**CE706 - Information Retrieval 2023**

**Assignment 1**

2207938

# Elasticsearch

The following was my approach towards a robust and effective search solution for Essex News Search company, using Elasticsearch and Python to process and analyze the Signal Media One Million News Articles dataset.

I had initially tried ingesting the downloaded signl media dataset into elastic search(kibana) using Bulk API, but the connectionns between my command prompt and elastic serach was inconsistent, so instead of loosing data maybe by splitting so as to reduce the file and thereafter just copying and pasting on Kibana, I opted to use Python which permitted the total upload of of the dataset.

I called a signal\_media\_index and defined the mapping for the index specifying the content field as a text type, and defining multiple fields for the content field, each with a different analyzer. I then definethe analyzers that will be used for the content field. Generating four analyzers: tokenization\_analyzer, stopword\_removal\_analyzer, ngrams\_analyzer, and stemming\_analyzer, I defined two filters: ngrams\_filter and stemming\_filter, which will be used by the ngrams\_analyzer and stemming\_analyzer, respectively. I created the custom analyzer by first closing the index, filter the index settings and then opening the index again.

After having uploaded the extracted dataset, I then mounted my google drive to Colab to enable easier opening. I then identified the path to the Signal Media dataset. I then defined a function called preprocess\_content() to preprocess the content field using the content\_analyzer, opening the file and looping through each line.

For the TD-IDF similarity, it was defined using the tfidf\_similarity dictionary with the scripted\_weight similarity type, then placed the put\_settings() method to set the tfidf\_similarity as the default similarity for the index.

To measure the performance of the analyzer this was done:Text

Description automatically generated

And this was the output: the analyzer has lowercased the text, removed the stopwords ("the", "over", "the"), and stemmed the remaining words ("quick" instead of "quickly", "lazi" instead of "lazy").

*Text, letter

Description automatically generated*

# Named entity recognition

To do this, I loaded the SpaCy English Mode (a python NLP library) that could easily extract name entities and reeturn them as semicolon strings, l then defined a function that takes text as input, thereafter looping though the ‘data’ file and extract named entities from the content field using the extract\_named\_entities() function. I then added the named entities to a new field that I titled content\_ner field. This field was then populated with the named entities, where I then defined a new analyzer that’ll easily read from the field and make the named entities serchable.

Thereafter, a new analyzer content\_ner\_anlyzer that uses a pattern tokenizer to split the input text into terms based on the semicolon delimiter called content\_ner\_analyzer was defined. The lowercase filter was also specified to convert all terms to lowercase. Finally, mapping the content\_ner field to the content\_ner\_analyzer analyzer permitting the search for exact matches on the content\_ner field. Using example of the term ‘New York:

*Graphical user interface, text, application, email

Description automatically generated*

# Indexing

To carry out the indexing, I had earlier defined a function called preprocess\_content() to enable preprocessing of the content field using the content analyzer during my elastic search, where I loaded and index the preprocessed document with the mappings, I then looped on the entire dataset and made a break at 1000. This processed the first 1000 documents from the Signal Media dataset, adding the content and content\_ner fields to the index using the analyzers specified in the mapping

Unfortunately, all of a sudden I started getting connection error, I troubled shoot countlessly running the Elastic search and kibana bin all over again, but yet still nothing. I also noticed that when I tried running every other cell having es.index the error popped up. This wasn’t occuring when I started so I really couldn’t decipher what connection was obstructed, considering the fact I was using a computer sytem in the Lab.

*Graphical user interface, text

Description automatically generated*

# Searching

For textural serches I searched for specific texts that could likely be in the datset user and a user might come up with : Climate Change, Apple, and United states Election. The following was how I created the textural queries.

Text

Description automatically generated