Designing Charts in R

Reproducible Graphic Design with {ggplot2}

Cédric Scherer // Data Visualization Society // March 9, 2023

Cédric Scherer // Data Visualization Society // March 9, 2023

Welcome!

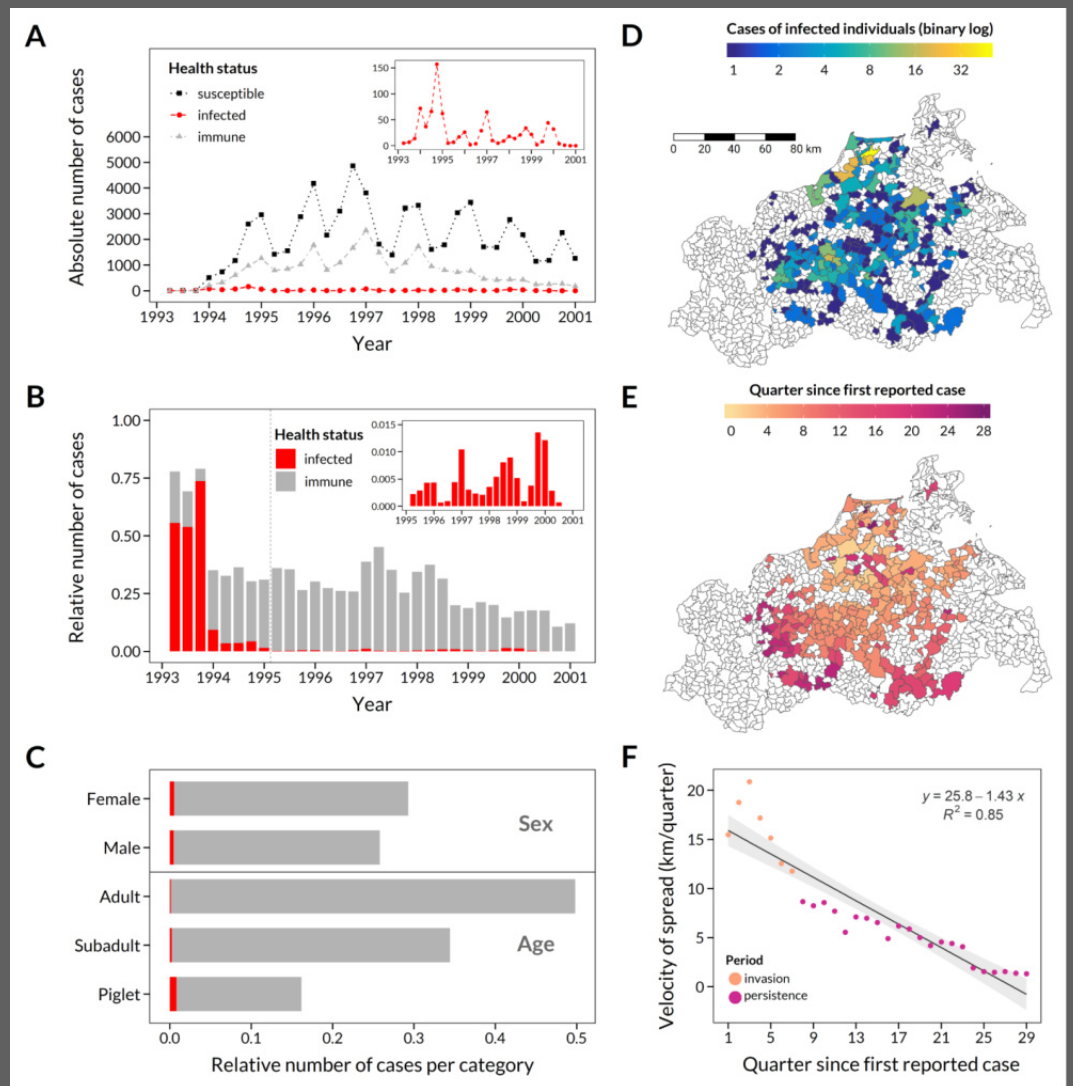
Cédric Scherer // Data Visualization Society // March 9, 2023

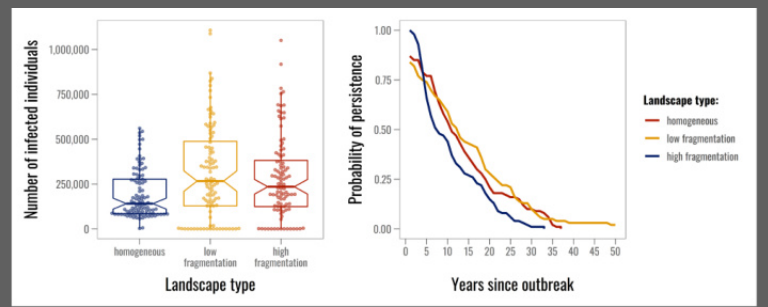

HELLO
my name is
Cédric

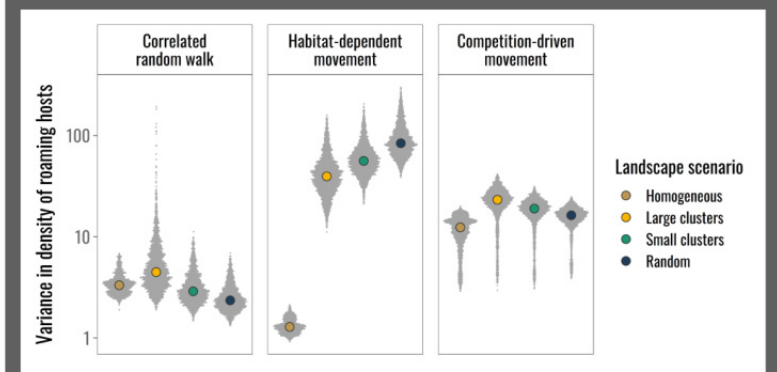
CÉDRIC SCHERER

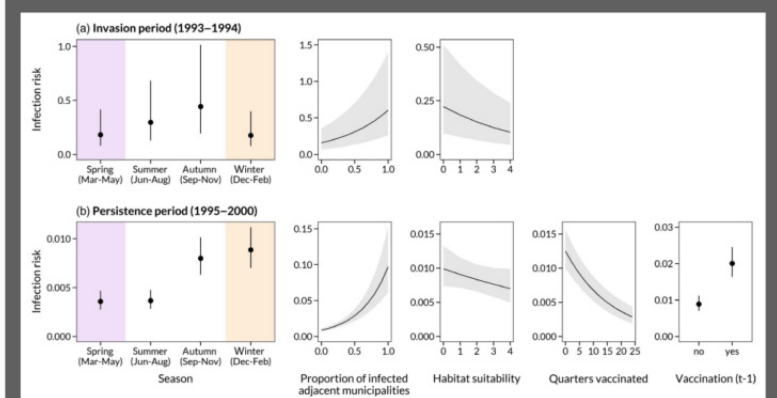
Data Visualization & Information Design

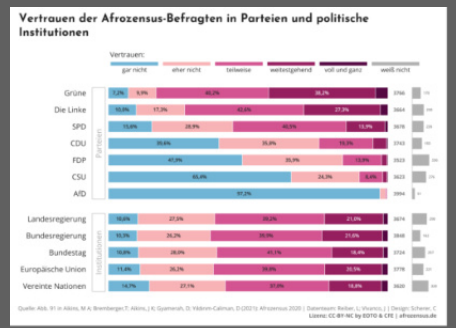
Consulting **Coaching** **Coding**

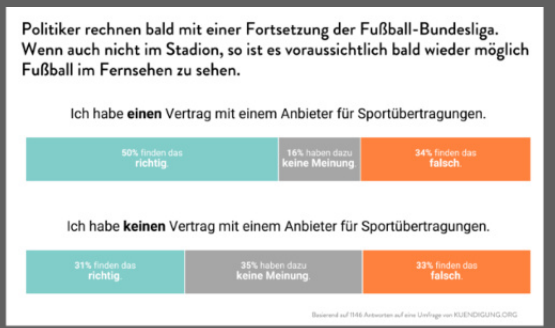

A
Health status
susceptible
infected
immune
Absolute number of cases
Year
B
Health status
infected
immune
Relative number of cases
Year
C
Female
Male
Sex
Adult
Subadult
Piglet
Age
Relative number of cases per category
D
Cases of infected individuals (binary log)
1 2 4 8 16 32
0 20 40 60 80 km
E
Quarter since first reported case
0 4 8 12 16 20 24 28
F
y = 25.8 − 1.43 x
R² = 0.85
Velocity of spread (km/quarter)
Period
invasion
persistence
Quarter since first reported case

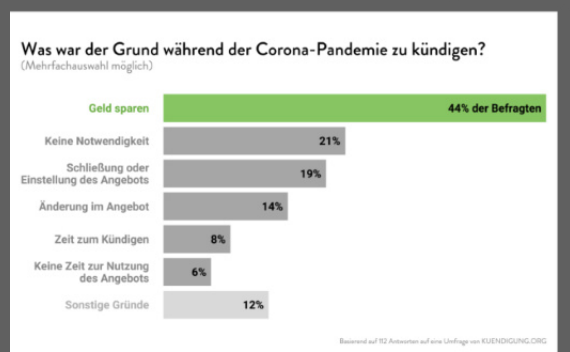

Number of infected individuals
homogeneous
low fragmentation
high fragmentation
Landscape type
Probability of persistence
Years since outbreak
Landscape type:
homogeneous
low fragmentation
high fragmentation


Correlated random walk
Habitat-dependent movement
Competition-driven movement
Variance in density of roaming hosts
Landscape scenario
Homogeneous
Large clusters
Small clusters
Random

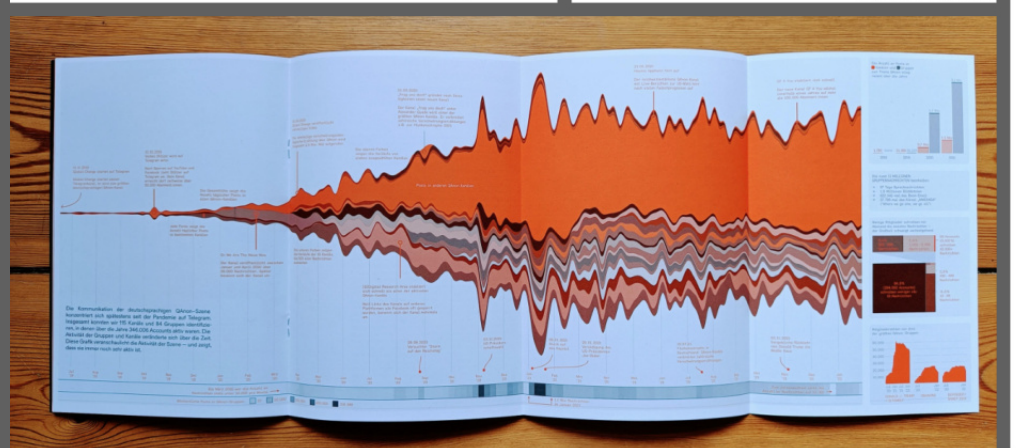

(a) Invasion period (1993–1994)
Infection risk
Spring (Mar-May)
Summer (Jun-Aug)
Autumn (Sep-Nov)
Winter (Dec-Feb)
(b) Persistence period (1995–2000)
Season
Proportion of infected adjacent municipalities
Habitat suitability
Quarters vaccinated
Vaccination (t-1)
no
yes

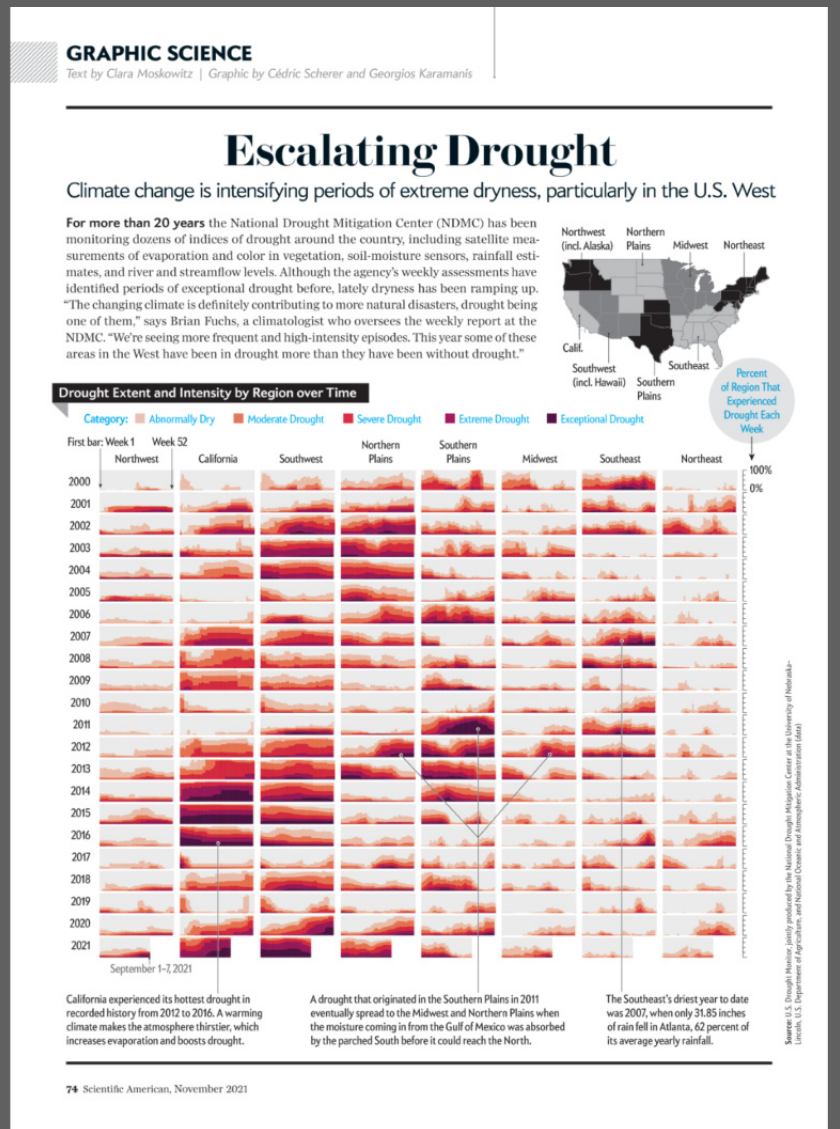

Vertrauen der Afrozensus-Befragten in Parteien und politische Institutionen
Vertrauen:
gar nicht
eher nicht
teilweise
weitestgehend
voll und ganz
weiß nicht
Grüne
Die Linke
SPD
CDU
FDP
CSU
AfD
Landesregierung
Bundesregierung
Bundestag
Europäische Union
Vereinte Nationen

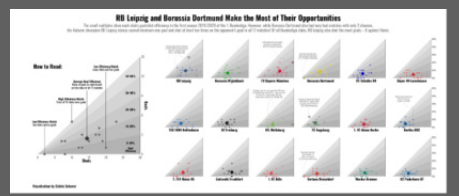

Politiker rechnen bald mit einer Fortsetzung der Fußball-Bundesliga. Wenn auch nicht im Stadion, so ist es voraussichtlich bald wieder möglich Fußball im Fernsehen zu sehen.
Ich habe einen Vertrag mit einem Anbieter für Sportübertragungen.
50% finden das richtig.
16% haben dazu keine Meinung.
34% finden das falsch.
Ich habe keinen Vertrag mit einem Anbieter für Sportübertragungen.
31% finden das richtig.
35% haben dazu keine Meinung.
33% finden das falsch.

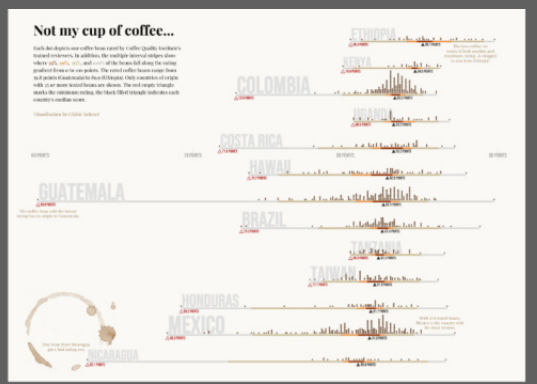

Was war der Grund während der Corona-Pandemie zu kündigen?
(Mehrfachauswahl möglich)
Geld sparen
44% der Befragten
Keine Notwendigkeit
21%
Schließung oder Einstellung des Angebots
19%
Änderung im Angebot
14%
Zeit zum Kündigen
8%
Keine Zeit zur Nutzung des Angebots
6%
Sonstige Gründe
12%

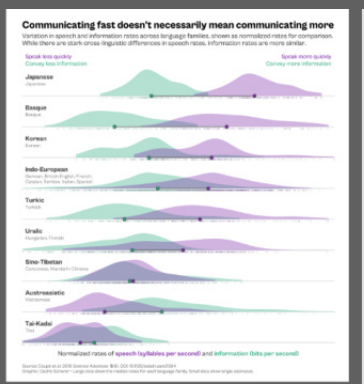

Häufigkeit von Diskriminierungserfahrungen entlang ausgewählter Vielfaltsdimensionen im Bereich „Arbeitsleben"
Anzahl afr. Elternteile
Beeinträchtigung
Bildungsgrad
Dt./EU Staatsangehörigkeit
Fluchterfahrung
Genderidentität
LSBAQ
Muslimisch
Nettoeinkommen




GRAPHIC SCIENCE
Text by Clara Moskowitz | Graphic by Cédric Scherer and Georgios Karamanis
Escalating Drought
Climate change is intensifying periods of extreme dryness, particularly in the U.S. West
For more than 20 years the National Drought Mitigation Center (NDMC) has been monitoring dozens of indices of drought around the country, including satellite measurements of evaporation and color in vegetation, soil-moisture sensors, rainfall estimates, and river and streamflow levels. Although the agency's weekly assessments have identified periods of exceptional drought before, lately dryness has been ramping up. "The changing climate is definitely contributing to more natural disasters, drought being one of them," says Brian Fuchs, a climatologist who oversees the weekly report at the NDMC. "We're seeing more frequent and high-intensity episodes. This year some of these areas in the West have been in drought more than they have been without drought."
Northwest (incl. Alaska)
Northern Plains
Midwest
Northeast
Calif.
Southwest (incl. Hawaii)
Southern Plains
Southeast
Drought Extent and Intensity by Region over Time
Category: Abnormally Dry Moderate Drought Severe Drought Extreme Drought Exceptional Drought
Percent of Region That Experienced Drought Each Week
First bar: Week 1
Week 52
Northwest
California
Southwest
Northern Plains
Southern Plains
Midwest
Southeast
Northeast
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
100%
0%
September 1–7, 2021
California experienced its hottest drought in recorded history from 2012 to 2016. A warming climate makes the atmosphere thirstier, which increases evaporation and boosts drought.
A drought that originated in the Southern Plains in 2011 eventually spread to the Midwest and Northern Plains when the moisture coming in from the Gulf of Mexico was absorbed by the parched South before it could reach the North.
The Southeast's driest year to date was 2007, when only 31.85 inches of rain fell in Atlanta, 62 percent of its average yearly rainfall.
Source: U.S. Drought Monitor, jointly produced by the National Drought Mitigation Center at the University of Nebraska–Lincoln, U.S. Department of Agriculture, and National Oceanic and Atmospheric Administration (data)
74 Scientific American, November 2021

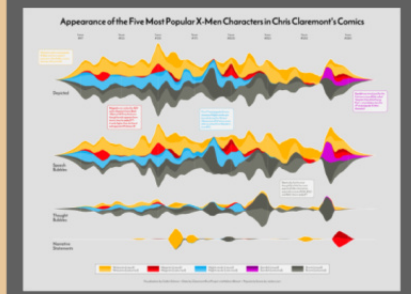

RB Leipzig and Borussia Dortmund Make the Most of Their Opportunities

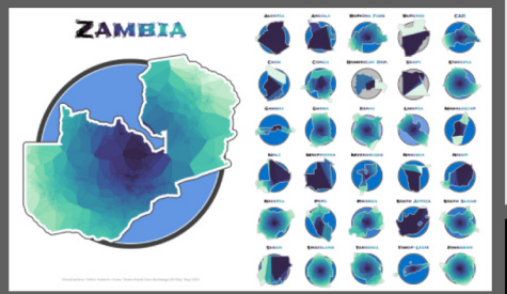

Not my cup of coffee...
ETHIOPIA
COLOMBIA
KENYA
COSTA RICA
HAWAII
GUATEMALA
BRAZIL
TANZANIA
TAIWAN
HONDURAS
MEXICO
NICARAGUA

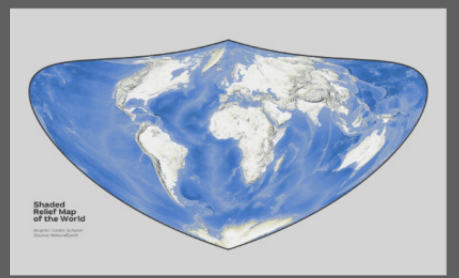

Communicating fast doesn't necessarily mean communicating more
Japanese
Basque
Korean
Indo-European
Turkic
Uralic
Sino-Tibetan
Austroasiatic
Tai-Kadai
Normalized rates of speech (syllables per second) and information (bits per second)

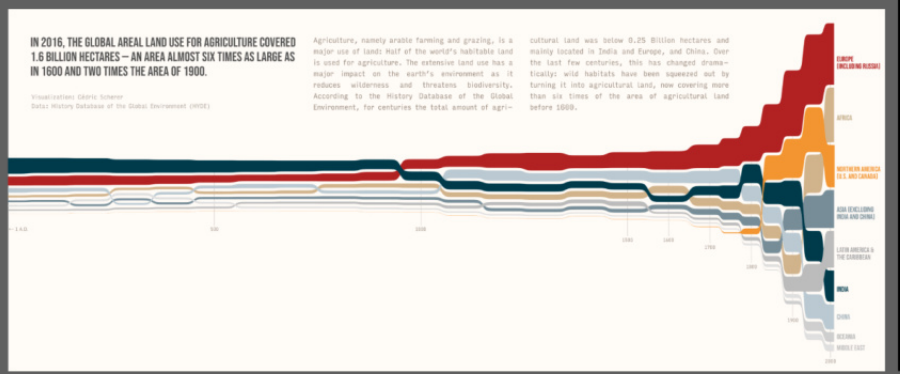

Ergebnisse der Bundestagswahl 2021
Die stärksten Parteien nach Prozent der Zweitstimmen.
SPD
CDU
CSU
Bündnis 90/Die Grünen
AfD
Grafik: Cédric Scherer · Daten: DIE ZEIT · Gridkarte: Ansgar Wolsing & Cédric Scherer

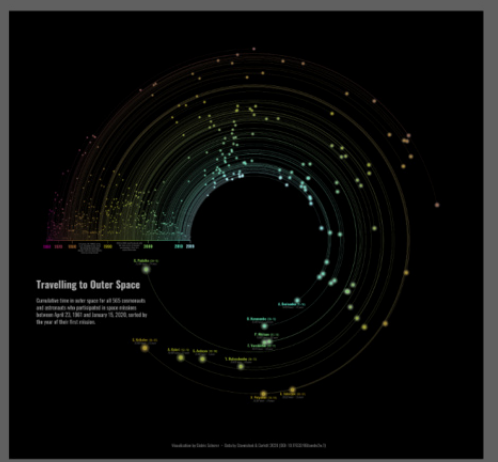

Show Me the Honey:
Where My Beekeepers At?!
Counts of Beekeepers in Germany Listed in OpenStreetMap
Graphic: Cédric Scherer • Source: OpenStreetMap Contributors

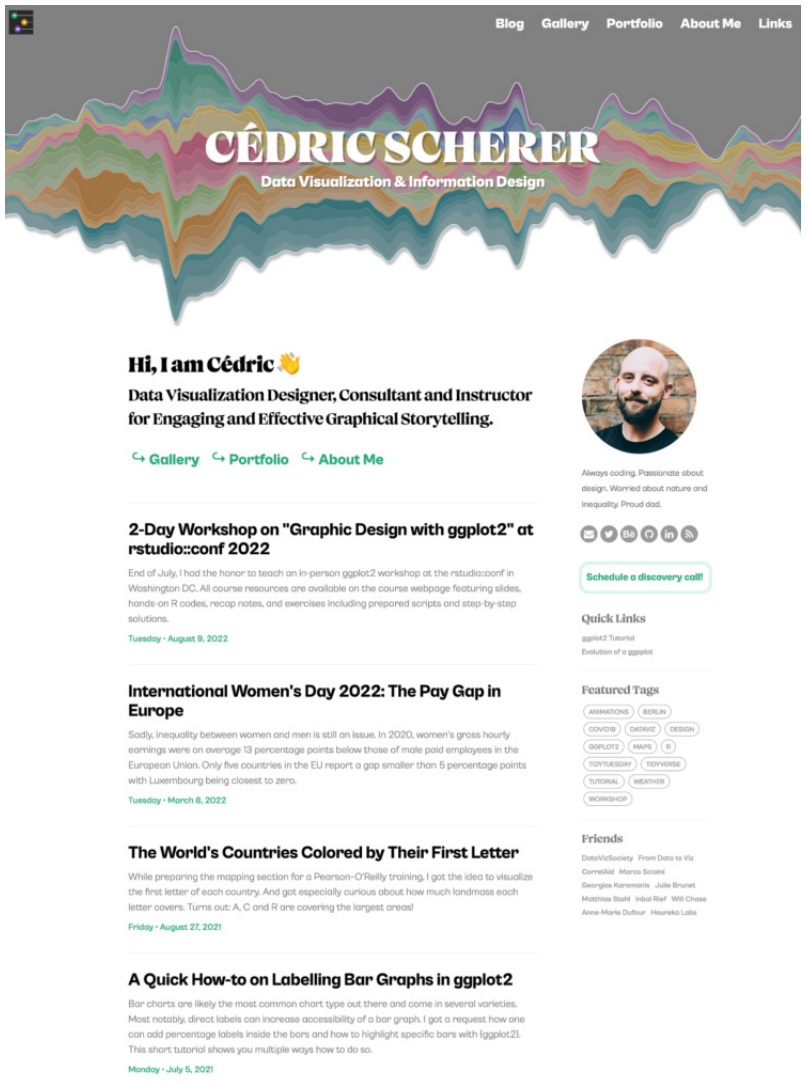

Appearance of the Five Most Popular X-Men Characters in Chris Claremont's Comics

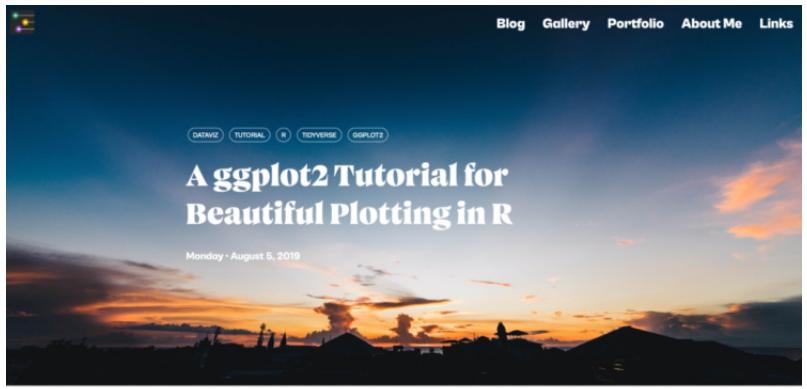

ZAMBIA


Shaded
Relief Map
of the World


IN 2016, THE GLOBAL AREAL LAND USE FOR AGRICULTURE COVERED 1.6 BILLION HECTARES – AN AREA ALMOST SIX TIMES AS LARGE AS IN 1600 AND TWO TIMES THE AREA OF 1900.
Agriculture, namely arable farming and grazing, is a major use of land: Half of the world's habitable land is used for agriculture. The extensive land use has a major impact on the earth's environment as it reduces wilderness and threatens biodiversity. According to the History Database of the Global Environment, for centuries the total amount of agricultural land was below 0.25 Billion hectares and mainly located in India and Europe, and China. Over the last few centuries, this has changed dramatically: wild habitats have been squeezed out by turning it into agricultural land, now covering more than six times of the area of agricultural land before 1600.
Visualisation: Cédric Scherer
Data: History Database of the Global Environment (HYDE)
EUROPE (INCLUDING RUSSIA)
AFRICA
NORTHERN AMERICA (U.S. AND CANADA)
ASIA (EXCLUDING INDIA AND CHINA)
LATIN AMERICA & THE CARIBBEAN
INDIA
CHINA
OCEANIA
MIDDLE EAST


Travelling to Outer Space


Blog Gallery Portfolio About Me Links
CÉDRIC SCHERER
Data Visualization & Information Design
Hi, I am Cédric 👋
Data Visualization Designer, Consultant and Instructor for Engaging and Effective Graphical Storytelling.
↪ Gallery ↪ Portfolio ↪ About Me
2-Day Workshop on "Graphic Design with ggplot2" at rstudio::conf 2022
End of July, I had the honor to teach an in-person ggplot2 workshop at the rstudio::conf in Washington DC. All course resources are available on the course webpage featuring slides, hands-on R codes, recap notes, and exercises including prepared scripts and step-by-step solutions.
Tuesday • August 9, 2022
International Women's Day 2022: The Pay Gap in Europe
Sadly, inequality between women and men is still an issue. In 2020, women's gross hourly earnings were on average 13 percentage points below those of male paid employees in the European Union. Only five countries in the EU report a gap smaller than 5 percentage points with Luxembourg being closest to zero.
Tuesday • March 8, 2022
The World's Countries Colored by Their First Letter
While preparing the mapping section for a Pearson-O'Reilly training, I got the idea to visualize the first letter of each country. And got especially curious about how much landmass each letter covers. Turns out: A, C and R are covering the largest areas!
Friday • August 27, 2021
A Quick How-to on Labelling Bar Graphs in ggplot2
Bar charts are likely the most common chart type out there and come in several varieties. Most notably, direct labels can increase accessibility of a bar graph. I got a request how one can add percentage labels inside the bars and how to highlight specific bars with {ggplot2}. This short tutorial shows you multiple ways how to do so.
Monday • July 5, 2021
Always coding. Passionate about design. Worried about nature and inequality. Proud dad.
Schedule a discovery call!
Quick Links
ggplot2 Tutorial
Evolution of a ggplot
Featured Tags
ANIMATIONS BERLIN COVID19 DATAVIZ DESIGN GGPLOT2 MAPS R TIDYTUESDAY TIDYVERSE TUTORIAL WEATHER WORKSHOP
Friends
DataVizSociety From Data to Viz CorrelAid Marco Sciaini Georgios Karamanis Julie Brunet Matthias Stahl Inbal Rief Will Chase Anne-Marie Dufour Heureka Labs



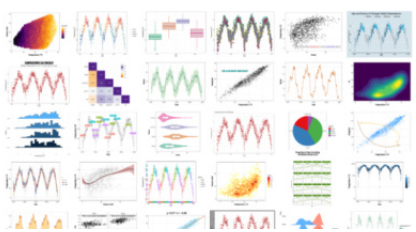
Last update: 2022-03-24

Introductory Words

I don't care, just show me the content!

Back in 2016, I had to prepare my PhD introductory talk and I started using `{ggplot2}` to visualize my data. I never liked the syntax and style of base plots in R, so I was quickly in love with ggplot. Especially useful was its faceting utility. But because I was short on time, I plotted these figures by trial and error and with the help of lots of googling. The resource I came always back to was a blog entry called Beautiful plotting in R: A ggplot2 cheatsheet by Zev Ross, updated last in January 2016. After giving the talk which contained some decent plots thanks to the blog post, I decided to go through this tutorial step-by-step. I learned so much from it and directly started modifying the codes and over the time I added additional code snippets, chart types and resources.

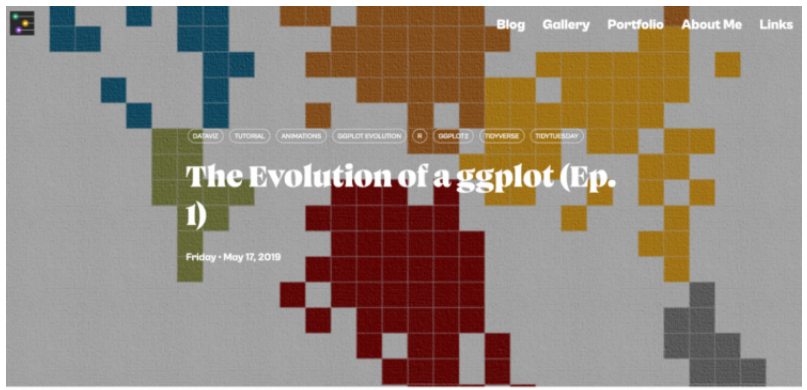
Since the blog entry by Zev Ross was not updated for some years and step by step this became a unique version of a tutorial, I decided to host the updated version on my GitHub. Now it finds its proper place on this homepage! (Plus I added a ton of other updates–just to name a few: The fantastic `{patchwork}`, `{ggtext}` and `{ggforce}` packages. How to deal with custom fonts and colors. A collection of R packages tailored to create interactive charts. And several other chart types including pie charts because everyone looooves pie charts!)

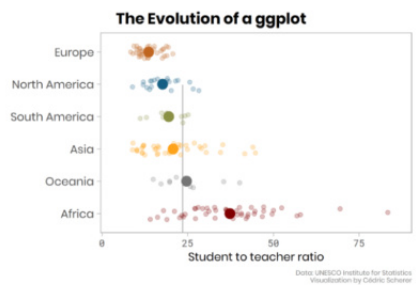


End of July, I had the honor to teach a **2-day, in-person workshop on "Graphic Design with ggplot2" at the rstudio::conf(2022)** in Washington DC. Invited by RStudio (now named **Posit**), I developed a new course that covers the most important steps and helpful tips to create visually appealing, engaging and complex graphics with ggplot2. The course focused on the main concepts of the grammar of graphics and used hands-on examples to explore ggplot2's utility to create multi-layered, more complex graphs. **All course resources are available as open-source material on the course page.**

The course webpage as well ass the slidedecks and the exercises and solutions were developed with the new open-source scientific and technical publishing system **Quarto**. The new workshop development was a perfect opportunity to give it a try and the experience was overall wonderful–the **reveal.js** integration for the slides works perfect and allows for a lot of customization. Thanks to Marco Sciaini for helping me setting up the course webpage which was, after learning about a few quirks, a smooth experience as well.

The **course webpage** which gives an overview of the course objectives and links to all session and the **GitHub repository**.





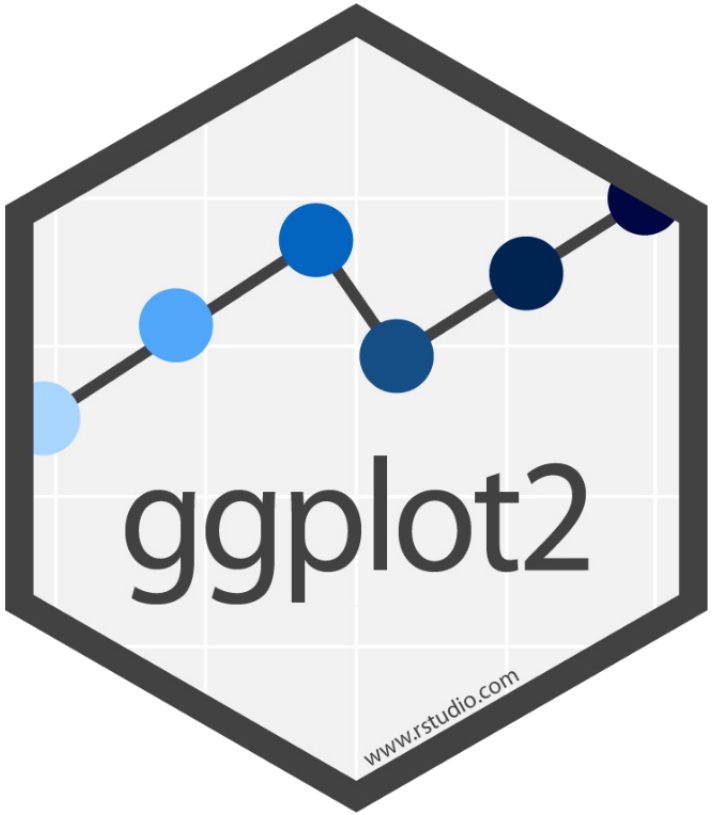
Aim of this Tutorial

In this series of blog posts, I aim to show you how to turn a default ggplot into a plot that visualizes information in an appealing and easily understandable way. The goal of each blog post is to provide a step-by-step tutorial explaining how my visualization have evolved from a typical basic ggplot. All plots are going to be created with 100% `{ggplot2}` and 0% Inkscape.


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
Overview
Repositories 81
Projects 1
Packages
Stars 441
Cédric Scherer
z3tt
Data Visualization Specialist · Computational Ecologist
Edit profile
1.4k followers · 119 following
Self-Employed · IZW Berlin
Berlin
cedricscherer.com
@CedScherer
Achievements
Beta Send feedback
Organizations
z3tt / README.md
CÉDRIC SCHERER
Data Visualization & Information Design
Commits trying to make funky typefaces work
Commits that actually contain code changes
My GitHub Activity
aka "How it feels to be a DataViz coder"
Commits fixing typographical errros typographical errors typos
What I Actually Do
aka "Do you need a DataViz coder?"
Buy me a coffee
Pinned
Customize your pins
TidyTuesday Public
My contributions to the #TidyTuesday challenge, a weekly data visualization challenge. All plots are created in R with ggplot2.
R 650 132
OutlierConf2021 Public
Slides and hands-on codes for my talk "ggplot Wizardry: My Favorite Tricks and Secrets for Beautiful Plots in R" at the 1st OutlierConf, February 4–7 2021.
HTML 404 72
rstudio-conf-2022/ggplot2-graphic-design Public
"Graphic Design with ggplot2: How to Create Engaging and Complex Visualizations in R", an rstudio::conf(2022) Workshop by Cédric Scherer
R 122 56
beyond-bar-and-box-plots Public
Slides, recording and hands-on tutorial for the USGS Data Science webinar
HTML 219 37
30DayMapChallenge2019 Public
My contributions to the #30DayMapChallenge 2019, a daily challenge focusing on spatial visualizations happening throughout November.
HTML 172 36
thebioengineer/camcorder Public
Record plots generated during an R session and replay as a gif!
R 154 4
822 contributions in the last year
Contribution settings
2023

The ggplot2 Package

Cédric Scherer // Data Visualization Society // March 9, 2023



`{ggplot2}` is a system for declaratively creating graphics, based on "The Grammar of Graphics" (Wilkinson, 2005).

You provide the data, tell `{ggplot2}` how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

Advantages of {ggplot2}

- code-first approach → reproducible and transparent workflow
- consistent underlying "grammar of graphics"
- very flexible, layered plot specification
- theme system for polishing plot appearance
- lots of additional functionality thanks to extensions
- active and helpful community

ggplot2: Visual Data Exploration

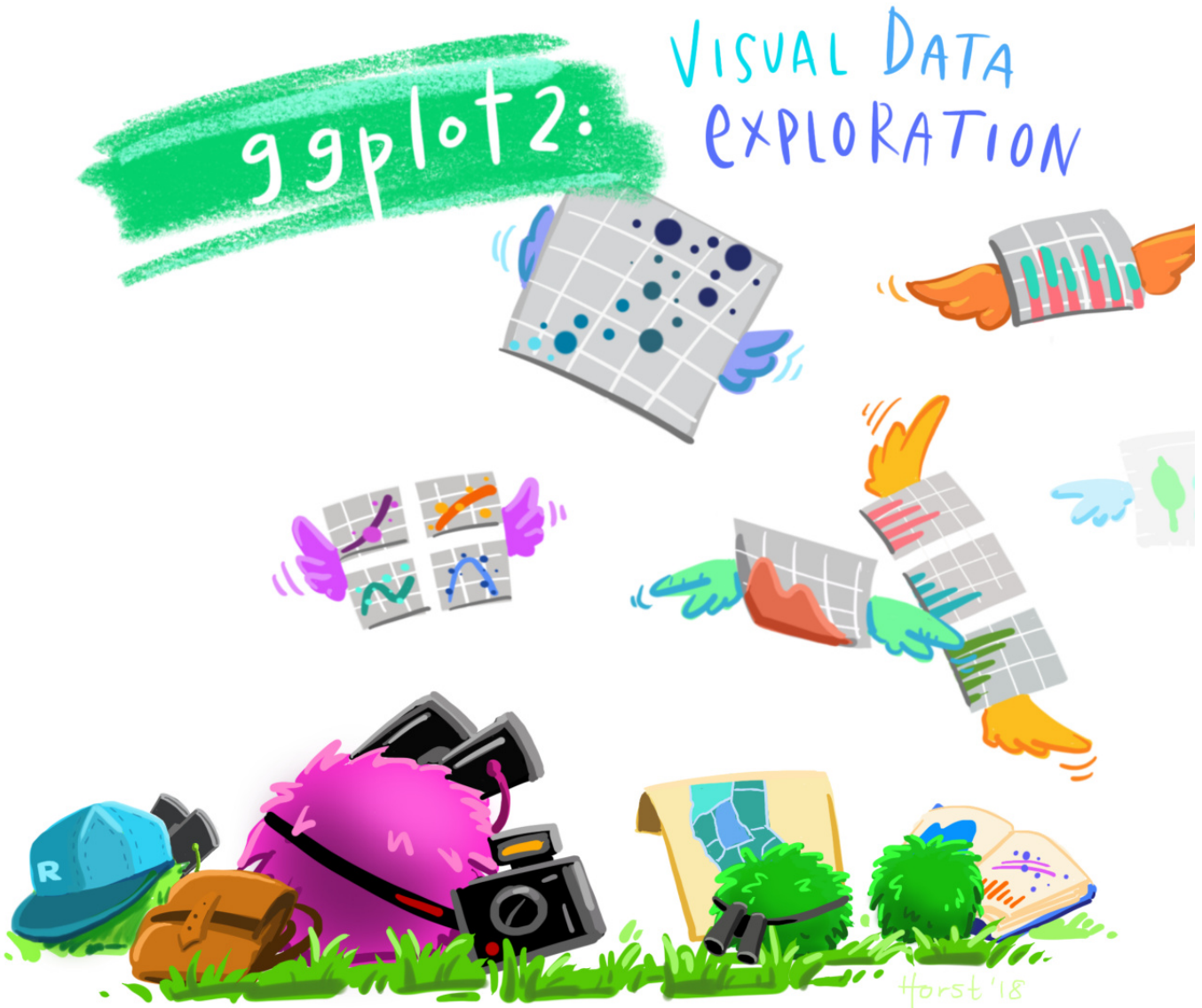
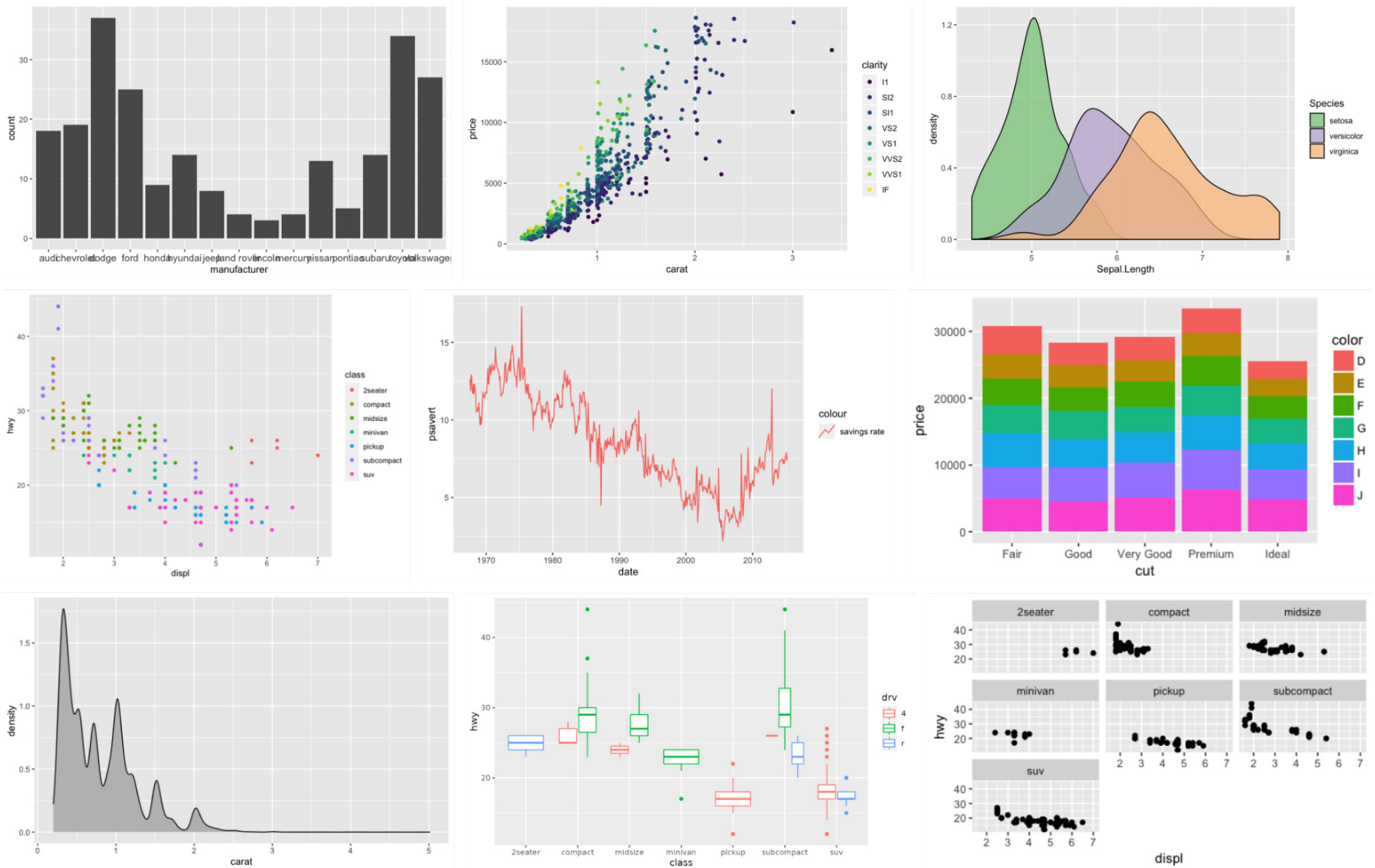


Illustration by Allison Horst

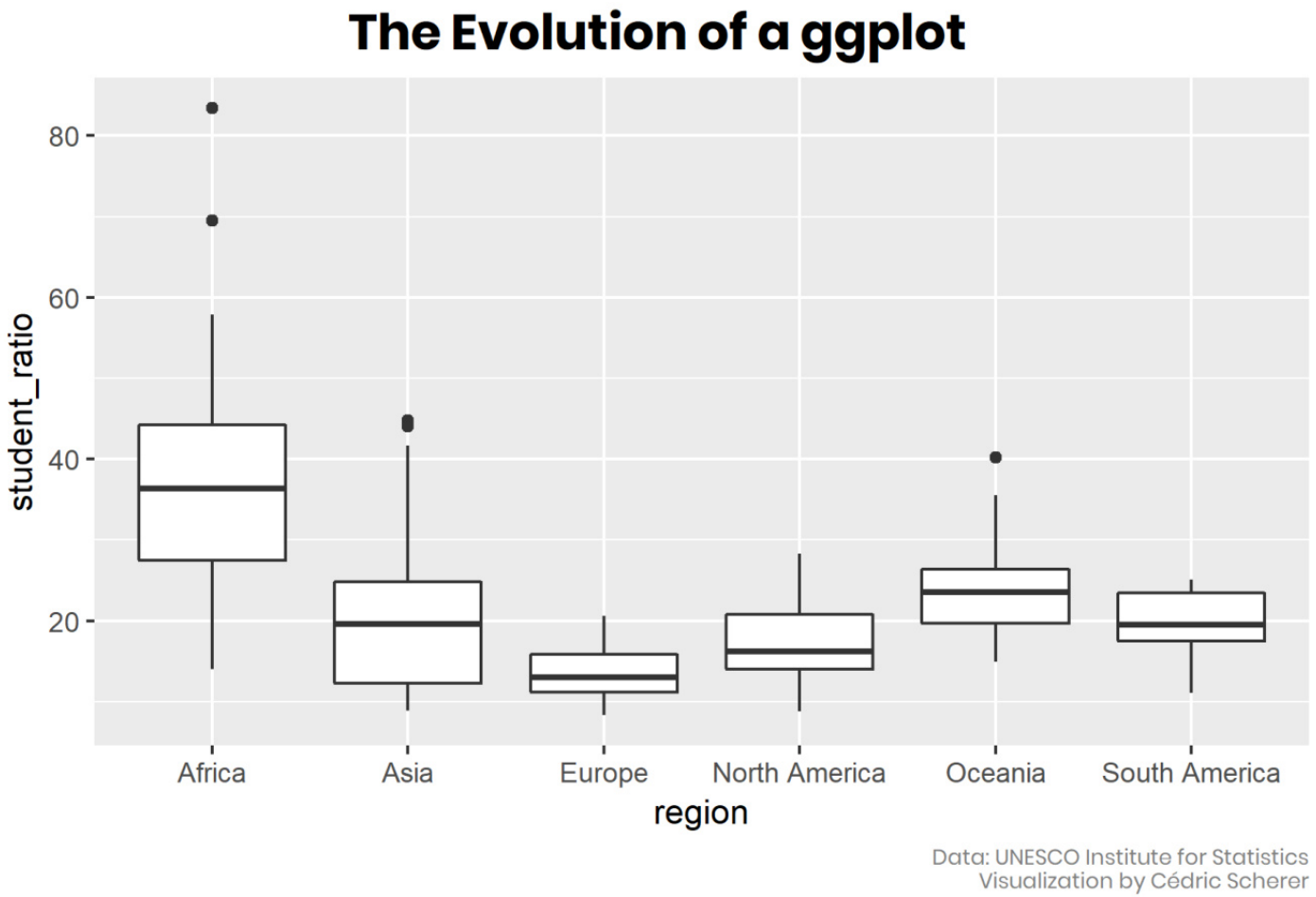
Cédric Scherer // Data Visualization Society // March 9, 2023

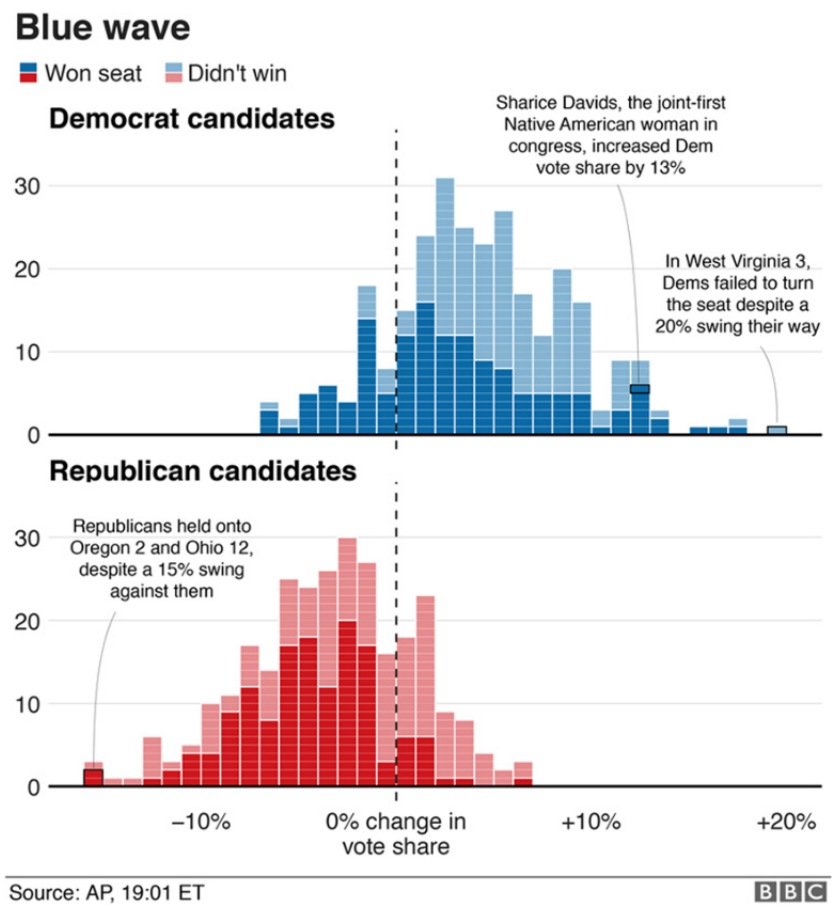


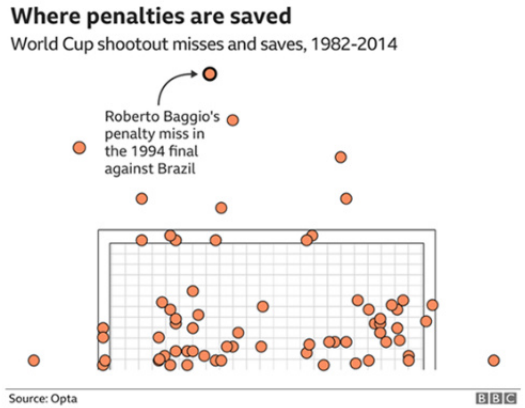
ggplot2 Examples featured on ggplot2.tidyverse.org

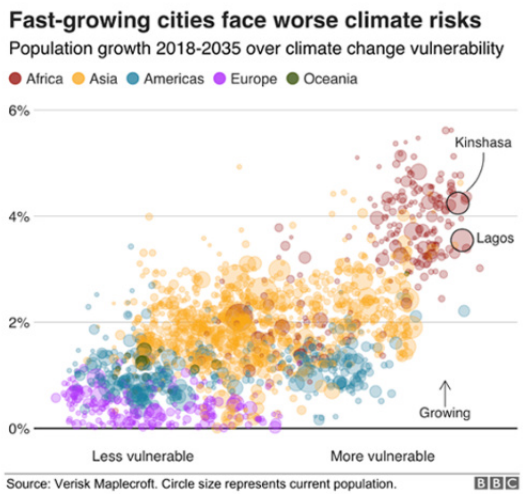
Illustration by Allison Horst

Cédric Scherer // Data Visualization Society // March 9, 2023

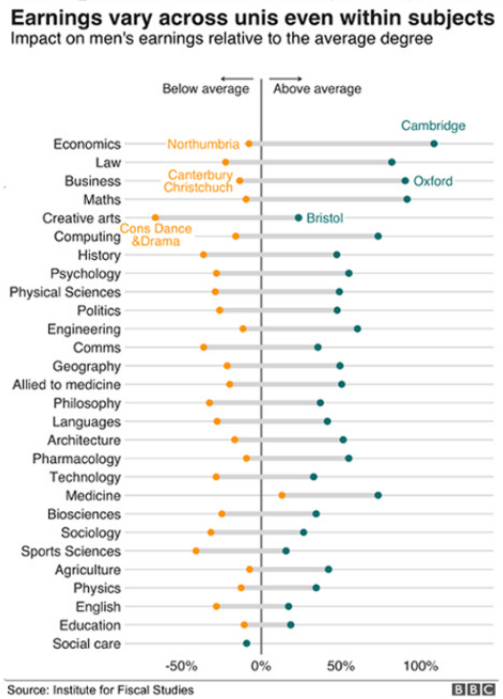

The Evolution of a ggplot
student_ratio
80
60
40
20
Africa
Asia
Europe
North America
Oceania
South America
region
Data: UNESCO Institute for Statistics
Visualization by Cédric Scherer

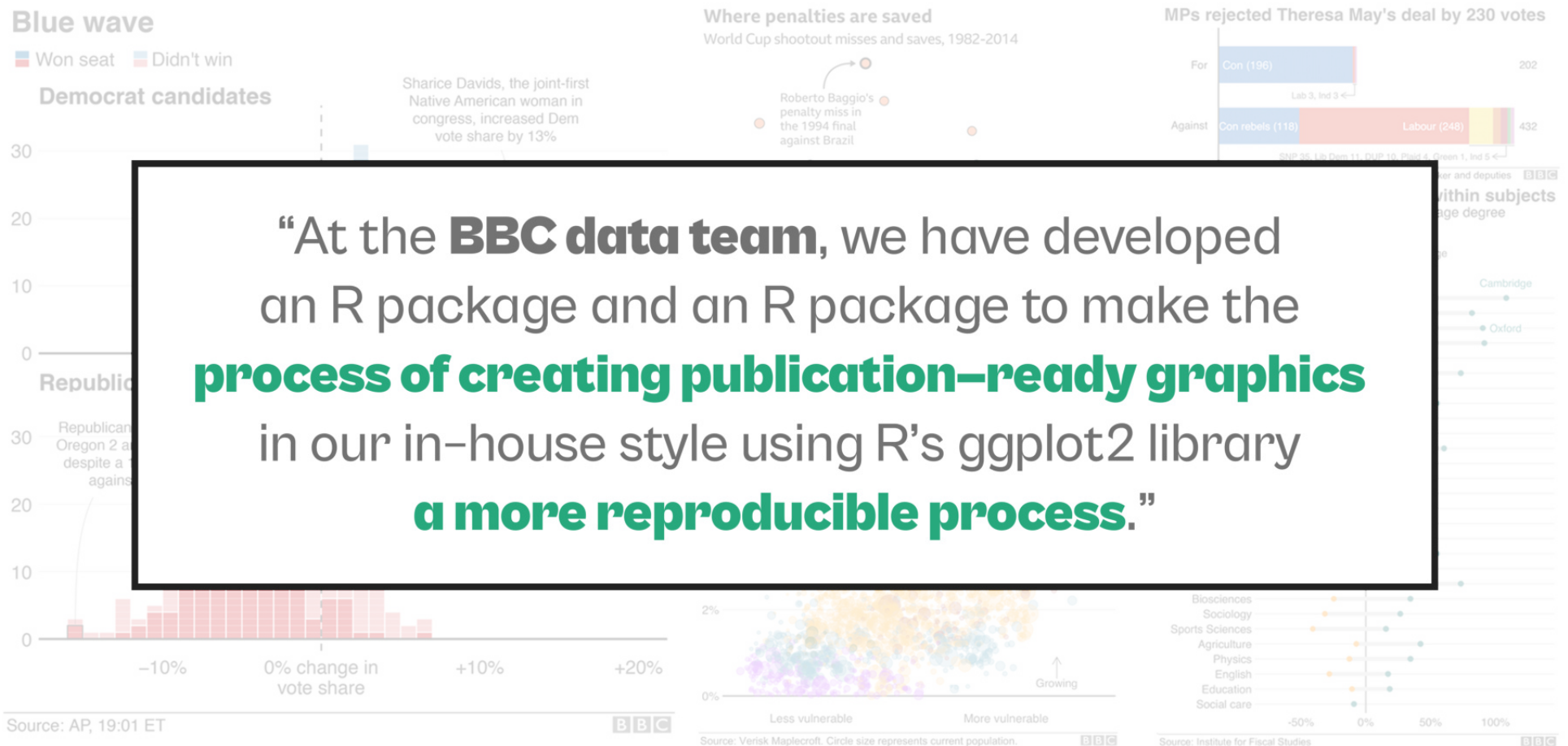

Blue wave
Won seat
Didn't win
Democrat candidates
Sharice Davids, the joint-first Native American woman in congress, increased Dem vote share by 13%
In West Virginia 3, Dems failed to turn the seat despite a 20% swing their way
30
20
10
0
Republican candidates
Republicans held onto Oregon 2 and Ohio 12, despite a 15% swing against them
30
20
10
0
−10%
0% change in vote share
+10%
+20%
Source: AP, 19:01 ET
BBC


Where penalties are saved
World Cup shootout misses and saves, 1982-2014
Roberto Baggio's penalty miss in the 1994 final against Brazil
Source: Opta
BBC

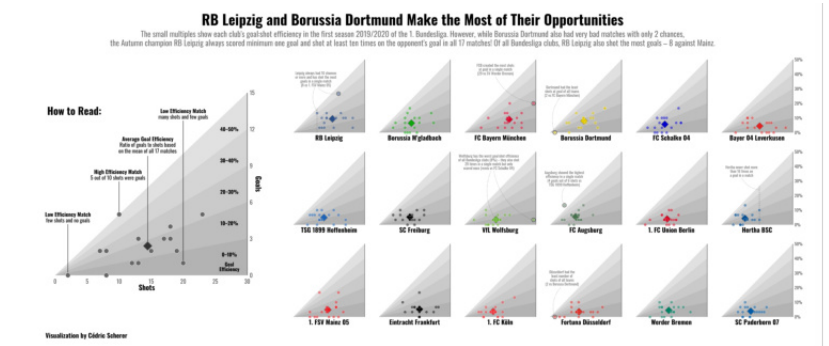

Fast-growing cities face worse climate risks
Population growth 2018-2035 over climate change vulnerability
Africa
Asia
Americas
Europe
Oceania
6%
4%
2%
0%
Kinshasa
Lagos
Growing
Less vulnerable
More vulnerable
Source: Verisk Maplecroft. Circle size represents current population.
BBC

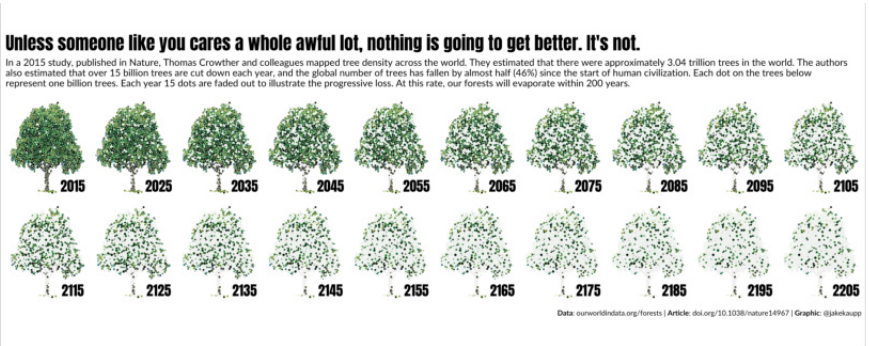

MPs rejected Theresa May's deal by 230 votes
For
Con (196)
202
Lab 3, Ind 3
Against
Con rebels (118)
Labour (248)
432
SNP 35, Lib Dem 11, DUP 10, Plaid 4, Green 1, Ind 5
Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies
BBC

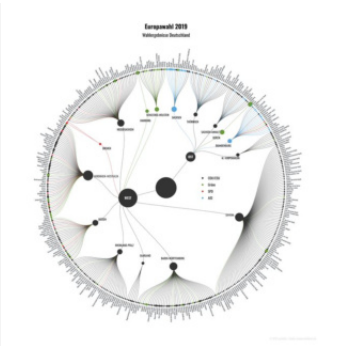

Earnings vary across unis even within subjects
Impact on men's earnings relative to the average degree
Below average
Above average
Economics
Law
Business
Maths
Creative arts
Computing
History
Psychology
Physical Sciences
Politics
Engineering
Comms
Geography
Allied to medicine
Philosophy
Languages
Architecture
Pharmacology
Technology
Medicine
Biosciences
Sociology
Sports Sciences
Agriculture
Physics
English
Education
Social care
Northumbria
Canterbury Christchuch
Cons Dance &Drama
Cambridge
Oxford
Bristol
-50%
0%
50%
100%
Source: Institute for Fiscal Studies
BBC

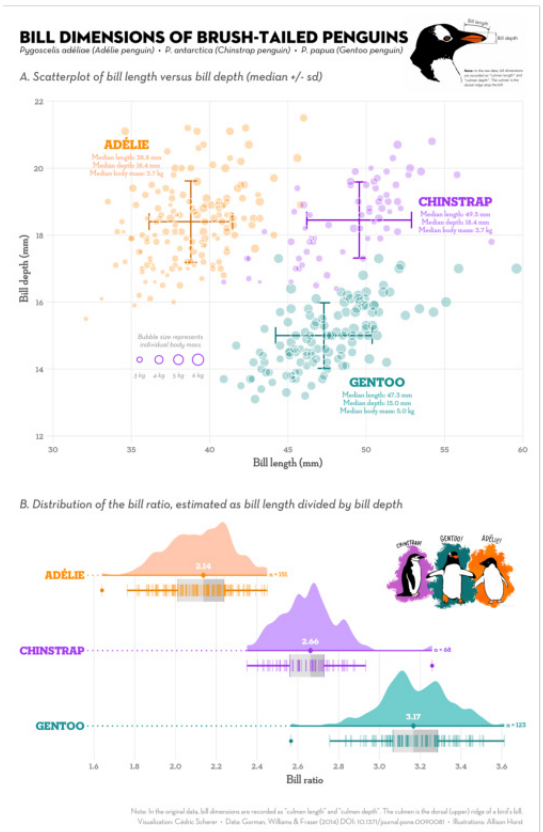


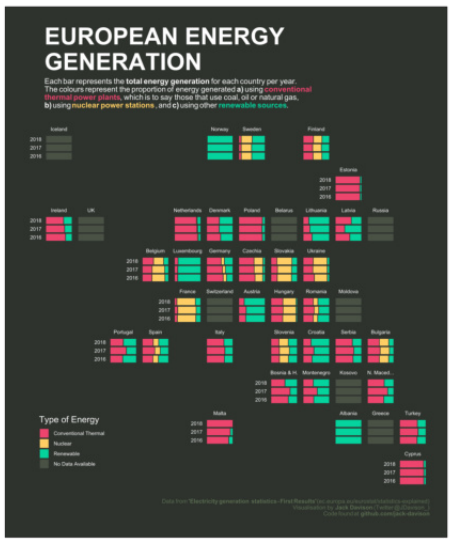
"At the **BBC data team**, we have developed an R package and an R package to make the **process of creating publication–ready graphics** in our in–house style using R's ggplot2 library **a more reproducible process**."

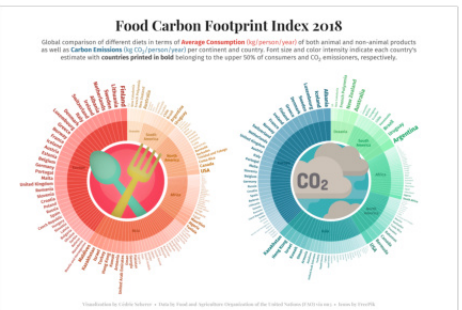


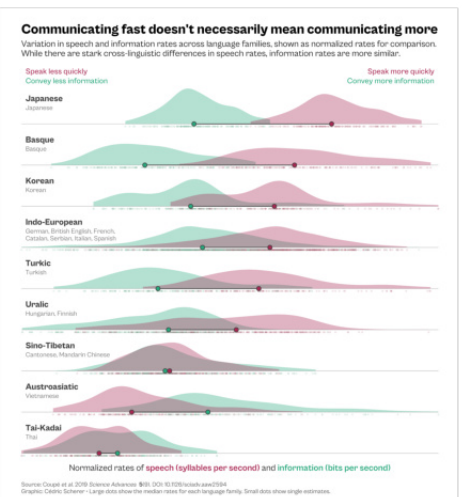


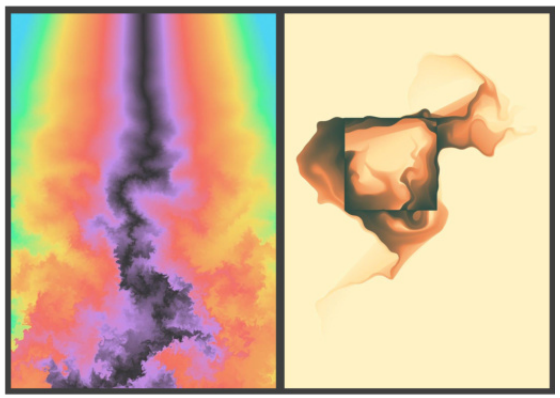


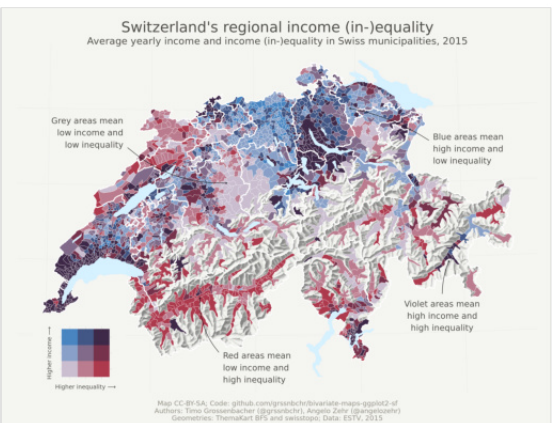






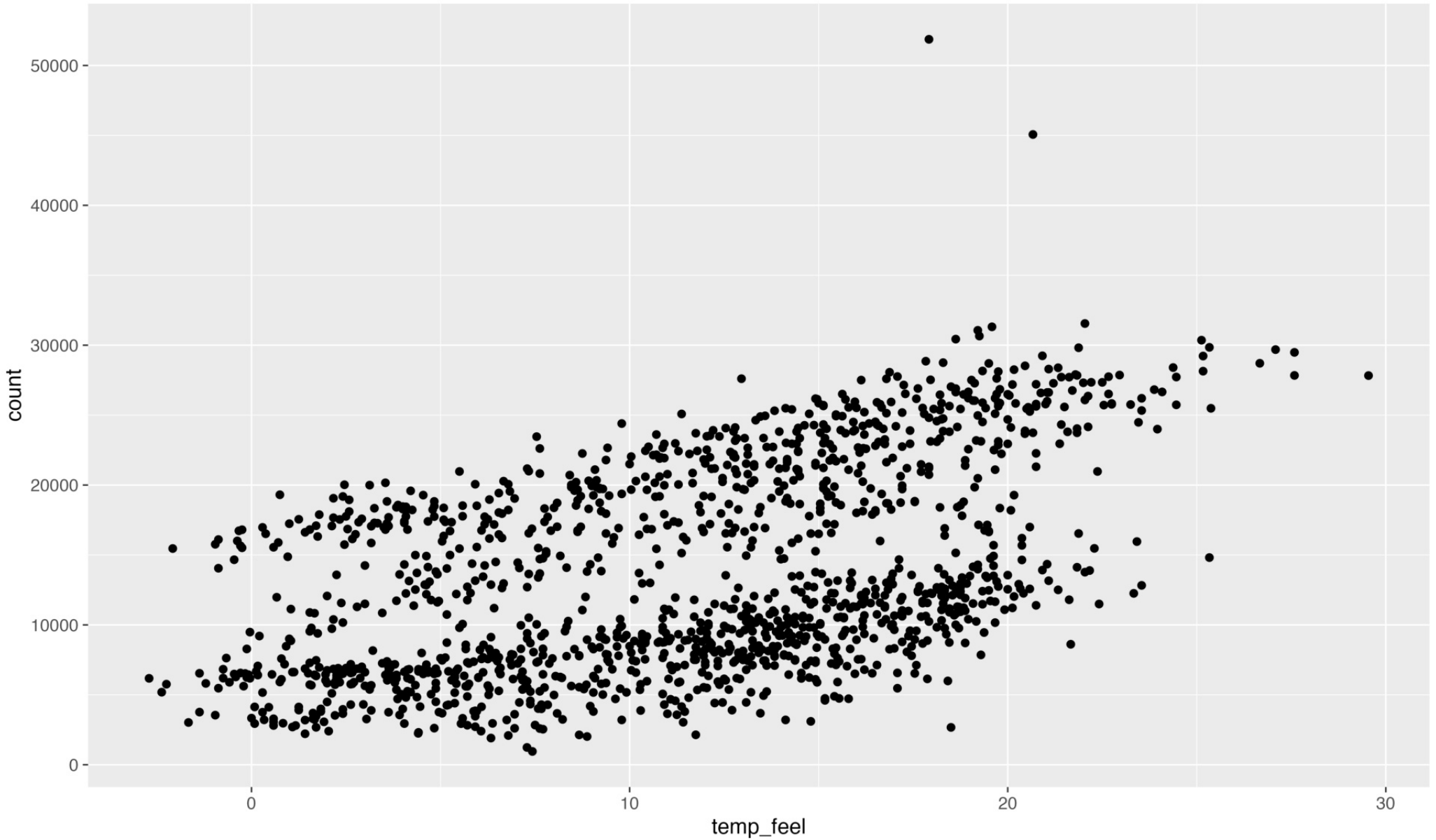




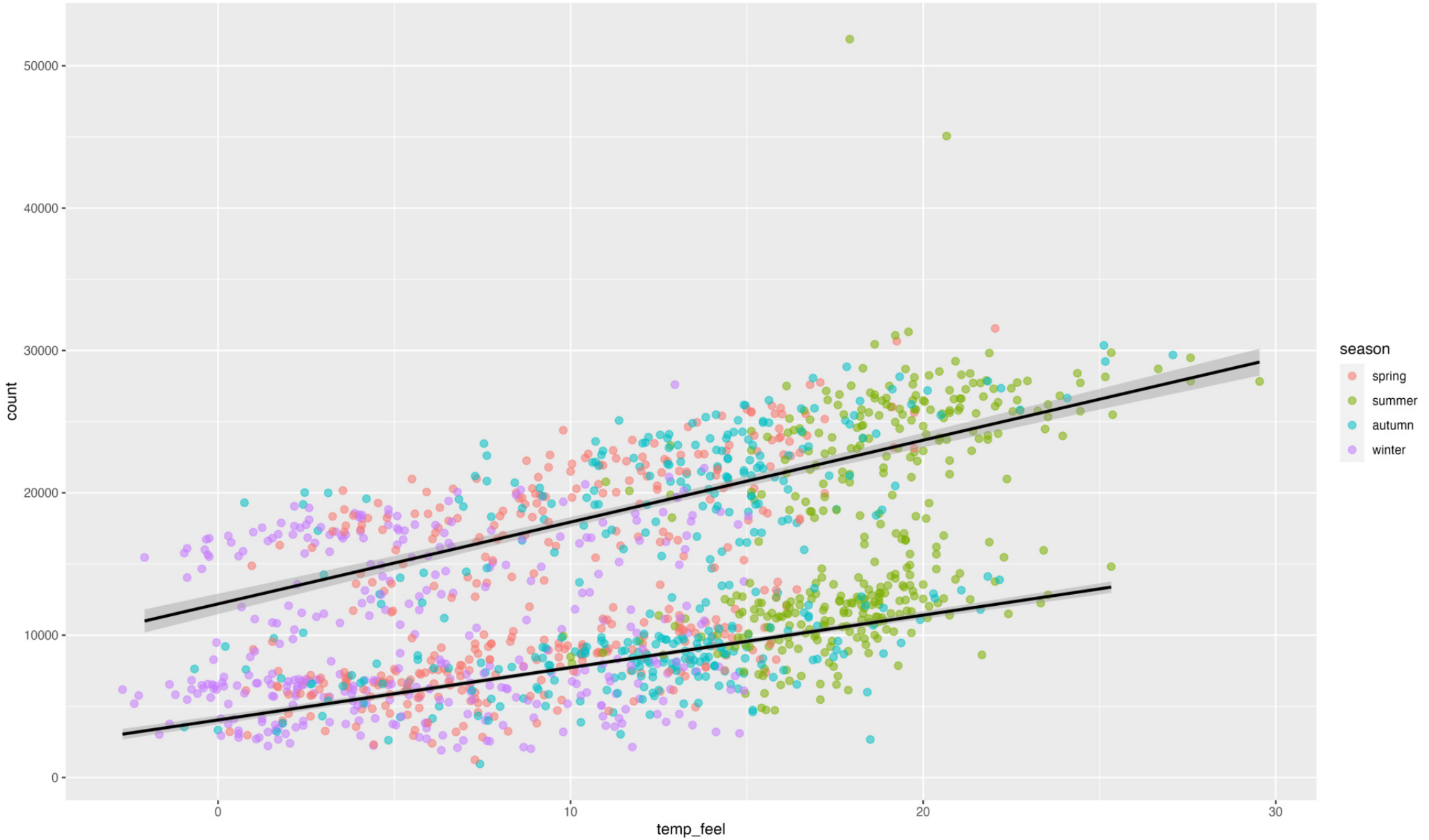


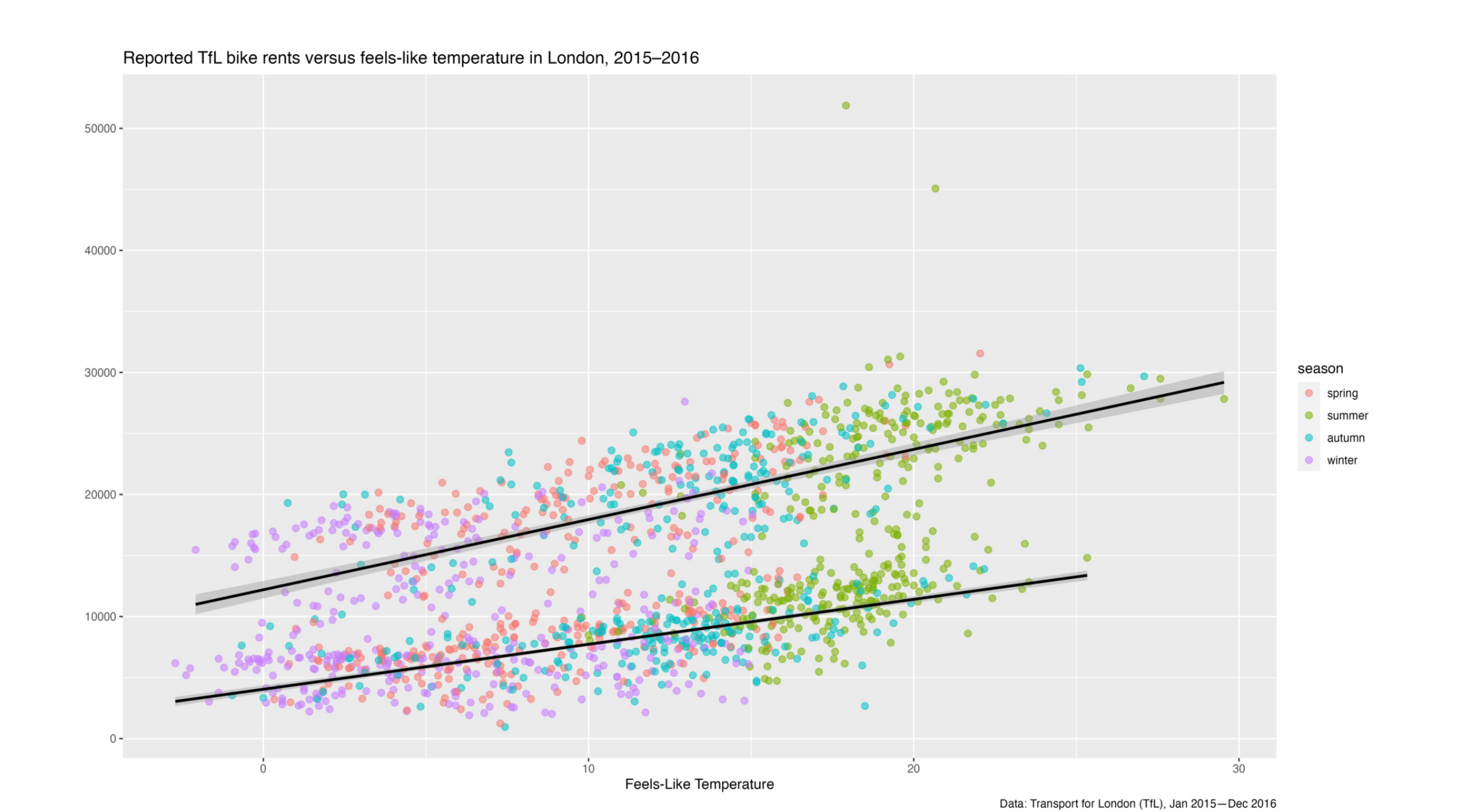
Selection of visualizations created 100% with ggplot2 by Thomas Linn Pedersen, Georgios Karamanis, Timo Gossenbacher, Torsten Sprengler, Jake Kaupp, Jack Davison, and myself.

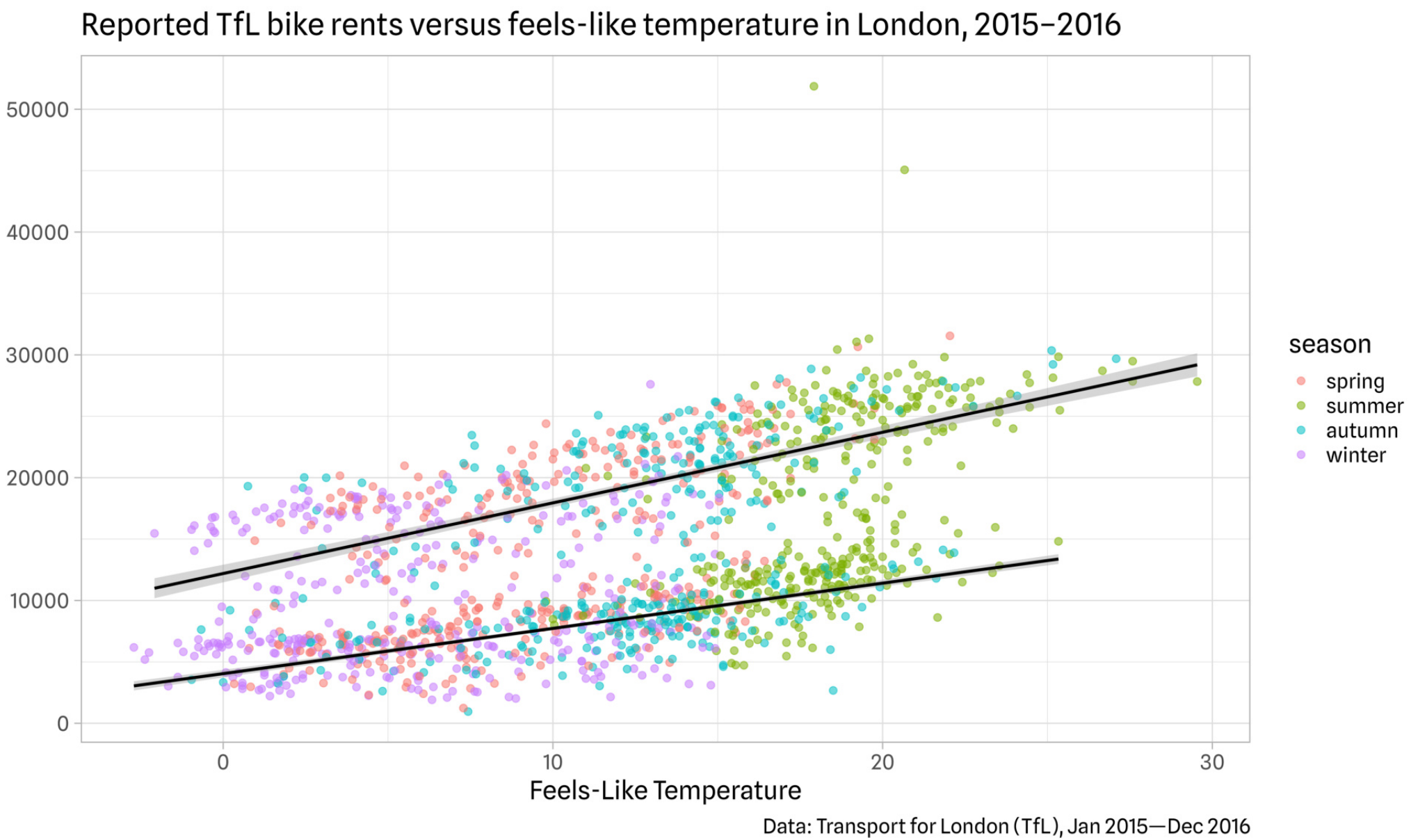
A Motivational Example

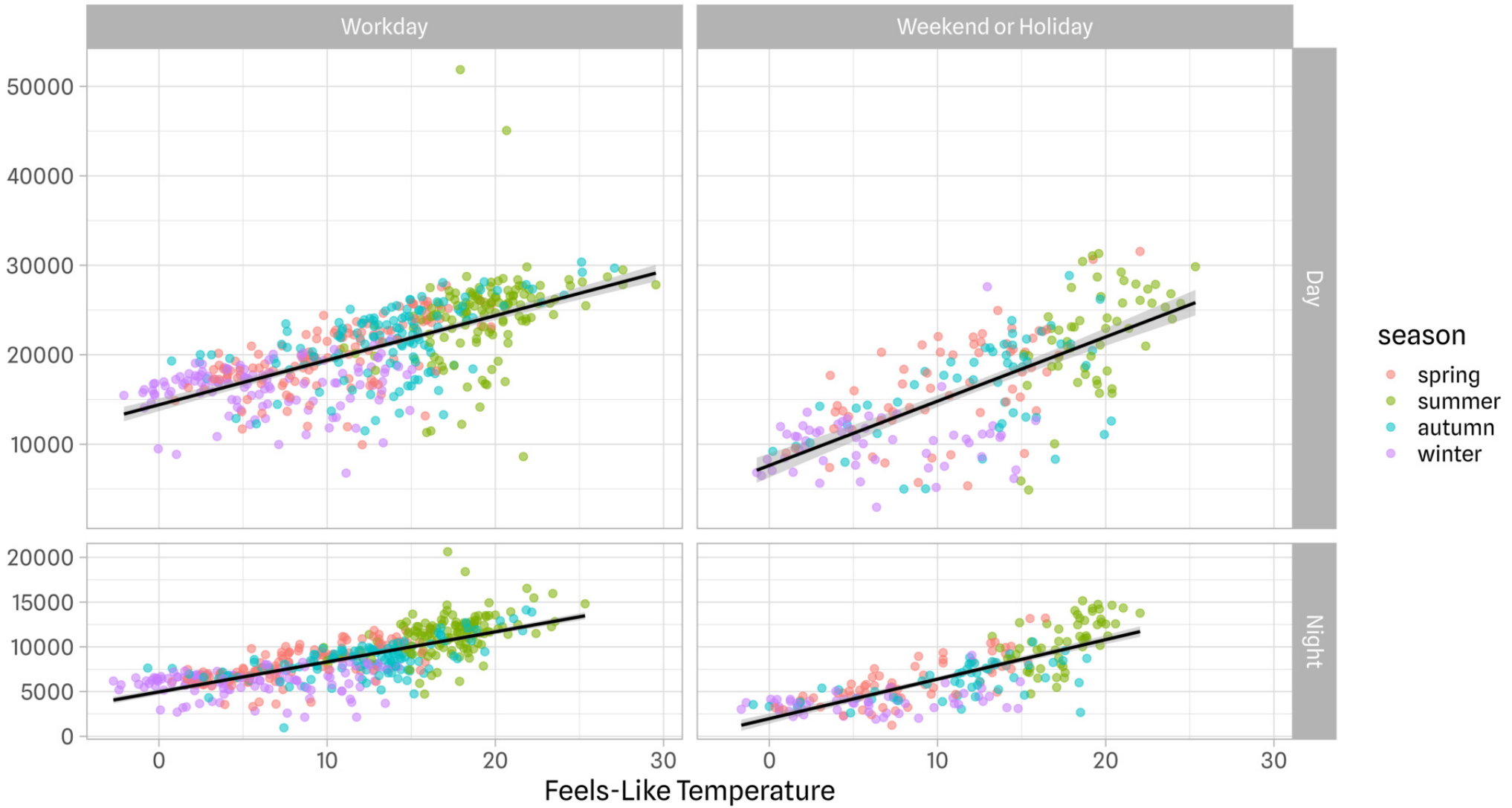

50000
40000
30000
20000
10000
0
count
0
10
20
30
temp_feel

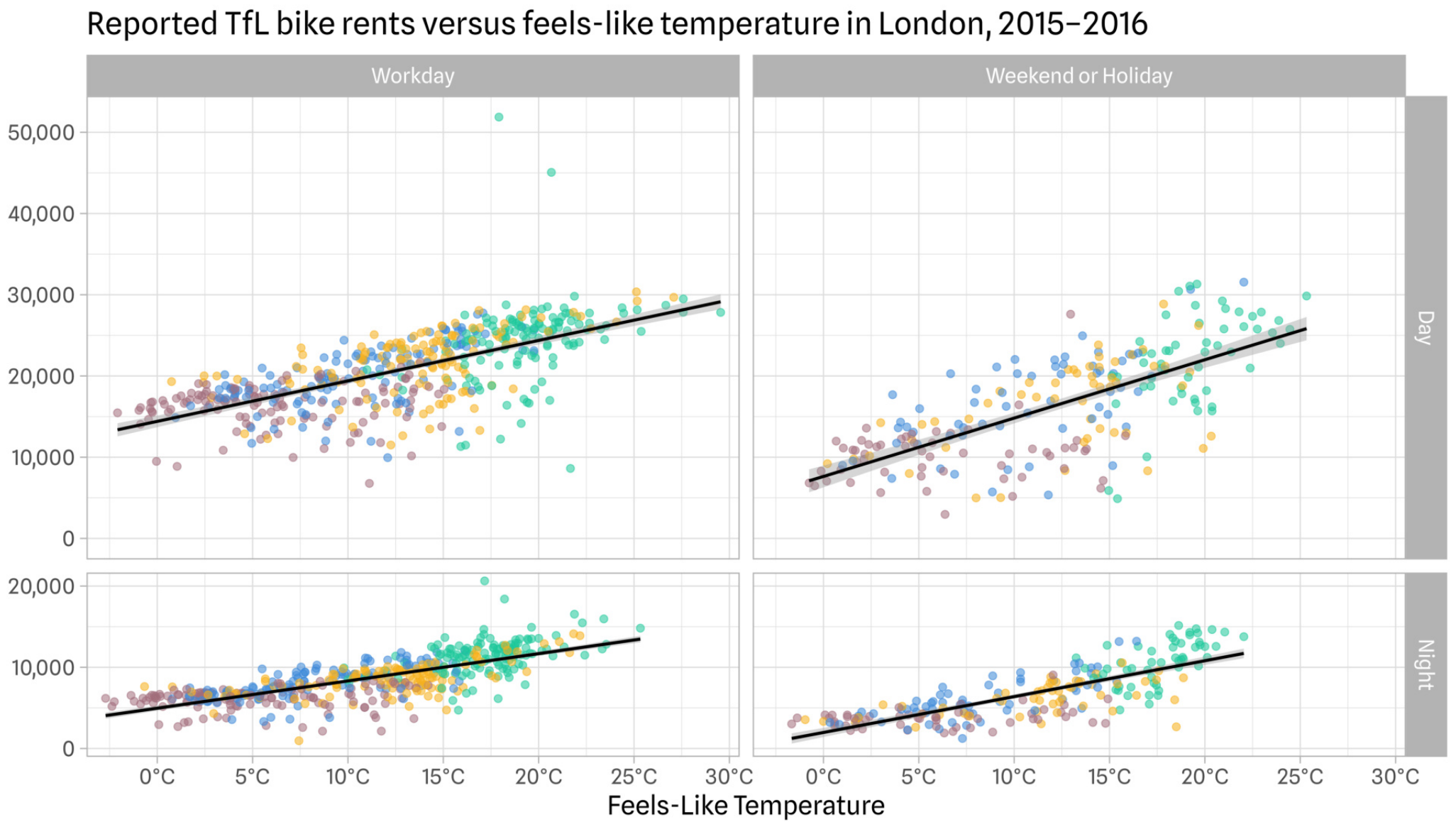

50000
40000
30000
20000
10000
0
count
0
10
20
30
temp_feel
season
spring
summer
autumn
winter


50000
40000
30000
20000
10000
0
count
0
10
20
30
temp_feel
season
spring
summer
autumn
winter

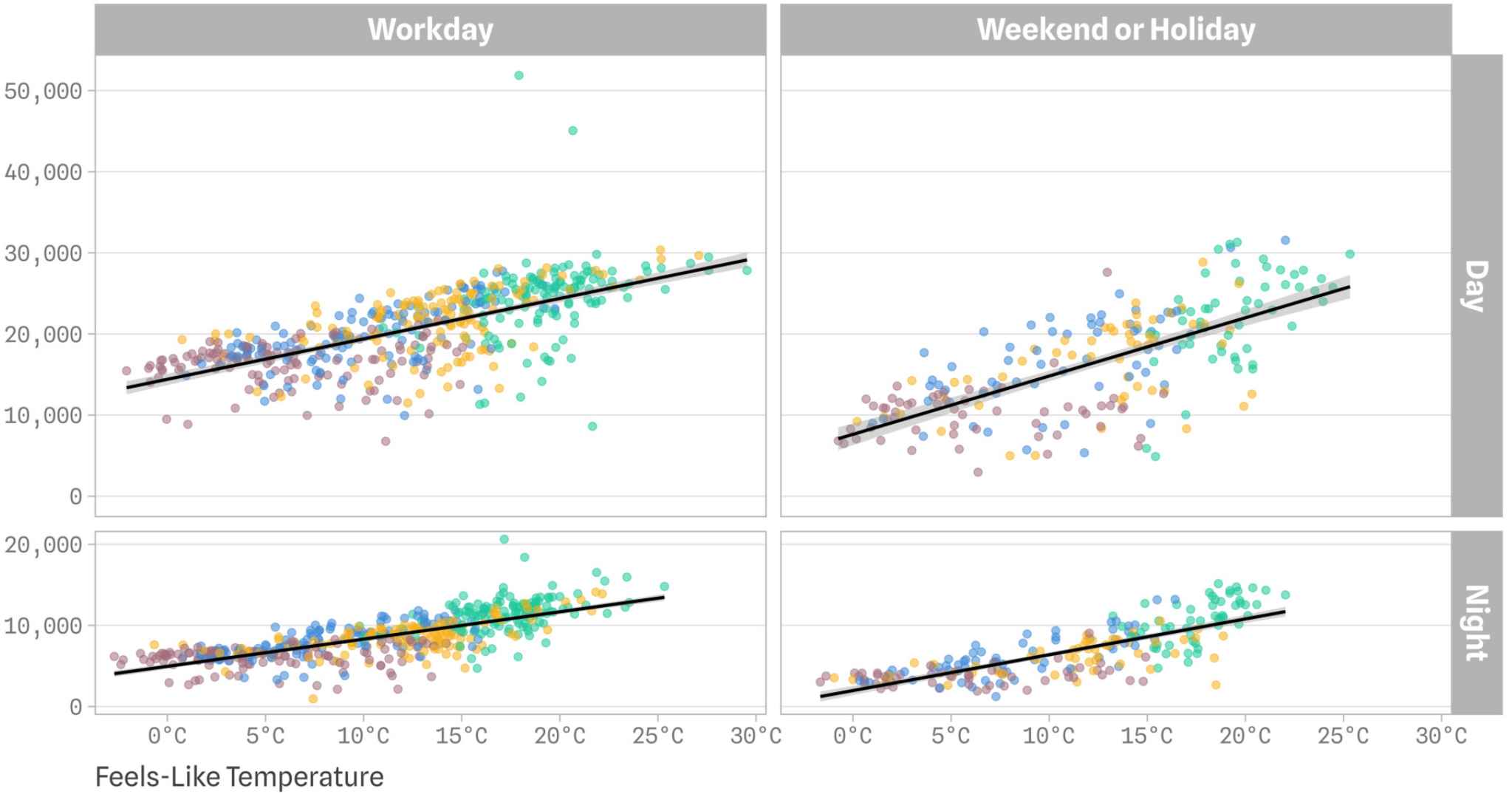

Reported TfL bike rents versus feels-like temperature in London, 2015–2016
50000
40000
30000
20000
10000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016


Reported TfL bike rents versus feels-like temperature in London, 2015–2016
50000
40000
30000
20000
10000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016


Reported TfL bike rents versus feels-like temperature in London, 2015–2016
Workday
Weekend or Holiday
Day
Night
50000
40000
30000
20000
10000
20000
15000
10000
5000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016


Reported TfL bike rents versus feels-like temperature in London, 2015–2016
Workday
Weekend or Holiday
Day
Night
50,000
40,000
30,000
20,000
10,000
0
0°C
5°C
10°C
15°C
20°C
25°C
30°C
Feels-Like Temperature
Data: Transport for London (TfL), Jan 2015—Dec 2016

Reported TfL bike rents versus feels-like temperature in London, 2015–2016



Data: Transport for London (TfL), Jan 2015—Dec 2016

Setup

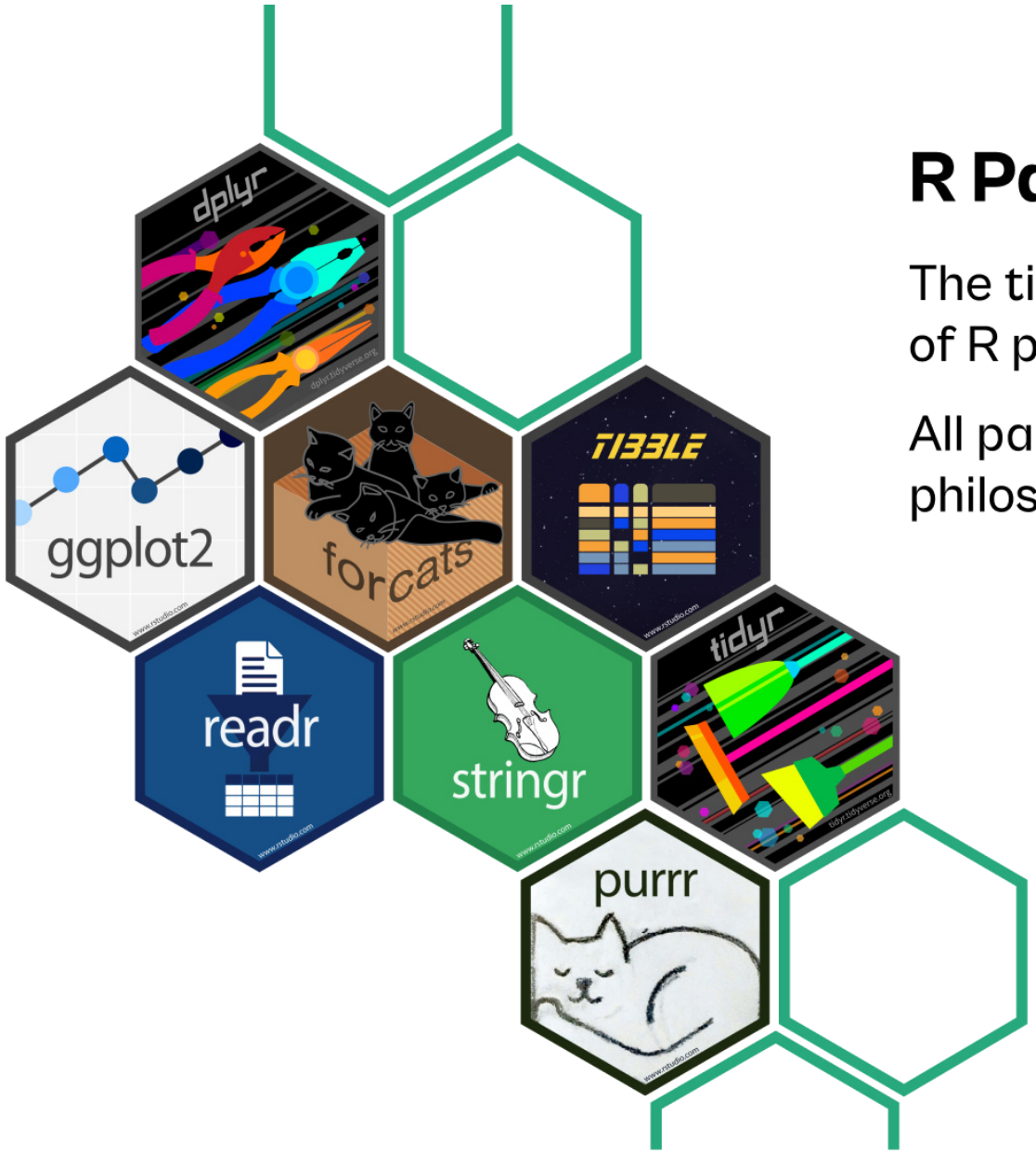
The ggplot2 Package

... is an **R package to visualize data** created by Hadley Wickham in 2005

```
# install.packages("ggplot2")
library(ggplot2)
```

... is part of the {tidyverse}

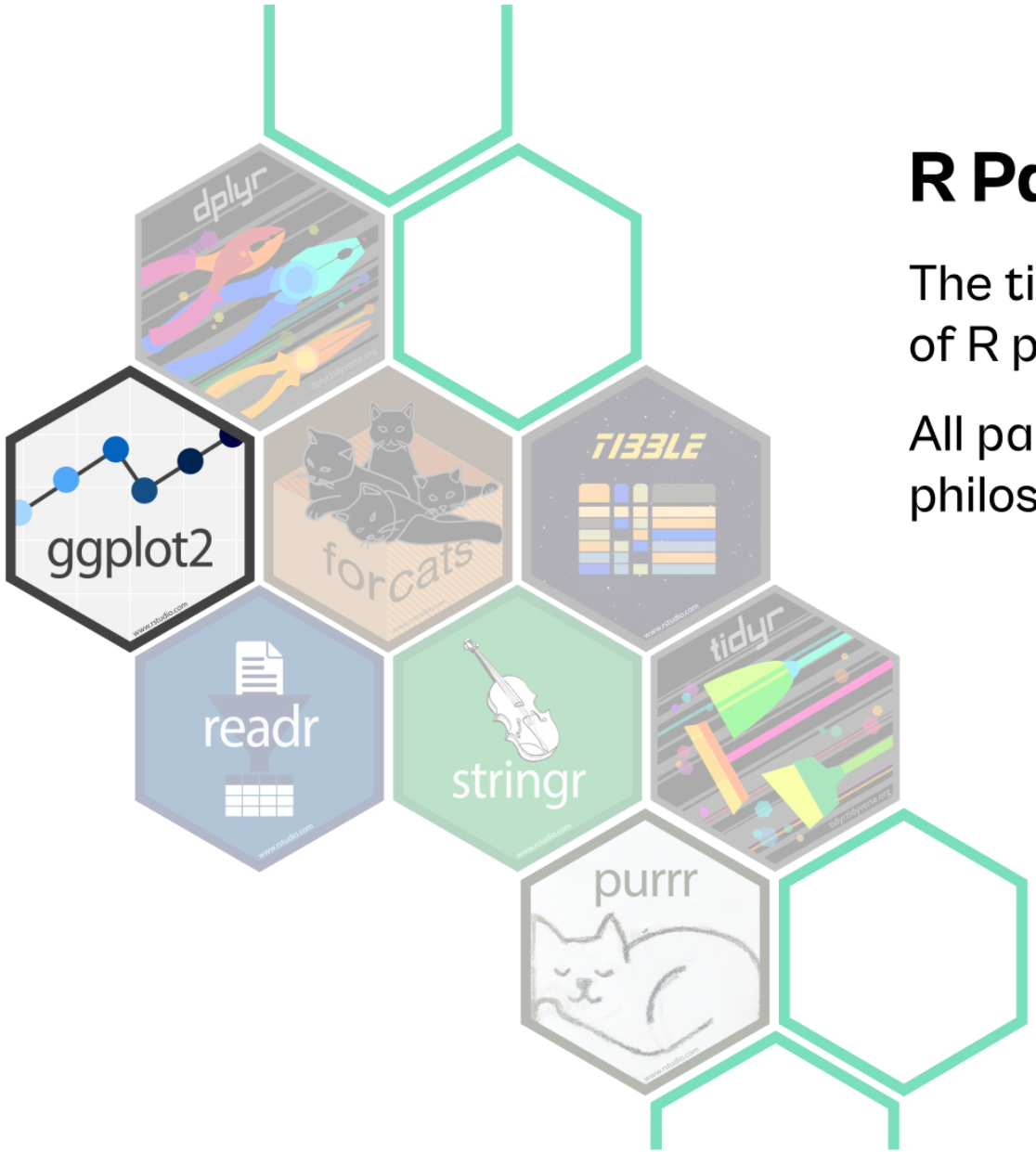
```
# install.packages("tidyverse")
library(tidyverse)
```

R Packages for Data Science

The tidyverse is an opinionated collection of R packages designed for data science.

All packages share an underlying design philosophy, grammar, and data structures.



R Packages for Data Science

The tidyverse is an opinionated collection of R packages designed for data science.

All packages share an underlying design philosophy, grammar, and data structures.

The Grammar of {ggplot2}

Cédric Scherer // Data Visualization Society // March 9, 2023

The Grammar of {ggplot2}

Component	Function	Explanation
Data	`ggplot(data)`	*The raw data that you want to visualise.*
Aesthetics	`aes()`	*Aesthetic mappings between variables and visual properties.*
Geometries	`geom_*()`	*The geometric shapes representing the data.*

The Grammar of {ggplot2}

Component	Function	Explanation
Data	`ggplot(data)`	*The raw data that you want to visualise.*
Aesthetics	`aes()`	*Aesthetic mappings between variables and visual properties.*
Geometries	`geom_*()`	*The geometric shapes representing the data.*
Statistics	`stat_*()`	*The statistical transformations applied to the data.*
Scales	`scale_*()`	*Maps between the data and the aesthetic dimensions.*
Facets	`facet_*()`	*The arrangement of the data into a grid of plots.*
Coordinate System	`coord_*()`	*Maps data into the plane of the data rectangle.*
Visual Themes	`theme()` and `theme_*()`	*The overall visual defaults of a plot.*

The Data

Bike sharing counts in London, UK, powered by TfL Open Data

- covers the years 2015 and 2016
- incl. weather data acquired from freemeteo.com
- prepared by Hristo Mavrodiev for Kaggle
- further modification by myself

```
bikes <- readr::read_csv(
  "./data/london-bikes-custom.csv",
  ## or: "https://cedricscherer.com/data/london-bikes-custom.csv"
  col_types = "Dcfffillldddddc"
)

bikes$season <- forcats::fct_inorder(bikes$season)
```


Variable	Description	Class
date	Date encoded as `YYYY-MM-DD`	date
day_night	`day` (6:00am–5:59pm) or `night` (6:00pm–5:59am)	character
year	`2015` or `2016`	factor
month	`1` (January) to `12` (December)	factor
season	`winter`, `spring`, `summer`, or `autumn`	factor
count	Sum of reported bikes rented	integer
is_workday	`TRUE` being Monday to Friday and no bank holiday	logical
is_weekend	`TRUE` being Saturday or Sunday	logical
is_holiday	`TRUE` being a bank holiday in the UK	logical
temp	Average air temperature (˚C)	double
temp_feel	Average feels like temperature (˚C)	double
humidity	Average air humidity (%)	double
wind_speed	Average wind speed (km/h)	double
weather_type	Most common weather type	character

Fundamentals of {ggplot2}

Cédric Scherer // Data Visualization Society // March 9, 2023

ggplot2::ggplot()

ggplot: Create a new ggplot

Description

`ggplot()` initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

Usage

```
ggplot(data = NULL, mapping = aes(), ..., environment = parent.frame())
```

Arguments

`data`	Default dataset to use for plot. If not already a data.frame, will be converted to one by `fortify()`. If not specified, must be supplied in each layer added to the plot.
`mapping`	Default list of aesthetic mappings to use for plot. If not specified, must be supplied in each layer added to the plot.
`...`	Other arguments passed on to methods. Not currently used.
`environment`	DEPRECATED. Used prior to tidy evaluation.

Details

`ggplot()` is used to construct the initial plot object, and is almost always followed by `+` to add component to the plot. There are three common ways to invoke `ggplot()`:

- `ggplot(df, aes(x, y, other aesthetics))`
- `ggplot(df)`
- `ggplot()`

Data

```
1  ggplot(data = bikes)
```

Aesthetic Mapping

= link variables to graphical properties

- positions (`x`, `y`)
- colors (`color`, `fill`)
- shapes (`shape`, `linetype`)
- size (`size`)
- transparency (`alpha`)
- groupings (`group`)

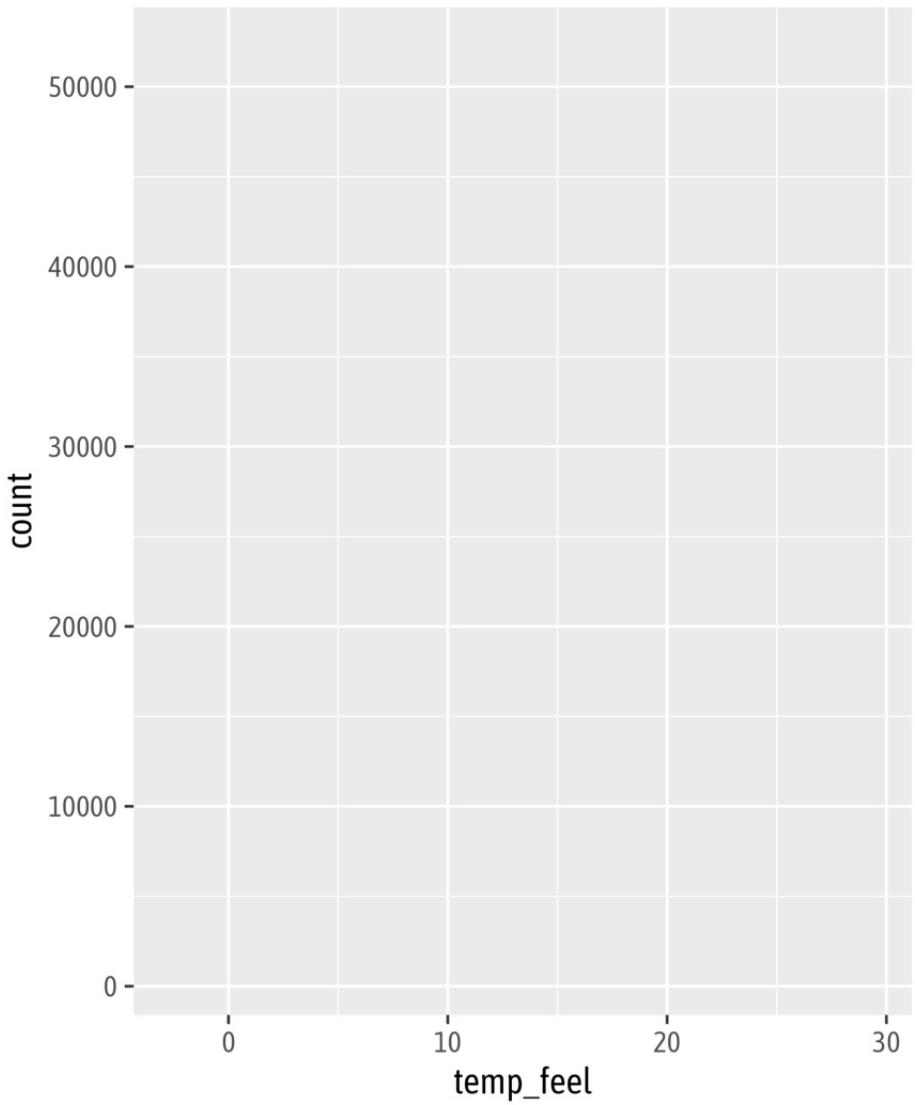
Aesthetic Mapping

= link variables to graphical properties

- feels-like temperature ⇄ x
- reported bike shares ⇄ y
- season ⇄ color
- year ⇄ shape
- ...

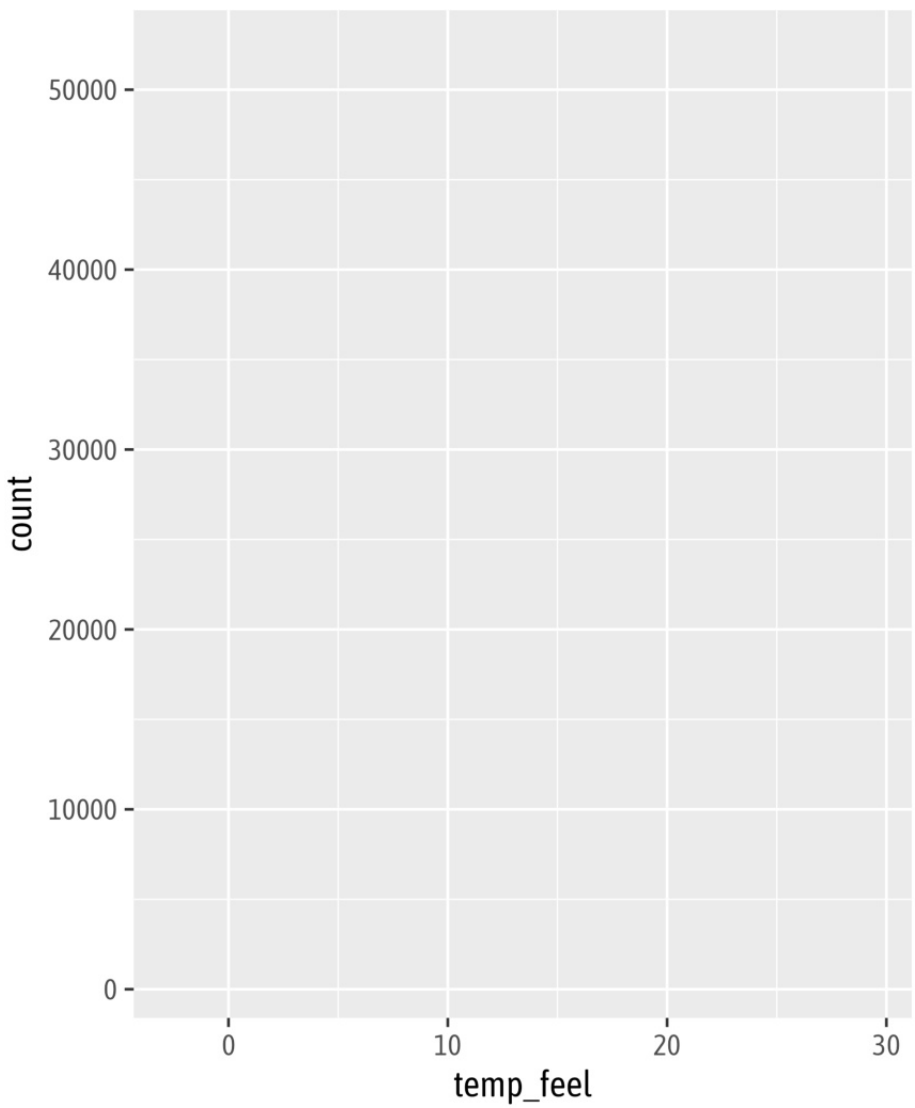
Aesthetic Mapping

```
1 ggplot(data = bikes) +
2   aes(x = temp_feel, y = count)
```



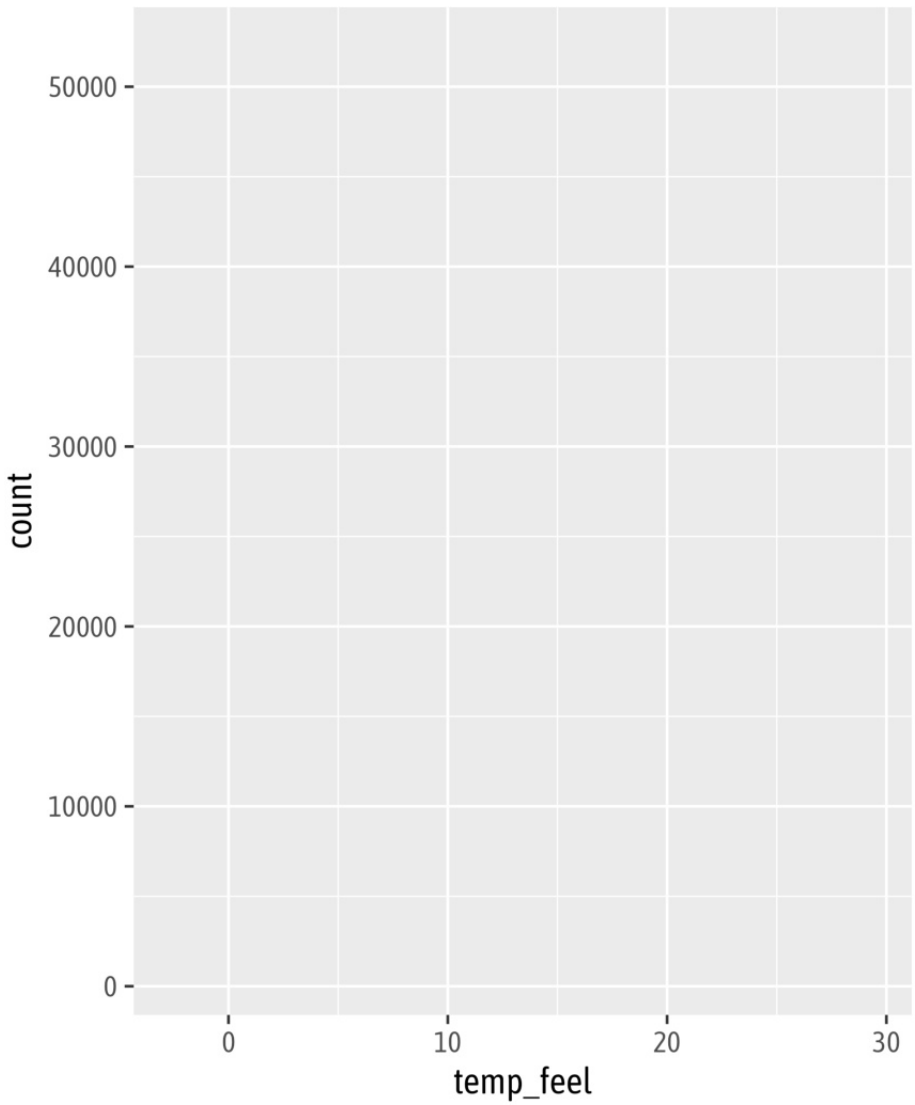
Aesthetic Mapping

```
1 ggplot(
2   data = bikes,
3   mapping = aes(x = temp_feel, y = count)
4 )
```



Aesthetic Mapping

```
1 ggplot(
2   bikes,
3   aes(x = temp_feel, y = count)
4 )
```



Geometrical Layers

Cédric Scherer // Data Visualization Society // March 9, 2023

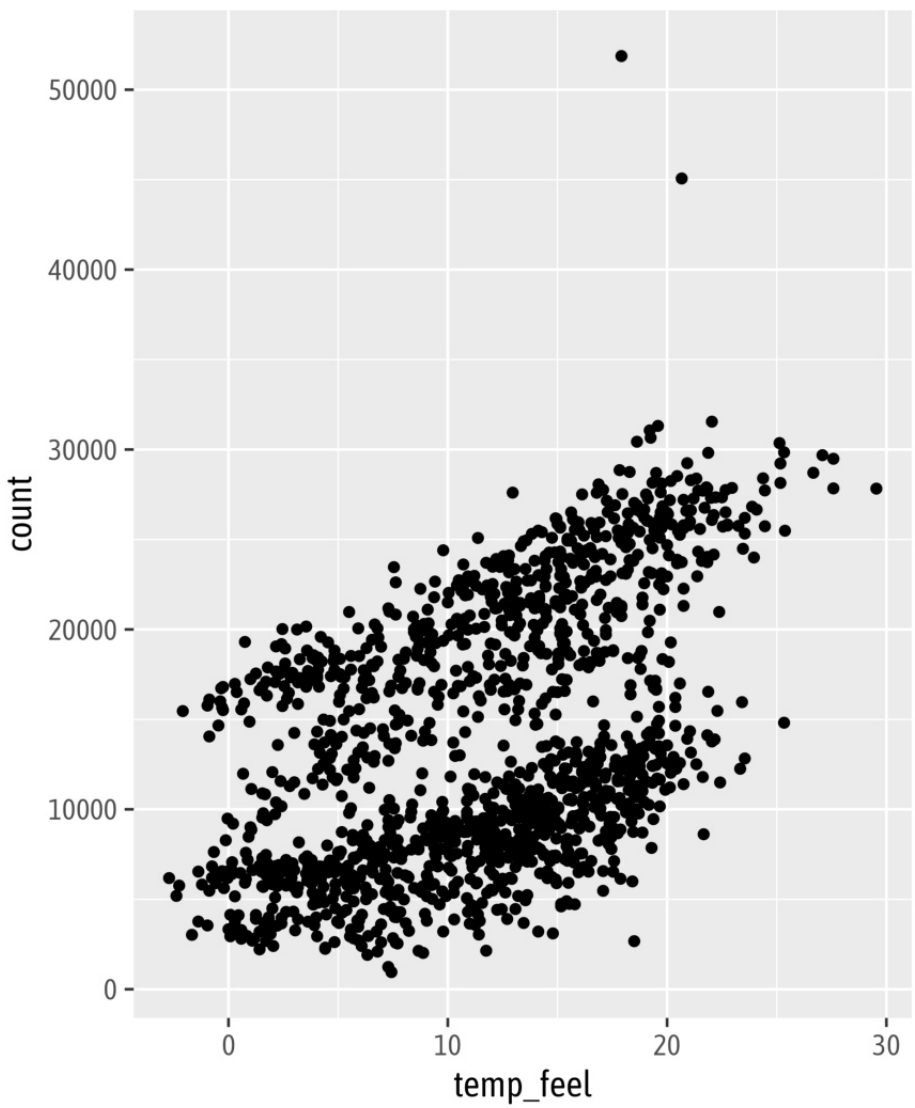
Geometries

= interpret aesthetics as graphical representations

- points
- lines
- polygons
- text labels
- ...

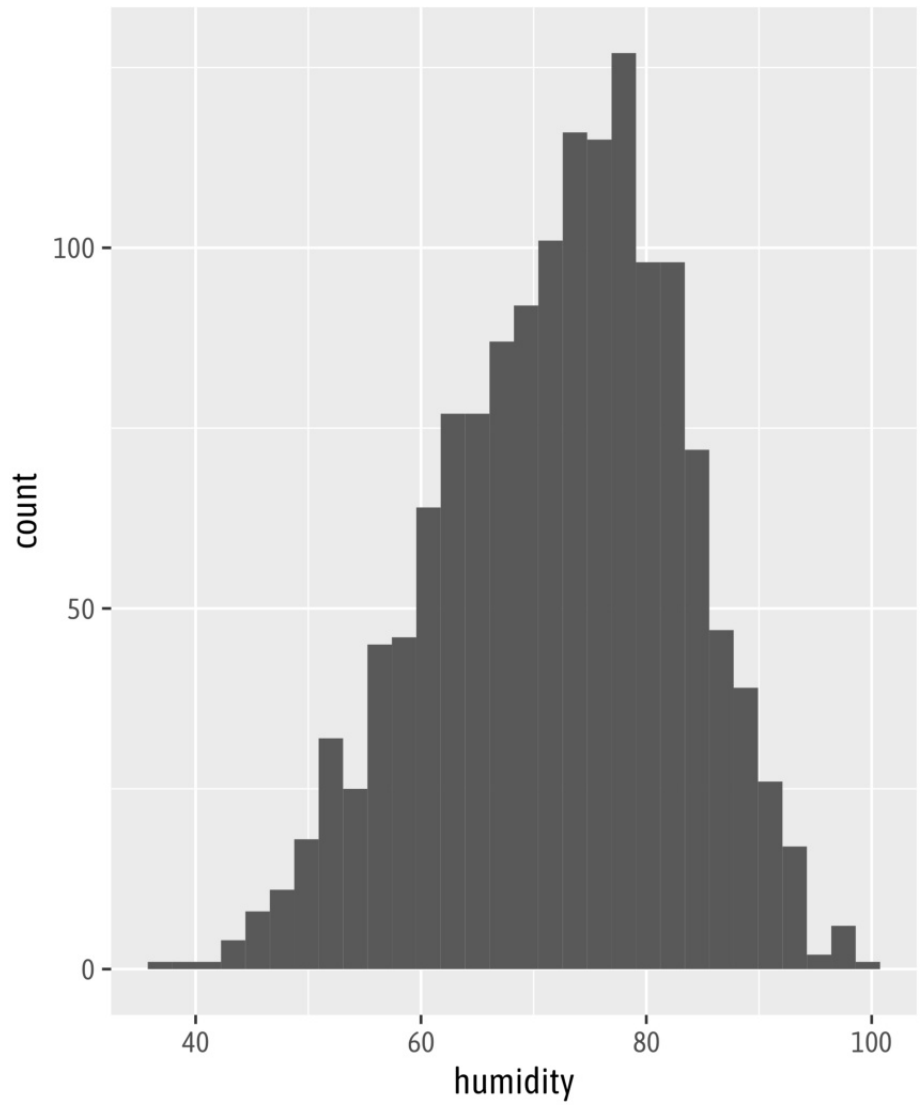
Geometries

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



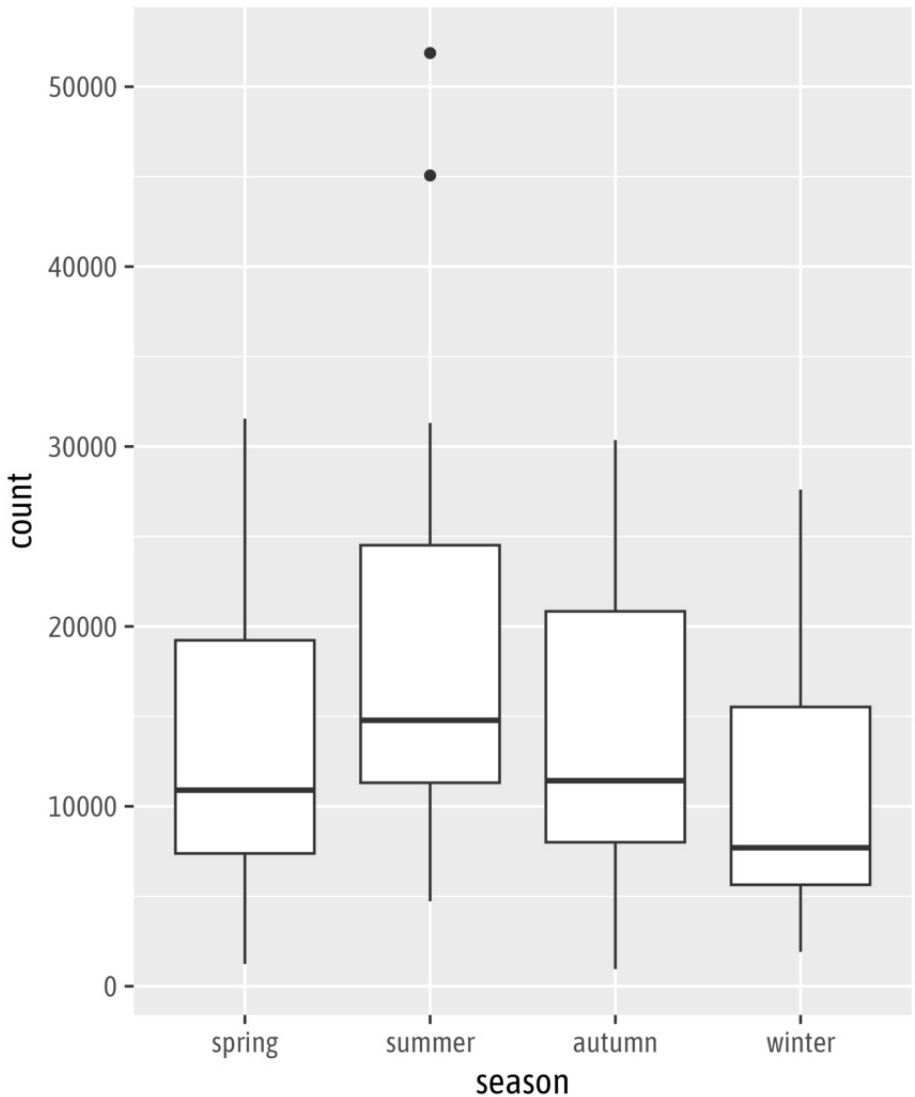
Geometries

```
1 ggplot(
2     bikes,
3     aes(x = humidity)
4   ) +
5   geom_histogram()
```



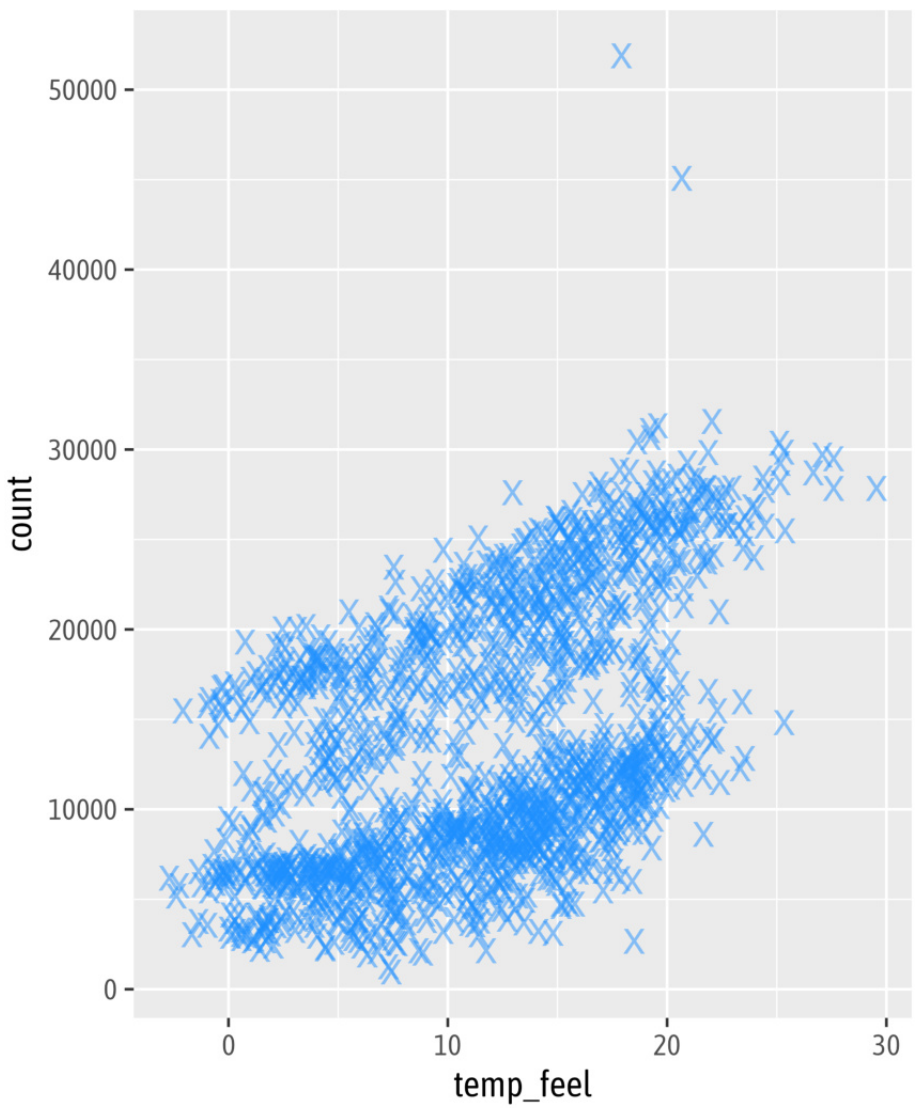
Geometries

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



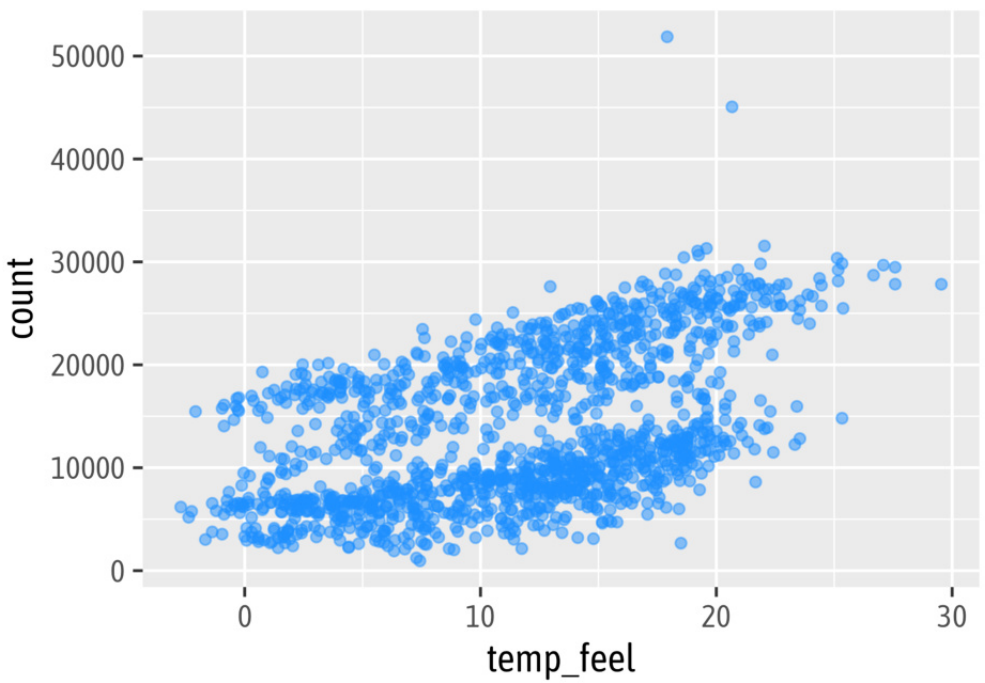
Visual Properties of Layers

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point(
    color = "dodgerblue",
    alpha = .5,
    shape = "X",
    stroke = 1,
    size = 4
  )
```

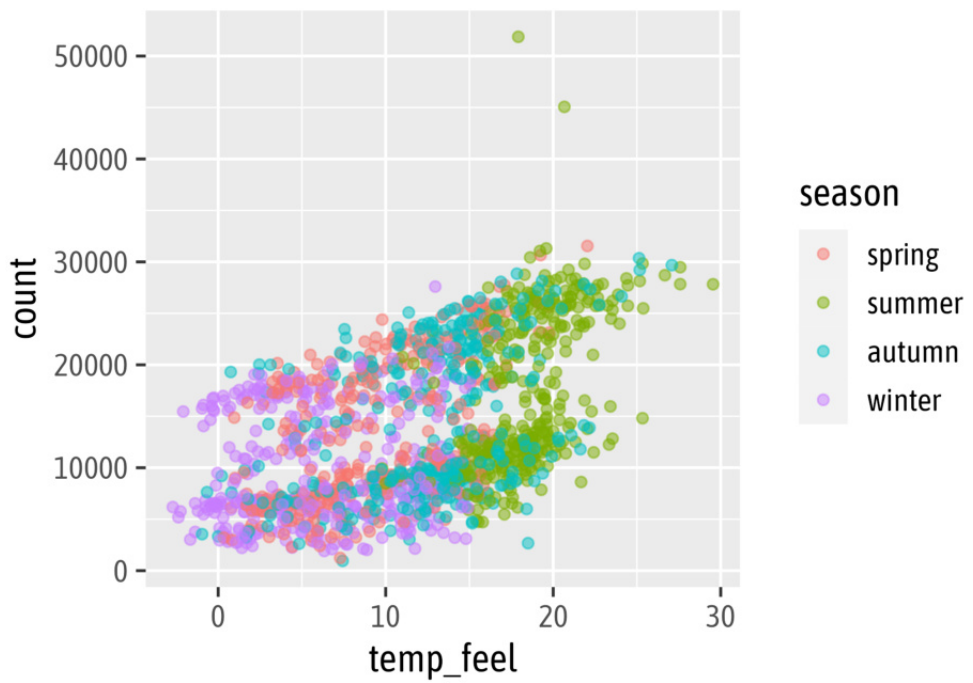


Setting vs Mapping of Visual Properties

```r
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point(
    color = "dodgerblue",
    alpha = .5
  )
```

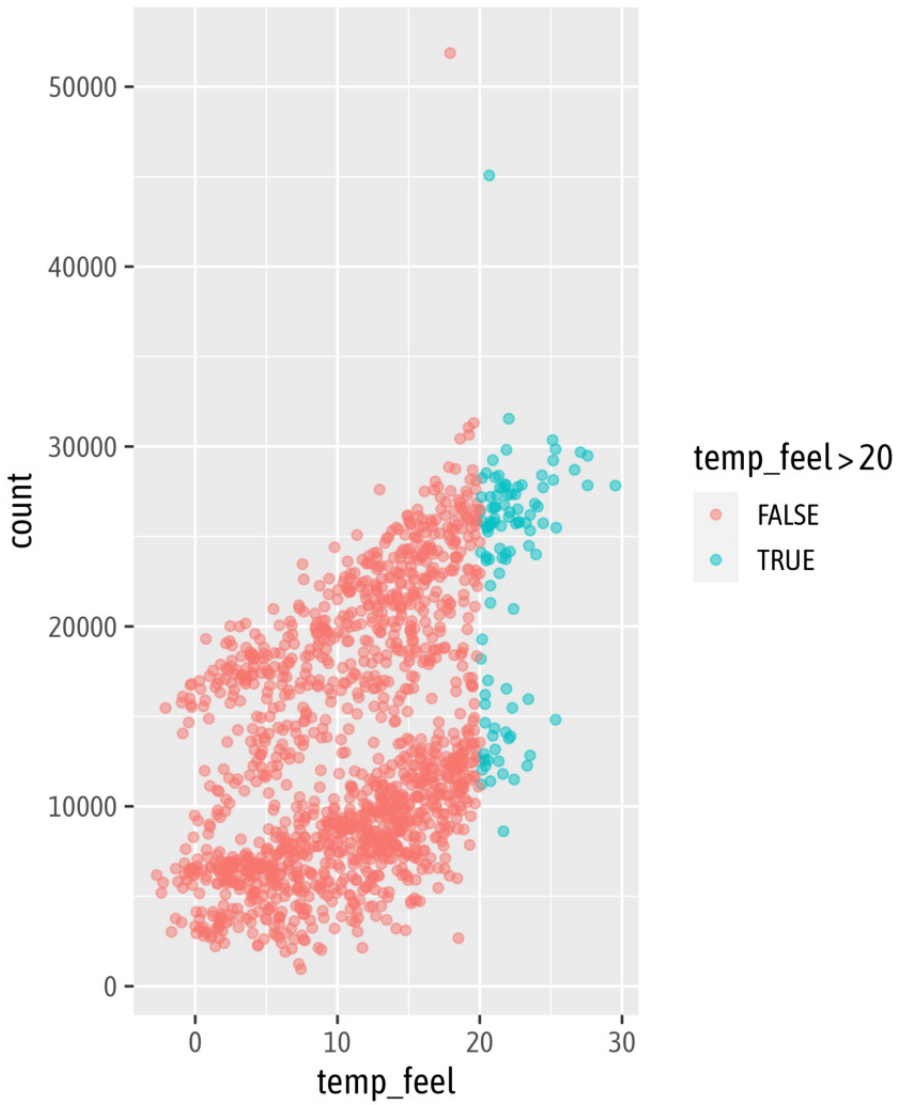



```r
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point(
    aes(color = season),
    alpha = .5
  )
```



Mapping Expressions

```
1 ggplot(
2     bikes,
3     aes(x = temp_feel, y = count)
4   ) +
5   geom_point(
6     aes(color = temp_feel > 20),
7     alpha = .5
8   )
```

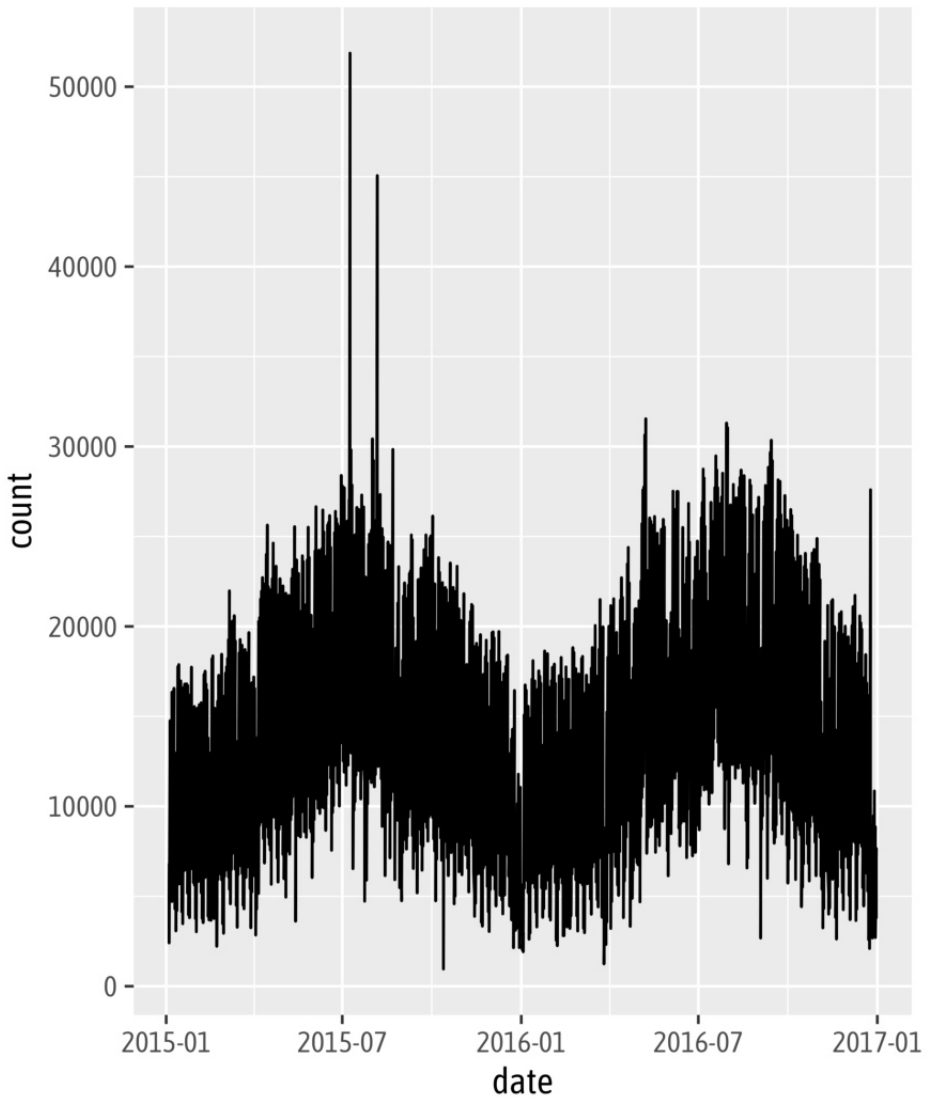


Your Turn

- **Create a chart showing a time series of reported bike shares**
  - What is the difference between `geom_line()` and `geom_path()`?
  - Map the color of the lines to `day_night`.
  - Add points for each observation, colored by the same variable.
  - Turn the points into diamonds.

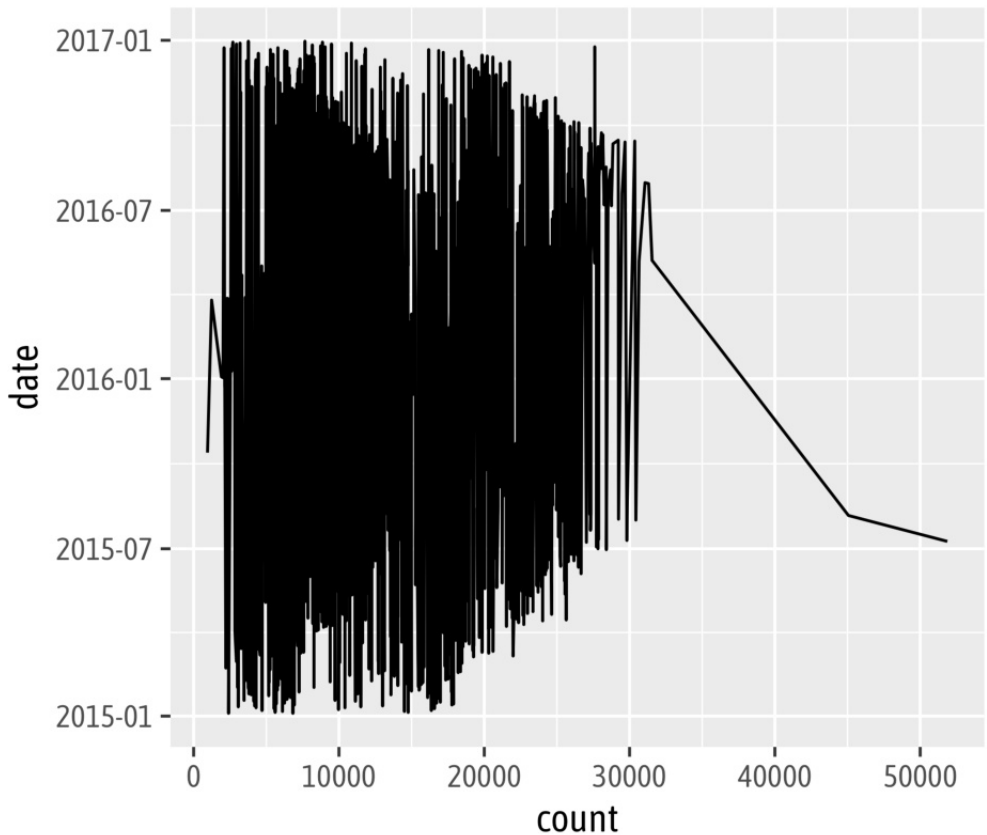
Solution Exercise

```
1 ggplot(
2     bikes,
3     aes(x = date, y = count)
4   ) +
5   geom_line()
```

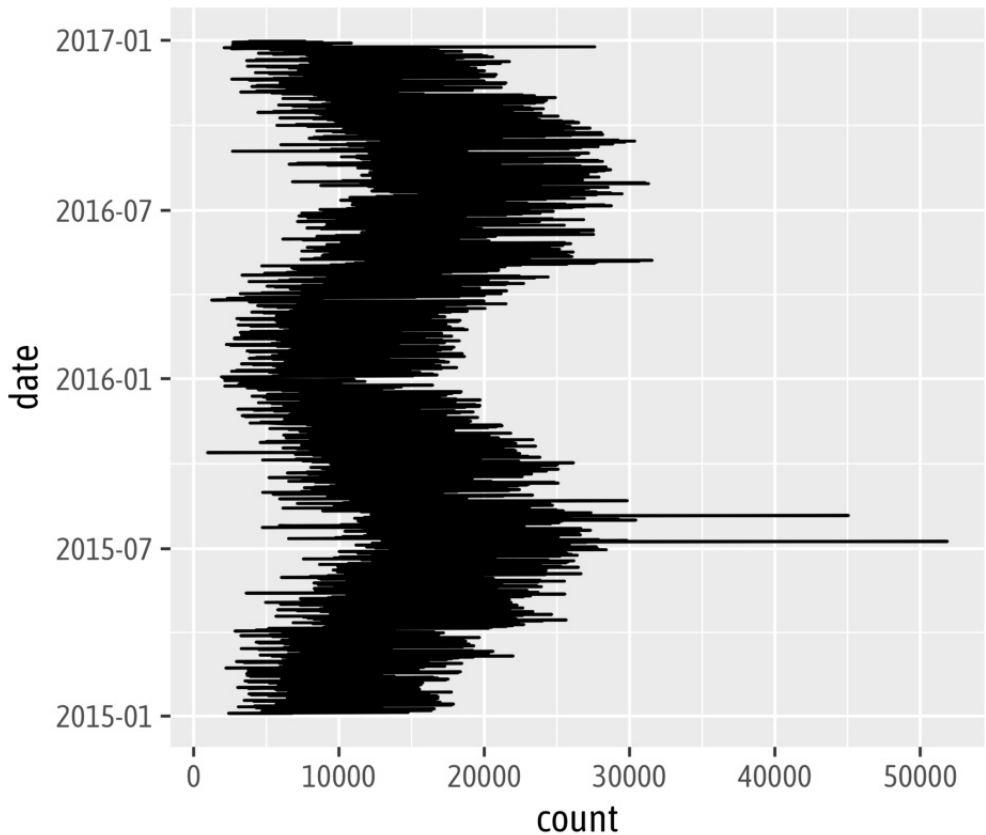


Solution Exercise

```
1  ggplot(
2      bikes,
3      aes(x = count, y = date)
4    ) +
5    geom_line()
```

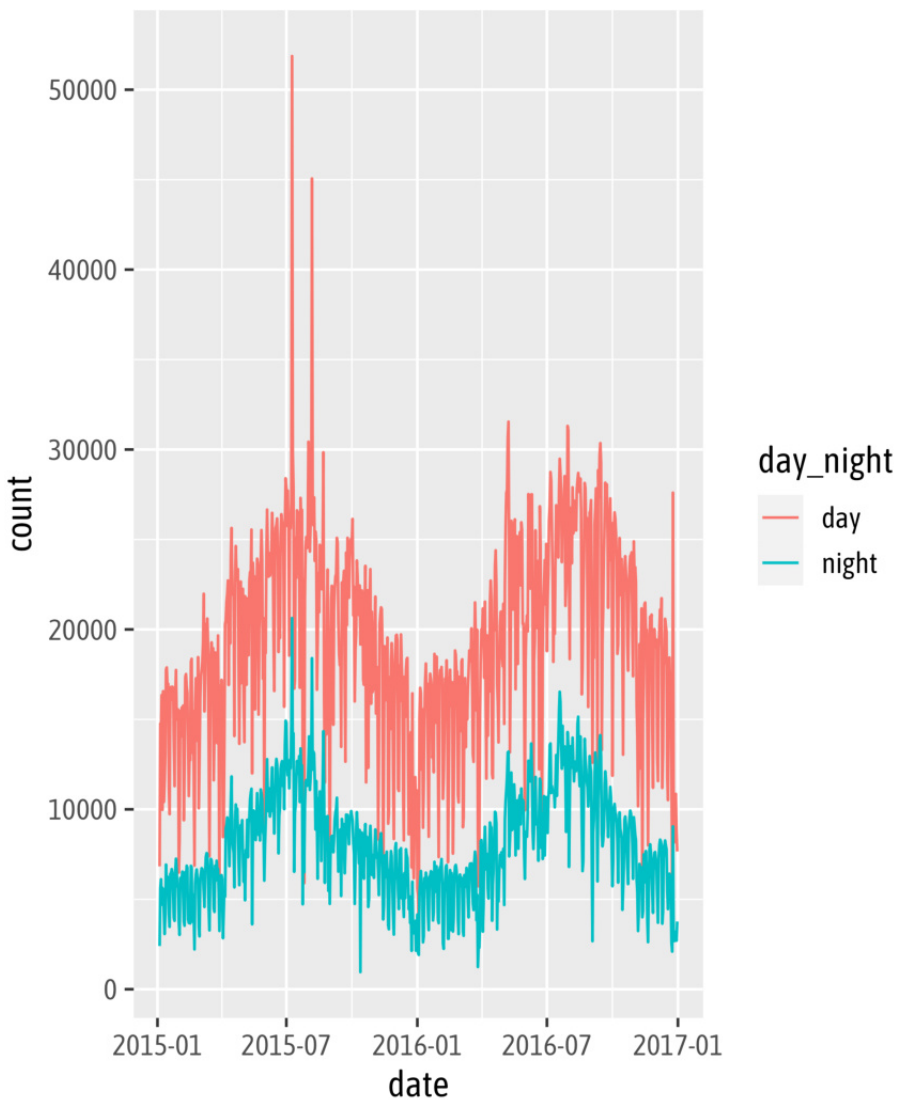



```
1  ggplot(
2      bikes,
3      aes(x = count, y = date)
4    ) +
5    geom_path()
```



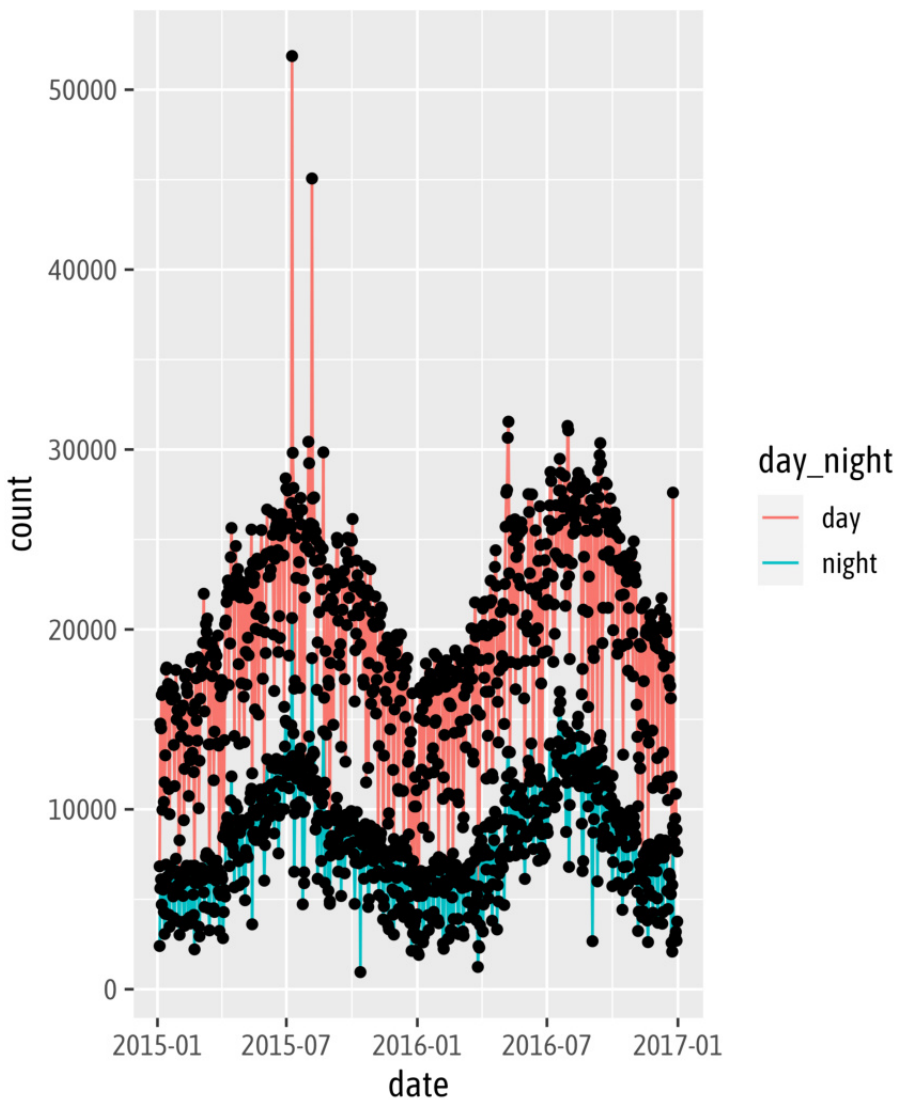
Solution Exercise

```
1 ggplot(
2     bikes,
3     aes(x = date, y = count)
4   ) +
5   geom_line(
6     aes(color = day_night)
7   )
```



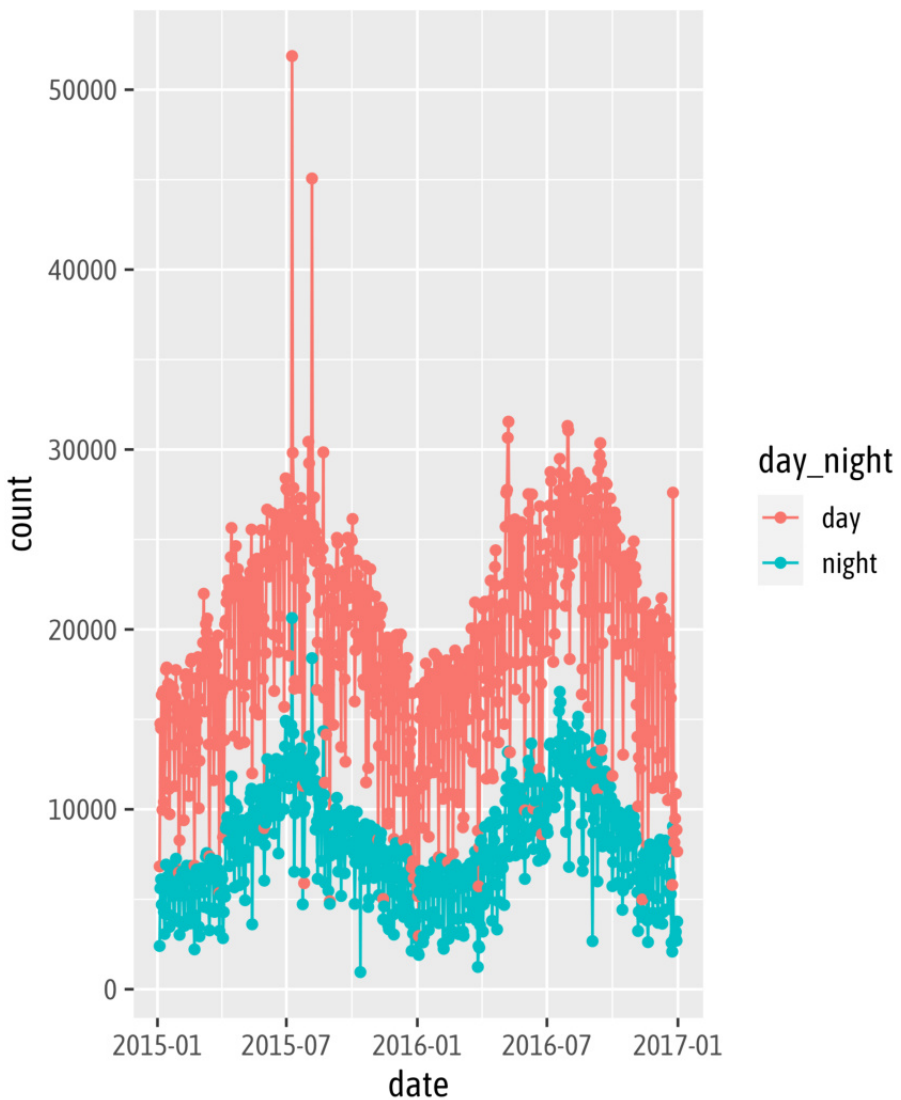
Solution Exercise

```
1 ggplot(
2     bikes,
3     aes(x = date, y = count)
4   ) +
5   geom_line(
6     aes(color = day_night)
7   ) +
8   geom_point()
```



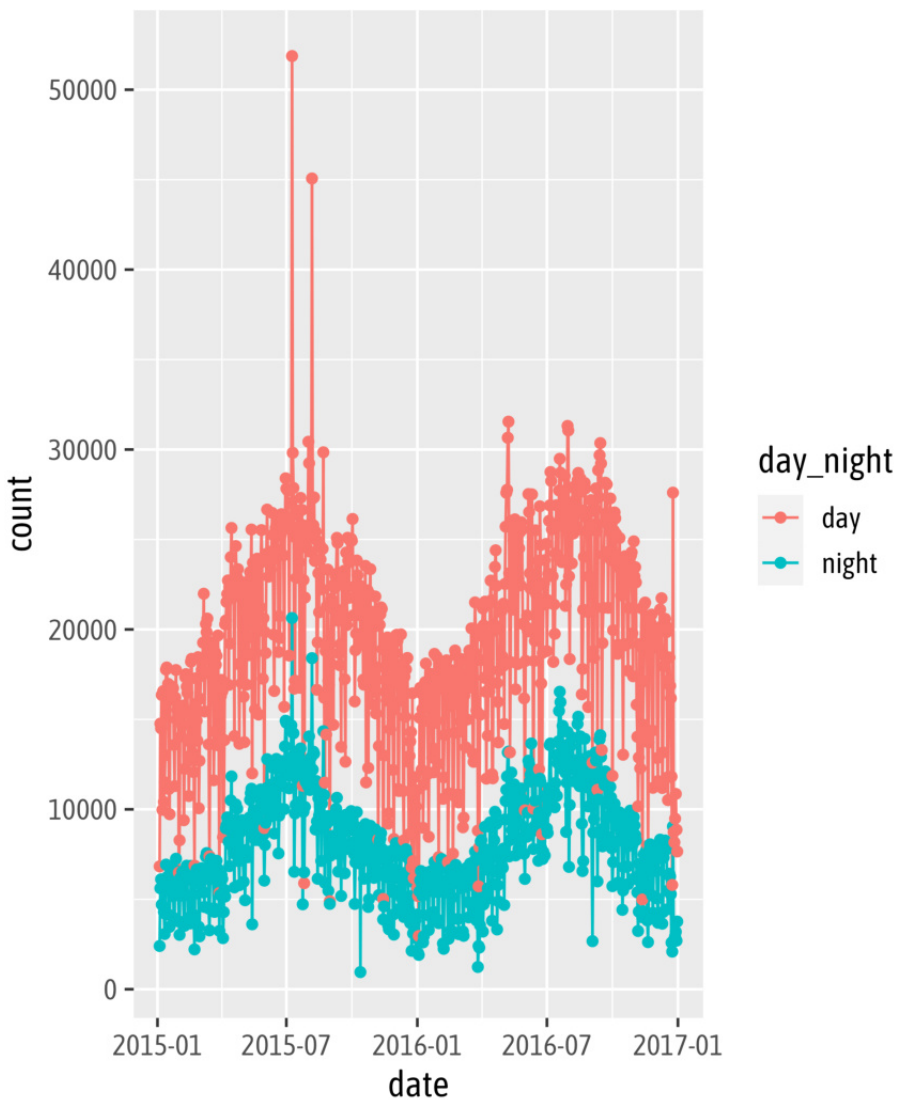
Solution Exercise

```
ggplot(
    bikes,
    aes(x = date, y = count)
  ) +
  geom_line(
    aes(color = day_night)
  ) +
  geom_point(
    aes(color = day_night)
  )
```



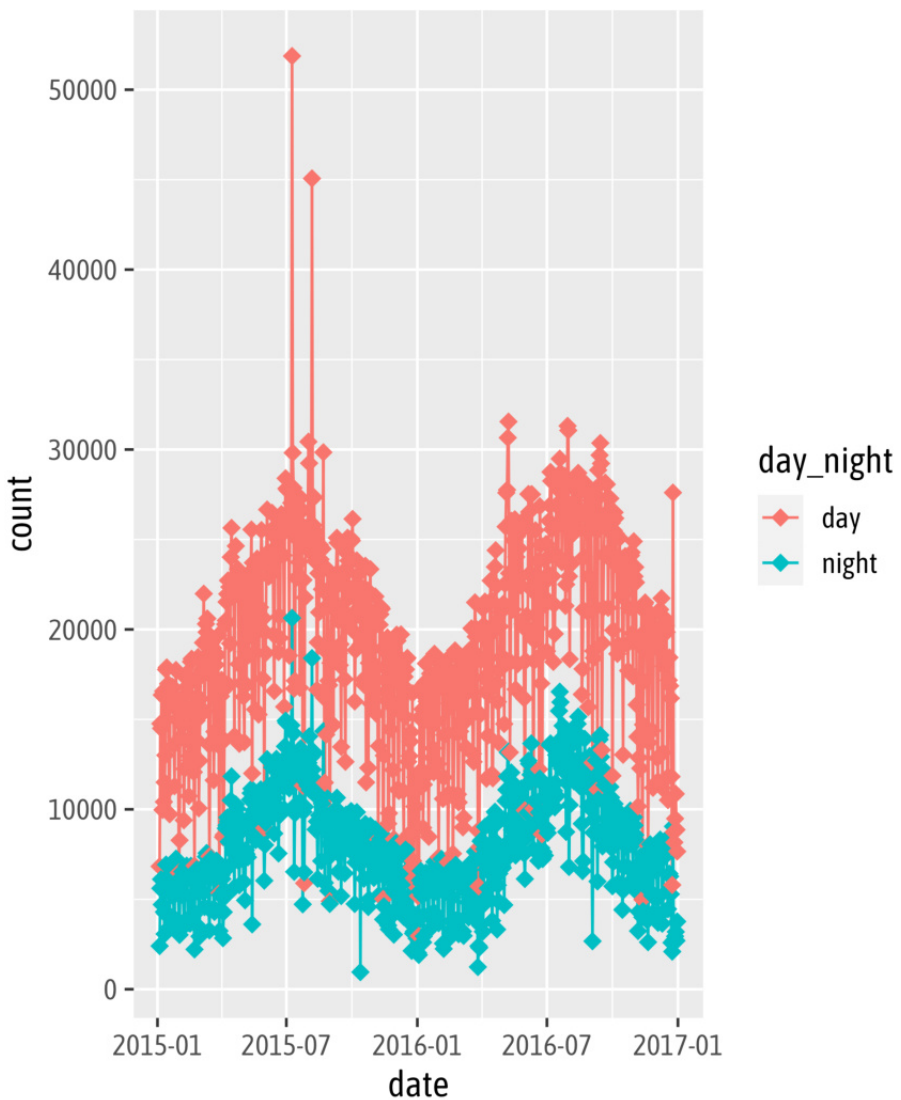
Solution Exercise

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



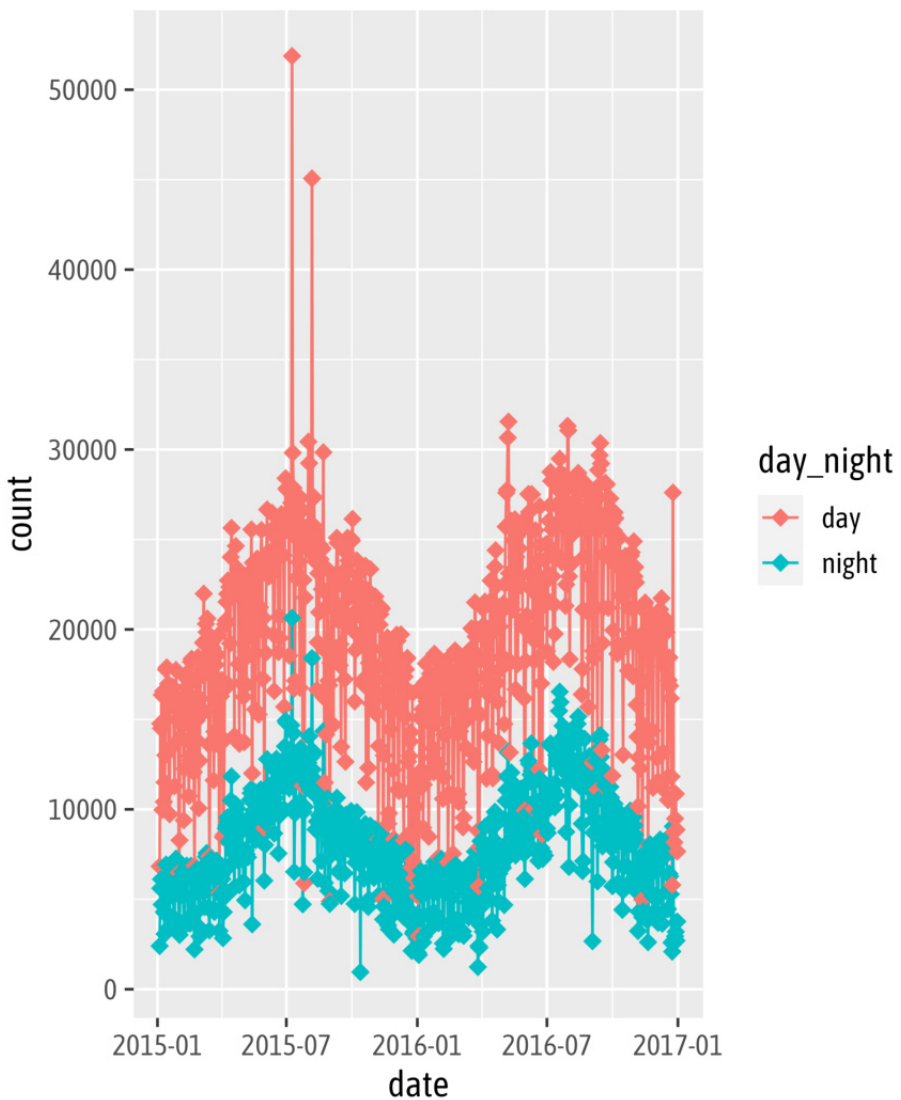
Solution Exercise

```
ggplot(
    bikes,
    aes(x = date, y = count,
        color = day_night)
  ) +
  geom_line() +
  geom_point(
    shape = "diamond",
    size = 3
  )
```



Solution Exercise

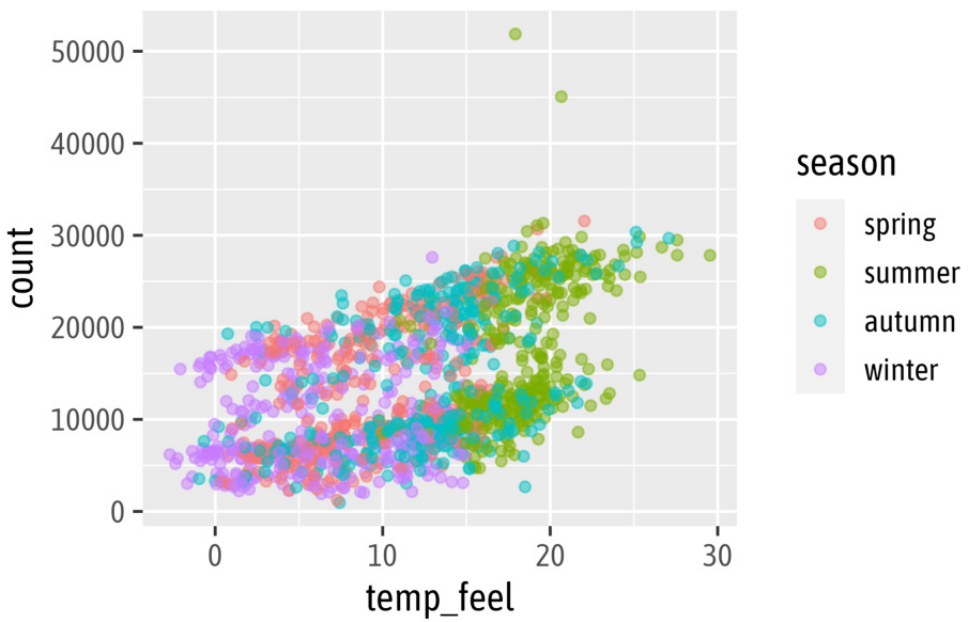
```
ggplot(
    bikes,
    aes(x = date, y = count,
        color = day_night)
  ) +
  geom_line() +
  geom_point(
    shape = 18,
    size = 3
  )
```



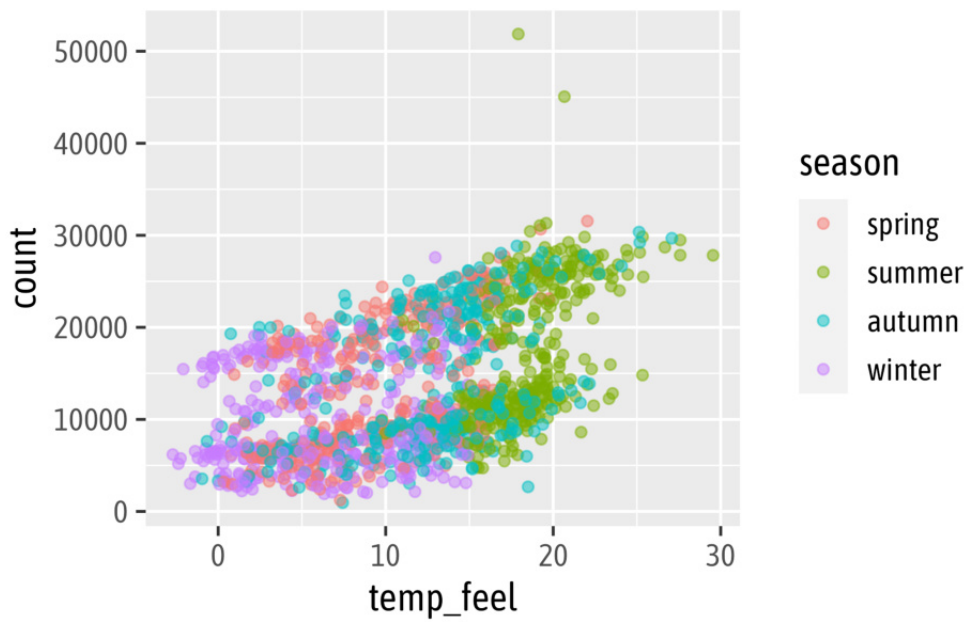
Source: Albert's Blog

Local vs. Global Encoding

```
1  ggplot(
2      bikes,
3      aes(x = temp_feel, y = count)
4    ) +
5    geom_point(
6      aes(color = season),
7      alpha = .5
8    )
```

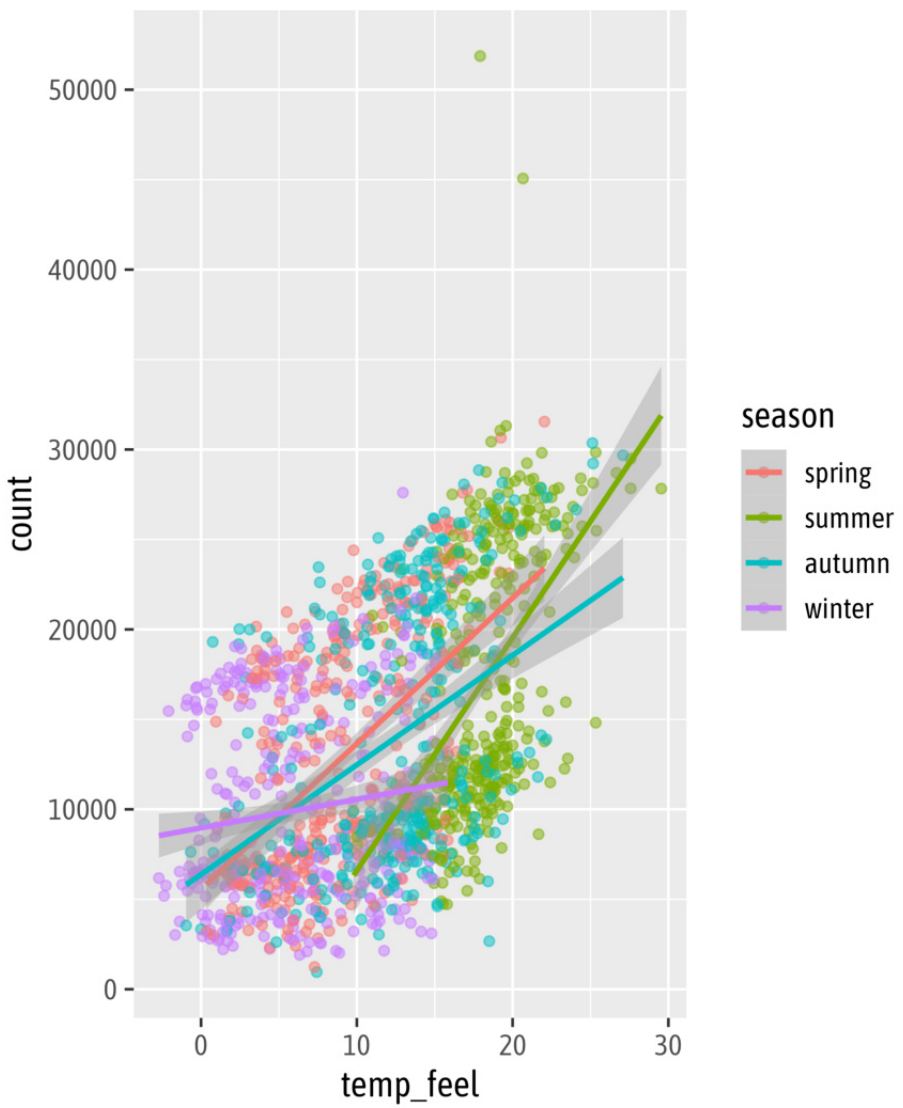



```
1  ggplot(
2      bikes,
3      aes(x = temp_feel, y = count,
4          color = season)
5    ) +
6    geom_point(
7      alpha = .5
8    )
```



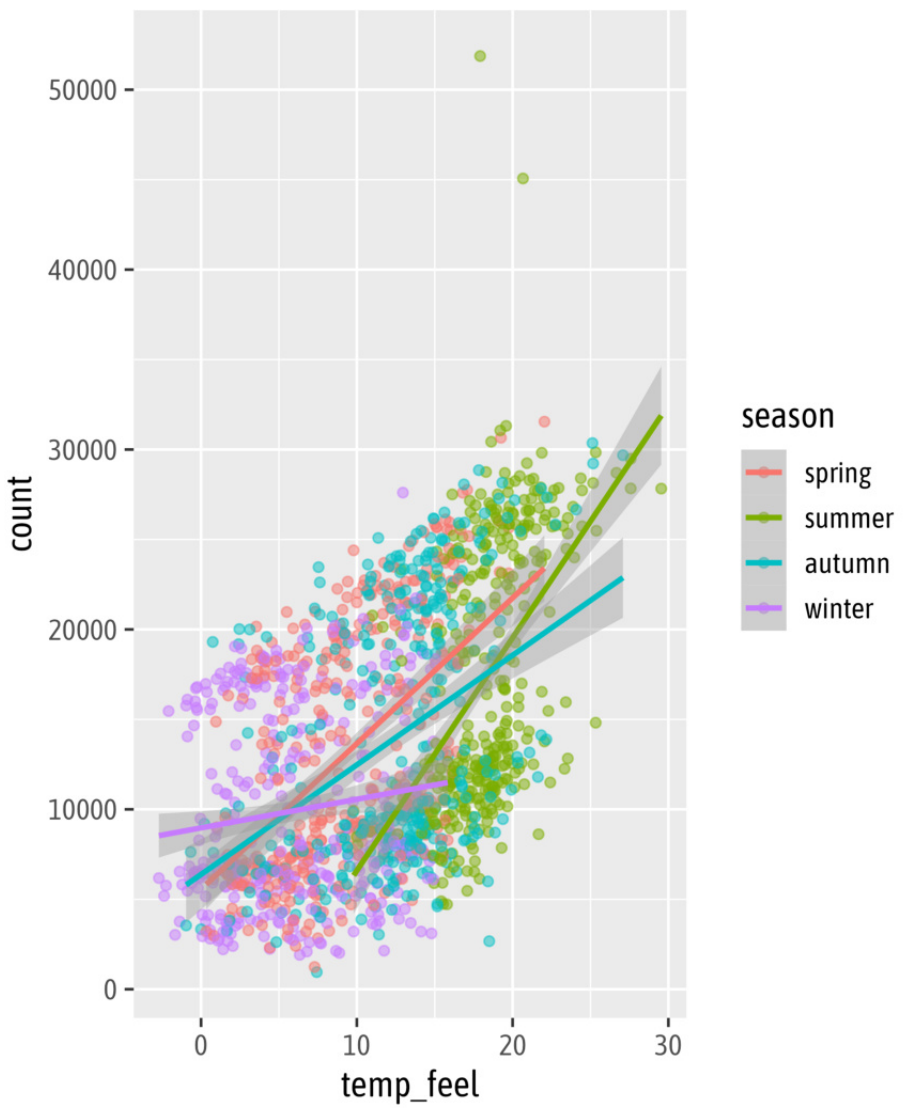
Adding More Layers

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season)
  ) +
  geom_point(
    alpha = .5
  ) +
  geom_smooth(
    method = "lm"
  )
```



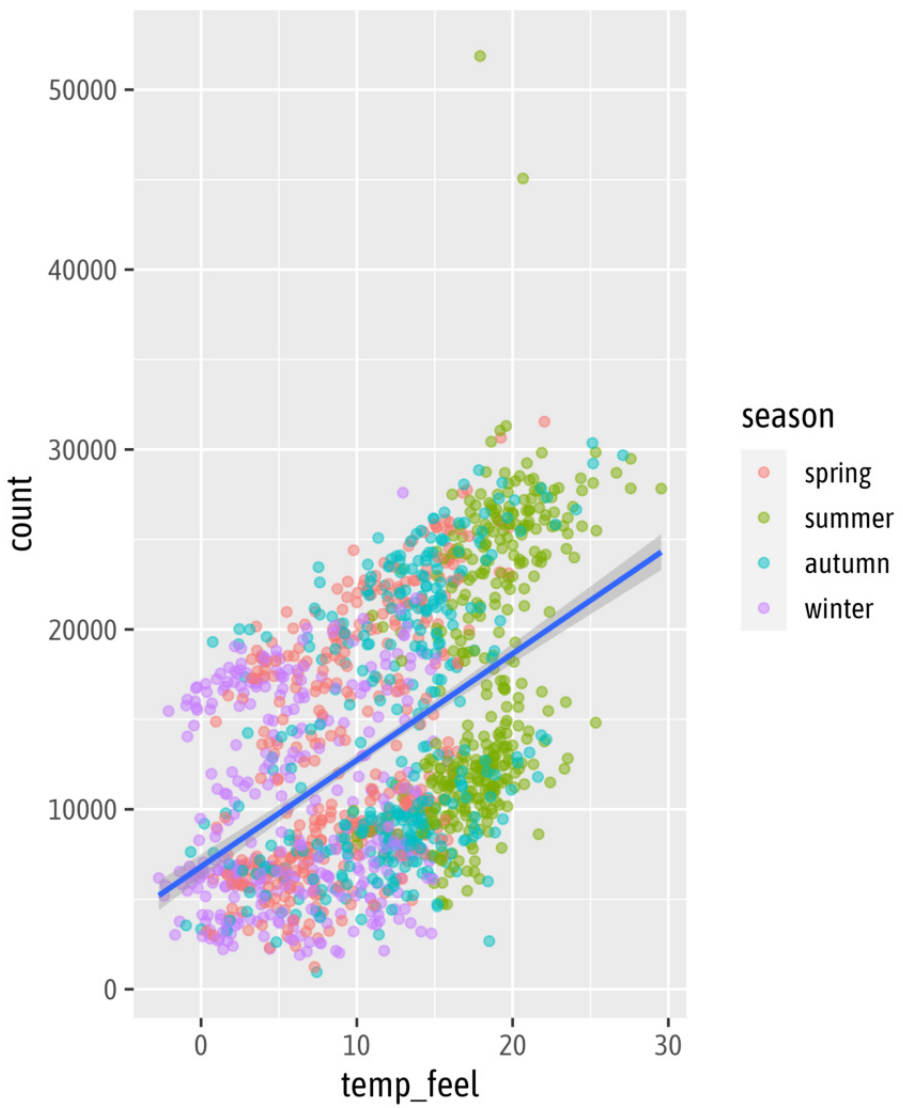
Global Color Encoding

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season)
  ) +
  geom_point(
    alpha = .5
  ) +
  geom_smooth(
    method = "lm"
  )
```



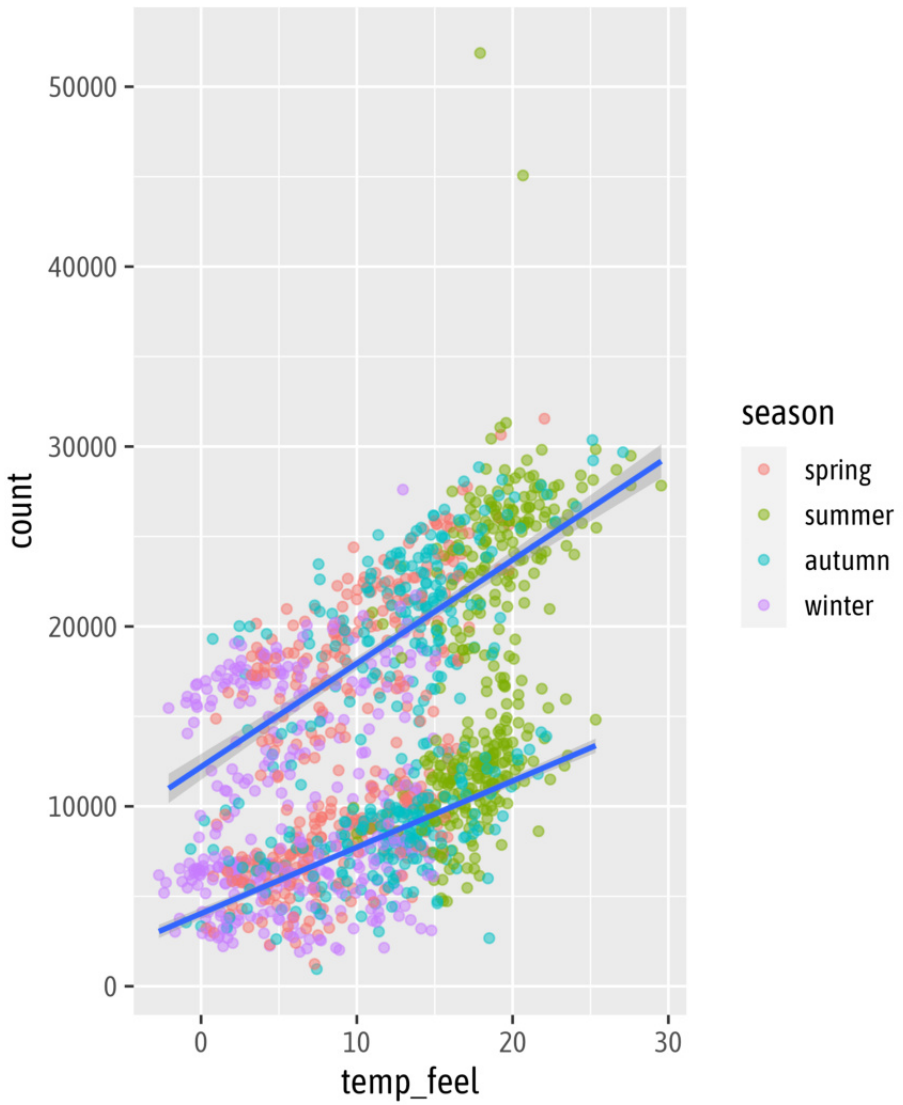
Local Color Encoding

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point(
    aes(color = season),
    alpha = .5
  ) +
  geom_smooth(
    method = "lm"
  )
```



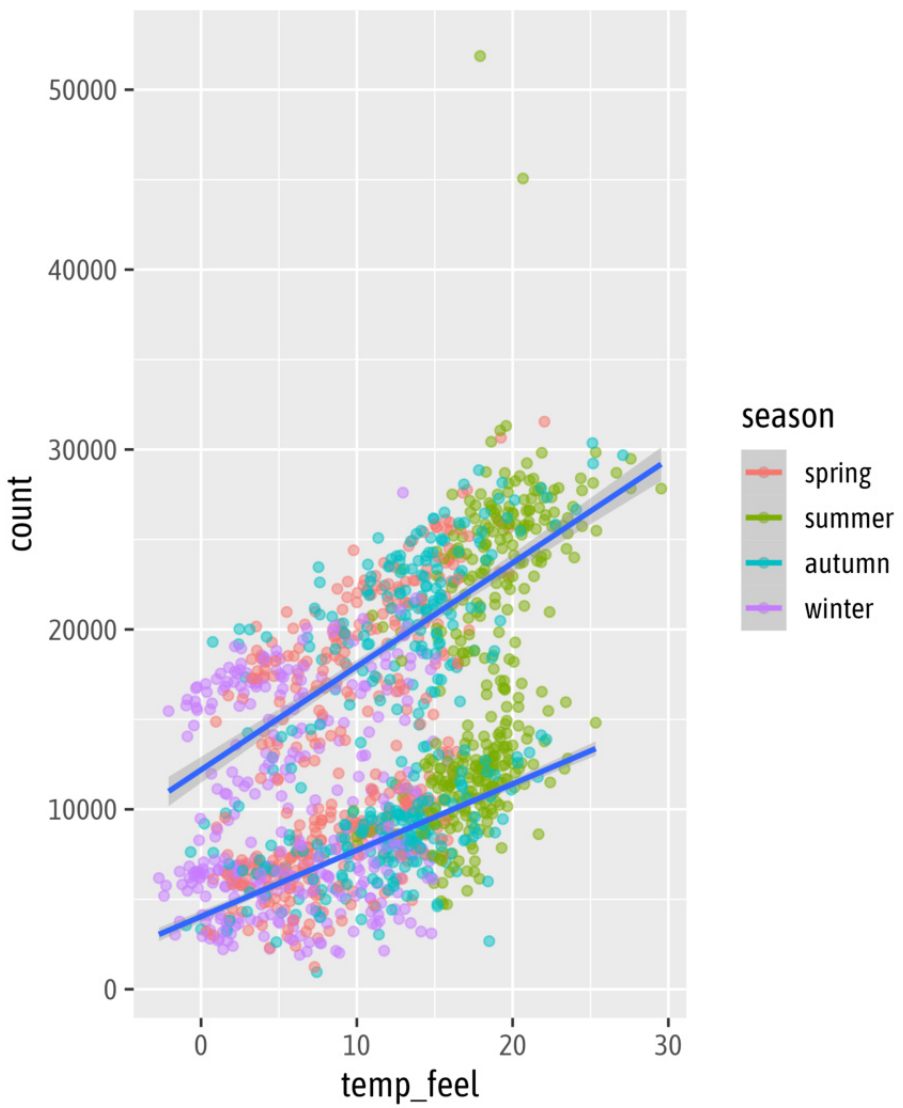
The `group` Aesthetic

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point(
    aes(color = season),
    alpha = .5
  ) +
  geom_smooth(
    aes(group = day_night),
    method = "lm"
  )
```



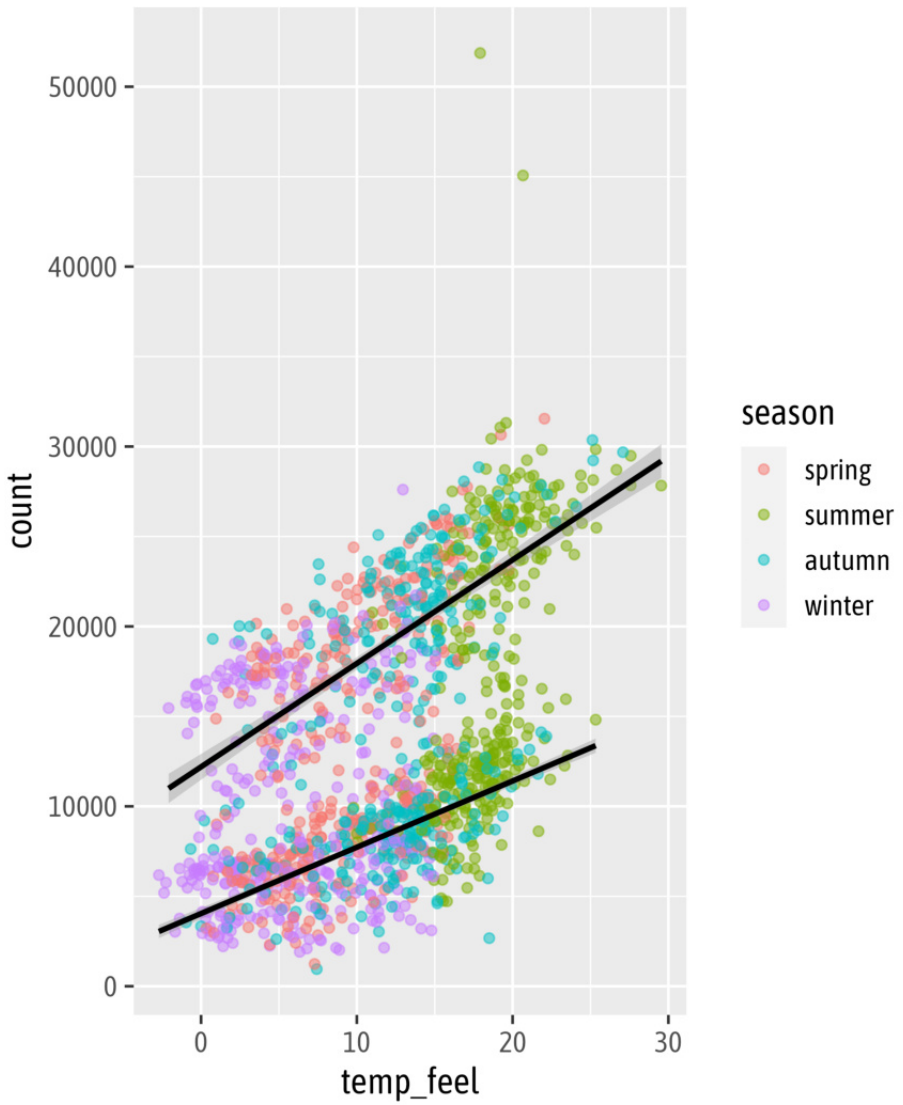
Set Both as Global Aesthetics

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season,
        group = day_night)
  ) +
  geom_point(
    alpha = .5
  ) +
  geom_smooth(
    method = "lm"
  )
```



Overwrite Global Aesthetics

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season,
        group = day_night)
  ) +
  geom_point(
    alpha = .5
  ) +
  geom_smooth(
    method = "lm",
    color = "black"
  )
```



Store a ggplot as Object

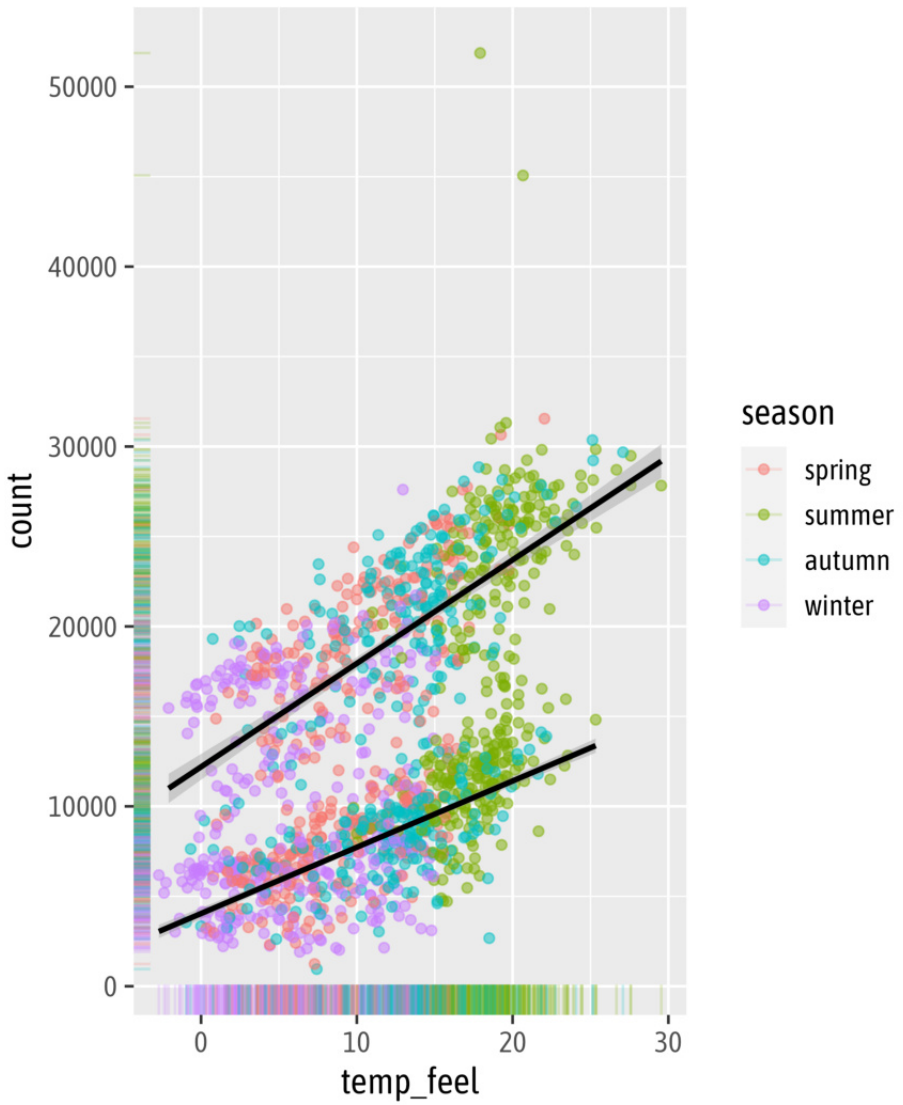
```r
g <-
  ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season,
        group = day_night)
  ) +
  geom_point(
    alpha = .5
  ) +
  geom_smooth(
    method = "lm",
    color = "black"
  )
```

```r
class(g)
```

```
[1] "gg"     "ggplot"
```

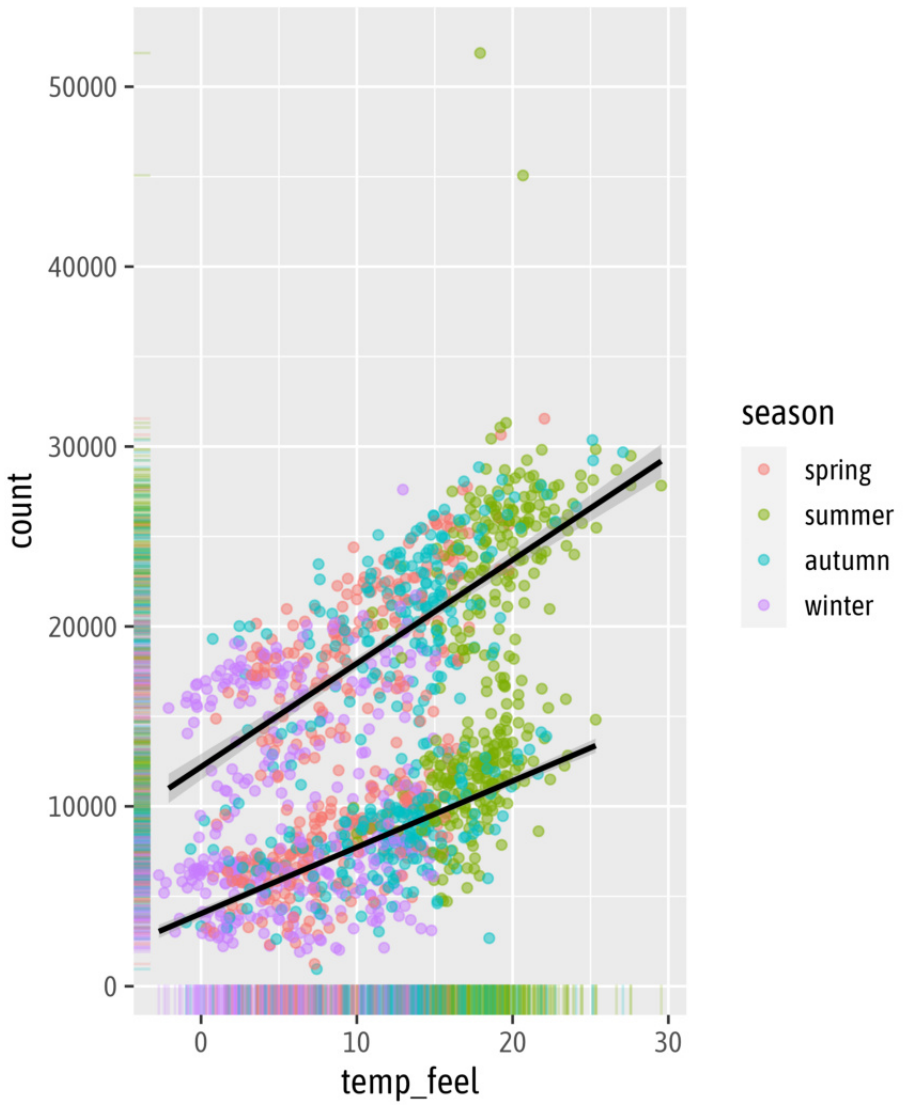

Add More Layers

```
g +
  geom_rug(
    alpha = .2
  )
```



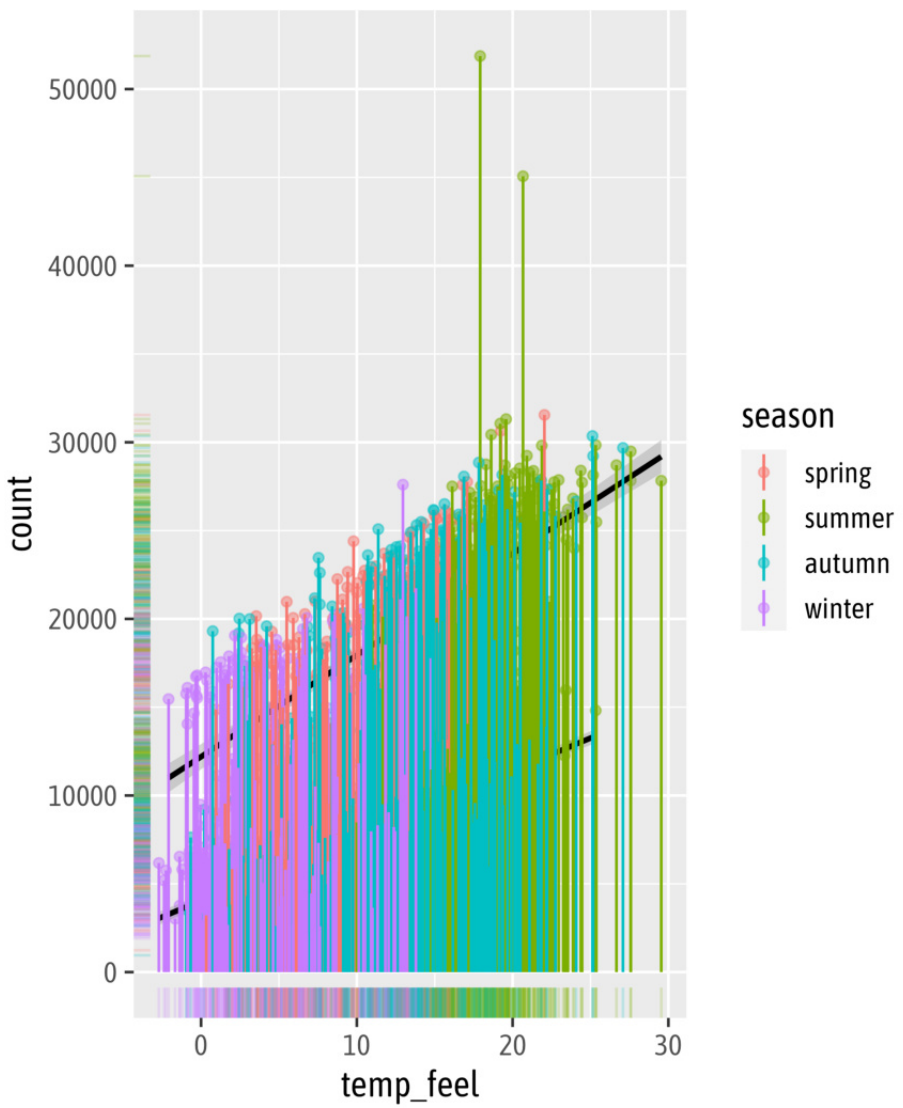
Remove a Layer from the Legend

```
g +
  geom_rug(
    alpha = .2,
    show.legend = FALSE
  )
```



Add More Layers

```
g +
  geom_rug(
    alpha = .2,
    show.legend = FALSE
  ) +
  geom_linerange(
    aes(ymin = 0, ymax = count)
  )
```

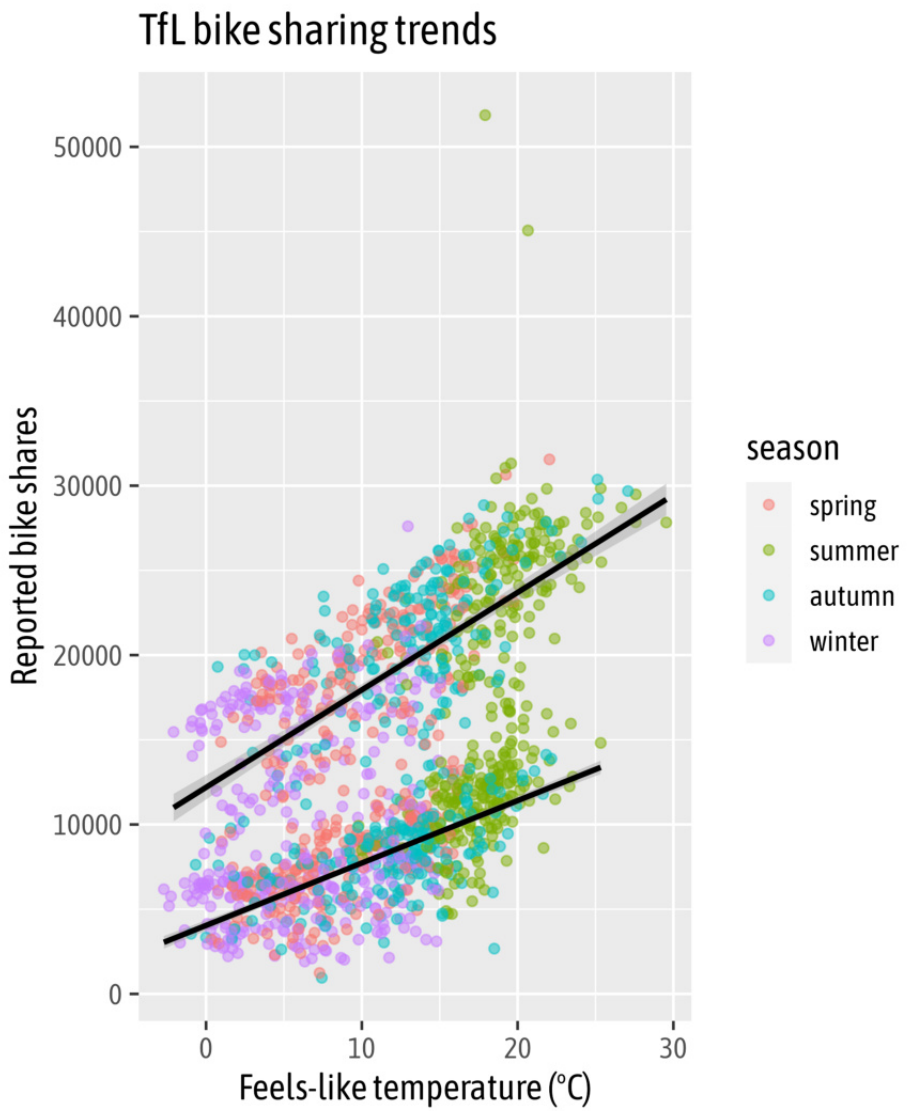


A Polished ggplot Example

Cédric Scherer // Data Visualization Society // March 9, 2023

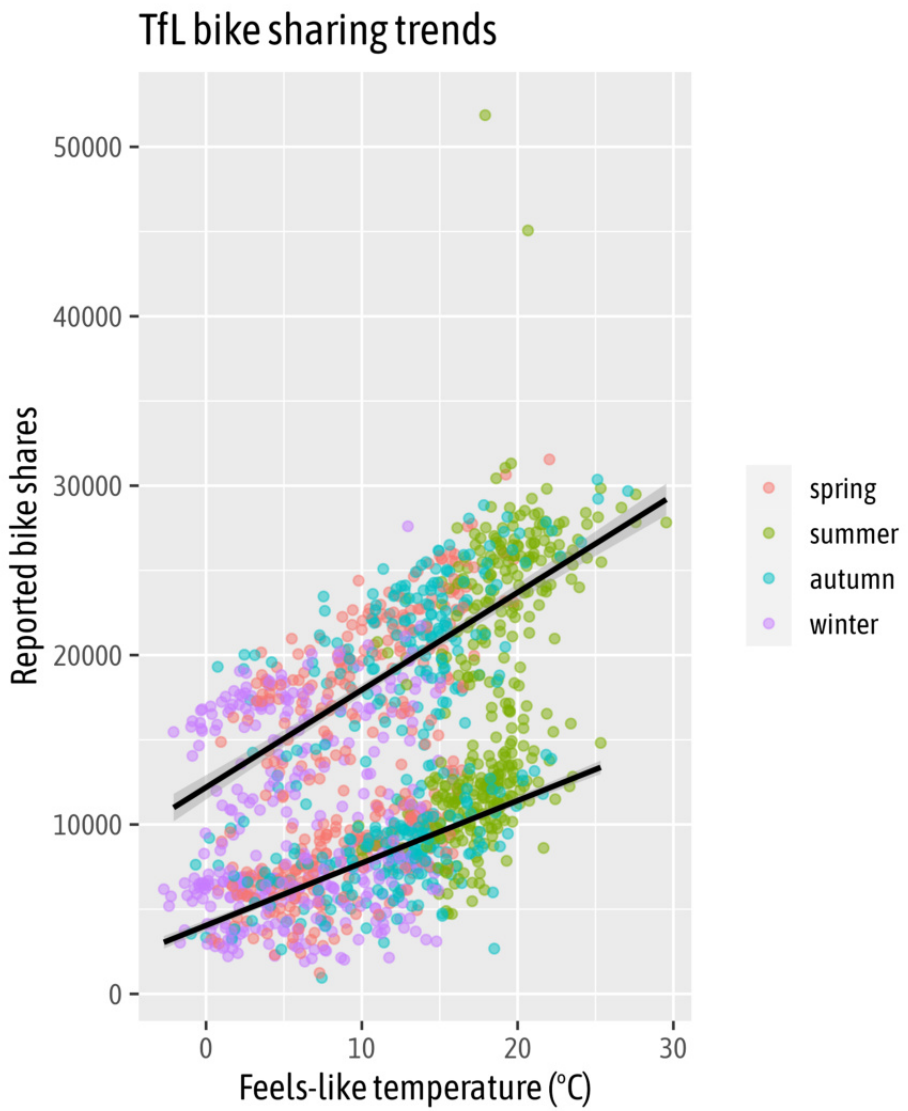
Add Labels

```
g +
  labs(
    x = "Feels-like temperature (°C)",
    y = "Reported bike shares",
    title = "TfL bike sharing trends"
  )
```



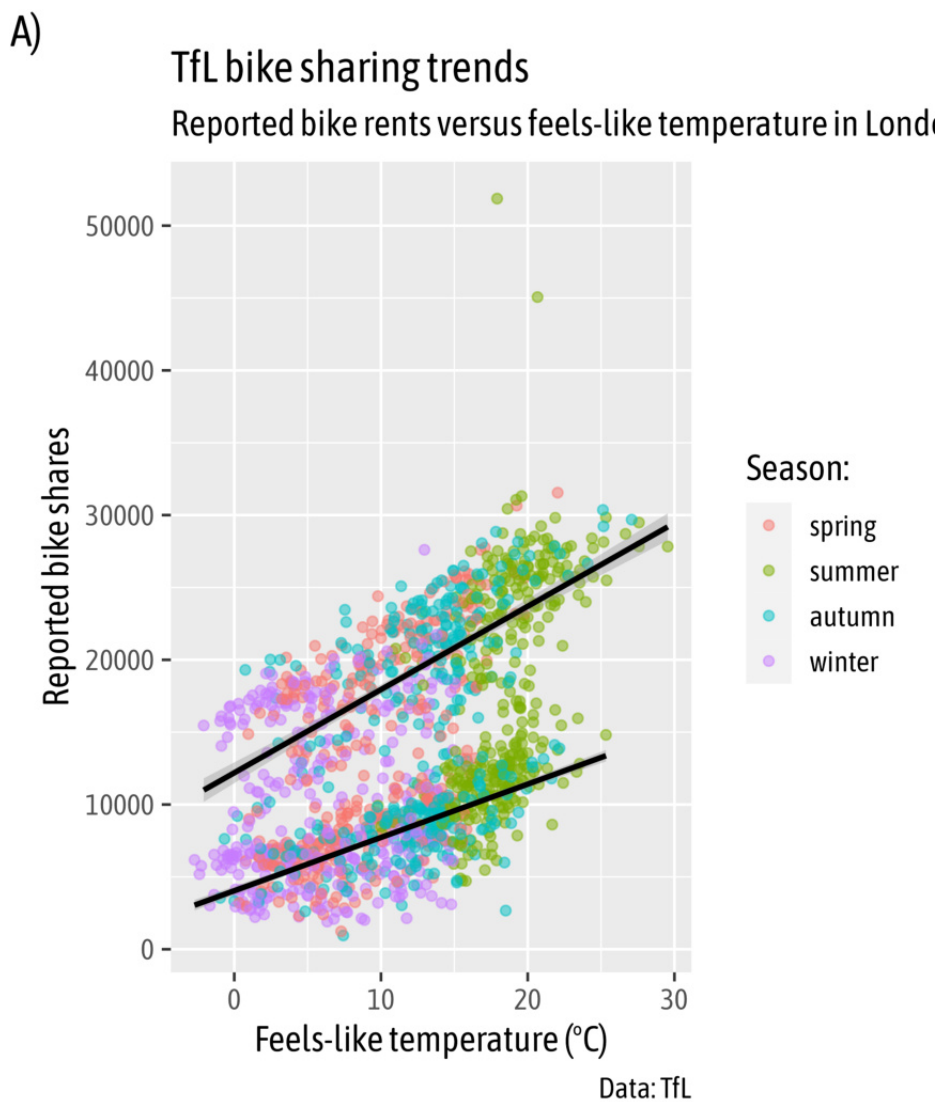
Add Labels

```
g <- g +
  labs(
    x = "Feels-like temperature (°C)",
    y = "Reported bike shares",
    title = "TfL bike sharing trends",
    color = NULL
  )

g
```



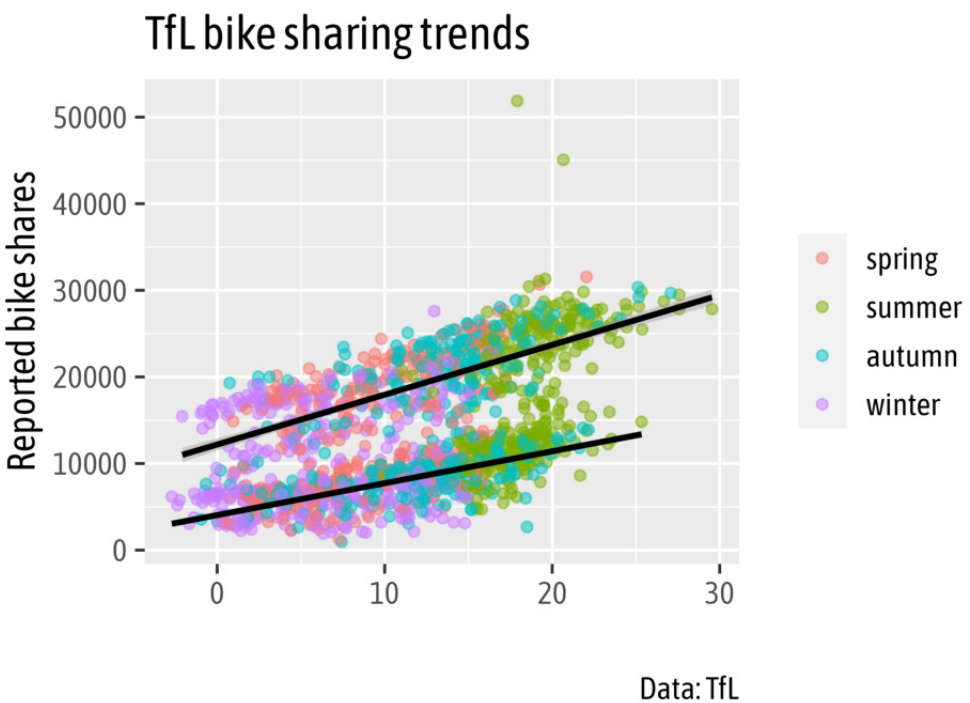
Add Labels

```
g +
  labs(
    subtitle = "Reported bike rents versus fe
    caption = "Data: TfL",
    tag = "A)",
    color = "Season:"
  )
```

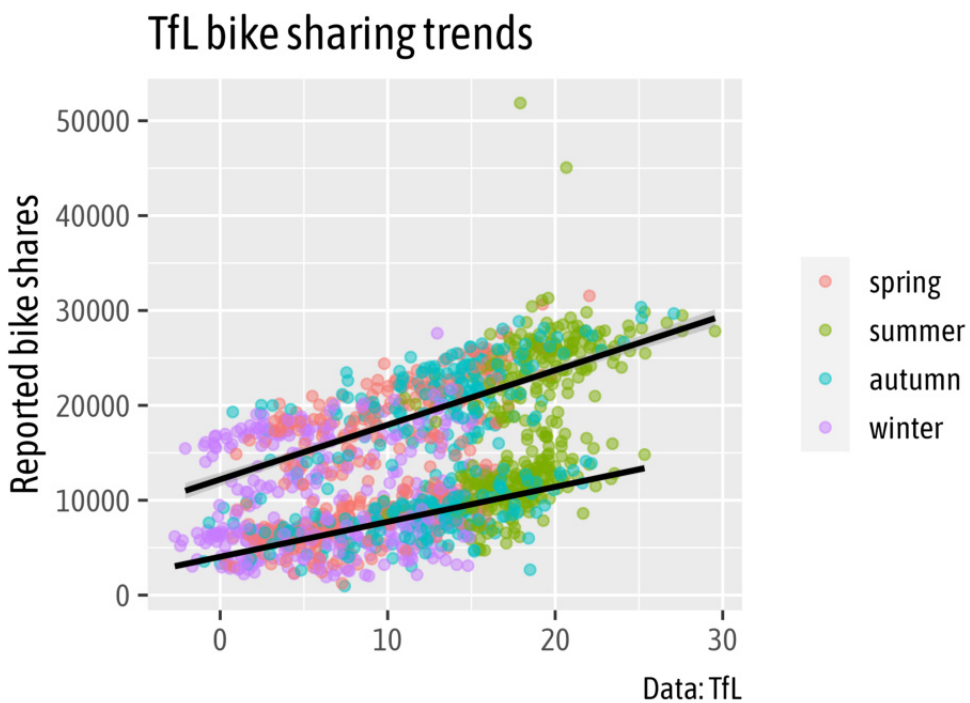


Add Labels

```
g +
  labs(
    x = "",
    caption = "Data: TfL"
  )
```

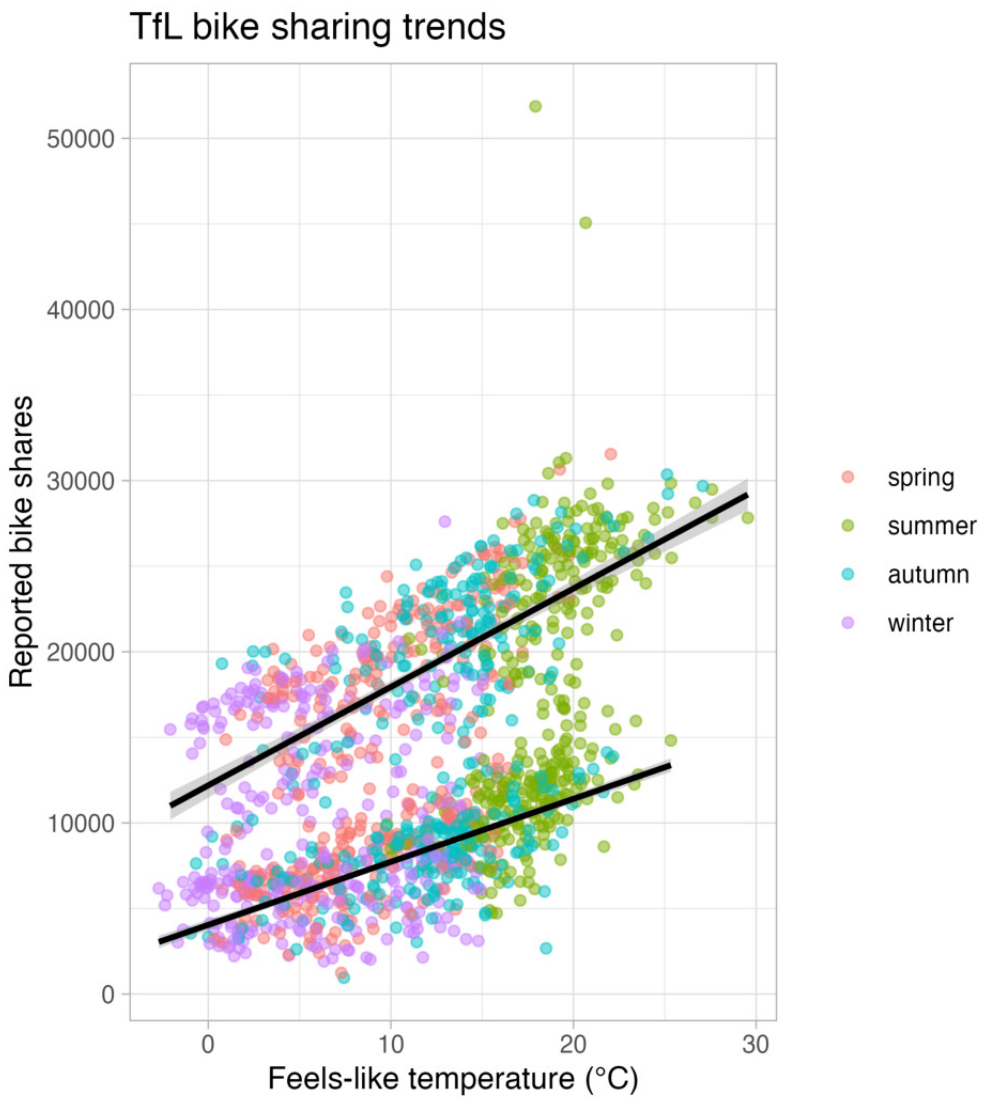



```
g +
  labs(
    x = NULL,
    caption = "Data: TfL"
  )
```

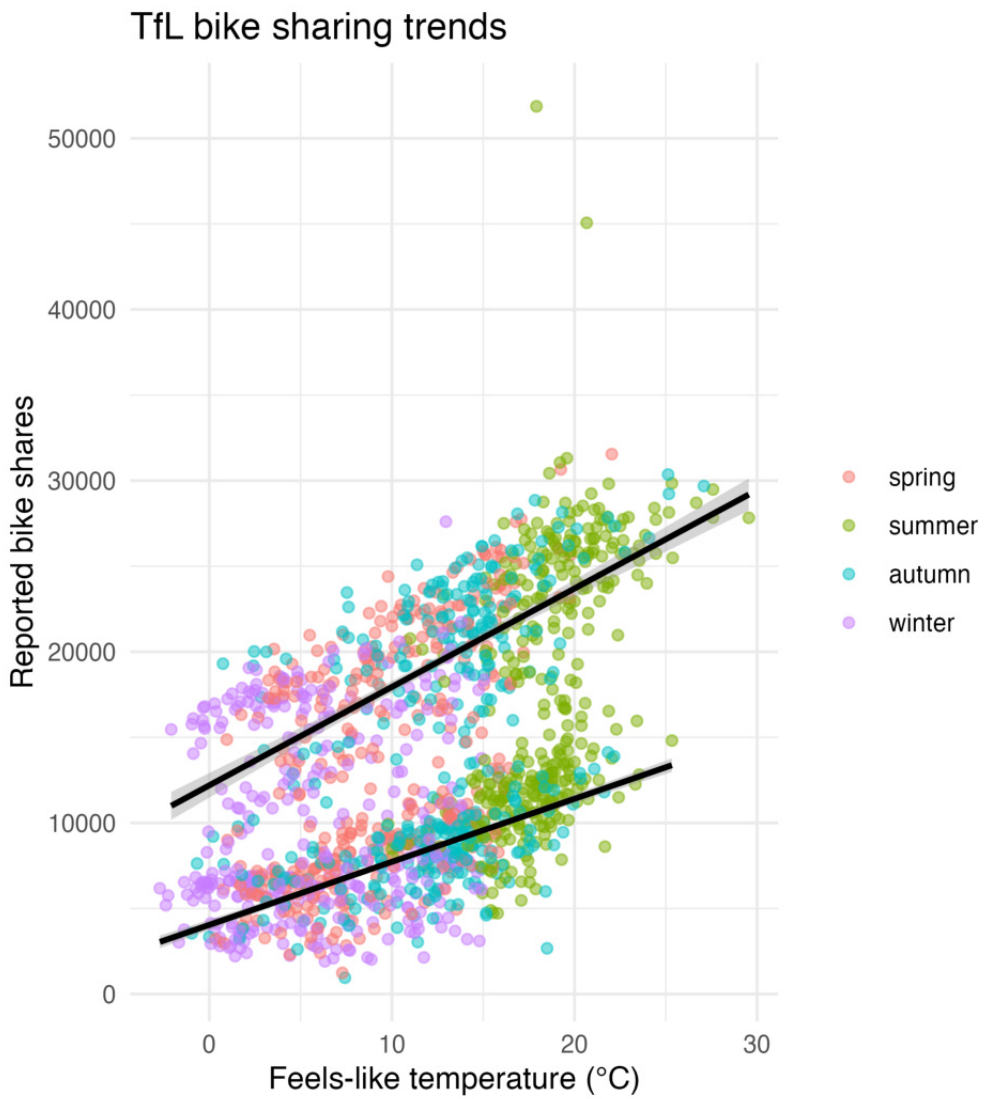


Themes

```
1  g + theme_light()
```

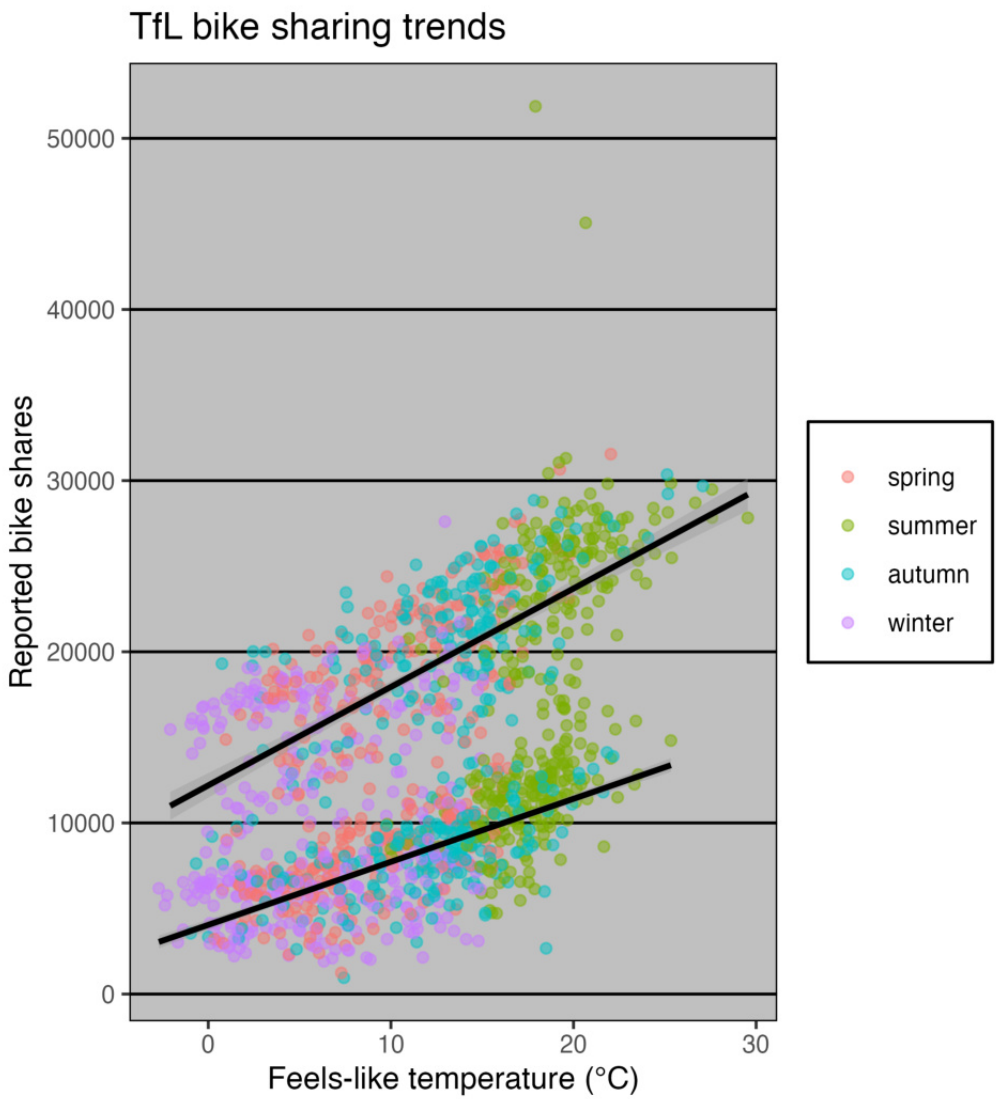



```
1  g + theme_minimal()
```

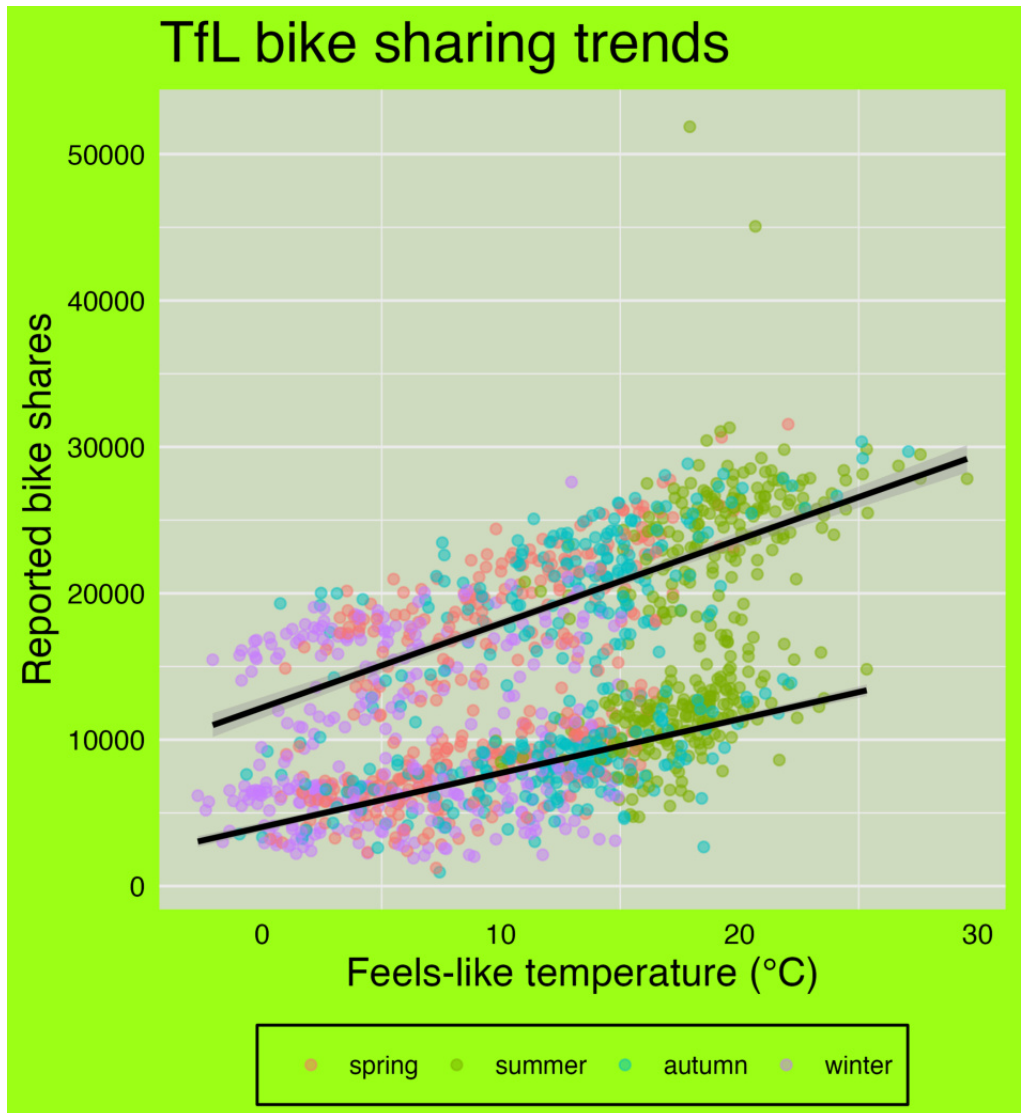


Themes

```
1  g + ggthemes::theme_excel()
```

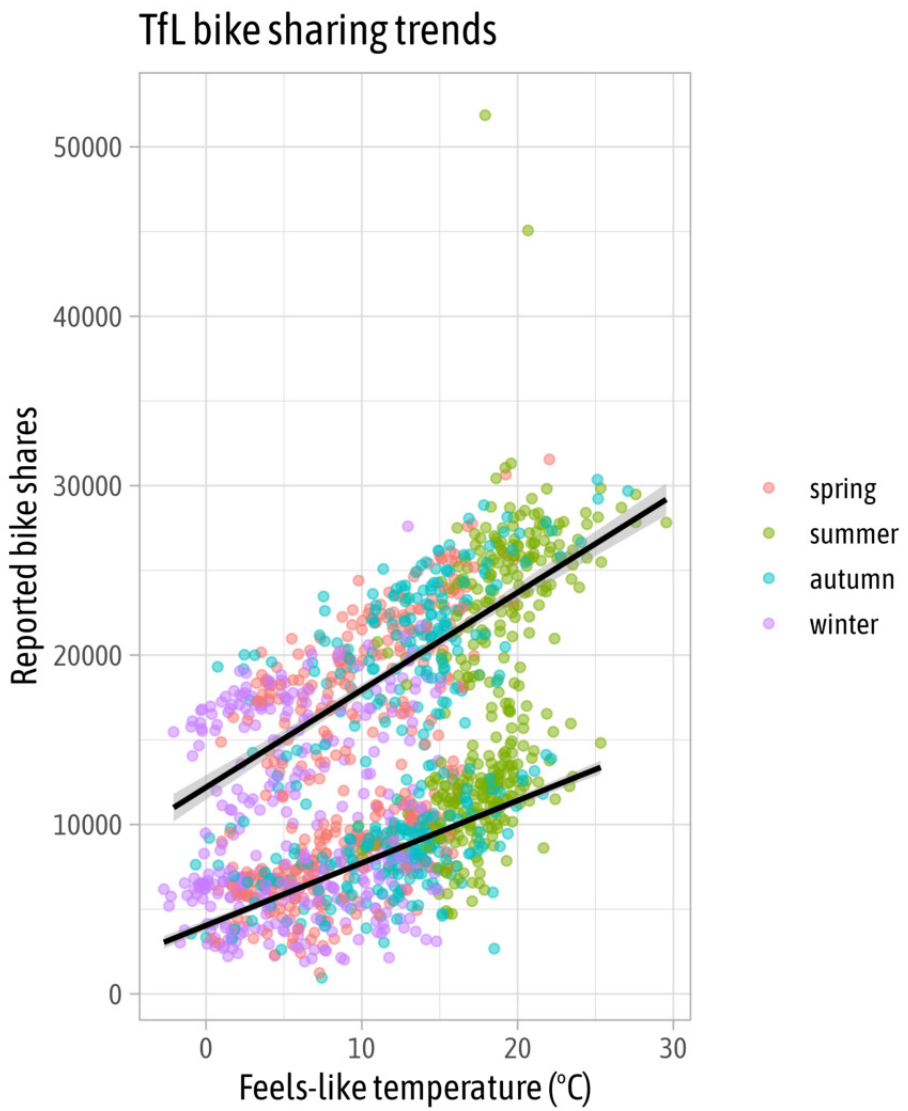



```
1  g + tvthemes::theme_rickAndMorty()
```



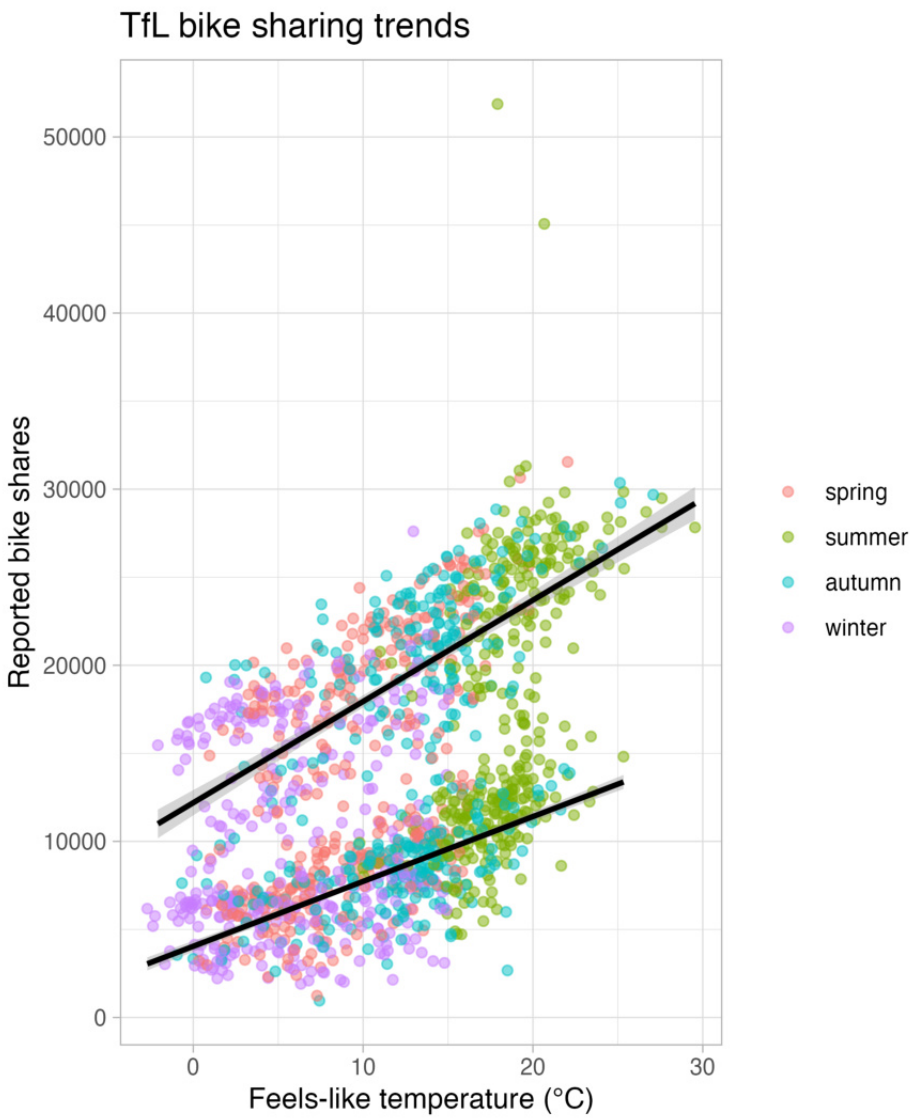
Change the Theme Base Settings

```
g + theme_light(
  base_size = 14,
  base_family = "Asap Condensed"
)
```



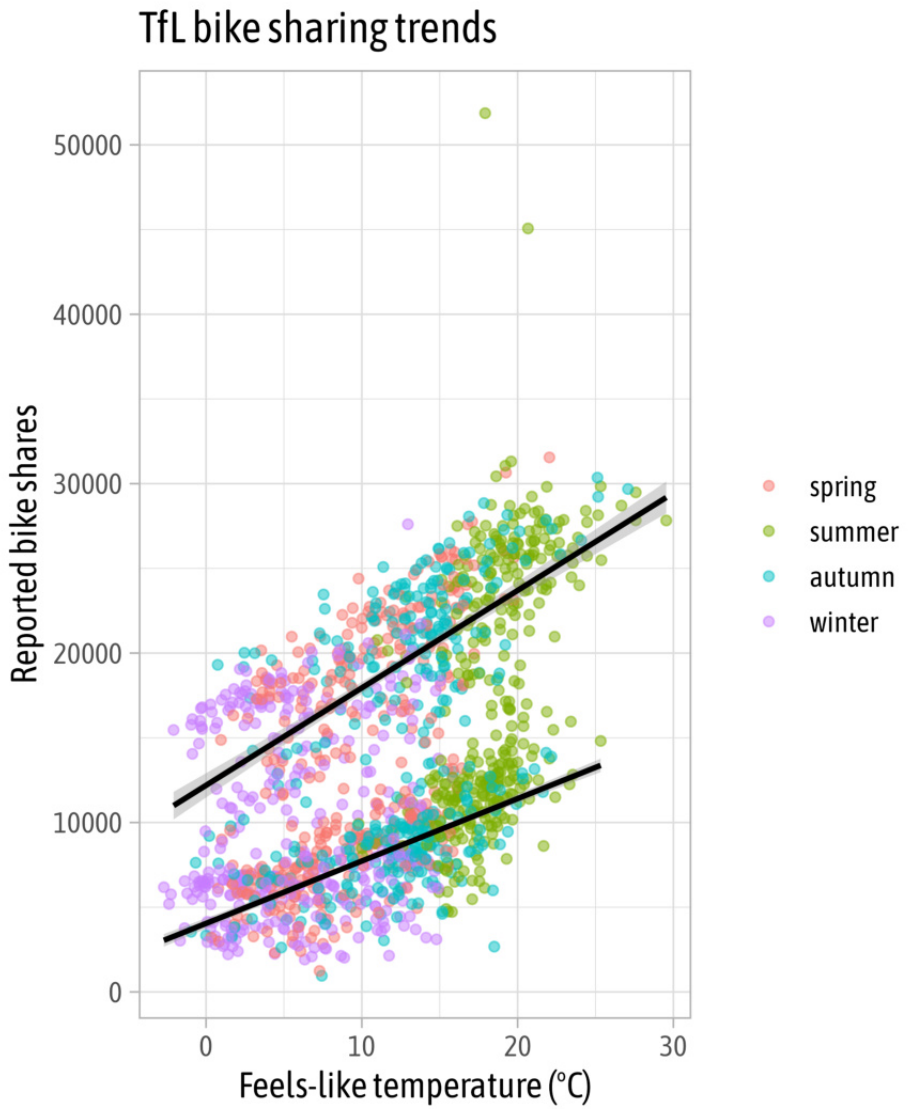
Set a Theme Globally

```
1 theme_set(theme_light())
2
3 g
```



Change the Theme Base Settings

```
theme_set(theme_light(
  base_size = 14,
  base_family = "Asap Condensed"
))

g
```



{systemfonts}

```r
# install.packages("systemfonts")
library(systemfonts)

system_fonts() %>%
  filter(stringr::str_detect(family, "Asap")) %>%
  pull(family) %>%
  unique() %>%
  sort()
```

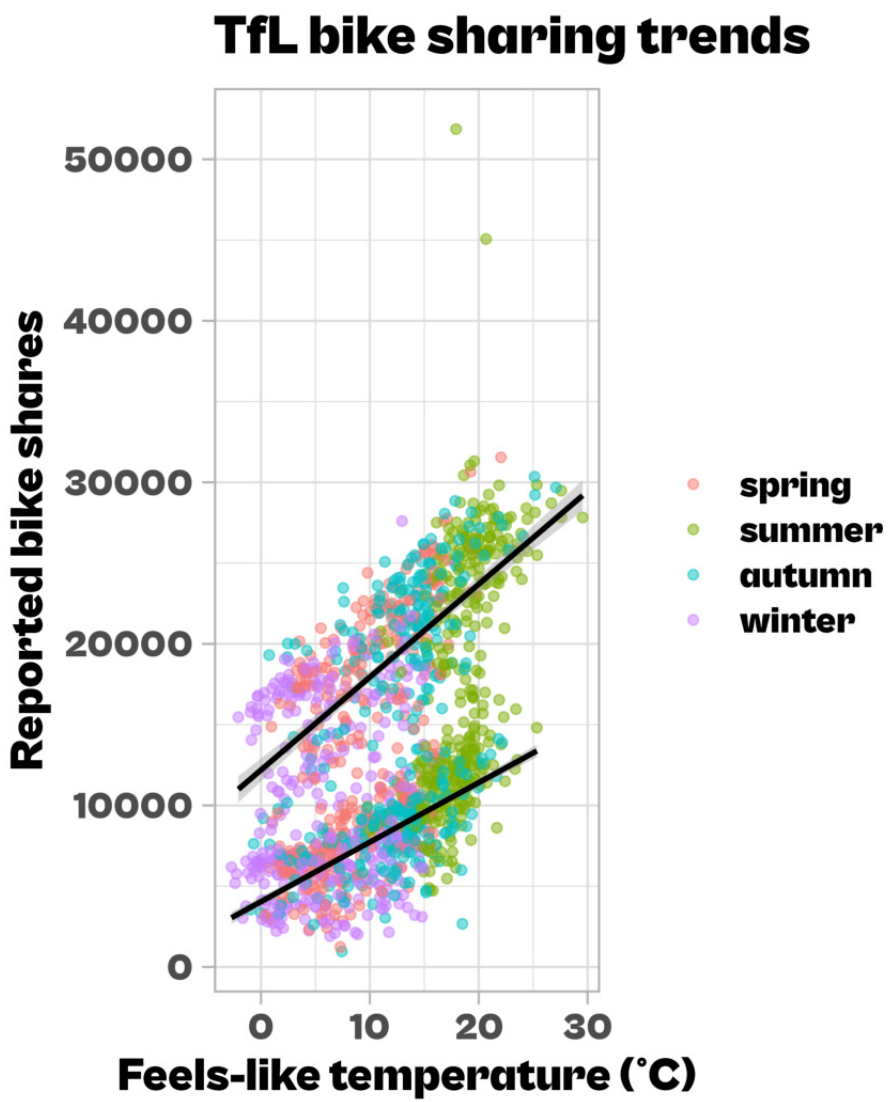
```
[1] "Asap"              "Asap Condensed"    "Asap Expanded"     "Asap SemiCondensed" "Asap SemiExpanded"
```


{systemfonts}

```
register_variant(
  name = "Cabinet Grotesk Black",
  family = "Cabinet Grotesk",
  weight = "heavy",
  features = font_feature(letters = "stylistic")
)
```

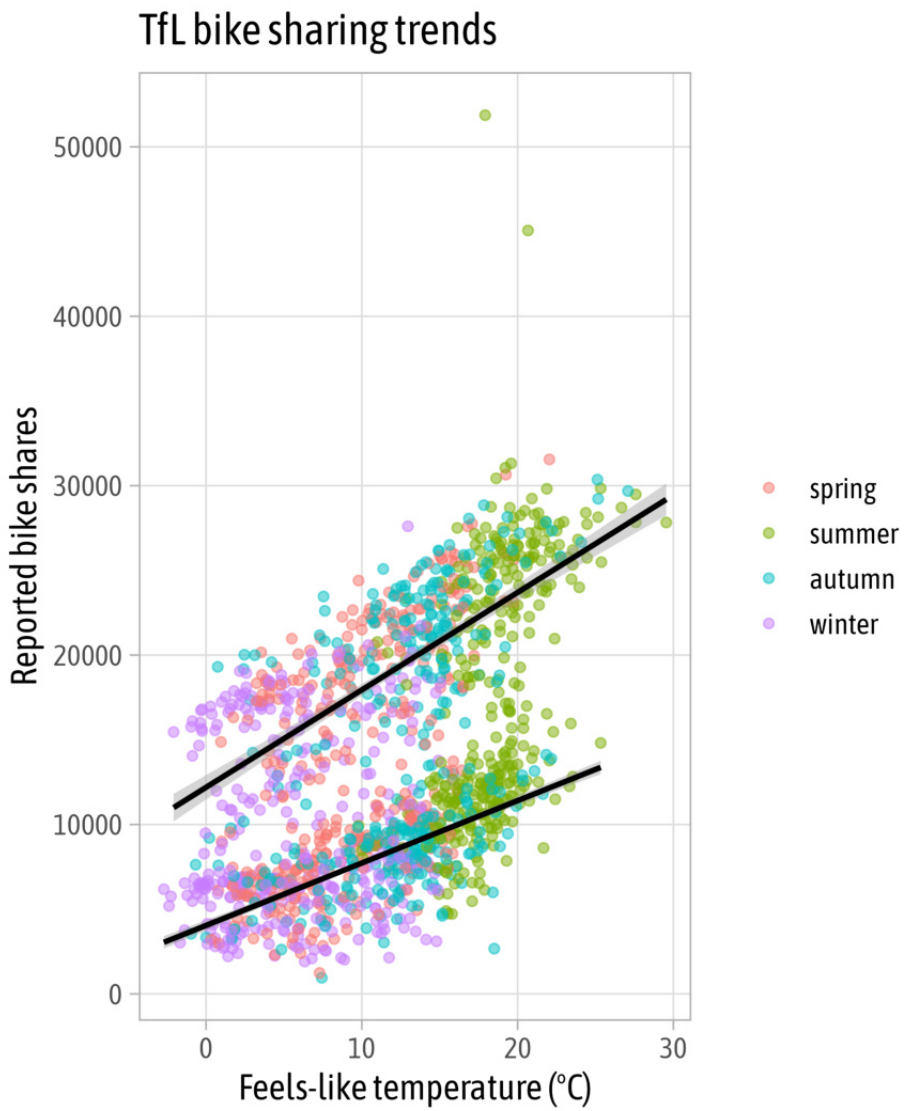

{systemfonts} + {ggplot2}

```
g +
  theme_light(
    base_size = 18,
    base_family = "Cabinet Grotesk Black"
  )
```



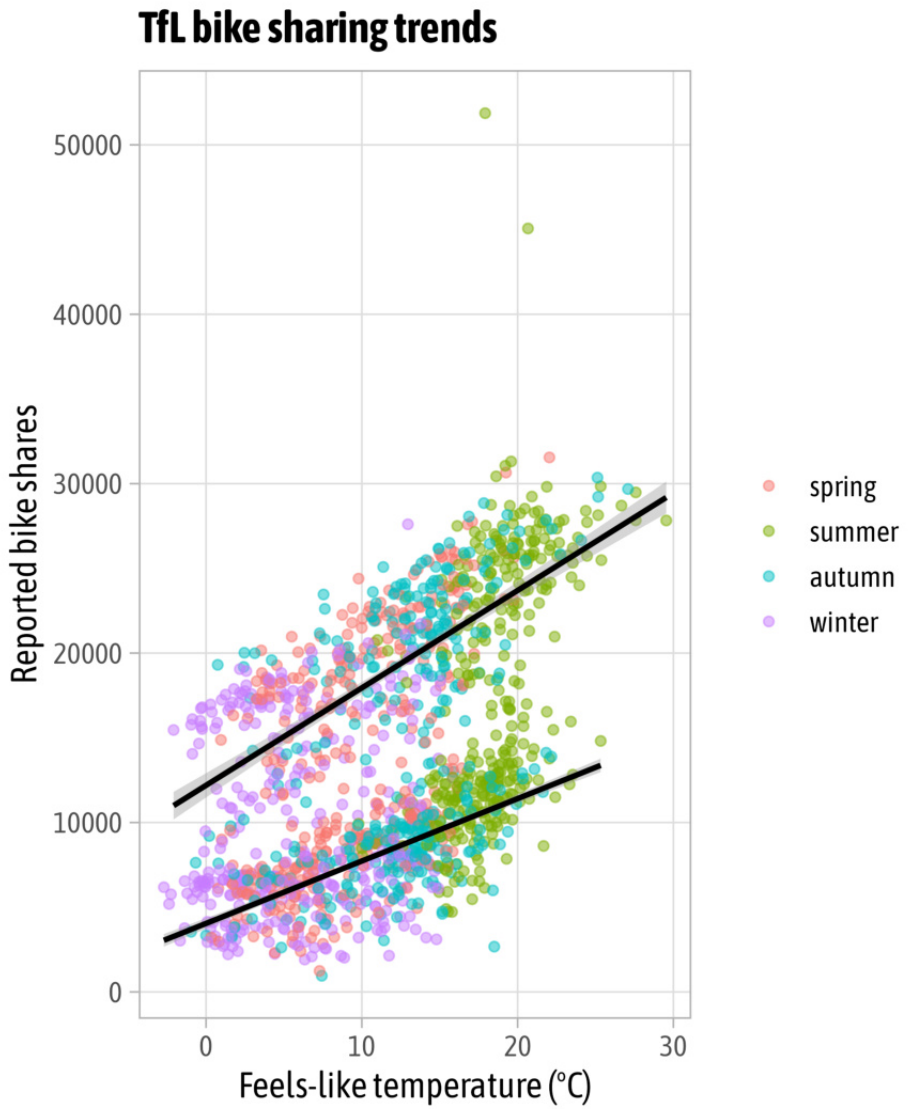
Overwrite Specific Theme Settings

```
1 g +
2   theme(
3     panel.grid.minor = element_blank()
4   )
```



Overwrite Specific Theme Settings

```
g +
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold")
  )
```

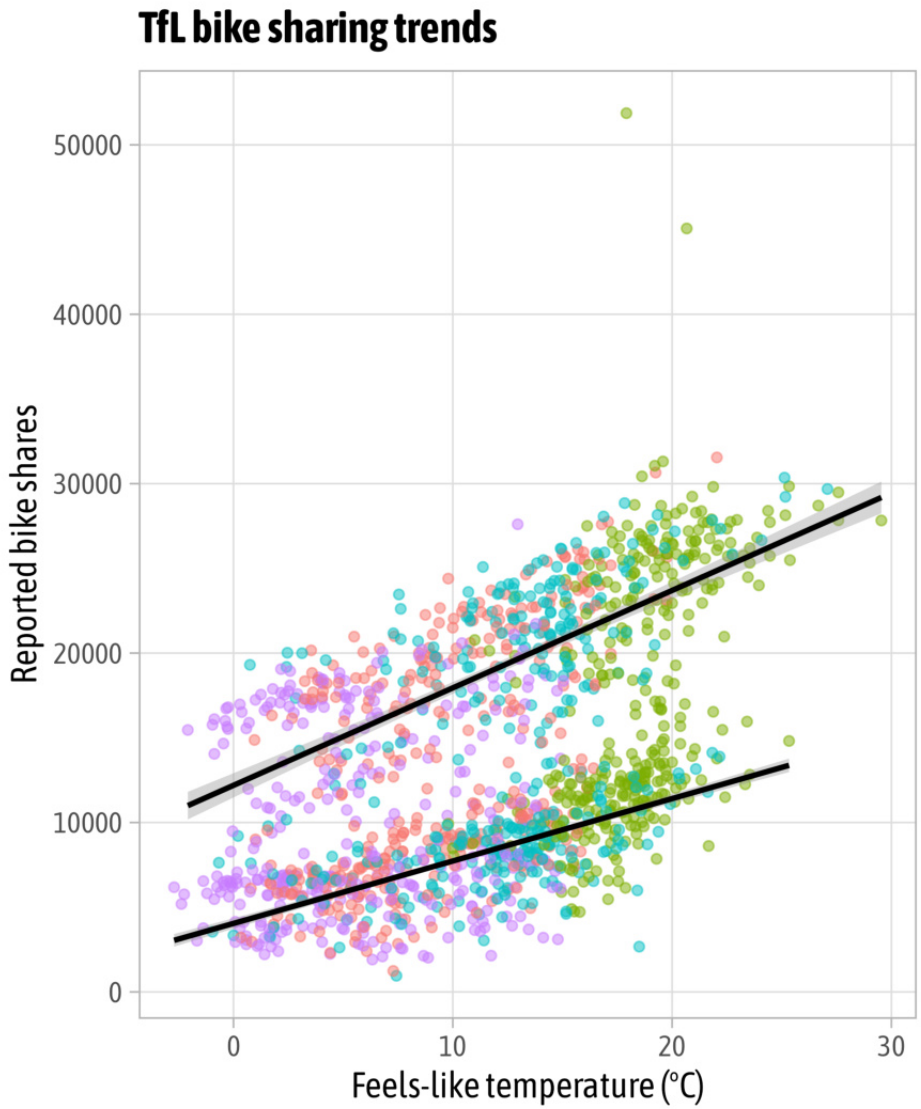


Overwrite Specific Theme Settings

```
g +
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold"),
    legend.position = "top"
  )
```

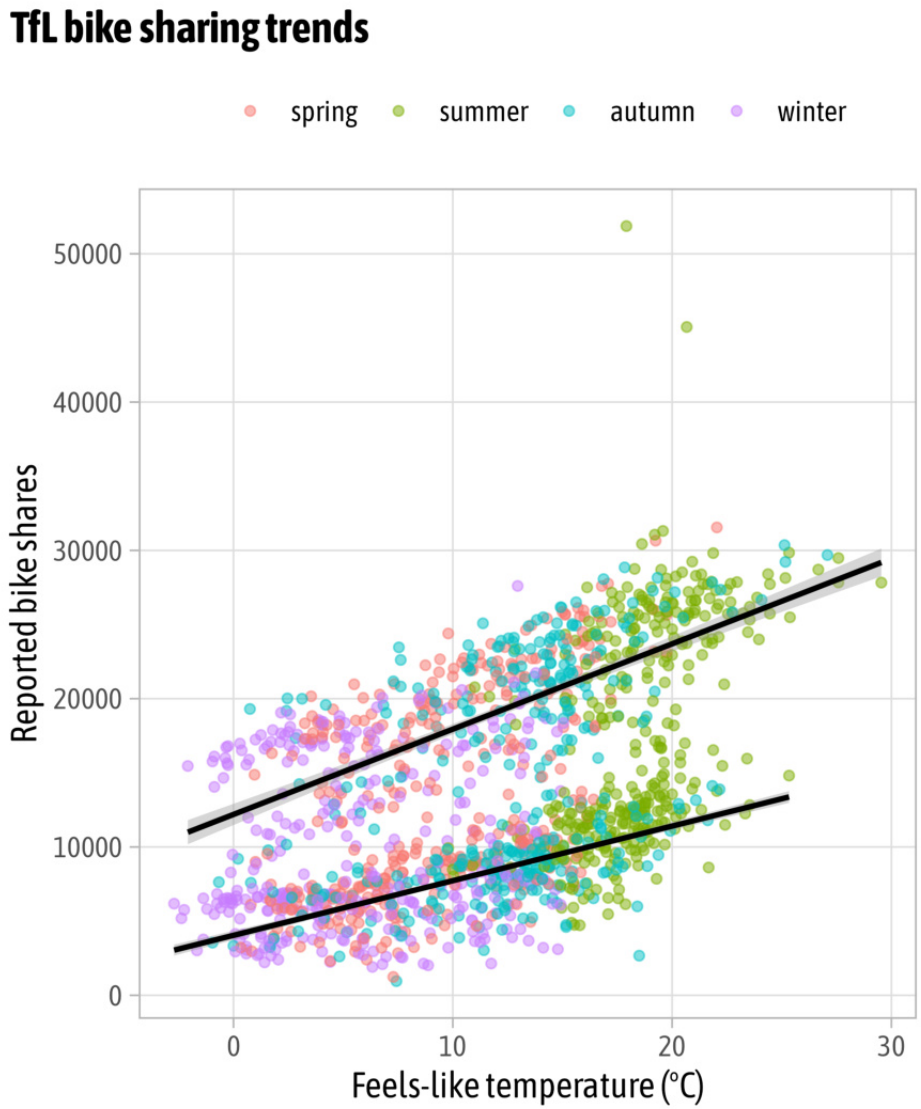
Overwrite Specific Theme Settings

```
g +
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold"),
    legend.position = "none"
  )
```



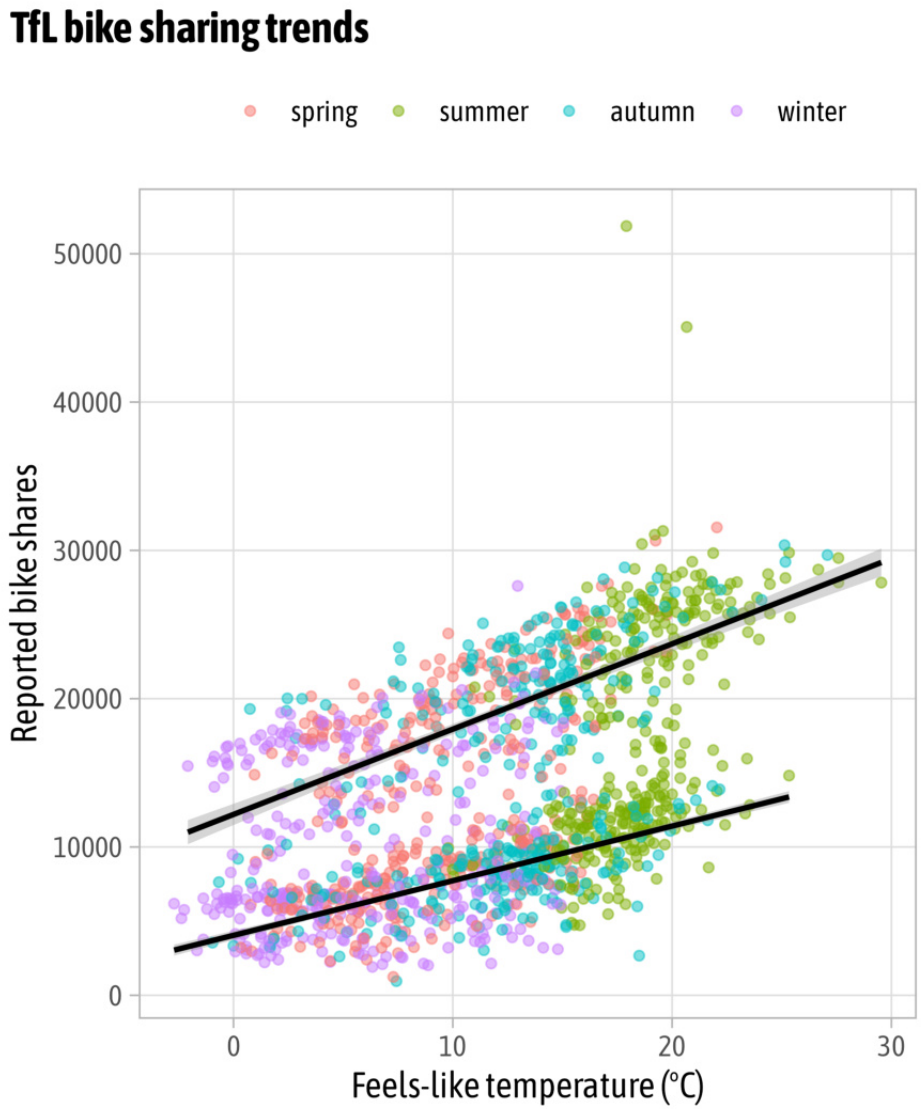
Overwrite Specific Theme Settings

```
g +
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold"),
    legend.position = "top",
    plot.title.position = "plot"
  )
```



Overwrite Theme Settings Globally

```
1 theme_update(
2   panel.grid.minor = element_blank(),
3   plot.title = element_text(face = "bold"),
4   legend.position = "top",
5   plot.title.position = "plot"
6 )
7
8 g
```



Export Your Graphic

Cédric Scherer // Data Visualization Society // March 9, 2023

Save the Graphic

```
ggsave(g, filename = "my_plot.png")
```

```
ggsave("my_plot.png")
```

```
ggsave("my_plot.png", width = 8, height = 5, dpi = 600)
```

```
ggsave("my_plot.pdf", width = 20, height = 12, unit = "cm", device = cairo_pdf)
```


Modified from canva.com

Scales

Cédric Scherer // Data Visualization Society // March 9, 2023

Scales

= translate between variable and property ranges

- feels-like temperature ⇄ x
- reported bike shares ⇄ y
- season ⇄ color
- year ⇄ 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_*()`

... with tons of options for `*` such as `continuous`, `discrete`, `manual`, `log10`, `gradient`, and many more

Scales

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

Use scales to:

- set breaks (tick marks)
- overwrite axis / legend labels
- adjust axis / legend title
- modify colors and palettes
- define shapes and line types
- scale sizes (e.g. bubbles)

Illustration by Allison Horst

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

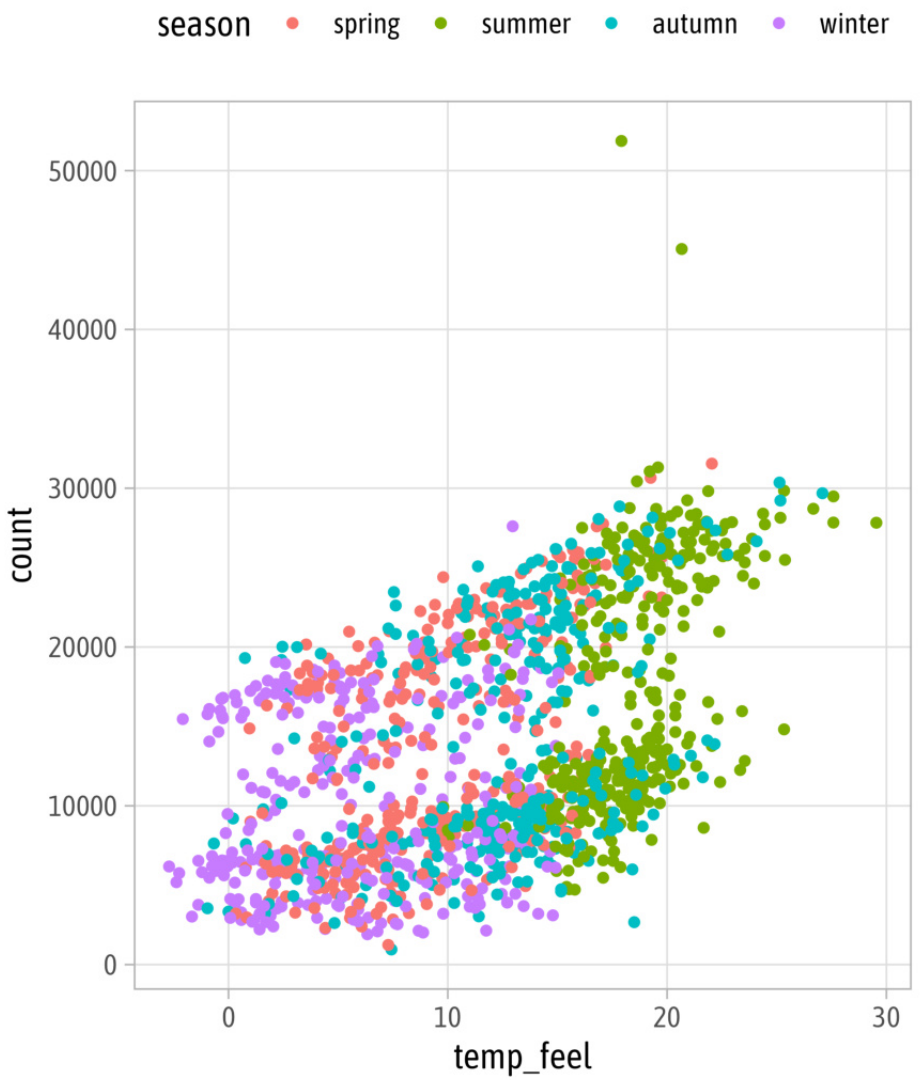
- height (continuous)
- weight (continuous)
- age (continuous or discrete)
- counts (discrete)

Discrete:
qualitative or categorical data

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

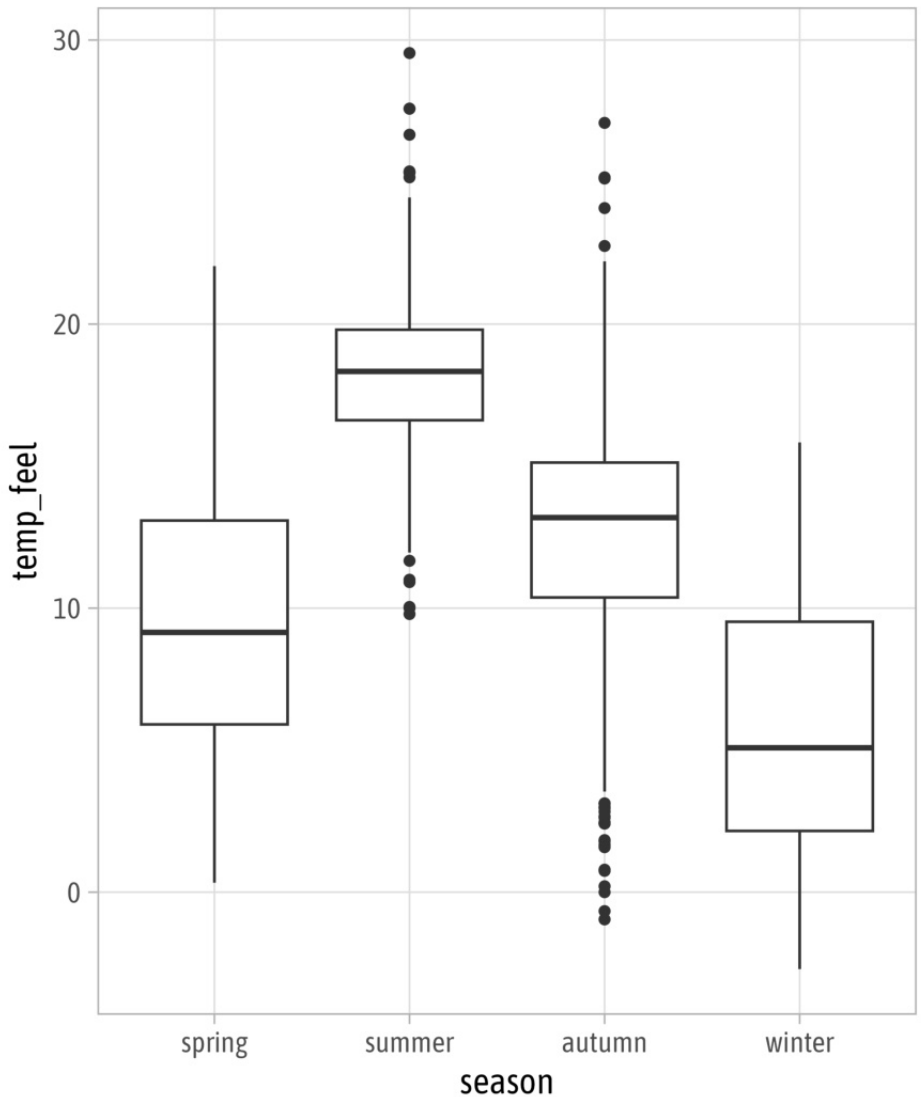
Aesthetics + Scales

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = season)
  ) +
  geom_point() +
  scale_x_continuous() +
  scale_y_continuous() +
  scale_color_discrete()
```



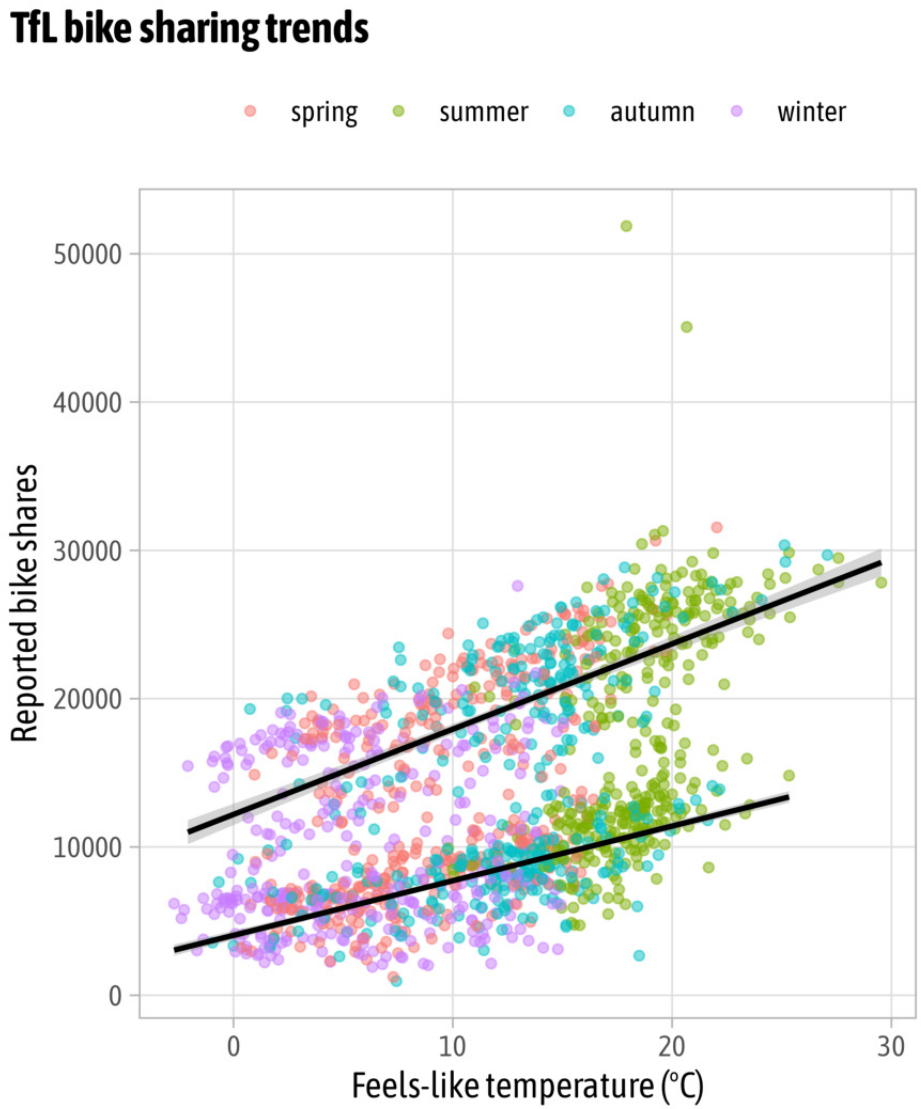
Aesthetics + Scales

```
ggplot(
    bikes,
    aes(x = season, y = temp_feel)
  ) +
  geom_boxplot() +
  scale_x_discrete() +
  scale_y_continuous()
```



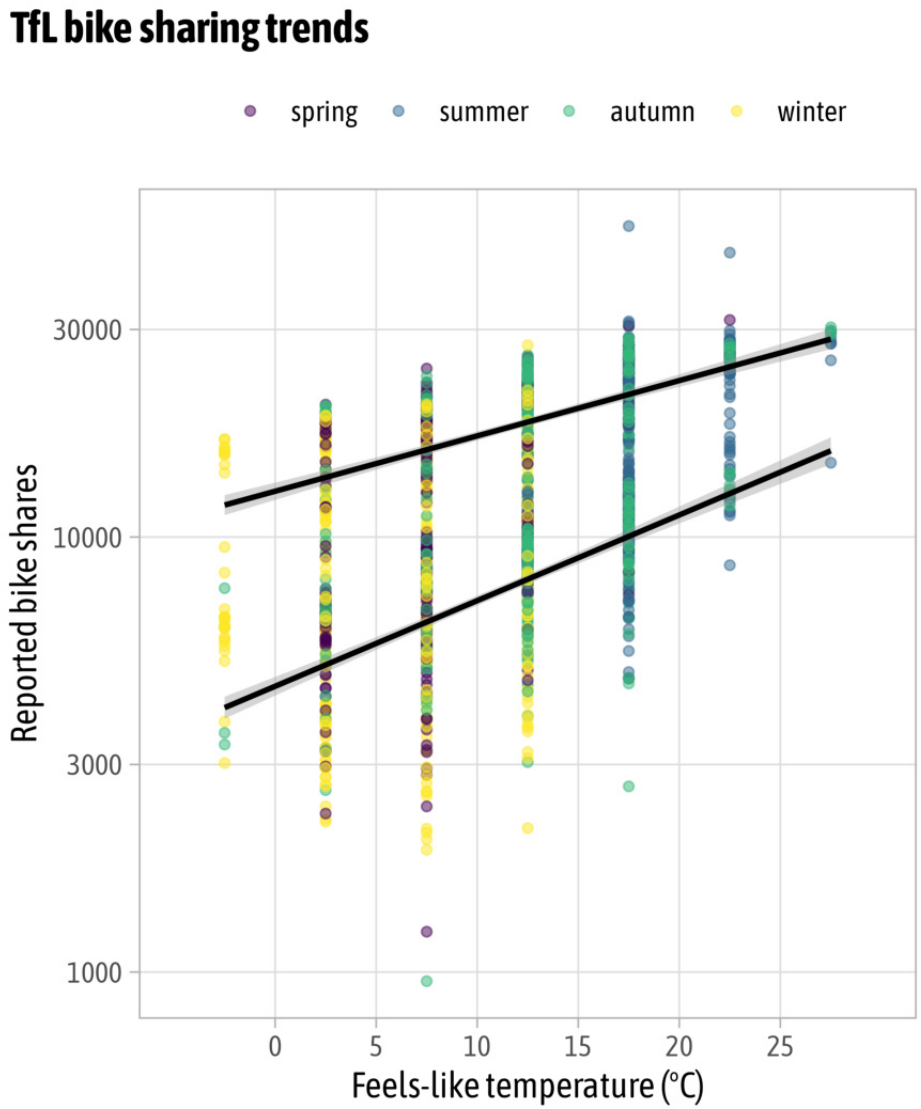
Aesthetics + Scales

```
g +
  scale_x_continuous() +
  scale_y_continuous() +
  scale_color_discrete()
```



Overwrite Scales

```
1 g +
2   scale_x_binned() +
3   scale_y_log10() +
4   scale_color_viridis_d()
```



Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0)
  ) +
  scale_y_continuous() +
  scale_color_discrete()
```

Modify Scales

```
1 g +
2   scale_x_continuous(
3     expand = c(mult = 0, add = 0),
4     breaks = seq(0, 30, by = 5)
5   ) +
6   scale_y_continuous() +
7   scale_color_discrete()
```

Modify Scales

```
1 g +
2   scale_x_continuous(
3     expand = c(mult = 0, add = 0),
4     breaks = seq(0, 30, by = 5),
5     labels = function(x) paste0(x, "°C")
6   ) +
7   scale_y_continuous() +
8   scale_color_discrete()
```

Modify Scales

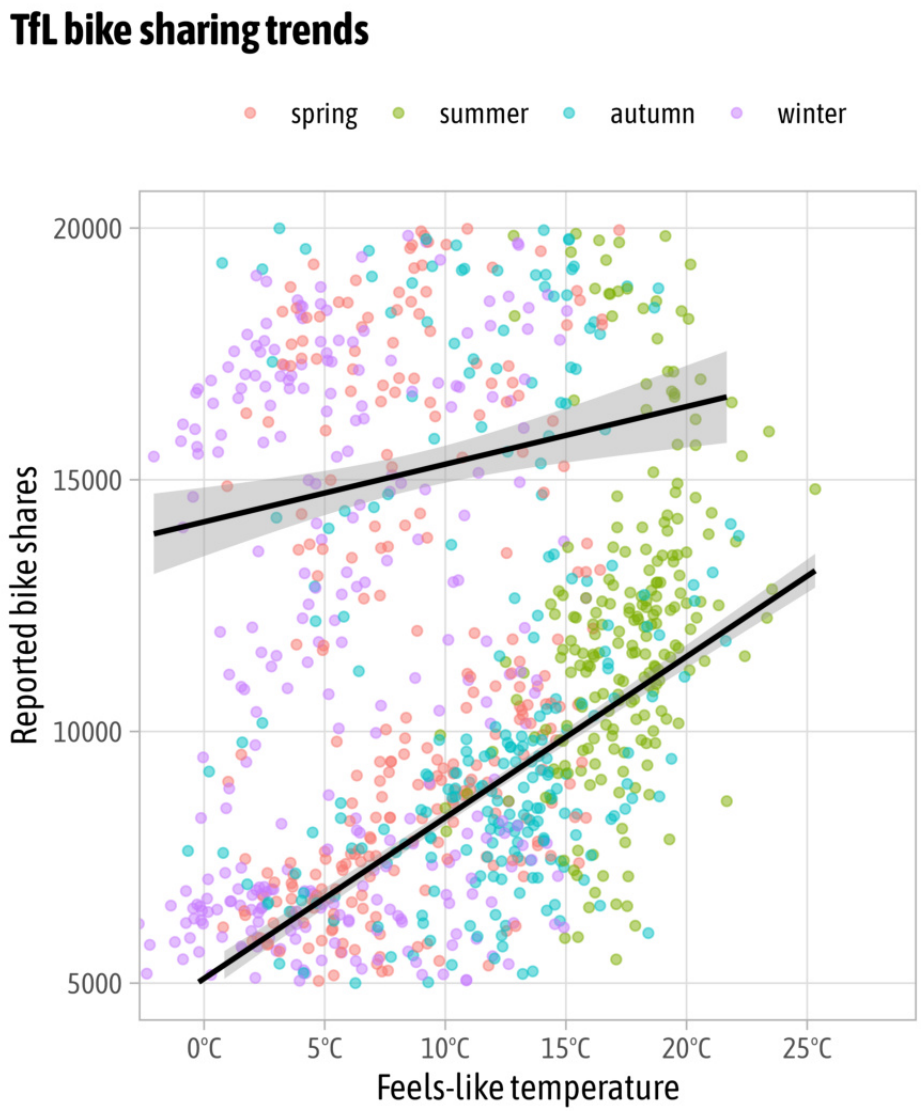
```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous() +
  scale_color_discrete()
```

Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    limits = c(0, NA)
  ) +
  scale_color_discrete()
```

Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    limits = c(5000, 20000)
  ) +
  scale_color_discrete()
```



Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = c(100, 10000, 20000, 50000)
  ) +
  scale_color_discrete()
```

Modify Scales

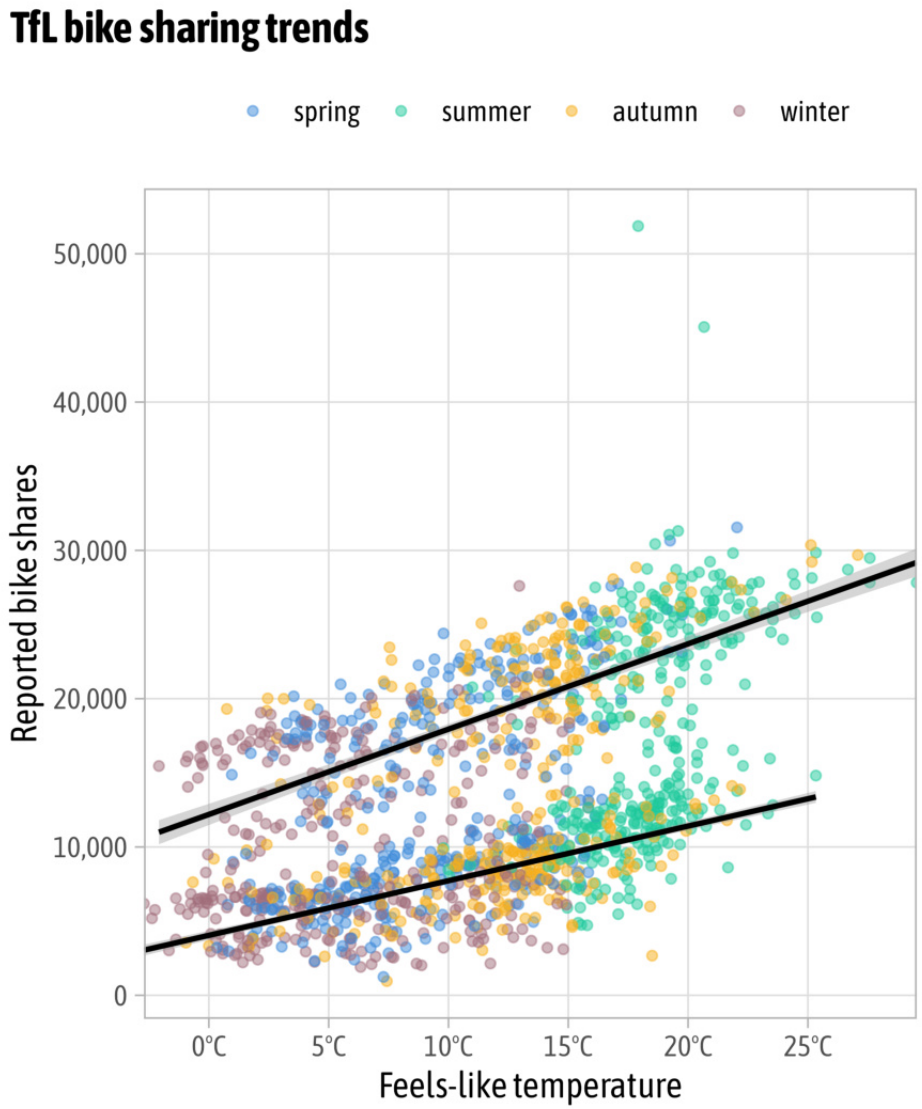
```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000
  ) +
  scale_color_discrete()
```

Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_discrete()
```

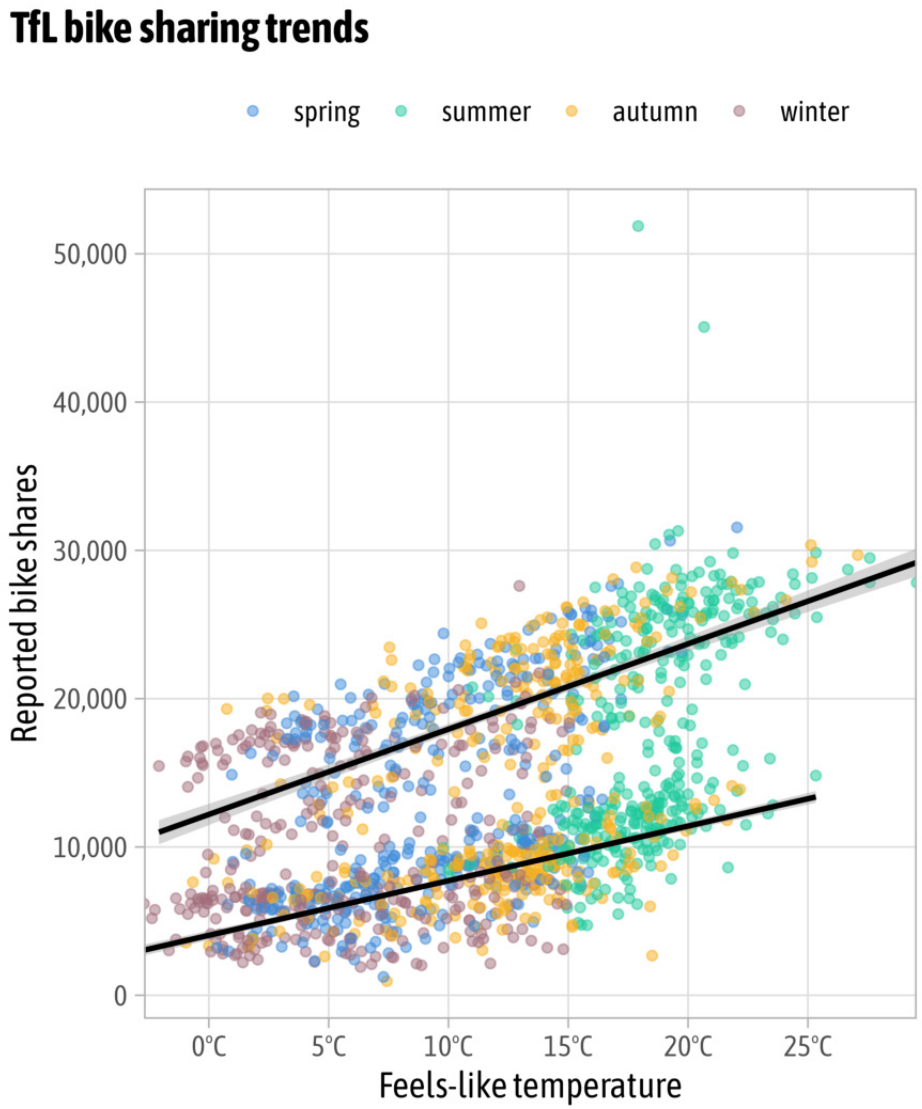
Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_discrete(
    type = c("#3c89d9", "#1ec99b", "#f7b01b",
  )
```



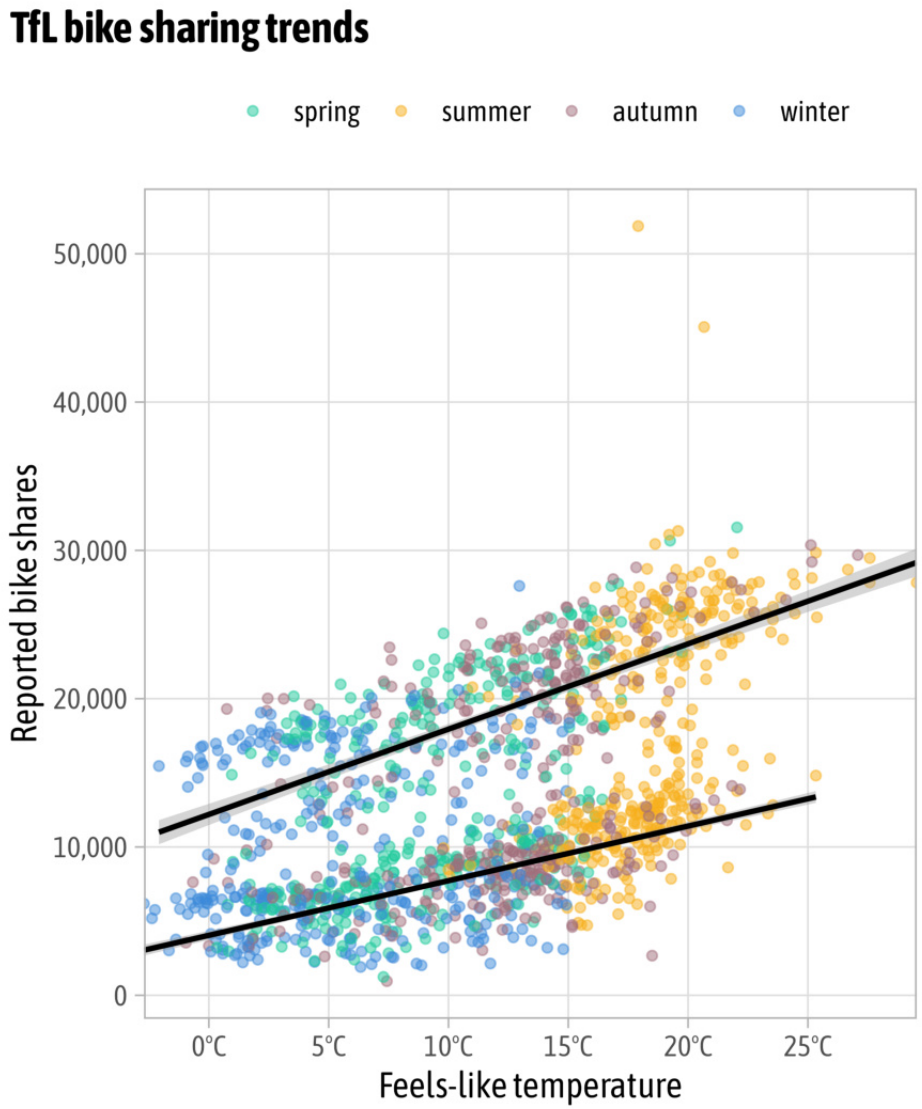
Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_manual(
    values = c("#3c89d9", "#1ec99b", "#f7b01b
  )
```



Modify Scales

```
colors_sorted <- c(
  autumn = "#a26e7c",
  spring = "#1ec99b",
  summer = "#f7b01b",
  winter = "#3c89d9"
)

g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_manual(
    values = colors_sorted
  )
```

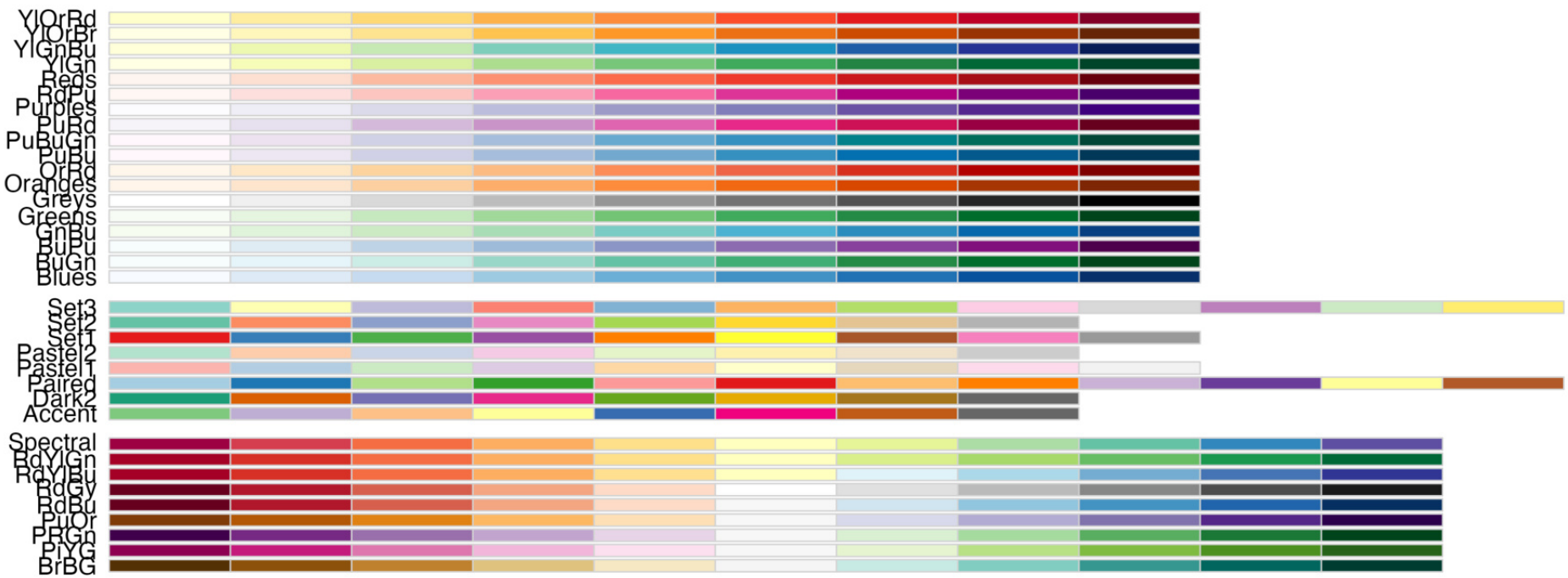


Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_brewer(
    palette = "Dark2"
  )
```

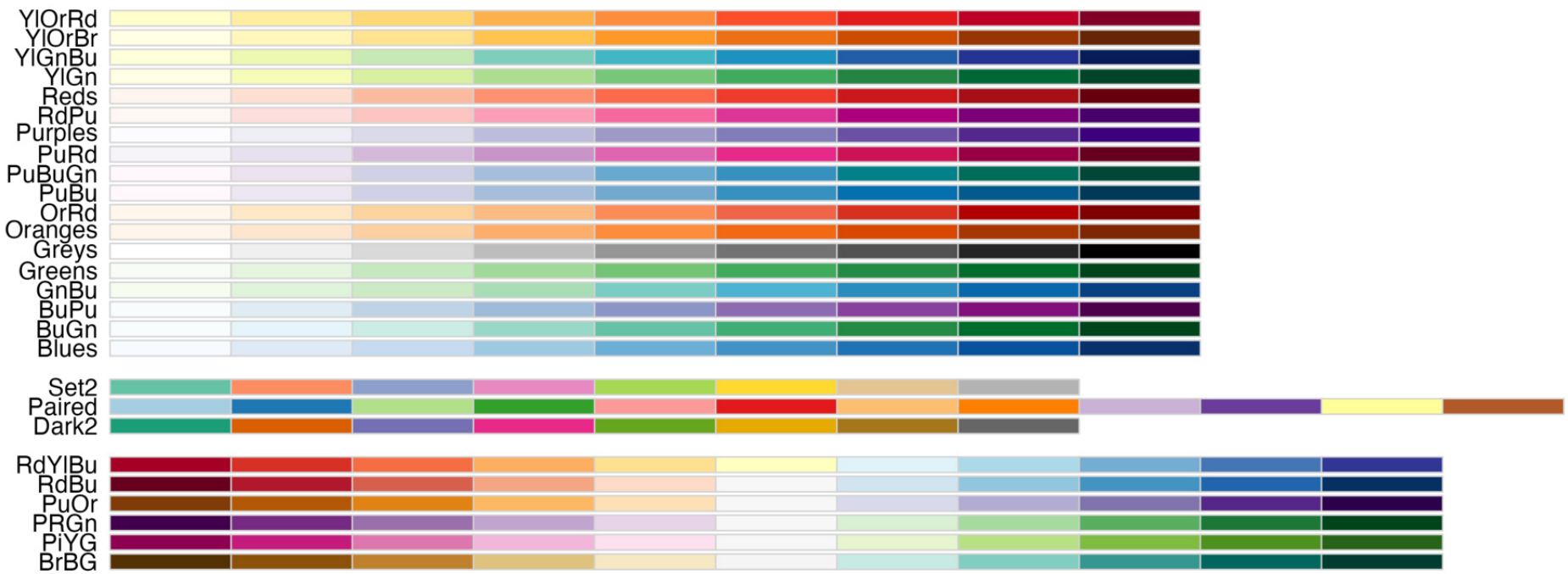
{RColorBrewer}

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



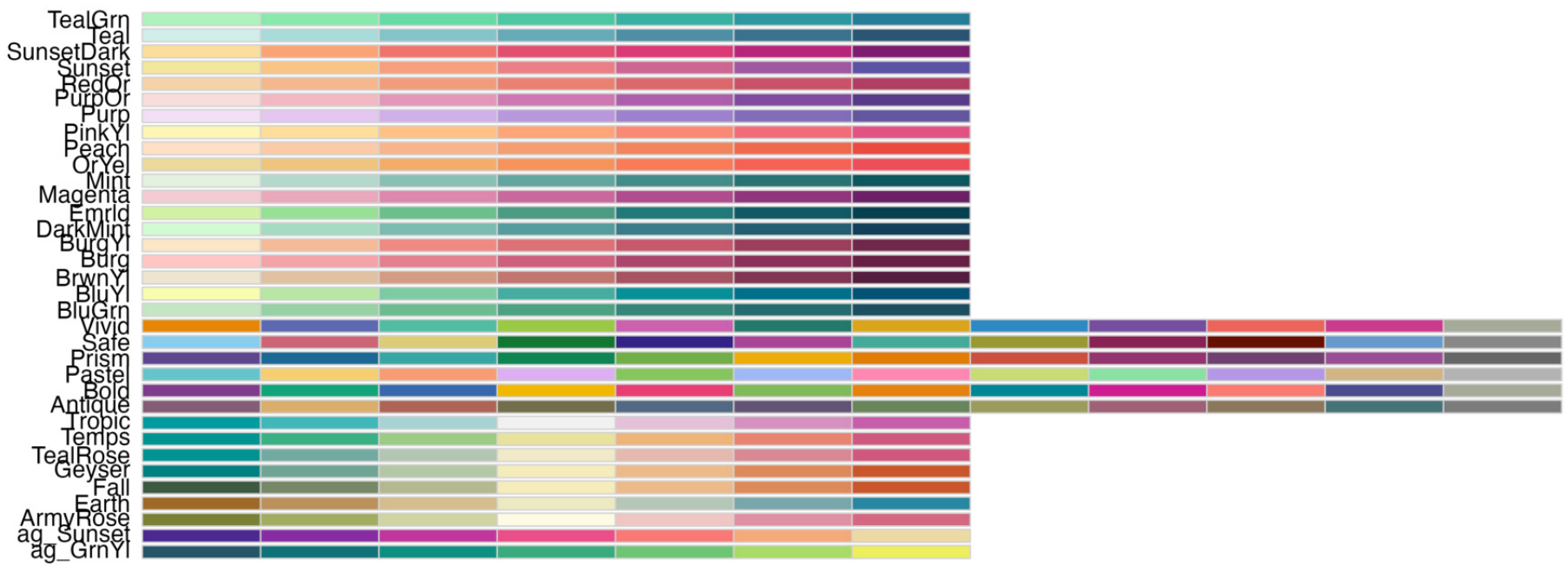
{RColorBrewer}

```
1  RColorBrewer::display.brewer.all(colorblindFriendly = TRUE)
```



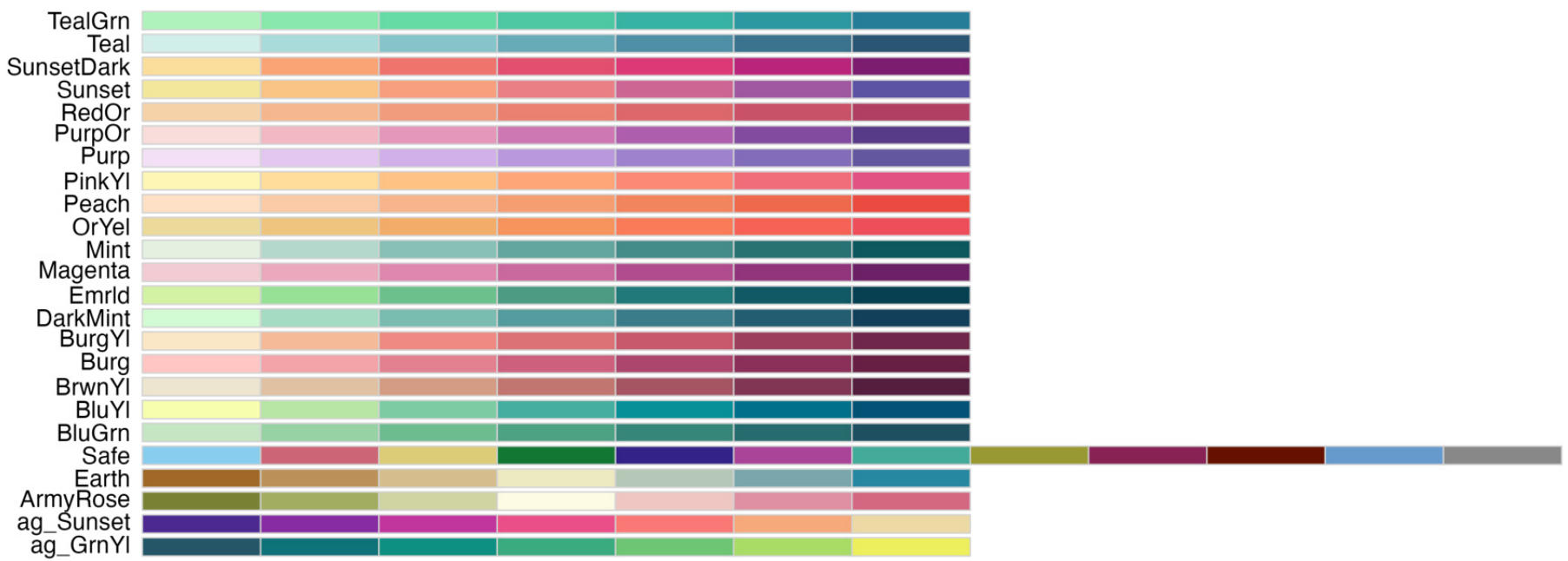
{rcartocolor}

```
1 # install.packages("rcartocolor")
2 rcartocolor::display_carto_all()
```



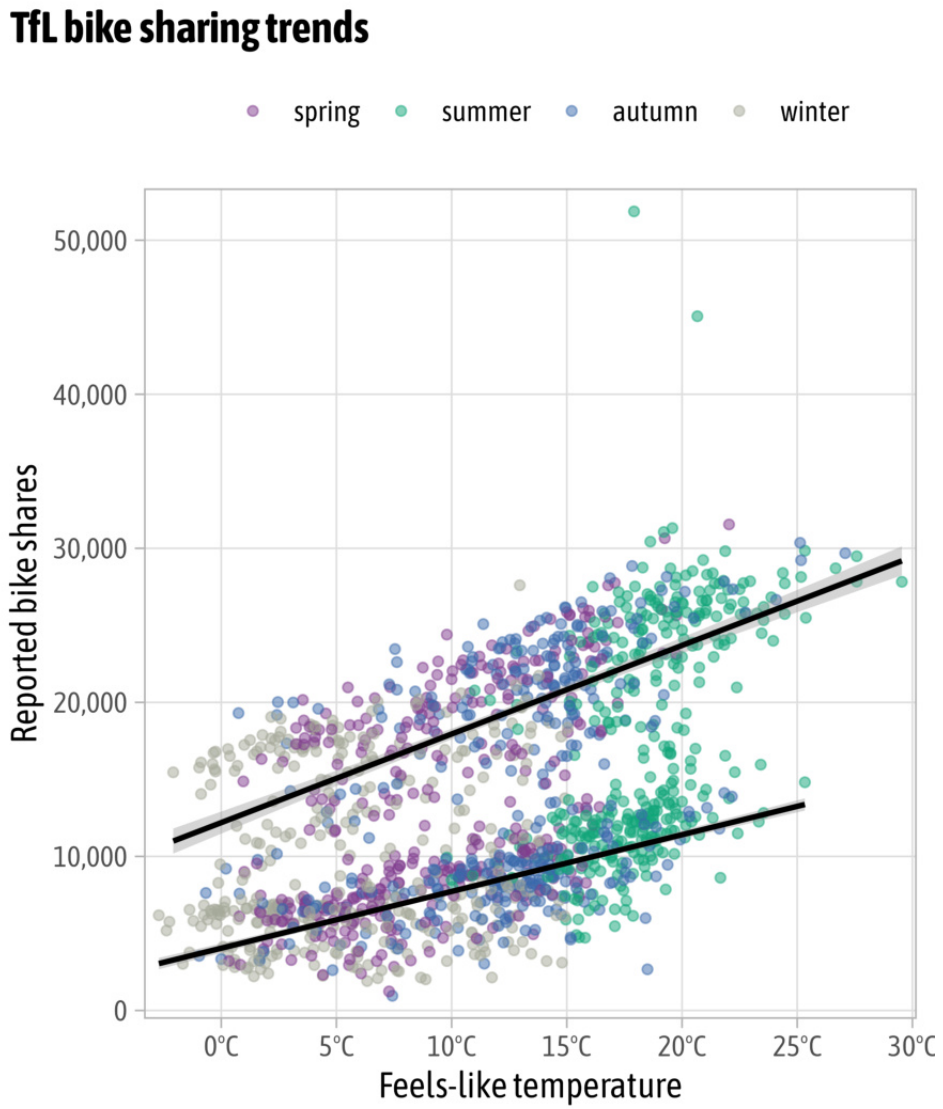
{rcartocolor}

```
# install.packages("rcartocolor")
rcartocolor::display_carto_all(colorblind_friendly = TRUE)
```



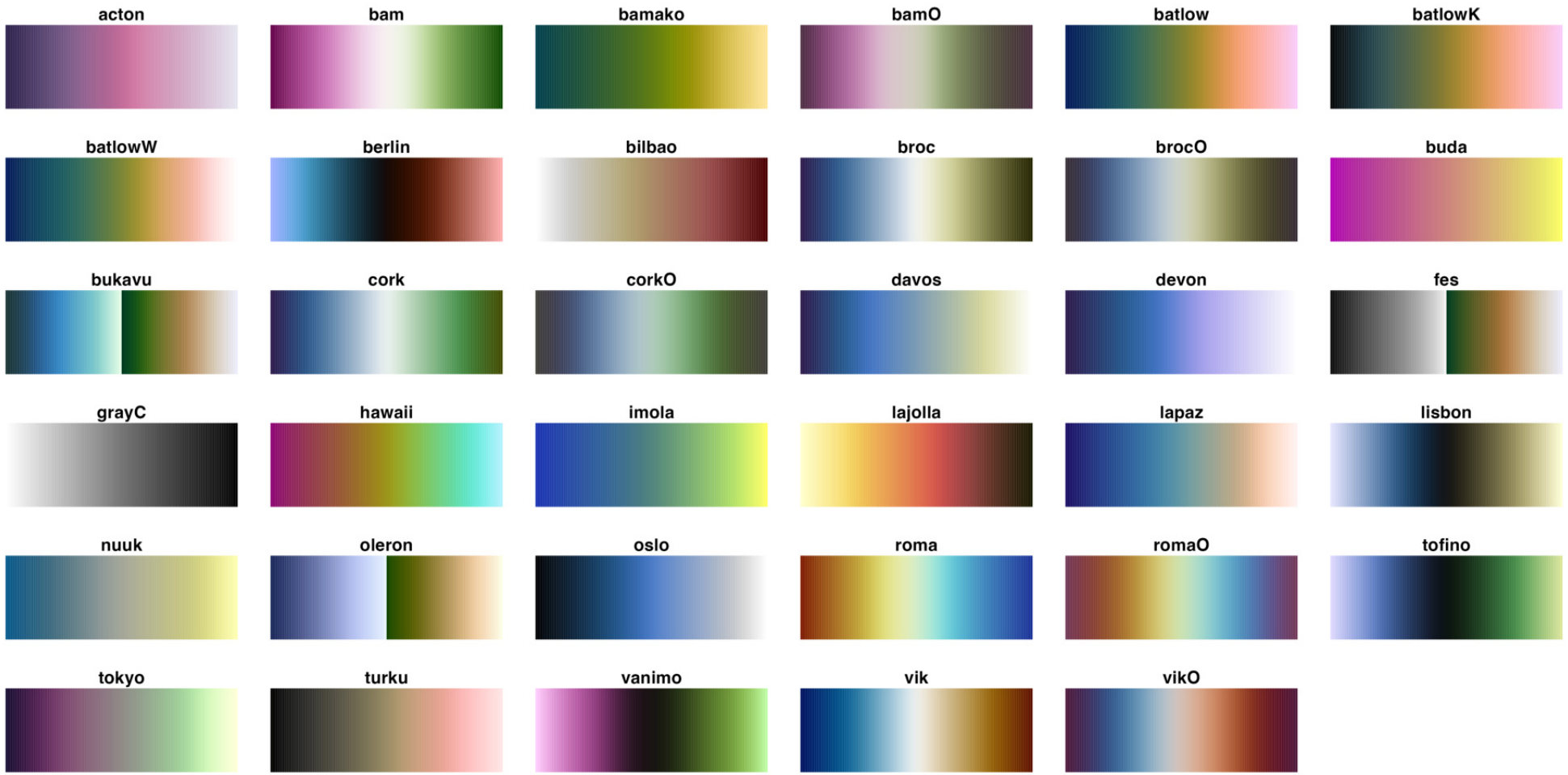
{rcartocolor}

```
g +
  scale_x_continuous(
    expand = c(mult = 0.02, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    expand = c(mult = 0, add = 1500),
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  rcartocolor::scale_color_carto_d(
    palette = "Bold"
  )
```



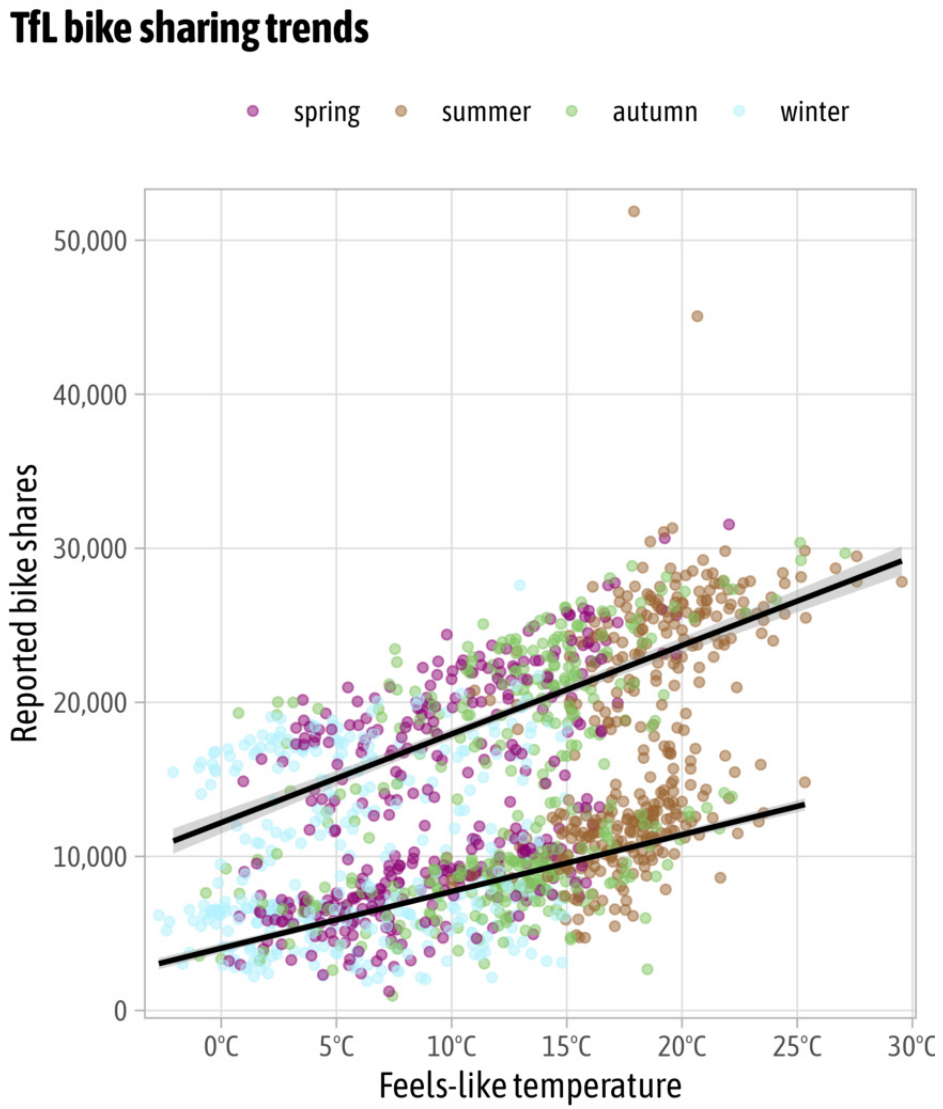
{scico}

```
1 # install.packages("scico")
2 scico::scico_palette_show()
```



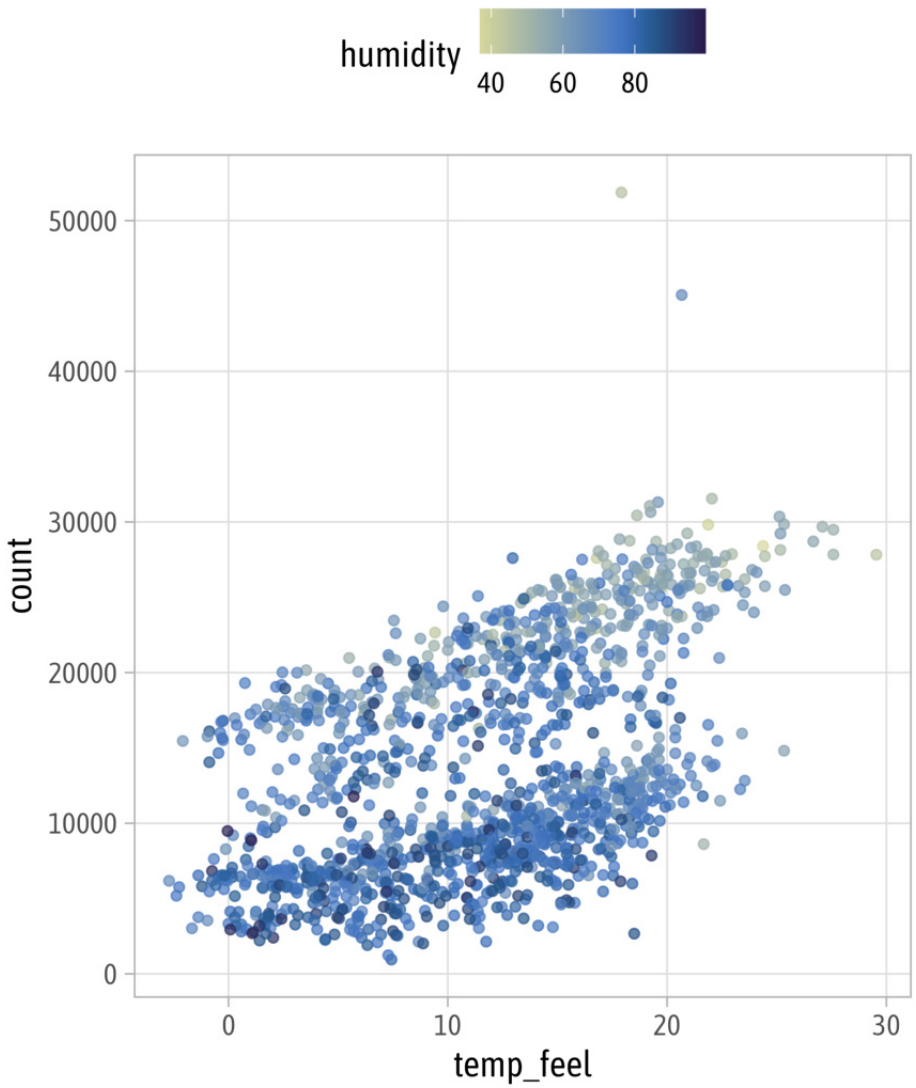
{scico}

```
g +
  scale_x_continuous(
    expand = c(mult = 0.02, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    expand = c(mult = 0, add = 1500),
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scico::scale_color_scico_d(
    palette = "hawaii"
  )
```



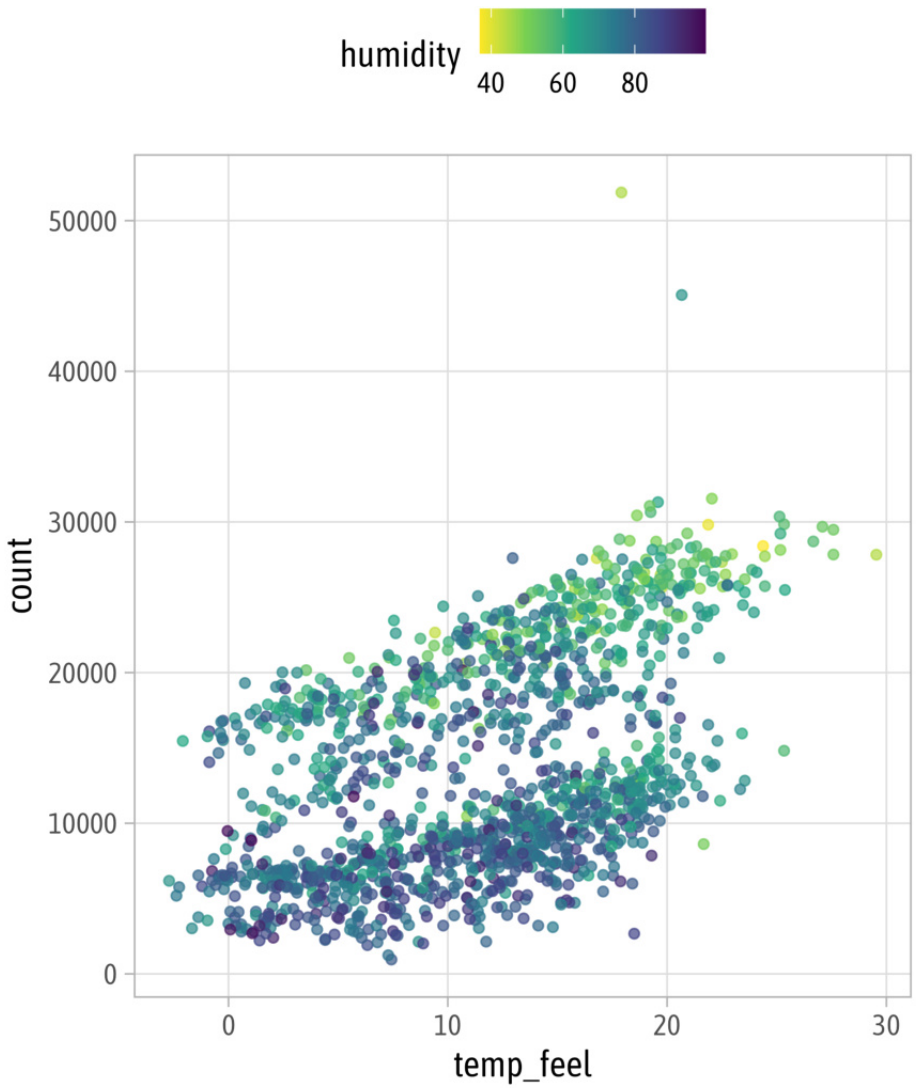
{scico}

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count,
        color = humidity)
  ) +
  geom_point(alpha = .7) +
  scico::scale_color_scico(
    palette = "davos",
    direction = -1,
    end = .8
  )
```



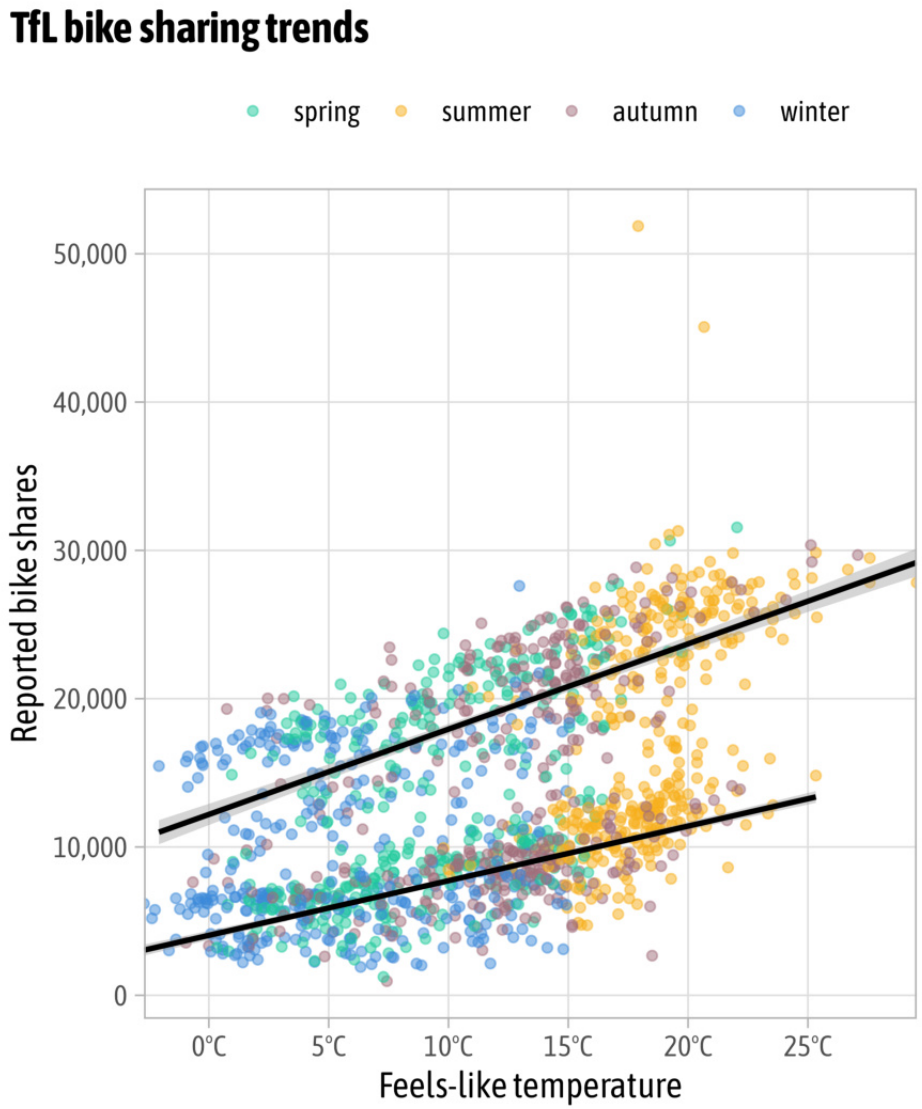
{scico}

```
1 ggplot(
2     bikes,
3     aes(x = temp_feel, y = count,
4         color = humidity)
5   ) +
6   geom_point(alpha = .7) +
7   scale_color_viridis_c(
8     direction = -1
9   )
```



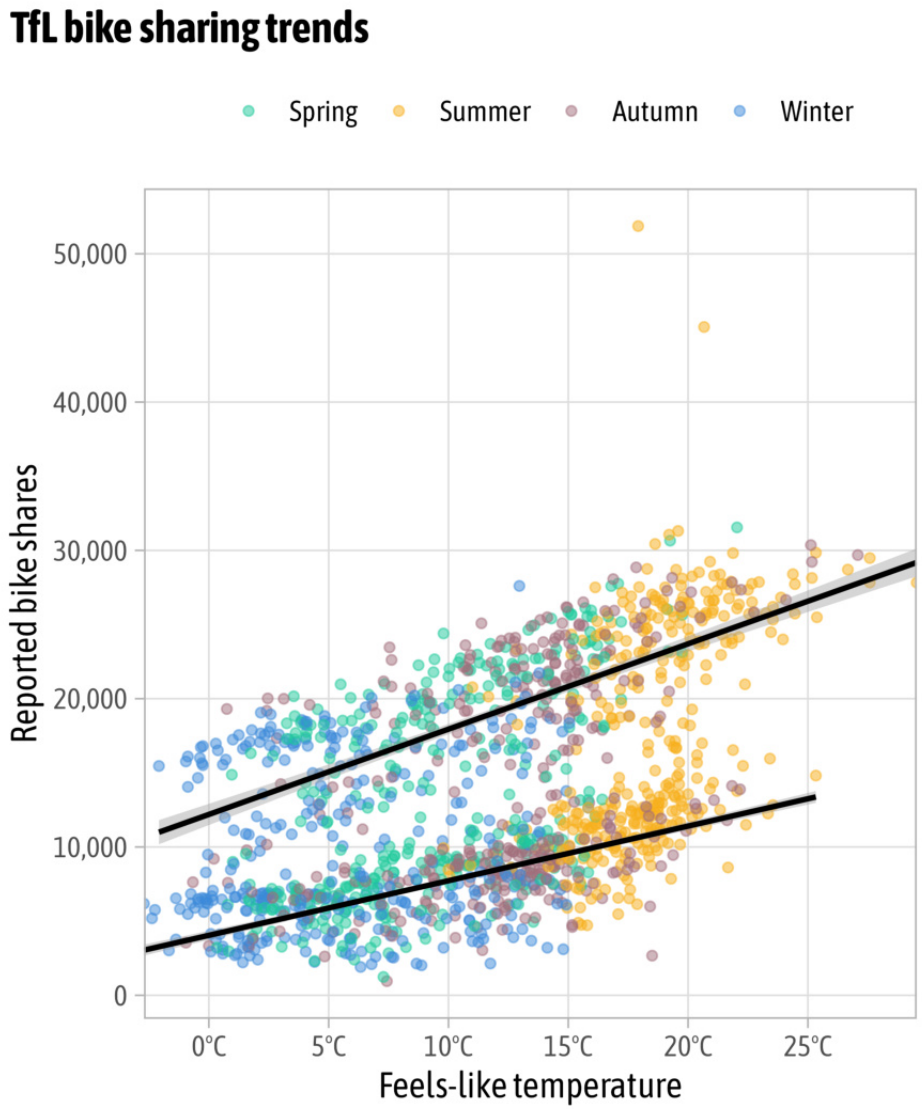
Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_manual(
    values = colors_sorted,
    name = NULL
  )
```



Modify Scales

```
g +
  scale_x_continuous(
    expand = c(mult = 0, add = 0),
    breaks = seq(0, 30, by = 5),
    labels = function(x) paste0(x, "°C"),
    name = "Feels-like temperature"
  ) +
  scale_y_continuous(
    breaks = 0:5*10000,
    labels = scales::label_comma()
  ) +
  scale_color_manual(
    values = colors_sorted,
    name = NULL,
    labels = stringr::str_to_title
  )
```

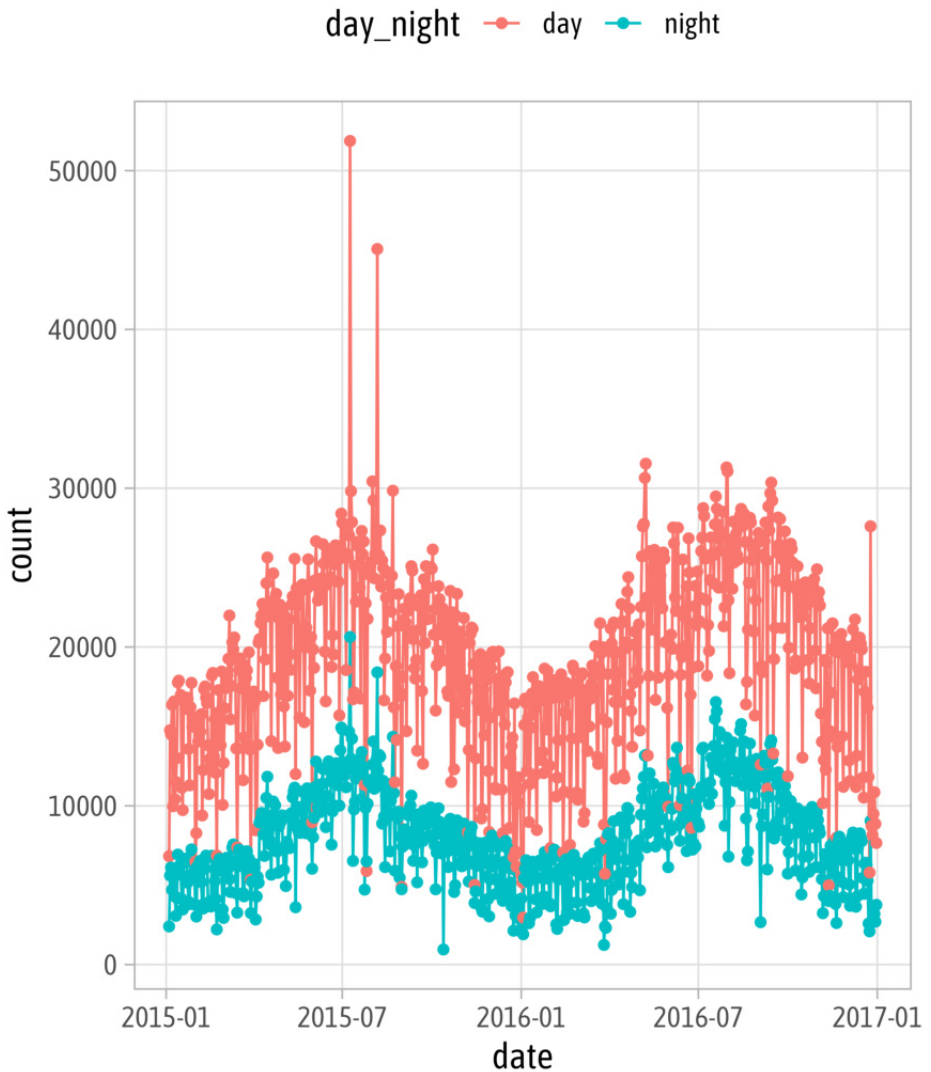


Your Turn

- **Style the time series of reported bike shares.**
  - Add a plot title and meaningful axis and legend titles.
  - Use a custom set of colors for day and night.
  - Explore complete themes and pick your favorite.
  - **Bonus:** Modify the x axis to show every four months along with the year.

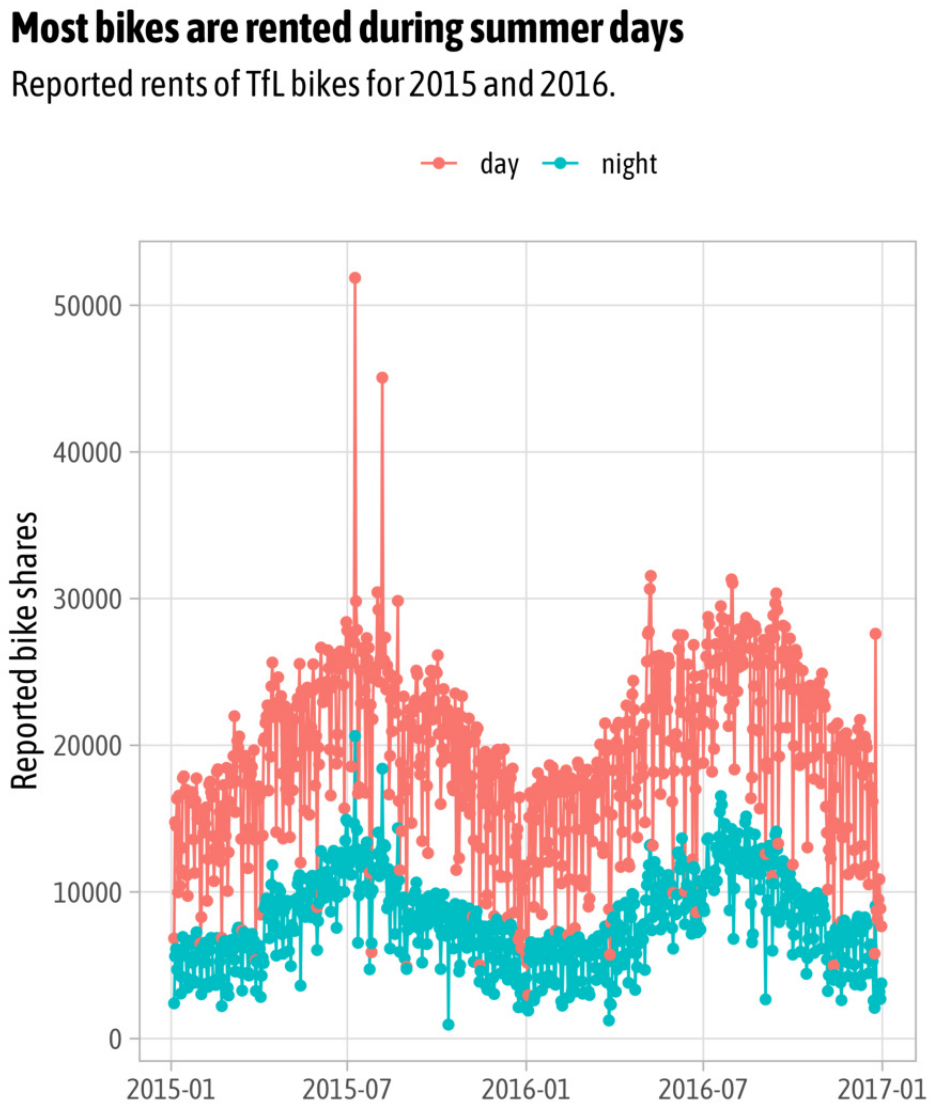
Solution Exercise

```
1 (p <-
2    ggplot(
3     bikes,
4     aes(x = date, y = count,
5         color = day_night)
6   ) +
7   geom_line() +
8   geom_point()
9 )
```



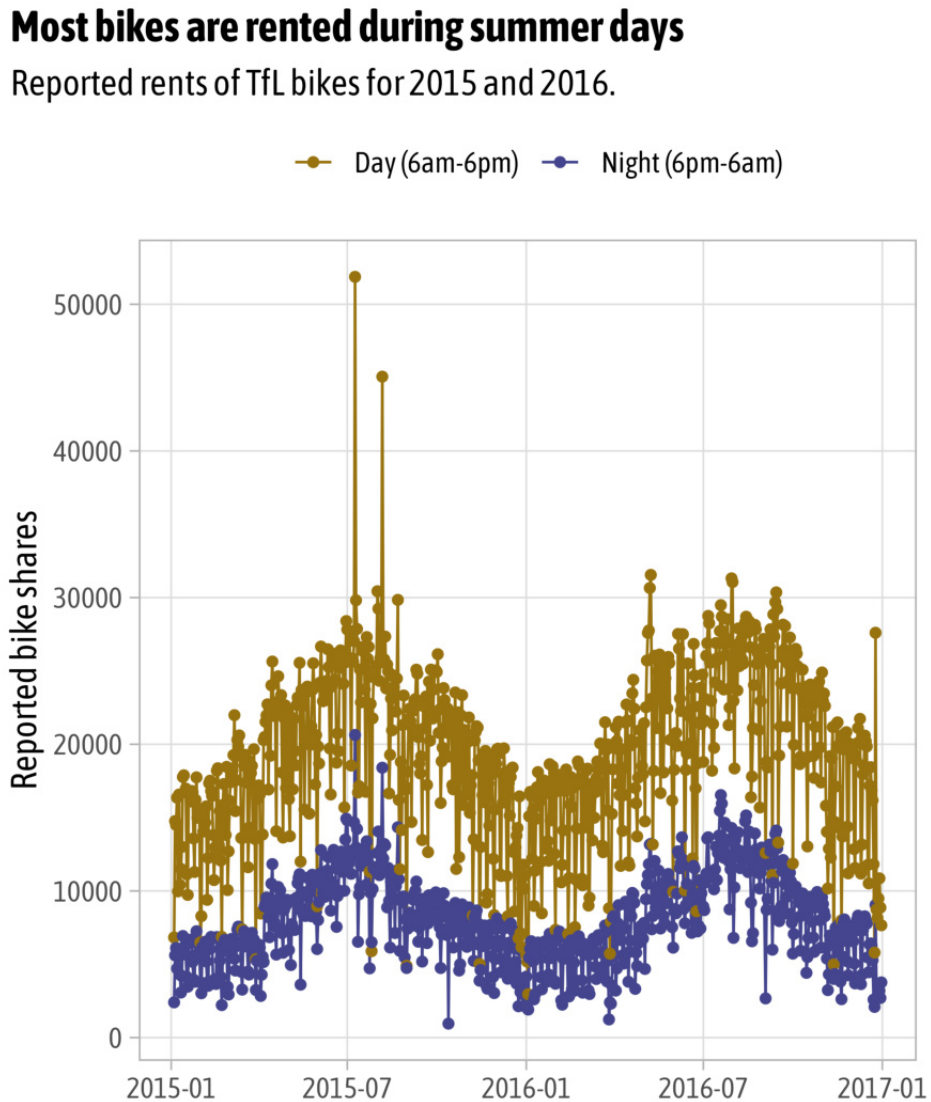
Solution Exercise

```
1 p +
2   labs(
3     title = "Most bikes are rented during sum
4     subtitle = "Reported rents of TfL bikes f
5     x = NULL,
6     y = "Reported bike shares",
7     color = NULL
8   )
```



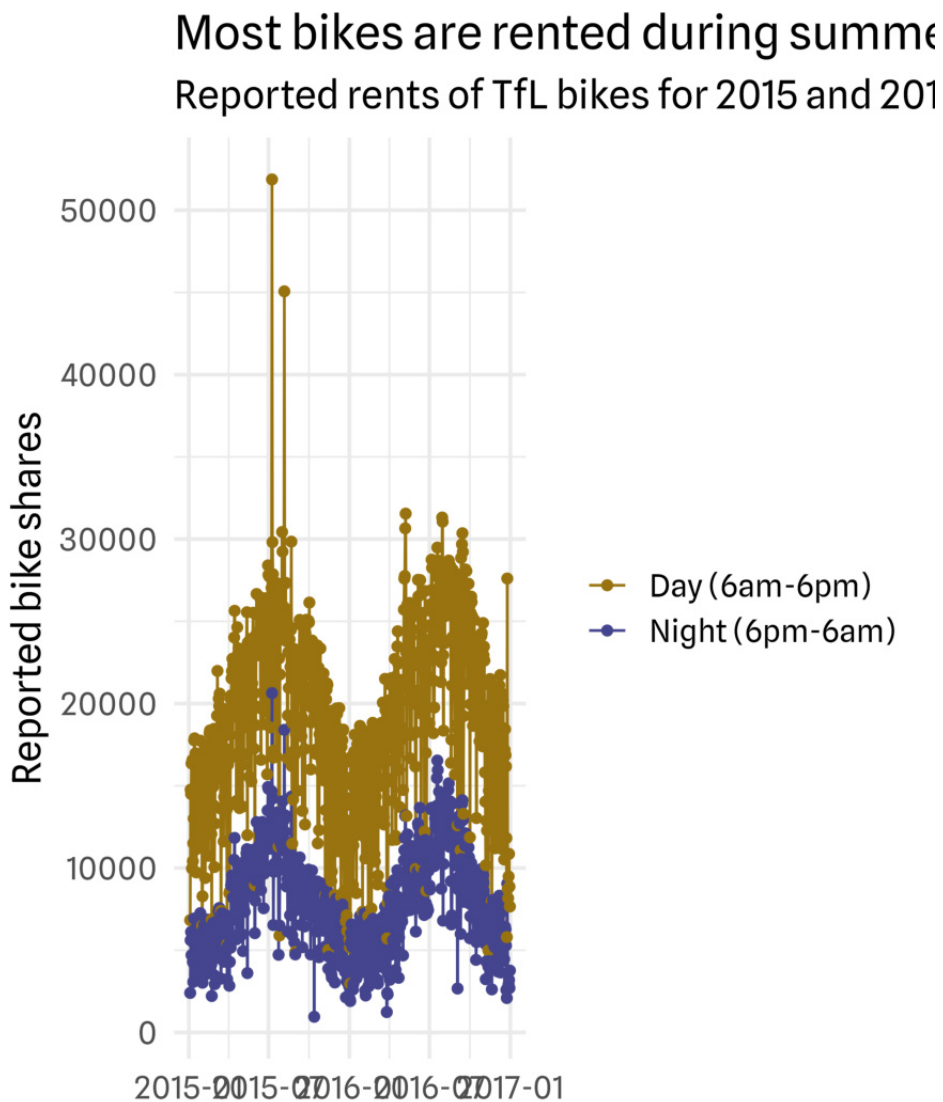
Solution Exercise

```
1  p +
2    scale_color_manual(
3      values = c("#98730F", "#44458e"),
4      labels = c("Day (6am-6pm)", "Night (6pm-6
5    ) +
6    labs(title = "Most bikes are rented during
7         subtitle = "Reported rents of TfL bike
8         x = NULL, y = "Reported bike shares",
```



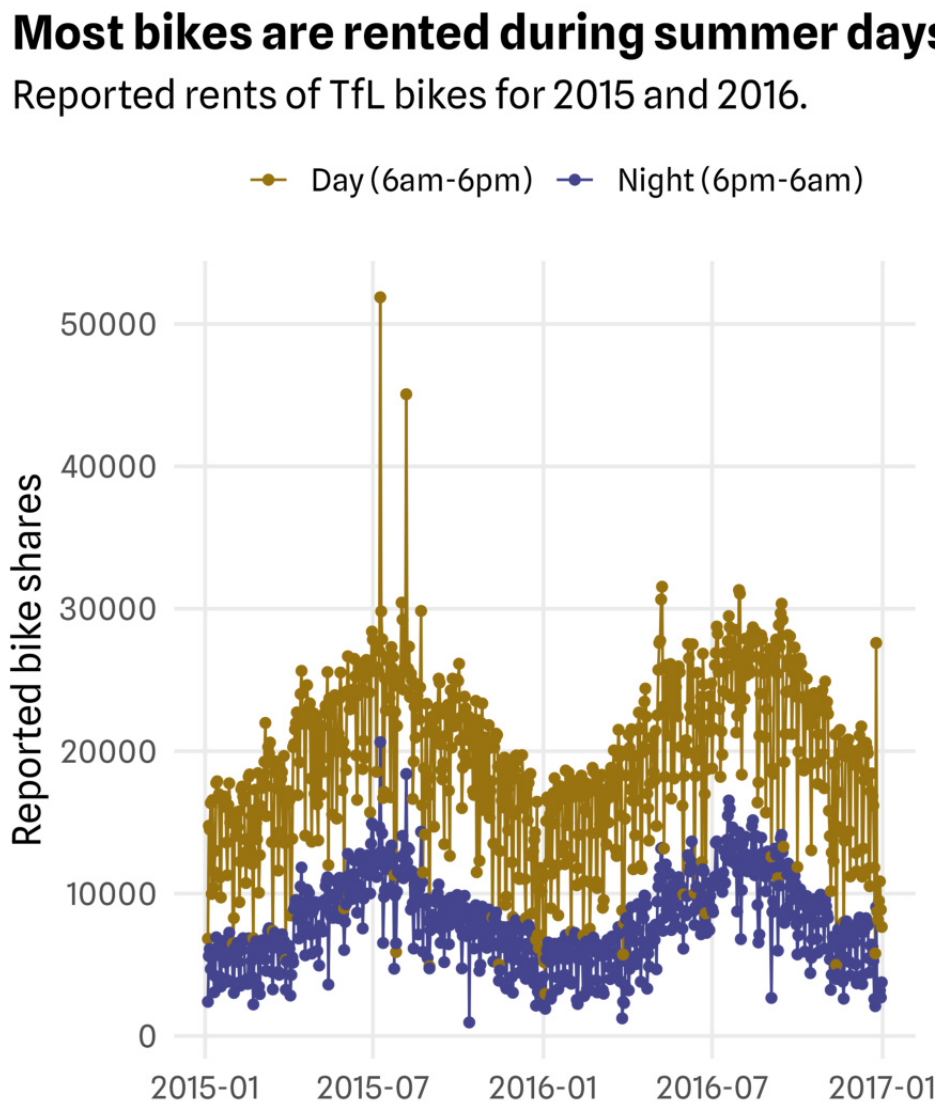
Solution Exercise

```
p +
  scale_color_manual(
    values = c("#98730F", "#44458e"),
    labels = c("Day (6am-6pm)", "Night (6pm-6
  ) +
  theme_minimal(base_family = "Spline Sans",
  labs(title = "Most bikes are rented during
       subtitle = "Reported rents of TfL bike
       x = NULL, y = "Reported bike shares",
```



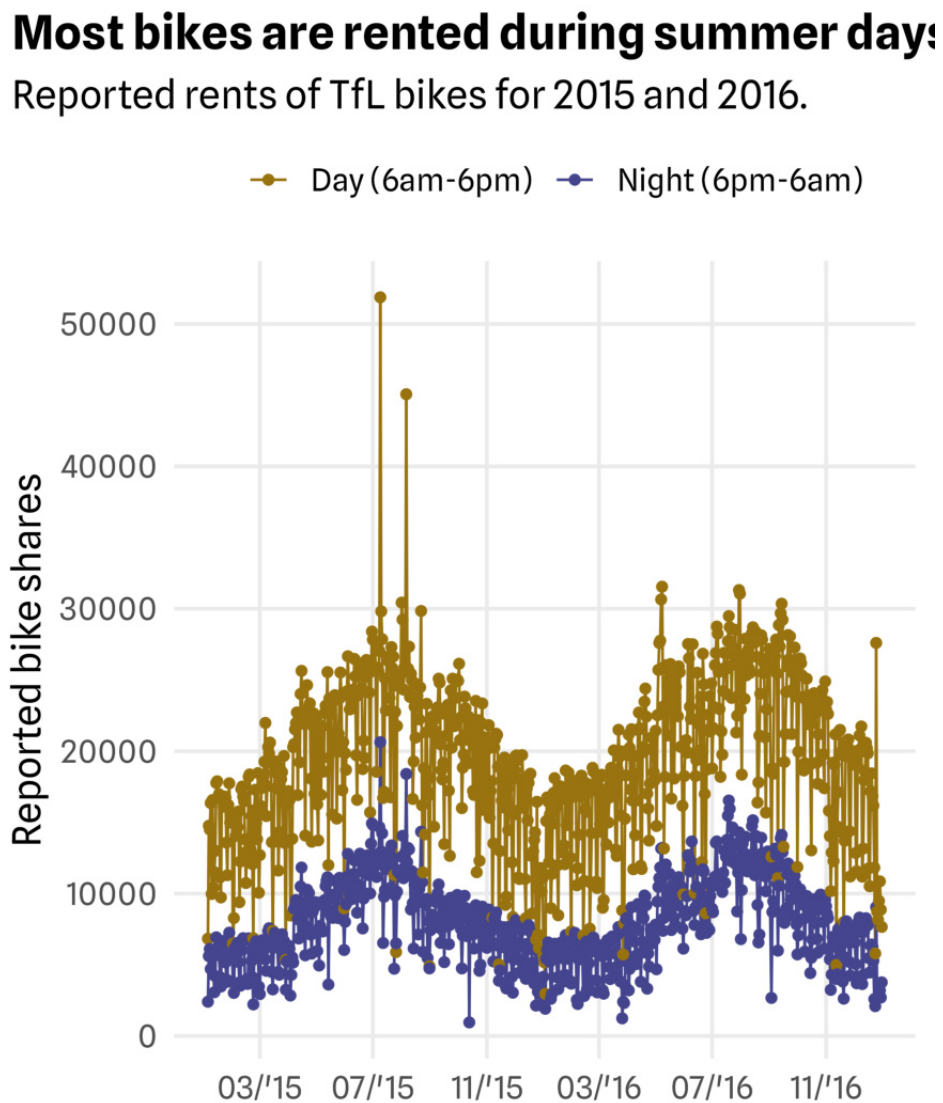
Solution Exercise

```
p +
  scale_color_manual(
    values = c("#98730F", "#44458e"),
    labels = c("Day (6am-6pm)", "Night (6pm-6
  ) +
  theme_minimal(base_family = "Spline Sans",
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold"),
    plot.title.position = "plot",
    legend.position = "top"
  ) +
  labs(title = "Most bikes are rented during
       subtitle = "Reported rents of TfL bike
       x = NULL, y = "Reported bike shares",
```



Solution Exercise

```
p +
  scale_x_date(
    date_breaks = "4 months",
    date_labels = "%m/'%y"
  ) +
  scale_color_manual(
    values = c("#98730F", "#44458e"),
    labels = c("Day (6am-6pm)", "Night (6pm-6
  ) +
  theme_minimal(base_family = "Spline Sans",
  theme(
    panel.grid.minor = element_blank(),
    plot.title = element_text(face = "bold"),
    plot.title.position = "plot",
    legend.position = "top"
  ) +
  labs(title = "Most bikes are rented during
       subtitle = "Reported rents of TfL bike
       x = NULL, y = "Reported bike shares",
```



Facets

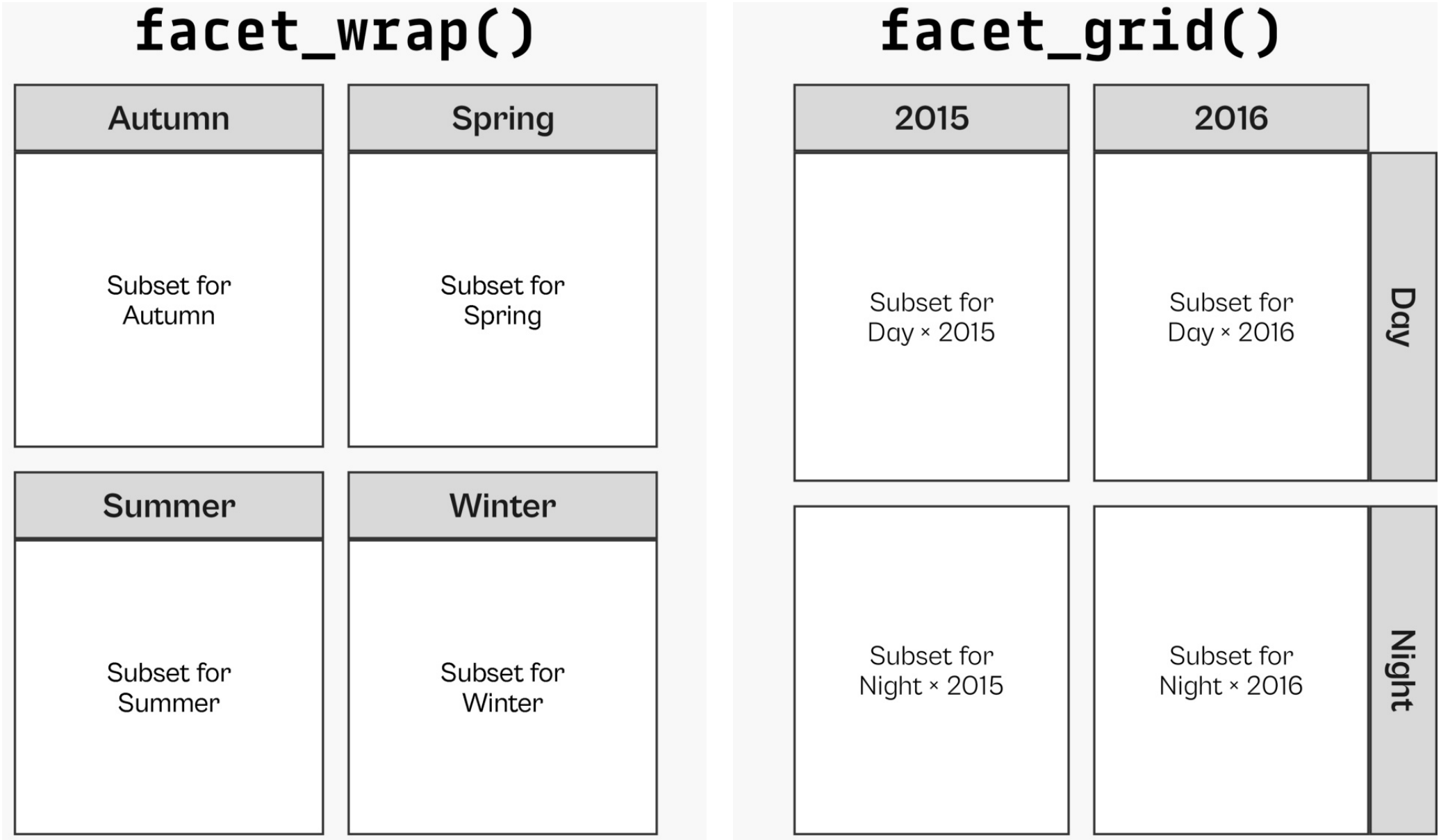
Cédric Scherer // Data Visualization Society // March 9, 2023

Facets

= split variables to multiple panels

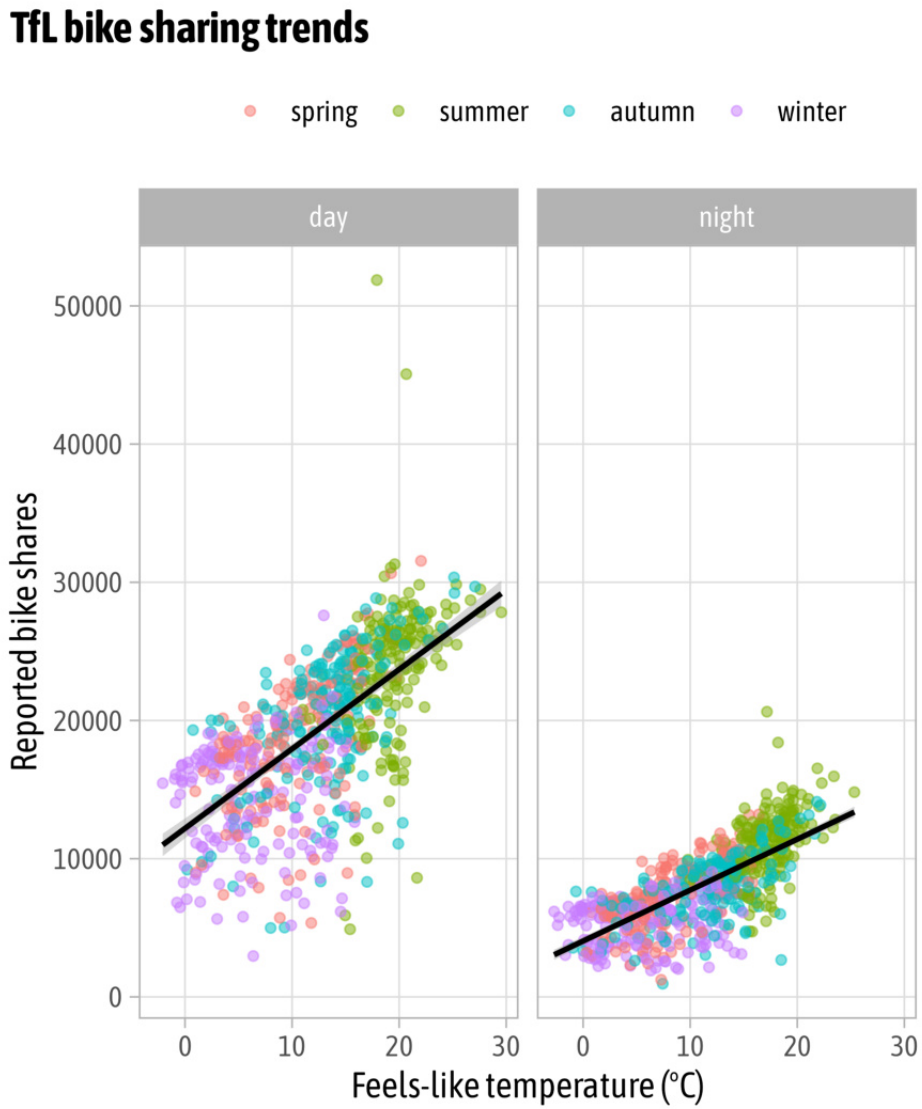
Facets are also known as:

- small multiples
- trellis graphs
- lattice plots
- conditioning


facet_wrap()
Autumn
Subset for Autumn
Spring
Subset for Spring
Summer
Subset for Summer
Winter
Subset for Winter
facet_grid()
2015
2016
Day
Subset for Day × 2015
Subset for Day × 2016
Night
Subset for Night × 2015
Subset for Night × 2016

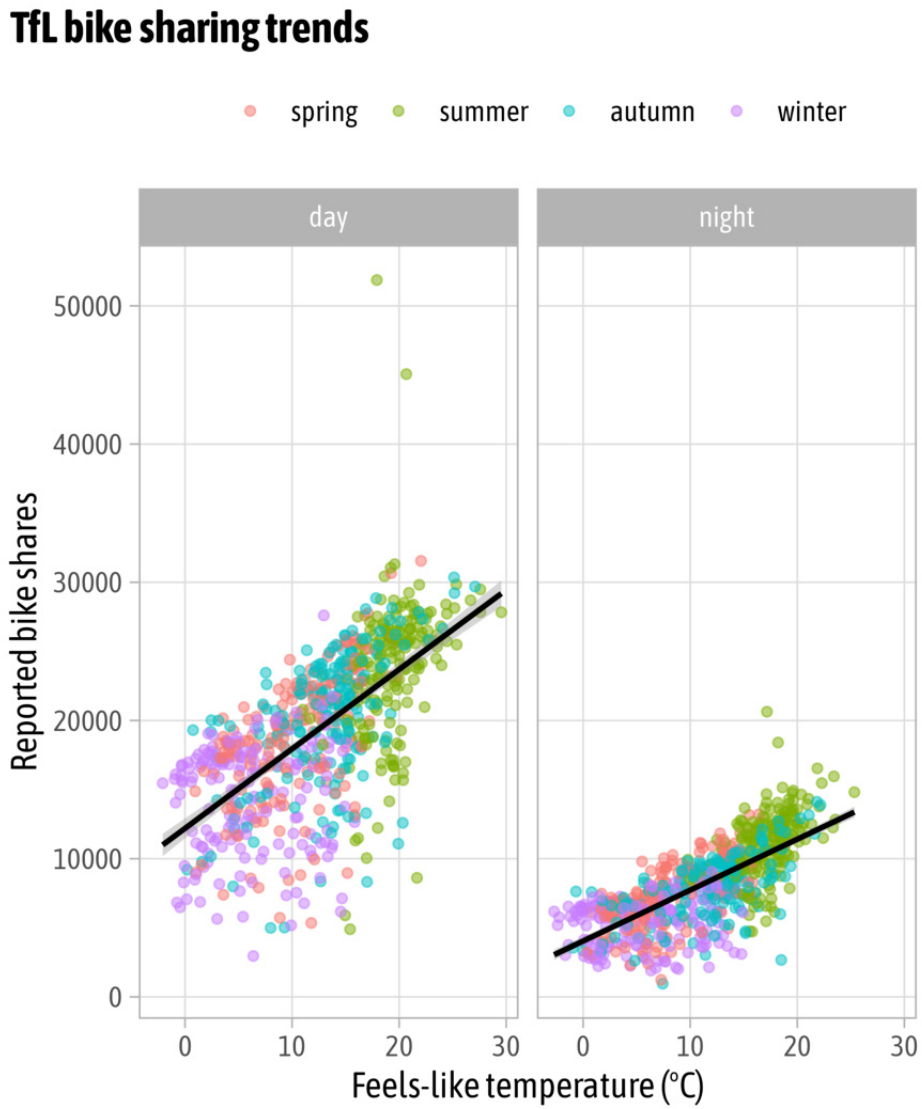
Wrapped Facets

```
g +
  facet_wrap(
    vars(day_night)
  )
```



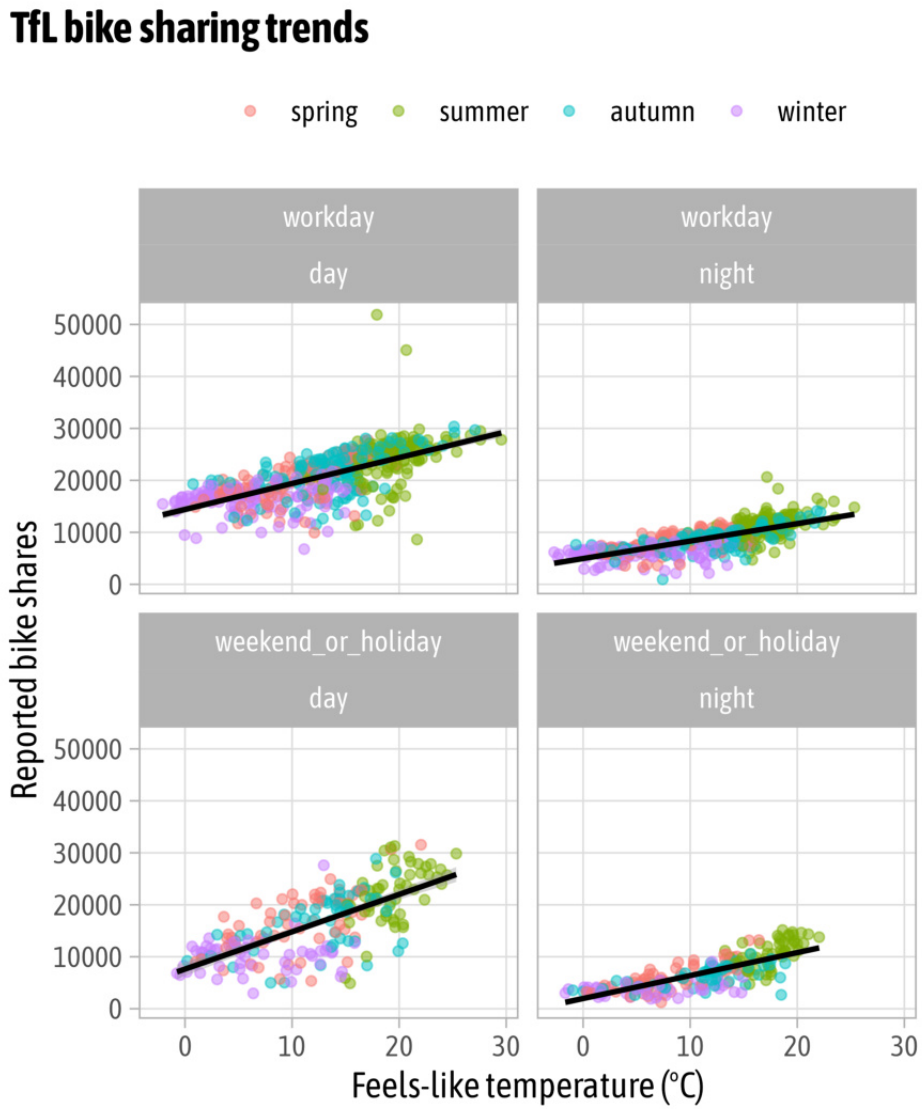
Wrapped Facets

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



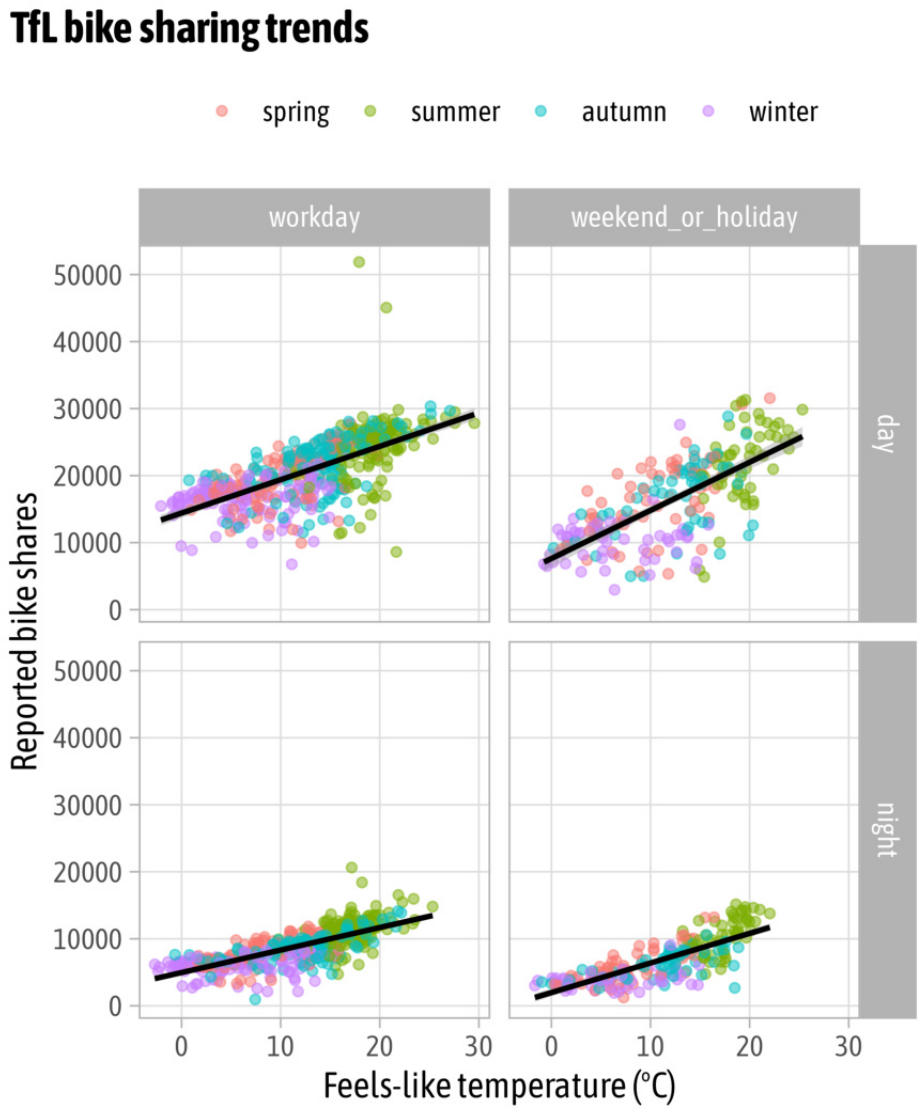
Facet Multiple Variables

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



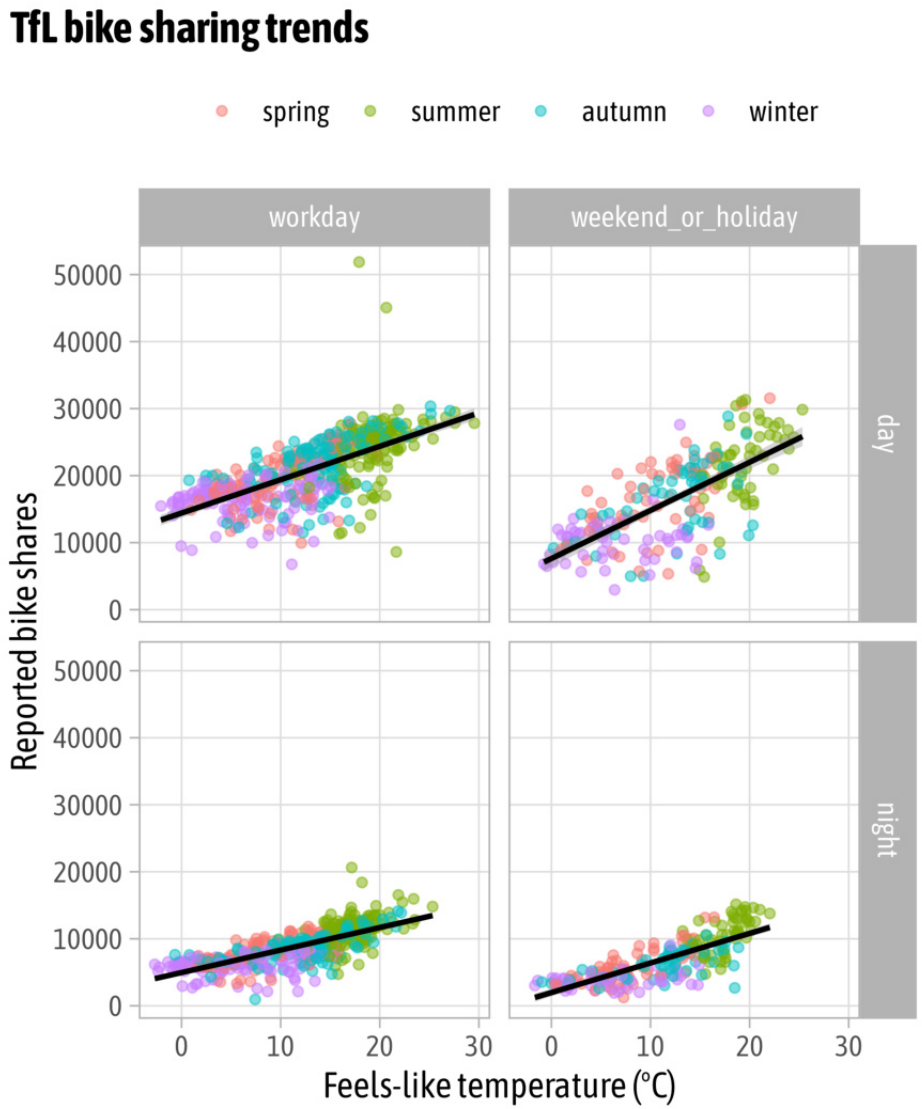
Gridded Facets

```
g +
  facet_grid(
    rows = vars(day_night),
    cols = vars(is_workday)
  )
```



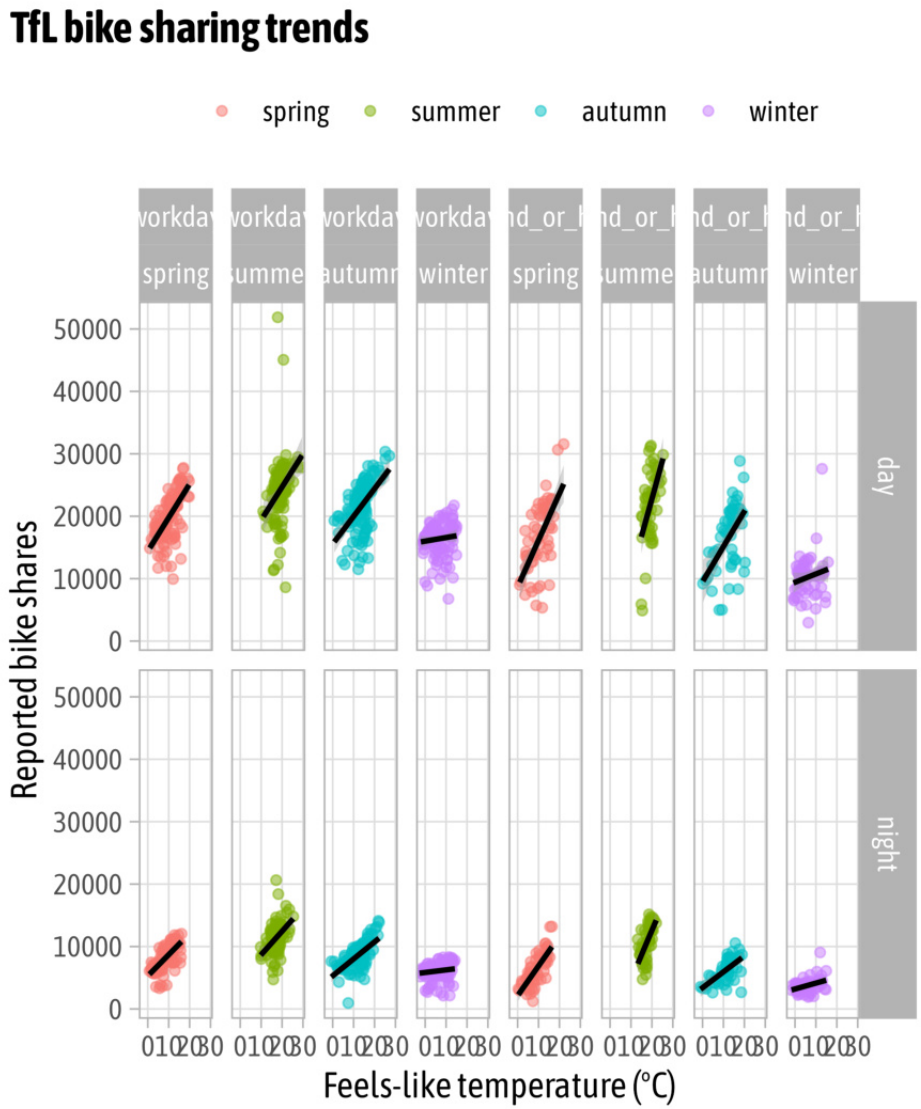
Gridded Facets

```
g +
  facet_grid(
    day_night ~ is_workday
  )
```



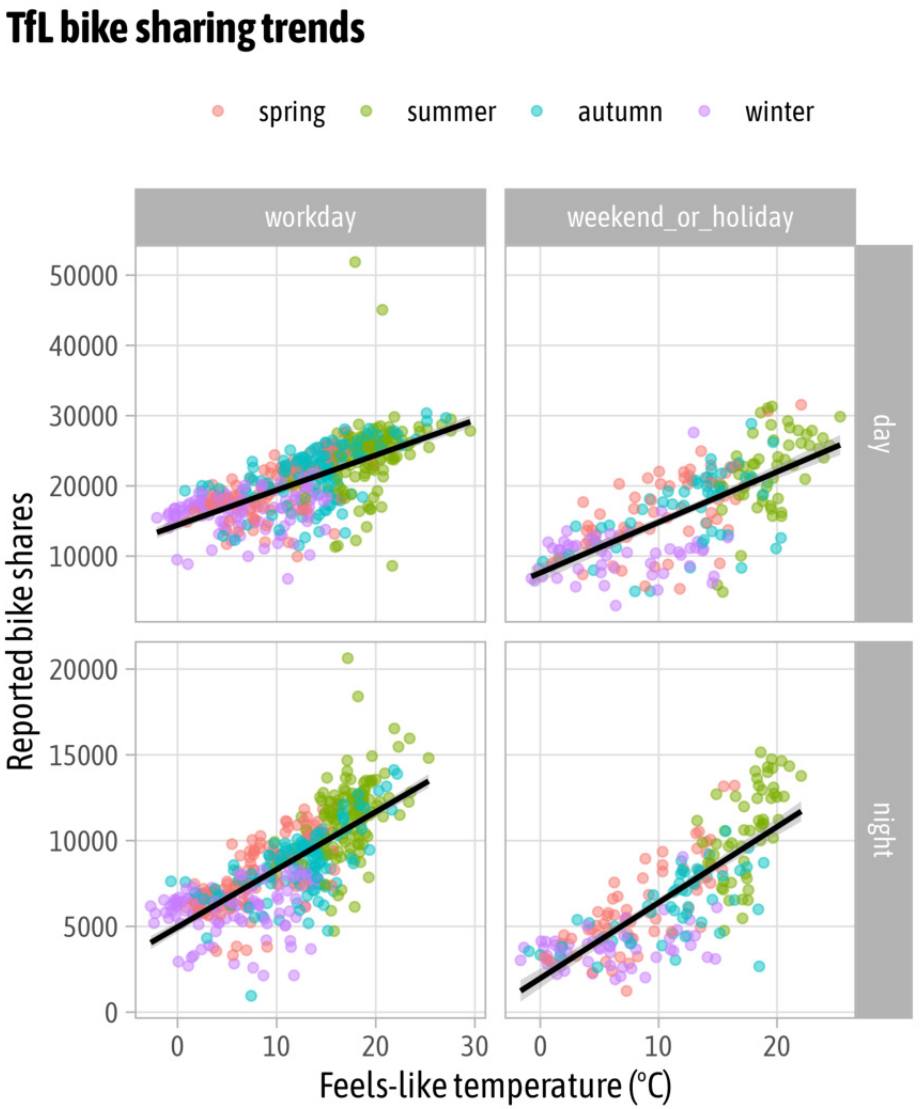
Facet Multiple Variables

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



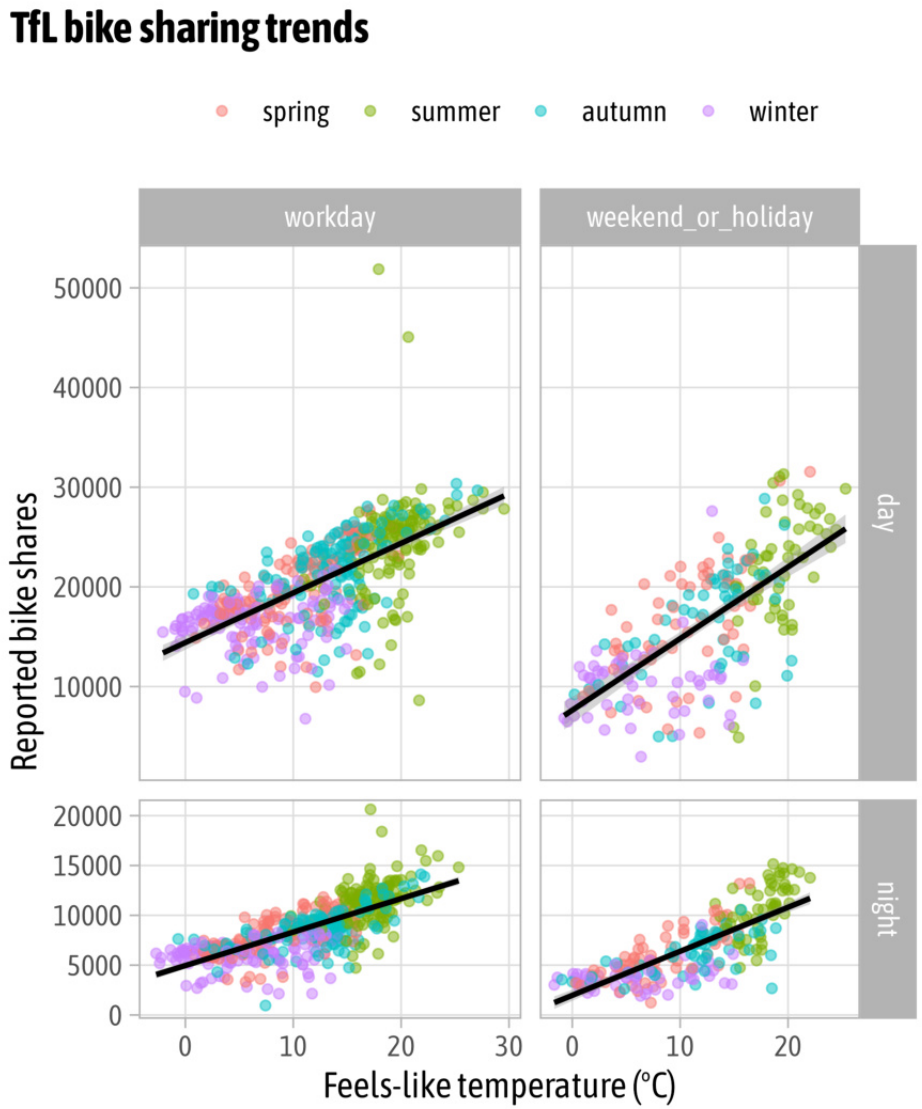
Facet Options: Free Scaling

```
g +
  facet_grid(
    day_night ~ is_workday,
    scales = "free"
  )
```



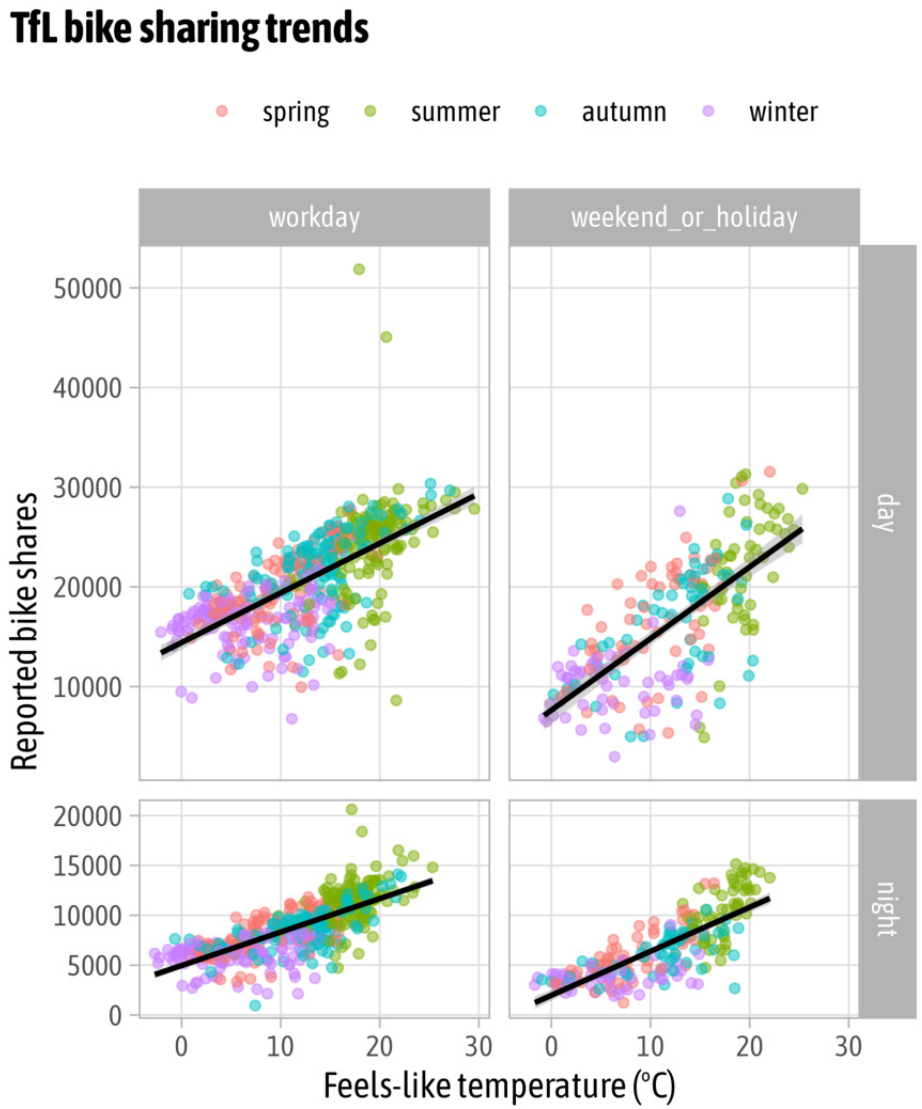
Facet Options: Proportional Spacing

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



Facet Options: Proportional Spacing

```
g +
  facet_grid(
    day_night ~ is_workday,
    scales = "free_y",
    space = "free_y"
  )
```



Coordinate Systems

Cédric Scherer // Data Visualization Society // March 9, 2023

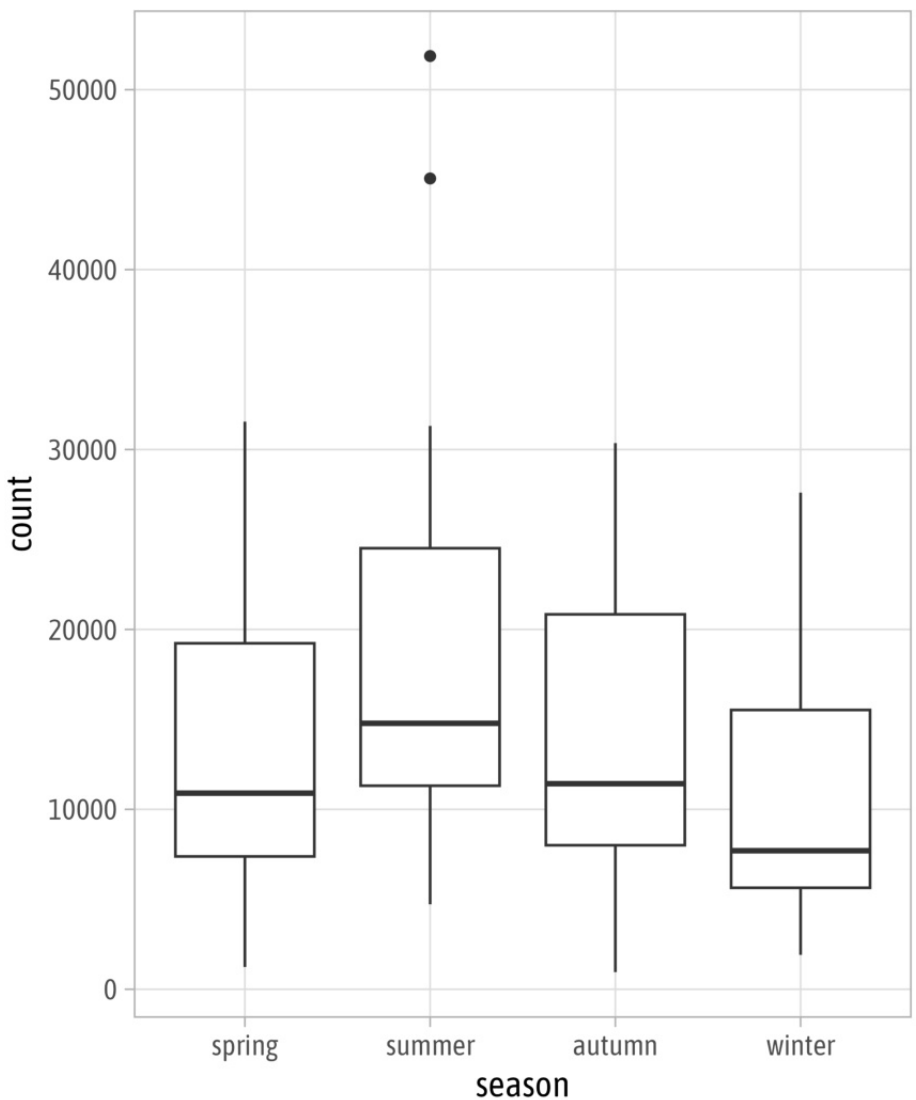
Coordinate Systems

= 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()`

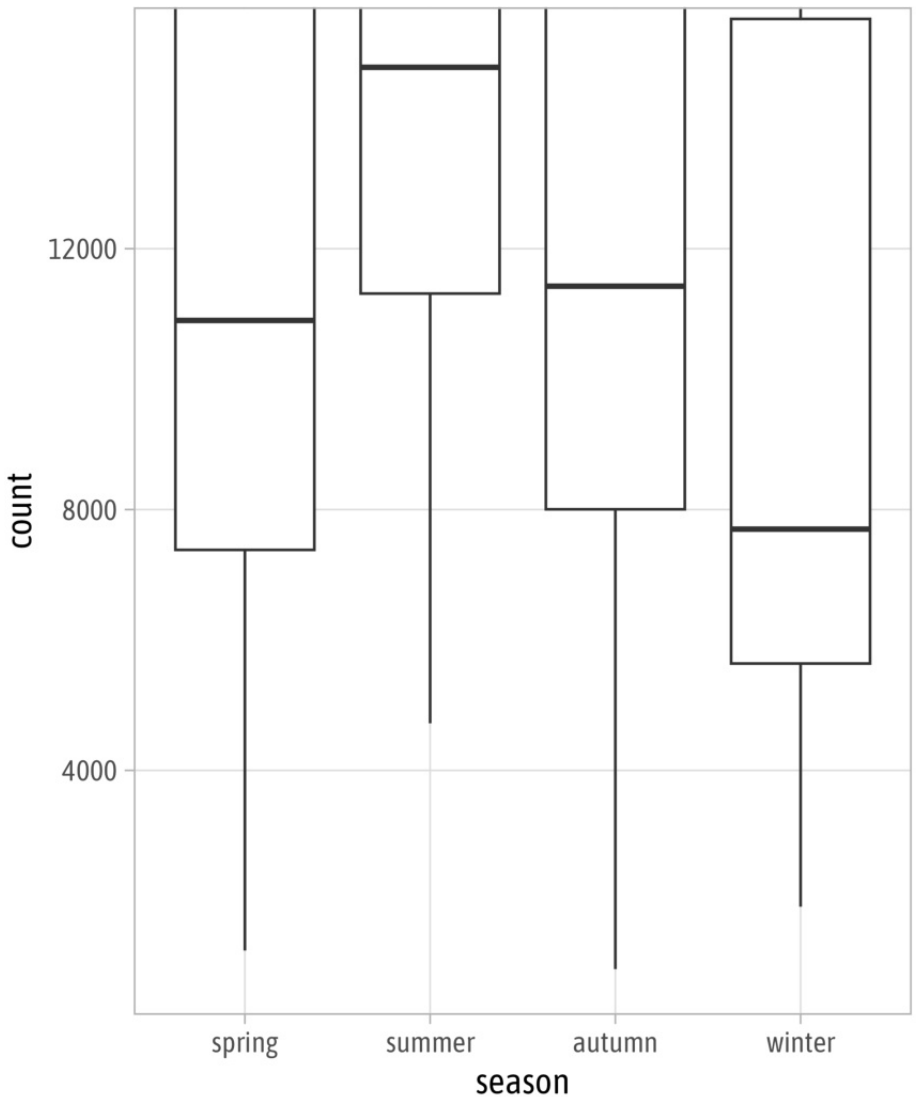
Cartesian Coordinate System

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



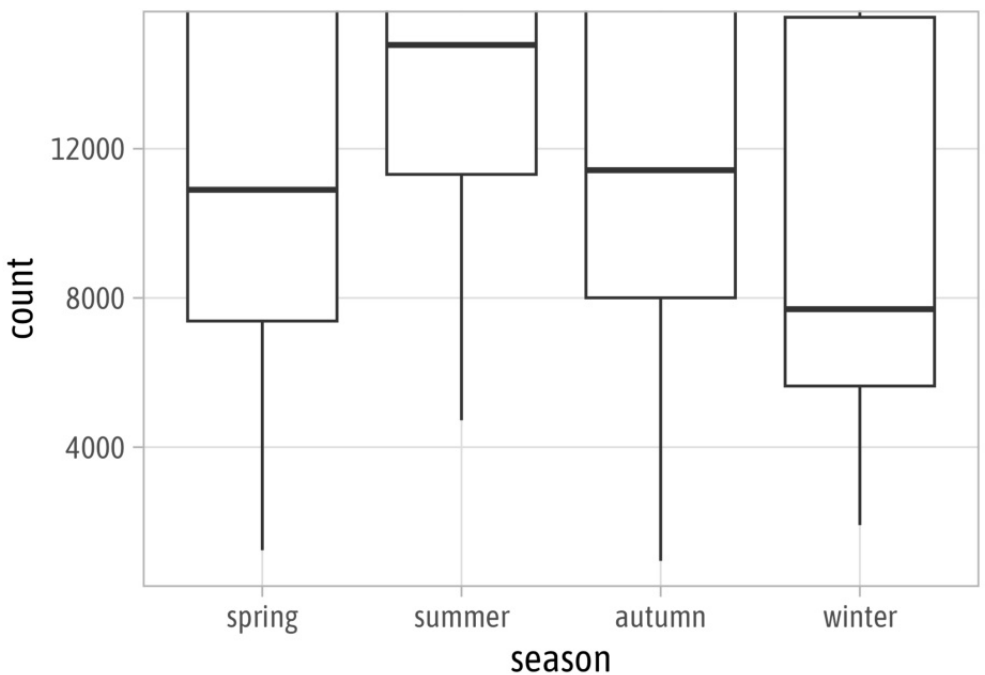
Cartesian Coordinate System

```
ggplot(
    bikes,
    aes(x = season, y = count)
  ) +
  geom_boxplot() +
  coord_cartesian(
    ylim = c(NA, 15000)
  )
```

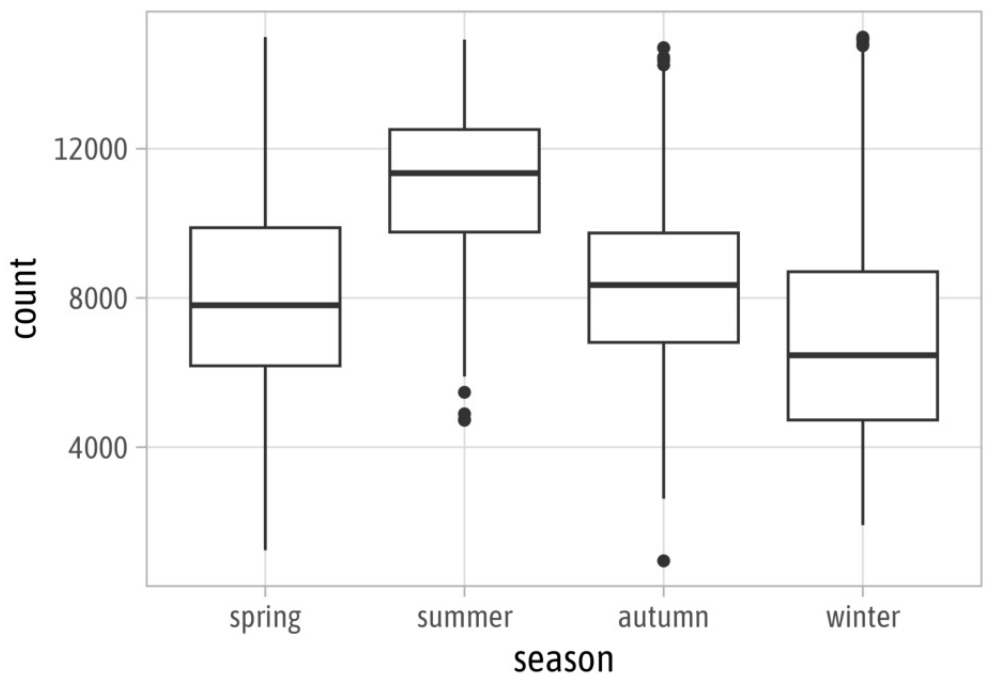


Changing Limits

```
ggplot(
    bikes,
    aes(x = season, y = count)
  ) +
  geom_boxplot() +
  coord_cartesian(
    ylim = c(NA, 15000)
  )
```

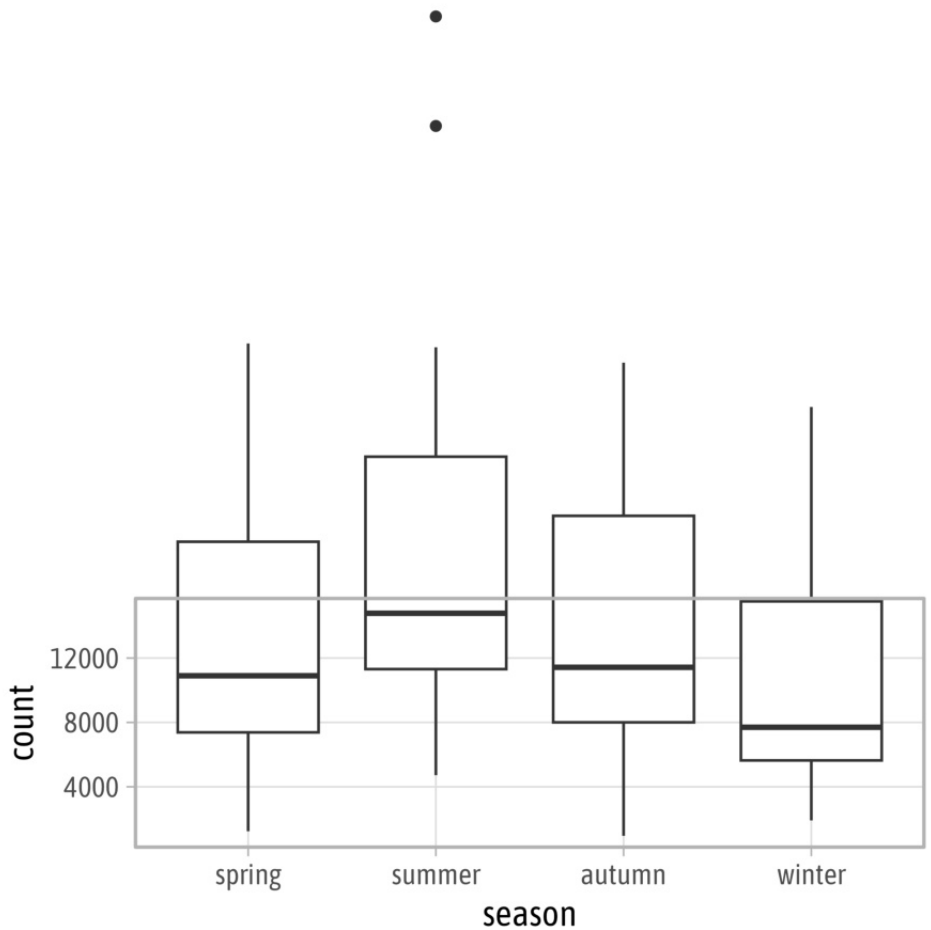



```
ggplot(
    bikes,
    aes(x = season, y = count)
  ) +
  geom_boxplot() +
  scale_y_continuous(
    limits = c(NA, 15000)
  )
```



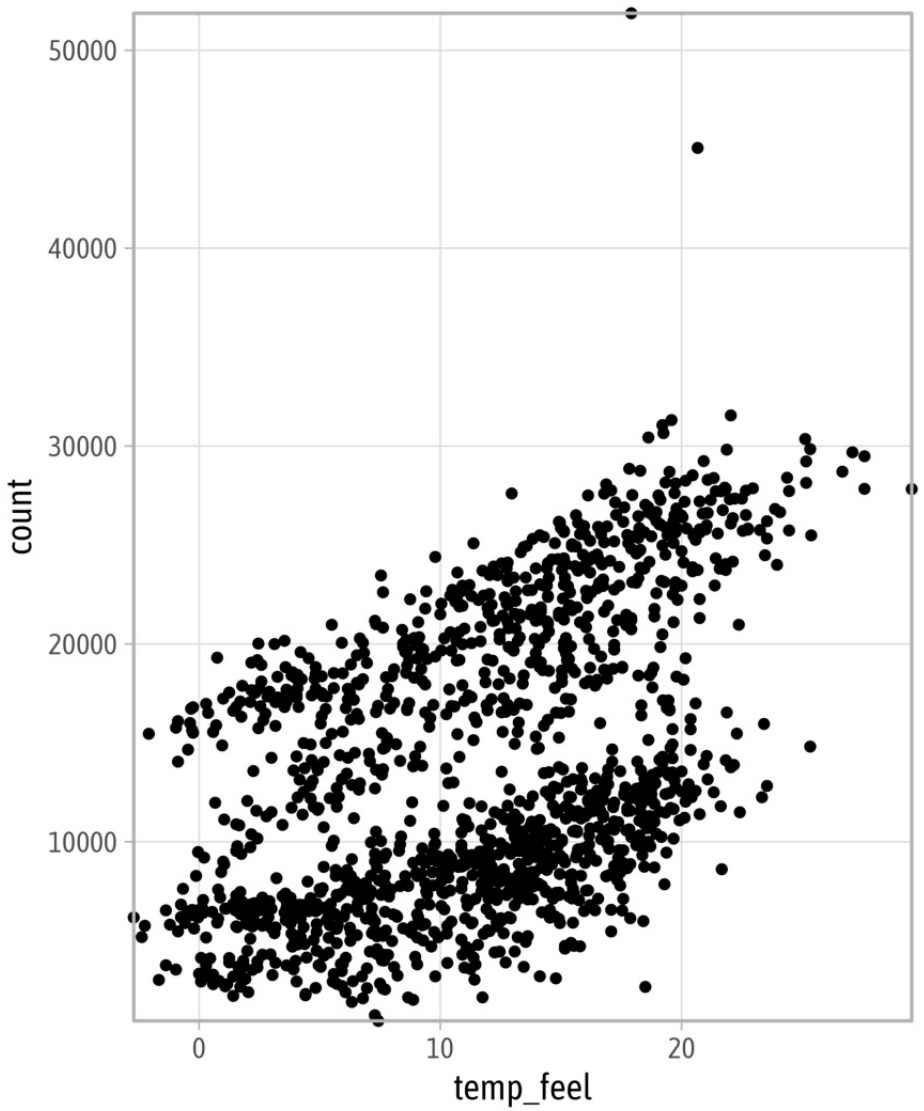
Clipping

```
ggplot(
    bikes,
    aes(x = season, y = count)
  ) +
  geom_boxplot() +
  coord_cartesian(
    ylim = c(NA, 15000),
    clip = "off"
  ) +
  theme(plot.margin = margin(300, 5, 5, 5))
```



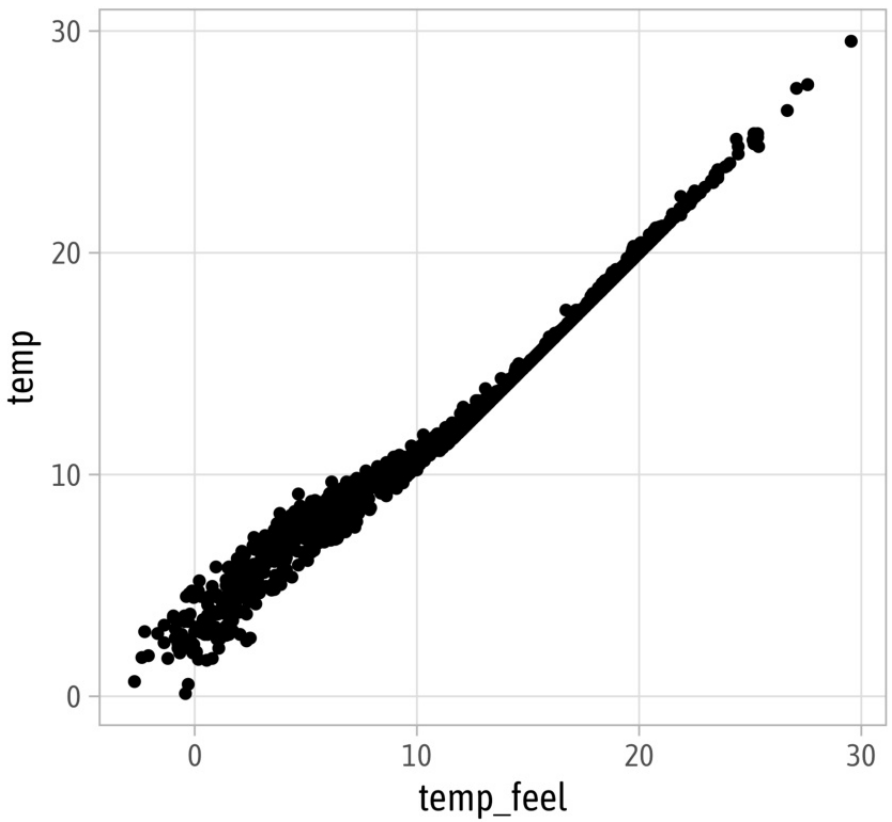
Remove All Padding

```
ggplot(
    bikes,
    aes(x = temp_feel, y = count)
  ) +
  geom_point() +
  coord_cartesian(
    expand = FALSE,
    clip = "off"
  )
```

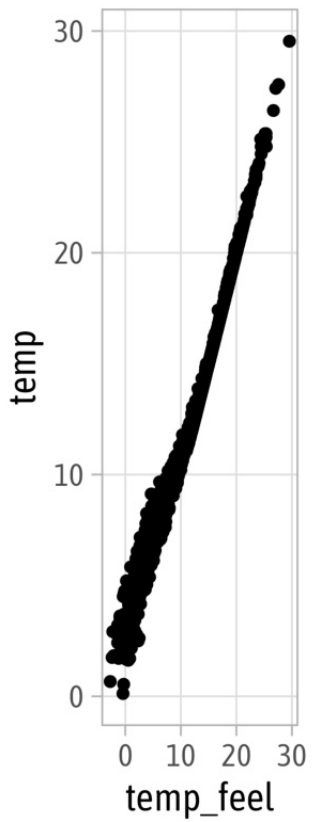


Fixed Coordinate System

```
ggplot(
    bikes,
    aes(x = temp_feel, y = temp)
  ) +
  geom_point() +
  coord_fixed()
```

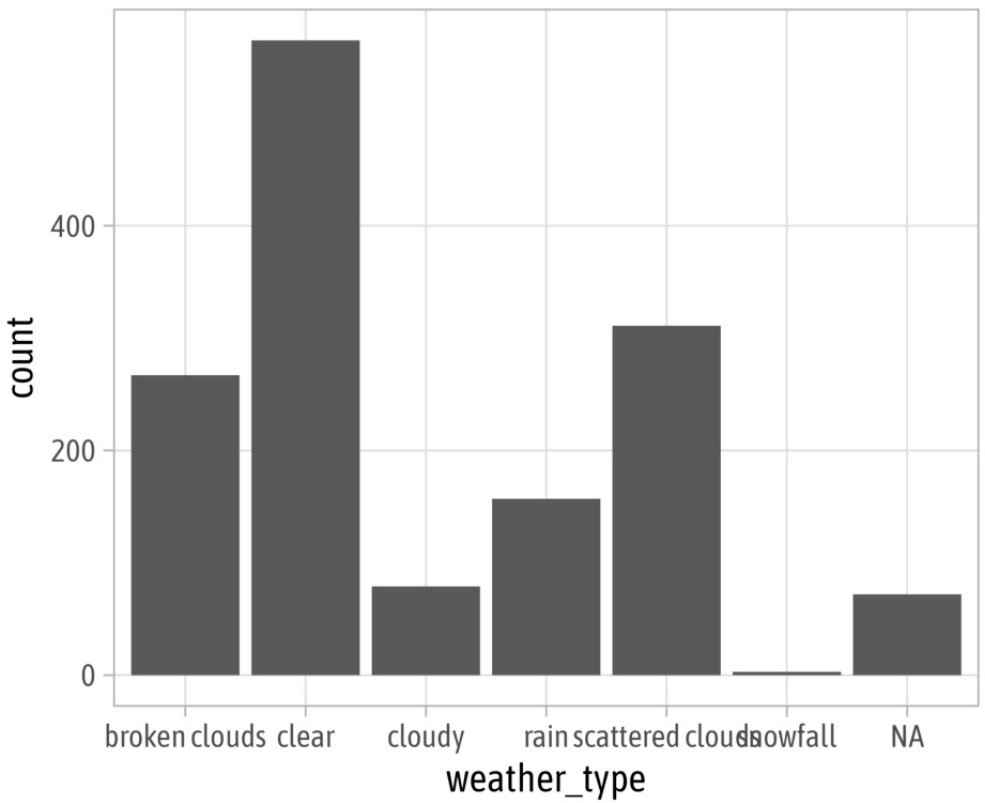



```
ggplot(
    bikes,
    aes(x = temp_feel, y = temp)
  ) +
  geom_point() +
  coord_fixed(ratio = 4)
```

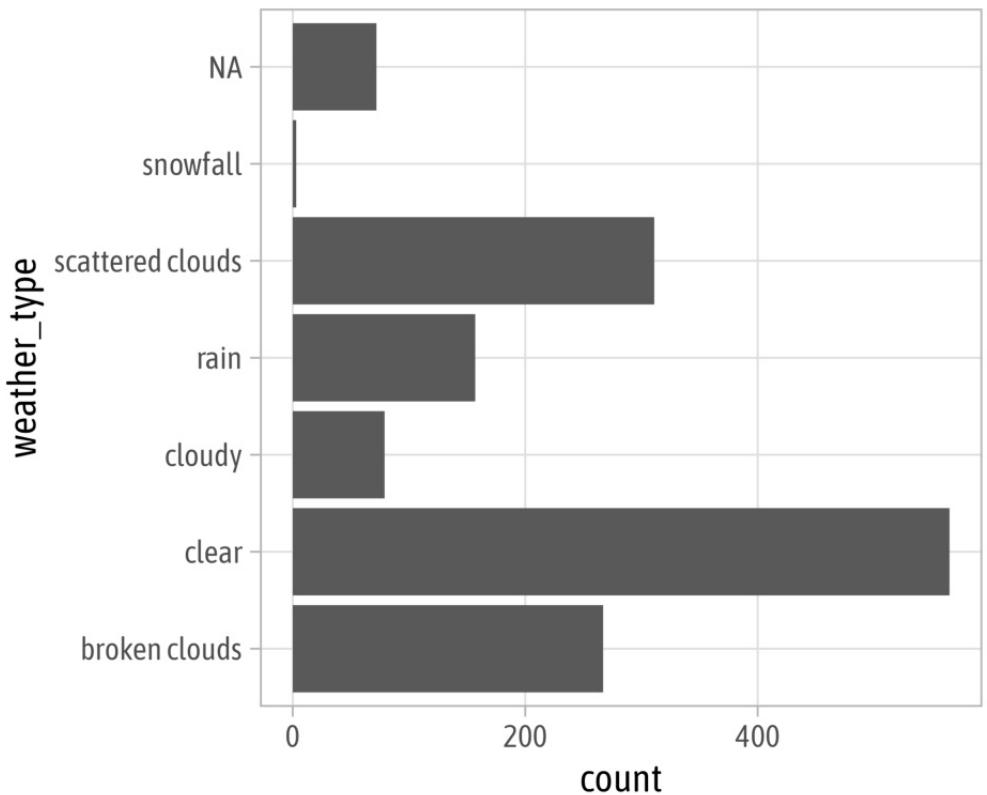


Flipped Coordinate System

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

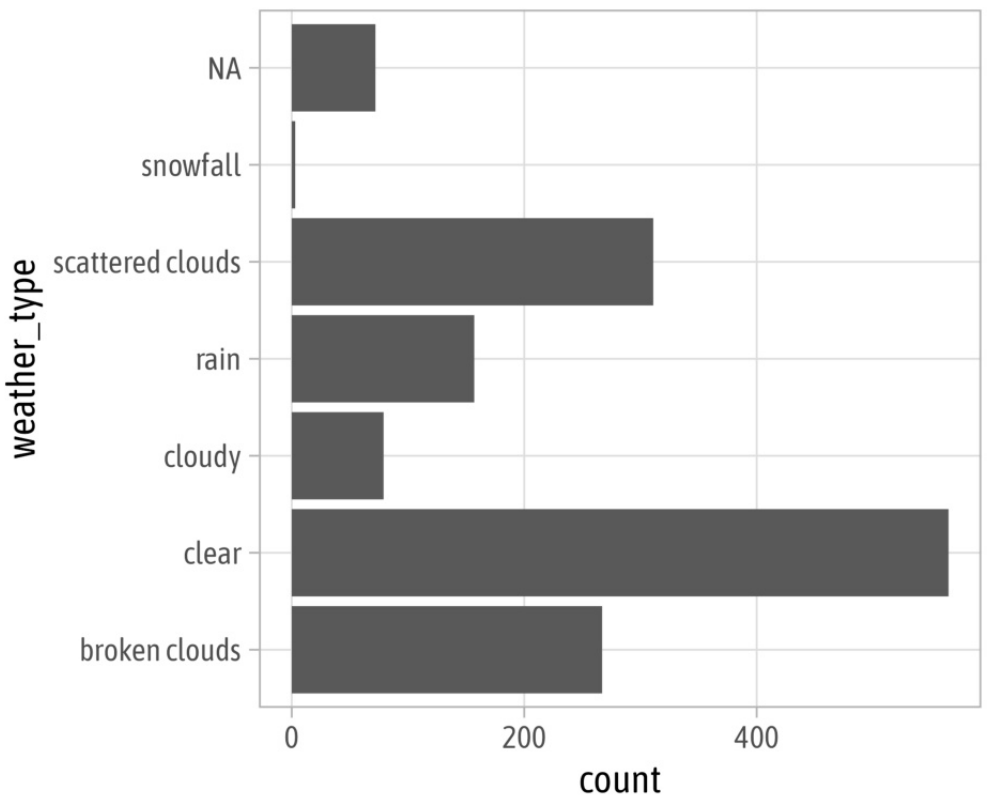



```
1 ggplot(
2     bikes,
3     aes(x = weather_type)
4   ) +
5   geom_bar() +
6   coord_flip()
```

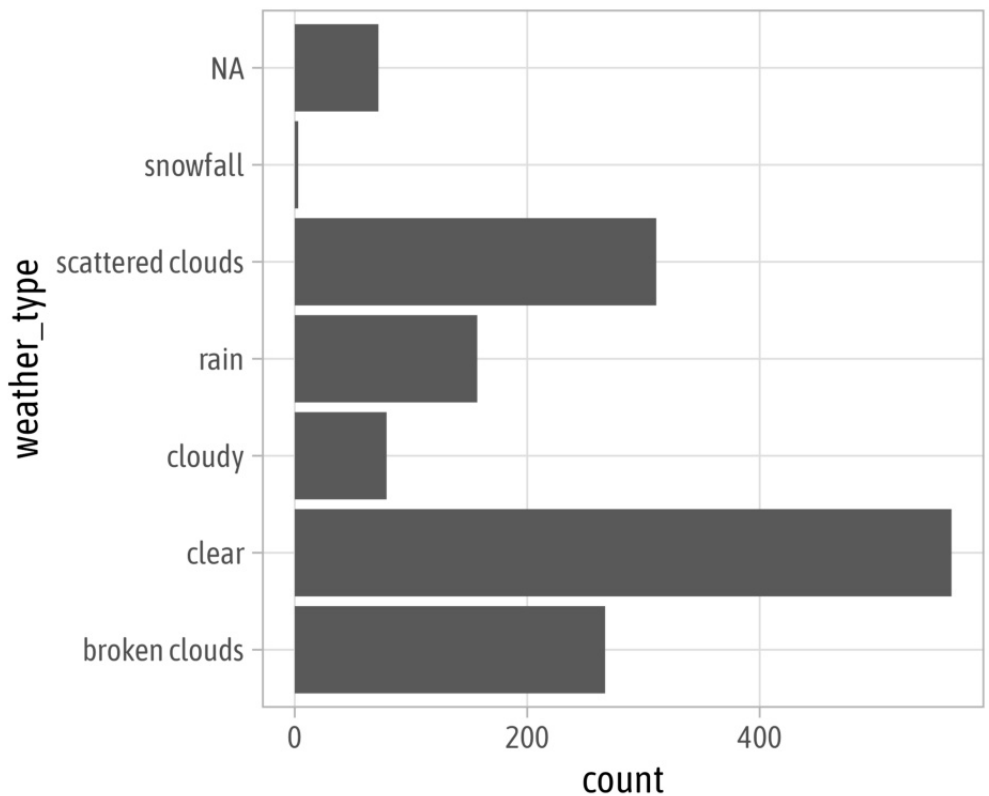


Flipped Coordinate System

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

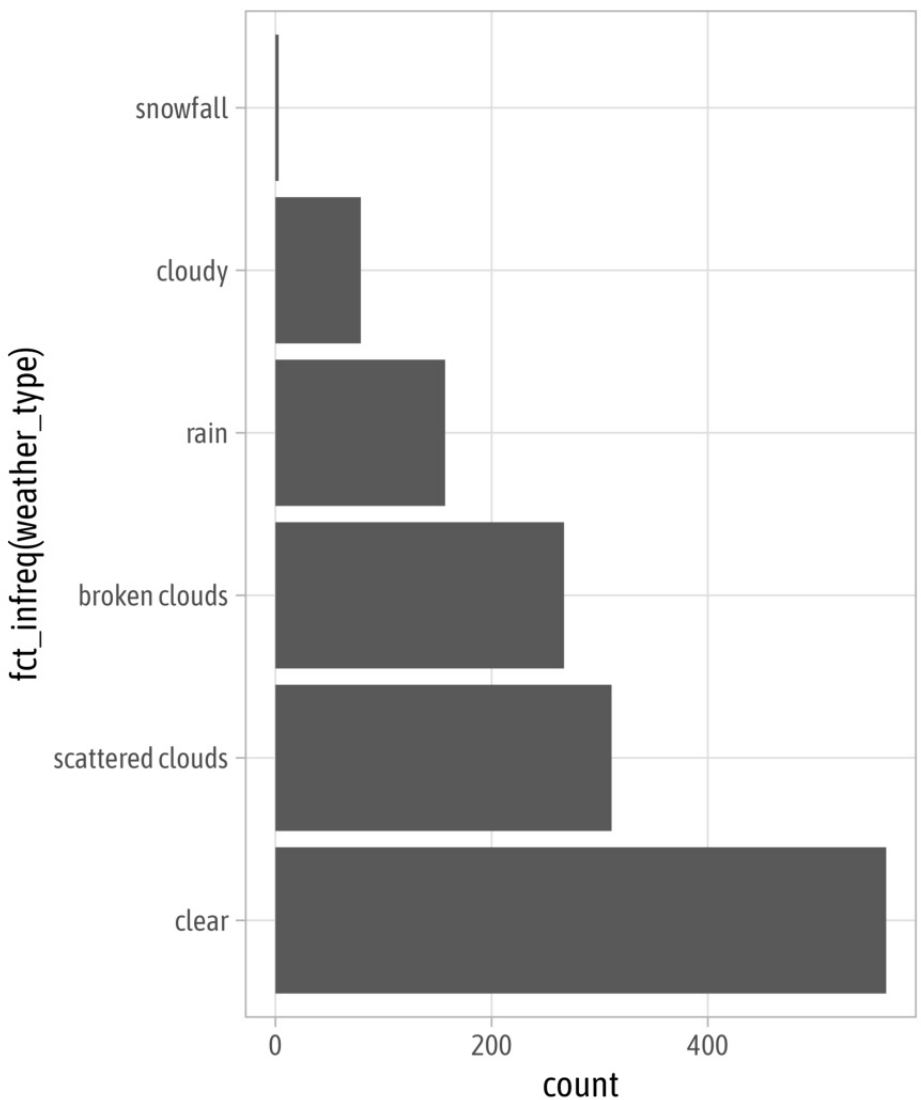



```
1 ggplot(
2     bikes,
3     aes(x = weather_type)
4   ) +
5   geom_bar() +
6   coord_flip()
```



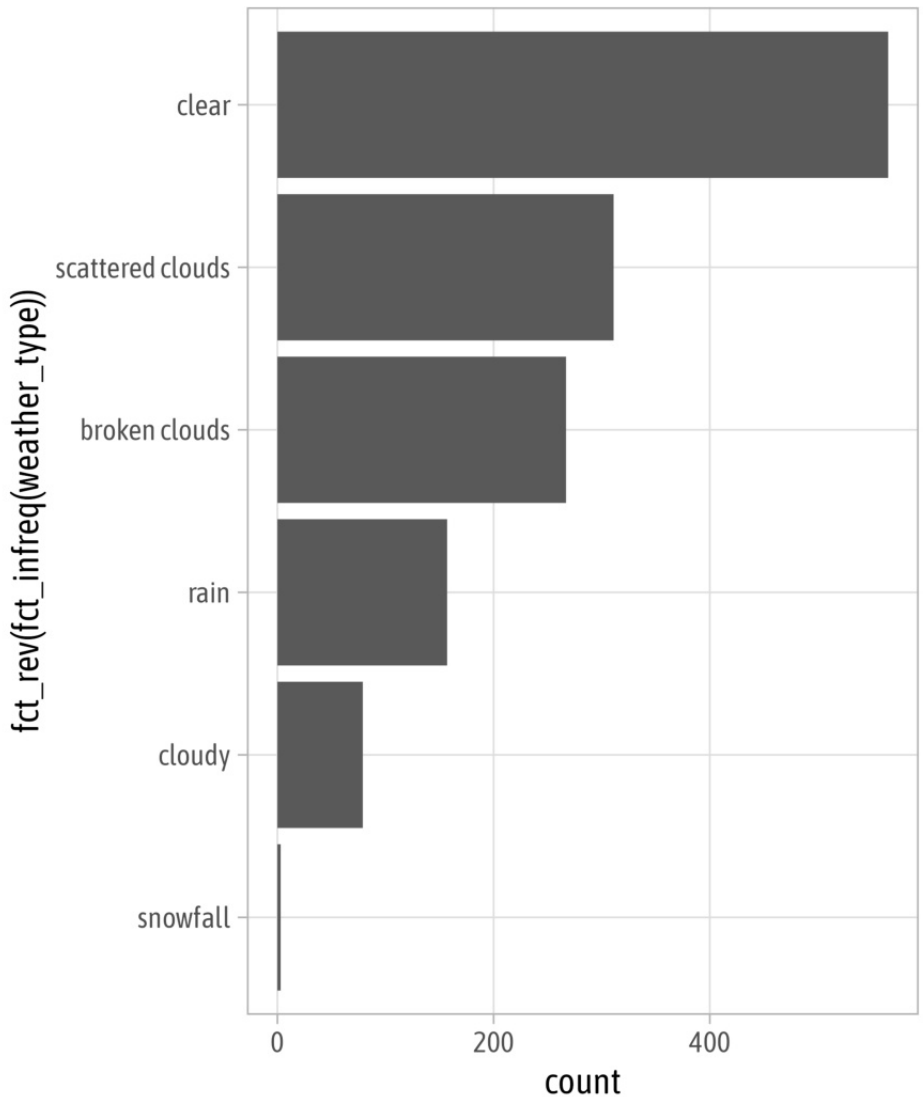
Reminder: Sort Your Bars!

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



Reminder: Sort Your Bars!

```
1  library(forcats)
2
3  ggplot(
4      filter(bikes, !is.na(weather_type)),
5      aes(y = fct_rev(
6        fct_infreq(weather_type)
7      ))
8    ) +
9    geom_bar()
```



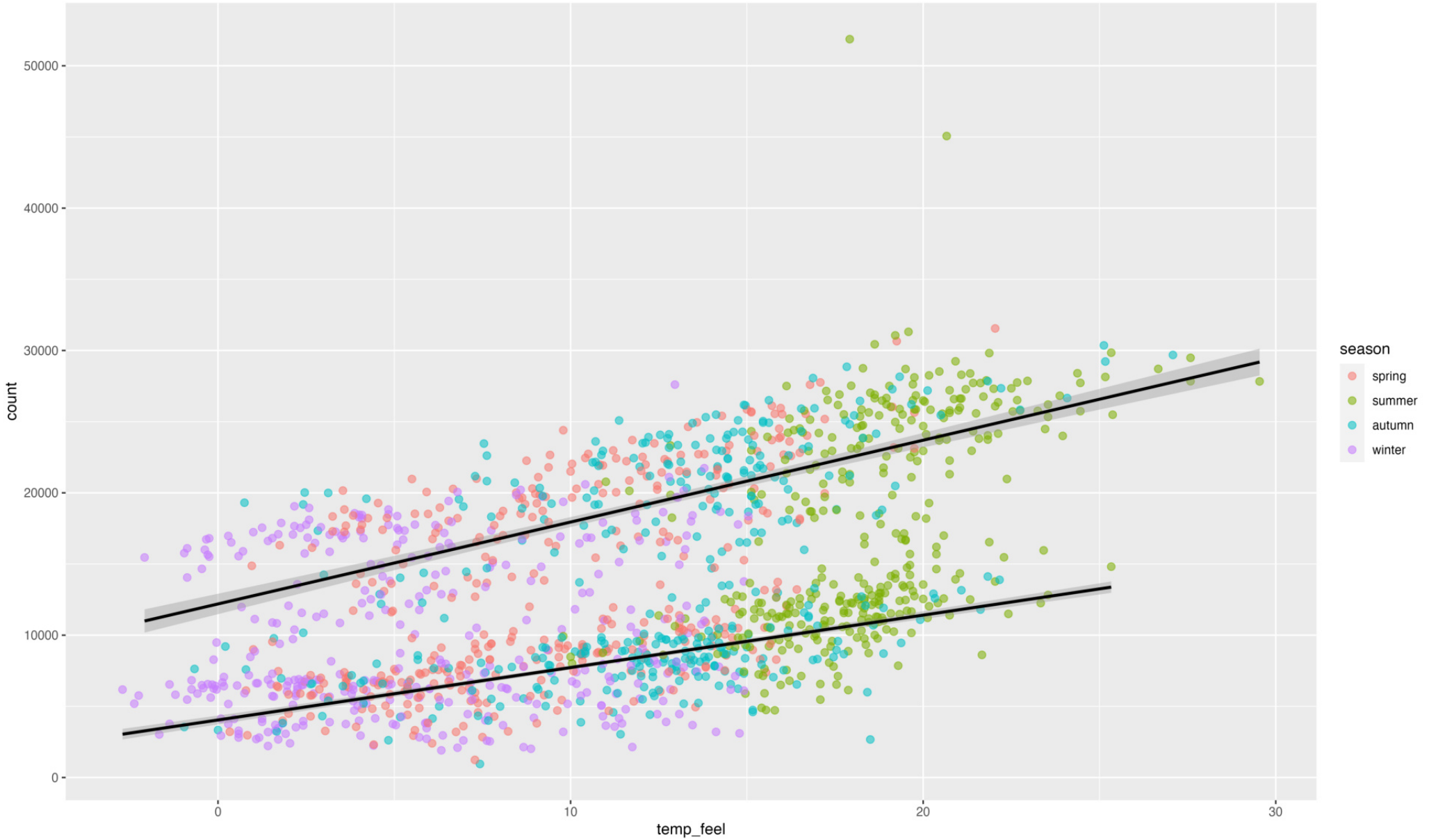
Wrap-Up

```r
g1 <-
  ## create scatter plot, encoded based on season
  ggplot(bikes, aes(temp_feel, count)) +
  geom_point(
    aes(color = season),
    size = 2.2, alpha = .55
  )

g1
```

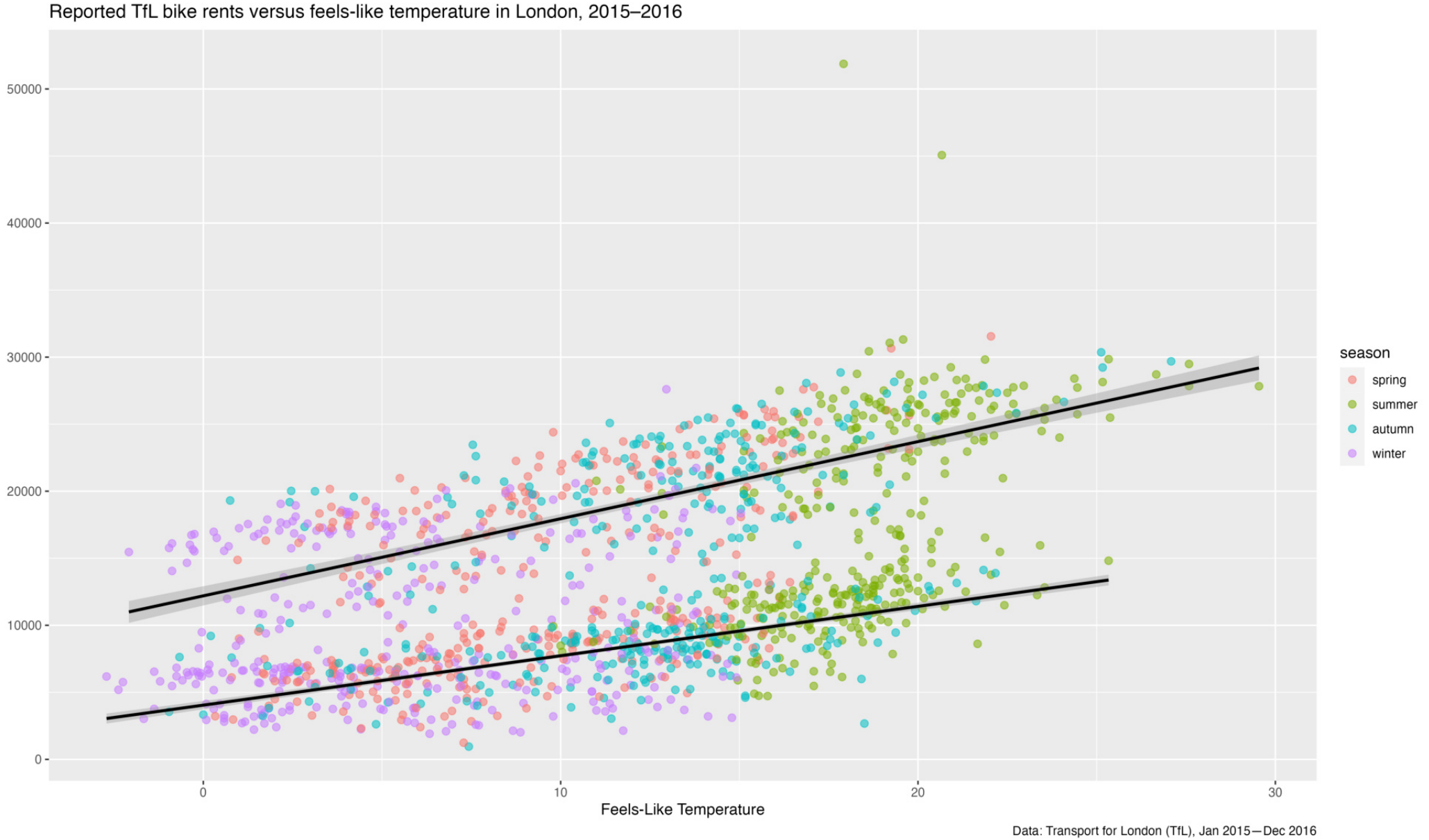


50000
40000
30000
20000
10000
0
count
0
10
20
30
temp_feel
season
spring
summer
autumn
winter

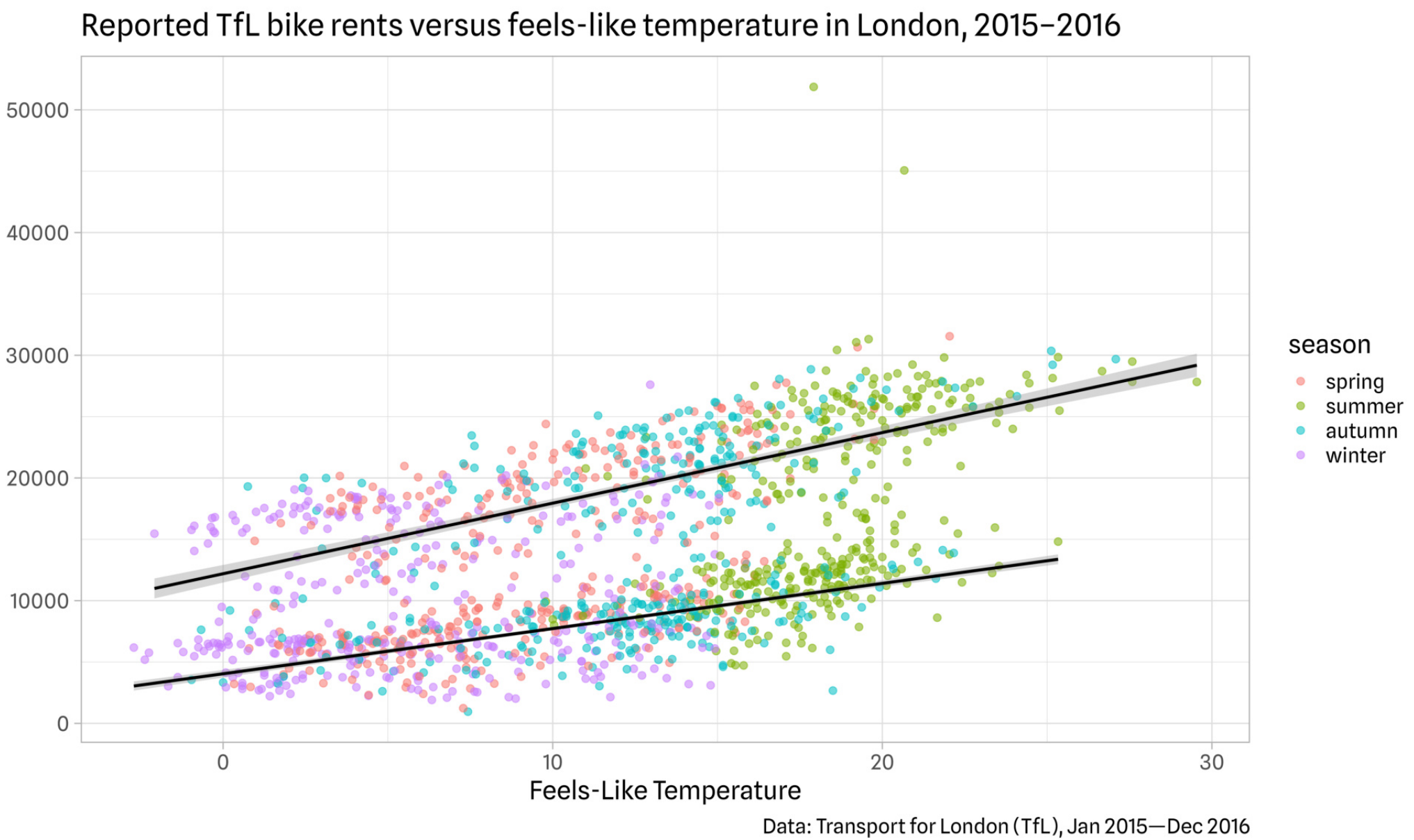
```r
g2 <- g1 +
  ## add a linear fitting for each time of the day
  geom_smooth(
    aes(group = day_night),
    method = "lm", color = "black"
  )

g2
```



50000
40000
30000
20000
10000
0
count
0
10
20
30
temp_feel
season
spring
summer
autumn
winter

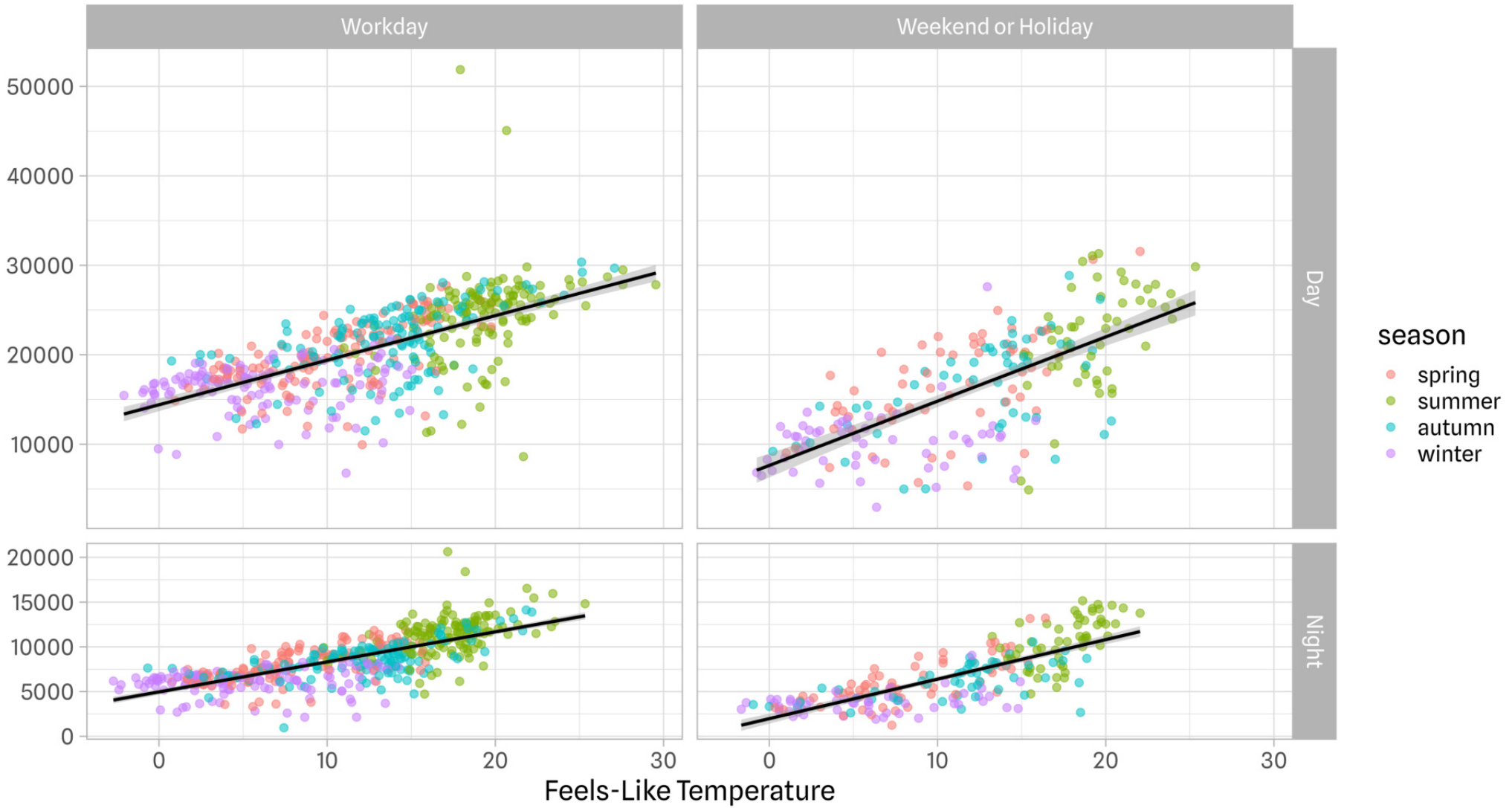
```r
g3 <- g2 +
  ## add titles + labels
  labs(
    x = "Feels-Like Temperature", y = NULL,
    caption = "Data: Transport for London (TfL), Jan 2015–Dec 2016",
    title = "Reported TfL bike rents versus feels-like temperature in London, 2015-2016"
  )

g3
```



Reported TfL bike rents versus feels-like temperature in London, 2015–2016
50000
40000
30000
20000
10000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016

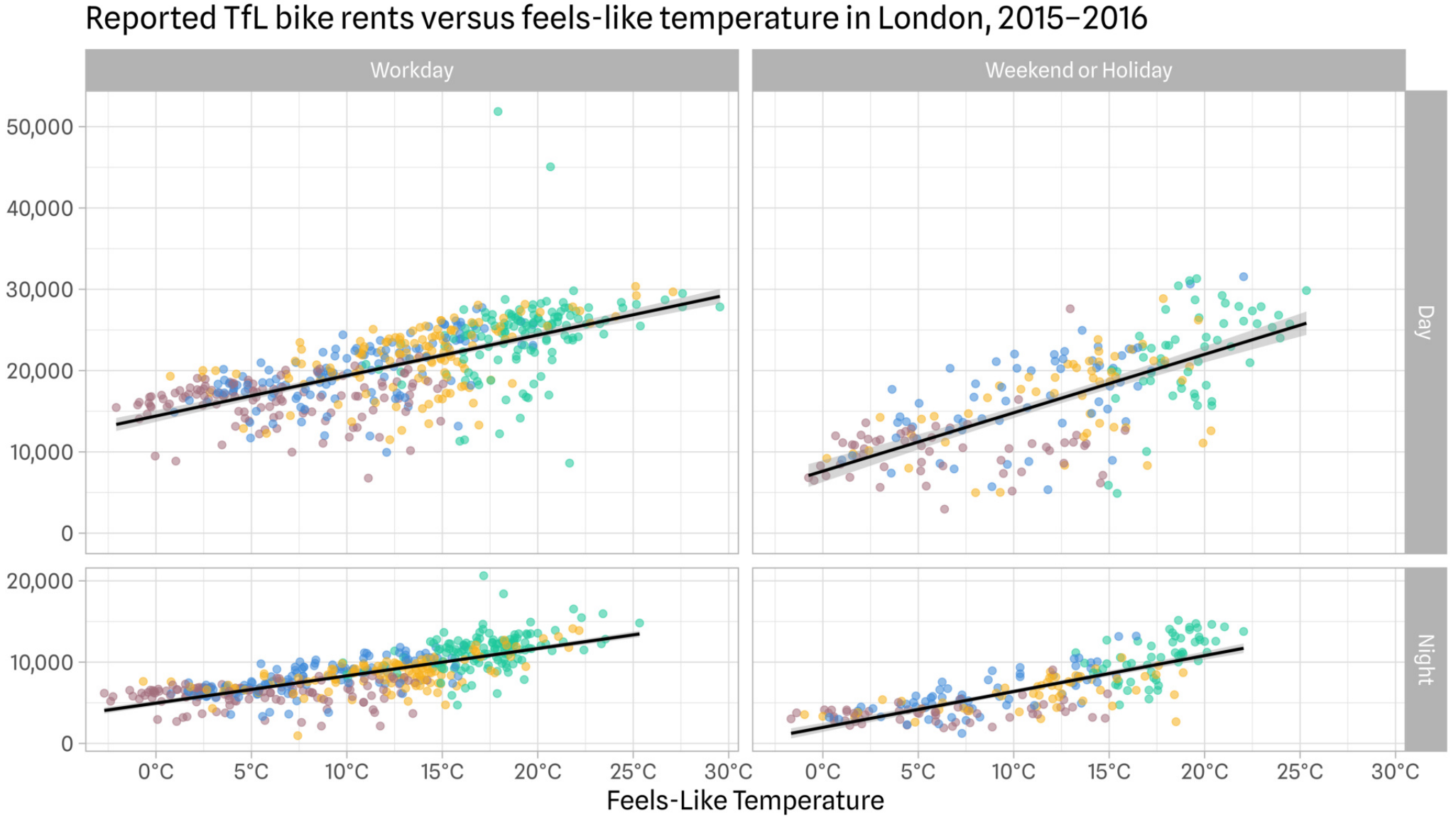
```r
g4 <- g3 +
  theme_light(base_size = 18, base_family = "Spline Sans")

g4
```



Reported TfL bike rents versus feels-like temperature in London, 2015–2016
50000
40000
30000
20000
10000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016

```r
codes <- c(
  workday = "Workday",
  weekend_or_holiday = "Weekend or Holiday"
)

g5 <- g4 +
  facet_grid(
    day_night ~ is_workday,
    scales = "free_y", space = "free_y",
    labeller = labeller(
      day_night = stringr::str_to_title,
      is_workday = codes
    )
  )

g5
```

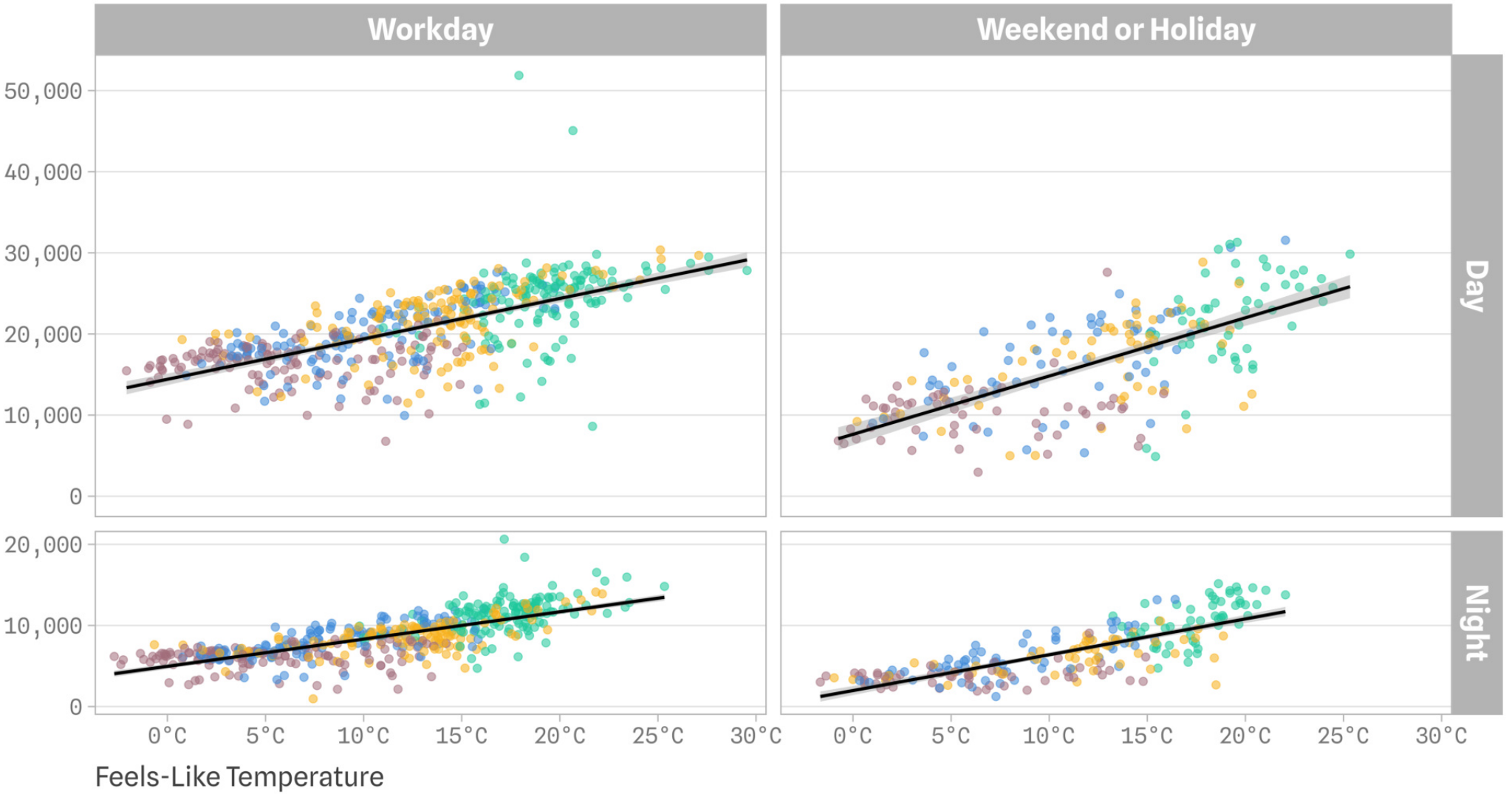


Reported TfL bike rents versus feels-like temperature in London, 2015–2016
Workday
Weekend or Holiday
Day
Night
50000
40000
30000
20000
10000
20000
15000
10000
5000
0
0
10
20
30
Feels-Like Temperature
season
spring
summer
autumn
winter
Data: Transport for London (TfL), Jan 2015—Dec 2016

```r
g6 <- g5  +
  ## adjust labels x-axis
  scale_x_continuous(
    expand = c(mult = 0, add = 1),
    breaks = 0:6*5, labels = function(x) paste0(x, "°C")
  ) +
  ## adjust labels y-axis
  scale_y_continuous(
    expand = c(mult = .05, add = 0),limits = c(0, NA),
    breaks = 0:5*10000, labels = scales::label_comma()
  ) +
  ## modify colors + legend
  scale_color_manual(
    values = c("#3c89d9", "#1ec99b", "#F7B01B", "#a26e7c"), name = NULL,
    breaks = stringr::str_to_title,
    guide = guide_legend(override.aes = list(size = 5))
  )

g6
```



Reported TfL bike rents versus feels-like temperature in London, 2015–2016
Workday
Weekend or Holiday
Day
Night
50,000
40,000
30,000
20,000
10,000
0
0°C
5°C
10°C
15°C
20°C
25°C
30°C
Feels-Like Temperature
Data: Transport for London (TfL), Jan 2015—Dec 2016

```r
g6 +
  ## theme modifications
  theme(
    plot.title.position = "plot",
    plot.caption.position = "plot",
    plot.title = element_text(face = "bold", size = rel(1.5)),
    axis.text = element_text(family = "Spline Sans Mono", color = "grey45"),
    axis.title.x = element_text(hjust = 0, margin = margin(t = 12), color = "grey25"),
    strip.text = element_text(face = "bold", size = rel(1.1)),
    panel.grid.major.x = element_blank(),
    panel.grid.minor = element_blank(),
    legend.position = "top"
  )
```


Reported TfL bike rents versus feels-like temperature in London, 2015–2016



Data: Transport for London (TfL), Jan 2015—Dec 2016

That's it Folks...

— Thank you! —

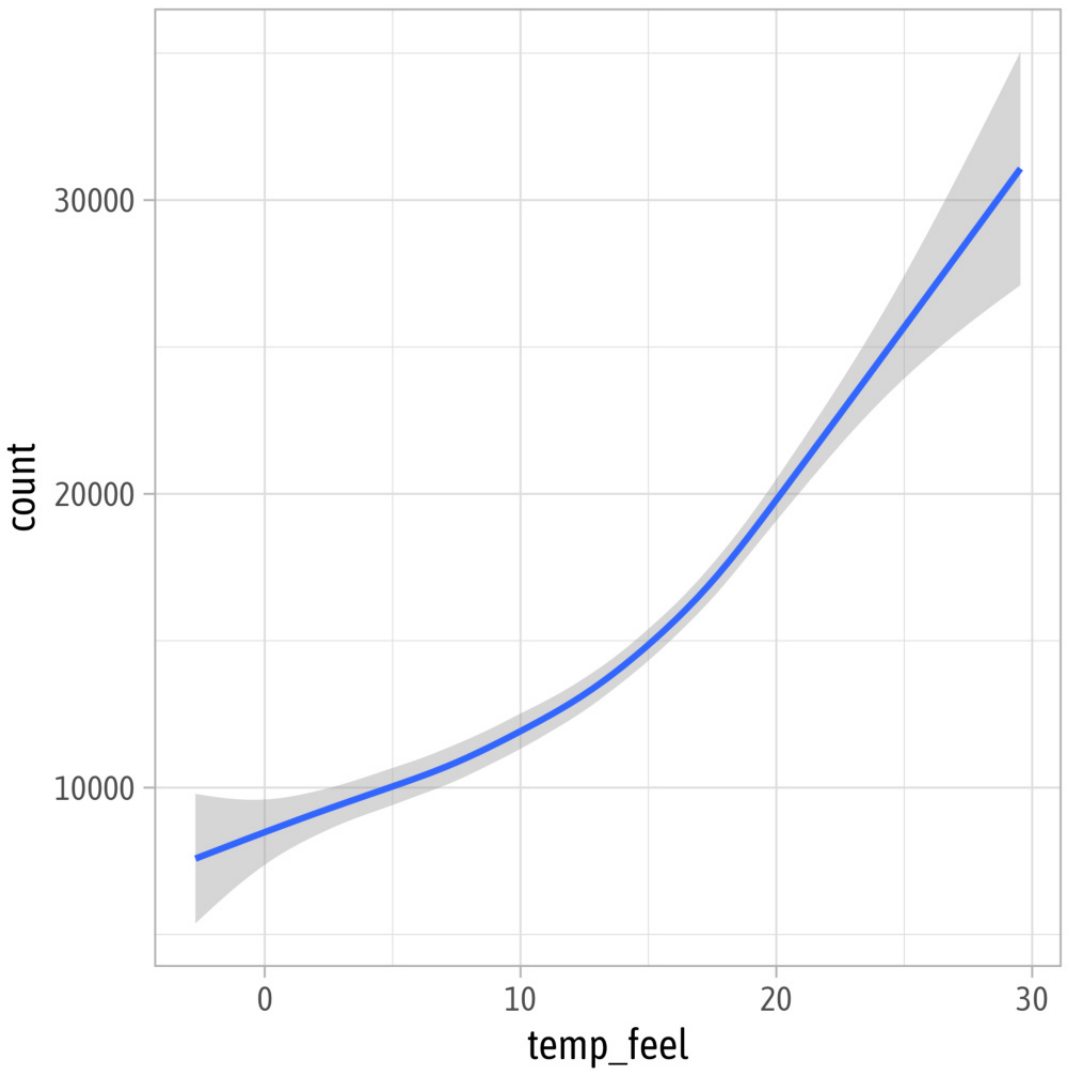
Appendix

Statistical Layers

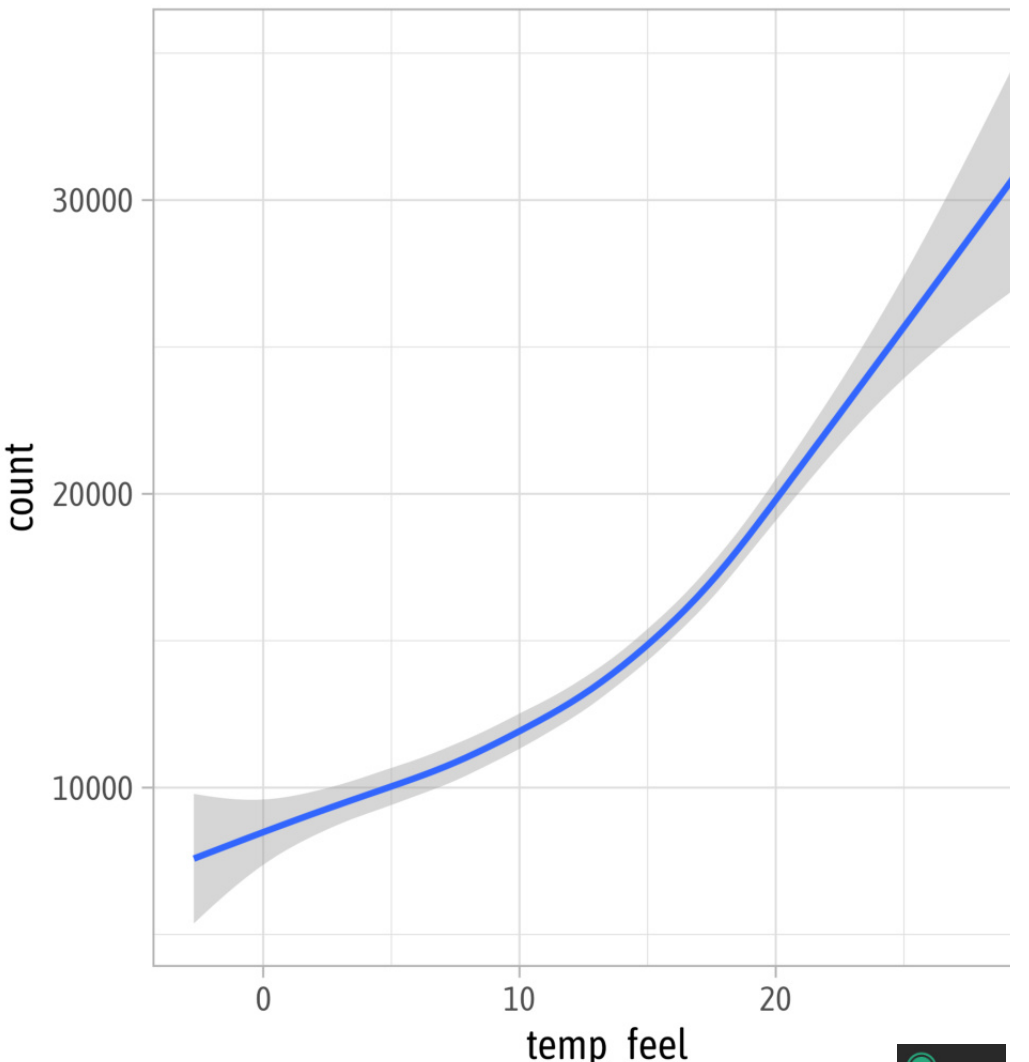
Cédric Scherer // Data Visualization Society // March 9, 2023

geom_*() and stat_*()

```
1 ggplot(bikes, aes(x = temp_feel, y = count)) +
2   geom_smooth(stat = "smooth")
```

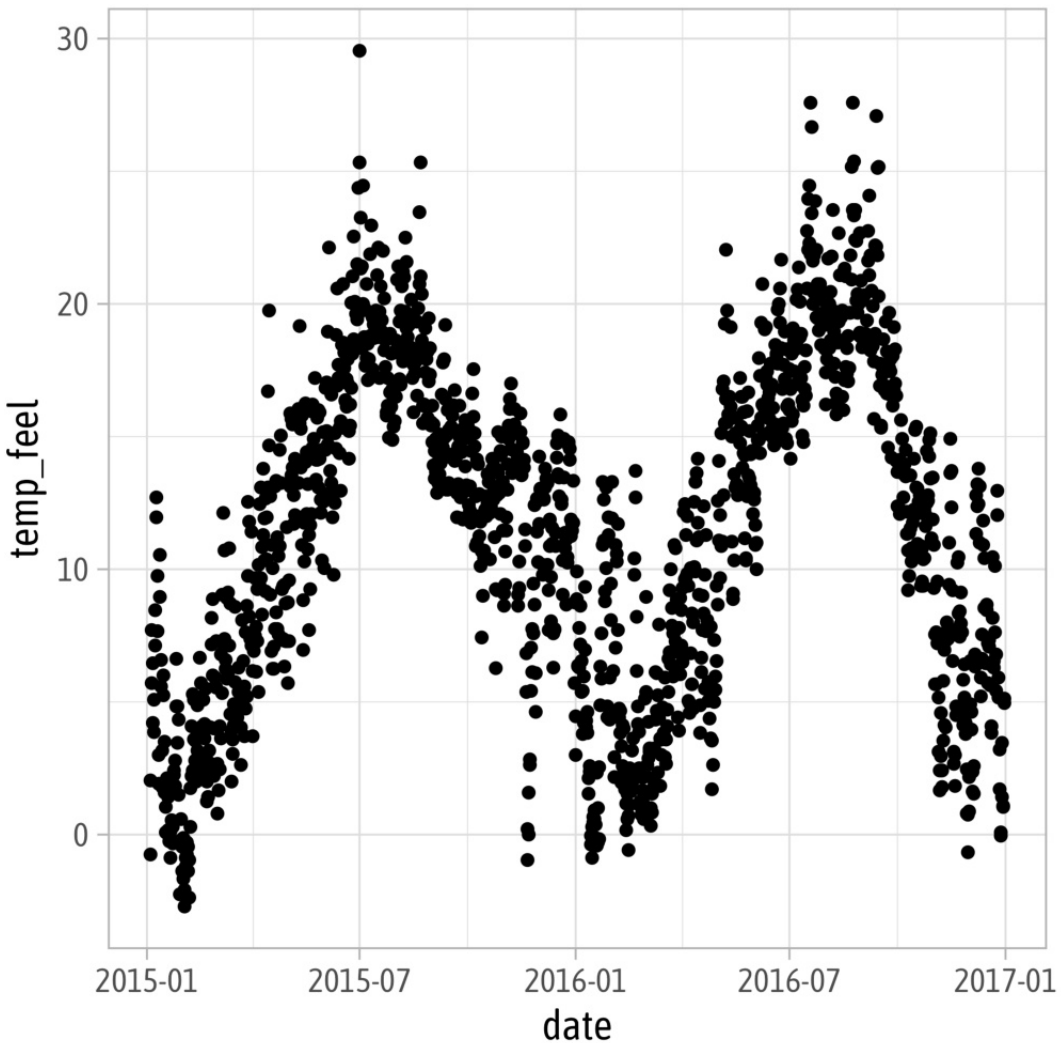



```
1 ggplot(bikes, aes(x = temp_feel, y = count)) +
2   stat_smooth(geom = "smooth")
```

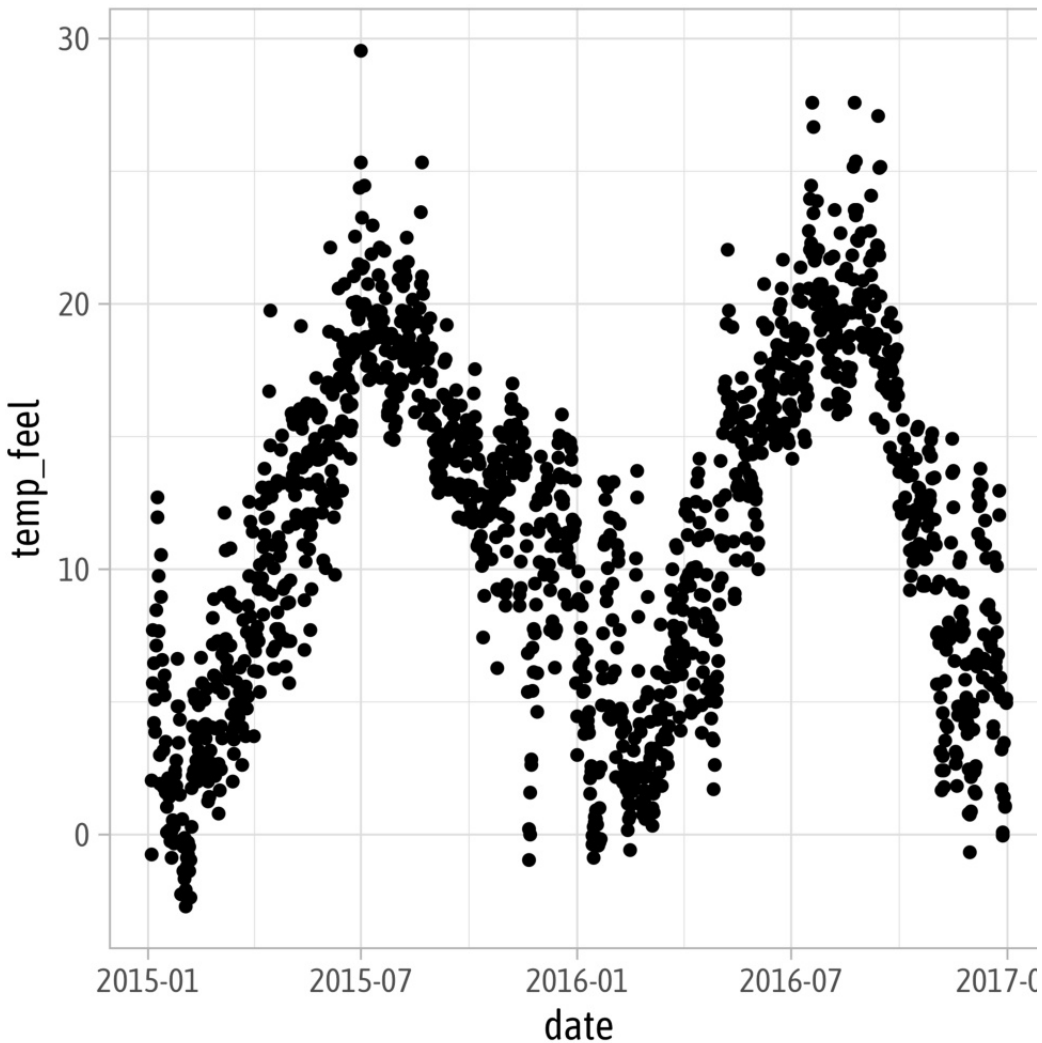


geom_*() and stat_*()

```
1 ggplot(bikes, aes(x = date, y = temp_feel)) +
2   geom_point(stat = "identity")
```

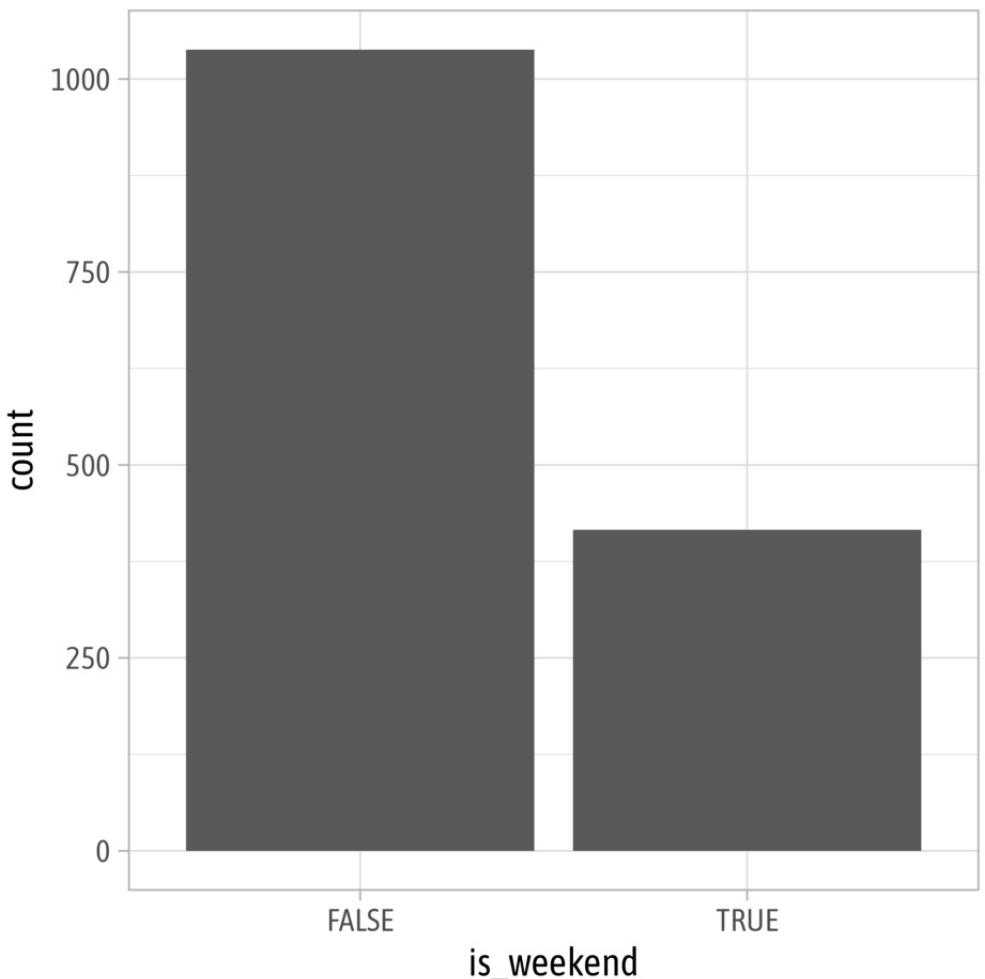



```
1 ggplot(bikes, aes(x = date, y = temp_feel)) +
2   stat_identity(geom = "point")
```

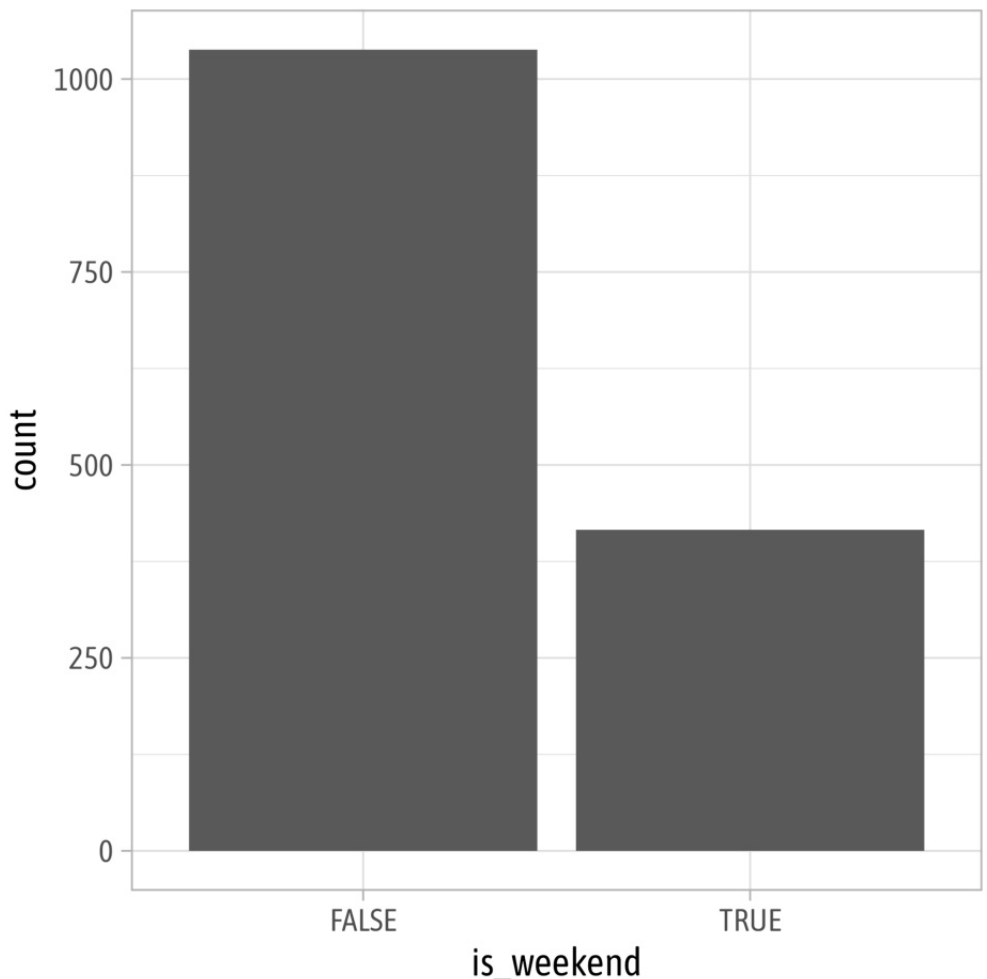


geom_*() and stat_*()

```
1 ggplot(bikes, aes(x = is_weekend)) +
2   geom_bar(stat = "count")
```

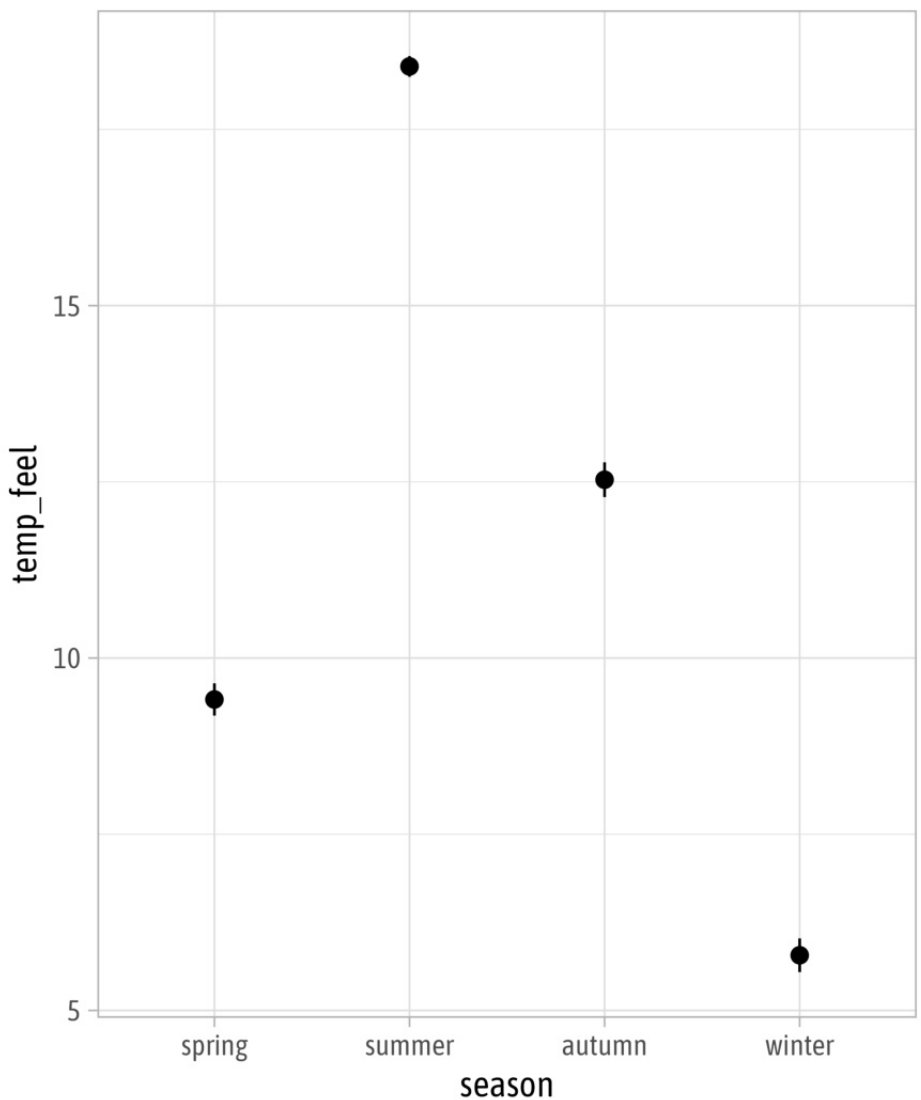



```
1 ggplot(bikes, aes(x = is_weekend)) +
2   stat_count(geom = "bar")
```



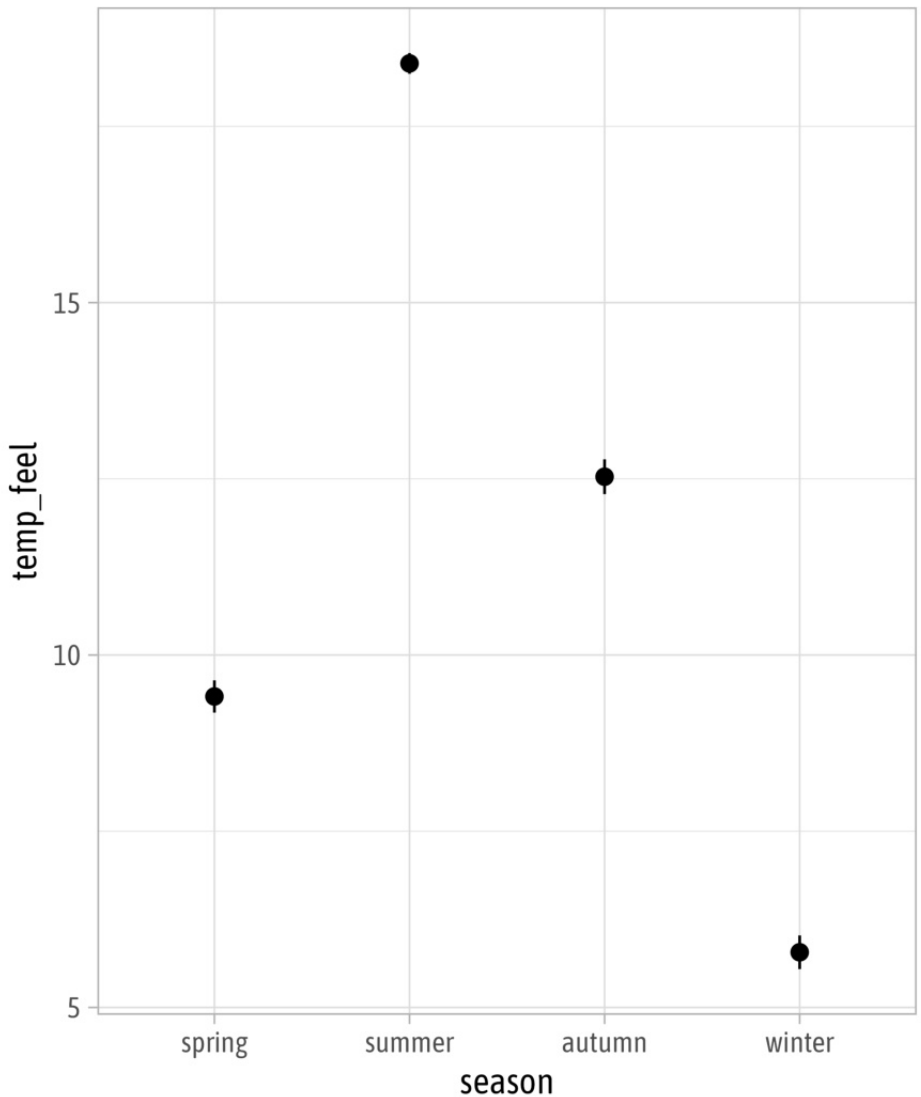
Statistical Summaries

```
1 ggplot(
2     bikes,
3     aes(x = season, y = temp_feel)
4   ) +
5   stat_summary()
```



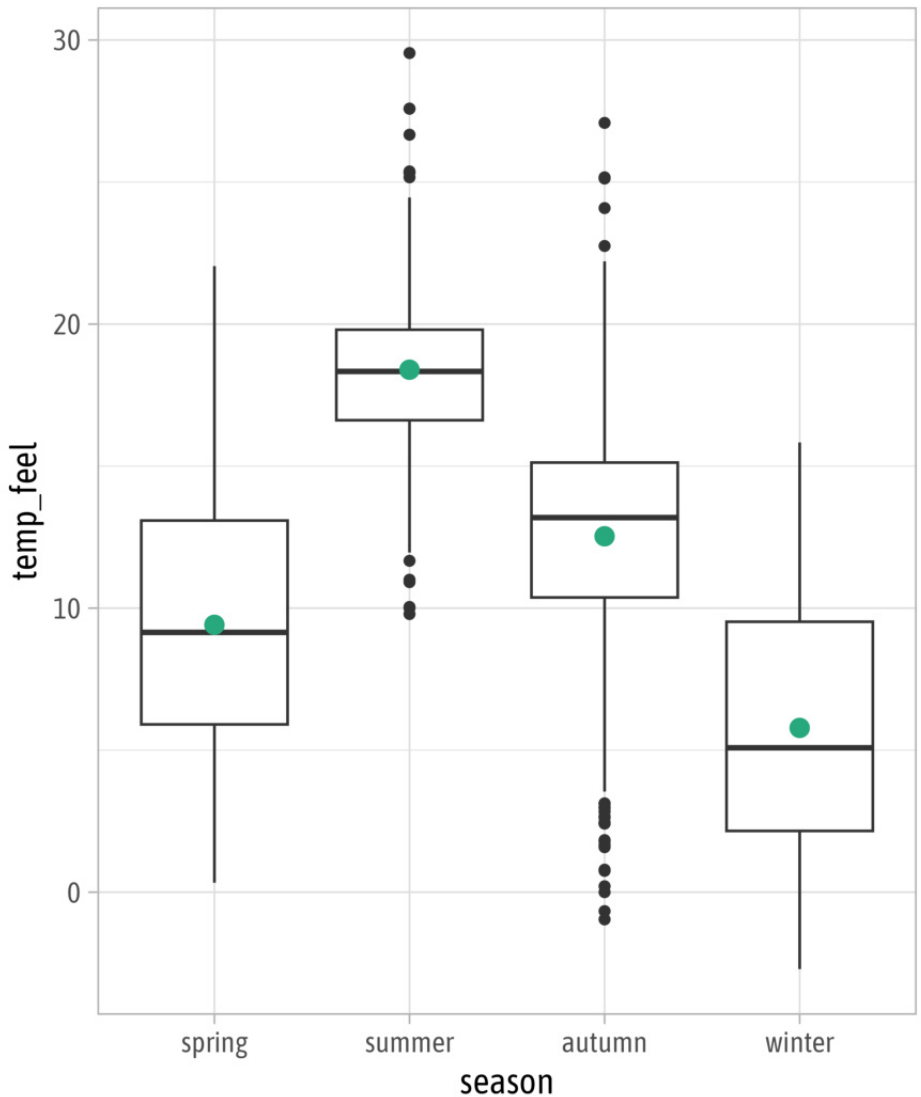
Statistical Summaries

```
ggplot(
    bikes,
    aes(x = season, y = temp_feel)
  ) +
  stat_summary(
    fun.data = mean_se, ## the default
    geom = "pointrange"  ## the default
  )
```



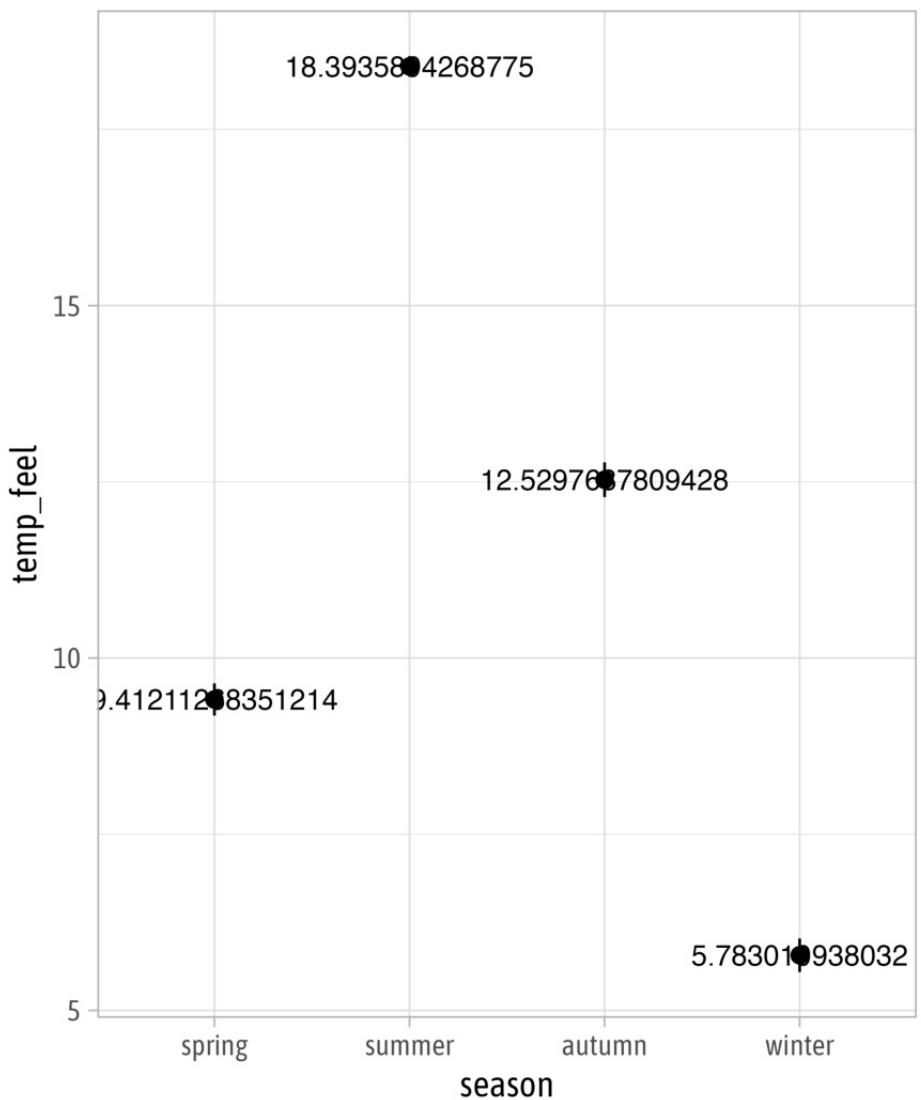
Statistical Summaries

```
ggplot(
    bikes,
    aes(x = season, y = temp_feel)
  ) +
  geom_boxplot() +
  stat_summary(
    fun = mean,
    geom = "point",
    color = "#28a87d",
    size = 3
  )
```



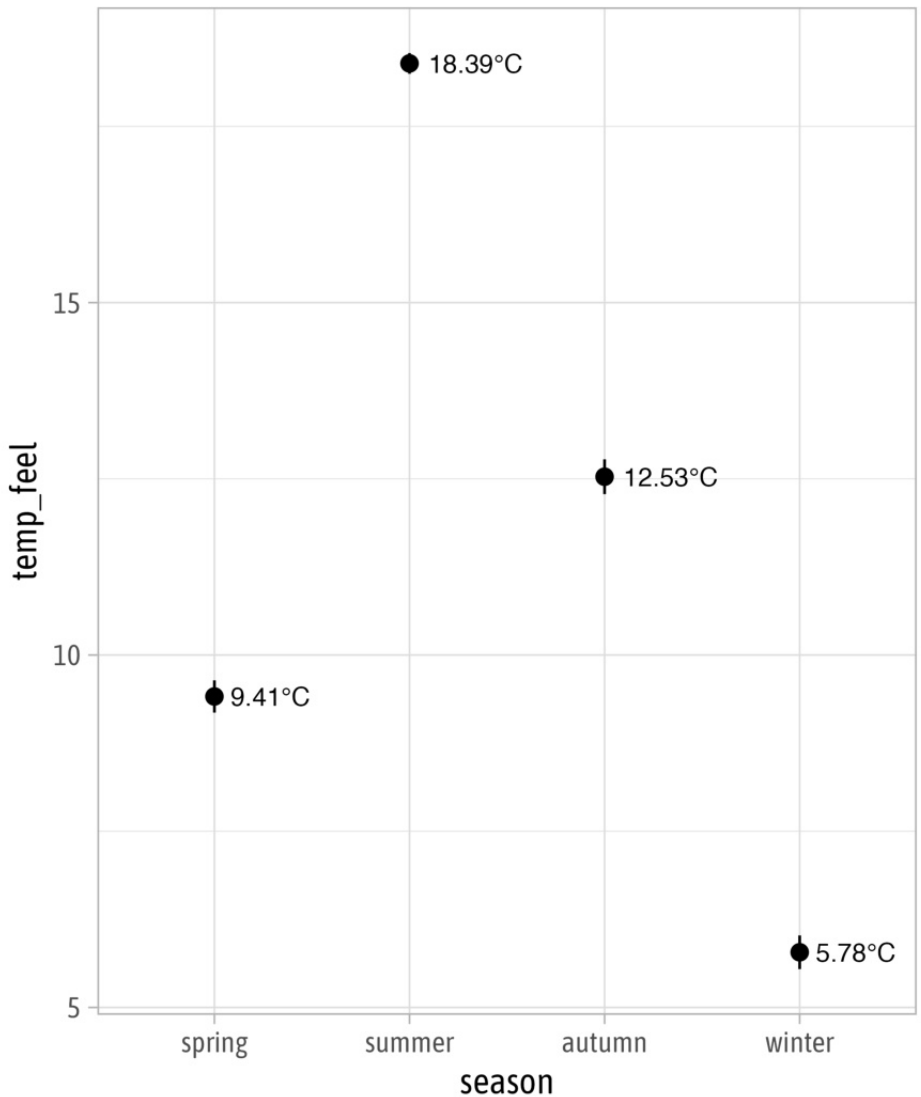
Statistical Summaries

```
ggplot(
    bikes,
    aes(x = season, y = temp_feel)
  ) +
  stat_summary() +
  stat_summary(
    fun = mean,
    geom = "text",
    aes(label = after_stat(y))
  )
```



Statistical Summaries

```
ggplot(
    bikes,
    aes(x = season, y = temp_feel)
  ) +
  stat_summary() +
  stat_summary(
    fun = mean,
    geom = "text",
    aes(label = after_stat(
      paste0(round(y, 2), "°C"))
    ),
    hjust = -.2,
    size = 3.5
  )
```



Aspect Ratios

Cédric Scherer // Data Visualization Society // March 9, 2023

How to Work with Aspect Ratios

- don't rely on the Rstudio viewer pane!
- once you have a "it's getting close" prototype, settle on a plot size

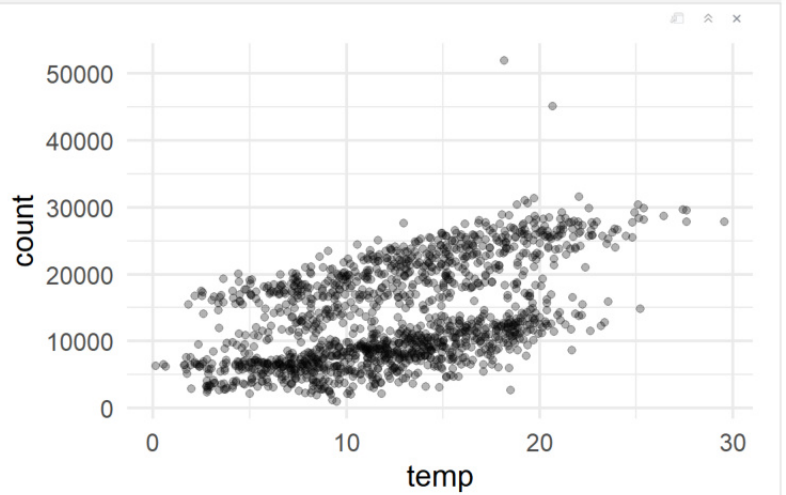
- **Approach 1:** save the file to disk and inspect it; go back to your IDE
  - tedious and time-consuming...

- **Approach 2:** use a qmd or rmd with inline output and chunk settings
  - set `fig.width` and `fig.height` per chunk or globally

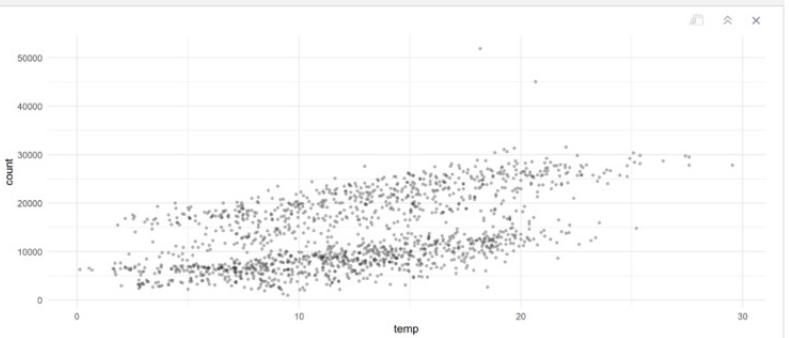
- **Approach 3:** use our {camcorder} package
  - saves output from all `ggplot()` calls and displays it in the viewer pane

Setting Plot Sizes in Rmd's

````
```{r plot-object}
g <- ggplot(bikes, aes(x = temp, y = count)) +
  geom_point(alpha = .3, size = 2) +
  theme_minimal(base_size = 18)
```

```{r plot-a}
g
```
````




````
```{r plot-b, fig.width=10, fig.height=4}
g
```
````



Setting Plot Sizes via {camcorder}

plot-size-camcorder-example.R

Source on Save | Run | Source

```r
library(tidyverse)
library(camcorder)

bikes <- readr::read_csv(
  "https://raw.githubusercontent.com/z3tt/graphic-design-ggplot2
)

gg_record(
  dir = here::here("temp_plots"), # where to save the recording
  device = "png", # device to use to save images
  width = 10, # width of saved image
  height = 5.5, # height of saved image
  units = "in", # units for width and height
  dpi = 600 # dpi to use when saving image
)

ggplot(bikes, aes(x = temp, y = count)) +
  geom_point(alpha = .3, size = 2) +
  theme_minimal(base_size = 18)
```

Files | Plots | Packages | Help | Viewer | Presentation