Data Science

Exploratory Data Analysis and Data Transformation

Carley Dziewicki

February 4, 2019

Movie Data Analysis

In this analysis, you will investigate IMDB and Rotten Tomatoes ratings for a random sample of 651 movies. The following command loads the data into a dataframe named `movies`..[1]

```
# load the data from an R binary data file into the data frame movies
load(file = "data/movies.Rdata")
#data.frame(movies)
```

a. Use the R command that will give an overview of the `movies` data frame by showing all the variable names and the first few observations from each.

```
glimpse(movies)
```

```
## Observations: 651
## Variables: 32
## $ title            <chr> "Filly Brown", "The Dish", "Waiting for Guffman", ...
## $ title_type       <fct> Feature Film, Feature Film, Feature Film, Feature ...
## $ genre            <fct> Drama, Drama, Comedy, Drama, Horror, Documentary, ...
## $ runtime          <dbl> 80, 101, 84, 139, 90, 78, 142, 93, 88, 119, 127, 1...
## $ mpaa_rating      <fct> R, PG-13, R, PG, R, Unrated, PG-13, R, Unrated, Un...
## $ studio           <fct> Indomina Media Inc., Warner Bros. Pictures, Sony P...
## $ thtr_rel_year    <dbl> 2013, 2001, 1996, 1993, 2004, 2009, 1986, 1996, 20...
## $ thtr_rel_month   <dbl> 4, 3, 8, 10, 9, 1, 1, 11, 9, 3, 6, 12, 1, 9, 6, 8,...
## $ thtr_rel_day     <dbl> 19, 14, 21, 1, 10, 15, 1, 8, 7, 2, 19, 18, 4, 23, ...
## $ dvd_rel_year     <dbl> 2013, 2001, 2001, 2001, 2005, 2010, 2003, 2004, 20...
## $ dvd_rel_month    <dbl> 7, 8, 8, 11, 4, 4, 2, 3, 1, 8, 5, 9, 7, 2, 3, 12, ...
## $ dvd_rel_day      <dbl> 30, 28, 21, 6, 19, 20, 18, 2, 21, 14, 1, 23, 9, 13...
## $ imdb_rating      <dbl> 5.5, 7.3, 7.6, 7.2, 5.1, 7.8, 7.2, 5.5, 7.5, 6.6, ...
## $ imdb_num_votes   <int> 899, 12285, 22381, 35096, 2386, 333, 5016, 2272, 8...
## $ critics_rating   <fct> Rotten, Certified Fresh, Certified Fresh, Certifie...
## $ critics_score    <dbl> 45, 96, 91, 80, 33, 91, 57, 17, 90, 83, 89, 67, 80...
## $ audience_rating  <fct> Upright, Upright, Upright, Upright, Spilled, Uprig...
## $ audience_score   <dbl> 73, 81, 91, 76, 27, 86, 76, 47, 89, 66, 75, 46, 89...
## $ best_pic_nom     <fct> no, no, no, no, no, no, no, no, no, no, no, no, no...
## $ best_pic_win     <fct> no, no, no, no, no, no, no, no, no, no, no, no, no...
## $ best_actor_win   <fct> no, no, no, yes, no, no, no, yes, no, no, yes, no,...
## $ best_actress_win <fct> no, no, no, no, no, no, no, no, no, no, no, no, ye...
## $ best_dir_win     <fct> no, no, no, yes, no, no, no, no, no, no, no, no, n...
## $ top200_box       <fct> no, no, no, no, no, no, no, no, no, no, yes, no, n...
## $ director         <chr> "Michael D. Olmos", "Rob Sitch", "Christopher Gues...
```

[1]This problem adapts a tutorial prepared by Iain Carmichael and makes use of the movies data generously provided by Mine Cetinkaya-Rundel. The original data set is available at her website.

```
## $ actor1           <chr> "Gina Rodriguez", "Sam Neill", "Christopher Guest"...
## $ actor2           <chr> "Jenni Rivera", "Kevin Harrington", "Catherine O'H...
## $ actor3           <chr> "Lou Diamond Phillips", "Patrick Warburton", "Park...
## $ actor4           <chr> "Emilio Rivera", "Tom Long", "Eugene Levy", "Richa...
## $ actor5           <chr> "Joseph Julian Soria", "Genevieve Mooy", "Bob Bala...
## $ imdb_url         <chr> "http://www.imdb.com/title/tt1869425/", "http://ww...
## $ rt_url           <chr> "//www.rottentomatoes.com/m/filly_brown_2012/", "/...
```

b. Use `count()` to obtain frequency tables for four factor variables: `title_type`, `genre` and `critics_rating` and `audience_rating`.

```
movies %>%
count(title_type, sort=TRUE)
```

```
## # A tibble: 3 x 2
##   title_type       n
##   <fct>        <int>
## 1 Feature Film   591
## 2 Documentary     55
## 3 TV Movie         5
```

```
movies %>%
count(genre, sort=TRUE)
```

```
## # A tibble: 11 x 2
##    genre                        n
##    <fct>                    <int>
##  1 Drama                      305
##  2 Comedy                      87
##  3 Action & Adventure          65
##  4 Mystery & Suspense          59
##  5 Documentary                 52
##  6 Horror                      23
##  7 Other                       16
##  8 Art House & International   14
##  9 Musical & Performing Arts   12
## 10 Animation                    9
## 11 Science Fiction & Fantasy    9
```

```
movies %>%
count(critics_rating, sort=TRUE)
```

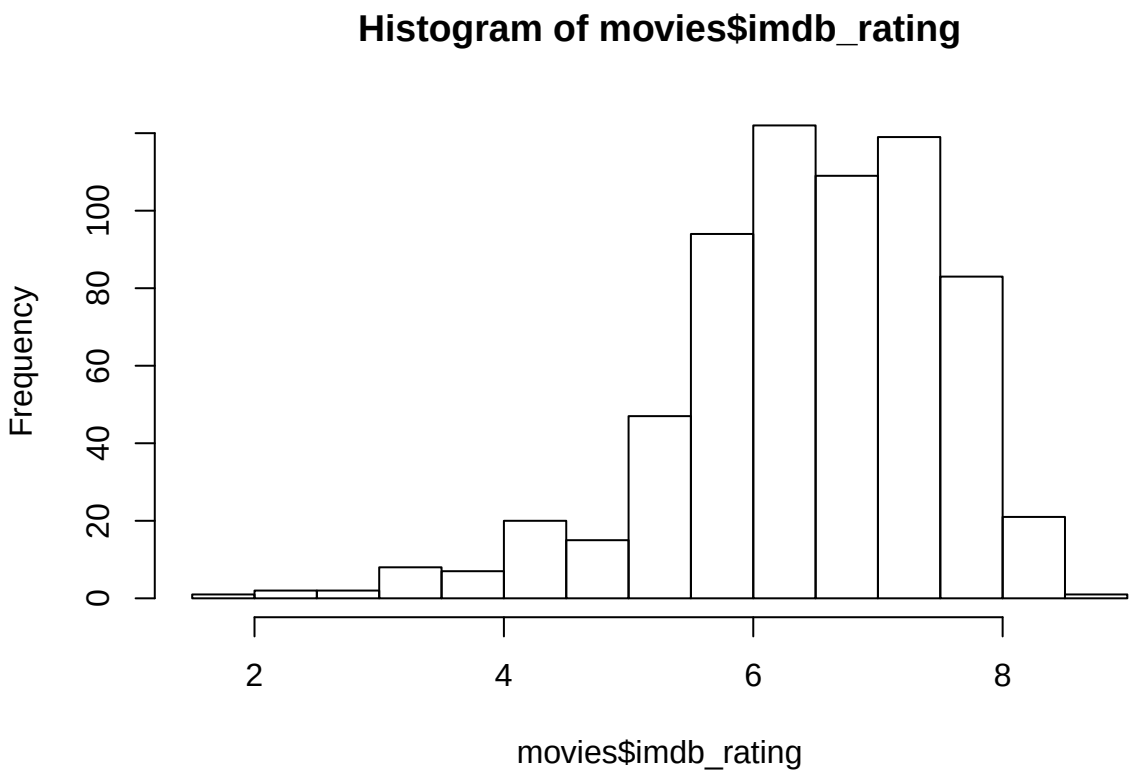
```
## # A tibble: 3 x 2
##   critics_rating      n
##   <fct>           <int>
## 1 Rotten            307
## 2 Fresh             209
## 3 Certified Fresh   135
```

```
movies %>%
count(audience_rating, sort=TRUE)
```

```
## # A tibble: 2 x 2
##   audience_rating     n
##   <fct>           <int>
## 1 Upright           376
## 2 Spilled           275
```

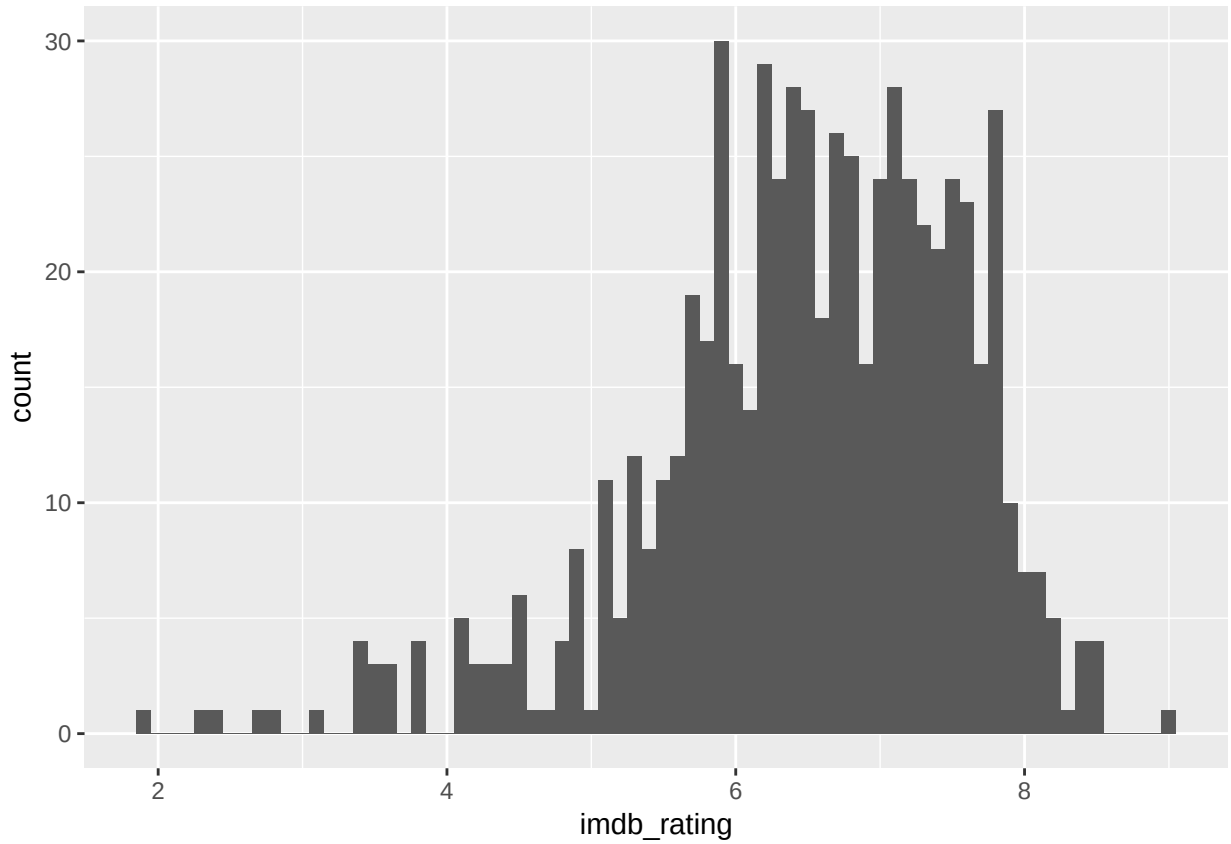
c. Make a histogram of **imdb_rating**.

```
hist(movies$imdb_rating)
```



d. Repeat the previous histogram but this time pick a binwidth such that each bar will correspond to just one distinct value of the `imdb_rating` variable.

```
ggplot(movies, mapping=aes(x=imdb_rating)) + geom_histogram(binwidth = .1)
```




```
movies %>% count(imdb_rating)
```

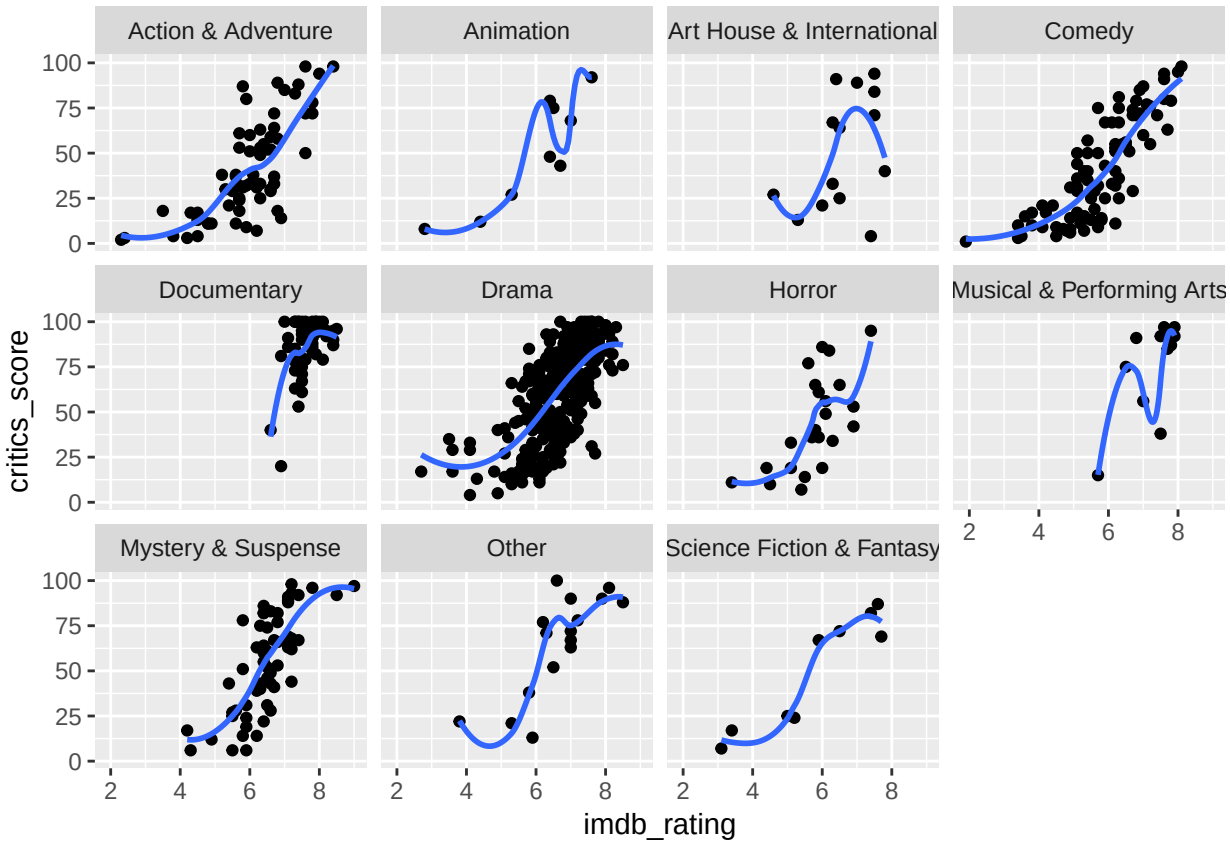
```
## # A tibble: 56 x 2
##    imdb_rating     n
##          <dbl> <int>
##  1         1.9     1
##  2         2.3     1
##  3         2.4     1
##  4         2.7     1
##  5         2.8     1
##  6         3.1     1
##  7         3.4     4
##  8         3.5     3
##  9         3.6     3
## 10         3.8     4
## # ... with 46 more rows
```

e. Write two or three sentences summarizing your observations about the `imdb_rating` scores.
Answer: From the histograms seen above the 'imdb_rating' scores are centered around 7, with the highest conentration from 6-8. There is a left skew for the shape and there don't seem to be any outliers and few observations below the rating of 5. However there are not many observations above 8 either, meaning not many movies get rated poorly and not many get rated extremely well.

f. Create a scatter plot comparing the Rotten Tomatoes `critics_score` (y) to imdb ratings (x).Add a smoothing line (no standard error band). If necessary, add jittering to avoid overplotting.

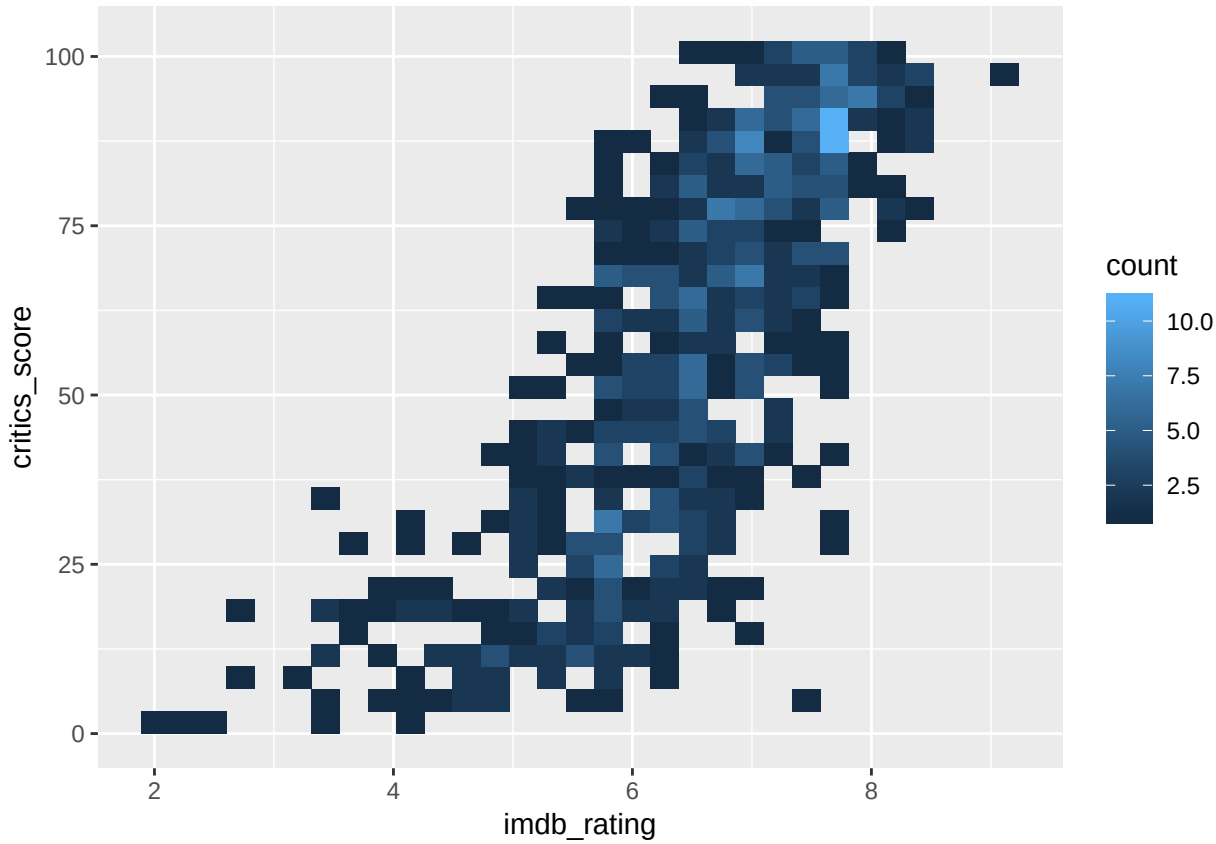
```
ggplot(movies, mapping= aes(y=critics_score, x= imdb_rating))+ geom_point() + facet_wrap(~genre) + geom
```

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

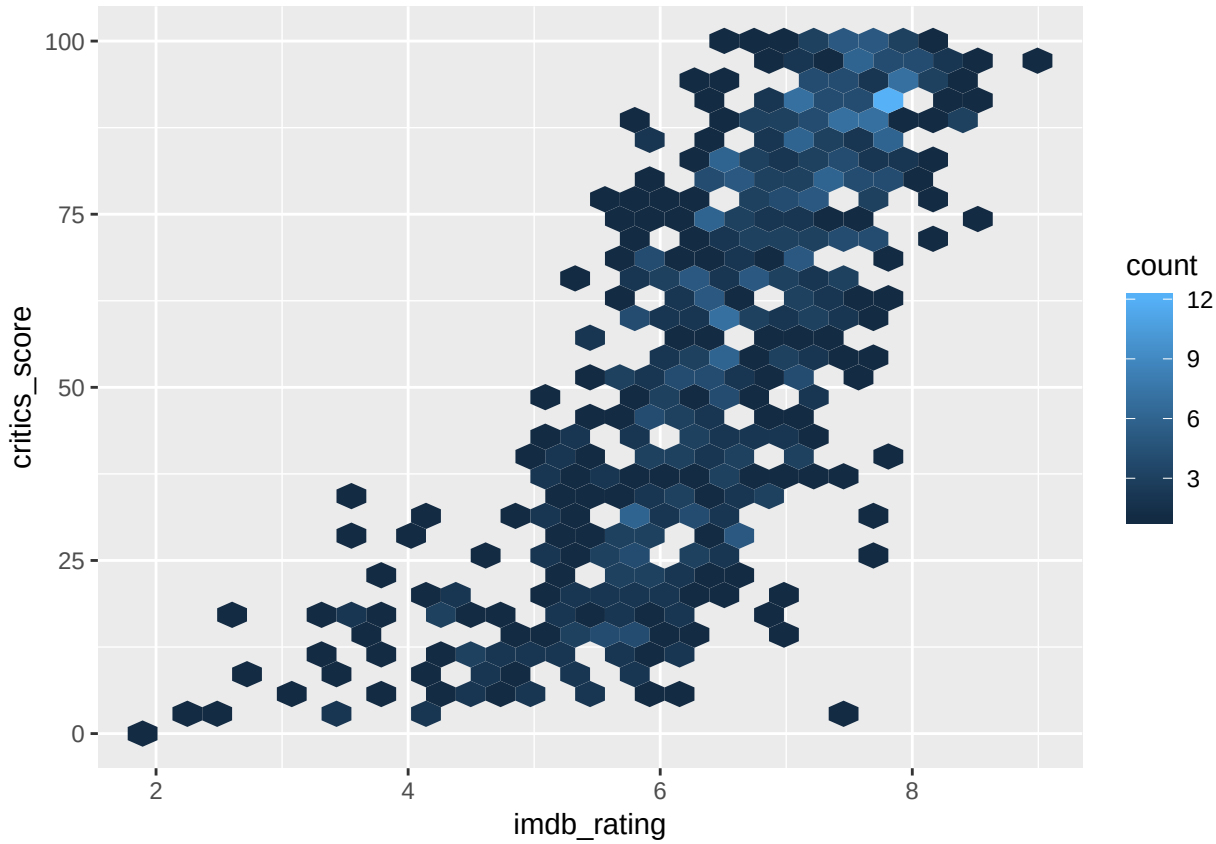


g. Now try some alternatives to traditional scatterplots for exploring the relationship between these two variables. In practice, some of these would be more appropriate for larger data sets but show that you can construct them for this pair of variables. Create each of the following plots described in the text of this section: (i) `geom_bin2d()`; (ii) `geom_hex()`; and (iii) a boxplot using either `cut_width()` or `cut_number()` that has ten boxes and whose width is proportional to the number of movies represented.

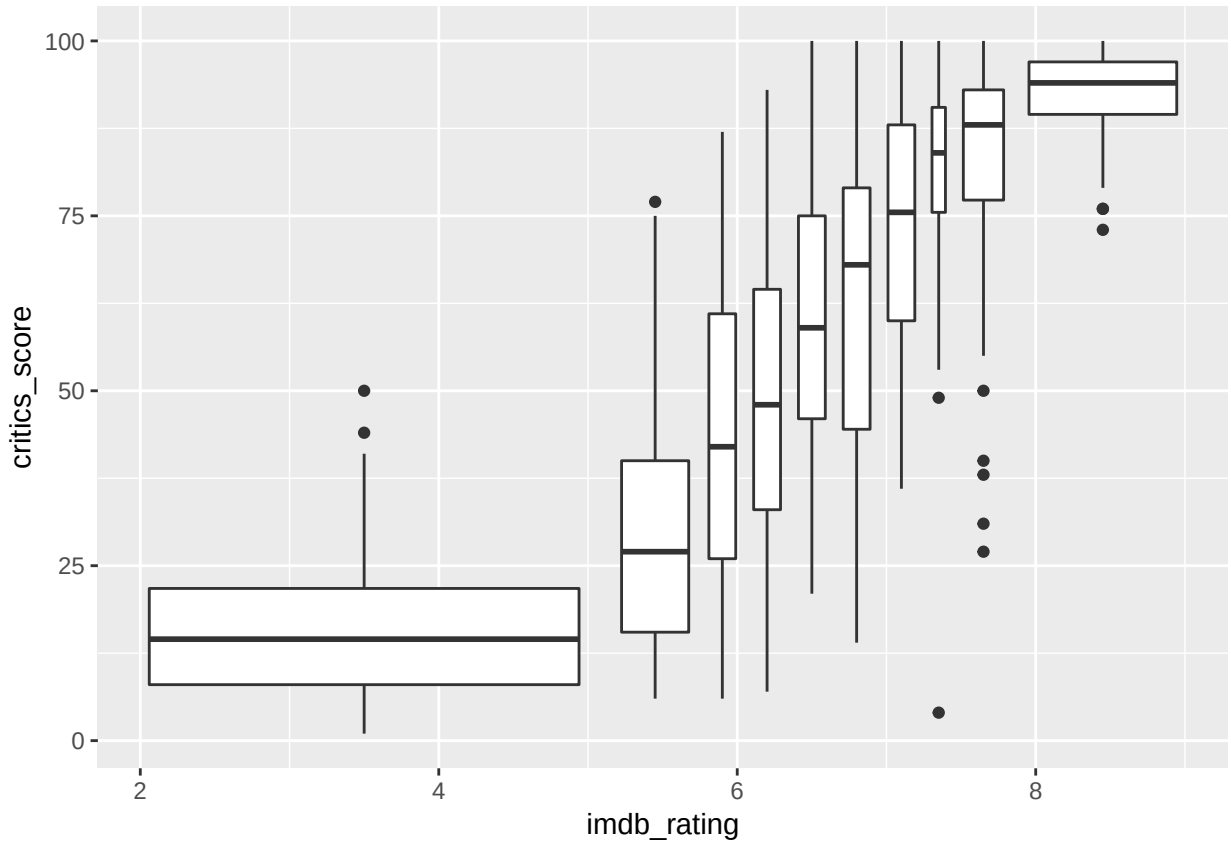
```
ggplot(movies, mapping=aes(y=critics_score, x= imdb_rating)) + geom_bin2d()
```




```
ggplot(movies, mapping=aes(y=critics_score, x= imdb_rating)) + geom_hex()
```




```
ggplot(movies, mapping=aes(y=critics_score, x= imdb_rating)) + geom_boxplot(mapping= aes(group=cut_numbe
```

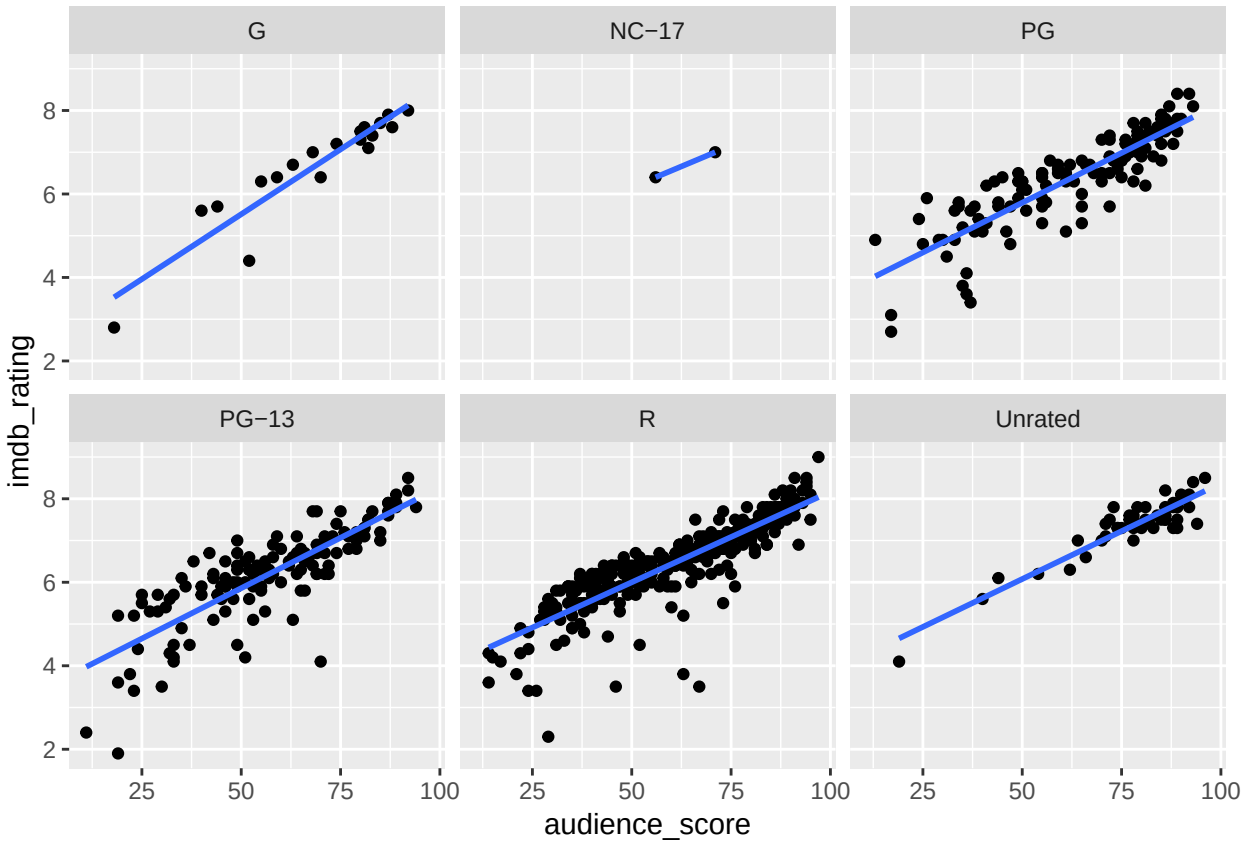


g cont. Inspect your scatterplots from part (f) and (g) and write a couple sentences observing the nature, strength and direction of the covariation of these two variables.

Answer: The scatter plots from (f) and (g) both reveal that the relationship between critics_score and imdb_rating have a strong positive realtionship. There do not seem to be too many outliers, which can be seen clearly in the box plots from (g) that most of the 651 movies fall within the trend of a positive linear relationship.

h. Use faceting to consider whether the relationship between Rotten Tomatoes rating and IMDB rating depends on the MPAA ratings. For each facet, add a least squares regression line without a standard error bar and write a couple sentences with your observations.

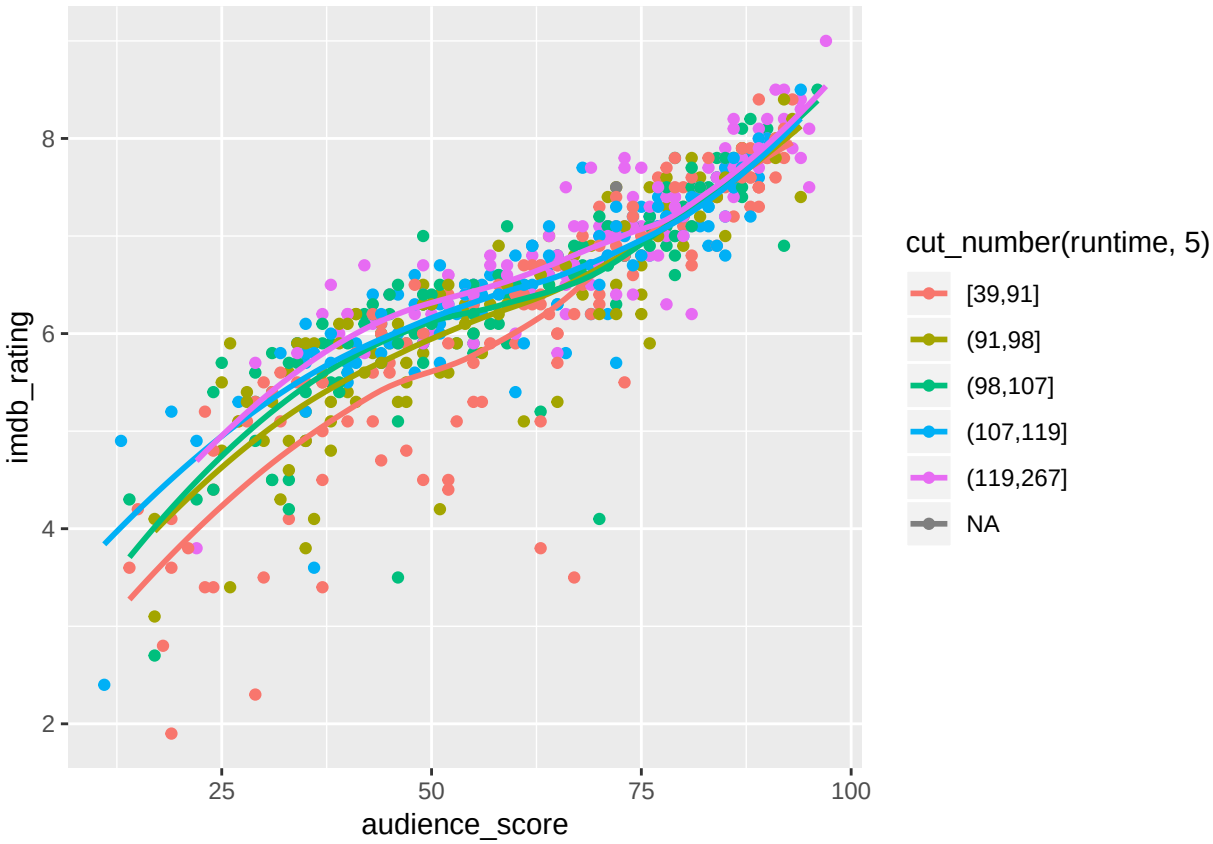
```
ggplot(movies, mapping=aes(y=imdb_rating, x=audience_score)) +geom_point() +facet_wrap(~mpaa_rating) + g
```



i. Instead of faceting, create a graph that shows how the relationship between IMDB rating and Rotten Tomatoes rating is related to `runtime`. That is, add `runtime` as a third variable to your scatterplot from (f) and map it to an aesthetic. You may find it useful to make use of one of the `cut_` functions for incorporating `runtime`. Comment on what you see.

```
ggplot(movies, mapping=aes(y=imdb_rating, x=audience_score, color=cut_number(runtime,5))) +geom_point()
```

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



Data Wrangling below

(j) Use `select()` to create a new data set **movies2** containing just the following variables: `title`, `runtime`, `genre`, `mpaa_rating`, `thtr_rel_year`, `imdb_rating`, `imdb_num_votes`, `critics_score`, `audience_score`, and `best_pic_win`.

```
movies2<- select(movies, title, genre, mpaa_rating, thtr_rel_year, runtime, imdb_rating, imdb_num_votes
```

(k) Compute the means of all the numeric variables. Note which variables have NA values for their means.

```
movies2 %>%
  select_if(is.numeric) %>%
  summarise_all(mean)
```

```
## # A tibble: 1 x 6
##   thtr_rel_year runtime imdb_rating imdb_num_votes critics_score audience_score
##           <dbl>   <dbl>       <dbl>          <dbl>         <dbl>          <dbl>
## 1         1998.      NA        6.49         57533.          57.7           62.4
```

```
mean(movies2$runtime)
```

```
## [1] NA
```

(l) You should have found that the `runtime` variable is missing for at least one movie. Identify the movie(s) missing `runtime` data and track down (Google) the missing data. Replace the NA value(s) with their correct value(s) and re-compute the mean for `runtime`.

```
filter(movies2, is.na(runtime))
```

```
## # A tibble: 1 x 10
```

```
##   title genre mpaa_rating thtr_rel_year runtime imdb_rating imdb_num_votes
##   <chr> <fct> <fct>               <dbl>   <dbl>       <dbl>          <int>
## 1 The ~ Docu~ Unrated              2008      NA         7.5            739
## # ... with 3 more variables: critics_score <dbl>, audience_score <dbl>,
## #   best_pic_win <fct>
```

```
moviesna<-movies2 %>% mutate(runtime= replace_na(runtime,74))
moviesna %>%
  select_if(is.numeric) %>%
  summarise_all(mean)
```

```
## # A tibble: 1 x 6
##   thtr_rel_year runtime imdb_rating imdb_num_votes critics_score audience_score
##           <dbl>   <dbl>       <dbl>          <dbl>         <dbl>          <dbl>
## 1         1998.    106.        6.49         57533.          57.7           62.4
```

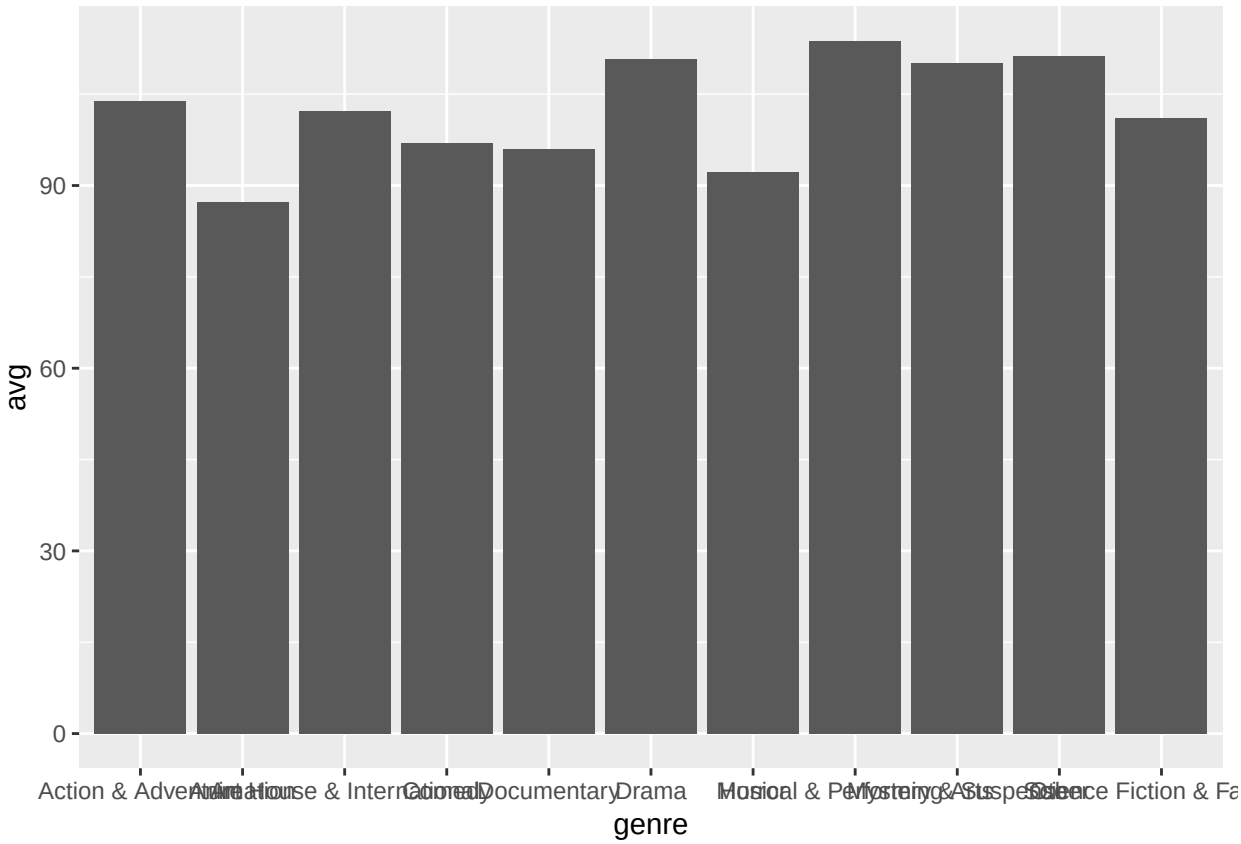
runtime= 74 minutes

(m) Compute the mean `runtime` for each `genre` and then create a bar chart of the mean `runtime` by `genre`. Which genres have the lowest and highest average `runtime`?

```
moviesavg<- moviesna %>% group_by(genre) %>% summarise(avg=mean(runtime))
moviesavg
```

```
## # A tibble: 11 x 2
##    genre                       avg
##    <fct>                     <dbl>
##  1 Action & Adventure        104.
##  2 Animation                  87.2
##  3 Art House & International 102.
##  4 Comedy                     96.9
##  5 Documentary                96.1
##  6 Drama                     111.
##  7 Horror                     92.1
##  8 Musical & Performing Arts 114.
##  9 Mystery & Suspense        110.
## 10 Other                     111.
## 11 Science Fiction & Fantasy 101
```

```
ggplot(moviesavg, mapping=aes(x=genre, y=avg)) + geom_bar(stat="identity")
```


avg
90
60
30
0
Action & Adventure
Animation
Art House & International
Comedy
Documentary
Drama
Musical & Performing Arts
Mystery & Suspense
Science Fiction & Fa
genre