International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 07 | July 2020 www.irjet.net p-ISSN: 2395-0072

Analysis of Movie Data to Gain Insights into Cinematography History

Sanjith K

Department of Computer Science and Engineering, BMS College of Engineering, Karnataka, India

---

Abstract -*The Movie data used for the analysis consists of various files with large amounts of data which contain information like movie names, the ratings of each movie from various users, and some tags associated with each movie. Each of these datasets are processed and analyzed to gain various insights into cinematic history by using R studio software and R programming language.*

***Key Words*: R programming language, R studio, cinematic history**

1. INTRODUCTION

This analysis of Movie data involves various process but before any data can be subject to analysis it must be prepared. This data preparation involves data cleaning and organisation of data to make it easier for the computers to process and the removal of unwanted and null data from the data set will increase speed of processing and reduce processing time. The data is then converted into data frames using R studio and various inbuilt packages in R studio are used to analyse the data by using a question and answer approach in which the answers will reveal the insights of cinematography history.[2]

2. DESCRIPTION OF THE DATA BEING USED

2.1 movies.csv

This file contains information about movieId, movie title and genre. Each row in the file indicates all these details for each movie. This data is gathered from imdb website and each movie also has a genre like action, adventure, animation, comedy, children's, crime, documentary, drama, fantasy, film-noir, Horror, musical, mystery, romance, sci-fi, thriller, war, western. If the movie cannot be associated with any genre then the column will have "no genre listed" in the corresponding row.[1]

2.2 ratings.csv

This file consists of ratings for all the movies listed in the movies file. Each line of this file after the header row represents one rating of one movie by a single user, and has the format: userId, movieId, rating, timestamp. The lines in this file are ordered by first userId and then within each user by movieId. Ratings are on a scale of five with 0.5 increments (0.5-5). Timestamps present in the file represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.[1]

2.3 tags.csv

This file contains various tags associated with each of the movies present in the movies file. Each line of this file represents one tag applied to one movie by one user, and has the format: userId, movieId, tag, timestamp. The lines within this file are ordered by userId, then, within user, by movieId. Tags are user-generated metadata about movies. Each tag is normally a single word or short phrase. The meaning, purpose, and value of a tag is determined by each user. Timestamps present in the file represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.[1]

3. PACAKAGES USED

tidyverse- tidyverse is a collection of packages that work in consonance because they share familiar data representations and 'API' design.[3]

lubridate- The 'lubridate' package has a consistent and easy to remember syntax that makes working with dates easy. [4]

stringr- This package has several functions used for character manipulation, whitespace manipulation and pattern matching.[5]

rvest- Package that makes it easy to scrape (or harvest) data from html web pages.[6]

xml- This package has tools for parsing and generating XML Within R.[7]

tidytext- This package is used for text mining for word refinement and sentiment analysis using dplyr, ggplot2, and supplemental tidy tools.[8]

wordcloud- Functionality to create pretty word clouds.[9]

dplyr- this package is used for text manipulation.[3]

tidyr- this package is used for data tidying.[3]

ggplot2- This package is used to create visualizations of data life graphs and charts.[3]

4. DATA LOADING AND CLEANING

The data in the above-mentioned files is loaded iteratively into the workspace using read_csv() function and is assigned variable names accordingly. The read_csv() function is very suitable because it automatically identifies column types based on the first 1000 rows and it never converts strings to factors. The read_ csv() function coverts the data into data frames and each of the data frames are of various sizes. The object size of the data frames created are as follows: movies frame size is 3.8 Mb, ratings object size is 465.5 Mb, tags frame size is 15.7 Mb as shown in Fig 1. [1] [2]

```
[1] "movies object size is 3.8 Mb"
[1] "ratings object size is 465.5 Mb"
[1] "links object size is 2.5 Mb"
```

Fig-1: Size of each of the data frame object sizes

The next step involves the process of data cleaning for data optimisation and to ensure the data quality is very high and increase overall efficiency of the analysis process.[2]

In the ratings data frame, it is seen that it has 24 million rows and 4 columns. All the data is in proper usable format except the timestamp column. We create a new data frame with the same name and copy all the data and then convert the timestamp to proper format using the as_datetime() function to convert the entire timestamp column into a proper format. [2][4][10]

```
ratingsdf <- ratings
  mutate(timestamp = as_datetime(timestamp))

summary(ratingsdf)
```

Fig-2: Code to convert timestamps into required format

```
Observations: 40,110
Variables: 3
$ movieId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,...
$ title   <chr> "Toy Story (1995)", "Jumanji (1995)", "Grumpier Old Me...
$ genres  <chr> "Adventure|Animation|Children|Comedy|Fantasy", "Advent...
```

Fig-3: Summary of movies data frame object

The movies data frame has 40 million rows and 3 columns. It is seen that movie names and their debut years are attached together so to separate into individual columns we use extract() function to separate movie names and their debut dates and then add then create a separate column for debut dates and also there are several movies which have no genre listed for them so each of them is replaced by 'NA' value.[3]

```
Observations: 668,953
Variables: 4
$ userId    <int> 28, 40, 40, 57, 73, 98, 98, 98, 98, 98, 98, 98, 141,...
$ movieId   <int> 63062, 4973, 117533, 356, 81591, 55247, 55247, 56174...
$ tag       <chr> "angelina jolie", "Poetic", "privacy", "life positiv...
$ timestamp <int> 1263047558, 1436439070, 1436439140, 1291771526, 1296...
```

Fig-4: Summary of tags data frame object

The tags data frame also does not have the timestamp in proper format so, datetime() function is used again here to convert all timestamps into the proper format. [2][4][10]

```
tags_df <- tags
  mutate(timestamp = as_datetime(timestamp))

summary(tags_df)
```

Fig-5: Code to convert timestamps into required format

5. DATA VISUALIZATION

Data visualization involves the process of exploring the datasets, which have been modified to increase the efficiency of data processing and the accuracy the results, to reveal various fascinating facts about the cinematography and its evolution through time.[2]

The first query we will seek to answer using the datasets is what were the number the movies produced in various years? This information is obtained by using the movies data frame suing the code shown in Fig-6.

```
moviesperyear <- moviesdf
  na.omit()
  select(movieId, year)
  group_by(year)
  summarise(count = n())
  arrange(year)
knitr::kable(head(moviesperyear, 10))
```

Fig 6. Code to find number of movies produced in various years

year	count
1874	1
1888	2
1890	1
1891	3
1892	1
1894	2
1895	2
1896	3
1898	4
1899	1

Fig-7: Glimpse of movies produced in a few years

There exist some years in the dataset which show no movies were produced at all, so we use the complete() function from the tidyr package to fill them as zero in the count.[3]

```
moviesperyear <- moviesperyear
  complete(year = full_seq(year, 1), fill = list(count = 0))
knitr::kable(head(moviesperyear, 10))
```

Fig-8: Code to add zero to count in years where no movies were produced

year	count
1875	0
1876	0
1877	0
1878	0
1879	0
1880	0
1881	0
1882	0
1883	0

Fig-9: Years in which no movies produced with word count changed

The data of all the movies produced and years they were produced in are plotted in a graph using ggplot() function and analyzed to see the number of movies produced in a year and how it varied over the years.[3]

```
movies_per_year %>%
  ggplot(aes(x = year, y = count)) +
  geom_line(color="blue")
```

Fig-10: Code to plot a graph of all movies produced in various years

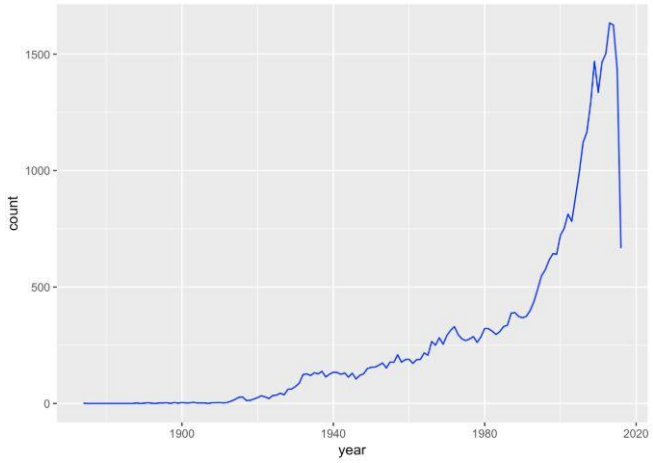

Fig-11: Graph showing number of movies produced in various years

The graph shows an exponential growth and then a drop as the data is only collected till mid of 2016 and since the data of entire year isn't available the drop is seen in the graph. The sudden rise in the graph can be linked to the start of the age of information. The popularity of the internet also had a positive impact on movies and the demand for them also increased dramatically and is still on the rise as of today as well.

The second query we will seek to answer using the datasets is what are the most popular genre of movies that were made? This query is answered by first considering all possible genres and number of movies produced in each genre. The code to generate all genre types and count of movies belonging to each of them is shown in Fig-12.

```
genresdf <- moviesdf
  separate_rows(genres, sep = "\\|")
  group_by(genres)
  summarise(number = n())
  arrange(desc(number))
knitr::kable(head(genres_df, 10))
```

Fig 12. Code to generate genres and count of movies for each of them

genres	number
Drama	17878
Comedy	11438
Thriller	6046
Romance	5485
Action	5095
Horror	3905
Crime	3819
Documentary	3589
Adventure	3031
Sci-Fi	2462

Fig-13: Different Genres and number of movies for each of them

It can be seen in Fig-13. that drama and comedy are the most popular genre by far. The genre popularity for each year is also found and plotted to gain a deeper insight. The code use to find the genre popularity for year is found using the code shown in Fig-14.

```
genrespopularity <- moviesdf
  na.omit()
  select(movieId, year, genres)
  separate_rows(genres, sep = "\\|")
  mutate(genres = as.factor(genres))
  group_by(year, genres)
  summarise(number = n())
  complete(year = full_seq(year, 1), genres, fill = list(number = 0))
```

Fig 14. Code to use to find genre popularity for each year

The data of genre popularity for each year has been gathered now we plot this data to obtain a graph. For intelligibility we consider four genre war, western movies, animation, sci-fi. Fig-15. shows the code used to plot the required graph.

```
genres_popularity
  filter(year > 1930)
  filter(genres %in% c("War", "Sci-Fi", "Animation", "Western"))
  ggplot(aes(x = year, y = number)) +
    geom_line(aes(color=genres)) +
    scale_fill_brewer(palette = "Paired")
```

Fig 15. Code to plot graph of various genre

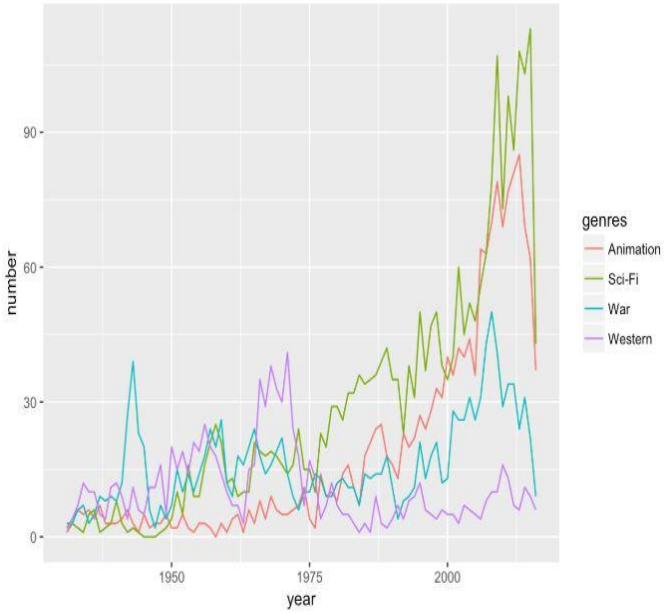

Fig-16. Graph of various genre and changes with time

From the graph we notice that the number of sci-fi movies had a sudden increase after 1969 which corresponds to the first man landing on the moon. We also notice a high number of western movies between 1950s and 1960s which is when the popularity of the west was rising. There is also a rise in animation movies when animation technology became more easier to use and good quality animations could be produced much quicker. War movies were also popular during time periods of world war II, Vietnam war and more lately the war in Afghanistan and Iraq. We can see that the major events throughout history influenced cinematography very deeply.

The third query is what tags will best sum up a movie genre? This query is answered by using the tags data frame object and creating various word clouds using the Word Cloud package in R and its functions to visualize the various tags for each genre in an aesthetically pleasing manner. The first thing to be done is to select the movieId, genres and tags associated with all movies. The code used for gathering this information is shown in Fig-17.[9]

```
genres_tags <- movies_df
  na.omit()
  select(movieId, year, genres)
  separate_rows(genres, sep = "\\|")
  inner_join(tags_df, by = "movieId")
  select(genres, tag)
  group_by(genres)
  nest()
```

Fig 17. Grouping of movieId, year and genre data

```
genre<-"Action"
genre_words <- genres_tags
 filter(genres == genre)
 unnest()
 mutate(tag = str_to_lower(tag, "en"))
 anti_join(tibble(tag=c(tolower(genre))))
 count(tag)
 wordcloud(genre_words$tag, genre_words$n, max.words = 50, colors=brewer.pal(8, "Dark2"))
```

Fig-18: Code to create word cloud of tags associated with each genre type

The code shown in Fig-18 is used to create the word clouds for specific genre. The same code is used for every genre by replacing the genre name whose word cloud needs to be generated.

Fig-19: Word cloud for action genre

Fig-20: Word cloud for comedy genre

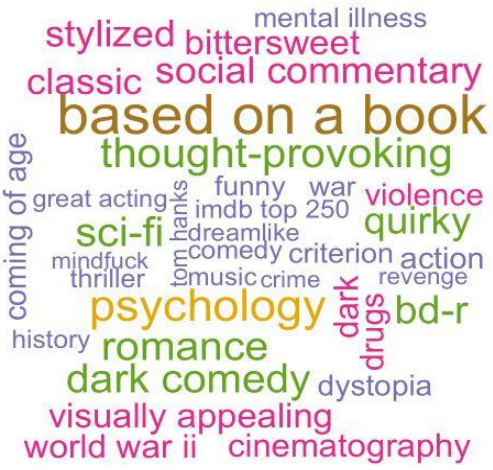

Fig-21: Word cloud for drama genre

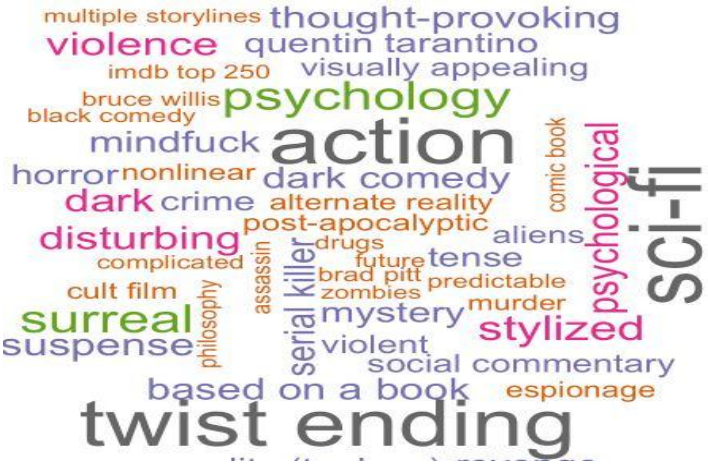

Fig-22: Word cloud for thriller genre

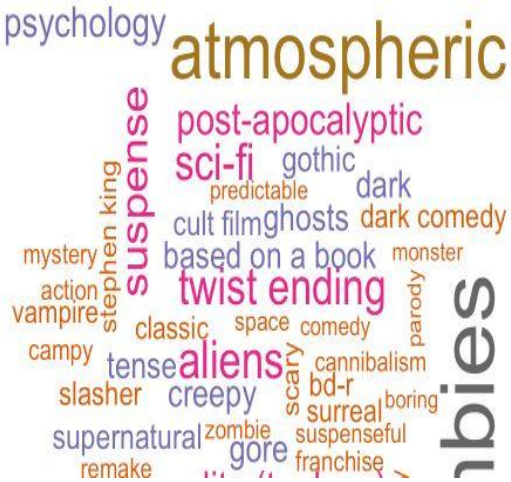

Fig-23: Word cloud for horror genre

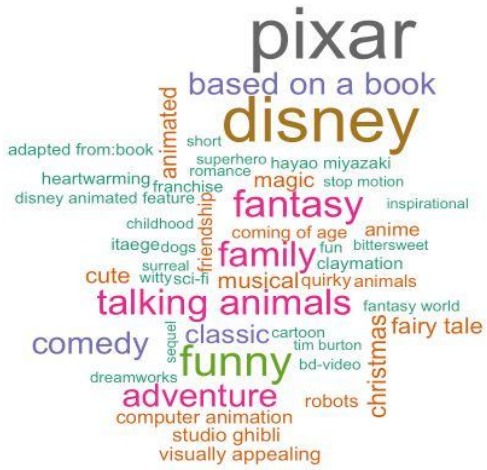

Fig-24: Word cloud for children genre

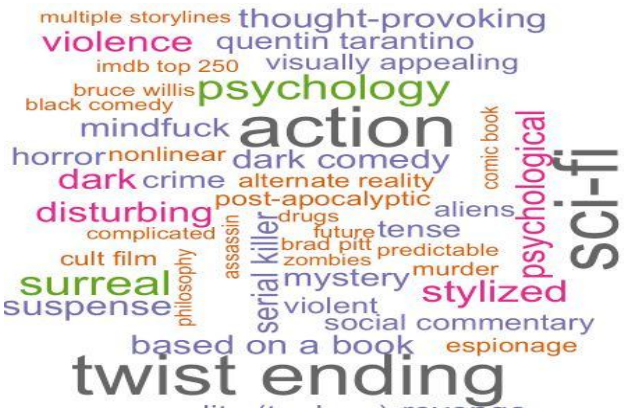

Fig-25: Word cloud for crime genre

Fig-26: Word cloud for romance genre

The fourth and final query is what were the best movies of each decade? We will use user ratings to determine the best movies of each decade. We use a weighted average rating for each movie before ranking. This weighted average method is used by IMDB to compile their top 250 movie list. The code used to compile this list for our datasets is shown in Fig-27.[11]

```
weighted_rating <- function(R, v, m, C) {
  return (v/(v+m))*R + (m/(v+m))*C
}
avg_rating <- avg_rating
  mutate(wr = weighted_rating(mean, count, 500, mean(mean)))
  arrange(desc(wr))
knitr::kable(head(avg_rating, 10))
```

Fig-27: Code to calculate weighted average rating for movies

In Fig 27. R, v, m, C denote the following R – average rating for the movie (mean) , v - number of votes for the movie (votes), m – minimum number of votes required to be listed in the top 250, C - the mean vote across the whole report. [11]

movieId	title	year	count	mean	min	max	wr
356	Forrest Gump	1994	86629	4.047109	0.5	5	0.9942614
318	Shawshank Redemption, The	1994	84455	4.433089	0.5	5	0.9941145
296	Pulp Fiction	1994	83523	4.163386	0.5	5	0.9940492
593	Silence of the Lambs, The	1991	80274	4.153854	0.5	5	0.9938099
260	Star Wars: Episode IV – A New Hope	1977	72215	4.142865	0.5	5	0.9931238
480	Jurassic Park	1993	72147	3.656063	0.5	5	0.9931174
2571	Matrix, The	1999	71450	4.160476	0.5	5	0.9930507
110	Braveheart	1995	63920	4.022716	0.5	5	0.9922384
527	Schindler's List	1993	63889	4.275963	0.5	5	0.9922347
1	Toy Story	1995	63469	3.889300	0.5	5	0.9921837

Fig-28: Shows weighted rating of movies

In Fig-28. the movies with more positive reviews got a higher weighted average rating. To find best rated movie for every decade we use the code shown in Fig-29.

```
best_per_decade <- avg_rating
  mutate(decade = year  %/% 10 * 10)
  arrange(year, desc(wr))
  group_by(decade)
  summarise(title = first(title), wr = first(wr), mean = first(mean), count = first(count))
knitr::kable(best_per_decade)
```

Fig-29: Code used to find best movie for every decade

decade	title	wr	mean	count
1870	Passage de Venus	0.1228070	3.142857	7
1880	Traffic Crossing Leeds Bridge	0.1379310	2.187500	8
1890	Monkeyshines, No. 1	0.1071429	1.250000	6
1900	The Kiss	0.1666667	3.000000	10
1910	Frankenstein	0.3055556	3.159091	22
1920	Cabinet of Dr. Caligari, The (Cabinet des Dr. Caligari., Das)	0.9696418	3.899186	1597
1930	All Quiet on the Western Front	0.9824129	3.934837	2793
1940	Pinocchio	0.9967075	3.449293	15136
1950	Cinderella	0.9952906	3.544809	10567
1960	Psycho	0.9977540	4.066563	22212
1970	MAS*H (a.k.a. MASH)	0.9963873	3.879007	13790
1980	Star Wars: Episode V – The Empire Strikes Back	0.9991331	4.148408	57625
1990	Dances with Wolves	0.9990168	3.741088	50803
2000	Gladiator	0.9988127	3.955173	42062
2010	Inception	0.9982969	4.158438	29308

Fig-30: Best movie for every decade and their rating score

For Fig 30. We do find out best movies for every decade, but we do also notice that the old movies don't get very higher rating scores. The reason for this is that the viewers for the movies were very less and it has nothing to do with the caliber of the movies.

6. CONCLUSION

Analysis of the dataset has revealed many interesting trends, insights into movies and the history of cinematography. Many facts from the number of movies produced in a period to measuring the quality of movies by using their viewer ratings have been examined numerically and visually as well. Generally, it was an intriguing dataset to perform analysis upon using many R programming packages and features.

REFERENCES

[1] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4: 19:1–19:19. https://doi.org/10.1145/2827872
[2] The 5 steps of data analysis process https://medium.com/@kunal_gohrani/the-5-steps-of-the-data-analysis-process-2512ba6ac31e
[3] A beginner's guide to tidverse: the most powerful collection of R packages https://www.analyticsvidhya.com/blog/2019/05/beginner-guide-tidyverse-most-powerful-collection-r-packages-data-science/
[4] https://www.rdocumentation.org/packages/lubridate/versions/1.7.9
[5] Introduction to stringr https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html
[6] Rvest: easy web scaping with R https://blog.rstudio.com/2014/11/24/rvest-easy-web-scraping-with-r/
[7] https://www.rdocumentation.org/packages/XML/versions/3.99-0.3
[8] https://www.rdocumentation.org/packages/tidytext/versions/0.2.4
[9] How to Generate Word Clouds in R https://towardsdatascience.com/create-a-word-cloud-with-r-bde3e7422e8a
[10] https://rdrr.io/cran/lubridate/man/as_date.html
[11] http://answers.google.com/answers/threadview/id/507508.html#:~:text=IMDB%20uses%20this%20famous%20formula,whole%20report%20(currently%206.8)%20This