CS838 - Project Stage-2

Crawling and Extracting Structured Data from Web Pages

Manjunath Nagaraj Shettar (shettar@wisc.edu)
Jayashankar Tekkedatha (tekkedatha@wisc.edu)
Samhith Venkatesh (svenkatesh5@wisc.edu)

Dataset Source:

1. www.imdb.com
2. www.metacritic.com

Data Scraped:

Movie data for the ranking 'best movies of all time' for the above-mentioned websites.

Schema Generated:

1. IMDB Schema: The production house name was not captured from imdb.com, but to match the schema with metacritic.com the column was added. Default value: NA

   [name *(text)*, release_year *(int)*, certificate *(text)*, runtime *(int)*, genre *(text)*, director *(text)*, gross *(long int)*, actors *(text)*, production_house *(text)*]
2. Metacritic Schema: The box-office collection gross value was not captured from metacritic.com, but to match the schema with imdb.com the column was added. Default value: NA

   [name *(text)*, release_year *(int)*, certificate *(text)*, runtime *(int)*, genre *(text)*, director *(text)*, gross *(text)*, actors *(text)*, production_house *(text)*]

IMDB Data Scrapping:

1. Identified the URL queried to obtain the required movie listing. '*https://www.imdb.com/search/title?title_type=feature&languages=en&sort=num_votes,desc&page=**1**&ref_=adv_nxt*'
2. Each webpage contains 50 movies listed with details about each movie. Iterated over 60 pages by editing the above query's 'page=1' value.
3. Scrapped **3050** movies from www.imdb.com

Metacritic Data Scrapping:

1. Identified the URL queried to obtain the required movie listing. '*http://www.metacritic.com/browse/movies/score/metascore/all/filtered?page=**1***'
2. Each webpage contains 100 movies listed with details about each movie. Iterated over 30 pages by editing the above query's 'page=1' value.
3. Scrapped **3000** movies from www.metacritic.com

Steps for Scrapping:

1. Using Request python module html source was downloaded.
2. Using Beautiful Soup python module html source was converted into soup objects to scrap individual tag data pertaining to each movie.
3. Example: to extract name from imdb web page:
    a. First the movie specific "div" tag was identified by its class name: ' lister-item mode-advanced'.
    b. Then movie name was extracted from the anchor tag "a" (residing in the headline tag "h3") with *h3.a.text*
4. Error handling was done for:
    a. Web page response status i.e. it should not be "404"
    b. Each individual movie data should be present else "None" was used to symbolize missing data.
5. Each movie attribute data was stored in lists.
6. Finally, Pandas python module was used to convert the scrapped data into csv.

Modules Used:

1. Requests Module: to download website's HTML source data
2. Beautiful Soup: To parse the HTML source data. Then scrap required HTML tags and generate the movie data.
3. Pandas: To convert movie data into CSV file.