Exploratory Analysis - Diamonds Tibble

UB Orengo

LIBRARIES USED

```
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.0.4

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(dplyr)
library(magrittr)

##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract

library(ggplot2)
library(stats)
library(scales)

##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

DATASET EXPLORED

```
head(diamonds)
```

```
## # 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 0.23  Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2 0.21  Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
## 3 0.23  Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4 0.290 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5 0.31  Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6 0.24  Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

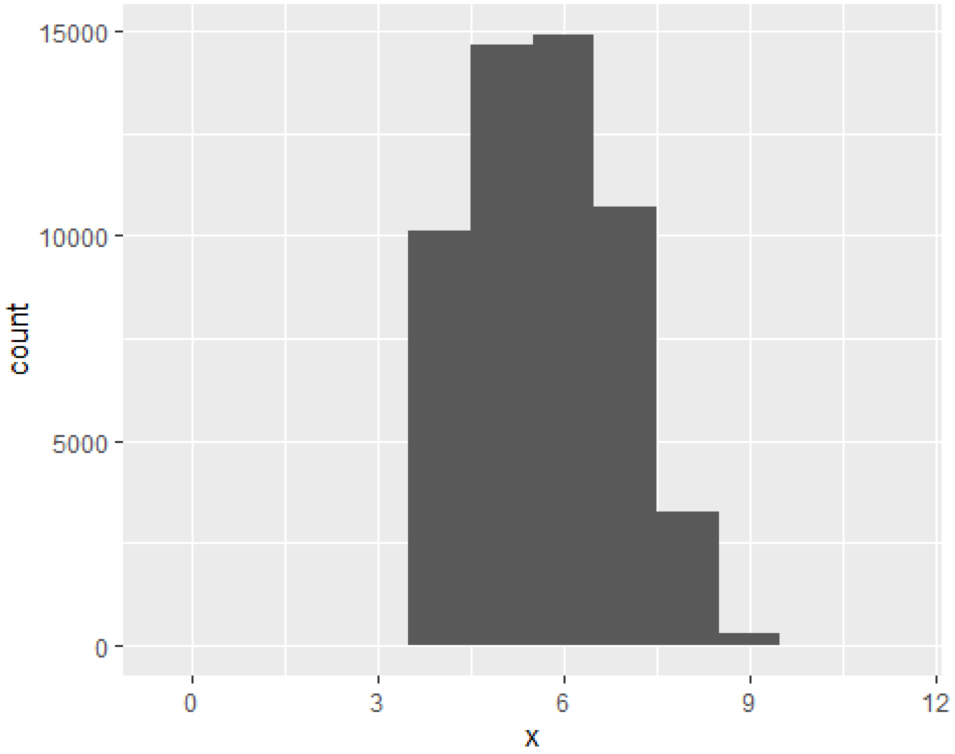
EXPLORING X Y Z VALUES

INTIAL HISTOGRAMS

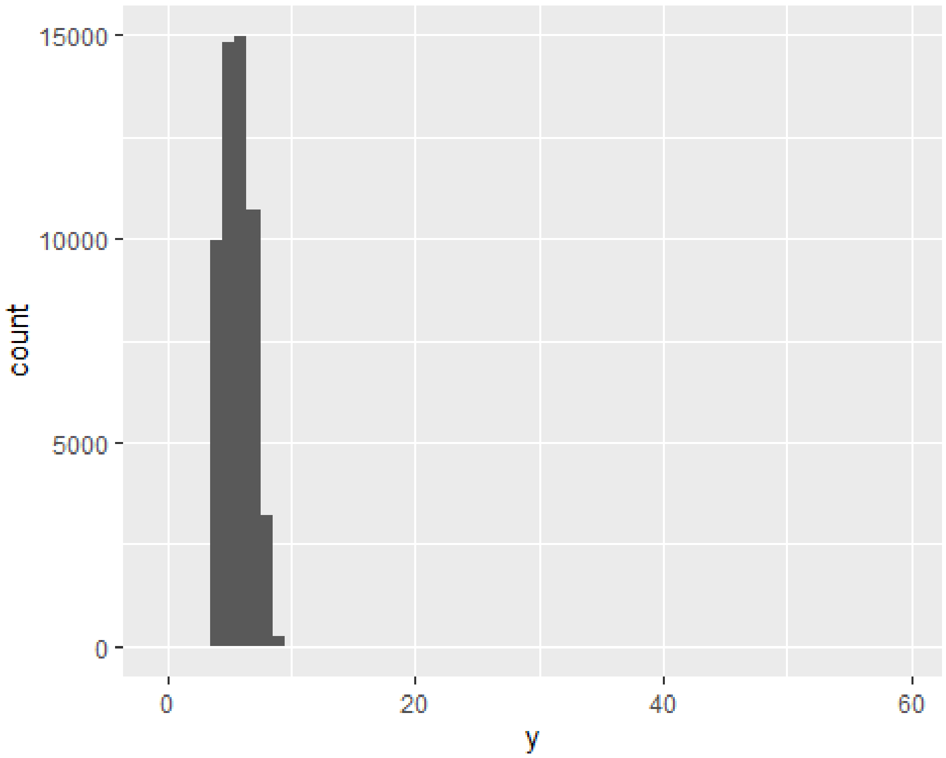
```
diamonds_hist <- ggplot(diamonds) +
                    geom_histogram(mapping = aes(x = x), binwidth = 1)
```

This histogram for the x value has a bit of a left skew

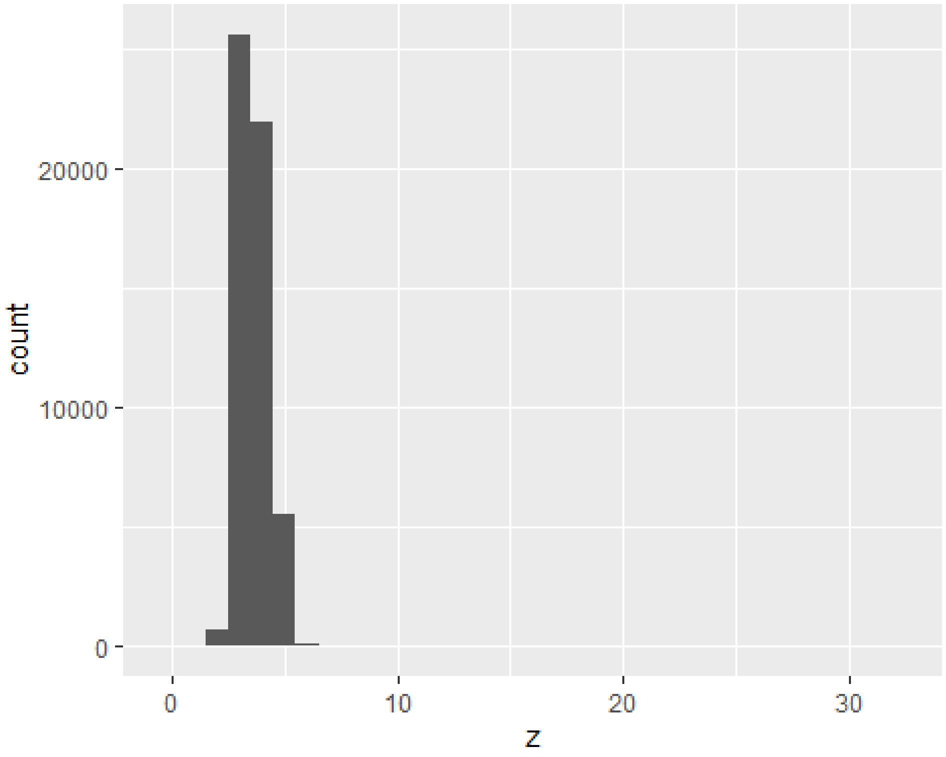
```
diamonds_hist2 <- ggplot(diamonds) +
                    geom_histogram(mapping = aes(x = y), binwidth = 1)

diamonds_hist3 <- ggplot(diamonds) +
                    geom_histogram(mapping = aes(x = z), binwidth = 1)
print(diamonds_hist)
```



```
print(diamonds_hist2)
```



```
print(diamonds_hist3)
```



We are unable to see the distribution properly of the y and z values because of outlier values

```
print(min(diamonds$y))

## [1] 0

print(max(diamonds$y))
```

```
## [1] 58.9
```

```
print(min(diamonds$z))
```

```
## [1] 0
```

```
print(max(diamonds$z))
```

```
## [1] 31.8
```

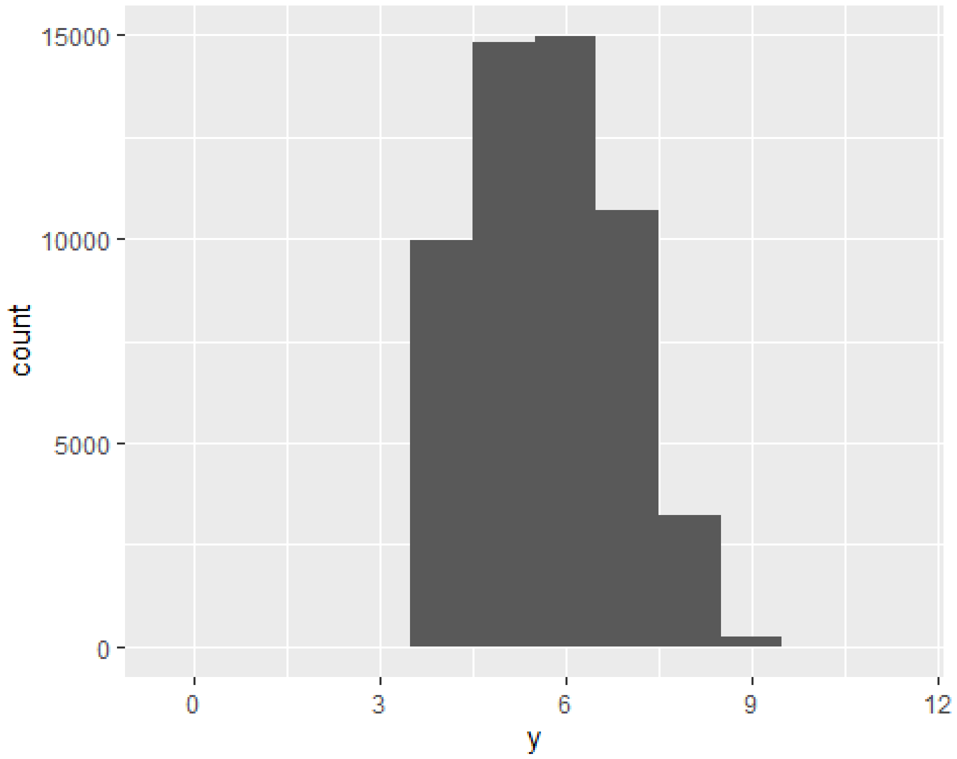
After looking at the max/min values and observing the histograms, we create a subset of the current table without the outliers

```
out_diamond_y <- diamonds[which(diamonds[,9]>10),]

out_diamond_z <- diamonds[which(diamonds[,10]>9),]

diamonds_subset_y <- subset(diamonds, diamonds$y <= 11)

diamonds_subset_z <- subset(diamonds, diamonds$z <= 8)
```

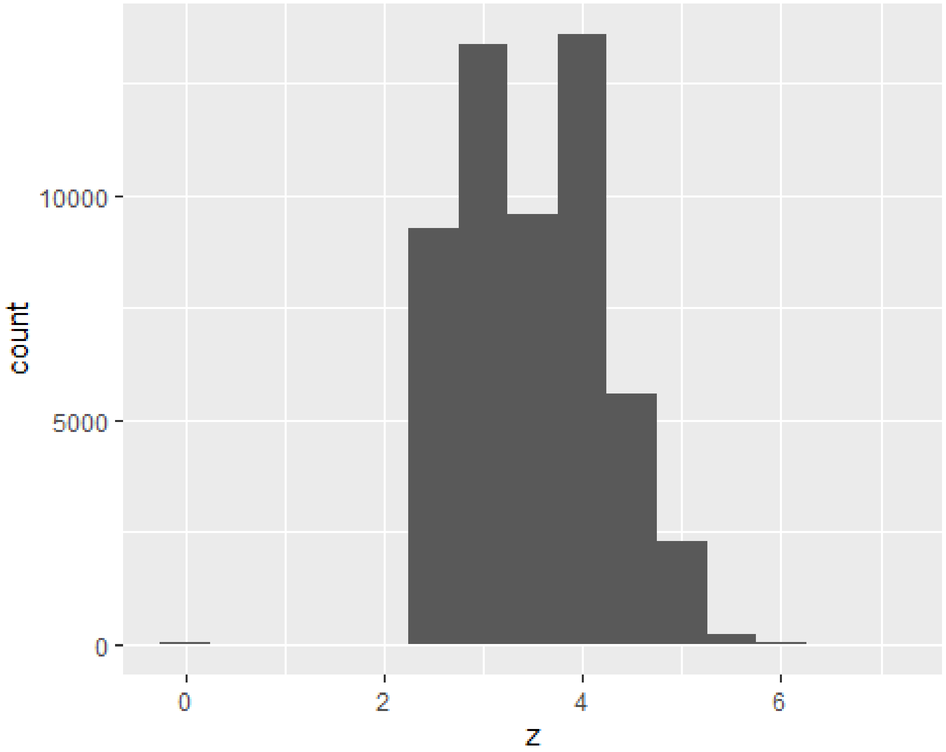
SUBSET GRAPHS – REMOVED OUTLIERS

We print new histograms, which are both semi-normally distributed, with new bins

```
diamonds_hist4 <- ggplot(diamonds_subset_y) +
  geom_histogram(mapping = aes(x = y), binwidth = 1)

diamonds_hist5 <- ggplot(diamonds_subset_z) +
  geom_histogram(mapping = aes(x = z), binwidth = .5)

print(diamonds_hist4)
```



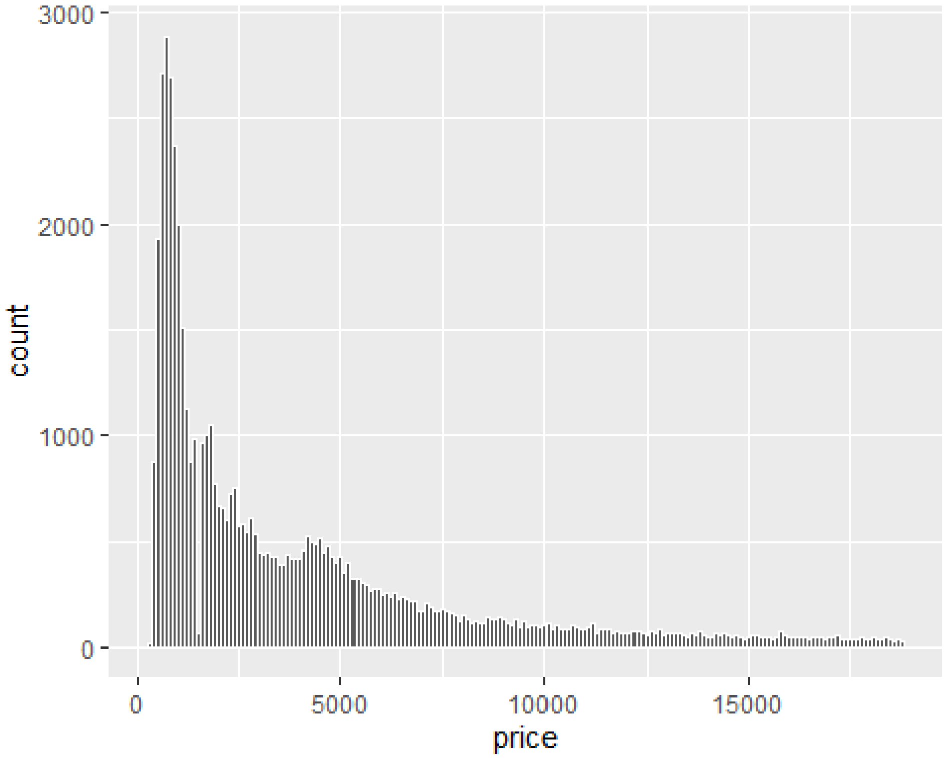
```
print(diamonds_hist5)
```



EXPLORING PRICE OF DIAMONDS

Histogram of Distribution of Price of Diamonds

```
diamonds_price <- ggplot(diamonds) +
                    geom_histogram(mapping = aes(x = price), binwidth = 100,
color="white")

print(min(diamonds$price))

## [1] 326

print(max(diamonds$price))

## [1] 18823

print(diamonds_price)
```



### It seems that a large portion of the diamonds in the dataset are between the first 10 bins, which would be 0-1000$

COMPARISION OF .99 CARAT DIAMONDS TO 1 CARAT DIAMONDS

```
carat_count <- diamonds[which(diamonds[,1] == 0.99),]
view(carat_count)
```

23 rows in diamonds$carat = 0.99, there are 23 diamonds in the dataset that equal .99 carats

```
carat_count2 <- diamonds[which(diamonds[,1] == 1),]
view(carat_count2)
```

1,558 rows in diamonds$carat = 1. there are 1,558 diamonds in the dataset that equal 1 carat.

The difference may be attributed to values being rounded up to be able to price the diamonds at 1 carat, as opposed to .99 carats.

Summary Table – Average Price by Carat

```
avg_price_by_carat <- diamonds %>%
                        group_by(carat) %>%
                        summarize(avg_price = mean(price))

## `summarise()` ungrouping output (override with `.groups` argument)

price_99_carat <- avg_price_by_carat[which(avg_price_by_carat[,1] == .99),]
price_1_carat <- avg_price_by_carat[which(avg_price_by_carat[,1] == 1),]

print(avg_price_by_carat)

## # A tibble: 273 x 2
##    carat avg_price
##    <dbl>     <dbl>
##  1 0.2        365.
##  2 0.21       380.
##  3 0.22       391.
##  4 0.23       486.
##  5 0.24       505.
##  6 0.25       551.
##  7 0.26       551.
##  8 0.27       575.
##  9 0.28       580.
## 10 0.290      601.
## # ... with 263 more rows
```

```
print(price_99_carat)
```

```
## # A tibble: 1 x 2
##   carat avg_price
##   <dbl>     <dbl>
## 1  0.99     4406.
```

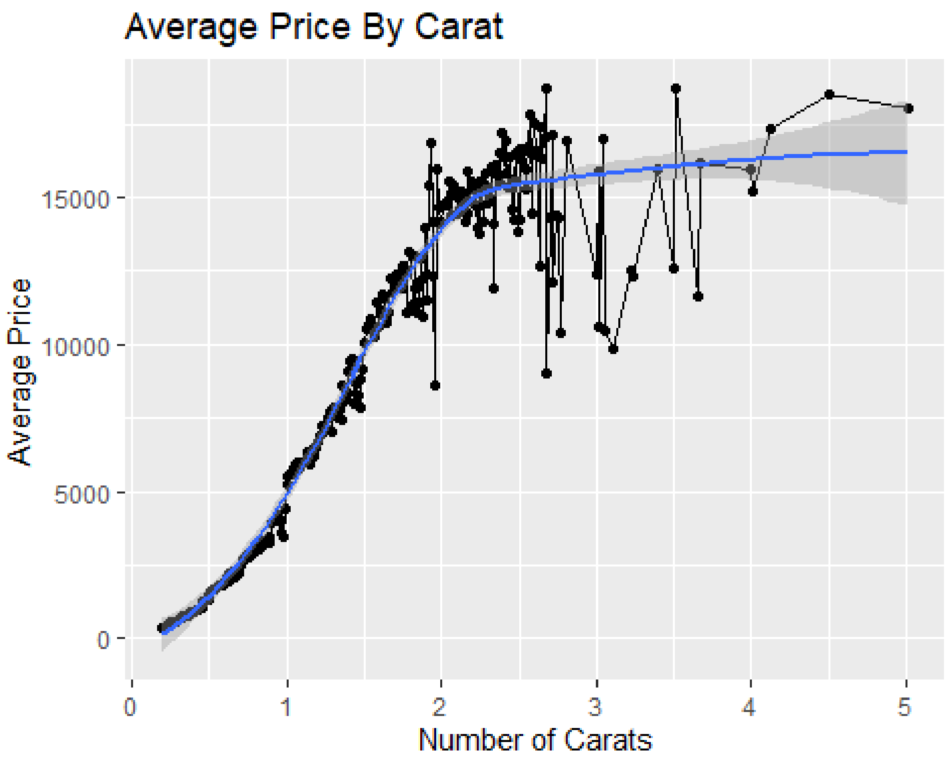
```
print(price_1_carat)
```

```
## # A tibble: 1 x 2
##   carat avg_price
##   <dbl>     <dbl>
## 1     1     5242.
```

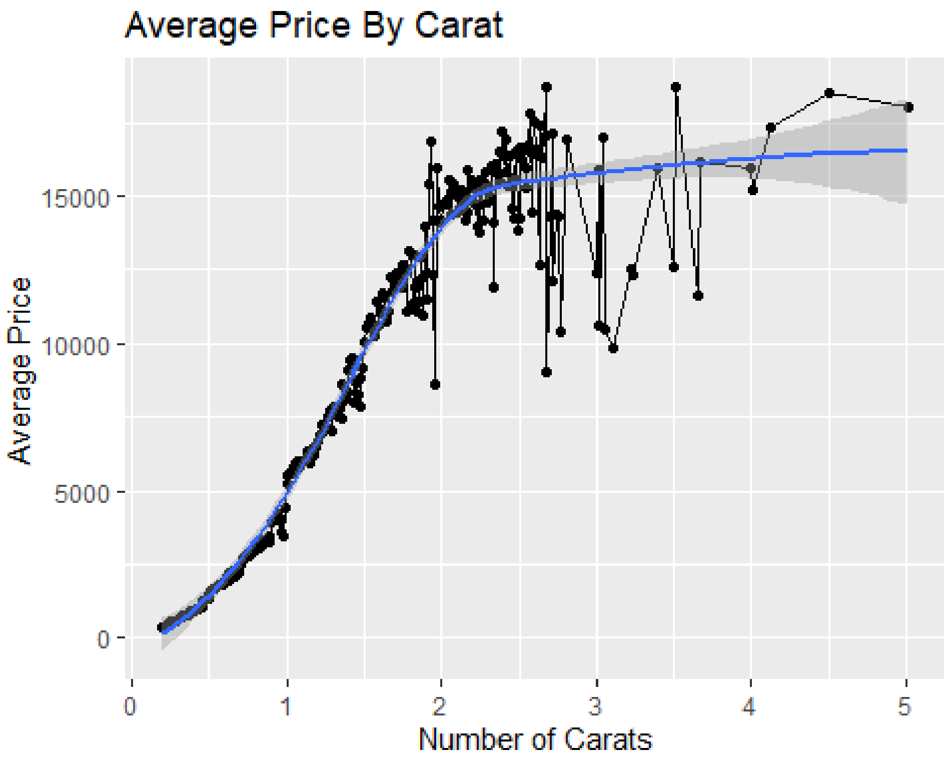
Line Plot of Average Price by Number of Carats

```
carat_plot <- ggplot(data = avg_price_by_carat, mapping = aes(x = carat, y = avg_price))
+
                  geom_line() +
                  geom_point() +
                  geom_smooth() +
                  ggtitle("Average Price By Carat") +
                  xlab("Number of Carats") + ylab("Average Price")

print(carat_plot) + ggtitle("Average Price By Carat") + xlab("Number of Carats") +
ylab("Average Price")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



You can see a distinct slope between .99 carat and 1 carat

```
print(price_99_carat)

## # A tibble: 1 x 2
##   carat avg_price
##   <dbl>     <dbl>
## 1  0.99     4406.
```

```
print(price_1_carat)
```

```
## # A tibble: 1 x 2
##   carat avg_price
##   <dbl>     <dbl>
## 1     1     5242.
```

```
price_diff <- price_1_carat[,2]-price_99_carat[,2]
print(price_diff)
```

```
##   avg_price
## 1  835.4159
```

There is an over 800$ difference between the average price of a .99 carat diamond and a 1 carat diamond