PROJECT REPORT ON

"Detection of Fraud using Topic Modelling techniques on text data"

A Project Work Submitted in Partial Fulfilment of the requirements for

The Course

ALGORITHMS FOR INTELLIGENCE WEB AND INFORMATION RETRIEVAL

by

Aronya Baksy (PES1201800002)

Ansh Sarkar (PES1201800275)

Vishesh P (PES1201800314)

Under the supervision of

Prof. Nagegowda K S

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PES UNIVERSITY

RR CAMPUS

TABLE OF CONTENTS:

1 Introduction

1.1 Abstract of the Project

The Securities and Exchange Commission (SEC) of the United States government is a regulatory body that was established in 1933 in order to protect financial markets against market manipulation and enforce laws regarding the same.
The SEC publishes **press releases** that highlight recent actions taken by the SEC as well as other newsworthy information regarding financial markets in the USA. These press releases are publicly available on the SEC's official website here. These can be accessed and retrieved in a year-wise fashion.

In our current scenario, where so much of financial activity worldwide takes place online, the press releases put out by the SEC at regular intervals are an important source of data regarding financial fraud. Text mining techniques have been used for a long time to detect instances of financial fraud in text data, most notably during the Enron financial scandal and the release of internal office communications on the same [1].

Text mining techniques such as TF-IDF based cosine-similarity models, and topic modelling techniques such as Latent Semantic Indexing (LSA) and its advanced cousin, the Latent Dirichlet Allocation (**LDA**) are used as techniques to extract semantic relations between topics. All of these models are unsupervised techniques that require human interpretation in order to produce useful results.

1.2 Problem Statement

The problem statement of this project is to be able to detect instances of financial fraud using text mining techniques, from the corpus of press releases output by the SEC on its website.

Using LDA, TF-IDF and cosine similarity as the selected approaches, we aim to be able to classify documents in the corpus as containing information about financial fraud or not, as well as allow a user to search for more specific instances of financial fraud using free-text queries.

2 Project Overview

2.1 Proposed Approach

2.1.1 Data Gathering

The first step in this project is the gathering of the data from the SEC's website into a usable format. This is achieved using a custom web crawler designed in Python using the `requests` library, and the `BeautifulSoup` HTML Parser.

The `requests` module is used to retrieve the HTML response from the server, and the `BeautifulSoup` module is used to parse the HTML, and extract the article text, as well as the metadata regarding the article, such as the place and date of publishing, the title and the unique identifier for the article.

Once gathered, the data is placed into a hierarchical directory structure. The root `Data` directory contains sub-folders organized by the year of the press release that are named in the format `Year_20xx` where 20xx is the 4-digit year.

The articles are stored in `.txt` format which is readable across platforms and carries minimal overhead, and hence is most efficient in storing in terms of disk space utilization.

2.1.2 Data Cleaning

The gathered text data must be cleaned, i.e. pre-processed into an appropriate and usable form so that useful information can be extracted from it.

The first step of cleaning is sentence tokenization, which is an algorithmic procedure for converting a body of text into a list of *sentences*. Each sentence is further tokenized into *words*.

Now that the corpus is tokenized into words, case folding is carried out, wherein the entire corpus is converted to lowercase.

The tokenized and case-folded corpus is put through a Parts-of-Speech Tagger (PoS Tagger) that outputs the token as well as the corresponding part of speech (verb, noun, adverb, adjective, etc.) that it belongs to. This extra information is useful for the next cleaning step.

The PoS-tagged text is then put through a *lemmatizer*. Lemmatization is a process by which various forms (i.e. verb forms, adjective forms etc.) are collected into a single base form. Lemmatization is different from stemming in its use of morphological and dictionary information in comparision to stemming which is a purely algorithmic approach.

2.1.3 Text Mining Approach 1: TF-IDF

This approaches uses a TF-IDF based approach to rank documents based on their similarity to a given query, and output the most relevant documents based on this score. The similarity metric used in this case is the **cosine similarity** which measures the semantic similarity of two entities as the cosine of the angle between their vector representations.

2.1.4 Text Mining Approach 2: LDA

First, each document in the corpus of documents is labelled manually to indicate whether it contains an instance of financial fraud (the label is a binary value, with a value of 1 indicating an instance of financial fraud and a value of 0 indicating otherwise).

The basic intuition behind an LDA topic model is that each word in each document comes from a topic and the topic is selected from a per-document distribution over topics. The topic that covers the most number of terms is considered to be the dominant topic, and for each document that is related to this dominant topic, the predicted label is said to be 1, and 0 for all other documents.

3 Algorithms and Pseudocode

3.1 Search Engine based on TF-IDF

After cleaning all the documents in our dataset, a matrix consisting of numerical values is constructed to represent all the documents. This matrix is used to determine the similarity between any pair of documents and between a user provided query and any document. This matrix is called the Term-Document Matrix. The rows of this matrix represent the unique terms collected from the documents and the columns represent all the documents in the form of numerical identifiers. The TF-IDF method is used to calculate the value inside every cell in the matrix. TF-IDF consists of two parts namely the Term-Frequency(TF) and the Inverse Document Frequency(IDF).
Term-Frequency(TF) of a term t is a function of the frequency of the term on a document d.

$$tf(t, d, D) = \frac{f_{t,d}}{\sum_{t' \in D} f_{t',d}} \tag{1}$$

The Inverse Document Frequency of a term acts as a weight for the TF value calculated earlier. The IDF value of a term is inversely proportional to the frequency of the word across the set of documents.

$$idf(t, D) = log_{10}\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right) \tag{2}$$

The two values for each term are multiplied to get the cell value of that particular term in the corrsesponding document. This process is performed on all the cells to get the Term-Document Matrix. Once the Term Document Matrix has been constructed, the next step is to determine the similarity between a search query provided by the user and every single document. To achieve this, the method of Cosine Similarity is implemented. The Cosine Similarity between two vectors is calculated by taking the dot product of the two vectors and dividing it by the product of the lengths of both the vectors. The value of the Cosine function ranges from -1 to 1 usually. However, the values in our query vector and Term-Document Matrix are always greater than or equal to 0. Hence the value of the similarity measure in this case always ranges from 0 to 1.

$$cosineSimilarity(q, d) = \frac{\overrightarrow{q} . \overrightarrow{d}}{\| \overrightarrow{q} \| \| \overrightarrow{d} \|} \tag{3}$$

The query q is first transformed as a vector on the Term-Document Matrix. Then the pair-wise similarity between the query vector and every document's vector d is calculated and sorted in descending order. Finally, we retrieve the required number of documents all having similarity greater than 0 with the query vector.

3.1.1 Pseudocode

The pseudocode below details the calculation of the top N most similar documents that are returned by the system given an user query q and a corpus of documents D.

Algorithm 1 Tf-IDF Similarity Algorithm

```
procedure getSimilarDocuments(D: corpus, query, N)
    vocab ← {}
    for document d ∈ D do
        for term t ∈ d do
            if t ∉ vocab then
                vocab.add(t)
    TfIDFModel ← TFIDFVectorizer(vocabulary)
    TfIDFModel.fit(D)
    q ← TfIDFModel.fit(query)
    documentScores ← ⟨⟩
    for v ∈ TfIDFModel do
        similarity = cosineSimilarity(v, q)
        if similarity > 0 then
            documentScores.add(docID, similarity)
    topNdocs ← sort(documentScores, on=similarity)[:N]
    return topNdocs
```

3.2 LDA-based classification

Latent Dirichlet Allocation (LDA) is a "generative probabilistic model" of a collection of composites made up of parts.It is a form of unsupervised learning that views documents as bags of words i.e order does not matter. This algorithm was used for topic modelling on our cleaned data. Topic modelling is the process of identifying topics in a set of documents.In terms of topic modelling, the composites are documents and the parts are words and/or phrases. This is essentially a clustering problem where both words and documents can be thought as being clustered.
LDA model has a few assumptions behind it:

- Documents exhibit multiple topics (but typically not many)
- LDA is a probabilistic model with a correspondinggenerative process – each document is assumed to be generated by this (simple) process
- A *topic* is a distribution over a fixed vocabulary – these topics are assumed to be generated first, before the documents
- Only the number of topics is specified in advance.

LDA works by first making a key assumption: the way a document was generated was by picking a set of topics and then for each topic picking a set of words. To find topics LDA simply reverse engineers this assumption.

The process can be defined for each document as follows:

- Assume there are k topics across all of the documents
- Distribute these k topics across document m (this distribution is known as α and can be symmetric or asymmetric) by assigning each word a topic.
- For each word w in document m, assume its topic is wrong but every other word is assigned the correct topic.
- Probabilistically assign word w a topic based on two things:
  - what topics are in document m.
  - how many times word w has been assigned a particular topic across all of the documents (this distribution is called β).
- Repeat this process a number of times for each document.

3.2.1 Dirichlet Distribution

The distribution described in the above process is called the Dirichlet distribution. [**Need to add some info pls help**].
The α and β values are defined as following:

- α is the per-document topic distributions
- β is the per-topic word distribution,

The distribution can be defined as follows:

$$p(x|\alpha) = \frac{\Gamma(\sum_{i=1}^{d} \alpha_i)}{\Pi_{i=1}^{d} \Gamma(\alpha_i)} \prod_{i=1}^{d} x_i^{\alpha_i - 1} \tag{4}$$

for observations

$$\sum_{i=1}^{d} x_i = 1, \; x_i \geq 0$$

where $\Gamma()$ is the Gamma distribution

3.2.2 Formal look at LDA algorithm

The following definitions are put in place for the definition of the LDA algorithm.

- $\beta_{1:K}$ are the K topics where each β_k is a distribution over the vocabulary
- θ_d is the topic proportions for each of the K topics in document d.
- $\theta_{d,k}$ is the topic proportion for topic k in document d
- z_d are the topic assignments for document d.
- $z_{d,n}$ is the topic assignment for word n in document d

- w_d are the observed words for document d

Then the joint distribution of the hidden and observed variables is given as below:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} \theta_d \prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n}) \tag{5}$$

3.2.3 Pseudocode

Algorithm 2 LDA Fraud Classification

```
procedure classifyFraud(D: corpus, query, N)
    vocab ← {}
    for document d ∈ D do
        for sentence s ∈ d do
            vocab.add(s)
    LDAModel ← LDAModel(vocabulary, numtopics = 5)
    documentTopics ← ⟨⟩
    for document d ∈ D do
        t ← getTopicwithHighestProp(LDAModel, d)
        documentTopics.add((d, t))
    fraud_dominant_topic ← LDAModel.getLargestTopic()
    unseenDocument ← testSet.getdocument()
    if unseenDocument.dominantTopic == fraud_dominant_topic then
        Label = 1
    else
        Label = 0
```

4 Results and Discussion

4.1 Output Screenshots 1

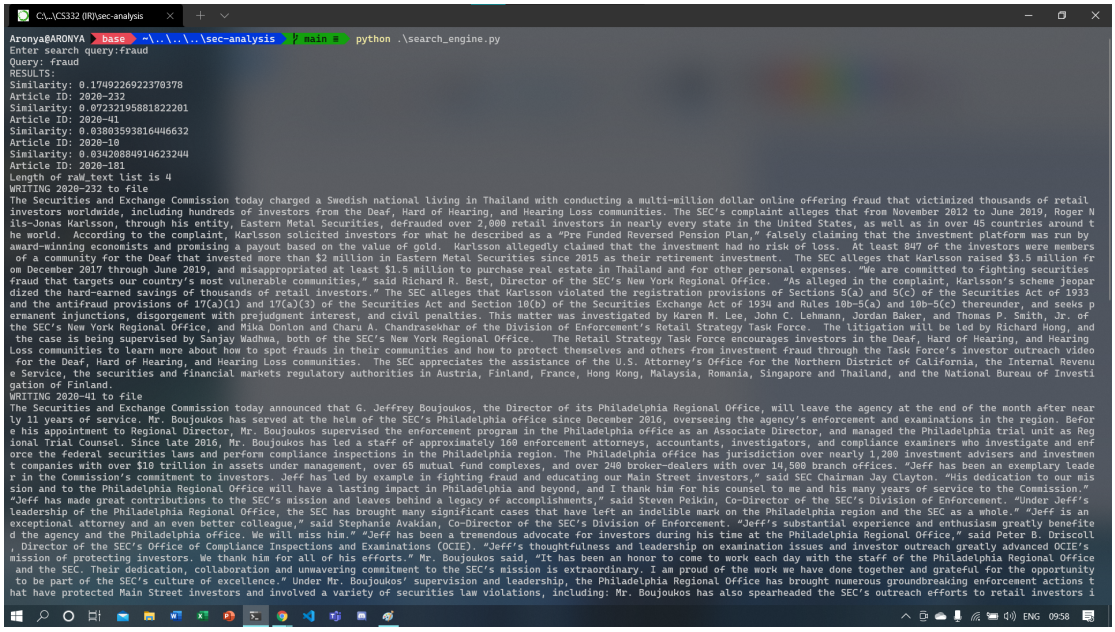

Figure 1: Output for the query `fraud`

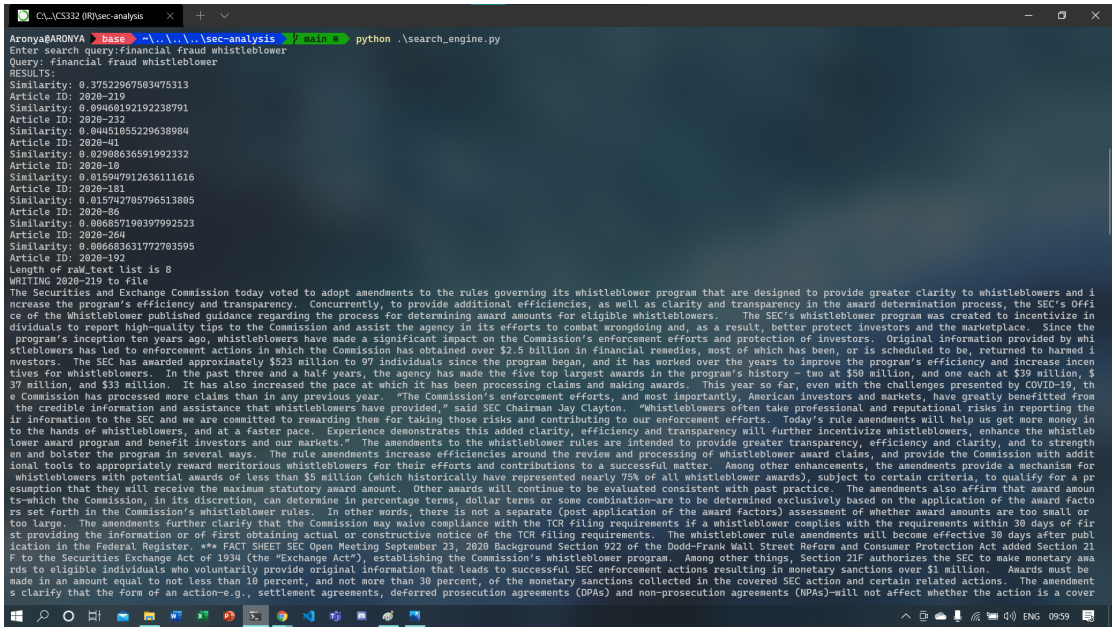

Figure 2: Output for the query `financial fraud whistleblower`

4.2 Description of results of Tf-IDF based retrieval system

From the above results, it is clear that the system is able to retrieve relevant documents with reasonable accuracy when queried. The vocabulary of the document set is a limi-

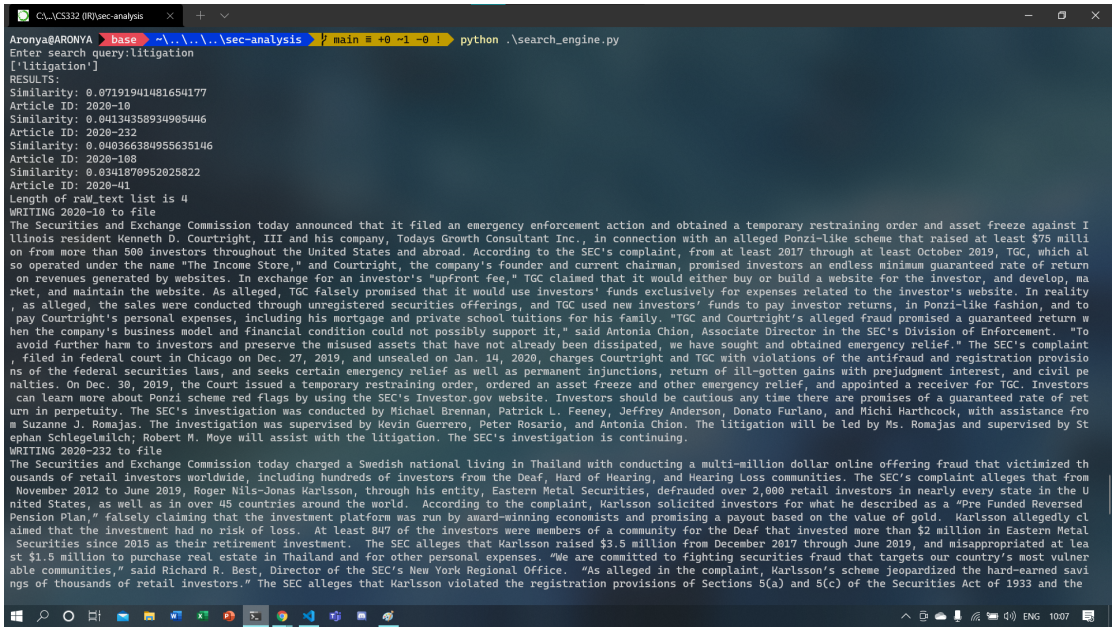

Figure 3: Output for the query `litigation`

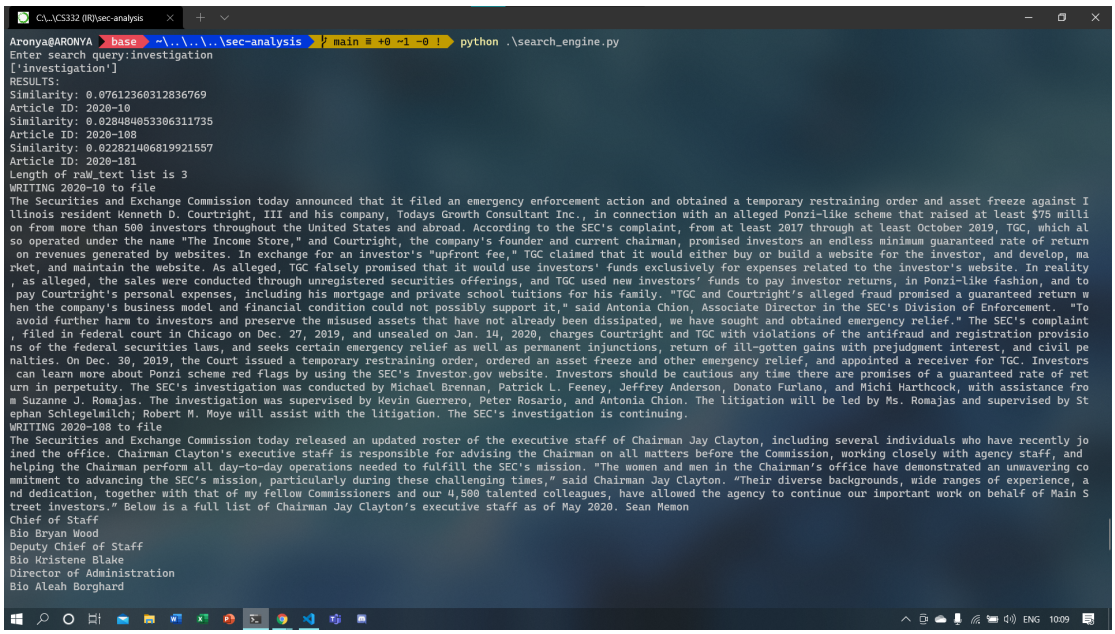

Figure 4: Output for the query `investigation`

Query	**Similarity Value**
"litigation"	0.072
"investigation"	0.076
"fraud"	0.174
"financial fraud whistleblower"	0.3752

Figure 5: Maximum cosine similarity value obtained for a set of representative queries

tation to the overall query performance, as being from an official source, the vocabulary

is very restricted in size (size is only around the order of 10^4 tokens for a set of 2600 documents). However on in-vocabulary words, the system manages to retrieve relevant documents.

From the table shown above, it is clear that the more specific a query becomes, the better is the performance of the system at retrieving relevant documents with regards to the submitted query.

4.3 Output Screenshots 2

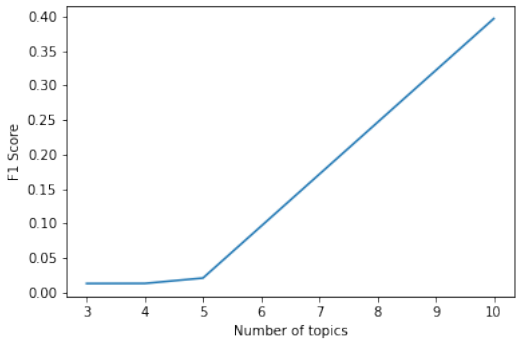

Figure 6: F1 Score for Classification

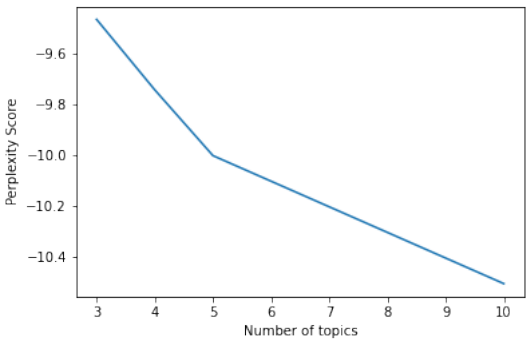

Figure 7: Model Perplexity

4.4 Description of results for LDA-based fraud classification

From the above plots, it is clear that up to a point, increasing the number of topics results in an improvement not only in the quality of the model (as evidenced by the perplexity score) but also the performance of the model on the classification task (as evidenced by the improvement in F1 score)

However, beyond a point, it is clear that the law of diminishing marginal returns affects the model's performance with respect to increasing number of topics. Hence we have chosen 5 topics as a good compromise between the model quality and classificaiton

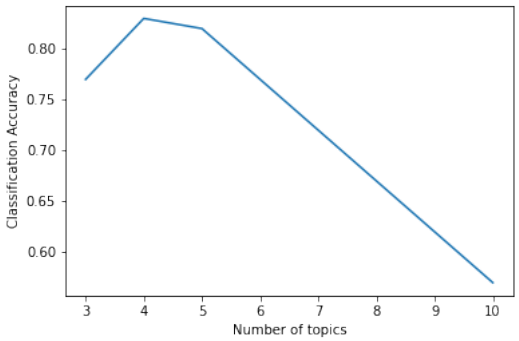

Figure 8: Model Accuracy

performance.
The images below show the term-topic distribution for a varying number of topics

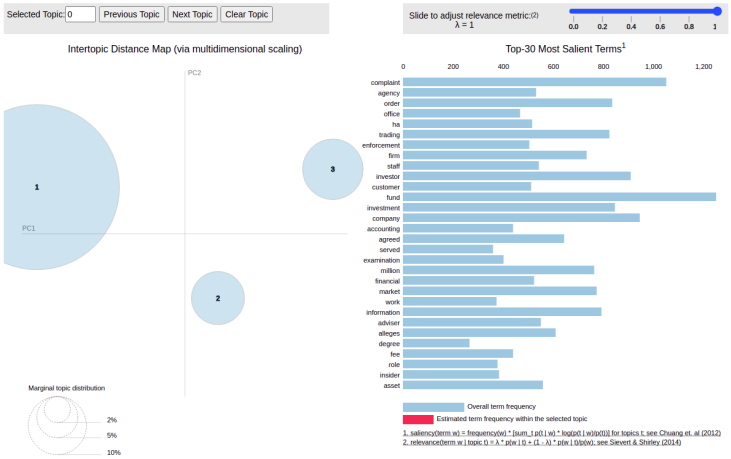

Figure 9: Visualization for 3 topics

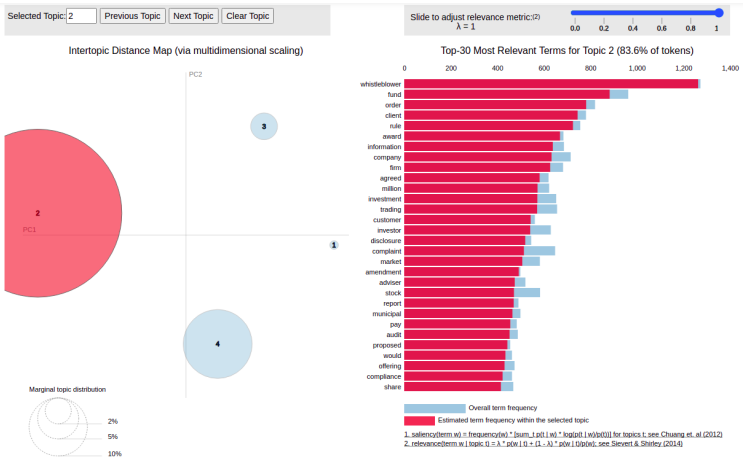

Figure 10: Visualization for 4 topics

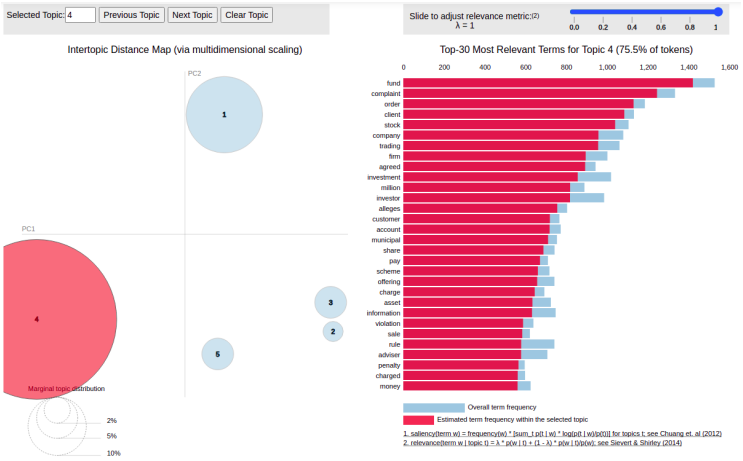

Figure 11: Visualization for 5 topics

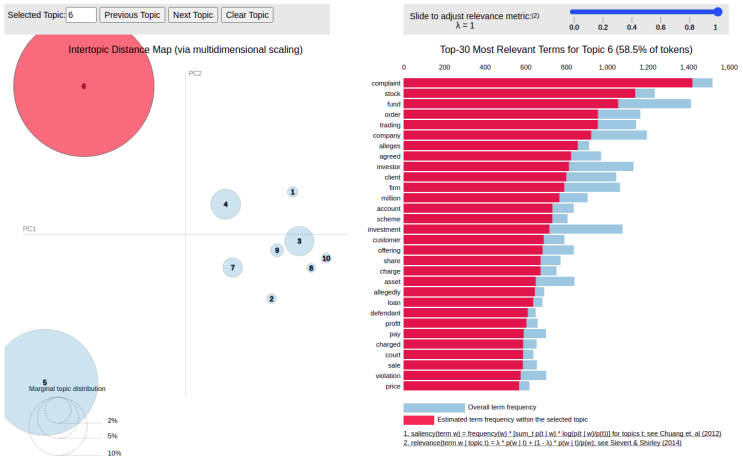

Figure 12: Visualization for 10 topics

5 Conclusion and Future Scope

5.1 Future Scope

The following areas for improvement can be suggested based on our experience of working with the dataset and modelling:

- Explore alternate corpora for financial fraud detection, and use them to either replace or augment the existing corpus of SEC Press Releases.
- Exploring other methods of data cleaning such as stemming, and their impact on the performance of both selected models
- Explore alternate formulae for calculating the TF and IDF scores and their impact on the relevance of the retrieved results
- Explore alternate methods of topic modelling such as latent semantic analysis (LSA) and their impact on the classification accuracy and the term-topic and document-topic distributions obtained.

5.2 Conclusion

This work implements text mining techniques and topic modelling techniques in order to detect instances of financial fraud. The steps involving gathering the data, cleaning the data, modelling the data and analyzing the results were succesfully carried out in the course of this project. While the results leave some room for improvement, the improvement strategies have been outlined in the future scope section. The project for the course Algorithms for Intelligence Web and Information Retreival gave us many ideas as to the real world implementation of the concepts taught in the course, as well as the constraints regarding said implementation, and we have come out of the project experience as more aware students.

We wish to thank Dr. Nagegowda KS, Asst. Professor at the Department of Computer Science and Engineering, PES University, for his unwavering support and insightful guidance in the course of this project.

References

[1] "Analysis of communication patterns with scammers in Enron corpus", Dinesh Balaji Sashikanth, arXiv:1509.00705

[2] "Financial Statement Fraud Detection using Text Mining", Nasib Singh Gill, International Journal of Advanced Computer Science and Applications 2012, DOI: 10.14569

[3] "Building a Semantic document search engine using TF-IDF", Zayed Rais, 2020 https://medium.com

[4] "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents", Shahzad Qaiser, Ramsha Ali, International Journal of Computer Applications 2019. volume 181(1), DOI: 10.5120

[5] "Latent dirichlet allocation in web spam filtering", Istvan Biro, Jacint Szabo, Andras A Benczur, AIRWeb '08

[6] "Financial Statement fraud detection using text mining: a systemic functional linguistics theory perspective", Wei Dong, Shaoyi Liao, Liang Liang, PAICS 2016

[7] "Topic Modeling in Financial Documents", Patrick Grafe, Dept. of Computer Science, Stanford University

[8] "Text mining techniques to detect corporate fraud", silfratech.com

[9] GenSim module for topic modelling applications, PyPI

[10] PyLDAVis for interactive topic model visualizations, PyPI