Exploratory Data Analysis

Yifan Jin

15/02/2020

Introduction

This chapter will show you heo to use visualisation and transformation to explore your data in a systematic way, a task that statisticians call exploratory data analysis, or EDA for short. EDA is an iterative cycle.

1. Generate questions about your data
2. Search for answers by visualising, transforming, and modelling your data.
3. Use what you learn to refine your questions and/or generate new questions

EDA is not a formal process with a strict set of rules. During the initial phases of EDA you should feel free to investigate every idea that occurs to you. Some of them will pan out, and some will be dead ends.

To do data cleaning, you will need to deploy all the tools of EDA: visualisation, transformation, and moelling.

Prerequisites

In this chapter, we will combine what you have learned about dplyr and ggplot2 to interactively ask questions, answer them with data, and then ask new questions.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.2
```

```
## -- Attaching packages ----------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
## Warning: package 'forcats' was built under R version 3.5.2
```

```
## -- Conflicts -------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Questions

Your goal during EDA is to develop an understanding of your data. The easiest way to do this is to use questions as tools to guide your investigation. When you ask a question, the question focuses your attention on a specific part of your dataset and helps you decide which graphs, models, or transformations to make.

EDA is fundamentally a creative process. And like most creative processes, the key to asking quality questions is to generate of questions. It is difficult to ask revealing questions at the start of your analysis because you do not know what insights are contained in your dataset. On the other hand, each new question that you ask will expose you to a new aspect of data and increase your chance of making a discovery.

There is no rule about which questions you should ask to guide your research. However, two types of questions will always be useful for making discoveries within your data.

1. What type of variation occurs within my variables?

2. What type of covariation occurs between my variables?

The rest of this part will look at these two questions.

1. Explain what variation and covariation are
2. several ways to answer each question.

`Variable`: A variable is a quantity, quality, or property that you can measure

`Value`: A value is the state of a variable when you can measure it. The value of a variable may change from measurement to measurement.

`Observation`: A observation is a set of measurements made under similar conditions (you usually make all of the measurements in an observation at the same time and on the same object). An observation will contain several values, each associated will a different variable. An observation is a data point.

`Tabular data`: Tabular data is a set of values, each associated with a variable and an observation. Tabular data is tidy if each value is placed in its own cell, each variable in its own column, and each observation in its own row.

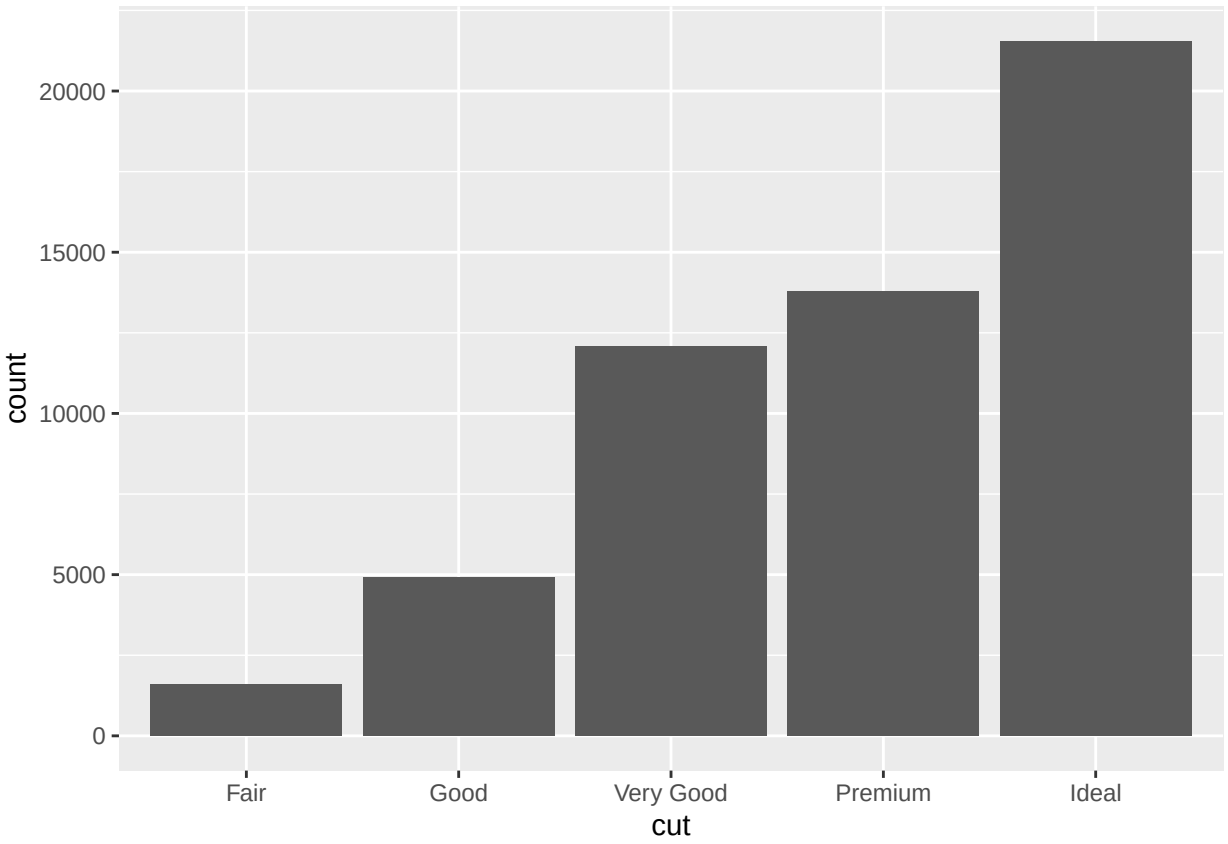
Variation

Variation is the tendency of the values of a variable to change from measuremnet to measurement. You can see variation easily in real life. Each variable has its own pattern of variation, which can reveal interesting information. The best way to understand that pattern is to visualse the distribution of the variable values.

Visualising distributions

For discrete variables: barplot A variable is `categorical` if it can only take one of a small set of values. In R, categorical variables are usually saved as factors or character vectors. To examine the distribution of a categorical variable, use a bar cahrt

```
ggplot(data=diamonds)+
  geom_bar(mapping=aes(x=cut))
```



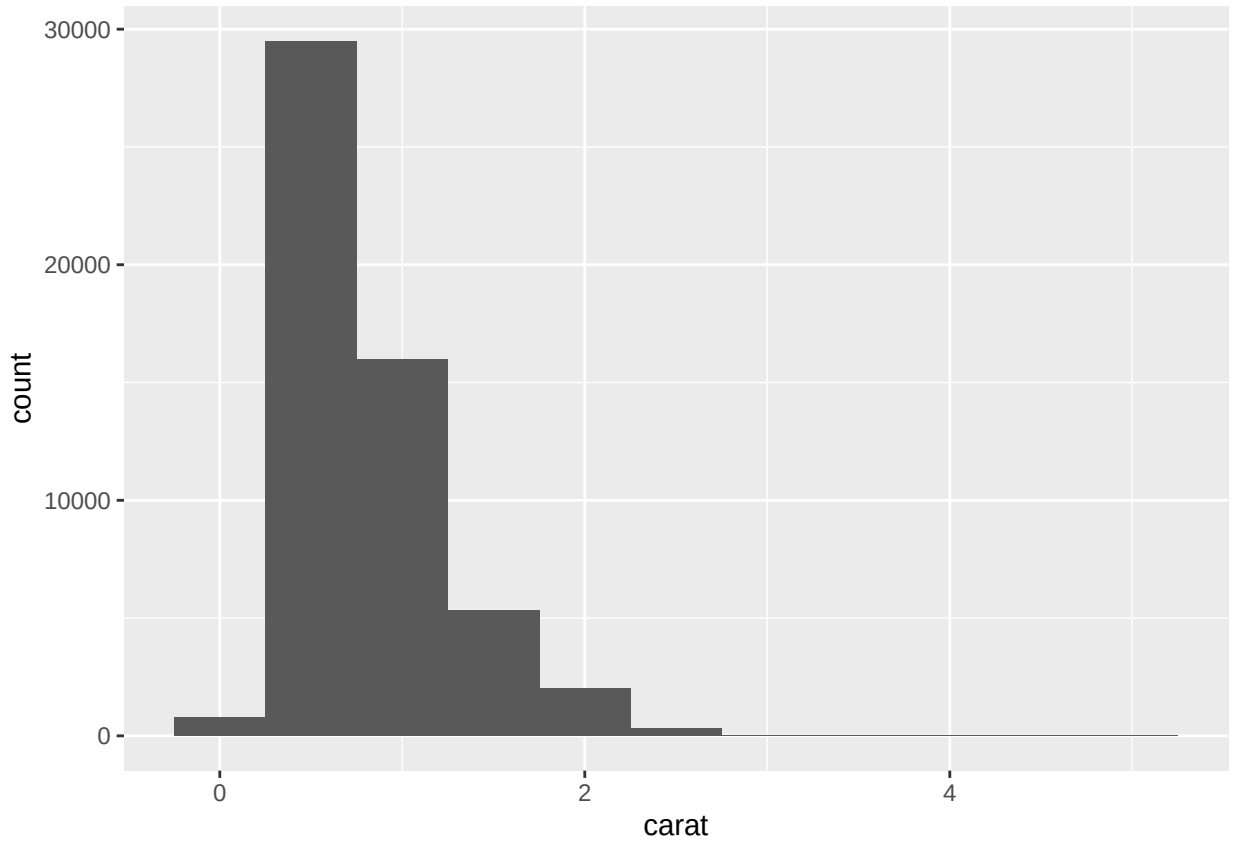
The height of the bars displays how many observations occurred with each x value. You can compute these values manually with: `dplyr::count()`

```
diamonds %>%
  count(cut)
```

```
## # A tibble: 5 x 2
##   cut           n
##   <ord>     <int>
## 1 Fair       1610
## 2 Good       4906
## 3 Very Good 12082
## 4 Premium   13791
## 5 Ideal     21551
```

For continuous variables: histogram A variable is `continuous` if it can take any of an infinite set of ordered values. Numbers and date-times are two examples of continuous variables. To examine the distribution of a continuous variable, we use histogram

```
ggplot(data=diamonds)+
  geom_histogram(mapping=aes(x=carat),binwidth = 0.5)
```



You can compute this by hand by combining `dplyr::count()` and `ggplot2::cut_width()`:

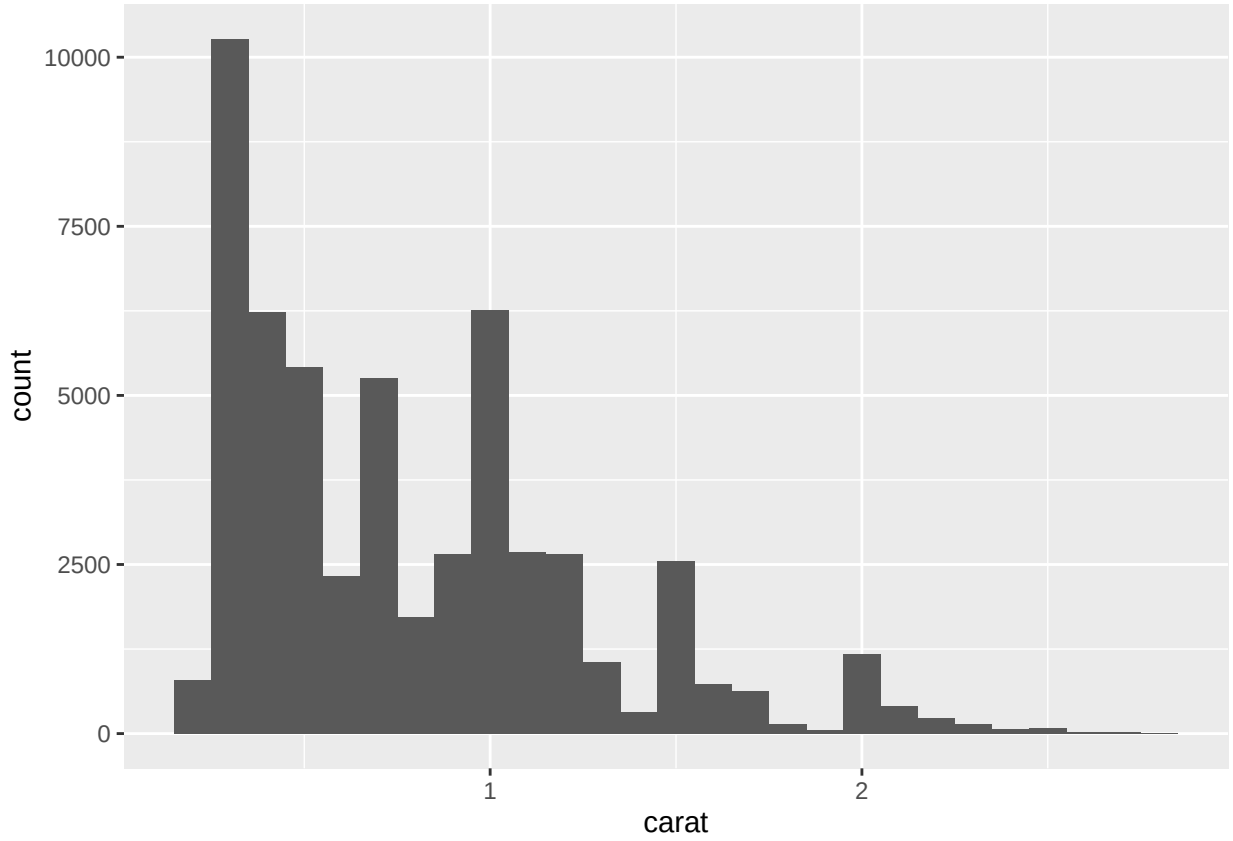
```
diamonds %>%
  count(cut_width(carat,0.5))
```

```
## # A tibble: 11 x 2
##    `cut_width(carat, 0.5)`     n
##    <fct>                   <int>
##  1 [-0.25,0.25]              785
##  2 (0.25,0.75]             29498
##  3 (0.75,1.25]             15977
##  4 (1.25,1.75]              5313
##  5 (1.75,2.25]              2002
##  6 (2.25,2.75]               322
##  7 (2.75,3.25]                32
##  8 (3.25,3.75]                 5
##  9 (3.75,4.25]                 4
## 10 (4.25,4.75]                 1
## 11 (4.75,5.25]                 1
```

A histogram divides the x-axis into equally bins and then uses the height of a bar to display the number of observations that fall in each bin. In the graph above, the tallest bar shows that almost 30000 observations that fall in each bin. In the graph above, the tallest bar shows that almost 30000 observations have `carat` value between 0.25 and 0.75, which are the left and right edges of the bar.

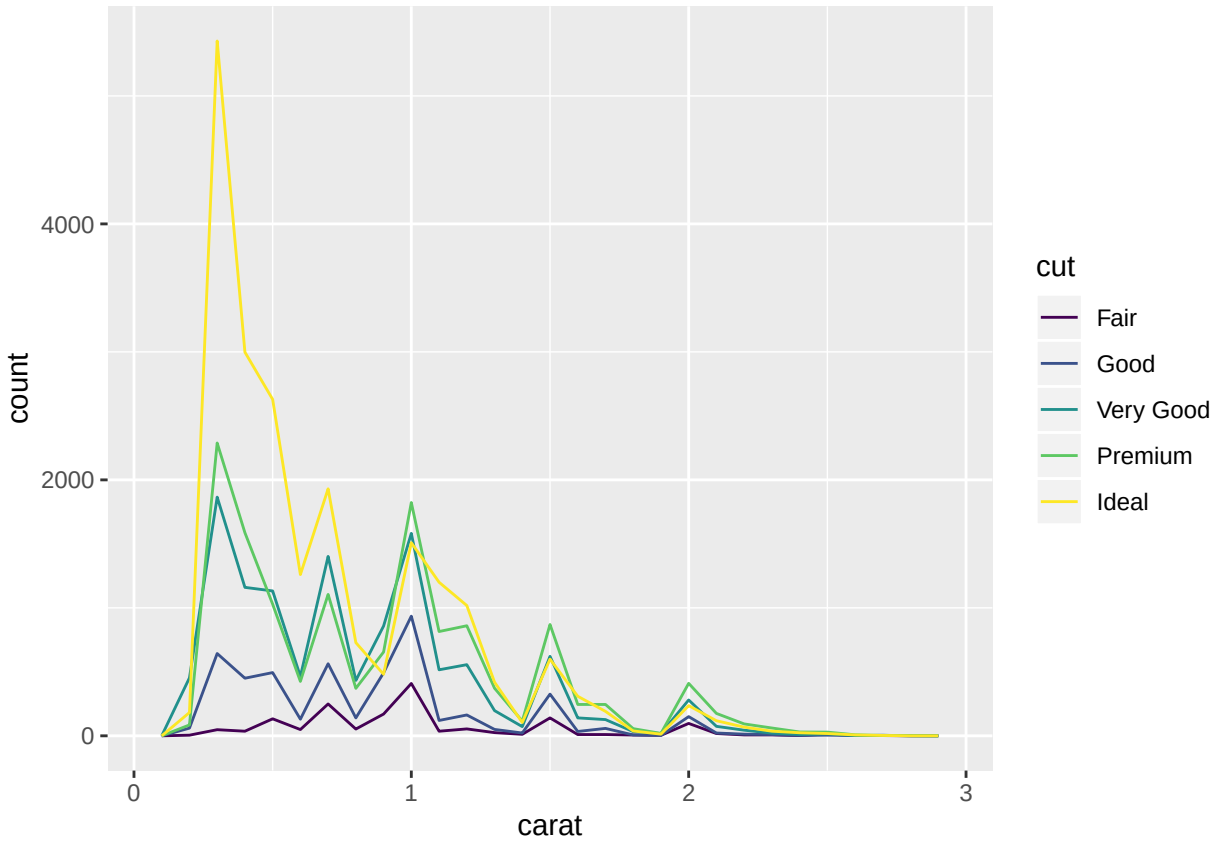
You can choose the width of each interval with `binwidth` argument, which is measured in the units of the `x` variable.

```
smaller<- diamonds %>%
  filter(carat<3)
ggplot(data=smaller,mapping=aes(x=carat))+
  geom_histogram(binwidth=0.1)
```



For different category, display the continuous variable for each category: multiple display If you wish to overlay multiple histograms in the same plot, I recommend using `geom_freqpoly()` instead of `geom_histogram()`. `geom_frepoly()` performs the same calculation as `geom_histogram()`, but instead of displaying the counts with bars, uses lines instead. It is much easier to understand overlapping lines than bars.

```
ggplot(data=smaller,mapping=aes(x=carat,color=cut))+
  geom_freqpoly(binwidth=0.1)
```



Typical values

In both bar charts and histograms, tall bars show the common values of a variable, and shorter bars show less-common values.

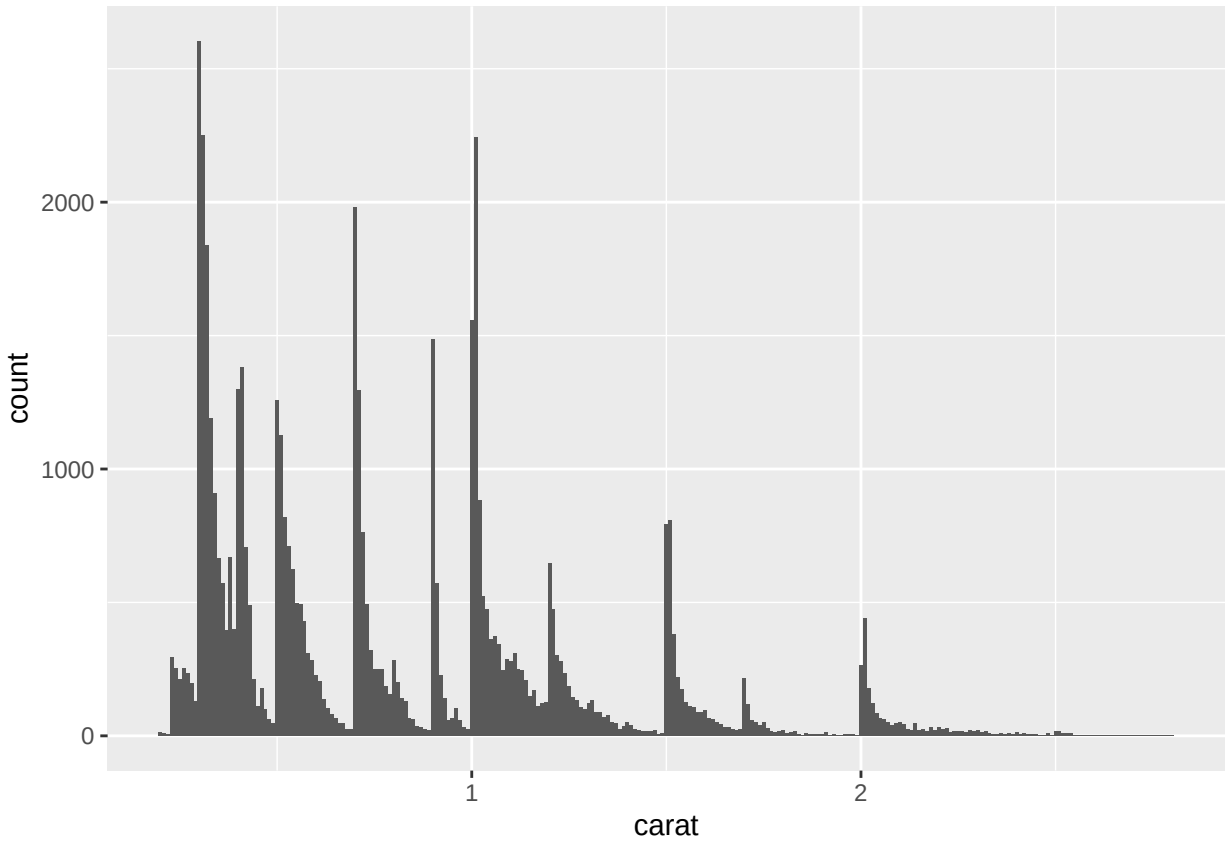
`Typical questions`

1. What values are the most common? Why?
2. Which values are rare? Why? Does that match your expectation?
3. Can you see any unusual patterns? What might explain them?

As an example, the histogram below suggests several interesting questions:

1. Why are there more diamonds at whole carates and common fractions of carats?
2. Why are there more diamonds slightly to the right of each peak than there are slightly to left of each peak?
3. Why are there no diamonds bigger than 3 carats?

```
ggplot(data=smaller,mapping=aes(x=carat))+
  geom_histogram(binwidth = 0.01)
```

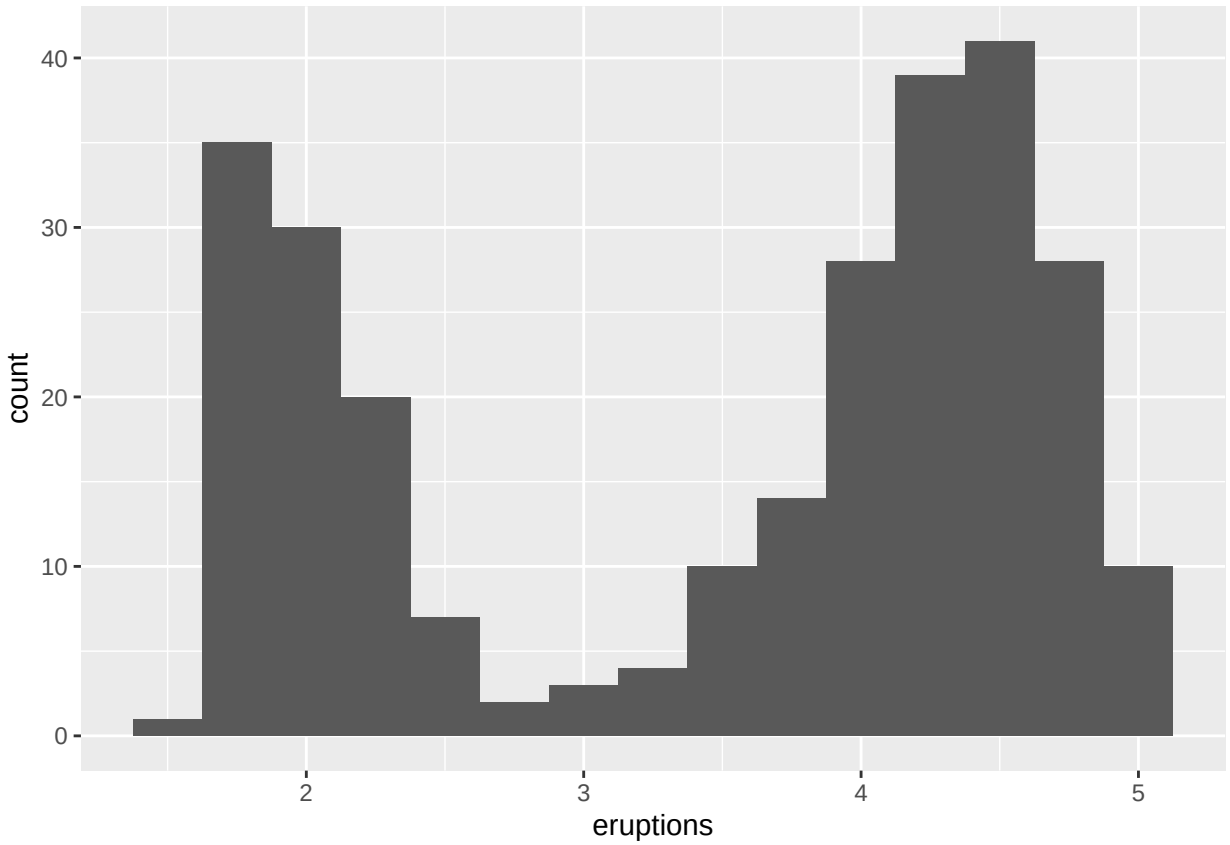


Clustering

Clusters of similar values suggest that subgroups exist in your data. To understand the subgroups, ask:

1. How are the observations within each cluster similar to each other?
2. How are the observations in separate clusters different from each other?
3. How can you explain or describe the clusters?
4. Why might the appearance of clusters be misleading?

```
ggplot(data=faithful,mapping=aes(x=eruptions))+
  geom_histogram(binwidth = 0.25)
```



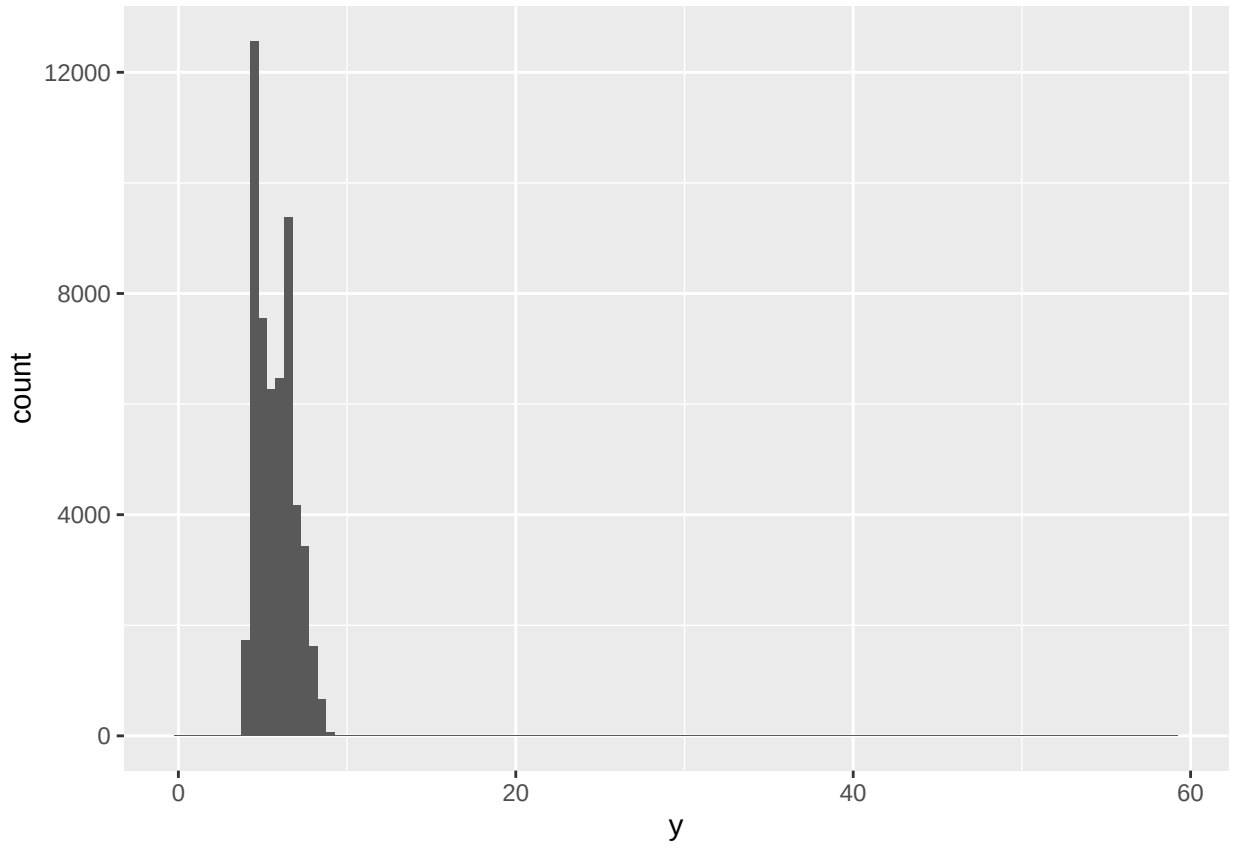
This can be divided into 2 groups.

Unusual values

Outliers are observations that are unusual; data points that don't seem to fit the pattern. Sometimes, outliers are data entry errors; other times outliers suggest new important science.

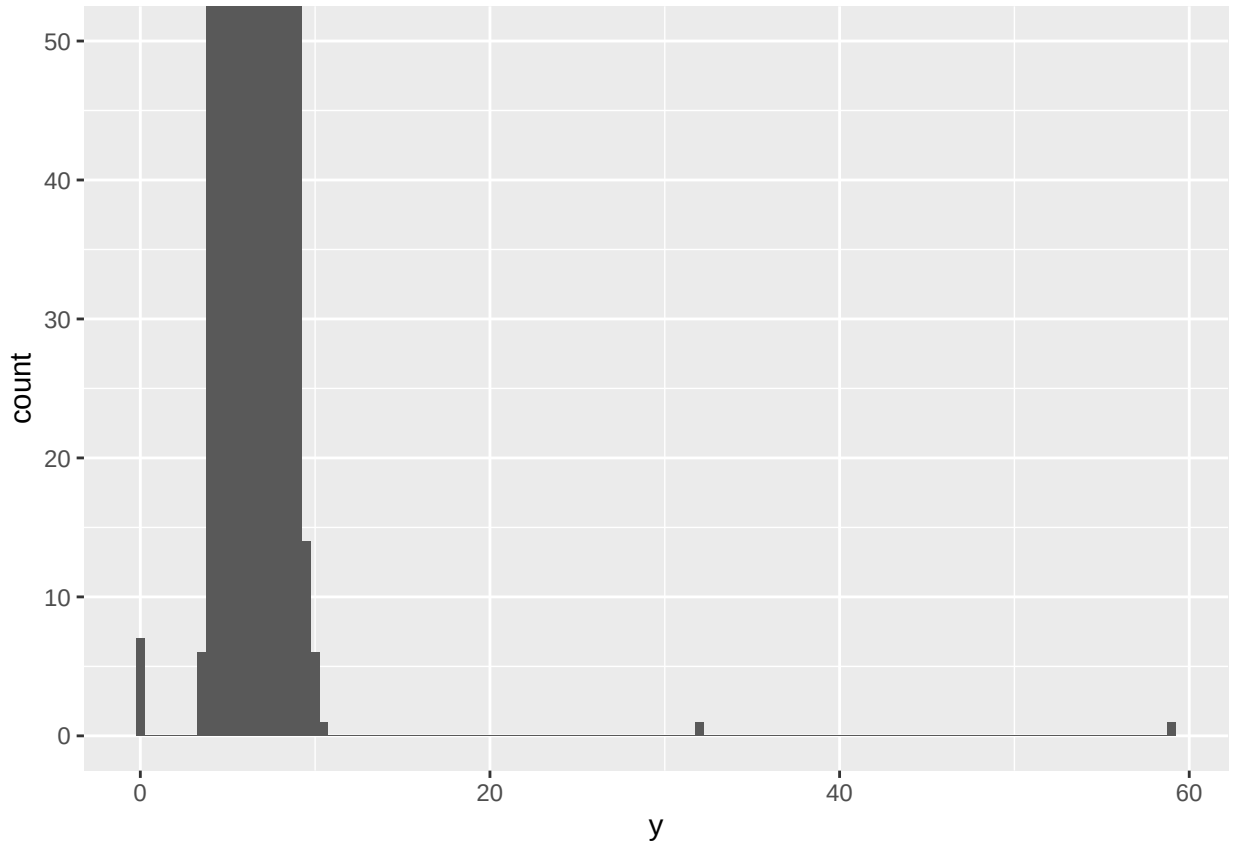
When you have a lot of data, outliers are sometimes difficult to see in a histogram. For example, take the distribution of `y` variable from the diamonds dataset. The only evidence of outliers is the unusually wide limits on the x-axis.

```
ggplot(diamonds)+
  geom_histogram(mapping=aes(x=y),binwidth=0.5)
```



There are so many observations in the common bins that the rare bins are so short that you cannot see them. To make it easy to see the unusual values, we need to zoom to small values of the y-axis with `coord_cartesian()`:

```
ggplot(diamonds)+
  geom_histogram(mapping = aes(x=y),binwidth=0.5)+
  coord_cartesian(ylim=c(0,50))
```



(`coord_cartesian()` also has an `xlim()` argument for when you need to zoom into the x-axis. ggplot2 also has `xlim()` and `ylim()` functions that work slightly differently: they throw away the data outside the limits)

This allows us to see that there are three unusual values: 0, ~30, and ~60. We pluck them out with dplyr:

```
unusual<-diamonds %>%
  filter(y<3|y>20) %>%
  select(price,x,y,z)%>%
  arrange(y)
unusual
```

```
## # A tibble: 9 x 4
##   price     x     y     z
##   <int> <dbl> <dbl> <dbl>
## 1  5139  0      0    0
## 2  6381  0      0    0
## 3 12800  0      0    0
## 4 15686  0      0    0
## 5 18034  0      0    0
## 6  2130  0      0    0
## 7  2130  0      0    0
## 8  2075  5.15  31.8  5.12
## 9 12210  8.09  58.9  8.06
```

We know that diamonds cannot have a width of 0mm, so these values must be incorrect. We might suspect that measurements of 32mm and 59mm are implausible: those diamonds are over an inch long, but don't cost hundreds of thousands of dollars!

Strategy: If the outliers doesn't affect the results too much, we can drop it can replace it with missing values.

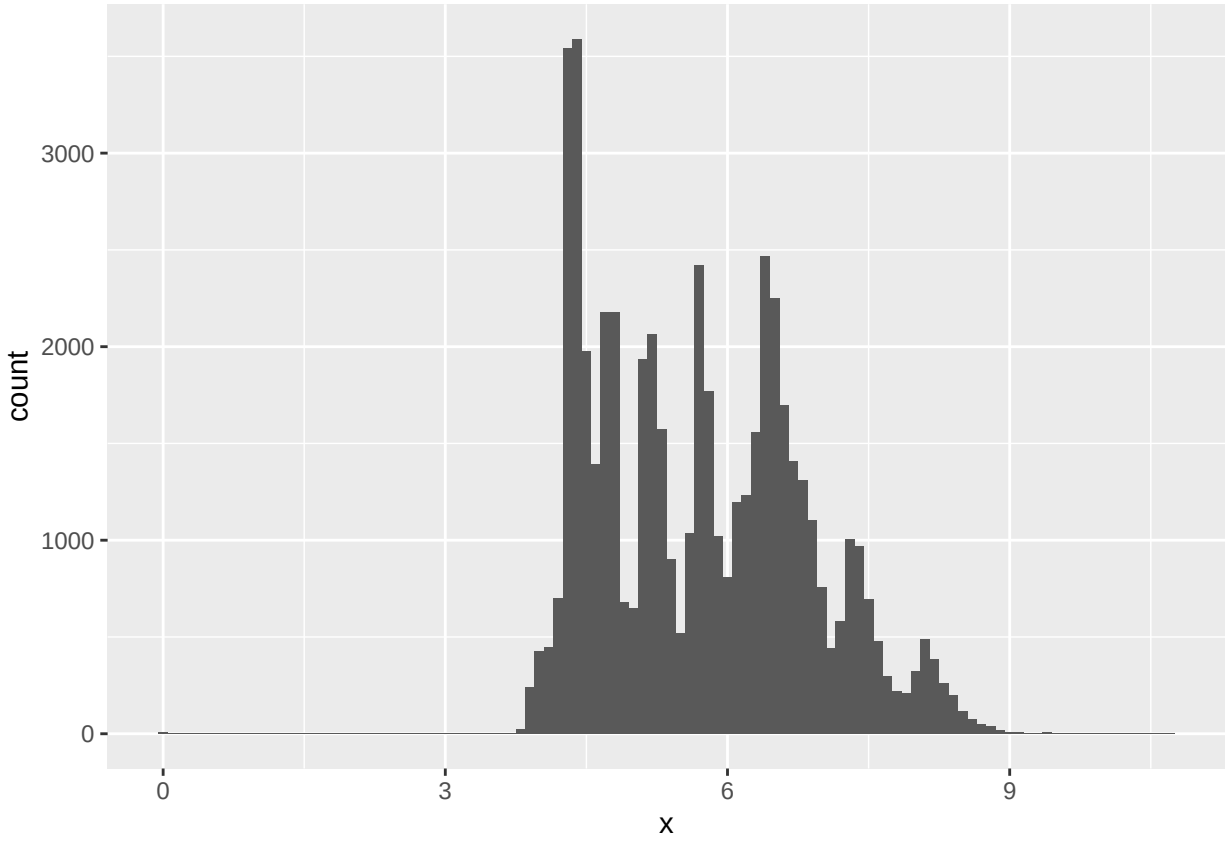
However, if it has huge effect, we can't drop it without justification. You need to figure out what caused them(data entry error) and disclose that you removed them in your write-up.

Exercises

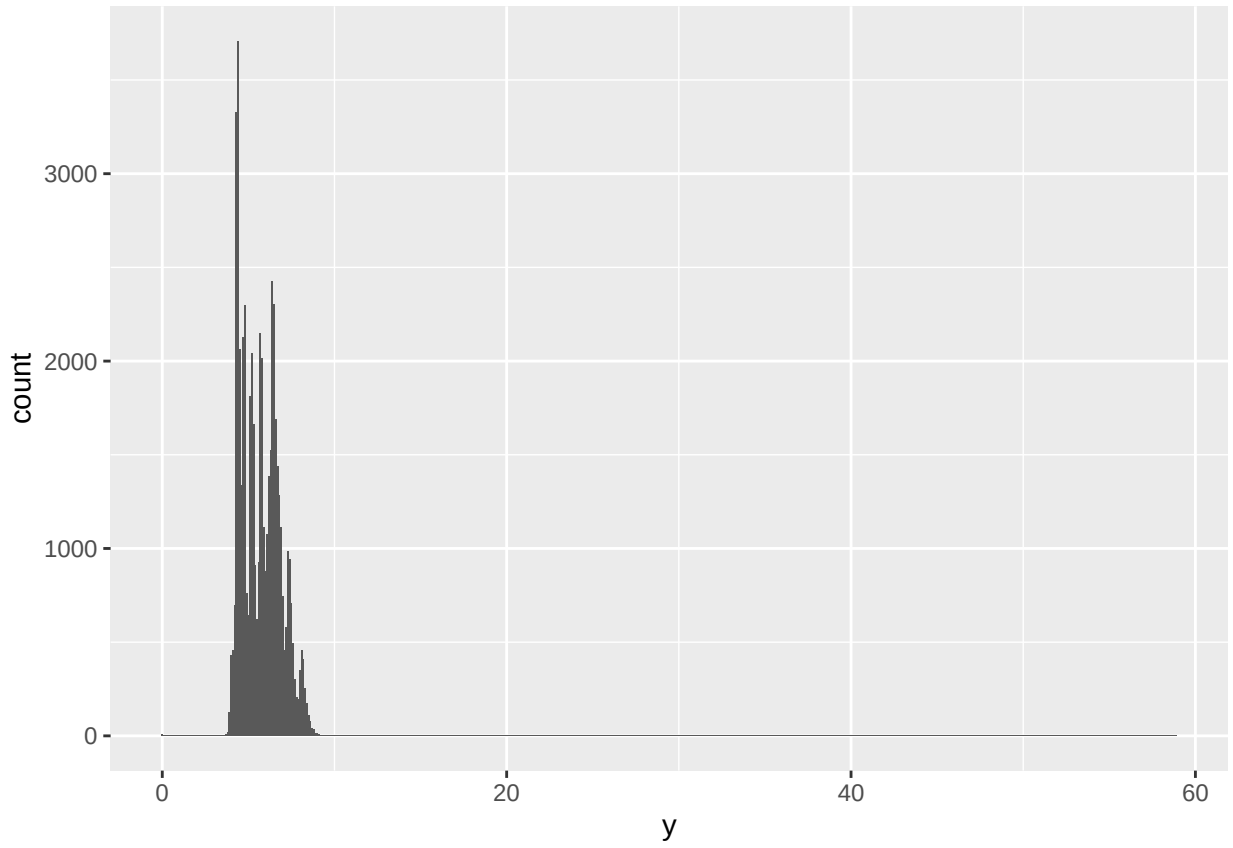
1. Explore the distribution of each of the `x`,`y`, and `z` variables in `diamonds`. What do you learn? Think about a diamond and how you might decide which dimension is the length, width and depth.

Answer

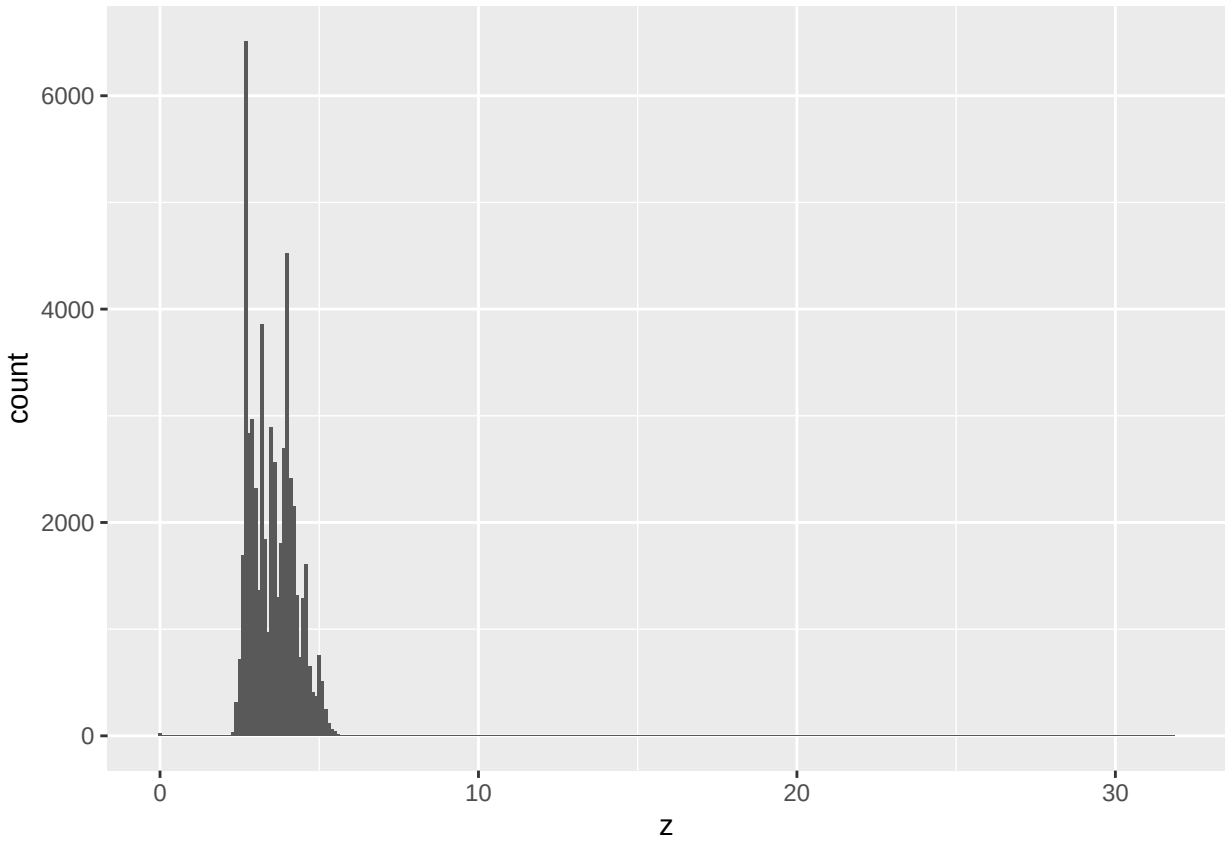
```
ggplot(diamonds)+
  geom_histogram(mapping=aes(x=x),binwidth = 0.1)+
  coord_cartesian()
```




```
ggplot(diamonds)+
  geom_histogram(mapping=aes(x=y),binwidth = 0.1)+
  coord_cartesian()
```




```
ggplot(diamonds)+
  geom_histogram(mapping=aes(x=z),binwidth = 0.1)+
  coord_cartesian()
```

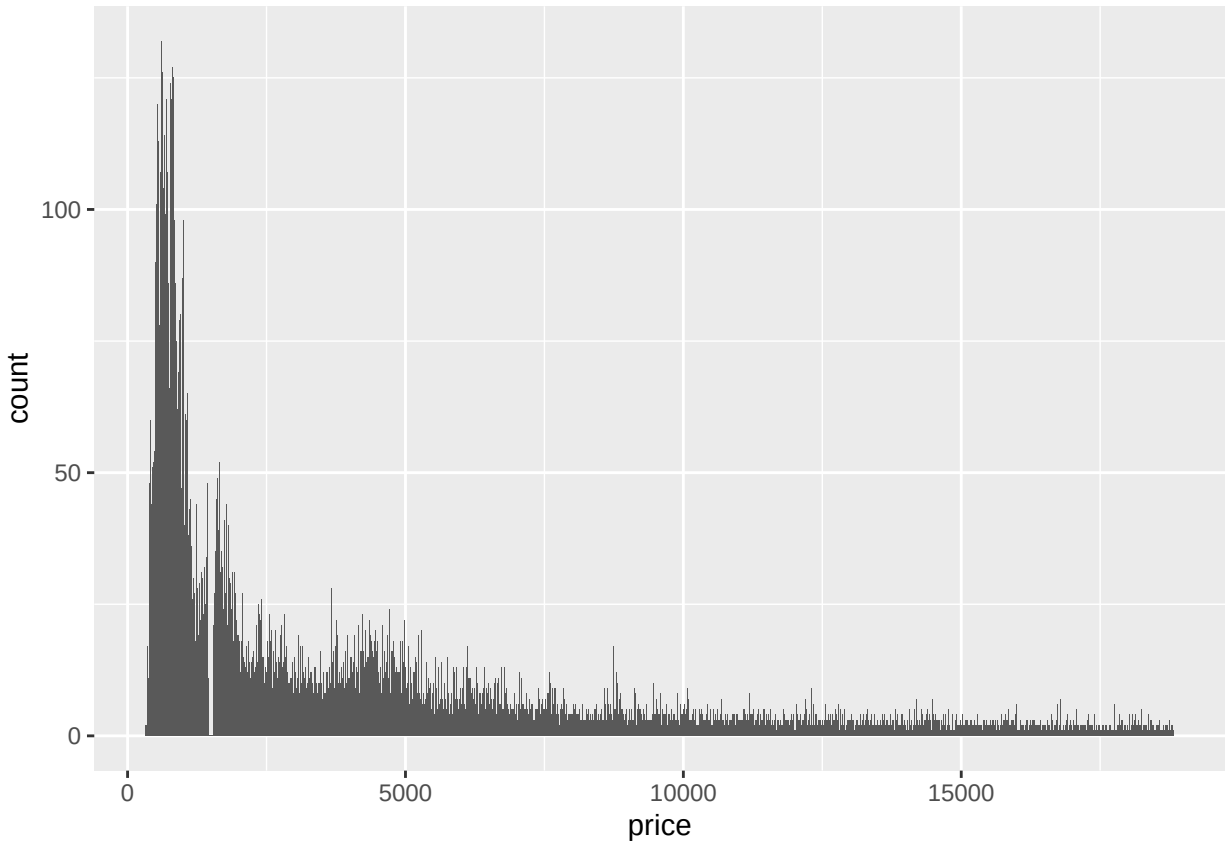


2. Explore the distribution of `price`. Do you discover anything unusual or surprising? (Hint: Carefully think about the `binwidth` and make sure you try a wide range of values)

Answer

It is quite suprising that most diamonds are relatively cheap.

```
ggplot(diamonds)+
  geom_histogram(mapping=aes(x=price),binwidth = 1)
```



3. How many diamonds are 0.99 carat? How many are 1 carat? What do you think is the cause of the difference

```
carat_99=diamonds %>% filter(carat==0.99) %>%count()
carat_100=diamonds %>% filter(carat==1) %>% count()
carat_99
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    23
```

```
carat_100
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  1558
```

Missing values