Stat405

Displaying distributions

Hadley Wickham

1. The diamonds data

2. Histograms and bar charts

3. Scatterplots for big data

Diamonds

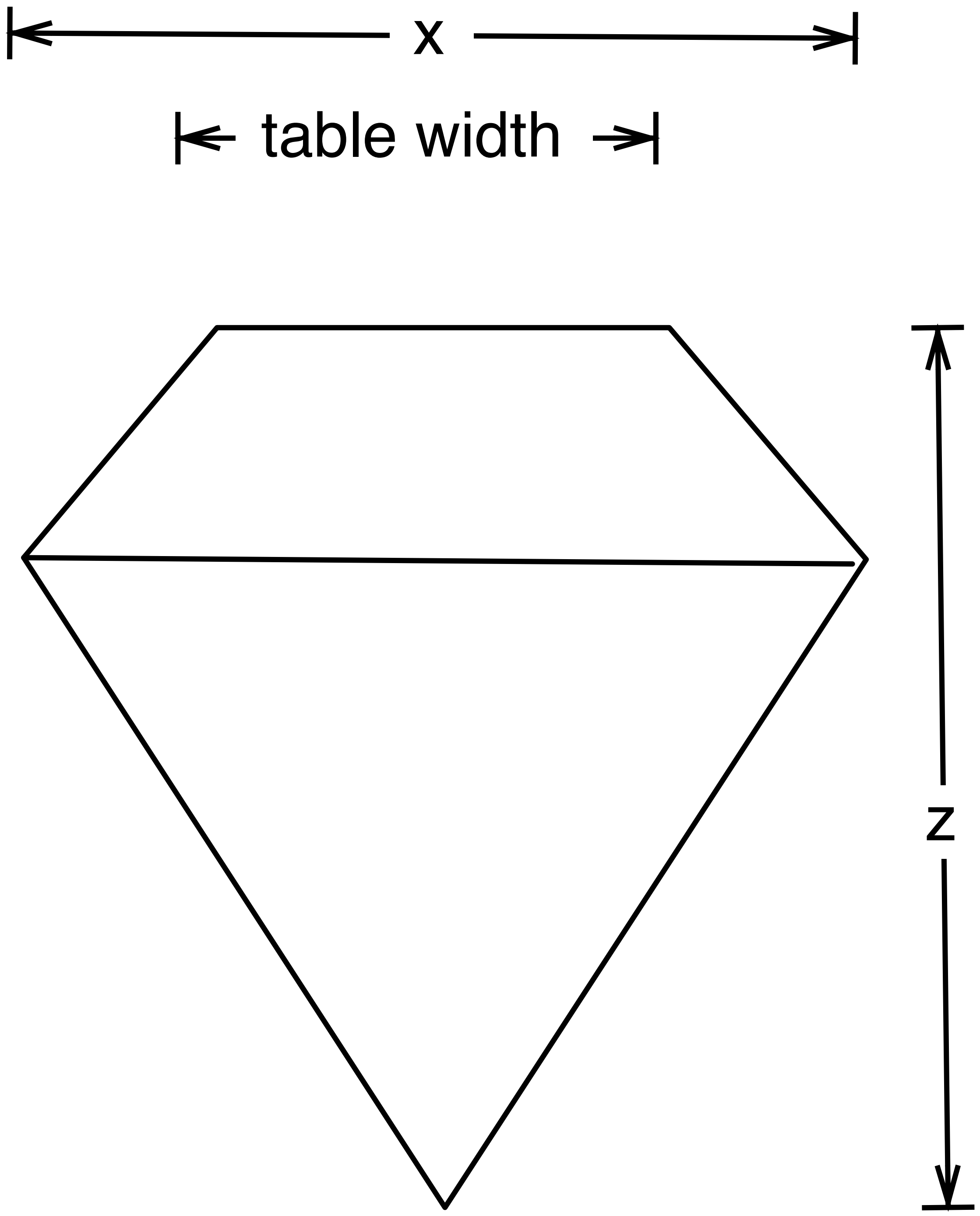
Diamonds data

~**54,000** round diamonds from http://www.diamondse.info/

Carat, colour, clarity, cut

Total depth, table, depth, width, height

Price



depth = z / diameter
table = table width / x * 100

Recall

Write down five ways to inspect the diamonds dataset.

You have one minute!

Histogram & bar charts

Histograms and barcharts

Used to display the **distribution** of a variable

Categorical variable → bar chart

Continuous variable → histogram

```
# With only one variable, qplot guesses that
# you want a bar chart or histogram
qplot(cut, data = diamonds)

qplot(carat, data = diamonds)

# Change binwidth:
qplot(carat, data = diamonds, binwidth = 1)
qplot(carat, data = diamonds, binwidth = 0.1)
qplot(carat, data = diamonds, binwidth = 0.01)
resolution(diamonds$carat)

last_plot() + xlim(0, 3)
```

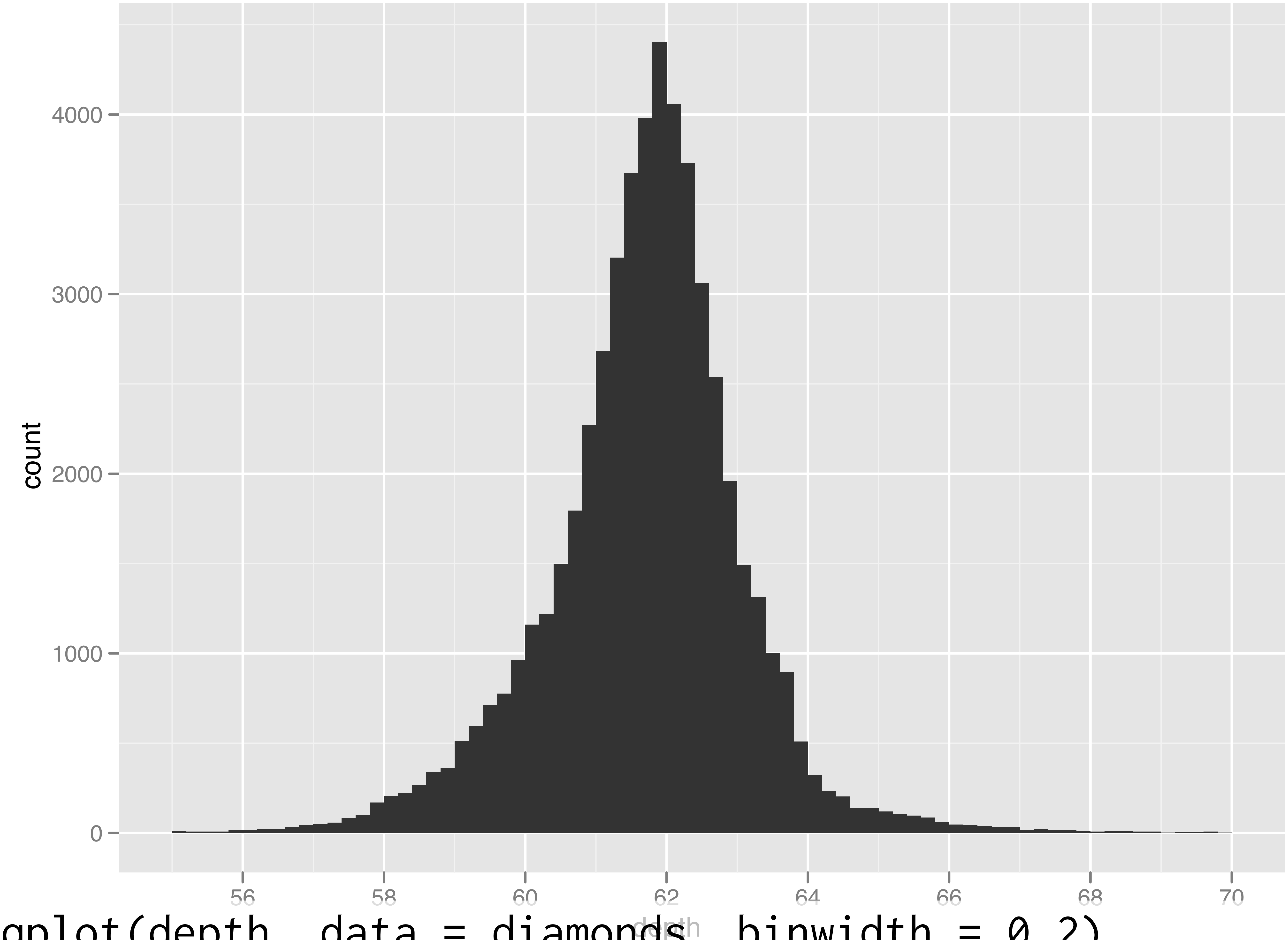
Always experiment with the bin width!

```
qplot(table, data = diamonds, binwidth = 1)

# To zoom in on a plot region use xlim() and ylim()
qplot(table, data = diamonds, binwidth = 1) +
   xlim(50, 70)
qplot(table, data = diamonds, binwidth = 0.1) +
  xlim(50, 70)
qplot(table, data = diamonds, binwidth = 0.1) +
  xlim(50, 70) + ylim(0, 50)

# Note that this type of zooming discards data
# outside of the plot regions. See
# ?coord_cartesian() for an alternative
```

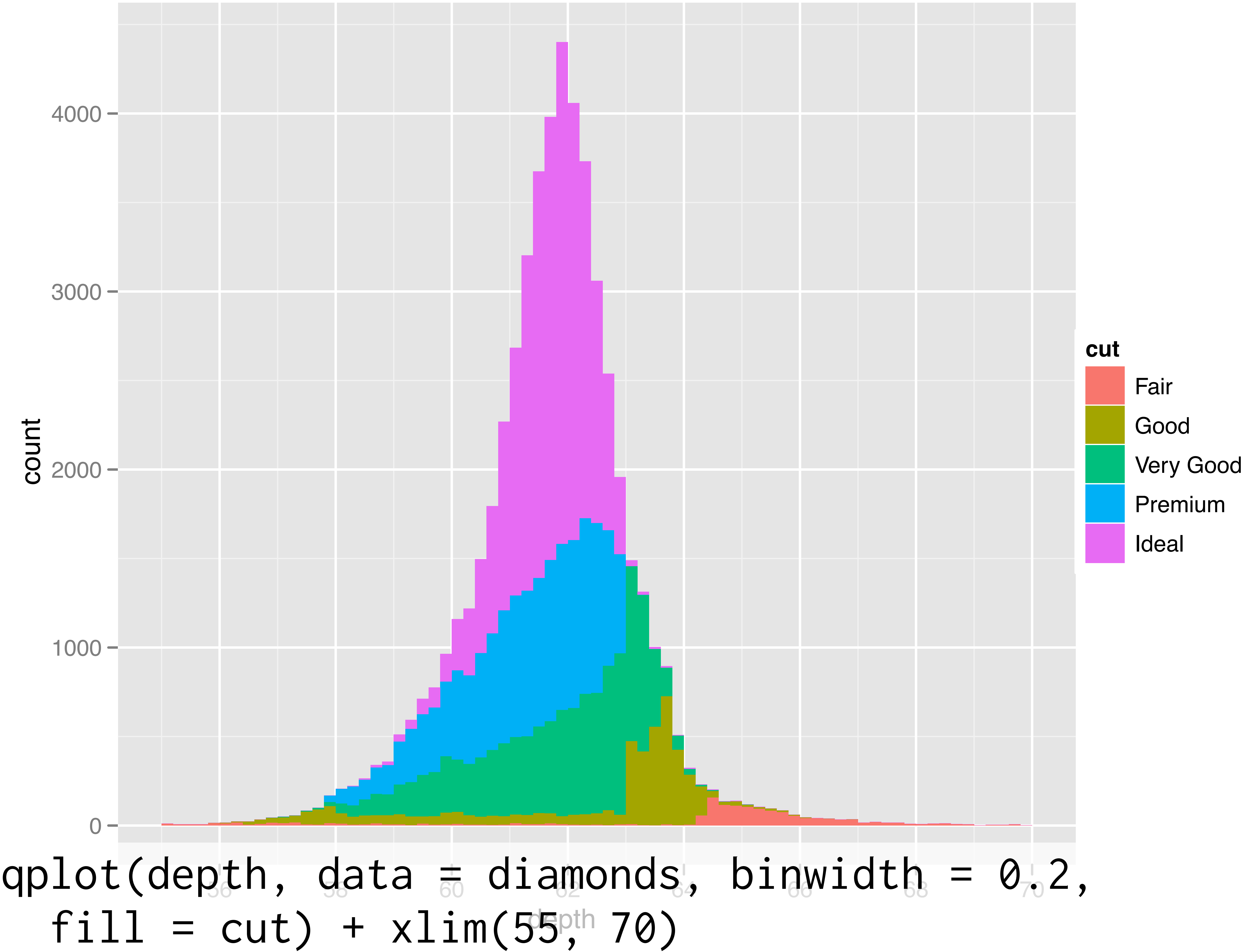
Additional variables

As with scatterplots can use **aesthetics** or **faceting**. Using aesthetics creates pretty, but ineffective, plots.

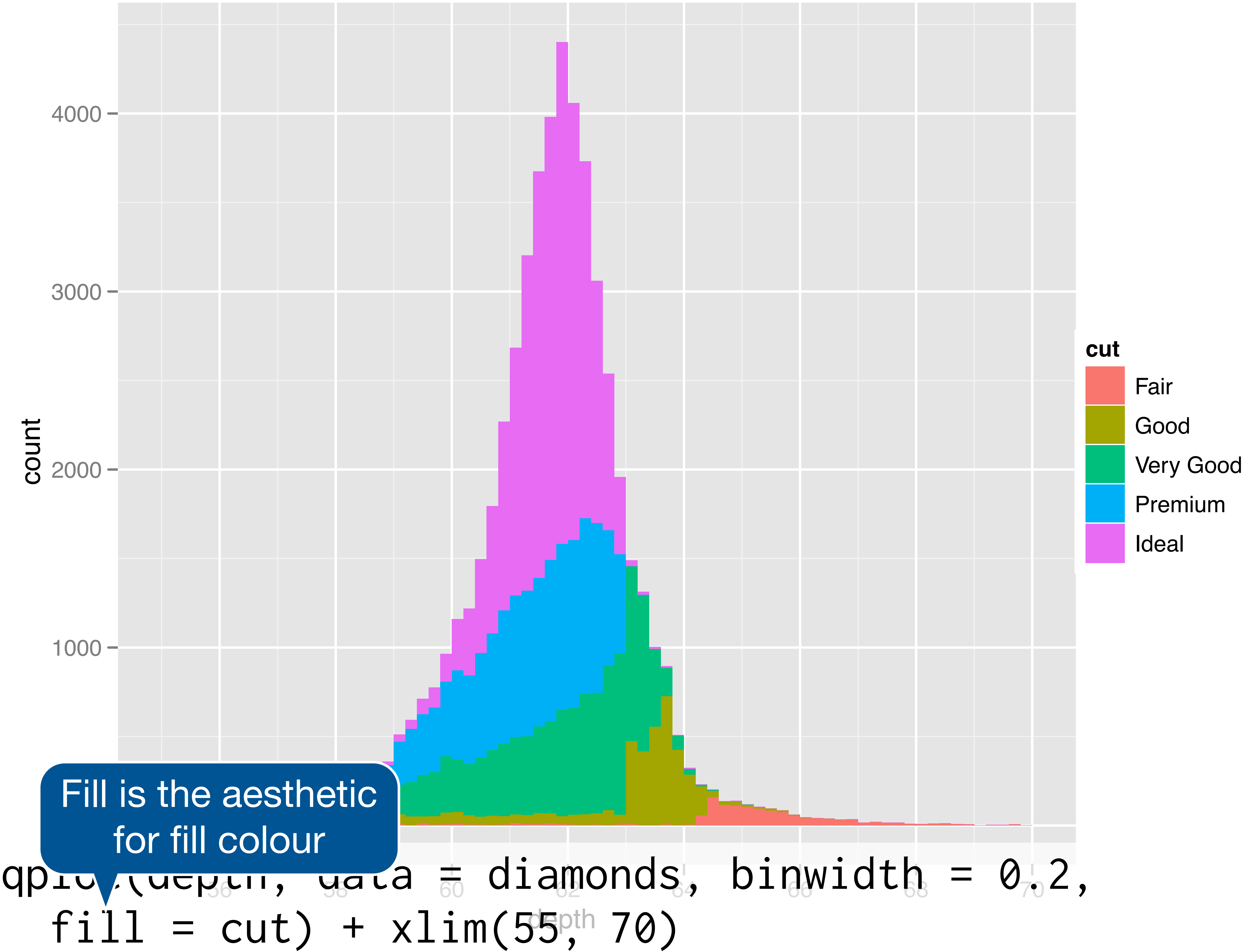
The following examples show the difference, when investigation the relationship between cut and depth.



```
qplot(depth, data = diamonds, binwidth = 0.2)
```

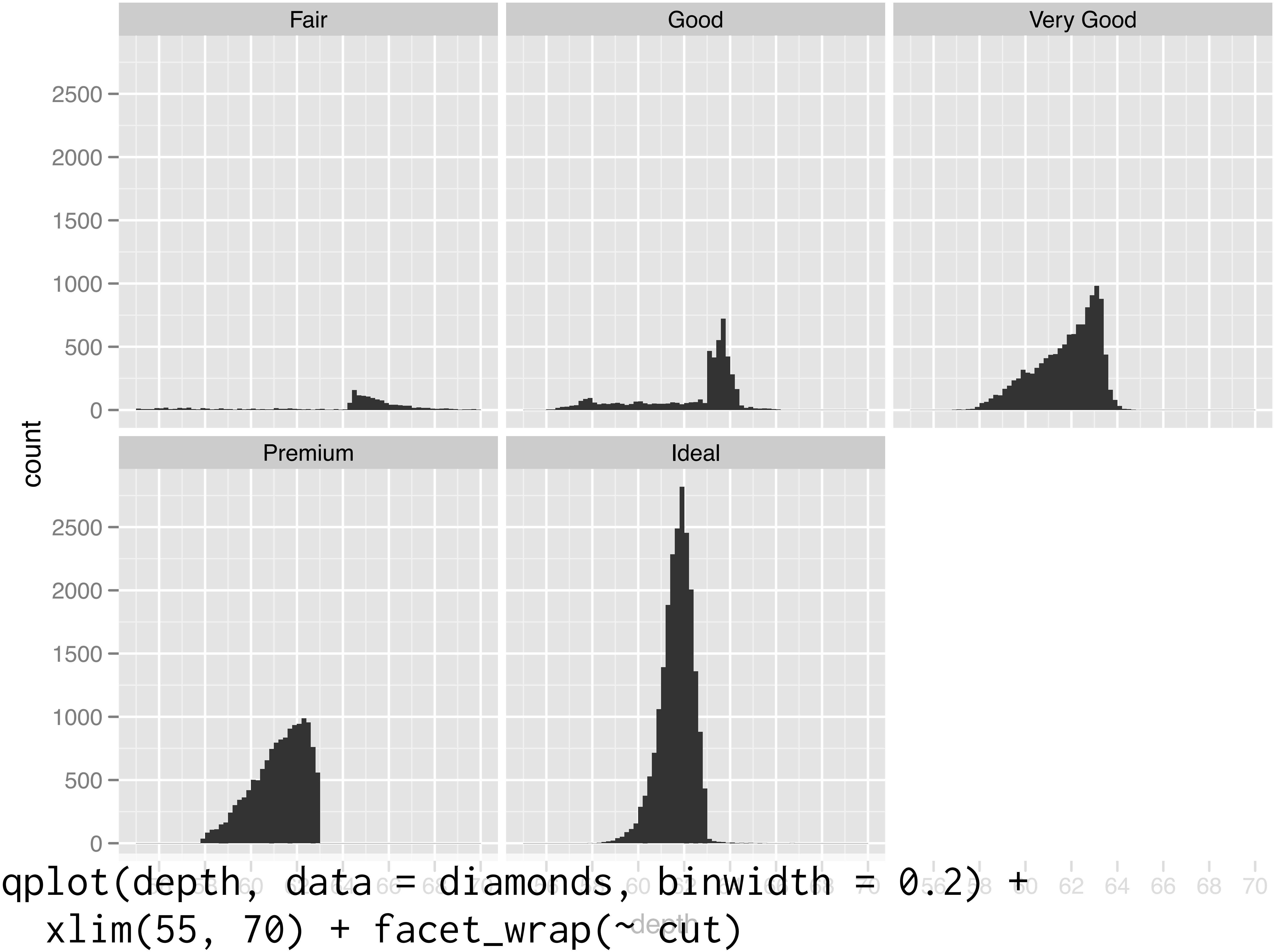


```
qplot(depth, data = diamonds, binwidth = 0.2,
  fill = cut) + xlim(55, 70)
```



Fill is the aesthetic for fill colour

```
qplot(depth, data = diamonds, binwidth = 0.2,
  fill = cut) + xlim(55, 70)
```

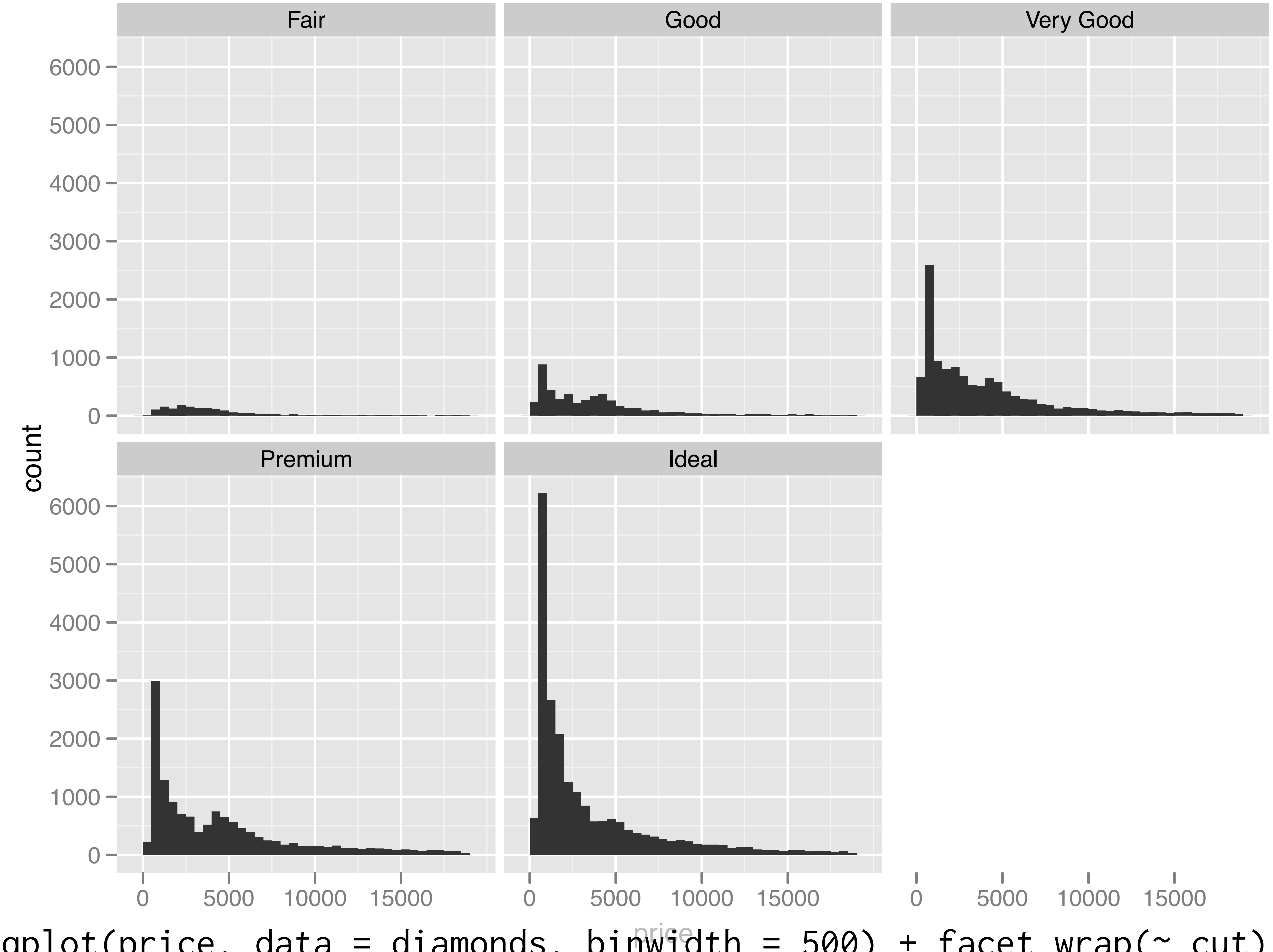


```
qplot(depth, data = diamonds, binwidth = 0.2) +
  xlim(55, 70) + facet_wrap(~ cut)
```

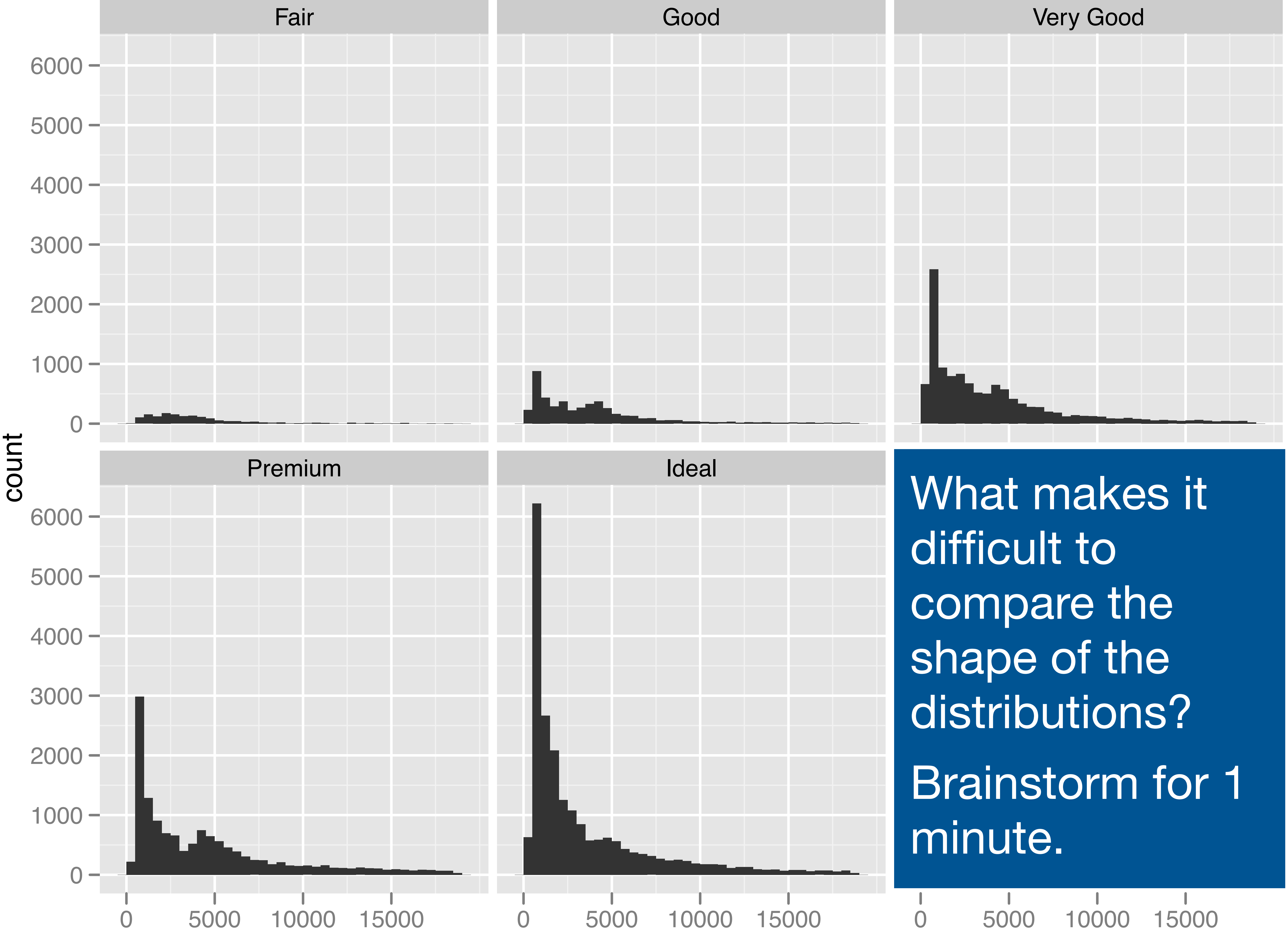
Your turn

Explore the distribution of price. What is a good binwidth to use? (**Hint**: How many bins will a binwidth of 1 give you?) Practice zooming in on regions of interest.

How does price vary with colour, cut, or clarity?



```
qplot(price, data = diamonds, binwidth = 500) + facet_wrap(~ cut)
```



What makes it difficult to compare the shape of the distributions?

Brainstorm for 1 minute.

```
qplot(price, data = diamonds, binwidth = 500) + facet_wrap(~ cut)
```

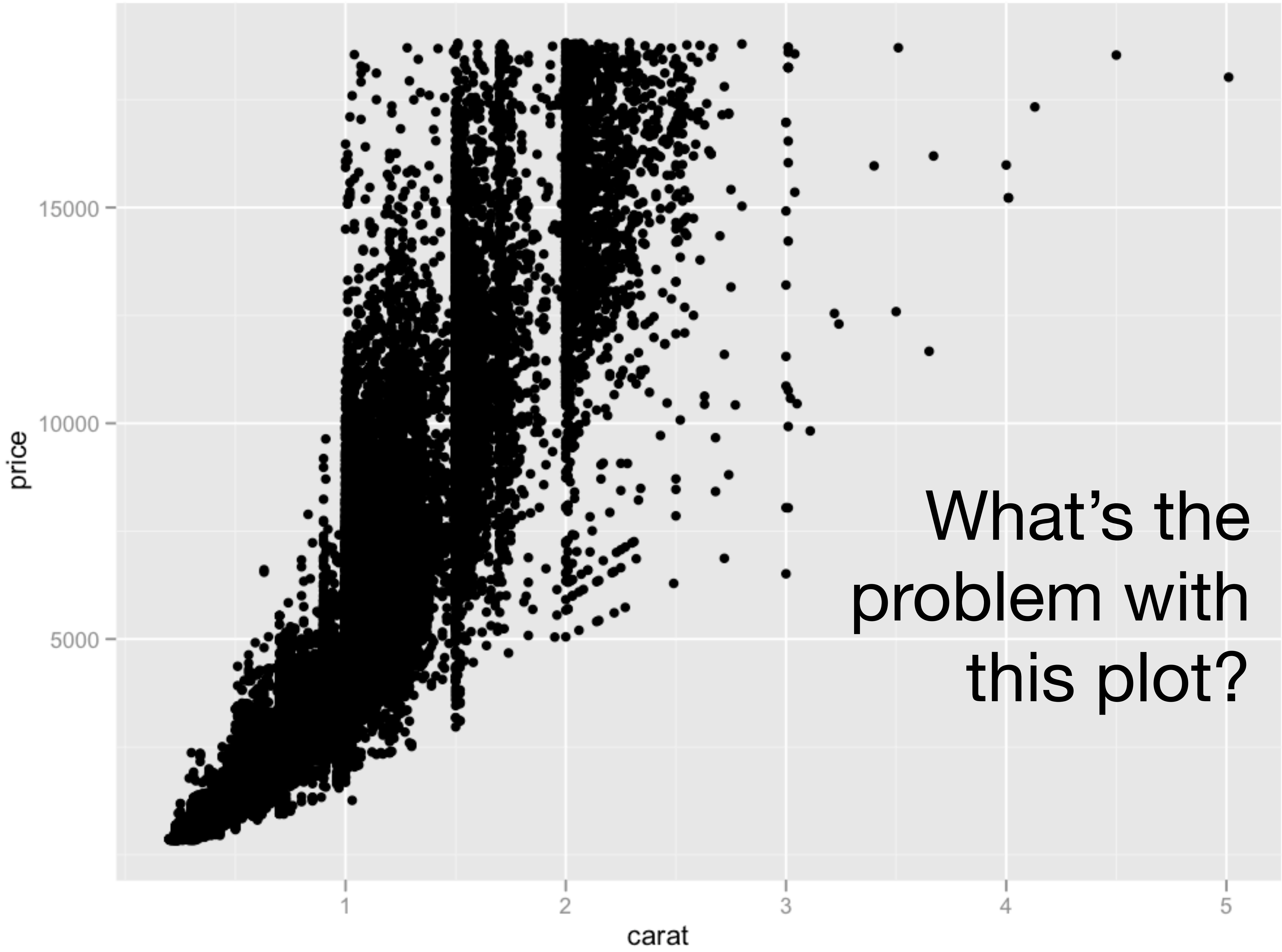
Problems

Each histogram far away from the others, but we know stacking is hard to read → *use another way of displaying densities*

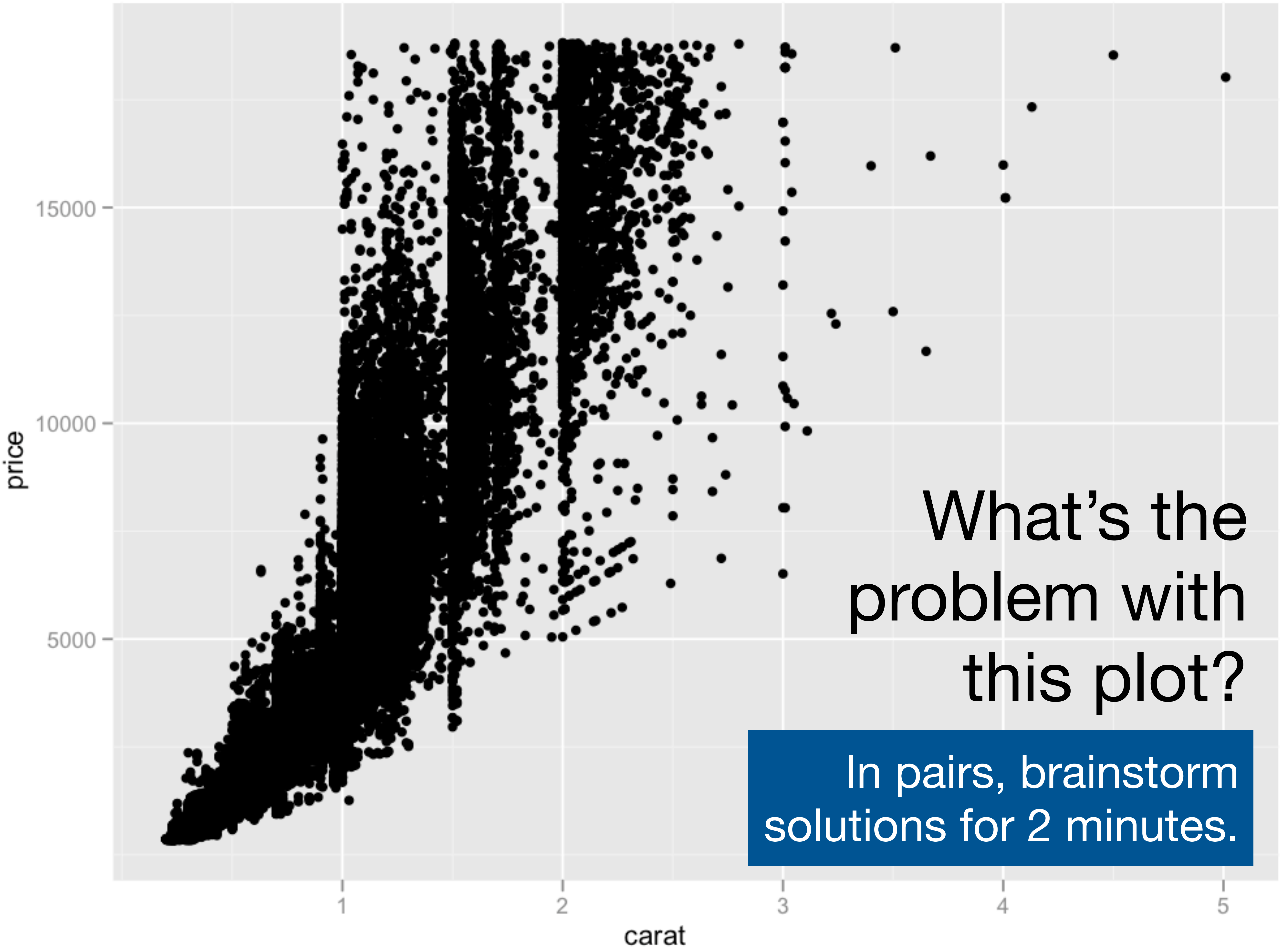
Varying relative abundance makes comparisons difficult → *rescale to ensure constant area*

```
# Large distances make comparisons hard
qplot(price, data = diamonds, binwidth = 500) +
  facet_wrap(~ cut)

# Stacked heights hard to compare
qplot(price, data = diamonds, binwidth = 500, fill = cut)

# Much better - but still have differing relative abundance
qplot(price, data = diamonds, binwidth = 500,
  geom = ”freqpoly”, colour = cut)

# Instead of displaying count on y-axis, display density
# .. indicates that variable isn’t in original data
qplot(price, ..density.., data = diamonds, binwidth = 500,
  geom = ”freqpoly”, colour = cut)

# To use with histogram, you need to be explicit
qplot(price, ..density.., data = diamonds, binwidth = 500,
  geom = ”histogram”) + facet_wrap(~ cut)
```

Scatterplots for big data



What’s the problem with this plot?



What’s the problem with this plot?

In pairs, brainstorm solutions for 2 minutes.

Idea	ggplot
Small points	`shape = I(”.”)`
Transparency	`alpha = I(1/50)`
Jittering	`geom = ”jitter”`
Smooth curve	`geom = ”smooth”`

2d bins	`geom = ”bin2d” or`
	`geom = ”hex”`

Density contours	`geom = ”density2d”`

Boxplots	`geom = ”boxplot” +`
	`group = ...`

```
# There are two ways to add additional geoms
# 1) A vector of geom names:
qplot(price, carat, data = diamonds,
  geom = c(”point”, ”smooth”))

# 2) Add on extra geoms
qplot(price, carat, data = diamonds) + geom_smooth()

# This is how you get help about a specific geom:
# ?geom_smooth
# To set aesthetics to a particular value, you need
# to wrap that value in I()

qplot(price, carat, data = diamonds, colour = ”blue”)
qplot(price, carat, data = diamonds, colour = I(”blue”))

# Practical application: varying alpha
qplot(carat, price, data = diamonds, alpha = I(1/10))
qplot(carat, price, data = diamonds, alpha = I(1/50))
qplot(carat, price, data = diamonds, alpha = I(1/100))
qplot(carat, price, data = diamonds, alpha = I(1/250))
qplot(table, price, data = diamonds)
qplot(table, price, data = diamonds,
  geom = ”boxplot”)

# Need to specify grouping variable: what determines
# which observations go into each boxplot
qplot(table, price, data = diamonds,
  geom = ”boxplot”, group = round_any(table, 1))

qplot(table, price, data = diamonds,
  geom = ”boxplot”, group = round_any(table, 1)) +
  xlim(50, 70)
```

Your turn

Explore the relationship between carat, price and cut using these techniques.

(i.e. make this plot more informative:

`qplot(carat, price, data = diamonds, colour = cut)`)

Which did you find most useful?