

# EDA

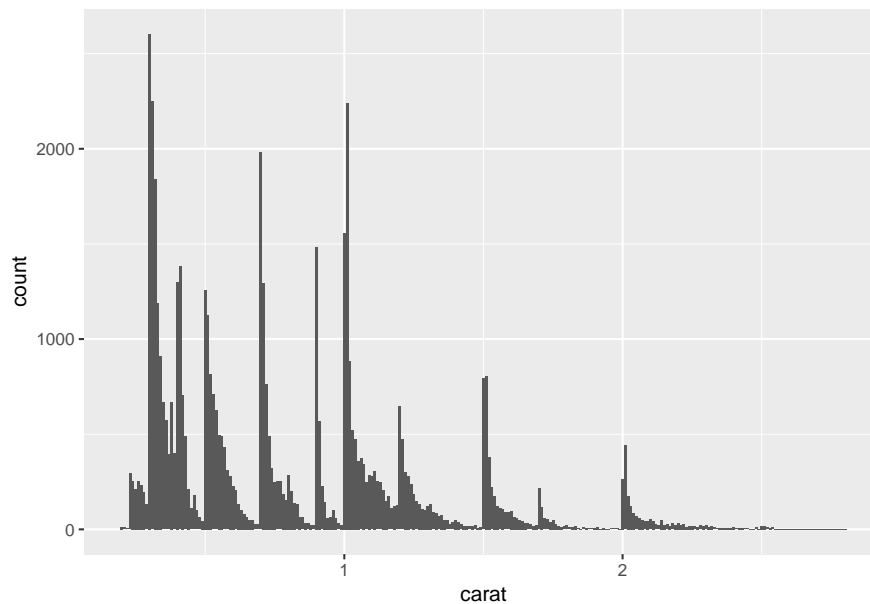
*group 6: Finn Womack, Lorne Curran, Martin Leandro Uranga Priore, Chenyang Duo,  
Emerson Webb, Jiarui Xu*

*5/16/2019*

## Visualizing distributions with `geom_freqpoly()` (7.3.1) - Lorne

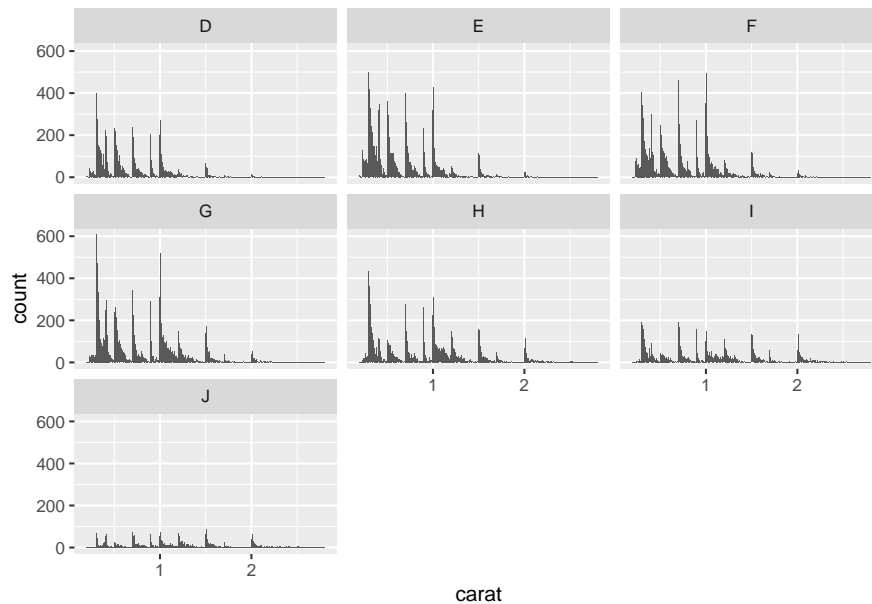
Histograms display the distribution of a continuous variable, preferably with a user-selected bin width:

```
smaller <- diamonds %>%  
  filter(carat < 3)  
  
smaller %>% ggplot(mapping = aes(x = carat)) +  
  geom_histogram(binwidth = 0.01)
```



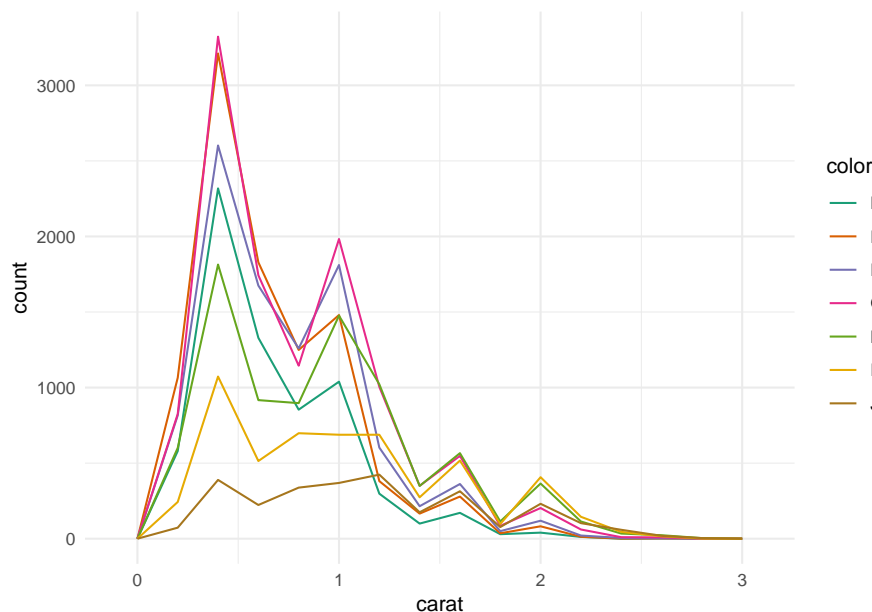
To look at `carat`'s distribution within groups of a categorical co-variable, we might use `facet_wrap`:

```
smaller %>% ggplot(mapping = aes(x = carat)) +  
  geom_histogram(binwidth = 0.01) +  
  facet_wrap(~ color)
```



Or we can employ `geom_freqpoly` to display the distribution of `carat` for each level of `color` as lines on the same plot:

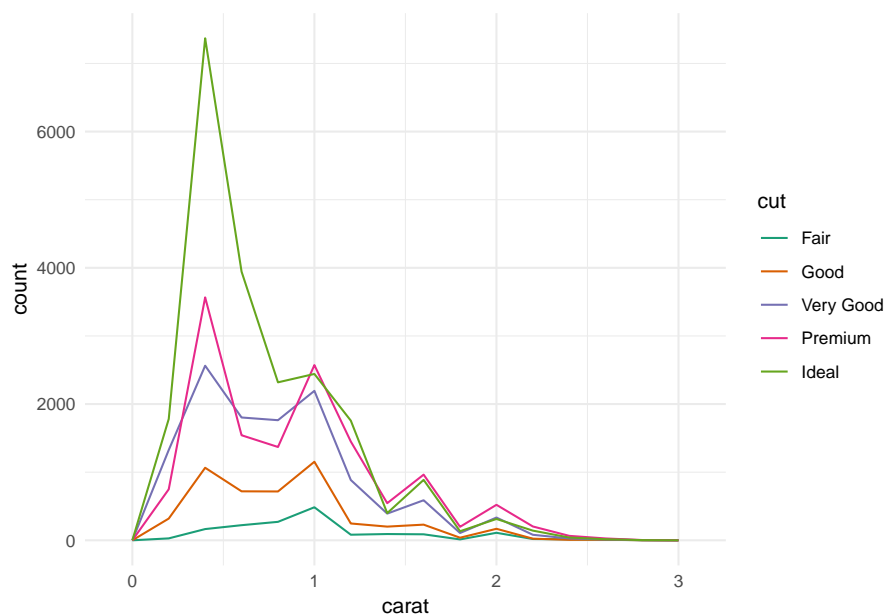
```
smaller %>% ggplot(mapping = aes(x = carat, color = color)) +  
  geom_freqpoly(binwidth = 0.2) + # upping binwidth  
  scale_color_brewer(palette = "Dark2") +  
  theme_minimal()
```



There's 2 problems with this. The number of levels displayed is at the limit of what we may want to present in a single graph. And, despite the increase in bin width, it's hard to distinguish the distribution of a given group when its values are much smaller than others. To avoid the first, we could go back to the `carat` and

cut comparison (below). To solve the 2nd, we'd go on to R4DS 7.5.1 A categorical and continuous variable.

```
smaller %>% ggplot(mapping = aes(x = carat, color = cut)) +  
  geom_freqpoly(binwidth = 0.2) +  
  scale_color_brewer(palette = "Dark2") +  
  theme_minimal()
```

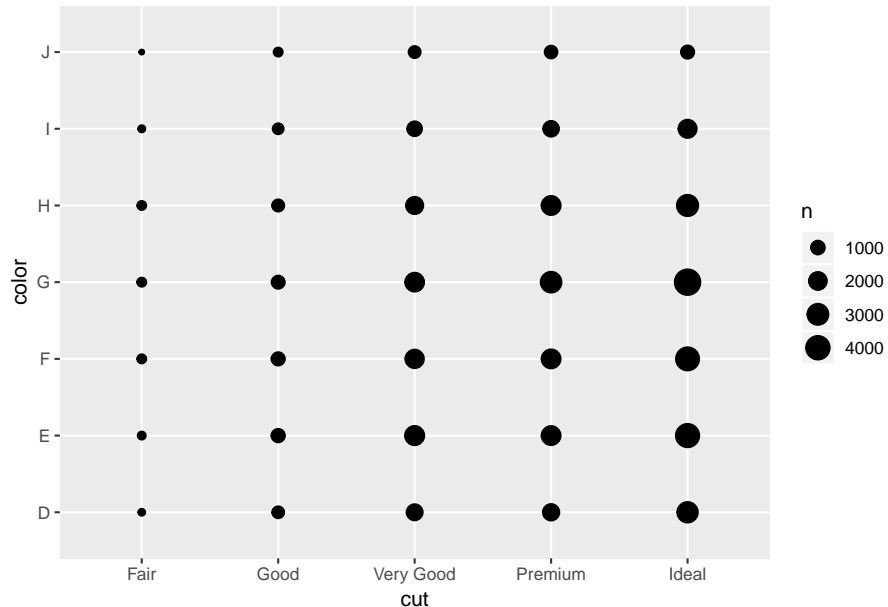


A categorical and continuous variable (7.5.1) - anyone?

## Covariation between two categorical variables (7.5.2) - Martin

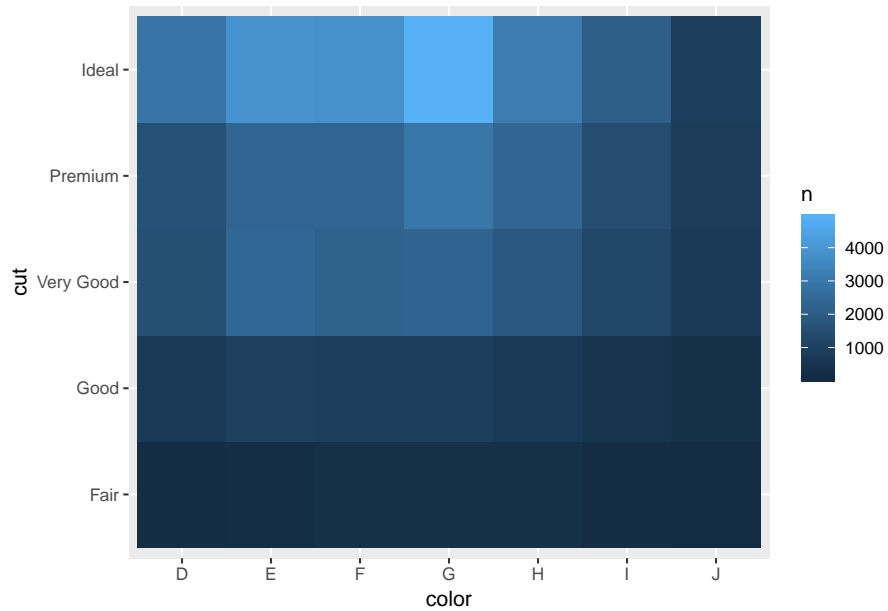
To see how two categorical variables are related we can count the number of observations for each level combination of both variables. We can do this using *geom\_count* like in the following example:

```
ggplot(data = diamonds) +  
  geom_count(mapping = aes(x = cut, y = color))
```



This graph has the disadvantage that we have to interpret the size of the dots which unless the difference between to neighbors is huge it can be difficult to assess the magnitude. One option, it to use *count* of *ddply* and then use the function *geom\_tile* which make the visualization easier to interpret since use colors to display the covariation between both variables.

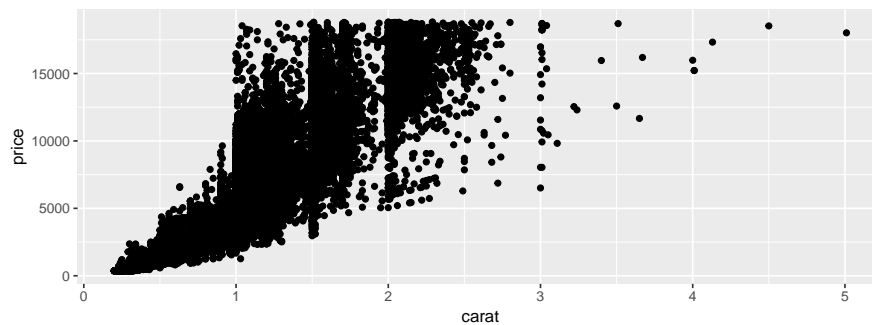
```
diamonds %>%  
  count(color, cut) %>%  
  ggplot(mapping = aes(x = color, y = cut)) +  
    geom_tile(mapping = aes(fill = n))
```



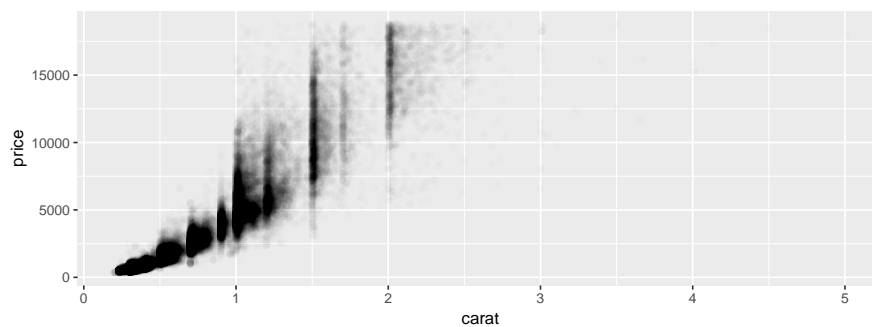
## Covariation between two continuous variables (7.5.2) (7.6) - Finn

We've already used scatterplots to examine the covariance between two continuous variables which tend to get cluttered with a lot of data. In the past we've added transparency using `alpha` to combat this when examining scatterplots with a lot of data:

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



```
ggplot(data = diamonds) +  
  geom_point(mapping = aes(x = carat, y = price), alpha = 1 / 100)
```

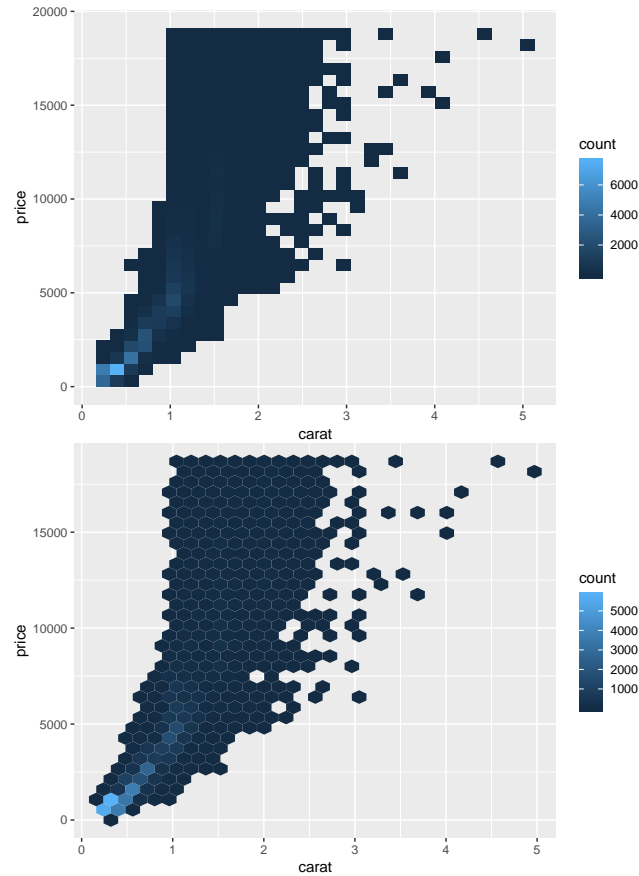


However, this technique of adjusting `alpha` can sometimes be difficult to use. If the data is too large and

transparency proved challenging then another possible way of exploring covariance between two continuous variables, which might be useful in this case, is to put them into bins and adjusting the color instead of transparency by using the `geom_bin2d()` and `geom_hex()` functions.

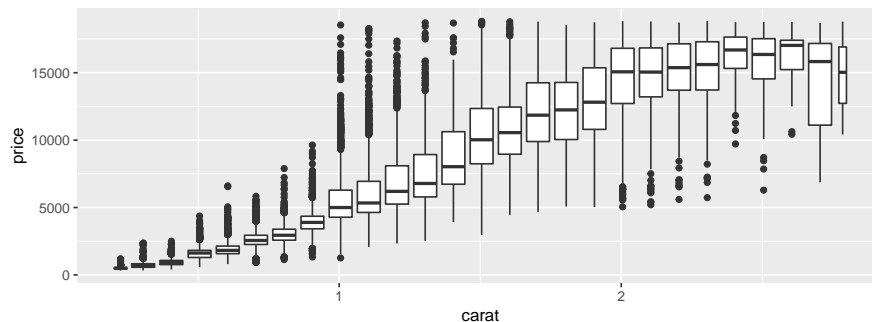
```
ggplot(data = diamonds) +  
  geom_bin2d(mapping = aes(x = carat, y = price))
```

```
ggplot(data = diamonds) +  
  geom_hex(mapping = aes(x = carat, y = price))
```



The binning strategy can also be selectively applied to just one of the variables then you can plot them as if one were continuous and the other categorical.

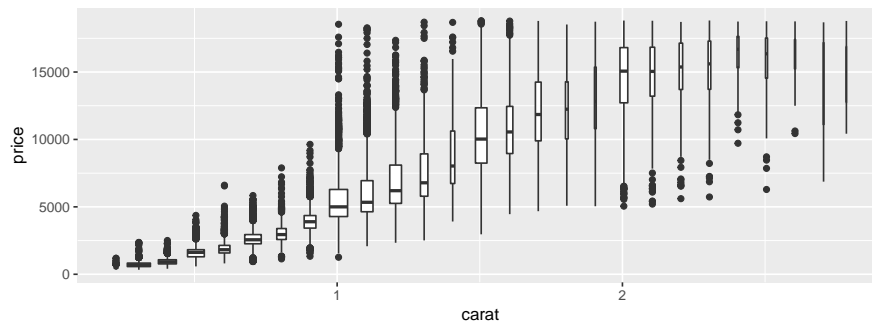
```
ggplot(data = smaller, mapping = aes(x = carat, y = price)) +  
  geom_boxplot(mapping = aes(group = cut_width(carat, 0.1)))
```



If we want to also have an indicator as to how many points each plot is summarizing we can use `varwidth =`

TRUE.

```
ggplot(data = smaller, mapping = aes(x = carat, y = price)) +  
  geom_boxplot(mapping = aes(group = cut_width(carat, 0.1)), varwidth = TRUE)
```



This way we can compare the boxes size to see which ones are summarizing smaller or larger sets of datapoints. Since in this case some of the boxes can be hard to see another way of dealing with this problem is to instead divide the variable into equally sized partitions for each bin with the size of the bins instead dictating the range over the axis (since they now all have roughly the same number of data points) using `cut_number()`.

```
ggplot(data = smaller, mapping = aes(x = carat, y = price)) +  
  geom_boxplot(mapping = aes(group = cut_number(carat, 20)))
```

