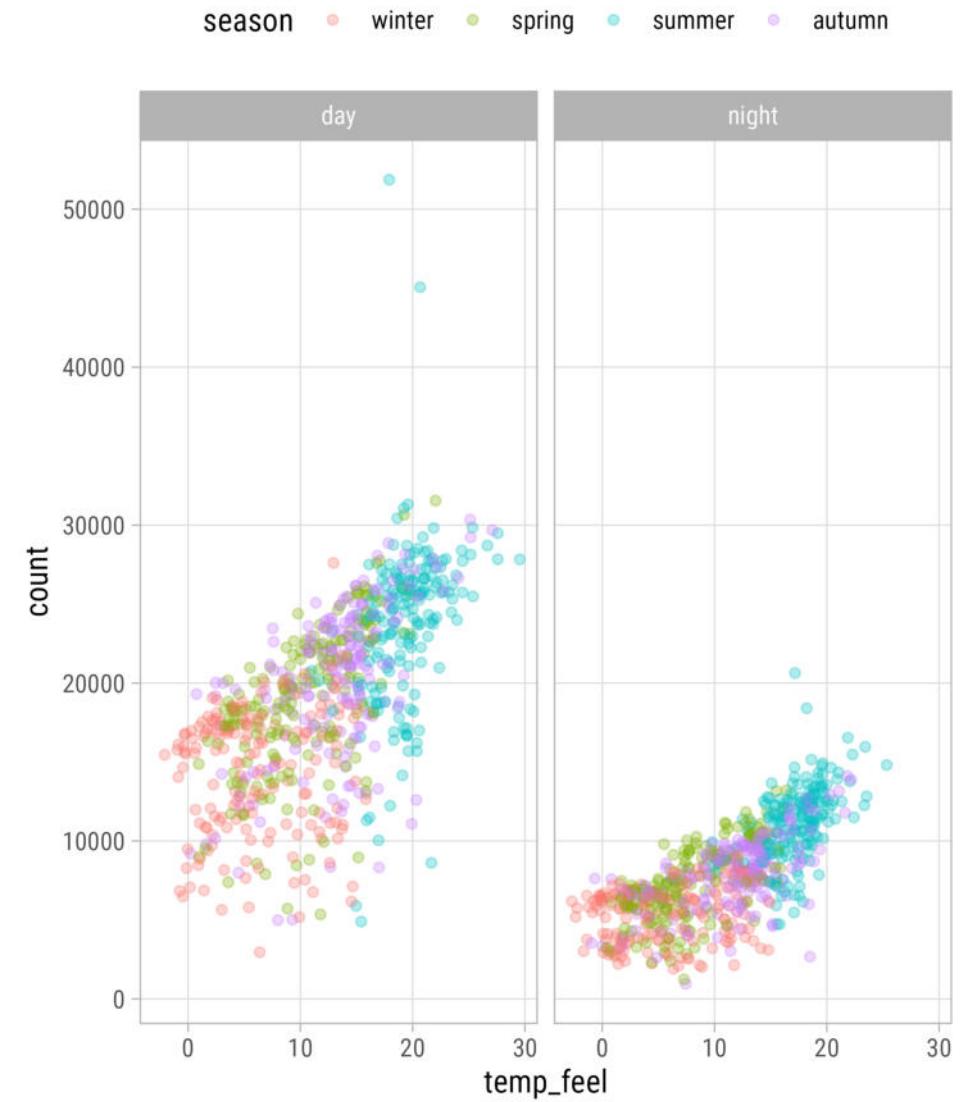
# Wrapped Facets

```
1 g +
2   facet_wrap(
3     vars(day_night)
4   )
```



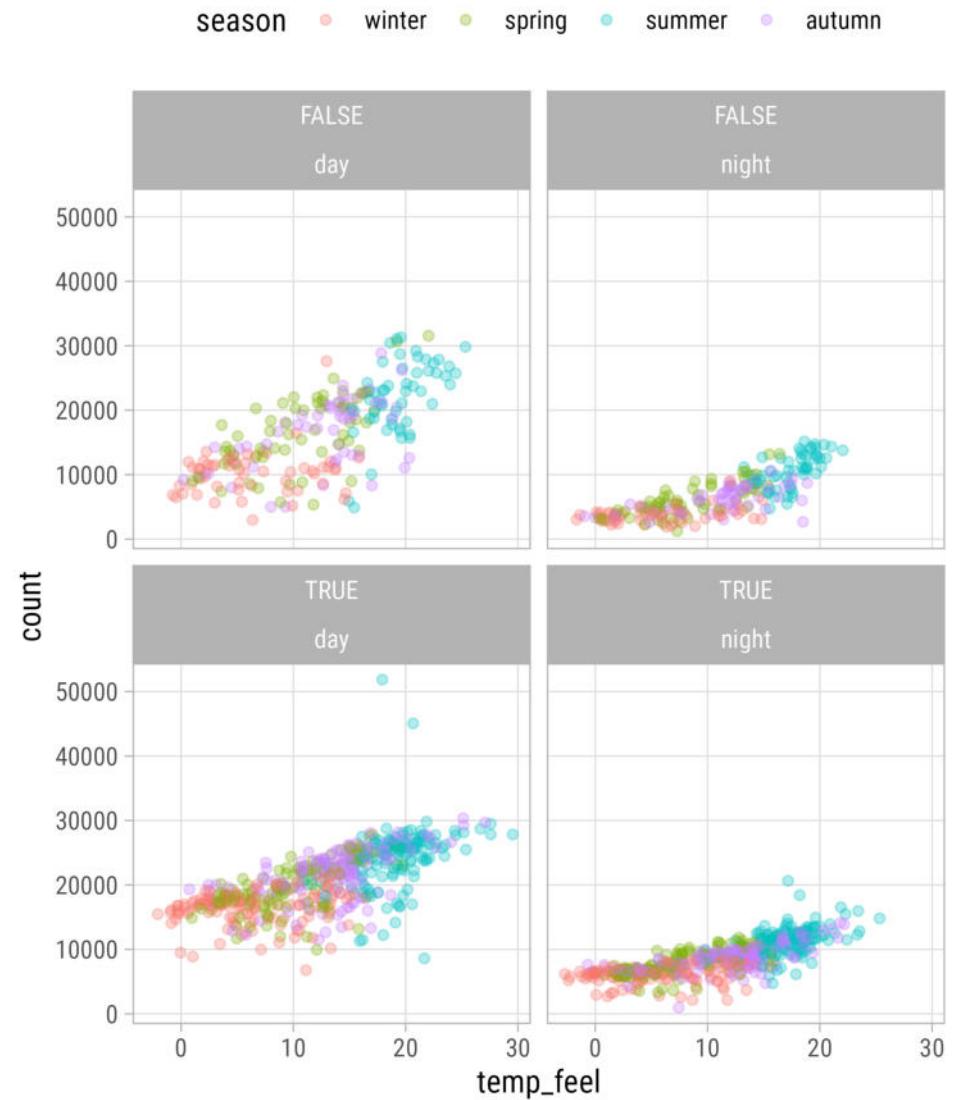
# Wrapped Facets

```
1 g +
2   facet_wrap(
3     ~ day_night
4   )
```



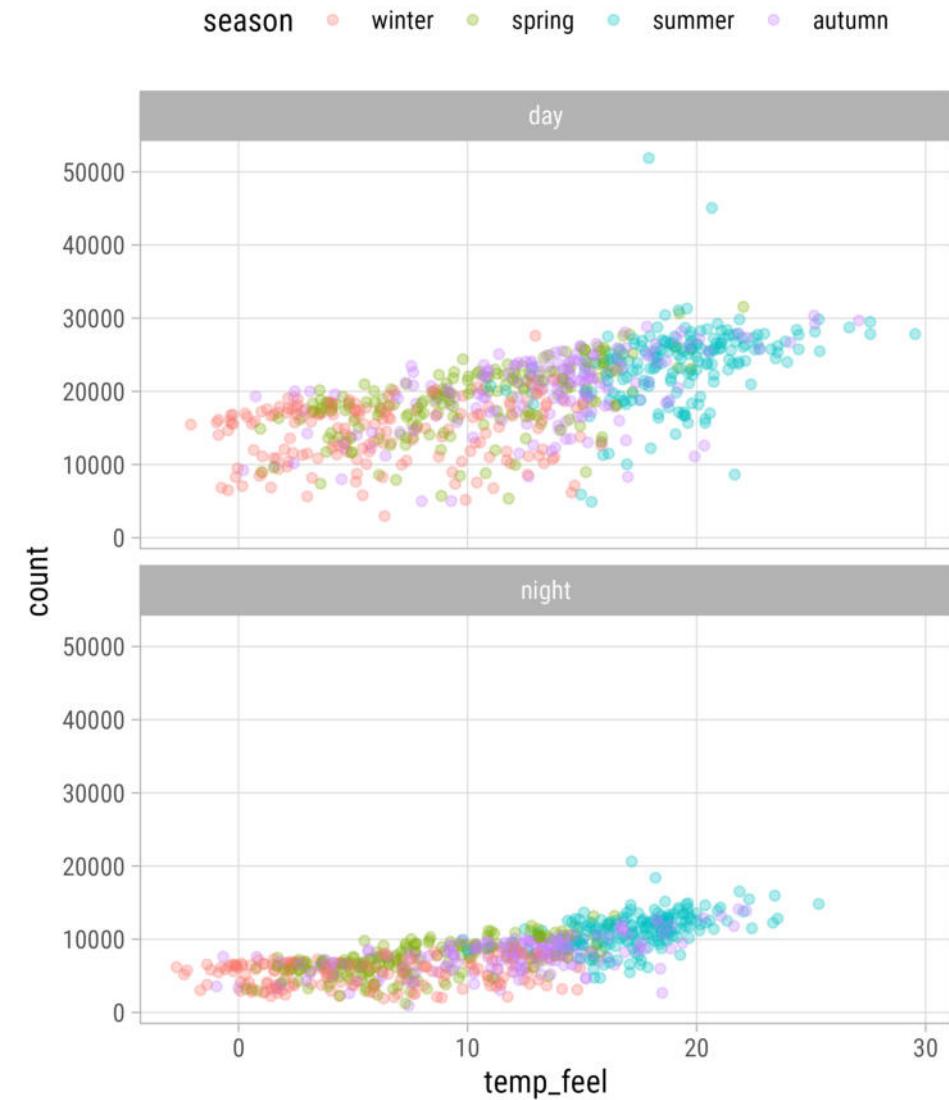
# Facet Multiple Variables

```
1 g +
2   facet_wrap(
3     ~ is_workday + day_night
4   )
```



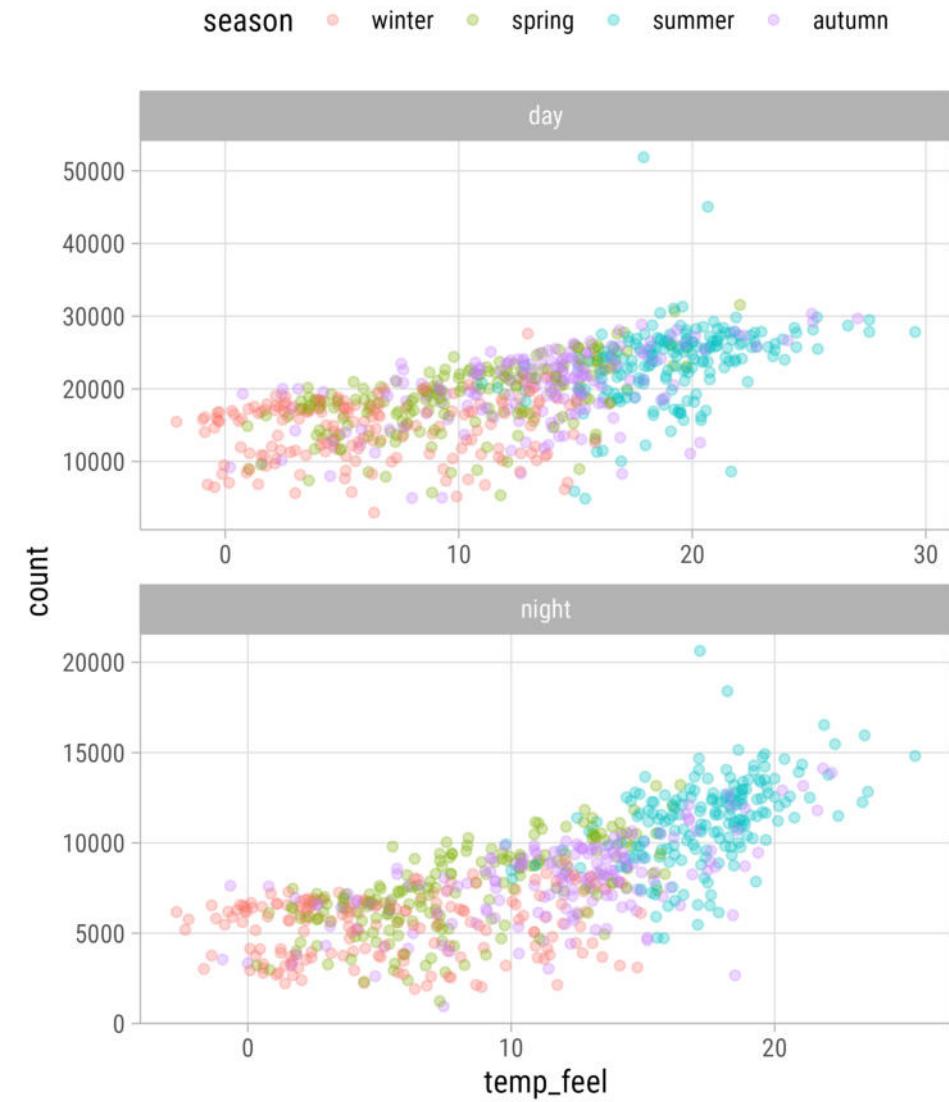
# Facet Options: Cols + Rows

```
1 g +
2   facet_wrap(
3     ~ day_night,
4     ncol = 1
5   )
```



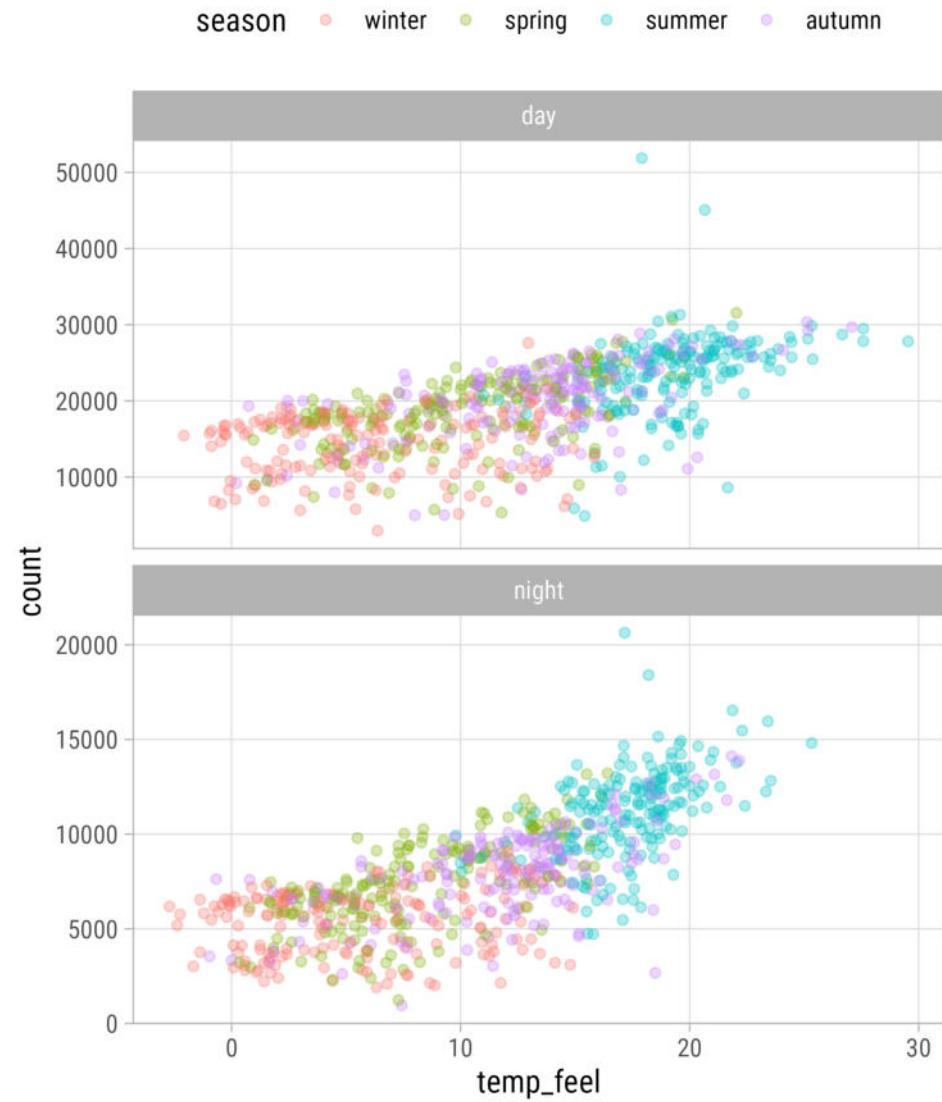
# Facet Options: Free Scaling

```
1 g +  
2   facet_wrap(  
3     ~ day_night,  
4     ncol = 1,  
5     scales = "free"  
6   )
```



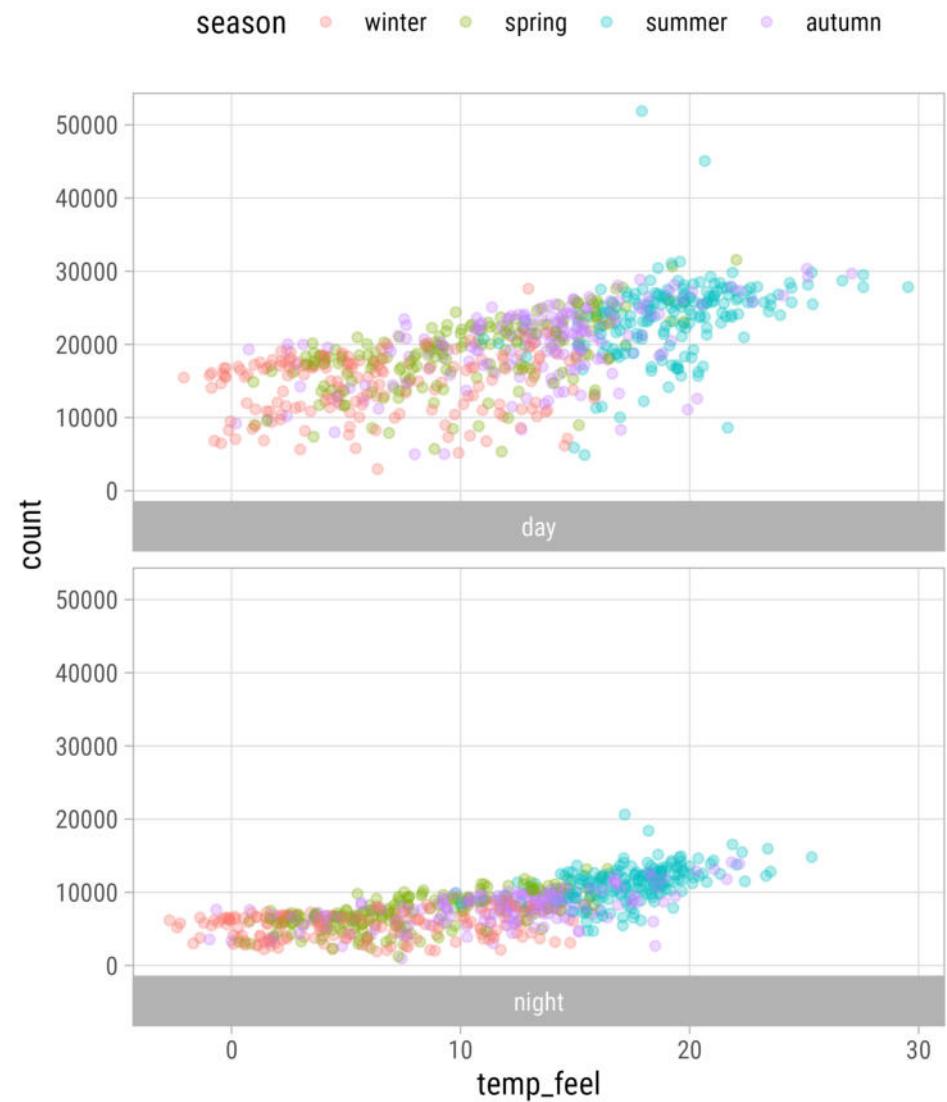
# Facet Options: Free Scaling

```
1 g +
2   facet_wrap(
3     ~ day_night,
4     ncol = 1,
5     scales = "free_y"
6   )
```



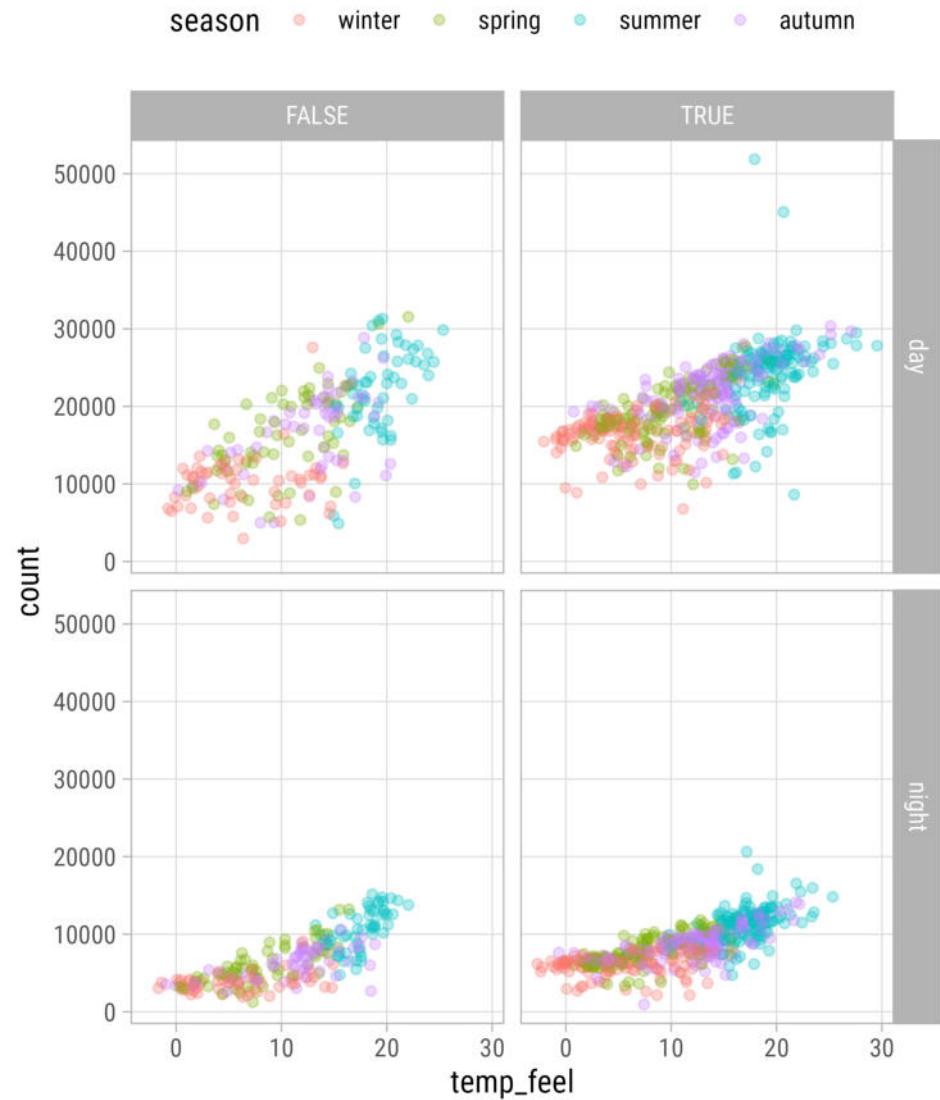
# Facet Options: Switch Labels

```
1 g +
2   facet_wrap(
3     ~ day_night,
4     ncol = 1,
5     switch = "x"
6   )
```



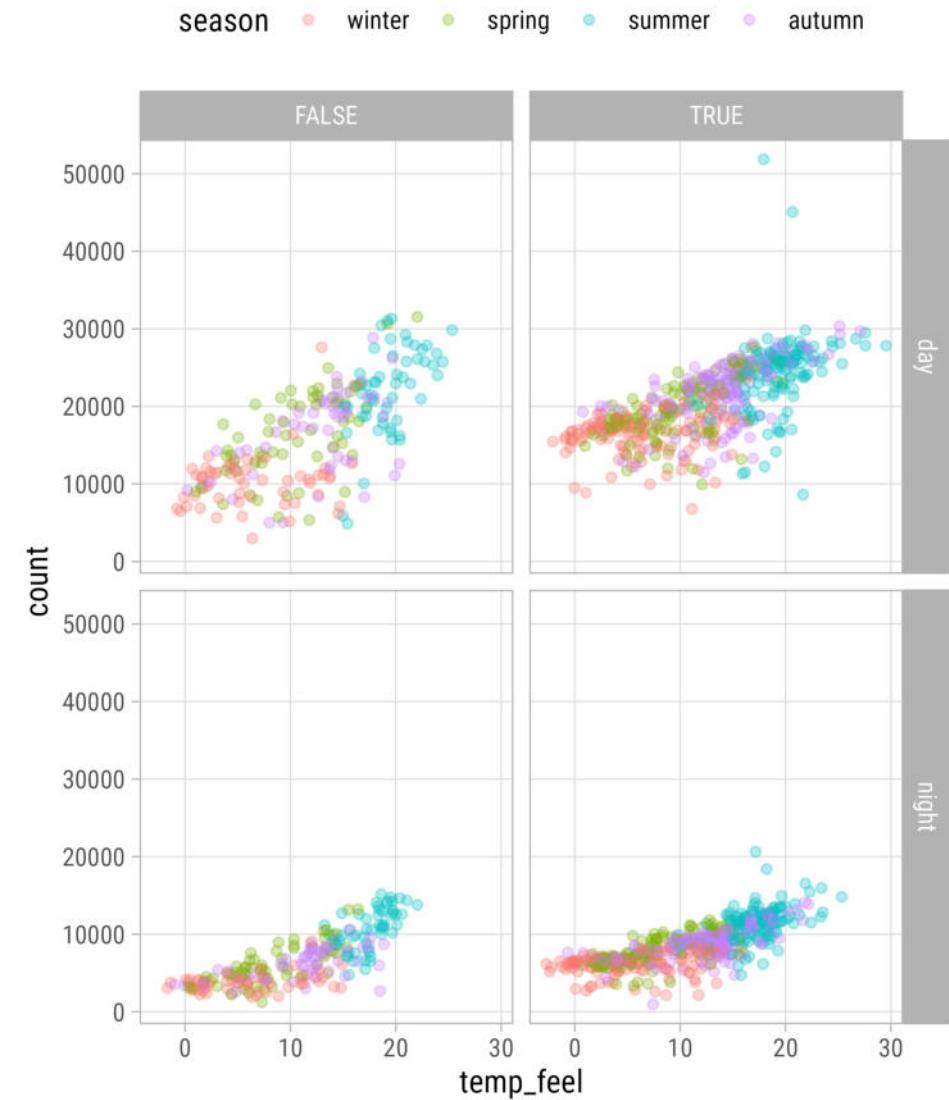
# Gridded Facets

```
1 g +
2   facet_grid(
3     rows = vars(day_night),
4     cols = vars(is_workday)
5   )
```



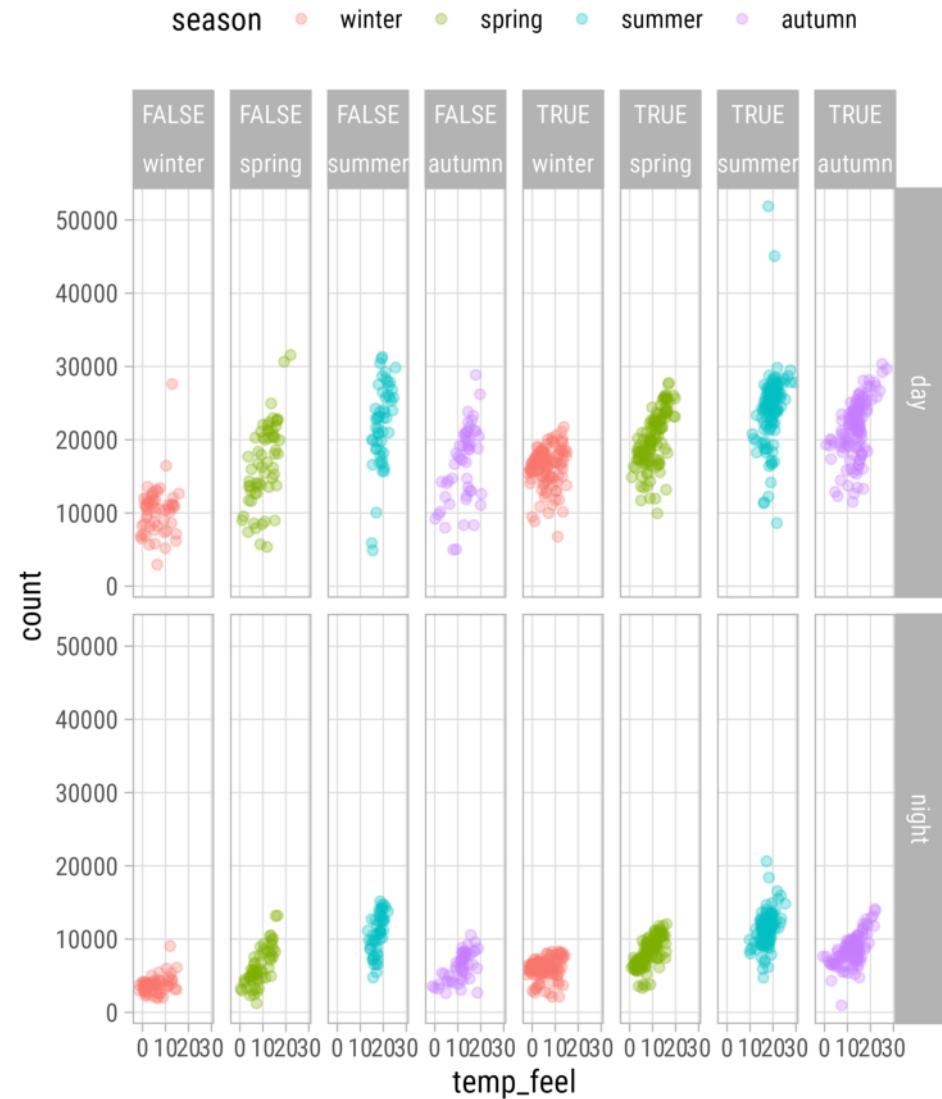
# Gridded Facets

```
1 g +
2   facet_grid(
3     day_night ~ is_workday
4   )
```



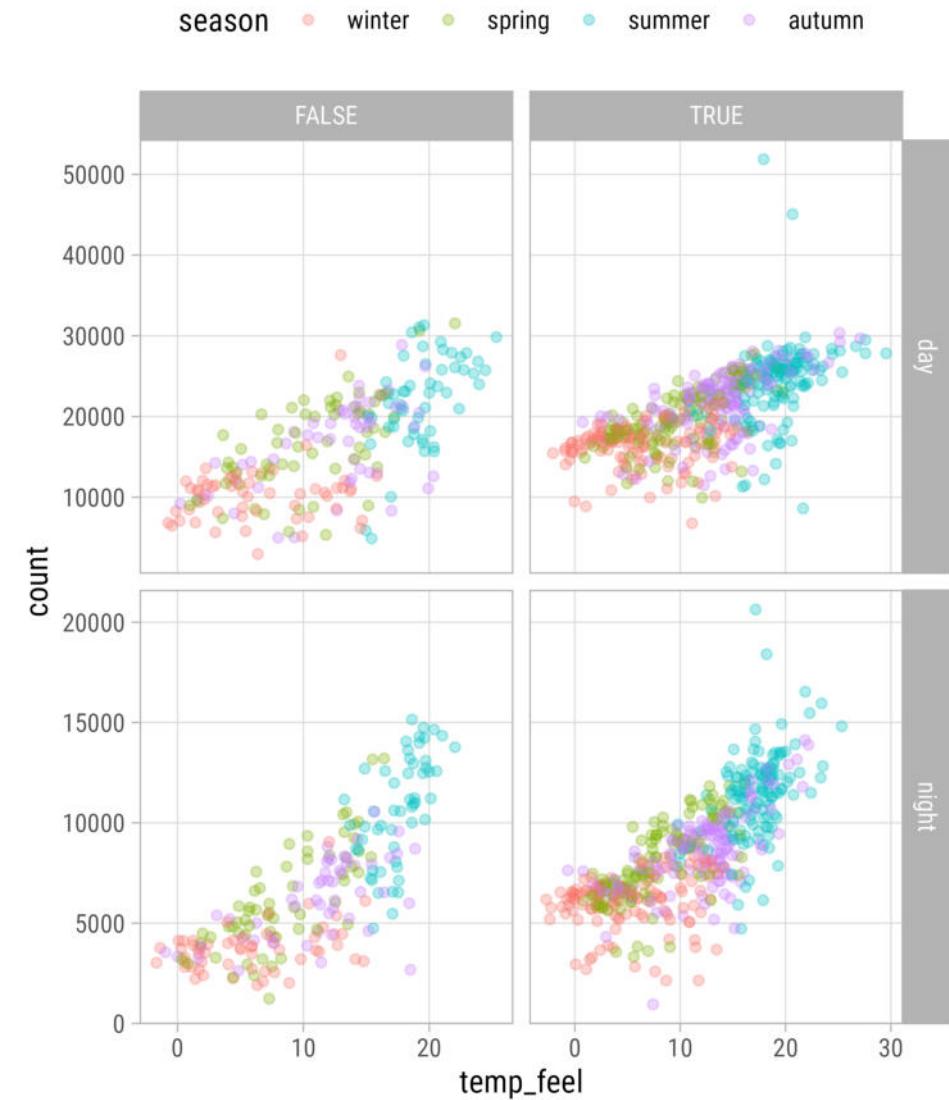
# Facet Multiple Variables

```
1 g +
2   facet_grid(
3     day_night ~ is_workday + season
4   )
```



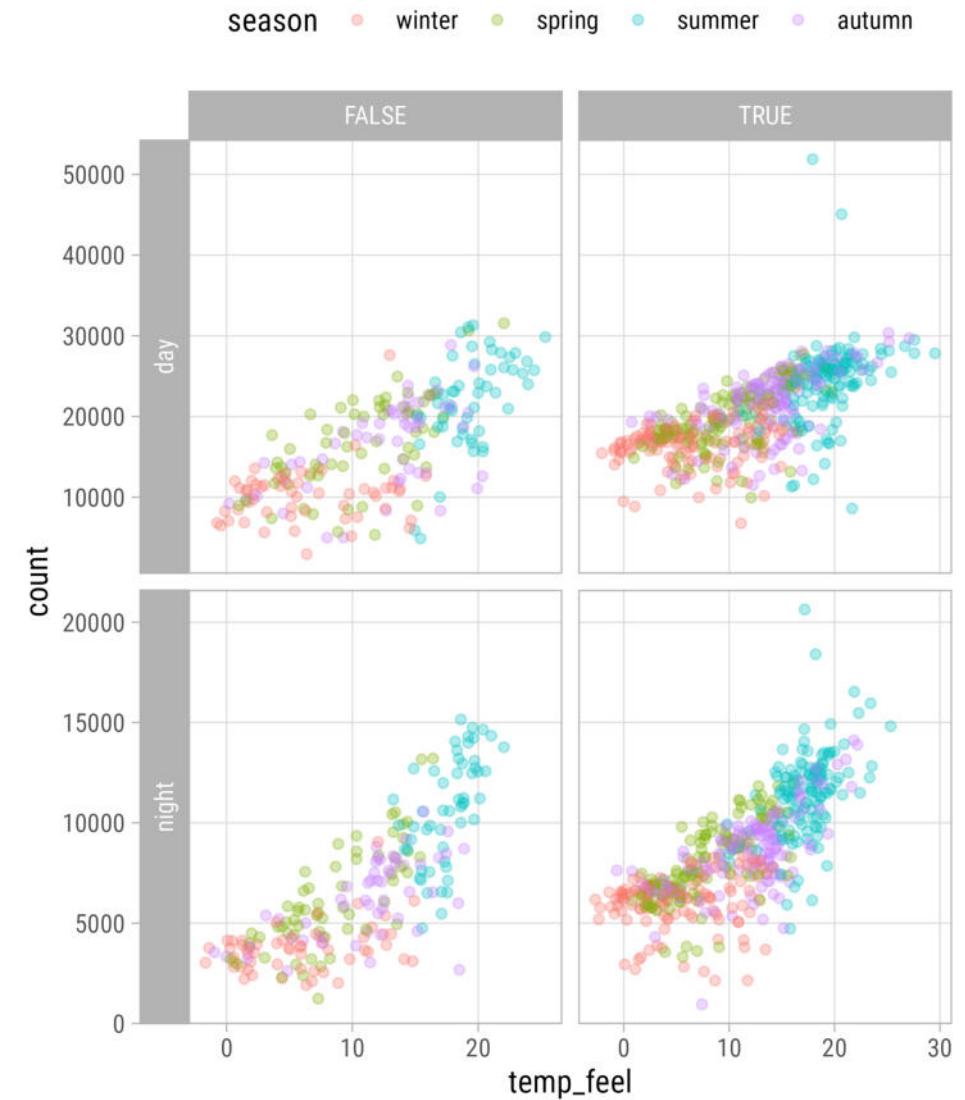
# Facet Options: Free Scaling

```
1 g +
2   facet_grid(
3     day_night ~ is_workday,
4     scales = "free"
5   )
```



# Facet Options: Switch Labels

```
1 g +
2   facet_grid(
3     day_night ~ is_workday,
4     scales = "free",
5     switch = "y"
6   )
```



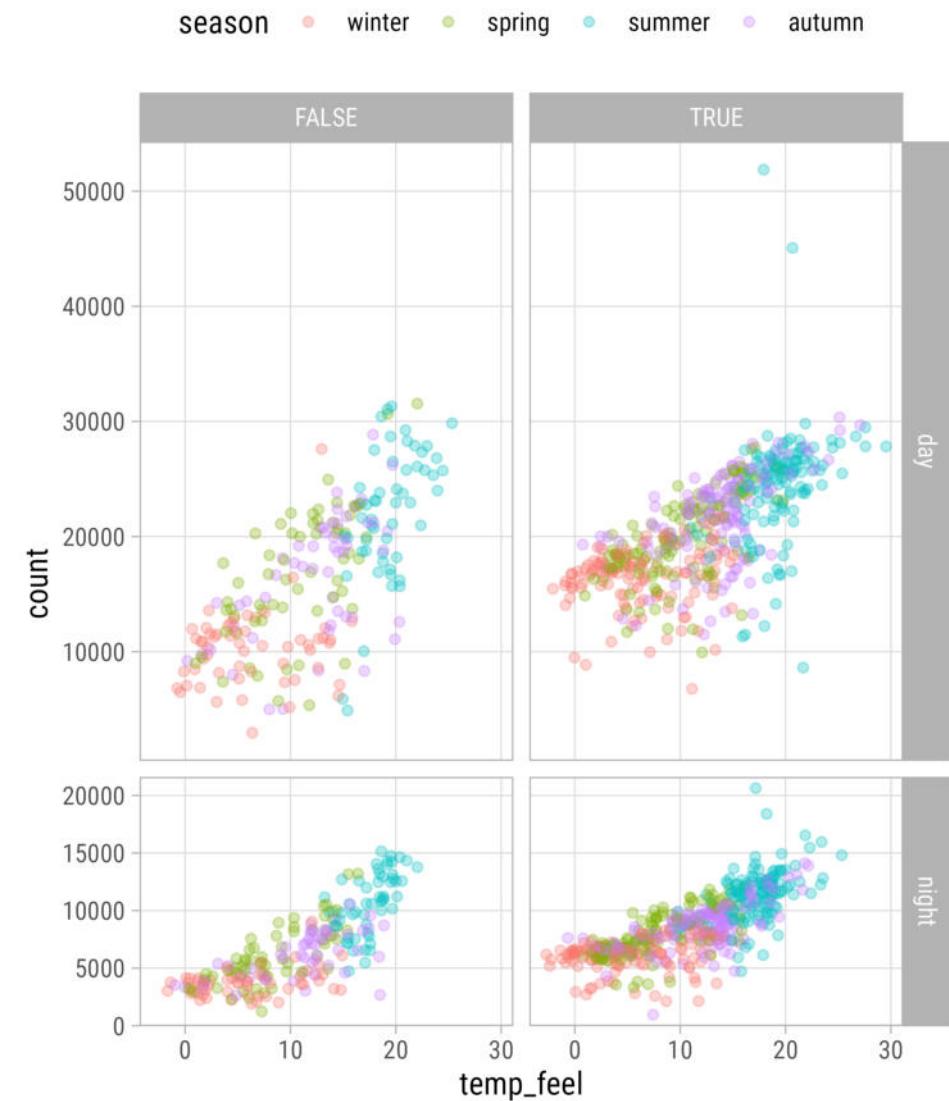
# Facet Options: Proportional Spacing

```
1 g +
2   facet_grid(
3     day_night ~ is_workday,
4     scales = "free",
5     space = "free"
6   )
```



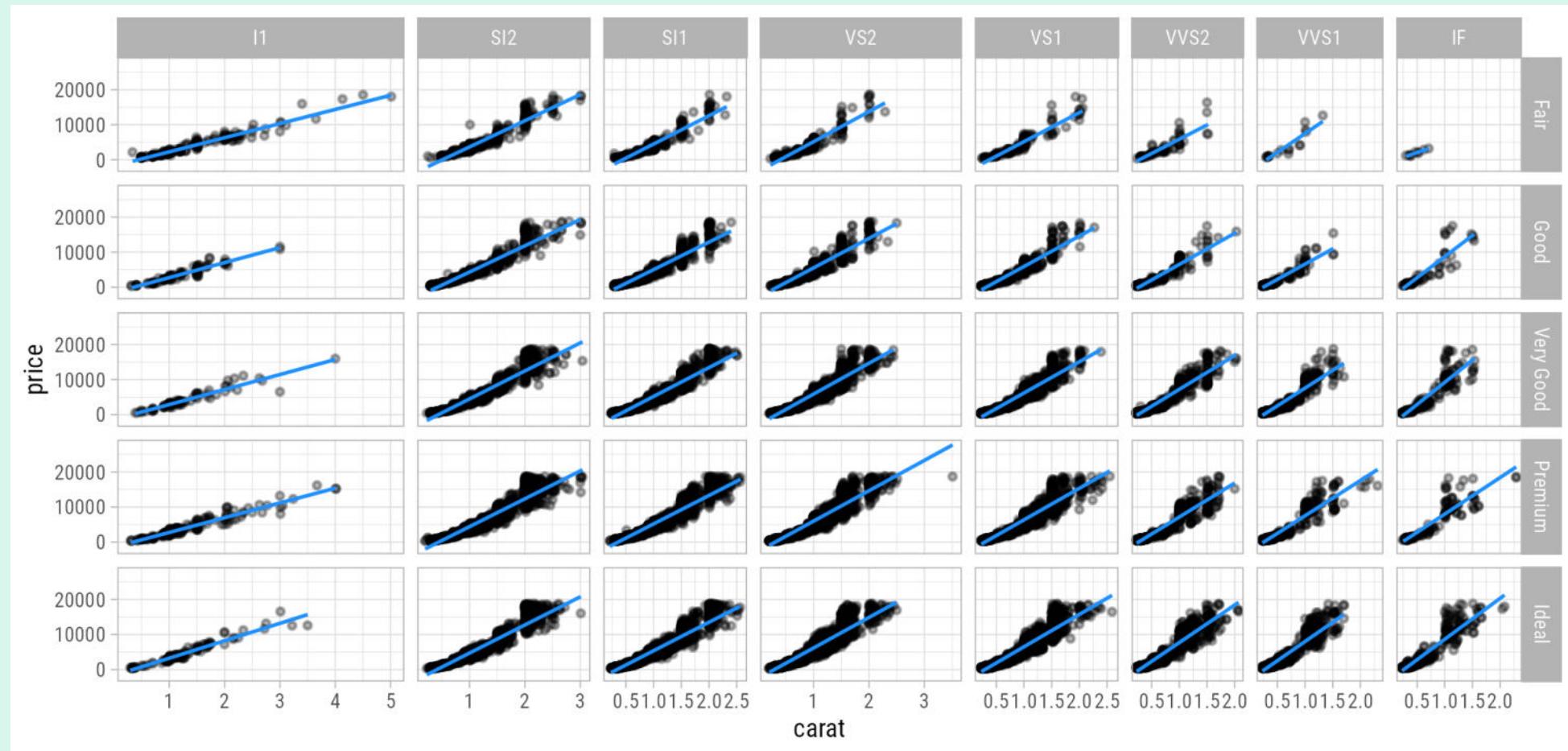
# Facet Options: Proportional Spacing

```
1 g +
2   facet_grid(
3     day_night ~ is_workday,
4     scales = "free_y",
5     space = "free_y"
6   )
```



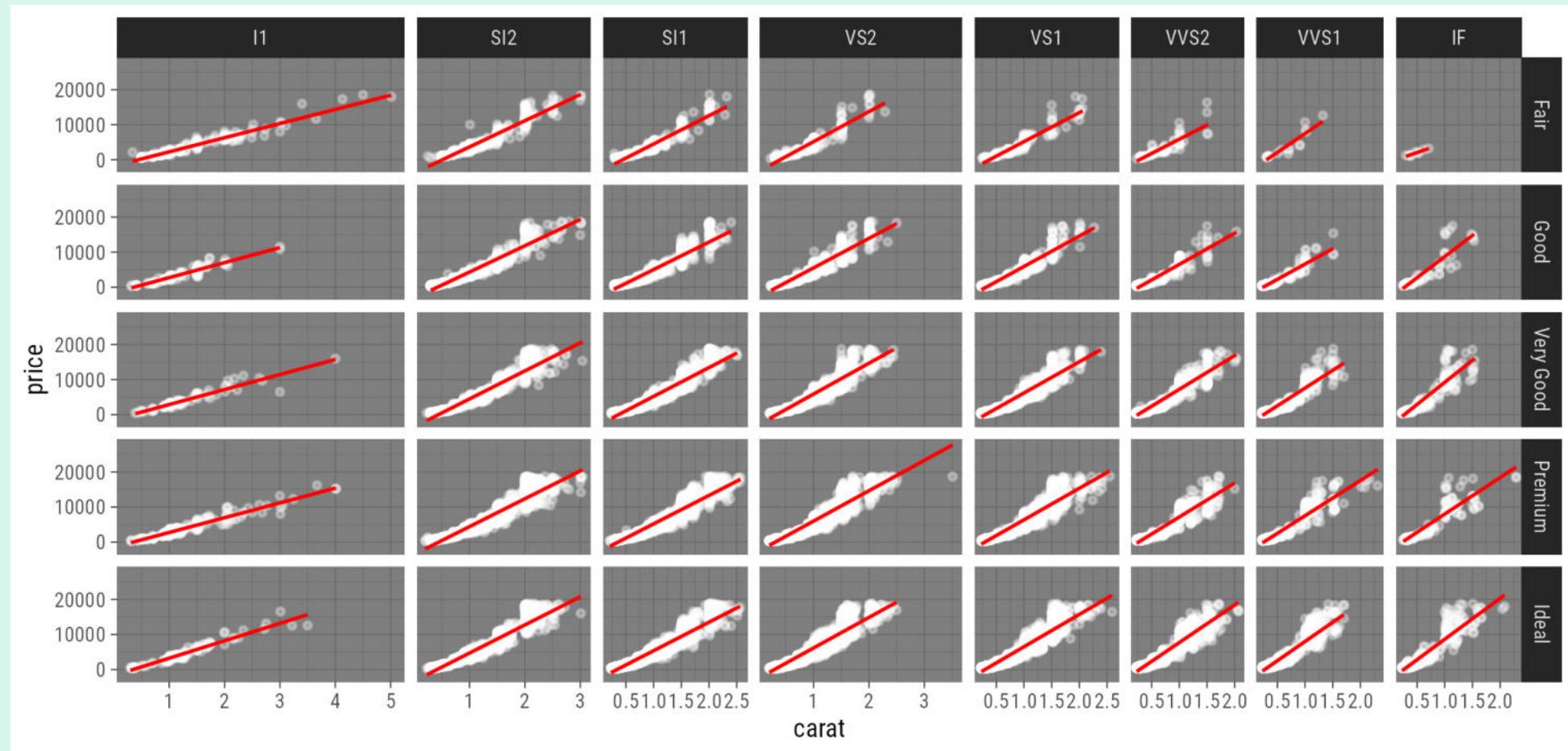
# Your Turn!

Create the following facet from the `diamonds` data.



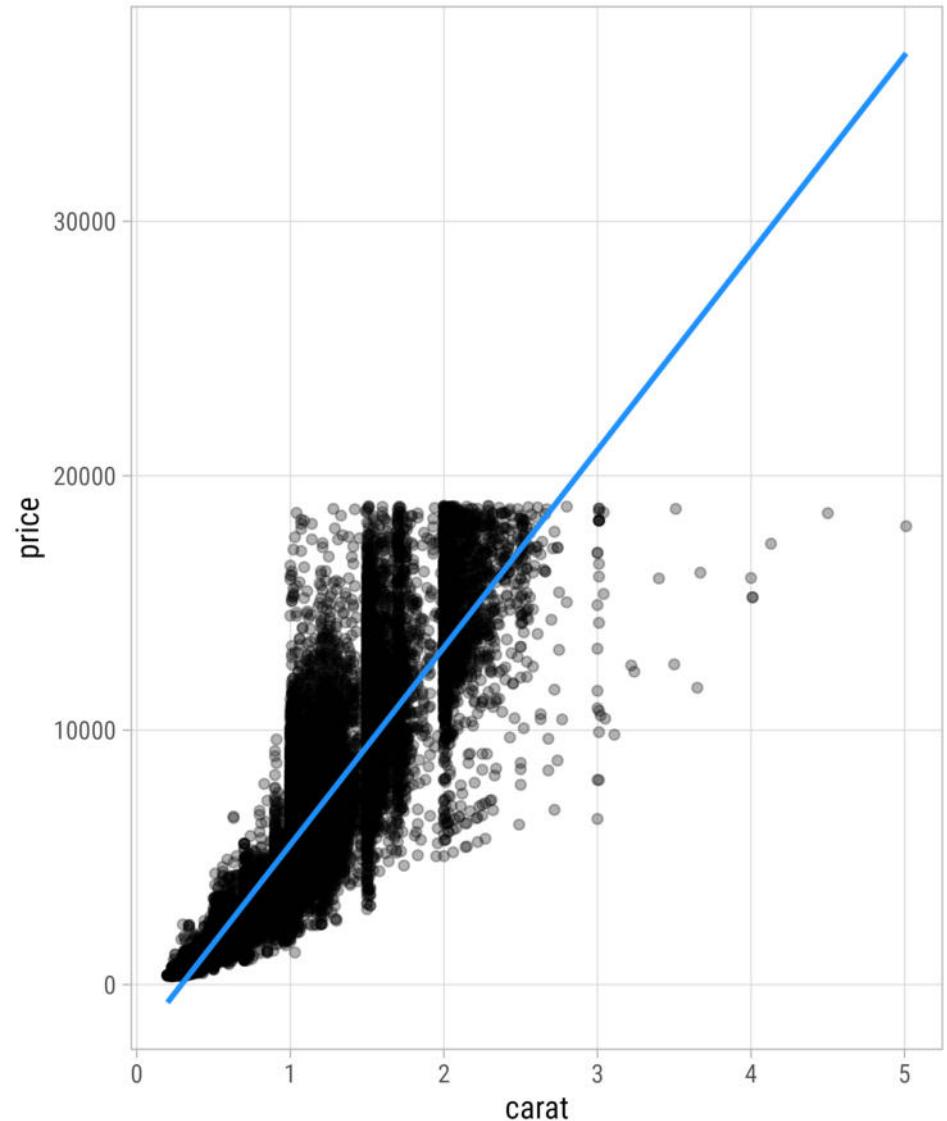
# Your Turn!

Bonus: Create this bloody-dark version.



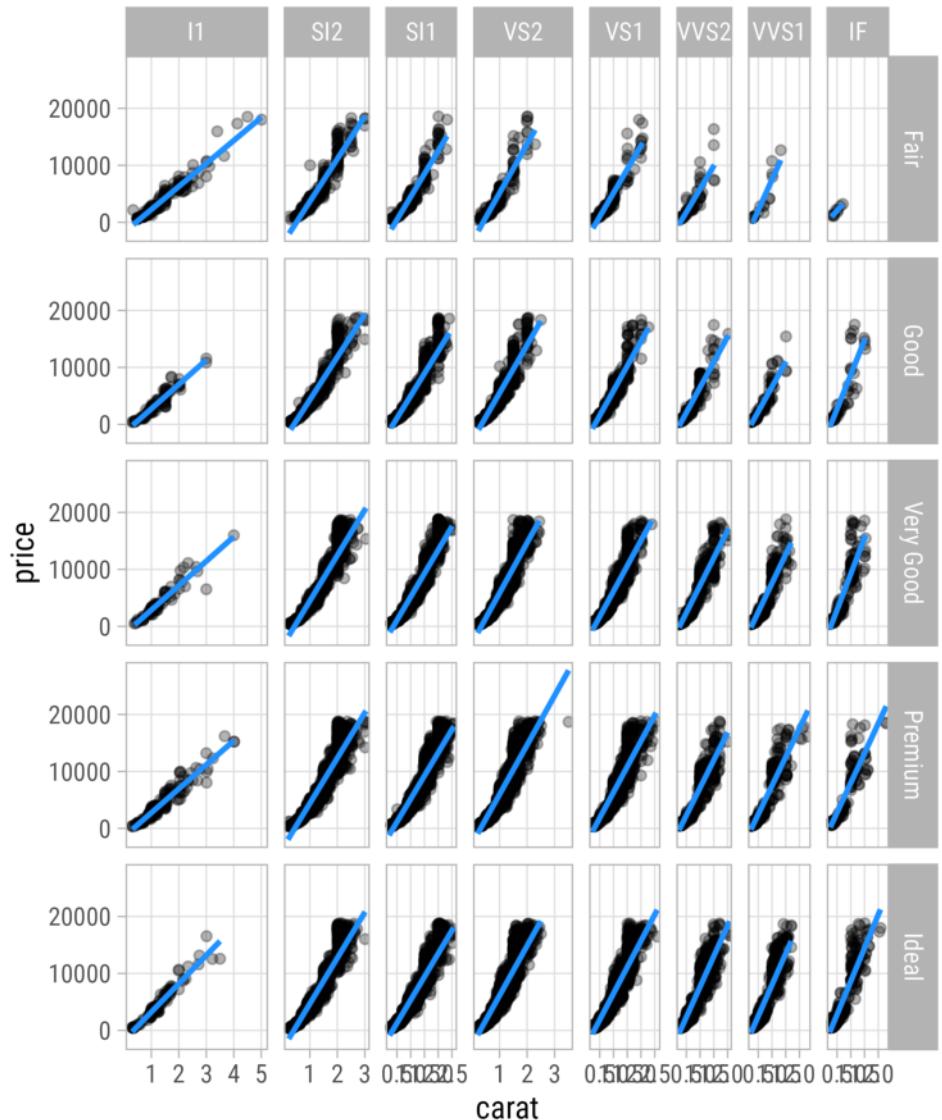
# Diamonds Facet

```
1 ggplot(  
2   diamonds,  
3   aes(x = carat, y = price)  
4 ) +  
5   geom_point(  
6   alpha = .3  
7 ) +  
8   geom_smooth(  
9   method = "lm",  
10  se = FALSE,  
11  color = "dodgerblue"  
12 )
```



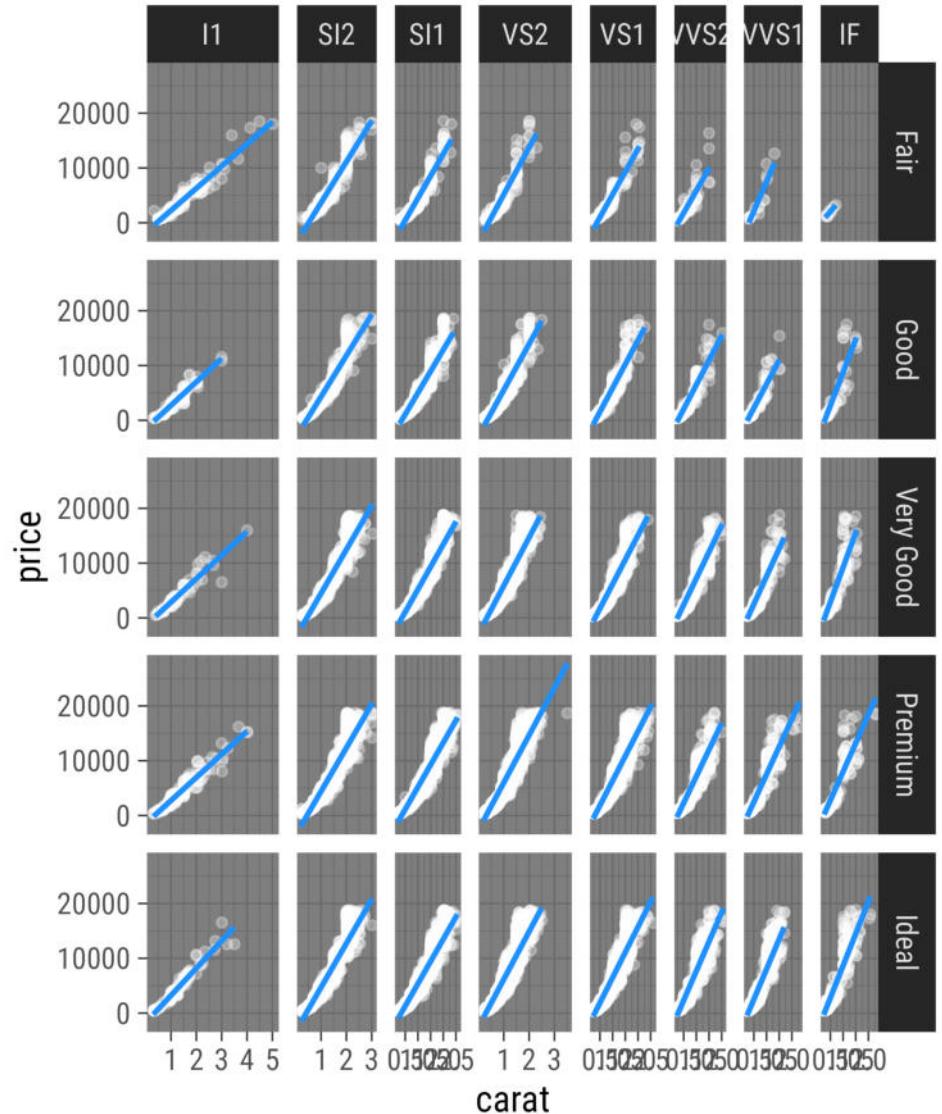
# Diamonds Facet

```
1 ggplot(  
2   diamonds,  
3   aes(x = carat, y = price)  
4 ) +  
5   geom_point(  
6     alpha = .3  
7 ) +  
8   geom_smooth(  
9     method = "lm",  
10    se = FALSE,  
11    color = "dodgerblue"  
12 ) +  
13   facet_grid(  
14     cut ~ clarity,  
15     space = "free_x",  
16     scales = "free_x"  
17 )
```



# Diamonds Facet (Dark Theme Bonus)

```
1 ggplot(  
2   diamonds,  
3   aes(x = carat, y = price)  
4 ) +  
5   geom_point(  
6     alpha = .3,  
7     color = "white"  
8 ) +  
9   geom_smooth(  
10    method = "lm",  
11    se = FALSE,  
12    color = "dodgerblue"  
13 ) +  
14   facet_grid(  
15     cut ~ clarity,  
16     space = "free_x",  
17     scales = "free_x"  
18 ) +  
19   theme_dark(  
20     base_size = 14,  
21     base_family = "Roboto Condensed"  
22 )
```



# Seales

# Scales

= translate between variable ranges and property ranges

- feels-like temperature  $\rightleftharpoons$  x
- reported bike shares  $\rightleftharpoons$  y
- season  $\rightleftharpoons$  color
- year  $\rightleftharpoons$  shape
- ...

# Scales

The `scale_*`() components control the properties of all the aesthetic dimensions mapped to the data.

Consequently, there are `scale_*`() functions for all aesthetics such as:

- **positions** via `scale_x_*`() and `scale_y_*`()
- **colors** via `scale_color_*`() and `scale_fill_*`()
- **sizes** via `scale_size_*`() and `scale_radius_*`()
- **shapes** via `scale_shape_*`() and `scale_linetype_*`()
- **transparency** via `scale_alpha_*`()

# Scales

The `scale_*`() components control the properties of all the aesthetic dimensions mapped to the data.

The extensions (\*) can be filled by e.g.:

- `continuous()`, `discrete()`, `reverse()`, `log10()`, `sqrt()`, `date()` for positions
- `continuous()`, `discrete()`, `manual()`, `gradient()`, `gradient2()`, `brewer()` for colors
- `continuous()`, `discrete()`, `manual()`, `ordinal()`, `area()`, `date()` for sizes
- `continuous()`, `discrete()`, `manual()`, `ordinal()` for shapes
- `continuous()`, `discrete()`, `manual()`, `ordinal()`, `date()` for transparency

# CONTINUOUS

measured data, can have  $\infty$  values within possible range.



I AM 3.1" TALL  
I WEIGH 34.16 grams

# DISCRETE

OBSERVATIONS CAN ONLY EXIST AT LIMITED VALUES, OFTEN COUNTS.



I HAVE 8 LEGS  
and  
4 SPOTS!

@allison\_horst

Illustration by Allison Horst

Cédric Scherer // rstudio::conf // July 2022

# Continuous vs. Discrete in {ggplot2}

Continuous:

quantitative or numerical data

- height
- weight
- age
- counts

Discrete:

qualitative or categorical data

- species
- sex
- study sites
- age group

# Continuous vs. Discrete in {ggplot2}

**Continuous:**

**quantitative or numerical data**

- height (continuous)
- weight (continuous)
- age (continuous or discrete)
- counts (discrete)

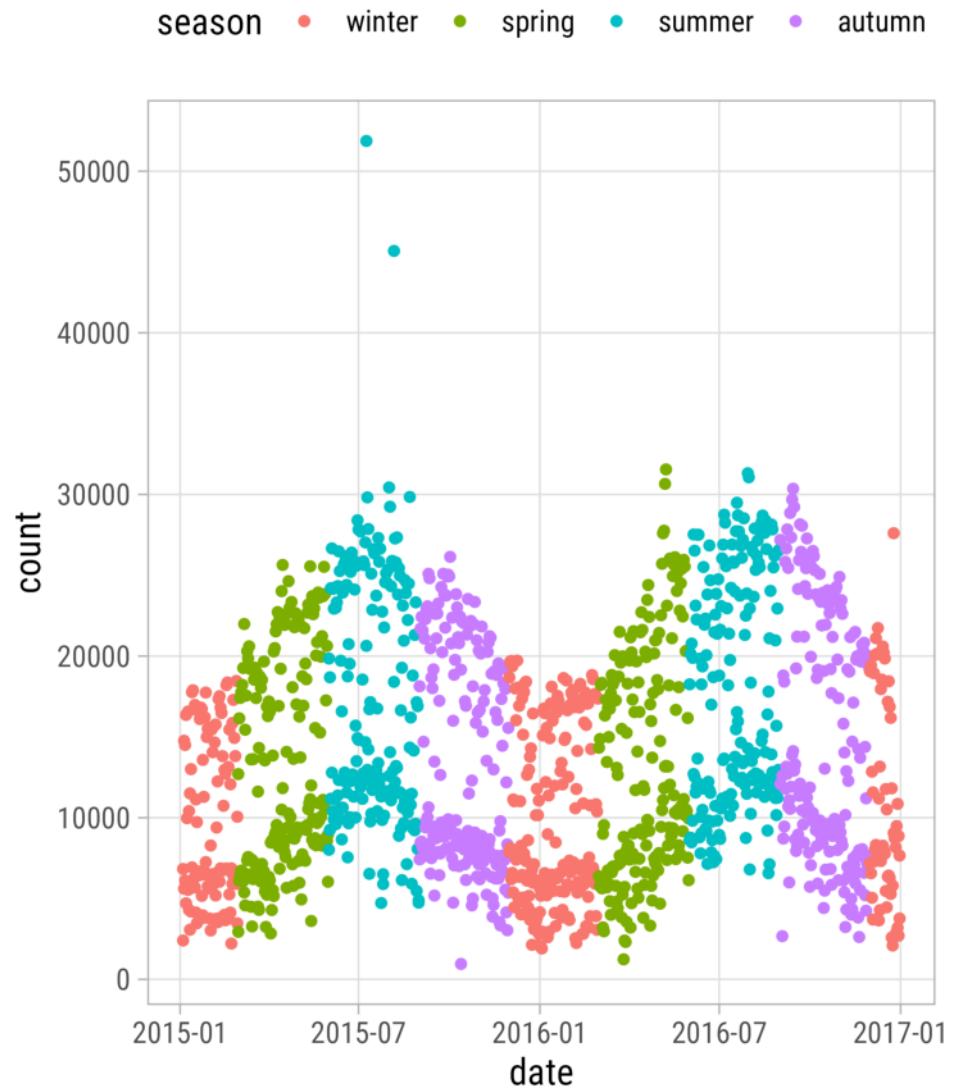
**Discrete:**

**qualitative or categorical data**

- species (nominal)
- sex (nominal)
- study site (nominal or ordinal)
- age group (ordinal)

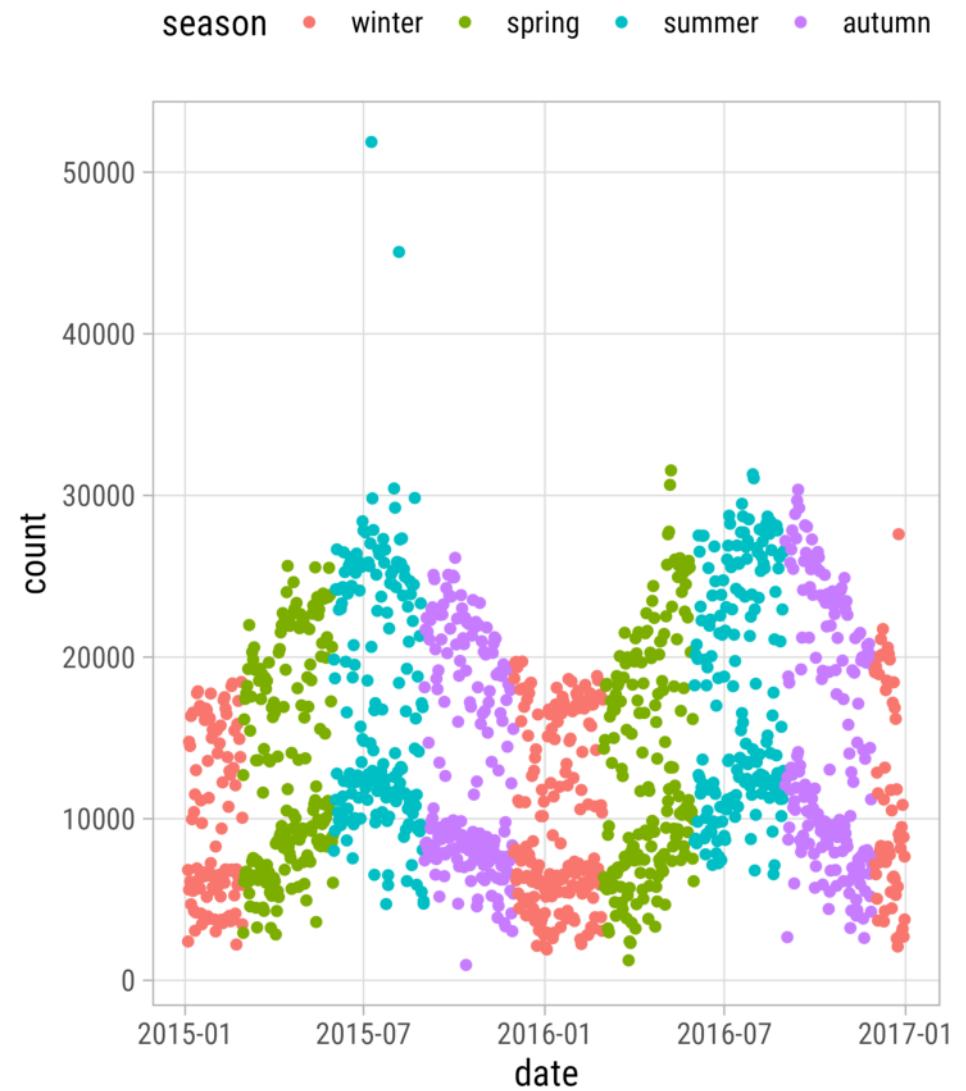
# Aesthetics + Scales

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point()
```



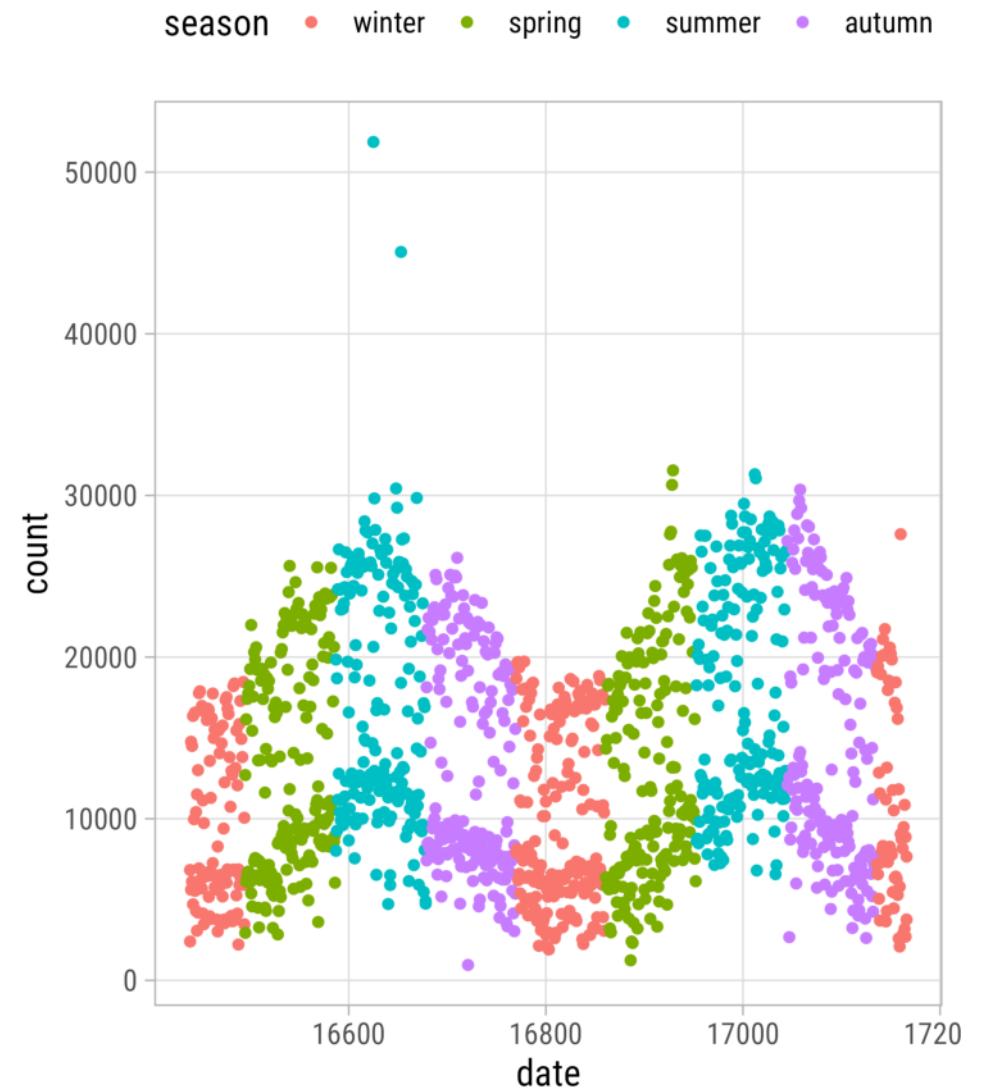
# Aesthetics + Scales

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_date() +  
8 scale_y_continuous() +  
9 scale_color_discrete()
```



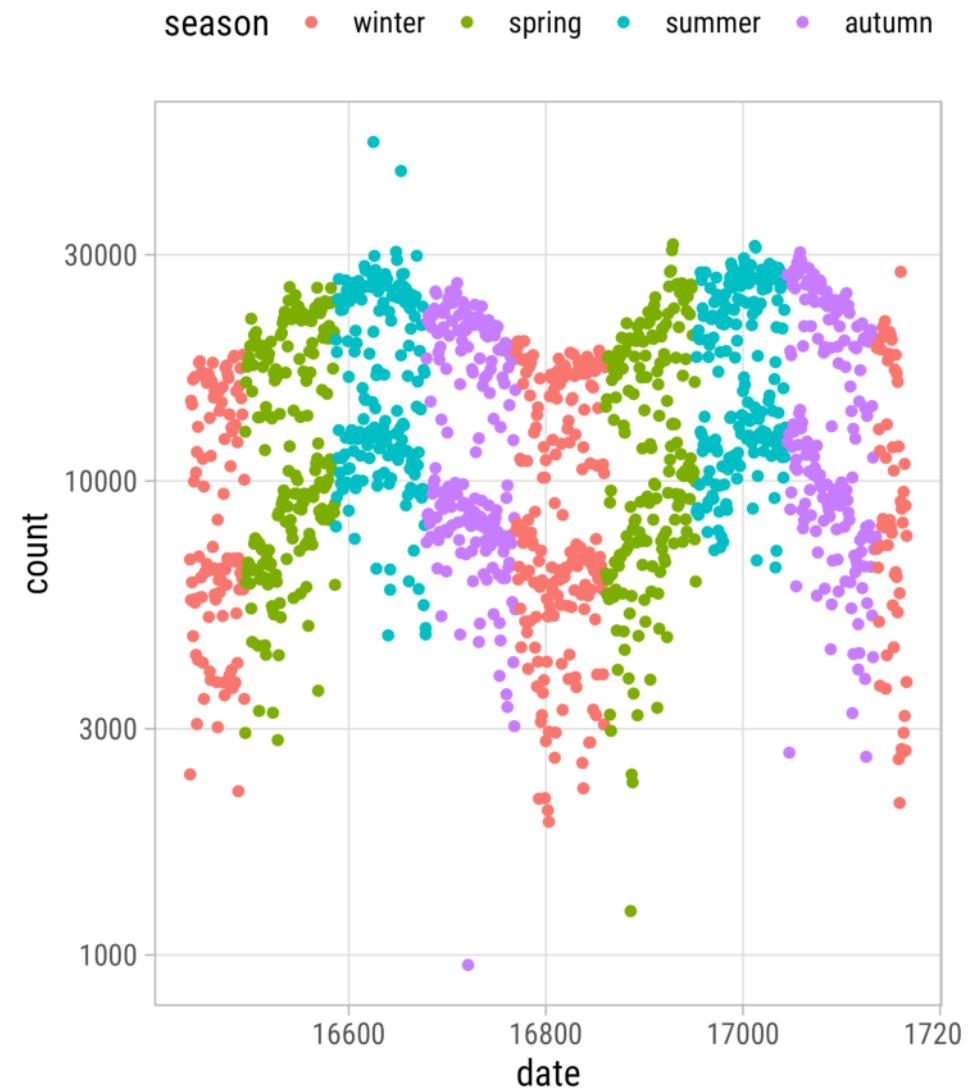
# Scales

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_continuous() +  
8 scale_y_continuous() +  
9 scale_color_discrete()
```



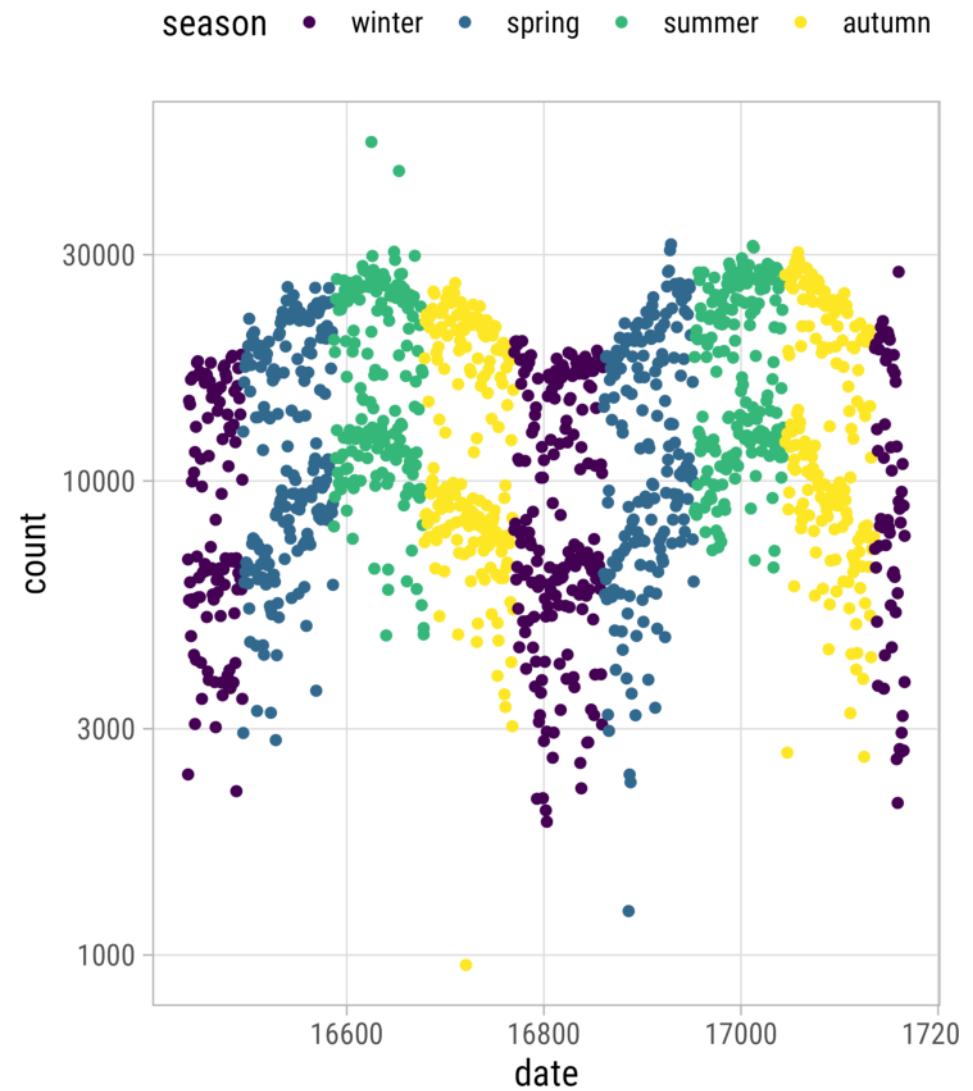
# Scales

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_continuous() +  
8 scale_y_log10() +  
9 scale_color_discrete()
```



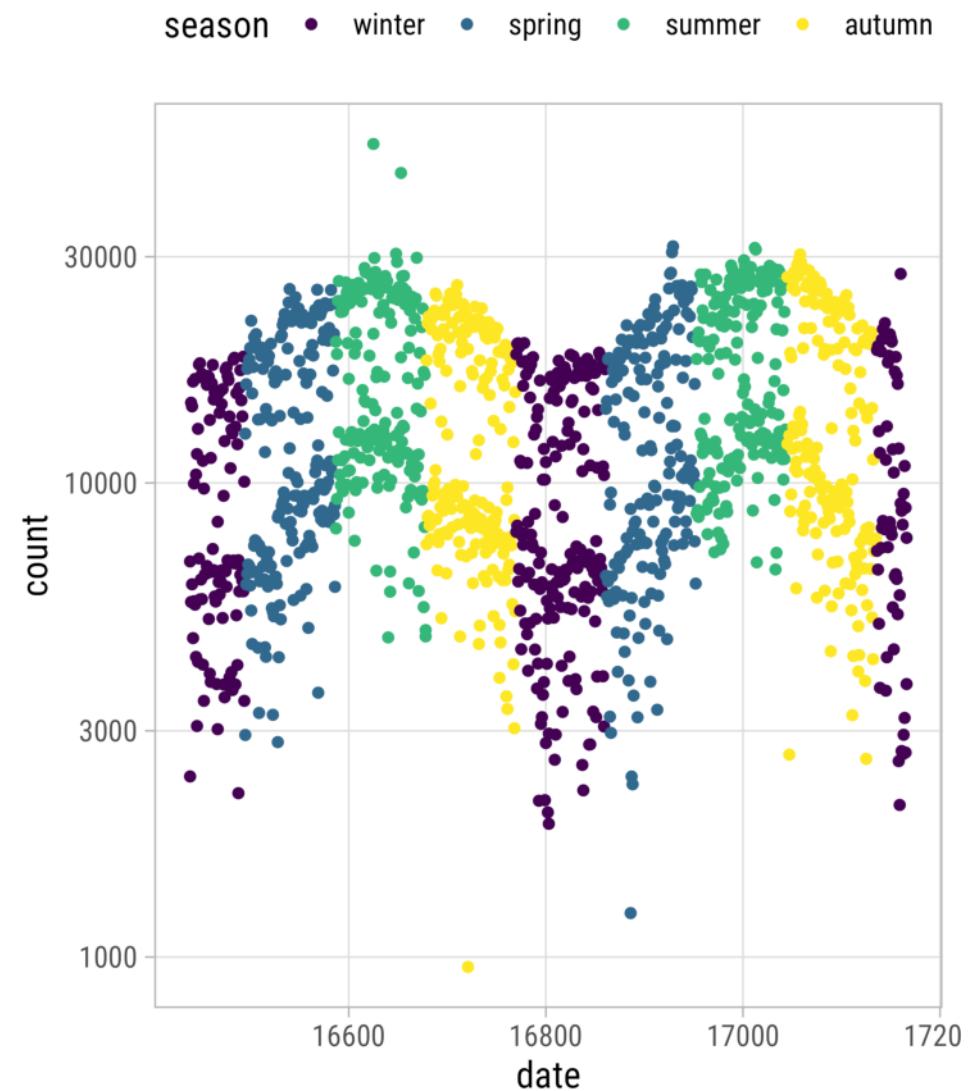
# Scales

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_continuous() +  
8 scale_y_log10() +  
9 scale_color_viridis_d()
```



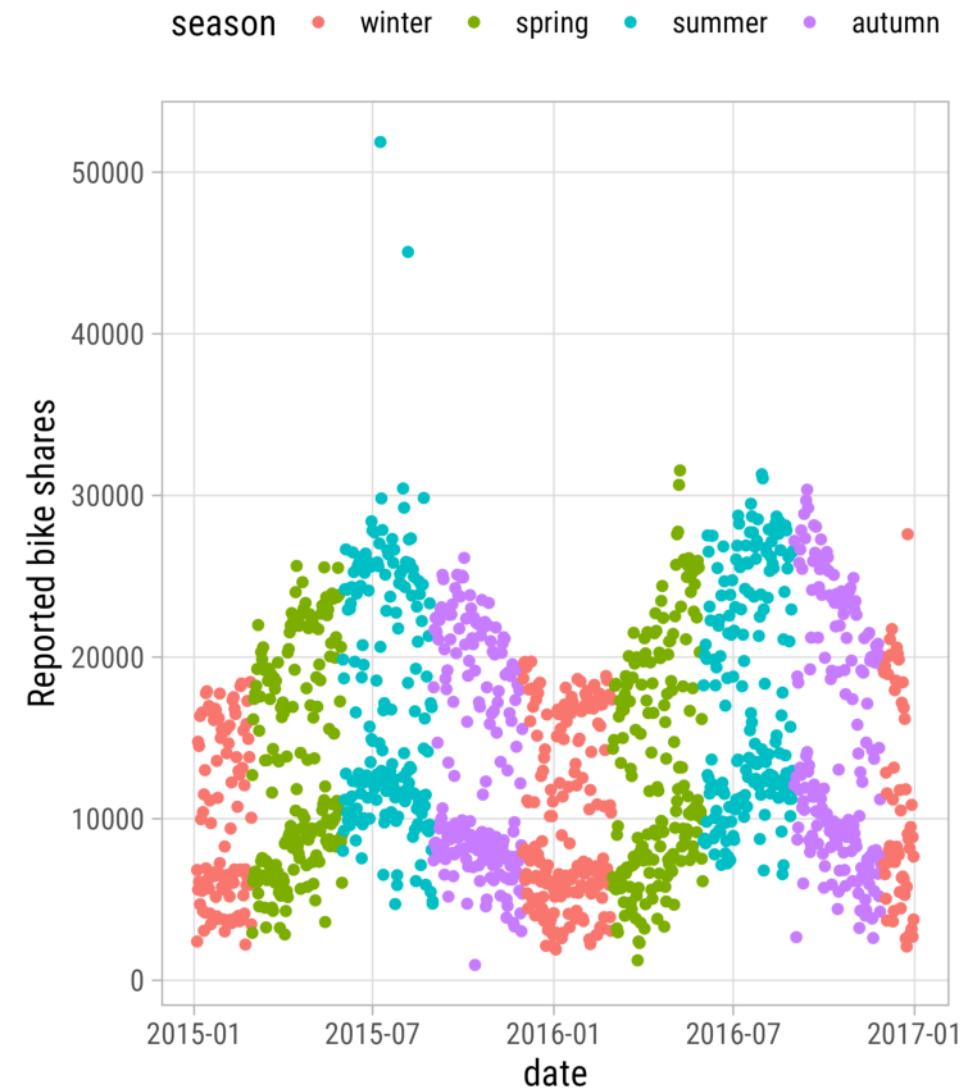
# `scale\_x|y\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6   geom_point() +  
7   scale_x_continuous() +  
8   scale_y_continuous(  
9     trans = "log10"  
10 ) +  
11   scale_color_viridis_d()
```



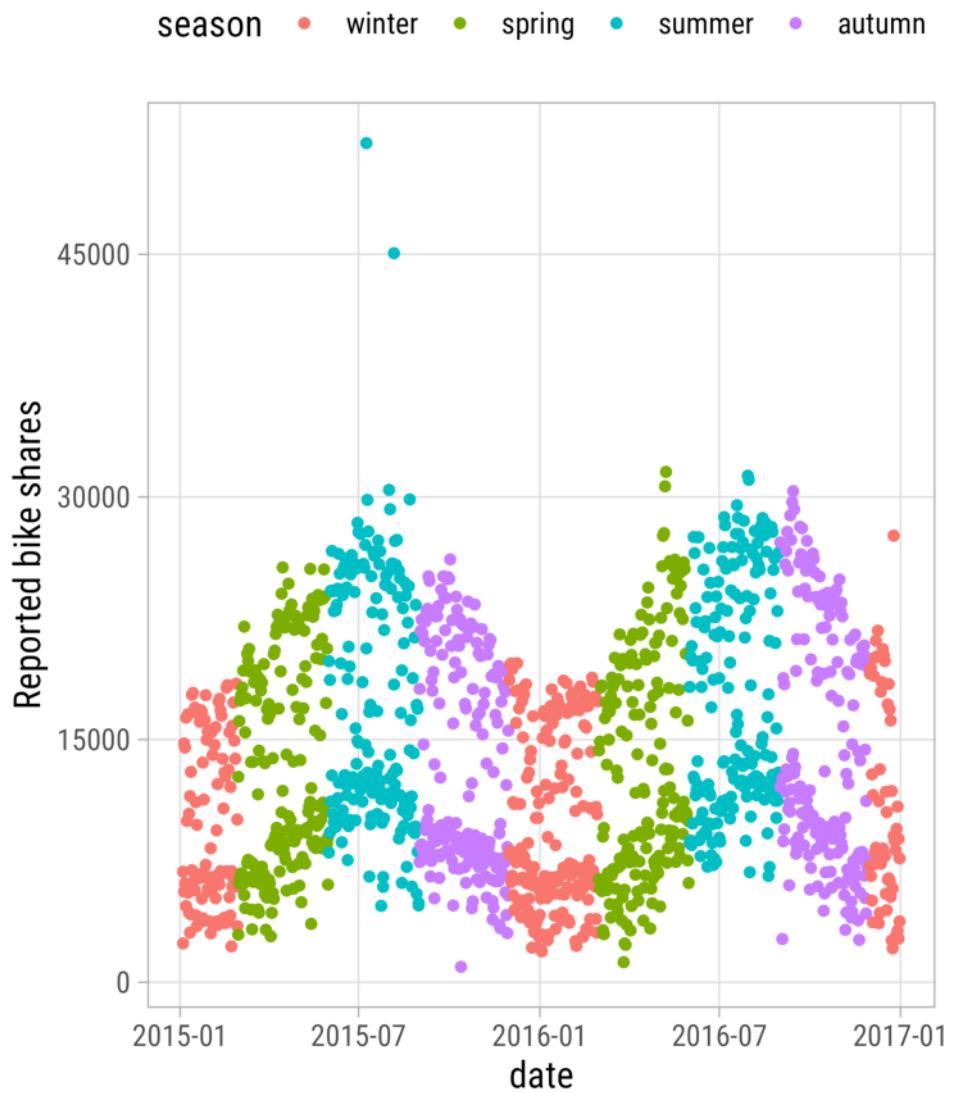
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares"  
9 )
```



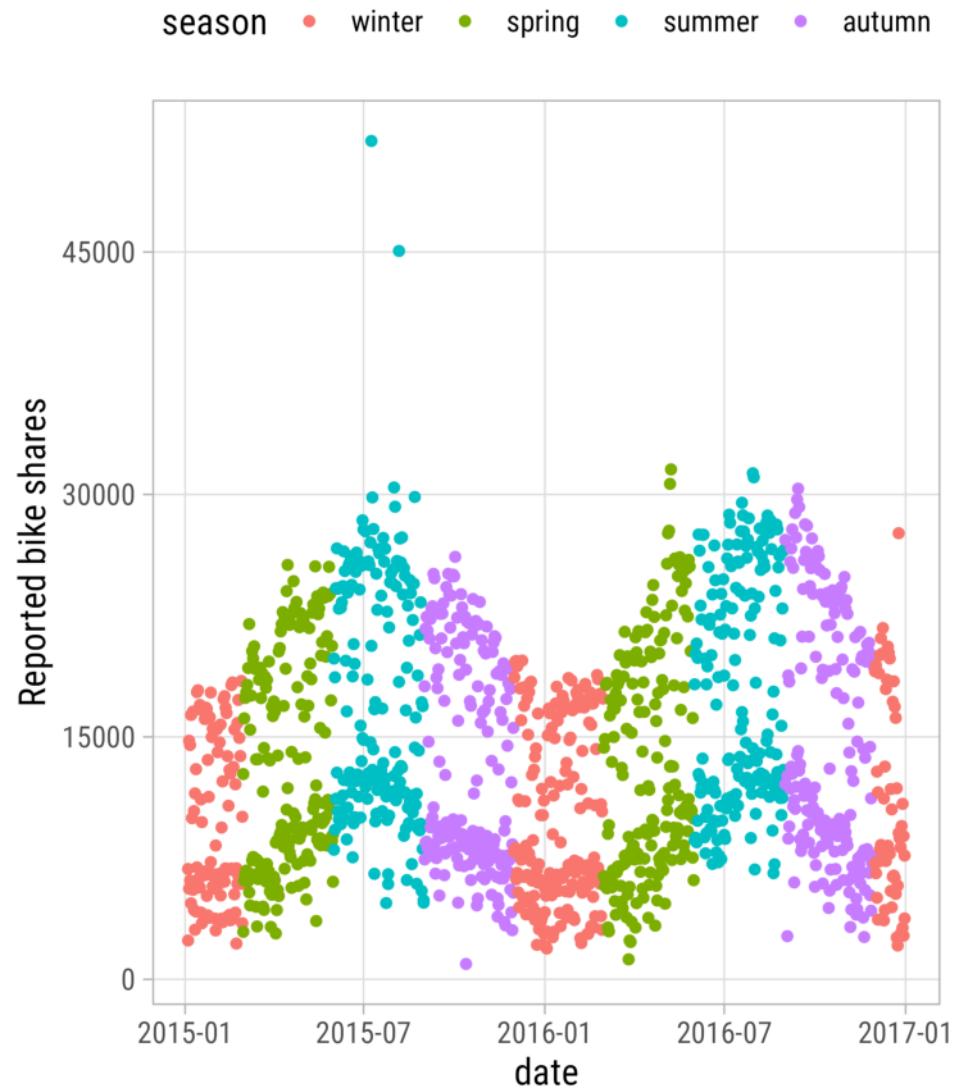
# `scale\_x|y\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = seq(0, 60000, by = 15000))  
10 )
```



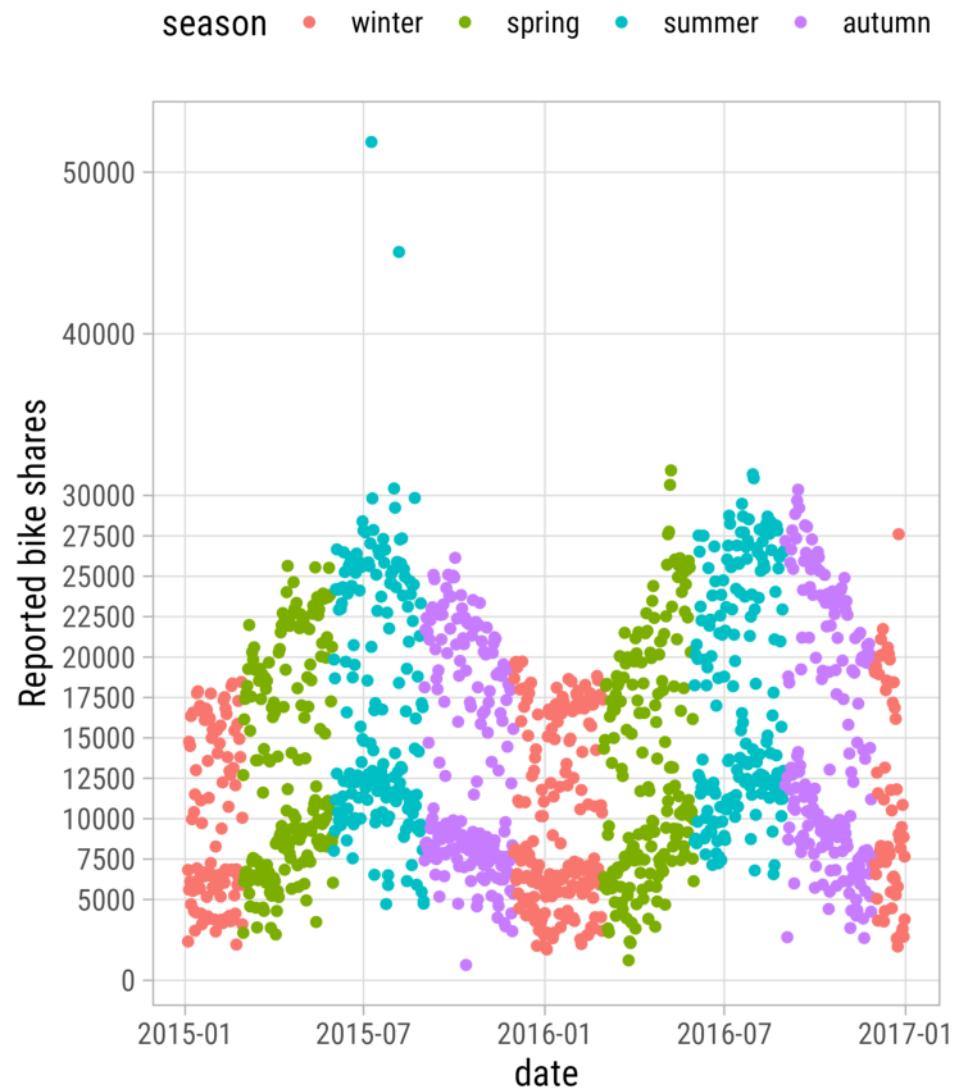
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = 0:4*15000  
10 )
```



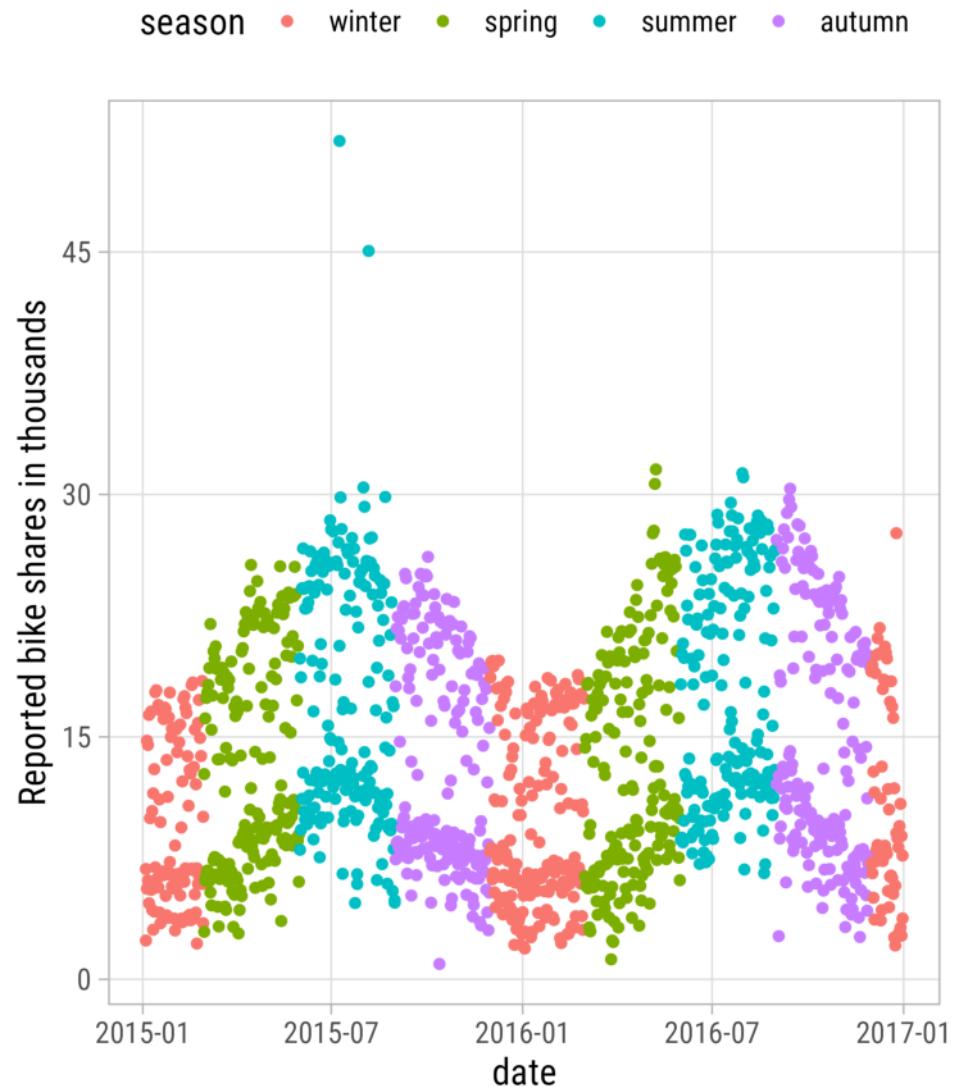
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks =  
10  c(0, seq(5000, 30000, by = 2500), 40000  
11 )
```



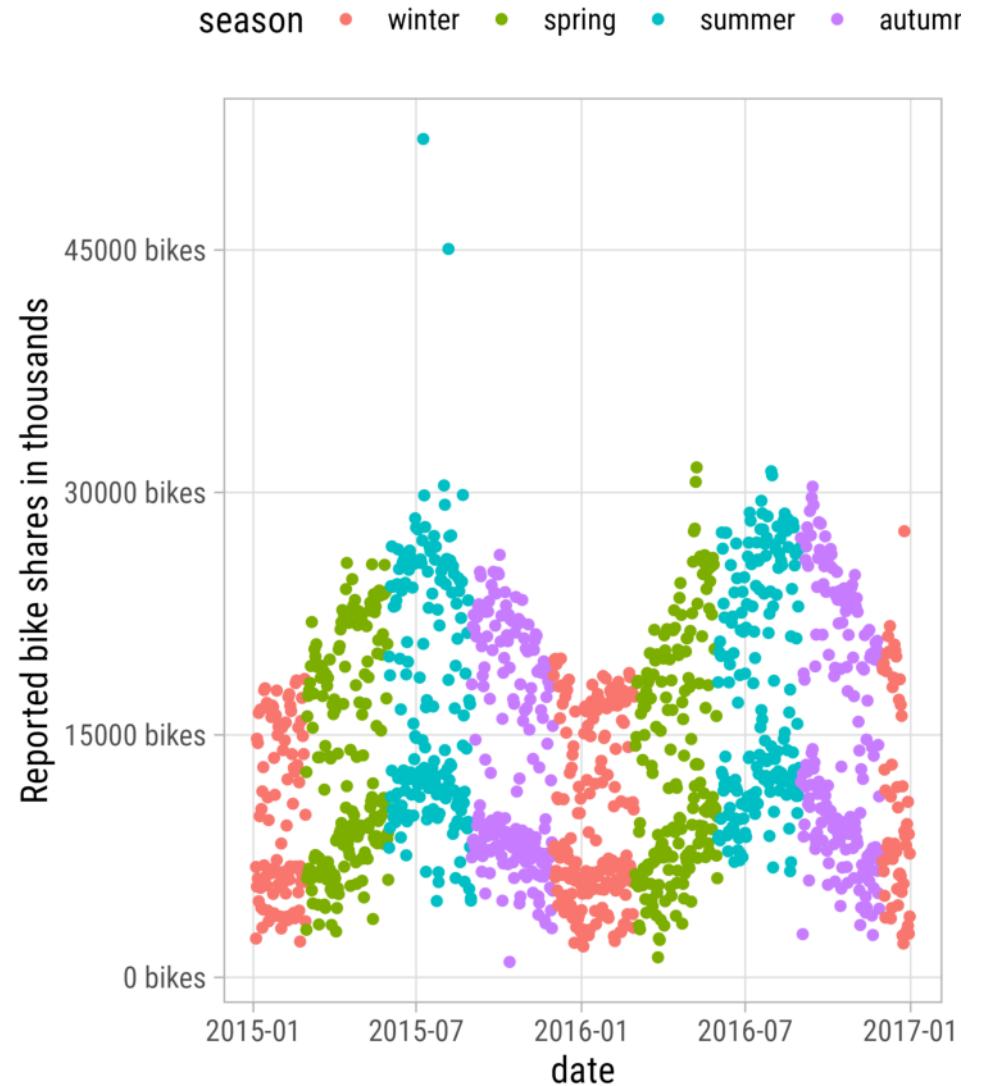
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares in thousands"  
9   breaks = 0:4*15000,  
10  labels = 0:4*15  
11 )
```



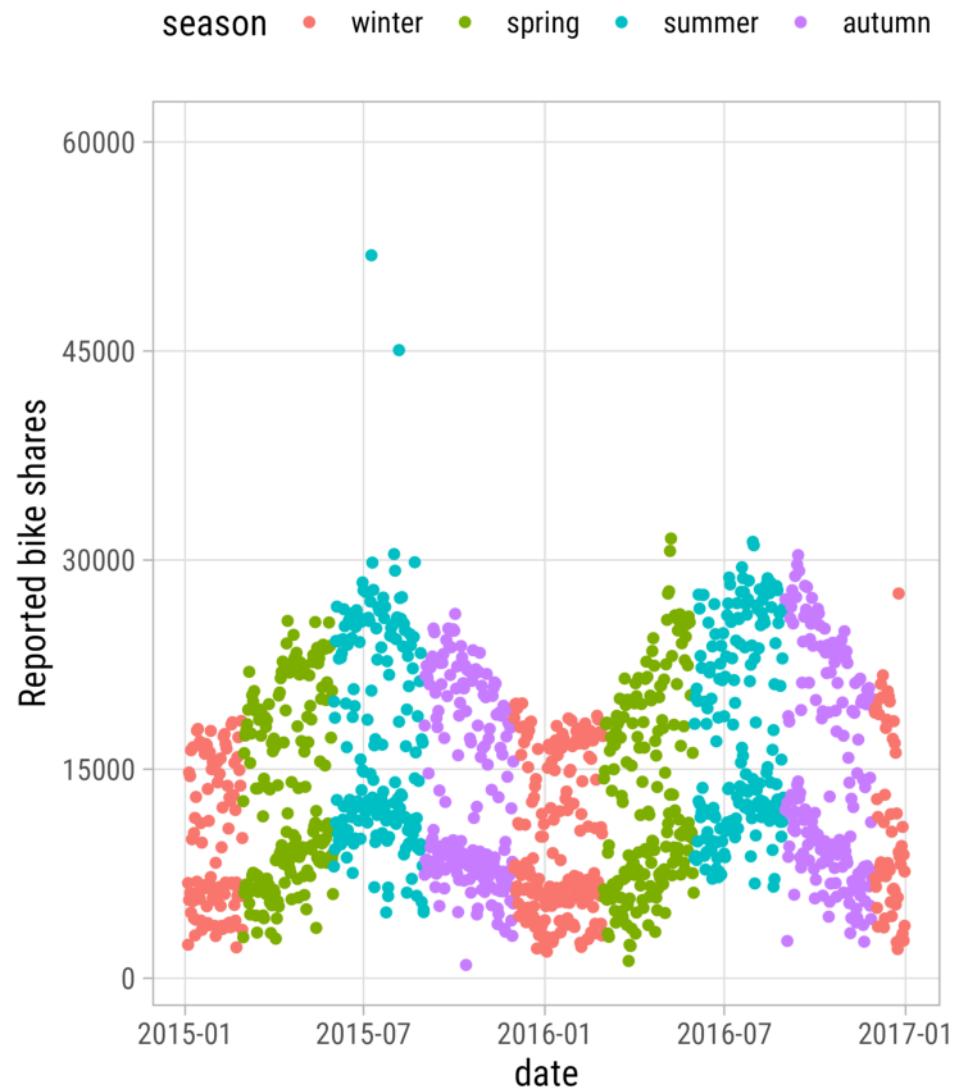
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares in thousands"  
9   breaks = 0:4*15000,  
10  labels = paste(0:4*15000, "bikes")  
11 )
```



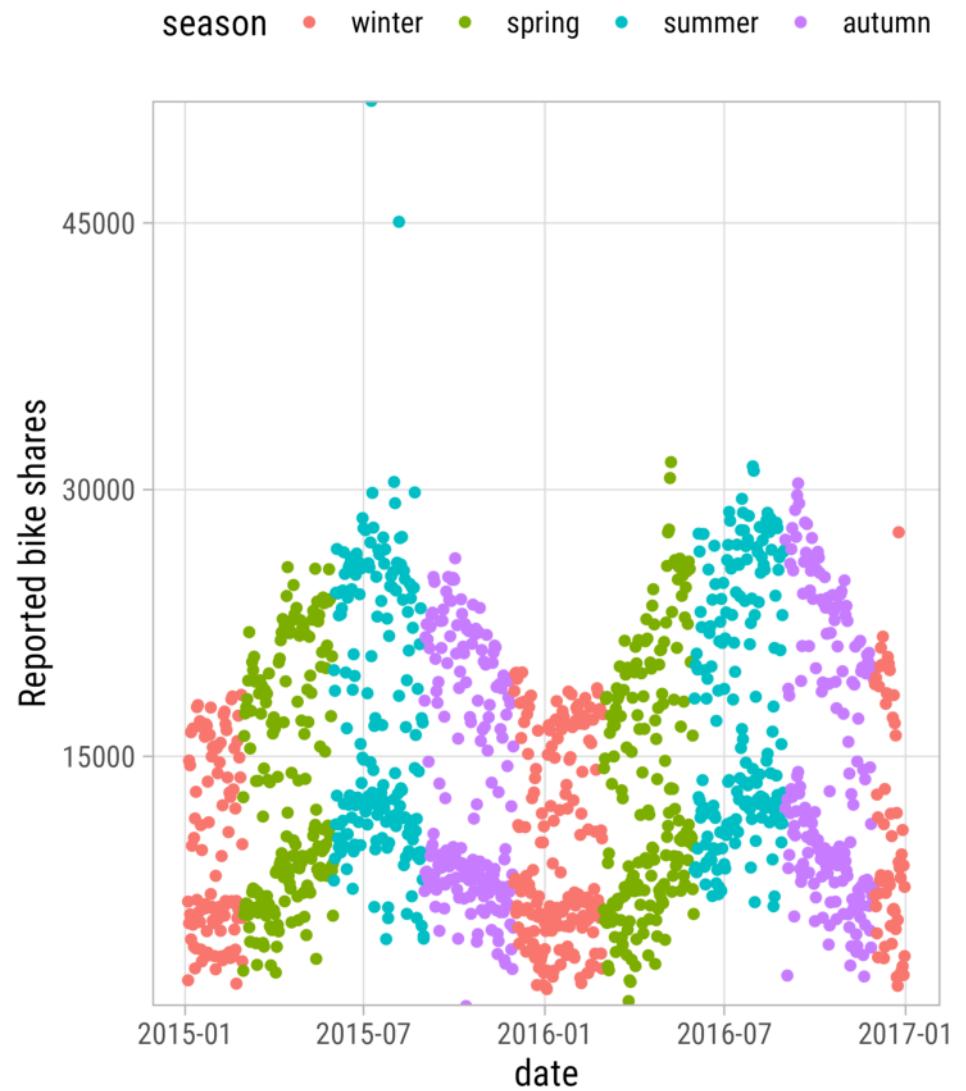
# `scale\_x|y\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = 0:4*15000,  
10  limits = c(NA, 60000)  
11 )
```



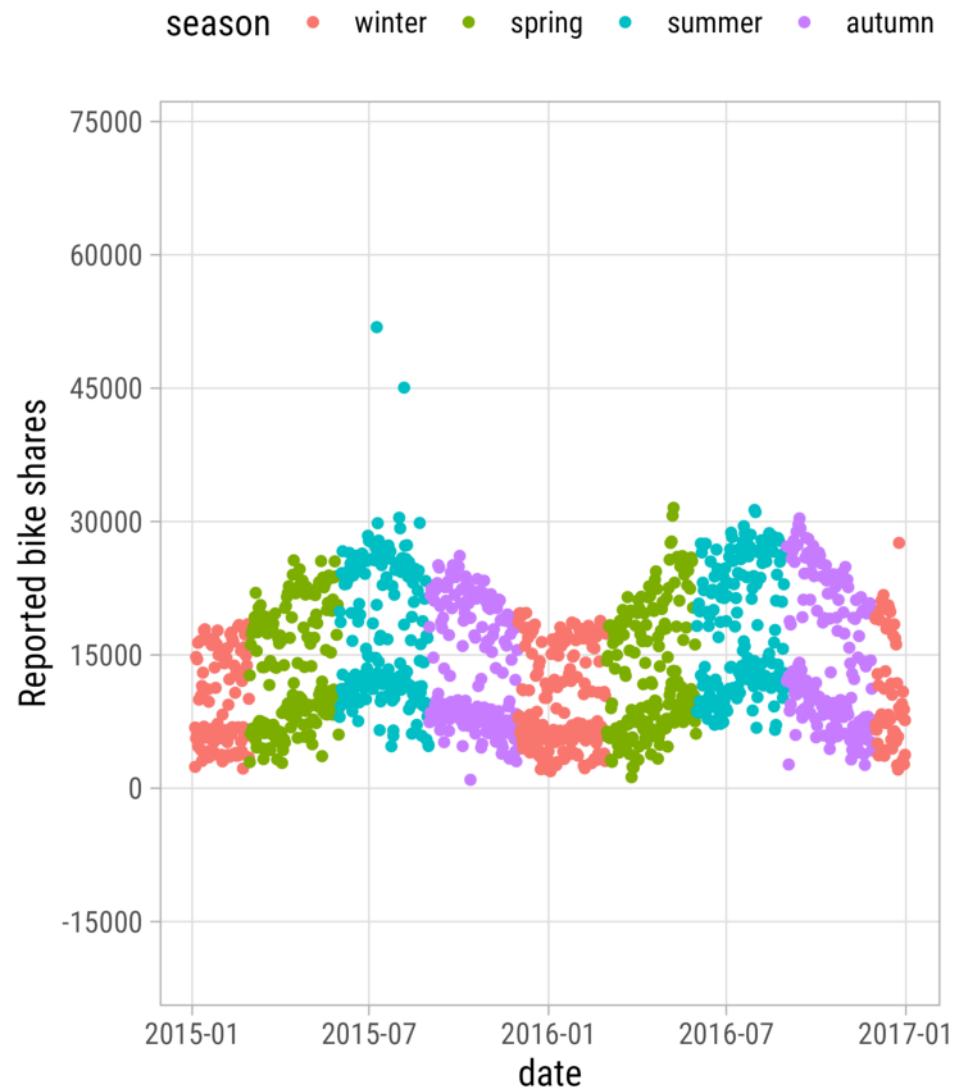
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = 0:4*15000,  
10  expand = c(0, 0)  
11 )
```



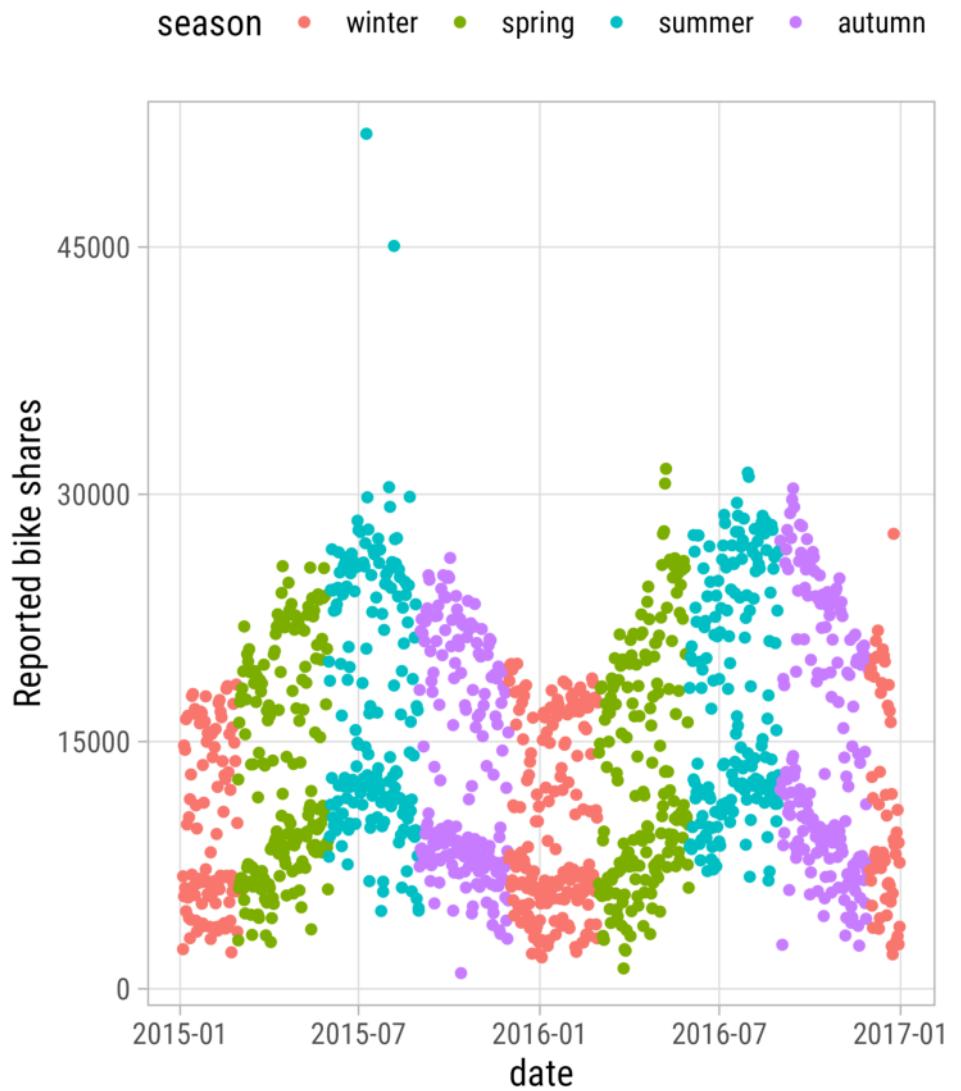
# `scale\_x|y\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = -1:5*15000,  
10  expand = c(.5, .5) ## c(add, mult)  
11 )
```



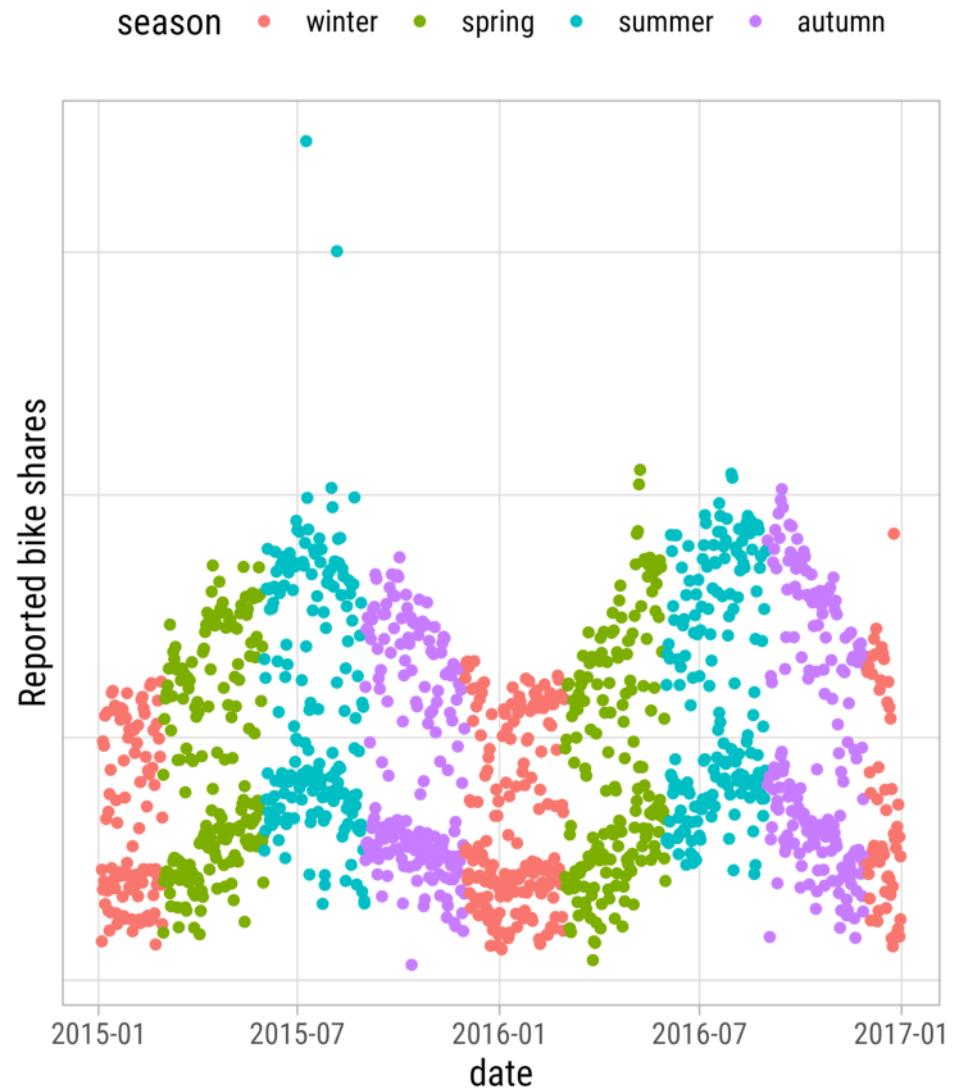
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = -1:5*15000,  
10  expand = expansion(add = 2000)  
11 )
```



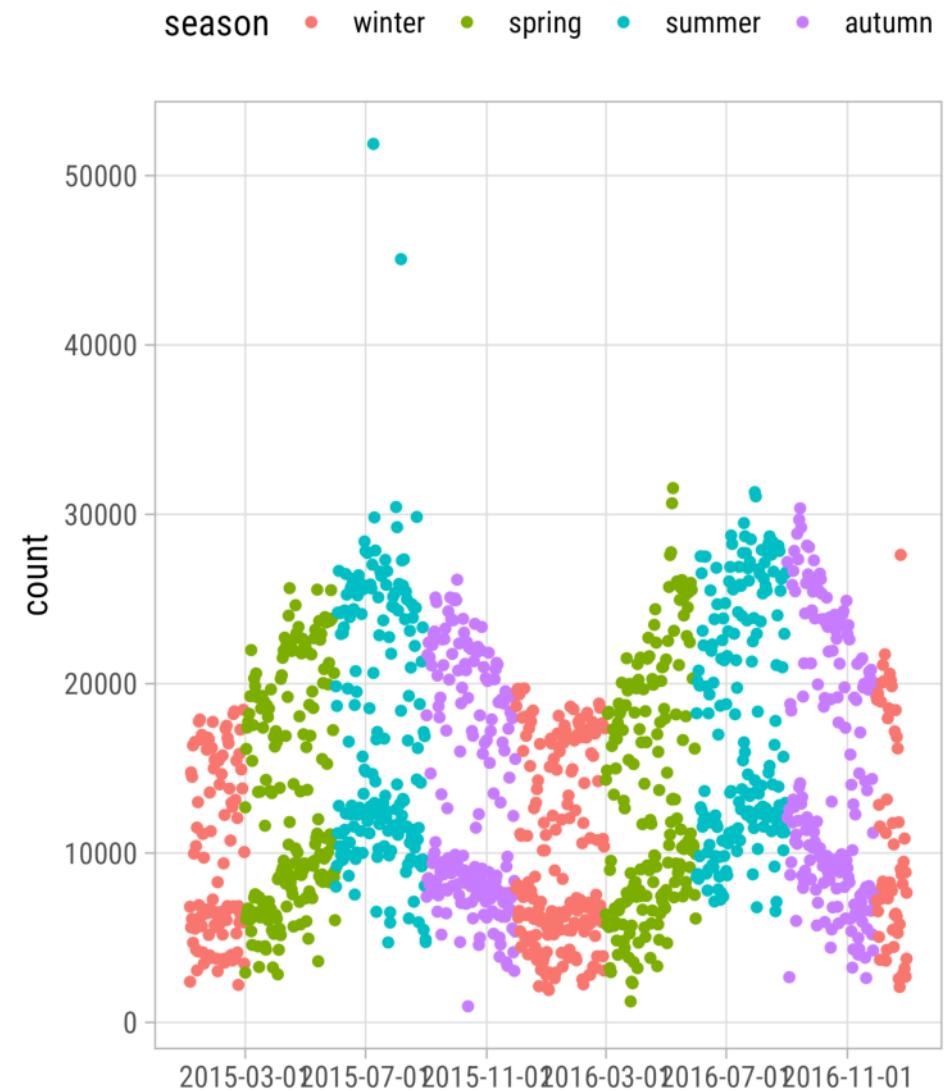
# `scale\_xly\_continuous`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_y_continuous(  
8   name = "Reported bike shares",  
9   breaks = 0:4*15000,  
10  guide = "none"  
11 )
```



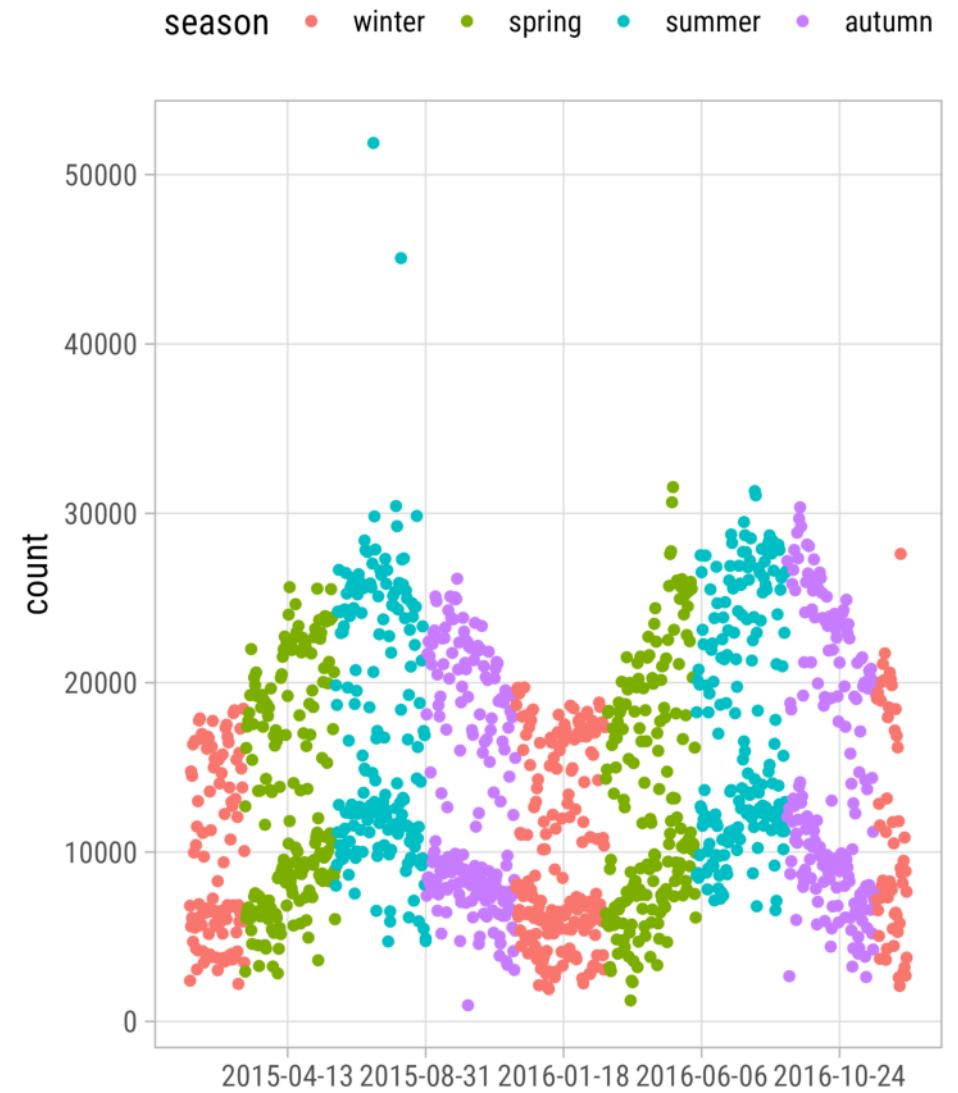
# `scale\_x|y\_date`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_date(  
8   name = NULL,  
9   date_breaks = "4 months"  
10 )
```



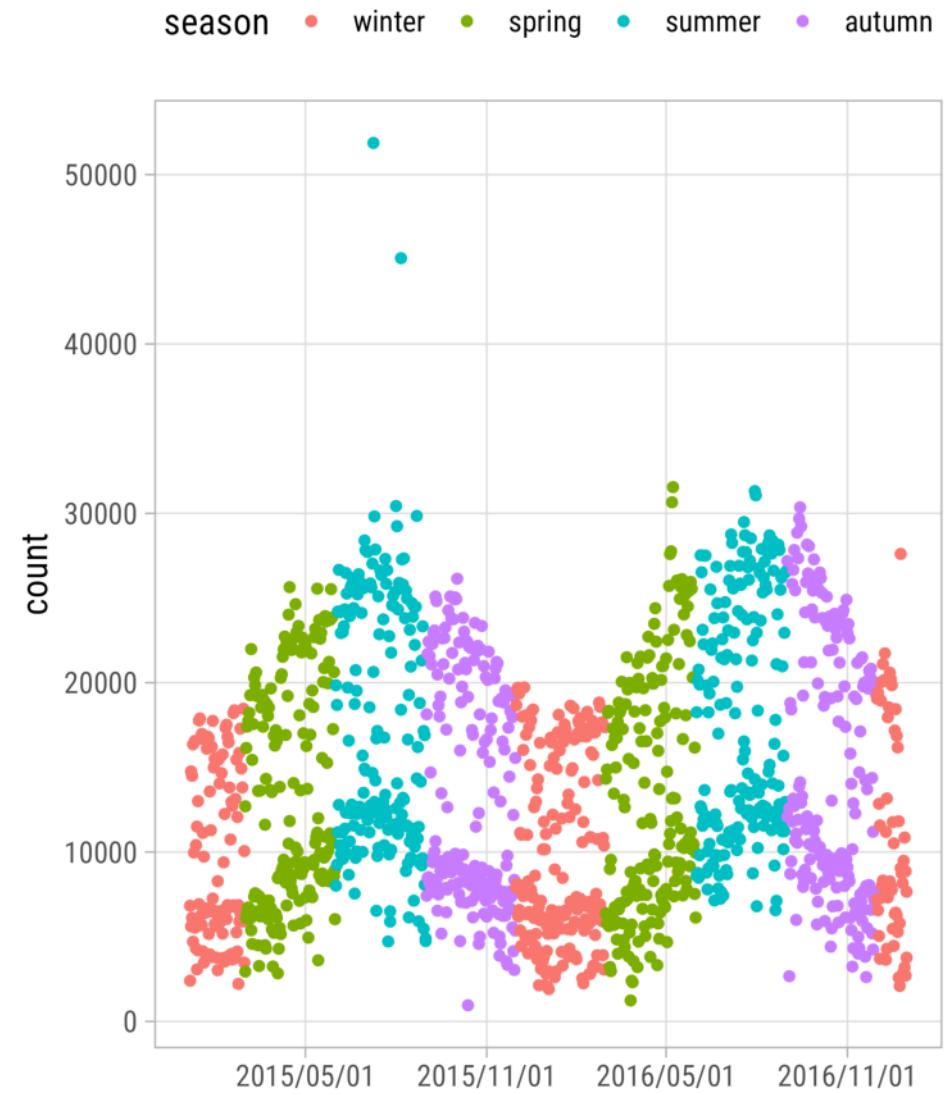
# `scale\_x|y\_date`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_date(  
8   name = NULL,  
9   date_breaks = "20 weeks"  
10 )
```



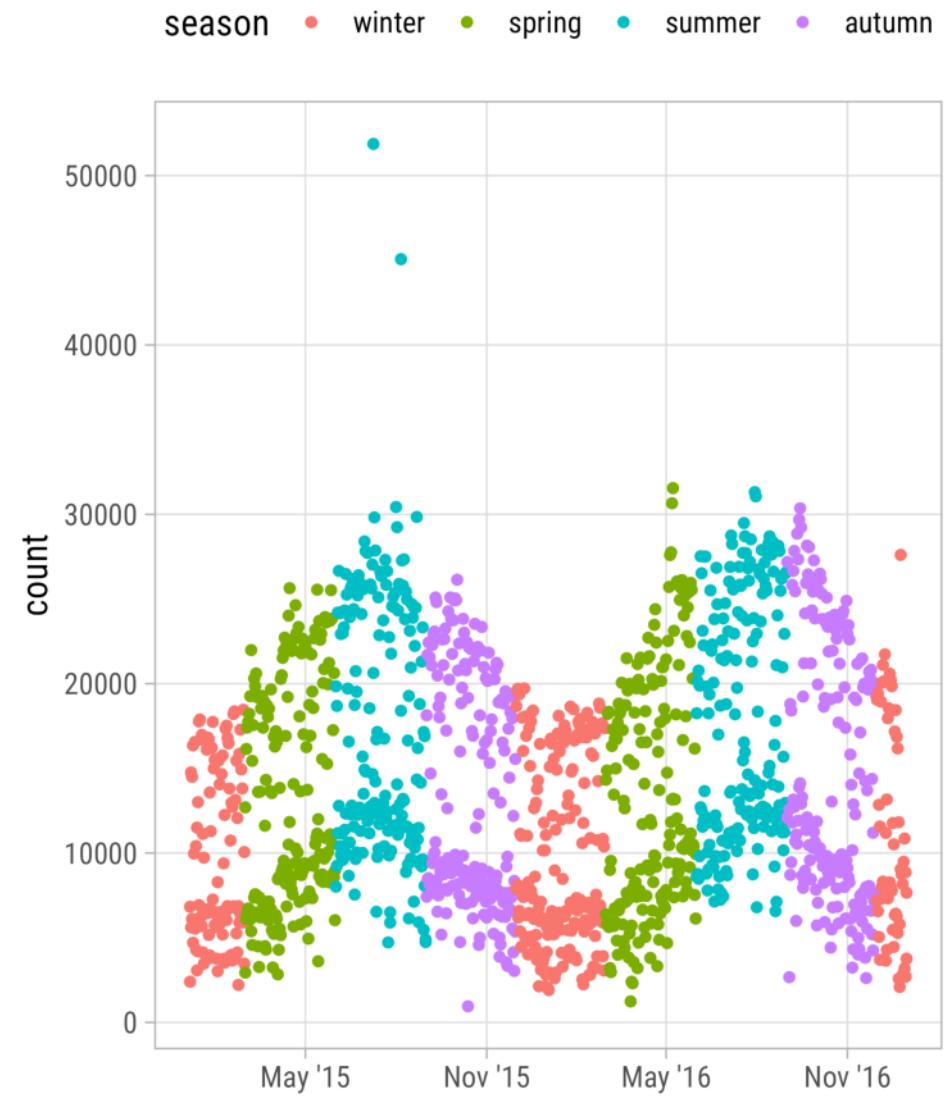
# ``scale_x|y_date` with `strftime()``

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_date(  
8   name = NULL,  
9   date_breaks = "6 months",  
10  date_labels = "%Y/%m/%d"  
11 )
```



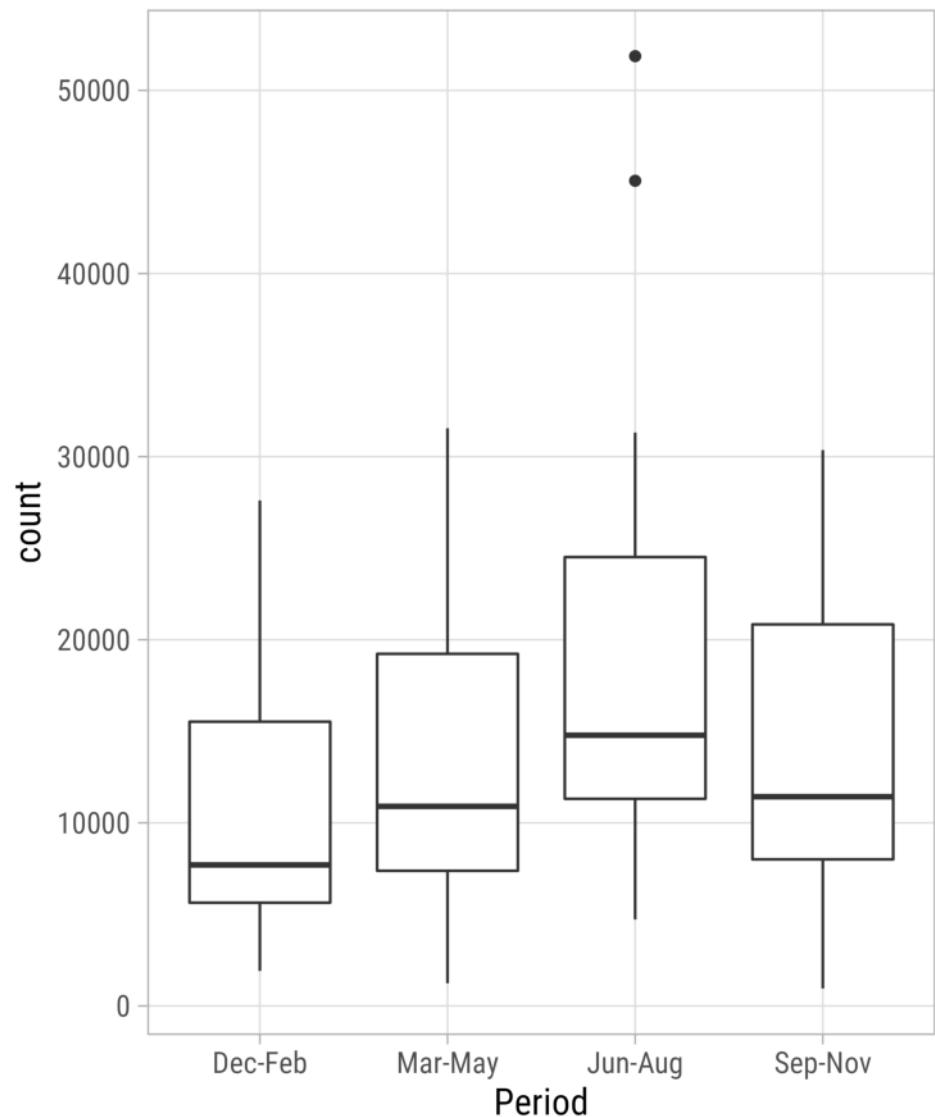
# ``scale_x|y_date` with `strftime()``

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_x_date(  
8   name = NULL,  
9   date_breaks = "6 months",  
10  date_labels = "%b %y"  
11 )
```



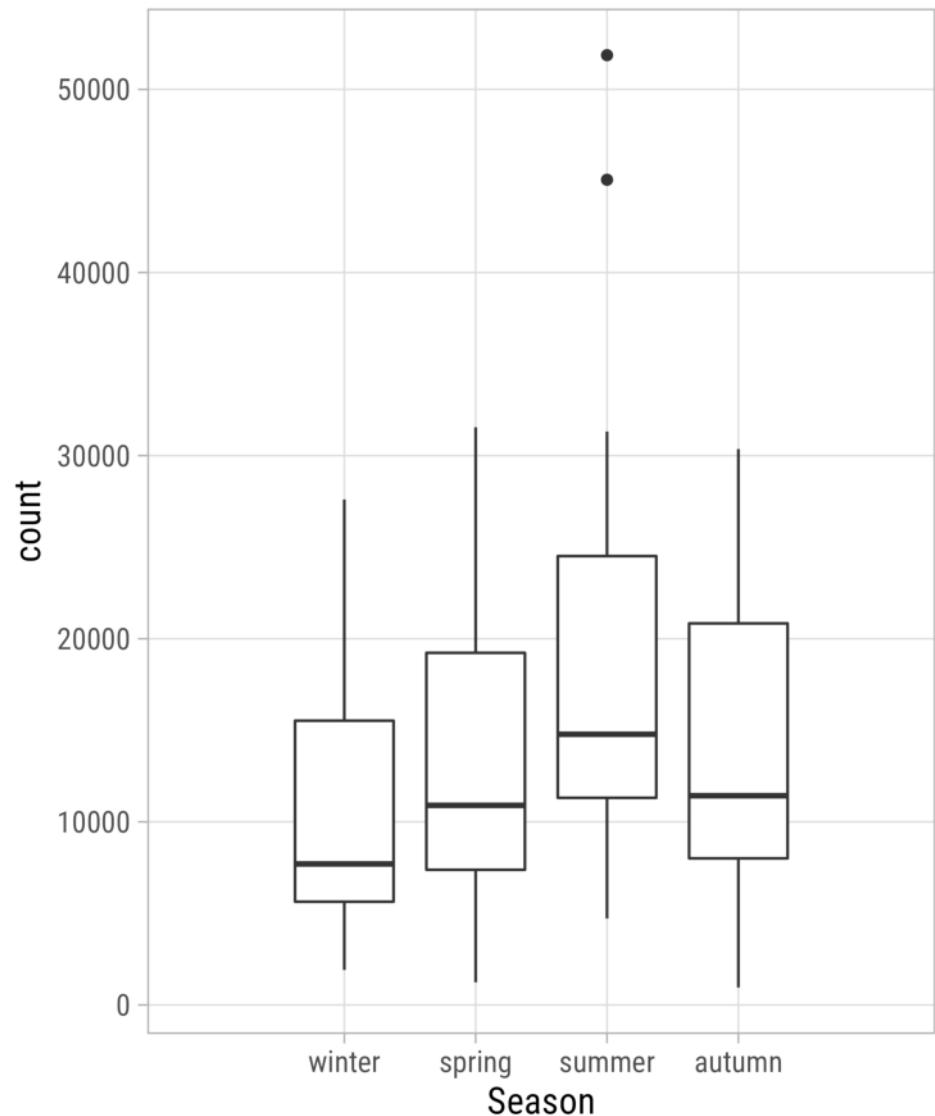
# `scale\_x|y\_discrete`

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 scale_x_discrete(  
7   name = "Period",  
8   labels = c("Dec-Feb", "Mar-May", "Jun-Aug")  
9 )
```



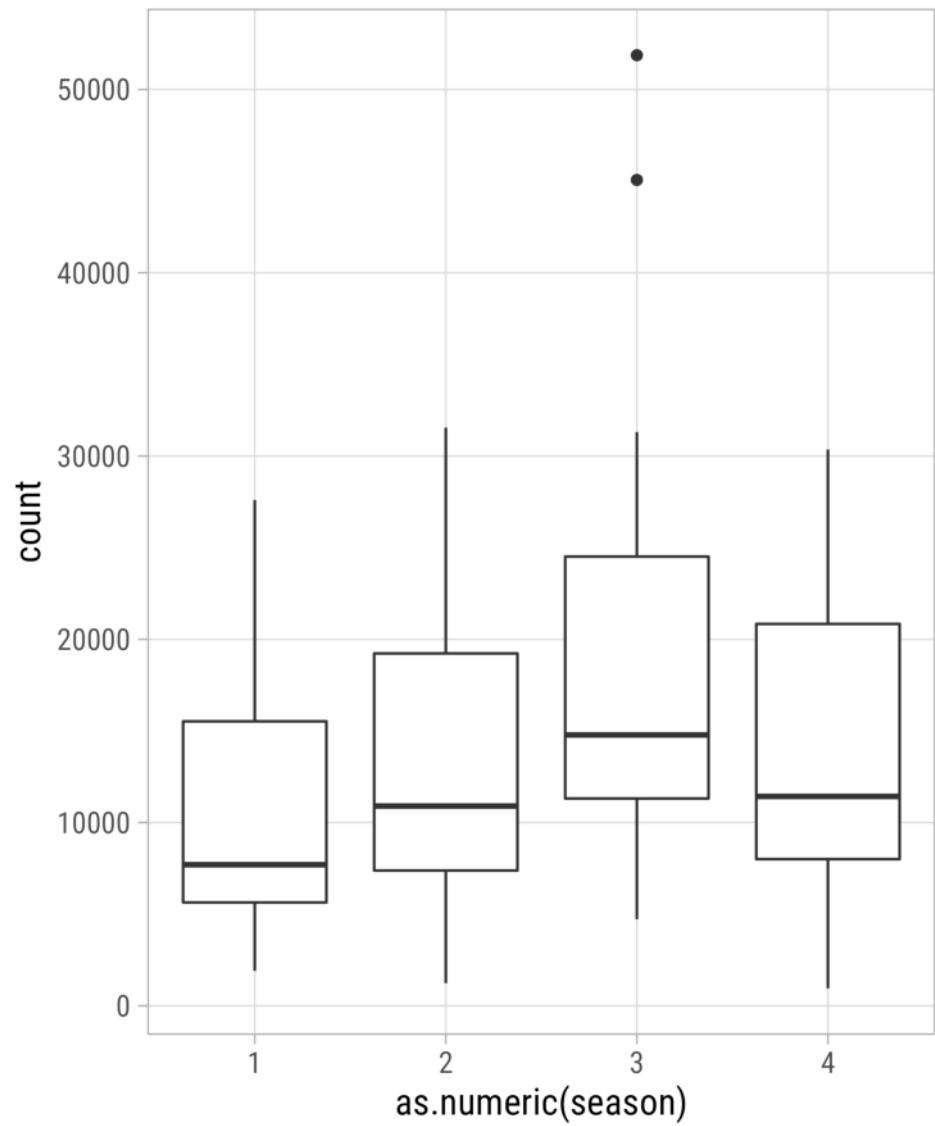
# `scale\_x|y\_discrete`

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 scale_x_discrete(  
7   name = "Season",  
8   expand = c(.5, 0) ## add, mult  
9 )
```



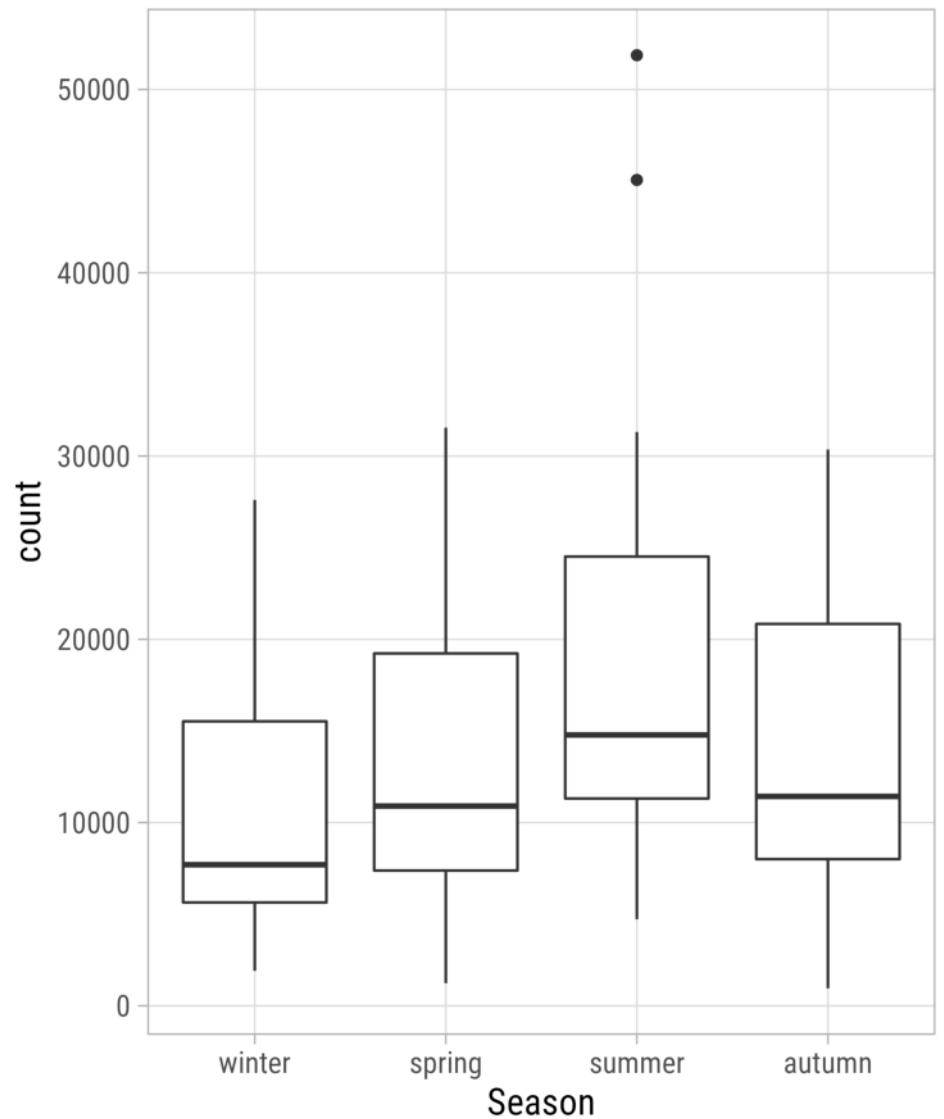
# Discrete or Continuous?

```
1 ggplot(  
2   bikes,  
3   aes(x = as.numeric(season), y = count)  
4 ) +  
5   geom_boxplot(  
6   aes(group = season)  
7 )
```



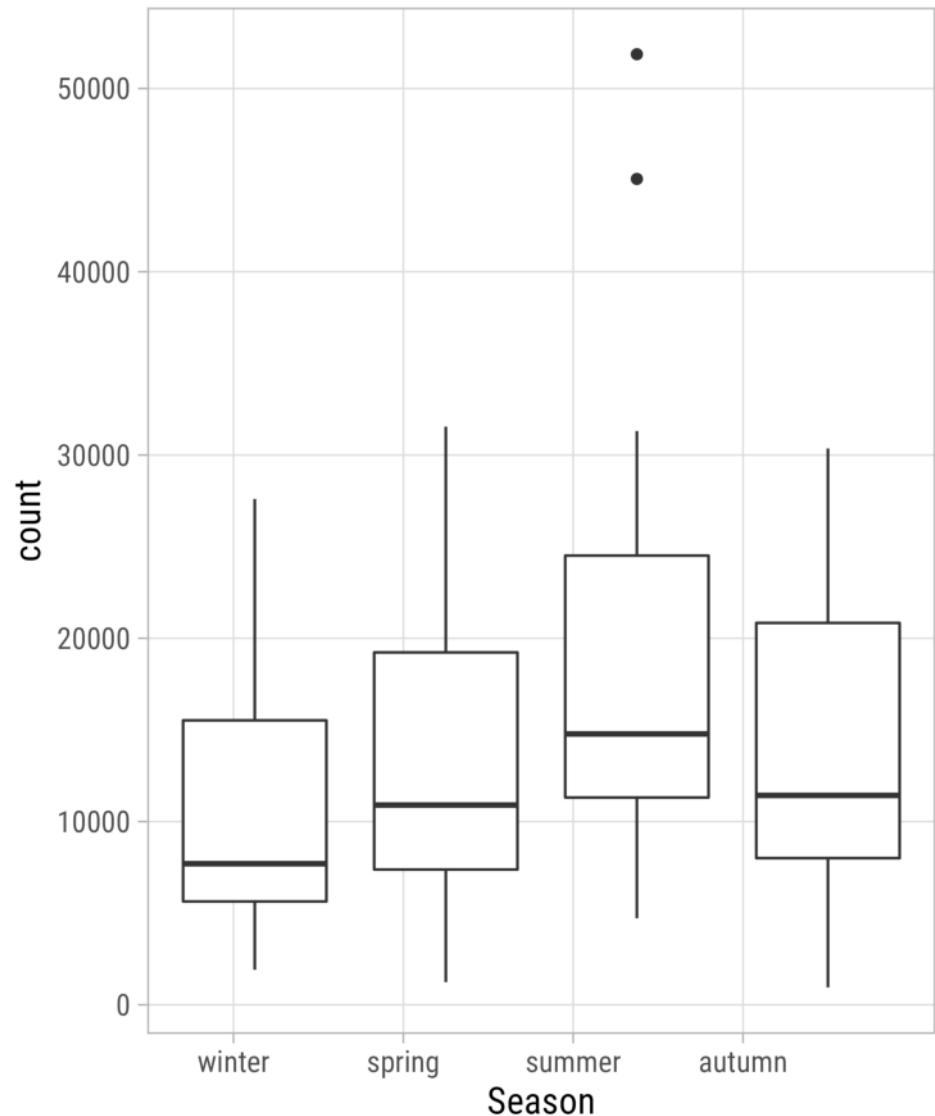
# Discrete or Continuous?

```
1 ggplot(  
2   bikes,  
3   aes(x = as.numeric(season),  
4       y = count)  
5 ) +  
6   geom_boxplot(  
7   aes(group = season))  
8 ) +  
9   scale_x_continuous(  
10  name = "Season",  
11  breaks = 1:4,  
12  labels = levels(bikes$season))  
13 )
```



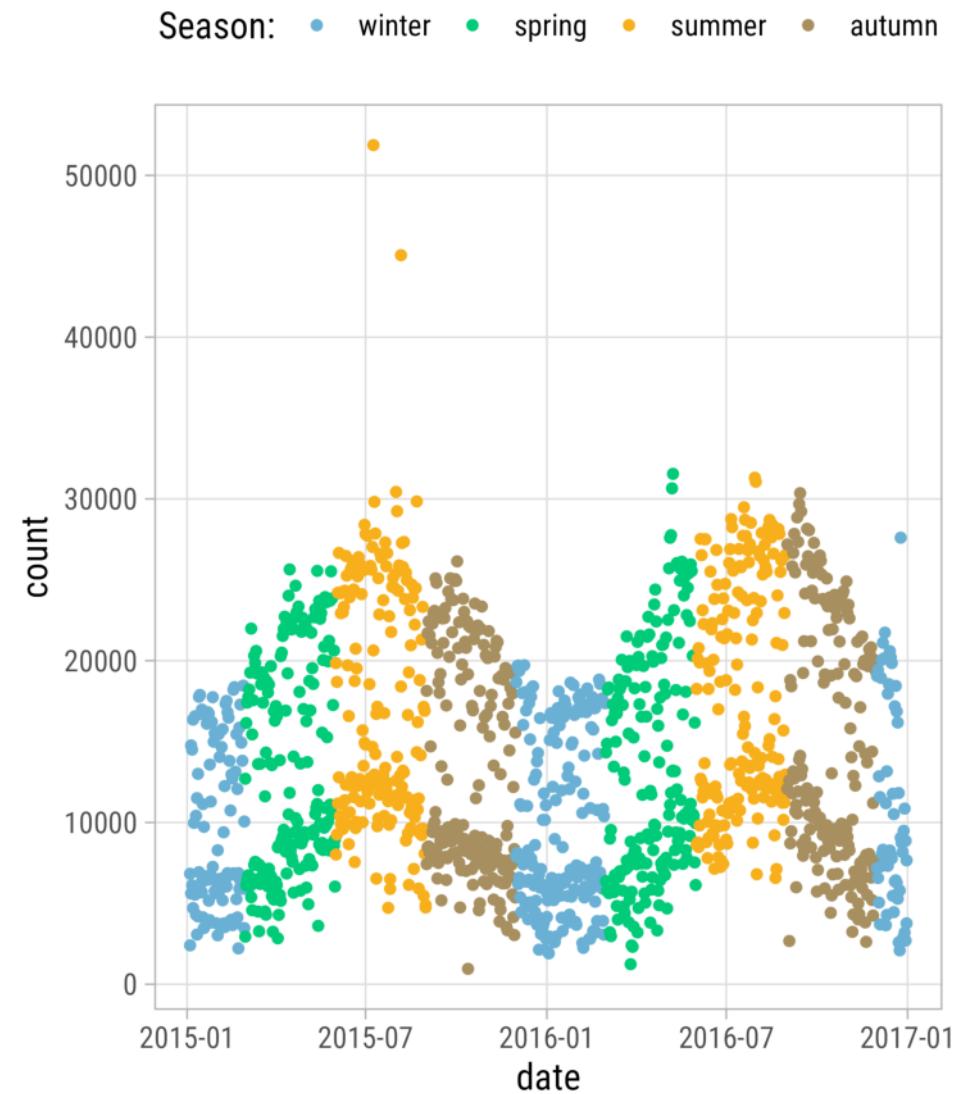
# Discrete or Continuous?

```
1 ggplot(  
2   bikes,  
3   aes(x = as.numeric(season) +  
4         as.numeric(season) / 8,  
5         y = count)  
6 ) +  
7   geom_boxplot(  
8     aes(group = season)  
9   ) +  
10  scale_x_continuous(  
11    name = "Season",  
12    breaks = 1:4,  
13    labels = levels(bikes$season)  
14 )
```



# `scale\_color|fill\_discrete`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = season)  
5 ) +  
6 geom_point() +  
7 scale_color_discrete(  
8   name = "Season:",  
9   type = c("#69b0d4", "#00CB79", "#F7B01B",  
10 ))
```



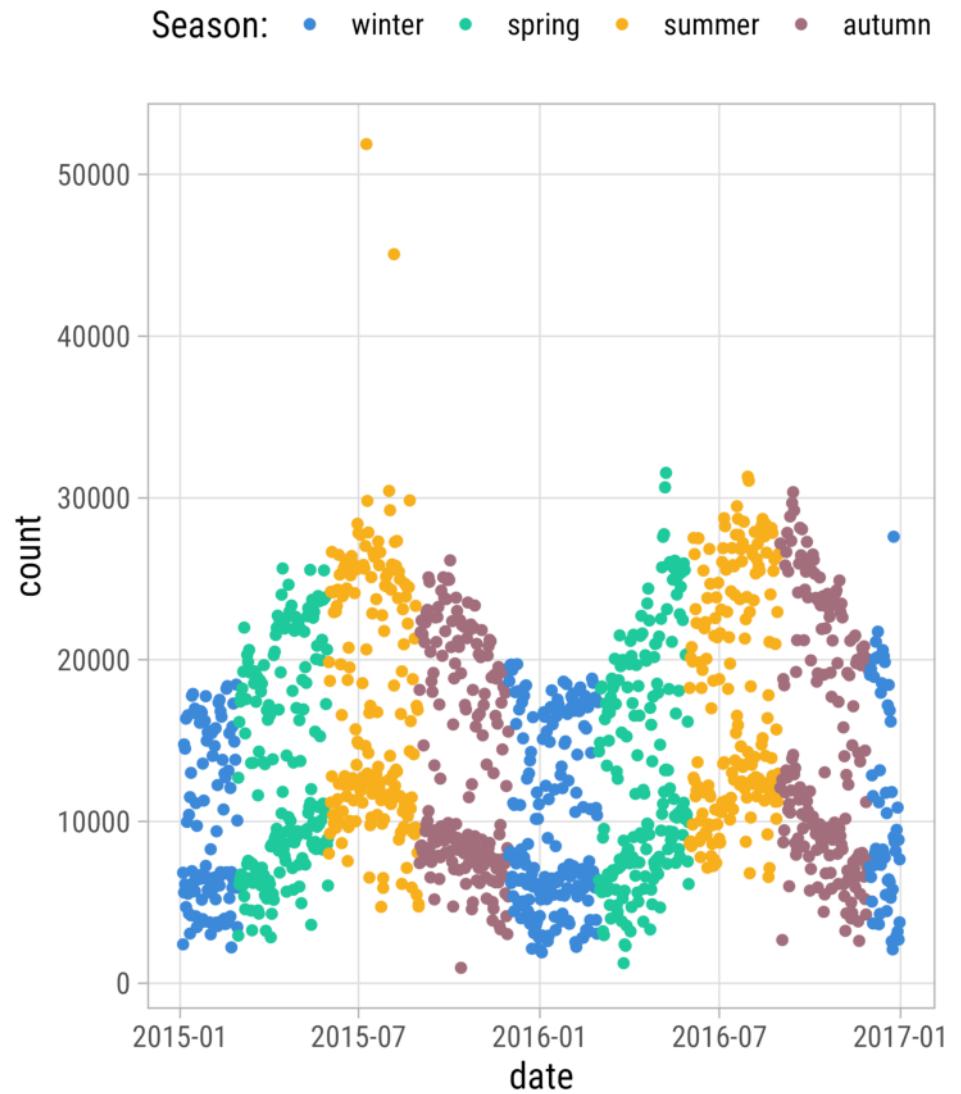
# Inspect Assigned Colors

```
1 g <- ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4     color = season)  
5 ) +  
6 geom_point() +  
7 scale_color_discrete(  
8   name = "Season:",  
9   type = c("#3ca7d9", "#1ec99b", "#F7B01B")  
10 )  
11  
12 gb <- ggplot_build(g)  
13  
14 gb$data[[1]][c(1:5, 200:205, 400:405), 1:5]
```

	colour	x	y	PANEL	group
1	#3ca7d9	16439	6830	1	1
2	#3ca7d9	16439	2404	1	1
3	#3ca7d9	16440	14763	1	1
4	#3ca7d9	16440	5609	1	1
5	#3ca7d9	16441	14501	1	1
200	#1ec99b	16538	8830	1	2
201	#1ec99b	16539	24019	1	2
202	#1ec99b	16539	10500	1	2
203	#1ec99b	16540	25640	1	2
204	#1ec99b	16540	11830	1	2
205	#1ec99b	16541	22216	1	2
400	#F7B01B	16638	12079	1	3
401	#F7B01B	16639	26646	1	3
402	#F7B01B	16639	12446	1	3
403	"#F7B01B"	16640	11210	1	3

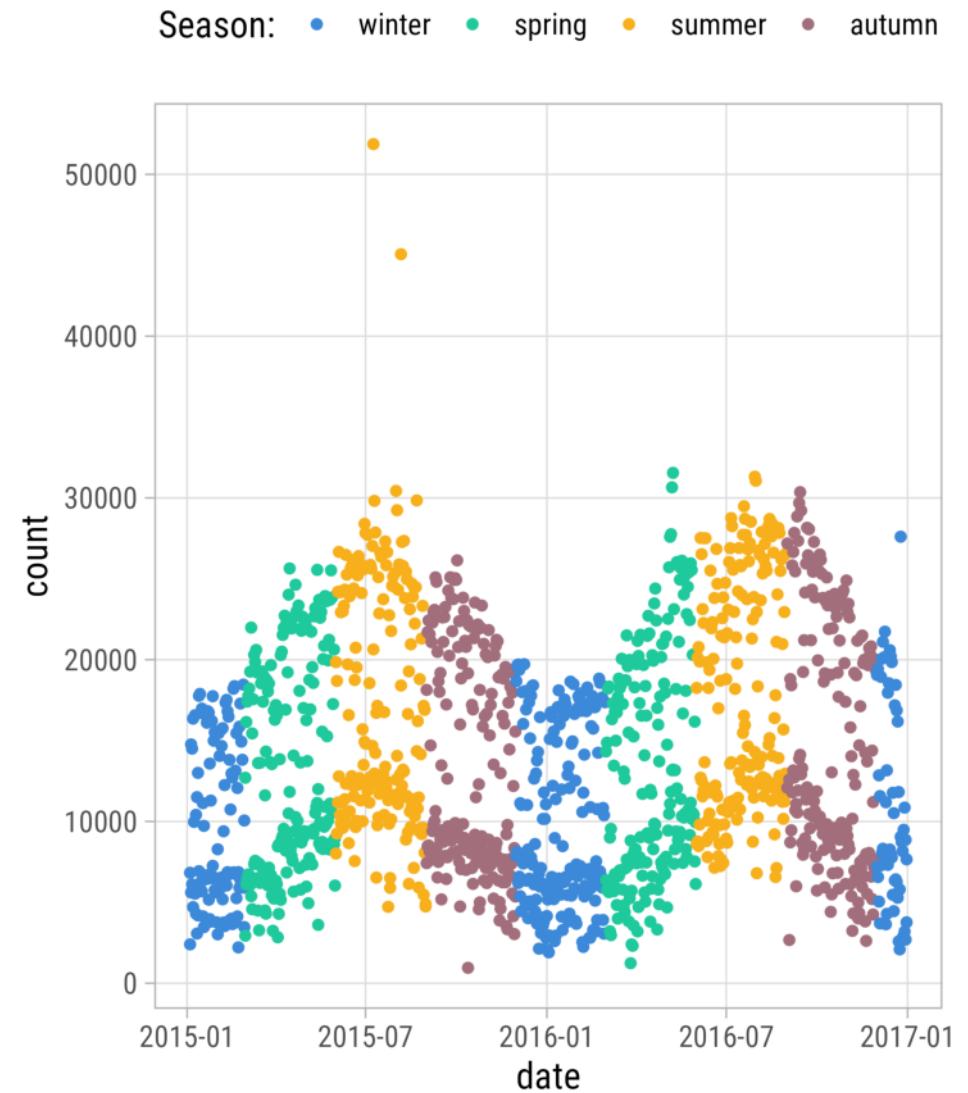
# `scale\_color|fill\_discrete`

```
1 my_colors <- c(
2   `winter` = "#3c89d9",
3   `spring` = "#1ec99b",
4   `summer` = "#F7B01B",
5   `autumn` = "#a26e7c"
6 )
7
8 ggplot(
9   bikes,
10  aes(x = date, y = count,
11       color = season)
12 ) +
13   geom_point() +
14   scale_color_discrete(
15     name = "Season:",
16     type = my_colors
17 )
```



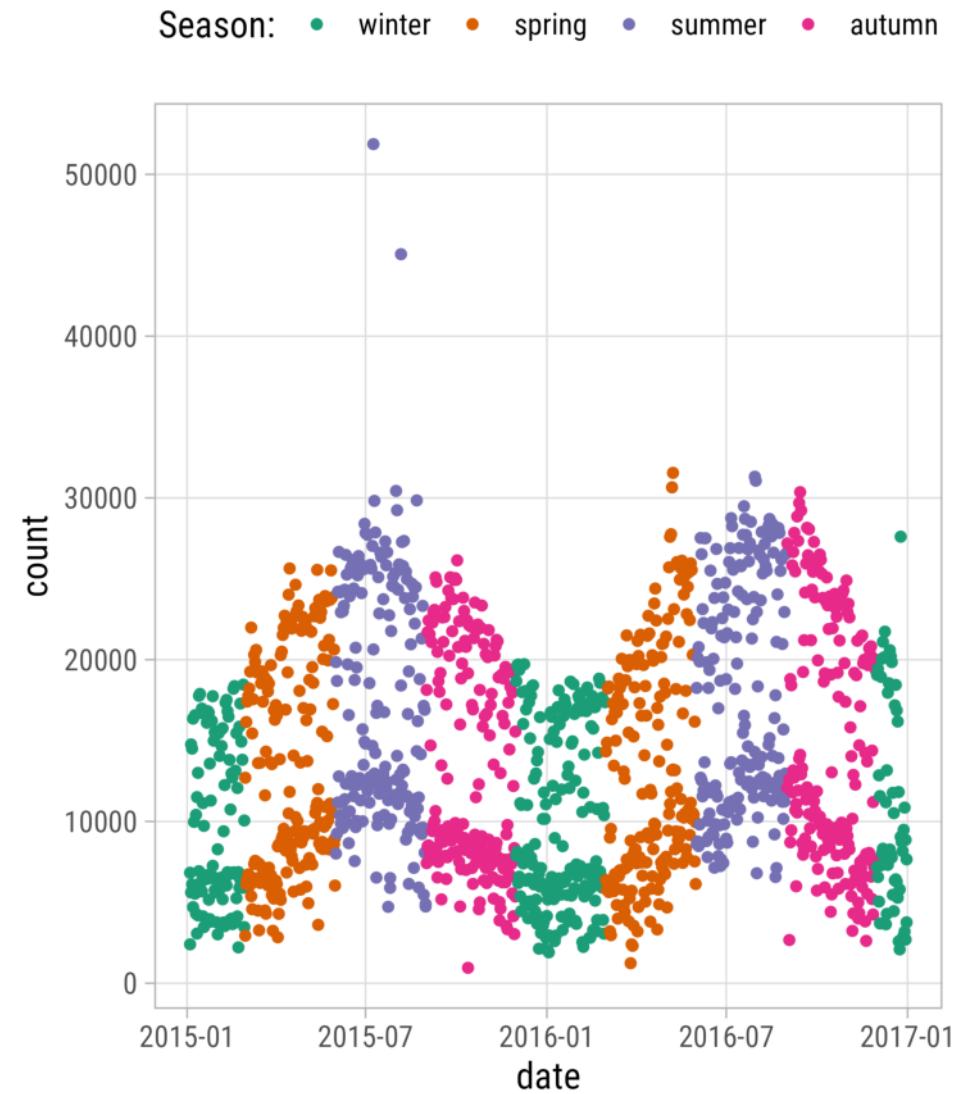
# `scale\_color|fill\_discrete`

```
1 my_colors_alphabetical <- c(  
2   `autumn` = "#a26e7c",  
3   `spring` = "#1ec99b",  
4   `summer` = "#F7B01B",  
5   `winter` = "#3c89d9"  
6 )  
7  
8 ggplot(  
9   bikes,  
10  aes(x = date, y = count,  
11    color = season)  
12 ) +  
13 geom_point() +  
14 scale_color_discrete(  
15   name = "Season:",  
16   type = my_colors_alphabetical  
17 )
```



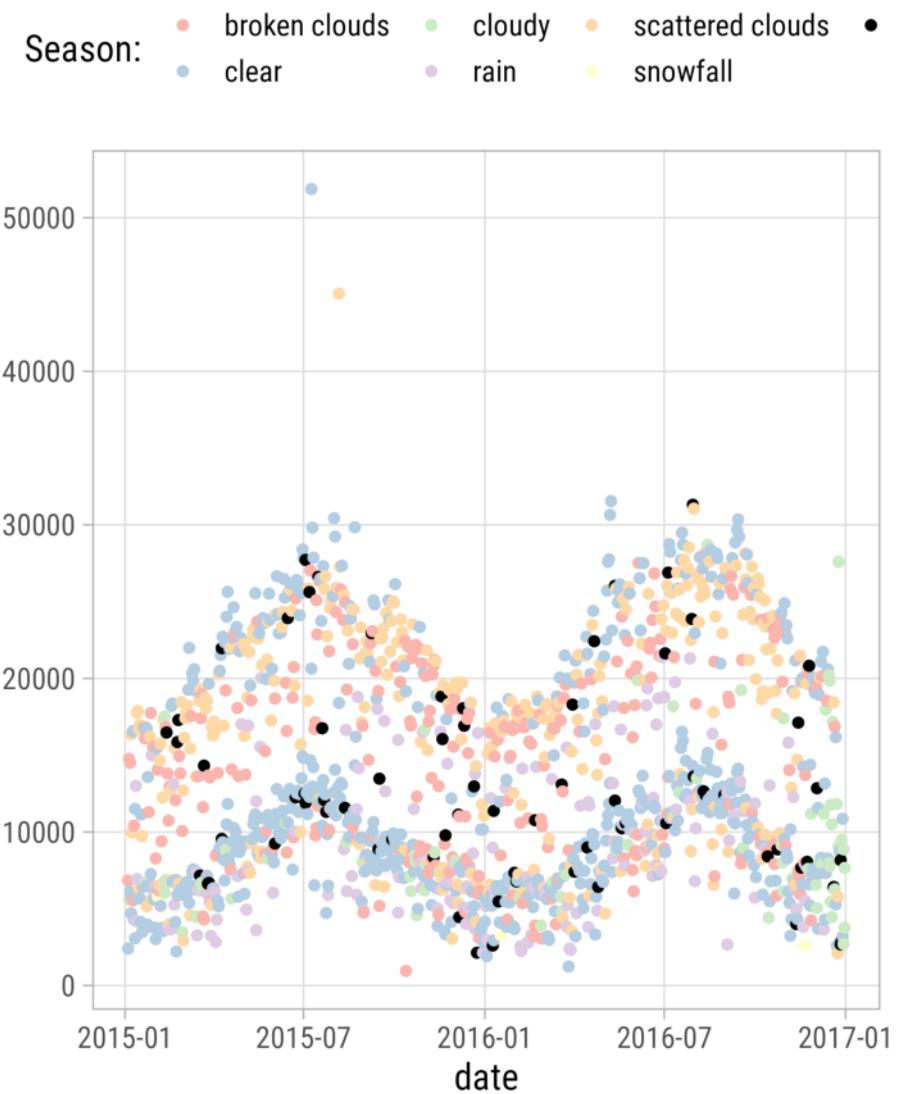
# `scale\_color|fill\_discrete`

```
1 library(RColorBrewer)
2
3 ggplot(
4   bikes,
5   aes(x = date, y = count,
6       color = season)
7 ) +
8 geom_point() +
9 scale_color_discrete(
10   name = "Season:",
11   type = brewer.pal(
12     n = 4, name = "Dark2"
13 )
14 )
```



# `scale\_color|fill\_manual`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = weather_type)  
5 ) +  
6 geom_point() +  
7 scale_color_manual(  
8   name = "Season:",  
9   values = brewer.pal(n = 6, name = "Pastel1"),  
10  na.value = "black"  
11 )
```

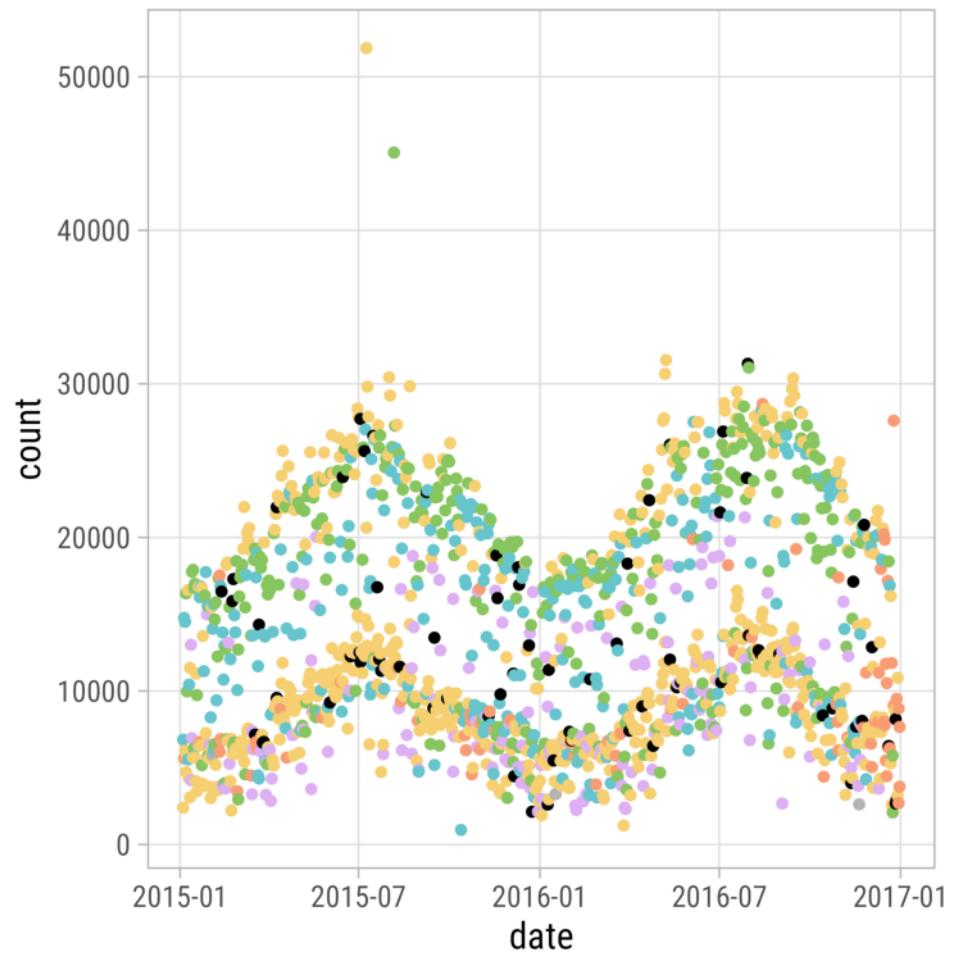


# `scale\_color|fill\_carto\_d`

```
1 ggplot(  
2   bikes,  
3   aes(x = date, y = count,  
4       color = weather_type)  
5 ) +  
6   geom_point() +  
7   rcartocolor::scale_color_carto_d(  
8     name = "Season:",  
9     palette = "Pastel",  
10    na.value = "black"  
11 )
```

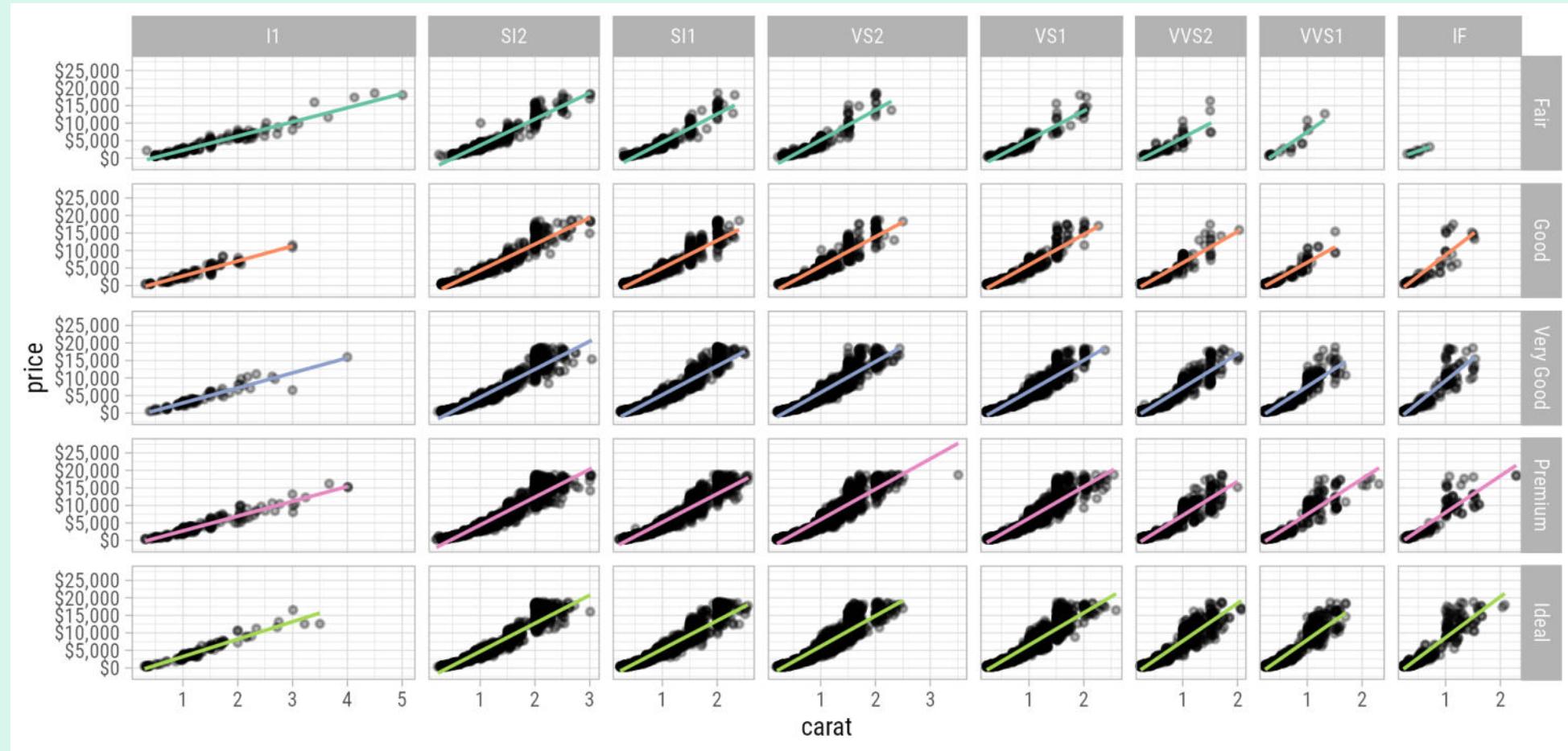
Season:

- broken clouds
- cloudy
- scattered clouds
- clear
- rain
- snowfall



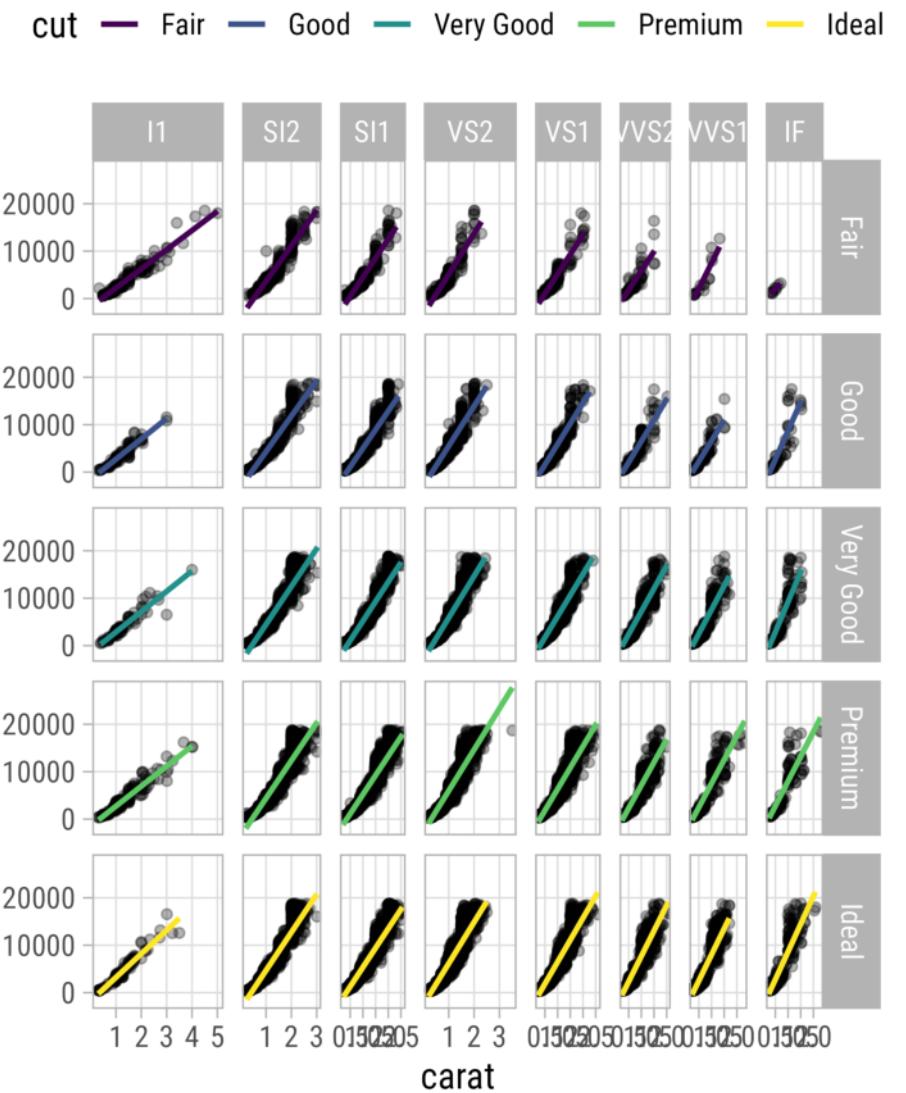
# Your Turn!

Modify our diamonds facet like this:



# Diamonds Facet

```
1 facet <-  
2   ggplot(  
3     diamonds,  
4     aes(x = carat, y = price)  
5   ) +  
6   geom_point(  
7     alpha = .3  
8   ) +  
9   geom_smooth(  
10    aes(color = cut),  
11    method = "lm",  
12    se = FALSE  
13  ) +  
14  facet_grid(  
15    cut ~ clarity,  
16    space = "free_x",  
17    scales = "free_x"  
18  )  
19 facet
```



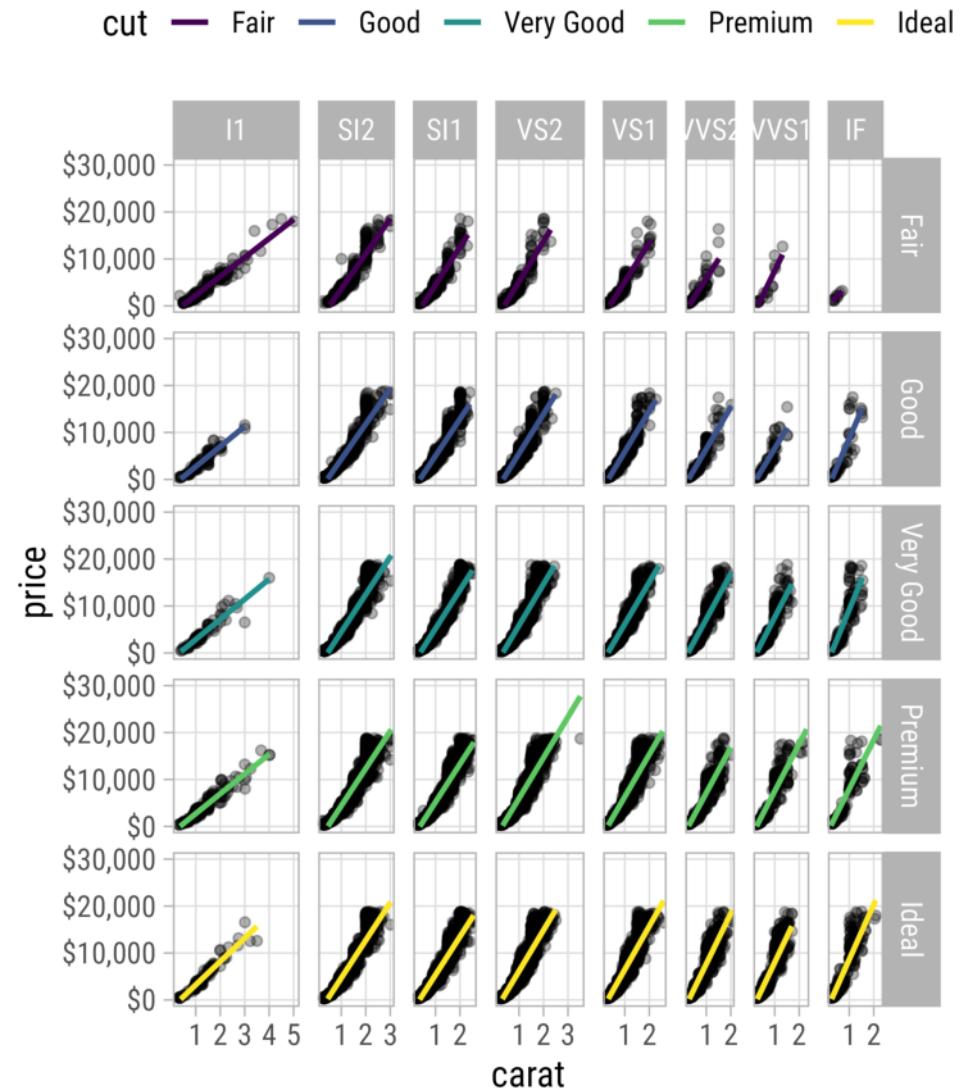
# Diamonds Facet

```
1 facet +
2   scale_x_continuous(
3     breaks = 0:5
4   ) +
5   scale_y_continuous(
6     limits = c(0, 30000),
7     breaks = 0:3*10000,
8     labels = c("$0", "$10,000", "$20,000", "$30,000")
9   )
```



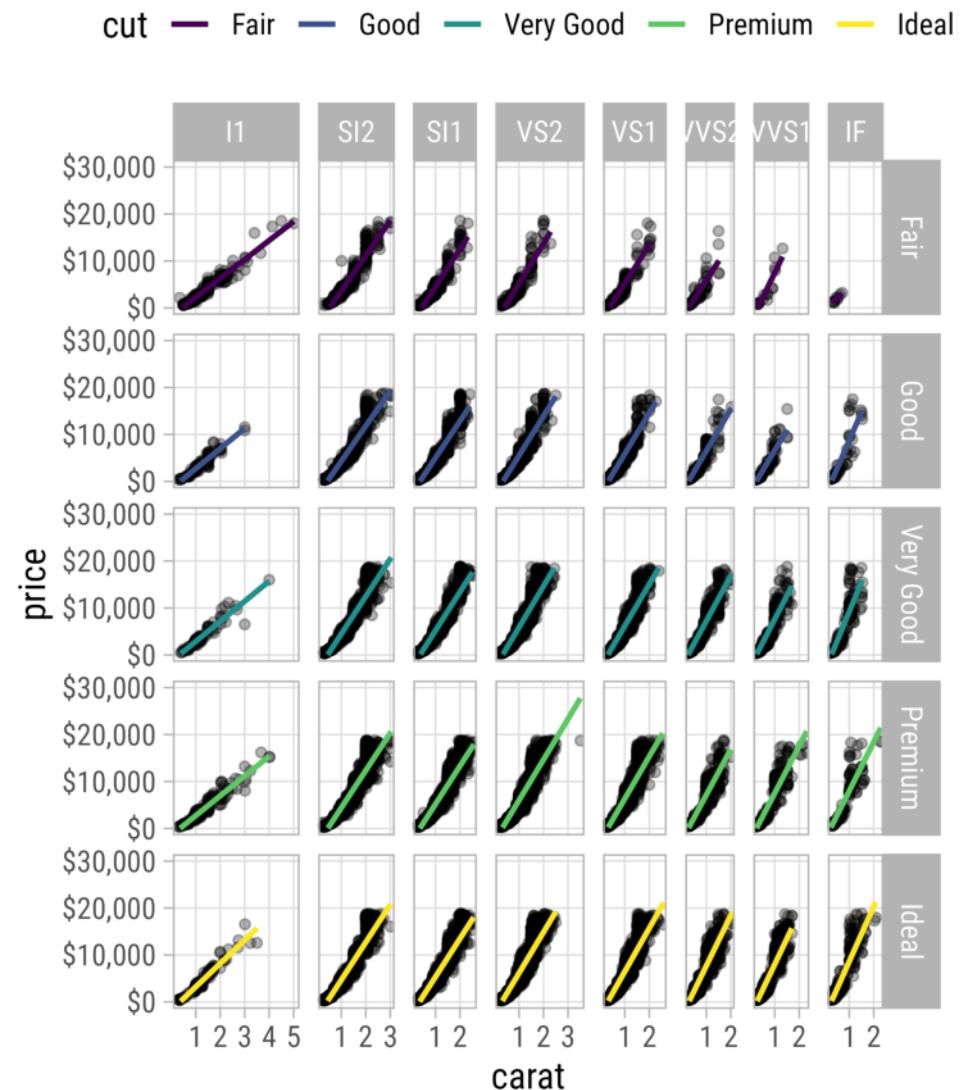
# Diamonds Facet

```
1 facet +
2   scale_x_continuous(
3     breaks = 0:5
4   ) +
5   scale_y_continuous(
6     limits = c(0, 30000),
7     breaks = 0:3*10000,
8     labels = paste0(
9       "$", format(0:3*10000, big.mark = ",",
10      )
11    )
```



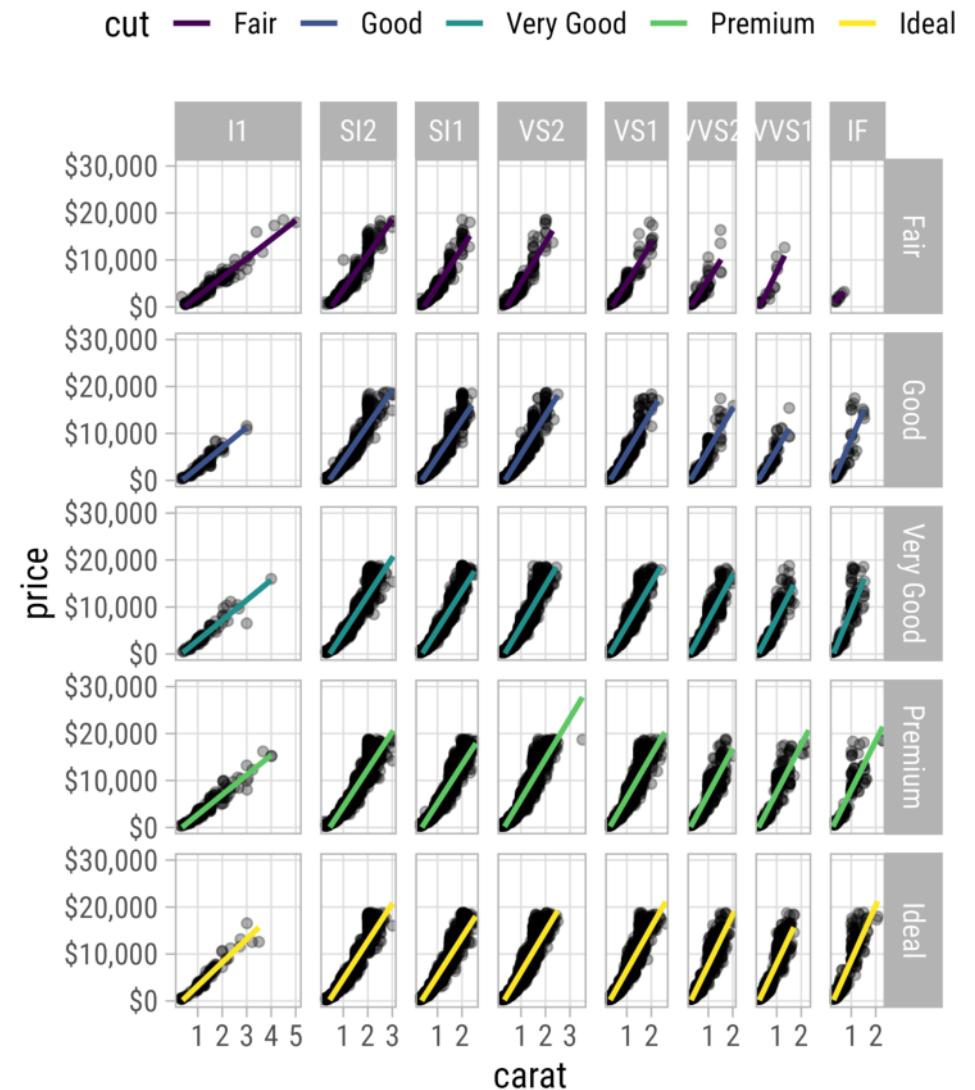
# Diamonds Facet

```
1 facet +
2   scale_x_continuous(
3     breaks = 0:5
4   ) +
5   scale_y_continuous(
6     limits = c(0, 30000),
7     breaks = 0:3*10000,
8     labels = function(y) paste0(
9       "$", format(y, big.mark=",", trim = TRUE
10      )
11    )
```



# Diamonds Facet

```
1 facet +
  scale_x_continuous(
    breaks = 0:5
) +
  scale_y_continuous(
    limits = c(0, 30000),
    breaks = 0:3*10000,
    labels = scales::dollar_format()
)
```



# Diamonds Facet

```
1 facet +
  scale_x_continuous(
    breaks = 0:5
) +
  scale_y_continuous(
    limits = c(0, 30000),
    breaks = 0:3*10000,
    labels = scales::dollar_format()
) +
  scale_color_brewer(
    palette = "Set2",
    guide = "none"
)
```



# Diamonds Facet

```
1 facet +
  scale_x_continuous(
    breaks = 0:5
  ) +
  scale_y_continuous(
    limits = c(0, 30000),
    breaks = 0:3*10000,
    labels = scales::dollar_format()
  ) +
  scale_color_brewer(
    palette = "Set2"
  ) +
  theme(
    legend.position = "none"
  )
```



# Coordinate Systems

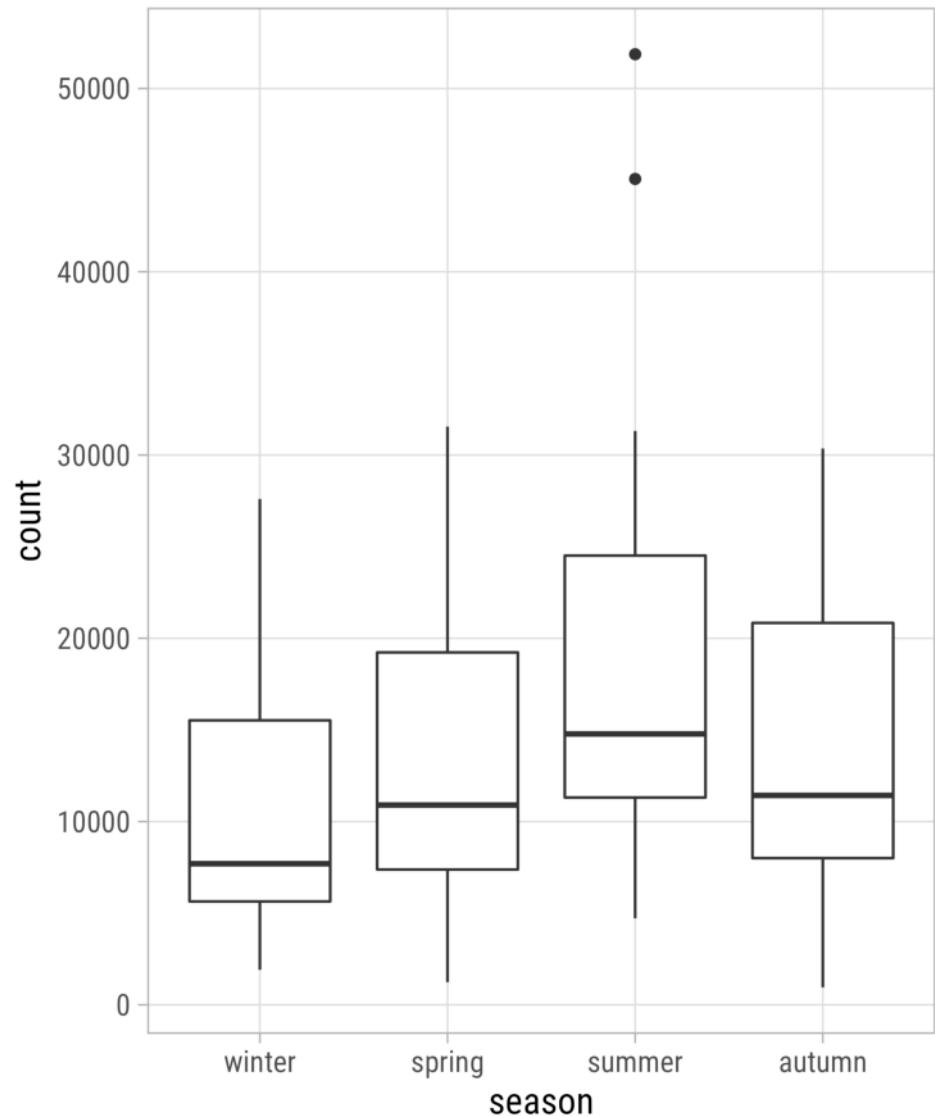
# Coordinate Systems

= interpret the position aesthetics

- **linear coordinate systems:** preserve the geometrical shapes
  - `coord_cartesian()`
  - `coord_fixed()`
  - `coord_flip()`
- **non-linear coordinate systems:** likely change the geometrical shapes
  - `coord_polar()`
  - `coord_map()` and `coord_sf()`
  - `coord_trans()`

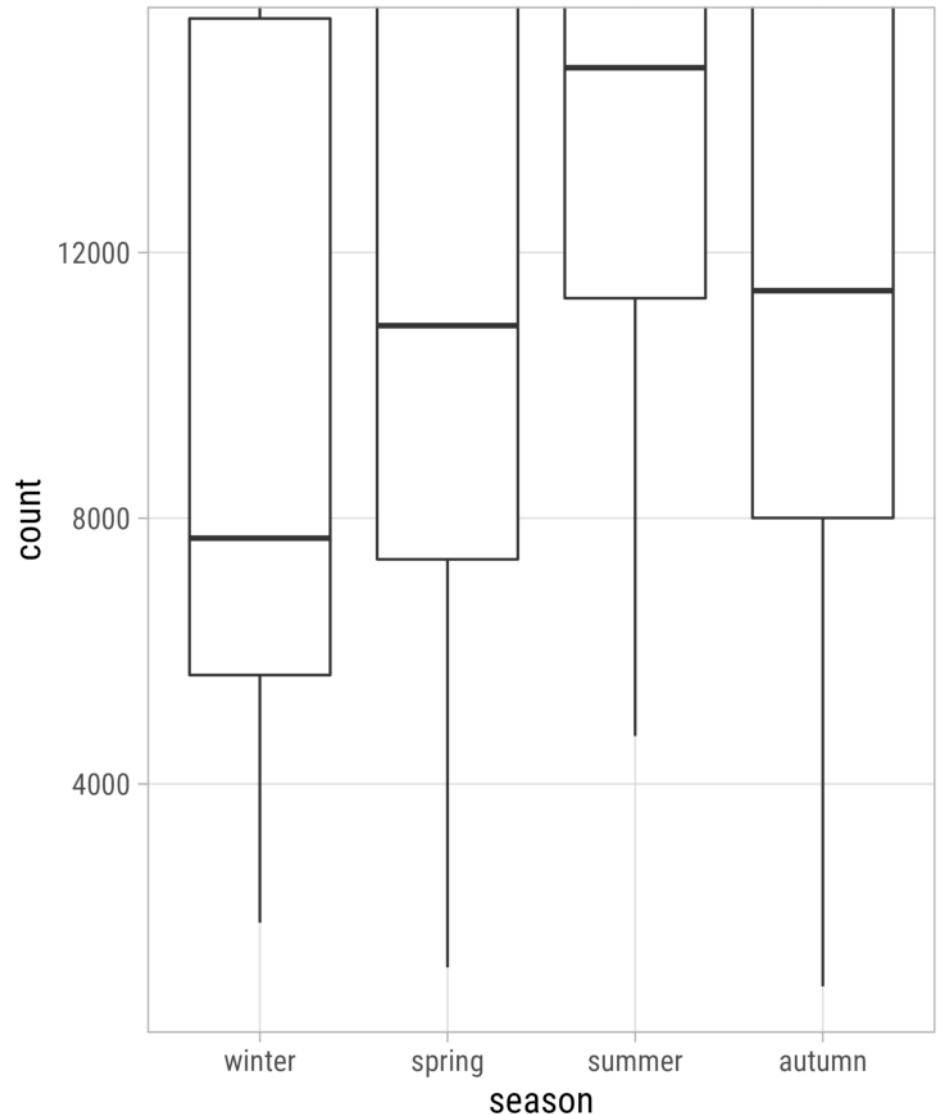
# Cartesian Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 coord_cartesian()
```



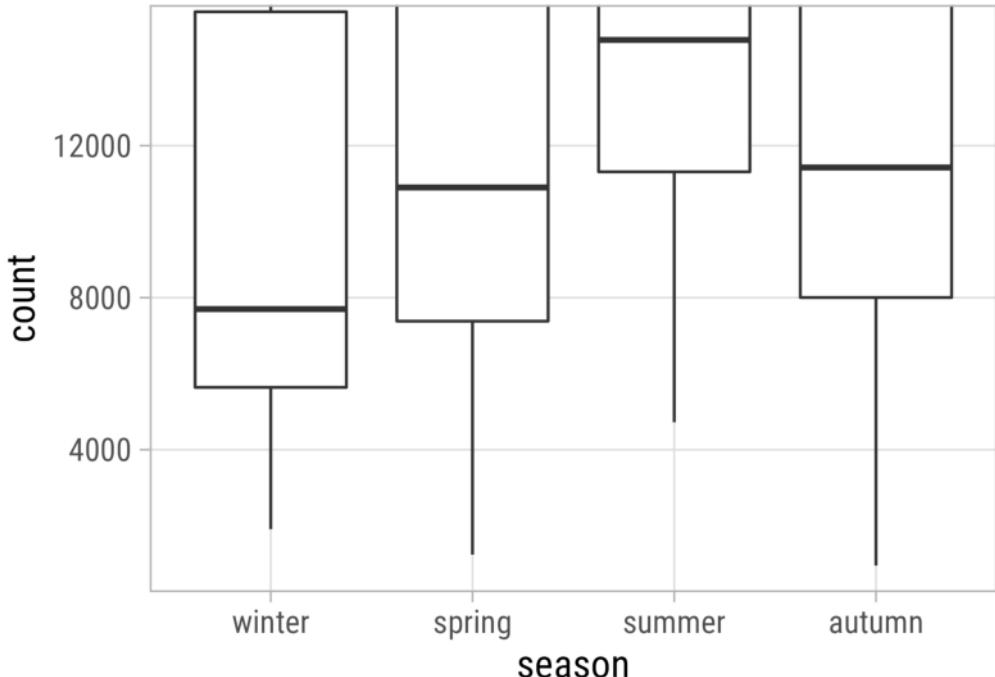
# Cartesian Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count)  
4 ) +  
5 geom_boxplot() +  
6 coord_cartesian(  
7   ylim = c(NA, 15000)  
8 )
```

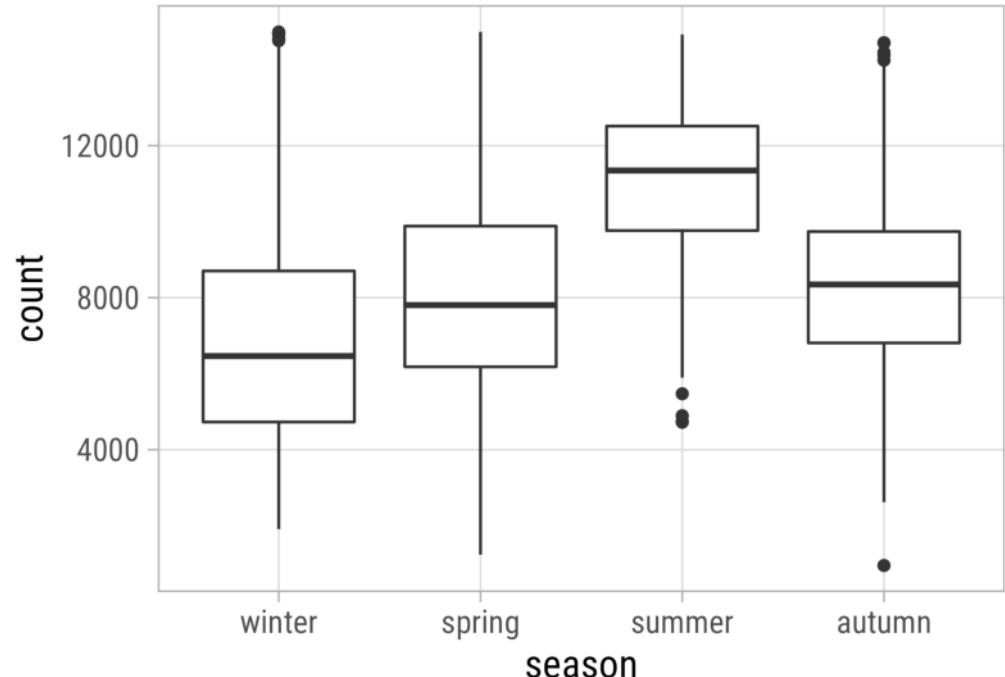


# Changing Limits

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 coord_cartesian(  
7   ylim = c(NA, 15000))  
8 )
```

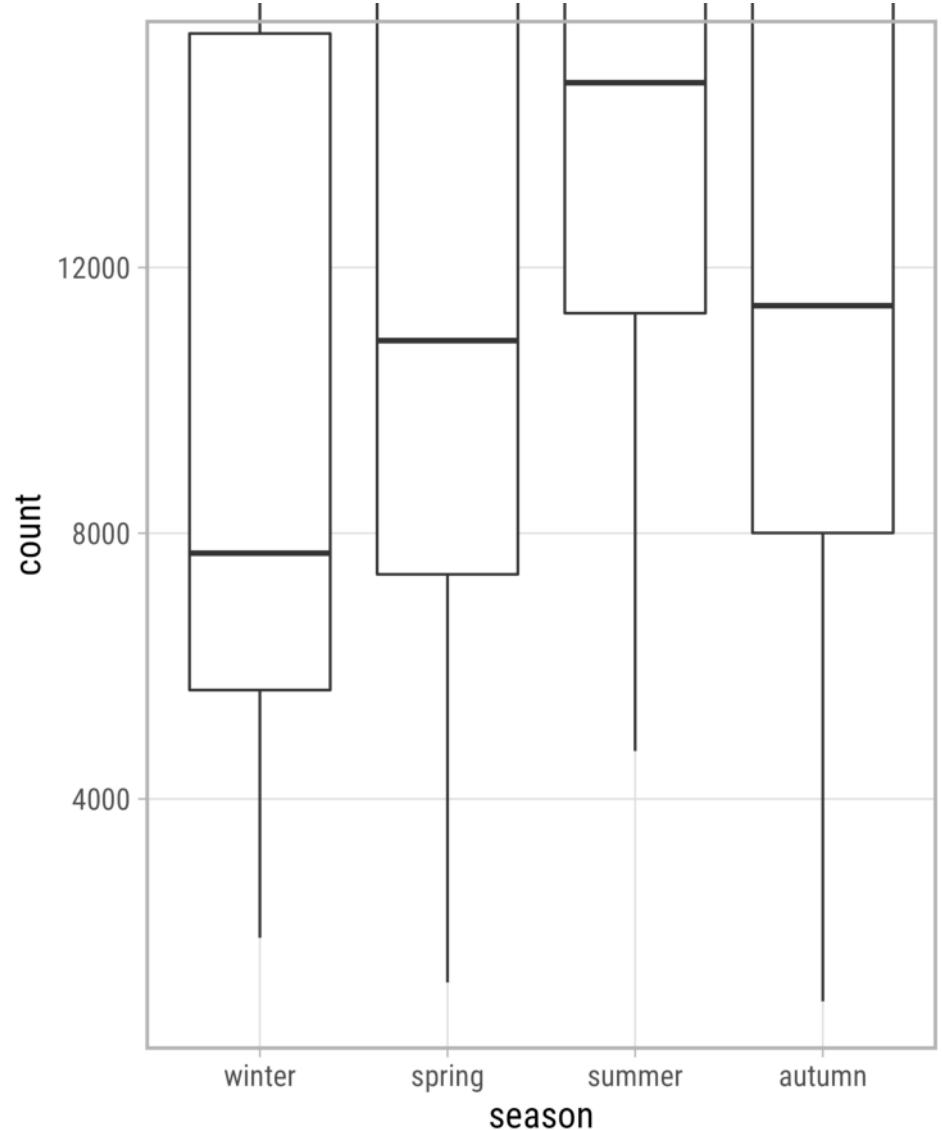


```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 scale_y_continuous(  
7   limits = c(NA, 15000))  
8 )
```



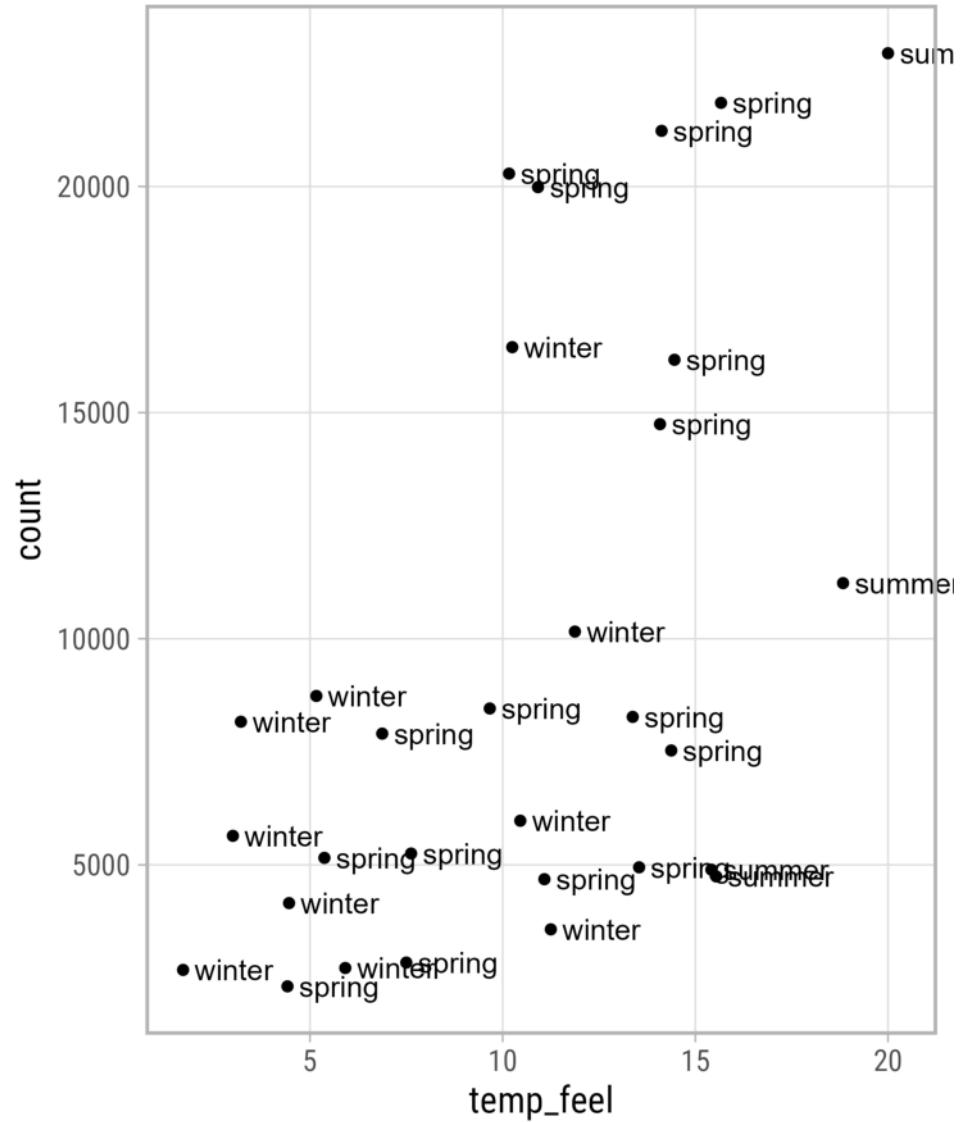
# Clipping

```
1 ggplot(  
2   bikes,  
3   aes(x = season, y = count))  
4 ) +  
5 geom_boxplot() +  
6 coord_cartesian(  
7   ylim = c(NA, 15000),  
8   clip = "off"  
9 )
```



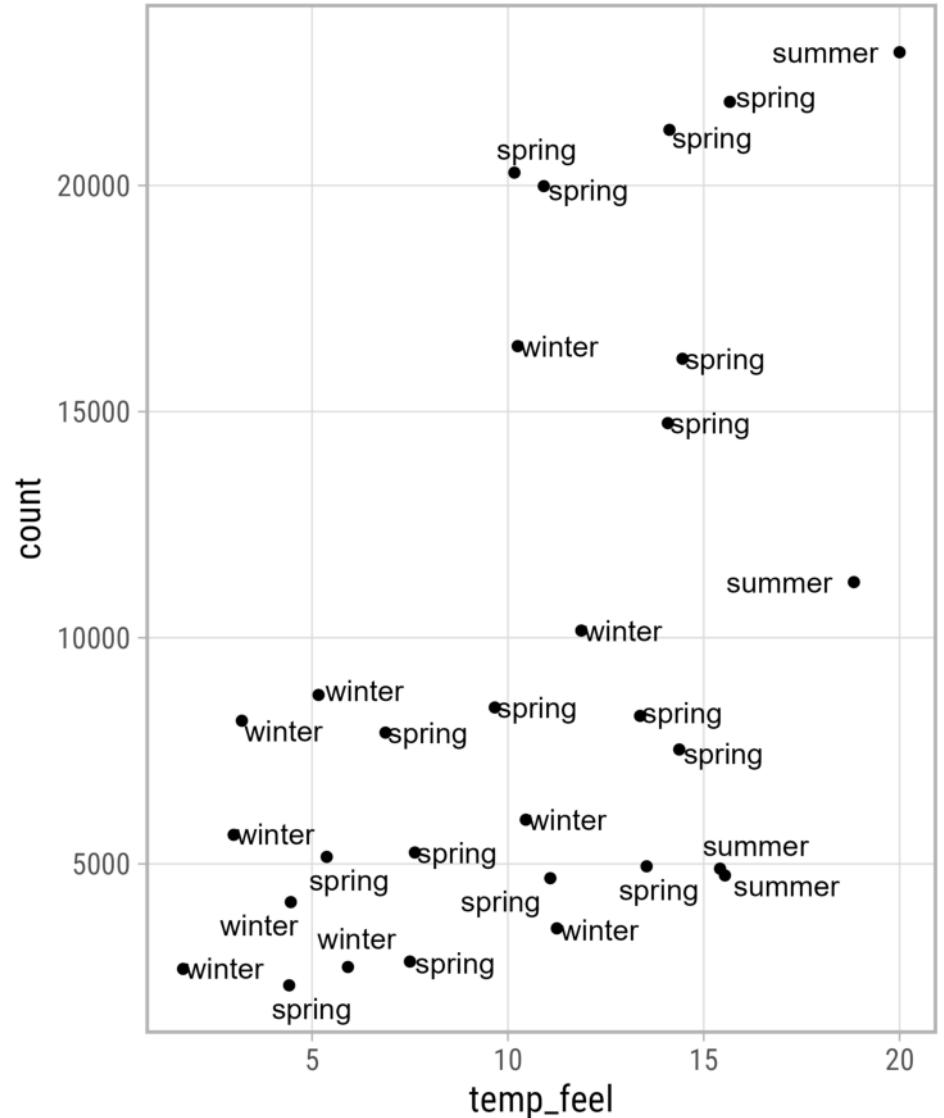
# Clipping

```
1 ggplot(  
2   filter(bikes, is_holiday == TRUE),  
3   aes(x = temp_feel, y = count))  
4 ) +  
5 geom_point() +  
6 geom_text(  
7   aes(label = season),  
8   nudge_x = .3,  
9   hjust = 0  
10 ) +  
11 coord_cartesian(  
12   clip = "off"  
13 )
```



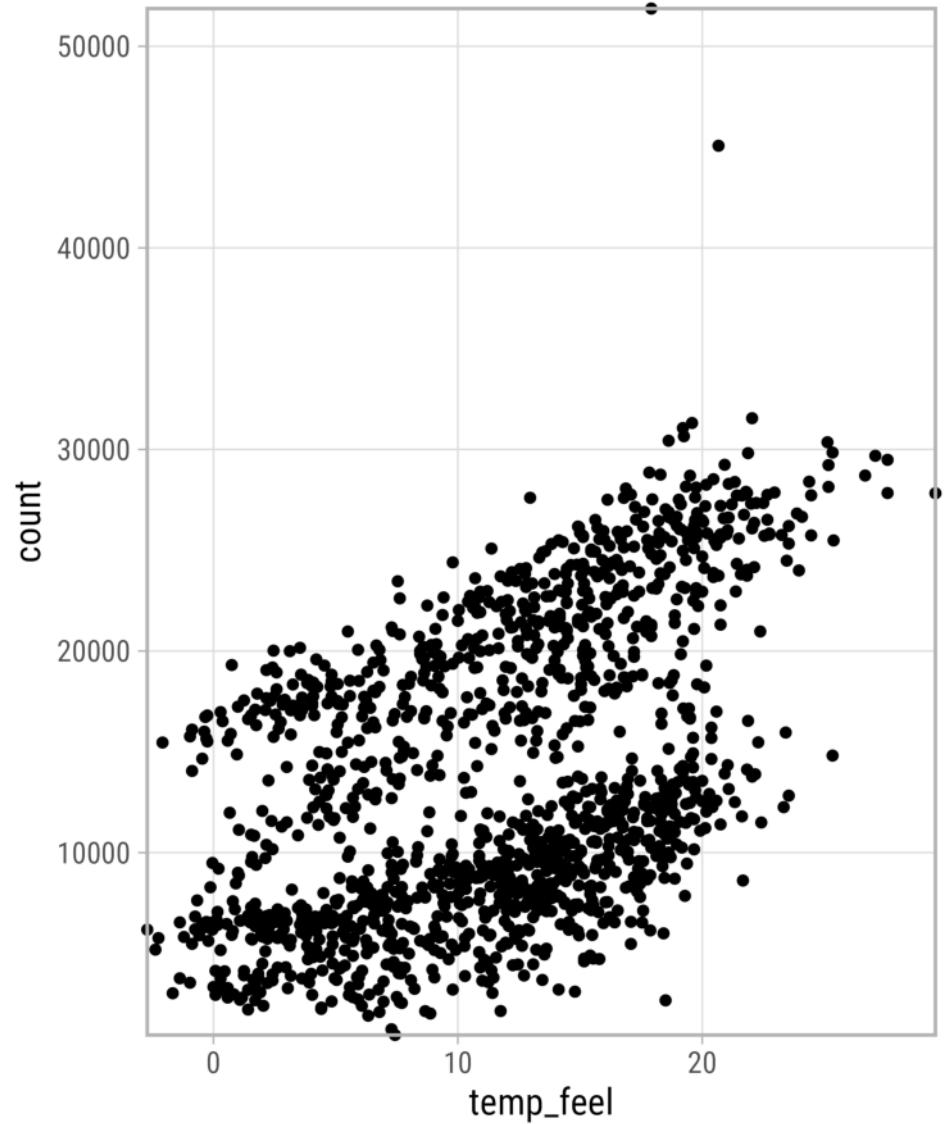
# ... or better use {ggrepel}

```
1 ggplot(  
2   filter(bikes, is_holiday == TRUE),  
3   aes(x = temp_feel, y = count))  
4 ) +  
5 geom_point() +  
6 ggrepel::geom_text_repel(  
7   aes(label = season),  
8   nudge_x = .3,  
9   hjust = 0  
10 ) +  
11 coord_cartesian(  
12 clip = "off"  
13 )
```



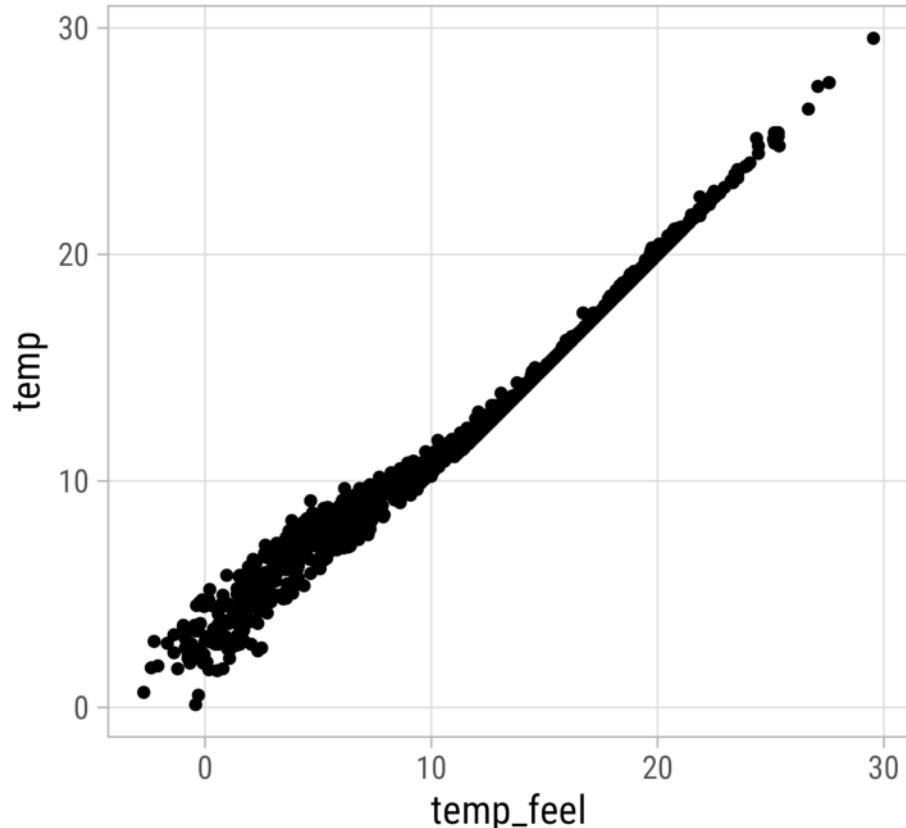
# Remove All Padding

```
1 ggplot(  
2   bikes,  
3   aes(x = temp_feel, y = count)  
4 ) +  
5   geom_point() +  
6   coord_cartesian(  
7     expand = FALSE,  
8     clip = "off"  
9 )
```

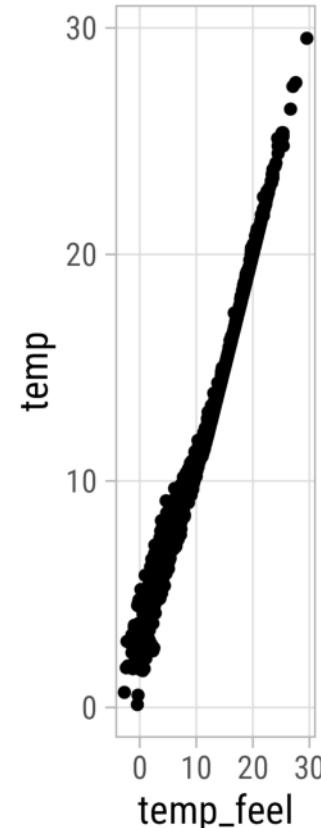


# Fixed Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = temp_feel, y = temp)  
4 ) +  
5   geom_point() +  
6   coord_fixed()
```

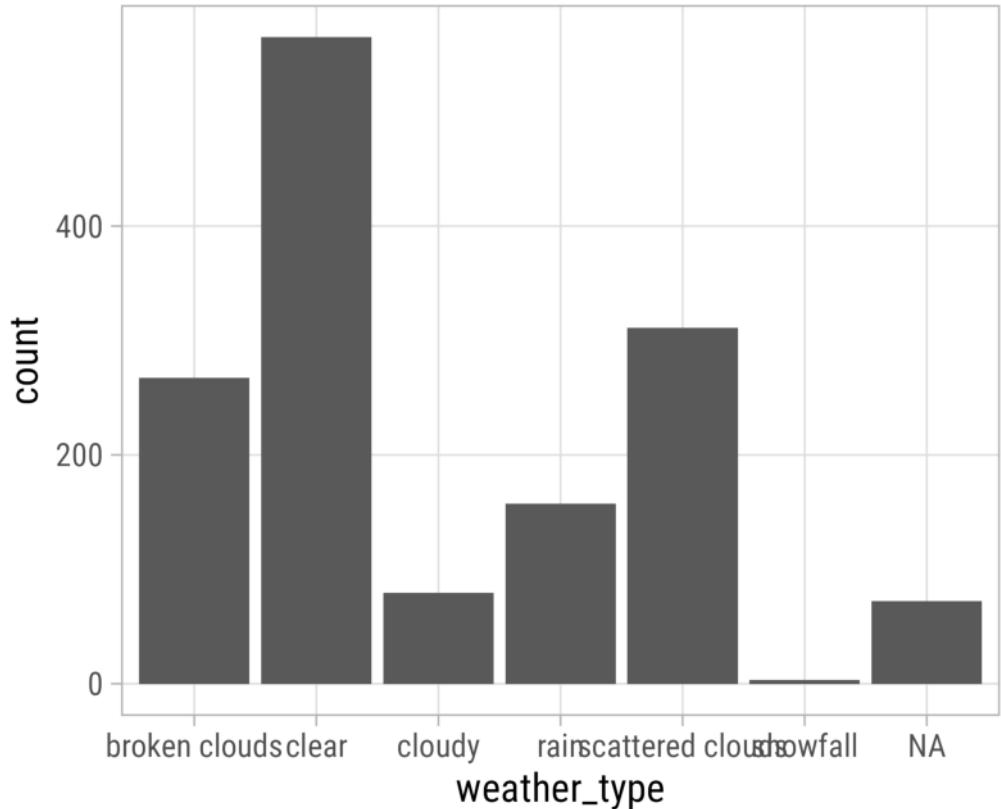


```
1 ggplot(  
2   bikes,  
3   aes(x = temp_feel, y = temp)  
4 ) +  
5   geom_point() +  
6   coord_fixed(ratio = 4)
```

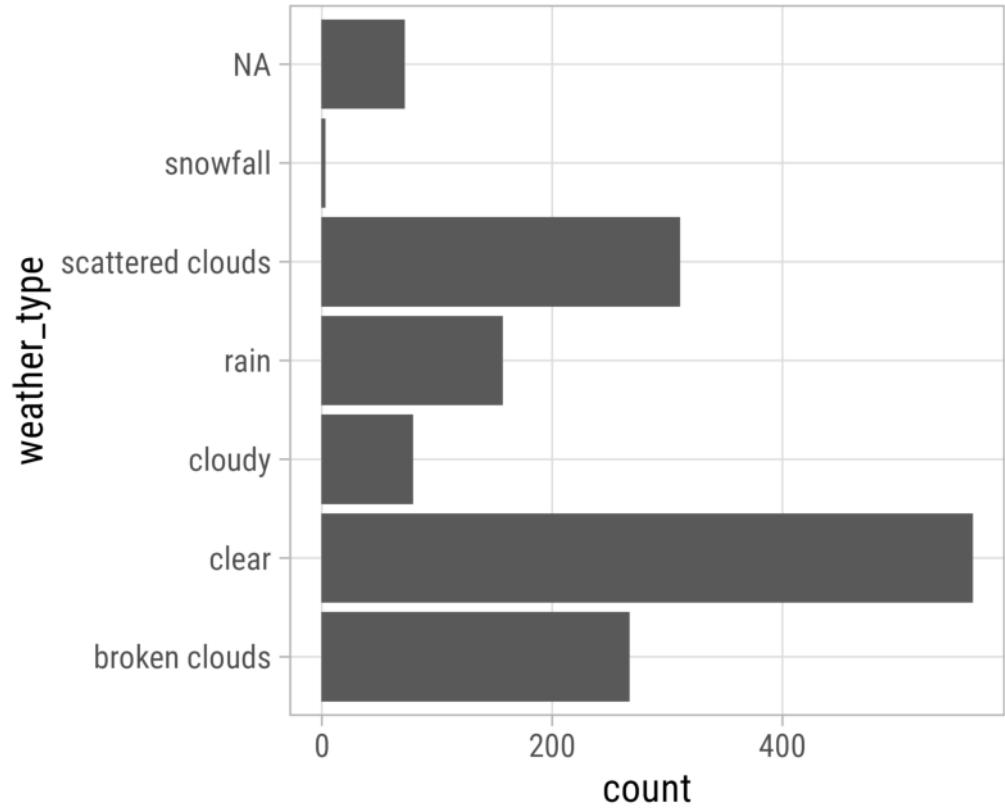


# Flipped Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = weather_type)  
4 ) +  
5 geom_bar() +  
6 coord_cartesian()
```

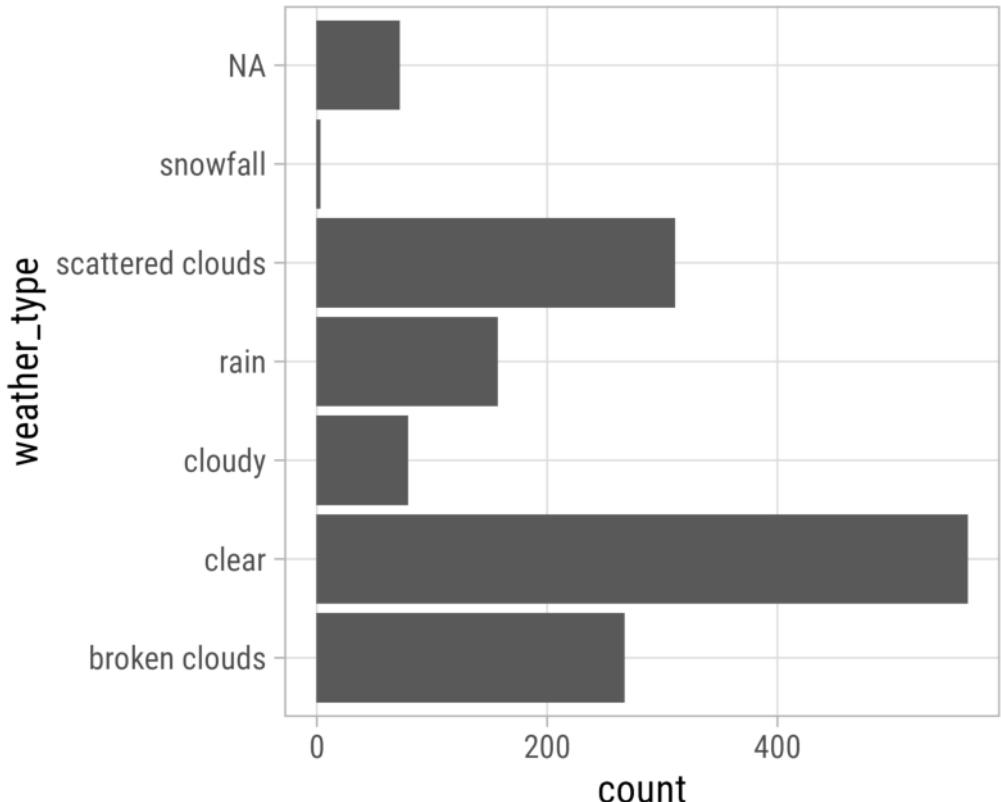


```
1 ggplot(  
2   bikes,  
3   aes(x = weather_type)  
4 ) +  
5 geom_bar() +  
6 coord_flip()
```

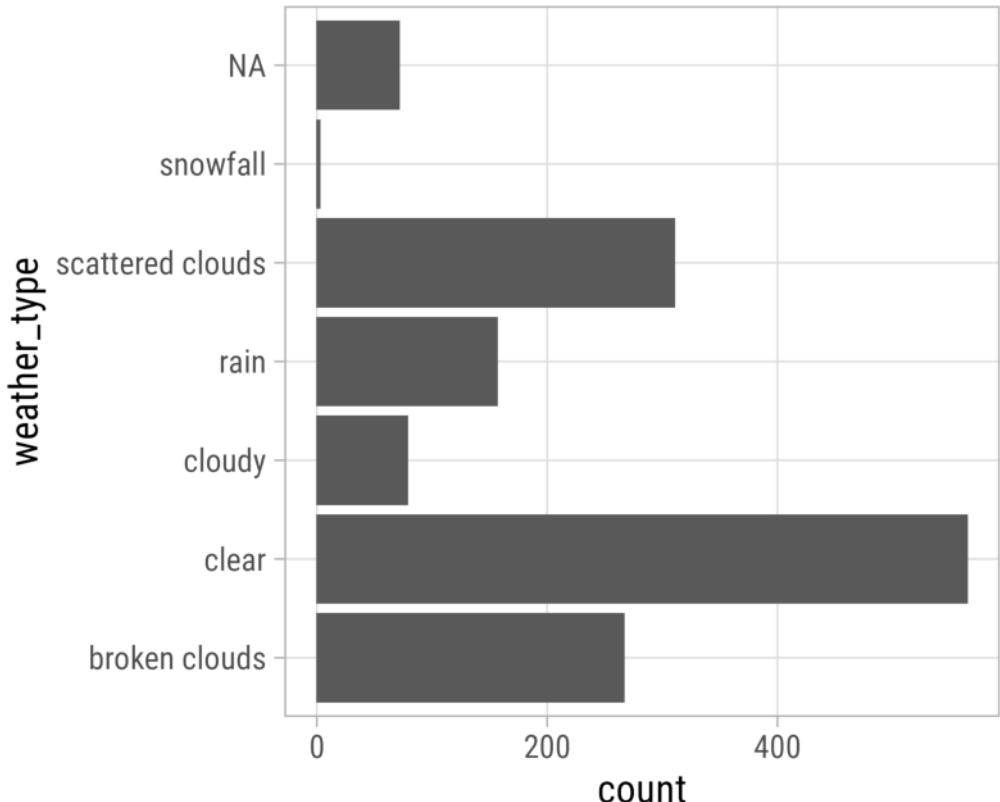


# Flipped Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(y = weather_type)  
4 ) +  
5 geom_bar() +  
6 coord_cartesian()
```

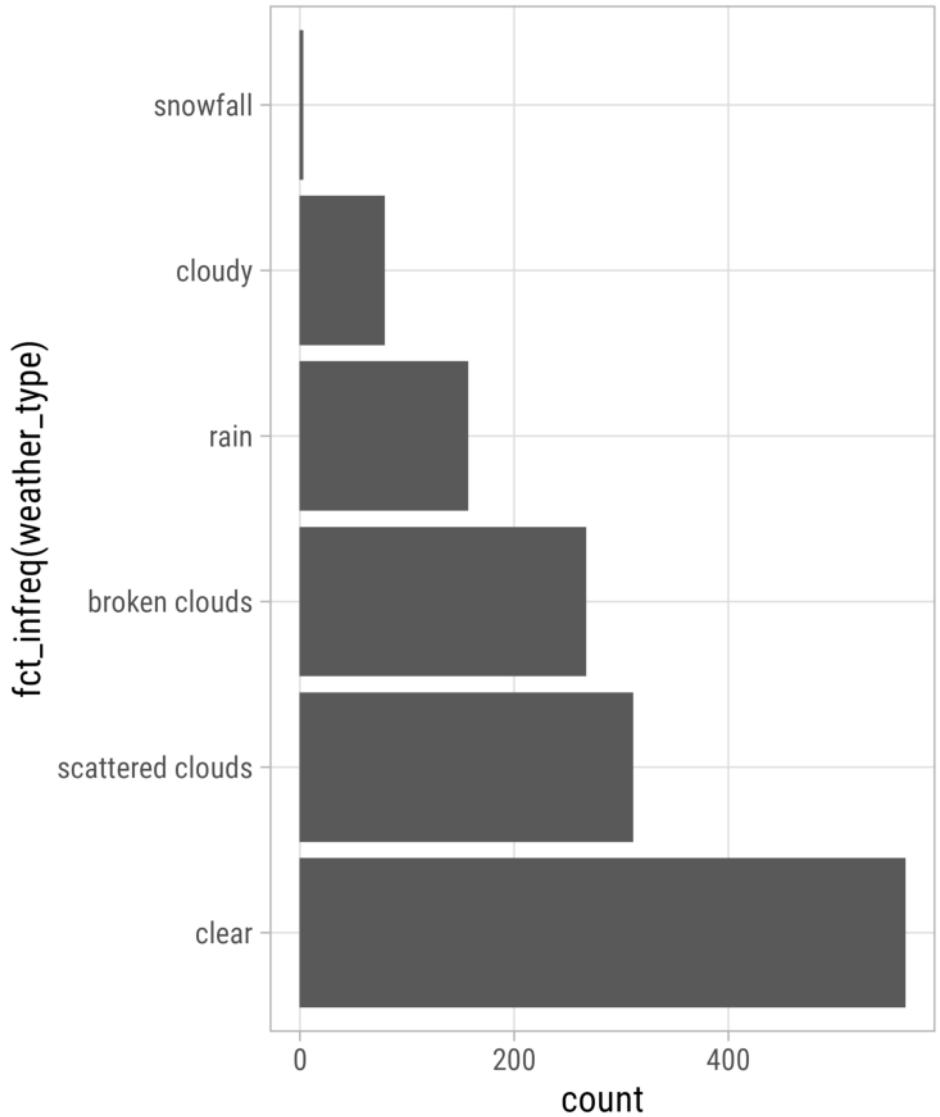


```
1 ggplot(  
2   bikes,  
3   aes(x = weather_type)  
4 ) +  
5 geom_bar() +  
6 coord_flip()
```



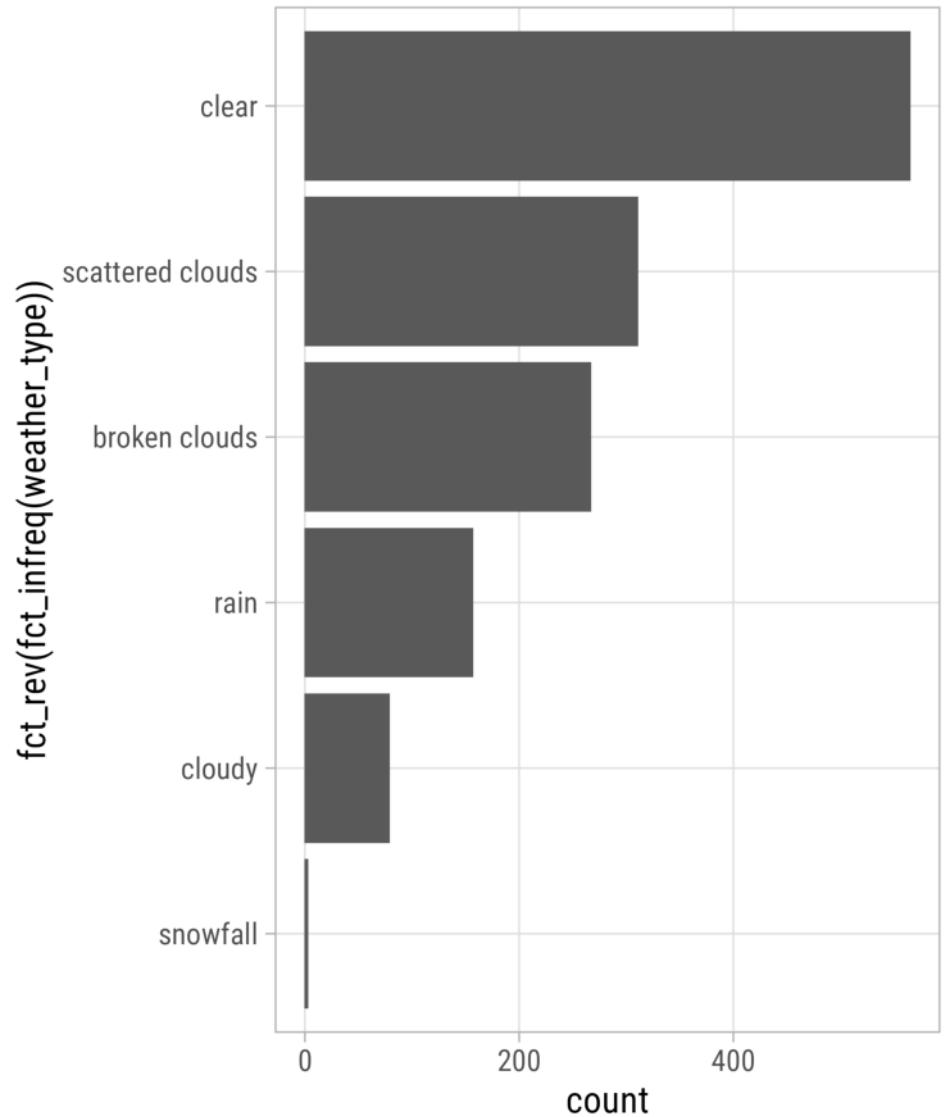
# Reminder: Sort Your Bars!

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(y = fct_infreq(weather_type))  
4 ) +  
5 geom_bar()
```



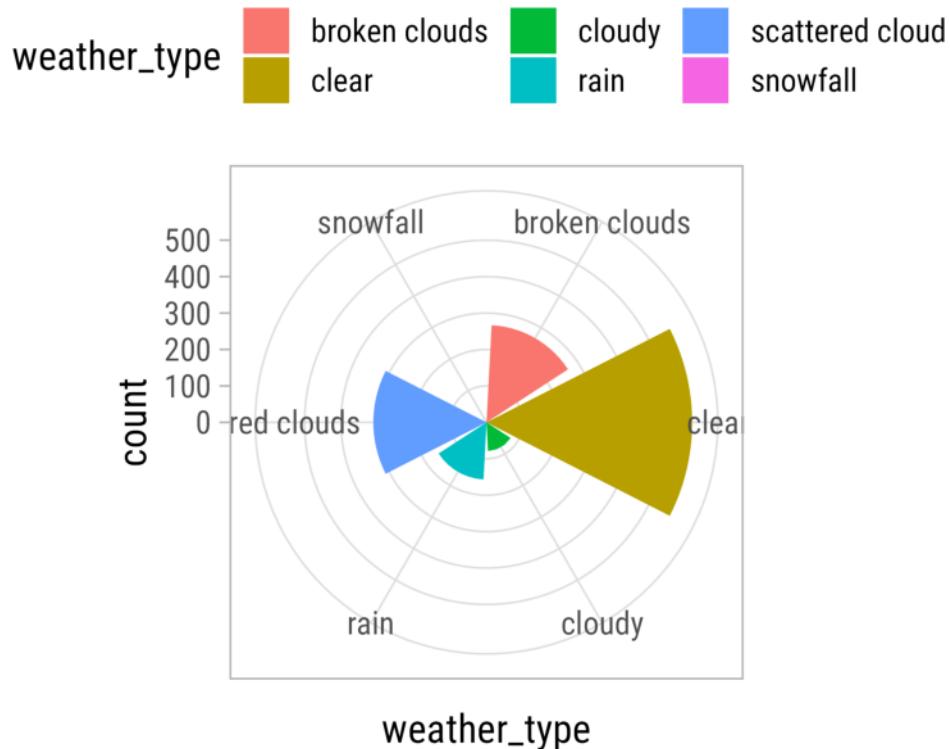
# Reminder: Sort Your Bars!

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(y = fct_rev(  
4     fct_infreq(weather_type)  
5   ))  
6 ) +  
7 geom_bar()
```

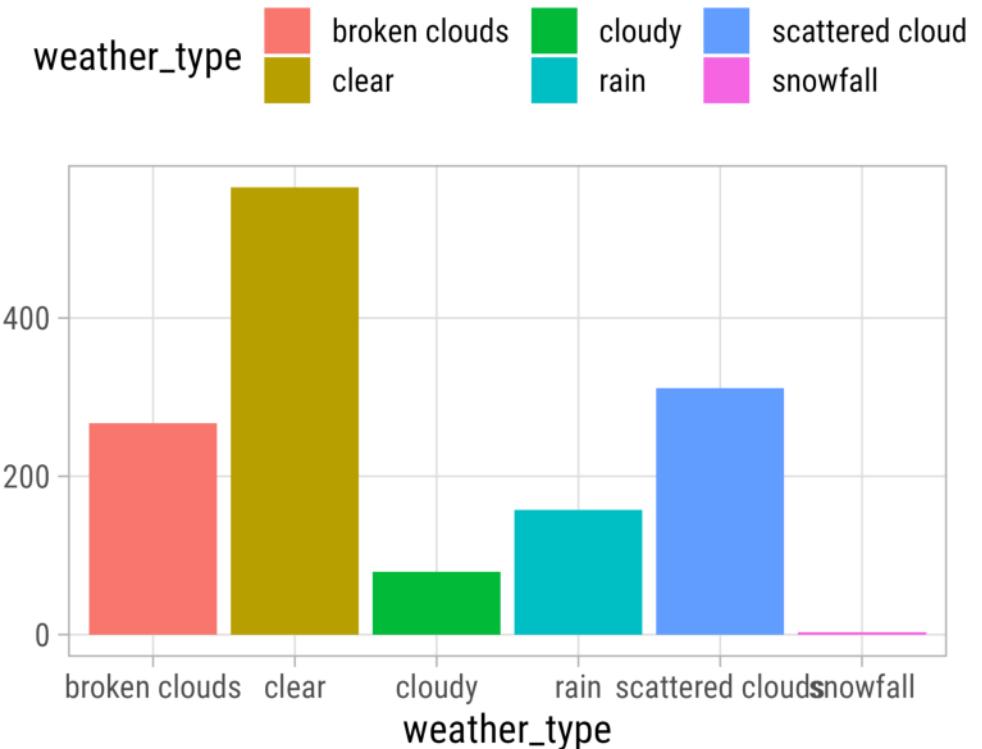


# Circular Corrdinate System

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = weather_type,  
4         fill = weather_type)  
5 ) +  
6 geom_bar() +  
7 coord_polar()
```

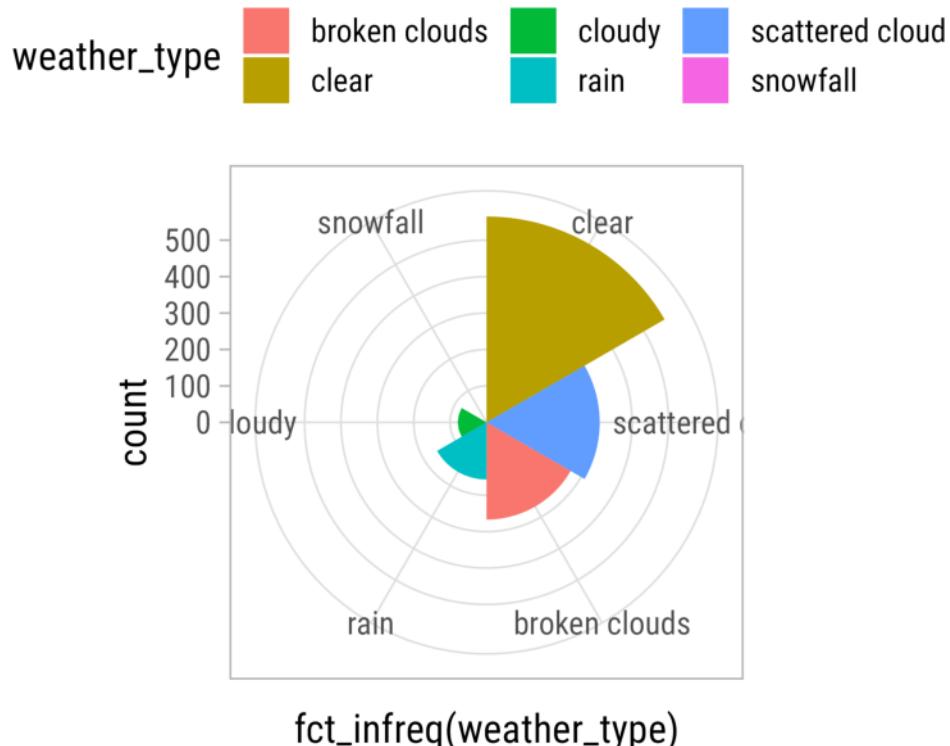


```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = weather_type,  
4         fill = weather_type)  
5 ) +  
6 geom_bar() +  
7 coord_cartesian()
```

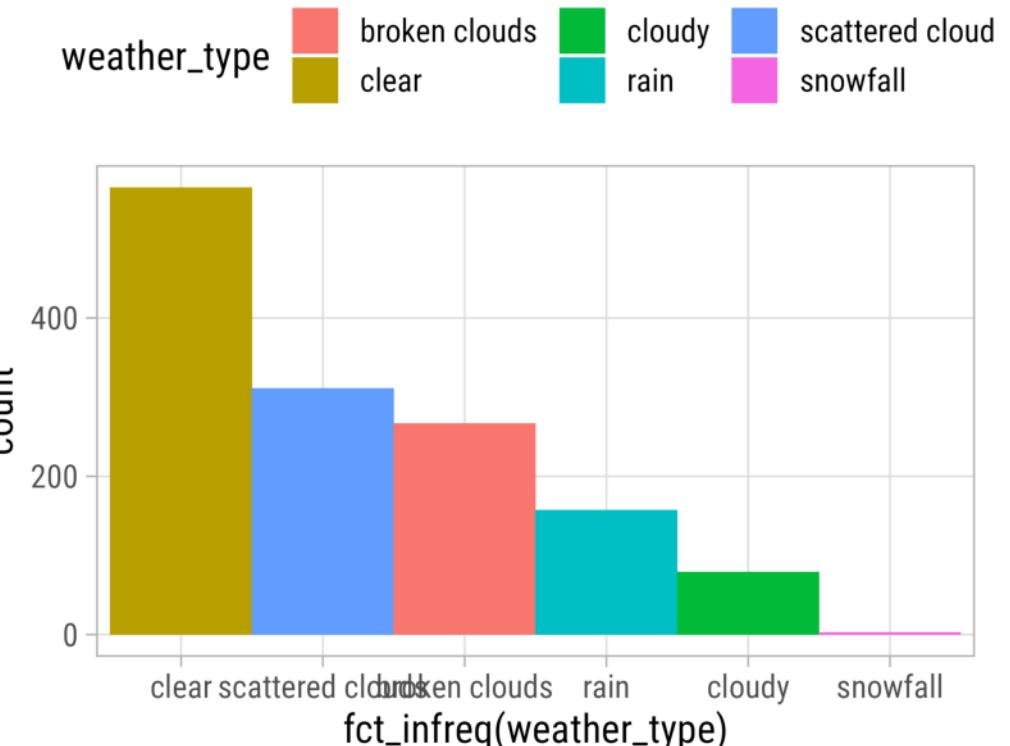


# Circular Coordinate System

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = fct_infreq(weather_type),  
4       fill = weather_type)  
5 ) +  
6 geom_bar(width = 1) +  
7 coord_polar()
```

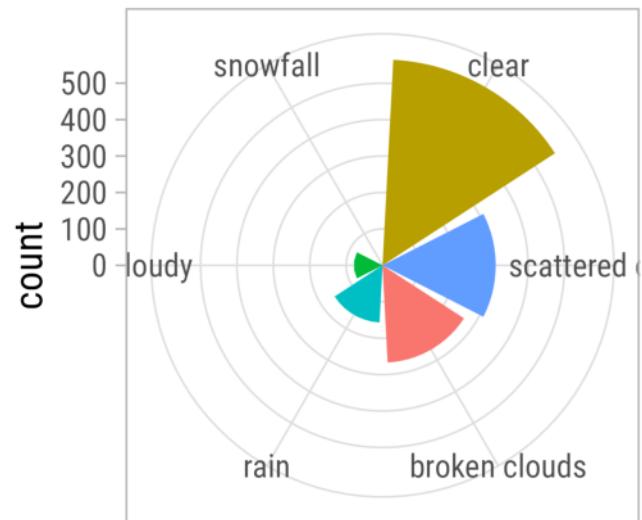


```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = fct_infreq(weather_type),  
4       fill = weather_type)  
5 ) +  
6 geom_bar(width = 1) +  
7 coord_cartesian()
```

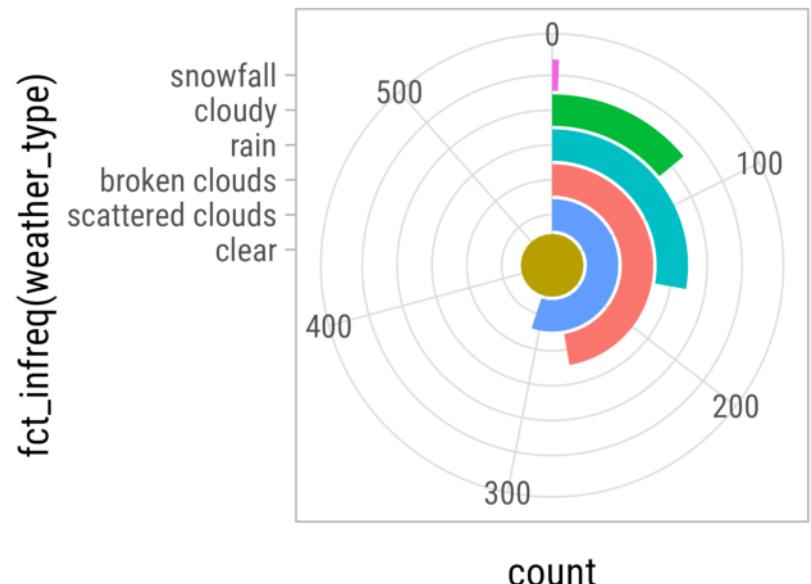
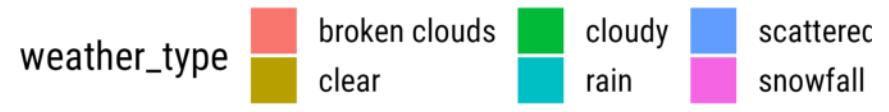


# Circular Corrdinate System

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = fct_infreq(weather_type),  
4       fill = weather_type)  
5 ) +  
6 geom_bar() +  
7 coord_polar(theta = "x")
```

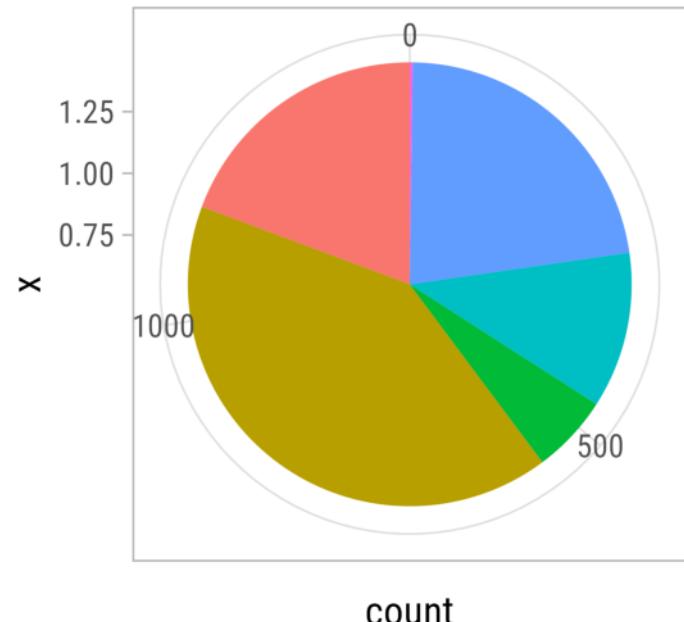


```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = fct_infreq(weather_type),  
4       fill = weather_type)  
5 ) +  
6 geom_bar() +  
7 coord_polar(theta = "y")
```

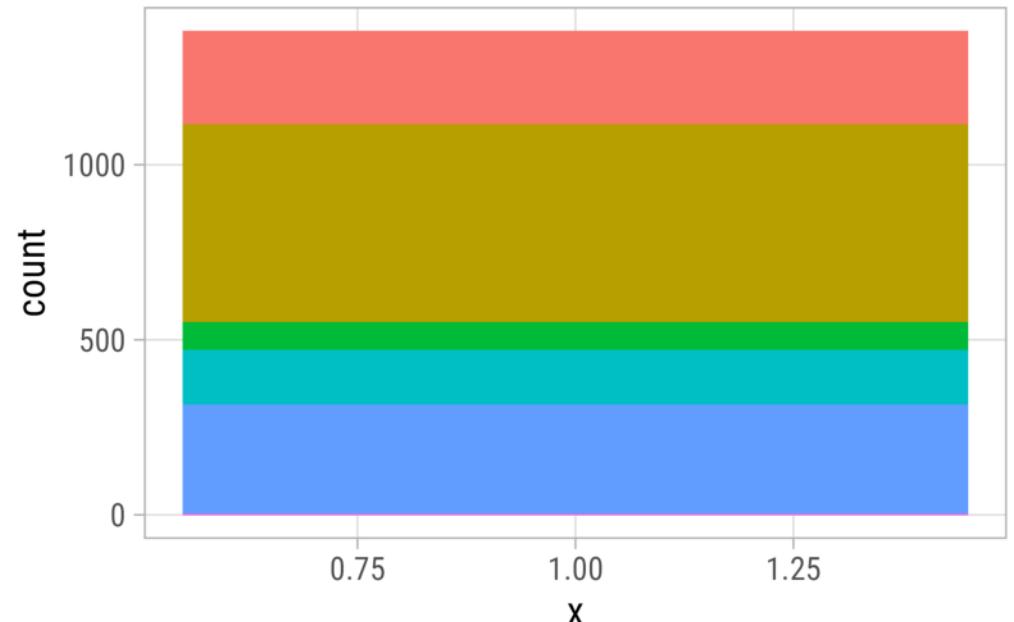


# Circular Corrdinate System

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = 1, fill = weather_type)  
4 ) +  
5 geom_bar(position = "stack") +  
6 coord_polar(theta = "y")
```

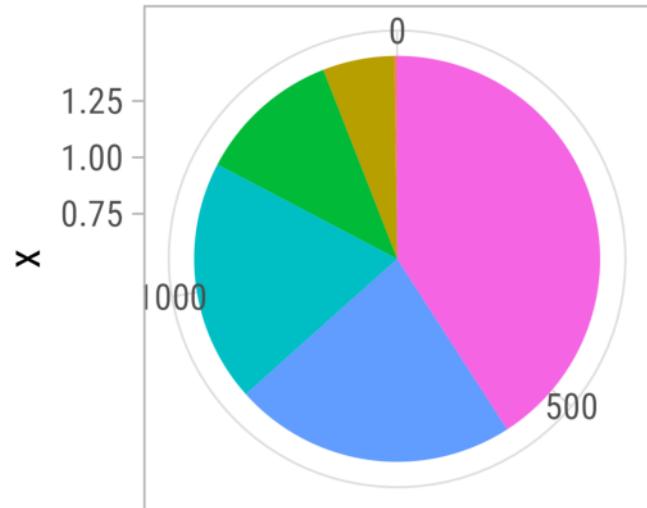
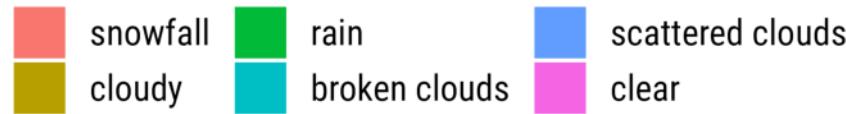


```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = 1, fill = weather_type)  
4 ) +  
5 geom_bar(position = "stack") +  
6 coord_cartesian()
```



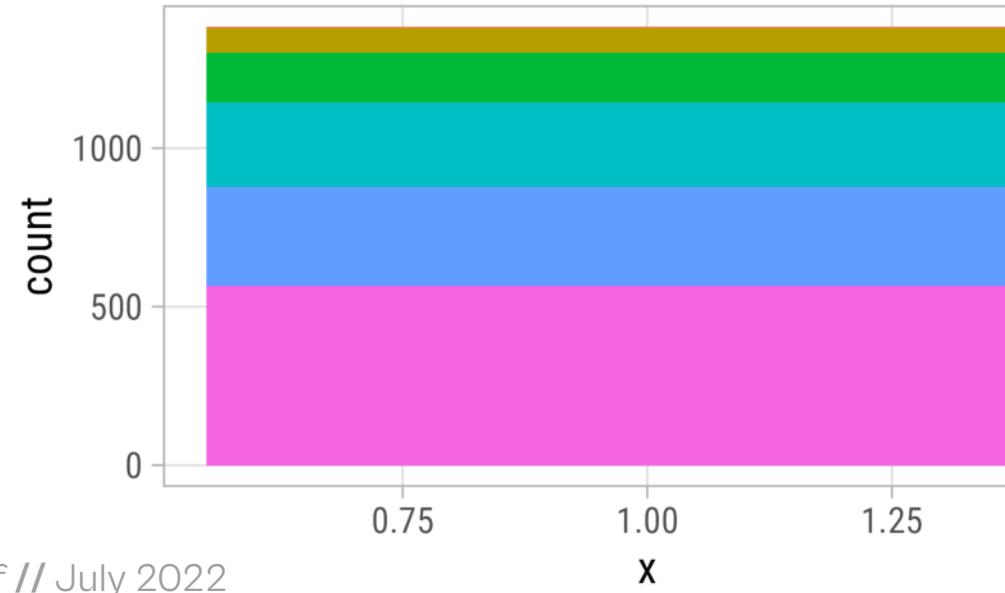
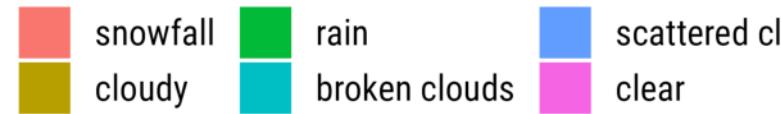
# Circular Corrdinate System

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = 1,  
4     fill = fct_rev(fct_infreq(weather_type)))  
5 ) +  
6 geom_bar(position = "stack") +  
7 coord_polar(theta = "y") +  
8 scale_fill_discrete(name = NULL)
```



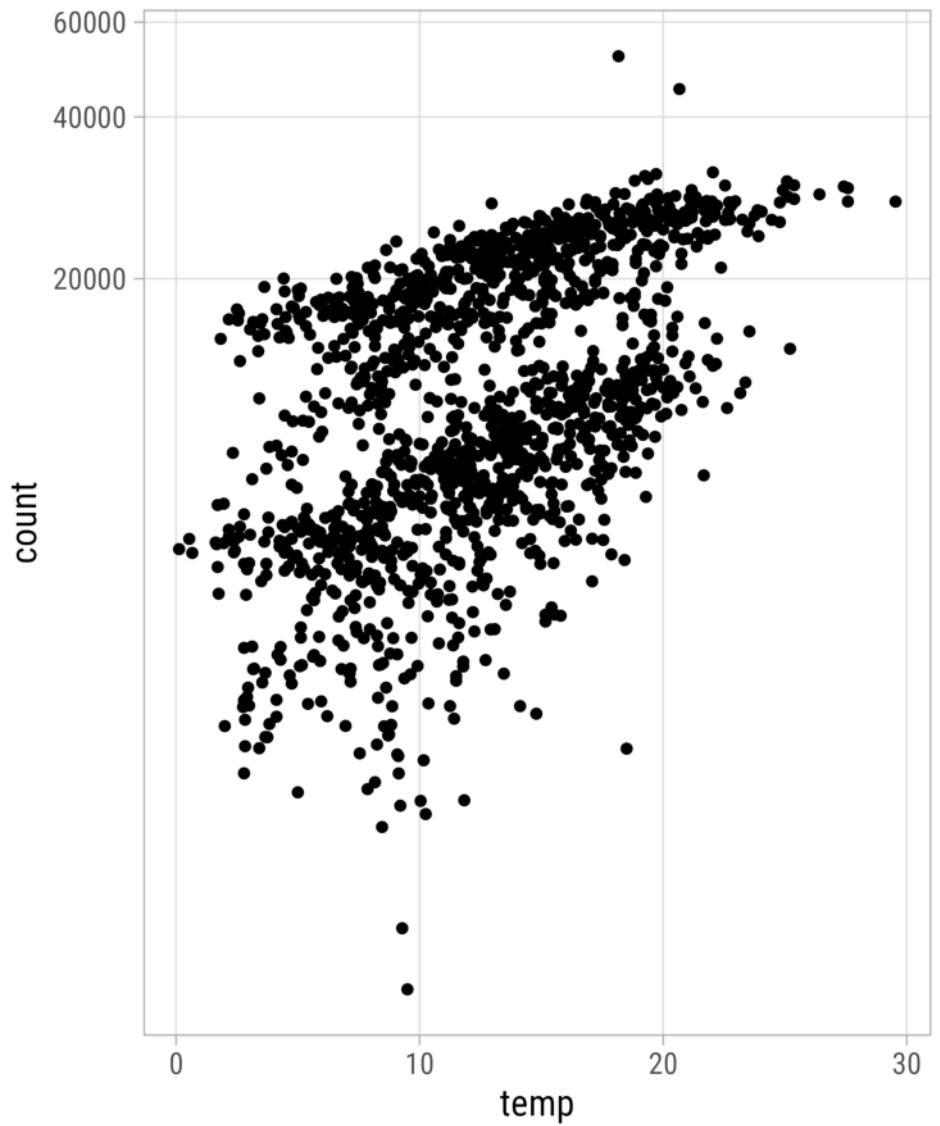
Cédric Scherer // rstudio::conf // July 2022

```
1 ggplot(  
2   filter(bikes, !is.na(weather_type)),  
3   aes(x = 1,  
4     fill = fct_rev(fct_infreq(weather_type)))  
5 ) +  
6 geom_bar(position = "stack") +  
7 coord_cartesian() +  
8 scale_fill_discrete(name = NULL)
```



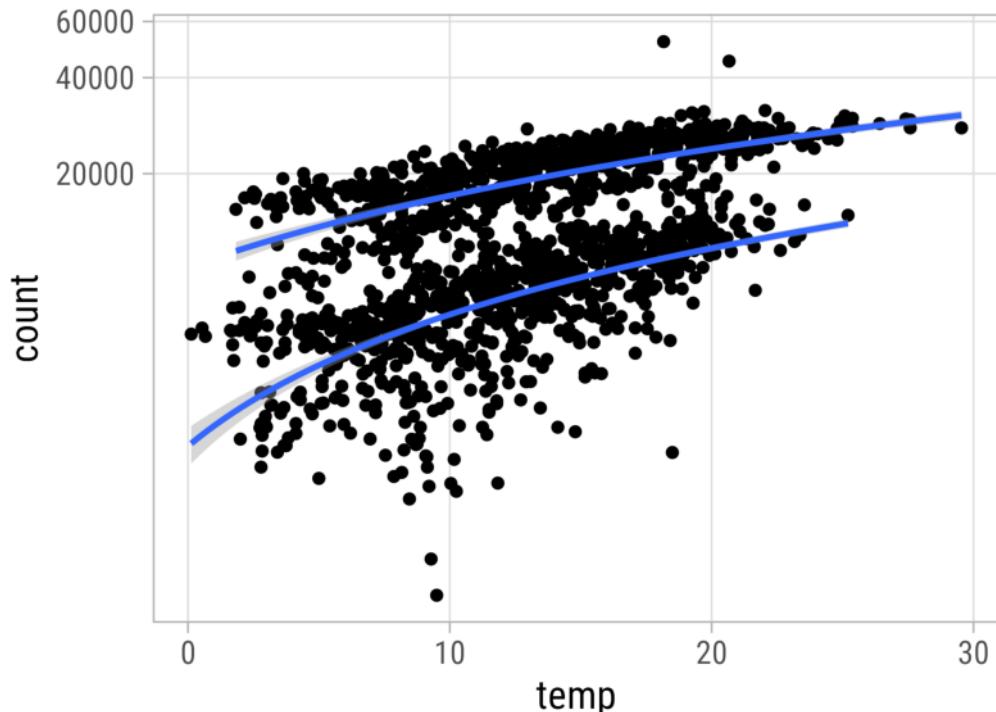
# Transform a Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = temp, y = count)  
4 ) +  
5   geom_point() +  
6   coord_trans(y = "log10")
```

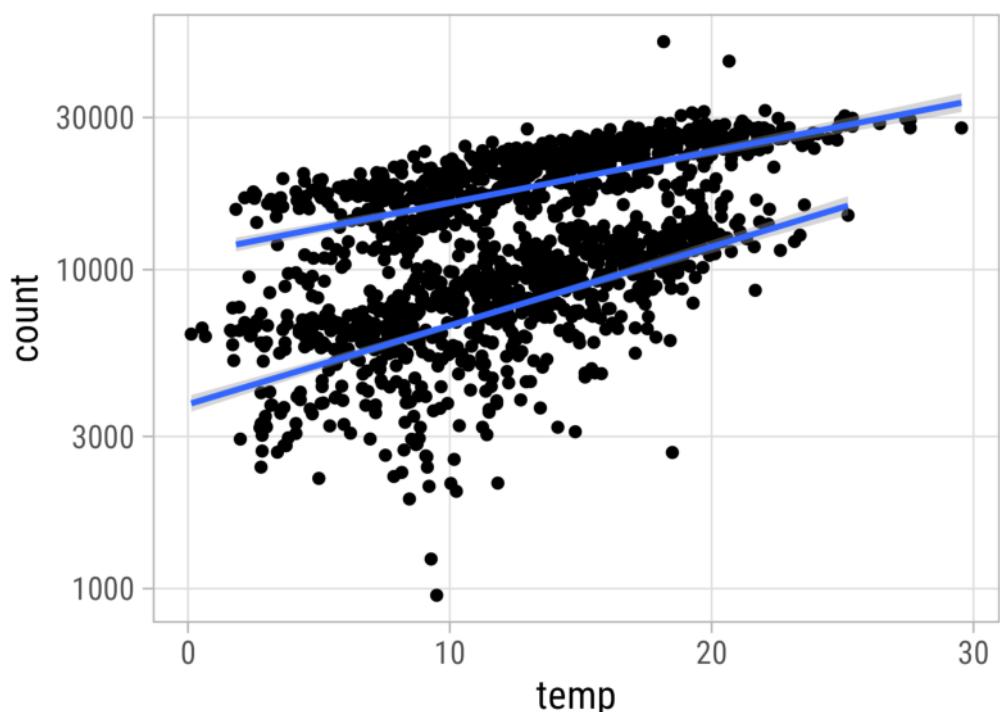


# Transform a Coordinate System

```
1 ggplot(  
2   bikes,  
3   aes(x = temp, y = count,  
4       group = day_night)  
5 ) +  
6   geom_point() +  
7   geom_smooth(method = "lm") +  
8   coord_trans(y = "log10")
```



```
1 ggplot(  
2   bikes,  
3   aes(x = temp, y = count,  
4       group = day_night)  
5 ) +  
6   geom_point() +  
7   geom_smooth(method = "lm") +  
8   scale_y_log10()
```



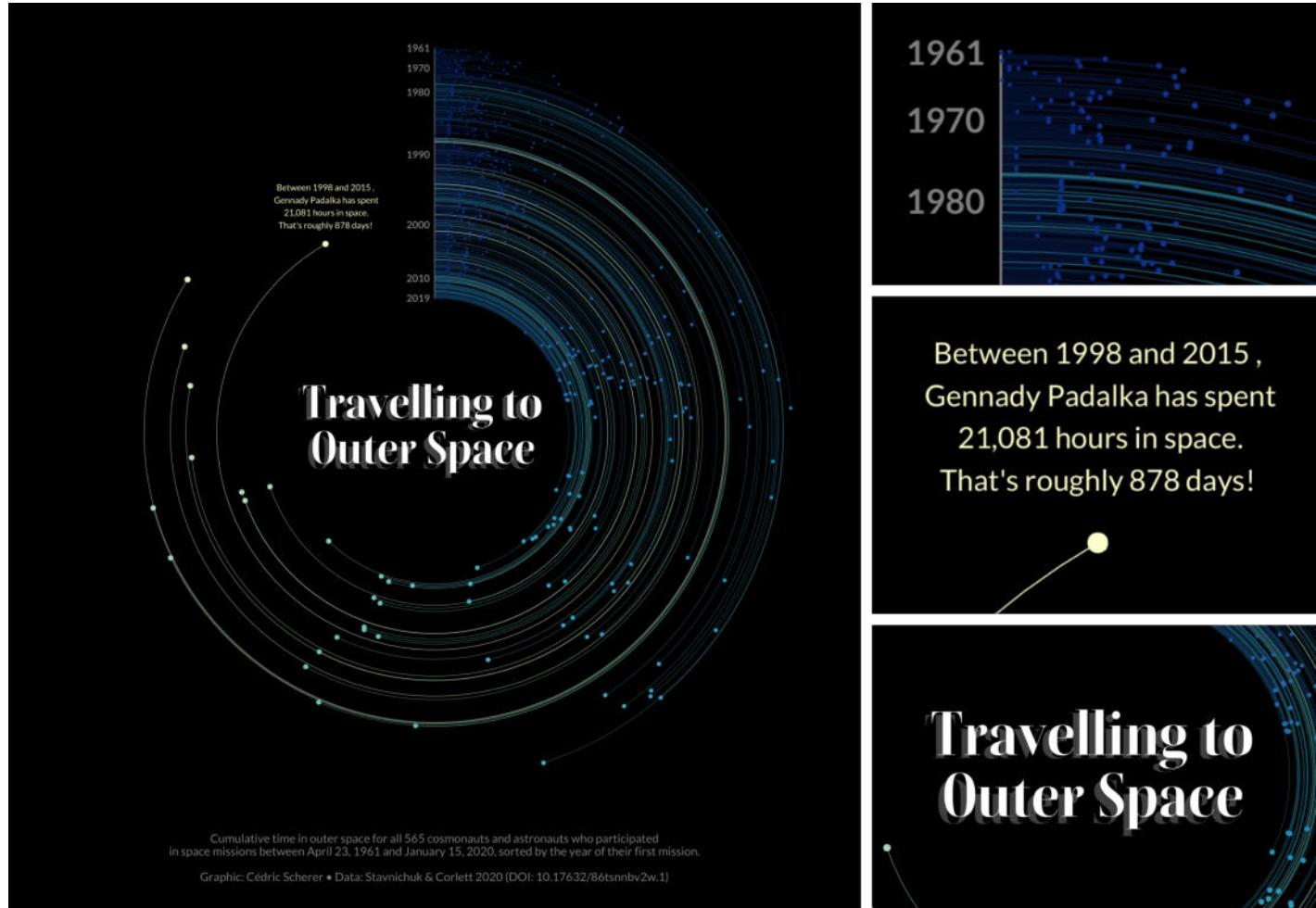
# Recap

- **facet\_\***() functions allow to create small multiples
- **scale\_\***() functions translate between **aes**hetics and properties
- use **\*\_continuous()** for numerical data
- ... and **\*\_discrete()** for categorical data
- use **scale\_color|fill\_\***() to customize data-related colors
- **coord\_\***() functions interpret the positional aesthetics
  - e.g. create polar plots or work with spatial projections
- be careful when adjusting axis limits:
  - **scale\_\*\_continuous(limits)** removes the data outside the range
  - **coord\_(\*lim)** keeps the data and zooms into the range

# Exercises

# Exercise 1

- Have a look at the following visualization of the cumulative time that cosmo- and astronauts have spent in outer space. The data also contains information on the year of their first and last travel, respectively.
- Together with your group, discuss which layers and modifications are needed to create such a chart with `{ggplot2}`.
- Note down the aesthetics, geometries, and scales used for each element of this graphic.
- What is the coordinate system? Have any adjustments been made?
- Which theme was used and how was it modified?



**I am done with the exercise!  
What now?**



Put on a **green** sticky note.



Is my neighbor done  
with the exercise as well?

**Yes**

**No**



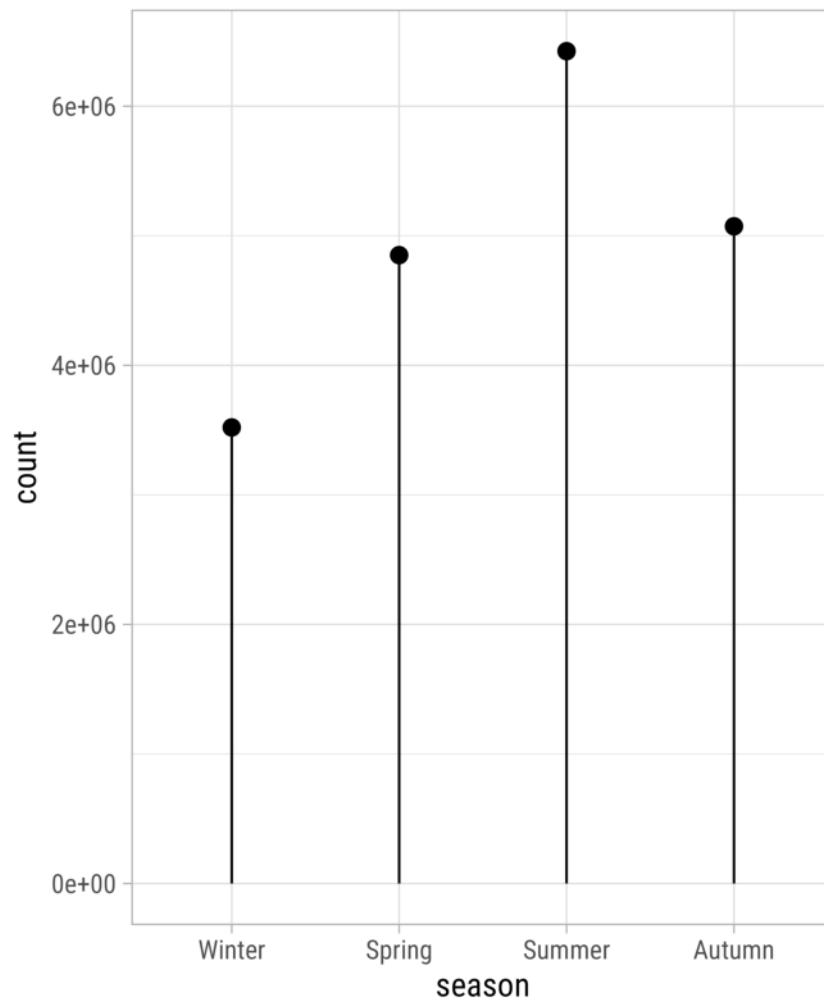
**Compare and discuss  
your solutions.**

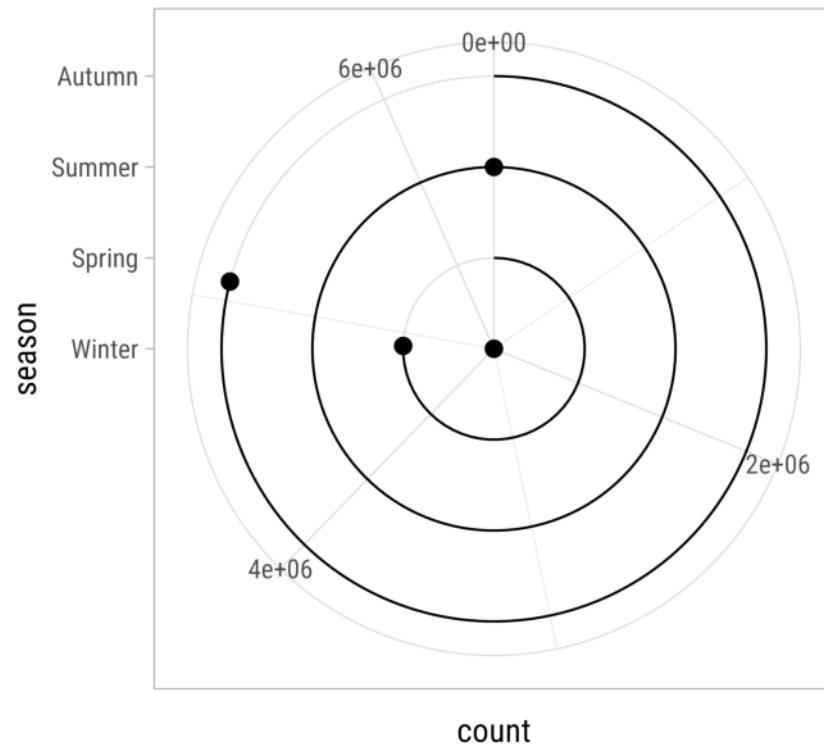
**Ask if you can help out.**

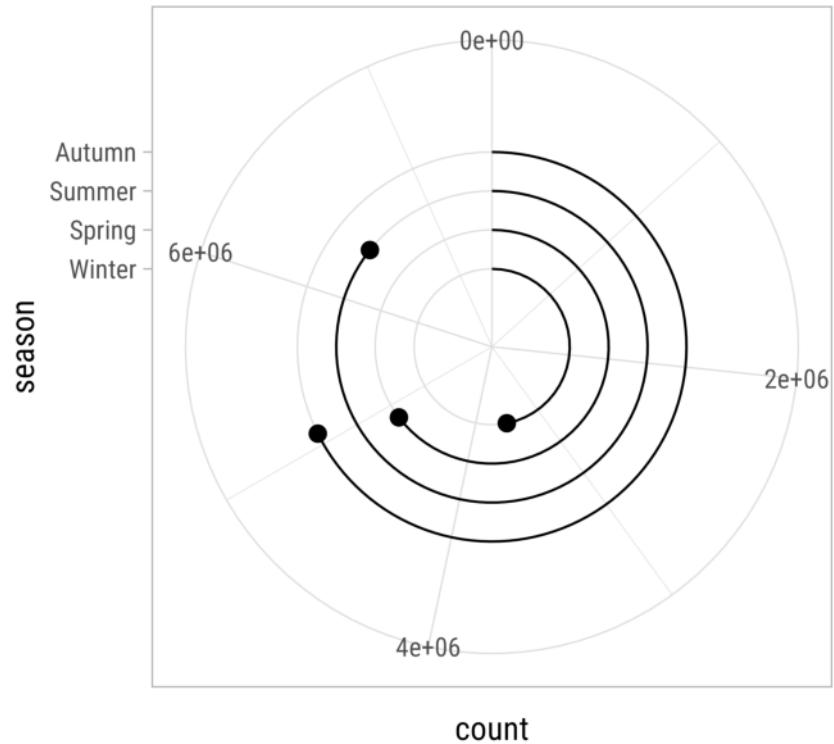
... or take a short break.

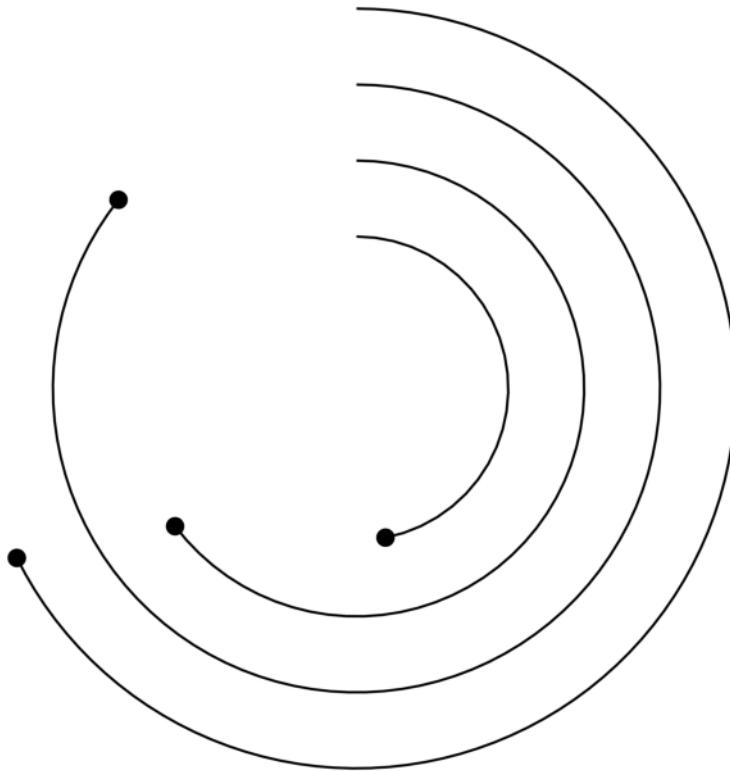
# Exercise 2

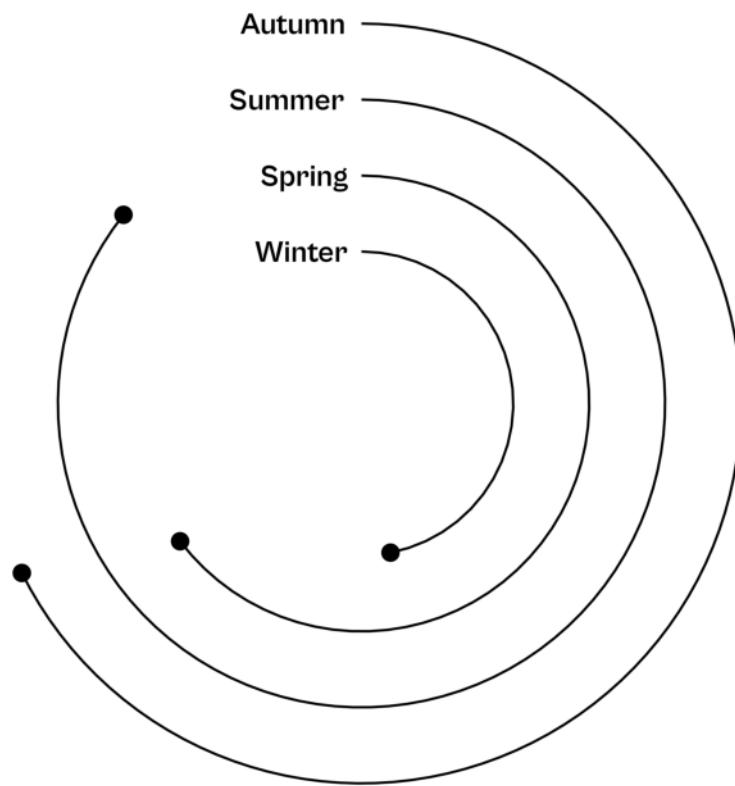
- Create a circular lollipop plot of reported bike shares per season.
- The data is not in the right format as there are no totals.  
How can you solve it?
- Remove all legend elements (with a single line of code).
- How can you add the labels next to the starting point of each lollipop?
- How could you add a baseline?

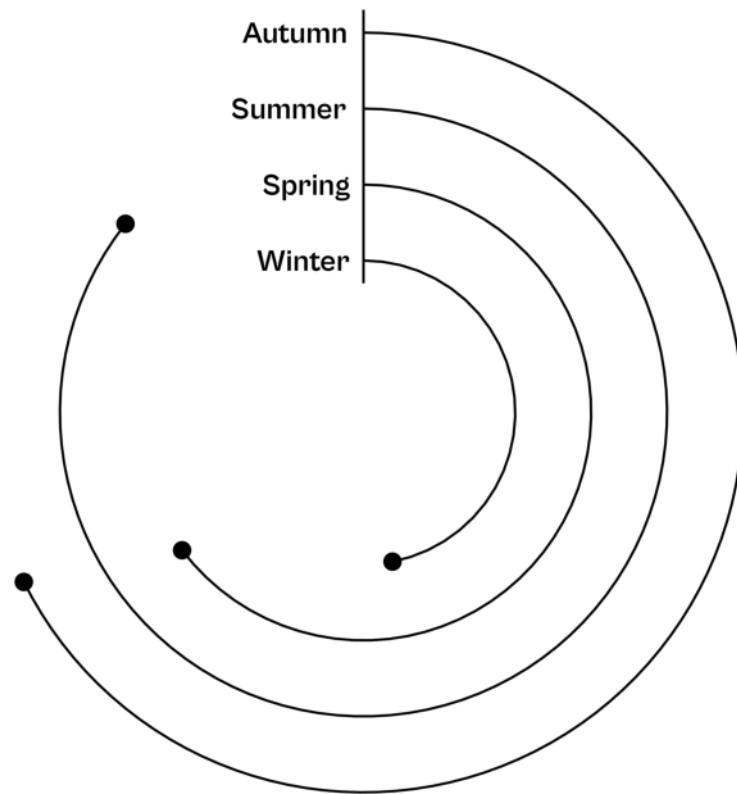












**I am done with the exercise!  
What now?**



Put on a **green** sticky note.



Is my neighbor done  
with the exercise as well?

**Yes**

**No**



**Compare and discuss  
your solutions.**

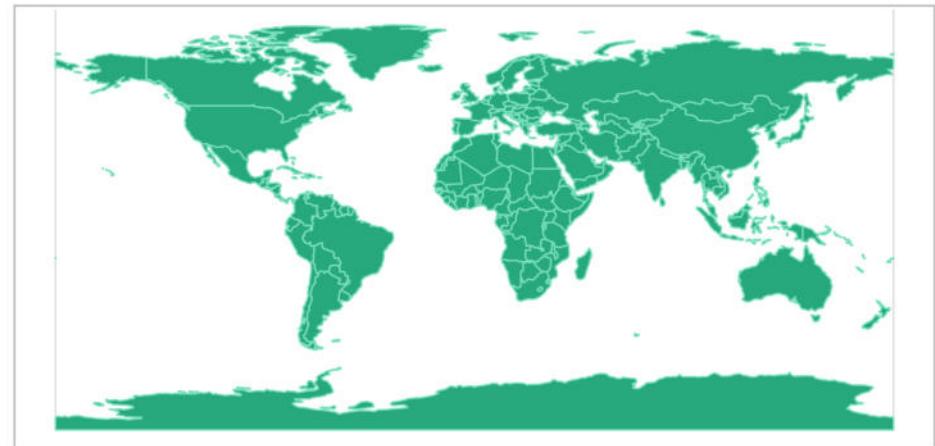
**Ask if you can help out.**

... or take a short break.

# Appendix

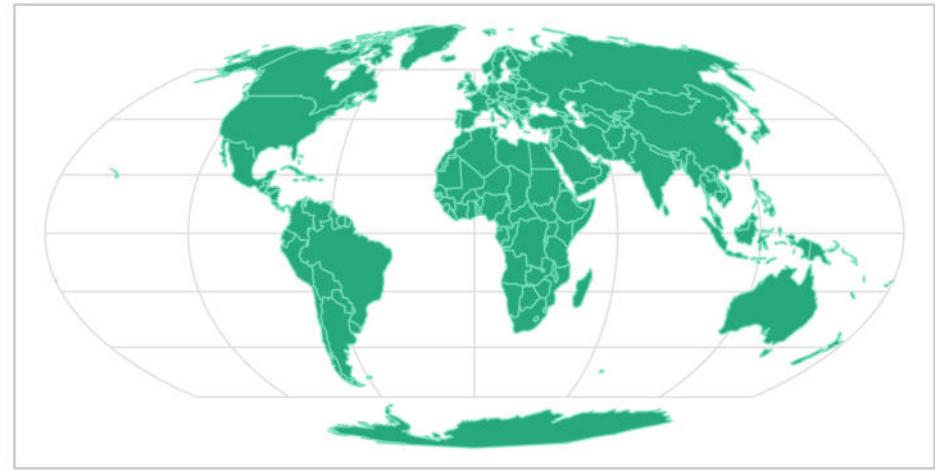
# Spatial Coordinate (Reference) Systems

```
1 countries <- rnaturalearth::ne_countries(  
2   returnclass = "sf"  
3 )  
4  
5 ggplot() +  
6   geom_sf(  
7     data = countries,  
8     color = "#79dfbd",  
9     fill = "#28a87d",  
10    size = .3  
11  )
```



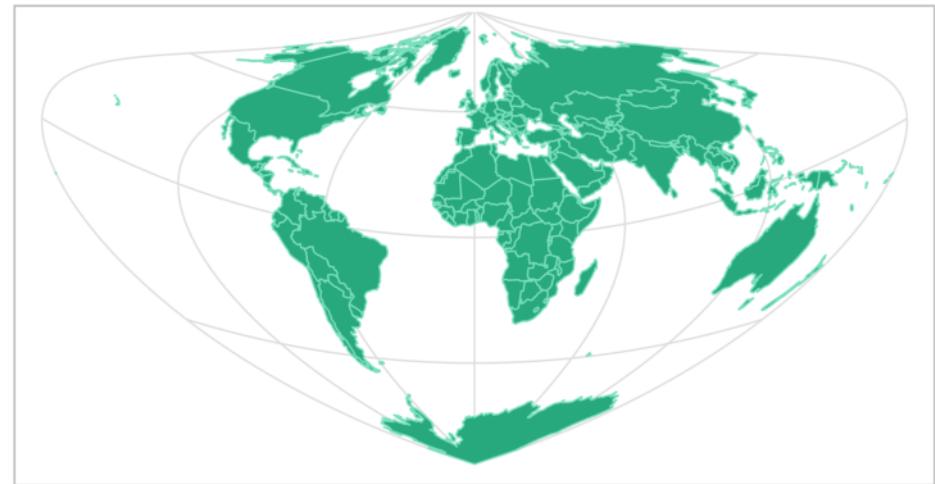
# Spatial Coordinate (Reference) Systems

```
1 ggplot() +  
2   geom_sf(  
3     data = countries,  
4     color = "#79dfbd",  
5     fill = "#28a87d",  
6     size = .3  
7   ) +  
8   coord_sf(  
9     crs = "+proj=moll"  
10  )
```



# Spatial Coordinate (Reference) Systems

```
1 ggplot() +  
2   geom_sf(  
3     data = countries,  
4     color = "#79dfbd",  
5     fill = "#28a87d",  
6     size = .3  
7   ) +  
8   coord_sf(  
9     crs = "+proj=bonne +lat_1=10"  
10  )
```



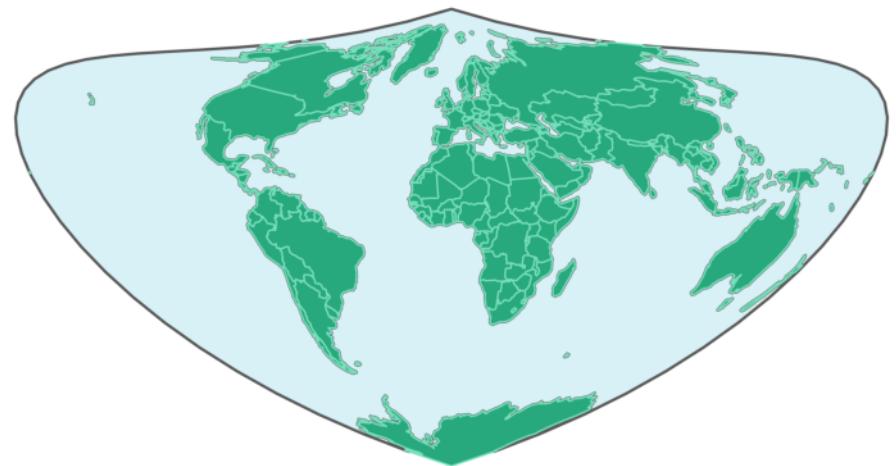
# Spatial Coordinate (Reference) Systems

```
1 oceans <- rnaturalearth::ne_download(  
2   category = "physical", type = "ocean", re·  
3 )
```

OGR data source with driver: ESRI  
Shapefile  
Source:  
"C:\Users\DaVizard\AppData\Local\Tem  
layer: "ne\_110m\_ocean"  
with 2 features  
It has 3 fields

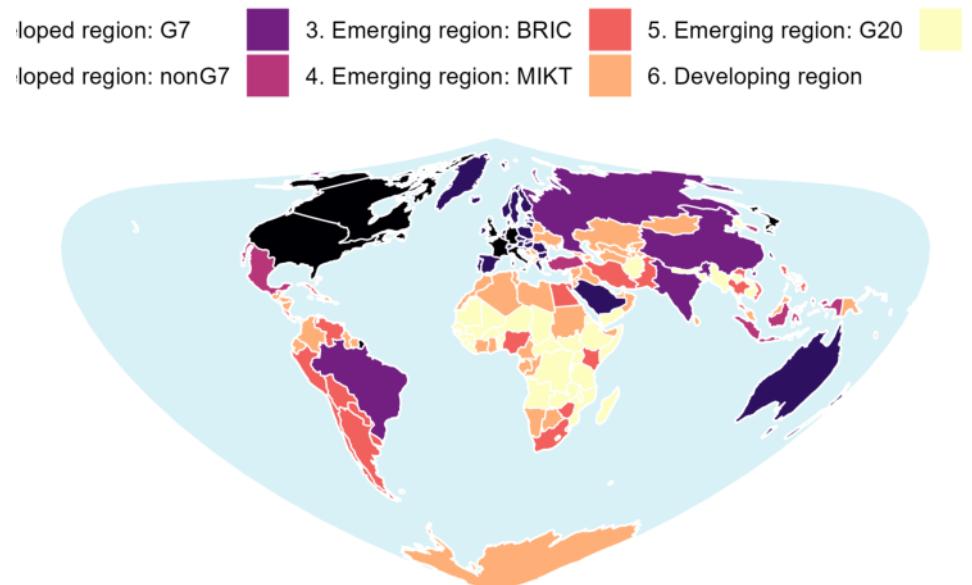
# Spatial Coordinate (Reference) Systems

```
1 ggplot() +  
2   geom_sf(  
3     data = oceans,  
4     fill = "#d8f1f6"  
5   ) +  
6   geom_sf(  
7     data = countries,  
8     color = "#79dfbd",  
9     fill = "#28a87d",  
10    size = .3  
11  ) +  
12  coord_sf(  
13    crs = "+proj=bonne +lat_1=10"  
14  ) +  
15  theme_void()
```



# Mapping of Visual Properties

```
1 ggplot() +  
2   geom_sf(  
3     data = oceans,  
4     fill = "#d8f1f6",  
5     color = "white"  
6   ) +  
7   geom_sf(  
8     data = countries,  
9     aes(fill = economy),  
10    color = "white",  
11    size = .3  
12  ) +  
13  coord_sf(  
14    crs = "+proj=bonne +lat_1=10"  
15  ) +  
16  scale_fill_viridis_d(option = "magma") +  
17  theme_void() +  
18  theme(legend.position = "top")
```



# Better Borders

```
1 borders <- rmapshaper::ms_innerlines(countries)
2
3 ggplot() +
4   geom_sf(
5     data = oceans,
6     fill = "#d8f1f6",
7     color = "white"
8   ) +
9   geom_sf(
10    data = countries,
11    aes(fill = economy),
12    color = "transparent"
13  ) +
14   geom_sf(
15    data = borders,
16    fill = "transparent",
17    color = "white",
18    size = .3
19  ) +
20   coord_sf(
21     crs = "+proj=bonne +lat_1=10"
22  ) +
23   scale_fill_viridis_d(option = "magma") +
```

