CS839 - Project Stage 2 Report
Extraction of Movies from Structured Web Sources

Team Members:

- Sri Harshal Parimi (sparimi@wisc.edu)
- Shebin Roy Yesudhas (royyesudhas@wisc.edu)
- Sankarshan Umesh Bhat (sbhat6@wisc.edu)

I. Web Data Sources:

a) IMDB: The first data source that we selected was (www.imdb.com). IMDb is an online database of information related to films, television programs, home videos and video games, and internet streams, including cast, production crew and personnel biographies, plot summaries, trivia, and fan reviews and ratings.

b) Rotten Tomatoes: The second data source was chosen to be Rotten Tomatoes (www.rottentomatoes.com/). Rotten Tomatoes, similar to IMDB is a collection of movies and TV shows information that includes various details of the movies such as release, genre, director, grossings, fan reviews, ratings, etc.,. The movies are listed according to the year of release and top listings by various genres.

II. Extraction of Structured data

In order to extract information from the structured HTML pages, we did some analysis with respect to the structure and format of the HTML from both the web sources. Finally, we decided to extract Name of the movie, Rating, Certificate, Grossing, Director, Genre, Release time and Run Time as these entities were common between the two sources. We used Scrapy - a open-source crawling framework, to implement the crawling. XPath and other selectors which are part of scrapy were used to extract relevant details.

IMDB:

Codebase for IMDB can be found at:
https://github.com/harshal95/CS839/blob/master/stage_2/code/cs839/cs839/spiders/Imdb_spider.py

Steps performed to extract tuples are as follows:

For 100 years between 2018 and 1918:

1. We used search filters in https://www.imdb.com/search/title page to extract the top 30 movies of the given year ordered by the number of votes. Ex- URL query (https://www.imdb.com/search/title?title_type=feature&release_date=2017-01-01,2017-12-31&sort=num_votes,desc&count=30)
2. For each of the individual movie link's page(Ex- https://www.imdb.com/title/tt3315342/?ref_=adv_li_tt):
   a. Extracted relevant info based on schema attributes like Name, ReleaseDate, Certificate, Rating, Runtime, Director, Genre, Grossing.
   b. Stored the extracted tuple in csv format.

Rotten Tomatoes:
Codebase for Rotten Tomatoes Crawler can be found at:
https://github.com/harshal95/CS839/blob/master/stage_2/code/cs839/cs839/spiders/RottonTom_spider.py

Steps we performed to extract the entities:

1. Similar to extraction from IMDB, we constructed URL of Rotten Tomatoes to list top movies with respect to year of release Ex- URL : https://www.rottentomatoes.com/top/bestofrt/?year=2018.
2. From the list of movies, extracted the URL corresponding to each movie and used it to request HTML with respect to each movie. URLs were of a particular format. Ex - URL: https://www.rottentomatoes.com/m/mission_impossible_fallout
3. For each of the individual movies link, as discussed in IMDB extraction various attributes were extracted.
4. One critical difference in the extraction is, some of info listed for a movie does not follow a specific order inside a HTML div. Thus, we had to iterate over all the listed elements, and matched it with the headings before extracting the required values.

III. Entity of choice:

The entity we choose to extract was movie information of different genres from the IMDB website and Rotten Tomatoes website.

Table A: IMDB(imdb.csv): The table from IMDB contains movies information (Name, ReleaseDate, Certificate, Rating, Runtime, Director, Genre, Grossing).

Table B: Rotten Tomatoes (rotton.csv): The table from Rotten Tomatoes contains movies information (Name, ReleaseDate, Certificate, Rating, Runtime, Director, Genre, Grossing).

Tuples:
IMDB: 3000
Rotten Tomatoes: 3072

Schema:

- ❏ Name: The name of the movie.
- ❏ ReleaseDate: Data of release of the movie.
- ❏ Certificate: Content rating provided for the movie
- ❏ Rating: The average rating based on fan reviews for the movie
- ❏ RunTime: Running length in minutes of the movie
- ❏ Director: The director(s) of the movie
- ❏ Genre: Conventional category which identifies the movie

IV. Open Source Tools:

Open Source crawling framework named Scrapy which is based on Python, is used to extract the relevant details from the respective web sources.

Scrapy framework is based on *Spiders*, which are self-contained crawlers that can be instructed to crawl a particular website. The framework does asynchronous scheduling and processing of web requests, which makes it faster to process and extract contents from the web pages. It has a rich set of APIs such as CSS selectors and XPath based selectors to parse through the HTML tree and get the elements or the text corresponding to the elements. Scrapy also comes with an interactive shell which the developers could use to explore the page structure before confirming to a particular selector. Scrapy has a wide range of application ranging from information extraction to data mining.