<u>Keyword Extraction</u>: Implemented by Sindhu Kopparam(sgk6)

The main purpose of a keyword extraction function is to help us in identifying the important keywords. Extracting specific keywords from a chunk of text will make it easier for the user to decide whether they want to read that article, social media comment, reviews, etc. Analyzing the unstructured text and choosing what to select is made easier by keyword extraction. For this project, we are implementing a function that takes a chunk of text as an input and extracts the specified POS tags (Noun, Verb, Adjective, etc.) as an output.

Implementing this function in MeTA will enhance the library and allow users to extract keywords effectively. Let's look at the set up and the implementation of this function using SpaCy library.

Setup:

1. Install the spaCy module via the pip install.
   pip install -U spacy
2. Next step would be to download the English model. We are using small English model for simplicity reason. Download "en_core_eng_sm"
   python -m spacy download en_core_web_sm
3. Run the following command to validate if spaCy is working properly.
   python -m spacy validate

Implementation:

- We will firstly import the libraries needed to implement this function. One library essential for this implementation would be spaCy as we are using its language model for text extraction.
  Apart from spaCy another library needed would be punctuation that contains the most commonly used punctuation.
  import spacy
  from string import punctuation
- We can now load the language model that is downloaded using the below command.
  nlp = spacy. load("en_core_Web_sm")
- We can now write a function *extract_keywords* for the keyword extraction that can be easily called anywhere when we need to extract keywords from a chunk of text. This function will accept a string as an input parameter.
- The output is stored as a list containing all the keywords extracted. We will create another list *"pos_tag"* that contains the part of speech tags that we want to be extracted. I have chosen Noun (NOUN), Adjective (ADJ) and Verb (VERB) in the above code but this can be customized to our requirement.
- Next step is tokenization. The input text is converted into lowercase and tokenized via the spacy model. This is stored in an object *doc*. This object contains token objects from the tokenization process.
- The next step would be to identify any stopwords or punctuation in the tokenized text. For this purpose, we can loop over each of the token and ignore if any stopwords or punctuation is part of the text. Move on to the next if it is not.

- We will finally store the output if it contains the pos tags of the tokenized text that we have mentioned above.
- Return the output as a list of strings.
- Let’s test the above code by using a sample text.
  sample_output = extract_keywords (‘This program will extract keywords from the input document and print all the nouns, adjectives and verbs.’)

  The output will look like this:

  ['program', 'will', 'extract', 'keywords', 'input', 'document', 'print', 'all', 'nouns', 'adjectives', 'verbs']

- There is also an option for the user to enter text of their choice which will then be served as an input for the keyword function to extract the words.