Information Retrival
MOVIE INSPECTION

Introduction:

This project is about searching for a movie based on title caste or some other keywords. For a given query if it matches with any of the movie names that are present in the dataset, then it displays the rating for that move. If the query does not match with the move name then it ranks the documents based on the query and retrieves the top 10 documents.
Project Domain: Movies

Dataset:

- The dataset contains text documents that are scraped from movie websites like IMDB.com.
- The text files contain information about the title, genre, cast, ratings and other fields related to a particular movie.
- For some movies, some fields are unavailable (They are unavailable on the website so the file related to that movie will not contain that field).
- There are 48690 documents scraped from two websites.
- Below is the snippet of a sample text file from the dataset.

```
The Proposal
https://www.metacritic.com/movie/the-proposal/details
108 min
Drama Comedy Romance
6.7
197
Touchstone Pictures
Ryan Reynolds
Sandra Bullock
Aasif Mandvi
Alexis Garcia
Alicia Hunt
Betty White
Craig T. Nelson
Dale Place
Denis O'Hare
Kortney Adams
Malin Åkerman
Mary Steenburgen
Michael Mosley
Michael Nouri
Oscar Nuñez
When high-powered book editor Margaret faces deportation to her native Canada, the quick-thinking
exec declares that she's actually engaged to her unsuspecting put-upon assistant Andrew, who she's
tormented for years. He agrees to participate in the charade, but with a few conditions of his own.
The unlikely couple heads to Alaska to meet his quirky family and the always-in-control city girl
finds herself in one comedic fish-out-of-water situation after another. With an impromptu wedding in
the works and an immigration official on their tails, Margaret and Andrew reluctantly vow to stick
to the plan despite the precarious consequences. [Touchstone Pictures]
```

Ranking Metric:

- For a given query the documents are ranked based on the tf-idf score.
- tf-idf score is computed as follows: $(1 + log(tf_{t,d})) * log(\frac{N}{df_t})$
    - Where tf = Frequency of a given term in the document.
    - N = Number of documents in the collection.
    - df = Number of documents containing the term t.

- For each word in the query, the tf-idf score is computed for each document. Then the documents are ordered in the descending order of their tf-idf score.

PART - 1:

Search and retrieval of documents relevant to the given query.

Index Creation:

- Terms are extracted from the documents and are preprocessed.
- The Inverted index is implemented using a Trie data structure. The posting list will be at the end of each term.
- After processing a new term if the term is already present in the Trie then the document id is added to the posting list associated to that particular term without disturbing the sorted order. If the term is not present in the Trie then the term is added to the trie with the posting list containing its document id.
- Along with the document id the term frequency, document frequency are also stored in the index.
- This inverted index is written to a file in sorted order after processing all the documents in the dataset.
- For the subsequent queries, the index is loaded into the memory from the disc.
- Below is the snippet of the index file.

```
milka (4)(4) ==> {9580=1, 29828=1, 35245=1, 44834=1}
milkchan (1)(1) ==> {10733=1}
milker (1)(1) ==> {5922=1}
milkflake (1)(1) ==> {48170=1}
milkie (1)(1) ==> {7114=1}
milking (3)(3) ==> {27876=1, 36784=1, 45958=1}
milkis (1)(1) ==> {11297=1}
milkman (2)(2) ==> {18582=1, 24640=1}
milkmen (2)(1) ==> {4487=2}
milko (2)(2) ==> {18453=1, 35247=1}
milkovich (2)(1) ==> {46090=2}
milkowski (1)(1) ==> {15435=1}
milks (1)(1) ==> {28659=1}
```

- `milkshake (3)(3) ==> {2327=1, 6974=1, 33089=1}`

Query processing and document ranking:

- For a given query all the document id's containing the terms present in the query are taken.
- Documents are ranked based on the above ranking metric and top 10 document id's and the titles of the movies in those documents are displayed to the user.

PART - 2:

Displaying the ratings for a movie if the query matches the movie title:

- A separate index is created with the tokens as movie titles and the posting list contains the documents related to the particular movie.
- Suppose if there are two matching documents for the given query then the movie rating and the website name are extracted from the documents and displayed to the user.

Complexity:

- Creating inverted index - O(M * N)
    - M = Number of documents in the collection
    - N = Number of tokens in the document

- Processing query - O(log(N))
    - N = Number of characters in the query

Sample query and results:

Query: Inception
Results:
Relevant Documents:
50 1277 5008 6972 7358 16229 19656 25099 28353 30353
Movie Name: inception
Ratings:
IMDB: 8.8
Metacritic: 8.8

Inception
Inception: The Cobol Job
Interface
The Palmer Supremacy
Regrets of the Past
Interception
The Uncanny Film Festival and Camp Meeting
Raiders, Raptors and Rebels: Behind the Magic of ILM
Alien Inception

The Vanished

Query: Leonardo DiCaprio
Results:
Relevant Documents:
37755 42006 35560 36121 36352 36537 37643 38240 41531 42302

The Revenant
This Boy's Life
The Aviator
Django Unchained
Once Upon a Time ... in Hollywood
The Departed
The Wolf of Wall Street
What's Eating Gilbert Grape
Romeo + Juliet
Before the Flood

Query: Avengers
Results:
Relevant Documents:
644 37640 14 77 3749 13240 35530 39685 133 713
Movie Name: avengers
Ratings:
IMDB: 5.8

Marvel's Avengers
Captain America: Civil War
Avengers: Endgame
Avengers: Infinity War
Ultimate Avengers II
Marvel Future Avengers
Avengers: Endgame
Avengers: Infinity War
Avengers: Age of Ultron
Avengers Assemble

Chandaluri Eswara Surya
S20180010035