431 Class 12

Thomas E. Love, Ph.D.

2022-10-06

Today's Agenda

- Ingesting the favorite movies data
- Cleaning and Managing the data
- Addressing Your Exploratory Questions from the Class 11 Breakout

Today's Packages

```
 1  library(googlesheets4)
 2  library(broom)
 3  library(equatiomatic)
 4  library(ggrepel)
 5  library(ggridges)
 6  library(glue)
 7  library(mosaic)
 8  library(janitor); library(naniar); library(patchwork)
 9  library(tidyverse)
10
11  theme_set(theme_bw())
```

Ingesting the Data

Ingesting from our Google Sheet

```
gs4_deauth()

movies22 <-
  read_sheet("https://docs.google.com/spreadsheets/d/19aELXovpY3_7EdbjaBzMX
  select(film_id, film, year, length,
         imdb_ratings, imdb_stars, imdb_categories) |>
  mutate(film_id = as.character(film_id))

dim(movies22)
```

```
[1] 159   7
```

```
names(movies22)
```

```
[1] "film_id"         "film"            "year"              "length"
[5] "imdb_ratings"    "imdb_stars"      "imdb_categories"
```

The favorite movies data

```
1 movies22
```

```
# A tibble: 159 × 7
   film_id film                         year length imdb_ratings imdb_s…¹
imdb_…²
   <chr>   <chr>                       <dbl>  <dbl>        <dbl>    <dbl> <chr>
 1 1       8 1/2                        1963    138       113258       8   Drama
 2 2       2001: A Space Odyssey        1968    149       628220       8.3
Advent…
 3 3       About Elly                   2009    119        53523       7.9
Drama,…
 4 4       About Time                   2013    123       321525       7.8
Comedy…
 5 5       Avatar                       2009    162      1154273       7.8
Action…
 6 6       Avengers: Infinity War       2018    149       919813       8.4
Action…
```

Broad Summary

```
1 movies22 |> summary()
```

```
   film_id              film                year          length      
 Length:159         Length:159         Min.   :1942   Min.   : 90.0  
 Class :character   Class :character   1st Qu.:1995   1st Qu.:103.0  
 Mode  :character   Mode  :character   Median :2006   Median :117.0  
                                       Mean   :2002   Mean   :123.5  
                                       3rd Qu.:2012   3rd Qu.:136.5  
                                       Max.   :2022   Max.   :207.0  
  imdb_ratings       imdb_stars    imdb_categories   
 Min.   :      9   Min.   :3.600   Length:159         
 1st Qu.: 127066   1st Qu.:7.100   Class :character   
 Median : 289313   Median :7.800   Mode  :character   
 Mean   : 505421   Mean   :7.576                      
 3rd Qu.: 739100   3rd Qu.:8.150                      
 Max.   :2457003   Max.   :9.300                      
```

```
1 pct_complete_case(movies22)  ## from naniar
```

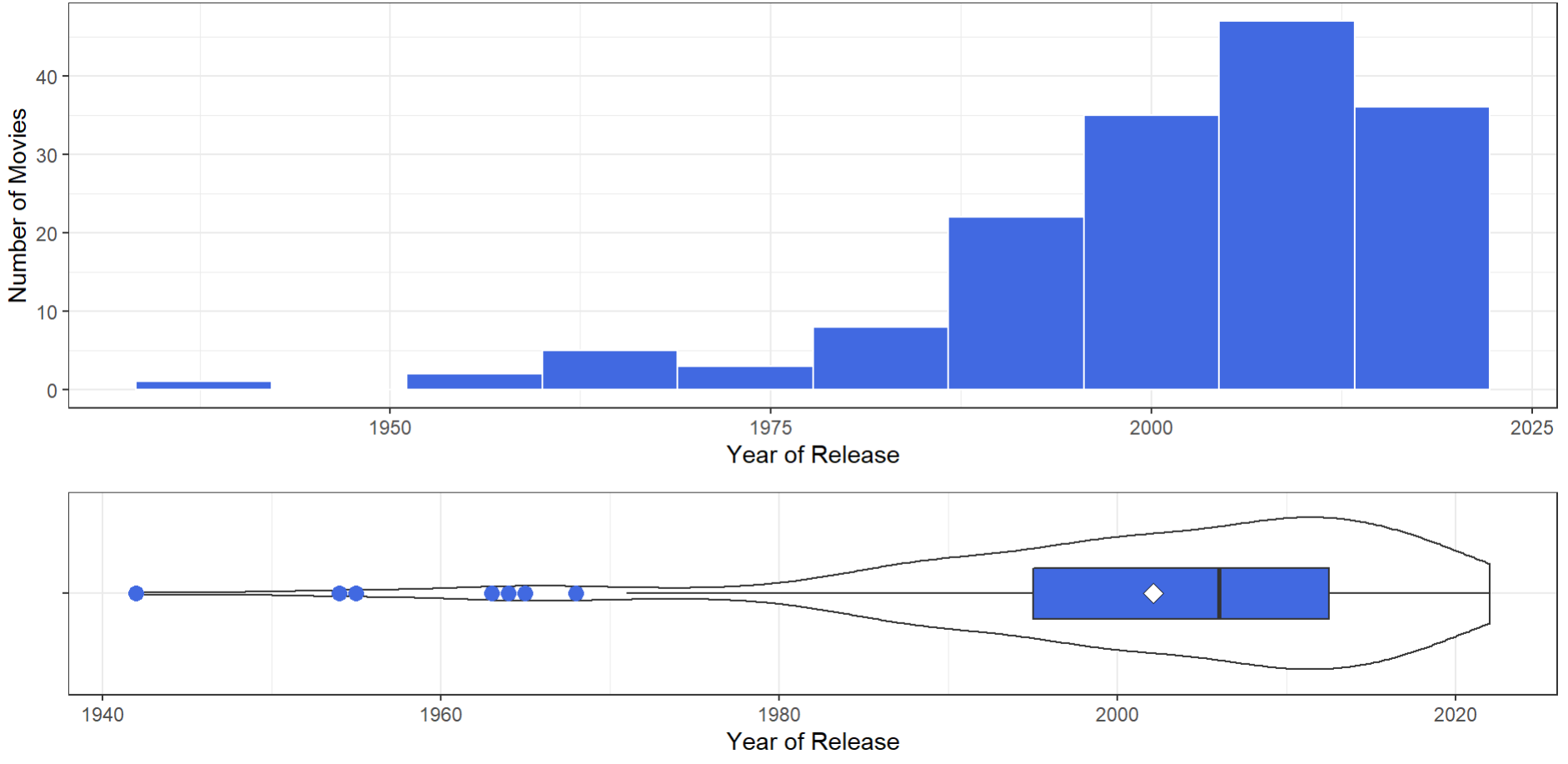
```
[1] 100
```

Exploring and Cleaning Data

Basic Exploration: year

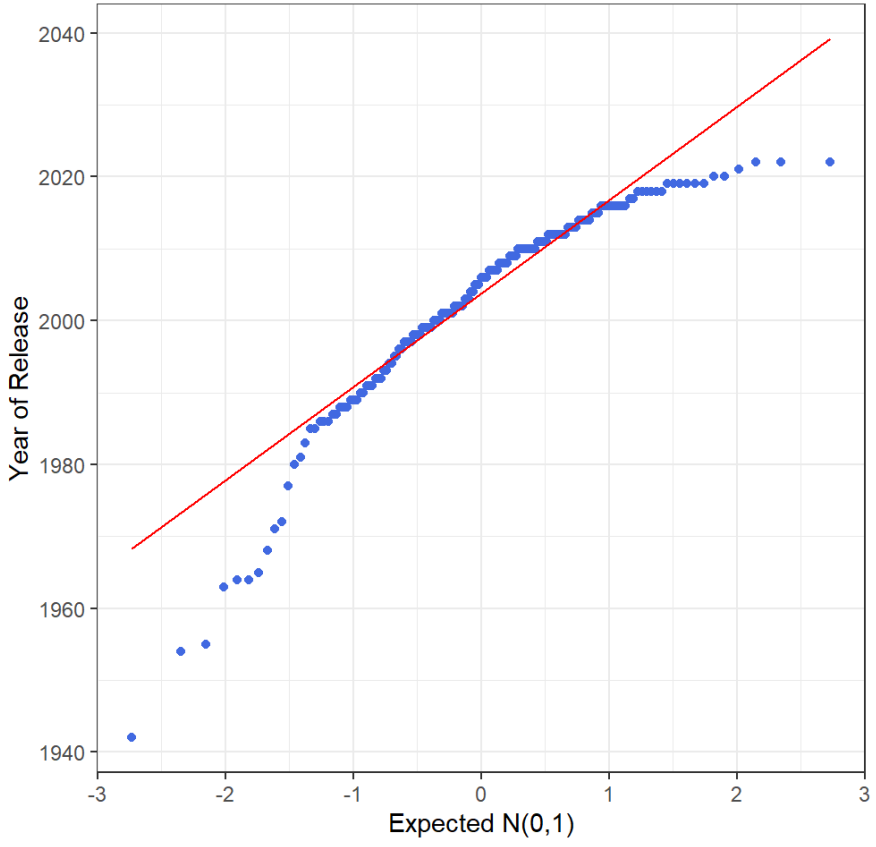
```
p1 <- ggplot(data = movies22, aes(x = year)) +
  geom_histogram(bins = 10, fill = "royalblue", col = "white") +
  labs(x = "Year of Release", y = "Number of Movies")

p2 <- ggplot(data = movies22, aes(x = year, y = "")) +
  geom_violin() +
  geom_boxplot(fill = "royalblue", width = 0.3,
               outlier.color = "royalblue", outlier.size = 3) +
  stat_summary(fun = "mean", geom = "point",
               shape = 23, size = 3, fill = "white") +
  labs(y = "", x = "Year of Release")

p1 / p2 + plot_layout(heights = c(2,1))
```

Basic Exploration: year



Normal Q-Q plot for year

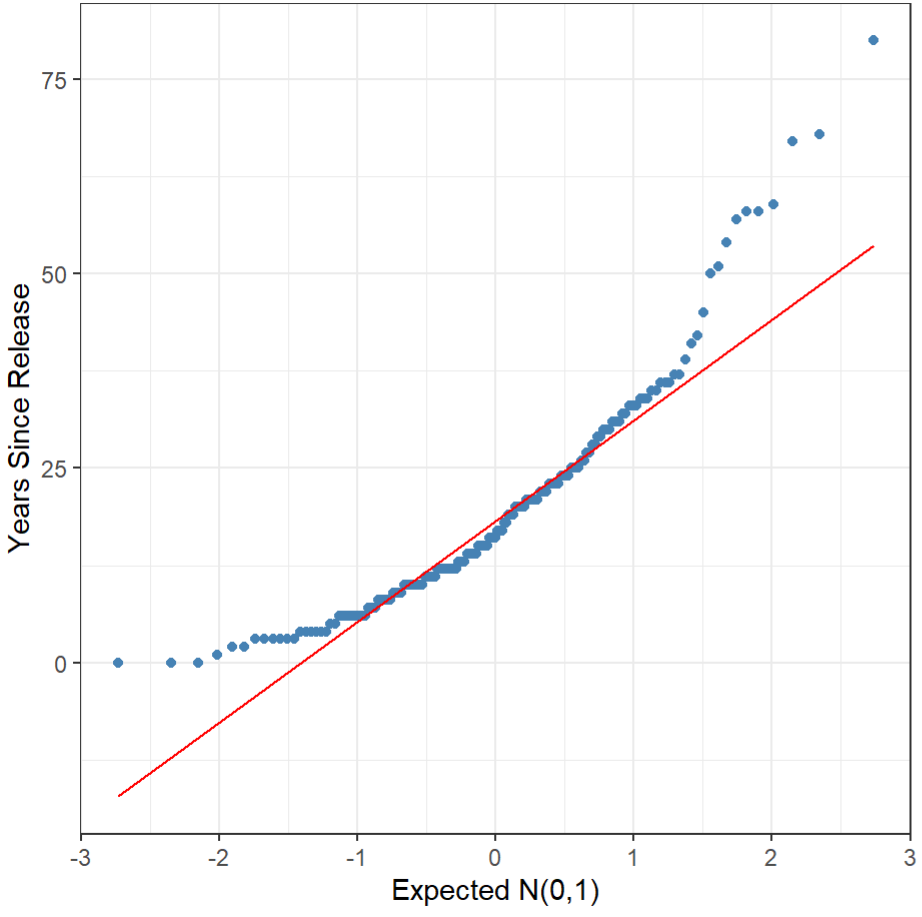
```
ggplot(data = movies22, aes(sample = year)) +
  geom_qq(col = "royalblue") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
  labs(x = "Expected N(0,1)", y = "Year of Release")
```

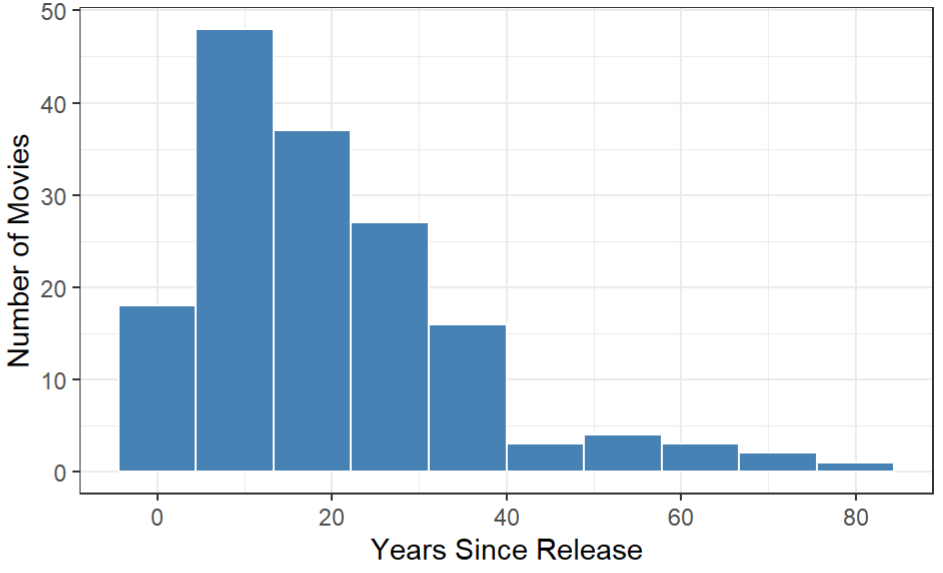


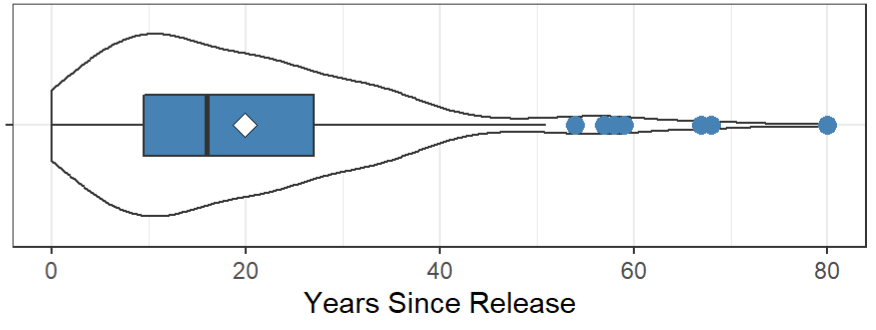
Consider age = 2022-year

```
movies22 <- movies22 |> mutate(age = 2022 - year)

p1 <- ggplot(data = movies22, aes(sample = age)) +
  geom_qq(col = "steelblue") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
  labs(x = "Expected N(0,1)", y = "Years Since Release")

p2 <- ggplot(data = movies22, aes(x = age)) +
  geom_histogram(bins = 10, fill = "steelblue", col = "white") +
  labs(x = "Years Since Release", y = "Number of Movies")

p3 <- ggplot(data = movies22, aes(x = age, y = "")) +
  geom_violin() +
  geom_boxplot(fill = "steelblue", width = 0.3,
               outlier.color = "steelblue", outlier.size = 3) +
  stat_summary(fun = "mean", geom = "point",
               shape = 23, size = 3, fill = "white") +
  labs(y = "", x = "Years Since Release")
```

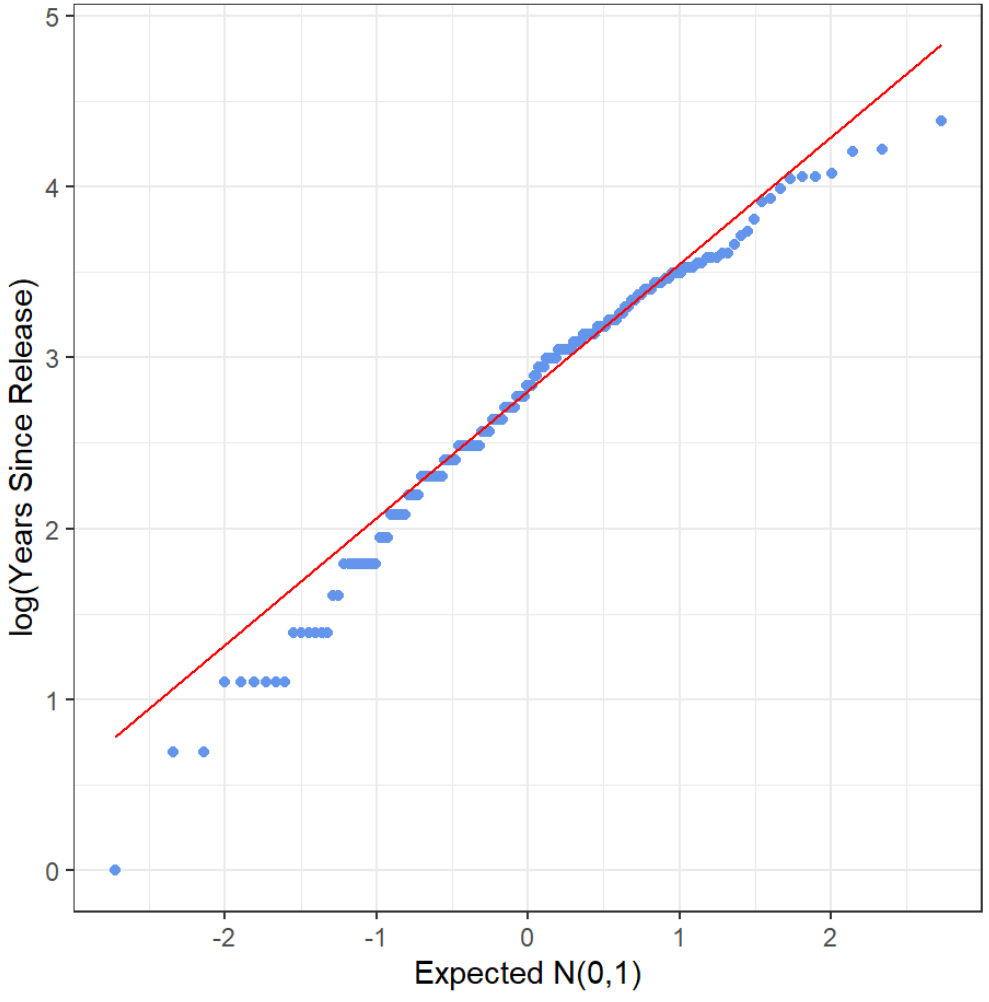
Consider age = 2022-year

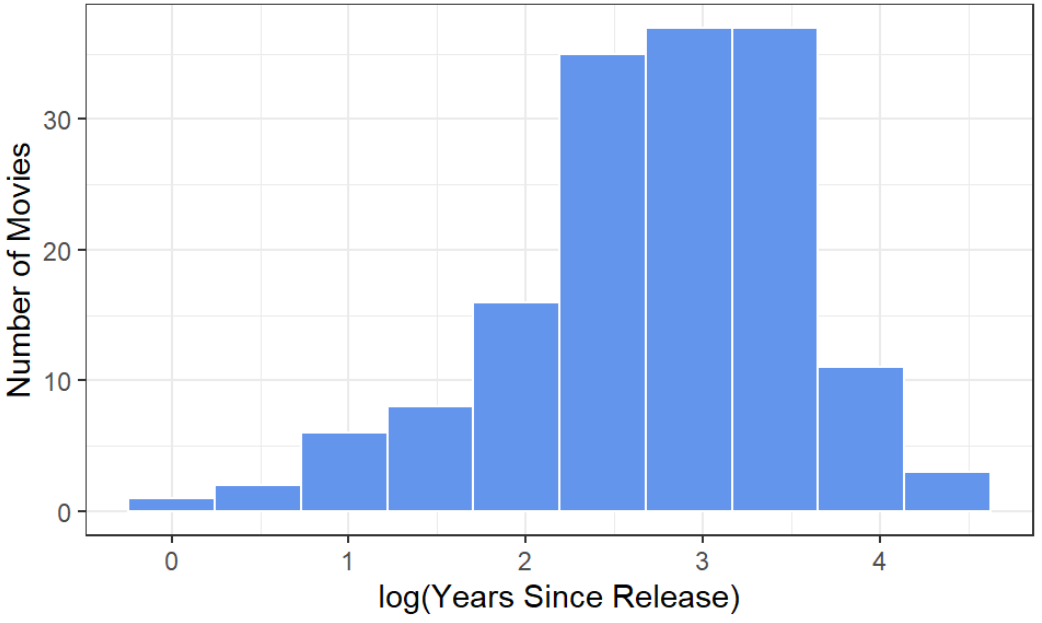


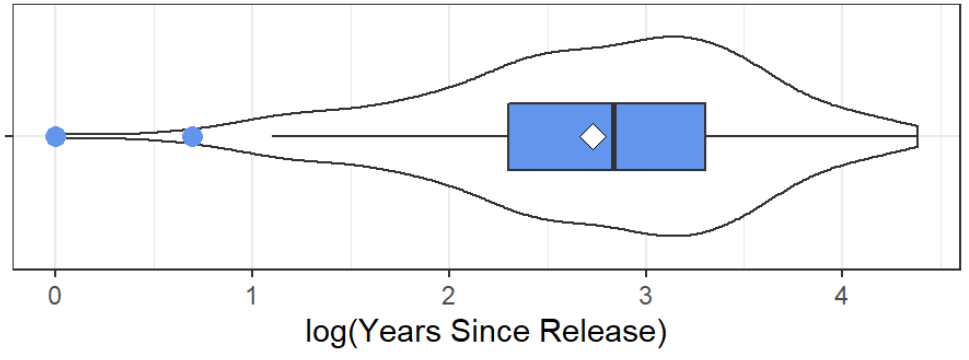




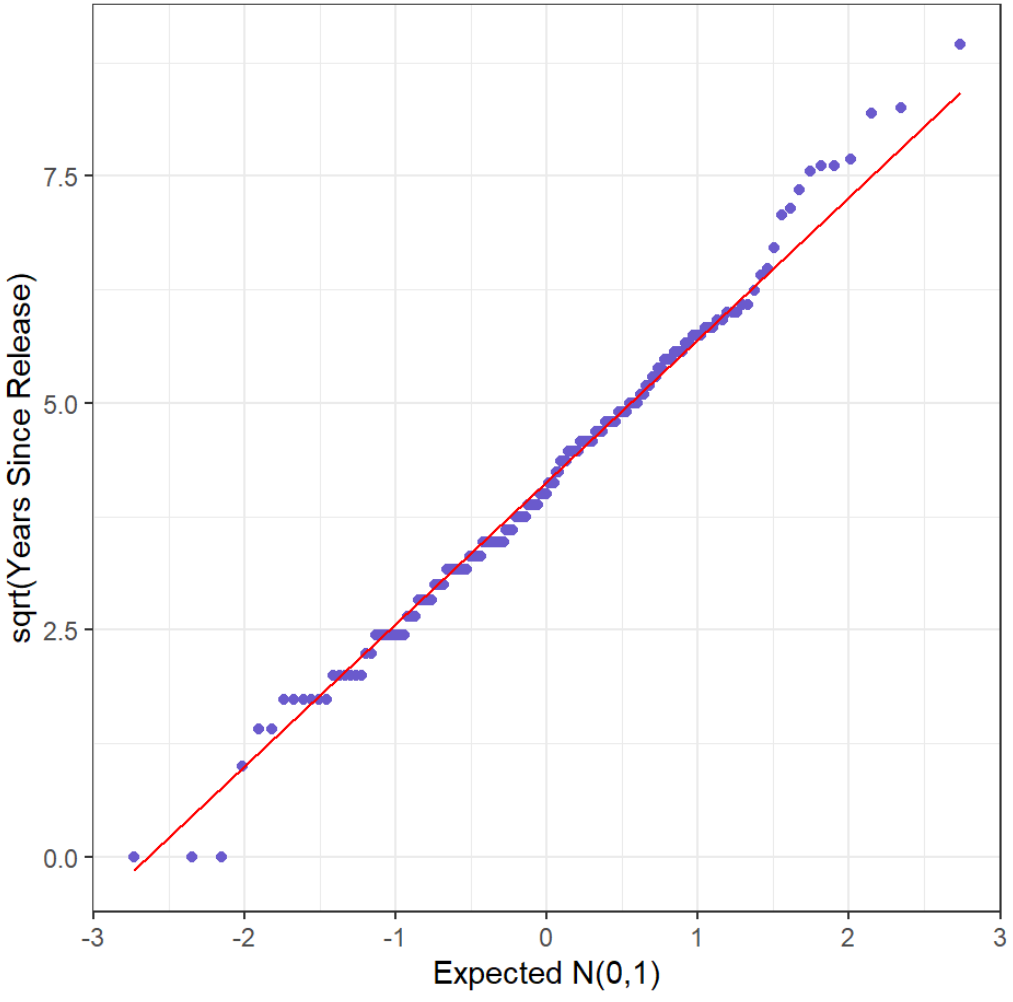
Consider \(log(age)\) = natural logarithm

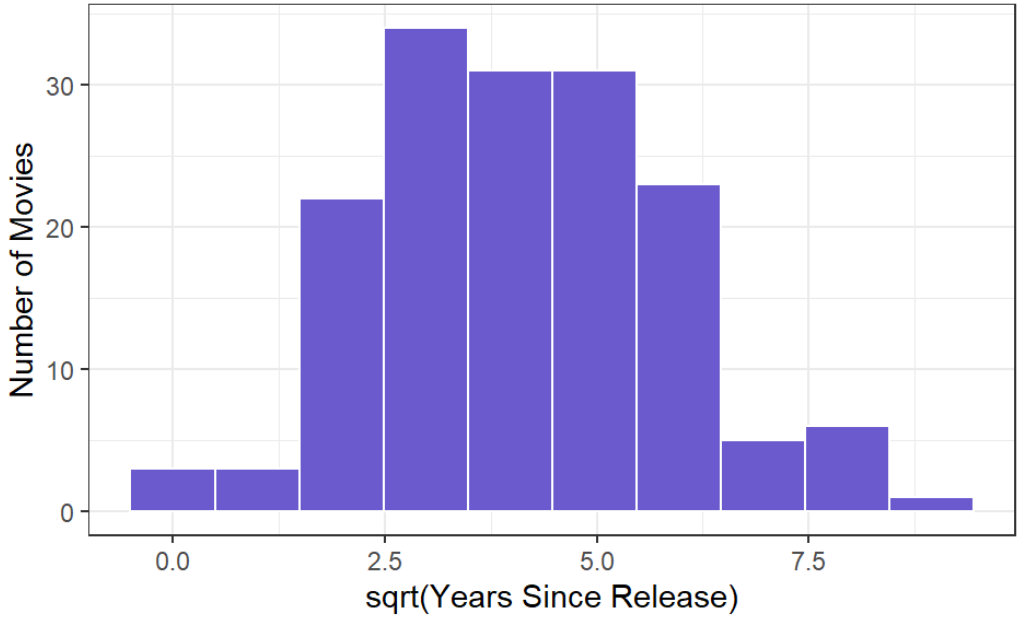


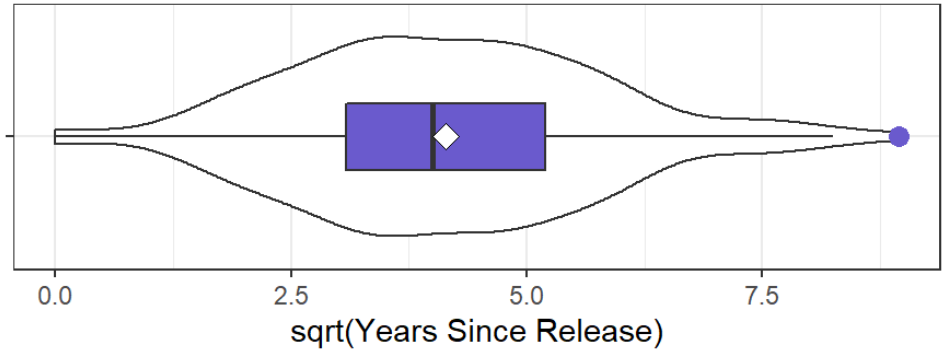




Consider \(\sqrt{age}\) = square root







Some Numerical Summaries for year

```
1 favstats(~ year, data = movies22)
```

```
  min   Q1 median     Q3  max     mean       sd   n missing
 1942 1995   2006 2012.5 2022 2002.101 14.87126 159       0
```

```
1 Hmisc::describe(movies22$year)
```

```
movies22$year
       n  missing distinct     Info     Mean      Gmd      .05      .10
     159        0       51    0.999     2002    15.83     1971     1986
     .25      .50      .75      .90      .95
    1995     2006     2012     2018     2019

lowest : 1942 1954 1955 1963 1964, highest: 2018 2019 2020 2021 2022
```

Additional Summaries for year

```
movies22 |> summarise(skew1 = (mean(year) - median(year))/sd(year))
```

```
# A tibble: 1 × 1
   skew1
   <dbl>
1 -0.262
```

```
movies22 |> count(year >= mean(year) - sd(year) &
                  year <= mean(year) + sd(year))
```

```
# A tibble: 2 × 2
  `year >= mean(year) - sd(year) & ...`     n
  <lgl>                                 <int>
1 FALSE                                    41
2 TRUE                                    118
```

```
118/159
```

```
[1] 0.7421384
```

Some Summaries for `sqrt(age)`

```
1 favstats(~ sqrt(age), data = movies22)
```

```
 min       Q1 median       Q3      max     mean       sd   n missing
   0 3.081139      4 5.196152 8.944272 4.140874 1.664318 159       0
```

```
1 Hmisc::describe(sqrt(movies22$age))
```

```
sqrt(movies22$age)
       n  missing distinct     Info     Mean      Gmd      .05      .10
     159        0       51    0.999    4.141    1.871    1.732    2.000
     .25      .50      .75      .90      .95
   3.081    4.000    5.196    6.017    7.162

lowest : 0.000000 1.000000 1.414214 1.732051 2.000000
highest: 7.615773 7.681146 8.185353 8.246211 8.944272
```

Additional Summaries for `sqrt(age)`

```
movies22 |>
  summarise(skew1 = (mean(sqrt(age)) - median(sqrt(age)))/sd(sqrt(age)))
```

```
# A tibble: 1 × 1
   skew1
   <dbl>
1 0.0846
```

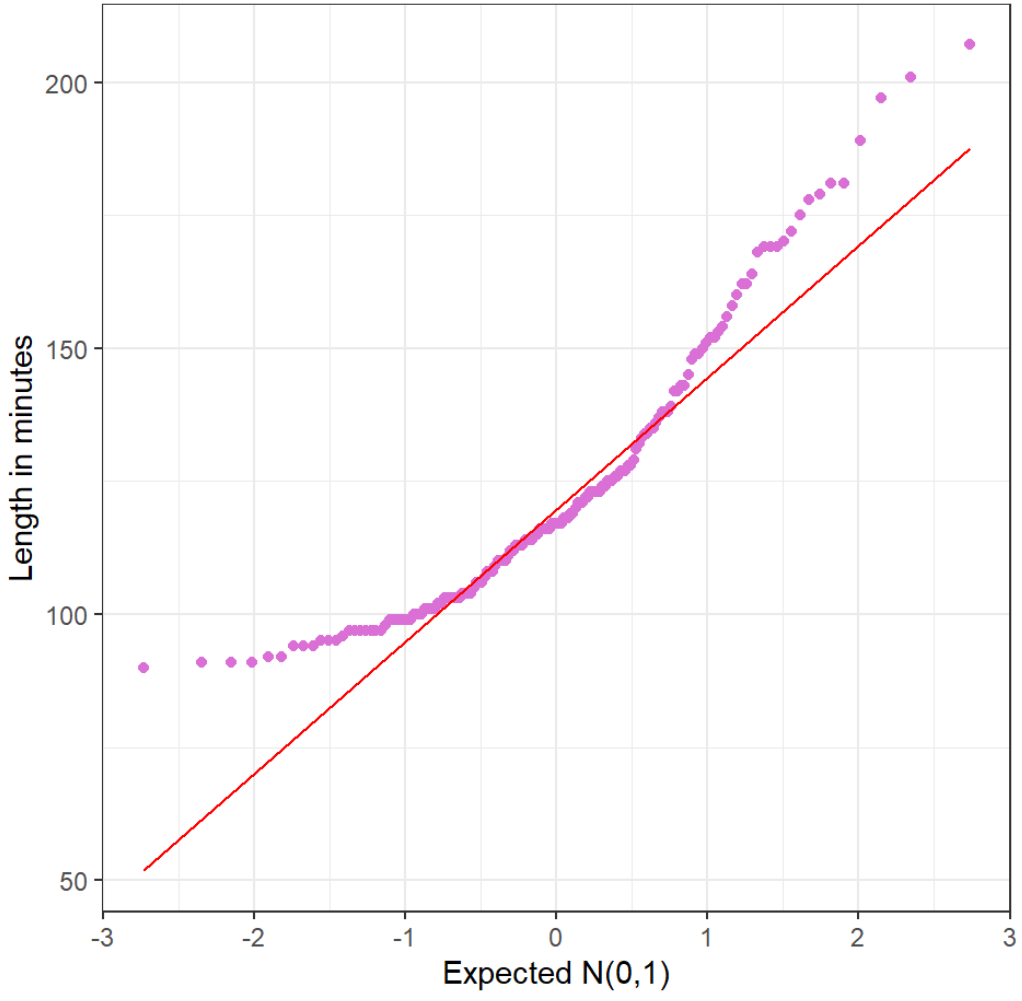
```
movies22 |> count(sqrt(age) >= mean(sqrt(age)) - sd(sqrt(age)) &
                    sqrt(age) <= mean(sqrt(age)) + sd(sqrt(age)))
```

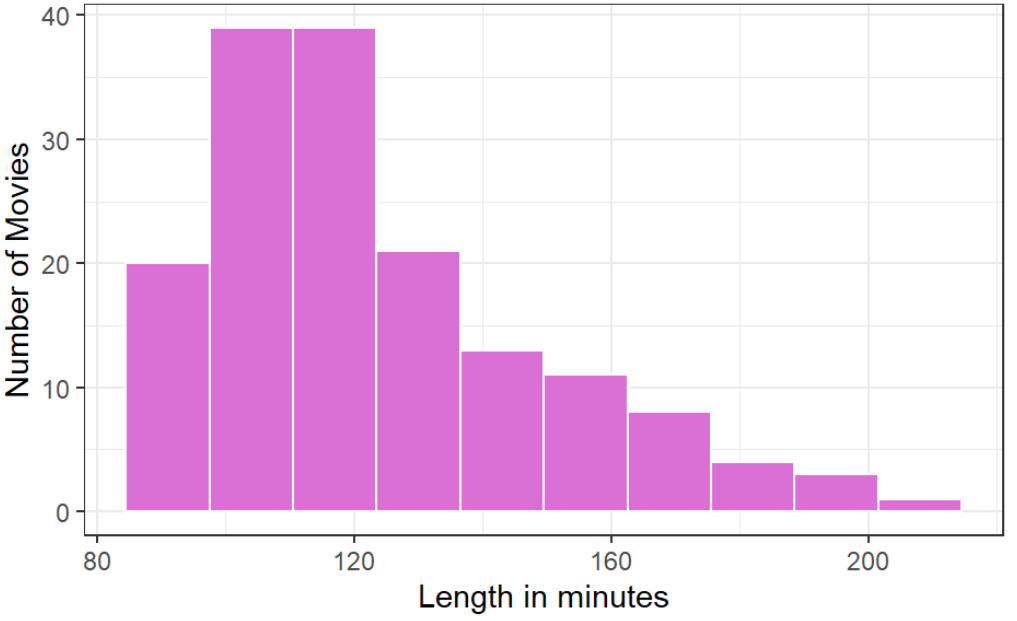
```
# A tibble: 2 × 2
  `sqrt(age) >= mean(sqrt(age)) - sd(sqrt(age)) & ...`     n
  <lgl>                                                <int>
1 FALSE                                                   52
2 TRUE                                                   107
```

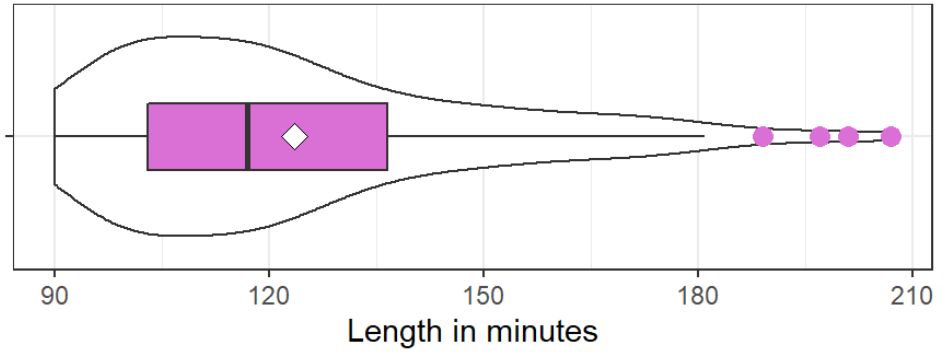
```
107/159
```

```
[1] 0.672956
```

Basic Exploration: `length`







Some Numerical Summaries for `length`

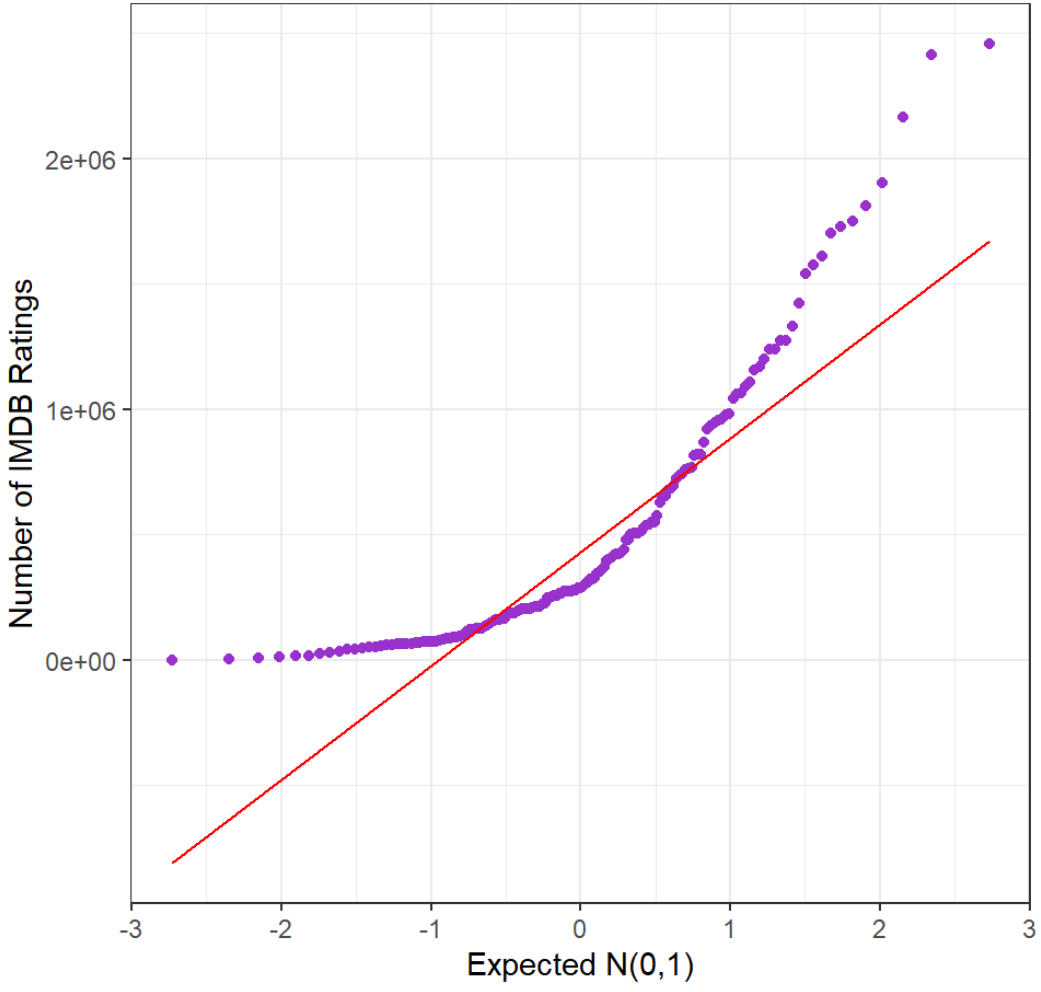
```
favstats(~ length, data = movies22)
```

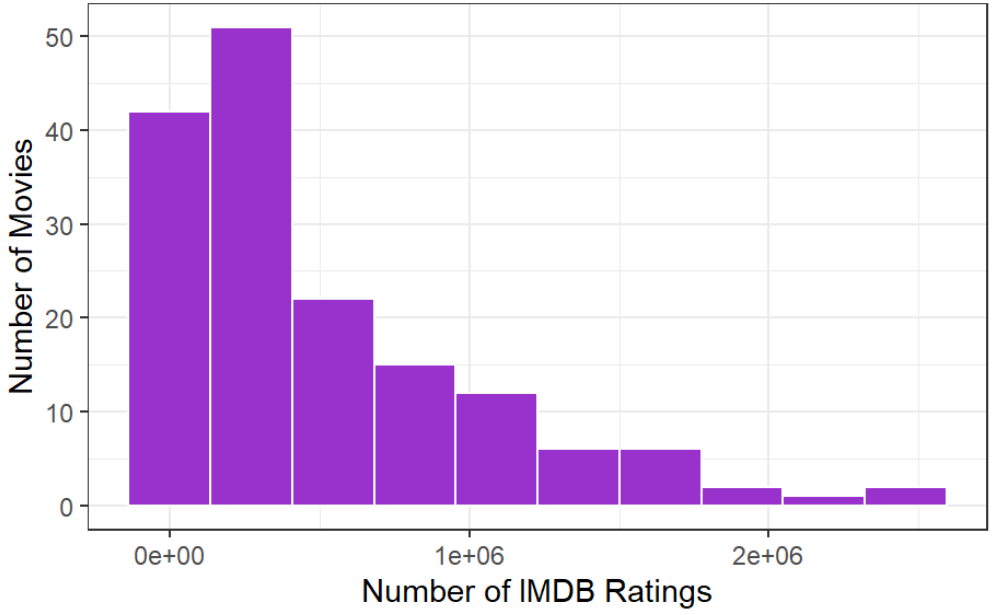
```
 min  Q1 median    Q3 max     mean       sd   n missing
  90 103    117 136.5 207 123.5157 25.74506 159       0
```

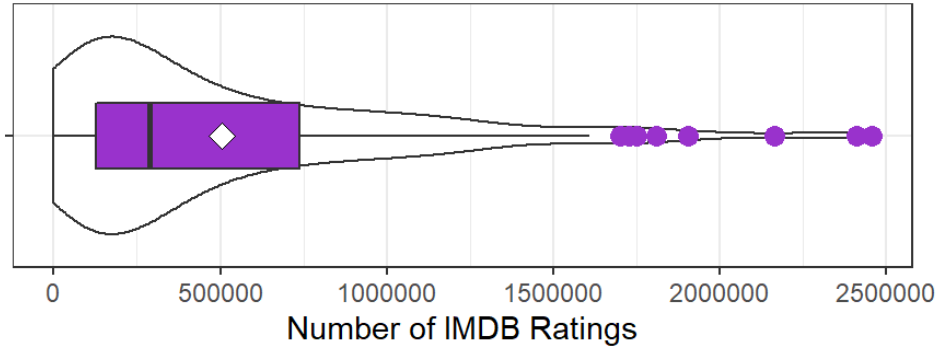
```
Hmisc::describe(movies22$length)
```

```
movies22$length
       n  missing distinct     Info     Mean      Gmd      .05      .10
     159        0       75        1    123.5    27.99     94.0     97.0
     .25      .50      .75      .90      .95
   103.0    117.0    136.5    162.4    175.3

lowest :  90  91  92  94  95, highest: 181 189 197 201 207
```

Basic Exploration: `imdb_ratings`







Some Summaries for `imdb_ratings`

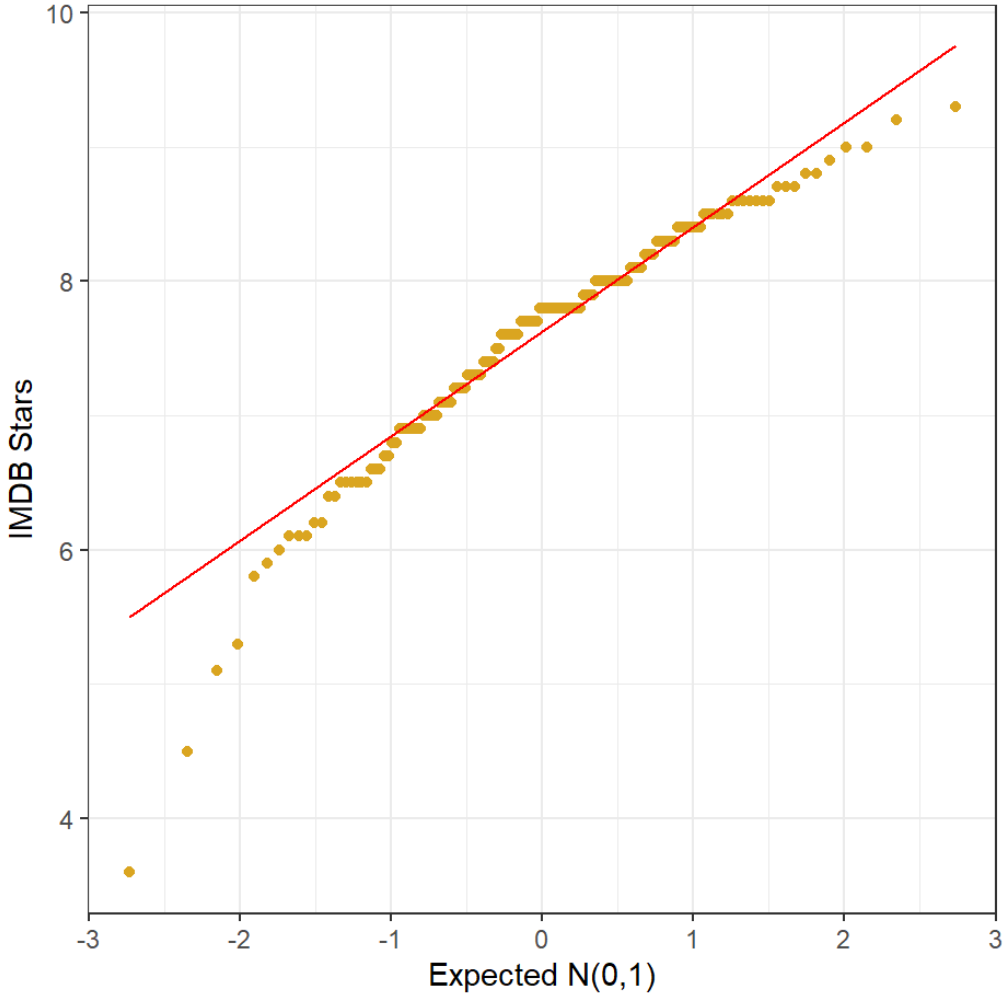
```
1 favstats(~ imdb_ratings, data = movies22)
```

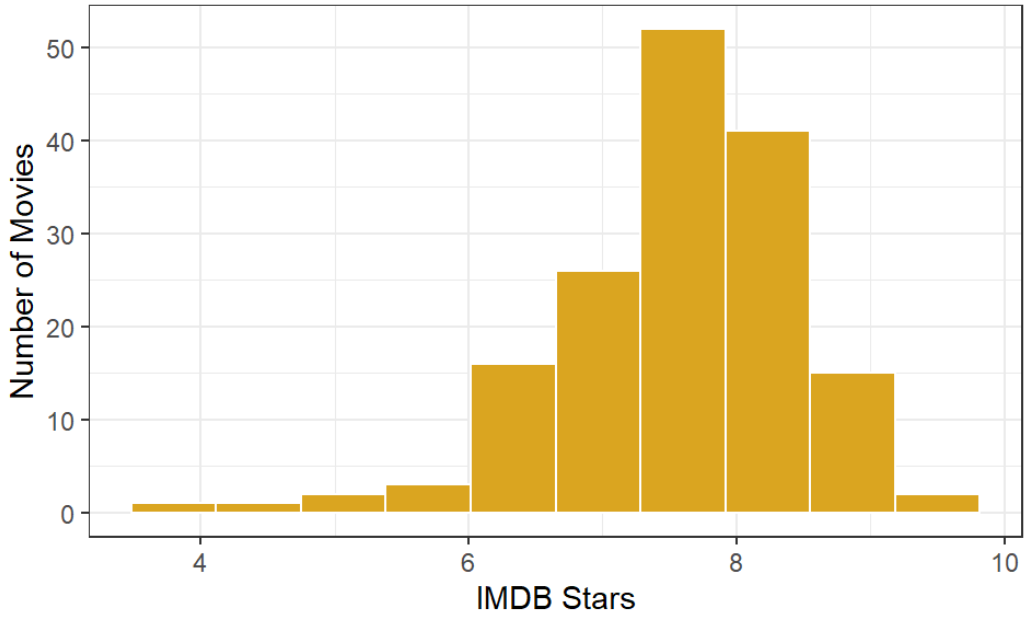
```
 min     Q1 median       Q3     max     mean       sd   n missing
   9 127066 289313 739099.5 2457003 505420.5 518816.7 159       0
```

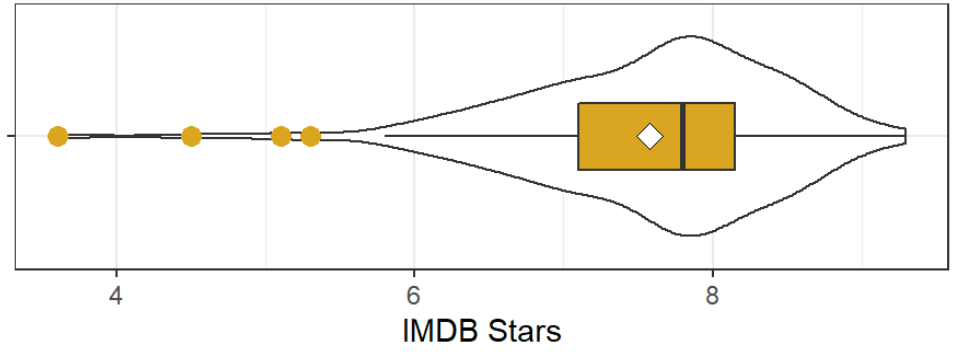
```
1 Hmisc::describe(movies22$imdb_ratings)
```

```
movies22$imdb_ratings
       n  missing distinct     Info     Mean      Gmd      .05      .10
     159        0      159        1   505420   531160    33035    60665
     .25      .50      .75      .90      .95
  127066   289313   739100  1237756  1618360

lowest :       9    3734    6309   11780   14712
highest: 1810834 1903850 2164457 2412730 2457003
```

Basic Exploration: `imdb_stars`







Some Summaries for `imdb_stars`

```
favstats(~ imdb_stars, data = movies22)
```

```
 min  Q1 median  Q3 max     mean       sd   n missing
 3.6 7.1    7.8 8.15 9.3 7.576101 0.883638 159       0
```

```
Hmisc::describe(movies22$imdb_stars)
```

```
movies22$imdb_stars
       n  missing distinct     Info     Mean      Gmd      .05      .10
     159        0       38    0.997    7.576   0.9548     6.10     6.50
     .25      .50      .75      .90      .95
    7.10     7.80     8.15     8.60     8.70

lowest : 3.6 4.5 5.1 5.3 5.8, highest: 8.8 8.9 9.0 9.2 9.3
```

What can we do with `imdb_categories`?

What is in `imdb_categories`?

```
1 movies22 |> tabyl(imdb_categories)
```

imdb_categories	n	percent
Action, Adventure	1	0.006289308
Action, Adventure, Comedy	3	0.018867925
Action, Adventure, Drama	6	0.037735849
Action, Adventure, Fantasy	4	0.025157233
Action, Adventure, Sci-Fi	8	0.050314465
Action, Adventure, Thriller	3	0.018867925
Action, Comedy, Fantasy	2	0.012578616
Action, Comedy, Mystery	1	0.006289308
Action, Crime, Drama	1	0.006289308
Action, Crime, Sci-Fi	1	0.006289308
Action, Crime, Thriller	2	0.012578616
Action, Drama	2	0.012578616
Action, Drama, Mystery	1	0.006289308
Action, Drama, Sci-Fi	1	0.006289308

Is `imdb_categories` useful?

```
1 movies22 |> tabyl(imdb_categories) |> arrange(-n) |> adorn_pct_formatting()
```

imdb_categories	n	percent
Drama	10	6.3%
Action, Adventure, Sci-Fi	8	5.0%
Animation, Adventure, Comedy	7	4.4%
Comedy, Drama, Romance	7	4.4%
Drama, Romance	7	4.4%
Action, Adventure, Drama	6	3.8%
Crime, Drama, Thriller	6	3.8%
Comedy, Drama	5	3.1%
Action, Adventure, Fantasy	4	2.5%
Comedy, Drama, Fantasy	4	2.5%
Action, Adventure, Comedy	3	1.9%
Action, Adventure, Thriller	3	1.9%
Comedy	3	1.9%
Comedy, Drama, Music	3	1.9%

Split into separate columns?

- Each movie has up to three categories identified in `imdb_categories`.
- There are 18 different categories represented across our 159 movies.

```
1 str_split_fixed(movies22$imdb_categories, ", ", n = 3) |> head()
```

```
     [,1]        [,2]        [,3]
[1,] "Drama"     ""          ""
[2,] "Adventure" "Sci-Fi"    ""
[3,] "Drama"     "Mystery"   ""
[4,] "Comedy"    "Drama"     "Fantasy"
[5,] "Action"    "Adventure" "Fantasy"
[6,] "Action"    "Adventure" "Sci-Fi"
```

Can we create an indicator for Action?

We want:

- a variable which is 1 if the movie's `imdb_categories` list includes Action and 0 otherwise
- and we'll call it `action`.

```
1 movies22 <- movies22 |>
2   mutate(action = as.numeric(str_detect(imdb_categories, fixed("Action"))))
```

Check our coding?

```
  1 movies22 |> select(film_id, film, imdb_categories, action) |> slice(128:137
```

```
# A tibble: 10 × 4
   film_id film                          imdb_categories
action
   <chr>   <chr>                         <chr>
<dbl>
 1 128     Seven Psychopaths             Comedy, Crime
0
 2 129     Seven Samurai                 Action, Drama
1
 3 130     The Shawshank Redemption      Drama
0
 4 131     The Silence of the Lambs      Crime, Drama, Thriller
0
 5 132     Skyfall                       Action, Adventure, Thriller
1
 6 133     Scrum To Bother You           Comedy, Drama, Fantasy
```

How many "Action" movies?

```
1 movies22 |> tabyl(action)
```

```
action   n   percent
     0 119 0.7484277
     1  40 0.2515723
```

OK. We need to do this for all 18 of the genres specified in `imdb_categories`.

Indicators of All 18 Genres

```
movies22 <- movies22 |>
  mutate(action = as.numeric(str_detect(imdb_categories, fixed("Action"))),
         adventure = as.numeric(str_detect(imdb_categories, fixed("Adventur
         animation = as.numeric(str_detect(imdb_categories, fixed("Animatio
         biography = as.numeric(str_detect(imdb_categories, fixed("Biograph
         comedy = as.numeric(str_detect(imdb_categories, fixed("Comedy"))),
         crime = as.numeric(str_detect(imdb_categories, fixed("Crime"))),
         drama = as.numeric(str_detect(imdb_categories, fixed("Drama"))),
         family = as.numeric(str_detect(imdb_categories, fixed("Family"))),
         fantasy = as.numeric(str_detect(imdb_categories, fixed("Fantasy"))
         horror = as.numeric(str_detect(imdb_categories, fixed("Horror"))),
         music = as.numeric(str_detect(imdb_categories, fixed("Music"))),
         musical = as.numeric(str_detect(imdb_categories, fixed("Musical"))
         romance = as.numeric(str_detect(imdb_categories, fixed("Romance"))
         scifi = as.numeric(str_detect(imdb_categories, fixed("Sci-Fi"))),
         sport = as.numeric(str_detect(imdb_categories, fixed("Sport"))),
         thriller = as.numeric(str_detect(imdb_categories, fixed("Thriller"
         war = as.numeric(str_detect(imdb_categories, fixed("War"))),
         western = as.numeric(str_detect(imdb_categories, fixed("Western"))
```

Summing Up Genres, Horizontally

```
movies22 |>
  summarise(across(.cols = action:western, .fns = sum))
```

```
# A tibble: 1 × 18
  action advent…¹ anima…² biogr…³ comedy crime drama family fantasy horror
music
   <dbl>    <dbl>   <dbl>   <dbl>  <dbl> <dbl> <dbl>  <dbl>   <dbl>  <dbl>
<dbl>
1     40       50      12       7     56    20    95     12       21      5
13
# … with 7 more variables: musical <dbl>, romance <dbl>, scifi <dbl>,
#   sport <dbl>, thriller <dbl>, war <dbl>, western <dbl>, and abbreviated
#   variable names ¹adventure, ²animation, ³biography
```

Sorted Counts of Movies by Genre

```
1 movies22 |>
2   summarise(across(.cols = action:western, .fns = sum)) |>
3   t() |> as.data.frame() |> rename(count = V1) |> arrange(-count)
```

```
          count
drama        95
comedy       56
adventure    50
action       40
romance      25
fantasy      21
scifi        21
crime        20
thriller     19
music        13
animation    12
family       12
biography     7
musical       6
horror        5
```

First Exploration from Class 11 breakout

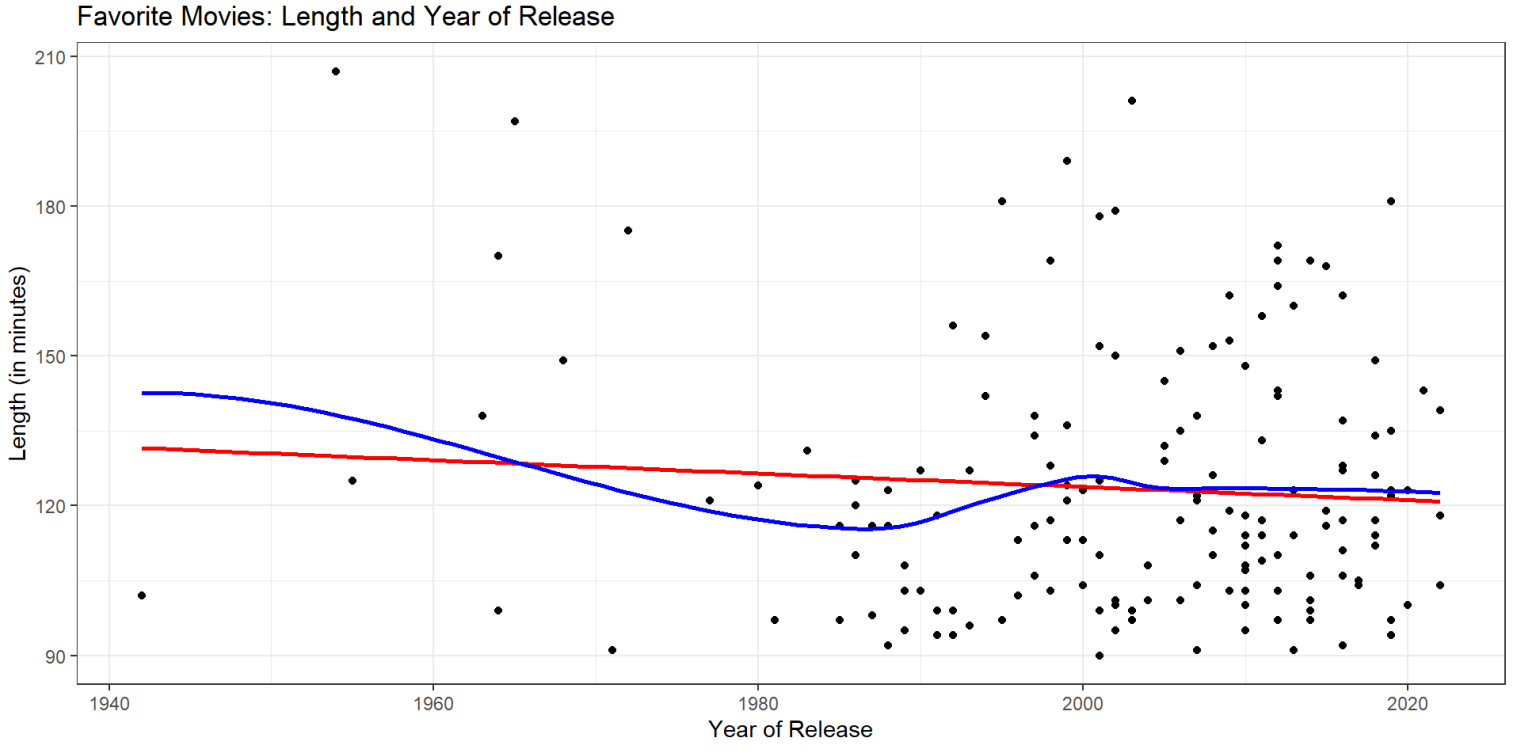
Questions about `year` and `length`

- Has the length of movies changed over time?
- Are new movies longer in length?
- Do movies released in 2000 or later have a longer run time than older movies?
- Are movies made prior to 2000 longer or shorter than movies after 2000?
- How has action movies' length changed over time?

We'll start by plotting the association of `year` and `length`.

Movie Lengths, over Time (ver. 1)

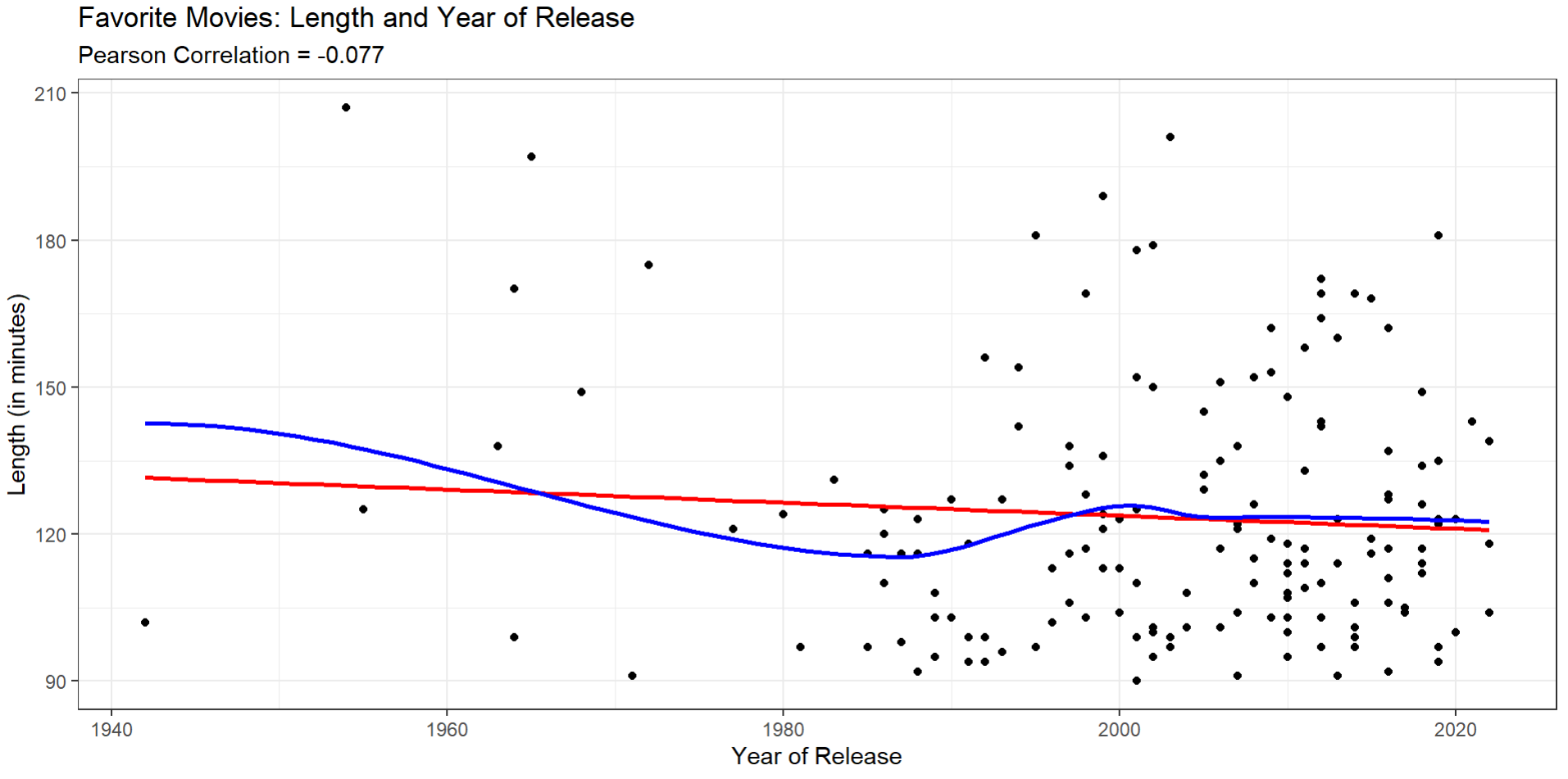
```
ggplot(movies22, aes(x = year, y = length)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release")
```



Add the correlation in a subtitle

```
ggplot(movies22, aes(x = year, y = length)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

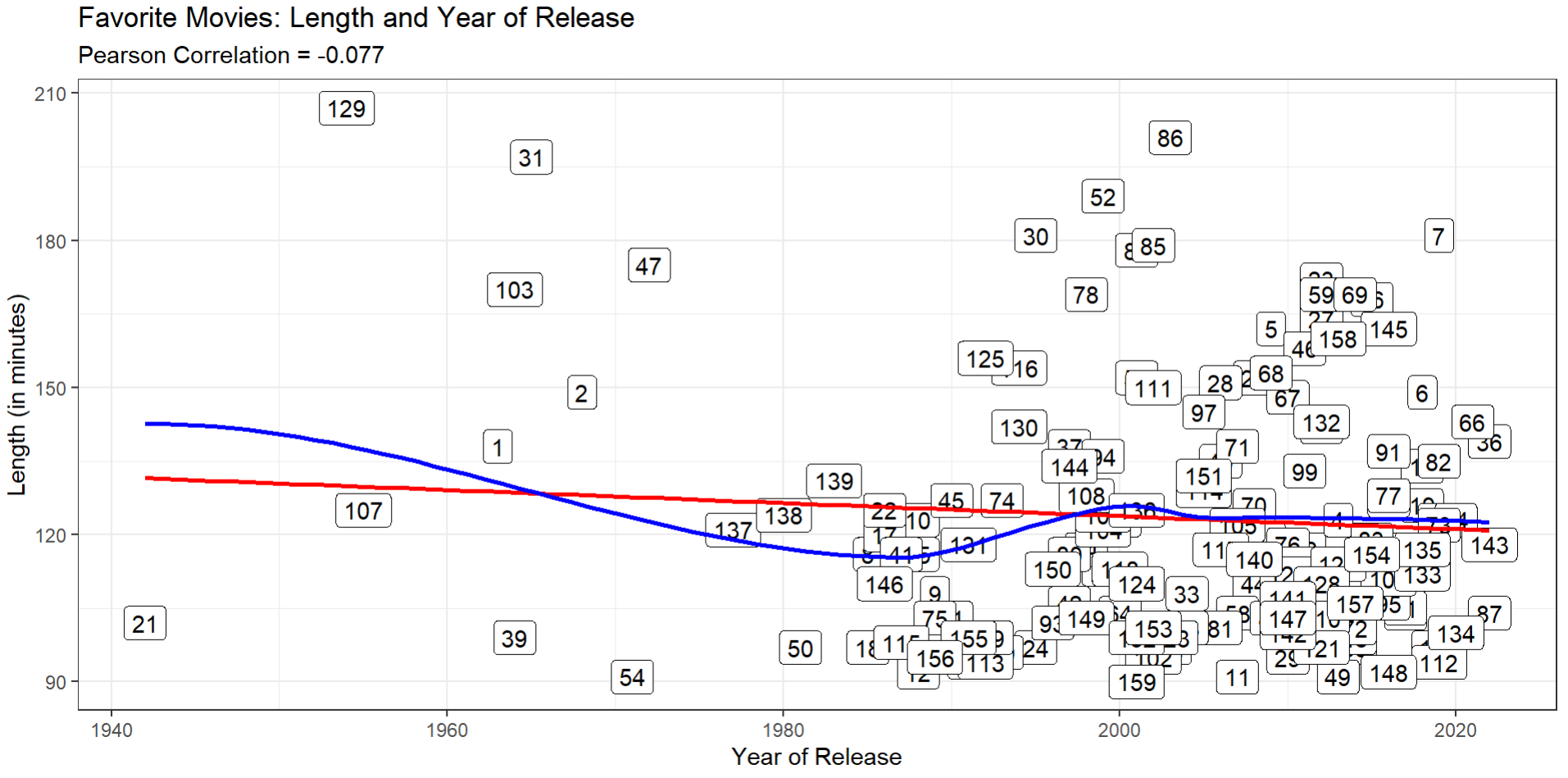
Add the correlation in a subtitle



Use `film_id` labels instead of points

```
ggplot(movies22, aes(x = year, y = length, label = film_id)) +
  geom_label() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

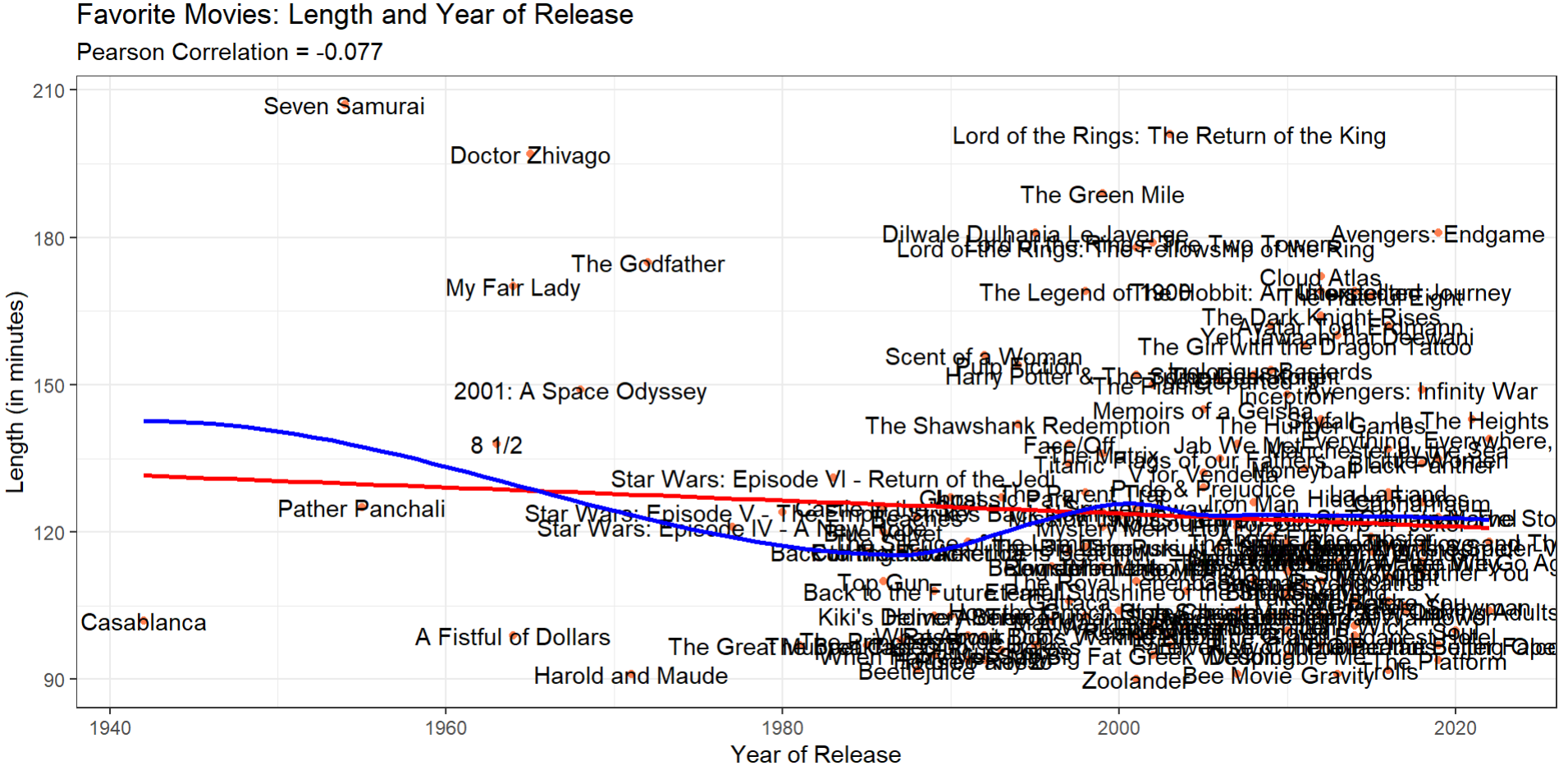
Use `film_id` labels instead of points



Use text to show `film` names

```
ggplot(movies22, aes(x = year, y = length, label = film)) +
  geom_point(col = "coral") +
  geom_text() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

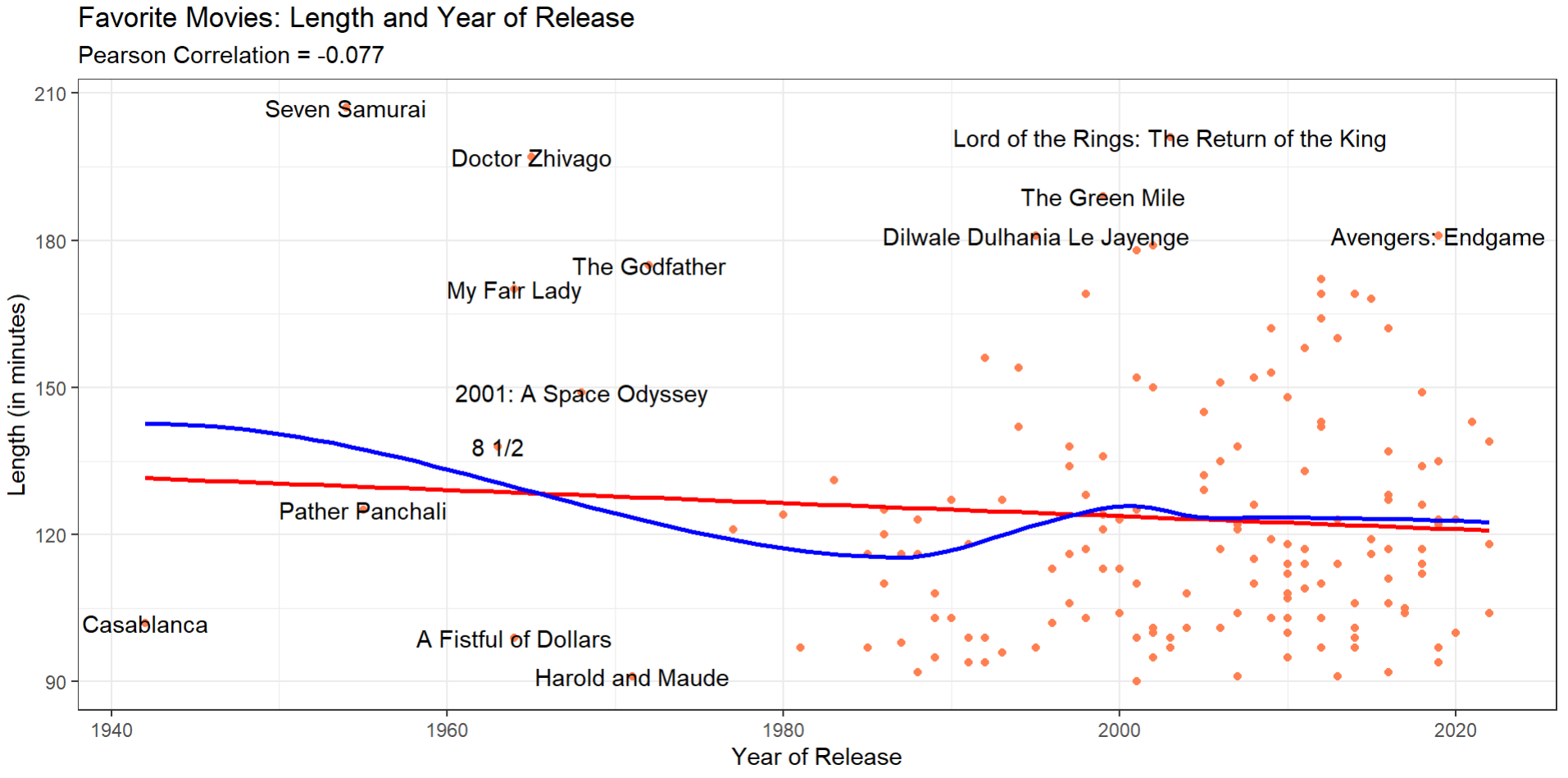
Use text to show `film` names



Show `film` text for selected movies

```
ggplot(movies22, aes(x = year, y = length, label = film)) +
  geom_point(col = "coral") +
  geom_text(data = movies22 |> filter(year < 1975 | length > 180)) +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

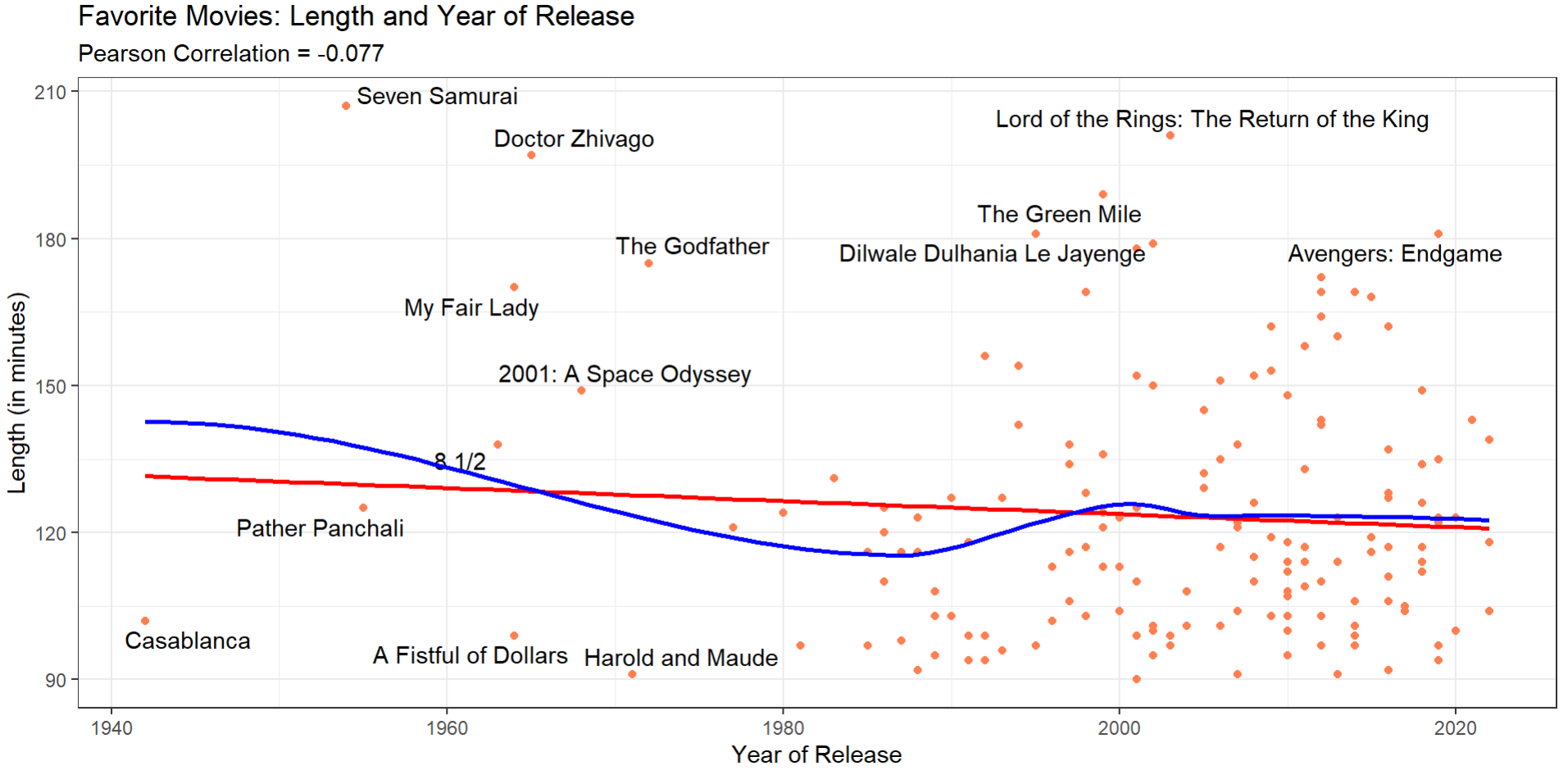
Show `film` text for selected movies



Try `geom_text_repel()`

```
ggplot(movies22, aes(x = year, y = length, label = film)) +
  geom_point(col = "coral") +
  geom_text_repel(data = movies22 |> filter(year < 1975 | length > 180)) +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

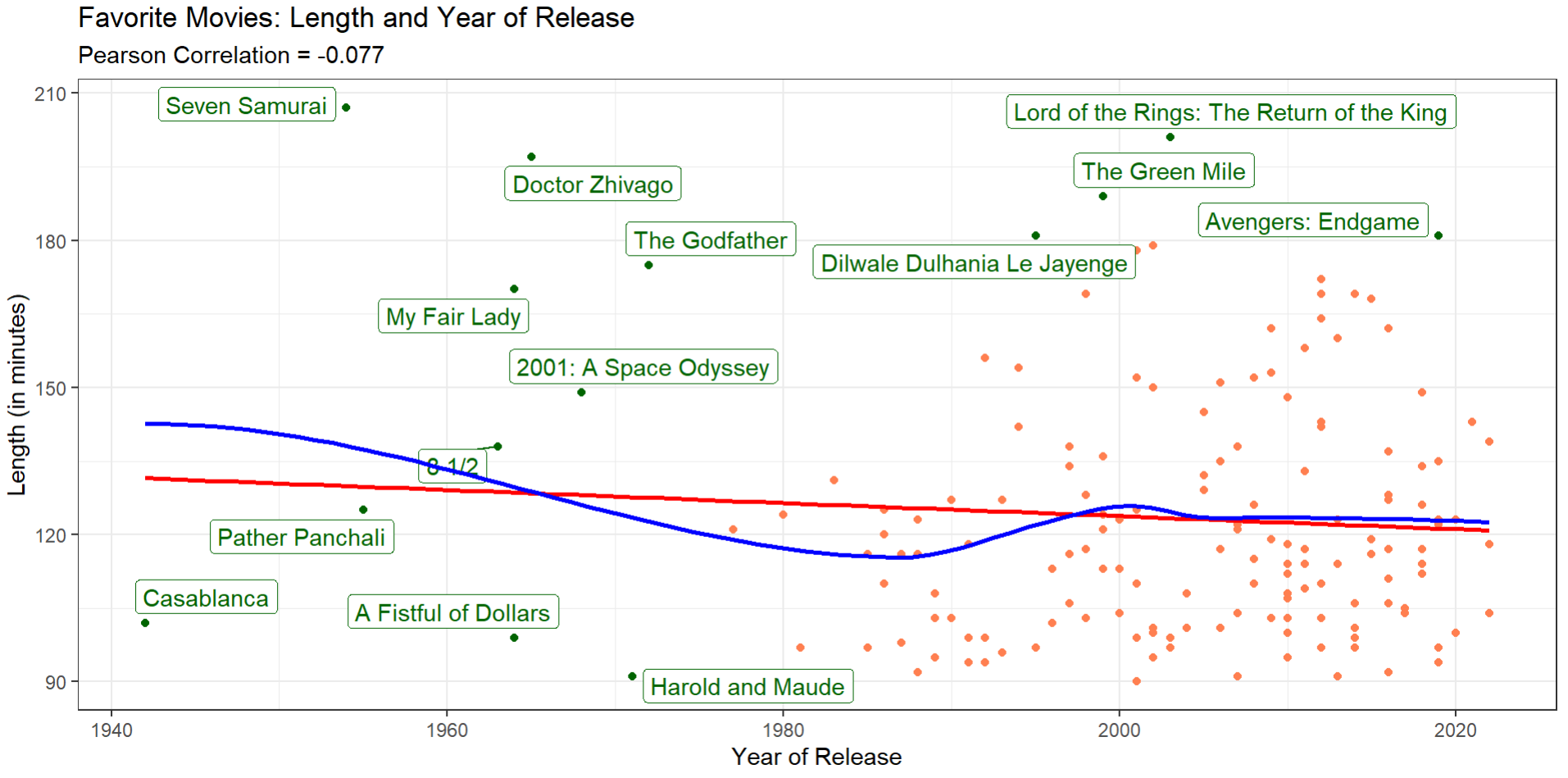
Try `geom_text_repel()`



geom_label_repel and colors?

```
ggplot(movies22, aes(x = year, y = length, label = film)) +
  geom_point(col = "coral") +
  geom_point(data = movies22 |> filter(year < 1975 | length > 180),
             color = "darkgreen") +
  geom_label_repel(data = movies22 |> filter(year < 1975 | length > 180),
                   color = "darkgreen") +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  geom_smooth(method = "loess", se = F, formula = y ~ x, col = "blue") +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$length),3)))
```

`geom_label_repel` and colors?



Model for Length, using Year?

```
m1 <- lm(length ~ year, data = movies22)
extract_eq(m1, use_coefs = TRUE, wrap = TRUE, operator_location = "start",
           terms_per_line = 2)
```

$$\begin{aligned} \operatorname{\widehat{length}} &= 390.54 - 0.13(\operatorname{year}) \end{aligned}$$

```
tidy(m1, conf.int = TRUE, conf.level = 0.90)
```

```
# A tibble: 2 × 7
  term        estimate std.error statistic p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>   <dbl>    <dbl>     <dbl>
1 (Intercept)  391.      276.         1.42    0.159  -65.8     847.
2 year          -0.133     0.138    -0.968   0.334   -0.361     0.0946
```

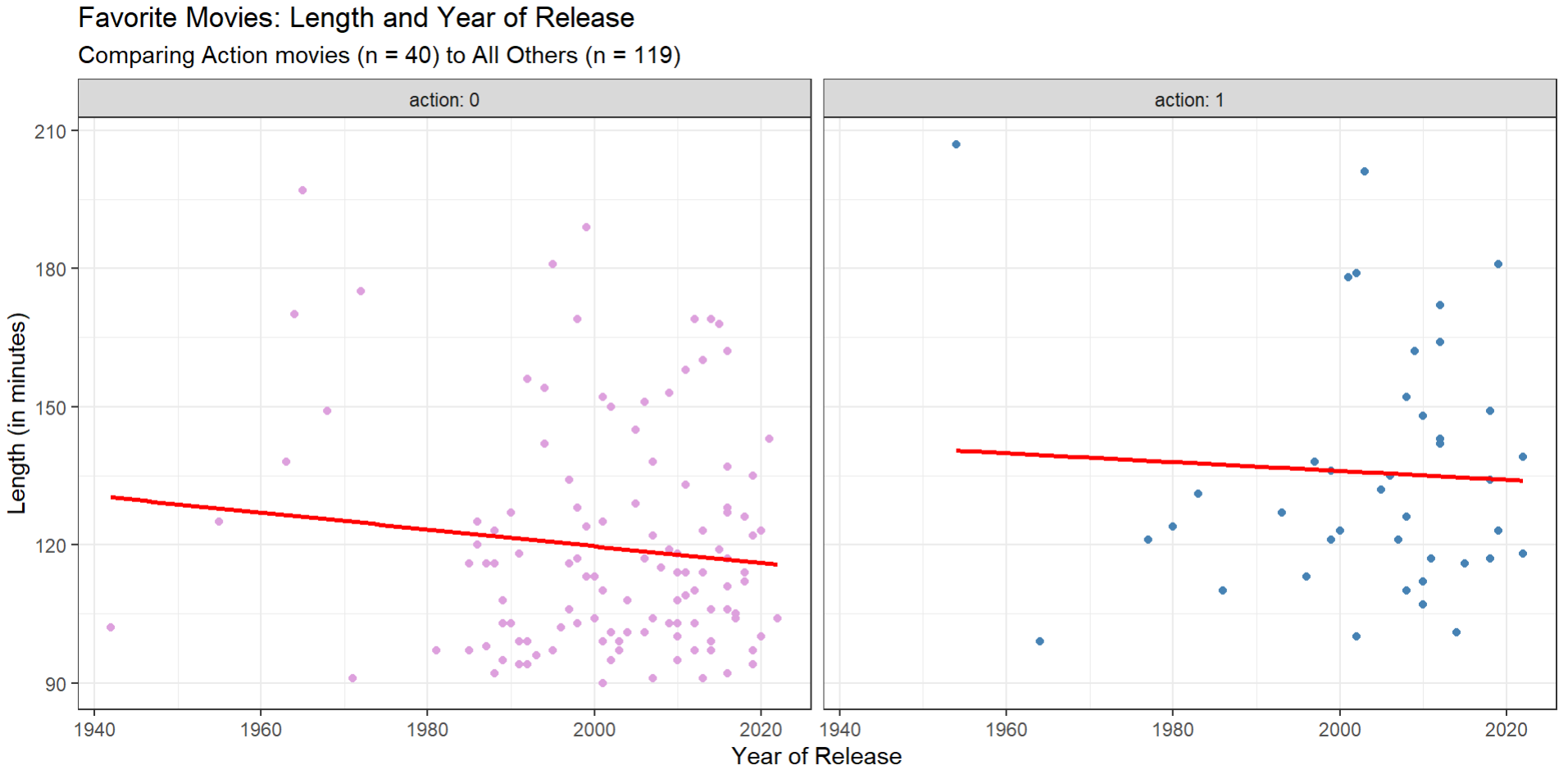
```
glance(m1) |> select(r.squared, sigma, AIC, nobs, df, df.residual)
```

```
# A tibble: 1 × 6
  r.squared sigma   AIC  nobs    df df.residual
      <dbl> <dbl> <dbl> <int> <dbl>       <int>
1   0.00594  25.8 1488.   159     1         157
```

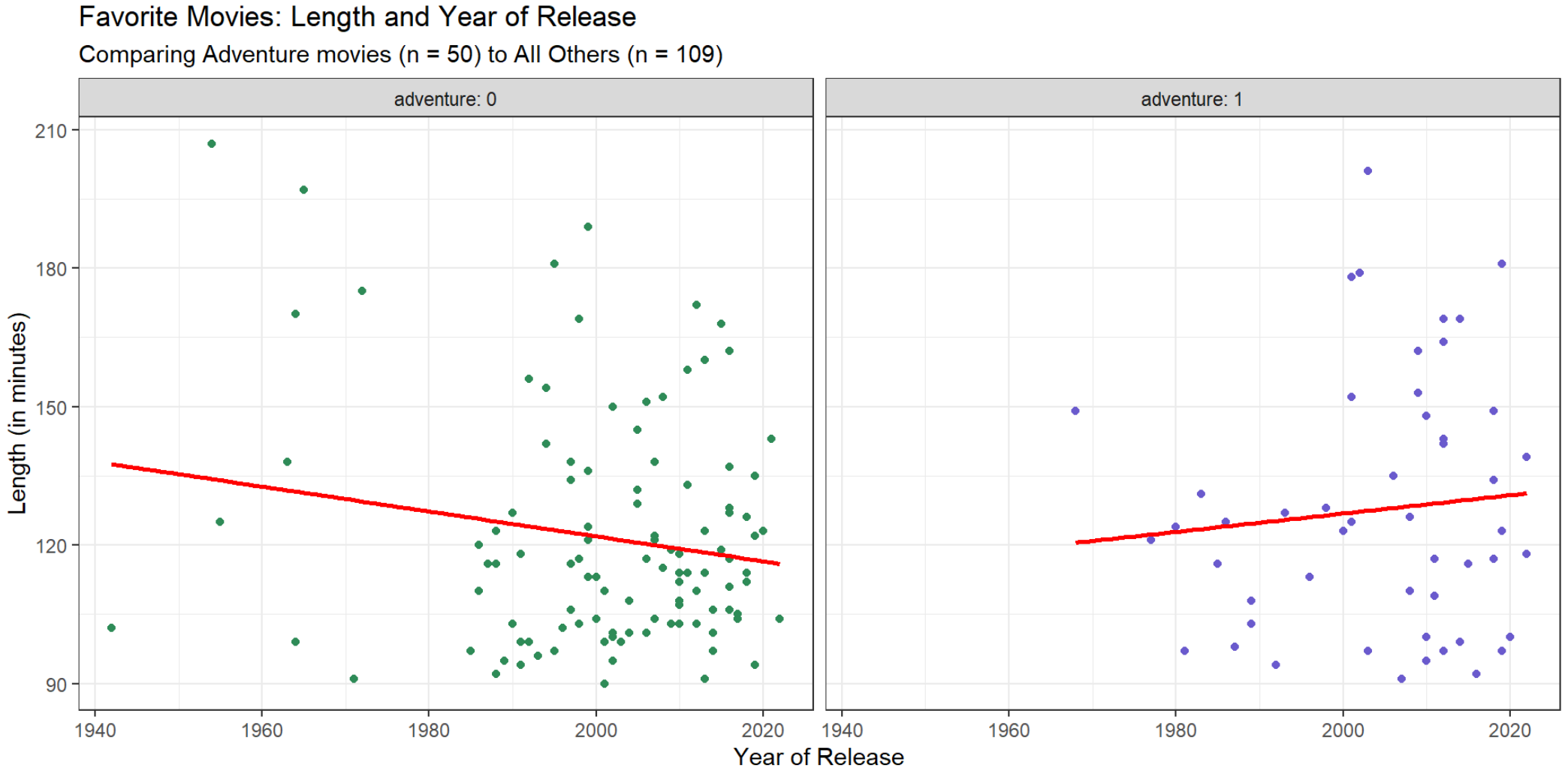
Year and Length for Action/non-Action

```
ggplot(movies22, aes(x = year, y = length, col = factor(action))) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  facet_wrap(~ action, labeller = "label_both") +
  guides(col = "none") +
  scale_color_manual(values = c("plum", "steelblue")) +
  labs(x = "Year of Release", y = "Length (in minutes)",
       title = "Favorite Movies: Length and Year of Release",
       subtitle = glue("Comparing Action movies (n = ",
                       sum(movies22$action), ") to All Others (n = ",
                       nrow(movies22) - sum(movies22$action), ")"))
```

Year and Length for Action/non-Action



Year and Length for Adventure or Not?



Interaction of Centered Year & Adventure

```
movies22 <- movies22 |> mutate(year_c = year - mean(year))

m2 <- lm(length ~ year_c * adventure, data = movies22)
extract_eq(m2, use_coefs = TRUE, wrap = TRUE, operator_location = "start",
           terms_per_line = 1)
```

$$ \begin{aligned} \operatorname{\widehat{length}} &= 121.35\\ &\quad - 0.27(\operatorname{year\_c})\\ &\quad + 5.92(\operatorname{adventure})\\ &\quad + 0.47(\operatorname{year\_c} \times \operatorname{adventure}) \end{aligned} $$

Coefficients and Summaries

```
  1 tidy(m2, conf.int = TRUE, conf.level = 0.90)
```

```
# A tibble: 4 × 7
  term             estimate std.error statistic  p.value conf.low conf.high
  <chr>               <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)       121.        2.45      49.5  7.30e-97 117.     125.
2 year_c             -0.270     0.159     -1.70 9.11e- 2  -0.532   -0.00724
3 adventure           5.92      4.40       1.34 1.81e- 1  -1.37    13.2
4 year_c:adventure    0.468     0.317      1.48 1.42e- 1  -0.0568   0.993
```

```
  1 glance(m2) |> select(r.squared, sigma, AIC, nobs, df, df.residual)
```

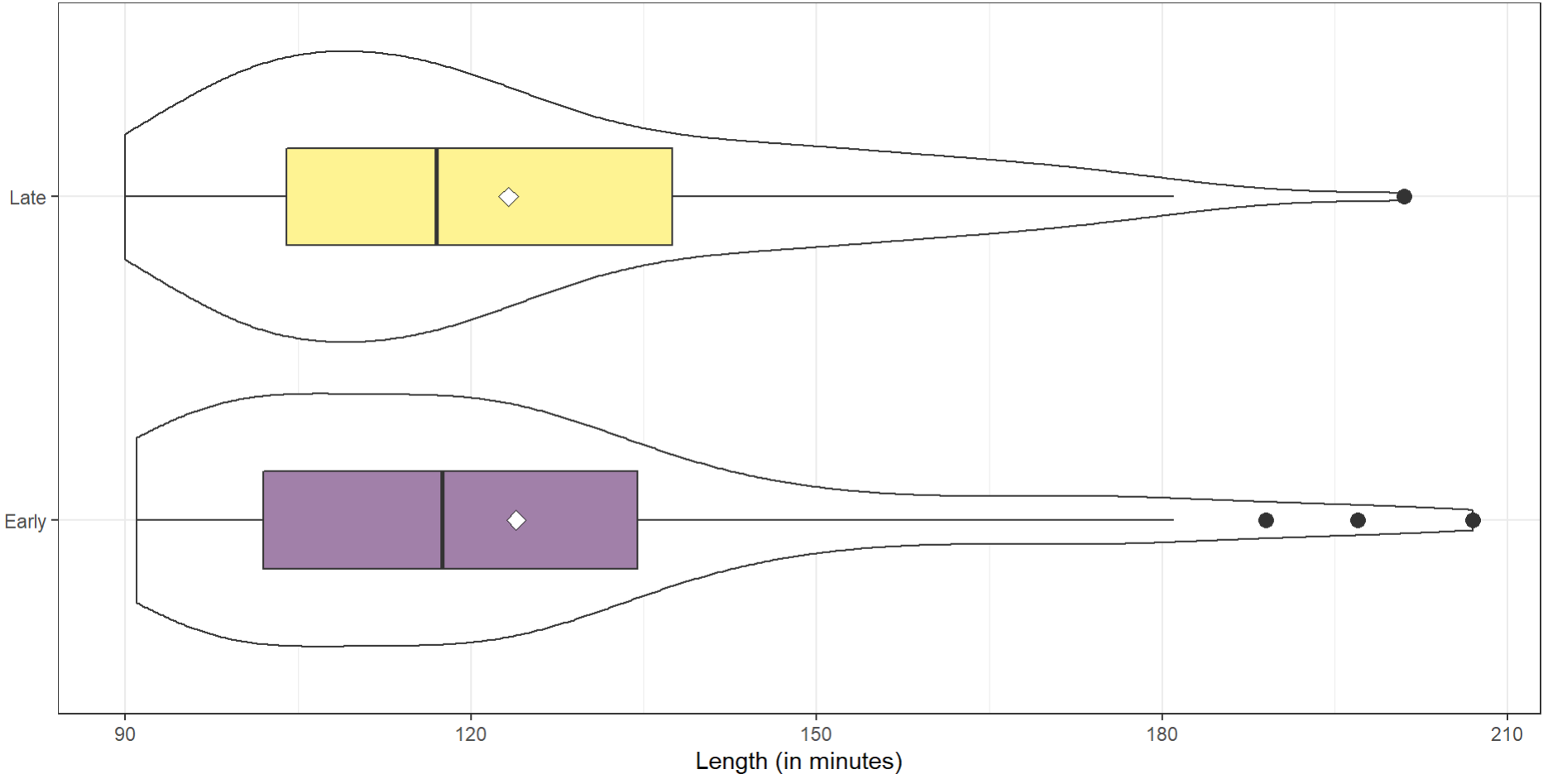
```
# A tibble: 1 × 6
  r.squared sigma   AIC  nobs    df df.residual
      <dbl> <dbl> <dbl> <int> <dbl>       <int>
1    0.0334  25.6 1488.   159     3         155
```

Tweak the Question?

Are movies made prior to 2000 longer or shorter than movies after 2000?

```
movies22 <- movies22 |>
  mutate(before2000 = factor(ifelse(year < 2000, "Early", "Late")))

ggplot(movies22, aes(x = before2000, y = length)) +
  geom_violin() +
  geom_boxplot(aes(fill = before2000), width = 0.3, outlier.size = 3) +
  stat_summary(fun = "mean", geom = "point",
               shape = 23, size = 3, fill = "white") +
  scale_fill_viridis_d(alpha = 0.5) +
  guides(fill = "none") +
  coord_flip() +
  labs(x = "", y = "Length (in minutes)")
```

Tweak the Question?



Meaningful difference in means?

```
favstats(length ~ before2000, data = movies22)
```

```
  before2000 min  Q1 median    Q3 max     mean       sd   n missing
1      Early  91 102  117.5 134.5 207 123.9464 28.28068  56       0
2       Late  90 104  117.0 137.5 201 123.2816 24.39842 103       0
```

```
m3 <- lm(length ~ before2000, data = movies22)
tidy(m3, conf.int = T, conf.level = 0.90)
```

```
# A tibble: 2 × 7
  term           estimate std.error statistic  p.value conf.low conf.high
  <chr>             <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)     124.         3.45    35.9   1.29e-77   118.      130.
2 before2000Late   -0.665      4.29    -0.155 8.77e- 1    -7.76      6.43
```

```
glance(m3) |> select(r.squared, sigma, AIC, nobs, df, df.residual)
```

```
# A tibble: 1 × 6
  r.squared sigma   AIC  nobs    df df.residual
      <dbl> <dbl> <dbl> <int> <dbl>       <int>
1  0.000153  25.8 1489.   159     1         157
```

Do Dramas have higher ratings (# of IMDB Stars) than Comedies?

Do Dramas have higher ratings than Comedies?

```
1 movies22 |> tabyl(comedy, drama) |> adorn_title()
```

```
        drama
comedy      0  1
     0     31 72
     1     33 23
```

- What should we do about this?
- Exclude the Movies that are both, or neither (Approach 1)
- Include all of the Movies, making 4 categories (Approach 2)

Approach 1

Do Dramas have higher ratings (more `imdb_stars`) than Comedies?

- excluding the Movies that are both, or neither…

```
1 mov_dc1 <- movies22 |>
2   filter(comedy + drama == 1)
3
4 mov_dc1 |> tabyl(comedy, drama) |> adorn_title()
```

```
        drama
comedy      0  1
     0      0 72
     1     33  0
```

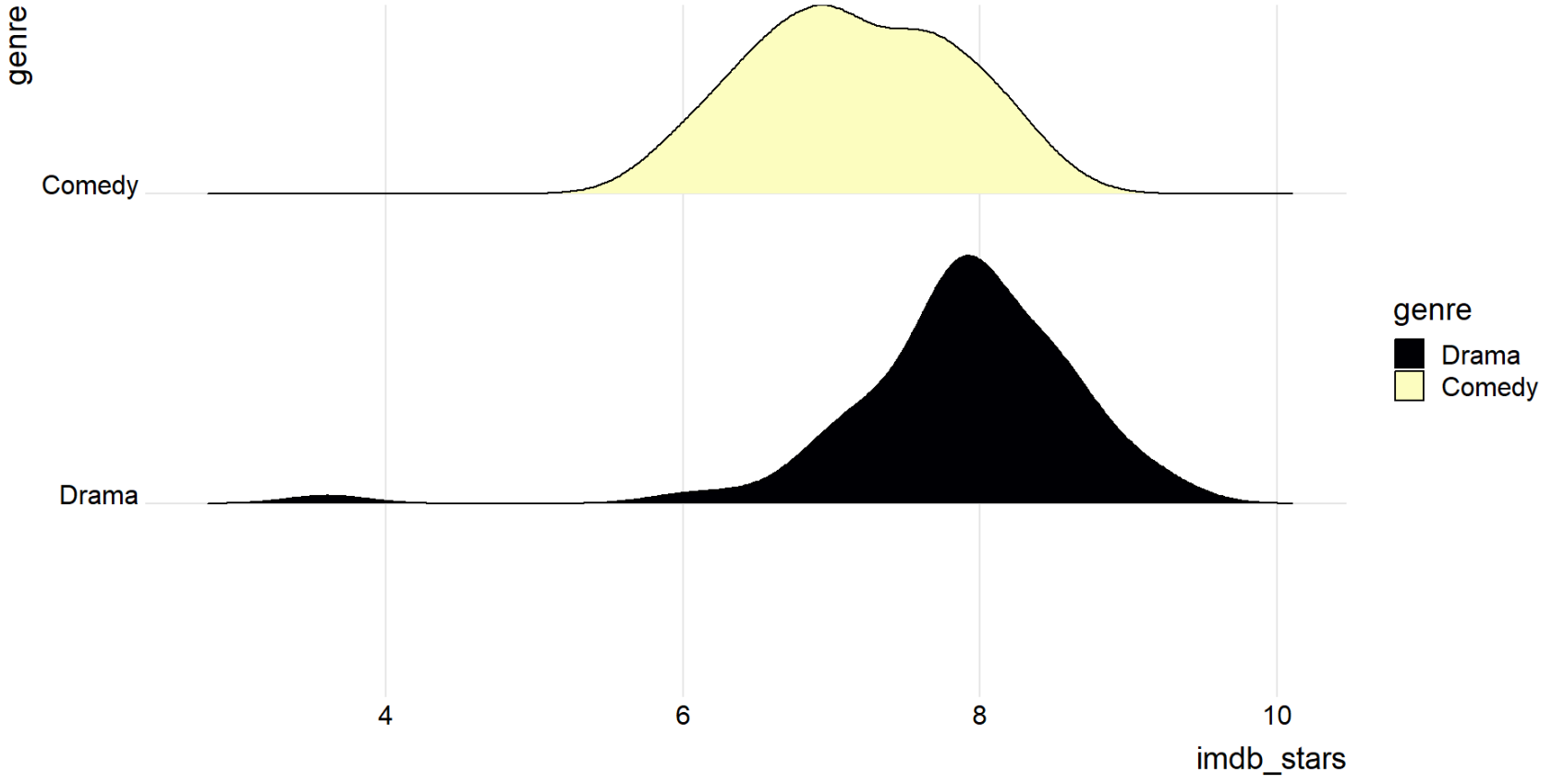
Approach 1 (continued)

```
1 mov_dc1 <- mov_dc1 |>
2   mutate(genre = fct_recode(factor(comedy), "Comedy" = "1", "Drama" = "0"))
3
4 mov_dc1 |> count(genre, comedy, drama)
```

```
# A tibble: 2 × 4
  genre  comedy drama     n
  <fct>   <dbl> <dbl> <int>
1 Drama       0     1    72
2 Comedy      1     0    33
```

Approach 1 (Stars by Genre)

```
ggplot(data = mov_dc1, aes(x = imdb_stars, y = genre,
                           fill = genre, height = ..density..)) +
  geom_density_ridges(scale = 0.8) +
  scale_fill_viridis_d(option = "A") + theme_ridges()
```



Approach 1 (Stars by Genre)

```
favstats(imdb_stars ~ genre, data = mov_dc1)
```

```
   genre min  Q1 median  Q3 max     mean        sd  n missing
1  Drama 3.6 7.6    8.0 8.4 9.3 7.870833 0.8116125 72       0
2 Comedy 5.8 6.6    7.1 7.7 8.5 7.154545 0.6887901 33       0
```

```
m4 <- lm(imdb_stars ~ genre, data = mov_dc1)

tidy(m4, conf.int = T, conf.level = 0.9)
```

```
# A tibble: 2 × 7
  term        estimate std.error statistic  p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)    7.87     0.0914     86.1  8.64e-98    7.72      8.02
2 genreComedy   -0.716    0.163      -4.39 2.71e- 5   -0.987    -0.446
```

Approach 2

Do Dramas have higher ratings (more `imdb_stars`) than Comedies?

- including all of the Movies, creating four categories

```
mov_dc2 <- movies22 |>
  mutate(genre4 = fct_recode(factor(10*comedy + drama),
                             "Comedy only" = "10",
                             "Drama only" = "1",
                             "Both" = "11",
                             "Neither" = "0"))
```

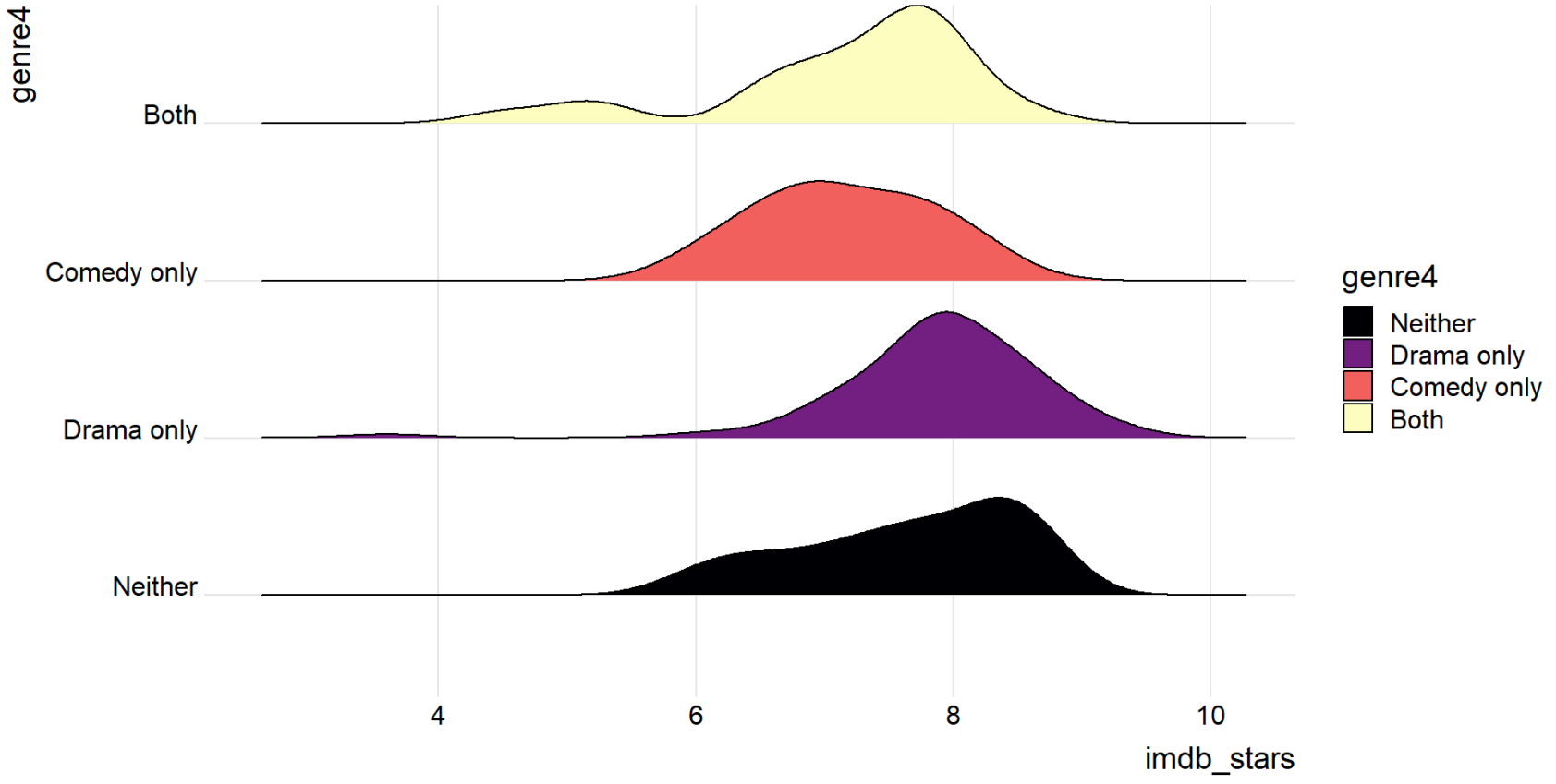
Check that We Recoded Correctly

```
1 mov_dc2 |> count(comedy, drama, genre4)
```

```
# A tibble: 4 × 4
  comedy drama genre4           n
   <dbl> <dbl> <fct>        <int>
1      0     0 Neither         31
2      0     1 Drama only      72
3      1     0 Comedy only     33
4      1     1 Both            23
```

Approach 2 (Stars by Genre)

```
ggplot(data = mov_dc2, aes(x = imdb_stars, y = genre4,
                           fill = genre4, height = ..density..)) +
  geom_density_ridges(scale = 0.8) +
  scale_fill_viridis_d(option = "A") + theme_ridges()
```



Approach 2 (Stars by Genre)

```
favstats(imdb_stars ~ genre4, data = mov_dc2)
```

```
       genre4 min   Q1 median  Q3 max     mean         sd  n missing
1     Neither 5.9 7.10    7.8 8.4 8.8 7.648387 0.8590192 31       0
2  Drama only 3.6 7.60    8.0 8.4 9.3 7.870833 0.8116125 72       0
3 Comedy only 5.8 6.60    7.1 7.7 8.5 7.154545 0.6887901 33       0
4        Both 4.5 6.75    7.6 7.8 8.6 7.160870 1.0232476 23       0
```

```
m5 <- lm(imdb_stars ~ genre4, data = mov_dc2)
tidy(m5, conf.int = T, conf.level = 0.9)
```

```
# A tibble: 4 × 7
  term              estimate std.error statistic  p.value conf.low conf.high
  <chr>                <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)          7.65      0.149     51.2  4.66e-99   7.40       7.90
2 genre4Drama only     0.222     0.179      1.25 2.15e- 1  -0.0731     0.518
3 genre4Comedy only   -0.494     0.208     -2.37 1.88e- 2  -0.838     -0.150
4 genre4Both          -0.488     0.229     -2.13 3.47e- 2  -0.866     -0.109
```

A Few More Scatterplots

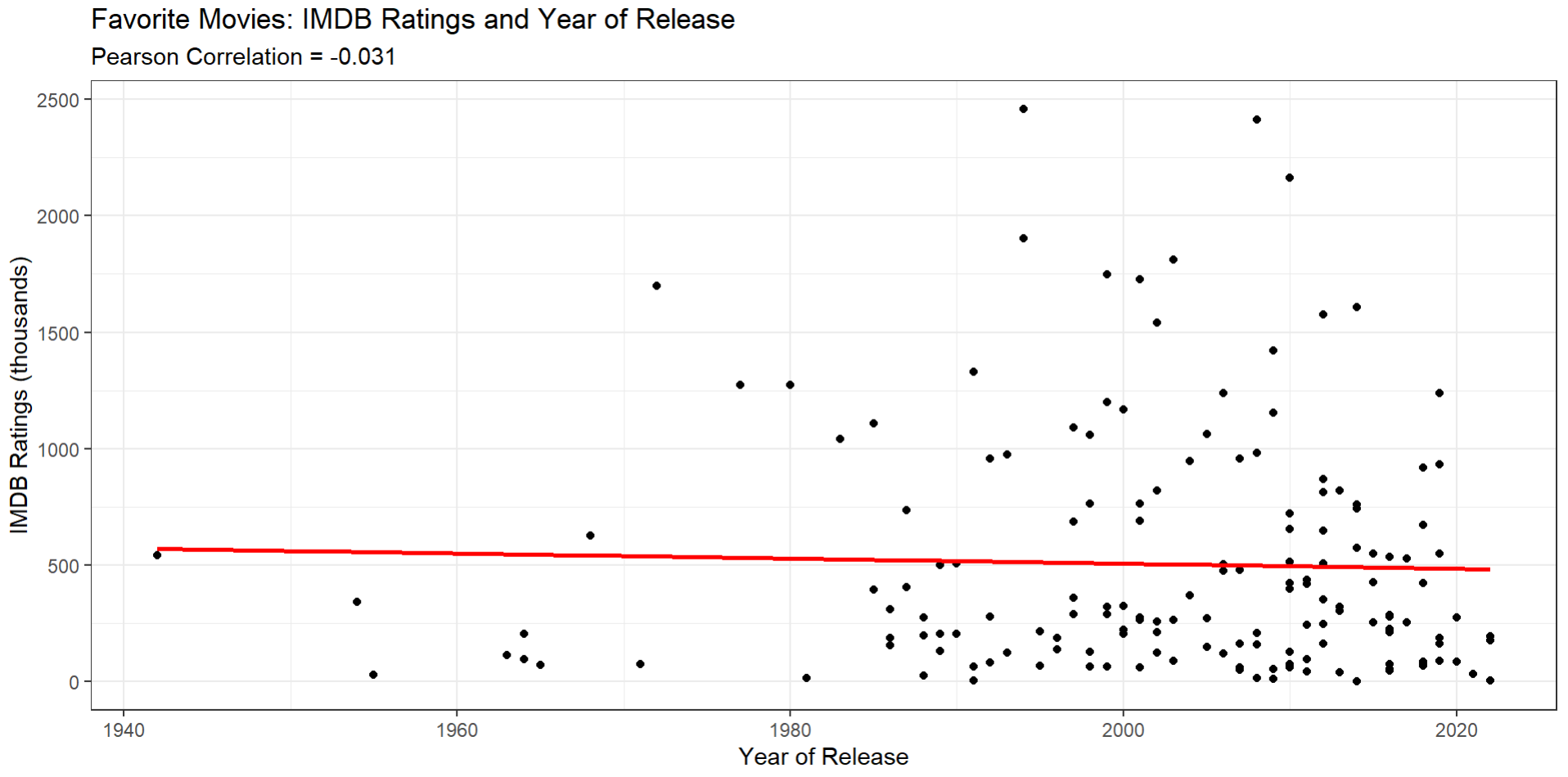
Some of Your Other Exploratory Questions

- What is the relationship between the year a movie was released and the number of star ratings at IMDB?
- How does IMDB rating (`imdb_stars`) differ between older and newer movies?
- Are the average IMDB ratings associated with the number of IMDB star ratings?
- Is there a relationship between movie length and number of star ratings?

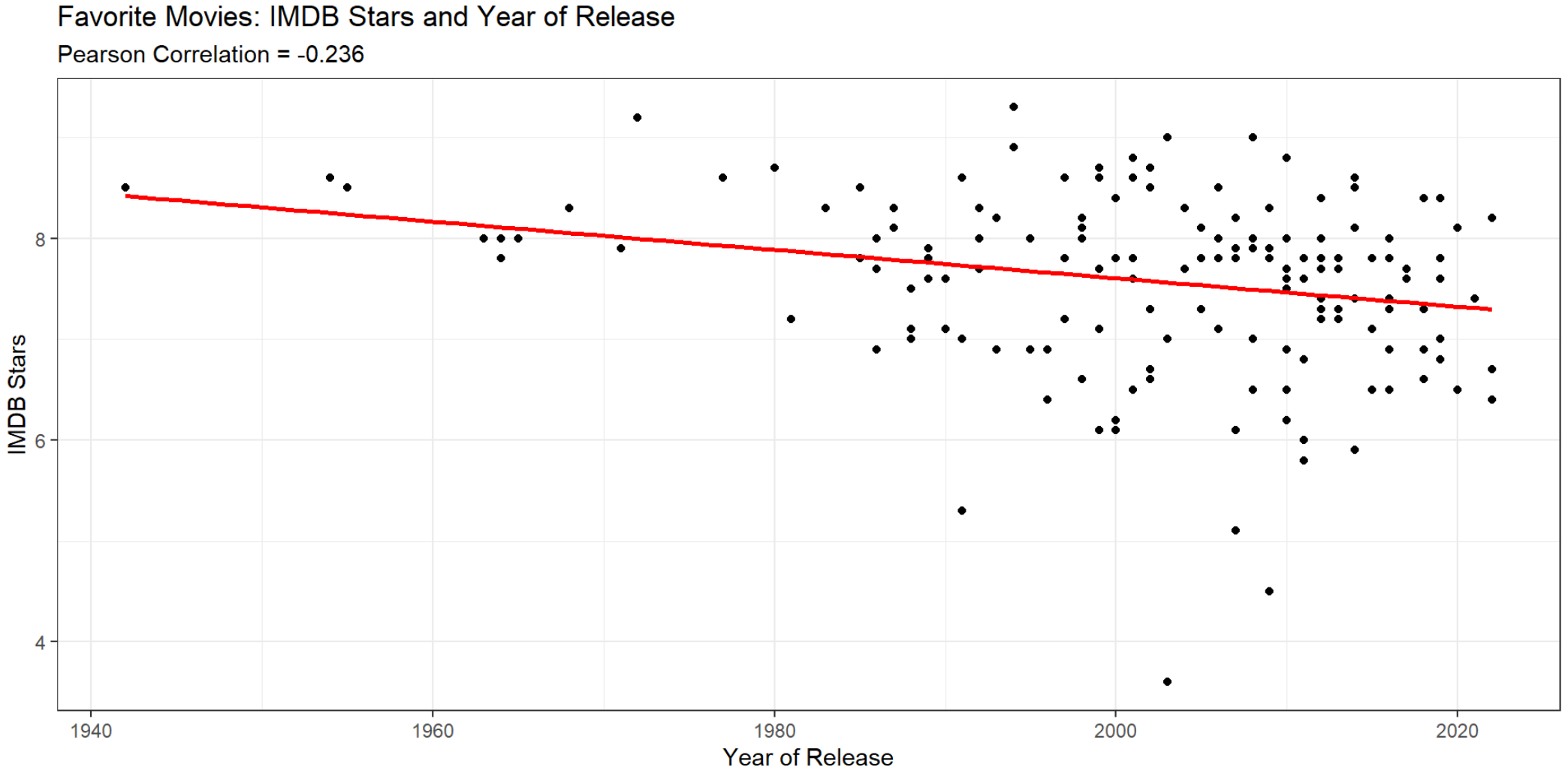
Year vs. # of Star Ratings?

```
ggplot(movies22, aes(x = year, y = imdb_ratings/1000)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, col = "red") +
  labs(x = "Year of Release", y = "IMDB Ratings (thousands)",
       title = "Favorite Movies: IMDB Ratings and Year of Release",
       subtitle = glue("Pearson Correlation = ", round_half_up(
         cor(movies22$year, movies22$imdb_ratings), 3)))
```

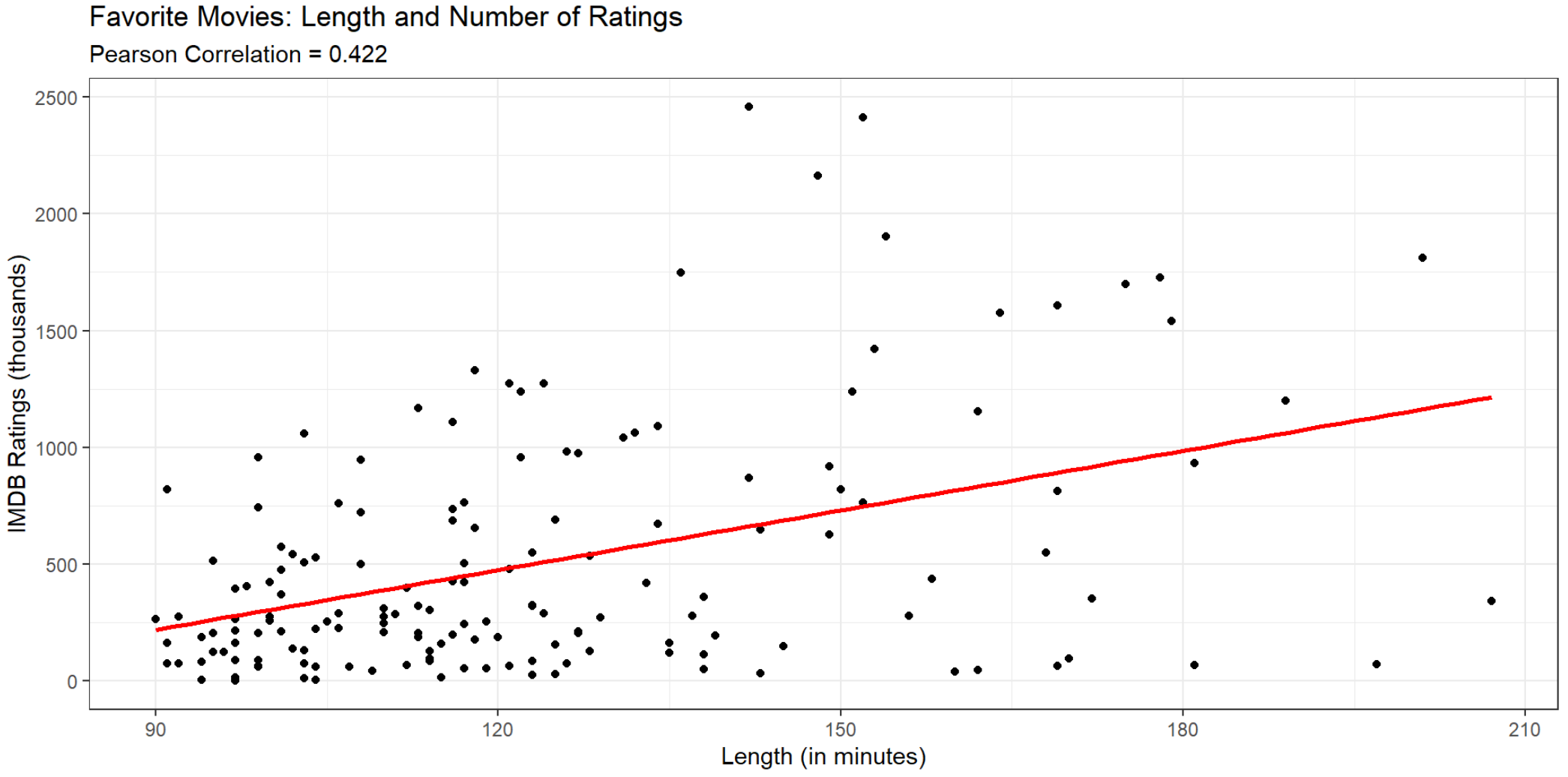
Year vs. # of Star Ratings?



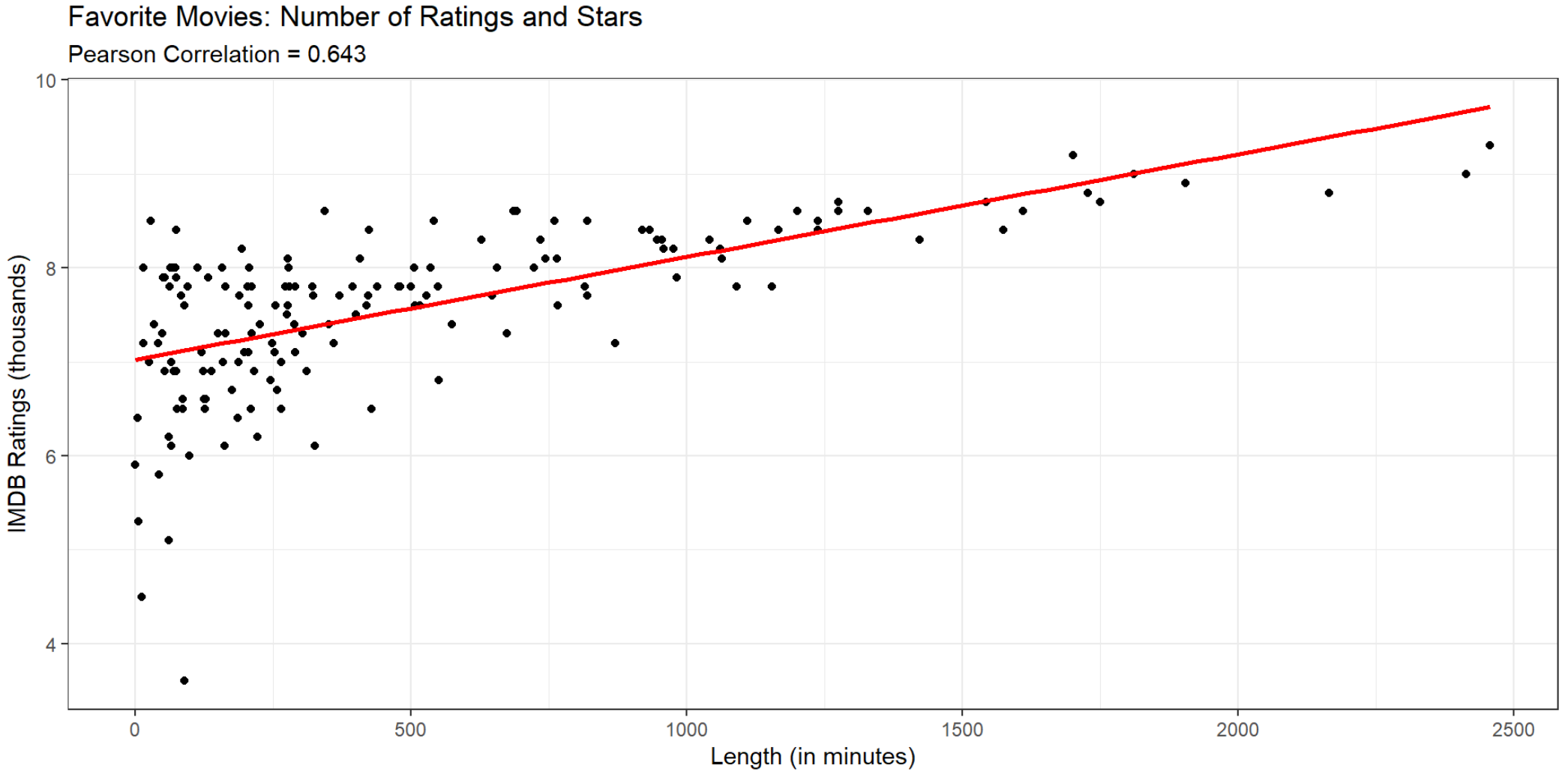
Year vs. Number of Stars?



Length vs. # of Star Ratings?



Number of Ratings vs. Number of Stars?



Session Information

```
1 sessionInfo()
```

```
R version 4.2.1 (2022-06-23 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 22000)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```