Exploratory Data Analysis in R

Pramod Duvvuri

3/30/2019

Introduction to EDA

The common steps in Exploratory Data Analysis (EDA) are:

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.

```r
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------ tidyverse 1.2.1 --
```

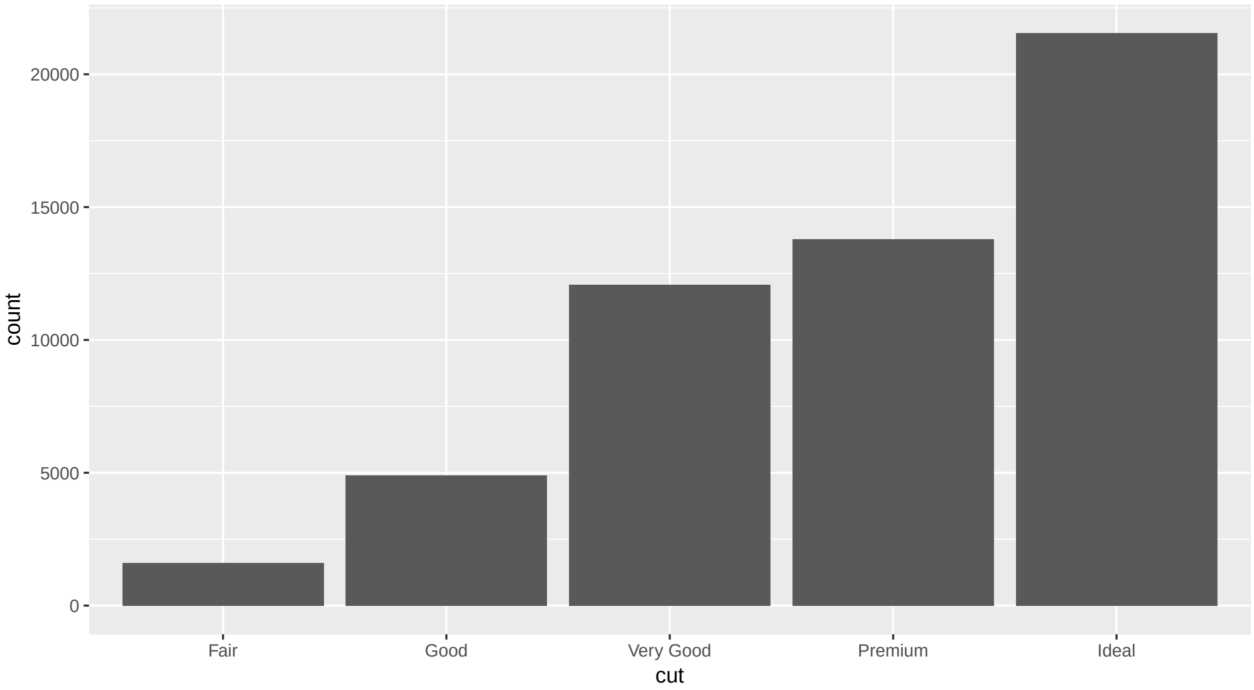
```
## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

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

```r
library(nycflights13)
```

Sample Visualizations

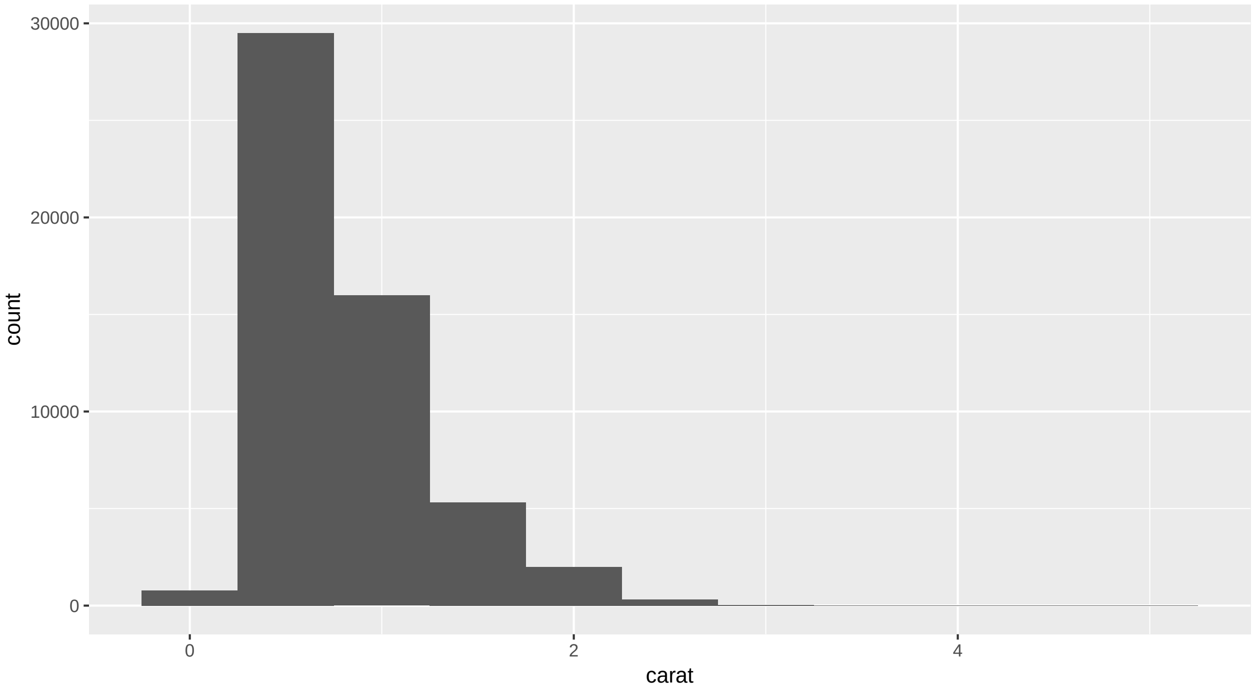
```r
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut))
```



```
# Heights of bars using 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
```

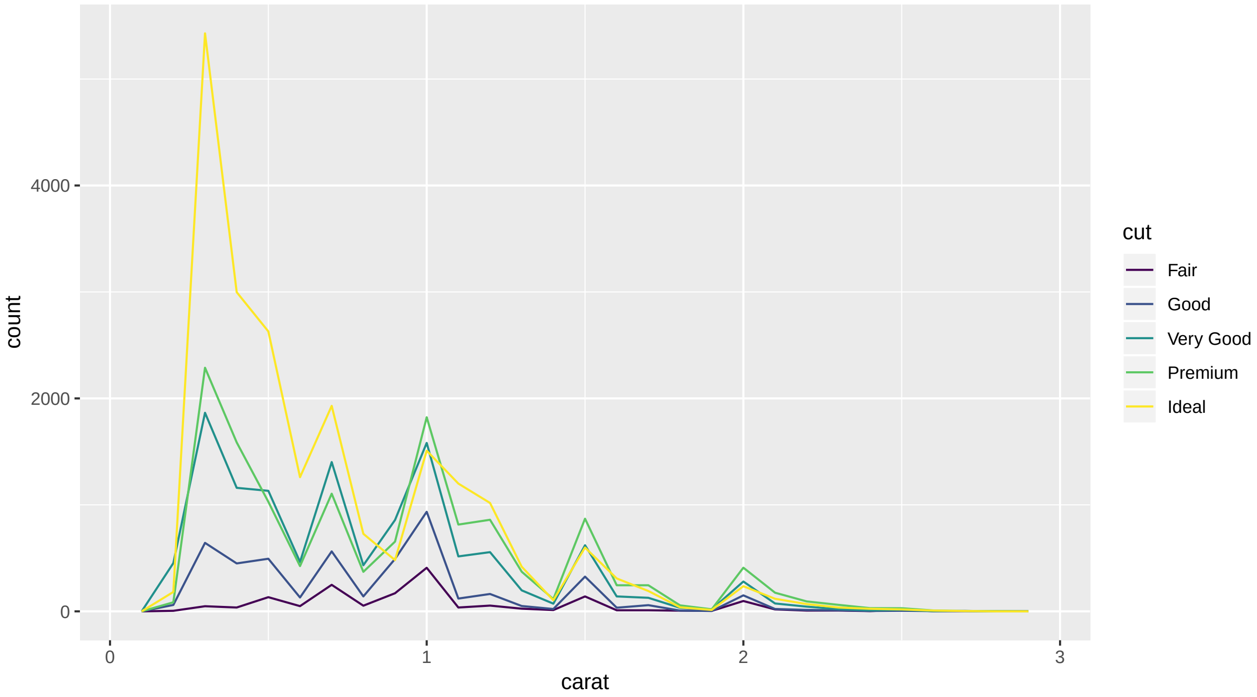
```
ggplot(data = diamonds) +
  geom_histogram(mapping = aes(x = carat), binwidth = 0.5)
```



```r
# Heights using dplyr and ggplot2
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
```

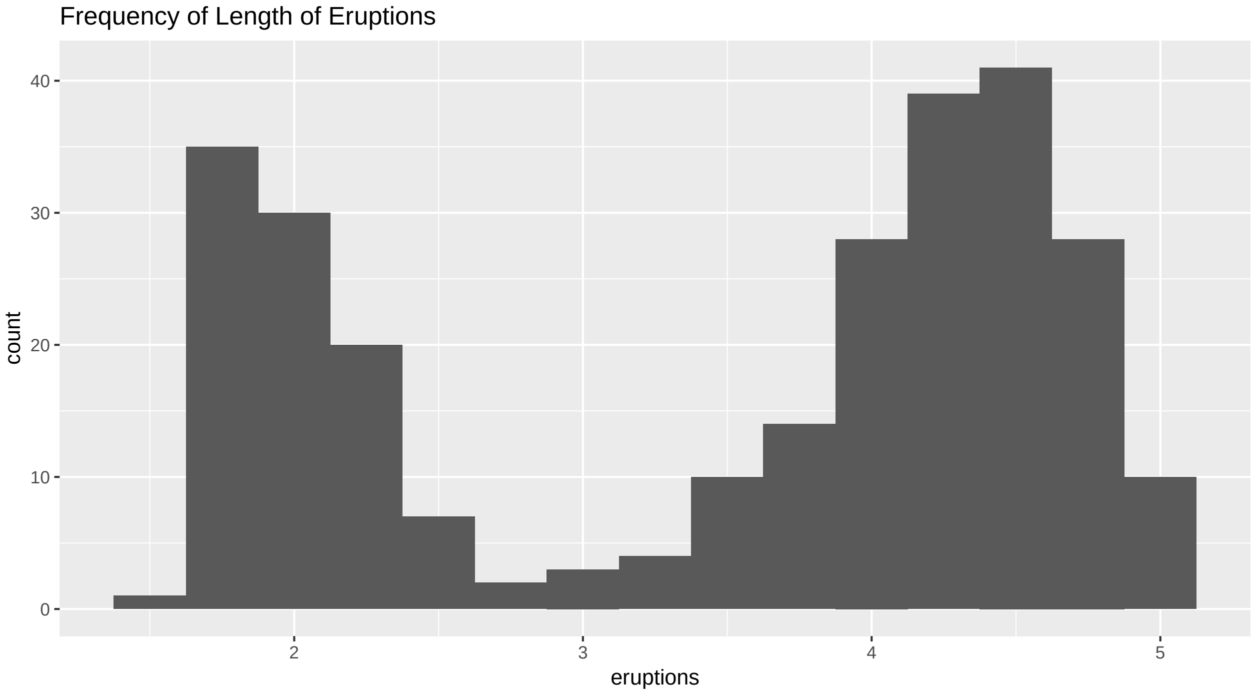
```r
# Multiple histograms ovrlapping in a single plot
ggplot(data = filter(diamonds, carat < 3), mapping = aes(x = carat, colour = cut)) +
  geom_freqpoly(binwidth = 0.1)
```



Sample Questions

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

```
ggplot(data = faithful, mapping = aes(x = eruptions)) +
  geom_histogram(binwidth = 0.25) + ggtitle('Frequency of Length of Eruptions')
```



Handling Data

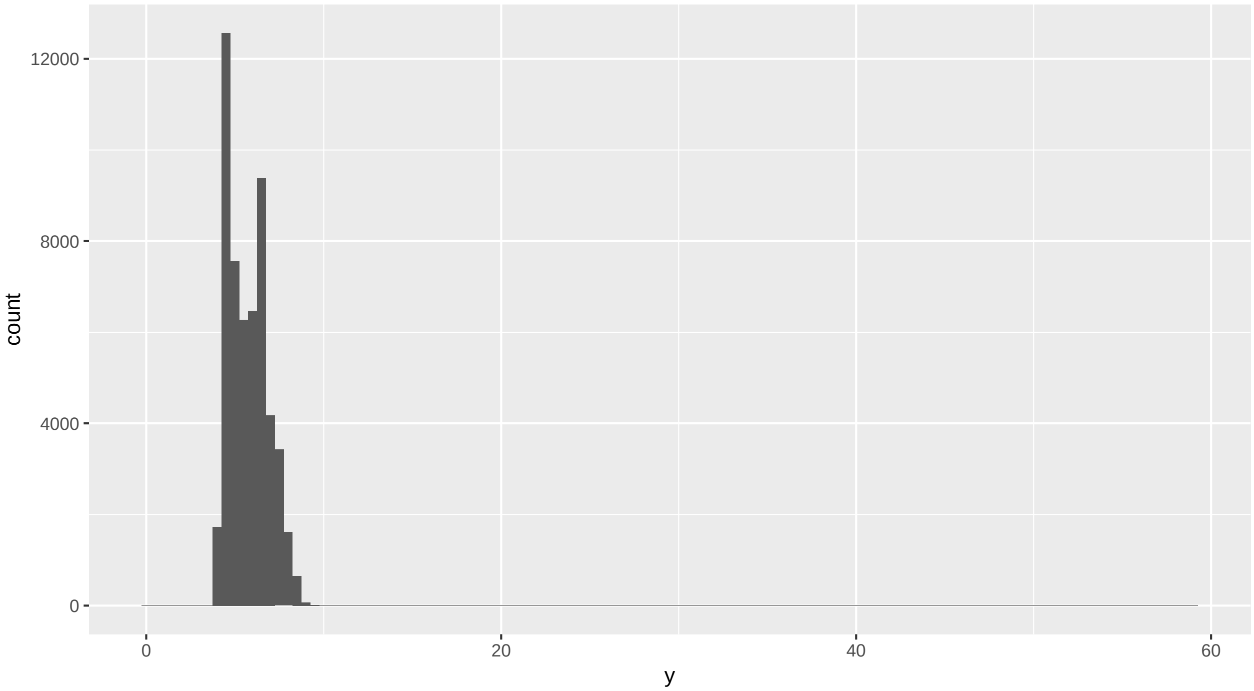
1. Typical Values
2. Unusual Values
3. Missing Values

Unusual Values

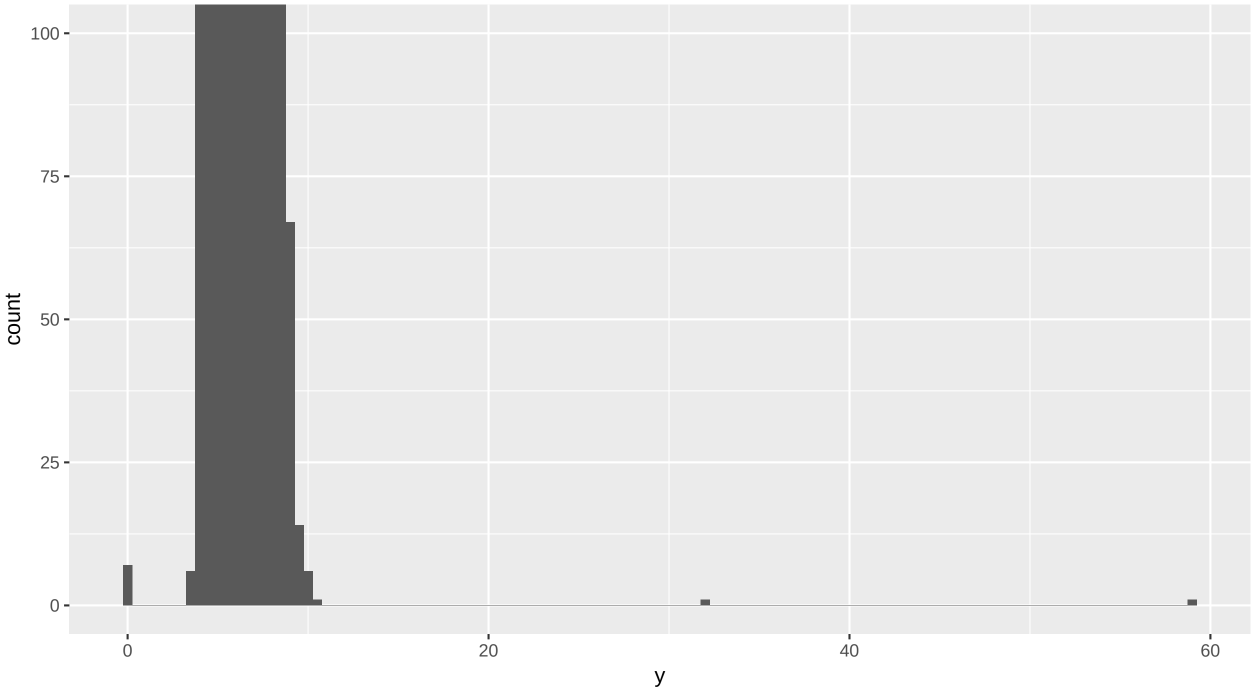
```
summary(diamonds)
```

```
##      carat               cut        color       clarity
##  Min.   :0.2000   Fair     : 1610   D: 6775   SI1    :13065
##  1st Qu.:0.4000   Good     : 4906   E: 9797   VS2    :12258
##  Median :0.7000   Very Good:12082   F: 9542   SI2    : 9194
##  Mean   :0.7979   Premium  :13791   G:11292   VS1    : 8171
##  3rd Qu.:1.0400   Ideal    :21551   H: 8304   VVS2   : 5066
##  Max.   :5.0100                     I: 5422   VVS1   : 3655
##                                     J: 2808   (Other): 2531
##      depth           table           price             x
##  Min.   :43.00   Min.   :43.00   Min.   :  326   Min.   : 0.000
##  1st Qu.:61.00   1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710
##  Median :61.80   Median :57.00   Median : 2401   Median : 5.700
##  Mean   :61.75   Mean   :57.46   Mean   : 3933   Mean   : 5.731
##  3rd Qu.:62.50   3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540
##  Max.   :79.00   Max.   :95.00   Max.   :18823   Max.   :10.740
##
##        y                z
##  Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 4.720   1st Qu.: 2.910
##  Median : 5.710   Median : 3.530
##  Mean   : 5.735   Mean   : 3.539
##  3rd Qu.: 6.540   3rd Qu.: 4.040
##  Max.   :58.900   Max.   :31.800
##
```

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



```
ggplot(data = diamonds) +
  geom_histogram(mapping = aes(x = y), binwidth = 0.5) +
  coord_cartesian(ylim = c(0,100))
```



```
# Digging unusual values using 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
```

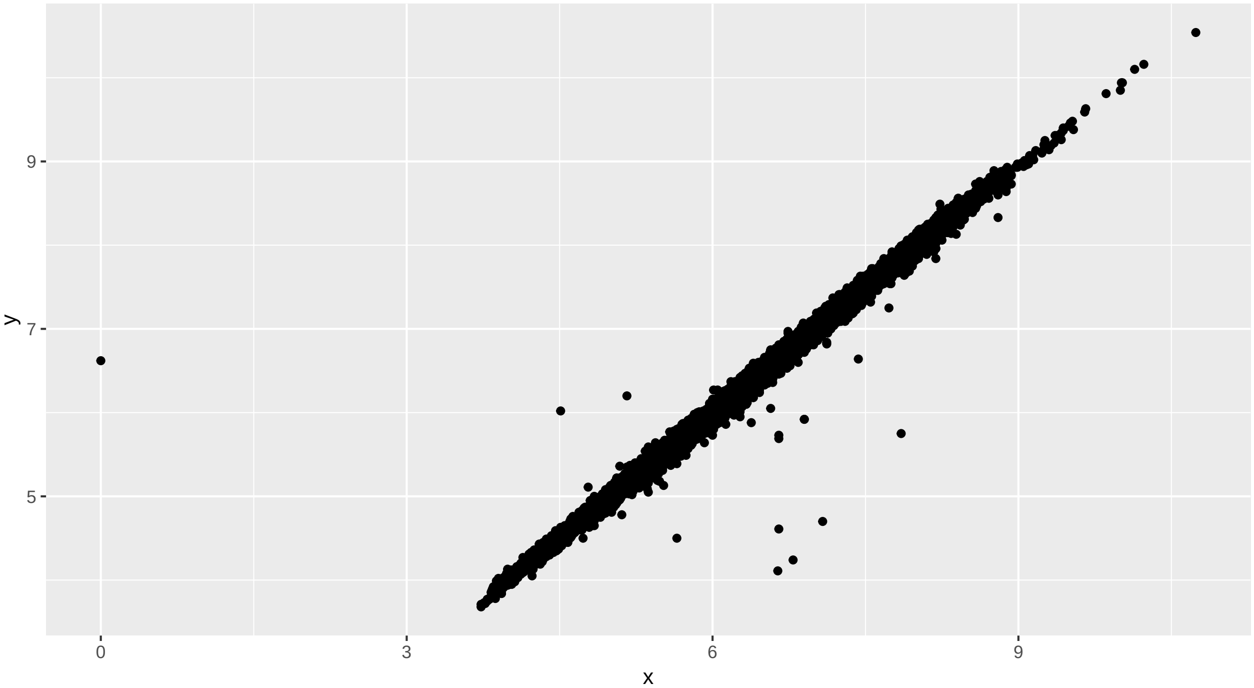
Missing Values

We shall replace unusual values in the data as missing values (NA) instead of dropping them

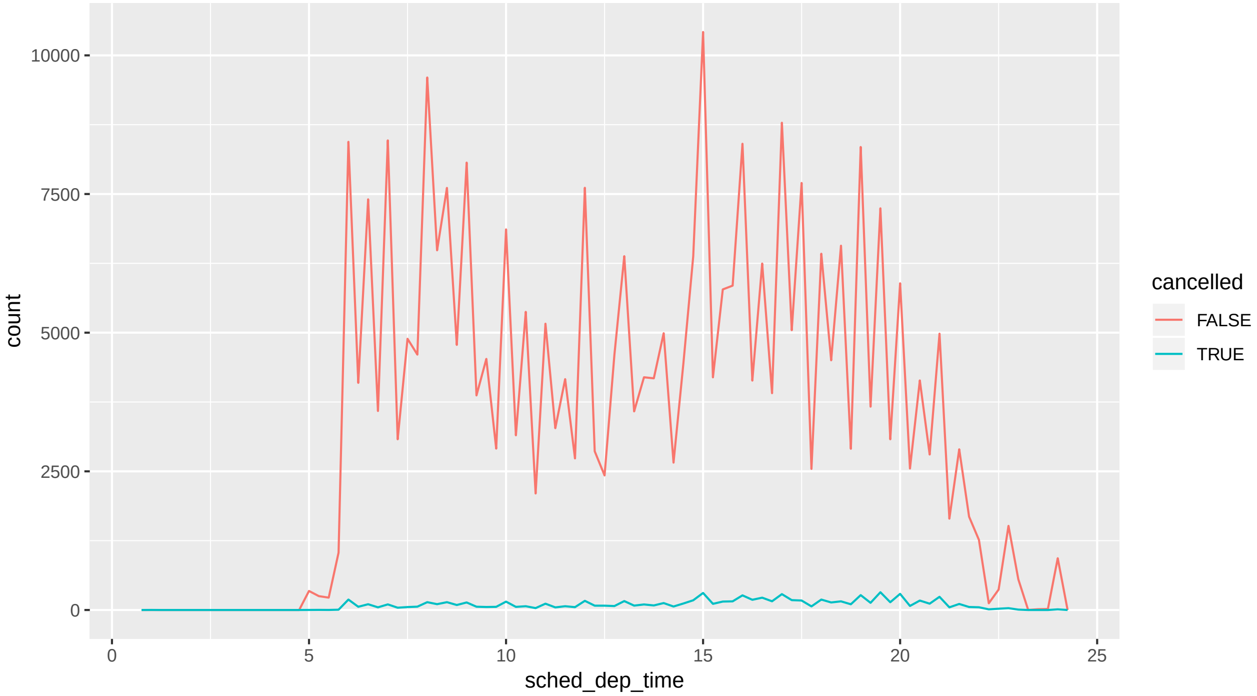
ifelse()

```
# case_when() can also be used to re-write the below line of code
diamonds2 <- diamonds %>%
  mutate(y = ifelse(y < 3 | y > 20, NA, y))
```

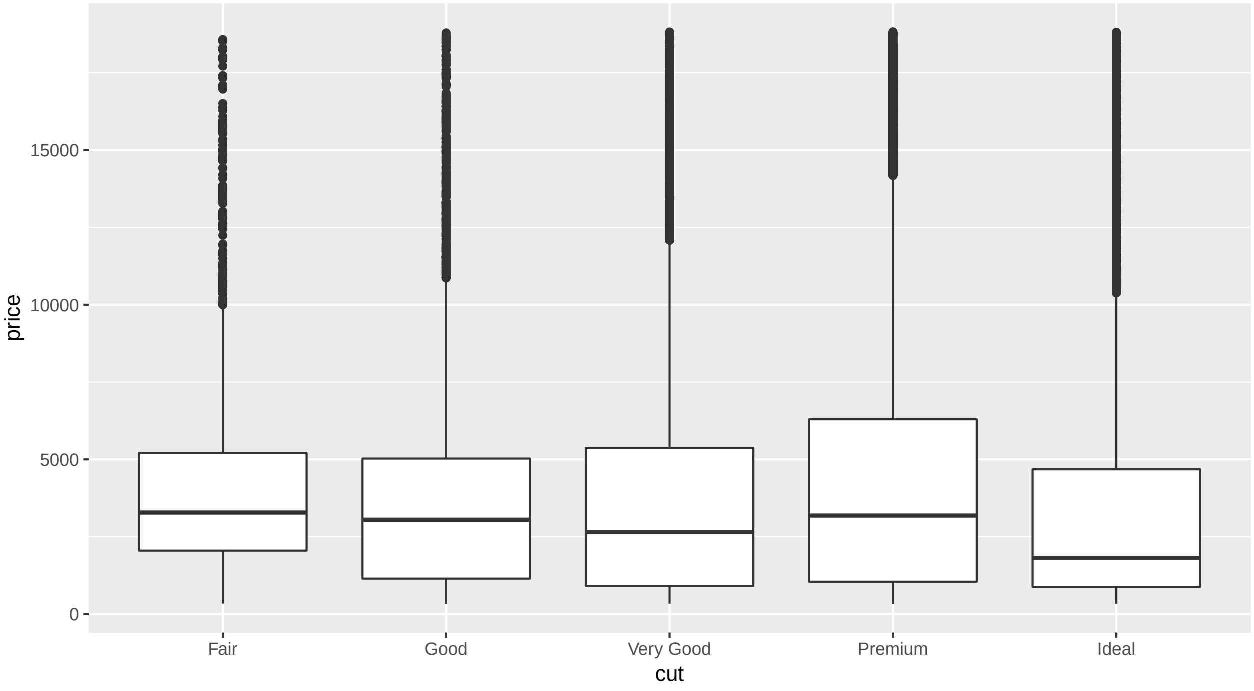
```
ggplot(data = diamonds2, mapping = aes(x = x, y = y)) +
  geom_point(na.rm = TRUE)
```



```
# Compare cancelled and non-cancelled flights
nycflights13::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(sched_dep_time)) +
    geom_freqpoly(mapping = aes(colour = cancelled), binwidth = 1/4)
```

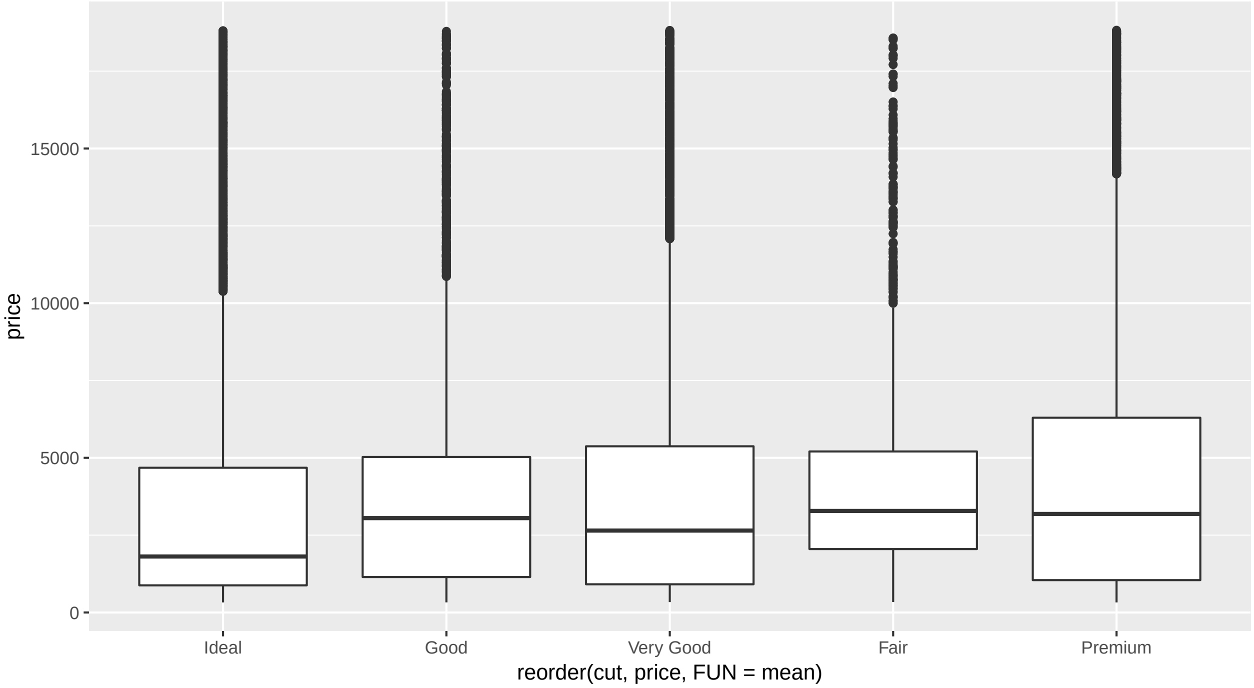


```
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = cut ,y = price))
```

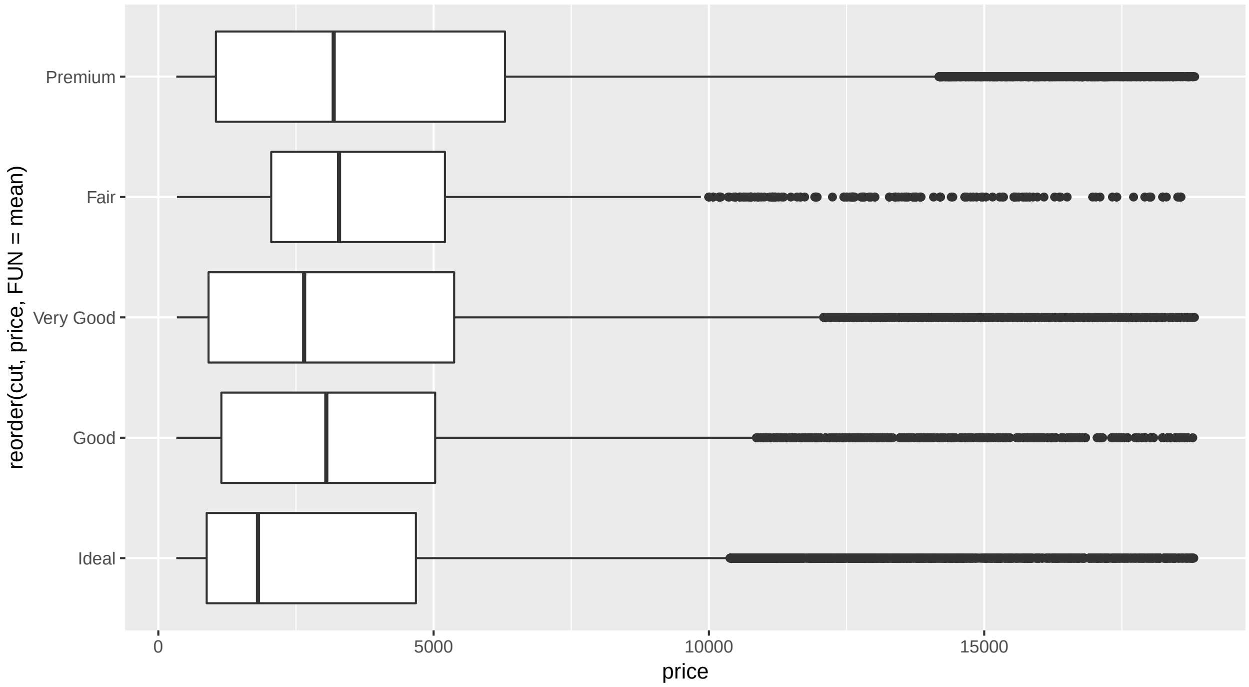


reorder()

```
# Reordering basing on average price from lower to higher
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = reorder(cut, price, FUN = mean), y = price))
```



```
# Flipping can be done if variable names are long
ggplot(data = diamonds) +
  geom_boxplot(mapping = aes(x = reorder(cut, price, FUN = mean), y = price)) +
  coord_flip()
```


Premium
Fair
Very Good
Good
Ideal
reorder(cut, price, FUN = mean)
0
5000
10000
15000
price