L04 Exploratory Data Analysis

Data Science I (STAT 301-1)

Shay Lebovitz

Contents

Overview **1**

Datasets **1**

Exercises **1**

Overview

The goal of this lab is to begin to use data visualization and transformation skills to explore data in a systematic way. We call this exploratory data analysis (EDA).

Datasets

This lab utilizes the `diamonds` and `flights` datasets contained in packages `ggplot2` and `nycflights13`, respectively. Documentation/codebook can be accessed with `?diamonds` and `?flights`, provided you have installed and loaded `nycflights13` and `tidyverse` to your current R session.

Exercises

Please complete the following exercises. Be sure your solutions are clearly indicated and that the document is neatly formatted.

```
library(tidyverse)
library(nycflights13)
library(patchwork)
library(ggstance)
library(lvplot)
```

Load Packages

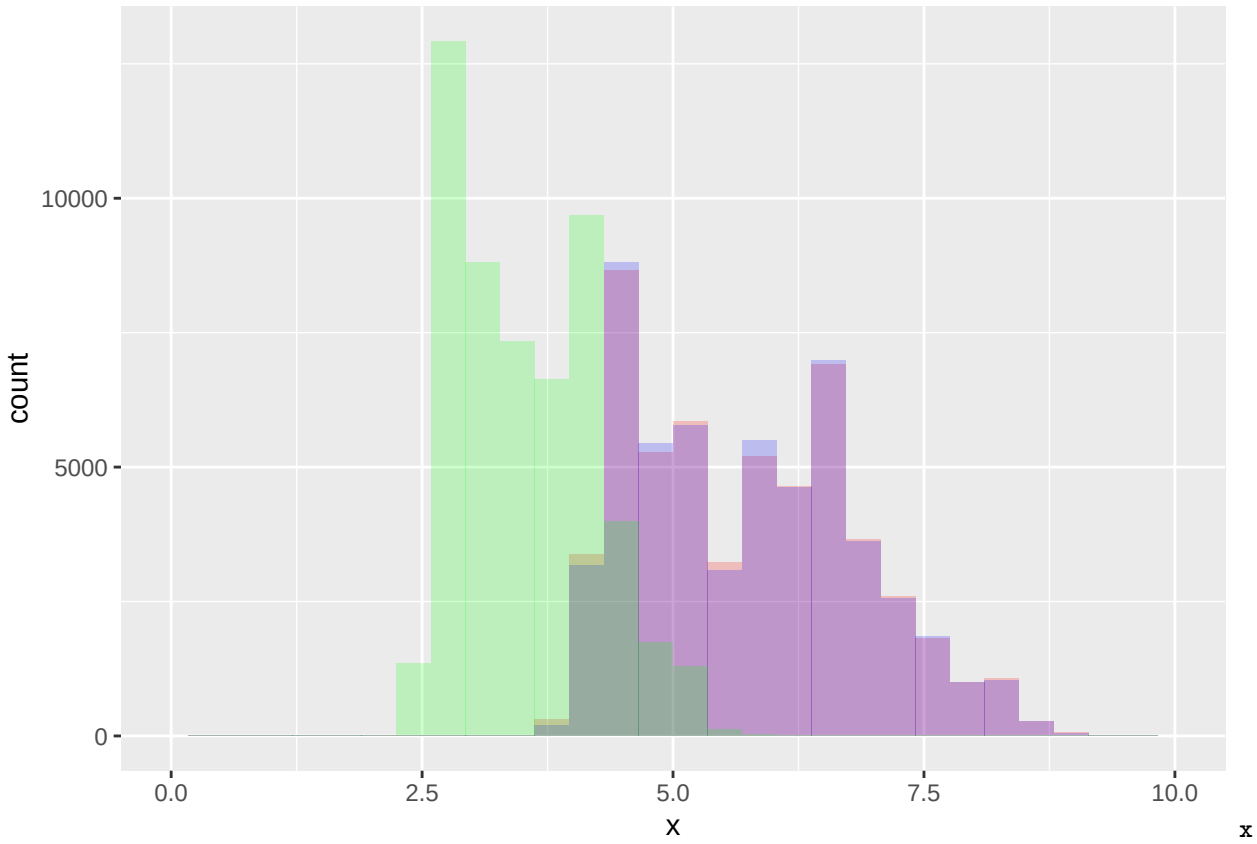
Exercise 1 There are a lot of resources out there to get help and insights into using RStudio. Please visit and explore the following:

- RStudio Tips twitter account (consider following): https://twitter.com/rstudiotips

- RStudio diagnostic report (consider changing your defaults): https://support.rstudio.com/hc/en-us/articles/205753617-Code-Diagnostics
- RStudio Community (consider joining/setting up an account): https://community.rstudio.com/

Exercise 2 (Website: 7.3.4 Ex. 1) Explore the distribution of each of the `x`, `y`, and `z` variables in `diamonds`. What do you learn? Think about a diamond; think about how we might determine which dimension of a diamond is its length, width, and depth.

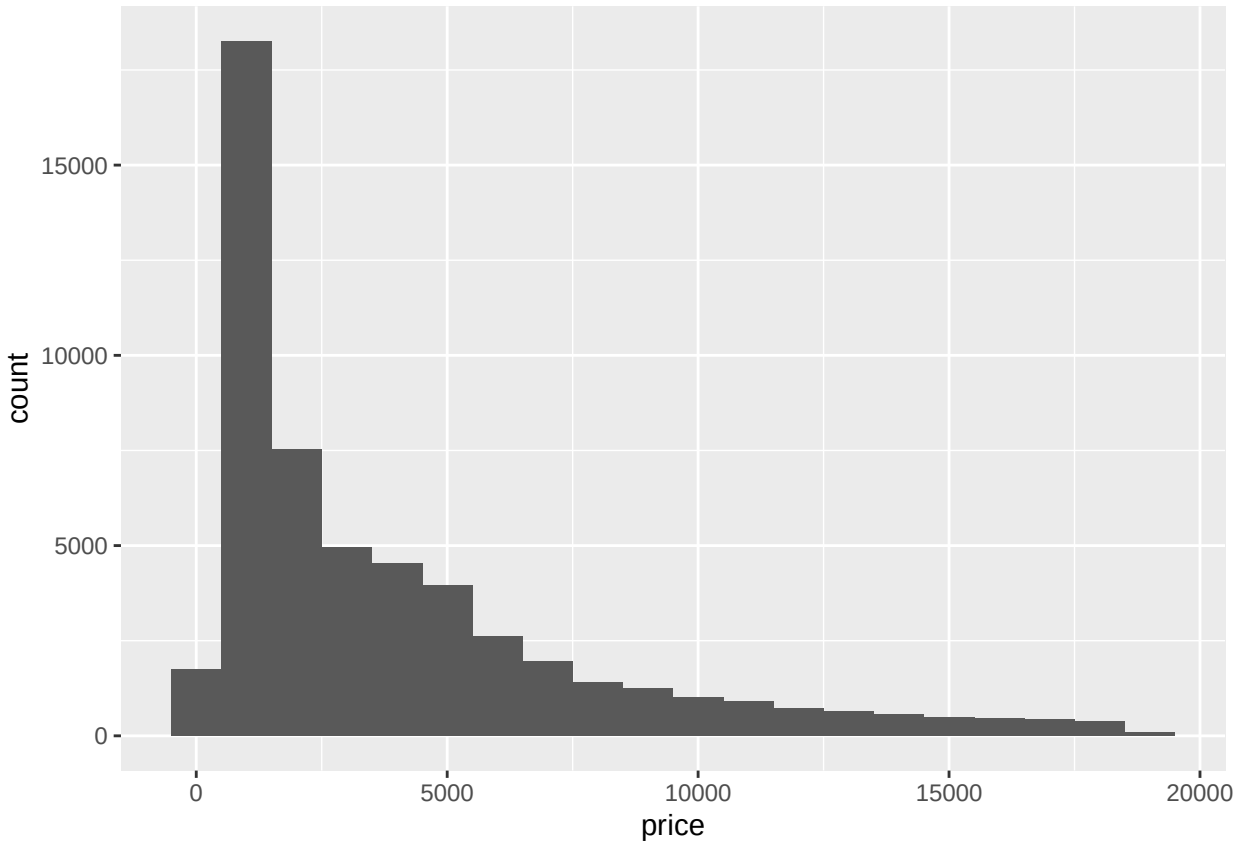
```
ggplot(data = diamonds) +
  geom_histogram(mapping = aes(x), fill = 'red', alpha = 0.2, bins = 30) +
  geom_histogram(mapping = aes(y), fill = 'blue', alpha = 0.2, bins = 30) +
  geom_histogram(mapping = aes(z), fill = 'green', alpha = 0.2, bins = 30) +
  xlim(0,10)
```



and y are practically the same, from about 4 - 9, whereas z is smaller from 2.5 - 5. This tells me that x and y are the interchangeable width and heigh, and z is the depth of the diamond

Exercise 3 (Website: 7.3.4 Ex. 2) Explore the distribution of `price`. Do you discover anything unusual or surprising? (Hint: Carefully think about the `binwidth` and make sure you try a wide range of values.)

```
ggplot(diamonds) +
  geom_histogram(mapping = aes(x = price), binwidth = 1000)
```



The vast majority are under 5000, most under 2000, with a long right tail.

Exercise 4 (Website: 7.3.4 Ex. 3) How many diamonds are 0.99 carat? How many are 1 carat? What do you think is the cause of the difference?

```
diamonds %>%
  count(carat == 1)
```

```
## # A tibble: 2 x 2
##   `carat == 1`     n
##   <lgl>        <int>
## 1 FALSE        52382
## 2 TRUE          1558
```

```
diamonds %>%
  count(carat == 0.99)
```

```
## # A tibble: 2 x 2
##   `carat == 0.99`     n
##   <lgl>           <int>
## 1 FALSE           53917
## 2 TRUE               23
```

1558 diamonds with `carat` of 1. 23 diamonds with `carat` of 0.99. They probably just round up to one or don't measure that precisely.

Exercise 5 (Website: 7.3.4 Ex. 4 — modified) What is the major difference between using `coord_cartesian()` vs `xlim()` or `ylim()` to zoom in on a histogram/graphic? **`xlim` and `ylim` cut off**

values outside of the range, `coord_cartesian` zooms in to see the small values.

Exercise 6 (Website: 7.4.1 Ex. 1) What happens to missing values in a histogram? What happens to missing values in a bar chart? Why is there a difference? **In `geom_histogram()`, missing values are removed, whereas in `geom_bar()`, missing values are added into their own category. This is because histograms must have a numerical x-variable whereas bar charts don't have to.**

Exercise 7 (Website: 7.4.1 Ex. 2) What does `na.rm = TRUE` do in `mean()` and `sum()`? What happens when it is not included and 'NA' values are present?

```
x = c(1, 4, 6, NA, 7)
(sum(x, na.rm = TRUE))
```

```
## [1] 18
```

```
(sum(x, na.rm = FALSE))
```

```
## [1] NA
```

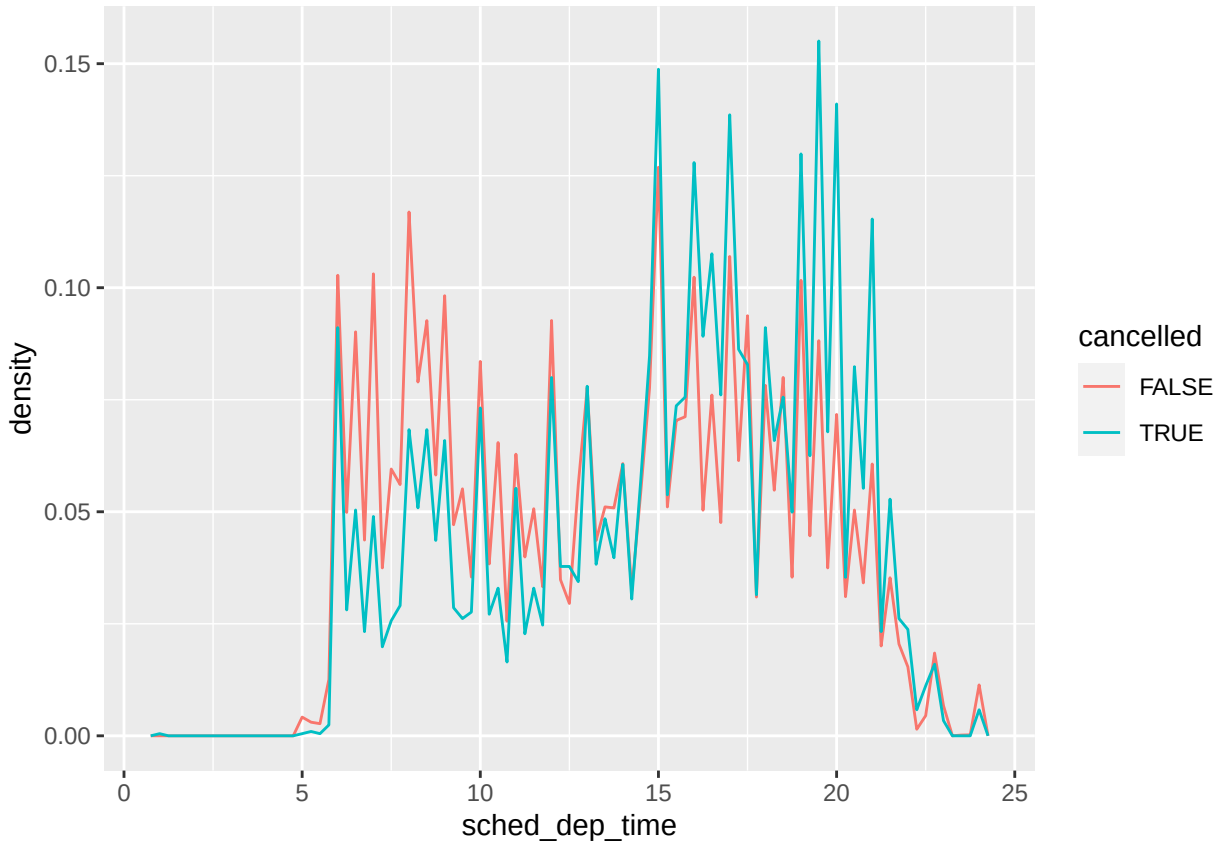
```
(mean(x, na.rm = FALSE))
```

```
## [1] NA
```

`na.rm = TRUE` removes `NA`s from the calculation of `sum()` and `mean()`. Without it, the result for the sum and the mean will be `NA`, as adding an `NA` to anything will result in an `NA`.

Exercise 8 (Website: 7.5.1.1 Ex. 1) Use what you've learned in *7.5.1 A Categorical and continuous variable* to improve this visualization of the departure times of cancelled vs. non-cancelled flights.

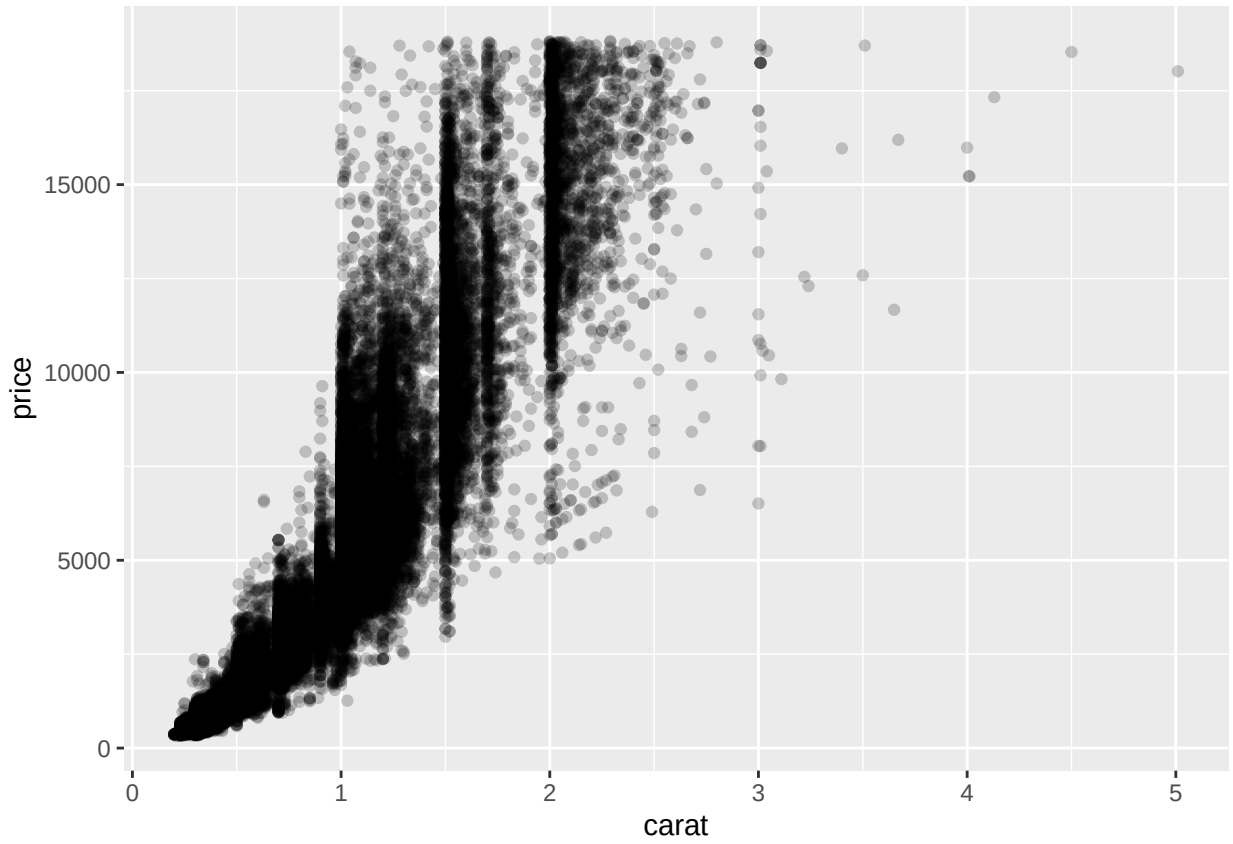
```
flights %>%
  mutate(
    cancelled = is.na(dep_time),
    sched_hour = sched_dep_time %/% 100,
    sched_min = sched_dep_time %% 100,
    sched_dep_time = sched_hour + sched_min / 60
  ) %>%
  ggplot(mapping = aes(x = sched_dep_time, y = ..density..)) +
    geom_freqpoly(mapping = aes(color = cancelled), binwidth = 1/4)
```



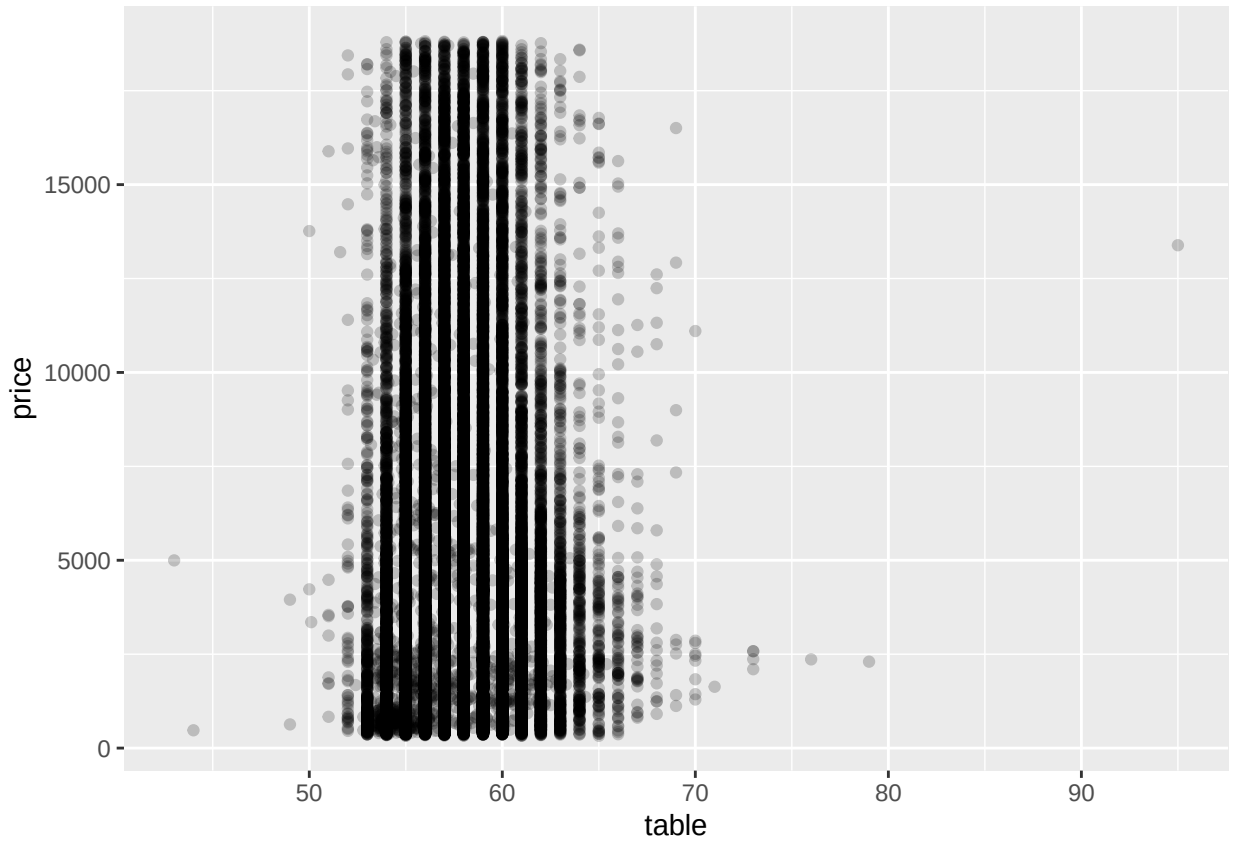
Because the count of cancelled flights is so much lower than that of non-cancelled flights, it is difficult to see the trend in cancelled flights due to scale. Using `..density..` instead of the default `count` ensures that the areas under the polygons are both one, thus normalizing the size of the data in the plot.

Exercise 9 (Website: 7.5.1.1 Ex. 2) What variable in the `diamonds` dataset is most important for predicting the `price` of a diamond? How is that variable correlated with `cut`? Why does the combination of those two relationships result in lower quality diamonds being more expensive?

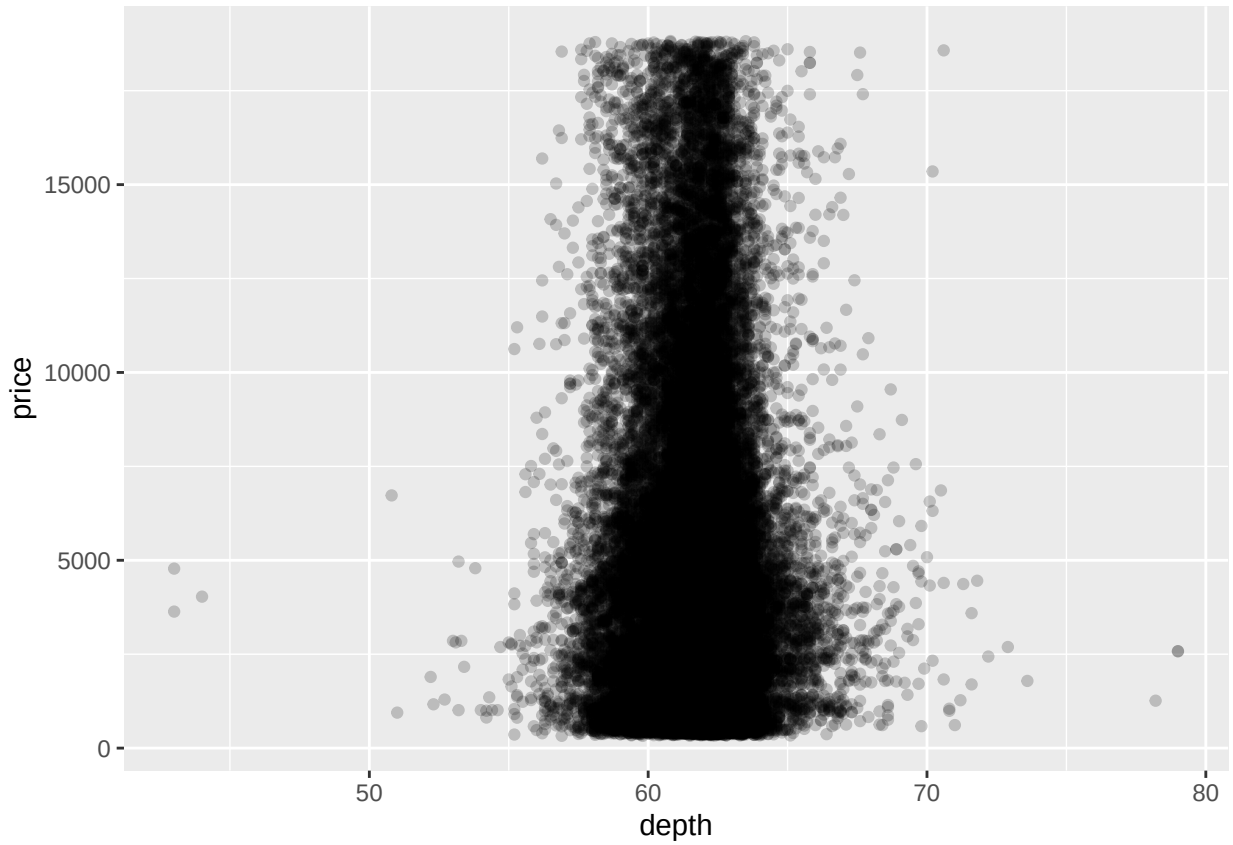
```
ggplot(data = diamonds) +
  geom_point(mapping = aes(x = carat, y = price), alpha = 0.2)
```



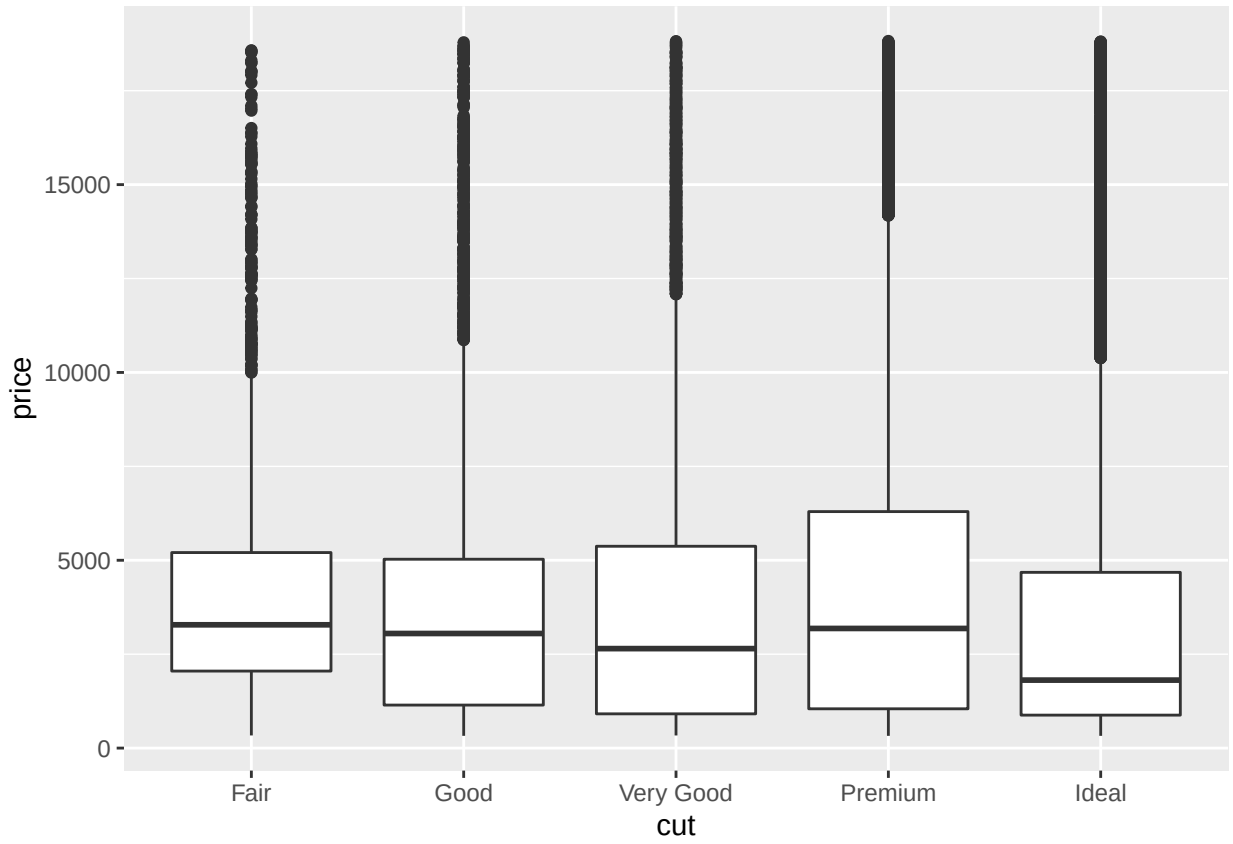

```
ggplot(data = diamonds) +
  geom_point(mapping = aes(x = table, y = price), alpha = 0.2)
```



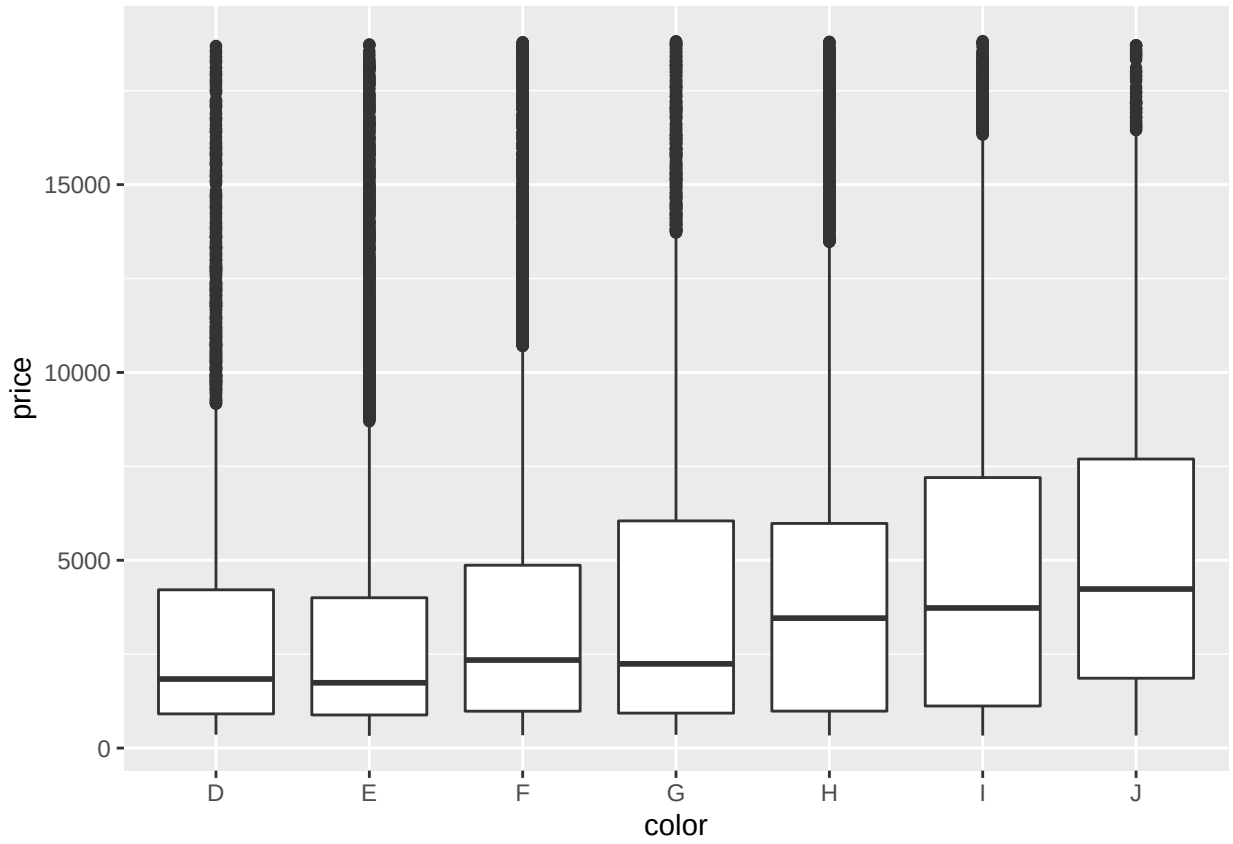

```
ggplot(data = diamonds) +
  geom_point(mapping = aes(x = depth, y = price), alpha = 0.2)
```



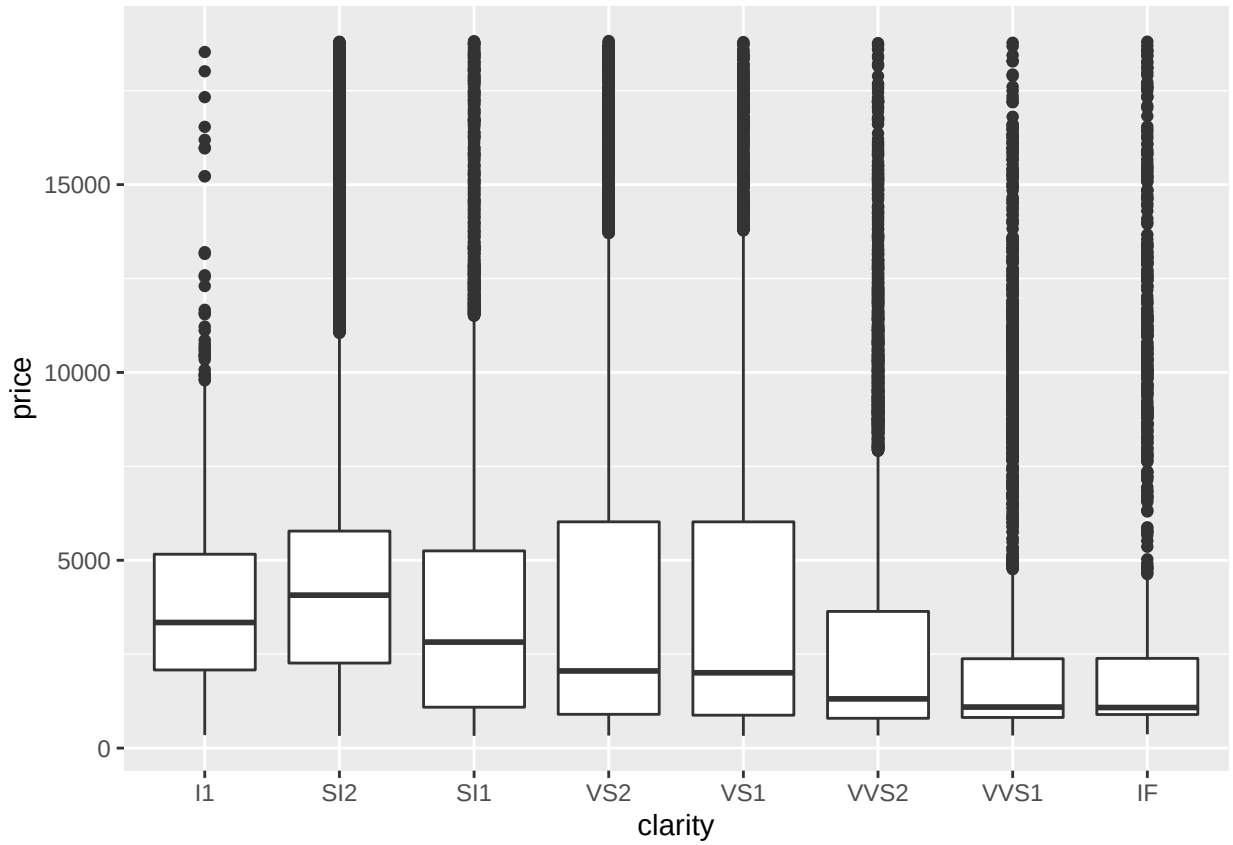

```
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = cut, y = price))
```



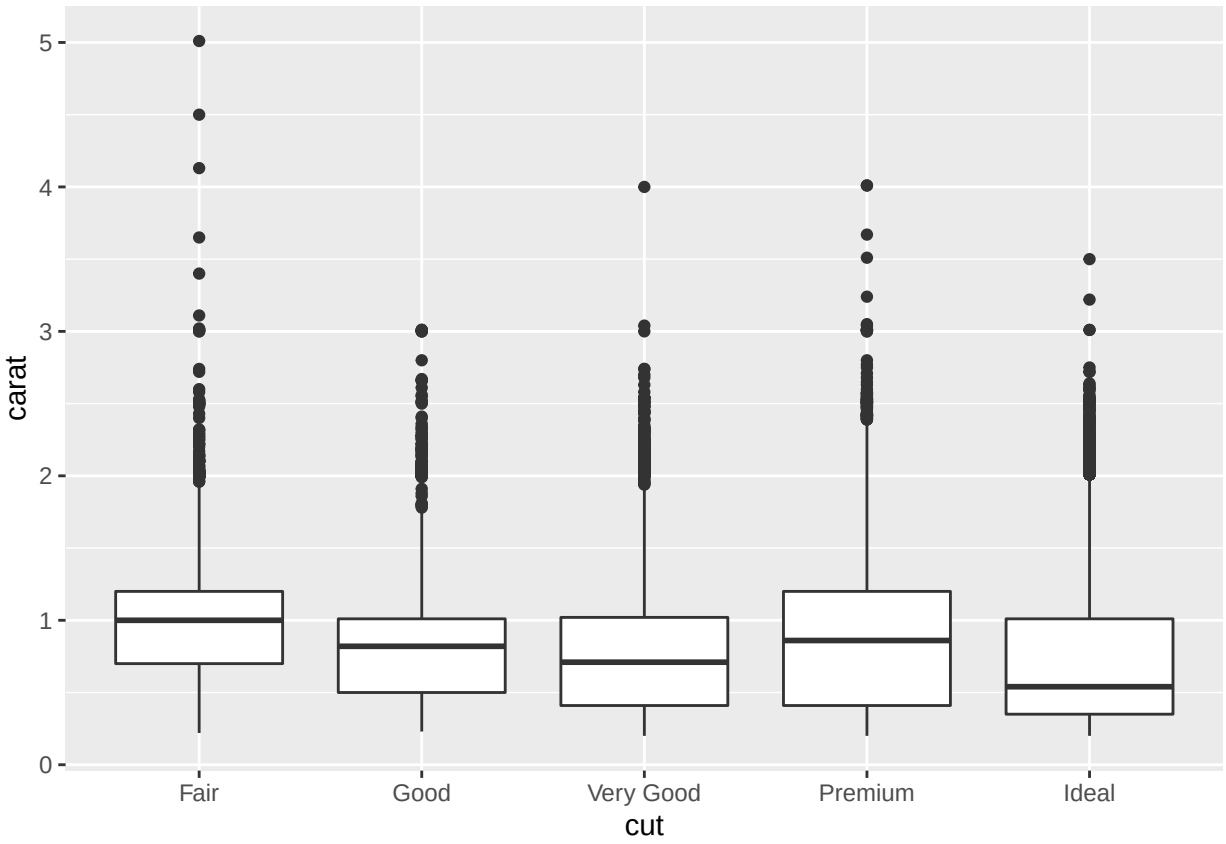

```
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = color, y = price))
```




```
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = clarity, y = price))
```



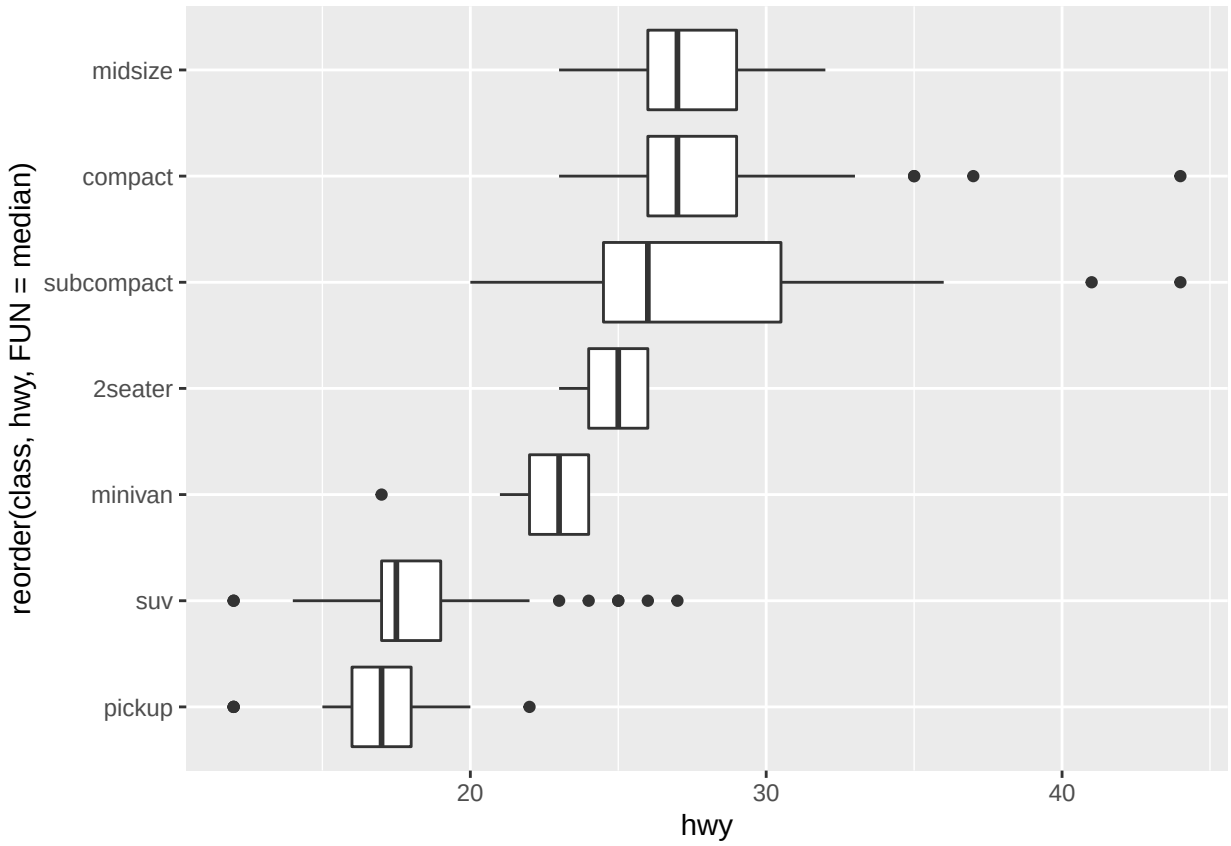

```
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = cut, y = carat))
```



`price` is most strongly correlated with `carat`. `carat` is negaitvely correlated with `cut`, with 'Fair' diamonds having the highest avgerage `carat` value and 'Ideal' diamonds having the lowest. Thus, even though we'd expect higher quality diamonds to be more expensive, we see that lower quality are larger, and thus more expensive.

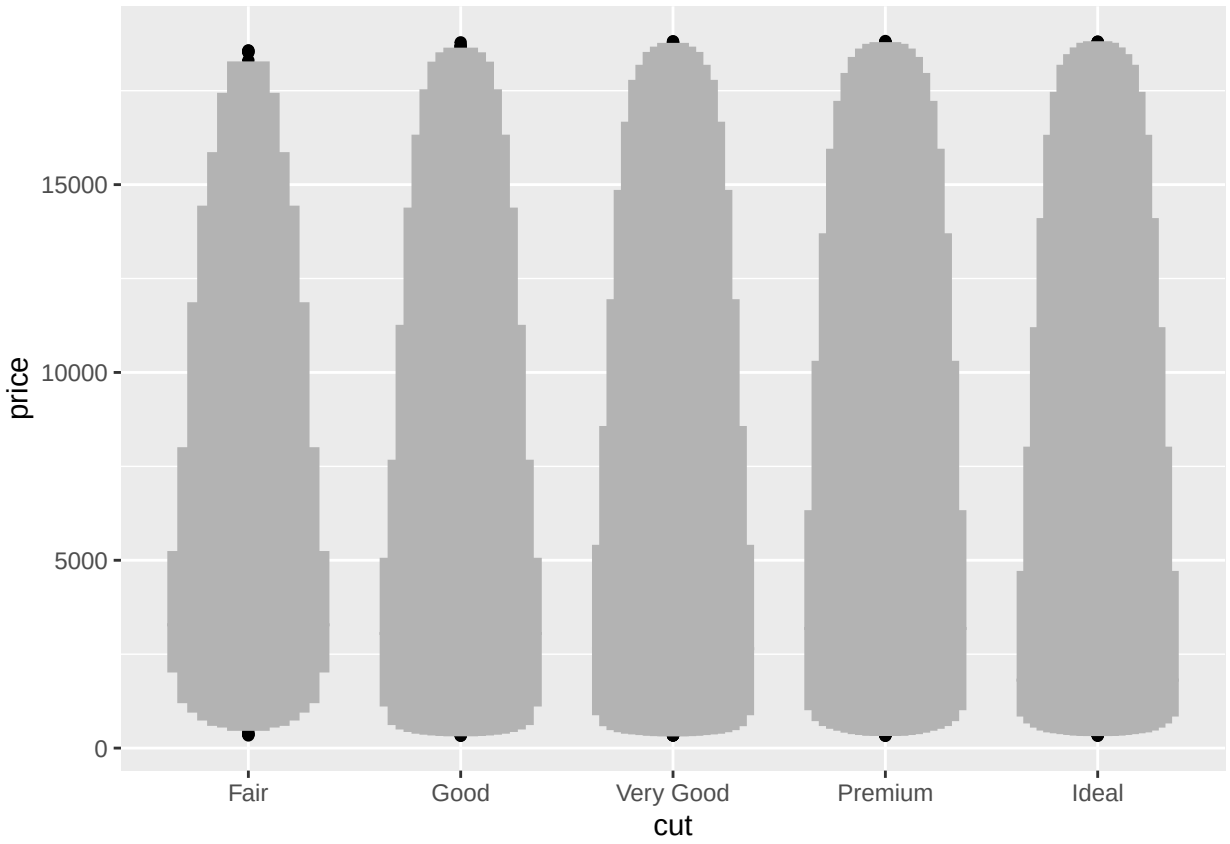
Exercise 10 (Website: 7.5.1.1 Ex. 3 — modified) Install the `ggstance` package, and recreate the horizontal boxplot below without using `coord_flip()`.

```
ggplot(data = mpg) +
  geom_boxploth(mapping = aes(x = hwy, y = reorder(class, hwy, FUN = median)))
```



Exercise 11 (Website: 7.5.1.1 Ex. 4) One problem with boxplots is that they were developed in an era of much smaller datasets and tend to display a prohibitively large number of "outlying values." One approach to remedy this problem is the letter value plot. Install the `lvplot` package and try using `geom_lv()` to display the distribution of `price` vs `cut`. What do you learn? How do you interpret the plots? Check out this paper

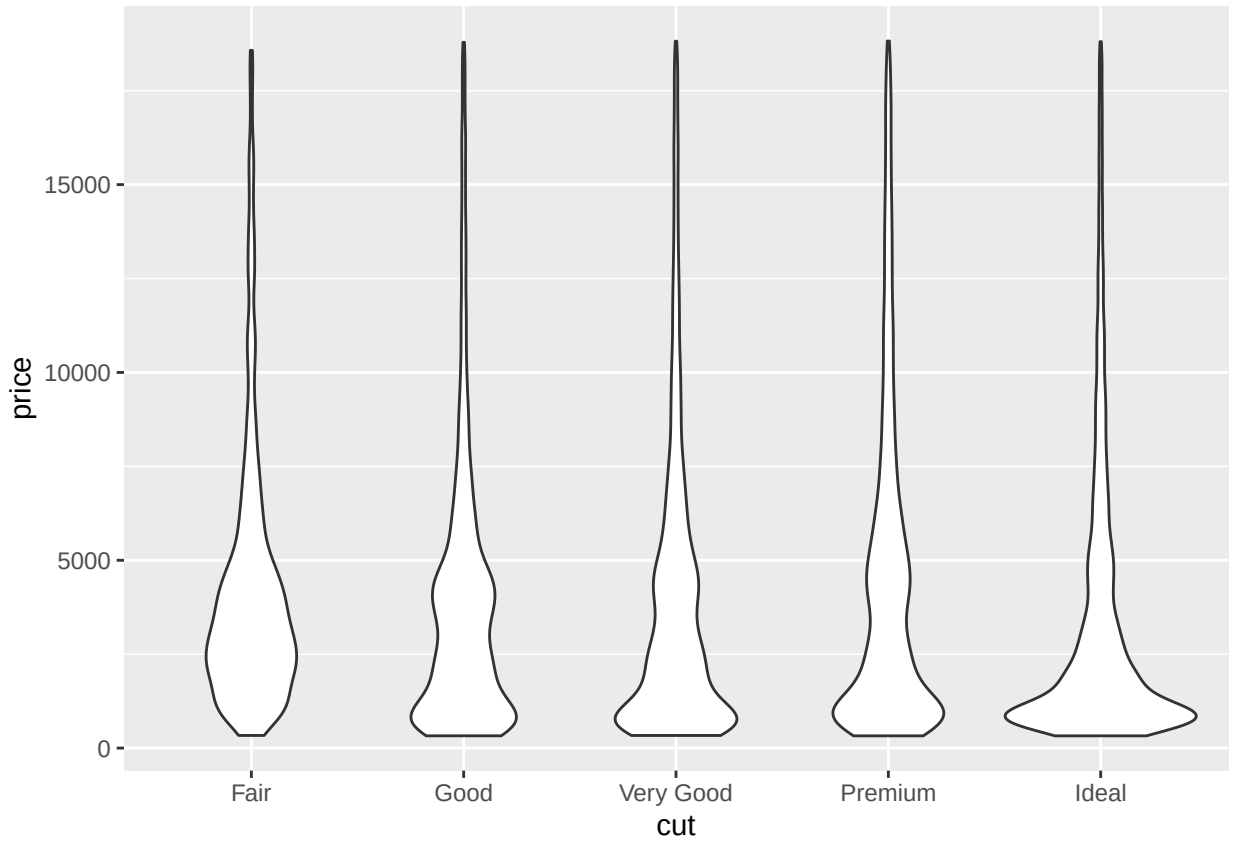
```
ggplot(data = diamonds) +
  geom_lv(mapping = aes(x = cut, y = price))
```



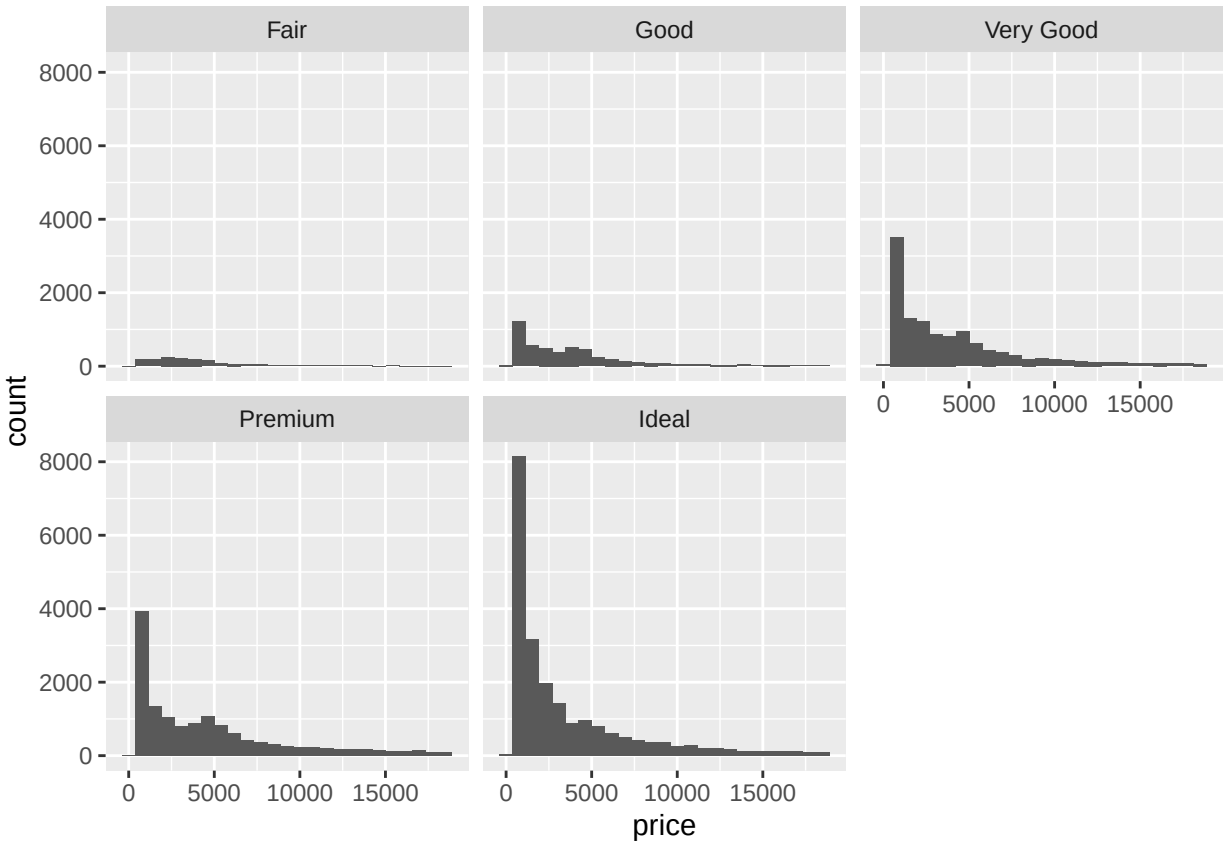
What this shows is there is actually a fairly uniform distribution of `price` across the cuts. There are slightly higher numbers at low price across the cuts, but overall pretty uniform. 'Fair' diamonds are a little more low-price heavy than the rest.

Exercise 12 (Website: 7.5.1.1 Ex. 5) Display the relationship between `price` and `cut` using `geom_violin()` and again using a facetted version of `geom_histogram()`. What are the pros and cons of each method?

```
ggplot(data = diamonds) +
  geom_violin(mapping = aes(x = cut, y = price), bins = 25)
```




```
ggplot(data = diamonds) +
  geom_histogram(mapping = aes(x = price), bins = 25) +
  facet_wrap(~ cut)
```

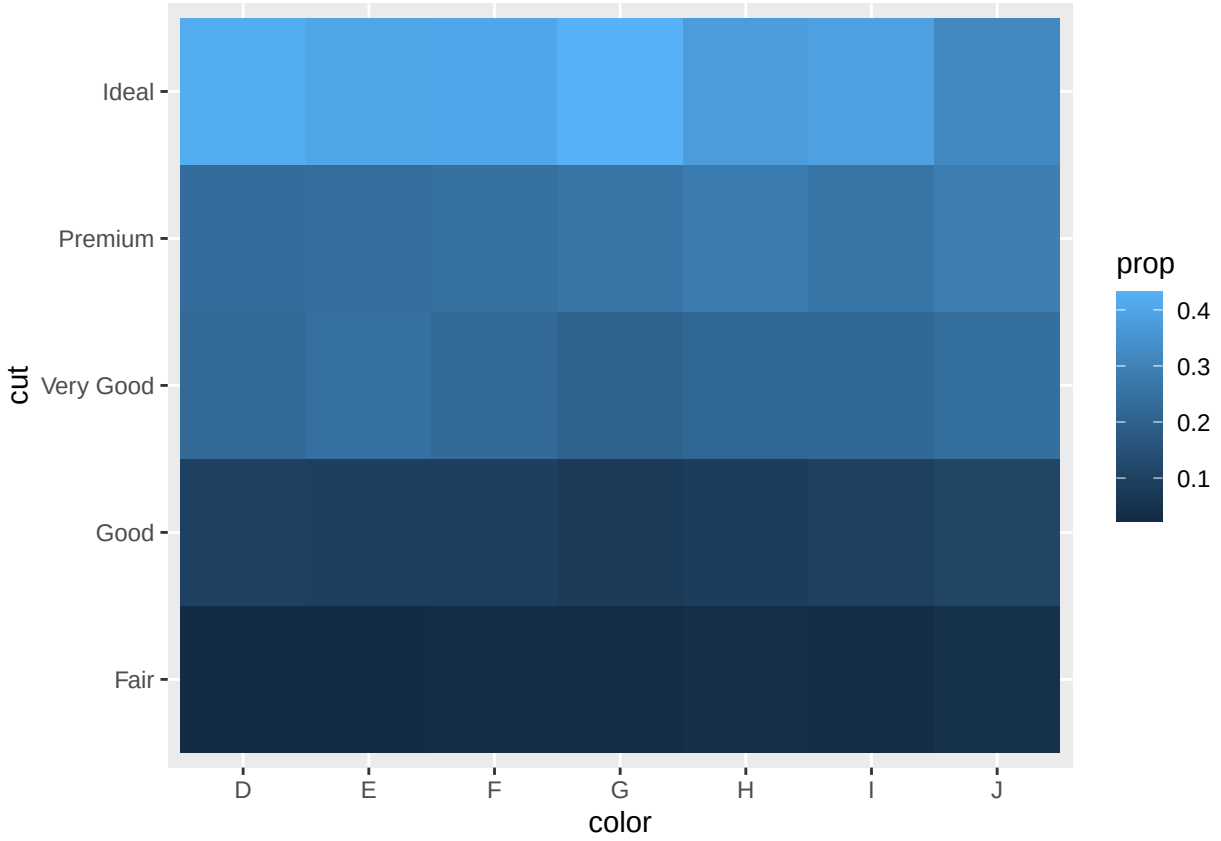


`geom_violin` and `geom_histogram` with `facet_wrap` both allow you to see distributions of a continous variable segmented by a discrete variable. I think `geom_violin` is a little more elegant, however `geom_histogram` allows you to see count, which can be important information.

Exercise 13 (Website: 7.5.2.1 Ex. 1) Can you rescale the data used in the plot below to more clearly show the distribution of cut within color, or color within cut? Starting from the code below, use rescaling to show the distribution of (a) cut within color and (b) color within cut.

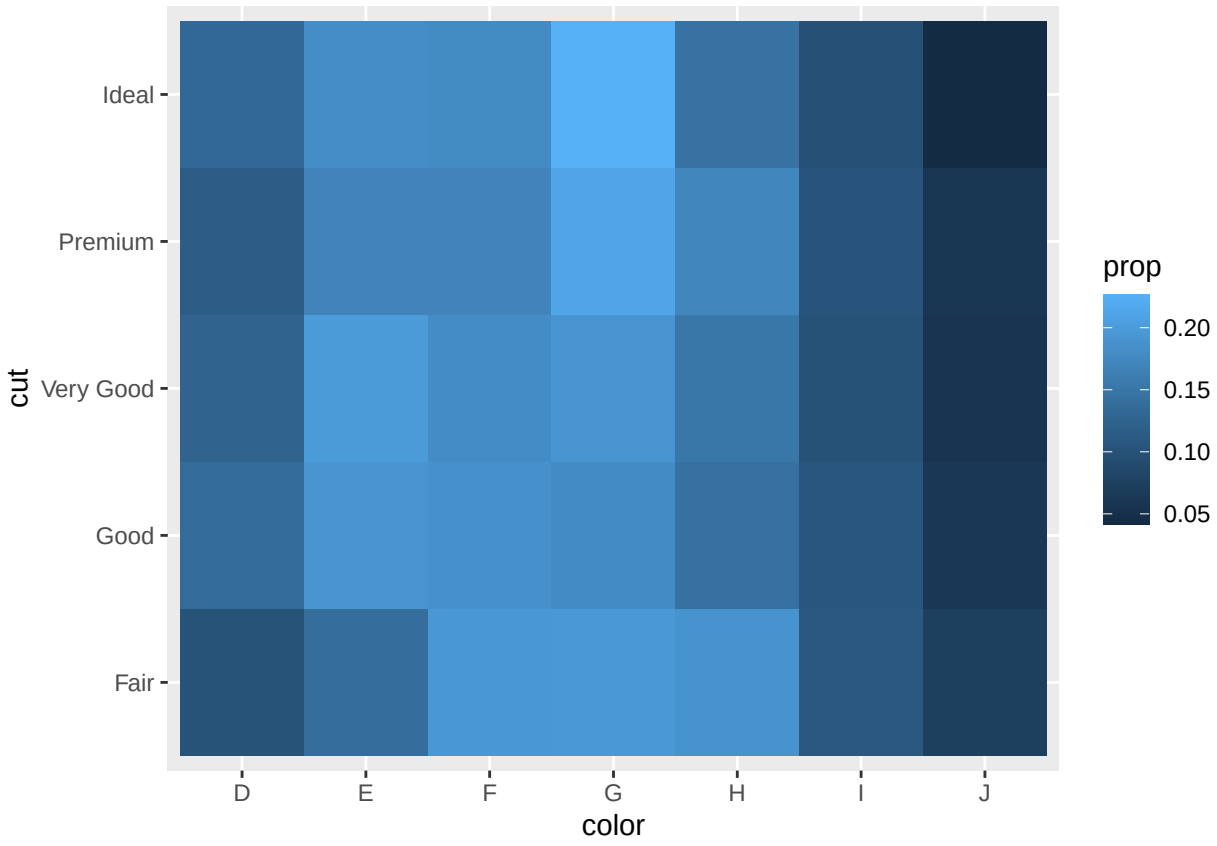
(a)

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



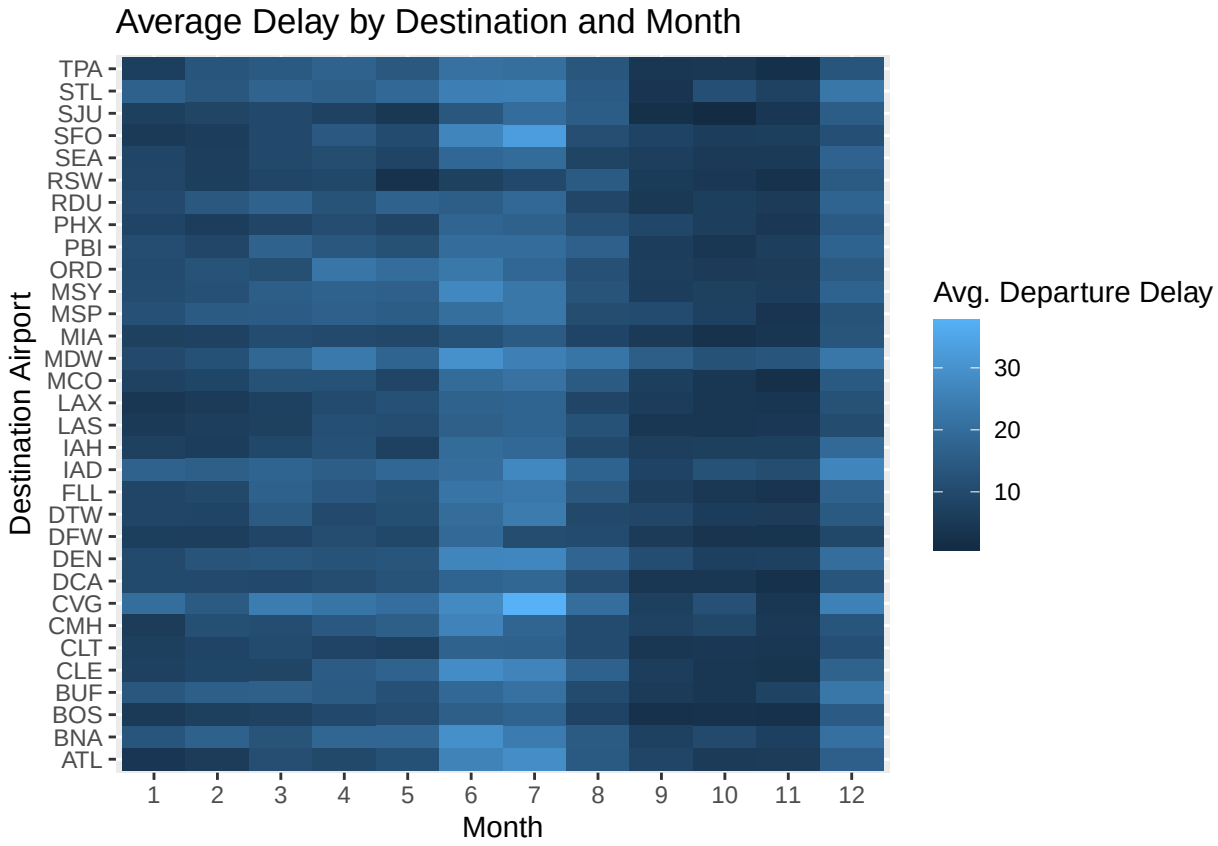
(b)

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



Exercise 14 (Website: 7.5.2.1 Ex. 2) Use `geom_tile()` together with `dplyr` to explore how average flight delays vary by destination and month of year. What makes the plot hard to read? Improve the plot.

```
flights %>%
  group_by(dest) %>%
  filter(n() >= 3000) %>%
  group_by(month, dest) %>%
  mutate(month = factor(month), avg_delay = mean(dep_delay, na.rm = TRUE)) %>%
  ggplot (mapping = aes(x = month, y = dest)) +
  geom_tile(mapping = aes(fill = avg_delay)) +
  labs(x = "Month", y = "Destination Airport", fill = "Avg. Departure Delay",
       title = "Average Delay by Destination and Month")
```



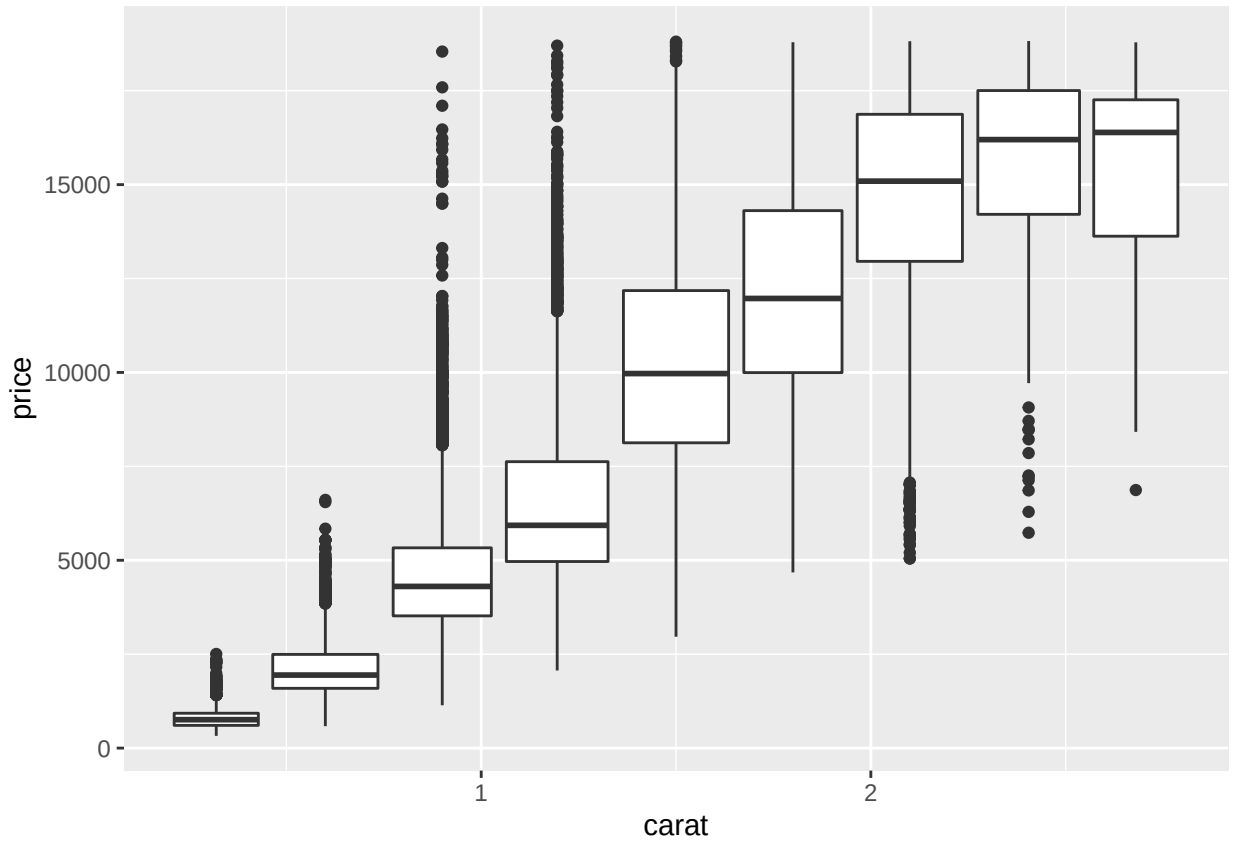
The plot is very hard to read because there are so many destinations that they do not fit on the graph. Thus, I filtered them so that only destinations with 3000 or more flights are shown.

Exercise 15 (Website: 7.5.2.1 Ex. 3) In Exercise 13, why is it slightly better to use `aes(x = color, y = cut)` instead of `aes(x = cut, y = color)`? **There are more `color` factors than there are `cut` factors, and the horizontal direciton is usually longer, so you want to put `color` on the x axis.**

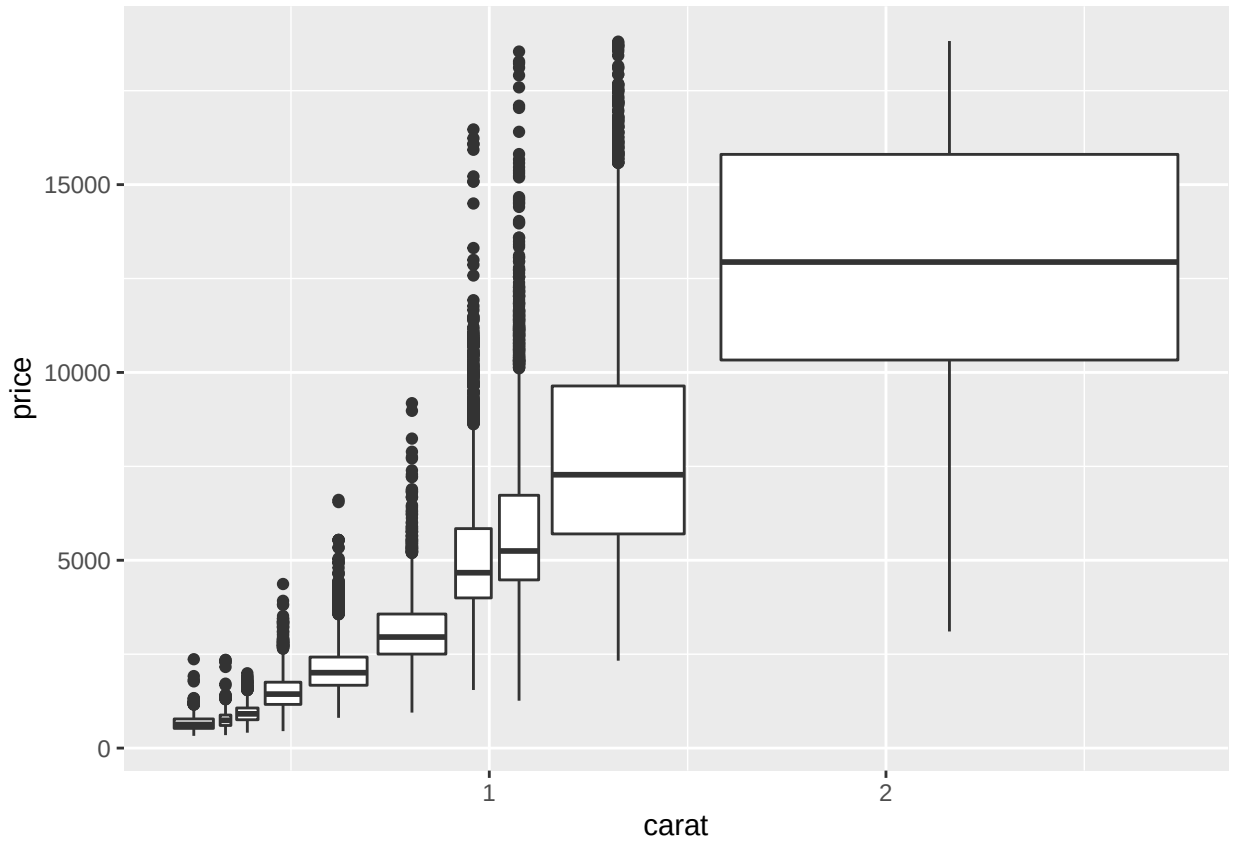
Exercise 16 (Website: 7.5.3.1 Ex. 2) Use the `smaller` dataset defined below for this exercise. Visualize the distribution of `carat` broken down by `price`. Construct 2 graphics, one using `cut_width()` and the other using `cut_number()`.

```
smaller <- diamonds %>%
  filter(carat < 3)

ggplot(data = smaller, mapping = aes(x = carat, y = price)) +
  geom_boxplot(mapping = aes(group = cut_width(carat, 0.3)))
```



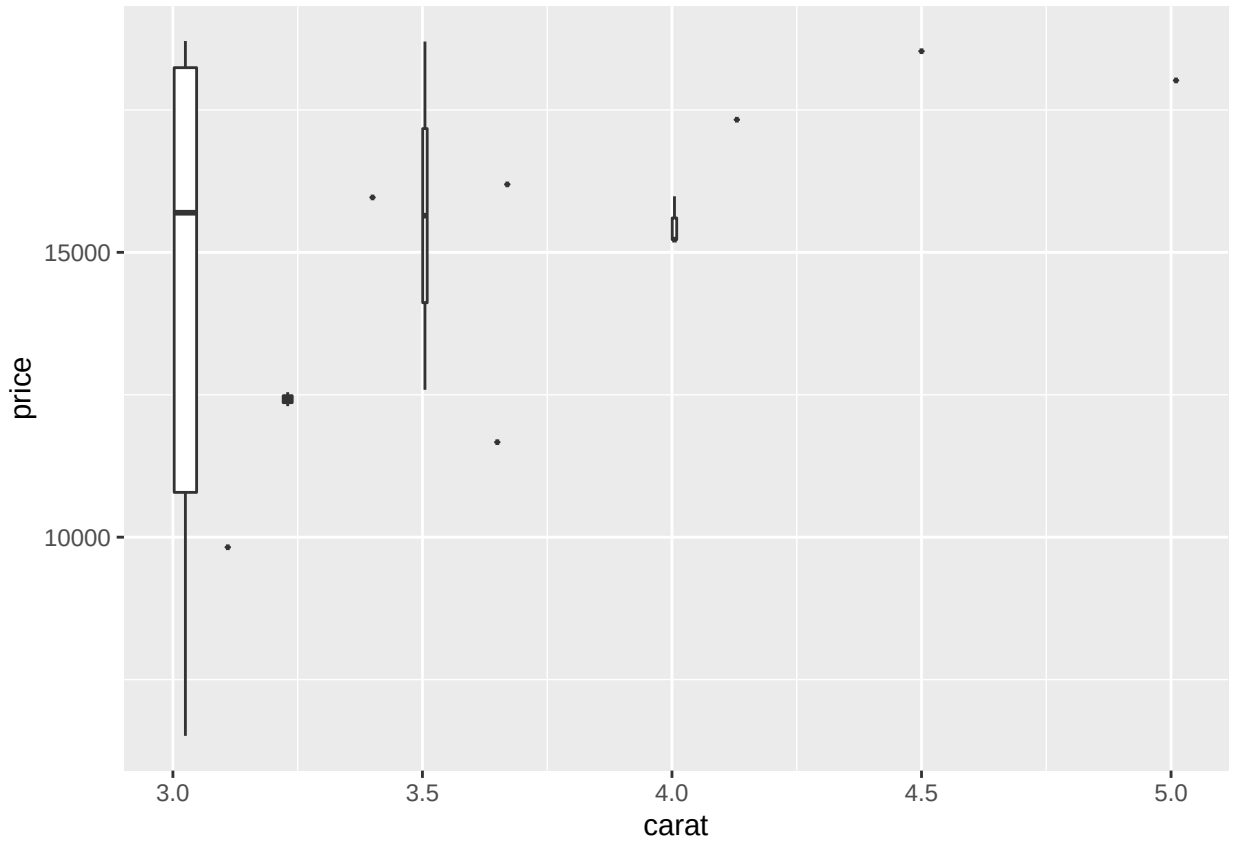

```
ggplot(data = smaller, mapping = aes(x = carat, y = price)) +
  geom_boxplot(mapping = aes(group = cut_number(carat, 10)))
```



Exercise 17 (Website: 7.5.3.1 Ex. 3) How does the distribution of `price` differ for very large diamonds (`carat` ≥ 3)? Is this result surprising? Why or why not?

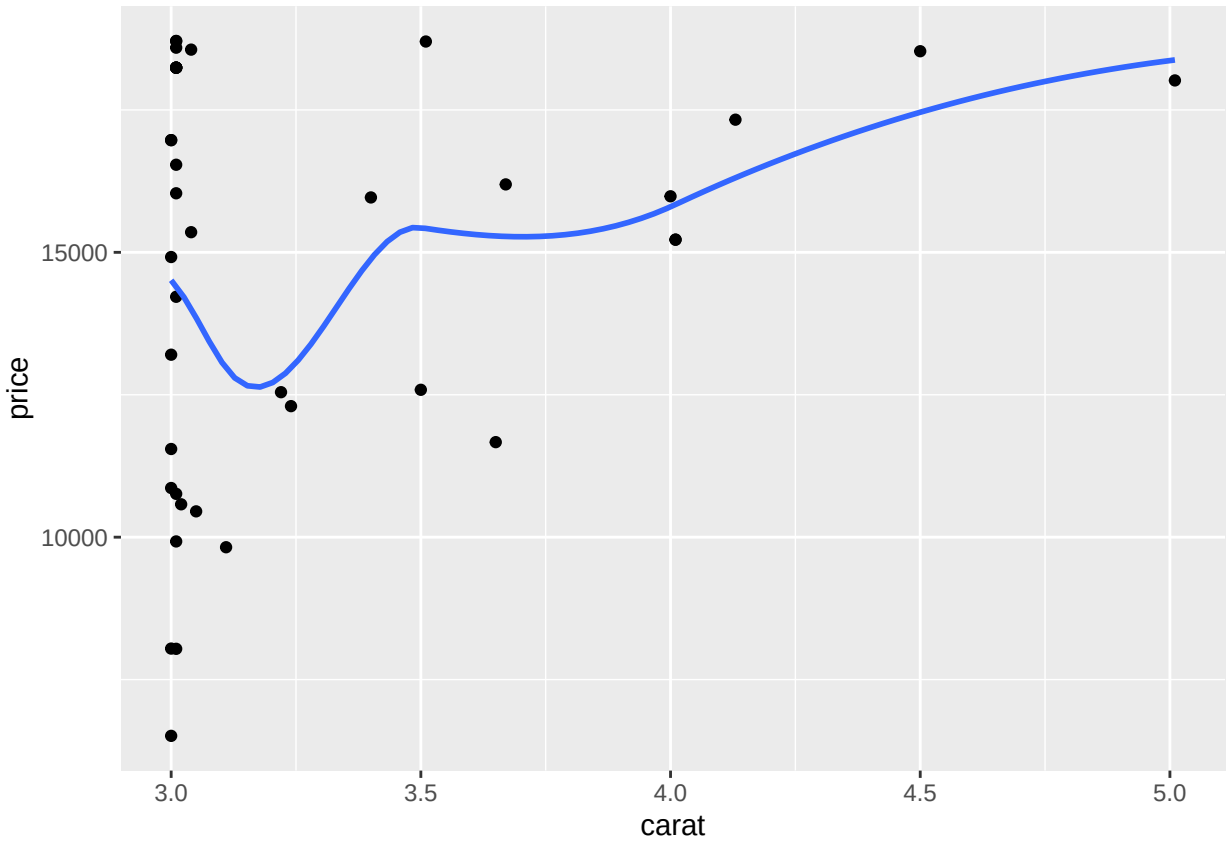
```
larger <- diamonds %>%
  filter(carat >= 3)

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




```
ggplot(data = larger, mapping = aes(x = carat, y  = price)) +
  geom_point() +
  geom_smooth(se = FALSE)
```

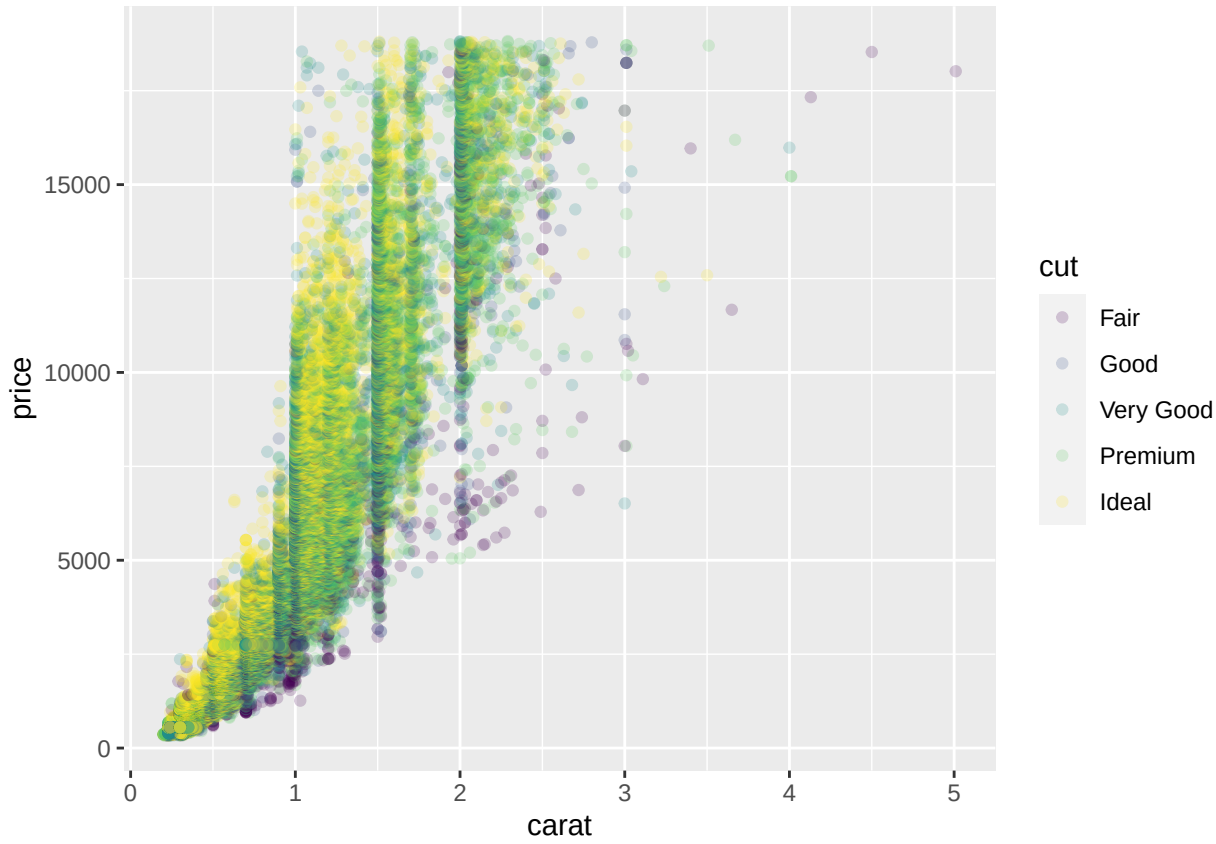
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



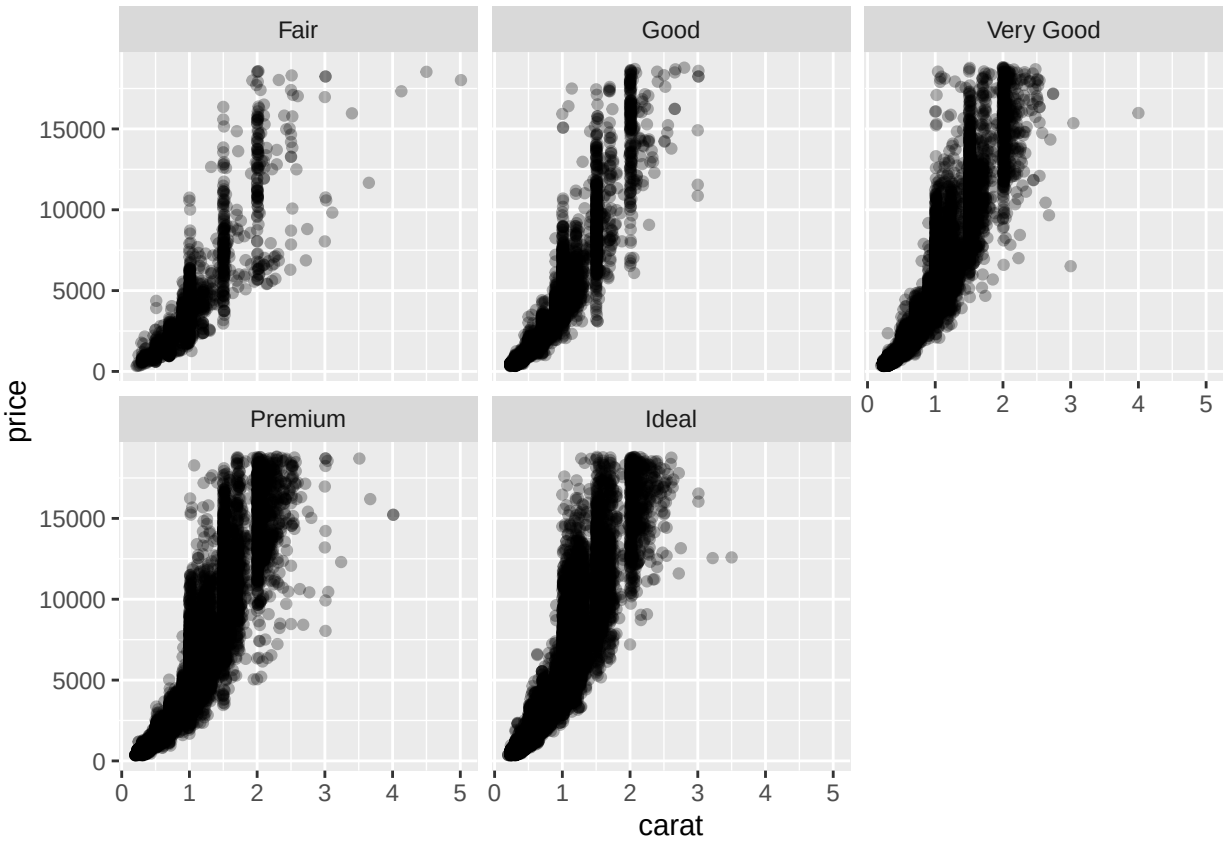
Most are exactly or slightly above 3 carat, and the price range is fairly spread out there. From 3.25 carat and larger, there is a positive correlation between `carat` and `price`. This isn't surprising, I'd expect larger diamonds to cost more.

Exercise 18 (Website: 7.5.3.1 Ex. 4) Combine two of the techniques you've learned to visualize the combined distribution of `cut`, `carat`, and `price`.

```
ggplot(data = diamonds) +
  geom_point(mapping = aes(x = carat, y = price, color = cut), alpha = 0.2)
```




```
ggplot (data = diamonds) +
  geom_point(mapping = aes(x = carat, y = price), alpha = 0.3) +
  facet_wrap (~ cut)
```


Fair
Good
Very Good
Premium
Ideal
price
carat
0
5000
10000
15000
0
1
2
3
4
5