PROJECT Y

JULY 31

PATHFINDERS

PROBLEM STATEMENT

This project consists of two components -

Component 1 - Web Scraping module

You are required to scrape two weeks (Feb 1 , 2021 to Feb 14 , 2021) of web news files from The Hindu archive site : https://www.thehindu.com/archive/web/2021/02/ . As per the Robots.txt at Hindu website(https://www.thehindu.com/robots.txt) , the archives section is NOT in disallow , thus it implies it is within legal norms to scrape the archive section.

You will need to find URL patterns with which you can go to each day of the two weeks period of February month , find web links and then recursively hit all the web news pages that have occurred on that given day page. From each of the page , you need to scrape out the web content and create an independent JSON file that will capture all the text details as values and store them in a key named "text" within that file.

For each of the file to be identifiable independently , create a sequence pattern which will be used for naming the file. For e.g. :

thehindu_feb_02_file_23.json, thehindu_feb_02_file_24.json etc .

You will also need to store all these file names with their respective URLs from which the content is scraped in another *lookup file* , say URLs_to_file_mapping.csv . Content of this will look like -

file_scraped | Corresponding_URL

thehindu_feb_02_file_1.json | https://www.thehindu.com/sport/other-sports/radjabov-closer-to-beating-dubov/article33475129.ece

thehindu_feb_02_file_2.json | https://www.thehindu.com/sport/other-sports/nba-wall-makes-rockets-debut-in-win-over-kings/article33475123.ece

The intent of look up file is to eventually tag the Topics identified for each URL back to them via the *topics -> scraped file -> URLs linkage

The web scraper that you create might be required to run on your own system / or may work on colab . Depends on the time taken for response from these sites , file storage capacity on a google drive (tied to a colab) or eventually saving scraped files on your local systems. Not all web page link will work via scraping , so try to identify metrics around :

- **No of web page hits**

- **Success Hit Ratio**

Component 2 - Topic Model

Once you've scraped the two weeks' worth of data , the idea is to perform Topic Model on this scraped data set. You can keep 90% of the web scraped data in training of the model and 10% as a holdout set on which you can assign topics based on the trained model.

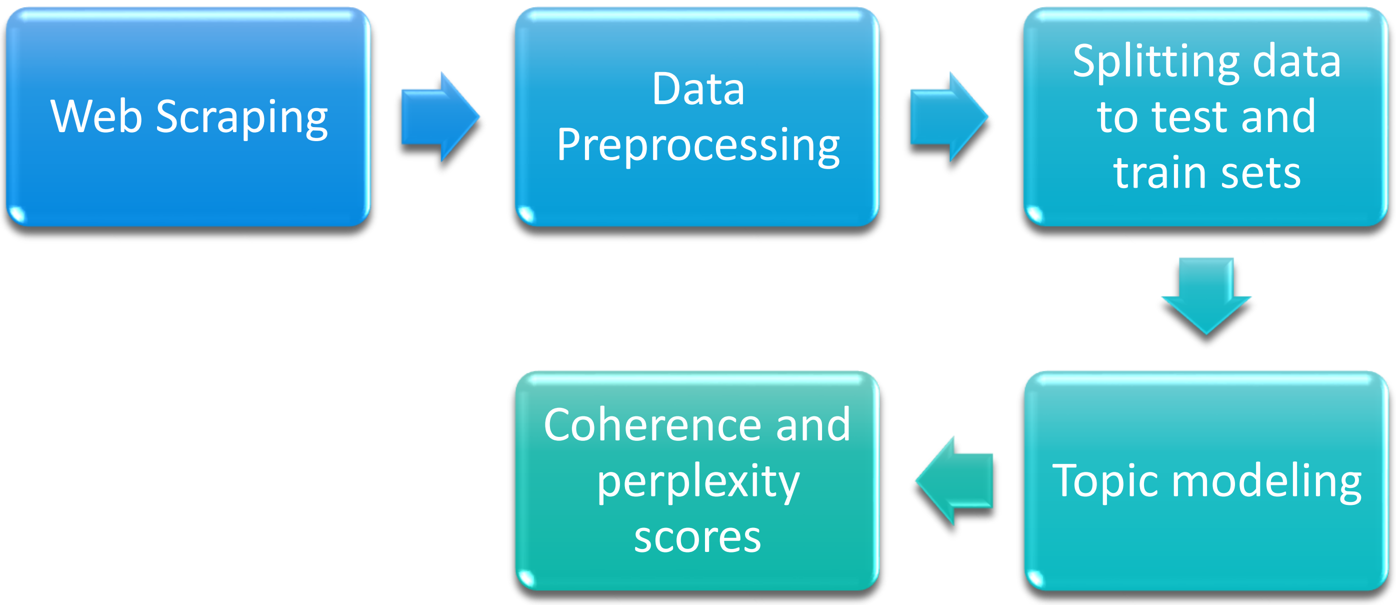
Find out the optimal topics number based on Topic Coherence score along with manual review of Topic - words quality. Iterate until you can get the diverse and best topics with words that closely define a theme/topic

For each of the Topic that have been identified - give the Topic Name based on IAB Taxonomy attached. IAB Taxonomy is a standard followed across web to provide web page content a consistent Topic naming convention. Highlighted in the attached XLS are two Tiers - I & II. Tier I is a broad category , whereas Tier II is subcategory under Tier I. First ensure you assign Tier I based Topics mapping based on the content and Topic words as they show up. If you feel two topics fall under the same Tier I category, then look at assigning further sub-category of Tier II.

Based on the above Topic names assignment , save the model, and then assign the topics to 10% test set. Try doing manual review for few sets of links i.e., whether the assigned topics make sense to what URL web page is about ? Come up with metrics around Accuracy or any other that you feel relevant to measure the quality of Topic Model

1. METHOD DESCRIPTION

Following flowchart describes the methodology adopted for tackling the problem statement. Each module in the flowchart will be further decomposed in subsequent sections of the report.


Web Scraping
Data Preprocessing
Splitting data to test and train sets
Topic modeling
Coherence and perplexity scores

2. MODULE DESCRIPTION

2.1. WEB SCRAPING

Before working on the scraper or writing code for scraper, we analysed the weblinks first which were to be scrapped. Following were some results:

- A total of 5565 links of Hindu news articles was required to be scrapped.

- Since each webpage contained some irrelevant data which were not required for scaping, the html tags which contained useful content was required to be analysed.
- HTML tags that contained useful content were:

  h1 : Title of the News article.

  h2 : Subtitle of the article.

  p : Body of the article.

Now since there was a total of 5565 links, it was difficult to scrape them in one go as we got blocked by the site after scraping almost 2000 files. We tried to use proxy IPs as well for scraping but it didn't work as a single IP took almost 25-30 min to establish connection and scrape a link.

For scraping we created two separate lists of weblinks for 7 days each and then scraped them separately.

2872 weblinks :	Took 1 hour and 45 minutes to scrape.
2683 weblinks :	Took 1 hour and 37 minutes to scrape.

For detecting the efficiency of our scraper, we did an outlier detection as to how many files contained number of words that were below the lower threshold. Following were its results:

Upper bound = 1786.00000

Lower bound = -897.2000

Files below lower bound = 0

Files above upper bound = 11

These 11 files above the upper threshold were removed and a total text of 5554 links were used for further preprocessing.

2.2 DATA PREPROCESSING

Raw data scraped from the webpages was not feasible for analysis and topic modeling as the data must have contained extra white spaces and contained those extra words which could have disturbed topic building. Hence data preprocessing is a key methodology that is needed to be adapted.

- Data Cleansing : In this particular process all extra white spaces, numbers (which is not necessary in topic building as it will give no-sense), and other html tags such as /w,/s etc.
- Stop Words Removal: It is basically removal of set of commonly used words in any language. In this we had increased the set of stopwords so as to create a cleaner data set for topic modeling. The set of stopwords contained all conjunctions, articles (a/an/the), and other unnecessary words which we came across while analysing the data set that were not required in topic modeling. The set was made with the help of frequency distribution plot for each word. It helped in extending the stopwords set.

```
['crore', 'first', 'said', 'work', 'also', 'case', 'would', 'take', 'time',
 'last', 'year', 'three', 'make', 'nthe', 'need', 'even', 'issu', 'well', '
come', 'made', 'come', 'howev', 'work', 'februari', 'like']
```

- Tokenization : In this process, text is splitted into smaller pieces of words known as tokens. These tokens help in understanding the context or developing the model for the NLP. It is basically creating a vocabulary for machine learning model.

- Stemming : Stemming basically breaks the words to its root word that contains only the suffix and prefix of the word. It is important as all additional forms of word are reduced to their root word. For ex: Words like “minister”, “ministry”, “ministers” are all reduced to their root word “minist”. It is important as it may happen that each form of word may take a separate topic in topic modelling.

- Lemmatization : In Lemmatization as well the words are broken into its root word called ‘lemma’. Although both stemming and lemmatization perform same task there is a difference between them as in lemmatization the algorithm chops the word in a way that makes sense in the language. However stemming just chops out the word without knowing the actual meaning.

- Bigram and Trigrams: Bigram and trigram simply makes a sequence of either two or three words, respectively.

2.3 SPLITTING DATA TO TRAIN AND TEST SET

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. Training dataset trains the algorithm for machine learning model whereas test dataset acts as an input so the ML algorithm can make predictions. The model is first to fit on the available data with known inputs and outputs. It is then run to make predictions on the rest of the data subset to learn from it. This can be used to make predictions on future data sets where the expected input and output values are non-existent.

In this 90% of the dataset is used for training the model and remaining 10% of data is used for testing the model. 4998 datasets were under training dataset whereas 556 datasets were under test dataset. The scikit-learn Machine learning library was used as it specifically contains train_test_split() function.

2.4 TOPIC MODELING