Layer!

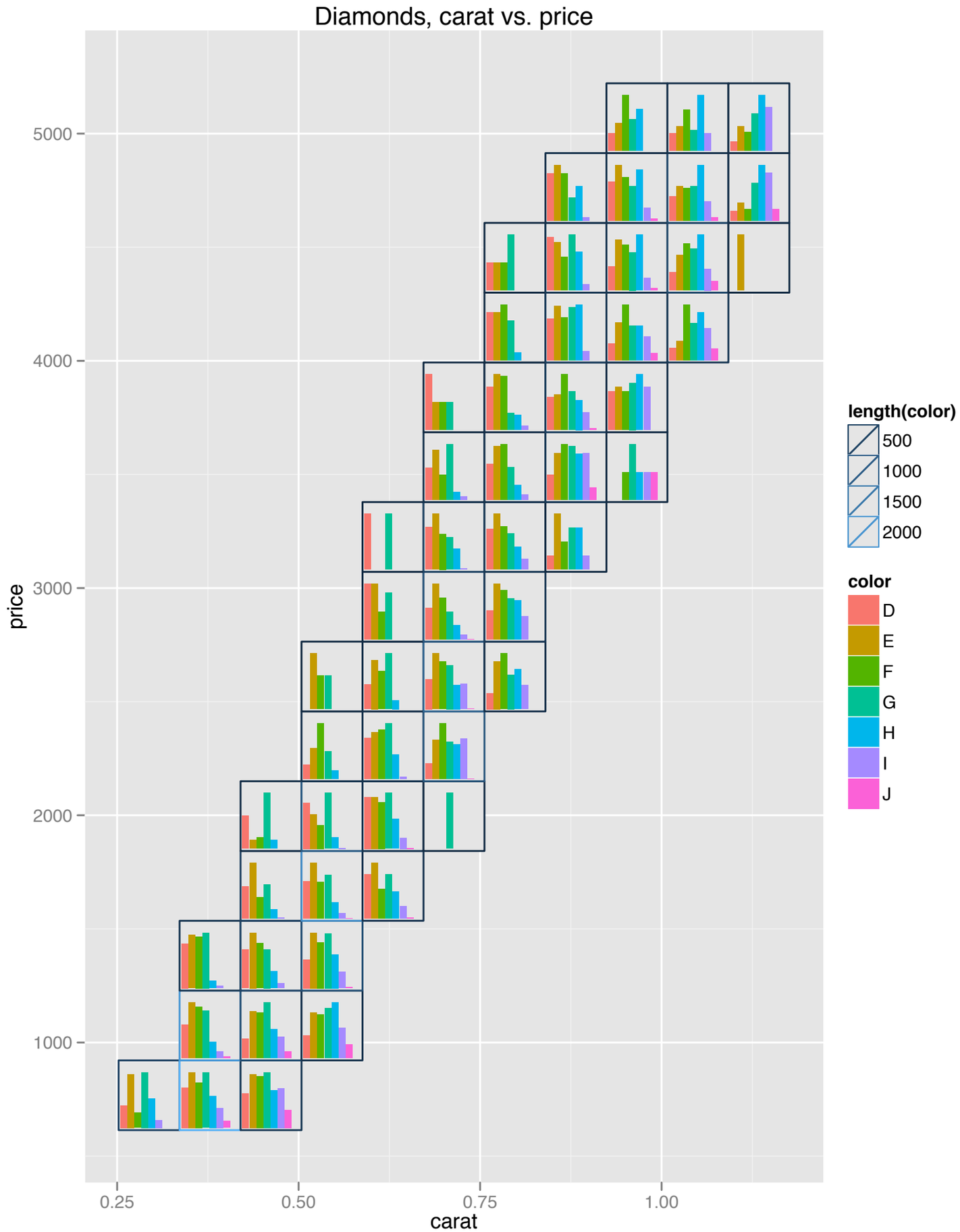
Hadley Wickham
@hadleywickham
Chief Scientist, RStudio

July 2015

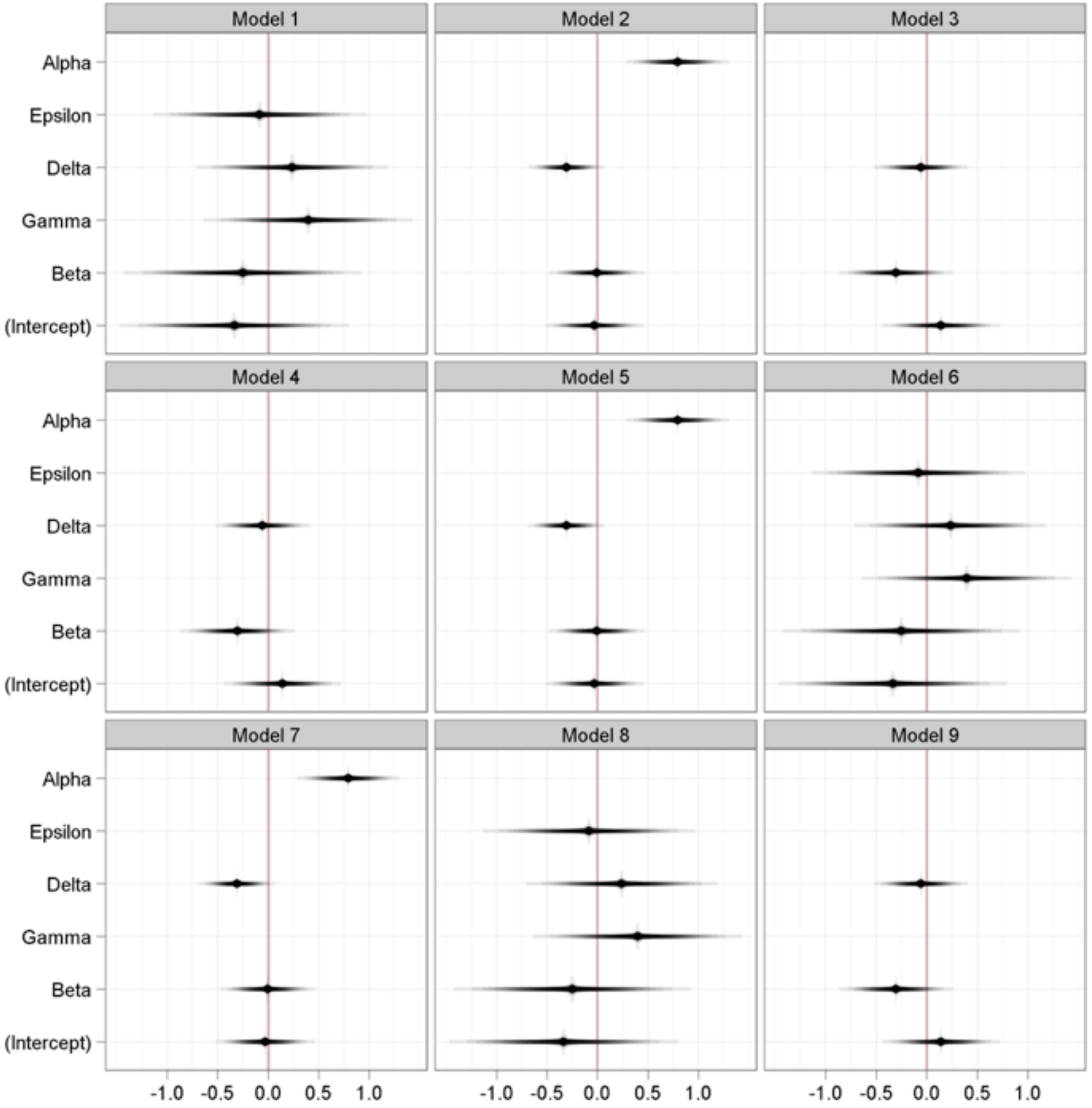
Outline

- Motivation
- Layer basics
- Big data warmups
- Useful techniques

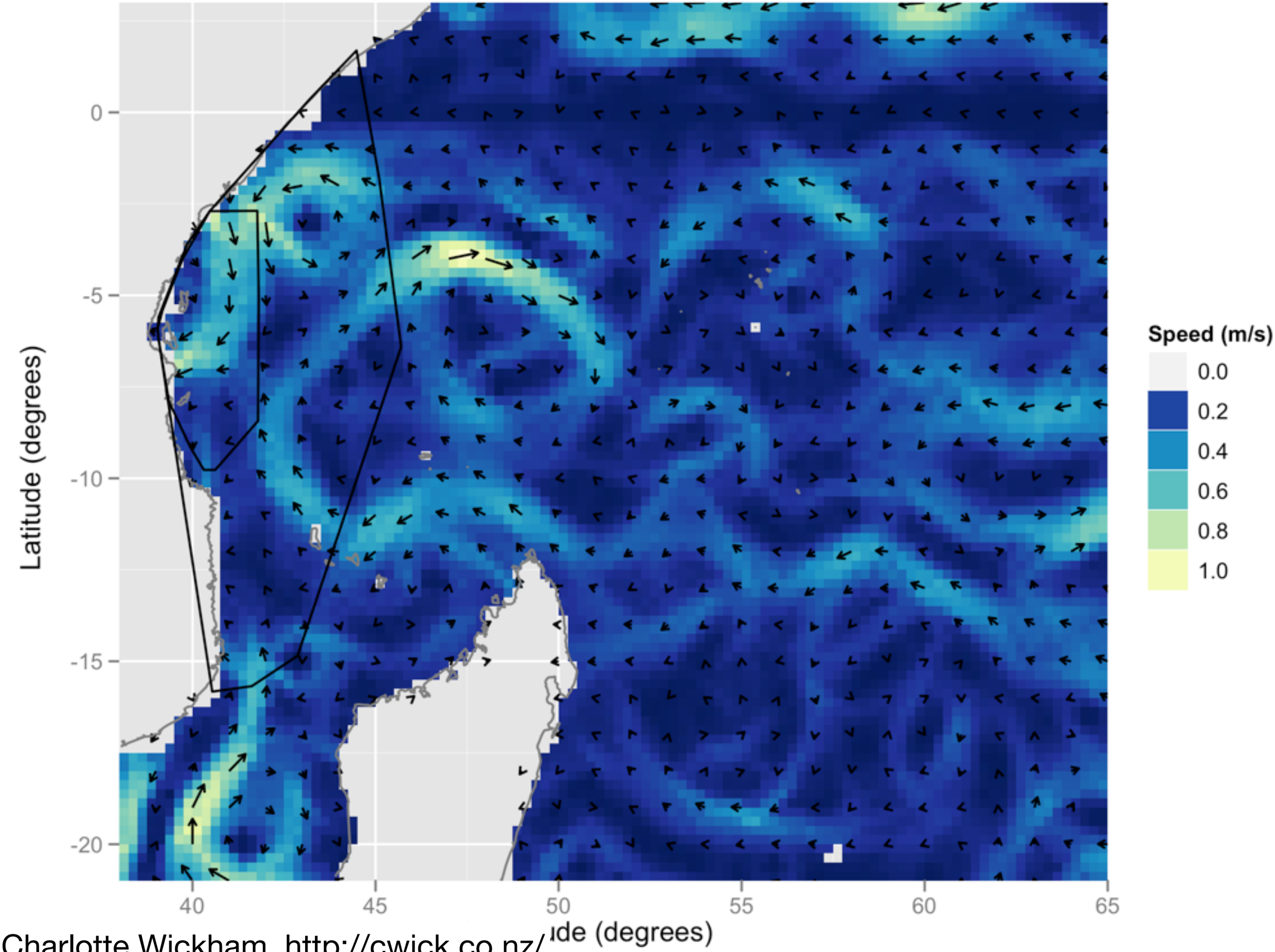
Motivation



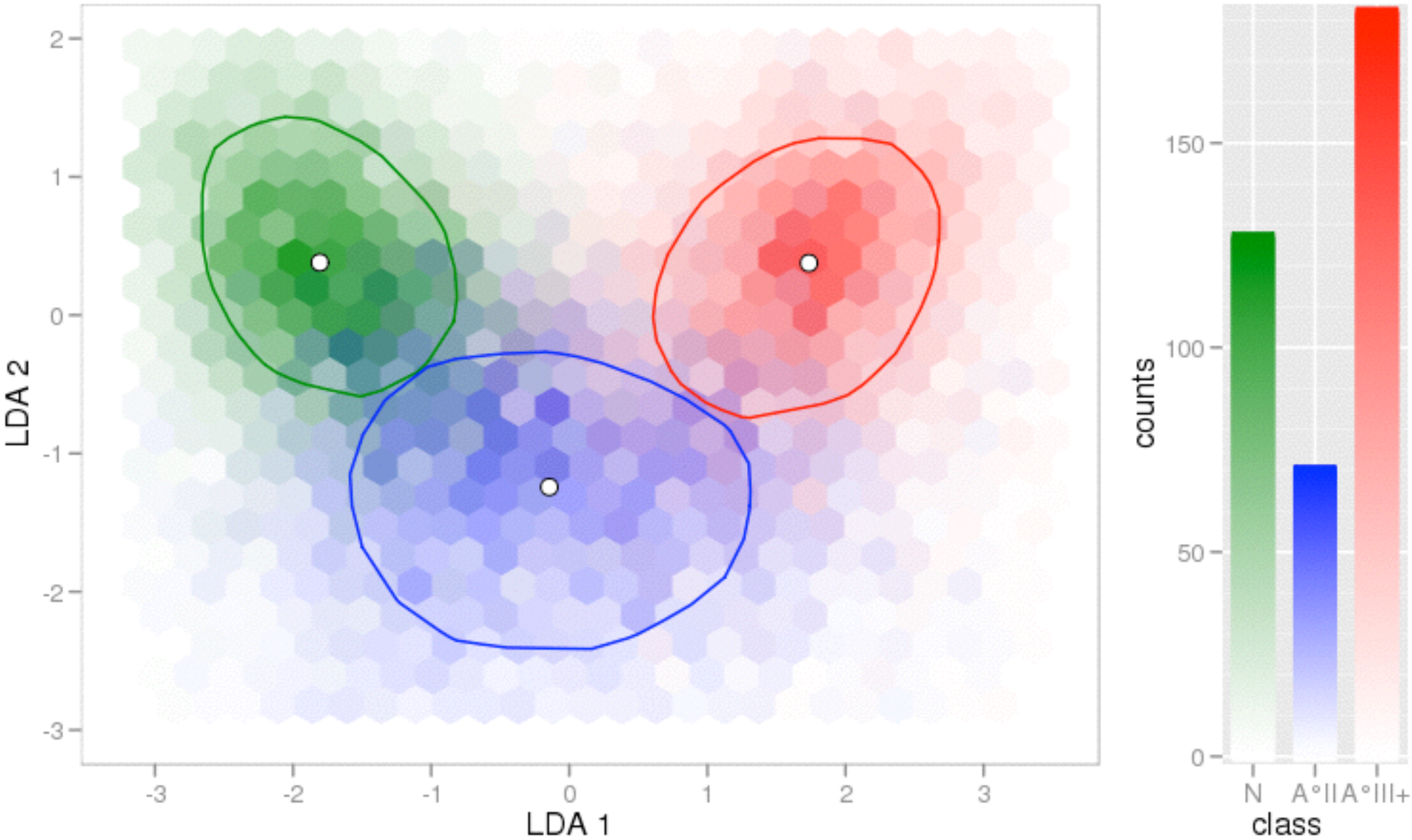
Garrett Grolemund, http://blog.revolutionanalytics.com/2012/09/visualize-complex-data-with-subplots.html



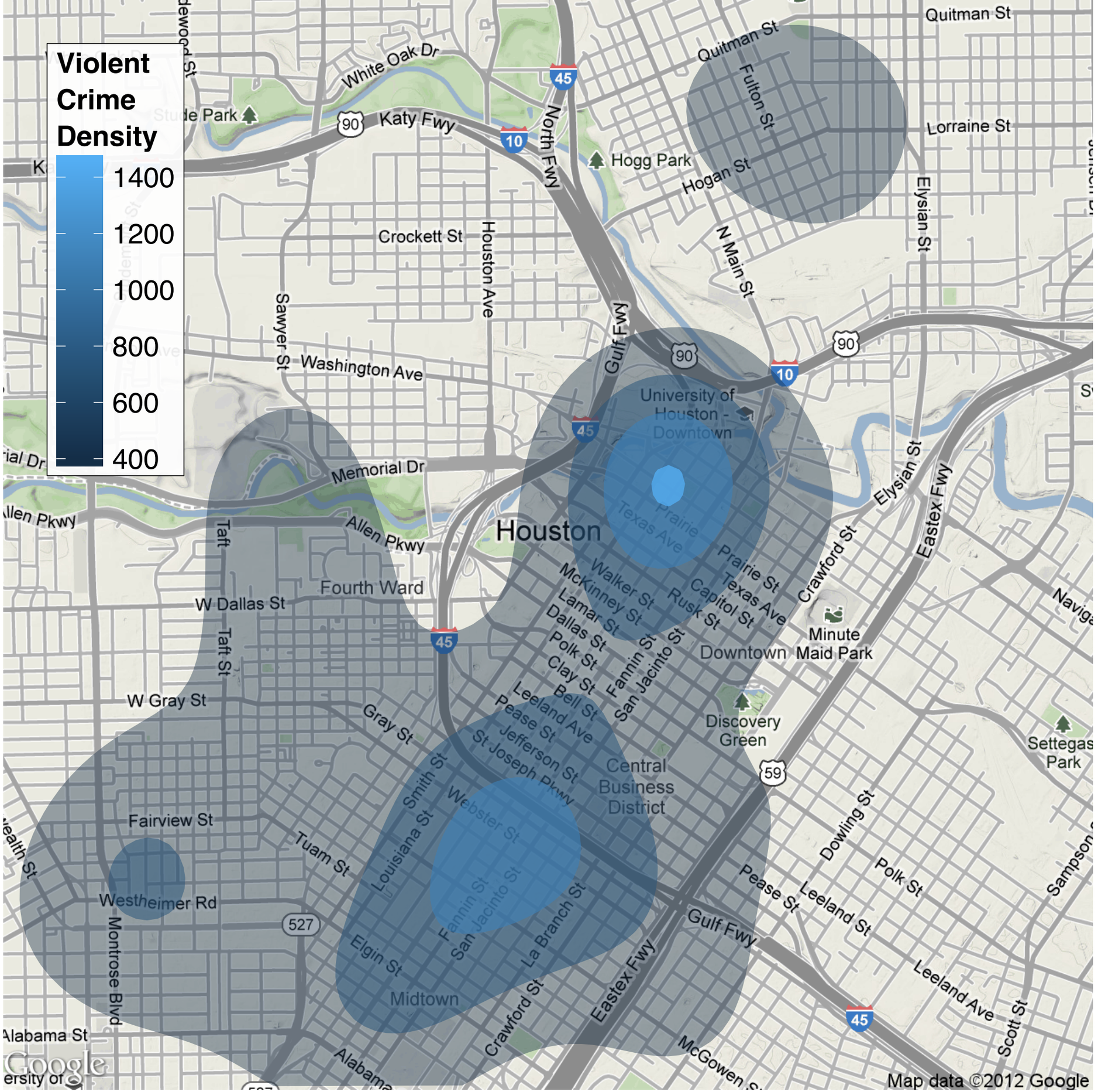
David B Sparks, http://bit.ly/hn54NW



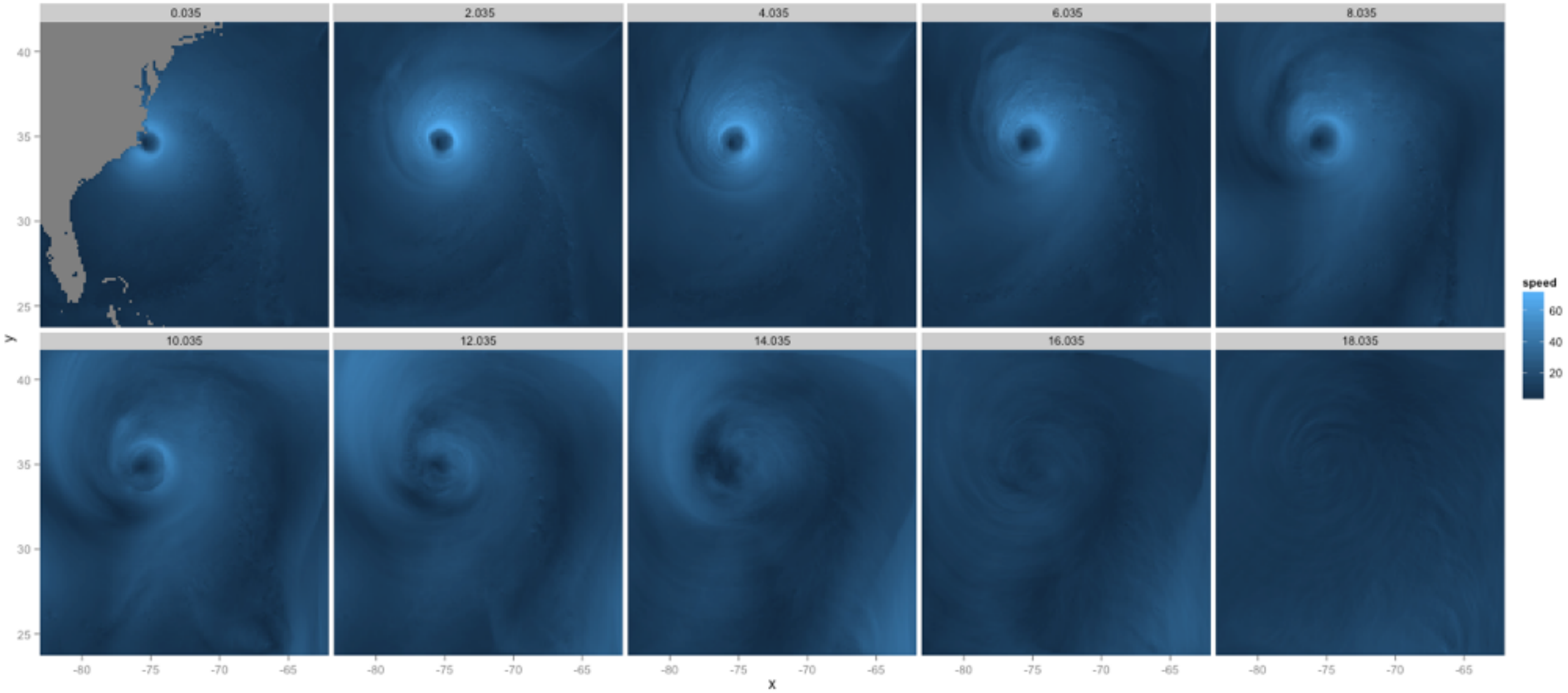
Charlotte Wickham, http://cwick.co.nz/



Claudia Beleites, http://bit.ly/yNqlpz



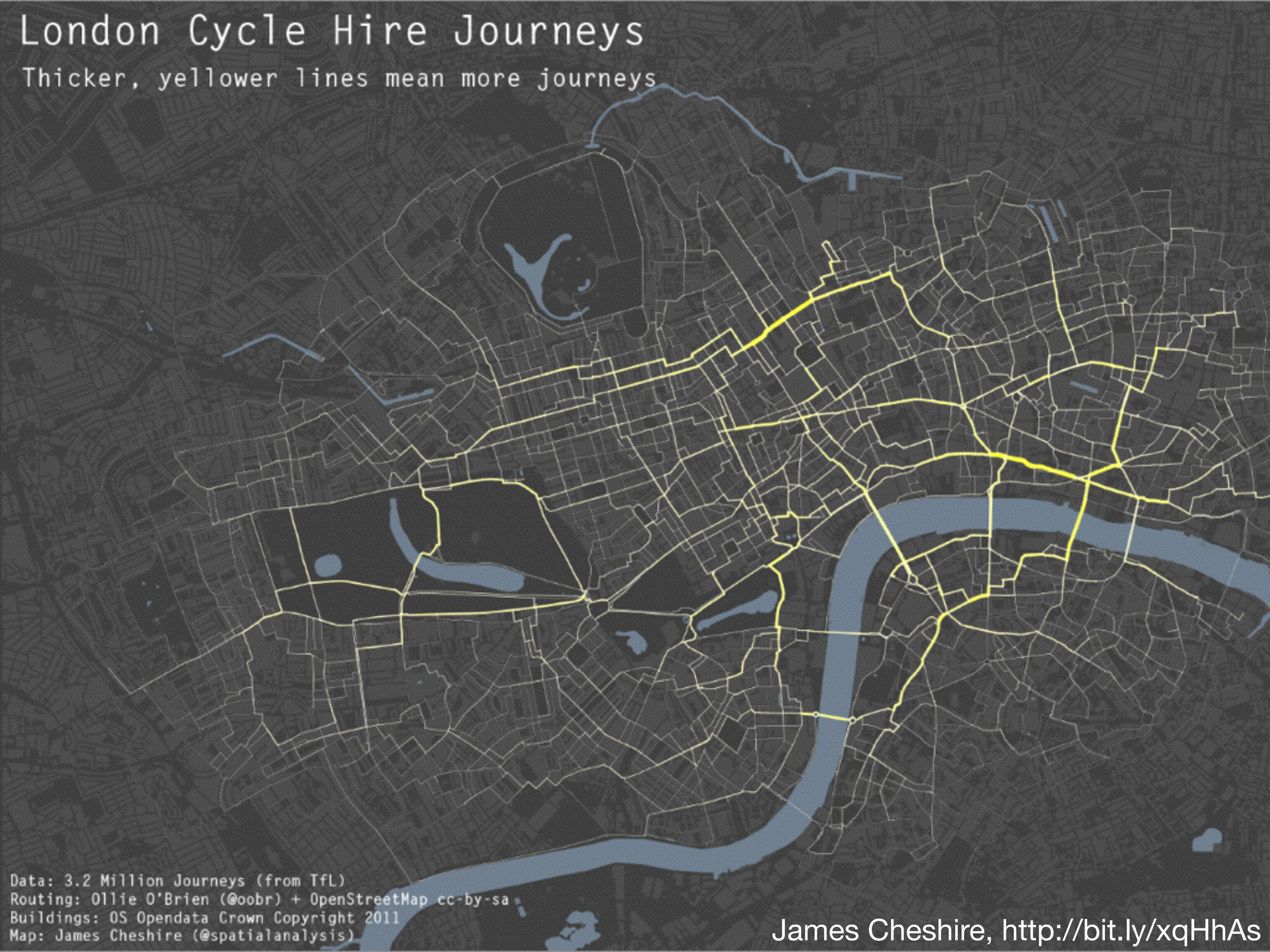
David Kahle, https://dl.dropbox.com/u/24648660/ggmap%20useR%202012.pdf



Winston Chang, http://shop.oreilly.com/product/0636920023135.do

London Cycle Hire Journeys

Thicker, yellower lines mean more journeys



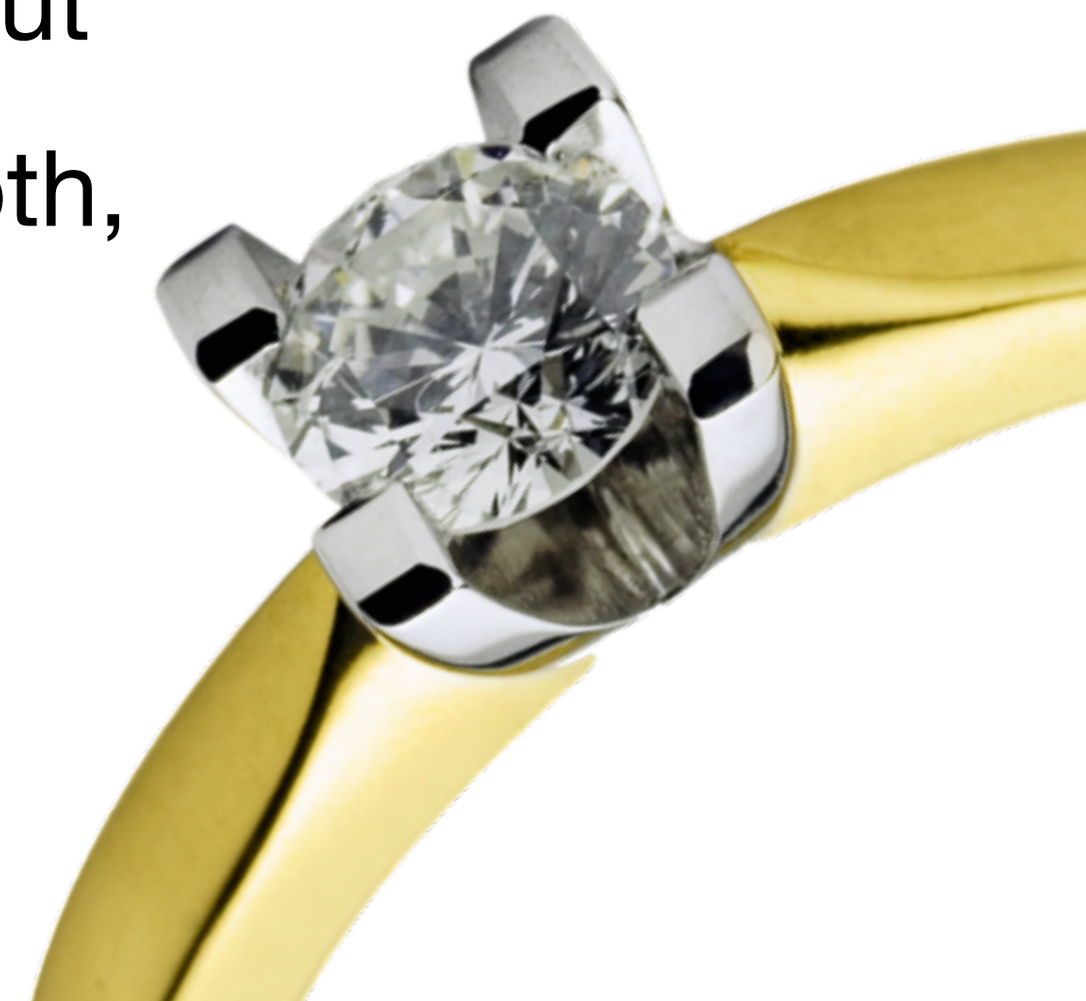
Data: 3.2 Million Journeys (from TfL)
Routing: Ollie O'Brien (@oobr) + OpenStreetMap cc-by-sa
Buildings: OS Opendata Crown Copyright 2011
Map: James Cheshire (@spatialanalysis)

James Cheshire, http://bit.ly/xqHhAs

Diamonds

Diamonds data

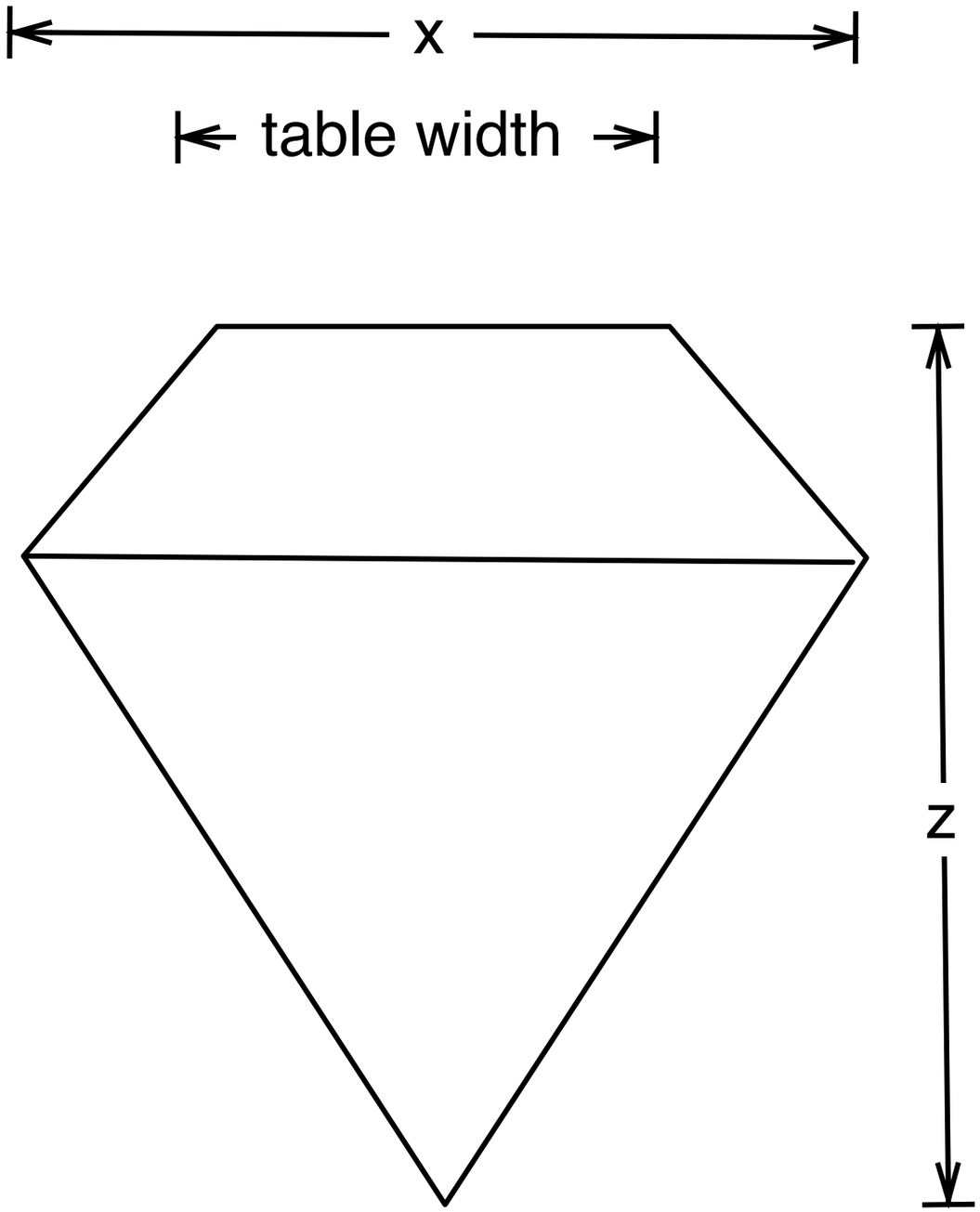
- ~**54,000** round diamonds from http://www.diamondse.info/
- Carat, colour, clarity, cut
- Total depth, table, depth, width, height
- Price



Your turn

Recall five ways to inspect a data set.

You have one minute!



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

COLOR GRADING SCALE

D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Colorless			Near Colorless				Faint Yellow			Very Light Yellow					Light Yellow							

http://www.mlcld.com/colour.aspx

Illustration of inclusions as seen under X10 magnification

http://www.thediamondsexperts.com/diamonds-guide

Layer basics

```
# What happens?
ggplot(diamonds)
aes(price, carat)
ggplot(diamonds, aes(price, carat))

geom_point()
geom_histogram()
```

Layers

- Every layer must have a dataset and aesthetic mappings
- If not supplied, they are inherited from the defaults supplied in `ggplot()`
- This means there are many ways to express the same plot

```
ggplot(diamonds, aes(price, carat)) +
  geom_point()

ggplot() +
  geom_point(aes(price, carat), diamonds)
```

```
ggplot(diamonds) +
  geom_point(aes(price, carat))

ggplot(diamonds, aes(price)) +
  geom_point(aes(y = carat))
```

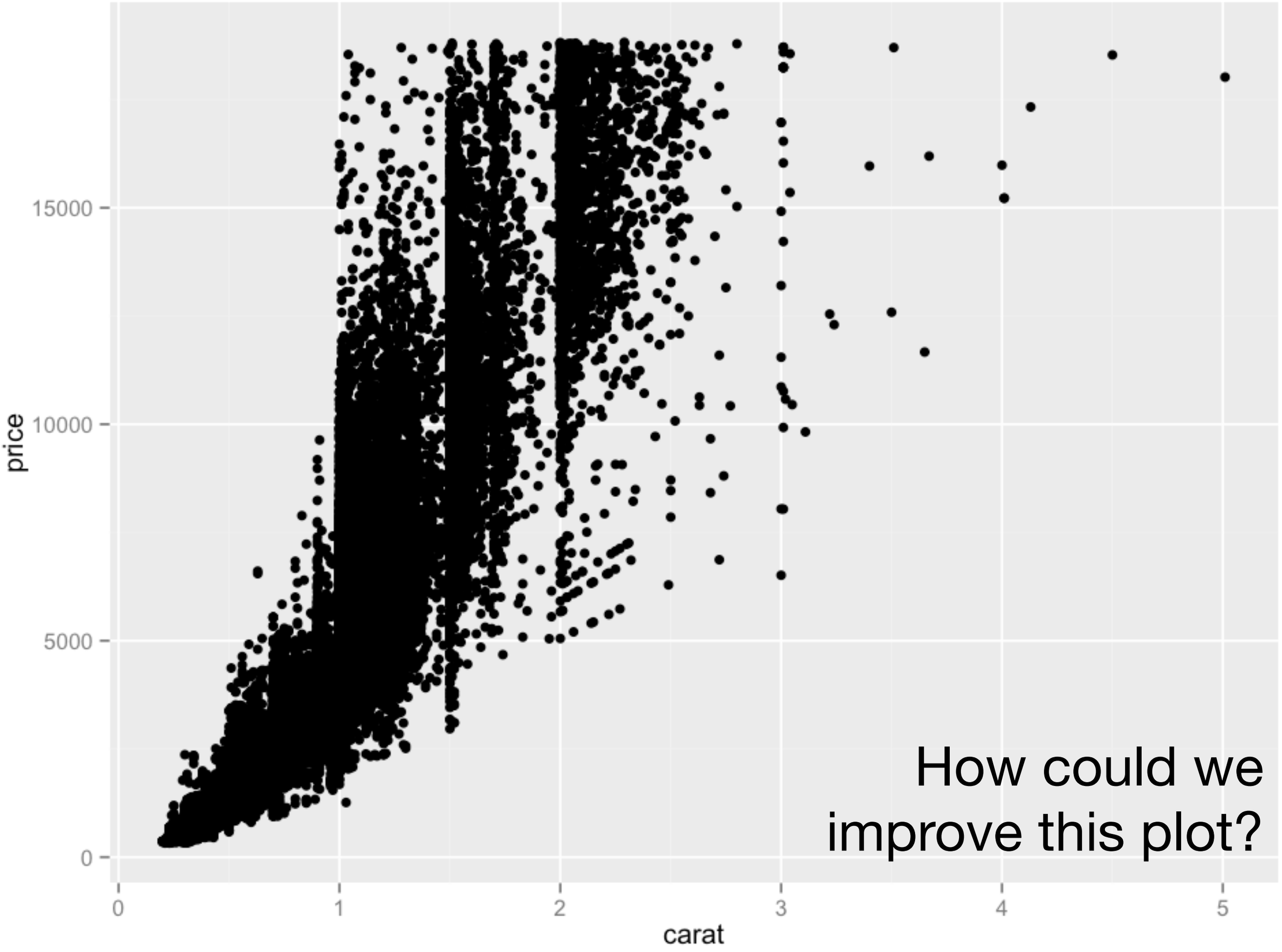
Layers

- Layers also have a geom, stat and position adjustment
  - **geom** = geometric object
  - **stat** = statistical transformation
  - **position** = how to handle overlapping points

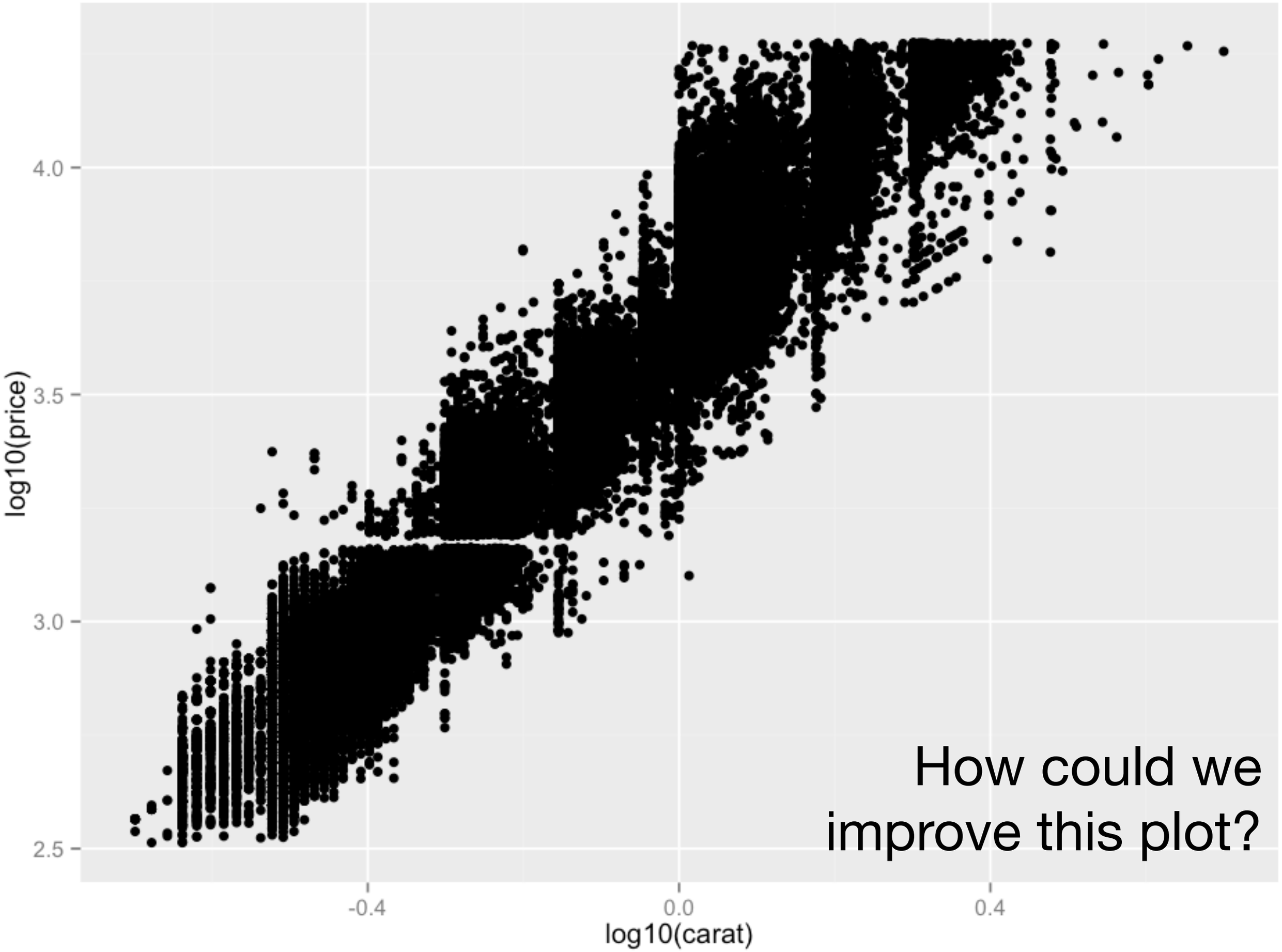
Layers

- Three basics *types* of layers
  - Raw data displays the data as is
  - Annotations display context (metadata)
  - **Summaries** display statistical transformations of the data

Warmups



How could we improve this plot?



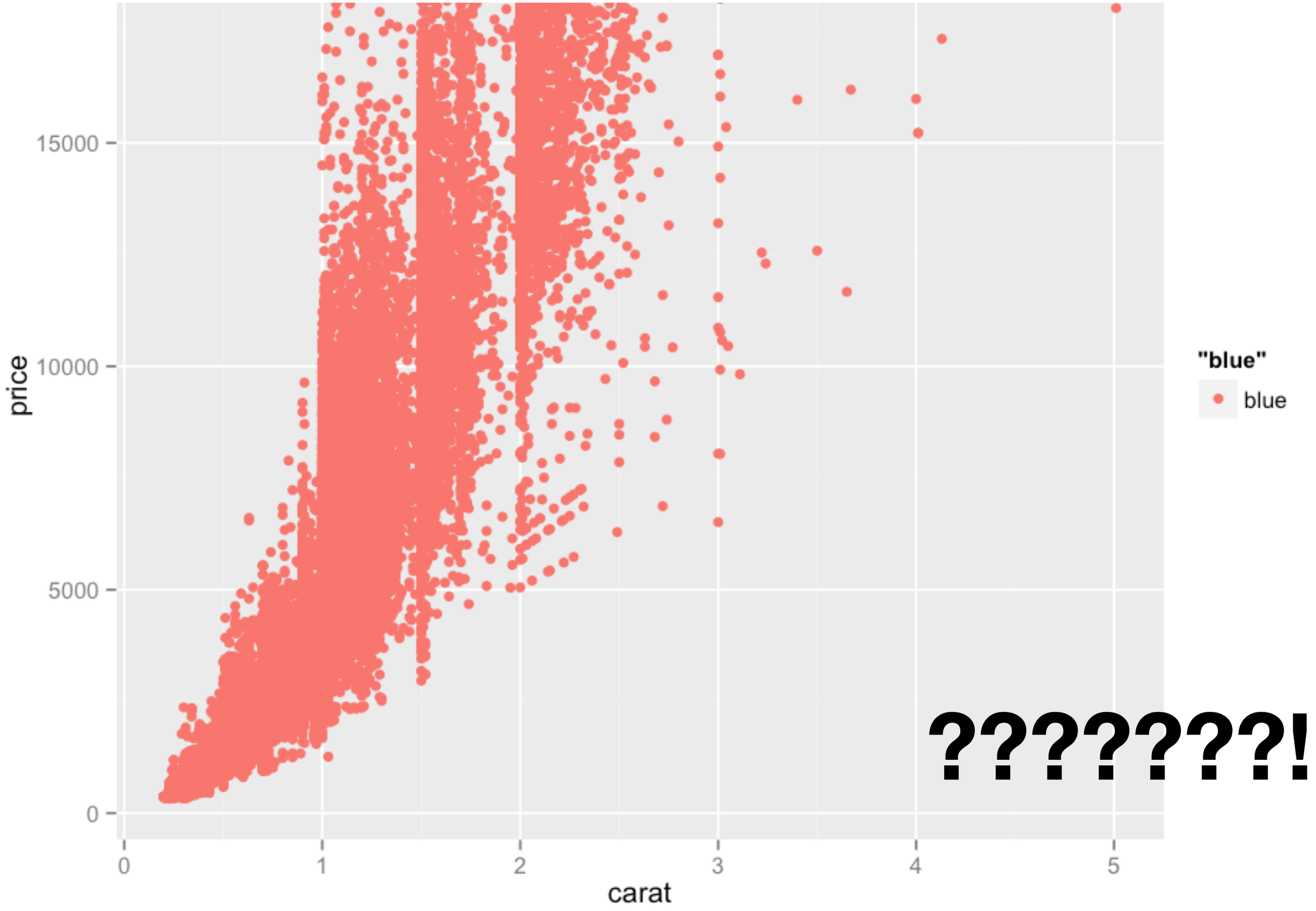
How could we improve this plot?

Techniques

- Change aesthetics: alpha, size, shape
- Conditional summaries:
  - smooth
  - boxplots, violin plots
  - histograms & variations
- Joint summaries: count, bin, density

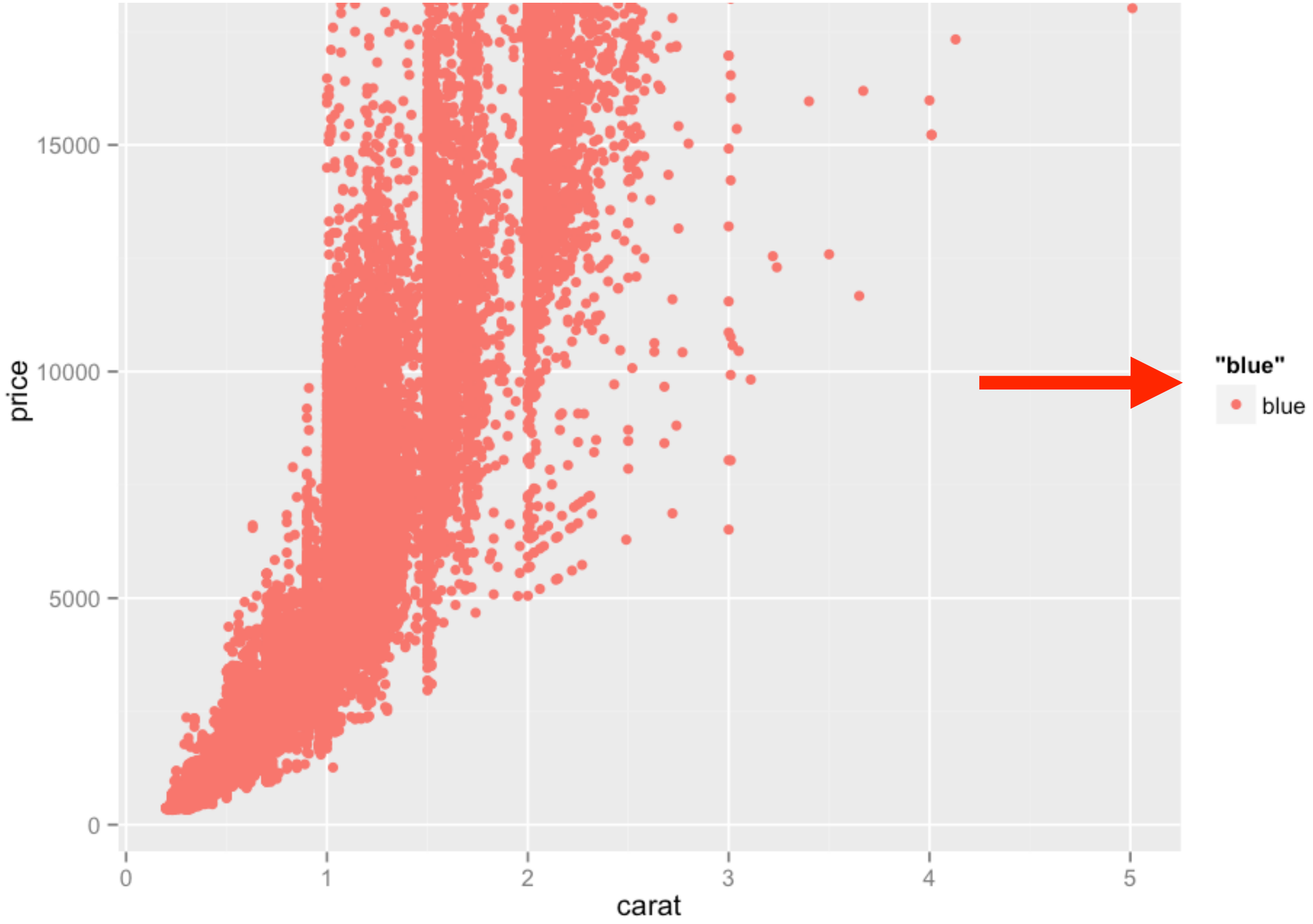
Changing aesthetics

```
ggplot(diamonds, aes(carat, price)) +
  geom_point(aes(colour = "blue"))
```

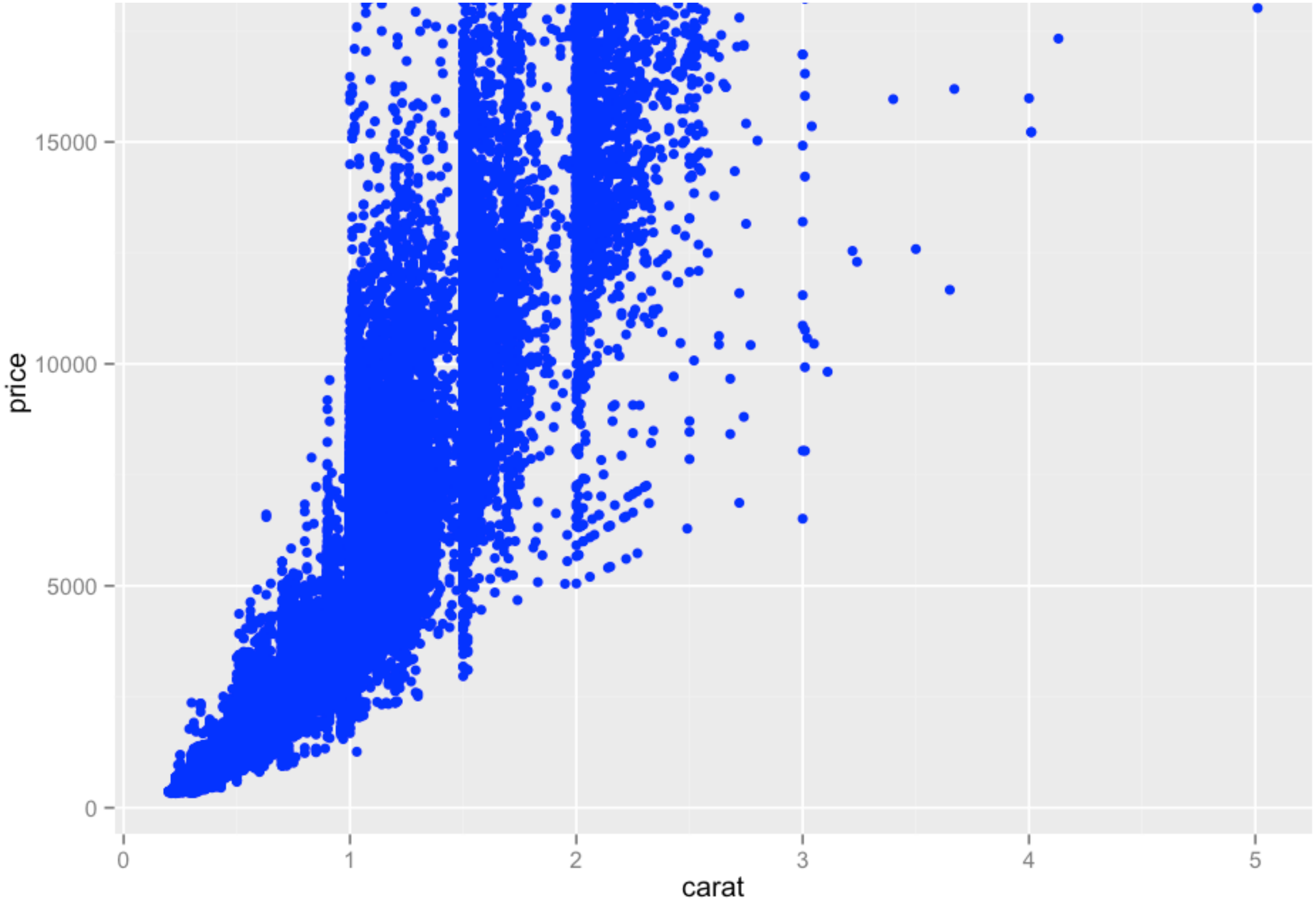


???????!

```
ggplot(diamonds, aes(carat, price)) +
  geom_point(aes(colour = "blue"))
```



```
ggplot(diamonds, aes(carat, price)) +
  geom_point(colour = "blue")
```



```
ggplot(diamonds, aes(carat, price)) +
  geom_point(alpha = 1/10)

ggplot(diamonds, aes(carat, price)) +
  geom_point(shape = 1)

ggplot(diamonds, aes(carat, price)) +
  geom_point(size = 0.1)
```

Your turn

What’s the best you can do by combining alpha, size and shape?

Conditional summaries

Your turn

What does `geom_smooth()` do? Read the documentation then try it out. What type of model is most appropriate here?

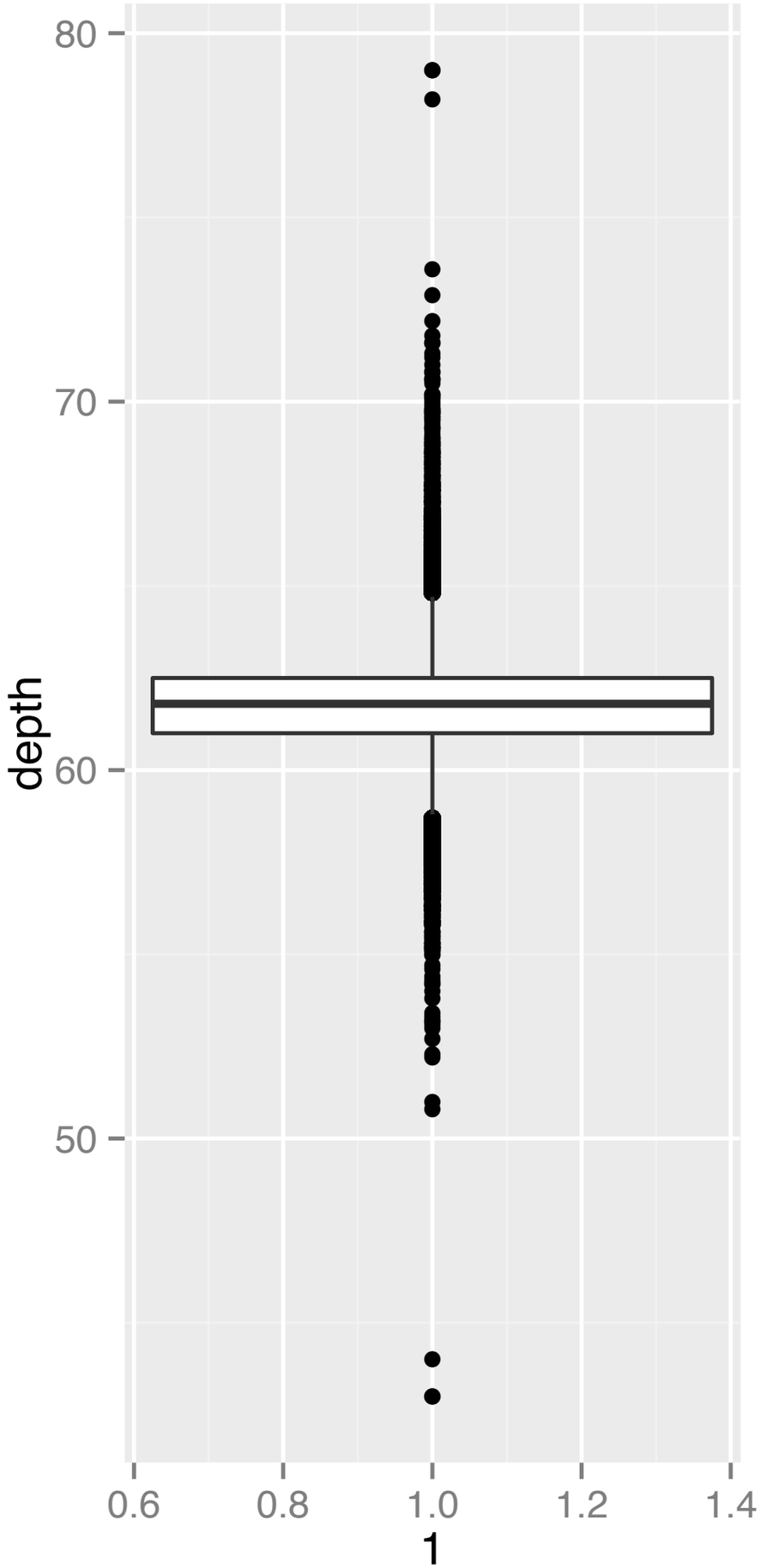
What happens if you try to use `geom_boxplot()`?

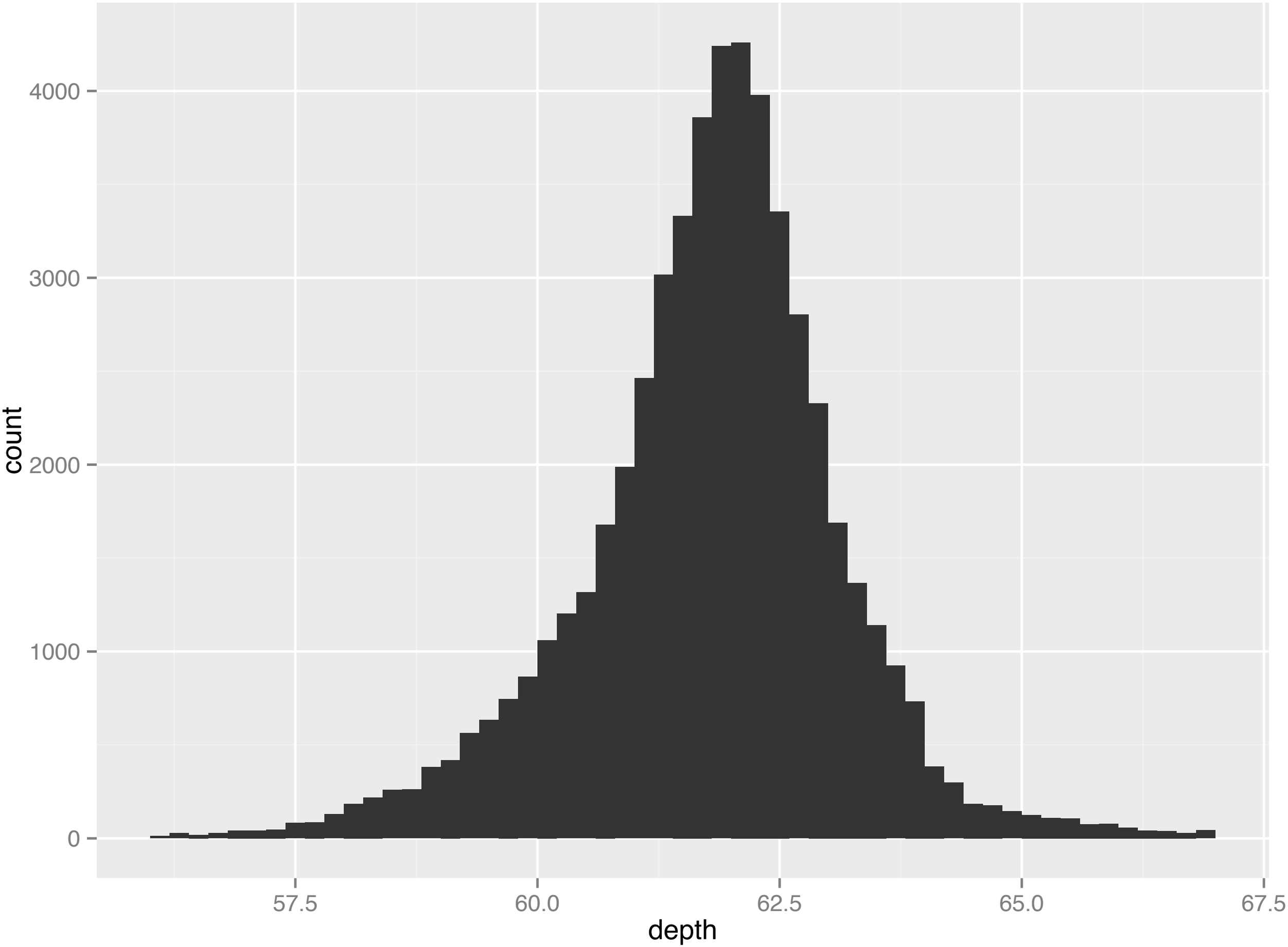
```
# Only one box!
ggplot(diamonds, aes(log10(carat), log10(price))) +
  geom_boxplot()

# Need to specify grouping to get multiple
# boxplots. plyr::round_any convenient tool
ggplot(diamonds, aes(log10(carat), log10(price))) +
  geom_boxplot(
   aes(group = plyr::round_any(log10(carat), 0.1))
  )
```

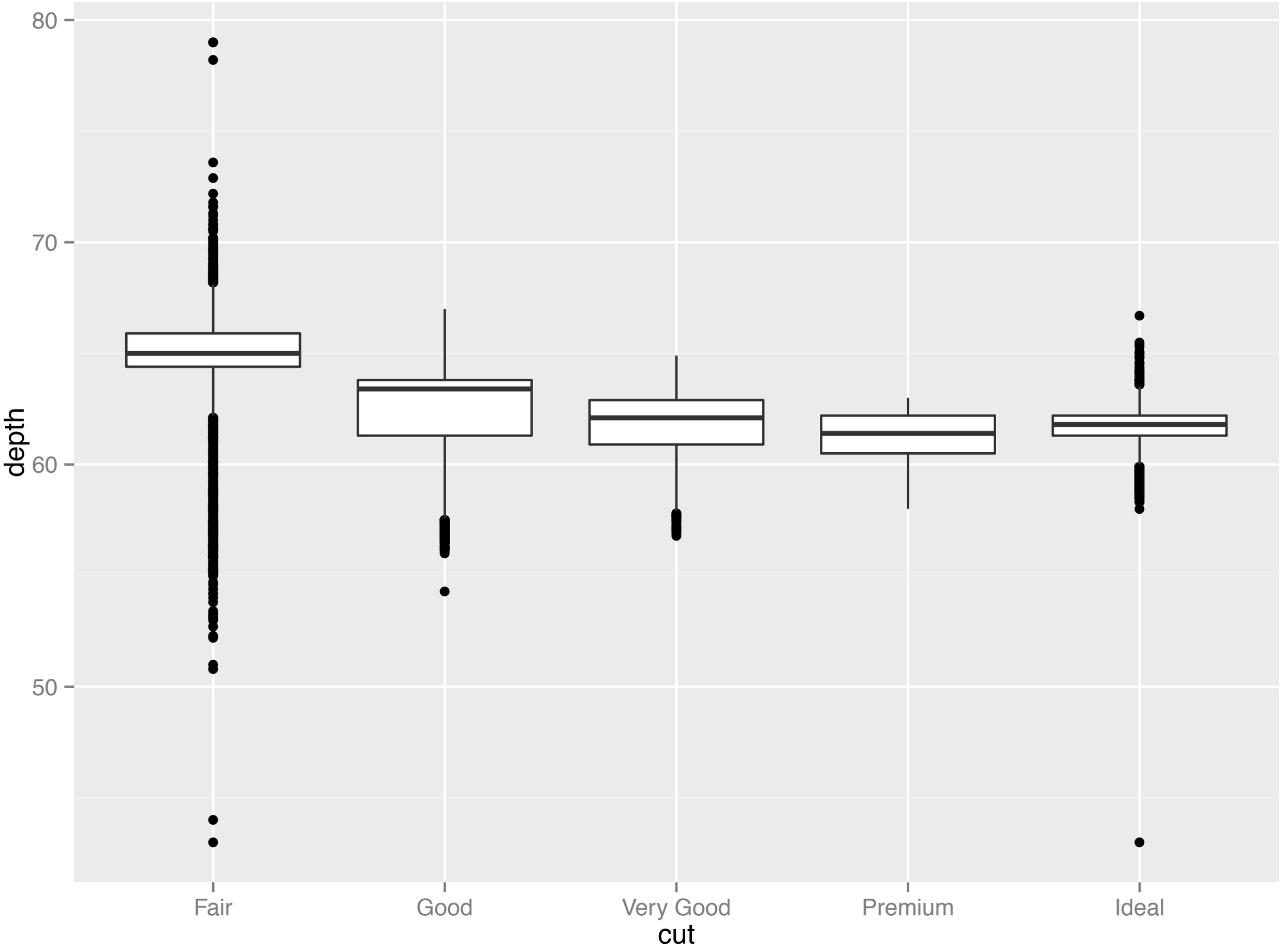
Boxplot vs histograms

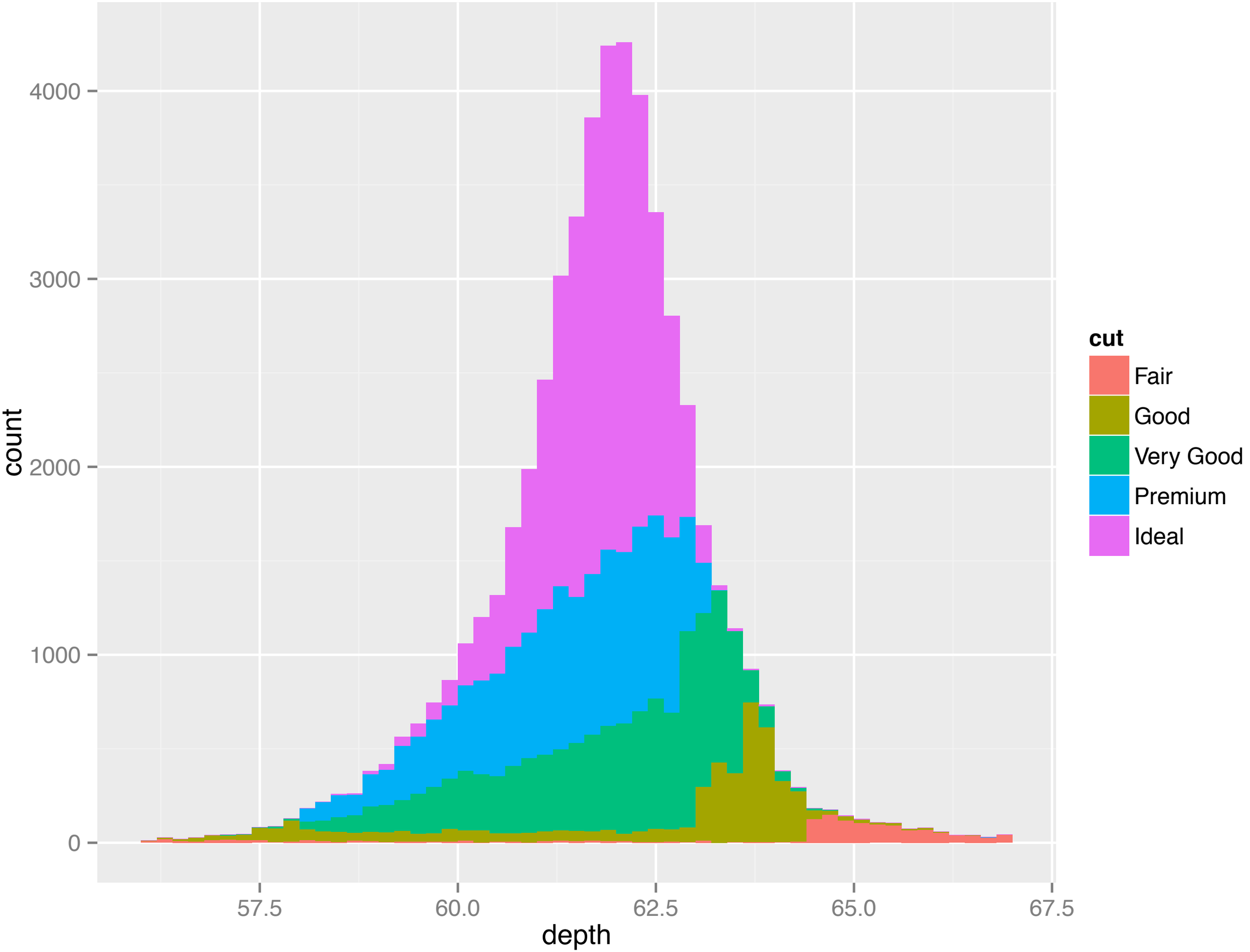
Boxplots: show handful of summary statistics; but compact

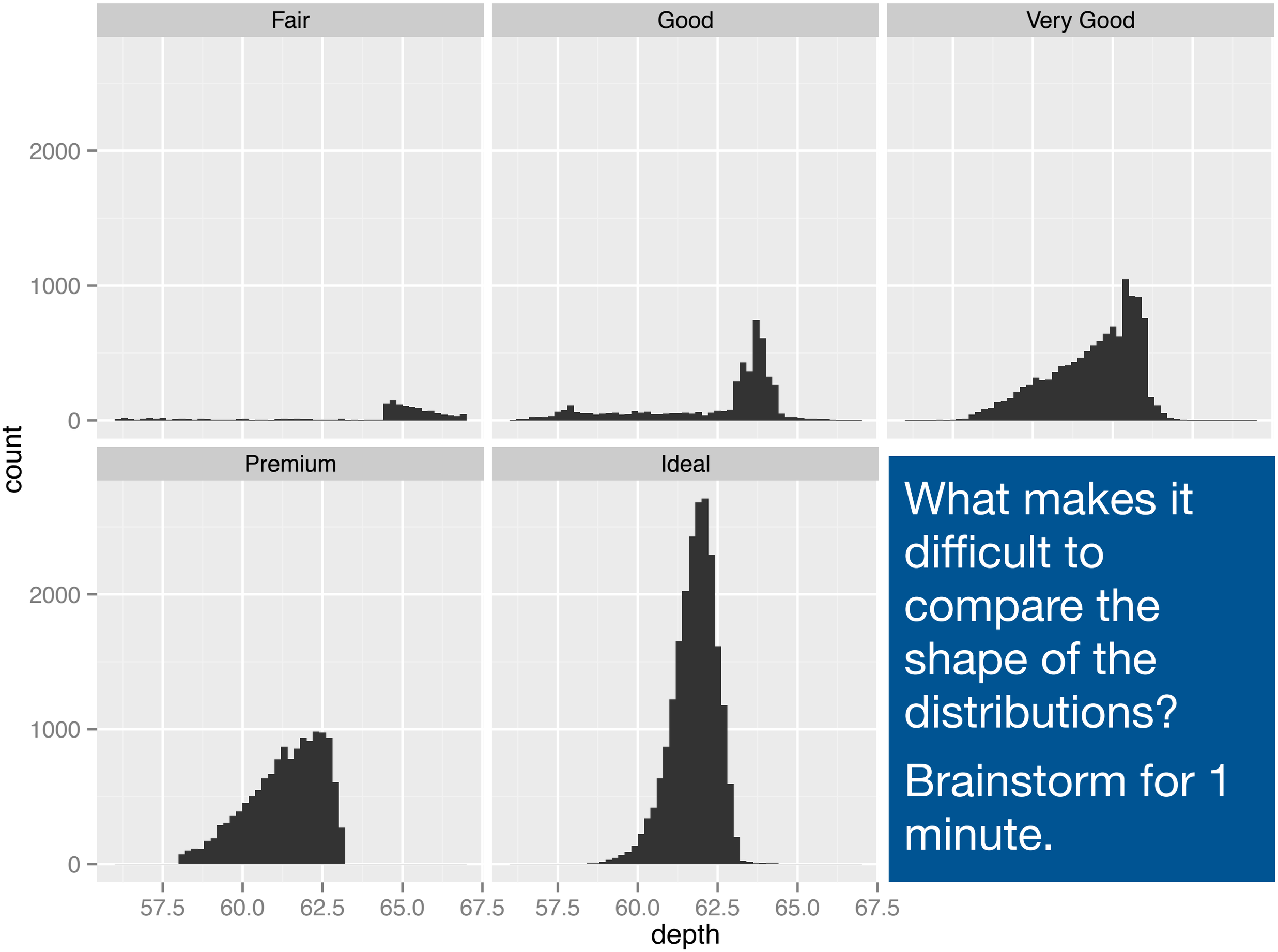
Histograms: show detailed distribution; but take up a lot of space


80
70
60
50
depth
0.6
0.8
1.0
1.2
1.4
1


4000
3000
2000
1000
0
count
57.5
60.0
62.5
65.0
67.5
depth

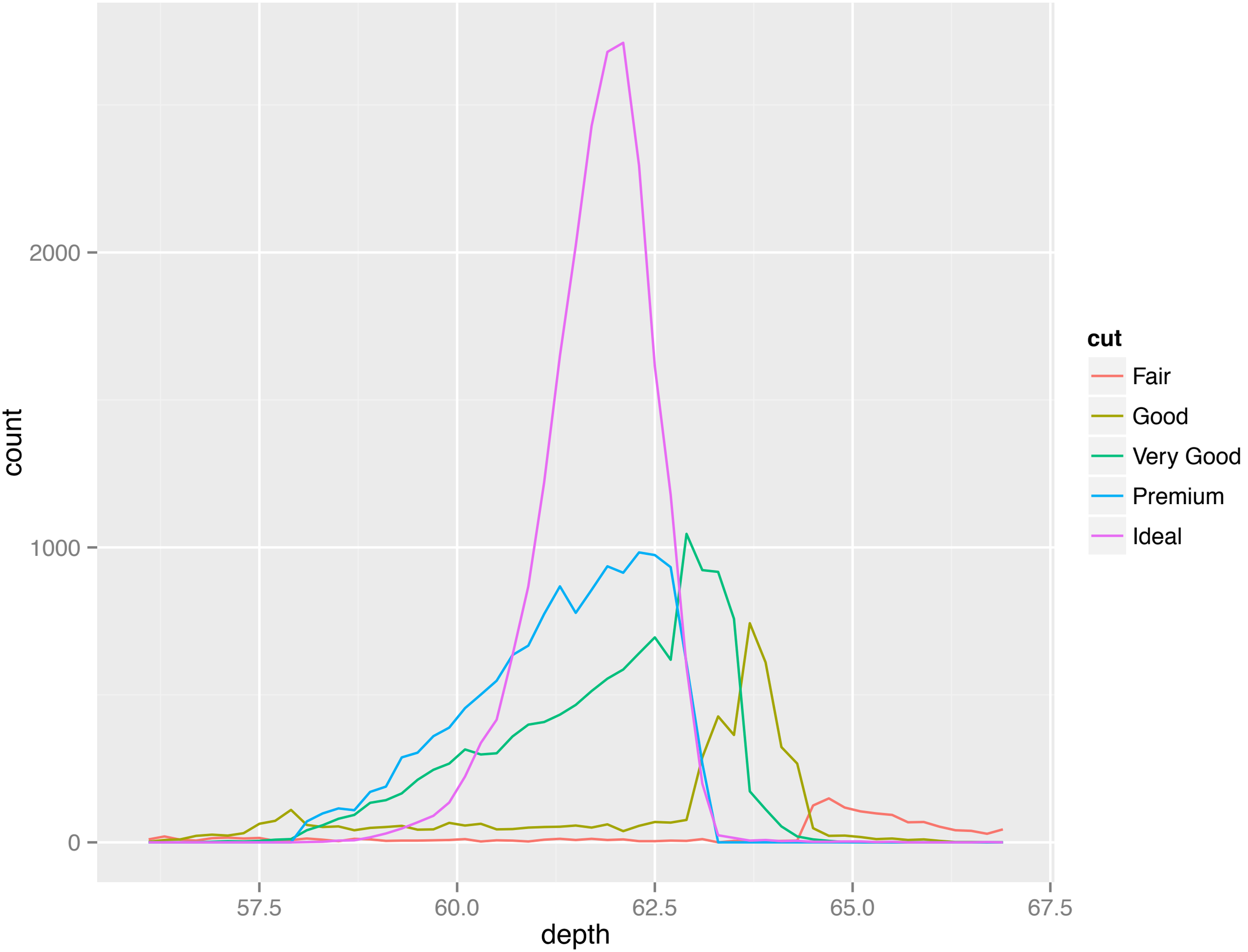

80
70
60
50
depth
Fair
Good
Very Good
Premium
Ideal
cut

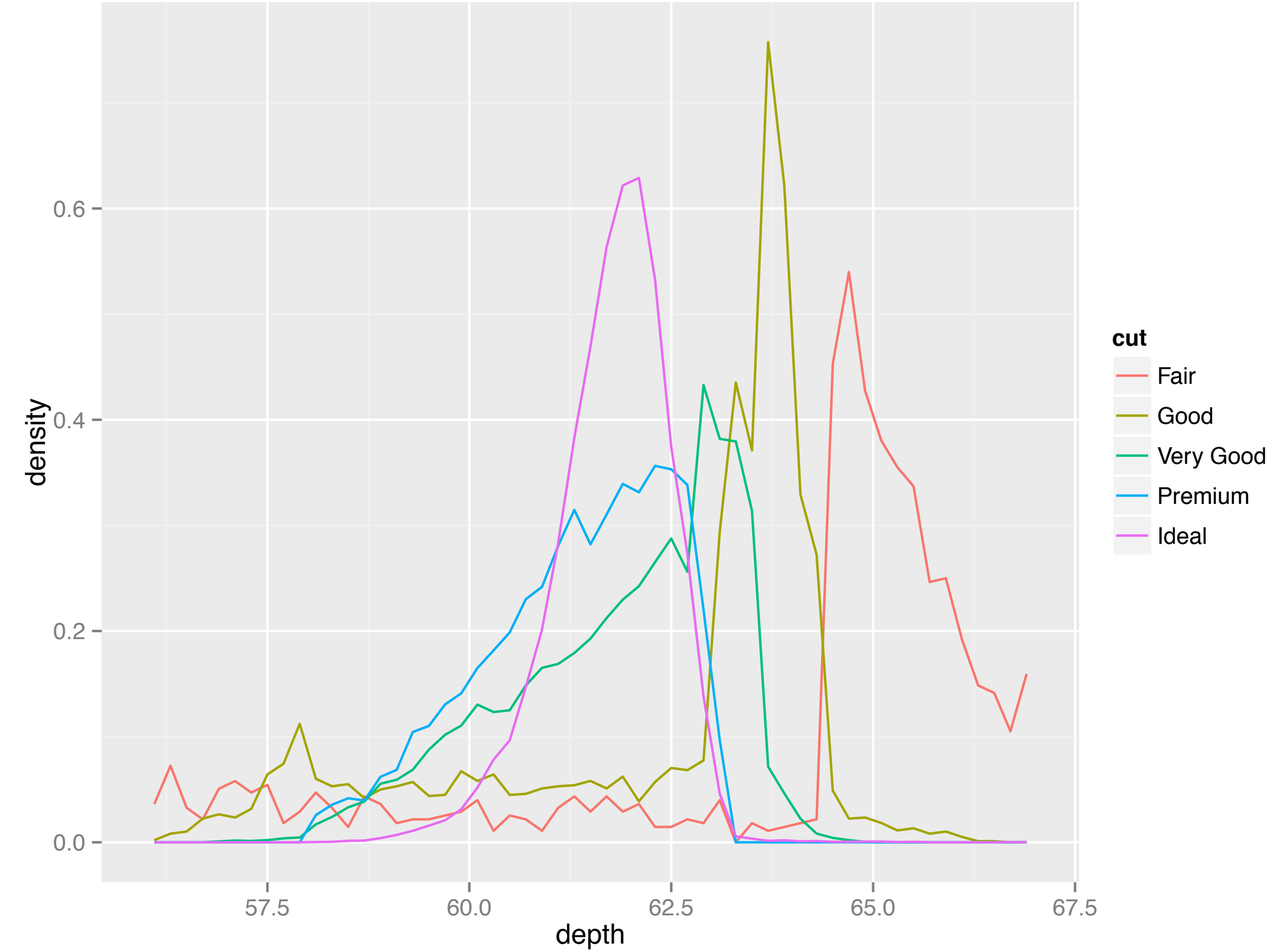

4000
3000
2000
1000
0
count
57.5
60.0
62.5
65.0
67.5
depth
cut
Fair
Good
Very Good
Premium
Ideal



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

Brainstorm for 1 minute.


2000
1000
0
count
57.5
60.0
62.5
65.0
67.5
depth
cut
Fair
Good
Very Good
Premium
Ideal


0.6
0.4
0.2
0.0
density
57.5
60.0
62.5
65.0
67.5
depth
cut
Fair
Good
Very Good
Premium
Ideal

Like histogram, but lines

```
ggplot(diamonds, aes(depth, colour = cut)) +
  geom_freqpoly(binwidth = 0.2) +
  xlim(56, 67)
```

```
ggplot(diamonds, aes(depth, colour = cut)) +
  geom_freqpoly(aes(y = ..density..), binwidth = 0.2) +
  xlim(56, 67)
```

Use computed variable

Your turn

Compare the distribution of price for the different cuts. Does anything seem unusual?

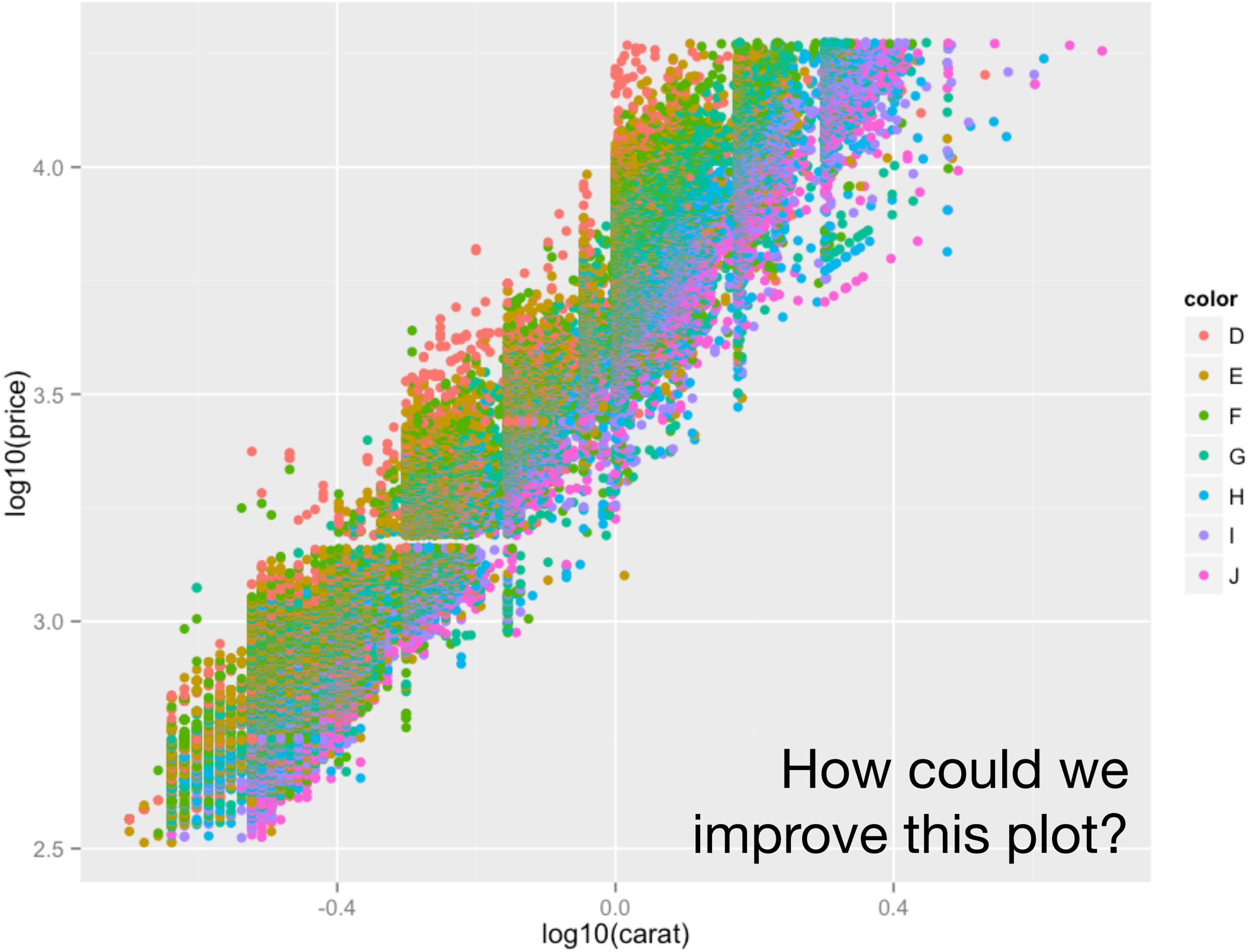
Start simple. First make sure you have a good visualisation of price, then figure out how to compare cuts.

Joint summaries

Your turn

Compare and contrast `geom_bin2d()`, and `geom_density2d()`. Which do you think works best for this data? Why?

Comparisons

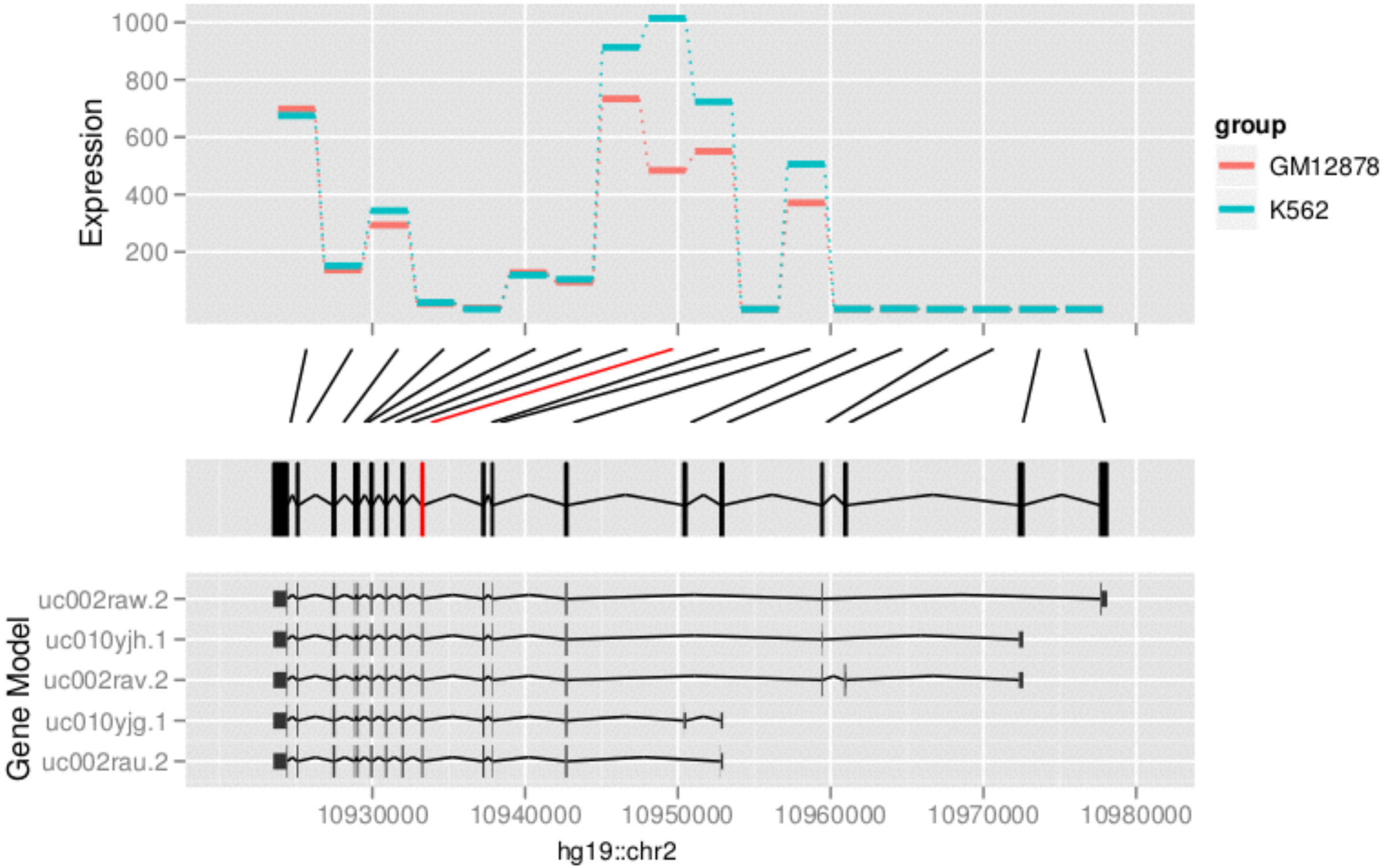

4.0
3.5
3.0
2.5
log10(price)
-0.4
0.0
0.4
log10(carat)
color
D
E
F
G
H
I
J
How could we improve this plot?

```
ggplot(diamonds, aes(log10(carat), log10(price))) +
  geom_bin2d() +
  facet_wrap(~color)

# Annotations can be very useful!
coef(lm(log10(price) ~ log10(carat), data = diamonds))
```

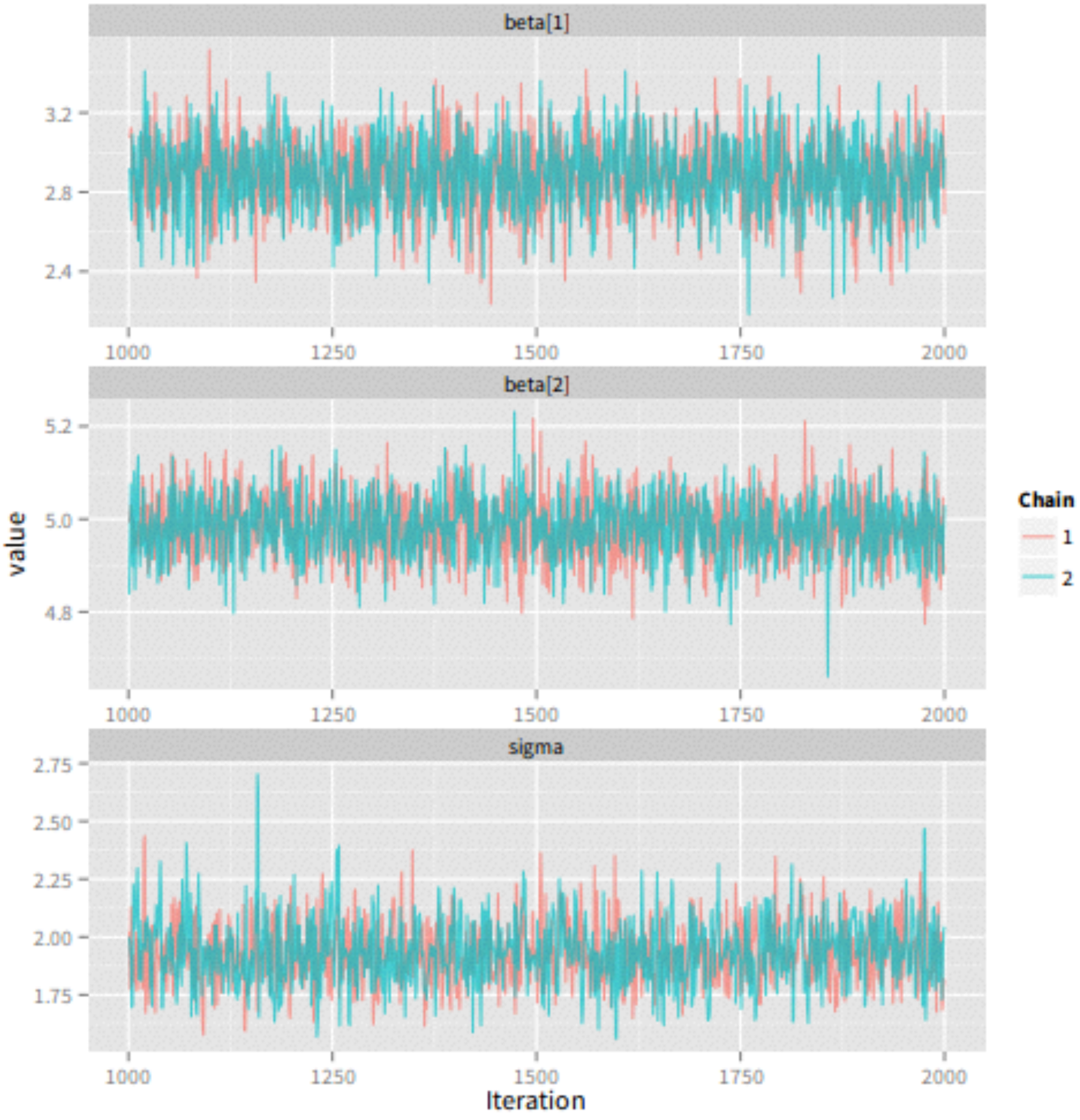
```
ggplot(diamonds, aes(log10(carat), log10(price))) +
  geom_bin2d() +
  geom_abline(intercept = 3.67, slope = 1.68,
    colour = "white") +
  facet_wrap(~color)
```

Where next

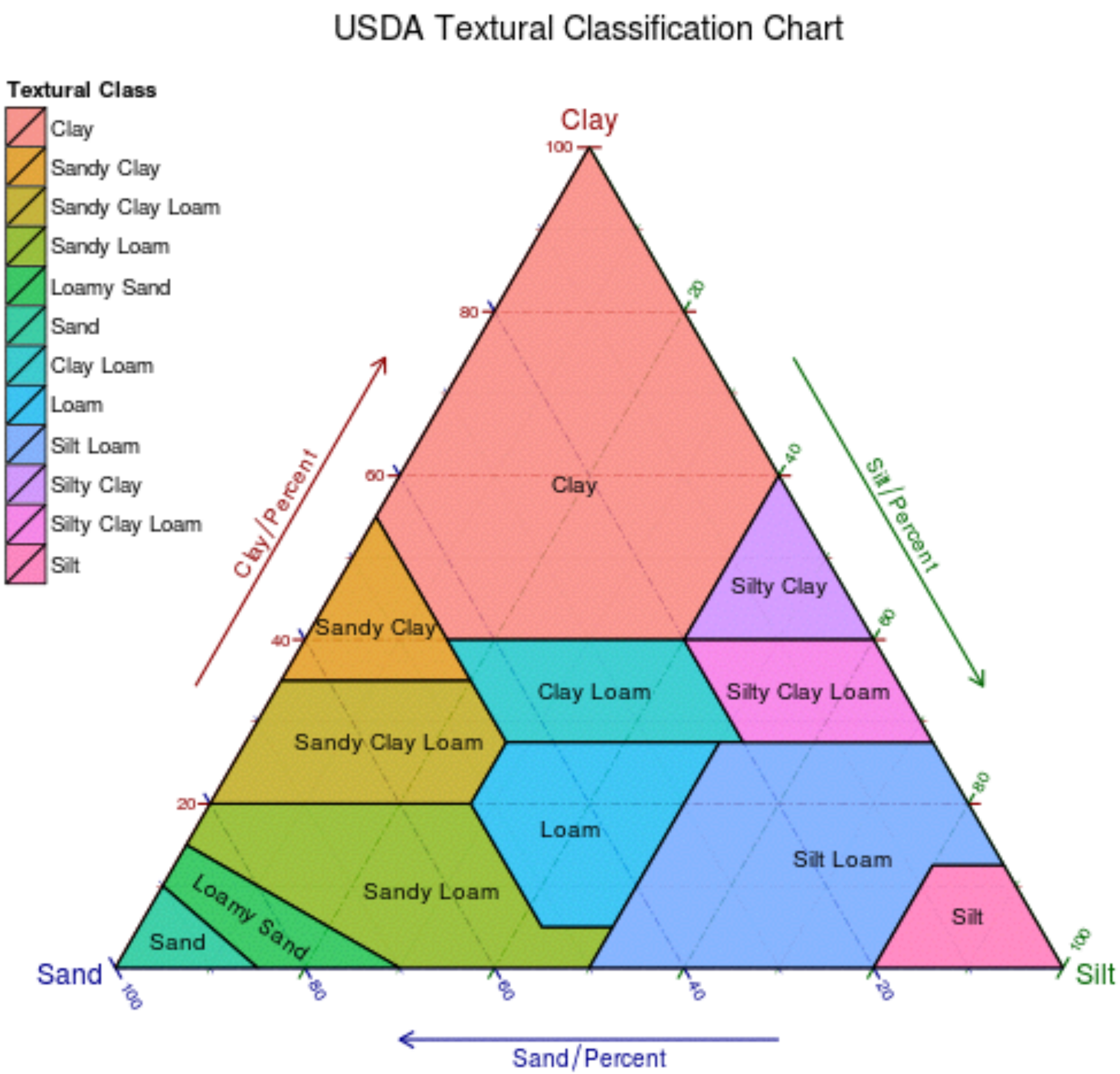
```
library(ggbio)
```



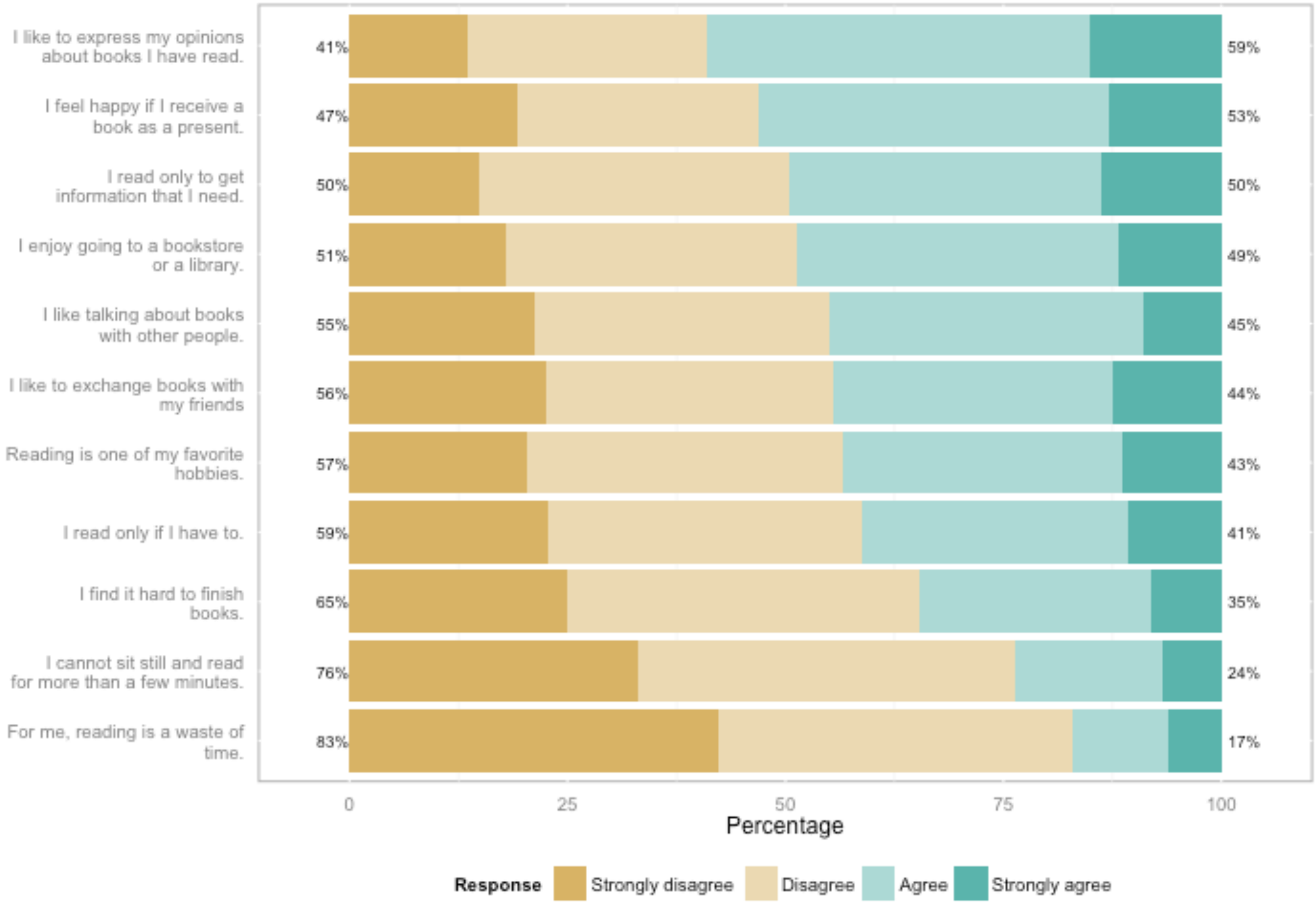
```
library(ggmcmc)
```



```
library(ggtern)
```



```
library(likert)
```



```
library(sjPlot)
```