Final Project Proposal

Scraping and Ranking RottenTomatoes

UIUC: Fall 2021 : CS 410 - Text Information Systems

Topic: Scraping and Ranking RottenTomatoes

Theme: Intelligent Browsing

Team Members: Jeremy Wisuthseriwong (jrw7), Munesh Bandaru (muneshb2), Supriya Puri (puri6)

Team Captain: Munesh Bandaru (muneshb2)

Table of contents

Abstract

Vertical search engines have become increasingly popular, now-a-days, as they sift through limited databases for information. A general web search cannot accommodate all of the users' searches when it comes to specific topics without implicit assumptions. In particular, using a vertical search, a user can extensively use query based searches to get the desired results with high user ratings and reviews. One major example for searching a specific topic is **"Rotten Tomatoes"** - a review aggregation website for movies and television series. Its content is specialised for users browsing information on top rated movies and television entertainment - genre, cast, network or the critic and user ratings. Results from services like Rotten Tomatoes allow a user to rank results by User Reviews, Critic Reviews, Genres, Audience Score. Being able to track user experience for various movies and tv series could lead to a larger audience and greater profits .

Goal

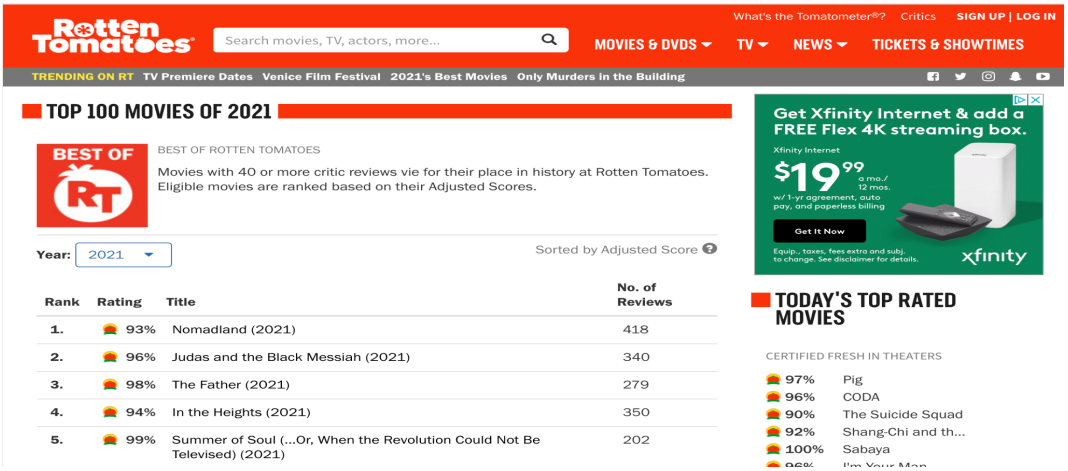
Our proposal is to scrape through the "Top 100 movies of 2021" on the Rotten Tomatoes website - https://www.rottentomatoes.com/top/bestofrt/?year=2021 and then rank and sort those 100 movies according to the user given query.

Problem statement

On the Rotten Tomatoes web page, users can find various pre-defines ratings and rankings for movies and TV series like Best Movies of 2021, Popular Shows on Netflix but there isn't a way to effectively search and rank for a list of movies within that particular ranking matching the user query/interest.

Example:

Lets say a user is browsing the "Top 100 movies of 2021" on Rotten Tomatoes.



But he is more interested in looking for thriller movies available on Hulu from this ranking list of 100 movies. When looking further, he could not find a way to filter and sort this list and needed to go through each movie description to find out whether a movie meets his interests or not.

Sample query: "thriller movies on Hulu"

Topic justification

As mentioned in the problem statement there isn't a way to effectively search and rank a list of the top 100 movies for 2021 based on a given user query.

Finding the right things on the internet is not easy - returning things that are of relevance to the user along with filtering the content based on the popularity is a challenge. Any general search engine would parse all the pages related to the query and search in a breadth-first manner to collect results. A query-specific search more efficiently searches a small subset of content by focusing on a particular requirement.

Through this project, we are trying to improve the user experience of browsing the content based on the user's interests. Here we are making our system to provide an intelligent way of browsing the content within the top 100 movies filtered and sorted based on the user query.

Dataset

https://www.rottentomatoes.com/top/bestofrt/?year=2021

"Movies with 40 or more critic reviews vie for their place in history at Rotten Tomatoes. Eligible movies are ranked based on their Adjusted Scores."

Movies' data is stored on several popular websites, but when it comes to critic reviews there is no better place than Rotten Tomatoes. In the movies dataset each record represents a movie available on Rotten Tomatoes, with the URL used for the scraping, movie title, etc.

We will be scraping our data from the url above and the dataset (CSV file) will include the below mentioned variables:

1. URL used for the scraping
2. movie title
3. Description
4. Genres
5. Duration
6. Cast
7. Director
8. Users' ratings
9. Critics' ratings
10. Network

Algorithm

We are considering the BM25 algorithm for this use case but we would like to experiment with other alternatives like jelinek-mercer , dirichlet-prior.

Programming language

We are considering **Python** as our primary programming language to design our algorithm and using relevant libraries to assist with scraping web pages. We may assess other web-based languages in the event we need to implement additional functionality.

We chose Python it has rich libraries for Scraping, Text processing and Modeling tasks

Evaluation

During our evaluation phase, we are considering using relevance feedback of a binary label (0=not relevant, 1=relevant) in order to reliably evaluate the ranked output. Moreover, we're planning to assess the effectiveness of the ranking algorithm by using mean average precision at 10 documents, since MAP is the standard measure for comparing ranking algorithms. We're aiming to use 10 documents as an evaluation threshold with the assumption that users will not likely scan through all 100 documents. Lastly, we'll consider statistical significance testing to evaluate the average precision results, as we experiment with our ranking algorithm to ensure differences aren't simply due to particular queries that are chosen.

Workload justification

Main Tasks	**Estimated Time Cost (Hrs.)**
Scraping the Main top 100 movies page	20
Scraping the description/content of each movie page	10
Refining the dataset	10
Modeling	10
Evaluation	10
Building an interface for user query interaction	20
End to end testing	10
Drafting Presentation and Project report	10
Miscellaneous Learning	10