A Framework for developing construction safety database using machine learning and text mining approaches

A THESIS

Submitted by

Kalagapudi Robin Francis Nanda

CE20M008

in partial fulfillment of the requirements

for the award of the degrees award of the degree

Masters of Technology

in

Building Technology and Construction Management

Under the guidance of

Dr. Nikhil Bugalia

DEPARTMENT OF CIVIL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

CHENNAI-600036

JUNE 2022

THESIS CERTIFICATE

This is to certify that the thesis entitled "**A Framework for developing construction safety database using machine learning and text mining approaches**" submitted by **Kalagapudi Robin Francis Nanda** to the Department of Civil Engineering, **Indian Institute of Technology, Madras,** for the award of the degree of **Master of Technology (M.Tech)** in **Building Technology and Construction Management (BTCM)** is a bonafide record of research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Nikhil Bugalia
Assistant Professor
Department of Civil Engineering
Indian Institute of Technology Madras
Chennai – 600 036.

Prof. R G Robinson
Professor and Head
Department of Civil Engineering
Indian Institute of Technology Madras
Chennai – 600 036.

Place: Chennai

Date: June 2022

ACKNOWLEDGEMENTS

First and foremost, praises and thanks to the God, the Almighty, for his grace for me to have the wisdom, ability, and showers of blessings throughout my research work to complete the thesis successfully.

It gives me a sense of fulfillment to write a few words to acknowledge people without whose support this thesis would not have been possible. I would like to express my deep and sincere gratitude to my guide **Dr. Nikhil Bugalia**, Assistant Professor, BTCM division, Department of Civil Engineering, IIT Madras, for his timely and valuable guidance. His vision, sincerity and motivation have deeply inspired me and his periodic reviews enabled smooth completion of the work. I could not have imagined having a better advisor and mentor for my study.

I would like to thank **Dr. Benny Raphael**, Professor, BTCM Division, Department of Civil Engineering, IIT Madras, the Faculty Advisor of our batch, for his extended support and guidance in the areas of machine learning.

I would like to acknowledge **Dr. Manu Santhanam**, Professor and former Head of Department, Civil Engineering, IIT Madras, for allowing me to undertake this project.

I am extremely grateful to my **parents** and for their love, prayers, caring and sacrifices for educating me.

I would like to express my gratitude to all my friends who have been a source of support and help whenever needed. Special thanks to Hrishikesh who helped me in this study.

Kalagapudi Robin Francis Nanda

ABSTRACT

Keywords: Machine learning, Construction safety, News articles, Text classification, Accident.

Safety is one of the important factors in any workplace to increase quality and employee health by reducing the accidents/injuries. The construction sector being the second largest employer in India, is vulnerable to many accidents and injuries. In spite of these accidents, there is no standard recording and notification system for Indian construction sector. The absence of this recording system is hurting India as the accidents are highly underreported and the reported accidents are scattered and not reconciled due to which there is no sufficient research on construction safety in India. This thesis aims to fill the gap between academia and practical conditions by providing a framework for developing a construction safety database using the online news articles and provide an end-to-end automated solution by using vector space models, topic modelling, machine learning (ML) and named entity recognition. The framework deals with automated approaches of duplicate removal of articles and filtration of irrelevant data. The filtered data undergoes resampling followed by ML based classification and finally information extraction such as date, time, location is done to obtain the information related to the construction accident data. In this study, experimentation is done exclusively on duplicate data removal, filtration of data and ML based binary classification to classify whether the article is construction accident-related article or not. Cosine similarity is used for removing the duplicate data, Guided Latent Dirichlet Allocation (GLDA) is used for filtration of irrelevant data, four resampling techniques SMOTE, ADASYN, SVM-SMOTE and Borderline-SMOTE are used to handle the imbalanced data and five baseline models, K-nearest neighbor, (KNN), decision tree (DT), support vector machine (SVM), extreme gradient boosting (XG Boost) and random forest (RF) are used for ML based binary classification. Experiment results show that duplicate removal is effective and the trained model was able to remove duplicates by retaining greater than 90% of construction accident-related articles. The results also show that GLDA is effective in filtering the irrelevant data and robust to outliers and through binary classification, 93% of accuracy is achieved on the testing data with 80% of the construction-related accidents being retained. The entire model is validated on five cross-validated datasets and found to be robust enough. This framework can be employed not just to construction accident domain as it is generalizable to work on any domain which deals with text data. This study also describes the challenges faced in the framework and proposes an ideal methodology for an end-to-end approach.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVATIONS

ML	Machine Learning
OHS	Occupational Health and Safety
FIR	First Information Reports
GLDA	Guided Latent Dirichlet Allocation
NLP	Natural Language Processing
RegEx	Regular Expressions
SRL	Semantic Role Labelling
RSS	Really Simple Syndication
LDA	Latent Dirichlet Allocation
SVM	Support Vector Machine
RF	Random Forest
DT	Decision Trees
KNN	K-Nearest Neighbors
BoW	Bag-of-Words
TF-IDF	Term-Frequency-Inverse-Document-Frequency
SMOTE	Synthetic Minority Oversampling Technique
PCA	Principal Component Analysis
XG Boost	Extreme Gradient Boosting
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Networks
POS	Parts of Speech
NER	Named Entity Recognition
NLTK	Natural Language Toolkit
GPE	Geo Political Entity
WCSS	Within Cluster Sum of Squares
SVC	Support Vector Classifier

CHAPTER 1 INTRODUCTION

1.1 Background and Motivation:

The construction sector has been an engine of global economic growth over the decades and continues to be a dominant sector for employment generation across different parts of the world. The construction sector in India, being the second largest employer in India after agriculture, is complex involving a lot of manpower and technology. Unfortunately, the construction sector is also infamous for poor Occupational Health and Safety (OHS) performance worldwide and specifically in developing countries such as India (Manu et al., 2019). The Indian construction labor force is 7.5% of the world's labor force but contributes to 16.4% of fatal global occupational accidents (Kulkarni, 2007). The construction sector in India is very hazardous, with many accidents and injuries taking place daily. Rough estimates, expected to be gross underestimations, suggest that 48,000 workers die yearly due to occupational accidents, with nearly 38 accidents occurring every day in the construction sector (Patel & Jha, 2016). Workplace deaths in India are 20 times higher than in Britain (Business and Human Rights Resource Centre, 2017). The occupational health and safety situations are poorer in India than in the world, resulting in more serious accidents (Samanta & Gochhayat, 2021).

To prevent workplace accidents, analysis of past accidents is crucial ((Zhang et al., 2019)). In India, despite the accidents, there is no unique standard recording and notification system for the Indian construction sector. The system exists, but implementation is an issue since the reports are scattered and not reconciled (Patel & Jha, 2016). In addition to this, accidental construction statistics are not easily available and highly underreported, leading to a situation where proper attention to safety is not paid. The unavailability of robust occupational health and safety (OHS) statistics for the construction sector in India is a systemic hurdle facing academia and practitioners (NDTV, 2017; Patel & Jha, 2016).

Previous knowledge and experience of accidents are highly valuable and could contribute to avoiding similar accidents in the future (Zou et al., 2017). The typical sources of OHS statistics are government departments and agencies, research studies, First Information Reports (FIRs) in police stations, medical and insurance registers, leading newspapers, and online search engines (Patel & Jha, 2016). Government agencies have no systematic efforts to collect and publish such statistics. Resource limitations often affect the research studies on the area, and

the information collected through small-scale surveys is challenging to generalize. FIRs, medical registers, and insurance databases are difficult to access and often do not contain statistics directly attributable to the construction sector (Patel & Jha, 2016; Yadav et al., 2020). Online search engines have the ease of availability, but we cannot rely on search engines as it contains some unauthorized and fake news. Hence, robust OHS statistics and trends covering wider geographic areas for India are yet to be developed (Patel & Jha, 2016).

In the absence of formal databases, online newspaper articles have been recognized as an essential source of information for developing OHS statistics, considering the feasibility, availability in digital format, and the quality of information from the data (Chiang et al., 2018; Patel & Jha, 2016). Over the world, more than two million articles are published every day on the web, and the present global size of online news websites is more than 200 million (Singh, 2021). However, even globally, no previous study has attempted to leverage the large-scale online news data for developing safety statistics for the construction sector (Chiang et al., 2018; Patel & Jha, 2016). Resource-intensive manual processing approaches have been used to leverage newspaper articles to obtain OHS statistics in construction. At the same time, the lack of credible data is widespread across the developing world (Chiang et al., 2018). Utilizing efficient data processing approaches such as Machine Learning (ML) and text mining for construction-related news articles remains scarce (Barberá et al., 2021; Koc & Gurgun, 2021; Ninan, 2021). There have been well-established efforts to analyze safety-related textual data using ML and text-mining approaches (Baker et al., 2020; Goh & Ubeynarayana, 2017; Kedia et al., 2021).

1.2 Problem Definition

As discussed in the above section, OHS statistics are poorer in India, resulting in hazardous workplaces and construction accidents that continue to occur frequently. Despite the accidents, there are no effective safety measures and no proper legal record of deaths of construction workers in India. The information available from the various resources is staggered and not reconciled. There is no standard recording and notification system for developing safety statistics for construction accidents in India. A substantial amount of data is available from the leading newspapers, which is not used effectively. This can be extracted and used to develop statistics related to construction accidents.

1.3 The objective of the study

The main objectives of the study are:

- Using newspaper articles, create an end-to-end automated process framework
- To identify and remove the duplicate data and refine the data by removing redundant data
- To classify news articles pertaining to construction accidents and injuries and generate a database for construction accidents and injuries in India

1.4 Scope of the project

To develop and establish an end-to-end automated process framework that can be operationalized for trend extraction in construction safety using newspaper data. The framework presented will be helpful for researchers to understand the information flow and associated challenges in processing newspaper articles for all domains. The current study also provides an in-depth discussion on challenges faced in the operationalization of the end-to-end process framework. The current study may serve as an essential reference document for research across the globe for utilizing newspaper data through automated ML approaches.

1.5 Thesis organization

Chapter 1 discusses the background of the Indian construction industry, with a focus on workplace safety and construction accident data.

Chapter 2 reviews the various literatures in this field and develops a framework based on the literatures.

Chapter 3 presents the dataset used in the framework and metrics used for the evaluation of data.

Chapter 4 gives an overview of vector space models used for duplicate data removal and the methodology for performing duplicate data removal using cosine similarity.

Chapter 5 shows the challenges faced in the application of unsupervised and semi-supervised learning approaches for binary classification of dataset and presents a modified framework based on the challenges.

Chapter 6 gives an overview of Guided LDA and implementation of Guided LDA on our dataset.

Chapter 7 gives an overview of the resampling techniques and machine learning classification models.

Chapter 8 shows the results and analysis of Guided-LDA, Resampling and ML based binary classification along with cross validation results.

Chapter 9 summarises the study and presents the conclusion along with the scope for further advances in the study area.

CHAPTER 2 LITERATURE REVIEW AND METHODOLOGY DEVELOPMENT

Development of the framework has started by doing preliminary studies and literature reviews. Data collection methods, data cleaning and pre-processing, converting text data into vector format, use of machine learning in the data, named entity recognition, regular expressions, etc., have been explored in various literature reviews. Then while working on the data, it is understood that there will be duplicate data, and studies are done to remove the duplicate data. In this way, all the dots are connected by building a framework that can give a comprehensive end-to-end solution.

2.1 Studies were done on news articles in the construction domain

Only a handful of previous studies have relied on automated ML and text-mining approaches to analyze newspaper data for construction safety. One of the most comprehensive analyses of fatal accidents in the construction sector using newspaper data has been presented by (Chiang et al., 2018). Using statistical approaches such as cluster analysis and principal component analysis, they obtained the date and time of the fatal accidents, demographic information such as the age and gender of the victim, and the type of accidents such as falls from height, strikes from local news reports. Overall, the study provides comprehensive ideas on processing the newspaper reports; however, the data collection and entity extraction processes adopted in the study were manually expensive.

Feng & Chen (2021) have proposed a natural language data augmentation-based framework for automatic information extraction using deep neural networks. They relied on accident news reports for the Chinese construction industry. However, the articles used for their study were handpicked, and their framework can only be applied for small datasets extracted manually and not for data containing many reports. Needless, they emphasize the necessity of a robust automatic information extraction model capable of handling large volumes of data (Feng & Chen, 2021).

The challenge of automatically extracting large quantities of construction safety data from newspapers has been partially addressed (Kim et al., 2021). They relied on a web-crawling technique to collect news articles from *The New York Times* using “construction accident” as a keyword. Thereafter, using big-data analytics techniques such as word embedding and network analysis, Kim et al. (2021) comprehensively explored factors and interrelationships that affect fire-related accidents in construction. Such an effort is commendable as it can

provide new insights into managing fire hazards. However, such an approach is not suitable for obtaining Spatio-temporal statistical trends, which are also essential to guide country-wide safety management programs (Chiang et al., 2018). A further limitation of the web-crawling technique and its applicability for developing countries are discussed later.

Overall, there is a scarcity of literature focussing on developing end-to-end automated approaches for extracting articles, processing them, and developing Spatio-temporal statistical trends from them for the construction sector, while the necessity of such a framework has been well recognized in the literature (Chiang et al., 2018; Feng & Chen, 2021)

2.2 Studies were done on news articles other than the construction domain used to support the development of the framework

Pablo Barbera et al. (Barberá et al., 2021) have used automated approaches for text classification of news articles and advocated the use of keyword searches rather than predefined subject categories provided by news archives. Ritika Singh et al. (Singh, 2021) has used cosine similarity, Jaccard similarity, and Euclidean distance on the news articles to calculate the text-similarity score between bilingual news articles. These studies paved the way for developing the framework for inputting news articles and removal of duplicate data.

Subhayu et al. (Chakravorty et al., 2015) have used topic modeling and named entity recognition for analyzing murder-related unstructured newspaper data. Gilberto Rivera et al. (Rivera et al., 2020) have used NLP and supervised ML algorithms with resampling techniques to the news provided by the RSS service in order to classify whether they are about a traffic incident or not with the final intention of notifying the citizens where the accidents occur. Chia-Mei et al. (Chen et al., 2019) apply machine learning and topic modeling with Latent Dirichlet Allocation (LDA) to online cybersecurity news in order to detect cybersecurity trends. Basanta et al. (Chaulagain et al., 2019) developed a system for automatic extraction and visualization of casualty information from news articles by extracting entities of the number of deaths, injuries, date, location, and vehicles involved using NLP, Regular Expressions (Regex), and Semantic Role Labelling (SRL). These studies paved the way to develop the framework for primary classification of articles, entity extraction, and topic modeling.

The majority of the previous research relies on supervised learning approaches for classification (Goh & Ubeynarayana, 2017; Rivera et al., 2020; Zhang et al., 2019). Such supervised approaches are efficient but are manually intensive (Barberá et al., 2021). It is evident from the previous studies that only one specific part of the whole framework has been

done, but none has done the complete end-to-end solution for this problem. Considering these studies, a complete end-to-end solution is bought up, proposing a framework that can bridge the gap of previous studies and develop safety statistics for the construction sector on large-scale online news data. The details of the framework and its components are detailed in the next section.

Table 1: Summary of the literature review and challenges to be faced for implementation

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
1	**News Classification for Identifying Traffic Incident Points in a Spanish-Speaking Country: A Real-World Case Study of Class Imbalance Learning** (Rivera et al., 2020)	Classify whether the data is about a traffic incident of not with the final intention of notifying citizens where such accidents occur	1. Getting the corpus of news by using RSS of that newspaper 2. Do vector characterization by using improved Bag-of-Words and normalize using the Tf-Idf method 3. Select features through a mutual information-based method 4. Train a supervised learning model 5. Classify the reports in real-time 6. Process the text of RSS reports retrieving the location 7. Notify users about the events on a map	NLP • Bag of Words Machine Learning • Classification and Regression Tree • Naïve Bayes • KNN • Random Forest • Support Vector Machine Sampling Algorithms • Synthetic minority oversampling technique (SMOTE) • Borderline SMOTE • Adaptive synthetic sampling • Random oversampling	Language – Spanish Data type: Online News articles Raw format: RSS Corpus size: 18,386 news articles, out of which 1894 were identified as traffic accidents	Permission to download and use RSS format news articles Availability of NLP libraries in native languages Enabling Domain-Specific features Loss of the information in articles that have multiple geolocations in the same article

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
				• Random under sampling		
2	A Study on Security Trends based on News Analysis (Chen et al., 2019)	To extract online cyber security news and apply ML methods to detect the emergence of cyber security events and trends	1. Using a web crawler to extract news from multiple sources and text mining with Latent Dirichlet Allocation (LDA) 2. Data cleaning, pre-processing, and selection of features 3. Data analysis with a topic model to find the main topics of recent news 4. Clustering of data to find new events under each topic 5. Presentation of new cyber security events in the web form for cyber	Text mining • LDA Selecting features • Extracted manually • TF-IDF Machine Learning • K-Means	Language – English Data type: Online news articles Corpus size: 3112 news articles	No domain available to directly extract the accident-related news without classification from the other news types

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
			security administrators			
3	Fatal Construction Accidents in Hong Kong (Chiang et al., 2018)	To do a comprehensive analysis of fatal accidents and provide insight for future research on solutions to reduce accidents in the construction industry	Data extraction: Keywords search in local and regional newspapers and manually select the data one by one Analyzing the common news from different newspapers Additional data collected from PRESS regarding the hour and day, gender, age, type of accident, and type of sector (private or government) Delete missing data Analyzing the data with PCA and cluster analysis	Machine Learning • PCA The algorithm used for cluster analysis is not mentioned	Source: local news search engines Period of data: 2006 to 2015 Corpus size: 256 cases Data analyzed after removing missing data: 225 cases	Manual extraction of data is time-consuming Getting additional data from PRESS is not viable

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
4	Arabic Text Classification of News Articles Using Classical Supervised Classifiers (Al Qadi et al., 2019)	To automatically identify the category of a document based on its linguistic features	Data extraction: web scraping (Python Scrapy) Data cleaning by removing Latin symbols, custom-made stop words, punctuations Text featuring by converting text into vectors Classification of the news articles with 11 Machine learning classifiers Evaluate the performance Test the best classifier with testing data Testing the data with an unbalanced dataset	NLP • Bag of Words • TF – IDF Machine Learning • Logistic Regression • Nearest Centroid • Decision Trees • Support Vector Machines • K – Nearest Neighbours • XG Boost • Random Forest Classifier • Multinomial Classifier • Ada – Boost Classifier • Multilayer perceptron • Voting Classifier	Source: Online websites Language: Arabic Corpus size: 89189 articles with more than 32.5 million words	Web scraping for online Indian news-papers is not permitted Unavailability of such a huge corpus with the labeling of articles
5	Classification of Indonesian News Articles based on	To overcome the limitations of conventional	Collection of news documents Text pre-processing Bag-of-Words	NLP • Bag of Words Machine Learning • LDA	Source: Local	Since it is a complete unsupervised learning algorithm, the

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
	Latent Dirichlet Allocation (Kusumaningrum et al., 2016)	methods and use LDA for news articles classification	LDA Classification model Kullback-Leibler Divergence Class prediction	Performance Metrics • KL Divergence	newspaper articles Language: Bahasa (Indonesian) Corpus: 100 articles divided into five classes	distribution of documents into the topics may not be efficient and as expected.
6	News Text Classification Based on Improved Bi-LSTM-CNN (Li et al., 2018)	To avoid data sparseness and dimension explosion and investigate the application issue of NLP in text classification by using the Bi-LSTM-CNN method	1. Collection of data 2. Text featuring using word2vec 3. Implementing Bi-LSTM-Layer 4. Model training using Deep Learning CNN algorithm 5. Classification of news text	NLP • TF-IDF • Word2Vec (CBOW) • LSTM Machine Learning • SVM Deep Learning • CNN	Corpus size: 65000 news articles divided into ten categories with 50k training set, 5k validation	The majorly occurring words may hold the decision making, and if they are removed, there may be a loss of semantics This requires extensive training data and

S.No	Reference paper	Objective	Process	Algorithms used	Kind of data	Challenges
					set, 10k testing set data	doesn't work well in imbalanced data
7	A guided latent Dirichlet allocation approach to investigate real-time latent topics of Twitter data during Hurricane Laura (Zhou et al., 2021)	To investigate real-time latent topics of Twitter data during Hurricane Laura. To quickly identify and extract situational awareness information to responders, stakeholders, and the general public to adopt timely responsive strategies and wisely allocate resources during hurricane events	Data collection (Using Twint in Python) Data pre-processing: lower case, stop words, hyperlink, hashtag, POS, Tokenize, Bi-gram, TF-IDF Guided LDA – Topic candidates, coherence model, visualization Latent topic cluster selection	NLP • TF-IDF Machine Learning • Guided-LDA	Language – English Source: Twitter	The challenge lies in tweets containing information across multiple events, so it is difficult to analyze the topics solely for hurricane Laura.

2.3 The framework was built based on the literature review

Based on the previous literature, the key components are identified, and a whole end-to-end process is bought up by connecting different components in the framework. Basically, the framework can be broadly classified into five components of tasks, i.e., Input news articles, Duplicate removal, ML-based binary classification, ML-based accident classification, and Entity extraction. Finally, the output attained will be a construction accident database containing the date, location, time, and type of accident. A general description of the methodology adopted in the proposed study and the underlying procedures have been represented as a flowchart in Figure 1.

2.3.1 **Input news articles:**

Most previous studies on the topic rely on readily accessible news sources like Google, Really Simple Syndication, and web crawling (Barberá et al., 2021; Feng & Chen, 2021; Kim et al., 2021). However, some media companies restrict crawling or limit the amount of crawling (Kim et al., 2021). Especially, Indian media companies do not accept crawling their websites. We tried contacting some Indian media companies to permit us to crawl their respective websites, but they did not accept the request. To extract data on a large scale, leveraging the services of specialized agencies in media monitoring and analytic tools would be useful. The extraction of data can be based on keywords search or pre-defined subject categories provided by news archives.

2.3.2 **Duplicate Removal:**

The extracted input data obtained may contain duplicates. The duplicates are of different types, such as multiple media houses reporting the same events with a similar style of writing or the same article represented in multiple media houses. For such articles, vector space models such as cosine similarity, Jaccard similarity, and Euclidean distances are used to calculate the text-similarity and remove the duplicates. These vector space models are discussed thoroughly in CHAPTER 4.

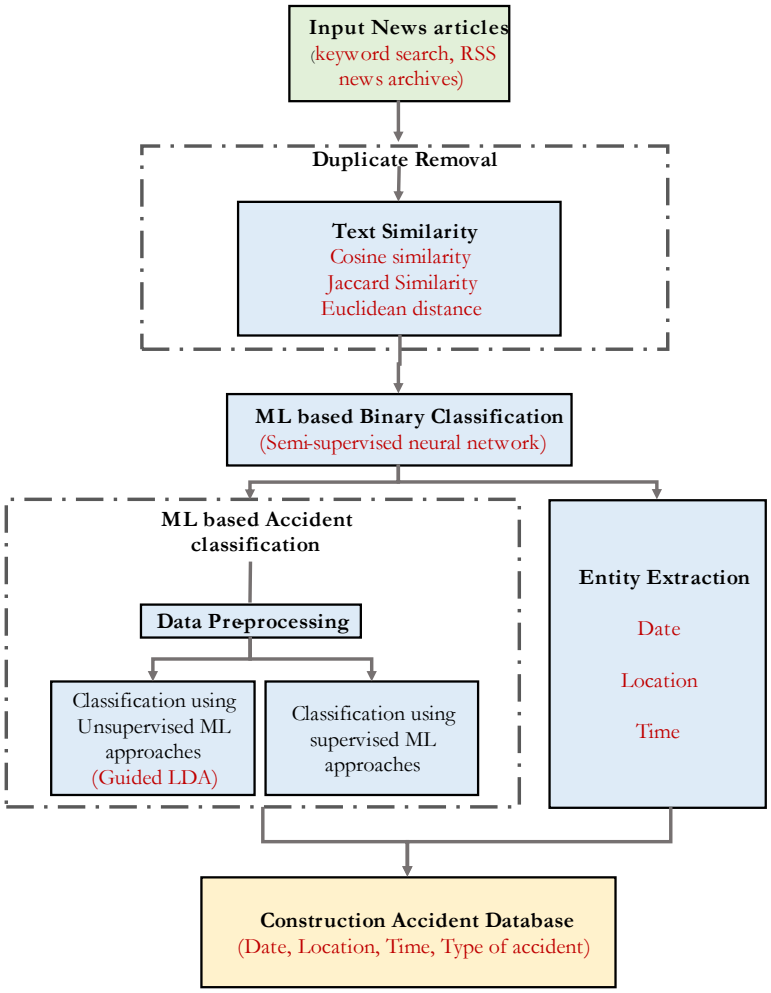


Figure 1: Proposed framework for developing construction safety database

2.3.3 **ML-based binary classification:**

To classify whether the extracted news article is related to a construction accident or not, we need a machine learning approach. Labeling the information for large-scale data requires high time and labor costs, and there should be close cooperation with domain experts and heavy reliability. Other approaches are unsupervised clustering approaches which require minimal manual efforts and are typically efficient for large volumes of data. However, when we are dealing with domain-specific data extracted through a keyword search or predefined subject categories from the preliminary studies, it is understood that they do not provide good results due to a high degree of semantic similarity, which makes the classification difficult in the vector space. Hence, to balance a high degree of automation and filtering efficiency, a semi-

supervised ML approach is necessary. After doing preliminary studies, it is understood that the unsupervised and semi-supervised ML models such as label propagation, label spreading, and self-training do not provide good results (see CHAPTER 5). The study proposes a trained supervised model which can be used to perform binary classification on the future datasets.

2.3.4 **ML-based accident classification:**

The filtered data should be classified based on the type of accident. Similar to the filtering, many studies utilizing ML use conventional supervised ML approaches (Barberá et al., 2021) and, to some extent, unsupervised clustering approaches such as Latent Dirichlet Allocation (LDA). There are significant limitations in the clustering implemented through LDA, as the clusters identified are not always human interpretable (Jagarlamudi & Daum, 2012). Guided-LDA (Lemaire et al., 2019) is a semi-supervised method that offers improvements over conventional LDA and allows users to seed multiple classification categories using domain-specific keywords, which helps in obtaining human interpretable clusters. The Guided LDA technique is slowly gaining momentum with applications in various multi-class classification problems related to safety domains (Ahadh et al., 2021; Zhou et al., 2021). The supervised ML approaches can also be used for training the ML model, but it does require a substantial amount of data to train, which may not be available if the amount of construction accident-related data is less.

2.3.5 **Entity Extraction:**

The main entities which can describe the accidents are date, location, and time. In most news articles, the date of the accident is not mentioned, whereas the day of the week is mentioned. This day of the week and date of reporting is used to calculate the actual date of the incident. The time of the incident is mentioned in three ways, 12hour format, 24hour format, and period of the day (morning, evening). Named entity recognition is used to extract location, time, and day. But the named entity recognition is not enough to extract location, time, and day). Further, as an extension, Regular expressions are used to extract the entities which are derived from named entity recognition (NER). Considerable study is not done in this area and the preliminary studies done are attached in APPENDIX A.

2.4 Studies were done on duplicate data removal:

(Zou et al., 2017) proposed a combined approach of two NLP techniques, i.e., Vector Space Model (VSM) and semantic query expansion, and outlined a framework for a risk case retrieval system. His study was mainly to retrieve useful information from the database quickly and

accurately. For his study, he used tokenization which is a process of chopping an article into pieces called ‘tokens’ followed by converting words into lowercase, lemmatization, stop word removal, and establishing risk case corpus. In this study, along with the stop words identified by Natural Language Toolkit (NLTK), a list of manually selected words from the top 100 words that have the most occurrences in the corpus but are identified with little value.

(Singh, 2021) measures the similarity between two same news items in two languages – Hindi and English, which refer to the same event. In their study, they used cosine similarity, Jaccard similarity, and Euclidean distance measure to calculate the news similarity score.

(Baraniak & Sydow, 2018) worked on tools to detect the information bias, which happens when multiple publishers publish the same type of information with a bias, and they analyzed the information bias. The methods of cosine similarity, Euclidean distance, Jaccard coefficient, Pearson coefficient of correlation, and averaged Kullback-Leibler Divergence are used for text similarity. A machine learning approach to determining similar articles automatically was also developed.

(Gibson et al., 2008) mentioned that for automated processing such as named entity recognition and visualization, redundant data can cause incorrectly weighted results, thereby skewing automated text processing applications. In this paper, they made a system called CEDAR which means Content Extraction and Duplicate Analysis and Recognition, and had divided their problem into a two-approach problem by creating a method for extracting news article text from webpages that is not website specific and then using the extracted content to identify pairs of documents with same news article content by calculating a resemblance score based on a technique called ‘shingling’ which is similar to Jaccard Similarity discussed in 4.3.3.

(Abid et al., 2020) highlighted that every newspaper has different news reporters and editors, so there is a chance of the same news presented using a different set of vocabulary and writing style of the reporter and editor, which results in duplicate news, failing to which may result in inconsistent statistics obtained after pre-processing the news text. They also mentioned that this problem becomes more pertinent when we deal with human loss news involving crime and accident-related news articles. In their study, they applied the Jaccard coefficient and cosine similarity to calculate similarity scores of any two news articles by using some data pre-processing techniques like stemming, stop word removal, diacritic word removal, tokenization, and cleaning HTML tags in the text.

CHAPTER 3 DATASET USED IN THE FRAMEWORK

3.1 Data Acquisition

The news articles which are to be input into the model are acquired from specialized agencies in media monitoring. The news articles are available in two types which are print data and online data. Print data is acquired from the printing newspapers such as The Hindu, Times of India, Indian Express, and many more. Online data is the data extracted from magazines, articles, online newsletters, online news websites, etc. For our study, we used data acquired from print data since it is more established and reliable. The newspaper articles can also be acquired in multiple languages, but for our study, we have only considered the English language and articles from Indian publications since foreign publications will only cover major construction accidents, which will obviously be covered in the Indian publications. The extraction of data can be based on keyword search or pre-defined subject categories provided by news archives. The keyword search type of extraction is more feasible because there is no specific subject category that contains construction accident newspaper articles. If we consider a broader subject category, then the proportion of the number of construction accident news articles will be very lower compared to the keyword search type of extraction. Hence, the keyword search type of extraction is chosen in this study.

The keywords given to the agency are ‘Construction’, ‘Accident’, ‘Injur*’, ‘Fall’, ‘Collapse’, ‘Struck’, ‘Dead’, ‘Worker’. The keyword ‘Injur*’ refers to any word which starts with ‘Injur’ like injury, injured, injure, etc. In the given keywords, the ‘Construction’ keyword is kept as a mandatory keyword along with other mentioned keywords as secondary keywords. So, any article which has the ‘Construction’ keyword and any of the other secondary keywords will be extracted. This condition is given to ensure that we get data that is specifically related to construction accidents and not all types of accidents. The other reason for selecting this condition is that if the ‘Construction’ keyword is also considered as a secondary keyword and there is no such mandatory keyword, the corpus will become huge due to the wide range of articles pertaining to construction. The mentioned keywords are selected by doing a preliminary study on hundred manually collected construction accident news articles from 17 different newspaper sources. The details of the newspaper sources are attached in the Appendix.

For our study, using the above-mentioned keywords, we bought data for a time period of seven months between July 2021 to January 2022.

3.2 Data Exploration

3.2.1 The original data

The raw format is given by the specialized agency in JSON format. JSON is abbreviated as JavaScript Object Notation which is a language-independent format used as syntax for storing and exchanging the data. Data is converted into excel format for better user interface, visualization, and operations on the data. The sample of raw data after converting into excel format is shown in Figure 2 below. The data has information related to the article's published date, name of publication, edition, publication type, headline, content, type, and name of the source, zone, and webpage link of the article. For further study, only the published date, headline, and content are filtered from this data. The headline and content are concatenated into a single cell for ease of operation.

Publish date	Publication	Edition	Publication type	Headline	Content	Source	Source name	Zone	Link
2021-11-04	The Times of India	Mumbai	Mainlines	Uran flyover pier collapse: Construction co engineers, officials booked for negligence	Construction co engineers, officials booked for negligence George Mendonca Navi Mumbai: Following the death of one Worker and injuries to six others due to the COLlapse of a flyover plier at Jasai Naka in Uran on Tuesday evening, police have taken suo motu cognisance of the incident and registered an FIR against two engineers and	Journalist	George Mendonca	West	https://clientportal.conceptbiu.com/mv/ad/4336-217249142-4336
2021-12-05	The Indian Express	Mumbai	Mainlines	One worker dead, another hurt while laying BMC pipeline	One worker dead, another hurt while laying BMC pipeline EXPRESS NEWS SERVICE MUMBAI, DECEMBER 4 A34-YEAR-OLD worker died and another one was injured in an accident during the construction work of laying of new water supply Pipeline work near Flora Fountain in south Mumbai. The work was	Agency	Bureau	West	https://clientportal.conceptbiu.com/mv/ad/4336-217959338-4336

Figure 2: Sample of data given by the specialized agency

The total seven months of data have 11208 rows which mean 11208 newspaper articles that are published by different publications all over India.

3.2.2 Distribution of articles based on words and characters

In python libraries, when dealing with data containing elements, the 'length' term is used. The length method is used to find the length of any object. For our data, the length of the row gives the count of the number of characters in each row. The distribution of a number of articles according to the article length is calculated, and from Figure 3 below, it is evident that most of the articles (approximately 90%) fall in the range of article length of 0 to 7500.

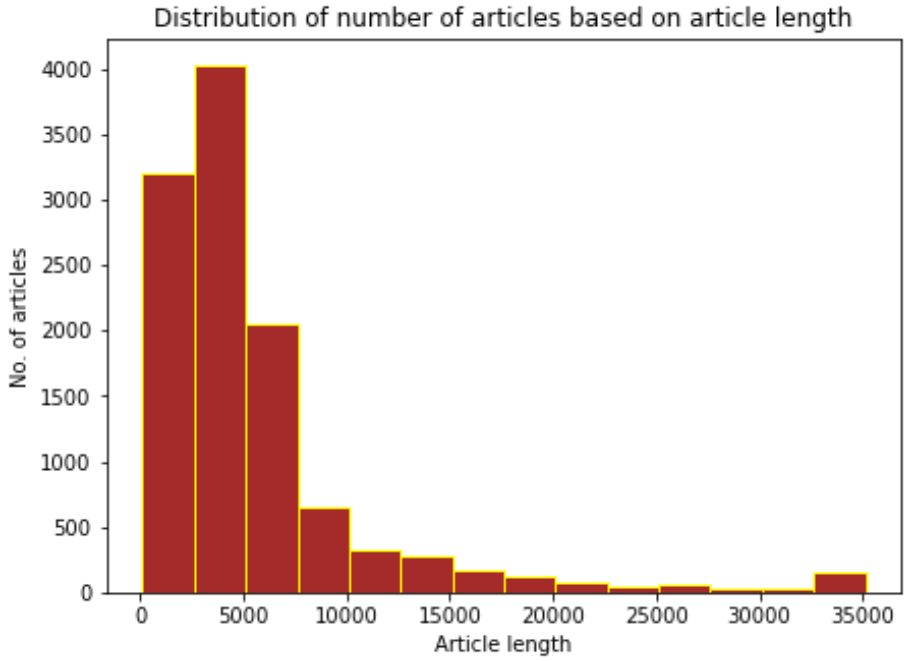


Figure 3: Distribution of article lengths over the corpus

It is important to know the count of words present in an article, and the distribution of articles based on word count is shown in Figure 4 below. The distribution depicts that most of the articles are in the range of 250-to-750-word count per article, with more than 75% of articles in the range of 0-100 words per article.

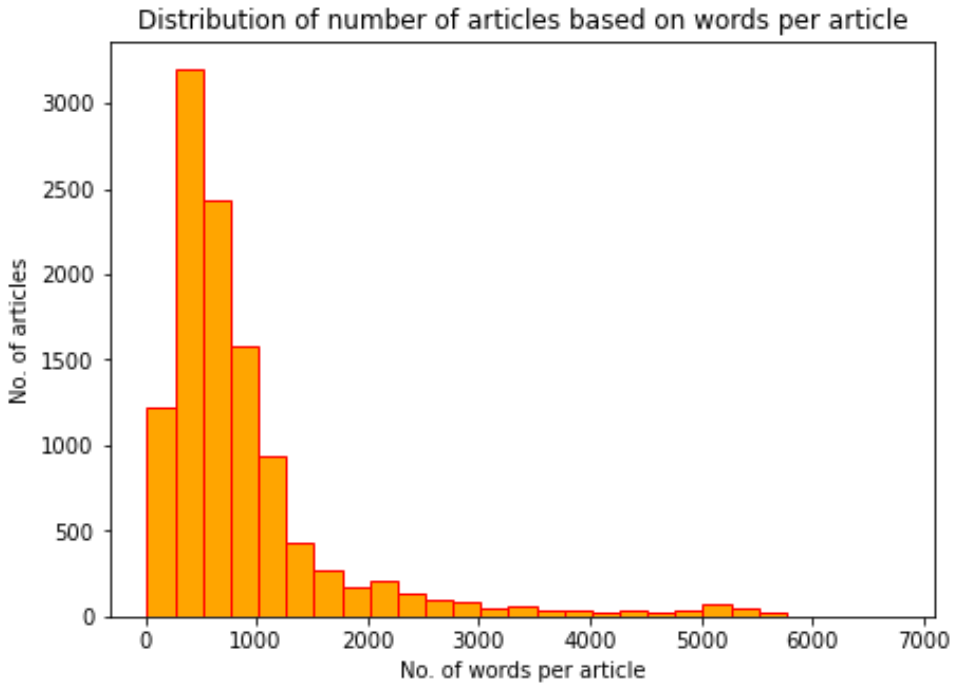


Figure 4: Distribution of word count per article over the corpus

3.2.3 Most occurring words in the data

To further explore the data, the frequency of words in the corpus is calculated. From Figure 5 below, it is observed that the high-frequency words are the most commonly used words, and they do not carry much weightage in developing the model. Hence, to eliminate these words and improve the speed and performance of the model, natural language processing (NLP) techniques are used. NLP has 'nltk' python library, which consists of a set of commonly occurring words known as stop words.

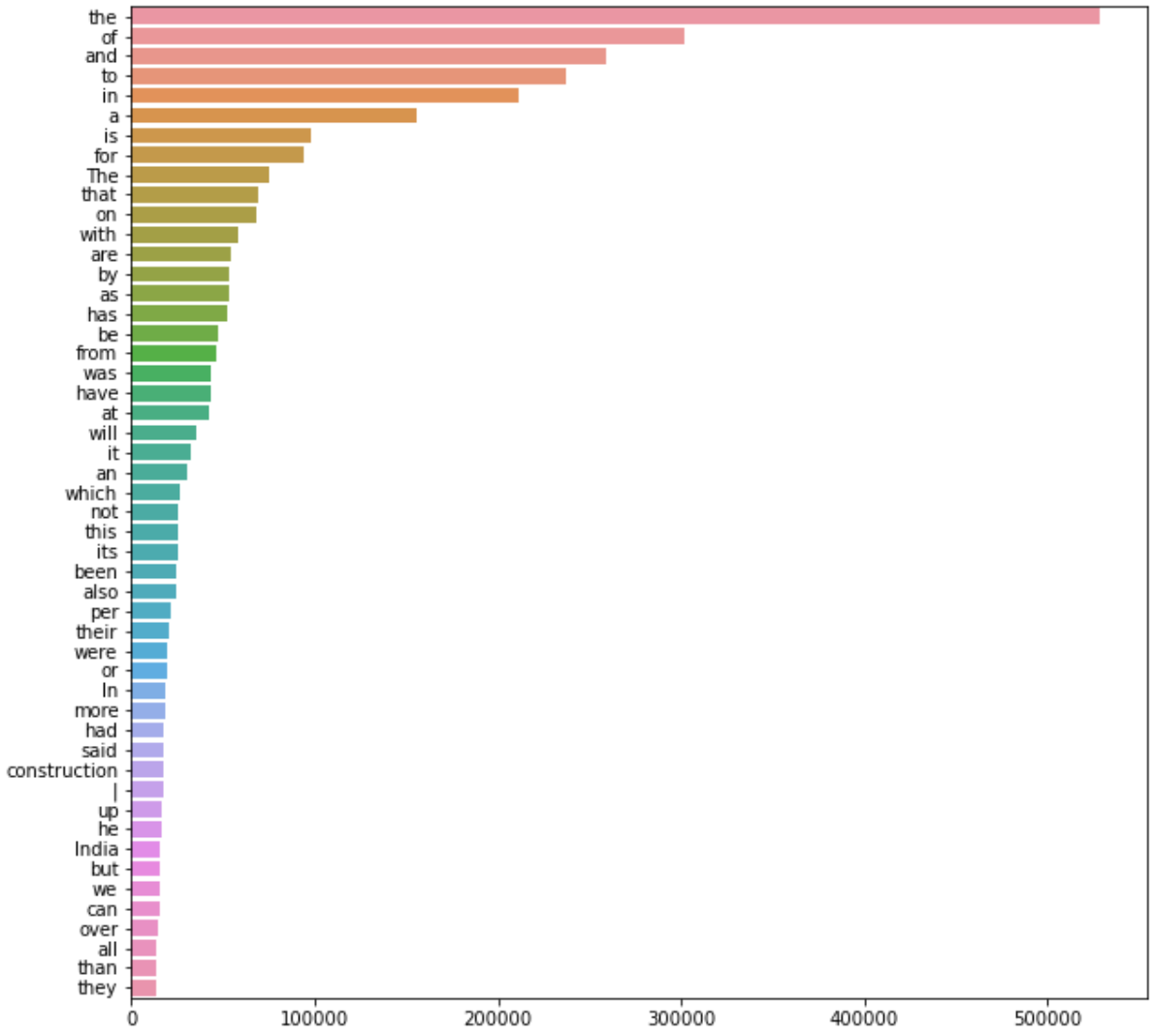


Figure 5: Most occurring words in the corpus

There are also high chances of the same word appearing in different tenses in the corpus. For example, word change has other forms such as changing, changes, changed, changer, etc. The root word of these words is ‘chang’, which we can get after doing a process using NLP called stemming. But sometimes, the resultant word may not have any meaning like ‘chang’ in this case. So, lemmatization is a way of having a meaningful representation of a word after reducing it to its root word. After eliminating these stop words, the lemmatization technique reduces the vocabulary and has a meaningful representation of words. The most occurring words are checked to understand the data. Figure 6 below shows the most occurring words after removing the stop words and lemmatization. Some of the most occurring words, such as ‘said’, ‘such’ ‘new’, have no value and may certainly affect the machine learning models used in the framework. These words can be gathered and should be added to the stop words list.

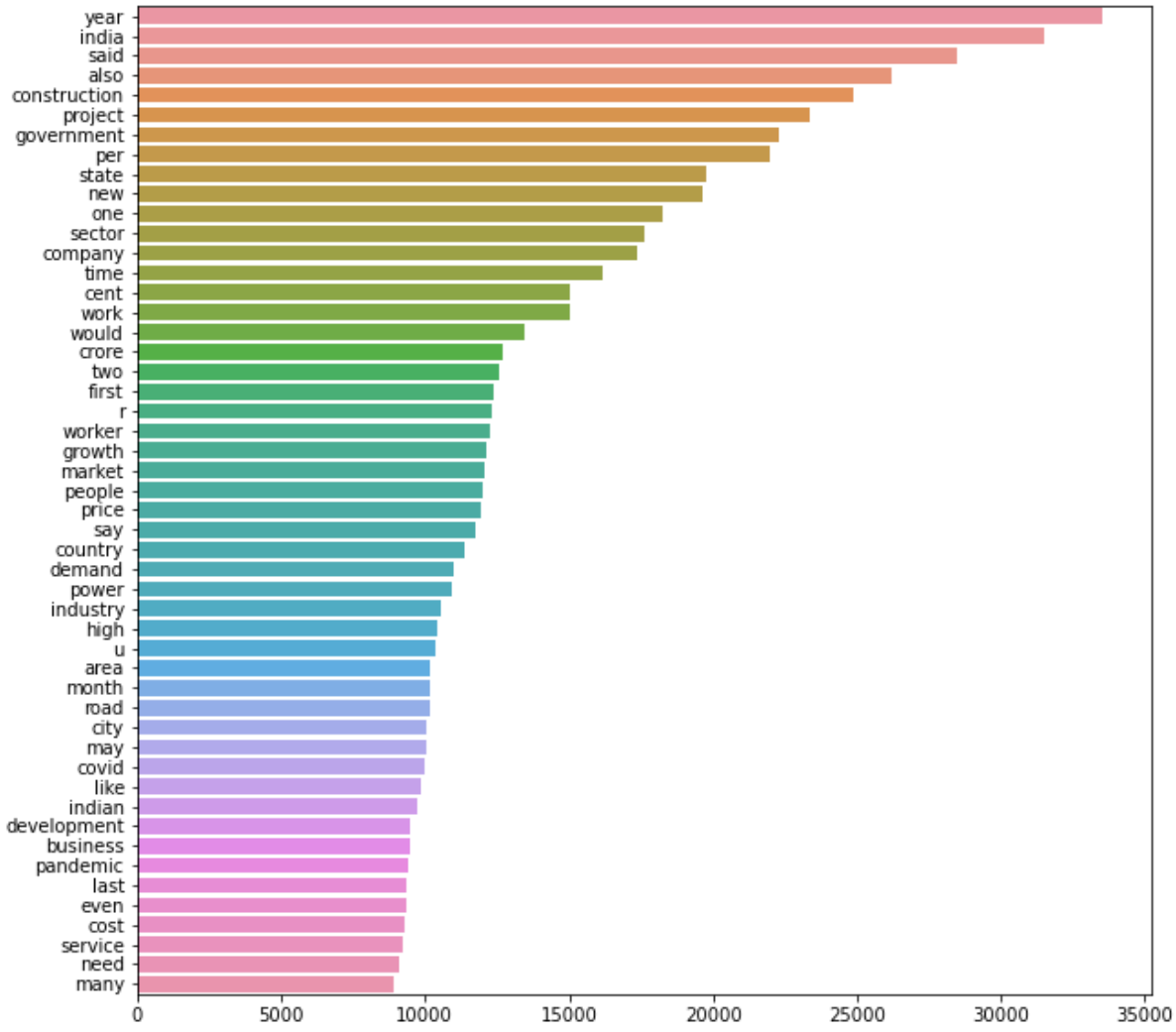


Figure 6: Most occurring words in the corpus after stop word removal and lemmatization

3.2.4 Duplicates in the data

News articles that are published usually appear in similar or rectified forms by several different publishers. After going through the data, it is observed that there are duplicates in the data. There are two types of duplicate data. One of them is the same article content published in different publisher editions on the same or different dates, as shown in Figure 7 below. The same type of article related to construction accident data is shown in Figure 8 below.

publish_date	publicati	Edition	Article
2021-07-02	Financial Express	Bangalore	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Hyderabad	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Mumbai	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Chennai	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Lucknow	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Pune	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Chandigarh	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Jaipur	L&T Construction bags \'significant\' orders L&T Construction bags 'significant' orders LARSEN AND
2021-07-02	Financial Express	Kolkata	L&T Construction bags \'significant\' orders L&T Construction bags 'Significant' orders LARSEN AND

Figure 7: Sample of duplicate data published in different editions of the same publisher

Publish d	Publication	Edition	Article
2021-12-10	The Indian Express	New Delhi	DECEMBER 10, 1981, FORTY YEARS AGODECEMBER 10, 1981, FORTY YEARS AGO FLYOVER COLLAPSE AHUGE CHUNK of the flyover under construction near Sewa Nagar for the Asian Games in Delhi crashed, injuring at least 17 labourers. Four have been admitted to the All India Institute of Medical
2021-12-10	The Indian Express	Kolkata	DECEMBER 10, 1981, FORTY YEARS AGODECEMBER 10, 1981, FORTY YEARS AGO FLYOVER COLLAPSE A HUGE CHUNK ofthe flyover under construction near Sewa Nagar for the Asian Games in Delhi crashed, injuring at least 17 labourers. Four have been admitted to the All India Institute of Medical
2021-12-10	The Indian Express	Mumbai	DECEMBER 10, 1981, FORTY YEARS AGODECEMBER 10, 1981, FORTY YEARS AGO FLYOVER COLLAPSE AHUGE CHUNK ofthe flyover under construction near Sewa Nagar for the Asian Games in Delhi crashed, injuring at least 17 labourers. Four have been admitted to the All India Institute of Medical

Figure 8: Sample of construction accident data published in different editions of the same publisher

The other type of duplicate data is articles published by different publications on the same or different dates. The sample of construction accidents with this type of duplicate is shown in Figure 9 below. The article discussing the accident of the Mumbai girder collapse is published by three different publications. Here, we can observe that the pattern of words and language in the text is changed.

publish_date	publication	Edition	Article
2021-09-18	The Hindu	Mumbai	14 hurt in Mumbai girder collapse 14 hurt in Mumbai girder collapse Maharashtra govt. orders inquiry into mishap; injured workers stable, say doctors STAFF REPORTER MUMBAI Fourteen workers sustained injuries in Mumbai after a girder of an under-construction flyover at MTNL junction in Bandra Kurla
2021-09-18	Hindustan Times	Kalyan	SCLR extension collapse: 14 hurt SCLR extension collapse: 14 hurt The condition of the injured in the underconstruction bridge collapse on SCLR is stable. MMRDA sets up a detailed inquiry into the incident, report expected within 15 days Mehul R Thakkar mehul thakkara hthve com MUMBAI: A total of 14
2021-09-18	The Indian Express	New Delhi	14 injured in Mumbai flyover accident 14 injured in Mumbai flyover accident Mumbai: Fourteen workers were injured after a girder span weighing over 250 metric tonne of an under-construction flyover tilted and collapsed while work was on at Bandra Kurla Complex (BKC) early Friday. The flyover is part of the

Figure 9: Sample of construction accident duplicate data published by different publishers

Duplicates are found very often in the data, and these have to be removed to reduce noise in the data. Noisy data is meaningless data that machines cannot understand and interpret correctly. It unnecessarily increases the amount of storage space and adversely affects the results of machine learning and deep learning models. These duplicates can be removed by vector space models, which will be discussed in CHAPTER 4.

3.2.5 Imbalance in the data

When exploring the data, it is observed that many articles have these keywords. The dataset is labeled to get an idea of the number of construction accidents in the whole dataset. The articles which are related to a construction accident or injury are labeled as '1', whereas the other articles are labeled as '0'. The labeling of 11208 articles consumes a lot of time as we have to manually read the article and assign a label, but it is necessary to know the distribution of news articles over the corpus to understand the data imbalance. From here on, we will discuss

construction accident data as positive data and the rest of the data as negative data. A sample of labeled data is shown in Figure 10: Sample of labeled data below, where the positively labeled data describes the lift collapse incident that happened at Worli. The negatively labeled data also has keywords such as ‘accident’, ‘dead’, and construction’, but this is a hit-and-run road accident case that happened near an under-construction work area. After labeling the articles, the positive data turned out to be 150 articles, and the negative data was 11058 articles. The positivity rate (positive articles / total articles) is just 1.338 %, i.e., there are only 1.338 positive articles for 100 negative articles.

publish_date	Article	Label
2021-07-26	Worli lift **collapse** incident: 2 held for death of 6 **workers** Worli lift collapse incident: 2 held for death of 6 workers The **workers** were working without any harness belt, helmet and other safety gear FAISAL TANDEL faisal.tandel@fpj.co.in The NM Joshi police have arrested two people, a contractor and supervisor, after six **workers** died at an under **construction** site on Saturday evening. The lift used for carrying **construction** material came down crashing and Killed the six **workers**. The police said the Six Workers were working without any safety belt, helmet and other precautionary measures. Paramijit Singh Dahiya, Deputy Commissioner Police, Zone 3 confirmed the arrest of two people and said a case has been registered at NM Joshi police station under section 304 (2) of the India n Penal Code (IPC). The case has been registered on the complaint of Ashok Jadhav 53, assistant sub-inspector, based with NM Joshi police station. The police said the arrested accused persons are "A total of six people died in the incident. Five died on the spot, while the sixth person succumbed to his **injuries** at KEM hospital, where he was shifted for treatment." Paramjit Singh Dahiya, Deputy Commissioner of Police, Zone 3 identified as Swapnil Ashok Mhamunkar the supervisor and Mukeshbhai Parsiya the contractor. According to NM Joshi police the incident took place on Saturday evening at 5:30 pm near Hanuman Galli, Worli. "It happened at the **construction** site of two towers, a twenty storey structure constructed by Lalithambika builders. The builder had hired a contractor for the job". "All the six were working near the car parking and were coming down using the lift to take up **construction** material. The life wire snapped and it crashed down from the height, killing five and **injuring** one who later succumbed to his injuries at KEM hospital," said a police officer from NM Joshi police station. The six deaths are identified as Chinmay Mandal 33, Bharat Mandal 30, Anil Kumar Yadav, Avinash Das 35, Abhay Yadav 32, and Laxman Mandal 35. "Total six people are **dead** in the incident. Five were dead on spot, while the sixth person succumbed to his injuries at KEM hospital, where he was shifted for treatment. The accused supervisor and contractor didn't use any safety and precautionary measures during the ongoing work. Investigation is going on," added Dahiya.	1
2021-09-13	Hit-and-run driver yet to be identified by cops ISCON flyover **accident** death Hit-and-run driver yet to be identified by cops Removal of traffic cams on SG Highway due to road work leaves cops with no clue, delays identification of driver who knocked down woman [Ahmedabad Mirror Bureau feedback@ahmedabadmirror.com TWEETS @ahmedabadmirror ore than 24 hours after a woman was hit by a speeding vehicle that led to her death on the ISCON flyover, cops are yet to make any headway in identifying the driver. Cops have no footage of the **accident**, which helps nab the culprits. In this case, thanks to major **construction** Work, traffic cameras have been removed from several places on the SG Highway. Meanwhile, the victim has been identified as a resident of Ghatlodia. SG-traffic PL AM Rathod said, "As multiple flyovers are under **construction** on the SG Highway, cameras were removed. In this case too, all the cameras in the vicinity were removed and hence we don't have any video footage of the accident." The 41-year-old woman was mentally-challenged, it has been learnt. She had left her home on Thursday and was missing since then. Her fam There are no CCTV cams on the stretch and no Vewitnesses to mishap ily members had approached the Ghatlodia police to look for her. Victim's **dead** body has been handed over to her brother. The case has raised questions about Ahmedabad being a smart city with CCTV cameras keeping an eye on every major road. On Saturday morning, the victim is believed to have been knocked down by a speeding vehicle on the stretch of ISCON flyover, adjacent to the Dev Arc Mall. The vehicle was coming from the Pakwan Crossroads towards the Karnavati Club. There are no eyewitnesses to the **accident**. A passer-by had alerted the police after seeing the woman''s body.	0

Figure 10: Sample of labeled data

3.2.6 Occurrence of each keyword in the dataset

The keywords are selected manually based on the occurrence of keywords in a manually collected dataset of 100 construction accident/injury news articles. The dataset is acquired from a specialized agency consisting of news articles with the keywords given in the time span of 7 months between July 2021 – January 2022. We wanted to check the occurrence of each keyword in the data acquired from the specialized agency. Table 2 shows the occurrence of

each keyword in the dataset. The left table describes the occurrence of the keyword in the positively labeled dataset and total dataset. As we have given construction as the mandatory keyword, it is occurring in all the 11208 articles. The next highest occurrence of keyword is 'fall' which is present in 4406 news articles, but it has a low positivity rate, i.e., less occurrence in the positive labeled data. But still, the 'fall' keyword holds significant importance in the positive data (Figure 11) as 29 out of 150 articles have this keyword occurred at least once in the article. The keywords 'collapse' and injur*' has relatively good positivity rates (Figure 12). The table on the right describes the occurrence of that specific keyword in the article and does not contain any other keyword. For example, the 'dead' keyword has occurred in 62 positive articles, but out of them, only five positive articles mentioned in the right table have the occurrence of only the 'dead' keyword and do not contain any other keyword. This shows that there is inter-dependency between the keywords, and many articles contain multiple keywords, as shown in Figure 10: Sample of labeled data.

Table 2: Data Exploration based on keyword occurrence in the labeled dataset

Articles containing keyword				**Articles containing only one keyword and not containing another keyword**			
Keyword	**Positive**	**Total**	**Positivity**	**Keyword**	**Positive**	**Total**	**Positivity**
construction	150	11208	1.34%	construction	0	0	0.00%
dead	62	1584	3.91%	dead	5	735	0.68%
accident	43	747	5.76%	accident	0	240	0.00%
injur*	111	606	18.32%	injur*	2	86	2.33%
fall	29	4406	0.66%	fall	2	2943	0.07%
collapse	110	747	14.73%	collapse	7	275	2.55%
struck	2	557	0.36%	struck	0	236	0.00%
worker	67	3057	2.19%	worker	4	1787	0.22%

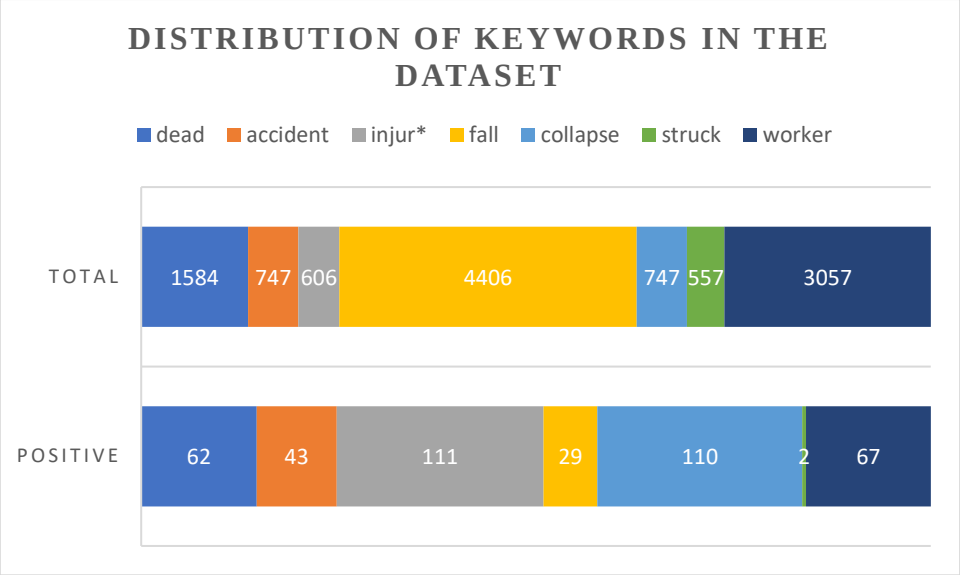


Figure 11: Graph showing distribution of keywords in the dataset

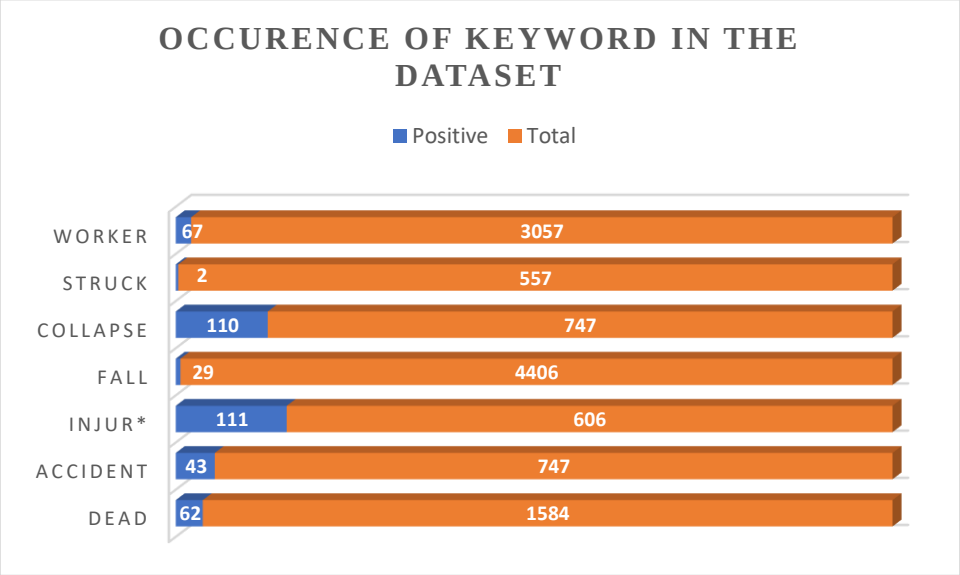


Figure 12: Graph showing occurrence of the keyword in the dataset

3.3 Metrics used for the evaluation of data

3.3.1 Confusion matrix:

A confusion matrix is a matrix used to describe the performance of a classifier. It can be determined if the true values of data are known. The confusion matrix presents a table layout of the different outcomes of the prediction and actual results of a classification problem and helps visualize its outcomes. The plot of the confusion matrix for binary classification is shown in Figure 13.

A true positive (TP) is an outcome where the model correctly predicts the positive class.

A false positive (FP) is where the model incorrectly predicts the positive class.

True negative (TN) is where the model correctly predicts the negative class.

False-negative (FN) is where the model incorrectly predicts the negative class.

	Prediced 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Figure 13: Confusion matrix for a binary classifier

3.3.2 Accuracy

Accuracy is the ratio of the number of predictions that are correct to the number of predictions that are made. Accuracy is a useful metric only when there is equal distribution of classes. If we are working on an imbalanced data where 99% are from one class and the rest 1% are of another class, at an instance where our model predicted all the data under only one class, and it completely failed in classification, it may give accuracy of 99% which sounds like a great result, but it has predicted all the minor class data wrongly. Since our problem deals with class imbalance, confusion matrices are more useful.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

3.3.3 Recall

The recall is the model's ability to predict positive values. It is the true positives divided by the total number of actual positives. A recall is a very important metric in this study as we want our model to predict positive values in binary classification.

$$Recall = \frac{TP}{TP + FN}$$

3.3.4 Precision

Precision is the model's ability to classify positive values correctly. It is the true positives divided by the total number of predicted positives.

$$Precision = \frac{TP}{TP + FP}$$

3.3.5 F1-score

F1 score is the harmonic mean of precision and recall, which ranges from 0 to 1. Precision measures how accurate the precisions are, and recall measures the proportion of true positives identified by the classifier. F1 score takes account of both the number of prediction errors that the model makes and the type of errors that are made. Ideally, we want a model that identifies all our positive cases and, at the same time, identifies only positive cases, which can be measured by using this F1 score. A model will obtain a high F1 score only if both precision and recall are high.

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

3.3.6 F-beta score

The f-beta score is the weighted harmonic mean of precision and recall, reaching its optimal value at 1 and worst value at 0. It is an extension of the F-1 score adding a configuration parameter beta. The beta parameter determines the weight of the recall in the combined score. Beta < 1 lends more weight to precision, while beta > 1 favors recall. Since we are more inclined towards the classification of recall, we use this metric in our study.

$$Fbeta - Score = \frac{(1 + beta^2) * Precision * Recall}{beta^2 * Precision + Recall}$$

A beta value of 2 is used in the entire study to favor the weight of recall in the combined score.

CHAPTER 4 DUPLICATE DATA REMOVAL USING VECTOR SPACE MODELS

4.1 Overview

The seven months of data consisting of 11208 news articles of data were initially split into training and test data based on the number of months. The data between July 2021 to September 2021 is considered the training data. The data between October 2021 to January 2022 is considered the texting data. Out of the total of 11208 news articles, training data consists of 8258 news articles in which 92 news articles are positively labeled data (construction accident-related articles), and the testing data consists of 2950 news articles in which 58 articles are positively labeled data. The main motive of our framework is to build a model using the training data and apply it to the test data on a regular basis for every 4 to 6 months. The correctly classified construction accident data obtained from the test data set is used for training the future test data sets.

4.2 Software used

To accomplish the tasks mentioned in the framework, coding is to be done as the plan is to automate the process. The current study extensively relies on the Python programming language for all the computations. The programming language is used in python for its easy interpretation, object-oriented, and high-level programming in data structures. Python is simple; the syntax is intuitive, easy to learn, and powerful, supported by various modules and packages. Python is an open-source language that is free, constantly updated, and used widely all across the globe. Python in our study has been used in different applications such as Visual Studio Code, Google Colaboratory, and Anaconda software package that contain both Python and Jupyter Notebook. Google Colaboratory (Google Colab) is a product from Google Research that allows anybody to write and execute the python code through the browser.

4.3 Vector Space models

The similarity between the texts slows down the process of discovering new information. As the name suggests, vector space contains a collection of vectors which are numerical representations of words, sentences, and even documents. A simple vector-like map coordinate only has two dimensions, but those used in natural language processing can have thousands and lakhs of coordinates. This vectorized representation of words helps us to easily determine the similarity between documents. There are three vector space models widely used for removing duplicates in the data. They are Cosine similarity, Jaccard similarity, and Euclidean

distance. These vector space models calculate and give a similarity score between two documents into a single numerical value by bringing out the degree of semantic similarity or distance from one another.

4.3.1 Feature Extraction

The basic concept of similarities is to identify feature vectors and thereafter measure the distance between the features. A feature vector is a vector containing multiple elements about an object. The low distance between those features implies that they are nearer and show a higher level of similarity value. On the other hand, high distance implies a low level of similarity value.

The intuition of a feature vector is the red-green-blue color descriptions. A color can be described by how much red, blue, and green it is. In Figure 14, shown below, the features are red, green, and blue in the three dimensions. A feature vector of this would be color = [R, G, B].

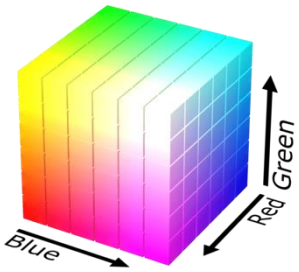


Figure 14: Features of color in a cube

(Source: https://commons.wikimedia.org/wiki/File:RGB_color_solid_cube.png)

Generating features from the text is done using the bag of words model and Tf-Idf vectors. These features practically represent the text in a numerical way to help with many kinds of analyses.

4.3.1.1 Bag of words model

A bag of words is a model which is often used in natural language processing for converting text data into vectors of numerical values, and it is also used in information retrieval. As the name suggests, the article is expressed as a bag of words, and it assigns the value based on the frequency of each word in the article. It does not consider word order, grammar, and syntax. Each word that appears in the text is independent and does not depend on other words. The intuition of the bag of words model is shown in Figure 15 below, where each word present in

the article is converted into a number based on its frequency. After applying this model, each article is represented as a row, and the generated vector will have columns/features equal to the total number of words in the bag.

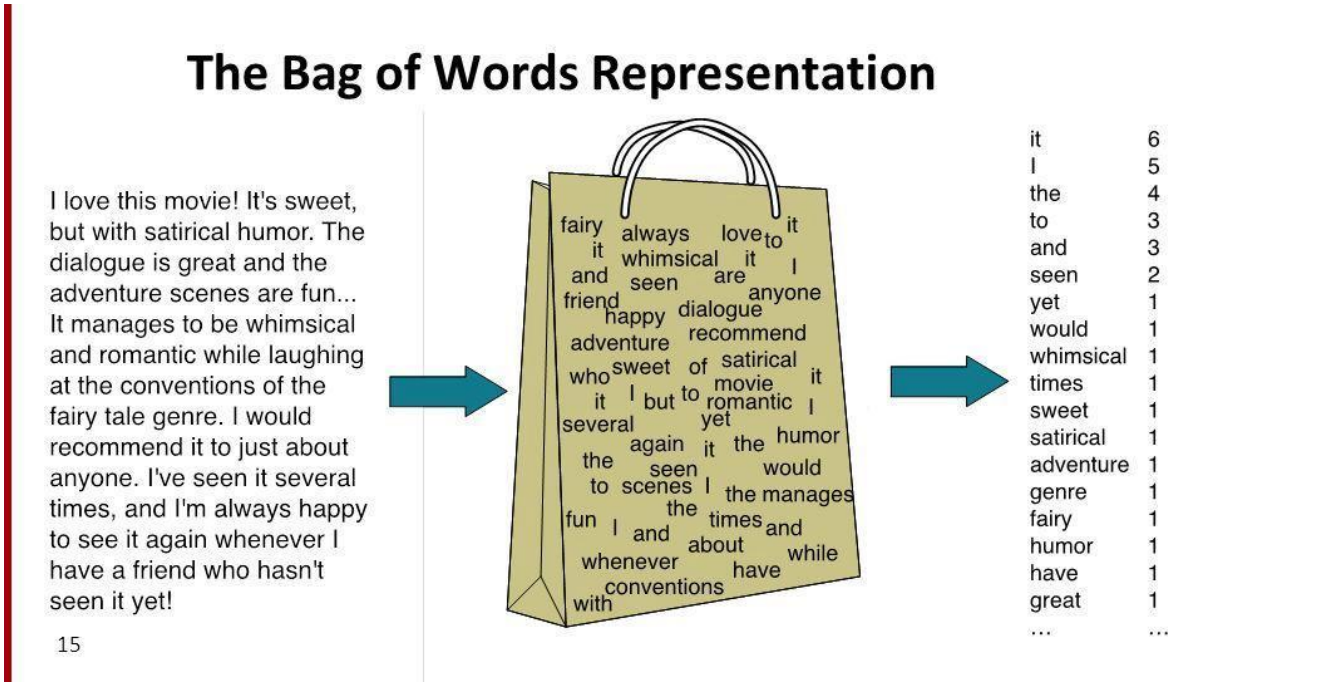


Figure 15: Bag of words representation of a single article

(Source: https://www.programmersought.com/images/947/0acb9279d17a1631bcfb154583cca443.JPEG)

4.3.1.2 TF-IDF vectorization

'TF' stands for term frequency, and 'IDF' stands for inverse document frequency. It is a simple algorithm that transforms the text into a meaningful representation of numbers. TF-IDF weight is a measure of the importance of a specific word in a text. In mathematical terms,

$$\text{Tfidf weight} = \sum_{i \in d} tf_{i,d} * \log\left(\frac{N}{df_i}\right) \quad (1)$$

Where in document d, $tf_{i,d}$ Is the number of occurrences of the i^{th} term, df_i is the number of articles that contain the i_{th} term, and N is the number of articles. In other words,

$$\text{Term frequency} = \frac{No.of\ repetition\ of\ specific\ word\ in\ the\ article}{Number\ of\ words\ in\ the\ article} \quad (2)$$

$$\text{Inverse document frequency} = \frac{No.of\ articles}{Number\ of\ articles\ containing\ that\ specific\ word} \quad (3)$$

The model can be constructed and used by a library in python called Scikit learn. By using TF-IDF vectorization, we can get features that can give us the importance of uncommon words in the document rather than just the frequency of the variables.

For our study, we considered using the TF-IDF vectorization for doing the similarity check due to its advantages over the bag of words model. Cosine Similarity and Jaccard Similarity are the two similarity measures that work well with TF-IDF vectorization.

4.3.2 Cosine similarity:

It is a measure of cosine angle in an n-dimensional space between two n-dimensional vectors. Mathematically, this is the dot product of two vectors, divided by the product of two vectors' lengths. The cosine similarity is measured by using the following formula:

$$\text{Cosine Similarity (A, B)} = \frac{A \,.\, B}{||A||\, X\, ||B||} = \frac{\sum_{i=1}^{n} A_i \, X \, B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (4)$$

Suppose there are two articles, A and B; as the distance between these articles, the similarity between the articles decreases and vice versa.

1 – Cosine Similarity = Cosine Distance

The angle between the two articles will show the result. If the angle between both articles is zero, then the Cosine Similarity is one which implies both the articles are the same. Any other angle will give a Cosine Similarity of less than 1. In this way, the cosine angle between the vectors decides the similarity between the articles.

4.3.3 Jaccard Similarity:

It is a measure of similarity among sets. The Jaccard Similarity of two sets is defined by the intersection size divided by the union size of two sets. Mathematically, Jaccard Similarity is determined using the following formula:

$$\text{Jaccard Similarity (A, B)} = \frac{A \cap B}{A \cup B} = \frac{A \cap B}{|A| + |B| - A \cap B} \quad (5)$$

where $\cap$ represents the intersection and $\cup$ represents the union.

Jaccard Similarity of an article with the same article is one. Jaccard Similarity of two articles is zero if $A \cap B = 0$. Here, A and B do not have to be the same size, and the similarity will always be a number between 0 to 1.

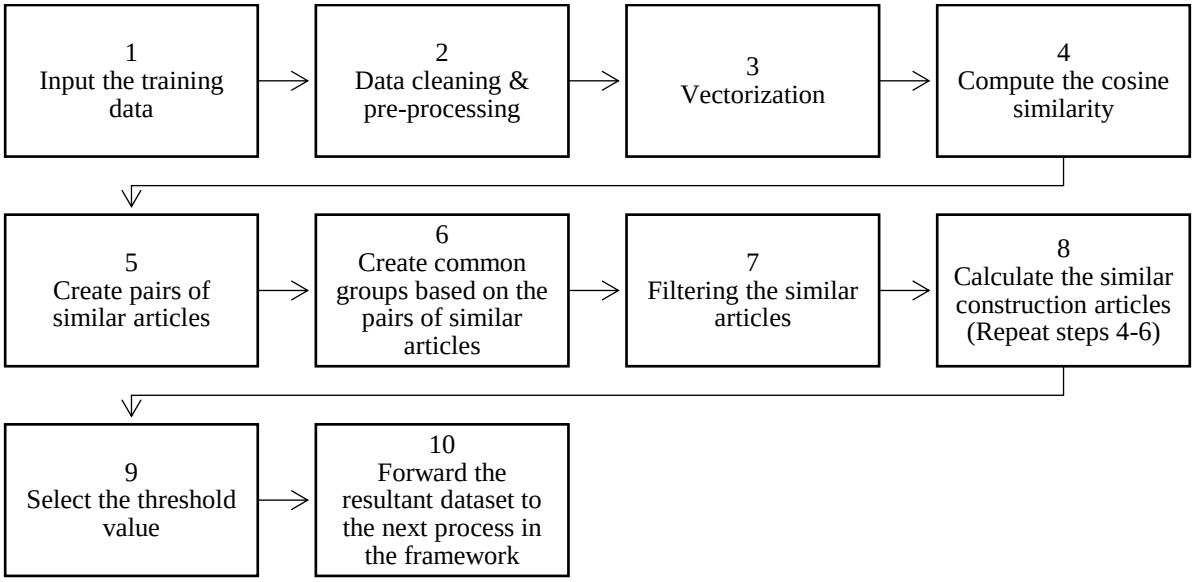
Both the similarity checks are done for a sample of 20 articles, and it is observed that Cosine similarity has much better accuracy in predicting the similarity of articles than the Jaccard similarity.

Hence, Cosine similarity with TF-IDF vectorization is used for our model.

4.4 Implementing Cosine similarity in the training data

The training data set of 8258 articles are used for implementing this step. The main motive of implementing cosine similarity is to identify and eliminate similar articles, thereby removing the redundant data and reducing the noise and size of the dataset. The machine learning models were initially tried on the entire training data set for implementing binary classification to classify the construction accident-related articles, but the results turned out to be very poor, with all the positively labeled data being classified as negatively labeled. This is because of the high imbalance in the data and a huge corpus of negatively labeled data which is dominating the positive data such that the machine is not able to classify the positive data from the negatively labeled data.

4.4.1 Process and working of cosine similarity



1. Input the training data: The training data of 8258 articles are inputted and loaded to perform the next steps.
2. Data cleaning & pre-processing: Data cleaning & pre-processing are two processes to clean the text data, process the data in a machine-readable format, and make it ready to feed data into the model. The data undergoes the process of removing punctuations, removing numbers, lower casing, tokenization, stemming, and lemmatization.

Removing punctuations and numbers: punctuations and extra spaces, hyphen marks, and colons are all removed in this step. In total, there are 32 main punctuations ('!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~') to be removed, which is done using the regular expressions to replace any punctuation in text with an empty string. The digits are also removed as they do not carry any valuable information.

Lower casing: If the text is in the same case, it is easy for a machine to interpret the words because the lower case and upper case are treated in a different way by the machine. For example, the machine treats 'Accident' and 'accident' differently. So, it is essential to convert the text into the same case, and lower case is preferred to avoid such problems.

Tokenization is the first step of data pre-processing where the text is split into chunks of information called tokens. These tokens are utilized to process every word and create a vector representation of a word suitable for machine learning.

Stop words removal: They are the most commonly occurring words in a text which do not provide any valuable information. 'where', 'they', 'this', 'the' are some of the examples of stop words. NLTK library is a common library used to remove stop words, and it has approximately 180 stop words that it removes.

Stemming and lemmatization: Stemming is a process of reducing the word to its root word. For example, play, plays, player, and playing are derived from the same word as play. Removing the prefix or suffix from the word like ing, es, s, etc. NLTK is a common library that is used to stem words. It is useful for standardizing vocabulary as well as dealing with sparsity issues. Various stemmers are available, such as Porter stemmer, Snowball stemmer, etc. Snowball stemmer is implemented in this step. But stemming is not very efficient as it derives a root word that does not have meaning and sometimes stems unwanted words (natural, nature converted into natur). So, to solve this problem, lemmatization is done. Lemmatization, similar to stemming, is a systematic way to reduce the words into their lemma by matching them with a language dictionary. From the NLTK library, WordNet Lemmatizer is used for lemmatizing.

3. Vectorization: The Tf-Idf vectorizer is used to convert the text into a number format. Tfidf Vectorizer is imported from the scikit learn library under the feature extraction module. Certain parameters are used in the Tfidf Vectorizer to get the best-vectorized format. This study's parameters used for tuning are 'ngram range', 'max_df', 'min_df'.

ngram_range: n-grams are continuous sequences of words or symbols, or tokens in a document. When n = 1, it is said to be a unigram where only one word is taken as a sequence at a time. When n=2, it is said to be a bigram where two words are taken as a sequence. For example, in an article with the sentence, 'I am a boy, ' generated unigrams are 'I', 'am', 'a', 'boy' and generated bigrams are 'I am', 'am a' 'a boy'. The N-gram range of (1,1) means only unigrams, n-gram range of (1,2) means both unigrams and bigrams. The lower and upper boundary of the range of n-values for different n-grams to be extracted. We do this in our step because different types of n-grams are suitable for different applications. In this study, we tried two iterations, one with only unigrams and the other as a combination of unigrams and bigrams.

Max_df: It is used for removing the terms that appear too frequently. When building the vocabulary, it ignores the terms with a corpus frequency strictly higher than the given threshold. If float in the range [0.0,1.0], the parameter represents a proportion of corpus, absolute integer counts. For example, max_df = 0.5 means 'ignore the terms that appear in more than 50% of the corpus'

Min_df: It is used for removing the terms that appear too infrequently. When building the vocabulary, it ignores the terms with a corpus frequency strictly lower than the given threshold. If float in the range [0.0,1.0], the parameter represents a proportion of corpus, absolute integer counts. For example, min_df = 0.05 means 'ignore the terms that appear in less than 5% of the corpus'

We tried many combinations by varying max_df with 0.6, 0.7, 0.8, 0.9 and varying min_df with 0.01, 0.02, 0.03, 0.04, 0.05.

4. Compute the cosine similarity: Cosine similarity is calculated by inputting the TF-IDF matrix. It has two parameters, X and Y, to input, and both of them will be the TF-IDF matrix obtained from the previous step. The output of the cosine similarity will be valued from 0 to 1 with a square matrix of size equal to the number of articles (8258 x 8258 in this case). In Figure 16, we can see the cosine similarity matrix of the first 10 articles in our data. The cosine similarity of an article with the same article is 1, which is the reason we see the diagonal elements in the matrix are 1. The cosine similarity of article 3 and article 4 is 1.00, which implies both the articles are similar. Since the lower triangular matrix and upper triangular convey the same information, we consider any one of the matrices to create pairs of similar articles.

	0	1	2	3	4	5	6	7	8	9
0	1.000000	0.228771	0.049048	0.130499	0.130499	0.079160	0.028248	0.068169	0.054437	0.054577
1	0.228771	1.000000	0.061554	0.242364	0.242364	0.124302	0.233101	0.126337	0.342776	0.115931
2	0.049048	0.061554	1.000000	0.068623	0.068623	0.095863	0.021416	0.132910	0.038998	0.144845
3	0.130499	0.242364	0.068623	1.000000	1.000000	0.144949	0.168298	0.131818	0.201956	0.139153
4	0.130499	0.242364	0.068623	1.000000	1.000000	0.144949	0.168298	0.131818	0.201956	0.139153
5	0.079160	0.124302	0.095863	0.144949	0.144949	1.000000	0.082954	0.466613	0.055245	0.256038
6	0.028248	0.233101	0.021416	0.168298	0.168298	0.082954	1.000000	0.057368	0.256057	0.080333
7	0.068169	0.126337	0.132910	0.131818	0.131818	0.466613	0.057368	1.000000	0.072115	0.326915
8	0.054437	0.342776	0.038998	0.201956	0.201956	0.055245	0.256057	0.072115	1.000000	0.065028
9	0.054577	0.115931	0.144845	0.139153	0.139153	0.256038	0.080333	0.326915	0.065028	1.000000

Figure 16: Screenshot of Cosine Similarity matrix of first 10 articles

5. Create pairs of similar articles: Each and every pair of articles have a certain similarity value between 0 to 1. But we want only those articles which have a high similarity index. To extract pairs of similar articles, we need to give a certain condition with a threshold value. For example, if we give a condition to extract pairs of articles that have a threshold value greater than 0.8, the resultant list will be the pairs of articles that have similarities greater than 80%. As we decrease the threshold value from 1 to 0, we get a greater number of similar pairs. In our study, we tried combinations of 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85 cosine similarity threshold value.
6. Create common groups based on the pairs of similar articles: The cosine similarity index gives us only the similarity between a pair of articles. In Figure 7, we have seen a number of articles (greater than 2) that are similar to each other. So, the similar articles should be grouped together to get a set of similar articles. For example, if articles 1 and 2 are a pair of similar articles and article 2 and article 6 are similar articles, then a common group is formed with a set of {article 1, article 2, article 6}. This algorithm is coded, and a sample of data is shown in Figure 17.

```
A sample of random 10 groups
 [[4, 2146, 3], [9, 41, 17], [18, 58], [24, 25], [38, 57, 37, 36, 35], [47, 56], [50, 51], [53, 54, 34, 30, 12], [62, 63], [64, 65]]
```

Figure 17: Screenshot of a sample of random 10 groups in the data

7. Filtering the similar articles: Now, we have duplicate articles separated into groups, and we want to store only one article from the group of articles as all the articles in the group will exhibit similar information about the same incident. There are various ways

to filter similar articles. One way to filter is to retain the article in the group which has maximum word length since it can contain more information about the data. The other ways are retaining the first article or retaining any random article in the group. In this study, we retained the first article in the group. For example, if there are a total of 1500 groups consisting of 4500 articles, we retain 1500 articles and remove the rest of 3000 articles.

8. Calculate the similar construction articles (Repeat steps 4-6): When the articles are getting grouped, and we are filtering the similar articles, among these articles, there may be articles that belong to the positively labeled data. The positively labeled data may be present in groups in three ways. One way is a group of only positively labeled data. The other way is to group one positive and one or more negatively labeled data. The third way is a group of more than one positive data and one or more negatively labeled data. The first and third way does not cost us anything since we are not using any precious information. But in a second way, we may have a chance of losing information. So, in this step, we calculate the first and third groups of data which contain groups of more than one positively labeled data, so that we could know the amount of positively labeled data that mingled with negatively labeled data and was thrown out. The intention of doing this step is to later maximize the data efficiency by minimizing the spill out of positive data with negative data.
9. Select the threshold value:

   In our study, a total of 483 iterations were done by changing the parameters of n-grams, min_df, max_df, and cosine similarity threshold value, as discussed above, to know the best parameters. A sample data of the first 10 iterations from the 483 iterations is shown in Figure 18. We can observe that at a low cosine similarity threshold value, data efficiency is very low, i.e., we could only retain 8 positively labeled data, and database size is reduced by 99.84%, which means it reduced from 11208 to 1793 articles. Though the database reduction is high, we lost all the efficiency of the data.

Iteration	n_gram	min_df	max_df	cosine similarity	construction accident	non-construction accident	similar cont acc	combined positive data	Data Efficiency	DB reduced by
0	Uni, Bi	0.01	0.6	0.2	0	13	8	8	8.70%	99.84%
1	Uni, Bi	0.01	0.6	0.25	1	91	11	12	13.04%	98.89%
2	Uni, Bi	0.01	0.6	0.3	4	373	12	16	17.39%	95.43%
3	Uni, Bi	0.01	0.6	0.35	14	828	14	28	30.43%	89.80%
4	Uni, Bi	0.01	0.6	0.4	19	1476	16	35	38.04%	81.90%
5	Uni, Bi	0.01	0.6	0.45	27	2133	15	42	45.65%	73.84%
6	Uni, Bi	0.01	0.6	0.5	37	2741	14	51	55.43%	66.36%
7	Uni, Bi	0.01	0.6	0.55	45	3208	14	59	64.13%	60.61%
8	Uni, Bi	0.01	0.6	0.6	52	3588	14	66	71.74%	55.92%
9	Uni, Bi	0.01	0.6	0.65	58	3846	14	72	78.26%	52.72%
10	Uni, Bi	0.01	0.6	0.7	64	4091	13	77	83.70%	49.69%

Figure 18: Results showing data efficiency and reduced database size for each iteration

The method chosen in this study is to select the parameters which could have at least 90% of the data efficiency (i.e., more than 90% retention of construction accident articles), and database size is reduced to a maximum. Out of 483 iterations, we got 120 iterations that have data efficiency greater than or equal to 90% and sorting the reduction of database size in descending order, and we get the best parameters of our model.

Figure 19 shows the top 10 iterations of our model. The best iteration has a data efficiency of 92.39%, and the database size is reduced by 49.69%. The best iteration has a cosine similarity of 0.65, an n-gram range of (1,1), and does not have any min_df and max_df parameters, which implies it considered all the features (columns) of the Tf-Idf vector. The columns' similar construction data gives the count of similar construction articles obtained through step 8. The column ‘combined positive data’ in Figure 19 gives the number of positively labeled data, which is the sum of construction accident data and similar construction accidents. Finally, the conclusion is we could retain 70 out of 92 construction accident data, and 15 are similar articles that got removed. So, 85 articles are known to us, and there are only 7 articles that might have been grouped with negatively labeled data and removed as duplicates. It means that by just sacrificing 7 articles, we were able to reduce the training dataset by almost half, from 8258 to 4156 articles.

Iteration	n_gram	min_df	max_df	cosine similarity	construction accident	non-construction accident	similar cont acc	combined positive data	Data Efficiency	DB reduced by
11	Unigram			0.65	70	4085	15	85	92.39%	49.69%
284	Uni, Bi			0.4	69	4099	16	85	92.39%	49.53%
263	Uni, Bi	0.05	0.8	0.75	71	4152	13	84	91.30%	48.86%
277	Uni, Bi	0.05	0.9	0.75	71	4152	13	84	91.30%	48.86%
235	Uni, Bi	0.05	0.6	0.75	71	4153	13	84	91.30%	48.85%
249	Uni, Bi	0.05	0.7	0.75	71	4153	13	84	91.30%	48.85%
39	Unigram	0.05	0.8	0.75	71	4153	14	85	92.39%	48.85%
40	Unigram	0.05	0.9	0.75	71	4153	14	85	92.39%	48.85%
41	Unigram	0.05	0.6	0.75	71	4154	14	85	92.39%	48.84%
53	Unigram	0.05	0.7	0.75	71	4156	14	85	92.39%	48.81%

Figure 19: Top 10 iterations based on data efficiency and database size

10. Forward the resultant dataset of the selected threshold value to the next process in the framework: The resultant dataset which came after choosing the best parameters is fed into the next step of the framework.

    Finally, the input data before cosine similarity is 8258 articles, of which 92 are positively labeled data and 8166 are negatively labeled data.

    The output data is 4156 articles, of which 70 articles are positively labeled data, and 4086 are negatively labeled data.

CHAPTER 5 APPLICATION OF UNSUPERVISED AND SUPERVISED LEARNING APPROACHES FOR CLASSIFICATION

5.1 Challenges faced in the binary classification

The framework shown in Figure 1 imposed many challenges due to the high imbalance in the dataset, i.e., construction accident-related articles are very less (1.34%) when compared to the whole dataset, which is bought from the specialized agency using construction-related keywords extraction. It is understood that the imbalance will be present even in future datasets, and our framework should handle the imbalance. On completing this, there are certain modifications in the framework in the process between duplicate data removal and ML-based binary classification.

5.1.1 **Unsupervised Learning:**

The dataset which we extract provides only large quantities of textual information and has no pre-labeled data. Our aim is to have minimal human intervention in the framework, which can be possible only with unsupervised learning. It is a technique where the machine automatically finds the relation between the input variables and gives the outcome without any use of labeling of data. Clustering is a technique that is used to automatically group data into different clusters for a given dataset such that the articles belonging to a cluster will have substantial similarities (Baek et al., 2021). Unsupervised learning techniques are fully automated and reduce the necessity of manual inputs to a great extent. On the other hand, unsupervised learning techniques also have certain limitations compared to other techniques, which is the reason they will have lower classification performance compared to semi-supervised and supervised. The optimal number of clusters is automatically determined by the clustering techniques, which may not match the users' desired number of clusters.

5.1.1.1 K-means clustering

In this study, we have used K-Means clustering and Hierarchical clustering techniques on the output dataset mentioned in 8.2.1 after duplicate data removal combined with the testing data. The optimal number of clusters using the K-means algorithm can be found by the Elbow method. The basic idea behind the K-means clustering is the total intra-cluster variation which is called the within-cluster sum of squares (WCSS), is minimized. The total WCSS measures the compactness of the clustering where the minimum is desirable. The optimal number of clusters is one where adding another cluster doesn’t improve much better than the total WCSS.

The location of a bend (knee) in the plot is generally considered an indicator of the optimal number of clusters. After the application of K-means clustering to the dataset, from Figure 20, we can see that there is no bend in the plot, indicating that the optimal number of clusters is greater than 10, but desirable clusters are only 2 since we are doing binary classification. The words in each cluster divided by the K-means can be seen in Figure 21, indicating that the K-means technique miserably failed in clustering as the keywords are not aligned with the topic. Moreover, the F-1 score of the K-means algorithm is 0.05, which is very low, and the results of the confusion matrix are shown in Figure 22, where it predicted most of the articles as 1, resulting in a low precision value.

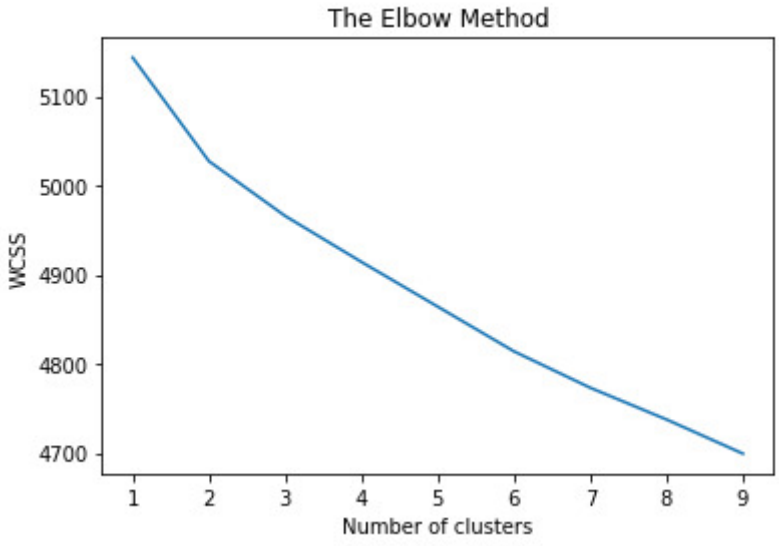


Figure 20: Results showing the optimal number of clusters in K-means clustering applied to the dataset

0 : price, per cent, cent, per, growth, sector, india, year, company, china, demand, market, steel, quarter, fy, bank, economy, industry, crore
1 : said, project, road, work, state, government, worker, also, police, year, minister, construction, water, delhi, area, one, people, building, district

Figure 21: Top words in both the clusters classified by K-means clustering

	Prediced 0	Predicted 1
Actual 0	1229	4282
Actual 1	0	128

Figure 22: Confusion matrix of K-means clustering

5.1.1.2 Agglomerative Hierarchical clustering

The agglomerative hierarchical method of clustering initiates by considering each point as a separate cluster and starts joining points to cluster in a hierarchical fashion based on distances. The optimal number of clusters is given by a dendrogram which shows the sequences of merges or splits of the cluster. If two clusters merge, the dendrogram will join them in a graph, and the height of the join will be the distance between those clusters. The more the distance between those clusters, the more the vertical line. The tallest vertical line, when cut horizontally, gives the number of clusters. Figure 23 shows the dendrogram for the applied dataset for the agglomerative hierarchical clustering technique. The F-1 score of the hierarchical clustering algorithm is 0, indicating very poor results, and the confusion matrix is shown to predict all the positive values as negative, resulting in zero recall value.

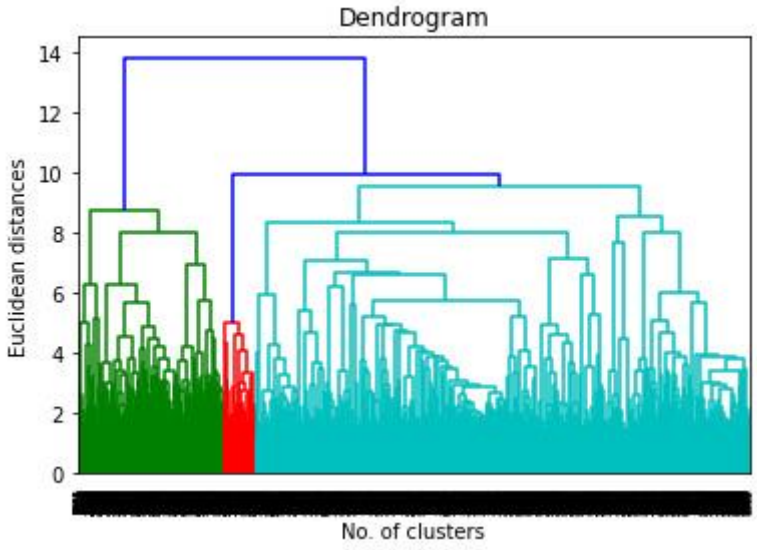


Figure 23: Dendrogram showing the agglomerative hierarchical clustering technique applied to the dataset

	Prediced 0	Predicted 1
Actual 0	4288	1223
Actual 1	128	0

Figure 24: Confusion matrix of hierarchical agglomerative clustering

The unsupervised learning techniques could not give the desirable classification results for the binary classification due to the above-mentioned limitations. This indicates that the machine is not able to understand the domain knowledge possessed by the human.

5.1.2 **Semi-Supervised Learning:**

It is a branch of machine learning which sits between supervised and unsupervised learning, where we use both labeled and unlabelled data to perform certain learning tasks. This is mainly suitable for datasets where a large amount of unlabelled data is available, which is tackled by smaller sets of labeled data. Semi-supervised classification methods are particularly relevant to scenarios where labeled data is scarce, expensive, or difficult to obtain. Semi-supervised classification methods try to utilize unlabelled data to construct a learning system whose performance exceeds the performance of the learning system obtained when using only the labeled data. If x is the input and y is the label, a necessary condition of semi-supervised learning is that the data distribution p(x) over the input space contains information about posterior distribution p(y/x). If this is the case, unlabelled data can be used to gain information about p(x) and thereby about p(y/x). If p(x) contains no information about p(y/x), it is nearly impossible to improve the accuracy of predictions based on additional unlabelled data (van Engelen & Hoos, 2020). It says that unlabelled data is only useful if it carries information useful for label prediction that is not contained in label data alone. The most widely recognized assumptions are the smoothness assumption, where if two samples are close in the input space, then their labels should be the same. The low-density assumption, where the decision boundary should not pass-through high-density areas in the input space, and manifold assumption, where data points lying on the same low-dimensional manifold should have the same label and cluster assumption, where data belonging to the same cluster belong to the same class. Most semi-supervised learning algorithms depend on one or more of these assumptions, either explicitly or implicitly (van Engelen & Hoos, 2020). In our study, we used label propagation, label spreading, and self-training algorithms to perform the binary classification. All mentioned algorithms can be extracted from the scikit learn python machine learning library.

5.1.2.1 Label propagation

Label propagation is an iterative algorithm that computes soft labels by propagating the estimated label at each node to its neighborhood nodes based on the edge weights. The new estimated label at each node is calculated as the weighted sum of the labels of its neighbors. In this way, we propagate labels from each node to the neighboring nodes and reset the predictions of the labeled data points to the corresponding true labels for more mathematical explanations

check (Zhu & Ghahramani, 2003). In our study, we performed label propagation by training the data on the training dataset and testing on the test dataset mentioned in 8.3. The data is trained with both Bag-of-words (Bow) and TF-IDF vectorizers. With the Bow vectorization, we could classify only one positive data correctly out of 58 construction accidents present in the testing dataset. The resulting confusion matrix is shown in Figure 25, and the final F1 score is 0.03125. The same training is done with the TF-IDF vectorizer, and the resultant F1 score is 0, and the machine predicted all the labels as 0.

	Prediced 0	Predicted 1
Actual 0	2886	5
Actual 1	57	1

Figure 25: Confusion matrix for label propagation with Bow vectorizer

5.1.2.2 Label spreading

Label spreading is an algorithm similar to label propagation with some differences. IT uses symmetric normalized graph Laplacian matrix in its calculations, while label propagation uses a random walk normalized Laplacian. Label propagation uses hard clamping, which means that labels of originally labeled points never change, but label spreading adopts a soft clamping which is controlled through an additional hyperparameter α, which describes the relative amount of information the point obtains from its neighbors versus its initial label information. The resulting confusion matrix is shown in Figure 26, and Label spreading trained with the Bow model gave a slightly improved F1 score of 0.1126 compared to the Label propagation model, but still, the results are far below par. The TF-IDF vectorizer trained with the label spreading model did not show any improvement and gave an F1 score of 0, with all the 1s predicted as 0s.

	Prediced 0	Predicted 1
Actual 0	2882	9
Actual 1	54	4

Figure 26: Confusion matrix for label spreading with Bow vectorizer

5.1.2.3 Self-training

Self-training is an algorithm that uses a supervised classifier that is iteratively retrained on its own most confident predictions. It is iteratively trained on both labeled data and data that has been pseudo-labeled in previous iterations of the algorithm. The pseudo-labeled data are the data that have more confidence (probability). The process is done until no more unlabelled data remain.

The data trained with Bow vectorizer and Random Forest (RF) classifier resulted in an F1 score of 0, Support Vector Classifier (SVC) gave an F1 score of 0.086, and SVC with 'rbf' kernel gave the same F1 score of 0.086. All the three classifiers, when working with the TF-IDF matrix, gave an F1 score of 0.

From these applications, it is understood that the machine is not able to identify the correct pattern among the data and is unable to classify the data. The reason for this could be the high imbalance in the data. The vocabulary used in the minority dataset (positively labeled data) is already used in the majority dataset, making it difficult for the differentiation between both datasets. To overcome these issues, certain modifications are done in the framework, which is discussed in detail in the next section.

5.2 Modification in the framework based on the challenges

From the previous section, the difficulties in the framework are understood, and the changes made in the framework are discussed below. The modified framework is shown in Figure 27.

5.2.1 Refining the data:

The cosine similarity is removing similarly related articles and decreasing the noise of the data. But even after this reduction, we can see that the algorithms are not giving good results. As a result, it is decided to further refine the data by using a semi-supervised state-of-the-art learning method, Guided Topic modeling with Latent Dirichlet Allocation (GLDA or GuidedLDA). LDA is an unsupervised learning method, which is made semi-supervised by giving certain keywords in each class by which the learning system will be guided. The meaning of refining is the make a substance-free from other substances or improve something by changing little details (Oxford Dictionary). In this process, we filtered the data and wanted to retain only the articles which contain the keywords related to construction accident-related data and separate the articles which contain irrelevant data. By this method, we can reduce the total number of articles and prepare the dataset for better classification. As the irrelevant data is removed, now the binary classification will be improved, and the machine will now effectively learn the

decision boundary and try to classify the articles which are among the articles which contain keywords of construction accident-related data. The details of this process and the procedure is discussed in CHAPTER 6.

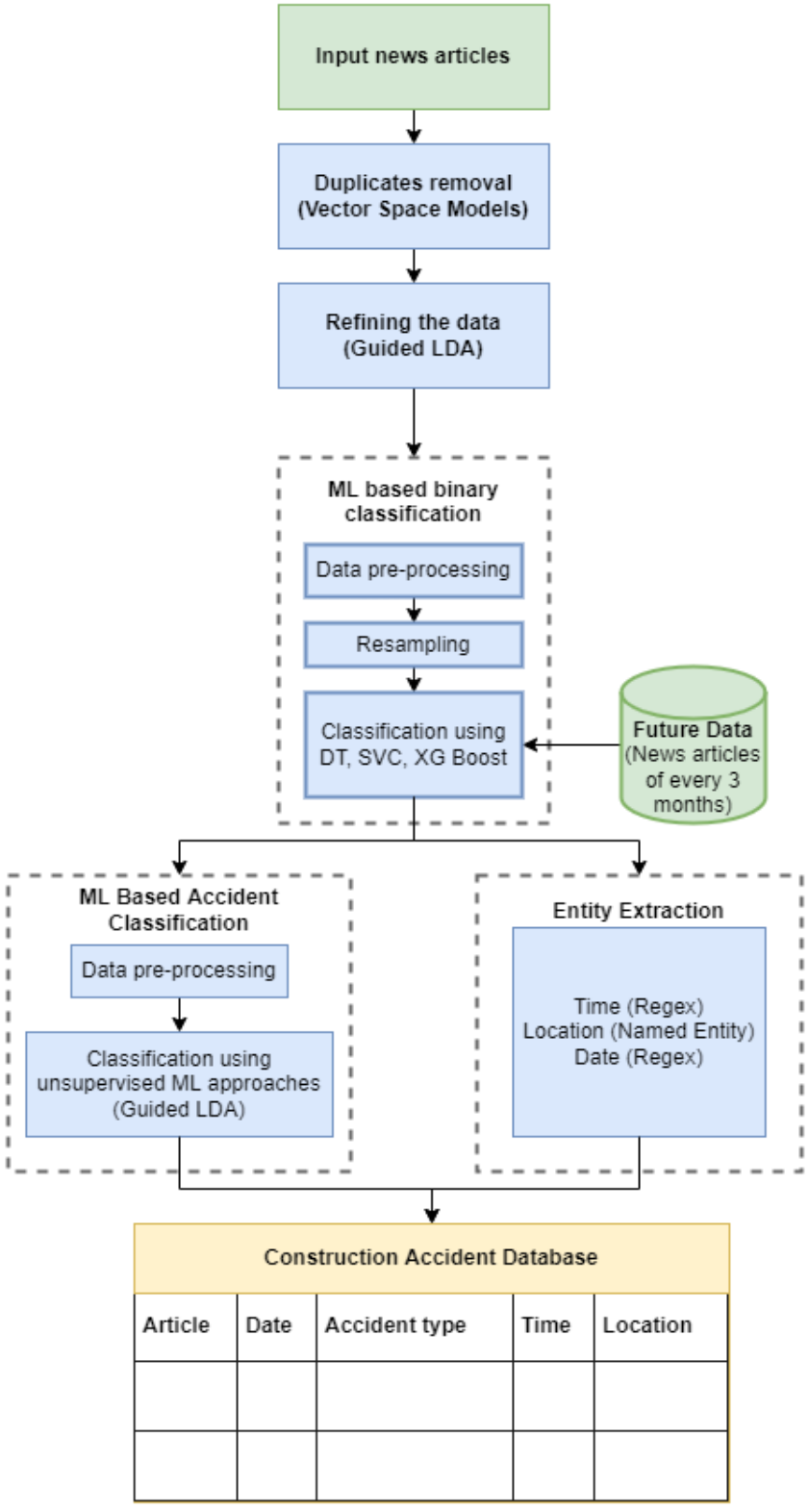


Figure 27: A modified framework for creating a construction safety database used in our analysis

5.2.2 Resampling the data

The challenge of working with imbalanced datasets is that most machine learning techniques will ignore the effect of imbalance and, as a result, end up with poor performance, especially in the minority class, which is the class that is most important to classify correctly. There are mainly two ways of addressing the imbalance. One way is through oversampling the minority class, and the other way is down-sampling the majority class. The simplest approach involves duplicating articles in the minority which do not add any new information to the model. There is another type of data augmentation technique called Synthetic Minority Oversampling Technique (SMOTE), where new articles can be synthesized from the existing articles(Chawla et al., 2002). The working of SMOTE is that it selects a random minority class instance (a) and finds its k nearest minority class neighbors. The synthesized article or sample is created by choosing one of the random k nearest neighbors (b) and connecting a and b to form a line segment in the feature space and created at a randomly selected point between these two articles in the feature space. The founder of SMOTE also carried out experiments on several imbalanced datasets and found that a combination of SMOTE and under-sampling works better than just applying SMOTE. This is because the initial bias of the learner towards the majority class is reversed in favor of the minority class (Chawla et al., 2002). An overview of various resampling techniques and experimentation processes is discussed in 7.1.

5.2.3 Binary classification based on supervised learning approaches:

From the previous attempts, unsupervised and semi-supervised learning methods turned out to be ineffective against this kind of dataset. Moreover, resampling has been incorporated in the data, which requires thorough labeling of the training data. Hence, supervised learning combined with resampling techniques could give a breakthrough for this problem. However, the author intends to prepare the model only using the training data, and the model should be robust enough to perform the binary classification on future imbalanced data. The plan is to build the model on the training data only and update the classified future data into the training data after each iteration. To give a clear explanation, in the first iteration, the training data is data between July-September 2021, and the testing data is the data between October 2021 – January 2022. For the second iteration, the training data will be July 2021 – January 2022, and the testing data will be the next 3- or 4-months data. At the end of each iteration, we update the training dataset and make the model robust enough to classify the future articles. More details of this classification are explained in 8.1.

CHAPTER 6 REFINING THE DATASET BY USING GUIDED-LDA

6.1 Overview and process of Guided LDA

Guided-LDA is a semi-supervised topic modeling method in which the topics are guided by the seed words or keywords input manually. Topics here are referred to as a collection of semantically linked terms that are commonly used to model documents in text mining and classification applications. To know Guided LDA, it is important first to understand LDA. LDA is an unsupervised topic modeling that is based on the assumption that each document is made of topics and each topic is in turn made up of different words. It groups the words together into topics based on the probability of the word belonging to a topic. Once the words are grouped into the topic, we can see which group the news article talks about by which we can classify that news article into that topic. LDA mathematically considers two distributions, topic-word distribution (∅) and a document-topic distribution (θ). Both ∅ and θ are taken as Dirichlet distributions. The topics have to be discovered rather than pre-determined. Starting with a random sample from a Dirichlet distribution, LDA determines the topics using an iterative approach. Using an iterative procedure, the parameters of the distribution that optimize the likelihood of a document *d* belonging to a subject *z* and the likelihood of a word *w* belonging to topic *z*. Each topic is defined by its topic-word distribution at convergence, which is the likelihood of a word belonging to the topic being completed. The converged topic-word distribution and frequency of words in the article are then used to compute the probability of each subject in the article. The methodology used for LDA is Gibbs sampling. In this method, we initially assign a random topic to the article and then iterate through each word by assuming it does not have any topic, and based on the probability of other words in the same article and the probability of other articles; we assign the topic which has maximum probability. The article is then classified into the topic having the highest topic probability in the document.

Using LDA, more often than not, the topics(classes) we get from an LDA model are not to our satisfaction as topics may contain information that is overlapping. Guided LDA addresses this problem by giving additional information to the model, which gives the topics a nudge in the direction we want them to converge. The additional information is the set of keywords or seed words that were given to improve both the topic-word and document-topic probability distributions. This enhances the model in generating words related to the topic more accurately and choosing document-level topics. For more mathematical understanding and how to

incorporate it, please see this (Jagarlamudi & Daum, 2012). The document[1] and open-source code[2] are freely available online.

6.2 Implementation of Guided LDA

In our study, Guided LDA is used to retain the articles which are having construction accident or incident-related keywords in them. In this way, we can filter the relevant data which has keywords related to the construction accident and remove the documents which do not have these keywords in a good proportion. Before sending the articles into the model directly, data is cleaned and pre-processed. The data cleaning steps involved are:

Lowercasing: It maintains uniformity by avoiding case variants of the same words. It reduces sparsity and vocabulary size.

Punctuation removal: The punctuations express sentiments in an observation, easily interpretable by humans, but it's not useful information to the machine.

Spelling correction: Incorrect spellings and grammatical errors are commonly present in texts, and these are present even in the dataset considered in this study. Autocorrect is a python library used for this purpose.

Number removal: All the unwanted numbers which are difficult for the machine to interpret are not useful information to the machine. Hence, they are removed from the dataset.

The data pre-processing steps used in this study are Stop words removal, Stemming, and Lemmatization which are already detailed in section 5.1. We have given a seed-words set containing 84 keywords that are related to the construction accident keywords to nudge towards the positive dataset and 128 keywords to nudge towards the negative dataset. The seed words are given in such a way that any construction accident-related article is guided through these seed words. The seed-words list is shown in Figure 28 and Figure 29.

[1] http://guidedlda.readthedocs.org/
[2] https://github.com/vi3k6i5/GuidedLDA

building	fire	trapped	fell	rescued	doctor	renovation	shop
police	contract	inquiry	declare	electrocuted	sustained	drowned	house
incident	contractor	mishap	register	led	underconstruction	collapse	span
work	spot	debris	person	condition	beam	road	old
hospital	girder	killed	negligence	structure	proportion	people	ground
flyover	lift	rushed	operation	storey	crash	year	injury
site	death	report	deceased	illegal	crashed	floor	lift
rescue	labourer	accident	worker	toppled	disaster	man	authority
bridge	safety	died	working	crane	supervisor	dead	identified
area	hurt	official	development	arrest	repair	around	body
				caved	situation	machine	place

Figure 28: Positive seed words input to the Guided-LDA

government	lakh	social	market	price	job	industrial	decade
state	department	president	cent	investment	private	estate	study
india	due	infrastructure	technology	data	order	social	network
minister	part	international	high	global	vehicle	opportunity	source
crore	km	station	need	bank	rate	corporation	benefit
land	power	system	many	million	fund	application	available
covid	home	union	business	scheme	future	quarter	research
national	centre	meeting	indian	policy	expected	economy	solar
world	health	forest	service	support	com	economic	education
court	cost	company	growth	solution	property	firm	committee
temple	airport	sector	demand	issue	housing	model	population
public	party	first	last	capacity	office	asset	developed
pandemic	security	second	month	customer	capital	limited	employment
country	urban	industry	product	facility	green	lockdown	action
production	complete	development	part	various	impact	digital	community
management	help	water	process	financial	environment	waste	challenge

Figure 29: Negative seed words input to the Guided LDA

The seeded keywords and input dataset are used by the Guided LDA to generate document-topic distribution, and the articles are labeled with the category showing the maximum probability. The classification performance is evaluated using the Recall score metric and F1 score metric. The recall score is a priority considering the purpose of doing this Guided-LDA as we wanted to retain the construction accident articles by removing the irrelevant data. The output of this dataset will be a set of articles with data labeled by Guided-LDA, and all the positively predicted labels are fed as input to the next process in the framework, which is binary classification.

For implementing the Guided-LDA, there are six parameters which are mentioned below:

1) n_topics(t: int): This refers to the number of topics the observations are to be classified into, which is pre-specified. For our dataset, the value is set to 2.

2) n_iter (ni: int): This is the number of sampling iterations to perform
3) alpha (a: float): The Dirichlet parameter of LDA related to document-topic distribution
4) eta (b: float): The Dirichlet parameter of LDA related to topic-word distribution
5) seed_topics (l: list): A list of lists, where each list consists of a set of seed keywords for a classification category.
6) seed_confidence (s: float): This expresses the confidence in the mapping between the seed keywords and topics. The default value is 0.

Parametric analysis is done to check whether the model is sensitive to any parameters (see 8.2.2.1). The iterations done on various parameters are:

Alpha = [0.01, 0.02, 0.05, 0.1]

Eta = [0.01, 0.05, 0.1, 0.15, 0.2, 0.25]

N_iter = [2000, 4000]

Seed_confidence = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.975]

CHAPTER 7 MACHINE LEARNING BASED BINARY CLASSIFICATION

With the unsupervised and semi-supervised learning approaches not being effective against the binary classification (section 4.1), supervised learning methods are adopted for building the classification model. On the other hand, in spite of the reduction in the size of the training dataset, the imbalance in the data is still very high, 98%-2%. Many works of literature followed resampling techniques to handle the class imbalance. (Rivera et al., 2020) has done news classification for traffic incident points where he dealt with classification using five resampling techniques. (Kaur et al., 2019) presented a systematic review of imbalanced data challenges, their applications, and solutions. The problem of imbalanced data distribution is a problem even in realistic applications such as fault detection, fraud detection, cancer detection in medical diagnoses, etc.

7.1 Resampling

Resampling techniques can be applied either by under-sampling or over sampling the dataset. Under-sampling is the process of decreasing the amount of majority class samples and over sampling is the process of increasing the amount of minority class samples by producing new samples or repeating some samples. Figure 30 shows the difference between oversampling and under-sampling.

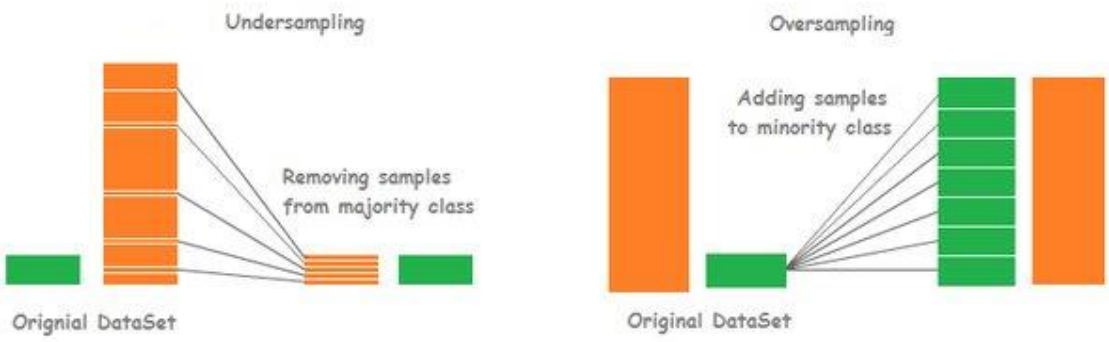


Figure 30: Difference between over sampling and under-sampling

(Source: (Mohammed et al., 2020)).

There are three broad approaches to address the imbalance issues in the data classification: pre-processing approach, algorithmic centered approaches, and hybrid approaches. In a pre-processing approach, methods involved are random over-/under-sampling to obtain an approximately equal count of samples in the classes. The algorithmic-centered approach

includes assumptions created to favor the minority class and changing the costs to get the balance classes. Hybrid sampling methods are those that apply both re-sampling techniques to attain balance in the data. Hybrid approaches minimize the chances of information loss and result in faster prediction. The synthetic Minority Over-Sampling technique (SMOTE) is a popular sampling method where it generates synthetic examples in minority classes, and it is an alternative to the duplication of minority class samples. (Chawla et al., 2002) proposed the concept of SMOTE and its combination with other learning algorithms. He suggested using a combination of sampling methods to improve classifier performance. He suggested combining SOME with random under-sampling to overcome the issue of imbalanced data and does not rely on under-sampling only. SMOTE increases minority class samples by generating synthetic samples rather than over-sampling with replacement by operating in feature space rather than data space. The illustration of SMOTE is shown in Figure 31. The minority class is oversampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. In the combined use of under-sampling and SMOTE, the minority class is first over-sampled using the SMOTE technique to a specified percentage of the majority class followed by the majority class, under-sampled by randomly removing samples from the majority class until the minority class becomes a specified percentage of the majority class followed by oversampling using SMOTE. It is very effective to apply SMOTE before under-sampling because the generated synthetic samples are based on the overall dataset and not the under-sampled dataset. Later, we do under-sampling to balance the class imbalance.

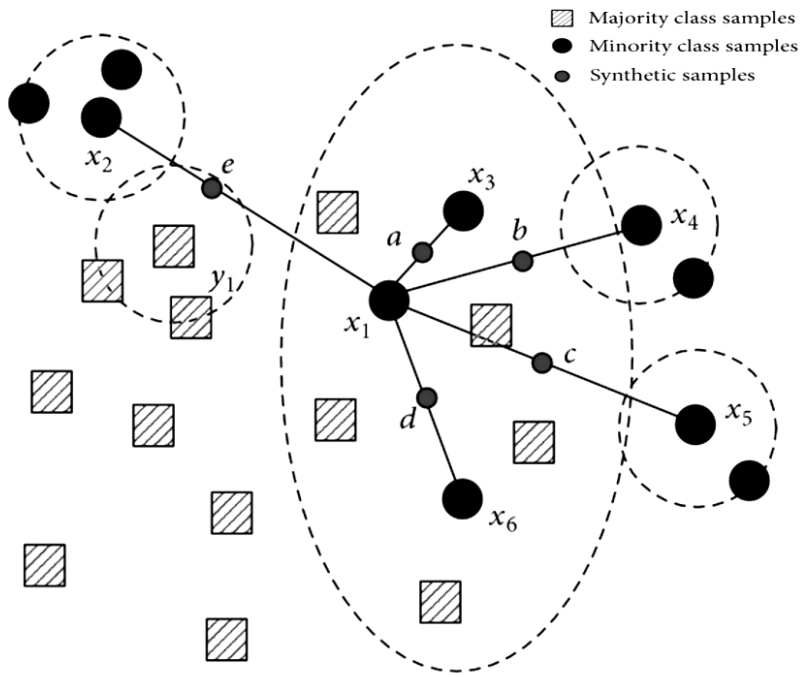


Figure 31: Illustration of SMOTE

*(Source: https://miro.medium.com/max/875/1*X8ef1P0PVmtWfX8Rnm8kHg.png)*

There are certain advances and extensions in the area of SMOTE. One of the popular extensions to SMOTE is Borderline SMOTE, Borderline SMOTE SVM, and Adaptive Synthetic Sampling (ADASYN).

Borderline SMOTE (Han et al., 2005): The difference with SMOTE is that instead of generating new synthetic samples for minority classes blindly, Borderline SMOTE creates synthetic samples along the decision boundary between two classes. This is because the samples near the borderline/decision boundary are more apt to be misclassified than the ones far from the borderline. Samples that are far from the decision boundary are not oversampled in this technique.

Borderline-SMOTE SVM (Nguyen et al., 2011): It is an alternative to Borderline-SMOTE where an SVM algorithm is used instead of a K-nearest neighbor algorithm to identify misclassified examples on the decision boundary. The minority class that is close to the support vectors becomes the focus for generating synthetic examples. This technique attempts to select regions at the decision boundary where there are fewer examples of the minority class.

Adaptive Synthetic Sampling(ADASYN) (He et al., 2008): It generates samples inversely proportional to the density of the samples in the minority class by generating more synthetic

samples in the region of feature space where the density of minority class examples are low, and fewer or none where density is high.

7.2 Binary Classification Algorithms

7.2.1 K-Nearest Neighbours (KNN)

It is a supervised machine learning algorithm that can be used to solve both classification and regression problems. The number of nearest neighbors to a new unknown variable that has to be predicted or classified is donated by the symbol 'K'. KNN calculates the distance from all points in the proximity of the unknown data and assigns the unknown point to the category where most neighbors are counted (majority voting). It is required to specify the value of 'K' for performing the algorithm. It is a distance-based method based on Euclidean distance. When dealing with the imbalanced dataset, the measurement becomes biased. So, resampling is done to ensure the balancing of data.

7.2.2 Support Vector Machines (Support Vector Classifiers)

Support Vector machines (SVM or SVC) are best suited for classification problems. The primary objective of SVC is to find a hyperplane in an n-dimensional space that distinctly classifies the data points. The dimensions of the hyperplane depend on the number of features (columns) in the feature vector. If the number of inputs is two, then the hyperplane is just a line, and if the number of input features is three, then the hyperplane becomes a 2-D plane. The best hyperplane is the one that represents the largest separation or margin between the two classes. SVC is robust to outliers. The algorithm has the characteristic of ignoring the outlier and finding the best hyperplane by adding a penalty (C) each time a point crosses the margin.

SVM has a kernel function that takes low dimensional input space and transforms it into a higher dimensional input space which could be useful in non-linear separation problems. In our study, we used linear and radial-based function (rbf) kernel to perform the binary classification.

The hyper-parameters used in the study are:

C (float): It is a regularization parameter, and its strength is inversely proportional to C. It must be strictly positive. The penalty is a squared l2 penalty. (default = 1.0)

Kernel: {'linear', 'rbf', 'sigmoid', 'poly', 'precomputed'}, default = 'rbf'.

It specifies the kernel type to be used in the algorithm.

Gamma (float): {'scale', 'auto'}

It is the kernel coefficient for ‘rbf’, ‘poly’, and ‘sigmoid’. (default = ‘scale’)

7.2.3 Decision Trees

The decision tree is a supervised learning algorithm that can be used for both classification and regression problems. It is mostly preferred for solving classification problems. It is a tree-structured classifier, where internal nodes represent the features of the dataset, branches represent the decision tree rules, and each leaf node represents the outcome. The two nodes present in the decision tree are the decision node and leaf node. Decision nodes are used to make a decision and have multiple branches, whereas leaf nodes are the output of those decisions made by the decision node, and they do not contain any further branches. To have a simple intuition, a decision tree simply asks a question, and based on the answer (yes/no), it further splits the tree into subtrees. The working of a decision tree classifier is shown in Figure 32. The root node represents the entire dataset which is where the decision tree starts.

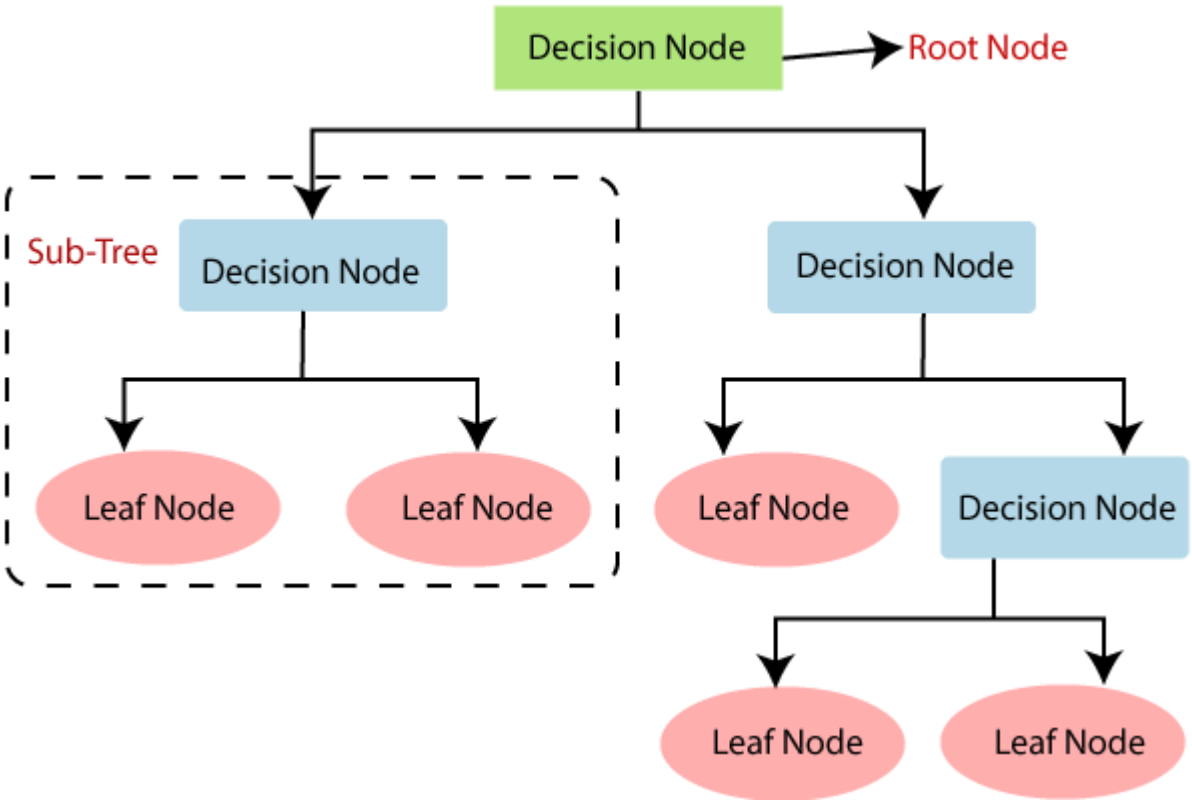


Figure 32: Working on a decision tree

(Source: https://static.javatpoint.com/tutorial/machine-learning/images/decision-tree-classification-algorithm.png)

The hyper-parameters of the decision tree used in the study are:

max_depth (int): It is the maximum depth of a tree. If none, then nodes are expanded until all leaves contain less than min_samples_split samples. (default = None)

min_samples_split (int or float): The minimum number of samples required to split an internal node. (default = 2)

min_samples_leaf (int or float): The minimum number of samples required to be at a leaf node. (default = 1)

7.2.4 Random Forest

Random forest is a very popular machine learning algorithm belonging to the supervised learning technique used for both classification and regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model. The random forest contains multiple decision trees on various subsets of the given dataset and takes the average value to improve the predictions of that dataset. The greater number of trees leads to higher accuracy and prevents overfitting. The working of a random forest classifier is shown in Figure 33. In addition to the parameters mentioned in decision trees, the random forest contains two more parameters, max_features for considering the features of the dataset and n_estimators, where we specify the number of trees we want to build.

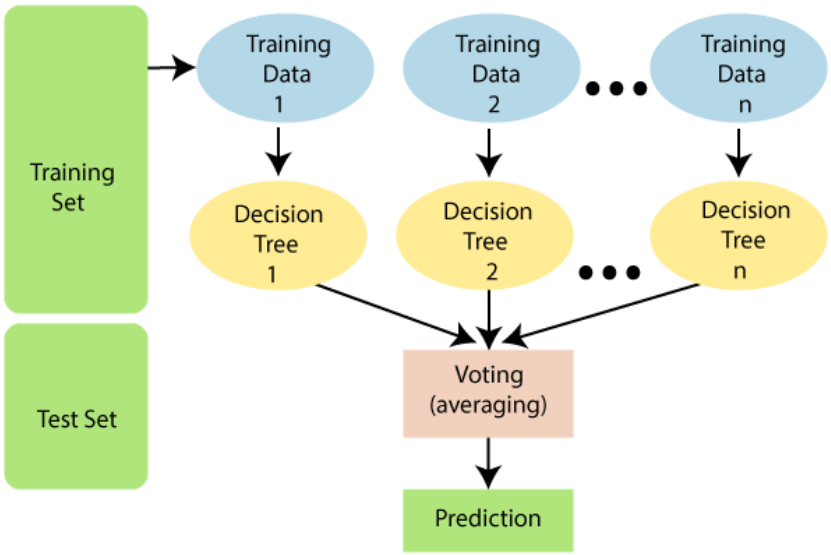


Figure 33: Working of a random forest classifier

7.2.5 Extreme Gradient Boosting (XG Boost)

XG Boost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XG Boost models majorly dominate in many Kaggle Competitions.

XB Boost is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework. Boosting is a technique of sequential tree building shown in Figure 34. Gradient boosting is a special case of boosting where errors are minimized by a gradient descent algorithm. It is an optimized gradient boosting algorithm through parallel processing, tree-pruning, and regularization to avoid overfitting or bias. In this algorithm, decision trees are created in sequential form. Weights play an important role in XG Boost. Weights are assigned to all the independent variables, which are then fed into the decision tree, which predicts results. The weight of variables predicted wrong by the tree is increased, and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

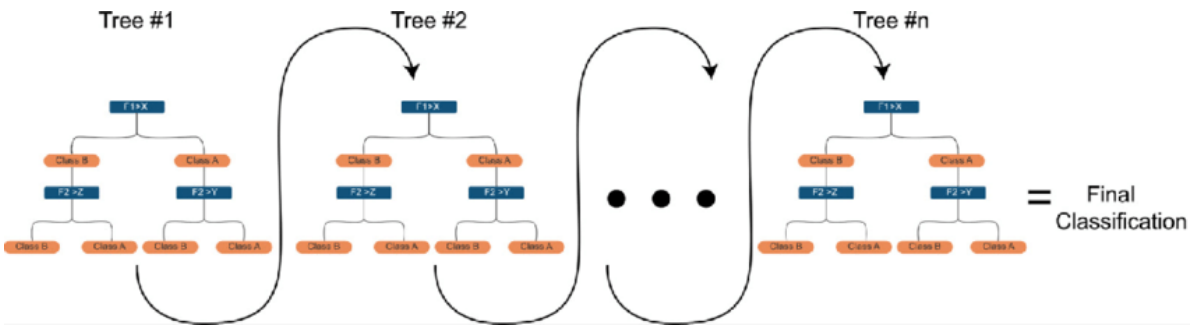


Figure 34: Working on a gradient boosting algorithm

The hyper-parameters of XG Boost used in this study are:

gamma (int or float): Minimum loss reduction required to make a further partition on a leaf node of the tree. (default = 0)

max_depth (int): The maximum depth of a tree. (default = 6)

min_child_weight: If the sum of weights in a leaf node is less than the min_child weight, then the building process will stop further partitioning. (default = 1)

a subsample (float): It is the ratio of training samples. Its range is (0,1]. A subsample value of 0.5 means randomly sampling half of the training data prior to growing trees. This will prevent overfitting. It will occur only once in every boosting iteration. (default =1)

colsample_bytree: It specifies the fraction of columns to be subsampled when constructing each tree. It occurs once for every tree constructed. Its range if (0,1]. (default =1)

CHAPTER 8 RESULTS AND ANALYSIS

8.1 Work Methodology

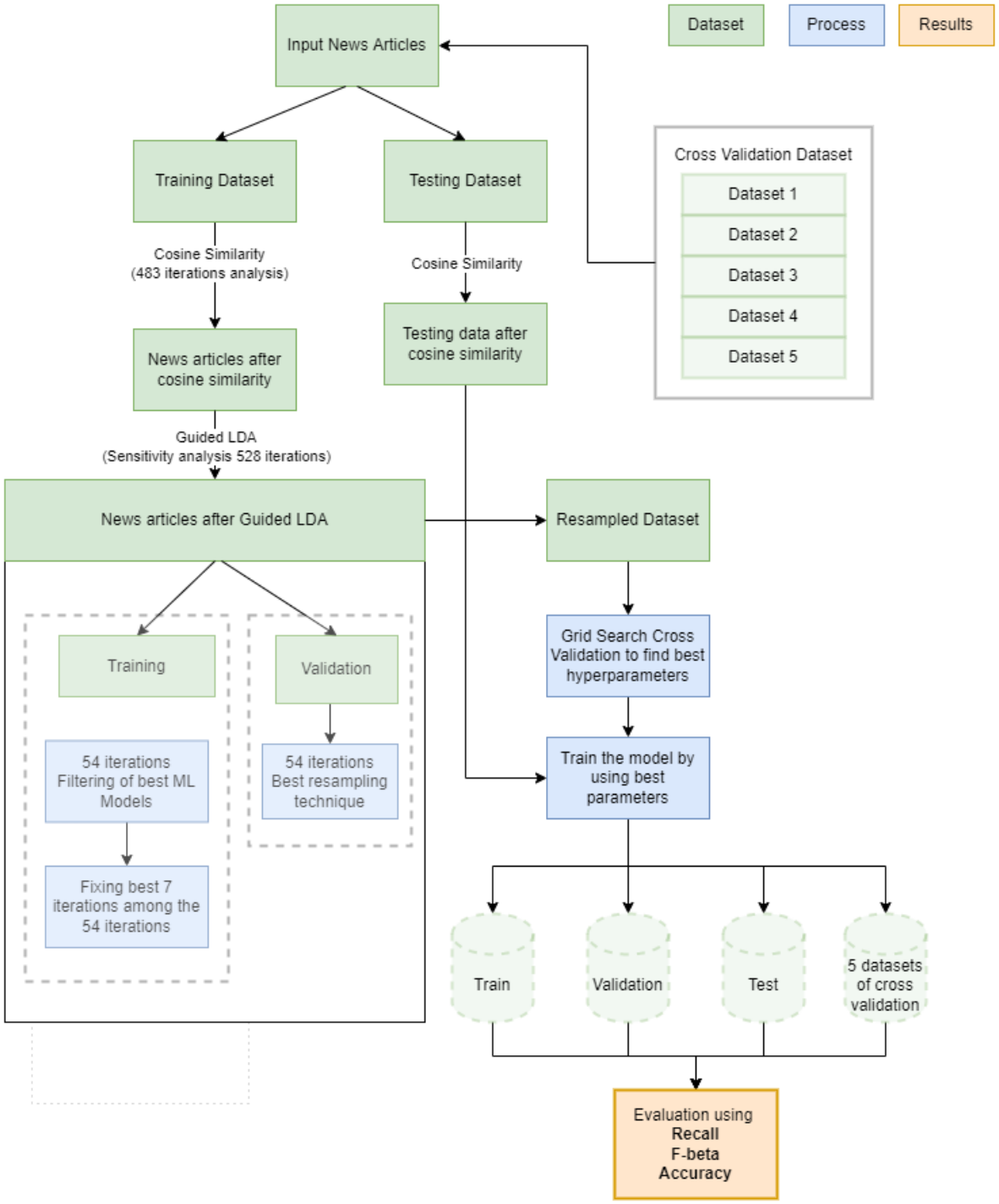


Figure 35: Work Methodology showing the entire process of this study

The work methodology for the entire study is shown in Figure 35. Comprehensive analysis is done on Duplicate Removal, Refining the dataset using Guided LDA, Resampling, and ML

Based Binary Classification. As discussed in chapter 4, the seven months of data consisting of 11208 articles are split into testing and training data based on the number of months. The first four months are considered as testing data, and the next three months of newspaper articles are considered as training data. The data distribution of testing and training data is shown in Figure 36. A process map of all the processes which the testing and the training data pass through in this analysis is shown in Figure 37. The testing data is applied later on to the best parameters achieved in each process by the training dataset.

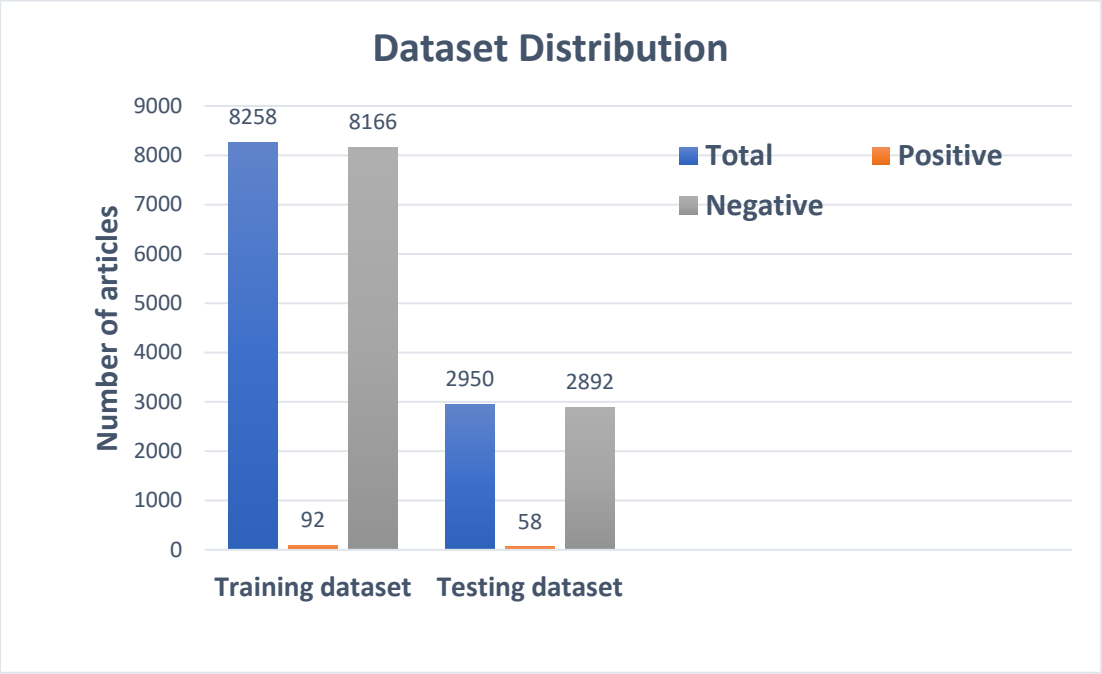


Figure 36: Chart showing dataset distribution of articles

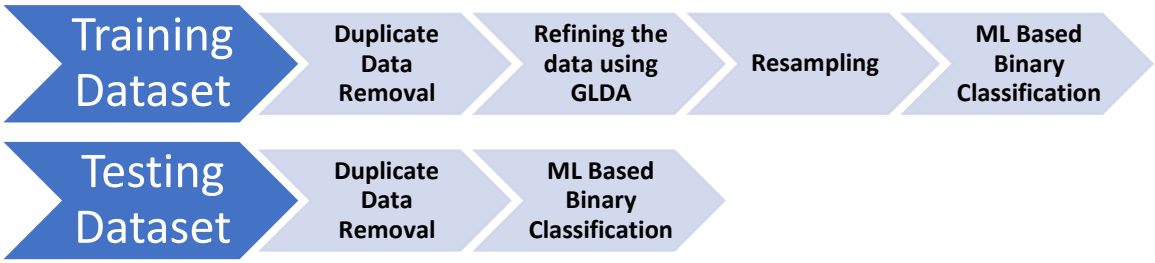


Figure 37: Process map of the testing and the training data

8.2 Training Dataset

8.2.1 Duplicate Data Removal

Best parameters after 483 iterations: n-grams = (1,1), cosine similarity value = 0.65

Input Dataset size: Total: 8258, Positive: 92, Negative: 8166

Output Dataset size: Total: 4155, Positive: 70, Negative: 4085

The condition given to the dataset is to remove articles which are having more than 65% similarity between them. There are 22 positive articles removed from the input data as they are found to be similar. Among them, 15 articles are similar articles. Out of 77 unique construction articles, 70 articles are retained which means 90.9% accuracy. Overall, by performing cosine similarity, 4080 duplicate negative articles were removed at the cost of information loss of 7 positive articles.

8.2.2 Refining the dataset using Guided-LDA

Input Dataset size: Total: 4155, Positive: 70, Negative: 4085

Output Dataset size: Total: 2689, Positive: 70, Negative: 2619

Best parameters after 528 iterations: alpha = 0.1, eta = 0.1, no of iterations = 4000, seed confidence = 0.1

By using Guided-LDA, all the positive articles and negative articles which are having keywords close to positive articles are retained, and we could filter out and remove 1466 irrelevant negative articles from the dataset, and this helped in improving the binary classification performance.

8.2.2.1 Sensitivity analysis of Guided LDA

The sensitivity analysis of GLDA is done by calculating the recall, precision and accuracy metrics by varying the hyper parameters. The total number of iterations are 528.

Table 3: Values of hyperparameters considered for GLDA

Hyperparameters	**The list of hyperparameters varied**	**No of parameters**
Alpha	[0.01, 0.02, 0.05, 0.1]	4
Beta	[0.01,0.05,0.1, 0.15,0.2,0.25]	6
Seed_confidence	[2000, 4000]	2
No. of iterations	[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.975]	11

Table 4: : Sensitivity analysis by varying hyper parameters

Alpha	Beta	n_iter	seed_conf	Recall	Accuracy
0.1	0.1	4000	0.1	100	36.19735
0.1	0.01	2000	0.1	100	36.17329
0.1	0.01	2000	0.975	100	36.17329
0.1	0.1	4000	0.2	100	36.17329
0.1	0.1	4000	0.6	100	36.17329
0.1	0.1	4000	0.7	100	36.17329
0.1	0.1	4000	0.3	100	36.14922
0.1	0.1	4000	0.9	100	36.14922
0.1	0.1	4000	0.95	100	36.14922
0.1	0.1	4000	0.5	100	36.12515
0.1	0.1	4000	0.8	100	36.12515
0.1	0.1	4000	0.975	100	36.12515
0.1	0.01	2000	0.4	100	36.10108
0.1	0.1	4000	0.4	100	36.10108

The results show that all the 528 iterations gave a recall score of 100 which means all the positive samples are predicted positively. The precision and accuracy values also doesn't vary much. These results shows that the Guided LDA is robust to the hyper parameters and successful in filtering the articles having construction related key words. The accuracy and values seem to look low because of the high imbalance in the dataset as the negative samples occupy more nearly 99% of the dataset. Though, the variation of accuracy is calculated with respect to every hyperparameter to check the variance and the results are plotted in Figure 38, Figure 39, Figure 40. The results indicate that the Guided LDA model is robust to hyper parameters.

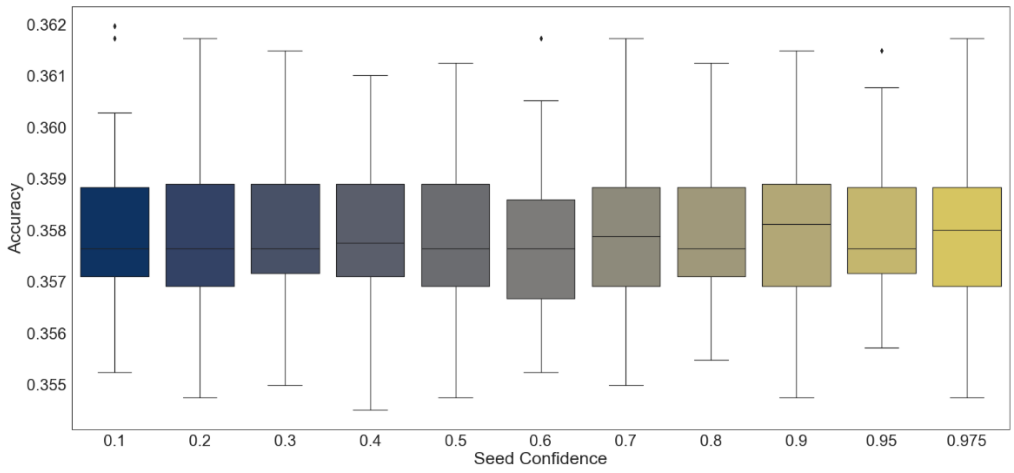


Figure 38: Variation of accuracy with seed confidence

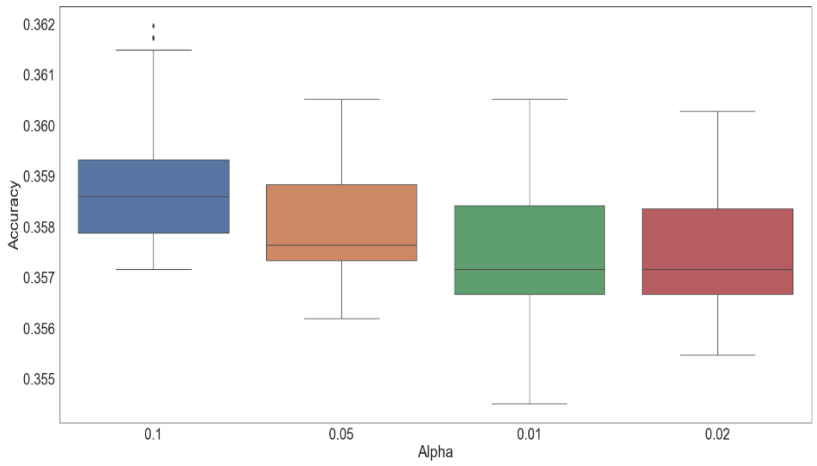


Figure 39: Variation of accuracy with alpha

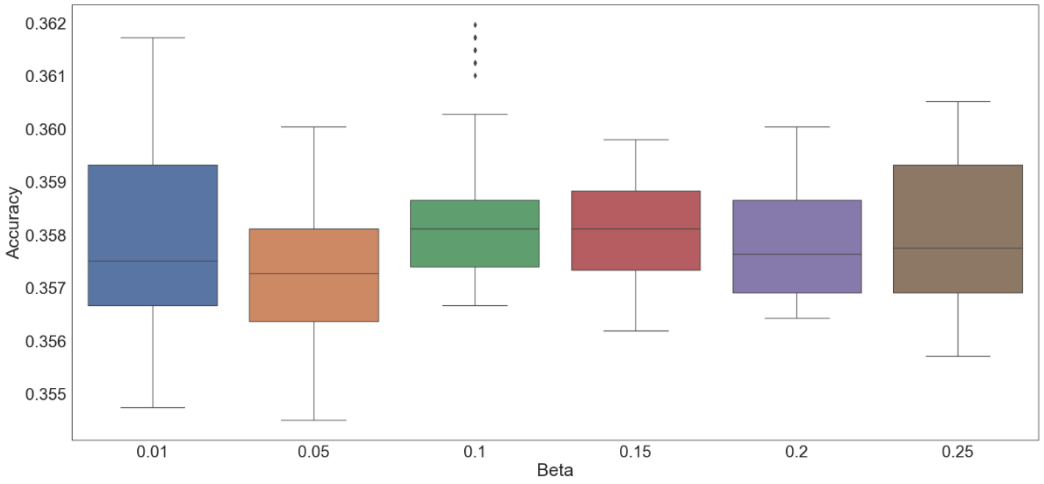


Figure 40: Variation of accuracy with beta

8.2.3 ML Based on binary classification

Input Dataset size: Total: 2689, Positive: 70, Negative: 2619

The Input Dataset size is split into training and validation datasets at an 80-20% ratio.

8.2.3.1 Resampling Analysis to find best resampling technique

To do the resampling, we require a robust re-sampling technique that is not sensitive to the data. The four resampling techniques: SMOTE, Borderline SMOTE, ADASYN, and SVM-SMOTE, are analyzed in this study for the validation dataset.

The hyper parameters for oversampling and under sampling used are:

Over = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]

Under = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]

For over sampling strategy of 0.1, we use all the under-sampling strategy values mentioned above. As we move towards higher over sampling percentages, we can only use the same or greater percentage of sampling strategy for under-sampling. For example, for over sampling strategy of 0.5, we can use under sampling strategy of 0.5, 0.6, 0.7, 0.8, 0.9, and 1. There is a total of 54 iterations considering all the possibilities using the mentioned sampling strategy values.

The resampling techniques are employed on all the iterations, and the average values of accuracy, recall, and f-beta score are calculated, and the results are shown in Table 5. The results convey that ADASYN and SVM-SMOTE performed well in accuracy score, and SVM-SMOTE and SMOTE techniques gave good results in recall score and f-beta score. SVM-SMOTE shown recall value greater than 70% in all the cases. SVM SMOTE proved to be good as it gave excellent results in all the three metrics and robust enough as it has been iterated over all the possible sampling strategies, and results are consistent. On the basis of this analysis, SVM-SMOTE's added advantages over the other sampling strategies resulted in selecting SVM-SMOTE as the resampling technique in this study.

Table 5: Resampling analysis to find the best resampling technique

Algorithm	Accuracy				Recall				Fbeta			
	Adasyn	SMOTE	Borderline	SVM	Adasyn	SMOTE	Borderline	SVM	Adasyn	SMOTE	Borderline	SVM
DT	**0.98**	0.97	0.97	0.97	0.65	**0.78**	0.67	**0.78**	0.65	**0.69**	0.62	0.67
KNN	**0.52**	0.35	0.37	0.38	0.93	**0.94**	**0.94**	**0.94**	**0.20**	0.16	0.17	0.17
RF	**0.98**	0.97	0.97	**0.98**	0.17	0.30	0.25	**0.33**	0.20	0.32	0.27	**0.36**
SVC_RBF	**0.99**	**0.99**	**0.99**	**0.99**	0.71	**0.74**	0.73	**0.74**	0.75	**0.76**	**0.76**	**0.76**
SVC_I	**0.99**	**0.99**	**0.99**	**0.99**	0.77	**0.81**	0.79	0.78	0.80	**0.83**	0.81	0.81
XBG	**0.99**	**0.99**	**0.99**	**0.99**	0.69	0.75	0.75	**0.77**	0.72	0.74	0.74	**0.76**

8.2.3.2 Analysis to find the variance of models

The ML algorithms used for training the dataset are KNN, XG Boost, Decision Trees, Random Forest, and SVC. The parameters used for training the dataset are default parameters which are done to check the feasibility of the parameters. Since we do not know the best parameters for under-sampling and over-sampling, the model is trained on all the 54 iterations mentioned in the resampling analysis. All the models except KNN gave accuracy close to 100% for all the iterations, but the KNN algorithm varied along with the iterations. The variation of KNN over the iterations is shown in Table 6. The highest accuracy itself is very low, and it classifies most of the articles as positively labeled, which results in low accuracy. So KNN algorithm is eliminated after this step as we want our model to be more robust towards various sampling strategies as we do not know how the future dataset is going to behave when algorithms having less sensitivity are used.

Table 6: KNN algorithm variation in 54 iterations

Value	***Accuracy***	***Recall***	***F-beta***
Minimum	0.526	1	0.61
Maximum	0.815	1	0.93
Average	0.695	1	0.85

Further, the machine learning algorithms are tested on the validation dataset to check the nature of algorithms based on the parameters. Analyzing the variance of the metrics, fbeta (Figure 41), accuracy (Figure 42), and recall (Figure 43) values over the ML classifiers, it is observed that KNN has low values of F-beta score and high variance in accuracy. The random forest has a high variance in the F-beta score and recalls score making it sensitive to the data. From this analysis, it is understood that KNN and RF are not suitable to use for this type of dataset.

Among the six classifiers, two are eliminated, and the remaining four classifiers are tested further on hyper-parameters.

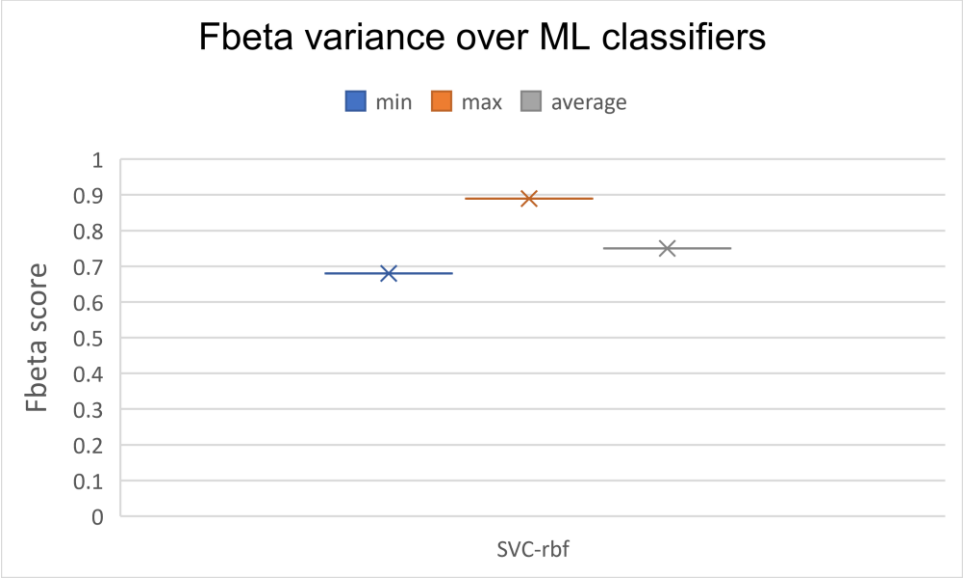


Figure 41: Fbeta variance over ML classifier

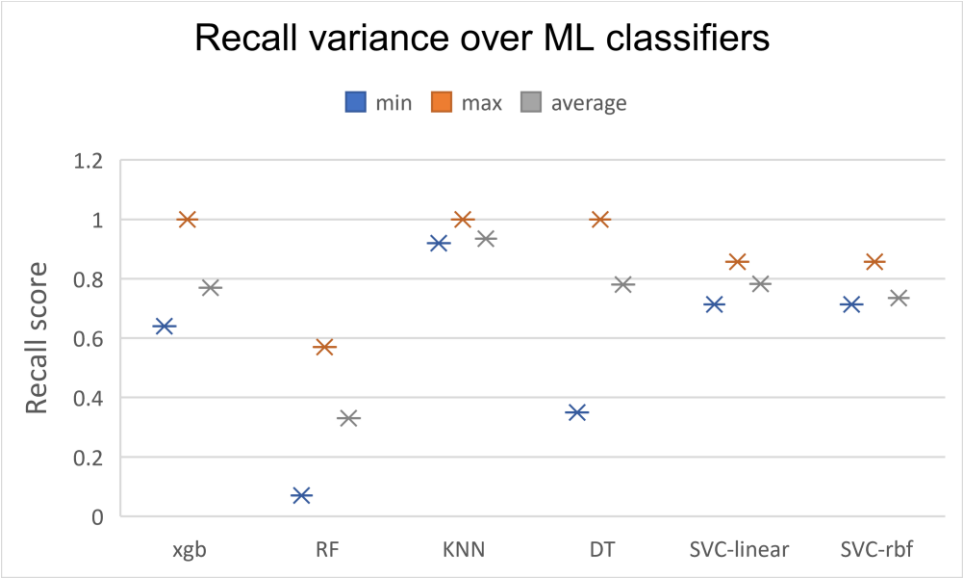


Figure 42: Recall variance over ML classifier

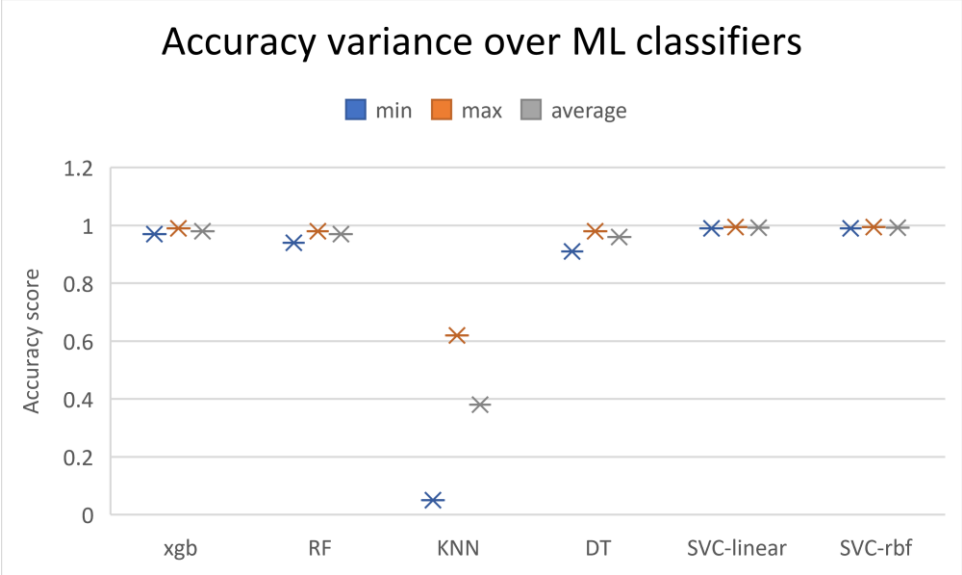


Figure 43: Accuracy variance over ML classifier

8.2.3.3 Hyperparameters analysis for fixing the best hyperparameters

The hyperparameters present in this data can be broadly classified into two classes. One class of hyperparameters is for choosing the sampling strategy, and the other class of hyperparameters is for choosing the parameters in the machine learning algorithms.

There are two hyperparameters in the former class (over and under), are used for giving a sampling ratio for oversampling the data using the oversampling techniques and under-sampling the data using random under-sampling respectively. F-beta with beta =2 is a reliable metric to find out that the classifier is performing decently in both attaining good recall and a good f1-score. Hence, this metric is chosen to set the values of over and under sampling strategies. The top 10 values of the f-beta score are achieved in SVC-linear, XG Boost, and SVC-rbf kernel, which is shown in Table 7, Table 8, and Table 9. It is observed that most of them have a low value of under-sampling, such as 0.1, and an over-sampling strategy varying between 0.5 to 1. These parameters are considered for performing the grid search cross-validation to each of the machine learning classifiers.

Final selected learning parameters for sampling strategy:

Over = [0.1]

Under = [0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1].

In total, there are 7 iterations done in each of the classifiers to find out the best hyperparameters of the classifier.

Table 7: Top 10 values of f-beta score in SVC - linear kernel classifier

Iteration	over	under	acc	recall	fbeta
1	0.1	0.2	0.994424	0.857143	0.869565
2	0.1	0.3	0.994424	0.857143	0.869565
3	0.1	0.4	0.994424	0.857143	0.869565
4	0.1	0.5	0.994424	0.857143	0.869565
5	0.1	0.6	0.994424	0.857143	0.869565
6	0.1	0.7	0.994424	0.857143	0.869565
7	0.1	0.8	0.994424	0.857143	0.869565
9	0.1	1	0.994424	0.857143	0.869565
12	0.2	0.4	0.994424	0.857143	0.869565
13	0.2	0.5	0.994424	0.857143	0.869565

Table 8: Top 10 values of f-beta score in XG Boost algorithm

Iteration	over	under	acc	recall	fbeta
16	0.2	0.8	0.990706	0.928571	0.890411
3	0.1	0.4	0.988848	0.928571	0.878378
17	0.2	0.9	0.98513	0.928571	0.855263
32	0.4	0.9	0.990706	0.857143	0.84507
4	0.1	0.5	0.983271	0.928571	0.844156
18	0.2	1	0.983271	0.928571	0.844156
15	0.2	0.7	0.988848	0.857143	0.833333
5	0.1	0.6	0.979554	0.928571	0.822785
7	0.1	0.8	0.979554	0.928571	0.822785

Table 9: Top 10 values of f-beta score in SVC- rbf kernel

Iteration	over	under	acc	recall	fbeta
6	0.1	0.7	0.994424	0.857143	0.869565
7	0.1	0.8	0.994424	0.857143	0.869565
8	0.1	0.9	0.994424	0.857143	0.869565
9	0.1	1	0.994424	0.857143	0.869565
3	0.1	0.4	0.992565	0.785714	0.808824
4	0.1	0.5	0.992565	0.785714	0.808824
5	0.1	0.6	0.992565	0.785714	0.808824
15	0.2	0.7	0.992565	0.785714	0.808824
16	0.2	0.8	0.992565	0.785714	0.808824
17	0.2	0.9	0.992565	0.785714	0.808824

8.2.3.4 Grid search cross validation for choosing the best ML & Resampling hyperparameters

The seven iterations mentioned above are used to find out the best parameters in the machine learning algorithms using grid search cross-validation.

8.2.3.4.1 Hyperparameters for Decision Tree Classifier

Table 10: Values of hyperparameters considered for grid search cross-validation of a decision tree classifier

Hyperparameters	**The list of hyperparameters varied**	**No of parameters**
Max_depth	[3, 5, 10, 15, 20, None]	6
Min_samples_split	[2, 5, 7, 10]	4
Min_samples_leaf	[1, 2, 5]	3

In total, there are 72 combinations that are varied in the grid search, and the optimal parameters for the seven sampling strategies are obtained. Figure 44 shows the best parameters obtained in each of the sampling strategies. The scoring parameter used in the grid search is recalled as the positive dataset classification is more important for us. It is observed that the values do not vary much along with the different values of the sampling strategy. In this case, the second iteration, which is over = 0.1 and under = 0.5, is used for the testing data since it has the parameter values which are mostly occurring in the 7 iterations.

The selected parameters are: over = 0.1, under = 0.5, max_depth = 10, min_samples_leaf =1, min_samples_split =2

```
over,under 0.1 0.4
Parameters {'max_depth': 5, 'min_samples_leaf': 5, 'min_samples_split': 2}
Best recall score 0.8962264150943398
Train F_beta Score :  0.9662576687116565
over,under 0.1 0.5
Parameters {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best recall score 0.9461538461538461
Train F_beta Score :  1.0
over,under 0.1 0.6
Parameters {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 10}
Best recall score 0.954644412191582
Train F_beta Score :  0.9961685823754789
over,under 0.1 0.7
Parameters {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 10}
Best recall score 0.9807692307692306
Train F_beta Score :  0.9969325153374233
over,under 0.1 0.8
Parameters {'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best recall score 0.9579100145137881
Train F_beta Score :  0.9969325153374233
over,under 0.1 0.9
Parameters {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best recall score 0.9695936139332366
Train F_beta Score :  1.0
over,under 0.1 1
Parameters {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best recall score 0.9579100145137881
Train F_beta Score :  1.0
```

Figure 44: Results of grid-search cross-validation of Decision Tree for extracting best parameters

8.2.3.4.2 Hyperparameters for XG Boost Classifier

Table 11: Values of hyperparameters considered for grid search cross-validation of XG Boost classifier

Hyperparameters	**The list of hyperparameters varied**	**No of parameters**
Min_child_weight	[1, 2, 4, 6, 8, 10]	6
Gamma	[0.5, 1, 1.5, 2, 5]	4
Sub_sample	[0.2, 0.4, 0.6, 0.8, 1]	5
Colsample_bytree	[0.2, 0.4, 0.6, 0.8, 1]	5
Max_depth	[1, 2, 3, 4, 5]	5

In total, there are 3000 combinations that are varied using the grid search, and the optimal parameters for the seven sampling strategies are obtained. Shows the best parameters obtained in each of the sampling strategies. The scoring parameter used in the grid search is recalled as the positive dataset classification is more important for us. It is observed that the values do not

vary much along with the different values of the sampling strategy. In this case, the second iteration, which is over = 0.1 and under = 0.9, is used for the testing data since it gave the best parameters, which are frequently occurring in all the other iterations.

The selected parameters are: over = 0.1, under = 0.9, colsample_bytree = 0.2, gamma = 0.5, max_depth = 4, min_child_weight = 1, subsample = 0.6.

```
over,under 0.1 0.4
Best Recall: 95.00 %
Best Parameters: {'colsample_bytree': 0.2, 'gamma': 1, 'max_depth': 4, 'min_child_weight': 1, 'subsample': 0.6}
Train F_beta Score :  1.0

over,under 0.1 0.5
Best Recall: 96.15 %
Best Parameters: {'colsample_bytree': 0.2, 'gamma': 1, 'max_depth': 5, 'min_child_weight': 1, 'subsample': 0.8}
Train F_beta Score :  1.0

over,under 0.1 0.6
Best Recall: 96.59 %
Best Parameters: {'colsample_bytree': 1.0, 'gamma': 0.5, 'max_depth': 2, 'min_child_weight': 1, 'subsample': 0.2}
Train F_beta Score :  0.9961685823754789

over,under 0.1 0.7
Best Recall: 96.92 %
Best Parameters: {'colsample_bytree': 0.2, 'gamma': 0.5, 'max_depth': 3, 'min_child_weight': 1, 'subsample': 0.6}
Train F_beta Score :  1.0

over,under 0.1 0.8
Best Recall: 97.70 %
Best Parameters: {'colsample_bytree': 0.2, 'gamma': 0.5, 'max_depth': 4, 'min_child_weight': 1, 'subsample': 0.6}
Train F_beta Score :  1.0

over,under 0.1 0.9
Best Recall: 96.97 %
Best Parameters: {'colsample_bytree': 0.2, 'gamma': 1, 'max_depth': 5, 'min_child_weight': 2, 'subsample': 0.6}
Train F_beta Score :  0.9992343032159265

over,under 0.1 1
Best Recall: 98.08 %
Best Parameters: {'colsample_bytree': 0.6, 'gamma': 0.5, 'max_depth': 3, 'min_child_weight': 1, 'subsample': 0.6}
Train F_beta Score :  1.0
```

Figure 45: Results of grid-search cross validation of XG Boost classifier for extracting best parameters

8.2.3.4.3 Hyperparameters for Support Vector Classifier

Table 12: : Values of hyperparameters considered for grid search cross validation of Support Vector classifier

Hyperparameters	List of hyperparameters varied	No of parameters
C	**[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]**	**9**
Kernel	**['linear', 'rbf']**	**2**
Gamma*	**[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]**	**9**

*Only applicable for 'rbf' kernel

In total, there were 9 parameters for the SVM-linear kernel and 81 parameters for the SVM-rbf kernel. The parameters are varied using the grid search, and the optimal parameters for the seven sampling strategies are obtained. Figure 46 shows the best parameters obtained in each of the sampling strategies. It is observed that the values do not vary much along with the different values of the sampling strategy. In this case, the second iteration, which is over = 0.1 and under = 0.9, is used for the testing data since it gave the best parameters with a high recall score and also contains parameter values that are frequently occurring in other iterations of the sampling strategies.

The selected parameters are: over = 0.1, under = 0.9, C = 0.9, kernel = "linear"

```
over,under 0.1 0.4
Best Recall: 98.46 %
Best Parameters: {'C': 0.9, 'kernel': 'linear'}
Train F_beta Score :  0.9954058192955589
over,under 0.1 0.5
Best Recall: 98.46 %
Best Parameters: {'C': 0.3, 'kernel': 'linear'}
Train F_beta Score :  0.9861857252494246
over,under 0.1 0.6
Best Recall: 98.11 %
Best Parameters: {'C': 0.5, 'kernel': 'linear'}
Train F_beta Score :  0.9900230237912511
over,under 0.1 0.7
Best Recall: 98.85 %
Best Parameters: {'C': 0.9, 'kernel': 'linear'}
Train F_beta Score :  0.9992343032159265
over,under 0.1 0.8
Best Recall: 98.85 %
Best Parameters: {'C': 0.9, 'kernel': 'linear'}
Train F_beta Score :  0.9992343032159265
over,under 0.1 0.9
Best Recall: 98.87 %
Best Parameters: {'C': 0.9, 'kernel': 'linear'}
Train F_beta Score :  0.9992343032159265
over,under 0.1 1
Best Recall: 98.85 %
Best Parameters: {'C': 0.8, 'kernel': 'linear'}
Train F_beta Score :  0.9992343032159265
```

Figure 46: Results of grid-search cross-validation of SVM Classifier for extracting best parameters

8.2.3.5 Performing binary classification on the training dataset

The parameters are trained on the validation set, and the best parameters are chosen using grid search cross-validation which is in turn applied to the testing dataset. The results of the performance on the training dataset by the three models are shown in Figure 47 and the performance on the validation set is shown in Figure 48. The SVC model gave the best accuracy with 99% and 94.5% f-beta score. The second-best results were obtained by XG Boost model with 98% accuracy and 88% f-beta score. But on the validation set, XG Boost turned out to be more reliable than SVC.

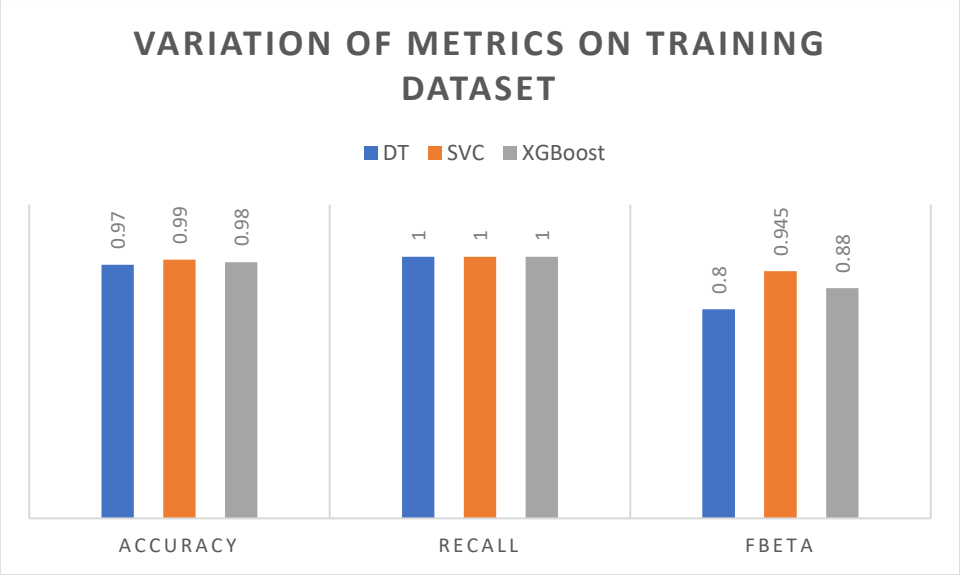


Figure 47: Performance of ML algorithms on the training dataset

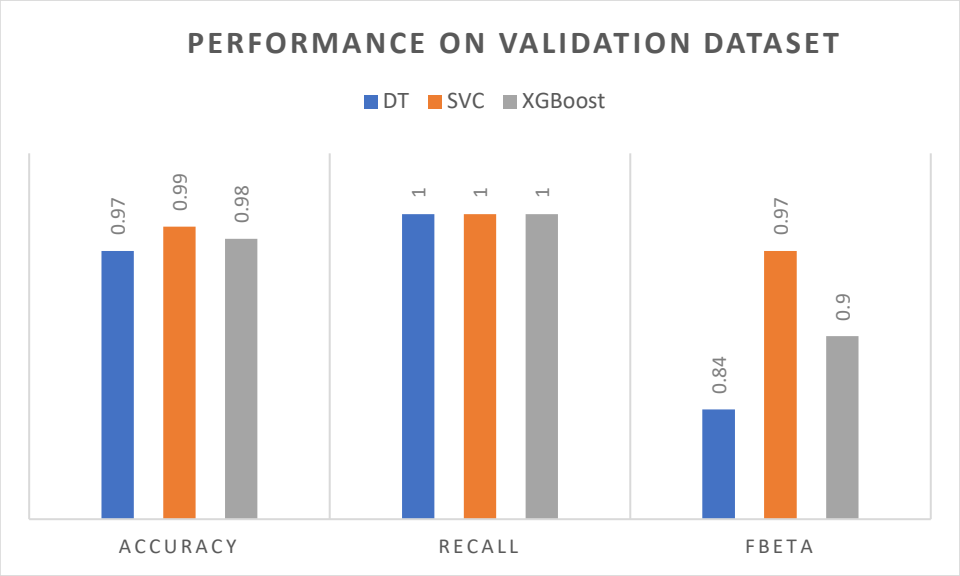


Figure 48: Performance of ML algorithms on the validation dataset

8.2.4 Summary of the training data

The Figure 49 shows the results of count of positive and negative samples at each process mentioned for the training dataset. The input articles under goes the process of duplicate removal where the duplicates were removed followed by filtration of data through Guided-LDA. The resultant dataset undergoes data cleaning, data pre-processing before resampling and finally it is fed as input for the machine learning model to fit.

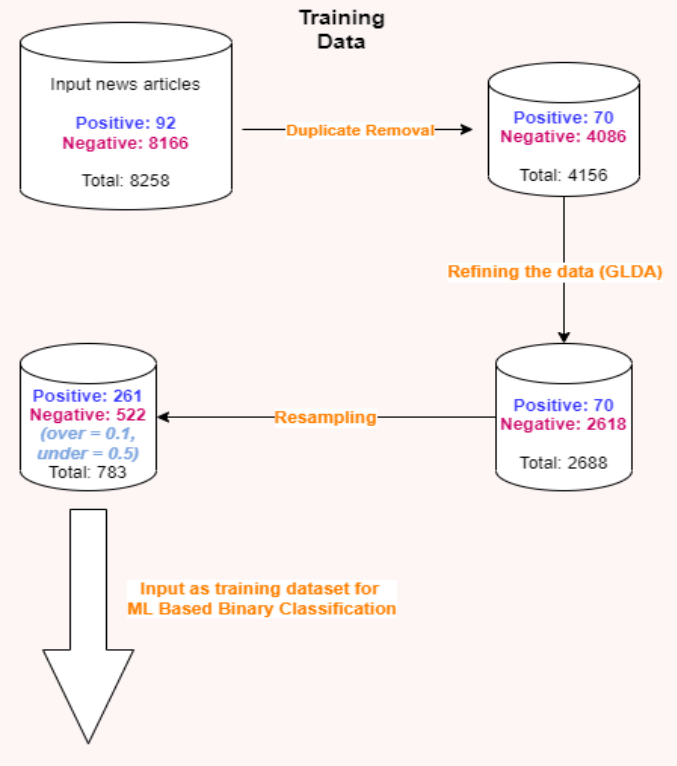


Figure 49: Results obtained step-by-step on the training data

8.3 Testing Dataset

The testing dataset has a size of 2950 news articles, of which 58 articles are positive articles. As mentioned, the testing dataset undergoes duplicate data removal and is then tested on the models trained by the training dataset.

8.3.1 Duplicate Data Removal

The duplicate removal is necessary to remove similar construction articles so that we do not over-estimate the number of construction accidents. The best parameters achieved by the duplicate data analysis through the training dataset are applied to the testing dataset. The cosine similarity value of 0.65 using unigrams TF-IDF vectorization and the articles which have similarity greater than 65% are removed. After the cosine similarity is done, the retained positive articles are found to be reasonably good with retention of 96.6% unique construction articles (only 2 articles are lost out of 58) and detected 11 similar positive articles are removed through cosine similarity. After the process, the testing dataset is reduced from a sample size of 2950 to almost half, which is 1456. The details are mentioned in Table 13.

Table 13: Testing dataset before and after cosine similarity

Testing Dataset							
Initial Dataset			After Cosine Similarity			similar positive articles	Retained positive articles
Positive	Negative	Total	Positive	Negative	Total		
58	2892	2950	45	1411	1456	11	96.6%

8.3.2 ML-based Binary classification

8.3.2.1 Parameters obtained from training on the training dataset

Table 14 summarises the best parameters obtained from the grid-search cross-validation on the machine learning classifiers by varying the best hyperparameters obtained by resampling analysis.

Table 14: Best hyperparameters obtained on ML classifiers

Best parameters obtained on XG Boost classifier:
over = 0.1, under = 0.9, colsample_bytree = 0.2, gamma = 0.5, max_depth = 4, min_child_weight = 1, subsample = 0.6.
Best parameters obtained on SVM classifier:
over = 0.1, under = 0.9, C = 0.9, kernel = “linear”
Best parameters obtained on Decision Tree classifier:
over = 0.1, under = 0.5, max_depth = 10, min_samples_leaf =1, min_samples_split =2

8.3.2.2 Performing binary classification on the testing dataset

The best parameters obtained from the respectively trained ML classifiers are used on the testing dataset. The performance metrics are shown in Figure 50. The best accuracy is achieved using XG Boost and SVC classifier. The best recall score (80%) is achieved using a Decision Tree followed by XG Boost. The best f-beta score is achieved using XG Boost followed by SVC linear. Depending on the results achieved from the training data, the Decision Tree gave the highest recall score by detecting 36 out of the 45 positive samples. But the disadvantage is that it has low f-beta and accuracy values compared to the other two classifiers. XG Boost, on the other hand, detected 33 out of 45 positive samples. The confusion matrix of the training dataset is shown in Table 15. The XG Boost algorithm provides a cutting edge over the other two algorithms by providing better accuracy and better recall.

Table 15: Confusion matrix on a testing dataset of the three ML classifiers

	Confusion matrix					
	Decision Tree		SVC		XG Boost	
	Predicted 0	predicted 1	Predicted 0	predicted 1	Predicted 0	predicted 1
Actual 0	1321	89	1392	18	1383	27
Actual 1	9	36	15	30	12	33

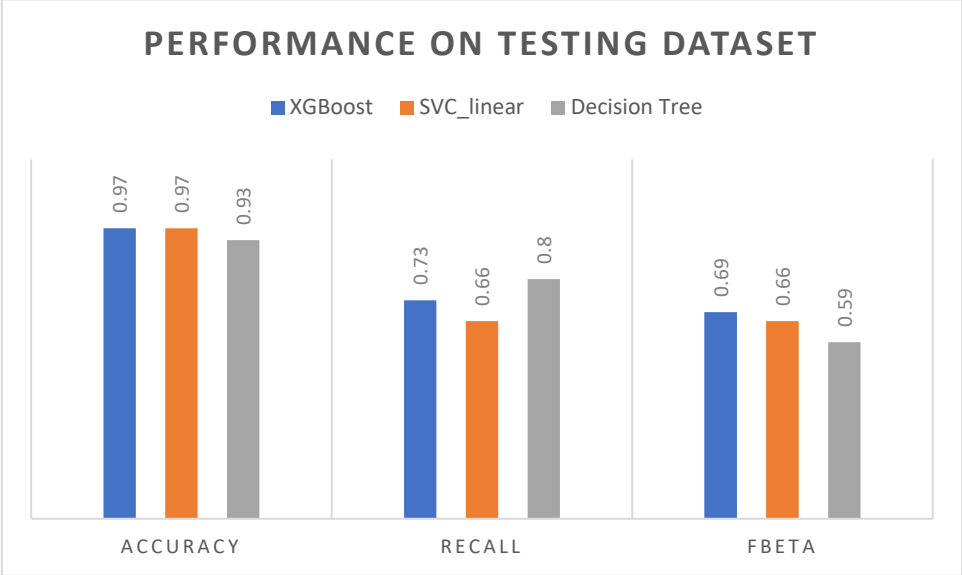


Figure 50: Performance of ML algorithms on Testing data

8.3.3 Summary

The Figure 51 shows the results of count of positive and negative samples at each process for the testing dataset. The input articles under goes the process of duplicate removal where the duplicates were removed followed by predictions using the training models.

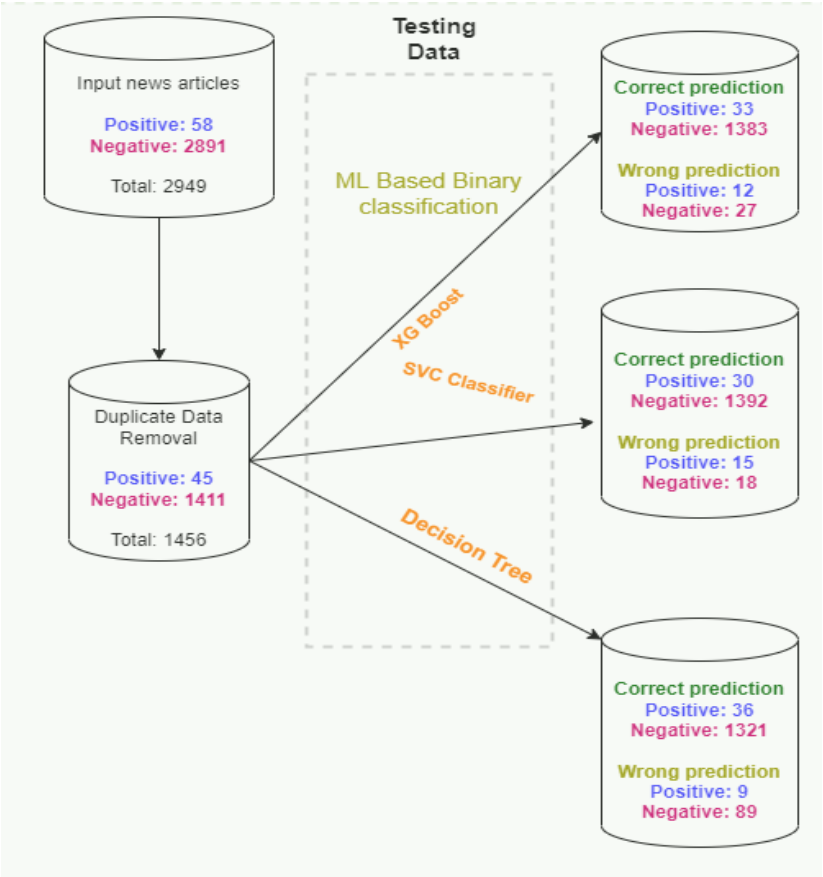


Figure 51: Results obtained step-by-step on the testing data

8.4 Cross-Validation

The proposed ML algorithms are predicted only on a single testing data. To know the robustness of the models, they have to be checked on multiple data samples. Since we have limited data samples, it is essential to know how the model performs in general and to check its performance, k-Fold cross-validation is a reliable method. It makes predictions on data not used during the training of the model. Stratified k-Fold cross-validation is implemented to ensure that each fold has the same proportion of observations within a given categorical value.

In our study, we have divided the 7 monthly news data sample sets of 11208 newspaper articles into 5 folds (cv = 5). Stratified k-fold cross validation shuffles the dataset randomly into 5 groups, and for each unique group, it takes the hold out-group as the test data set and the remaining group as the training dataset making an 80-20% train-test data ratio by maintaining the proportions of class imbalance. The five training datasets are passed through the whole process of duplicate removal, Guided-LDA, and then the classifiers with the proposed parameters fit a model on the training dataset and evaluate it on the test data set, which is passed

through duplicate removal. The maximum, mean, and minimum values obtained from the five datasets are examined to know the variability of the model on a given dataset.

8.4.1 Cosine Similarity & Guided LDA

The 5-fold training and testing datasets undergo the process mentioned in chapter5. Table 16 and Table 17 shows the results of the training and testing data. In both the datasets, the unique positive samples which are not similar are retained with greater than 95% accuracy in all the datasets, ensuring the robustness of the model in the cosine similarity process.

The Guided-LDA model is also found to be reliable in Table 16; Table 17 shows a recall of 100% by ensuring all the positive samples remained after the cosine similarity check was retained and the irrelevant data was found to be removed.

Table 16: 5-fold training dataset size at each process of the training dataset

Training Dataset											
5 Fold Dataset	Initial Dataset			After Cosine Similarity			similar positive articles	Retained positive	After Guided-LDA		
	Positive	Negative	Total	Positive	Negative	Total			Positive	Negative	Total
Dataset1	120	8751	8871	88	4372	4460	30	98.3%	88	2733	2821
Dataset2	120	8716	8836	95	4398	4493	23	98.3%	95	2764	2859
Dataset3	120	8701	8821	92	4486	4578	24	96.7%	92	2686	2778
Dataset4	120	8703	8823	90	4414	4504	26	96.7%	90	2613	2703
Dataset5	120	8765	8885	95	4417	4512	22	97.5%	95	2821	2916

Table 17: 5-fold testing dataset size at each process of the testing dataset

Testing Dataset								
5 Fold Dataset	Initial Dataset			After Cosine Similarity			similar positive articles	Retained positive
	Positive	Negative	Total	Positive	Negative	Total		
Dataset1	30	2212	2242	26	1122	1148	2	93.3%
Dataset2	30	2212	2242	20	1157	1177	9	96.7%
Dataset3	30	2212	2242	23	1180	1203	7	100.0%
Dataset4	30	2211	2241	25	1231	1256	5	100.0%
Dataset5	30	2211	2241	22	1144	1166	8	100.0%

8.4.2 ML-based Binary Classification

The training dataset, after undergoing the process of duplicate removal and filtration of data, code is trained on the parameters mentioned in 6.2.2.1. The model is fit on the training dataset, and the predictions are made on the testing dataset, which had already undergone cosine similarity. The results of the testing dataset are shown in Table 18. From Figure 52, Figure 53, and Figure 54, it is observed that the f-beta values of the Decision tree are low, and there is

high variance in the accuracy score and recall score of the decision tree classifier. XG Boost proved to be the best in correctly classifying the positive samples by showing a high average recall score of 84.2%. Though the variability of XG Boost is slightly high compared to the SVC classifier, its performance in correctly predicting the positive class qualifies XG Boost as the best classifier.

Table 18: Results showing the performance of the 5-fold testing dataset

Description	Decision Tree			SVC			XG Boost		
	Accuracy	Recall	Fbeta	Accuracy	Recall	Fbeta	Accuracy	Recall	Fbeta
Dataset1	0.93	0.61	0.43	0.97	0.77	0.65	0.95	0.81	0.62
Dataset2	0.89	0.8	0.36	0.96	0.75	0.57	0.96	0.75	0.53
Dataset3	0.96	0.96	0.67	0.99	0.91	0.87	0.98	0.96	0.83
Dataset4	0.96	0.76	0.6	0.99	0.84	0.84	0.99	0.92	0.89
Dataset5	0.99	0.73	0.71	0.99	0.81	0.85	0.99	0.77	0.8
Average	0.946	0.772	0.554	0.98	0.816	0.756	0.974	0.842	0.734
Max	0.99	0.96	0.71	0.99	0.91	0.87	0.99	0.96	0.89
Min	0.89	0.61	0.36	0.96	0.75	0.57	0.95	0.75	0.53

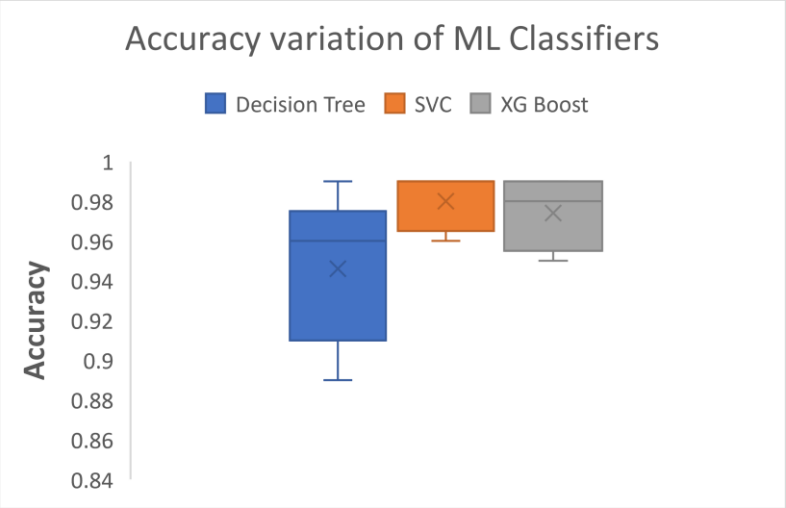


Figure 52: Accuracy variation of ML classifiers on the 5-fold testing data

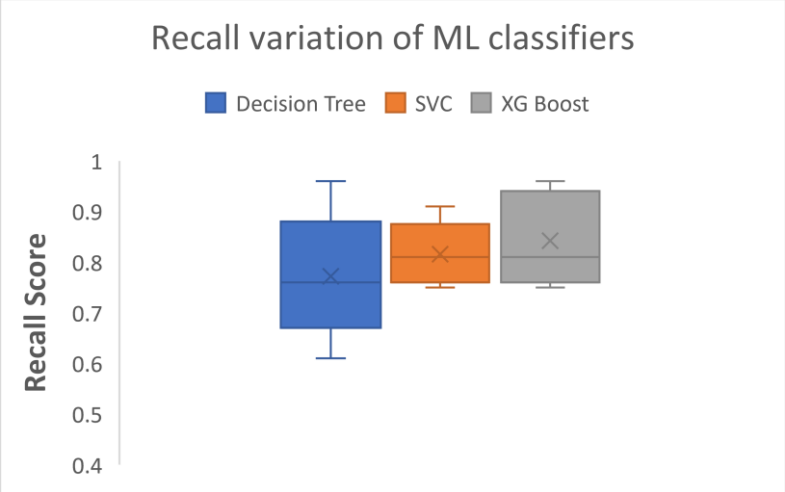


Figure 53:: Recall variation of ML classifiers on the 5-fold testing data

Figure 54: F-beta variation of ML classifiers on the 5-fold testing data

8.5 Overall Results

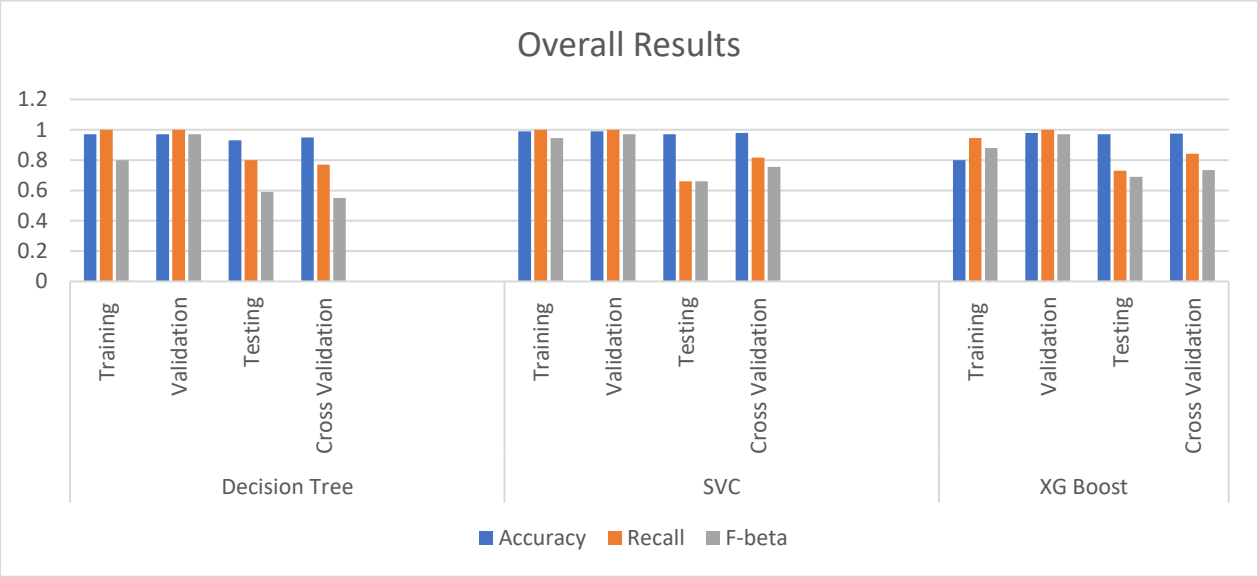


Figure 55: Consolidated Results of all the datasets

8.6 Discussions & Limitations

8.6.1 Discussion

To summarize the study, newspaper articles acquired from the specialized agency have been split into the training and testing dataset. The training data undergo the whole process of duplicate data removal, filtration of data using Guided-LDA, resampling, and ML-based binary classification. Sufficient sensitivity analysis is done in the area of cosine similarity to fix the best hyperparameter. Also, sensitivity analysis is done in the area of GLDA to prove that GLDA can filter the articles having construction-related keywords in it. When going through the process, it is important that the construction accident-related articles are preserved and not lost in the articles. Cosine similarity gave an accuracy of more than 90%, and GLDA gave an accuracy of 100% in preserving the construction accident-related articles.

The ML-based binary classification has undergone several iterations in the hyperparameters to check the robustness of the classification models with respect to the variation in the hyperparameters. KNN and RF algorithms are found to be sensitive to the data when the hyperparameters are varied; hence they are not used further for the classification, and the other models, DT, XG Boost, and SVC, are further assessed. The grid search cross-validation is performed on the ML models to find the best ML hyperparameters, and the best resampling strategy adopted for the model is obtained from this grid search.

Finally, the model is trained using the best parameters. The trained model is evaluated using the accuracy, recall, and f-beta scores for the training, testing, and validation dataset. The model is further validated by applying the full training and testing dataset procedure on the 5-fold cross-validated dataset. The results of the cross-validated dataset seem to be promising, with an average recall of 84%, 81.6%, and 77% on XG Boost, SVC, and Decision Tree, respectively. Overall when are final results are compared, the SVC algorithm is highly insensitive to the outliers. XG Boost algorithm will be the best model for our study as it has the best recall value, i.e., predicting positive articles correctly. It was able to achieve the best recall value in all types of datasets evaluated.

We can classify construction accident-related newspaper articles from a large set of data using this system. Our approach proved to be robust to any data during the cross-validation procedure. The final positively predicted articles from the future data can be manually reviewed and fed into the training model for future datasets that will be sent in as a testing dataset. These reviewed articles can improve the model's decision boundary and improve the predictions in the future. In this way, construction accident-related newspaper articles can be reconciled, so they can serve as a database of construction accident data in India in the future.

The current study also explored the possibilities of extracted of entities such as date, time, and part of the day using named entity recognition and regular expressions. Details are mentioned in Appendix A. The explored rule-based extraction of entities is less accurate and can be improved by extracting entities using semantic meaning rather than rule-based extraction.

8.7 Study Limitations

- The current study uses only information from printed newspaper articles, and other sources such as online magazines, articles, online newsletters, and news blogs were not explored.
- The current study has only considered the English language for the study, and the models are trained only for the English language. When working on multiple languages, this model should be trained on the same.
- The text similarity obtained from the vector space models is only based on the words in the text. If the article describing the same accident is completely using a different vocabulary, it cannot be predicted by cosine similarity.
- The current study has used only ML approaches to perform the binary classification. The other deep learning approaches are yet to be explored.

- The Binary classifier trained is only tested on a dataset for the next three months and cross-validated on the total dataset of 7 months of data. We can't really say that it will function similarly in a future dataset.

CHAPTER 9 CONCLUSION AND FUTURE WORK

9.1 Conclusion

Most of the studies done using the newspaper reports have used manual hand-picked data for doing the analysis, especially in the studies which are doing entity extraction. Very few works of literature mention the automated extraction of data. There were applied frameworks available that can only be applied to datasets of small size (Feng & Chen, 2021). There is an overall scarcity of literature focussing on developing end-to-end automated approaches for extracting articles suitable for attaining Spatio-temporal statistical trends, which can be essential to guide countrywide safety management programs.

The current study contributes to reducing the scarcity of the literature focussing on end-to-end automated approaches for extracting articles and processing them. The present study bridges the gap between previous studies by showing a comprehensive way to do a complete end-to-end process using the framework. Through this study, an attempt is made to reconcile the unused and scattered data of construction accident news articles in different newspapers, and the potential of newspaper articles has been recognized as an essential source of information. The study has carried out automated processing in duplicate data removal and automated filtering of data through Guided-LDA.

The current study brings a novelty in handling large volumes of data, especially in the handling of news articles. The study proposes a novel two-step approach of using vector space models and Guided LDA. The current study is the first application of this novel two-step approach of Guided LDA and cosine similarity. This approach can remove duplicates and unwanted data and also reduce the dataset by a large amount. Every approach used in the literature tries to apply machine learning models directly to the dataset by following basic pre-processing techniques. This may result in the low performance of the machine learning models, even if they may have methods to handle the outliers in the data. Using this combined approach will also improve the processing time along with the performance.

The study also brings a novelty in the classification of news articles on imbalanced data. There are few articles that classified the news articles on an imbalanced dataset. There was literature that has done classification on imbalanced data in predicting the construction injuries (Tixier et al., 2016). There is no literature applying the classification to an imbalanced dataset related to the construction industry using newspaper articles. The classification of construction-related

news articles from an imbalanced dataset using newspapers is first of its kind to the author's best knowledge.

The human intervention is minimal in the binary classification of data as the model is already trained, and the manual intervention in the future involves only parsing through the positively predicted articles instead of parsing the whole dataset, which is manually intensive and time-consuming. The framework still poses some challenges in the accuracy of positively predicted articles and duplicate removal, which is done only based on the words in the text.

The current study can serve as an essential reference document for research across the globe for utilizing newspaper data through machine learning and data mining approaches.

The study will help extract more construction-accident related articles that could be useful for developing the OHS performance worldwide and, more specifically, for India. Previous knowledge and experience of accidents are highly valuable and could contribute to avoiding similar accidents in the future. Through sufficient research taking place on the OHS statistics, insights can be generated which can lower the accident death rates. The current study attempts to improve the OHS situation in India by analyzing past accidents and preventing workplace accidents.

9.2 Future work

- The current study can be extended further by using advanced deep learning neural network approaches for better classifications.
- The duplicate data removal can be further improved by imparting more metrics for comparing the duplicates, such as data and location. Data and location entities extracted from the articles can be used to check the duplicates in the data.
- The current study can be extended by extracting the correct type of accident, entities of location, time, and the date, and a construction safety database can be established. This data can be used to analyze patterns and identify bottlenecks in primary causes of construction accidents or injuries
- A standard construction accident recording and notification system in India can be developed by building this type of database, and adequate research can be done to improve construction safety statistics in India

REFERENCES

Abid, A., Ali, W., Farooq, M. S., Farooq, U., Khan, N. S., & Abid, K. (2020). Semi-automatic classification and duplicate detection from human loss news corpus. *IEEE Access*, *8*, 97737–97747. https://doi.org/10.1109/ACCESS.2020.2995789

Ahadh, A., Binish, G. V., & Srinivasan, R. (2021). Text mining of accident reports using semi-supervised keyword extraction and topic modeling. *Process Safety and Environmental Protection*, *155*, 455–465. https://doi.org/10.1016/j.psep.2021.09.022

Al Qadi, L., El Rifai, H., Obaid, S., & Elnagar, A. (2019). *Arabic Text Classification of News Articles Using Classical Supervised Classifiers*. https://doi.org/10.1109/ICTCS.2019.8923073

Baek, S., Jung, W., & Han, S. H. (2021). A critical review of text-based research in construction: Data source, analysis method, and implications. *Automation in Construction*, *132*, 103915. https://doi.org/https://doi.org/10.1016/j.autcon.2021.103915

Baker, H., Hallowell, M. R., & Tixier, A. J. P. (2020). Automatically learning construction injury precursors from text. *Automation in Construction*, *118*(June). https://doi.org/10.1016/j.autcon.2020.103145

Baraniak, K., & Sydow, M. (2018). News articles similarity for automatic media bias detection in Polish news portals. *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems, FedCSIS 2018*, *15*, 21–24. https://doi.org/10.15439/2018F359

Barberá, P., Boydstun, A. E., Linn, S., McMahon, R., & Nagler, J. (2021). Automated Text Classification of News Articles: A Practical Guide. *Political Analysis*, *29*(1), 19–42. https://doi.org/10.1017/pan.2020.8

Business and Human Rights Resource Centre. (2017). *India: British Safety Council report says 48,000 die yearly due to occupational accidents*. https://www.business-humanrights.org/en/latest-news/india-british-safety-council-report-says-48000-die-yearly-due-to-occupational-accidents/

Chakravorty, S., Daripa, S., Saha, U., Bose, S., Goswami, S., & Mitra, S. (2015). Data mining techniques for analyzing murder related structured and unstructured data. *American Journal Of Advanced Computing*, *II*(2), 47–54.

Chaulagain, B., Bhatt, B., Panday, S. P., Shakya, A., Newar, D. K. P., & Pandey, R. K. (2019). Casualty information extraction and analysis from news. *Proceedings of the International ISCRAM Conference*, *2019-May*(May 2019), 1002–1011.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*(Sept. 28), 321–357. https://doi.org/10.1613/jair.953

Chen, C. M., Wen, D. W. M., Fang, J. J., Lai, G. H., & Liu, Y. H. (2019). A Study on Security Trend based on News Analysis. *2019 IEEE 10th International Conference on Awareness Science and Technology, ICAST 2019 - Proceedings*, 8–11. https://doi.org/10.1109/ICAwST.2019.8923373

Chiang, Y.-H., Wong, F. K.-W., & Liang, S. (2018). Fatal Construction Accidents in Hong Kong. *Journal of Construction Engineering and Management*, *144*(3), 04017121.

https://doi.org/10.1061/(asce)co.1943-7862.0001433

Feng, D., & Chen, H. (2021). A small samples training framework for deep Learning-based automatic information extraction: Case study of construction accident news reports analysis. *Advanced Engineering Informatics*, *47*(January), 101256. https://doi.org/10.1016/j.aei.2021.101256

Gibson, J., Wellner, B., & Lubar, S. (2008). *Identification of Duplicate News Stories in Web Pages*. *4*, 187. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Identification+of+Duplicate+News+Stories+in+Web+Pages#0

Goh, Y. M., & Ubeynarayana, C. U. (2017). Construction accident narrative classi fi cation : An evaluation of text mining techniques. *Accident Analysis and Prevention*, *108*(September), 122–130. https://doi.org/10.1016/j.aap.2017.08.026

Han, H., Wang, W.-Y., & Mao, B.-H. (2005). *Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning BT - Advances in Intelligent Computing* (D.-S. Huang, X.-P. Zhang, & G.-B. Huang (Eds.); pp. 878–887). Springer Berlin Heidelberg.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

Jagarlamudi, J., & Daum, H. (2012). *Incorporating Lexical Priors into Topic Models*. 204–213.

Kaur, H., Pannu, H. S., & Malhi, A. K. (2019). A systematic review on imbalanced data challenges in machine learning: Applications and solutions. *ACM Computing Surveys*, *52*(4). https://doi.org/10.1145/3343440

Kedia, J., Vurukuti, T., Bugalia, N., & Mahalingam, A. (2021). Classification of safety observation reports from a construction site: An evaluation of text mining approaches. *PMI Research & Academic Virtual Conference 2021*, *March*, 50–66.

Kim, J., Youm, S., Shan, Y., & Kim, J. (2021). *Analysis of Fire Accident Factors on Construction Sites Using Web Crawling and Deep Learning Approach*. 1–16.

Koc, K., & Gurgun, A. P. (2021). Machine learning applications in construction safety literature. *Proceedings of International Structural Engineering and Construction*, *8*(1), CSA-05-1-CSA-05-6. https://doi.org/10.14455/ISEC.2021.8(1).CSA-05

Kulkarni, G. K. (2007). Construction industry: More needs to be done. *Indian Journal of Occupational and Environmental Medicine*, *11*(1), 1–2. https://doi.org/10.4103/0019-5278.32455

Kusumaningrum, R., Wiedjayanto, M. I. A., Adhy, S., & Suryono. (2016). Classification of Indonesian news articles based on Latent Dirichlet Allocation. *2016 International Conference on Data and Software Engineering (ICoDSE)*, 1–5. https://doi.org/10.1109/ICODSE.2016.7936106

Lemaire, A., Toubia, O., & Iyengar, G. (2019). *Extracting Features of Entertainment Products : A Guided Latent Dirichlet Allocation Approach Informed by the Psychology of Media Consumption*. *56*(1), 18–36. https://doi.org/10.1177/0022243718820559

Li, C., Zhan, G., & Li, Z. (2018). News Text Classification Based on Improved Bi-LSTM-CNN. *Proceedings - 9th International Conference on Information Technology in Medicine and Education, ITME 2018*, 890–893. https://doi.org/10.1109/ITME.2018.00199

Manu, P., Emuze, F., Saurin, T.A., & Hadikusumo, B. H. W. (2019). *Construction Health and Safety in Developing Countries* (1st ed.). https://doi.org/https://doi.org/10.1201/9780429455377

Mohammed, R., Rawashdeh, J., & Abdullah, M. (2020). Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results. *2020 11th International Conference on Information and Communication Systems, ICICS 2020*, *May*, 243–248. https://doi.org/10.1109/ICICS49469.2020.239556

NDTV. (2017). *Fatal Heights: The Untold Deaths Of India's Construction Workers*. https://www.ndtv.com/india-news/fatal-heights-the-untold-deaths-of-indias-construction-workers-1733974

Nguyen, H. M., Cooper, E. W., & Kamei, K. (2011). Borderline over-sampling for imbalanced data classification. *International Journal of Knowledge Engineering and Soft Data Paradigms*, *3*(1), 4. https://doi.org/10.1504/ijkesdp.2011.039875

Ninan, J. (2021). Construction safety in media: an overview of its interpretation and strategic use. *International Journal of Construction Management*, *0*(0), 1–9. https://doi.org/10.1080/15623599.2021.1946898

Patel, D. A., & Jha, K. N. (2016). An estimate of fatal accidents in Indian construction. *Proceedings of the 32nd Annual ARCOM Conference, ARCOM 2016*, *September*, 539–548.

Rivera, G., Florencia, R., García, V., Ruiz, A., & Sánchez-Solís, J. P. (2020). News classification for identifying traffic incident points in a Spanish-speaking country: A real-world case study of class imbalance learning. *Applied Sciences (Switzerland)*, *10*(18). https://doi.org/10.3390/APP10186253

Robust Semi-supervised Learning for Fake News Detection. (n.d.). https://doi.org/10.13140/RG.2.2.12675.94246

Samanta, S., & Gochhayat, J. (2021). Critique on occupational safety and health in construction sector: An Indian perspective. *Materials Today: Proceedings*, *xxxx*. https://doi.org/10.1016/j.matpr.2021.05.707

Singh, R. (2021). Text Similarity Measures in News Articles by Vector Space Model Using NLP. *Journal of The Institution of Engineers (India): Series B*, *102*(2), 329–338. https://doi.org/10.1007/s40031-020-00501-5

Tixier, A. J. P., Hallowell, M. R., Rajagopalan, B., & Bowman, D. (2016). Application of machine learning to construction injury prediction. *Automation in Construction*, *69*, 102–114. https://doi.org/10.1016/j.autcon.2016.05.016

van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, *109*(2), 373–440. https://doi.org/10.1007/s10994-019-05855-6

Yadav, S. S., Edwards, P., & Porter, J. (2020). Completeness of Ascertainment of Construction Site Injuries Using First Information Reports (FIRs) of Indian Police: Capture-Recapture Study. *Indian Journal of Occupational and Environmental Medicine*, *24*(3), 194–198.

https://doi.org/10.4103/ijoem.IJOEM_202_20

Zhang, F., Fleyeh, H., Wang, X., & Lu, M. (2019). Construction site accident analysis using text mining and natural language processing techniques. *Automation in Construction*, *99*(December 2018), 238–248. https://doi.org/10.1016/j.autcon.2018.12.016

Zhou, S., Kan, P., Huang, Q., & Silbernagel, J. (2021). A guided latent Dirichlet allocation approach to investigate real-time latent topics of Twitter data during Hurricane Laura. *Journal of Information Science*, *August 2020*, 1–15. https://doi.org/10.1177/01655515211007724

Zhu, X., & Ghahramani, Z. (2003). *Learning from Labeled and Unlabeled Data with Label Propagation*.

Zou, Y., Kiviniemi, A., & Jones, S. W. (2017). Retrieving similar cases for construction project risk management using Natural Language Processing techniques. *Automation in Construction*, *80*(April), 66–76. https://doi.org/10.1016/j.autcon.2017.04.003

APPENDIX A

Entity Extraction using NER and Regular expressions

Natural Language Processing (NLP) is a field where information is extracted from text data, and one of the sub-tasks of information extraction is the recognition and extraction of named entities. Entities are the most important chunks of a particular sentence or a phrase. Based upon the problem, the names of the people (person, PER), names of organizations (ORG), names of the locations (location, LOC), geographical names (geopolitical entity, GPE), time (TIME), date (DATE), and number (CARDINAL) can be extracted from a text. The examples of commonly used entity types are shown in

Data pre-processing: The data before being fed into the pre-trained libraries is cleaned and pre-processed. The casing is lowered to have uniformity. Symbols and punctuations are removed and replaced with a space. Stop words that are the most frequently occurring words are removed. Data pre-processing is done using the NLTK library and regular expressions. NLTK library is used for tokenization and removing stop-words. Regular expressions are used for cleaning the data.

Table 19. These entities are detected by algorithms which are ensemble models of Rule-based parsing, dictionary lookups, parts of speech (POS) tagging, and dependency parsing. The rule-based system may achieve a high degree of precision, but its development process is time-consuming, and rules may change from one domain to other. On the other hand, machine learning systems require a large amount of annotated documents. In our study, we have only limited data which are related to construction accidents, and hence machine learning-based information extraction may not be feasible for small datasets. Hence the rule-based system is adopted in this study. Solving the problems of entity extraction can be of two types: rule-based and machine learning-based. There are certain well-known libraries, such as Stanford NER, spaCy, NLTK, etc., which are freely available and pre-trained on millions of data. Since we do not have enough data for training, the development of specific domain corpora is difficult, and hence we use these pre-trained libraries. Stanford NER is a tool for named entity recognition developed by the Stanford NLP group. Natural Language Toolkit (NLTK) is a library mainly used for symbol and statistical processing of NLP implemented in a python programming language. It is vastly used for text processing tasks such as tokenization, parts-of-speech tagging, etc. SpaCy[1] is a very efficient tool that uses the latest and best algorithms with performance better than other libraries, especially in named entities. A sample showing named entities using SpaCy is shown in Figure 56.

In our study, we followed the following process for extracting the desired entity's location, date, time, and period of the day.

Data pre-processing: The data before being fed into the pre-trained libraries is cleaned and pre-processed. The casing is lowered to have uniformity. Symbols and punctuations are removed and replaced with a space. Stop words that are the most frequently occurring words are removed. Data pre-processing is done using the NLTK library and regular expressions. NLTK library is used for tokenization and removing stop-words. Regular expressions are used for cleaning the data.

Table 19: Commonly used types of named entity

Named Entity Type	Examples
Organization	NHAI, AAI, Microsoft
Person	Manoj
Date	12-12-2022, Monday
Time	afternoon, 3 P.M
Location	Himalayas
GPE	Vizag

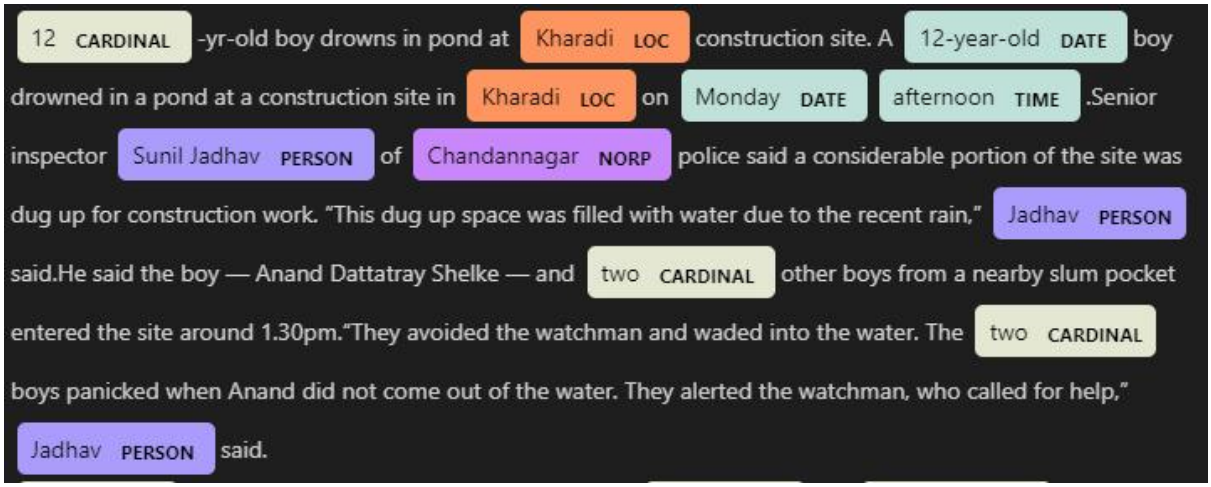


Figure 56: Sample showing named entities using SpaCy (2.0.6)

Parts of speech tagging (POS tagging): POS tagging is done to the corpus to analyze the frequency of each type of part of speech. A sample of 10 titles is taken to understand the frequency, and from Figure 57, it is observed that noun is the most occurring POS, followed by the verb. The occurrence of digits is also high, which raises ambiguities in extracting the number of people killed in an accident.

Named Entity Recognition: Named entity recognition is done using the spaCy tool licensed by the Massachusetts Institute of Technology (MIT), which works under the python programming language. There are four main models used in spaCy: 'en_core_web_sm', 'en_core_web_md', 'en_core_web_lg' and 'en_core_web_trf'. 'en' stands for the language – English, 'core' stands for the type, 'web' stands for written text and last component describes the pre-trained dataset size – small (sm), medium (md), large (lg). The 'trf' refers to transformers pipeline, which is superior compared to the other methods. We used 'en_core_web_trf' in our study because of its high-level pre-training and better performance on our dataset.

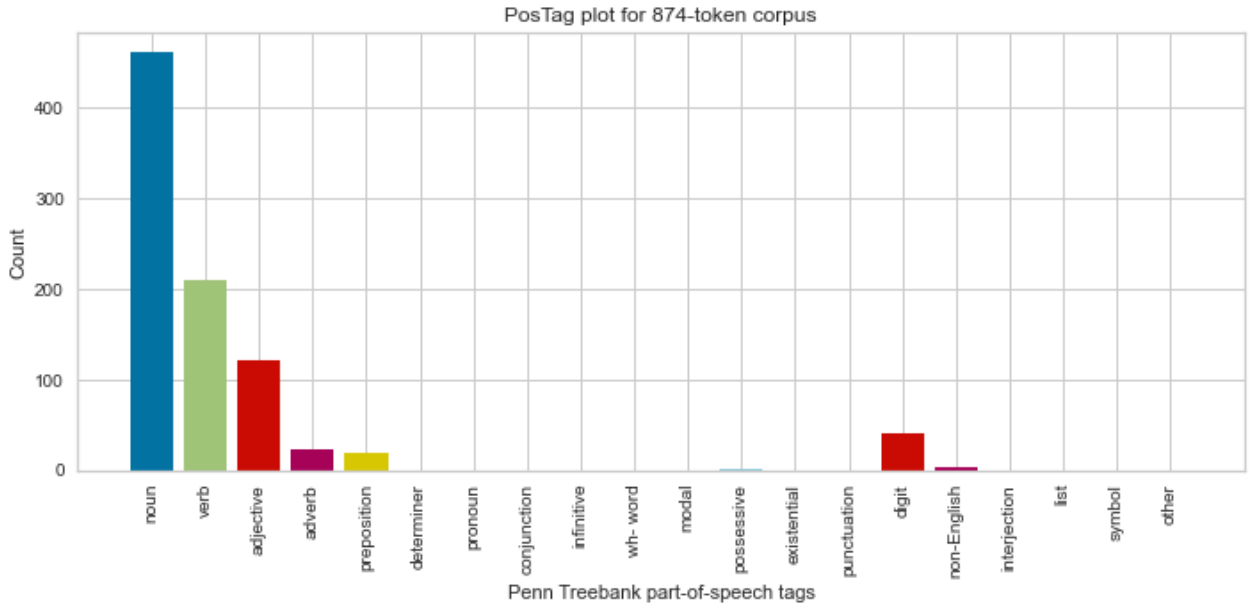


Figure 57: POS tag plot for a sample of data

The newspaper articles are lengthy, with a range of 100-1000 words per article. So, there are multiple occurrences of named entities in every article. To cater to this, we extracted a maximum of 5 entities for each type of entity. This ensures that our required entity, which is time or day, or date, will be present in any of them. So, we extracted five-date entities and five-time entities in every article.

Regular expressions: Further, to filter the desired entities from the extracted entities, we used regular expressions.

Date: Date is generally not mentioned in the text of the newspaper articles. Since most of the articles get published the next day or within the next two days, the day of the week is mentioned. For example, 'The accident happened on Tuesday evening at…..'. This day of the week is extracted using the regular expression pattern – '\w+day'. We have the details of the

published date of the news article. So, the day of the week between the accident and the published date is compared to check the difference in the number of days between the published date and the accident. In this way, the accident date is approximated.

For example, the news article published date is 7th August 2021and the accident that happened on Tuesday is the information we have. Then, the published date is converted into the day of the week using the excel '=TEXT(cell, 'dddd')', which gives the output as Saturday. Each weekday is assigned a number like Monday -1, Tuesday – 2, Wednesday -3, etc. The difference between the weekday is calculated (=6-2=4 here), and the accident date is back-calculated. In the given an example, from the back calculation, the accident date should be 3rd August 2021. Here, we are assuming that the accident happened within a range of 6 days before the article got published, which may not hold true for each and every article, although it is true for most of the articles.

Time: Time is extracted in the entities mostly in two ways: time format and part of the day. A few examples are 'evening', 'around 3:30 pm', ' 11:30 AM', 'night' etc. In most of the articles, either of them is mentioned, so extracting both these formats may be useful in knowing the time of the accident. Time is mostly mentioned in 12 hour format and these are filtered using the following regular expression pattern: '.*pm|.*am|.*PM|.*AM|.*a.m|.*p.m'. In a similar manner, part of the day is extracted from the regular expression pattern: '.*night|.*morning|.*evening|.*noon'. It is essential to go through the Python RegEx module[2] to understand the syntax of the regular expressions,

This whole process is applied to a dataset of 100 manually collected articles from 17 different newspaper publishers. A sample of the article and its results are shown below:

Sample-1: *"Three laborers die after lift collapses at Delhi Development Authority site. Three laborers were killed while another was battling for life after they fell down at a Delhi Development Authority (DDA) construction site in the Dwarka Sector-14 area on Tuesday afternoon. Prima facie, it appears that the brakes of a lift failed, which led to its sudden collapse and the fall of the laborers. Police said a mechanical inspection would be carried out to ascertain the exact cause of the incident. According to police, a call was received around 3 pm about the incident. The caller reported that some laborers fell from a height at a DDA construction site. "Three of them were rescued and rushed to Shree Hospital in Sector-12, Dwarka, while another was rushed to Tarak Hospital at Dwarka Mor. Panna Lal Yadav (50) was declared brought dead at Tarak Hospital, while Basant and Mangal Prasad Singh were*

declared brought dead at Shree Hospital. Another laborer, Surender Rai (48), is under treatment for grievous injuries, and doctors said he was unfit to give a statement. A case under appropriate sections of the Indian Penal Code (IPC) has been registered," said Santosh Meena, deputy commissioner of police, Dwarka. Police said four of the laborers were working at the construction site and took the lift to go downstairs when it suddenly collapsed. So far, it is not clear if the laborers fell out of the lift or collapsed along with the lift. Senior DDA officials stated that a residential housing complex was under construction at the site. "An inquiry committee is being constituted to look into the matter, and appropriate action will be taken based on the findings of the report," said a DDA official."

Publisher: Times of India, **Published date:** 30th June 2021

Final Entities Extracted using spacy and regular expressions:

Time: around 3 pm, Part of day: afternoon, Accident day: Tuesday, Date of accident: 29th June 2021.

There are chances that the extracted entities may not be correct. Here, in this case, it is mentioned in the article that police received a call at around 3 pm, which is extracted as the time of the accident in our machine. The method which we adopted does not consider the semantic meaning of the sentence. Machine learning-based models could be more useful in this study, keeping in mind the length of the articles and the number of entities present in the article. But machine learning models require more labeled datasets to train, which made us limit this area of study to rule-based methods.

The extracted entities are verified manually to check the accuracy of our model. As said, the articles either mentioned the time of the incident or the part of the day, so the total number of extracted entities is only 45 in each of the columns. 83 out of 100 articles mentioned the day of the incident. The spaCy model is very effective as only a few articles that mentioned the entities got undetected (penultimate row in the table). The final accuracies of time, part of the day, and day are 80%, 88.89%, and 75.9%, respectively. The accuracies are low because of a lot of ambiguities in the data which can be rectified only by semantic learning.

Table 20: Results showing the accuracy of extracted entities

	Time	Part of day	Day
Total	45	45	83
Correct	36	40	63
Detected other info	6	4	12

Undetected	3	1	8
Total accuracy	**80.00%**	**88.89%**	**75.90%**