Homework 3
MSBA 400: Statistical Foundations for Data Analytics
Professor Rossi

Question 1 : Prediction from Multiple Regressions

Q1, part A

Run the multiple regression of `Sales` on `p1` and `p2` using the dataset, `multi`.

Soluton:

```
library(DataAnalytics)
data(multi)
summary(lm(formula = Sales ~ p1 + p2, data = multi))
```

```
##
## Call:
## lm(formula = Sales ~ p1 + p2, data = multi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.916 -15.663  -0.509  18.904  63.302
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  115.717      8.548   13.54   <2e-16 ***
## p1           -97.657      2.669  -36.59   <2e-16 ***
## p2           108.800      1.409   77.20   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 28.42 on 97 degrees of freedom
## Multiple R-squared:  0.9871, Adjusted R-squared:  0.9869
## F-statistic:  3717 on 2 and 97 DF,  p-value: < 2.2e-16
```

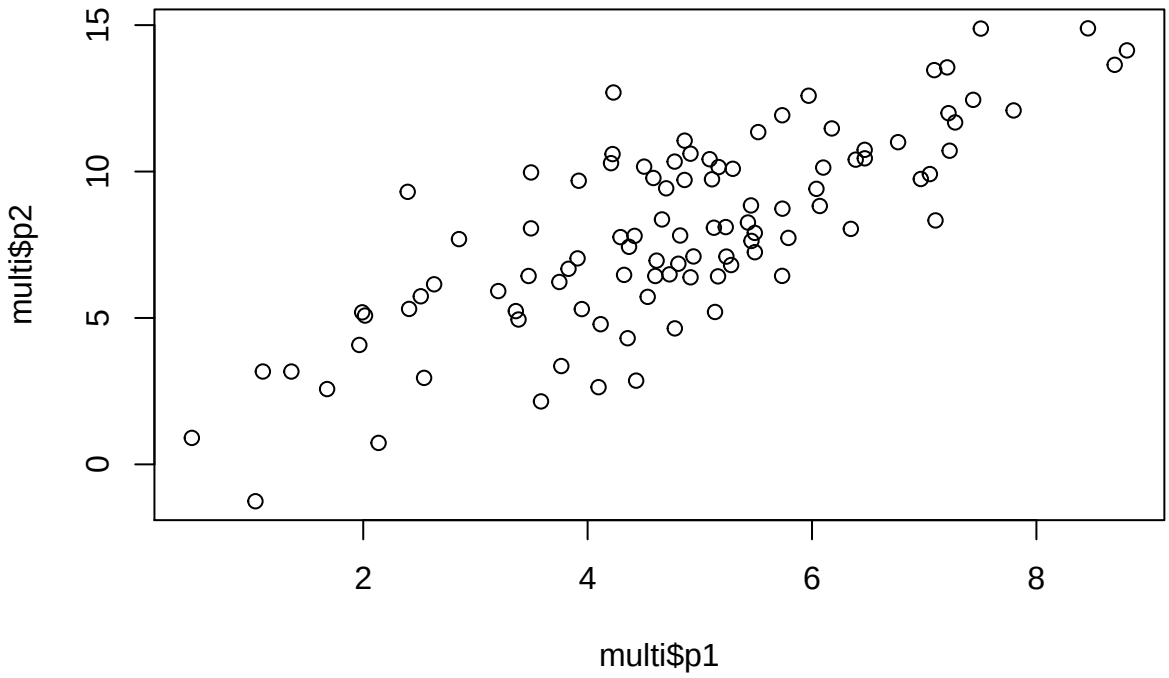
Q1, part B

Suppose we wish to use the regression from part A to estimate sales of this firm's product with, `p1` = \$7.5. To make predictions from the multiple regression, we will have to predict what p2 will be given that `p1` =\$7.5.

Explain why setting `p2 = mean(p2)` would be a bad choice. Be specific and comment on why this is true for this particular case (value of `p1`).

Solution: For this case, we begin by examining the relationship between `p1` and `p2`:

```
plot(multi$p1, multi$p2)
```




```
print(mean(multi$p1))
```

```
## [1] 4.802319
```

```
print(mean(multi$p2))
```

```
## [1] 8
```

Based on the above information, we can see that setting `p2 = mean(p2)` is a bad choice because, when `p1` = \$7.5, the actual corresponding value of `p2` should be around \$13. The mean of `p2`, however, is \$8. In general, setting `p2 = mean(p2)` is a bad choice because it does not take into account the fact that `p2` (competing product price) and `p1` (our product price) are inherently related. That is, setting `p2 = mean(p2)` gives us the expected value of `p2` but, to account for the relationship between the two quantities, we want the expected value of `p2` given `p1`.

Q1, part C

Use a regression of `p2` on `p1` to predict what `p2` would be given that `p1` = \$7.5.

Solution:

```
outlm = lm(p2 ~ p1, data = multi)
p1_star = data.frame(p1 = c(7.5))
p2_pred = predict(outlm, newdata = p1_star)
p2_pred
```

```
##        1
## 12.00116
```

So, if p1 = $7.5, then we predict that p2 = $12.

Q1, part D

Use the predicted value of p2 from part C, to predict Sales. Show that this is the same predicted value of sales as you would get from the simple regression of Sales on p1. Explain why this must be true.

Solution:

```
# regression from part A
outlm = lm(formula = Sales ~ p1 + p2, data = multi)
p1_star = 7.5
# fill in value from above
p2_star = p2_pred
p1_p2 = data.frame(p1 = p1_star, p2 = p2_star)
# predicting sales with p1 and p2
predict(outlm, newdata = p1_p2)
```

```
##        1
## 689.0118
```

```
outlm = lm(Sales ~ p1, data = multi)
p1_star = data.frame(p1 = c(7.5))
predict(outlm, newdata = p1_star)
```

```
##        1
## 689.0118
```

So, we can see that, using the predicted value of p2 from part C to predict Sales, we get the same predicted value as one would get from the simple regression of Sales on p1. This happens because, when we run the regression of p2 on p1 and predict p2, the only variation in p2 we are able to capture is that which is explained by p1. Thus, when running the multiple regression, the variation of the predicted p2 value is accounted for completely by p1, causing the resulting values of the multiple regression and the simple regression to be the same.

Question 2: Interactions

An interaction term in a regression is formed by taking the product of two independent or predictor variables as in:

$$Y_i = \beta_0 + \beta_1 X1_i + \beta_2 X2_i + \beta_3 X1_i * X2_i + \varepsilon_i$$

This term has a non-linear effect, which allows the effect of variable $X1$ to be moderated by the level of $X2$. We can take the partial derivative of the conditional mean function to see this:

$$\frac{\partial}{\partial X1} E[Y|X1, X2] = \beta_1 + \beta_3 X2$$

Return to the regression in Chapter 6 of log(emv) on luxury, sporty and add the interaction term luxury*sporty.

Q2, part A

Compute the change in emv we would expect to see if sporty increased by .1 units, holding luxury constant at .30 units.

Solution:

We have the regression equation as

$$log(emv) = \beta_0 + \beta_1 l + \beta_2 s + \beta_3 ls + \varepsilon$$

where $l =$ `luxury` and $s =$ `sporty`. Taking the partial derivative of the conditional mean function with respect to `sporty`, we get

$$\frac{\partial}{\partial s} E[log(emv)|l, s] = \beta_2 + \beta_3 l,$$

which we can use to calculate changes in `emv` that result from changes in `sporty` given some constant value of `luxury`.

```
# running the regression
data(mvehicles)
cars = mvehicles[mvehicles$bodytype != 'Truck',]
outslr = lm(log(emv)~luxury + sporty + luxury*sporty, data = cars)
lmSumm(outslr)
```

```
## Multiple Regression Analysis:
##     4 regressors(including intercept) and 1395 observations
##
## lm(formula = log(emv) ~ luxury + sporty + luxury * sporty, data = cars)
##
## Coefficients:
##                Estimate Std Error t value p value
## (Intercept)     9.7350   0.04385  222.02       0
## luxury          1.3220   0.10900   12.12       0
## sporty         -0.4096   0.11600   -3.53       0
## luxury:sporty   1.2930   0.22210    5.82       0
## ---
## Standard Error of the Regression:  0.3122
## Multiple R-squared:  0.588  Adjusted R-squared:  0.587
## Overall F stat: 662.49 on 3 and 1391 DF, pvalue= 0
```

```
# intercept
beta_0 = outslr$coef[1]
# coefficient on luxury
beta_1 = outslr$coef[2]
# coefficient on sporty
beta_2 = outslr$coef[3]
# coefficient on luxury * sporty
beta_3 = outslr$coef[4]
# compute change in log(emv)
l = 0.3
d_sporty_a = 0.1
d_log_emv_a = d_sporty_a * (beta_2 + beta_3 * l)
# convert from change in log to change in true value
d_emv_a = exp(d_log_emv_a)
d_emv_a
```

```
##    sporty
## 0.9978489
```

So, with each increase in `sporty` of 0.1 units when `luxury` $= 0.3$, we expect `emv` to stay relatively constant (i.e. decrease to 99.78% of its initial value).

Q2, part B

Compute the change in `emv` we would expect to see if sporty was increased by .1 units, holding luxury constant at .70 units.

Solution:

```
# compute change in log(emv)
l = 0.7
d_sporty_b = 0.1
d_log_emv_b = d_sporty_b * (beta_2 + beta_3 * l)
# convert from change in log to change in true value
d_emv_b = exp(d_log_emv_b)
d_emv_b
```

```
##   sporty
## 1.050834
```

So, with each increase in `sporty` of 0.1 units when `luxury` $= 0.7$, we expect `emv` to increase slightly (i.e. increase to 105.08% of its initial value).

Q2, part C

Why are the answers different in part A and part B? Does the interaction term make intuitive sense to you? Why?

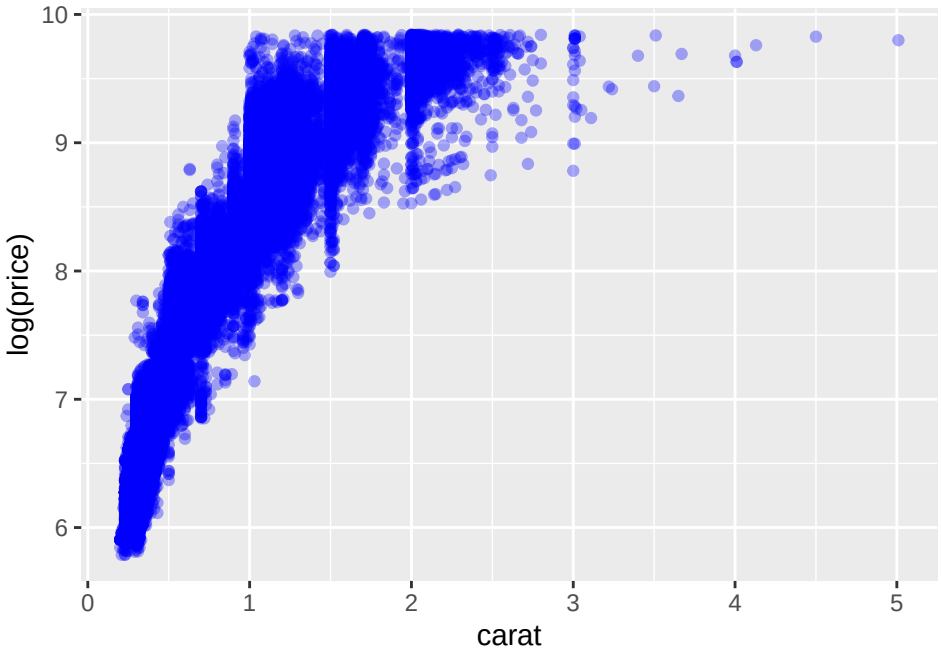
Solution: The answers in part A and part B are different because the value of `luxury` is different in each case. This interaction term does make sense to me as it implies that, if a car is more luxurious, then a certain increase in the sportiness of the car will make its price increase as long as the luxury of the car is sufficiently high. We saw a slight decrease in price with an increase in sportiness when the luxury of the car was low, and this is likely because there is not much of a relationship between luxury and sportiness for lower-end cars.

Question 3: More on ggplot2 and regression planes

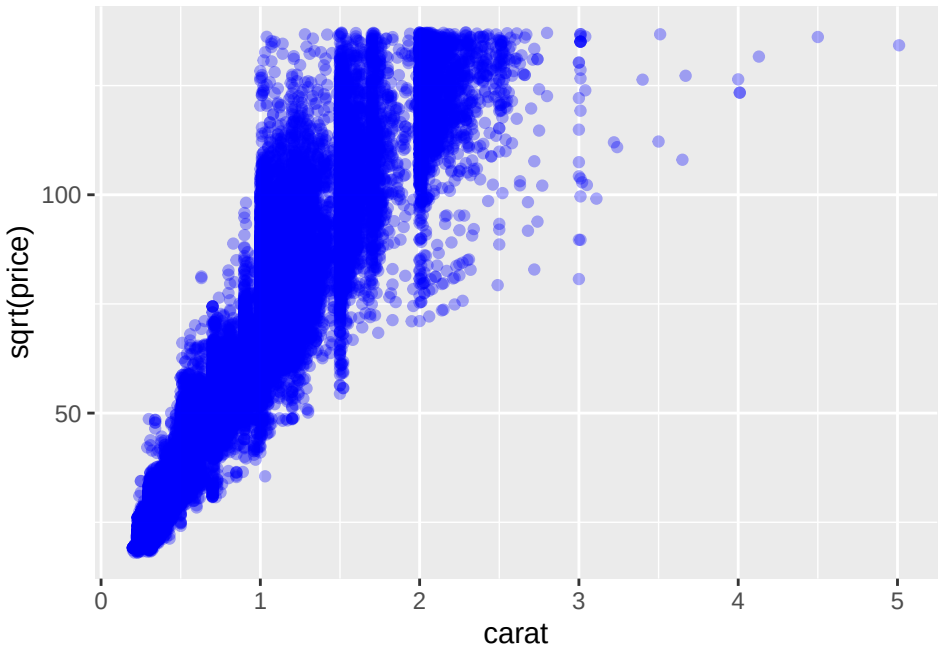
The classic dataset, `diamonds`, (you must load the `ggplot2` package to access this data) has about 50,000 prices of diamonds along with weight (`carat`) and quality of cut (`cut`).

1. Use ggplot2 to visualize the relationship between price and carat and cut. 'price' is the dependent variable. Consider both the log() and sqrt() transformation of price.

```
library(ggplot2)
data(diamonds)
# plotting carat vs. log(price)
qplot(carat, log(price), data = diamonds, col = I("blue"), alpha=I(1/3))
```

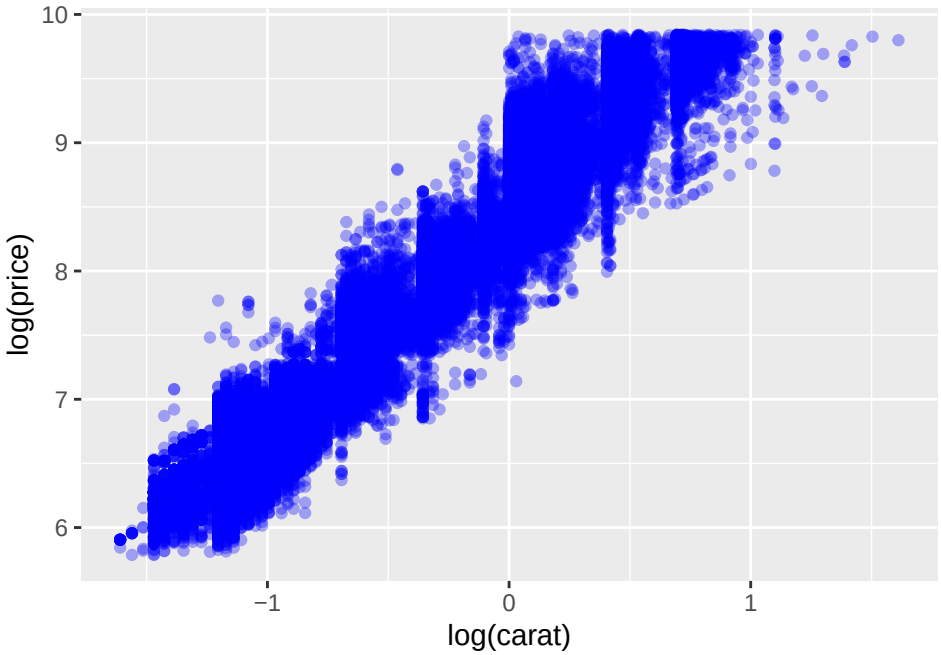



```
# plotting carat vs. sqrt(price)
qplot(carat, sqrt(price), data = diamonds, col = I("blue"), alpha=I(1/3))
```



The `sqrt()` transformation seems to yield a more linear relationship between `carat` and `price`.

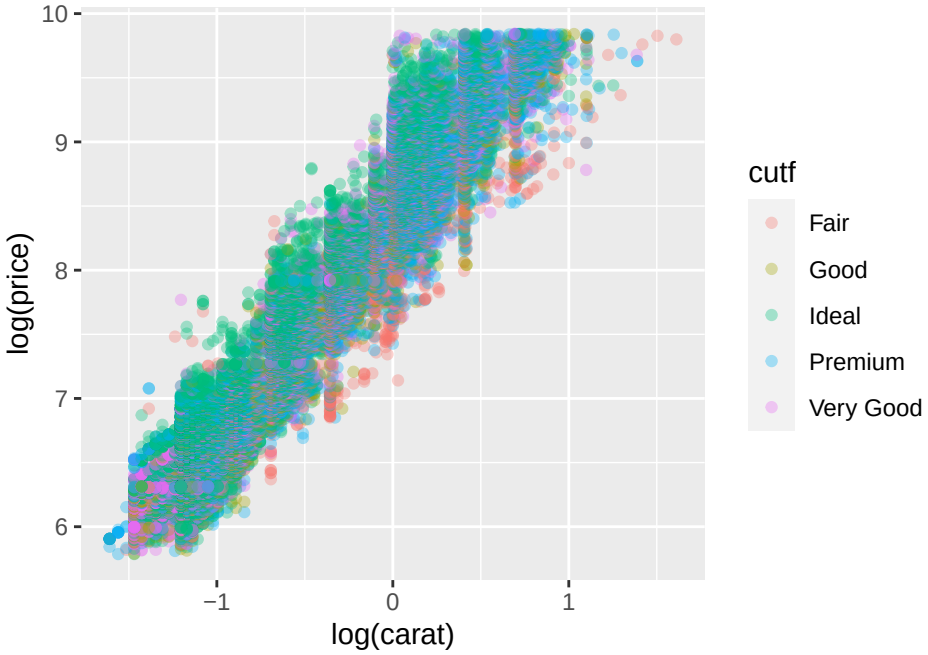
```
# plotting log(carat) vs. log(price)
qplot(log(carat), log(price), data = diamonds, col = I("blue"), alpha=I(1/3))
```



However, we see that there is very linear relationship between `log(carat)` and `log(price)`, so these are the transformations that we will use in our regression.

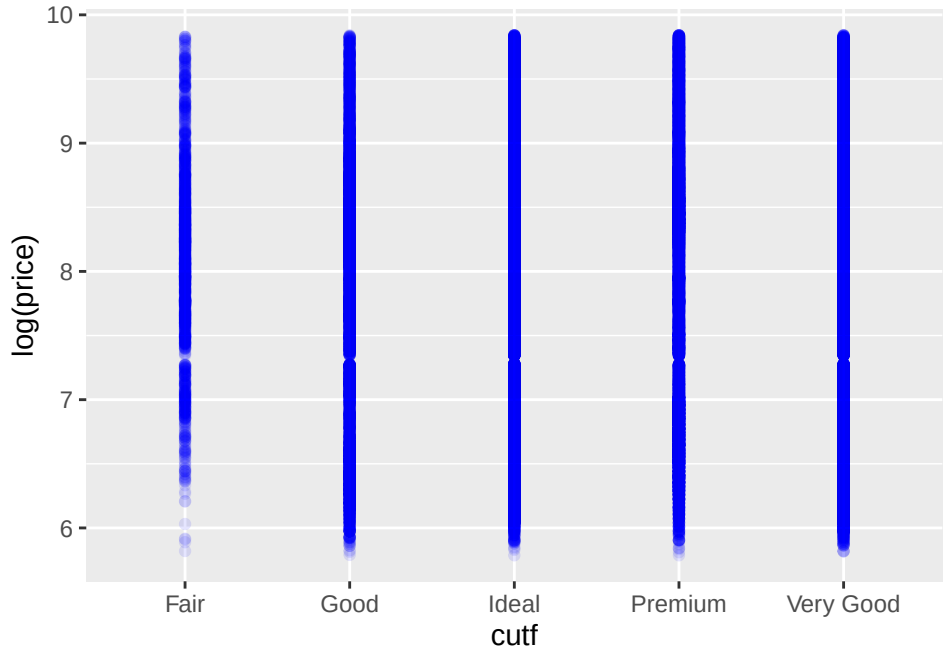
Next, let's investigate a how `cut` tends to vary with `log(carat)` and `log(price)`:

```
cutf = as.character(diamonds$cut)
cutf = as.factor(cutf)
qplot(log(carat), log(price), data = diamonds, col = cutf, alpha=I(1/3))
```

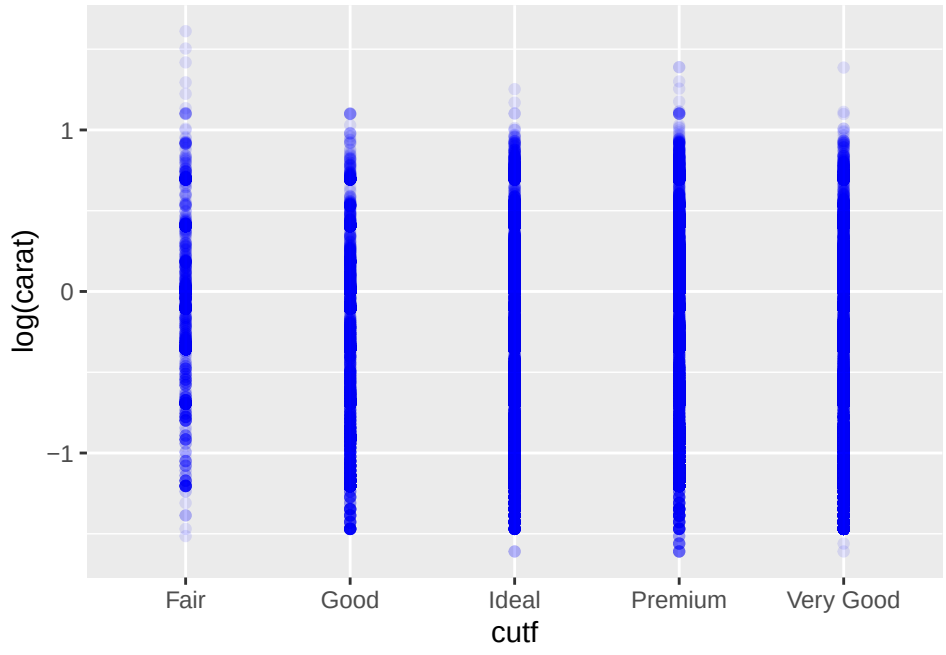


There doesn't appear to be much of a relationship between `cut`, `log(carat)`, and `log(price)`. Let's confirm this with individual bivariate plots:

```
# plotting cut vs. log(price)
qplot(cutf, log(price), data = diamonds, col = I("blue"), alpha=I(1/10))
```




```
# plotting cut vs. carat
qplot(cutf, log(carat), data = diamonds, col = I("blue"), alpha=I(1/10))
```



Based on the above plots, we can confirm that there is not much of a relationship between `cut` and `log(carat)` and `log(price)`.

2. Run a regression of your preferred specification. Perform residual diagnostics. What do you conclude from your regression diagnostic plots of residuals vs. fitted and residuals vs. carat?

note: `cut` is a special type of variable called an ordered factor in R. For ease of interpretation, convert the ordered factor into a "regular" or non-ordinal factor.

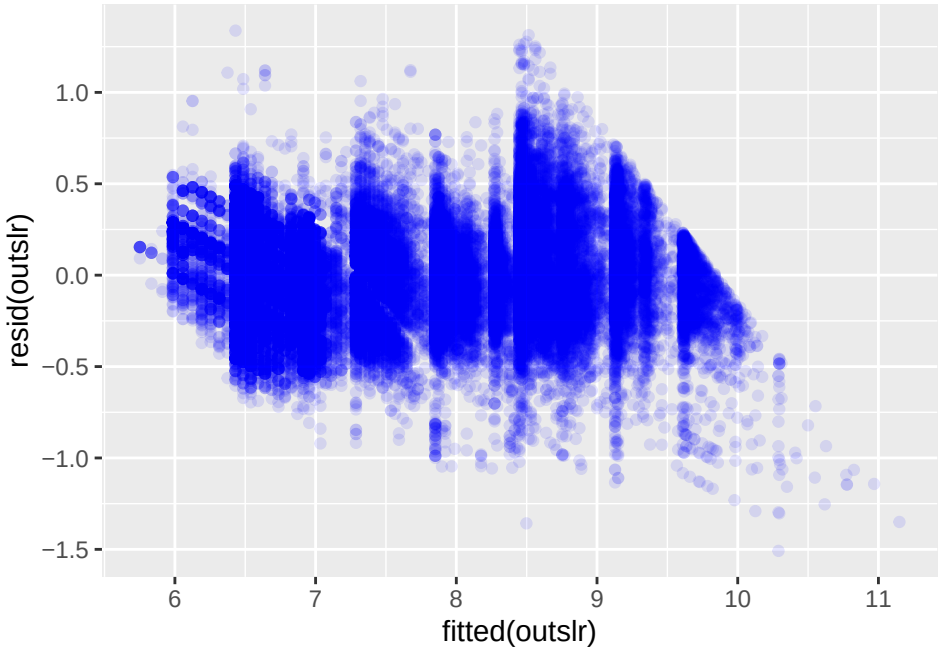
```
library(ggplot2)
data(diamonds)
cutf=as.character(diamonds$cut)
cutf=as.factor(cutf)
```

Solution: Based on our exploratory analysis above in which we identified a linear relationship between `log(price)` and `log(carat)` and not much of a relationship between `cut` and these variables, we will use a regression with just `log(price)` and `log(carat)`,

```
# running a regression of log(price) on log(carat)
outslr = lm(log(price) ~ log(carat), data = diamonds)
summary(outslr)
```

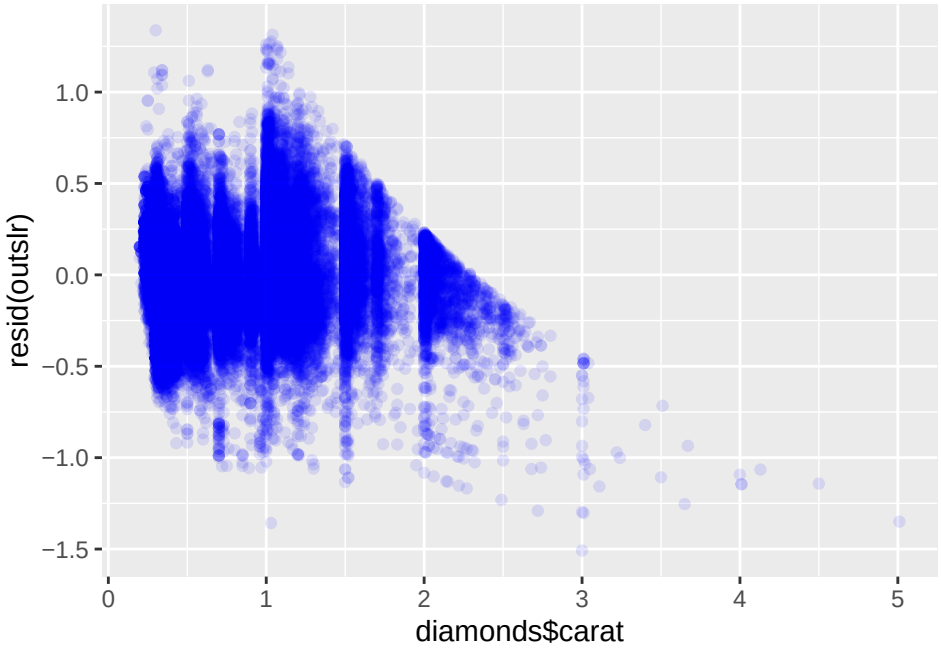
```
##
## Call:
## lm(formula = log(price) ~ log(carat), data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50833 -0.16951 -0.00591  0.16637  1.33793
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.448661   0.001365  6190.9   <2e-16 ***
## log(carat)  1.675817   0.001934   866.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2627 on 53938 degrees of freedom
## Multiple R-squared:  0.933,  Adjusted R-squared:  0.933
## F-statistic: 7.51e+05 on 1 and 53938 DF,  p-value: < 2.2e-16
```

```
# plot fitted values vs. residuals
qplot(fitted(outslr), resid(outslr), col = I("blue"), alpha=I(1/10))
```



We can see from the plot above that the majority of the fitted values seem to have corresponding residuals with a mean of zero. The linearity assumption thus appears to be valid. Additionally, we can see that most of the fitted values exhibit a spread of residuals that is roughly the same. This indicates that the the constant variance of the residuals (i.e. homoscedasticity) is upheld.

```
# plot carat vs. residuals
qplot(diamonds$carat, resid(outslr), col = I("blue"), alpha=I(1/10))
```



While it could be argued that there is a slight negative correlation between `carat` and the residuals, the majority of the points on the plot say otherwise. This further upholds the validity of the regression.