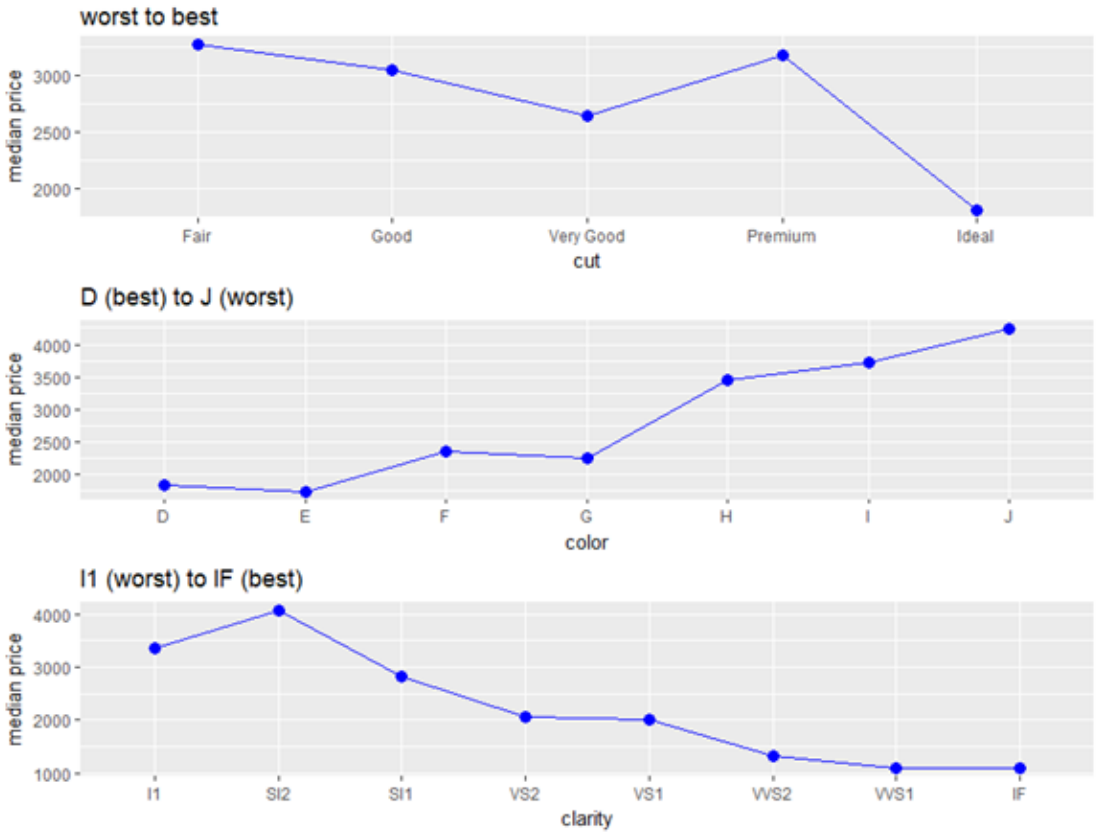
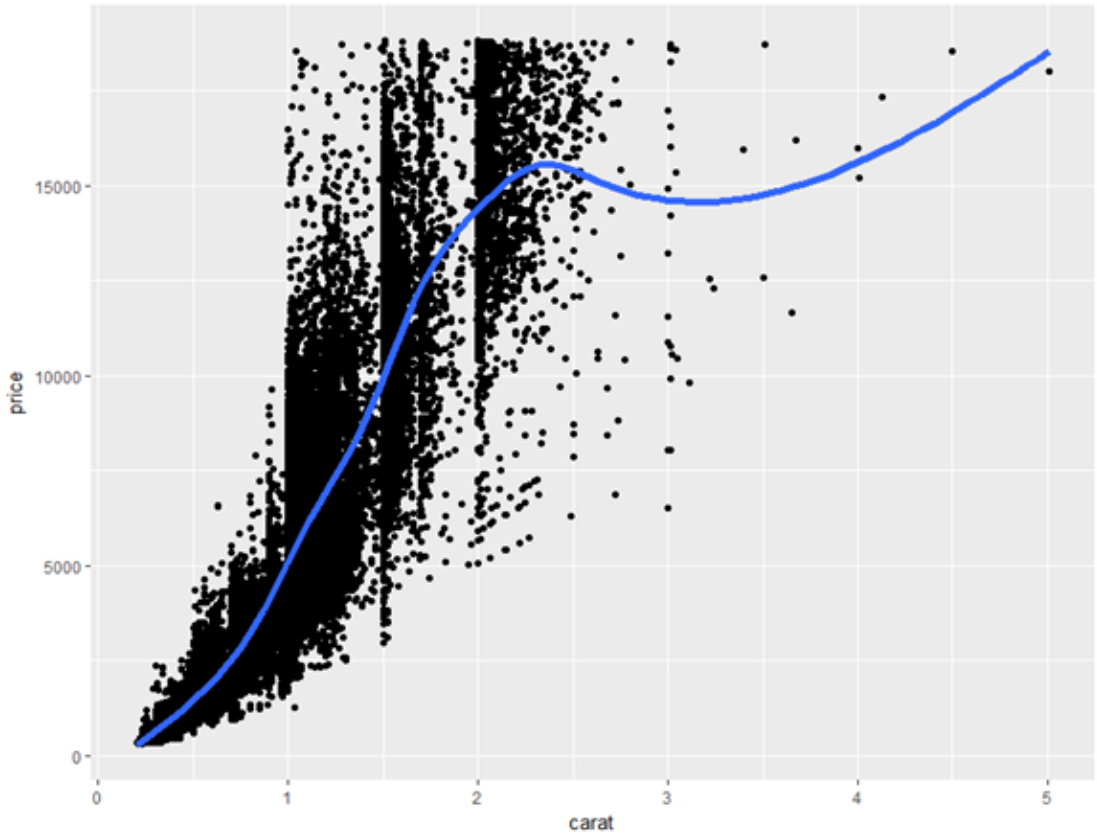
Low quality diamonds more expensive?

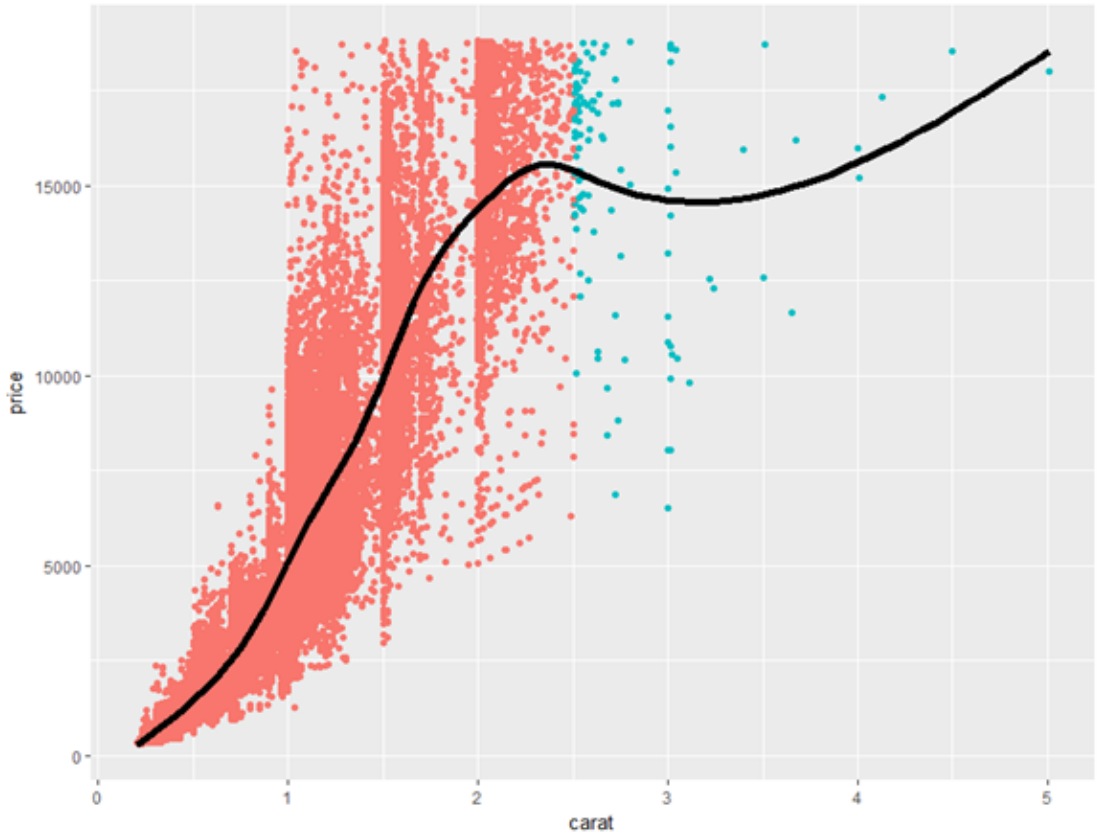
The quality of a diamond is determined by its cut, color and clarity. Each of these are ordered categorical variables. The plots below show that the median price of diamonds for each of the quality variables cut, color and clarity tends to decrease. This seems counter-intuitive!



Another variable influencing the price is carat (weight, 1 carat = 200 milligrams). The plot below shows that price rapidly increases with carat over the range 0 to 2.5.



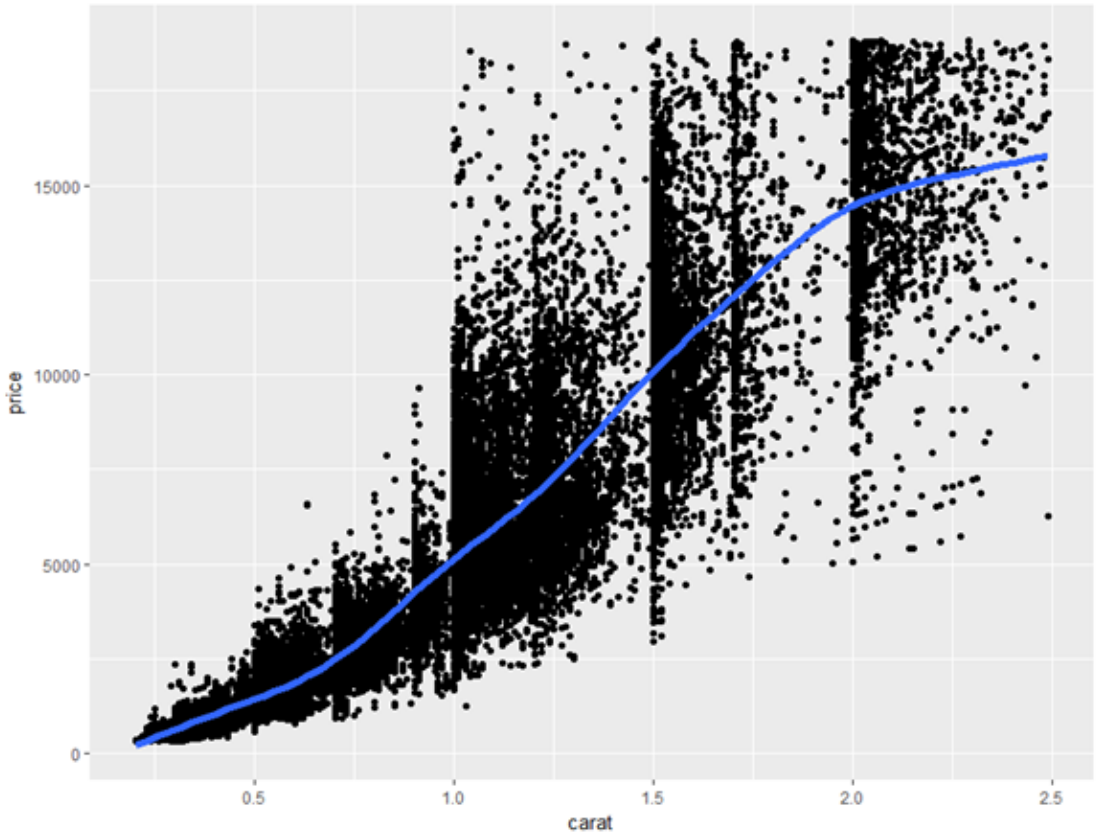
This plot shows even more clearly that the bulk of diamonds are less than 2.5 carats.



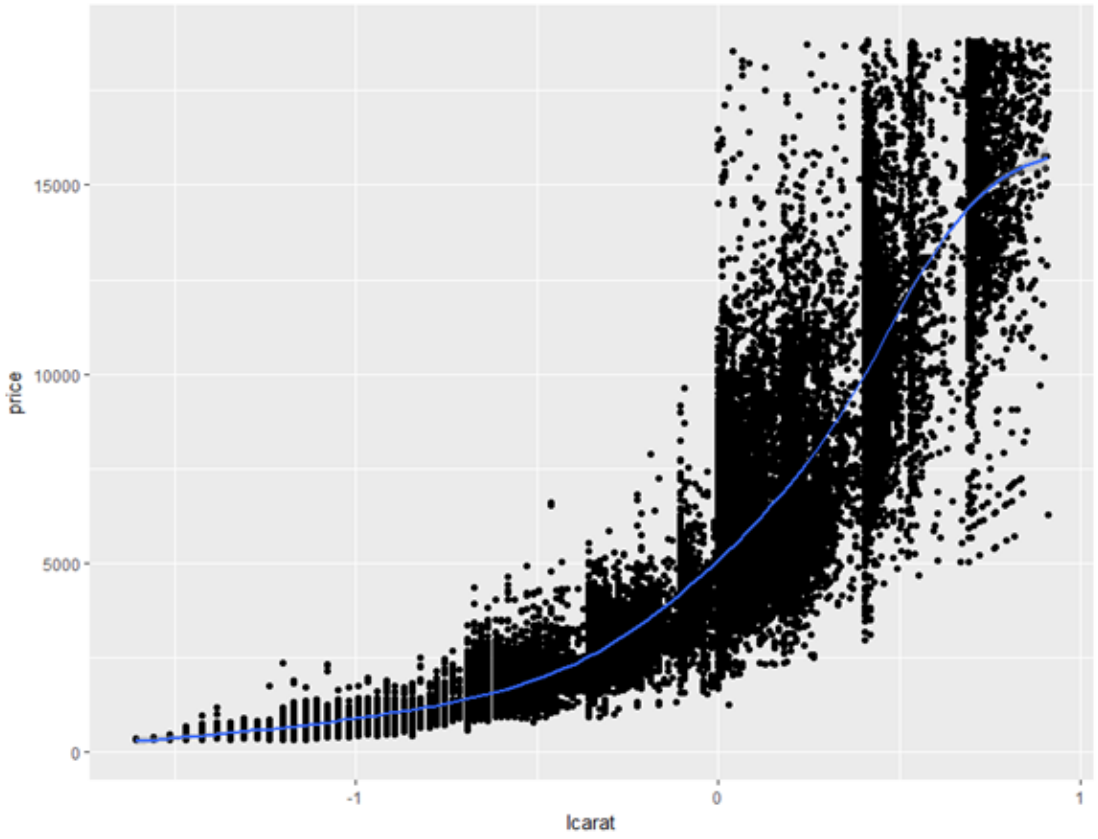
Only 126 out of nearly 54000 diamonds exceed 2.5 carats.

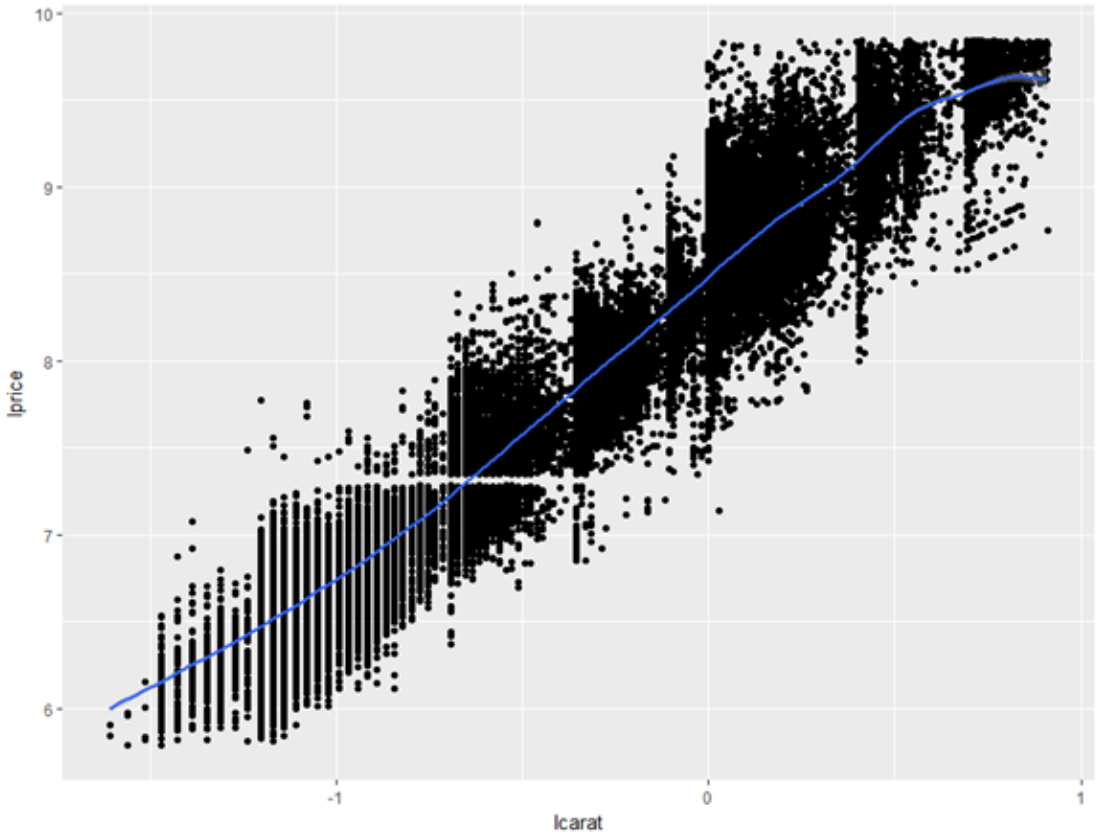
```
> diamonds %>%
+  mutate(large = carat>2.5) %>%
+  group_by(large) %>%
+  summarize(count = n())
# A tibble: 2 x 2
  large count
  <lgl> <int>
1 FALSE 53814
2  TRUE   126
```

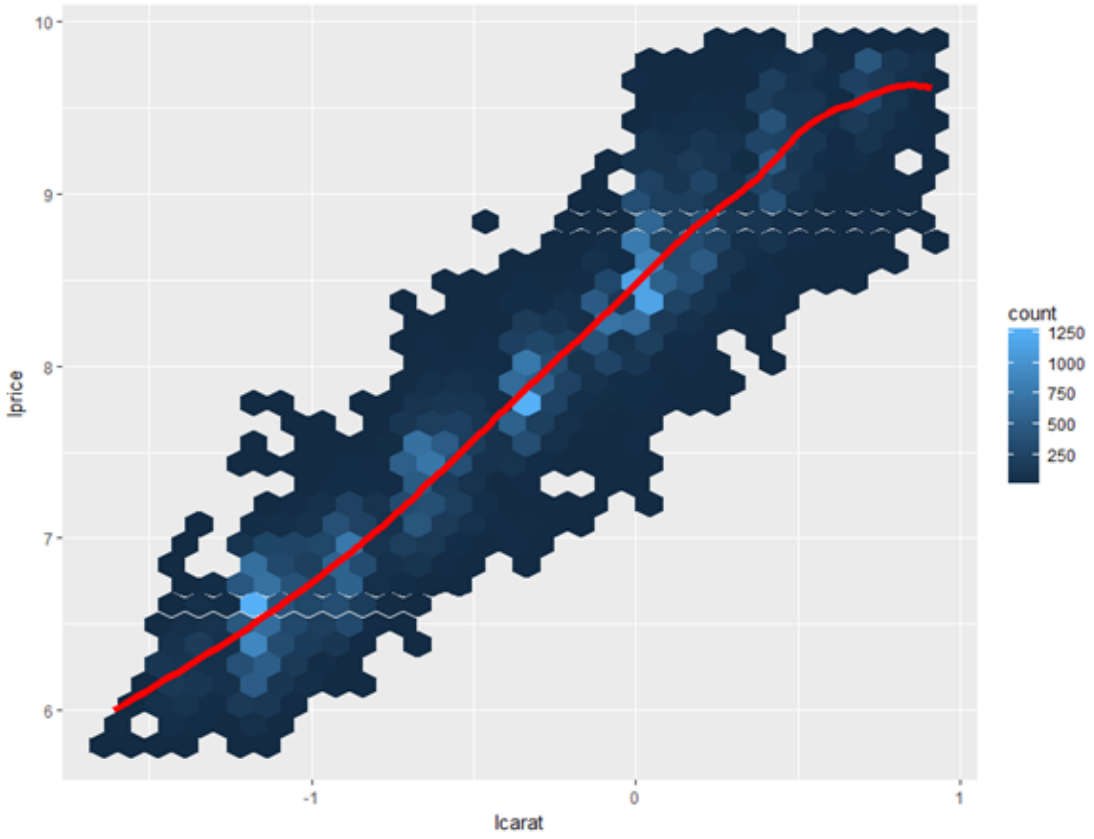
The scatterplot of the truncated data with carat values less than 2.5 suggests non-linear shape. Our arrow rule suggests trying a log or square-root transformation on carat.

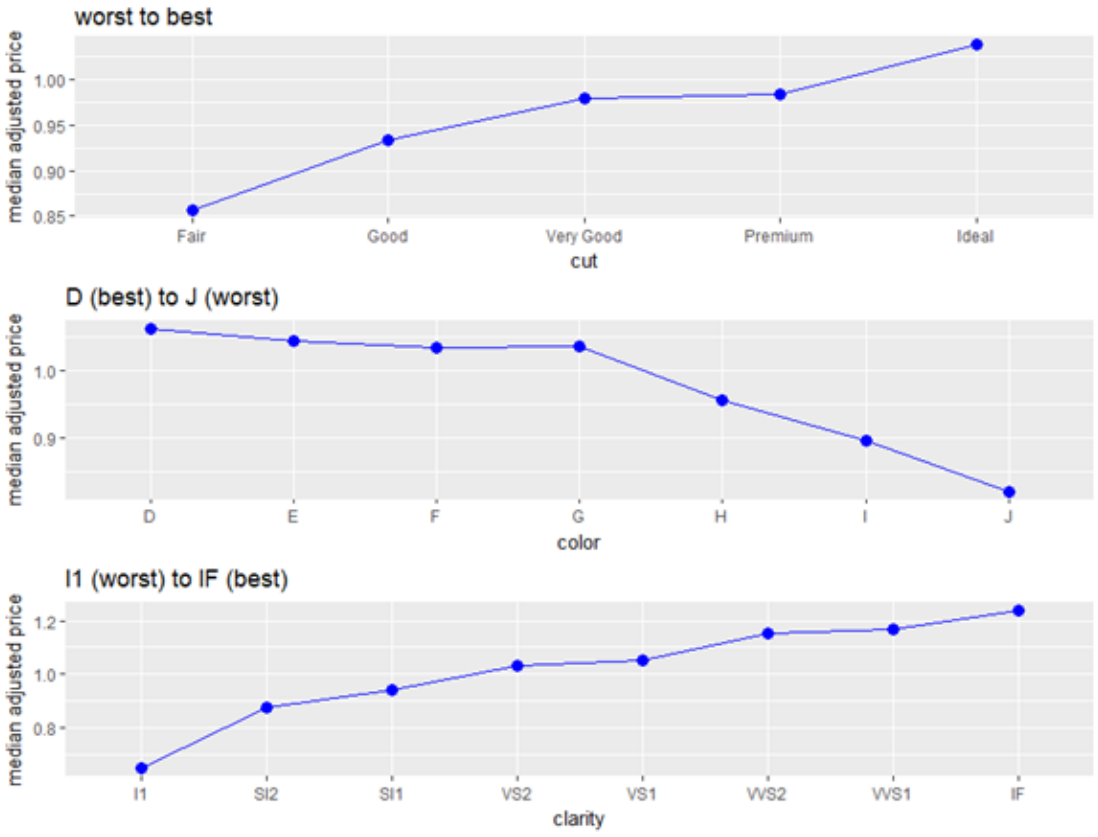


After a log transformation on carat, the arrow rule suggests could try a log transformation on price.


price
15000
10000
5000
0
-1
0
1
lcarat


10
9
8
7
6
lprice
-1
0
1
lcarat


10
9
8
7
6
lprice
-1
0
1
lcarat
count
1250
1000
750
500
250




```
#Source: LowQualityDiamondsQ.R
#
library(tidyverse)
library(grid)
library(gridExtra)
#
plot1 <- diamonds %>%
 group_by(cut) %>%
 summarize(median=median(price)) %>%
 ggplot(mapping=aes(cut, median)) +
   geom_point(size=3, col="blue") +
   stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
   ggtitle("worst to best") +
   ylab("median price")
#
plot2 <- diamonds %>%
 group_by(color) %>%
 summarize(median=median(price)) %>%
 ggplot(mapping=aes(color, median)) +
 geom_point(size=3, col="blue") +
 stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
 ggtitle("D (best) to J (worst)") +
 ylab("median price")
#
plot3 <- diamonds %>%
 group_by(clarity) %>%
 summarize(median=median(price)) %>%
 ggplot(mapping=aes(clarity, median)) +
 geom_point(size=3, col="blue") +
```

```
 stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
 ggtitle("I1 (worst) to IF (best)") +
 ylab("median price")
#
grid.arrange(plot1, plot2, plot3)
#
#relationship between price and carat
ggplot(data = diamonds, mapping = aes(x = carat, y = price)) +
 geom_point() +
 geom_smooth(se=FALSE, lwd=2)

diamonds %>%
 mutate(large = carat>2.5) %>%
 group_by(large) %>%
 ggplot(mapping = aes(x = carat, y = price, color=large)) +
 geom_point(show.legend=FALSE) +
 geom_smooth(se=FALSE, lwd=2, color="black")

diamonds %>%
 mutate(large = carat>2.5) %>%
 group_by(large) %>%
 summarize(count = n())

#fit model - use adjusted price
diamonds2 <- diamonds %>%
 filter(carat < 2.5)
ggplot(diamonds2, mapping = aes(x = carat, y = price)) +
 geom_point() +
 geom_smooth(se=FALSE, lwd=2)
#
#lets try log carat first as arrow-rule suggests
#first new dataframe
diamonds2 <- diamonds2 %>%
 mutate(lcarat = log(carat), lprice=log(price))
#
ggplot(diamonds2, aes(x = lcarat, y = price)) +
 geom_point() +
 geom_smooth()
#now arrow rule suggests log price
diamonds2 %>%
 ggplot(aes(x = lcarat, y = lprice)) +
 geom_point() +
 geom_smooth()
#hexogonal bins provides another visualization
ggplot(diamonds2, mapping = aes(x = lcarat, y = lprice)) +
 geom_hex(bin=50) +
 geom_smooth(se=FALSE, color="red", lwd=2)
#
#adjust the price by removing the effect of carat
diamonds3 <- diamonds2 %>%
 mutate(lpriceAdj=resid(lm(lprice ~ lcarat, data = diamonds2))) %>%
 mutate(priceAdj=2^lpriceAdj)
#
ggplot(diamonds3, mapping = aes(x = lcarat, y = lpriceAdj)) +
 geom_hex(bin=50) +
 geom_smooth(se=FALSE, color="red", lwd=2)
#
plot1 <- diamonds3 %>%
 group_by(cut) %>%
 summarize(median=median(priceAdj)) %>%
```

```
 ggplot(mapping=aes(cut, median)) +
 geom_point(size=3, col="blue") +
 stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
 ggtitle("worst to best") +
 ylab("median adjusted price")
#
plot2 <- diamonds3 %>%
 group_by(color) %>%
 summarize(median=median(priceAdj)) %>%
 ggplot(mapping=aes(color, median)) +
 geom_point(size=3, col="blue") +
 stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
 ggtitle("D (best) to J (worst)") +
 ylab("median adjusted price")
#
plot3 <- diamonds3 %>%
 group_by(clarity) %>%
 summarize(median=median(priceAdj)) %>%
 ggplot(mapping=aes(clarity, median)) +
 geom_point(size=3, col="blue") +
 stat_summary(fun.y=mean, colour="blue", geom="line", aes(group = 1)) +
 ggtitle("I1 (worst) to IF (best)") +
 ylab("median adjusted price")
#
grid.arrange(plot1, plot2, plot3




###################################
###################################
#Tip of the week.
#find all functions in ggplot2 that contain `stat`
#
library(ggplot2)
library(stringr)
ind <- !is.na(str_match(obj <- objects("package:ggplot2"), "stat"))
obj[ind]
```