Information Retrieval Project

Team Members:

- 1. K.Sai Sri Thanya (S20180010083)
- 2. E.Sumasree (\$20180010052)

Project idea: IR System for News Articles

Tasks Performed:

- 1. Crawling
- 2. Scraping
- 3. Creating Inverted Index
- 4. Scoring and Ranking

Detailed View:

Crawling:

Libraries used:

- 1. urllib: To request urls and also parse them
- 2. From bs4 : BeautifulSoup : To read html files
- 3. Time
- 4. re : for comparing regular expressions
- 5. Sys

Algorithm or Code Flow:

Crawler function:

- Takes in an url and checks whether it is in this format "http://en.wikipedia.org/wiki.../"
- Also checks if that url is already visited or not. If not visited it is appended to a list which stores all urls.
- · Gives out other urls of above format
- First an url is given .
- Its html text is read and stored using Beautiful soup.
- Crawler function is called.

Every link is generated and stored

 Ω

 Depth increases and all the urls are stored in "file.txt"

Scraping:

Libraries used:

- 1. Requests
- 2. From bs4 : Beautiful Soup: To read html files

Algorithm or Code Flow:

File "file.txt" generated using crawler is opened



For every url in file.txt , Using requests we get the url Read the content of that url using beautifulsoup



Content is filtered , which is first level filtering by removing unknown symbols



Inside documents folder,

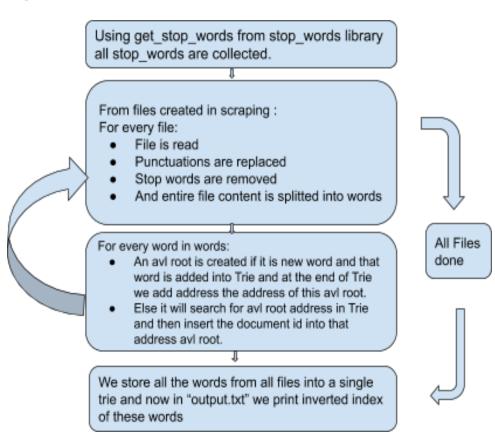
For every url the generated text is stored with file name: "%d.txt" where %d is 1,2,3,.....

Inverted Index:

Libraries used:

- **1.** Glob: The glob module finds all the pathnames matching a specified pattern according to the rules
- 2. String
- 3. From collections : defaultdict : Defaultdict is a sub-class of the dict class that returns a dictionary-like object.
- 4. Math: To use functions like square root, log etc.

Algorithm or code flow:



Scoring and Ranking: A query is taken as input We use Vector space model tf-idf for scoring Removing stop words from the given query Using already generated inverted index we calculate tf, idf for query terms and their weights and also for those terms we also calculate tf, weights , normalized weights in for each document Now we calculate the products of query weights and every documents normalized weights and then calculate score

In descending order of score we give ranking

Output:

```
sree@SUMASREE:~/Education/ir$ gedit index_rank.py
sree@SUMASREE:~/Education/ir$ python3 index_rank.py
enter query:
transport developement meaning made record
303
281
123
256
56
122
6
75
285
76
sree@SUMASREE:~/Education/ir$
```