Data exploration

Reference: “Data Science” Chapter 1.

Types of features in a data set

Categorical: the answer to a multiple choice question:

- Chevy/Honda/Tesla
- ice cream/cake/pie

Ordinal: categorical, where the answers have an ordering but not a magnitude

- Poor, Moderate, Good, Great
- Private, Corporal, Lieutenant, Colonel, General

Numerical: numbers, whether integer or real-valued

- Beware the “faux numerical” ordinal scale

The basics of data exploration

- Boxplots
- Scatter plots
- Line graphs
- Faceting
- Tables
- Grouping/piping/summarizing
- Bar plots
- Histograms/density plots

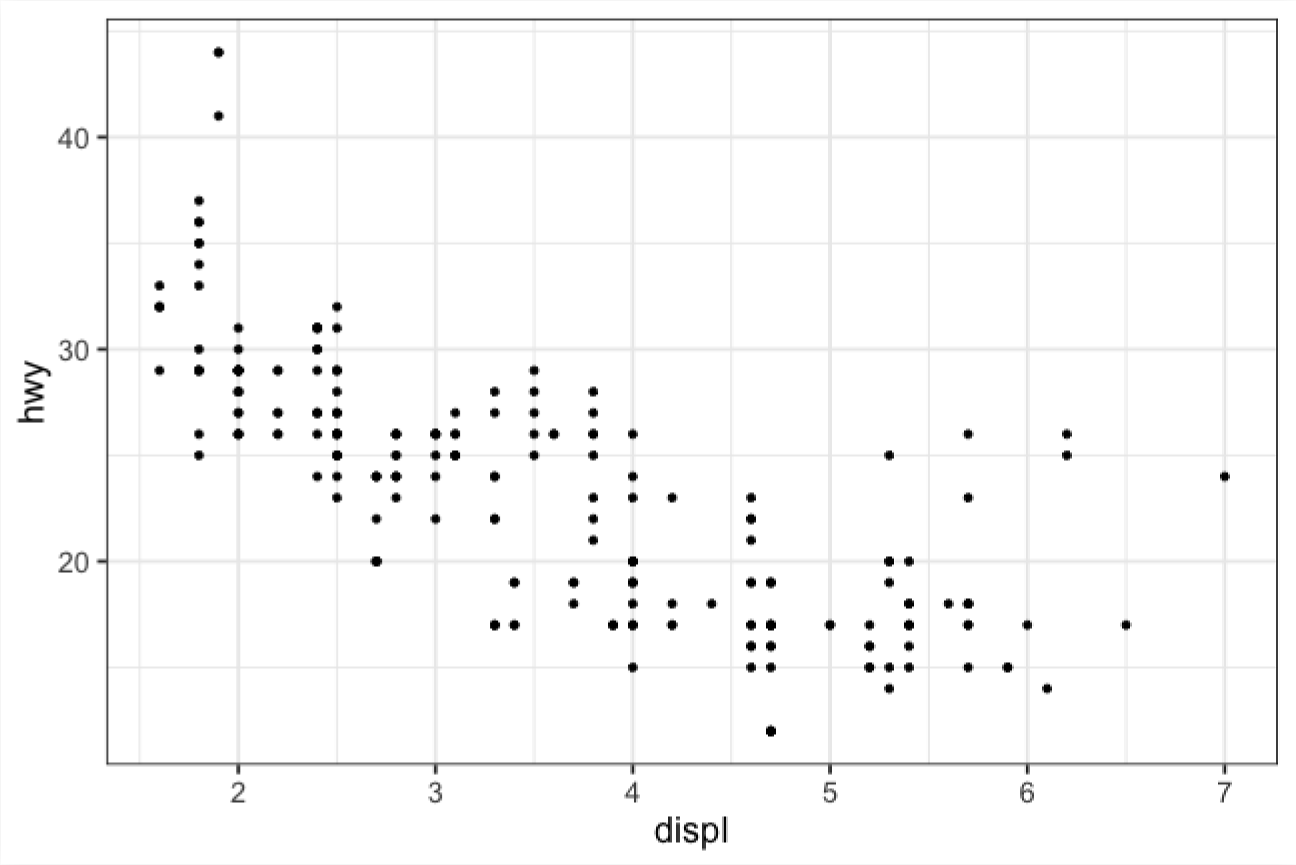
Scatter plots

Here are several rows of a data frame about cars. Every row is a car. Every column is a feature describing the car.

```
# A tibble: 6 x 12
  manufacturer model displ  year   cyl trans drv     cty   hwy fl    class
  <chr>        <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
1 nissan       path…   3.3  1999     6 manu… 4        15    17 r     suv
2 ford         f150…   5.4  1999     8 auto… 4        11    15 r     pick…
3 toyota       camry   3    1999     6 manu… f        18    26 r     mids…
4 dodge        cara…   3.3  1999     6 auto… f        16    22 r     mini…
5 dodge        ram …   5.2  1999     8 manu… 4        11    16 r     pick…
6 chevrolet    k150…   6.5  1999     8 auto… 4        14    17 d     suv
# … with 1 more variable: orig.id <chr>
```

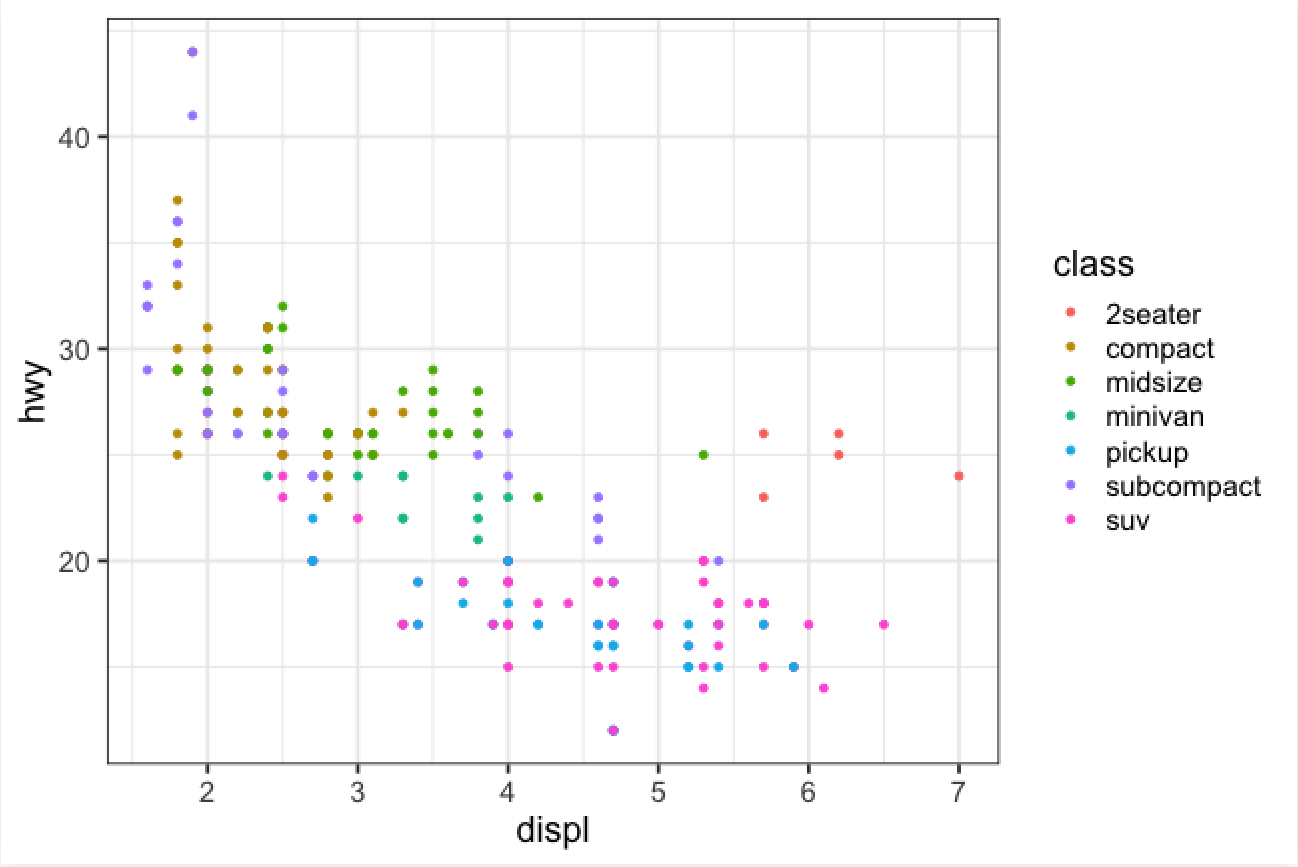
Scatter plots

For numerical data, our workhorse is the humble scatter plot.



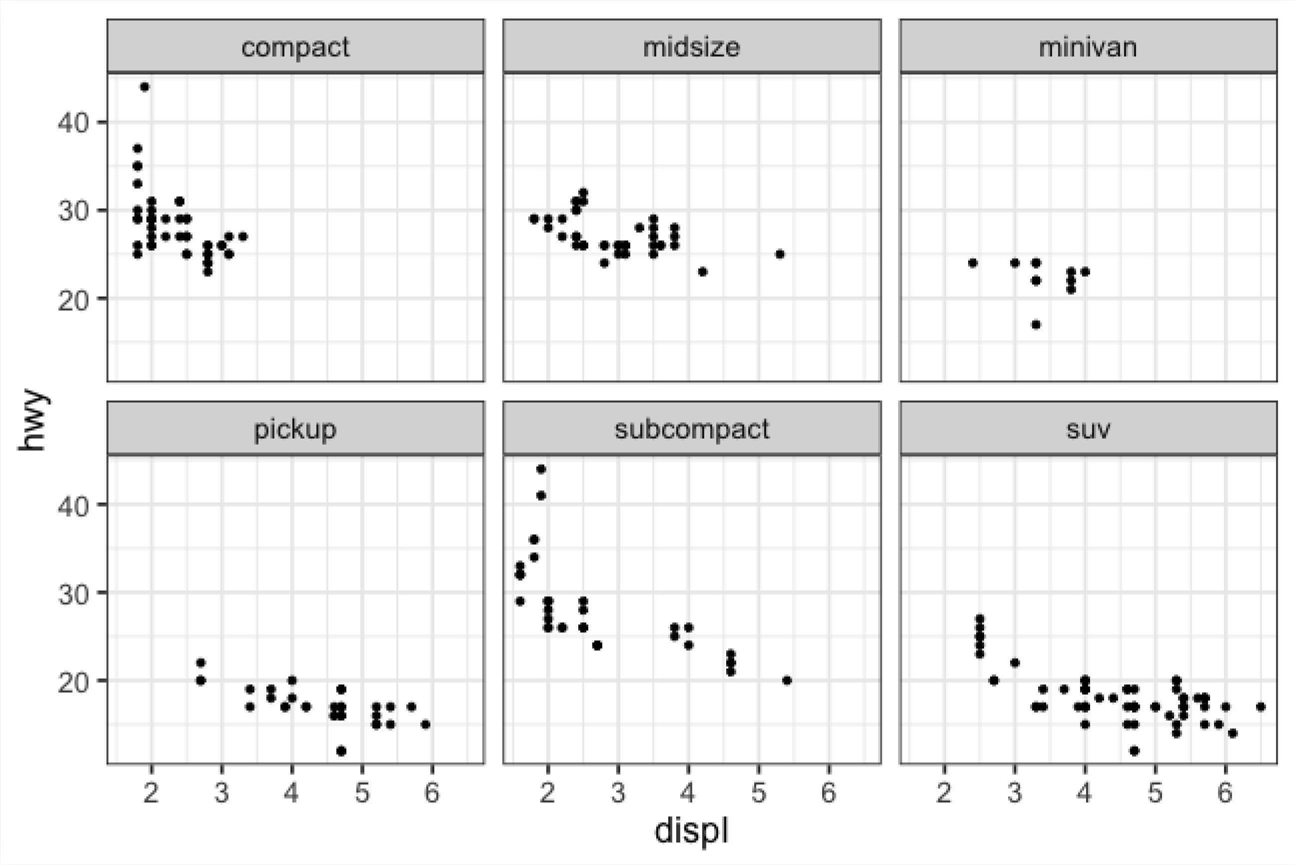
Scatter plots

There are lots of strategies for enriching a scatter plot with additional information. We can color points according to a key…



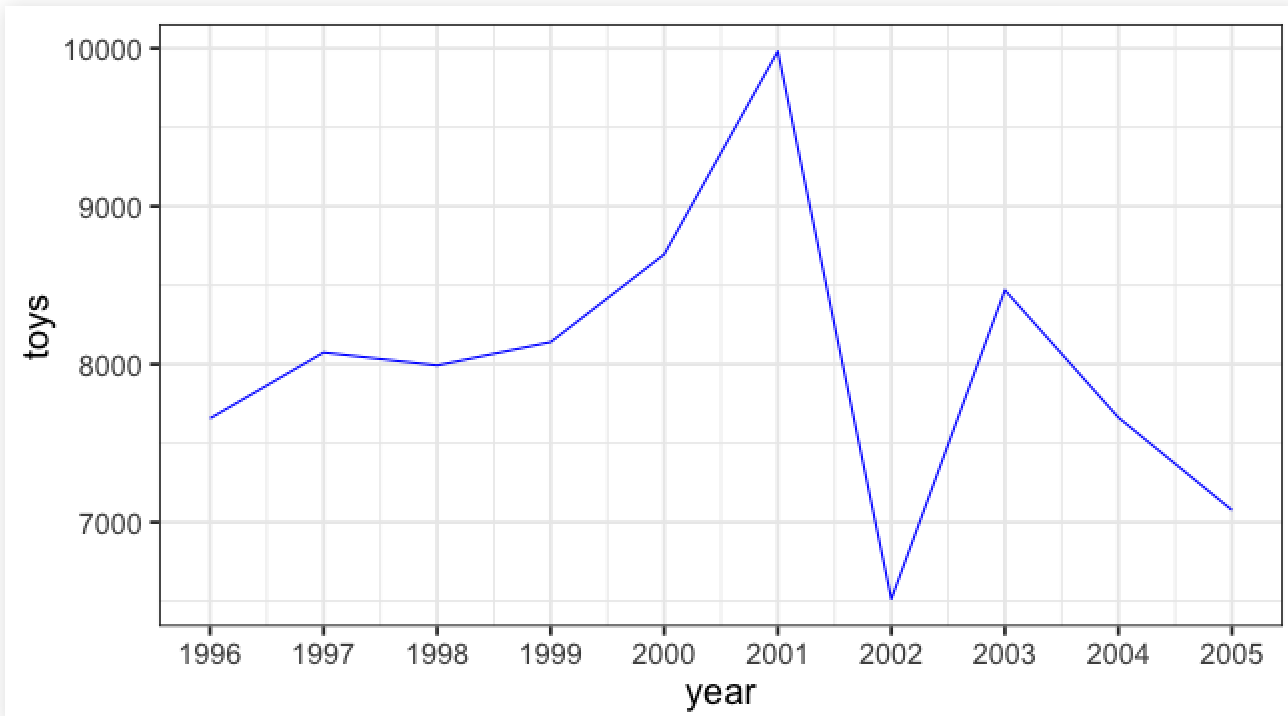
Scatter plots: faceting

Or *facet* on a third variable…



Line graphs

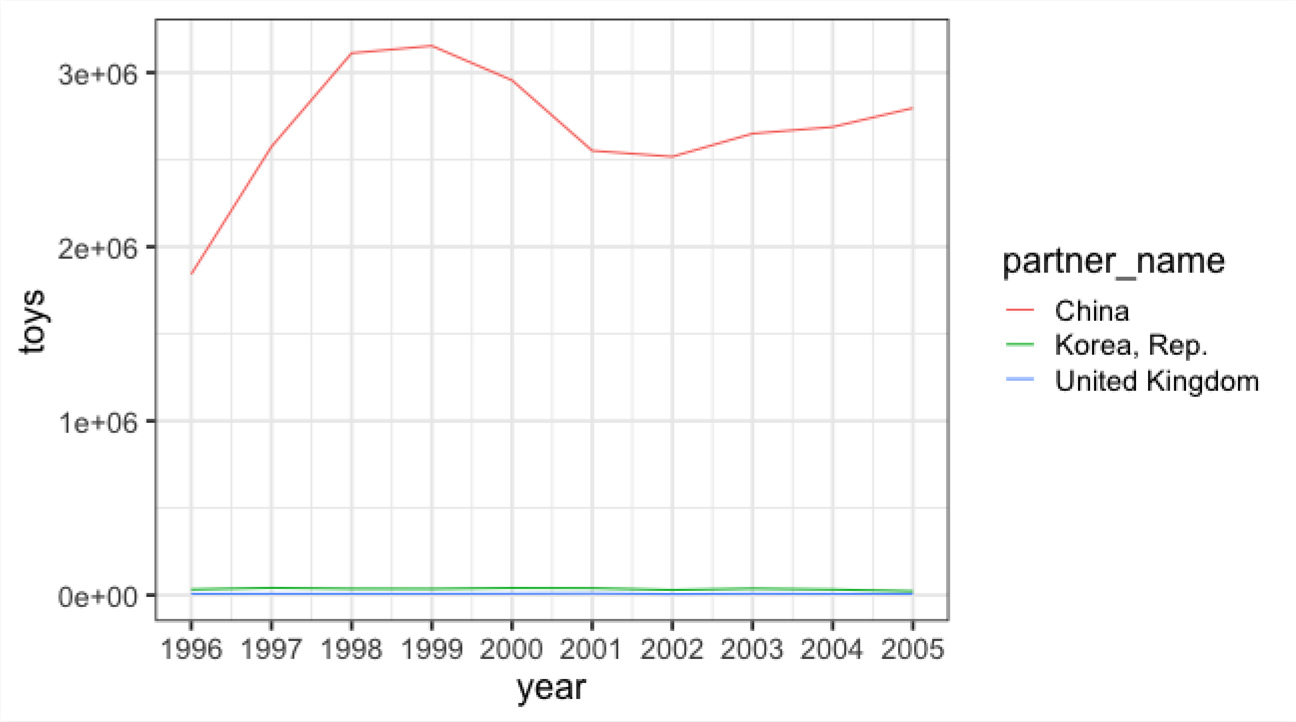
When it's important to emphasize continuity of a set of points (e.g. over time), use a line graph.



Total value of toy imports from the United Kingdom over time (thousands USD). Q: what might account for the big spikes?

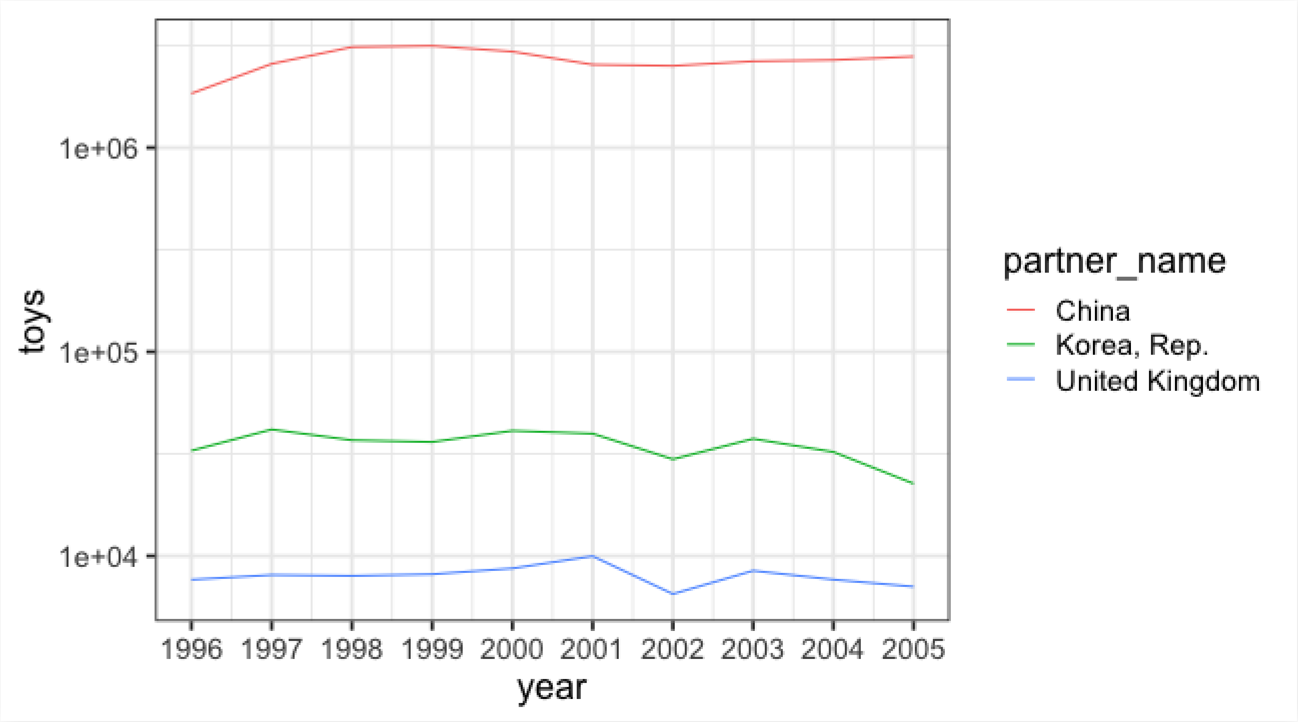
Line graphs

Actually, all those Harry Potter toys come from China…



Line graphs

This is what logarthimic scales were invented for :-)



Note: OK not to start this y axis at 0, because we don't use height to judge relative size on a log scale.

Tables

Here are the first several rows of a data frame about passengers on the Titanic. Every row is a passenger. Every column is a feature describing the passenger.

```
                                name survived    sex     age passengerClass
1   Allen, Miss. Elisabeth Walton      yes female 29.0000            1st
2  Allison, Master. Hudson Trevor      yes   male  0.9167            1st
3    Allison, Miss. Helen Loraine       no female  2.0000            1st
4 Allison, Mr. Hudson Joshua Crei       no   male 30.0000            1st
5 Allison, Mrs. Hudson J C (Bessi       no female 25.0000            1st
6             Anderson, Mr. Harry      yes   male 48.0000            1st
```

We see both categorical (sex, passengerClass, survived) and numerical (age) variables.

Tables

A natural thing might be to cross-tabulate survival by passenger class:

```
xtabs(~survived + passengerClass, data=TitanicSurvival)
```

```
        passengerClass
survived 1st 2nd 3rd
     no  123 158 528
     yes 200 119 181
```

We're literally just counting how many passengers have each combination of features. (If you know Excel: this is like a pivot table.)

An aside: piping

A really useful operation is *piping*. Example:

```
a = log(3)
b = exp(a)
c = sqrt(b)
c
```

```
[1] 1.732051
```

versus:

```
a = log(3)
a %>% exp() %>% sqrt()
```

```
[1] 1.732051
```

Tables

We can pipe our table to `prop.table` to standardize along the columns (`margin=2`):

```
xtabs(~survived + passengerClass, data=TitanicSurvival) %>%
  prop.table(margin=2)
```

```
        passengerClass
survived       1st       2nd       3rd
     no  0.3808050 0.5703971 0.7447109
     yes 0.6191950 0.4296029 0.2552891
```

Now you can compare survival proportions by passenger class.

Tables

Seven decimal places seems overkill – let's round to third decimal place by piping the the result to round:

```
xtabs(~survived + passengerClass, data=TitanicSurvival) %>%
  prop.table(margin=2) %>%
  round(3)
```

```
        passengerClass
survived   1st   2nd   3rd
     no  0.381 0.570 0.745
     yes 0.619 0.430 0.255
```

Tables

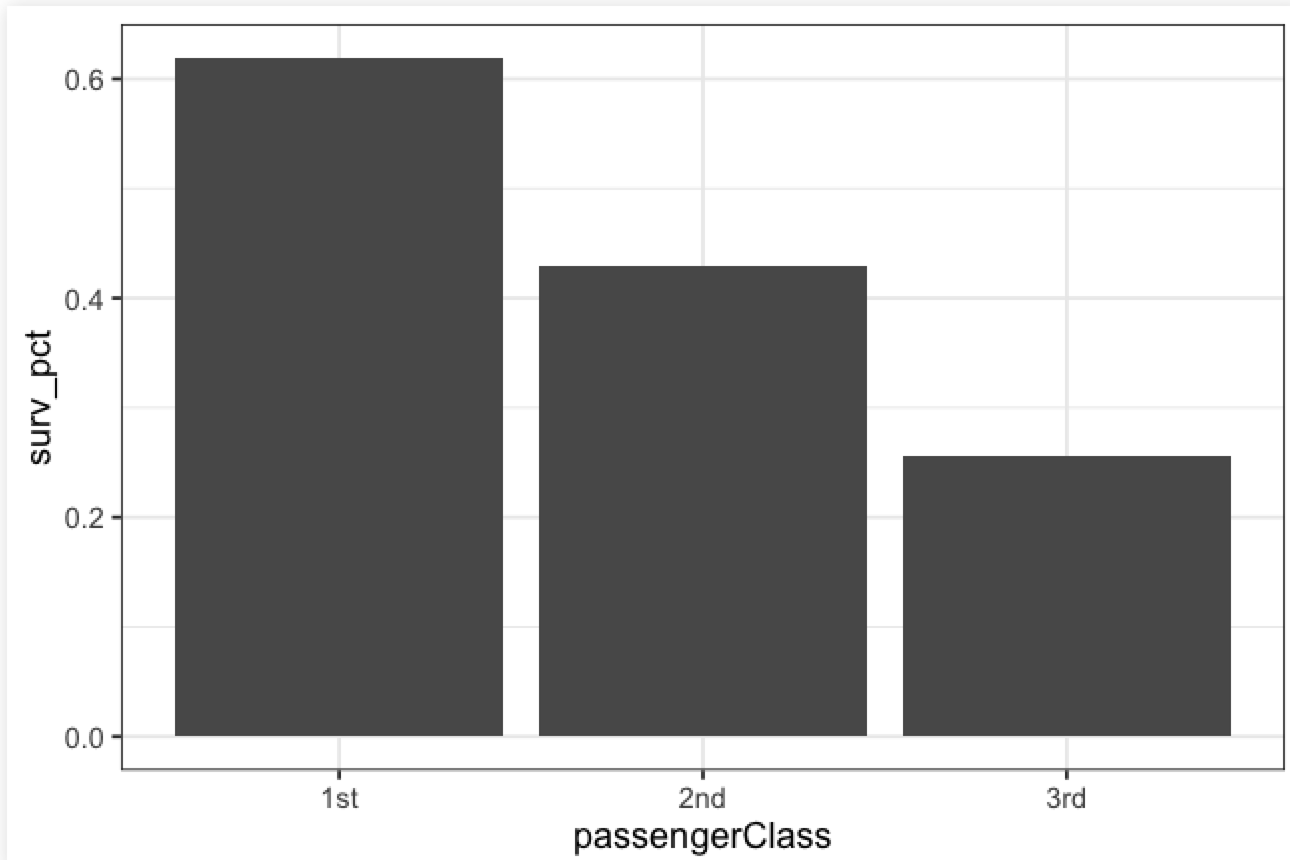
If you pipe the result to `kable`, you'll get a prettier table (formatted in Markdown):

```
library(knitr)
xtabs(~survived + passengerClass, data=TitanicSurvival) %>%
  prop.table(margin=2) %>%
  round(3) %>%
  kable()
```

	1st	**2nd**	**3rd**
no	0.381	0.57	0.745
yes	0.619	0.43	0.255

Bar plots

We can also turn this information into a bar plot.



Remember to start your y-axis at 0!

Grouping and summarizing

Another good use of tables is to display *summary statistics* of numerical variables. For example, here's how we'd use pipes to compute the average age by passenger class:

```
TitanicSurvival %>%
  group_by(passengerClass) %>%
  summarize(mean_age = mean(age,na.rm=TRUE))
```

```
# A tibble: 3 x 2
  passengerClass mean_age
  <fct>             <dbl>
1 1st                39.2
2 2nd                29.5
3 3rd                24.8
```

Use `group_by` to group cases according to the `passengerClass` variable. Then compute a summary statistic by averaging `age`. (`na.rm = TRUE` tells R to ignore missing values.)

Grouping and summarizing

Now with two variables defining the groups:

```
TitanicSurvival %>%
  group_by(passengerClass, survived) %>%
  summarize(mean_age = mean(age,na.rm=TRUE))
```

```
# A tibble: 6 x 3
# Groups:   passengerClass [3]
  passengerClass survived mean_age
  <fct>          <fct>       <dbl>
1 1st            no           43.2
2 1st            yes          36.8
3 2nd            no           33.2
4 2nd            yes          24.9
5 3rd            no           26.0
6 3rd            yes          21.5
```

This gives us a “flat” table.

Grouping and summarizing

If you want to un-flatten the table, use `spread`:

```
TitanicSurvival %>%
  group_by(passengerClass, survived) %>%
  summarize(mean_age = mean(age,na.rm=TRUE)) %>%
  spread(survived, mean_age)
```

```
# A tibble: 3 x 3
# Groups:   passengerClass [3]
  passengerClass    no   yes
  <fct>          <dbl> <dbl>
1 1st             43.2  36.8
2 2nd             33.2  24.9
3 3rd             26.0  21.5
```

`spread` says to spread out the levels of the `survived` variables along the columns of the table and put the `mean_age` variable in each entry.

Grouping and summarizing

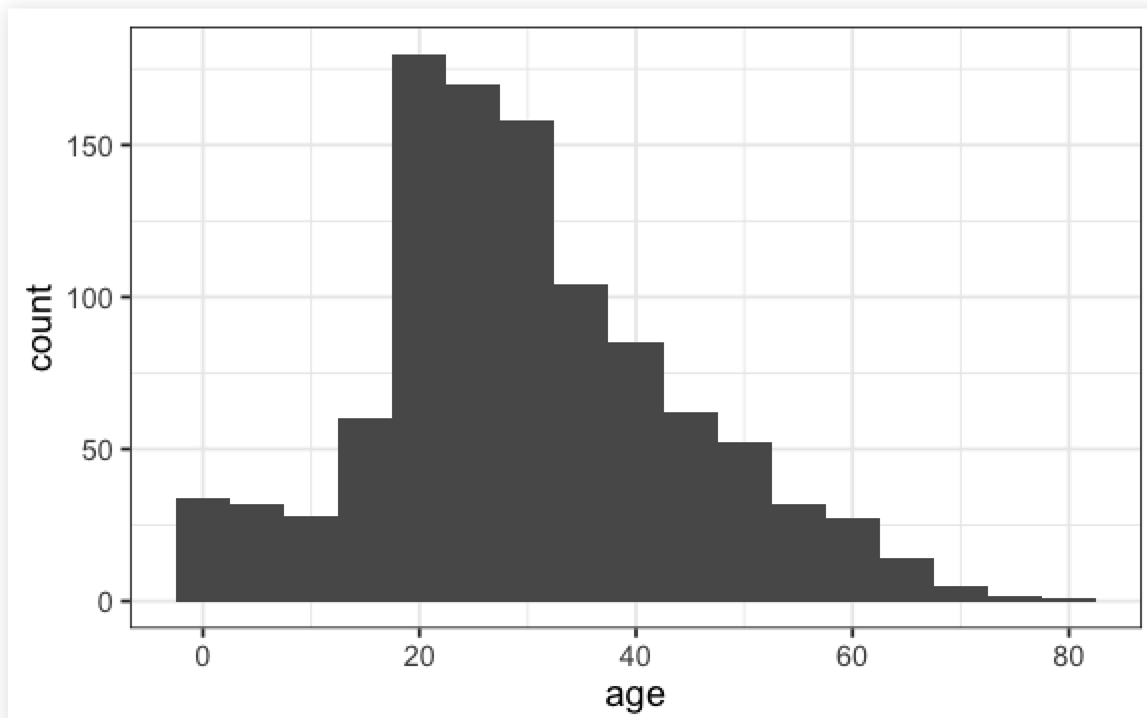
You can compute lots of summary statistics this way:

```
TitanicSurvival %>%
  group_by(passengerClass) %>%
  summarize(mean_age = mean(age,na.rm=TRUE),
            sd_age = sd(age,na.rm=TRUE),
            max_age = max(age,na.rm=TRUE))
```

```
# A tibble: 3 x 4
  passengerClass mean_age sd_age max_age
  <fct>             <dbl>  <dbl>   <dbl>
1 1st                39.2   14.5      80
2 2nd                29.5   13.6      70
3 3rd                24.8   12.0      74
```

Histograms

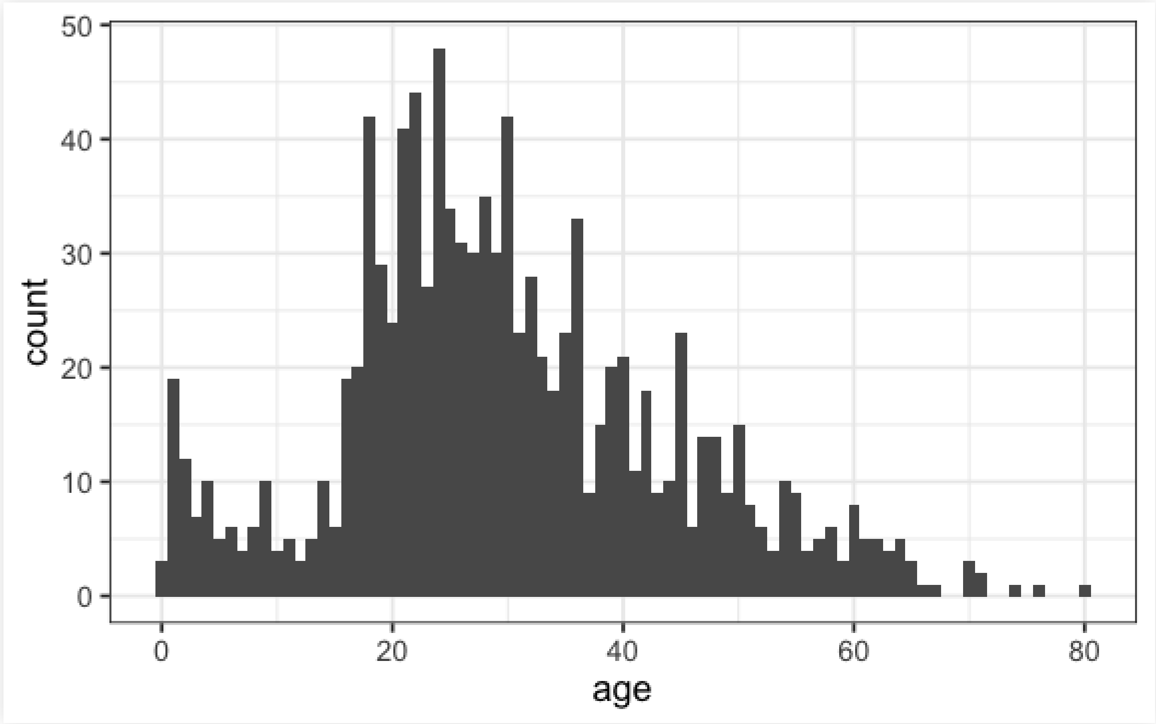
Now let's say we wanted to look at the *full* distribution of ages on the Titanic (i.e. not just a summary like the average).



Our workhorse for this kind of thing is a histogram.

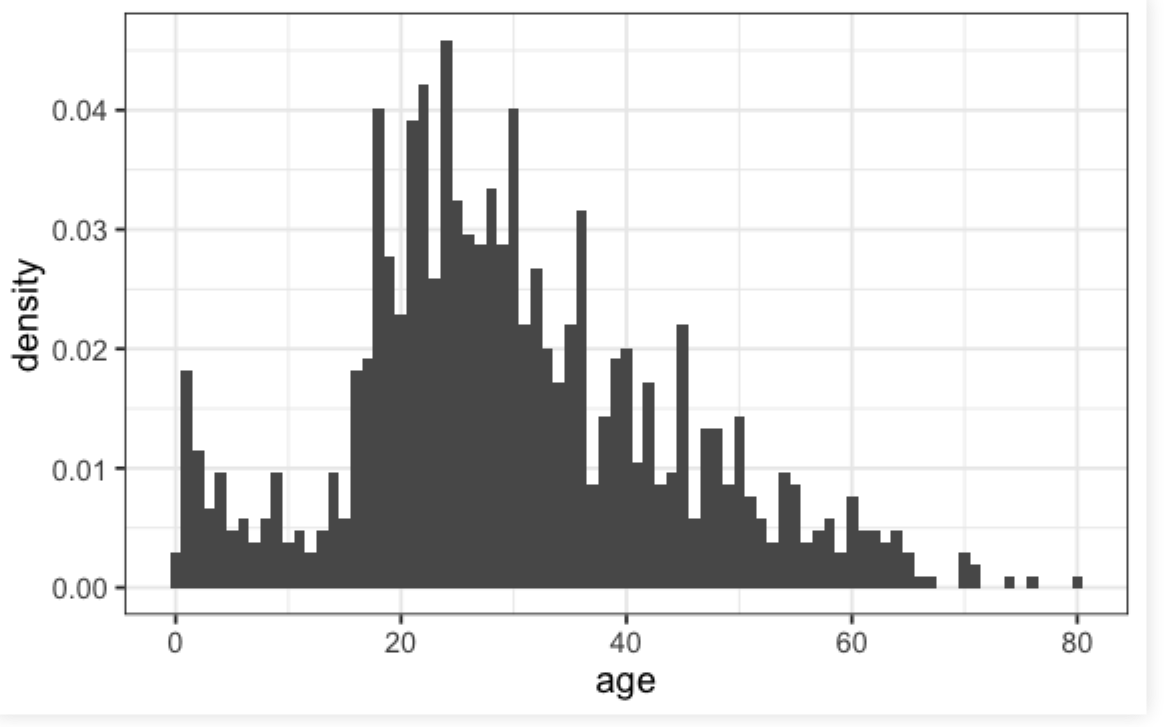
Histograms

We can change the bin width on a histogram (here 1, versus 5):



Histograms

We can also normalize the total area to sum to 1:

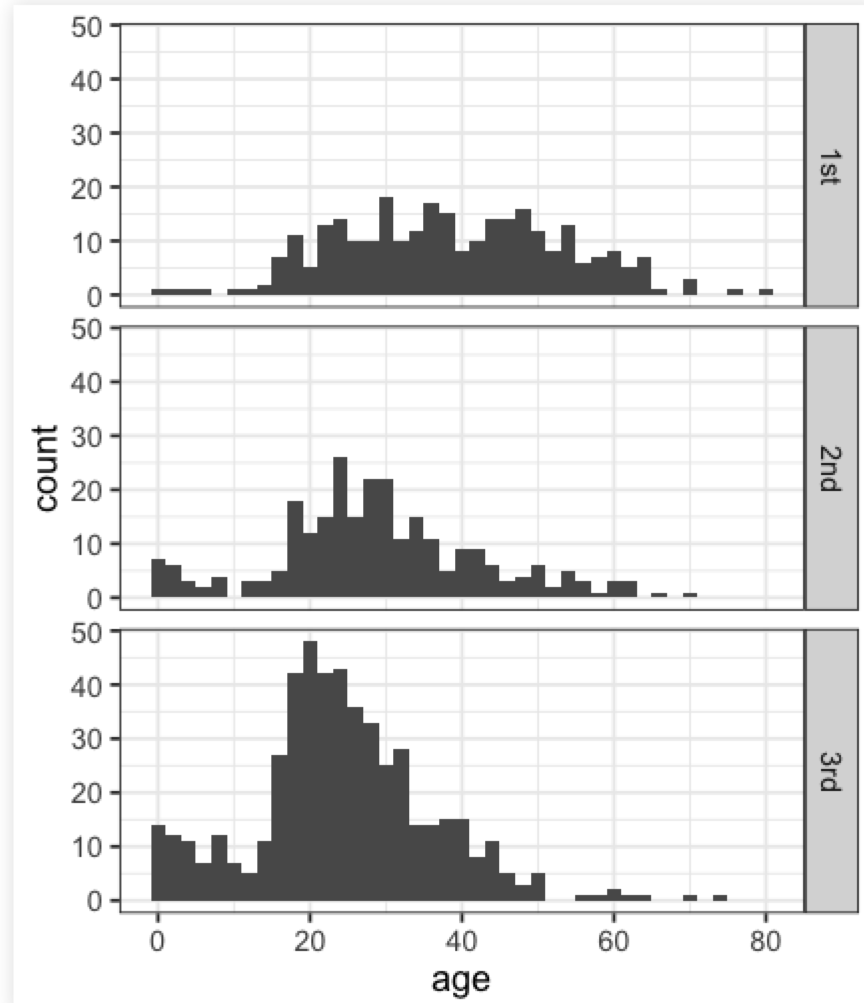


This is called a density histogram. It's like an estimated probability density.

Histograms

We can also compare histograms across different levels of a categorical variable (recall this is *faceting*).

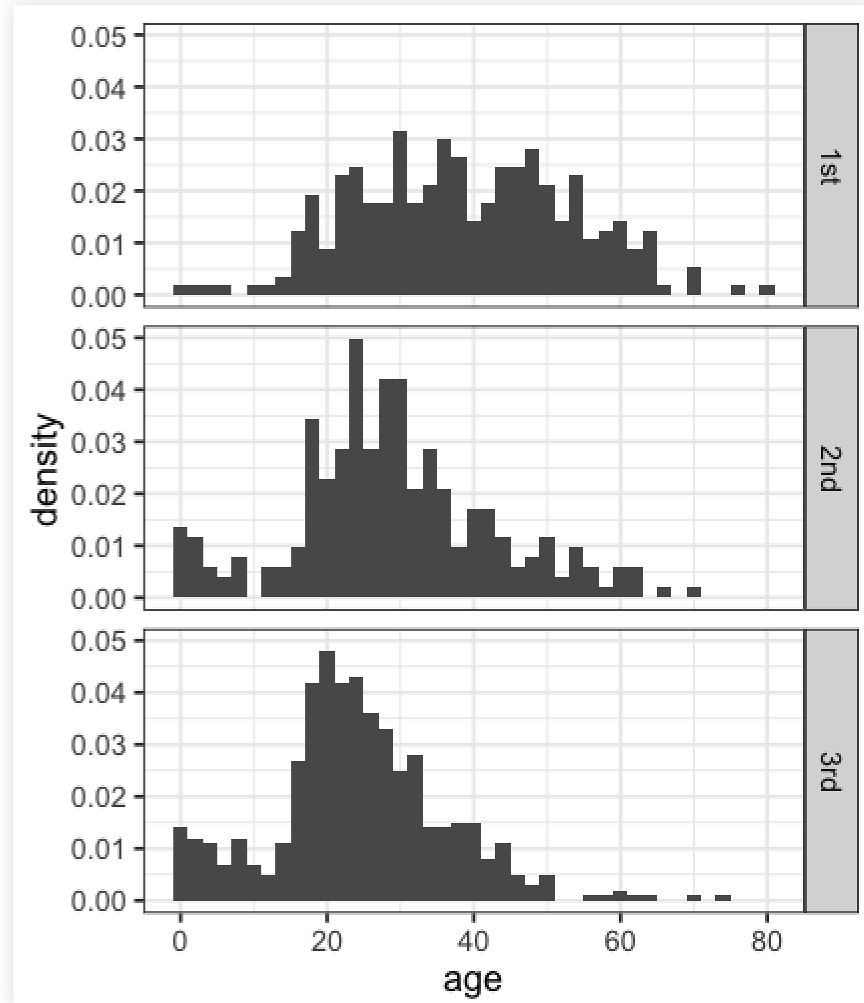
With raw counts, each histogram has a different total area.



Histograms

In density form.

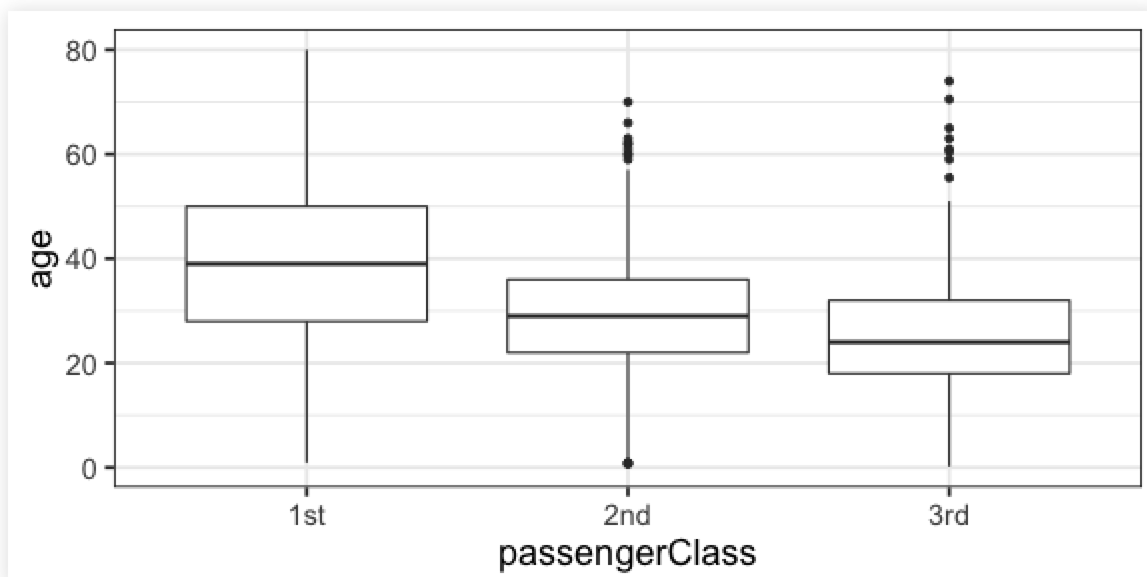
Notice that now, each panel has total area 1.



Boxplots

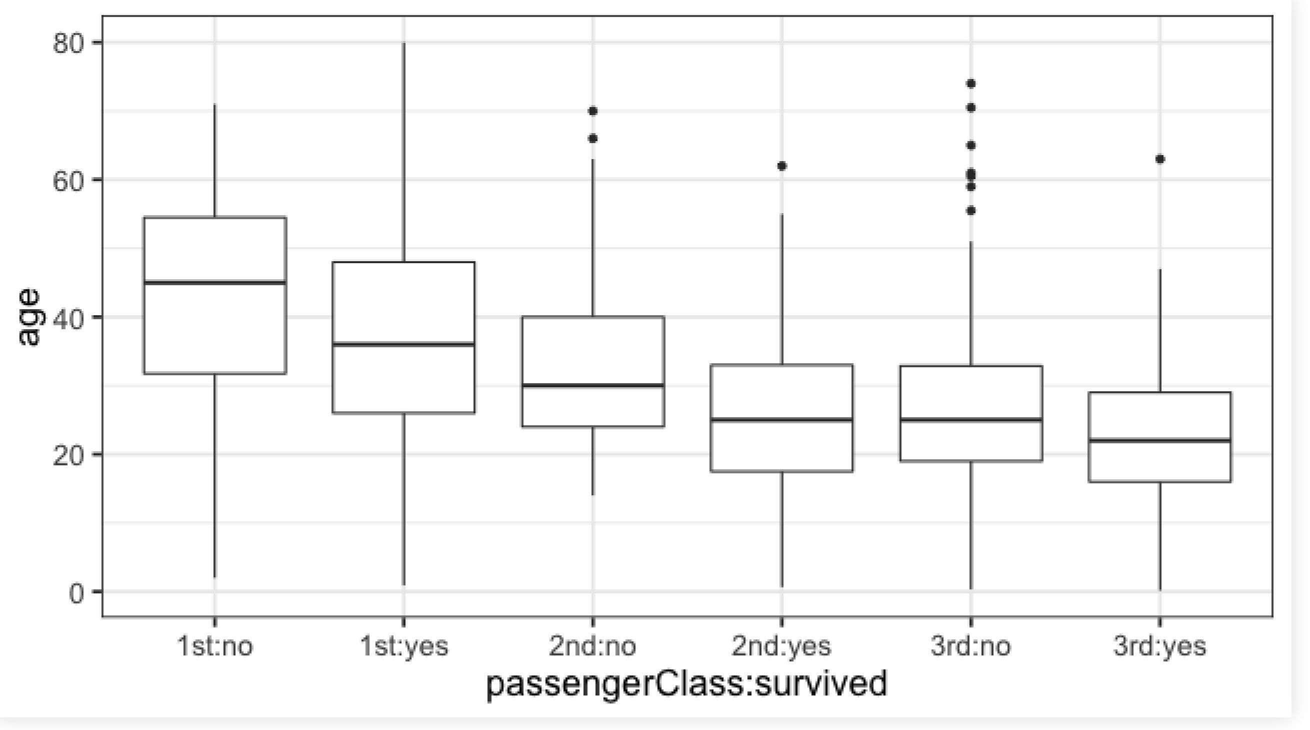
Another way to compare data across categories is with a boxplot.

- Each box shows the median and first/third quartiles.
- By default, the whiskers extend 1.5 times the inter-quartile range. Points outside these are shown individually.



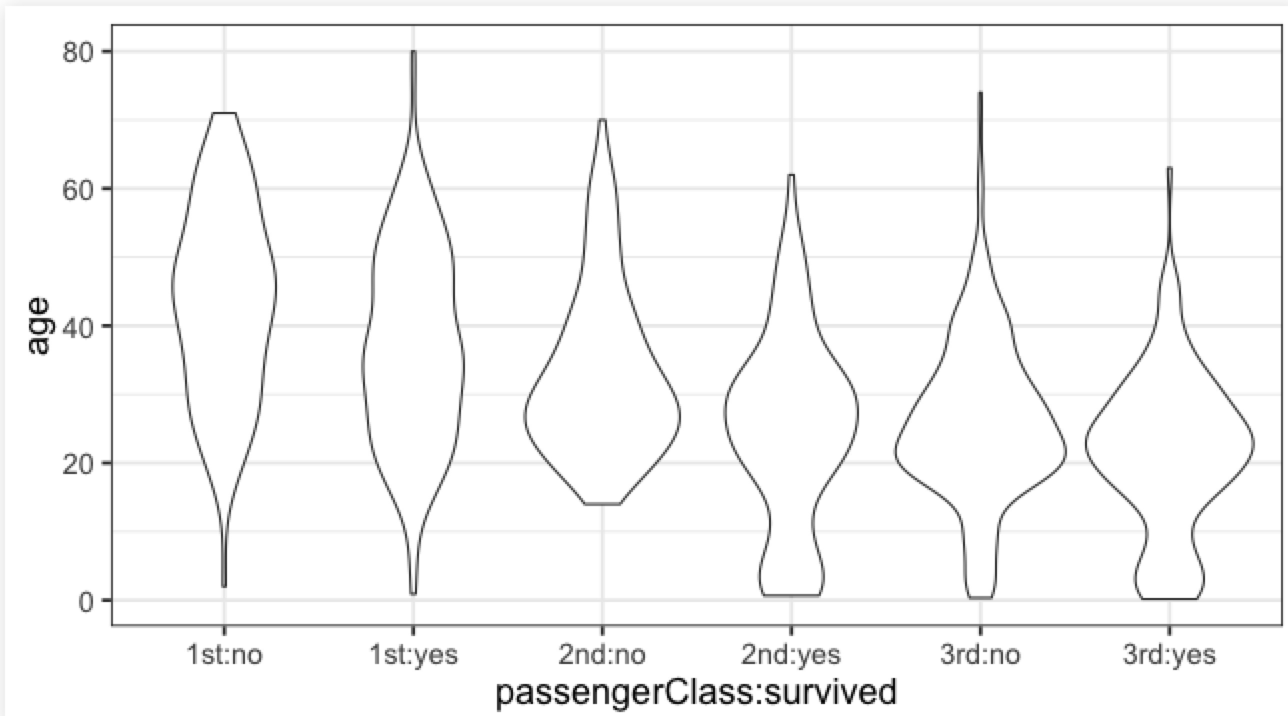
Boxplots

Boxplots are preferred when there are lots of categories, because individual histograms can look cluttered.



Violin plots

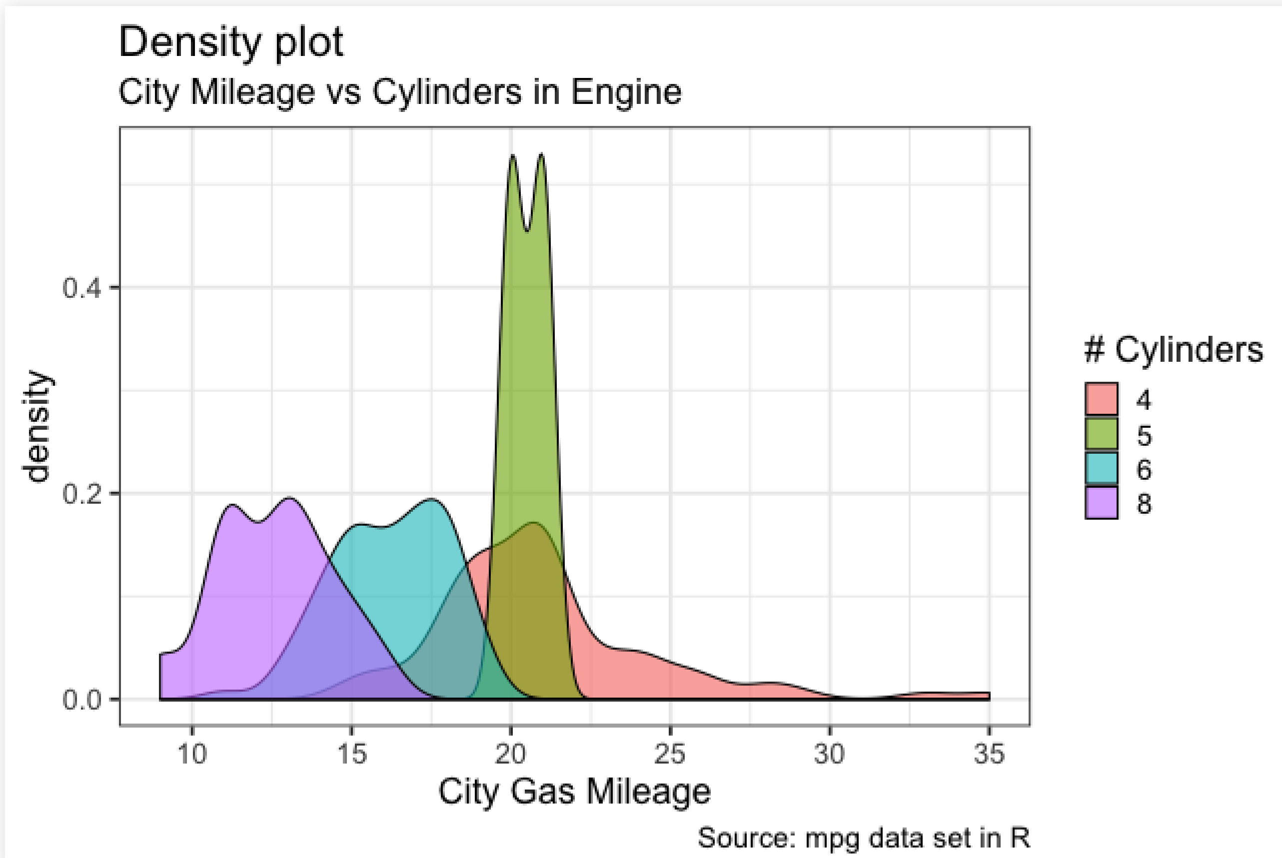
A violin plot is a variant; it attempts to show a bit more of the shape of each distribution.



The width of the violin is kind of like the height of the histogram.

Density plots

Another variant is the density plot, which is like a smooth version of a histogram:



Take-home skills

We've covered:

- data types (categorica/ordinal/numerical)
- cross tabulation and contingency tables
- some basic plots (bar charts, scatter plots, line graphs, histograms and their variations)
- basics of data workflow (pipe/group/summarize)

To the code!

Let's look at the code examples in:

- `mpg.R`
- `titanic.R`
- `toyimports_linegraph.R`