diamonds_analysis

João Calheiros

28/12/2021

```
library(tidyverse)
library(ggplot2)
library(scales)
library(ggthemes)
library(gridExtra)
```

Source

The *diamonds* data set is available with the *ggplot2* package. It contains approximately 54K observations and 10 variables, which include carat, cut, color, clarity, depth, table, price, x (length in mm), y (width in mm), and z (depth in mm). It is a clean data set, with no missing values.

Data Description

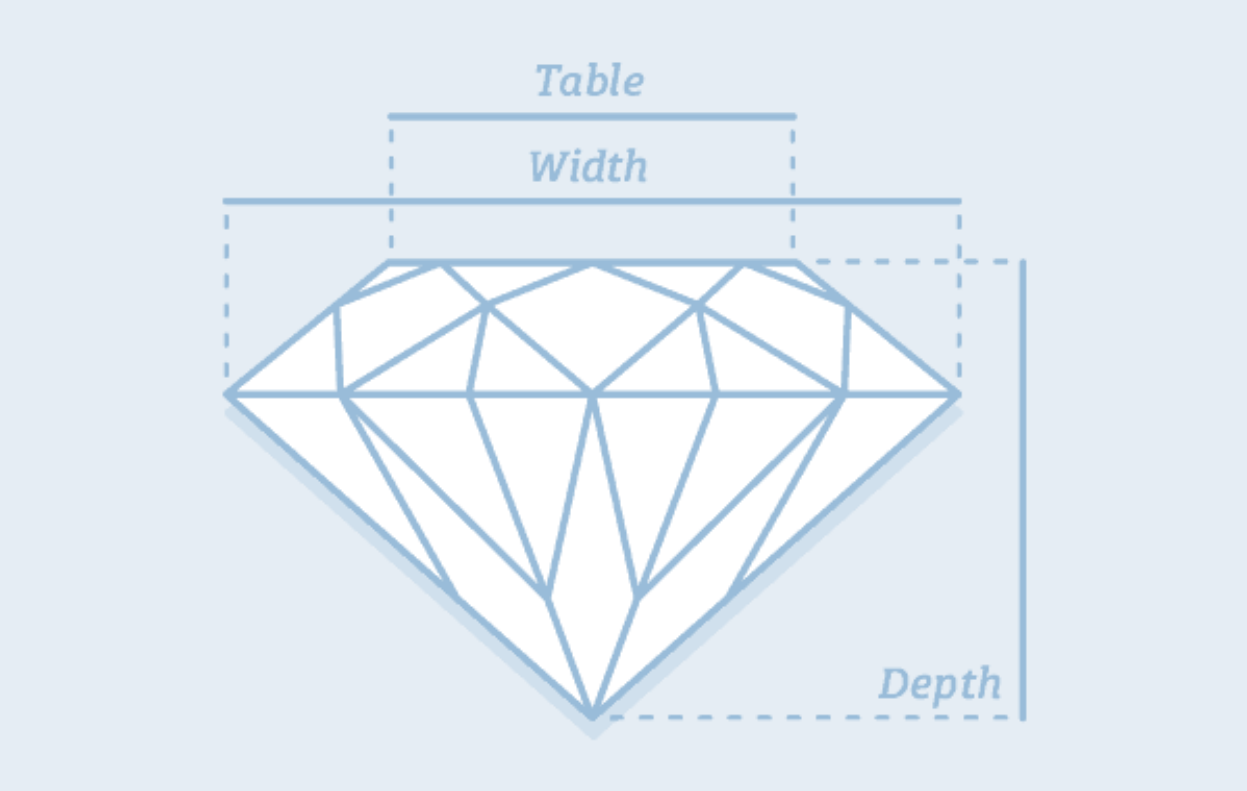
```
summary(diamonds)
```

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

```r
# First 10 lines
head(diamonds, 10)
```

```
## # A tibble: 10 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
```

- Notes:
- Diamond color include 7 types ranging from D (best) to J (worst).
- Diamond clarity include 8 types: I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best). In the summary, both IF and I1 (the extremes) are categorized as 'other', having approximately 2.5K observations both toegether.
- Length in mm as the variable x.
- Width in mm as the variable y.
- Depth in mm as the variable z.
- Small amounts of very expensive pieces drive the mean up. In this case the median is a better measurement variable of the center of the distribution.



1. Exploratory Analysis on Price (max, min, mean, median)

1.1 By cut, color and clarity

```
diamonds %>%
  group_by(cut) %>%
  summarise(max_price = max(price),
            min_price = min(price),
            mean_price = mean(price),
            median_price = median(price),
            count = n())
```

```
## # A tibble: 5 x 6
##   cut       max_price min_price mean_price median_price count
##   <ord>         <int>     <int>      <dbl>        <dbl> <int>
## 1 Fair          18574       337      4359.         3282   1610
## 2 Good          18788       327      3929.         3050.  4906
## 3 Very Good     18818       336      3982.         2648  12082
## 4 Premium       18823       326      4584.         3185  13791
## 5 Ideal         18806       326      3458.         1810  21551
```

```
diamonds %>%
  group_by(color) %>%
  summarise(max_price = max(price),
            min_price = min(price),
            mean_price = mean(price),
            median_price = median(price),
            count = n())
```

```
## # A tibble: 7 x 6
##   color max_price min_price mean_price median_price count
##   <ord>     <int>     <int>      <dbl>        <dbl> <int>
## 1 D         18693       357      3170.         1838   6775
## 2 E         18731       326      3077.         1739   9797
## 3 F         18791       342      3725.         2344.  9542
## 4 G         18818       354      3999.         2242  11292
## 5 H         18803       337      4487.         3460   8304
## 6 I         18823       334      5092.         3730   5422
## 7 J         18710       335      5324.         4234   2808
```

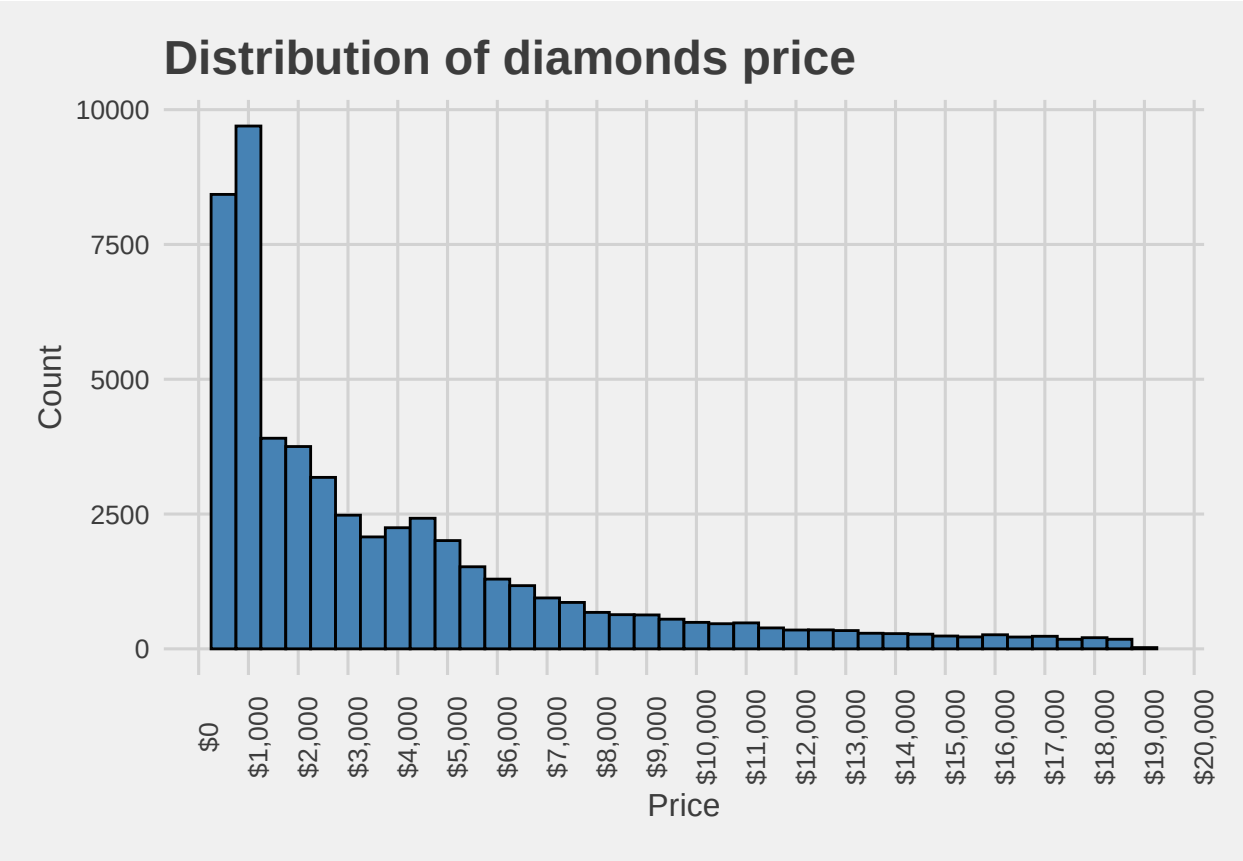
```
diamonds %>%
  group_by(clarity) %>%
  summarise(max_price = max(price),
            min_price = min(price),
            mean_price = mean(price),
            median_price = median(price),
            count = n())
```

```
## # A tibble: 8 x 6
##   clarity max_price min_price mean_price median_price count
##   <ord>       <int>     <int>      <dbl>        <dbl> <int>
## 1 I1          18531       345      3924.         3344   741
## 2 SI2         18804       326      5063.         4072  9194
## 3 SI1         18818       326      3996.         2822 13065
## 4 VS2         18823       334      3925.         2054 12258
## 5 VS1         18795       327      3839.         2005  8171
```

```
## 6 VVS2        18768       336      3284.         1311  5066
## 7 VVS1        18777       336      2523.         1093  3655
## 8 IF          18806       369      2865.         1080  1790
```

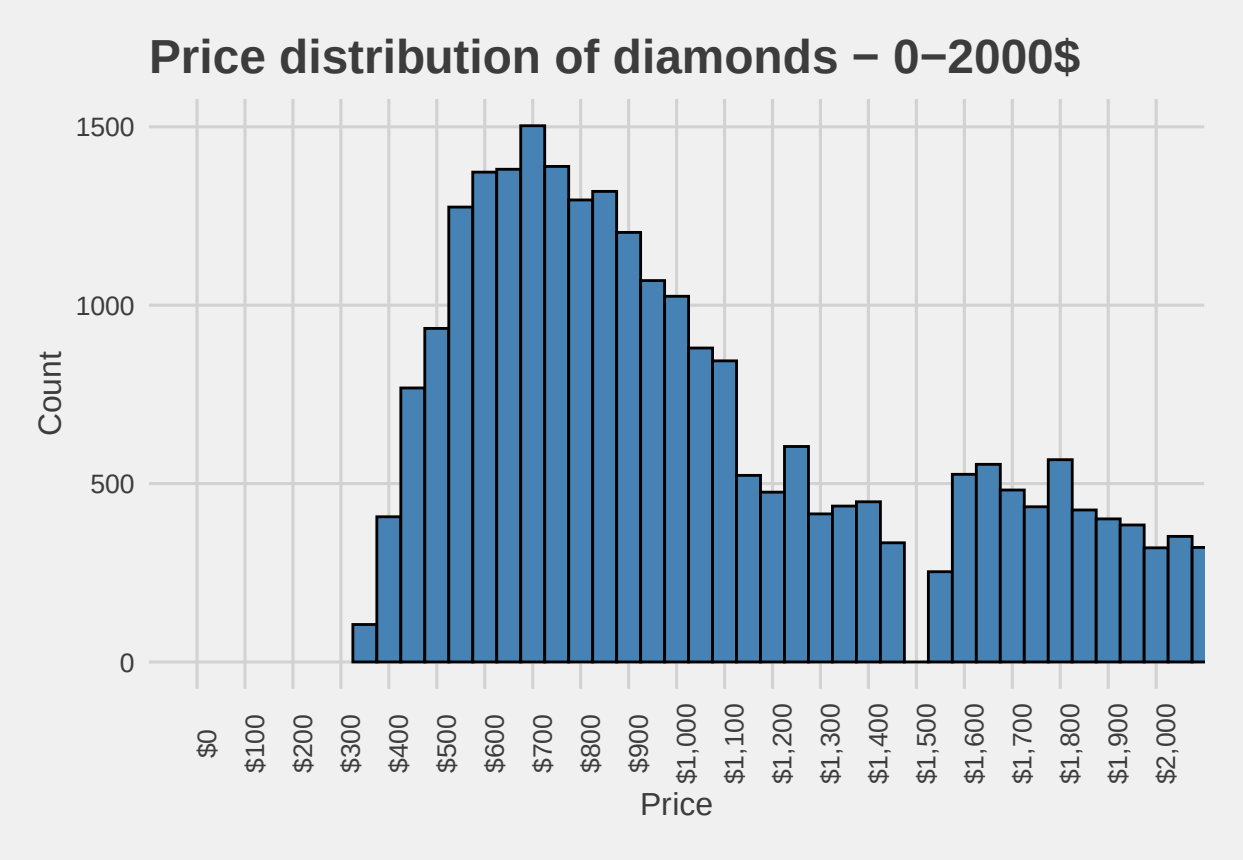
1.2 Price distribution

```
ggplot(diamonds, aes(x = price)) +
  geom_histogram(color = 'black', fill = 'SteelBlue', binwidth = 500) +
  scale_x_continuous(labels = dollar, breaks = seq(0, 20000, 1000)) +
  labs(title = 'Distribution of diamonds price',
       x = 'Price', y = 'Count') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text(angle=90))
```



1.2.1 Zoom in the largest peak of the price distribution

```
ggplot(diamonds, aes(x = price)) +
  geom_histogram(color = 'black', fill = 'SteelBlue', binwidth = 50) +
  scale_x_continuous(labels = dollar, breaks = seq(0, 2000, 100)) +
  labs(title = 'Price distribution of diamonds - 0-2000$',
       x = 'Price', y = 'Count') +
  coord_cartesian(c(0, 2000)) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text(angle=90))
```



1.3 Price related prints

How many diamonds are in the price range

Less than $500

```
diamonds %>%
  filter(price < 500) %>%
  summarise(count = n())
```

```
## # A tibble: 1 x 1
##   count
##   <int>
## 1  1729
```

Between $500 and $1000

```
diamonds %>%
  filter(price >= 500 & price < 1000) %>%
  summarise(count = n())
```

```
## # A tibble: 1 x 1
##   count
##   <int>
## 1 12770
```

Between $1000 and $1500

```
diamonds %>%
  filter(price >= 1000 & price < 1500) %>%
  summarise(count = n())
```

```
## # A tibble: 1 x 1
##   count
##   <int>
## 1  5511
```

Between $1500 and $2000

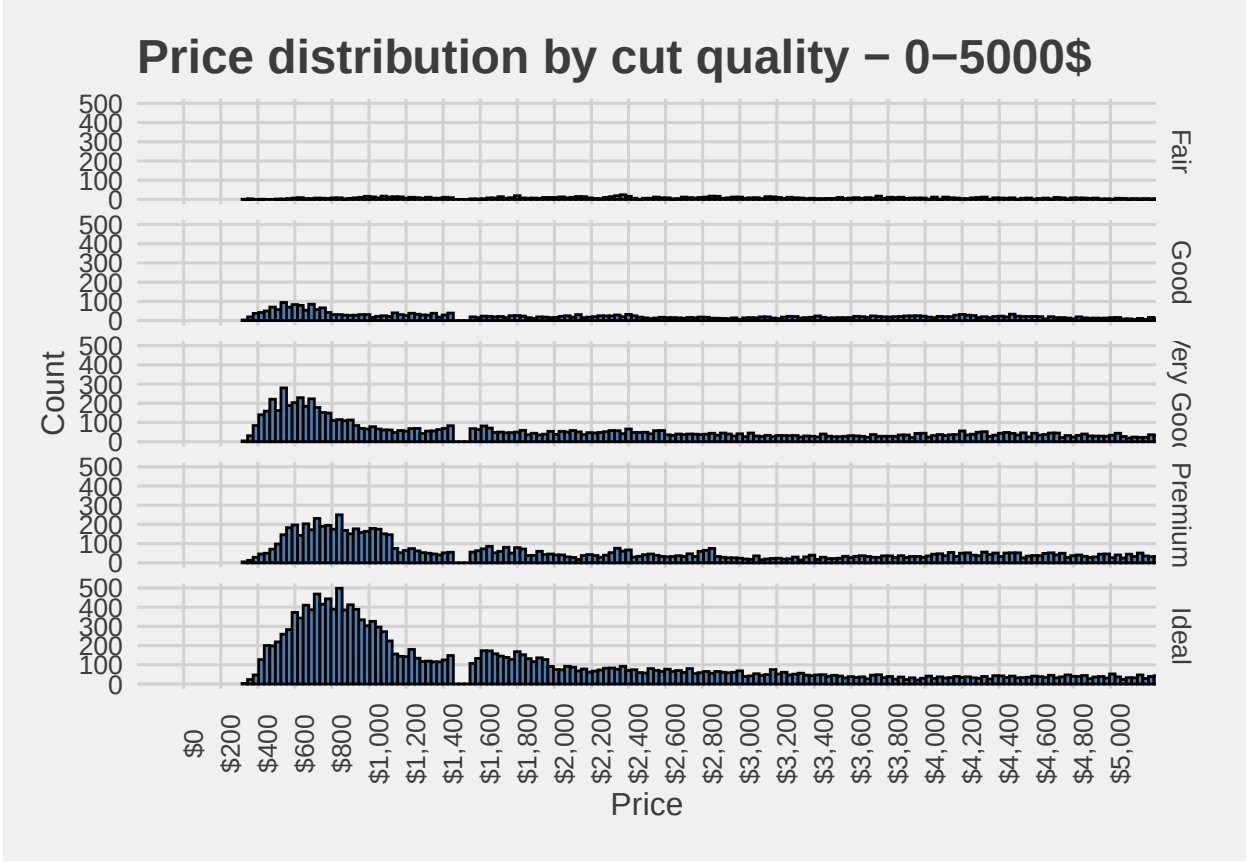
```
diamonds %>%
  filter(price >= 1500 & price < 2000) %>%
  summarise(count = n())
```

```
## # A tibble: 1 x 1
##   count
##   <int>
## 1  4193
```

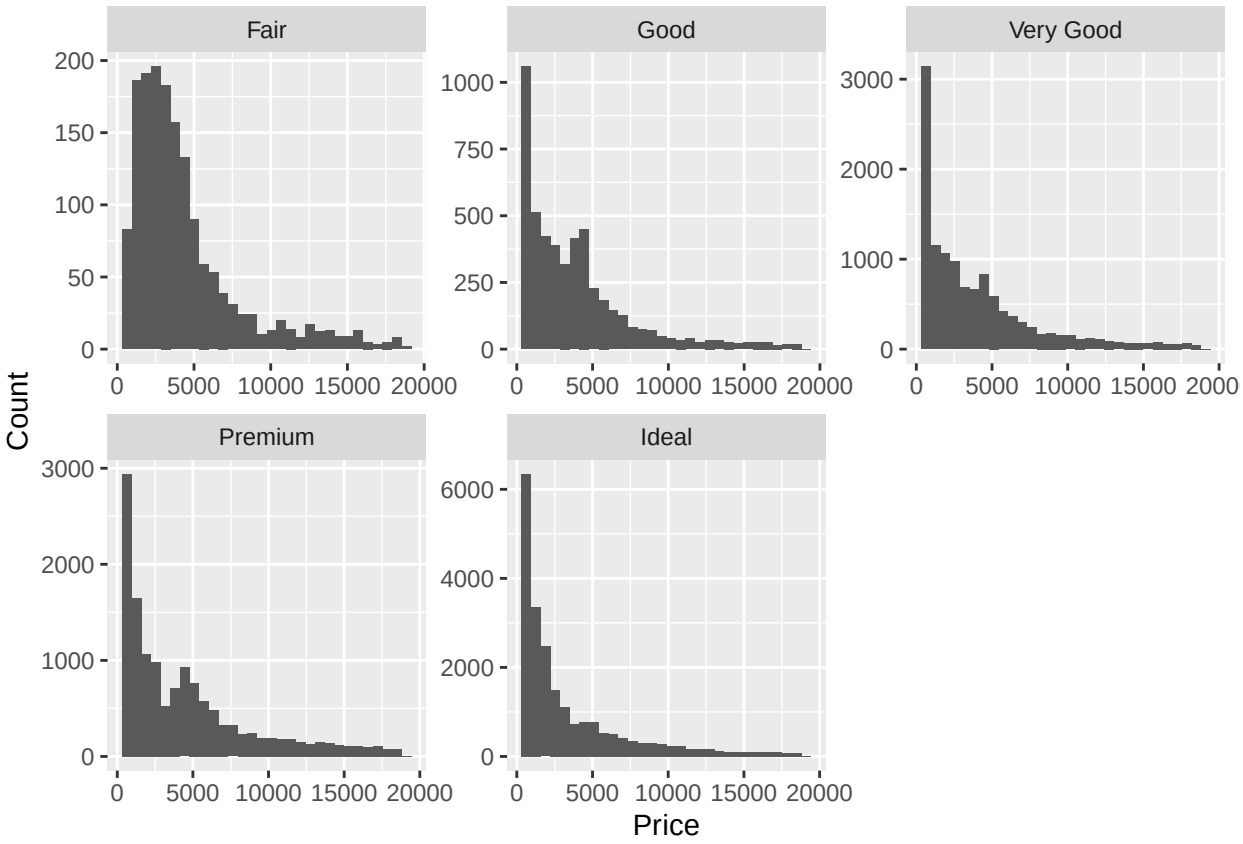
2. Further analysis on diamond prices

2.1 Price distribution per cut

```
ggplot(diamonds, aes(x=price)) +
  geom_histogram(color='black', fill='SteelBlue', binwidth = 30) +
  scale_x_continuous(labels=dollar, breaks=seq(0, 5000, 200)) +
  labs(title='Price distribution by cut quality - 0-5000$',
       x = 'Price', y = 'Count') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text(angle=90)) +
  coord_cartesian(c(0, 5000)) +
  facet_grid(cut~.)
```




```
qplot(x = price, data=diamonds) +
  facet_wrap(~cut, scales = 'free') +
  labs(x = 'Price', y = 'Count')
```



2.2 Summaries on the prices per quality cut

```
cut_price <- data.frame(diamonds$cut, diamonds$price)

fair <- cut_price %>%
  filter(cut_price$diamonds.cut == 'Fair')
summary(fair$diamonds.price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     337    2050    3282    4359    5206   18574
```

```
good <- cut_price %>%
  filter(cut_price$diamonds.cut == 'Good')
summary(good$diamonds.price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     327    1145    3050    3929    5028   18788
```

```
v_good <- cut_price %>%
  filter(cut_price$diamonds.cut == 'Very Good')
summary(v_good$diamonds.price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     336     912    2648    3982    5373   18818
```

```
premium <- cut_price %>%
  filter(cut_price$diamonds.cut == 'Premium')
summary(premium$diamonds.price)
```

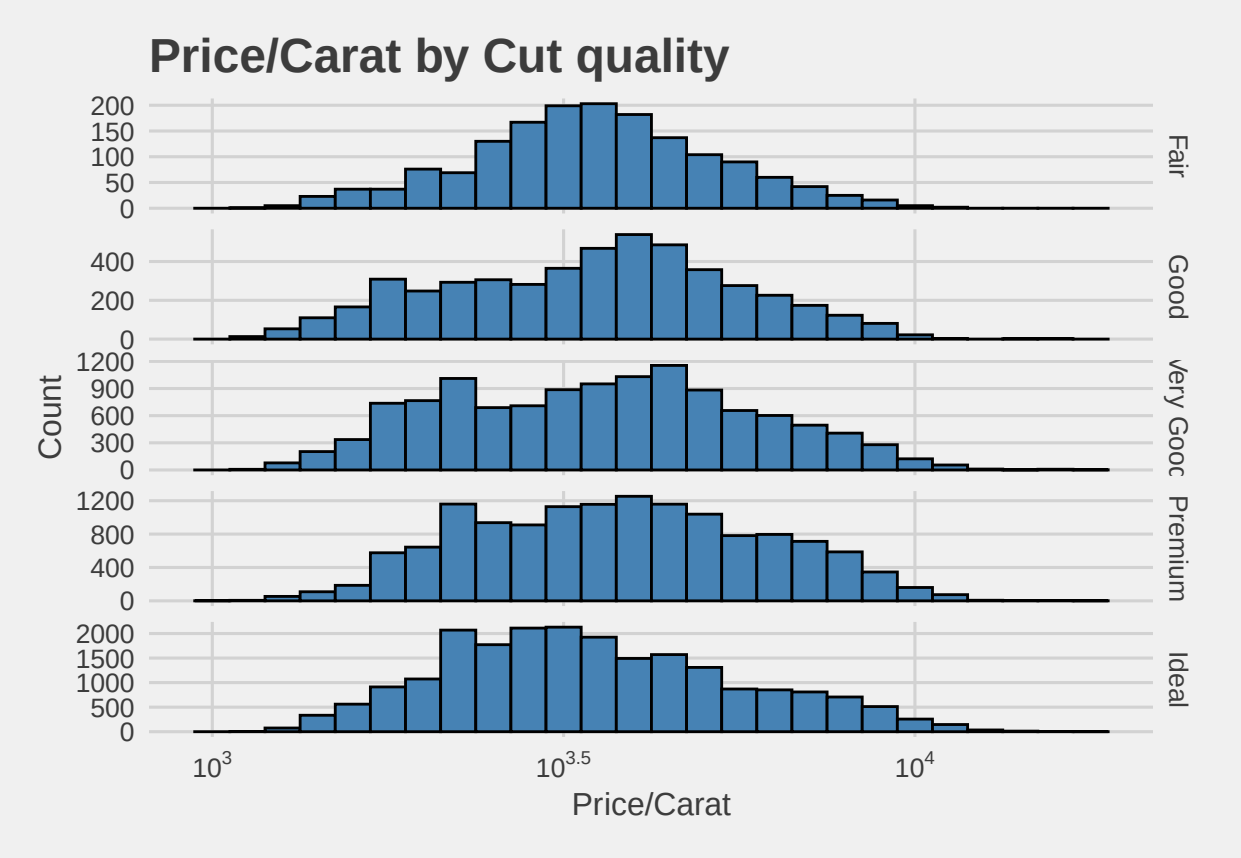
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     326    1046    3185    4584    6296   18823
```

```
ideal <- cut_price %>%
  filter(cut_price$diamonds.cut == 'Ideal')
summary(ideal$diamonds.price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     326     878    1810    3458    4678   18806
```

2.3 Price per carat by cut quality

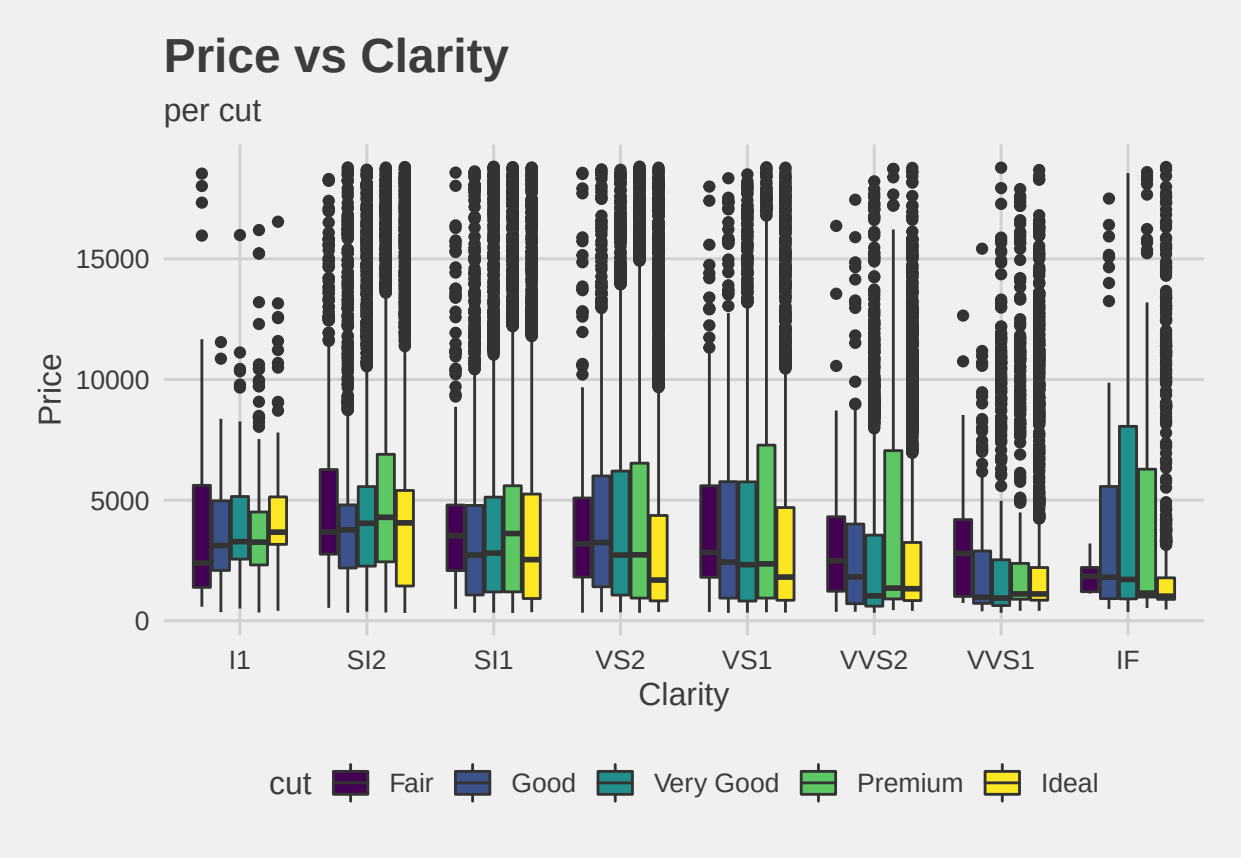
```
ggplot(diamonds, aes(x=price/carat)) +
  geom_histogram(color='black', fill='SteelBlue', binwidth=0.05) +
  labs(title='Price/Carat by Cut quality',
       x = 'Price/Carat', y = 'Count') +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", math_format(10^.x))) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text()) +
  facet_grid(cut~., scale = "free")
```



2.4 Price vs clarity for all cuts

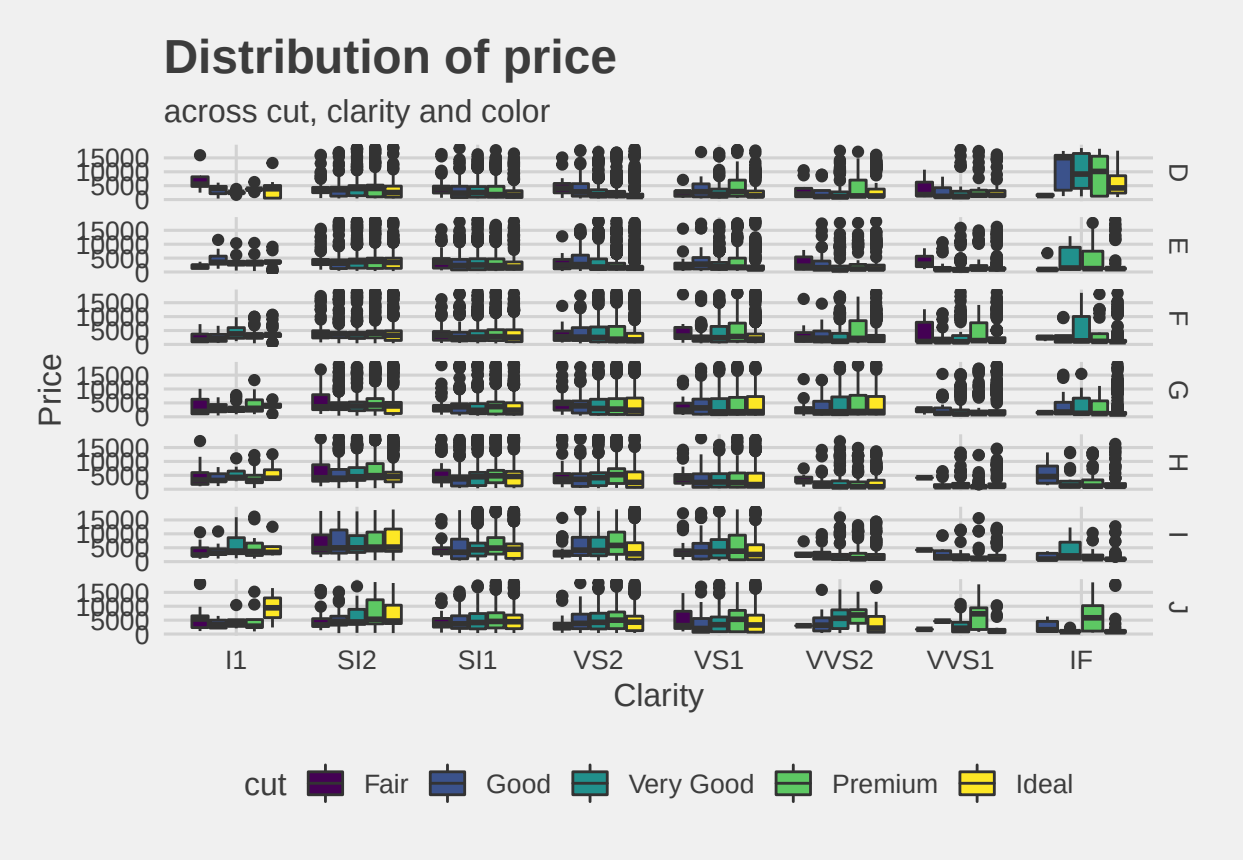
```
ggplot(diamonds, aes(x=clarity, y=price, fill=cut)) +
  geom_boxplot() +
```

```
  labs(title='Price vs Clarity',
       subtitle='per cut',
       x = 'Clarity', y = 'Price') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text())
```



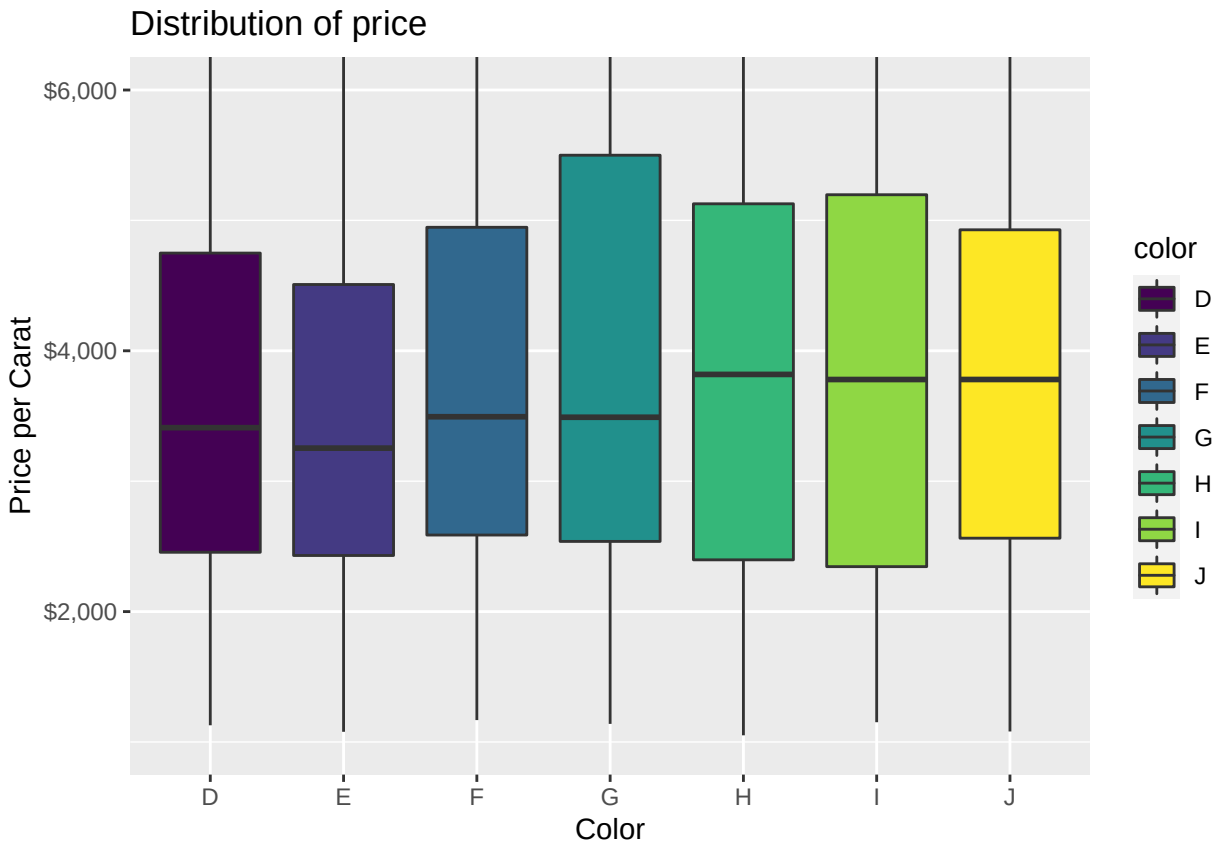
2.5 Matrix with price vs clarity for all cuts per color

```
ggplot(diamonds, aes(x=clarity, y=price, fill=cut)) +
  geom_boxplot() +
  labs(x = 'Clarity', y = 'Price',
       title='Distribution of price',
       subtitle='across cut, clarity and color') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text()) +
  facet_grid(color~.)
```



2.6 Price per carat of diamonds across the different colors

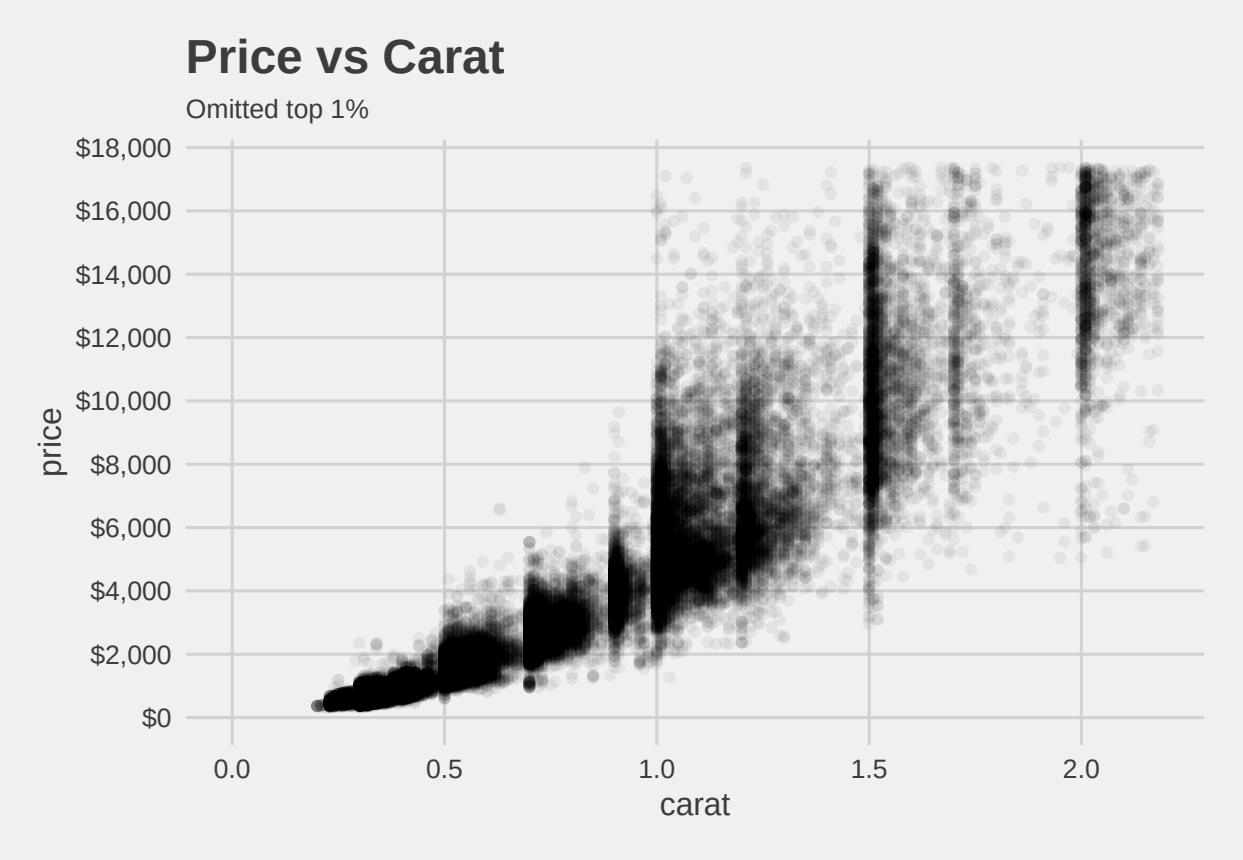
```
ggplot(diamonds, aes(x=color, y=price/carat, fill=color)) +
  geom_boxplot() +
  labs(title='Distribution of price',
       subsitle='across colors',
       x='Color',
       y='Price per Carat') +
  coord_cartesian(ylim=c(1000, 6000)) +
  scale_y_continuous(labels=dollar)
```



2.7 Price vs carat with top 1% of both omitted

```
ggplot(diamonds, aes(x=carat, y=price)) +
  geom_point(alpha=.05) +
  labs(title='Price vs Carat',
       subtitle='Omitted top 1%') +
  scale_x_continuous(limits=c(0, quantile(diamonds$carat, 0.99))) +
  scale_y_continuous(breaks=seq(0, 18000, 2000),
                     limits=c(0, quantile(diamonds$price, 0.99)),
                     labels=dollar) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text()) +
  theme(plot.subtitle = element_text(size=10))
```

```
## Warning: Removed 926 rows containing missing values (geom_point).
```



Takeaways

1. **Cut**

- The number of *Ideal* diamonds(21551) is much larger than the others but its average price($3458) its also the lowest;
- The number of *Fair* diamonds(1610) is the lowest and its average price($4359) its the second highest;
- The average price for *Premium* diamonds($4584) is the highest for all cuts;

2. **Color**

- Color *G* is the most represented in the data set - 11292 occurrences;
- Color *J* is the least represented in the data set - 2808 occurrences - while still having the highest average price: $5324;
- The lowest average price is for the color *E*: $3077;

3. **Clarity**

- Clarity of *SI1* and *VS2* are the most represented with 13065 and 12258 occurrences respectively;
- The highest average price belongs to clarity *SI2* - $5063 - which is greater than all the others by at least $1000;
- *I1*, *SI1*, *VS2* and *VS1* average prices range between $3800-$3900;
- The lowest average price belongs to *VVS1* with a value of $2523.

3. Interquartile range (IQR)

3.1 For the *best* and *worst* colors

```
diamonds %>%
  group_by(color) %>%
  filter(color == 'D') %>%
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   color quartile_25 quartile_75   IQR
##   <ord>       <dbl>       <dbl> <dbl>
## 1 D             911       4214. 3302.
```

```
diamonds %>%
  group_by(color) %>%
  filter(color == 'J') %>%
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   color quartile_25 quartile_75   IQR
##   <ord>       <dbl>       <dbl> <dbl>
## 1 J           1860.        7695 5834.
```

3.2 For the *best* and *worst* clarities

```
diamonds %>%
  group_by(clarity) %>%
  filter(clarity == 'I1') %>%
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   clarity quartile_25 quartile_75   IQR
##   <ord>         <dbl>       <dbl> <dbl>
## 1 I1             2080        5161  3081
```

```
diamonds %>%
  group_by(clarity) %>%
  filter(clarity == 'IF') %>%
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   clarity quartile_25 quartile_75   IQR
##   <ord>         <dbl>       <dbl> <dbl>
## 1 IF              895       2388. 1494.
```

3.3 For the *best* and *worst* cuts

```
diamonds %>%
  group_by(cut) %>%
  filter(cut == 'Fair') %>%
```

```
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   cut   quartile_25 quartile_75   IQR
##   <ord>       <dbl>       <dbl> <dbl>
## 1 Fair        2050.       5206. 3155.
```

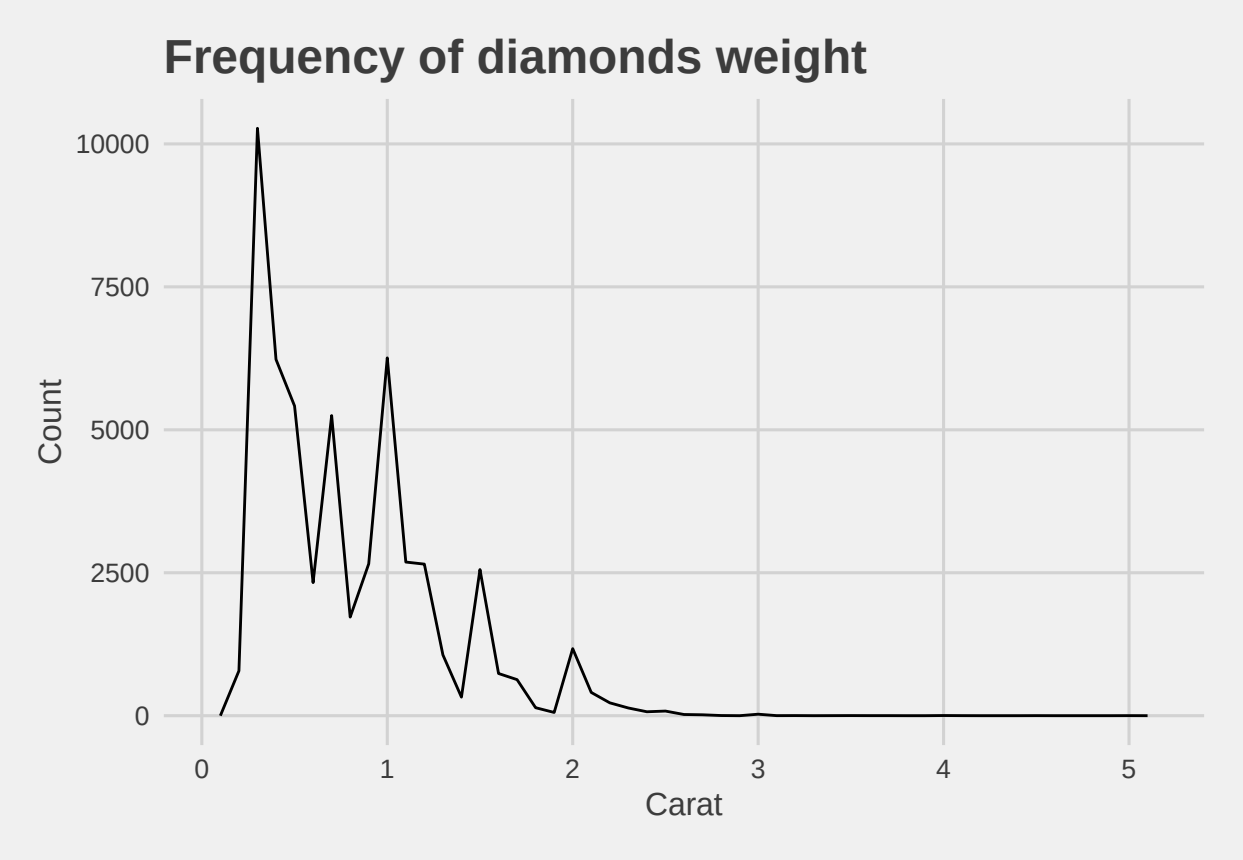
```
diamonds %>%
  group_by(cut) %>%
  filter(cut == 'Ideal') %>%
  summarise(quartile_25 = quantile(price, 0.25),
            quartile_75 = quantile(price, 0.75),
            IQR = quartile_75 - quartile_25)
```

```
## # A tibble: 1 x 4
##   cut   quartile_25 quartile_75   IQR
##   <ord>       <dbl>       <dbl> <dbl>
## 1 Ideal         878       4678. 3800.
```

4. Distribution of the weight (carat) of diamonds

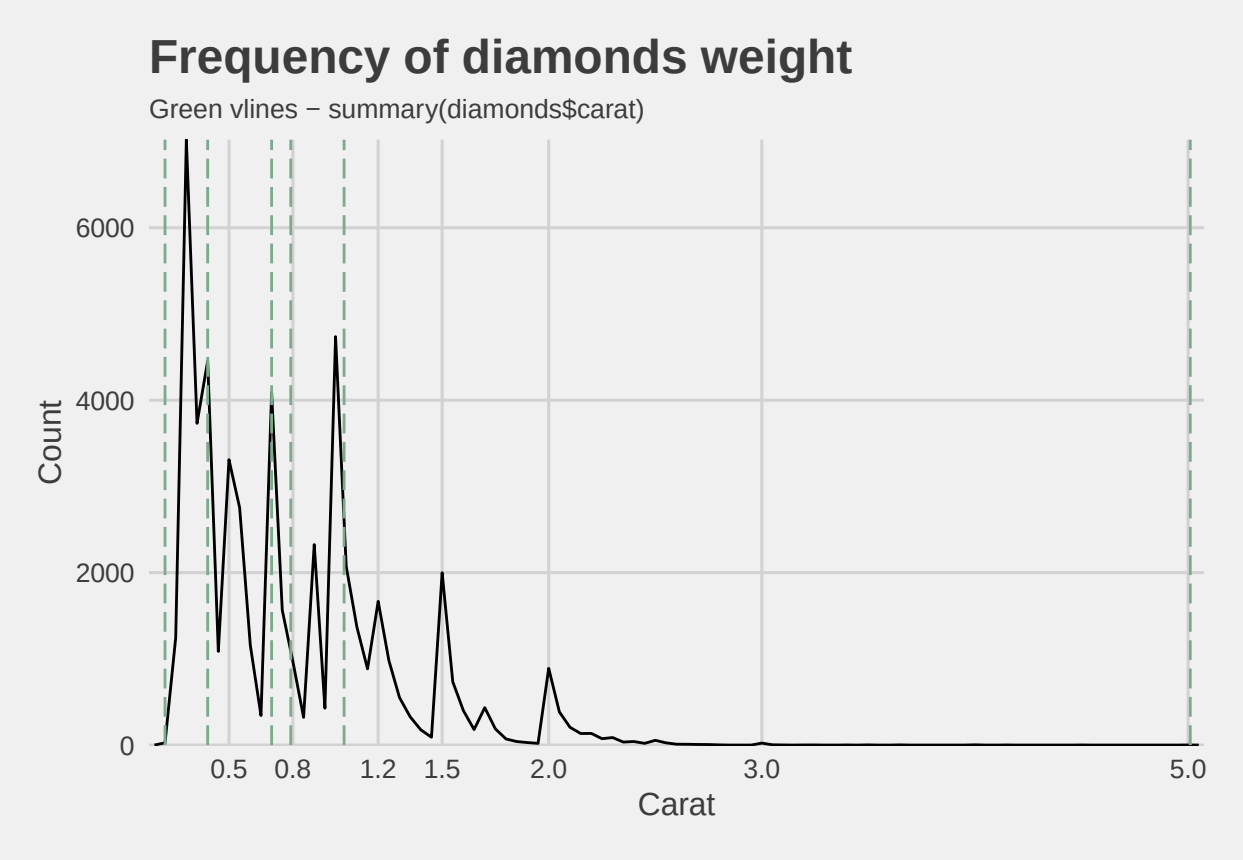
4.1 Frequency polygon

```
ggplot(diamonds, aes(x=carat)) +
  geom_freqpoly(binwidth=0.1) +
  scale_x_continuous() +
  scale_y_continuous() +
  labs(title='Frequency of diamonds weight',
       subsitle='across colors',
       x='Carat',
       y='Count') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text())
```



4.2 Frequency polygon (in more depth)

```
ggplot(diamonds, aes(x=carat)) +
  geom_freqpoly(binwidth=0.05) +
  scale_x_continuous(breaks=c(0.1, 0.5, 0.8, 1.2, 1.5, 2.0, 3.0, 5.0), expand = c(0,0)) +
  scale_y_continuous(expand=c(0,0)) +
  geom_vline(xintercept=c(0.2, 0.4, 0.7, 0.79, 1.04, 5.01), color='#7daa8e', linetype='longdash') +
  labs(title='Frequency of diamonds weight',
       subtitle='Green vlines - summary(diamonds$carat)',
       x='Carat',
       y='Count') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text()) +
  theme(plot.subtitle = element_text(size=10))
```




```
summary(diamonds$carat)
```

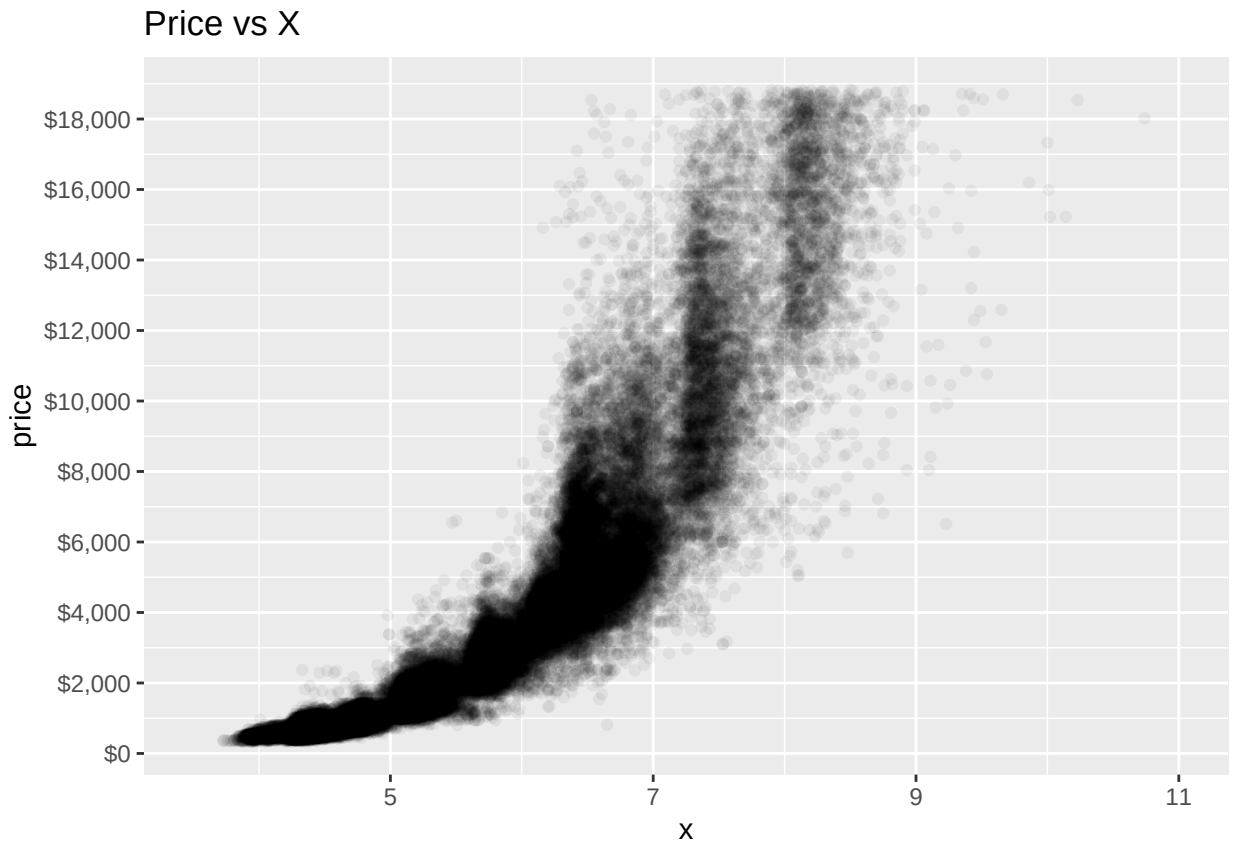
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2000  0.4000  0.7000  0.7979  1.0400  5.0100
```

5. Correlation of price and x, y and z

5.1 Plots

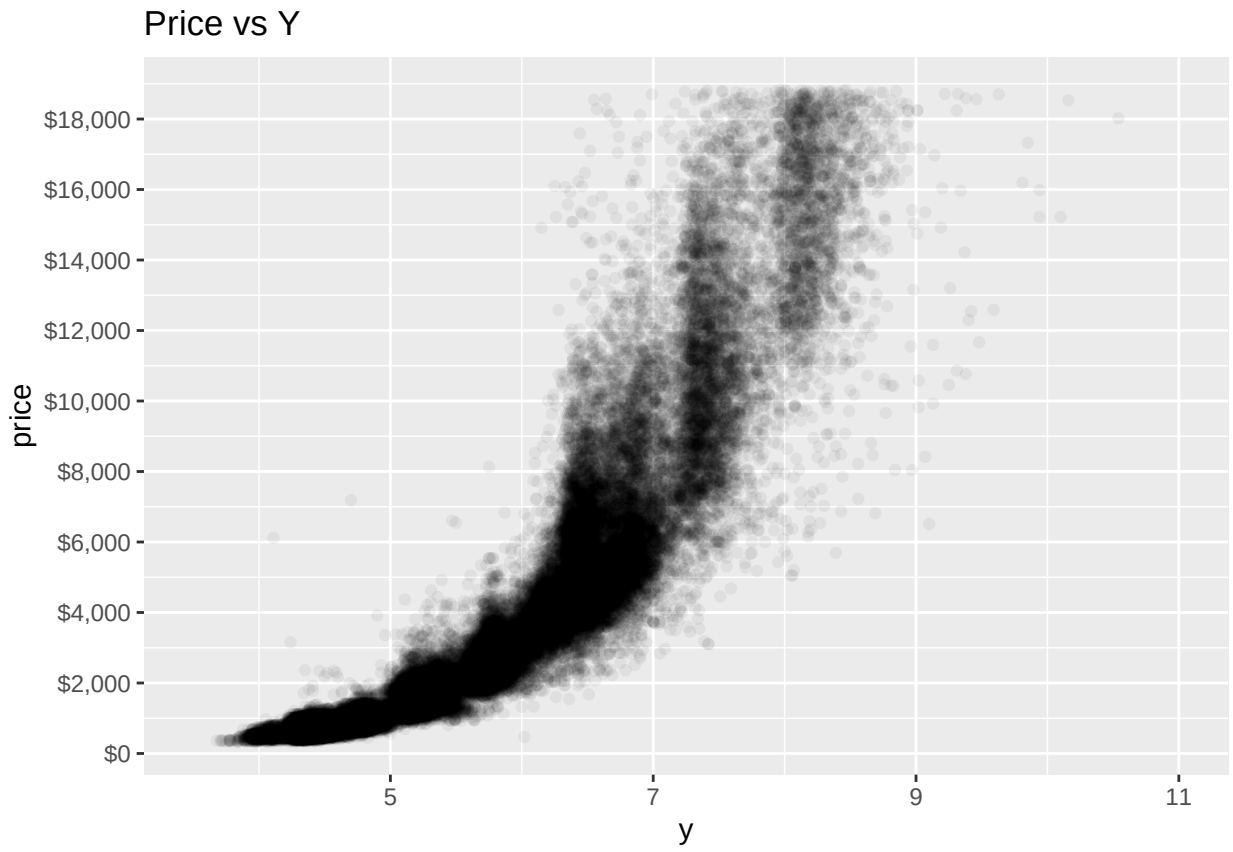
5.1.1 Scatter plot of price vs x

```
ggplot(diamonds, aes(x=x, y=price)) +
  geom_point(alpha=.05) +
  coord_cartesian(xlim=c(3.5, 11))+
  labs(title='Price vs X') +
  scale_y_continuous(breaks=seq(0, 20000, 2000), label=dollar)
```



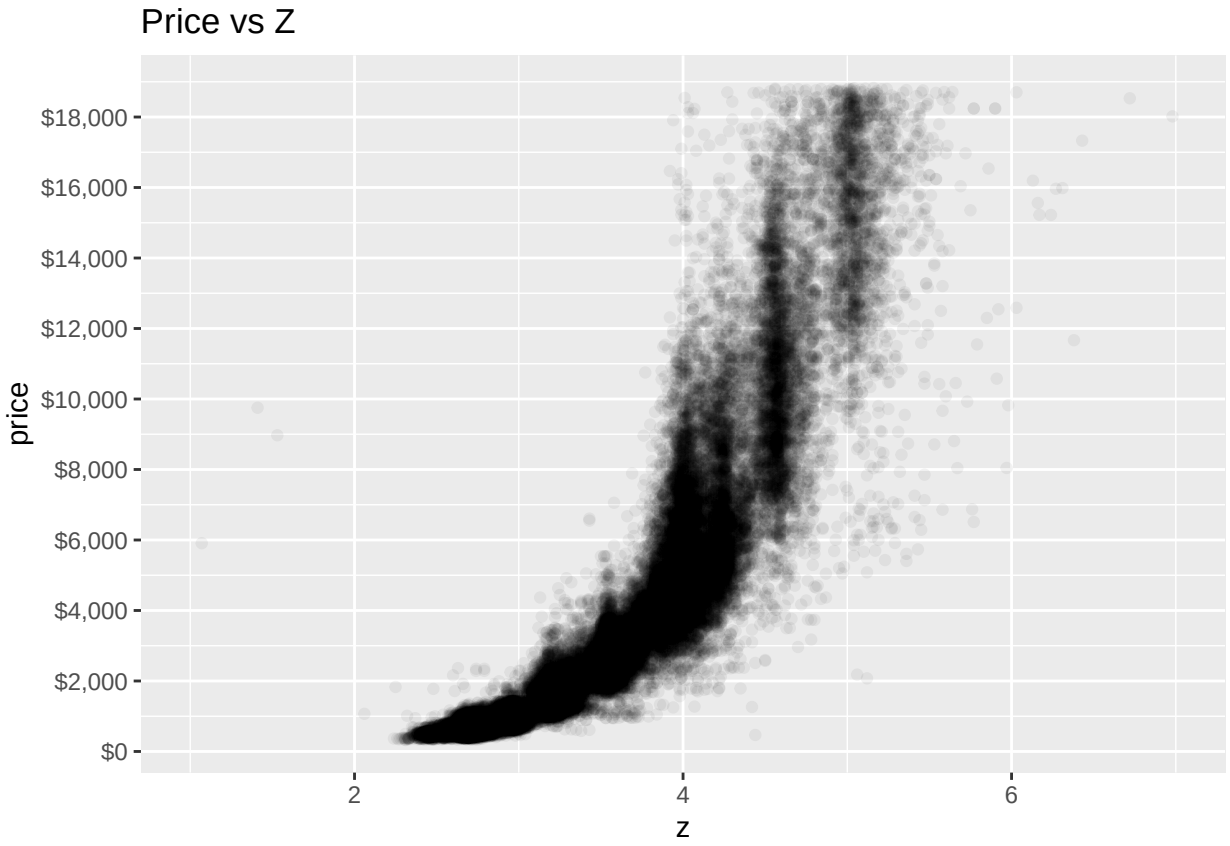
5.1.2 Scatter plot of price vs y

```
ggplot(diamonds, aes(x=y, y=price)) +
  geom_point(alpha=.05) +
  coord_cartesian(xlim=c(3.5, 11))+
  labs(title='Price vs Y') +
  scale_y_continuous(breaks=seq(0, 20000, 2000), label=dollar)
```



5.1.3 Scatter plot of price vs z

```
ggplot(diamonds, aes(x=z, y=price)) +
  geom_point(alpha=.05) +
  coord_cartesian(xlim=c(1, 7))+
  labs(title='Price vs Z') +
  scale_y_continuous(breaks=seq(0, 20000, 2000), label=dollar)
```



- Note: Data appears to have an artificial barrier at $19 000

5.2 Correlation measures

```
# For the p-value, a typical threshold is 0.05, anything smaller counts as statistically significant

# For price vs X
with(diamonds, cor.test(price, x))
```

```
##
##  Pearson's product-moment correlation
##
## data:  price and x
## t = 440.16, df = 53938, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8825835 0.8862594
## sample estimates:
##       cor
## 0.8844352
```

```
# For price vs Y
with(diamonds, cor.test(price, y))
```

```
##
##  Pearson's product-moment correlation
##
## data:  price and y
```

```
## t = 401.14, df = 53938, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8632867 0.8675241
## sample estimates:
##       cor
## 0.8654209
```

```
# For price vs Z
with(diamonds, cor.test(price, z))
```

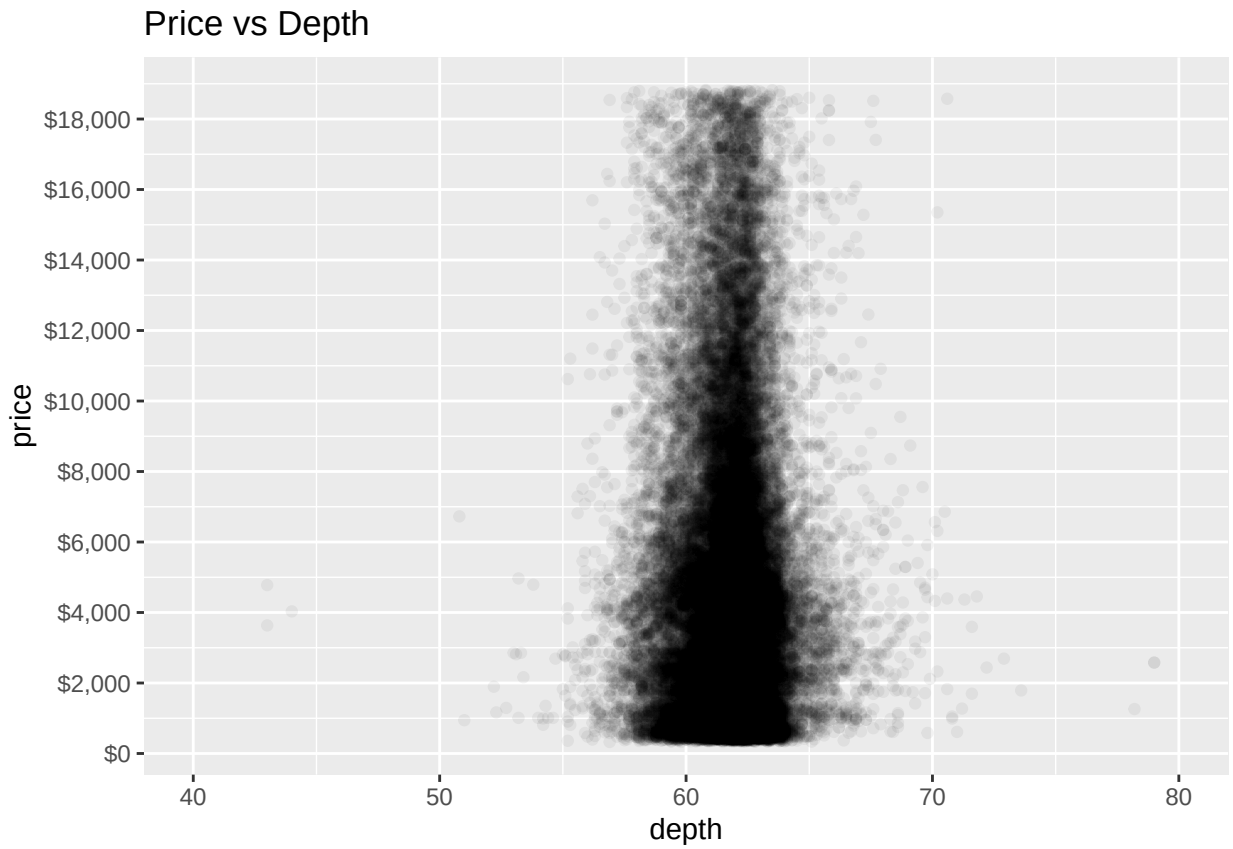
```
##
##  Pearson's product-moment correlation
##
## data:  price and z
## t = 393.6, df = 53938, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8590541 0.8634131
## sample estimates:
##       cor
## 0.8612494
```

- Note: There is a strong correlation between the price and the x, y and z values

6. Depth

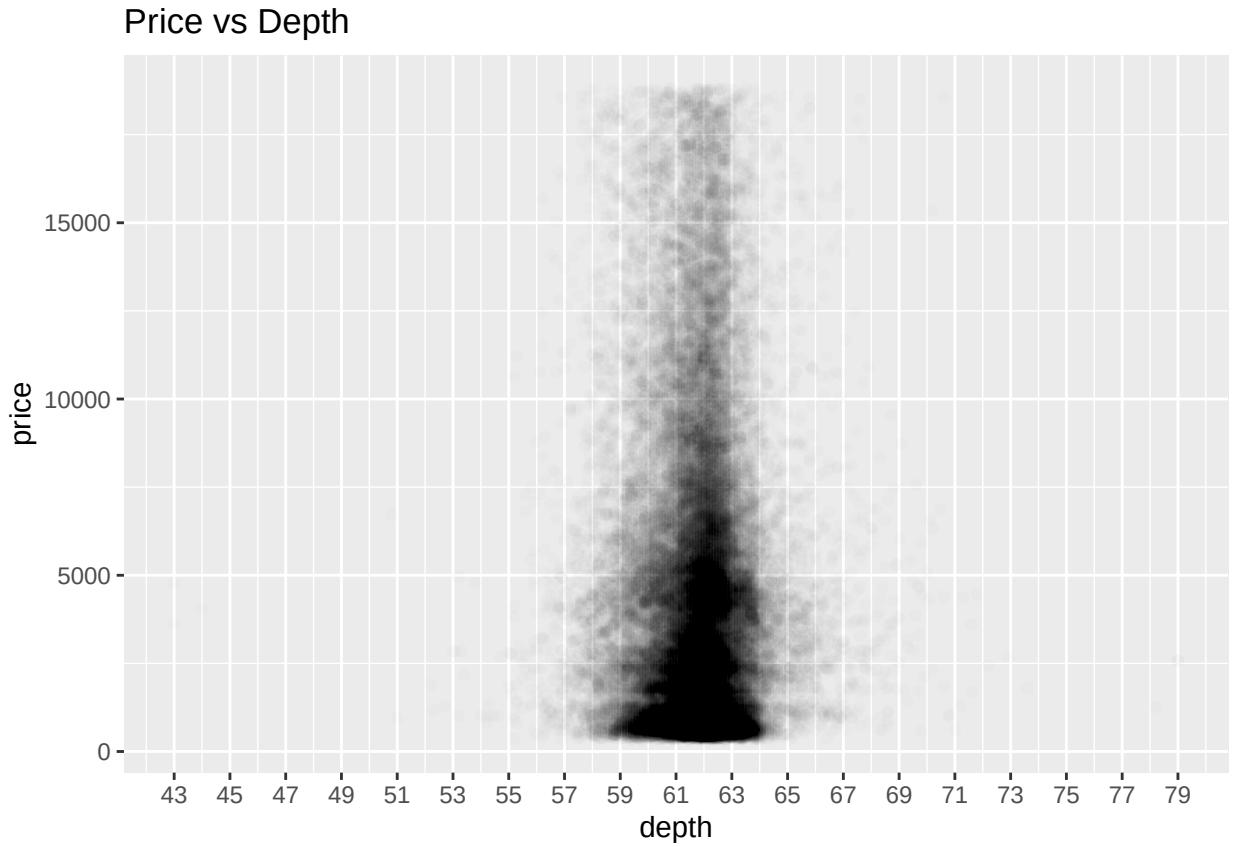
6.1 Scatter Plot of price vs depth

```
ggplot(diamonds, aes(x=depth, y=price)) +
  geom_point(alpha=.05) +
  coord_cartesian(xlim=c(40, 80))+
  labs(title='Price vs Depth') +
  scale_y_continuous(breaks=seq(0, 20000, 2000), label=dollar)
```



6.2 Add transparency and labels every 2 units

```
ggplot(diamonds, aes(x=depth, y=price)) +
  labs(title='Price vs Depth') +
  geom_point(alpha=1/100) +
  scale_x_continuous(breaks=seq(min(diamonds$depth), max(diamonds$depth), 2),
                     labels=seq(min(diamonds$depth), max(diamonds$depth), 2))
```




```
with(diamonds,  cor.test(price, depth))
```

```
##
##  Pearson's product-moment correlation
##
## data:  price and depth
## t = -2.473, df = 53938, p-value = 0.0134
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.019084756 -0.002208537
## sample estimates:
##        cor
## -0.0106474
```

- Note: Depth does not influence the price of diamonds. That is visible in the plots and also because p-value = 0.0134 and cor = -0.0106474.

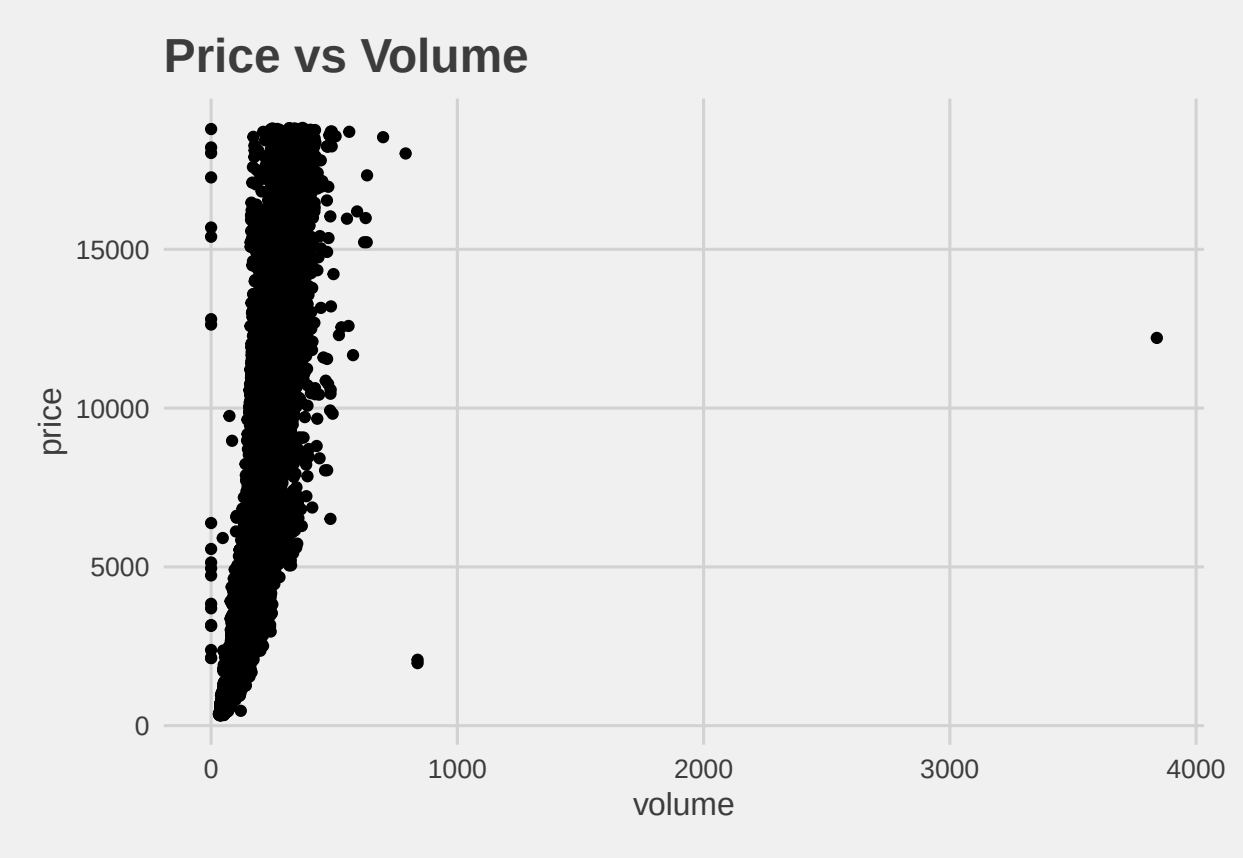
7. Price vs Volume

- Note: this is a rough approximation of a diamond's volume

```
diamonds_v <- diamonds %>%
  mutate(volume=x*y*z)

diamonds_v2 <- diamonds_v %>%
  filter(volume != 0,
         volume <= 800)
```

7.1 Scatter Plot of price vs volume

```
ggplot(diamonds_v, aes(x=volume, y=price)) +
  geom_point() +
  labs(title='Price vs Volume') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text())
```



- Notes:
- Volumes ranging from 0-500 hold the huge majority of diamonds, whatever the price
- Volumes ranging from 1000-3000 seem to be nonexistent
- 3 outliers 1 of which expands the plot(horizontaly) by a very significant size
- Diamonds with volumes near 0 exist

7.2 Correlation of price and volume

- Note: Excluding diamonds with volume of 0 or greater than 800

```
with(subset(diamonds_v, !(volume == 0 | volume >= 800)), cor.test(price, volume))
```

```
##
##  Pearson's product-moment correlation
##
## data:  price and volume
## t = 559.19, df = 53915, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9222944 0.9247772
```

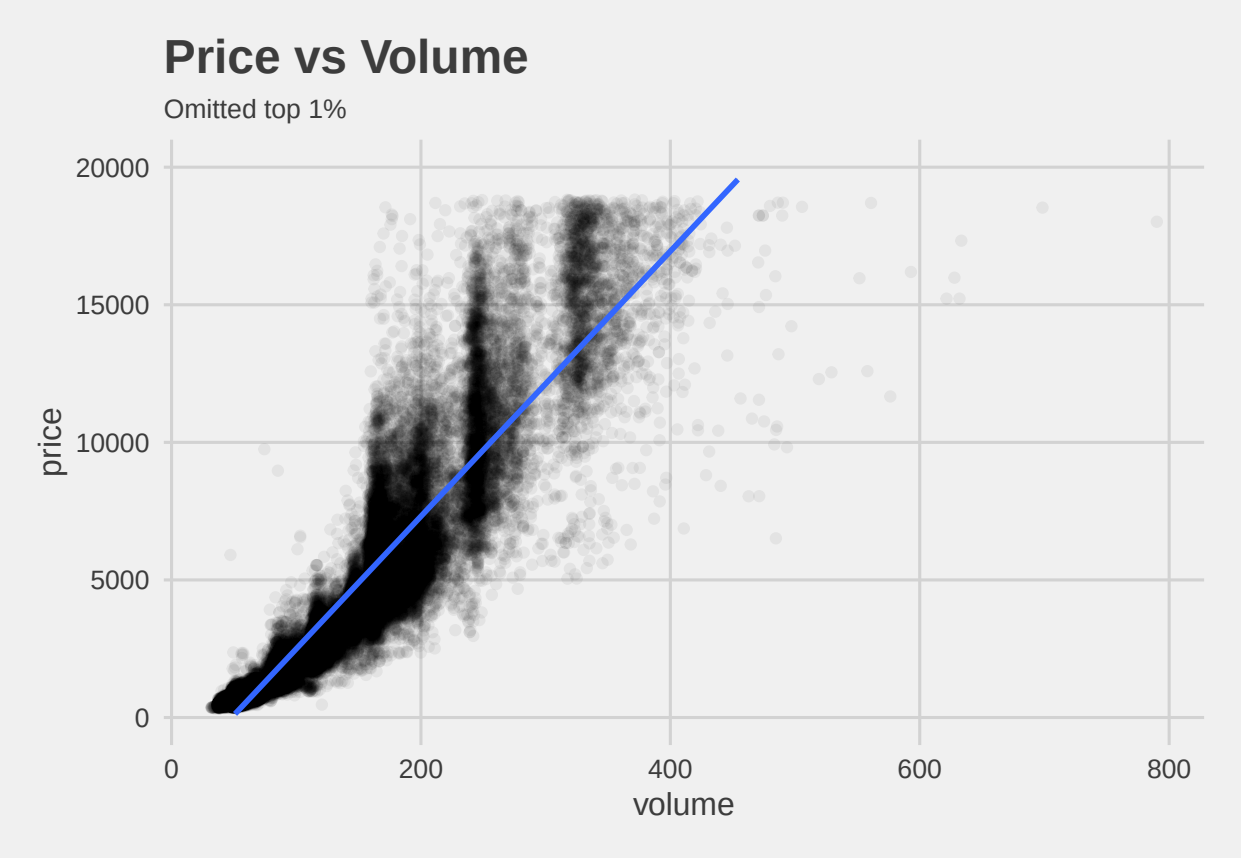
```
## sample estimates:
##       cor
## 0.9235455
```

7.3 Scatter Plot of price vs volume (omitted top 1%)

```
ggplot(diamonds_v2, aes(x=volume, y=price)) +
  geom_point(alpha=.05) +
  geom_smooth(method='lm') +
  labs(title='Price vs Volume',
       subtitle='Omitted top 1%') +
  scale_y_continuous(limits=c(0, 20000)) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  theme(plot.subtitle = element_text(size=10))
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 37 rows containing missing values (geom_smooth).
```



8. Information on diamonds by clarity

- I created a new data frame with the following columns:
- mean_price
- median_price
- min_price
- max_price

- n (number of diamonds on each level of clarity)

```
diamondsByClarity <- diamonds %>%
  group_by(clarity) %>%
  summarise(mean_price = mean(price),
            median_price = median(price),
            min_price = min(price),
            max_price = max(price),
            count = n()) %>%
  arrange(clarity)
```

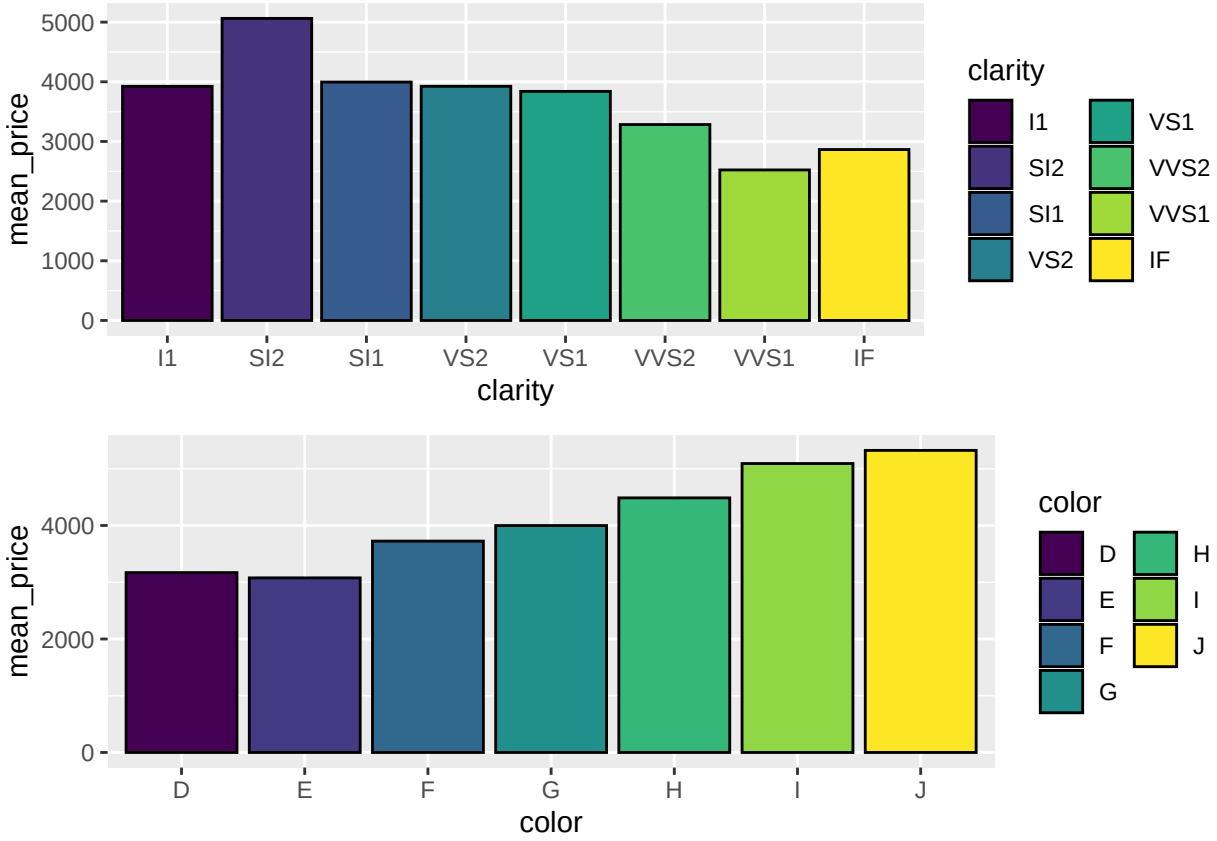
8.1 Mean price vs color and clarity

8.1.1 Create summary dataframes

```
# By clarity and color separately
diamonds_mp_cla <- diamonds %>%
  group_by(clarity) %>%
  summarise(mean_price = mean(price))

diamonds_mp_col <- diamonds %>%
  group_by(color) %>%
  summarise(mean_price = mean(price))
```

8.1.2 Plot the summary dataframes

```
# Using library(gridExtra)
plt1 <- ggplot(diamonds_mp_cla, aes(x=clarity, y=mean_price, fill=clarity)) +
  geom_bar(stat = "identity", color='black') +
  guides(fill=guide_legend(ncol=2))

plt2 <- ggplot(diamonds_mp_col, aes(x=color, y=mean_price, fill=color)) +
  geom_bar(stat = "identity", color='black') +
  guides(fill=guide_legend(ncol=2))

grid.arrange(plt1, plt2)
```



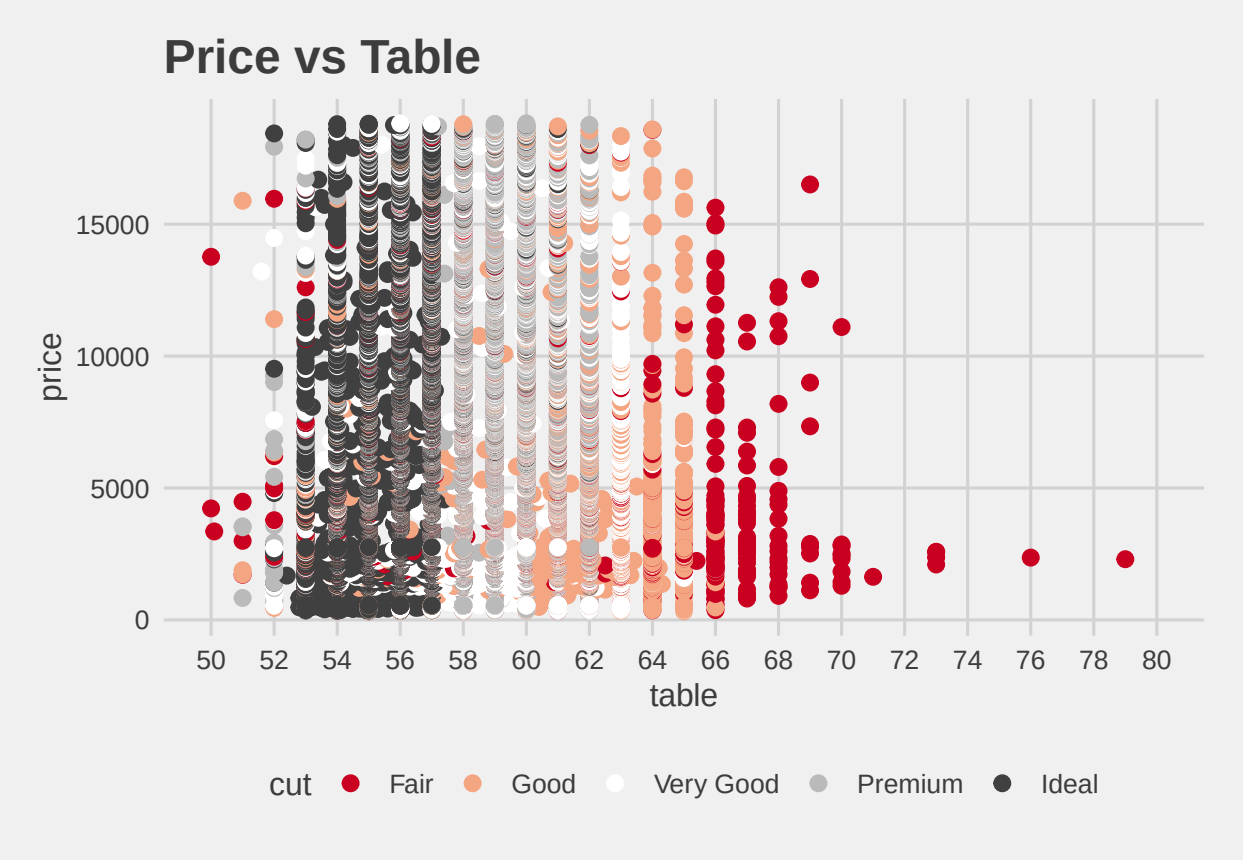
- Notes:
- Upward trend in average price as color goes from D to J;
- Downward trend in average price as clarity goes from I1 to IF

9. Aditional plots

9.1 Price vs table by color

```
ggplot(diamonds, aes(x=table, y=price, color=cut)) +
  geom_point(size=2.5) +
  labs(title='Price vs Table',
       x='table',
       y='price') +
  scale_x_continuous(breaks=seq(50, 80, 2),
                     limits=c(50, 80)) +
  scale_color_brewer(palette = 'RdGy') +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(), axis.text.x = element_text())
```

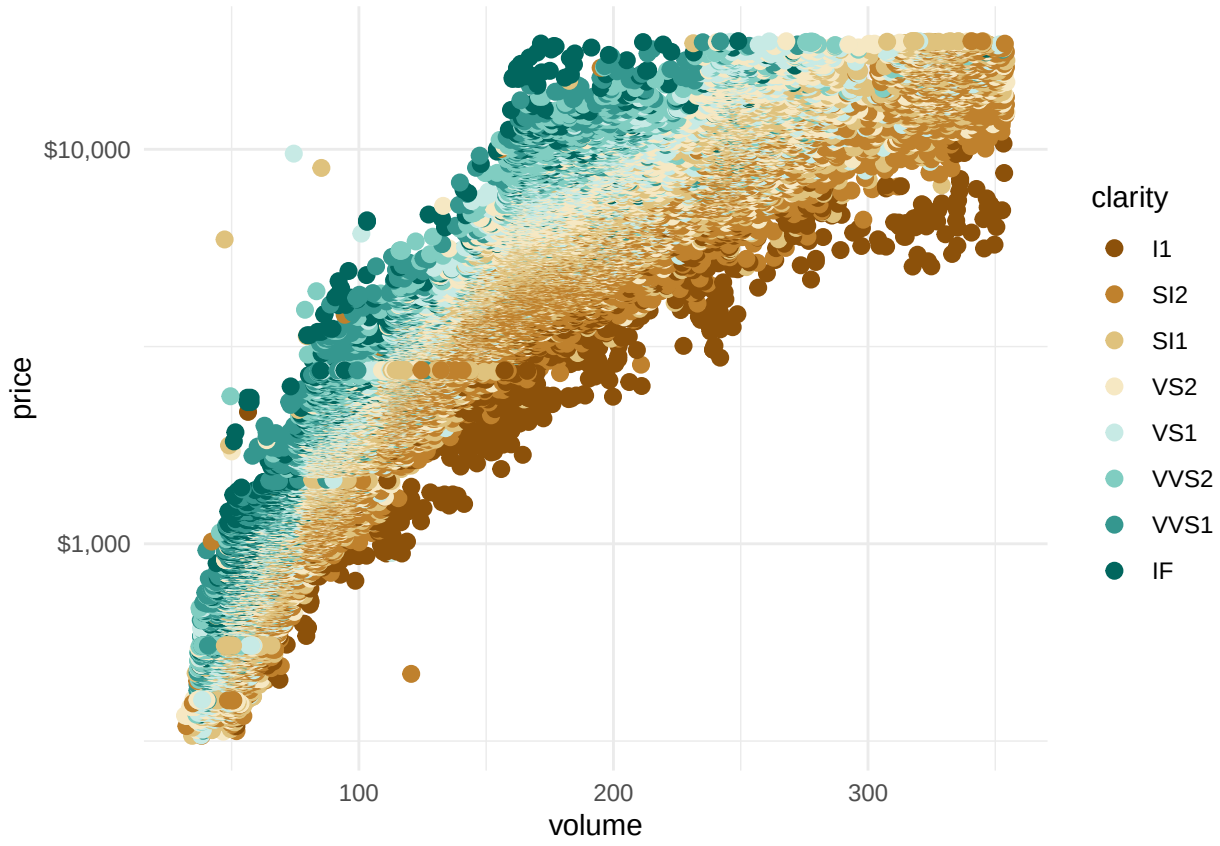
```
## Warning: Removed 5 rows containing missing values (geom_point).
```



9.2 Price vs volume by clarity

- Notes:
- Top 1% of diamond volumes omitted
- Again, this is a rough approximation of a diamond's volume

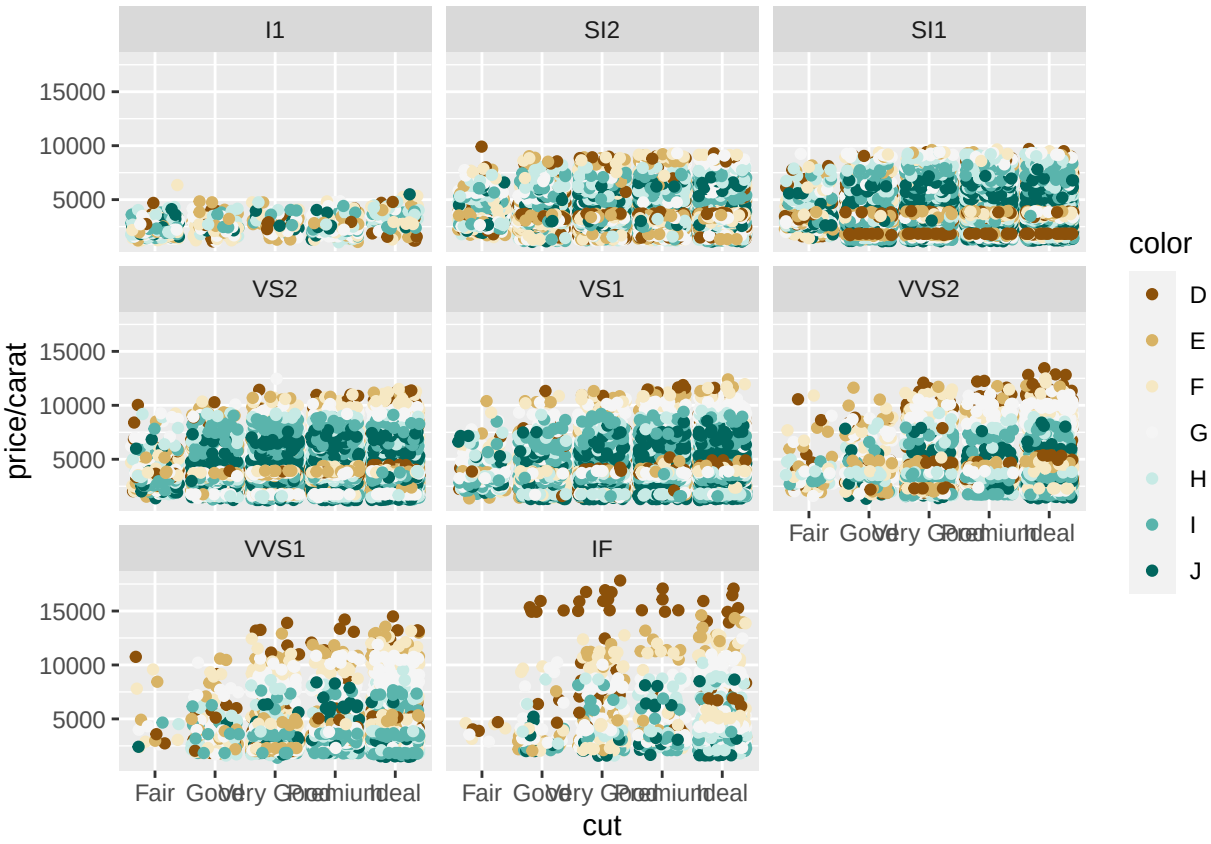
```
# Add volume column
diamonds <- diamonds %>%
  mutate(volume = x * y * z)

ggplot(subset(diamonds, volume <= quantile(volume, 0.99) & volume > 0),
       aes(x=volume, y=price, color=clarity)) +
  geom_point(size=2.5) +
  scale_y_log10(labels=dollar,
                breaks=c(0, 1000, 10000)) +
  scale_color_brewer(palette = 'BrBG') +
  theme_minimal()
```



9.3 Price/carat ratio of diamonds

- Notes:
- Variable x is assigned to cut
- Points are colored by diamond color
- Plot is faceted by clarity

```
ggplot(diamonds, aes(x=cut, y=price/carat, color=color)) +
  geom_jitter() +
  facet_wrap(~clarity) +
  scale_color_brewer(palette = 'BrBG')
```


I1
SI2
SI1
VS2
VS1
VVS2
VVS1
IF
price/carat
15000
10000
5000
Fair
Good
Very Good
Premium
Ideal
cut
color
D
E
F
G
H
I
J