Question 1 : Prediction from Multiple Regressions

Q1, part A

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

Answer Q1, Part A:

```
# Loading required libraries and dataset
library("DataAnalytics")
data("multi")

# Multiple Linear Regression (Sales ~ p1 + p2)
multi_lm = lm(formula = Sales ~ p1 + p2, data = multi)

summary(multi_lm)
```

```
##
## 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.

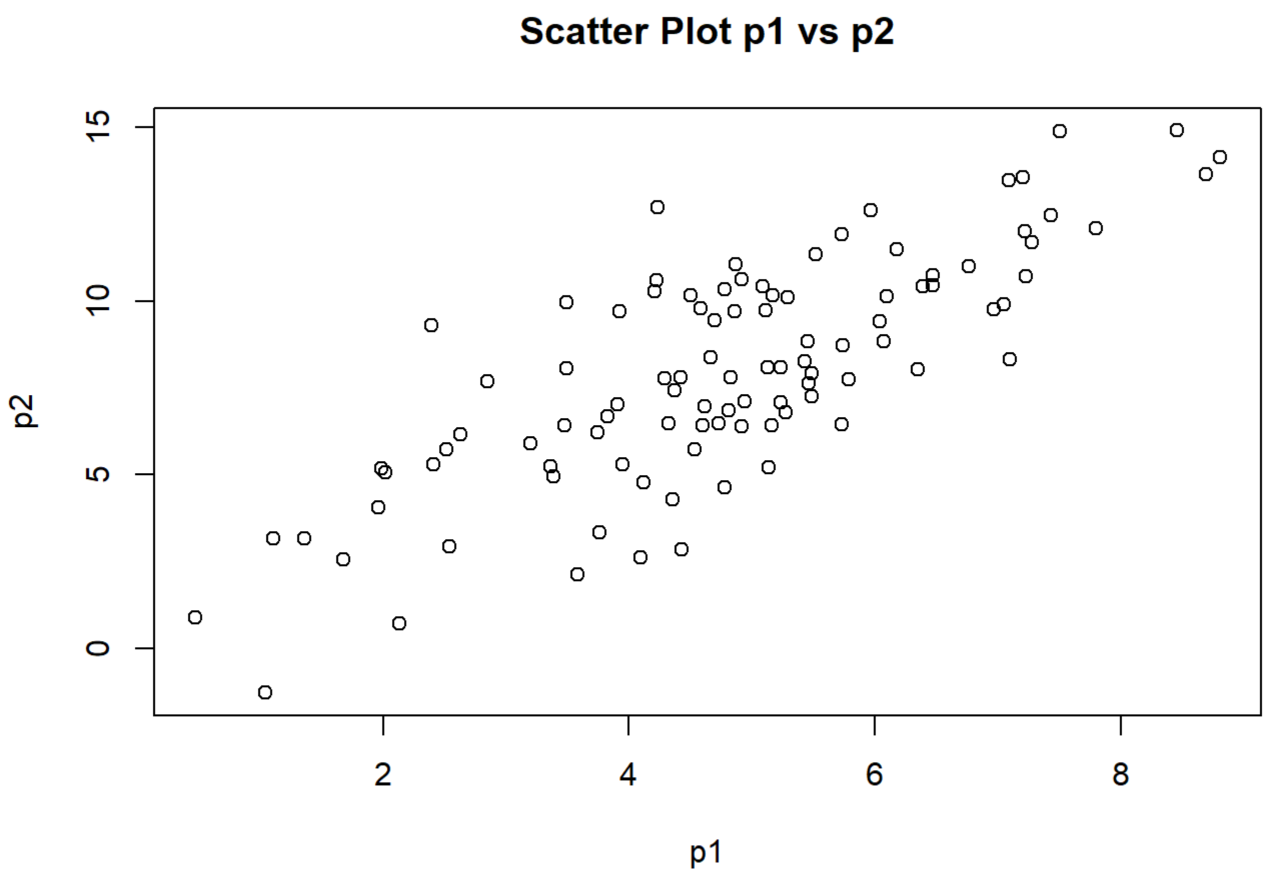
Explain why setting `p2=mean(p2)` would be a bad choice. Be specific and comment on why this is true for this particular case (value of `p1).

Answer Q1, Part B:

To estimate sales using the multiple regression model, we need both `p1` and `p2`. While `p1` is provided, we should not assume `p2=mean(p2)` to get reasonably accurate predictions because there could be an inherent relation between `p1` and `p2`.

Let's see scatter plot and correlation between `p1` and `p2`

```
plot(multi$p1, multi$p2, xlab = "p1", ylab = "p2", main = "Scatter Plot p1 vs p2")
```




```
print(paste0("Correlation between `p1` and `p2` is: ", cor(multi$p1, multi$p2)))
```

```
## [1] "Correlation between `p1` and `p2` is: 0.78333451317552"
```

From the plot and the correlation value, we can see that there is some correlation between `p1` and `p2`. Furthermore, if we fit a simple linear regression between `sales` and `p1`, we see-

```
slr = lm(formula = Sales ~ p1, data = multi)

summary(slr)
```

```
##
## Call:
## lm(formula = Sales ~ p1, data = multi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -513.91 -157.69   -1.42  155.20  650.20
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   211.16      66.49   3.176    0.002 **
## p1             63.71      13.04   4.886 4.01e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 223.4 on 98 degrees of freedom
## Multiple R-squared:  0.1959, Adjusted R-squared:  0.1877
## F-statistic: 23.87 on 1 and 98 DF,  p-value: 4.015e-06
```

In the simple linear regression model (`Sales ~ p1`), we observe the coefficient of `p1` very different from the same coefficient in the corresponding multiple linear regression model (`Sales ~ p1 + p2`). The difference in `p1` 's coefficient (-97.66 vs. 63.71) implies that there is an interaction between `p1` and `p2` and hence, we expect `p1` to change with change in `p2` . Thus it is a bad choice to assume `p2=mean(p2)` .

```
print(paste0("Mean value of `p2` is: ", mean(multi$p2)))
```

```
## [1] "Mean value of `p2` is: 7.999999929477"
```

```
print(paste0("Mean value of `p1` is: ", mean(multi$p1)))
```

```
## [1] "Mean value of `p1` is: 4.802319425021"
```

From the above values and the scatter plot, we can see that when `p2 = mean(p2)` = $8, we would expect `p1` to be ~$4.8. But, we want to measure the `sales` for `p1` = $7.5. Hence, it is incorrect to use `p2 = mean(p2)` when `p1` = $7.5. We should use the corresponding value of `p2` , which is ~$12.5 to predict `sales`

Q1, part C

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

Answer Q1, Part C:

```
# Multiple Linear Regression (p2 ~ p1)
multi_lm_p1p2 = lm(formula = p2 ~ p1, data=multi)

summary(multi_lm_p1p2)
```

```
##
## Call:
## lm(formula = p2 ~ p1, data = multi)
##
## 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
```

```
# Predict
predout = predict(multi_lm_p1p2, new=data.frame(p1=7.5))

predout
```

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

Hence, from the above regression model when `p1` = $7.5, then `p2` should be equal to $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.

Answer Q1, Part D:

Leveraging the multiple linear regression model (Sales ~ p1 + p2) to predict sales:

```
# Leveraging the multiple linear regression model (Sales ~ p1 + p2) to predict sales
pred_sales_mlr = predict(multi_lm, new=data.frame(p1=7.5, p2=12.00116))
print(paste0("Estimated sales when p1=$7.5 and p2=$12 is: ",pred_sales_mlr))
```

```
## [1] "Estimated sales when p1=$7.5 and p2=$12 is: 689.012059668933"
```

Leveraging the Simple linear regression model (Sales ~ p1) to predict sales:

```
# Leveraging the Simple linear regression model (Sales ~ p1) to predict sales
pred_sales_slr = predict(slr, new=data.frame(p1=7.5))
print(paste0("Estimated sales when p1=$7.5 is: ",pred_sales_slr))
```

```
## [1] "Estimated sales when p1=$7.5 is: 689.011805760678"
```

Hence, we see the estimated sales from both the models (SLR and MLR) to be same when `p1` = $7.5. This has to be true because in the SLR model (simple linear regression model) the coefficient of `p1` accounts for the impact of `p1` and the impact of all other variables which are related to `p1` (e.g. `p2`) on `sales` . Similarly, in the MLR model, by separating out `p2` and estimating `p2` by regressing `p2` on `p1` , we are essentially separating out the impact of `p2` explained by `p1` on `sales` . Thus both the models return the same `sales` estimate.

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

Answer Q2, Part A:

```
# Loading mvehicles dataset
data(mvehicles)

# Filtering only cars from the mvehicles dataset
cars = mvehicles[mvehicles$bodytype != "Truck",]

# Creating a new variable -> luxury * sporty
cars$luxury_sporty = cars$luxury * cars$sporty

# Fitting multiple linear regression model
vehicle_model = lm(log(emv)~luxury + sporty + luxury_sporty, data = cars)
summary(vehicle_model)
```

```
##
## 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
```

Using the relation

$$\frac{\partial}{\partial X1} E[Y|X1, X2] = \beta_1 + \beta 3X2$$

, we can derive the change in price as follows-

```
# Coefficients of the model
b_sporty = vehicle_model$coefficients['sporty']
b_luxury_sporty = vehicle_model$coefficients['luxury_sporty']

# Values to estimate sales
luxury_val = 0.30
change_in_sporty = 0.10

rate_of_change_sporty = b_sporty + (b_luxury_sporty * luxury_val)
emv_change = rate_of_change_sporty * change_in_sporty

# Since we regress on log(price), we take exponential
emv_change = exp(emv_change)

print(paste0("If `sporty` was increased by .1 units, holding `luxury` constant at .30 units, the
n we would expect `emv` to multiply by: ", emv_change))
```

```
## [1] "If `sporty` was increased by .1 units, holding `luxury` constant at .30 units, then we w
ould expect `emv` to multiply by: 0.997848887750542"
```

Hence, when we hold `luxury` constant at .30 unit and increase `sporty` by 0.1 units, we expect the price of the car to 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.

Answer Q2, Part B:

```
# Coefficients of the model
b_sporty = vehicle_model$coefficients['sporty']
b_luxury_sporty = vehicle_model$coefficients['luxury_sporty']

# Values to estimate sales
luxury_val = 0.70
change_in_sporty = 0.10

rate_of_change_sporty = b_sporty + (b_luxury_sporty * luxury_val)
emv_change = rate_of_change_sporty * change_in_sporty

# Since we regress on log(price), we take exponential
emv_change = exp(emv_change)

print(paste0("If `sporty` was increased by .1 units, holding `luxury` constant at .70 units, the
n we would expect `emv` to multiply by: ", emv_change))
```

```
## [1] "If `sporty` was increased by .1 units, holding `luxury` constant at .70 units, then we w
ould expect `emv` to multiply by: 1.05083380964118"
```

Hence, when we hold `luxury` constant at .70 unit and increase `sporty` by 0.1 units, we expect the price of the car to increase to 105% 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?

Answer Q2, Part C:

The answers in part A and part B are different because we expect the inherent interaction between `sporty` and `luxury` to influence change in `price` . The impact of `sporty` on `price` changes with `luxury` . Using the relation

$$\frac{\partial}{\partial X1} E[Y|X1, X2] = \beta_1 + \beta 3X2$$

, we can say that the rate of change in log(price) by change in `sporty` is a linear relation which depends on `luxury` . Hence, as the value of `luxury` changes (0.3 vs. 0.7), we expect the impact of `sporty` on `price` to change.

The interaction term “ `sporty * luxury` ” and its coefficient are intuitive. The positive coefficient (1.29) for the interaction term implies that the impact of `sporty` on `price` increases as `luxury` index increases. This is expected because the more luxurious a car is , we can expect its price to increase a lot more as we increase the “sportiness” of the car. The decrease in the price of cars at lower values of `luxury` is because there is not much relationship between the sportiness of a car and its luxury for less luxurious 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.

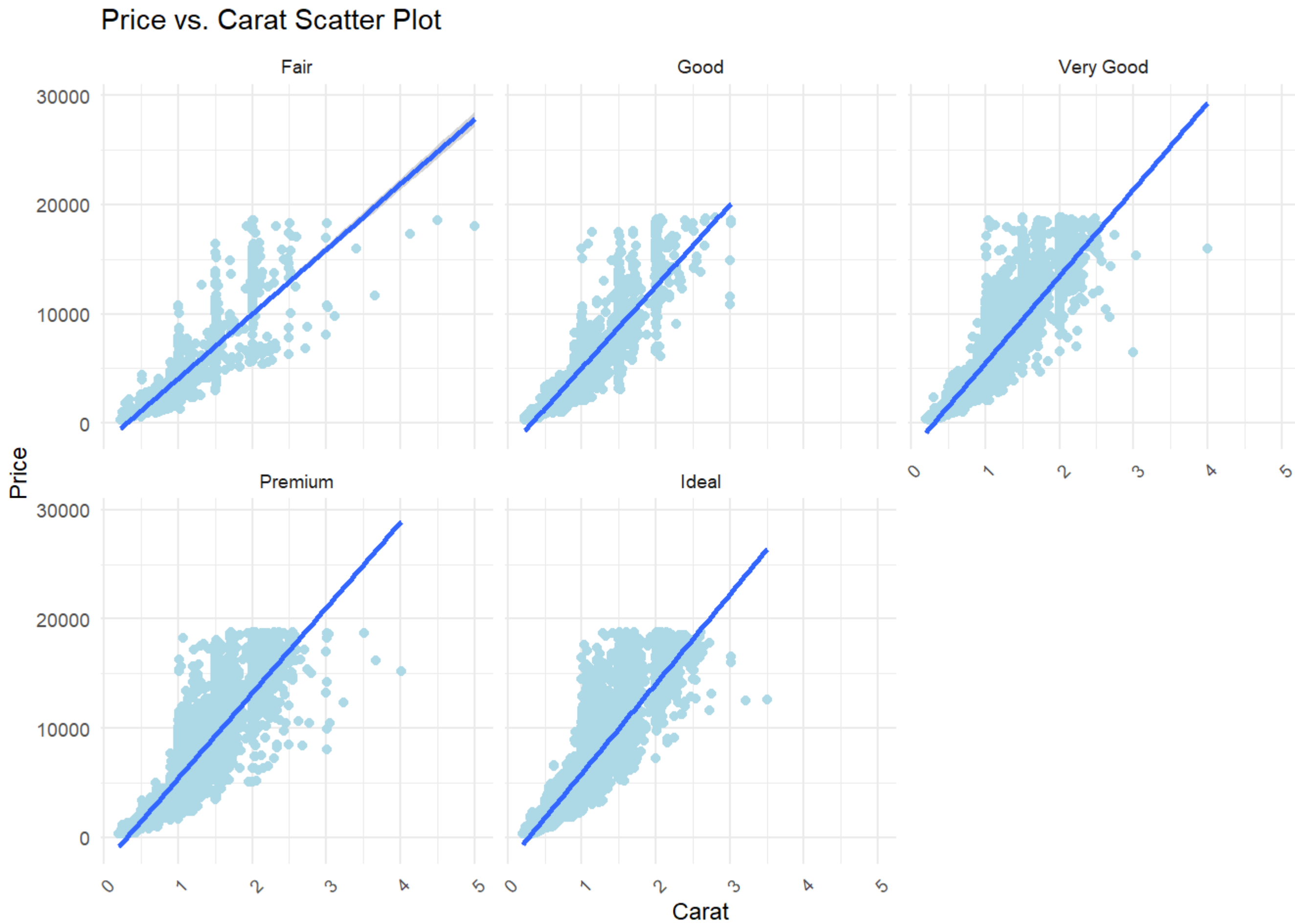
Answer Q3, Part 1:

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

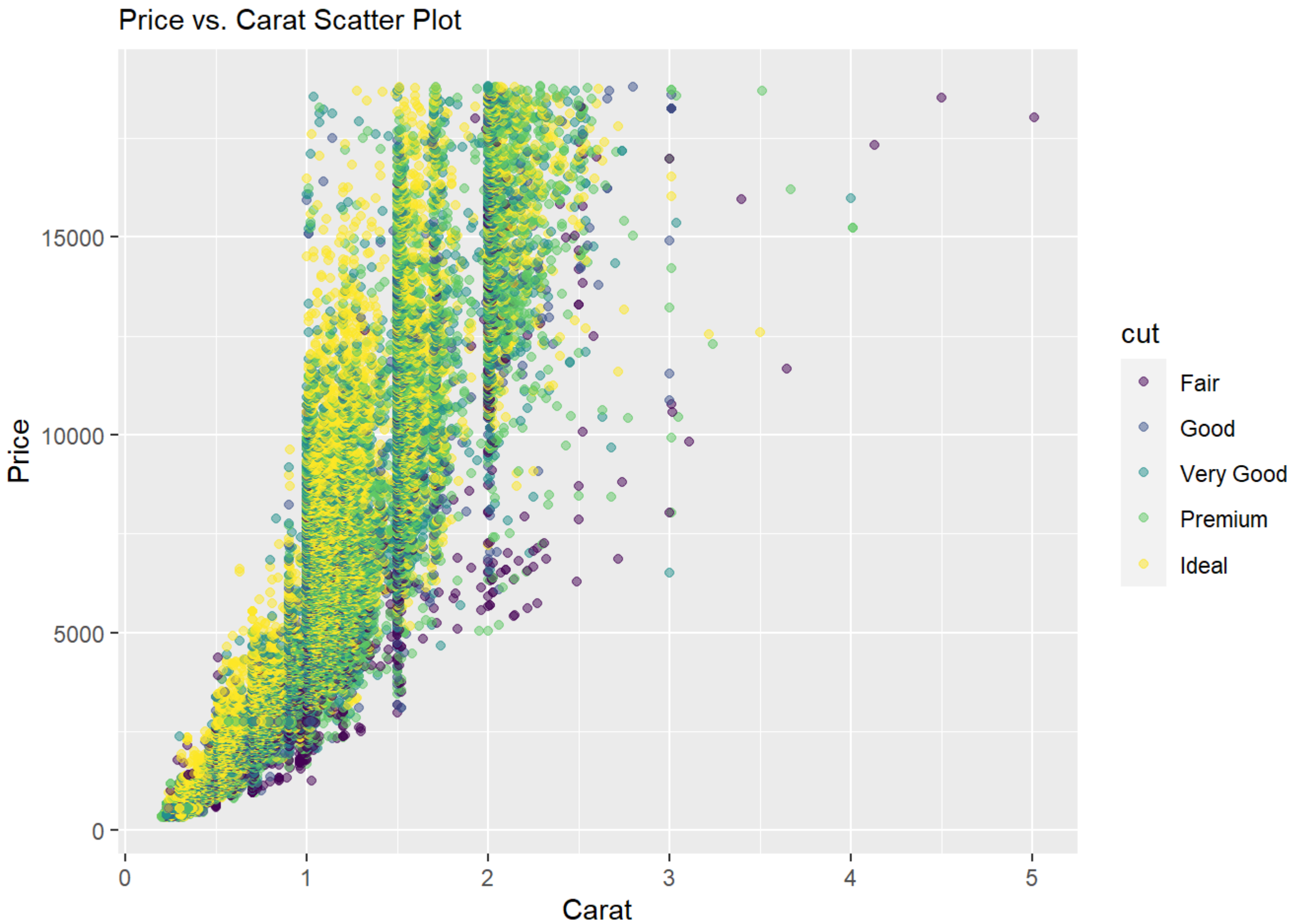
Scatter Plot with Actual Values (i.e. No Transformation):

```
ggplot(diamonds, aes(x= carat, y = price)) +
  geom_point(color="light blue") +
  facet_wrap(~cut) +
  theme_minimal(base_size = 9) +
  theme(axis.text.x = element_text(angle = 45,
                                    hjust = 1)) +
  geom_smooth(method = "lm") +
  labs(title = "Price vs. Carat Scatter Plot",
       x = "Carat",
       y = "Price")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(data=diamonds, aes(x=carat, y =price, color=cut)) +
  geom_point(alpha=0.5) +
  labs(y="Price", x="Carat", subtitle="Price vs. Carat Scatter Plot")
```



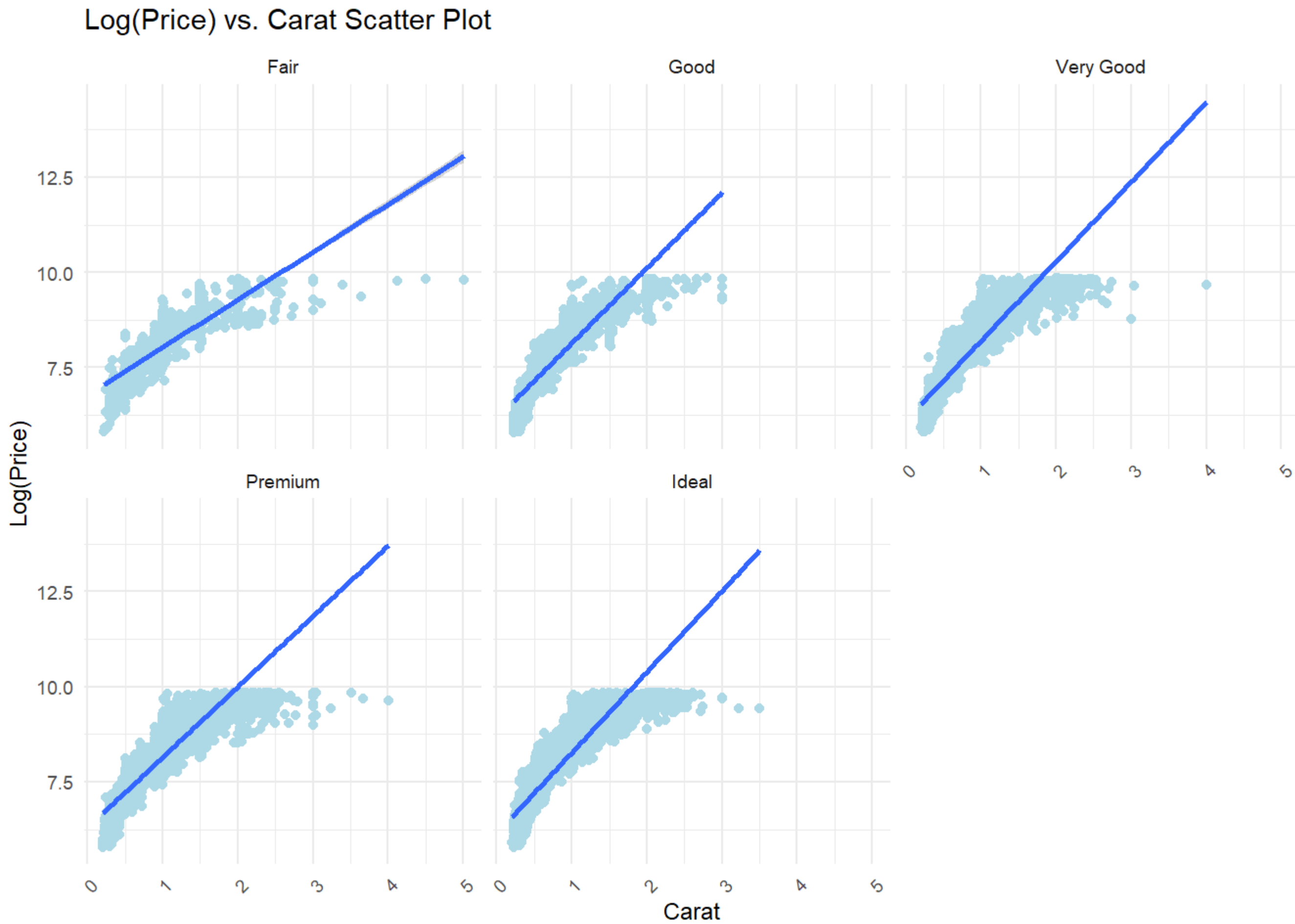
Key Observations: - 1. Across all `cut` categories, the price of a diamond has an increasing trend as `Carat` value increases. This is expected as we know that high carat diamonds are more expensive than lower carat diamonds.

- 2. The variance in prices is increasing with increasing values of `Carat` (i.e. `x`). Therefore, the variance cannot be assumed to be approximately constant as `x` (i.e. `Carat`) increases. This is true across all `cut` categories.

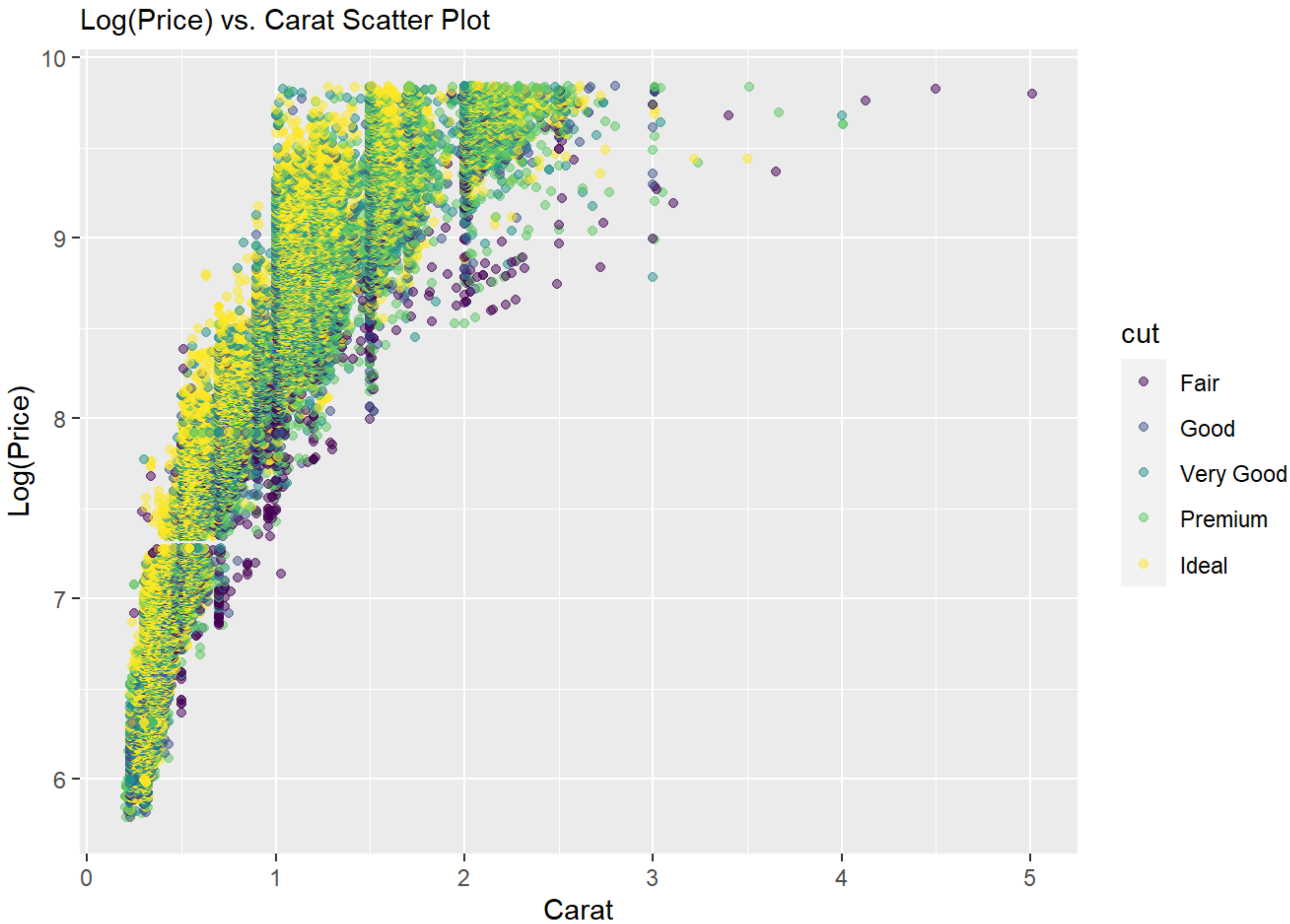
Log Transformation:

```
ggplot(diamonds, aes(x= carat, y = log(price))) +
  geom_point(color="light blue") +
  facet_wrap(~cut) +
  theme_minimal(base_size = 9) +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1)) +
  geom_smooth(method = "lm") +
  labs(title = "Log(Price) vs. Carat Scatter Plot",
       x = "Carat",
       y = "Log(Price)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



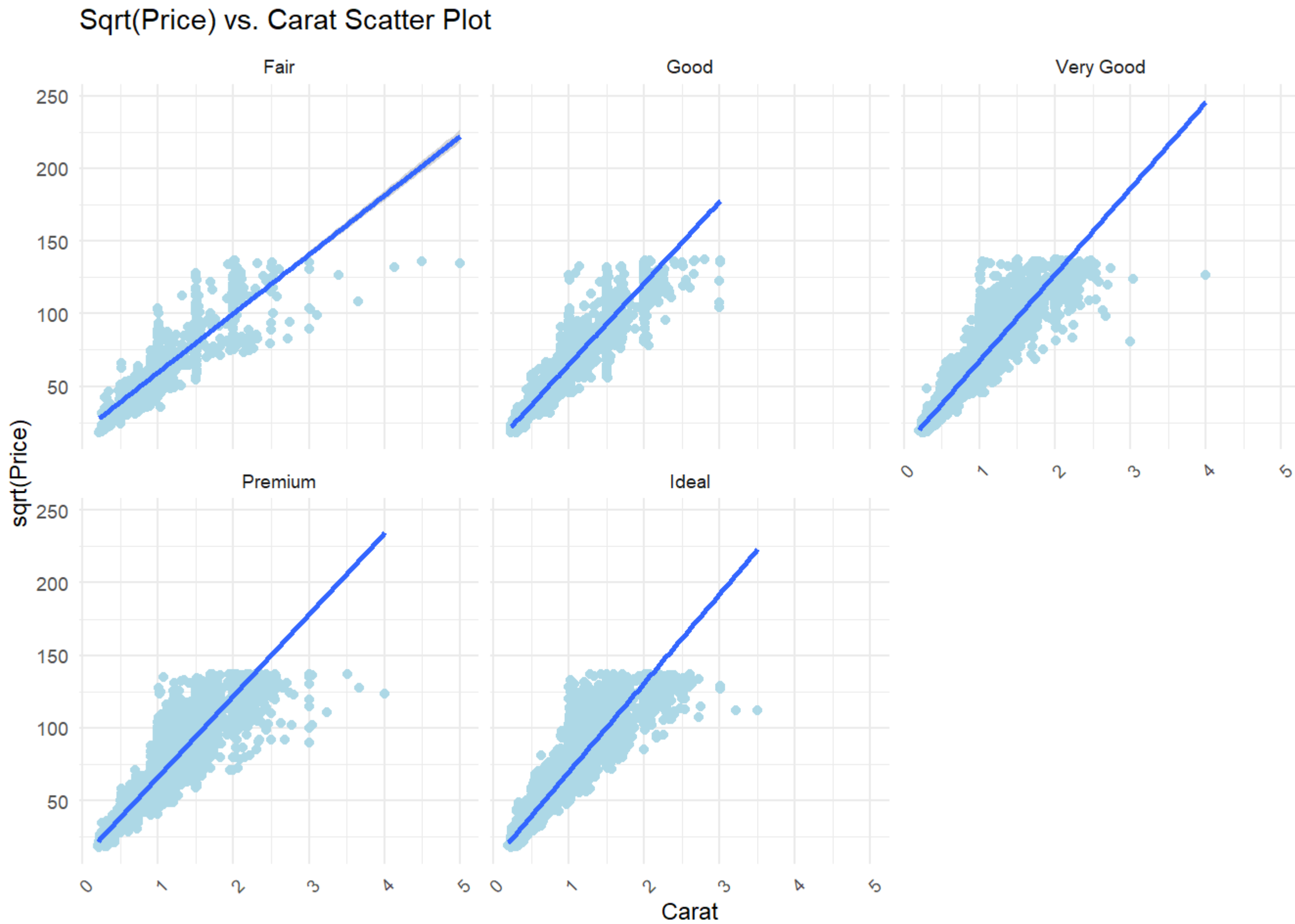

```
ggplot(data=diamonds, aes(x=carat, y =log(price), color=cut)) +
  geom_point(alpha=0.5) +
  labs(y="Log(Price)", x="Carat", subtitle="Log(Price) vs. Carat Scatter Plot")
```



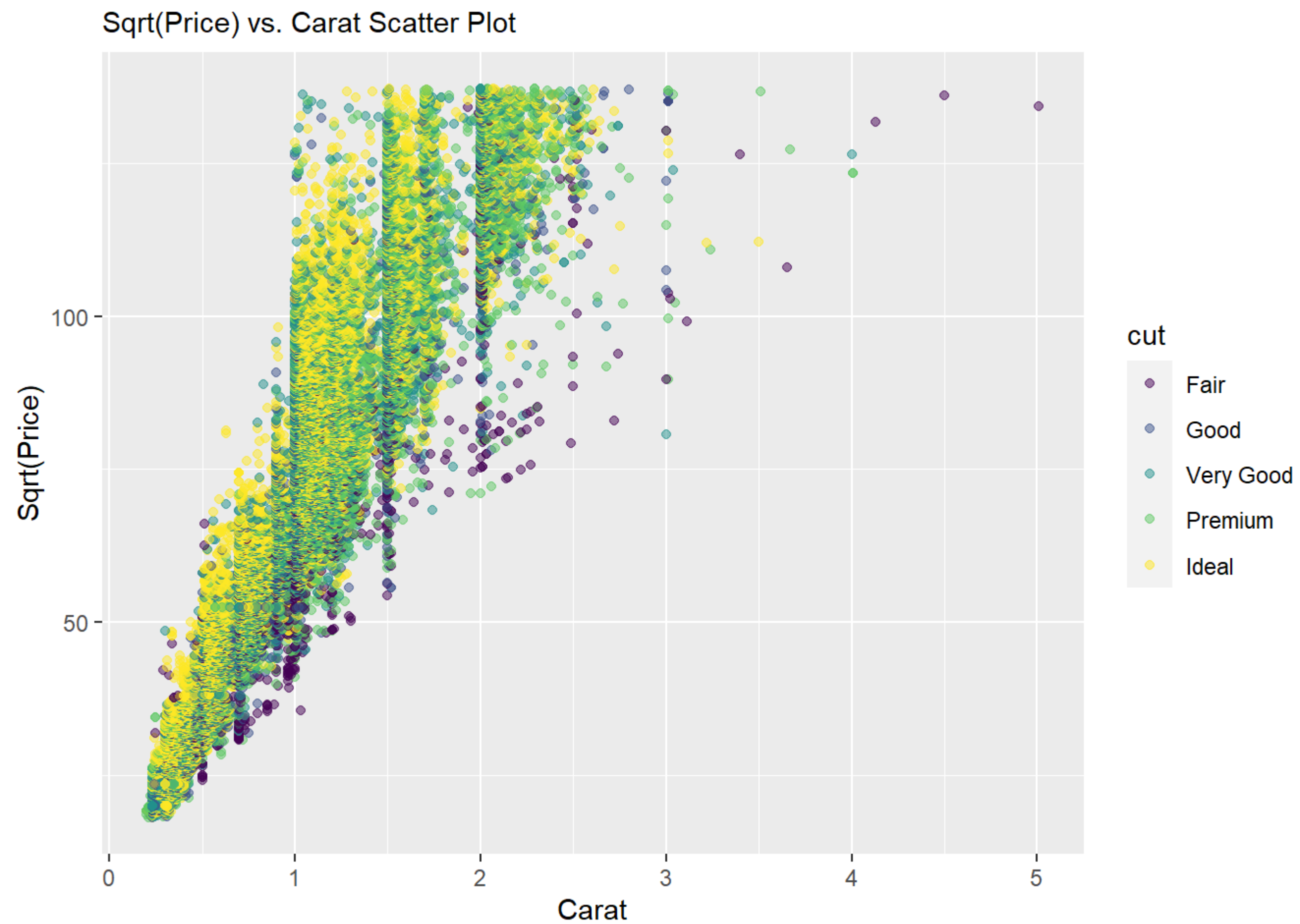
Square Root Transformation:

```
ggplot(diamonds, aes(x= carat, y = sqrt(price))) +
  geom_point(color="light blue") +
  facet_wrap(~cut) +
  theme_minimal(base_size = 9) +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1)) +
  geom_smooth(method = "lm") +
  labs(title = "Sqrt(Price) vs. Carat Scatter Plot",
       x = "Carat",
       y = "sqrt(Price)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(data=diamonds, aes(x=carat, y =sqrt(price), color=cut)) +
  geom_point(alpha=0.5) +
  labs(y="Sqrt(Price)", x="Carat", subtitle="Sqrt(Price) vs. Carat Scatter Plot")
```



Key Observations: If we compare the plots for the actual values of `price` vs. `carat` with the transformed values of `price` (i.e. log() and sqrt() transformation), we observe that variance of `price` is changing with `carat`. This is true for `Log(price)` and `sqrt(price)` as well. However, overall the square root transformation of `price` seems suitable to fit a regression line on the data as it has the lowest change in variance in `Price` and yields are relatively more linear relationship between `price` and `carat`.

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?

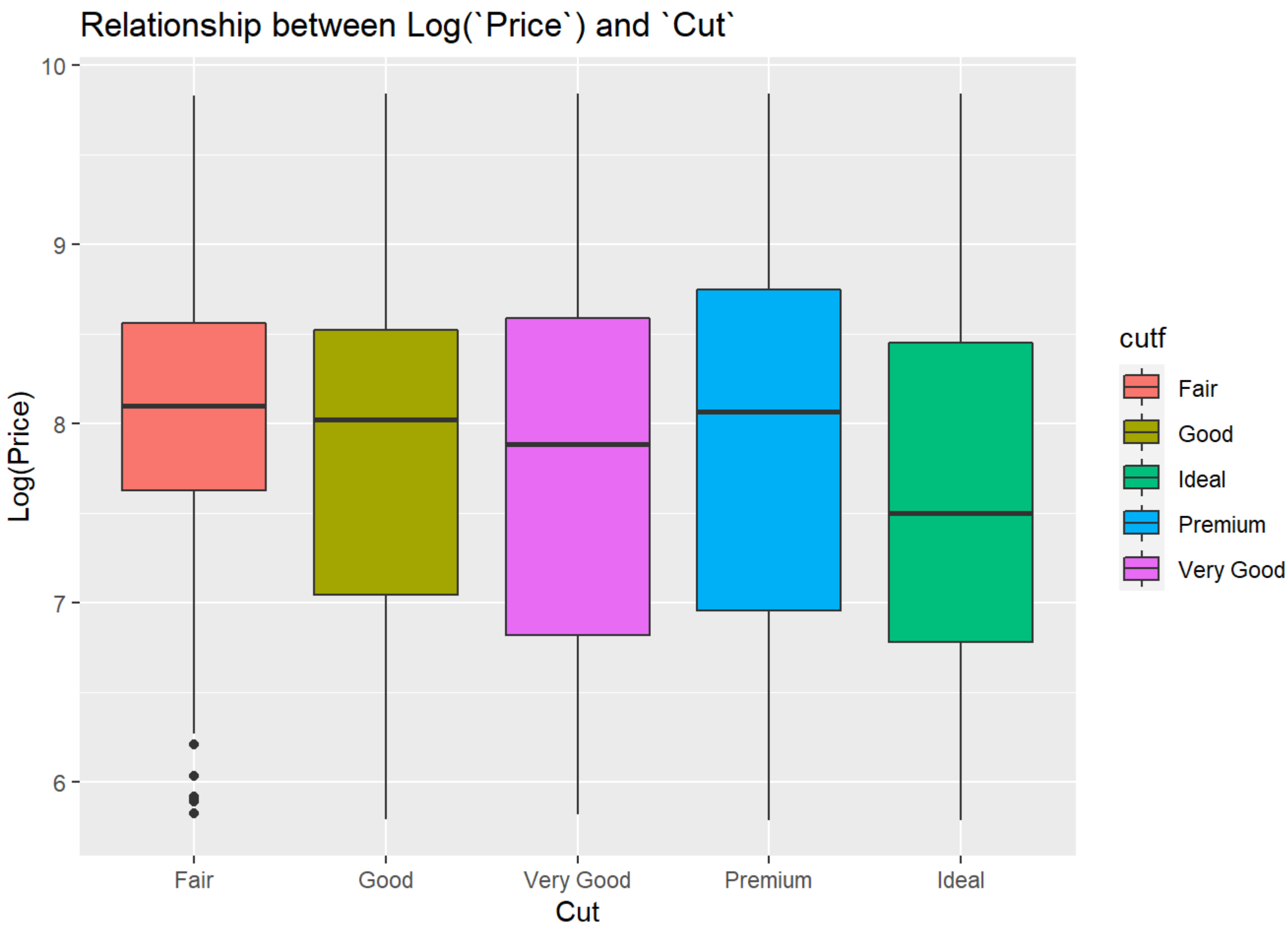
note: `cut` is a special type of variable called an ordered factor in R. For ease of interpretation, convert the ordered factor into a "regular" or non-ordinal factor.

Answer Q3, Part 2:

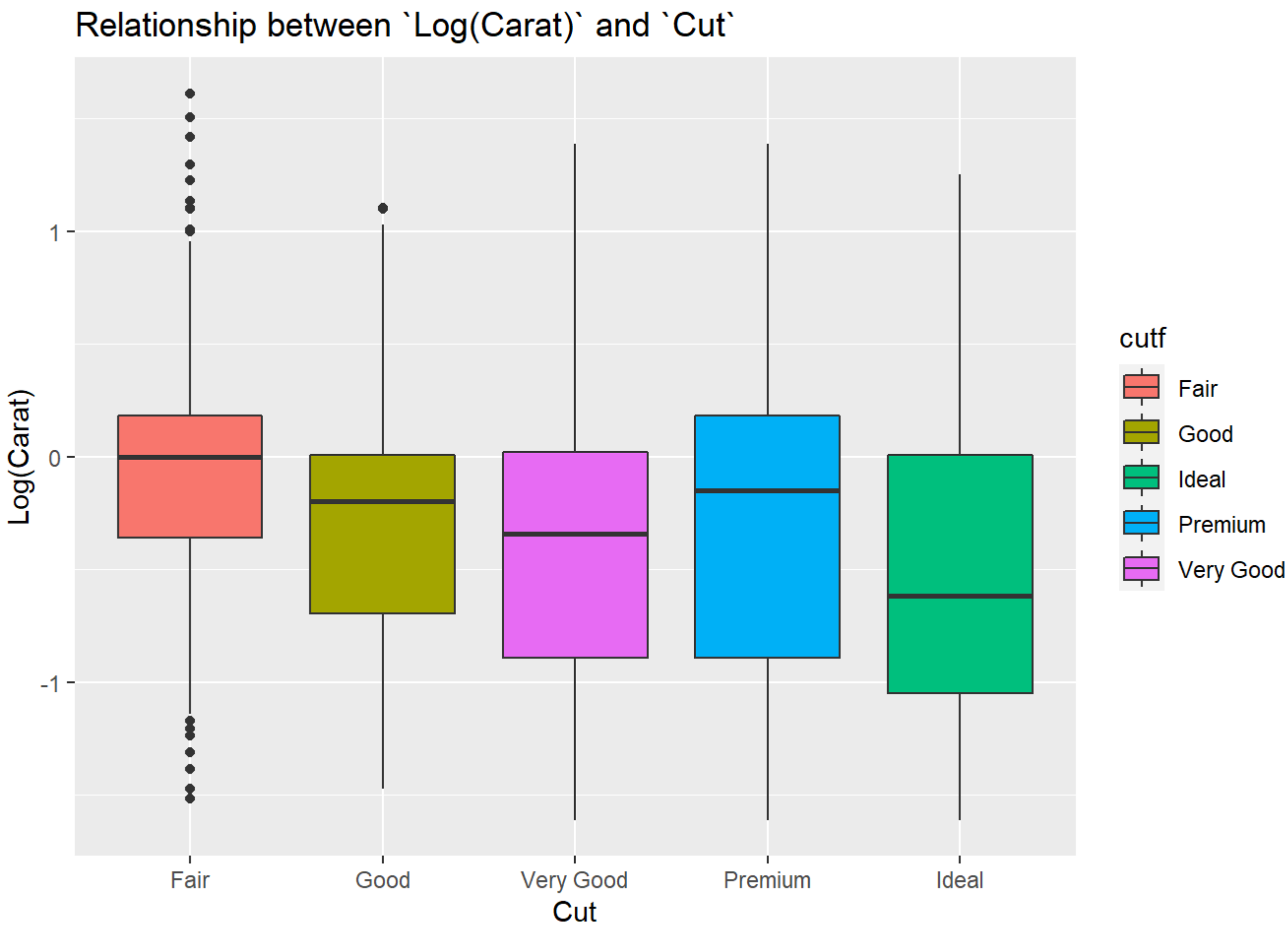
We have two decisions to make to build our regression model:
- 1. Should we use both `carat` and `cut` or just `carat`? - 2. Which transformation is most suitable to build the linear regression model

1. Relationship between a. `Log(Price) - Cut` and b. `Log(carat) - cut`

```
qplot(y=log(price), x= factor(cutf, levels = c("Fair", "Good", "Very Good", "Premium", "Ideal"
)), data=diamonds, geom=c("boxplot"),
   fill=cutf, main="Relationship between Log(`Price`) and `Cut`",
   xlab="Cut", ylab="Log(Price)")
```




```
qplot(y=log(carat), x= factor(cutf, levels = c("Fair", "Good", "Very Good", "Premium", "Ideal"
)), data=diamonds, geom=c("boxplot"),
   fill=cutf, main="Relationship between `Log(Carat)` and `Cut`",
   xlab="Cut", ylab="Log(Carat)")
```

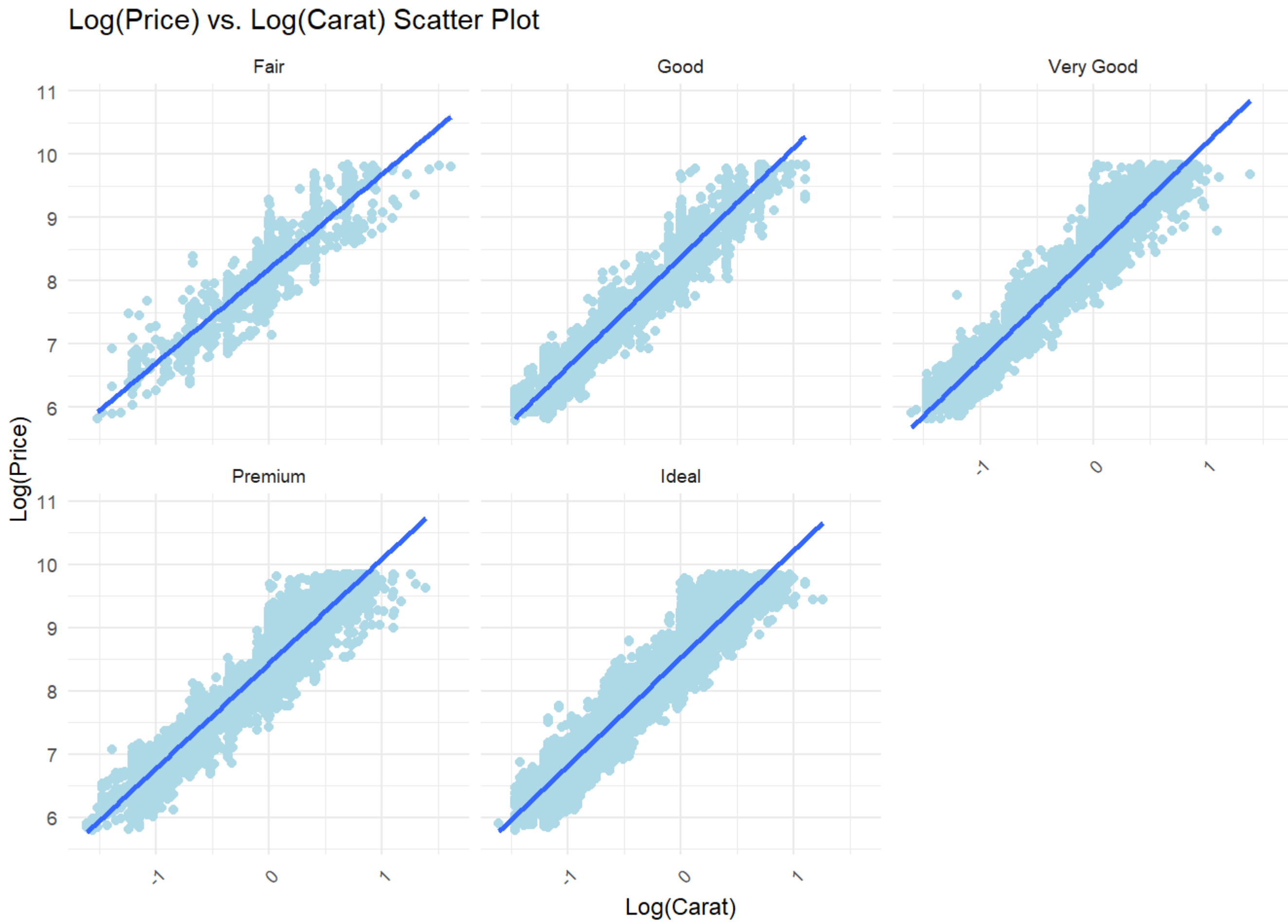


Key Observations: From the above plots, we can see that `cut` does not have a strong relationship with `price` or with `carat` . The min `price` values and the max `price` values across all `cut` categories is very similar (i.e. min `price` value of “Ideal” cut is very similar to min `price` value “Fair” cut diamonds). The same can be said for the 3rd Quartile `price` value. Hence, we are discarding this feature from `price` prediction.

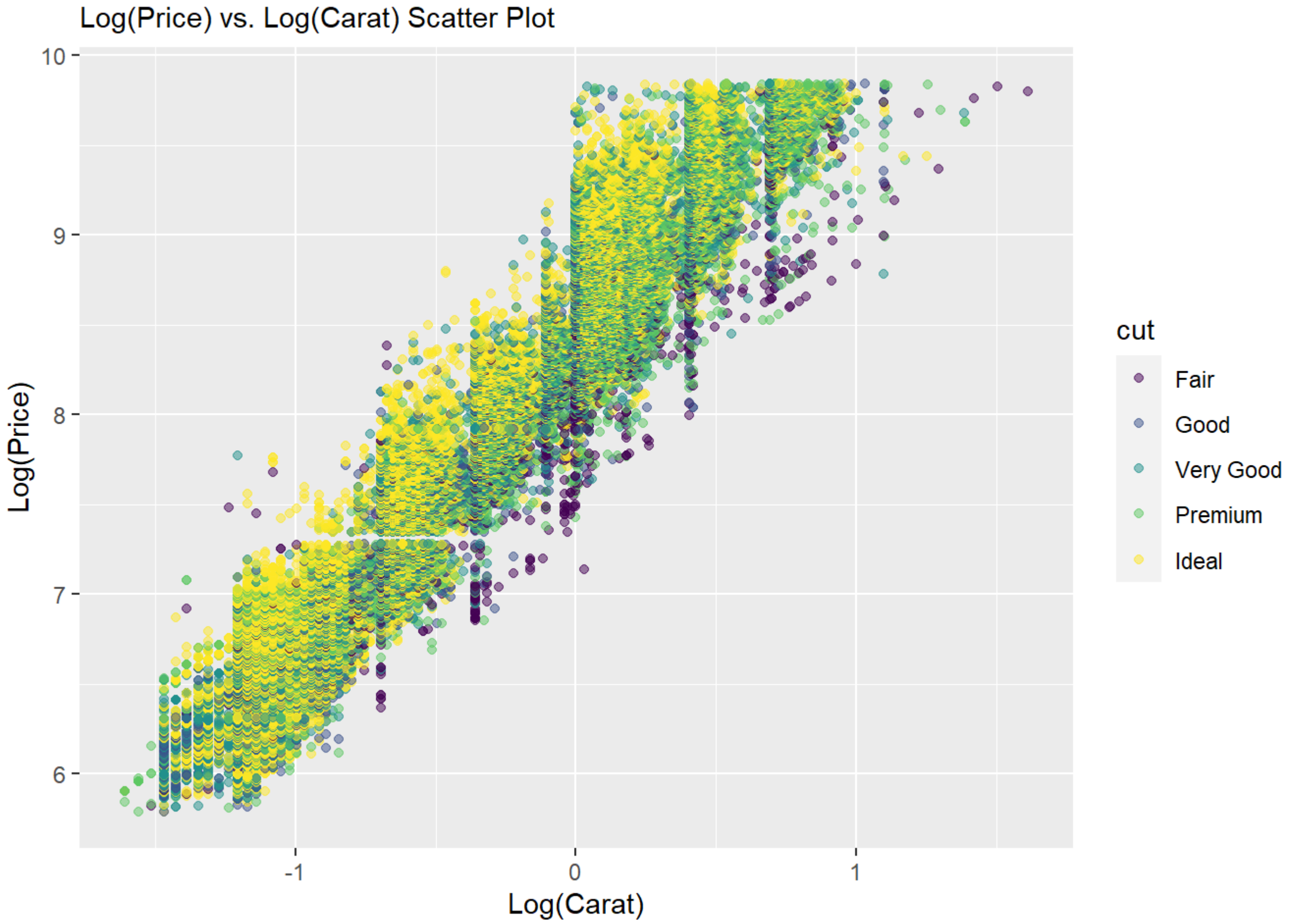
We will use Log-Log transformation on `price ~ carat` as that yields the best linear relationship between `price` and `carat` .

```
ggplot(diamonds, aes(x= log(carat), y = log(price))) +
  geom_point(color="light blue") +
  facet_wrap(~cut) +
  theme_minimal(base_size = 9) +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1)) +
  geom_smooth(method = "lm") +
  labs(title = "Log(Price) vs. Log(Carat) Scatter Plot",
       x = "Log(Carat)",
       y = "Log(Price)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(data=diamonds, aes(x=log(carat), y =log(price), color=cut)) +
  geom_point(alpha=0.5) +
  labs(y="Log(Price)", x="Log(Carat)", subtitle="Log(Price) vs. Log(Carat) Scatter Plot")
```



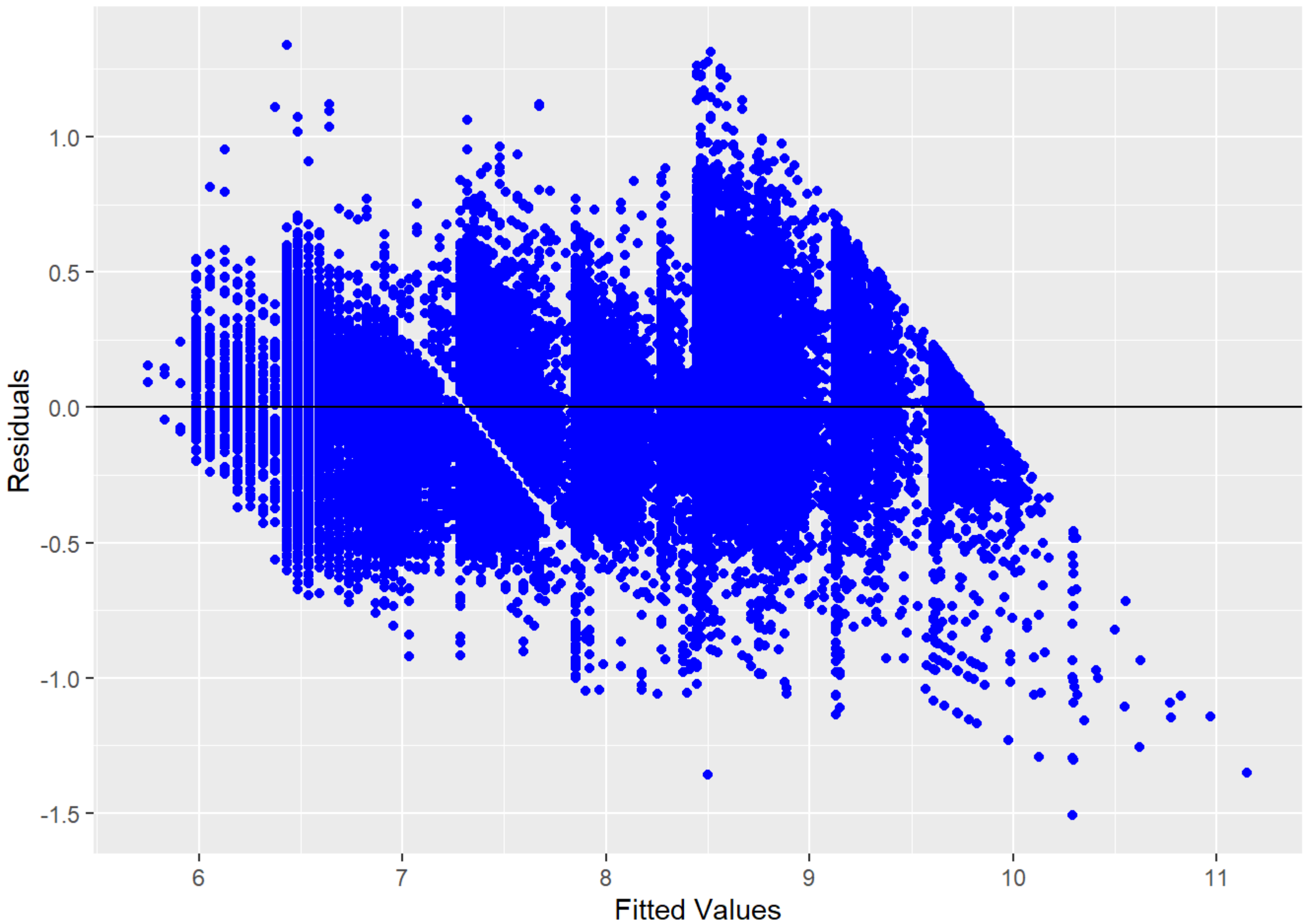
Let’s fit our regression model based on the above two decisions:

```
# Fitting a multiple linear regression model
diamond_mlr_2 = lm(formula=log(price) ~ log(carat) ,data=diamonds)
summary(diamond_mlr_2)
```

```
##
## 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
```

Regression diagnostic plots of residuals vs. fitted values

```
qplot(x=fitted(diamond_mlr_2), y=resid(diamond_mlr_2), colour = I("blue"), xlab="Fitted Values",
ylab="Residuals") + geom_hline(yintercept=0)
```



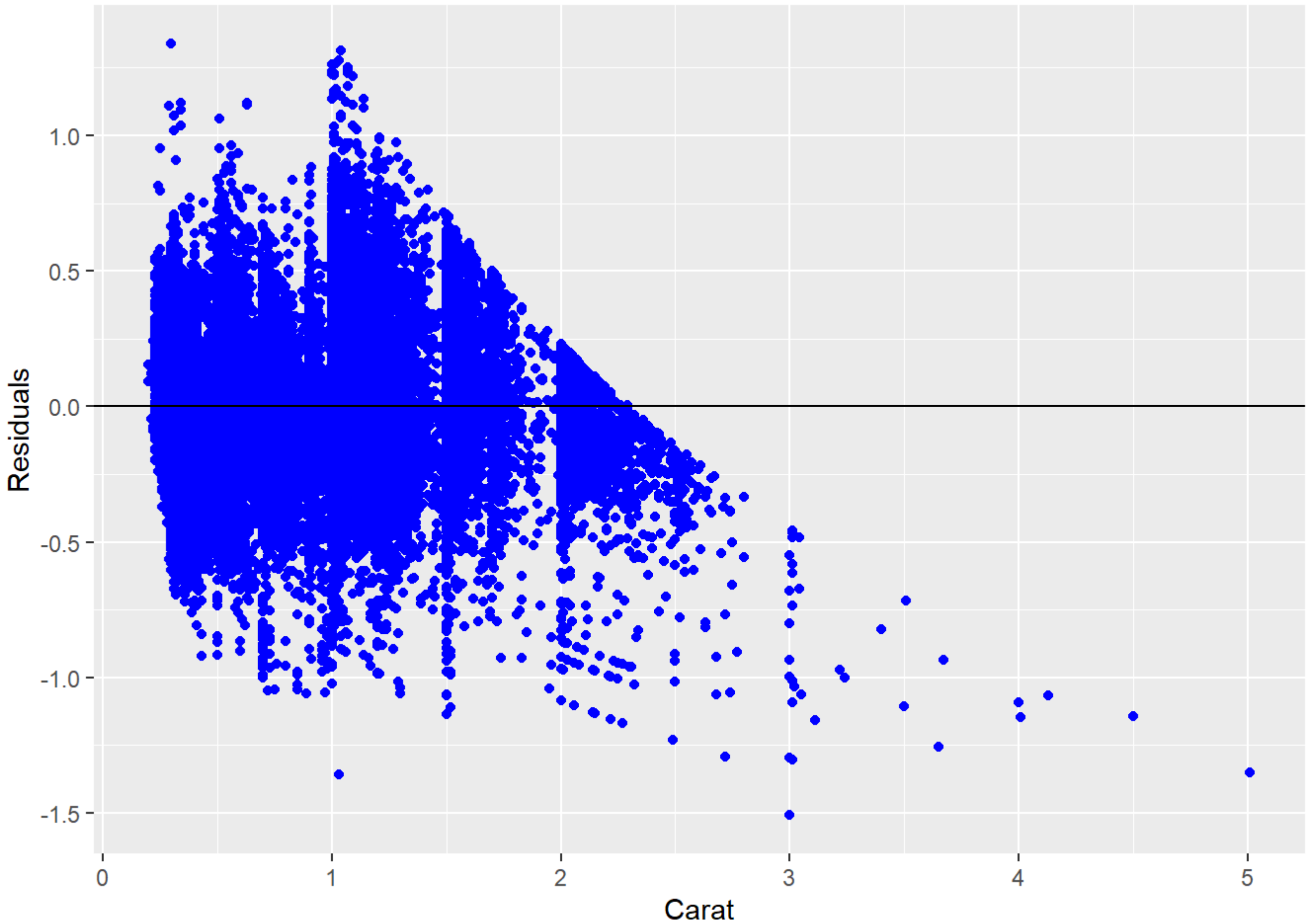

```
cor(fitted(diamond_mlr_2), resid(diamond_mlr_2))
```

```
## [1] 1.095253e-16
```

Key observations: The residuals are randomly distributed above and below the x-axis and have no correlation with the fitted values. This implies that the linear model assumption holds true Corr(residuals, fitted values) = 0. Furthermore, the variance in residuals is approximately constant across the range of fitted values. This is another assumption of linear model which appears to be valid.

Regression diagnostic plots of residuals vs. `carat`

```
qplot(x=diamonds$carat, y=resid(diamond_mlr_2), colour = I("blue"), xlab="Carat", ylab="Residual
s") + geom_hline(yintercept=0)
```




```
cor(diamonds$carat, resid(diamond_mlr_2))
```

```
## [1] -0.01388275
```

Key observations: The residuals mostly appear to be randomly distributed above and below the x-axis and have ~0 correlation with `carat` . This implies that the linear model assumption holds true Corr(residuals, fitted values) = 0.