ggplot

Maria Cuellar

2025-09-16

ggplot

The Grammar of Graphics was a book published in 1999 by a statistician named Leland Wilkinson, and later Hadley Wickham used it to write a package called ggplot in R. ggplot stands for Grammar of Graphics plot.

It is called a grammar because it is a structure for producing plots that resembles a language. Nearly every current software tool used to build plots has been informed by this book. Its influence can be found in Tableau, Plotly, and the Python libraries bokeh, altair, seaborn, and plotnine. The most complete implementation of the grammar is found in an R package called ggplot2 by Hadley Wickham.

In Wickham's adaptation of the grammar of graphics, a plot can be decomposed into seven elements:

- *Data*: The data frame that contains the data you want to visualize.
- *Aesthetic* (aes) mapping of the variables in the data to visual cues: What is your x or y.

Why it's called "aesthetics": It's not about style or beauty — it's about visual mappings. An aesthetic is anything that controls what you see on the plot. (x and y, color, fill, shape, size, alpha (transparency), linetype). This contrasts with "settings". If you don't want an aesthetic to depend on data, you set it outside aes(). For example: `ggplot(mtcars, aes(x = mpg, y = hp)) + geom_point(color = "red")`.

- *Geometry*: Is used to encode the observations on the plot: What kind of plot you're making, a histogram, a bar plot, a line, etc.
- *Facets*: If you want to split up the plots by categories, e.g., one plot for females and one for males.
- *Statistics*: If you want to fit a model to the data. Geoms decide how to draw (points, lines, bars). Stats decide what to draw by computing summaries (counts, means, model fits).
- *Coordinates*: If you want to change the scales. Theme: If you want to make it pretty.

Data

We will use the `mpg` dataset, which is built into the `ggplot2` package, which is part of the `tidyverse` package.

```
# load packages
library(tidyverse)
library(ggthemes)
```

```
# See the data we'll be using
mpg
```

```
## # A tibble: 234 x 11
##    manufacturer model      displ  year   cyl trans drv     cty   hwy fl    class
##    <chr>        <chr>      <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
##  1 audi         a4           1.8  1999     4 auto~ f        18    29 p     comp~
##  2 audi         a4           1.8  1999     4 manu~ f        21    29 p     comp~
##  3 audi         a4           2    2008     4 manu~ f        20    31 p     comp~
##  4 audi         a4           2    2008     4 auto~ f        21    30 p     comp~
```

```
##  5 audi         a4           2.8  1999     6 auto~ f        16    26 p     comp~
##  6 audi         a4           2.8  1999     6 manu~ f        18    26 p     comp~
##  7 audi         a4           3.1  2008     6 auto~ f        18    27 p     comp~
##  8 audi         a4 quattro   1.8  1999     4 manu~ 4        18    26 p     comp~
##  9 audi         a4 quattro   1.8  1999     4 auto~ 4        16    25 p     comp~
## 10 audi         a4 quattro   2    2008     4 manu~ 4        20    28 p     comp~
## # i 224 more rows
```

We will use the variables `cty` and `hwy` as our quantitative variables, and the variables `drv`, `cyl` as our categorical ones.

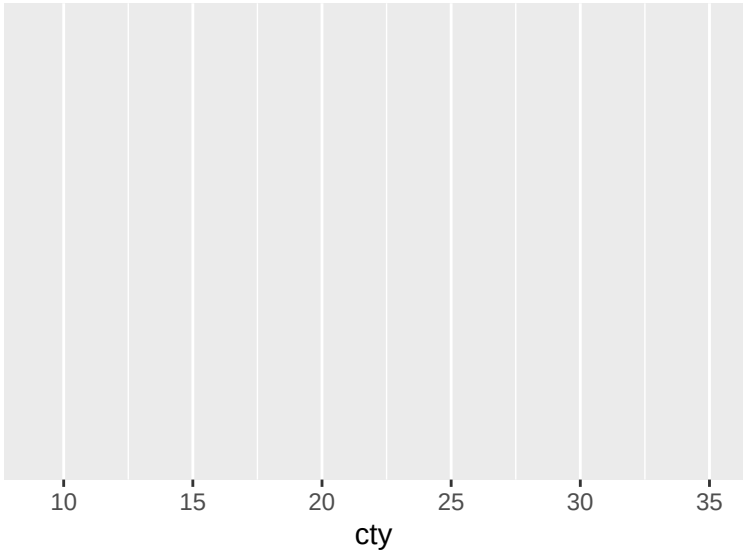
Different elements of a plot

If you only use ggplot, then nothing happens.

```
mpg %>% ggplot()
```


If you add the aesthetics, you get a coordinate system where the plot will go. We've defined two variables, x and y, that are quantitative variables.

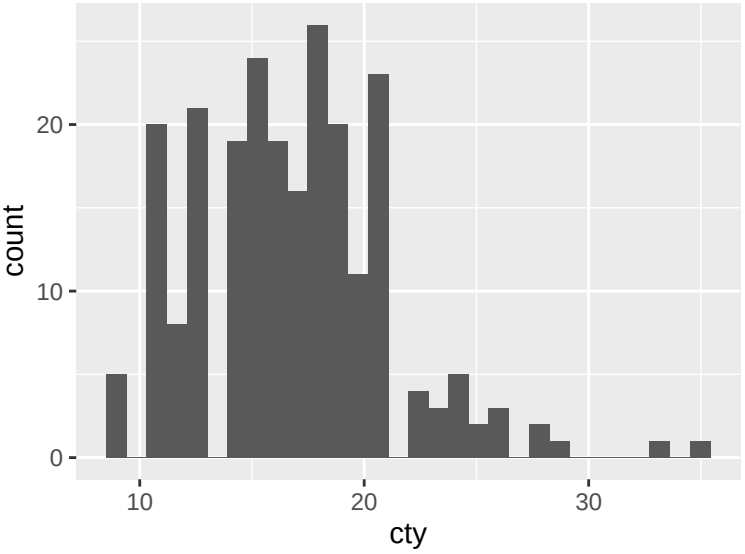
```
mpg %>% ggplot(aes(x=cty))
```



If you then add the geometry (shortened as geom), then you get a plot.

```
mpg %>% ggplot(aes(x=cty)) + geom_histogram()
```

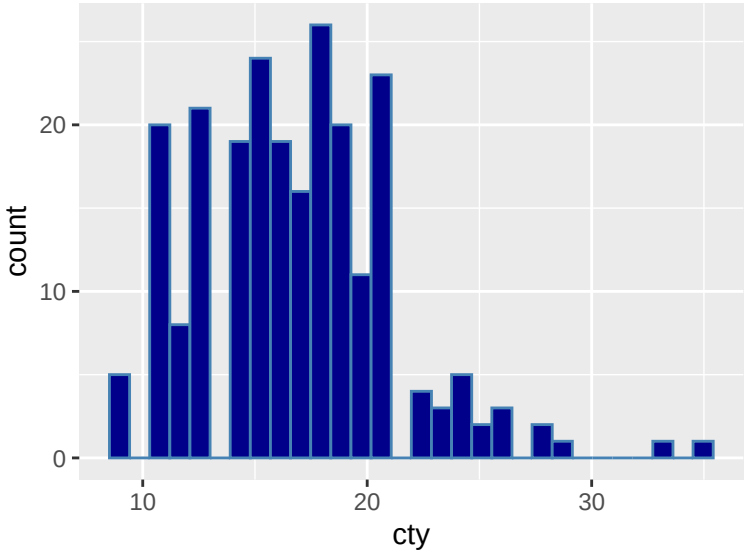
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



You can change the color of the points for fun, without having any particular meaning. Note that there is no legend. This is SETTING a constant color, not mapping (no legend). That's why it's outside aes.

```
mpg %>% ggplot(aes(x=cty)) + geom_histogram(color = "steelblue", fill="darkblue")
```

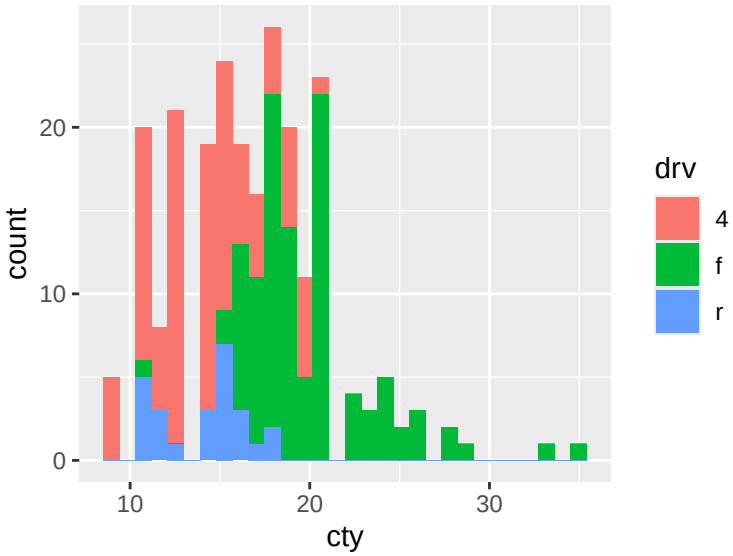
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Or you can use color with meaning. Here we use it to tell us how the points split up by a third categorical variable, drv. This splits up the points by color, one for each category of drv. This is MAPPING color to a variable (legend appears).

```
mpg %>% ggplot(aes(x=cty, fill = drv)) + geom_histogram()
```

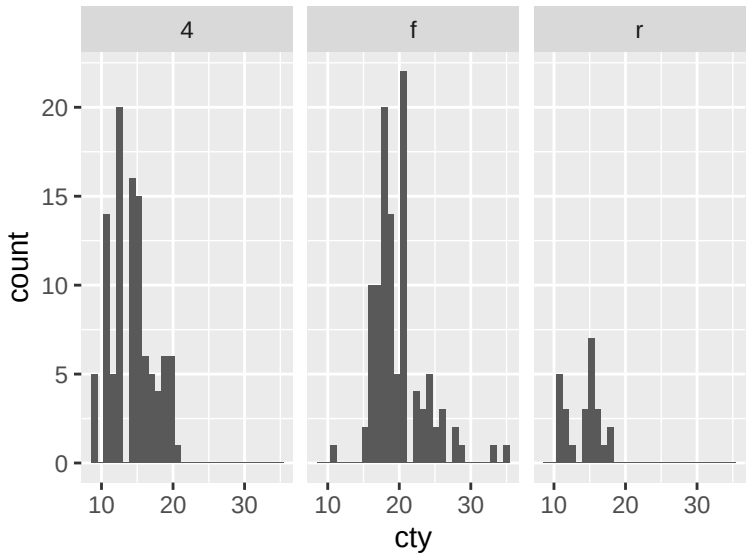
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



You can use facets instead to see the differences in the points according to a third categorical variable. This splits up the plot into three plots, one for each category of drv.

```
mpg %>% ggplot(aes(x=cty)) + geom_histogram() + facet_grid(~drv)
```

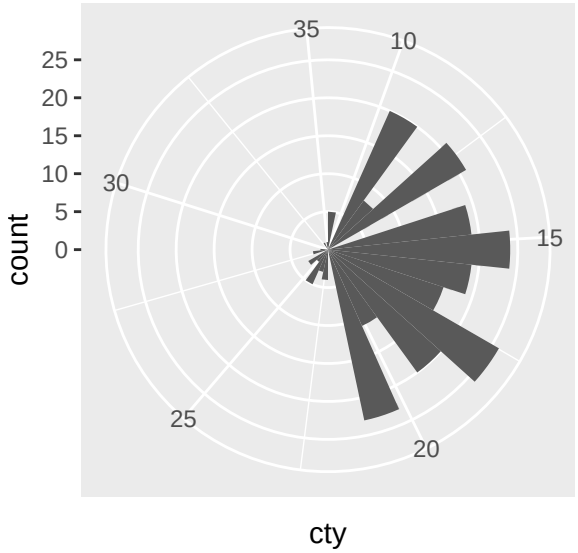
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



You can also change the coordinates (although we won't do this often).

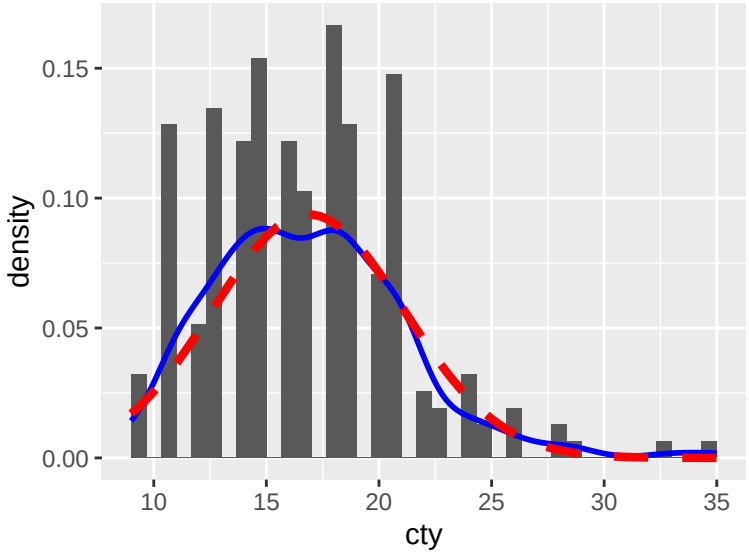
```
# coord_polar() for pie/polar charts
mpg %>% ggplot(aes(x=cty)) + geom_histogram() + coord_polar()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



You can add "statistics", meaning some modeling to your plot. Note that ggplot is doing this in the background - we will learn later how to model ourselves.

```
mpg %>% ggplot(aes(x=cty)) +
  geom_histogram(bins=40, aes(y = after_stat(density))) +
  stat_density(geom = "line", color = "blue", linewidth = 1) + # Computes a kernel density estimate
  stat_function(fun = dnorm,
                args = list(mean = mean(mpg$cty), sd = sd(mpg$cty)),  # Plots any function (e.g., theor
                color = "red", linetype = "dashed", linewidth=1.5)
```

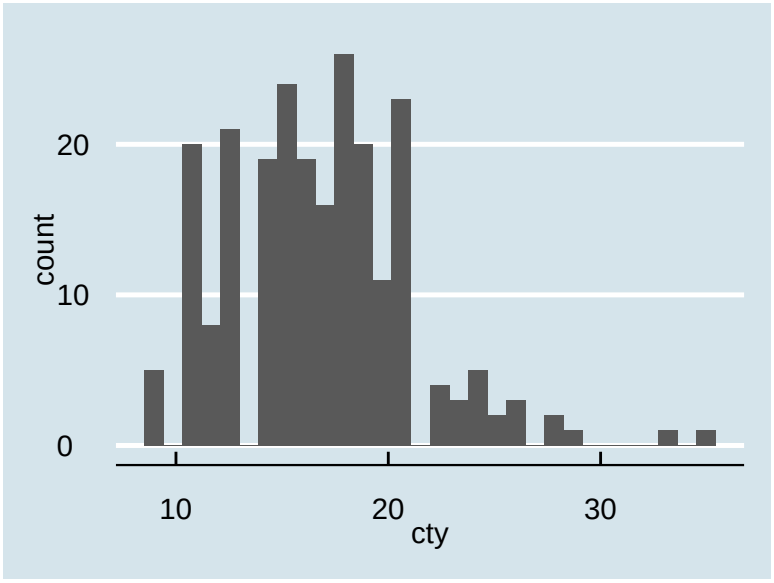


Finally, you can change the look of the plot.

Themes are made for you:

```
mpg %>% ggplot(aes(x=cty)) + geom_histogram() + theme_economist()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

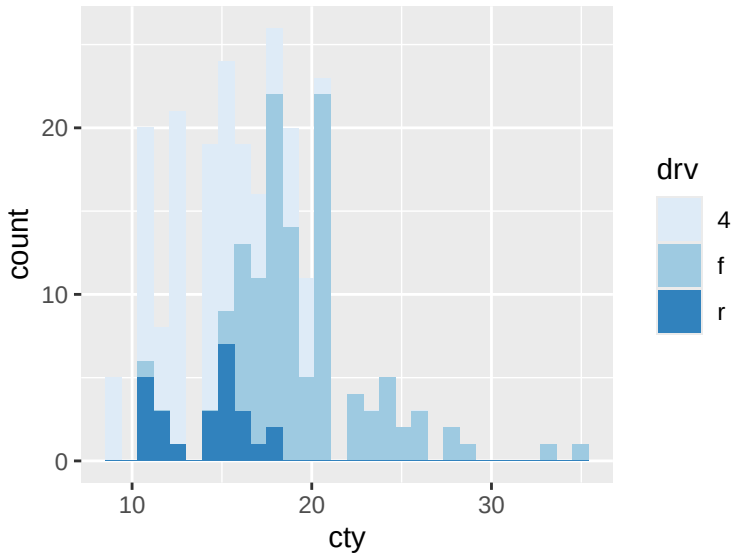



```
# make sure you've loaded the ggthemes package
```

And you can also change the palettes. There's a nice set of palettes called the Brewer palettes: https://r-graph-gallery.com/38-rcolorbrewers-palettes.html. Designed by Cynthia Brewer: https://en.wikipedia.org/wiki/Cynthia_Brewer.

```
library(tidyverse)
mpg %>% ggplot(aes(x=cty, fill=drv)) + geom_histogram() + scale_fill_brewer(palette = )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Single variable EDA

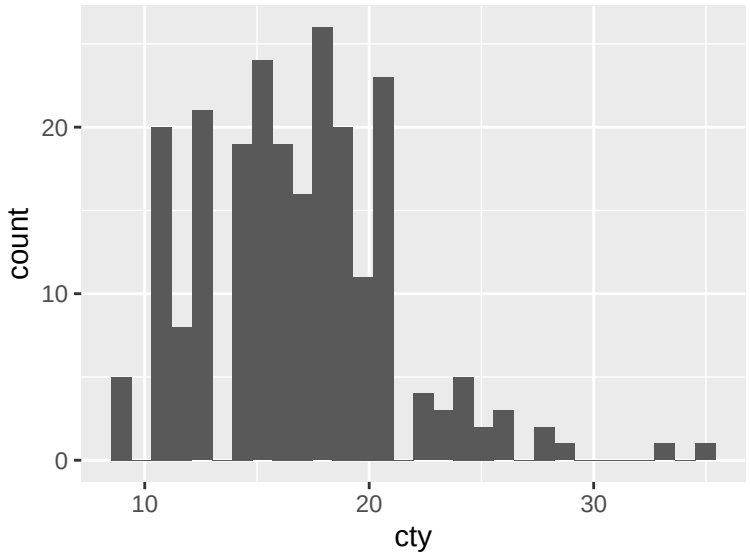
Quantitative varible:

Histogram

Plot a histogram to visualize a single quantitative variable.

```
mpg %>% ggplot(aes(x=cty)) + geom_histogram()
```

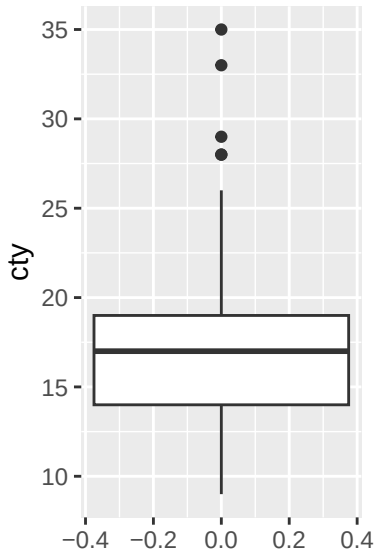
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Boxplot

Plot a boxplot to visualize a single quantitative variable.

```
mpg %>% ggplot(aes(y=cty)) + geom_boxplot()
```

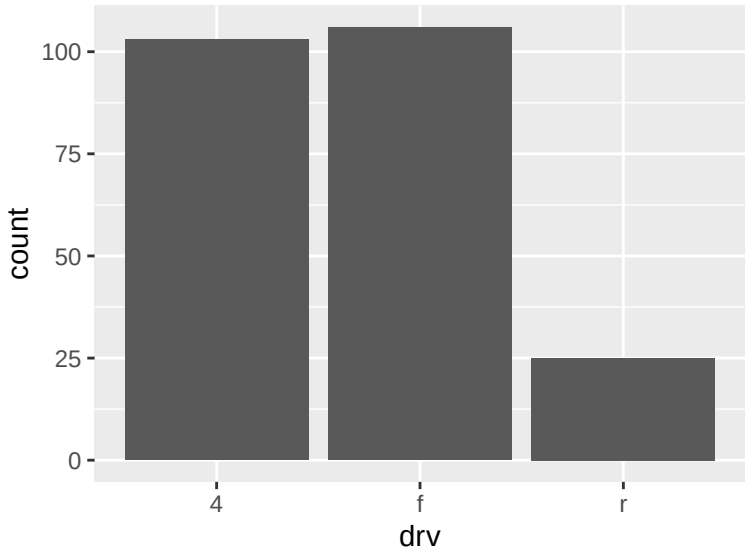


Categorical variable

Barplot

Draw a barplot to visualize a single categorical variable.

```
mpg %>% ggplot(aes(x=drv)) + geom_bar()
```



Pie chart

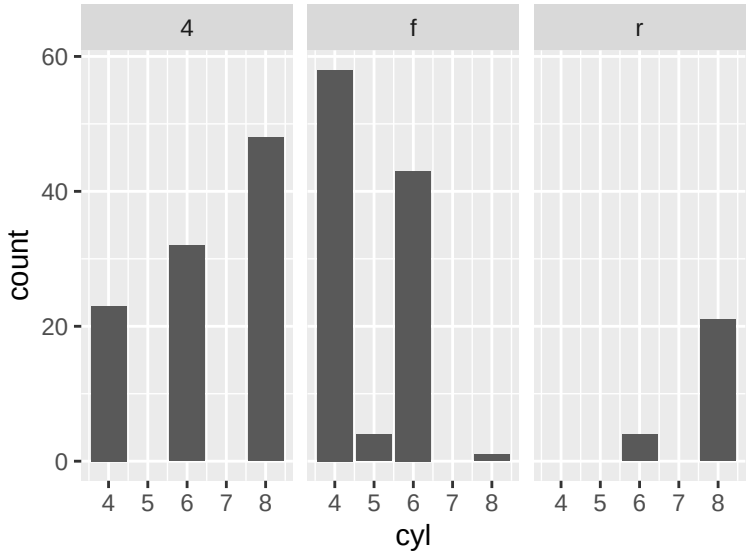
(Actually, don't use pie charts. They're often more deceiving than helpful - this is because humans are bad at comparing areas to each other. Just stick to barplots, or tables.)

```
#mpg %>% ggplot(aes(x="", fill=drv)) + geom_bar() + coord_polar("y") + theme_void()
```

Two categorical variables

Side-by-side barplots

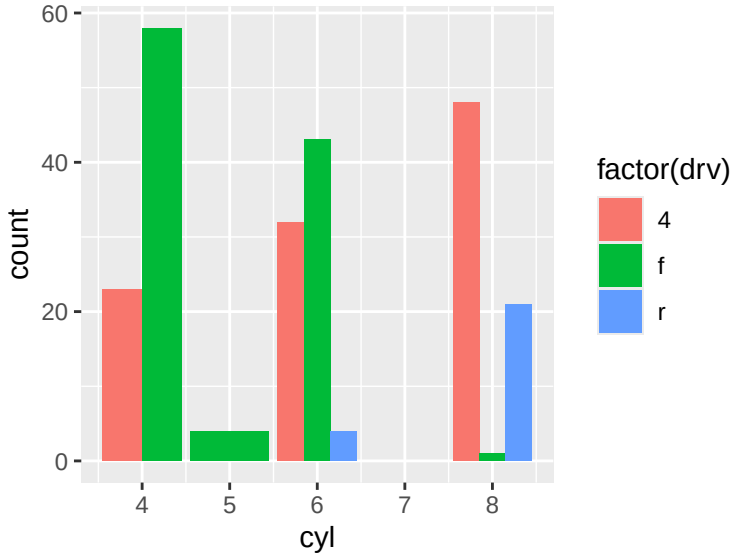
```
library(tidyverse)
mpg %>% ggplot(aes(x=cyl)) + geom_bar() + facet_wrap(~drv)
```



Different color barplots

Note: position dodge is almost always prefereable to identity because identity stacks the bars up, and it's very difficult for humans to compare the sizes of the bars when they start at different heights.

```
mpg %>% ggplot(aes(x=cyl, fill=factor(drv))) + geom_bar(position="dodge")
```



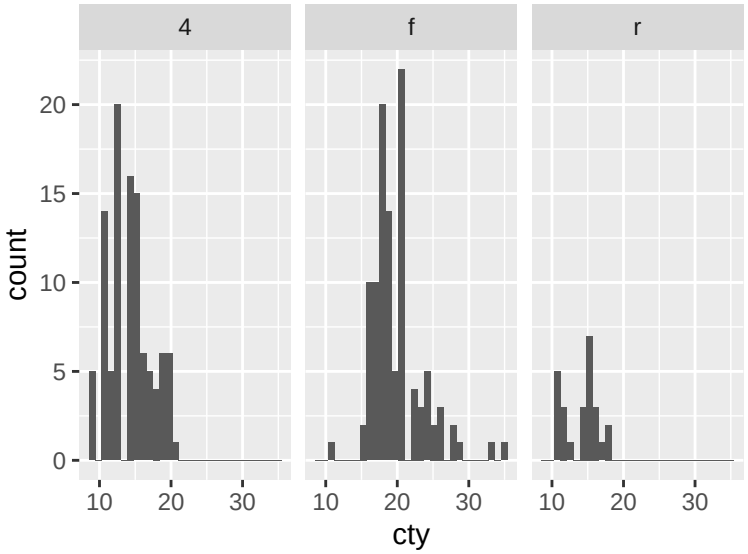

```
# note that sometimes if you don't add factor for the aesthetics, ggplot gets confused
```

A quantitative variable and a categorical variable

Side-by-side histograms

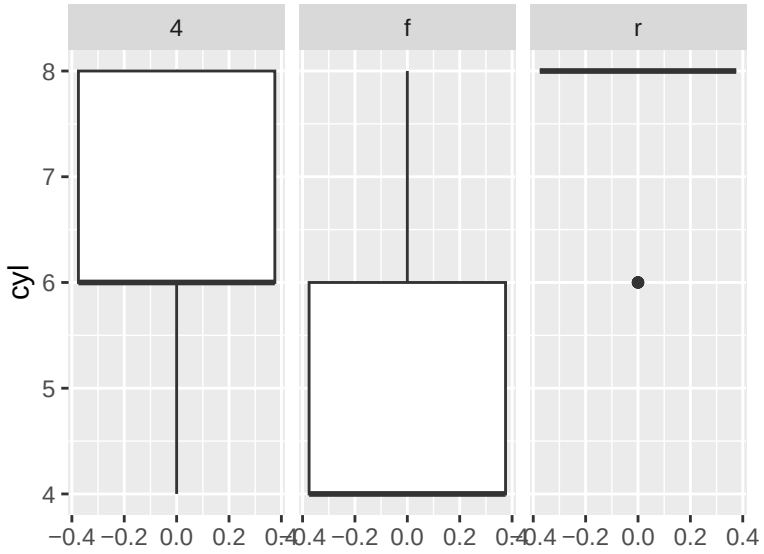
```
mpg %>% ggplot(aes(x=cty)) + geom_histogram() + facet_grid(~drv)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Side-by-side boxplots

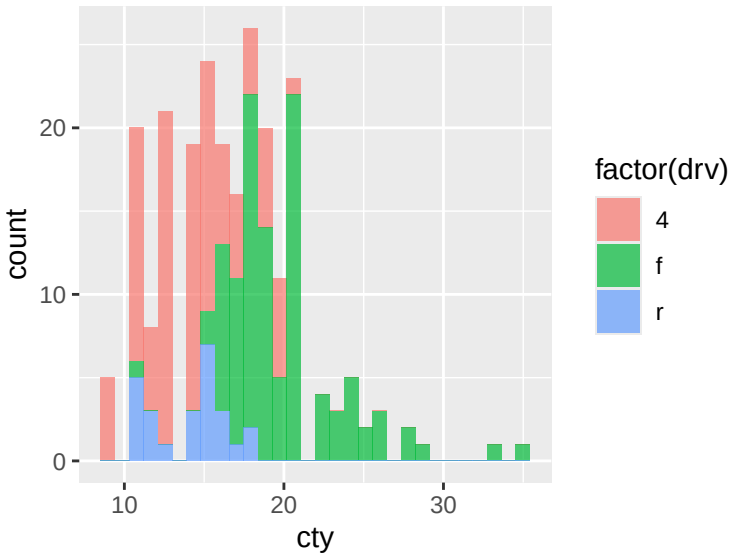
```
mpg %>% ggplot(aes(y=cyl)) + geom_boxplot() + facet_grid(~drv)
```



Different color histograms

```
mpg %>% ggplot(aes(x=cty, fill=factor(drv))) + geom_histogram(alpha=.7)
```

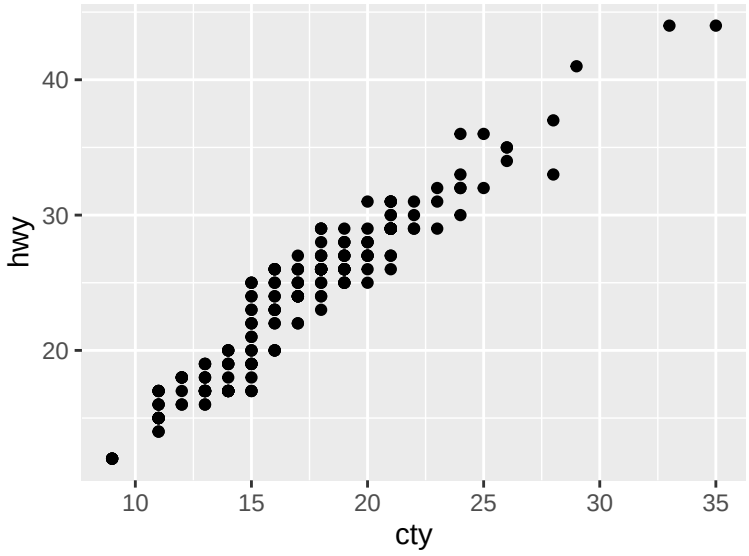
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Two quantitative variables

Scatterplot

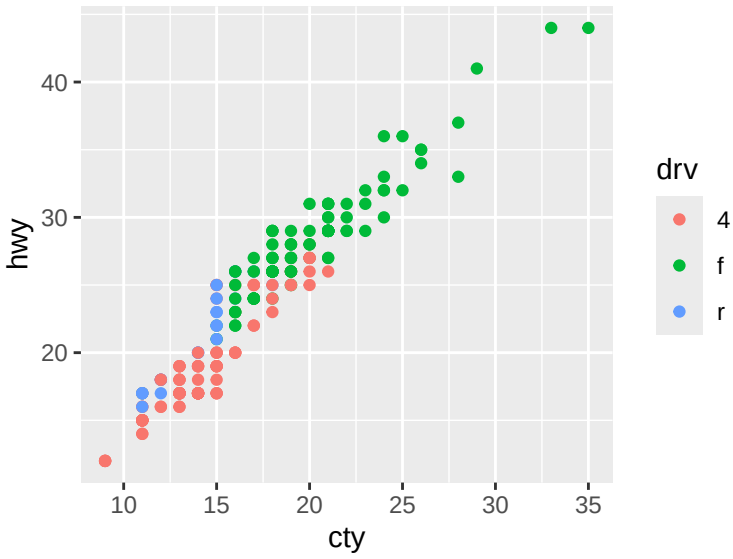
```
mpg %>% ggplot(aes(x=cty, y=hwy)) + geom_point()
```



Two quantitative variables and a categorical variable

Scatterplot with different colors by the categories.

```
mpg %>% ggplot(aes(x=cty, y=hwy, color=drv)) + geom_point()
```



Three quantitative variables

Scatterplot with different colors by the shade of the third quant variable.

```
mpg %>% ggplot(aes(x=cty, y=hwy, color=displ)) + geom_point()
```