# Final Project 603

Jerin Jacob

03/10/2023

Now a days, movies are a well marketed entertainment product. Just like any other products in the market, movies are also having an allocated a marketing budget and promotional activities are done in scale. This often result in the opening weekend's gross ticketing volume to rise. But are the pre-release promotional activities helping the movie to collect more or is it just creating a hype initially? Or does the movie's gross collection is not at all dependant on pre release promotions? This dataset has 200 highest grossing movies of 2022. It has both the opening week's gross as well as the total gross collection of the movies, along with other variables. Assuming that opening week's collection is depending on the pre-release promotion, by looking on the relationship between opening week's gross and total gross, I am trying to see how the pre-release activities help the producers earn more in boxoffice.

**Research Question**: To what extent does the success of a movie depend on its opening week's collection?

**Hypothesis**: Opening week's collection is positively correlated with the Box Office total collection.

Loading all the packages required for the project.

```
library(readxl)
library(tidyverse)
library(lubridate)
library(dplyr)
library(stringr)
```

## Reading the data

```
df <- read_excel("_data/project_data.xlsx") |>
  as.data.frame()
head(df)
```

```
##   Rank                                     Release      Budget     Box Office     Gross max_th   Oper
## 1    1                           Top Gun: Maverick 177 million $1.493 billion 718732821   4751 126707
## 2    2                    Avatar: The Way of Water 460 million $2.319 billion 636955746   4340 134100
## 3    3              Black Panther: Wakanda Forever 250 million $859.1 million 453474324   4396 181339
## 4    4 Doctor Strange in the Multiverse of Madness 200 million $955.8 million 411331607   4534 187420
## 5    5                    Jurassic World: Dominion 185 million $1.004 billion 376851080   4697 145075
## 6    6                    Minions: The Rise of Gru 100 million $940.5 million 369695210   4427 107010
##   open_th  Open Close                         Distributor  int_gross world_gross
## 1    4735 44708 44911                  Paramount Pictures  770000000  1488732821
## 2    4202 44911    NA                20th Century Studios 1539273359  2176229105
## 3    4396 44876    NA Walt Disney Studios Motion Pictures  389276658   842750982
## 4    4534 44687    NA Walt Disney Studios Motion Pictures  544444197   955775804
## 5    4676 44722 44827                  Universal Pictures  625127000  1001978080
## 6    4391 44743    NA                  Universal Pictures  569933000   939628210
```

There are 14 variables with 200 rows.

### COLUMN DESCRIPTION

'Rank': rank of the movie 'Release': release date of the movie 'Budget': The budget of the movie production 'Box Office': The total Box Office collection 'Gross': domestic gross of the movie 'max_th': maximum number of theaters the movie was released in 'Opening': gross on opening weekend 'perc_tot_gr': domestic percentage of the total gross 'open_th': number of theaters the movie opened in 'Open': opening date 'Close': closing date 'Distributor': name of the distributor 'int_gross': international gross 'world_gross': worldwide gross

- 'Release': release date of the movie
- 'Distributor': name of the distributor
- 'Small_Dist': Whether a small distributor or not
-
- 'Open_date': Date of release
- 'season': The season in which the movie was released
- 'Opening': gross on opening weekend
- 'open_th': number of theaters the movie opened in
- 'max_th': maximum number of theaters the movie was released in

## Cleaning the Data

For our purpose of analysis, we need to clean and transform the data a bit.

First, the Budget and Boxoffice columns are cleaned using stringr function so that the column values are numeric. We are getting rid of the character part in those values including '$' and the value unit. Also, there are certain values which are in Indian Rupees and South Korean won. So we need to convert thoses values to US dollars. The data for columns Budget and Box office were taken from Wikipedia.

```
df <- df |>
  mutate(Budget = gsub("\\$", "", Budget)) |>
  mutate(Budget = sub(".*-", "", Budget)) |>
  mutate(`Box Office` = gsub("\\$", "", `Box Office`))

df[c('Budget', 'Unit')] <- str_split_fixed(df$Budget, ' ', 2)

df$Budget <- as.numeric(df$Budget)
```

```
## Warning: NAs introduced by coercion
```

```
df$Budget <- ifelse(df$Unit == "million", df$Budget * 1000000,
                    ifelse(df$Unit == "billion", df$Budget * 1000000000,
                           ifelse(df$Unit == "Kmillion", df$Budget * 1000000*0.00074,
                                  ifelse(df$Unit == "Kbillion", df$Budget * 1000000000*0.00074,
                                         ifelse(df$Unit == "crore", df$Budget * 10000000*80, df$Budget))

df[c('Box Office', 'BXUnit')] <- str_split_fixed(df$`Box Office`, ' ', 2)

df$`Box Office` <- as.numeric(df$`Box Office`)
```

```
## Warning: NAs introduced by coercion
```

```
df$`Box Office` <- ifelse(df$BXUnit == "million", df$`Box Office` * 1000000,
                    ifelse(df$BXUnit == "billion", df$`Box Office` * 1000000000,
                           ifelse(df$BXUnit == "Kmillion", df$`Box Office` * 1000000*0.00074,
                                  ifelse(df$BXUnit == "Kbillion", df$`Box Office` * 1000000000*0.00074,
                                         ifelse(df$BXUnit == "crore", df$`Box Office` * 10000000*80, df$
```

Since the original dataset from Kaggle had a column named world_gross, we can compare both variables and assume that the highest value in either of the column can be considerd as the final world_gross.

```
df$`Box Office` <- ifelse(df$`Box Office` < df$world_gross, df$world_gross, df$`Box Office`)
```

We can count the number of movies in the list for each distributor and any distributor who don't have more than 3 movies in their name can be considered as a smaller distributor and thus assuming that they won't have cash rich promotional campaigns that would lead to an audience pull to the theatre in the initial week.

```
df <- df %>% group_by(Distributor) %>% mutate(Count=n_distinct(`Box Office`))

df$Small_Dist <- ifelse(df$Count <= 3, 1, 0)
```

We can convert the dbl to date format and set the reference date so that the dates are correct. After that, from the Open_date, we can categorize thsoe dates to the season so that it can be used as a confounder. Seasons might have some effect on the theatre footfall and thereby, box office collections.

```
df$Open_date <- as.Date(df$Open, origin = "1899-12-30")

# Create a new column with the season for each date
df <- df %>%
  mutate(season = case_when(
    between(month(Open_date), 3, 5) ~ "Spring",
    between(month(Open_date), 6, 8) ~ "Summer",
    between(month(Open_date), 9, 11) ~ "Fall",
    TRUE ~ "Winter"
  ))
```

```
head(df)
```

```
## # A tibble: 6 x 20
## # Groups:   Distributor [4]
##    Rank Release        Budget `Box Office`  Gross max_th Opening perc_tot_gr open_th  Open Close Dist
##   <dbl> <chr>           <dbl>        <dbl>  <dbl>  <dbl>   <dbl>       <dbl>   <dbl> <dbl> <dbl> <chr
## 1     1 Top Gun: Mave~ 1.77e8   1493000000 7.19e8   4751  1.27e8        17.6    4735 44708 44911 Para
## 2     2 Avatar: The W~ 4.6 e8   2319000000 6.37e8   4340  1.34e8        21.1    4202 44911    NA 20th
## 3     3 Black Panther~ 2.5 e8    859100000 4.53e8   4396  1.81e8        40      4396 44876    NA Walt
## 4     4 Doctor Strang~ 2   e8    955800000 4.11e8   4534  1.87e8        45.6    4534 44687    NA Walt
## 5     5 Jurassic Worl~ 1.85e8   1004000000 3.77e8   4697  1.45e8        38.5    4676 44722 44827 Univ
## 6     6 Minions: The ~ 1   e8    940500000 3.70e8   4427  1.07e8        28.9    4391 44743    NA Univ
## # i 7 more variables: world_gross <dbl>, Unit <chr>, BXUnit <chr>, Count <int>, Small_Dist <dbl>, Ope
## #   season <chr>
```

Checking for NA values in each variable. There are 90 NA values in Budget variable, 38 in Box Office, 155 in Close date variable and 3 in int_gross. All othe variables seems to be good in terms of NA values.

```
colSums(is.na(df))
```

```
##        Rank     Release      Budget  Box Office       Gross      max_th     Opening perc_tot_gr
##           0           0          90          38           0           0           0           0
##        Open       Close Distributor   int_gross world_gross        Unit      BXUnit       Count  Smal
##           0         155           0           3           0           0           0           0
##   Open_date      season
##           0           0
```

```r
table(df$Small_Dist)
```

```
##
##   0   1
## 140  60
```

Since 155 values of Close are NAs, it is better not to include that variable in the analysis. Most of the NA values are for the movies by small distributors, which need to be noted.

```r
df <- subset(df, select = -Close)

df <- na.omit(df)
```

Have an idea about the structure of the dataset.

```r
str(df)
```

```
## gropd_df [104 x 19] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ Rank       : num [1:104] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Release    : chr [1:104] "Top Gun: Maverick" "Avatar: The Way of Water" "Black Panther: Wakanda Fo
##  $ Budget     : num [1:104] 1.77e+08 4.60e+08 2.50e+08 2.00e+08 1.85e+08 1.00e+08 2.00e+08 2.50e+08
##  $ Box Office : num [1:104] 1.49e+09 2.32e+09 8.59e+08 9.56e+08 1.00e+09 ...
##  $ Gross      : num [1:104] 7.19e+08 6.37e+08 4.53e+08 4.11e+08 3.77e+08 ...
##  $ max_th     : num [1:104] 4751 4340 4396 4534 4697 ...
##  $ Opening    : num [1:104] 1.27e+08 1.34e+08 1.81e+08 1.87e+08 1.45e+08 ...
##  $ perc_tot_gr: num [1:104] 17.6 21.1 40 45.6 38.5 28.9 36.3 42 37.8 39.8 ...
##  $ open_th    : num [1:104] 4735 4202 4396 4534 4676 ...
##  $ Open       : num [1:104] 44708 44911 44876 44687 44722 ...
##  $ Distributor: chr [1:104] "Paramount Pictures" "20th Century Studios" "Walt Disney Studios Motion P
##  $ int_gross  : num [1:104] 7.70e+08 1.54e+09 3.89e+08 5.44e+08 6.25e+08 ...
##  $ world_gross: num [1:104] 1.49e+09 2.18e+09 8.43e+08 9.56e+08 1.00e+09 ...
##  $ Unit       : chr [1:104] "million" "million" "million" "million" ...
##  $ BXUnit     : chr [1:104] "billion" "billion" "million" "million" ...
##  $ Count      : int [1:104] 12 4 9 9 19 19 6 9 12 6 ...
##  $ Small_Dist : num [1:104] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Open_date  : Date[1:104], format: "2022-05-27" "2022-12-16" "2022-11-11" "2022-05-06" ...
##  $ season     : chr [1:104] "Spring" "Winter" "Fall" "Spring" ...
##  - attr(*, "groups")= tibble [32 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ Distributor: chr [1:32] "-" "20th Century Studios" "A24" "Blue Fox Entertainment" ...
##   ..$ .rows      : list<int> [1:32]
##   .. ..$ : int [1:3] 87 94 95
##   .. ..$ : int [1:4] 2 35 38 42
##   .. ..$ : int [1:5] 26 54 60 67 102
##   .. ..$ : int 100
##   .. ..$ : int 66
##   .. ..$ : int 97
##   .. ..$ : int 62
##   .. ..$ : int 103
##   .. ..$ : int [1:3] 25 32 49
##   .. ..$ : int 76
##   .. ..$ : int [1:6] 37 41 63 74 80 86
##   .. ..$ : int 82
##   .. ..$ : int 89
##   .. ..$ : int [1:3] 85 98 104
##   .. ..$ : int [1:3] 50 51 73
##   .. ..$ : int [1:3] 77 81 83
##   .. ..$ : int 58
##   .. ..$ : int [1:10] 1 9 16 17 24 31 52 55 79 99
##   .. ..$ : int 88
##   .. ..$ : int 90
##   .. ..$ : int 84
##   .. ..$ : int [1:2] 39 64
##   .. ..$ : int [1:6] 13 18 22 34 44 48
##   .. ..$ : int 93
##   .. ..$ : int [1:2] 28 45
##   .. ..$ : int 91
##   .. ..$ : int [1:5] 30 69 70 72 78
##   .. ..$ : int [1:19] 5 6 11 14 19 23 27 29 33 43 ...
##   .. ..$ : int [1:7] 3 4 8 15 40 56 71
##   .. ..$ : int [1:6] 7 10 12 20 21 36
##   .. ..$ : int [1:2] 96 101
##   .. ..$ : int 92
##   .. ..@ ptype: int(0)
##   ..- attr(*, ".drop")= logi TRUE
##  - attr(*, "na.action")= 'omit' Named int [1:96] 40 42 47 54 61 63 66 69 75 77 ...
##   ..- attr(*, "names")= chr [1:96] "40" "42" "47" "54" ...
```

Using the glimpse() function, let's have a look at how our data would look like!

```
glimpse(df )
```

```
## Rows: 104
## Columns: 19
## Groups: Distributor [32]
## $ Rank         <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 
## $ Release      <chr> "Top Gun: Maverick", "Avatar: The Way of Water", "Black Panther: Wakanda Foreve
## $ Budget       <dbl> 1.77e+08, 4.60e+08, 2.50e+08, 2.00e+08, 1.85e+08, 1.00e+08, 2.00e+08, 2.50e+08,
## $ `Box Office` <dbl> 1493000000, 2319000000, 859100000, 955800000, 1004000000, 940500000, 770945583,
## $ Gross        <dbl> 718732821, 636955746, 453474324, 411331607, 376851080, 369695210, 369345583, 34
## $ max_th       <dbl> 4751, 4340, 4396, 4534, 4697, 4427, 4417, 4375, 4258, 4402, 4121, 3932, 4275, 3
## $ Opening      <dbl> 126707459, 134100226, 181339761, 187420998, 145075625, 107010140, 134008624, 14
## $ perc_tot_gr  <dbl> 17.6, 21.1, 40.0, 45.6, 38.5, 28.9, 36.3, 42.0, 37.8, 39.8, 8.2, 20.7, 29.6, 36
## $ open_th      <dbl> 4735, 4202, 4396, 4534, 4676, 4391, 4417, 4375, 4234, 4402, 4099, 3906, 4275, 3
## $ Open         <dbl> 44708, 44911, 44876, 44687, 44722, 44743, 44624, 44750, 44659, 44855, 44916, 44
## $ Distributor  <chr> "Paramount Pictures", "20th Century Studios", "Walt Disney Studios Motion Pictu
## $ int_gross    <dbl> 770000000, 1539273359, 389276658, 544444197, 625127000, 569933000, 401600000, 4
## $ world_gross  <dbl> 1488732821, 2176229105, 842750982, 955775804, 1001978080, 939628210, 770945583,
## $ Unit         <chr> "million", "million", "million", "million", "million", "million", "million", "m
## $ BXUnit       <chr> "billion", "billion", "million", "million", "billion", "million", "million", "m
## $ Count        <int> 12, 4, 9, 9, 19, 19, 6, 9, 12, 6, 19, 6, 9, 19, 9, 12, 12, 9, 19, 6, 6, 9, 19, 
## $ Small_Dist   <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
## $ Open_date    <date> 2022-05-27, 2022-12-16, 2022-11-11, 2022-05-06, 2022-06-10, 2022-07-01, 2022-0
## $ season       <chr> "Spring", "Winter", "Fall", "Spring", "Summer", "Summer", "Spring", "Summer", "
```

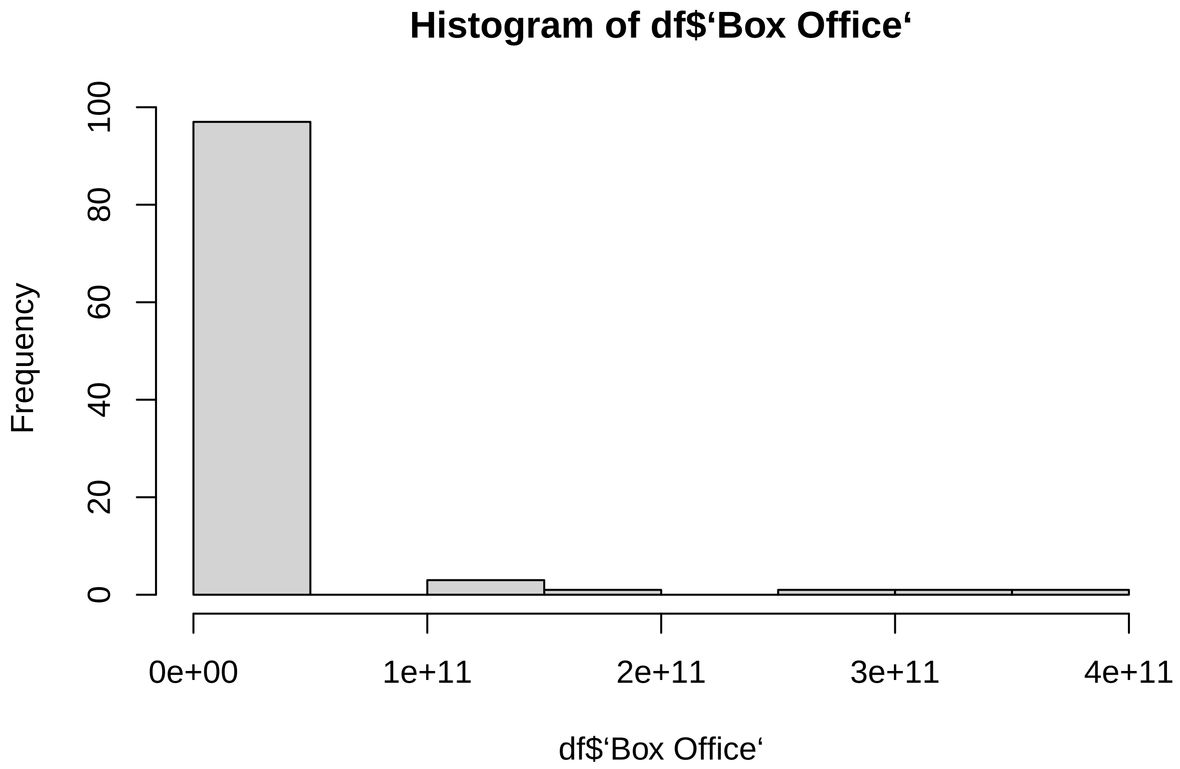
### Summary of each variables

```
summary(df)
```

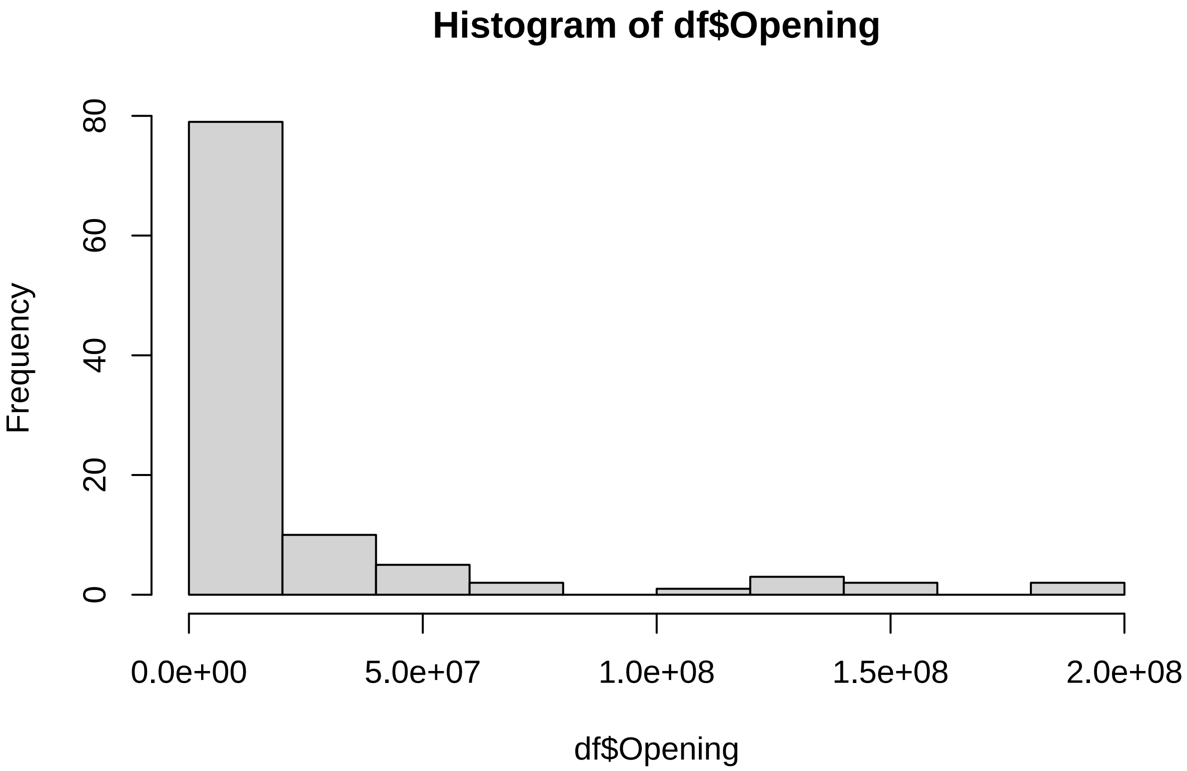
```
##       Rank          Release              Budget           Box Office           Gross
##  Min.   :  1.00   Length:104         Min.   :1.500e+05   Min.   :3.250e+05   Min.   :   325042   Min
##  1st Qu.: 26.75   Class :character   1st Qu.:1.665e+07   1st Qu.:2.170e+07   1st Qu.:  3755174   1st
##  Median : 56.50   Mode  :character   Median :3.550e+07   Median :6.535e+07   Median : 17247468   Medi
##  Mean   : 67.90                      Mean   :1.059e+10   Mean   :1.467e+10   Mean   : 67815629   Mean
##  3rd Qu.:102.50                      3rd Qu.:9.000e+07   3rd Qu.:1.966e+08   3rd Qu.: 69210756   3rd
##  Max.   :196.00                      Max.   :3.200e+11   Max.   :4.000e+11   Max.   :718732821   Max
##     Opening           perc_tot_gr       open_th            Open       Distributor          int_gross
##  Min.   :     8416   Min.   : 0.10   Min.   :   2.0   Min.   :44568   Length:104          Min.   :6.75
##  1st Qu.:   825579   1st Qu.:21.25   1st Qu.: 661.5   1st Qu.:44673   Class :character   1st Qu.:2.54
##  Median :  5128384   Median :31.85   Median :3075.0   Median :44768   Mode  :character   Median :2.46
##  Mean   : 20890734   Mean   :29.89   Mean   :2400.9   Mean   :44757                      Mean   :8.87
##  3rd Qu.: 19126885   3rd Qu.:39.85   3rd Qu.:3770.0   3rd Qu.:44841                      3rd Qu.:6.10
##  Max.   :187420998   Max.   :62.90   Max.   :4735.0   Max.   :44925                      Max.   :1.53
##   world_gross            Unit              BXUnit              Count          Small_Dist       Open_
##  Min.   :8.416e+03   Length:104         Length:104         Min.   : 1.000   Min.   :0.0000   Min.
##  1st Qu.:9.576e+06   Class :character   Class :character   1st Qu.: 4.750   1st Qu.:0.0000   1st Qu.
##  Median :4.348e+07   Mode  :character   Mode  :character   Median : 9.000   Median :0.0000   Median
##  Mean   :1.563e+08                                         Mean   : 8.721   Mean   :0.2115   Mean
##  3rd Qu.:1.445e+08                                         3rd Qu.:12.000   3rd Qu.:0.0000   3rd Qu.
##  Max.   :2.176e+09                                         Max.   :19.000   Max.   :1.0000   Max.
##     season
##  Length:104
##  Class :character
##  Mode  :character
##
##
##
```

Let's look on the distribution of data

```
hist(df$`Box Office`)
```



```
hist(df$Opening)
```



### Check correlation between Box Office and Opening

```
cor.test(df$Opening, df$`Box Office`)
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Opening and df$`Box Office`
## t = -1.0944, df = 102, p-value = 0.2764
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.29421354  0.08665663
## sample estimates:
##        cor
## -0.1077295
```

```
cor.test(log(df$Opening), log(df$`Box Office`))
```

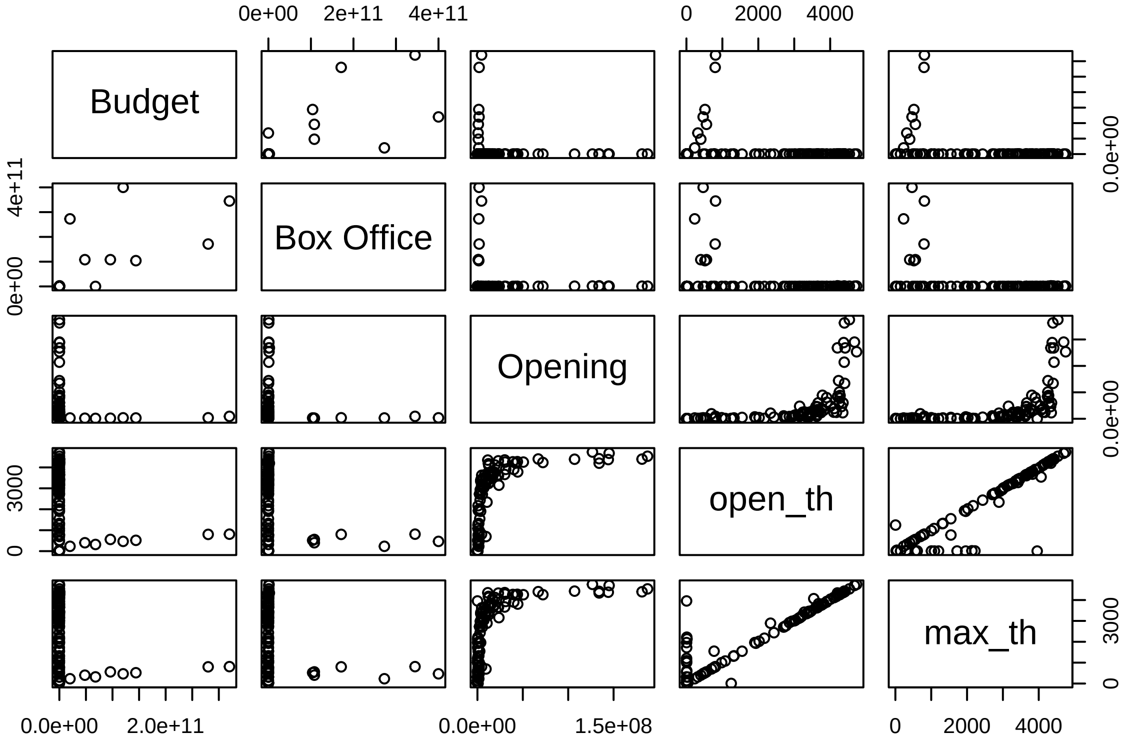
```
##
##  Pearson's product-moment correlation
##
## data:  log(df$Opening) and log(df$`Box Office`)
## t = 4.8371, df = 102, p-value = 4.681e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2610906 0.5765810
## sample estimates:
##       cor
## 0.4319587
```

The data appears to be skewed. Therefore we can try some data transformation. Log transformation would be the first option. Let's visualize the correlation of variables in the dataset.

```
pairs_df <- df[, c("Budget", "Box Office", "Opening", "open_th", "max_th")]
pairs_df
```

```
## # A tibble: 104 x 5
##       Budget `Box Office`   Opening open_th max_th
##        <dbl>        <dbl>     <dbl>   <dbl>  <dbl>
##  1 177000000   1493000000 126707459    4735   4751
##  2 460000000   2319000000 134100226    4202   4340
##  3 250000000    859100000 181339761    4396   4396
##  4 200000000    955800000 187420998    4534   4534
##  5 185000000   1004000000 145075625    4676   4697
##  6 100000000    940500000 107010140    4391   4427
##  7 200000000    770945583 134008624    4417   4417
##  8 250000000    760928081 144165107    4375   4375
##  9 110000000    405400000  72105176    4234   4258
## 10 260000000    393000000  67004323    4402   4402
## # i 94 more rows
```

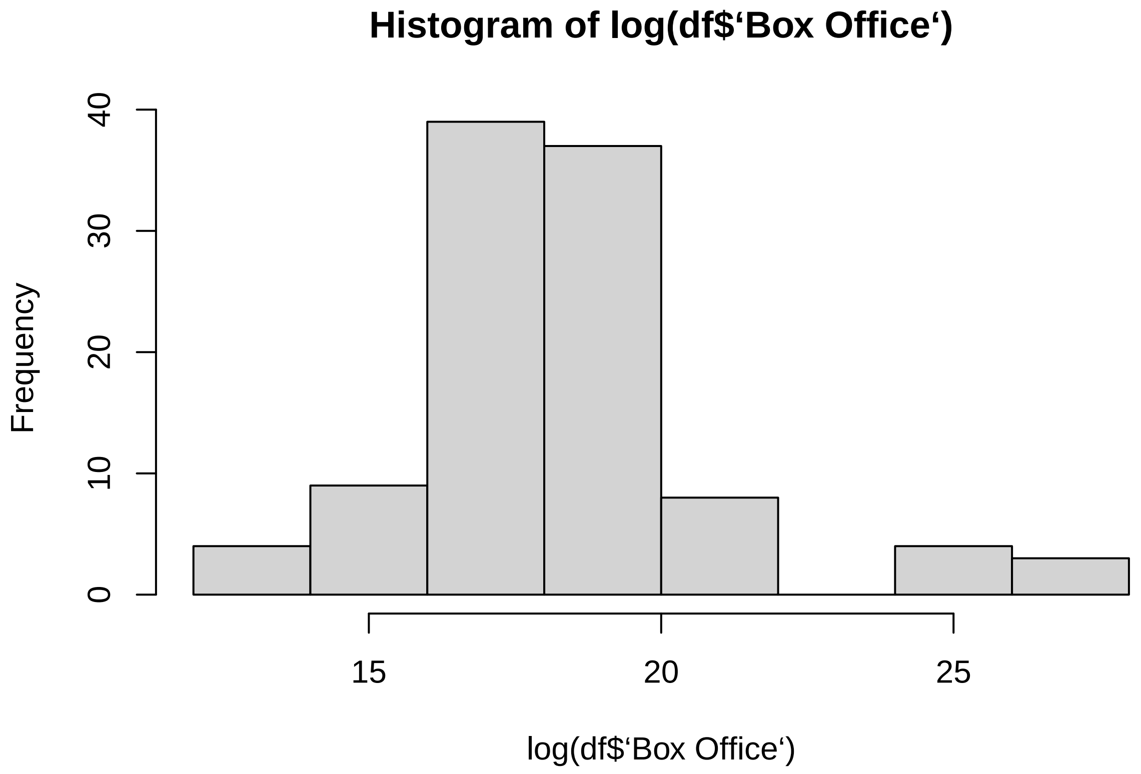
```
pairs(pairs_df)
```



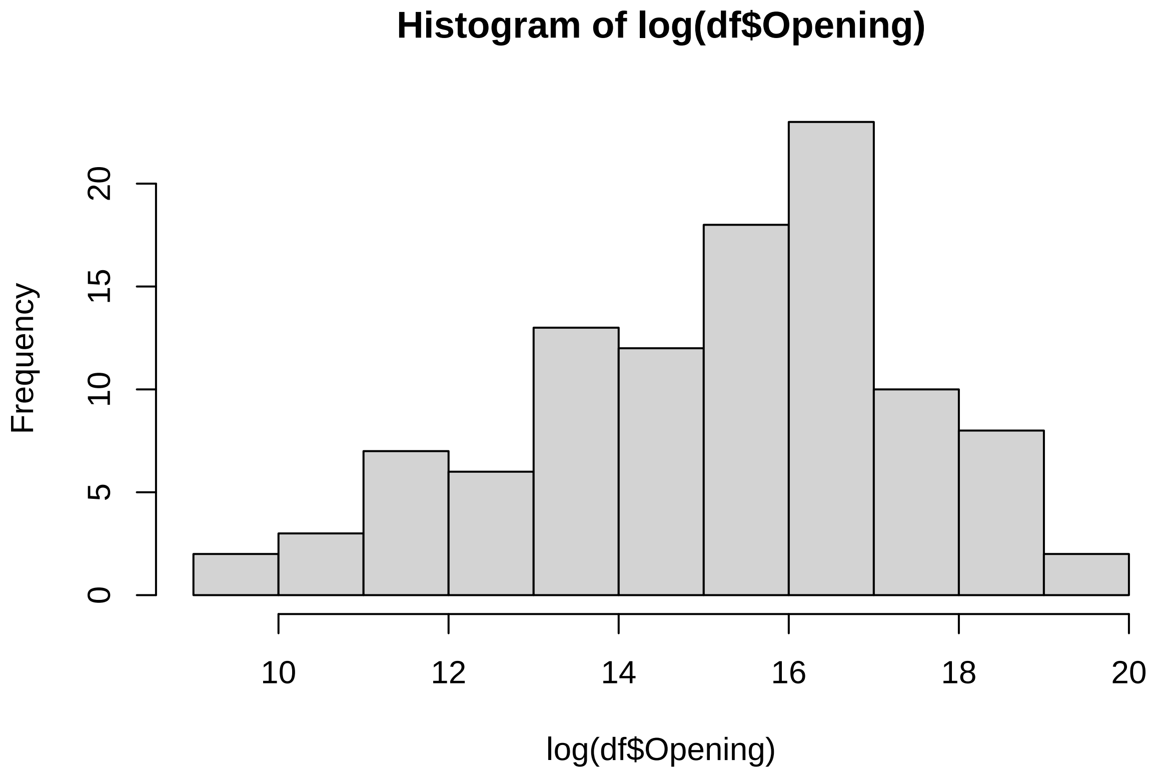
```
#cor(df)
```

From the visualization, we can conclude that all the variables are correlated by open_th and max_th are moving exactly the same in the graph which means we should drop one of them to avoid the multicollinearity.

```
hist(log(df$`Box Office`))
```



```
hist(log(df$Opening))
```



The log transformation made the data look like a normal distribution.

### Running different models to get the best fit

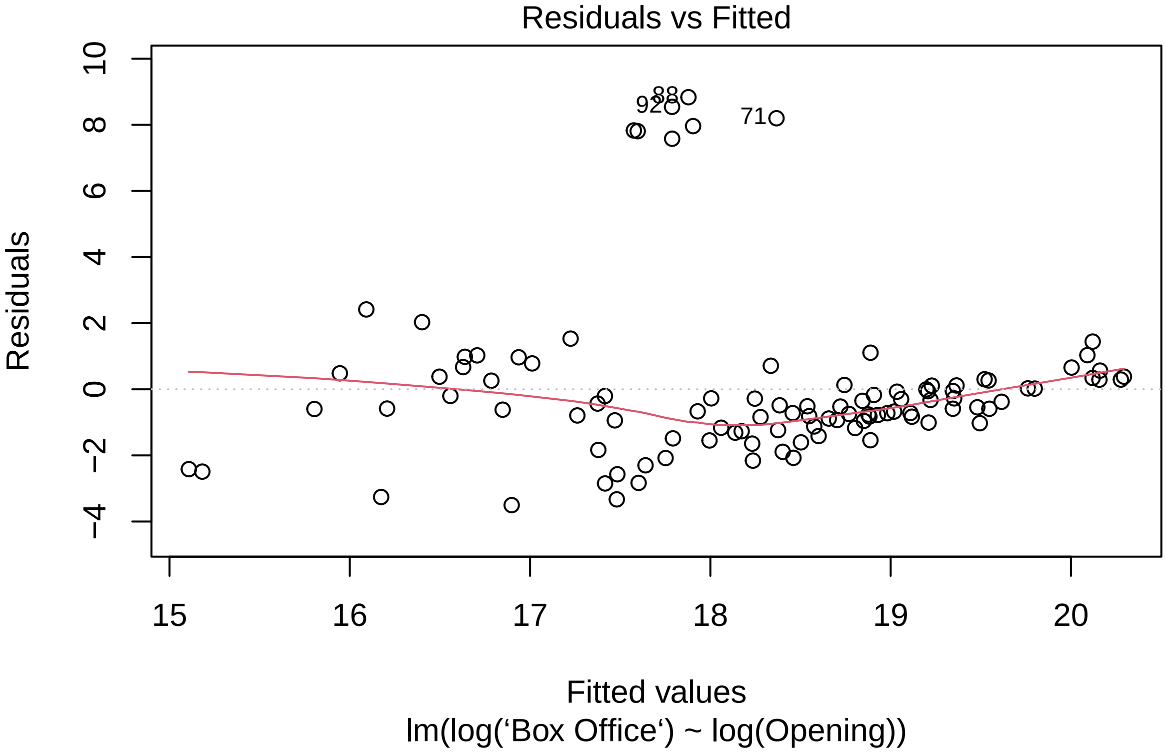
Log transformed model with only the predictor and the dependant variable

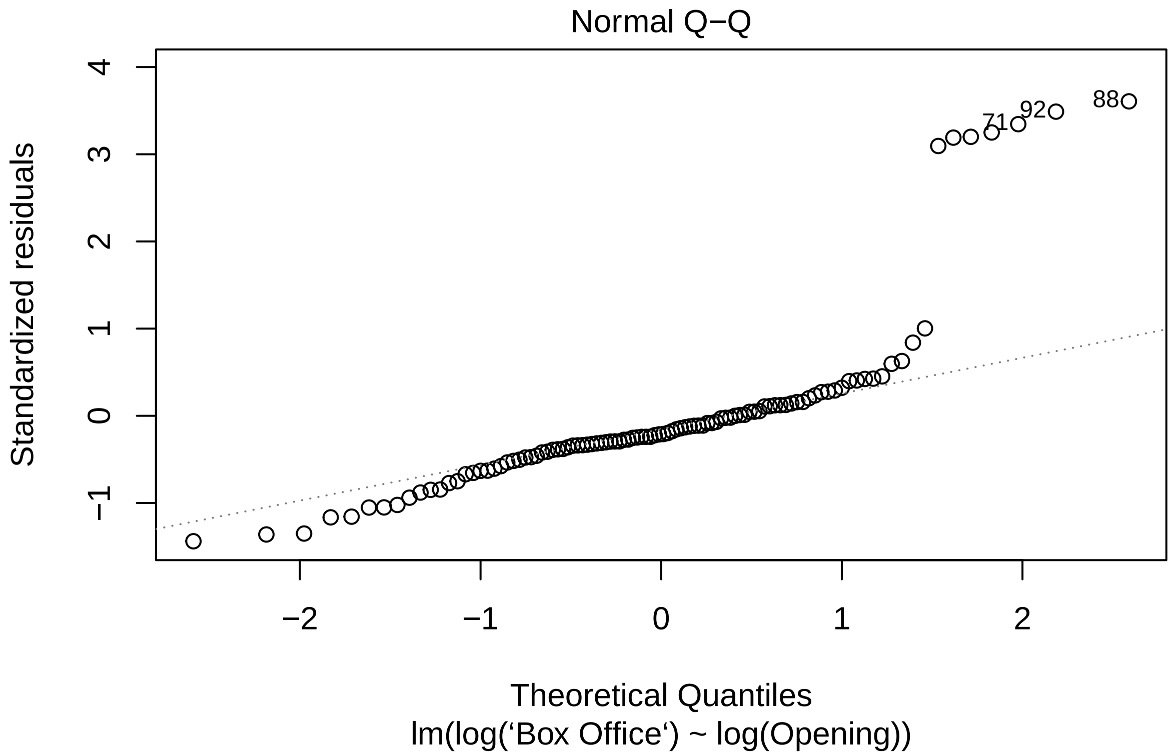
```
# Fit the multiple regression model using the log-transformed data
model1 <- lm(log(`Box Office`) ~ log(Opening), data = df)

# Print the model summary
summary(model1)
```

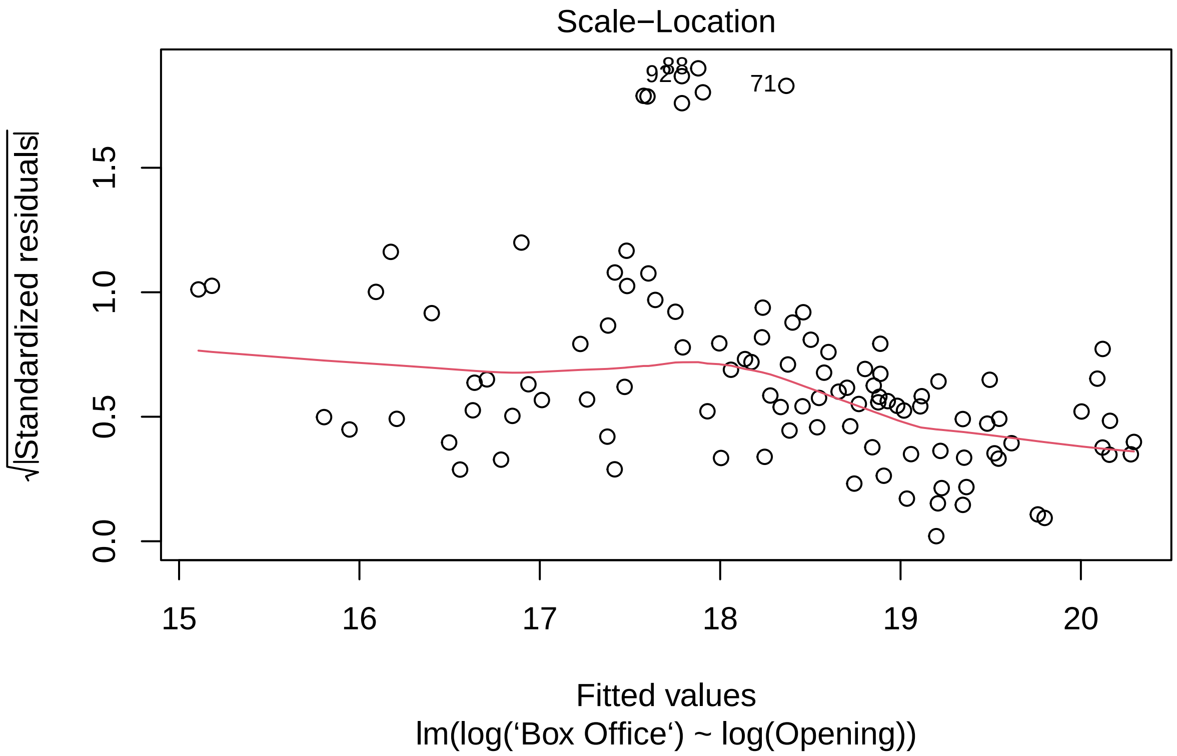
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5031 -1.0495 -0.5179  0.2968  8.8363
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   10.4248     1.6403   6.356 5.90e-09 ***
## log(Opening)   0.5181     0.1071   4.837 4.68e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

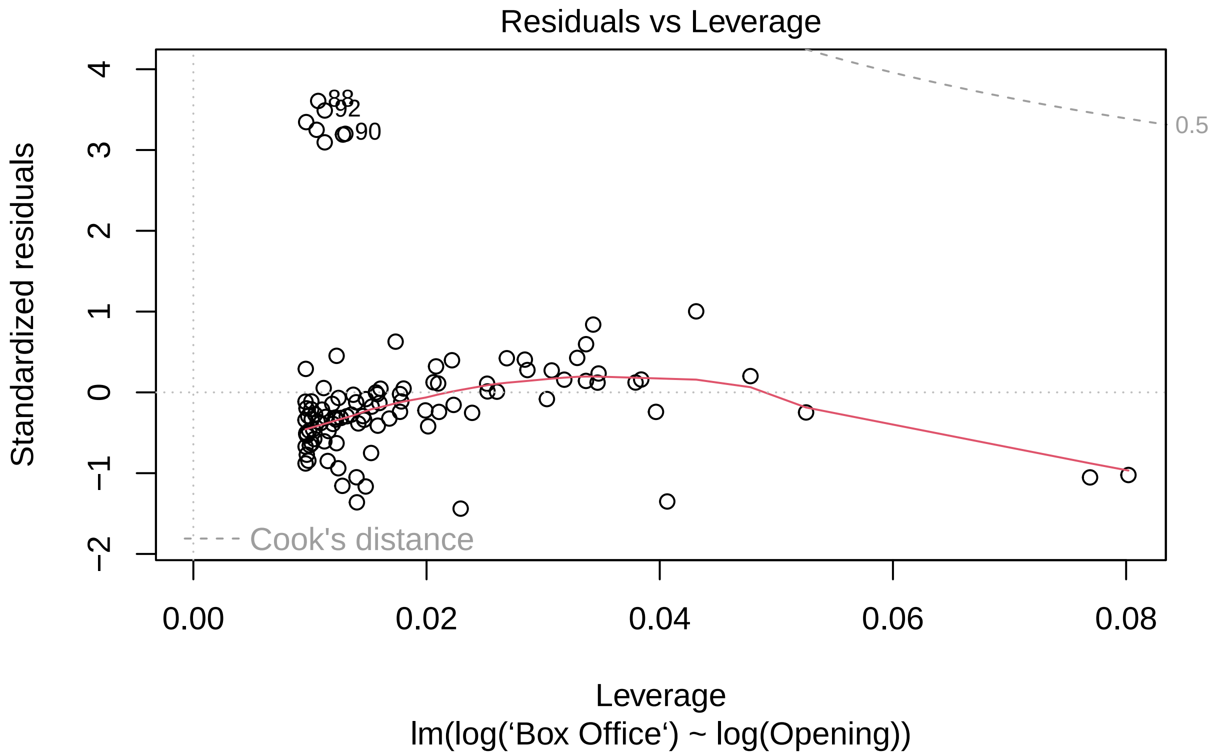
```
## Residual standard error: 2.463 on 102 degrees of freedom
## Multiple R-squared:  0.1866, Adjusted R-squared:  0.1786
## F-statistic:  23.4 on 1 and 102 DF,  p-value: 4.681e-06
```

```
plot(model1)
```




Normal Q−Q
Standardized residuals
4
3
2
1
0
−1
88
92
71
−2
−1
0
1
2
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening))


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
15
16
17
18
19
20
92
88
71
Fitted values
lm(log(‘Box Office‘) ~ log(Opening))



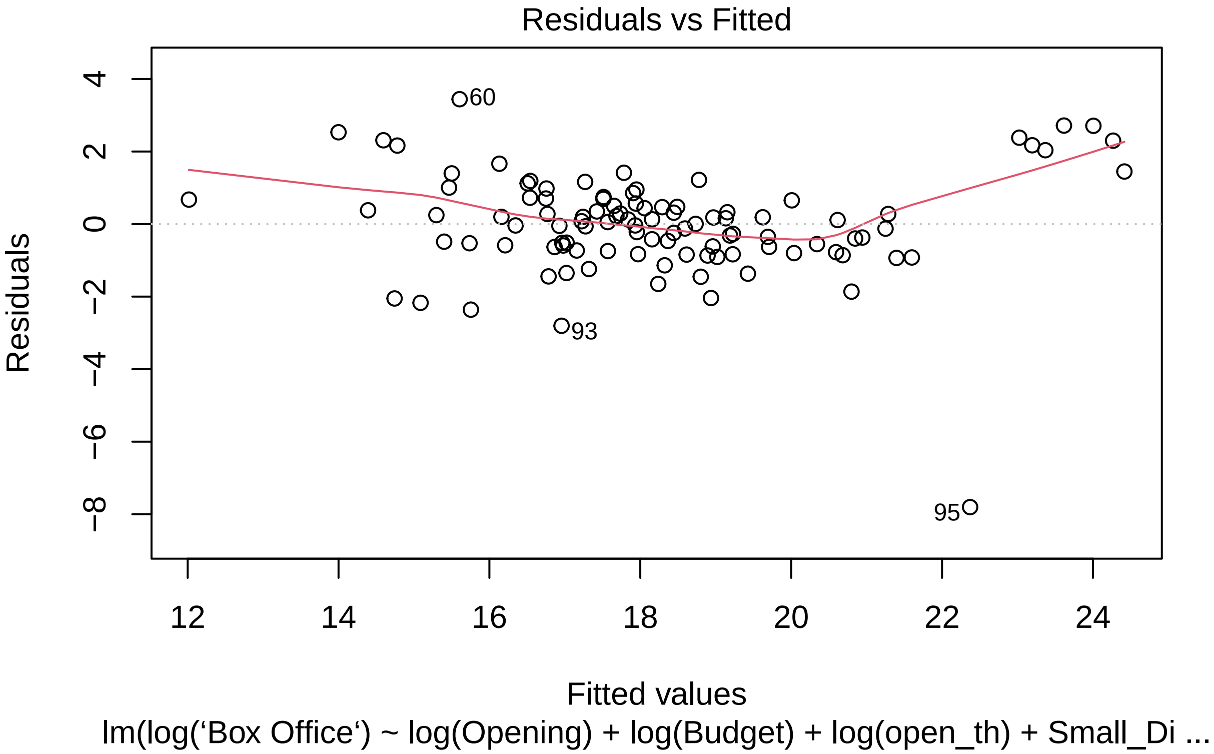
Log trasnformation for all continous variables.

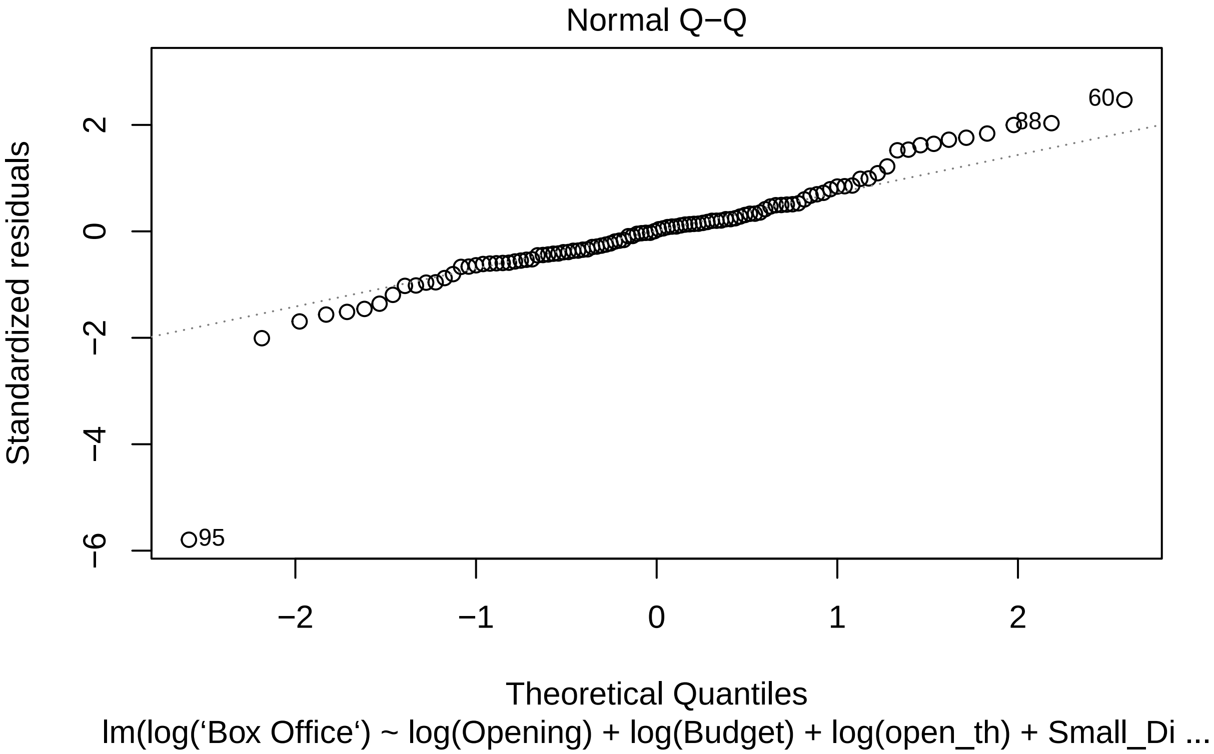
```
model2 <- lm(log(`Box Office`) ~ log(Opening) + log(Budget) + log(open_th) + Small_Dist + season, data =
summary(model2)
```

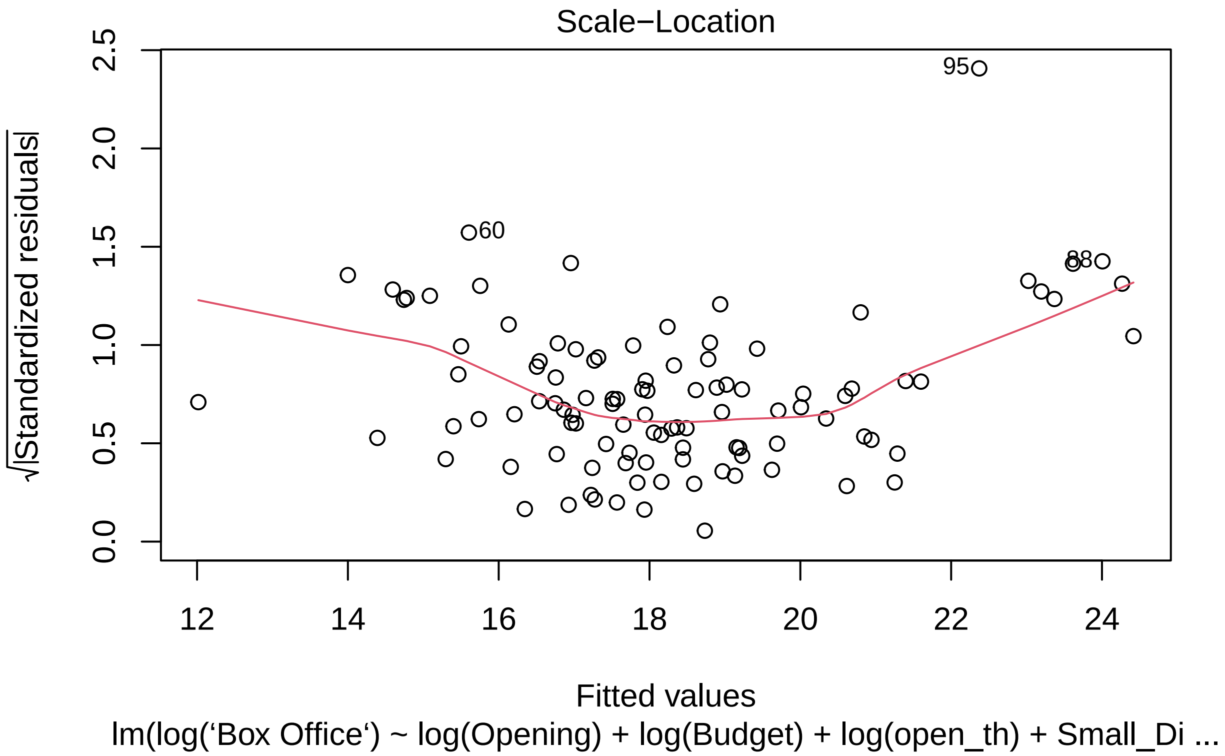
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening) + log(Budget) +
##     log(open_th) + Small_Dist + season, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8042 -0.6565  0.0295  0.6794  3.4428
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.96744    1.51812  -1.296  0.19809
## log(Opening)  0.63119    0.13312   4.741 7.36e-06 ***
## log(Budget)   0.71024    0.05637  12.598  < 2e-16 ***
## log(open_th) -0.33688    0.12115  -2.781  0.00653 **
## Small_Dist    0.79521    0.37723   2.108  0.03763 *
## seasonSpring  0.80434    0.40511   1.985  0.04994 *
## seasonSummer  0.04825    0.37890   0.127  0.89894
## seasonWinter  0.08596    0.41005   0.210  0.83440
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

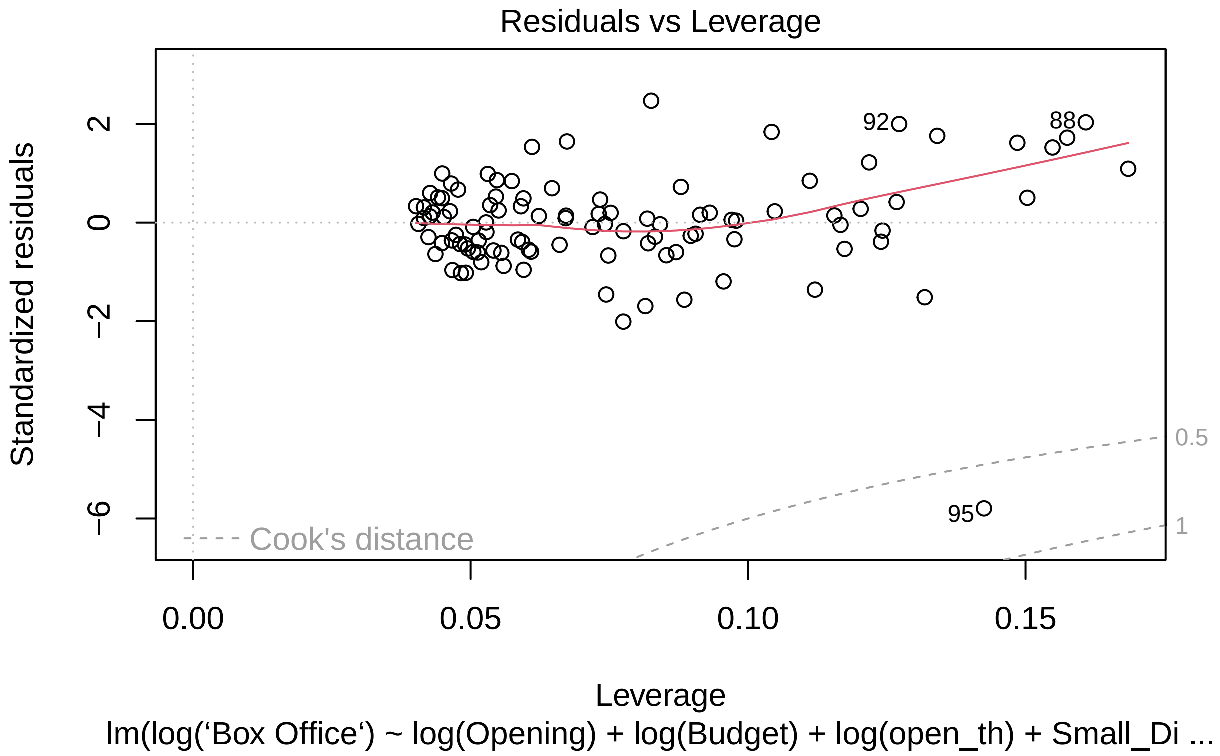
```
## Residual standard error: 1.454 on 96 degrees of freedom
## Multiple R-squared:  0.733,  Adjusted R-squared:  0.7136
## F-statistic: 37.66 on 7 and 96 DF,  p-value: < 2.2e-16
```

```
plot(model2)
```




Normal Q−Q
Standardized residuals
2
0
−2
−4
−6
−2
−1
0
1
2
60
88
95
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + log(open_th) + Small_Di ...


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + log(open_th) + Small_Di ...
95
60
88



The R-square has significantly improved for this model. Let us try more models.

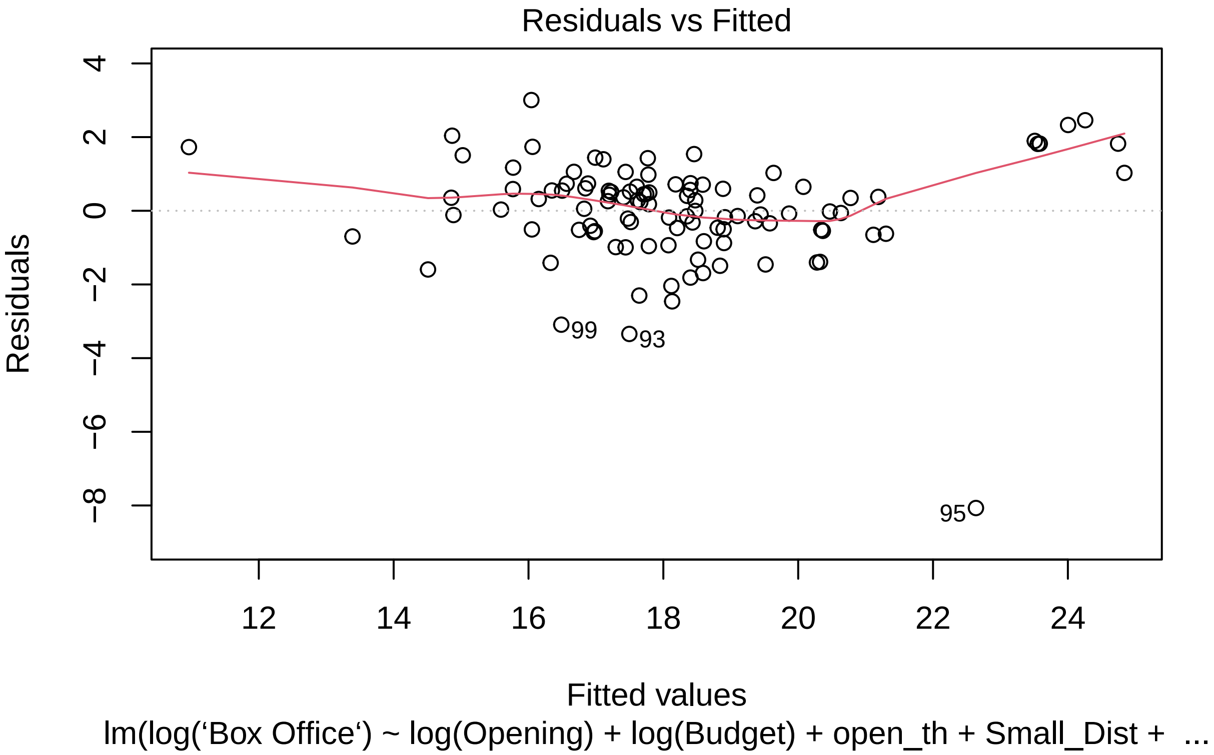
Log transformation for only Box Office, Opening and Budget.

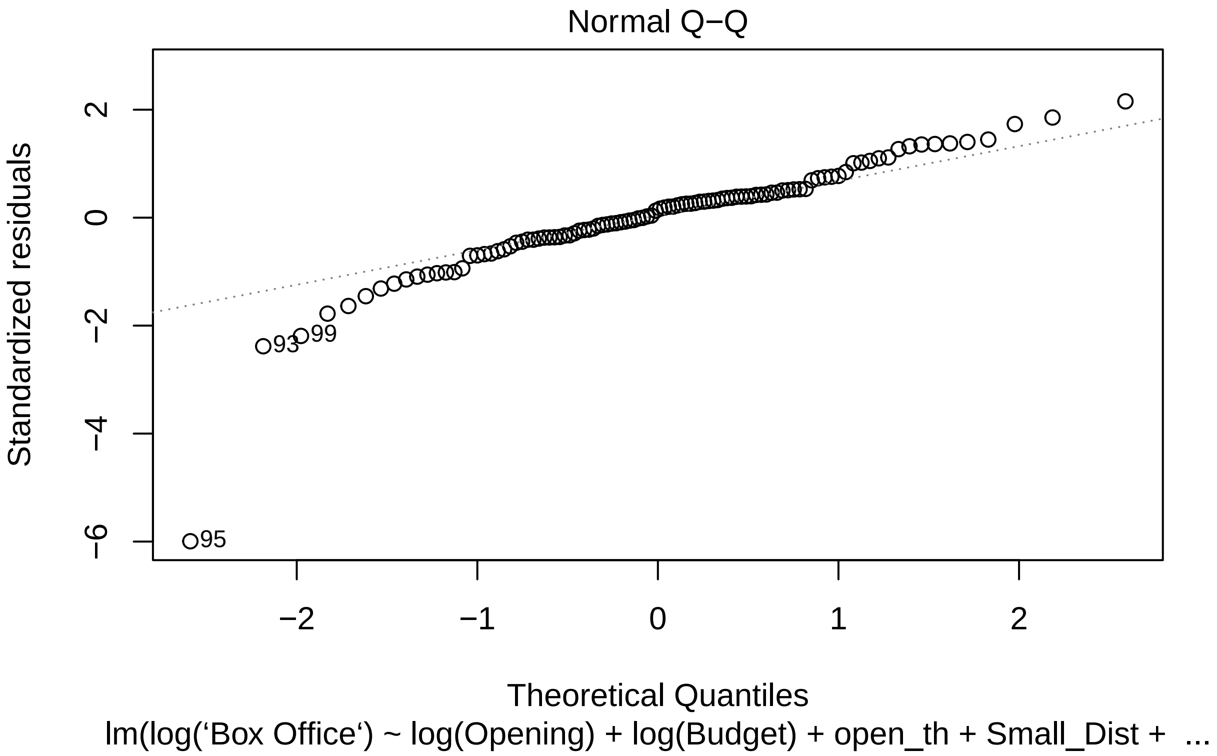
```
model3 <- lm(log(`Box Office`) ~ log(Opening) + log(Budget) + open_th + Small_Dist + season, data = df)
summary(model3)
```

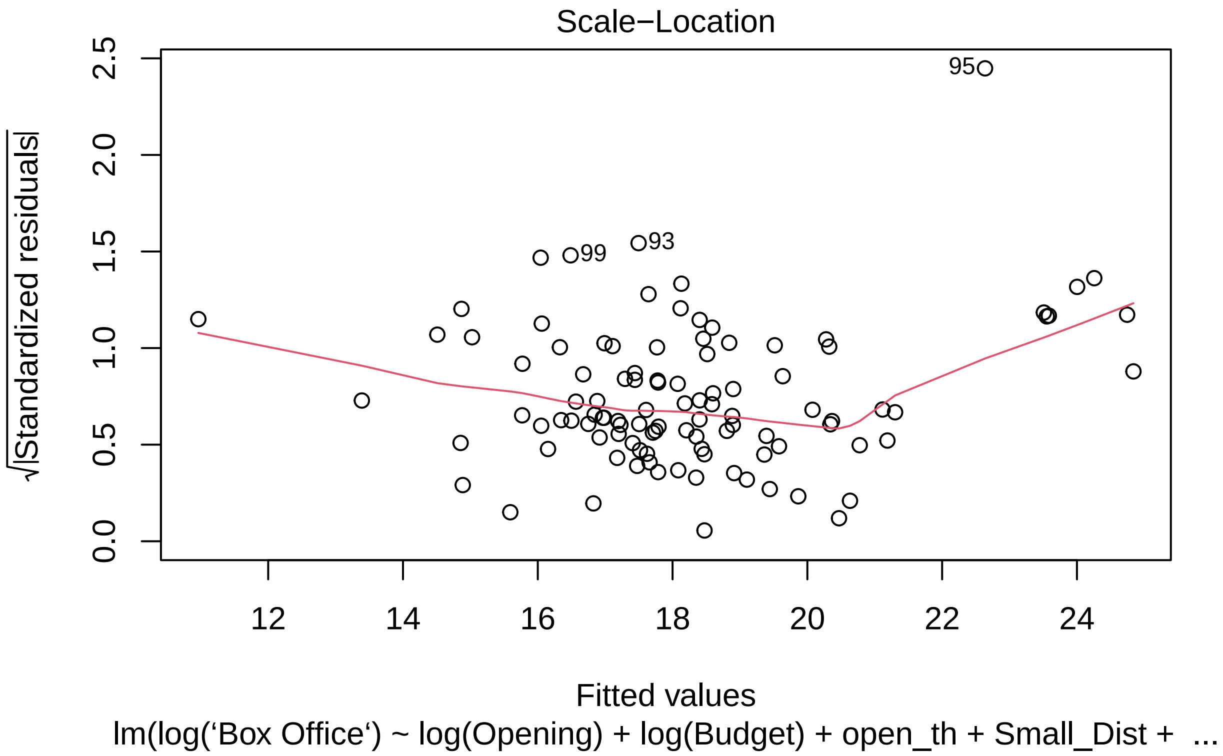
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening) + log(Budget) +
##     open_th + Small_Dist + season, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0685 -0.5475  0.2075  0.6655  3.0050
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.8452903  1.9146324  -2.008  0.04741 *
## log(Opening)  0.7917073  0.1779766   4.448 2.33e-05 ***
## log(Budget)   0.6418791  0.0628810  10.208  < 2e-16 ***
## open_th      -0.0006799  0.0002342  -2.903  0.00459 **
## Small_Dist    0.6493099  0.3721462   1.745  0.08422 .
## seasonSpring  0.8799495  0.4041060   2.178  0.03189 *
## seasonSummer  0.0045385  0.3772137   0.012  0.99043
## seasonWinter  0.2707220  0.4142579   0.654  0.51499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

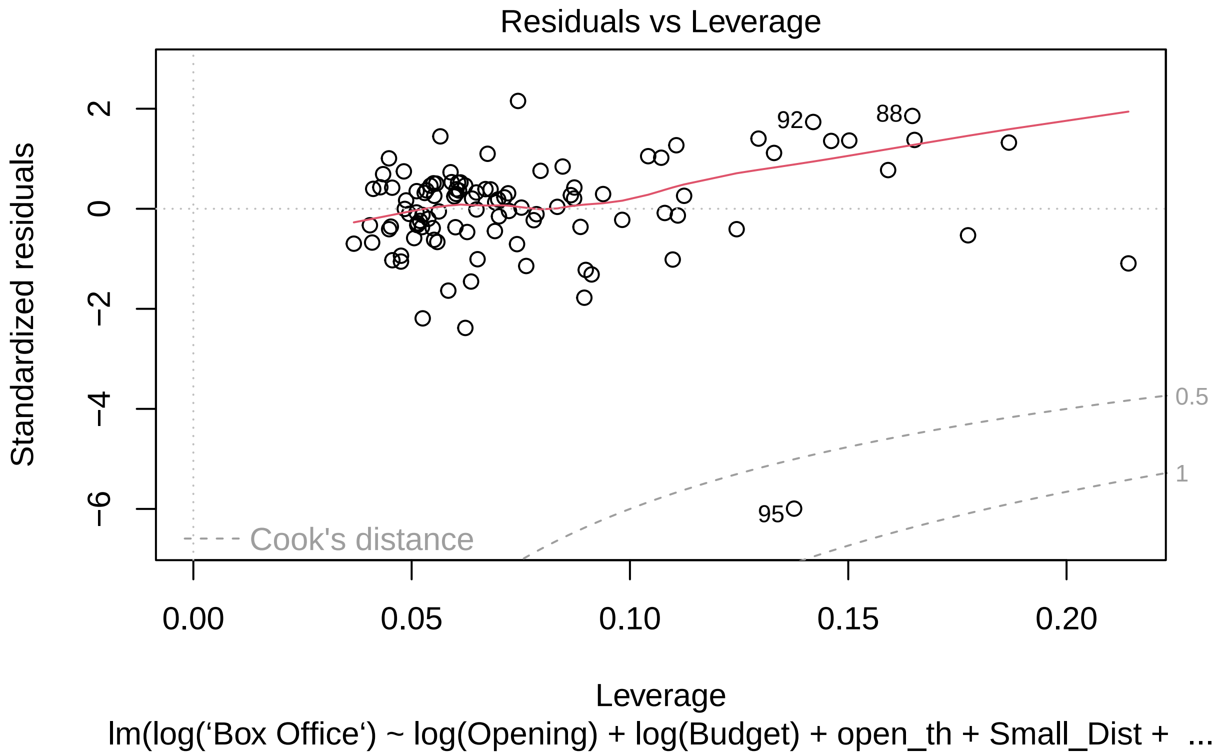
```
##
## Residual standard error: 1.45 on 96 degrees of freedom
## Multiple R-squared:  0.7348, Adjusted R-squared:  0.7155
## F-statistic:    38 on 7 and 96 DF,  p-value: < 2.2e-16
```

```
plot(model3)
```




Normal Q−Q
Standardized residuals
2
0
−2
−4
−6
93
99
95
−2
−1
0
1
2
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th + Small_Dist + ...


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
20
22
24
95
99
93
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th + Small_Dist + ...

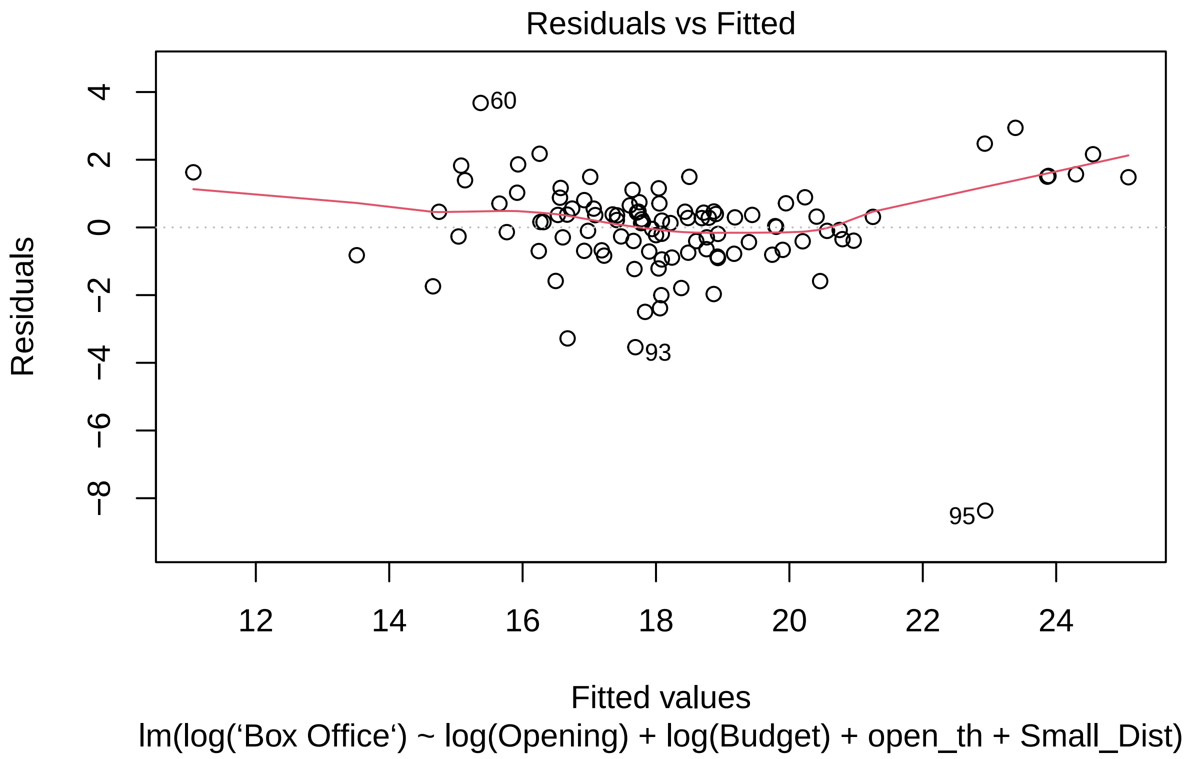


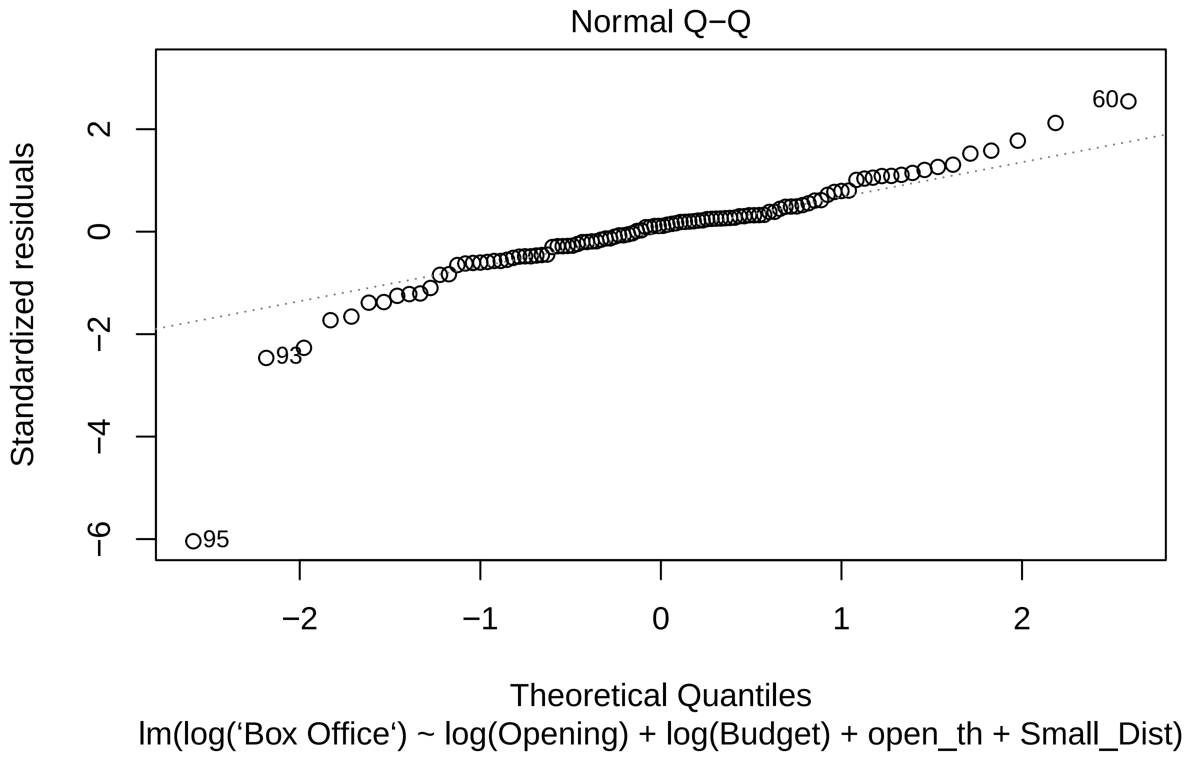
Log transformation for only Box Office, Opening and Budget. Removing season confounder.

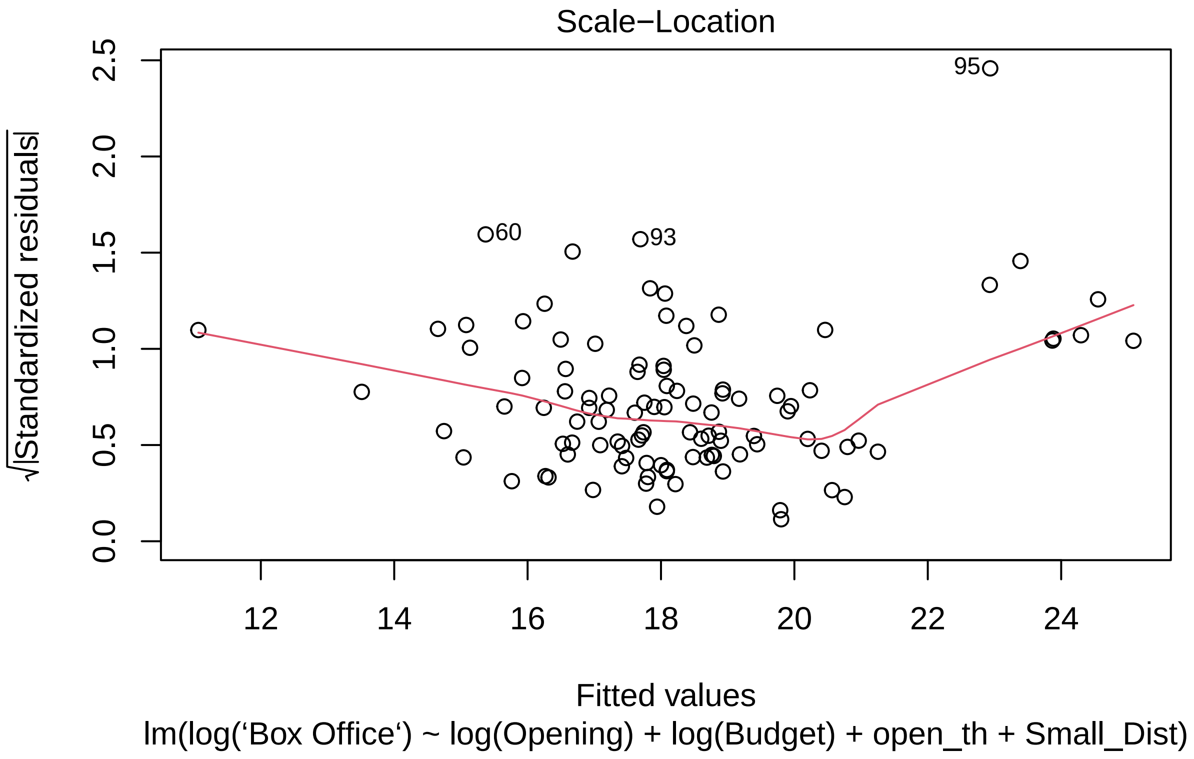
```
model4 <- lm(log(`Box Office`) ~ log(Opening) + log(Budget) + open_th + Small_Dist, data = df)
summary(model4)
```

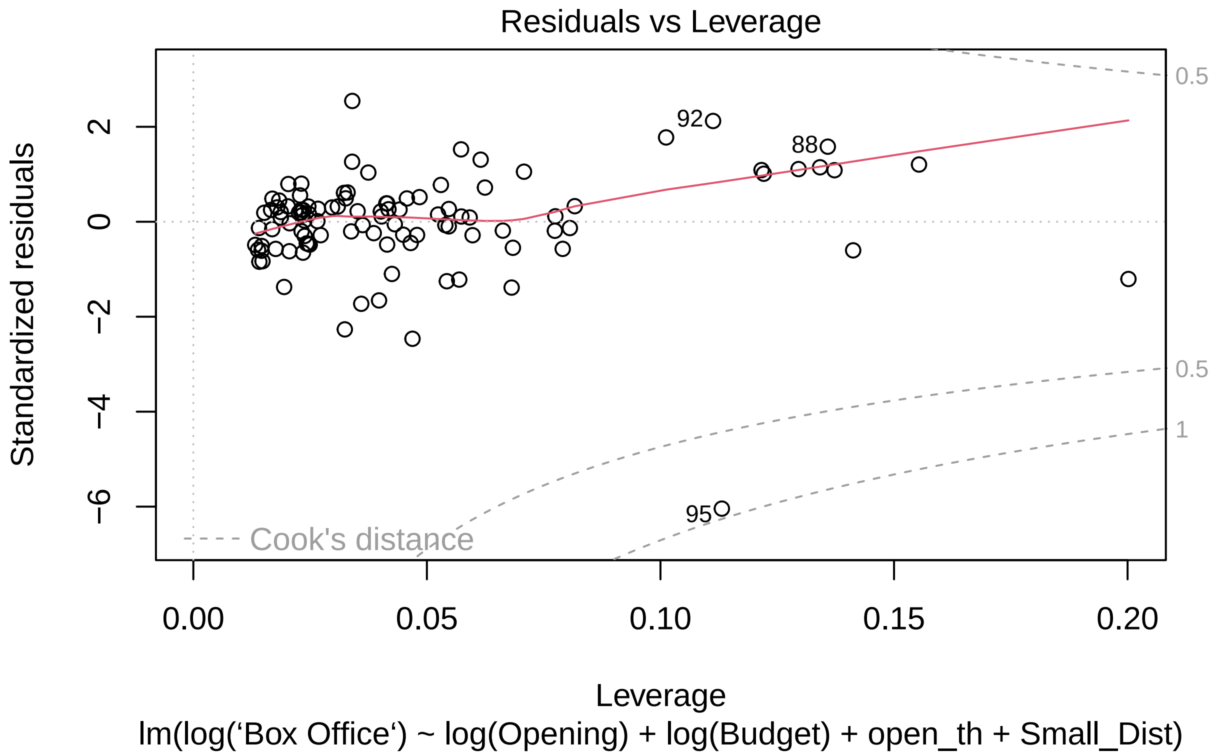
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening) + log(Budget) +
##     open_th + Small_Dist, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3680 -0.6661  0.1612  0.6639  3.6768
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.9070368  1.8994592  -2.057   0.0423 *
## log(Opening)  0.7900377  0.1783030   4.431 2.43e-05 ***
## log(Budget)   0.6570662  0.0634366  10.358  < 2e-16 ***
## open_th      -0.0006501  0.0002338  -2.780   0.0065 **
## Small_Dist    0.6342201  0.3761470   1.686   0.0949 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.471 on 99 degrees of freedom
## Multiple R-squared:  0.7185, Adjusted R-squared:  0.7071
## F-statistic: 63.17 on 4 and 99 DF,  p-value: < 2.2e-16
```

```
plot(model4)
```


Residuals vs Fitted
Residuals
4
2
0
−2
−4
−6
−8
60
93
95
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th + Small_Dist)


Normal Q−Q
Standardized residuals
2
0
−2
−4
−6
−2
−1
0
1
2
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th + Small_Dist)
60
93
95


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th + Small_Dist)
60
93
95

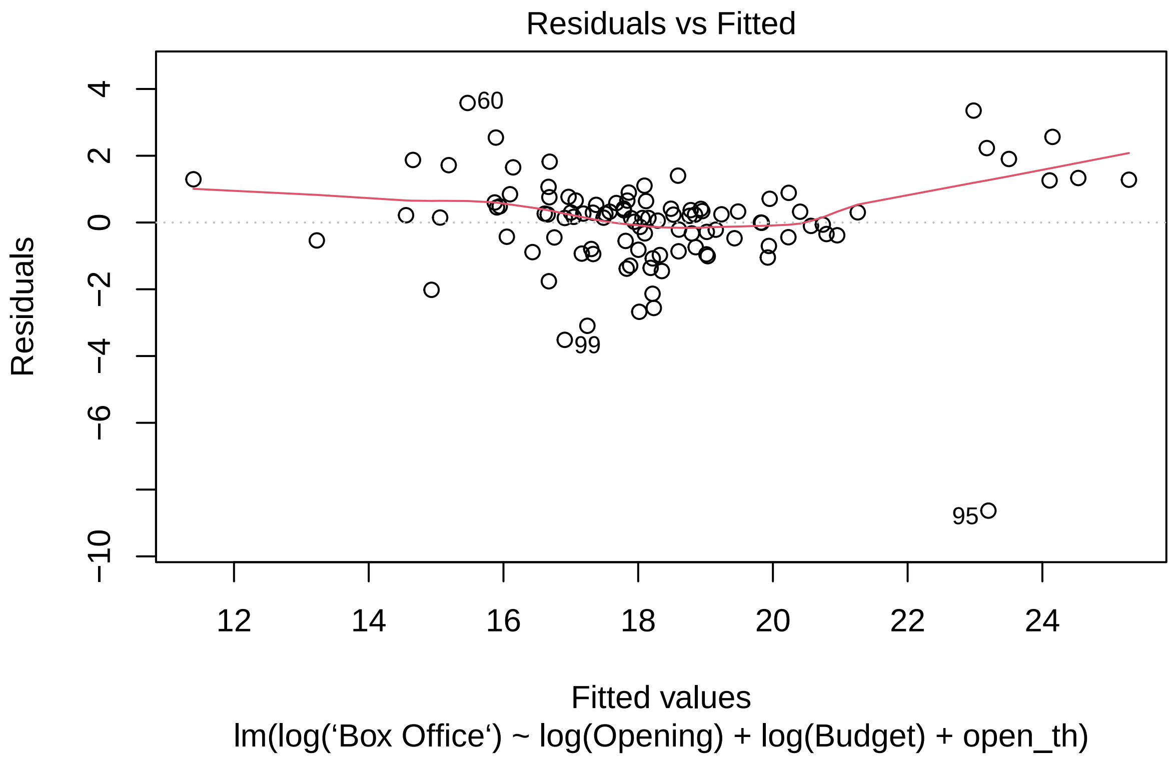


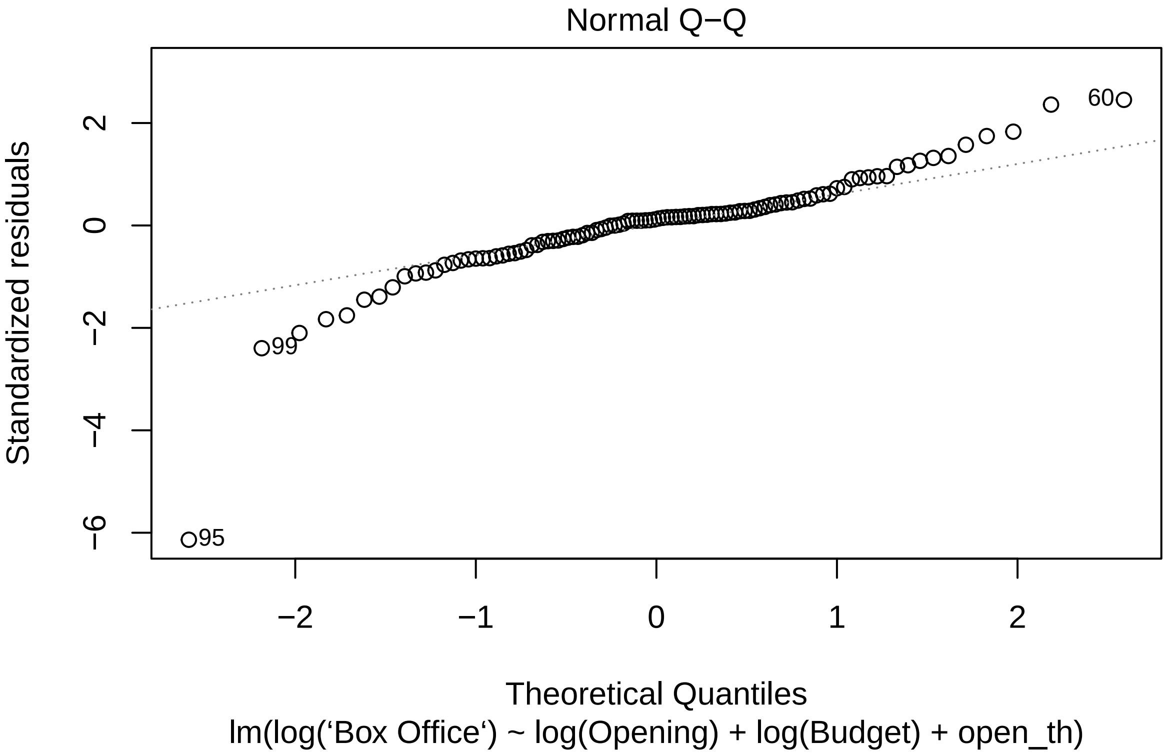
R square values are further below the previous model. Let us try the log transformed model with only the Budget and Small_Distributor as confounders.

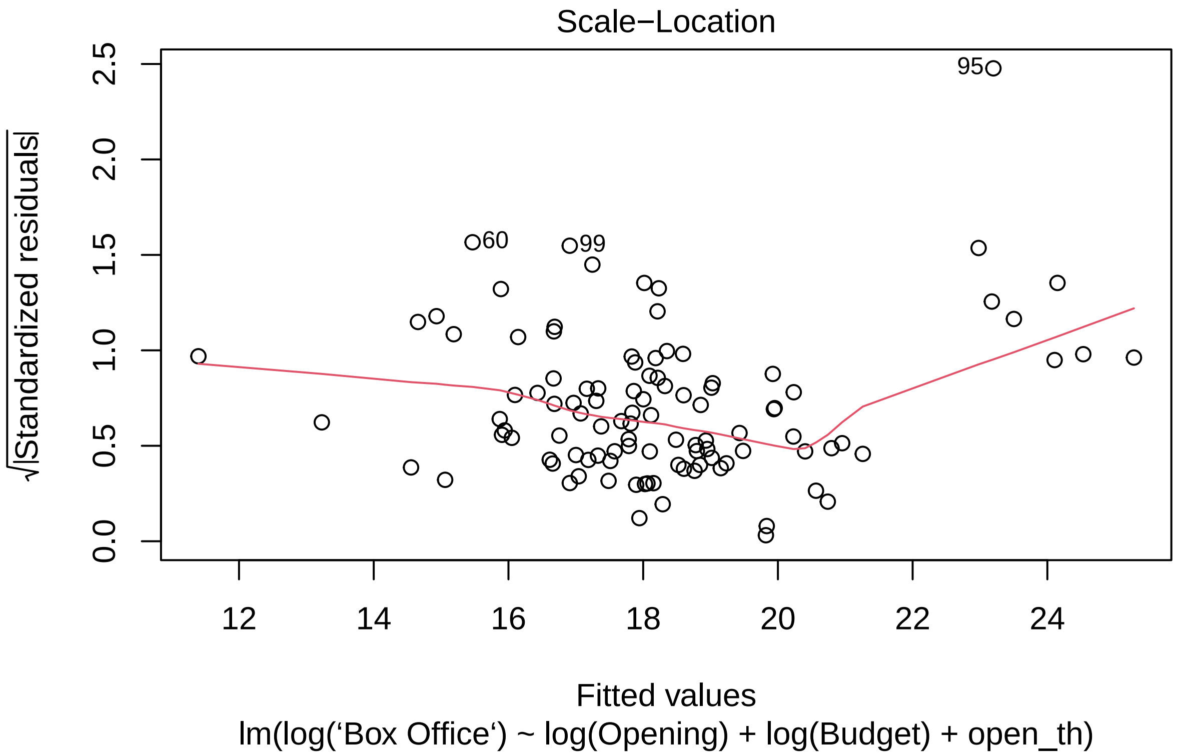
```
model5 <- lm(log(`Box Office`) ~ log(Opening) + log(Budget) + open_th, data = df)
summary(model5)
```

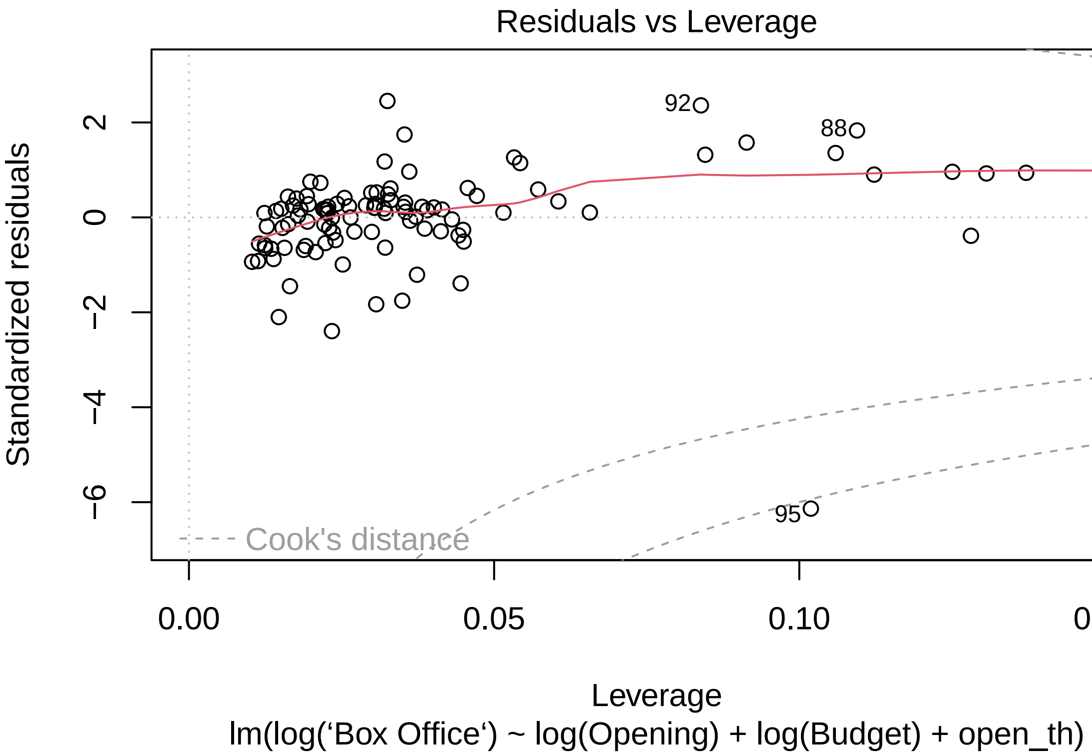
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening) + log(Budget) +
##     open_th, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6310 -0.5408  0.1846  0.6106  3.5804
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.3075637  1.8830028  -1.757   0.0821 .
## log(Opening)  0.7496919  0.1783107   4.204 5.71e-05 ***
## log(Budget)   0.6654124  0.0638233  10.426  < 2e-16 ***
## open_th      -0.0006509  0.0002360  -2.759   0.0069 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.484 on 100 degrees of freedom
## Multiple R-squared:  0.7104, Adjusted R-squared:  0.7017
## F-statistic: 81.78 on 3 and 100 DF,  p-value: < 2.2e-16
```

```
plot(model5)
```


Residuals vs Fitted
Residuals
4
2
0
−2
−4
−6
−10
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th)
60
99
95


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th)
2
0
−2
−4
−6
−2
−1
0
1
2
60
99
95


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget) + open_th)
95
60
99

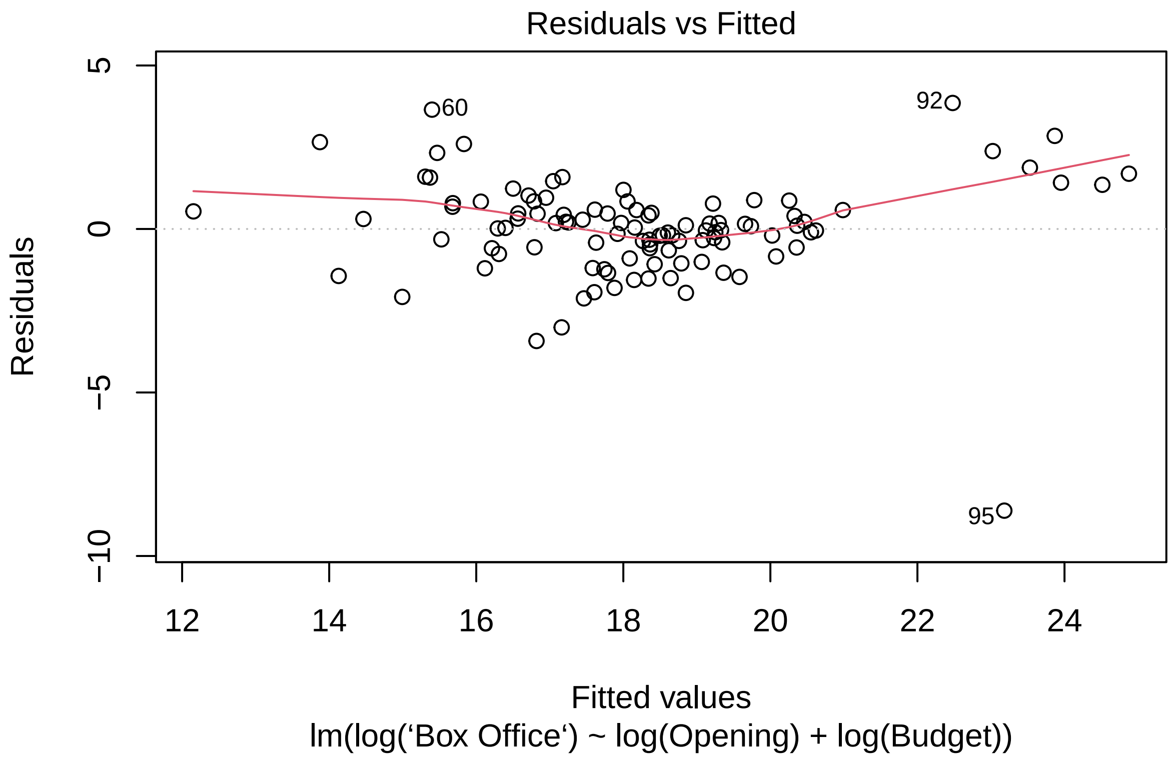


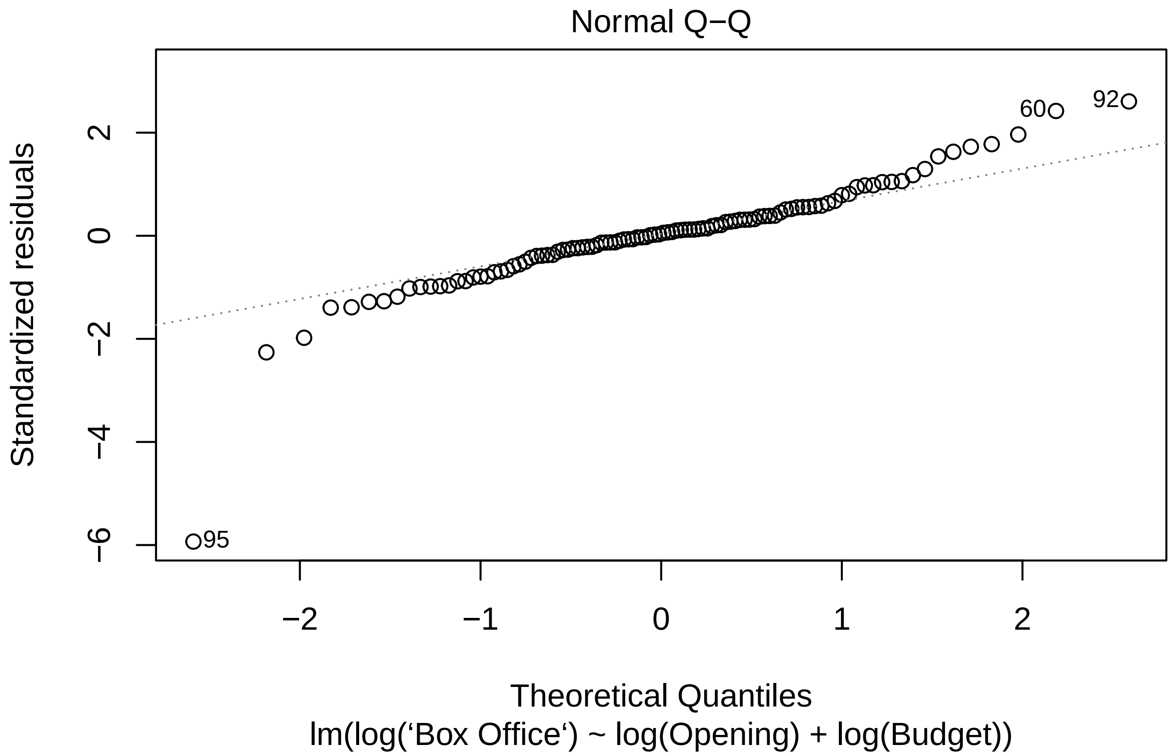
Now, we can run a log transformed model with only the Opening and Budget.

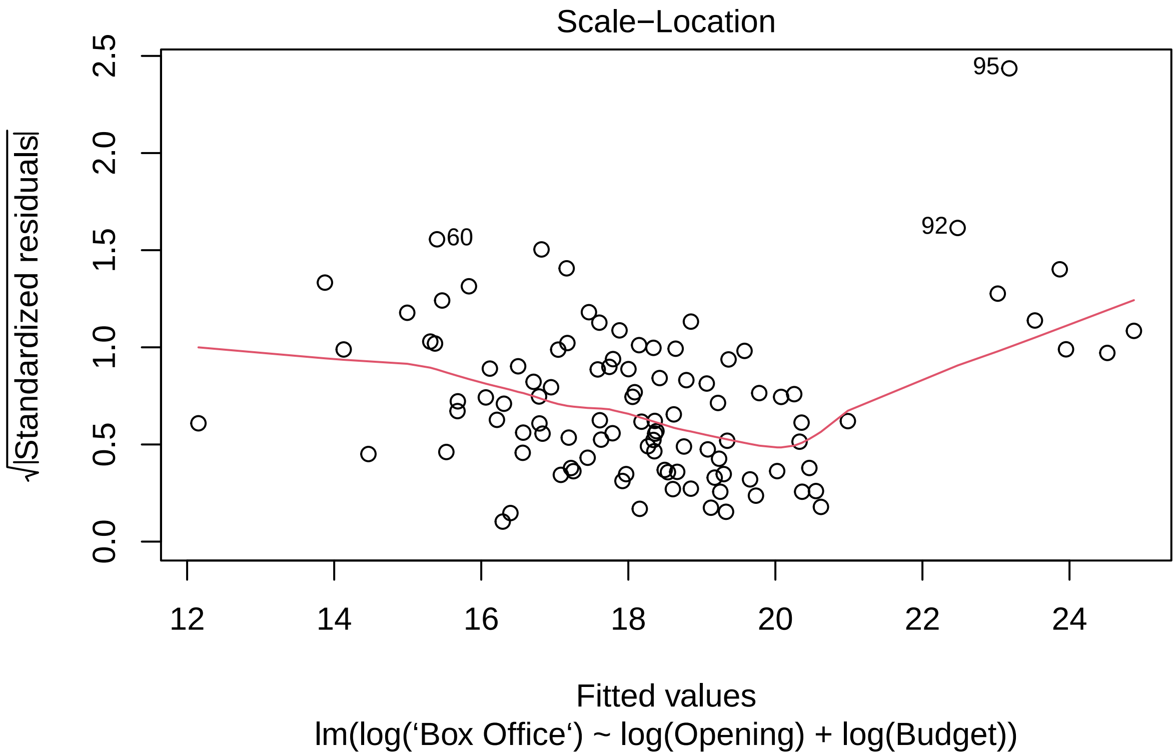
```
model6 <- lm(log(`Box Office`) ~ log(Opening) + log(Budget), data = df)
summary(model6)
```

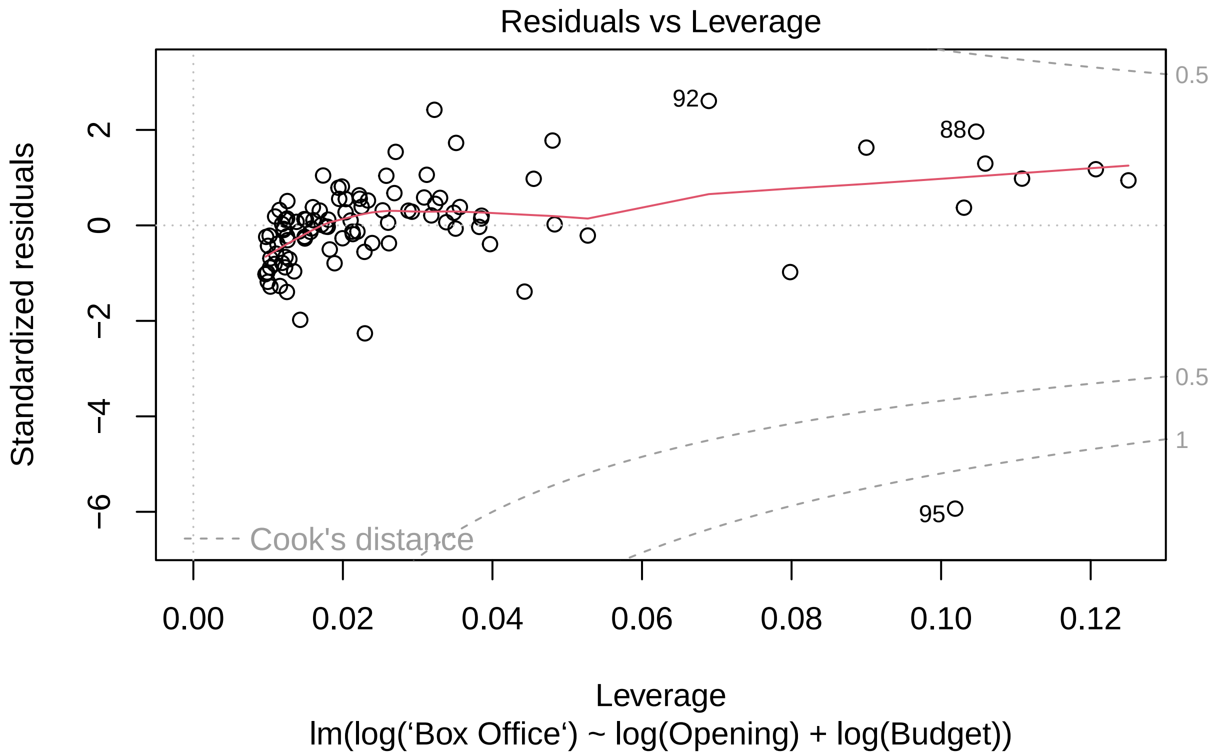
```
##
## Call:
## lm(formula = log(`Box Office`) ~ log(Opening) + log(Budget),
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6137 -0.5880  0.0639  0.7038  3.8528
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.60584    1.27815   0.474    0.637
## log(Opening)  0.29357    0.06891   4.260 4.59e-05 ***
## log(Budget)   0.74629    0.05852  12.753  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.532 on 101 degrees of freedom
## Multiple R-squared:  0.6884, Adjusted R-squared:  0.6822
## F-statistic: 111.6 on 2 and 101 DF,  p-value: < 2.2e-16
```

```
plot(model6)
```


Residuals vs Fitted
Residuals
5
0
−5
−10
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget))
60
92
95


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget))
2
0
−2
−4
−6
−2
−1
0
1
2
60
92
95


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
20
22
24
Fitted values
lm(log(‘Box Office‘) ~ log(Opening) + log(Budget))
95
92
60



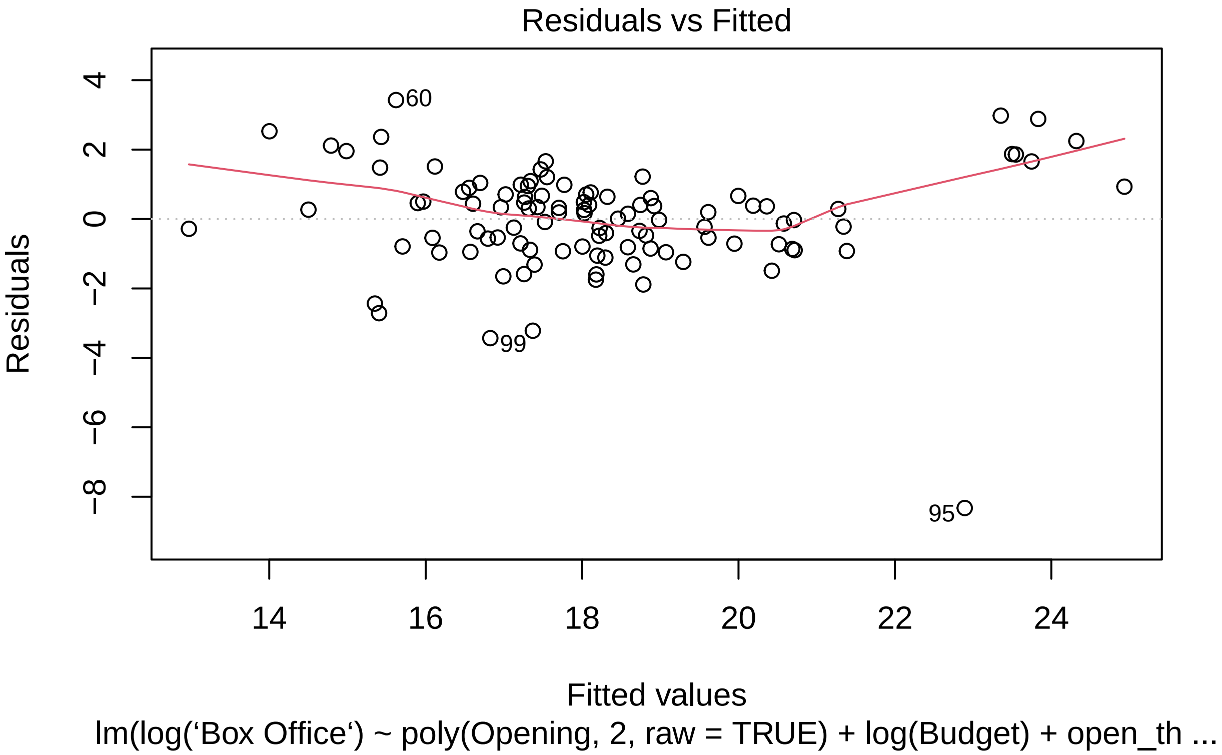
Let us run models after quadratic transformation

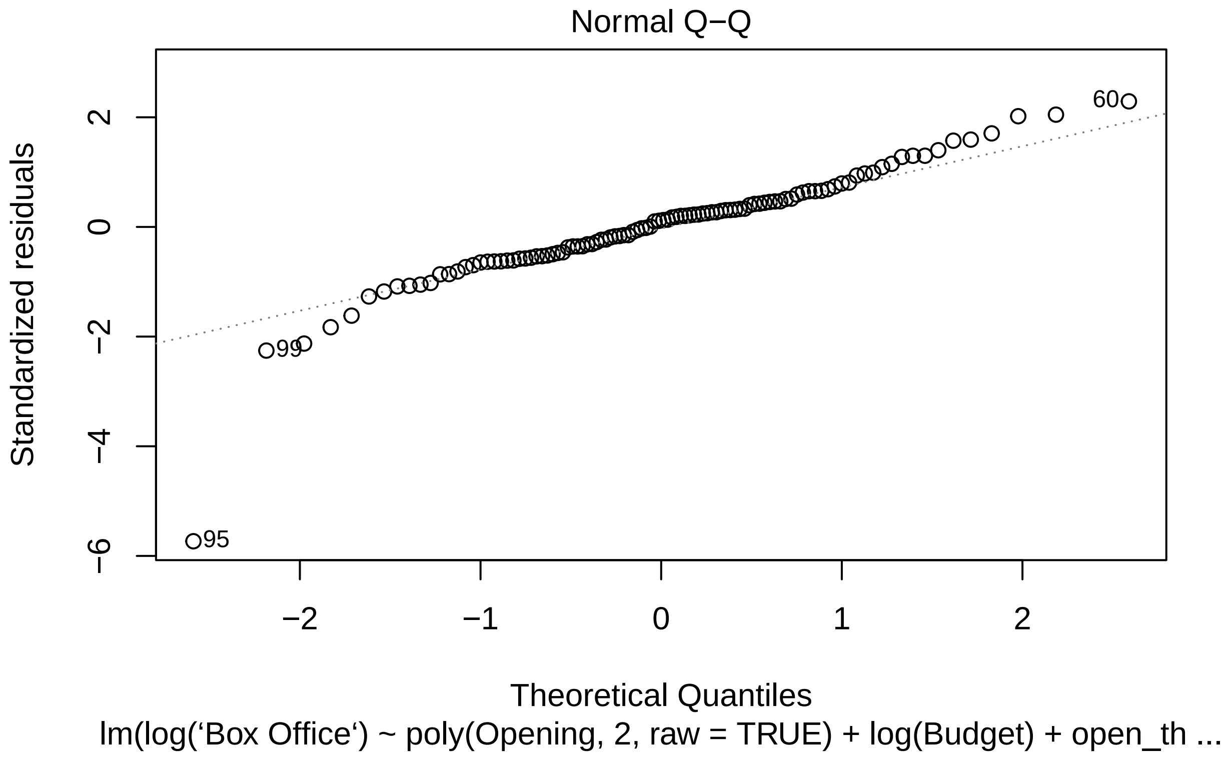
```
model_quad1 <- lm(log(`Box Office`) ~ poly(Opening, 2, raw=TRUE) + log(Budget) + open_th + Small_Dist +
summary(model_quad1)
```

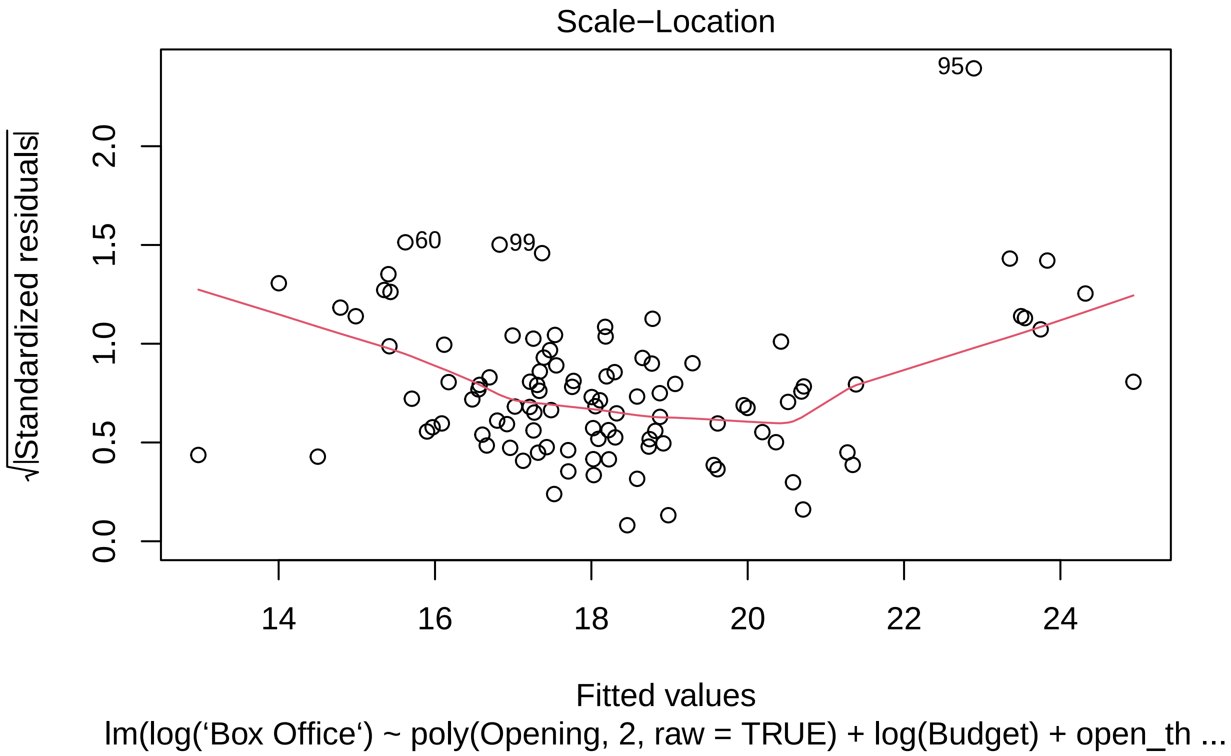
```
##
## Call:
## lm(formula = log(`Box Office`) ~ poly(Opening, 2, raw = TRUE) +
##     log(Budget) + open_th + Small_Dist + season, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3245 -0.7981  0.1786  0.7227  3.4267
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    4.037e+00  1.116e+00   3.616 0.000481 ***
## poly(Opening, 2, raw = TRUE)1  3.202e-08  1.865e-08   1.717 0.089184 .
## poly(Opening, 2, raw = TRUE)2 -1.395e-16  1.062e-16  -1.314 0.192175
## log(Budget)                    7.584e-01  5.956e-02  12.734  < 2e-16 ***
## open_th                        5.997e-05  1.444e-04   0.415 0.678821
## Small_Dist                     4.668e-01  3.979e-01   1.173 0.243623
## seasonSpring                   7.997e-01  4.373e-01   1.829 0.070557 .
## seasonSummer                  -1.038e-01  4.108e-01  -0.253 0.801060
## seasonWinter                   7.185e-02  4.434e-01   0.162 0.871604
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

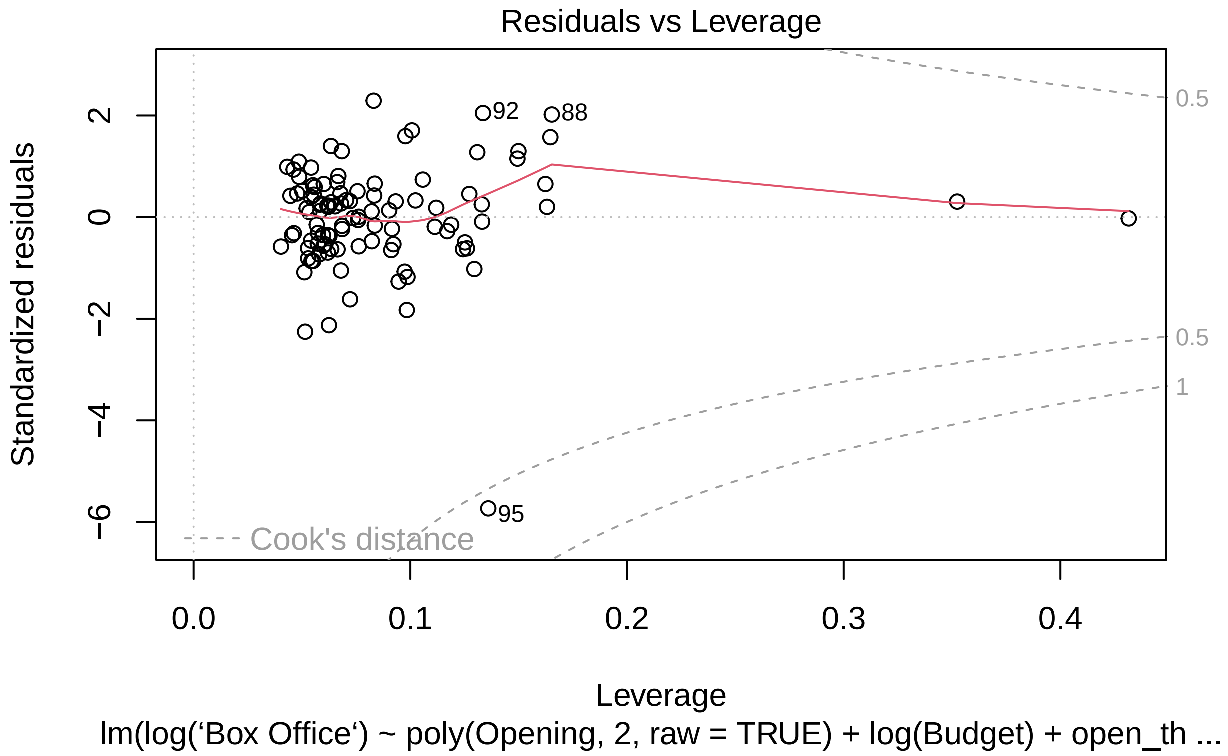
```
##
## Residual standard error: 1.562 on 95 degrees of freedom
## Multiple R-squared:  0.6952, Adjusted R-squared:  0.6695
## F-statistic: 27.08 on 8 and 95 DF,  p-value: < 2.2e-16
```

```
plot(model_quad1)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(‘Box Office‘) ~ poly(Opening, 2, raw = TRUE) + log(Budget) + open_th ...
2
0
−2
−4
−6
−2
−1
0
1
2
60
99
95


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
14
16
18
20
22
24
60
99
95
Fitted values
lm(log(‘Box Office‘) ~ poly(Opening, 2, raw = TRUE) + log(Budget) + open_th ...



## Selecting a model and Interpretation of the result

Out of all the models we ran, model3 looks as the best fit.

The linear regression model predicts the logarithm of Box Office collections based on the logarithm of Opening weekend collections, logarithm of Budget, number of theaters the movie opened in, Small_Dist (a binary variable indicating whether the distributor is small or not), and the season in which the movie was released (Spring, Summer, or Winter).

The coefficients of the independent variables show the direction and magnitude of their effect on the dependent variable. The p-value associated with each coefficient indicates whether the coefficient is statistically significant or not.

The intercept coefficient is -3.845, which means that if all independent variables are zero, the model predicts that the logarithm of Box Office collections is -3.845. However, since all the independent variables are not zero in practice, this value is not meaningful.

The coefficient of the logarithm of Opening weekend collections is 0.792, which means that a one percent increase in Opening weekend collections is associated with a 0.792 percent increase in Box Office collections.

The coefficient of the logarithm of Budget is 0.642, which means that a one percent increase in Budget is associated with a 0.642 percent increase in Box Office collections.

The coefficient of the number of theaters the movie opened in (open_th) is negative (-0.00068), which means that as the number of theaters increases by one, the predicted logarithm of Box Office collections decreases by 0.00068. This is something to be studied further, as it goes against our logic and expectation.

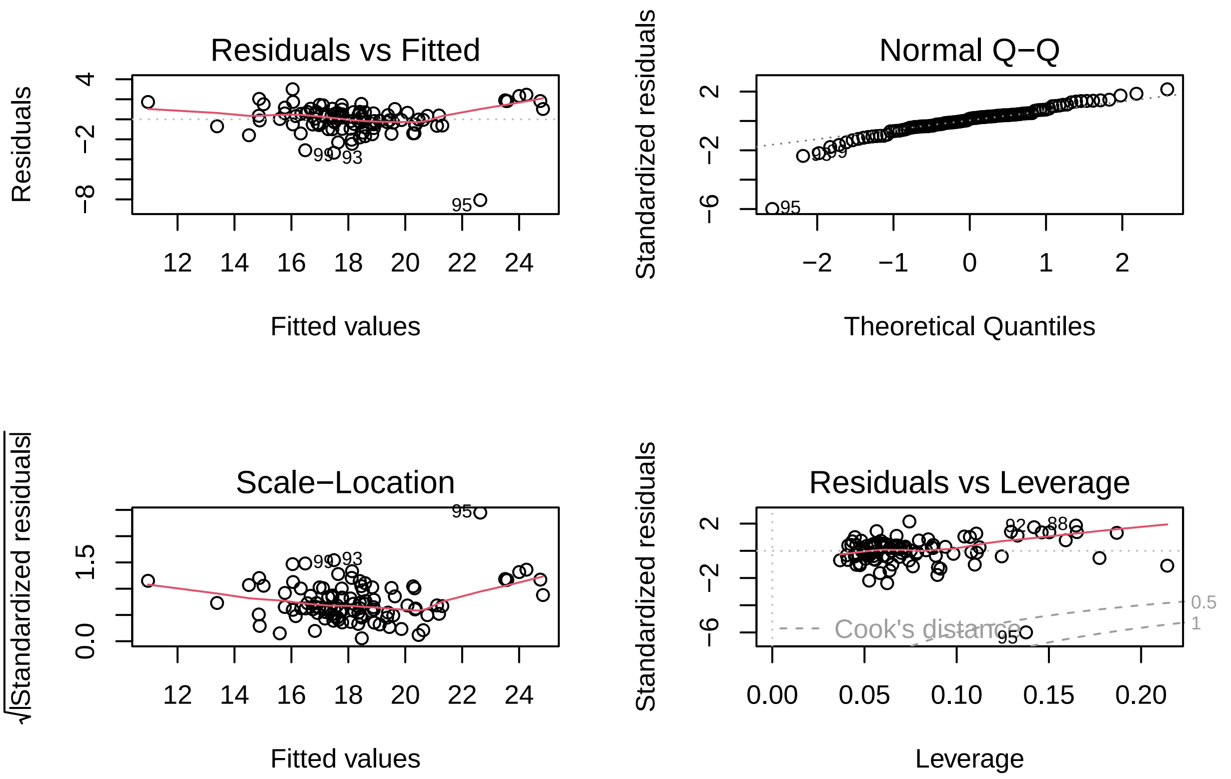
The coefficient of Small_Dist is 0.649, which means that the Box Office collections of movies released by small distributors are 0.649 times higher than those of movies released by large distributors, holding other variables constant. This also need more detailed study.

The coefficients associated with season indicate the difference in Box Office collections between movies released in that season and movies released in Fall (omitted reference level). For example, the coefficient of season Spring is 0.88, which means that the predicted Box Office collections of movies released in Spring are 0.88 times higher than those of movies released in Fall.

The adjusted R-squared of the model is 0.7155, which means that 71.55% of the variation in the logarithm of Box Office collections can be explained by the independent variables in the model.

### Visualising the regression model.

```r
# Diagnostic plots
par(mfrow=c(2,2))
plot(model3)
```



### Conclusion:

The model that we build shows that the Box Office collection of a movie is affected positively by the Opening week collection. 1 percentage of increase in Opening week collection results in 0.792 percent increase in Box office collection after takeing into account the intercept(-3.845).

The dataset has only 200 rows out of which 96 rows were dropped as they have NA values. Also, many movies had Box office collections much lesser than their budget. But that does not necessarily mean that those movies made a loss. In the age of online streaming platforms, many movies are making money not from Box Office only. But our study was focused only on the revenue from theatre collection.

## References:

1. Nasir, Suphan & Öcal, Figen. (2016). Film Marketing: The Impact of Publicity Activities on Demand Generation. 10.4018/978-1-5225-0143-5.ch019.

2. *Elizabeth Cooper-Martin (1991) ,"Consumers and Movies: Some Findings on Experiential Products", in NA - Advances in Consumer Research Volume 18, eds. Rebecca H. Holman and Michael R. Solomon, Provo, UT : Association for Consumer Research, Pages: 372-378.*

3. Kaggle.com (Original dataset)

4. Wikipedia.com (Movie pages to get budget and box office collections)