Schlumberger's New Year Hackathon

Team Name: Need the hope

Members:

Sanyam Jain

Implementation:

Dataset:
I have used two of the websites:
https://climate.mit.edu/explainers
https://netl.doe.gov/carbon-management
for the dataset in my model.

Data pre-processing:
I have used beautifulSoup for web crawling to fetch the headlines of the article, get the URL of the article, and the written article. As the website has a different type to name the article headlines and the text in it. So, I have found the common class name or tags or id by which using a single code I can fetch all article headlines and their article content from a website directly.
For the corpus fetched from an article they have different tags present as sub-elements, punctuations, some brackets, special characters, numbers, and also stop words for which I have used re library to clean the data and make it full of characters from a-z (in lowercase) and nltk stopwords list to remove the stopping words like (the, at, in, etc.) and also the short words like (u, r).

Vectorization:
I have used TFIDF vectorizer because it can count the words and use some formulas like IDF, and IF to find the most important words from them. So, that we can limit the maximum number of features, I have set the value of the maximum number of features as 1000. This is more helpful than counting vectorize or bag-of-words as they can't limit the number of words and can overcome the drawbacks like more dimensions.

LSA (Latent Semantic Analysis):
As the number of dimensions is still large so I have used LSA, why I did not limit vectorization itself? Because LSA helps us to find the relation between the words and give us the best of them not just remove the words which have less count for example if we have two times 'long' and one-time 'carbon' word then by using vectorization the 'carbon' word will be removed but by using the 'carbon' word will not.

Model:
I have used the clustering model because we don't have anything to classify or any amount to predict we have different URLs that we should cluster so that we can get the articles with similar matches and can use the corpus from different URLs to summarize the articles into a summary based on the feed asked. As, in K-means, the most difficult task is to find the k value for which I have elbow method, where using a for loop I am finding the best clustering

and plotting it to find the k value at which the plot gets edge (elbow) and use that edge to train the model.

Input:
Using the input method in python I am taking the input for the feed and then I am also pre-processing the feed, vectorizing it, and then also using the made lsa model to transform it.

Summarizing the feed:
From the feed, I am predicting the cluster number in the K-means algorithm then after finding the cluster number. All the URLs in that cluster are fetched from the made dataset and then I fetch the corpus of the article from the URLs. Then by using the heap queue library, I am summarizing the corpus based on their score, to find the score I first made the word frequency to get the count of each word in a sentence, and then by using the word frequency I have found the total score of the sentence. Then by taking the average score as the limit I found the most valuable sentences and then join them to form a paragraph.

Paragraph phrasing the summary:
As the summary is collected and fetched with many operations in between they may be chances that the summary is getting very confused. So, to correctly phrase the summary I have used the transformers. In that using the model and tokenizer I was able to find a good phrase in the summary. I have also used textbold to correct the spellings of the words in the summary.

Code:

Github code link – https://github.com/Sanyam-jain30/Need_The_Hope-Energy_data_feed_using_NLP