PRI - Information Processing and Retrieval, 2022/2023

Bernardo Ferreira
up201806581@up.pt

Pedro Pereira
up201905508@up.pt

Telmo Botelho
up201806821@up.pt

1. Abstract

Nowadays with the regular boom of data handy online, it has become extraordinarily essential to come up with data systems able to process, index, and search that information effectively. In this article, we find out about the case of films and respective critiques with statistics handy on the two most famous movie-related web sites on the total Internet: Rotten Tomatoes and IMDB. We use data-sets that are public and available on kaggle. The data was cleaned and prepared for the retrieval phase, with the help of python. In the implementation of the Information Retrieval system, as usual we use Solr platform. An indexation of the documents is done and our collection is defined, after that we selected a set of records desires that would permit us to discover the engine's functionalities and consider the search system.

2. Introduction

The first records of cinema belongs to the end of the nineteenth century. The first public cinema session, organized by the Lumière brothers in 1895, was quick and cheap. For 1 franc each, 33 seats were occupied for about 20 minutes in the basement of a Paris cafe. Since that day, advances in the technological era have allowed incredible changes and achievements in cinema, also known as the seventh art. Taking that in consideration this article describes the development phases of a movie platform. Firstly, we collect, prepare and characterize the data that we choose to work with. We talk about the thematic of our data, how we prepared, cleaned and joined that data. An characterization of our information is done with several graphs, also is showed and explained the conceptual data model. The second phase is focused on Information Retrieval. The collections and documents are specified in Collections and Documents, describing our different approaches to solve our problem of joining movies and their respective reviews. In the following sections, the field types were applied to the respective relevant fields (Indexing Process) and, finally, to evaluate the system's performance

3. M1 - Data Preparation

In this first milestone we focus on data preparation and characterization. In this first phase we first choose the thematic and the data sets that we wanted to use. We search those data sets from the kaggle website. We focused on data sets with a lot of information and rich in text. After that we started to explore the data to understand more clearly what we had. After that we proceed to clean and join all the data sets that we have to end up with the final data set. All of this is shown more clearly in the following pipeline.

3.1. Pipeline

After a thorough analysis we designed our final data pipeline.

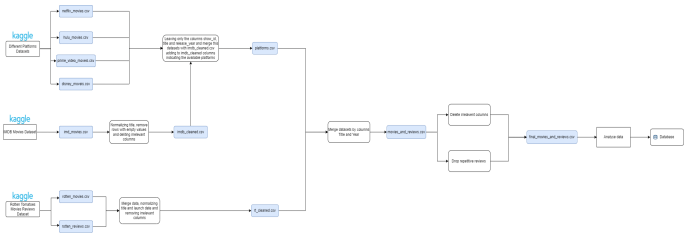

Figure 1. Pipeline

We decided that we should merge all different platforms' data sets leaving only the important columns, and afterwards merge it with a clean and normalized IMBD movies data set. The normalization we applied consists in removing spaces, putting all characters on lower case and removing strange characters (we found this to be extremly efficient at corresponding movies between different data sets with different origins).

This results in a data set with all known IMDB movies, each one with columns indicating if the given movie is available in each platform.

We also chose to merge the rotten-movies data set with the rotten-reviews data set, normalizing the movie title and launch date columns and removing other irrelevant columns. This results in a new "rt-cleaned.csv" that can be merged by the movies title and launch date with the "platforms.csv".

Subsequently we merged the two resulting data sets into one "movies-and-reviews.csv" by matching the normalized title and launch date columns.

Finally the resulting "movies-and-reviews.csv" is cleaned by deleting irrelevant columns to our project and dropping some repetitive reviews that we found after reviewing the data. After all this transformations we came up with the final data set "final-movies-and-reviews" that

can now be analysed and turned into a database for our next milestone.

3.2. Data Characterization

In this section we characterize our final dataset that we obtain during the preparation phase.

All the data sets we extracted from kaggle are decently sized, but the IMDB movies and reviews data set is the biggest one, containing all IMDB movies with over 100 reviews until 2020. This results on a final data set with over 270 000 lines.

Here we can understand that the average vote is a 6 on a scale form 0 to 10, showing that IMDB's rate is quite harsh being more common a movie with a rating of 2 than a movie with a rating of 9. After analysing the number of IMDB reviews we thought it would be interesting to compare the ratings given by the IMDB versus the ratings given by Rotten Tomatoes.

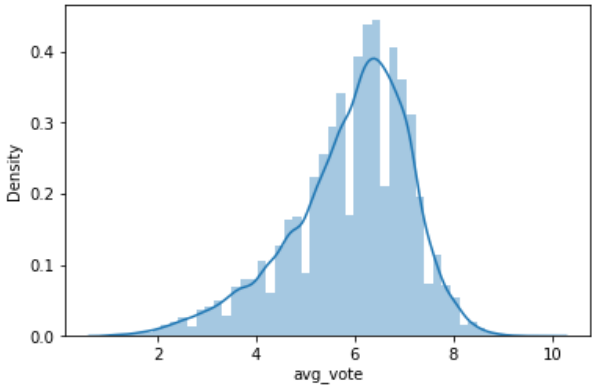

Figure 2. IMDB Rating

As we can see, the Rotten Tomatoes rating are much more permissive than the IMDB ratings, being 85 percent the most given rating. It's also very apparent that this ratings are much more distributed.

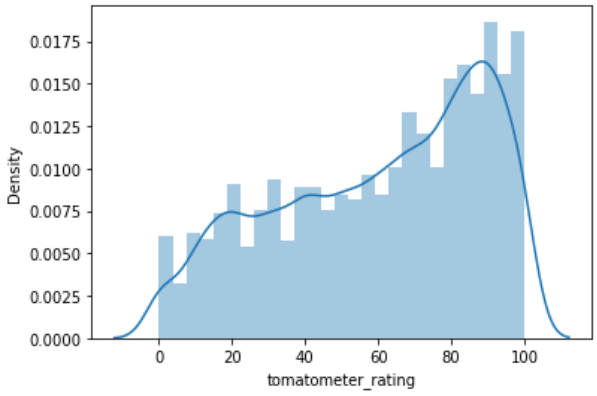

Figure 3. Rating

This graph demonstrates the exponential growth of movie making. The major part of the movies present on our database are from 2000 and afterwards so there's no need to remove old and irrelevant movies that might have been useless.

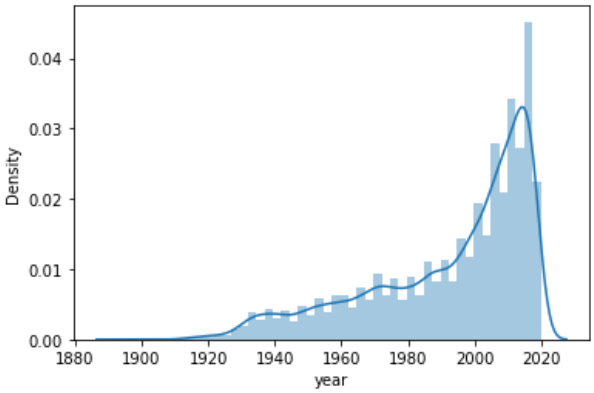

Figure 4. Density

Drama is, by a large margin, the most present movie type on our data set, followed by action and then comedy that is pretty much tied up with action.

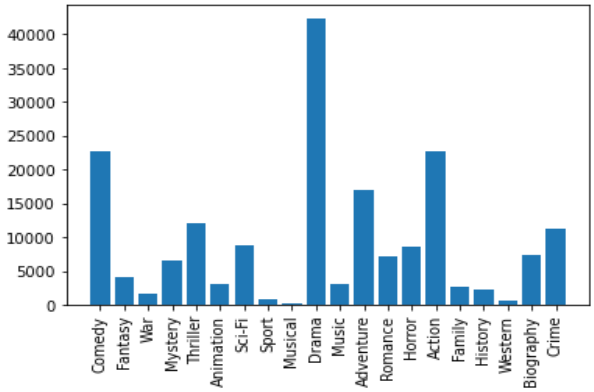

Figure 5. Genres

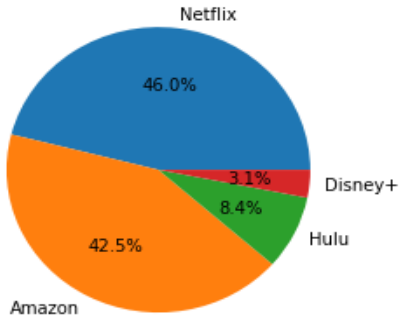

Figure 6. Platforms

Netflix and Amazon Prime Video have the larger share of movies present on our data set. Nonetheless, Disney

Plus and Hulu have a share big enough that shouldn't be discarded or ignored.

Here we can find the most common word usages on the reviews from IMDB and Rotten Tomatoes, respectively, given to the movies present in our data set.

Figure 7. Reviews

Figure 8. Words

3.3. Search Tasks

The data collected and prepared can be obtained by the information retrieval system in several ways. Some of the possible retrieval tasks are listed below:

- Search movies about specific historical events
- Search movies based on real events
- Search films that strategy positive matters
- Search films that different users determined emotive

3.4. Conceptual Data Model

After we prepare the data that we collected, we design the following conceptual data model.

As a main class we have the Movie class with the following attributes:

- **original_title**: This is the title of the movie.
- **year**: This is the year the movie was release in the theaters.
- **description**: This is the imdb movie's description.
- **avg_vote**: This is the average classification
- **votes**: This is the number of votes a movie had in the imdb.
- **movie_info** This is information about the movie.
- **available_hulu**: This tells if the movie is current available on hulu streaming service.
- **available_Netflix**: This tells if the movie is current available on Netflix streaming service.
- **available_amazon**: This tells if the movie is current available on amazon streaming service.
- **available_Disney**: This tells if the movie is current available on Disney streaming service.

Then each movie has a Review text that corresponds to all the reviews that people made relative to that movie. This class has the following attributes.

- **description**: This attribute corresponds to what was written about that movie.

Also, each movie has a review that corresponds to its classification in imdb and rotten tomatoes websites. This class has:

- **tomatometer_count**: This attribute is the number of classifications that was attributed to the movie.
- **tomatometer_rating**: This is the average of the classifications given by experts to the movie.
- **audience_rating**: This is the average of the classifications given by the audience to the movie.

Finally, the movie also has genres, actors and directors. This three classes have the same attribute:

- **name**: This attribute can be the name of the genre, actor or director.

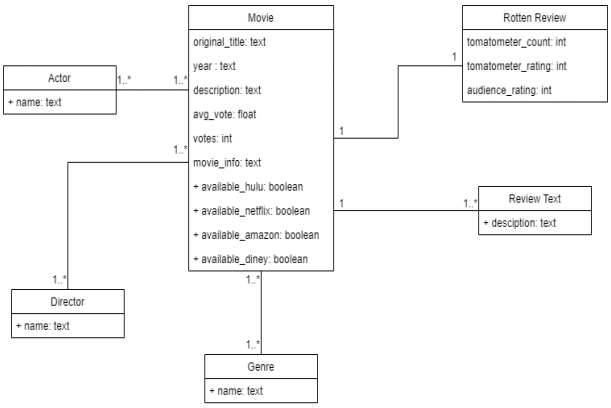

Figure 9. Conceptual Data Model

4. Indexation Process and Schema

To index our data set we use the recommended tool, solr. We use docker to run the application. We have a dockerfile where we save the files, that we need and a script that runs the commands to index our data set. We also added a schema to our system to improve the performance. The schema was defined using Solr's schema API. The schema is saves in a json file that was published

using Solr's post tool. Our schema has two costume types described in the following table:

Name	Tokenizers	Filters
text	StandardTokenizerFactory	ASCIIFoldingFilterFactory LowerCaseFilterFactory EnglishMinimalStemFilterFactory
commaText	PatternTokenizerFactory	ASCIIFoldingFilterFactory LowerCaseFilterFactory EnglishMinimalStemFilterFactory

Tab1: Table with two customized fields.

Field	Type	Indexed
title	StandardTokenizerFactory	YES
original_title	text	YES
year	solr.IntPointField	YES
genre	commaText	YES
director	commaText	YES
actors	text	YES
description	text	YES
avg_vote	solr.FloatPointField	YES
votes	solr.IntPointField	YES
available_hulu	solr.BoolField	YES
available_amazon	solr.BoolField	YES
available_disney	solr.BoolField	YES
available_netflix	solr.BoolField	YES
movie_info	text	YES
tomatometer_rating	solr.FloatPointField	YES
tomatometer_count	solr.FloatPointField	YES
audience_rating	solr.FloatPointField	YES
reviews	text	YES

Tab2: Table with every type used in the schema.

4.1. Information needs and Evaluation

4.1.1. Information Need 1 - A person likes movies about spies and wants to see a movie.. The objective of this information need is for people to be able to search for movies related to specific topics like spies or even places or dinosaurs for example. In order to do this, the topic word (spy) and its transformations are searched for on the movies description and movie rating parameters.

	Regular	Boosted
query	spy	spy^10 movie_title^2 original_title^2
Parameters	**q.op**: OR **fl**: movie_title original_title year genre director actors description tomatometer_rating reviews	**q.op**: OR **fl**: movie_title original_title year genre director actors description tomatometer_rating reviews

Tab3: Table with all the parameters used in the first query.

Tab4: Regular		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@2)	1.0
	3	Recall	0.625

Tab5: Boosted		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@10)	1.0
	3	Recall	0.9375

4.1.2. Information Need 2 - The favorite actors of a person is Brad Pitt and Angelina Jolie, so she wants to see movies with those actors.. With this information need a user will be able to find movies where two actors are present. By querying the names of both actors, it will be searched on actors parameter.

	Regular	Boosted
query	**actors**:brad pitt angelina jolie	**actors**:brad pitt angelina jolie^10 original_title^2 movie_title^2
Parameters	**q.op**: OR **fl**: movie_title original_title year genre director actors description tomatometer_rating reviews	**q.op**: OR title original_title year genre director actors description tomatometer_rating reviews

Tab6: Table with all the parameters used in the second query.

Tab7: Regular		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@10)	1.0
	3	Recall	0.322581

Tab8: Boosted		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@10)	1.0
	3	Recall	0.548387

4.1.3. Information Need 3 - A person likes excitement and wants to see a thrilling and exiting movie, but the movies needs to be very good. For this information need, a user is able to find movies that are expected to transmit a certain feeling to the spectator, by searching for verbs like thrilling or exciting thru the movie reviews parameters.

	Regular	Boosted
query	**description**: thrilling exiting exhilarating	**description**: thrilling exhilarating^10 exiting
Parameters	**q.op**: OR **fl**: movie_title **fq**: tomatometer_rating: [90 TO *] year genre director actors description tomatometer_rating reviews	***

Tab9: Table with all the parameters used in the third query.

Tab10: Regular		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@10)	1.0
	3	Recall	0.060241

Tab11: Boosted		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 10 (P@10)	1.0
	3	Recall	0.060241

4.1.4. Information Need 4 - A person only has Netflix and want to see the best movie available in that streaming service.. With this information need a user is able to specify which streaming service they have and search for movies by ratings.

	Regular	Boosted
query	available_Netflix:true movie_title original_title	available_Netflix: true^10 movie_title^2 original_title^2
Parameters	**q.op**: OR **sort**: tomatometer_rating desc **fq**: tomatometer_rating: [100 TO *] **fl**: movie_title original_title year genre director actors description tomatometer_rating reviews	

Tab12: Table with all the parameters used in the fourth query.

Tab13: Regular		Regular	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 2 (P@2)	1.0
	3	Recall	1.0

Tab14: Boosted		0	1
	0	Metric	Value
	1	Average Precision	1.0
	2	Precision at 2 (P@2)	1.0
	3	Recall	1.0

References

[1] https://solr.apache.org/guide/solr/latest/indexing-guide/filters.html

[2] https://solr.apache.org/guide/8_1/tokenizers.html

[3] https://solr.apache.org/guide/8_1/field-types-included-with-solr.htmlfield-types-included-with-solr