CE706 - Information Retrieval 2021

Assignment 2

Registration Number: 2004351

Test collection (Task 1)

Information need	**Query**
Retrospective describes the epidemiology and clinical features patients of previous smallpox ring vaccination models based on contact in Saudi Arabia. Methods developed a simulation of smallpox transmission, vaccination, vaccination of contacts of contacts. The aim of this to develop methods for estimating reproduction which are simple and could be applied with data or in real time during an outbreak from January through December were identified through the records by Alexander in Nucleic Acids Res.	`{ "_source":["abstract", "title", "authors", "journal"], "size": 10, "query": { "bool": { "should": [ { "match": { "abstract": "smallpox patients" } }, { "match_phrase": { "abstract": "result of vaccination" } } ], "must_not" :{ "match": { "abstract": "time during an outbreak" } } } } }`
Infections caused by tuberculosis Respiratory syncytial virus (RSV) and pneumonia virus of mice (PVM). Supernatants were cells suppressed macrophage. After stimulation of the antibiotic-exposed CA-MRSA isolates the effect cells bacteria, ketamine was added selected individuals were interviewed.	`{ "_source": ["abstract","title","authors","journal"], "size": 10, "query": { "bool": { "filter": [ { "multi_match" :{`

A common technique used for sensitive and specific diagnostic virus detection in clinical samples is PCR that can identify one or several viruses by authors Leblanc, Jayati, Hine, Erin in BMC Genomics and Public Health.	"query": "Reasons of infections occured",
	"fields": "abstract^3",
	"minimum_should_match": "50%"
	}
	}
	],
	"should": [
	{
	"match":
	{
	"abstract": "clinical samples"
	}
	}
	]
	}
	}
	}

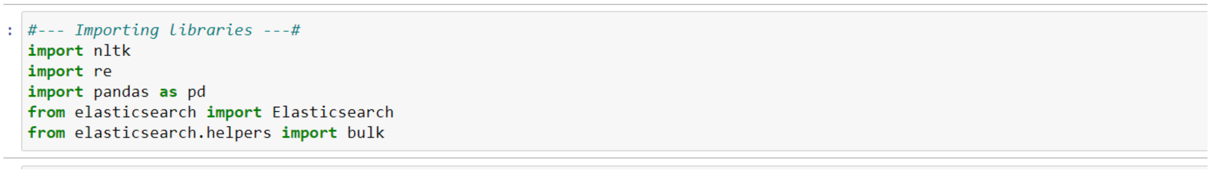
Foot-and-mouth disease (FMD) is one of the highly contagious diseases. The spread of infectious disease epidemics is mediated by human travel. In each of area 10,000 selected in Anhui. Travel behavior was examined as of age, sex, status and home location. This study aimed developed and economically middle-income countries populations were by kernel function. Authors Alex, Honda, Amin and Campanella were published in journal named PLoS One	{
	"_source":["title", "abstract", "authors", "journal"],
	"size": 10,
	"query": {
	"query_string": {
	"query": "infectious disease AND countries population OR human travel"
	}
	},
	"highlight": {
	"fields" : {
	"abstract": {}
	}
	}
	}

IR systems (Task 2)

- ❖ I have created two different Information Retrieval system in which both systems contain different pre-processing steps.

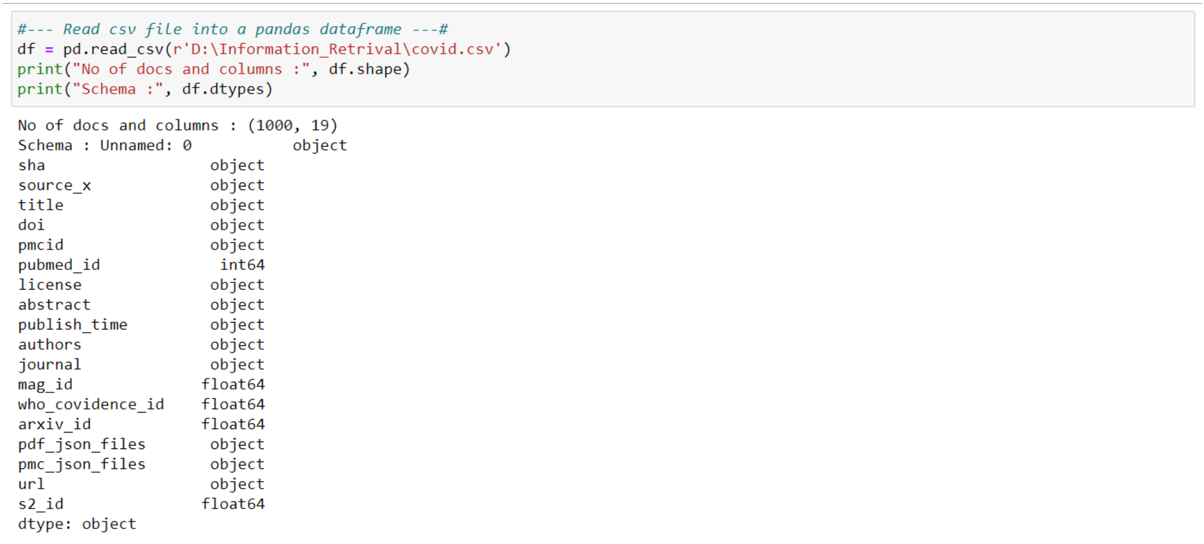
- ❖ To perform different operations on both Information Retrieval system one should follow below basic steps:

1. Import all necessary libraries:



```
#--- Importing Libraries ---#
import nltk
import re
import pandas as pd
from elasticsearch import Elasticsearch
from elasticsearch.helpers import bulk
```

2. Download the dataset from Kaggle website COVID-19 Open Research Dataset Challenge (CORD-19) | Kaggle, here I considered only first 1000 documents for both the system, and read the data using pandas data frame.



```
#--- Read csv file into a pandas dataframe ---#
df = pd.read_csv(r'D:\Information_Retrival\covid.csv')
print("No of docs and columns :", df.shape)
print("Schema :", df.dtypes)
```

```
No of docs and columns : (1000, 19)
Schema : Unnamed: 0           object
sha                    object
source_x               object
title                  object
doi                    object
pmcid                  object
pubmed_id               int64
license                object
abstract               object
publish_time           object
authors                object
journal                object
mag_id                float64
who_covidence_id      float64
arxiv_id              float64
pdf_json_files         object
pmc_json_files         object
url                    object
s2_id                 float64
dtype: object
```

After loading the data into data frame, I performed some preprocessing steps in both systems to get better relevant result from query.

❖ IR_System_1 :

In **IR_System_1**, I have done preprocessing steps which includes Sentence Splitting, Tokenization and Normalization.

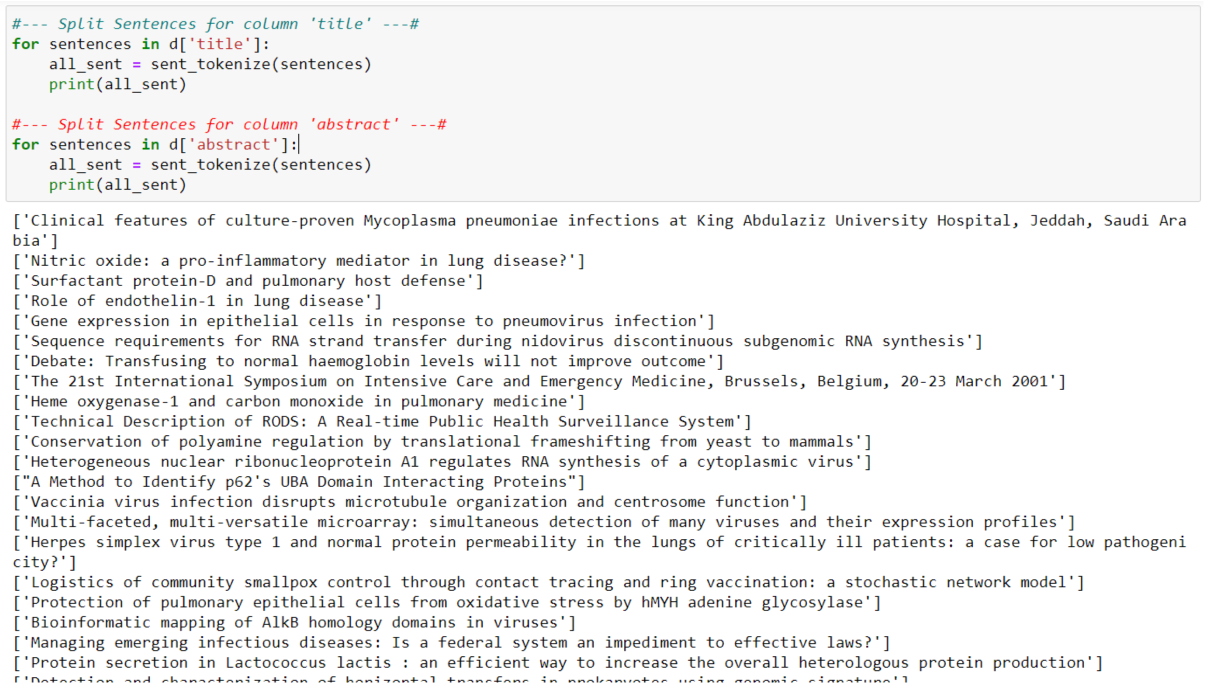
❖ Splitting:

Sentence Splitting :

I have considered two columns for sentences splitting are **"title"** and **"abstract",** We have to import **"sent_tokenize"** package to split the sentences.

```
#--- Import sent_tokenize package from NLTK Library ---#
from nltk.tokenize import sent_tokenize
```

Following is the result I retrieved from the both columns in which whole paragraphs splits into multiple sentences.

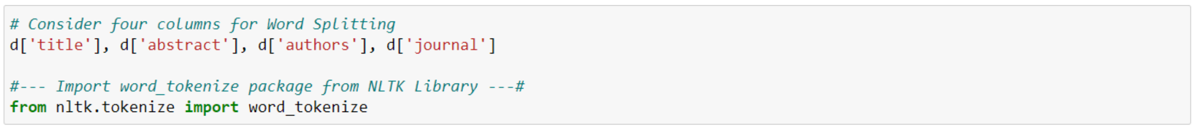


```
#--- Split Sentences for column 'title' ---#
for sentences in d['title']:
    all_sent = sent_tokenize(sentences)
    print(all_sent)

#--- Split Sentences for column 'abstract' ---#
for sentences in d['abstract']:
    all_sent = sent_tokenize(sentences)
    print(all_sent)
```

```
['Clinical features of culture-proven Mycoplasma pneumoniae infections at King Abdulaziz University Hospital, Jeddah, Saudi Ara
bia']
['Nitric oxide: a pro-inflammatory mediator in lung disease?']
['Surfactant protein-D and pulmonary host defense']
['Role of endothelin-1 in lung disease']
['Gene expression in epithelial cells in response to pneumovirus infection']
['Sequence requirements for RNA strand transfer during nidovirus discontinuous subgenomic RNA synthesis']
['Debate: Transfusing to normal haemoglobin levels will not improve outcome']
['The 21st International Symposium on Intensive Care and Emergency Medicine, Brussels, Belgium, 20-23 March 2001']
['Heme oxygenase-1 and carbon monoxide in pulmonary medicine']
['Technical Description of RODS: A Real-time Public Health Surveillance System']
['Conservation of polyamine regulation by translational frameshifting from yeast to mammals']
['Heterogeneous nuclear ribonucleoprotein A1 regulates RNA synthesis of a cytoplasmic virus']
["A Method to Identify p62's UBA Domain Interacting Proteins"]
['Vaccinia virus infection disrupts microtubule organization and centrosome function']
['Multi-faceted, multi-versatile microarray: simultaneous detection of many viruses and their expression profiles']
['Herpes simplex virus type 1 and normal protein permeability in the lungs of critically ill patients: a case for low pathogeni
city?']
['Logistics of community smallpox control through contact tracing and ring vaccination: a stochastic network model']
['Protection of pulmonary epithelial cells from oxidative stress by hMYH adenine glycosylase']
['Bioinformatic mapping of AlkB homology domains in viruses']
['Managing emerging infectious diseases: Is a federal system an impediment to effective laws?']
['Protein secretion in Lactococcus lactis : an efficient way to increase the overall heterologous protein production']
['Detection and characterization of horizontal transfers in prokaryotes using genomic signatures']
```

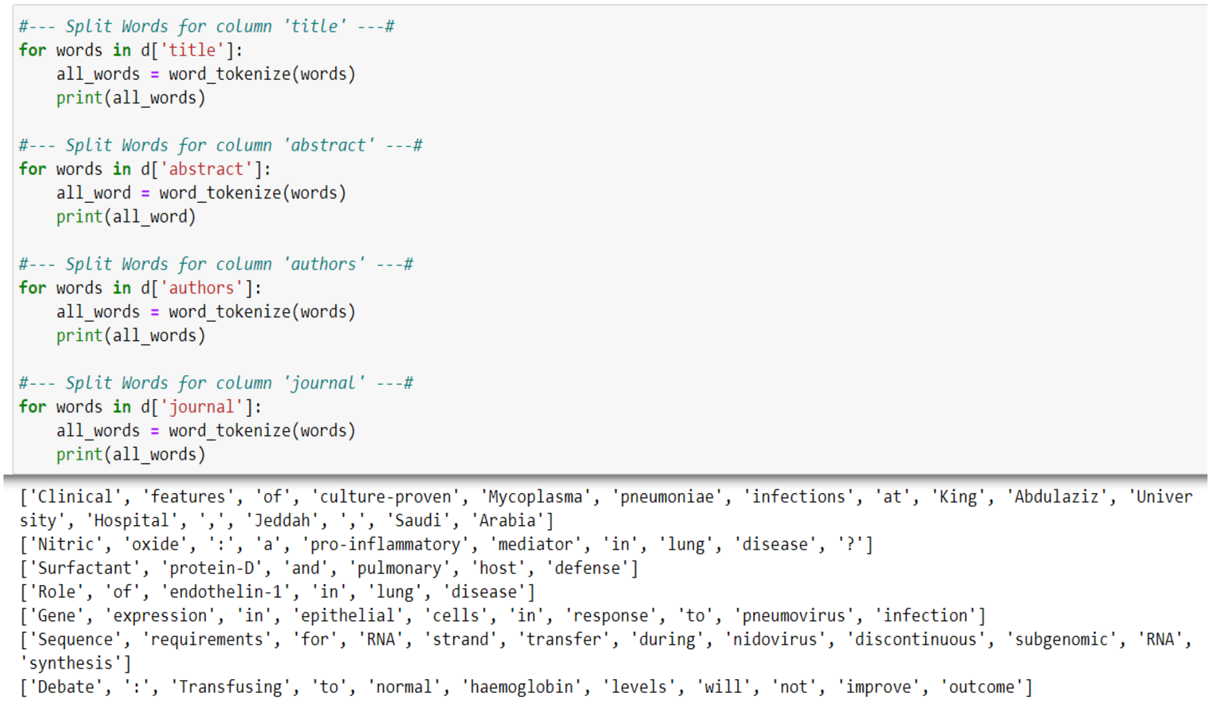
Word Splitting :

I have considered four columns for **word splitting** and for splitting the words one need to download the **“word_tokenize”** package.



```
# Consider four columns for Word Splitting
d['title'], d['abstract'], d['authors'], d['journal']

#--- Import word_tokenize package from NLTK Library ---#
from nltk.tokenize import word_tokenize
```

Here all the four columns data is splitted into multiple words.

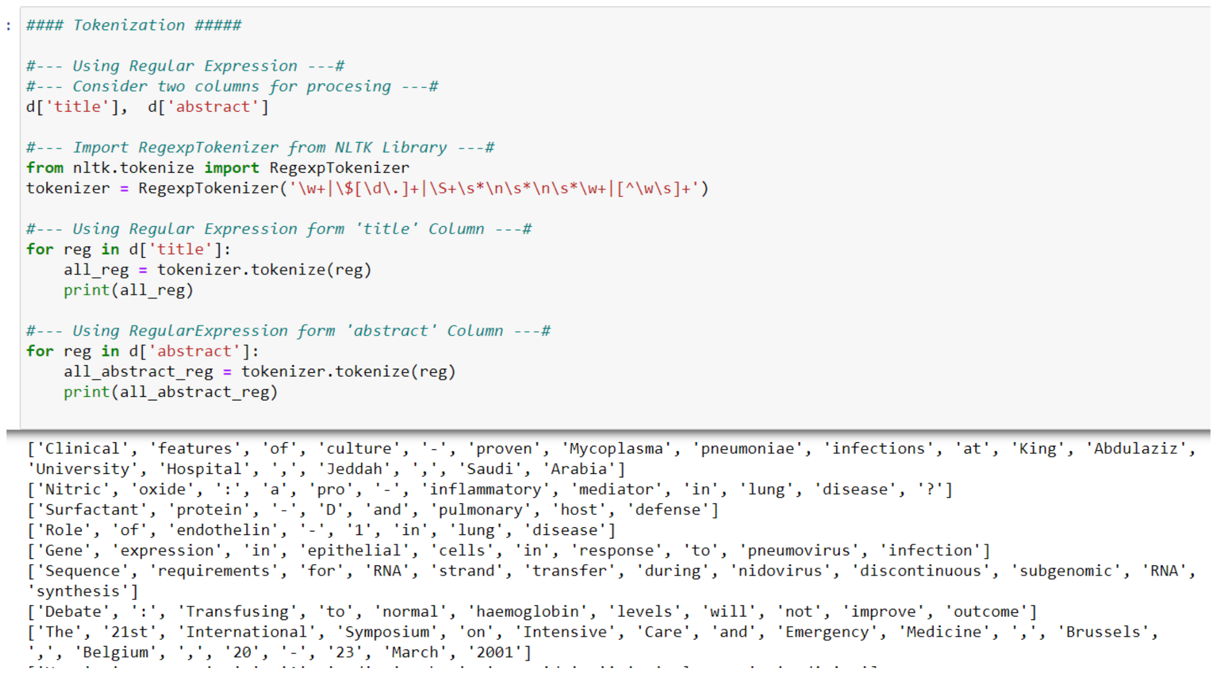


```
#--- Split Words for column 'title' ---#
for words in d['title']:
    all_words = word_tokenize(words)
    print(all_words)

#--- Split Words for column 'abstract' ---#
for words in d['abstract']:
    all_word = word_tokenize(words)
    print(all_word)

#--- Split Words for column 'authors' ---#
for words in d['authors']:
    all_words = word_tokenize(words)
    print(all_words)

#--- Split Words for column 'journal' ---#
for words in d['journal']:
    all_words = word_tokenize(words)
    print(all_words)
```

```
['Clinical', 'features', 'of', 'culture-proven', 'Mycoplasma', 'pneumoniae', 'infections', 'at', 'King', 'Abdulaziz', 'Univer
sity', 'Hospital', ',', 'Jeddah', ',', 'Saudi', 'Arabia']
['Nitric', 'oxide', ':', 'a', 'pro-inflammatory', 'mediator', 'in', 'lung', 'disease', '?']
['Surfactant', 'protein-D', 'and', 'pulmonary', 'host', 'defense']
['Role', 'of', 'endothelin-1', 'in', 'lung', 'disease']
['Gene', 'expression', 'in', 'epithelial', 'cells', 'in', 'response', 'to', 'pneumovirus', 'infection']
['Sequence', 'requirements', 'for', 'RNA', 'strand', 'transfer', 'during', 'nidovirus', 'discontinuous', 'subgenomic', 'RNA',
'synthesis']
['Debate', ':', 'Transfusing', 'to', 'normal', 'haemoglobin', 'levels', 'will', 'not', 'improve', 'outcome']
```

❖ Tokenization:

Tokenization is used for divide the whole data into different chunks of small small tokens. I have considered two columns **“title”** and **“abstract”** to divide the sentences into tokens.

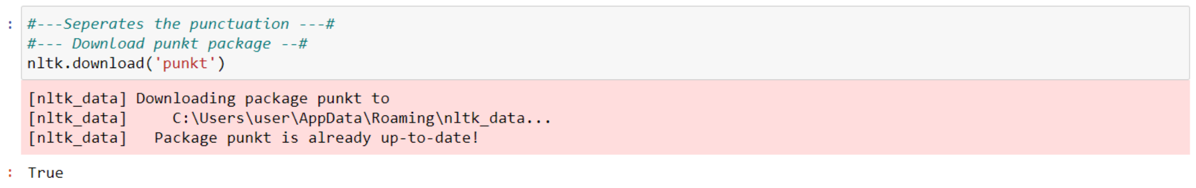


```
#### Tokenization #####

#--- Using Regular Expression ---#
#--- Consider two columns for procesing ---#
d['title'],  d['abstract']

#--- Import RegexpTokenizer from NLTK Library ---#
from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer('\w+|\$[\d\.]+|\S+\s*\n\s*\n\s*\w+|[^\w\s]+')

#--- Using Regular Expression form 'title' Column ---#
for reg in d['title']:
    all_reg = tokenizer.tokenize(reg)
    print(all_reg)

#--- Using RegularExpression form 'abstract' Column ---#
for reg in d['abstract']:
    all_abstract_reg = tokenizer.tokenize(reg)
    print(all_abstract_reg)
```

```
['Clinical', 'features', 'of', 'culture', '-', 'proven', 'Mycoplasma', 'pneumoniae', 'infections', 'at', 'King', 'Abdulaziz',
'University', 'Hospital', ',', 'Jeddah', ',', 'Saudi', 'Arabia']
['Nitric', 'oxide', ':', 'a', 'pro', '-', 'inflammatory', 'mediator', 'in', 'lung', 'disease', '?']
['Surfactant', 'protein', '-', 'D', 'and', 'pulmonary', 'host', 'defense']
['Role', 'of', 'endothelin', '-', '1', 'in', 'lung', 'disease']
['Gene', 'expression', 'in', 'epithelial', 'cells', 'in', 'response', 'to', 'pneumovirus', 'infection']
['Sequence', 'requirements', 'for', 'RNA', 'strand', 'transfer', 'during', 'nidovirus', 'discontinuous', 'subgenomic', 'RNA',
'synthesis']
['Debate', ':', 'Transfusing', 'to', 'normal', 'haemoglobin', 'levels', 'will', 'not', 'improve', 'outcome']
['The', '21st', 'International', 'Symposium', 'on', 'Intensive', 'Care', 'and', 'Emergency', 'Medicine', ',', 'Brussels',
',', 'Belgium', ',', '20', '-', '23', 'March', '2001']
```

❖ Remove Punctuation:

Download the package **“punkt”** for remove punctuation marks from the given data.



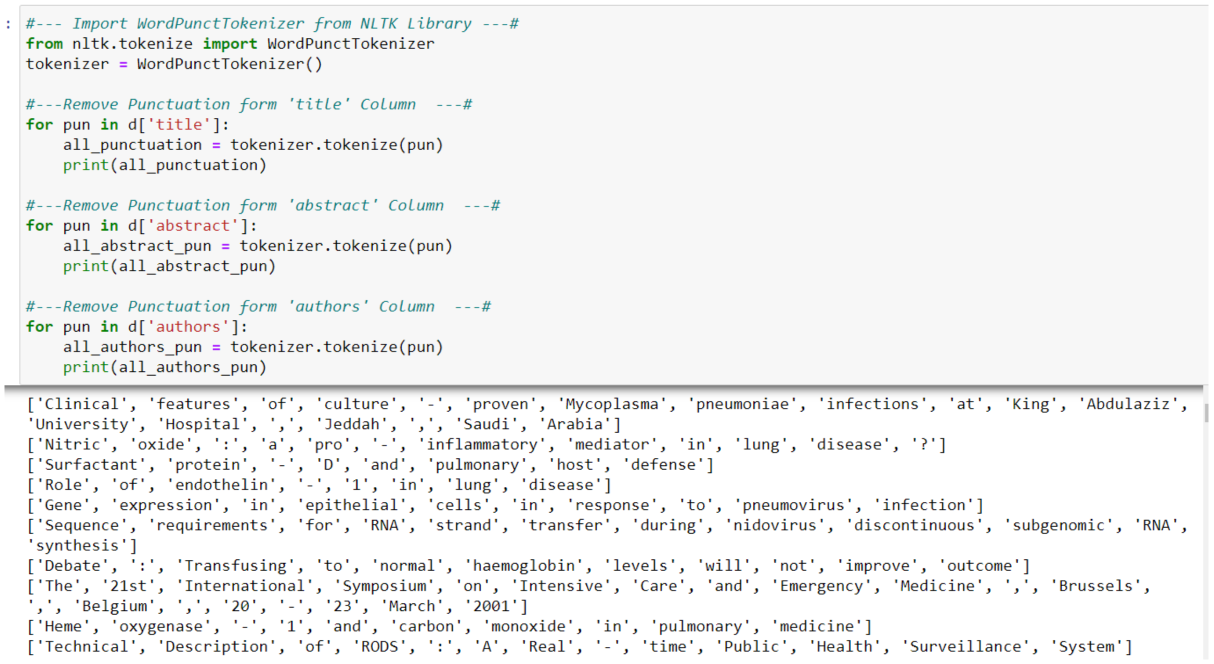
```
#---Seperates the punctuation ---#
#--- Download punkt package --#
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\user\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

```
True
```

Import **“WordPunctTokenizer”** library to remove punctuation from the four columns, “title”, “abstract”, “authors” and “journal”.

❖ Following is the retrieved result from the data.

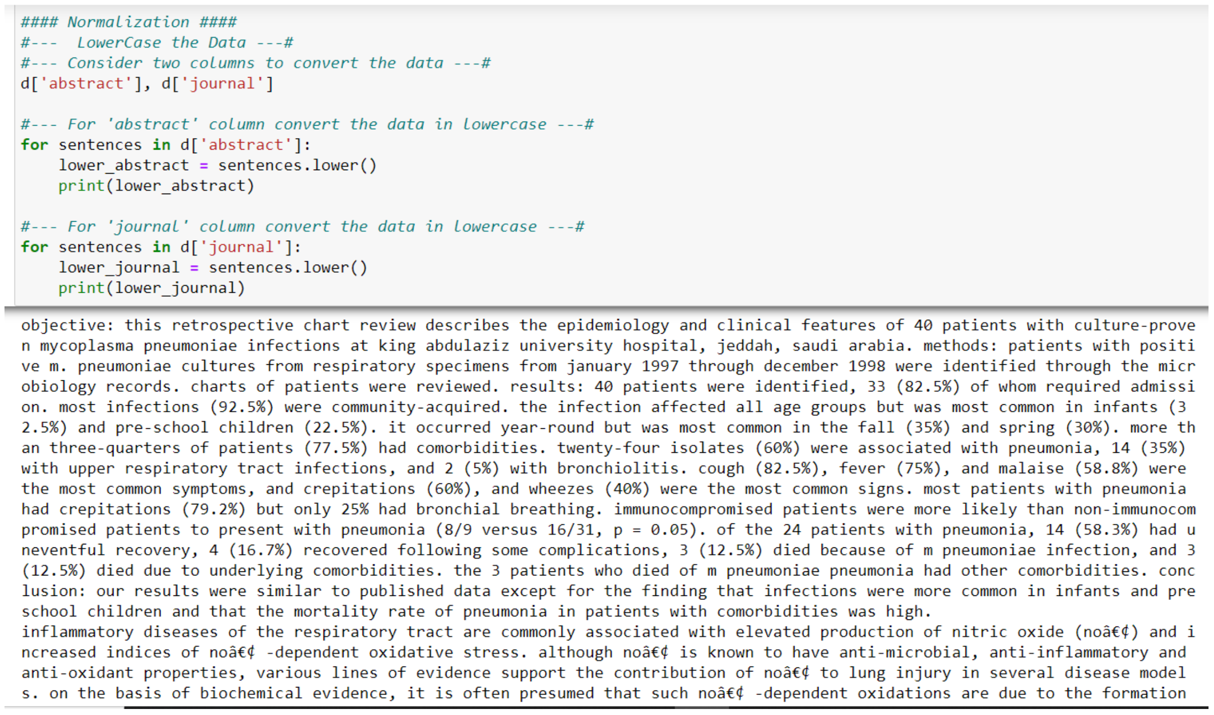


```
#--- Import WordPunctTokenizer from NLTK Library ---#
from nltk.tokenize import WordPunctTokenizer
tokenizer = WordPunctTokenizer()

#---Remove Punctuation form 'title' Column  ---#
for pun in d['title']:
    all_punctuation = tokenizer.tokenize(pun)
    print(all_punctuation)

#---Remove Punctuation form 'abstract' Column  ---#
for pun in d['abstract']:
    all_abstract_pun = tokenizer.tokenize(pun)
    print(all_abstract_pun)

#---Remove Punctuation form 'authors' Column  ---#
for pun in d['authors']:
    all_authors_pun = tokenizer.tokenize(pun)
    print(all_authors_pun)
```

```
['Clinical', 'features', 'of', 'culture', '-', 'proven', 'Mycoplasma', 'pneumoniae', 'infections', 'at', 'King', 'Abdulaziz',
'University', 'Hospital', ',', 'Jeddah', ',', 'Saudi', 'Arabia']
['Nitric', 'oxide', ':', 'a', 'pro', '-', 'inflammatory', 'mediator', 'in', 'lung', 'disease', '?']
['Surfactant', 'protein', '-', 'D', 'and', 'pulmonary', 'host', 'defense']
['Role', 'of', 'endothelin', '-', '1', 'in', 'lung', 'disease']
['Gene', 'expression', 'in', 'epithelial', 'cells', 'in', 'response', 'to', 'pneumovirus', 'infection']
['Sequence', 'requirements', 'for', 'RNA', 'strand', 'transfer', 'during', 'nidovirus', 'discontinuous', 'subgenomic', 'RNA',
'synthesis']
['Debate', ':', 'Transfusing', 'to', 'normal', 'haemoglobin', 'levels', 'will', 'not', 'improve', 'outcome']
['The', '21st', 'International', 'Symposium', 'on', 'Intensive', 'Care', 'and', 'Emergency', 'Medicine', ',', 'Brussels',
',', 'Belgium', ',', '20', '-', '23', 'March', '2001']
['Heme', 'oxygenase', '-', '1', 'and', 'carbon', 'monoxide', 'in', 'pulmonary', 'medicine']
['Technical', 'Description', 'of', 'RODS', ':', 'A', 'Real', '-', 'time', 'Public', 'Health', 'Surveillance', 'System']
```

❖ Normalization:

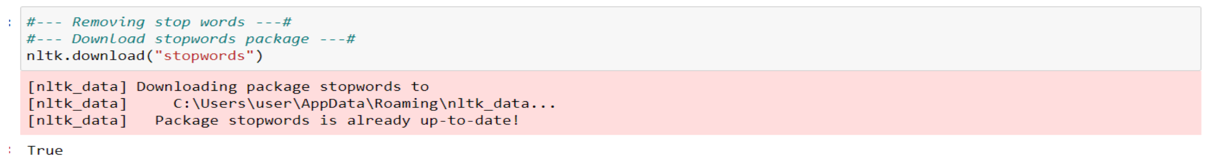
I have considered two columns for Normalization **“abstract”** and **“journal”**



```
#### Normalization ####
#---  LowerCase the Data ---#
#--- Consider two columns to convert the data ---#
d['abstract'], d['journal']

#--- For 'abstract' column convert the data in lowercase ---#
for sentences in d['abstract']:
    lower_abstract = sentences.lower()
    print(lower_abstract)

#--- For 'journal' column convert the data in lowercase ---#
for sentences in d['journal']:
    lower_journal = sentences.lower()
    print(lower_journal)
```

```
objective: this retrospective chart review describes the epidemiology and clinical features of 40 patients with culture-prove
n mycoplasma pneumoniae infections at king abdulaziz university hospital, jeddah, saudi arabia. methods: patients with positi
ve m. pneumoniae cultures from respiratory specimens from january 1997 through december 1998 were identified through the micr
obiology records. charts of patients were reviewed. results: 40 patients were identified, 33 (82.5%) of whom required admissi
on. most infections (92.5%) were community-acquired. the infection affected all age groups but was most common in infants (3
2.5%) and pre-school children (22.5%). it occurred year-round but was most common in the fall (35%) and spring (30%). more th
an three-quarters of patients (77.5%) had comorbidities. twenty-four isolates (60%) were associated with pneumonia, 14 (35%)
with upper respiratory tract infections, and 2 (5%) with bronchiolitis. cough (82.5%), fever (75%), and malaise (58.8%) were
the most common symptoms, and crepitations (60%), and wheezes (40%) were the most common signs. most patients with pneumonia
had crepitations (79.2%) but only 25% had bronchial breathing. immunocompromised patients were more likely than non-immunocom
promised patients to present with pneumonia (8/9 versus 16/31, p = 0.05). of the 24 patients with pneumonia, 14 (58.3%) had u
neventful recovery, 4 (16.7%) recovered following some complications, 3 (12.5%) died because of m pneumoniae infection, and 3
(12.5%) died due to underlying comorbidities. the 3 patients who died of m pneumoniae pneumonia had other comorbidities. conc
lusion: our results were similar to published data except for the finding that infections were more common in infants and pre
school children and that the mortality rate of pneumonia in patients with comorbidities was high.
inflammatory diseases of the respiratory tract are commonly associated with elevated production of nitric oxide (noâ€¢) and i
ncreased indices of noâ€¢ -dependent oxidative stress. although noâ€¢ is known to have anti-microbial, anti-inflammatory and
anti-oxidant properties, various lines of evidence support the contribution of noâ€¢ to lung injury in several disease model
s. on the basis of biochemical evidence, it is often presumed that such noâ€¢ -dependent oxidations are due to the formation
```

❖ Removing StopWords:

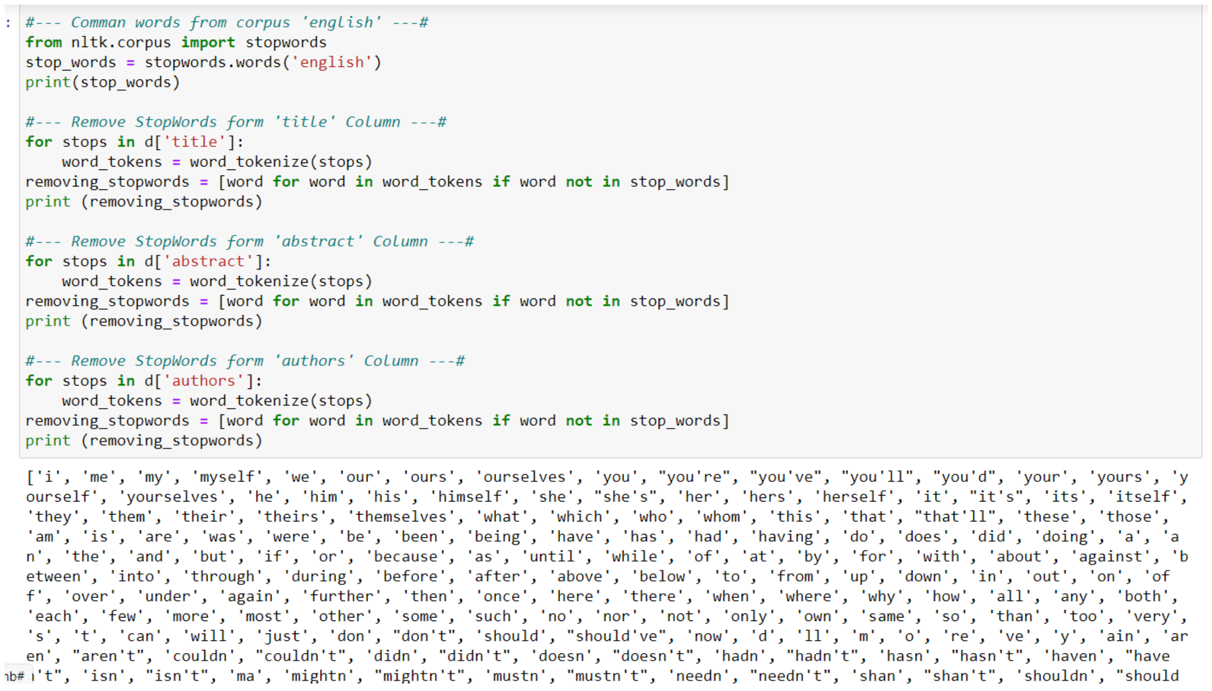


```
#--- Removing stop words ---#
#--- Download stopwords package ---#
nltk.download("stopwords")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\user\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

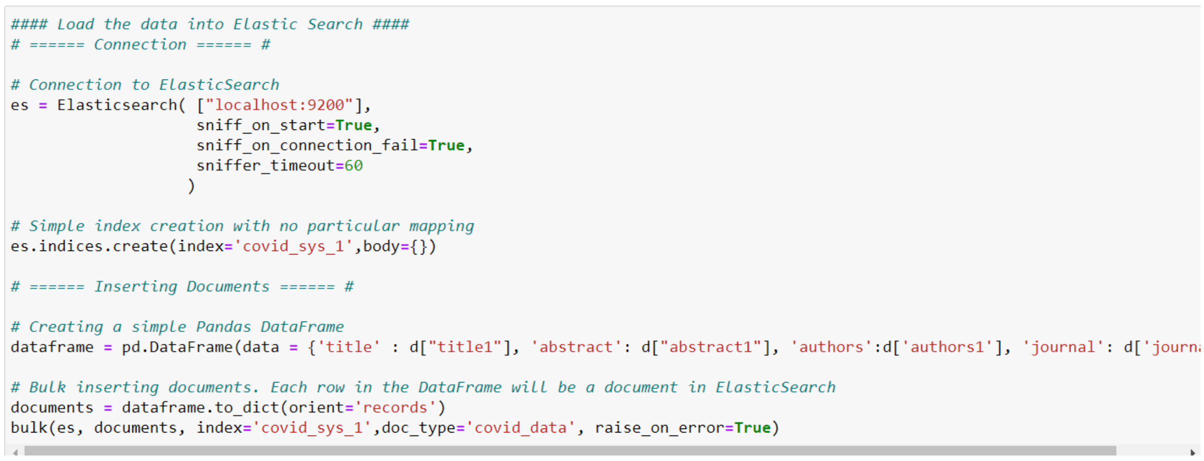
To remove stop words from paragraph one should download the package **“stopwords”,** Here I have considered three columns to remove stopwords, **“title”, “abstract”** and **“authors”.**



```
#--- Comman words from corpus 'english' ---#
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(stop_words)

#--- Remove StopWords form 'title' Column ---#
for stops in d['title']:
    word_tokens = word_tokenize(stops)
removing_stopwords = [word for word in word_tokens if word not in stop_words]
print (removing_stopwords)

#--- Remove StopWords form 'abstract' Column ---#
for stops in d['abstract']:
    word_tokens = word_tokenize(stops)
removing_stopwords = [word for word in word_tokens if word not in stop_words]
print (removing_stopwords)

#--- Remove StopWords form 'authors' Column ---#
for stops in d['authors']:
    word_tokens = word_tokenize(stops)
removing_stopwords = [word for word in word_tokens if word not in stop_words]
print (removing_stopwords)
```

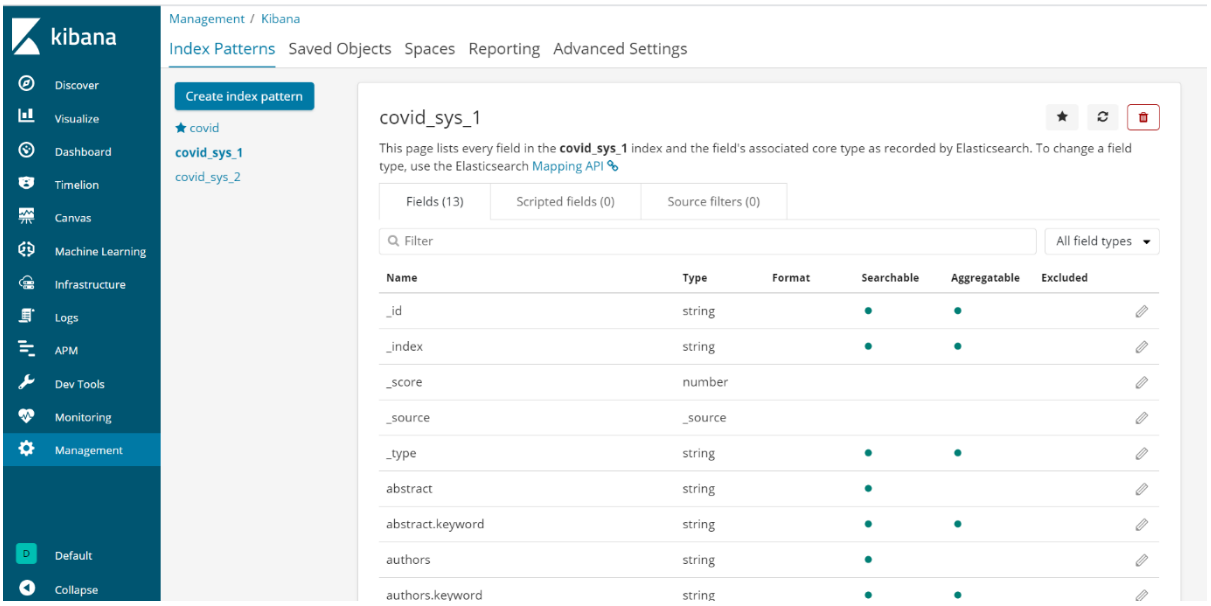
```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'y
ourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself',
'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those',
'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'a
n', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'b
etween', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'of
f', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both',
'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'ar
en', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "have
't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "should
```

After all these pre-processing steps I have created index named **“covid_sys_1”.**



```
#### Load the data into Elastic Search ####
# ====== Connection ====== #

# Connection to ElasticSearch
es = Elasticsearch( ["localhost:9200"],
                    sniff_on_start=True,
                    sniff_on_connection_fail=True,
                    sniffer_timeout=60
                  )

# Simple index creation with no particular mapping
es.indices.create(index='covid_sys_1',body={})

# ====== Inserting Documents ====== #

# Creating a simple Pandas DataFrame
dataframe = pd.DataFrame(data = {'title' : d["title1"], 'abstract': d["abstract1"], 'authors':d['authors1'], 'journal': d['journ

# Bulk inserting documents. Each row in the DataFrame will be a document in ElasticSearch
documents = dataframe.to_dict(orient='records')
bulk(es, documents, index='covid_sys_1',doc_type='covid_data', raise_on_error=True)
```

In Kibana, to create index go to the **Management > Index Pattern > covid_sys_1**.



covid_sys_1

This page lists every field in the **covid_sys_1** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch Mapping API

Name	Type	Format	Searchable	Aggregatable	Excluded
_id	string		●	●	
_index	string		●	●	
_score	number				
_source	_source				
_type	string		●	●	
abstract	string		●		
abstract.keyword	string		●	●	
authors	string		●		
authors.keyword	string		●	●	

❖ <u>IR_System_2 :</u>

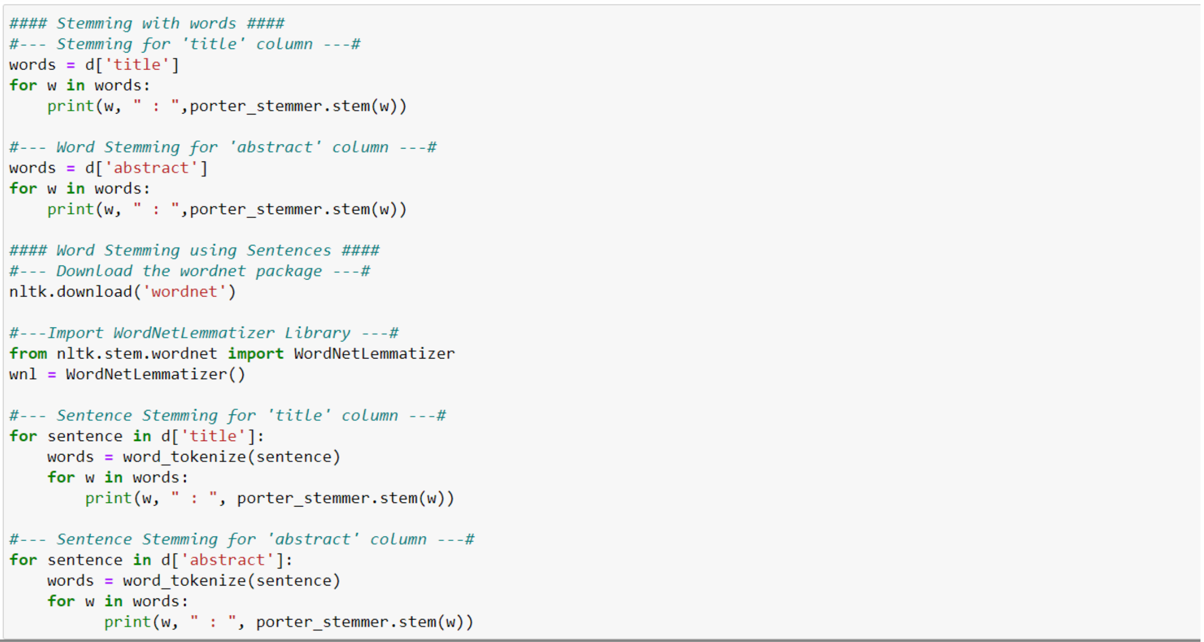
In IR_System_2, I have done pre-processing steps like Stemming and Morphological Analysis and selecting keywords.

❖ Stemming:

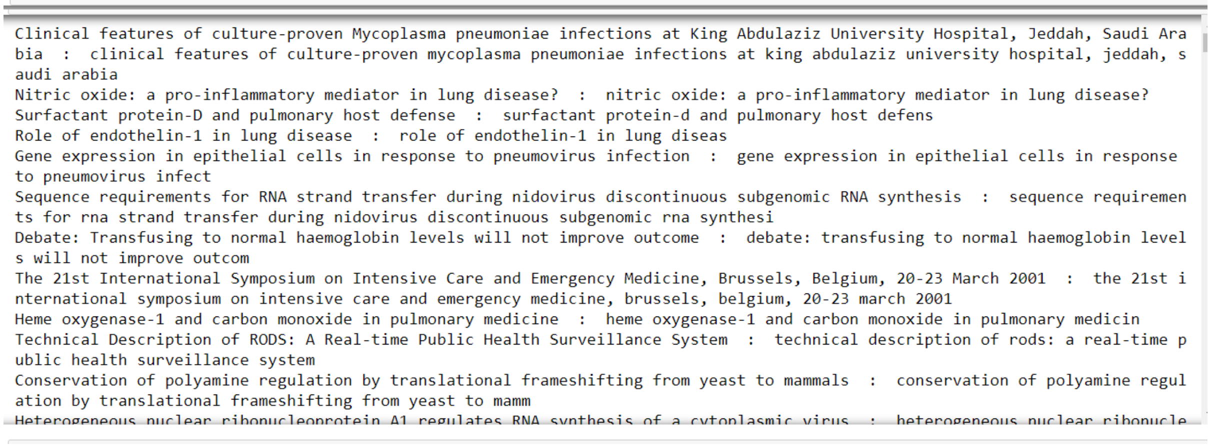
For Stemming one should import library called **“PorterStemmer”**

```
#### Stemming or Morphological Analysis ####

#--- Import Stemming Libraries ---#
from nltk.stem.porter import PorterStemmer
from nltk.tokenize import word_tokenize
porter_stemmer = PorterStemmer()
```

I have applied the stemming on two columns **“title”, “abstract”.**



```
#### Stemming with words ####
#--- Stemming for 'title' column ---#
words = d['title']
for w in words:
    print(w, " : ",porter_stemmer.stem(w))

#--- Word Stemming for 'abstract' column ---#
words = d['abstract']
for w in words:
    print(w, " : ",porter_stemmer.stem(w))

#### Word Stemming using Sentences ####
#--- Download the wordnet package ---#
nltk.download('wordnet')

#---Import WordNetLemmatizer Library ---#
from nltk.stem.wordnet import WordNetLemmatizer
wnl = WordNetLemmatizer()

#--- Sentence Stemming for 'title' column ---#
for sentence in d['title']:
    words = word_tokenize(sentence)
    for w in words:
        print(w, " : ", porter_stemmer.stem(w))

#--- Sentence Stemming for 'abstract' column ---#
for sentence in d['abstract']:
    words = word_tokenize(sentence)
    for w in words:
        print(w, " : ", porter_stemmer.stem(w))
```

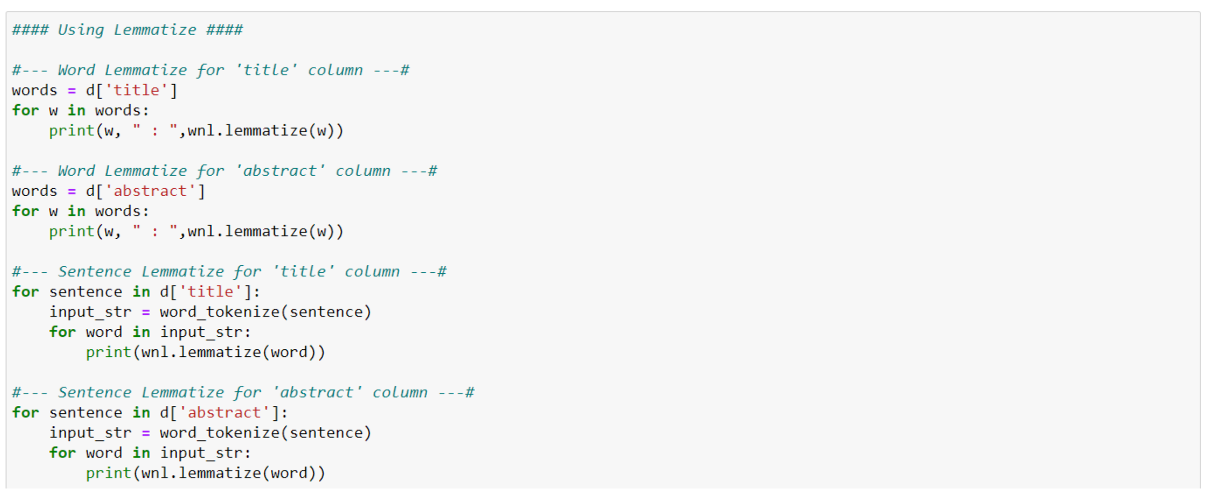


```
Clinical features of culture-proven Mycoplasma pneumoniae infections at King Abdulaziz University Hospital, Jeddah, Saudi Ara
bia  :  clinical features of culture-proven mycoplasma pneumoniae infections at king abdulaziz university hospital, jeddah, s
audi arabia
Nitric oxide: a pro-inflammatory mediator in lung disease?  :  nitric oxide: a pro-inflammatory mediator in lung disease?
Surfactant protein-D and pulmonary host defense  :  surfactant protein-d and pulmonary host defens
Role of endothelin-1 in lung disease  :  role of endothelin-1 in lung diseas
Gene expression in epithelial cells in response to pneumovirus infection  :  gene expression in epithelial cells in response
to pneumovirus infect
Sequence requirements for RNA strand transfer during nidovirus discontinuous subgenomic RNA synthesis  :  sequence requiremen
ts for rna strand transfer during nidovirus discontinuous subgenomic rna synthesi
Debate: Transfusing to normal haemoglobin levels will not improve outcome  :  debate: transfusing to normal haemoglobin level
s will not improve outcom
The 21st International Symposium on Intensive Care and Emergency Medicine, Brussels, Belgium, 20-23 March 2001  :  the 21st i
nternational symposium on intensive care and emergency medicine, brussels, belgium, 20-23 march 2001
Heme oxygenase-1 and carbon monoxide in pulmonary medicine  :  heme oxygenase-1 and carbon monoxide in pulmonary medicin
Technical Description of RODS: A Real-time Public Health Surveillance System  :  technical description of rods: a real-time p
ublic health surveillance system
Conservation of polyamine regulation by translational frameshifting from yeast to mammals  :  conservation of polyamine regul
ation by translational frameshifting from yeast to mamm
Heterogeneous nuclear ribonucleoprotein A1 regulates RNA synthesis of a cytoplasmic virus  :  heterogeneous nuclear ribonucle
```

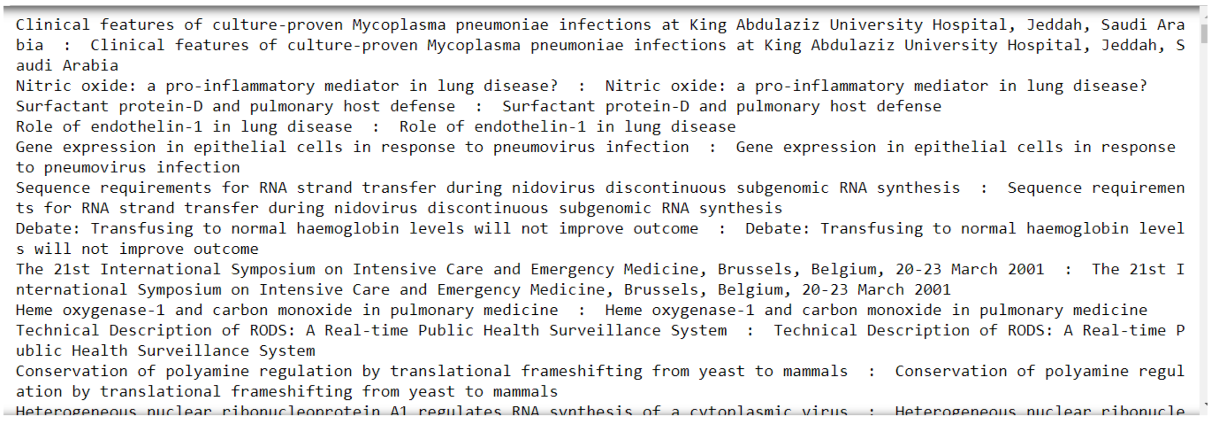
❖ Lemmatize:

Import **“WordNetLemmatizer”** library from the nltk package to use lemmatize

I have applied both word and sentences lemmatize on **“title”** and **“abstract”** columns.



```
#### Using Lemmatize ####

#--- Word Lemmatize for 'title' column ---#
words = d['title']
for w in words:
    print(w, " : ",wnl.lemmatize(w))

#--- Word Lemmatize for 'abstract' column ---#
words = d['abstract']
for w in words:
    print(w, " : ",wnl.lemmatize(w))

#--- Sentence Lemmatize for 'title' column ---#
for sentence in d['title']:
    input_str = word_tokenize(sentence)
    for word in input_str:
        print(wnl.lemmatize(word))

#--- Sentence Lemmatize for 'abstract' column ---#
for sentence in d['abstract']:
    input_str = word_tokenize(sentence)
    for word in input_str:
        print(wnl.lemmatize(word))
```

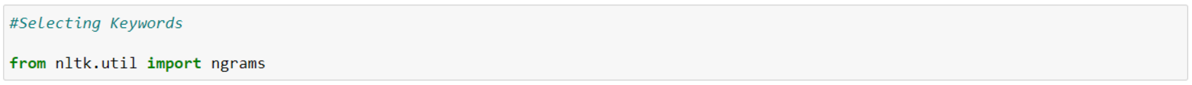


```
Clinical features of culture-proven Mycoplasma pneumoniae infections at King Abdulaziz University Hospital, Jeddah, Saudi Ara
bia  :  Clinical features of culture-proven Mycoplasma pneumoniae infections at King Abdulaziz University Hospital, Jeddah, S
audi Arabia
Nitric oxide: a pro-inflammatory mediator in lung disease?  :  Nitric oxide: a pro-inflammatory mediator in lung disease?
Surfactant protein-D and pulmonary host defense  :  Surfactant protein-D and pulmonary host defense
Role of endothelin-1 in lung disease  :  Role of endothelin-1 in lung disease
Gene expression in epithelial cells in response to pneumovirus infection  :  Gene expression in epithelial cells in response
to pneumovirus infection
Sequence requirements for RNA strand transfer during nidovirus discontinuous subgenomic RNA synthesis  :  Sequence requiremen
ts for RNA strand transfer during nidovirus discontinuous subgenomic RNA synthesis
Debate: Transfusing to normal haemoglobin levels will not improve outcome  :  Debate: Transfusing to normal haemoglobin level
s will not improve outcome
The 21st International Symposium on Intensive Care and Emergency Medicine, Brussels, Belgium, 20-23 March 2001  :  The 21st I
nternational Symposium on Intensive Care and Emergency Medicine, Brussels, Belgium, 20-23 March 2001
Heme oxygenase-1 and carbon monoxide in pulmonary medicine  :  Heme oxygenase-1 and carbon monoxide in pulmonary medicine
Technical Description of RODS: A Real-time Public Health Surveillance System  :  Technical Description of RODS: A Real-time P
ublic Health Surveillance System
Conservation of polyamine regulation by translational frameshifting from yeast to mammals  :  Conservation of polyamine regul
ation by translational frameshifting from yeast to mammals
Heterogeneous nuclear ribonucleoprotein A1 regulates RNA synthesis of a cytoplasmic virus  :  Heterogeneous nuclear ribonucle
```

❖ Selecting Keywords:

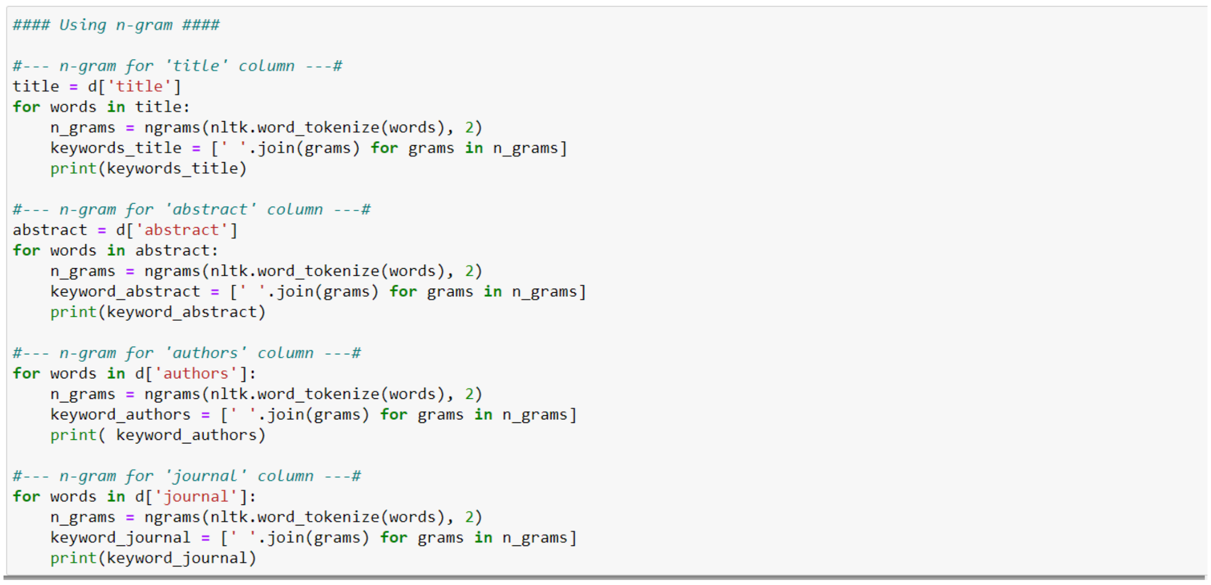
I have used **n-gram** to select keywords from the given corpus, and I have applied n-gram to the all four columns.

- ❖ Import n-gram library from nltk package.



```python
#Selecting Keywords

from nltk.util import ngrams
```

Using n-grams we can divide our data into number of word like if we pass number equal to 2 then the whole statement will divide in group of two words.



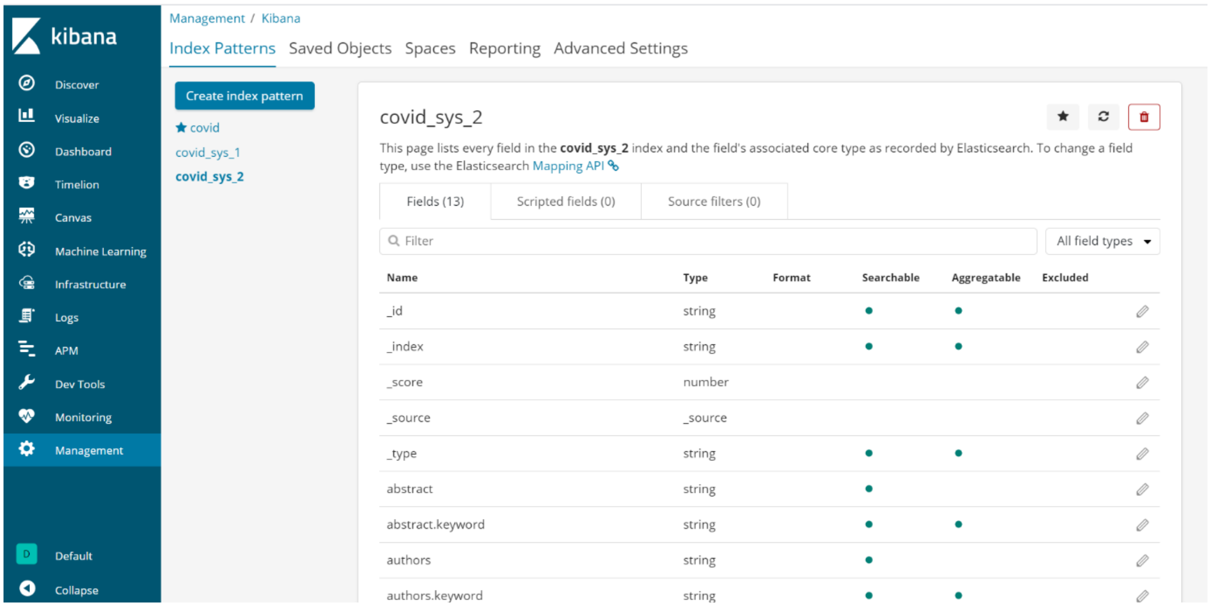
```python
#### Using n-gram ####

#--- n-gram for 'title' column ---#
title = d['title']
for words in title:
    n_grams = ngrams(nltk.word_tokenize(words), 2)
    keywords_title = [' '.join(grams) for grams in n_grams]
    print(keywords_title)

#--- n-gram for 'abstract' column ---#
abstract = d['abstract']
for words in abstract:
    n_grams = ngrams(nltk.word_tokenize(words), 2)
    keyword_abstract = [' '.join(grams) for grams in n_grams]
    print(keyword_abstract)

#--- n-gram for 'authors' column ---#
for words in d['authors']:
    n_grams = ngrams(nltk.word_tokenize(words), 2)
    keyword_authors = [' '.join(grams) for grams in n_grams]
    print( keyword_authors)

#--- n-gram for 'journal' column ---#
for words in d['journal']:
    n_grams = ngrams(nltk.word_tokenize(words), 2)
    keyword_journal = [' '.join(grams) for grams in n_grams]
    print(keyword_journal)
```

```
['Clinical features', 'features of', 'of culture-proven', 'culture-proven Mycoplasma', 'Mycoplasma pneumoniae', 'pneumoniae i
nfections', 'infections at', 'at King', 'King Abdulaziz', 'Abdulaziz University', 'University Hospital', 'Hospital ,', ', Jed
dah', 'Jeddah ,', ', Saudi', 'Saudi Arabia']
['Nitric oxide', 'oxide :', ': a', 'a pro-inflammatory', 'pro-inflammatory mediator', 'mediator in', 'in lung', 'lung diseas
e', 'disease ?']
['Surfactant protein-D', 'protein-D and', 'and pulmonary', 'pulmonary host', 'host defense']
['Role of', 'of endothelin-1', 'endothelin-1 in', 'in lung', 'lung disease']
['Gene expression', 'expression in', 'in epithelial', 'epithelial cells', 'cells in', 'in response', 'response to', 'to pneum
ovirus', 'pneumovirus infection']
['Sequence requirements', 'requirements for', 'for RNA', 'RNA strand', 'strand transfer', 'transfer during', 'during nidoviru
s', 'nidovirus discontinuous', 'discontinuous subgenomic', 'subgenomic RNA', 'RNA synthesis']
['Debate :', ': Transfusing', 'Transfusing to', 'to normal', 'normal haemoglobin', 'haemoglobin levels', 'levels will', 'will
not', 'not improve', 'improve outcome']
['The 21st', '21st International', 'International Symposium', 'Symposium on', 'on Intensive', 'Intensive Care', 'Care and',
'and Emergency', 'Emergency Medicine', 'Medicine ,', ', Brussels', 'Brussels ,', ', Belgium', 'Belgium ,', ', 20-23', '20-23
March', 'March 2001']
['Heme oxygenase-1', 'oxygenase-1 and', 'and carbon', 'carbon monoxide', 'monoxide in', 'in pulmonary', 'pulmonary medicine']
['Technical Description', 'Description of', 'of RODS', 'RODS :', ': A', 'A Real-time', 'Real-time Public', 'Public Health',
'Health Surveillance', 'Surveillance System']
```

After pre- processing the data I have created index named **“covid_sys_2”.**

```python
#### Load the data into Elastic Search ####
# ====== Connection ====== #

# Connection to ElasticSearch
es = Elasticsearch( ["localhost:9200"],
                    sniff_on_start=True,
                    sniff_on_connection_fail=True,
                    sniffer_timeout=60
                   )

# Simple index creation with no particular mapping
es.indices.create(index='covid_sys_2',body={})

# ====== Inserting Documents ====== #

# Creating a simple Pandas DataFrame
dataframe = pd.DataFrame(data = {'title' : d["title1"], 'abstract': d["abstract1"], 'authors':d['authors1'], 'journal': d['journa

# Bulk inserting documents. Each row in the DataFrame will be a document in ElasticSearch
documents = dataframe.to_dict(orient='records')
bulk(es, documents, index='covid_sys_2',doc_type='covid_data', raise_on_error=True)
```

In Kibana go to **Management** > **Index Pattern** > **covid_sys_2**.



covid_sys_2

This page lists every field in the **covid_sys_2** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch Mapping API

Name	Type	Format	Searchable	Aggregatable	Excluded
_id	string		●	●	
_index	string		●	●	
_score	number				
_source	_source				
_type	string		●	●	
abstract	string		●		
abstract.keyword	string		●	●	
authors	string		●		
authors.keyword	string		●	●	

<u>Difference between two Systems:</u>

- In first IR system **IR_system_1**, I have done pre-processing steps like Splitting words, sentences, Normalization, Removing Punctuation marks and Stop words, and I have created index named **“covid_sys_1”.**

- In second IR System **IR_System_2**, I have performed Stemming and Lemmatizing and Selecting Keywords using nltk library and then created index named **“covid_sys_2”.**

Pool method (Task 3)

Query : 1

Query	different documents	Id of the documents retrieve by System 1	Id of the documents retrieve by System 2

*{
"_source":["abstract", "title","authors","journal"],
"size": 10,
"query": {
"bool": {
"should": [
{
"match":{
"abstract": "smallpox patients"
}
},
{ "match_phrase": {
"abstract": "result of vaccination"
}
}
],
"must_not" :{"match": {
"abstract": "time during an outbreak"
}
} } }
}*
	*ix4k0rkt
	qjunkqh5
	fhm8abxp
	bujydnde
	eiqypt0m
	a0kh4kap
	gkl708nu
	qb9ewpzd
	ug7v899j
	ycxyn2a2
	ligqoj24
	1gd006zy
	3tt99oax
	9ofqelrm
	j8t98qc8
	44bm52ch
	mzn448zk*
		ycxyn2a2	*ycxyn2a2*

		1gd006zy	*1gd006zy*
		ix4k0rkt	*fhm8abxp*
		eiqypt0m	*gkl708nu*
		ug7v899j	*ligqoj24*
		3tt99oax	*j8t98qc8*
		44bm52ch	*9ofqelrm*
		9ofqelrm	*mzn448zk*
		qjunkqh5	*bujydnde*
		a0kh4kap	*qb9ewpzd*

Query : 2

Query	**different documents**	**Id of the documents retrieve by System 1**	**Id of the documents retrieve by System 2**
{ "_source": ["abstract","title","authors", "journal"], "size": 10, "query": { "bool": { "filter": [{ "multi_match" :{ "query": "Reasons of infections occured", "fields": "abstract^3", "minimum_should_match": "50%" } }], "should": [{ "match": { "abstract": "clinical samples" } }] } } }	ix4k0rkt *fmgnavfq 6f8vyziv y7urtkxo mrst93rh vuvgvz4n jy7j8sh0 leedutqo kfkumsux d5d3y7jf g0ke3xh1 pebc17zw*	ix4k0rkt	*6f8vyziv*
		6f8vyziv	*ix4k0rkt*
		mrst93rh	*vuvgvz4n*
		vuvgvz4n	*mrst93rh*
		jy7j8sh0	*jy7j8sh0*
		leedutqo	*leedutqo*
		kfkumsux	*kfkumsux*
		d5d3y7jf	*g0ke3xh1*
		pebc17zw	*d5d3y7jf*
		fmgnavfq	*y7urtkxo*

Query : 3

Query	**different documents**	**Id of the documents retrieve by System 1**	**Id of the documents retrieve by System 2**
{ "_source":["title", "abstract","authors","journal"], "size": 10,	z4sutqos *73nlxqgk nzh87aux sk9lud0o*	z4sutqos	*z4sutqos*
		73nlxqgk	*73nlxqgk*

*"query": {
"query_string": {
"query": "infectious disease AND countries population OR human travel"
}
},
"highlight": {
"fields" : {
"abstract": {}
}
}
}*
	*pixbry0c
	d0eur1hq
	emnln2ix
	l3z27806
	36bfeoqv
	zzc7n84w*
		nzh87aux	*nzh87aux*

		sk9lud0o	*l3z27806*
		pixbry0c	*sk9lud0o*
		d0eur1hq	*d0eur1hq*
		emnln2ix	*emnln2ix*
		l3z27806	*pixbry0c*
		36bfeoqv	*36bfeoqv*
		zzc7n84w	*zzc7n84w*

Relevance assessments (Task 4)

Relevance criteria:

Query	**ID of relevant documents 1**	**ID of relevant documents 2**

Query : 1
 	
*{
"_source":["abstract", "title","authors","journal"],
"size": 10,
"query": {
"bool": {
"should": [
{
"match":{
"abstract": "smallpox patients"
}
},
{ "match_phrase": {
"abstract": "result of vaccination"
}
}
],
"must_not" :{"match": {
"abstract": "time during an outbreak"
}
} } }
}*
	1gd006zy
	
	ycxyn2a2
	
	ix4k0rkt
		1gd006zy
		
		ycxyn2a2
		
		fhm8abxp
		
		gkl708nu

Query : 2	*ix4k0rkt*	*ix4k0rkt*

*{
"_source": ["abstract","title","authors", "journal"],
"size": 10,
"query": {
"bool": {
"filter": [
{
"multi_match" :{
"query": "Reasons of infections occured",
"fields": "abstract^3",
 	
"minimum_should_match": "50%"
}
}
],
"should": [
{
"match":
{
"abstract": "clinical samples"
}
}
]
}
}
}*
	*6f8vyziv
	
	mrst93rh
	
	jy7j8sh0
	
	leedutqo
	
	d5d3y7jf
	
	pebc17zw
	
	fmgnavfq*
		*6f8vyziv
		
		mrst93rh
		
		jy7j8sh0
		
		leedutqo
		
		d5d3y7jf
		
		g0ke3xh1*

Query : 3
 	
*{
"_source":["title", "abstract","authors","journal"],
"size": 10,
"query": {
"query_string": {
"query": "infectious disease AND countries population OR human travel"
}
},
"highlight": {
"fields" : {
"abstract": {}
}
}
}*
	*z4sutqos
	
	73nlxqgk
	
	nzh87aux
	
	sk9lud0o
	
	pixbry0c
	
	emnln2ix*
		*z4sutqos
		
		73nlxqgk
		
		nzh87aux
		
		sk9lud0o
		
		pixbry0c
		
		emnln2ix*

- I have created query in Kiabana Dev tools , here are some results which matchs my given information.

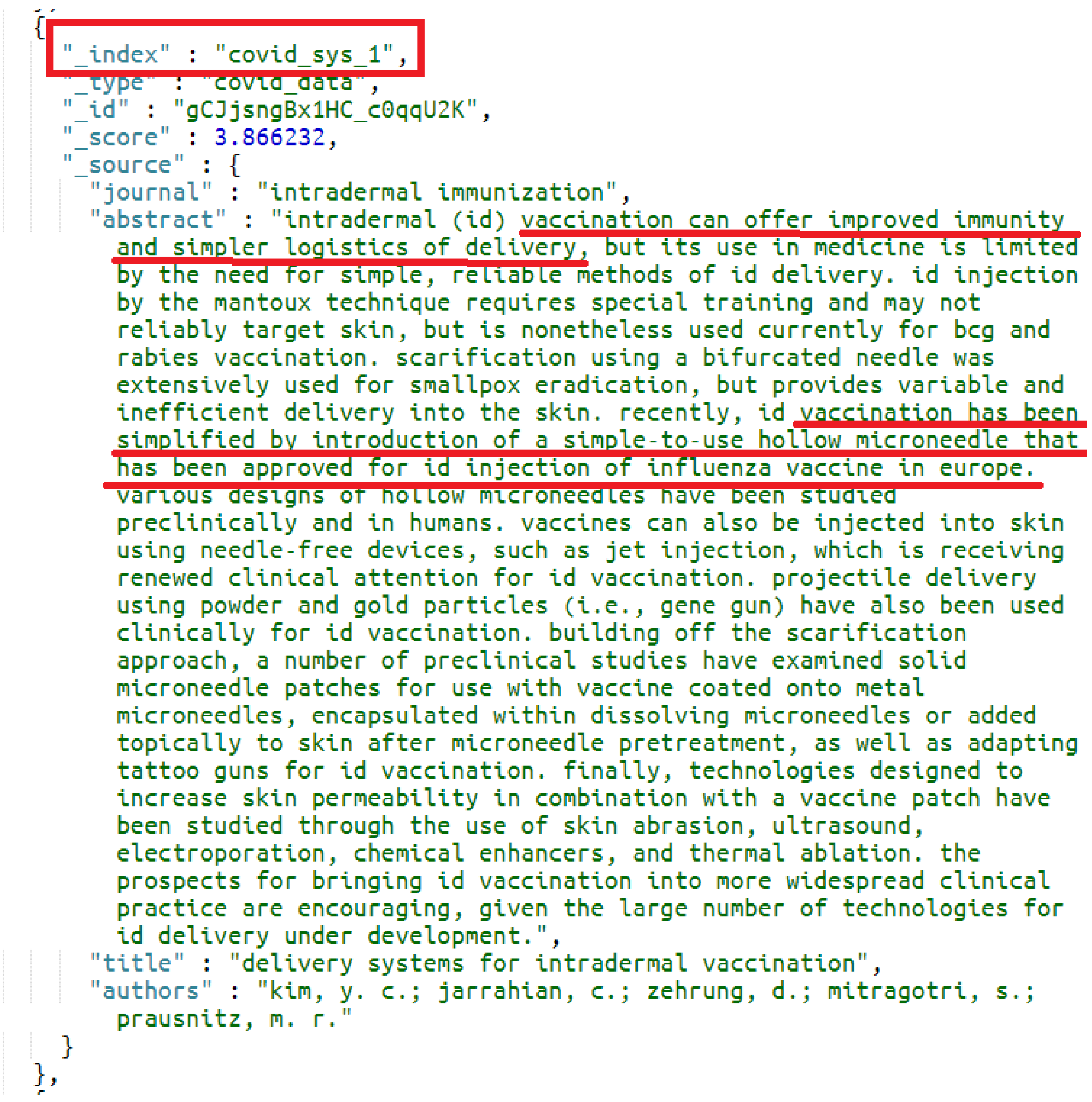
Query 1 :

System 1 Result :

My Query is : "smallpox patients" and "result of vaccination"

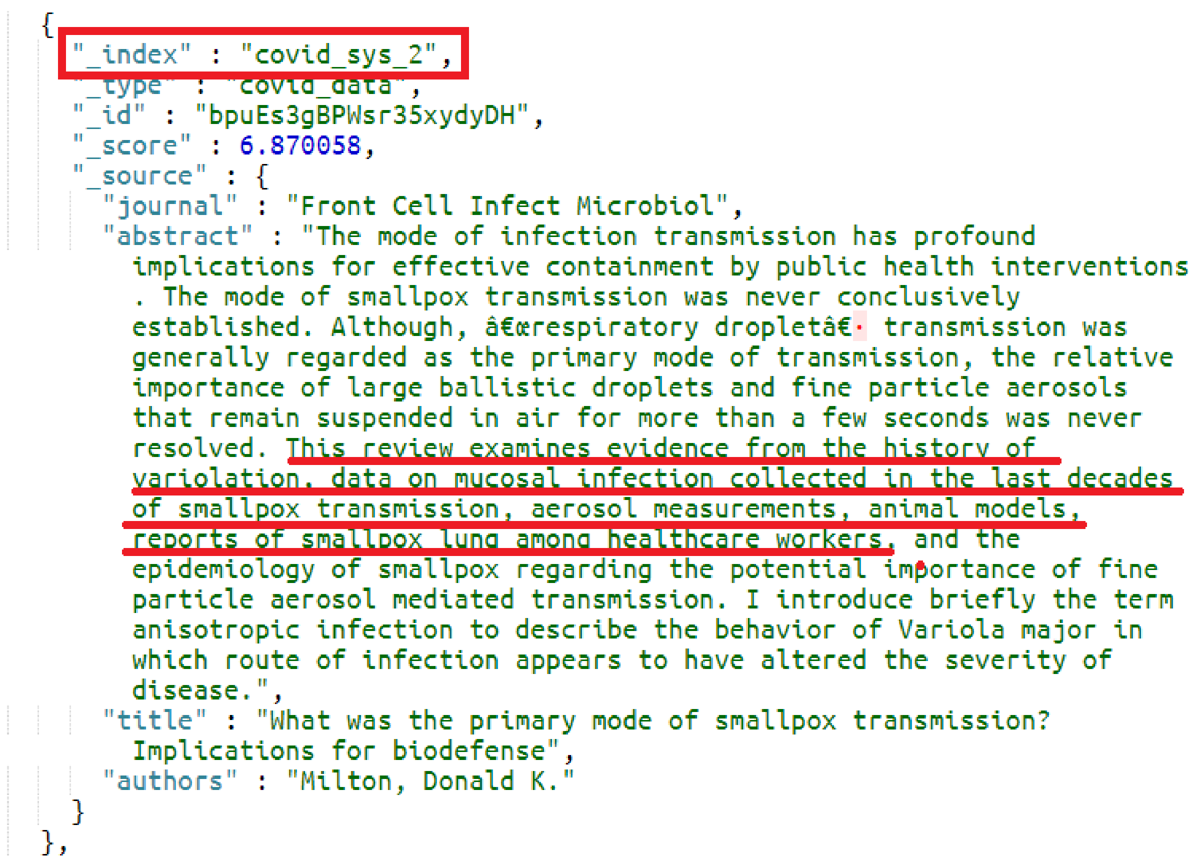
Following is the mentioned result which is related information with my given query.

I have used **match, match_phrase, must_not** like bool operations to retrieve the matching information from the given data.



```
{
  "_index" : "covid_sys_1",
  "_type" : "covid_data",
  "_id" : "gCJjsngBx1HC_c0qqU2K",
  "_score" : 3.866232,
  "_source" : {
    "journal" : "intradermal immunization",
    "abstract" : "intradermal (id) vaccination can offer improved immunity
      and simpler logistics of delivery, but its use in medicine is limited
      by the need for simple, reliable methods of id delivery. id injection
      by the mantoux technique requires special training and may not
      reliably target skin, but is nonetheless used currently for bcg and
      rabies vaccination. scarification using a bifurcated needle was
      extensively used for smallpox eradication, but provides variable and
      inefficient delivery into the skin. recently, id vaccination has been
      simplified by introduction of a simple-to-use hollow microneedle that
      has been approved for id injection of influenza vaccine in europe.
      various designs of hollow microneedles have been studied
      preclinically and in humans. vaccines can also be injected into skin
      using needle-free devices, such as jet injection, which is receiving
      renewed clinical attention for id vaccination. projectile delivery
      using powder and gold particles (i.e., gene gun) have also been used
      clinically for id vaccination. building off the scarification
      approach, a number of preclinical studies have examined solid
      microneedle patches for use with vaccine coated onto metal
      microneedles, encapsulated within dissolving microneedles or added
      topically to skin after microneedle pretreatment, as well as adapting
      tattoo guns for id vaccination. finally, technologies designed to
      increase skin permeability in combination with a vaccine patch have
      been studied through the use of skin abrasion, ultrasound,
      electroporation, chemical enhancers, and thermal ablation. the
      prospects for bringing id vaccination into more widespread clinical
      practice are encouraging, given the large number of technologies for
      id delivery under development.",
    "title" : "delivery systems for intradermal vaccination",
    "authors" : "kim, y. c.; jarrahian, c.; zehrung, d.; mitragotri, s.;
      prausnitz, m. r."
  }
},
```

System 2 Result :



```
{
  "_index" : "covid_sys_2",
  "_type" : "covid_data",
  "_id" : "bpuEs3gBPWsr35xydyDH",
  "_score" : 6.870058,
  "_source" : {
    "journal" : "Front Cell Infect Microbiol",
    "abstract" : "The mode of infection transmission has profound
      implications for effective containment by public health interventions
      . The mode of smallpox transmission was never conclusively
      established. Although, â€œrespiratory dropletâ€· transmission was
      generally regarded as the primary mode of transmission, the relative
      importance of large ballistic droplets and fine particle aerosols
      that remain suspended in air for more than a few seconds was never
      resolved. This review examines evidence from the history of
      variolation, data on mucosal infection collected in the last decades
      of smallpox transmission, aerosol measurements, animal models,
      reports of smallpox lung among healthcare workers, and the
      epidemiology of smallpox regarding the potential importance of fine
      particle aerosol mediated transmission. I introduce briefly the term
      anisotropic infection to describe the behavior of Variola major in
      which route of infection appears to have altered the severity of
      disease.",
    "title" : "What was the primary mode of smallpox transmission?
      Implications for biodefense",
    "authors" : "Milton, Donald K."
  }
},
```

Query 2 :

System 1 Result :

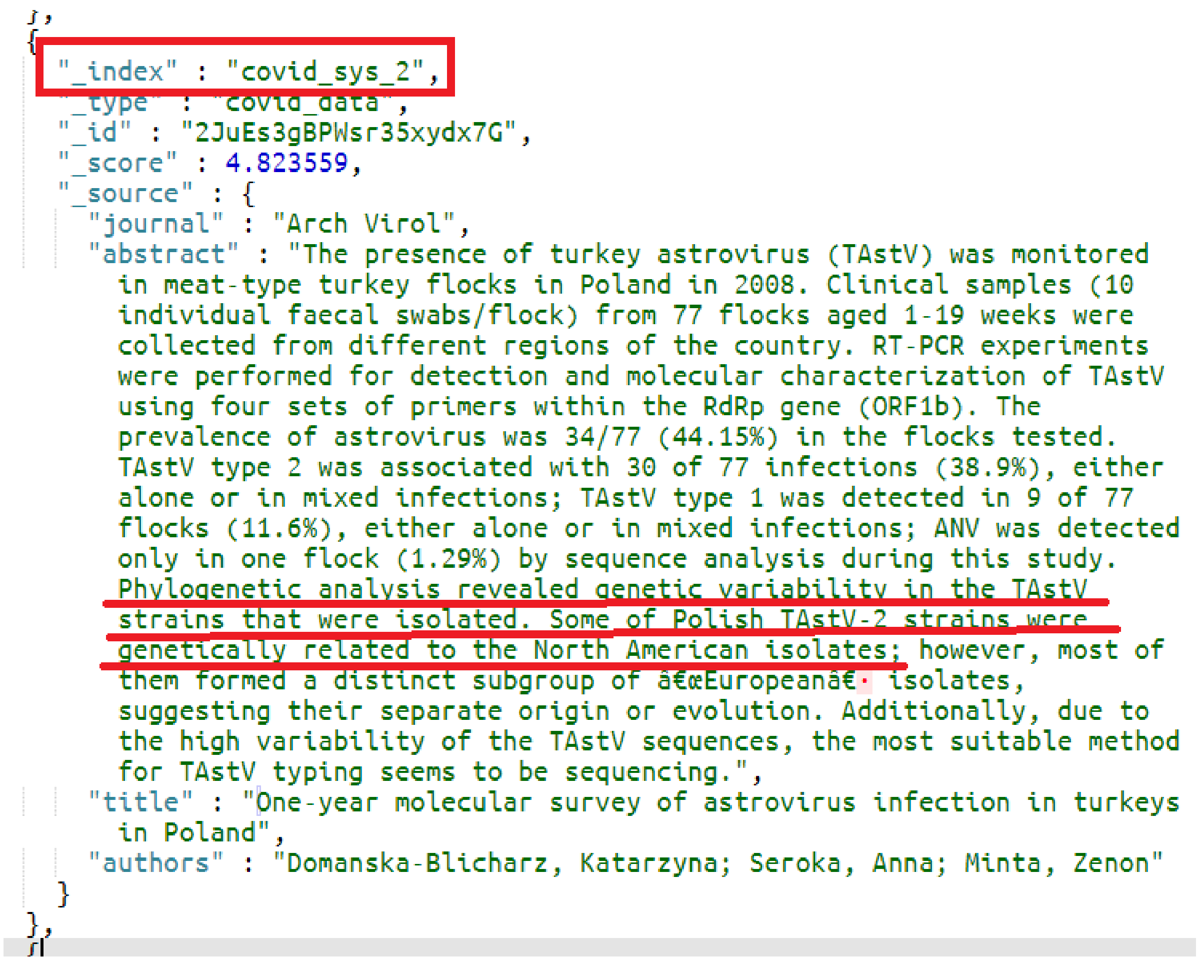
<u>My Query is</u> : “Reasons of infections occurred",

Following is the mentioned result which is related information with my given query.

I have used **multi_match** to retrieve the matching phrases from the given data in the query.

```
{
  "_index" : "covid_sys_1",
  "_type" : "covid_data",
  "_id" : "pyJjsngBx1HC_c0qqU2K",
  "_score" : 6.3305507,
  "_source" : {
    "journal" : "virol j",
    "abstract" : "background: numerous reports have described the
      epidemiological and clinical characteristics of influenza a (h1n  200
      infected patients. however, data on the effects of bacterial
      coinfection on these patients are very scarce. therefore, this study
      explores the impact of bacterial coinfection on the clinical and
      laboratory parameters amongst h1n hospitalized patients. findings:
      this retrospective study involved hospitalized patients with
      laboratory-confirmed h1n infections (september 200 to may 201 .
      relevant clinical data and the detection of bacterial coinfection
      from respiratory or sterile site samples were obtained. multiplex pcr
      was used to determine the co-existence of other respiratory viruses.
      comparison was made between patients with and without bacterial
      coinfection. the occurrence of coinfection was 3 ; 1 (2 ) bacterial
      and only  ( ) viral. mycoplasma pneumoniae (n =   was the commonest
      bacteria followed by staphylococcus aureus (n =  . in univariate
      analysis, clinical factors associated with bacterial coinfection were
      age > 5 years (p =  0 , presence of comorbidity (p =  0 , liver
      impairment (p =  0 , development of complications (p =  00  and
      supplemental oxygen requirement (p =  0 . leukocytosis (p =  0  and
      neutrophilia (p =  00  were higher in bacterial coinfected patients.
      multivariate logistic regression analysis revealed that age > 5 years
      and combined complications were predictive of bacterial coinfection.
      conclusions: bacterial coinfection is not uncommon in h1n infected
      patients and is more frequently noted in the older aged patients and
      is associated with higher rates of complications. also, as adjunct to
      clinical findings, clinicians need to have a higher index of
      suspicion if neutrophilia was identified at admission as it may
      denote bacterial coinfection.",
    "title" : "epidemiology and clinical characteristics of hospitalized
      patients with pandemic influenza a (h1n  200 infections: the effects
      of bacterial coinfection",
    "authors" : "dhanoa, amreeta; fang, ngim c; hassan, sharifah s;
      kaniappan, priyatharisni; rajasekaram, ganeswrie"
  }
},
```

System 2 Result :



```
},
{
  "_index" : "covid_sys_2",
  "_type" : "covid_data",
  "_id" : "2JuEs3gBPWsr35xydx7G",
  "_score" : 4.823559,
  "_source" : {
    "journal" : "Arch Virol",
    "abstract" : "The presence of turkey astrovirus (TAstV) was monitored
      in meat-type turkey flocks in Poland in 2008. Clinical samples (10
      individual faecal swabs/flock) from 77 flocks aged 1-19 weeks were
      collected from different regions of the country. RT-PCR experiments
      were performed for detection and molecular characterization of TAstV
      using four sets of primers within the RdRp gene (ORF1b). The
      prevalence of astrovirus was 34/77 (44.15%) in the flocks tested.
      TAstV type 2 was associated with 30 of 77 infections (38.9%), either
      alone or in mixed infections; TAstV type 1 was detected in 9 of 77
      flocks (11.6%), either alone or in mixed infections; ANV was detected
      only in one flock (1.29%) by sequence analysis during this study.
      Phylogenetic analysis revealed genetic variability in the TAstV
      strains that were isolated. Some of Polish TAstV-2 strains were
      genetically related to the North American isolates; however, most of
      them formed a distinct subgroup of â€œEuropeanâ€· isolates,
      suggesting their separate origin or evolution. Additionally, due to
      the high variability of the TAstV sequences, the most suitable method
      for TAstV typing seems to be sequencing.",
    "title" : "One-year molecular survey of astrovirus infection in turkeys
      in Poland",
    "authors" : "Domanska-Blicharz, Katarzyna; Seroka, Anna; Minta, Zenon"
  }
},
{
```

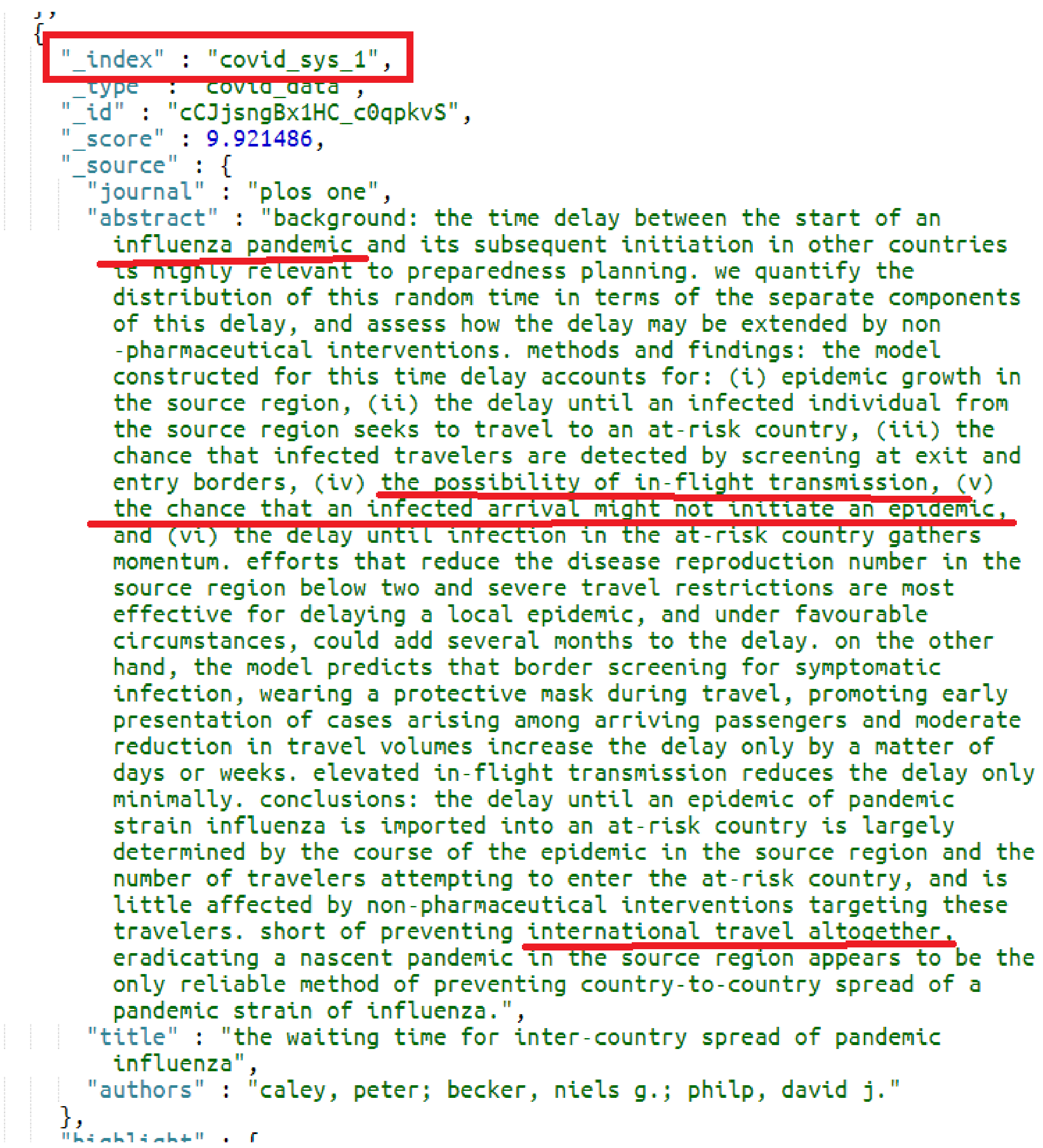
Query 3 :

System 1 Result :

__**My Query is**__ : “infectious disease **AND** countries population **OR** human travel"

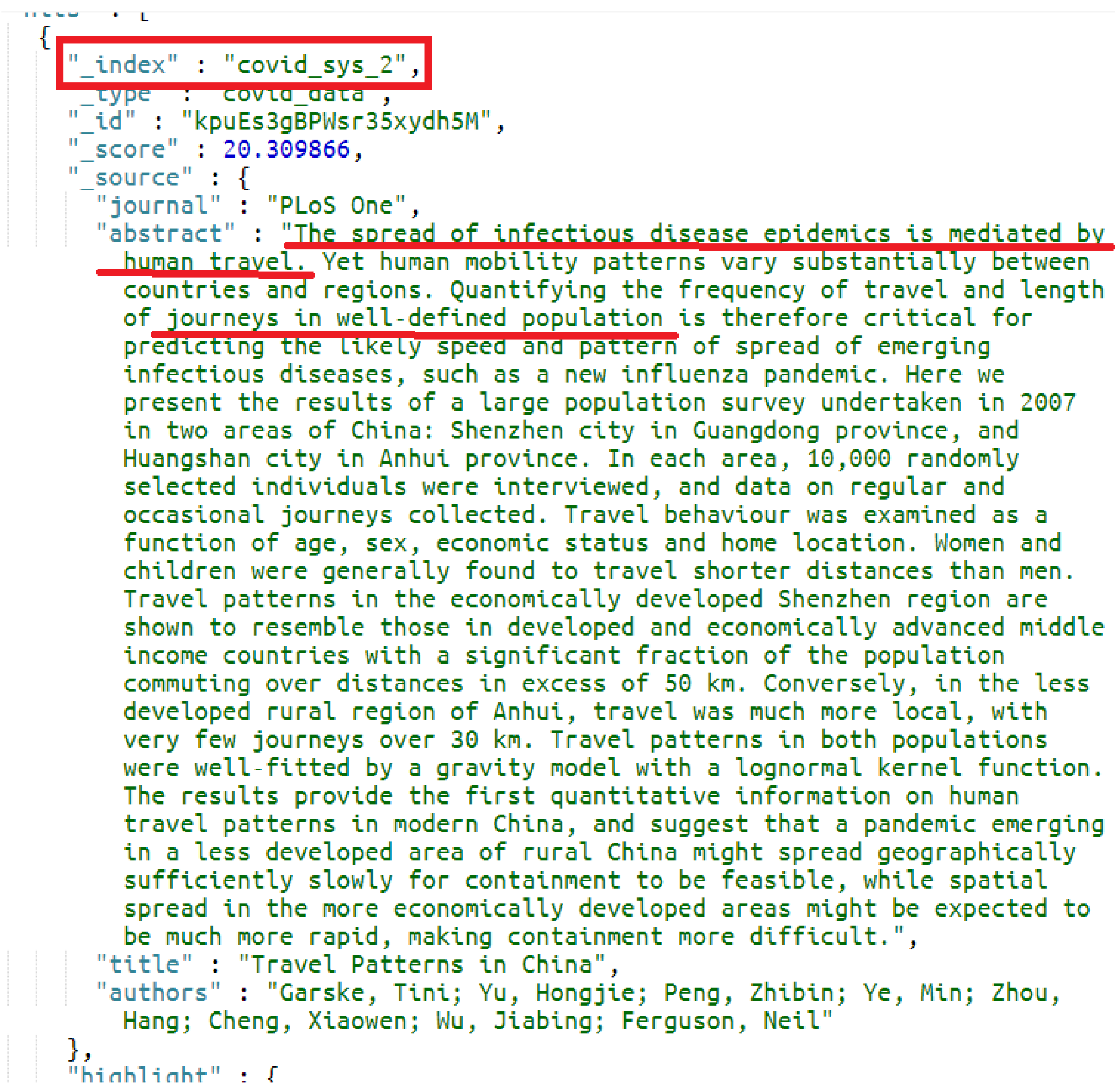
Following is the mentioned result which is related information with my given query.

I have used query string in the query, and highlighted filed **“abstract”** that will be highlight in tags.



```
},
{
  "_index" : "covid_sys_1",
  "_type" : "covid_data",
  "_id" : "cCJjsngBx1HC_c0qpkvS",
  "_score" : 9.921486,
  "_source" : {
    "journal" : "plos one",
    "abstract" : "background: the time delay between the start of an
      influenza pandemic and its subsequent initiation in other countries
      is highly relevant to preparedness planning. we quantify the
      distribution of this random time in terms of the separate components
      of this delay, and assess how the delay may be extended by non
      -pharmaceutical interventions. methods and findings: the model
      constructed for this time delay accounts for: (i) epidemic growth in
      the source region, (ii) the delay until an infected individual from
      the source region seeks to travel to an at-risk country, (iii) the
      chance that infected travelers are detected by screening at exit and
      entry borders, (iv) the possibility of in-flight transmission, (v)
      the chance that an infected arrival might not initiate an epidemic,
      and (vi) the delay until infection in the at-risk country gathers
      momentum. efforts that reduce the disease reproduction number in the
      source region below two and severe travel restrictions are most
      effective for delaying a local epidemic, and under favourable
      circumstances, could add several months to the delay. on the other
      hand, the model predicts that border screening for symptomatic
      infection, wearing a protective mask during travel, promoting early
      presentation of cases arising among arriving passengers and moderate
      reduction in travel volumes increase the delay only by a matter of
      days or weeks. elevated in-flight transmission reduces the delay only
      minimally. conclusions: the delay until an epidemic of pandemic
      strain influenza is imported into an at-risk country is largely
      determined by the course of the epidemic in the source region and the
      number of travelers attempting to enter the at-risk country, and is
      little affected by non-pharmaceutical interventions targeting these
      travelers. short of preventing international travel altogether,
      eradicating a nascent pandemic in the source region appears to be the
      only reliable method of preventing country-to-country spread of a
      pandemic strain of influenza.",
    "title" : "the waiting time for inter-country spread of pandemic
      influenza",
    "authors" : "caley, peter; becker, niels g.; philp, david j."
  },
  "highlight" : {
```

System 2 Result :



```
  {
    "_index" : "covid_sys_2",
    "_type" : "covid_data",
    "_id" : "kpuEs3gBPWsr35xydh5M",
    "_score" : 20.309866,
    "_source" : {
      "journal" : "PLoS One",
      "abstract" : "The spread of infectious disease epidemics is mediated by
        human travel. Yet human mobility patterns vary substantially between
        countries and regions. Quantifying the frequency of travel and length
        of journeys in well-defined population is therefore critical for
        predicting the likely speed and pattern of spread of emerging
        infectious diseases, such as a new influenza pandemic. Here we
        present the results of a large population survey undertaken in 2007
        in two areas of China: Shenzhen city in Guangdong province, and
        Huangshan city in Anhui province. In each area, 10,000 randomly
        selected individuals were interviewed, and data on regular and
        occasional journeys collected. Travel behaviour was examined as a
        function of age, sex, economic status and home location. Women and
        children were generally found to travel shorter distances than men.
        Travel patterns in the economically developed Shenzhen region are
        shown to resemble those in developed and economically advanced middle
        income countries with a significant fraction of the population
        commuting over distances in excess of 50 km. Conversely, in the less
        developed rural region of Anhui, travel was much more local, with
        very few journeys over 30 km. Travel patterns in both populations
        were well-fitted by a gravity model with a lognormal kernel function.
        The results provide the first quantitative information on human
        travel patterns in modern China, and suggest that a pandemic emerging
        in a less developed area of rural China might spread geographically
        sufficiently slowly for containment to be feasible, while spatial
        spread in the more economically developed areas might be expected to
        be much more rapid, making containment more difficult.",
      "title" : "Travel Patterns in China",
      "authors" : "Garske, Tini; Yu, Hongjie; Peng, Zhibin; Ye, Min; Zhou,
        Hang; Cheng, Xiaowen; Wu, Jiabing; Ferguson, Neil"
    },
```

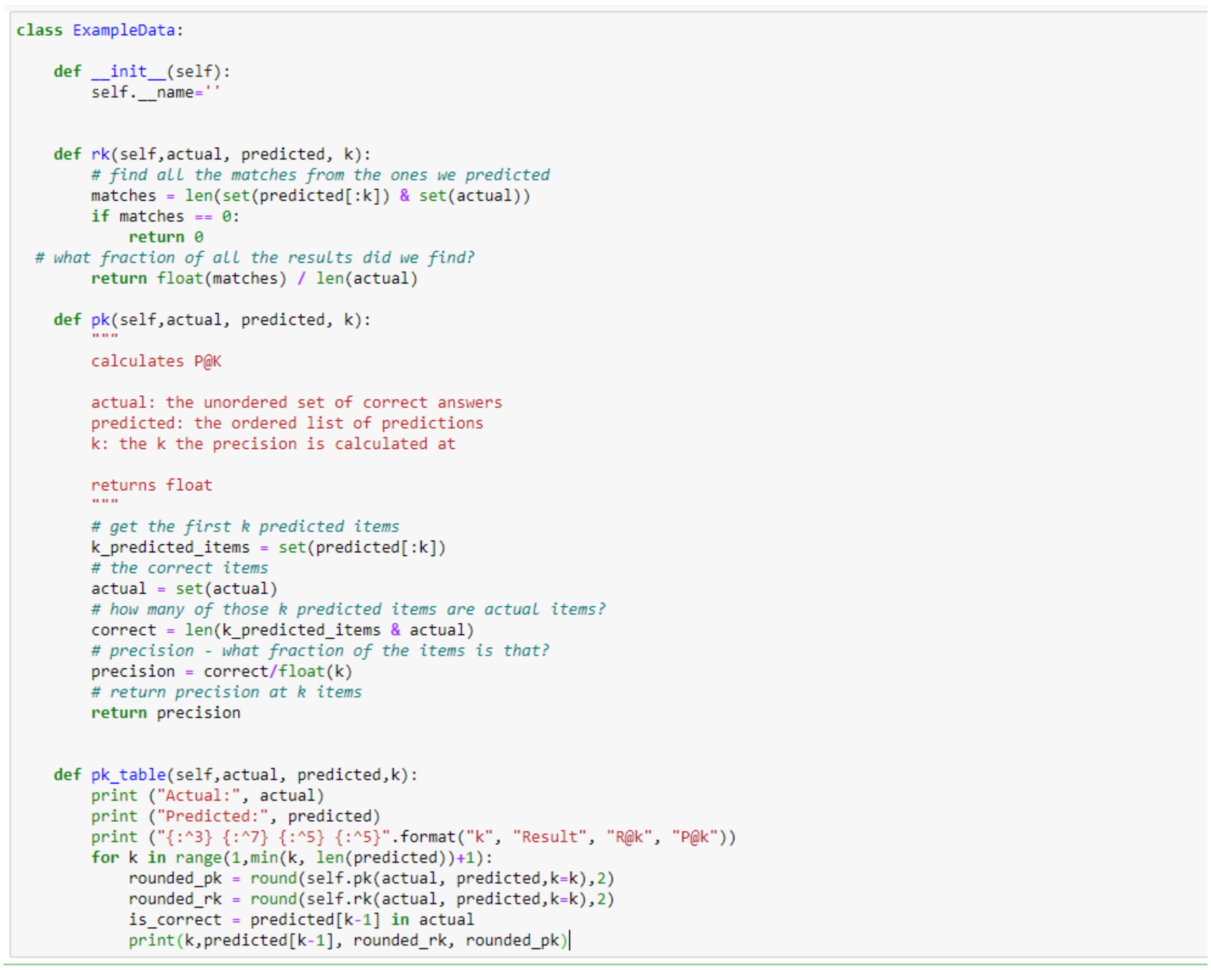
- So, this way I have calculated my all-relevant and irrelevant documents from retrieved documents for each three queries and for both individual systems.

Evaluation (Task 5)

- The given table contains information about both of the system's Precision and Recall values.

	System 1		System 2	
	P@5	**R@5**	**P@5**	**R@5**
Q1	0.6	1.0	0.8	1.0
Q2	0.8	0.5	0.8	0.57
Q3	1.0	0.83	0.8	0.66

- I have written code to calculate the precision, Recall and show Precision and Recall Table.

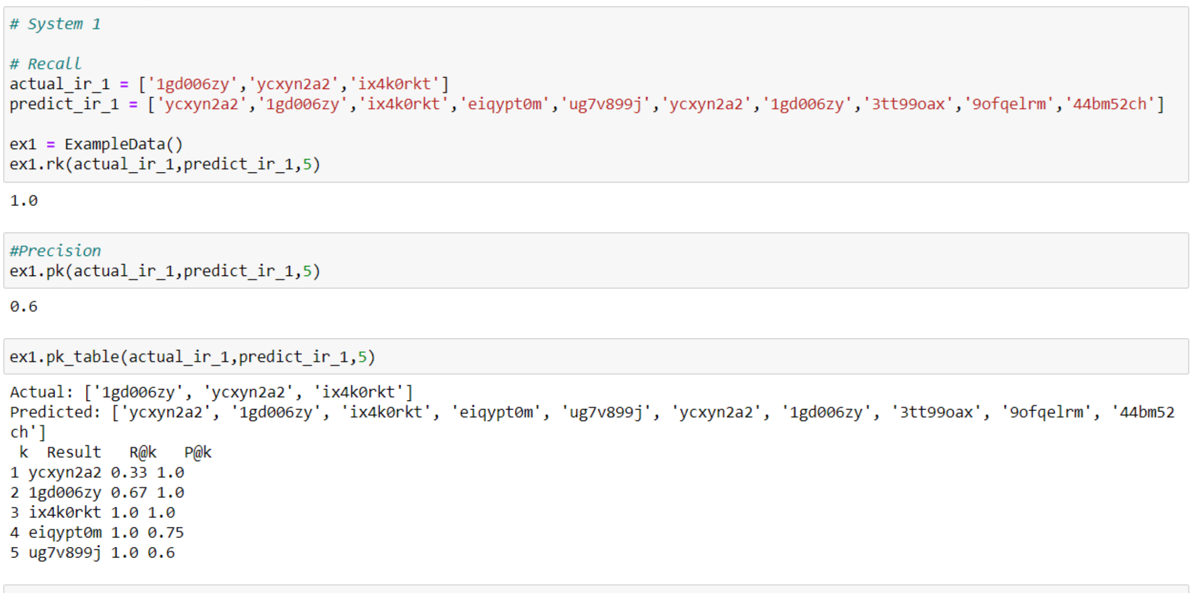


```python
class ExampleData:

    def __init__(self):
        self.__name=''


    def rk(self,actual, predicted, k):
        # find all the matches from the ones we predicted
        matches = len(set(predicted[:k]) & set(actual))
        if matches == 0:
            return 0
  # what fraction of all the results did we find?
        return float(matches) / len(actual)

    def pk(self,actual, predicted, k):
        """
        calculates P@K

        actual: the unordered set of correct answers
        predicted: the ordered list of predictions
        k: the k the precision is calculated at

        returns float
        """
        # get the first k predicted items
        k_predicted_items = set(predicted[:k])
        # the correct items
        actual = set(actual)
        # how many of those k predicted items are actual items?
        correct = len(k_predicted_items & actual)
        # precision - what fraction of the items is that?
        precision = correct/float(k)
        # return precision at k items
        return precision


    def pk_table(self,actual, predicted,k):
        print ("Actual:", actual)
        print ("Predicted:", predicted)
        print ("{:^3} {:^7} {:^5} {:^5}".format("k", "Result", "R@k", "P@k"))
        for k in range(1,min(k, len(predicted))+1):
            rounded_pk = round(self.pk(actual, predicted,k=k),2)
            rounded_rk = round(self.rk(actual, predicted,k=k),2)
            is_correct = predicted[k-1] in actual
            print(k,predicted[k-1], rounded_rk, rounded_pk)
```

- I passed actual document as my relevant document and as predicated documents in which I considered 10 most result from query for both individual system.

For Query 1 :

Consider System 1 Precision and Recall Calculation:



```
# System 1

# Recall
actual_ir_1 = ['1gd006zy','ycxyn2a2','ix4k0rkt']
predict_ir_1 = ['ycxyn2a2','1gd006zy','ix4k0rkt','eiqypt0m','ug7v899j','ycxyn2a2','1gd006zy','3tt99oax','9ofqelrm','44bm52ch']

ex1 = ExampleData()
ex1.rk(actual_ir_1,predict_ir_1,5)
```

```
1.0
```

```
#Precision
ex1.pk(actual_ir_1,predict_ir_1,5)
```

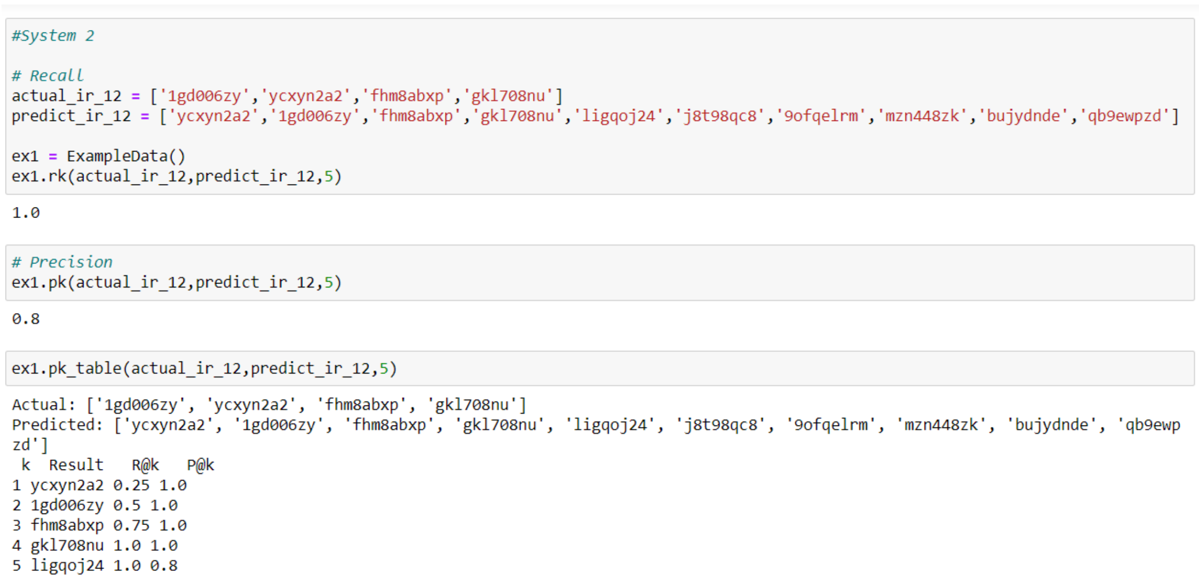
```
0.6
```

```
ex1.pk_table(actual_ir_1,predict_ir_1,5)
```

```
Actual: ['1gd006zy', 'ycxyn2a2', 'ix4k0rkt']
Predicted: ['ycxyn2a2', '1gd006zy', 'ix4k0rkt', 'eiqypt0m', 'ug7v899j', 'ycxyn2a2', '1gd006zy', '3tt99oax', '9ofqelrm', '44bm52
ch']
 k  Result   R@k   P@k
1 ycxyn2a2 0.33 1.0
2 1gd006zy 0.67 1.0
3 ix4k0rkt 1.0 1.0
4 eiqypt0m 1.0 0.75
5 ug7v899j 1.0 0.6
```

.

Consider System 2 Precision and Recall Calculation:



```
#System 2

# Recall
actual_ir_12 = ['1gd006zy','ycxyn2a2','fhm8abxp','gkl708nu']
predict_ir_12 = ['ycxyn2a2','1gd006zy','fhm8abxp','gkl708nu','ligqoj24','j8t98qc8','9ofqelrm','mzn448zk','bujydnde','qb9ewpzd']

ex1 = ExampleData()
ex1.rk(actual_ir_12,predict_ir_12,5)
```

```
1.0
```

```
# Precision
ex1.pk(actual_ir_12,predict_ir_12,5)
```

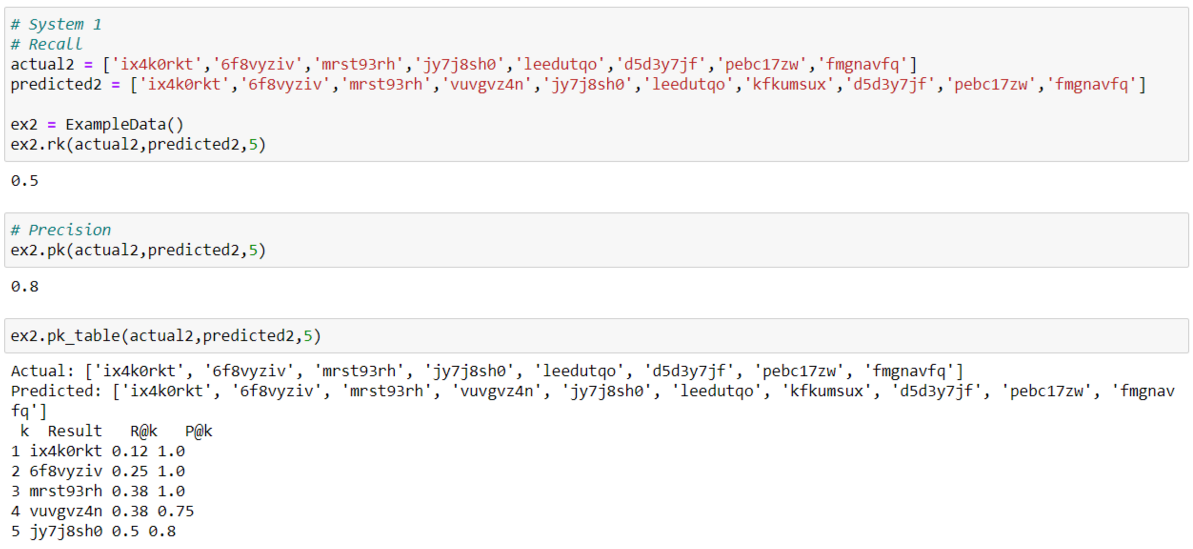
```
0.8
```

```
ex1.pk_table(actual_ir_12,predict_ir_12,5)
```

```
Actual: ['1gd006zy', 'ycxyn2a2', 'fhm8abxp', 'gkl708nu']
Predicted: ['ycxyn2a2', '1gd006zy', 'fhm8abxp', 'gkl708nu', 'ligqoj24', 'j8t98qc8', '9ofqelrm', 'mzn448zk', 'bujydnde', 'qb9ewp
zd']
 k  Result   R@k   P@k
1 ycxyn2a2 0.25 1.0
2 1gd006zy 0.5 1.0
3 fhm8abxp 0.75 1.0
4 gkl708nu 1.0 1.0
5 ligqoj24 1.0 0.8
```

For Query 2 :

Consider System 1 Precision and Recall Calculation:



```
# System 1
# Recall
actual2 = ['ix4k0rkt','6f8vyziv','mrst93rh','jy7j8sh0','leedutqo','d5d3y7jf','pebc17zw','fmgnavfq']
predicted2 = ['ix4k0rkt','6f8vyziv','mrst93rh','vuvgvz4n','jy7j8sh0','leedutqo','kfkumsux','d5d3y7jf','pebc17zw','fmgnavfq']

ex2 = ExampleData()
ex2.rk(actual2,predicted2,5)
```

```
0.5
```

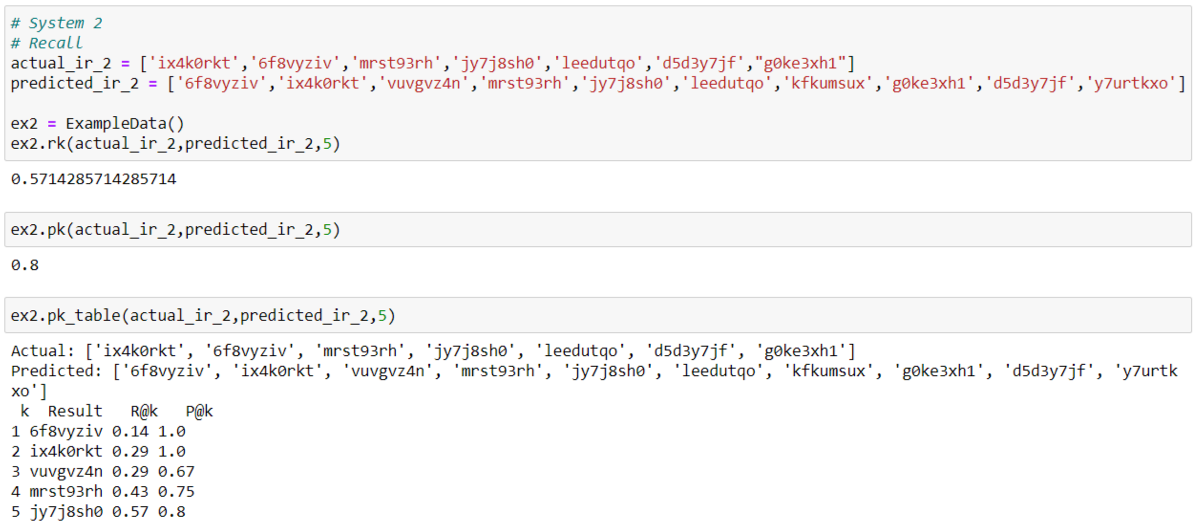
```
# Precision
ex2.pk(actual2,predicted2,5)
```

```
0.8
```

```
ex2.pk_table(actual2,predicted2,5)
```

```
Actual: ['ix4k0rkt', '6f8vyziv', 'mrst93rh', 'jy7j8sh0', 'leedutqo', 'd5d3y7jf', 'pebc17zw', 'fmgnavfq']
Predicted: ['ix4k0rkt', '6f8vyziv', 'mrst93rh', 'vuvgvz4n', 'jy7j8sh0', 'leedutqo', 'kfkumsux', 'd5d3y7jf', 'pebc17zw', 'fmgnav
fq']
 k  Result   R@k   P@k
1 ix4k0rkt 0.12 1.0
2 6f8vyziv 0.25 1.0
3 mrst93rh 0.38 1.0
4 vuvgvz4n 0.38 0.75
5 jy7j8sh0 0.5 0.8
```

Consider System 2 Precision and Recall Calculation:



```
# System 2
# Recall
actual_ir_2 = ['ix4k0rkt','6f8vyziv','mrst93rh','jy7j8sh0','leedutqo','d5d3y7jf',"g0ke3xh1"]
predicted_ir_2 = ['6f8vyziv','ix4k0rkt','vuvgvz4n','mrst93rh','jy7j8sh0','leedutqo','kfkumsux','g0ke3xh1','d5d3y7jf','y7urtkxo']

ex2 = ExampleData()
ex2.rk(actual_ir_2,predicted_ir_2,5)
```

```
0.5714285714285714
```

```
ex2.pk(actual_ir_2,predicted_ir_2,5)
```

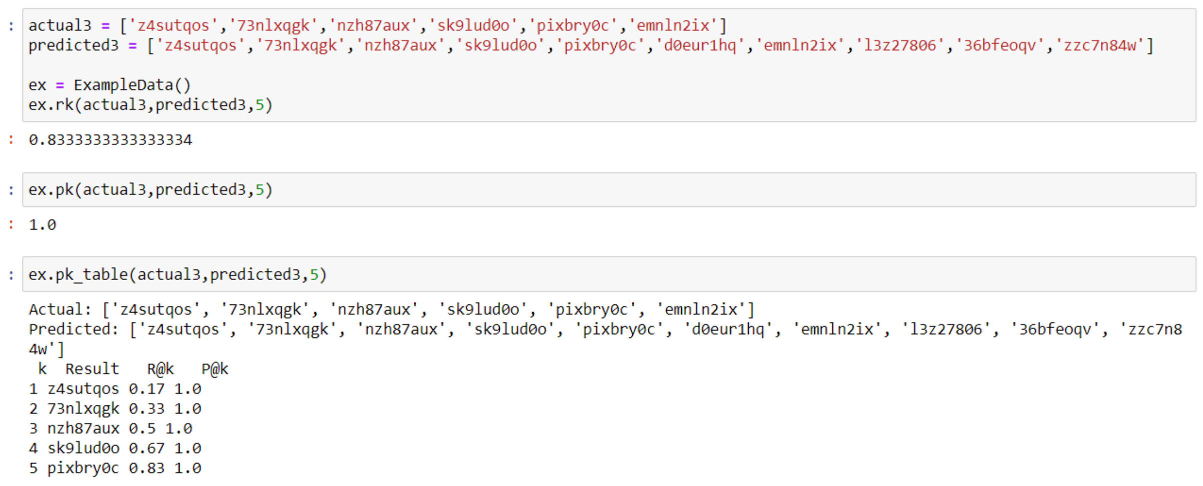
```
0.8
```

```
ex2.pk_table(actual_ir_2,predicted_ir_2,5)
```

```
Actual: ['ix4k0rkt', '6f8vyziv', 'mrst93rh', 'jy7j8sh0', 'leedutqo', 'd5d3y7jf', 'g0ke3xh1']
Predicted: ['6f8vyziv', 'ix4k0rkt', 'vuvgvz4n', 'mrst93rh', 'jy7j8sh0', 'leedutqo', 'kfkumsux', 'g0ke3xh1', 'd5d3y7jf', 'y7urtk
xo']
 k  Result   R@k   P@k
1 6f8vyziv 0.14 1.0
2 ix4k0rkt 0.29 1.0
3 vuvgvz4n 0.29 0.67
4 mrst93rh 0.43 0.75
5 jy7j8sh0 0.57 0.8
```

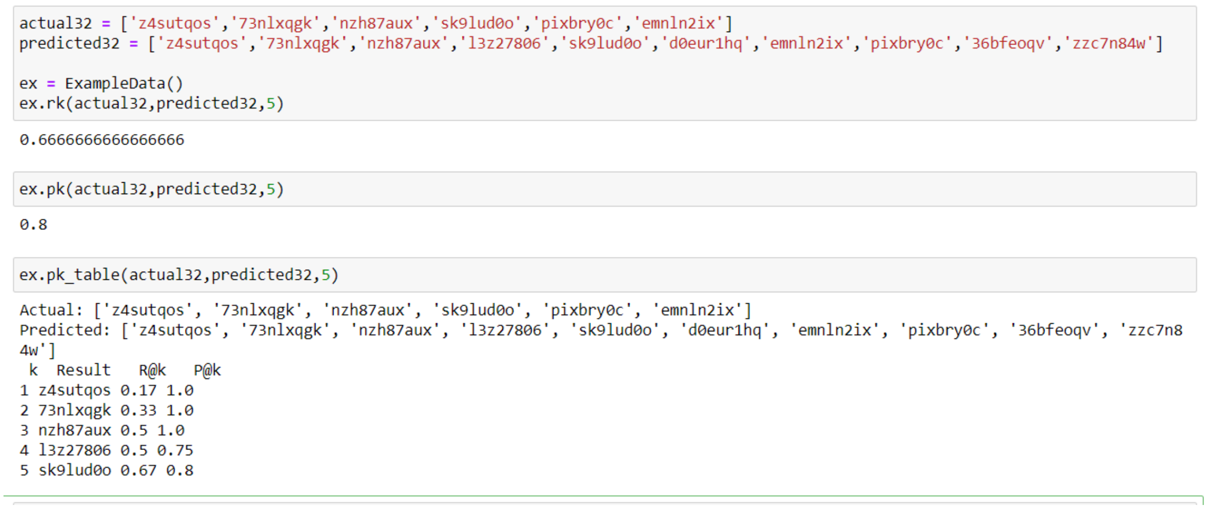
For Query 3 :

Consider System 1 Precision and Recall Calculation:



```
: actual3 = ['z4sutqos','73nlxqgk','nzh87aux','sk9lud0o','pixbry0c','emnln2ix']
  predicted3 = ['z4sutqos','73nlxqgk','nzh87aux','sk9lud0o','pixbry0c','d0eur1hq','emnln2ix','l3z27806','36bfeoqv','zzc7n84w']

  ex = ExampleData()
  ex.rk(actual3,predicted3,5)

: 0.8333333333333334

: ex.pk(actual3,predicted3,5)

: 1.0

: ex.pk_table(actual3,predicted3,5)

  Actual: ['z4sutqos', '73nlxqgk', 'nzh87aux', 'sk9lud0o', 'pixbry0c', 'emnln2ix']
  Predicted: ['z4sutqos', '73nlxqgk', 'nzh87aux', 'sk9lud0o', 'pixbry0c', 'd0eur1hq', 'emnln2ix', 'l3z27806', '36bfeoqv', 'zzc7n8
  4w']
   k  Result    R@k   P@k
  1 z4sutqos 0.17 1.0
  2 73nlxqgk 0.33 1.0
  3 nzh87aux 0.5 1.0
  4 sk9lud0o 0.67 1.0
  5 pixbry0c 0.83 1.0
```

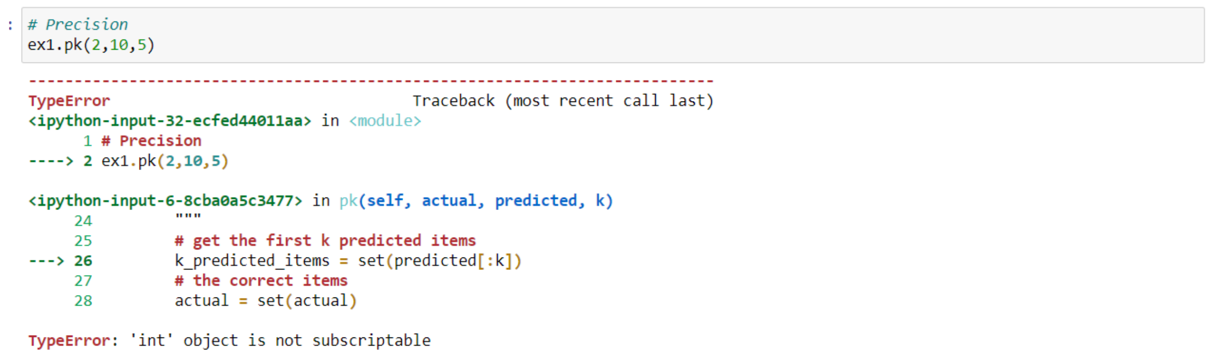
Consider System 2 Precision and Recall Calculation:



```
actual32 = ['z4sutqos','73nlxqgk','nzh87aux','sk9lud0o','pixbry0c','emnln2ix']
predicted32 = ['z4sutqos','73nlxqgk','nzh87aux','l3z27806','sk9lud0o','d0eur1hq','emnln2ix','pixbry0c','36bfeoqv','zzc7n84w']

ex = ExampleData()
ex.rk(actual32,predicted32,5)

0.6666666666666666

ex.pk(actual32,predicted32,5)

0.8

ex.pk_table(actual32,predicted32,5)

Actual: ['z4sutqos', '73nlxqgk', 'nzh87aux', 'sk9lud0o', 'pixbry0c', 'emnln2ix']
Predicted: ['z4sutqos', '73nlxqgk', 'nzh87aux', 'l3z27806', 'sk9lud0o', 'd0eur1hq', 'emnln2ix', 'pixbry0c', '36bfeoqv', 'zzc7n8
4w']
 k  Result    R@k   P@k
1 z4sutqos 0.17 1.0
2 73nlxqgk 0.33 1.0
3 nzh87aux 0.5 1.0
4 l3z27806 0.5 0.75
5 sk9lud0o 0.67 0.8
```

Problems while calculating Precision:

I have passed integer arguments in function so, it gave me the following error. After understanding the code properly, I got to know that in that arguments I have to pass list of integer or list of strings, so after replacing with list of strings, error solved.



```
: # Precision
  ex1.pk(2,10,5)

  ---------------------------------------------------------------------------
  TypeError                                 Traceback (most recent call last)
  <ipython-input-32-ecfed44011aa> in <module>
        1 # Precision
  ----> 2 ex1.pk(2,10,5)

  <ipython-input-6-8cba0a5c3477> in pk(self, actual, predicted, k)
       24         """
       25         # get the first k predicted items
  ---> 26         k_predicted_items = set(predicted[:k])
       27         # the correct items
       28         actual = set(actual)

  TypeError: 'int' object is not subscriptable
```

Conclusion:

- ❖ In first system IR_System_1, I have done pre-processing using Word Splitting, Sentences Splitting, Tokenization and Normalization and from the system, I got result for three queries for **Precision value 0.6, 0.8 and 1.0** and **Recall value 1.0, 0.5** and **0.83** respectively.

- ❖ In Second system IR_System_2, I have considered pre-processing steps as Stemming and Lemmatizing words and sentences and used n-gram library to select the keywords. I got the result from system 2 for three queries for **Precision value 0.8, 0.8 and 0.8** and **Recall value 1.0, 0.57** and **0.66** accordingly.

- ❖ Considering the above two systems, IR_System_1 has a good result in compare to IR_Syetem_2 in all three queries as it is having high Precision Value and Recall values than IR_System_2. So, We can say that IR_Sytem_1, has good pre-processing steps involved for data cleaning, which converts in better searching result.

References:

[1] https://dzone.com/articles/23-useful-elasticsearch-example-queries

[2] https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-ngram-tokenfilter.html

[3] https://www.datacamp.com/community/tutorials/stemming-lemmatization-python

[4] https://medium.com/@datamonsters/text-preprocessing-in-python-steps-tools-and-examples-bf025f872908