Ethical Movie Recommendations

By Sarah Amiraslani, Gabriel Alon, Kevin Deloria

Motivation

Introducing recommendation systems to movie streaming services has mutually benefited businesses and users. Successful recommendation systems save users time searching for a movie while encouraging them to engage more on the platform, which is good for business.

However, when maximizing engagement is the goal, recommendation systems may gloss over the fact that **users may not want to see what they often engage with**. For example, a recommender may present a recovering addict triggering content due to previous or current engagement. These recommendations could harm this user's sobriety and make them feel self-shame for engaging in the material they wish to be no longer drawn to. In these cases, it is essential to have safeguards in place so that users can control what content they see, and these controls should also be as transparent and easy to use.

Objective

We aim to explore how recommendations can be more sensitive to users preferences and to demonstrate a need for sensitive recommenders.

Deliverables

We have produced an **Altair-based user interface** that provides movie recommendations to users. Upon use, users are prompted to input what kind of film they want to see and, importantly, what they would like not to see. The latter can be content they find triggering or inappropriate.

For example, in the visualization section of this report, we show a recommendation in which we verified that ‘smoking’ is not a tag for that film. A user may wish to avoid renewing their smoking habit by reducing the chance of stimulating their craving for it. The recommendation algorithm will be based on popularity such that after filtering for what users wish to see and what they do not want to see, we will return the top movies with the highest mean review.

This is project inspired by a post by Alon Halevy about personal data timelines and a Medium post on an application of the MovieLens dataset for building recommenders.

Data Sources

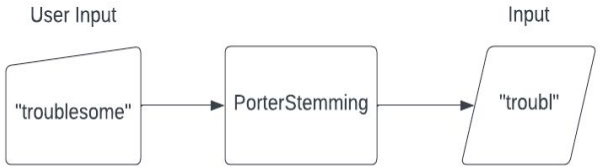
Name	Location	Format	# of Records	Time Covered	Important Variables
MovieLens	Link	CSV	25 million ratings and one million tag applications applied to 62,000 movies by 162,000 users. See movies.csv, tags.csv and ratings.csv	Movies published between 1874 and 2019. Ratings and tags from 1995 to 2019.	tags, movieId, genres, rating, title, year
IMDB Reviews	Link	JSON	5,571,499 unique reviews where each record is unique to a reviewer_id.	Reviews published between 2020 and 2021	rating, title, review_detail, review_summary
Posters	Link	CSV	41,979 unique poster urls for a movie poster image that is identifiable by title and year	Movies published between 2006 and 2021	title, poster, year

Data Manipulation Methods

A majority of our data manipulation involved cleaning up text data and also aggregating user provided reviews and tags to the movie level for comparison of movies.

Manipulating Tags & User Input

All user input went through a series of string manipulations to ensure that inconsistencies in string formatting would not differentiate similar reviews. The Porter Stemming algorithm was also applied to be able to compare the roots of words in review and tags.



For example, the Porter stemmer would transform the word "running" into "run", the word "cats" into "cat", and the word "troublesome" into "troubl".

Applying Porter Stemming helps us query and filter the dataframe by normalizing the tags for matching & filtering.

Joining the Datasets

Processed ML Tags/Movies

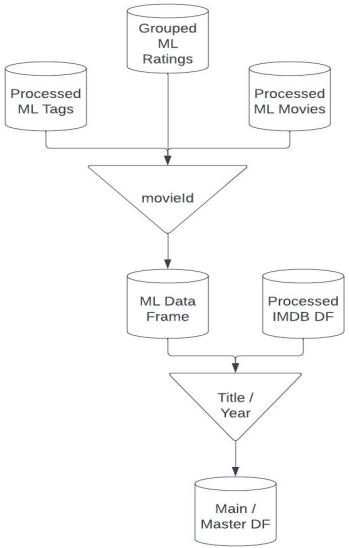
MovieLens' tags and movies pre-processed using the stem_words and prep_string function.

Grouped ML Ratings

This dataframe is the result of the raw MovieLens ratings, a list of ratings provided by anonymous users, grouped by movieId.

Processed IMDB DF

This dataframe is the result of splitting IMDB's movie column into two: title and year.

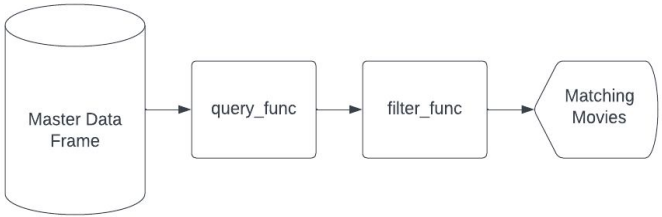


Handling Missing Data

Due to performing inner joins between the MovieLens and IMDB datasets, movies missing from either dataset were dropped. This was done to keep the dataset of manageable size for the goals of this project.

Data Manipulation Methods

As part of our deliverable - **we have created a UI that queries and filters the dataset based on user input.**



The query_func creates a boolean mask of all rows that contain the user's query keyword and returns the resulting masked data frame.

The filter_func creates a boolean mask of all rows that contain the user's filter keyword and returns the resulting masked data frame.

The Challenge of Loading Large Data

Dask is a Python library that provides a parallel computing framework for large datasets that are too big to fit into memory. When working with large datasets such as the IMDb dataset, Dask can help solve memory issues by breaking up the dataset into smaller chunks or "partitions", and then performing operations on each partition in parallel.

In our original Google Colab notebook - we mitigate this problem by using dask to load all the IMDB datasets. The benefit of Dask is that **it's a drop-in replacement over the standard Pandas DataFrame object**. However, we still encountered memory issues down the line when running transformations and aggregations against the full dataset, not to mention each iteration took up to 20 minutes to finish.

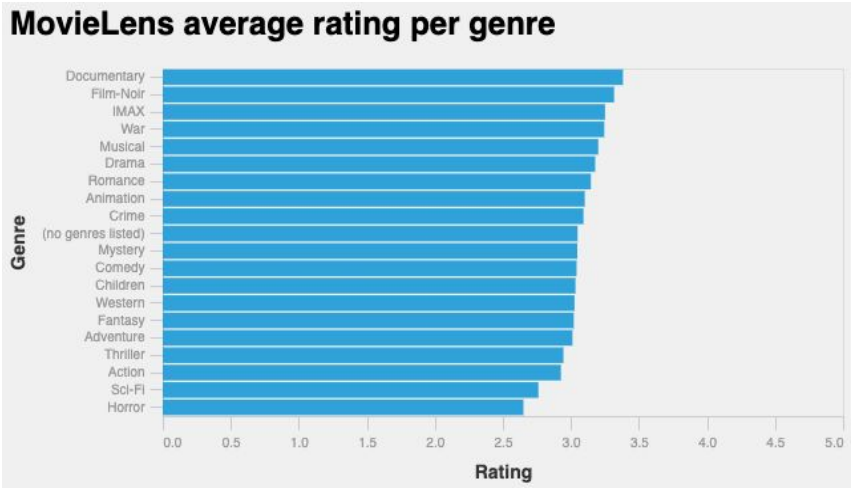
Analysis

Exploring the MovieLens Dataset

When exploring the Movielens we checked for distributions of ratings across categorical variables of interest (e.g., genres, movies, and tags) and explored outliers among continuous variables, such as rating. We decided to keep most outliers as long as the data seemed accurate.

MovieLens Average Rating per Genre

This bar chart is useful for quickly identifying which movie genres tend to be rated more highly by users in the Movielens dataset, and provides insights into which genres may be more popular or highly regarded by a large sample of moviegoers. From this bar chart we can see that there are no genres that stand out as having higher ratings than the others, so we can feel confident that our recommendations will not be biased to any particular genre.



Analysis

Word Cloud Insights

Using word clouds to show the tags in the MovieLens dataset can provide several insights into the characteristics and preferences of movies and their users. A word cloud is a visual representation of the most commonly used words in a text, with larger and bolder words indicating higher frequency.

One of the insights we gain from analyzing the tags is user sentiment & emotion. Base on the word cloud below, we can see examples such as `creepi`, `kidnap`, `violent`, and `alcohol` which may be triggering for some users.

Note that this was run against the full dataset - the sampled dataset will have other keywords such as `drug` , `nuditi`.

Popular Tags

Analysis

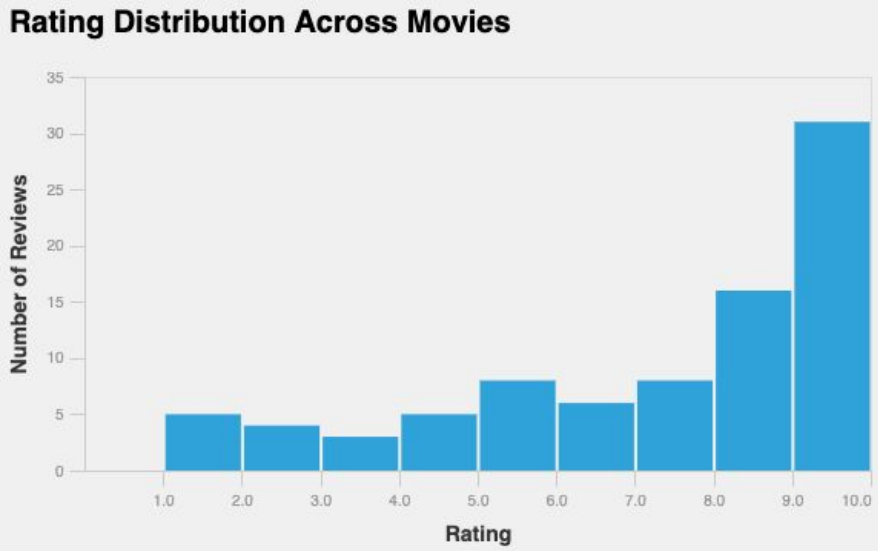
Exploring the IMDB Movie Ratings Dataset

The IMDB rating dataset is a collection of movie ratings from the popular website IMDB, which provides users with an opportunity to rate movies on a scale of 1-10.

Understanding the distribution of the IMDB Movie ratings

This histogram is useful in displaying the distribution of movie ratings, which can provide insight into how users tend to rate movies and what patterns or trends may exist in the data.

From this histogram we can glean that a majority of the IMDB reviews are highly positive, which could prove challenging when recommending movies based on ratings. If there are many highly rated movies, choosing the top n to recommend to a user would require the use of other features.



Visualizations

Our user interface has a search field for entering what **keywords** describe what we **don't** want to see in a movie recommendation, and another search field for what we **do** want to see. The ability to filter out what you don't want to see is the **ethical contribution.**

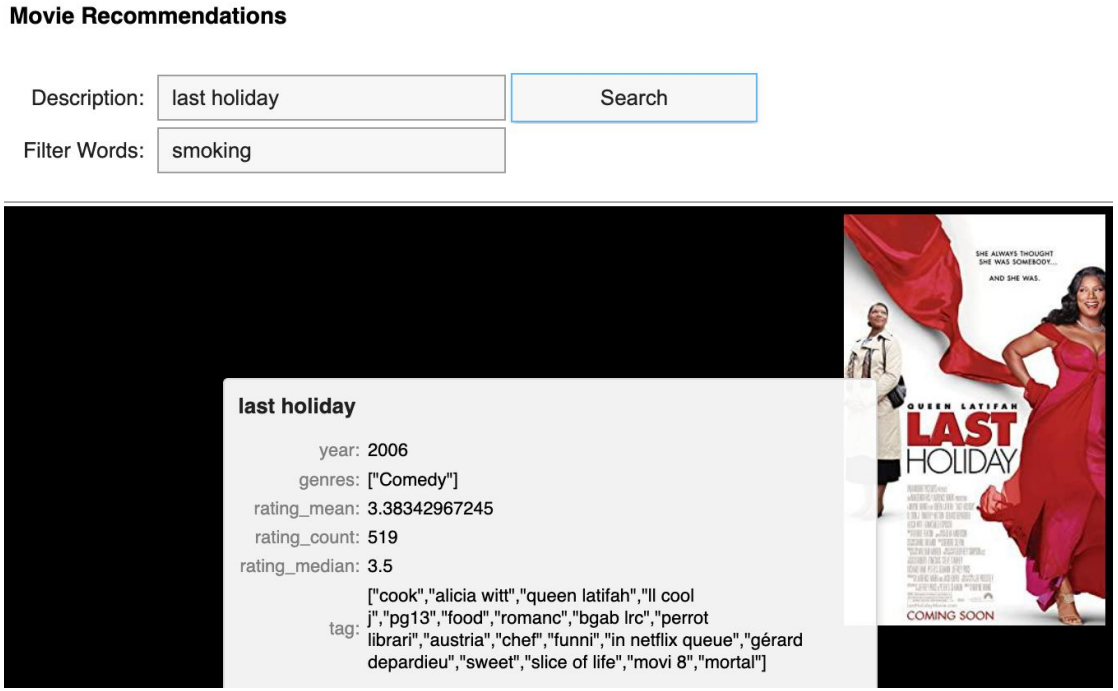
The **tooltip** includes information about the movie including its title, year, genres, mean and median rating, number of ratings, and human-labeled description tags. These pieces of information are similar to what a user might find on a movie if they used an existing movie search platform like Netflix. The tooltip allows one to confirm that the filter worked.

This uses a dataset of visual poster images to increase the effectiveness and expressiveness of the recommendations. The **effectiveness** should increase because movie posters are popular and easy to consume. Furthermore images offer a new way of thinking about a recommendation so it is a new dimension of **expressiveness**. The dataset is not complete, so if we don't find the posters we instead return a short dataframe of recommendations with the same data that would otherwise appear as a visual with a tooltip.

Visualizations

To the **right** is a filtered recommendation with a **Poster**.

The tooltip gives rating and description info about the film



Below is an example of the Altair output when posters are not found:

No Posters Found for the Following Film Results

	title	genres	year	tag	rating_mean	rating_count	rating_median
30876	ocean of pearls	[Drama]	2008	surgeon assimil racism indian lead sikh	4.8	2.0	4.8
39775	red dot on the ocean	[Adventure, Documentary]	2014	woman director	4.5	1.0	4.5
49559	blue planet ii	[Documentary]	2017	never seen ocean fish wild life sceneri natur ...	4.3	659.0	4.5

Endnotes

Statement of Work

- Sarah: Data cleaning, and writing about the ethical implications of the project
- Gabriel: First project description and references, organizing documents, assessing prerequisite coursework, visualization section code and writing
- Kevin: Exploratory data analysis section, as well as parts of the data cleaning, UI, and data importation

References

- Halevy, A. (2022, December 28). Your Personal Data Timeline: Data timelines will protect your privacy and make AI better. DeepLearning.AI The Batch. January 8, 2023,
- Alfaddagh, Maysam. "What Are the Top Rated 25 Movies?" *Medium*, Medium, 3 Jan. 2020,

Statement on Collaboration

To collaborate with one another we assigned tasks based on interest. We had frequent meetings to check in on progress and spent the time between meetings connecting asynchronously through slack.

Regarding collaboration on code, we began our collaboration on Google Colab, but as more our coding advanced we needed version control so we used Git to merge contributions.

Movie Recommendations GitHub Repository