Multiple Regression and interaction terms

MSBA 400: Statistical Foundations for Data Analytics
Professor Rossi
Author: Sai Akhil Siruguppa
ID number: 205635364

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

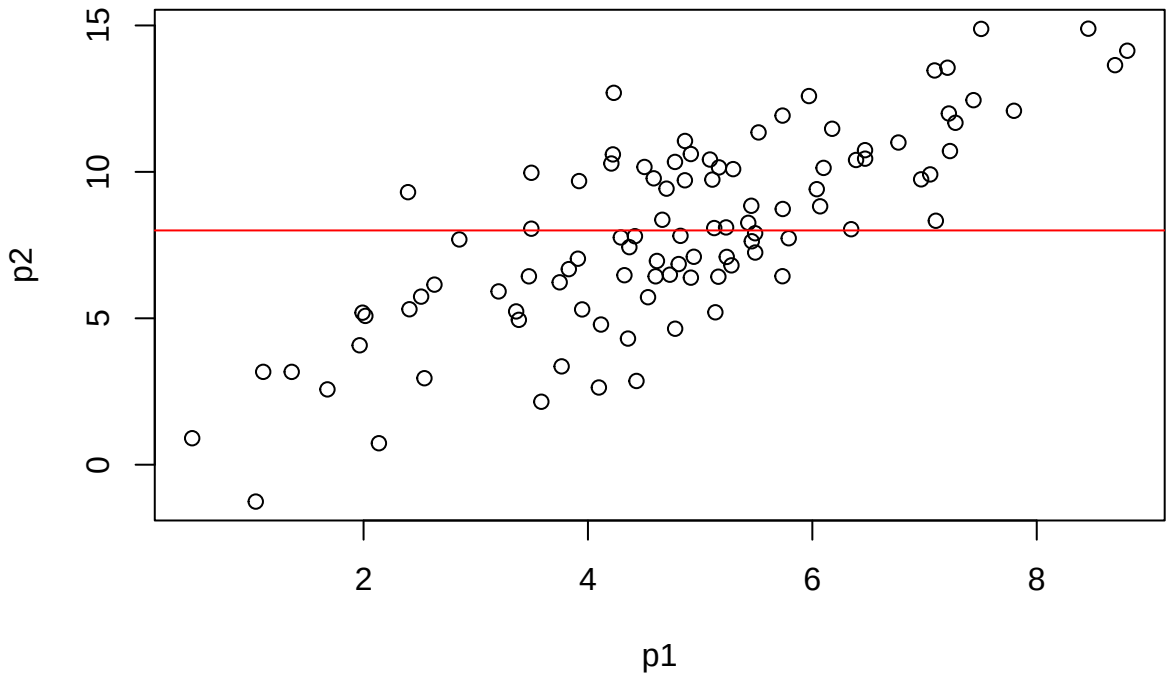
```
library("DataAnalytics")
data("multi")
Sales <- multi$Sales
p1 <- multi$p1
p2 <- multi$p2
outlm <- lm(Sales~p1+p2)
summary(outlm)
```

```
##
## Call:
## lm(formula = Sales ~ p1 + p2)
##
## 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
```

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`).

```
plot(p1, p2, xlab = "p1", ylab = "p2")
abline(c(mean(p2), 0), col="red")
```



By plotting p1 and p2 we can see the correlation between them and thus can see that the value of p2 changes with the value of p1. We can also see the red line in the plot indicating the mean of p2 for all values of p1. We can clearly observe that when the p1 is 7.5, considering the value of p2 to be the mean of p2 is not appropriate.

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

```
outlm_prices <- lm(p2~p1)
summary(outlm_prices)
```

```
##
## Call:
## lm(formula = p2 ~ p1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5921 -1.3602  0.0299  1.3851  5.5472
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.8773     0.6062   1.447    0.151
## p1            1.4832     0.1189  12.475   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.037 on 98 degrees of freedom
## Multiple R-squared:  0.6136, Adjusted R-squared:  0.6097
## F-statistic: 155.6 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
p <-  as.data.frame(7.5)
colnames(p) <- "p1"
pred_p2 = predict(outlm_prices, newdata=p)
```

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.

```
df <- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("p1", "p2")
colnames(df) <- x
df[nrow(df) + 1,] = list(7.5, pred_p2)
predict(outlm, newdata=df)
```

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

```
outlm_sales_p1 <- lm(Sales~p1)
predict(outlm_sales_p1, newdata=p)
```

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

We can see that the predicted value is the same in both cases. Multiple regression tries to estimate the partial or pure effect of p1 on Sales controlling for co-variation in p2. However, when we use `p1` to predict `p2`, we assume that `p2` is perfectly correlated with `p1`. Mathematically we can see this:

We have a multiple regression of `Sales` on `p1` and `p2` as follows:

$$Sales_i = \beta_0 + \beta_1 p1_i + \beta_2 p2_i + \varepsilon_i$$

Simple regression of `p2` on `p1` to predict p2:

$$p2_i = \beta_0' + \beta_1' p1_i + \varepsilon_i'$$

We finally can rewrite the multiple regression to predict `Sales` as:

$$Sales_i = \beta_0 + \beta_1 p1_i + \beta_2 \cdot (\beta_0' + \beta_1' p1_i + \varepsilon_i') + \varepsilon_i$$

$$Sales_i = (\beta_0 + \beta_2 \cdot \beta_0') + (\beta_1 + \beta_2 \cdot \beta_1') p1_i + (\beta_2 \varepsilon_i' + \varepsilon_i)$$

which is the same as simple regression of `Sales` on `p1`:

$$Sales_i = \beta_0^* + \beta_1^* p1_i + \varepsilon_i^*$$

nteractions

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`.

$$\log(emv) = \beta_0 + \beta_1 \cdot luxury + \beta_2 \cdot sporty + \beta_3 \cdot luxury * sporty + \varepsilon_i$$

```
data("mvehicles")
cars=mvehicles[mvehicles$bodytype != "Truck",]
outs_lr=lm(log(emv)~luxury+sporty+(luxury*sporty),data=cars)
summary(outs_lr)
```

```
##
## Call:
## lm(formula = log(emv) ~ luxury + sporty + (luxury * sporty),
##     data = cars)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.77690 -0.20474 -0.03719  0.19434  2.50271
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.73506    0.04385 222.016  < 2e-16 ***
## luxury         1.32184    0.10904  12.122  < 2e-16 ***
## sporty        -0.40956    0.11601  -3.530 0.000429 ***
## luxury:sporty  1.29343    0.22206   5.825  7.1e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3122 on 1391 degrees of freedom
## Multiple R-squared:  0.5883, Adjusted R-squared:  0.5874
## F-statistic: 662.5 on 3 and 1391 DF,  p-value: < 2.2e-16
```

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

$$\frac{\partial}{\partial sporty} E[\log emv | luxury, sporty] = b_1 + b_3 \cdot luxury$$

```
b_1 = outs_lr$coef[3]
b_3 = outs_lr$coef[4]
change_in_sporty = 0.1
luxury = 0.3
change_in_emv = (b_1 + (b_3*luxury))*change_in_sporty
cat(exp(change_in_emv))
```

```
## 0.9978489
```

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

```
b_1 = outs_lr$coef[3]
b_3 = outs_lr$coef[4]
change_in_sporty = 0.1
luxury = 0.7
change_in_emv = (b_1 + (b_3*luxury))*change_in_sporty
cat(exp(change_in_emv))
```

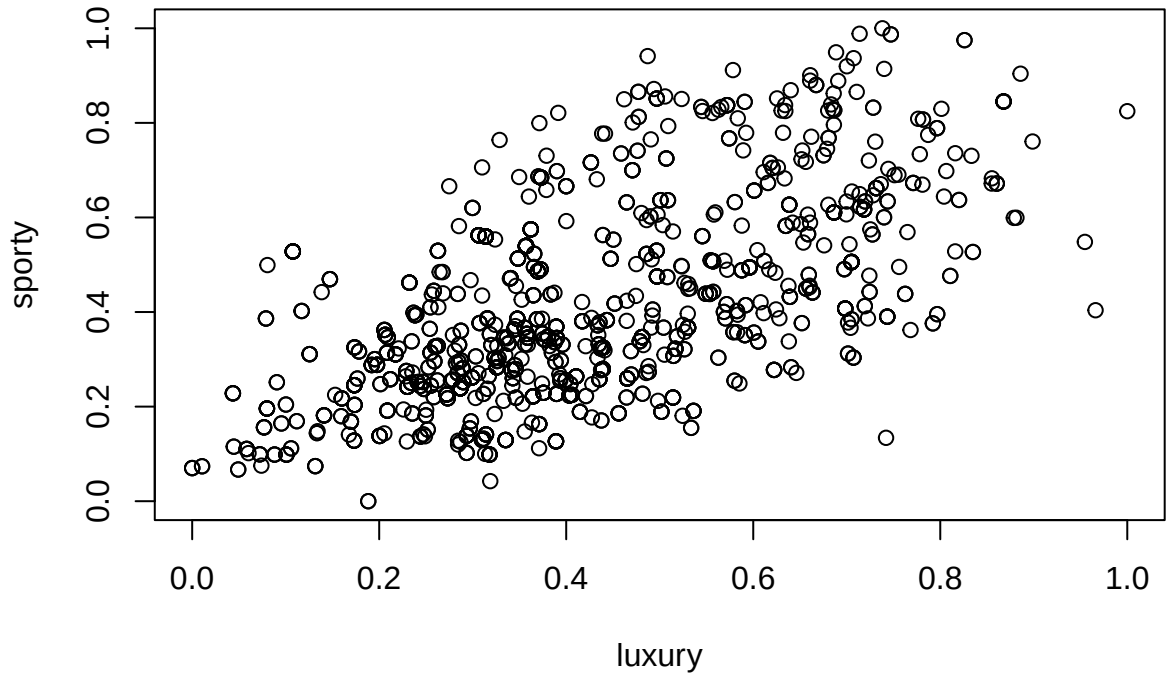
```
## 1.050834
```

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

```
corr(cbind(cars$luxury, cars$sporty))
```

```
## Full Correlation Matrix
##      [,1] [,2]
## [1,]  1.0  0.6
## [2,]  0.6  1.0
##
## Correlation Matrix trimmed with cutoff = 0.25
##      [,1]  [,2]
## [1,]  1.00  0.6
## [2,]  0.6   1.00
```

```
plot(cars$luxury, cars$sporty, xlab = "luxury", ylab = "sporty")
```



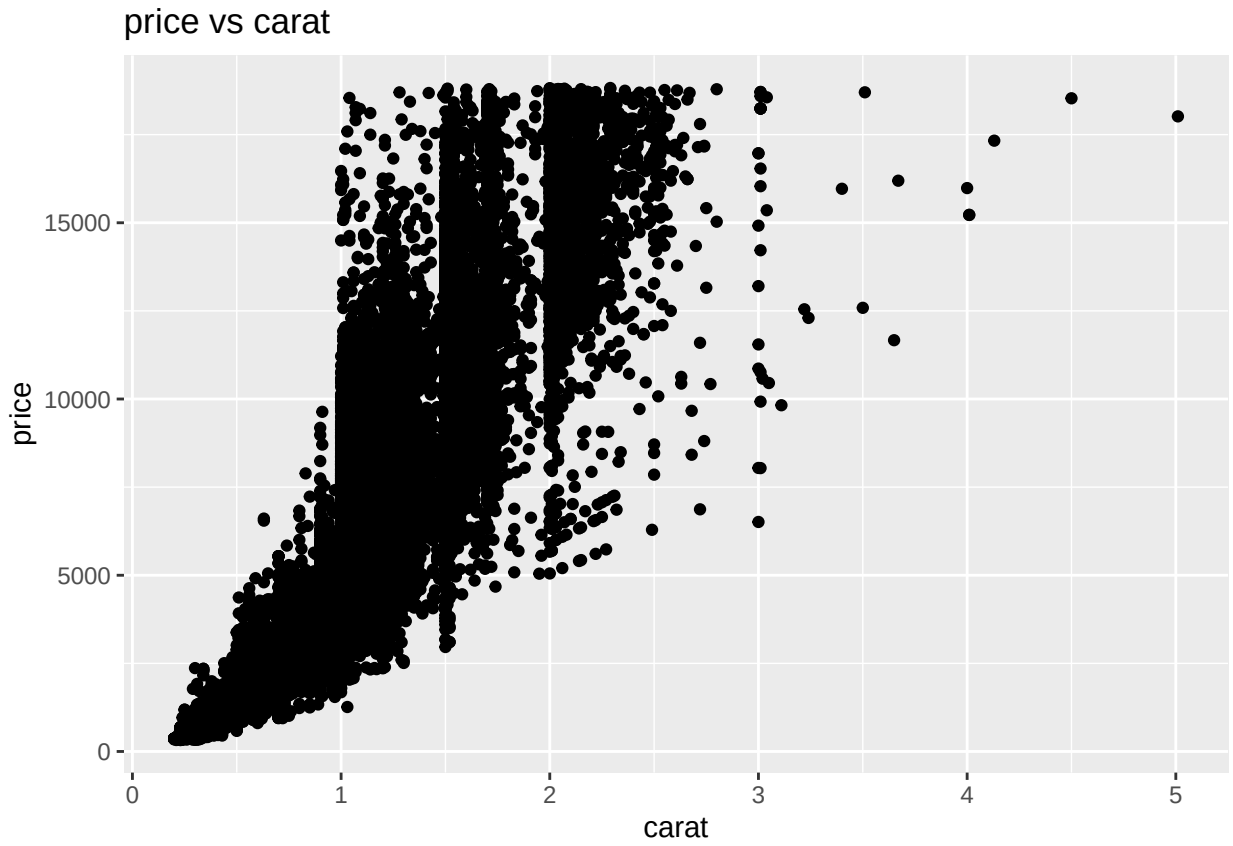
We can see from the above graph, that there exists some correlation between `luxury` and `sporty` attributes while predicting the `emv`. Intuitively the interaction term allows to calculate the change in emv with respect to change in one of the variables. In our case, the change of `log(emv)` with respect to change in `sporty` or price is linearly dependent on the `luxury` and thus the interaction term helps to achieve this relationship.

Regression planes

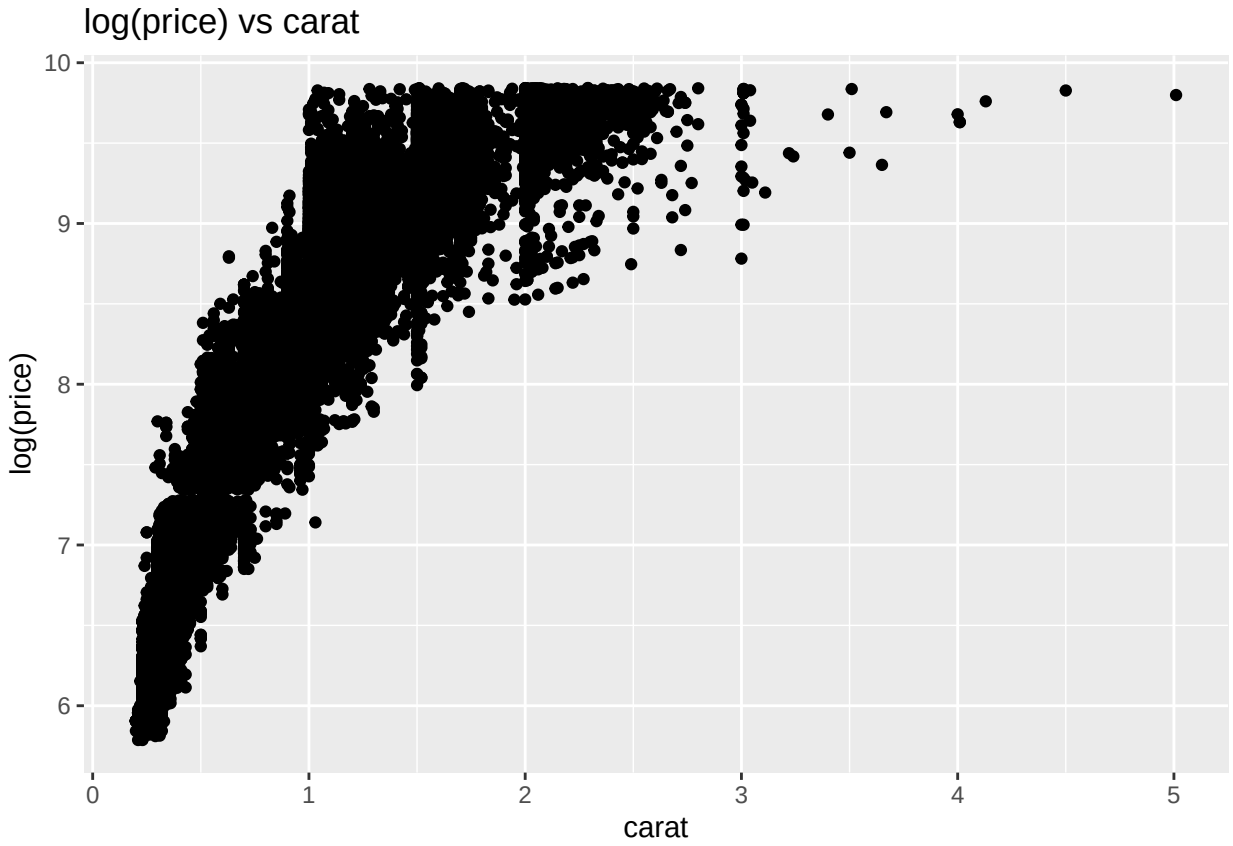
The classic dataset, `diamonds`, (you must load the `ggplot2` package to access this data) has about 50,000 prices of diamonds along with weight (`carat`) and quality of cut (`cut`).

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.

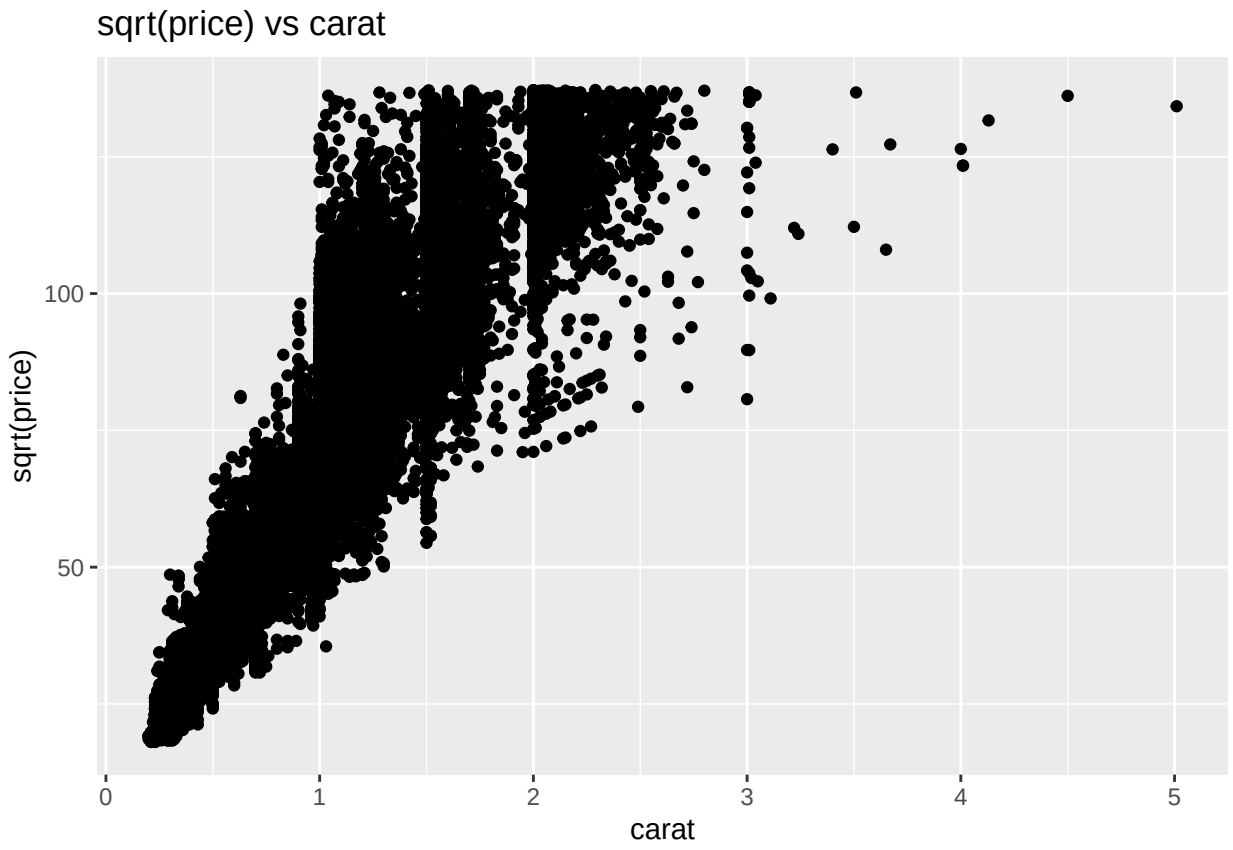
```
library(ggplot2)
data(diamonds)
ggplot(data = diamonds, mapping = aes(x=carat, y = price)) + ggtitle("price vs carat") + geom_point()
```



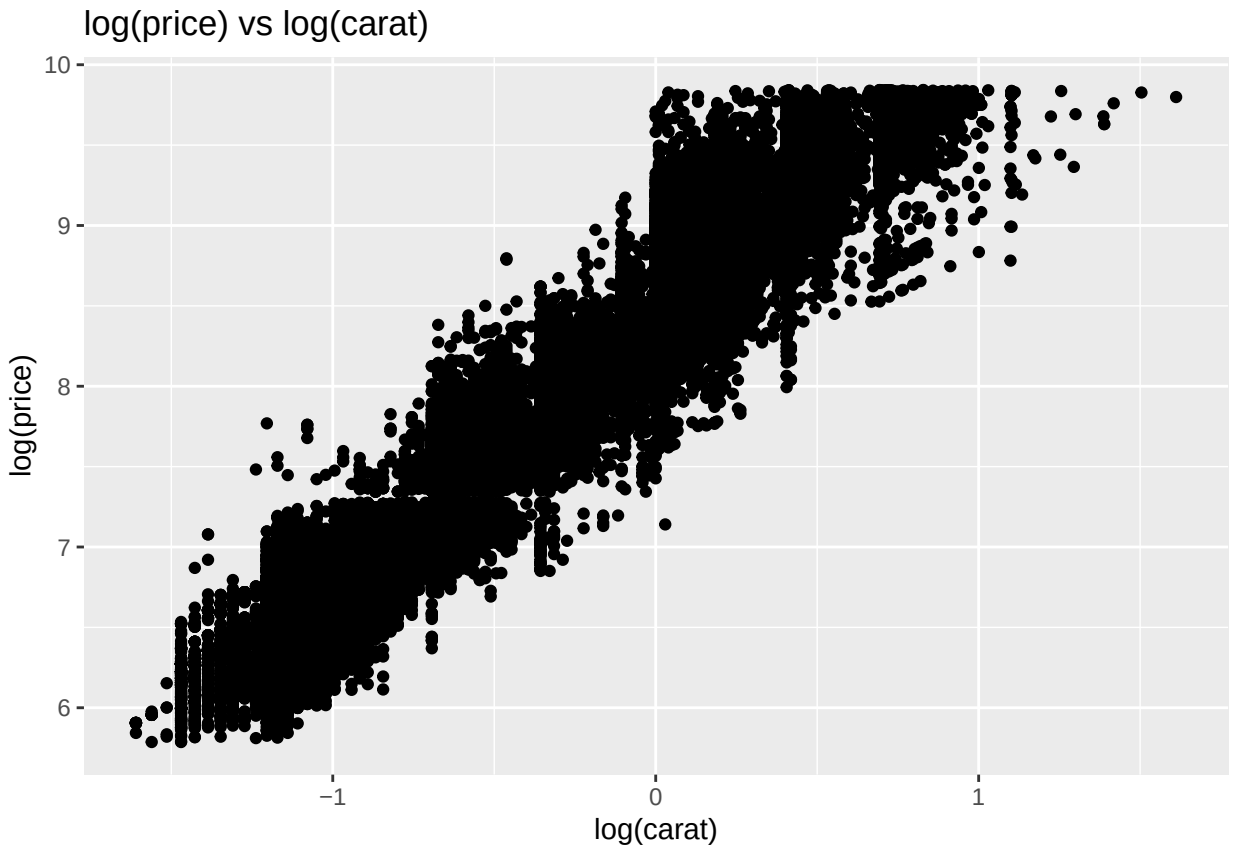

```
ggplot(data = diamonds, mapping = aes(x=carat, y = log(price))) + ggtitle("log(price) vs carat") + geom_
```




```
ggplot(data = diamonds, mapping = aes(x=carat, y = sqrt(price))) + ggtitle("sqrt(price) vs carat") + ge
```

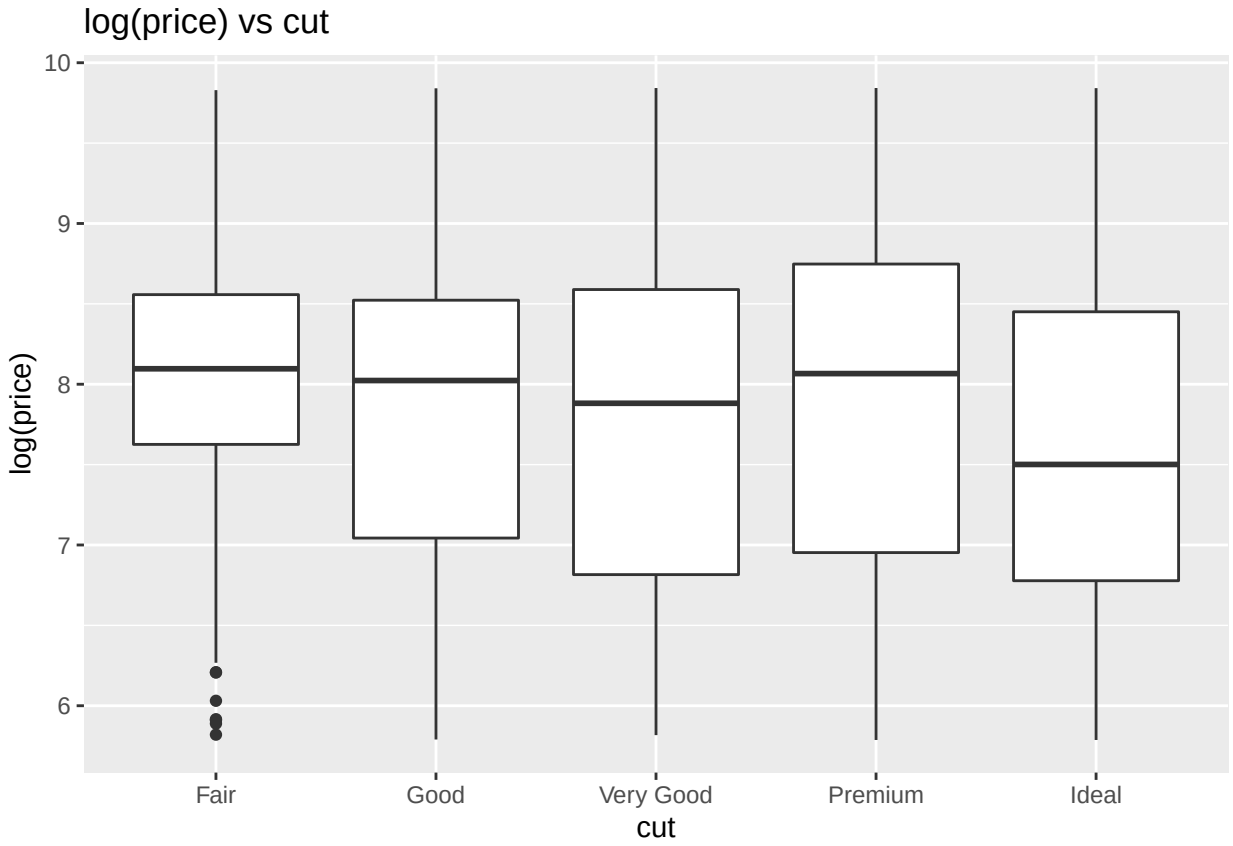



```
ggplot(data = diamonds, mapping = aes(x=log(carat), y = log(price))) + ggtitle("log(price) vs log(carat)
```



Hence, we can see a linear relationship between `log(price)` and `log(carat)`

```
ggplot(data = diamonds, mapping = aes(x=cut, y = log(price))) + ggtitle("log(price) vs cut") + geom_boxp
```



Hence, we can see no clear relationship between `log(price)` and `carat`. As the factors of `carat` change, we see no significant pattern in the `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.

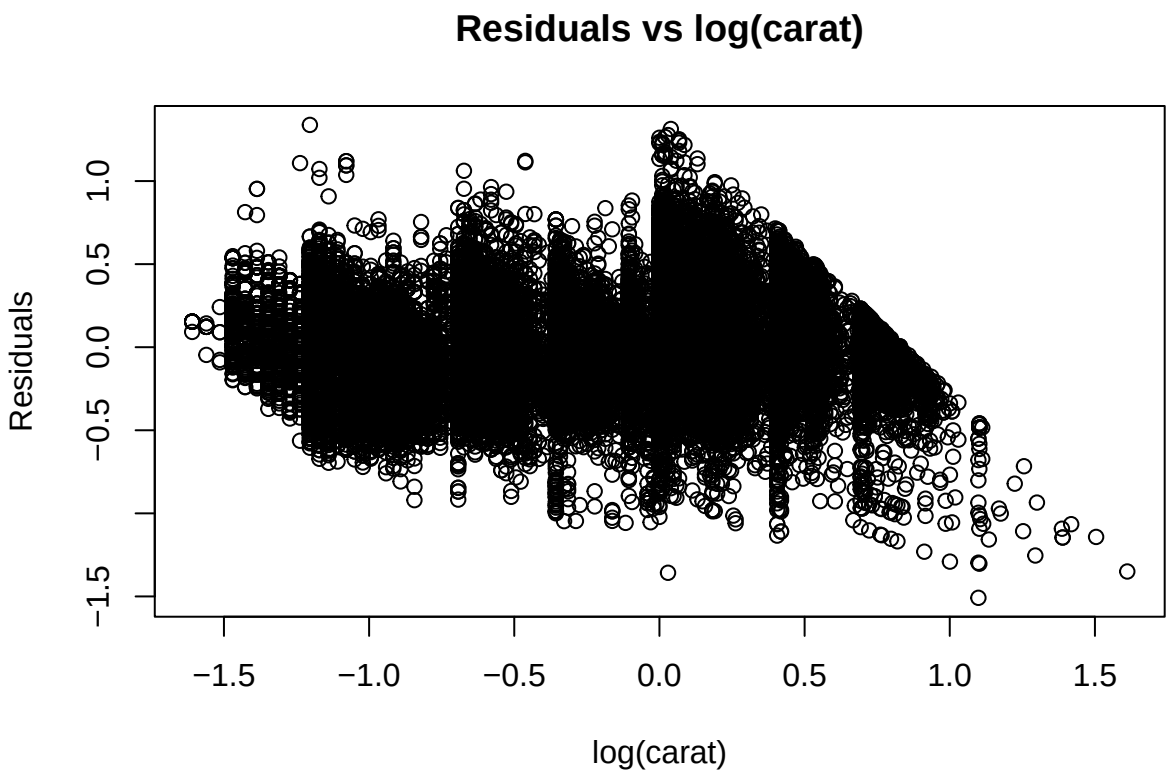
From the above data visualizations and analysis we can see that the `log(price)` is linearly dependent on `log(carat)`. Also, we can see from all the plots that the `price` does not vary much with respect to different levels of `cut`. Hence for the final linear model, I have chosen to predict `log(price)` by regressing it on `log(carat)`

```
cutf=as.character(diamonds$cut)
cutf=as.factor(cutf)
outlm_diamonds=lm(log(price)~log(carat), data=diamonds)
summary(outlm_diamonds)
```

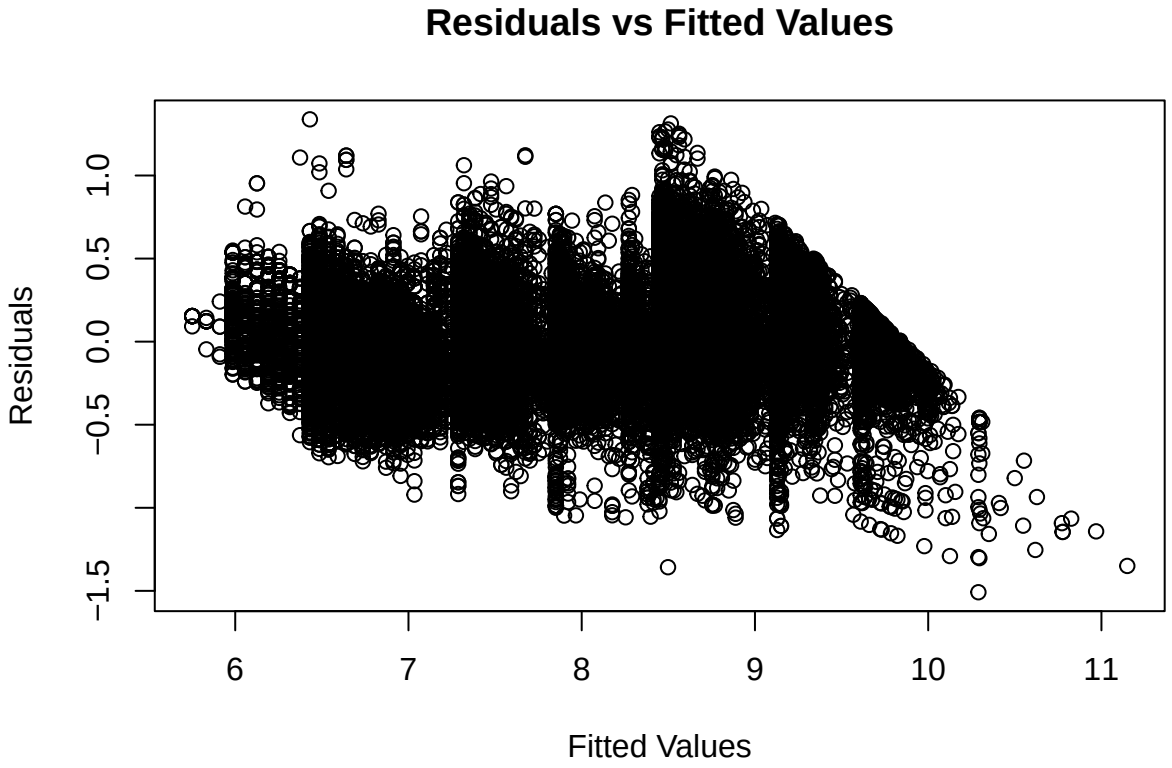
```
##
## Call:
## lm(formula = log(price) ~ log(carat), data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50833 -0.16951 -0.00591  0.16637  1.33793
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (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(log(diamonds$carat), outlm_diamonds$residuals, xlab = "log(carat)", ylab = "Residuals", main = "Re
```




```
plot(outlm_diamonds$fitted.values, outlm_diamonds$residuals, xlab = "Fitted Values", ylab = "Residuals"
```



After plotting `residuals` against `log(carat)` and `residuals` against the fitted values, we can see that there is no clear pattern or relation between them from the above graphs. Since the fitted values represent all the prices predicted by regressing the independent variable of `log(carat)` we can see that there is no pattern to the `residuals`. Hence, we can conclude that we have a "good" fit.