Regression Analysis on Diamonds Dataset

Hannah Nguyen

2025-06-13

Introduction

In this project, I explored how to create regression models by analyzing the diamonds dataset obtained from **Kaggle**. I learned how to approach creating regression models and ways to linearize the model. I also explored other methods, like backwards elimination with AIC and variable inflation factor (VIF), in creating the best regression model to predict diamond prices.

Part 1: Data Description and Descriptive Statistics

First, I read in the data and created the corresponding dataframe. I also downloaded the necessary packages used throughout the project.

```
# Download necessary packages
library(readr)
library(dplyr)
library(tidyr)
library(ggplot2)
library(MASS) # for stepAIC
library(car) # for VIF
library(knitr) # for visualization
set.seed(06042025) # for reproducibility

# Read in the data and create a dataframe

diamonds_data <- read_csv("Diamonds_Prices2022.csv")
diamonds_dataframe <- data.frame(carat = diamonds_data$carat,
                                 cut = diamonds_data$cut,
                                 color = diamonds_data$color,
                                 clarity = diamonds_data$clarity,
                                 depth = diamonds_data$depth,
                                 table = diamonds_data$table,
                                 price = diamonds_data$price,
                                 x = diamonds_data$x,
                                 y = diamonds_data$y,
                                 z = diamonds_data$z)
```

1) Select a random sample

```r
sample_df <- sample_n(diamonds_dataframe, 1000, )
str(sample_df)
```

```
## 'data.frame':    1000 obs. of  10 variables:
##  $ carat  : num  0.33 0.72 0.95 2.01 1.03 0.57 0.9 0.33 1.51 0.43 ...
##  $ cut    : chr  "Ideal" "Premium" "Premium" "Fair" ...
##  $ color  : chr  "E" "F" "E" "H" ...
##  $ clarity: chr  "VS2" "SI1" "SI2" "SI2" ...
##  $ depth  : num  62.1 59.3 60.7 66.7 60.9 60.4 63 63 62.5 62.6 ...
##  $ table  : num  56 59 59 56 56 57 58 56 59 58 ...
##  $ price  : num  723 2360 4070 10772 4891 ...
##  $ x      : num  4.4 5.85 6.36 7.8 6.56 5.39 6.13 4.42 7.29 4.79 ...
##  $ y      : num  4.45 5.81 6.3 7.76 6.51 5.44 6.16 4.41 7.34 4.82 ...
##  $ z      : num  2.75 3.46 3.84 5.19 3.98 3.27 3.87 2.78 4.57 3.01 ...
```

In this dataframe, there are 10 different variables. 3 are categorical, cut, color, and clarity, while 7 are numerical, carat, depth, table, price, x, y, and z. I selected a random sample of 1000 observations from the main data frame.

2) Describing each variable

```r
# Summary
kable(t(summary(sample_df)))
```

carat	Min. :0.2300	1st Qu.:0.4000	Median :0.7000	Mean :0.7919	3rd Qu.:1.0400	Max. :3.0400
cut	Length:1000	Class :character	Mode :character	NA	NA	NA
color	Length:1000	Class :character	Mode :character	NA	NA	NA
clarity	Length:1000	Class :character	Mode :character	NA	NA	NA
depth	Min. :56.80	1st Qu.:61.10	Median :61.90	Mean :61.73	3rd Qu.:62.50	Max. :68.70
table	Min. :44.00	1st Qu.:56.00	Median :57.00	Mean :57.43	3rd Qu.:59.00	Max. :66.00
price	Min. : 377	1st Qu.: 945	Median : 2455	Mean : 3888	3rd Qu.: 5215	Max. :18788
x	Min. :3.930	1st Qu.:4.730	Median :5.690	Mean :5.723	3rd Qu.:6.532	Max. :9.510
y	Min. :3.950	1st Qu.:4.730	Median :5.715	Mean :5.726	3rd Qu.:6.532	Max. :9.460
z	Min. :0.000	1st Qu.:2.900	Median :3.520	Mean :3.528	3rd Qu.:4.022	Max. :5.620

Carat is a numerical variable with a minimum value of 0.23 and maximum value of 3.04. It has a median of 0.7 and mean of 0.79.

Cut is a categorical variable taking values "Fair", "Good", "Very Good", "Ideal", or "Premium".

Color is a categorical variable taking values “D”, “E”, “F”, “G”, “H”, or “I”.

Clarity is a categorical variable taking values “I1”, “IF”, “SI1”, “SI2”, “VS1”, “VS2”, “VVS1”, or “VVS2”.

Depth is a numerical variable with a minimum value of 56.80 and maximum value of 68.70. It has a median of 61.90 and a mean of 61.73.

Table is a numerical variable with a minimum value of 44.00 and maximum value of 66.00. It has a median of 57.00 and a mean of 57.43.

Price is a numerical variable with a minimum value of 377 and maximum value of 18788. It has a median of 2455 and a mean of 3888.

X is a numerical variable with a minimum value of 3.93 and maximum value of 9.51. It has a median of 5.69 and a mean of 5.72.

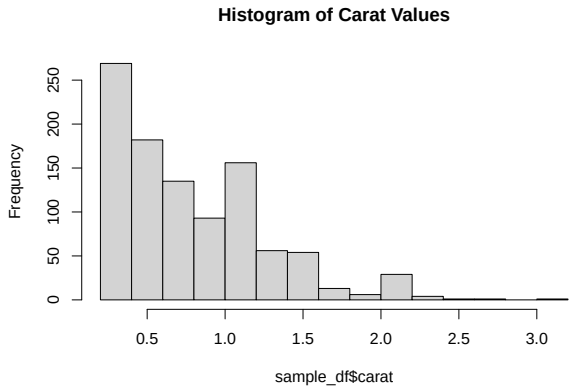
Y is a numerical variable with a minimum value of 3.95 and maximum value of 9.46. It has a median of 5.72 and a mean of 5.726.

Z is a numerical variable with a minimum value of 0.00 and maximum value of 5.62. It has a median of 3.52 and a mean of 3.528.

Histogram and Barplots

Next, I created a histogram for one continuous variable and described it.

```
# Histograms for continuous variables
hist(sample_df$carat, main = paste("Histogram of Carat Values"))
```



The distribution for carat appears to be right-skeId, with a right tail. Most of the values seem to be within the 0.5-1.5 range, while feIr values are closer towards 3.0.

Next, I created a barplot containing all categorical random variables, folloId by a description of each variable.

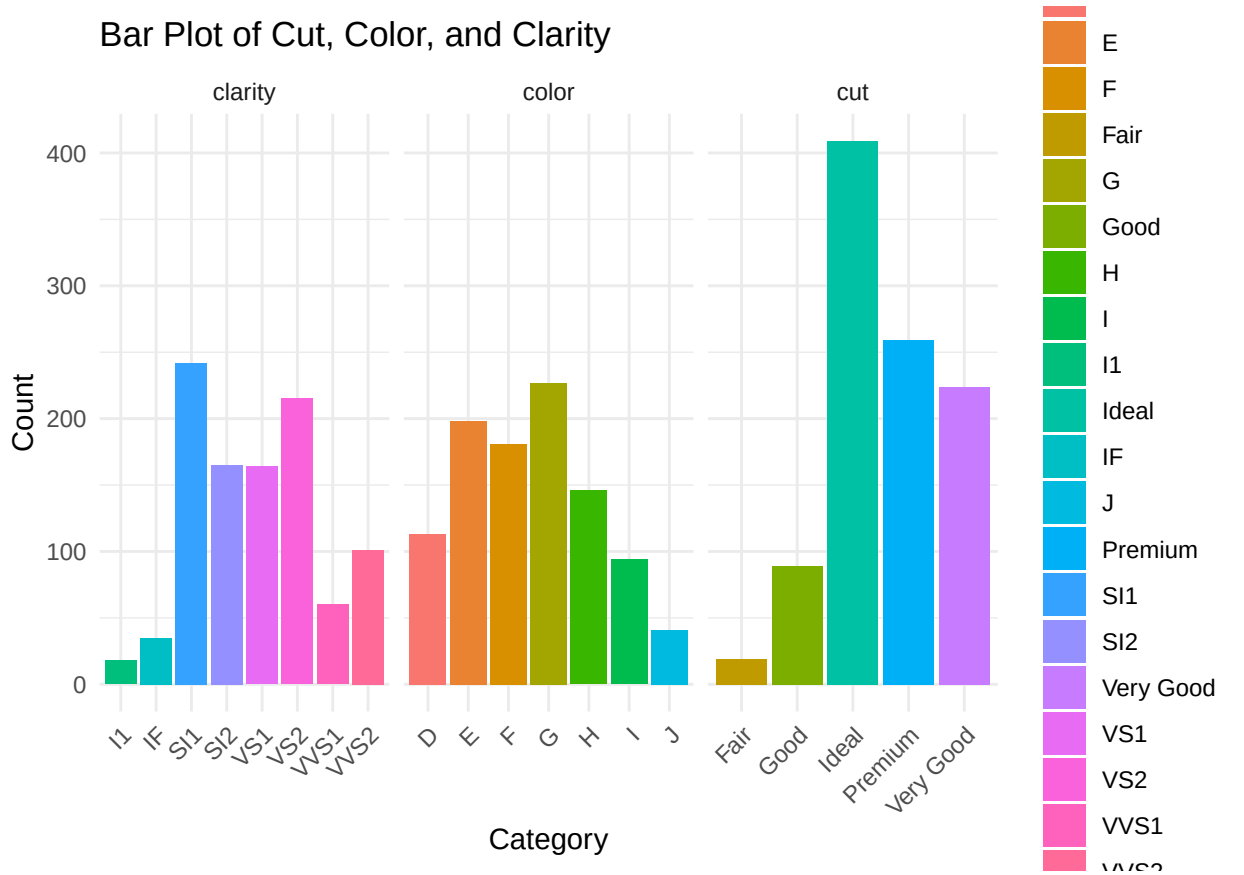
```
# Bar Plot for categorical random variables

df_categorical <- sample_df %>% dplyr::select(cut, color, clarity)

D_cat <- df_categorical %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Category")
kable(head(D_cat))
```

Variable	Category
cut	Ideal
color	E
clarity	VS2
cut	Premium
color	F
clarity	SI1

```
# Creating a bar plot
ggplot(D_cat, aes(x = Category, fill = Category)) +
  geom_bar() +
  facet_wrap(~ Variable, scales = "free_x") +
  theme_minimal() +
  labs(title = "Bar Plot of Cut, Color, and Clarity",
       x = "Category",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Based on the bar plots, Clarity appears to have high frequency with the “SL1” and “VS2” values. On both ends of the plot, the frequency decreases. For Color, the highest frequency is with values “G” and “E”, while the loIst is with value “J”. Lastly, for Cut, the highest frequency is with values “Ideal”, folloId by “Premium” and “Very Good”, while the least frequent values are “Good” and “Fair”.

3) Correlation between Variables

To determine whether there is any correlation between the variables, I dropped the categorical variables from the dataset and checked the correlation matrix.

```
# Drop the categorical columns from sample_df
quantitative_sample_df <- sample_df %>%
  dplyr::select(-cut, -color, -clarity)

# Check the correlation matrix
kable(cor(quantitative_sample_df))
```

	carat	depth	table	price	x	y	z
carat	1.0000000	0.0323413	0.1682324	0.9265223	0.9788262	0.9779573	0.9328586
depth	0.0323413	1.0000000	-0.3080420	-0.0069444	-0.0061910	-0.0101441	0.0986699
table	0.1682324	-0.3080420	1.0000000	0.1202346	0.1803493	0.1753002	0.1379081
price	0.9265223	-0.0069444	0.1202346	1.0000000	0.8917672	0.8941722	0.8471297
x	0.9788262	-0.0061910	0.1803493	0.8917672	1.0000000	0.9988779	0.9606866
y	0.9779573	-0.0101441	0.1753002	0.8941722	0.9988779	1.0000000	0.9603421
z	0.9328586	0.0986699	0.1379081	0.8471297	0.9606866	0.9603421	1.0000000

Based on the correlation matrix, the variables Carat and Price have high correlation with a coefficient of 0.923. Carat and X is also highly correlated with 0.979, as well as Carat and Y with 0.977. Price and X are highly correlated with coefficient 0.892, as well as Price and Y with 0.894. X and Y are highly correlated with coefficient 0.999. Z is highly correlated with Carat with a coefficient of 0.933, as well as Z and Price with coefficient 0.847. Z and X are highly correlated with a coefficient of 0.96, as well as Z and Y with coefficient 0.96.

4) Run the full model

First, I ran the full linear model, using each variable and viewed the results.

```
full_mod <- lm(price ~ carat + cut + color + clarity + depth + table + x + y + z,
               data = sample_df)
kable(summary(full_mod)$coefficients)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	6218.70187	2898.16334	2.1457389	0.0321399
carat	11648.62827	366.33545	31.7977092	0.0000000
cutGood	486.32205	274.78520	1.7698262	0.0770683
cutIdeal	756.46777	275.45861	2.7462121	0.0061397
cutPremium	815.71642	268.35980	3.0396372	0.0024318
cutVery Good	586.48778	269.74968	2.1741927	0.0299303
colorE	-166.15138	123.17755	-1.3488771	0.1776893
colorF	-177.50008	125.72227	-1.4118427	0.1583150
colorG	-383.04700	123.12944	-3.1109293	0.0019192
colorH	-1055.57071	132.64079	-7.9581153	0.0000000
colorI	-1605.13945	148.40663	-10.8158203	0.0000000
colorJ	-1966.13307	192.82524	-10.1964509	0.0000000

	Estimate	Std. Error	t value	Pr(>|t|)
clarityIF	4323.66101	318.09015	13.5925649	0.0000000
claritySI1	3042.19380	259.13294	11.7398962	0.0000000
claritySI2	1974.49789	261.79419	7.5421761	0.0000000
clarityVS1	3953.11839	266.42092	14.8378678	0.0000000
clarityVS2	3521.80995	261.91921	13.4461689	0.0000000
clarityVVS1	4459.20095	291.28626	15.3086555	0.0000000
clarityVVS2	4268.06579	275.44575	15.4951233	0.0000000
depth	-118.56898	32.02426	-3.7024734	0.0002255
table	-17.06026	20.16082	-0.8462089	0.3976436
x	-3870.88528	723.00649	-5.3538735	0.0000001
y	2393.46285	719.74914	3.3254126	0.0009158
z	487.19749	191.89275	2.5389051	0.0112743

5) Comments

The data surprised me, since based on the correlation matrix, Z was highly correlated with most of the other numerical variables, while depth and table was not highly correlated with any of the other numerical variables.

Part 2: Simple Linear Regression

1) One Variable linear model

I first started with one predictor variable, carat, and created a linear model using it.

```
slr <- lm(price ~ carat, data = sample_df)
```

2) Summary

Next, I ran a summary on the model. I described each coefficient and the output

```
summary(slr)
```

```
##
## Call:
## lm(formula = price ~ carat, data = sample_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6913.4  -813.1    13.9   500.7  7863.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2253.27      91.73  -24.56   <2e-16 ***
## carat        7755.31      99.69   77.80   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 1477 on 998 degrees of freedom
## Multiple R-squared:  0.8584, Adjusted R-squared:  0.8583
## F-statistic:  6052 on 1 and 998 DF,  p-value: < 2.2e-16
```

The function under Call: lm(formula = price ~ carat, data = sample_df) is the name of the formula being used. The values under Residuals show the min, first quartile, median, 3rd quartile, and maximum residuals.

Under Coefficients:, The intercept estimate is -2253.27, so if the carat value is 0, then the price is expected to be -$2253.27. The average deviation of the estimated coefficient from its true population value is $91.73. The hypothesis testing involves testing the hypothesis, $H_0 : \beta_0 = 0$ *vs* $H_A : \beta_0 \neq 0$, where the t-statistic obtained was -24.56 with a p-value of <2e-16. The t-value is how many standard errors the estimated coefficient is from 0. The p-value helps us determine if the observed relationship could have occurred by chance or if there is strong evidence to support the effect. Since the p-value $< \alpha = 0.05$, I reject the null hypothesis and conclude that there is significant evidence that $\beta_0 \neq 0$, meaning the parameter estimator should be included in the model.

The carat coefficient estimate is $7755.31. For every 1.0 carat value increase, the price is expected to increase by $7755.31. The average deviation of the estimated coefficient from its true population value is $99.69. The hypothesis testing involves testing the hypothesis, $H_0 : \beta_0 = 0$ *vs* $H_A : \beta_0 \neq 0$, where the t-statistic obtained was 77.80 with a p-value of <2e-16. The t-value is how many standard errors the estimated coefficient is from 0. The p-value helps us determine if the observed relationship could have occurred by chance or if there is strong evidence to support the effect. Since the p-value $< \alpha = 0.05$, I reject the null hypothesis and conclude that there is significant evidence that $\beta_0 \neq 0$, meaning the parameter estimator should be included in the model.

The residual standard error is 1477, the average deviation of the actual data points from the predicted values by the model, where loIr value means predictions are closer to the actual data points on average, with 998 degrees of freedom. The multiple R-squared is 0.8584, which means that 85.84% of the variance in the price is explained by the carat predictor through the linear model. The adjusted R-squared is 0.8583, which means that 85.83% of the variance in the price is explained by the combination of predictor variables through the linear model. The F-statistic is 6052 with 1 predictor and 998 degrees of freedom with p-value <2.2e-16. At $\alpha = 0.05$, I rejected the null hypothesis, since the p-value, <2.2e-16, is less than $\alpha = 0.05$. I concluded that this model explains a significant amount of variance in the price.

Confidence Intervals

Then, I created confidence and prediction intervals, using a sample value of 0.5 carats.

```r
# Confidence Interval
confint(slr, level = .95)
```

```
##                 2.5 %    97.5 %
## (Intercept) -2433.282 -2073.263
## carat        7559.690  7950.934
```

```r
# Confidence and Prediction Interval
test_df <- data.frame(carat = 0.5)
predict(slr, newdata = test_df, level = .95, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 1624.383 1516.381 1732.385
```

```r
predict(slr, newdata = test_df, level = .95, interval = "prediction")
```

```
##        fit       lwr      upr
## 1 1624.383 -1276.404 4525.171
```

The estimated value of the intercept is expected to be between -2303.548 and -1928.422 with 95% confidence. Since the interval doesn't contain 0, this implies that this coefficient is significant.

The estimated value of the carat coefficient is expected to be between 7344.076 and 7733.388 with 95% confidence. Since the interval doesn't contain 0, this implies that this coefficient is significant.

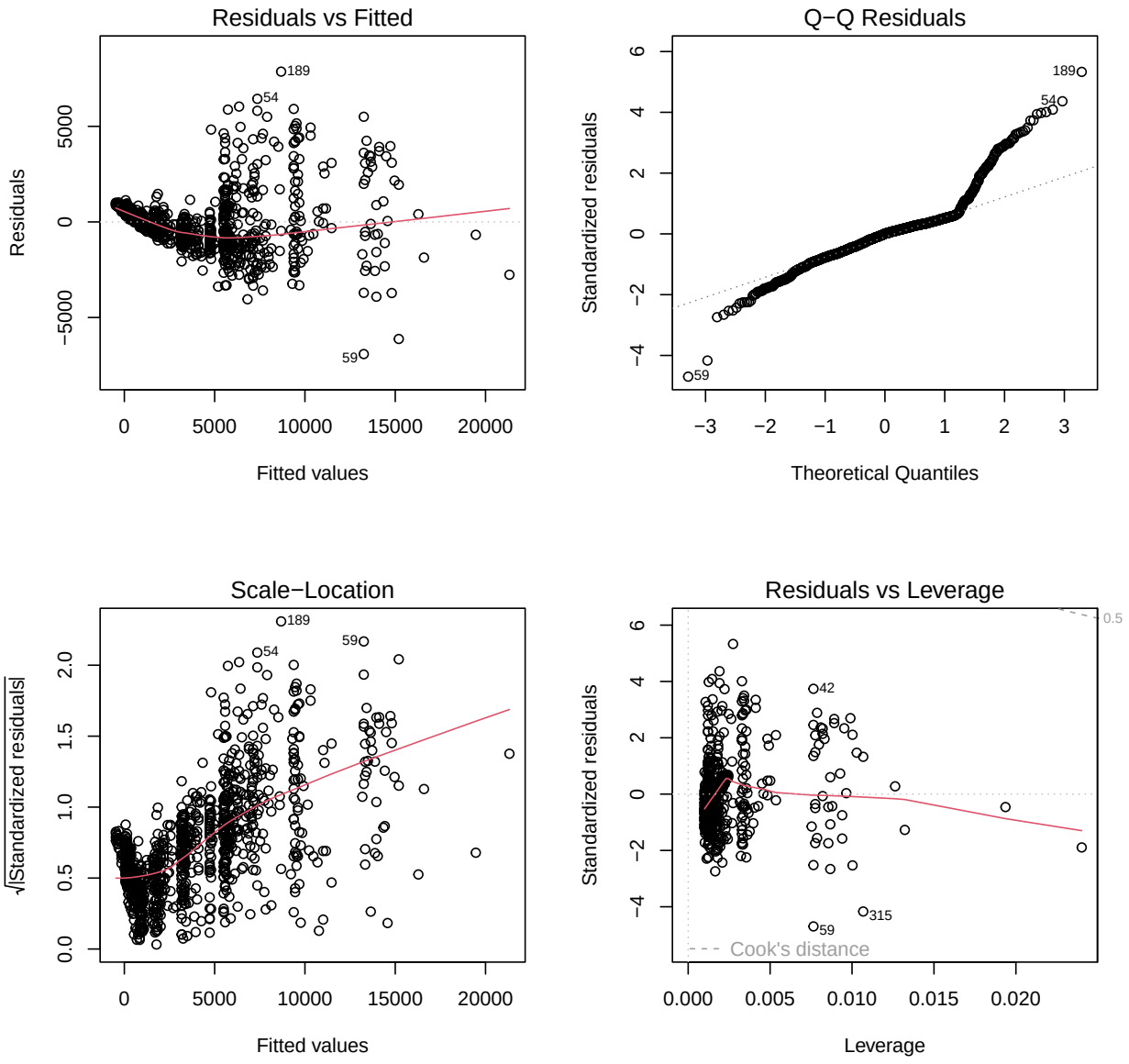
Prediction intervals show where the expected price value would be for a given carat value. For example, for a diamond with 0.5 carats, the expected price is $1653.38 with a 95% confidence interval of [1516.38, 1732.38] and a 95% prediction interval of [-1276,40, 4525.17]

For the plots generated by the plot() function, Residuals vs. Fitted is used to check the linearity of the model. If the red line is not straight and centered near 0, then it would indicate that there is nonlinearity. The QQ plot is used to check for normality of the errors, where it plots the sample quantiles of the residuals against theoretical quantiles. The Scale-Location plot is used to check whether the spread of residuals is constant, which would imply constant variance or homoscedasticity. The Residuals vs Leverage plot checks for any points with high leverage and large residuals that can significantly influence the regression model.

3) Linear Model Transformation

To ensure that the regression model is linear, I performed various transformations on the model to ensure linearity between the dependent variable, price, and independent variable, carat.

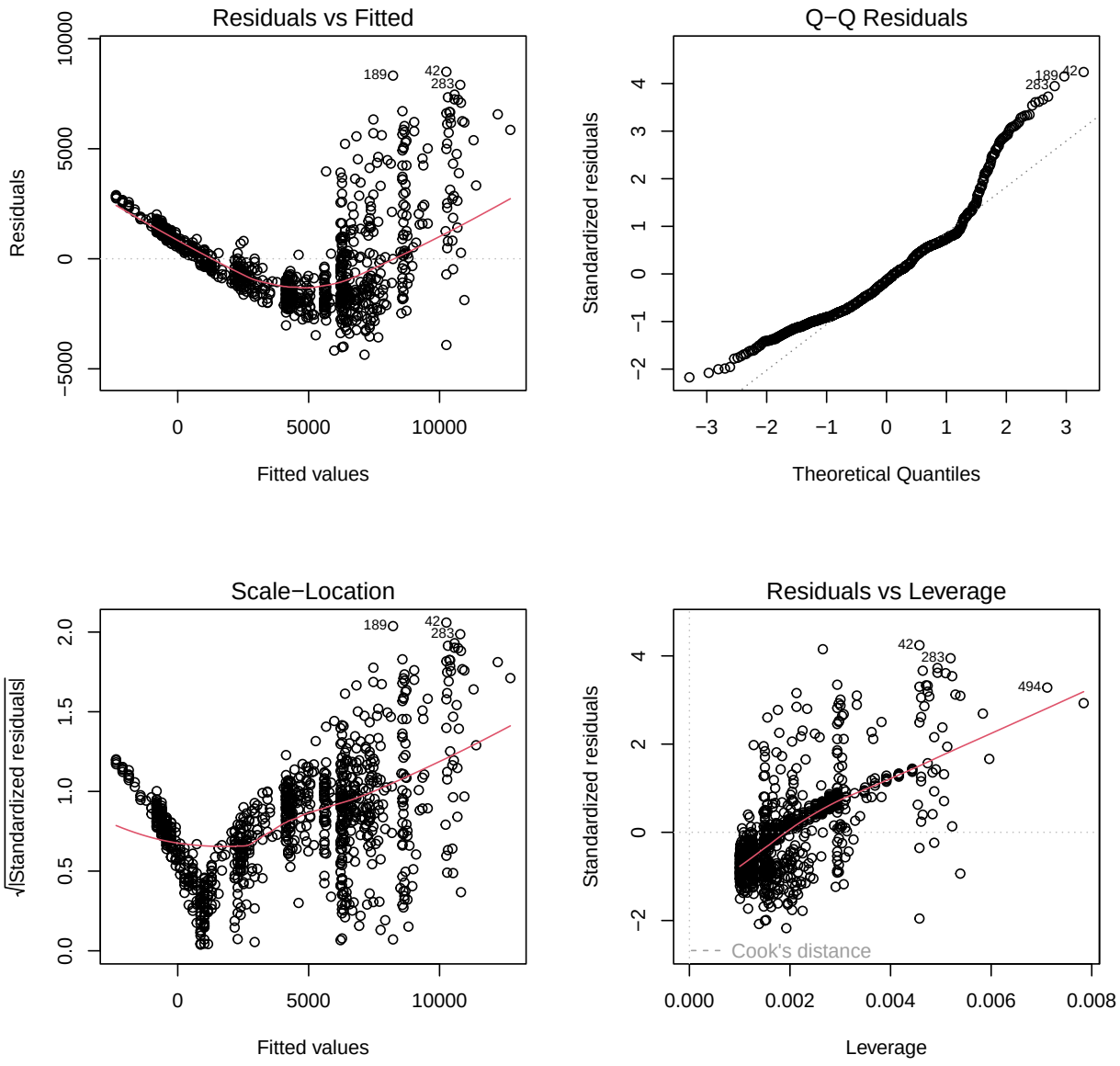
```r
par(mfrow = c(2, 2))
# Check the 4 plots
plot(slr)
```



Since the residuals vs. fitted plot isn't linear, I will have to transform the predictor variable, carat. Carat's max/min > 10, so a log transformation will be appropriate.

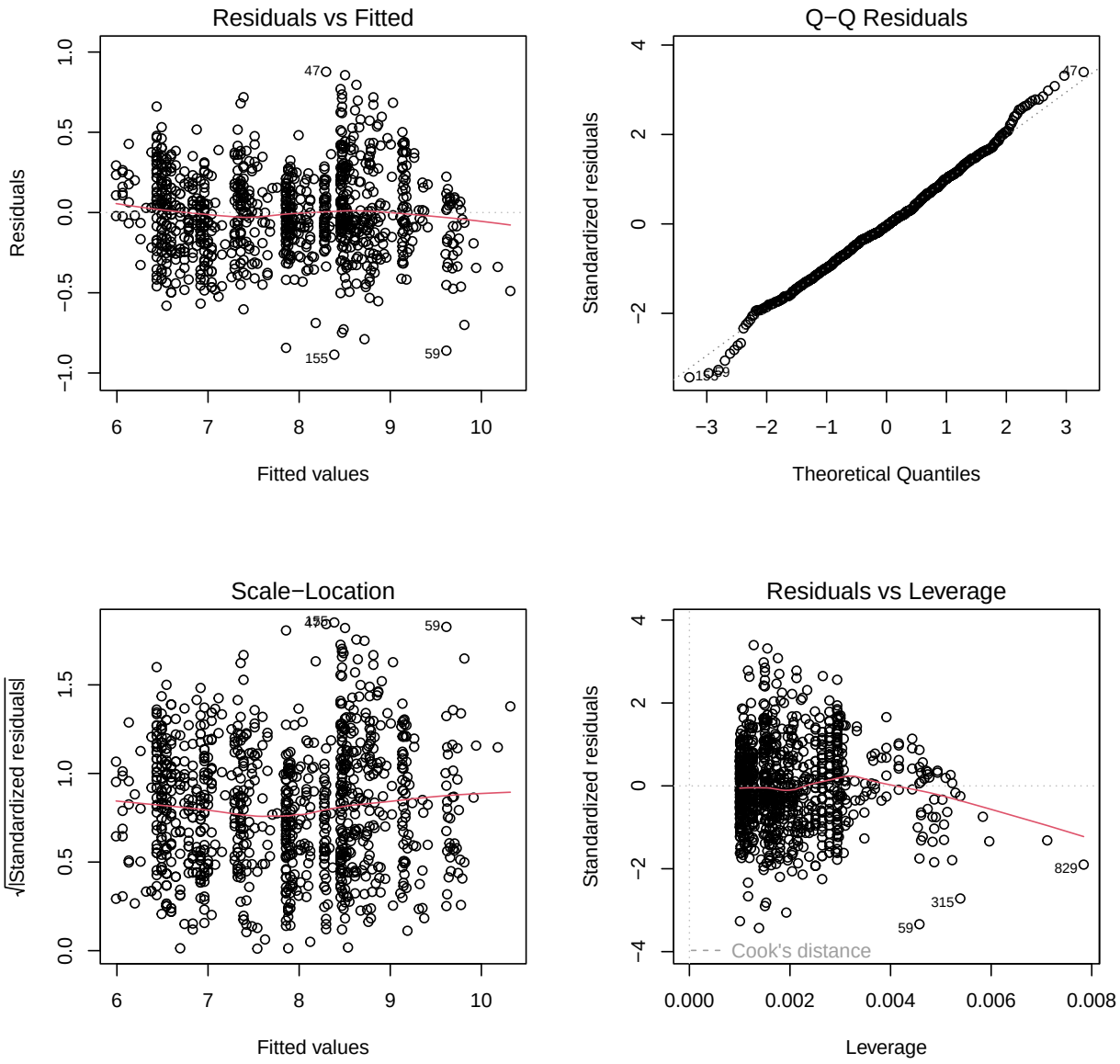
```
par(mfrow = c(2, 2))
slr_transformed <- lm(price ~ log(carat), data = sample_df)

plot(slr_transformed)
```



To fix the inconsistency of the variance, I would log transform the dependent variable as well.

```
par(mfrow = c(2, 2))
slr_transformed <- lm(log(price) ~ log(carat), data = sample_df)
plot(slr_transformed)
```



From the Residuals vs. Fitted values plot, since the red line is mostly flat and centered around zero, with the residuals spread staying consistent around the line, it meets the linearity and homoscedacity assumptions. The normality assumptions are also met since the theoretical quantiles vs standardized residuals follow the linear line.

4) Summary of the Transformed Linear Model

```
summary(slr_transformed)
```

```
##
## Call:
```

```
## lm(formula = log(price) ~ log(carat), data = sample_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.88575 -0.17020 -0.01248  0.17167  0.87719
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.454152   0.009933   851.1   <2e-16 ***
## log(carat)  1.676128   0.014141   118.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2584 on 998 degrees of freedom
## Multiple R-squared:  0.9337, Adjusted R-squared:  0.9336
## F-statistic: 1.405e+04 on 1 and 998 DF,  p-value: < 2.2e-16
```

Log transformation was performed on both the independent variable (carat) and dependent variable (price). Comparing back with the slr model, the intercept changed from -2253.27 to 8.454. The carat variable's coefficient was 7755.31, while log(carat) was 1.676128. The standard errors had been drastically decreased from the slr model to the slr_transformed model.

The multiple R squared of the slr model was 0.8584 and adjusted R squared was 0.8583. For the slr_transformed model, the multiple R squared was 0.9337, while the adjusted R squared was 0.9336, so both the multiple R squared and adjusted R squared increased after transforming the independent and dependent variables.

5) Adding variables to the Simple Linear Regression Model

To transform the simple linear regression model to a multilinear regression model, I added the rest of the predictor variables and individually checked if they contributed to increasing the adjusted R squared. If they did not or only increased it minimally, they were excluded. Below, I showed the final model.

```
regression_model <- lm(log(price) ~ log(carat) + table + depth +
                         cut + color + clarity + x + y, data = sample_df)
summary(regression_model)
```

```
##
## Call:
## lm(formula = log(price) ~ log(carat) + table + depth + cut +
##     color + clarity + x + y, data = sample_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41083 -0.08214  0.00073  0.08511  0.39182
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.7676547  0.5006749  15.514  < 2e-16 ***
## log(carat)    1.8152581  0.0724570  25.053  < 2e-16 ***
## table        -0.0011091  0.0025723  -0.431 0.666426
## depth        -0.0008003  0.0041243  -0.194 0.846182
## cutGood       0.0389479  0.0345802   1.126 0.260312
```

```
## cutIdeal      0.1209467  0.0347414   3.481 0.000521 ***
## cutPremium    0.1017639  0.0338847   3.003 0.002739 **
## cutVery Good  0.0834314  0.0340318   2.452 0.014398 *
## colorE       -0.0387456  0.0155373  -2.494 0.012806 *
## colorF       -0.0879366  0.0158519  -5.547 3.73e-08 ***
## colorG       -0.1445980  0.0155194  -9.317  < 2e-16 ***
## colorH       -0.2569777  0.0167234 -15.366  < 2e-16 ***
## colorI       -0.3843372  0.0187180 -20.533  < 2e-16 ***
## colorJ       -0.4630846  0.0243281 -19.035  < 2e-16 ***
## clarityIF     1.0993458  0.0402054  27.343  < 2e-16 ***
## claritySI1    0.6011369  0.0326982  18.384  < 2e-16 ***
## claritySI2    0.4267636  0.0330234  12.923  < 2e-16 ***
## clarityVS1    0.8160976  0.0336326  24.265  < 2e-16 ***
## clarityVS2    0.7382227  0.0330529  22.335  < 2e-16 ***
## clarityVVS1   1.0274992  0.0367926  27.927  < 2e-16 ***
## clarityVVS2   0.9500626  0.0347853  27.312  < 2e-16 ***
## x             0.2031133  0.0916306   2.217 0.026876 *
## y            -0.1678031  0.0920742  -1.822 0.068688 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1305 on 977 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9831
## F-statistic:  2637 on 22 and 977 DF,  p-value: < 2.2e-16
```

After adding predictors and testing whether the adjusted R squared increased, I reach a final model for dependent variable, log(price), with predictors log(carat), table, depth, cut, color, clarity, x, and y. For every 1% increase in the carat value, I would expect the log(price) to increase by 1.82%, while holding other variables constant. The p-value is <2e-16, so it indicates that it is significant to the model at $\alpha = 0.05$. The p-values indicate how significant the coefficients are, for example, the coefficient, table, has a p-value of 0.667, which would indicate that it is not statistically significant to include in the model, even though the adjusted R squared increased slightly with that predictor added. I achieved an adjusted R squared of 0.9831, while the previous model with only log(carat) had an adjusted R squared of 0.9336. Since the new model's adjusted R squared is higher, the model explains more variance in the log(price) variable (98.31%) with the combination of the independent variables through the linear model.

The residual standard error is 0.1305, the average deviation of the actual data points from the predicted values by the model, where loIr value means predictions are closer to the actual data points on average, with 977 degrees of freedom. The F-statistic is 2637 with 22 predictors and 977 degrees of freedom with p-value <2.2e-16. At $\alpha = 0.05$, I reject the null hypothesis, since the p-value, <2.2e-16, is less than $\alpha = 0.05$. I conclude that this model explains a significant amount of variance in the price.

6) Comments

I noticed that clarity contributed a significant portion to the adjusted R squared. Some variables, like Z, did not affect or contribute significantly to the adjusted R squared, so they were dropped from the model. I found it interesting that log transformation on the independent and dependent variables help fix the variance and linearity of the model.

Part 3: Regression Model using Backwards Elimination

1) Using Backwards AIC Method

Using the multilinear regression model from earlier, I performed backwards elimination with AIC on the model to calculate which variables are unnecessary and needed to be removed.

```
step_aic <- stepAIC(regression_model, direction = "backward", trace = FALSE)
summary(step_aic)
```

```
##
## Call:
## lm(formula = log(price) ~ log(carat) + cut + color + clarity +
##     x + y, data = sample_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40912 -0.08183  0.00056  0.08511  0.39191
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.63511    0.21663  35.245  < 2e-16 ***
## log(carat)    1.81007    0.06450  28.061  < 2e-16 ***
## cutGood       0.03991    0.03401   1.173  0.24091
## cutIdeal      0.12539    0.03246   3.863  0.00012 ***
## cutPremium    0.10367    0.03232   3.207  0.00138 **
## cutVery Good  0.08549    0.03298   2.592  0.00967 **
## colorE       -0.03849    0.01551  -2.482  0.01325 *
## colorF       -0.08775    0.01582  -5.547 3.73e-08 ***
## colorG       -0.14438    0.01548  -9.328  < 2e-16 ***
## colorH       -0.25683    0.01667 -15.410  < 2e-16 ***
## colorI       -0.38429    0.01866 -20.598  < 2e-16 ***
## colorJ       -0.46301    0.02428 -19.069  < 2e-16 ***
## clarityIF     1.09943    0.03997  27.506  < 2e-16 ***
## claritySI1    0.60077    0.03252  18.475  < 2e-16 ***
## claritySI2    0.42684    0.03278  13.020  < 2e-16 ***
## clarityVS1    0.81612    0.03342  24.418  < 2e-16 ***
## clarityVS2    0.73796    0.03286  22.459  < 2e-16 ***
## clarityVVS1   1.02768    0.03663  28.055  < 2e-16 ***
## clarityVVS2   0.95002    0.03462  27.441  < 2e-16 ***
## x             0.20210    0.09149   2.209  0.02740 *
## y            -0.16426    0.08974  -1.830  0.06750 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1304 on 979 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9831
## F-statistic:  2906 on 20 and 979 DF,  p-value: < 2.2e-16
```

Using the backwards elimination method with AIC, I noticed that the model only kept cut, color, clarity, X, and Y. It dropped table and depth, which I thought was interesting because I added it to my previous model since I noticed that they had increased adjusted R squared, but it's p-value was high, indicating that it was best to drop them from the model.

2) Checking Variance Inflation Factor (VIF)

To check for possible correlation between the variables, I used the VIF() function on step_aic. I decided to remove predictor variables whos VIF was greater than 5.

```
vif_vals <- vif(step_aic)
vif_adj <- vif_vals[, "GVIF^(1/(2*Df))"]

high_vif <- vif_adj[vif_adj > 5]

high_vif
```

```
## log(carat)          x          y
##   9.040284  24.610387  23.960866
```

Since log(carat), x, and y have high VIF over 5, I will start by removing the predictor x, then checking the VIF again.

```
test <- lm(formula = log(price) ~ log(carat) + cut + color + clarity + y, data = sample_df)

vif_vals <- vif(test)
vif_adj <- vif_vals[, "GVIF^(1/(2*Df))"]

high_vif <- vif_adj[vif_adj > 5]

high_vif
```

```
## log(carat)          y
##   8.772013   8.730964
```

The VIF is high for the y predictor, so I will remove that, since log(carat) was our transformed variable from earlier.

```
final_model <- lm(formula = log(price) ~ log(carat) + cut + color + clarity, data = sample_df)

vif_vals <- vif(test)
vif_adj <- vif_vals[, "GVIF^(1/(2*Df))"]

high_vif <- vif_adj[vif_adj > 5]

high_vif
```

```
## log(carat)          y
##   8.772013   8.730964
```

There are no more predictors with high VIF, so this would be the best model obtained by AIC backwards selection method.

3) Interpreting Confidence and Prediction Intervals

Using a sample observation with fixed values, I found its confidence and prediction intervals, then interpreted them.

```
# Create one combination of X's

one_obs <- data.frame(
  carat = 0.33,
  cut = "Ideal",
  color = "E",
  clarity = "VS2"
)

# Prediction Intervals
predict(final_model, newdata = one_obs, interval = "prediction")
```

```
##       fit      lwr      upr
## 1 6.61492 6.357267 6.872573
```

```
# Confidence Intervals
predict(final_model, newdata = one_obs, interval = "confidence")
```

```
##       fit     lwr     upr
## 1 6.61492 6.58842 6.64142
```

For a diamond with 0.33 carats, "Ideal" cut, "E" color, and "VS2" clarity, its expected log(price) is \$6.61 with a 95% prediction interval of [6.357, 6.873] and a 95% confidence interval of [6.588, 6.641].

4) Final Summary

First, I started by analyzing the diamonds dataset and its variables and their corresponding values. I then took a random sample of 1000 observations to use in the rest of our analyses. I checked for multicollinearity between any of the variables.

Next, I worked with the simple linear regression model, consisting of only the dependent variable, price, and a predictor variable, carat. I interpreted the output and proceeded to perform the necessary transformations on the variables to ensure it follows the assumptions of a linear regression model. I then continued adding more predictor variables to the simple linear regression model to turn it into a multiple linear regression model and increase its adjusted R squared.

Lastly, I performed backward elimination using AIC method to find the best model. I called the VIF function on it to test for multicollinearity between the predictors and dropped the X and Y predictor as necessary. Next, I created a test observation with sample values to calculate its confidence and prediction intervals.