DS_Nov_10-Graphics_Using_ggplot-1

Srijan Kundu

2022-11-10

```
set.seed(1)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

The dataset that will be used to demonstrate are `diamonds` dataset.

```
diamonds
```

```
## # A tibble: 53,940 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
## # ... with 53,930 more rows
```

The scales of measurements are given as `dbl` or `ord` as ordinal etc.

```
head(diamonds)
```

```
## # A tibble: 6 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
```

```
glimpse(diamonds)
```

```
## Rows: 53,940
## Columns: 10
## $ carat   <dbl> 0.23, 0.21, 0.23, 0.29, 0.31, 0.24, 0.24, 0.26, 0.22, 0.23, 0.~
## $ cut     <ord> Ideal, Premium, Good, Premium, Good, Very Good, Very Good, Ver~
## $ color   <ord> E, E, E, I, J, J, I, H, E, H, J, J, F, J, E, E, I, J, J, J, I,~
## $ clarity <ord> SI2, SI1, VS1, VS2, SI2, VVS2, VVS1, SI1, VS2, VS1, SI1, VS1, ~
## $ depth   <dbl> 61.5, 59.8, 56.9, 62.4, 63.3, 62.8, 62.3, 61.9, 65.1, 59.4, 64~
## $ table   <dbl> 55, 61, 65, 58, 58, 57, 57, 55, 61, 61, 55, 56, 61, 54, 62, 58~
## $ price   <int> 326, 326, 327, 334, 335, 336, 336, 337, 337, 338, 339, 340, 34~
## $ x       <dbl> 3.95, 3.89, 4.05, 4.20, 4.34, 3.94, 3.95, 4.07, 3.87, 4.00, 4.~
## $ y       <dbl> 3.98, 3.84, 4.07, 4.23, 4.35, 3.96, 3.98, 4.11, 3.78, 4.05, 4.~
## $ z       <dbl> 2.43, 2.31, 2.31, 2.63, 2.75, 2.48, 2.47, 2.53, 2.49, 2.39, 2.~
```

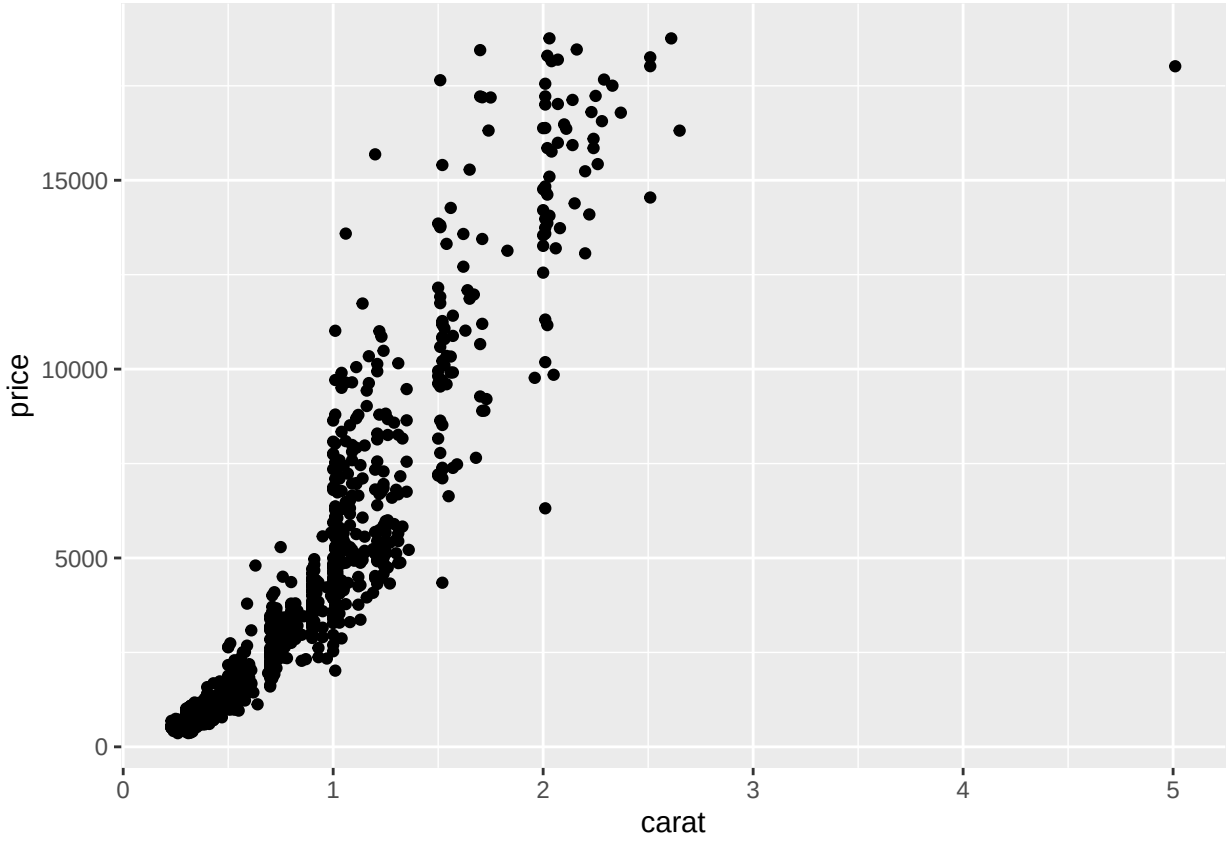
Obtain a random sample of 1000 rows from the `diamond` dateset.

```
sample = slice_sample(diamonds,  n = 1000)
sample
```

```
## # A tibble: 1,000 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.41 Very Good D     SI2      62.3    61   638  4.72  4.75  2.95
##  2  0.5  Very Good F     VS2      62.8    57  1402  5.05  5.08  3.18
##  3  1.03 Fair      I     SI2      65.2    56  3530  6.42  6.35  4.16
##  4  1.1  Ideal     I     SI1      62.1    57  5037  6.6   6.64  4.11
##  5  1.51 Very Good E     VS2      63.3    61 13757  7.24  7.17  4.56
##  6  0.3  Ideal     H     VS2      62.1    55   457  4.3   4.33  2.68
##  7  0.87 Premium   J     SI1      61.4    57  2321  6.17  6.14  3.78
##  8  1.05 Very Good H     VS1      63.3    57  5657  6.45  6.4   4.07
##  9  1    Good      F     SI1      64      57  4372  6.29  6.33  4.04
## 10  2.01 Good      I     SI1      63.8    57 13976  7.95  7.91  5.06
## # ... with 990 more rows
```

Obtain a scatterplot of price against carat from this sample. Use the `qplot` function.

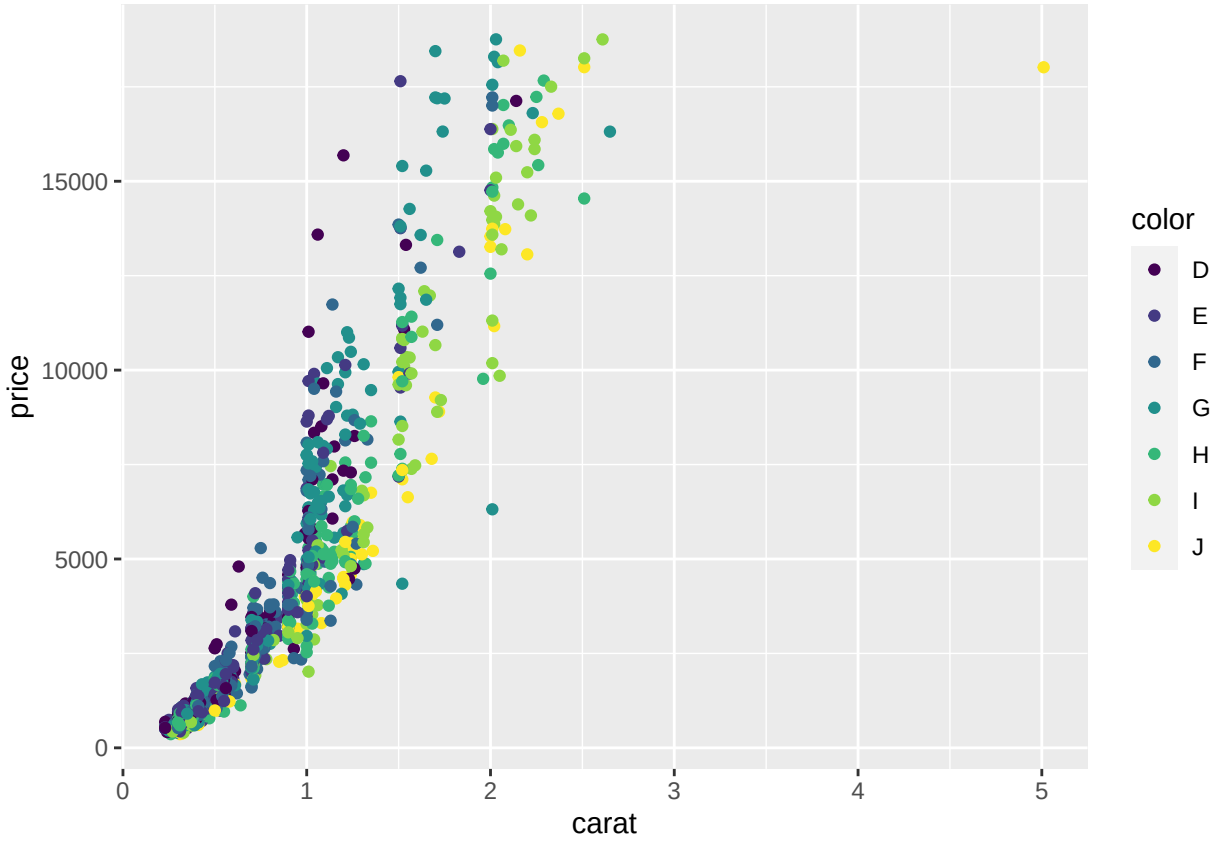
```
qplot(carat, price, data = sample)
```




```
#ggplot(sample, aes(x = carat, y = price)) + geom_point()
```

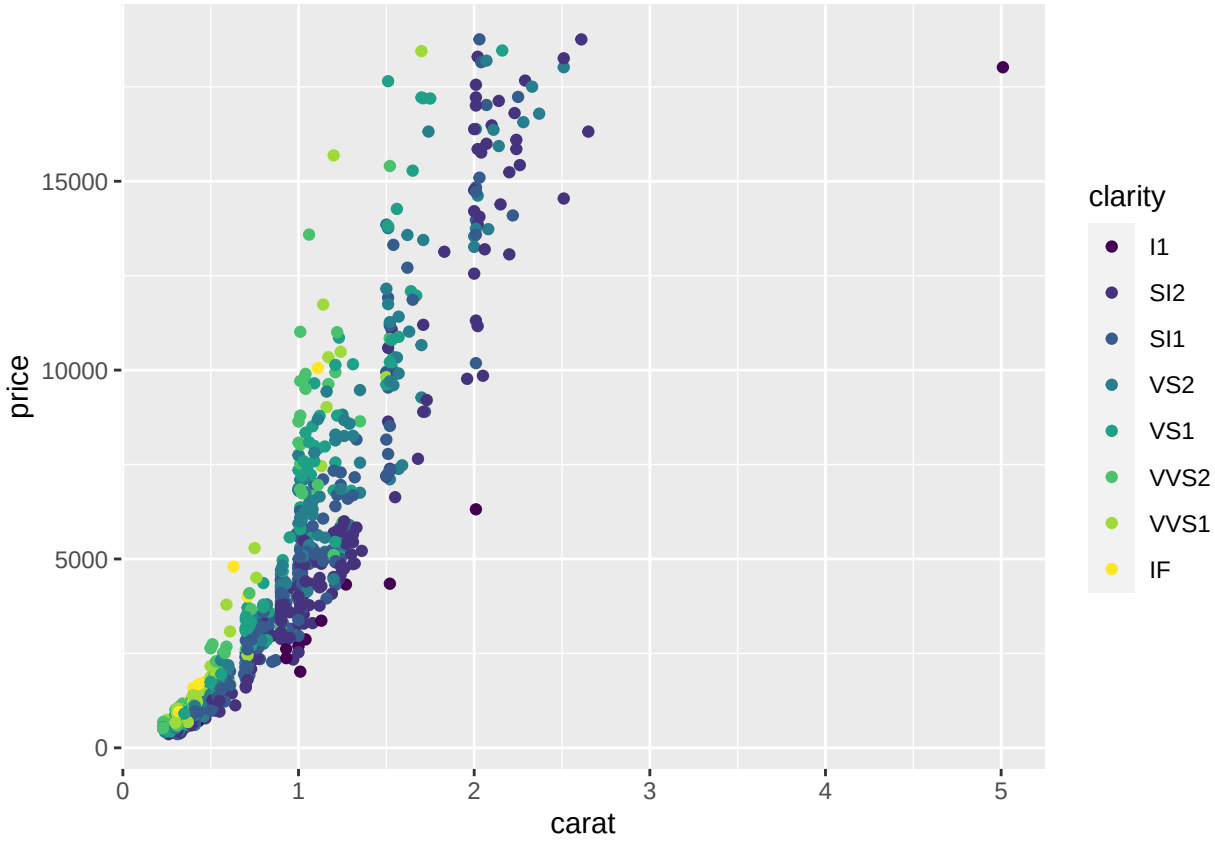
Faceting is subseting the data and using different plots for each characteistic. But, here we want to see which point corresponds to which color in the same scatterplot demonstrated above.

```
qplot(carat, price, data = sample, color = color)
```



Same thing as above using `clarity`:

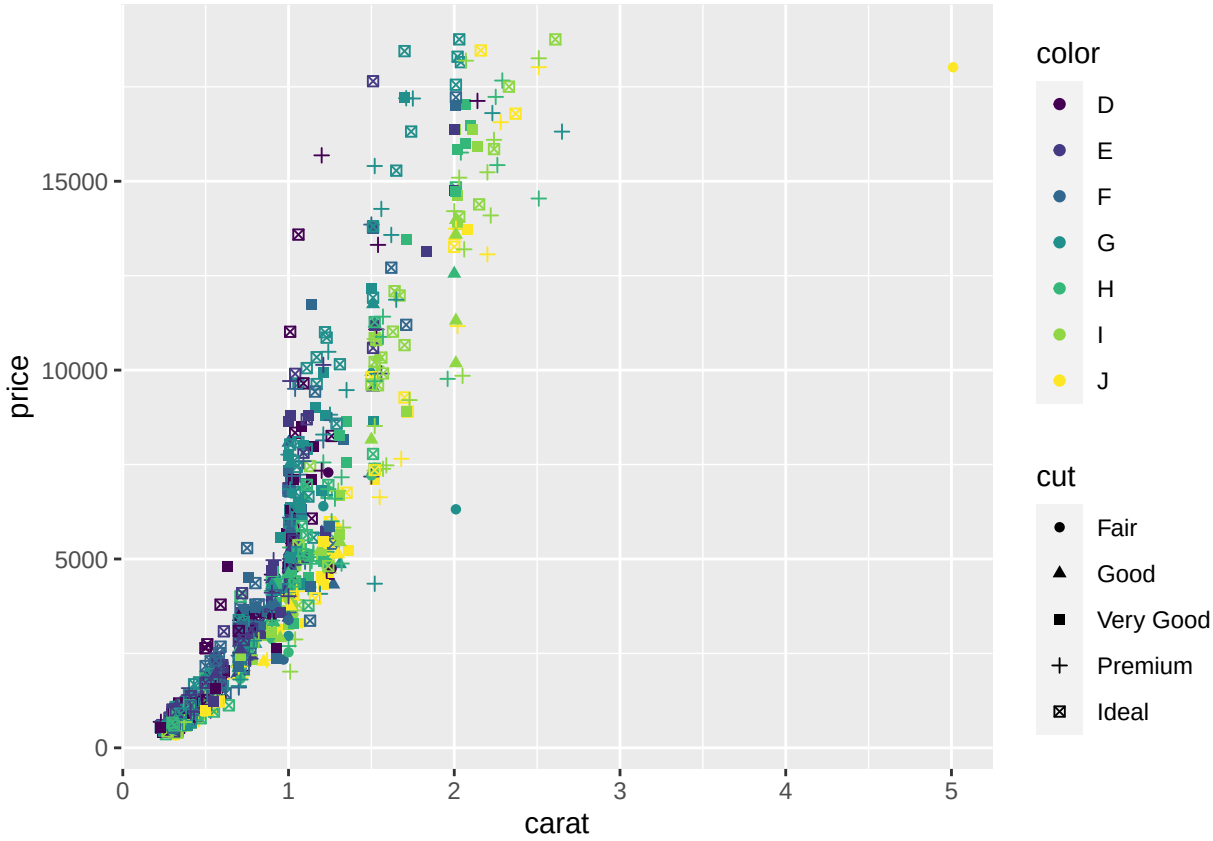
```
qplot(carat, price, data = sample, color = clarity)
```



Show different "cuts" is different shapes

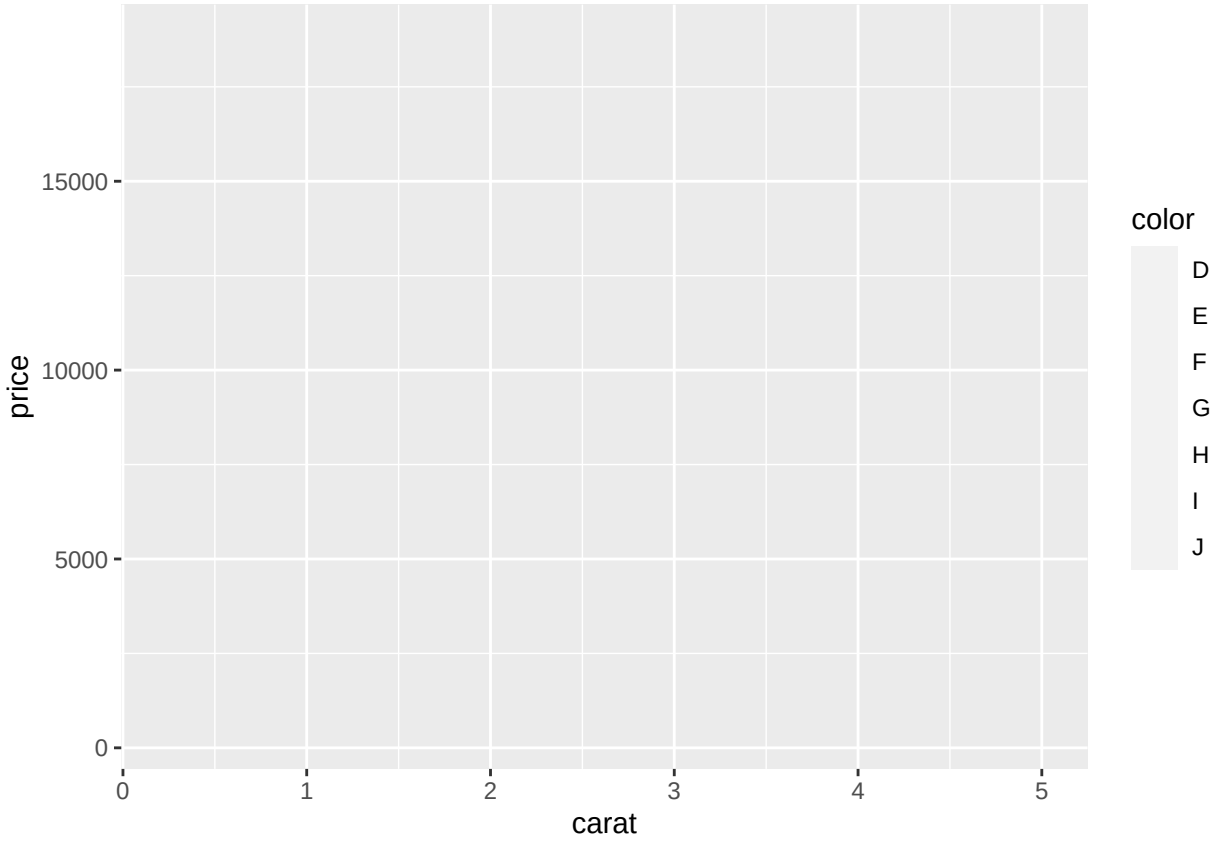
```
qplot(carat, price, data = sample, color = color, shape = cut)
```

```
## Warning: Using shapes for an ordinal variable is not advised
```

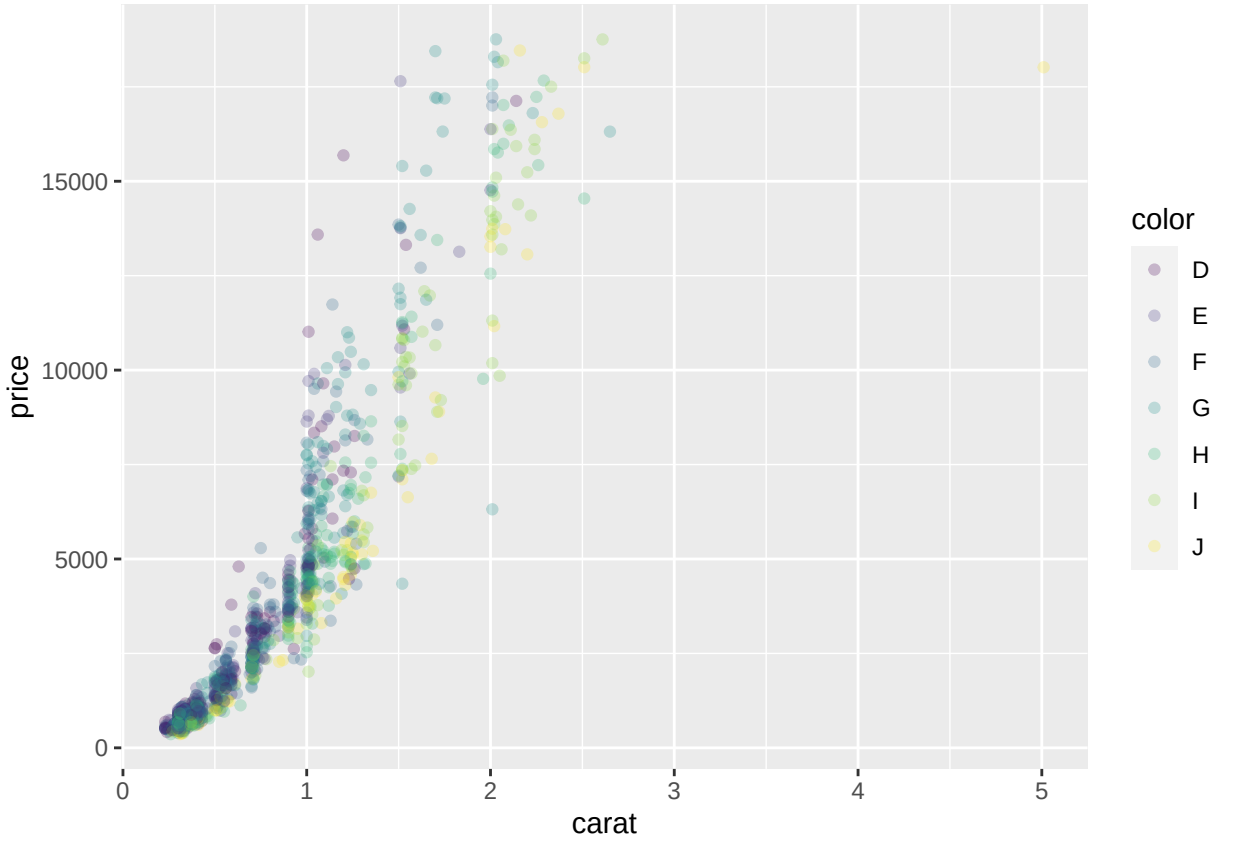


Make points in the bottom left corner brighter than those in the top right corner, or control the transperancy.

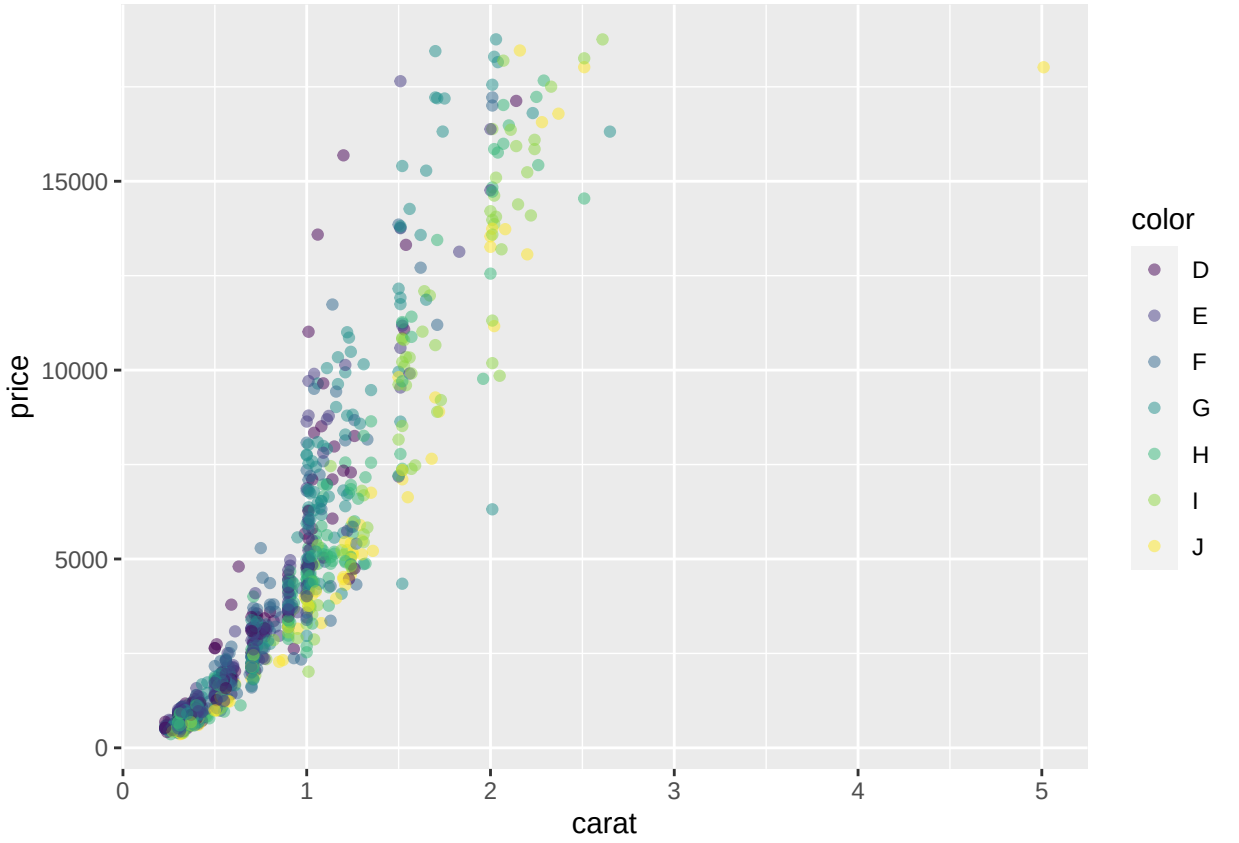
```
qplot(carat, price, data = sample, color = color, alpha = I(0))
```



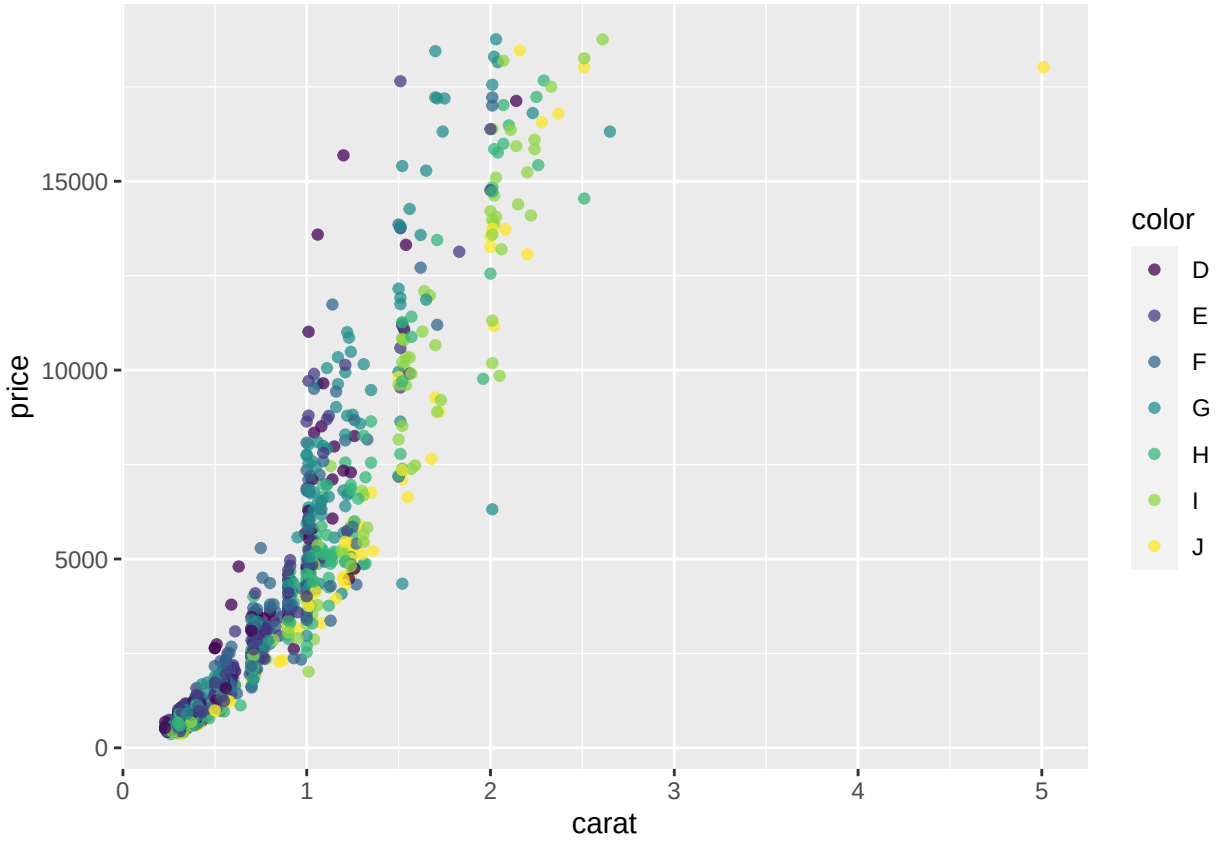

```
qplot(carat, price, data = sample, color = color, alpha = I(0.25))
```



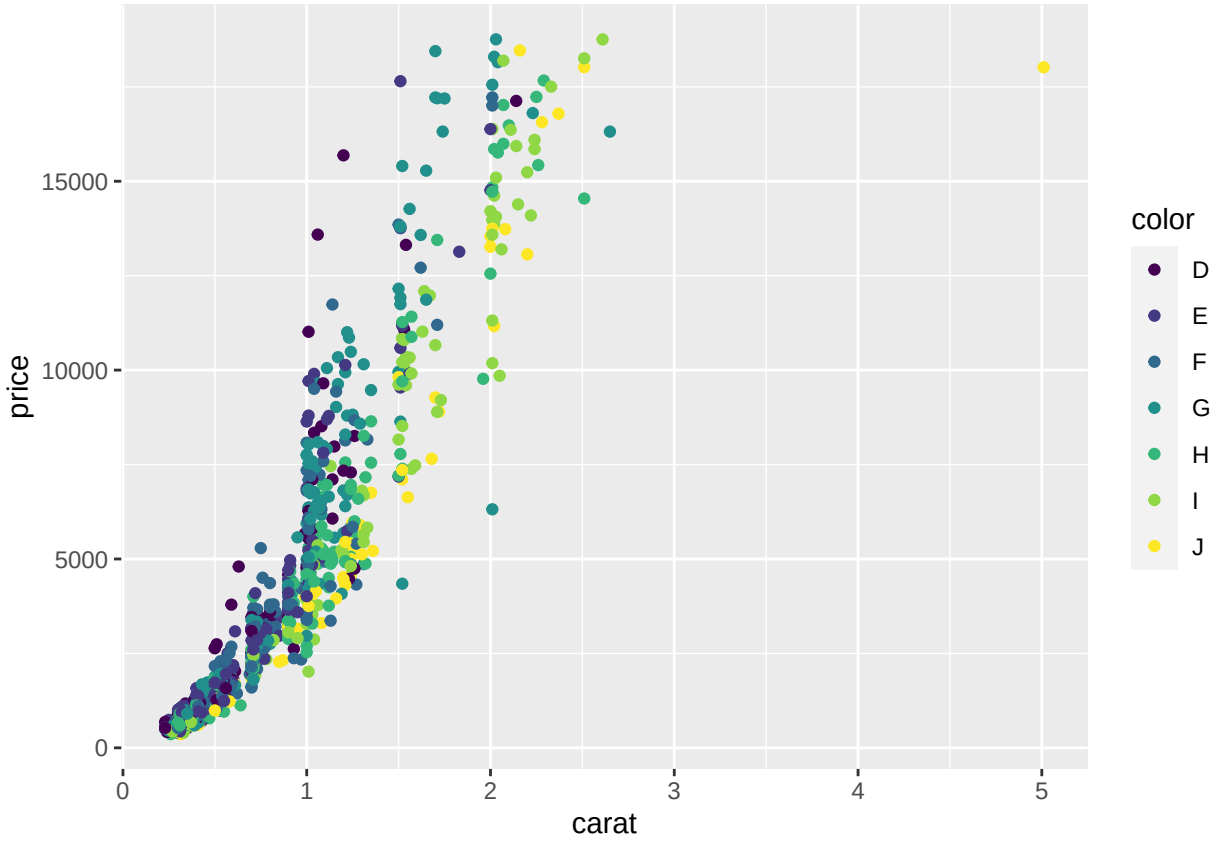

```
qplot(carat, price, data = sample, color = color, alpha = I(0.5))
```




```
qplot(carat, price, data = sample, color = color, alpha = I(0.75))
```

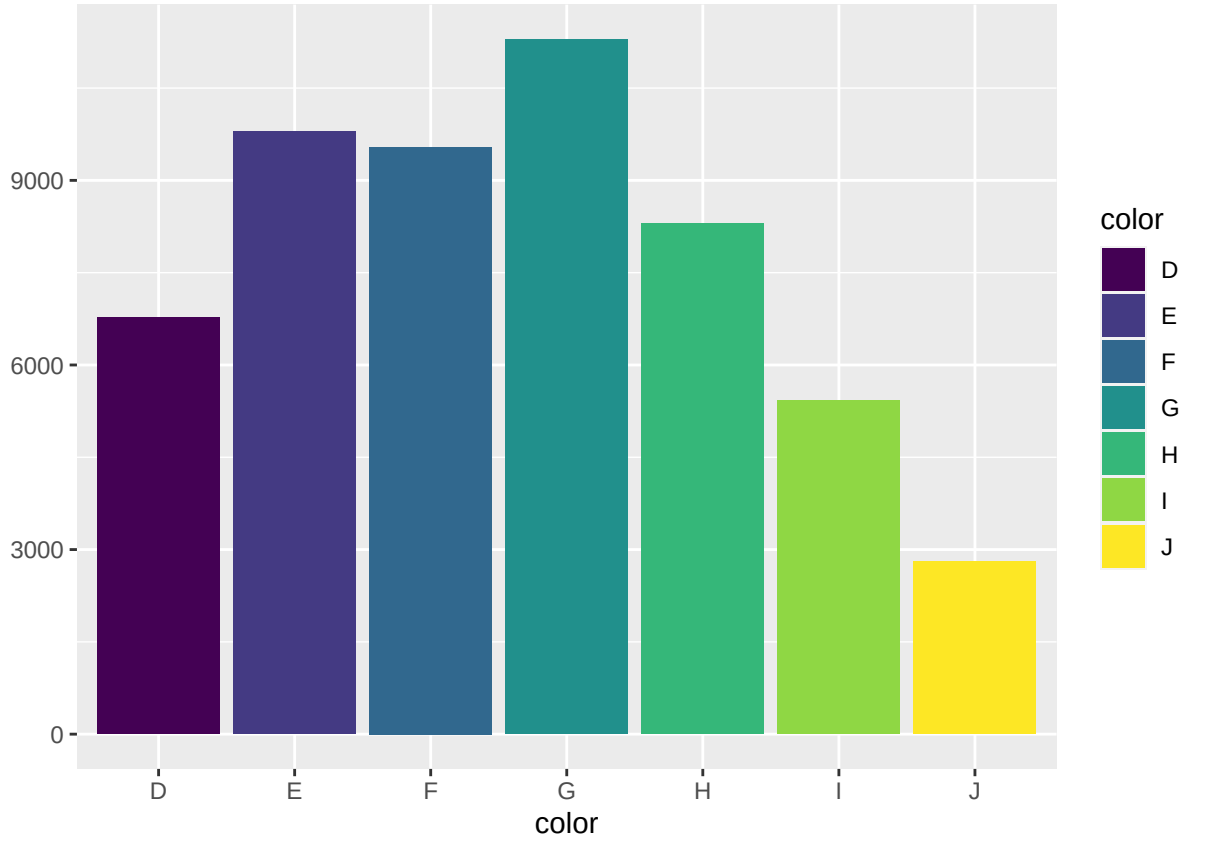



```
qplot(carat, price, data = sample, color = color, alpha = I(1))
```



Try using bar diagram for frequency distribution for different colors.

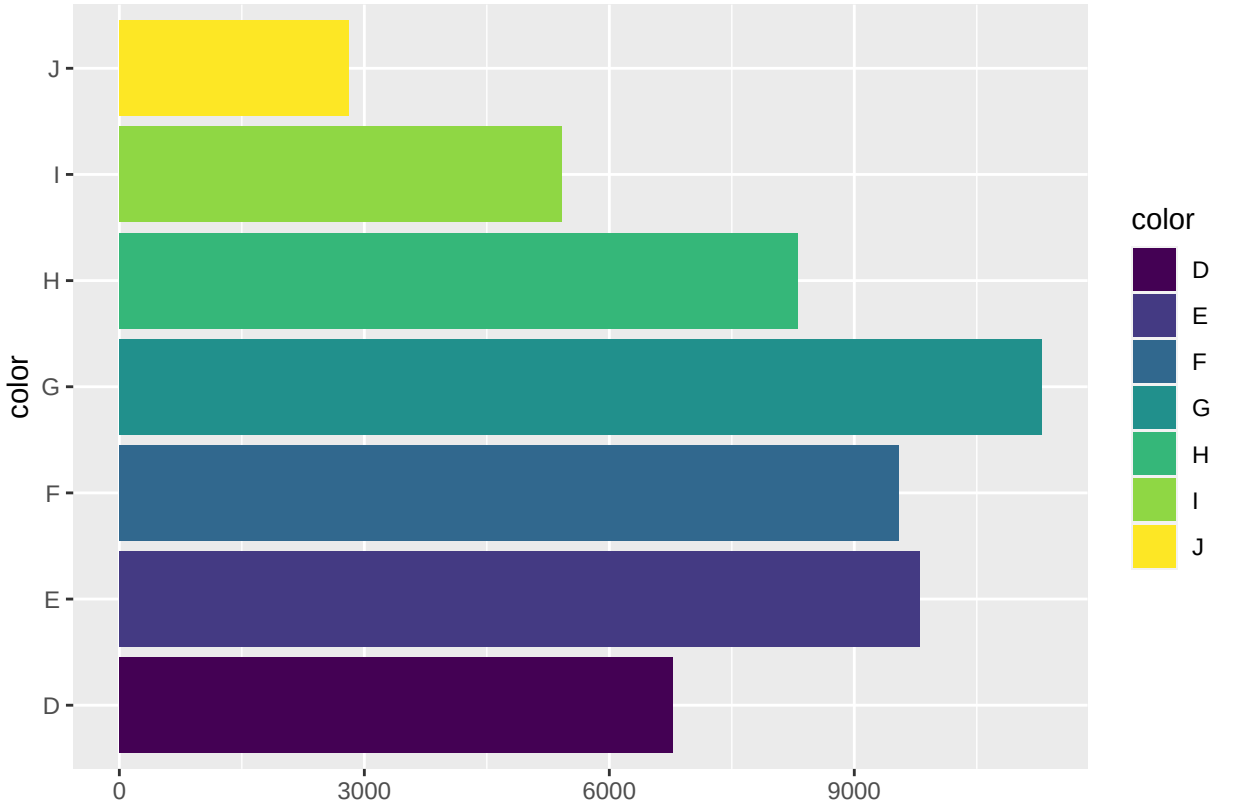
```
qplot(color, data = diamonds, fill = color)
```




```
#ggplot(diamonds, aes(x = color, fill = color)) + geom_bar()
```

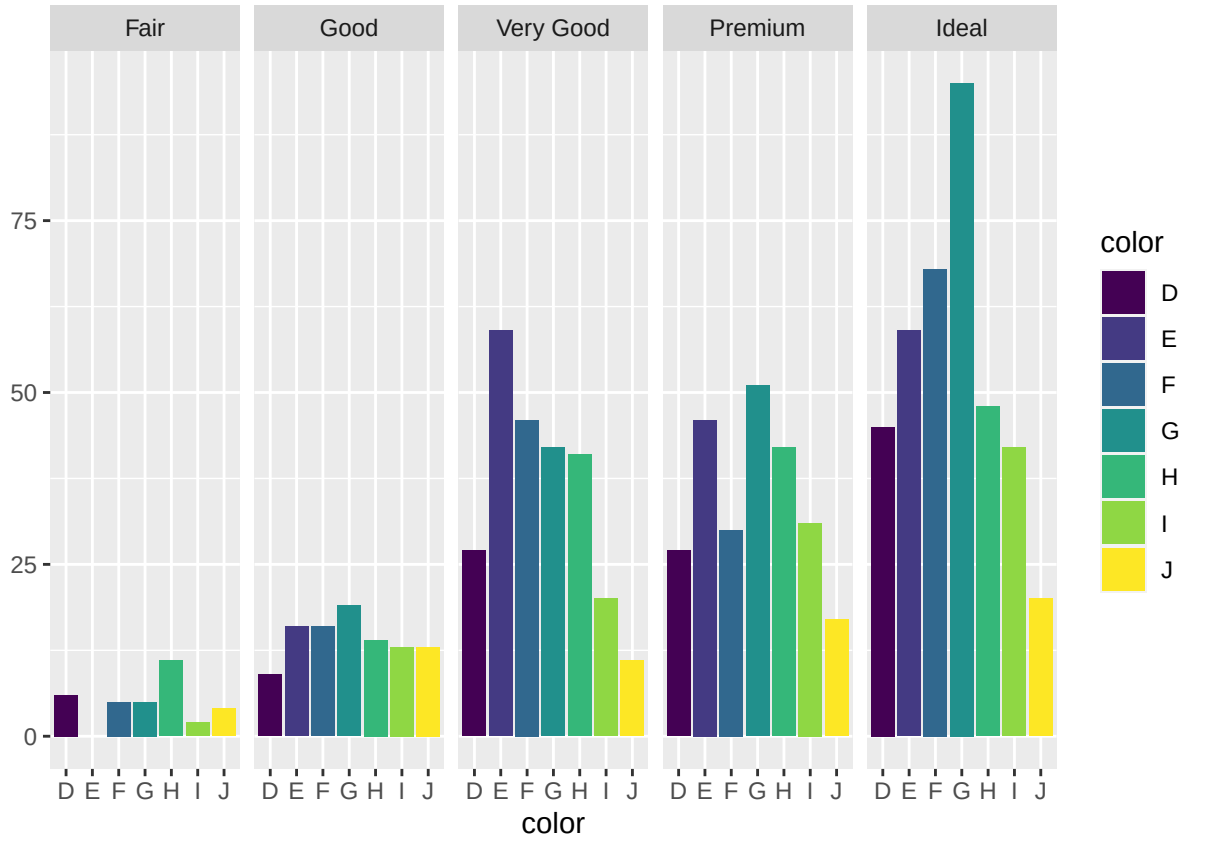
Change the vertical bars to horizontal bars.

```
plot = qplot(color, data = diamonds, fill = color)
plot + coord_flip()     #this is called layering
```



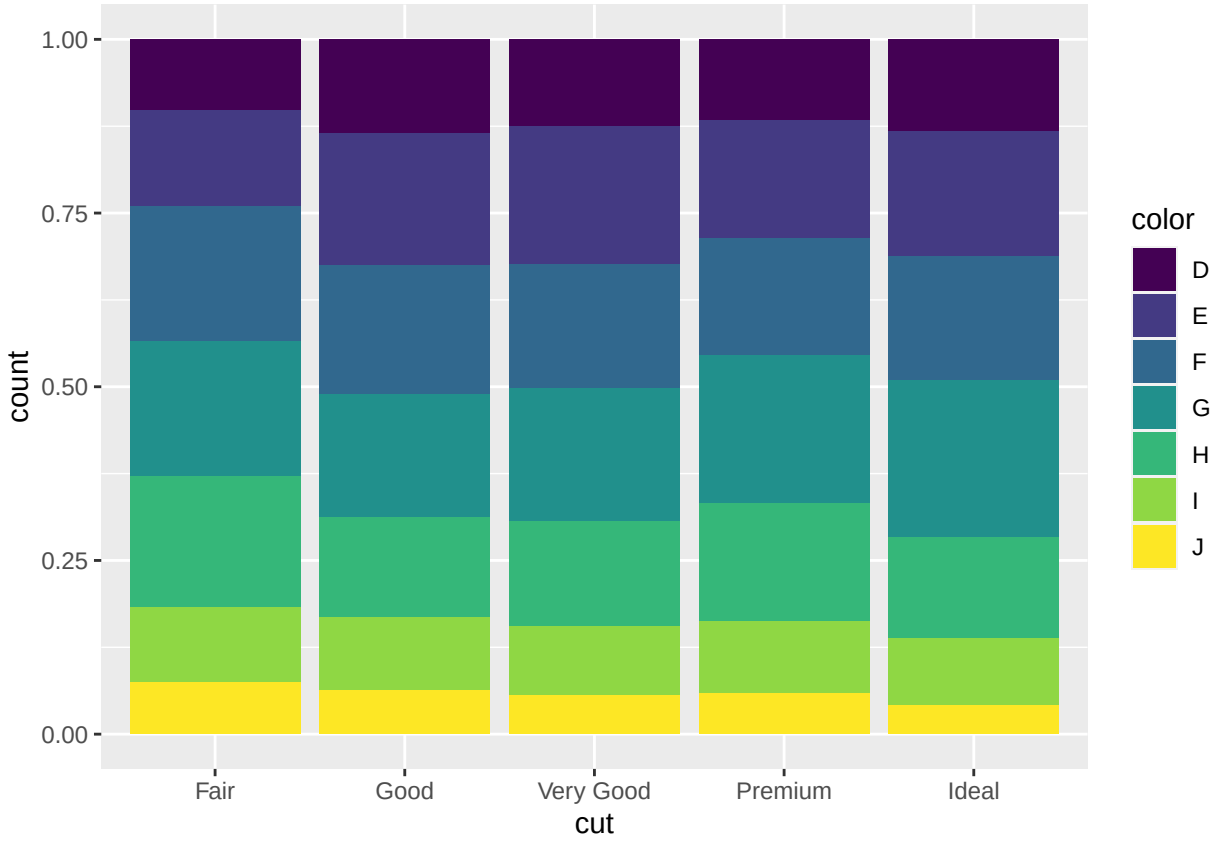
For each type of cut, obtain freq dist of number of different colors.

```
qplot(color, data = sample, fill = color, geom = "bar") + facet_grid(~cut)
```



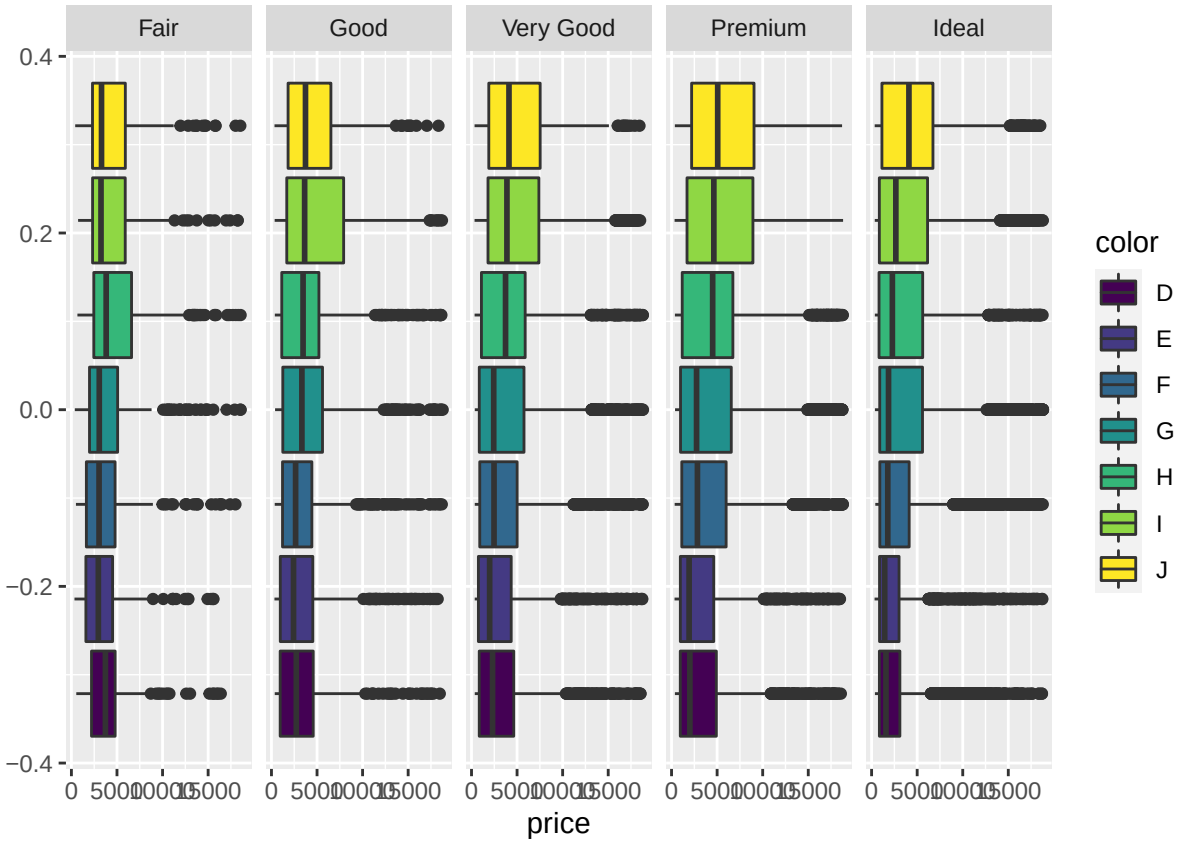
Use subdevided bars

```
#qplot(color, data = diamonds, fill = color, geom = "bar", )
ggplot(diamonds, aes(fill = color, x = cut)) + geom_bar(position="fill")
```

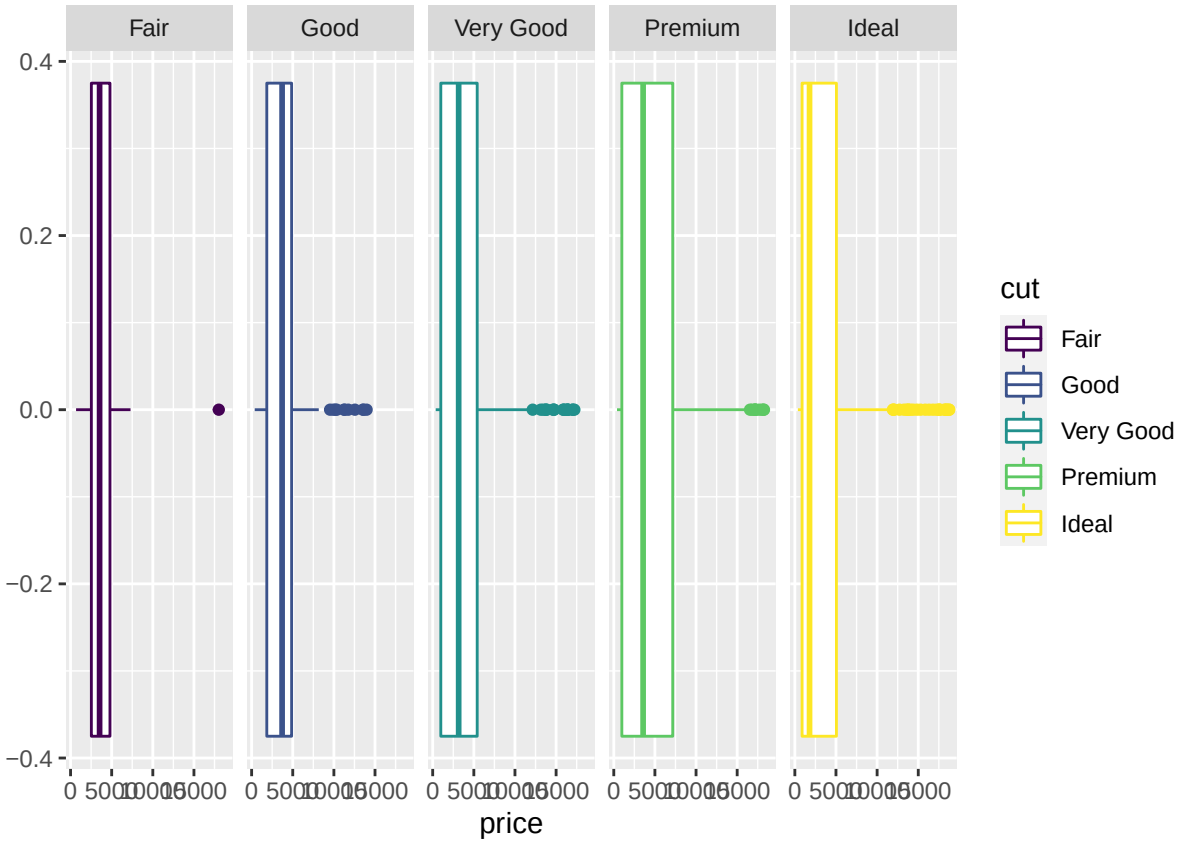


Obtain a boxplot of the price of diamonds for each cut of diamonds

```
#for the entire dataset
qplot(price, data = diamonds, fill = color, geom = "boxplot") + facet_grid(~cut)
```




```
qplot(price, data = sample, color = cut, geom = "boxplot") + facet_grid(~cut)
```


Fair
Good
Very Good
Premium
Ideal
0.4
0.2
0.0
−0.2
−0.4
0 5000 10000 15000
price
cut
Fair
Good
Very Good
Premium
Ideal