University of Essex
Department of Computer Sciences

CE706-Information Retrieval
Information Retrieval Assignment 2: Elasticsearch and Evaluation

Submitted By
DILPA RAO: **1906319**
VIRAJ KUMAR DEWANGAN**: 1901181**

Guided By
Dr Alba Garcia Seco Herrera

Date of Submission

9 April 2020

TABLE OF CONTENTS

Abstract

ABSTRACT

In this report, we will explain how we made a search engine of signalmedia-1m.json dataset using Elastic Search Engine and the GUI interface Kibana and loaded the data in Jupyter Notebook using Python. In Kibana we used the Kibana Query Language (KQL) and did simple searches and executed queries and built a Test Collection based on certain events. We used evaluation metrics like precision and recall determining the accuracy of our search results.

KEYWORDS:

Indexing, Searching, building a Test Collection, Evaluation, Engineering a Complete System

A. INTRODUCTION

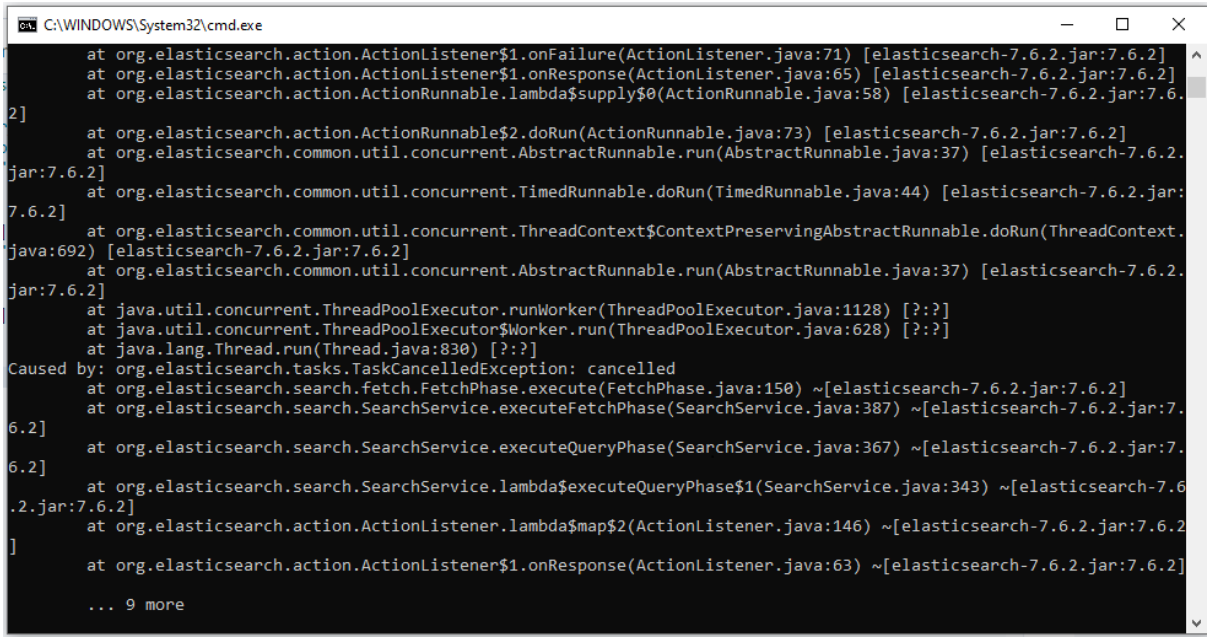
The data was downloaded from the site http://research.signalmedia.co/newsir16/signal-dataset.html and in order to start Elastic search on windows the following programs were used Java, Elasticsearch and Kibana and Python using Anaconda . Out of the 1 million documents, 5000 documents were loaded into elastic search by writing a python script in Jupyter Notebook. The Kibana software tool was used for indexing, searching, building a test collection and evaluation.

Software Installation was done using the following sites:

1. Java : https://www.java.com/en/download/win10.jsp
2. Elasticsearch : https://www.elastic.co/downloads/elasticsearch
3. Kibana : https://www.elastic.co/downloads/kibana

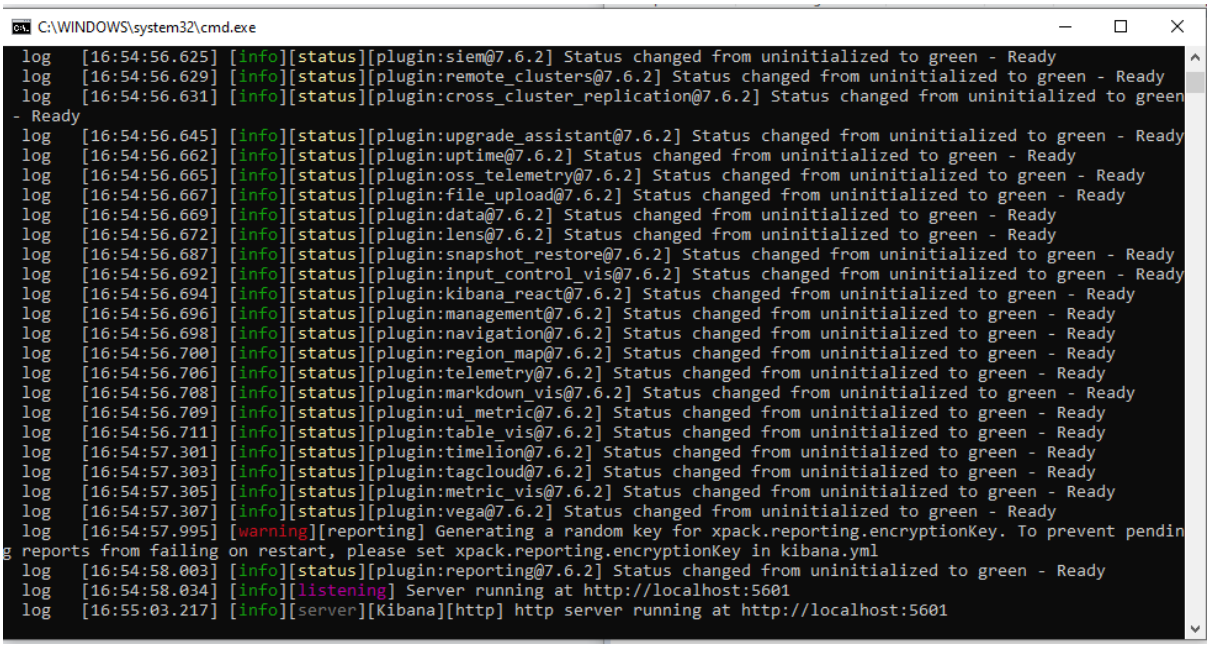
B. SETTING UP ELASTICSEARCH

After downloading the Elasticsearch, it was in a zip format and we extracted the file. The Bin folder was opened, then clicked on the **elasticsearch.bat** and **'Run as administrator'.**



C. SETTING UP KIBANA

Similarly, like Elasticsearch we extracted the zip file and selected the Bin folder and clicked on Kibana.bat and 'Run as Administrator'. Once properly run, it will show an output like this:



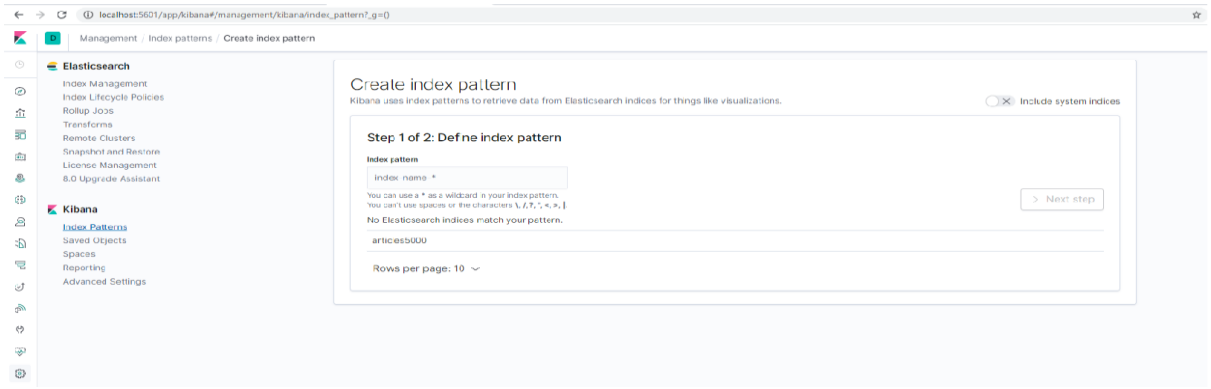
Both should remain working in the background. We can for proper installation by entering this URL for Elasticsearch and Kibana respectively - http://localhost:9200/ and http://localhost:5601/ .

D. LOADING THE DATASET

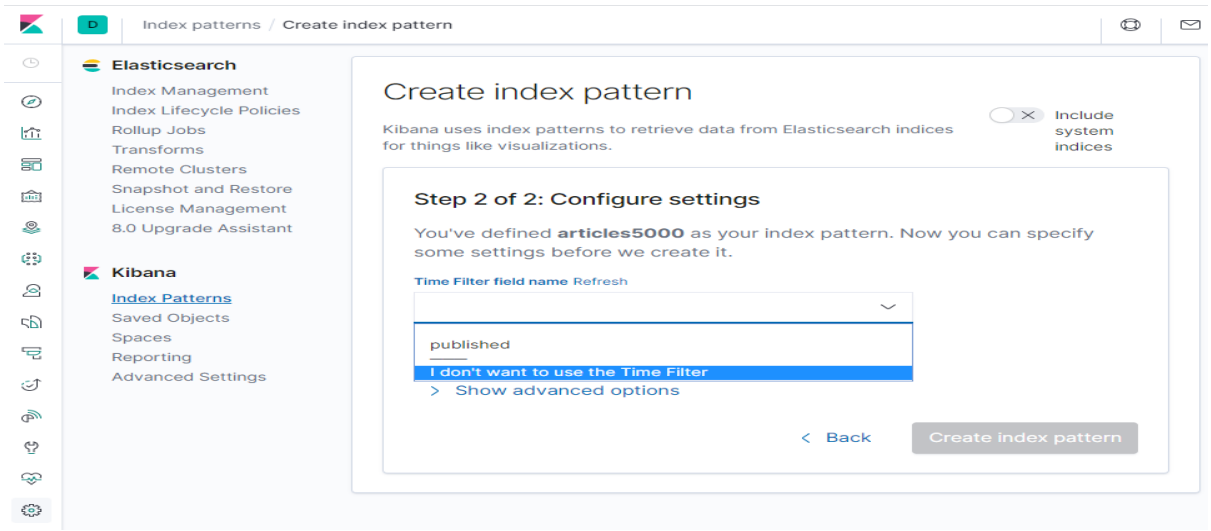
The dataset was loaded in Elasticsearch by running a script file in python. We selected only 5000 documents as loading more data was giving system errors and taking a long time, making the machine hang.

E. INDEXING IN KIBANA

After ensuring that the dataset is loaded, then the next step was indexing. Index was created by the following way:



- In Kibana, open Management, then click Index Patterns.
- Click on Create index pattern and type ‘articles5000’ in the textbox.
- Click on the next step button.
- In Step 2 of 2: Configure Settings click on the down button arrow and select ‘I don’t want to use the Time Filter’.
- Finally click on the ‘Create Index pattern button’.



F. DESCRIPTION OF INDEXED DOCUMENT

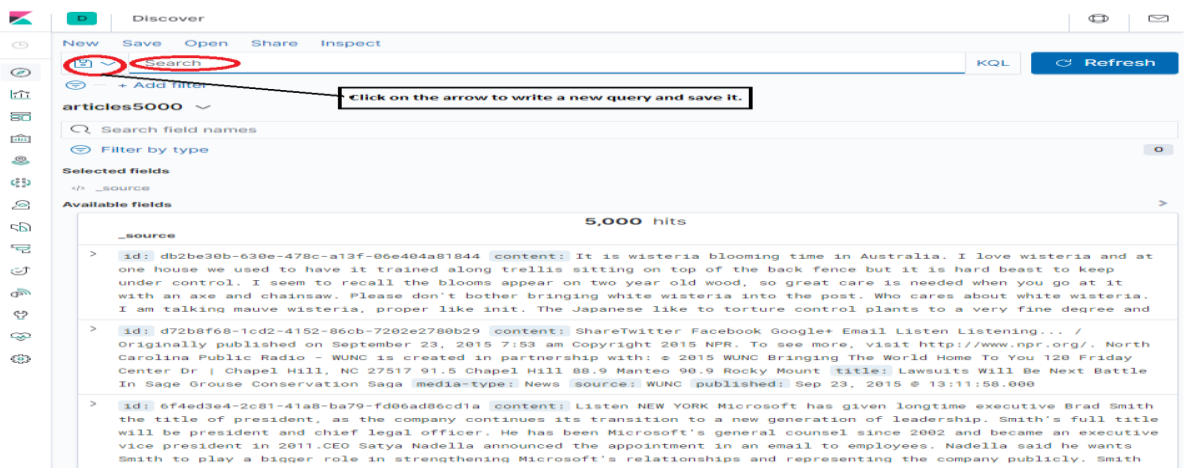
So, now the index pattern is defined on our 5000 documents. It is to be noted that the pattern must exist in the Elasticsearch and it must contain data for an index to be created. We can check for the indices in the management icon the last tool on the left menu of Kibana and click on it to get to Elasticsearch – Index management and Kibana – Index Patterns screen page.

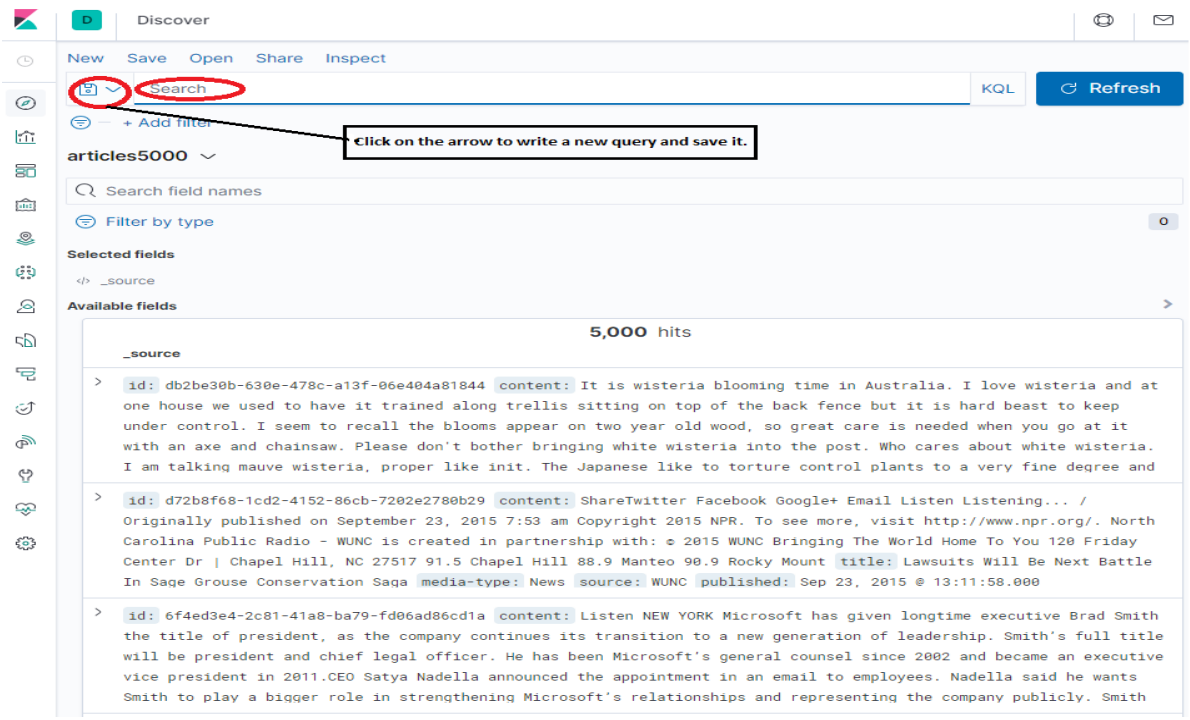
The screenshot of our indexed document is displayed below-

The indexed document contains various fields like source, id, index, type, content, media type, date of publication on which the data is available and can be used to define custom searches and results in Kibana.

G. SEARCHING IN KIBANA

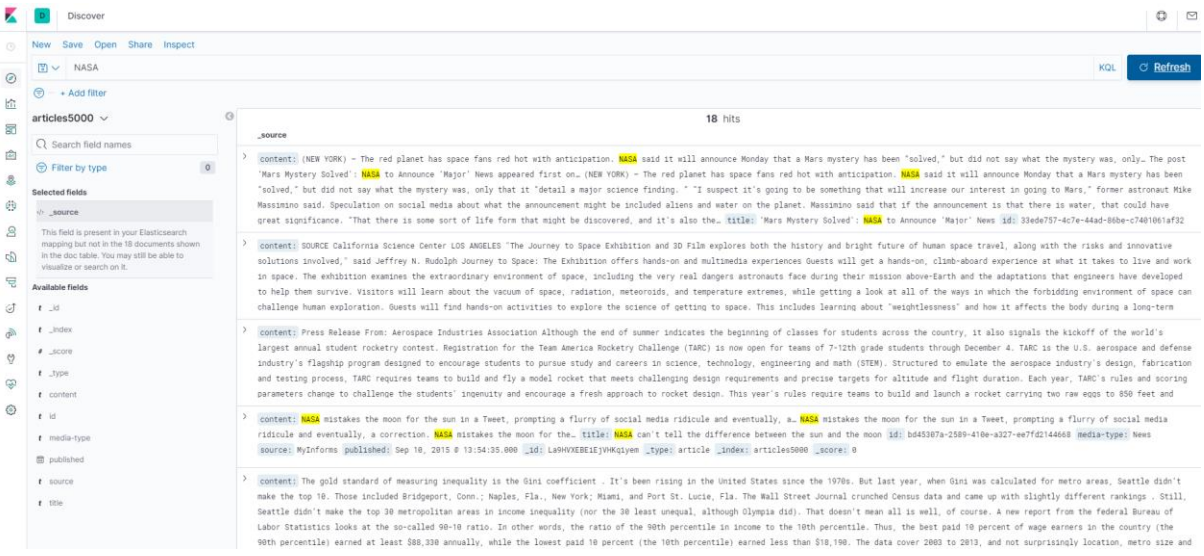
We can do simple search and more complex queries in Kibana. The Discover icon which is on the top of the left menu allows us to search, write complex queries and to save it. We can alternatively write the queries using the Dev Tools icon on the left menu.





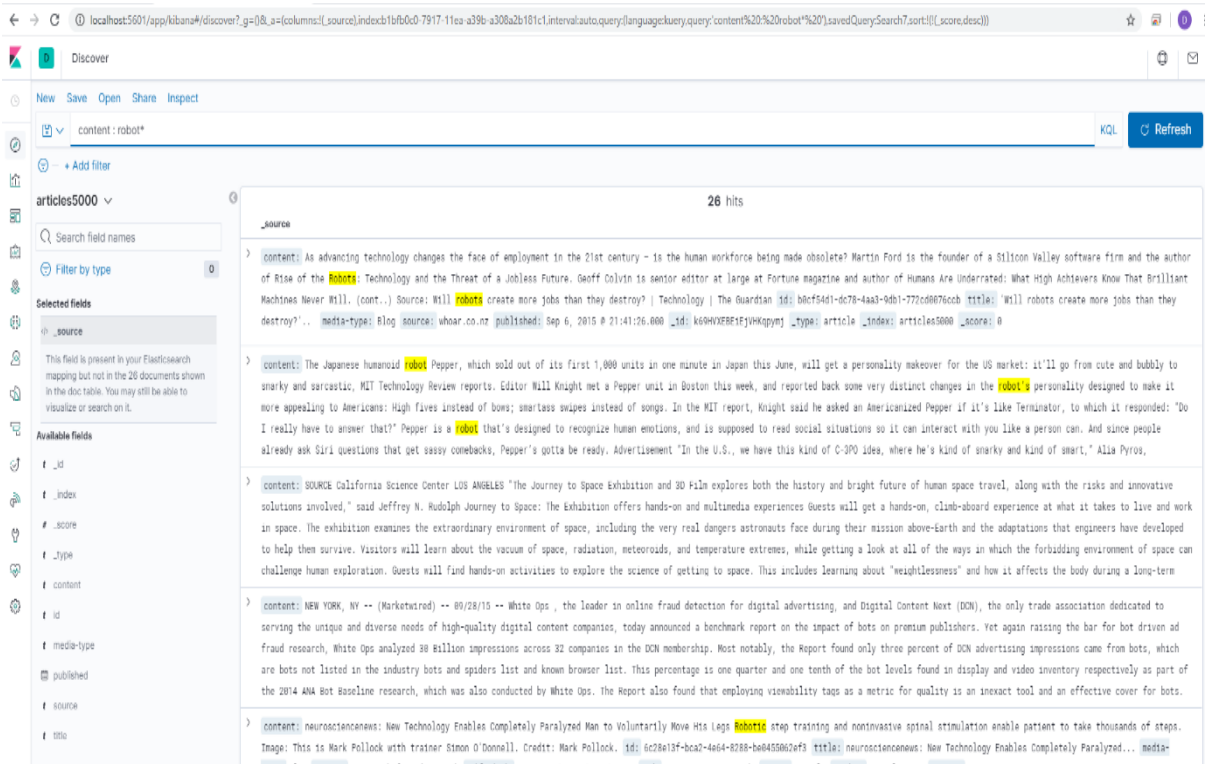
H. TYPES OF QUERIES IN KIBANA

1. Simple Query

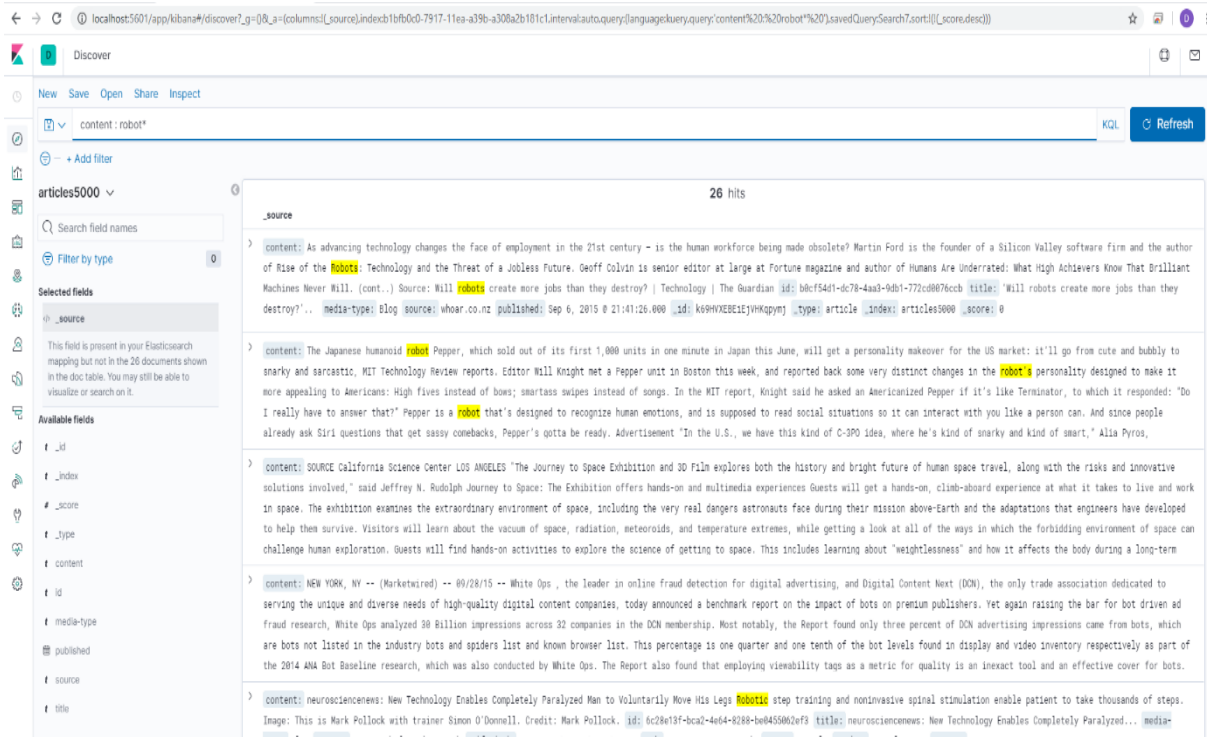


2. Content Based Query

3. Regexp Query



4. Range Query



5. Boolean Query

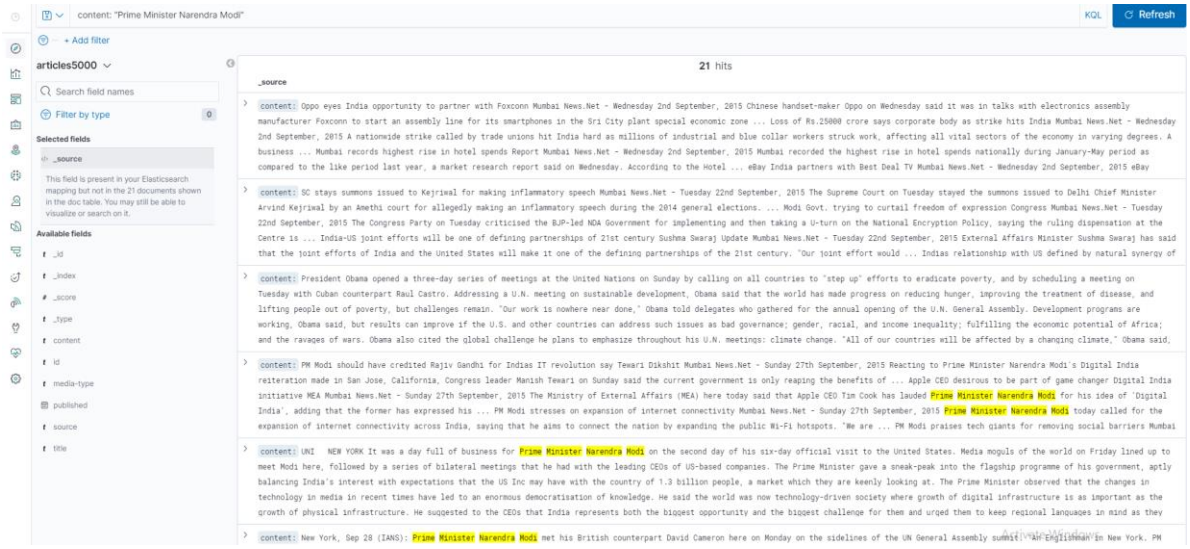
I. BUILDING A TEST COLLECTION

The following 10 queries were built, and their results are captured as follows-

1. Politics

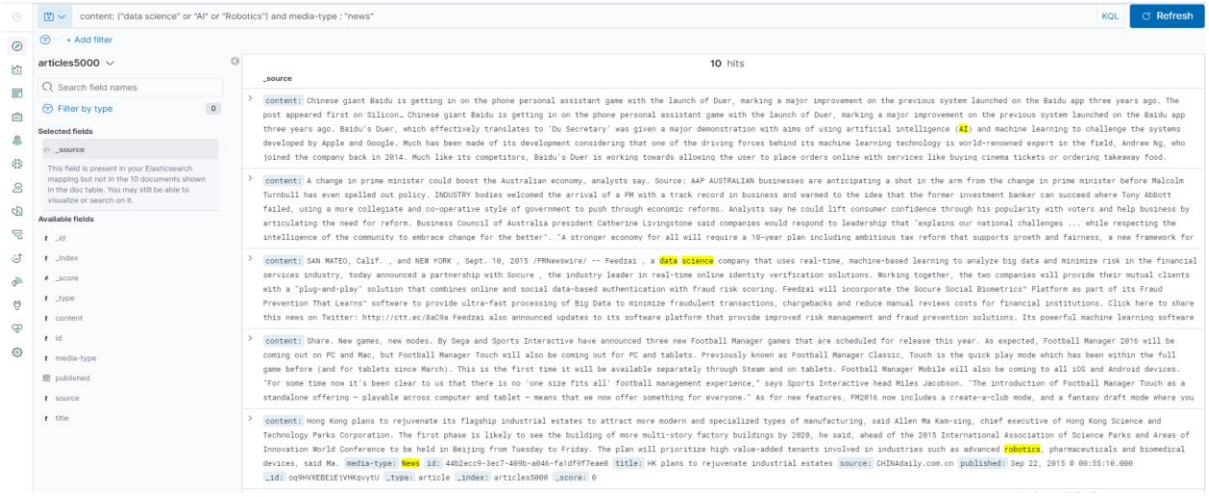
First query retrieves 21 documents based on the content: **Prime Minister Narendra Modi** and adding additional condition of **media-type: "news" and published >= "2015-**

09-15", we retrieve 12 documents. So, the number of hits were reduced from 21 to 12. It can be clearly shown from the figures displayed below.



2. Technology

First query retrieves 14 documents based on the content: **"data science" or "AI" or "Robotics"** and adding additional condition of **media-type: "news",** we retrieve 10 documents. So, the number of hits were reduced from 14 to 10. It can be clearly shown from the figures displayed below.



3. Health

First query retrieves 12 documents based on the content: **"Lung Cancer"** and adding additional condition of **media-type: "news",** we retrieve 9 documents. So, the number of hits were reduced from 12 to 9.

4. Software giants

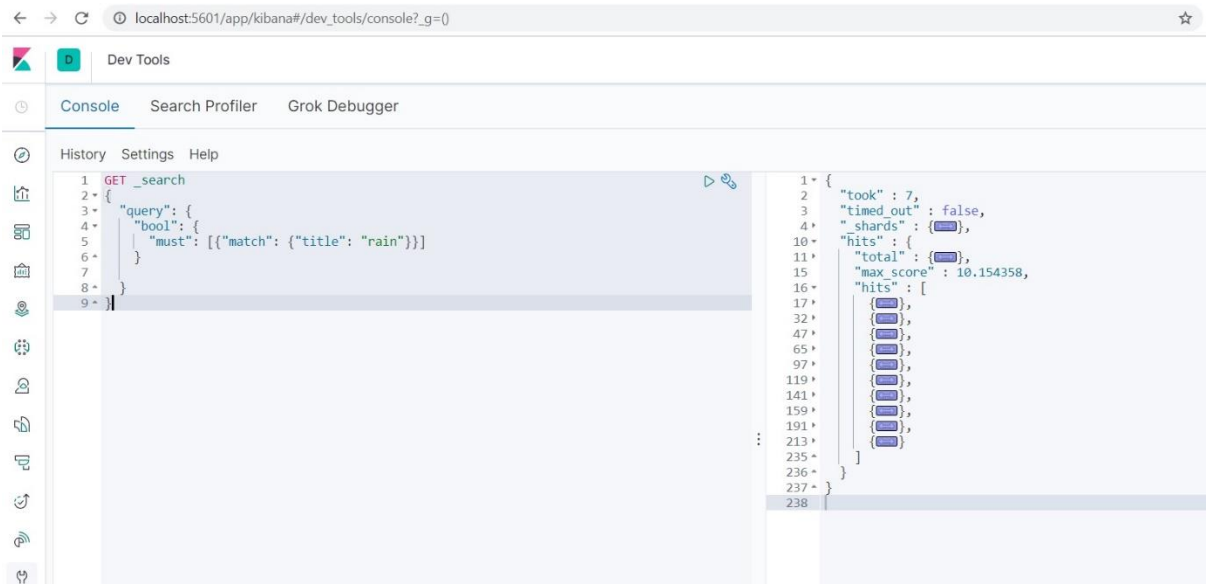
First query retrieves 21 documents based on the content: **"Microsoft Corp" or "IBM"** and adding additional condition of **media-type: "blog",** we retrieve 4 documents. So, the number of hits were reduced from 21 to 4.

5. Sports

First query retrieves 48 documents based on the content: **" Rugby "** and adding additional condition of **"World Cup 2015"**, we retrieve 4 documents. So, the number of hits were reduced from 48 to 4. It can be clearly shown from the figures displayed below.

6. Rain

Rain yields the following search with hit number 10. This is obvious by counting the number of hits in blue rectangles in the figure.

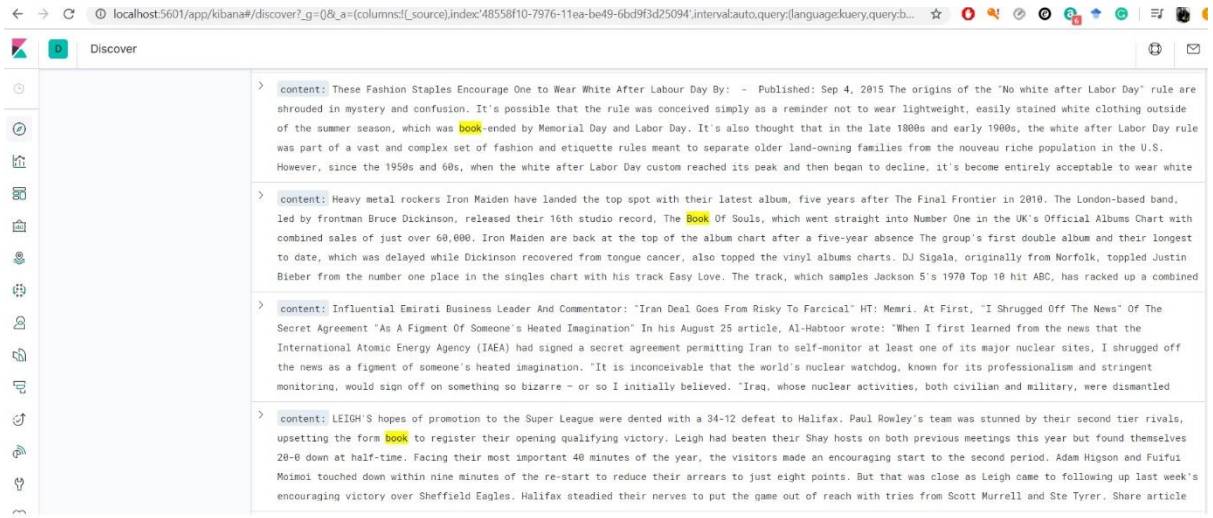


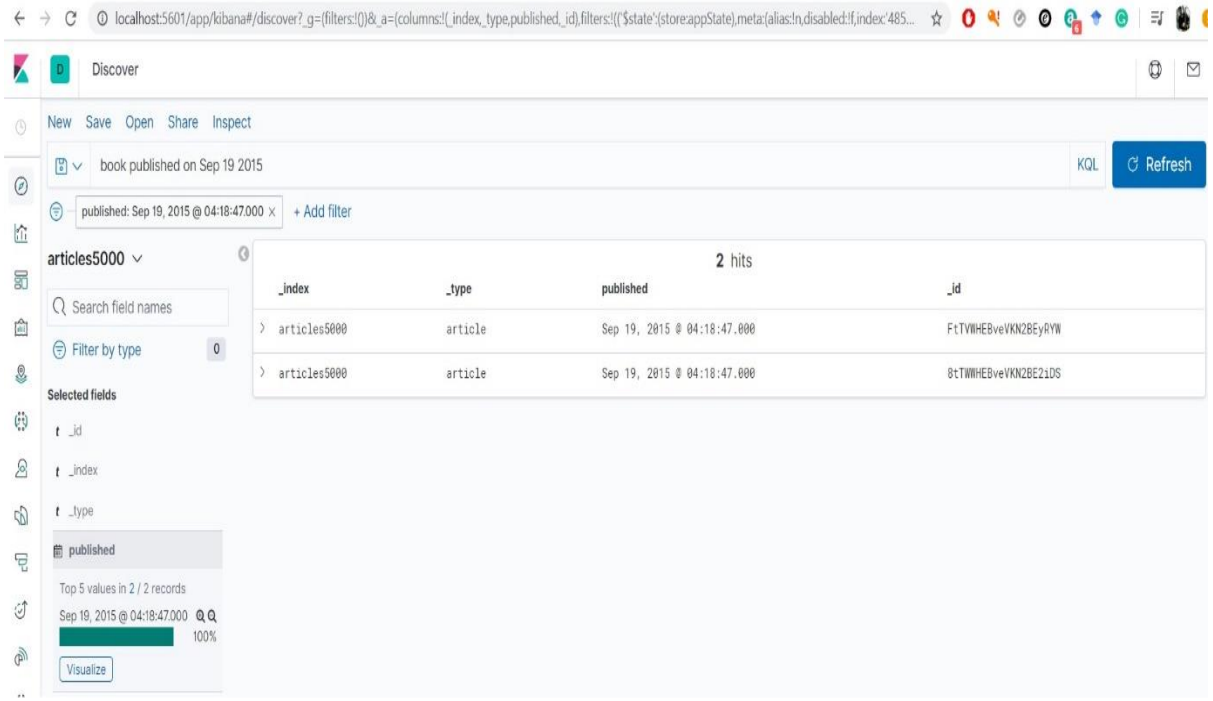
7. Movie

Similarly, the search on movie gives 119 hits which is shown below –

8. Book

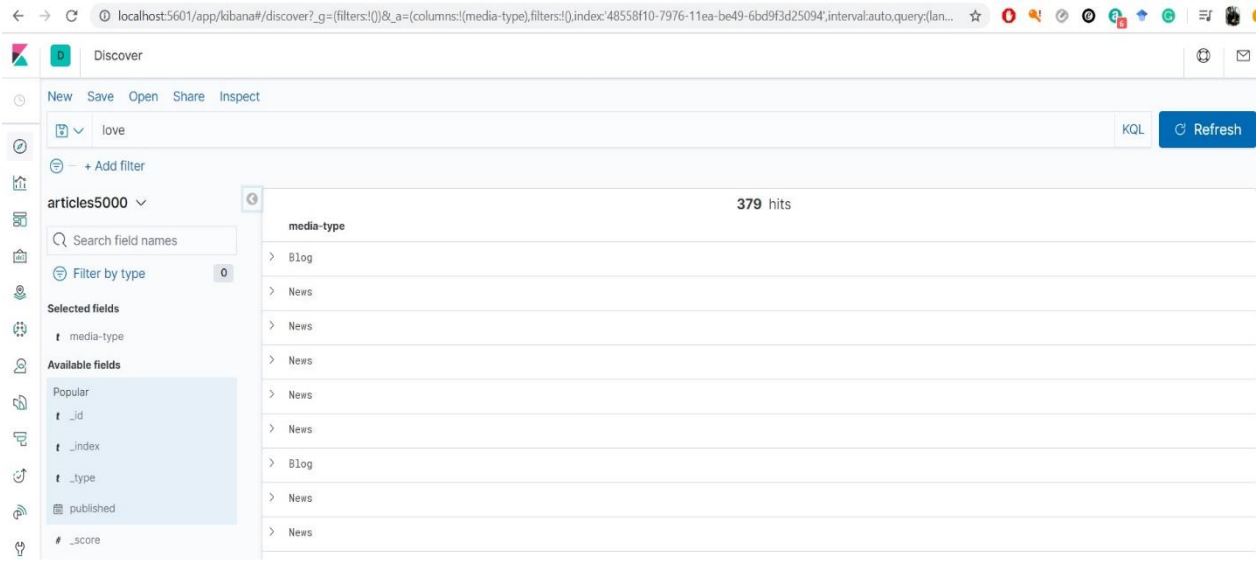
Initial query on the word: “book” gives 211 hits which reduces to 2 hits when publishing date filter of Sep 19 2015 is added. This is displayed below-

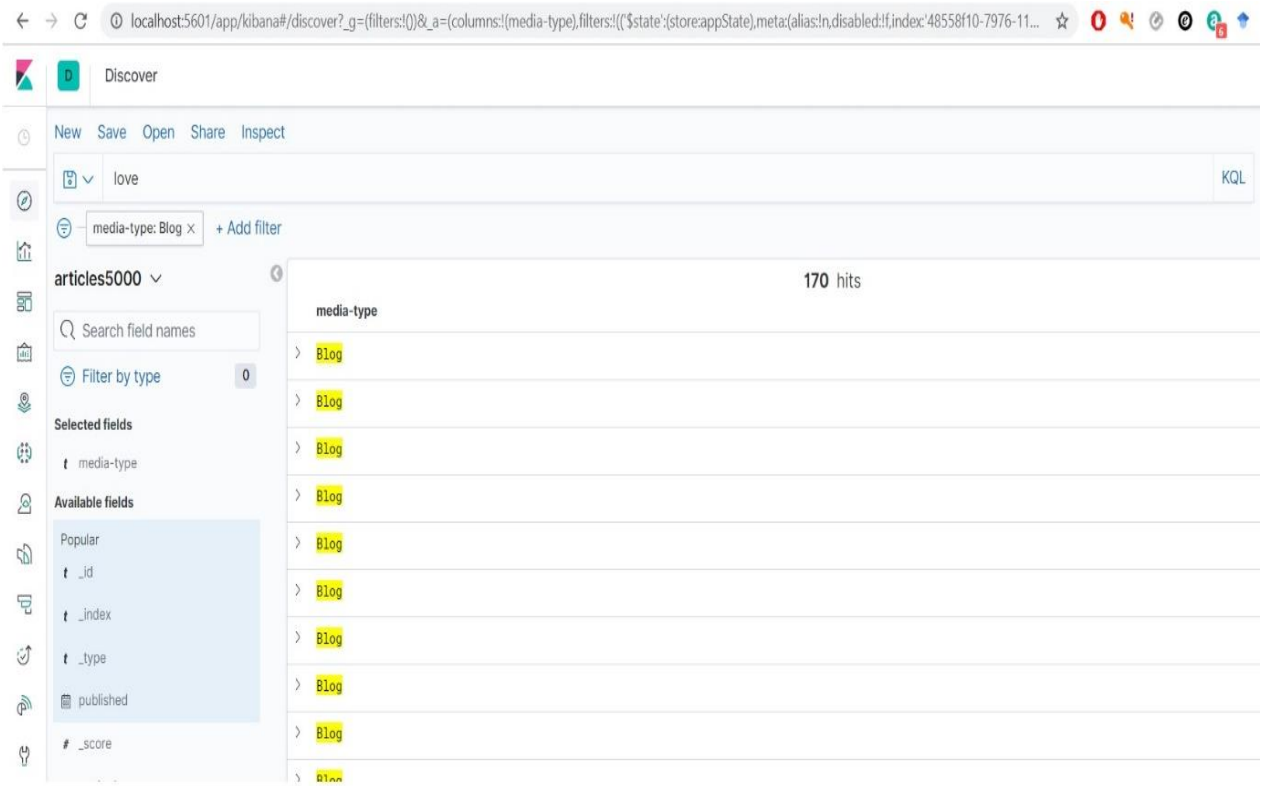




9. Love

The plain vanilla query on Love yields 379 hits and when filtered with media type blog just gives 170 results. This is displayed below –

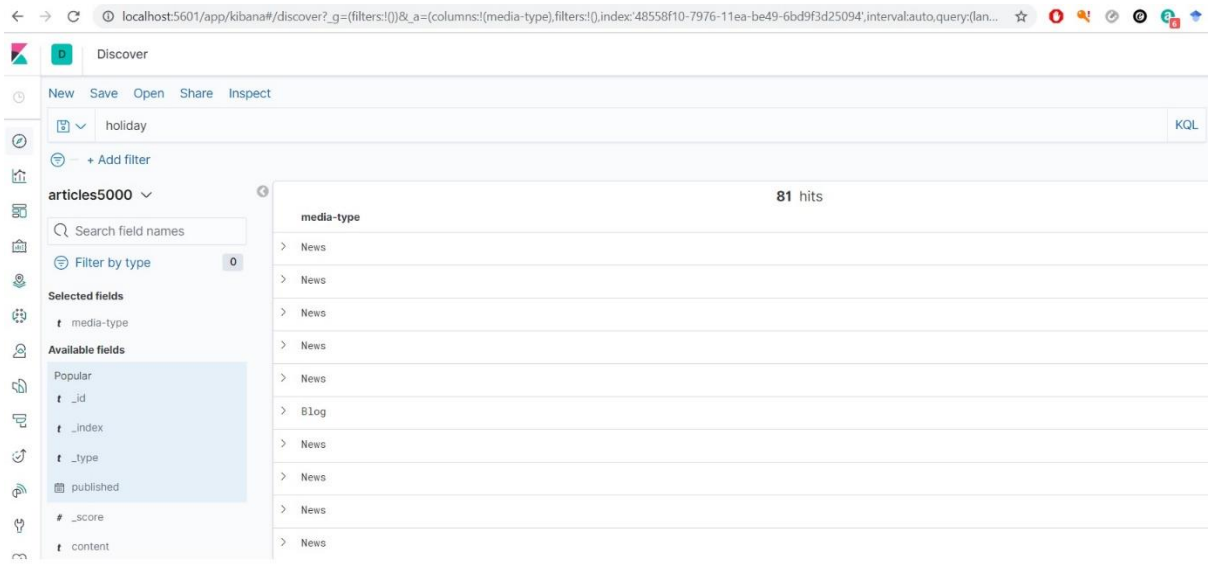


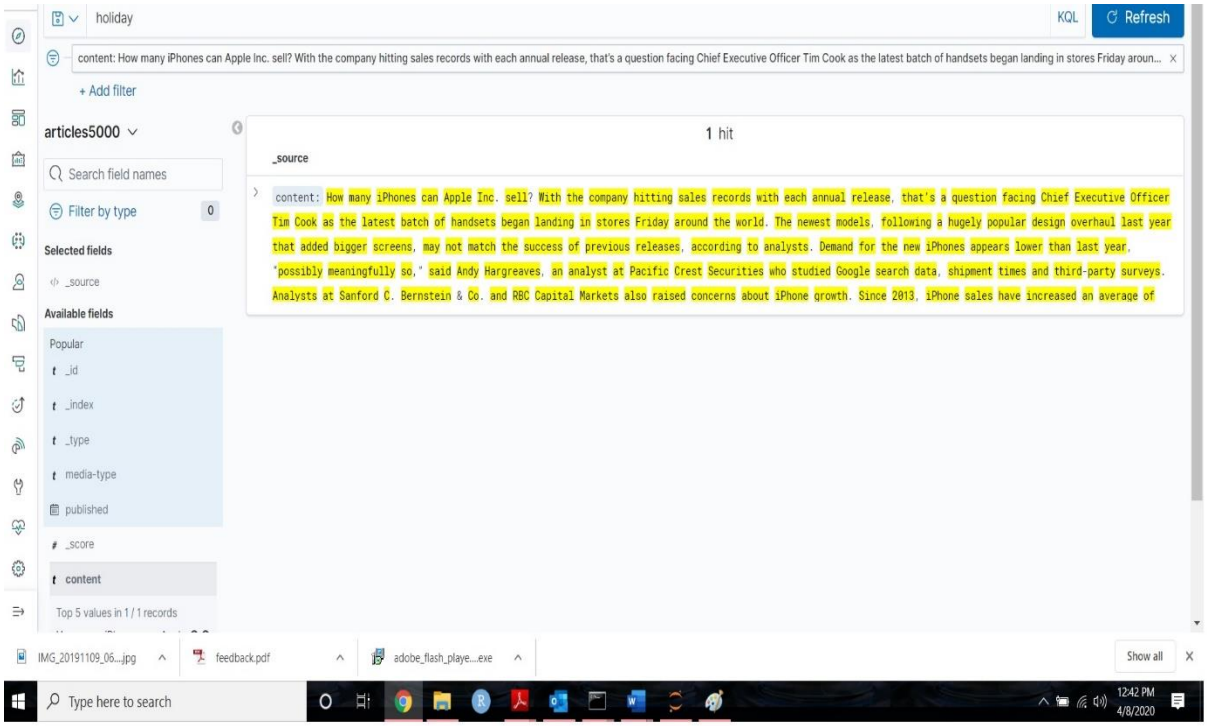


10. Holiday

The query on holiday gives 81 hits and on further adding a filter content on iPhone gave 1 hit. The kibana screenshots are as under –

`





J. EVALUATION AND QUERY RESULTS (RECALL AND PRECISION)

We evaluated our system using the parameters Recall and Precision.

Recall is defined by the following ratio –

Recall = Number of relevant documents returned / Total number of relevant documents

Precision is given as –

Precision = Number of relevant documents returned/ Total number of returned documents

The sample size of returned documents taken for evaluation in our assignment is K= 10.

The Test Collection consisted of ten queries relating to specific events together with their expected results which is tabulated as under –

EVALUATION RESULTS TABLE			
Index			
K = Number of documents taken as sample size for evaluation or calculation.K=10			
S.No	**Query**	**Recall**	**Precision**
1	Politics	8/12	7/10
2	Sports	3/4	3/10
3	Technology	5/10	5/10
4	Software Giants	2/4	2/10
5	Health	4/9	4/10
6	Love	3/7	3/10
7	Rain	2/6	2/10
8	Movie	1/8	1/10
9	Book	3/9	3/10
10	Holiday	5/7	5/10

K. CROWDSOURCING TASK AND ITS EXPERIENCE

There was a crowdsourcing task in the assignment 2 which consisted of 20 % marks of the total assignment and was independent in nature, that is, this crowd sourcing task could be done irrespective of implementing this Elastic Search system.

The task consisted of 2 phases. Each phase composed of small videos to be seen and the task was to hit spacebar if the video presented was a repeated one. Each video was of around 5 seconds length and the focus was on checking the recalling and memory of the participant.

We had to login using a link given in the mail using our University of Essex credentials and also in the settings of the google chrome browser Javascript was to be enabled in the site settings to view the videos of the task.

The videos were extremely funny and relaxing. We scored above 90 % in the first phase.

There was a gap of 24 – 72 hours between Phase 1 and 2. The password for those who completed the phase 1 was given as Registration number_DaY2. Again, the same task was to be performed. This phase was focussed on how much one retains after the gap of 1 or 2 days in recognising the videos are repeated or new.

This phase was also very nice with each one of us scoring above 80%. It shows that with time, the memory and recalling capacity shows a decrease.

L. ENGINEERING A COMPLETE SYSTEM

We successfully engineered a holistic elasticsearch system whose blueprint is displayed below –

1. Installing → JAVA environment
2. Installing → Elasticsearch using batch file on Command prompt(cmd). Then running elastic search on localhost 9200
3. Installing → Elasticsearch using batch file on Command prompt(cmd). Then running elastic search on localhost 5601
4. Installing → Jupyter and Anaconda for Python 3 environment required for the system
5. Signalmedia.jsonl Dataset → Loaded into elasticsearch localhost using Anaconda Jupyter Python Environment

6. Indexing → Implemented for 5000 articles from the dataset since the system was getting freeze on including complete 1 million documents of the dataset into the system. **[LIMITATION OF SYSTEM]**

7. Searching → Using Kibana Query Language (KQL) we executed various searches outlined in the test collection section of this report.

8. Evaluation → Using recall and precision parameters we evaluated the system individually for each query in the test collection and tabulated them properly.

M. SUGGESTIONS ON SCOPE AND FUTURE IMPROVEMENTS POSSIBLE –

Due to practical and runtime problems of too much system hanging and freezing, we limited our search and article size to 5000.

For more comprehensiveness, this can be increased further to 1 million documents and evaluation parameter K can be increased which was fixed in our study to 10 because huge manual calculations involved in recall and precision was making the evaluation very complex.

This can be solved using faster data processing Tensor Processing Units (TPUs).