Regression R Notebook

Subash Chandra and Abigail SOlomon

```
library(rlang)
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)

df <- diamonds
df$cut <- factor(df$cut)
df$color <-factor(df$color)
df$clarity <-factor(df$color)
```

##Train/Test split

```
set.seed(69)
i <- sample(1:nrow(df), nrow(df)*0.8, replace=FALSE)
train <- df[i,]
test<-df[-i,]
```

##Data Exploration

```
summary(train)
```

```
##      carat               cut        color    clarity      depth
##  Min.   :0.2000   Fair      : 1265   D:5377   D:5377   Min.   :44.00
##  1st Qu.:0.4000   Good      : 3964   E:7836   E:7836   1st Qu.:61.00
##  Median :0.7000   Very Good: 9602   F:7650   F:7650   Median :61.80
##  Mean   :0.7966   Premium   :11049   G:9051   G:9051   Mean   :61.75
##  3rd Qu.:1.0400   Ideal     :17272   H:6679   H:6679   3rd Qu.:62.50
##  Max.   :5.0100                     I:4349   I:4349   Max.   :79.00
##                                     J:2210   J:2210
##      table           price             x                y
##  Min.   :43.00   Min.   :  326   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:56.00   1st Qu.:  949   1st Qu.: 4.710   1st Qu.: 4.720
##  Median :57.00   Median : 2399   Median : 5.695   Median : 5.710
##  Mean   :57.45   Mean   : 3927   Mean   : 5.729   Mean   : 5.732
##  3rd Qu.:59.00   3rd Qu.: 5315   3rd Qu.: 6.540   3rd Qu.: 6.530
##  Max.   :79.00   Max.   :18823   Max.   :10.740   Max.   :58.900
```

```
##
##        z
##  Min.   :0.000
##  1st Qu.:2.910
##  Median :3.520
##  Mean   :3.536
##  3rd Qu.:4.030
##  Max.   :8.060
##
```

```
#Check for NAs

colSums(is.na(train))
```

```
##   carat     cut   color clarity   depth   table   price       x       y       z
##       0       0       0       0       0       0       0       0       0       0
```

```
#structure
str(train)
```

```
## tibble [43,152 x 10] (S3: tbl_df/tbl/data.frame)
##  $ carat  : num [1:43152] 0.52 0.93 0.77 0.4 1.02 1.29 0.43 0.74 0.34 1.51 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 4 3 2 4 5 3 5 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 3 3 2 7 1 1 4 4 5 2 ...
##  $ clarity: Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 3 3 2 7 1 1 4 4 5 2 ...
##  $ depth  : num [1:43152] 59.8 61.3 60.4 63 64.6 59.4 60.5 63.8 62.7 61.1 ...
##  $ table  : num [1:43152] 59 59 58 58 58 62 57 56 57 60 ...
##  $ price  : int [1:43152] 2251 2396 2975 631 6797 6935 1207 2423 596 9343 ...
##  $ x      : num [1:43152] 5.21 6.27 6 4.66 6.22 7.06 4.86 5.7 4.45 7.27 ...
##  $ y      : num [1:43152] 5.25 6.22 5.88 4.7 6.28 6.99 4.92 5.74 4.48 7.37 ...
##  $ z      : num [1:43152] 3.13 3.83 3.59 2.95 4.04 4.17 2.96 3.65 2.8 4.47 ...
```

```
names(train)
```

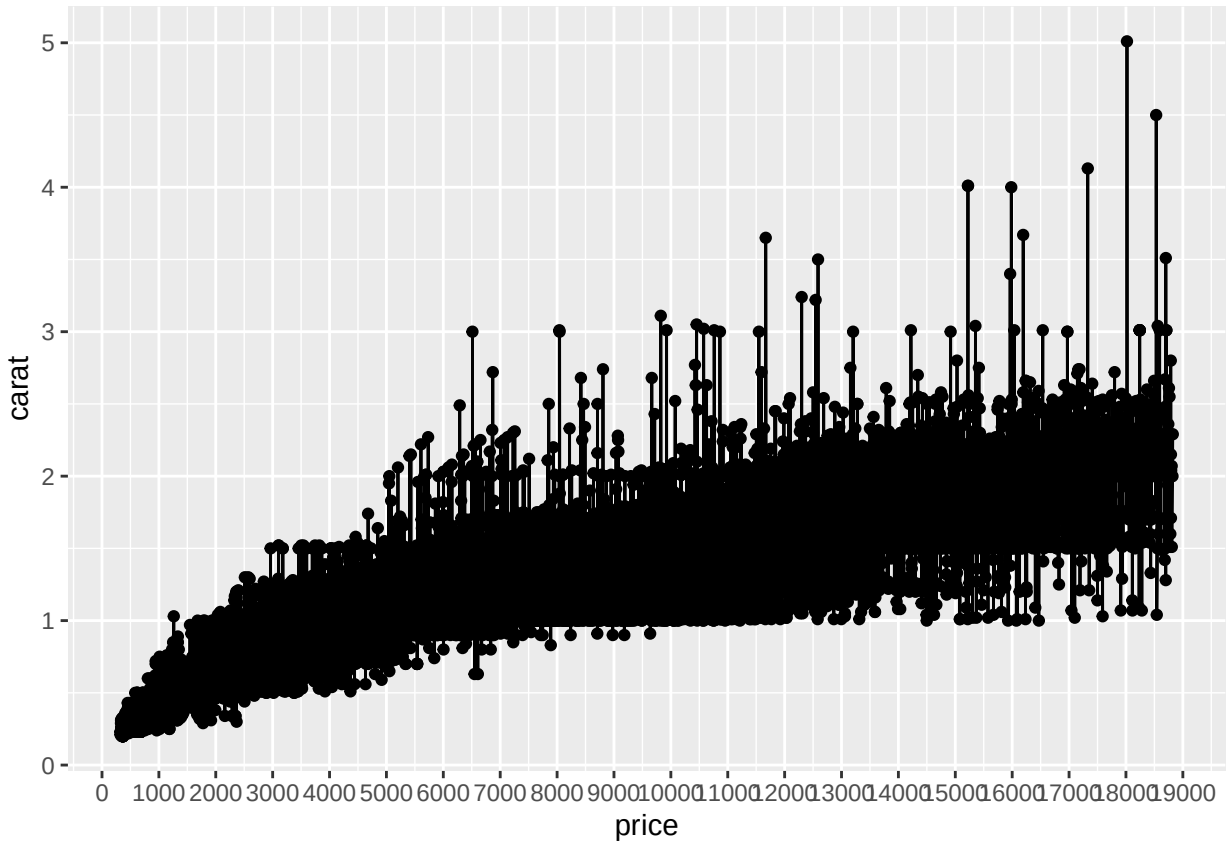
```
##  [1] "carat"   "cut"     "color"   "clarity" "depth"   "table"   "price"
##  [8] "x"       "y"       "z"
```

```
head(train)
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.52 Ideal     F     F        59.8    59  2251  5.21  5.25  3.13
## 2  0.93 Premium   F     F        61.3    59  2396  6.27  6.22  3.83
## 3  0.77 Premium   E     E        60.4    58  2975  6     5.88  3.59
## 4  0.4  Very Good J     J        63      58   631  4.66  4.7   2.95
## 5  1.02 Good      D     D        64.6    58  6797  6.22  6.28  4.04
## 6  1.29 Premium   D     D        59.4    62  6935  7.06  6.99  4.17
```

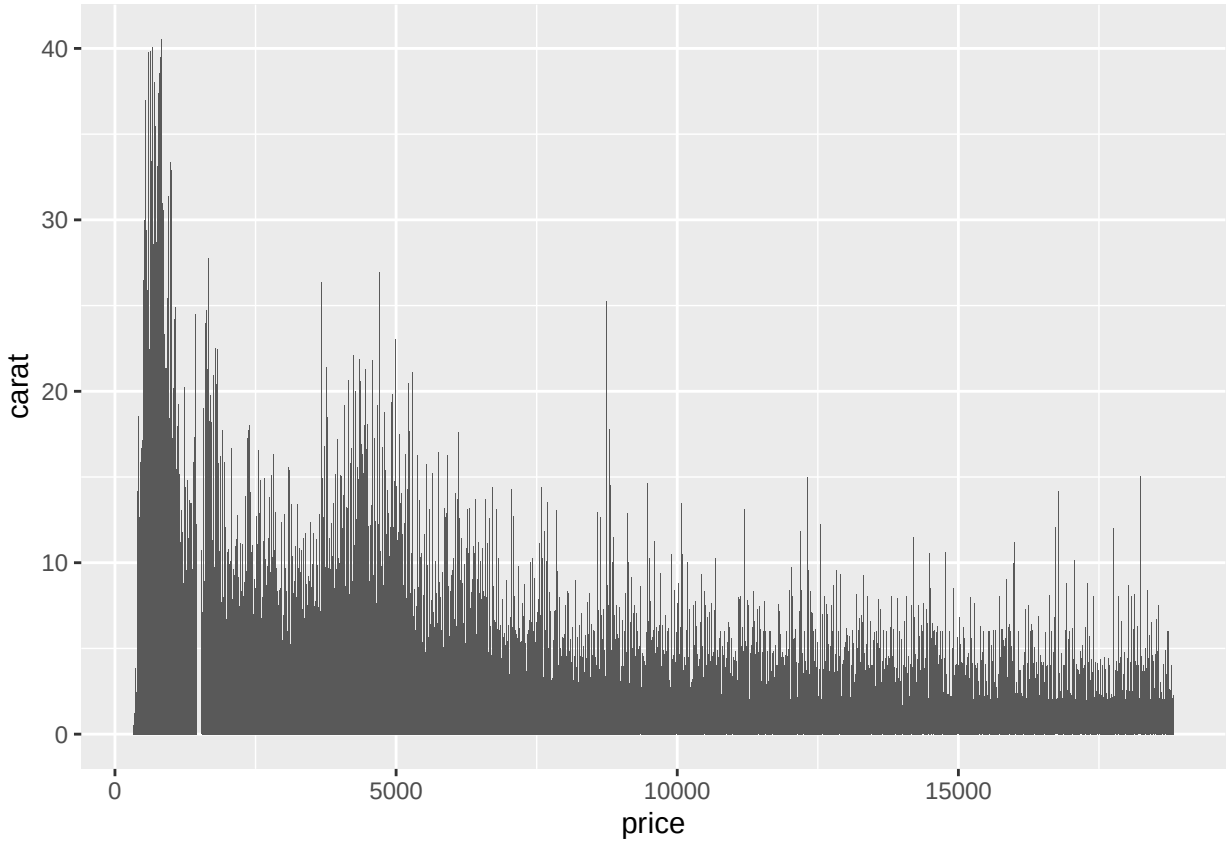
"Informative graphs"

```
#plot price vs carat
ggplot(df, aes(price, carat))+
  geom_point()+
  geom_line()+
  scale_x_continuous(n.breaks = 30)
```




```
#I am not really sure how to scale the axis in this, but you can see the general increase in price as c
```

```
ggplot(df,aes(price,carat))+geom_bar(stat = "identity")
```




```
# Here I think we see that there are many small diamonds that are still exorbitantly priced, and that c
```

##Linear Regression

```
lm1 <- lm(formula = price ~ carat, data = df)
summary(lm1)
```

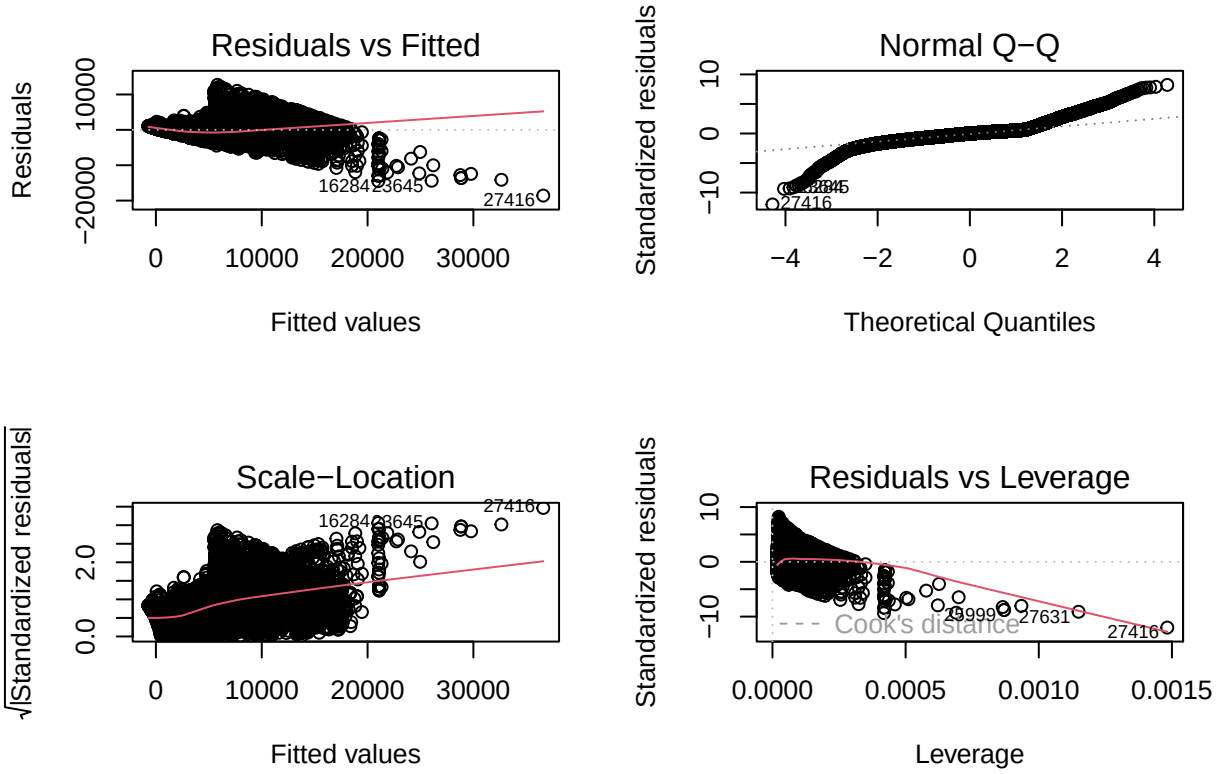
```
##
## Call:
## lm(formula = price ~ carat, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18585.3   -804.8    -18.9    537.4  12731.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2256.36      13.06  -172.8   <2e-16 ***
## carat        7756.43      14.07   551.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1549 on 53938 degrees of freedom
## Multiple R-squared:  0.8493, Adjusted R-squared:  0.8493
## F-statistic: 3.041e+05 on 1 and 53938 DF,  p-value: < 2.2e-16
```

- answer is ~7K per carat, which is ~accurate. real prices are anywhere from 2.5k-18k per carat.
- The Standard error is ~13 which is not great, but not terrible, and the P value is sufficiently low that

Carats seem to be a good predictor of the price of a diamond. A Multiple R^2 of 0.84 is pretty good, all things considered, and the low P value means that this model is pretty good, for the data that it was given.

##Plot Residuals * In the Residuals vs Fitted graph, we can see that there is a line, but it is not followed very well. The model is relatively linear though, so its not terrible. * In the Q-Q graph, we can see that between [-2,2], the values almost perfectly follow the lines. Not sure what exactly this could mean. At least it's continous? * The Scale-location graph has a fairly constant slope, but the heavy clumping is a problem. * There seems to be no red-dashed lines for the Cooks distance, so I'm not sure what to make of that. Maybe all the data is inside it. Maybe all the data is outside it. Nobody knows. * We have very low P Values, but this model is pretty terrible at actually predicting the value of the diamond.

```
par(mfrow = c(2,2))
plot(lm1)
```



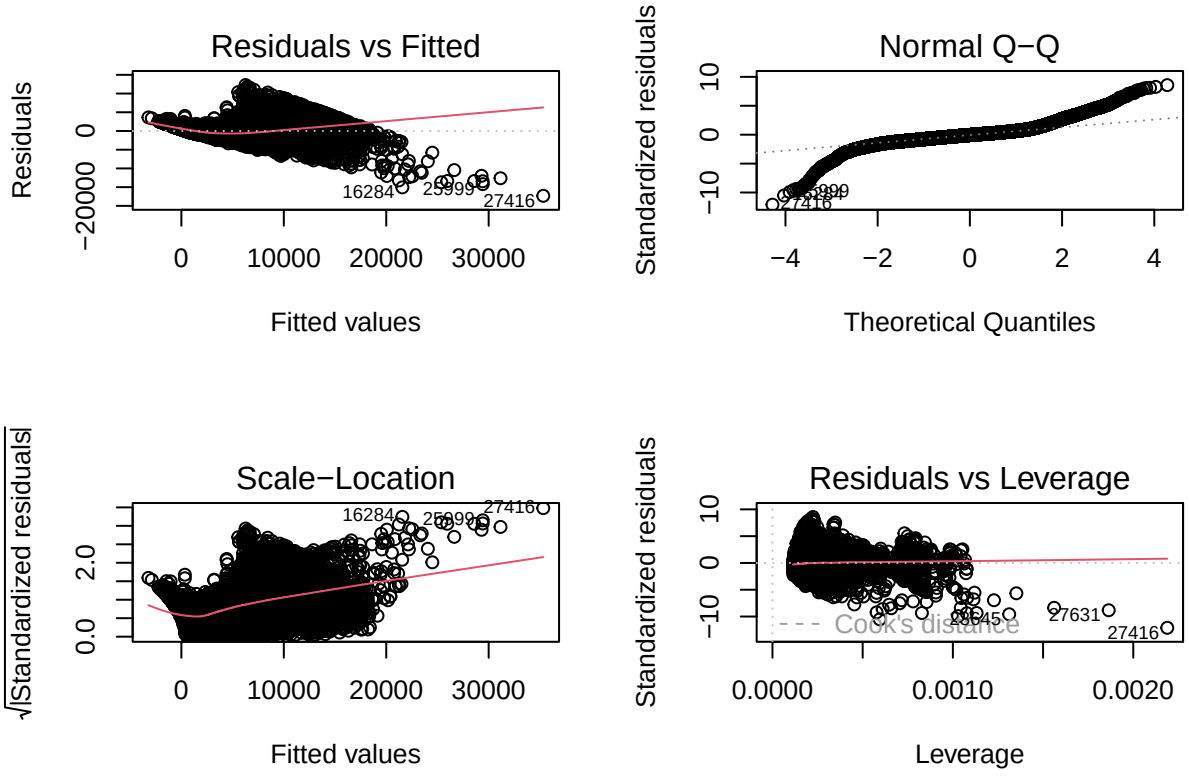
##Multiple Linear Regression

```
lm2 <- lm(price ~ carat + cut + clarity, data = df)
summary(lm2)
```

```
##
## Call:
## lm(formula = price ~ carat + cut + clarity, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17313.9   -751.2    -83.9    543.6  12273.0
##
```

```
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -3149.82      15.76 -199.905  < 2e-16 ***
## carat        8183.74      13.90  588.885  < 2e-16 ***
## cut.L        1243.35      24.74   50.260  < 2e-16 ***
## cut.Q        -531.75      21.93  -24.252  < 2e-16 ***
## cut.C         372.06      19.16   19.417  < 2e-16 ***
## cut^4          76.15      15.39    4.949 7.49e-07 ***
## clarity.L   -1579.17      21.72  -72.699  < 2e-16 ***
## clarity.Q    -732.85      19.86  -36.902  < 2e-16 ***
## clarity.C    -107.41      18.64   -5.763 8.32e-09 ***
## clarity^4      81.63      17.12    4.769 1.85e-06 ***
## clarity^5    -138.64      16.18   -8.568  < 2e-16 ***
## clarity^6    -161.09      14.68  -10.973  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1432 on 53928 degrees of freedom
## Multiple R-squared:  0.8711, Adjusted R-squared:  0.8711
## F-statistic: 3.315e+04 on 11 and 53928 DF,  p-value: < 2.2e-16
```

```
#residuals
par(mfrow = c(2,2))
plot(lm2)
```




```
#summary
summary(lm2)
```

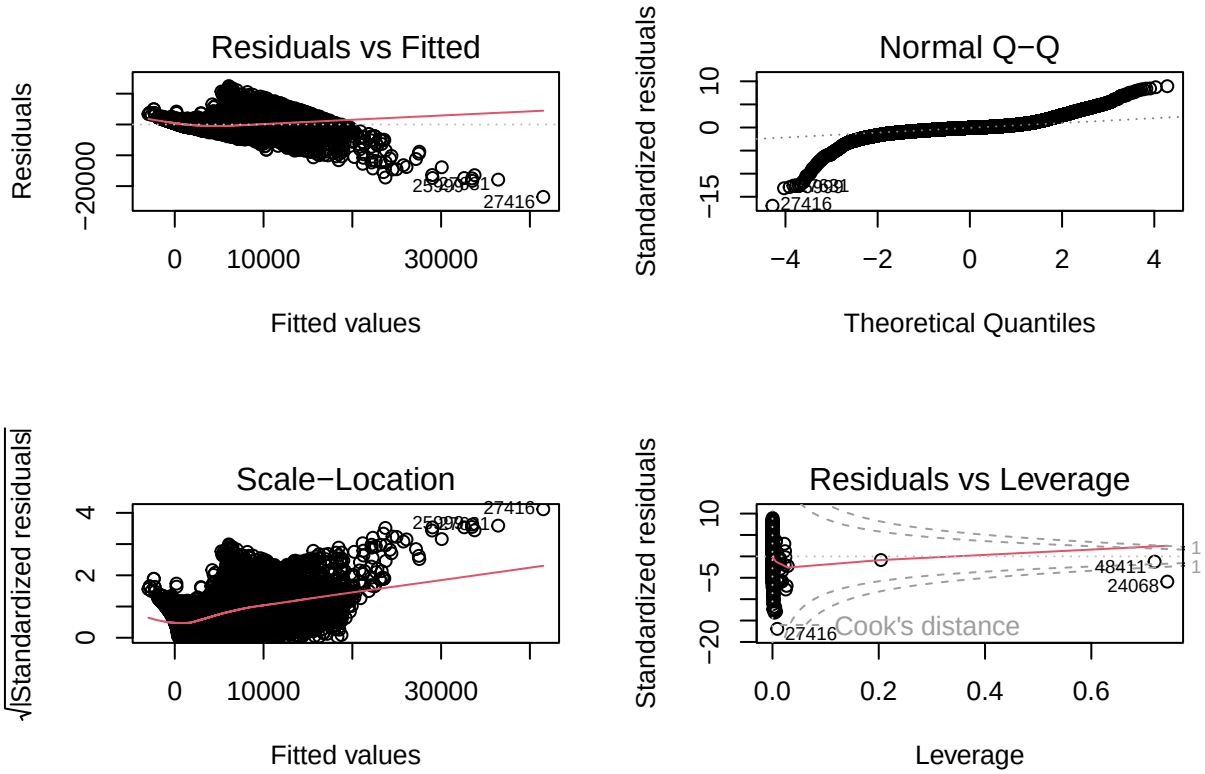
```
##
## Call:
## lm(formula = price ~ carat + cut + clarity, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17313.9   -751.2    -83.9    543.6  12273.0
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -3149.82      15.76 -199.905  < 2e-16 ***
## carat        8183.74      13.90  588.885  < 2e-16 ***
## cut.L        1243.35      24.74   50.260  < 2e-16 ***
## cut.Q        -531.75      21.93  -24.252  < 2e-16 ***
## cut.C         372.06      19.16   19.417  < 2e-16 ***
## cut^4          76.15      15.39    4.949 7.49e-07 ***
## clarity.L   -1579.17      21.72  -72.699  < 2e-16 ***
## clarity.Q    -732.85      19.86  -36.902  < 2e-16 ***
## clarity.C    -107.41      18.64   -5.763 8.32e-09 ***
## clarity^4      81.63      17.12    4.769 1.85e-06 ***
## clarity^5    -138.64      16.18   -8.568  < 2e-16 ***
## clarity^6    -161.09      14.68  -10.973  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1432 on 53928 degrees of freedom
## Multiple R-squared:  0.8711, Adjusted R-squared:  0.8711
## F-statistic: 3.315e+04 on 11 and 53928 DF,  p-value: < 2.2e-16
```

```
lm3 <- lm(price ~ carat + cut + clarity + x + y + z + clarity + depth + table, data = df)
summary(lm3)
```

```
##
## Call:
## lm(formula = price ~ carat + cut + clarity + x + y + z + clarity +
##     depth + table, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23489.4   -589.0   -105.3    392.0  12452.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11676.805    485.513  24.050  < 2e-16 ***
## carat       11327.076     59.404 190.678  < 2e-16 ***
## cut.L        1018.665     27.418  37.153  < 2e-16 ***
## cut.Q        -479.438     21.949 -21.844  < 2e-16 ***
## cut.C         318.957     18.994  16.793  < 2e-16 ***
## cut^4          42.142     15.220   2.769  0.00563 **
## clarity.L   -1646.082     21.181 -77.715  < 2e-16 ***
## clarity.Q    -772.283     19.329 -39.954  < 2e-16 ***
## clarity.C    -104.705     18.125  -5.777 7.66e-09 ***
## clarity^4      98.834     16.647   5.937 2.92e-09 ***
## clarity^5    -147.403     15.736  -9.367  < 2e-16 ***
## clarity^6    -151.888     14.274 -10.641  < 2e-16 ***
```

```
## x           -1408.436     40.264 -34.980  < 2e-16 ***
## y              38.952     23.819   1.635  0.10199
## z              34.994     41.255   0.848  0.39630
## depth        -117.277      5.562 -21.085  < 2e-16 ***
## table         -40.234      3.585 -11.222  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1393 on 53923 degrees of freedom
## Multiple R-squared:  0.8782, Adjusted R-squared:  0.8782
## F-statistic: 2.43e+04 on 16 and 53923 DF,  p-value: < 2.2e-16
```

```
#residuals
par(mfrow = c(2,2))
plot(lm3)
```




```
#summary
summary(lm3)
```

```
##
## Call:
## lm(formula = price ~ carat + cut + clarity + x + y + z + clarity +
##     depth + table, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23489.4   -589.0   -105.3    392.0  12452.7
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11676.805    485.513  24.050  < 2e-16 ***
## carat       11327.076     59.404 190.678  < 2e-16 ***
## cut.L        1018.665     27.418  37.153  < 2e-16 ***
## cut.Q        -479.438     21.949 -21.844  < 2e-16 ***
## cut.C         318.957     18.994  16.793  < 2e-16 ***
## cut^4          42.142     15.220   2.769  0.00563 **
## clarity.L   -1646.082     21.181 -77.715  < 2e-16 ***
## clarity.Q    -772.283     19.329 -39.954  < 2e-16 ***
## clarity.C    -104.705     18.125  -5.777 7.66e-09 ***
## clarity^4      98.834     16.647   5.937 2.92e-09 ***
## clarity^5    -147.403     15.736  -9.367  < 2e-16 ***
## clarity^6    -151.888     14.274 -10.641  < 2e-16 ***
## x           -1408.436     40.264 -34.980  < 2e-16 ***
## y              38.952     23.819   1.635  0.10199
## z              34.994     41.255   0.848  0.39630
## depth        -117.277      5.562 -21.085  < 2e-16 ***
## table         -40.234      3.585 -11.222  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1393 on 53923 degrees of freedom
## Multiple R-squared:  0.8782, Adjusted R-squared:  0.8782
## F-statistic: 2.43e+04 on 16 and 53923 DF,  p-value: < 2.2e-16
```

###Predictions

##Model 1

```
pred1 <- predict(lm1,newdata = test)
cor1 <- cor(pred1,test$price)
mse1 <- mean((pred1 - test$price)^2)
rmse1 <- sqrt(mse1)

head(pred1)
```

```
##           1           2           3           4           5           6
## -472.382688   -6.997151 -472.382688 -472.382688 -549.946944  148.131362
```

```
cor1
```

```
## [1] 0.92341
```

```
mse1
```

```
## [1] 2372968
```

```
rmse1
```

```
## [1] 1540.444
```

##Model 2

```
pred2 <- predict(lm2,newdata = test)
cor2 <- cor(pred2,test$price)
mse2 <- mean((pred2 - test$price)^2)
rmse2 <- sqrt(mse2)
```

```
head(pred2)
```

```
##          1          2          3          4          5          6
##  -841.4611 -1019.7138 -1091.9656 -1978.3187  -588.6226 -1323.6193
```

```
cor2
```

```
## [1] 0.9356444
```

```
mse2
```

```
## [1] 2006423
```

```
rmse2
```

```
## [1] 1416.483
```

##Model 3

```
pred3 <- predict(lm3,newdata = test)
cor3 <- cor(pred3,test$price)
mse3 <- mean((pred3 - test$price)^2)
rmse3 <- sqrt(mse3)

head(pred3)
```

```
##           1           2           3           4           5           6
##    5.854293  -661.099237  -407.313037 -1533.828924   160.271713 -1041.315595
```

```
cor3
```

```
## [1] 0.9396452
```

```
mse3
```

```
## [1] 1885668
```

```
rmse3
```

```
## [1] 1373.196
```

*All the models had approximately the same correlation of between 92.3% and 93.9%. All the MSE and RMSE's are hilariously high. This could be for a variety of reasons. I believe that the reason is that the diamond market is highly monopolized and diamonds are an artificially limited commodity, and that the reason is definitely not that I have badly overfit my model to the excessively large dataset that I found.

*Ultimately, the last model is the best because it is also able to take into account the color and other features of the Diamond to get a holistic view of it, and possibly even gaining value from knowing the X,Y,Z dimensions of the diamond, though this may actually negatively contribute due to the irregular nature of the shapes of diamonds, and the things they sit on.