University of Essex

Department of Mathematical Sciences

Assignment 2: Elasticsearch & Evaluation

Submitted as part of the requirements for:

CE706-7-SP-CO: Information Retrieval

Name: Divya Arora

Registration Number: 1901423

Name: Aritra Ganguly

Registration Number: 1906467

Supervisor: Alba Garcia Seco de Herrera

Date of submission (9th April 2020)

Introduction

This task is majorly focusing on Elasticsearch.It is open source search engine known for text search and analytics. It is capable of handling complex search features and big data by storing, searching and analyzing rapidly. With the fact that it provides distributed system and have compatibility with JSON it is used to carry out the process of indexing [1].

We have to proceed with the task with the consideration that we are new to the organisation where they are in urgent need of a search engine that helps the employees to search the document collection [2].

1 Main Body

Our task is to convert documents into structured index by using information retrieval models and evaluate them. We have to follow certain steps to fulfill our requirements i.e. indexing, searching, building a text collection, evaluation, engineering a complete system [2].

1.1 *Software Requirements*

There is a need to use three software for transforming the documents.

- Elasticsearch
  - Download Elasticsearch software.
  - Extract Zip file from download folder.
  - Right click on Elasticsearch batch file and run as administrator.
  - Command prompt window open and continue running.
  - To check whether it is installed properly.We need to open any web browser and use this URL: http://localhost:9200/

Elasticsearch is used for creating indexes for the documents.

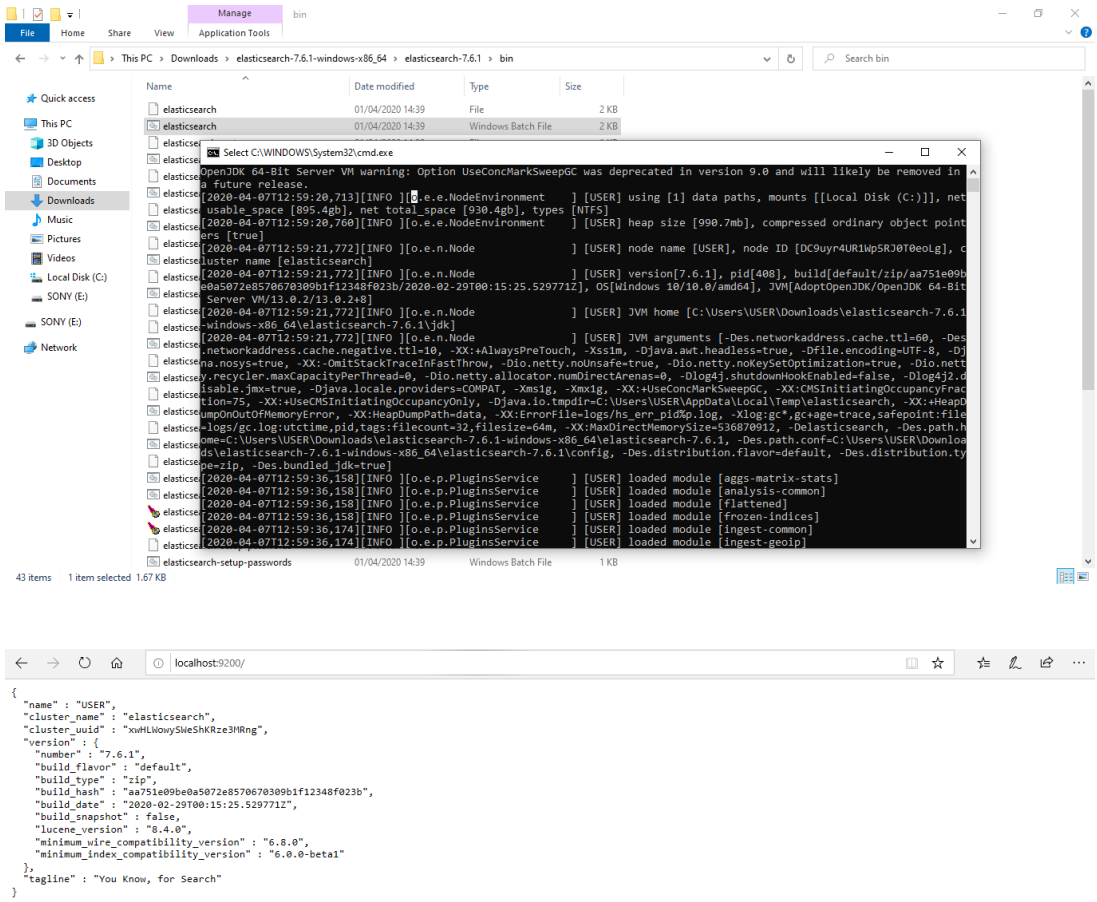


Figure 1: Running Elasticsearch

- Kibana
  - Download Kibana software.
  - Extracting Zip folder.
  - Right click on Kibana batch file and run as administrator.
  - Command prompt open showing the execution of software.

– To check whether Kibana is installed, open web browser and access URL: http://localhost:5601/

Kibana is used to search for the index created using Elasticsearch.

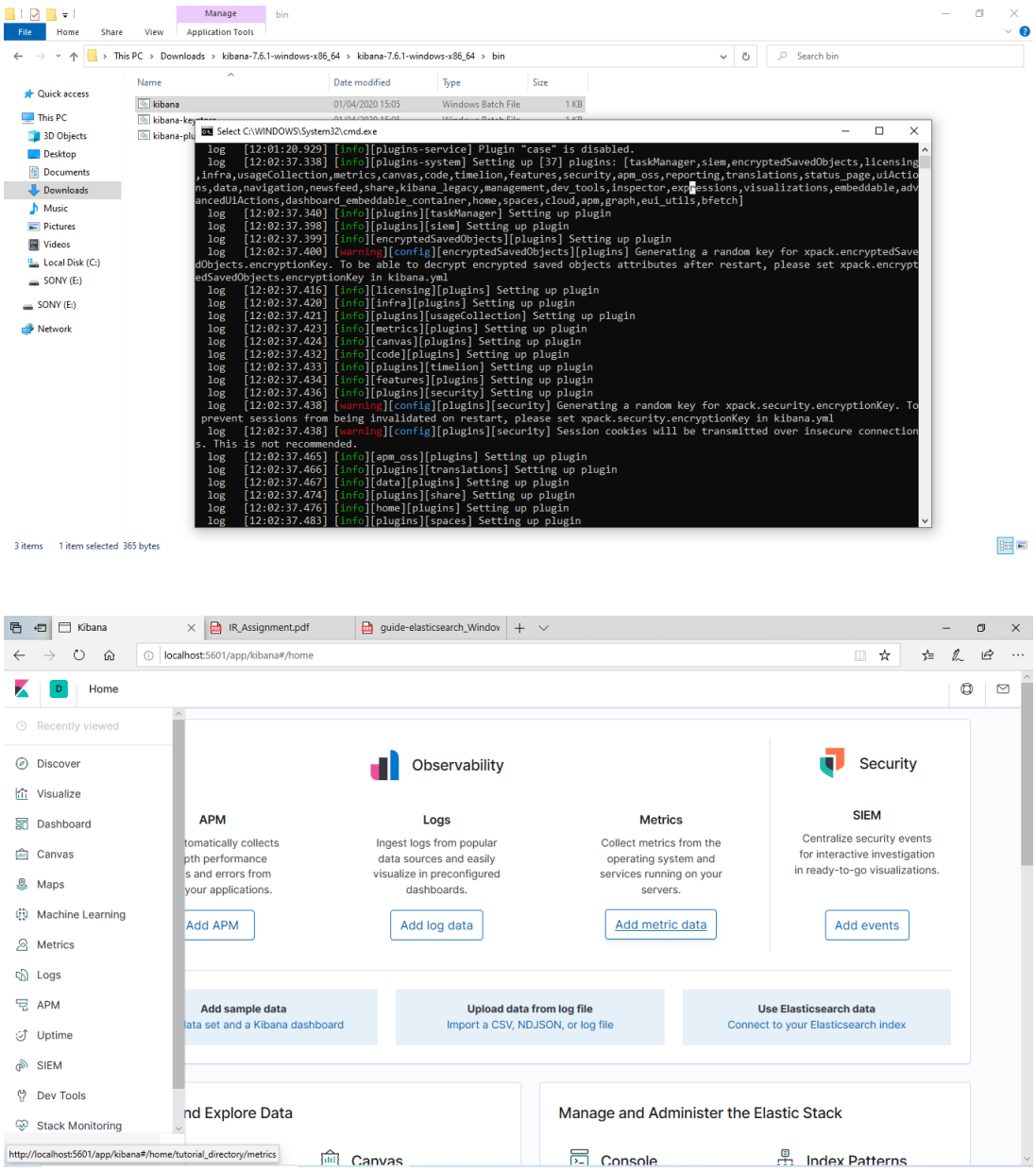


Figure 2: Running Kibana

- Python Python 3.8.0 to access the data in Kibana for indexing and searching.

  Three packages of python have been used i.e elasticsearch to connect with Elasticsearch software running in the system, json_lines to read lines where each line is json encoded, json for reading and writing from json file and to convert string into json file. Along with these three important packages we used mathematical library numpy and nltk library.

1.2 *Indexing*

The group of documents having indistinguishable attributes referred to as index. It is recognised by its distinctive name while carrying out task such as searching. Cluster are nodes that collectively defines the data and are capable enough to provide indexing and searching of documents. There is no limit of indexes with in an cluster it can be as many as we want. Similarly, there is no limitation of storing number of documents in an index [2].

From the dataset, we are using 8000 articles for which we are going to create index using Elasticsearch and python in *indexing()* function of "searchengine.py" python file by following certain steps.

- Download the file used for indexing signalmedia-1m.jsonl.
- Read this json file by using json_lines package.
- Output obtained in the form of string.
- Convert string into json form using json.dumps() function of json package.
- Load the file in Elasticsearch.

To see the index created in Kibana after proper execution of file.

- Open Kibana site from the browser.
- Click on Management option on left side of the software.
- Select Index Management.
- Check whether the index is created or not.
- Click on the respective index and access mapping tab which are automatically generated while feeding the data in Elasticsearch.
- Fields of data: id, content, title, media-type, source, and published

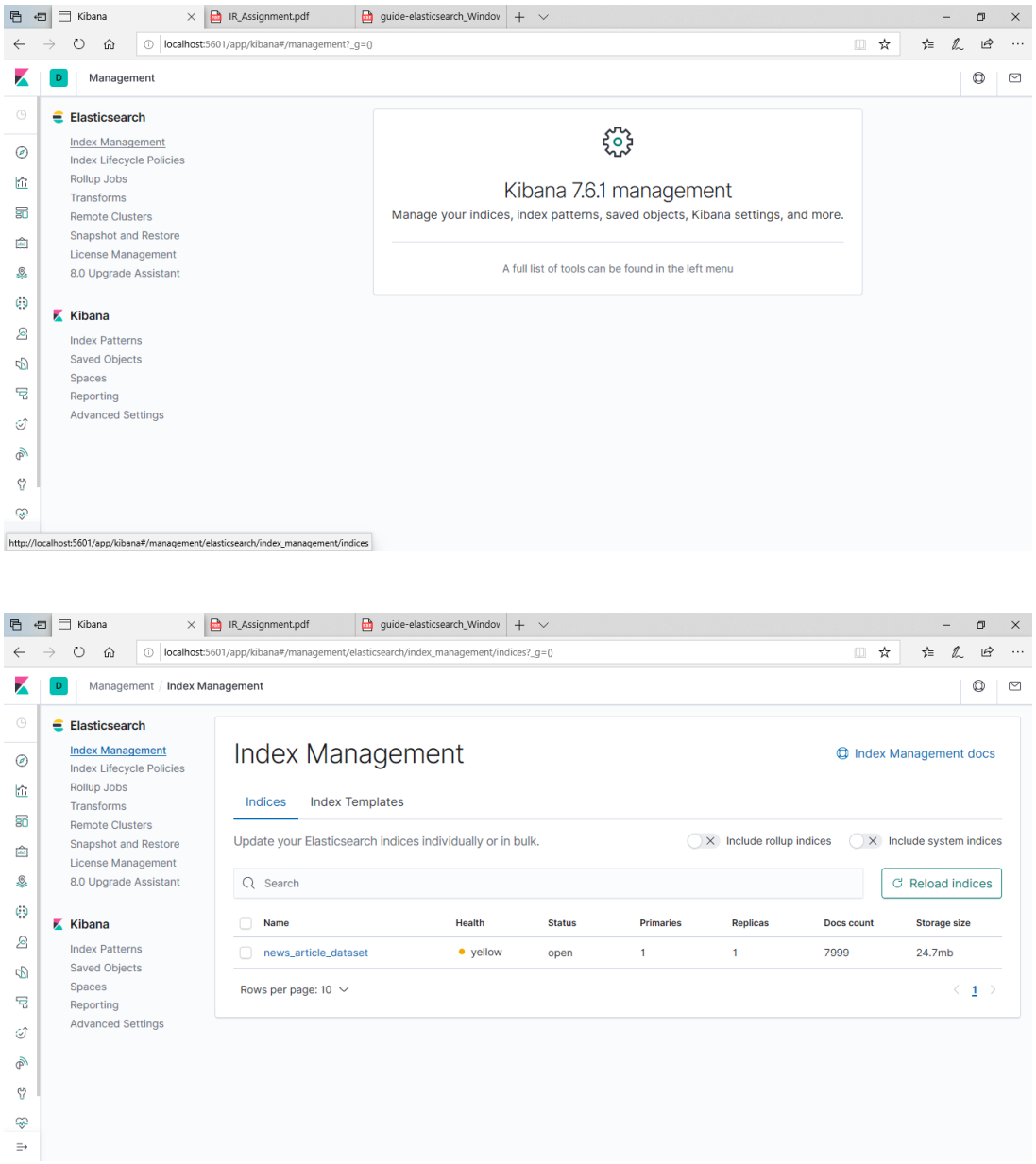


Figure 3: Indexing Json Dataset

In order to see the mapping of the dataset, we need to click on the index first then click to mapping. The mapping are created automatically after we load the dataset onto elastic search using the *indexing()* function.

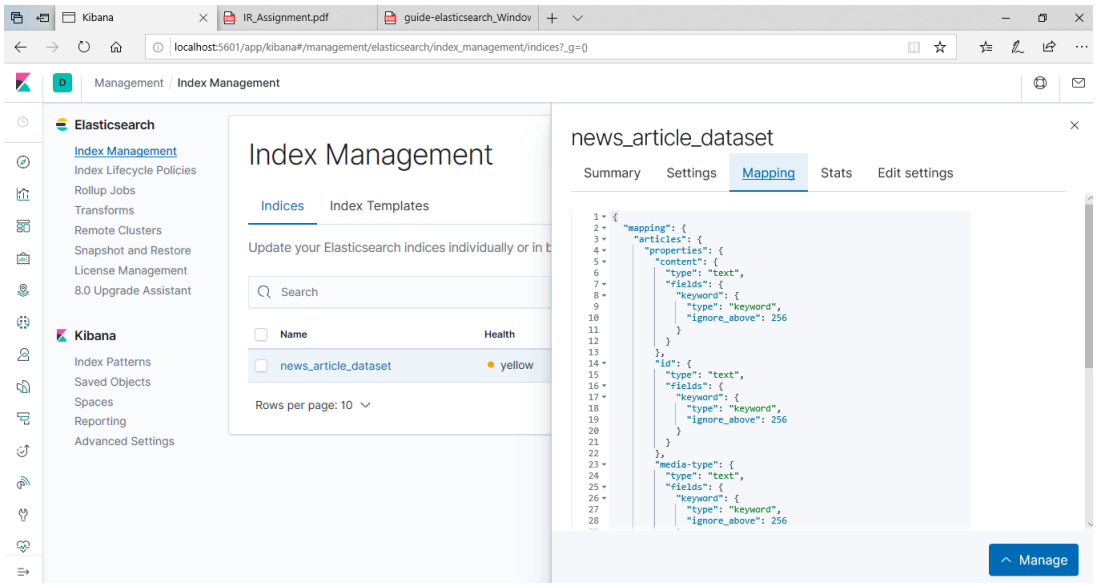


Figure 3: Mapping Json Dataset

1.3 *Searching*

In order to use Elasticsearch there is a need of interface. So we are using Kibana which is a web interface required to do the manipulation as well as visualization of data to execute more complex queries [3]. Along with this, we are searching uploaded files with the help of created index in Kibana site.

- Open Kibana site from the browser.
- Click on Management option on bottom left side of the software.
- Select Index Patterns.
- Click on Create Index Patterns.

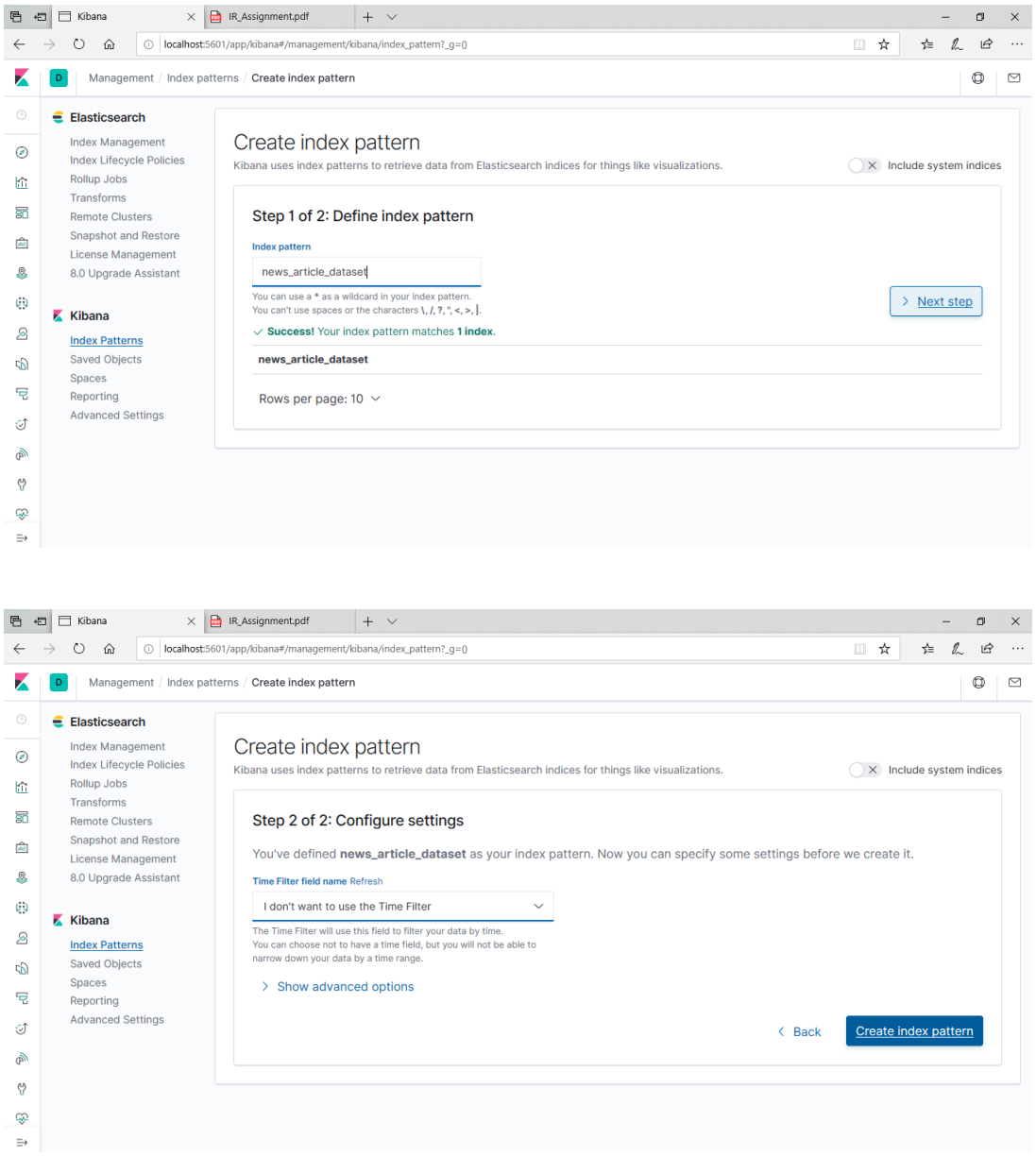


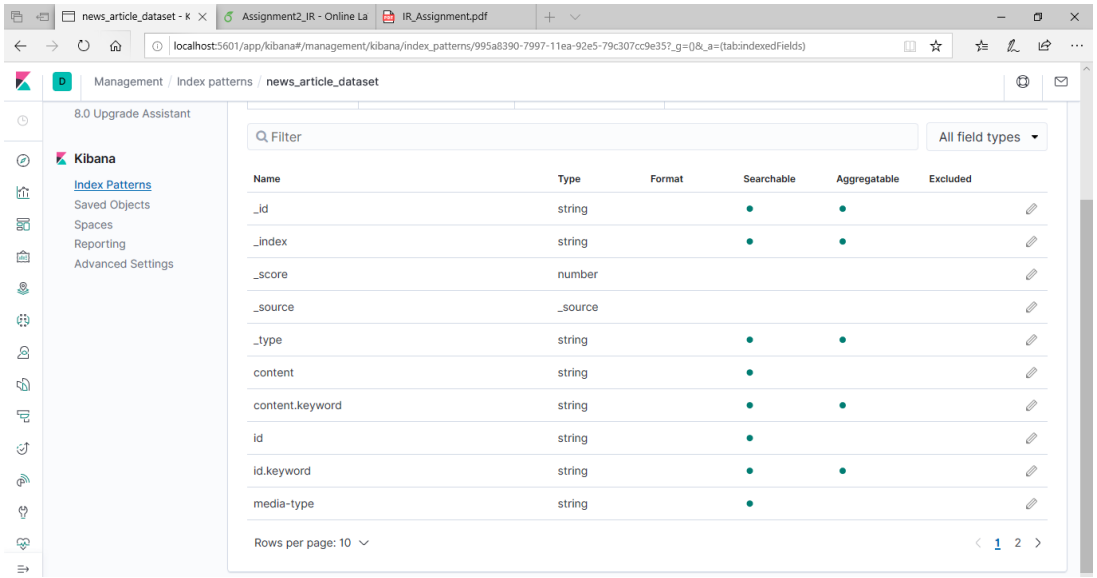
Figure 4: Creating Index Pattern

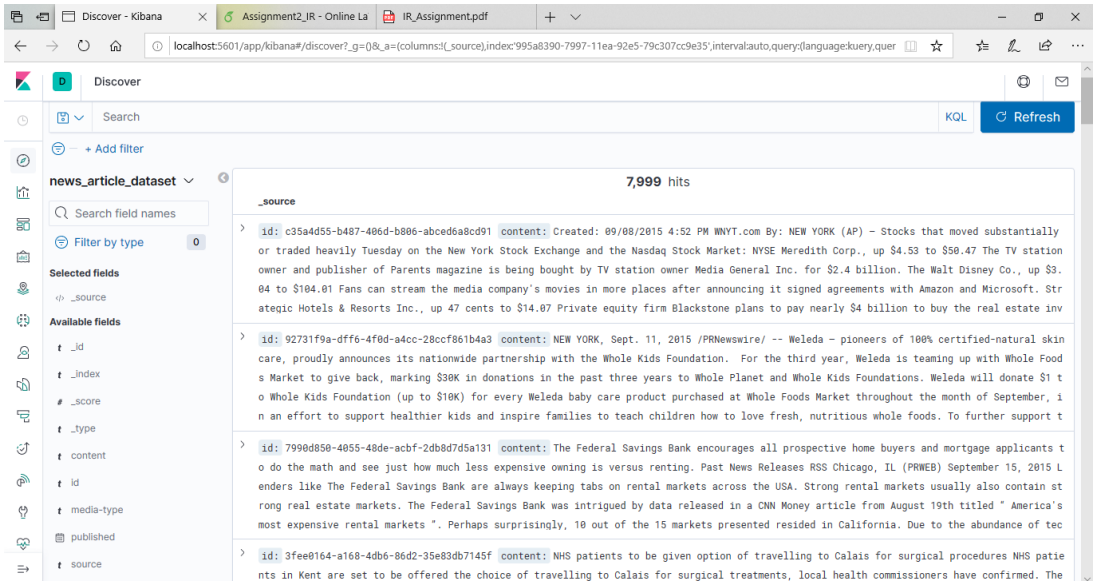
After creating indexes, we need to search indexes which can be done by using discover tab which is visible on upper left corner of Kibana site.

Searching operation is carried out by following below steps:

- Select the keyword to search just like media-type.
- Go to edit filter in Kibana and add the filter with the selected keywords.
- Save the search.

- Results of search are visible.
- We can also add multiple filters just like accidents.
- if we search for accident's then the action related to accidents are visible. It indicates that, there are articles about accident or occurrence of unfortunate incidents which contain in these keywords
- In case we are specifying search and title then consequently it will give the output in a precise way.





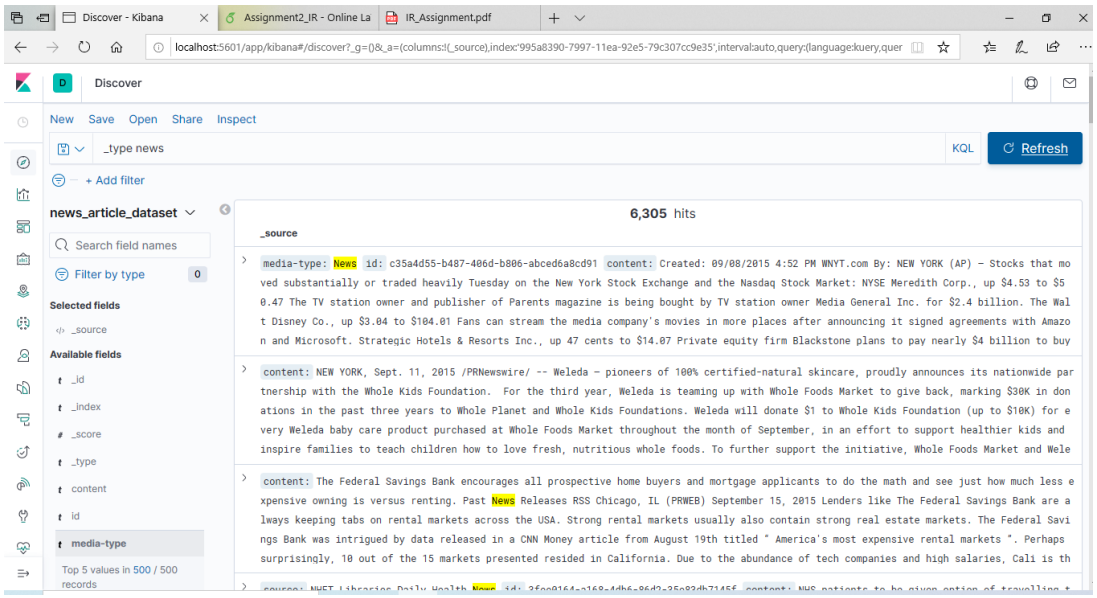


Figure 5: Searching Information

1.4 *Building a Test Collection*

Test collection are developed in order to test the efficiency of information retrieval systems.It gives us the opportunity to explore how can we optimize our search engine for the particular collection we are indexing and make it more efficient. Following test collection has been devised.

Outputs of 6 different queries:

1. Query 1: Query will search title and content of the article.

```
{"query":
        {"bool":
                {"should":
                        [
                                {"match":
                                        {"title":title
                                        }
                                },
                                {"match":
                                        {"content":content
                                        }
                                }
                        ]
                }
        }
},size=8000
```

2. Query matches with the exact title content type

```
{"query":
        {"match_phrase":
                {"title":title
                }
        }
},size=8000
```

3. Query for content which have higher importance than title

```
{"query":
        {"multi_match":
                {"query":keyword,"fields":
                        ["content^2", "title"
                        ]
                }
        }
},size=8000
```

4. Query for content to begin with letter

```
{"query":
        {"wildcard":
                {"content": content
                }
        }
},size=8000
```

5. Query for Source and Content

```
{"query":
        {"bool":
                {"should":
                        [
                                {"match":
                                        {"source":source
                                        }
                                },
                                {"match":
                                        {"content":content
                                        }
                                }
                        ]
                }
        }
},size=8000
```

6. Query to search Media type and content range of published date

```
{"query":
        {"bool":
                {"should":
                        [
                                {"match":
                                        {"media-type":media
                                        }
                                },
                                {"match":
                                        {"content":content
                                        }
                                },
                                {"range":
                                        {"published":
                                                {"gte": start,"lte": end,"format":
"yyyy/MM/dd||yyyy"
                                                }
                                        }
                                }
                        ]
                }
        }
},size=8000
```

1.5 *Evaluation*

Evaluation are also used primarily for measuring the effectiveness of system. It is carried out basically to find out goodness of the system and its performance.

For the evaluation, we are using mean average precision popular metrics as the measure for accuracy. Mean average is the combination of recall and precision for the retrieved outcomes. It is calculated as ratio of number of queries in a set divided by average precision for a given query [4].

The *mapr()* function has been used to carry out the evaluation.

Follow the below steps to evaluate:

- Initialization of precision and recall values equal to zero.
- Evaluate ranked recall and precision for the search engine.
- Array data structure is used to store indexes and sort them for every document retrieved which helps us to compute precision and recall values.
- It is a cycle where precision and recall are calculated at each stage continuously.
- After calculating precision and recall for each document,compute the average of all values of the precision and recall.
- Output is shown with the respective document id,precision,recall,mean average preci-

sion,mean average recall.

Figure 7: Evaluation

1.6 *Engineering a Complete System*

The system has been developed using Elasticsearch, Kibana and python. The system initiate by taking the confirmation that the file is indexed. Then further proceed by firing the query and results will be displayed, followed by displaying precision and recall at every stage of the system and at last showing mean average precision.

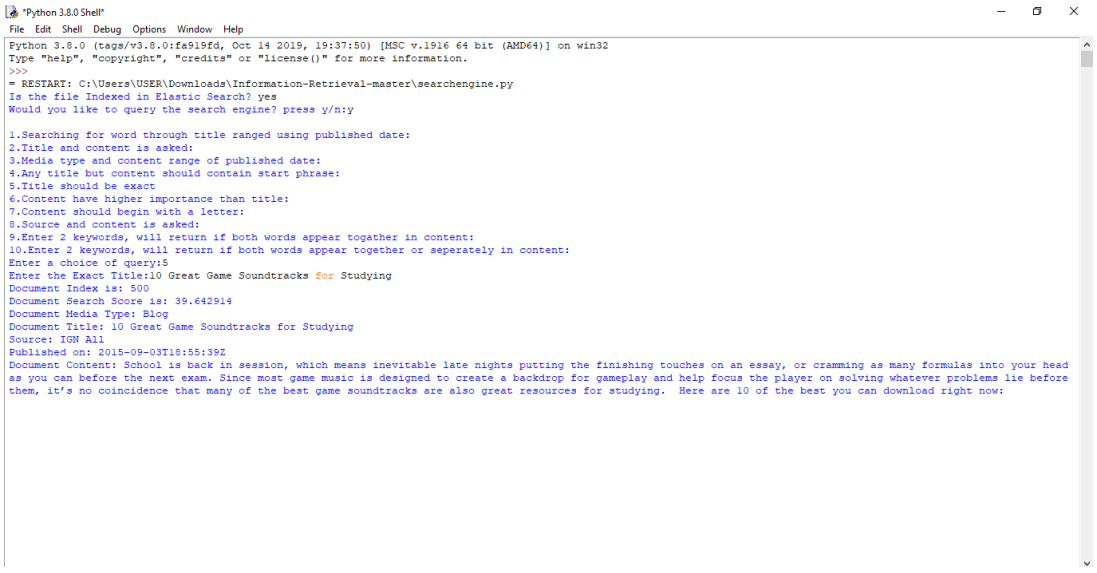


Figure 7: Engineering a Complete System

1.7 *Crowdsourcing*

Crowd sourcing is the process where large group of people contribute by investing their time for a common objective may be any research or any other goal such as innovation, problem-solving. The exercise we have completed majorly test our memory power. It majorly tests our capability to remember the videos and analysing the patterns of different brains. It is a simple test to learn the efficiency of our mind. It is based on reflexive actions taken by the user by coordinating their brain with their action by clicking space bar whenever we found repeated videos. It increases our concentration by doing some task with full dedication for 20 minutes time span. The user experience can be enhanced by having proper instructions at what time we should press space bar at the end of the video or in the middle. The interface can be more lively for the exercise. It can be made more interesting by adding the idea of ramification to it because in some sense it resembles towards playing any mindcentric game. After some videos of seven second there is white blank screen appears which makes the experience somewhat less interesting. We have learned to be more focused and attentive towards our task and user experience can be improved by making the platform more interactive and attractive.

2 *References*

[1] https://towardsdatascience.com/an-overview-on-elasticsearch-and-its-usage-e26df1d1d24a

[2] https://moodle.essex.ac.uk/pluginfile.php/1007586/course/section/139844/assignment_2_306_706.pdf?tim

[3] https://medium.com/@victorsmelopoa/an-introduction-to-elasticsearch-with-kibana-78071db3704

[4] https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52