Exploratory Data Analysis

Lesson 6: Diamonds & Price Predictions

Quick links

Welcome to Lesson 6 of EDA!
Scatterplot Review
    Programming Quiz
    Answer
Price and Carat Relationship
    Answer
Frances Gerety
    Quiz
    Answer
The Rise of Diamonds
ggpairs Function
    Quiz
    Answer
The Demand of Diamonds
    Programming Quiz
    Answer
Connecting Demand and Price Distribution
    Answer
Scatterplot Transformation
Overplotting Revisited
    Programming Quiz
    Answer
Plot Colors for Qualitative Factors
Price vs. Carat and Clarity
    Programming Quiz
    Answer
Clarity and Price
    Answer
Price vs. Carat and Cut
    Programming Quiz
    Answer
Cut and Price
    Answer
Price vs. Carat and Color

- Programming Quiz
- Answer

Color and Price
- Answer

Linear Models in R
- Answer

Building the Linear Model

Model Problems
- Quiz
- Answer

A Bigger, Better Data Set
- Programming Quiz
- Answer

Predictions
- Quiz
- Answer

Final Thoughts

Congratulations and Next Steps

Welcome to Lesson 6 of EDA!

Chris: Congratulations on making it to lesson six. This is the final lesson for the course, and we brought in Facebook data scientist Solomon Messing to share with you some of his own work in EDA.

Solomon: In this lesson, I'll walk you through an analysis of diamonds. You'll learn about the rich history behind the diamond market, and use the EDA techniques that you've learned to develop a quantitative understanding of it. The ultimate goal is to build a predictive model of diamonds that's going to help you figure out whether a given diamond is a good deal, or a rip-off.

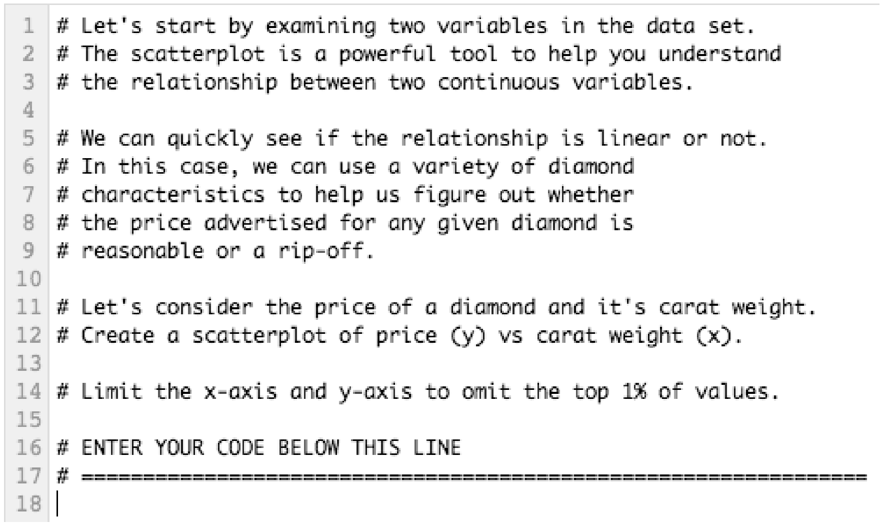
Chris: If you're in the market for a diamond, exploring this data set can help you understand what goes into the price of a diamond, and even if you're not looking to buy, the socioeconomic and political history of the diamond industry is fascinating.

Solomon: Diamonds gave rise to the mining industry in South Africa, which is now the most advanced economy in the region. Diamonds also drove the British and the Dutch to colonize southern Africa in the first place, and have driven conflicts ranging from the Boer wars to modern day civil strife across the region.

Chris: Now, you should have some experience working with the diamonds data based on the problem sets in this course. I'll let Solomon guide you through his exploration.

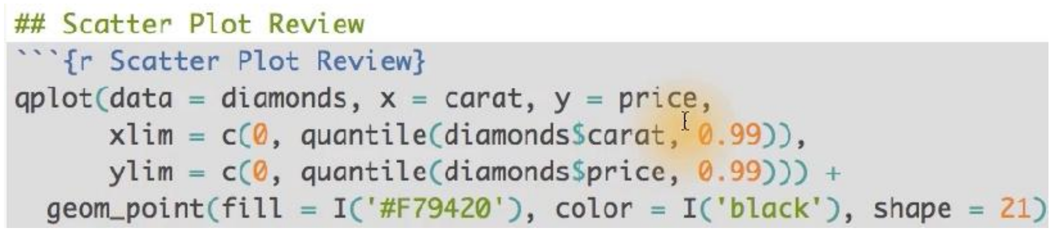
Scatterplot Review

Solomon: Let's run our usual code to load in the data side. For review, let's have you create a scatter plot in the next exercise. [(Lesson 6 RMD File)]

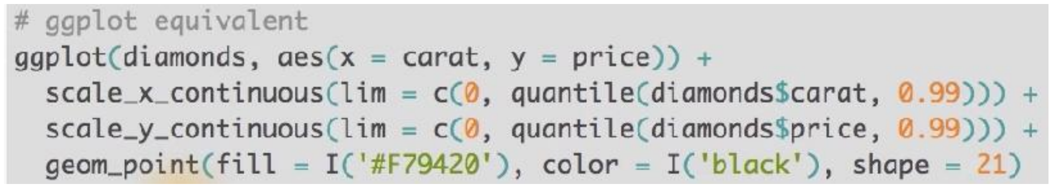


```
# Let's start by examining two variables in the data set.
# The scatterplot is a powerful tool to help you understand
# the relationship between two continuous variables.

# We can quickly see if the relationship is linear or not.
# In this case, we can use a variety of diamond
# characteristics to help us figure out whether
# the price advertised for any given diamond is
# reasonable or a rip-off.

# Let's consider the price of a diamond and it's carat weight.
# Create a scatterplot of price (y) vs carat weight (x).

# Limit the x-axis and y-axis to omit the top 1% of values.

# ENTER YOUR CODE BELOW THIS LINE
# ================================================================
```

Programming Quiz



~~~
## Scatter Plot Review
```{r Scatter Plot Review}
qplot(data = diamonds, x = carat, y = price,
      xlim = c(0, quantile(diamonds$carat, 0.99)),
      ylim = c(0, quantile(diamonds$price, 0.99))) +
  geom_point(fill = I('#F79420'), color = I('black'), shape = 21)
~~~



```
# ggplot equivalent
ggplot(diamonds, aes(x = carat, y = price)) +
  scale_x_continuous(lim = c(0, quantile(diamonds$carat, 0.99))) +
  scale_y_continuous(lim = c(0, quantile(diamonds$price, 0.99))) +
  geom_point(fill = I('#F79420'), color = I('black'), shape = 21)
```

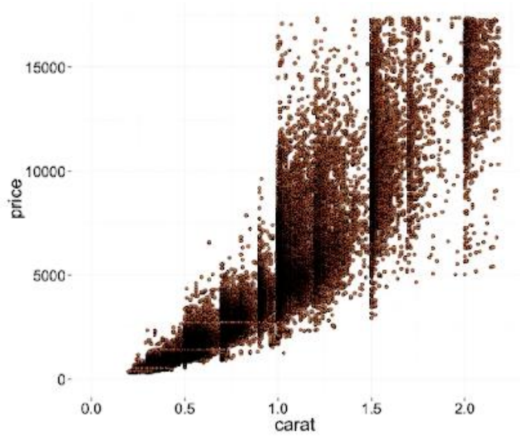
Answer

We've seen this kind of code before. All we're doing is we're using **qplot** to plot the price of diamond against its carat weight. And we're going to trim the top percentile off of both carat and price. If you use the full ggplot syntax, your code will look something like this.

Price and Carat Relationship

Alright, so let me ask you this question. What do you notice about the relationship here between price and carat? Go ahead and enter your response.

Answer



When we look at the plot, a few things pop out right away. We can see a nonlinear relationship. Maybe it's exponential or maybe it's something else. We can see that the dispersion or variance of the relationship also increases as carat size increases. With just a quick look at the data we've learned two important things about the functional relationship between price and carat size. We can add a linear trim line to the plot by using the stat smooth function with method equals lm. We can see that the linear trend line doesn't go through the center of the data at some key places. It misses it here. It should curve a little bit in the center of the relationship, and it should slope up more toward the end. And if we tried to use this to make predictions, we might be off for some key places inside and outside of the existing data that we have displayed.

Frances Gerety

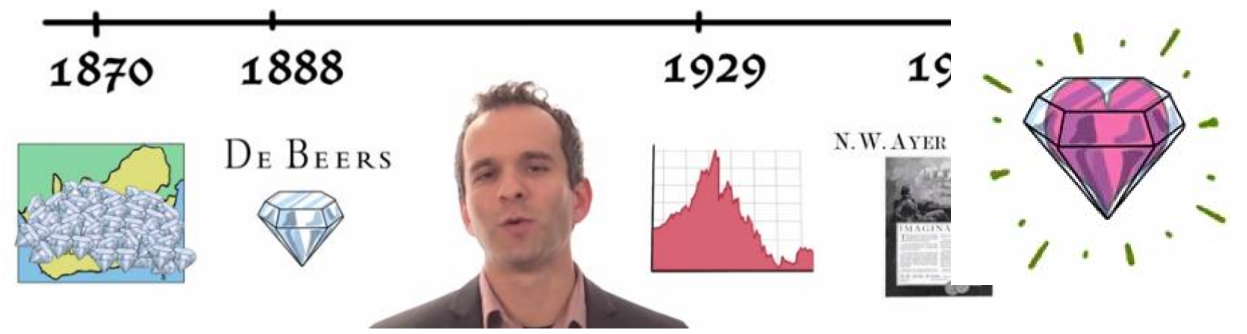
So far, we've only considered the bivariate relationship between price and carat weight. But there's much more to this data set and I'd like to give it a proper introduction. The diamonds data set ships with **ggplot2** and contains the prices and the specs for more than 50,000 diamonds collected in 2008 from diamondse.info. Now, analyzing this data is particularly useful. Because diamonds are unique in a way that just isn't true of most manufacture products that we're used to buying. You can't just plug in the model number and just look up the price. Though the diamonds data set is full of prices and fairly esoteric certification ratings. Hidden in the data are reflections of how a legendary marketing campaign permeated and was subsumed by our culture. Hints about how different social strata responded and how the diamond market functions today as a result. The story starts in 1870. When many tons of diamonds were discovered in South Africa near the Orange River. Until then the diamond market had been small, only a few pounds of diamonds were mined each year from India and Brazil. At the time, there was no use for diamonds outside of jewelry, so price depended only on scarce supply. Hence, the project's investors formed the De Beers Cartel in By most accounts, this has been the most successful cartel in history. But World War I and the Great Depression saw diamond sales plummet. In 1938, the De Beers Cartel contacted Philadelphia ad agency N.W. Ayer & Son to inquire whether, “The use of propaganda in various forms might help jump start diamond sales in the U.S.” Whi for diamonds at the time. Surveys showed, however, that among couples contemplating marriage, diamonds were low on the list of priorities. A luxury for the rich, money down the drain. Frances Gerety took on the DeBeers account at N.W. Ayer & Son. And worked towards the company's goal to “create a situation in which every couple contemplating marriage feels the need to acquire a diamond engagement rings.” A few years later, she would coin a famous slogan.

Quiz

Finish the famous DeBeers slogan, a diamond is: ________.

Answer

A diamond is forever. This tag along has appeared in every De Beers ad since 1948. In 1999, two weeks before Mrs. Gerety died, Advertising Age named it the slogan of the century.



The Rise of Diamonds

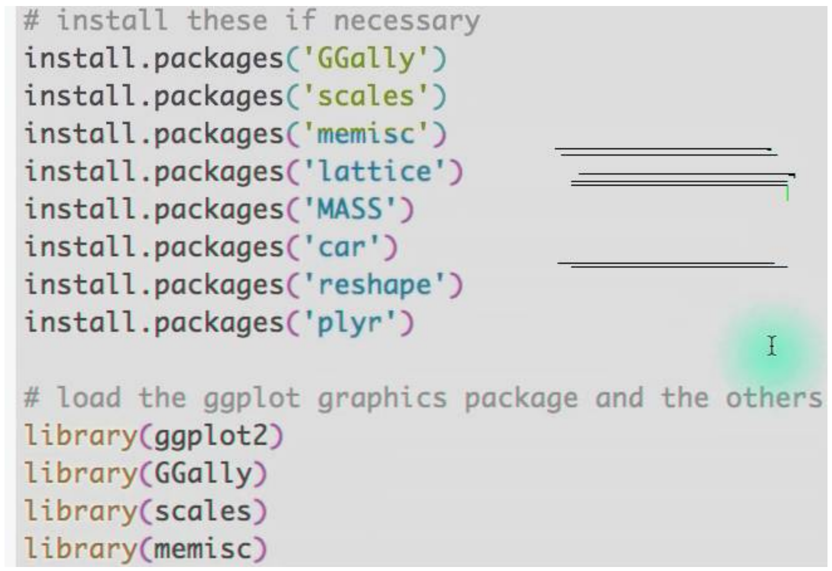
Many argue that this campaign gave birth to modern demand advertising. The objective here was not demand generation nor branch strengthening. But simply to impress the glamor, the sentiment and the emotional charge contained in the product, itself. The company gave diamonds to movie stars. They sent out press packets emphasizing the size of diamonds that celebrities gave each other. They loaned diamonds to prominent socialites attending events like the Kentucky Derby or the Academy Awards. And even persuaded the British royal family to wear diamonds over other gems. Later, De Beers sought to market diamond rings to couples as a status symbol. To reflect quote, a man's success in life. A 1980's add introduced the famous two month bench mark. Isn't two months salary a small price to pay for something that lasts forever? By any reasonable measure, Francis Geary succeeded. Getting engaged in America means getting a diamond ring. Can you think of a movie where two people get engaged with out a diamond ?

When you get engaged on Facebook, what icon does the site display? Still think this might not be the most successful mass persuasion effort in history? I present to you a James Bond film whose title bears the diamond cartel's trademark. Awe-inspiring and a little terrifying. Let's open the data set.

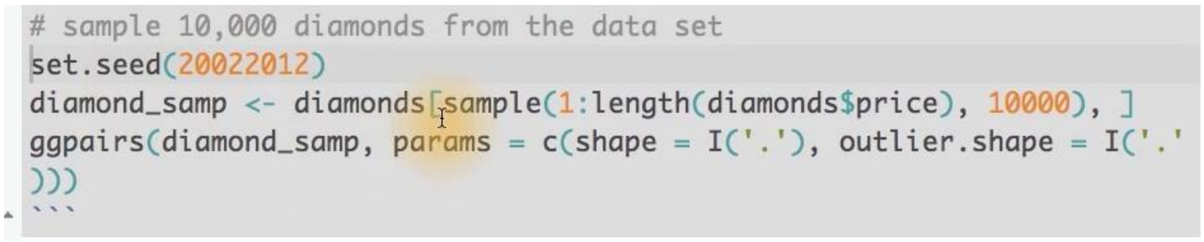
ggpairs Function

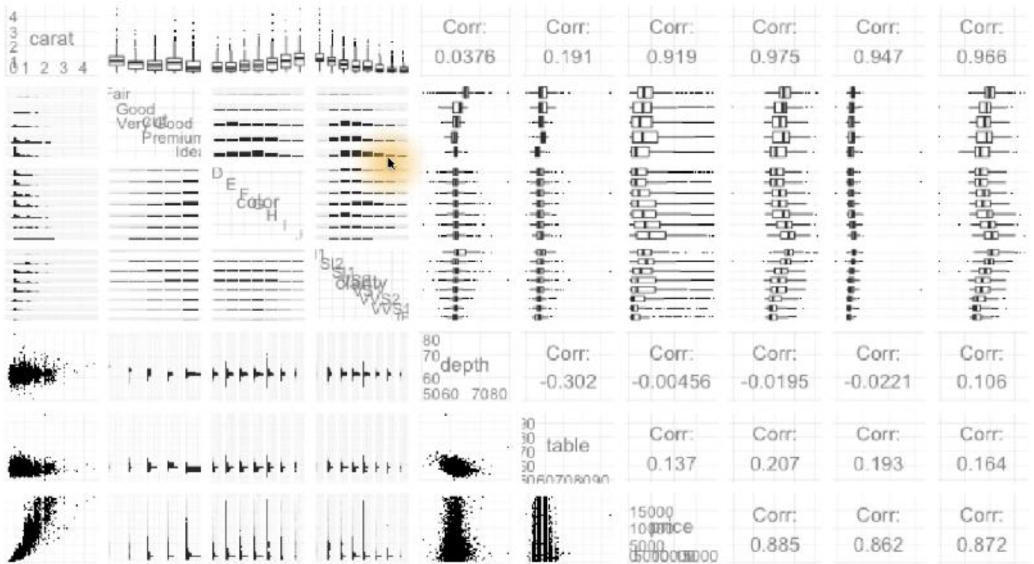
Keep this history in mind as we work through our exploratory analysis. The first thing you might consider doing is plotting key variables against each other using the **ggpairs** function. This function plots each variable against each other variable, pairwise. You may want to sample first, otherwise the function will take a long time to render the plots. Also, if your data set has more than about 10 columns, there will be too many plotting windows. So, subset on your columns first if that's the case. The first thing we want to do is make sure you have these packages installed. We use **ggally** for this particular plot. We use **scales** for a variety of things. **memisc** to summarize the regression. **lattice** for few other things. **mass** for various functions. **car** to recode variables. **reshape** to reshape and wrangle your data. **plyr** to create interesting summaries and transmissions that you've done. We'll go ahead and load

these packages, and then set the seed for randomization purposes and sample happening is that **ggpairs** is plotting each variable against the other in a pretty smart way. In the lower triangle of the plot matrix, it uses grouped histograms for qualitative, qualitative pairs and scatter plots for quantitative, quantitative pairs. In the upper triangle, it plots grouped histograms for qualitative, qualitative pairs, this time using the x instead of the y variable as the grouping factor. Box plots for qualitative, quantitative pairs, and it provides the correlation for quantitative quantitative pairs. Remember that our goal is to understand the price of diamonds.



```
# install these if necessary
install.packages('GGally')
install.packages('scales')
install.packages('memisc')
install.packages('lattice')
install.packages('MASS')
install.packages('car')
install.packages('reshape')
install.packages('plyr')

# load the ggplot graphics package and the others
library(ggplot2)
library(GGally)
library(scales)
library(memisc)
```



```
# sample 10,000 diamonds from the data set
set.seed(20022012)
diamond_samp <- diamonds[sample(1:length(diamonds$price), 10000), ]
ggpairs(diamond_samp, params = c(shape = I('.'), outlier.shape = I('.'
)))
```
```

So, let's focus on that.


```

### Quiz
The price variable is here. So, let's look at the relationships that correspond to price. What are some things you notice? There's no right or wrong answer here, but think deeply about the plots and the associations you see. Write a few sentences about what stands out to you.

### Answer
We can see what might be relationships between price and clarity and price and color, which we'll keep in mind for later when start modelling our data. You might remember this when you create the box plots in problem set three. Be that as it may, the critical factor driving price is the size, or the carat weight of the diamond. As we saw at the start of the lesson, the relationship between price and diamond size is nonlinear. What might explain this pattern? On the supply side, larger continuous chunks of diamonds without significant flaws are probably harder to find than smaller ones. This might help explain the sort of exponential looking curve, and I thought I noticed this when I was shopping for a diamond for my soon to be wife. Of course, this is related to the fact that the weight of a diamond is a function of volume, and volume is a function of the

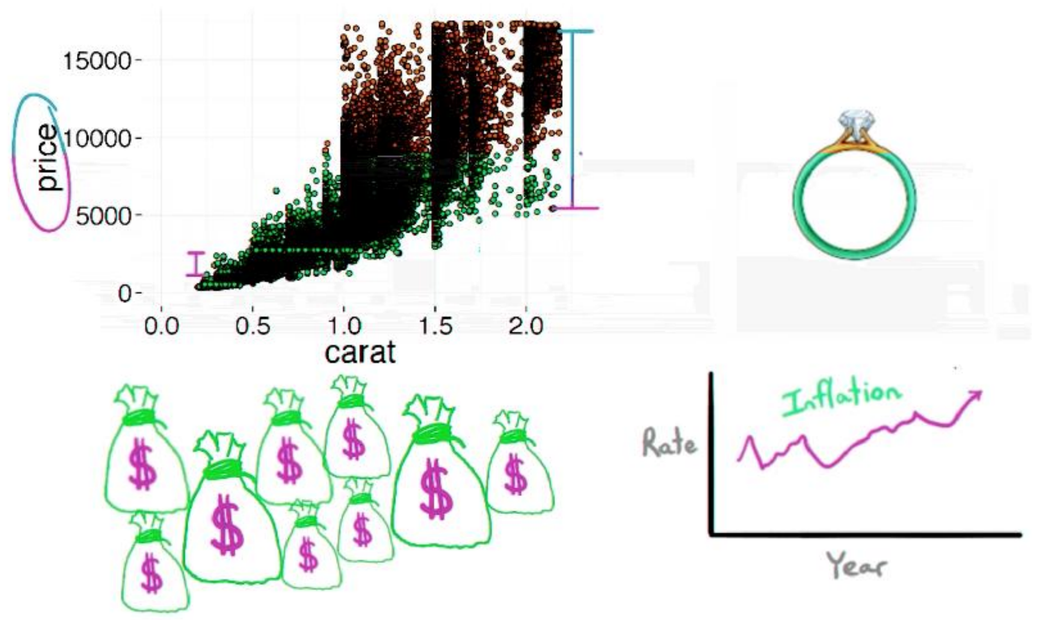
$$\text{Weight} \approx f(\text{Volume}) \approx f(x \cdot y \cdot z)$$

$$\sqrt[3]{\text{carat weight}}$$

length times thewidth times the height of a diamond, and this suggests that we might be especially interested in the cube root of carat weight. It's often the case, that leveraging substantive knowledge about your data like this can lead to especially fruitful transformations, and we'll see that in a second.

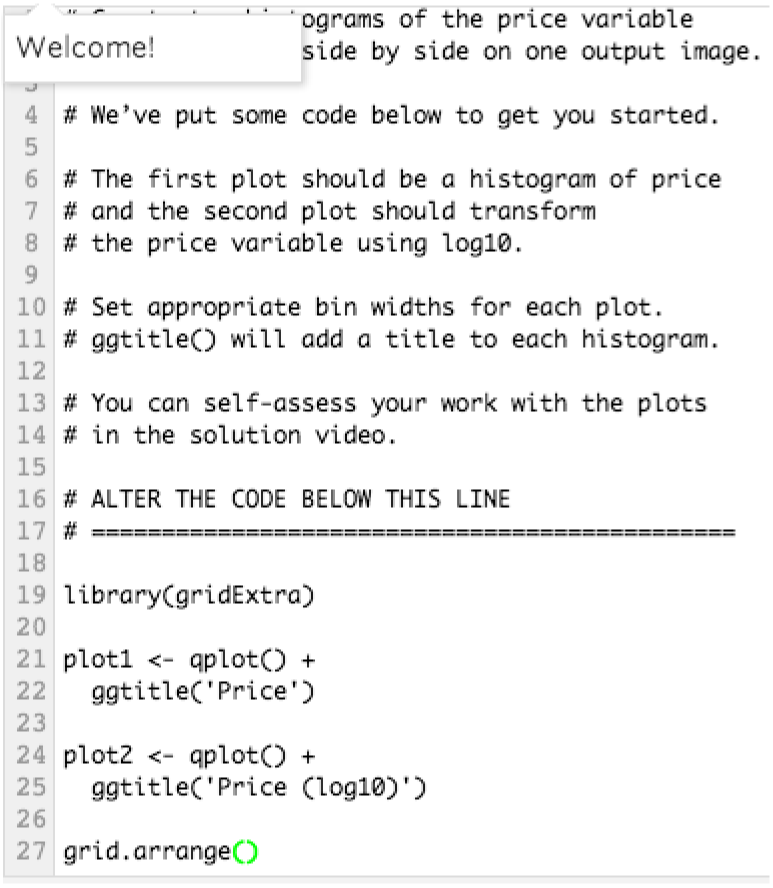
## The Demand of Diamonds

On the demand side, customers in the market for a less expensive, smaller diamond are probably more sensitive to price than more well-to-do buyers. Many less than one carat customers would surely never buy a diamond. Diamond were not for the social norm of presenting one when proposing. And there are fewer customers who can afford a bigger diamond that is one that is larger than than one carat, hence we shouldn't expect the market for bigger diamonds to be as competitive as the one for smaller diamonds. So it makes sense that the variants As well as the price would increase with carat size. Now often the distribution of any monetary variable like dollars will be highly skewed and vary over orders of magnitude. Now this can result from path dependence for example the rich getting richer, or multiplicative processes



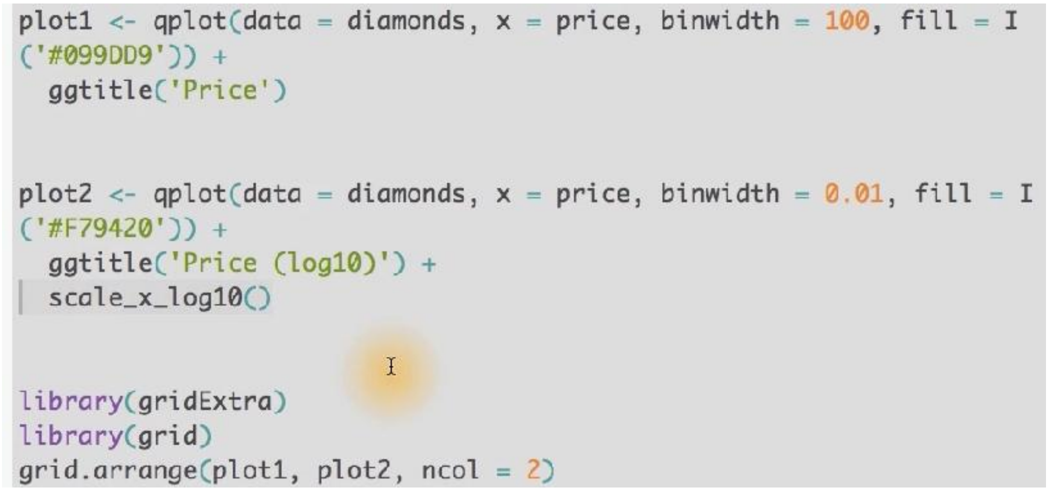
like year on year inflation, or some combination of both. Hence it's a good idea to look into compressing any such variable by putting it on a log scale. For more tips on when to use the log scale and how to transform your variables check out the link. Let's examine the distribution of price again. In this next programming exercise you complete the code to generate two price histograms.

## Programming Quiz



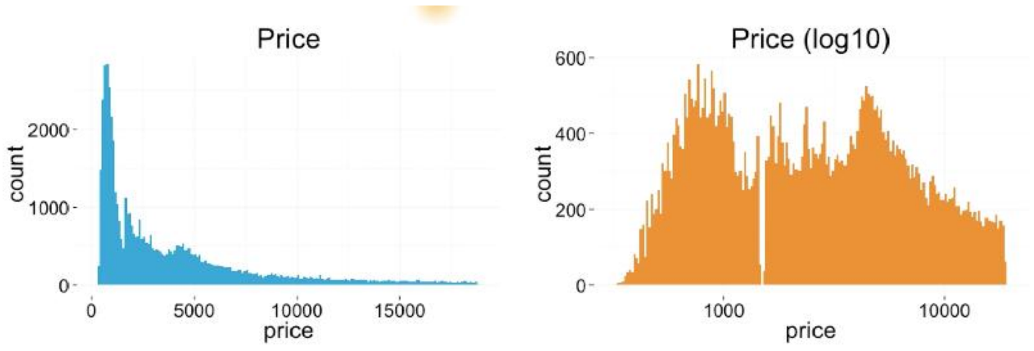
```
# C       h       ograms of the price variable
#                 side by side on one output image.

# We've put some code below to get you started.

# The first plot should be a histogram of price
# and the second plot should transform
# the price variable using log10.

# Set appropriate bin widths for each plot.
# ggtitle() will add a title to each histogram.

# You can self-assess your work with the plots
# in the solution video.

# ALTER THE CODE BELOW THIS LINE
# ==============================================

library(gridExtra)

plot1 <- qplot() +
  ggtitle('Price')

plot2 <- qplot() +
  ggtitle('Price (log10)')

grid.arrange()
```

Answer



```
plot1 <- qplot(data = diamonds, x = price, binwidth = 100, fill = I
('#099DD9')) +
  ggtitle('Price')


plot2 <- qplot(data = diamonds, x = price, binwidth = 0.01, fill = I
('#F79420')) +
  ggtitle('Price (log10)') +
  scale_x_log10()


library(gridExtra)
library(grid)
grid.arrange(plot1, plot2, ncol = 2)
```

Here we're just using **qplot** to produce two histograms of price. One on just a regular scale and one on the **log10** scale. We're to use this **gridExtra** library to plot both



plots in the same window and here's what the result looks like.

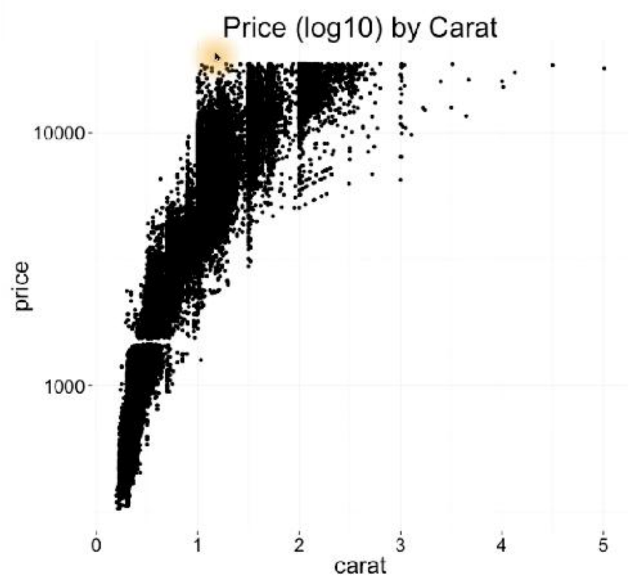
Connecting Demand and Price Distribution

When looking at these plots, what do you notice? Think specifically about the two peaks in the transformed plot and how they relate to the demand for diamonds. Alright let me know what you think here.

Answer

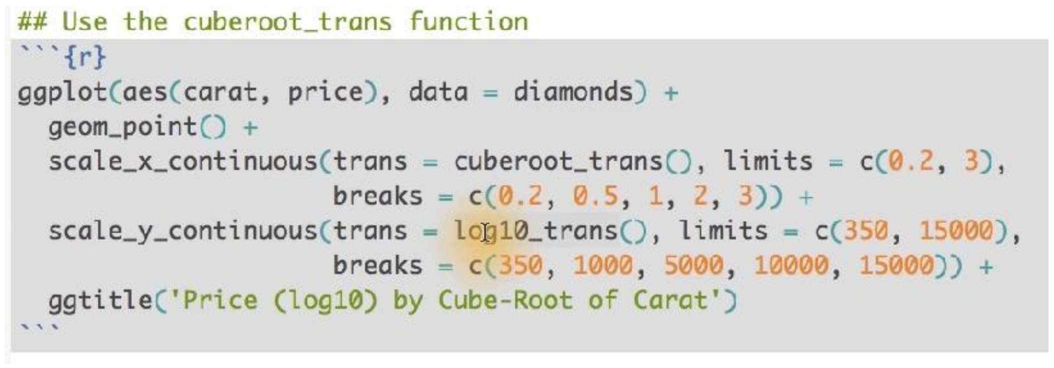
Indeed we can see that the prices for diamonds are pretty heavily skewed, but when you put those prices on a log ten scale, they seem much better behaved. They're much closer to the bell curve of a normal distribution. We can even see a little bit of evidence of bimodality on this $\log_{10}$ scale. Which is consistent with our two class rich buyer poor buyer speculation about the nature of customers for diamonds. You actually saw a sneak peek of this in problem set set three when you looked at the $\log_{10}$ of price by cut.

Scatterplot Transformation



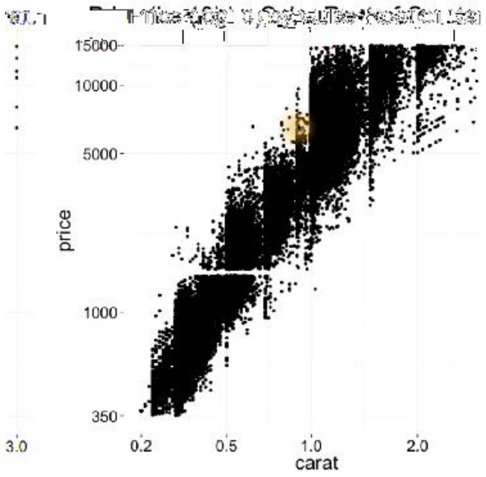
Now that we have a better understanding of our variables, and the overall demand for diamonds, let's replot the data. This time we'll put price on a $\log_{10}$ scale, and here's what it looks like. This plot looks better than before. On the log scale, the prices look less dispersed at the high end of Carat size and price, but actually we can do better. Let's try using the cube root of Carat in light of our speculation about flaws being exponentially more likely in diamonds with more volume. Remember, volume is on a cubic scale. First, we need a function to transform the Carat variable. If you'd like to learn more about writing your own functions in R, check out this link about the basic structure of functions and this blog post about using defining new transformations in ggplot2 and the scales package

This may seem like a lot of code, but really, there's only one new piece here. It's this **cuberoot_trans** function. It's a function that takes the cube root of any input variable, and it also has an inverse function to undo that operation, which we need to display the plot correctly. Then when we get to our actual **ggplot** command. What



~~~
## Use the cuberoot_trans function
```{r}
ggplot(aes(carat, price), data = diamonds) +
  geom_point() +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat')
```
~~~

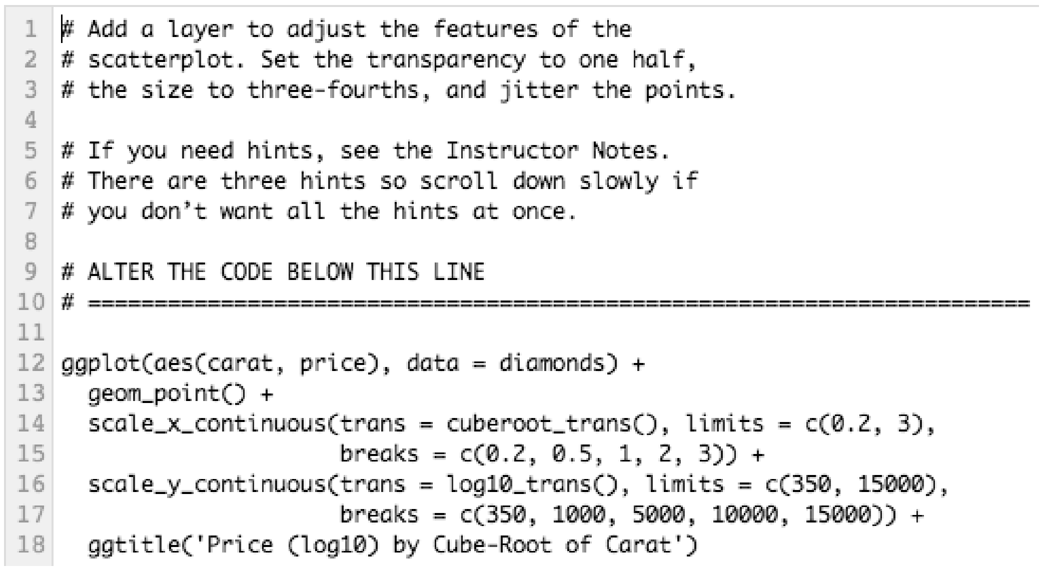
we'll do is we'll use the **scale_x_continuous** argument to transform the x-axis with this cube root transformation function. Keep in mind we're also transforming the y axis with this $\log_{10}$ transformation that we discussed previously. And, let's see what this plot looks like. Taking a look at the plot, we can actually see that with these transformations that we used to get our data on this nice scale. Things look almost linear. We can now move forward and see about modelling our data using just a linear model.



Overplotting Revisited

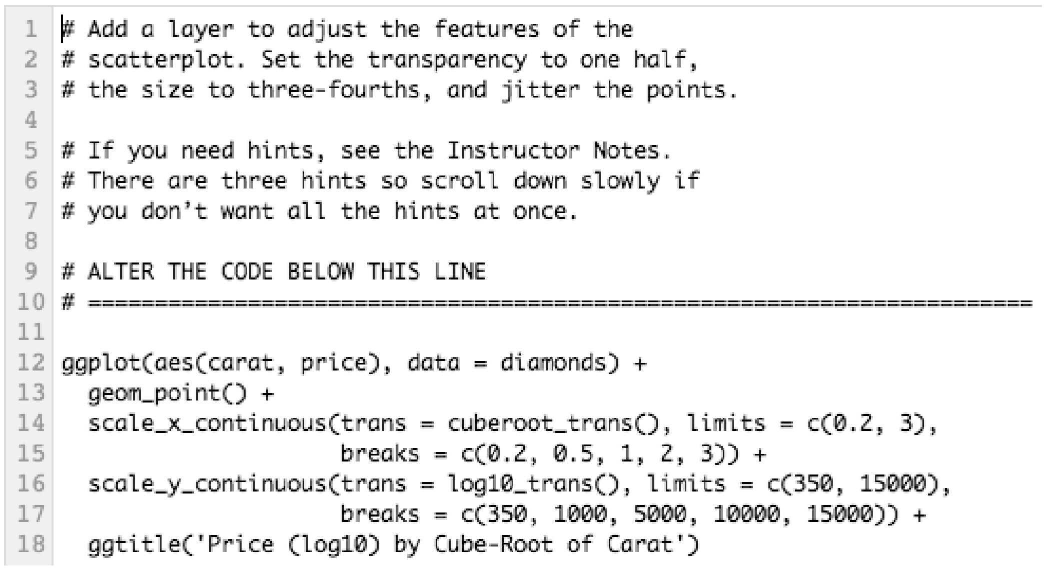
Until now we haven't really done anything about over plotting. That's when multiple points take on the same value. This is often due to rounding. So let's look at this by running some code. We're going to run the **table** command on both carat, and price. Then we're going to sort that so that the highest values appear first. We're just going to look at the top six which is the default for the head function. Of course, what's happening here is that the first line is the price or the carat, and the second line is the count for each one of these values. We can see that these are really high numbers which is going to result in a substantial amount of overplotting. When you have this much data, you're going to have serious overplotting, even when you're plotting the variables against each other, and this can really obscure some of the density and the sparsity of our data at really key points. Okay, so as we discussed in lesson four, you can deal with this by making your points smaller by jittering your points and by adding transparency. In **ggplot**, this is done with the **alpha** parameter. In the next coding exercise, you'll do just that.

Programming Quiz



```
# Add a layer to adjust the features of the
# scatterplot. Set the transparency to one half,
# the size to three-fourths, and jitter the points.

# If you need hints, see the Instructor Notes.
# There are three hints so scroll down slowly if
# you don’t want all the hints at once.

# ALTER THE CODE BELOW THIS LINE
# ==========================================================================

ggplot(aes(carat, price), data = diamonds) +
  geom_point() +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat')
```

Answer



```
# Add a layer to adjust the features of the
# scatterplot. Set the transparency to one half,
# the size to three-fourths, and jitter the points.

# If you need hints, see the Instructor Notes.
# There are three hints so scroll down slowly if
# you don't want all the hints at once.

# ALTER THE CODE BELOW THIS LINE
# ======================================================================

ggplot(aes(carat, price), data = diamonds) +
  geom_point() +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat')
```

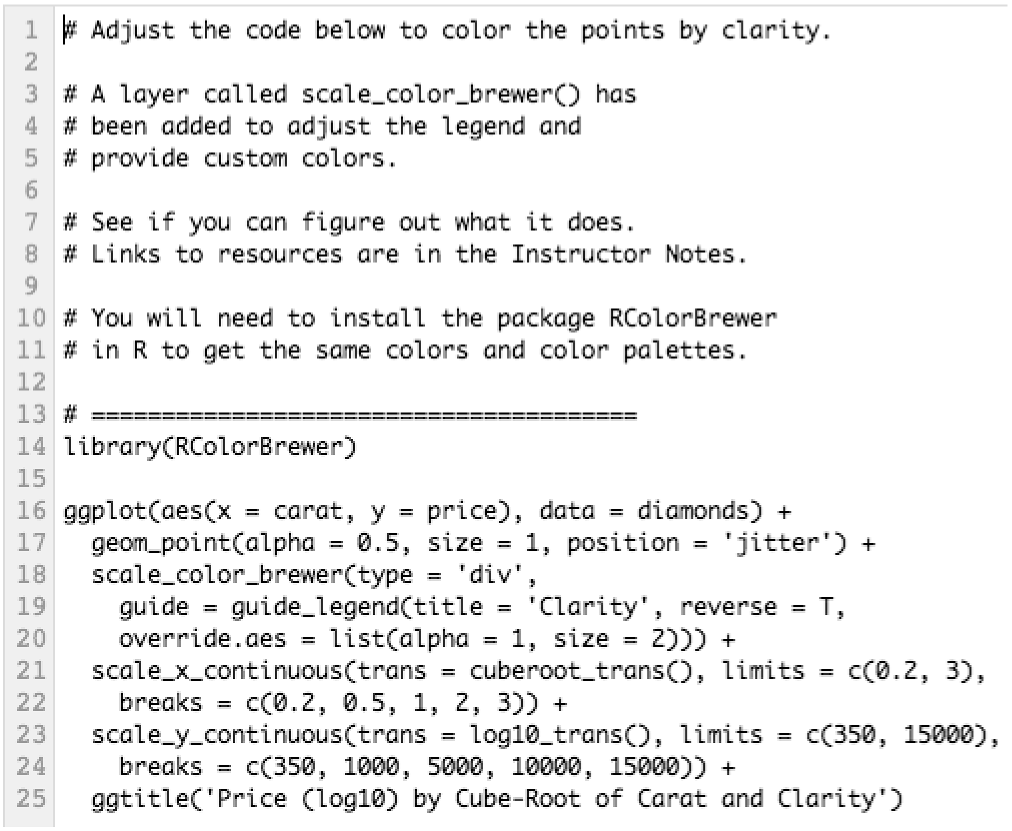
Your code should look something like this. After making these 3 adjustments, you can see that the plot gives us a better sense of how dense and how sparse our data is at key places. See?

Plot Colors for Qualitative Factors

We can see what looks like a almost linear relationship between carat weight and price after doing some transformations, but surely there are other factors that influence the price of a diamond. When I was looking around at diamonds I noticed that clarity seemed to factor into price. Of course, many consumers are looking for a diamond of a certain minimum size. So we shouldn't expect clarity to be as strong a factor as carat weight. And I must admit that even though my grandparents were jewelers. I initially had a hard time discerning a diamond rated VVS1 from one rated SI2. Surely, most people need a jeweler's loupe to tell the difference. What makes a diamond sparkle anyway? According to blue nile, the cut of a diamond has a much more consequential impact on that fiery quality that jewelers describe when they talk about diamonds. On clarity, the website states many of these imperfections are microscopic and do not affect the diamonds beauty in any discernible way.

Price vs. Carat and Clarity

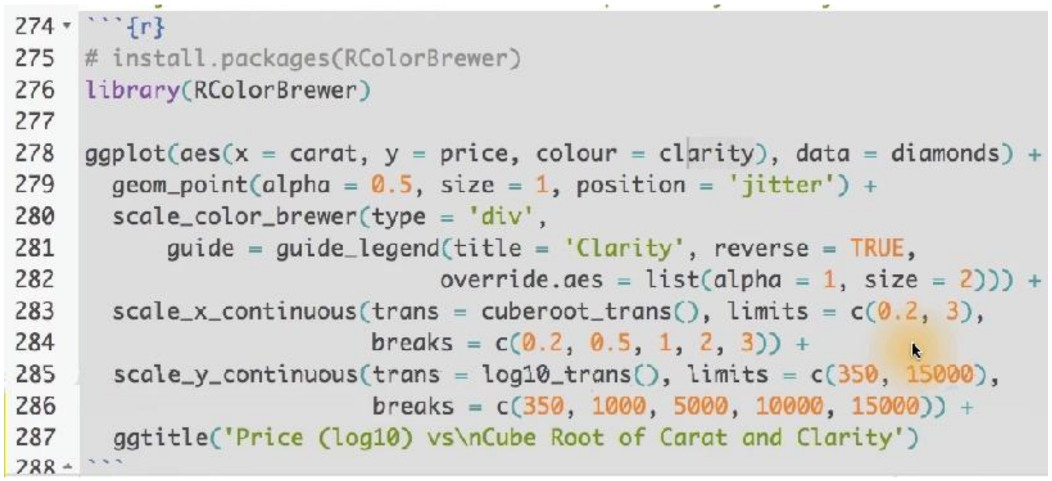
Let's see if clarity, cut, or color can explain some of the variants in price when we visualize it on our plot using color. We'll start by examining clarity. This code visualizes price by carat. Adjust it to color the points by clarity.



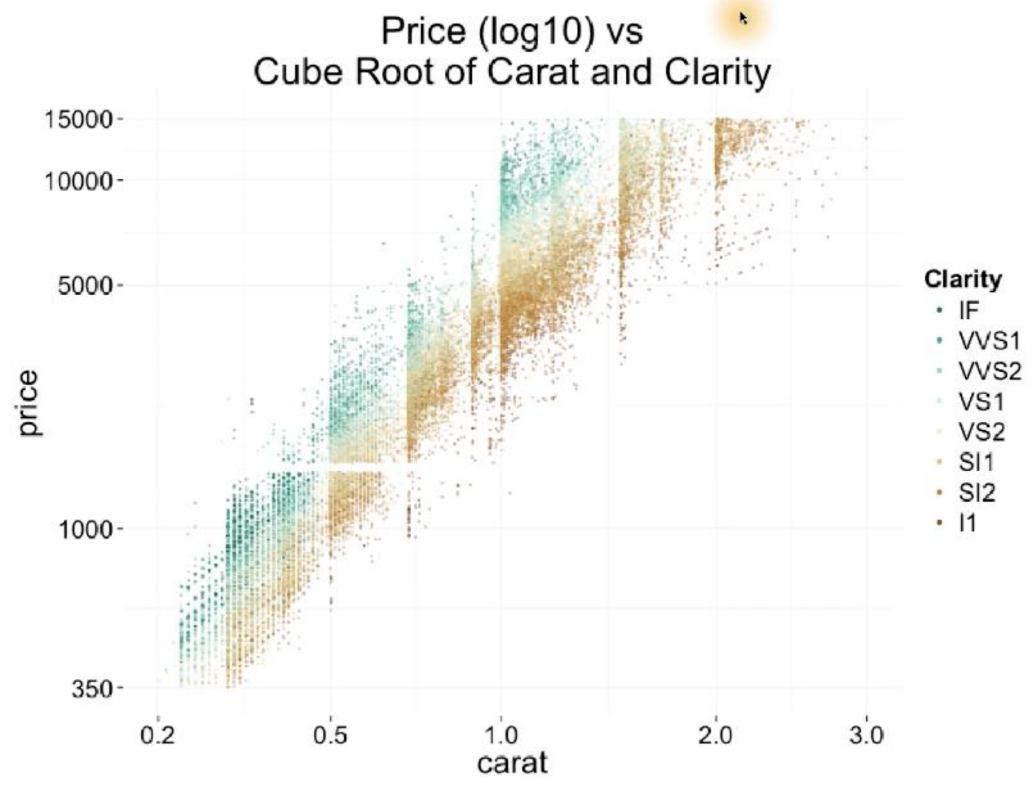
```r
# Adjust the code below to color the points by clarity.

# A layer called scale_color_brewer() has
# been added to adjust the legend and
# provide custom colors.

# See if you can figure out what it does.
# Links to resources are in the Instructor Notes.

# You will need to install the package RColorBrewer
# in R to get the same colors and color palettes.

# =======================================
library(RColorBrewer)

ggplot(aes(x = carat, y = price), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
    guide = guide_legend(title = 'Clarity', reverse = T,
    override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
    breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
    breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat and Clarity')
```

Programming Quiz

Answer



~~~r
```{r}
# install.packages(RColorBrewer)
library(RColorBrewer)

ggplot(aes(x = carat, y = price, colour = clarity), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
      guide = guide_legend(title = 'Clarity', reverse = TRUE,
                          override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) vs\nCube Root of Carat and Clarity')
~~~
```

We just need to add one parameter to our aesthetic wrapper in ggplot like so. Set the color parameter for equal to clarity. And when we run it, here's what it looks like.
```



Clarity and Price

Based on the plot, do you think clarity explains some of the change in price? Why? Enter your response here.

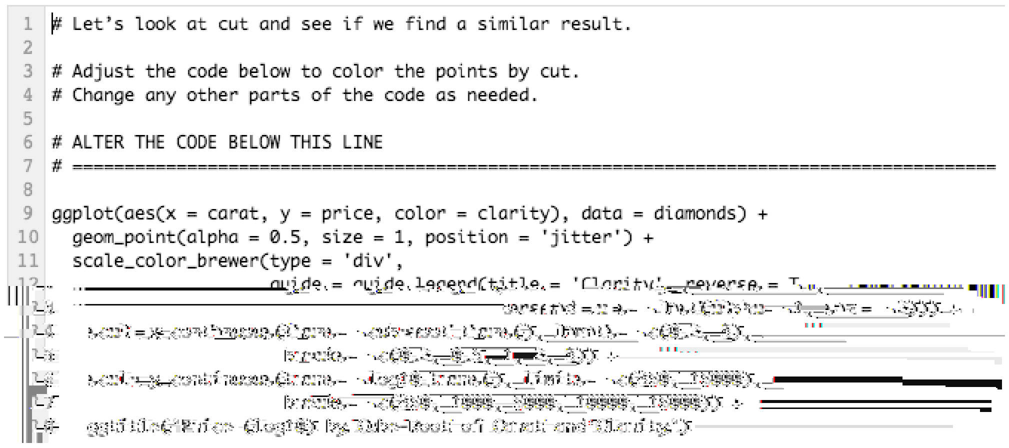
Answer

Clarity does seem to explain an awful lot of the remaining variance in price, after adding color to our plot. Holding carat weight constant, we're looking at one part of the plot. We see the diamonds with lower clarity are almost always cheaper than diamonds with better clarity.

Price vs. Carat and Cut

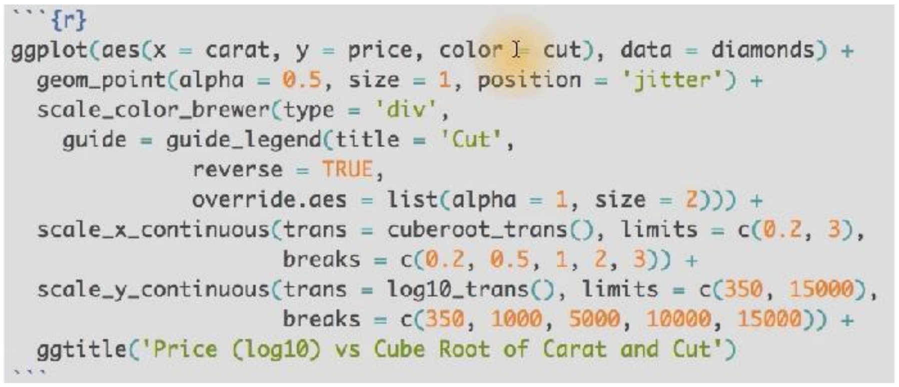
Now let's look at cut. Change the code such that you're coloring the points by cut, and don't forget to change the titles.

Programming Quiz



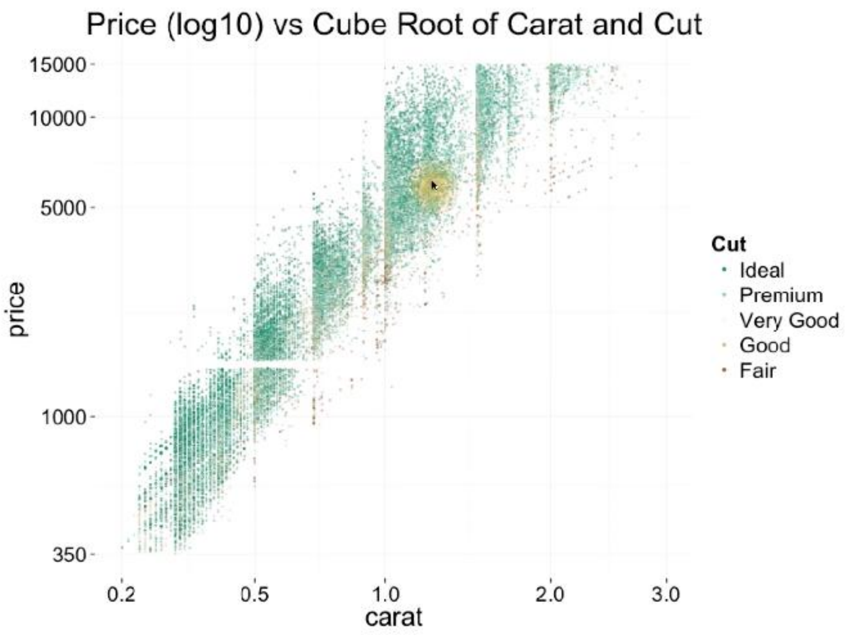
```
# Let's look at cut and see if we find a similar result.

# Adjust the code below to color the points by cut.
# Change any other parts of the code as needed.

# ALTER THE CODE BELOW THIS LINE
# ==========================================================================================

ggplot(aes(x = carat, y = price, color = clarity), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
                     guide = guide_legend(title = 'Clarity', reverse = T,
                                          override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat and Clarity')
```

Answer



```{r}
ggplot(aes(x = carat, y = price, color = cut), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
    guide = guide_legend(title = 'Cut',
                         reverse = TRUE,
                         override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                           breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                           breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) vs Cube Root of Carat and Cut')
```

You might have made just one change to the code by swapping out the cut variable with the Clarity variable, but I hope you changed your titles as well. Running the code, here is the plot that we get.



Cut and Price

Based on the plot, do you think that cut accounts for some of the variance in price, why? Enter your response here.

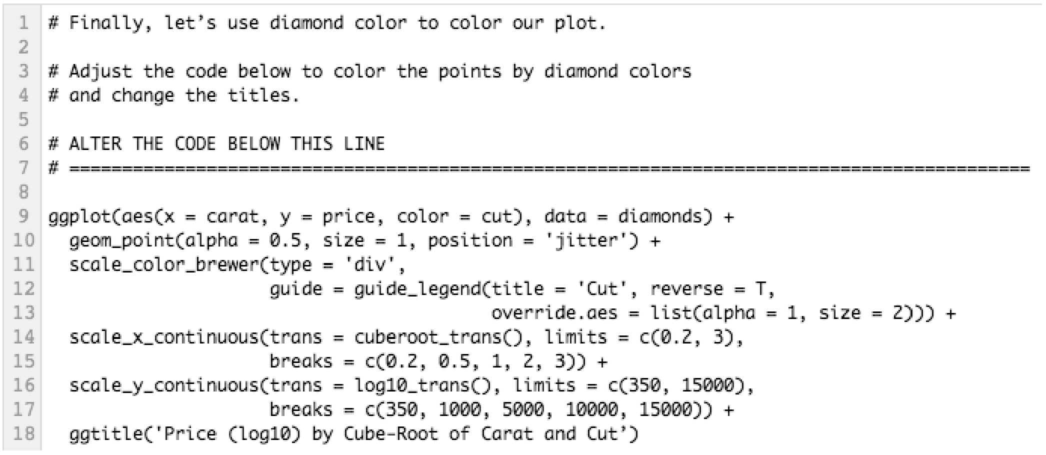
Answer

Despite what Blue Nile says, we don't see much variation on cut. Most of the diamonds in the data are ideal cut anyway, so we've lost the color pattern that we saw before.

Price vs. Carat and Color

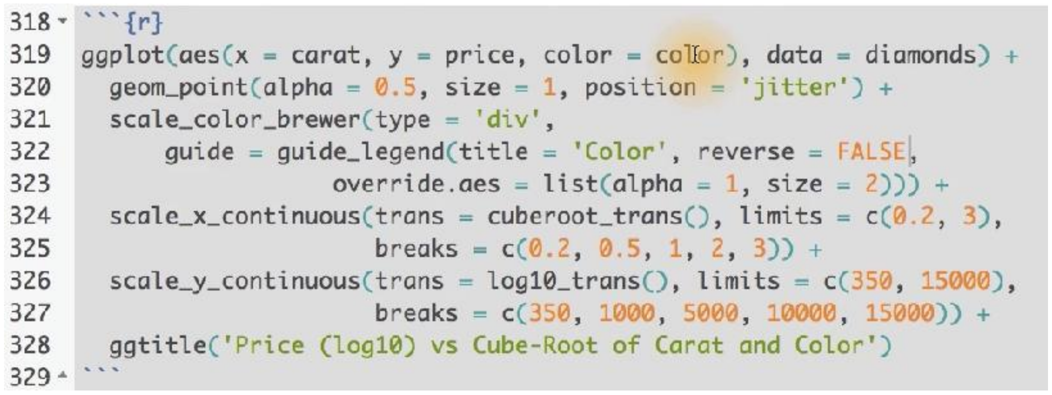
Finally, let's use diamond color as a color in our plot. Adjust the code below, to color the points by diamond color.

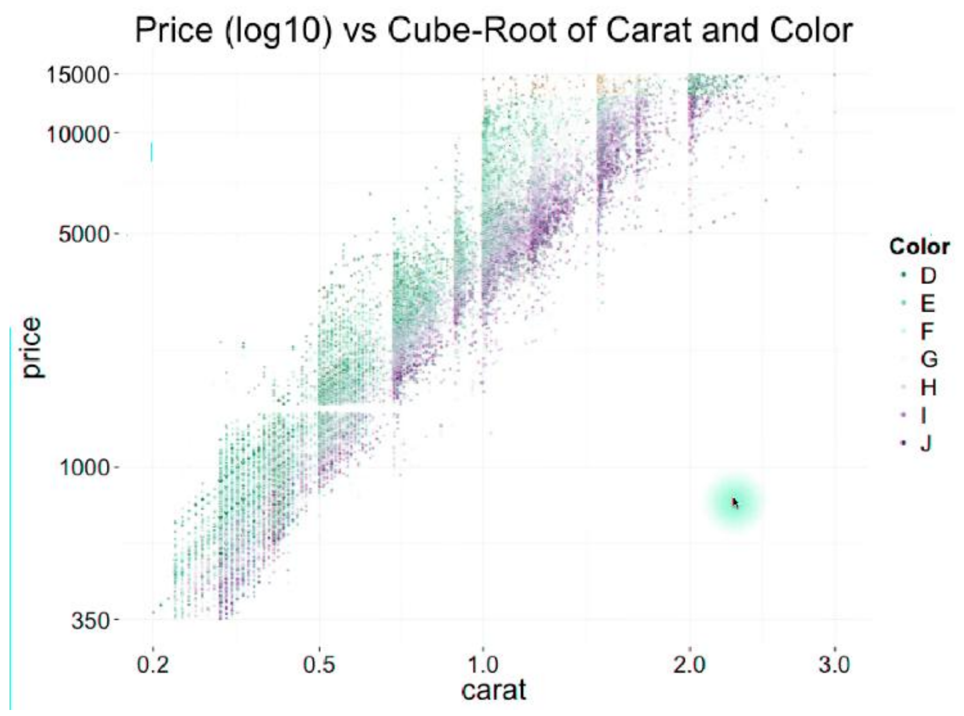
Programming Quiz



```
# Finally, let’s use diamond color to color our plot.

# Adjust the code below to color the points by diamond colors
# and change the titles.

# ALTER THE CODE BELOW THIS LINE
# ===========================================================================================

ggplot(aes(x = carat, y = price, color = cut), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
                     guide = guide_legend(title = 'Cut', reverse = T,
                                          override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) by Cube-Root of Carat and Cut’)
```

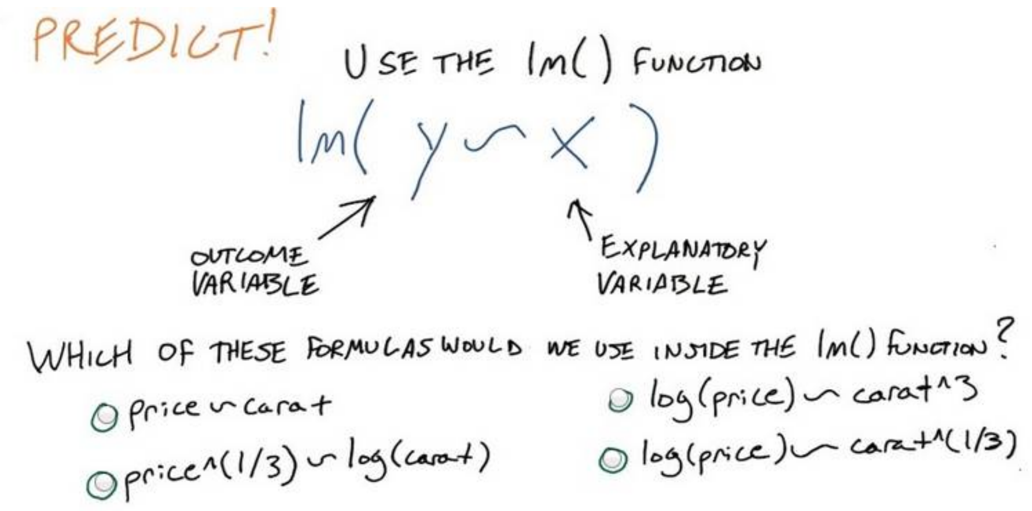
Answer



~~~r
```{r}
ggplot(aes(x = carat, y = price, color = color), data = diamonds) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter') +
  scale_color_brewer(type = 'div',
      guide = guide_legend(title = 'Color', reverse = FALSE,
                    override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                      breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                      breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Price (log10) vs Cube-Root of Carat and Color')
~~~
```

Here, I made four changes to the code. I replaced cut with color in the aesthetic wrapper. I also changed the legend title and the plot title, to use the word color. And finally, I removed the reverse parameter in the legend, so that the best color would be at the top of the list in the legend. Now that we've run our code, this is the output.



Color and Price

Based on the plot, do you think that the color of the diamond influences price? Why? Enter your answer in the box.

Answer

Color does seem to explain some of the variance in price. Just like we saw with the clarity variable. Blue now however, states that the difference between all color grades from D to J are basically not noticeable to the naked eye. Yet, we do see the color difference in the price tag.
```

## Linear Models in R

In R we can create models using the lm function. We need to supply a formula in the form of y~x. Here, y is the outcome variable and x is the explanatory variable. Which of the following formulas would we use inside the lm function? Price~^. Price to the cube root ~log^ log price ~^ carat to the third, log price ~^ to the 1 3rd, select one.

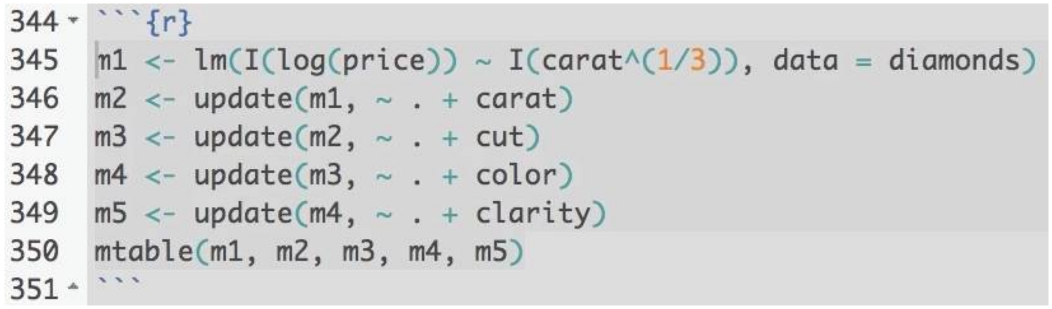


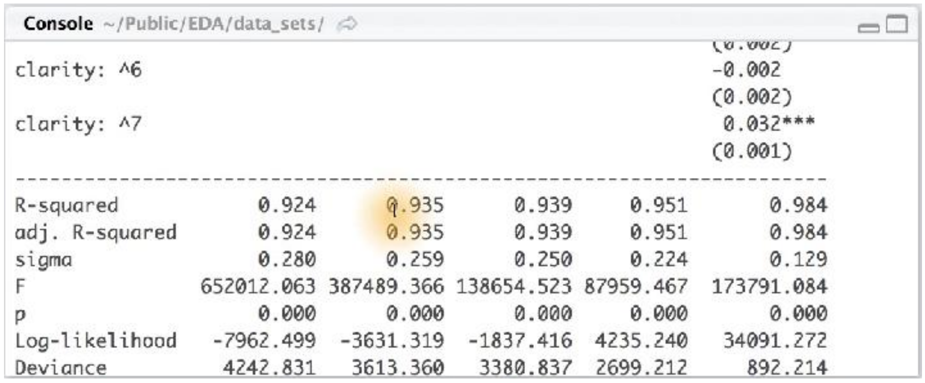
### Answer

Remember we applied the log transformation to our long tailed dollar variable, and we speculated that the flawless diamond should become exponentially rarer as diamond volume increases. So we should be interested in the cube root of carat weight.

## Building the Linear Model

Let's build up our linear model for price. I'm going to store the first model in a variable called **m1**. You probably also noticed how I used the I wrapper around each of the variables. The "I" stands for as is. In this case, it tells R to use the expression inside the "**I**" function to transform a variable before using it in the regression. This is instead of instructing R to interpret these symbols as part of the formula to construct the design matrix for the regression. You can read more about the syntax for linear models online. I can also update the previous model to add the carat variable in the regression, using the syntax. The real functional relationship is surely not as simple as the cubed root of carat, so we add a simple linear function of carat in our model predicting price. And we can continue to make more complex models by adding more variables. We add cut even though we don't expect it to have much influence on price. Next, we add color to a fourth model and clarity to a fifth. When we run the code, we can see that we're getting some very nice R square values. We're accounting



~~~r
```{r}
m1 <- lm(I(log(price)) ~ I(carat^(1/3)), data = diamonds)
m2 <- update(m1, ~ . + carat)
m3 <- update(m2, ~ . + cut)
m4 <- update(m3, ~ . + color)
m5 <- update(m4, ~ . + clarity)
mtable(m1, m2, m3, m4, m5)
~~~
```

for almost all of the variance in price using the four "C"s. If we want to know whether the price for, a diamond is reasonable, we might now use this model.



```
                                                                    (0.002)
clarity: ^6                                                         -0.002
                                                                    (0.002)
clarity: ^7                                                          0.032***
                                                                    (0.001)
------------------------------------------------------------------------------
R-squared              0.924      0.935      0.939      0.951       0.984
adj. R-squared         0.924      0.935      0.939      0.951       0.984
sigma                  0.280      0.259      0.250      0.224       0.129
F                 652012.063 387489.366 138654.523 87959.467  173791.084
p                      0.000      0.000      0.000      0.000       0.000
Log-likelihood     -7962.499  -3631.319  -1837.416  4235.240   34091.272
Deviance            4242.831   3613.360   3380.837  2699.212     892.214
```

$$\ln(\text{Price}) = 0.415 + 9.144 * \sqrt[3]{\text{carat}} - 1.093 * \text{carat} + (\ldots * \text{cut} + \ldots * \text{color} + \ldots * \text{clarity}) + \varepsilon$$

Model Problems

Our model is the log of price equals 0.415 plus 9.144 times the cube root of carat, minus 1.093 times carat plus a series of coefficient times each factor in cut another series of coefficient times each factor in color. And another series of coefficients times each factor in clarity plus an error term.

Quiz
```

What could be some problems when using this model. What else should we think about when we're using it. Go ahead and enter your answer here.

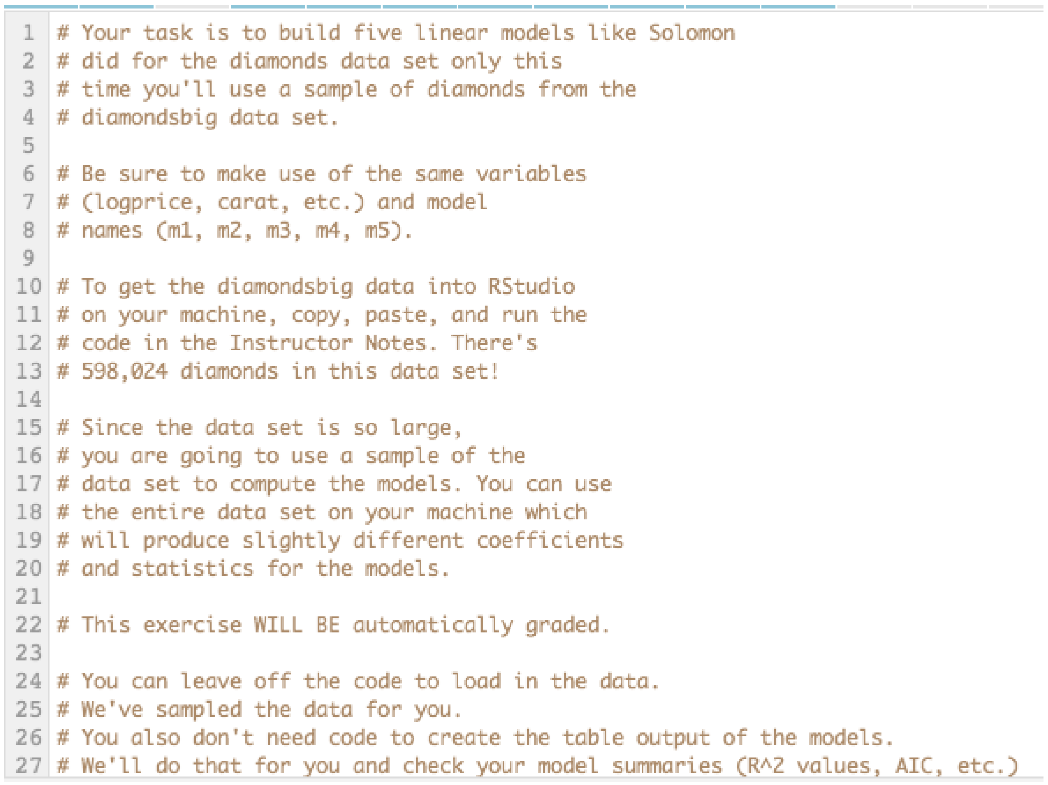
Answer

There was so much you might have said here. And, if you have any ideas that I don't mention, please share them in the forum. To start, our data's from 2008. So, not only do we need to account for inflation, but the diamond market is quite different now than it was. In fact, when I fit models to this data and predicted the price of the diamonds that I found off a market, I kept getting predictions that were way too low. After some additional digging, I found that global diamonds were poor. It turns out that prices plummeted in 2008 due to the global financial crisis. And since then prices, at least for wholesale polished diamonds, have grown at about of couples in China buying diamond engagement rings might also explain this increase. You can click on the links in the instructor notes to learn more about that. And finally, after looking at the data on price scope, I realize that diamond prices grew unevenly across different carat sizes since 2008, meaning that the model I initially estimated couldn't simply be adjusted by inflation.

A Bigger, Better Data Set

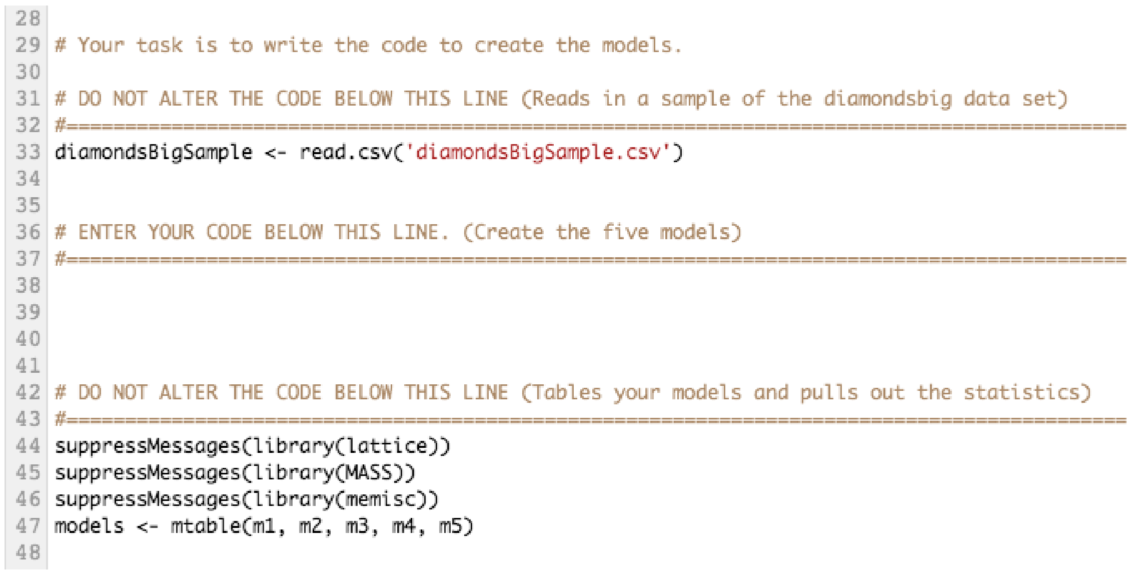
Thankfully, I was able to put together a Python script to get the current diamond price data, similar to the original diamonds data set, from diamondse.info without too much trouble. This data set is about ten times the size of the 2008 diamonds data set, and features diamonds from all over the world certified by an array of authorities, besides just the Gemological Institute Of America, or the GIA. You can read in this data set as follows, but make sure you've installed the RCurl and bitops libraries before you do. This might take a while as a data set contains over about

500,000 cases. The code used to obtain the data is available at this link. Let's fit the model again to the big data set. We'll only use GIA certified diamonds in this log. I look only at diamonds under $10,000 because these are the type of diamonds sold at most retailers, and hence, the kind we care most about. By trimming the most expensive diamonds from the data set, our model will also be less likely to be thrown off by outliers, and the higher variants at the high-end of price and carat. You can learn how to scrape data from the web by taking Data Wrangling with MongoDB: Data Manipulation and Retrieval



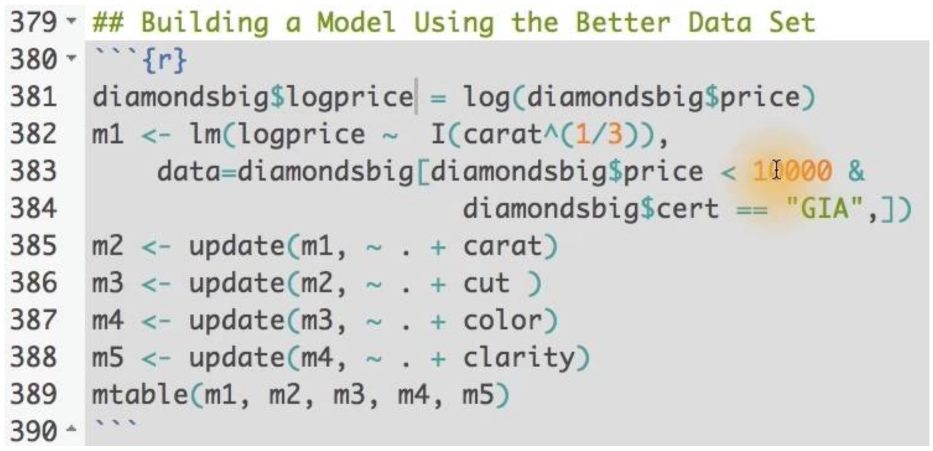
```
# Your task is to build five linear models like Solomon
# did for the diamonds data set only this
# time you'll use a sample of diamonds from the
# diamondsbig data set.

# Be sure to make use of the same variables
# (logprice, carat, etc.) and model
# names (m1, m2, m3, m4, m5).

# To get the diamondsbig data into RStudio
# on your machine, copy, paste, and run the
# code in the Instructor Notes. There's
# 598,024 diamonds in this data set!

# Since the data set is so large,
# you are going to use a sample of the
# data set to compute the models. You can use
# the entire data set on your machine which
# will produce slightly different coefficients
# and statistics for the models.

# This exercise WILL BE automatically graded.

# You can leave off the code to load in the data.
# We've sampled the data for you.
# You also don't need code to create the table output of the models.
# We'll do that for you and check your model summaries (R^2 values, AIC, etc.)
```

Programming Quiz



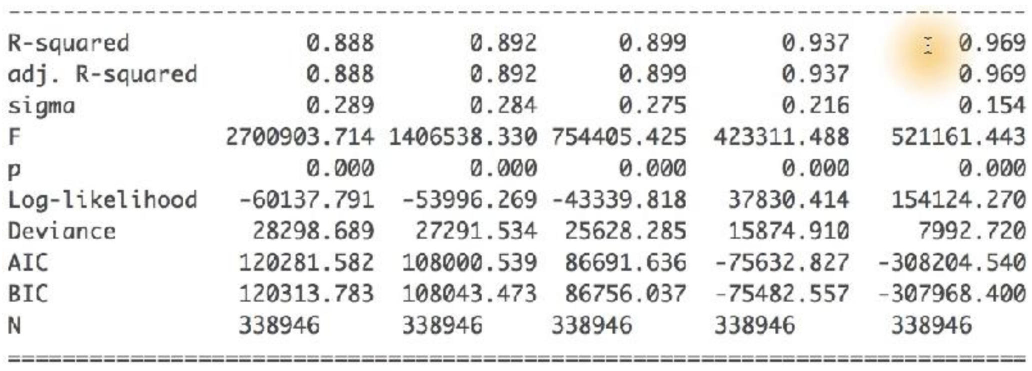
```r

# Your task is to write the code to create the models.

# DO NOT ALTER THE CODE BELOW THIS LINE (Reads in a sample of the diamondsbig data set)
#===========================================================================================
diamondsBigSample <- read.csv('diamondsBigSample.csv')


# ENTER YOUR CODE BELOW THIS LINE. (Create the five models)
#===========================================================================================




# DO NOT ALTER THE CODE BELOW THIS LINE (Tables your models and pulls out the statistics)
#===========================================================================================
suppressMessages(library(lattice))
suppressMessages(library(MASS))
suppressMessages(library(memisc))
models <- mtable(m1, m2, m3, m4, m5)

```

Answer



~~~
## Building a Model Using the Better Data Set
```{r}
diamondsbig$logprice = log(diamondsbig$price)
m1 <- lm(logprice ~  I(carat^(1/3)),
    data=diamondsbig[diamondsbig$price < 10000 &
                         diamondsbig$cert == "GIA",])
m2 <- update(m1, ~ . + carat)
m3 <- update(m2, ~ . + cut )
m4 <- update(m3, ~ . + color)
m5 <- update(m4, ~ . + clarity)
mtable(m1, m2, m3, m4, m5)
```
~~~

What we will do first is make a variable called logged price, the log of diamond price. And then we will build our model similarly to the way that we did for the small diamond status set. Notice that we're setting price by diamonds whose price is less than $10,000 and whose certificate is G.I.A.. Thankfully our models look quite a bit like they did for the smaller diamond status set. Although our R squared values are just a touch weaker.

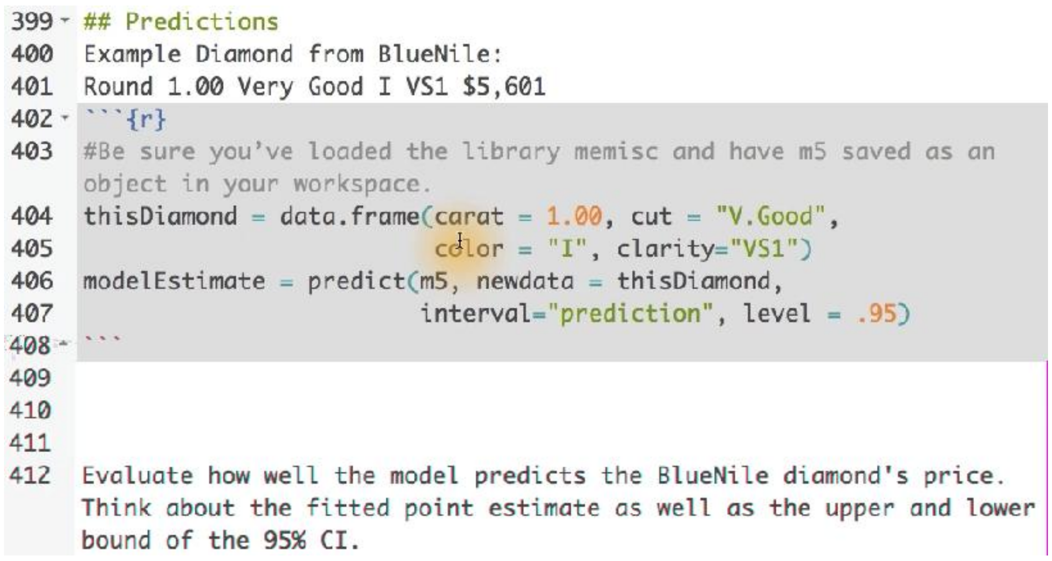


```
---------------------------------------------------------------------------
R-squared              0.888       0.892       0.899       0.937       0.969
adj. R-squared         0.888       0.892       0.899       0.937       0.969
sigma                  0.289       0.284       0.275       0.216       0.154
F                2700903.714 1406538.330  754405.425  423311.488  521161.443
p                      0.000       0.000       0.000       0.000       0.000
Log-likelihood    -60137.791  -53996.269  -43339.818   37830.414  154124.270
Deviance           28298.689   27291.534   25628.285   15874.910    7992.720
AIC               120281.582  108000.539   86691.636  -75632.827 -308204.540
BIC               120313.783  108043.473   86756.037  -75482.557 -307968.400
N                 338946      338946      338946      338946      338946
===========================================================================
```

Predictions

After all this work, let's use our model to make a prediction. We need to exponentiate the result of our model. Since we took the log of price. Let's take a look at an example from Blue Nile. We'll use the full model m5, to predict the value of the diamond. Try to understand what this code does. If you see something you don't understand, look it up in the documentation, then answer the question. Evaluate how well the model predicts the Blue Nile diamond price. Think about the fitted point estimate. As well as, the lower and upper bound of the 95% confidence interval.

Quiz



~~~
## Predictions
Example Diamond from BlueNile:
Round 1.00 Very Good I VS1 $5,601
```{r}
#Be sure you've loaded the library memisc and have m5 saved as an
object in your workspace.
thisDiamond = data.frame(carat = 1.00, cut = "V.Good",
                         color = "I", clarity="VS1")
modelEstimate = predict(m5, newdata = thisDiamond,
                        interval="prediction", level = .95)
```



Evaluate how well the model predicts the BlueNile diamond's price.
Think about the fitted point estimate as well as the upper and lower
bound of the 95% CI.
~~~

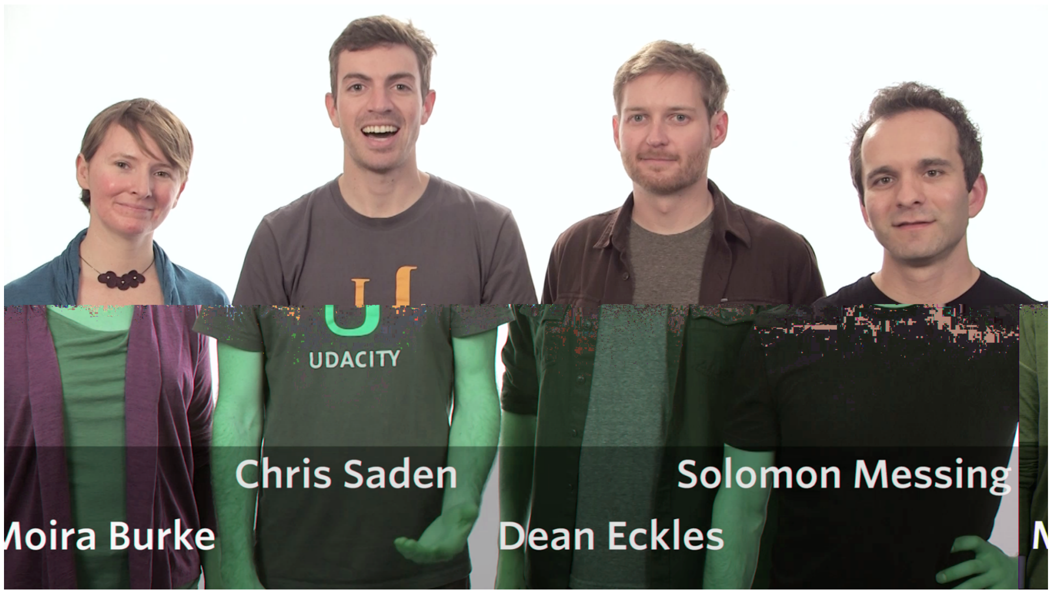
Answer

The results yield an expected value for price given the characteristics of our diamond, and the upper and lower bounds of a 95% confidence level. Note, because this is a linear model, predict is just multiplying each model coefficient by each value in our data. It turns out that this diamond is a touch pricier than the expected value under the full model, though it is by no means outside of the 95% confidence interval. Blue now has by most accounts a better reputation than diamond SE.info however. And reputation is worth a lot in a business that relies on easy to forge certificates in which the non-expert can be easily fooled. So, while this model might give you a sense of whether your diamond is a ripoff against diamondSE.info diamonds, it's not clear that diamondSE.info should be regarded as the universal source of truth over whether the price of a diamond is reasonable. Nonetheless, to have the expected price and diamondassay.info with a 95% interval, is a lot more information than we had about the price we should be willing to pay for a diamond before we started this exercise.

Final Thoughts

An important point. Even though we can predict the price of a diamond based on a function for c's. One thing you should not conclude with this exercise is that where you buy your diamond is irrelevant. You almost surely pay more for the same diamond at Tiffany's compared to Costco. Regardless you can use a model like this to get a sense of whether you were overpaying. One last thing, data and models are never infallible and you can still get taken even equipped with this kind of analysis. There's no substitute for establishing a personal connection and lasting business relationship with a jeweler you can trust.

Congratulations and Next Steps



Solomon: Congratulations on completing lesson six.

Moira: You've also reached the end of the course. We hope you feel prepared to explore data on your own, and answer questions that are interesting to you.

Dean: One next step is for you to find or collect some data, and start exploring.

Chris: Good luck in your future explorations, and thanks for joining us.