# PSTAT 126 Final Project

Kacie Chong (#7423544)

2025-05-21

## Load the data and all libraries

```
set.seed(123)
data <- read.csv("/Users/kaciechong/Desktop/Diamonds.csv")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(faraway)
```

## Part 1: Data Description and Descriptive Statistic

### 1

The first task was to select a random sample of at least 1000 observations, incorporating all categorical and numerical variables, for a total of 10 variables.

```
sample <- data %>%
  select(carat, cut, color, clarity, depth, price, table, x, y, z) %>%
  sample_n(1000)
```

### 2

Then, I called the summary function and explored the structure of the sample. summary(sample) outputs statistical summaries per column, such as min, max, median, mean, quartiles, while str(sample) outputs the structure of the data frame, giving insight into data types and each variable's content.

```r
summary(sample)
```

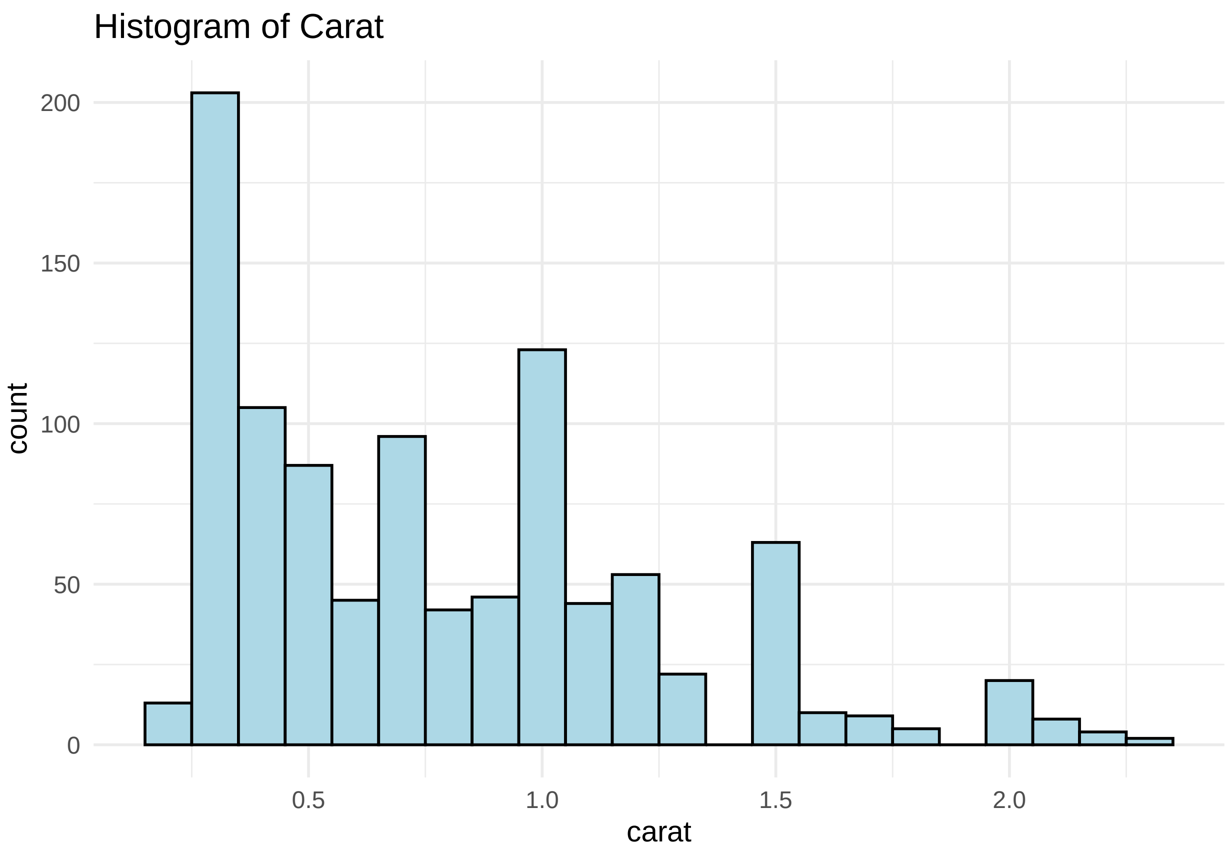
```
##      carat            cut               color             clarity        
##  Min.   :0.2300   Length:1000        Length:1000        Length:1000      
##  1st Qu.:0.4000   Class :character   Class :character   Class :character 
##  Median :0.7100   Mode  :character   Mode  :character   Mode  :character 
##  Mean   :0.7962                                                          
##  3rd Qu.:1.0400                                                          
##  Max.   :2.3200                                                          
##      depth           price             table             x        
##  Min.   :57.50   Min.   :  368.0   Min.   :52.00   Min.   :3.920  
##  1st Qu.:61.00   1st Qu.:  956.8   1st Qu.:56.00   1st Qu.:4.707  
##  Median :61.90   Median : 2522.0   Median :57.00   Median :5.720  
##  Mean   :61.79   Mean   : 3935.9   Mean   :57.41   Mean   :5.733  
##  3rd Qu.:62.50   3rd Qu.: 5301.8   3rd Qu.:59.00   3rd Qu.:6.530  
##  Max.   :69.60   Max.   :18706.0   Max.   :66.00   Max.   :8.570  
##        y               z        
##  Min.   :3.960   Min.   :0.000  
##  1st Qu.:4.718   1st Qu.:2.900  
##  Median :5.720   Median :3.540  
##  Mean   :5.736   Mean   :3.539  
##  3rd Qu.:6.513   3rd Qu.:4.030  
##  Max.   :8.520   Max.   :5.280
```

```r
str(sample)
```

```
## ’data.frame’:    1000 obs. of  10 variables:
##  $ carat  : num  0.73 0.7 0.31 0.31 0.31 0.83 0.51 0.7 0.4 1.1 ...
##  $ cut    : chr  "Ideal" "Ideal" "Ideal" "Ideal" ...
##  $ color  : chr  "I" "G" "D" "H" ...
##  $ clarity: chr  "VS1" "VS1" "VS1" "VVS1" ...
##  $ depth  : num  60.7 60.8 61.6 62.2 60.9 63.7 62.5 64.2 61.6 61.2 ...
##  $ price  : int  2397 3300 713 707 987 3250 1668 1771 1053 4640 ...
##  $ table  : num  56 56 55 56 55 59 58 58 56 61 ...
##  $ x      : num  5.85 5.73 4.3 4.34 4.39 5.95 5.12 5.59 4.73 6.61 ...
##  $ y      : num  5.81 5.8 4.33 4.37 4.41 5.89 5.18 5.62 4.78 6.66 ...
##  $ z      : num  3.54 3.51 2.66 2.71 2.68 3.77 3.22 3.6 2.93 4.01 ...
```

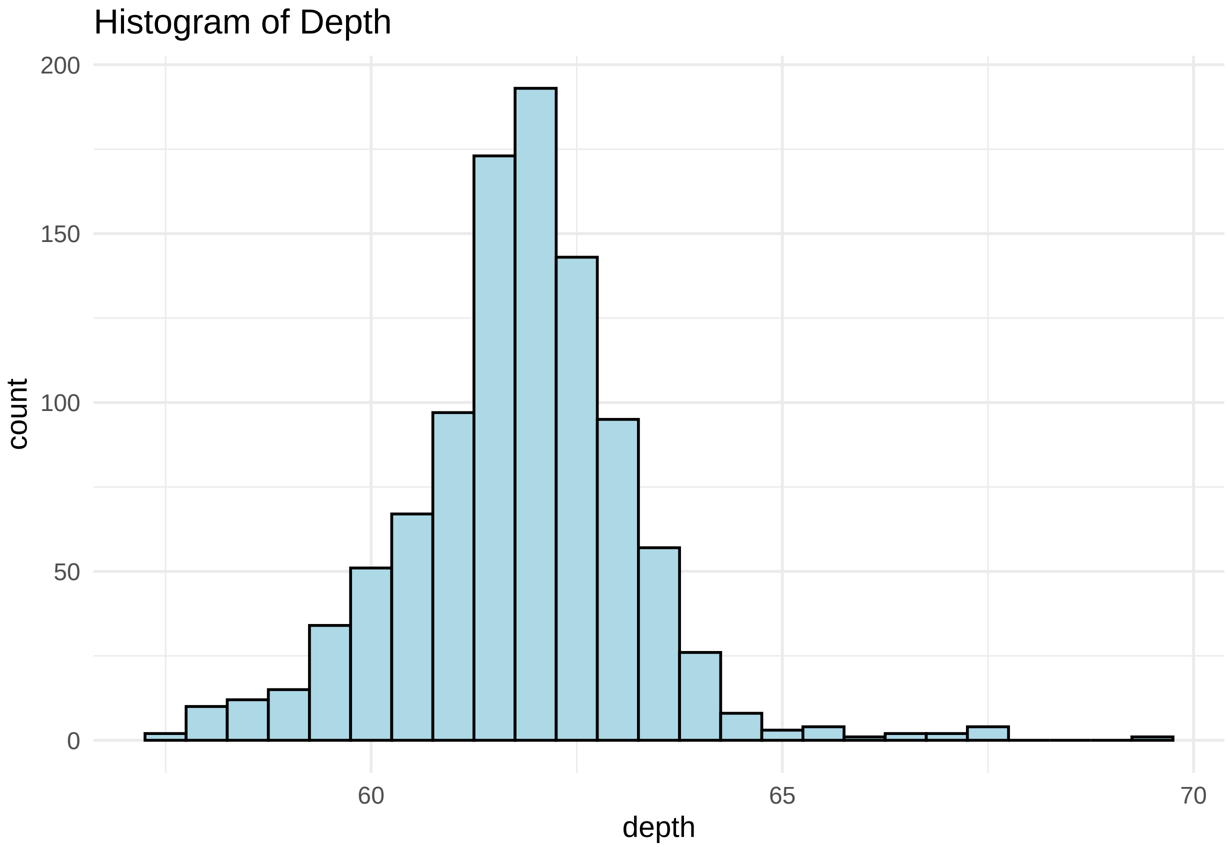
For the continuous random variables, I created histograms as shown below:

```r
ggplot(sample, aes(x = carat)) +
  geom_histogram(binwidth = 0.1, fill = "lightblue", color = "black") +
  ggtitle("Histogram of Carat") +
  theme_minimal()
```



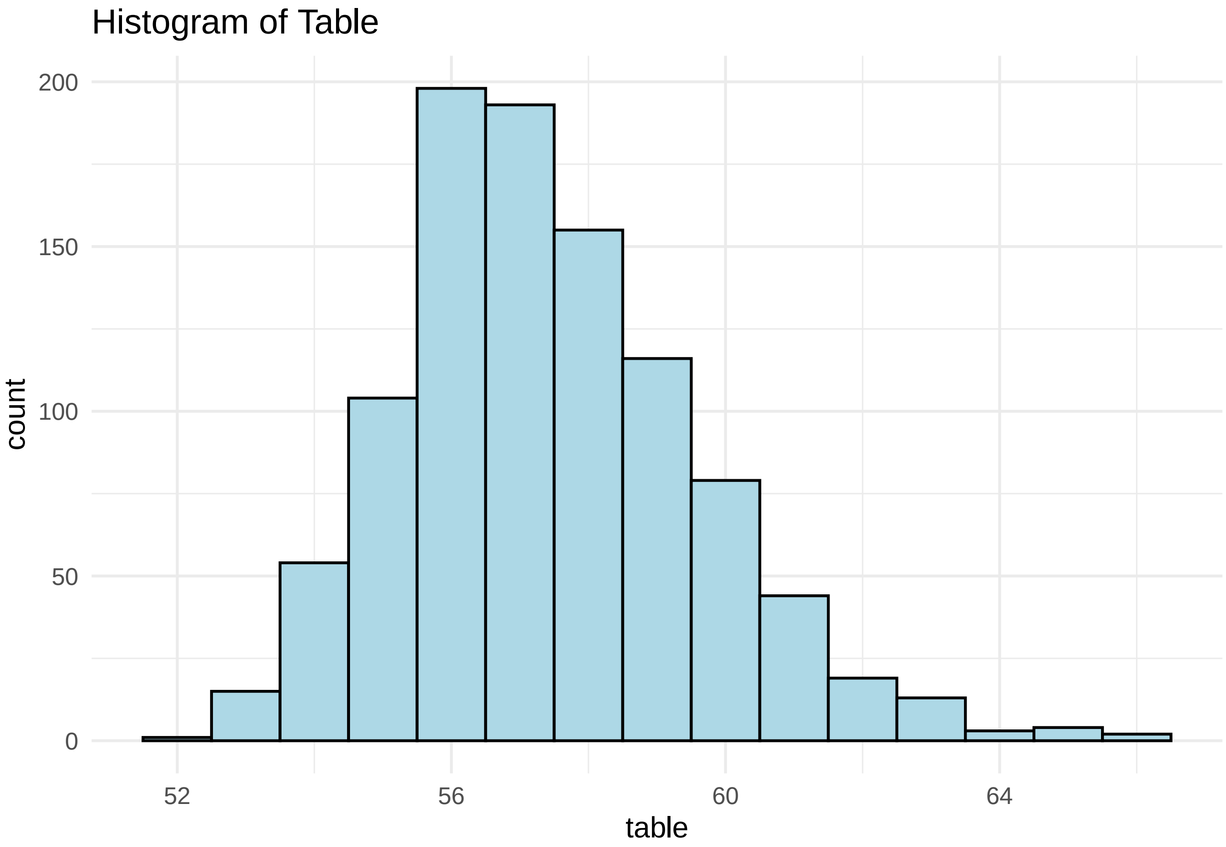
- Histogram for carat: The histogram is right skewed with a maximum count of roughly 200 at a carat of 3. Therefore, there are mostly lower carat diamonds in the sample. The highest frequency is for carats of 0 to 1, with the frequencies dropping as the carat increases; this is reasonable as higher carats are rarer.

```r
ggplot(sample, aes(x = depth)) +
  geom_histogram(binwidth = 0.5, fill = "lightblue", color = "black") +
  ggtitle("Histogram of Depth") +
  theme_minimal()
```



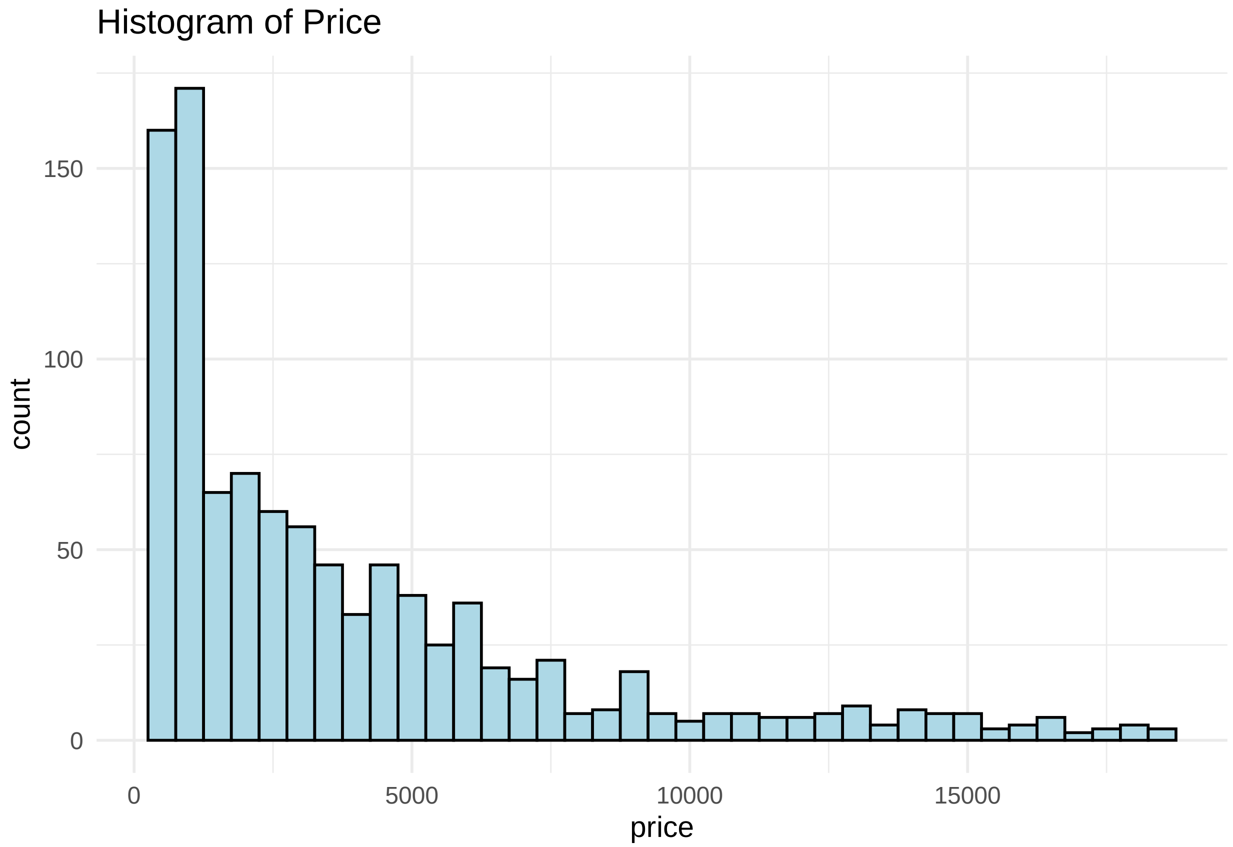
- Histogram for depth: The histogram seems to be roughly normally distributed with a maximum count of 190 at a depth of around 62. There seems to be a few outliers but it is mostly centered around the depth of 62.

```
ggplot(sample, aes(x = table)) +
  geom_histogram(binwidth = 1, fill = "lightblue", color = "black") +
  ggtitle("Histogram of Table") +
  theme_minimal()
```



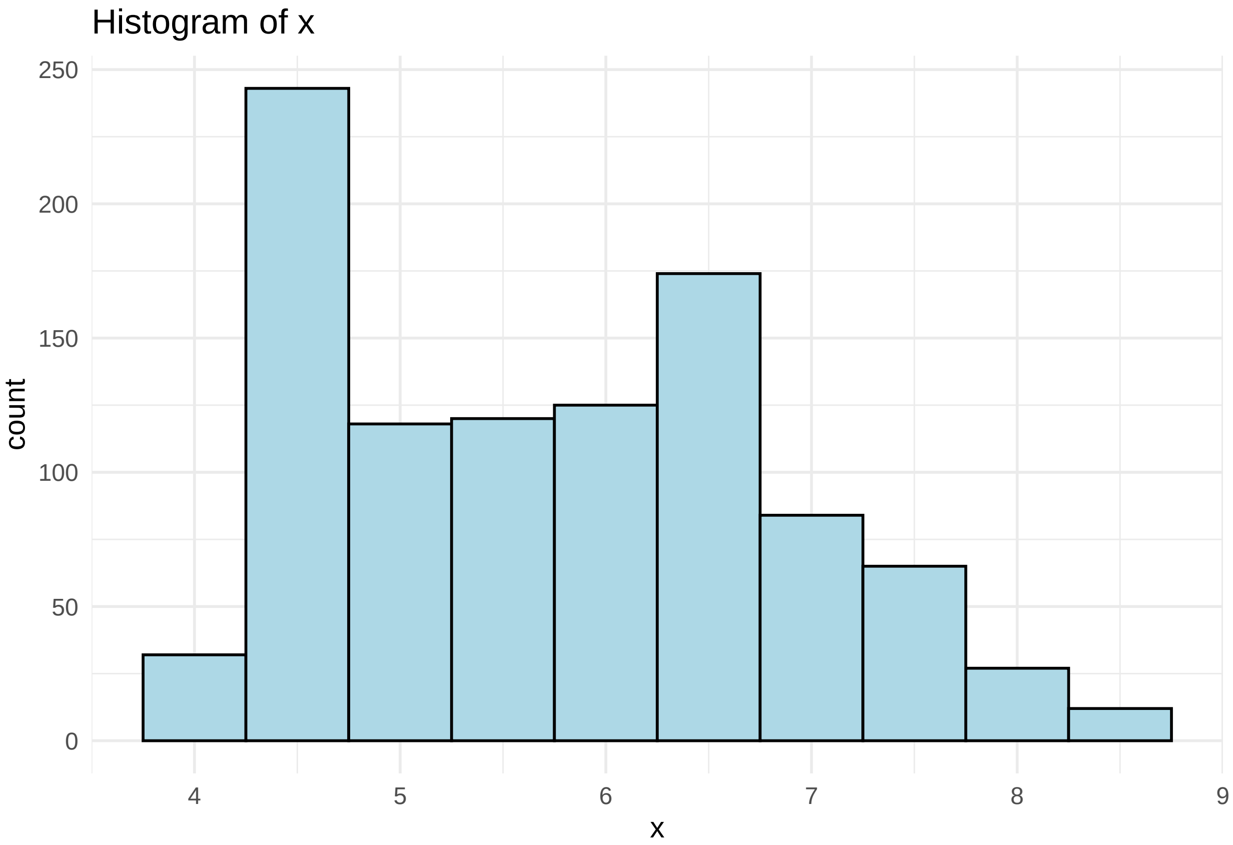
- Histogram for table: The histogram is roughly normally distributed with a peak centered between 56 and 58. The maximum count seems to occur at 56 with a few outliers after 64.

```
ggplot(sample, aes(x = price)) +
  geom_histogram(binwidth = 500, fill = "lightblue", color = "black") +
  ggtitle("Histogram of Price") +
  theme_minimal()
```



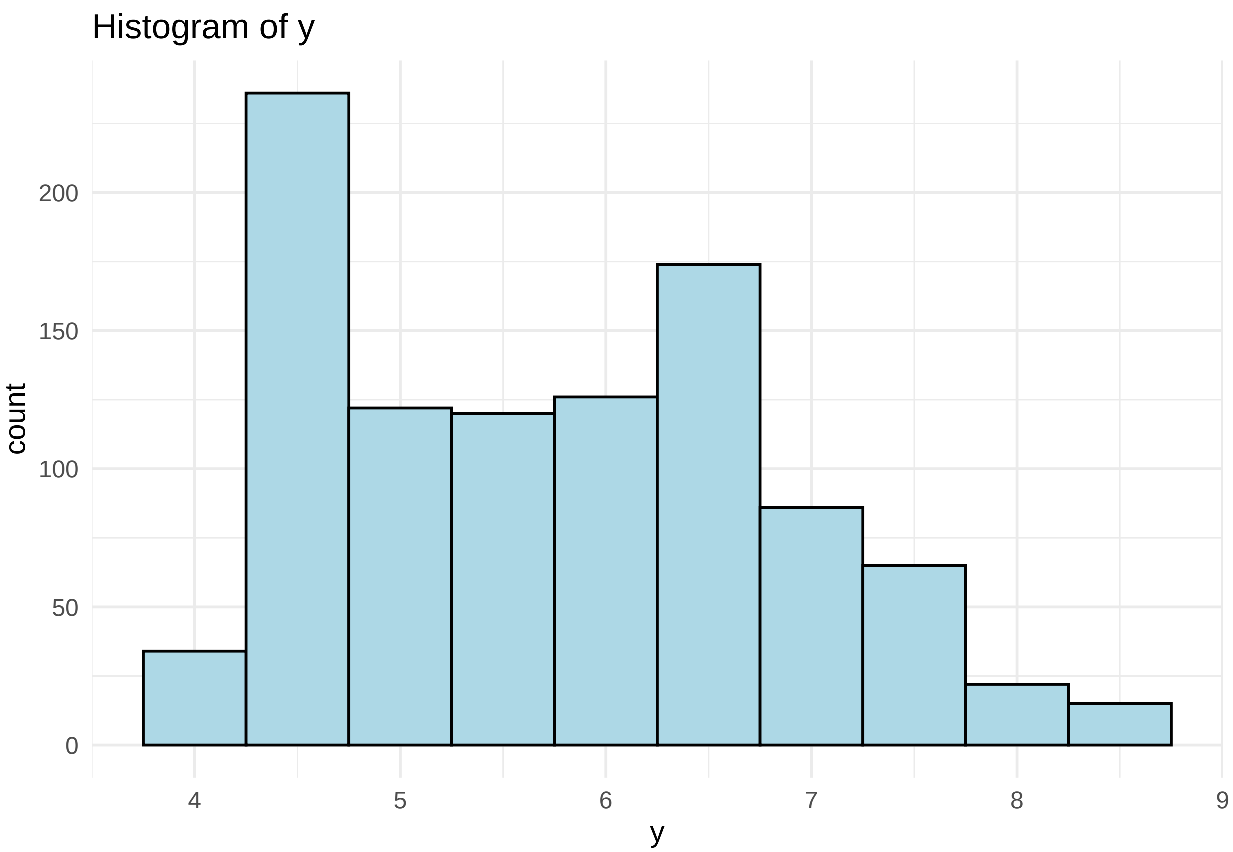
- Histogram for price: The histogram seems to be right skewed with a maximum count of around 170. This means that the price of the diamonds in the sample are mostly of lower price and there are few expensive diamonds that are more than $10000. Clearly, higher-priced diamonds tend to be rarer.

```
# Histogram for x
ggplot(sample, aes(x = x)) +
  geom_histogram(binwidth = .5, fill = "lightblue", color = "black") +
  ggtitle("Histogram of x") +
  theme_minimal()
```



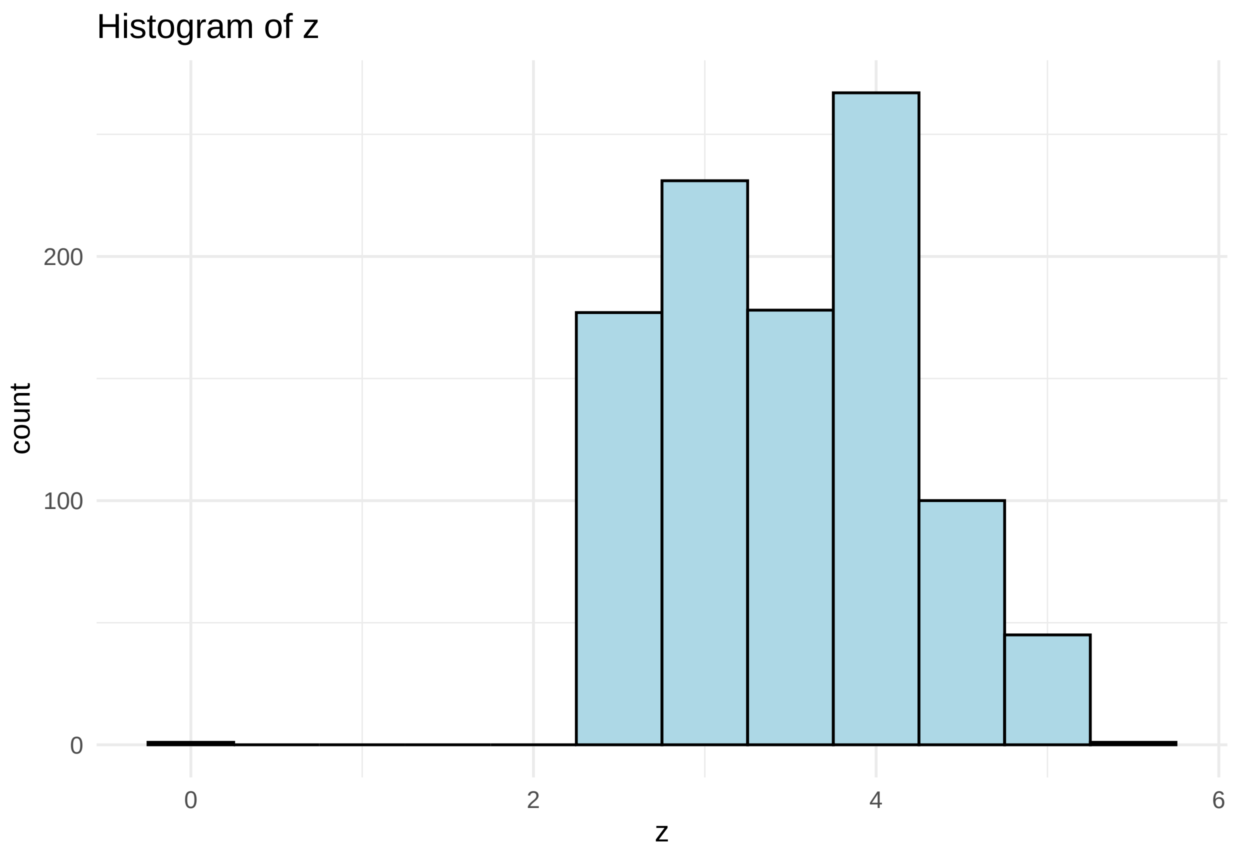
- Histogram for x: The histogram is roughly unimodal and symmetric. It shows a rough bell-like shape, suggesting a near-normal distribution.

```r
# Histogram for y
ggplot(sample, aes(x = y)) +
  geom_histogram(binwidth = .5, fill = "lightblue", color = "black") +
  ggtitle("Histogram of y") +
  theme_minimal()
```



- Histogram for y: The histogram looks nearly identical to that of x.

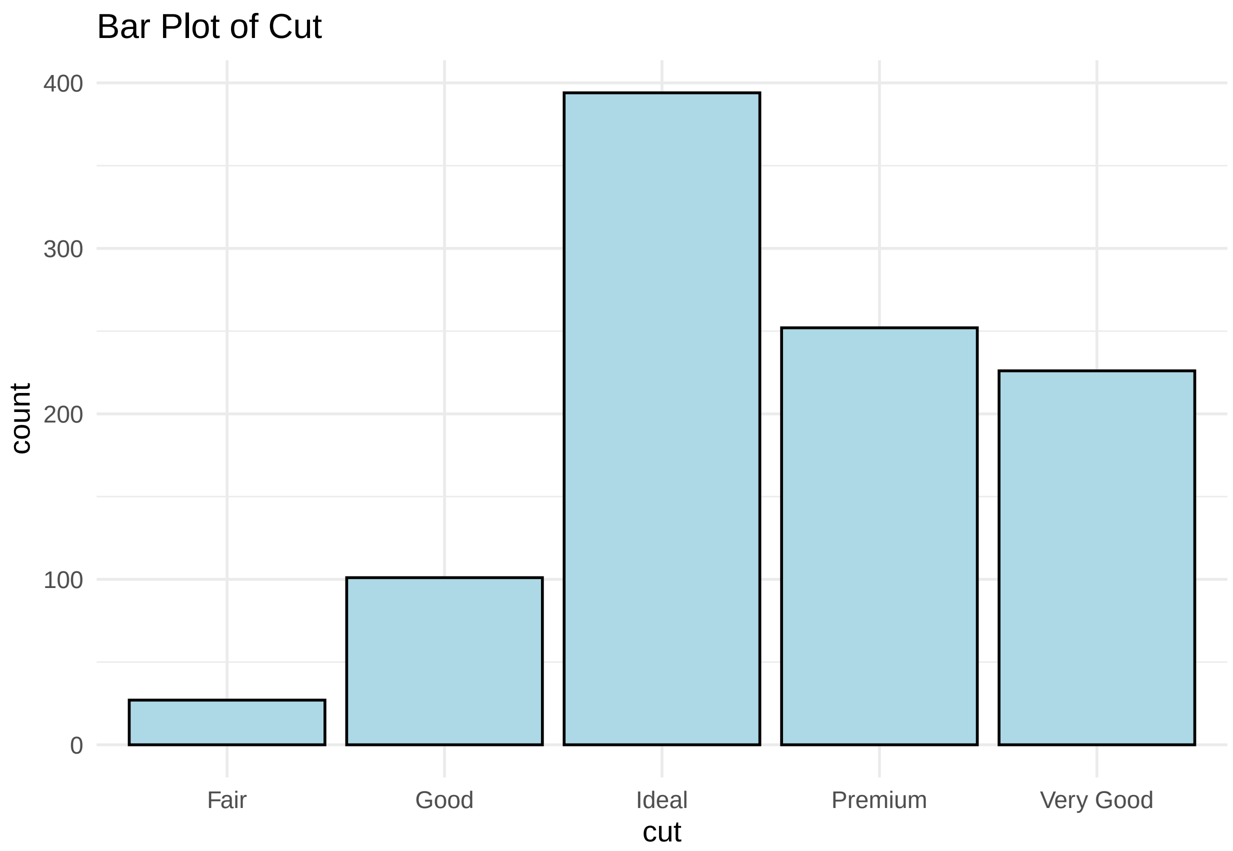
```
# Histogram for z
ggplot(sample, aes(x = z)) +
  geom_histogram(binwidth = 0.5, fill = "lightblue", color = "black") +
  ggtitle("Histogram of z") +
  theme_minimal()
```



- Histogram for z: The histogram is right-skewed, so most values are concentrated on the higher end around 3 to 4. It has a narrower range than x and y, and the skew indicates possible outliers or a non-normal distribution.

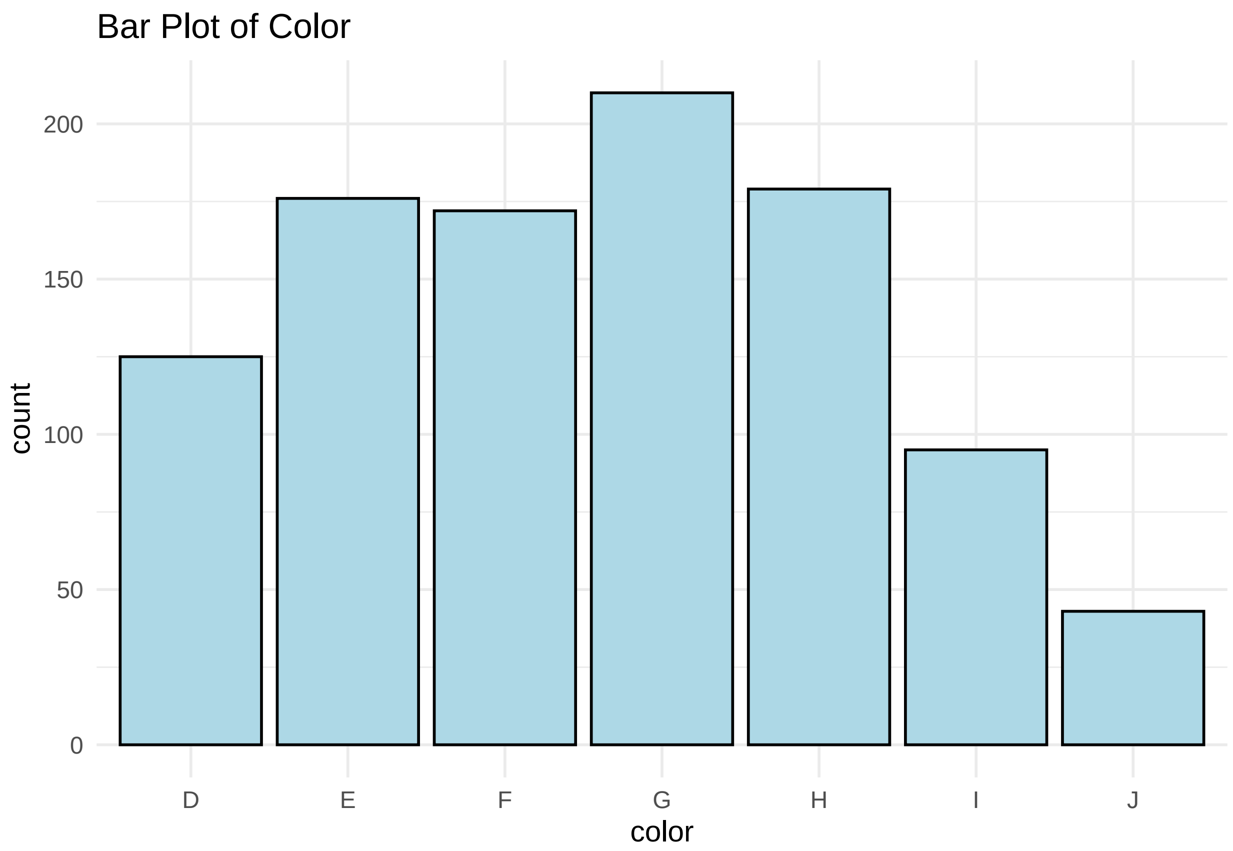
For the categorical random variables, I created bar plots as shown below:

```
ggplot(sample, aes(x = cut)) +
  geom_bar(fill = "lightblue", color = "black") +
  ggtitle("Bar Plot of Cut") +
  theme_minimal()
```



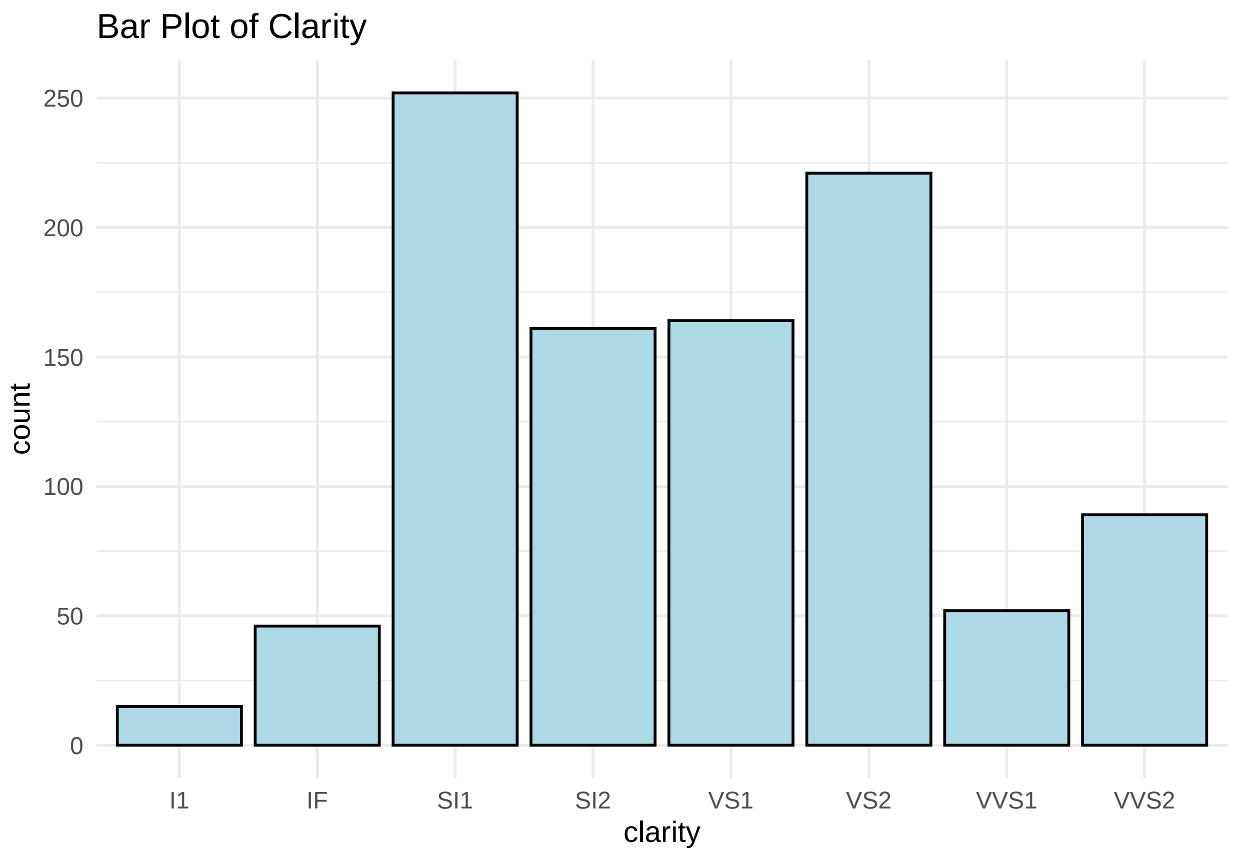
- Bar plot for cut: The bar plot shows that diamonds in the sample mostly have the “Ideal” cut (roughly 400), followed by “Premium” (roughly 250), “Very Good” (roughly 225), “Good” (roughly 100), and, lastly, “Fair” (roughly 25). Therefore, most diamonds are at least “Ideal”.

```
ggplot(sample, aes(x = color)) +
  geom_bar(fill = "lightblue", color = "black") +
  ggtitle("Bar Plot of Color") +
  theme_minimal()
```



- Bar plot for color: The bar plot shows that most diamonds in the sample are color "G" (roughly 210), followed by "H" (roughly 178), "E" (roughly 176), "F" (roughly 173), "D" (roughly 125), "I" (roughly 98), and, lastly, "J" (roughly 47). Therefore, "J" seems to be the rarest color while "G" is the most common.

```
ggplot(sample, aes(x = clarity)) +
  geom_bar(fill = "lightblue", color = "black") +
  ggtitle("Bar Plot of Clarity") +
  theme_minimal()
```



- Bar plot for clarity: The bar plot shows that most diamonds in the sample have a clarity of "SI1" (roughly 250), followed by "VS2" (roughly 223), "VS1" (roughly 168), "SI2" (roughly 166), "VVS2" (roughly 95), "VVS1" (roughly 53), "IF" (roughly 48), and, lastly, "I1" (roughly 20). Therefore, most diamonds in the sample have mid-range clarity grades (SI1, VS2, and VS1), while both high-clarity (VVS1, IF) and low-clarity (I1) diamonds are less common.

### 3

```
# Convert categorical variables into ordered factors
sample <- sample %>%
  mutate(
    cut = factor(cut, levels = c("Fair", "Good", "Very Good", "Premium", "Ideal"), ordered = TRUE),
    color = factor(color, levels = c("J", "I", "H", "G", "F", "E", "D"), ordered = TRUE),
    clarity = factor(clarity, levels = c("I1", "SI2", "SI1", "VS2", "VS1", "VVS2", "VVS1", "IF"), ordere
  )

# Convert the ordered factors to numeric values
sample_numeric <- sample %>%
  mutate(
    cut = as.numeric(cut),
    color = as.numeric(color),
    clarity = as.numeric(clarity)
  ) %>%
  select(carat, cut, color, clarity, depth, price, table, x, y, z)
```

```r
# Calculate correlation
corr <- cor(sample_numeric, use = "complete.obs")
corr
```

```
##              carat         cut       color     clarity        depth
## carat    1.0000000 -0.18711528 -0.23985091 -0.35923738  0.052779405
## cut     -0.1871153  1.00000000 -0.04595169  0.21820270 -0.284708238
## color   -0.2398509 -0.04595169  1.00000000 -0.06583316 -0.066508986
## clarity -0.3592374  0.21820270 -0.06583316  1.00000000 -0.107197082
## depth    0.0527794 -0.28470824 -0.06650899 -0.10719708  1.000000000
## price    0.9233404 -0.11218809 -0.13228989 -0.16364385  0.009338803
## table    0.2114770 -0.44624933  0.03497772 -0.16076067 -0.256659487
## x        0.9807258 -0.18031642 -0.21432036 -0.38613724  0.004846103
## y        0.9808579 -0.18218133 -0.21390996 -0.38018245  0.001814092
## z        0.9644749 -0.21252038 -0.21262804 -0.37737913  0.137704234
##                price       table            x            y         z
## carat    0.923340431  0.21147702  0.980725752  0.980857857  0.9644749
## cut     -0.112188088 -0.44624933 -0.180316420 -0.182181332 -0.2125204
## color   -0.132289888  0.03497772 -0.214320360 -0.213909964 -0.2126280
## clarity -0.163643852 -0.16076067 -0.386137242 -0.380182451 -0.3773791
## depth    0.009338803 -0.25665949  0.004846103  0.001814092  0.1377042
## price    1.000000000  0.14760919  0.881271345  0.884552920  0.8669840
## table    0.147609194  1.00000000  0.233127900  0.230644626  0.1932476
## x        0.881271345  0.23312790  1.000000000  0.998950088  0.9747044
## y        0.884552920  0.23064463  0.998950088  1.000000000  0.9746257
## z        0.866983958  0.19324763  0.974704370  0.974625715  1.0000000
```

To determine the correlation between the variables in the dataset, I computed the Pearson correlation matrix. To accurately capture the ordinal nature of categorical variables cut, color, and clarity, these were converted into ordered factors with appropriate levels before numeric transformation. Since correlation requires numeric data, I then converted the categorical variables cut, color, and clarity into numeric codes representing their ordinal levels.

- Key Observations: Diamond size variables, carat, and its dimensions (x, y, z), are strongly positively correlated with price with correlations above 0.86, indicating that larger diamonds tend to be more expensive. Interestingly, carat also shows strong positive correlations with x, y, and z with correlations above 0.96, confirming that these physical measurements consistently reflect diamond size. On the other hand, quality-related variables such as cut, color, and clarity exhibit weak to moderate negative correlations with carat, price, and the dimensions, suggesting that higher quality grades tend to correspond to smaller diamonds and lower prices in this dataset. Specifically, clarity shows a moderate negative correlation of -0.36 with carat and of -0.38 with the dimensions, implying that diamonds with better clarity might be smaller. Interestingly, cut and color show weak negative correlations with price (-0.11 and -0.13 respectively), suggesting that in this dataset, better cut and color grades do not linearly increase price as might be expected. Depth and table percentages show very weak or small negative correlations with other variables, with table notably negatively correlated with cut, suggesting that diamonds with better cut grades tend to have narrower tables. Overall, diamond size has a strong influence on price, while quality factors like cut, color, and clarity show weaker, and sometimes even opposite, linear relationships.

# 4

```r
model1 <- lm(price ~ carat + depth + cut + color + clarity + table + x + y + z, data = sample)
summary(model1)
```

```
##
## Call:
## lm(formula = price ~ carat + depth + cut + color + clarity +
##     table + x + y + z, data = sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4375.3  -563.2  -177.1   392.9  6463.4
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11524.13    3088.12   3.732 0.000201 ***
## carat       14152.49     405.88  34.869  < 2e-16 ***
## depth         -79.38      36.78  -2.158 0.031153 *
## cut.L         372.63     180.29   2.067 0.039010 *
## cut.Q        -143.28     143.18  -1.001 0.317206
## cut.C         149.42     119.76   1.248 0.212481
## cut^4         110.85      89.19   1.243 0.214250
## color.L      2086.61     127.53  16.361  < 2e-16 ***
## color.Q      -684.50     117.01  -5.850 6.71e-09 ***
## color.C        98.04     107.07   0.916 0.360059
## color^4       114.25      97.56   1.171 0.241842
## color^5        99.18      88.97   1.115 0.265229
## color^6        13.07      80.38   0.163 0.870858
## clarity.L    3234.93     215.41  15.018  < 2e-16 ***
## clarity.Q   -1846.17     197.36  -9.354  < 2e-16 ***
## clarity.C     429.56     171.69   2.502 0.012517 *
## clarity^4     -76.14     141.14  -0.539 0.589702
## clarity^5      71.68     116.29   0.616 0.537754
## clarity^6     -30.74      99.38  -0.309 0.757148
## clarity^7     105.79      84.73   1.249 0.212138
## table         -38.56      20.96  -1.840 0.066077 .
## x           -2845.46     809.25  -3.516 0.000458 ***
## y            1003.05     810.83   1.237 0.216363
## z            -476.83     288.48  -1.653 0.098675 .
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1070 on 976 degrees of freedom
## Multiple R-squared:  0.9294, Adjusted R-squared:  0.9278
## F-statistic:   559 on 23 and 976 DF,  p-value: < 2.2e-16
```

The multiple linear regression model was used to examine how various diamond attributes influence price. After adjusting for the number of predictors, the model explains approximately 92.78% of the variance in diamond price, indicating a very strong overall fit. Among the predictors, carat has a p-value well below the 0.05 significance level, allowing us to reject the null hypothesis that carat has no effect on price. This confirms that larger diamonds tend to be substantially more expensive. Similarly, variables such as depth and x also have significant p-values, indicating they meaningfully contribute to explaining price. In contrast, predictors such as table, y, and z have p-values greater than the 0.05 significance level, indicating that their individual effects on price are not statistically significant in the presence of other variables. The model also shows that color, cut, and clarity have statistically significant overall effects on price. However, not all individual levels within color, cut, and clarity are significant, indicating the influence may be more general. The residual standard error is 1,070, representing the typical deviation of predicted prices from actual prices. Additionally, the very low p-value for the F-statistic allows us to reject the null hypothesis that all coefficients are zero, confirming that the model is statistically significant overall.

### 5

As anticipated, carat size was found to be the strongest driver of price, with both the correlation matrix and regression model showing a very strong positive relationship. This is consistent with the well-known fact that larger diamonds usually have higher prices. However, it was somewhat surprising that cut did not show a significant linear effect on price in the multiple regression model, especially given how cut is emphasized in diamond grading and marketing. Another interesting observation was the weak or even negative linear relationships between quality measures (cut, color, clarity) and price. While better clarity and color grades were associated with higher prices overall, the effects were not strong across all levels, and some individual levels were not significant. This complexity suggests that while quality matters, its pricing impact may not be linear, and it may interact with other attributes like size. Overall, the general pricing patterns made sense, but how quality features like cut influenced price was less straightforward than expected. The weak or inconsistent impact of cut, in particular, shows that diamond pricing in the real world can be more nuanced than simple grading scales suggest.

## Part 2: Simple Linear Regression

### 1

To explore the relationship between diamond size and price, I conducted a simple linear regression using carat as the predictor and price as the response variable.

```r
# price ~ carat
simple_model <- lm(price ~ carat, data = sample)
summary(simple_model)
```

```
##
## Call:
## lm(formula = price ~ carat, data = sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5932.7  -905.1     0.8   631.1  8572.6
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2456.41      97.05  -25.31   <2e-16 ***
## carat        8028.56     105.69   75.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1529 on 998 degrees of freedom
## Multiple R-squared:  0.8526, Adjusted R-squared:  0.8524
## F-statistic:  5771 on 1 and 998 DF,  p-value: < 2.2e-16
```

## 2

After running the model above, I examined the summary statistics, and the goal was to test the hypothesis that carat has a statistically significant effect on diamond price. In this context, the null hypothesis states that carat has no effect on price. The alternative hypothesis states that carat does affect price. The results indicate a strong and statistically significant relationship between these two variables, as shown by an F-statistic of 5771 and a p-value less than 2.2e-16, which is far below the typical 0.05 significance level. Therefore, we can reject the null hypothesis that carat has no effect on price.

The model suggests that for every additional carat, the price of a diamond increases by approximately $8,029, holding everything else constant. The estimate varies by about $105.69 across samples and is statistically significant, as indicated by the three stars, leading us to reject the null hypothesis. The intercept, while not meaningful in a practical sense, predicting a negative price at 0 carats, also has a statistically significant estimate of –$2,456.41 with a standard error of $97.05. The R-squared value of 0.8526 indicates that over 85% of the variation in diamond price can be explained by carat alone, proving that diamond size is a major predictor of price. Similarly, the adjusted R-squared of 0.8524 confirms this explanatory power, even after adjusting for the number of parameters in the model. Additionally, the residual standard error of approximately $1,529 means that the typical prediction from this model deviates from the actual price by that amount. Overall, this analysis confirms that carat is a statistically significant predictor of diamond price, although some variability remains that could be accounted for by additional characteristics such as cut, color, and clarity.

Now, I will calculate both the confidence interval for the expected price and the prediction interval for the predicted price of a new diamond using carat as the predictor. Based on the output of summary() and str() from Part 1, I observed that the range of carat values in the sample spans from 0.23 to 2.32, and the mean carat is approximately 0.7962. Therefore, I will use this mean value to compute both intervals.

```r
new_data <- data.frame(carat = 0.7962)

# Confidence interval (mean response)
predict(simple_model, newdata = new_data, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 3935.932 3841.055 4030.809
```

```r
# Prediction interval (individual prediction)
predict(simple_model, newdata = new_data, interval = "prediction")
```

```
##        fit      lwr      upr
## 1 3935.932 934.1527 6937.712
```

The confidence interval for the mean price is [$3841.06, $4030.81]. This means we are 95% confident that the average price of all diamonds with a carat of 0.7962 falls within this range.

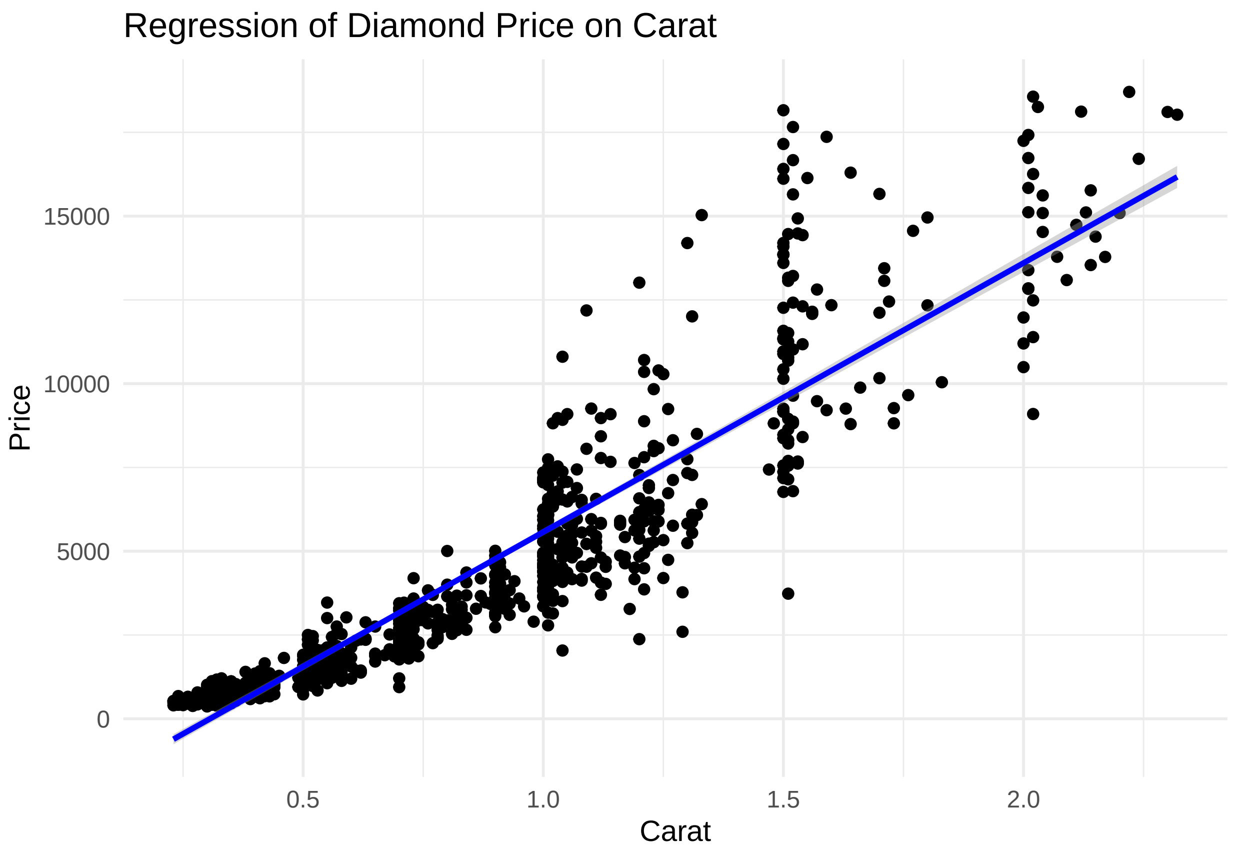
The prediction interval for the price of a new diamond is [$934.15, $6937.71]. This much wider interval reflects the uncertainty in predicting the price of an individual diamond, accounting for natural variability beyond just estimation error.

The narrow confidence interval for the mean response suggests that carat is a strong predictor of average diamond price. However, the wide prediction interval shows that carat alone does not fully explain individual price variability. This indicates that while carat significantly influences price, other variables such as cut, color, and clarity likely play an important role.

Next, I will create a scatterplot of diamond price against carat with a regression line and a 95% confidence band as shown below:

```r
# Create a scatterplot with regression line
ggplot(sample, aes(x = carat, y = price)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue", se = TRUE) +
  ggtitle("Regression of Diamond Price on Carat") +
  xlab("Carat") +
  ylab("Price") +
  theme_minimal()
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

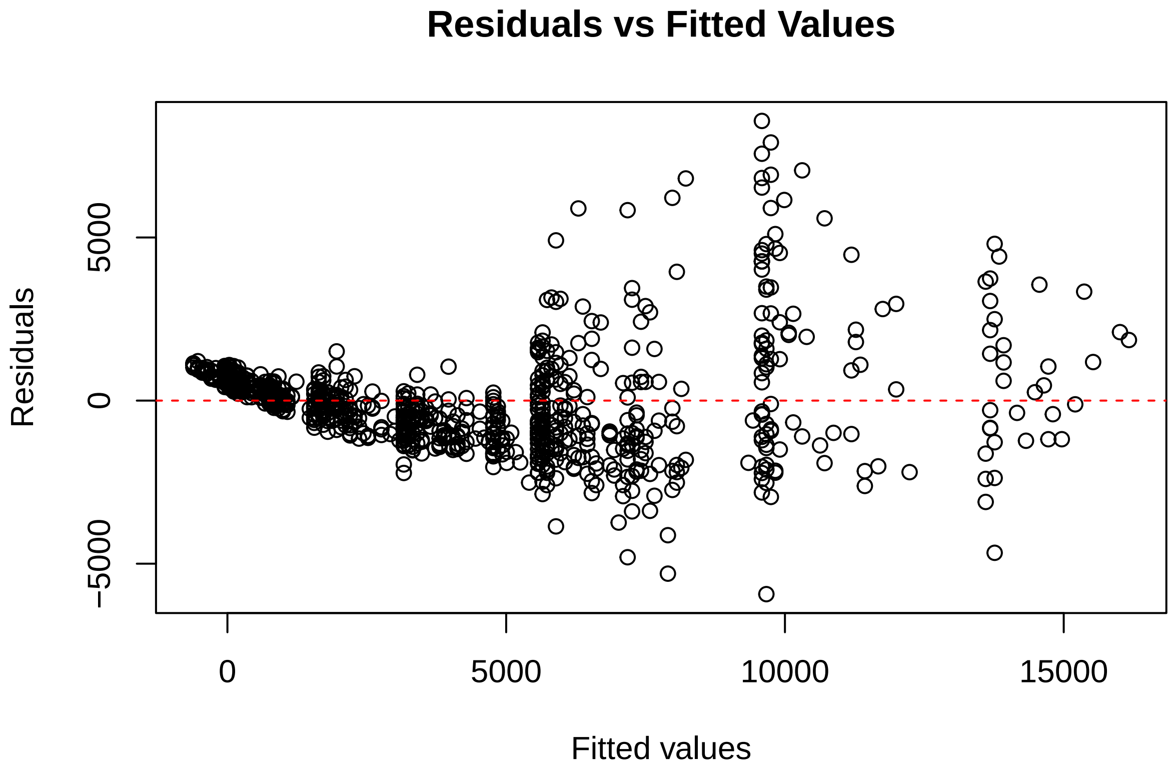


The plot confirms a strong positive linear trend, with a narrow confidence band around the regression line, indicating consistent price across different carat values. The tightness of this band further supports the model’s reliability in predicting average diamond prices based on carat. However, the scatterplot shows some variance of data points around the regression line. This suggests that while carat is a strong predictor of price, other factors also influence diamond price, as stated above.

### 3

**Checking Linearity and Homoscedasticity:**

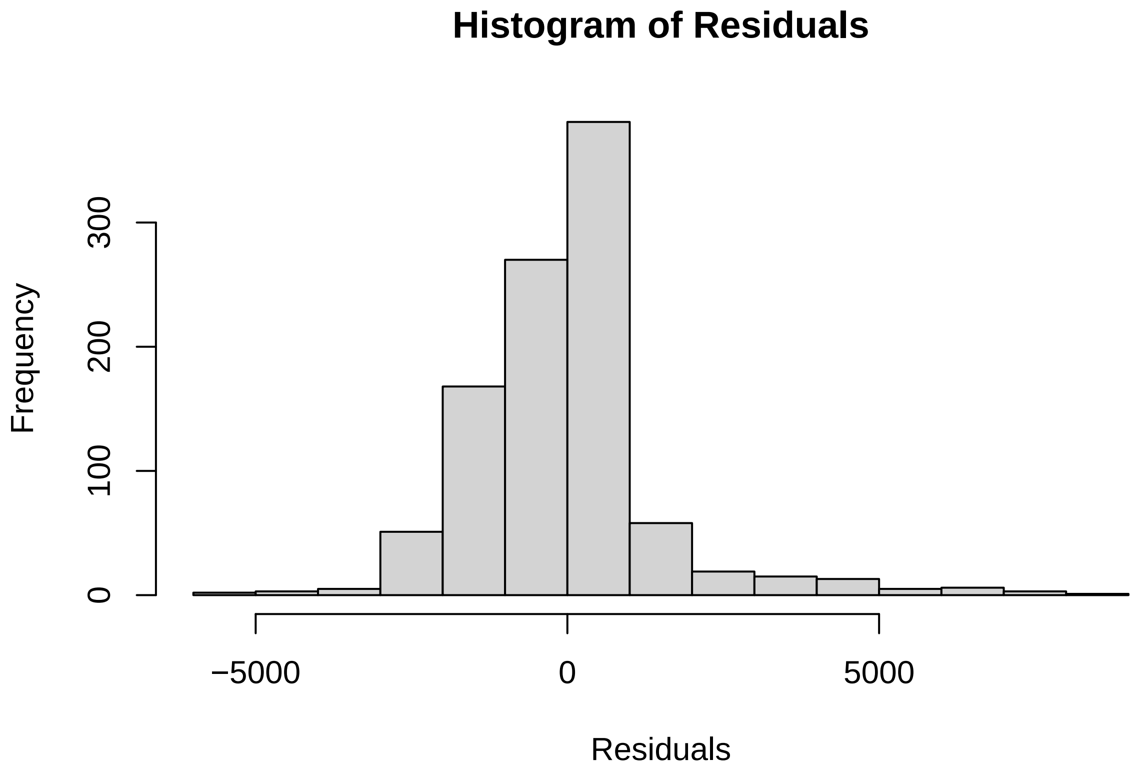
```
plot(simple_model$fitted.values, simple_model$residuals,
     xlab = "Fitted values",
     ylab = "Residuals",
     main = "Residuals vs Fitted Values")
abline(h = 0, col = "red", lty = 2)
```



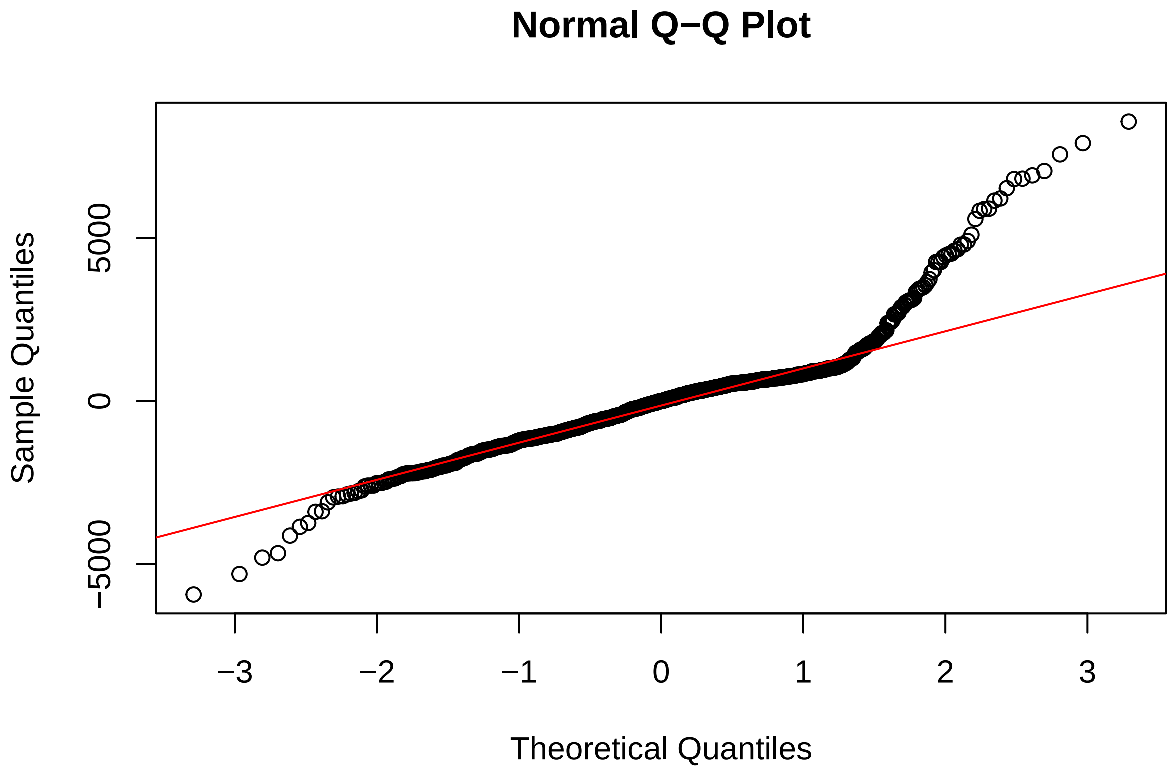
Looking at the plot of residuals vs. fitted values, we can see that the residuals seem to fan out and show some curvature, rather than being randomly scattered around 0. This suggests that the linearity assumption may be violated and there most likely is heteroscedasticity.

**Checking Normality of Residuals:**

```
hist(simple_model$residuals, main = "Histogram of Residuals", xlab = "Residuals")
```



```r
qqnorm(simple_model$residuals, main = "Normal Q-Q Plot")
qqline(simple_model$residuals, col = "red")
```



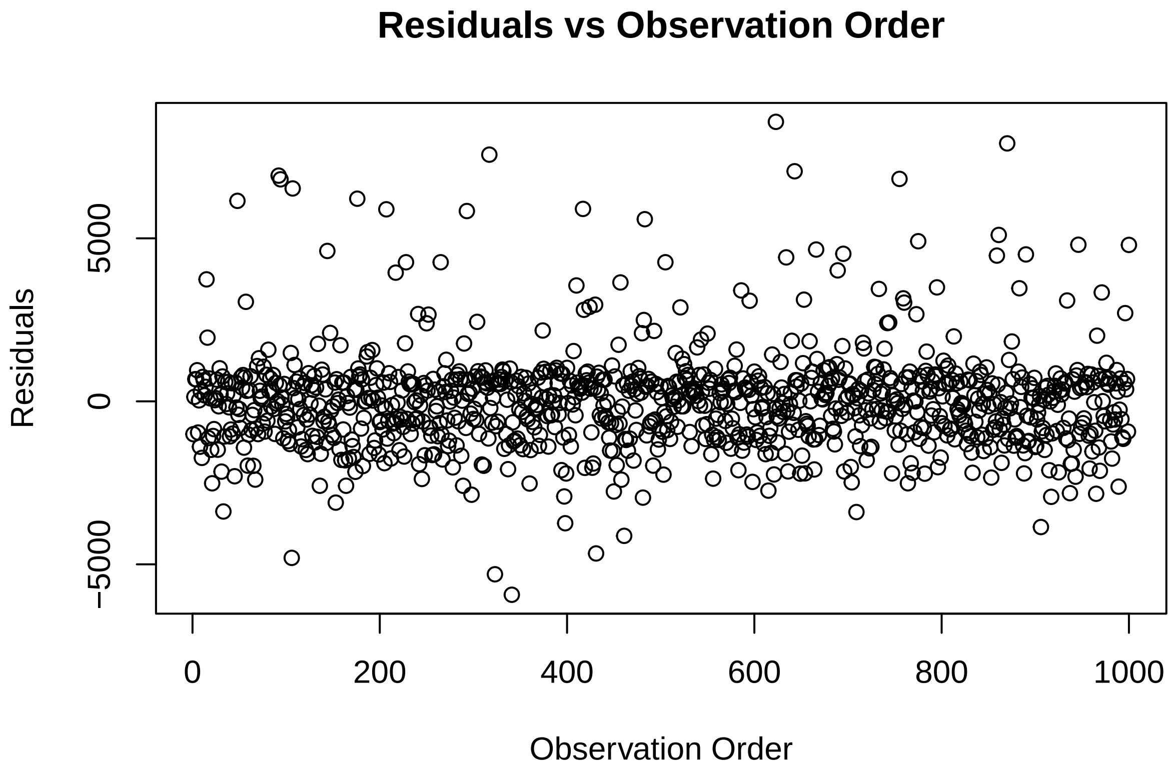
```
shapiro.test(simple_model$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  simple_model$residuals
## W = 0.88958, p-value < 2.2e-16
```

The histogram appears roughly bell-shaped, but show some skewness or outliers. To confirm, I created a Q-Q plot, which shows a noticeable deviation from the red line at both ends. This suggests the residuals are not normally distributed, especially in the tails. The Shapiro-Wilk test outputs a p-value of less than 2.2e-16, which is less that the significance level of 0.05; therefore, the null hypothesis of normality is rejected, confirming non-normal residuals.

**Checking Independence of Errors:**

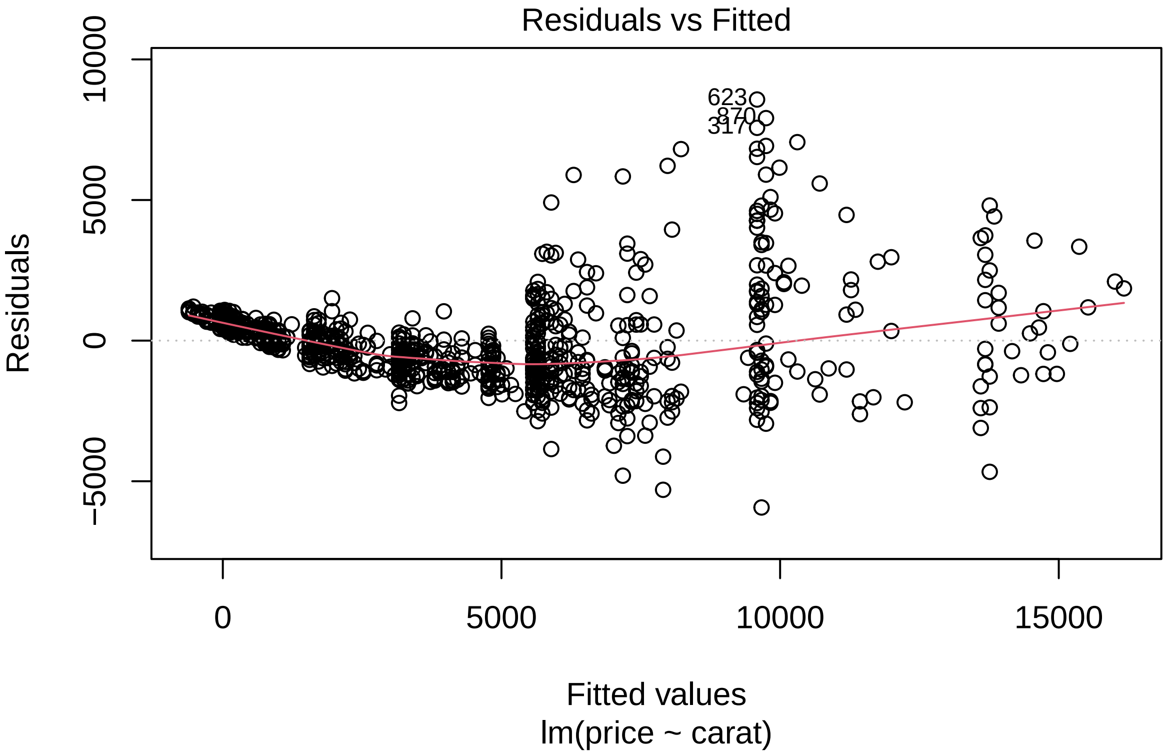
```
time <- 1:length(simple_model$residuals)
plot(time, simple_model$residuals,
     xlab = "Observation Order",
     ylab = "Residuals",
     main = "Residuals vs Observation Order")
```

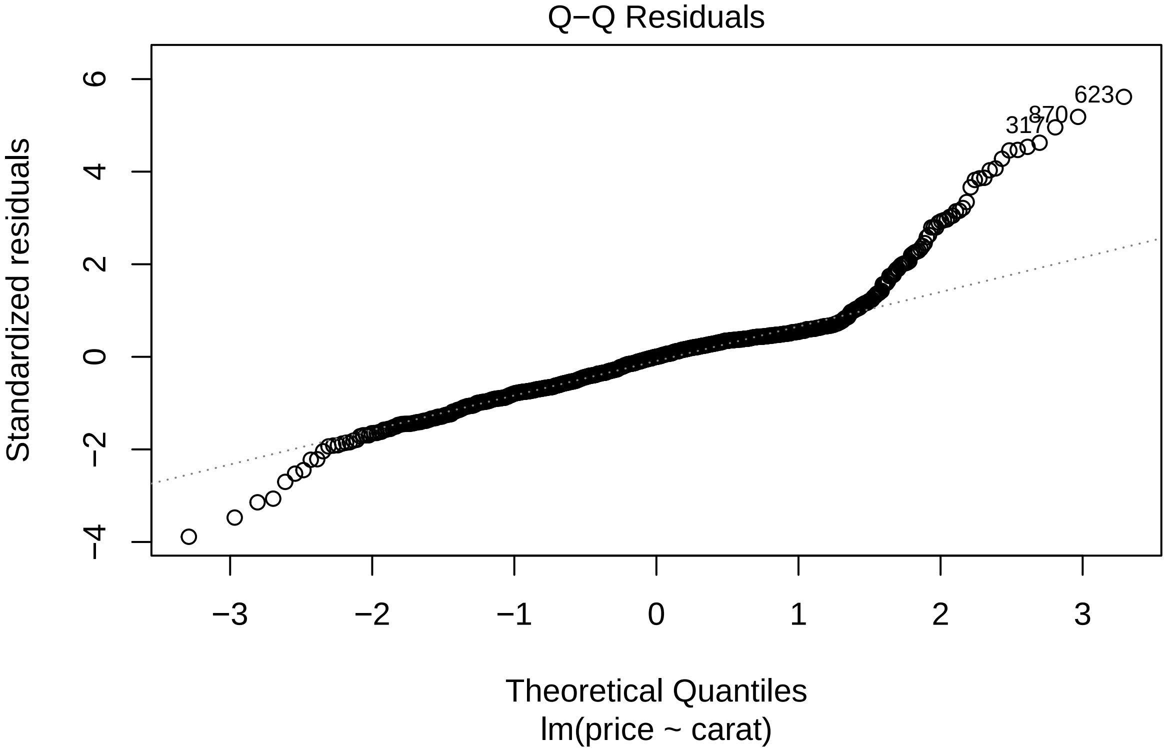


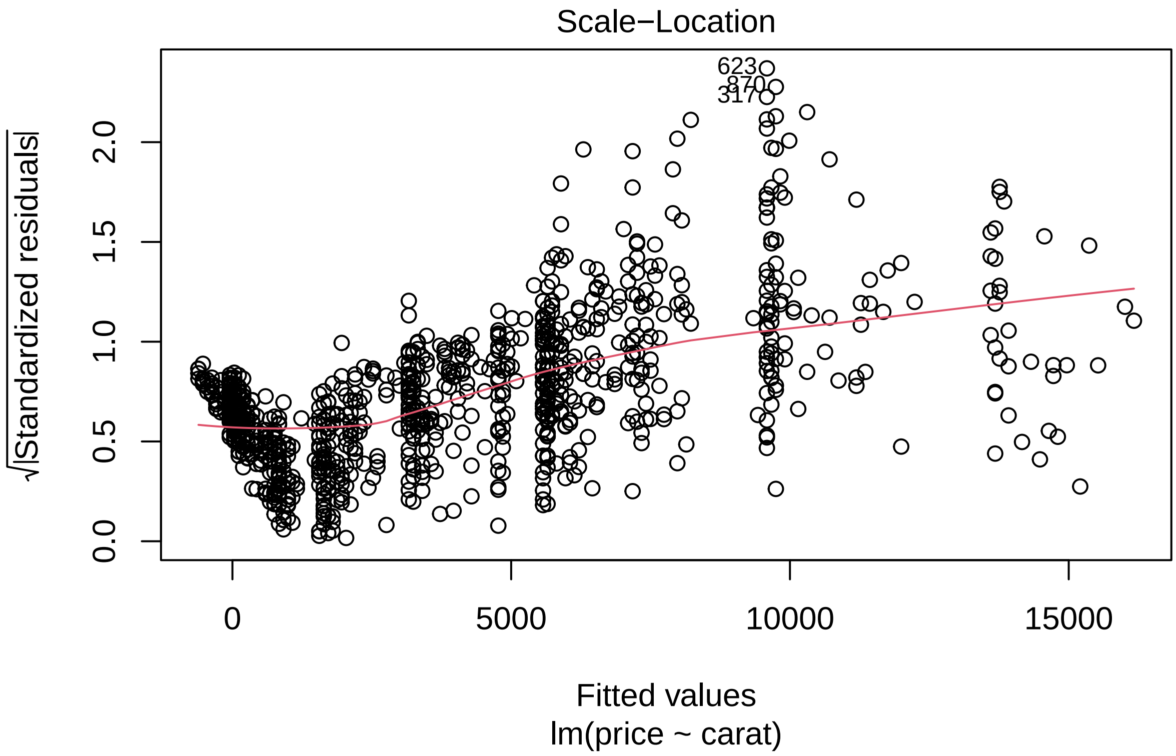
The residuals are fairly randomly scattered across the observation order. Therefore, the assumption of independent residuals appears to be met.

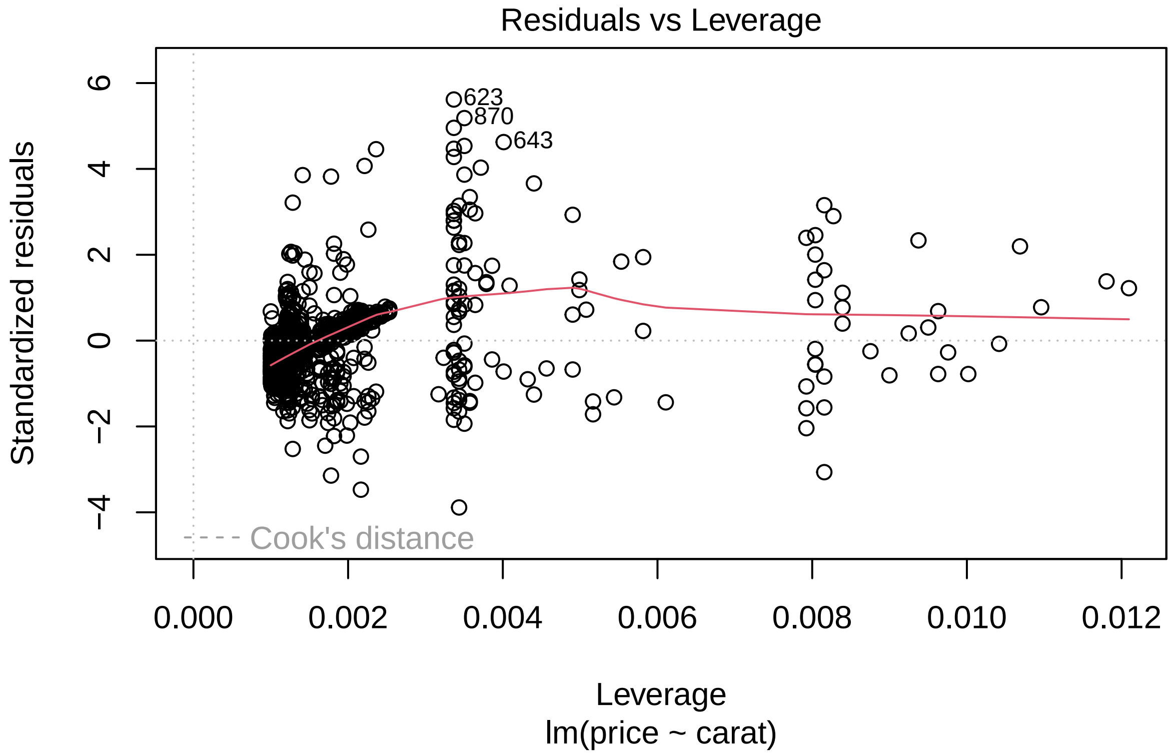
An alternative way to view the diagnostic plots is:

```
plot(simple_model)
```


Residuals vs Fitted
Residuals
10000
5000
0
−5000
623
870
317
0
5000
10000
15000
Fitted values
lm(price ~ carat)


Q−Q Residuals
Standardized residuals
6
4
2
0
−2
−4
−3
−2
−1
0
1
2
3
623
870
317
Theoretical Quantiles
lm(price ~ carat)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
0
5000
10000
15000
Fitted values
lm(price ~ carat)
623
870
317

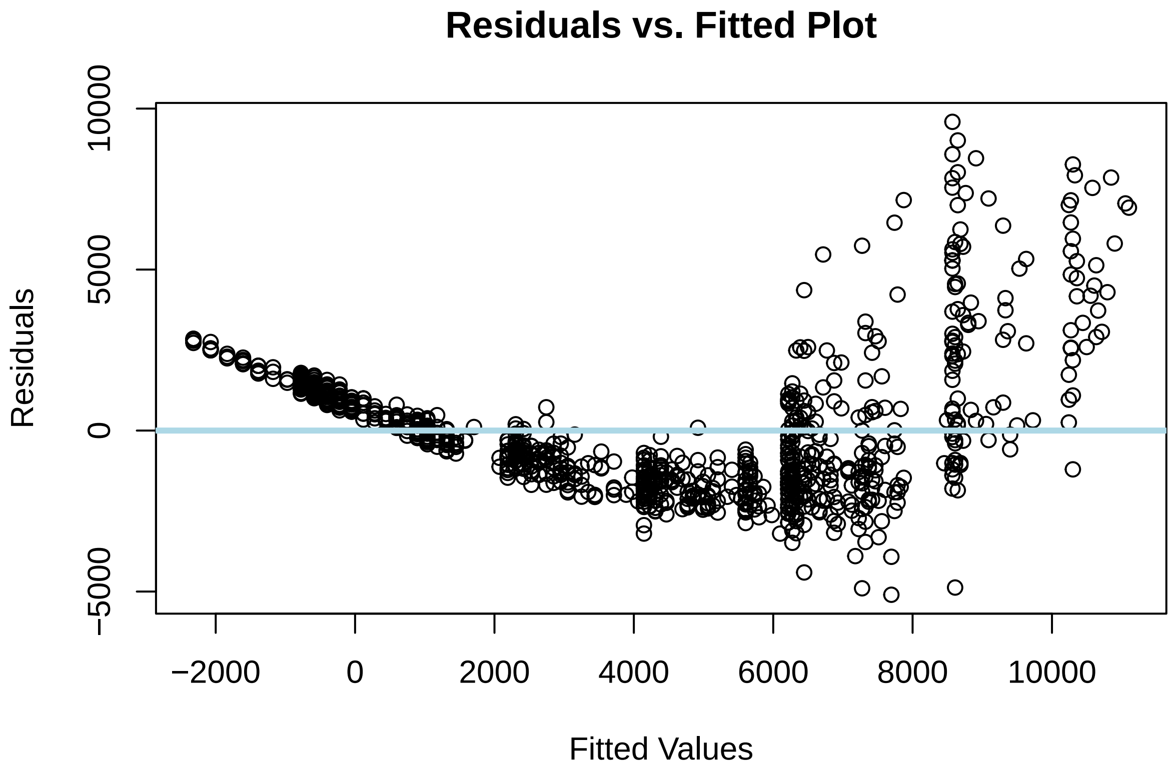


**Transformations:**

After reviewing the model diagnostics, I identified that the assumptions of linearity, homoscedasticity, and normality of residuals are not met. To address these issues, I applied log transformations to the response variable (price) and the predictor (carat).

- Log Transformation of Independent Variable: I first log-transformed the independent variable to try to correct heteroscedasticity.

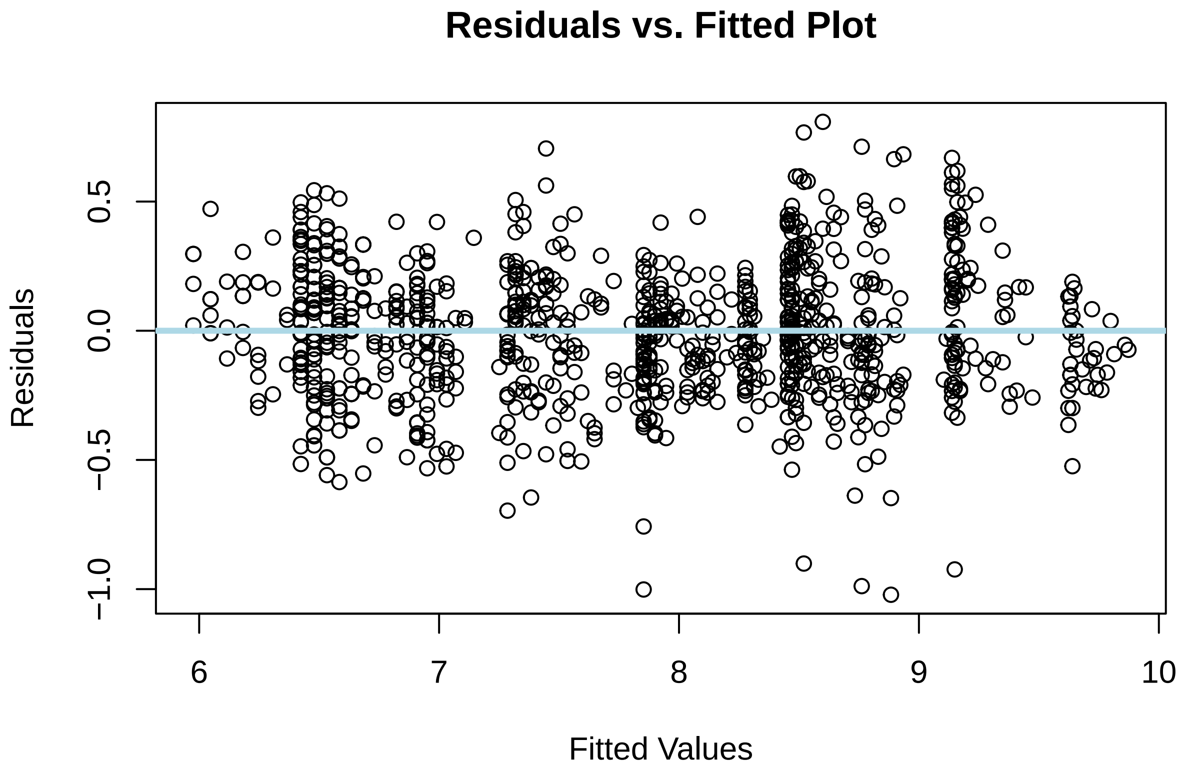
```
sample$log_carat <- log(sample$carat)  # Log transformation of carat
model <- lm(price ~ log_carat, data = sample)  # Fit new model

# Residuals vs. Fitted Plot
plot(model$fitted.values, residuals(model),
     xlab = "Fitted Values",
     ylab = "Residuals",
     main = "Residuals vs. Fitted Plot")
abline(h = 0, col = "lightblue", lwd = 3)
```



The residuals still displayed a pattern, indicating that the assumptions were not yet satisfied.

- Log Transformation of Dependent Variable: I also log-transformed the dependent variable to address the remaining curvature in the residuals.

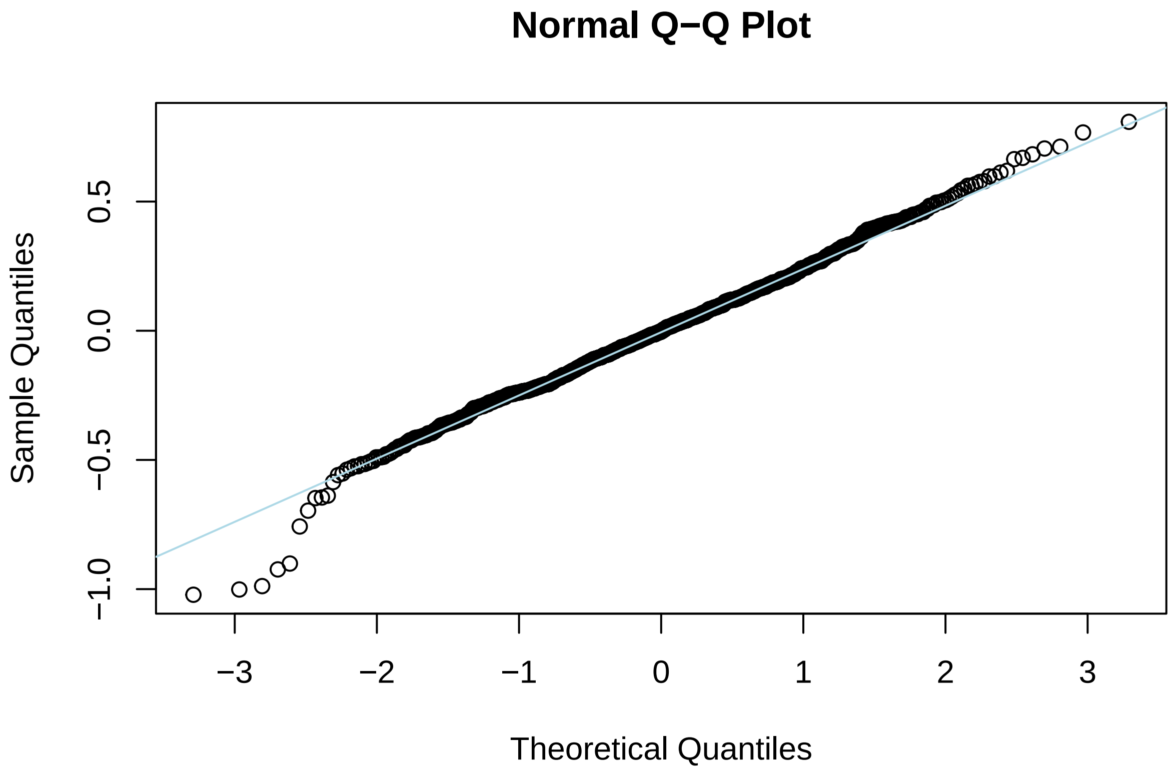
```
sample$log_price <- log(sample$price)  # Log transformation of price
model1 <- lm(log_price ~ log_carat, data = sample)  # Fit new model

# Residuals vs. Fitted Plot
plot(model1$fitted.values, residuals(model1),
     xlab = "Fitted Values",
     ylab = "Residuals",
     main = "Residuals vs. Fitted Plot")
abline(h = 0, col = "lightblue", lwd = 3)
```



After this second transformation, the residuals vs. fitted plot no longer shows a structured pattern or evidence of non-constant variance. This suggests that both the linearity and homoscedasticity assumptions are now met.

- Rechecking Normality of Residuals: To check whether the normality assumption is satisfied, I examined a Q-Q plot of the residuals using the transformed model.

```
qqnorm(residuals(model1))
qqline(residuals(model1), col = "lightblue")
```



The points closely follow the reference line, indicating that the residuals are approximately normally distributed.

All in all, by applying log transformations to both price and carat, I improved the model fit and satisfied key regression assumptions, including linearity, homoscedasticity, and normality of residuals. The assumption of independence was likely met throughout.

## 4

```r
summary(model1)
```

```
##
## Call:
## lm(formula = log_price ~ log_carat, data = sample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.02178 -0.17069 -0.00159  0.15935  0.80854
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.454101   0.009627   878.1   <2e-16 ***
## log_carat   1.686313   0.013727   122.8   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

```
##
## Residual standard error: 0.2523 on 998 degrees of freedom
## Multiple R-squared:  0.938,  Adjusted R-squared:  0.9379
## F-statistic: 1.509e+04 on 1 and 998 DF,  p-value: < 2.2e-16
```

After transforming both price and carat using logarithms, I fit a new linear model and evaluated its summary statistics. The residuals in this transformed model are much more tightly centered around zero compared to the original model, suggesting that the assumption of normality is now better satisfied. The intercept of 8.454 represents the expected log-price when log-carat is zero. The slope coefficient of 1.686 indicates that for every 1% increase in carat, the price increases by approximately 1.686% on average. Both coefficients are highly statistically significant, with p-values well below the 0.05 significance level; therefore, there is strong evidence of a meaningful relationship between log-transformed carat and log-transformed price. Model performance also improved. The R-squared value increased from 0.8526 in the original model to 0.938, meaning over 93% of the variability in log-price is now explained by log-carat. This high explanatory power, along with an adjusted R-squared that matches closely, suggests the model fits the data well without overfitting. Additionally, the residual standard error decreased to 0.2523, and the F-statistic rose dramatically to 15,090, further confirming the model's strength.

## 5

To further improve the model, I explored whether adding additional variables beyond carat would enhance model performance. Each variable was added individually to the log-log regression model, and the impact on Adjusted R-squared was assessed. The baseline Adjusted R-squared was approximately 0.938, as calculated in the previous step, which served as the benchmark to evaluate whether each variable improved the model.

- After adding depth, I observed a slight increase in the Adjusted R-squared to 0.9392, indicating that depth does provide additional explanatory power beyond log_carat. Therefore, depth was considered a useful predictor.
- Next, I added cut, which further improved the Adjusted R-squared to 0.9424. This suggests that cut meaningfully contributes to explaining price.
- Adding color increased the Adjusted R-squared to 0.9480, showing it also adds important explanatory information.
- When clarity was included, the model showed a substantial improvement, with the Adjusted R-squared increasing to 0.9672. This indicates clarity is a very strong predictor of diamond price.
- Adding table resulted in only a very slight increase in Adjusted R-squared to 0.9384 when added alone with log_carat. While this increase is lower than some other variables, it still increased the adjusted R-squared.
- Again, adding x resulted in only a very slight increase in Adjusted R-squared to 0.9387 when added alone with log_carat. While this increase is lower than some other variables, it still increased the adjusted R-squared.
- After adding y, I observed a slight increase in the Adjusted R-squared to 0.939, indicating that y does provide additional explanatory power beyond log_carat.
- Finally, adding z resulted in a slight decrease in Adjusted R-squared to 0.937 when added alone with log_carat; therefore, it should be excluded.

### 6

One of the most interesting parts of this analysis was seeing how dramatically the model improved after applying log transformations. Initially, the linear model violated the linear regression assumptions, but after the log-log transformation, all assumptions were satisfied. I realized the power of data transformation in improving model performance. The simple linear regression analysis revealed a strong and statistically significant relationship between diamond carat and price, with carat alone explaining over 85% of the variation in price. After the log transformations were applied to both price and carat, the model greatly improved, explaining about 93.8% of the variability in log-transformed price.

Another surprising finding was just how much explanatory power clarity added to the model, far greater than I expected. This emphasized the importance of diamond quality traits beyond size, and how nuanced factors like cut and clarity can significantly affect price. I also gained a deeper understanding of the distinction between confidence intervals and prediction intervals. Confidence intervals tend to be narrower because they estimate the range in which the mean response is likely to fall for a given set of predictor values. In contrast, prediction intervals are wider, as they account for the additional variability in predicting individual outcomes. Lastly, working step by step to add one variable at a time and observing how it affects the Adjusted R-squared value was a valuable exercise in model selection.

## Part 3

### 1

Now, I will apply backward elimination using AIC to identify the best model. I will use the log-log transformed model (log_price ~ log_carat + . . . ) because, as demonstrated earlier, the original model violated some regression assumptions.

```r
full_model <- lm(log_price ~ log_carat + depth + cut + color + clarity + table + x + y + z, data = sampl
summary(full_model)
```

```
##
## Call:
## lm(formula = log_price ~ log_carat + depth + cut + color + clarity +
##     table + x + y + z, data = sample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43190 -0.08448  0.00248  0.08782  0.37819
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.915551   0.523028  11.310  < 2e-16 ***
## log_carat    1.459959   0.072991  20.002  < 2e-16 ***
## depth        0.014725   0.004697   3.135 0.001771 **
## cut.L        0.135637   0.021288   6.372 2.88e-10 ***
## cut.Q       -0.049156   0.016902  -2.908 0.003717 **
## cut.C        0.052801   0.014124   3.738 0.000196 ***
## cut^4        0.019574   0.010521   1.860 0.063120 .
## color.L      0.458186   0.015071  30.403  < 2e-16 ***
## color.Q     -0.123860   0.013841  -8.949  < 2e-16 ***
## color.C      0.011118   0.012636   0.880 0.379144
## color^4      0.020036   0.011510   1.741 0.082038 .
## color^5     -0.004993   0.010500  -0.475 0.634550    
## color^6     -0.010820   0.009485  -1.141 0.254296    
## clarity.L    0.881226   0.025364  34.743  < 2e-16 ***
## clarity.Q   -0.296055   0.023325 -12.693  < 2e-16 ***
## clarity.C    0.127964   0.020326   6.296 4.62e-10 ***
## clarity^4   -0.075185   0.016651  -4.515 7.10e-06 ***
## clarity^5    0.014028   0.013725   1.022 0.306992    
## clarity^6   -0.013932   0.011724  -1.188 0.235001    
## clarity^7    0.019596   0.009997   1.960 0.050265 .  
## table        0.002475   0.002513   0.985 0.324997    
## x            0.378702   0.097715   3.876 0.000113 ***
## y           -0.100265   0.096467  -1.039 0.298890    
## z           -0.083617   0.034023  -2.458 0.014157 *  
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 0.1262 on 976 degrees of freedom
## Multiple R-squared:  0.9848, Adjusted R-squared:  0.9845 
## F-statistic:  2753 on 23 and 976 DF,  p-value: < 2.2e-16
```

```r
# Backward elimination starting from full model
backward_aic <- step(full_model, direction = "backward")
```

```
## Start:  AIC=-4116.03
## log_price ~ log_carat + depth + cut + color + clarity + table + 
##     x + y + z
## 
##             Df Sum of Sq    RSS     AIC
## - table      1    0.0154 15.560 -4117.0
## - y          1    0.0172 15.562 -4116.9
## <none>                   15.545 -4116.0
## - z          1    0.0962 15.641 -4111.9
## - depth      1    0.1565 15.701 -4108.0
## - x          1    0.2392 15.784 -4102.8
## - cut        4    0.9250 16.470 -4066.2
## - log_carat  1    6.3720 21.917 -3774.5
## - color      6   16.3275 31.872 -3410.0
## - clarity    7   30.2688 45.814 -3049.2
## 
## Step:  AIC=-4117.04
## log_price ~ log_carat + depth + cut + color + clarity + x + y + 
##     z
## 
##             Df Sum of Sq    RSS     AIC
## - y          1    0.0175 15.578 -4117.9
## <none>                   15.560 -4117.0
## - z          1    0.0935 15.654 -4113.0
## - depth      1    0.1421 15.702 -4109.9
## - x          1    0.2325 15.793 -4104.2
## - cut        4    1.1169 16.677 -4055.7
## - log_carat  1    6.6977 22.258 -3761.1
## - color      6   16.3129 31.873 -3412.0
## - clarity    7   30.2537 45.814 -3051.2
## 
```

```
## Step:  AIC=-4117.91
## log_price ~ log_carat + depth + cut + color + clarity + x + z
##
##             Df Sum of Sq    RSS     AIC
## <none>                    15.578 -4117.9
## - z          1    0.1043 15.682 -4113.2
## - depth      1    0.1688 15.747 -4109.1
## - x          1    0.6913 16.269 -4076.5
## - cut        4    1.1164 16.694 -4056.7
## - log_carat  1    6.7744 22.352 -3758.8
## - color      6   16.3633 31.941 -3411.9
## - clarity    7   30.6475 46.225 -3044.2
```

```
summary(backward_aic)
```

```
##
## Call:
## lm(formula = log_price ~ log_carat + depth + cut + color + clarity +
##     x + z, data = sample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42579 -0.08349  0.00105  0.08756  0.37765
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.114362   0.417279  14.653  < 2e-16 ***
## log_carat    1.460119   0.070800  20.623  < 2e-16 ***
## depth        0.013817   0.004245   3.255 0.001173 **
## cut.L        0.126407   0.019840   6.371 2.88e-10 ***
## cut.Q       -0.046057   0.016621  -2.771 0.005695 **
## cut.C        0.045096   0.013015   3.465 0.000553 ***
## cut^4        0.015698   0.010114   1.552 0.120965
## color.L      0.458237   0.015048  30.451  < 2e-16 ***
## color.Q     -0.123398   0.013782  -8.954  < 2e-16 ***
## color.C      0.010601   0.012614   0.840 0.400873
## color^4      0.018989   0.011487   1.653 0.098638 .
## color^5     -0.004913   0.010497  -0.468 0.639873
## color^6     -0.010816   0.009481  -1.141 0.254268
## clarity.L    0.877861   0.025225  34.801  < 2e-16 ***
## clarity.Q   -0.295064   0.023315 -12.655  < 2e-16 ***
## clarity.C    0.127795   0.020325   6.287 4.86e-10 ***
## clarity^4   -0.074606   0.016644  -4.482 8.25e-06 ***
## clarity^5    0.013943   0.013725   1.016 0.309948
## clarity^6   -0.013001   0.011706  -1.111 0.267029
## clarity^7    0.019512   0.009987   1.954 0.051009 .
## x            0.280532   0.042584   6.588 7.28e-11 ***
## z           -0.086433   0.033779  -2.559 0.010652 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.1262 on 978 degrees of freedom
## Multiple R-squared:  0.9848, Adjusted R-squared:  0.9845
## F-statistic:  3015 on 21 and 978 DF,  p-value: < 2.2e-16
```

The best model chosen by backward elimination includes these predictors: log_carat, depth, cut, color, clarity, x, and z. Therefore, table and y were removed from the model. The adjusted R-squared increased to 0.9845. Overall, backward elimination helped simplify the model by excluding insignificant predictors.

## 2

```r
# Fit the model
model1 <- lm(log_price ~ log_carat + depth + cut + color + clarity + x + z, data = sample)

# Calculate VIF
vif_values1 <- faraway::vif(model1)
print(vif_values1)
```

```
##  log_carat      depth      cut.L      cut.Q      cut.C      cut^4    color.L 
## 106.294909   2.138922   3.098433   3.641947   1.944823   1.291009   1.414703 
##    color.Q    color.C    color^4    color^5    color^6  clarity.L  clarity.Q 
##   1.364244   1.315174   1.218713   1.087362   1.055151   2.616626   2.147061 
##  clarity.C  clarity^4  clarity^5  clarity^6  clarity^7          x          z 
##   2.376187   2.067590   1.624697   1.388702   1.153252 138.063205  34.330505
```

I assessed multicollinearity using VIF, and most variables had VIF values below 5, indicating low multicollinearity. However, z had a VIF of 34.33 and log_carat and x both showed extremely high VIFs of greater than 100, suggesting severe multicollinearity.

```r
cor(sample$log_carat, sample$x)
```

```
## [1] 0.9932371
```

I found that log_carat and x had a correlation of 0.993, indicating near-perfect linear dependence. I chose to remove x while keeping log_carat because log_carat is a transformed variable, which helps meet the linearity and normality assumptions of the regression model.

I did not remove z yet because we should only remove one predictor at a time. Therefore, I will now call the summary function on the updated model without x as shown below:

```r
model2 <- lm(log_price ~ log_carat + depth + cut + color + clarity + z, data = sample)
summary(model2)
```

```
## 
## Call:
## lm(formula = log_price ~ log_carat + depth + cut + color + clarity + 
##     z, data = sample)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.38576 -0.08704 -0.00060  0.08684  0.37556 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  8.493891   0.213405  39.802  < 2e-16 ***
## log_carat    1.863872   0.036205  51.481  < 2e-16 ***
## depth       -0.002207   0.003553  -0.621 0.534578
## cut.L        0.141737   0.020125   7.043 3.55e-12 ***
## cut.Q       -0.050634   0.016963  -2.985 0.002906 **
## cut.C        0.042020   0.013285   3.163 0.001610 **
## cut^4        0.014108   0.010328   1.366 0.172242
## color.L      0.443167   0.015192  29.171  < 2e-16 ***
## color.Q     -0.116424   0.014035  -8.295 3.56e-16 ***
## color.C      0.014120   0.012873   1.097 0.272942
## color^4      0.018934   0.011733   1.614 0.106924
## color^5     -0.003961   0.010721  -0.369 0.711859
## color^6     -0.007383   0.009670  -0.764 0.445319
## clarity.L    0.856936   0.025560  33.526  < 2e-16 ***
## clarity.Q   -0.268305   0.023451 -11.441  < 2e-16 ***
## clarity.C    0.104010   0.020431   5.091 4.27e-07 ***
## clarity^4   -0.063461   0.016913  -3.752 0.000186 ***
## clarity^5    0.007426   0.013983   0.531 0.595461
## clarity^6   -0.010851   0.011953  -0.908 0.364169
## clarity^7    0.019385   0.010201   1.900 0.057676 .
## z            0.021351   0.030185   0.707 0.479530
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.1289 on 979 degrees of freedom
## Multiple R-squared:  0.9841, Adjusted R-squared:  0.9838
## F-statistic:  3032 on 20 and 979 DF,  p-value: < 2.2e-16
```

Here, we observe that depth has a p-value of 0.534578, which is greater than the significance level of 0.05. This indicates that depth is not statistically significant and does not contribute meaningfully to the model, so we can remove it.

Although z also has a relatively large p-value, we follow a stepwise approach and remove only one predictor at a time. Now, we will remove depth and then re-evaluate the model by checking the VIF again as shown below:

```r
# Calculate VIF
model3 <- lm(log_price ~ log_carat + cut + color + clarity + z, data = sample)
vif_values2 <- faraway::vif(model3)
print(vif_values2)
```

```
## log_carat     cut.L     cut.Q     cut.C     cut^4   color.L   color.Q   color.C
## 23.690119  2.633097  3.382123  1.931938  1.273027  1.380336  1.349629  1.308002
##   color^4   color^5   color^6 clarity.L clarity.Q clarity.C clarity^4 clarity^5
##  1.218048  1.085676  1.051848  2.567466  2.080833  2.300156  2.046227  1.615573
## clarity^6 clarity^7         z
##  1.385793  1.151729 23.249947
```

The predictor z has a VIF of 23.25, which is above the commonly used threshold of 5, indicating a high degree of multicollinearity. The final model is now:

**log_price ~ log_carat + cut + color + clarity**

### 3

```
final_model <- lm(log_price ~ log_carat + cut + color + clarity, data = sample)

summary(final_model)
```

```
## 
## Call:
## lm(formula = log_price ~ log_carat + cut + color + clarity, data = sample)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.38245 -0.08795 -0.00107  0.08769  0.37658 
## 
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)    
## (Intercept)  8.441758   0.008136 1037.574  < 2e-16 ***
## log_carat    1.888861   0.008120  232.627  < 2e-16 ***
## cut.L        0.144664   0.018382    7.870 9.34e-15 ***
## cut.Q       -0.051908   0.016098   -3.224 0.001304 ** 
## cut.C        0.042099   0.013203    3.189 0.001475 ** 
## cut^4        0.014096   0.010158    1.388 0.165534    
## color.L      0.442876   0.015127   29.277  < 2e-16 ***
## color.Q     -0.115640   0.013988   -8.267 4.42e-16 ***
## color.C      0.013669   0.012840    1.065 0.287329    
## color^4      0.018950   0.011719    1.617 0.106176    
## color^5     -0.003474   0.010697   -0.325 0.745425    
## color^6     -0.007444   0.009660   -0.771 0.441135    
## clarity.L    0.860542   0.024972   34.460  < 2e-16 ***
## clarity.Q   -0.271041   0.022981  -11.794  < 2e-16 ***
## clarity.C    0.105622   0.020098    5.255 1.81e-07 ***
## clarity^4   -0.064884   0.016688   -3.888 0.000108 ***
## clarity^5    0.008280   0.013912    0.595 0.551845    
## clarity^6   -0.011418   0.011923   -0.958 0.338502    
## clarity^7    0.019338   0.010181    1.899 0.057805 .  
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 0.1288 on 981 degrees of freedom
## Multiple R-squared:  0.9841, Adjusted R-squared:  0.9838 
## F-statistic:  3373 on 18 and 981 DF,  p-value: < 2.2e-16
```

```
# Choose one specific set of values for your predictor variables (your X’s):
new_data <- data.frame(
  log_carat = log(0.23),
  cut = factor("Ideal", levels = levels(sample$cut)),
  color = factor("E", levels = levels(sample$color)),
  clarity = factor("SI2", levels = levels(sample$clarity))
)

# Predict with confidence interval (CI) for mean predicted log_price
exp(predict(final_model, newdata = new_data, interval = "confidence", level = 0.95))
```

```
##        fit      lwr      upr
## 1 276.4725 267.1737 286.0949
```

```
# Predict with prediction interval (PI) for a future log_price
exp(predict(final_model, newdata = new_data, interval = "prediction", level = 0.95))
```

```
##        fit      lwr      upr
## 1 276.4725 214.2206 356.8145
```

I used the final linear regression model, which includes log_carat, cut, color, and clarity as predictors, to estimate the price of a diamond with specific characteristics (because the instructions asked me to provide intervals for at least one combination of X's):

- carat = 0.23
- cut = Ideal
- color = E
- clarity = SI2

I chose these values for my one combination because they represent a typical example of a diamond in the dataset and are frequently observed categories.

It is important to note that my regression model predicts log_price rather than the price itself. To interpret these predictions in the original price scale, I took the exponential of the predicted log values and their interval bounds.

- Confidence Interval for the Mean Price: We are 95% confident that the true average price of diamonds with these features lies between approximately $267.17 and $286.09.
- Prediction Interval for an Individual Diamond Price: For a single future diamond with these same features, we are 95% confident that its price will fall between approximately $214.22 and $356.81. This wider range reflects the greater uncertainty involved in predicting individual outcomes, accounting for natural variation, random noise, and unobserved factors beyond the model.

## 4 Final Summary

This analysis investigates how various diamond attributes, such as carat, depth, cut, color, and clarity, influence price, using a random sample of 1,000 observations from Kaggle's diamonds dataset. The primary objective was to build an accurate and interpretable model to predict diamond prices and to understand which factors have the most influence on price. To initially explore the dataset, I visualized both numerical and categorical variables to assess distributions and relationships. A Pearson correlation matrix revealed strong positive correlations between price and physical size metrics (carat, x, y, and z) while quality-related categorical variables (cut, color, and clarity) showed weaker, often negative correlations with size and price.

Next, I began with a simple linear regression using carat as the sole predictor of price. The model yielded an adjusted $R^2$ of 0.8524, indicating that carat alone explains a large portion of price variability. Hypothesis testing confirmed that carat is a statistically significant predictor. However, residual diagnostics revealed violations of some of the linear regression assumptions, specifically heteroscedasticity and non-normal residuals. To address these issues, I applied a log-log transformation to both the predictor and response, which greatly improved fit with an adjusted R-squared of 0.938.

Then, I incorporated additional predictors one at a time, observing changes in adjusted R-squared. All variables improved model performance except for z. To find the best model, I applied backward elimination using Akaike Information Criterion, and the resulting model included log_carat, depth, cut, color, clarity, x, and z. This full model achieved an exceptional adjusted R-squared of 0.9845. Despite the model's high performance, I investigated multicollinearity using Variance Inflation Factors. Most predictors had VIF values below 5, indicating low multicollinearity. However, log_carat and x both had VIFs exceeding 100, signaling severe multicollinearity. A Pearson correlation between log_carat and x confirmed a near-perfect linear relationship, so I decided to remove x and kept log_carat due to its log transformation. This near-perfect correlation highlighted how variables can overlap and distort model interpretation if not addressed.

Upon re-evaluating the updated model, I also found that depth was statistically insignificant and removed it from the model. I then recalculated VIF and identified z as another problematic predictor, leading to its removal as well. The resulting model had a slightly lower adjusted $R^2$ of 0.9838, but the reduction in severe multicollinearity justified the trade-off; this is because a minor decrease in explanatory power is acceptable when it leads to more stable and reliable coefficient estimates.

Lastly, the final model was used to estimate both confidence and prediction intervals for diamond price using at least one combination of X's. For a diamond with specified characteristics that are common, the predicted average price had a 95% confidence interval of [$267.17, $286.09] and a prediction interval of [$214.22, $356.81], reflecting typical variation around the mean. As expected, the prediction interval was broader than the confidence interval. This project demonstrated the power of data transformation to meet linear regression assumptions, careful variable selection, and multicollinearity control. Overall, I arrived at a final model that explains over 98% of the variability in diamond prices using key physical and quality attributes.