CA Assignment 1

Group#2

Avinash Kumar Chaurasia,

Ambuj Kumar Mondal,

Shashi Prakash,

Saurabh Raut

9th May 2021

Assignment I: Data Acquisition

Crawling Strategy:

- Using the framework scrapy we have first crawled the URL: https://www.debate.org/opinions/?sort=popular which results into the list of popular opinions.
- We have created a function 'parse_tag' in which we get the URL of first five opinion cards on the opinions page.
- After inspecting the opinions page, all the opinion cards are put inside a list having id 'opinions-list'. And for every element in ''opinions-list', there is an anchor tag which consists of the URL redirecting to the page of that specific opinion. Thus, after getting the link to every specific opinion, we have concatenated it with the base url:

  https://www.debate.org/opinions/

  For example: tags.css('a ::attr(href)').get() results in :

  kirk-will-always-be-better-than-picard

  And the complete URL for this topic after concatenation will be:

  https://www.debate.org/opinions/ kirk-will-always-be-better-than-picard

- Thus, after getting the URL for each opinion, we crawled the specific web page of the opinion using parse_urls(self, response). This web page contains a navigation bar which has path to the current page specifying the category of the opinion. The value at the seconds position inside the breadcrumb tag represents the category of the opinion:

  ```
  category=resposne.css('#breadcrumb a::text')[2].get()
  ```

- A tag named 'debate-content' contains the heading and display image for the particular opinion. We have fetched the topic from the span tag inside the debate-content tag as follows:

  ```
  topic_page = resposne.css('.debate-content .r-contain span ::text').get()
  ```

- Each opinion has a list of yes arguments inside a div named 'yes-arguments'. We declared a variable Yes_tag_list and fetched the list of yes-arguments as follows:

  ```
  Yes_tag_list = resposne.css('#debate #yes-arguments li')
  ```

- And for every yes-argument in the list, it's header and paragraph text is fetched as follows:
  Header: header=tags.css('h2 ::text').get()
  Text: bod=tags.css('p ::text').getall()

- The same strategy is applied for the list of no arguments as stated in the last two points above. Only the difference is in the div id of the no-arguments list in HTML.
  No_tag_list = resposne.css('#debate #no-arguments li')

- After getting all the category, topic, argument heads and body. We have first arrange them into the format specified in sample_data.json and then yield the crawled data into a new Json file by using the command scrapy crawl debate_crawl -O data.json

- Now data.json contains all the crawled data for five most popular posts from debate.org.

Functions in myspider.py

- A function **start_requests(url)** is used for hitting the URL of the popular opinions page. It extracts the page containing all the popular opinions.

- We have declared a function: **parse_tag(self, response)**. It gets the crawled scrapy response of the popular opinions page as the parameter and yields the URL of each individual opinion page.

- **parse_urls(self, response**) gets the crawled response for the particular opinion page and it yields the category, topic, header and text of the particular opinion yes/no arguments.

Histograms and Bar plots strategy in Jupyter Notebook

- First we imported packages for Pandas, Matplotlib, and nltk. Also, we have used nltk.download('punkt') for using the nltk function word_tokenize(). Using this function, we have tokenize all the arguments for pros and cons.

- We loaded the data.json which contains the crawled data and then wrote logic for counting the length of arguments(as number of tokens), total number of Arguments for each topic and category and add the counts as columns in Dataframe.

- In order to create the histograms of the length of Arguments (as number of tokens), we have appended all the tokens, generated using word_tokenize, into one single list called "**New_list**" and then form the histograms using plt.hist(New_list,bins=16). Please find below screenshot of our histograms.

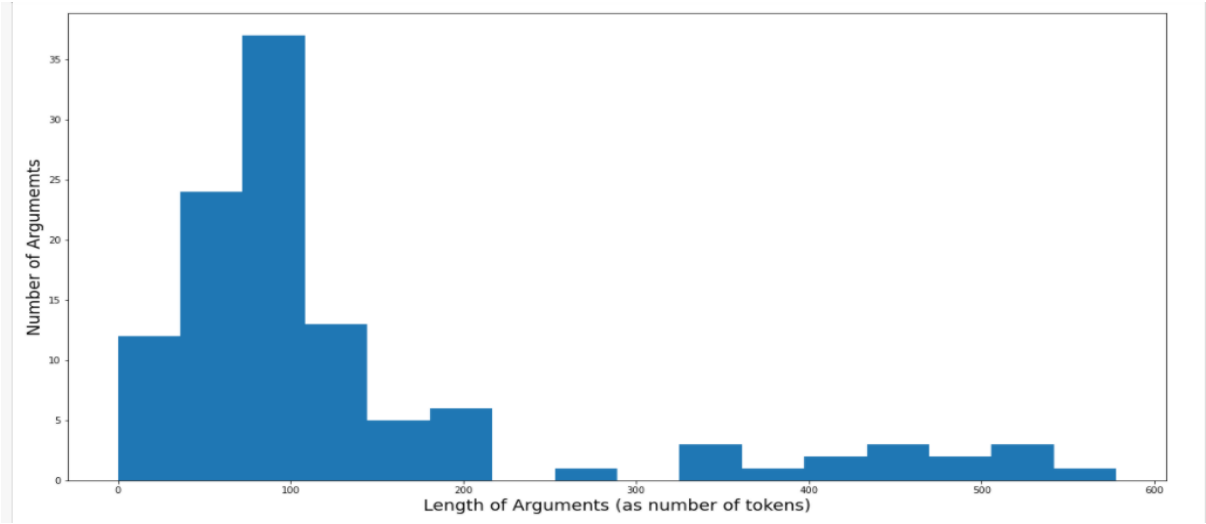

Fig 1 – Histogram for Length of Arguments

- Finally, we have created the bar plots for Length of arguments per topic and category. Please find the below screenshots.

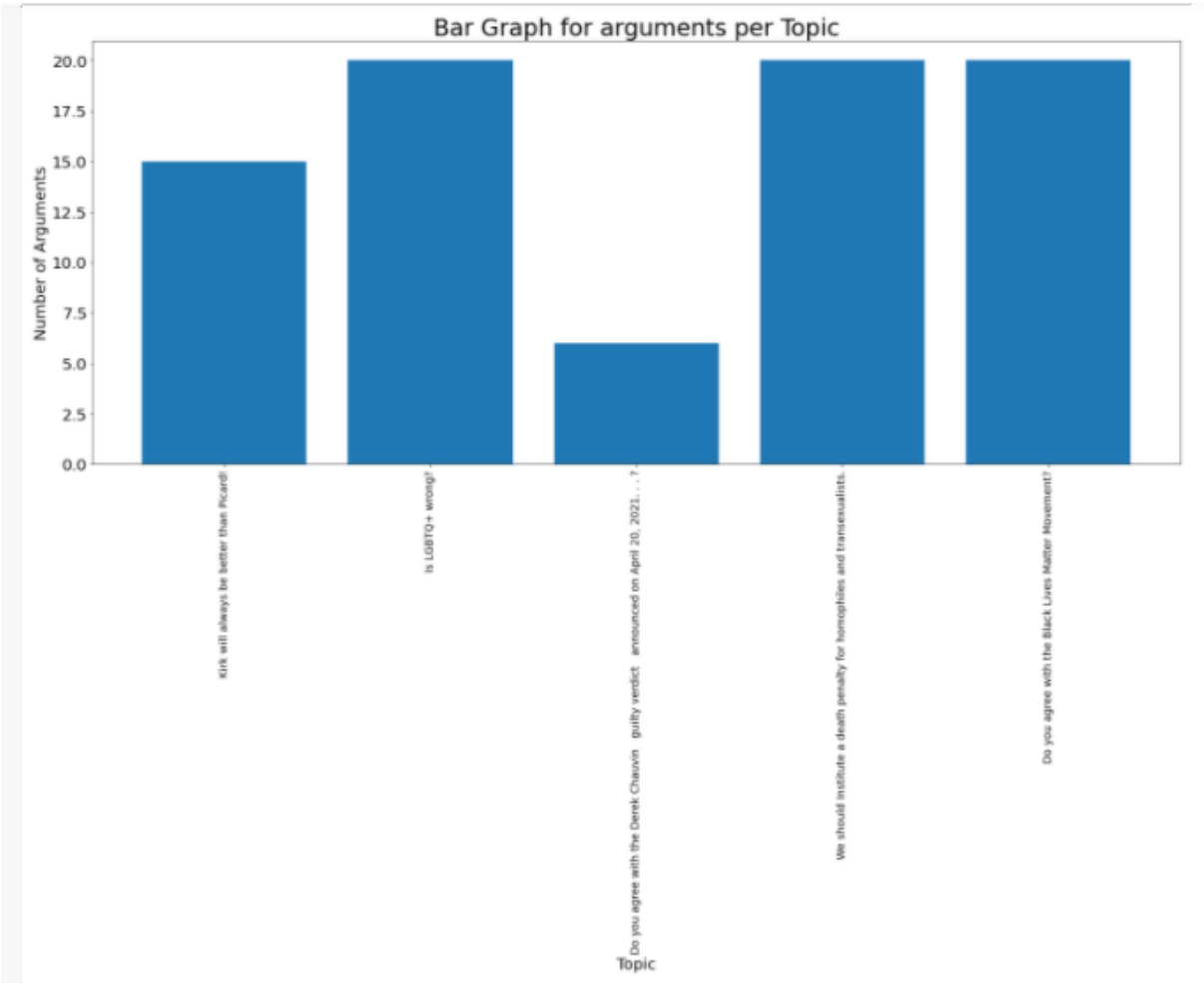

Fig 2.1 – Bar Graph for Number of Arguments per Topic

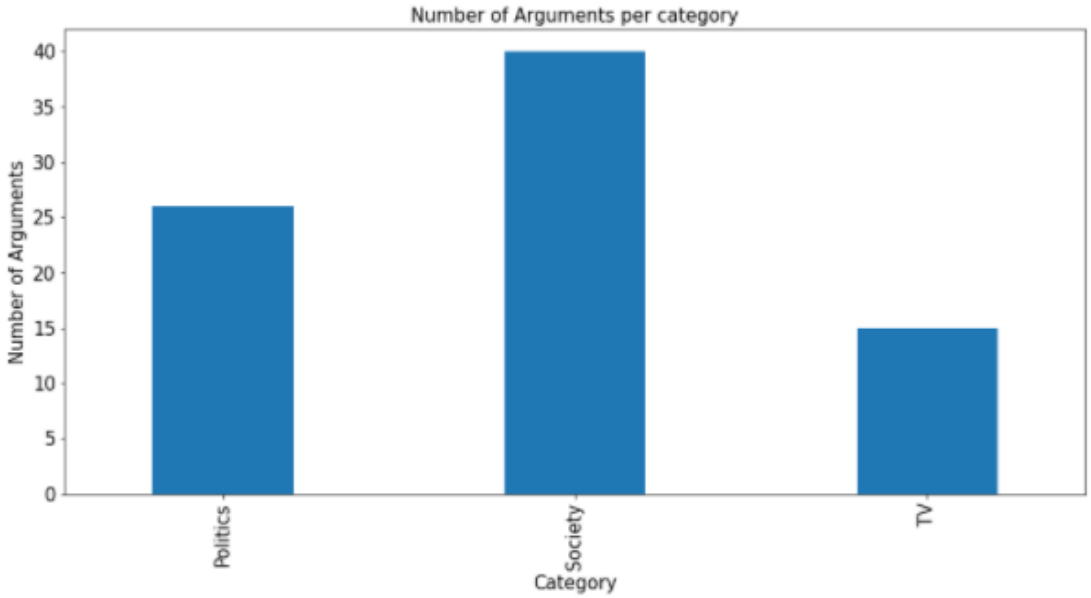

Fig 2.1 – Bar Graph for Number of Arguments per Topic