Regression

Meinhard Capucao, Khang Thai

This data-set contains information about the price and various attributes of about 54,000 diamonds.

Here is a link to the dataset.

Linear regression is used for regression. This has its strengths and weaknesses describe below.

Linear regression consists of predictor values (x) and target values (y), where the goal is to find the relationship between x and y and be able to predict future values from this relationship. Simple linear regression has only one predictor variable. Adding more predictors makes it multiple linear regression. This algorithm works with quantitative data. Linear regression strength lies in that it is useful when the relationship between predictor and target values indicate a linear relationship. Although many algorithms are better, when we know the data is linear linear regression excels. Linear regression has low variance as well. However, most data will not be linear, causing linear regression to be less favorable in most cases. Linear regression also tends to have high bias.

Data reading and Installing Packages

First, install the diamond.csv data-set. Then, we look at the columns and their specific data types.

```
install.packages("tidyverse",repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/meinc/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\meinc\AppData\Local\Temp\RtmpeKmkAW\downloaded_packages
```

```
library(tidyverse)
```

```
## ── Attaching packages
## ───────────────────────────────────────
## tidyverse 1.3.2 ──
```

```
## ✔ ggplot2 3.3.6      ✔ purrr   0.3.4
## ✔ tibble  3.1.8      ✔ dplyr   1.0.10
## ✔ tidyr   1.2.1      ✔ stringr 1.4.1
## ✔ readr   2.1.2      ✔ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
diamonds <- read_csv("diamonds.csv")
```

```
## New names:
## Rows: 53940 Columns: 11
## ── Column specification
## ──────────────────────────────────────────────────────── Delimiter: "," chr
## (3): cut, color, clarity dbl (8): ...1, carat, depth, table, price, x, y, z
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

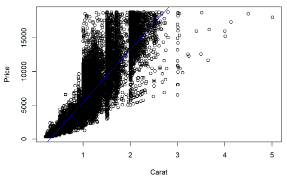
```
str(diamonds)
```

```
## spec_tbl_df [53,940 × 11] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ...1   : num [1:53940] 1 2 3 4 5 6 7 8 9 10 ...
##  $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : chr [1:53940] "Ideal" "Premium" "Good" "Premium" ...
##  $ color  : chr [1:53940] "E" "E" "E" "I" ...
##  $ clarity: chr [1:53940] "SI2" "SI1" "VS1" "VS2" ...
##  $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : num [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ...1 = col_double(),
##   ..   carat = col_double(),
##   ..   cut = col_character(),
##   ..   color = col_character(),
##   ..   clarity = col_character(),
##   ..   depth = col_double(),
##   ..   table = col_double(),
##   ..   price = col_double(),
##   ..   x = col_double(),
##   ..   y = col_double(),
##   ..   z = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Graphs and Plotting

Here, we plotted count as a function of carat to observe any relationships or trends. The abline function plots a general line through the graph.

```
plot(diamonds$price~diamonds$carat, xlab= "Carat", ylab= "Price")
abline(lm(diamonds$price~diamonds$carat), col = "blue")
```



The graph of diamond price as a function of carat is split into colors and made easier to read, where any carat values above 2.0 are colored blue. Anything below that is colored colored red.

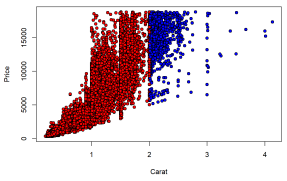
Divide into Train and Test Data

We will go ahead and divide the test data into training and test data, with 80% being training and 20% being test.

```
set.seed(1234)
i <- sample(1:nrow(diamonds), nrow(diamonds)*0.8, replace = FALSE)
train <- diamonds[i,]
test <- diamonds[-i,]
```

Creating Graphs

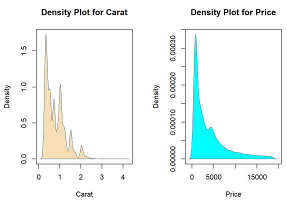
```
par(mfrow=c(1,1))
train$large <- factor(ifelse(train$carat>2,1,0))
plot(train$carat, train$price, xlab = "Carat", ylab = "Price", pch=21, bg=c("red","blue")[unclass(train$large)])
```



Then, a density plot is created for price and density. This reveals how diamond carats concentrate around values from 0.0 to 2.0. For the price, a similar trend is seen as well where they concentrate towards the left side.

```
par(mfrow=c(1,2))
d <-density(train$carat, na.rm = TRUE)
plot(d, main = "Density Plot for Carat", xlab = "Carat")
polygon(d, col="wheat", border="slategrey")

d <-density(train$price, na.rm = TRUE)
plot(d, main = "Density Plot for Price", xlab = "Price")
polygon(d, col="cyan", border="slategrey")
```



Data Exploration

Various R functions are used to further explore and understand the diamond data set. The initial linear regression model will predict price through carats, so the summary and range of both are inputted. Although the carat values range from 0.2 to 4.13, through the data from 3Q about 75% ofm carat values are below 1.0.

```
summary(train$carat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2000  0.4000  0.7000  0.7978  1.0400  4.1300
```

```
summary(train$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     326     954    2396    3931    5315   18823
```

```
range(train$carat)
```

```
## [1] 0.20 4.13
```

```
range(train$price)
```

```
## [1]   326 18823
```

We can see that carat and price are highly correlated through the cor function, which helps show the good fit for the upcoming linear regression model.

```
cor(train$price, train$carat, use = "complete.obs")
```

```
## [1] 0.9221438
```

Lastly, to fully understand where the carats lies and how much is in each category, the sum function is used for five different carat breakpoints.There are only three diamonds above four carats, so how much is the price for the biggest carat value?

The diamond with the greatest price does not match any of the diamonds with the greatest value, so now to find where it is.

```
sum(train$carat<=1)
```

```
## [1] 29161
```

```
sum(train$carat<=2 & train$carat>1)
```

```
## [1] 12484
```

```
sum(train$carat<=3 & train$carat>2)
```

```
## [1] 1486
```

```
sum(train$carat<=4 & train$carat>3)
```

```
## [1] 18
```

```
sum(train$carat>4)
```

```
## [1] 3
```

```
A <- subset(train, train$carat >= 4)
print(A)
```

```
## # A tibble: 4 × 12
##    ...1 carat cut       color clarity depth table price     x     y     z large
##   <dbl> <dbl> <chr>     <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <fct>
## 1 27131  4.13 Fair      H     I1       64.8    61 17329  10    9.85  6.43 1
## 2 26000  4.01 Premium   J     I1       62.5    62 15223  10.0  9.94  6.24 1
## 3 25999  4.01 Premium   I     I1       61      61 15223  10.1 10.1   6.17 1
## 4 26445  4    Very Good I     I1       63.3    58 15984  10.0  9.94  6.31 1
```

```
B <- subset(train, train$price == max(train$price))
print(B)
```

```
## # A tibble: 1 × 12
##    ...1 carat cut     color clarity depth table price     x     y     z large
##   <dbl> <dbl> <chr>   <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <fct>
## 1 27750  2.29 Premium I     VS2      60.8    60 18823   8.5  8.47  5.16 1
```

Building a Linear Regression

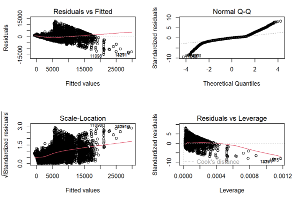
```
lm1 <- lm(price~carat, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = price ~ carat, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14560.1   -806.2    -14.2    547.3  12725.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2278.61      14.58  -156.3   <2e-16 ***
## carat        7783.58      15.72   495.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1545 on 43150 degrees of freedom
## Multiple R-squared:  0.8503, Adjusted R-squared:  0.8503
## F-statistic: 2.452e+05 on 1 and 43150 DF,  p-value: < 2.2e-16
```

Based on our summary, every one-unit increase in carat would result in a $7783 increase in price, with an error of about $15. The R^squared is 0.8503, which indicates a high goodness of fit, since the closer it is to 1 the better. This provides evidence we have a good model. Our F statistic is high and way greater than one, and considering that our p-value is low, our model has good confidence.

Residual Analysis

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



Residuals vs Fitted (1) Because there is a relatively equally spread residual around a horizontal line, our graph indicates that it there is a linear relationship.

Normal Q-Q (2) This plot shows a normal distribution since most of the residuals follow a straight line aside from a few outliers.

Scale-Location (3) The residuals appear to be randomly spread along the range of predictors. This assumes that there is equal variance.

Residuals vs Leverage (4) For the most part, our residual vs leverage graph indicates that there are a few influential cases. For example, case 1975 indicates that the price of the diamond was way less than it should be for its high carat value.

Building our Second Linear Regression Function

We build our second linear regression model with two new predictors, depth, and table. Essentially, this is multi-linear regression.

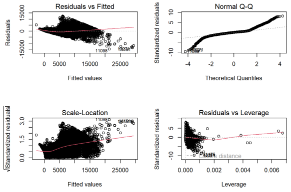
```
lm2 <- lm(price~carat+depth+table, data=train)
summary(lm2)
```

```
##
## Call:
## lm(formula = price ~ carat + depth + table, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14847.8   -786.5    -28.6    534.0  12479.3
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12921.925    436.868   29.58   <2e-16 ***
## carat        7887.317     15.816  498.68   <2e-16 ***
## depth        -148.909      5.385  -27.65   <2e-16 ***
## table        -105.963      3.501  -30.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1522 on 43148 degrees of freedom
## Multiple R-squared:  0.8547, Adjusted R-squared:  0.8547
## F-statistic: 8.459e+04 on 3 and 43148 DF,  p-value: < 2.2e-16
```

Residuals

We can see that for multi-linear regression, all four residuals are similar to last ones. They show that for the most part, the data fits the model well.

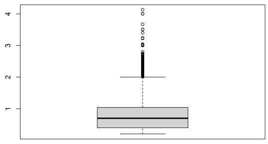
```
par(mfrow=c(2,2))
plot(lm2)
```



Boxplot to Find Outliers

We want to create a third model in hopes of achieving higher accuracy for our model. First, we wanted to **get rid of outliers** in general dictated by the $out function in R. We created a boxplot of our training data to see that the outliers are about 3. Then, we create a new vector for all the outliers to see the **minimum value** of the outliers, which is 2.01. We also see that there are **1507** outliers.

```
boxplot(train$carat)
```



```
outliers <- boxplot(train$carat, plot=FALSE)$out

min(outliers)
```

```
## [1] 2.01
```

```
length(outliers)
```

```
## [1] 1507
```

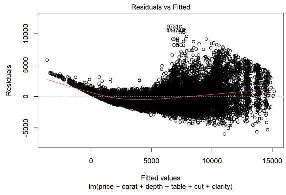
Trying to improve our New Linear Regression Model

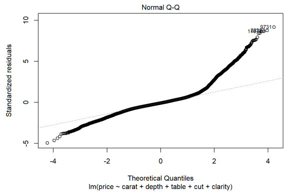
In our new linear regression model, we tried to improve it in two ways. First, we **added two new predictors** we thought were useful (**cut**, and **clarity**). Second, by **removing all outliers** through the subset() function, which was any carats above 2.01.

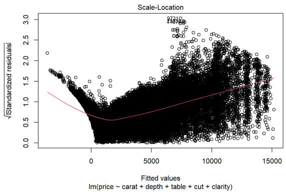
```
newdata <- subset(train, carat <= 2.01)
lm3 <- lm(price~carat+depth+table+cut+clarity, data=newdata)
summary(lm3)
```

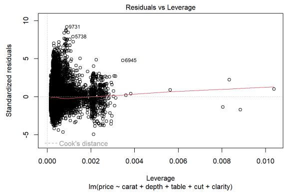
```
##
## Call:
## lm(formula = price ~ carat + depth + table + cut + clarity, data = newdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5992.0  -619.5  -105.0   467.0 11122.4
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2502.845    445.892  -5.613 2.00e-08 ***
## carat         8510.237     15.269 557.357  < 2e-16 ***
## depth          -38.006      4.869  -7.806 6.02e-15 ***
## table          -30.581      3.540  -8.638  < 2e-16 ***
## cutGood        520.203     41.150  12.642  < 2e-16 ***
## cutIdeal       740.613     41.090  18.024  < 2e-16 ***
## cutPremium     666.998     39.628  16.832  < 2e-16 ***
## cutVery Good   648.868     39.569  16.398  < 2e-16 ***
## clarityIF     4465.239     63.171  70.685  < 2e-16 ***
## claritySI1    2706.247     54.733  49.444  < 2e-16 ***
## claritySI2    1848.219     55.190  33.488  < 2e-16 ***
## clarityVS1    3584.647     55.706  64.350  < 2e-16 ***
## clarityVS2    3359.101     54.986  61.090  < 2e-16 ***
## clarityVVS1   4102.087     58.748  69.825  < 2e-16 ***
## clarityVVS2   4091.059     57.216  71.502  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1214 on 41991 degrees of freedom
## Multiple R-squared:  0.8842, Adjusted R-squared:  0.8842
## F-statistic: 2.291e+04 on 14 and 41991 DF,  p-value: < 2.2e-16
```

```
plot(lm3)
```









Analysis Between Models

```
anova(lm1, lm2)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	43150	102935581198	NA	NA	NA	NA
2	43148	99956089879	2	2979491319	643.0778	6.257507e-276

2 rows

This shows that models 1 and 2 are pretty similar since they have very similar RSS and Res. Df.

Results

Model one was a simple linear regression model that plotted the price of a diamond as a function of its carat. The R^2 was 0.8503, indicating that about 85% of the variance of price being studied is explained by the variance of carat. Model 2 was a multiple linear regression model that added depth and table as predictors for price. Comparing model one with model 2, the performance is slightly better since the R value was 0.8547, and the RSS was shown to be less.

We tried to improve the model by removing outliers that we thought affected the data through the subset functions, and introducing two more predictors that we thought would improve our model's prediction for price. Our third model had a 3% increase in accuracy, with the R^2 being 0.8842. The residual standard error also decreased by about 300, which indicates our regression model fits the dataset better.

```
pred1 <- predict(lm3, newdata=test)
cor1 <- cor(pred1, test$price)
mse1 <- mean((pred1-test$price)^2)
rmse1 <- sqrt(mse1)

print(paste('correlation: ', cor1))
```

```
## [1] "correlation:  0.945958996419197"
```

```
print(paste('mse: ', mse1))
```

```
## [1] "mse:  1666382.69132104"
```

```
print(paste('rmse: ', rmse1))
```

```
## [1] "rmse:  1290.88446087209"
```

Conclusion

We can conclude that there is a strong correlation between the price of a diamond and a combination of its carat, depth, table, clarity, and cut. Since there's a high positive correlation between the variables we can say that they are good predictors of the price.

Each data set is on average, far from the fitted line. Although the variables show good correlation as a predictor for price, there is a lot of variance that may stem from other factors.