CDS 101 – Final Project Report

JAD - John D Villanueva, Akshaya Pabbisetty, David Brown

December 09, 2025

Contents

1. Problem Definition

We are going to be investigating a variety of factors that may affect the price of a diamond. Diamonds are highly sought-after gems, used in all sorts of precious jewelry. With a dataset containing information on many aspects of diamonds, we will determine the particular effects of the clarity, carats, and cut quality of any diamond upon its price. We believe that the clarity and carat of a diamond will be strong predictors of pricing while the cut quality will be a weaker predictor.

2. Data Acquisition & Description

The dataset is, quite simply, called "diamonds". It was obtained from the website Kaggle at the link https://www.kaggle.com/datasets/ayeshaseherr/diamonds.

There are ten variables creating each column; three are categorical, these being the cut quality, color grade, and clarity rating. The remaining seven are carat, depth, table, x, y, z, and price.

The variables x, y, and z are the length, width, and depth in millimeters, carats are a measure of diamond weight, table is the top width percentage of each diamond, and price is standardly measured in USD.

There are just under 54000 entries/rows in the dataset, making it very large overall.

```r
library(readr)
diamond_df <- read_csv("data/diamonds.csv")
```

```
## Rows: 53940 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): cut, color, clarity
## dbl (7): carat, depth, table, x, y, z, price
##
## 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.
```

3. Data Cleaning & Preprocessing

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v purrr     1.1.0
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   4.0.0     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
diamond_1 <- diamonds %>%
  select(carat, cut, clarity, price) %>%
  mutate(
    cut = as.character(cut),
    clarity = as.character(clarity)
  )
```

4. Exploratory Data Analysis (EDA)

```r
library(broom)
library(tidyverse)
library(modelr)
Diamond_model_a <- lm(price ~ carat, data = diamond_1)
Diamond_model_a %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df   logLik     AIC     BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>    <dbl>   <dbl>   <dbl>
## 1     0.849         0.849 1549.   304051.       0     1 -472730. 945467. 945493.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
Diamond_model_u <- lm(price ~ cut, data = diamond_1)
Diamond_model_u %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic   p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1    0.0129        0.0128 3964.      176. 8.43e-150     4 -523426. 1.05e6 1.05e6
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
Diamond_model_y <- lm(price ~ clarity, data = diamond_1)
Diamond_model_y %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic   p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1    0.0272        0.0270 3935.      215. 1.93e-316     7 -523033. 1.05e6 1.05e6
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
diamond_ma_df <- diamonds %>%
  add_predictions(Diamond_model_a) %>%
  add_residuals(Diamond_model_a)
diamond_mu_df <- diamonds %>%
  add_predictions(Diamond_model_u) %>%
  add_residuals(Diamond_model_u)
```

```
diamond_my_df <- diamonds %>%
  add_predictions(Diamond_model_y) %>%
  add_residuals(Diamond_model_y)
```

When glancing at each model, the squared residual for the Carat Model is much closer to one in comparison to Cut's model and Clarity's model. As well as the creastion of the dataframes of said models.

5. Visualization Quality and Storytelling

- Explain why your plot types are appropriate.

We used a heat map to visualize the relationship and the density of weight in Carats in comparison to the price of a diamond. Due to the quantity of entries, there is a high density that would not be able to be explored efficiently with a point plot. The heat map shows the points with a higher density of diamonds and lesser density of diamonds without getting muddled.

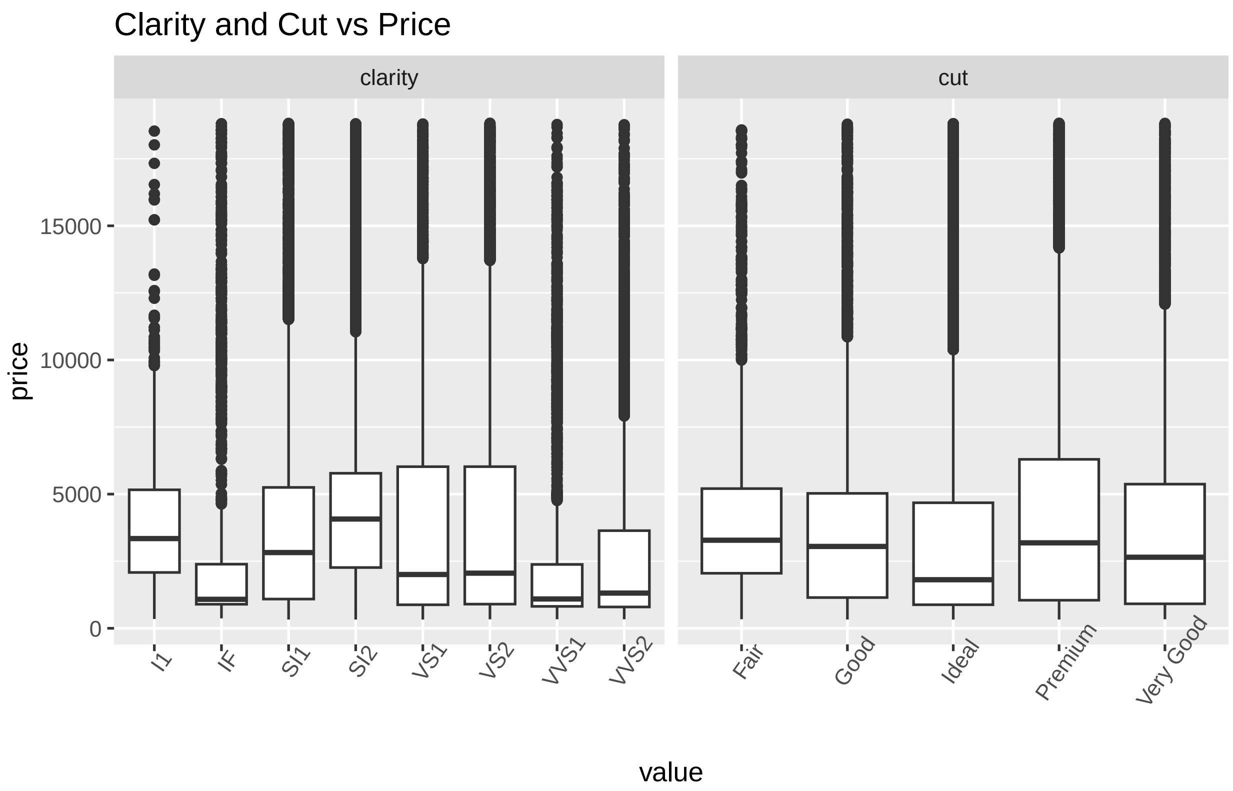
- Comment on labels, legends, colors, and overall readability.

The heatmaps show the density in different colors, making it more readable and understandable of the relationship. All graphs are labeled with accurate titles and labels.

- Mention any steps you took to make plots accessible and interpretable.

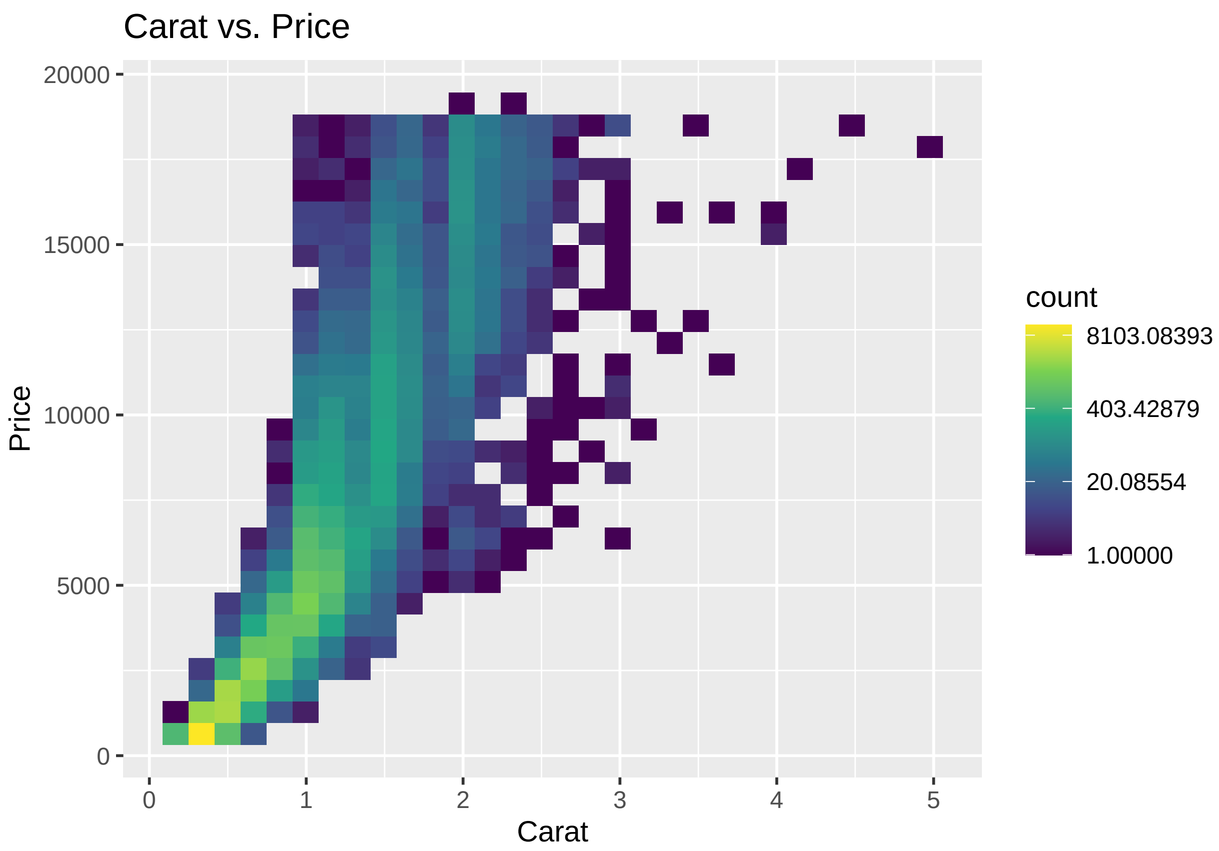
We chose to utilize a heat map instead of a normal point plot to make it more interpretable and accessible due to the sheer amount of entries.

```
diamond_1 %>%
  pivot_longer(cols = cut:clarity,
               names_to = "measurement",
               values_to = "value") %>%
  ggplot() +
  geom_boxplot(aes(x = value, y = price)) +
  facet_wrap(~ measurement, scales = "free_x") +
  labs(title = "Clarity and Cut vs Price") +
  theme(axis.text.x = element_text(angle = 55))
```



```
diamond_1 %>%
  ggplot() +
  geom_bin2d(aes(x = carat, y = price)) +
  labs(
    title = "Carat vs. Price",
    x = "Carat",
    y = "Price"
  ) +
scale_fill_viridis_c(trans = "log")
```

```
## `stat_bin2d()` using `bins = 30`. Pick better value `binwidth`.
```



6. Modeling Approach

Explain how you framed the problem and which models you chose:

- Type of task (regression, classification, etc.).

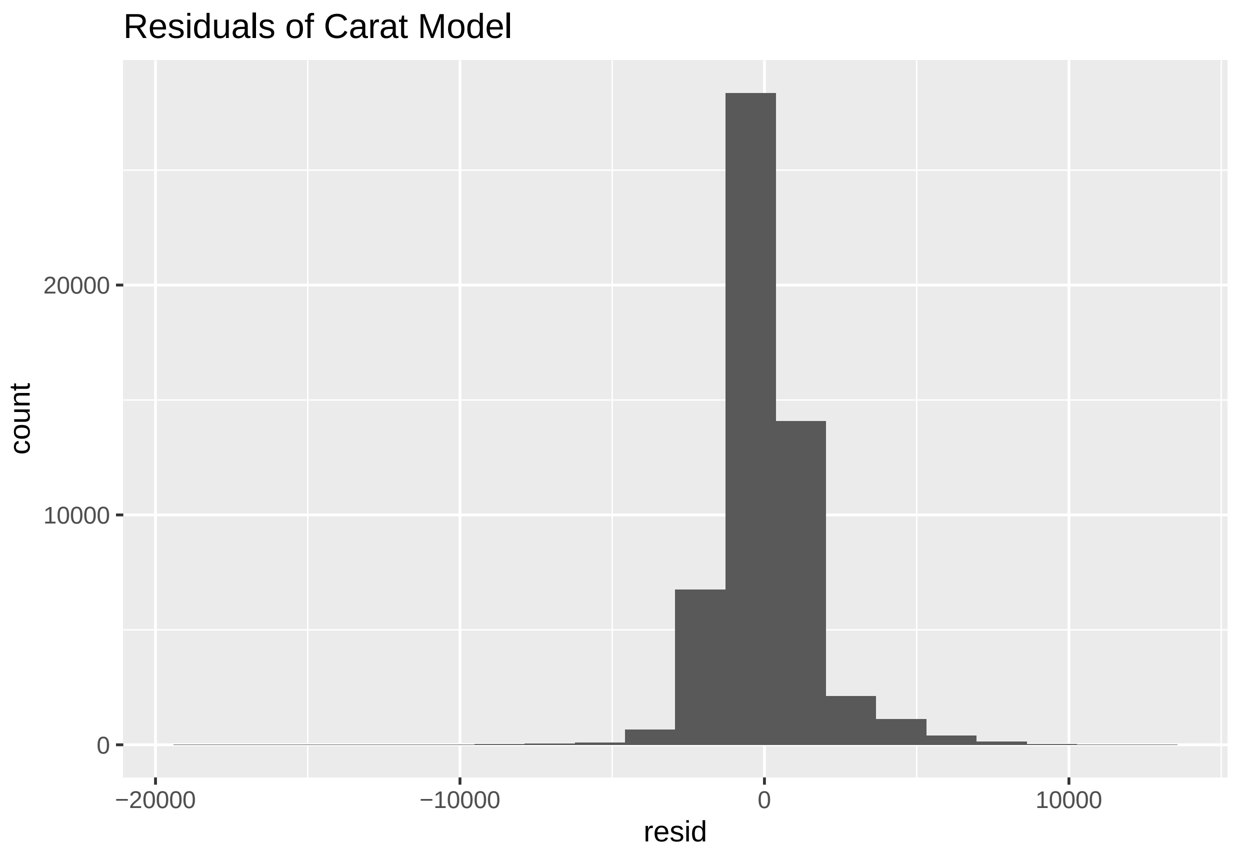
A basic linear regression model was used as it was the most appropriate model to use as our research question was looking at the correlation between three variables and which one affected the pricing of diamonds the most among all entries, and a linear regression model was the most convenient method to do so.

- Main model(s) chosen and why they are appropriate.

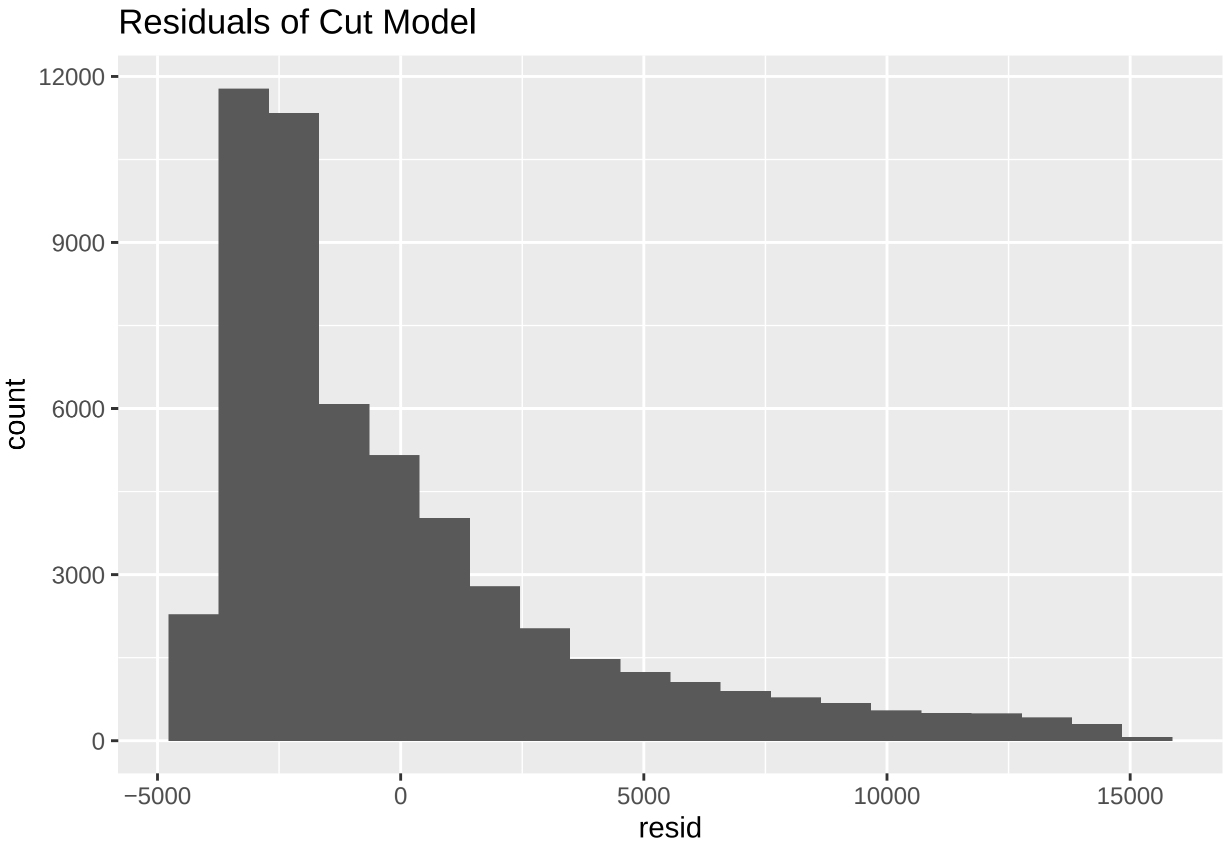
The main models we used were the box plots and the heat maps (2d bin plots). The box plots were used on the cut and clarity predictors due to them being categorical, and the heat maps were used with the carat predictor since it is a continuous predictor.

7. Model Implementation & Evaluation

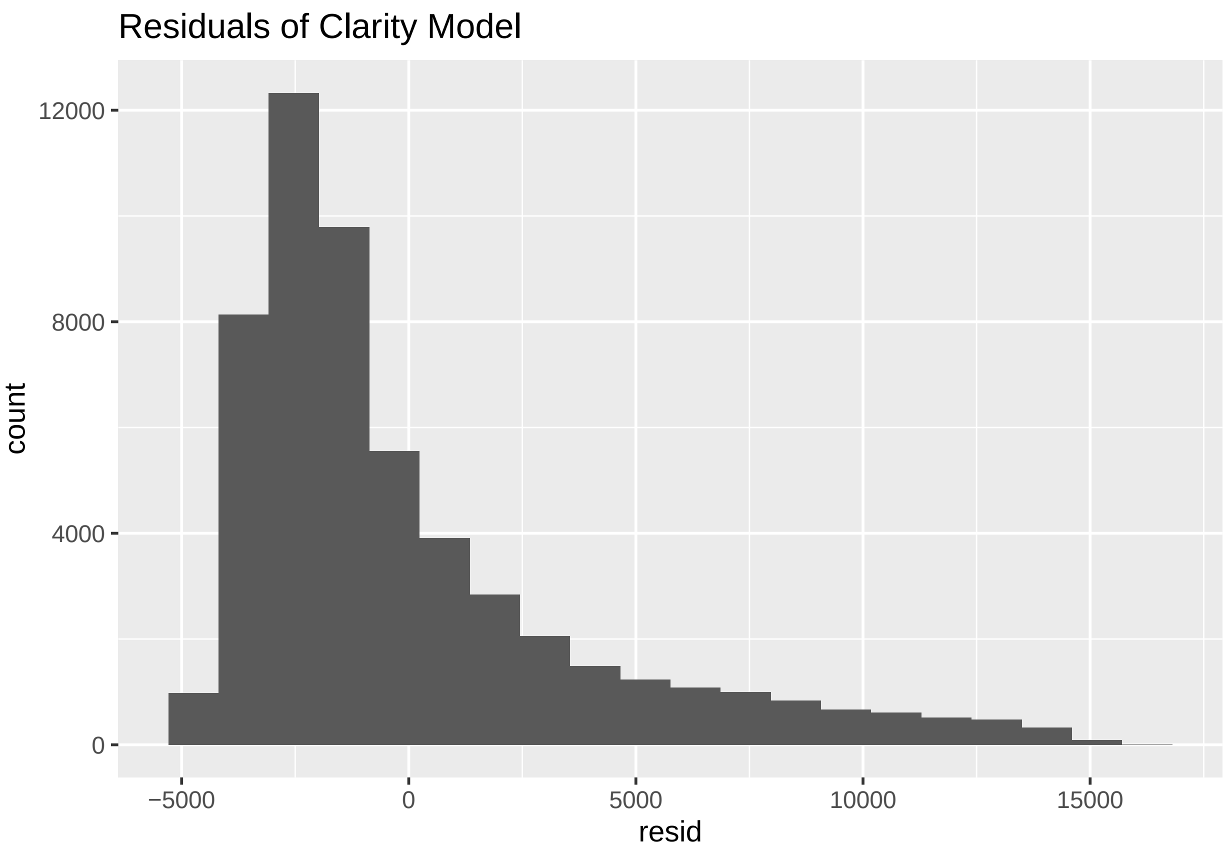
```
diamond_ma_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Carat Model")
```



```
diamond_mu_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Cut Model")
```



```
diamond_my_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Clarity Model")
```



Model for Carat is Skewed left in total, but there is a higher spike at 0, where as Cut and clarity are skewed right.

8. Conclusions & Recommendations

Carat seems to have a higher correlation with the Price of a diamond in comparison to Clarity and Cut because of the feature's squared residuals being very close to 1. Since this dataset has nearly 54 thousand entries with the price range approximately 18497 dollars, the sheer amount of data does hinder some comprehension. So taking some smaller sample of the data would help with comprehension. To quote an article by Alicia Briggs from Vrai, in relation to the reasoning behind carat's influence, "Carat weight is a significant factor in determining the total price of a diamond, but it's not the only one. The price of a diamond exponentially rises after 1 carat, with a 2 carat diamond often more than double the cost. Why? Because the higher the carat weight, the bigger the loose diamond it was cut from. This means the diamond is rarer and therefore more valuable."

9. Code Quality & Reproducibility

Briefly document how someone else can reproduce your results:

Set up.

```
## R version 4.5.1 (2025-06-13)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] modelr_0.1.11   broom_1.0.9     lubridate_1.9.4 forcats_1.0.0
##  [5] stringr_1.5.1   dplyr_1.1.4     purrr_1.1.0     tidyr_1.3.1
##  [9] tibble_3.3.0    ggplot2_4.0.0   tidyverse_2.0.0 readr_2.1.5
##
## loaded via a namespace (and not attached):
##  [1] bit_4.6.0          gtable_0.3.6       compiler_4.5.1     crayon_1.5.3
##  [5] tidyselect_1.2.1   parallel_4.5.1     scales_1.4.0       yaml_2.3.10
##  [9] fastmap_1.2.0      R6_2.6.1           labeling_0.4.3     generics_0.1.4
## [13] knitr_1.50         backports_1.5.0    pillar_1.11.0      RColorBrewer_1.1-3
## [17] tzdb_0.5.0         rlang_1.1.6        stringi_1.8.7      xfun_0.53
## [21] S7_0.2.0           bit64_4.6.0-1      viridisLite_0.4.2  timechange_0.3.0
## [25] cli_3.6.5          withr_3.0.2        magrittr_2.0.3     digest_0.6.37
## [29] grid_4.5.1         vroom_1.6.5        rstudioapi_0.17.1  hms_1.1.3
## [33] lifecycle_1.0.4    vctrs_0.6.5        evaluate_1.0.4     glue_1.8.0
## [37] farver_2.1.2       rmarkdown_2.29     tools_4.5.1        pkgconfig_2.0.3
## [41] htmltools_0.5.8.1
```

```
## Rows: 53940 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): cut, color, clarity
## dbl (7): carat, depth, table, x, y, z, price
##
## 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.
```

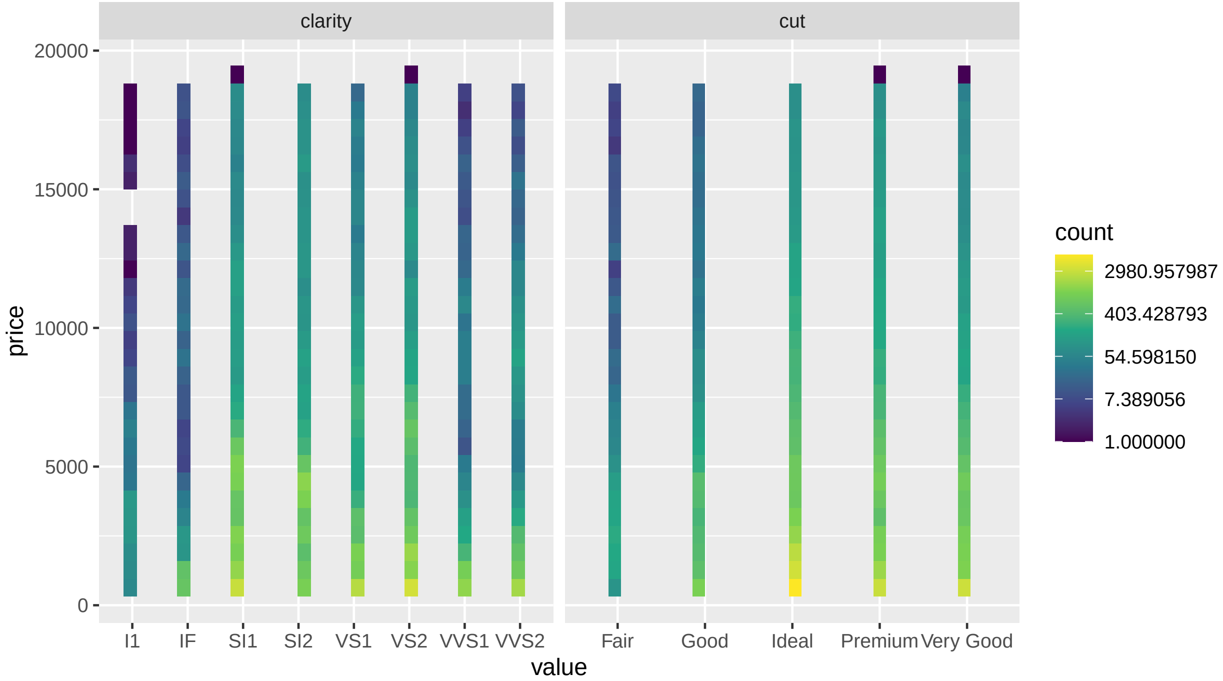
Cleaning data, only getting the needed values

```r
diamond_1 <- diamonds %>%
  select(carat, cut, clarity, price)
```

Clarity and cut vs prices point plots

```
diamond_1 %>%
  pivot_longer(cols = cut:clarity,
               names_to = "measurement",
               values_to = "value") %>%
  ggplot() +
    geom_bin2d(aes(x = value, y = price)) +
    scale_fill_viridis_c(trans = "log") +
    facet_wrap(~ measurement, scales = "free_x")
```

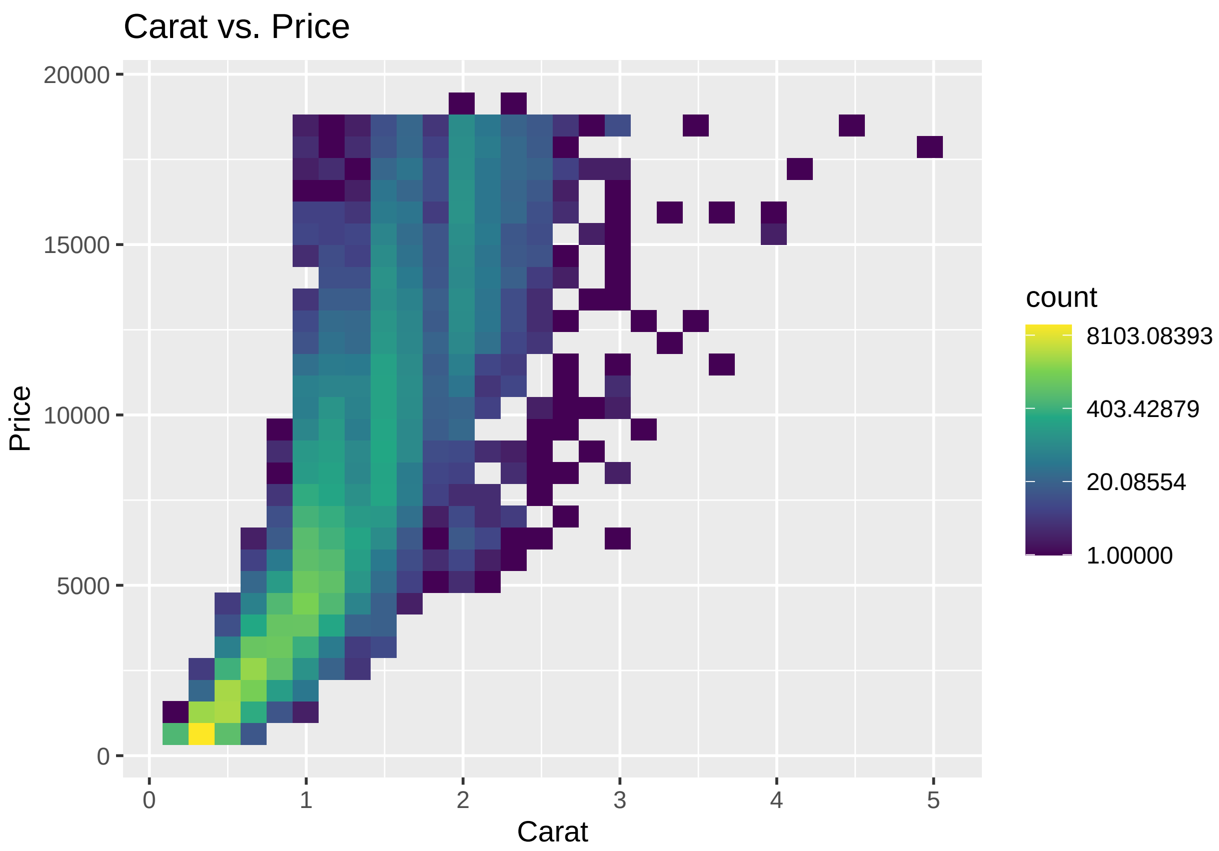
```
## `stat_bin2d()` using `bins = 30`. Pick better value `binwidth`.
```



Carat vs price heat plot

```
diamond_1 %>%
  ggplot() +
  geom_bin2d(aes(x = carat, y = price)) +
  labs(
    title = "Carat vs. Price",
    x = "Carat",
    y = "Price"
  ) +
scale_fill_viridis_c(trans = "log")
```

```
## `stat_bin2d()` using `bins = 30`. Pick better value `binwidth`.
```



Linear regression model for carat

```
Diamond_model_a <- lm(price ~ carat, data = diamond_1)
```

```
Diamond_model_a %>%
  tidy()
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)   -2256.      13.1     -173.       0
## 2 carat          7756.      14.1      551.       0
```

```
Diamond_model_a %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df   logLik     AIC     BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>    <dbl>   <dbl>   <dbl>
## 1     0.849         0.849 1549.   304051.       0     1 -472730. 945467. 945493.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Linear regression model for cut

```
Diamond_model_u <- lm(price ~ cut, data = diamond_1)
```

```
Diamond_model_u %>%
  tidy()
```

```
## # A tibble: 5 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     4359.      98.8     44.1  0
## 2 cutGood         -430.     114.      -3.78 1.60e- 4
## 3 cutIdeal        -901.     102.      -8.80 1.41e-18
## 4 cutPremium       225.     104.       2.16 3.08e- 2
## 5 cutVery Good    -377.     105.      -3.58 3.38e- 4
```

```
Diamond_model_u %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic   p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1    0.0129        0.0128 3964.      176. 8.43e-150     4 -523426. 1.05e6 1.05e6
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Linear regression model for clarity

```
Diamond_model_y <- lm(price ~ clarity, data = diamond_1)
```

```
Diamond_model_y %>%
  tidy()
```

```
## # A tibble: 8 x 5
##   term         estimate std.error statistic   p.value
##   <chr>           <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)  3924.         145.  27.1     3.51e-161
## 2 clarityIF   -1059.         172.  -6.16    7.21e- 10
## 3 claritySI1     71.8        149.   0.483   6.29e-  1
## 4 claritySI2   1139.         150.   7.58    3.55e- 14
## 5 clarityVS1    -84.7        151.  -0.561   5.75e-  1
## 6 clarityVS2      0.821      149.   0.00551 9.96e-  1
## 7 clarityVVS1 -1401.         159.  -8.84    1.01e- 18
## 8 clarityVVS2  -640.         155.  -4.14    3.51e-  5
```

```
Diamond_model_y %>%
  glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic   p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1    0.0272        0.0270 3935.      215. 1.93e-316     7 -523033. 1.05e6 1.05e6
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Graphing Carat vs price in relation to its Model

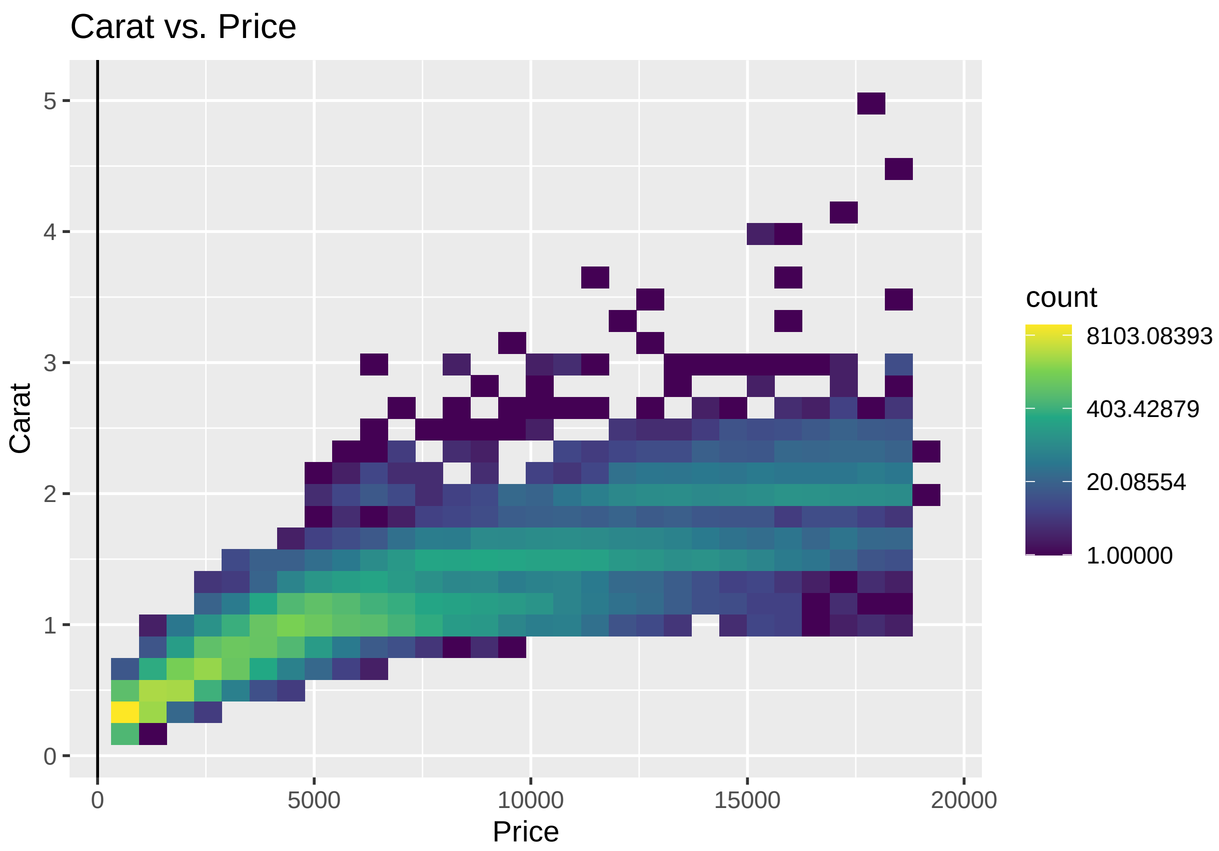
```
diamond_ma_df <- diamonds %>%
  add_predictions(Diamond_model_a) %>%
  add_residuals(Diamond_model_a)
```

```
diamond_ma_df
```

```
## # A tibble: 53,940 x 12
##    carat cut     color clarity depth table     x     y     z price    pred resid
##    <dbl> <chr>   <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
##  1  0.23 Ideal   E     SI2      61.5    55  3.95  3.98  2.43   326 -472.    798.
##  2  0.21 Premium E     SI1      59.8    61  3.89  3.84  2.31   326 -628.    954.
##  3  0.23 Good    E     VS1      56.9    65  4.05  4.07  2.31   327 -472.    799.
##  4  0.29 Premium I     VS2      62.4    58  4.2   4.23  2.63   334   -7.00  341.
##  5  0.31 Good    J     SI2      63.3    58  4.34  4.35  2.75   335  148.    187.
##  6  0.24 Very G~ J     VVS2     62.8    57  3.94  3.96  2.48   336 -395.    731.
##  7  0.24 Very G~ I     VVS1     62.3    57  3.95  3.98  2.47   336 -395.    731.
##  8  0.26 Very G~ H     SI1      61.9    55  4.07  4.11  2.53   337 -240.    577.
##  9  0.22 Fair    E     VS2      65.1    61  3.87  3.78  2.49   337 -550.    887.
## 10  0.23 Very G~ H     VS1      59.4    61  4     4.05  2.39   338 -472.    810.
## # i 53,930 more rows
```

```
ggplot(diamond_ma_df) +
  geom_bin2d(aes(x = price, y = carat)) +
  labs(
    title = "Carat vs. Price",
    x = "Price",
    y = "Carat"
  ) +
scale_fill_viridis_c(trans = "log") +
  geom_abline(slope = Diamond_model_a$coefficients[2], intercept = Diamond_model_a$coefficients[1])
```

```
## `stat_bin2d()` using `bins = 30`. Pick better value `binwidth`.
```



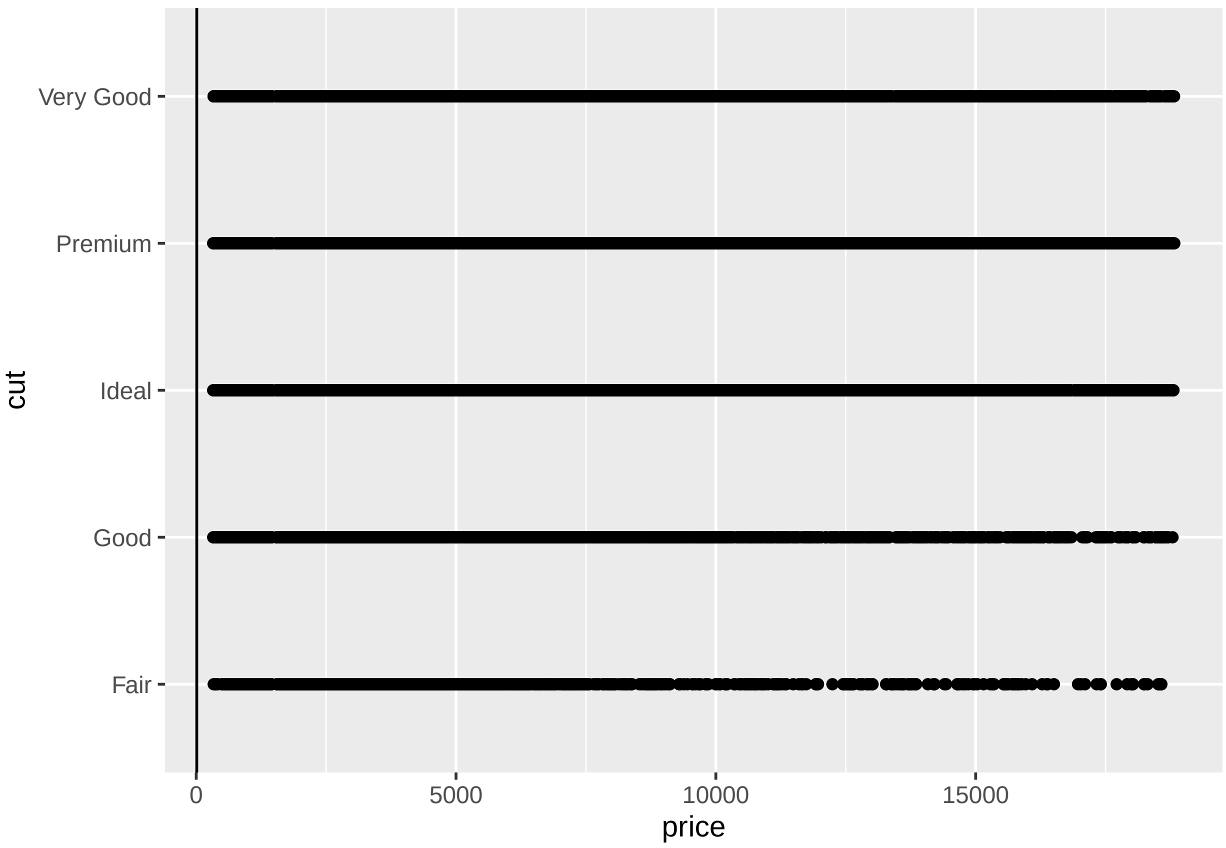
Graphing cut vs price in relation to its Model

```
diamond_mu_df <- diamonds %>%
  add_predictions(Diamond_model_u) %>%
  add_residuals(Diamond_model_u)
```

```
diamond_mu_df
```

```
## # A tibble: 53,940 x 12
##    carat cut       color clarity depth table     x     y     z price  pred  resid
##    <dbl> <chr>     <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55  3.95  3.98  2.43   326 3458. -3132.
##  2  0.21 Premium   E     SI1      59.8    61  3.89  3.84  2.31   326 4584. -4258.
##  3  0.23 Good      E     VS1      56.9    65  4.05  4.07  2.31   327 3929. -3602.
##  4  0.29 Premium   I     VS2      62.4    58  4.2   4.23  2.63   334 4584. -4250.
##  5  0.31 Good      J     SI2      63.3    58  4.34  4.35  2.75   335 3929. -3594.
##  6  0.24 Very Go~ J     VVS2     62.8    57  3.94  3.96  2.48   336 3982. -3646.
##  7  0.24 Very Go~ I     VVS1     62.3    57  3.95  3.98  2.47   336 3982. -3646.
##  8  0.26 Very Go~ H     SI1      61.9    55  4.07  4.11  2.53   337 3982. -3645.
##  9  0.22 Fair      E     VS2      65.1    61  3.87  3.78  2.49   337 4359. -4022.
## 10  0.23 Very Go~ H     VS1      59.4    61  4     4.05  2.39   338 3982. -3644.
## # i 53,930 more rows
```

```
ggplot(diamond_mu_df) +
  geom_point(mapping = aes(x = price, y = cut)) +
  geom_abline(slope = Diamond_model_u$coefficients[2], intercept = Diamond_model_u$coefficients[1])
```



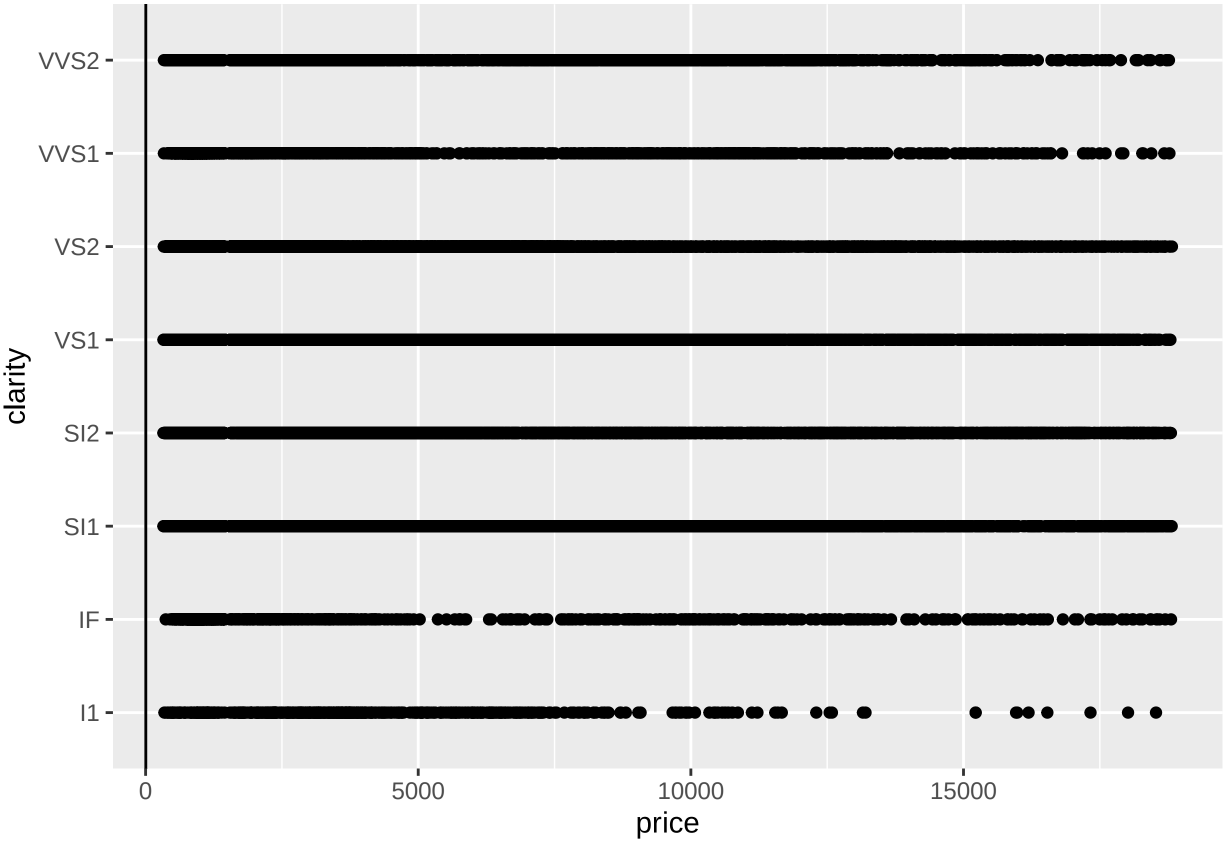
Graphing clarity vs price in relation to its Model

```
diamond_my_df <- diamonds %>%
  add_predictions(Diamond_model_y) %>%
  add_residuals(Diamond_model_y)
```

```
diamond_my_df
```

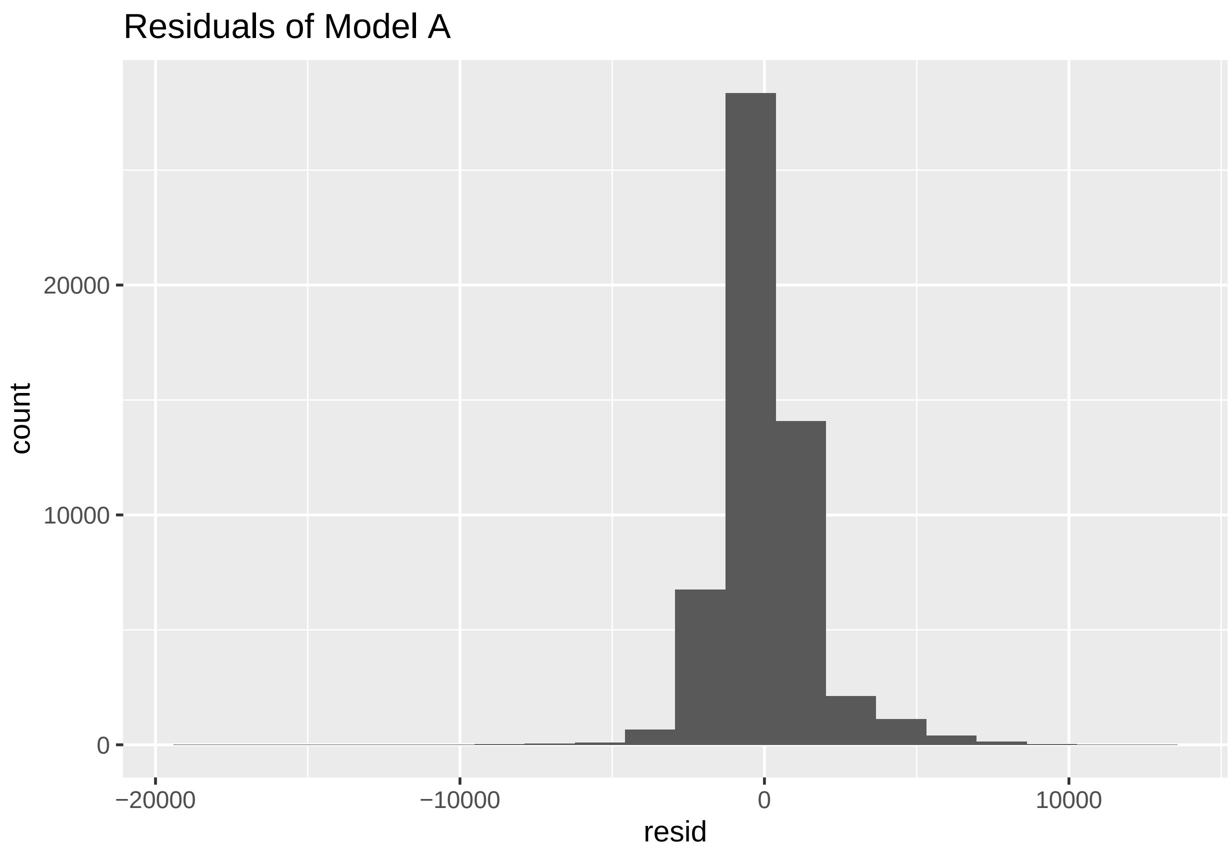
```
## # A tibble: 53,940 x 12
##    carat cut      color clarity depth table     x     y     z price  pred  resid
##    <dbl> <chr>    <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
##  1  0.23 Ideal    E     SI2      61.5    55  3.95  3.98  2.43   326 5063. -4737.
##  2  0.21 Premium  E     SI1      59.8    61  3.89  3.84  2.31   326 3996. -3670.
##  3  0.23 Good     E     VS1      56.9    65  4.05  4.07  2.31   327 3839. -3512.
##  4  0.29 Premium  I     VS2      62.4    58  4.2   4.23  2.63   334 3925. -3591.
##  5  0.31 Good     J     SI2      63.3    58  4.34  4.35  2.75   335 5063. -4728.
##  6  0.24 Very Go~ J     VVS2     62.8    57  3.94  3.96  2.48   336 3284. -2948.
##  7  0.24 Very Go~ I     VVS1     62.3    57  3.95  3.98  2.47   336 2523. -2187.
##  8  0.26 Very Go~ H     SI1      61.9    55  4.07  4.11  2.53   337 3996. -3659.
##  9  0.22 Fair     E     VS2      65.1    61  3.87  3.78  2.49   337 3925. -3588.
## 10  0.23 Very Go~ H     VS1      59.4    61  4     4.05  2.39   338 3839. -3501.
## # i 53,930 more rows
```

```
ggplot(diamond_my_df) +
  geom_point(mapping = aes(x = price, y = clarity)) +
  geom_abline(slope = Diamond_model_y$coefficients[2], intercept = Diamond_model_y$coefficients[1])
```

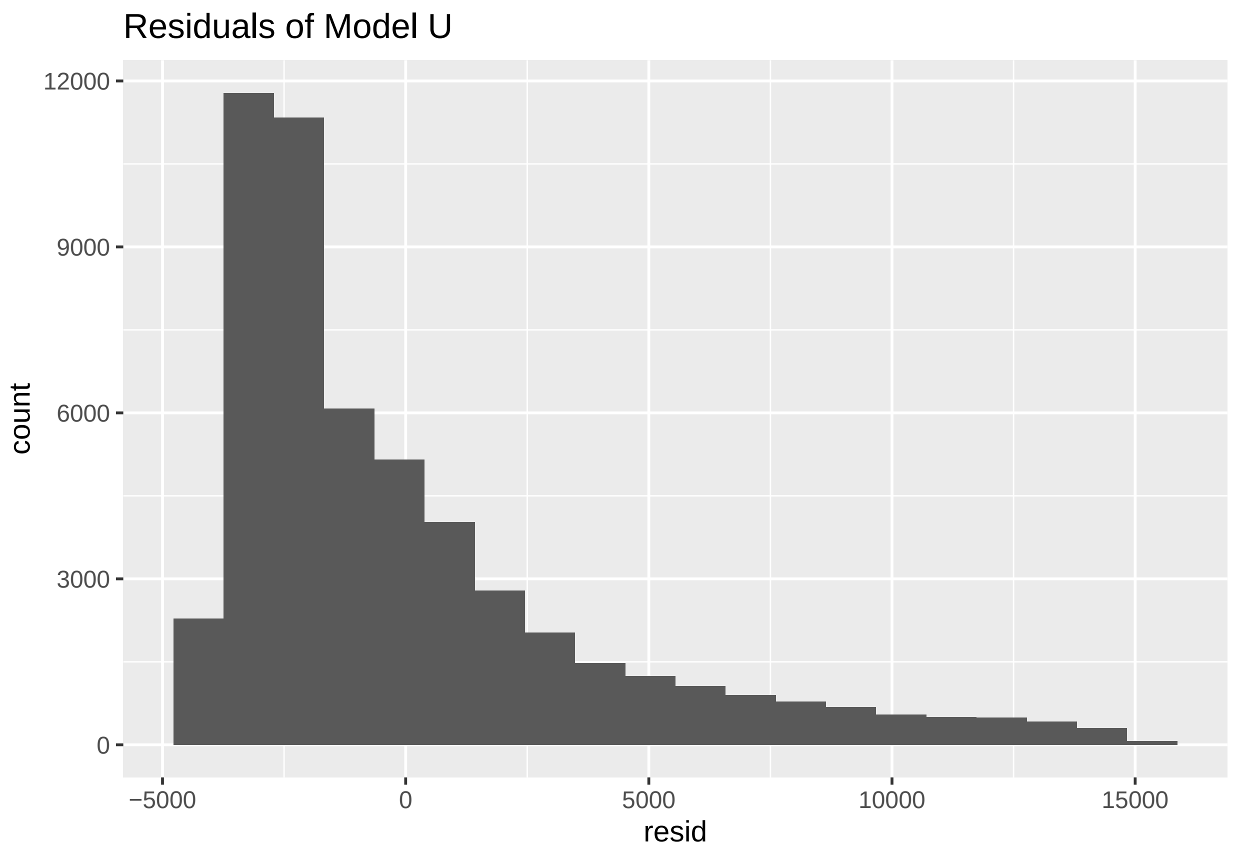


Residual histogram plots of all 3 models.

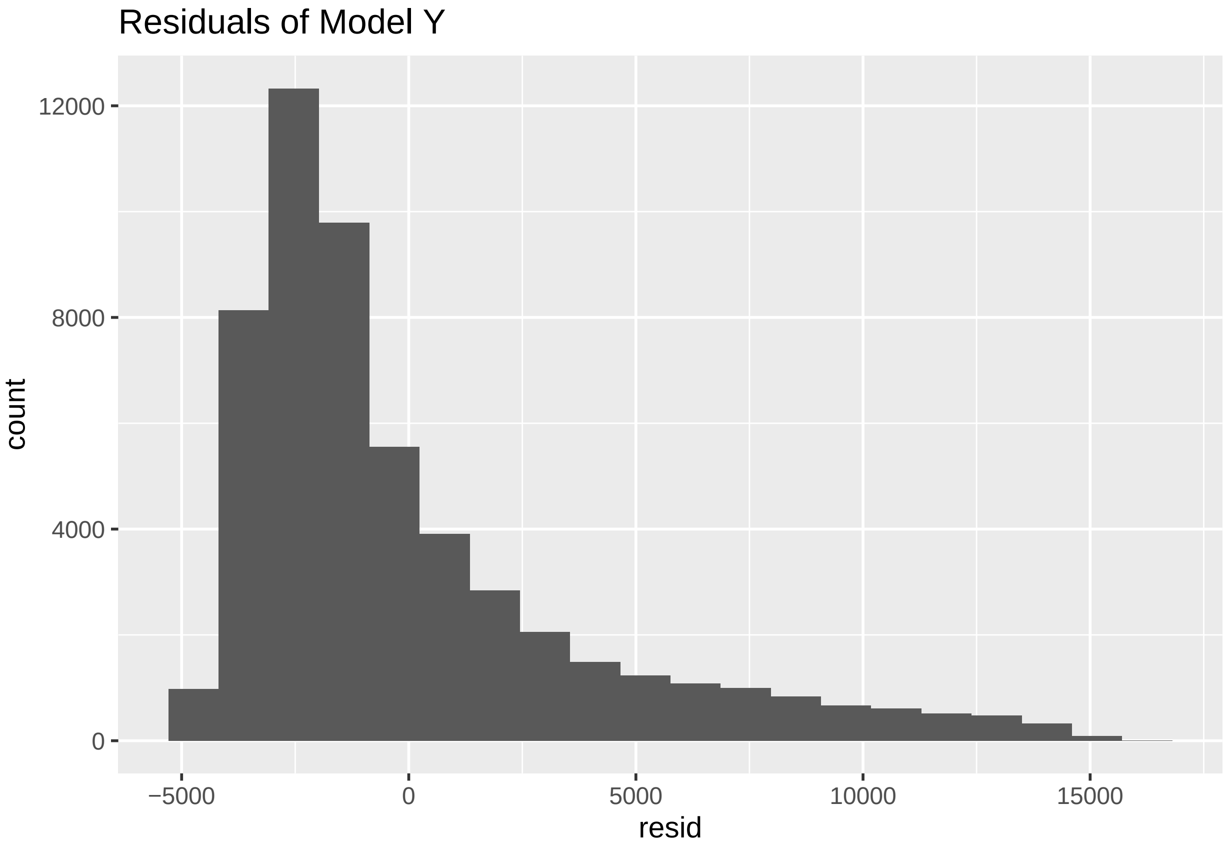
```
diamond_ma_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Model A")
```



```
diamond_mu_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Model U")
```



```
diamond_my_df %>%
  ggplot()+
  geom_histogram(mapping = aes(x = resid), bins = 20) +
  labs(title = "Residuals of Model Y")
```



10. References

References:

https://www.vrai.com/journal/post/diamond-carat-price article by Alicia Briggs | February 06, 2023

https://www.kaggle.com/datasets/ayeshaseherr/diamonds.