Diamond

Zhanhao Zhang

2/17/2020

Required Packages:

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------- tidyverse 1.2.1.9000 --

## v ggplot2 3.2.0     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

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

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

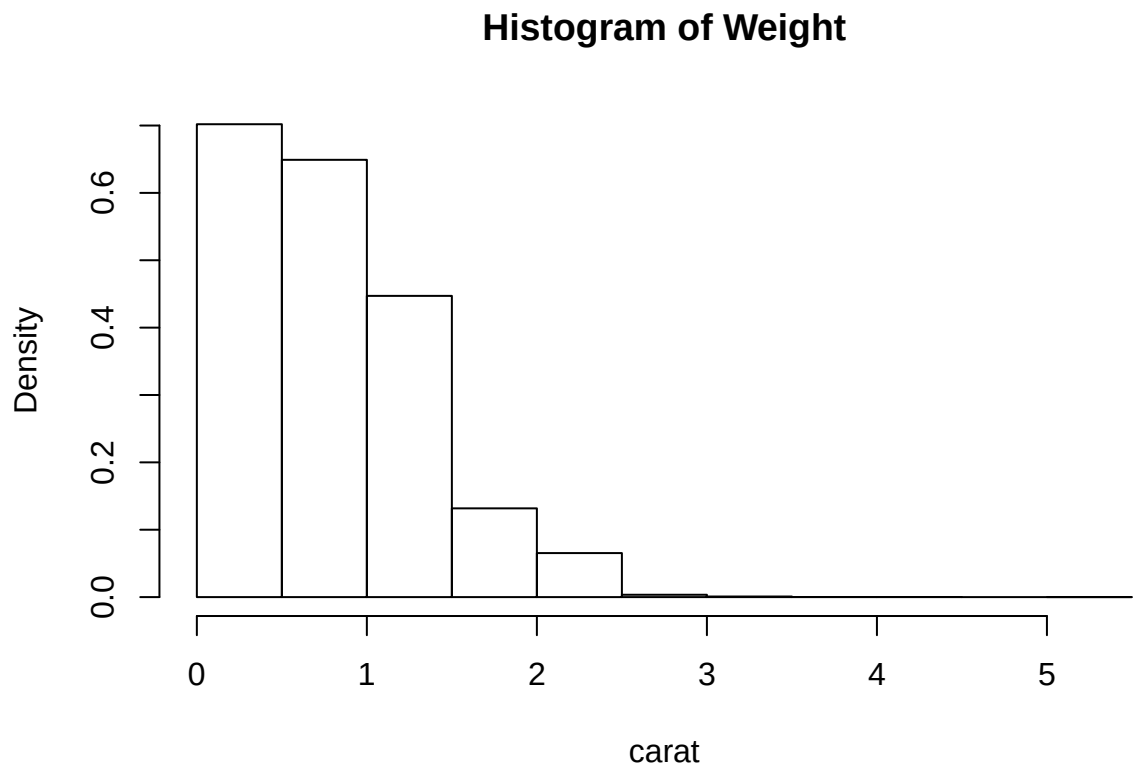
Load Data:

```
diamond <- read.csv("diamonds.csv")
summary(diamond)
```

```
##        X             carat               cut        color
##  Min.   :    1   Min.   :0.2000   Fair     : 1610   D: 6775
##  1st Qu.:13486   1st Qu.:0.4000   Good     : 4906   E: 9797
##  Median :26970   Median :0.7000   Ideal    :21551   F: 9542
##  Mean   :26970   Mean   :0.7979   Premium  :13791   G:11292
##  3rd Qu.:40455   3rd Qu.:1.0400   Very Good:12082   H: 8304
```

```
##  Max.   :53940   Max.   :5.0100                    I: 5422
##                                                    J: 2808
##     clarity          depth           table           price
##  SI1    :13065   Min.   :43.00   Min.   :43.00   Min.   :  326
##  VS2    :12258   1st Qu.:61.00   1st Qu.:56.00   1st Qu.:  950
##  SI2    : 9194   Median :61.80   Median :57.00   Median : 2401
##  VS1    : 8171   Mean   :61.75   Mean   :57.46   Mean   : 3933
##  VVS2   : 5066   3rd Qu.:62.50   3rd Qu.:59.00   3rd Qu.: 5324
##  VVS1   : 3655   Max.   :79.00   Max.   :95.00   Max.   :18823
##  (Other): 2531
##        x                y                z
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
##  Median : 5.700   Median : 5.710   Median : 3.530
##  Mean   : 5.731   Mean   : 5.735   Mean   : 3.539
##  3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040
##  Max.   :10.740   Max.   :58.900   Max.   :31.800
##
```

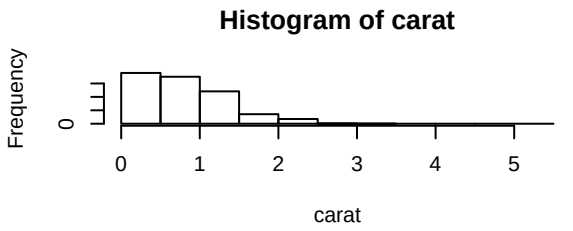
```
hist(diamond$carat, probability = T, main = "Histogram of Weight",
     xlab = "carat", breaks = 10)
```

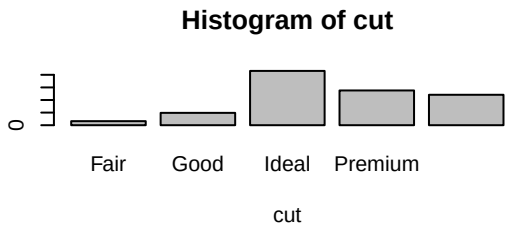


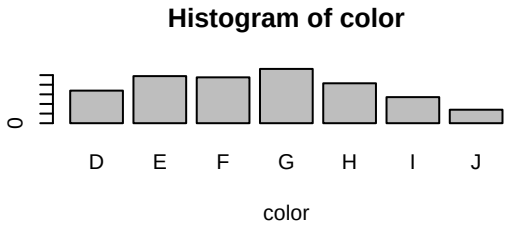
Histograms:

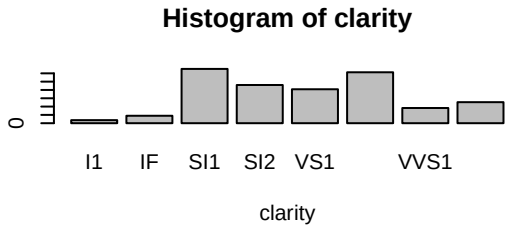
```
plot_histograms <- function(var = "all", num_plot_row = 2){
  plot_single_histogram <- function(single_var){
```

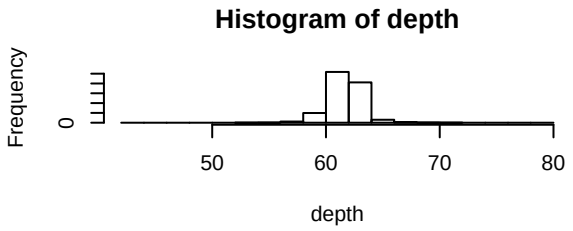
```
    if(is.numeric(diamond[[single_var]])){
      hist(diamond[[single_var]],
                main = paste("Histogram of", single_var), xlab = single_var)
    } else{
      plot(diamond[[single_var]], xlab = single_var,
                main = paste("Histogram of", single_var))
    }
  }

  plot_multiple_histograms <- function(multiple_var){
    len <- length(multiple_var)
    if(ceiling(len / num_plot_row) > 3){
      par(mfrow = c(3, num_plot_row))
      for(i in 1:(3 * num_plot_row)){
        plot_single_histogram(multiple_var[i])
      }
      par(mfrow = c(1, 1))
      plot_multiple_histograms(multiple_var[(3 * num_plot_row + 1) : len])
    } else{
      n.rows <- ceiling(length(var) / num_plot_row)
      par(mfrow = c(n.rows, num_plot_row))
      for(i in 1:len){
        plot_single_histogram(multiple_var[i])
      }
      par(mfrow = c(1, 1))
    }
  }

  if(length(var) > 1){
    plot_multiple_histograms(var)
  } else if(var == "all"){
    plot_multiple_histograms(colnames(diamond)[2:ncol(diamond)])
  } else{
    plot_single_histogram(var)
  }
}

plot_histograms()
```

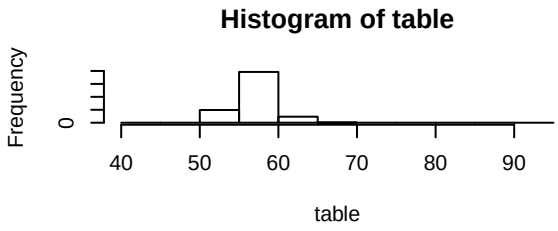

Histogram of carat
Frequency
0
0
1
2
3
4
5
carat

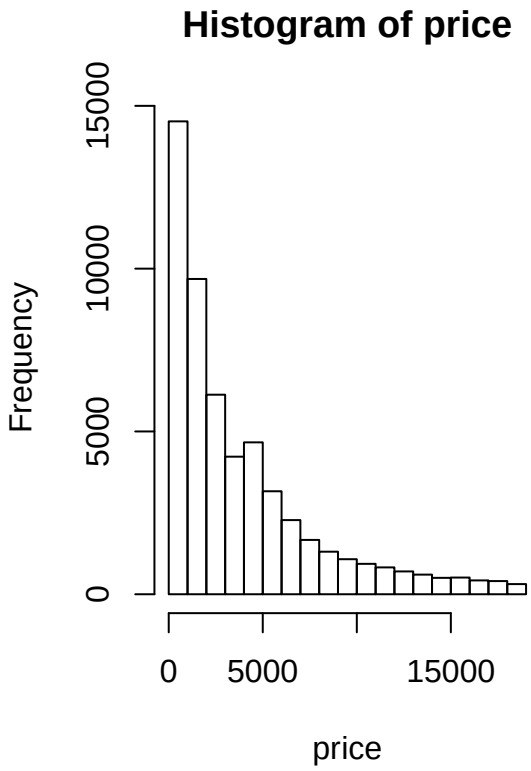

Histogram of cut
0
Fair
Good
Ideal
Premium
cut

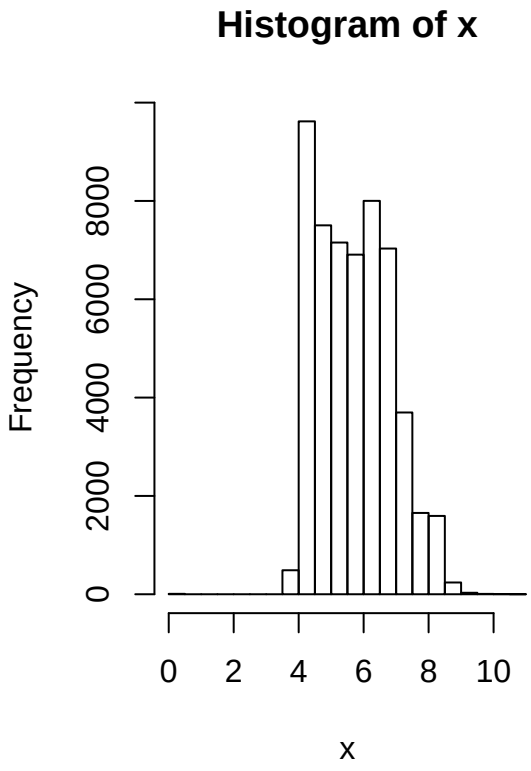

Histogram of color
0
D
E
F
G
H
I
J
color

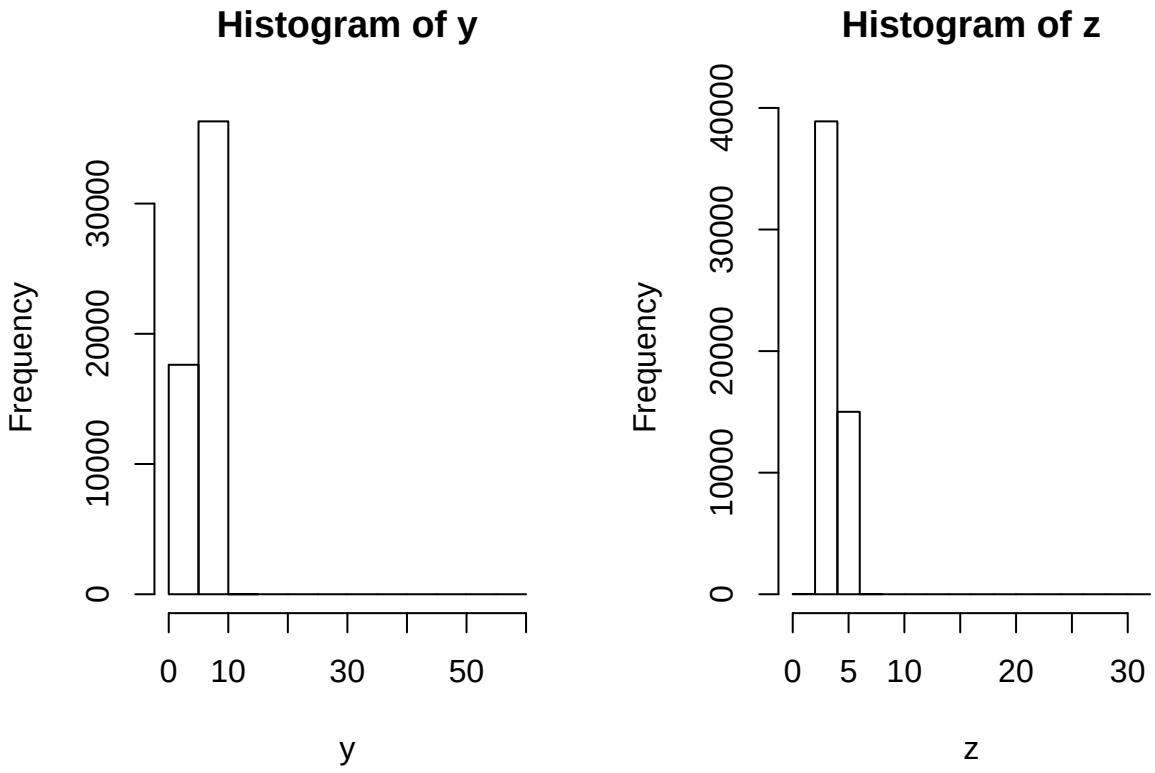

Histogram of clarity
0
I1
IF
SI1
SI2
VS1
VVS1
clarity


Histogram of depth
Frequency
0
50
60
70
80
depth

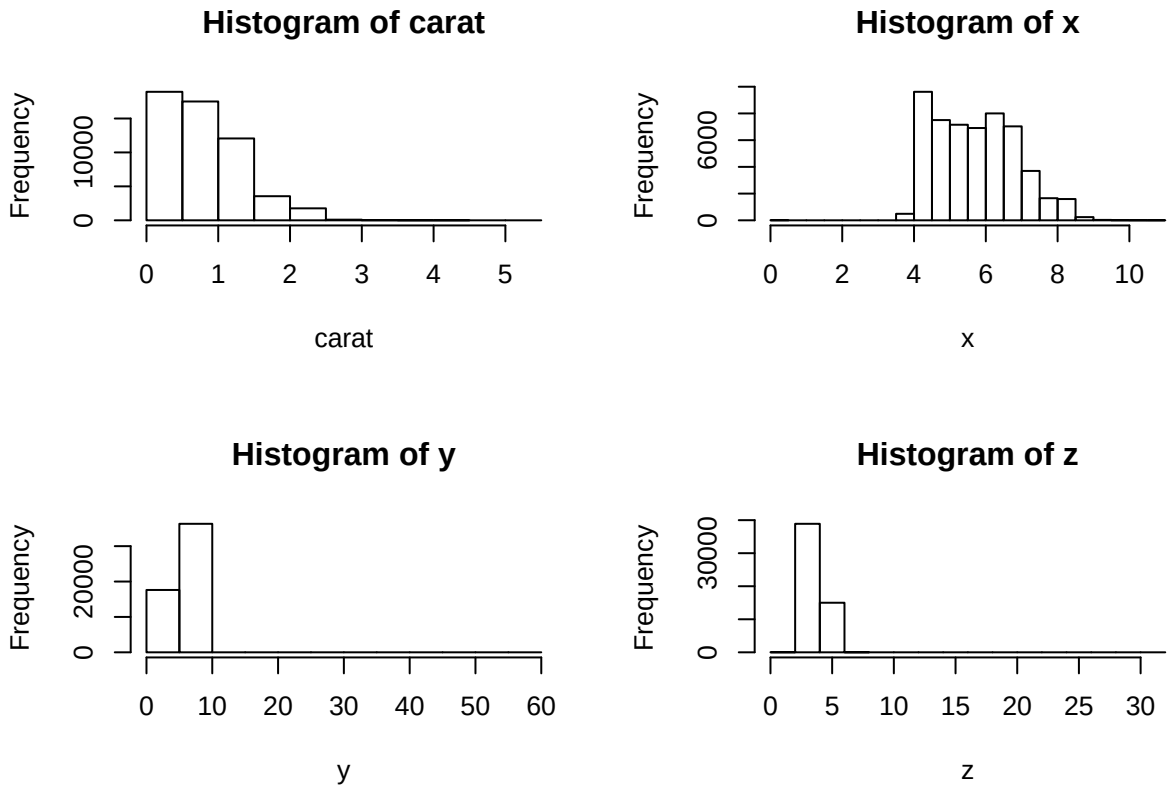

Histogram of table
Frequency
0
40
50
60
70
80
90
table


Histogram of price
Frequency
0
5000
10000
15000
0
5000
15000
price


Histogram of x
Frequency
0
2000
4000
6000
8000
0
2
4
6
8
10
x

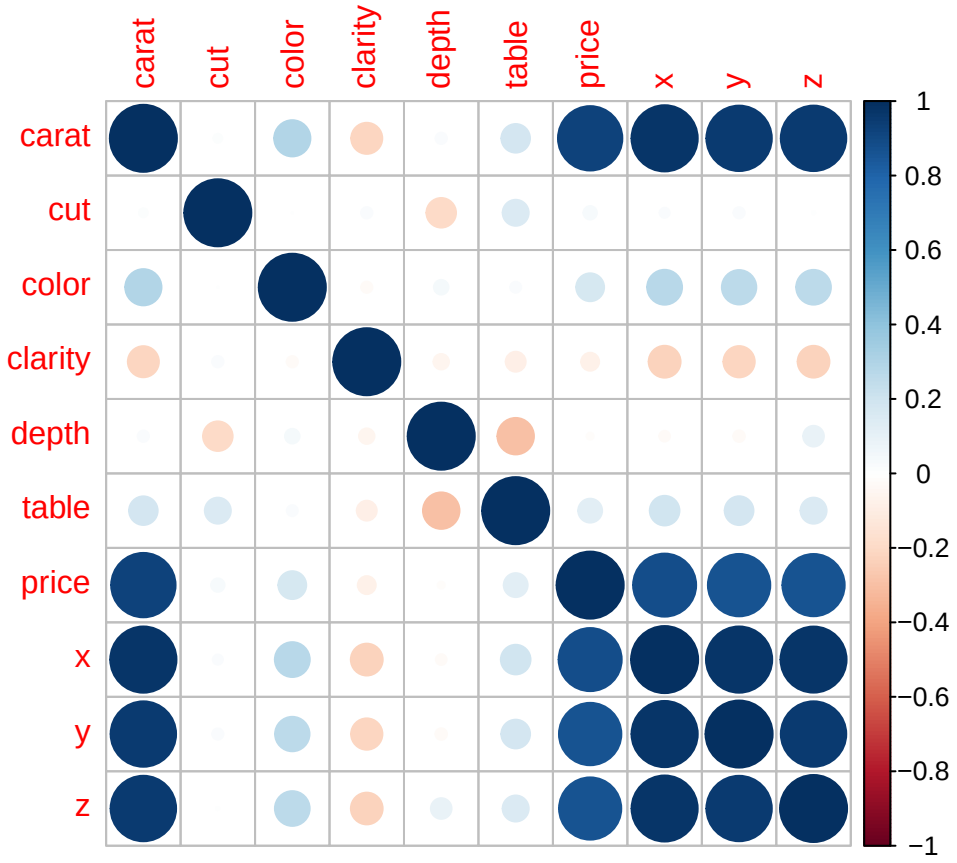



```
plot_histograms("carat")
plot_histograms(c("carat", "x", "y", "z"))
```



Correlation Matrices and Plots:

```
plot_correlations <- function(var = "all"){
  if(length(var) > 1){
    df <- c()
    for(v in var){
      df <- cbind(df, as.numeric(diamond[[v]]))
    }
    colnames(df) <- var
    cor(df)
    corrplot(cor(df), method = "circle")
  } else if(var == "all"){
    plot_correlations(colnames(diamond)[2:ncol(diamond)])
  } else{
    stop("You should give at least 2 predictors!")
  }
}

plot_correlations()
```



Compress Data Frame:

```
get_compressed_df <- function(dataframe, predictor_to_compress,
                              compressed_vector, bin.width){
  df_new <- c()
  for(v in compressed_vector){
    tmp <- dataframe[dataframe[[predictor_to_compress]] <= v + bin.width &
                     dataframe[[predictor_to_compress]] >= v - bin.width,]
    tmp[[predictor_to_compress]] <- v
    df_new <- rbind(data.frame(df_new), data.frame(tmp))
  }
  return(df_new)
}

compressed_df <- get_compressed_df(diamond, "carat", c(0.3, 0.5, 0.8, 1, 1.5), 0.03)
summary(compressed_df)
```

```
##        X             carat             cut       color       clarity
##  Min.   :    4   Min.   :0.30   Fair      : 684   D:2831   VS2    :4987
##  1st Qu.:13819   1st Qu.:0.30   Good      :2220   E:3967   SI1    :4807
##  Median :27892   Median :0.50   Ideal     :7839   F:3699   VS1    :2997
##  Mean   :26667   Mean   :0.69   Premium   :4933   G:4470   SI2    :2961
##  3rd Qu.:40074   3rd Qu.:1.00   Very Good:4603   H:2989   VVS2   :2046
##  Max.   :53929   Max.   :1.50                    I:1581   VVS1   :1436
##                                                  J: 742   (Other):1045
##      depth           table           price             x
```

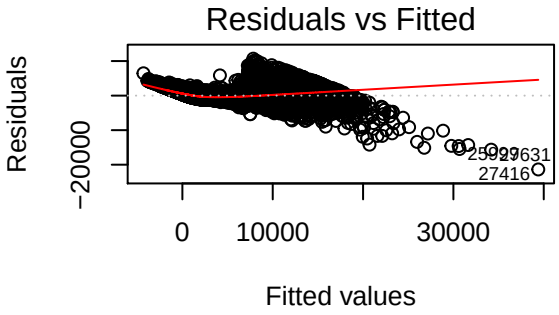
```
##  Min.   :43.00   Min.   :44.00   Min.   :  334   Min.   :0.000
##  1st Qu.:61.10   1st Qu.:56.00   1st Qu.:  780   1st Qu.:4.370
##  Median :61.90   Median :57.00   Median : 1758   Median :5.170
##  Mean   :61.84   Mean   :57.45   Mean   : 3337   Mean   :5.479
##  3rd Qu.:62.60   3rd Qu.:59.00   3rd Qu.: 4868   3rd Qu.:6.400
##  Max.   :79.00   Max.   :79.00   Max.   :18806   Max.   :7.750
##
##        y               z
##  Min.   : 0.000   Min.   : 0.00
##  1st Qu.: 4.380   1st Qu.: 2.71
##  Median : 5.180   Median : 3.20
##  Mean   : 5.483   Mean   : 3.39
##  3rd Qu.: 6.400   3rd Qu.: 3.98
##  Max.   :31.800   Max.   :31.80
##
```

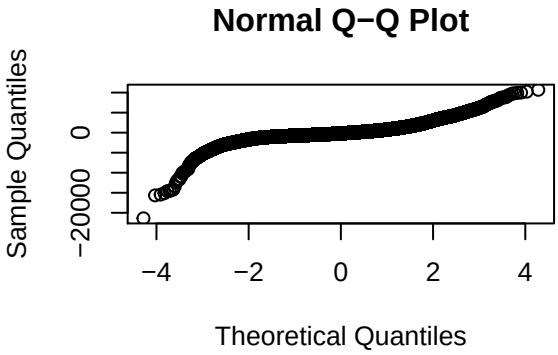
```
head(compressed_df)
```

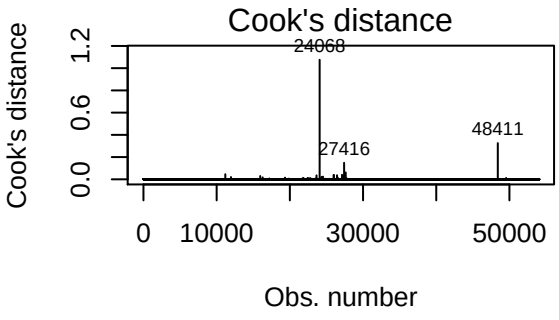
```
##     X carat     cut color clarity depth table price    x    y    z
## 4   4   0.3 Premium     I     VS2  62.4    58   334 4.20 4.23 2.63
## 5   5   0.3    Good     J     SI2  63.3    58   335 4.34 4.35 2.75
## 11 11   0.3    Good     J     SI1  64.0    55   339 4.25 4.28 2.73
## 14 14   0.3   Ideal     J     SI2  62.2    54   344 4.35 4.37 2.71
## 16 16   0.3 Premium     E      I1  60.9    58   345 4.38 4.42 2.68
## 17 17   0.3   Ideal     I     SI2  62.0    54   348 4.31 4.34 2.68
```

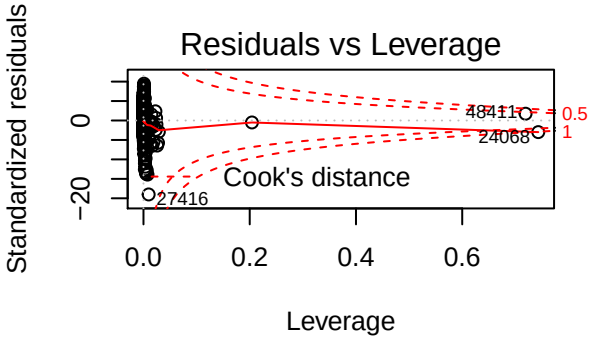
Model Evaluation – Assumptions on Residual:

```
plot_residual_assumptions <- function(lm){
  par(mfrow = c(2, 2))
  plot(lm, which = 1)
  qqnorm(lm$residuals)
  plot(lm, which = 4)
  plot(lm, which = 5)
  par(mfrow = c(1, 1))
}
plot_residual_assumptions(lm(price ~ . - X, data = diamond))
```


Residuals vs Fitted
Residuals
−20000
0
10000
30000
Fitted values
25929
27416


Normal Q−Q Plot
Sample Quantiles
−20000
0
−4
−2
0
2
4
Theoretical Quantiles


Cook's distance
Cook's distance
0.0
0.6
1.2
0
10000
30000
50000
Obs. number
24068
27416
48411


Residuals vs Leverage
Standardized residuals
−20
0
0.0
0.2
0.4
0.6
Leverage
Cook's distance
0.5
1
48411
24068
27416