Text-Mining on EPSRC (Engineering and Physical Sciences Research Council) grant abstracts

Sai Sudharshan Govindarajan
956299

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Bachelors in Science

Bachelor of Science
Doctor of Philosophy

Department of Computer Science
Swansea University

May 8, 2020

Contents

Abstract

It is often said that “Words are often tied to money”. This means that for every word that is present in a grant abstract it shows how much grant funding an abstract should ideally be provided with. In this project, I will analyse data from EPSRC grant abstracts and show how text analysis techniques and derive meaningful and insightful information can be used to analyse the amount of grant funding that should be provided for the respective project.

Introduction

This project is based on Natural Language Processing, a very well-known concept in the Computing world, yet it is at its experimental stages of development. I will be following the programming principles of this concept very closely to develop a text-mining (Linguamatics, 2019) software which will perform successful text analysis on EPSRC grant abstracts. I will be doing this to derive insightful information from them. The information gathered will contain aspects of the word count, the quality of writing though capture of the use of words containing a higher level of verbal confidence and spotting the use of complex writing styles from the abstracts. This information would provide insights on the grant funding decisions for a grant administrator.

For this project, I am using Python as my main language of scripting and I am writing this in PyCharm which is an IDE that handles all the language libraries and language processing itself very well. I decided on this as it is an open-source language which means that it is very easily accessible, and it has a large provision of support libraries which have aided me in completing my project. In Python, I am using the NLTK library which is also known as the Natural Language Tool Kit. It is used in Natural Language Processing (NLP) (Brownlee, 2019) which is a section of artificial inelegance that aims to help computers better understand the human natural language by, automated reading of textual data, spotting patterns in textual writing styles and derive meaningful information from them, in this case the grant abstracts.

The dataset I’m using, will be taken from the UK Research and Innovation website which contains all the grant abstract data which have been submitted to the EPSRC. This dataset more specifically contains the title of each project and their URL path which leads to the page containing the abstract for the respective to title. This dataset is stored as a Comma-separated file format or ‘.csv’ which means that the file uses comma to separate values. This dataset will be used throughout the programming and testing phases for my application.

In order to collect the grant abstract data from the webpage, I am using a python library named Beautiful Soup. This library has the main purpose of pulling data from an html page. As the page containing the abstract data is an HTML page, I will be using this library to scrape the data and perform text analysis on the scraped data.

Not only am I using Beautiful Soup library, but I also am using the Pandas library as well which helps in converting the dataset which is stored as a .csv file format into python objects with rows and columns called a Data Frame. This looks very similar to a table in a statistical software and will help me when sorting the data for use in Latent Dirichlet Allocation.

Moreover, I am using Latent Dirichlet Allocation (LDA) algorithm for modelling the project titles in the dataset by common topics, which will then be used for text mining. LDA is a form of unsupervised machine learning technique used for topic modelling. This is a statistical technique that can capture and extract primary themes or topics from the grant abstracts. I am using this algorithm on the grant abstracts in the dataset to find the most common topics as it also fares well as a dimensionality reduction algorithm. This means that it aims to reduce the issues of overfitting and longer training times by reducing the number of features or dimensions.

Motivations

Additionally, being able to successfully reduce human error would also be the motivation behind the project. Physically reading and understanding texts from grant abstracts is something that's been done for years and now even more so as many more new ideas begin to emerge. However, this can be overwhelming for a grant administrator as the grant abstracts are usually very worded and they would have to adapt to the diversity of the language being used. As this puts both the EPSRC and the person being provided the grant liable to a lot of money purely relying on lack of human error, with my text mining application, I will be using the advancements in machine learning and a greater understanding of Artificial Intelligence (AI) to my advantage and create a software which will aim to reduce human error by performing automatic text analysis on multiple grant abstracts.

Finally, the motivation of my project would be to aim to also reduce time-consumption by using text mining or text analysis. This is the process of transforming unstructured text into meaningful and actionable information. Manually reading through grant abstracts can be a time-consuming task for the grant administrator. In order to reduce time consumption, my tool will be performing sequential text analysis on each abstract. The abstracts will be taken from the EPSRC website which have been converted into data sets to be fed into my text mining tool for simultaneous text analysis of multiple abstract copra.

Aims

For this project, I aim to write an application that can perform successful text analysis on a single grant abstract. Granted, this is a very modest aim to meet but, by doing this, I will be able to better fix any bugs or errors that occur in early stages itself. Doing this would give me an idea on how the application processes the grant information given to it so that I can adapt it to perform text analysis on multiple grant abstracts.

Later, I aim to program this application to perform the same successful text analysis on multiple grant abstracts. By doing this I will be saving time by, only needing to prepare the dataset once in-order to feed it into the application rather than preparing each individual grant abstract to be analysed. Doing this would show me the way the application handles the dataset; the errors and bugs caused by the data load and derive insightful information from interpreting the dataset.

Finally, my application aims to derive crucial and insightful information from the dataset which will be used in grant funding decisions. It aims to derive information on the word count, the amount of misspelled words if any and if the information contains the right punctuation and grammar from the grant abstracts in the dataset. Additionally, my application aims to differentiate the well-written grant abstracts from the rest by spotting the use of words that contain a higher level of verbal confidence.

By doing this, I aim to gain experience on new skills and, knowledge on the topics explored throughout this project. This will include the concept of Natural Language Processing and understanding and programming Latent Dirichlet Allocation algorithm in Python. I also aim to build up on my existing knowledge of programming such as applying my existing knowledge of logic and machine-learning in Java to Python when writing my application.

Related Works

Latent Semantic Analysis

Latent Semantic Analysis r (Matveeva and Levow, 2005) or LSA for short, is a mathematical method developed for improving the accuracy on information retrieval. This is achieved by use of a technique called Singular Value Decomposition which is used for scanning unstructured data within multiple documents and spot relationships between the topics present in them.

In this research paper, they combine the graph-based dimensionality reduction algorithm which is intended to represent high dimensional data into a low dimensionality space with a corpus-based association with Generalised Latent Semantic Analysis.

But as the results are collected, the paper goes on to summarise that Generalised LSA did not act as an efficient way of information retrieval because it was too sensitive to the parameters of the neighbouring graphs. Contrarily, the paper also informs that more analysis on many different documents is required for a more definite answer on the use of Generalised Latent Semantic Analysis.

Latent Dirichlet Allocation

Latent Dirichlet Allocation (M.Blei, Y.Ng and I.Jordan, 2003) or LDA for short, is generative probabilistic model which is mostly used in Natural Language Processing with topic modelling. This is a type of unsupervised learning which shows topic probabilities from a mixture of topics in a copra. The concept of LDA is thoroughly guided by the two principles which is as follows:

- Every document is a mixture of topics: If there is a document primarily focused on cars, then the document may contain several words from several topics in their particular proportions.

- Every topic is a mixture of words: if we chose two topics, for example, carts and helicopters. The topic about cars will have words such as 'Bonnet', 'Door' and 'Windscreen' whereas the topic about helicopters would have words such as 'Rudder', 'Blades', 'Flight' but, more importantly words such as 'Price' and 'Wheels' would be shared between the topics.

The research paper by (M.Blei, Y.Ng and I.Jordan, 2003) shows LDA as a mathematical method for estimating both these principles at the same time by finding the mixture of word-tokens that are associated with each topic. The research paper also finds the mixture of topics that describe each document. It then, then compares the LDA with unigram model and the probabilistic semantic indexing. Upon comparing both the models, the research paper concludes by saying that LDA, when compared to LSA is a better topic modelling and indexing model as it follows a straightforward dimensionality reduction technique using the de Finetti's representation algorithm (Diaconis, 1988).

Analysing NIH funding patterns over time with statistical text analysis

In this research paper(Park et al., 2016), National Cancer Institute which is a part of National Institute of Health in the United States of America performs text analysis by using statistical text mining techniques on data extracted from the NIH RePORTER online system for a total period of 20 years, from 1994 to 2013. They use Research Categorization and Disease Classification (RCDC) labels, project titles, project abstracts and the annual funding amounts to create a quantitative picture of how the funding for cancer has changed over the years since 1994.

The study was conducted using a combination of Labelled Latent Dirichlet Allocation topic modelling and logistic regression for each document as the logistic classifiers to analyse the NIH funding. The grant funding data was extracted and, upon performing statistical text analysis revealed to have insightful information on the funding pattern such as Brain Cancer and Lung Cancer for which there was a consistent and sustained increase in funding over the 20-year period.

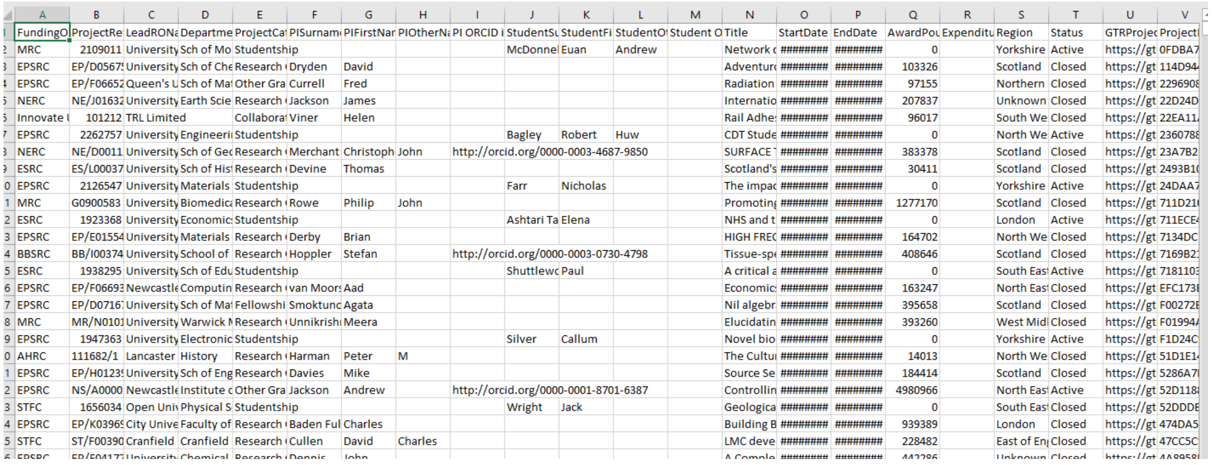
Finally, Upon performing successful text analysis on the data, the paper goes on to conclude that the grant funding decisions shouldn't only be based off the grant abstract content but, other factors as well, such as the scientific articles that were followed from the funded projects and other relevant articles.

Compared to this paper, my project also uses Latent Dirichlet Allocation for topic modelling in a similar way. By this I mean that, it also aims to perform statistical text analysis on the grant abstracts to output insightful results. But unlike this paper where they find an exact funding pattern using LDA and logistic regression, my project aims to analyse the likelihood of topic occurrence with the important words present in each topic based off which decide the area of research that is provided the most funding for.

Implementation

Preparing the Dataset

The dataset is taken from the UK Research and Innovation website consisting of 102,882 items and was initially visualised in Microsoft Excel, which is as seen in [Figure 1]. It consists of all the project topics and their respective information such as the submission date, funding provided and the URL to the abstracts along with the funding organization, including the EPSRC. The dataset was downloaded in a .csv format for it to be processed and analysed in python.

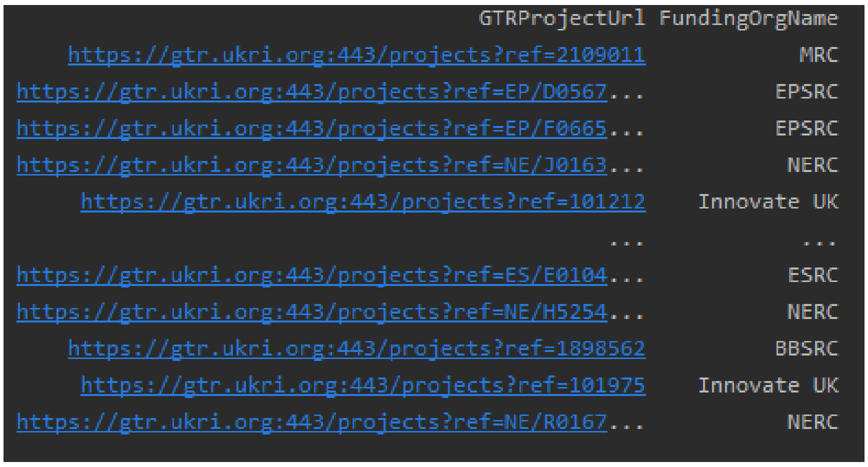


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	FundingO	ProjectRe	LeadRONa	Departme	ProjectCat	PISurname	PIFirstNar	PIOtherNa	PI ORCID i	StudentSu	StudentFi	StudentOt	Student O	Title	StartDate	EndDate	AwardPou	Expenditu	Region	Status	GTRProjec	ProjectI
2	MRC	2109011	University	Sch of Mo	Studentship					McDonnel	Euan	Andrew		Network c	########	########	0		Yorkshire	Active	https://gt	0FDBA7
3	EPSRC	EP/D0567!	University	Sch of Che	Research (	Dryden	David							Adventur	########	########	103326		Scotland	Closed	https://gt	114D94
4	EPSRC	EP/F06652	Queen's U	Sch of Mat	Other Gra	Currell	Fred							Radiation	########	########	97155		Northern	Closed	https://gt	2296908
5	NERC	NE/J01632	University	Earth Scie	Research (	Jackson	James							Internatio	########	########	207837		Unknown	Closed	https://gt	22D24D
6	Innovate U	101212	TRL Limited		Collaborat	Viner	Helen							Rail Adhe	########	########	96017		South We	Closed	https://gt	22EA11
7	EPSRC	2262757	University	Engineeri	Studentship					Bagley	Robert	Huw		CDT Stude	########	########	0		North We	Active	https://gt	2360788
8	NERC	NE/D0011	University	Sch of Geo	Research (	Merchant	Christoph	John	http://orcid.org/0000-0003-4687-9850					SURFACE T	########	########	383378		Scotland	Closed	https://gt	23A7B2
9	ESRC	ES/L00037	University	Sch of Hist	Research (	Devine	Thomas							Scotland's	########	########	30411		Scotland	Closed	https://gt	2493B1
10	EPSRC	2126547	University	Materials	Studentship					Farr	Nicholas			The impac	########	########	0		Yorkshire	Active	https://gt	24DAA7
11	MRC	G0900583	University	Biomedica	Research (	Rowe	Philip	John						Promoting	########	########	1277170		Scotland	Closed	https://gt	711D21
12	ESRC	1923368	University	Economics	Studentship					Ashtari Ta	Elena			NHS and t	########	########	0		London	Active	https://gt	711ECE
13	EPSRC	EP/E01554	University	Materials	Research (	Derby	Brian							HIGH FREQ	########	########	164702		North We	Closed	https://gt	7134DC
14	BBSRC	BB/I00374	University	School of	Research (	Hoppler	Stefan		http://orcid.org/0000-0003-0730-4798					Tissue-spe	########	########	408646		Scotland	Closed	https://gt	7169B2
15	ESRC	1938295	University	Sch of Edu	Studentship					Shuttlewo	Paul			A critical a	########	########	0		South East	Active	https://gt	7181103
16	EPSRC	EP/F06693	Newcastle	Computin	Research (	van Moors	Aad							Economics	########	########	163247		North East	Closed	https://gt	EFC173E
17	EPSRC	EP/D0716	University	Sch of Mat	Fellowshi	Smoktunc	Agata							Nil algebr	########	########	395658		Scotland	Closed	https://gt	F00272E
18	MRC	MR/N0101	University	Warwick M	Research (	Unnikrish	Meera							Elucidatin	########	########	393260		West Midl	Closed	https://gt	F01994A
19	EPSRC	1947363	University	Electronic	Studentship					Silver	Callum			Novel bio	########	########	0		Yorkshire	Active	https://gt	F1D24C
20	AHRC	111682/1	Lancaster	History	Research (	Harman	Peter	M						The Cultu	########	########	14013		North We	Closed	https://gt	51D1E1
21	EPSRC	EP/H0123!	University	Sch of Eng	Research (	Davies	Mike							Source Se	########	########	184414		Scotland	Closed	https://gt	5286A7
22	EPSRC	NS/A0000	Newcastle	Institute o	Other Gra	Jackson	Andrew		http://orcid.org/0000-0001-8701-6387					Controllin	########	########	4980966		North East	Active	https://gt	52D118
23	STFC	1656034	Open Univ	Physical S	Studentship					Wright	Jack			Geologica	########	########	0		South East	Closed	https://gt	52DDDE
24	EPSRC	EP/K0396!	City Unive	Faculty of	Research (	Baden Ful	Charles							Building B	########	########	939389		London	Closed	https://gt	474DA5
25	STFC	ST/F00390	Cranfield	Cranfield	Research (	Cullen	David	Charles						LMC deve	########	########	228482		East of En	Closed	https://gt	47CC5C
26	EPSRC	EP/F0417	University	Chemical	Research (	Dennis	John							A Comple	########	########	442286		Unknown	Closed	https://gt	4A8958

Figure 1: The Dataset

To apply the Latent Dirichlet Allocation algorithm on the abstracts, I had to load the dataset in python as objects. In order to do this, I used the Pandas library (Mckinney, Wes,2011), which converts the .csv file into python objects called Data Frame. Upon doing this, I had to 'clean out' the data to only show relevant columns. This included, the Funding organization name and the link to the project abstracts.

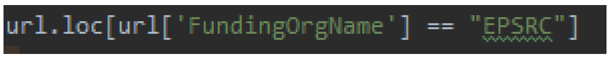
My thinking behind this was, if I were to drop the other rows, then by using the links to the abstracts, I will be able to access data for each project. This includes details such as the abstract itself and the funding provided for the project by their respective funding organization, which is as shown in [Figure 2].



```
                            GTRProjectUrl FundingOrgName
     https://gtr.ukri.org:443/projects?ref=2109011            MRC
https://gtr.ukri.org:443/projects?ref=EP/D0567...          EPSRC
https://gtr.ukri.org:443/projects?ref=EP/F0665...          EPSRC
https://gtr.ukri.org:443/projects?ref=NE/J0163...           NERC
      https://gtr.ukri.org:443/projects?ref=101212    Innovate UK
                                               ...            ...
https://gtr.ukri.org:443/projects?ref=ES/E0104...           ESRC
https://gtr.ukri.org:443/projects?ref=NE/H5254...           NERC
     https://gtr.ukri.org:443/projects?ref=1898562          BBSRC
      https://gtr.ukri.org:443/projects?ref=101975    Innovate UK
https://gtr.ukri.org:443/projects?ref=NE/R0167...           NERC
```

Figure 2: Dataframe showing the links to abstracts and their respective Funding Organization names

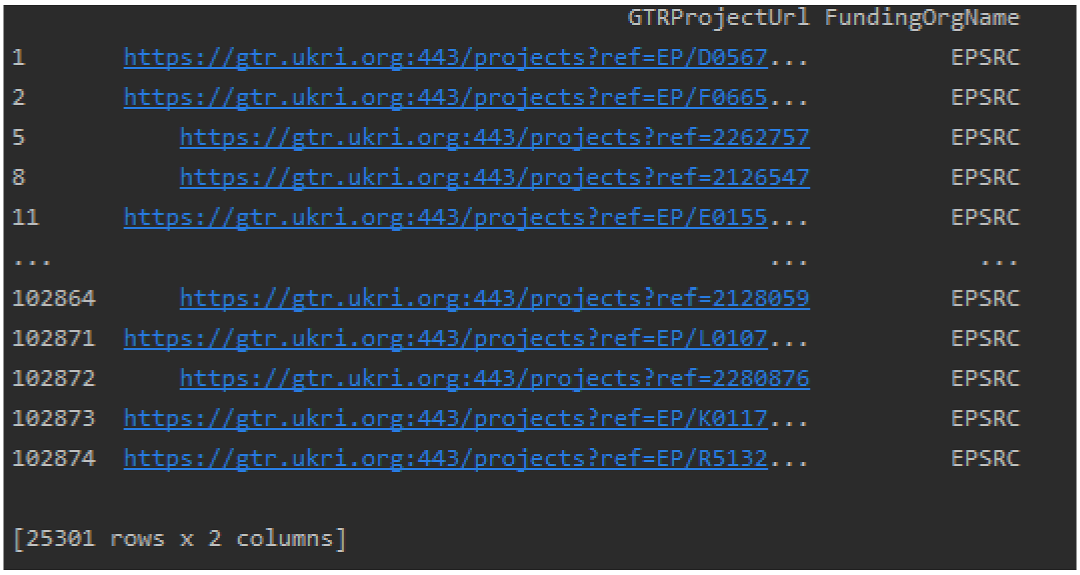
As shown in [Figure 2], I realized that the dataset contained project details by many different organizations including the EPSRC. Now, as I want to perform analysis on the grant abstracts by the EPSRC only I had to, show only the links to the projects funded by the EPSRC alone and traverse through them in order to scrape the data off them.



```
url.loc[url['FundingOrgName'] == "EPSRC"]
```

Figure 3: implementation of the .loc function

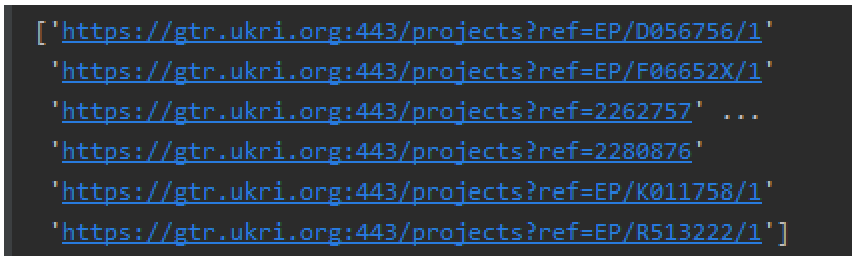
By using the 'loc' feature in Pandas as shown in [Figure 3] and specifying it to only show data by the EPSRC, I was able to narrow the dataset to only show links by the EPSRC. Now, this only consisted of 25,301 objects as seen in [Figure 4], which was a better number to handle comparing to the previous 102,882 objects.



```
                                           GTRProjectUrl FundingOrgName
1       https://gtr.ukri.org:443/projects?ref=EP/D0567...          EPSRC
2       https://gtr.ukri.org:443/projects?ref=EP/F0665...          EPSRC
5           https://gtr.ukri.org:443/projects?ref=2262757          EPSRC
8           https://gtr.ukri.org:443/projects?ref=2126547          EPSRC
11      https://gtr.ukri.org:443/projects?ref=EP/E0155...          EPSRC
...                                                   ...            ...
102864      https://gtr.ukri.org:443/projects?ref=2128059          EPSRC
102871  https://gtr.ukri.org:443/projects?ref=EP/L0107...          EPSRC
102872      https://gtr.ukri.org:443/projects?ref=2280876          EPSRC
102873  https://gtr.ukri.org:443/projects?ref=EP/K0117...          EPSRC
102874  https://gtr.ukri.org:443/projects?ref=EP/R5132...          EPSRC

[25301 rows x 2 columns]
```

Figure 4: Dataframe showing only the links to the abstracts sent to EPSRC and the funding organization names

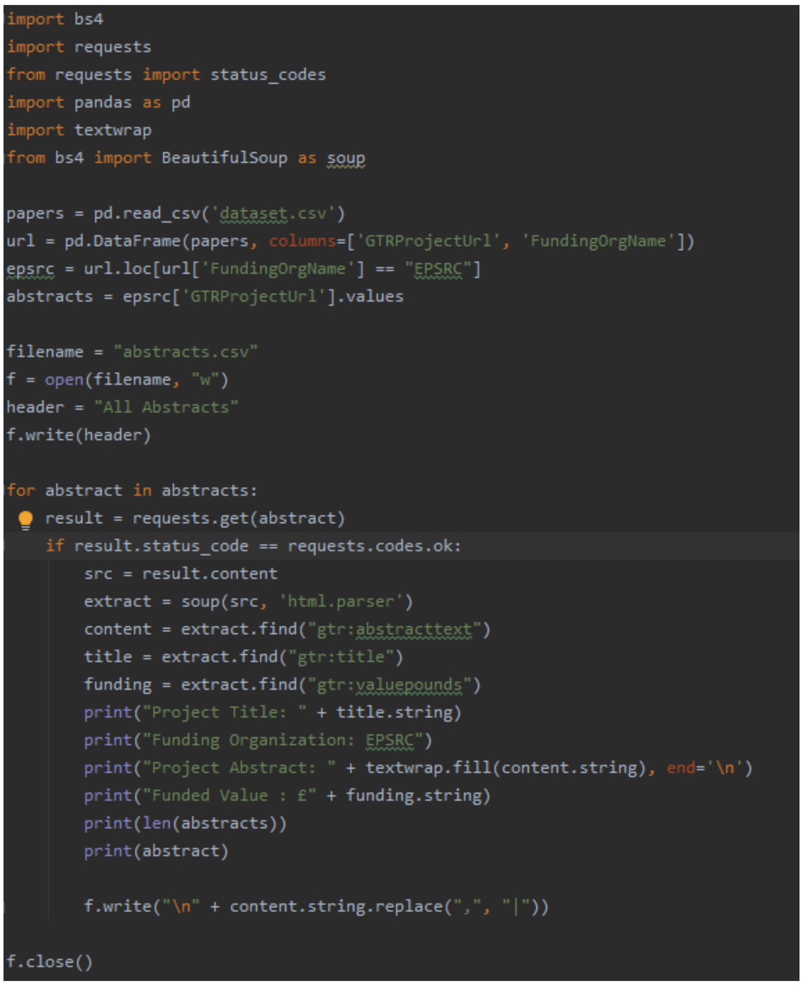
For me to perform automatic scraping on the links with ease in the DataFrame, I had to convert it into a Python array of links as shown in [Figure 5]. Upon doing this, I specified my program by writing a for-loop which iterates through the links in the array. When iterating through the array, the program threw some 'NoneType' errors as some of the links did not lead to abstract page but, into an error page. To solve this, I added an if-statement within the for-loop which's purpose is to ignore all the error links with status-code 500 and gets content from the links that work, specifically with links that produce the status-code 200.



```
['https://gtr.ukri.org:443/projects?ref=EP/D056756/1'
 'https://gtr.ukri.org:443/projects?ref=EP/F06652X/1'
 'https://gtr.ukri.org:443/projects?ref=2262757' ...
 'https://gtr.ukri.org:443/projects?ref=2280876'
 'https://gtr.ukri.org:443/projects?ref=EP/K011758/1'
 'https://gtr.ukri.org:443/projects?ref=EP/R513222/1']
```

Figure 5: showing the array of links

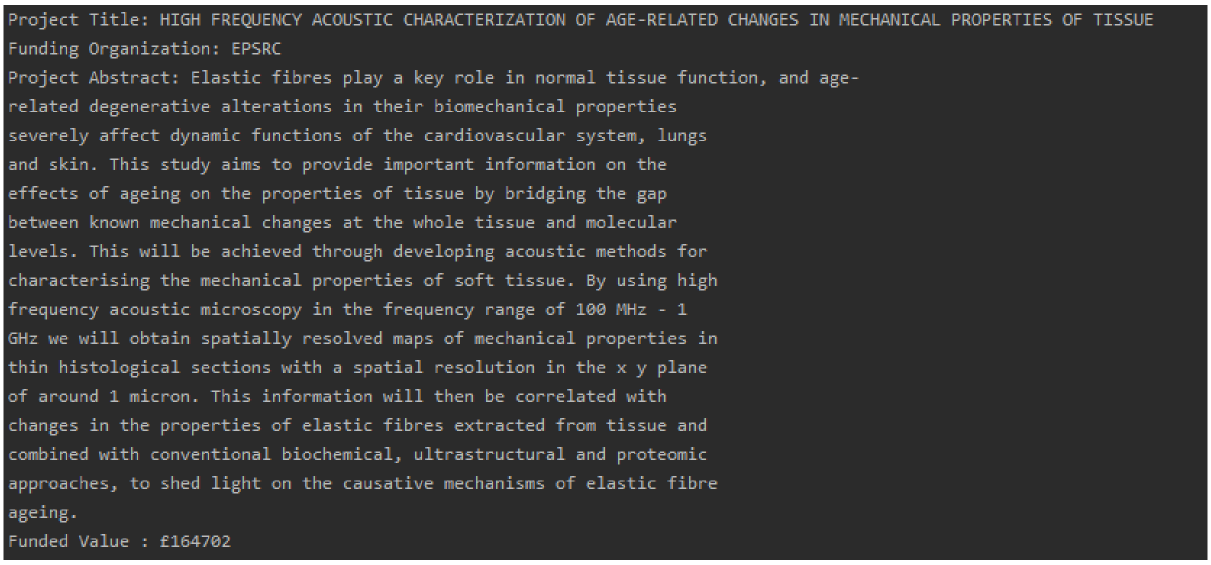
Now my strategy was to, extract the data present within the links, specifically the abstracts within the links that produced the status-code 200. In order to do this, I used the Beautiful Soup (Wieringa, 2012) which is a python library used for web-scraping. When running the program shown in [Figure 6], the content when extracted was to be written to another .csv file which only consisted of the abstracts. I decided on this method as, the data format required when running the Latent Dirichlet Allocation algorithm had to be in a list of documents in a .csv format.



```
import bs4
import requests
from requests import status_codes
import pandas as pd
import textwrap
from bs4 import BeautifulSoup as soup

papers = pd.read_csv('dataset.csv')
url = pd.DataFrame(papers, columns=['GTRProjectUrl', 'FundingOrgName'])
epsrc = url.loc[url['FundingOrgName'] == "EPSRC"]
abstracts = epsrc['GTRProjectUrl'].values

filename = "abstracts.csv"
f = open(filename, "w")
header = "All Abstracts"
f.write(header)

for abstract in abstracts:
    result = requests.get(abstract)
    if result.status_code == requests.codes.ok:
        src = result.content
        extract = soup(src, 'html.parser')
        content = extract.find("gtr:abstracttext")
        title = extract.find("gtr:title")
        funding = extract.find("gtr:valuepounds")
        print("Project Title: " + title.string)
        print("Funding Organization: EPSRC")
        print("Project Abstract: " + textwrap.fill(content.string), end='\n')
        print("Funded Value : £" + funding.string)
        print(len(abstracts))
        print(abstract)

        f.write("\n" + content.string.replace(",", "|"))

f.close()
```

Figure 6: Showing the implementation of the Web-Scraping program

Upon running the program shown in [Figure6], I was resulted with the project title, the funding organization, which is the EPSRC, the project abstracts and the funded value for the project. This is exactly how I predicted for the program to run and successfully scrape the data from the project abstract's links. Additionally, with resulting the abstracts to the python console as shown in [Figure 7], I also wrote all the abstracts without any other information such as the project title and the funding value for the project as I felt that such features to the data would produce extra noise when running LDA algorithm as the goal is to perform topic modelling on the text data from the abstracts.



```
Project Title: HIGH FREQUENCY ACOUSTIC CHARACTERIZATION OF AGE-RELATED CHANGES IN MECHANICAL PROPERTIES OF TISSUE
Funding Organization: EPSRC
Project Abstract: Elastic fibres play a key role in normal tissue function, and age-
related degenerative alterations in their biomechanical properties
severely affect dynamic functions of the cardiovascular system, lungs
and skin. This study aims to provide important information on the
effects of ageing on the properties of tissue by bridging the gap
between known mechanical changes at the whole tissue and molecular
levels. This will be achieved through developing acoustic methods for
characterising the mechanical properties of soft tissue. By using high
frequency acoustic microscopy in the frequency range of 100 MHz - 1
GHz we will obtain spatially resolved maps of mechanical properties in
thin histological sections with a spatial resolution in the x y plane
of around 1 micron. This information will then be correlated with
changes in the properties of elastic fibres extracted from tissue and
combined with conventional biochemical, ultrastructural and proteomic
approaches, to shed light on the causative mechanisms of elastic fibre
ageing.
Funded Value : £164702
```

Figure 7: Showing one of the web-scraped abstracts and their funding value

When writing the data to the new .csv file, I experienced some errors. More specifically, errors in the writing format due to the commas present in the text abstract. Comma-separated values or .csv has a particular writing format. This format includes separating any text with a comma present in them to a new cell.

This is not what I wanted as LDA analyses the data based off the header of the column and not on any text present in the document. So, to fix this I replaced every comma in the text abstract with a '|' symbol when writing to the new .csv file to prevent this error from happening again. Upon the program finishing, I then replaced every '|' symbol with the commas again using the find function in Microsoft Excel to perform successful Data Pre-Processing.

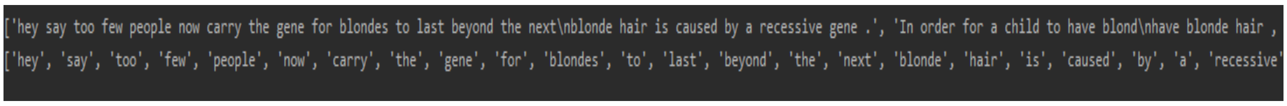
Data Pre-Processing

In this stage, I will be taking the raw abstract text dataset in the newly written .csv file and 'clean' the dataset to apply the Latent Dirichlet Allocation algorithm to it. I will be using the Natural Language Toolkit and the Gensim python libraries (Ebeid, Islam & Arango, Juan., 2016) for the pre-processing steps. This includes applying Tokenization, removing stop words, removing any words that have more than 3 characters, applying Lemmatization and finally Stemming the words in the text. The steps go as follows:

Tokenization

This is the first step in Natural Language Processing (Brownlee, 2019). This is the process (Guo, 1997) by which a group of strings are broken down into sentences and then sentences into words or tokens which helps the computer better understand human natural language, as shown in [Figure 8]. It is based on 2 crucial concepts:

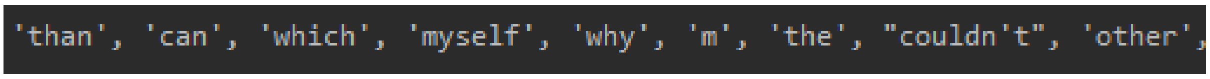
- All characters within the neighbouring group of strings are part of a single token. Tokens can be made of alphanumeric characters or numeric characters.
- Tokens are separated by whitespace, punctuation marks or line breaks.



```
['hey say too few people now carry the gene for blondes to last beyond the next\nblonde hair is caused by a recessive gene .', 'In order for a child to have blond\nhave blonde hair ,
['hey', 'say', 'too', 'few', 'people', 'now', 'carry', 'the', 'gene', 'for', 'blondes', 'to', 'last', 'beyond', 'the', 'next', 'blonde', 'hair', 'is', 'caused', 'by', 'a', 'recessive'
```

Figure 8: showing a sentence and its tokenized form

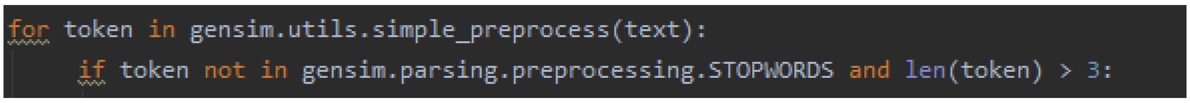
Stop-Words



```
'than', 'can', 'which', 'myself', 'why', 'm', 'the', "couldn't", 'other',
```

Figure 9: Showing example of some of the stop-words

Upon performing Tokenization, I noticed that there was a lot of filler words present in the abstracts. To be exact, it included words such as “the”,”is”,”in” and many more, an example is shown in [Figure 9]. These are called Stop Words (Baradad and Mugabushaka, n.d.) and they do not provide any value to the meaning of the abstracts and produce noise. So, to remove the noise, I filtered out these words from the abstracts along with any words that have less than 3 characters in them, to then perform the next stages of data pre-processing, as shown in [Figure 10].

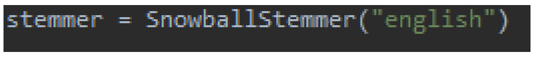


```
for token in gensim.utils.simple_preprocess(text):
    if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
```

Figure 10: Showing implementation of filtering stop-words

Stemming and Lemmatization

This step follows from Tokenization, as the aim now is to normalize the words to their root forms for a much easier to understand bag of words for the Latent Dirichlet Allocation algorithm.



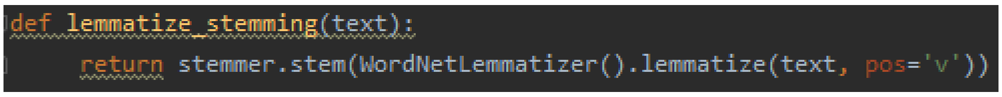
```
stemmer = SnowballStemmer("english")
```

Figure 11: Showing implementation of initialising the Snowball Stemmer

Stemming (Jivani, 2011) is the process of reducing the inflection in the tokens to their base forms such as mapping the group of words to the same stem even if the stem itself is not a valid word. For me to perform stemming on the abstracts, I have used the Snowball Stemmer as shown in [Figure 11], which takes the language of the text into account when performing stemming operations. Using Snowball stemmer alone would not give valid results as this is a more aggressive stemmer over the other stemmer named, Porter Stemmer. It is also a less aggressive stemmer compared to another stemming algorithm called the Lancaster stemmer.

I felt that by using Snowball stemmer (Developing a Stemmer for German Based on a Comparative Analysis of Publicly Available Stemmers, 2018), I would fare better because, though it’s more computationally expensive and aggressive than porter stemmer, it’s a gentler option than Lancaster stemmer. Snowball stemmer is essentially an upgrade on Porter stemmer as the issues of porter being very gentle are fixed. But, using Snowball Stemmer alone can cause over-stemming on words which would cause the program to create words that do not exist. This is when the process of Lemmatization comes into play.

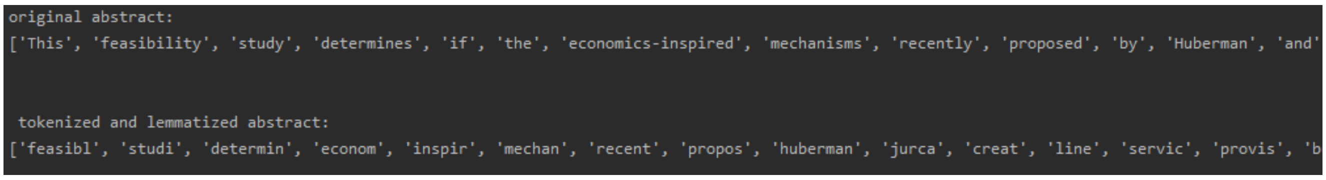
Lemmatization (Skorkovská, Lucie, 2012) is the process of converting the tokens that are in third person into first person words and verbs in the past and future tenses are changed to present tense. Additionally, it ensures that the converted tokens belong in the English language. Stemming, compared to Lemmatization, would just removes the last few characters of the word, often leading to incorrect meanings and spelling errors. Lemmatization on the other hand, would correctly identify the base form of word. For both stemming and lemmatization to work correctly, I wrote the program in a way that it stems on the lemmatized words so that the resulted output is understandable and is part of the English language as shown in [Figure 12].



```
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
```

Figure 12: Showing implementation of stemming and lemmatization function

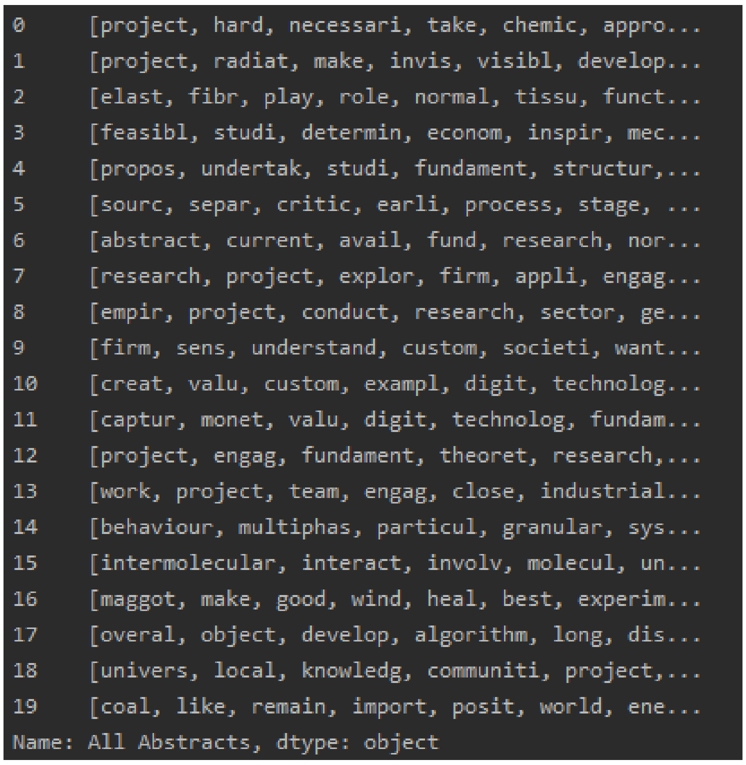
To test, whether my program works so far, I tested it on a single piece of project abstract in the dataset and it managed to correctly tokenize and lemmatize the words along with performing the just the right amount of stemming operations on each word so that the words don't lose their meanings, as seen in [Figure 13]. It had also removed any stop words, punctuations and whitespaces and essentially outputted a list of words in the abstract that give its meaning.



```
original abstract:
['This', 'feasibility', 'study', 'determines', 'if', 'the', 'economics-inspired', 'mechanisms', 'recently', 'proposed', 'by', 'Huberman', 'and'


 tokenized and lemmatized abstract:
['feasibl', 'studi', 'determin', 'econom', 'inspir', 'mechan', 'recent', 'propos', 'huberman', 'jurca', 'creat', 'line', 'servic', 'provis', 'b
```

Figure 13: Showing an abstract with its tokenized and lemmatized form.

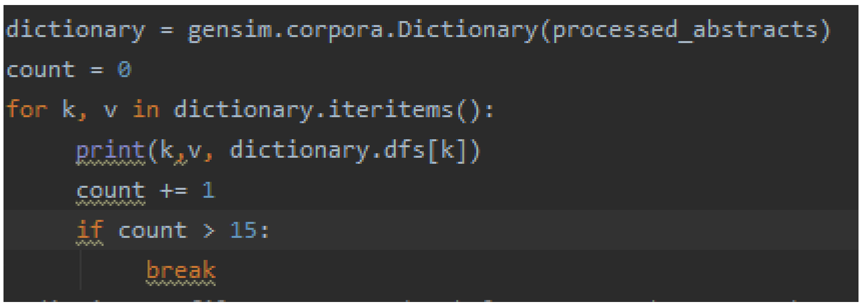
Upon passing this, I then ran the program on all the abstracts in the dataset and told it to print the results for the first 20 abstracts, as shown in [Figure 14]. It took the program about 10 minutes to process all the abstracts in the dataset and the results were outputted as expected by the program. Now, that my program was able to process all the abstracts in the dataset, I will turn the processed abstracts into a bag of words model.



```
0     [project, hard, necessari, take, chemic, appro...
1     [project, radiat, make, invis, visibl, develop...
2     [elast, fibr, play, role, normal, tissu, funct...
3     [feasibl, studi, determin, econom, inspir, mec...
4     [propos, undertak, studi, fundament, structur,...
5     [sourc, separ, critic, earli, process, stage, ...
6     [abstract, current, avail, fund, research, nor...
7     [research, project, explor, firm, appli, engag...
8     [empir, project, conduct, research, sector, ge...
9     [firm, sens, understand, custom, societi, want...
10    [creat, valu, custom, exampl, digit, technolog...
11    [captur, monet, valu, digit, technolog, fundam...
12    [project, engag, fundament, theoret, research,...
13    [work, project, team, engag, close, industrial...
14    [behaviour, multiphas, particul, granular, sys...
15    [intermolecular, interact, involv, molecul, un...
16    [maggot, make, good, wind, heal, best, experim...
17    [overal, object, develop, algorithm, long, dis...
18    [univers, local, knowledg, communiti, project,...
19    [coal, like, remain, import, posit, world, ene...
Name: All Abstracts, dtype: object
```

Figure 14: Showing the first 20 tokenized and lemmatized abstracts

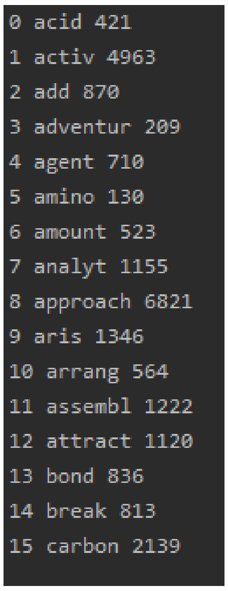
Preparing and running the Latent Dirichlet Allocation Algorithm



```
dictionary = gensim.corpora.Dictionary(processed_abstracts)
count = 0
for k, v in dictionary.iteritems():
    print(k,v, dictionary.dfs[k])
    count += 1
    if count > 15:
        break
```

Figure 15: Showing implementation of the dictionary and the for-loop with k-the fixed number of topics a document belongs to, v-size of the vocabulary

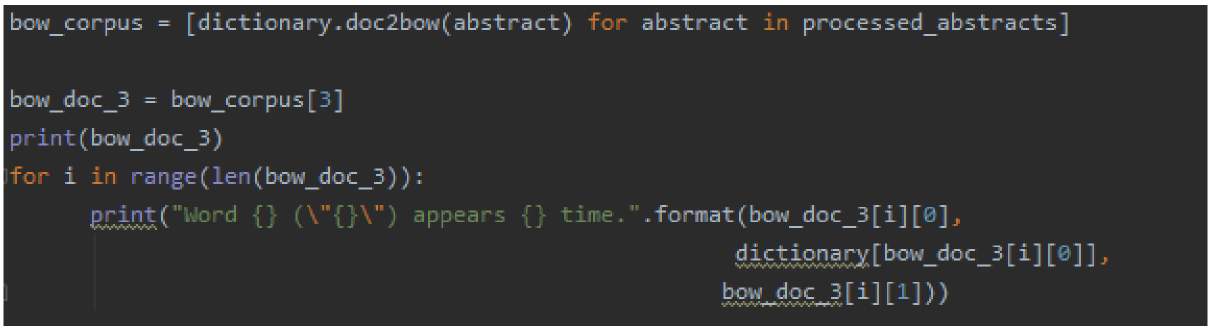
The Latent Dirichlet Allocation algorithm is based on the probability of word occurrence in a corpus. So, in order to run it on the abstracts in the dataset, I had to convert the pre-processed abstracts into a Bag of Words Model. This essentially is a dictionary of the amount of times each word appears in the dataset. Additionally, it is a form of representing text data when modelling text with machine learning algorithms. I have done so by creating a dictionary object using the Gensim library (Ebeid, Islam & Arango, Juan., 2016) on the processed abstracts from the dataset. I then wrote a for-loop which iterates through the bag of words and outputs the index of the words, the word itself and its frequency of occurrence throughout the dataset, as seen in [Figure 15]. The results for the first 15 words show that words such as “active” and “carbon” were used a lot more than words such as “adventure” and “acid”, as shown in [Figure 16].



```
0 acid 421
1 activ 4963
2 add 870
3 adventur 209
4 agent 710
5 amino 130
6 amount 523
7 analyt 1155
8 approach 6821
9 aris 1346
10 arrang 564
11 assembl 1222
12 attract 1120
13 bond 836
14 break 813
15 carbon 2139
```

Figure 16: Showing the first 15 most occurring words with the amount of times they occur in the dictionary

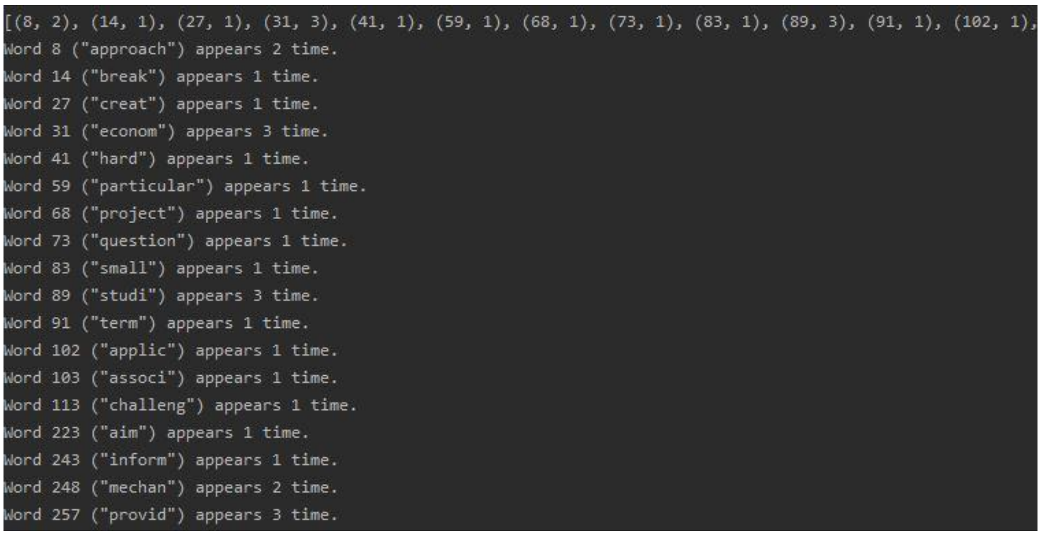
I now decided to filter the tokens analysed by removing any tokens that occur in less that 20 abstracts or more than half of the abstracts and upon doing that only keep the first 50,000 most frequently occurring tokens. I then created, bag of words corpus for each abstract which stores the total number of tokens in each abstract and shows the amount of times those word occur in the abstract, like the one shown in [Figure 17].



```python
bow_corpus = [dictionary.doc2bow(abstract) for abstract in processed_abstracts]

bow_doc_3 = bow_corpus[3]
print(bow_doc_3)
for i in range(len(bow_doc_3)):
    print("Word {} (\"{}\") appears {} time.".format(bow_doc_3[i][0],
                                                     dictionary[bow_doc_3[i][0]],
                                                    bow_doc_3[i][1]))
```

Figure 17: Showing implementation of the bag of word corpus

Each abstract resulted with a list of tuples, which shows the integer-id of the word and the amount of times that word occurs in the abstract, which can be seen in [Figure 18]. Now, the data is ready to be analysed using Latent Dirichlet Allocation.

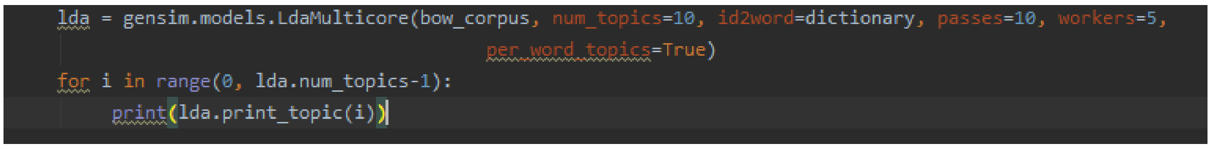


```
[(8, 2), (14, 1), (27, 1), (31, 3), (41, 1), (59, 1), (68, 1), (73, 1), (83, 1), (89, 3), (91, 1), (102, 1),
Word 8 ("approach") appears 2 time.
Word 14 ("break") appears 1 time.
Word 27 ("creat") appears 1 time.
Word 31 ("econom") appears 3 time.
Word 41 ("hard") appears 1 time.
Word 59 ("particular") appears 1 time.
Word 68 ("project") appears 1 time.
Word 73 ("question") appears 1 time.
Word 83 ("small") appears 1 time.
Word 89 ("studi") appears 3 time.
Word 91 ("term") appears 1 time.
Word 102 ("applic") appears 1 time.
Word 103 ("associ") appears 1 time.
Word 113 ("challeng") appears 1 time.
Word 223 ("aim") appears 1 time.
Word 243 ("inform") appears 1 time.
Word 248 ("mechan") appears 2 time.
Word 257 ("provid") appears 3 time.
```

Figure 18: Showing a list of tuples and the amount of times a words occur in a specific document

Testing the Latent Dirichlet Allocation Model

Latent Dirichlet Allocation Model or LDA Model (M.Blei, Y.Ng and I.Jordan, 2003) is an example of topic modelling used for classifying text in a corpus or corpora to a topic. It is a "generative statistical model" of a collection of combined data made up of different parts. It's a form of unsupervised learning which views documents as bags of words therefore, the order of the documents does not matter. It treats documents as probabilistic distribution sets of topic or words. These topics are not strongly defined as they are identified based on the likelihood of co-occurrences of words contained in them.

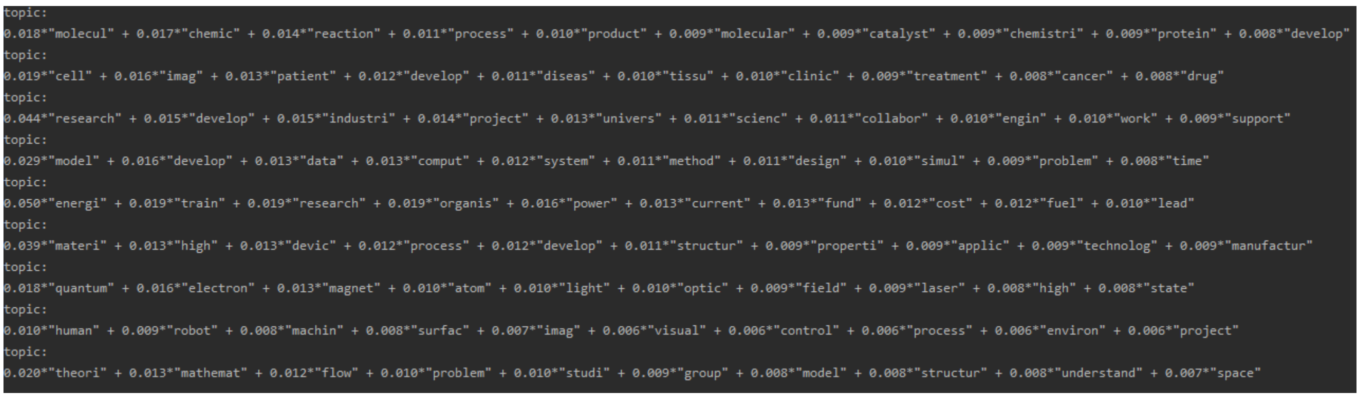


```python
lda = gensim.models.LdaMulticore(bow_corpus, num_topics=10, id2word=dictionary, passes=10, workers=5,
                                 per_word_topics=True)
for i in range(0, lda.num_topics-1):
    print(lda.print_topic(i))
```

Figure 19: Showing implementation of the Latent Dirichlet Allocation Algorithm

Now, I created an LDA object and called on the LDA Model using the Gensim library (Ebeid, Islam & Arango, Juan., 2016). The bag of words when fed into the algorithm would output a result containing generated topics and the most important words that occur in that topic. For this LDA object, I have specified parameters that includes that it use the bag of words corpus for each abstract that stores the total number of tokens in each abstract, generate 10 topics from the bag of words, use the id to words dictionary which contains the vocabulary of the bag of word corpus. Additionally, I've specified that it iterates through the data 10 times using 'passes' and with 5 workers processes using 'workers and print out the topics and the most important words present in each topic, as seen in [Figure 19]. Increasing the number of passes should yield results of important words that are similar in each topic. The important words present in each topic, would help a grant administrator in predicting the field of research for which the EPSRC receives the most funding requests. So, the closer the topic's relation to one another, the easier it is to predict the area of research that is funded the most. Upon running the algorithm, I was resulted with the following figures:

Test 1

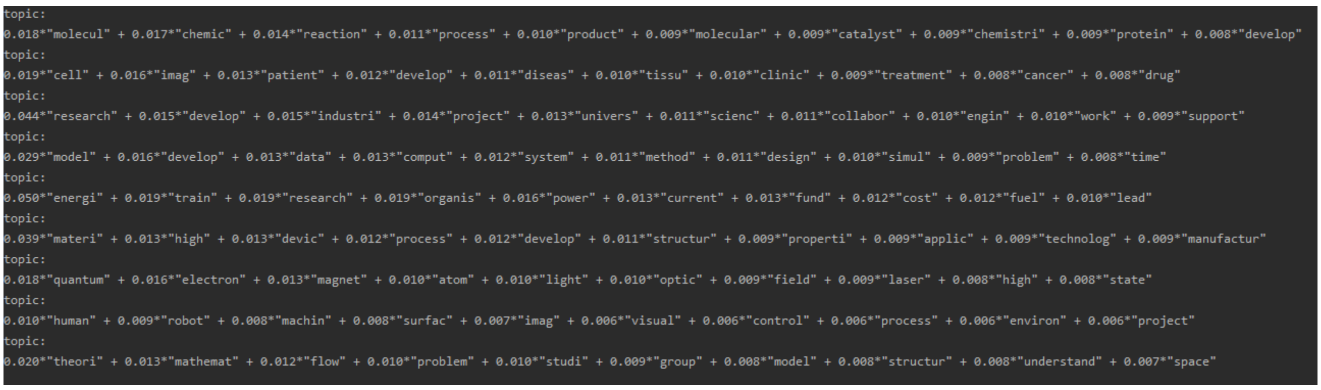


```
topic:
0.018*"molecul" + 0.017*"chemic" + 0.014*"reaction" + 0.011*"process" + 0.010*"product" + 0.009*"molecular" + 0.009*"catalyst" + 0.009*"chemistri" + 0.009*"protein" + 0.008*"develop"
topic:
0.019*"cell" + 0.016*"imag" + 0.013*"patient" + 0.012*"develop" + 0.011*"diseas" + 0.010*"tissu" + 0.010*"clinic" + 0.009*"treatment" + 0.008*"cancer" + 0.008*"drug"
topic:
0.044*"research" + 0.015*"develop" + 0.015*"industri" + 0.014*"project" + 0.013*"univers" + 0.011*"scienc" + 0.011*"collabor" + 0.010*"engin" + 0.010*"work" + 0.009*"support"
topic:
0.029*"model" + 0.016*"develop" + 0.013*"data" + 0.013*"comput" + 0.012*"system" + 0.011*"method" + 0.011*"design" + 0.010*"simul" + 0.009*"problem" + 0.008*"time"
topic:
0.050*"energi" + 0.019*"train" + 0.019*"research" + 0.019*"organis" + 0.016*"power" + 0.013*"current" + 0.013*"fund" + 0.012*"cost" + 0.012*"fuel" + 0.010*"lead"
topic:
0.039*"materi" + 0.013*"high" + 0.013*"devic" + 0.012*"process" + 0.012*"develop" + 0.011*"structur" + 0.009*"properti" + 0.009*"applic" + 0.009*"technolog" + 0.009*"manufactur"
topic:
0.018*"quantum" + 0.016*"electron" + 0.013*"magnet" + 0.010*"atom" + 0.010*"light" + 0.010*"optic" + 0.009*"field" + 0.009*"laser" + 0.008*"high" + 0.008*"state"
topic:
0.010*"human" + 0.009*"robot" + 0.008*"machin" + 0.008*"surfac" + 0.007*"imag" + 0.006*"visual" + 0.006*"control" + 0.006*"process" + 0.006*"environ" + 0.006*"project"
topic:
0.020*"theori" + 0.013*"mathemat" + 0.012*"flow" + 0.010*"problem" + 0.010*"studi" + 0.009*"group" + 0.008*"model" + 0.008*"structur" + 0.008*"understand" + 0.007*"space"
```

Figure 20: Results for the Test 1

This figure [Figure 20] shows 10 topics generated and the most important words present in that topic, for example in the first topic, the most important word generated is "molecul" or "molecule" with a probability of 0.018 and the least important word is "develop" with a probability of 0.008. This give me an idea that the first topic is about chemical process in the field of Chemistry. As seen in [Figure 20] the important words present in each topic have similarities in-terms of the field of research they belong in such as the word "molecular" and "molecule" being present in the first topic. [Figure 20] above also shows that the top 10 topics generated are closely related to a field in science, for example the second topic includes words such as "cell","imag" and "cancer" which probably relates to a field in Biology that focuses on "imaging cancer cells".

Test 2

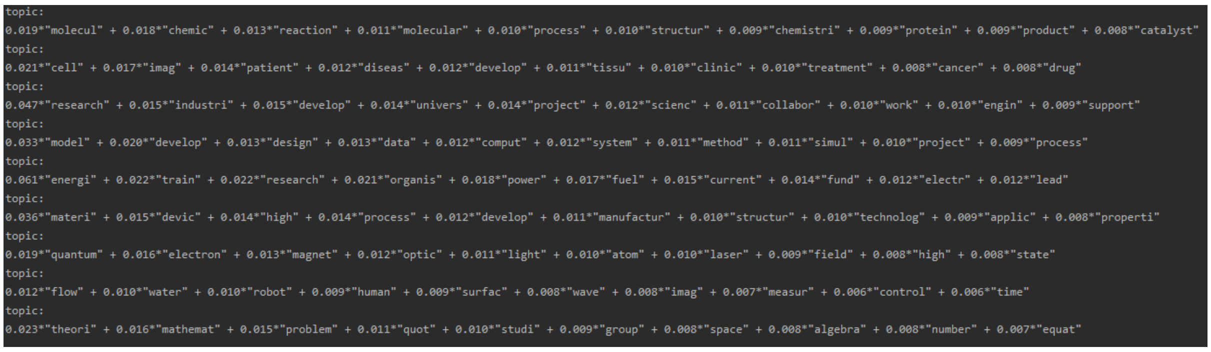


```
topic:
0.018*"molecul" + 0.017*"chemic" + 0.014*"reaction" + 0.011*"process" + 0.010*"product" + 0.009*"molecular" + 0.009*"catalyst" + 0.009*"chemistri" + 0.009*"protein" + 0.008*"develop"
topic:
0.019*"cell" + 0.016*"imag" + 0.013*"patient" + 0.012*"develop" + 0.011*"diseas" + 0.010*"tissu" + 0.010*"clinic" + 0.009*"treatment" + 0.008*"cancer" + 0.008*"drug"
topic:
0.044*"research" + 0.015*"develop" + 0.015*"industri" + 0.014*"project" + 0.013*"univers" + 0.011*"scienc" + 0.011*"collabor" + 0.010*"engin" + 0.010*"work" + 0.009*"support"
topic:
0.029*"model" + 0.016*"develop" + 0.013*"data" + 0.013*"comput" + 0.012*"system" + 0.011*"method" + 0.011*"design" + 0.010*"simul" + 0.009*"problem" + 0.008*"time"
topic:
0.050*"energi" + 0.019*"train" + 0.019*"research" + 0.019*"organis" + 0.016*"power" + 0.013*"current" + 0.013*"fund" + 0.012*"cost" + 0.012*"fuel" + 0.010*"lead"
topic:
0.039*"materi" + 0.013*"high" + 0.013*"devic" + 0.012*"process" + 0.012*"develop" + 0.011*"structur" + 0.009*"properti" + 0.009*"applic" + 0.009*"technolog" + 0.009*"manufactur"
topic:
0.018*"quantum" + 0.016*"electron" + 0.013*"magnet" + 0.010*"atom" + 0.010*"light" + 0.010*"optic" + 0.009*"field" + 0.009*"laser" + 0.008*"high" + 0.008*"state"
topic:
0.010*"human" + 0.009*"robot" + 0.008*"machin" + 0.008*"surfac" + 0.007*"imag" + 0.006*"visual" + 0.006*"control" + 0.006*"process" + 0.006*"environ" + 0.006*"project"
topic:
0.020*"theori" + 0.013*"mathemat" + 0.012*"flow" + 0.010*"problem" + 0.010*"studi" + 0.009*"group" + 0.008*"model" + 0.008*"structur" + 0.008*"understand" + 0.007*"space"
```

Figure 21: Results for Test 2

I now will be increasing the number of passes to 50 and the workers will be kept at the same value of 5. The result is as shown in [Figure 21]. This increase comparing to Test 1 has not changed any values or added any new words to the topics.

Test 3

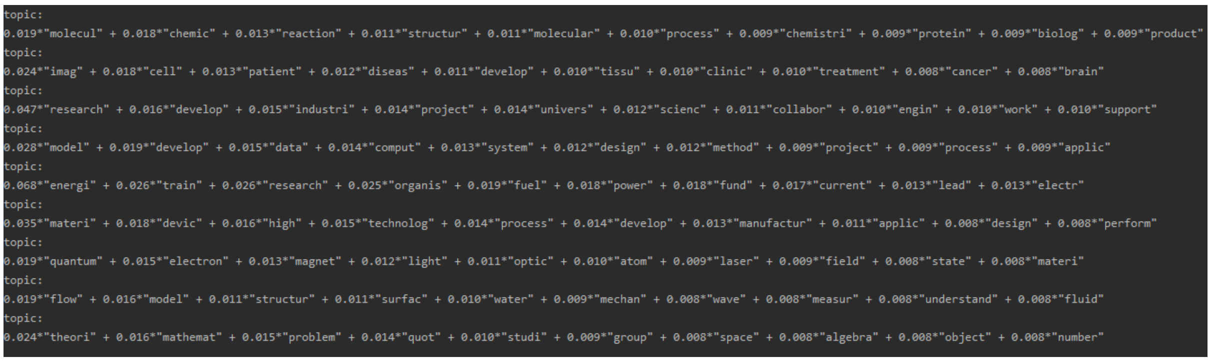


```
topic:
0.019*"molecul" + 0.018*"chemic" + 0.013*"reaction" + 0.011*"molecular" + 0.010*"process" + 0.010*"structur" + 0.009*"chemistri" + 0.009*"protein" + 0.009*"product" + 0.008*"catalyst"
topic:
0.021*"cell" + 0.017*"imag" + 0.014*"patient" + 0.012*"diseas" + 0.012*"develop" + 0.011*"tissu" + 0.010*"clinic" + 0.010*"treatment" + 0.008*"cancer" + 0.008*"drug"
topic:
0.047*"research" + 0.015*"industri" + 0.015*"develop" + 0.014*"univers" + 0.014*"project" + 0.012*"scienc" + 0.011*"collabor" + 0.010*"work" + 0.010*"engin" + 0.009*"support"
topic:
0.033*"model" + 0.020*"develop" + 0.013*"design" + 0.013*"data" + 0.012*"comput" + 0.012*"system" + 0.011*"method" + 0.011*"simul" + 0.010*"project" + 0.009*"process"
topic:
0.061*"energi" + 0.022*"train" + 0.022*"research" + 0.021*"organis" + 0.018*"power" + 0.017*"fuel" + 0.015*"current" + 0.014*"fund" + 0.012*"electr" + 0.012*"lead"
topic:
0.036*"materi" + 0.015*"devic" + 0.014*"high" + 0.014*"process" + 0.012*"develop" + 0.011*"manufactur" + 0.010*"structur" + 0.010*"technolog" + 0.009*"applic" + 0.008*"properti"
topic:
0.019*"quantum" + 0.016*"electron" + 0.013*"magnet" + 0.012*"optic" + 0.011*"light" + 0.010*"atom" + 0.010*"laser" + 0.009*"field" + 0.008*"high" + 0.008*"state"
topic:
0.012*"flow" + 0.010*"water" + 0.010*"robot" + 0.009*"human" + 0.009*"surfac" + 0.008*"wave" + 0.008*"imag" + 0.007*"measur" + 0.006*"control" + 0.006*"time"
topic:
0.023*"theori" + 0.016*"mathemat" + 0.015*"problem" + 0.011*"quot" + 0.010*"studi" + 0.009*"group" + 0.008*"space" + 0.008*"algebra" + 0.008*"number" + 0.007*"equat"
```

Figure 22: Results for Test 3

Now I will be increasing the number of passes to 100 and the worker processes remain the same at the value of 5. The result for this test is shown in [Figure 22]. From this test, I gather that for topic 8 in Test 3, new words such as "flow" and "water" were added on compared to the previous Test 2. Including these words still doesn't change the general topic for topic 8 of human-robot interaction but, within this topic it changes the specifics of the topic. This means that increasing the passes from 50 in Test 2 to 100 in Test 3 provided, a much better understanding of the field of research topic 8 belongs to.

Test 4

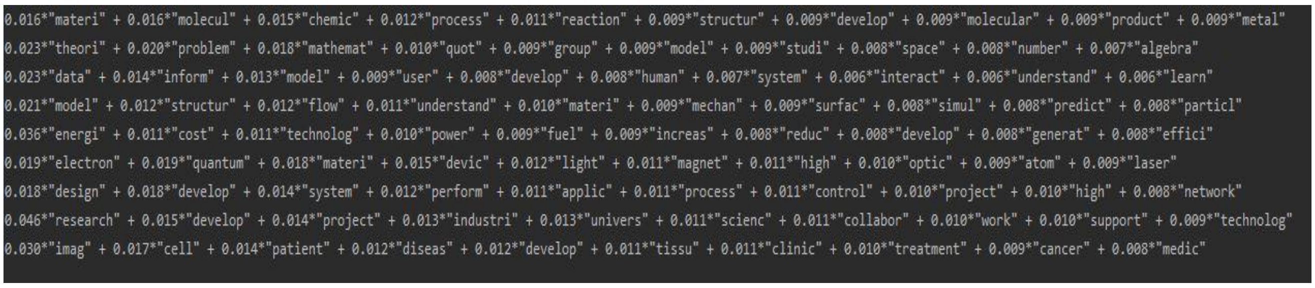


```
topic:
0.019*"molecul" + 0.018*"chemic" + 0.013*"reaction" + 0.011*"structur" + 0.011*"molecular" + 0.010*"process" + 0.009*"chemistri" + 0.009*"protein" + 0.009*"biolog" + 0.009*"product"
topic:
0.024*"imag" + 0.018*"cell" + 0.013*"patient" + 0.012*"diseas" + 0.011*"develop" + 0.010*"tissu" + 0.010*"clinic" + 0.010*"treatment" + 0.008*"cancer" + 0.008*"brain"
topic:
0.047*"research" + 0.016*"develop" + 0.015*"industri" + 0.014*"project" + 0.014*"univers" + 0.012*"scienc" + 0.011*"collabor" + 0.010*"engin" + 0.010*"work" + 0.010*"support"
topic:
0.028*"model" + 0.019*"develop" + 0.015*"data" + 0.014*"comput" + 0.013*"system" + 0.012*"design" + 0.012*"method" + 0.009*"project" + 0.009*"process" + 0.009*"applic"
topic:
0.068*"energi" + 0.026*"train" + 0.026*"research" + 0.025*"organis" + 0.019*"fuel" + 0.018*"power" + 0.018*"fund" + 0.017*"current" + 0.013*"lead" + 0.013*"electr"
topic:
0.035*"materi" + 0.018*"devic" + 0.016*"high" + 0.015*"technolog" + 0.014*"process" + 0.014*"develop" + 0.013*"manufactur" + 0.011*"applic" + 0.008*"design" + 0.008*"perform"
topic:
0.019*"quantum" + 0.015*"electron" + 0.013*"magnet" + 0.012*"light" + 0.011*"optic" + 0.010*"atom" + 0.009*"laser" + 0.009*"field" + 0.008*"state" + 0.008*"materi"
topic:
0.019*"flow" + 0.016*"model" + 0.011*"structur" + 0.011*"surfac" + 0.010*"water" + 0.009*"mechan" + 0.008*"wave" + 0.008*"measur" + 0.008*"understand" + 0.008*"fluid"
topic:
0.024*"theori" + 0.016*"mathemat" + 0.015*"problem" + 0.014*"quot" + 0.010*"studi" + 0.009*"group" + 0.008*"space" + 0.008*"algebra" + 0.008*"object" + 0.008*"number"
```

Figure 23: Results for Test 4

Now I will be increasing the number passes to 500 and the worker processes remain the same at the value of 5. Upon running this test, I gather that topic 2 has a new word "brain" included in this test as seen in [Figure 23]. Comparing this to topic 2 in Test 3, from which I gather, the topic to be about imaging cancer cells, the inclusion of the word "brain" in Test 4 gives me a better idea of the area of cancer that the research is about. Additionally, when comparing topic 8 of all the previous tests, I noticed that the word "robot" and "human" is missing in the topic in this test, changing the interpretation of the topic itself. For this test, I conclude that Increasing the passes as indeed improved the probability of the interpretation for some topics but, for others such as topic 8, has completely changed the interpretation completely reducing the probability of getting the right interpretation.

Test 5



```
0.016*"materi" + 0.016*"molecul" + 0.015*"chemic" + 0.012*"process" + 0.011*"reaction" + 0.009*"structur" + 0.009*"develop" + 0.009*"molecular" + 0.009*"product" + 0.009*"metal"
0.023*"theori" + 0.020*"problem" + 0.018*"mathemat" + 0.010*"quot" + 0.009*"group" + 0.009*"model" + 0.009*"studi" + 0.008*"space" + 0.008*"number" + 0.007*"algebra"
0.023*"data" + 0.014*"inform" + 0.013*"model" + 0.009*"user" + 0.008*"develop" + 0.008*"human" + 0.007*"system" + 0.006*"interact" + 0.006*"understand" + 0.006*"learn"
0.021*"model" + 0.012*"structur" + 0.012*"flow" + 0.011*"understand" + 0.010*"materi" + 0.009*"mechan" + 0.009*"surfac" + 0.008*"simul" + 0.008*"predict" + 0.008*"particl"
0.036*"energi" + 0.011*"cost" + 0.011*"technolog" + 0.010*"power" + 0.009*"fuel" + 0.009*"increas" + 0.008*"reduc" + 0.008*"develop" + 0.008*"generat" + 0.008*"effici"
0.019*"electron" + 0.019*"quantum" + 0.018*"materi" + 0.015*"devic" + 0.012*"light" + 0.011*"magnet" + 0.011*"high" + 0.010*"optic" + 0.009*"atom" + 0.009*"laser"
0.018*"design" + 0.018*"develop" + 0.014*"system" + 0.012*"perform" + 0.011*"applic" + 0.011*"process" + 0.011*"control" + 0.010*"project" + 0.010*"high" + 0.008*"network"
0.046*"research" + 0.015*"develop" + 0.014*"project" + 0.013*"industri" + 0.013*"univers" + 0.011*"scienc" + 0.011*"collabor" + 0.010*"work" + 0.010*"support" + 0.009*"technolog"
0.030*"imag" + 0.017*"cell" + 0.014*"patient" + 0.012*"diseas" + 0.012*"develop" + 0.011*"tissu" + 0.011*"clinic" + 0.010*"treatment" + 0.009*"cancer" + 0.008*"medic"
```

Figure 24: Results for Test 5

I now will be increasing the ‘passes’ parameter to 1000 and keeping the ‘workers’ parameter the same which is 5. This means that LDA will run 1000 times on the data in order to stabilize the words produced in each topic. The result is as seen [Figure 25]. It shows that word “technolog” meaning “technology” was repeated in multiple topics across the result and comparing to the previous result in Test 2 and Test 3, the word “molecul” or “molecule” had decreased probability of 0.016. Additionally, the word “develop” also occurred in the same topic as “molecul” and had an increased probability of 0.009. Overall, this again shows the overall meaning of each topic using the important words present in them but, when comparing topic 4 with the previous tests, I noticed that the word “particl” was added on to the words as seen in [Figure 24] which was not present [Figure 21]. So initially, I predicted topic 4 to be about computers but upon conducting Test 5, I realized the topic could be about chemistry and its relation to computers. This would mean that the topic was interpreted wrong in the previous tests and increasing the passes improved the result and therefore the interpretation of the topic.

Upon analysing the results produced, I thought that each topic generated by the algorithm was descriptive enough to guess the topic of research through the important words present in each topic. There were some anomalies present such as in Test 4, topic 8 where excluding some words caused confusion on the topic description. Additionally, there were some strange results produced such [Figure 24] in Test 5 with running 1000 passes through the data where some topics such as topic 2 had no relation with any of the other tests. This could mean that the other tests are wrong and Test 5, as it had a higher number of passes, yielded a more valid result. Conversely, this could also mean that Test 5 yielded the most anomalous result due to its distinctive properties. But, Latent Dirichlet Allocation is a probability modal, so the topic interpretation is left to the person reading the words present in the topic. More analysis would be needed to find a definite result, but the list below shows my general interpretations on the topics generated based off the important words present in all the 5 tests.

Topic interpretation:

Topic 1: Chemistry and Biology

Topic 2: Cancer cells imaging

Topic 3: Physics

Topic 4: Computers

Topic 5: Physics and fuels

Topic 6: Technology and manufacturing

Topic 7: Quantum physics

Topic 8: fluid mechanics

Topic 9: Physics and Mathematics

Evaluation

Overall, the development for the project went well. But there were a few minor setbacks and areas of difficulty when programming the application. This included writing in a language that I have never programmed with before. The aim of the project was, to provide insightful information on the grant abstracts upon performing successful text analysis, this was proved to be achieved.

When developing the application, there were several aspects that went very well. This includes programming the web-scraping stages of the project as, I was able to find the right guides while researching about the concept. Due to this, I was able to successfully understand the concept and with my knowledge, program this into my application. Additionally, I was also able to learn and understand the concepts of text mining which is a crucial concept for the project. Upon doing this, I was able to find its relation to Python and its libraries. By gaining this knowledge, I was able to rightfully pick and choose the text mining techniques that were relevant to this project.

In the later developmental stages of the project, I was able successfully understand the Latent Dirichlet Allocation model. With this knowledge, I was able to research the relevant information on this topic and successfully understand its relation to my project. By doing this, I was able to use the right tools such as identifying the relevant libraries provided by Python effectively. Doing this, helped me spot errors in the initial results outputted by the program and fix them in an efficient manner. Understanding this model also helped me interpret the results correctly and find hidden relations to my project.

During early developmental stages of the project, I ran into a lot of problems, mainly due to my lack of knowledge in Python and its syntax. So, this led me to make a lot of errors when programming such as, preparing the dataset in the wrong format. This has a major impact on my work completion as it took a very long time for me to write the data. But by making these errors, I was able to avoid errors similar to them when developing the future stages of development. For me to prevent this from happening, I must perform thorough reading on the topics I research before implementing a feature that has a big impact on my project.

Additionally, when progressing through the key developmental stages of this project, UK was going through the COVID-19 crisis. This turned out to be a major setback for my project as I was developing my project on the university computers in the library and couldn't access them due to the lockdown which was announced in mid-March. This meant starting over the whole project again on my home computer. But, as I had already done some stages of development before, I was able to use that knowledge to catch-up the work on my home computer effectively. To prevent something like this from happening again, I must always save a back-up on my home computer or save the work done on university computers on a cloud storage to be accessed elsewhere.

The testing phase of this project proved to be a challenge due to the amount of time it takes for the program to output its result due to the sheer size of the dataset. Whether this be the web-scraping or the actual text analysis. Now, this proved to be a problem as when testing on a single data in the set, outputted desired results but when performing the same tests on a larger number of data, the program seemed to output invalid results after a long period of time. This also meant that when performing on operation on the data, I wouldn't be able to do other things with them as the IDE can only process one command at a time. But, in order to overcome the time being wasted between each test, I decided to use other IDEs to run different aspects of the application in small bits, which later helped me prevent the time being wasted. A better way to prevent this from happening again would be to use an additional computer such my laptop, which will be running another aspect of the program simultaneously.

The results produced by the algorithm proved to show all the information I required to interpret the data effectively. When writing the interpretation for them, I thought that the results provided me with the relevant information efficiently. But this could prove to be a challenge for a person that is still understanding the concept of Latent Dirichlet Allocation or maybe even the grant administrator, especially [Figure 24] in Test 5. To fix this, I must make the results clearer by, showing the topic indexes and provide a brief description on how the Latent Dirichlet Allocation model works before providing the results.

Conclusion

The title of this project is to perform text mining on EPSRC grant abstracts. Additionally, provide the user with insightful information. When we look back at the aim of this project, I feel like my project has achieved those aims successfully. Upon performing text analysis using many different algorithms including a machine learning algorithm on the abstracts, I was able to output insightful information on the most common abstract topics found throughout the dataset. This information could very closely relate to the grant funding provided by the EPSRC for research. For example, based of the topics discovered one could arrive at the conclusion that one topic is funded more than the other due to the topic being discovered by the algorithm. But I feel that this information alone would not be enough to arrive at that conclusion. I feel that there are a lot more features to analyse such as the current world problems and the area of research that the EPSRC wishes to focus its funding upon.

Bibliography

[1] Linguamatics. (2019). *What is Text Mining, Text Analytics and Natural Language Processing?*. [online] Available at: https://www.linguamatics.com/what-text-mining-text-analytics-and-natural-language-processing.

[2] Brownlee, J. (2019). What Is Natural Language Processing?. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/natural-language-processing/.

[3] M.Blei, D., Y.Ng, A. and I.Jordan, M. (2003). Latent Dirichlet Allocation. [online] Jmlr.org. Available at: http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf.

[4] Techopedia.com. (n.d.). What is the Natural Language Toolkit (NLTK)? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/30343/natural-languagetoolkit-nltk.

[5] Guo, J. (1997). Critical Tokenization and its Properties. [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/b484/7557362f1428ab0ef63f2c8e96f9fe89aa5e.pdf

[6] Jivani, A. (2011). A Comparative Study of Stemming Algorithms. [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/1c0c/0fa35d4ff8a2f925eb955e48d655494bd167.pdf.

[7] Wieringa, J. (2012). Intro to Beautiful Soup. [online] Programminghistorian.org. Available at: https://programminghistorian.org/en/lessons/intro-to-beautiful-soup.

[8] Matveeva, I. and Levow, G. (2005). http://people.cs.uchicago.edu/~matveeva/HLTworkshop.pdf. [online] People.cs.uchicago.edu. Available at: http://people.cs.uchicago.edu/~matveeva/HLTworkshop.pdf.

[9] Park, J., Blume-kohout, M., Krestel, R., Nalisnick, E. and Smyth, P. (2016). Analyzing NIH Funding Patterns over Time with Statistical Text Analysis. [online] Enalisnick.github.io. Available at: https://enalisnick.github.io/sbd16.pdf.

[10] Diaconis, P. (1988). De Finetti's Algorithm. [online] Statweb.stanford.edu. Available at: https://statweb.stanford.edu/~cgates/PERSI/papers/recent_progress.pdf.

[11] Mckinney, Wes. (2011). pandas: a Foundational Python Library for Data Analysis and Statistics. Python High Performance Science Computer. Available at: https://www.researchgate.net/publication/265194455_pandas_a_Foundational_Python_Library_for_Data_Analysis_and_Statistics.

[12] Skorkovská, Lucie. (2012). Application of Lemmatization and Summarization Methods in Topic Identification Module for Large Scale Language Modeling Data Filtering. 7499. 10.1007/978-3-642-32790-2_23. Available at: https://www.researchgate.net/profile/Lucie_Skorkovska/publication/259928409_Application_of_Lemmatization_and_Summarization_Methods_in_Topic_Identification_Module_for_Large_Scale_Language_Modeling_Data_Filtering/links/5795fa0408ae33e89fad7043/Application-of-Lemmatization-and-Summarization-Methods-in-Topic-Identification-Module-for-Large-Scale-Language-Modeling-Data-Filtering.pdf.

[13] Baradad, V. and Mugabushaka, A., n.d. *Corpus Specific Stop Words To Improve The Textual Analysis In Scientometrics*. [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/618d/a4f6d5cd329d3bc498d9457f575cbdfaf53d.pdf.

[14] Weissweiler, L. and Fraser, A., 2018. *Developing A Stemmer For German Based On A Comparative Analysis Of Publicly Available Stemmers*. [online] Rdcu.be. Available at: https://rdcu.be/b31EX.

[15] Ebeid, Islam & Arango, Juan. (2016). Mallet vs GenSim: Topic Modeling Evaluation Report. 10.13140/RG.2.2.19179.39205/1. Available at: https://www.researchgate.net/profile/Islam_Ebeid/publication/331972126_Mallet_vs_GenSim_Topic_Modeling_Evaluation_Report/links/5c96e229a6fdccd46036707e/Mallet-vs-GenSim-Topic-Modeling-Evaluation-Report.pdf.