Movies personal project

Ian

2022-11-02

Looking at IMDb Movies and Genres dataset

Being a movie junkie, I have a personal interest in exploring this dataset. Some possible points of interests can be

- Big directors and their genres
- Have Ratings changed over time
- Is there an interaction between main and side genre that gives higher Ratings
- Censorship over time
- Relationships between actors and directors

Potential modelling

- Can we predict ratings given variables
  - Need to first check for relationships

Let us do a cursory check on the data

```
kable(sapply(movies, typeof)) %>% kable_styling()
```

We have a few columns: Movie_Title, Year, Director, Actors, Rating, Runtime.Mins., Censor, Total_Gross, main_genre, side_genre, and we have a total of 5562 rows of data (i.e. 5562 movies)

We can see that something looks off - for example, total_gross should not be a character, lets take a look at a sample: Gross Unkown, $534.86M, $377.85M, $292.58M, $342.55M, $315.54M - we can already see that it is probably made into a string due to presence of "Gross Unknown": Gross Unkown, $534.86M, $377.85M, $292.58M, $342.55M, $315.54M, $171.48M, $290.48M, $204.84M, $322.74M - hence we need to clean this up

We may also want to change some columns to factor, such as censor and genres

	x
Movie_Title	character
Year	integer
Director	character
Actors	character
Rating	double
Runtime.Mins.	integer
Censor	character
Total_Gross	character
main_genre	character
side_genre	character

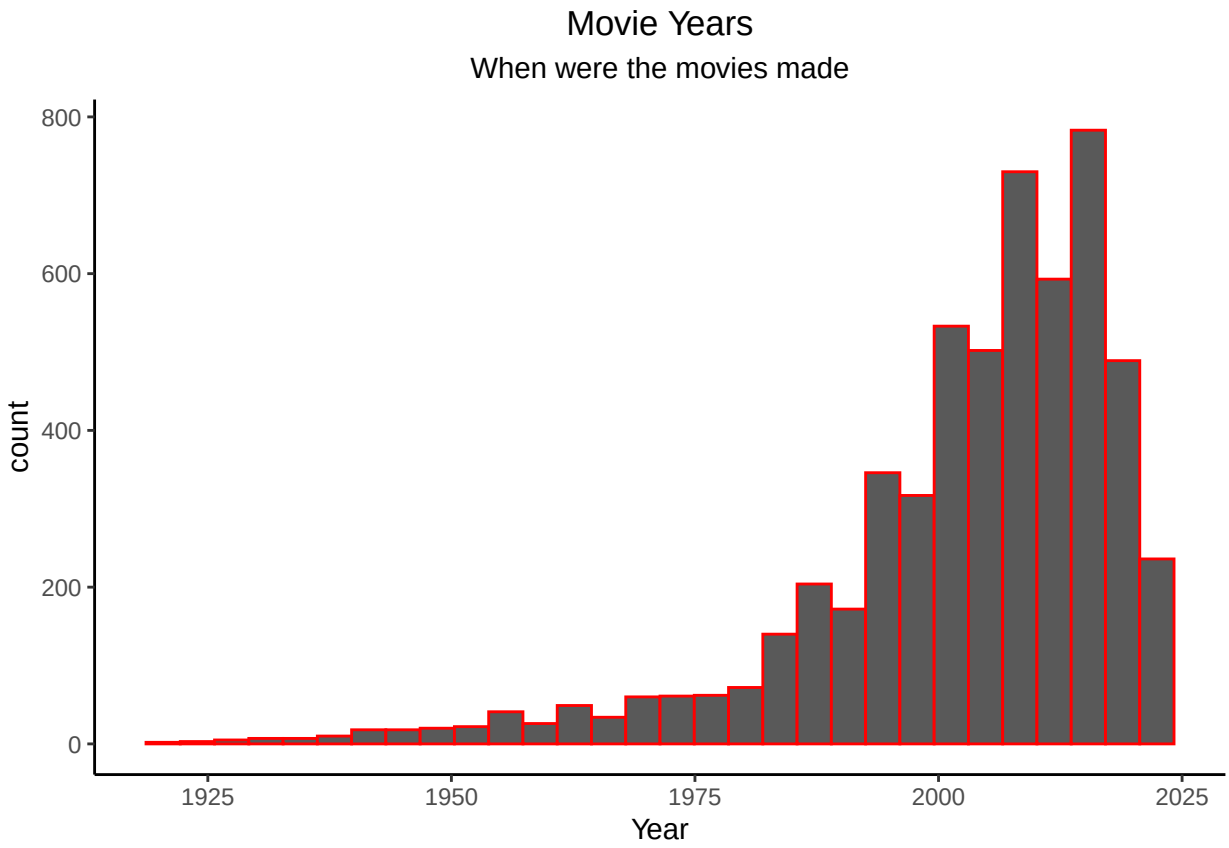
```
# sum(movies$Total_Gross == "Gross Unkown")

actorlist <- unlist(strsplit(movies$Actors, ",")) ## split according to commas, but here we lose granul

## replace "Gross Unkown" with NA
movies[movies$Total_Gross == "Gross Unkown",]$Total_Gross = NA

## remove the "$" and "M" prefix and suffix from the non-problematic entries
library(stringr)
movies[!is.na(movies$Total_Gross),]$Total_Gross <-
  str_sub(movies[!is.na(movies$Total_Gross),]$Total_Gross, 2, -2)

## make numeric
movies$Total_Gross <-as.numeric(movies$Total_Gross)

## set Censorship, Main genre to factor
movies$Censor <- as.factor(movies$Censor)
movies$main_genre <- as.factor(movies$main_genre)
```

Univariate analysis

To explore the data, we can look at some columns of interest one by one

```
movies %>% ggplot(aes(x = Year)) + geom_histogram(col = "red") +
  theme_classic() + ## for a nicer background
  ggtitle(label = "Movie Years", subtitle = "When were the movies made") +
  theme(plot.title = element_text(hjust = .5),
        plot.subtitle = element_text(hjust = .5))
```

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

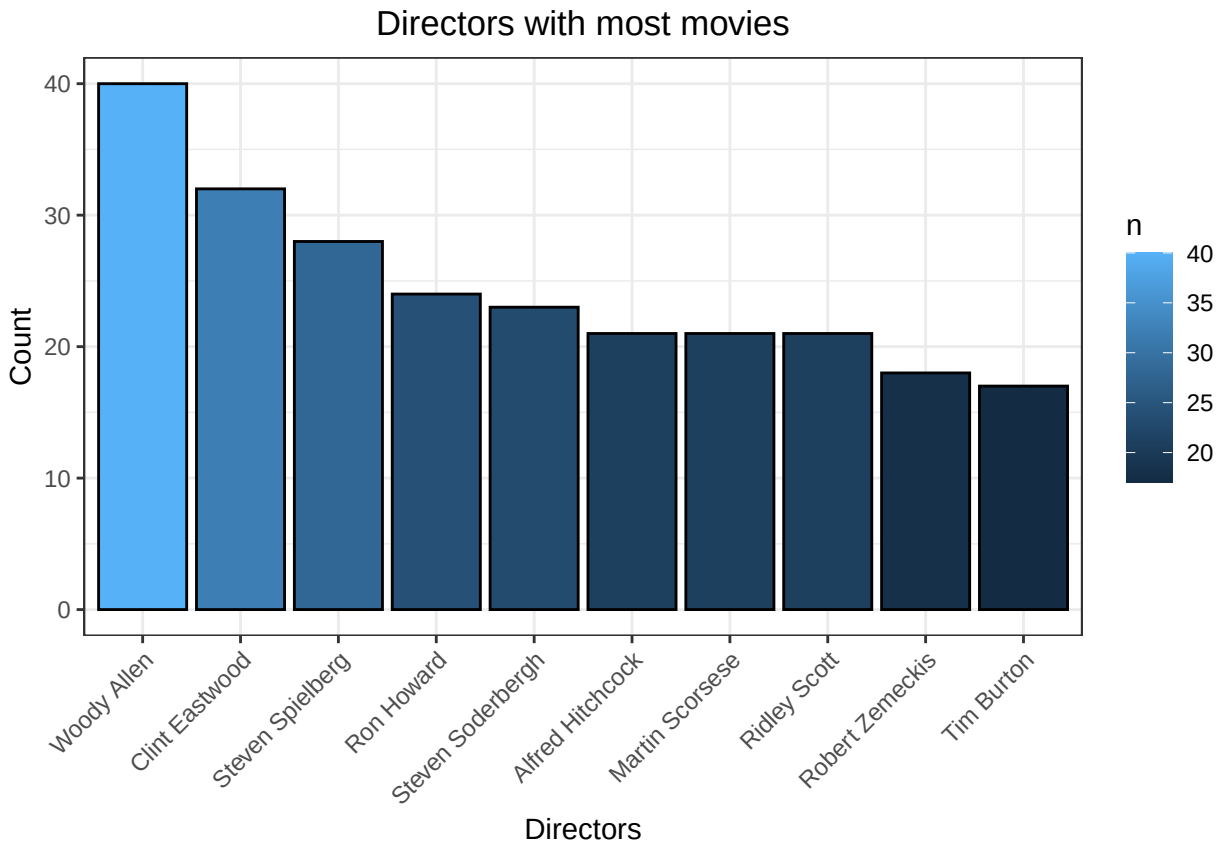


The years for the movies in this dataset are left skewed – there are fewer movies made in the past than closer to the present. - The range of years are 1920, 2022, representing more than 100 years of movies

```
top10directors <- movies %>% select(Director) %>% count(Director) %>% arrange(desc(n)) %>%
  top_n(n = 10)
```

```
## Selecting by n
```

```
## reorder the x in terms of -n in order to arrange bars in descending order
ggplot(top10directors, aes(x = reorder(Director, -n), y = n, fill = n)) +
  theme_bw() +
  geom_bar(stat = "identity", color = "black")+
  labs(title = "Directors with most movies",
       x = "Directors",
       y = "Count")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(hjust = .5))  ## more visible tick markets for x axis
```

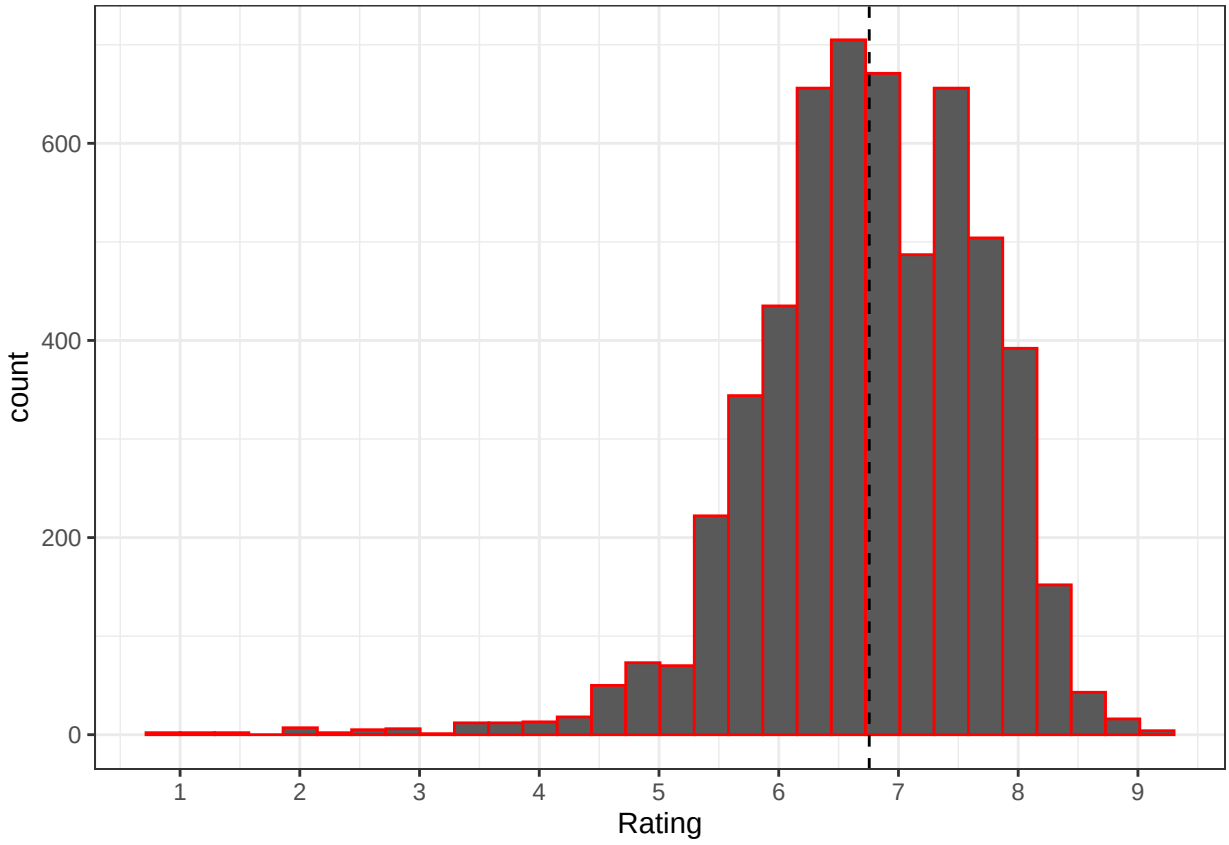



```
typeof(movies$Actors)
is.vector(movies$Actors) ## Actors column is a vector of vectors, i.e. for every observation we have a

actorlist <- unlist(strsplit(movies$Actors, ","))
## split according to commas, but here we lose granularity i.e. which actors were in which movies
## although we know that every 4 actors correspond to one movie

# for (i in (1:nrow(movies))){
#   print(length(unlist(strsplit(movies$Actors[i], ","))))
# }

## Import library for splitting columns
library(splitstackshape)

## Split actors column (list of 4 elements) into four column, each containing one element (one actor)
movies <- cSplit(indt = movies, splitCols = "Actors", sep = ", ")
names(movies) ## check result of the split

## Clean up: need to apply same function across columns, therefore use APPLY functions
#movies[,10:13] <- trimws(movies[,10:13])
#movies[,10] <- trimws(movies[,10])
cleaned <- apply(movies[,10:13], MARGIN = 2 , FUN = trimws)

## get 20 biggest actors in terms of number of movies
top20actorstable <- sort(table(Actors = actorlist), decreasing = T)[1:20]

## one hot encoding -- We should try to create a variable that captures how many top actors (out of 4)
```

```
top20actorsnames <- names(top20actorstable)

movies %>% mutate(num_big_actors = is)

is.list(movies[1,10:13])
sum(is.element(movies[3,10:13], top20actorsnames)) ## this counts the number of actors that exist in th

## "Encoding" for number of big actors
movies$num_top_actors <- sum((is.element(movies[,10:13], top20actorsnames)))

##debug
actors_columns <- movies[,10:13]
is.element(actors_columns[8], top20actorsnames)

kable(top20actorstable, format = "html") %>% kable_styling()
```

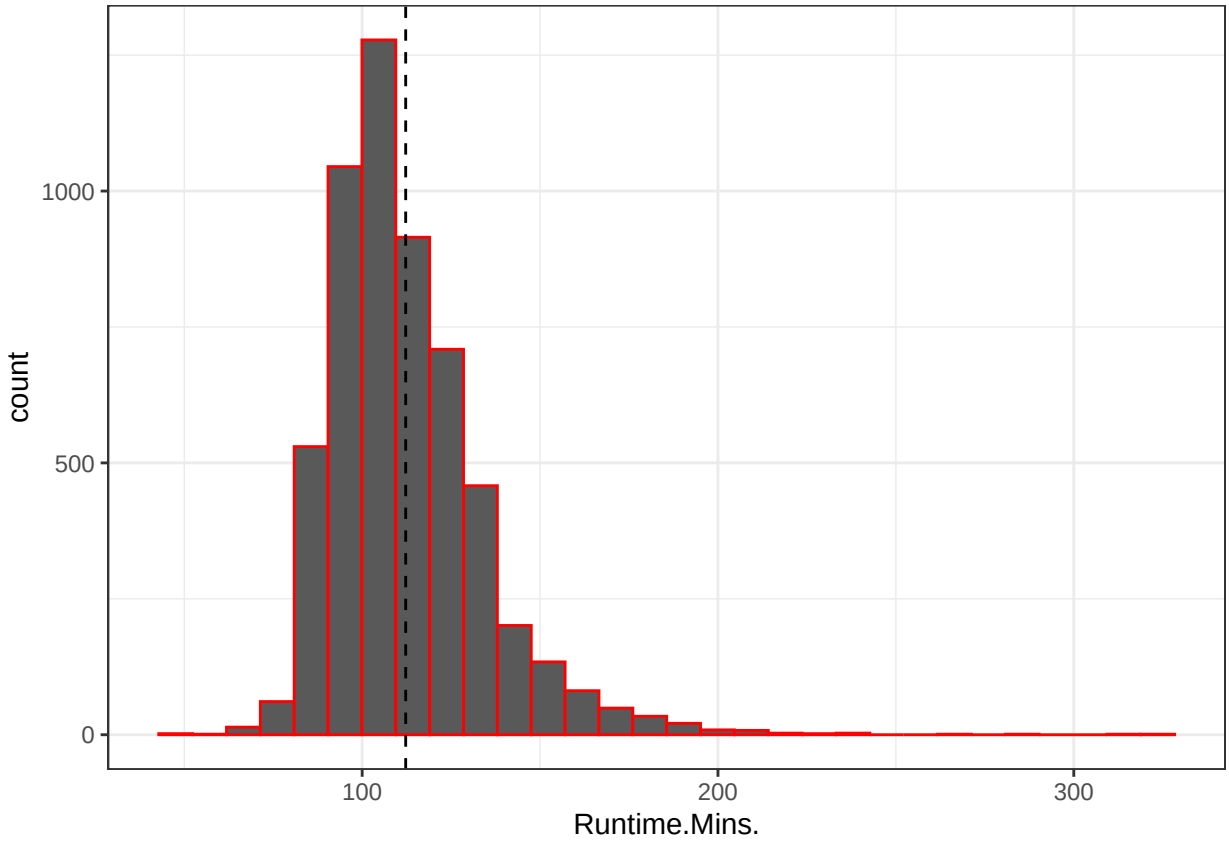
```
movies %>% ggplot(aes(Rating)) + geom_histogram(col = "red") +
  theme_bw() +
  geom_vline(xintercept = mean(movies$Rating), linetype = "dashed") +
  scale_x_continuous(breaks = (seq(1,10,by = 1)))
```

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



```
movies %>% ggplot(aes(Runtime.Mins.)) + geom_histogram(col = "red") +
  theme_bw() +
  geom_vline(xintercept = mean(movies$Runtime.Mins.), linetype = "dashed")
```

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



Very right skewed movie durations, with most centering around low 100 minutes.

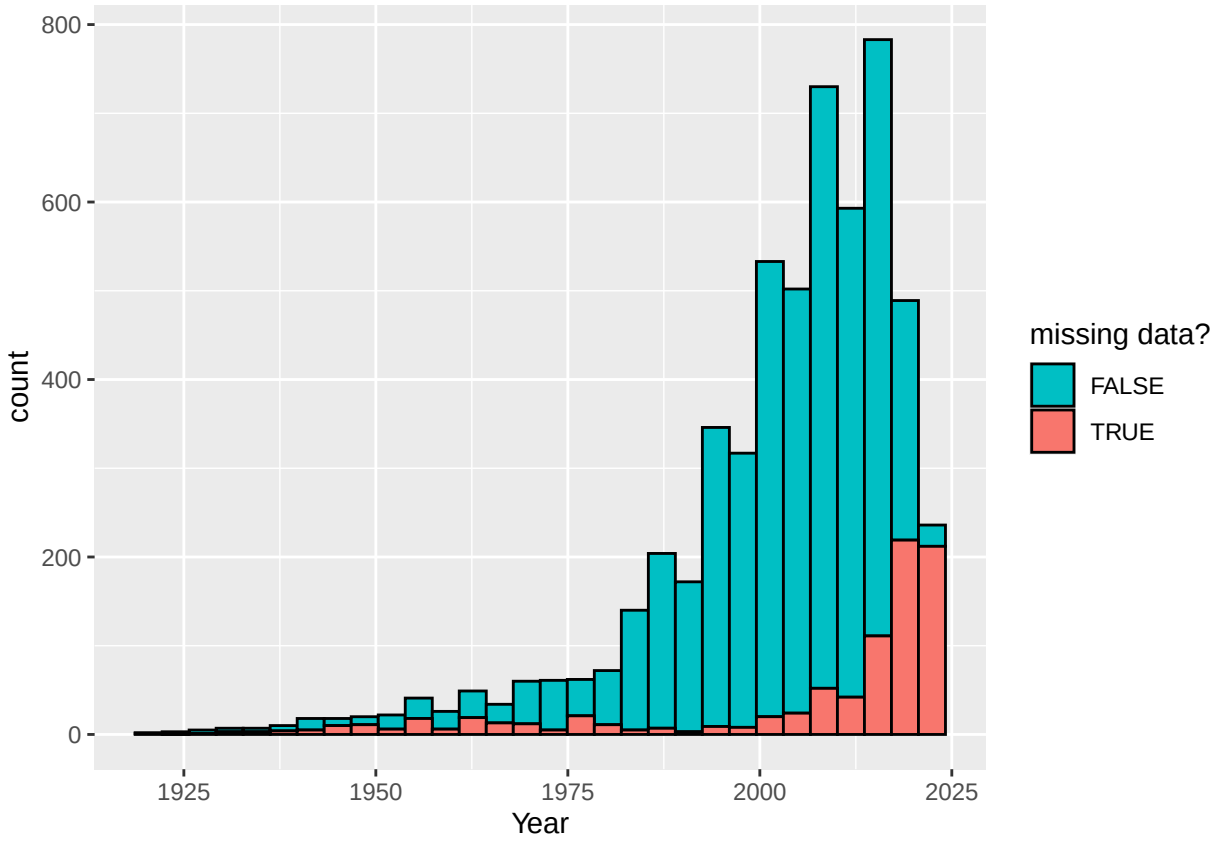
- Range of values: there are obviously outliers, 45, 321, with the highest at 321, 5.35 hours of movies, which is very long! Compared to the average movie at 112.226717, this is almost 3 times as long
- The movie(s) is(are) Gangs of Wasseypur, Novecento

```
sum(is.na(movies$Total_Gross))
```

```
## [1] 861
```

```
missinggross_movies <- movies[is.na(movies$Total_Gross),]

## Investigate?

# in terms of Year
movies %>% ggplot(aes(x = Year)) + geom_histogram(aes(fill = is.na(Total_Gross)), col = "black") +
  scale_fill_manual(values = c("#00BFC4","#F8766D" ))+ ## flip the colours
  guides(fill = guide_legend(title = "missing data?"))
```

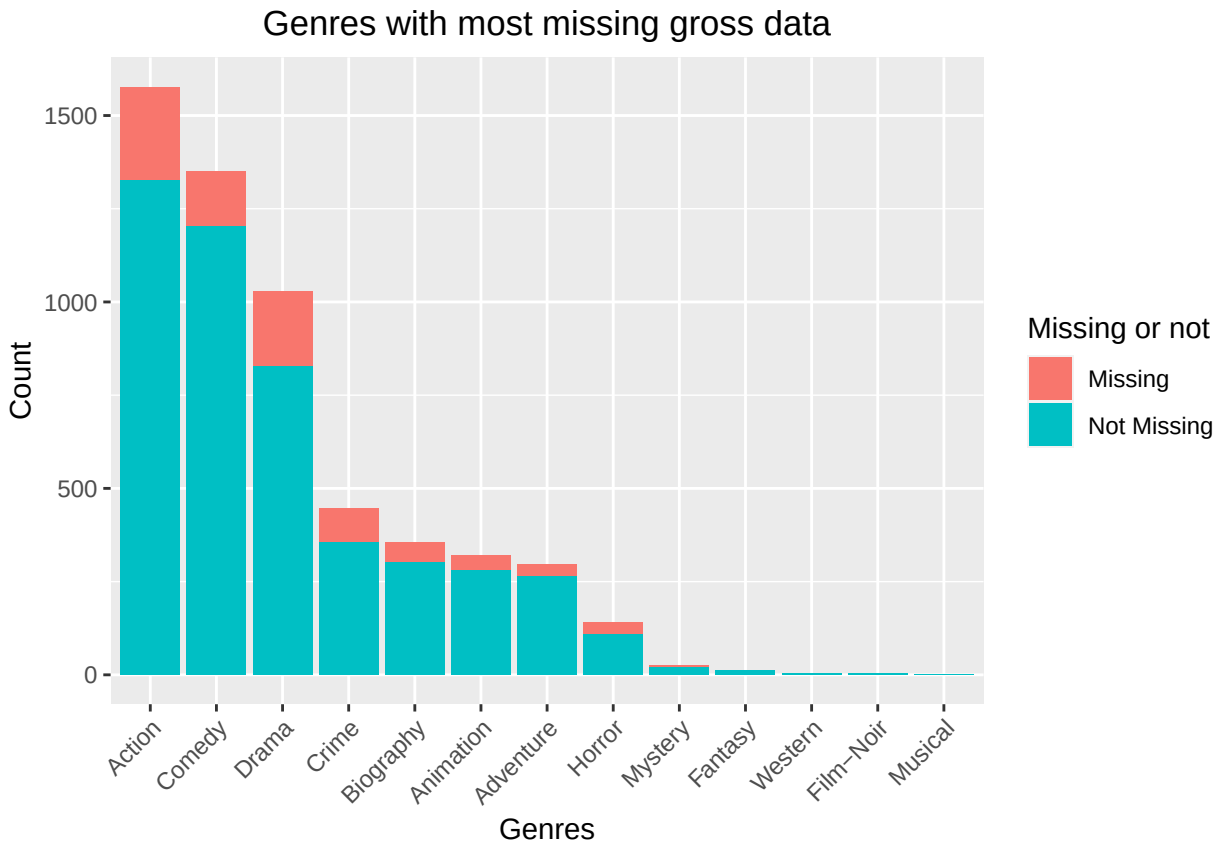
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
moviesgenrecount <- movies %>% group_by(main_genre) %>% count()

## table of missing vs not missing genre (for plotting barplots)
table_plot <- movies %>% group_by(main_genre) %>% summarise(no_missing = sum(is.na(Total_Gross))) %>% l
```

```
## Joining with `by = join_by(main_genre)`
```

```
## make long table to be able to plot on ggplot barplot
longtable <- table_plot %>%pivot_longer(cols = c("no_missing", "no_not_missing"), names_to = "Missing o
                          values_to = "Count")

longtable %>% ggplot(aes(x = reorder(main_genre, -Count), y = Count, fill = `Missing or not`)) + geom_b
       x = "Genres",
       y = "Count")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(hjust = .5)) +
  scale_fill_discrete(labels = c("Missing", "Not Missing")) ## change legend labels in scale_fill_discr
```



From the histogram, there is a skew towards modern day movies that are missing gross totals.

In terms of directors, 0 of the missing ones come from the top 10 directors.

Missing data for gross come somewhat proportionate from genres as well.

```
movies %>% group_by(Censor) %>% count() %>% arrange(desc(n))
```

```
## # A tibble: 25 x 2
## # Groups:   Censor [25]
##    Censor        n
##    <fct>     <int>
##  1 UA         1118
##  2 A          1101
##  3 U          1023
##  4 R           926
##  5 Not Rated   495
##  6 PG-13       405
##  7 18          136
##  8 PG          120
##  9 16           71
## 10 13           53
## # i 15 more rows
```

We will ignore censorship for now

Question 1: Big Directors Data

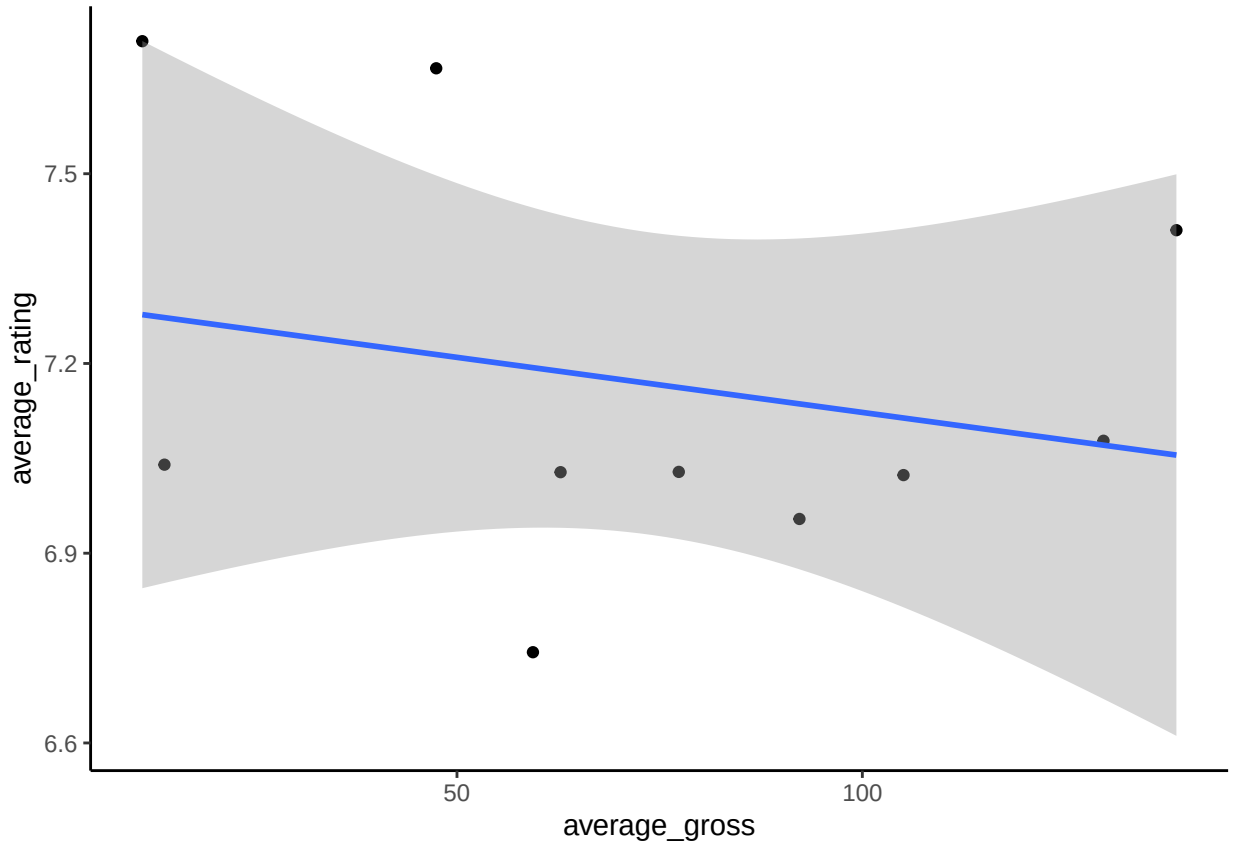
- Explore the ratings and gross of the top directors

```
director1 <- movies %>% filter(Director %in% top10directors$Director) %>%
  group_by(Director) %>%
  summarise(average_gross = mean(Total_Gross, na.rm = T),
            average_rating = mean(Rating, na.rm = T)) %>%
  mutate(gross_rank = dense_rank(desc(average_gross))) %>%
  mutate(rating_rank = dense_rank(desc(average_rating))) %>%
  arrange(desc(average_gross))

director1
```

```
## # A tibble: 10 x 5
##    Director          average_gross average_rating gross_rank rating_rank
##    <chr>                     <dbl>          <dbl>      <int>       <int>
##  1 Steven Spielberg          139.            7.41          1           3
##  2 Robert Zemeckis           130.            7.08          2           4
##  3 Tim Burton                105.            7.02          3           8
##  4 Ron Howard                 92.2           6.95          4           9
##  5 Ridley Scott               77.4           7.03          5           6
##  6 Clint Eastwood             62.8           7.03          6           7
##  7 Steven Soderbergh          59.4           6.74          7          10
##  8 Martin Scorsese            47.4           7.67          8           2
##  9 Woody Allen                13.9           7.04          9           5
## 10 Alfred Hitchcock           11.2           7.71         10           1
```

```
director1 %>% ggplot(aes(average_gross, average_rating)) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

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




```
movies %>% filter(Director %in% top10directors$Director) %>%
  group_by(Director) %>%
  count(main_genre) %>%
  slice_max(n =1, order_by = n)
```

```
## # A tibble: 12 x 3
## # Groups:   Director [10]
##    Director          main_genre     n
##    <chr>             <fct>      <int>
##  1 Alfred Hitchcock  Drama          7
##  2 Clint Eastwood    Action        10
##  3 Clint Eastwood    Drama         10
##  4 Martin Scorsese   Biography      6
##  5 Martin Scorsese   Crime          6
##  6 Ridley Scott      Action        10
##  7 Robert Zemeckis   Adventure      7
##  8 Ron Howard        Action        10
##  9 Steven Soderbergh Crime          8
## 10 Steven Spielberg  Action        12
## 11 Tim Burton        Adventure      5
## 12 Woody Allen       Comedy        37
```

Can see that there is no clear correlation between ratings and gross for top directors, but our sample size is very small here

This begs the question: in general, is there a clear relationship between the two variables? Take a bit of a detour

Relationship between Gross and Rating - is it clearly positive?

Do better movies make more money?

```
names(movies)
```

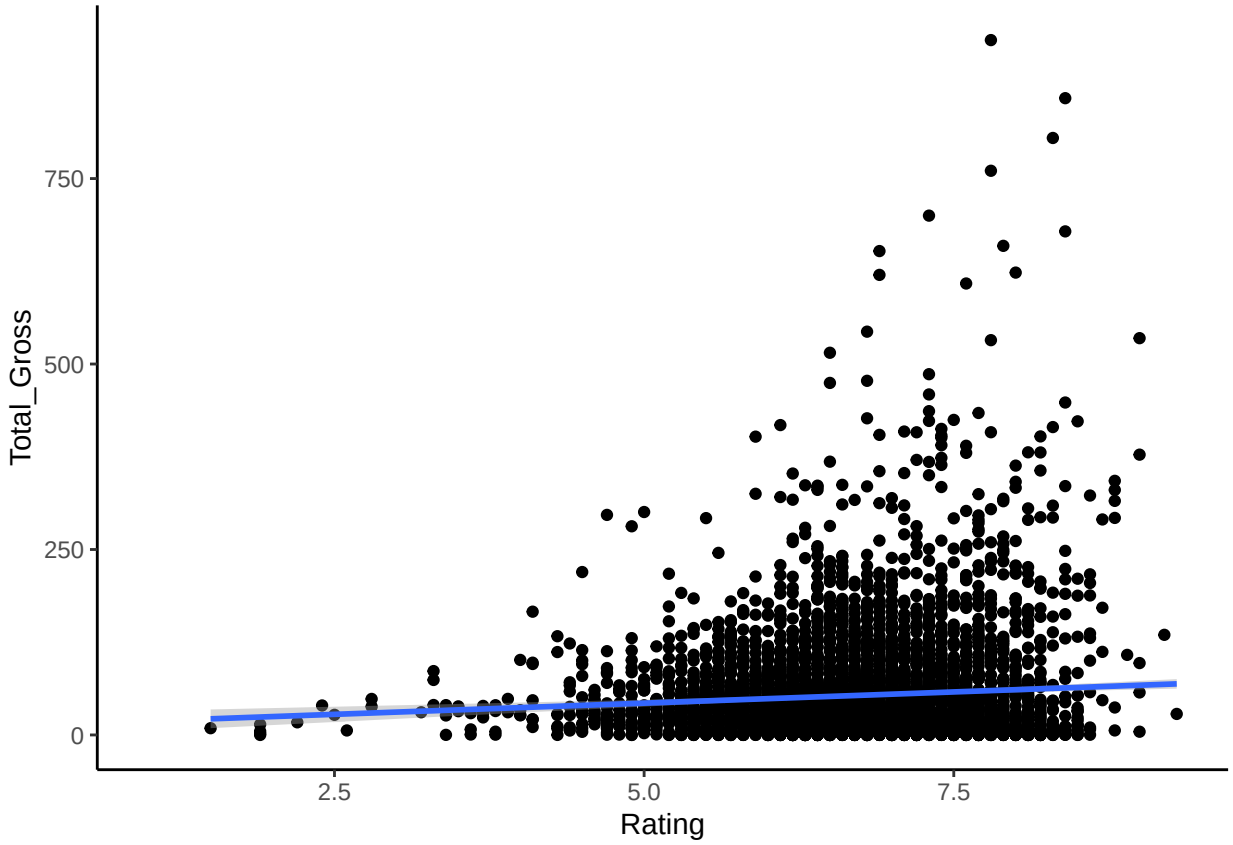
```
##  [1] "Movie_Title"   "Year"          "Director"      "Actors"
##  [5] "Rating"        "Runtime.Mins." "Censor"        "Total_Gross"
##  [9] "main_genre"    "side_genre"
```

```
## Plot Total Gross against Ratings
movies %>% ggplot(aes(y = Total_Gross, x = Rating)) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

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

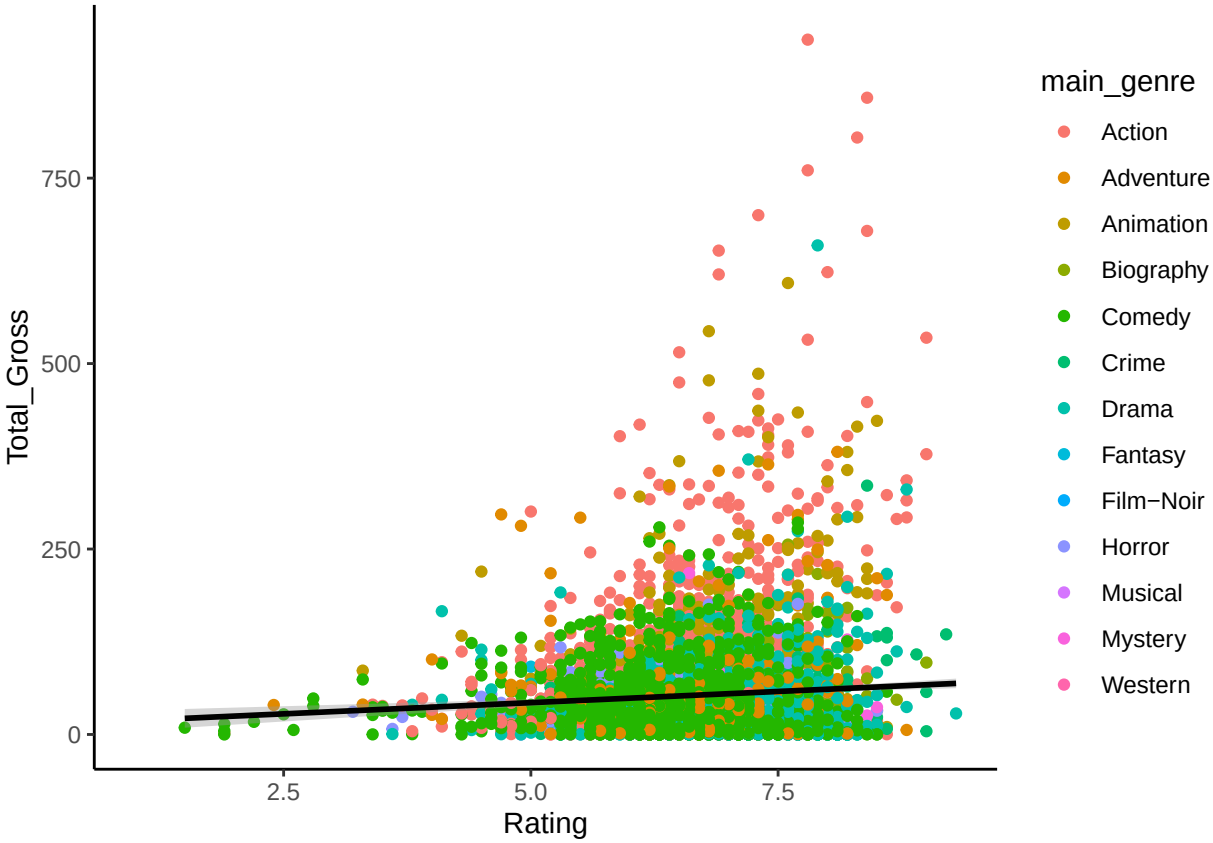
```
## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 861 rows containing missing values (`geom_point()`).
```




```
## add colours for genre
movies %>% ggplot(aes(y = Total_Gross, x = Rating, colour = main_genre)) + geom_point() +
  theme_classic() +
  geom_smooth(method = 'lm',color = "black")
```

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

## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).
## Removed 861 rows containing missing values (`geom_point()`).
```



Strong statistical significance of linear effect of rating on total gross

- Interpretation: For every 1 unit increase in rating, movies make ~ 6 million more

Not great linear modelling since

- Very few X axis data (total gross) for high values : left skewed
  - 53.4367198 vs 0, 7.1, 30.98, 67.36, 936.66
  - therefore confidence interval is very wide once we move rightwards
  - for each of the higher ratings, there are anomalies as well (data points that are much higher than normal) and this is related to the next point on variance

Some of the exploding variance can be accounted for when we segment the plot into each genre – we get to see a clear pattern of total gross - rating relationship for each genre.

Using interaction terms, we can see that

- Variance of Rating
  - the higher the rating, the higher the variance (spread of data points) for total gross

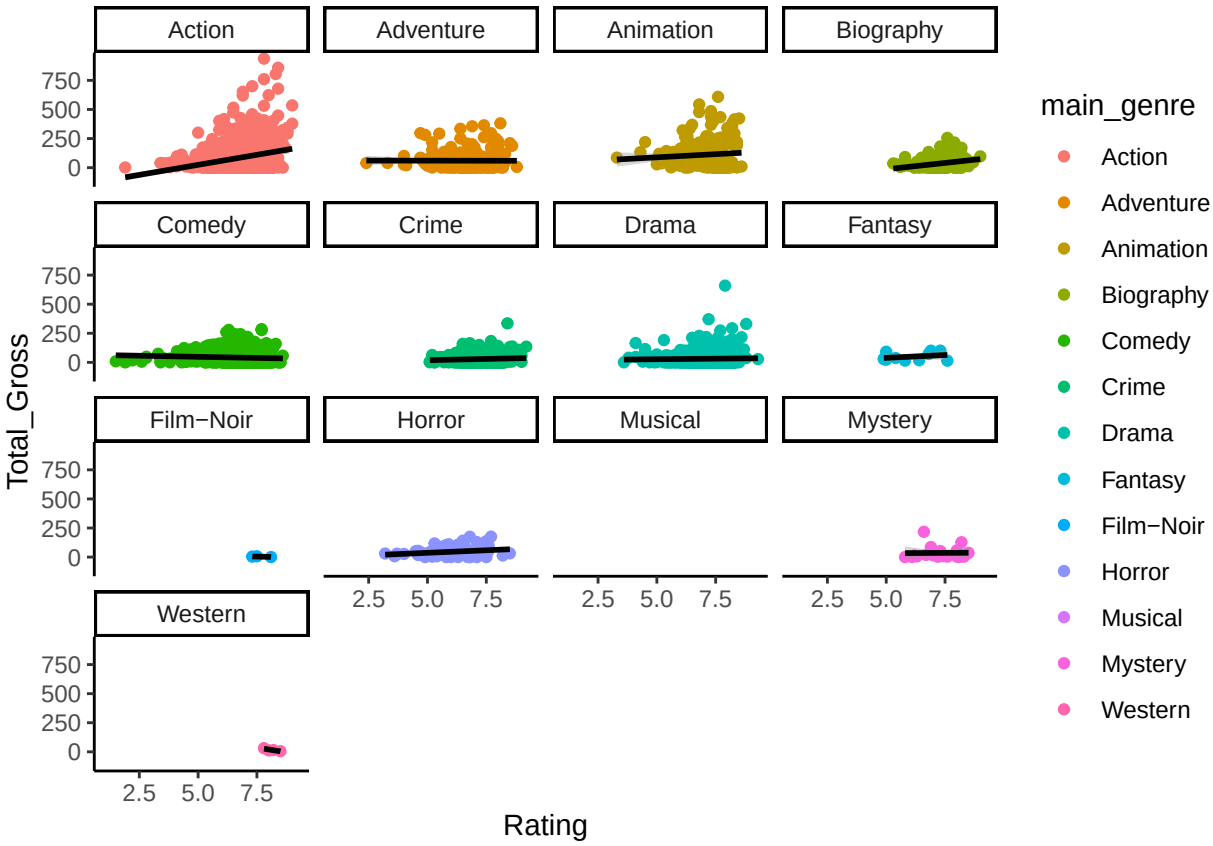
Perhaps we should try to segment based on categorical variables?

```
## add facet across genre for many plots per genre (see for interactions )
movies %>% ggplot(aes(y = Total_Gross, x = Rating, colour = main_genre)) + geom_point() +
  theme_classic() +
  facet_wrap(~main_genre) +
  geom_smooth(method = 'lm', color = "black")
```

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

```
## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 861 rows containing missing values (`geom_point()`).
```




```
## Regress without and with genre interactions
fit1 <- (lm(Total_Gross ~ Rating, movies))
summary(fit1)
```

```
##
## Call:
## lm(formula = Total_Gross ~ Rating, data = movies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -64.35 -46.63 -20.95  15.88 876.87
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   12.694      8.236   1.541    0.123
## Rating         6.038      1.210   4.991 6.23e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 74.64 on 4699 degrees of freedom
##   (861 observations deleted due to missingness)
## Multiple R-squared:  0.005273,   Adjusted R-squared:  0.005061
## F-statistic: 24.91 on 1 and 4699 DF,  p-value: 6.225e-07
```

```
## With interactions
fit2 <- (lm(Total_Gross ~ Rating*main_genre, movies)) ## Action = reference category
summary(fit2)
```

```
##
## Call:
## lm(formula = Total_Gross ~ Rating * main_genre, data = movies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -148.10  -32.36  -16.38   17.27  815.90
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -148.464     14.925  -9.947  < 2e-16 ***
## Rating                         34.515      2.267  15.223  < 2e-16 ***
## main_genreAdventure           209.726     33.552   6.251 4.44e-10 ***
## main_genreAnimation           182.673     39.915   4.577 4.85e-06 ***
## main_genreBiography            24.755     54.951   0.450 0.652372
## main_genreComedy              215.168     20.651  10.419  < 2e-16 ***
## main_genreCrime               145.065     37.883   3.829 0.000130 ***
## main_genreDrama               165.744     25.993   6.376 1.99e-10 ***
## main_genreFantasy             138.571    130.000   1.066 0.286512
## main_genreFilm-Noir           201.403    900.879   0.224 0.823107
## main_genreHorror              141.395     43.170   3.275 0.001063 **
## main_genreMystery             176.963    141.114   1.254 0.209891
## main_genreWestern             431.387   1091.040   0.395 0.692573
## Rating:main_genreAdventure    -34.757      4.976  -6.985 3.26e-12 ***
## Rating:main_genreAnimation    -23.631      5.729  -4.125 3.78e-05 ***
## Rating:main_genreBiography    -12.608      7.618  -1.655 0.097956 .
## Rating:main_genreComedy       -38.315      3.135 -12.223  < 2e-16 ***
## Rating:main_genreCrime        -30.106      5.367  -5.609 2.15e-08 ***
## Rating:main_genreDrama        -32.579      3.745  -8.700  < 2e-16 ***
## Rating:main_genreFantasy      -24.791     21.360  -1.161 0.245853
## Rating:main_genreFilm-Noir    -40.929    117.908  -0.347 0.728512
## Rating:main_genreHorror       -25.742      7.084  -3.634 0.000282 ***
## Rating:main_genreMystery      -33.491     19.246  -1.740 0.081899 .
## Rating:main_genreWestern      -67.347    134.220  -0.502 0.615856
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 69.41 on 4677 degrees of freedom
##   (861 observations deleted due to missingness)
## Multiple R-squared:  0.1438, Adjusted R-squared:  0.1396
## F-statistic: 34.16 on 23 and 4677 DF,  p-value: < 2.2e-16
```

```
## there are significant interaction terms for genres i.e. movies have different gross-rating relationsl
```

Linear model

We get a very significant coefficient/slope of rating. However, as we know from the plot, this is very misleading - we have exploding variance of the gross and ratings increase. This means that there is a lot of imprecision of the slope estimator (imagine fitting a line where you can pivot using the left side of the line, since the right side has so much variance).

Hence we cannot interpret this slope p value, despite looking very significant. However, according to OLS theory, the slope estimate itself is still OK in the sense that it is unbiased. Hence we do know that there is a positive relationship between ratings and gross total, i.e. the better the movie, the more money it makes.

Model evaluation (AIC) : 5.3892679×10^4, however useless as a standalone metric.

Segmenting on genre Looking at a deeper level, there are different trend/slopes for movies according to the genre they fall in

Linear regression looks much cleaner after faceting for genre, although some still suffer from increasing variance.

- Action movies may start at the lowest gross for the lowest ratings
- But they overtake the other genres once ratings start to go up, having the largest increase in gross per unit increase in ratings
  - this means that action movies are very profitable, as long as they are good
  - but action movies are plagued by bad heteroskedasticity - when movies are good, there is a lot of uncertainty about how much they can make!

This is seen from the estimate of slope (34) while every interaction term is negative - meaning that the other genres have a lower slope than Action which is the reference category.

- Comedy starts at the highest base total gross (i.e. intercept/ when rating = 0), but increases at a very modest rate (one of the smallest slopes with Noir and Western which is the worst)

Was it worth the split? 5.3231493×10^4 VS 5.3892679×10^4

A drop in AIC, which is a good sign.

c(4699, 4677), c(26178927.6830318, 22532241.0310387), c(NA, 22), c(NA, 3646686.65199313), c(NA, 34.4063615087809), c(NA, 1.75194698231913e-134)

The extra sums of squares are significant!

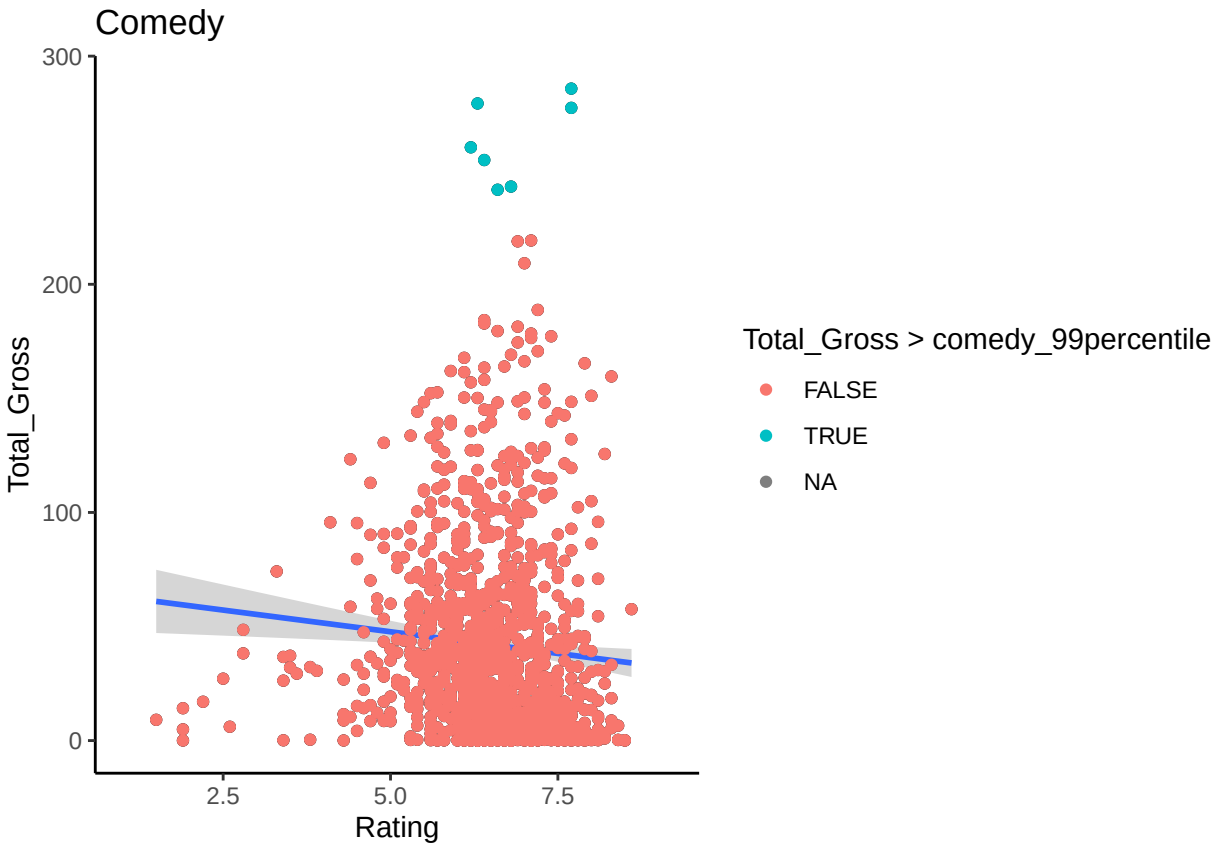
We have some anomalies from Comedy, Crime and Drama. Lets see if we can spot those movies

```
comedy_subset <- movies %>% filter(main_genre == "Comedy")
comedy_99percentile <- quantile(comedy_subset$Total_Gross, 0.995, na.rm = T)

comedy_subset %>% ggplot(aes(y = Total_Gross, x = Rating )) + geom_point() +
  theme_classic() +
  geom_smooth(method = "lm") +
  geom_point(aes(col = Total_Gross > comedy_99percentile)) +
  labs(title = "Comedy")
```

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

```
## Warning: Removed 147 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 147 rows containing missing values (`geom_point()`).
## Removed 147 rows containing missing values (`geom_point()`).
```




```
comedy_subset %>% filter(Total_Gross > comedy_99percentile)
```

```
##                     Movie_Title Year       Director
## 1       My Big Fat Greek Wedding 2002     Joel Zwick
## 2                    The Hangover 2009  Todd Phillips
## 3                      Home Alone 1990 Chris Columbus
## 4                  Bruce Almighty 2003    Tom Shadyac
## 5           The Hangover Part II 2011  Todd Phillips
```

```
## 6               Meet the Fockers 2004      Jay Roach
## 7 How the Grinch Stole Christmas 2000     Ron Howard
##                                                                      Actors
## 1 Nia Vardalos, John Corbett, Michael Constantine, Christina Eleusiniotis
## 2              Zach Galifianakis, Bradley Cooper, Justin Bartha, Ed Helms
## 3                    Macaulay Culkin, Joe Pesci, Daniel Stern, John Heard
## 4         Jim Carrey, Jennifer Aniston, Morgan Freeman, Philip Baker Hall
## 5              Bradley Cooper, Zach Galifianakis, Ed Helms, Justin Bartha
## 6                   Ben Stiller, Robert De Niro, Blythe Danner, Teri Polo
## 7                       Jim Carrey, Taylor Momsen, Kelley, Jeffrey Tambor
##   Rating Runtime.Mins. Censor Total_Gross main_genre        side_genre
## 1    6.6            95      U      241.44     Comedy   Drama,  Romance
## 2    7.7           100     UA      277.32     Comedy            Comedy
## 3    7.7           103      U      285.76     Comedy            Family
## 4    6.8           101     UA      242.83     Comedy           Fantasy
## 5    6.4           102      A      254.46     Comedy            Comedy
## 6    6.3           115      A      279.26     Comedy           Romance
## 7    6.2           104      U      260.04     Comedy  Family,  Fantasy
```

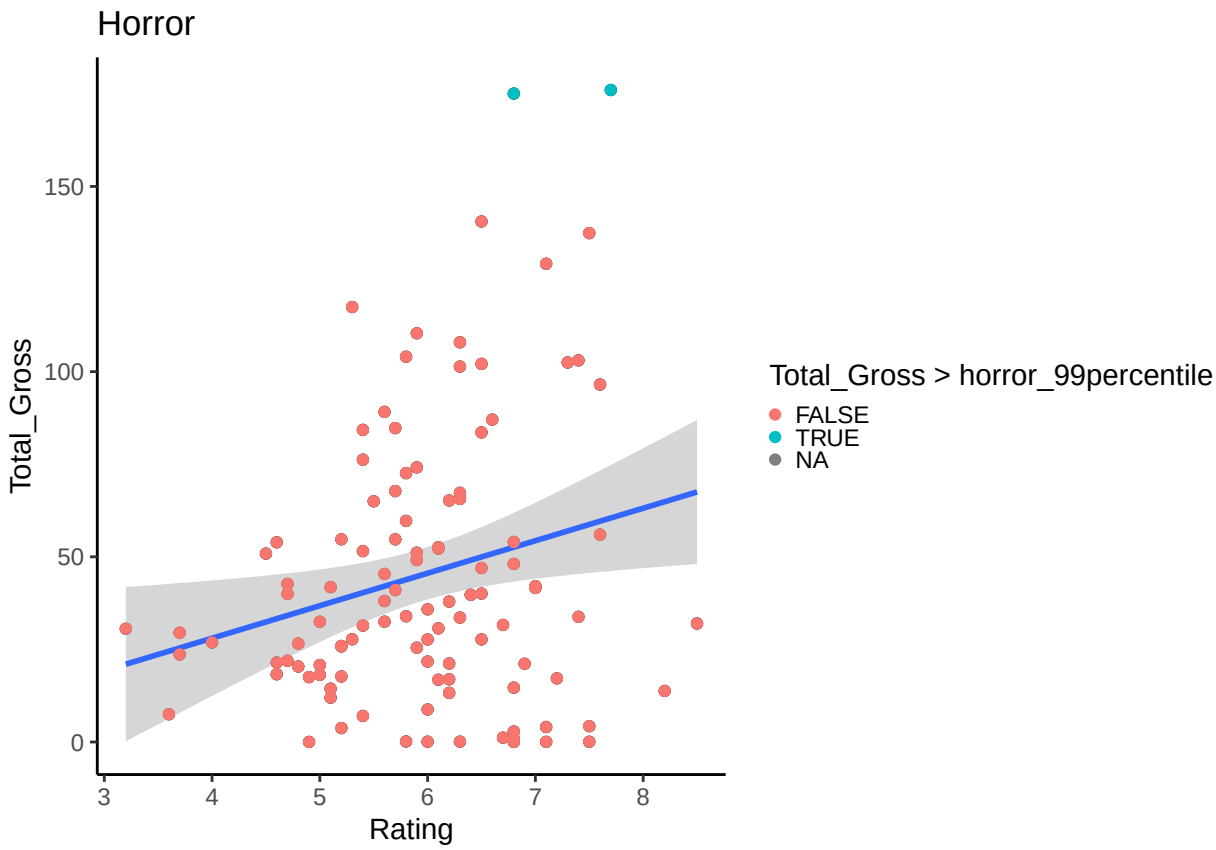
Closer inspection reveals that they are not clear anomalies, but nevertheless, we got to see the most popular overall comedy movies.

```
horror_subset <- movies %>% filter(main_genre == "Horror")
horror_99percentile <- quantile(horror_subset$Total_Gross, 0.99, na.rm = T)

horror_subset %>% ggplot(aes(y = Total_Gross, x = Rating)) + geom_point() +
  theme_classic() + theme(legend.key.size = unit(0.1, "cm")) +
    geom_smooth(method = "lm") +
  geom_point(aes(col = Total_Gross > horror_99percentile)) +
  labs(title = "Horror")
```

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

```
## Warning: Removed 34 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 34 rows containing missing values (`geom_point()`).
## Removed 34 rows containing missing values (`geom_point()`).
```




```
horror_subset %>% filter(Total_Gross > horror_99percentile)
```

```
##   Movie_Title Year     Director
## 1     Get Out 2017 Jordan Peele
## 2          Us 2019 Jordan Peele
##                                                                  Actors Rating
## 1 Daniel Kaluuya, Allison Williams, Bradley Whitford, Catherine Keener    7.7
## 2          Lupita Nyong'o, Winston Duke, Elisabeth Moss, Tim Heidecker    6.8
##   Runtime.Mins. Censor Total_Gross main_genre          side_genre
## 1           104    15+      176.04     Horror  Mystery,  Thriller
## 2           116      A      175.08     Horror  Mystery,  Thriller
```

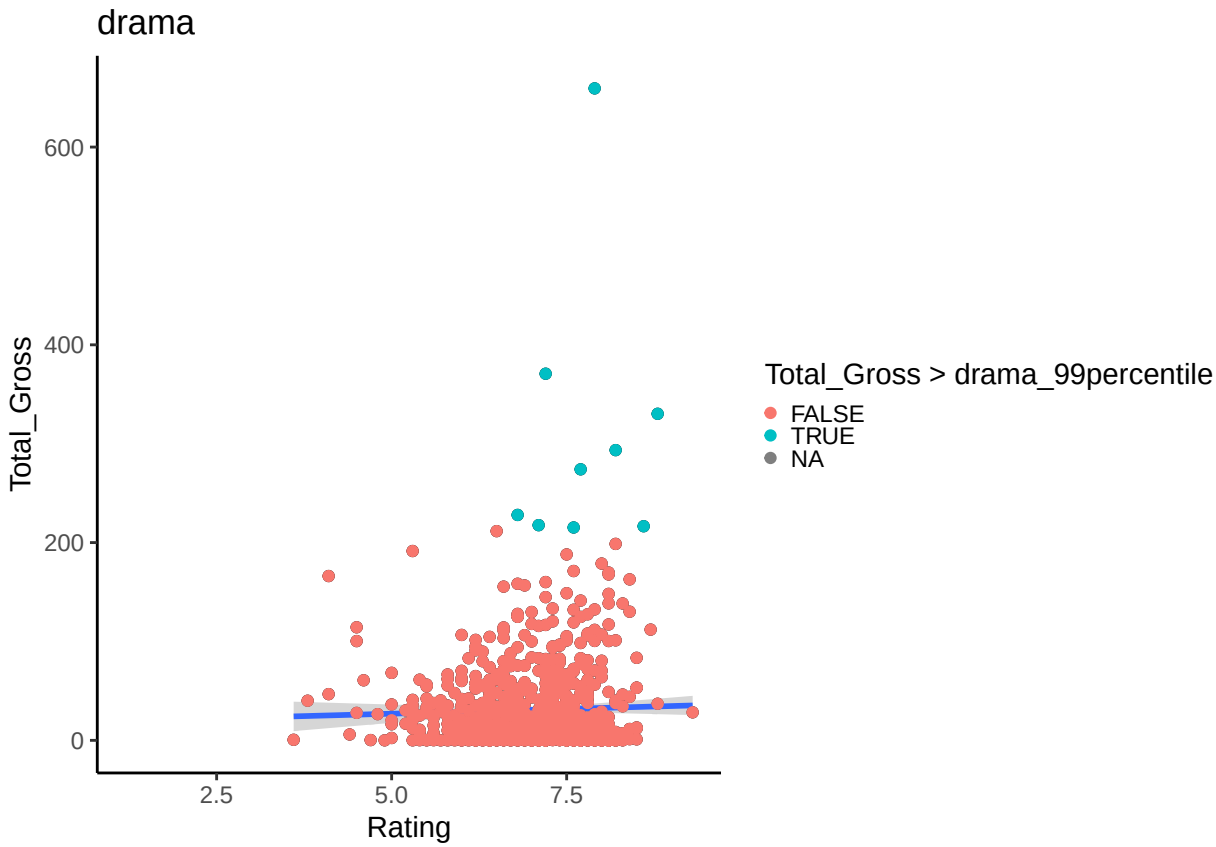
Jorden Peele is really making his mark.

```
drama_subset <- movies %>% filter(main_genre == "Drama")
drama_99percentile <- quantile(drama_subset$Total_Gross, 0.99, na.rm = T)

drama_subset %>% ggplot(aes(y = Total_Gross, x = Rating)) + geom_point() +
  theme_classic() + theme(legend.key.size = unit(0.1, "cm")) +
    geom_smooth(method = "lm") +
  geom_point(aes(col = Total_Gross > drama_99percentile)) +
  labs(title = "drama")
```

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

```
## Warning: Removed 199 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 199 rows containing missing values (`geom_point()`).
## Removed 199 rows containing missing values (`geom_point()`).
```




```
drama_subset %>% filter(Total_Gross > drama_99percentile) %>% arrange(desc(Total_Gross))
```

```
##                 Movie_Title Year          Director
## 1                   Titanic 1997     James Cameron
## 2 The Passion of the Christ 2004        Mel Gibson
## 3              Forrest Gump 1994    Robert Zemeckis
## 4           The Sixth Sense 1999 M. Night Shyamalan
## 5                   Gravity 2013     Alfonso Cuarón
## 6                     Signs 2002 M. Night Shyamalan
## 7                     Ghost 1990       Jerry Zucker
## 8       Saving Private Ryan 1998   Steven Spielberg
## 9            A Star Is Born 2018     Bradley Cooper
##                                                    Actors Rating
## 1       Leonardo DiCaprio, Kate Winslet, Billy Zane, Kathy Bates    7.9
## 2 Jim Caviezel, Monica Bellucci, Maia Morgenstern, Christo Jivkov    7.2
## 3              Tom Hanks, Robin Wright, Gary Sinise, Sally Field    8.8
## 4 Bruce Willis, Haley Joel Osment, Toni Collette, Olivia Williams    8.2
## 5     Sandra Bullock, George Clooney, Ed Harris, Orto Ignatiussen    7.7
## 6       Mel Gibson, Joaquin Phoenix, Rory Culkin, Abigail Breslin    6.8
## 7       Patrick Swayze, Demi Moore, Whoopi Goldberg, Tony Goldwyn    7.1
## 8              Tom Hanks, Matt Damon, Tom Sizemore, Edward Burns    8.6
## 9          Lady Gaga, Bradley Cooper, Sam Elliott, Greg Grunberg    7.6
##   Runtime.Mins. Censor Total_Gross main_genre          side_genre
```

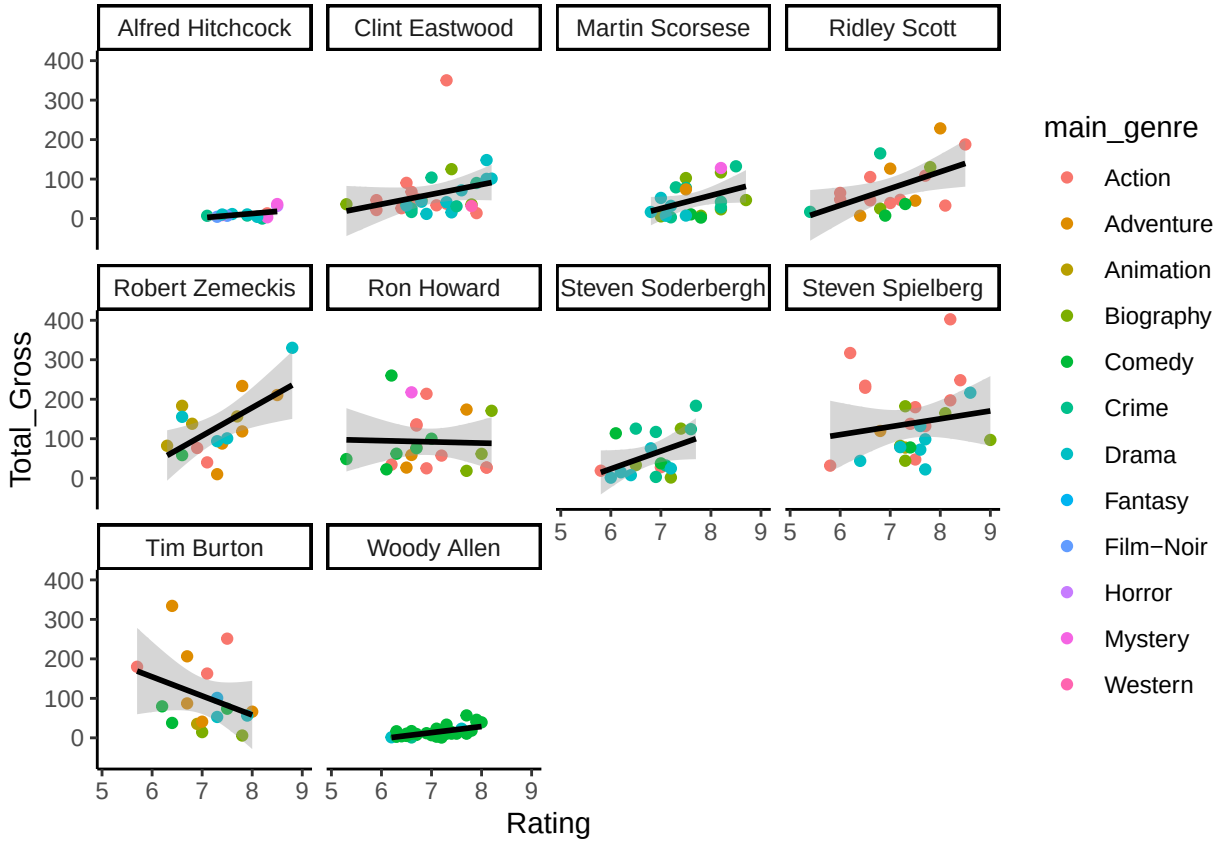
```
## 1           194     UA      659.33      Drama             Romance
## 2           127      A      370.78      Drama               Drama
## 3           142     UA      330.25      Drama             Romance
## 4           107      A      293.51      Drama  Mystery,  Thriller
## 5            91     UA      274.09      Drama   Sci-Fi,  Thriller
## 6           106     UA      227.97      Drama    Mystery,  Sci-Fi
## 7           127      A      217.63      Drama   Fantasy,  Romance
## 8           169      A      216.54      Drama                 War
## 9           136      A      215.29      Drama     Music,  Romance
```

```
## add facet for directors
movies %>% filter(Director %in% top10directors$Director)%>% ggplot(aes(y = Total_Gross, x = Rating, colo
  theme_classic() +
  facet_wrap(~Director) +
  geom_smooth(method = 'lm', color = "black")
```

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

```
## Warning: Removed 24 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 24 rows containing missing values (`geom_point()`).
```




```
## Regress with directors as interaction terms
top_directors_subset <- movies %>% filter(is.element(Director, top10directors$Director))
lm_fit_directors_interactions <- lm(Total_Gross~Rating*Director, data = top_directors_subset)
summary(lm_fit_directors_interactions) ## Woody Allen = reference category
```

```
##
## Call:
## lm(formula = Total_Gross ~ Rating * Director, data = top_directors_subset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -122.136  -36.628   -6.044   17.159  281.302
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -74.095    283.779  -0.261    0.794
## Rating                             10.854     36.057   0.301    0.764
## DirectorClint Eastwood            -38.617    305.265  -0.127    0.899
## DirectorMartin Scorsese          -131.909    344.108  -0.383    0.702
## DirectorRidley Scott             -147.186    311.803  -0.472    0.637
## DirectorRobert Zemeckis          -317.749    329.034  -0.966    0.335
## DirectorRon Howard                187.372    309.067   0.606    0.545
## DirectorSteven Soderbergh        -172.241    338.340  -0.509    0.611
## DirectorSteven Spielberg           64.289    308.532   0.208    0.835
## DirectorTim Burton                518.763    331.496   1.565    0.119
## DirectorWoody Allen               -21.011    313.533  -0.067    0.947
## Rating:DirectorClint Eastwood      14.015     39.392   0.356    0.722
## Rating:DirectorMartin Scorsese     22.204     44.063   0.504    0.615
## Rating:DirectorRidley Scott        31.617     40.419   0.782    0.435
## Rating:DirectorRobert Zemeckis     60.475     42.595   1.420    0.157
## Rating:DirectorRon Howard         -13.890     40.112  -0.346    0.729
## Rating:DirectorSteven Soderbergh   34.139     45.065   0.758    0.450
## Rating:DirectorSteven Spielberg     9.170     39.547   0.232    0.817
## Rating:DirectorTim Burton         -59.205     43.483  -1.362    0.175
## Rating:DirectorWoody Allen          4.619     40.695   0.113    0.910
##
## Residual standard error: 61.34 on 201 degrees of freedom
##   (24 observations deleted due to missingness)
## Multiple R-squared:  0.3997, Adjusted R-squared:  0.3429
## F-statistic: 7.044 on 19 and 201 DF,  p-value: 3.234e-14
```

Segmenting on directors Despite segmenting (interacting) based on directors, we still get a lot of noise, and we

Overall, to try to see if we can still get linearity, we should first try some simple transformations. If not:

- We may look into *robust standard errors* so that we can still make inferences, or *weighted least squares*

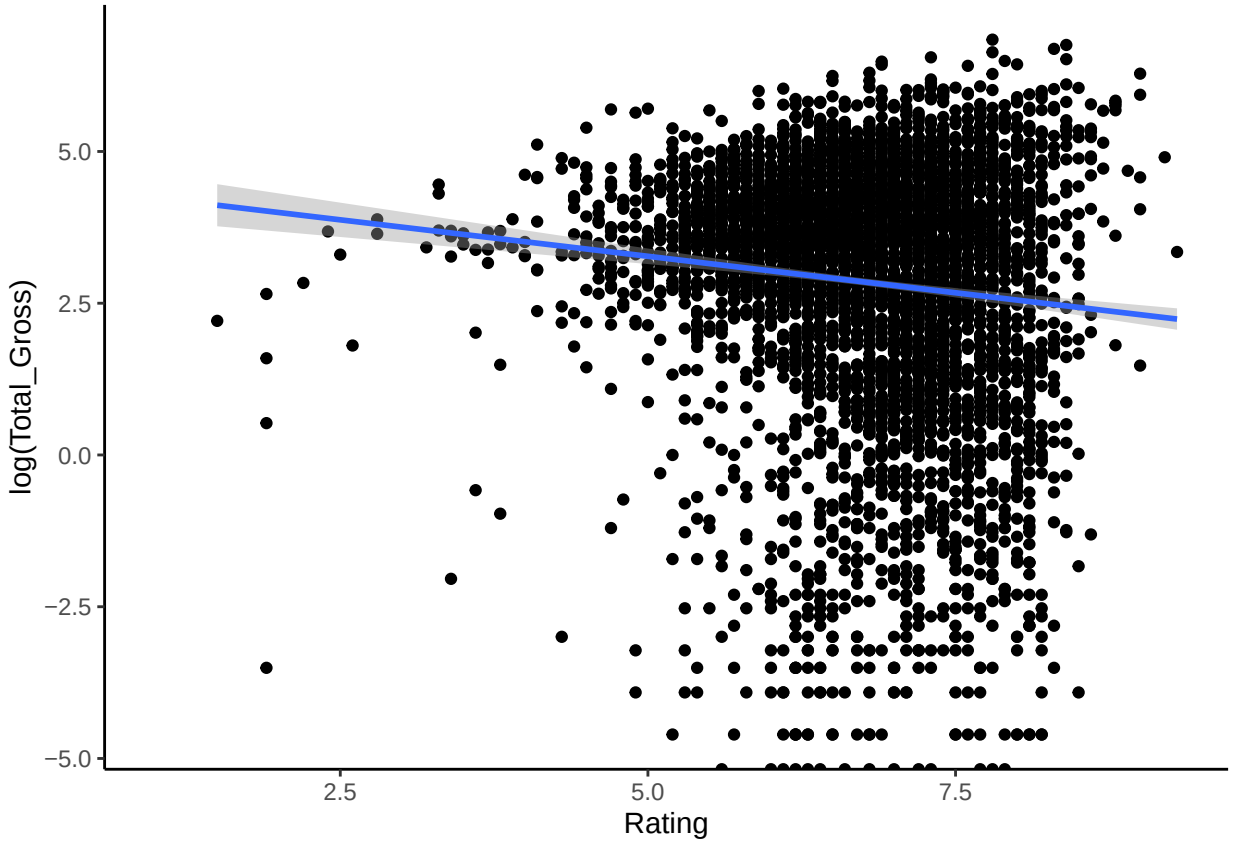
When we take a microcosmic look into the top directors, we do see that some directors have greater variability than others. And this highly correlates with genre once again -

```
## Log total gross
movies %>% ggplot(aes(y = log(Total_Gross), x = Rating)) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

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

```
## Warning: Removed 875 rows containing non-finite values (`stat_smooth()`).
```

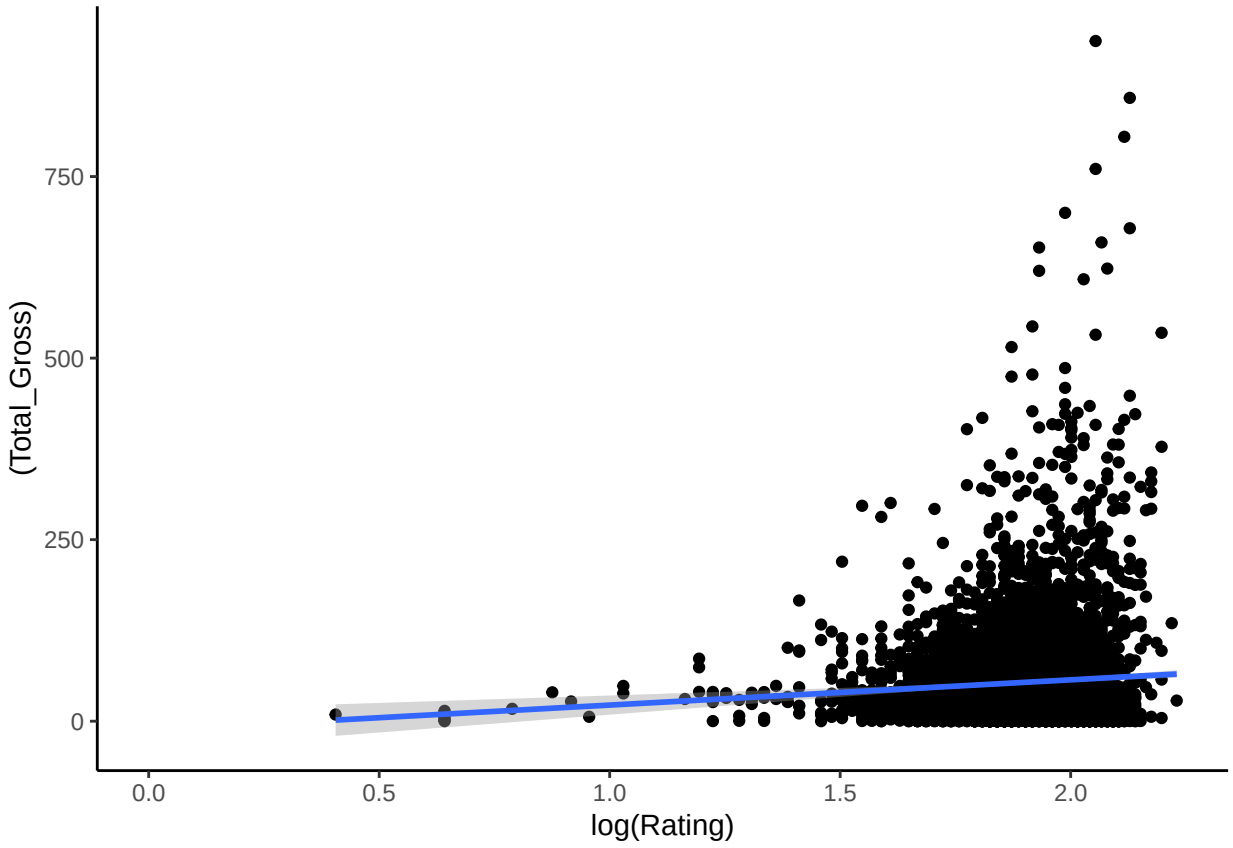
```
## Warning: Removed 861 rows containing missing values (`geom_point()`).
```




```
## Log rating
movies %>% ggplot(aes(y = (Total_Gross), x = log(Rating))) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

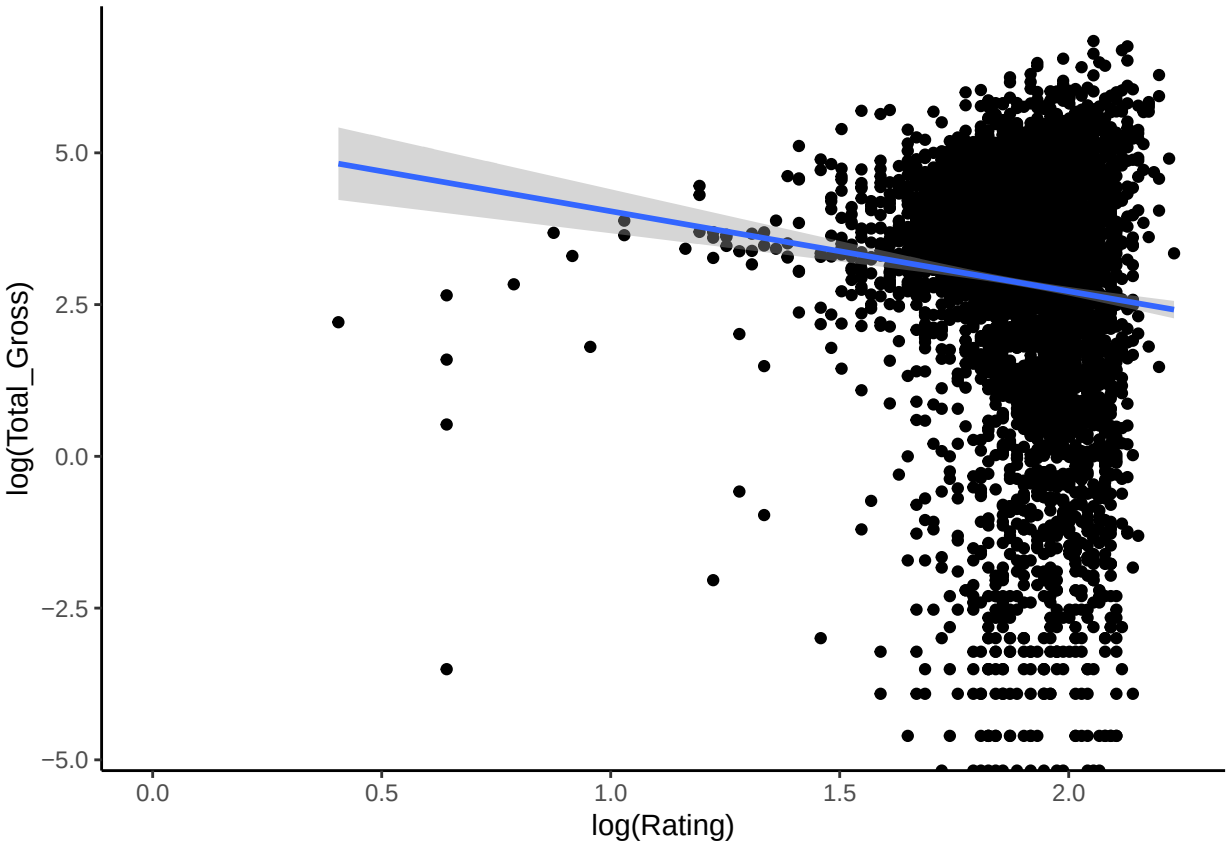
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).
## Removed 861 rows containing missing values (`geom_point()`).
```




```
## Log both
movies %>% ggplot(aes(y = log(Total_Gross), x = log(Rating))) + geom_point() +geom_smooth(method = 'lm'
  theme_classic()
```

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

## Warning: Removed 875 rows containing non-finite values (`stat_smooth()`).
## Removed 861 rows containing missing values (`geom_point()`).
```



When logging the data, we may end up getting a decreasing relationship. Hence, we will not use log transformations.

Non linear modelling: Polynomials and Splines

Local regression

LOESS: (Wiki) combines the simplicity of linear least squares with the flexbility of nonlinear regression - fits simple models to localised regions of the data

- main benefit is that the analyst does not need to specify the function f(x)
- allows modelling complex processes for which no theoretical models exist

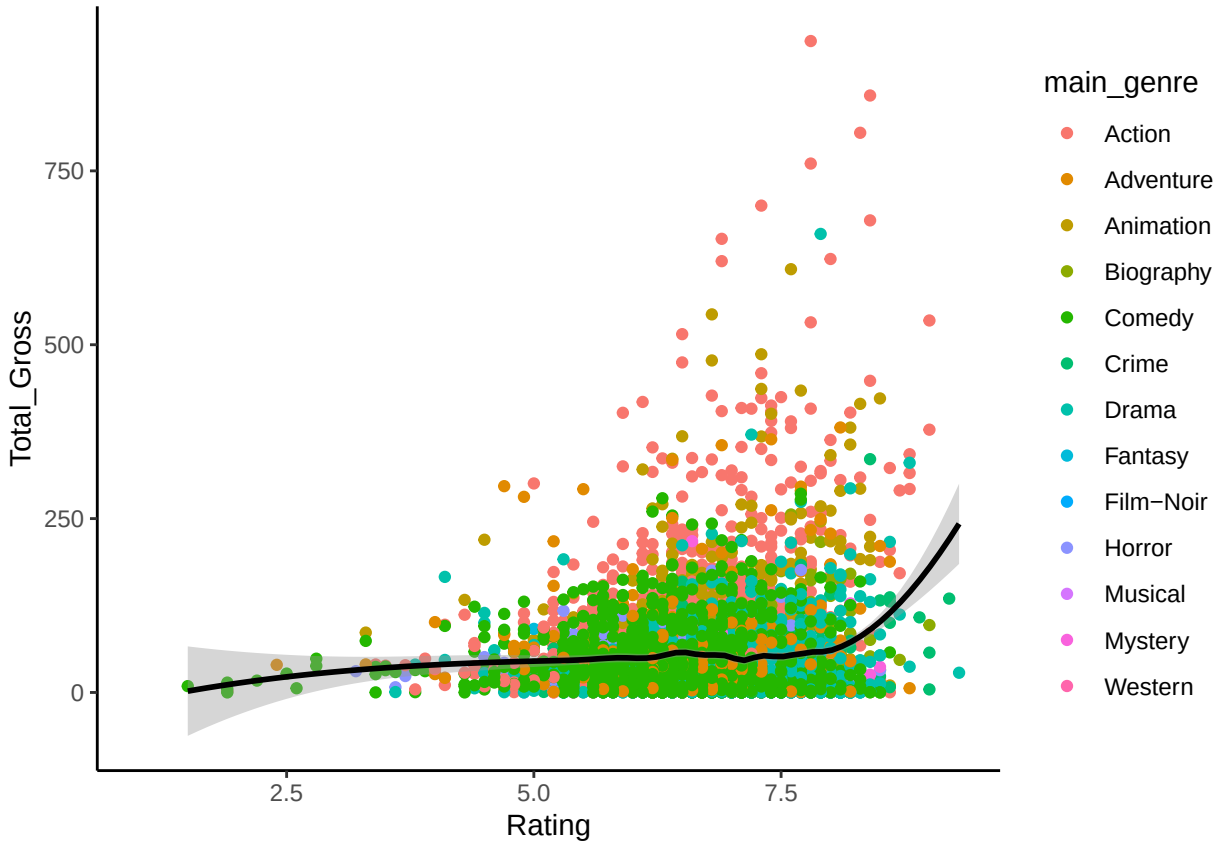
```
## 1: Locally estimated scatterplot smoothing
# What is the model?
loessfit <- loess(Total_Gross ~ Rating, data = movies, span = 0.33)
summary(loessfit)
```

```
## Call:
## loess(formula = Total_Gross ~ Rating, data = movies, span = 0.33)
##
## Number of Observations: 4701
## Equivalent Number of Parameters: 10.86
## Residual Standard Error: 74.14
## Trace of smoother matrix: 11.99  (exact)
```

```
##
## Control settings:
##   span     :  0.33
##   degree   :  2
##   family   :  gaussian
##   surface  :  interpolate      cell = 0.2
##   normalize:  TRUE
##  parametric:  FALSE
## drop.square:  FALSE
```

```
movies %>% ggplot(aes(y = Total_Gross, x = Rating, colour = main_genre)) + geom_point() +
  theme_classic() +
  geom_smooth(method = 'loess', span = 1/3 ,color = "black")
```

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

## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 861 rows containing missing values (`geom_point()`).
```




```
#2: Polynomial regression
movies1 <- movies %>% filter(!is.na(Total_Gross))
fit_order2 <- lm(movies1$Rating ~ poly(movies1$Total_Gross,2))
fit_order4 <- lm(movies1$Rating ~ poly(movies1$Total_Gross,4))

summary(fit_order2) ## Adjusted R-squared:  0.01306
```

```
##
## Call:
## lm(formula = movies1$Rating ~ poly(movies1$Total_Gross, 2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2471 -0.5345  0.0569  0.6476  2.5657
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    6.74765    0.01304 517.473  < 2e-16 ***
## poly(movies1$Total_Gross, 2)1  4.48019    0.89405   5.011 5.61e-07 ***
## poly(movies1$Total_Gross, 2)2  5.58830    0.89405   6.251 4.45e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.894 on 4698 degrees of freedom
## Multiple R-squared:  0.01348,    Adjusted R-squared:  0.01306
## F-statistic: 32.09 on 2 and 4698 DF,  p-value: 1.437e-14
```

```
summary(fit_order4) ## Adjusted R-squared:  0.04498
```

```
##
## Call:
## lm(formula = movies1$Rating ~ poly(movies1$Total_Gross, 4))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3240 -0.5305  0.0522  0.6161  2.6142
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    6.74765    0.01283 526.051  < 2e-16 ***
## poly(movies1$Total_Gross, 4)1  4.48019    0.87947   5.094 3.64e-07 ***
## poly(movies1$Total_Gross, 4)2  5.58830    0.87947   6.354 2.29e-10 ***
## poly(movies1$Total_Gross, 4)3 -7.86370    0.87947  -8.941  < 2e-16 ***
## poly(movies1$Total_Gross, 4)4  7.82231    0.87947   8.894  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8795 on 4696 degrees of freedom
## Multiple R-squared:  0.0458, Adjusted R-squared:  0.04498
## F-statistic: 56.35 on 4 and 4696 DF,  p-value: < 2.2e-16
```

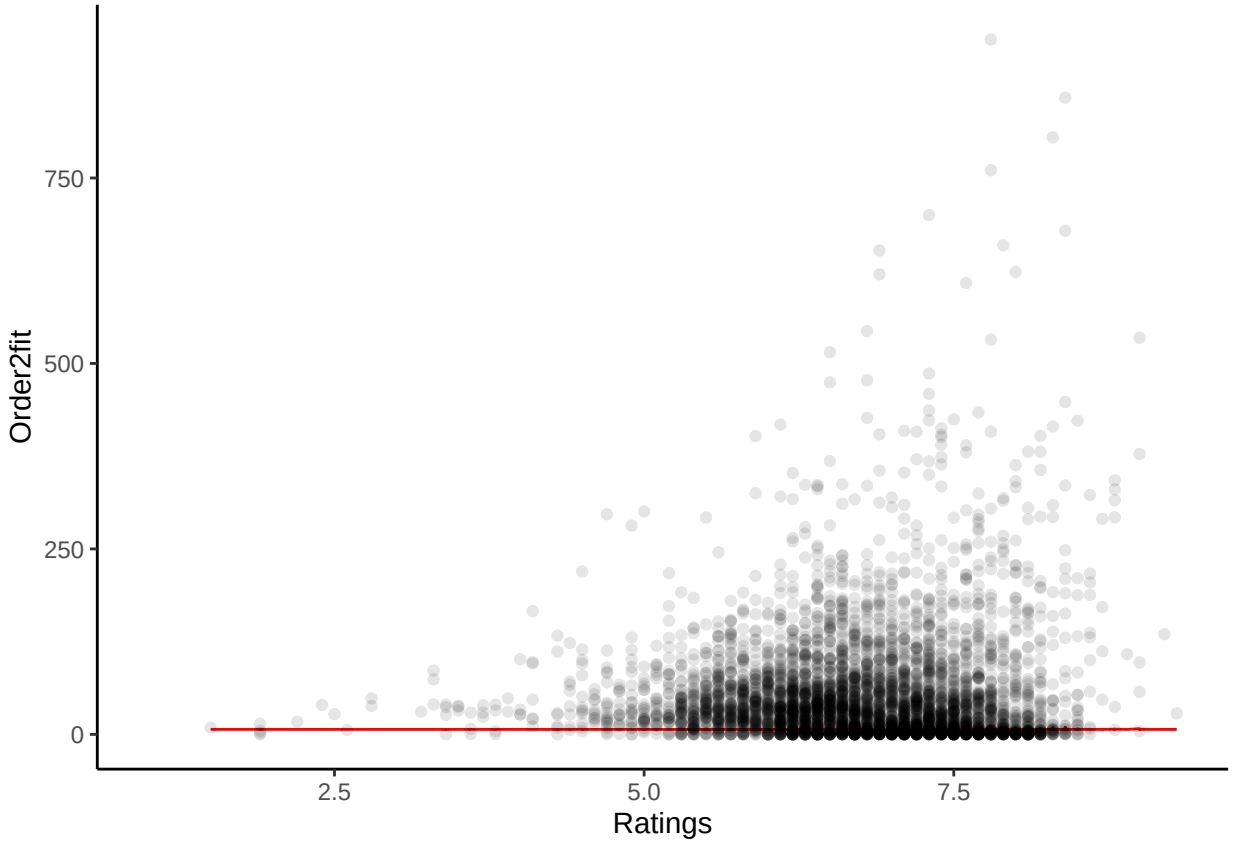
```
anova(fit_order2, fit_order4) ## restricted model, then full model for df to be positive
```

```
## Analysis of Variance Table
##
## Model 1: movies1$Rating ~ poly(movies1$Total_Gross, 2)
## Model 2: movies1$Rating ~ poly(movies1$Total_Gross, 4)
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1   4698 3755.2
## 2   4696 3632.2  2    123.03 79.53 < 2.2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Plot out polynomials
# But we need new dataframe with predictions of fit_order2 and fit_order4
temp_dataset <- data.frame(Ratings = seq(min(movies1$Rating), max(movies1$Rating), length.out = 1000))
# temp_predict <- predict(fit_order2, temp_dataset)
# temp_dataset$order2pred <- temp_predict
# temp_predict2 <- predict(fit_order4, temp_dataset)
# temp_dataset$order4pred <- temp_predict2

## try another
temp_dataset <- data.frame(Ratings = movies1$Rating, Order2fit = fit_order2$fitted.values, Order4fit = f


temp_dataset %>% ggplot(aes(x = Ratings, y = Order2fit)) + geom_line(col = "red") +
  theme_classic() +
  geom_point(data = movies, aes(x = Rating, y = Total_Gross), alpha = 0.1)
```

```
## Warning: Removed 861 rows containing missing values (`geom_point()`).
```



Weighted least squares

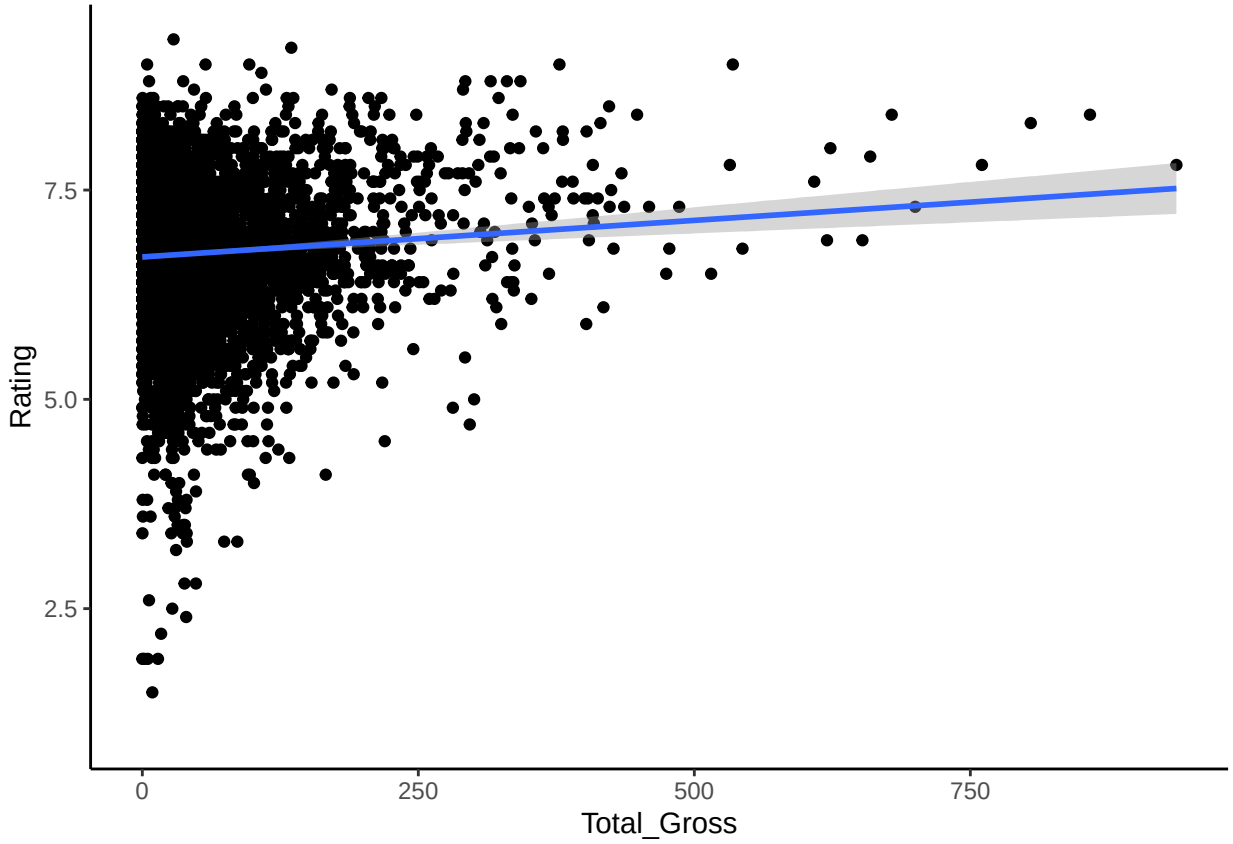
If we aggregate based on directors

```
movies %>% ggplot(aes(Total_Gross, Rating)) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

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

```
## Warning: Removed 861 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 861 rows containing missing values (`geom_point()`).
```

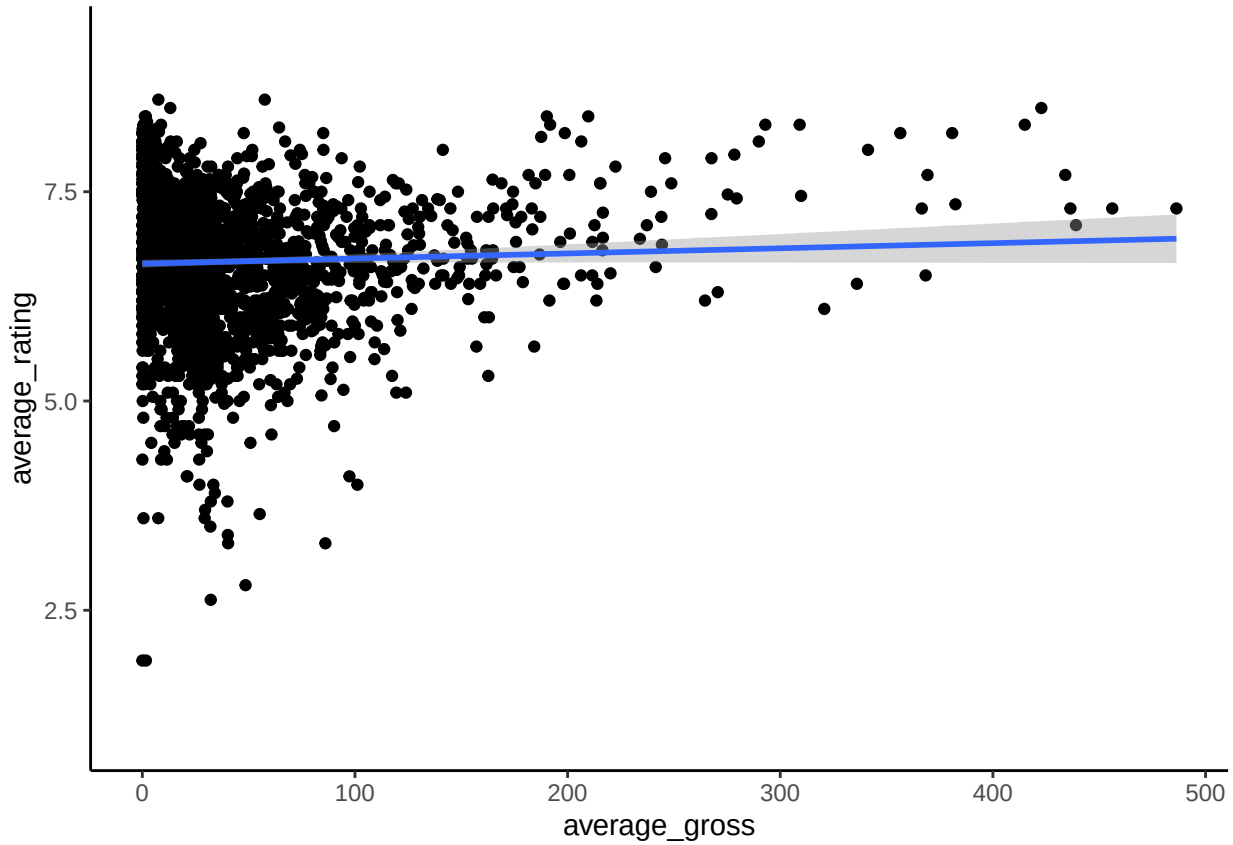



```
movies %>% group_by(Director) %>%
  summarise(average_gross = mean(Total_Gross, na.rm = T),
            average_rating = mean(Rating, na.rm = T)) %>%
  ggplot(aes(average_gross, average_rating)) + geom_point() +geom_smooth(method = 'lm')+
  theme_classic()
```

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

```
## Warning: Removed 399 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 399 rows containing missing values (`geom_point()`).
```



Predicting Ratings from other variables

```
names(movies)
```

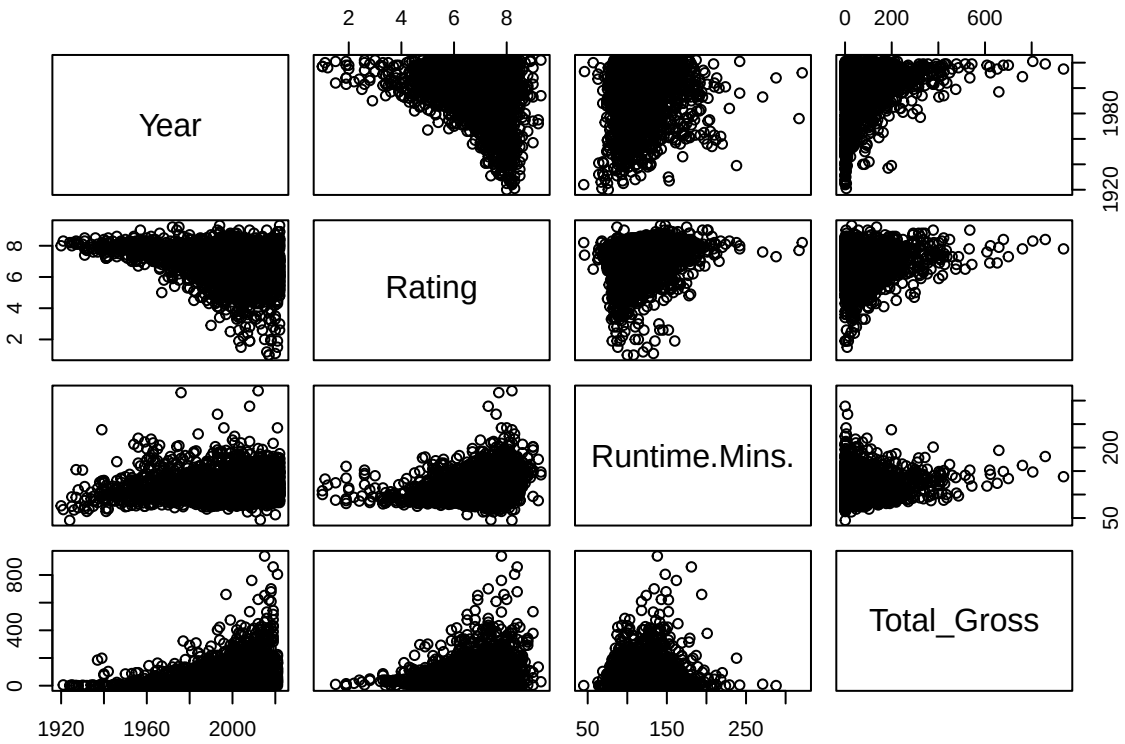
```
##  [1] "Movie_Title"   "Year"          "Director"      "Actors"
##  [5] "Rating"        "Runtime.Mins." "Censor"        "Total_Gross"
##  [9] "main_genre"    "side_genre"
```

```
head(movies)
```

```
##                                       Movie_Title Year          Director
## 1                                         Kantara 2022     Rishab Shetty
## 2                                The Dark Knight 2008 Christopher Nolan
## 3     The Lord of the Rings: The Return of the King 2003     Peter Jackson
## 4                                       Inception 2010 Christopher Nolan
## 5             The Lord of the Rings: The Two Towers 2002     Peter Jackson
## 6 The Lord of the Rings: The Fellowship of the Ring 2001     Peter Jackson
##                                                          Actors Rating
## 1     Rishab Shetty, Sapthami Gowda, Kishore Kumar G., Achyuth Kumar    9.3
## 2         Christian Bale, Heath Ledger, Aaron Eckhart, Michael Caine    9.0
## 3          Elijah Wood, Viggo Mortensen, Ian McKellen, Orlando Bloom    9.0
## 4 Leonardo DiCaprio, Joseph Gordon-Levitt, Elliot Page, Ken Watanabe    8.8
## 5          Elijah Wood, Ian McKellen, Viggo Mortensen, Orlando Bloom    8.8
## 6               Elijah Wood, Ian McKellen, Orlando Bloom, Sean Bean    8.8
```

```
##   Runtime.Mins. Censor Total_Gross main_genre          side_genre
## 1           148     UA          NA     Action   Adventure,  Drama
## 2           152     UA      534.86     Action       Crime,  Drama
## 3           201      U      377.85     Action   Adventure,  Drama
## 4           148     UA      292.58     Action  Adventure,  Sci-Fi
## 5           179     UA      342.55     Action   Adventure,  Drama
## 6           178      U      315.54     Action   Adventure,  Drama
```

```r
## Get only numeric variables
numeric_vars <- unlist(lapply(movies, is.numeric), use.names = F)
## Explore pairs plot - look for (linear) relationships
pairs(movies[numeric_vars]) # - likely to have issues with interpretation due to increasing variances -
```




```r
## MLR: NUMERIC ONLY
mlr1_num_only <- movies %>% select_if(is.numeric) %>% lm(Rating ~., data = .)
summary(mlr1_num_only)
```

```
##
## Call:
## lm(formula = Rating ~ ., data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2865 -0.4742  0.0545  0.5550  2.0513
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.3288755  1.6393221  22.771  < 2e-16 ***
## Year         -0.0161247  0.0008173 -19.729  < 2e-16 ***
```

```
## Runtime.Mins.  0.0149763  0.0005734  26.118  < 2e-16 ***
## Total_Gross    0.0005623  0.0001604   3.505  0.00046 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.807 on 4697 degrees of freedom
##   (861 observations deleted due to missingness)
## Multiple R-squared:  0.1963, Adjusted R-squared:  0.1958
## F-statistic: 382.5 on 3 and 4697 DF,  p-value: < 2.2e-16
```

```
## Check model!

#1. M.collinearity
vif(mlr1_num_only)
```

```
##          Year Runtime.Mins.   Total_Gross
##      1.014230      1.027379      1.039576
```

From the pairs plots, despite the increasing variance, the conditional means of Ratings given the other variables look to be linear wrt those other variables (Total gross, Runtime, even Year)

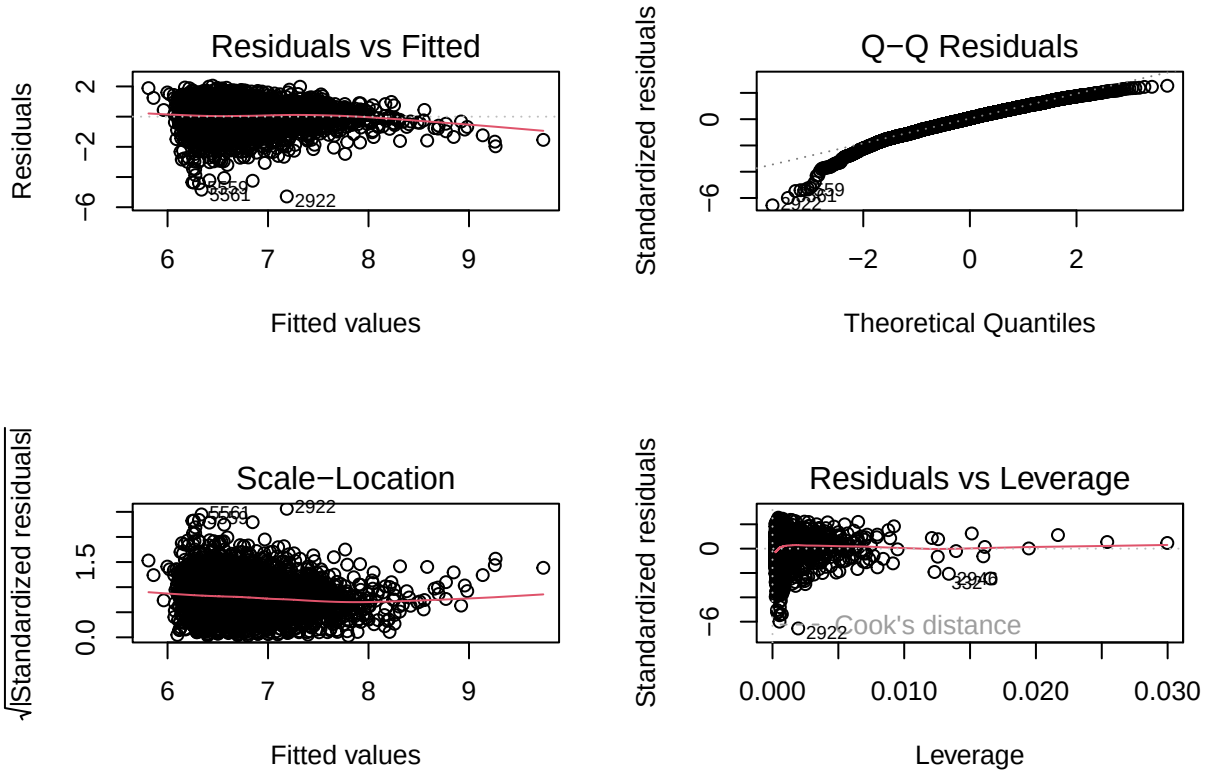
Checks for the numerical MLR model Multicollinearity - correlation *within* the covariates

However, from the same pairs plot, we can see that there is reasonable correlation within the numerical factors chosen. We use the Variance Inflation Factor to measure the extent of multicollinearity for each variable, the higher the VIF, the higher the correlation between the chosen variable and Ratings. (*From STHDA: VIF measures how much the variance of a regression coefficient i.e. slope estimate is inflated due to m.collinearity in the model*)

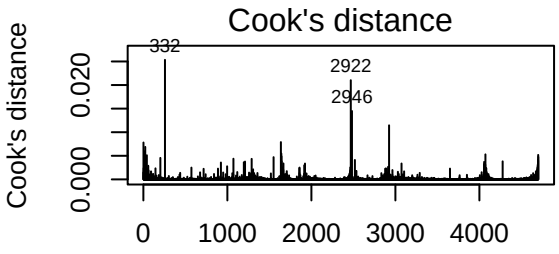
Since VIF is not high for the numerical covariates, m.collinearity is not of issue.

Diagnostic checks

```
#2. Diagnostic plots - check for assumptions of linearity, homoskedasticity
par(mfrow = c(2,2))
plot(mlr1_num_only)
```




```
plot(mlr1_num_only, 4)
```



####### Residual vs fitted

We ideally want the residuals to be randomly scattered about the horizontal line at 0, without any systematic pattern. This would show an averaging of 0 i.e. indication or approximation of expectation of the errors around 0. Here, we can see that for smaller values of the fitted values (y_hat), they mostly cluster around 0, but then begin to tail off downwards as fitted values get larger. Hence we are not capturing some systematic component of Ratings for higher values of Ratings, and therefore this is captured by the residual terms on the right hand side of the plot.

QQplots of standardised residuals

We can see that at both tail ends, our points dip below the straight line of equivalence between the sample and theorectical quantiles. This means that our sample quantiles are consistently smaller than that of the theorectical quantiles at both ends, and since the theoretical quantile represents the standardised normal distribution, this means that our residuals quantiles are smaller on both ends and this means that our residuals suffer from *fatter tails*.

Leverage and influence plots

We do suffer from quite a lot of outliers or extreme points that suffer from high standardised residuals (bigger than 3 in magnitude)

Cook's distance is a measurement of influence of points (points that generally are on the extreme end of the covaraite scales and have a large influence on the OLS estimated parameters of the slopes). We have a few points that have a high Cook's distance: points 332, 2922, 2946. c("Chi bi", "Memento", "Gettysburg"), c(2008, 2000, 1993), c("John Woo", "Christopher Nolan", "Ron Maxwell"), c("Tony Chiu-Wai Leung, Takeshi Kaneshiro, Fengyi Zhang, Chang Chen", "Guy Pearce, Carrie-Anne Moss, Joe Pantoliano, Mark Boone Junior", "Tom Berenger, Martin Sheen, Stephen Lang, Richard Jordan"), c(7.3, 8.4, 7.6), c(288, 113, 271), c(18, 21, 16), c(0.63, 25.54, 10.77), c(1, 12, 7), c(" Adventure, Drama", " Thriller", " History, War")

```
# ## vectors of all actors in the dataset
# actorlist <- unlist(strsplit(movies$Actors, ",")) ## split according to commas, but here we lose gran
# movies$Actors <- strsplit(movies$Actors, ",")
#
# movies$Actors <- str_squish(movies$Actors) ## remove whitespace from front
```