THE UNIVERSITY OF ADELAIDE

PROJECT THESIS

Masters Research Project

Author:
Bhavya PANDYA

Supervisor:
Prof. Nickolas FALKNER

A thesis submitted in fulfillment of the requirements
for the degree of Master of Data Science

in the

Department of Engineer Computer Mathematics and Sciences

June 18, 2021

Declaration of Authorship

I, Bhavya PANDYA, declare that this thesis titled, "Masters Research Project" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Bhavya Kavit Pandya

Date: 18 - June - 2021

"Today a story is not told it's sold."

Amit Abraham

THE UNIVERSITY OF ADELAIDE

Abstract

Data Science
Department of Engineer Computer Mathematics and Sciences

Master of Data Science

Masters Research Project

by Bhavya PANDYA

Consumption of news in text format is not a new phenomenon, but the medium of consumption has changed and transformed at a lightning speed. With growing acknowledgement from the social media and digital content consumers about them falling into a loop of an endless 'Digital isolated bubble', there may not be a direct solution of all the recommendation algorithms on these platforms and the targeted advertisements, but the tone analysis of the news and the comparison between the news article and the headline will allow the public to be more conscious about the type of content they are consuming and the whether is the news media accurately representing the actual content of the article.

With the shift towards the digital platforms, there is much more of a scope to store, model and classify the data to achieve the required results. With the data in text format, we need to work on Natural Language Processing techniques in order to make the Computer understand and map the patterns. We can use text classification techniques either by using heuristics, machine learning or Deep Learning methods.

Furthermore, by implementing the various Machine Learning and Deep Learning techniques to determine the sentiment behind the news articles and headlines, we aim to group them bu category in order to extract meaningful insights of the patterns that correlate the news article, news headline, news category, news publishing date with the overall sentiment that the author wanted to convey while exploring the different learning techniques used for sentiment analysis applications.

Acknowledgements

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would first like to thank my supervisor, Professor Nickolas Falkner, whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.I want to thank you for your patient support and for all of the opportunities I was given to further my research.

I would also like to thank my tutors for their valuable guidance throughout my studies. You provided me with the tools that I needed to choose the right direction and successfully complete my dissertation.

In addition, I would like to thank my parents for their wise counsel and sympathetic ear. You are always there for me...

Contents

List of Figures

List of Abbreviations

ML	**Machine Learning**
NLP	**Natural Language Programming**
SVM	**Support Vector Machine**
RNN	**Recurrent Neural Networks**
CNN	**Convolution Neural Networks**
LSTM	**Long Short Term Memory**
BERT	**Bi-Directional Encoded Representations *from* Transformers**
NLTK	**Natural Language Toolkit**
CSV	**Comma Separated Values**
URL	**Uniform Resource Locator**
TFIDF	**Term Frequency - Inverse Document Frequency**

Chapter 1

INTRODUCTION

1.1 Overview

With the increasing digital presence of large news corporations and even the local news outlets on all the different social media platforms as well as websites, we have seen a shift of public preference from the print and broadcast mediums of news consumption to a digital based news consumption, mainly through three platforms - Social Media applications, News Aggregator applications and websites.

The common factor for all the three platforms is that they are interlinked and that means that readers are only able to view a headline on their respective social media or news aggregator platform. This means there is an added incentive for the news agencies both local and large corporations to have headlines which entice the readers to actually engage the actual full article.

Another factor to take into account is the emergence of digital bubbles, meaning that people who follow one type of news are then further subject to similar recommendations through targeted advertisements and can find themselves in the midst of a ' digital isolated bubble' where they are recommended the same type of content continuously.

We aim to perform Text based sentiment analysis of NEWS data, by comparing different models to review the sentiment scores and determine if some aspects of the NEWS are more polarised than others.

Sentiment Analysis is defined as the process of computationally identifying and categorizing opinions from a text data and determining whether the writer's attitude towards a particular topic is either positive, negative or neutral. In this paper, we want to use sentiment analysis of news articles to determine whether if the tone of the news article actually reflected in the news headline and whether we can determine a pattern based on the data collected from news articles over a time period and use the patterns obtain to correctly predict the correlation between the news category, the sentiment scores and the news articles.

1.2 Research Question

1. To use sentiment analysis to determine the sentiment score for News Articles and News Headlines.

2. To find patterns in NEWS data collected over a period of time, to determine the weight of the features in order to predict the correlation between the different features of the News Data to answer the following questions :
   (a) Are headlines more polarised than the article?
   (b) Are NEWS articles in certain categories more polarised?
   (c) Does the polarity of NEWS data depend on events?
3. To compare the sentiment scores generated using different Machine Learning and Deep learning models.

1.3 Motivation

News articles are concerned with real world events which compromise a layer of emotion - good, bad or neutral. Now with advancements in the digital technologies and onus being put on digital creators and publications, the amount of data to be collected and processed is ever increasing and we need sophisticated machine learning and deep learning techniques in order to classify these articles based on the emotions.Sentiment analysis is the use of machine learning techniques for classification of emotions that are present in any form of text data.

With the increase in existence of sheer number of sources from where people opt to get their news and coupled with the ever increasing busy lifestyle people are choosing for themselves, there is much more of a value put on the opinions and reactions based on a certain news article rather than the actual information gathered or the factual analysis of the event that a certain news article may be focusing upon.

When we couple the increased focus on opinions and reactions by different sections of the news cycle, i.e the news publishing websites, social media platforms with the fact that most people don't really take out the time to read full articles, fact-check the sources and the statistics. This means the news consumers are ever so increasingly targeted and baited in exchange for increased engagement on the digital news publishing and sharing platforms.

With the effects of this seen in the increased polarization of society, there is an increasingly growing trend amongst people who lack the ability to debate and relate to opinions which are not congruent to theirs. This is leading to a society where disengagement is the only peaceful measure but in-turn just divides the society in blocks where people inside the box are unable to see events from a different viewpoint, a viewpoint which may challenge their beliefs and position on a certain topic.

This phenomenon has been defined as the Echo Chamber, and the ever increasing proportion of existence of echo chambers among the society is contributed by different factors, firstly the social media recommendation systems play a huge role, wherein these algorithms across platforms recommend and suggest users related content based on the content consumption history and patterns.Secondly, with the general News consumption cycle all geared towards increasing engagement, there is added value to the views and opinions rather than the actual value that is present in understand the subject and the facts related to the subject.

The existence of these Echo Chambers and the related effects can only be neutralised with the understanding of the type of news that one may be consuming on a daily basis and the added focus put on reading news articles on the same topic from

different points of view.

Layered upon the ignorance of humans as a society and the existence of these Echo Chambers, there are the vices of the digital consumption pattern where the people are able to get news hands on, sometimes even as the event is taking place, and thus begins the inevitable race for the publishers to engage people as early as possible and for the consumers to form an opinion and give a reaction as early as possible and this means there are added scenarios wherein whole mobs of people can be left aggravated sometimes even just by reading a headline.

1.4 Project Outline

This research project is geared towards 2 main outcomes, text classification and sentiment analysis. For text classification, we have performed data mining and data cleaning methods along with NLP techniques for cleaning the news text data and make a corpus of word pairs, known as n-grams. To achieve the above aims for the research we need to set the following sets of objectives to get to the desired output.

We will do a web scraping methods and overview to understand how the datasets are curated, how the websites are arranged and how to collect, store, organise and clean the data. Secondly, we have to select from the different dataset to have data that is normalized and annotated and labelled without it causing an effect on the output accuracy. For the event dataset and news dataset standardization, we can make the news and even category features into categorical features and then compare the both.

For text data cleaning we need to perform the different NLP pre - processing basic techniques to remove all the words in the text data that is not determinant of the sentiment of the news. We then make the token pairs and store them as vectors to train the model on this data for text classification. After text classification is performed, we can use it to find the patterns, perform sentiment analysis and then use this sentiment analysis output score to find patterns between the features of the NEWS data : Article, Headline, Date of Publish and Category

- Step 1: Web data scraping and collecting data
- Step 2: Data Pre-processing
- Step 3: Text data Pre-processing
- Step 4: Ground Truth Sentiment Labelling
- Step 5: Model Implementation
- Step 6: Model output evaluation and visualisation
- Step 7: Conclusion and future directions

Chapter 2

Related Works Review

2.1 Sentiment Analysis

Sentiment analysis is the use of NLP techniques for determining the emotions of the text data. This finds applications in fields where the language used needs to be labelled for certain categories of sentiment - positive, negative, neutral and this information can further help in determining the overall sentiment the writer or author or commenter wanted to convey. This helps businesses to get feedback about their product or service and understand the needs of the customer. This adds to the businesses abilities to review the large amounts of data they gather through reviews, comments, emails, messages and then structure the data for a deeper understanding of the overall sentiment pattern of the consumer. Where there are many ways to implement sentiment analysis, most of the process methods are constructed based on the output requirement such as requirement of fine grained analysis, subjectivity analysis, tone analysis and more.

Sentiment analysis problems on a machine level are perceived more like any classification problem, wherein features are extracted from the text data and fed into the classifier system, which simultaneously is fed pre-labelled class tags. The feature extraction is the process here of conversion of the text data into a numerical form of data. The techniques used can range from a simple bag of words technique where each word occurrence frequency is counted for and made up into a matrix to the n-grams technique wherein, the word pair combinations are taken into account before counting their occurrence frequency, these word pairs can be n words long. Other feature extraction methods include word embeddings, which not only determine the occurrence frequency of words or word pairs, but also establish a relationship between the words. These features are used as input to the classification model pipeline for statistical modelling.

2.2 Natural Language Processing

2.2.1 Overview

Using the manual techniques for determining the sentiments of each article is not scalable and hence we need to apply Natural Language Processing Techniques for sentiment analysis purposes.

NLP is a collection of techniques which allow the machine to process natural languages such as audio, text. With these natural languages being a vital model of communication for conveying of thoughts, ideas and emotions, there lies a vast amount of data for machines to work on and extract meaningful patterns and reach results

which via manual computation can be a very time consuming task.

Use of natural language for machine learning applications has proven a difficult task due to the lack of coherent rules present among different languages that people choose to communicate in. Thus NLP is based on the fundamentals of linguistics i.e rules of language but with advancement in new technologies, this has been worked upon to expand for various applications. Some of the more common applications of NLP include : Sentiment Analysis, Text Classification, Auto-Correction, Machine Translation, Text Summarization and many more.

Hence we can conclude that today's NLP applications for machine learning have come a long way from the initial attempts of linguistics analysis, computational linguistics and statistical natural language processing. Modern Natural language processing can be considered more as an agglomeration of linguistics and statistical language processing, with it retaining the vast linguistic knowledge, adding the computability of large amounts of data and the ability to perform complex statistical operations on the resultant data to compute necessary inferences.

Applied NLP allows use of many third-party library packages which have been carefully curated for use on text data eg. nltk. Moving ahead NLP has incorporated the deep learning techniques in methods for analysis of text data in order to obtain meaningful inferences as well as maintain an end-to-end system.

2.2.2 Architecture

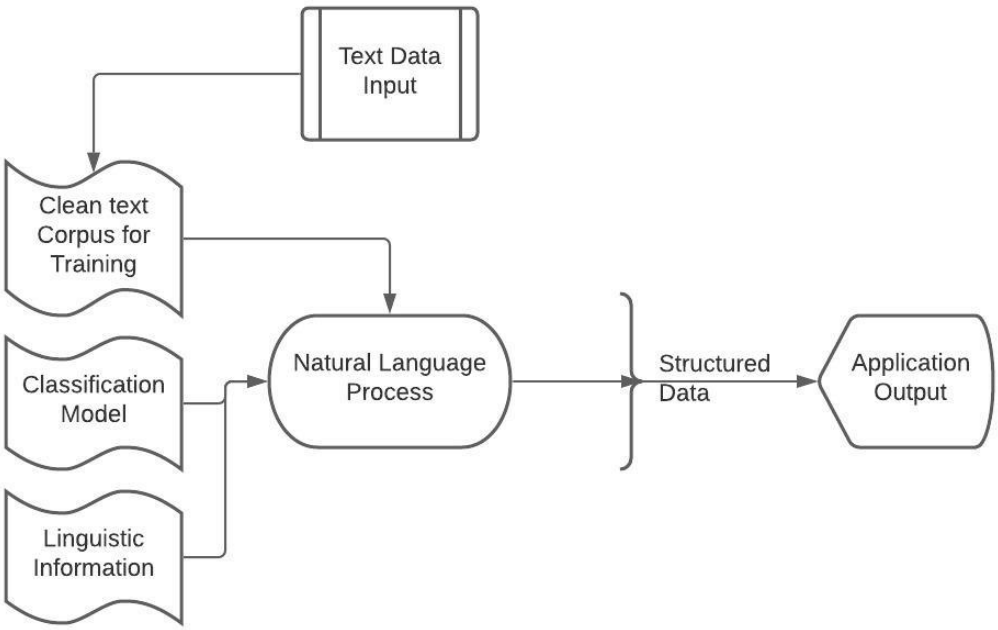

FIGURE 2.1: NLP Architechture

2.3 Natural Language Processing Techniques

2.3.1 K - Means Clustering

An unsupervised learning technique for classifying data containing n observations into “k” numbers of groups where k<n. Here the value of k is used to determine the number of clusters the data needs to be classified in.

The first step while processing raw data is to randomly select “k” data points which act as our initial clusters. This step is followed by the measurement of the distance of the first data point from each of the “k” initial clusters. This data point is assigned to the cluster that is nearest, i.e. the cluster which has the lowest distance value.

For data points existing on a single line, i.e 1-D data, distance is calculated, but for data points existing in 2-D, the euclidean distance between points is calculated and this distance is equal to the pythagorean distance between the two points and the centre of the cluster.

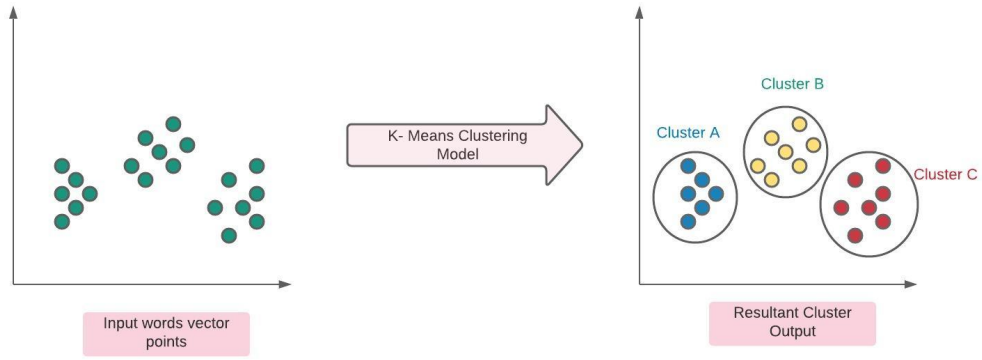

FIGURE 2.2: K - Means Clustering

Based on the different points of reference for distance calculation, the linkages can be classified into 4 types : Single Linkage, Complete Linkage, Average Linkage and Average group Linkage.

Similarly, the process is to be repeated for all the data points and is to be assigned to the relative closest initial cluster. The next step is the calculation of the mean distance of all the data points in each cluster. Now to verify the clustering map output of the given iteration, calculation of the variation inside of each cluster is essential. This step is required in order to remove any skew in the random selection of the initial clusters. Further, more such iterations of clustering are observed for different data points as initial clusters and each is measured for the variation inside the initial clusters.

The model checks for the most balanced variation values for each clustering iteration for “k” number of iterations and then selects the best clusters. This value of k can be checked for until the variation in each clustering is equal to zero, meaning that there are as many clusters as there are data points, but for more efficient results,

we can observe from the clustering vs reduction in variation from the Elbow Plot, wherein we can estimate the point after which the reduction of variation is minimal. This value of k will give the most efficient output for the k-means clusters.

2.3.2 Random Forest Classification

To serve the purpose of classification of data into a certain number of groups, this is a supervised learning technique, which requires pre-labelled data in order to further classify the data into different categories. Random forest classifiers belong to the top of the classifier hierarchy, and are a combination of the different decision tree classifier output. Thus, decision trees function as the building block of the random forest classifier and these individual decision trees combine as an ensemble to provide the required output.

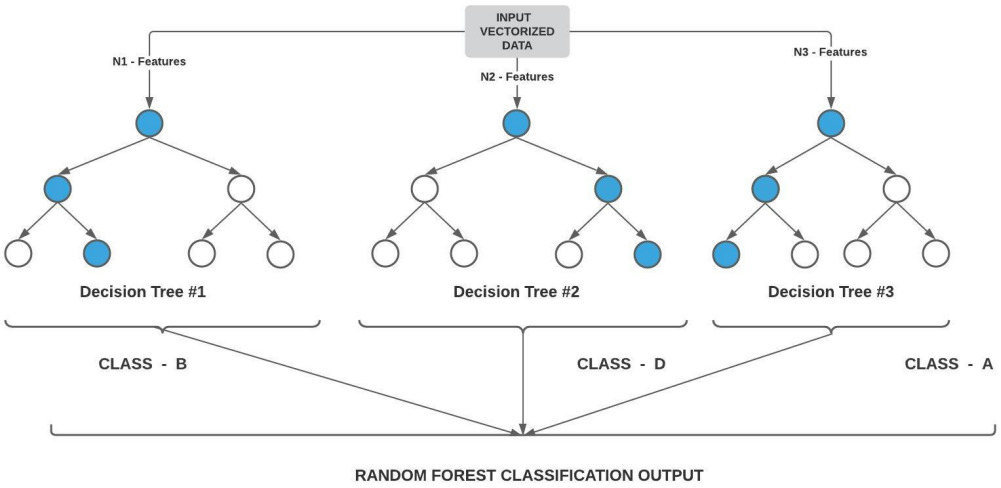

FIGURE 2.3: Random Forest Classification

The concept of combining the output of these relatively uncorrelated decision tree outputs is to get an aggregate bagging output and increase the overall accuracy and stability of the resultant prediction. Another advantage that is observed with random forest classification application is the ability of the algorithm to introduce randomness in the feature selection, wherein it doesn't just select the best features but instead searches for and chooses the best features from the randomly chosen subset of features. These random subsets are used for node splitting and can be further improved by implementing random thresholding.

Random forests also mostly allow the problem of overfitting, especially when a large number of features are to be taken into account. As random forests allow easy methods of feature selection, based on their importance and then the subset of these features is used as input to the multiple uncorrelated decision trees, thus, the model is able to eliminate overfitting to a larger extent.

2.3.3 LSTM [Long Short Term Memory]

Long - Short Term Memory is a type of supervised deep learning technique that uses Recurrent Neural Networks (RNNs) to establish long term token word dependencies. For applications wherein the input data is sequential or time-series data, then these models can provide an added advantage wherein the previous state is stored and used for text classification purpose.

These eliminate the problem in RNNs models, which can only hold short term memory and can thus, possess problems to store and use information from previous states to the new state. This is due to the vanishing gradient problem, wherein the gradient i.e values which update the weights in the neural networks are diminished as the sequences move ahead with time, and when the gradient value is extremely low, the model doesn't contribute anything new to the learning process.

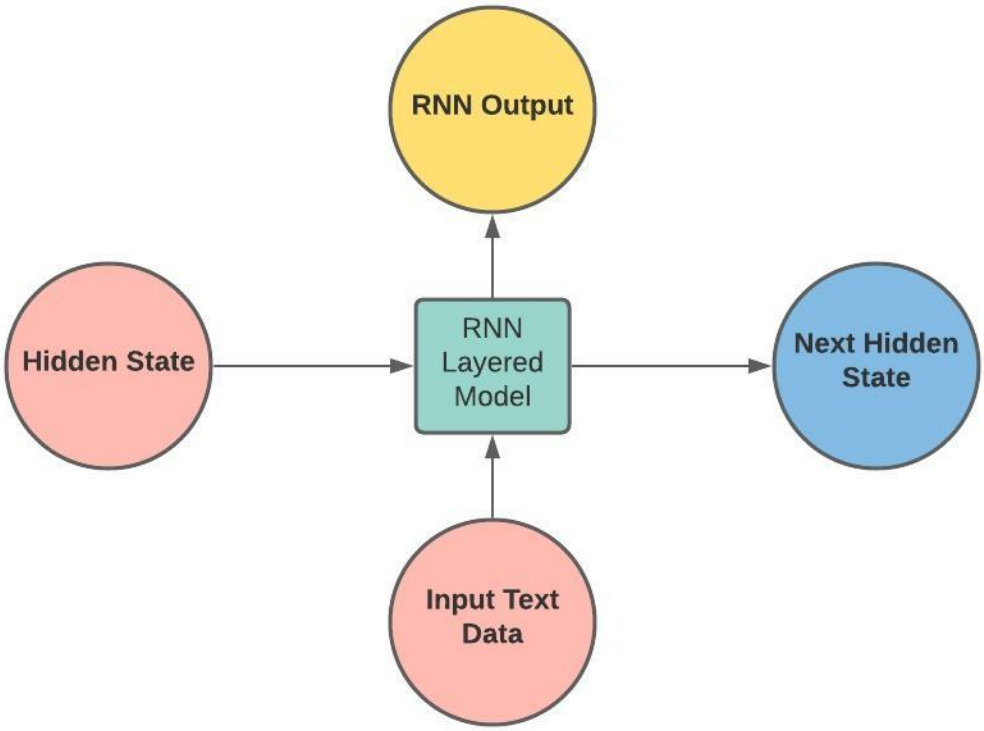

FIGURE 2.4: Recurrent Neural Network

LSTMs were introduced in order to cipher through the data, in order to filter the data which is necessary to store and neglect the rest of the data. These LSTMs, act as gates, where, the flow is the same as that of the RNNs but additionally allow ease of application and tuning, as they have three basic layers : Input layer, Hidden Layer and Output Layer and the hidden layers is the layer which consists of the different emory cells and the gate units which allow these recurrent networks to keep or neglect data as per application requirement.

To prevent overfitting with the LSTM models, there needs to be an accompanying

dropout layer, which helps reduce the sensitivity of the model learning process to individual learning gates of the hidden layer in the LSTM network.

Also, an activation layer needs to be added after the output is generated from the LSTM network, in order to help in the interpretation of the output values as generated in numerical form. The activation function depends on the type of classification output needed, i.e in case of the binary classification, it is efficient to use the softmax activation function. The best method to choose the activation function is to test the various configurations and compare the resultant output.

2.3.4 BERT [Bi-directional Encoder Representations from Transformers]

These are state of the art ML algorithms, which allow us to perform NLP tasks.As the term suggests these collections of deep learning models, which are bi-directional in nature i.e. it can learn textual information from left to right and right to left.

This is a pre-trained on words chosen from the Bookscorpus and English Wikipedia corpus. Pretrained is based on contextualized word embeddings from these corpuses. These transformers are better at handling long term dependencies of the words as compared to the word embeddings. There are two different pre-trained models that can be accessed using Transformers - BERT base model and BERT large model. The base model allows use of 12 bi-directional heads and the large model allows the use of 24 bi-directional heads.

Transformers are used to detect, split and read entire sequences of tokens from a sentence all at once. This is unlike the LSTM model, where the tokens are read individually and unidirectionally, i.e. either from left to right or right to left. This is known as attention modelling, and can allow the model to get an innate understanding of the relationship between words in a sentence, e.g detect the nouns and the subsequent connecting pronouns. Transformers are transformer encoder stacks that can be pretrained.

To use the BERT model for sentiment analysis applications, we can either use the helpers that are provided by the Transformers library or we can just use the basic BERT model and then build a classifier on top of it.

BERT finds application in many text data based real world applications, such as question - answering, sentence classification, next sentence prediction etc. Some features to be kept in mind while handling BERT models is that it is better to pad input sequences in the right rather than on to the left of the input vector. Secondly, BERT models can be trained as Masked Language Models(MLM) ,which are better at understanding the Natural language and the word dependencies, whereas the CLS(Sentence Level Classification) models are more efficient when used for sentence prediction.

2.4 Evaluation Metrics Calculation

The confusion matrix for each model, for both the different datasets separately, we are able to see the count of the following metrics, namely : Accuracy, Recall, Specificity and Precision.

The four resultant quarters of the confusion matrix are termed as:

- Accuracy = True Positive+True Negative/Total : 1->1+[(-1)->(-1)]/Total
- Precision = True Positive/Total Positive Predictions : 1->1/[1->(-1)]+[1->1]
- Recall = True Positive/Correct Observed Positive : 1->1/[(-1)->1]+[1->1]
- Specificity = True Negative/Total Observed Negative : (-1)->(-1)/[(-1)->(-1)]+[1->(-1)]

Where X -> Y denotes Predicted -> Observed,and

- True Positive : Prediction and Observation values are same and True values
- True Negative : Prediction and Observation values are same and False values
- False Positive : Prediction was True values but Observation was False values
- False Negative : Prediction was False values but Observation was True values

2.5 Literature Overview

News tone analysis has been done previously on financial data to predict the impact of the sentiment of the news article on the variations in the stock prices.The whole system is based on the technique of Natural Language Programming called Sentiment Analysis which provides two types of functions : Opinion Analysis and Opinion forecasting.

According to the design principles of [15], a general workflow of a sentiment analysis pipeline has a workflow which starts with Web Scraping and web crawling techniques to get the text data from the webpage. [20]shows that the data extracted from the web pages has to be further processed before it is modelled using machine learning and deep learning techniques in order to initially create a parsing tree which can then be traversed and processed to convert the raw text into tokens.

Tokens as explained by [2]is the division of the text data into smaller portions depending on the task at hand. Calculating the number of tokens after data cleaning has been done using techniques such as lemmatizations, stemming, removal of stop words and Part of Speech tagging as shown by [13], we find that there is a tendency for the number of tokens to keep on increasing exponentially.

This has been worked upon by [18] by using the process of n-grams and by [2] making use of deep learning techniques to decrease the number of features that are extracted from this text data corpus. Researchers have developed various ways to group tokens whether as unigrams, 2- grams or 3-grams. As shown by ..... that results of performing TF-IDF techniques on the n-grams results in three categories of output essentially : High frequency n-grams, low- frequency n-grams and medium frequency n-grams.

They show how low-frequency n-grams can lead to overfitting of the data and high frequency are essentially anomalies in the grammatical structure, both of which need to be omitted from our model which we are to train in order to get the required accuracy of the result. TF is the technique to determine the frequency of a term t in

a document d. IDF is the technique to determine the log-normalised value of the inverse document frequency i.e. the total number of document / no. of documents in which the term t appears

$$idf(t, D) = log|D|1 + |dD : td| \tag{2.1}$$

TF-IDF is the product of the tf(t,d) and idf(t,D) values, whose result if high signifies that the term has a high frequency in one of the documents and low frequency in other documents. This is implemented using sklearn for feature extraction in text using the TFIDFvectorizer.

Let TF=Term Frequency

$$TF = termfrequencyindocument/totalwordsindocument \tag{2.2}$$

Let IDF=Inverse Document Frequency

$$IDF(t) = log2(totaldocumentsincorpus/documentswithterm) \tag{2.3}$$

Let t=Term

$$tfidf(t, d, D) = tf(t, d)idf(t, D) \tag{2.4}$$

The types of NLP approaches to extract the features and train the model include a dictionary based approach in which the model makes use of the pre-existing libraries such a WordNet in order to form the necessary distinction between the positive and negative emotions behind the text of an article and whether if the article was factual based or opinion based. Secondly, there are approaches which involve various word counting methods as shown by [14] , to then find the resultant patterns.

This is known as the vectorization process and as shown by '[22] this can be done using either Word2vec, GloVe or fastText techniques. For word2vec, two words will have almost the same vectors if they occur in the same way in the english text.That is, more than the relation between the words is the semantic relation between the different words. [26]This is a major part of keyword extraction, such as we can say that both Serie A and Champions League are both related to soccer thus to sports directly.This is how the vectors that are used to train the model for text classification provide an added value to the output and we can categorize then in the same cluster.

These features are then used as input to a classification model which essentially finds the patterns in these features which in textual format can be seen as tags and labels. This as shown by [11] can be done using supervised modelling techniques and as shown by [27] and [23] using unsupervised techniques to find the patterns and group these features.

The types of unsupervised techniques that have been worked upon include Principal Component Analysis, Autoencoder and Boltzmann machine techniques. The

advantages as depicted by [18] are the ability to extract non-linear features, capture non-linear relationships between words and find the meaning of the content behind the words. As determined by [10] the overall accuracy gain for a deep learn- ing model for a smaller dataset is not very significant when com- pared to performing sentiment classification. To give the actual sentiment score and the polarity to any given document in the text data, as shown by [6]. There is a statistics based method which is a knowledge-based technique.

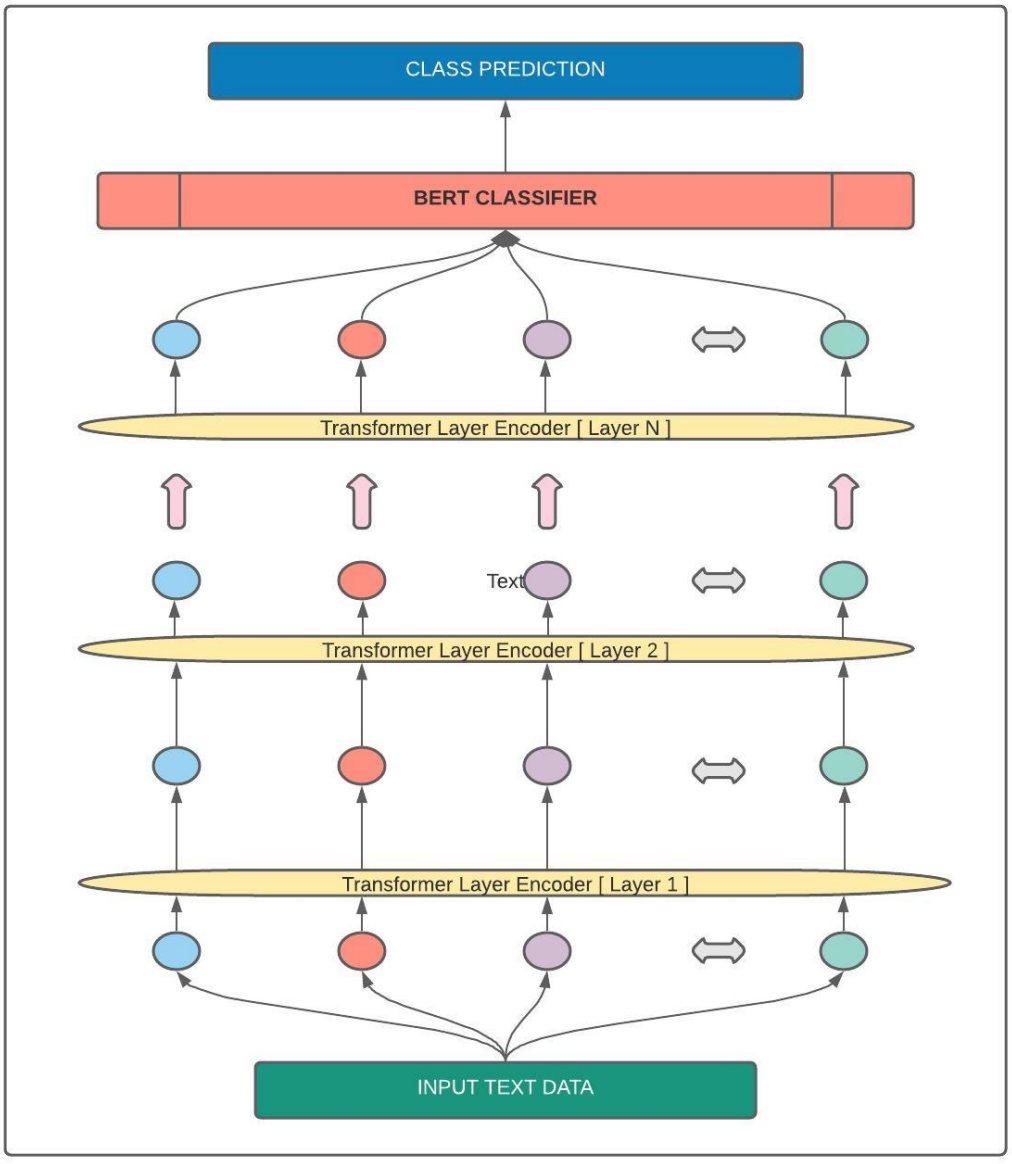

FIGURE 2.5: Bidirectional Encoder Representations from Transformers

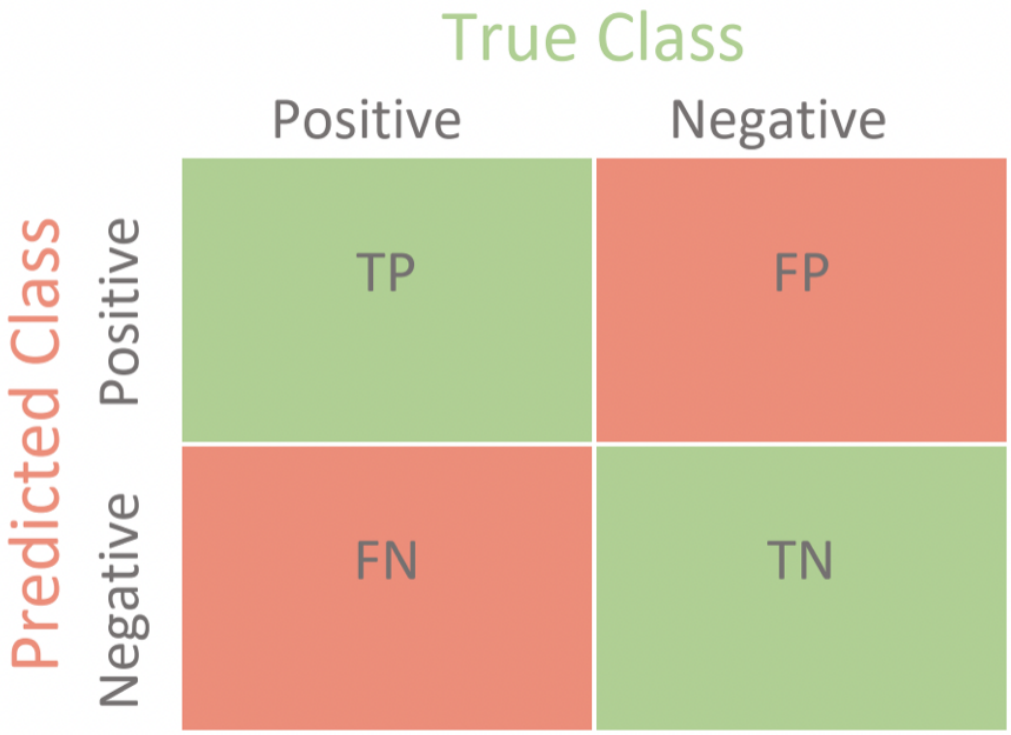

FIGURE 2.6: Confusion Matrix Evaluation

Chapter 3

Methodology Analysis

3.1 Dataset Overview

For this paper, we wanted to mine data from News Publication websites, in large amounts, in order to obtain a better perspective on the distribution of the sentiments across the features.

Data mining for websites is done through the web scraping and data crawling techniques. Our objective was to collect and tabulate the data from the different websites in order to further run text processing and text classification techniques and then label each instance with their subsequent sentiments.

It is essential to establish the exact features that we need to identify and extract data about from the websites. So keeping the aim of the project in mind, we have established that the data collected from the websites needs to have the news article, headline, publishing date, and the category the news was published under.

Now due to limited understanding of the 24 hour news cycle of the publishing websites of the different countries and the limited understanding of the society outlook on the news in general, we have preferred to target Indian News publishing websites which predominantly publish their articles in the English language as we have better tools in hand if and when we encounter peculiarities and obstacles.

3.1.1 Indian Digital News Scenario

The numbers that hit digital platforms across the board overwhelmingly show the eagerness and the shift in preferred mode of content consumption of Indian. Along with this, with the increasing popularity of the english language among the young users, and with 500 million new internet users since 2015, there has been an uptake in the people who prefer to get news from sources online - news publishing websites, social media sharing platforms and news aggregator platforms.

Now 56 per cent of youth below the age of 25 years in India, preferring to obtain news from online sources and 64 per cent of people who access the internet being moderately or highly interested in news content. There are just a significant number of people who are in ways affected by the news that is published on the news sites, which is then subsequently shared on the various social media sharing platforms and news aggregator platforms.

Now looking at the distribution of the readership and the popularity of the various top news publishing websites, we can see that these have engagement in numbers

that are larger than populations of any moderate sized countries.

With digital news publishers like, NDTV, The Times of India, BBC News, Hindustan Times and Republic News being in the top 10 preferred digital news websites, with upwords of 100M readership count statistics half-yearly.

3.1.2 Web Scraping

We can select websites from different spectrums if the political bias scales and thus have a better point of reference for comparison regarding the aims of our project. Having identified our geographical domain and the features we need to grab from the different news websites, we further need to go through the individual websites and see the structure of the websites to set up web scraping functions.

Each website has a different code structure and for each the scraping functions are customized to obtain the relevant content. We have to check for the code structure for where each feature information is present. What we find is that most of the websites have a limit on the amount of requests we can send for data retrieval beyond which they restrict the amount of data that can be scraped per day or even block the ip addresses from any further web scraping. This is circumvented by addressing the issue of data stored in cookies on the website and thus we can obtain the access to retrieve data in large scale to have a significant dataset, upon which we can perform the different machine learning and natural language processing techniques.

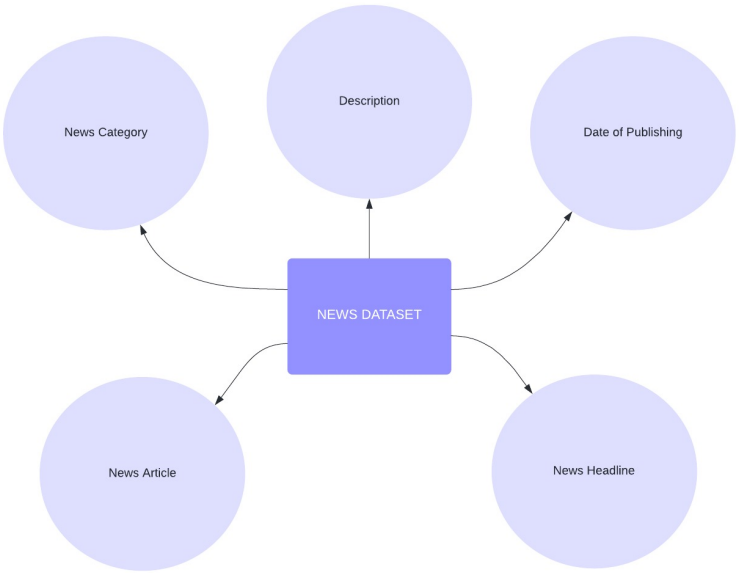

FIGURE 3.1: NEWS DATA feature columns

We have opted to use the spiders like possibility of the scrapers that are available under the *Scrapy framework*. We have opted for Scrapy over other tools such as Beautiful Soup and Flair as Scrapy provided significant advantages with respect to reusability rather than the XML and HTML parser tools offered by *BeautifulSoup* and makes it significantly efficient to scrape data from different News Categories and save it in a file. Additionally, we can take advantage of the Twister functionality provided by Scrapy, through which we can circumvent the blocking issues encountered on the websites and thus, we can use it to smoothly scrape data on a large scale, asynchronously for concurrency.

For data cleaning purposes, we have an added advantage using the scrapy tools as, we can easily choose to eliminate the

Once we have inspected the websites we can lock down our websites, which in our case was two Digital News Websites - NDTV News and Republic News. Navigating all the different urls in each website and further implementing spiders for extracting the relevant data into the feature columns, we are able to obtain data directly into CSV format file.

Once we have the dataset ready, we can read files into our notebooks, using the pandas library in python and perform further exploratory data analysis, which yields the below shown results.

We see that our spider was able to extract data into the following feature columns : Date Published, Article Text, Article Headline, Article Description, News Category and News Sub - Category. We choose to add two independent columns for Category and Subcategory after observations in the data collected to differentiate between the News pertaining the different sub-categories which are further clubbed into world news and India news. From here on, for matters of categories, we will be using the data contained in the sub-category feature column as category information.

3.2 Model Pipeline

Firstly the research aims to establish the polarity of the news articles that appear on the news websites over the world. For this research the project aims make a collection of different articles and make a corpus which can be further analysed to find the patterns in polarity of different news publications, authors and news topics.

For news data collection and analysis, we aim to understand the methods of data mining and text extraction. Furthermore for this we aim to understand how datasets and databases are curated and how to select the datasets for collection news data in terms of text of news article, news headline, news summary and related data such as publication website,date and category of news.

Lastly, the data we collect needs to be ready for classification and modelling for next semester and for that for this research we started with basic NLP techniques implementation on a single article and then moved further towards a more standardized function for multiple articles and features.

3.2.1 Text Data Pre-Processing

So after we have explored our datasets for any missing values, any anomalies and basic features, we can start cleaning the dataset and perform necessary operations firstly for data cleaning and then to make the data in a format which can be trained in a classification model.

There is string data in all features and the category data is converted into categorical data by using labels.This makes the dataset more standardised.We clean the dataset for anomalies in features that give information about the article, e.g. publisher, timestamp and author. For the features such as headlines, summaries and the actual body, we will use NLP techniques for pre-processing this data.

Data fields except the timestamp for news and articles are in String format. For the category field, we can normalize the values in the field in order to make it categorical.

The three different sections being : Headline, Short Summary and Article. In the main article page there are further unwanted parts of the text data that we omit like images and URLs. Text for all purposes is viewed as a sequence of characters which further form words, phrases, sentences and paragraphs.

Our pre-processing step uses the *Natural Language ToolKit [NLTK] library* in python to implement Tokenization, Stemming and Lemmatization techniques. In order to get the implicit meaning behind the different articles, we need to first break down the text into " Tokens" using functions in python for removing the white spaces, punctuations and handle contradictory words i.e. words which have a 'Not', 'Will' attached to it. Even though this allows us to convert the text into different smaller parts, still there will be a large number of repetitive tokens and will underfitting the model. To overcome this problem, we use Stemming and Lemmatization techniques using *ProterStemmer* and *WordNet Lemmatizer* respectively.

By application of lemmatizing after the stemming process where the base word for tokens is taken and suffixes are omitted, it doesn't convert irregular words like 'feet' into its base form and does give non-word tokens for words, e.g 'Wolves' is converted into ' Wolv' which doesn't make much sense for analysis. Thus by using the WordNet dictionary by the techniques of lemmatization, we convert the tokens into lemmas which give us meaningful tokens. Even after the optimization of tokens, there are many words which may result in ambiguity due to either capital words or acronyms present in the article, headline or summary.

Furthermore basic NLP techniques need to apply on the 3 critical features individually to clean the text data for any urls, , stop words, HTML tags and punctuations. Also with the average number of words for articles being close to 400 words, we can use techniques to determine the basic and root form of words in order to restrict the number of tokens from the corpus. In stemming we find the basic form of the word and the stem word may or may not be in the dictionary Whereas, for lemmatization, we convert into the root form of the words which can be found in the dictionary. For tokenization we can use the NLTK python library to choose from *white space tokenizer*, *word punkt tokenizer* and *treebank word tokenizer*.

After this, we get a clean corpus, free from any contractions, any irregular expressions, all converted to their basic root word form and converted into tokens of different n-gram pairs. These tokens are stored as vectors for text classification purposes.These tokens are used as training data for sentiment analysis purposes with the different text classification models.

3.2.2 Feature Engineering

Bag - of - Words Vectorization

These tokens in the corpus still need one last process to be ready for the feature extraction step in the project workflow and that is to convert the text data into vectors in order to form n-grams [collection of n number of words].

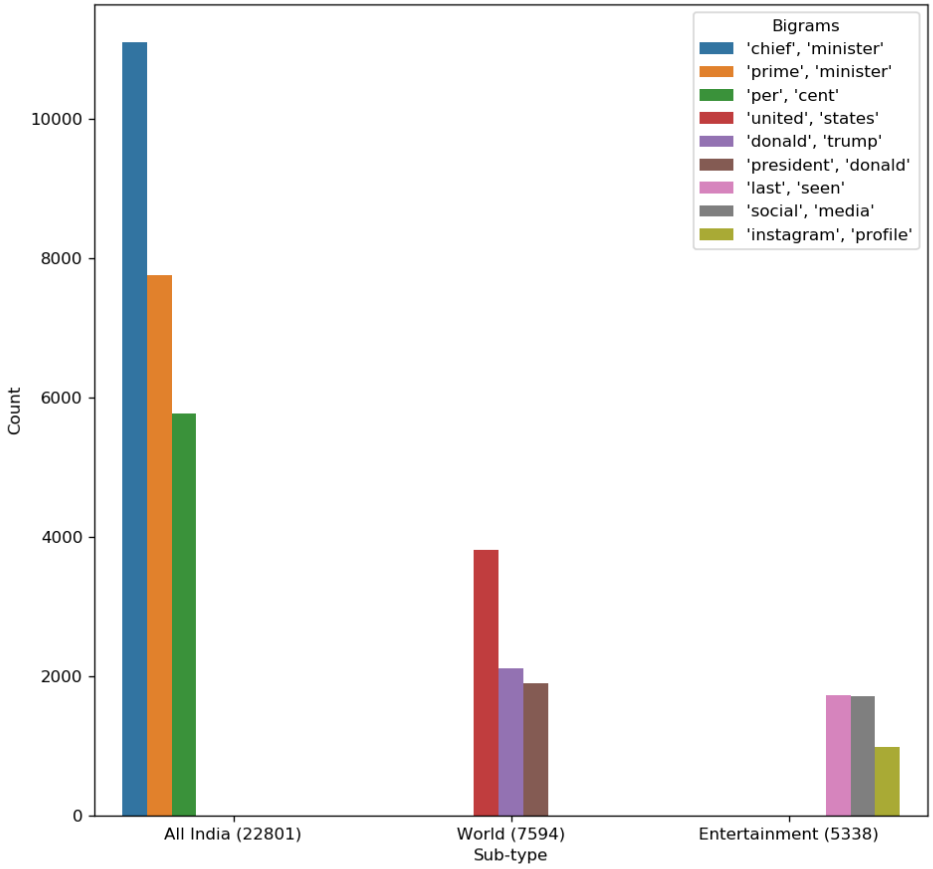

FIGURE 3.2: Bi - Grams [Publication 1]

To find out the most relevant and the key words from the text, defined as per our context. We can leverage the output of this text analysis technique by using NLP tools to

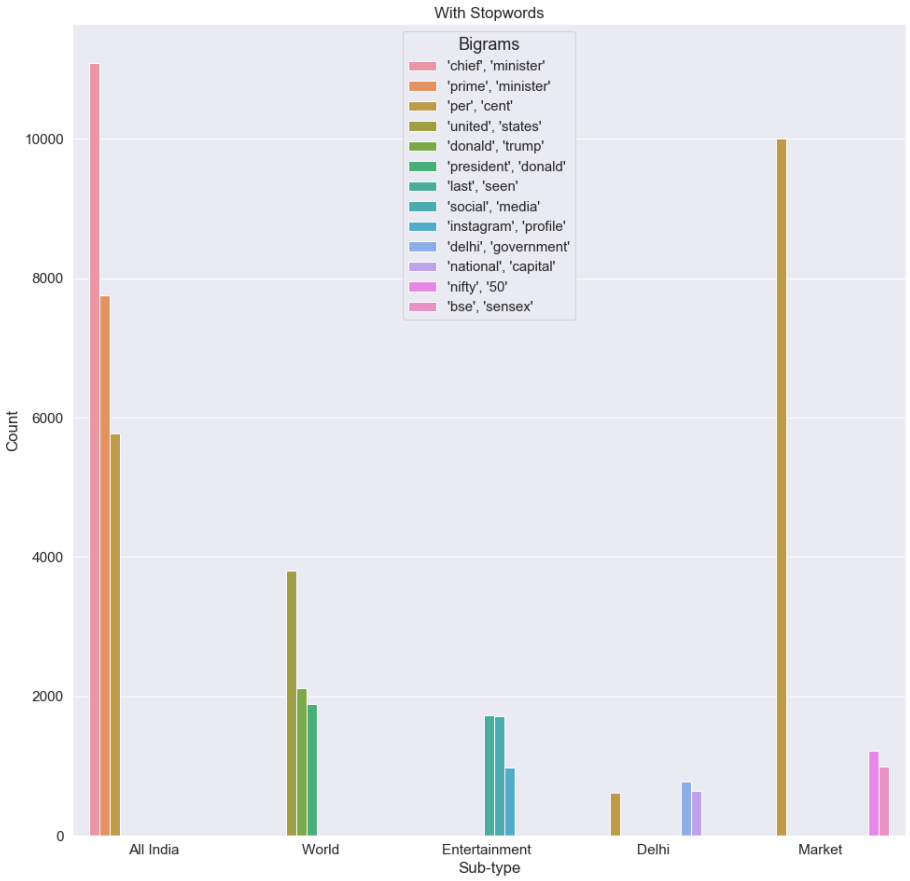

FIGURE 3.3: Bi - Grams [Publication 2]

break down the barrier between human language and machine language.This is visualized in terms of a word cloud. This can allow us to obtain key tags for the event feature in the event dataset and allows to to convert this into a categorical feature which is easier to match with the actual text dataset. For our unstructured data, this is the optimum method to find the key words. By implementing word co-location techniques using n-grams allow the model to count separate words as one.

As we can observe the frequency distribution of the 1-gram, 2- gram and 3-grams to evaluate the vectorization process. The process of word to vector is done for the model to be able to extract features which are used to determine the polarity and subjectivity of the text data. Our model will form n-grams that are token pairs to then be run through layers of convolution filters in order to determine the log-normalised value of their occurrence frequency in the whole collection of articles. The 1-D convolution windows are preferred to the linear Bag of Words technique which when applied to all the documents gives a large number of 2-grams.This number of 2-grams is exponentially increasing as we increase the number of articles analysed. In our approach we work with dense representation of the tokens due to its increased

comparability of vectors which are similar, i.e it makes a recursive filtering of the 2-grams through the convolution filter so as to understand the larger bracket for similar words.e.g cats and dogs and combined to come under a larger bracket of animals.

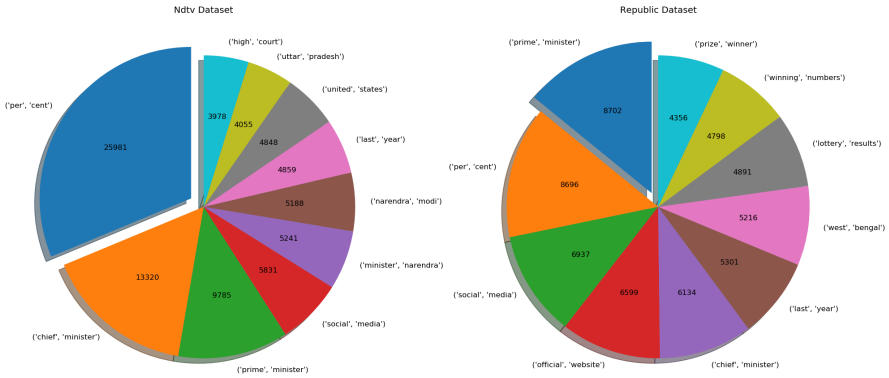

FIGURE 3.4: Pie-Chart of Bi-Grams Distribution for both News Publications

TF-IDF Vectorization

From our text data, we have made two separate dictionaries for Article Text and the Article body which have a collection of all the tokenized words for both the respective feature columns.

Now for these dictionaries of the words, instead of using a simple bag of words method to calculate the frequency of occurrence of the words, we are able to attach the importance of the words that are occurring, with respect to the sentence and the document. This method is called the TF-IDF method.

In this method, we can calculate the frequency of occurrence of the words in a document, adjusted for the frequency of occurrence of the word in the whole document, i.e the more frequently the word occurs in the whole document, it is much less likelier to make any significant different to the calculation of the text classification process and thus mathematically, we are able to give it a lesser score, than the words which can provide significant information for the classification process.

For this we use from *sklearn.feature-extraction.text import TfidfVectorizer*. For the K-means clustering model for text classification, we have used the TFidfVectorizer as the word to vector converter as we can have added advantage as compared to a simple Bag-of-words model, as we can understand the significance of the words with respect to the whole document.

We have opted for TF-IDF over a CountVectorizer as we are able to get a normalised output for the sparse matrix output value and are easily able to convert it into a numpy array, added with the information of the importance of the words in the document and the linguistic similarity of the words in the document, that can be calculated by the tf-idf vectorizer.

Word Embeddings

By establishing more complex features from our model for the input text data, we further implement max pooling over time to reduce the number of features of our model while making sure our model is not over fitting the data. This is done by treating the different documents as a coagulation of the probability distribution of different topics. For word2vec we want to keep the order and the number of words per sentence the same; therefore we will replace these words with a random word 'abc". We will remove frequent words and stopwords since they probably bring little meaning and maybe even create noise when we want to classify later on.

This is a type of vectorization that divides the sentences into meaningful word representations.These allow us to have a distributed distribution, i.e we can have impressive performance results in deep learning methods. We use word embeddings as the input to our LSTM model, which uses improvised neural networks, to make use of the learning capabilities and allow the model to hold the required word relations in the memory for a longer period of time, as compared to a simple Recurrent Neural Network model.

Using word embeddings, we can visualize the individual words, in the form of vectors and then calculate their resultant similarity scores.These words which are mapped as resultant neural networks, are represented as vectors and further measured with respect to a normal vector, thus holding the capacity to clump words which have similar meaning, together and thus allow the classification model input vectors to capture word meaning.We can train the word vector space to optimize the output, by placing the words which have similar meaning, closer to each other.

We use Glove vectors, which are a type of word representations technique that is an extension of the word2vec model, and can use unsupervised learning techniques to learn the similarity between the words and representations in terms of vectors.

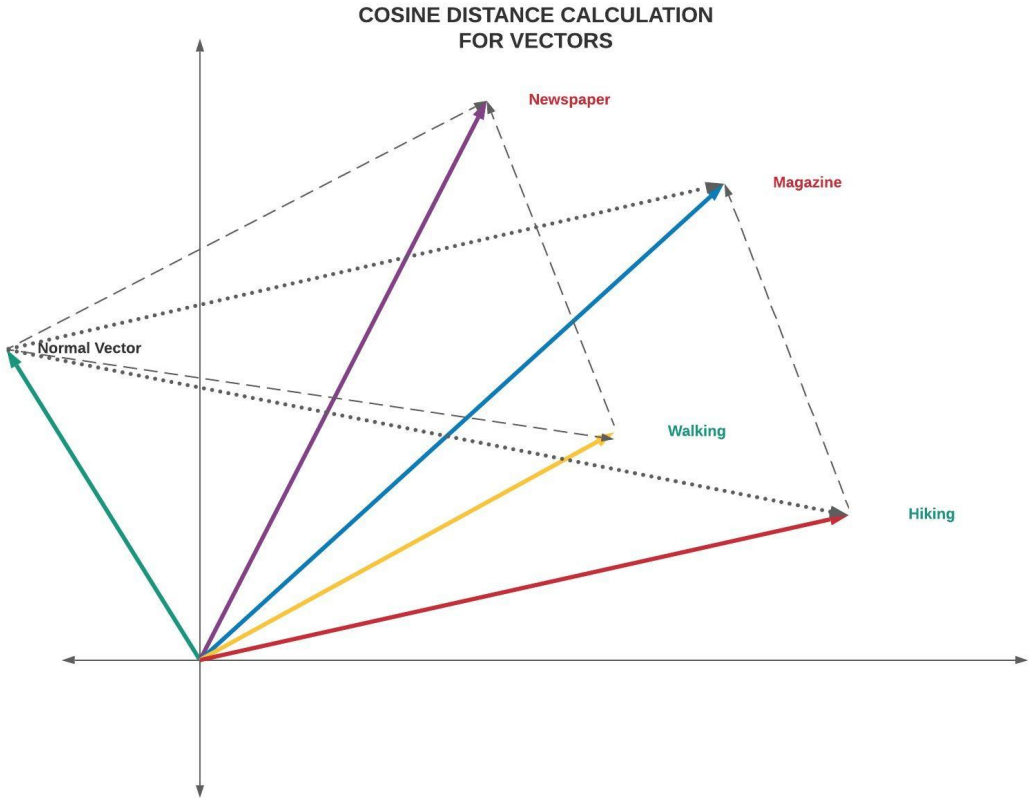

FIGURE 3.5: Cosine Similarity for Vector Distance Calculation

From the input text data word corpus, a word to word co-occurrence is established and then the glove vector can combine the statistics approach to the word based semantic learning and give a vector output. We make use of the pre-trained Glove vectors which are trained on the words collected from the Wikipedia or Gigaword word corpus database.

Now we can use these vectors to measure the similarity between the words and by calculating the relation between the similar words and thus allowing the model to learn the relations between the words. This is achieved by calculating the cosine similarity between the words by calculating the cosine of the angle between the two vectors, using from sklearn.metrics.pairwise import cosine-similarity.

3.2.3 Ground Truth Sentiment Labelling

TextBlob library is used with python to determine two factors from the extracted features : Polarity and Subjectivity of the text data from the article, summary and the headline.

This is applied on all three separately and the results are then compared to give output in terms of varying factors such as News Publication, News Author and News Topic with respect to time. To find polarity we determine the overall emotion of

the text data by comparing our features with pre-existing features which are further customised depending on the topic of the news and to find the subjectivity of the text data, i.e. the level of opinion involved in the representation of the news we determine by the application of weighted encoders on each article on a -1 to +1 scale with -1 being negative, +1 being positive and 0 being neutral.

Aggregate tone of each word in a sentence gives the tone of the sentence and the aggregate tone of the sentence is marked to make the aggregate tone of the paragraph, similarly scaling to give the tone of the article.This is then represented with respect to time and is used to compare the variation in the tone of the news in terms of the polarity over time. We apply news focus techniques in order to give differential weights for recurring topics as a news article will not have the same effect on the readers with each passing day. Application of news focus techniques is needed to determine the varying impact for each article on public opinion and events.

For our text classification model comparison, we have used both, supervised and unsupervised techniques, and as in our scraped dataset, we have no sentiment label tag, we determine the sentiment using the textblob library, in order to use as a reference for the supervised techniques.

For sentiment labelling using textblob, we prefer it over flair as it's more accurate and faster and does give results as we wanted it to. so, we used textblob to get the labels for our news to test the models for accuracy.Although it is not as fast as using spacy library package, but we are still able to get the resultant output, in a better time frame for large dataset, as compared to the time taken by nltk.

We use the library package : TextBlob(text).sentiment.polarity TextBlob can be used to give the polarity of the sentence, i.e. mark the sentence based on the lexicons, in a range of -1 to 1 to mark the sentiment score, where negative score notifies a negative sentiment being conveyed in the sentence and a positive score denotes a positive tone.Now while parsing the sentence, a negation, i.e words which are universally recognised as negation words, eg. not, no turn a polar opposite of the polarity score and hence can be efficient and accurate in the output generation.

3.2.4 Text Classification Model

We have used four different models for comparison of the final sentiment score. For the purpose of this research project we have used 3 different supervised learning techniques and 1 unsupervised learning technique. For the supervised learning classification model, we have options such as Logistic Regression,Random Forest Classifier, Näive Bayes and Linear Support Vector Machine.

As we can view our task as a clustering application, where we need to determine the clusters formed by the vectors in the vector space. We use the unsupervised learning technique : K-Means Clustering as it is widely recognised as the most basic clustering algorithm and with our dataset being highly-dimensional and being quite large, we make the clustering process as simple and clear as possible to increase the efficiency. Unsupervised learning techniques are not very widely used for text classification purposes, although we can use these to determine unknown patterns in the data.

Deep learning techniques try to simulate the human brain style learning processes, into a set of algorithms which contain different tunable parameters and layers which can be modified to suit the application. The two main domains explored for text classification using deep learning models are Convolutional Neural Networks(CNNs) and Recurrent Neural Networks(RNNs). With the ability to build hierarchical systems, which are based upon multiple algorithms, we can set up the classifications models, which can inculcate large amounts of training data, without the limitation of a threshold breach.

Using RNNs we can allow the model to learn over time rather than the learning outcome of CNNs which are based on space, thus allowing us to find meaningful patterns based on long range semantic similarity and dependency of the words.

We use LSTM, which is further an improvement on RNNs and can allow the model to filter and retain important information for a longer time period and thus improving the accuracy, by preserving the dependency information between the different words in a sentence in the text data.

BERT models by design allow us to reach state of the art application results, by just tuning one layer. With BERT models, having shifted the NLP landscape on a large scale with many significant advances in the ease of application. We are able to replace the LSTM based architecture with the transformer encoders and thus achieve unprecedented results particularly for semantic similarity purposes. With the model allowing us to use pre-trained unlabelled data that has been trained on a very large scale, we can use the pre-learned knowledge achieved by this model and can fine-tune to achieve the sentiment scores with high-level of accuracy.

3.3 Classification Models for Sentiment Analysis

The input data we have used is from two News websites, with 50k and 70k elements each. Having applied the text pre-processing steps on the text data on each of the two feature columns, we were able to get standardized text data, which were converted into tokens and subsequently into word embeddings and vector notations, which allow us to use it as mathematical input to the machine learning and deep learning models, which we will be exploring.

3.3.1 K-Means Clustering

In this project we have used k-means clustering to identify the sentiment labels for each of the news articles and the news headlines, on both the datasets individually.

As we know that the k-means clustering is an unsupervised learning technique, we are able to cluster the input data which we have vectorized using the TFIDF vectorizer which allows us to have information on the importance of the words in the document.

We use this vector as input to the clustering algorithm, where we are able to set the value of k-for the number of clusters, which here is 3. We separate the data into 3 clusters to be able to classify the data into the 3 groups : negative, positive and neutral.

Thus, regarding the parameters used for k-means, the number of batches is equal to 3, initializing size =3000, batch size =3000 and the max. Iterations equal to 100000. We have applied this to both the article text data and the article headline data.

Now as we know, that this is an unsupervised technique, i.e there are no sentiment tags or y labels, we are able to form cluster the data by using a process called k-seeds, in which we further adjust the centroids using the cosine distance as metric and then this step is repeated for every calculation of the centroid, that is it is iterated 3 times in the process. This step is performed to increase the accuracy of the clustering model.

We had to optimize the input data to make it more standardized as this is an algorithm that is using the distance between the points and the centroid and thus through more standardization steps through manual regular expression cleaning, we were able to achieve a satisfactory output. Another limitation we had to encounter was the randomized initialization of the centroids in the k-means algorithm and for this we have to identify the global optimum and then allow the model to pick the results based on the distances calculated from that process.

3.3.2 Random Forest Classification Technique

We choose to perform our text classification using Random Forest classifier even though SVM works very efficiently with high-dimensional text data. As support Vector Machines are preferable to use when the problem data is not linearly separable and thus we can take advantage of the non-linear kernel functionality with SVM.

With Random Forest classifiers we are able to handle the combination of high-dimensional data present in a large dataset, and thus be suitable for our application to get efficient and highly accurate data. The main advantage of Random forest classifiers which are essentially a compounded combination of the different decision tree outputs, we are able to design it to perform and learn out of the box and hence enhance the output test accuracy.

For this project we have used a bag-of-words model for the vectorization process on both the feature columns, on both the datasets. As we know, bag-of-words is much simpler model when compared to an advanced vectorization model such as TFIDF but because we are using a much simpler supervised text classification algorithm, the TFIDF output is not of that much significance as it become redundant as we don't need to apply weight reduction techniques on the word vector features.

The parameters we have used are default, that is the no. of estimators we try to keep is as large as possible to avoid the case wherein any important features are left out from the decision tree classification process. Next we use the parameter for splitting the node, and we use the gini over entropy for classification tasks. We get less error with gini splitting mode.Another important parameter is the definition of the depth of the split nodes, and for the project we allow the tree to split until all leaves are pure or until all leaves contain less than minimum split samples.

For evaluation purposes, we have split the data in a 60:40 ratio of Train and Test

data. We use this to check the performance of the model, and then evaluate using the accuracy metric.

3.3.3 LSTM for Text Classification

We now come to the implementation of the deep neural networks for text classification application. As we know that the LSTM models work with high accuracy with word embeddings, we have made use of the pre-trained Glove vector library, which is a library package available in different packages, which contains, pre trained word embedding vectors. With the necessity to use more common formal english language for the news articles, we can get a good word dictionary out of the text data corpus. Word embeddings are one of the most advanced methods for vectorization text data.

We have created a sequential model using keras, which has parameters where we keep the length of the vector to 100 to represent each of the words, and then further use a dropout using the 1-D Spatial dropout which is needed for NLP models for text classification applications.

Furthermore, we add the subsequent layer of the LSTM model and for this layer we make use of 100 memory units. Moving to the output layer, we know the output must correspond to the number of the classes that our data needs to classify into, thus our output layer is able to create 3 different classes.

The model takes, 2-D vector as input,where each layer of the LSTM has the same number of the cells and time steps. The input has time steps equal to 3 and has two input features.Layer 1 outputs 128 separate features from the input data provided, further the layer 2 of LSTM to prepare the output classified data for decoder. The decoder and encoder are opposites of each other. For each layer between the encoder and decoder, we can see 1-D vectors.

Activation function has to be softmax, as we need the model output for multi-class classification. Thus the loss function is categorical cross entropy and the optimizer is adam. -optimizer. The output didn't have an overfitting problem maybe due the large amounts of training features that were present in the article post data, but if we were to make the corpus of the text data alone, but we have created a combined text data corpus, which is made from data both in the article and the headline.

We evaluate this model on the data that was split using train-test-split and then check for the accuracy metric.

3.3.4 BERT for Text Classification

Next for the project we want to design a text classification model, using the BERT transformers, which is a supervised transform based deep learning technique. BERT is a transformer based neural net model which is open source by google. We can fine tune BERT according to any text related problem we want to use it in. Here we have a classification problem so we will use it for classification.

We can use the transformers package in python and use the extensions such as : tensorflow bert model, bert tokenizer and bert configuration. For the tokenization process we are able to use the bert based tokenizer, which allows us to have deeper and intimate knowledge of the different patterns and the relations between the words in the large corpus, and especially for a large corpus. This is due to the large 110 million words upon which the data is trained upon.

For the model training we have used the bert based uncased model, which further has parameters, which allow us to tune the model. In the model, we add dense layers, because the BERT model returns three main aspects : pooled output, encoder output and the sequence output. The pooled output is defined as an representation of the input sequence as a whole, meaning the parameters fed are the model batch size, which allow us to embed the whole text corpus data.

Further, the encoder output functions as the activation triggers for the intermediate layers of the model. The different layers of the BERT model now are, the input layer, the preprocessing layers, which is fetched through keras, the BERT encoder layer, the dropout layer for which the parameters have been manually defined and the dense layers which allow us to set the value of the number of classes we need to classify the data into.

We have defined the loss function and the optimizer functions for fine tuning the model performance, which we have selected based upon which suits our output the best.

We will use AdamW as our optimizer. It is an improved version of the Adam optimizer. We will fine tune the BERT model to work with 2 categories. We add a Dense layer at top of the BERT to and then train it using our data for sentiment analysis.

3.3.5 Evaluation

We have used four different classification techniques for text classification purposes and further compared their respective sentiment scores and checked for the same when compared to the original ground truth sentiment labels that we have tagged using textbolb.

Further having compared their resultant accuracy scores, we are also able to plot the confusion matrix which is visual matrix representation for summarizing the performance of the different machine learning and deep learning techniques.

Dataset	KMEANS	Random Forest	BERT	LSTM
NDTV	61.8	80.1	78.2	82.6
Republic Tv	51.3	81.2	78.6	82.3

FIGURE 3.6: Accuracy Comparison for all models

News Publishing Website 1

For Publisher 1, we can observe that we have a total of 50k observations, i.e 50k rows. From the confusion matrix plot , we are able to observe that the no. of correct predictions are three times as high for Random Forest Model as compared to that of K-Means and as a consequence we can see that the accuracy of Random Forest model is double that of K-means. For Publication-1 we can observe that all the metric scores for the three supervised learning models are very comparable, with the accuracy of LSTM>Random Forest>Bert models.

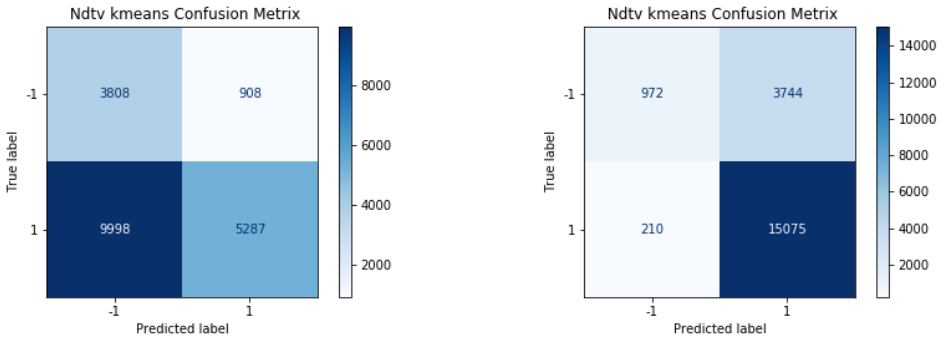

FIGURE 3.7: Confusion Matrix for K-Means Clustering and Random Forest

News Publication 1	True Positive	False Positive	True Negative	False Negative		Accuracy	Precision	Recall	Specificity
K-means Clustering	5287	908	3808	9998		0.4547272636	0.8534301856	0.345894668	0.2758221063
Random Forest Classification	15075	3744	972	210		0.8023098845	0.8010521282	0.9862610402	0.8223350254
BERT	14514	3682	1034	771		0.7773611319	0.7976478347	0.9495583906	0.5728531856
LSTM	13812	1984	2732	1473		0.8271586421	0.8743985819	0.9036310108	0.6497027348

FIGURE 3.8: Evaluation Scores for News Publication 1

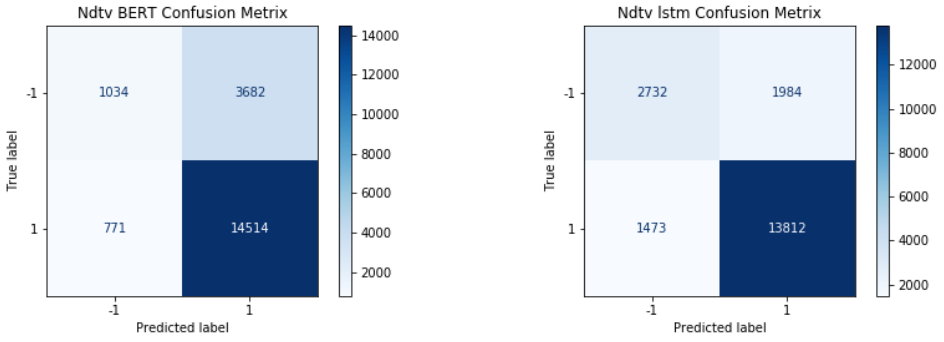

FIGURE 3.9: Confusion Matrix for LSTM and BERT

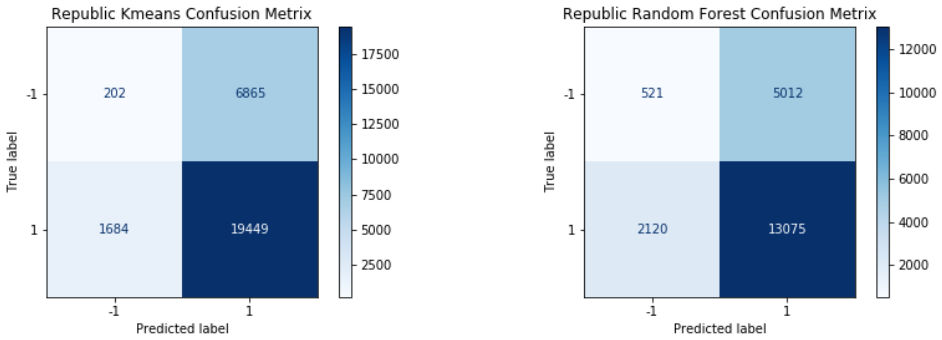

FIGURE 3.10: Confusion Matrix for K-Means Clustering and Random Forest

News Publishing Website 2

For Publisher 2, we can observe that we have a total of 70k observations, i.e 70k rows. From the confusion matrix plot, we are able to observe that the no. of correct predictions are not as diverse as that of the model outputs for publication 1 as a consequence we can observe that the accuracy of all supervised models are more than that of the k-means clustering model which is an unsupervised model. For Publication-2 we can observe that all the metric scores for the three supervised learning models are very comparable, with the accuracy of Bert models>LSTM>Random Forest>. Similarly, we can observe that the precision for BERT>LSTM>Random Forest>K-Means.

News Publication 2	True Positive	False Positive	True Negative	False Negative		Accuracy	Precision	Recall	Specificity
K-means Clustering	13075	5012	202	1684		0.664747409	0.7228948969	0.8859001287	0.1071049841
Random Forest Classification	19449	6865	521	2120		0.6896908997	0.7391122596	0.9017107886	0.1972737599
BERT	20310	4435	250	1605		0.7729323308	0.8207718731	0.9267624914	0.1347708895
LSTM	22942	5057	252	2052		0.7654027654	0.8193864067	0.9179002961	0.109375

FIGURE 3.11: Evaluation Scores for News Publication 2

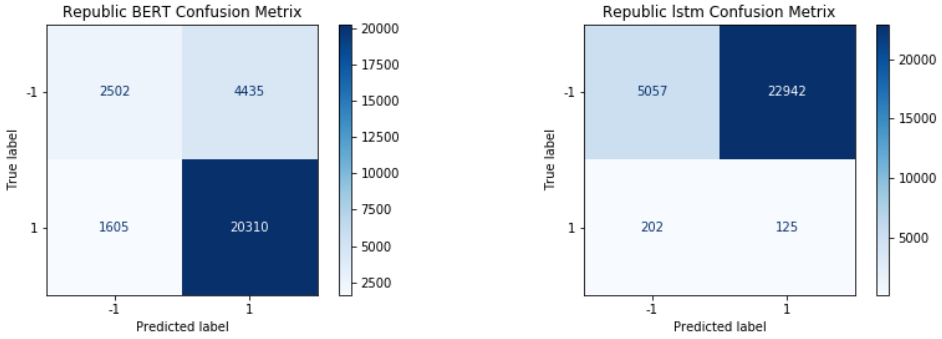

FIGURE 3.12: Confusion Matrix for LSTM and BERT

For the models, we have performed text classifications with to determine the similarity scores, we can say that the main score is that of the true positive, as the positive sentiment predictions, are much more value when taking the application scenarios into account.

Chapter 4

Key Findings and Outputs

Having scraped the data from multiple websites, using Scrapy spiders frameworks, further using multiple text pre-processing techniques to help standardize the data, using techniques such as stopwords removal, lemmatizer and different Regular Expressions, which allowed us to have a text corpus which devoid of all abnormalities, while still retaining the anomalies the vital information regarding the semantics and the word to word relations.

Now having made a corpus from the text data for the two feature columns : Article Post and Article Headlines. We apply different vectorization techniques so as to format the text data into a format such that it is machine readable. For this we use a bag of words model, TFIDFvectorizer and pre-trained word embeddings based vectors. Now we feed the vectorized output into the four different models, which are a mix of different supervised learning techniques, unsupervised learning techniques with Machine Learning approach and Deep Learning Models.

For the purpose of using the sentiment data for matching the different patterns with regards to our aims of comparing the sentiment score of the Headline and Article Post data, and comparing the sentiment score for both News websites based on category and date of publishing, we have decided to use the LSTM model sentiment scores over the BERT model scores.

4.1 ARE ARTICLE POSTS MORE POLARIZED THAN THE ARTICLE HEADLINES?

For our two News publication websites, we use two features columns : Article Post and Article Headline.

For this we created two corpuses, one separate for each of the news publication websites as for the project we felt, a common text corpus, as the headlines, being a very shorter than the article post, could lead to a smaller input data corpus.This step allowed us to make sure that the headlines sentiment output is not unnecessarily polarized, due to factors of less amount of text data.

Comparing the news articles from both the websites we can observe that both the publications do overall have much more positive things to say when it comes to long form writing, with almost similar distribution plots for both the News publications. We can also observe that there are very less amount of negative tags than the positive tags, and almost none are absolutely classified as negative scores.

Comparing the News Headlines based on their sentiment scores as taken from the

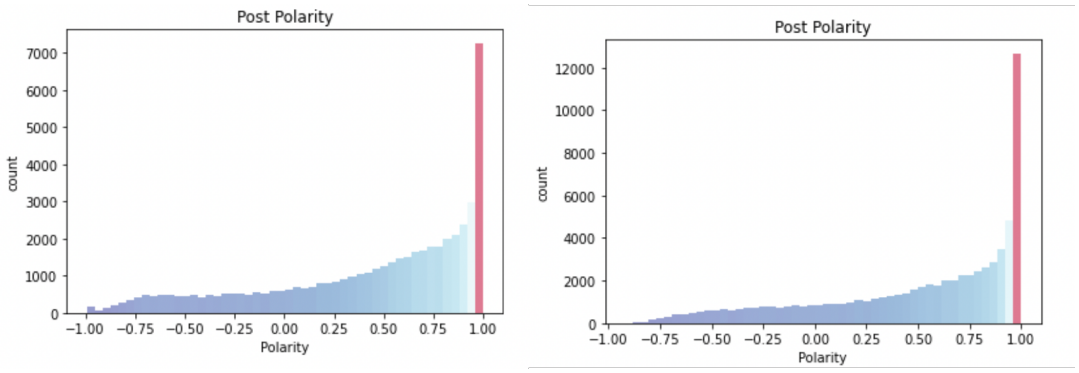

FIGURE 4.1: Article Post Sentiment Score

LSTM model which has approx. 82 per cent accuracy, we can observe a bigger difference between the distribution plots for the two news publications websites. The headlines for Publication 2 are much more neutral as compared to that of publication 1, for which we clearly see a distribution plot that is much more spread out. This indicated that the publication 1 uses overtly positive headlines for some purposes and doesn't shy away from using negative headlines for other purposes when it suits their purpose.

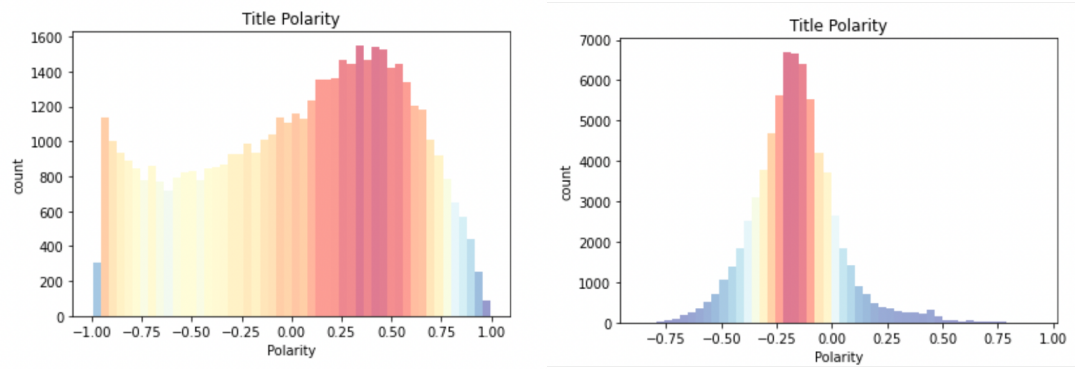

FIGURE 4.2: Headline Sentiment Score Distribution Plot

In the graphs you can analyse that mostly the post content is positive and up to the marks. The content is very least negative and even neutral. The title is showing more of a spread between positive and negative polarity. We can assume that Publication 1 websites mostly have more polar headlines to attract the readers but have positive content most of the time.

This graph was built for Publication 2 websites and to observe whether they have more polar posts or headlines. Same as publication 1 , publication 2 also kept most of their post content positive. They even have more positive content than the publication 1. With the title news publication 2 remains neutral most of the time. They were very least positive, but sometimes had negative titles as well.

4.2 DOES A PARTICULAR PUBLICATION TARGET SPECIFIC CATEGORIES WITH POLARISED NEWS?

We aim to compare the category wise distribution of the sentiment scores of both the News websites. For this we use the sentiment score from the LSTM model which has approx 82 per cent accuracy for both the websites data. We use the text corpus which we have extracted from the article texts and use as input to the text classification model.

Now comparing the different categories of the news we can observe certain factors, such as : The graph shows the topmost controversial and polar post from Publica-

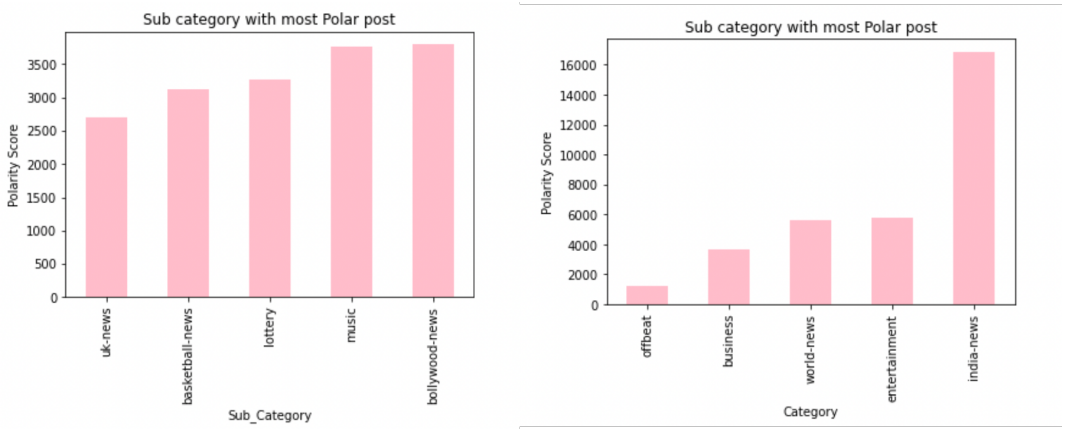

FIGURE 4.3: Plot for Article Sentiment Score vs News Category

tion 1 . The subcategory Indian-news was the most polar of all with entertainment coming second.

We wanted to find out which subcategory is most polar in Republic Tv. We built this bar graph and plot the top 5 subcategories. Bollywood-news was the top one and the music was second on the list.

That publication 2 is increasingly positive about the polarity score for news regarding India whereas publication 1 is most positive about a varied selection of categories.

This could suggest to the fact that the the publication 2 has increased focussed on giving a positive viewpoint on things in regards to matters of Indian matters of foreign policy and overall image while news publication 1 is much more focussed on a well rounded viewpoint on a variety of matters

One of the limitations we had to overcome was the categorization of the articles from multiple fields into single field and thus we labelling techniques for subcategory feature field creation.

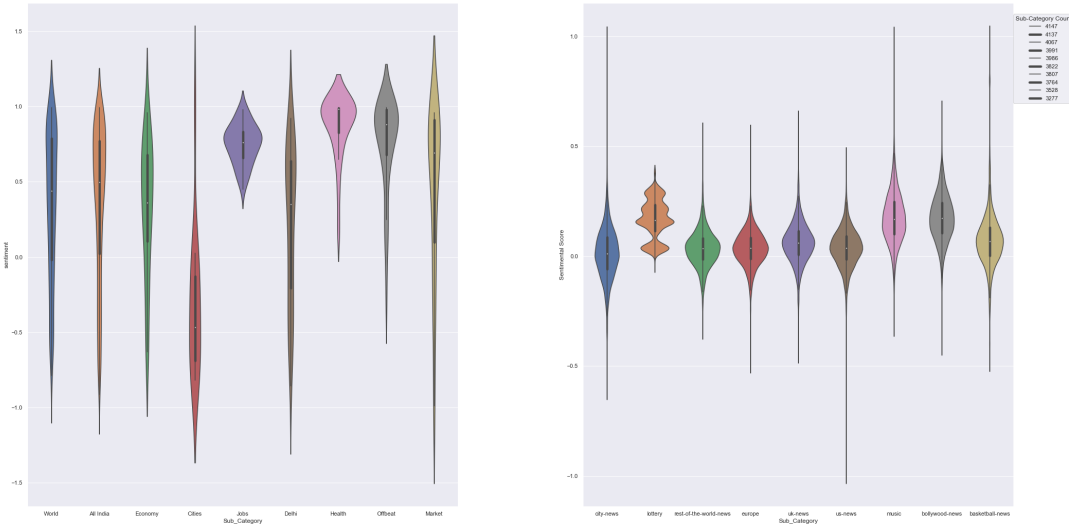

FIGURE 4.4: Violin Plot for Article Sentiment Score vs News Category

4.3 ARE THERE MORE POLARISED NEWS AROUND CERTAIN DATES/EVENTS?

We aim to compare the publishing date wise distribution of the sentiment scores of both the News websites. For this we use the sentiment score from the LSTM model which has approx 82 per cent accuracy for both the websites data. We use the text corpus which we have extracted from the article texts and use as input to the text classification model.

Now comparing the different sentiment scores for the various publishing dates of the news publications and tried to recognise patterns, we can observe certain factors, such as : To analyse if a few dates are having more polar content, then others,

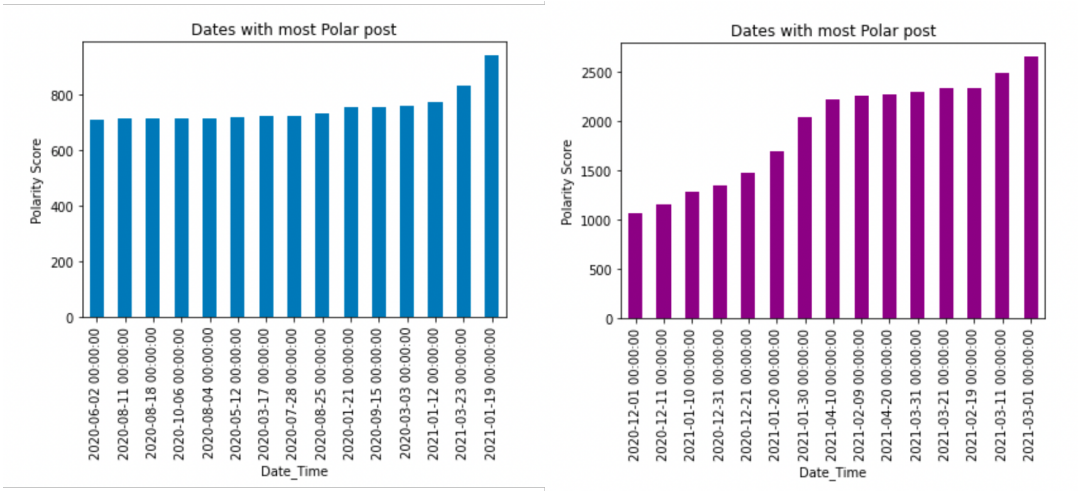

FIGURE 4.5: Plot for Article Sentiment Score vs News Publish Date

we created a chunk of a week to analyse it. We calculate the polarity score for that week and then create a bar graph out of it for the top 15 polar dates. According to

the graph almost all weeks are having the same polarity score showing that there were no days which were more polar than others.

The graph is for the top 15 most polar dates. Here we can see that Publication 2 was more polarized during March and the start of April. Their polarity score rose a lot during these days while having low the previous year.

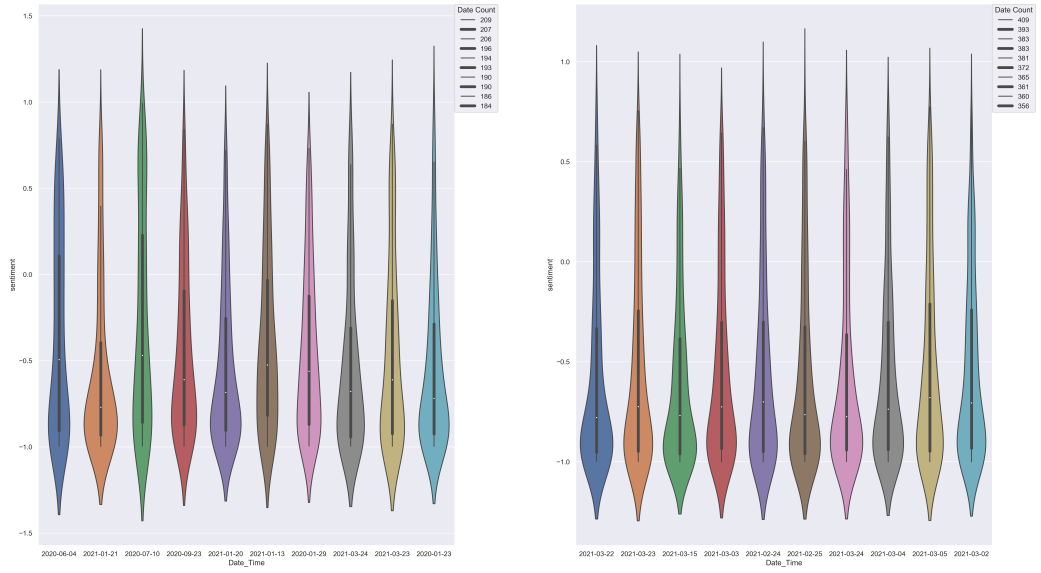

FIGURE 4.6: Violin Plot for Article Sentiment Score vs News Publish Date

This was an exercise to compare the sentiment scores and the resultant patterns in the dates where they have increasingly polar scores : From having a little bit of background on indian election schedules, movie release patterns and world sports events and matching them with the polarity :

We can observe that

- Around dates of sept/2020 and march 2021

  Both had increased positive polarity scores

- Around dates of June 2020

  Both had average to below average scores pertaining to the aftermath of the first covid-19 wave and the resultant blame game that started from all parties involved

- We also see that for entertainment categories,

  Around dates of EID>DIWALI>NEW Year there was increased positivity in tone of news in that order due to increased audiences/viewers during the holiday seasons

Chapter 5

Conclusion

5.1 Model Comparison

From the plot for the News Publication 1, we can see the sentiment score compared for all the 4 models we have used for text classification for sentiment analysis purposes along with the actual ground truth value sentiment score.

First we calculate the overall aggregate sentiment score for all 4 models, plus the ground truth model. When we calculate the sentiment score output, one thing we have to keep in mind is that a higher sentiment score for a model, means the classified data for the model, is skewed into certain tags, more than the others.

That on the face of it is not an ideal outcome, but if that is the case for all the models, for different sets of data, which we have used, then we can have an understanding from the model, that maybe the data on the whole is skewed.

For sentiment tag comparison, when each of the models is evaluated on a test subset of data, we can see that the best results are obtained using the Random Forest model. Now here we see that the BERT score is lower due to the more proportions of the positive to negative tags whereas the random forest model, most of the predictions are positive, but when we coupled the sentiment score metrics, with the accuracy metrics, we can observe that random forest model is not ideal at predicting the negative and neutral tags as accurately as other models, even though the accuracy of the random forest model is quite satisfactory.

We recommend checking the aggregate sentiment score in convolution to the accuracy score to understand the final performance evaluation of the models.

Now when we can see the flaws in the random forest model, we move to comparing the BERT and LSTM outputs and the accuracy score for BERT was 78.2 and 78.6, whereas for LSTM, the accuracy scores were, 82.6 and 82.3 for the two respective publications. Moving on the sentiment score, we can observe that BERT has a score that is lower than that of LSTM, meaning that it is more capable of classifying the data into positive and negative labels.

Thus we can conclude that the BERT model did give us a better performance, but LSTM output was very comparable to the BERT output. If we are to optimize the BERT output, we can make use of better hardware access and get rid of the overfitting limitation, which we observed after layer 2 while working on the BERT model.

Our main task was to prepare the dataset and apply the techniques to check which

model will work best. Here we tried 4 different techniques and found that LSTM was the best working one. We took this LSTM and tried to find the pattern within our dataset and try to prove our assumptions using data visualization.

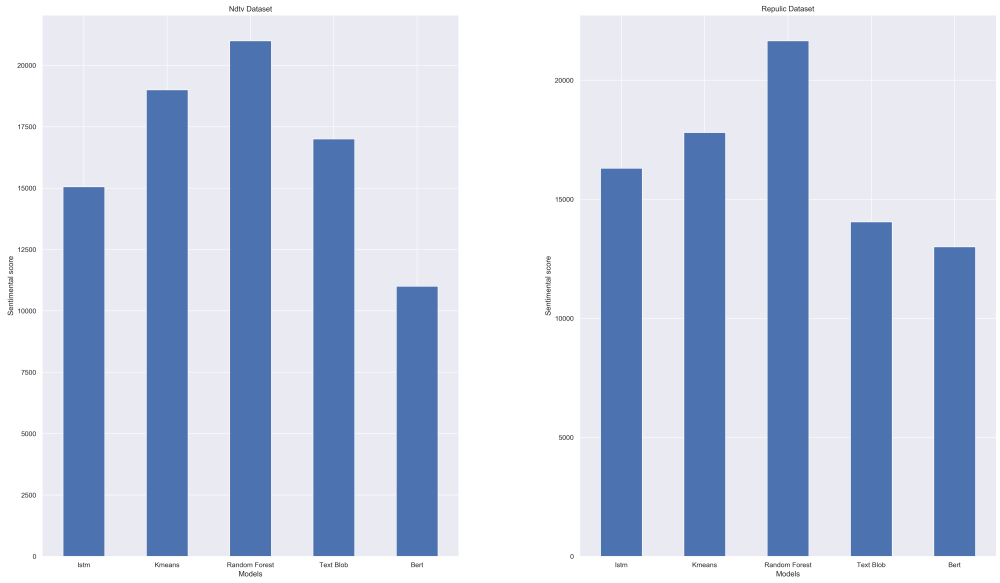

FIGURE 5.1: Model Sentiment Score Comparison Plot

5.2 Benefits

- Readers are better informed about the inclinations of the different publications and can abstain from trivial reactions
- Which further can result in less reactionary sharing on social media sites based on headlines alone
- Social media sharing platforms can use different text classification models for tagging of posts/links based on their sentiments
- News publications can better judge and model their team or editors and reporters to put out content that is more focused on facts and less on sentiments and emotions
- News aggregators platforms can accumulate news articles from source which are on all points on the sentiment spectrum
- The machine learning developers can further use the findings of this report to optimize their solutions for large datasets, which have multiple text fields and which need to serve text classification purposes.

5.3 Conclusion

From the work performed, we can establish some of the facts with regards to the features of any News Article, namely, that headlines DO tend to be more polar than the actual article, which if looked at face value can be aggressive and misleading but works just as well for publishing companies, who aim primarily to garner as many eyeballs as possible.

Secondly, we were able to establish that articles, DO tend to be more polar when written about some categories, but the key observations have to be made in categories where there are no clear wins or losses as these may be areas where the publishers can introduce their opinions and biases, one particular such category can be geopolitics, or world events.

Moreover, thirdly we were able to successfully conclude that world events, do tend to garner much more attention and these events can be geopolitical world events, major sports events, economic summits, and by sheer numbers, there is an increase in numbers of polar articles when in proportions of the number of news articles, i.e. the News is more polarized and we think we can attribute this to journalists and publishers trying to stand out with their views on a topic when everyone else is writing about the same topic at the same time.

Lastly, comparing the models, we are able to see that except for the K-Means clustering model, all the other three models, which are all supervised learning models, do tend to perform with a high level of satisfaction. Secondly, with fine tuning of the parameters of LSTM we were able to achieve more precise results which took them ahead in comparison to the BERT model output, but with better hardware specifications, BERT should give a much more state of the art-esque satisfactory output.

To conclude, from this research we can say that the combination of these polarization factors, may be a reflection on the NEWS cycle as a whole, starting with the publishers and end with the readers, but if we are to make the information and knowledge gained worthwhile and stand the test of logic and time, we should try to find NEWS which come from a multitude of perspectives rather than just a singular point of view.

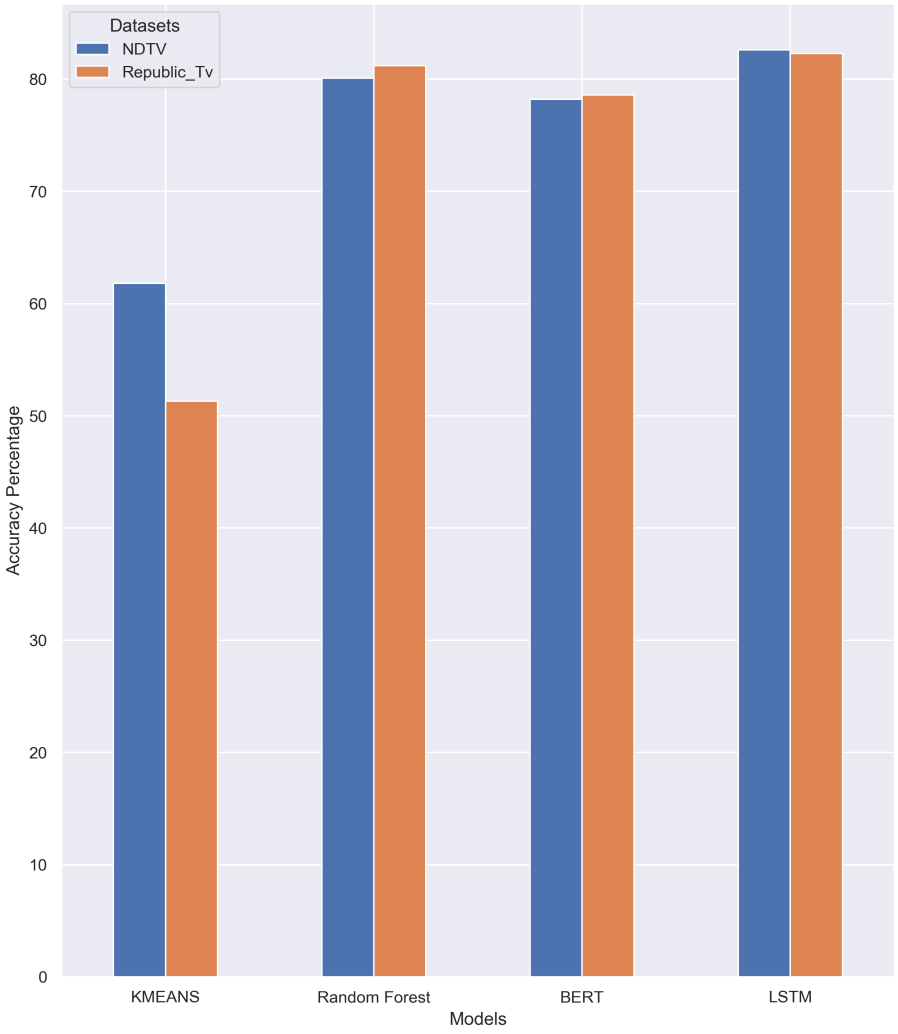

FIGURE 5.2: Model Accuracy Comparison Joint Plot

Bibliography

adeoyewole (2018). *Simple Guide to Scraping News Articles in Python*. URL: https://medium.com/@adeoyewole/scraping-news-articles-in-python-53c567282e25.

Agarwal, Apoorv et al. (2011). "Sentiment analysis of twitter data". In: *Proceedings of the workshop on language in social media (LSM 2011)*, pp. 30–38.

Ajao, Oluwaseun, Deepayan Bhowmik, and Shahrzad Zargari (2019). "Sentiment aware fake news detection on online social networks". In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 2507–2511.

Araque, Oscar et al. (2019). "Depechemood++: a bilingual emotion lexicon built through simple yet powerful techniques". In: *IEEE transactions on affective computing*.

Badjatiya, Pinkesh et al. (2017). "Deep learning for hate speech detection in tweets". In: *Proceedings of the 26th international conference on World Wide Web companion*, pp. 759–760.

Bai, Xuemei (2018). "Text classification based on LSTM and attention". In: *2018 Thirteenth International Conference on Digital Information Management (ICDIM)*. IEEE, pp. 29–32.

Bakshi, Rushlene Kaur et al. (2016). "Opinion mining and sentiment analysis". In: *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, pp. 452–455.

Bautin, Mikhail, Lohit Vijayarenu, and Steven Skiena (2008). "International sentiment analysis for news and blogs." In: *ICWSM*.

Bruns, Axel and Stefan Stieglitz (2013). "Towards more systematic Twitter analysis: metrics for tweeting activities". In: *International journal of social research methodology* 16.2, pp. 91–108.

datenstrom (2017). *TFIDF*. URL: http://datenstrom.gitlab.io/cs532-s17/notebooks/TFIDF.html#TFIDF.

Godbole, Namrata, Manja Srinivasaiah, and Steven Skiena (2007). "Large-Scale Sentiment Analysis for News and Blogs." In: *Icwsm* 7.21, pp. 219–222.

Gopalakrishnan, Karthik and Fathi M Salem (2020). "Sentiment Analysis Using Simplified Long Short-term Memory Recurrent Neural Networks". In: *arXiv preprint arXiv:2005.03993*.

Hansen, Lars Kai et al. (2011). "Good friends, bad news-affect and virality in twitter". In: *Future information technology*. Springer, pp. 34–43.

Hatzivassiloglou, Vasileios and Kathleen McKeown (1997). "Predicting the semantic orientation of adjectives". In: *35th annual meeting of the association for computational linguistics and 8th conference of the european chapter of the association for computational linguistics*, pp. 174–181.

Hoang, Mickel, Oskar Alija Bihorac, and Jacobo Rouces (2019). "Aspect-based sentiment analysis using bert". In: *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pp. 187–196.

Hoang, Tuan-Anh and Ee Peng LIM (2012). "Virality and susceptibility in information diffusions". In:

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Islam, Muhammad Usama et al. (2017). "Polarity detection of online news articles based on sentence structure and dynamic dictionary". In: *2017 20th International Conference of Computer and Information Technology (ICCIT)*. IEEE, pp. 1–5.

Karthika, P, R Murugeswari, and R Manoranjithem (2019). "Sentiment Analysis of Social Media Network Using Random Forest Algorithm". In: *2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS)*. IEEE, pp. 1–5.

Kaur, Sumandeep, Geeta Sikka, and Lalit Kumar Awasthi (2018). "Sentiment Analysis Approach Based on N-gram and KNN Classifier". In: *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*. IEEE, pp. 1–4.

Kharde, Vishal, Prof Sonawane, et al. (2016). "Sentiment analysis of twitter data: a survey of techniques". In: *arXiv preprint arXiv:1601.06971*.

Kohut, Andrew et al. (2010). "Americans spending more time following the news". In: *Pew Research Center*.

Kowsari, Kamran et al. (2019). "Text classification algorithms: A survey". In: *Information* 10.4, p. 150.

Lei, Jingsheng et al. (2014). "Towards building a social emotion detection system for online news". In: *Future Generation Computer Systems* 37, pp. 438–448.

Lerman, Kevin et al. (2008). "Reading the markets: Forecasting public opinion of political candidates by news analysis". In:

Liu, Bing (2012). "Sentiment analysis and opinion mining". In: *Synthesis lectures on human language technologies* 5.1, pp. 1–167.

MacRoberts, Michael H and Barbara R MacRoberts (2018). "The mismeasure of science: Citation analysis". In: *Journal of the Association for Information Science and Technology* 69.3, pp. 474–482.

Malik, Farhad (2019). *NLP: Introduction To NLP Sentiment Analysis*. URL: https://medium.com/fintechexplained/sentimental-analysis-an-introduction-7fc21d9b8625.

Medhat, Walaa, Ahmed Hassan, and Hoda Korashy (2014). "Sentiment analysis algorithms and applications: A survey". In: *Ain Shams engineering journal* 5.4, pp. 1093–1113.

Montalenti, Andrew (2019). *Machine learning for news*. URL: https://blog.parse.ly/post/7790/machine-learning-nlp-parse-ly-currents/.

Pak, Alexander and Patrick Paroubek (2010). "Twitter as a corpus for sentiment analysis and opinion mining." In: *LREc*. Vol. 10. 2010, pp. 1320–1326.

Poria, Soujanya et al. (2018). "Meld: A multimodal multi-party dataset for emotion recognition in conversations". In: *arXiv preprint arXiv:1810.02508*.

Reis, Julio et al. (2015). "Breaking the news: First impressions matter on online news". In: *arXiv preprint arXiv:1503.07921*.

Sak, Hasim, Andrew W Senior, and Françoise Beaufays (2014). "Long short-term memory recurrent neural network architectures for large scale acoustic modeling". In:

Shapiro, Adam Hale, Moritz Sudhof, and Daniel Wilson (2020). "Measuring news sentiment". In: Federal Reserve Bank of San Francisco.

Singh, Jaspreet, Gurvinder Singh, and Rajinder Singh (2017). "Optimization of sentiment analysis using machine learning classifiers". In: *Human-centric Computing and information Sciences* 7.1, pp. 1–12.

Souma, Wataru, Irena Vodenska, and Hideaki Aoyama (2019). "Enhanced news sentiment analysis using deep learning methods". In: *Journal of Computational Social Science* 2.1, pp. 33–46.

Swati, Ubale, Chilekar Pranali, and Sonkamble Pragati (2015). "Sentiment analysis of news articles using machine learning approach". In: *Proceedings of 20th IRF International Conference, 22nd February*.

Wang, Nianxin et al. (2016). "Cloud computing research in the IS discipline: A citation/co-citation analysis". In: *Decision Support Systems* 86, pp. 35–47.

Yadav, Kajal (2020). *Scraping 1000's of News Articles using 10 simple steps*. URL: https://towardsdatascience.com/scraping-1000s-of-news-articles-using-10-simple-steps-d57636a49755.

Zhang, Wenbin and Steven Skiena (2009). "Improving movie gross prediction through news analysis". In: *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*. Vol. 1. IEEE, pp. 301–304.

Zhou, Peng et al. (2016). "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling". In: *arXiv preprint arXiv:1611.06639*.