User intentions and behavior exploration when rating in Twitter

A Comparison of movie ratings in MovieTwittering dataset and IMDB dataset using Spark

Motivation

The goal of this project is to analyze how twitters ratings different from IMDB's ratings among different scales, and most importantly, trying to understand users intentions behind MovieTweeting behaviors. By doing that, we can give film companies intuitions of what type of genres are more likely to become popular in social media, why and when is the best time to do movie propagation, and etc.

There are 3 major questions that will be answered in this data report.

1) What are the differences between the most popular and loved genres in IMDB and twitter platforms? Do most voted movies usually get higher ratings?

2) Would twitter users more likely to share their reviews(ratings) for old movies or for most recent movies?

3) Do movie exposure to Twitter boost their ticket sales? In other words, analyze the relationship between BoxOffice with total votes and rating scores.

Data Sources

I. Datasets Introduction

1) IMDB datasets

The IMDB dataset will be scraped from OMDB API (http://www.omdbapi.com/). The crawled movies will be based the IMDB_ID from online Github Repository https://github.com/sidooms/MovieTweetings. The format of the return values from OMDB API will be in json format.

- Variables Introduction

Variables	imdb_ID(key)	Title	Genre	Released_time	imdb_rating	imdb_Votes	Courtry	BoxOffice
Attribute	string	string	string	String(with time like format)	float	int	string	int

Table1 Variables in IMDB dataset

- Records Explanation

I got total 26036 movies and total of 33,64566 votes. The time period for all movies is 1888-2017. The overall mean is relatively low, but we have a mild rating standard deviation.

Dataset	Total Movies Count	Total Votes Count	Movie Time Range	Rating Mean	Rating Std
imdb	26036	3364566	1888-2017	6.39153819269157	1.1735844383219116

Table2 Information about IMDB dataset

2) MovieTwitting dataset

The MovieTwitting dataset can be downloaded directly from https://github.com/sidooms/MovieTweetings. There are two separate tables used in this research: movies.dat and ratings.dat. There are originally 26,044 records for movies.dat and 649,320 records for ratings.dat.

- Variables Introduction

movies.dat Variables	imdb_ID (key)	Title	Genre
Attribute	string/num	String	String

*Table 3 Variables in MovieTwitting dataset *movies.table(after pre-processing)*

rating.dat Variables	imdb_ID (key)	Twitter_UserID	TwitterRating	TwitterTime
Attribute	String/num	String	Float	UnixTime

*Table4 Variables in MovieTwitting dataset *rating.table(after pre-processing)*

- Records Explanation

The total movies count is derived from rating.dat's imdb_ID. Although we have 649320's ratings, total 29290's movies have been rated. The time period for all movies is 2013-2017. The overall mean is relatively high, and we have a relatively larger rating standard deviation.

Table5 Information about MovieTweetering dataset

Dataset	Total Movies Count	Total Votes Count	Movie Time Range	Rating Mean	Rating Std
Twitter	29290	649320	2013-2017	7.303748536930943	1.861147111350471

Data Manipulation Methods

In this chapter, I will explain go through my source code and logistic first, then explain how I preprocess my data, including data clean, debugging and deals with incorrect and redundant data. Then I will explain explicitly about Spark and Hadoop part with source code illustrated as well.

1) Data Pre-Processing

- SourceCode WorkFlow

In order to crawl data on OMDB API, I need to have original imdb_ID as key to get those jsons. I first cleaned the original data in movie.dat, adding "tt" to all numbers of ID in that file in order to fit the format of imdb_ID. Then I use the request format "http://www.omdbapi.com/?i=" + str(id) + "&apikey=?" to get 26036 records of movie information in json format .

Next step is Large Scale Data Computation and analysis. I first upload those raw data into hadoop (Flux) System and then used Map function to fetch all the variables I wanted into RDD and then transferred them as DataFrame. In order to do further SparkSQL analysis, I also renamed them and registered them temporary data table, more details will be explained later.

Final step is for data visualization. I used R studio as visualization tools, and used all the csv files produced in hadoop system. The main library I use for R visualization is ggplot2 library. I also used lattice package's xyplot in order to plot two variables into one plot and review their relationships. Reference link is https://www.stat.ubc.ca/~jenny/STAT545A/block09_xyplotLattice.html.

- Time Variable Conversion

I manipulate two formate conversion here to regularize with the time variables.

- Unix Time transformation

The time format of rating.dat is UnixTime, thus I transferred them to date time type first and then to timestamp with year as an integer output. For time object in movie.dat is a string type, like '10 Jan 1994', I extracted the year part from the string and store as integer output as well. In most of my analysis, I only use year to present the date variables because it appears more visible when making visualizations and are more useful as time series analysis to see trend.

- Dealing with 'Year Out Of Range' problem

For time series analysis, where I can analyze in how many days twitter users rate movies most after movies released, we still need complete transform the time string in movies datable and unix time in twitter datable all into datatime objects in order to retrieve TwitterTime minus ReleasedTime.

The problem of time.strptime method cannot recognize for years that were so old.. For example if I transfer "10 Jan 1830" to (1830,1,10,0,0) will product an error of "Year Out Of Range". So what I do was I uniform them all into iso-format when I want to do time series analysis later. After match the movie_imdb data table with the twitter data table, I got the matched records in both tables, then I start to map those date with time.strptime from iso-format and thus avoid the error.

- Redundant data and Incorrect data manipulation

In originally movies information, each movie have lots of genres, for computation convenience and visualization, I only use the first genre type as their genre.

There is one genre named ".N", which belong to uncategorized genre, so I removed them from the data.

- Variables and steps used to JOIN IMDB data and twitter data

The common key for two dataset is imdb_ID. For most cases, I join two datasets in order to get genre information to explain deeper information about certain types of film. In some cases, I need BoxOffice data and Country data from IMDB dataset to unearth their relationship with users' total votes and users' ratings. In terms of twitter data, I am curious about twitter users sharing voting behaviors, thus their rating amounts, rating score and rating time are the major selected variables to be shown in the analysis output.

2) Data Manipulation

- Map

I used to map function to do data cleaning and also read my json file and I used flatmap to pass a data selection function into the map.

Examples of data cleaning is:

```
twitter = twitter_r.map(lambda r:r.split('::'))
rating_tw=twitter.map(lambda id:
[id[0],'tt'+str(id[1]),id[2],t_date_tf(id[3])]).toDF()
```

Examples of read a file and then pass a function using flatmap is:

```
movie_data = imdb_m.map(lambda line:
json.loads(line)).flatMap(function).toDF().cache()
```

- SparkSQL

1) Question 1: What are the differences between the most popular and preferred genres in IMDB and twitter platforms?

To calculate most popular genres in IMDB, I use count(totalVotes) from IMDB datable,combined twitter datable with IMDB datable on imdb_ID, grouped by genre, and finally order based on totalVotes number on descending order.

```
sqlContext.sql('select Genre, count(twitterRating) AS twitterVotes from
twitter_data JOIN movie_imdb ON twitter_data.imdbID= movie_imdb.imdbID group by
movie_imdb.Genre order by twitterVotes')
```

In terms of ratings, I used average rating as evaluation of public preference. I added all of the ratings based on Genres and then divided by the count(imdbRating numbers) as average rating score, and the sort them by descending order.
The same process applied to twitter dataset as well.The reason I separate those processes are for better visualization concerns.

```
sqlContext.sql('select Genre, sum(twitterRating)/count(twitterRating) AS
twitterAvgRating from twitter_data JOIN movie_imdb ON twitter_data.imdbID=
movie_imdb.imdbID group by movie_imdb.Genre order by twitterAvgRating DESC')
```

2)Would twitter users more likely to share their reviews(ratings) for old movies or for most recent movies?

To analyze twitter users rating time range, I use Twitter Users rating time minus IDMB movie released year as period, and then count how many times those periods occurs group by period and sort them based on count(period) in descending order.

```
select(twitter_data.twitterTime-movie_imdb.Year),count(twitter_data.twitterTime-
movie_imdb.Year) AS TwitterVotes from twitter_data JOIN movie_imdb ON
twitter_data.imdbID= movie_imdb.imdbID group by (twitter_data.twitterTime -
movie_imdb.Year),twitter_data.imdbID Order by TwitterVotes DESC’)
```

To analyze how many days a twitter user would rate the movie after those movies released, I first joined the movies data table with twitter data table based on their common imdbID so that only time range from 2013-2017 will be selected and thus help me increase the proficiency as well as avoid the time.strptime error because of year out of range.

```
# join two datasets together

t1= sqlContext.sql('select * from imdb JOIN twitter_t ON imdb.imdbID =
twitter_t.imdbID’)
```

And after that, I transformed them to timestamp in data frame and user map date.days to see how many days they will rate after the movie released.

```
# use map to get one as their time period

t_final = t1.rdd.map(lambda x:[x[0],x[1],x[2],x[3],x[4],x[5],x[6],x[7],x[8],x[9],
(x[12]-x[8]).days,x[11],x[12]]).toDF().cache()

… ignoring process of transform them to data frame with desired columns

time_vote_total = sqlContext.sql('select time_period, count(imdbRating) AS
TotalVotes from twitter_imdb group by time_period order by TotalVotes DESC')
```

After having a general information, I am most curiously about the action movie and biography movie and I wanna analyze how user's rating pattern per genre. So simply I just select the Genre=="Action" and Genre=="Biography". For computation convenience, I only selected the most hotted action movies as well biography movies.

```
action = sqlContext.sql('select Title, sum(twitterRating)/count(twitterRating) AS
twitterAvgRating, count(imdbRating) AS TotalVotes, time_period from twitter_imdb
where Genre="Action"  group by Title, time_period order by TotalVotes DESC LIMIT
1000')
```

After saw the patter for biography movie, I wondering if I can separate biographies with ratings before they released with biographies doesn't receive before ratings average rating scores to see whether trailers can help biographies receive more reputation.

I first select all those biographies movies who have time range < 0 , which means users rate before released time, so the number will be smaller than 0.

```
bio =  sqlContext.sql('select Title, sum(twitterRating)/count(twitterRating) AS
twitterAvgRating, time_period from twitter_imdb where Genre="Biography" AND
time_period<0 group by Title, time_period')
```

Then I select the titles in "bio" data table that have common titles with the original biography data. I stored them as name_list in order to select them from biography's Title Columns. I used col function from pyspark.

```
biography= sqlContext.sql('select Title,sum(twitterRating)/count(twitterRating) AS
twitterAvgRating, time_period from twitter_imdb where Genre="Biography" group by
Title, time_period')
filter_bio=biography.where(col('Title').isin(name_list))
filter_bio.registerTempTable("filter_bio")
```

Then I calculate average ratings in biographies with trailers and without by select the total sum a movie would have and the total number of reviews they receive.

```
# calculate biography movies with preview

filter_bio_score = sqlContext.sql('select sum(twitterAvgRating)/
count(twitterAvgRating) AS previewAvgRating from filter_bio')
```

```
# calculate biography movies without preview

bio_no= sqlContext.sql('select Title,sum(twitterRating)/count(twitterRating) AS
twitterAvgRating from twitter_imdb where Genre="Biography" AND time_period>0 group
by Title')
bio_no.registerTempTable("bio_no")
bio_no_score = sqlContext.sql('select sum(twitterAvgRating)/count(twitterAvgRating)
as nopreviewwAvgRating from bio_no')
```

To also analyze whether twitter ratings to different genres change across time, I just add TwitterTime variable into the query and group it with not only Genre but also TwitterTime.

```
sqlContext.sql('select twitterTime,Genre,
count(twitterRating),sum(twitterRating)/count(twitterRating) As twitterAvgRating
from twitter_data JOIN movie_imdb ON twitter_data .imdbID= movie_imdb.imdbID
group by twitterTime,Genre order by twitterAvgRating,count(twitterRating) DESC')
```

3)Do movie exposure to Twitter boost their ticket sales? In other words, analyze the relationship between BoxOffice with total votes and rating scores?

I excluded all movies that doesn't have BoxOffice records first. Then I grouped those movies with their genre, and the select genre, count(TotalVotes) and TwitterAvgRating as we did before, and totalTicketSales from the twitter data JOIN IMDB data on twitter.imdbID= movie.imdbID, then finally ordered them based on the TotalVotes of twitter.

```
sqlContext.sql('select count(twitterRating) AS twitterVotes, ticket_sales,Genre,
sum(twitterRating)/count(twitterRating) AS twitterAvgRating from twitter_data
JOIN movie_imdb ON twitter_data.imdbID= movie_imdb.imdbID where ticket_sales !=
"N/A" group by Genre order by twitterVotes DESC')
```

Analysis and Visualization

In this chapter, I will demonstrate the final results that come with my three initial research questions and trying to illustrate the pattern and trend by presenting visualizations. I will then make some guesses based on those visualizations.

Exploration1 : User preferences to Genre

- Basic Data Exploration:

1) Popular genres in IMDB and twitter are similar. Top 3 genres are in the range of Comedy, Drama and Action

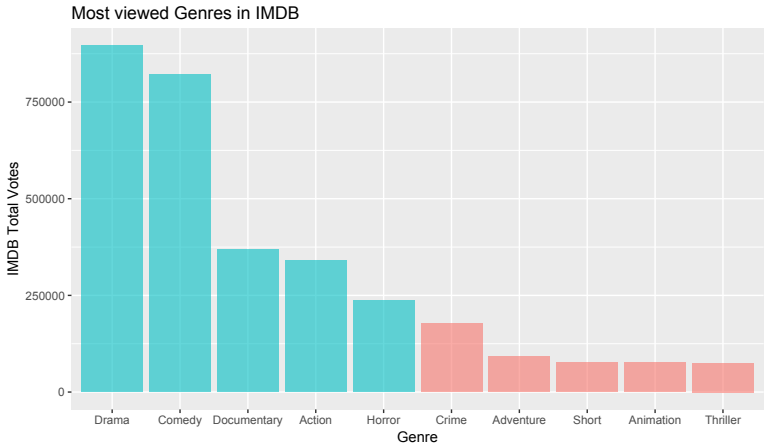


Figure 1 Popular Genres in IMDB

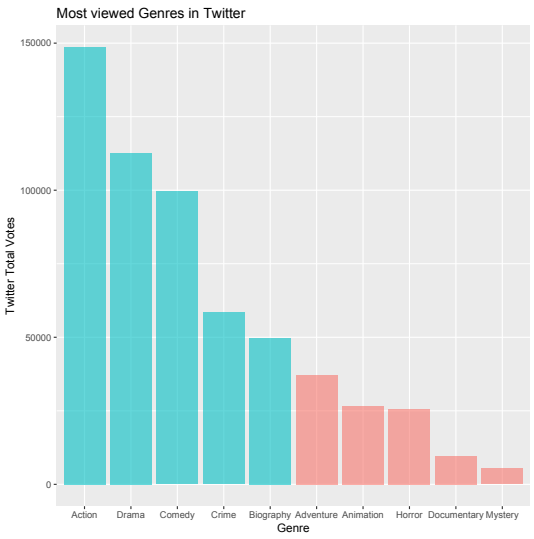


Figure 2 Popular Genres in Twitter

2) Most highly rated genres in IMDB are Film, Documentary, Biography, Animation and War. While most highly rated genres in Twitter are Western, Short,Documentary, Biography and Film.

As we observe those two ratings per genre visualizations, we can find that the ratings patterns between different genres scores of IMDB are more smoother than twitter's data. (*note in here Game is actually an outliner, it has the least voters - 47 people - in IMDB) Although twitter users give an average high score to different genres, scores between different genres change sharply. From that we can see twitter users have more obvious preference to certain genres than IMDB users do. Thus twitter users rating are more likely to be influenced by the genre itself. From MovieTweeting's genre preference, film companies should analyze their rating behaviors toward different genres and make propagation plans thusly. All in all, it is definably worth to analyze twitter users MovieTweeting behaviors and intentions in future research.

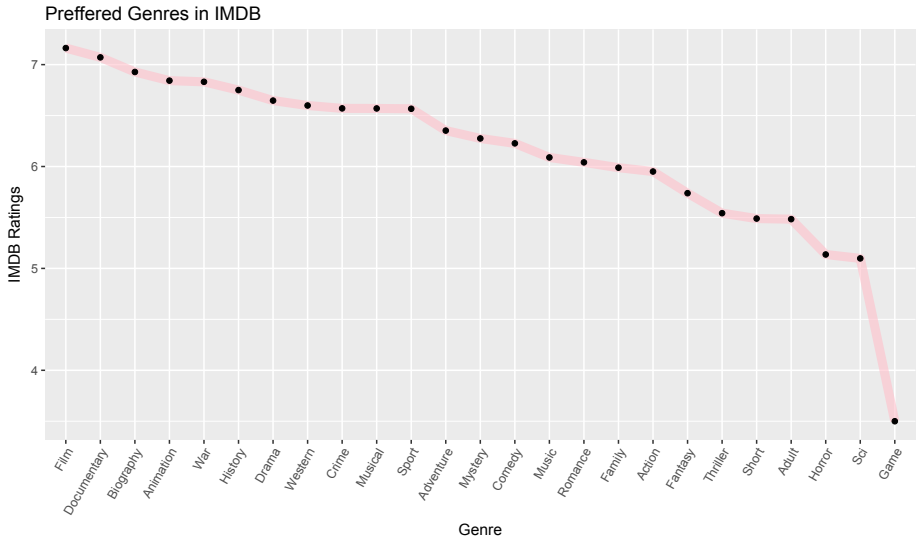


Figure 3 Most highly rated Genres in IMDB

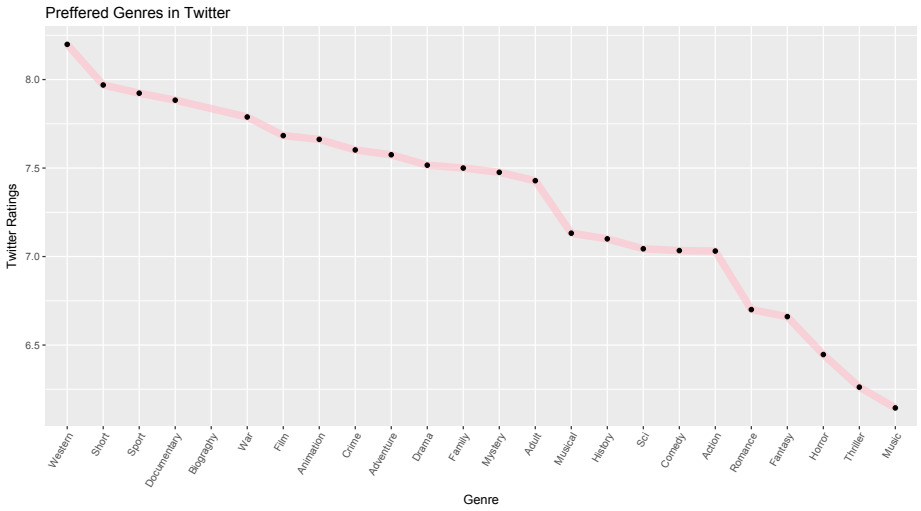


Figure 4 Most highly rated Genres in Twitter

- Interesting findings:

1) Documentary and Biography are both being most mentioned and rated high for both IMDB and twitter. Twitter users prefer to share biography movies and IMDB users voted a lot for Documentary.

From this result, we will advise Documentary and Biography movie directors to do movie propagation in twitter before they release.

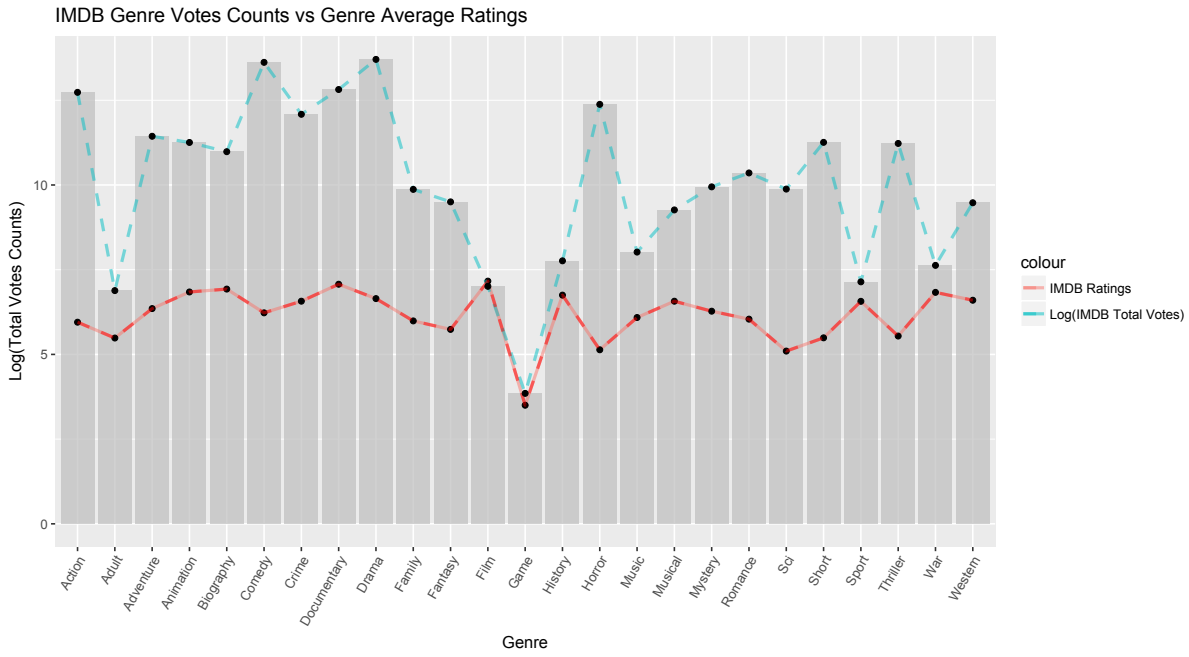
IMDB (TotalVotes)	Twitter (TotalVotes)	IMDB(ratings)	Twitter (ratings)
Comedy 6.22	Action 7.03	Film	Western
Drama 6.65	Drama 7.51	**Documentary**	Short
Action 5.98	Comedy 7.03	Biography	Documentary
Documentary 7.07	Crime 6.598	Animation	**Biography**
Crime 6.57	**Biography** 7.86	War	Film
Horror 5.14	Adventure 7.58	History	Animation

Table6 Most mentioned and highly rated in Genres in IMDB and twitter data

2) Movie genres that being mentioned more usually not being rated too high.

As we can see from figure 5, we can see Action,Comedy, Drama, Horror in IMDB dataset have large votes audiences, but they receive relatively low rating score. Here, in order to plot ratings and votes on a same y scale, I use log function on IMDB total votes to see a zoom in version.

Figure 5 Most mentioned vs most highly rated Genres contrast in IMDB



However, if we observe patterns on twitter's data, we will find the contrast between audience attention and audience evaluations are much more smaller than in IMDB's data. For example, Action movies receive lots of attention from both IMDB and Twitter, the difference is that Twitter users give them average 7.03 score , while IMDB only give them 5.98 score. The same happens with Drama movies, they get 6.65 on IMDB but receive 7.51 high score on Twitter. From this point, we can conclude that twitter users are not "strict" grader as general IMDB users are.

As we know IMDB has total votes of 3364566, while Twitter has total votes of 649320, which is 6 times smaller than IMDB rating amounts. Although twitter and IMDB users have similar preference for certain genres, however film companies should expect to see high scores posted from twitter users. They could then try to win twitter users hearts and encourage twitter users post their relatively high ratings to appeal potentials audiences to increase their BoxOffice Sales.

Also there are some special cases in Twitter as well, some movies genres like Family, Film, Sci, Short and Western receive relatively small attention but received higher average score than other genres.

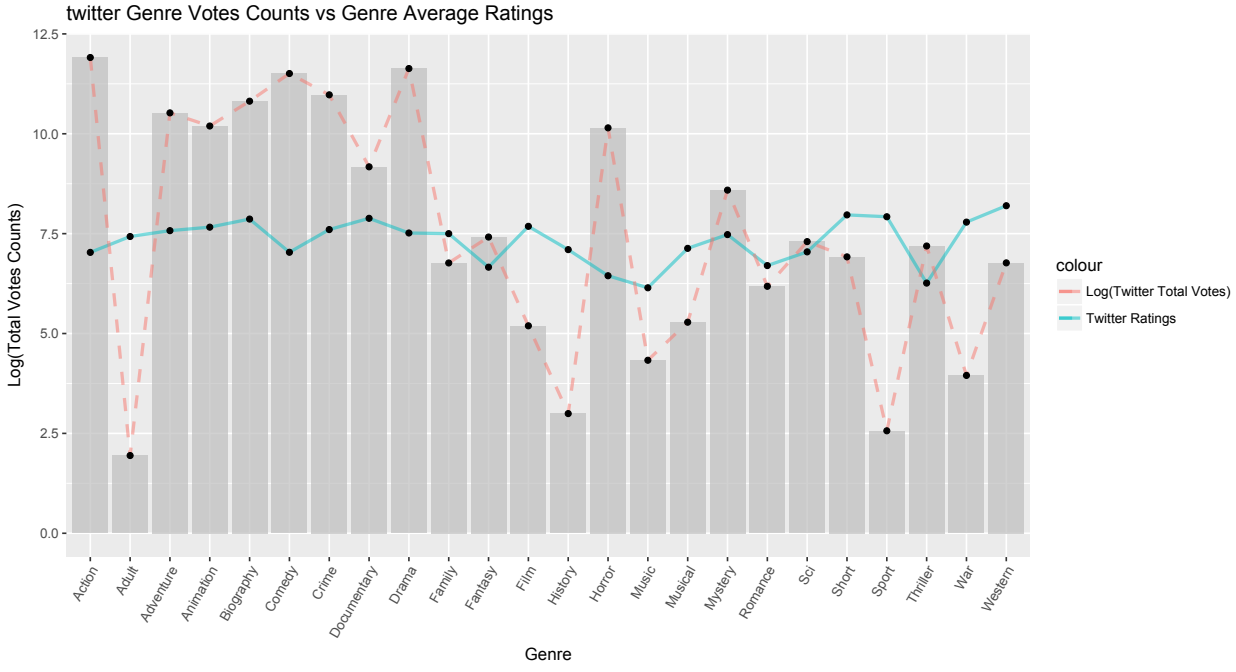


Figure 6 Most mentioned vs most highly rated Genres contrast in Twitter data

Exploration 2 : Twitter Ratings time series analysis

In this exploration section, we want to explore the fact whether twitter users are more likely to share their reviews(ratings) for old movies or for most recent movies(Movie release time>1).

As it turns out, twitter users rating behaviors have timeline effect.

From the output, most of the ratings(172332 out of 649320) vote for most recent movies, and the number decreases through time.

```
Row(_c0=0.0, _c1=172332),Row(_c0=2.0, _c1=37038), Row(_c0=1.0,
_c1=162307, Row(_c0=5.0, _c1=12993), Row(_c0=4.0, _c1=15543),
Row(_c0=3.0, _c1=22190), Row(_c0=16.0, _c1=5405), Row(_c0=14.0,
_c1=5700), Row(_c0=13.0, _c1=5841), Row(_c0=12.0, _c1=6198),
Row(_c0=11.0, _c1=6580), Row(_c0=10.0, _c1=7467), Row(_c0=9.0,
_c1=8543), Row(_c0=8.0, _c1=8882), Row(_c0=7.0, _c1=9982),
Row(_c0=6.0, _c1=11683), Row(_c0=5.0, _c1=12993)……
```

If we keep on zoom to this phenomenon, we wanted to discover how many days after movie release will the the highest boost for twitter users to vote. As Figure 7 turns out, although the pattern is quite smooth from the on two tails, the most of the movie are rated in 10 days. The top rated days with descending order is day(1,2,0,3,9,8,4,5,7,6).

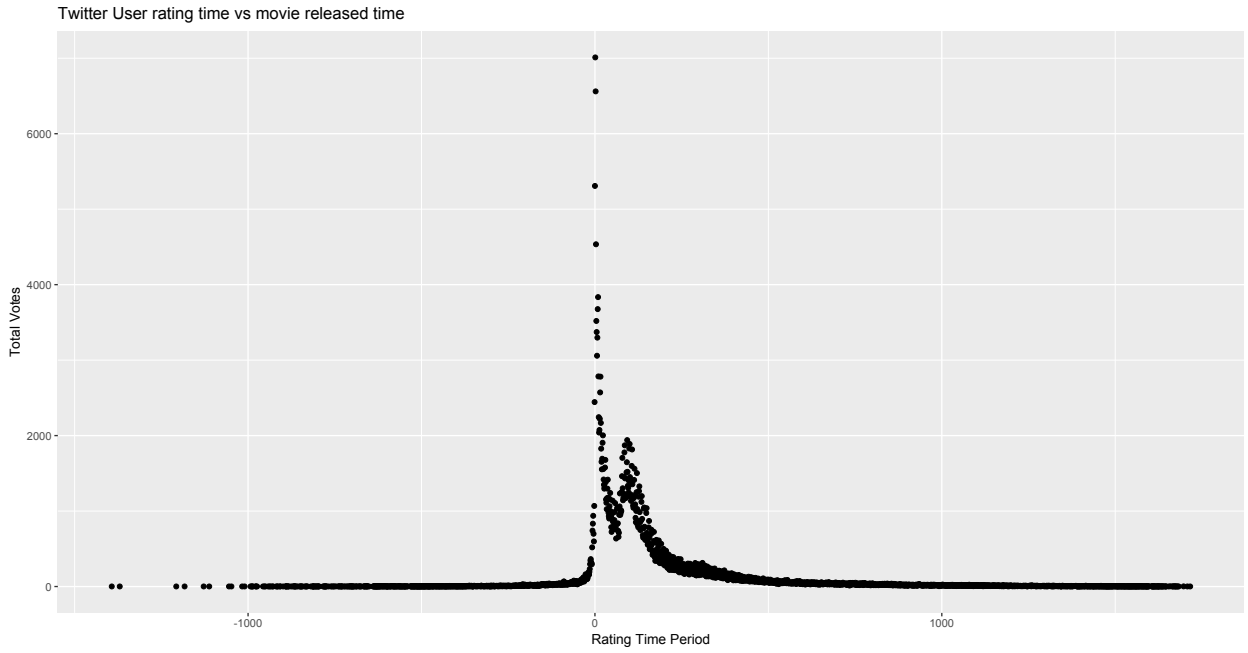


Figure 7 Twitter users voting time after movie released

If we search deeper to the genre pattern of twitter rating, we can find that for action movies, their Total Votes have very clear boundaries. Although most people votes for movies in 10 days to 20 days after they release, some people love to share their rating to their preference for classic action movies.

However if we look at Figure 9, biography movies rating time, we cannot see a very obvious patter in here, which means biography movies don't pursue for time effect. Interesting, we can see lots of biography have ratings before they release. So they must put their trailer on internet before they released and twitter users love to rate them!

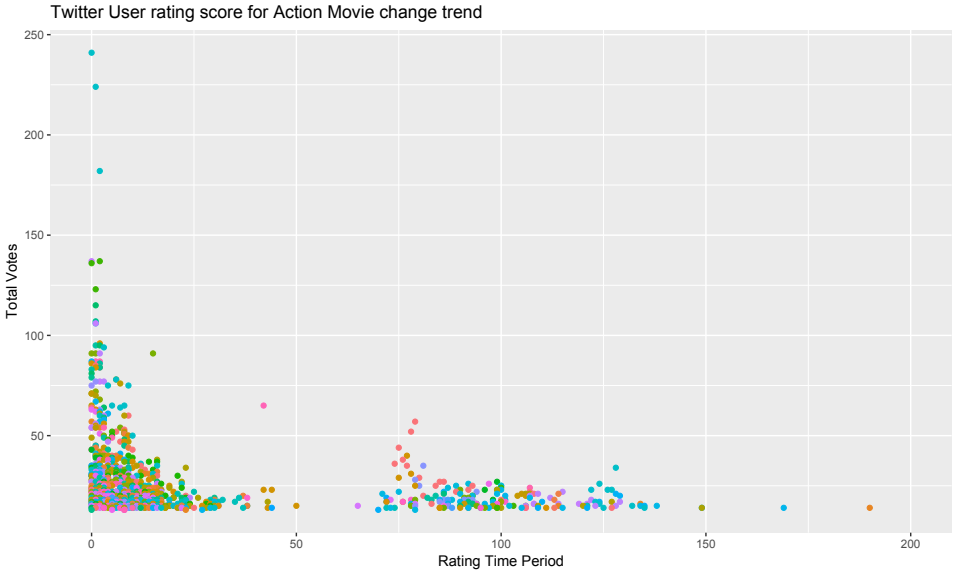


Figure 8 Twitter users voting time after Action movie released

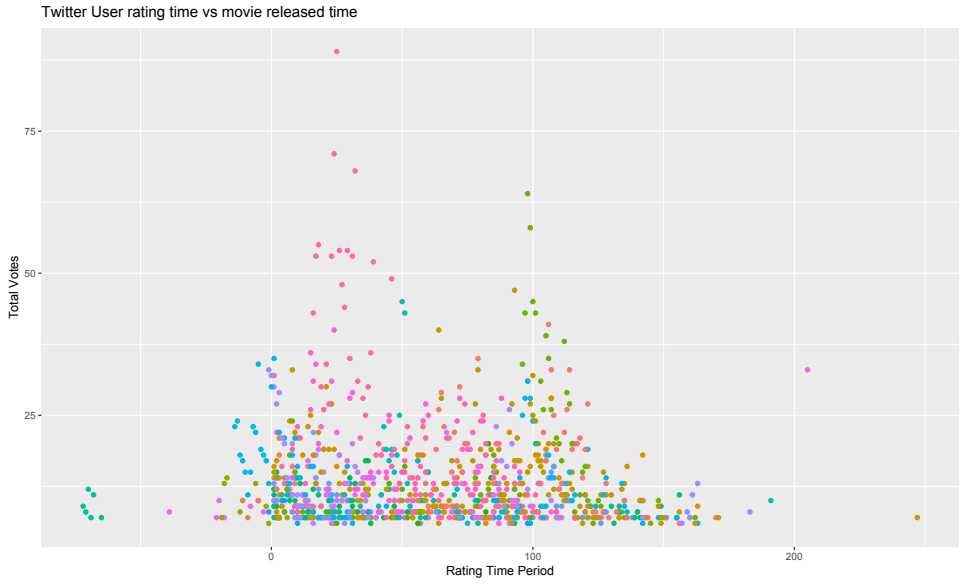


Figure 9 Twitter users voting time after Biography ction movie released

So knowing that twitter users rate biography trailers, I was wondering whether this preview version is the reason that make biography get higher score/reputation.

Thus I compared the movies with trailers and movies with trailers and wanna see their average score differences. Turns out, from Table 7, biography movies companies who publish their trailers in advance and receive twitter users' rating will gain much more higher rating.

From that, we can reasonably guess biography movies reputation have close relationship with their media exposure.

Type	previewAvgRating	NoPreviewAvgRating
Average rating score	7.527802159507689	7.132364885279287

Table7 Twitter Rating changes for Biography with Trailer and without Trailer

Beside of that, we also curious about how twitter users change their 'Genre taste' across time.

From figure 7 we can find some interesting insights. Notice the horror movies are receiving higher and higher ratings through year. And also the same for Action, Comedy and Mystery. While Adventure movie decrease their ratings quite a lot through time. The most steady high performance candidates are Animation and Documentary.

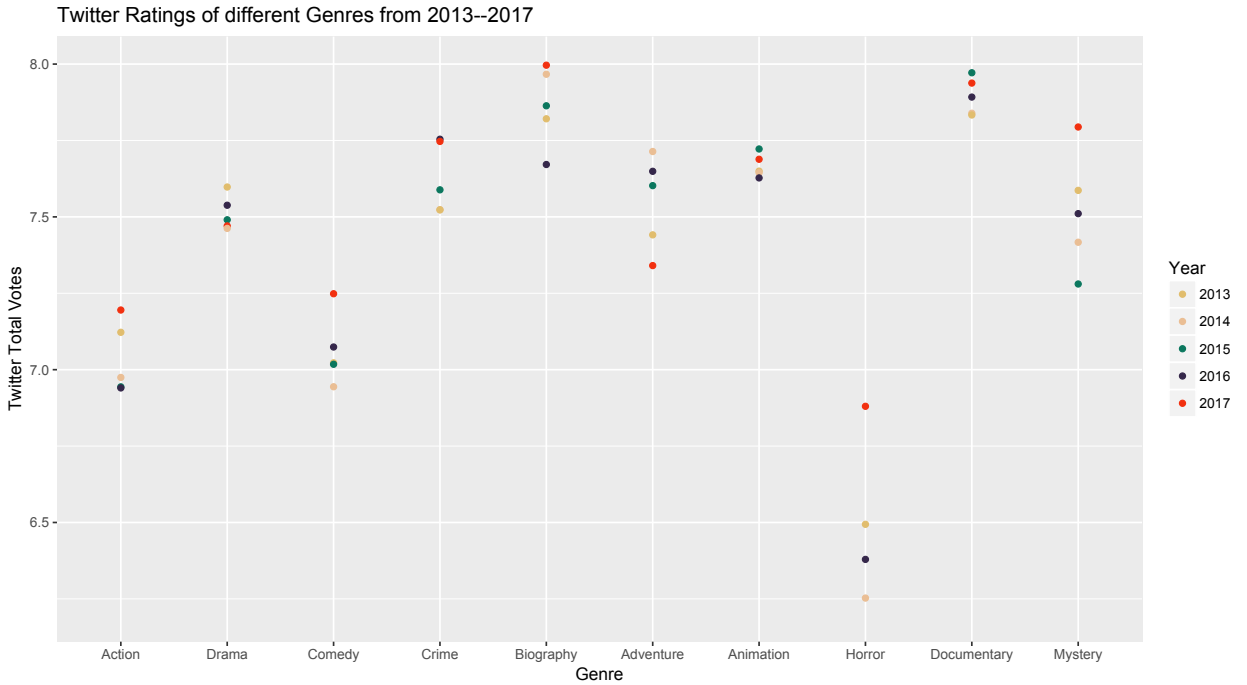


Figure 10 Twitter Ratings per genre change through time

Exploration 3: Movies Exposure in twitter relationship with their BoxOffice performance

When talking about films exposure in twitter, I mean both the movie TotalVotes count and their average rating scores. From figure, we can see that there is possible relationship between Twitter Votes count with TicketSales, while there is no clear linear relationship between twitter ratings with TicketSales.

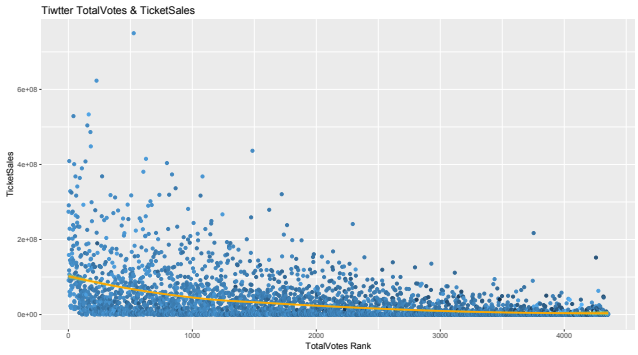


Figure 11 Twitter Total Votes relationship with TicketSales

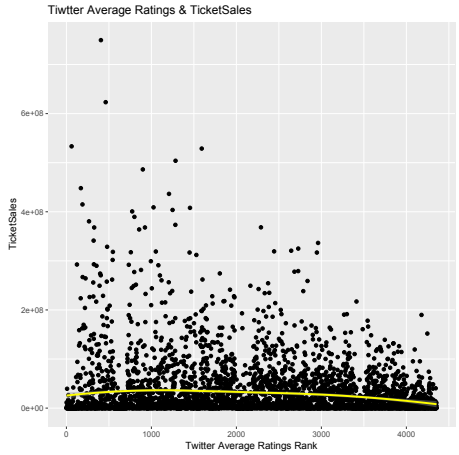


Figure 12 Twitter Average Ratings relationship with TicketSales

However from figure , it's pretty hard to see a clear linear relationship between TotalVotes with the TicketSales. Thus I grouped those data with genres and to see if there is correlation between them. As we can see here(figure 9), after using log transformation of TicketSales, there are positive linear regression between BoxOffice with Twitter TotalVotes counts for Animation, Biography, Action and Adventure movie. Among those movies, biography movie has a most obvious linear correlation.

There must be other features contribute to movies' ticket sales, but through visualization, it was good to know that twitter votes amounts can have an influence on ticket sales.

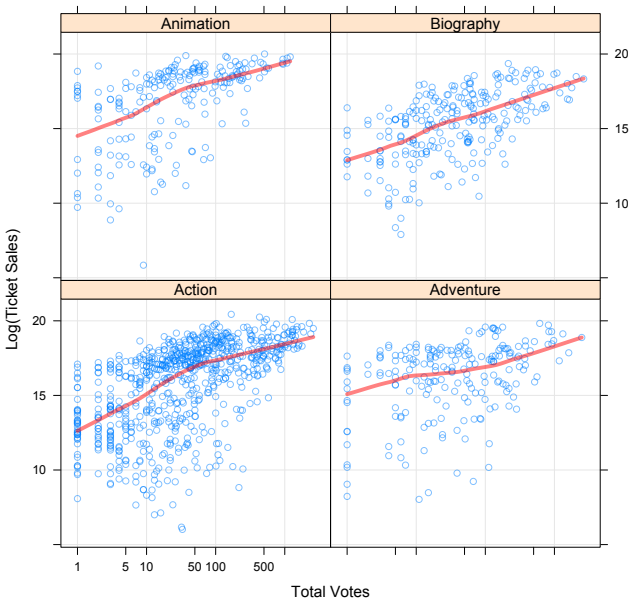


Figure 13 Twitter Total Votes relationship with TicketSales grouped by Genre

Challenges and Problem Solving

There are many challenges explained in data pre-processing part. Other problems I encountered are:

1) I found some packages and library doesn't suit for Spark and our school's spark system is under an older version. Thus to solve this problem, I have to change a date time transformation function from date time string to strptime and datetime.datetime. Also with datetime module we need to notice that datetime has a module name as well as class in module to be referenced, when reference both of them we best specify which module or class we are calling. A better answer in Stackoverflow is "import datetime as dt // from datetime import datetime" (https://stackoverflow.com/questions/15707532/python-import-datetime-v-s-from-datetime-import-datetime)

2) Also worth mention here is that, when crawling for IMDB data, researchers better write the json file as text object into a file when crawl OMDB instead of using json.dumps( ). In this case, I used the requests library and store requests object with .text method when writing into my file. The reason is that the IMDB dataset is currently updating some movies names(mostly foreign movies) with translated English names, thus in their original website, there are two names listed on the website, which can make the API confused and thus return 504 Error, if we don't want to miss those movies(don't use try and except) and keep the file continue writing, the best way would be to write as pure text format and write the error message into the file in order to fix them later.

3) Also I encountered some problem when trying to join same rows instead of same columns. A better solution is to use col function in pysparkSQL because that will return you whole data frame.