HW week 12

w203: Statistics for Data Science

w203 teaching team

```
library(tidyverse)
library(ggplot2)

library(sandwich)
library(stargazer)

library(lmtest)
```

```
d <- load_and_clean(input = 'videos.txt')
```

```
## Rows: 9618 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (3): video_id, uploader, category
## dbl (6): age, length, views, rate, ratings, comments
##
## 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.
```

Regression analysis of YouTube dataset

You want to explain how much the quality of a video affects the number of views it receives on social media. In a world where people can now buy followers and likes, would such an investment increase the number of views that their content receives? **This is a causal question.**

You will use a dataset created by Cheng, Dale and Liu at Simon Fraser University. It includes observations about 9618 videos shared on YouTube. Please see this link for details about how the data was collected.

You will use the following variables:

- `views`: the number of views by YouTube users.
- `average_rating`: This is the average of the ratings that the video received, it is a renamed feature from `rate` that is provided in the original dataset. (Notice that this is different from `cout_of_ratings` which is a count of the total number of ratings that a video has received.
- `length`: the duration of the video in seconds.

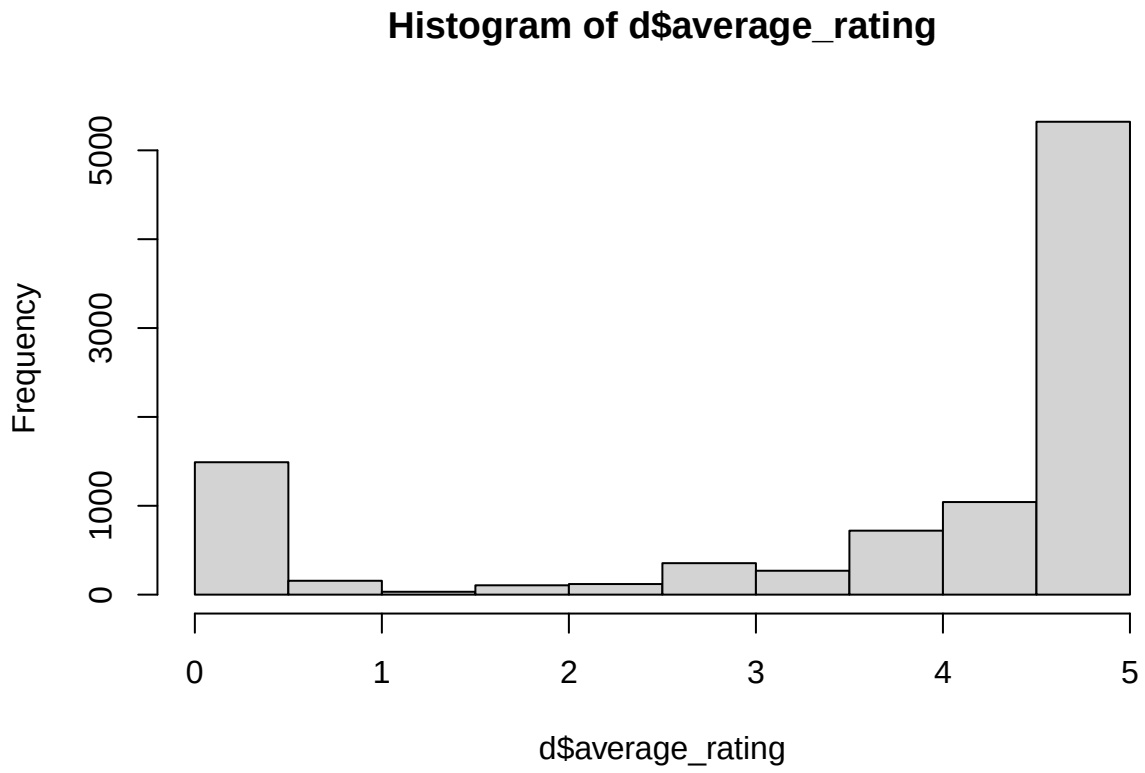
a. Perform a brief exploratory data analysis on the data to discover patterns, outliers, or wrong data entries and summarize your findings.

```
# view col names
names(d)
```

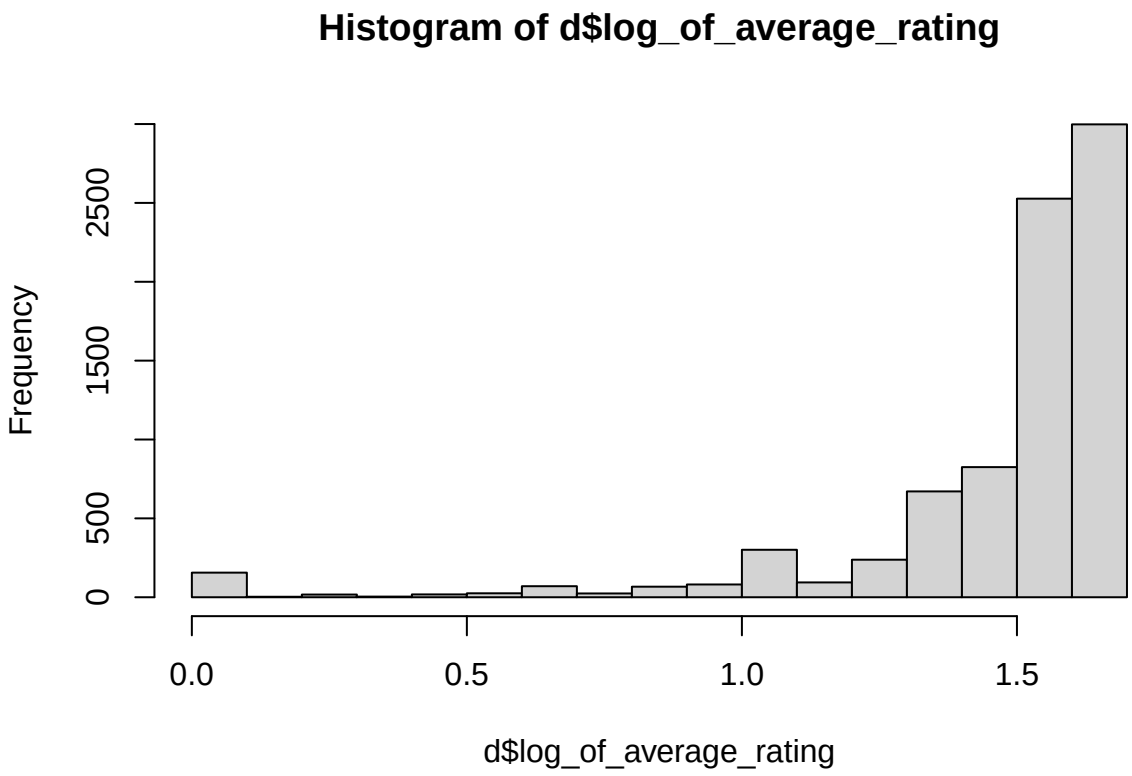
```
##  [1] "video_id"             "uploader"             "age"
##  [4] "category"             "length"               "views"
##  [7] "average_rating"       "count_of_ratings"     "comments"
```

```
## [10] "log_of_average_rating"
```

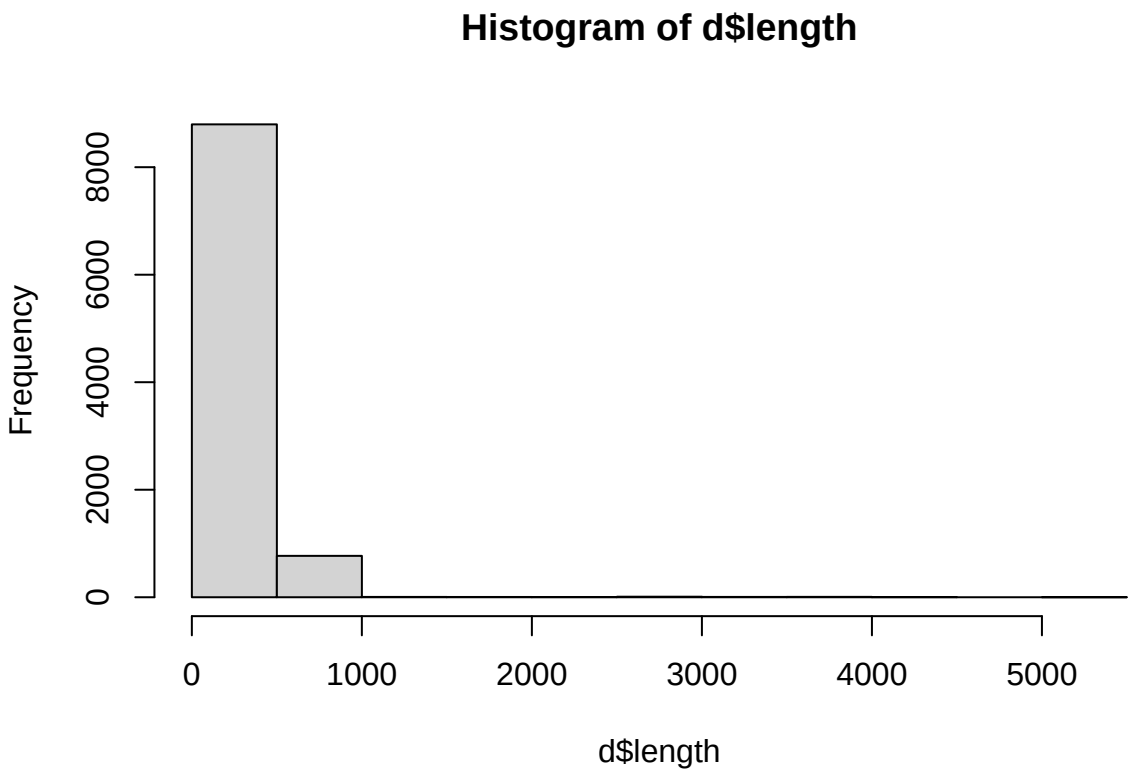
```
# view data
# View(d)

# basic pre EDA histograms of ratings
hist(d$average_rating)
```



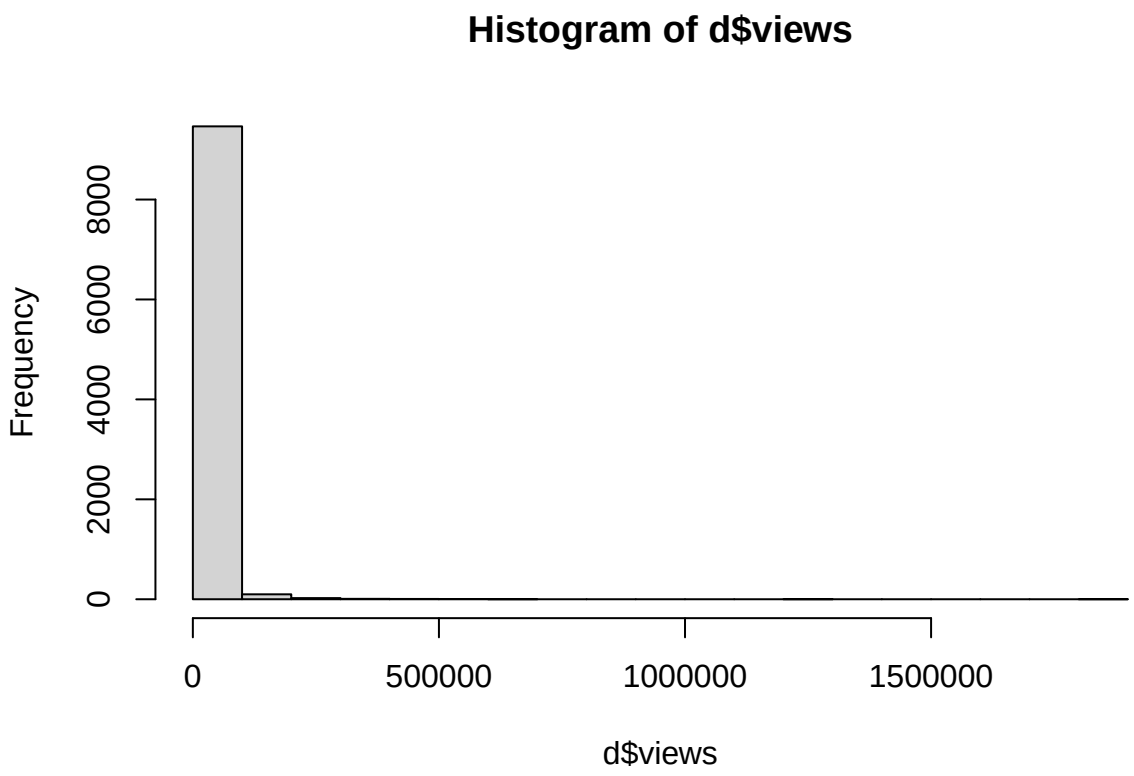

```
hist(d$log_of_average_rating)
```



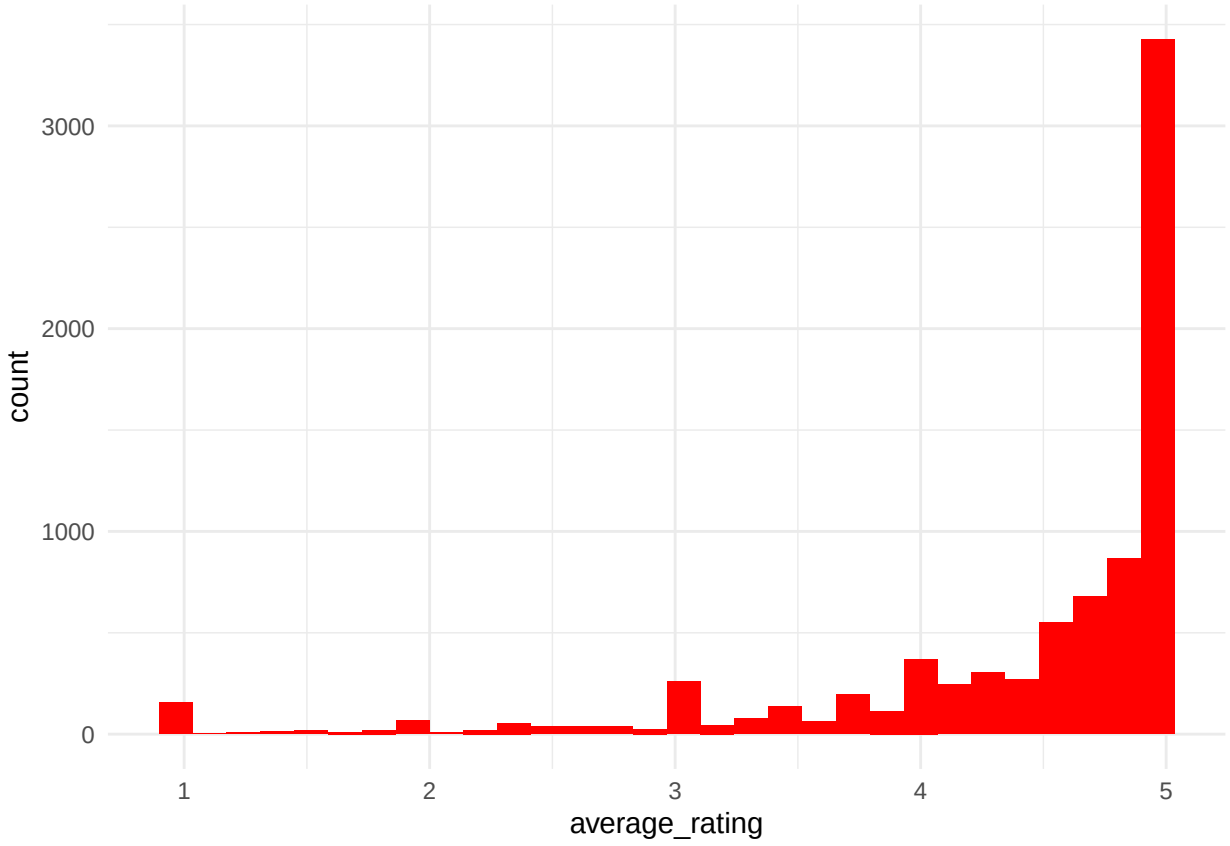

```
hist(d$length)
```


Histogram of d$length
Frequency
0
2000
4000
6000
8000
0
1000
2000
3000
4000
5000
d$length

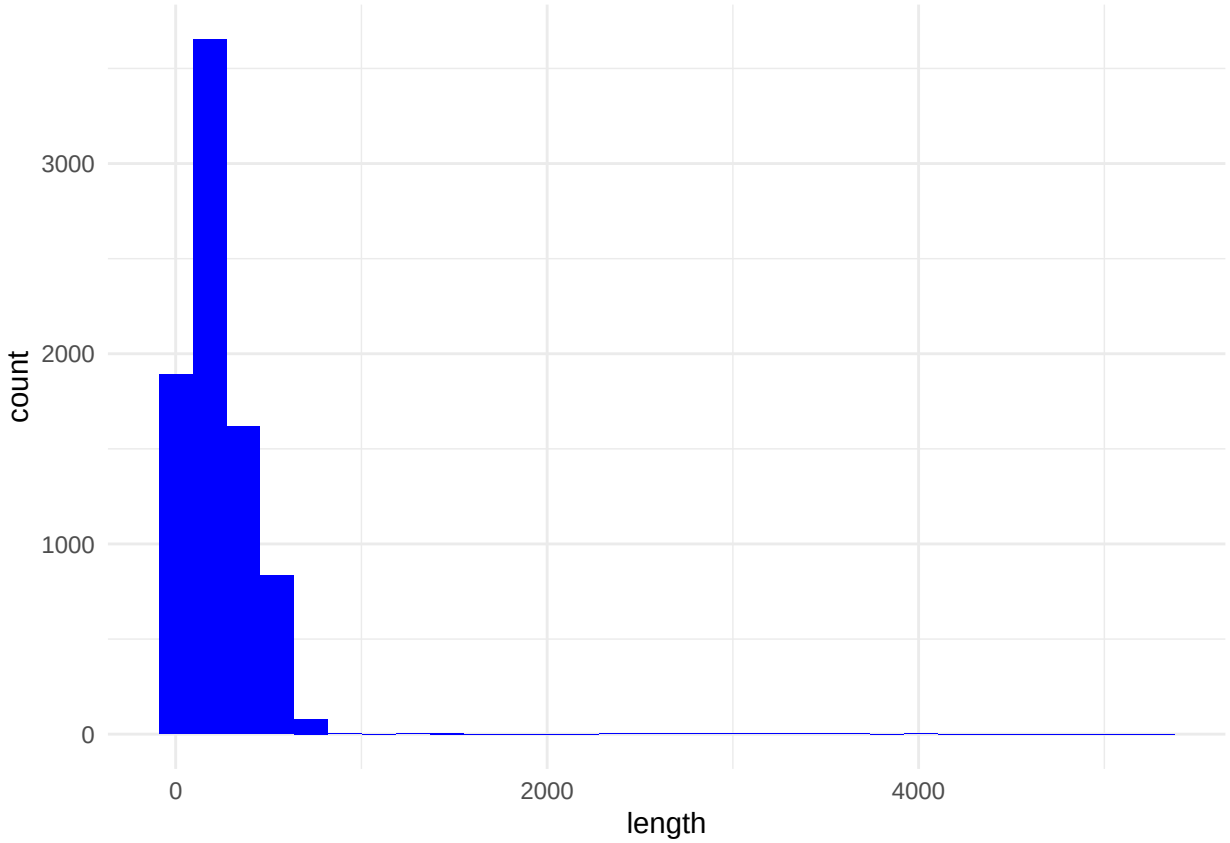
```
hist(d$views)
```



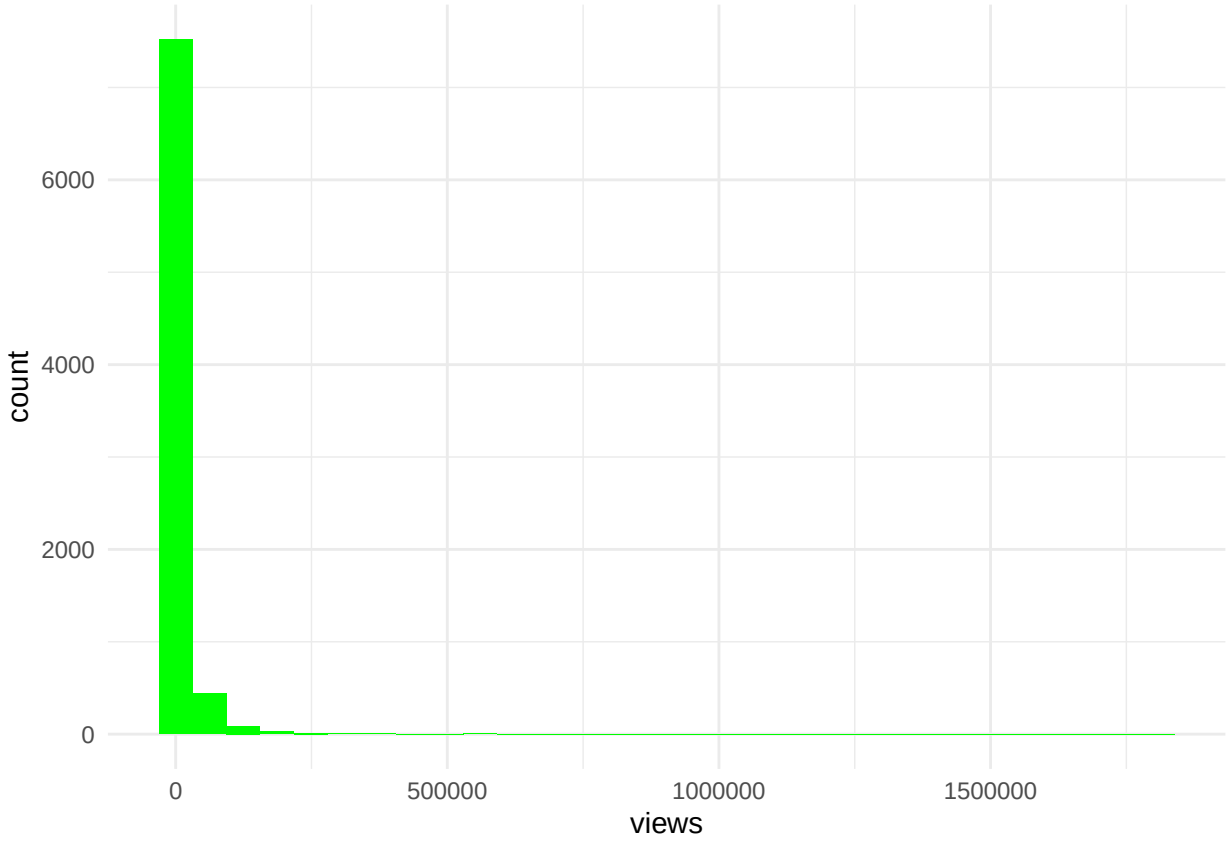

```
# remove all invalid data
# remove 0 average ratings
d <- d[!d$average_rating == 0,]

# remove 0 length
d <- d[!is.na(d$length),]

# remove 0 views
d <- d[!is.na(d$views),]

# View(d)

# histogram plots
ggplot(d) +
  aes(x = average_rating) +
  geom_histogram(bins = 30L, fill = "red") +
  theme_minimal()
```



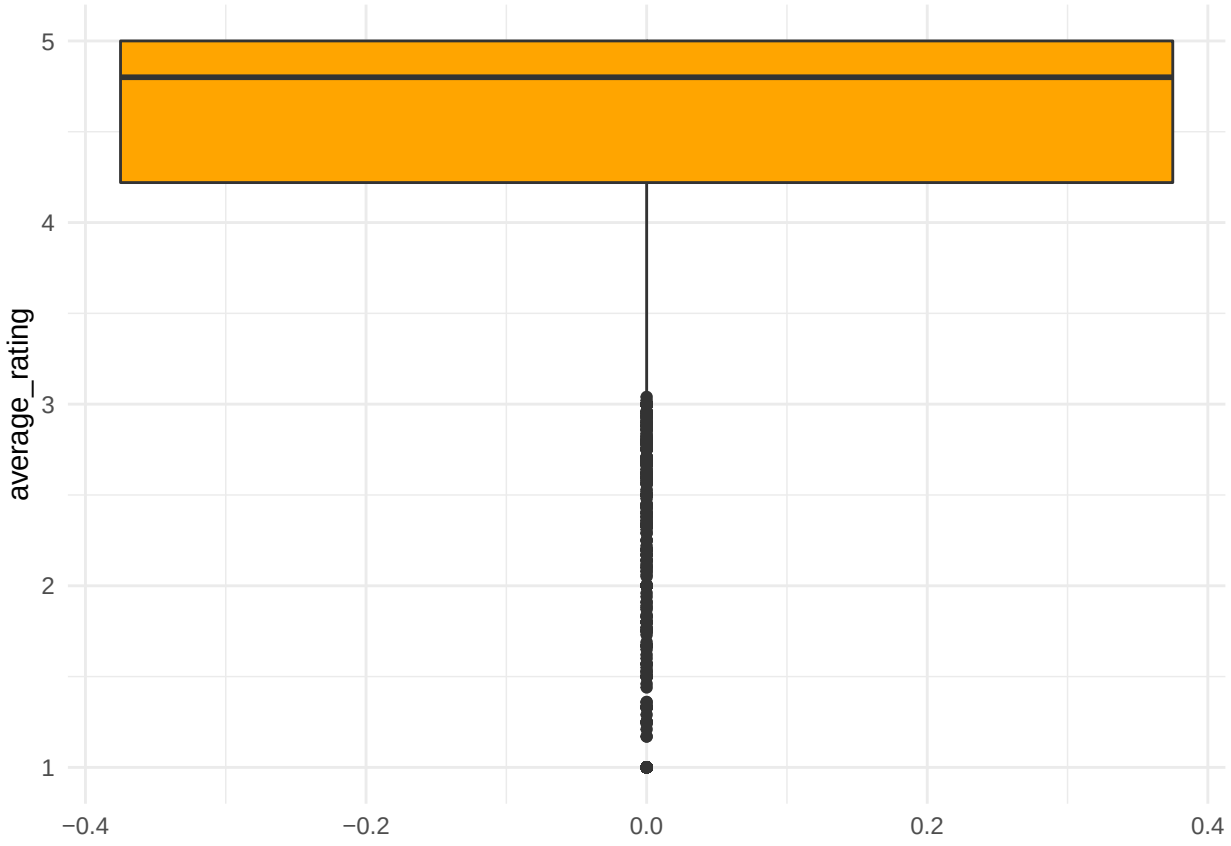

```
ggplot(d) +
  aes(x = length) +
  geom_histogram(bins = 30L, fill = "blue") +
  theme_minimal()
```



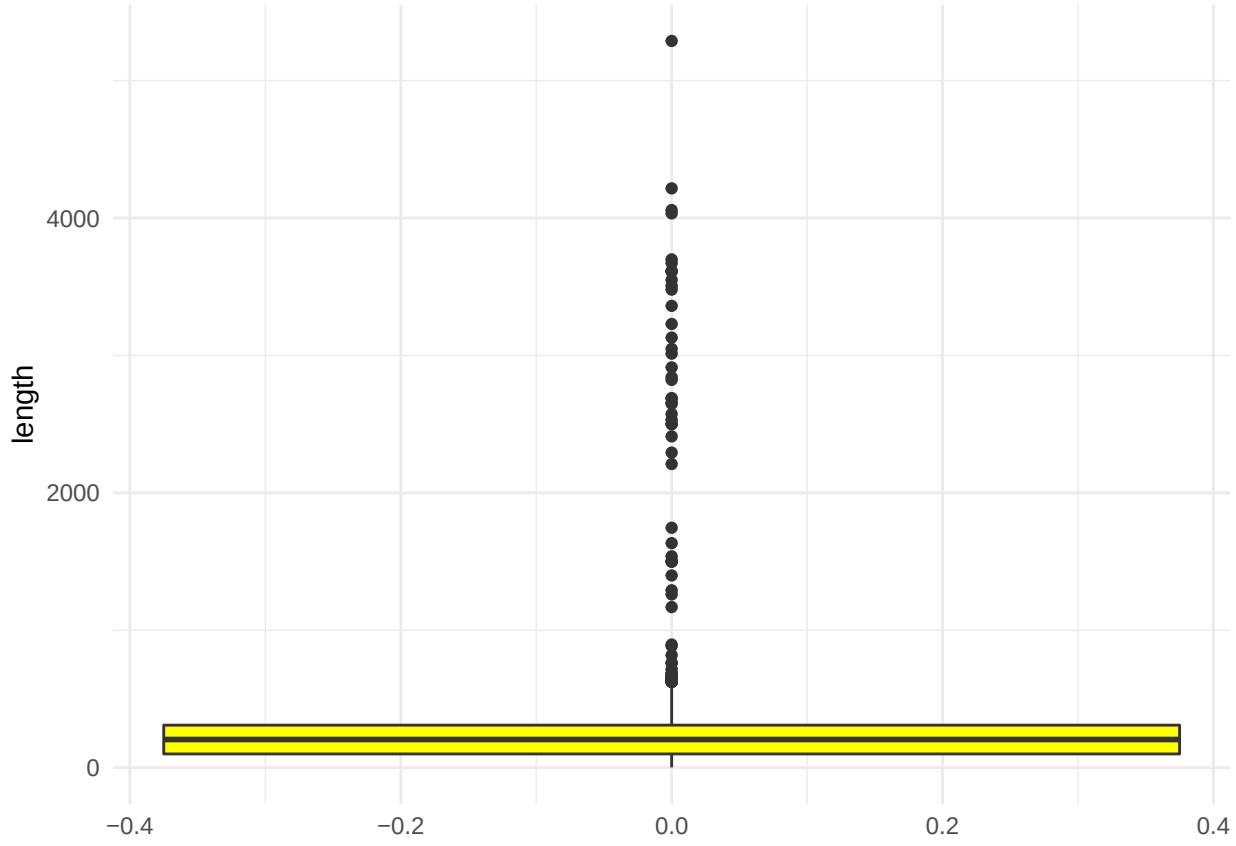

```
ggplot(d) +
  aes(x = views) +
  geom_histogram(bins = 30L, fill = "green") +
  theme_minimal()
```



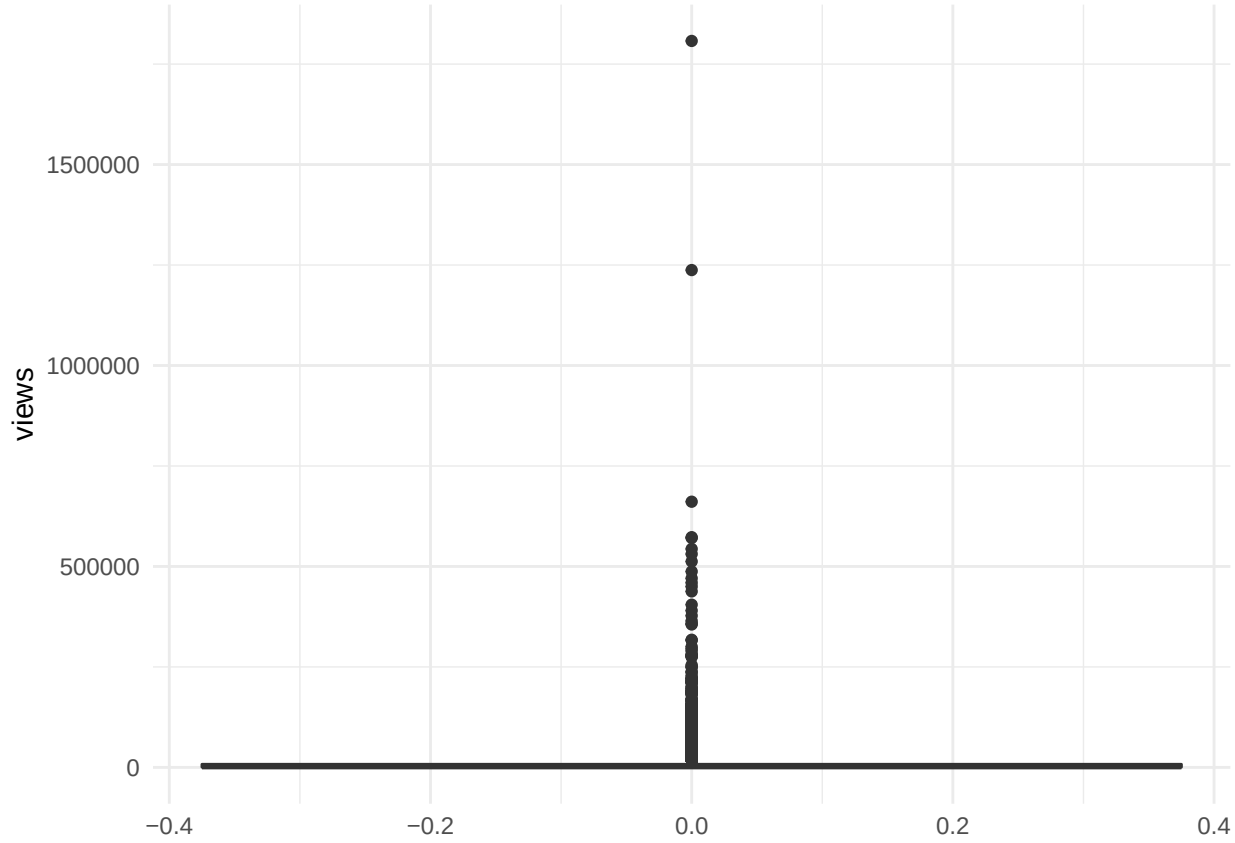

```
# boxplots
ggplot(d) +
  aes(y = average_rating) +
  geom_boxplot(fill = "orange") +
  theme_minimal()
```




```
ggplot(d) +
  aes(y = length) +
  geom_boxplot(fill = "yellow") +
  theme_minimal()
```



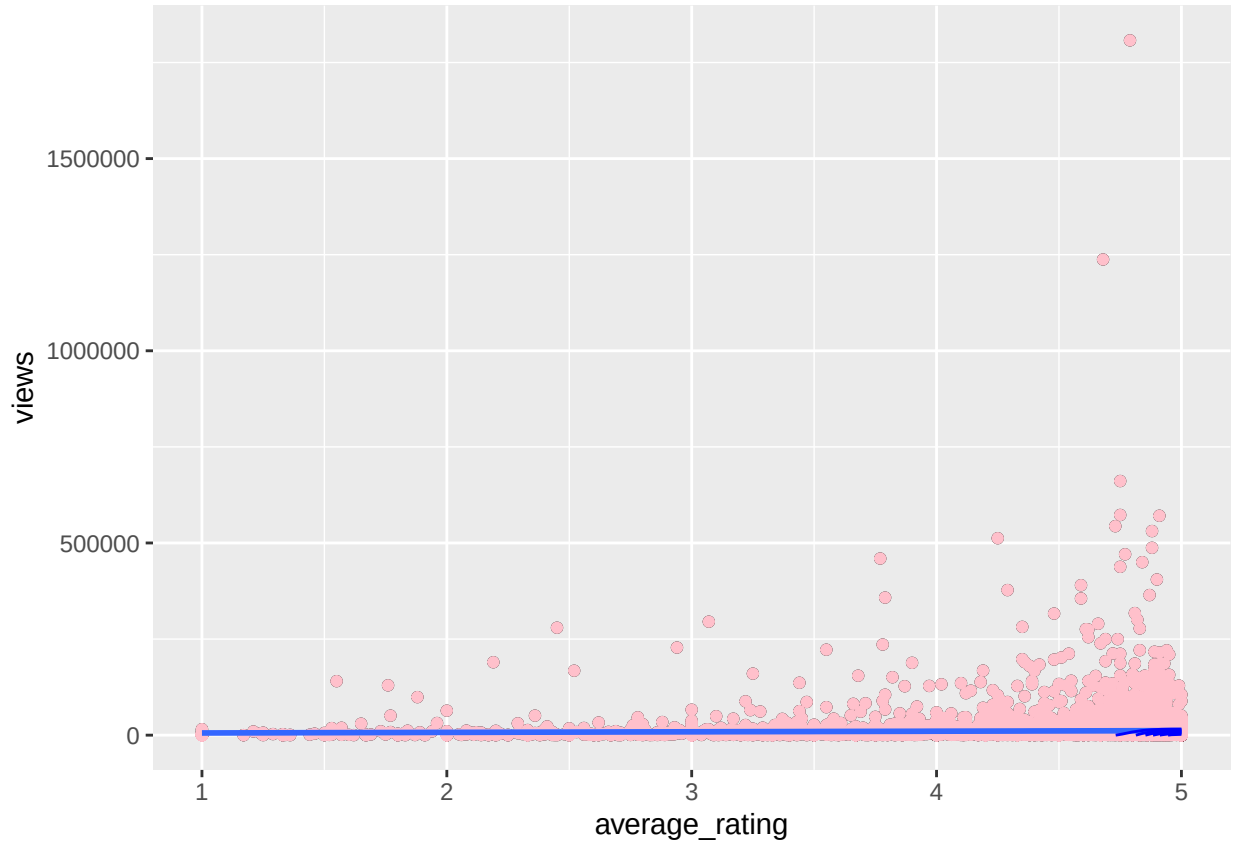

```
ggplot(d) +
  aes(y = views) +
  geom_boxplot(fill = "cyan") +
  theme_minimal()
```




```
#
# # potential outliers
# # extract the values of the potential outliers based on the IQR criterion using the boxplot.stats()$o
# out <- boxplot.stats(d$average_rating)$out
# # Use the which() function to extract the row number corresponding to these outliers
# out_ind <- which(d$average_rating %in% c(out))
# out_ind
# # go back to the specific rows in the dataset to verify them, or print all variables for these outlie
# d[out_ind, ]
#
# out <- boxplot.stats(d$length)$out
# out_ind <- which(d$length %in% c(out))
# out_ind
#
# d[out_ind, ]
#
# boxplot.stats(d$views)$out
# out <- boxplot.stats(d$views)$out
# out_ind <- which(d$views %in% c(out))
# out_ind
#
# d[out_ind, ]

# scatter plots
ggplot(d, aes(x = average_rating, y = views) ) +
  geom_point() +
```

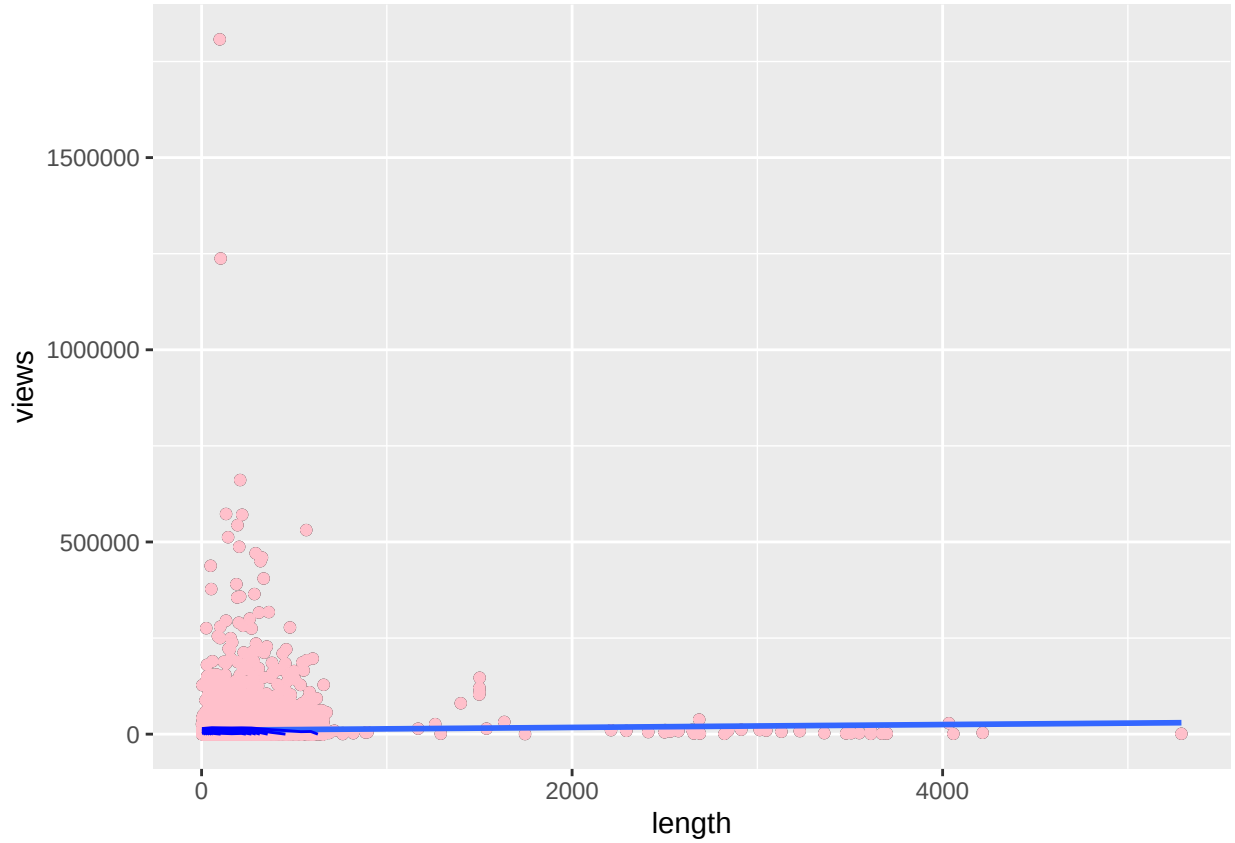
```
  geom_point(col='pink') +
  geom_smooth(method = lm, se = FALSE) +
  geom_density2d(col = 'blue')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



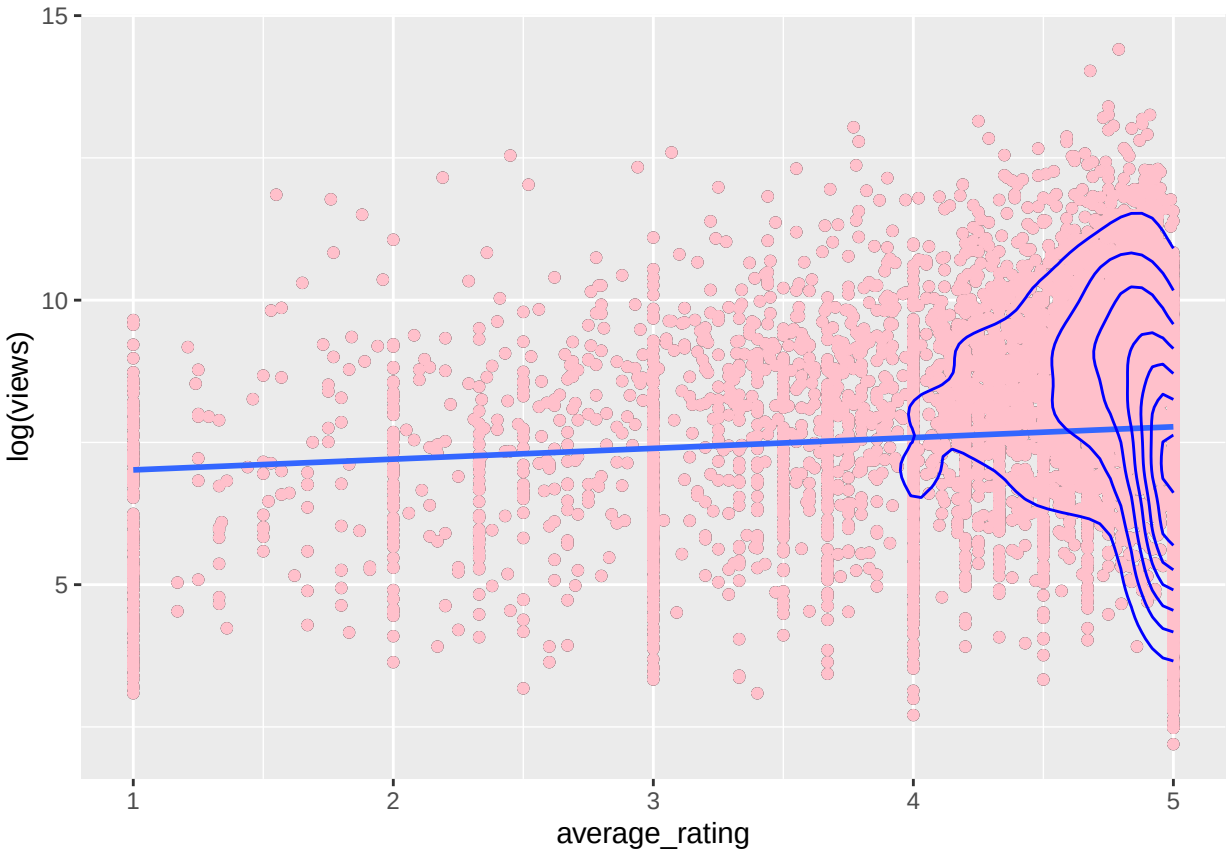

```
ggplot(d, aes(x = length, y = views) ) +
  geom_point() +
  geom_point(col='pink') +
  geom_smooth(method = lm, se = FALSE) +
  geom_density2d(col = 'blue')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



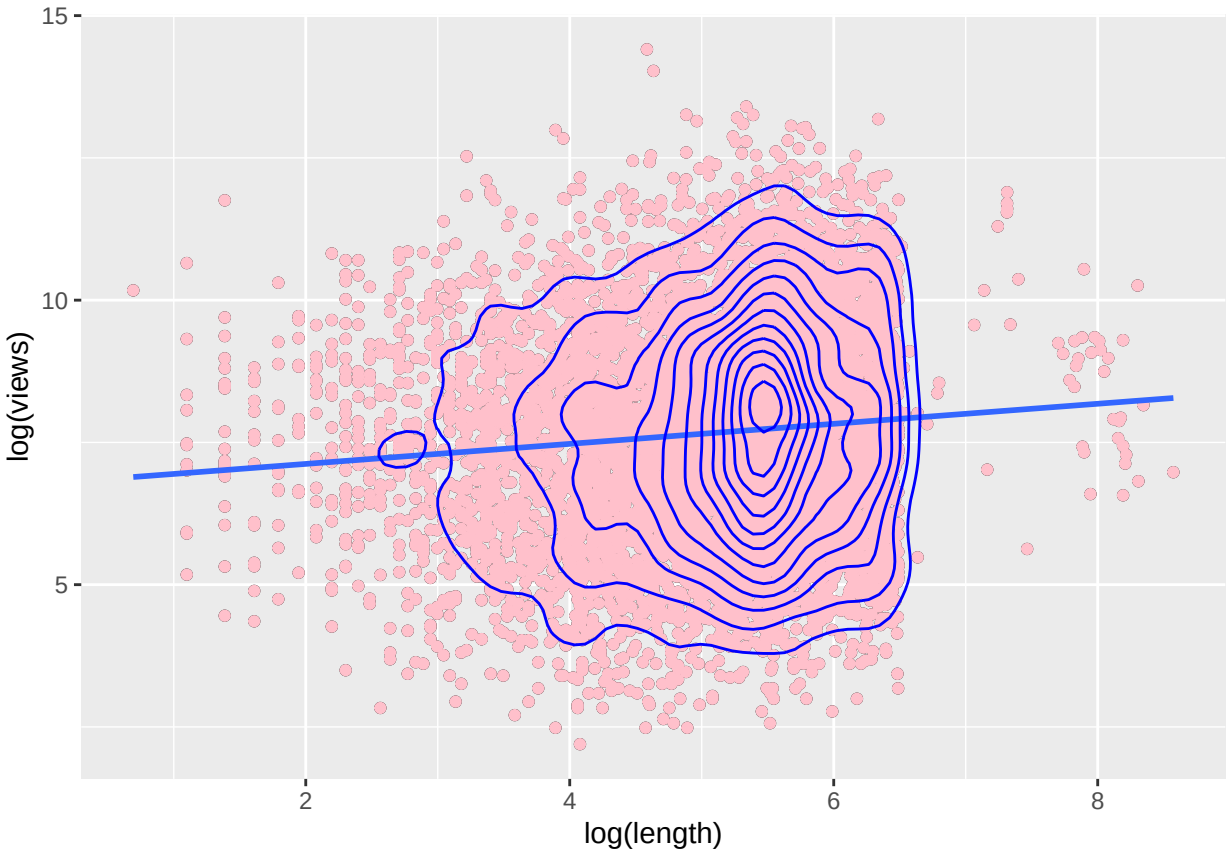

```
ggplot(d, aes(x = average_rating, y = log(views) ) ) +
  geom_point() +
  geom_point(col='pink') +
  geom_smooth(method = lm, se = FALSE) +
  geom_density2d(col = 'blue')
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(d, aes(x = log(length), y = log(views) ) ) +
  geom_point() +
  geom_point(col='pink') +
  geom_smooth(method = lm, se = FALSE) +
  geom_density2d(col = 'blue')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



'What did you learn from your EDA? Cut this quoted text and describe your analysis in the quote block.' Had to remove NA values and zereo values from the data. There are outliers based on the histograms and boxplots created. The average rating has some very low outliers, the length data some high outliers, and the views and higly concentrated on the low end values.

b. Based on your EDA, select an appropriate variable transformation (if any) to apply to each of your three variables. You will fit a model of the type,

$$f(\text{views}) = \beta_0 + \beta_1 g(\text{rate}) + \beta_3 h(\text{length})$$

Where f, g and h are sensible transformations, which might include making *no* transformation.

```
model <-lm(log(views) ~ average_rating + log(length), data = d)
model_two <-lm(views ~ average_rating + length, data = d)

summary(model)
```

```
##
## Call:
## lm(formula = log(views) ~ average_rating + log(length), data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4061 -1.3160  0.0114  1.2993  6.7609
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)     6.21247    0.13644  45.531  < 2e-16 ***
## average_rating  0.15588    0.02403   6.487 9.27e-11 ***
## log(length)     0.14995    0.02083   7.197 6.70e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.83 on 8116 degrees of freedom
## Multiple R-squared:  0.01414,    Adjusted R-squared:  0.01389
## F-statistic: 58.19 on 2 and 8116 DF,  p-value: < 2.2e-16
```

```
summary(model_two)
```

```
##
## Call:
## lm(formula = views ~ average_rating + length, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -26754  -10492   -8191   -2987 1796636
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3837.604   2343.055   1.638  0.10149
## average_rating 1431.016    520.816   2.748  0.00602 **
## length            3.183      1.797   1.771  0.07659 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40170 on 8116 degrees of freedom
## Multiple R-squared:  0.001465,   Adjusted R-squared:  0.001219
## F-statistic: 5.955 on 2 and 8116 DF,  p-value: 0.002603
```

```
stargazer(
 model,
 type = 'text',
 se = list(get_robust_se(model))
 )
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                              log(views)
## ------------------------------------------------
## average_rating                0.156***
##                                (0.024)
##
## log(length)                   0.150***
##                                (0.020)
##
## Constant                      6.212***
##                                (0.136)
##
## ------------------------------------------------
## Observations                    8,119
```

```
## R2                             0.014
## Adjusted R2                    0.014
## Residual Std. Error      1.830 (df = 8116)
## F Statistic          58.190*** (df = 2; 8116)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

c. Using diagnostic plots, background knowledge, and statistical tests, assess all five assumptions of the CLM. When an assumption is violated, state what response you will take. As part of this process, you should decide what transformation (if any) to apply to each variable. Iterate against your model until your satisfied that at least four of the five assumption have been reasonably addressed.

1. **IID Data:** The data is gather by the reaerch the team based on predefined catagories of: "Recently Featured", "Most Viewed", "Top Rated" and "Most Discussed". Then there is a search / crawl based on the same related catagories. This leads to NOT an independant assumption, as the catagories are related. The data is gathered at different points in time so the identical assumption is met.

2. **No Perfect Colinearity:** There is no perfect colineraty between the features of views and length. This is tested with the buitlin lm() function. We can also look at our coefficients, and notice that R has not dropped any variables.

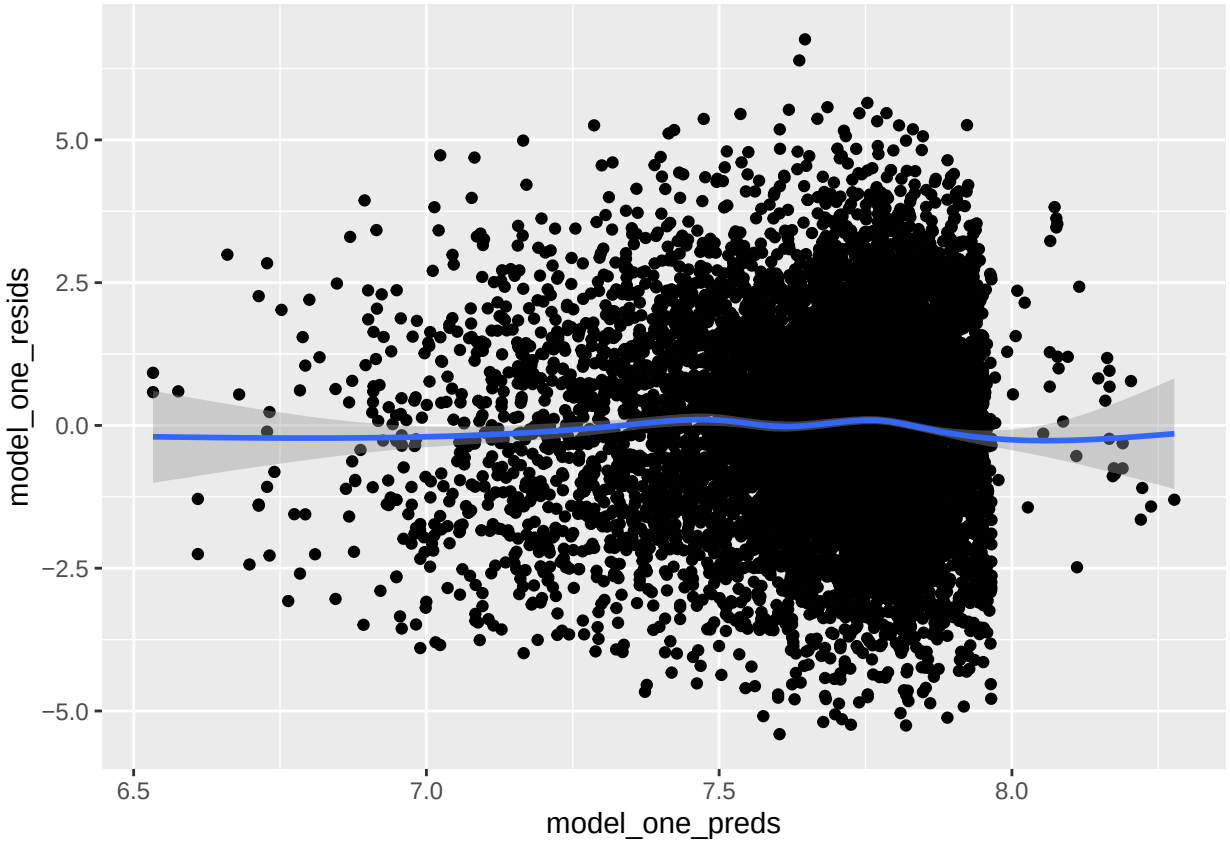
```
model$coefficients
```

```
##    (Intercept) average_rating    log(length)
##      6.2124738      0.1558825      0.1499482
```

3. **Linear Conditional Expectation:** To assess whether there is a linear conditional expectation, we've learned to look at the predicted vs. residuals of the model. We can see that the we have a more linear pattern in this plot.

```
d %>%
  mutate(
    model_one_preds = predict(model),
    model_one_resids = resid(model)
  ) %>%
  ggplot(aes(model_one_preds, model_one_resids)) +
  geom_point() +
  stat_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

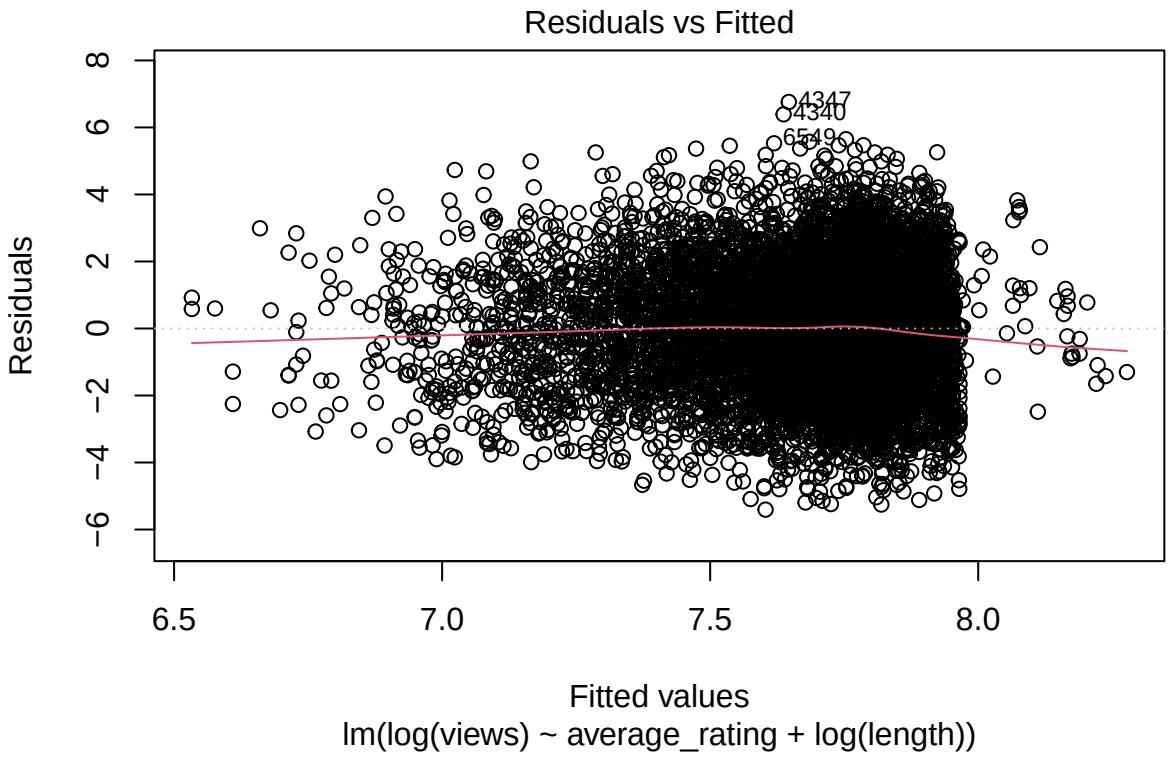


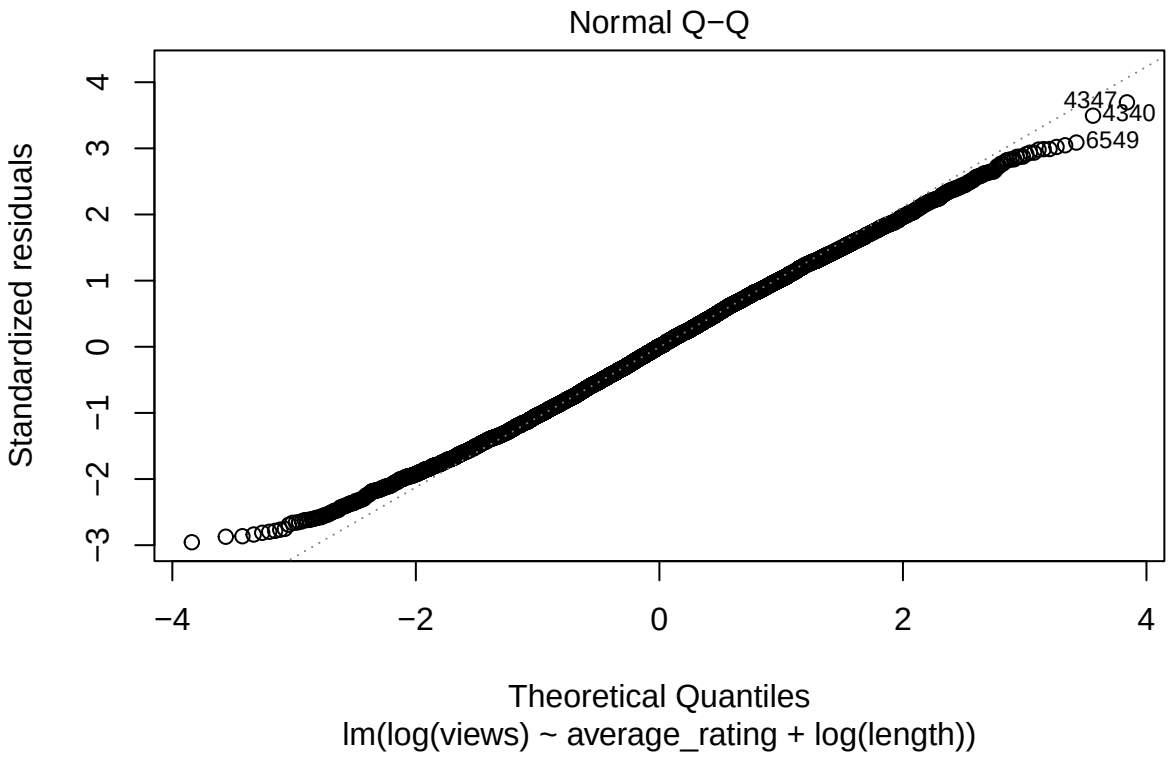
> 4. **Homoskedastic Errors:** we can examine the residuals versus fitted plot again. We can examine the scale-location plot, Homoskedasticity would show up on this plot as a flat smoothing curve. We can also use the Breusch-Pagan test. From the output we can see that the p-value of the test is 0.006429. Since this p-value is less than 0.05, we reject the null hypothesis. We do have sufficient evidence to say that heteroscedasticity is present in the regression model.

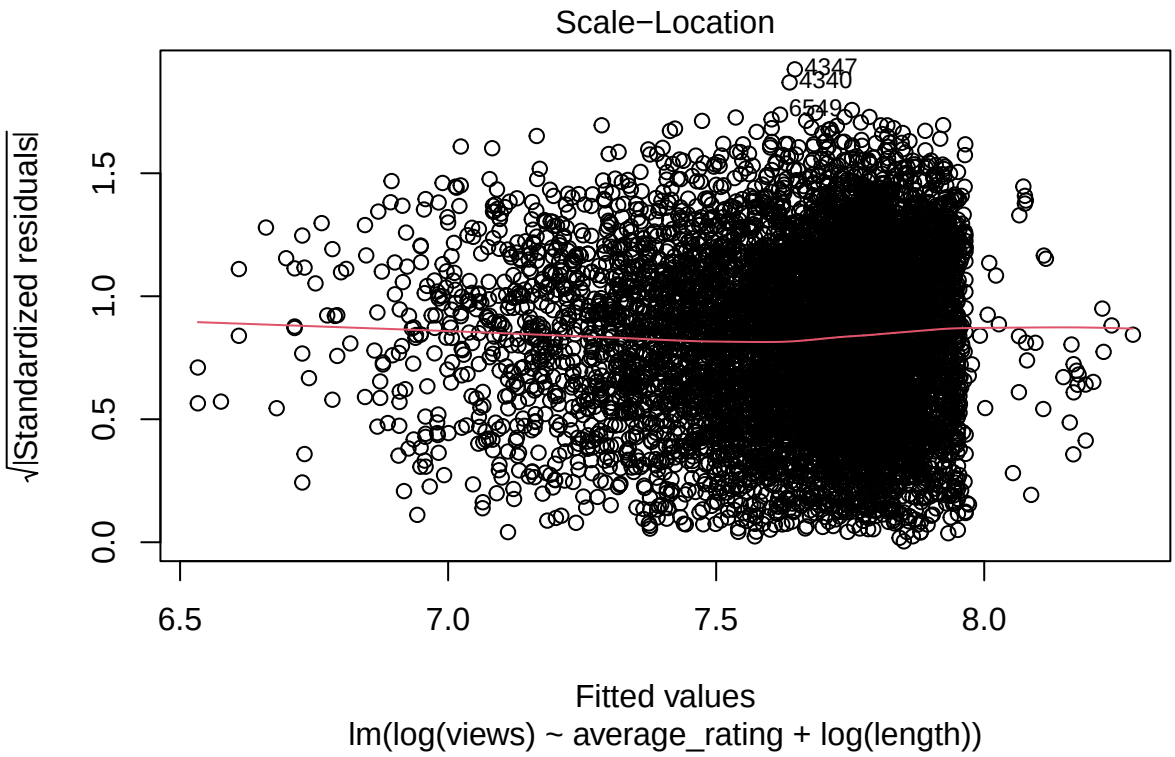
```
bptest(model)
```

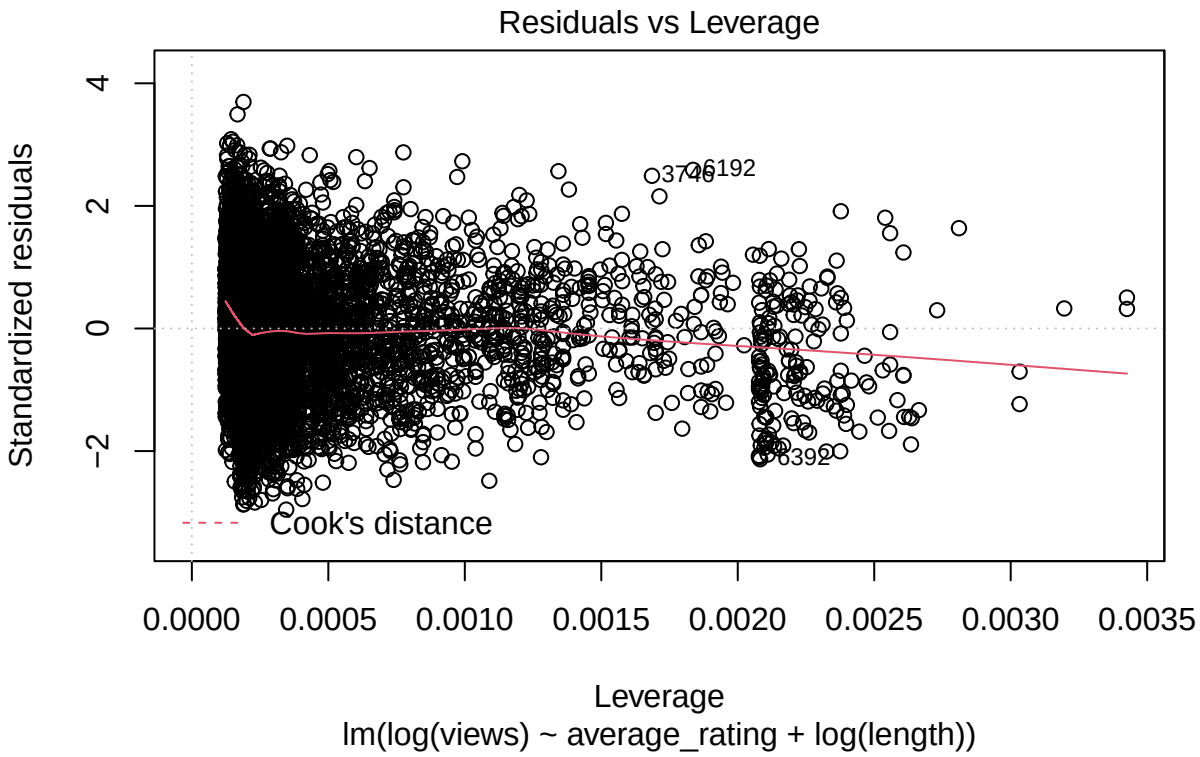
```
##
##  studentized Breusch-Pagan test
##
## data:  model
## BP = 10.094, df = 2, p-value = 0.006429
```

```
plot(model)
```


Residuals vs Fitted
Residuals
8
6
4
2
0
−2
−4
−6
4347
4340
6549
6.5
7.0
7.5
8.0
Fitted values
lm(log(views) ~ average_rating + log(length))

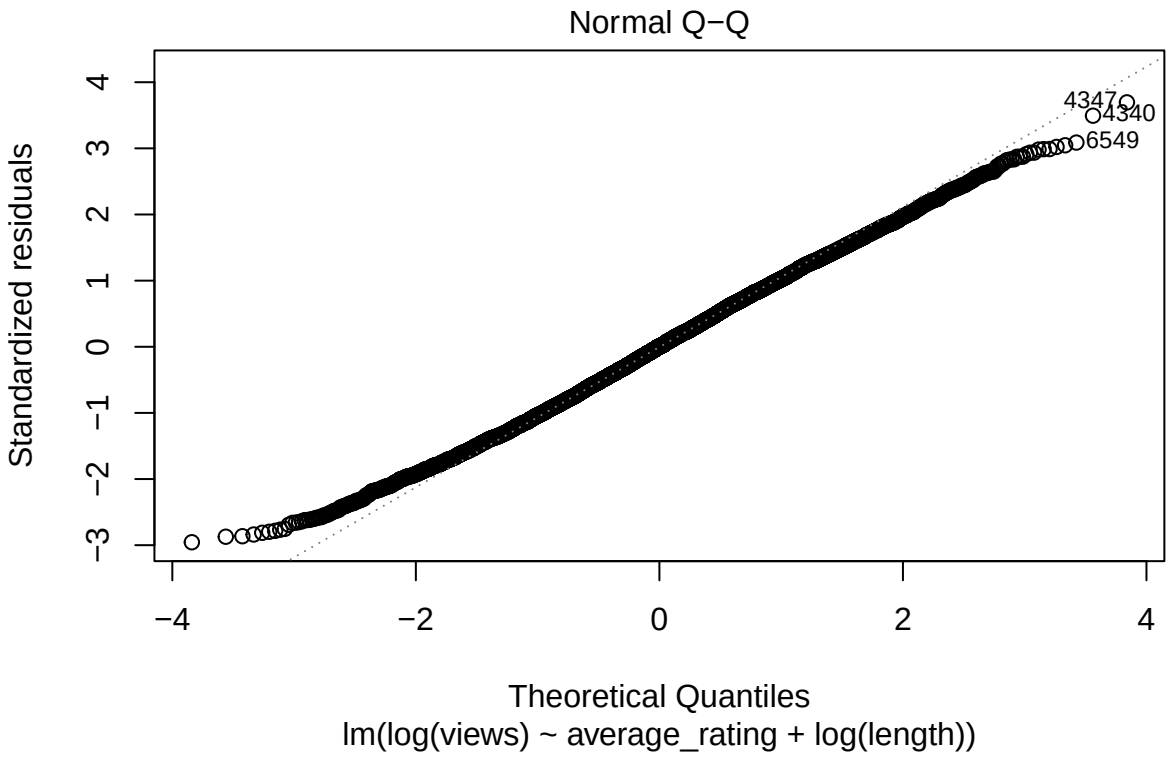

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(views) ~ average_rating + log(length))
4347
4340
6549
−3
−2
−1
0
1
2
3
4
−4
−2
0
2
4


Scale−Location
4347
4340
6549
√|Standardized residuals|
0.0
0.5
1.0
1.5
6.5
7.0
7.5
8.0
Fitted values
lm(log(views) ~ average_rating + log(length))



5. **Normally Distributed Errors:** The qq plot shows specific ways in which the data deviate from normality. The Q-Q plot shows that most of the points in the middle are close to the fitted line, which means that the residuals are matching closly to the normal distribution in the middle section, and only starts deviating on both tails. The shape of the Q-Q plots indicates that our errords distribution has tails thinner than the normally distributed.

```
plot(model, which=2)
```


Normal Q−Q
Standardized residuals
−3
−2
−1
0
1
2
3
4
4347
4340
6549
−4
−2
0
2
4
Theoretical Quantiles
lm(log(views) ~ average_rating + log(length))