Unit 12 Homework

w203: Statistics for Data Science

July 24, 2022

Part 2 - CLM Practice

For the following questions, your task is to evaluate the Classical Linear Model assumptions. It is not enough to say that an assumption is met or not met; instead, present evidence based on your background knowledge, visualizations, and numerical summaries.

The file `videos.txt` contains 9618 observations of videos shared on YouTube. It was created by Cheng, Dale and Liu at Simon Fraser University. Please see this link for details about how the data was collected.

You wish to run the following regression:

$$ln(\text{views}) = \beta_0 + \beta_1 \text{rate} + \beta_3 \text{length}$$

The variables are as follows:

- `views`: the number of views by YouTube users.
- `rate`: This is the average of the ratings that the video received. You may think of this as a proxy for video quality. (Notice that this is different from the variable `ratings` which is a count of the total number of ratings that a video has received.)
- `length:` the duration of the video in seconds.

1. Evaluate the **IID** assumption.
2. Evaluate the **No perfect Colinearity** assumption.
3. Evaluate the **Linear Conditional Expectation:** assumption.
4. Evaluate the **Homoskedastic Errors:** assumption.
5. Evaluate the **Normally Distributed Errors:** assumption.

```r
df <- read.delim('videos.txt')
```

Solution

1.IID

(1) Independent There are competition between video makers, so some videos are dependent on other videos. This independence assumption does not apply.

(2) Identical distributed youtude algorithm is same for every draws, so all the sample videos are from the same and identical distribution.

2.No perfect Colinearity

The rate means the proxy for the video quality, it is not related with video length. Therefore, this assumption "No perfect colinearity" is met.

3.Linear Conditional Expectation

```
# check empty row
summary(df['rate'])
```

```
##       rate
##  Min.   :0.000
##  1st Qu.:3.400
##  Median :4.670
##  Mean   :3.744
##  3rd Qu.:5.000
##  Max.   :5.000
##  NA's   :9
```

```
# drop ampty rows
df <- df[!is.na(df$views),]
dim(df)
```

```
## [1] 9609    9
```

```
#find Q1, Q3, and interquartile range for values in column A
Q1 <- quantile(df$rate, .25)
Q3 <- quantile(df$rate, .75)
IQR <- IQR(df$rate)

#only keep rows in dataframe that have values within 1.5*IQR of Q1 and Q3
df <- subset(df, df$rate> (Q1 - 1.5*IQR) & df$rate< (Q3 + 1.5*IQR))

#view row and column count of new data frame
dim(df)
```

```
## [1] 8119    9
```

```
#find Q1, Q3, and interquartile range for values in column A
Q1 <- quantile(df$length, .25)
Q3 <- quantile(df$length, .75)
IQR <- IQR(df$length)

#only keep rows in dataframe that have values within 1.5*IQR of Q1 and Q3
df <- subset(df, df$length> (Q1 - 1.5*IQR) & df$length< (Q3 + 1.5*IQR))

#view row and column count of new data frame
dim(df)
```

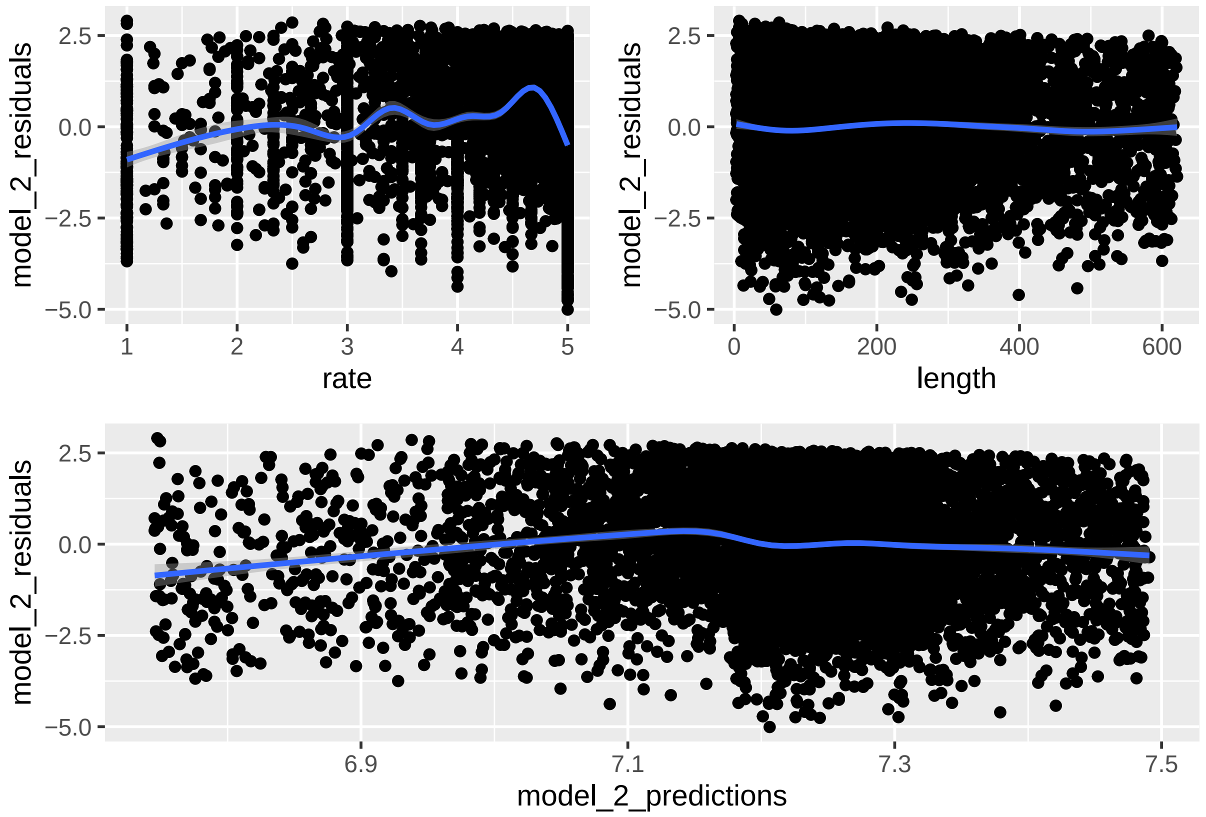
```
## [1] 7968    9
```

```
#find Q1, Q3, and interquartile range for values in column A
Q1 <- quantile(df$views, .25)
Q3 <- quantile(df$views, .75)
IQR <- IQR(df$views)

#only keep rows in dataframe that have values within 1.5*IQR of Q1 and Q3
df <- subset(df, df$views> (Q1 - 1.5*IQR) & df$views< (Q3 + 1.5*IQR))

#view row and column count of new data frame
dim(df)
```

```
## [1] 6962    9
```

```r
model_2 <- lm(log(views) ~ rate + length, data=df)

# df_2 <- df[!is.na(df$views),]
# df <- filter(df, views < 1000000)
# boxplot(df$views<1000000, data = df, subset = df$views)

df <- df %>%
  mutate(
    model_2_predictions = predict(model_2),
    model_2_residuals = resid(model_2)
  )
```

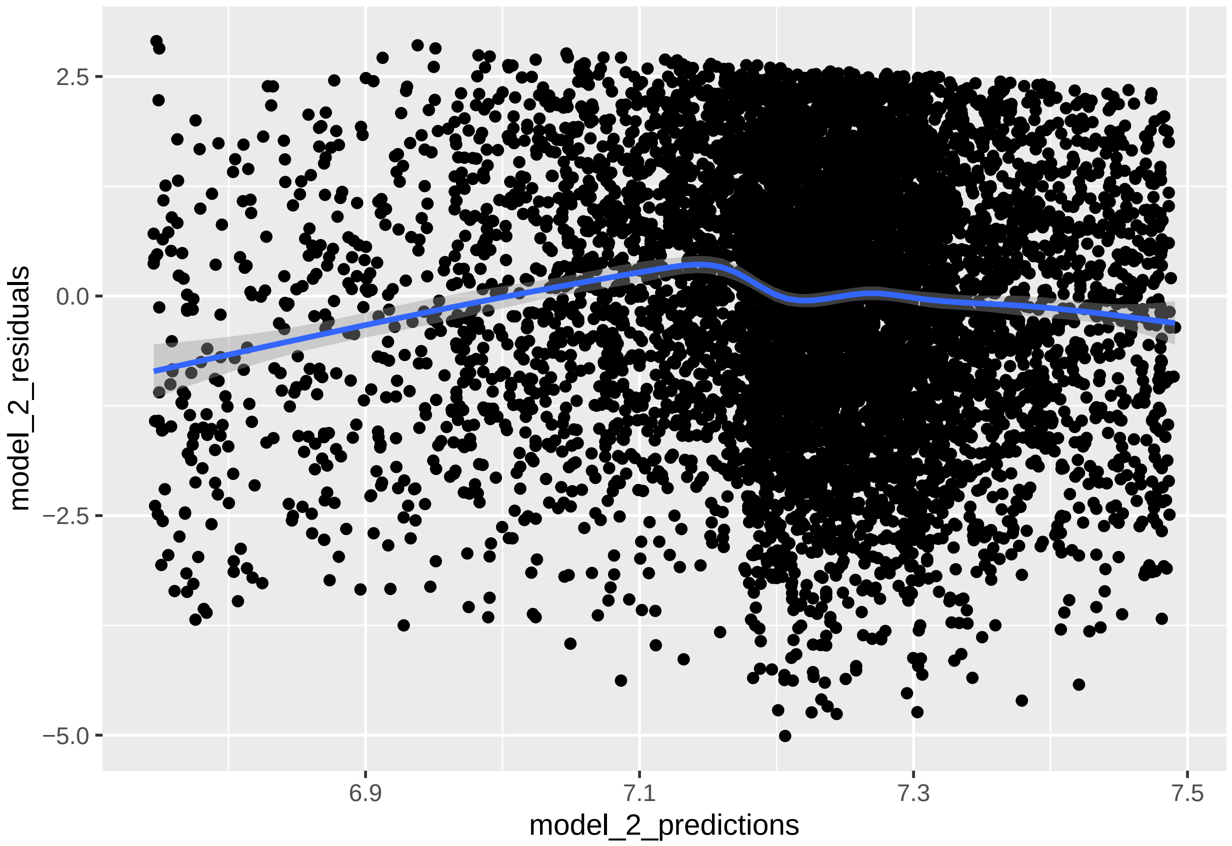
```r
plot_model_1a <- df %>%
  ggplot(aes(x = rate, y = model_2_residuals)) +
  geom_point() + stat_smooth()

plot_model_1b <- df %>%
  ggplot(aes(x = length, y = model_2_residuals)) +
  geom_point() + stat_smooth()


plot_model_1c <- df %>%
  ggplot(aes(x = model_2_predictions, y = model_2_residuals)) +
  geom_point() + stat_smooth()

(plot_model_1a | plot_model_1b) / plot_model_1c
```



4. Homoskedastic Errors

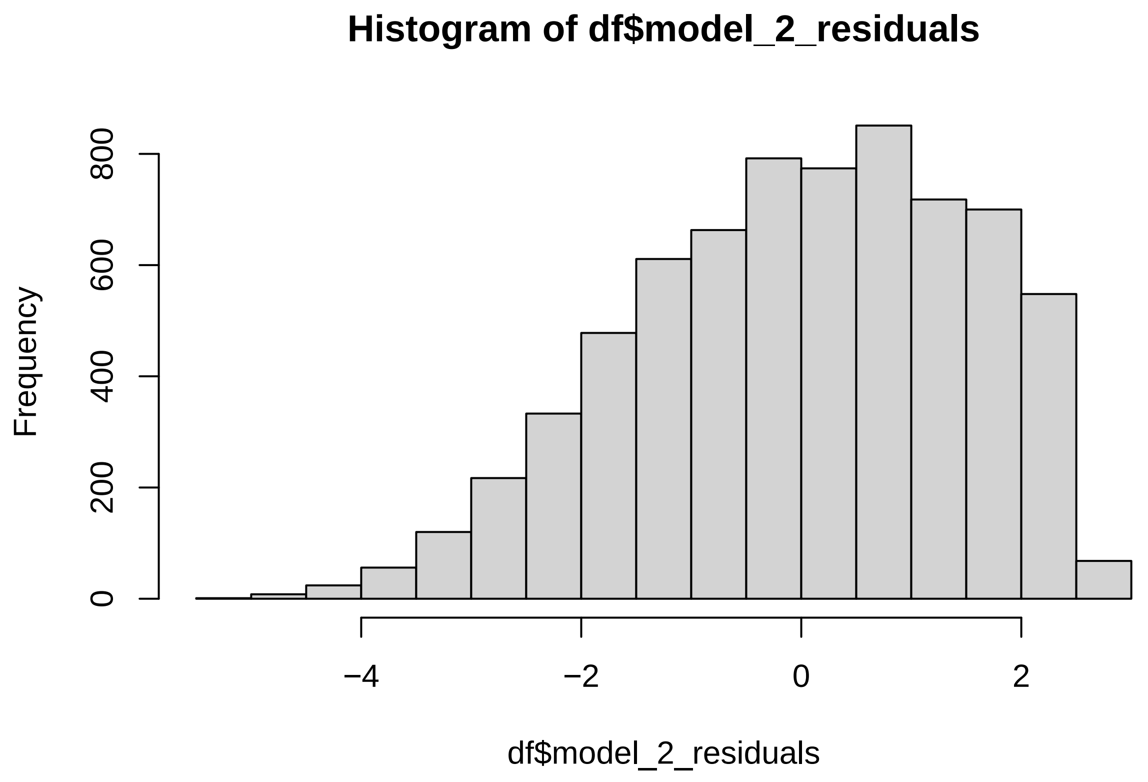
```
plot_model_1c
```

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



5. Normally Distributed Errors

```
hist(df$model_2_residuals)
```


Histogram of df$model_2_residuals
Frequency
df$model_2_residuals