INFORMATION RETRIEVAL (7071CEM)

Module Leader : Dr Seyed Mousavi

Student Name : Priyanka Nelli Komath

Student ID : 11979142

Email: nellikomap@coventry.ac.uk

Task 1. Vertical Search Engine

The main objective of this task is to create an identical vertical search engine to google scholar to specifically return search results such as books, articles, and papers published by the members of SEFA “School of Economics Finance And Accounting” at the Coventry university.

A vertical search engine is referred to focus and retrieve data on a particular area. There are various different processes followed in creating this vertical search engine. Initially, I crawled the website to get the necessary data and stored the same in a Jason file, and then the crawled data is indexed to store in a structured format. After indexing the data is ranked based on the relevance to resolving the user's query.

Data crawling

A tool or a robotic or an automated process that finds and scans the web pages to retrieve information is referred to as “crawling”. The crawler in this project is used for crawling and retrieving necessary information from the SEFA page of the Coventry website and it is implemented using the pythons Beautiful Soup package. The crawler checks the robots.txt (https://pureportal.coventry.ac.uk/en/robots.txt) file to make sure not to hit the blocked websites and to access the pages with the specified time delay in the robots.txt file. This is known as “polite crawling” as it follows the correct procedure by including the robots file in the crawling process.

First, I crawled the SEFA member profiles from the website and stored necessary details in an array. Publication details are then crawled from the pureportal publications page.

Data is crawled from the below given websites:

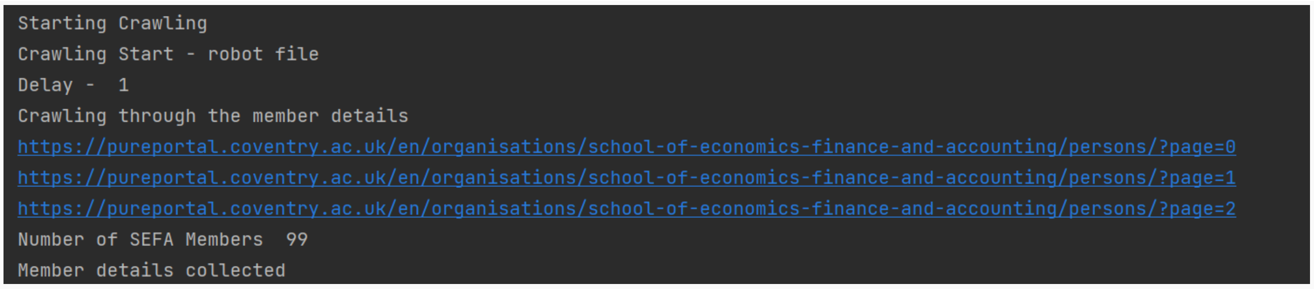
SEFA Member data:
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting/persons/

Publication Details:
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting/publications/

Result:

A. Crawled a total number of 99 members of SEFA. Out of these, 70 members have co-authored one or more publications. The crawled result set contains

1. Member name
2. Link to the member profile

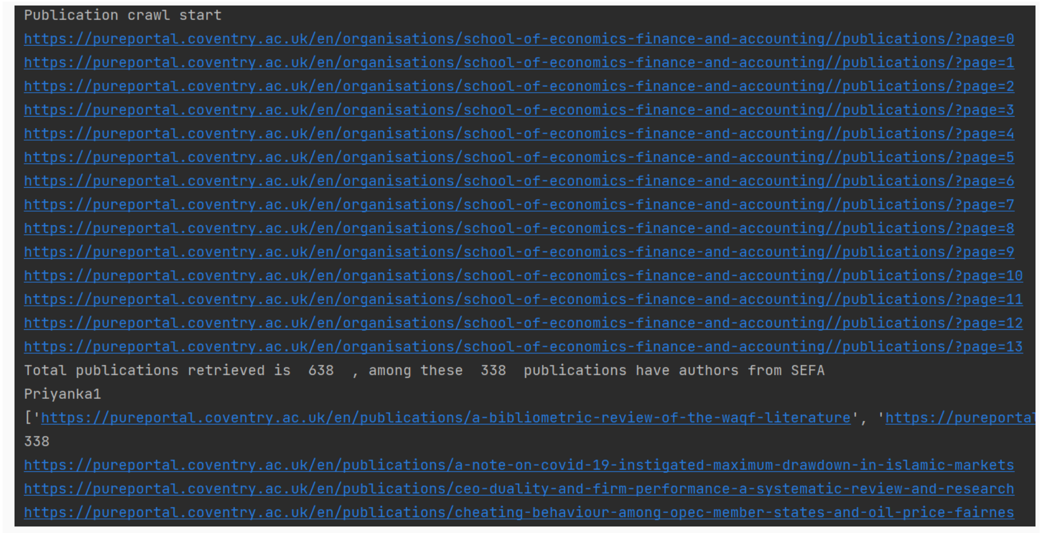


```
Starting Crawling
Crawling Start - robot file
Delay -  1
Crawling through the member details
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting/persons/?page=0
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting/persons/?page=1
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting/persons/?page=2
Number of SEFA Members  99
Member details collected
```

Fig 1: Crawling through the SEFA member details

B. 638 publications were crawled, and 338 relevant results are retrieved with atleast one SEFA member in the author list. The crawled result set contains

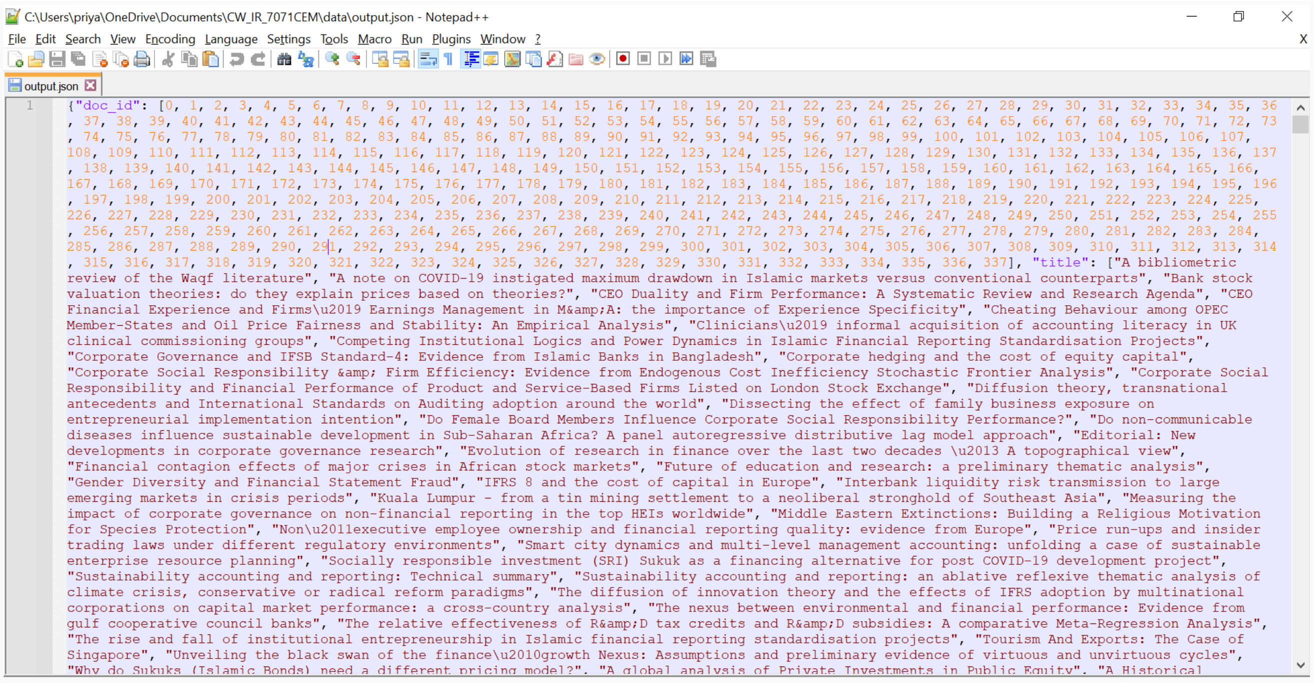
1. Document ID
2. Title
3. Link to the publication
4. Text Data
5. Publishes year
6. Authors
7. Publication content



```
Publication crawl start
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=0
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=1
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=2
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=3
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=4
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=5
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=6
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=7
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=8
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=9
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=10
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=11
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=12
https://pureportal.coventry.ac.uk/en/organisations/school-of-economics-finance-and-accounting//publications/?page=13
Total publications retrieved is  638  , among these  338  publications have authors from SEFA
Priyanka1
['https://pureportal.coventry.ac.uk/en/publications/a-bibliometric-review-of-the-waqf-literature', 'https://pureporta
338
https://pureportal.coventry.ac.uk/en/publications/a-note-on-covid-19-instigated-maximum-drawdown-in-islamic-markets
https://pureportal.coventry.ac.uk/en/publications/ceo-duality-and-firm-performance-a-systematic-review-and-research
https://pureportal.coventry.ac.uk/en/publications/cheating-behaviour-among-opec-member-states-and-oil-price-fairnes
```

Fig 2: Crawling through the publications

Retrieved results of publications is then stored in the output.json file after filtering the results by comparing the publication author details with the SEFA member array.


C:\Users\priya\OneDrive\Documents\CW_IR_7071CEM\data\output.json - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
output.json

```
{"doc_id": [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36
, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73
, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137
, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166,
167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196
, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225,
226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255
, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284,
285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314
, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337], "title": ["A bibliometric
review of the Waqf literature", "A note on COVID-19 instigated maximum drawdown in Islamic markets versus conventional counterparts", "Bank stock
valuation theories: do they explain prices based on theories?", "CEO Duality and Firm Performance: A Systematic Review and Research Agenda", "CEO
Financial Experience and Firms\u2019 Earnings Management in M&amp;A: the importance of Experience Specificity", "Cheating Behaviour among OPEC
Member-States and Oil Price Fairness and Stability: An Empirical Analysis", "Clinicians\u2019 informal acquisition of accounting literacy in UK
clinical commissioning groups", "Competing Institutional Logics and Power Dynamics in Islamic Financial Reporting Standardisation Projects",
"Corporate Governance and IFSB Standard-4: Evidence from Islamic Banks in Bangladesh", "Corporate hedging and the cost of equity capital",
"Corporate Social Responsibility &amp; Firm Efficiency: Evidence from Endogenous Cost Inefficiency Stochastic Frontier Analysis", "Corporate Social
Responsibility and Financial Performance of Product and Service-Based Firms Listed on London Stock Exchange", "Diffusion theory, transnational
antecedents and International Standards on Auditing adoption around the world", "Dissecting the effect of family business exposure on
entrepreneurial implementation intention", "Do Female Board Members Influence Corporate Social Responsibility Performance?", "Do non-communicable
diseases influence sustainable development in Sub-Saharan Africa? A panel autoregressive distributive lag model approach", "Editorial: New
developments in corporate governance research", "Evolution of research in finance over the last two decades \u2013 A topographical view",
"Financial contagion effects of major crises in African stock markets", "Future of education and research: a preliminary thematic analysis",
"Gender Diversity and Financial Statement Fraud", "IFRS 8 and the cost of capital in Europe", "Interbank liquidity risk transmission to large
emerging markets in crisis periods", "Kuala Lumpur - from a tin mining settlement to a neoliberal stronghold of Southeast Asia", "Measuring the
impact of corporate governance on non-financial reporting in the top HEIs worldwide", "Middle Eastern Extinctions: Building a Religious Motivation
for Species Protection", "Non\u2011executive employee ownership and financial reporting quality: evidence from Europe", "Price run-ups and insider
trading laws under different regulatory environments", "Smart city dynamics and multi-level management accounting: unfolding a case of sustainable
enterprise resource planning", "Socially responsible investment (SRI) Sukuk as a financing alternative for post COVID-19 development project",
"Sustainability accounting and reporting: Technical summary", "Sustainability accounting and reporting: an ablative reflexive thematic analysis of
climate crisis, conservative or radical reform paradigms", "The diffusion of innovation theory and the effects of IFRS adoption by multinational
corporations on capital market performance: a cross-country analysis", "The nexus between environmental and financial performance: Evidence from
gulf cooperative council banks", "The relative effectiveness of R&amp;D tax credits and R&amp;D subsidies: A comparative Meta-Regression Analysis",
"The rise and fall of institutional entrepreneurship in Islamic financial reporting standardisation projects", "Tourism And Exports: The Case of
Singapore", "Unveiling the black swan of the finance\u2010growth Nexus: Assumptions and preliminary evidence of virtuous and unvirtuous cycles",
"Why do Sukuks (Islamic Bonds) need a different pricing model?", "A global analysis of Private Investments in Public Equity", "A Historical
```

Fig 3: Result of crawled data in the json file

Pre-Processing: There are certain pre-processing steps implemented before starting the data indexing process and they are

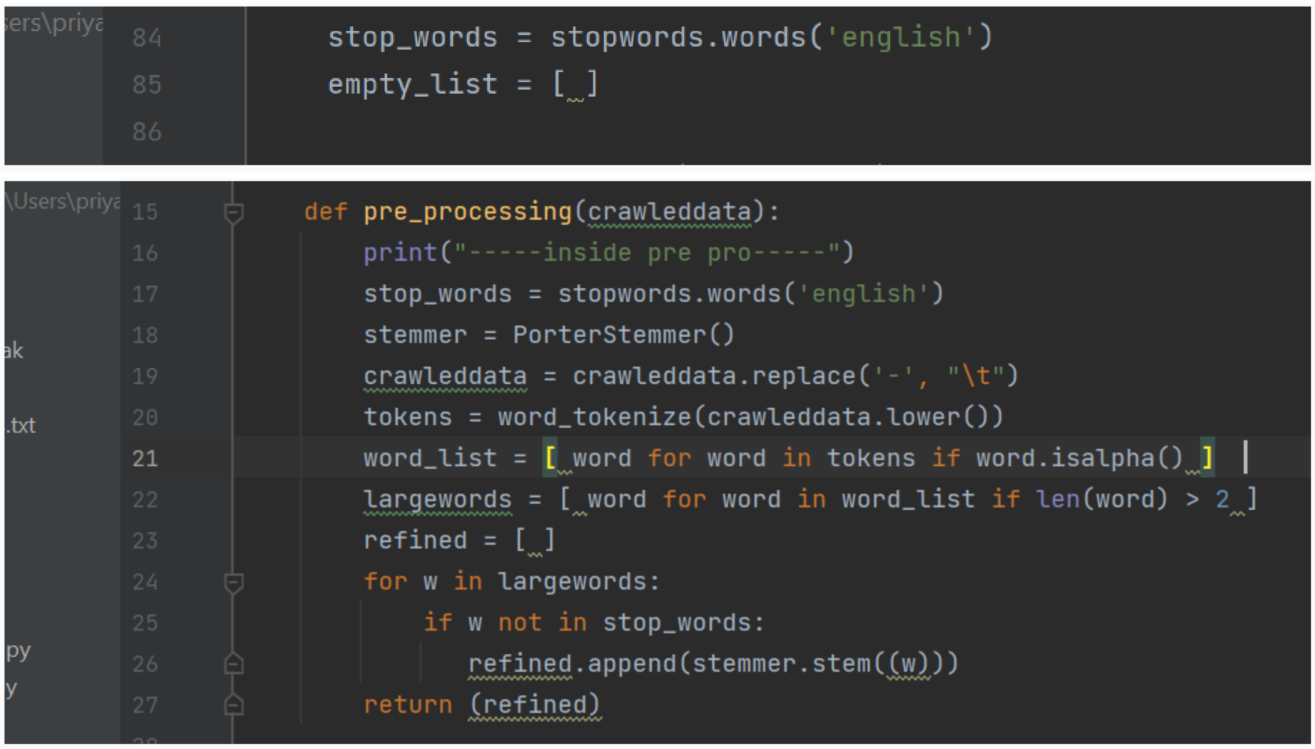
1. <u>Removing stop words:</u> There will be multiple stop words available in a word file. To make the key information more prominent in our text, we eliminate the low-level information which is not required to be indexed and needs to be eliminated from the search.

2. <u>Converting all characters into the lower case:</u> This is part of the text normalisation and is done to make sure that the case sensitivity does not treat similar terms differently.

3. <u>Word tokenization:</u> It is the process of splitting bigger texts into tokens, which will be helpful for easy classification and analysis.

4. <u>Stemming:</u> This process is used for the grouping of words into the same stem.

5. <u>Removing special characters:</u> The punctuations and hyphens are removed from the data to make it less affected while processing the data.

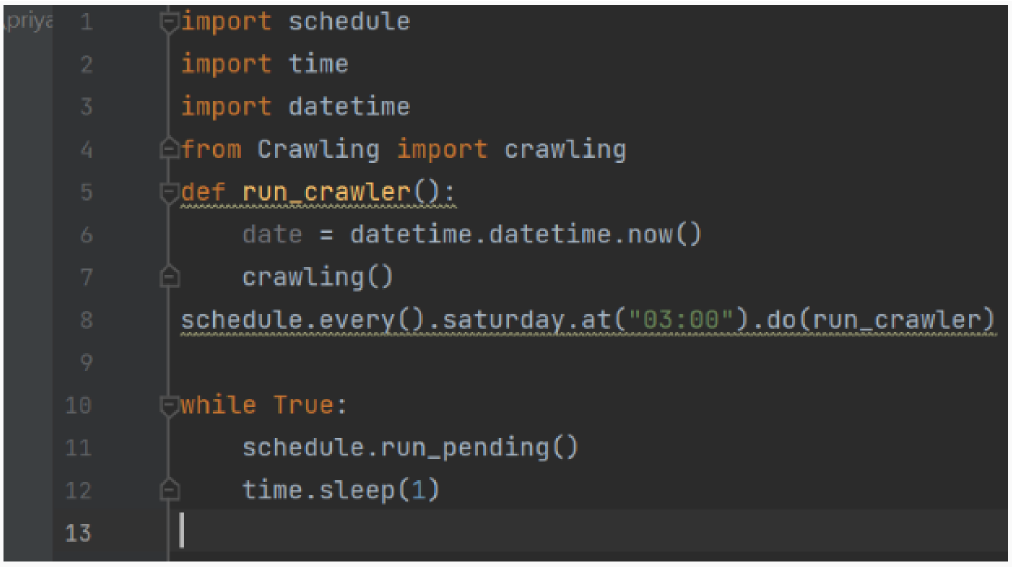


```python
    stop_words = stopwords.words('english')
    empty_list = [ ]
```

```python
def pre_processing(crawleddata):
    print("-----inside pre pro-----")
    stop_words = stopwords.words('english')
    stemmer = PorterStemmer()
    crawleddata = crawleddata.replace('-', "\t")
    tokens = word_tokenize(crawleddata.lower())
    word_list = [ word for word in tokens if word.isalpha() ]
    largewords = [ word for word in word_list if len(word) > 2 ]
    refined = [ ]
    for w in largewords:
        if w not in stop_words:
            refined.append(stemmer.stem((w)))
    return (refined)
```

Fig 4: Code for data pre-processing

Schedule: We have automated the crawling process using the ‘schedule’ library in python, the crawler is scheduled to run every Saturday early morning at 03.00



```python
import schedule
import time
import datetime
from Crawling import crawling
def run_crawler():
    date = datetime.datetime.now()
    crawling()
schedule.every().saturday.at("03:00").do(run_crawler)

while True:
    schedule.run_pending()
    time.sleep(1)

```

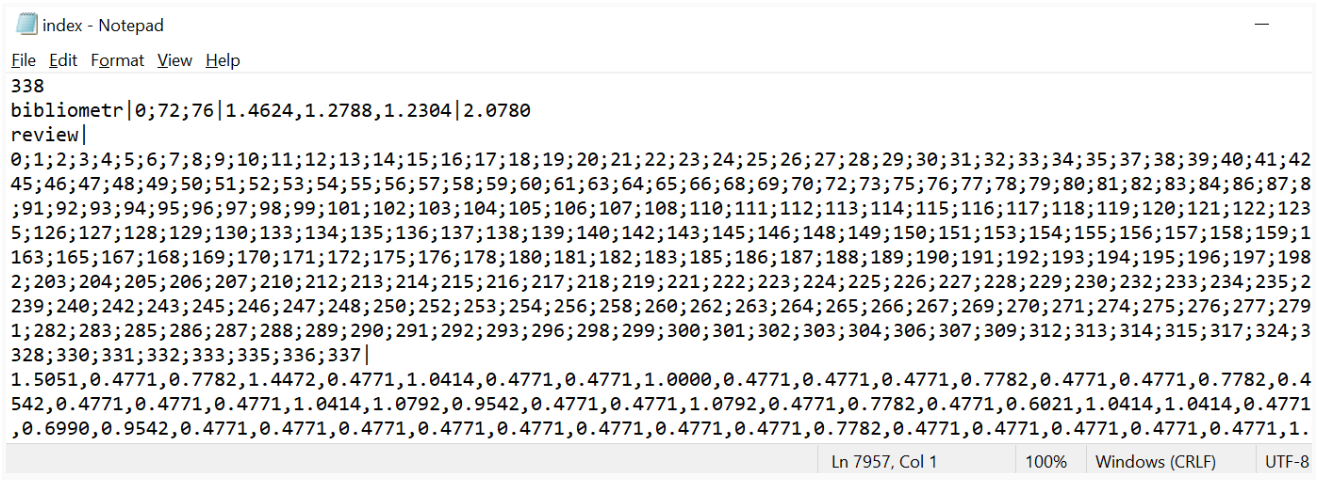
Fig 5: Scheduling the crawling

Data Indexing

The indexing process is implemented mainly to store crawled data in an organised way to retrieve the query results quickly. Without indexing, the search engines would need to go through each page of results to find the matching keywords and to extract the data.

In this study I have used the **inverted index** method to structure and store mapping from the created json file which contains the crawled data. We have implemented preprocessing techniques to the crawled data and applied TF-IDF vectorization technique to the preprocessed data. All the data from the input document is converted to tokens for the indexing and are stored along with the pointers and TFIDF values. These stored TFIDF values are then used for ranking during the query processing.

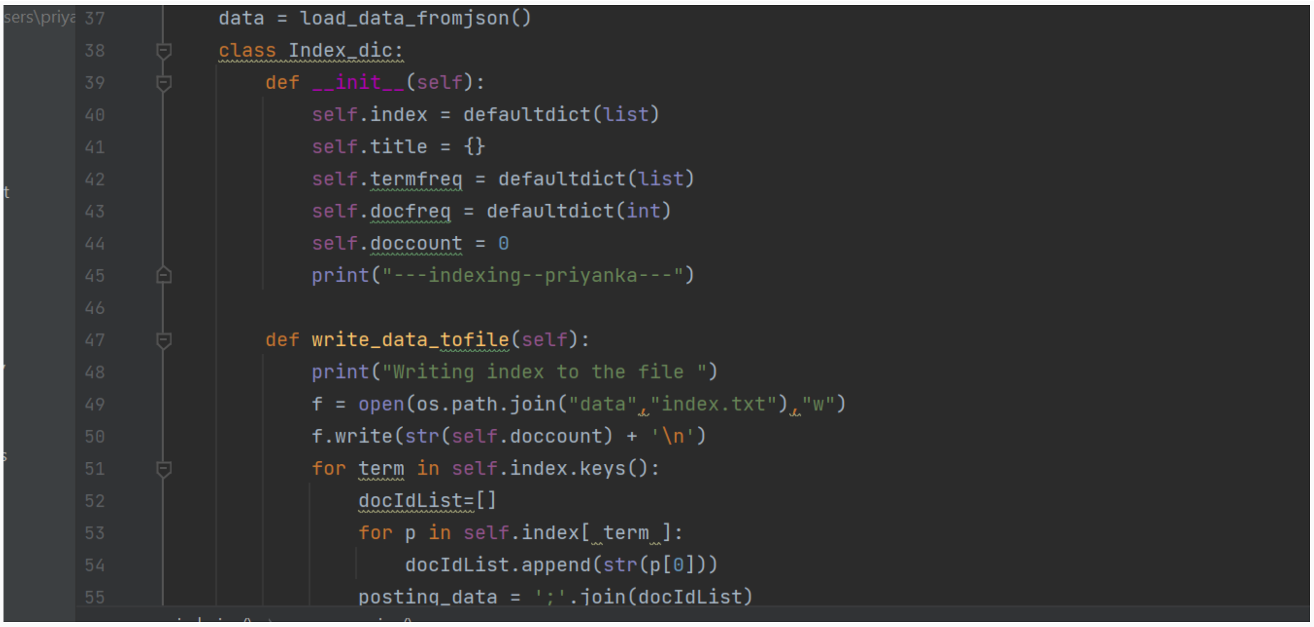
After the indexing and ranking is done, the result is stored as a text file with name index.txt and the file content is as given below.



```
338
bibliometr|0;72;76|1.4624,1.2788,1.2304|2.0780
review|
0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;37;38;39;40;41;42
45;46;47;48;49;50;51;52;53;54;55;56;57;58;59;60;61;63;64;65;66;68;69;70;72;73;75;76;77;78;79;80;81;82;83;84;86;87;8
;91;92;93;94;95;96;97;98;99;101;102;103;104;105;106;107;108;110;111;112;113;114;115;116;117;118;119;120;121;122;123
5;126;127;128;129;130;133;134;135;136;137;138;139;140;142;143;145;146;148;149;150;151;153;154;155;156;157;158;159;1
163;165;167;168;169;170;171;172;175;176;178;180;181;182;183;185;186;187;188;189;190;191;192;193;194;195;196;197;198
2;203;204;205;206;207;210;212;213;214;215;216;217;218;219;221;222;223;224;225;226;227;228;229;230;232;233;234;235;2
239;240;242;243;245;246;247;248;250;252;253;254;256;258;260;262;263;264;265;266;267;269;270;271;274;275;276;277;279
1;282;283;285;286;287;288;289;290;291;292;293;296;298;299;300;301;302;303;304;306;307;309;312;313;314;315;317;324;3
328;330;331;332;333;335;336;337|
1.5051,0.4771,0.7782,1.4472,0.4771,1.0414,0.4771,0.4771,1.0000,0.4771,0.4771,0.4771,0.7782,0.4771,0.4771,0.7782,0.4
542,0.4771,0.4771,0.4771,1.0414,1.0792,0.9542,0.4771,0.4771,1.0792,0.4771,0.7782,0.4771,0.6021,1.0414,1.0414,0.4771
,0.6990,0.9542,0.4771,0.4771,0.4771,0.4771,0.4771,0.4771,0.4771,0.4771,0.7782,0.4771,0.4771,0.4771,0.4771,0.4771,1.
```

Fig 6: Indexed file result

Code to implement the indexing is attached in the appendix and the code is written as given in the below figure.



```python
data = load_data_fromjson()
class Index_dic:
    def __init__(self):
        self.index = defaultdict(list)
        self.title = {}
        self.termfreq = defaultdict(list)
        self.docfreq = defaultdict(int)
        self.doccount = 0
        print("---indexing--priyanka---")

    def write_data_tofile(self):
        print("Writing index to the file ")
        f = open(os.path.join("data","index.txt"),"w")
        f.write(str(self.doccount) + '\n')
        for term in self.index.keys():
            docIdList=[]
            for p in self.index[ term ]:
                docIdList.append(str(p[0]))
            posting_data = ';'.join(docIdList)
```

Fig 7: Indexing code

As the crawler is scheduled to run every Saturday, the associated indexing also will be run along with it. The code is not written for the incremental indexing, the index file will be recreated every time after the crawler process implemented.

Query Processing

I have created a web interface "pywebio" library. The "pywebio" library is used to create simple python web application. I have used the "input" function to get the search term from the user. "Input" function basically gives a search box with two buttons "Submit" and "Reset". "Submit" button can be used to submit the search query and the "Reset" is used to clear the text entered in the text field.

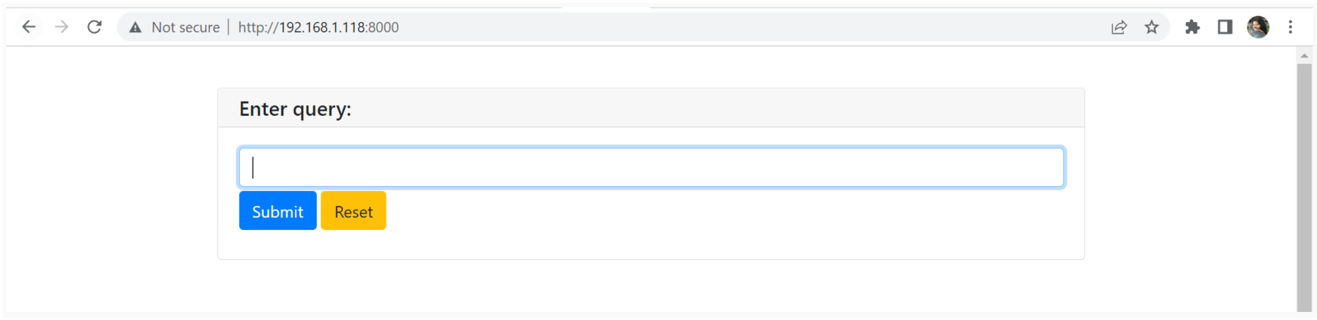
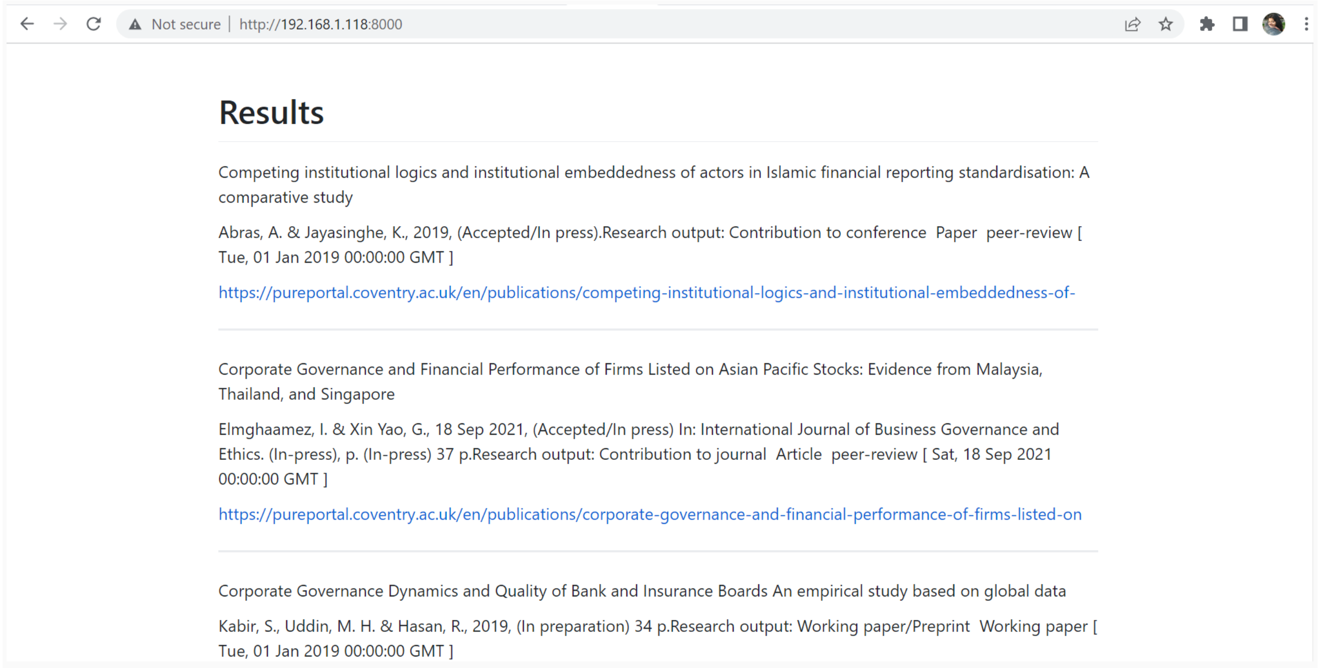


Fig 8: Web interface

Once the query is submitted, it will be taken through a number of pre-processing techniques like tokenization, eliminating special characters and stop words, stemming and converting all the entered characters into lower case.

In the next step, the entered query will be vectorized using TD-IDF technique. And the entered query will be compared with the content in the index file and similar terms. Similarities will be measured and the most relevant results will be returned and displayed in the search results page. Elastic seach techniques are not used in this course work, cosine similarity measure is used here as both the input query and the indexed data is vectorised.

The search result page will be presented like below:


Not secure | http://192.168.1.118:8000

Results

Competing institutional logics and institutional embeddedness of actors in Islamic financial reporting standardisation: A comparative study

Abras, A. & Jayasinghe, K., 2019, (Accepted/In press).Research output: Contribution to conference Paper peer-review [Tue, 01 Jan 2019 00:00:00 GMT]

https://pureportal.coventry.ac.uk/en/publications/competing-institutional-logics-and-institutional-embeddedness-of-

Corporate Governance and Financial Performance of Firms Listed on Asian Pacific Stocks: Evidence from Malaysia, Thailand, and Singapore

Elmghaamez, I. & Xin Yao, G., 18 Sep 2021, (Accepted/In press) In: International Journal of Business Governance and Ethics. (In-press), p. (In-press) 37 p.Research output: Contribution to journal Article peer-review [Sat, 18 Sep 2021 00:00:00 GMT]

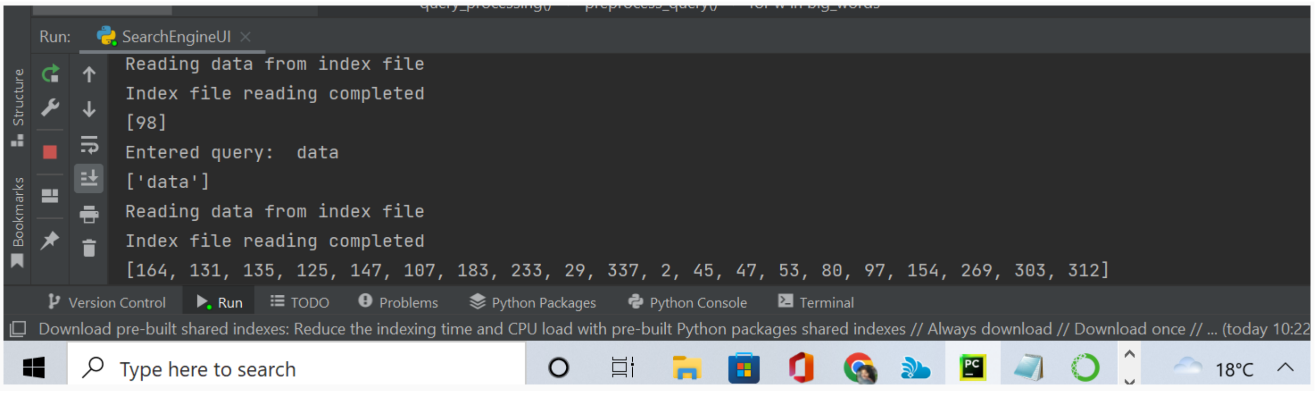
https://pureportal.coventry.ac.uk/en/publications/corporate-governance-and-financial-performance-of-firms-listed-on

Corporate Governance Dynamics and Quality of Bank and Insurance Boards An empirical study based on global data

Kabir, S., Uddin, M. H. & Hasan, R., 2019, (In preparation) 34 p.Research output: Working paper/Preprint Working paper [Tue, 01 Jan 2019 00:00:00 GMT]

Fig 9: Successful Search Result

We can use a single word and or multiple words in the search bar to retrieve results. The pre-processed term and the index file reading details can be checked from the command line results.

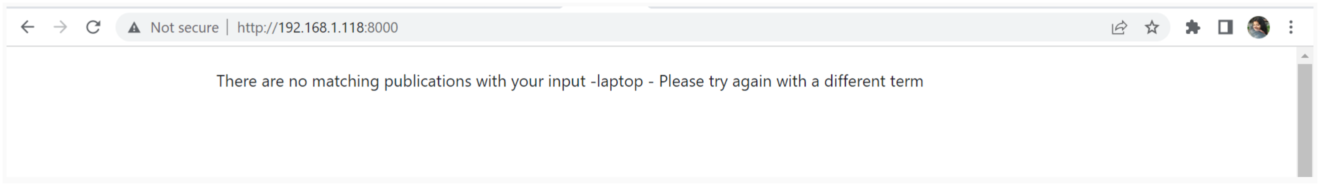

Run: SearchEngineUI

```
Reading data from index file
Index file reading completed
[98]
Entered query:  data
['data']
Reading data from index file
Index file reading completed
[164, 131, 135, 125, 147, 107, 183, 233, 29, 337, 2, 45, 47, 53, 80, 97, 154, 269, 303, 312]
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // ... (today 10:22

Fig 10: Command line result

The webapp also gives a message in the screen as well as in the command line if the entered query doesn't match with the crawled publications.



Not secure | http://192.168.1.118:8000

There are no matching publications with your input -laptop - Please try again with a different term

Fig 11: Result for the keyword with no matching document

Task 2. Document Clustering

Clusters are collections of related objects. The method used to divide the objects into these groups is called clustering. Clustering can be done to categorise a huge set of data to the required clusters. Clustering techniques are used in various industries to label huge data, which enables the users to access data with ease and process them.

In this task, I have used 109 lines of data to perform the document clustering. The data is stored in a csv file and are categorised into 'Politics', 'Health' and 'Sports'. The 1/3rd of used data is manually extracted from the BBC website and the remaining are extracted from the online kaggle repository. The data set has 2 columns - 'Category' and 'Doc_title'. Category column contains the labels (health, politics, sports) and the second column contains the news headlines extracted from various sources.

I have used python code and google colab IDE to implement the written code. Dataset is loaded into the google drive. Google drive is then mounted into the google colab to access the input data.

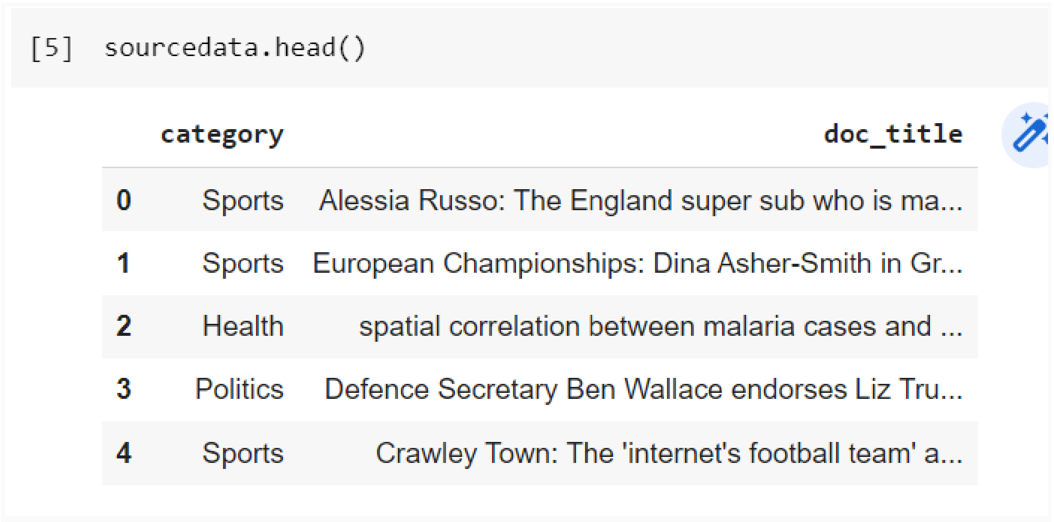
I have used the TF-IDF vectorisation technique for feature extraction. After feature extraction, we are preprocessing the data to remove stop_words.

I am using the K-means clustering model in the study to categorise the input documents. Since the documents are of 3 different labels, I have used the K number as 3 and the max iteration is set to 500.

I am using the hard clustering method in this study as it groups the data items to belong to a single cluster.

After the clustering model is implemented I am performing a test by passing a sample text for the program to predict the best fit cluster.

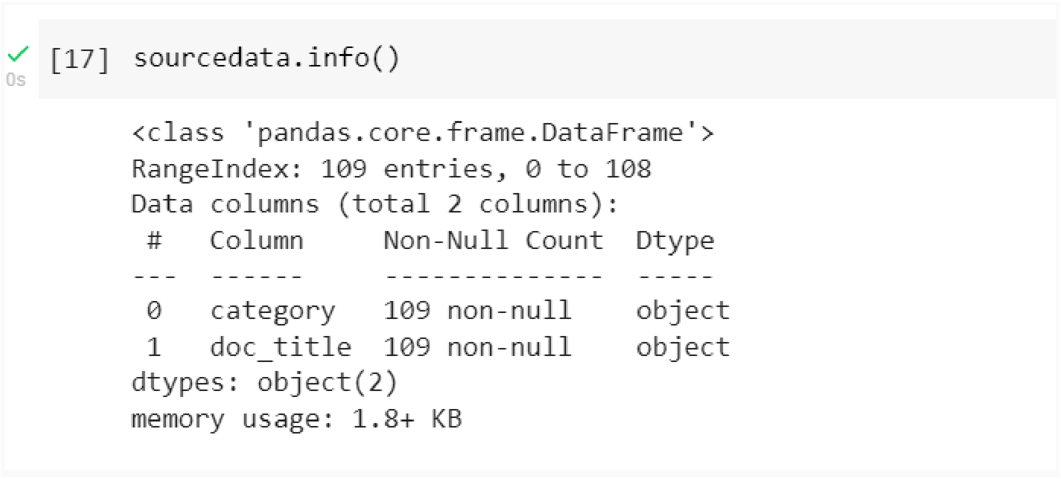
Code used to implement the document clustering is attached in the appendix.



```
[5] sourcedata.head()
```

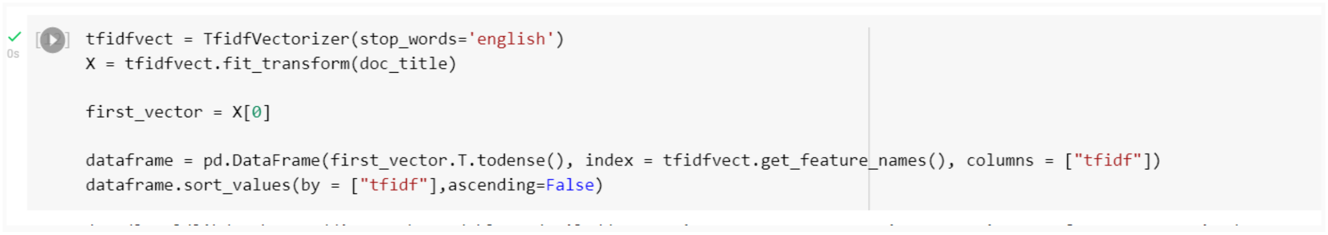
	category	doc_title
0	Sports	Alessia Russo: The England super sub who is ma...
1	Sports	European Championships: Dina Asher-Smith in Gr...
2	Health	spatial correlation between malaria cases and ...
3	Politics	Defence Secretary Ben Wallace endorses Liz Tru...
4	Sports	Crawley Town: The 'internet's football team' a...

Fig 12: Top 5 lines from the input file



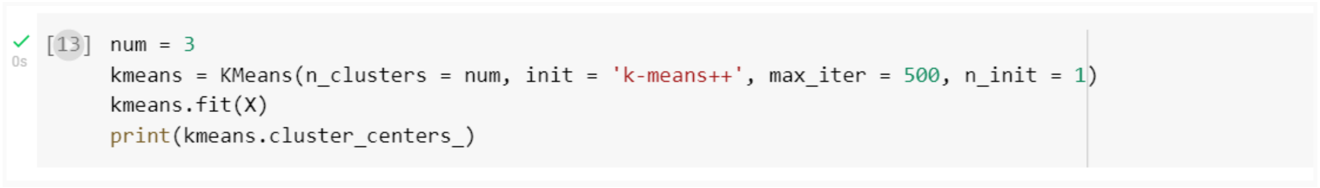
```
[17] sourcedata.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 109 entries, 0 to 108
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   category   109 non-null    object
 1   doc_title  109 non-null    object
dtypes: object(2)
memory usage: 1.8+ KB
```

Fig 13: Information about the used dataset



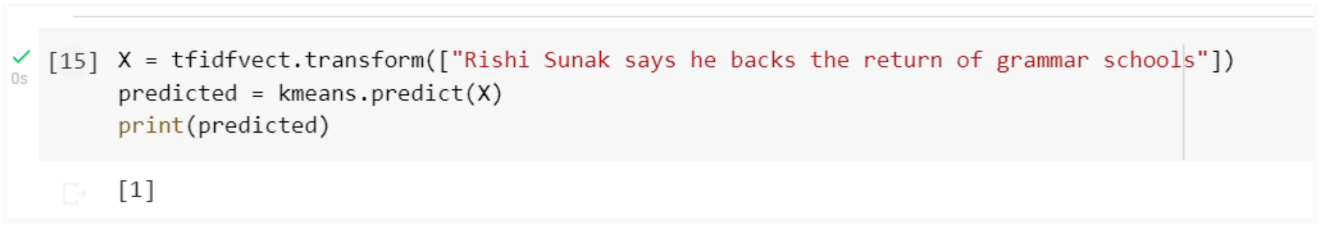
```
tfidfvect = TfidfVectorizer(stop_words='english')
X = tfidfvect.fit_transform(doc_title)

first_vector = X[0]

dataframe = pd.DataFrame(first_vector.T.todense(), index = tfidfvect.get_feature_names(), columns = ["tfidf"])
dataframe.sort_values(by = ["tfidf"],ascending=False)
```

Fig 14: TF-IDF vectorization



```
[13] num = 3
     kmeans = KMeans(n_clusters = num, init = 'k-means++', max_iter = 500, n_init = 1)
     kmeans.fit(X)
     print(kmeans.cluster_centers_)
```

Fig 15: Implementing K-Means clustering model



```
[15] X = tfidfvect.transform(["Rishi Sunak says he backs the return of grammar schools"])
     predicted = kmeans.predict(X)
     print(predicted)

     [1]
```

Fig 16: Predicting the cluster with the sample document input

APPENDIX

Written code is uploaded in the github repository, and can be accessed using below link.

https://github.com/priyankaharidasnk/IR

REFERENCES

1. Medium. 2022. *Text Preprocessing in Natural Language Processing using Python*. [online] Available at: <https://towardsdatascience.com/text-preprocessing-in-natural-language-processing-using-python-6113ff5decd8>
2. Scrapingbee.com. 2022. *Web Scraping with Python: Everything you need to know (2022)*. [online] Available at: <https://www.scrapingbee.com/blog/web-scraping-101-with-python/>
3. En.wikipedia.org. 2022. *Beautiful Soup (HTML parser) - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Beautiful_Soup_(HTML_parser)>
4. En.wikipedia.org. 2022. *Web crawler - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Web_crawler>
5. Deepcrawl. 2022. *What is Search Engine Indexing & How Does it Work? - Deepcrawl*. [online] Available at: <https://www.deepcrawl.com/knowledge/technical-seo-library/search-engine-indexing/>
6. Medium. 2022. *Let's build a search engine with python*. [online] Available at: <https://blog.devgenius.io/lets-build-a-search-engine-with-python-3f8dd3320210>
7. Moz. 2022. *How Search Engines Work: Crawling, Indexing, and Ranking - Beginner's Guide to SEO*. [online] Available at: <https://moz.com/beginners-guide-to-seo/how-search-engines-operate>

8. Tutorialspoint.com. 2022. *Python - Word Tokenization*. [online] Available at: <https://www.tutorialspoint.com/python_data_science/python_word_tokenization.htm#:~:text=Word%20tokenization%20is%20the%20process,for%20a%20particular%20sentiment%20etc.>
9. En.wikipedia.org. 2022. *Inverted index - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Inverted_index>
10. Kaggle.com. 2022. *Clustering documents with TFIDF and KMeans*. [online] Available at: <https://www.kaggle.com/code/jbencina/clustering-documents-with-tfidf-and-kmeans/notebook>
11. Brandonrose.org. 2022. *Document Clustering with Python*. [online] Available at: <http://brandonrose.org/clustering>