orial

Getting started

First, we need to install and load the package,

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

Diamonds data

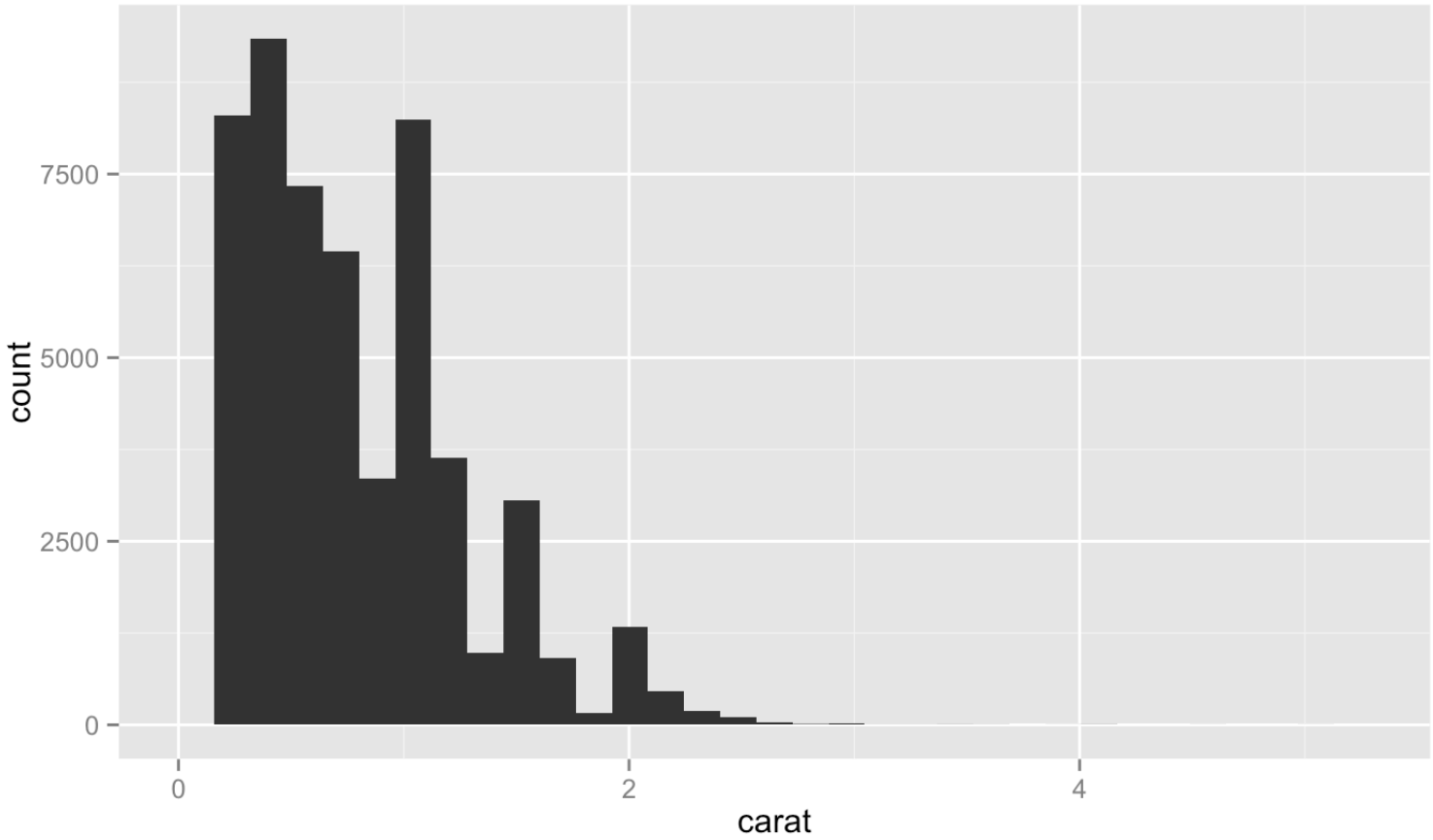
To start, I'm going to use the diamonds data that comes with ggplot2,

```
str(diamonds)
```

```
## 'data.frame':    53940 obs. of  10 variables:
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ..
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5
##  $ depth  : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num  55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int  326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

qplot()– the easy way out

```
qplot(carat, data=diamonds)
```



ggplot2 syntax

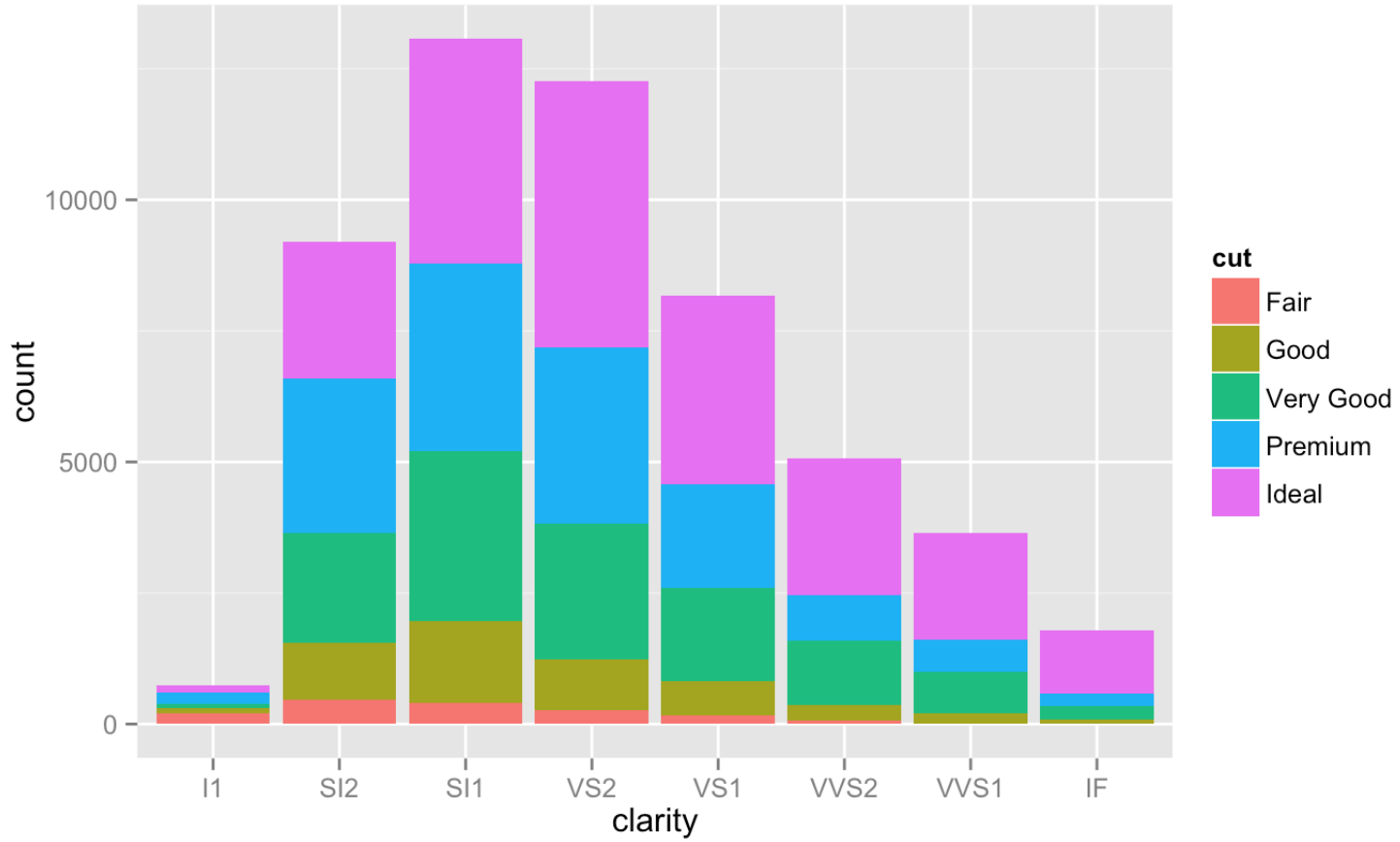
```
qplot(carat, data=diamonds)
```

qplot() performs similar functionality to the base R graphics function plot(). But it aleady might seem a little different, because we're not using the $ operator.

Instead, you're listing the name of the variable(s) and then telling R where to "look" for that variable with "data=". This is like what we do when modeling using functions like lm().

More qplot()

```
qplot(clarity, fill=cut, data=diamonds)
```



ggplot()

But, in order to really harness the power of ggplot2 you need to use the more general ggplot() command. The idea of the package is you can "layer" pieces on top of a plot to build it up over time.

You always need to use a ggplot() call to initialize the plot. I usually put my dataset in here, and at least some of my "aesthetics." But, one of the things that can make ggplot2 tough to understand is that there are no hard and fast rules.

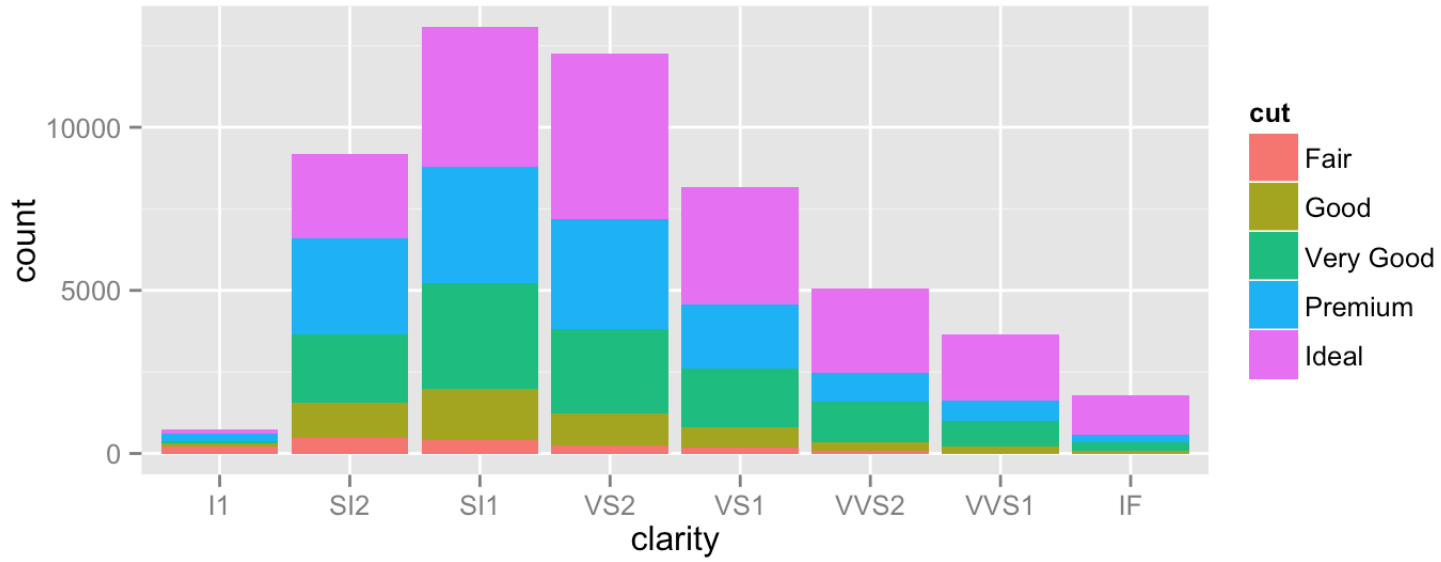
```
p1 <- ggplot(aes(x=clarity, fill=cut), data=diamonds)
```

If you try to show p1 at this point, you will get "Error: No layers in plot." This is because we haven't given it any geometric objects yet.

geoms

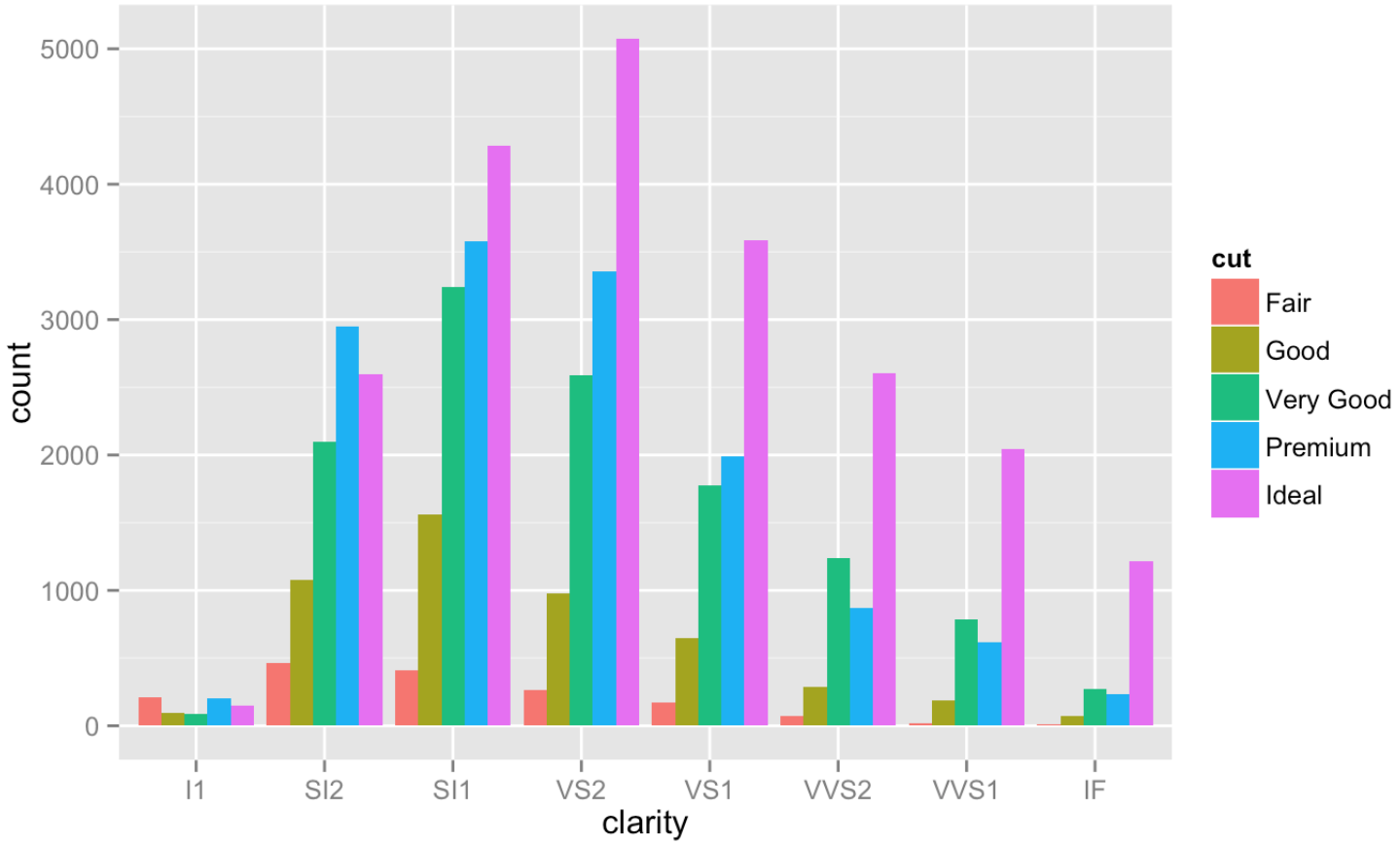
In order to get a plot to work, you need to use "geoms" (geometric objects) to specify the way you want your variables mapped to graphical parameters.

```
p1 + geom_bar()
```



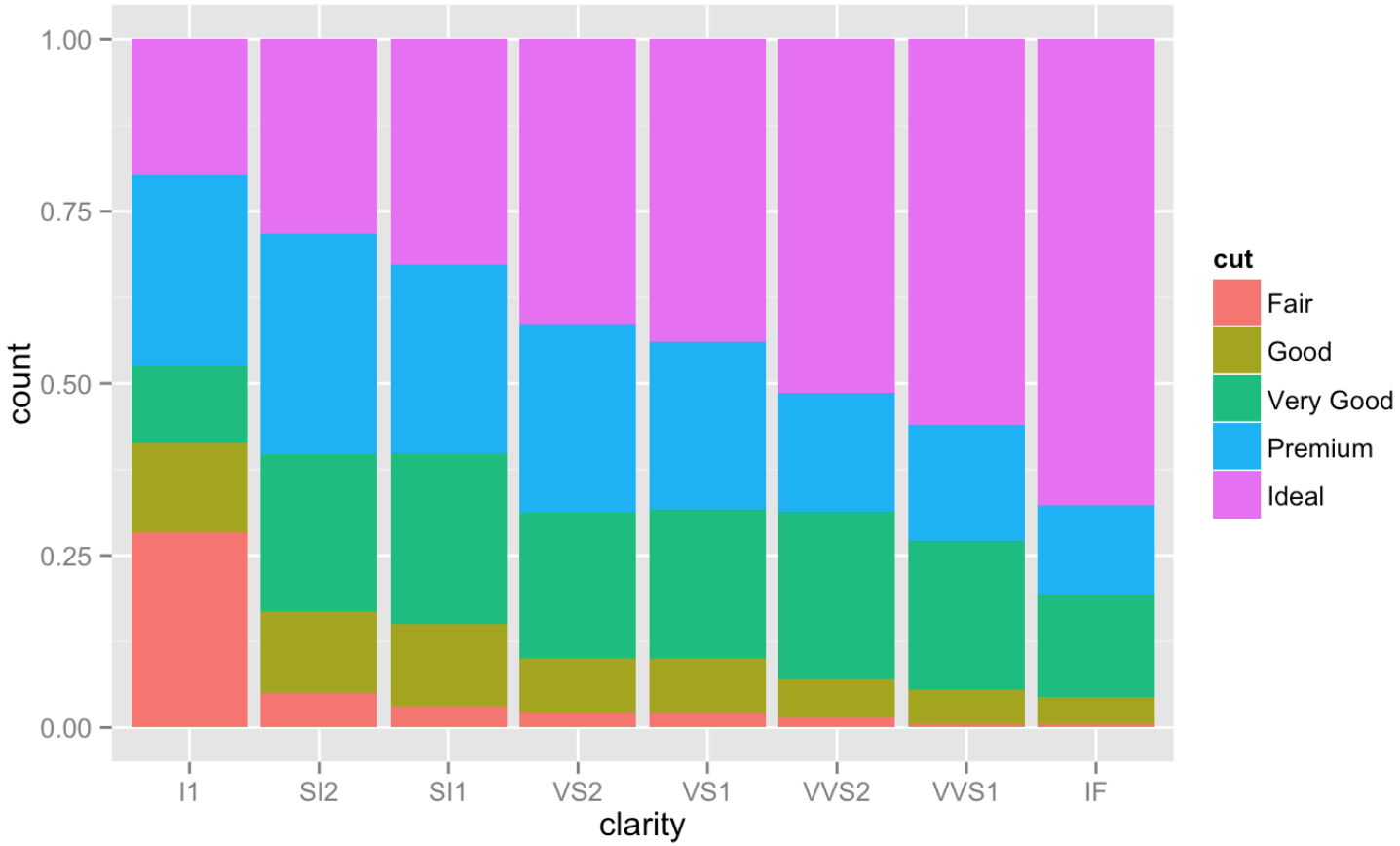
geoms have options

```
p1 + geom_bar(position="dodge")
```



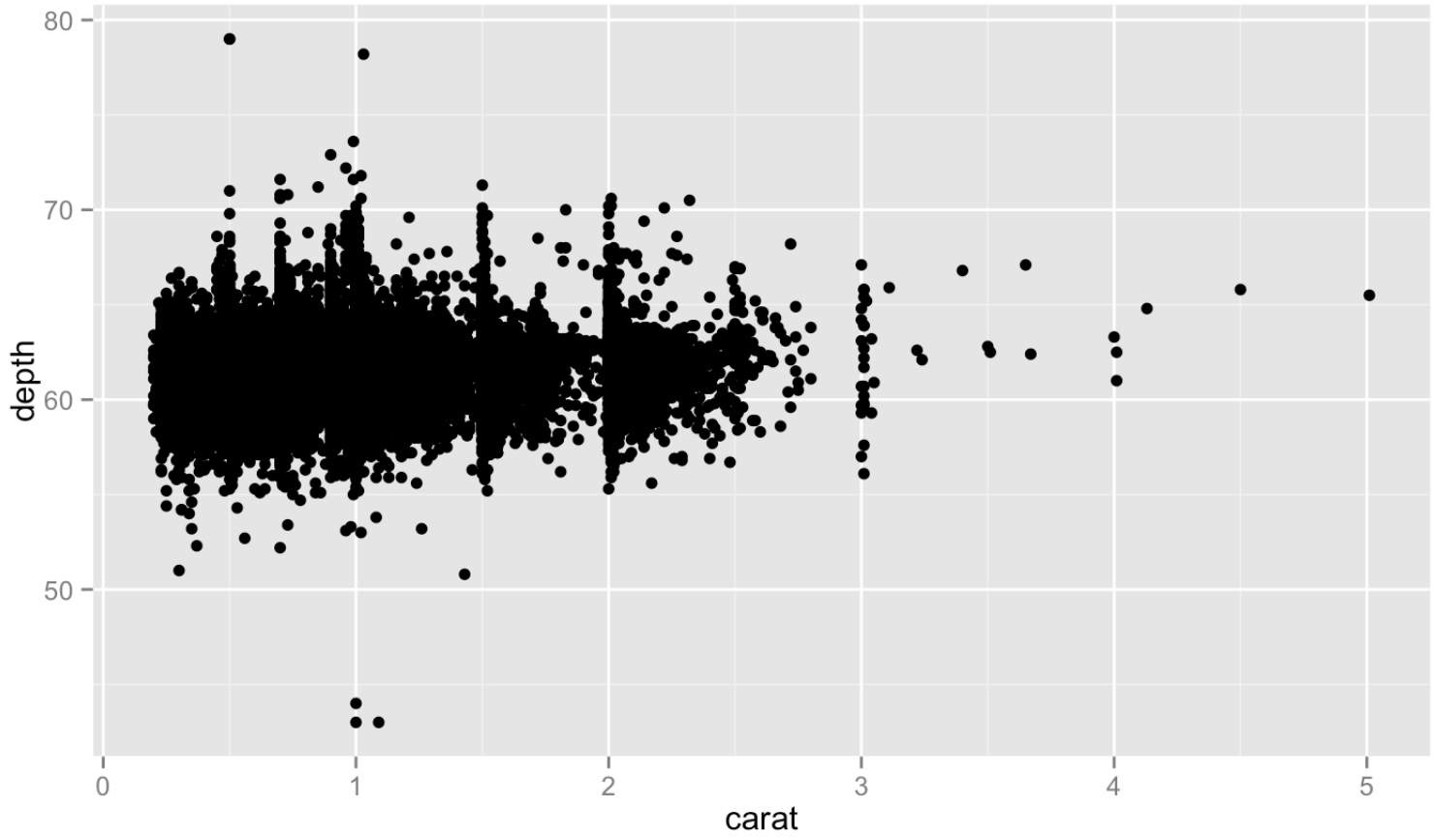
Lots of options

```
p1 + geom_bar(position="fill")
```



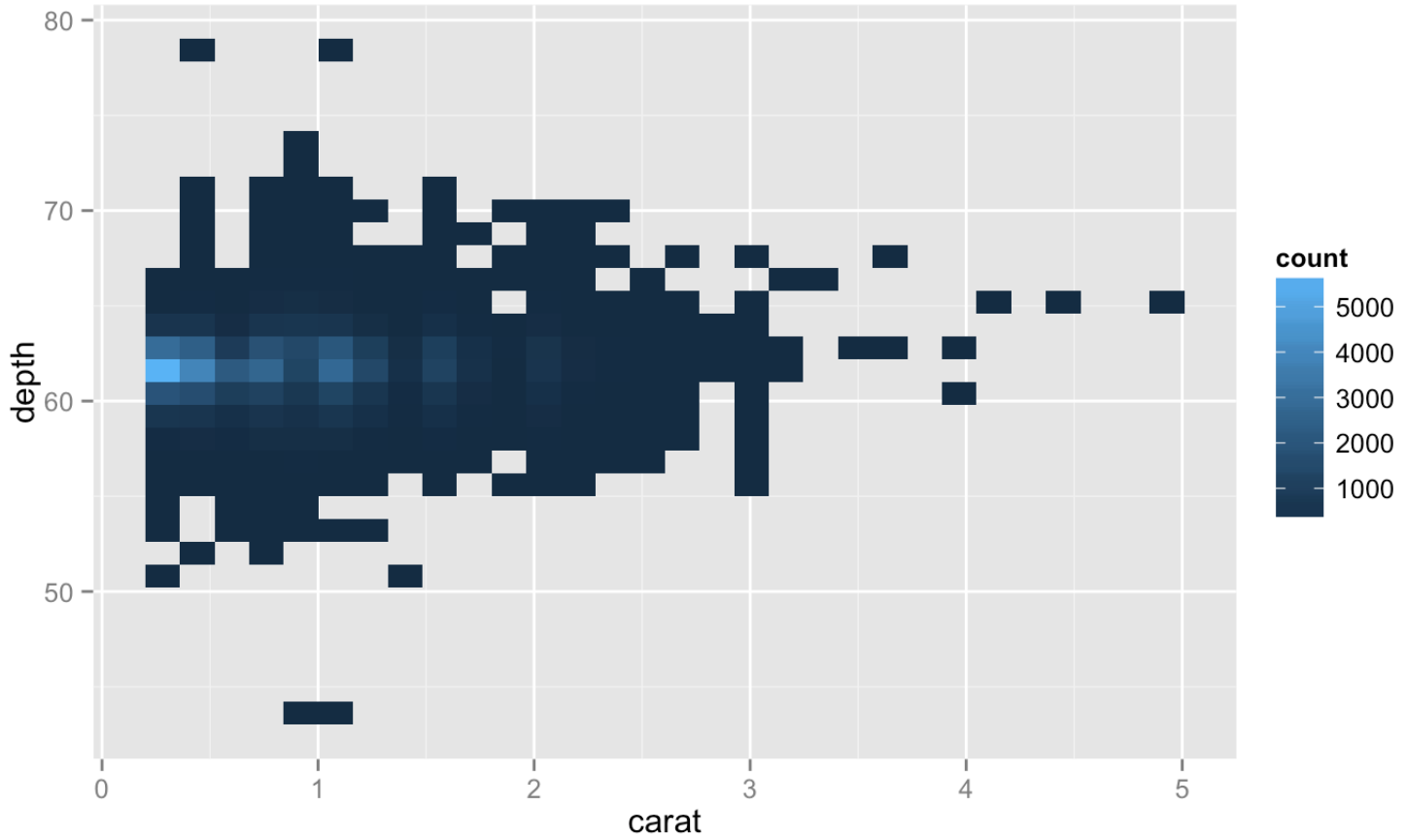
Two variables

```
p2 <- ggplot(aes(x=carat, y=depth), data=diamonds)
p2 + geom_point()
```



Same data, different geom

```
p2 + geom_bin2d()
```



Saving your work (or not)

Notice that I'm not saving these geom layers– I'm just running

p2 + [something]

to see what happens. But, I can save the new version to start building up my plot,

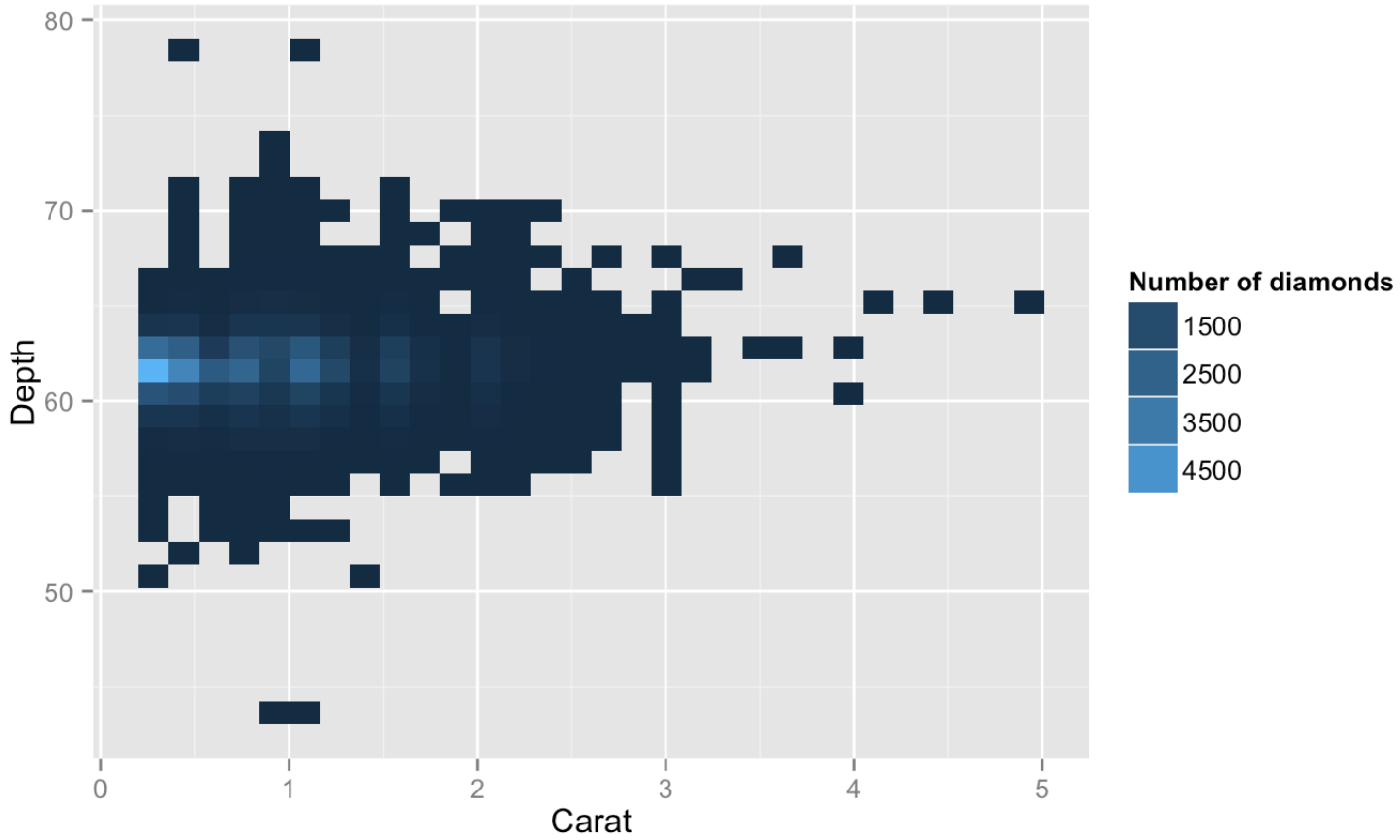
```
p2 <- p2 + geom_bin2d()
```

Better labels

```
p2 <- p2 + xlab("Carat") + ylab("Depth") +
 guides(fill=guide_legend(title="Number of diamonds"))
p2
```

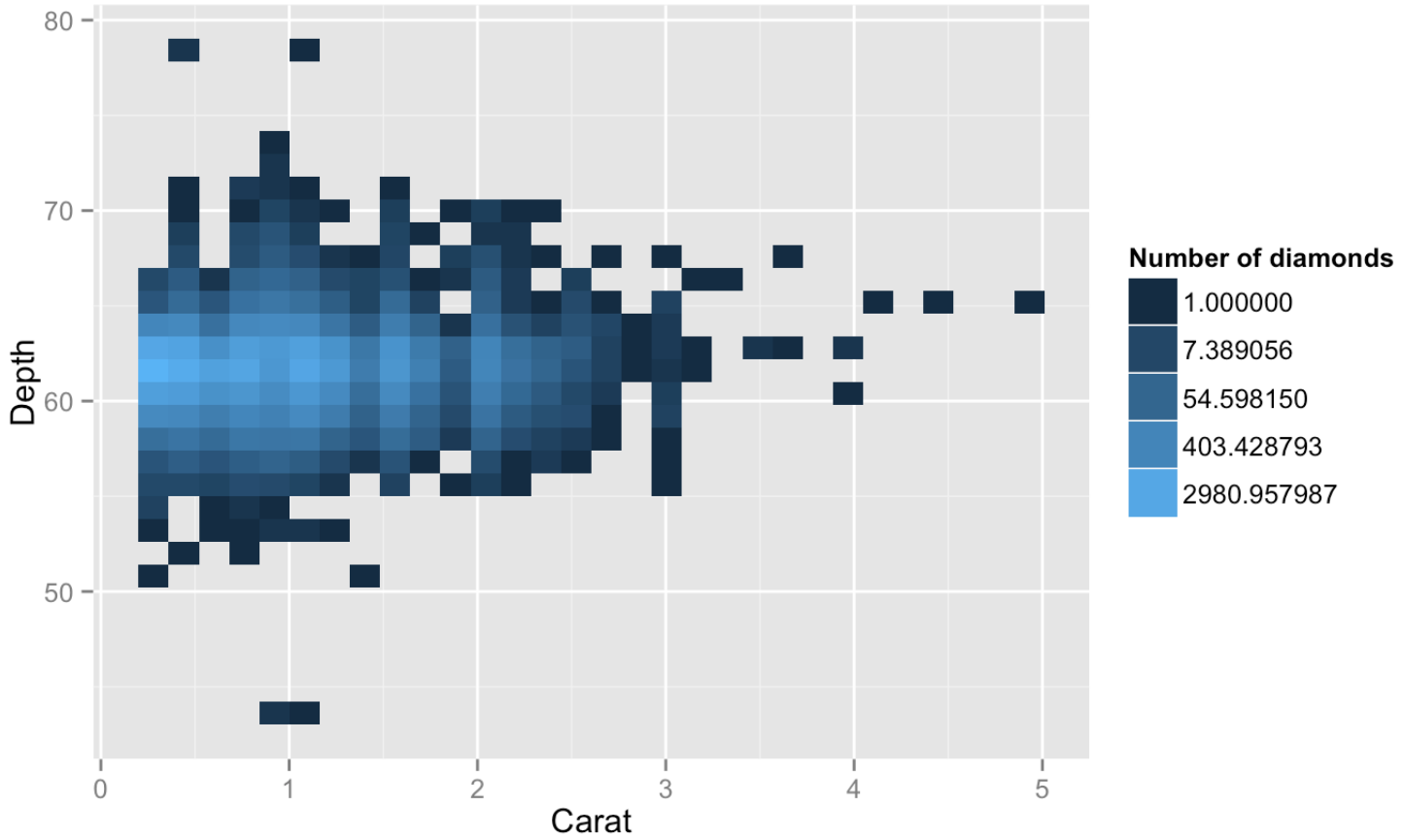
Different breaks

```
p2 + scale_fill_continuous(breaks=c(1500, 2500, 3500,4500))
```



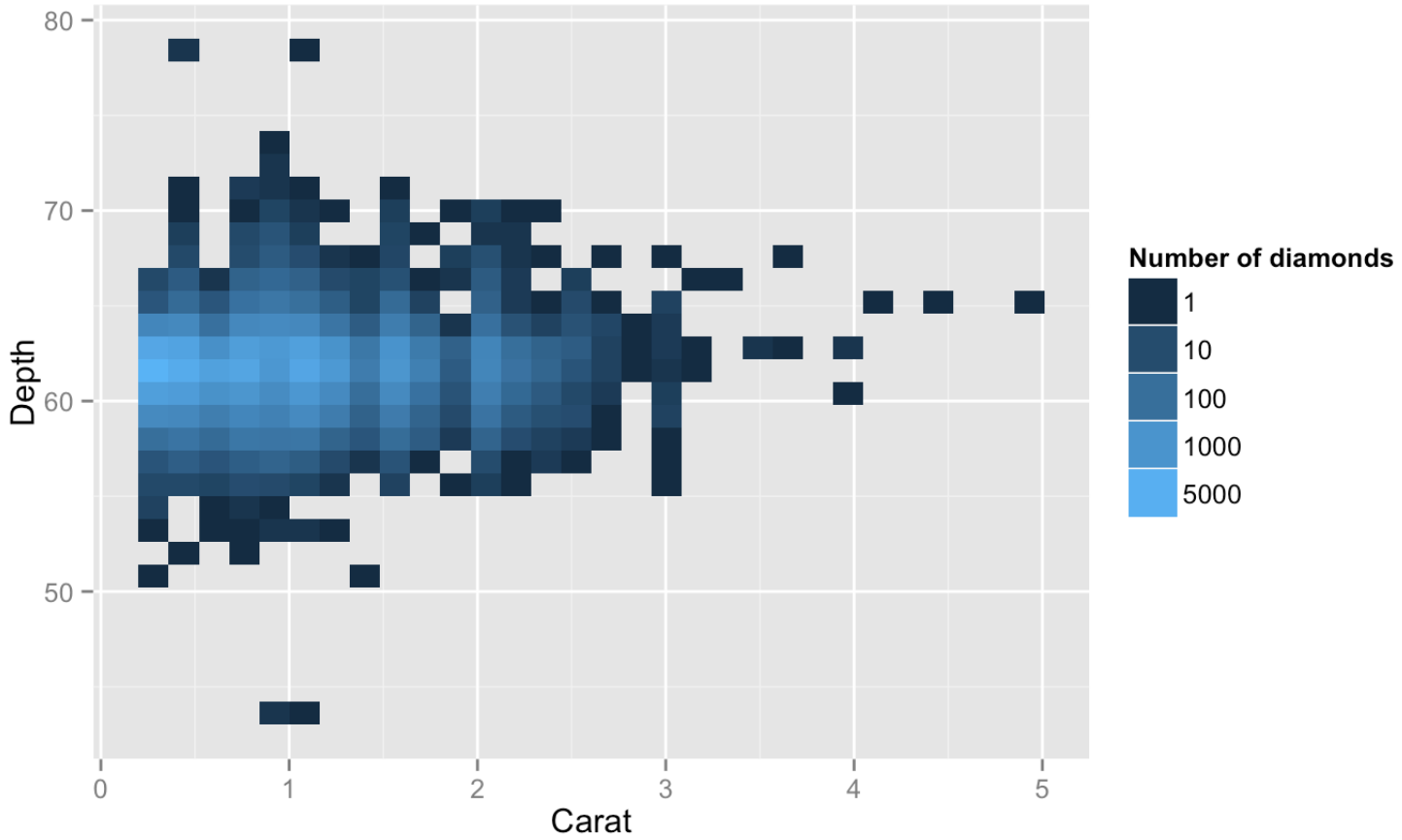
Log scale

```
p2 + scale_fill_continuous(trans="log")
```



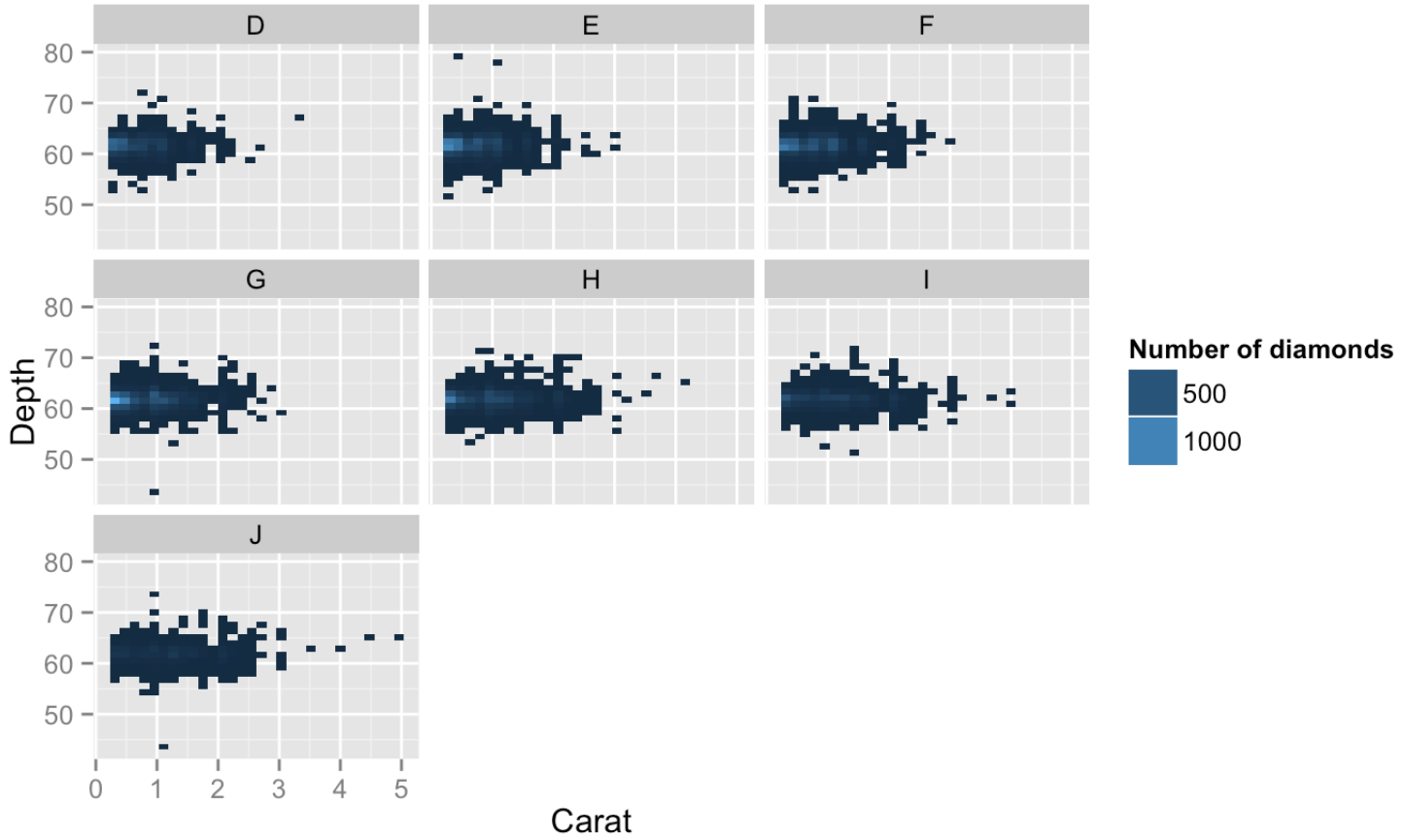
Log scale, different breaks

```
p2 + scale_fill_continuous(trans="log", breaks=c(1,10,100,1000,5000))
```



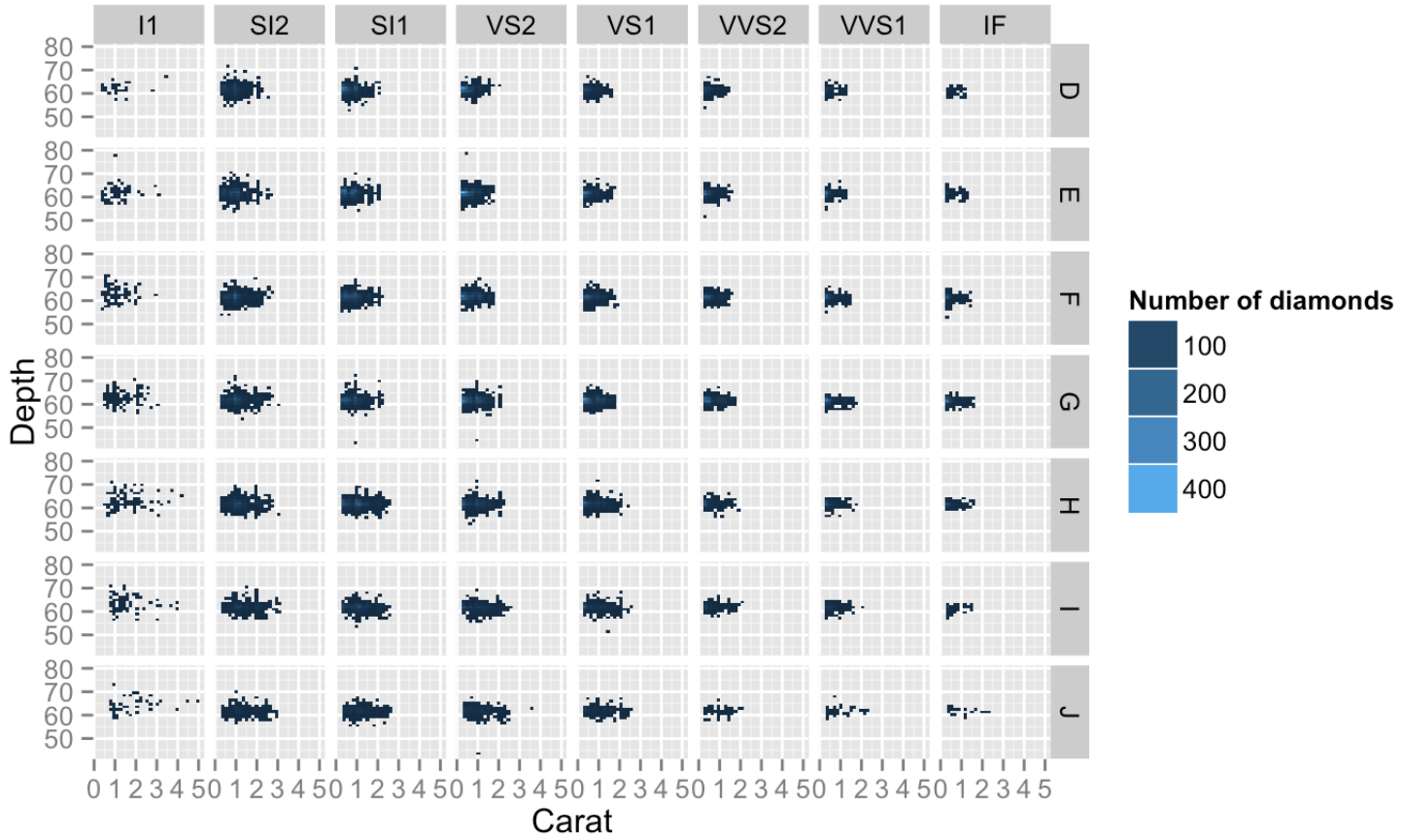
Faceting (wrapping)

```
p2 + facet_wrap(~color)
```



Faceting (grid)

```
p2 + facet_grid(color~clarity)
```



Working with dates

```
# install.packages("lubridate")
# install.packages("dplyr")
library(lubridate)
library(dplyr)
```

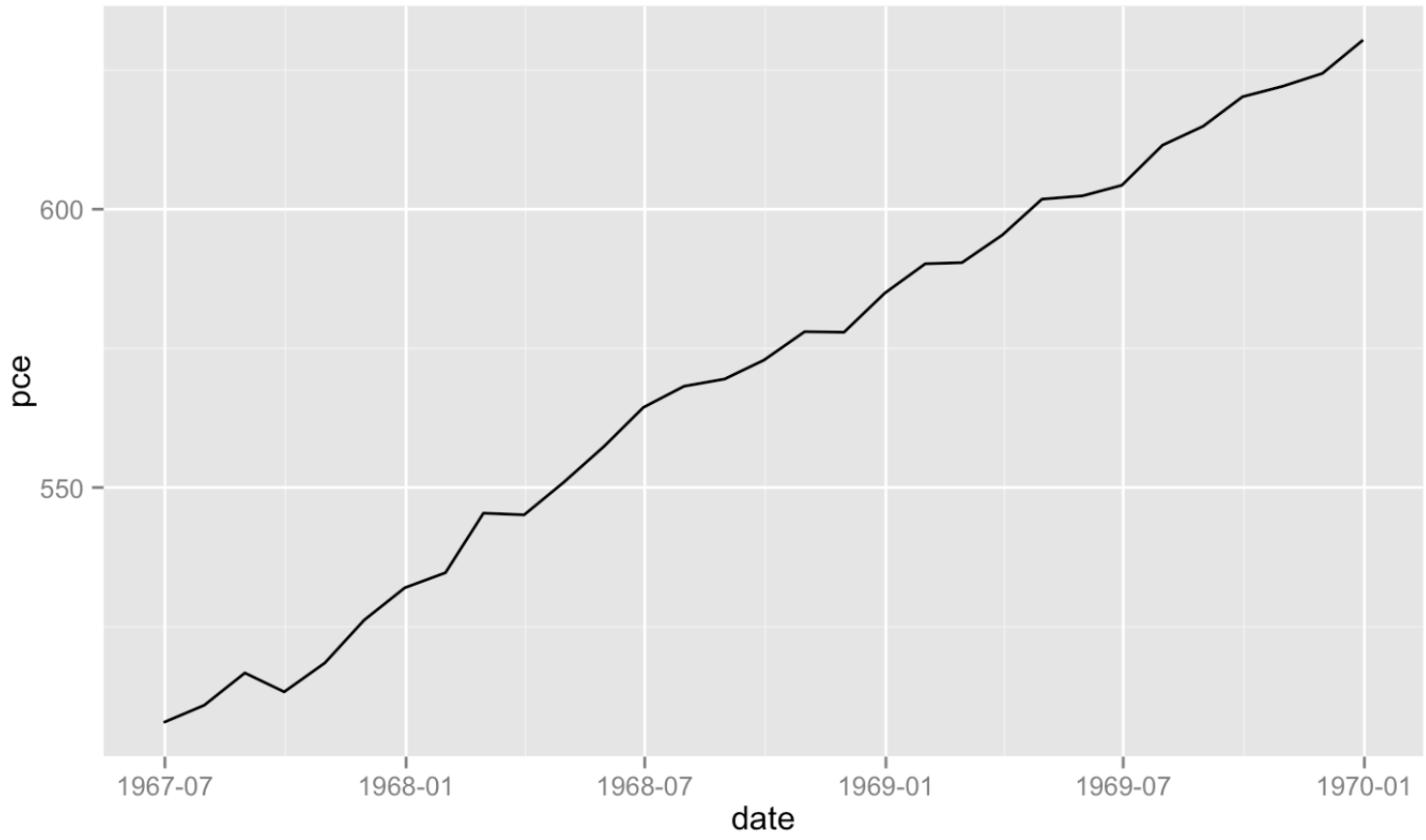
Data with dates

```
econ <- economics
str(econ)
```

```
## 'data.frame':    478 obs. of  6 variables:
##  $ date    : Date, format: "1967-06-30" "1967-07-31" ...
##  $ pce     : num  508 511 517 513 518 ...
##  $ pop     : int  198712 198911 199113 199311 199498 199657 199808 199920 2(
##  $ psavert : num  9.8 9.8 9 9.8 9.7 9.4 9 9.5 8.9 9.6 ...
##  $ uempmed : num  4.5 4.7 4.6 4.9 4.7 4.8 5.1 4.5 4.1 4.6 ...
##  $ unemploy: int  2944 2945 2958 3143 3066 3018 2878 3001 2877 2709 ...
```

```
econ$date <- ymd(econ$date)
old <- filter(econ, year(date)<"1970")
old$date <- ymd(old$date)
```

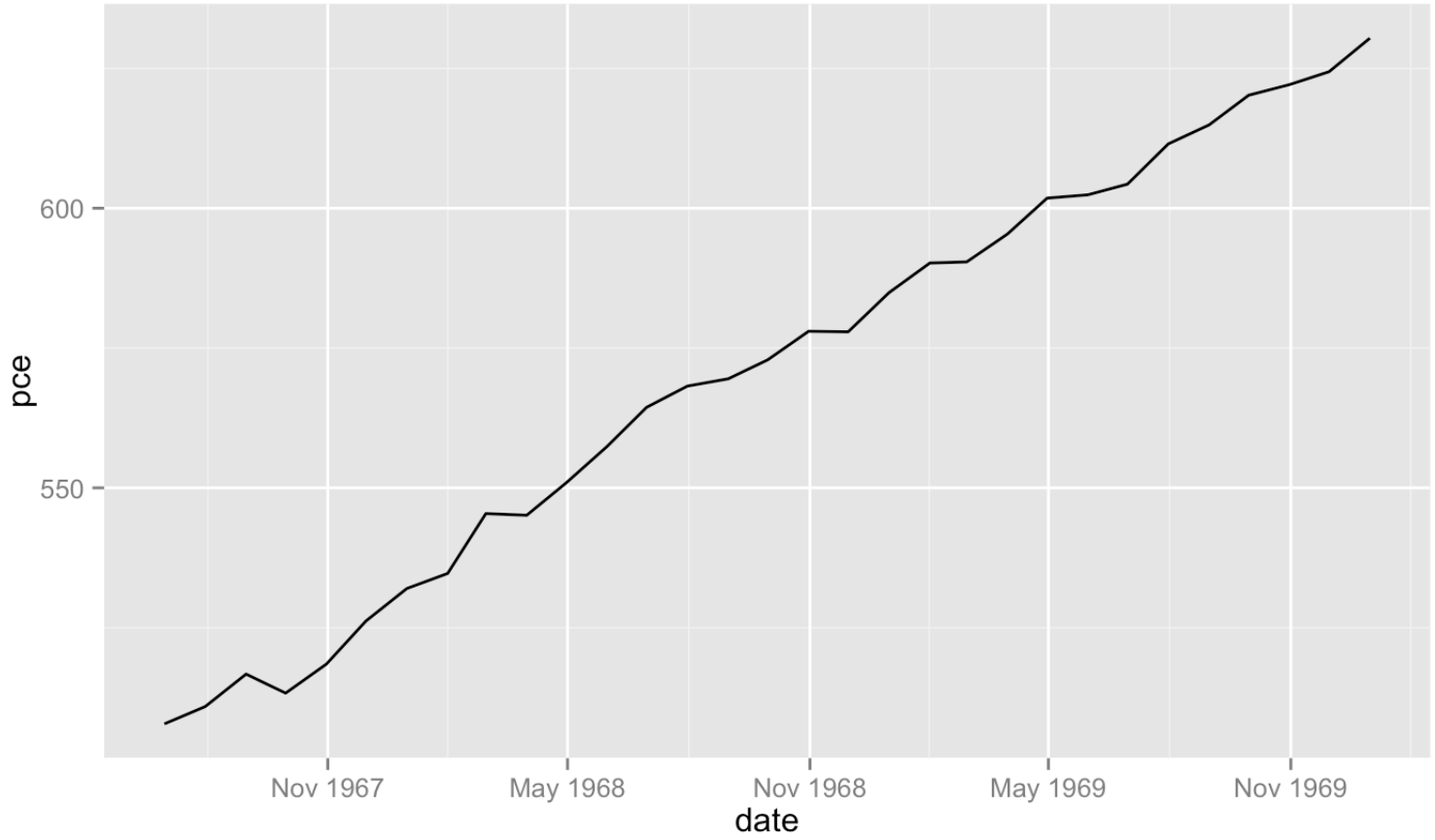
```
p3 <- ggplot(data=old) + geom_line(aes(x=date, y=pce))
p3
```



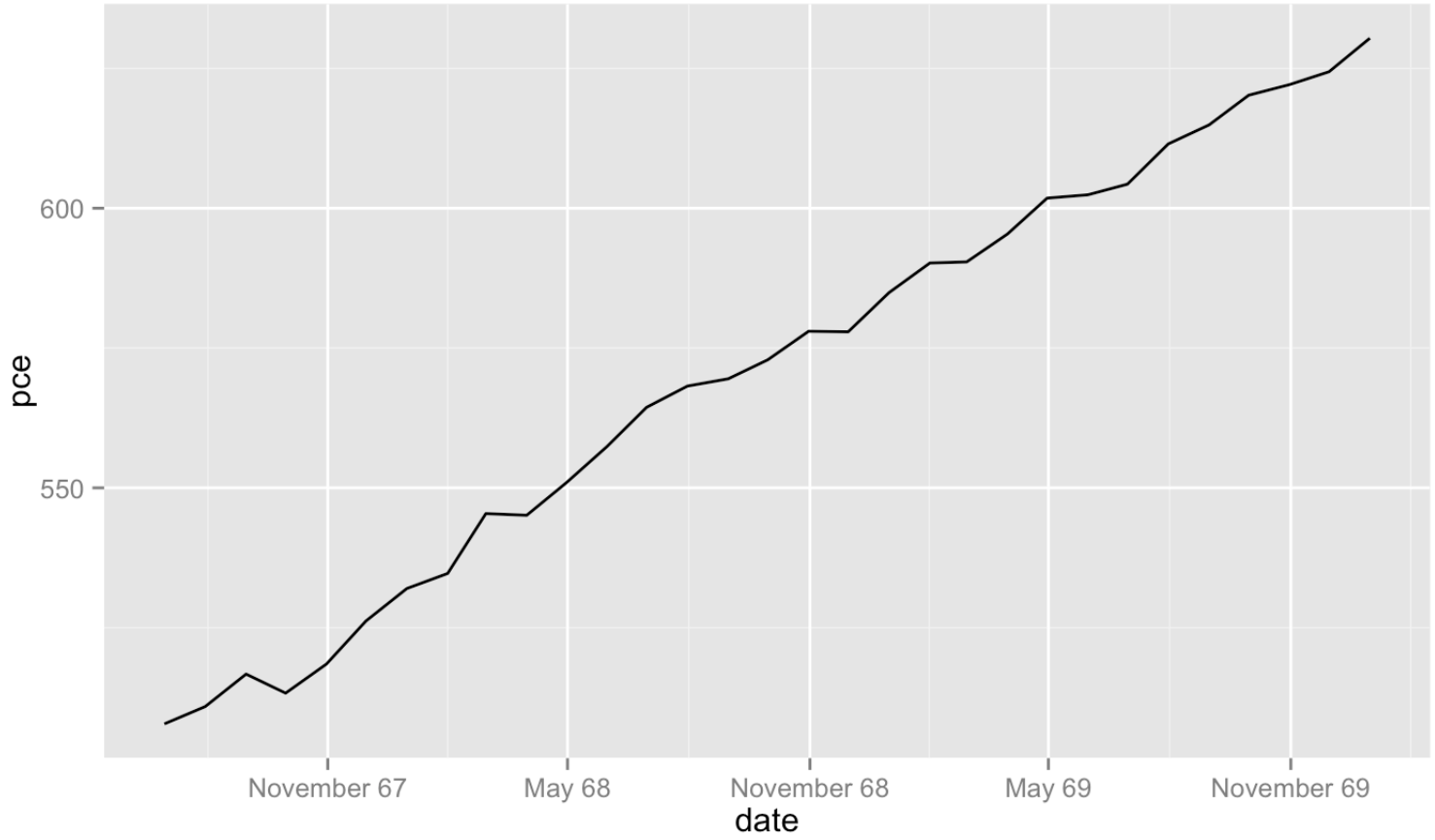
For nicer date formatting

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

```
p3 +
  scale_x_datetime(breaks = date_breaks("6 months"),
                   labels = date_format("%b %Y"))
```



```
p3 +
  scale_x_datetime(breaks = date_breaks("6 months"),
                   labels = date_format("%B %y"))
```



Working with geographic data

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

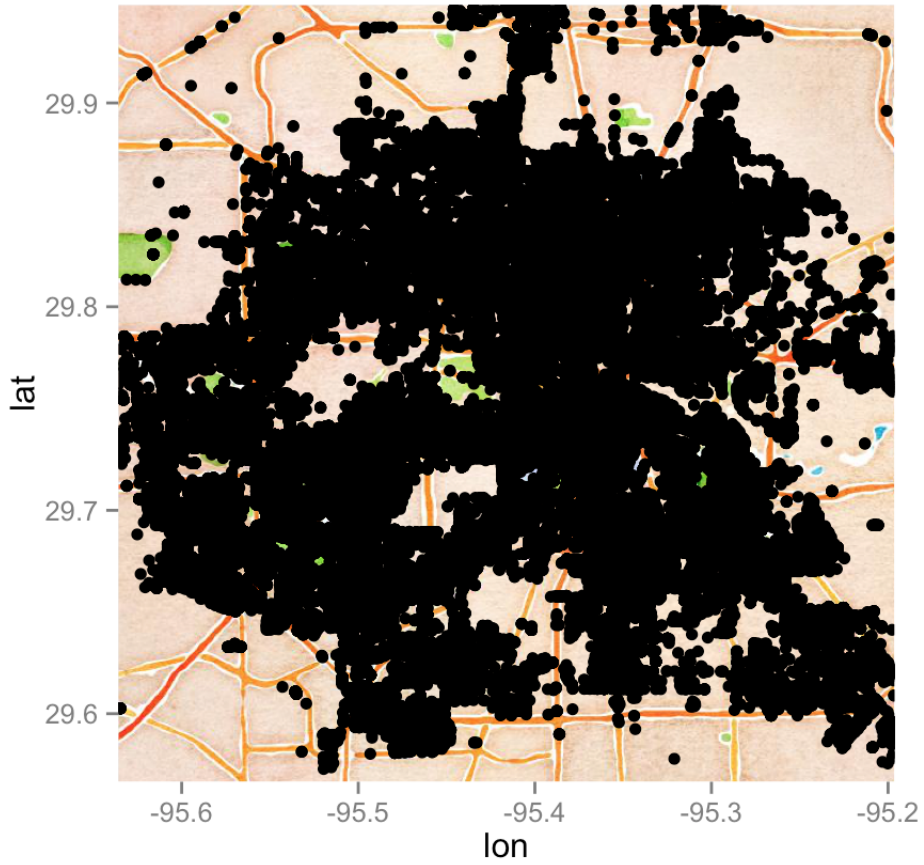
```
str(crime) # crime data from ggmap package
```

```
## 'data.frame':    86314 obs. of  17 variables:
##  $ time    : POSIXt, format: "2009-12-31 22:00:00" "2009-12-31 22:00:00" ..
##  $ date    : chr  "1/1/2010" "1/1/2010" "1/1/2010" "1/1/2010" ...
##  $ hour    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ premise : chr  "18A" "13R" "20R" "20R" ...
##  $ offense : Factor w/ 7 levels "aggravated assault",..: 4 6 1 1 1 3 3 3 3 3
##  $ beat    : chr  "15E30" "13D10" "16E20" "2A30" ...
##  $ block   : chr  "9600-9699" "4700-4799" "5000-5099" "1000-1099" ...
##  $ street  : chr  "marlive" "telephone" "wickview" "ashland" ...
##  $ type    : chr  "ln" "rd" "ln" "st" ...
##  $ suffix  : chr  "-" "-" "-" "-" ...
##  $ number  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ month   : Ord.factor w/ 8 levels "january"<"february"<..: 1 1 1 1 1 1 1 1
##  $ day     : Ord.factor w/ 7 levels "monday"<"tuesday"<..: 5 5 5 5 5 5 5 5 5
##  $ location: chr  "apartment parking lot" "road / street / sidewalk" "reside
##  $ address : chr  "9650 marlive ln" "4750 telephone rd" "5050 wickview ln"
##  $ lon     : num  -95.4 -95.3 -95.5 -95.4 -95.4 ...
##  $ lat     : num  29.7 29.7 29.6 29.8 29.7 ...
```

```
midlat <- mean(crime$lat, na.rm=TRUE)
midlon <- mean(crime$lon, na.rm=TRUE)
m1 <- get_map(source='stamen', location=c(midlon, midlat),
              maptype="watercolor", zoom=11)
p4 <- ggmap(m1)
p4
```

With points

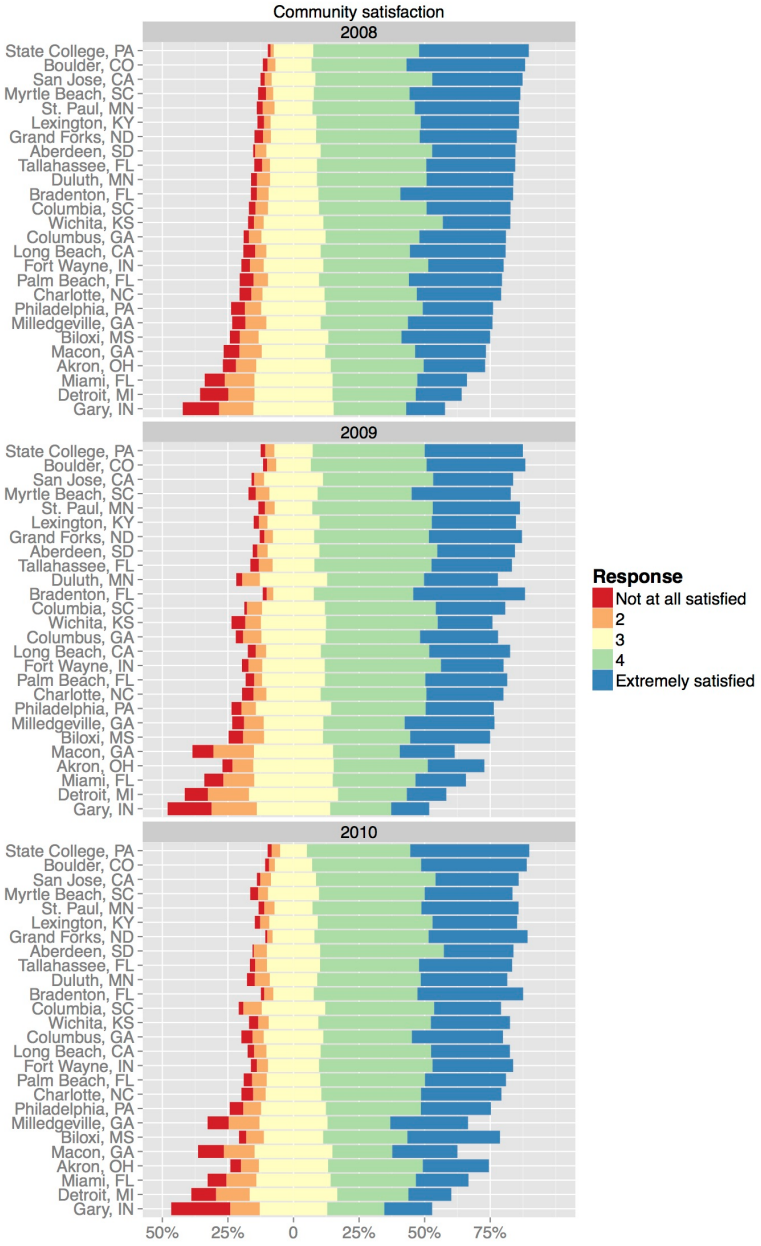
```
p4 + geom_point(aes(x=lon, y=lat), data=crime)
```



```
p4 + geom_hex(aes(x=lon, y=lat), data=crime, alpha=0.8) +
  guides(fill=guide_legend(alpha=0.5, title="Number of crimes")) +
  xlab("") + ylab("") + theme(axis.line=element_blank(),
                              axis.text=element_blank(),
                              axis.ticks=element_blank(),
                              axis.title=element_blank())
```

Many layers

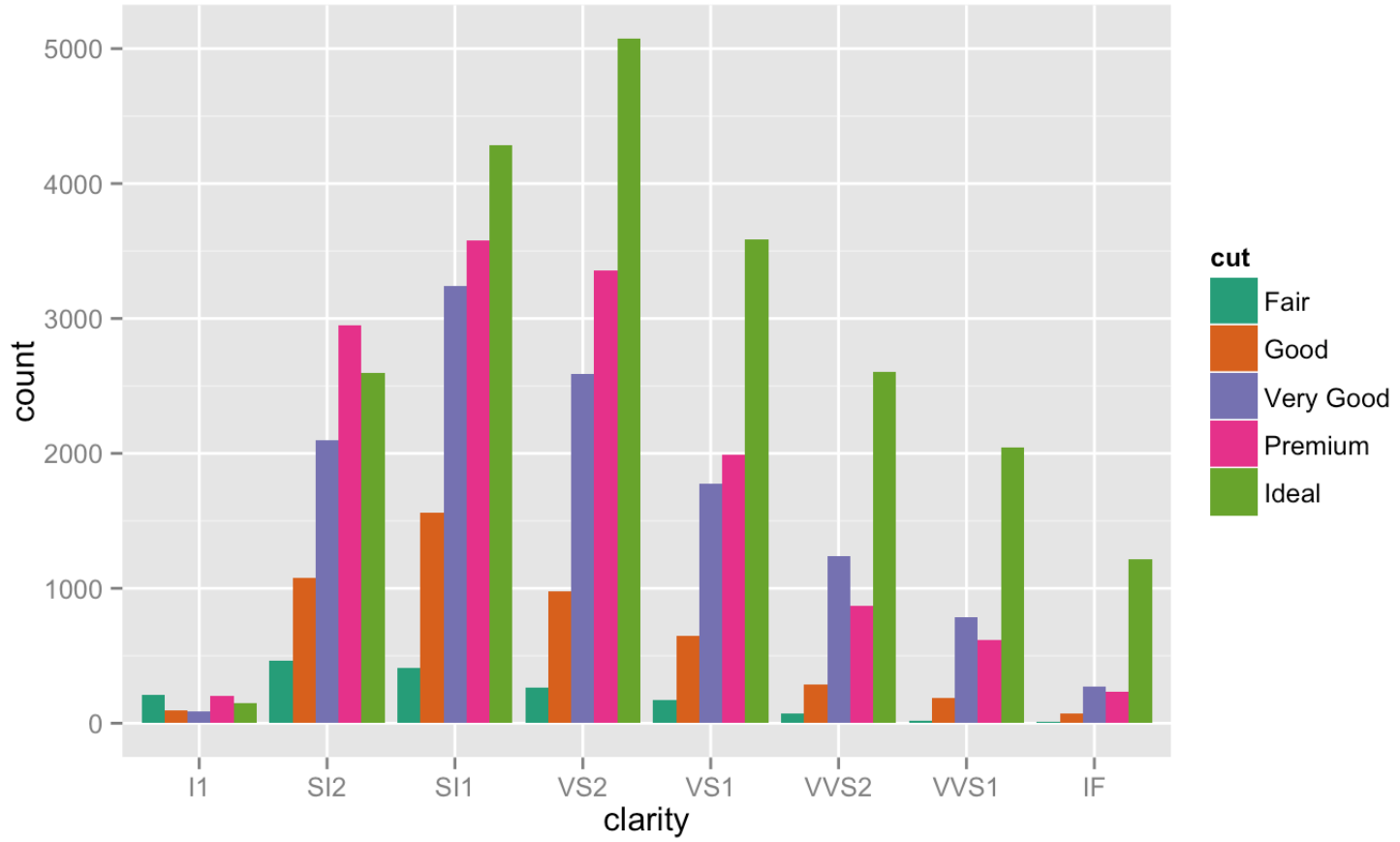
```
baseplot <- ggplot() +
  aes(x=citystate, y=Freq, fill = Response, order=Response) +
  facet_wrap(~year, nrow=3)+geom_bar(data = trial2$neg, stat = "identity") +
  scale_fill_manual(breaks=c("Not at all satisfied", "2", "3", "4",
                             "Extremely satisfied"), values=colorsB,
                    name="Response") +
  geom_bar(data = trial2$pos, stat = "identity") + coord_flip() +
  ggtitle("Community satisfaction") + xlab("") +ylab("") +
  scale_y_continuous(limits=c(-0.5, 1),
                     breaks=seq(from=-0.5, to=0.75, by=0.25),
                     labels=c("50%", "25%", "0", "25%", "50%", "75%")) +
  theme(legend.text=element_text(size=14),
        legend.title=element_text(size=16),
        axis.text=element_text(size=14),
        strip.text=element_text(size=14))
baseplot
```


Community satisfaction
2008
2009
2010
State College, PA
Boulder, CO
San Jose, CA
Myrtle Beach, SC
St. Paul, MN
Lexington, KY
Grand Forks, ND
Aberdeen, SD
Tallahassee, FL
Duluth, MN
Bradenton, FL
Columbia, SC
Wichita, KS
Columbus, GA
Long Beach, CA
Fort Wayne, IN
Palm Beach, FL
Charlotte, NC
Philadelphia, PA
Milledgeville, GA
Biloxi, MS
Macon, GA
Akron, OH
Miami, FL
Detroit, MI
Gary, IN
50%
25%
0
25%
50%
75%
Response
Not at all satisfied
2
3
4
Extremely satisfied

Colors

```
# install.packages("RColorBrewer")
library(RColorBrewer)
display.brewer.all(n=5, type="qual")
```

```
p1 + geom_bar(position="dodge") + scale_fill_brewer(palette="Dark2")
```



Saving your work

```
ggsave("beautiful_hex_map.jpg", plot=p4)
```

Resources for ggplot2

- ggplot2 cheatsheet
- R graphics cookbook