A Comparative Study of Machine and Deep Learning Methods for News Classification

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science (Engineering) in Computer Science and Engineering

Submitted By

Anik Hasan

ID: CE18009

Arnab Bairagi

ID: CE18022

Session: 2017-18

Supervised By

Lubna Yasmin Pinky

Assistant Professor,

Dept. of CSE, MBSTU

Department of Computer Science and Engineering

Mawlana Bhashani Science and Technology University

Santosh, Tangail-1902, Bangladesh

Approval

This Project titled **"A Comparative Study of Machine and Deep Learning Methods for News Classification",** submitted by Anik Hasan, ID: CE18009, and Arnab Bairagi, ID: CE18022 to the Department of Computer Science and Engineering, Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents.

BOARD OF EXAMINERS

1. **…… (Supervisor)**

2. **……. (Examiner)**

3. **……(Examiner)**

DECLARATION

We, hereby declare that this thesis has been done by us under the supervision of Lubna Yasmin Pinky, Assistant Professor, Department of Computer Science and Engineering, Mawlana Bhashani Science and Technology University, Santosh Tangail-1902, Bangladesh. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Countersigned	**Signature**	**Signature**
……..	…..…..……	……
(Lubna Yasmin Pinky)	(Anik Hasan)	(Arnab Bairagi)
(Supervisor)	ID: CE18009	ID: CE18022
	(Candidate)	(Candidate)

Acknowledgment

We want to start by giving thanks to the Almighty. He provided us with the skills, the opportunity, and a supportive supervisor so that we are now successful in finishing our work with such instances. We would like to thank our esteemed supervisor, Lubna Yasmin Pinky, from the bottom of our hearts for all of her help and support during the research, including her invaluable guidance and insight. Her advice and suggestions were quite helpful to us as we were writing our thesis report. Finally, we would like to express our gratitude to the authors whose works were referenced and quoted in this article.

Abstract

In order to organize and analyze the large amount of information available in the digital age, news classification is essential. The demand for automated systems that can effectively categorize news has increased as a result of the rapid growth of online news sources. In our research, we used machine learning and deep learning models on news headlines from a pre-existing American news dataset to see which gave us the best accuracy. As the pre-existing dataset was unbalanced, we created our own Bangladeshi news dataset, used the same models for classification, and compared their results. In our proposed methodology, we first preprocess the text data and encode the categories into numerical values. Then we map our text data into feature vectors using techniques such as TF-IDF vectorization and word embedding. For classification, we used Linear SVC, Multinomial Naive Bayes, Logistic Regression machine learning models, and a feed-forward neural network(FFNN). After training and testing our models with the American news dataset, Linear SVC gave us the highest accuracy of 74.4%, while Multinomial Naive Bayes had the lowest accuracy of 55.3%. On the other hand, when we used our proposed news dataset to train and test our models, the FFNN classifier had the highest accuracy of 81%, and Logistic Regression had the lowest accuracy of 70%. Thus, from our findings, we conclude that using a more balanced dataset improves the performance of classification models, and for our proposed Bangladeshi news dataset, the FFNN classifier gave the most satisfactory result.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AI - Artificial Intelligence

ML - Machine Learning

LR - Logistic Regression

NB - Naive Bayes

SVC - Support Vector Classifier

BERT- Bidirectional Encoder Representations from Transformers

NLP - Natural Language Processing

TF -Term Frequency

IDF - Inverse Document Frequency

FFNN - Feed Forward Neural Network

Chapter 1: Introduction

1.1 Overview

News classification refers to the process of categorizing news articles into different topics or categories. Different methods, including deep learning models, natural language processing, and machine learning algorithms, can be used to do this.

News classification is to automatically recognize and categorize news items with the appropriate category, such as politics, sports, entertainment, technology, health, etc. This helps publishers arrange their material and enhances the overall user experience on their website or platform while enabling readers to quickly access news stories that are relevant to them.

The classification of news presents a variety of difficulties, including managing a large number of categories, processing news pieces that cover various themes, and assuring the consistency and accuracy of the classification. However, thanks to developments in AI and machine learning, news classification has grown more precise and effective, and it is now extensively employed in the media sector.

1.2 Problem Statement

The purpose is to develop a system that can automatically classify news articles into relevant themes or categories based on their headlines. Nowadays, every news website has a large number of stories that are not organized properly. It is impossible to organize these news stories manually. To solve this issue, the development of a system that can take large amounts of news headlines from multiple sources and accurately group them into subcategories like politics, sports, entertainment, technology, business, etc. is required. Such a system can also categorize Bangla news articles from different online platforms. By automating the classification process, the major goal is to increase the effectiveness of news organization, retrieval, and recommendation systems.

1.3 Motivation

i. Effective Information Retrieval: News classification aids in effectively organizing and categorizing news stories given the abundance of news content available online. Users will find it simpler to access the information they need as a result.

ii. Customization: Users' news feeds can be made more relevant to them by using news categorization. News can be sorted and suggested based on readers' interests by examining their reading preferences and history.

iii. Targeted Advertising: News categorization can also be used to target ads to particular readership segments. Advertisers may provide more relevant advertising and increase click-through and conversion rates by knowing the subjects that readers are interested in.

iv. Analysis and Research: Trends in news reporting can be examined and researched using news classification. It is possible to spot and evaluate patterns and trends by grouping news stories into certain topic categories.

v. Sentiment Analysis: Sentiment analysis of news articles can also be done using the classification of news. Sentiment analysis can be used to understand the public's sentiment on various subjects by categorizing news stories based on their tone.

Overall, news classification is a helpful tool for media monitoring, search and retrieval, trend analysis, and personalization. The enormous volume of news content that is produced every day can be organized and analyzed more easily by grouping news pieces into several subjects or categories.

1.4 Purpose

i. The goal of news classification research is to create automated tools that can correctly classify news stories according to their subject matter and substance into several categories or subjects. This can be helpful for a variety of factors, such as:

ii. Increasing the effectiveness of news curation: It might be difficult for editors and journalists to keep up with all the most recent news due to the abundance of internet news information. Automating the process of sorting through news articles to find those that are most pertinent to a given topic or category is possible with the use of news classification systems.

iii. Improving the user experience: News classification can assist news organizations in offering their users more individualized content recommendations. News classification algorithms can recommend articles that are most likely to be of interest to certain readers by examining user behavior and preferences.

iv. Supporting analysis and research: Research in disciplines like media studies, political science, and sociology can be supported by news classification. Researchers can discover trends and media biases by examining patterns in news coverage across various categories and themes.

v. Increasing the precision of information retrieval: The classification of news can also increase the precision of search results. Search engines are better able to match user searches to pertinent news articles by categorizing news articles based on their content.

1.5 Outline of the research

Chapter 1: Includes introduction.

Chapter 2: Includes literature review

Chapter 3: Includes proposed methodology

Chapter 4: Includes the performance analysis

Chapter 5: Includes results analysis

Chapter 6: Includes conclusion and future works

Chapter 2: Literature Review

News classification is the task of automatically categorizing news articles into predefined categories. It has gained significant attention in recent years due to the rapid increase in the volume of news articles being published online. In this literature review, we will explore the various approaches and techniques used in news classification.

The author of the paper [1], developed the intelligent News Classifier and experiment with online news for the category Finance, Politics, and Sports. Here Hidden Markov Model and Support Vector Machine had been used to classify the problem. It provides good accuracy by providing several preprocessing techniques and the application of filters to reduce noise. Preprocessing in the training data set significantly reduced the training computational time.

In the paper [2], the main objective is to predict the classify the Financial news based content which is to predict classify the financial news-based content which is able to classify the news as either rising or dropping. Here, the author used two types of data: past intraday prices and past news articles. The prediction model applies all types of news related to the auto industry and others related to competition and compares the result with the current prediction model.

Here [3] the author aims to classify news into a variety of groups so that users can identify the popular news in a country at a time for his news classification he used Term Frequency-Inverse Document Frequency (TF-IDF) and Support Vector Machine (SVM).

Here [4] in the paper, a text classification system was created utilizing machine learning methods. Text categorization can be automated successfully using machine learning techniques, however, the size and caliber of training input provided to the classifier, which in turn influences the classifier performance, depend critically on pre-processing and feature selection processes. For a number of regional languages, sophisticated text classifiers are not yet accessible; but, if they were, they would be helpful for numerous governmental and commercial enterprises.

Here [5] the authors presented an advertisement detection, segmentation, and classification framework to facilitate advertisement studies in newspaper images and website snapshots. We classify advertisements based on visual analysis that attracted little attention before.

In this paper [6], the author used unlabeled data in order to solve a multi-label classification problem for an Indonesian news story by using Word2vec to create a vector representation of

the words. Using these vectors, we were able to extract the classification features by capturing the semantic similarity between words.

In this paper [7], News headline classification was the subject of the author's review. They explored which model is ideal for which sort of data, looked over more research papers, and briefly talked about feature selection, document indexing, and text preprocessing approaches. To make it simple for new researchers to understand models and preprocess deftness, preprocess data step-by-step procedures are presented.

From BBC Urdu and Urdu-point news headlines, Zaidi, Syed Adnan Ali, and Syed Muhammad Hassan [8] implement text classification using various machine learning algorithms, in which they find the best result from the Ridge Classifier. They create their own dataset by collecting data from different sites, classifying text based on news headlines, and analyzing which algorithm is best for the dataset they created.

Chapter 3: Methodology

3.1 Proposed Methodology

For performing news classification from their headlines, we proposed two different methods. In the first method, we start with an initial dataset. The headlines of the news stories from the dataset are preprocessed. First, the headlines are changed to lowercase. Then we remove stopwords and punctuation from the headlines. We finish the preprocessing by performing stemming on the headlines. After that, we split the dataset into training and testing data. Then we extract features from the training data and feed these features to machine learning classifiers for training. After training, we use the testing dataset to evaluate the performance of the classifiers. figure 3.1 illustrates the whole process.

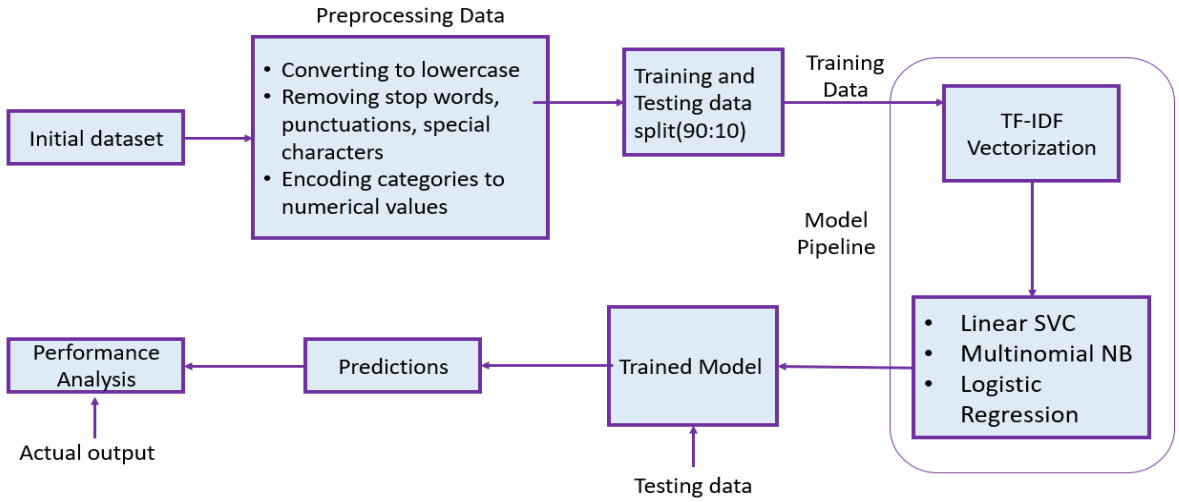

Figure 3.1: Proposed Methodology(TF-IDF and ML classifiers)

Similarly, in the second method, we start with an initial dataset. We perform preprocessing on the headlines of the dataset and convert the news categories to numerical values. After that, we split the dataset into training and testing sets. Then we tokenize the news headlines of both the training and testing datasets. From the tokenized headlines of the training dataset, we generate word embeddings with the help of a pretrained BERT model. These word embeddings are used for training the classifier. Here, a feed-forward neural network is used as a classifier. This classifier is trained for multiple epochs. Finally, we use the testing dataset to evaluate the performance of the classifier. figure 3.2 illustrates the whole process.

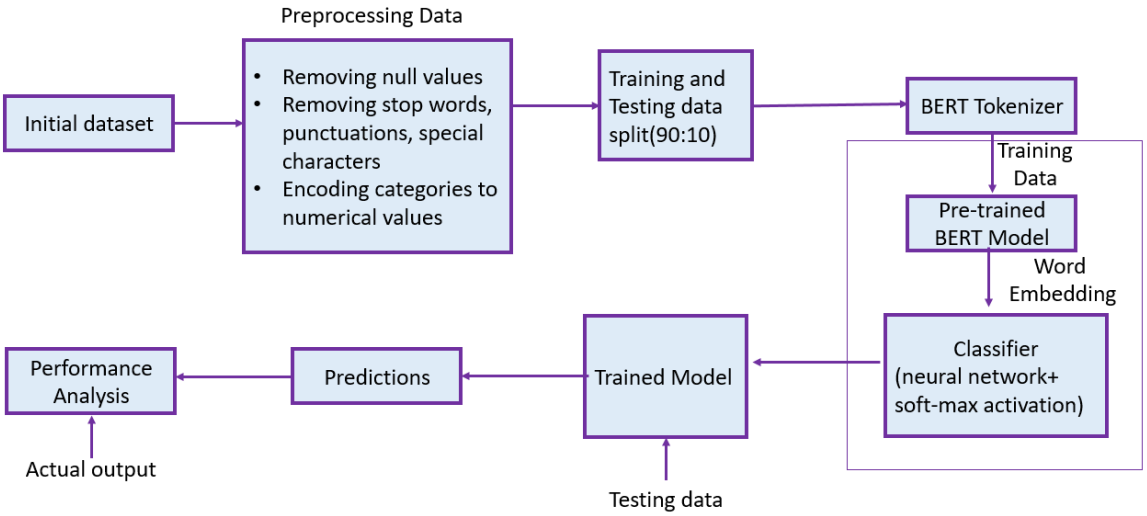

Figure 3.2: Proposed Methodology(Word Embedding and Neural Network)

3.2 Dataset Description

A dataset with a good number of data points is required for building a classification model. A data set is used to train and test the classification model. Thus, data sets are a crucial part of a machine learning model.

3.2.1 American News Dataset

To build our news classification model, we initially used a dataset from Kaggle that contains around 20K news headlines from 2012 to 2022 from HuffPost [9]. HuffPost is an American progressive news website. This dataset contains news headlines from 21 categories. figure 3.3 illustrates the frequencies of news categories in the data set.

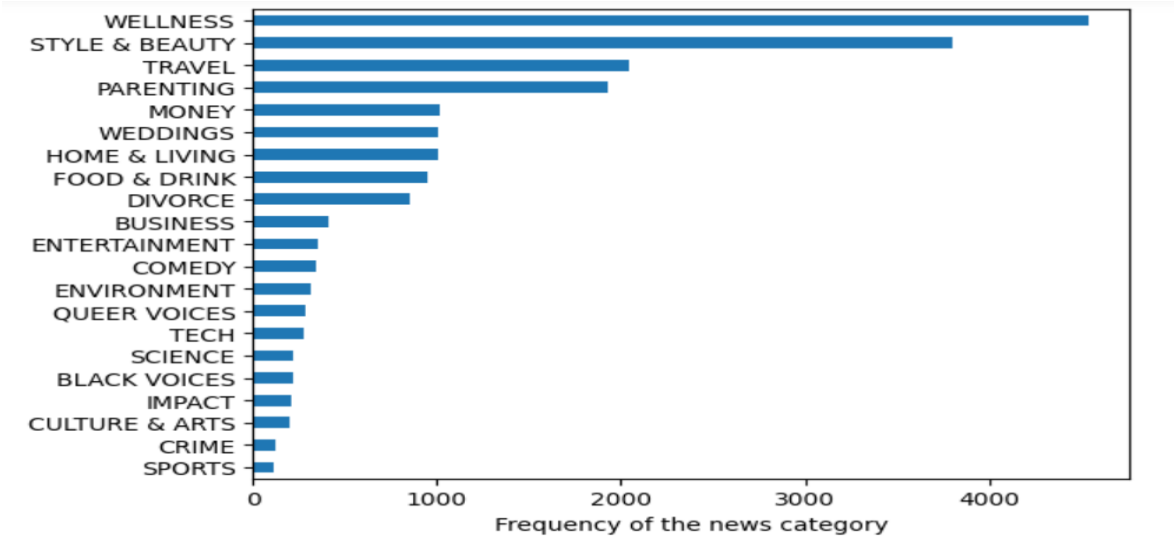

Figure 3.3: Frequency of the categories from the American news dataset

3.2.2 Bangladeshi News Dataset

For classifying Bangladeshi news and improving the performance of the news classification models, we created a dataset that contains 2K news headlines from various Bangladeshi news websites. The headlines are in English. A large portion of the headlines are from local news. This dataset contains 10 categories of news. figure 3.4 illustrates the frequencies of news categories in the data set.

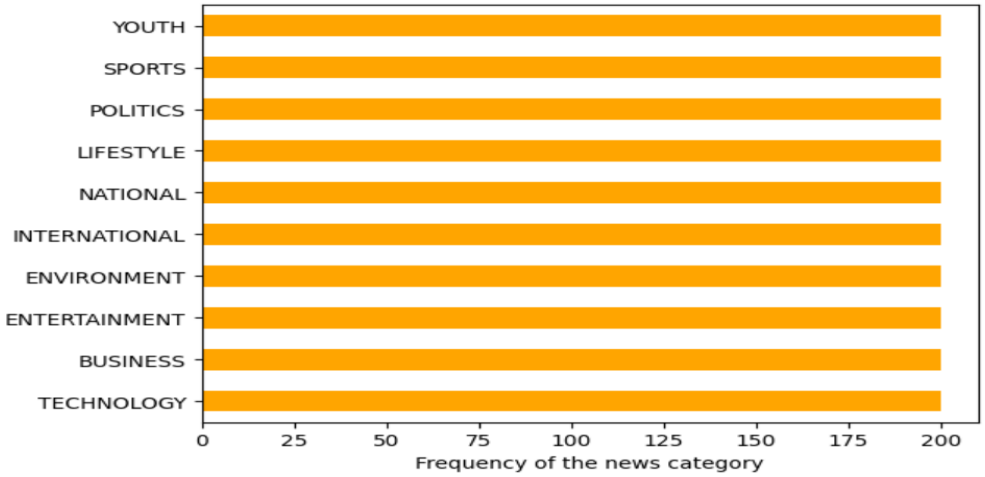

Figure 3.4: Frequency of the categories from the Bangladeshi news dataset

3.2.3 Dataset Column Description

Both datasets consist of three columns, such as category, headline, and link. Table 3.1 gives a description of each column in the data set.

Table 3.1: Dataset Column Description

Column Name	Description
category	Category of the news i.e. 'SPORTS','POLITICS',etc.
headline	Headline of the news article. The length of each headline is less than 128 words.
link	Link to the news website from which the headline is collected.

Figure 3.5 shows a snapshot of the Bangladeshi news dataset after it is loaded into a dataframe.

category	headline	link
ENVIRONMENT	Cabinet approves proposals for import of LNG, ...	https://www.thedailystar.net/environment/natur...
NATIONAL	58 markets inspected in three months, all foun...	https://en.prothomalo.com/bangladesh/nw5jj4y4x9
BUSINESS	EU unveils reforms for cheaper drugs and to av...	https://en.prothomalo.com/business/global/53db...
NATIONAL	Foreign competitor worries local airlines	https://www.thedailystar.net/news/bangladesh/n...
LIFESTYLE	https://www.thedailystar.net/life-living/healt...	https://www.thedailystar.net/life-living/healt...
...	...	...
ENVIRONMENT	Mild heat wave likely to continue and spread	https://www.thedailystar.net/environment/weath...
SPORTS	WWE, UFC merge together, creating $21B fightin...	https://en.prothomalo.com/sports/xoqp7vflk1
INTERNATIONAL	Russia arrests US reporter on spy charges	https://en.prothomalo.com/international/europe...
SPORTS	FIFA hands two-year ban to BFF secretary Shohag	https://en.prothomalo.com/sports/football/4fda...
INTERNATIONAL	'Release of hydrogen sulphide was a reason of ...	https://en.prothomalo.com/international/india/...

Figure 3.5: Bangladeshi news dataset

3.3 Data Preprocessing

Taking raw data and converting it to be acceptable for a ML model is referred to as data preprocessing. It is the initial and most important step in developing a machine learning model [10].

Real-world data typically includes noise, missing values, and may be in an undesirable format, making it impossible to build machine learning models directly on it. Some preprocessing is required to clean the data and make it acceptable for a machine learning model, which also improves the model's accuracy and effectiveness [11].

Figure 3.6 shows the preprocessing techniques used on our dataset for news classification.

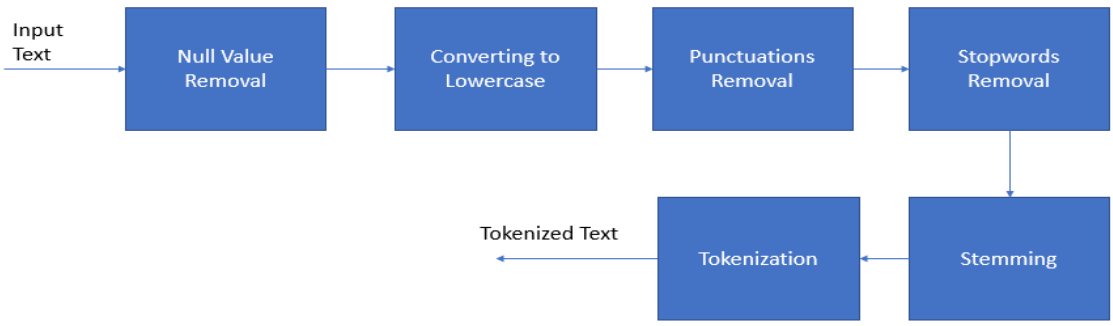

Figure 3.6: Data preprocessing technique

3.3.1 Null Value Removal

Most datasets from the actual world will have some missing values. There may be several causes for this. It's possible that the data generation system made a mistake, resulting in missing observations. Whatever the reason, the majority of ML algorithms cannot process null values to solve classification problems, and it is, therefore, necessary to treat these values in some way [10].

For our news classification problem, we dropped those rows in the dataset that had any null values in them. We used the Python Pandas dropna() method to remove these records [12].

3.3.2 Punctuation Removal

Taking punctuation out of text data is one of the most popular text preprocessing approaches. Punctuations in the Python string class contain the following symbols:

!"#$%&\'()*+,-./:;?@[\\]^_{|}~`.

Through this technique, words that would have been treated differently due to punctuation are treated equally. For example, the words "bear" and "bear!" are regarded similarly [13].

3.3.3 Converting Text to Lowercase

This is another popular preprocessing technique for textual data. In this process, every text in the dataset is converted into the same casing format. For example, the text "Russia declares war on Ukraine" becomes "russia declares war on ukraine" [13].

3.3.4 Stopwords Removal

For building classification models that deal with textual data, stopwords might not carry any significant importance in the textual data. By removing stopwords, more focus is given to words that represent the meaning of the textual data.

For our system, we use Python's NLTK library to get the stopwords of the English language. Then we remove the stopwords from every news headline in the dataset. [14].

3.3.5 Encoding Categories to Numerical Values

The news categories in our dataset are in string format. Machines cannot understand these categories directly. Thus, these categories are encoded into numerical values so that our models can process the dataset for solving classification problems. For the models that use ML

classifiers, we encode the categories using LabelEncoder() from the sklearn library in Python [15].

This method encodes the categories in the following way:

{'TECHNOLOGY', 'BUSINESS', 'ENTERTAINMENT', 'ENVIRONMENT'} = {0,1,2,3}

For the model that uses a neural network as a classifier, we further encode the categories using the to_categorical() method from keras Library in Python. This method converts the numeric value of each category into an array that only has binary values, and the size of this array is equal to the number of categories in the dataset [16]. The following figure illustrates this process:

```
output=to_categorical([0,1,2], num_classes = 3)
print(output)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

Figure 3.7: Category encoding with to_categorical() method

3.3.6 Stemming

Stemming is an NLP technique that reduces words to their basic form, or stem, thereby aiding in the preprocessing of textual data. In the methodology illustrated in figure 3.1, we used stemming on the headlines of the dataset. There are different stemming algorithms, and all of them are included in Python NLTK. We used Porter Stemmer, which is the original stemmer and is very well known [17].

The following figure shows how we can use the PorterStemmer( ).stem( ) method for stemming textual data:

```
from nltk.stem.porter import PorterStemmer
ps=PorterStemmer()
def stemWords(text):
    return " ".join([ps.stem(word) for word in text.split()])

print("Output = "+stemWords("He is running away."));
```

```
Output = he is run away.
```

Figure 3.8: Stemming on textual data

3.3.7 Tokenization

Tokenization is a common task that is required while building machine learning models for textual data. In this process, we split an entire text into small units, also known as tokens. In the methodology illustrated in figure 3.1, this tokenization occurs during the TF-IDF vectorization process with the help of the Count Vectorizer from sklearn in Python. Count Vectorizer helps us to tokenize the texts and also converts the collection of textual data to a vector of token counts. These token counts are later used for feature extraction processes such as TF-IDF vectorization [18]. The following figure shows the output of the Count Vectorizer :

```
text = ["I choose a lazy person to do a hard job. Because a lazy person will find an easy way to do it."]

vectorizer = CountVectorizer()
# tokenize and build vocab
vectorizer.fit(text)

print(vectorizer.vocabulary_)

# encode document
vector = vectorizer.transform(text)
# summarize encoded vector
print(vector.shape)
print(vector.toarray())
```

```
{'choose': 2, 'lazy': 9, 'person': 10, 'to': 11, 'do': 3, 'hard': 6, 'job': 8, 'because': 1, 'will': 13, 'find': 5, 'an': 0, 'e
asy': 4, 'way': 12, 'it': 7}
(1, 14)
[[1 1 1 2 1 1 1 1 1 2 2 2 1 1]]
```

Figure 3.9: Tokenization with Count Vectorizer

In the second methodology illustrated in figure 3.2, we used the BERT tokenizer. The BERT tokenizer is a specific component of the BERT model that performs the tokenization process. The output of this tokenizer is later used by a pretrained BERT model to generate word embeddings. The BERT tokenizer also adds special tokens to the tokenized input, such as [CLS] at the beginning of the sequence to represent the classification task and [SEP] to separate two sentences. These special tokens help the BERT model understand the structure and relationships within the input text and generate contextual word embedding. The BERT tokenizer adds padding to each of the tokenized texts to make them equal length. For every tokenized text, the BERT tokenizer generates three vectors: input_ids, token_type_ids, and attention_masks. Here input_ids consists of unique integers for each token in the tokenized input based on its position in the tokenizer's vocabulary. The attention_masks consists of binary sequences of 0s and 1s for each tokenized input, with 1s representing tokens that should be

attended to and 0s representing tokens that should be masked. The token_type_ids indicate the segment or sentence to which each token belongs [19]. The following figure shows the output of the BERT tokenizer for a text input:

```
encoding = tokenizer(
    'Anthony Edward Stark known as Tony Stark is a fictional character in Avengers',
    add_special_tokens=True,
    max_length=20,
    truncation=False,
    padding='max_length',
    )
for key, value in encoding.items():
    print( '{} : {}'.format( key, value ) )
```

```
input_ids : [101, 4938, 3487, 9762, 2124, 2004, 4116, 9762, 2003, 1037, 7214, 2839, 1999, 14936, 102, 0, 0, 0, 0, 0]
token_type_ids : [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
attention_mask : [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0]
```

Figure 3.10: Tokenization with BERT tokenizer

3.4 Feature Extraction

A set of representative features are created through the feature extraction process, which involves choosing and converting pertinent information from raw data. These characteristics are typically more condensed, significant, and appropriate for additional analysis or machine learning activities. An initial set of raw data is reduced to more manageable groupings for processing using the dimensionality reduction technique known as feature extraction [20].

There are several techniques by which we can achieve this task. For our models, we have used techniques such as TF-IDF vectorization and the generation of word embeddings.

3.4.1 TF-IDF Vectorization

Term frequency -inverse document frequency (TF-IDF) is a popular technique that is used to map textual data to a finite-length vector. Its working principle is derived from the Bag of Words model. Term frequency (TF) is a measure of the frequency of a word (w_i) in a document (d_i) [21]. TF is calculated using the following formula:

$$TF(w_i, d_i) = \frac{occurences\ of\ w_i\ \ in\ document\ d_i}{total\ number\ of\ words\ in\ document\ d_i} \quad \text{.....(i)}$$

Inverse document frequency (IDF) of a word (w_i) reflects the proportion of documents in the corpus (D) that contain the word. Higher importance is given to words that occur rarely in the corpus than to words that occur frequently across all documents [21]. IDF is calculated using the following formula:

$$IDF(w_i, D) = log\left(\frac{total\ number\ of\ document(N)\ in\ the\ corpus\ D}{number\ of\ documents\ containing\ w_i}\right) \quad \text{.....(ii)}$$

Finally TF-IDF for a word(w_i) in a document(d_i) is calculated as following way:

$$TF\text{-}IDF(w_i, d_i, D) = TF(w_i, d_i) * IDF(w_i, D) \quad \text{.....(iii)}$$

The model from figure 3.1 uses the count matrixes of the news headlines given by the Count Vectorizer to calculate the TF-IDF value for each term in the tokenized news headlines. We implement this using the TfidfTransformer from sklearn in Python. Thus, this process maps each news headline to a feature vector consisting of TF-IDF values.

3.4.2 Word Embeddings

Word embedding is a way of mapping words in a text to a feature vector, where words having similar meaning will have similar feature vectors. Word embeddings are a class of techniques where every word is represented by a real-valued vector in a fixed vector space. Every word in the corpus is mapped to a vector, and the vector values are learned through a process that resembles a neural network [22].

To create word embedding for the token IDs provided by the BERT tokenizer, the model in figure 3.2 employs a pretrained BERT model. We applied the BERT base model to our classification model. The hidden layer size for this is 768. This indicates that each token is assigned to a 768-sized vector. The "contextualized embedding" or "BERT embedding," which is the final hidden state representation from BERT, includes contextual information about each character in the input sequence. These embeddings take into account the context that the complete input text provides to capture the linkages and dependencies between the tokens [23].

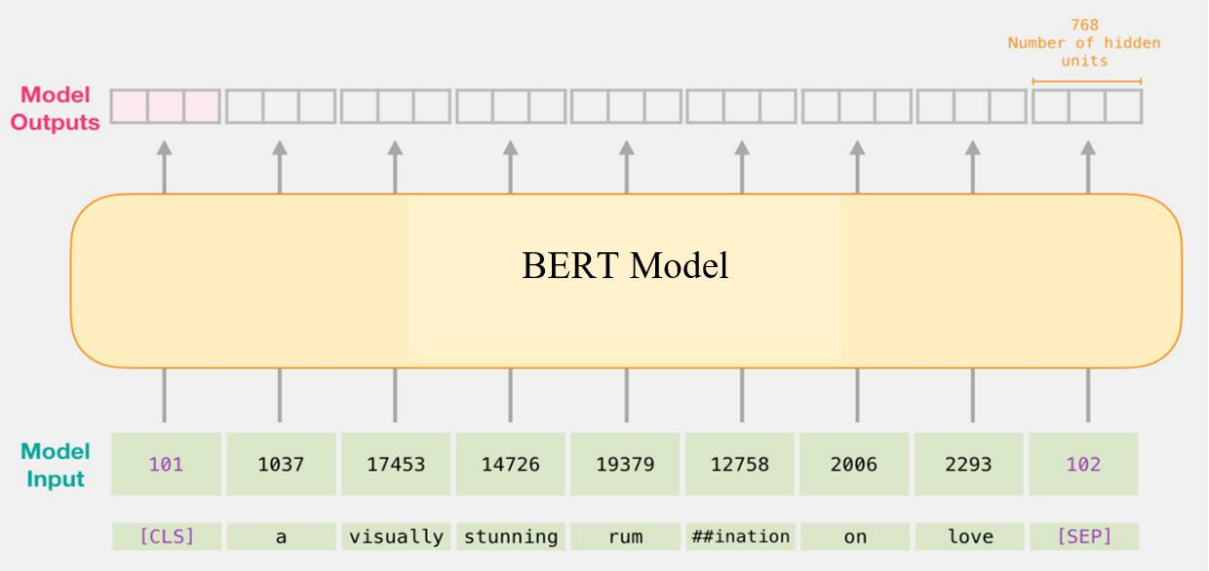

Figure 3.11: Word embedding with BERT model [24]

3.5 Training and Testing Data

In the context of machine learning, training and testing data are two distinct subsets of a dataset that are used for different purposes:

i. Training Data: A machine learning model learns to classify its inputs through training data. It consists of a collection of labeled instances, where each example has a target or output value and related input attributes. In order to make predictions or categorize data, the model learns patterns and correlations within the training data. In order to reduce the error or disagreement between the projected outputs and the actual outputs, the model makes adjustments to its internal parameters during training based on the input and output pairs provided.

ii. Testing Data: A trained machine learning model is evaluated for performance and generalizability using testing data. It consists of an extra set of labeled instances that weren't utilized in the training. Based on the knowledge it has gained from the training data, the model makes predictions or assigns categories to the testing data.

For building our news classification models, we used 90% of the entire dataset as a training dataset and the remaining 10% as a testing dataset.

3.6 Classification Models

After finishing all the steps in the methodology of figure 3.1, we fed the extracted features of our dataset into various ML classifiers such as linear SVC, multinomial NB, and logistic regression to train the model and make predictions for the news category. Similarly, we use a feed-forward neural network in the methodology depicted in figure 3.2 for categorizing news. Finally, we assess which classifier performs best for predicting the news category from its headline.

3.6.1 Linear Support Vector Classifier

Linear SVC (Support Vector Classifier) is a type of linear classification algorithm that is based on support vector machines (SVM). It works by locating a hyperplane in the feature space that divides instances of various classes as widely as possible. The margin, or the separation between the hyperplane and the closest data points (support vectors) in each class, is what it seeks to maximize. The hyperplane is a decision boundary that helps in making predictions for new, unseen instances. The linear kernel function used by linear SVC makes the assumption that the data may effectively be segregated from one another by a straight line or hyperplane in the feature space [24].

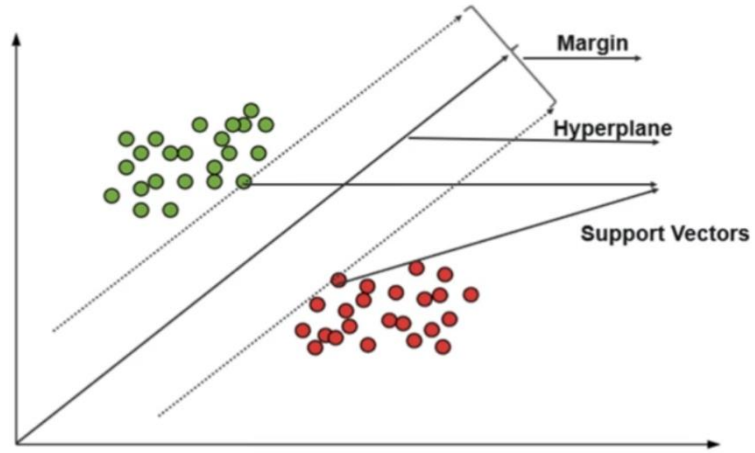

Figure 3.12: Classification with Linear SVC [24]

Though linear SVC inherently performs binary classification, it can be extended to handle multi-class classification problems such as our news classification problem by using techniques such as one vs. rest (here each class is treated as a separate binary classification task) or one vs. one (each pair of classes is treated as a separate binary classification task). These techniques break down the multiclassification problem into multiple binary classification problems. Linear SVC is computationally efficient for large-scale datasets, such as text datasets. Text data is

typically depicted as a high-dimensional feature vector. Linear SVC assumes that the feature space can be separated by a hyperplane. While this assumption may not hold for other types of data, it is often true for text data [25].

3.6.2 Multinomial Naïve Bayes

Multinomial Naïve Bayes is a widely used supervised classifier that can be used for categorical text data. Multinomial NB is a probabilistic learning method for NLP tasks, such as our news classification task. For a multi-class problem, it calculates the probability of data belonging to each class and then assigns the class with the highest probability as output. The only calculations this classifier requires are probabilities. It can easily handle large datasets, such as text data [26].

Given a feature vector of a text document W=[w1, w2,.., wn], where w_i represents the frequency of a particular word in the document, and a set of classes C = {c_1, c_2, .., c_k}, the formula for Naive Bayes classification is as follows:

$$P(c_i|W) = (P(W|c_i) * P(c_i)) / P(W) \quad \text{.....(iv)}$$

Where,

- $P(c_i|W)$ is the posterior probability of class c_i given the feature vector W. This represents the probability that the document belongs to class c_i given its observed features.
- $P(W|c_i)$ is the likelihood of the feature vector W given class c_i. This represents the probability of observing the feature vector W given that the document belongs to class c_i.
- $P(c_i)$ is the prior probability of class c. This represents the probability of a document belonging to class c_i, irrespective of its features.
- P(W) is the probability of the feature vector W. This represents the probability of observing the feature vector W, regardless of the class.

Classifier calculates the posterior probability for each class and labels the input with the class with the maximum posterior probability.

3.6.3 Logistic Regression

Logistic regression is widely used for binary classification tasks, where the goal is to determine the probability of an instance belonging to one of two classes. It is a supervised learning algorithm that is commonly used in various fields, including machine learning and statistics.

Logistic regression uses sigmoid the function to map predictions and their probabilities. The sigmoid function refers to y =1/(1+e^{-x}) which has an S-shaped curve [27].

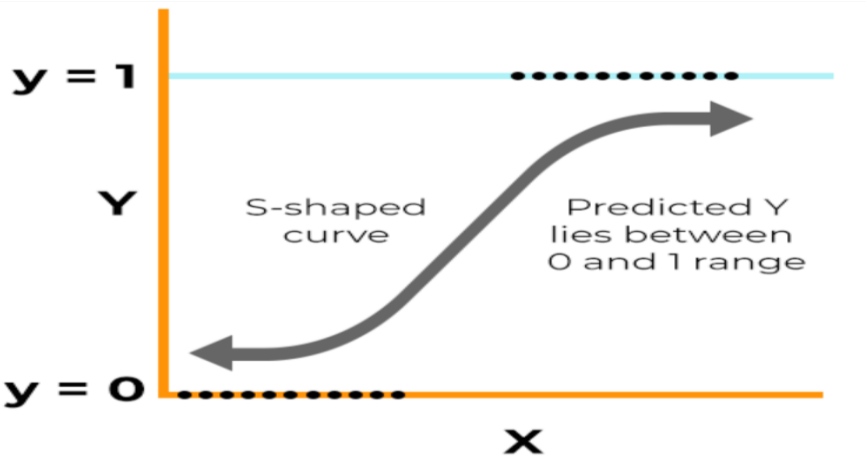

Figure 3.13: Classification with logistic regression [27]

Logistic regression can be used for text classification tasks. For this classifier to work, text is mapped to a feature vector of numerical values. For handling multi-class classification problems, logistic regression can be extended using different strategies. Here are two common approaches:

- One-vs-All: In this approach, a separate logistic regression model is built for each class. It treats one class as the positive class, while the rest of the classes are combined as the negative class. During training, each model is trained to predict the probability of an associated class versus the probability of all other classes. During prediction, the class with the highest predicted probability is assigned the final class label [28].

- Multinomial Logistic Regression: Multinomial logistic regression generalizes logistic regression to handle multiple classes directly. Here, a single model is trained to predict the probabilities of all classes. The softmax function is used as the activation function to compute the probabilities of each class. The softmax function ensures that the predicted probabilities always sum up to 1. During training, the model's parameters are fine-tuned to maximize the likelihood of the observed data across all classes. During prediction, the class with the highest predicted probability is selected as the final class label [28].

3.6.4 Feed Forward Neural Network

A fully connected feed-forward neural network (FFNN) is a type of artificial neural network where each neuron in one layer is connected to every neuron in the subsequent layer. It is commonly used for various classification tasks. In the methodology illustrated in figure 3.2, we used a similar feed-forward neural network to categorize the news headlines. Three layers make up our neural network: an input layer, a hidden layer, and an output layer. The input layer accepts word embeddings generated by the BERT model. The input layer of the network has the same size as the dimensionality of the word embeddings. Here, the size of the input layer is 768, the same size as the BERT embeddings. The size of hidden layers can vary from problem to problem. Here, our network has 256 hidden layers. The output layer of our network has the same size as the number of classes in the data set [29]. A soft-max activation function is applied to the output layer. The equation for the soft-max activation function is given here:

$$\text{Softmax}(y_i) = \frac{e^{yi}}{\sum_j e^{yj}} \quad \text{.....(v)}$$

where y represents the output layer's value. These values are transformed into probabilities by dividing by the product of exponential values. The neural network's structure is depicted in the following figure:

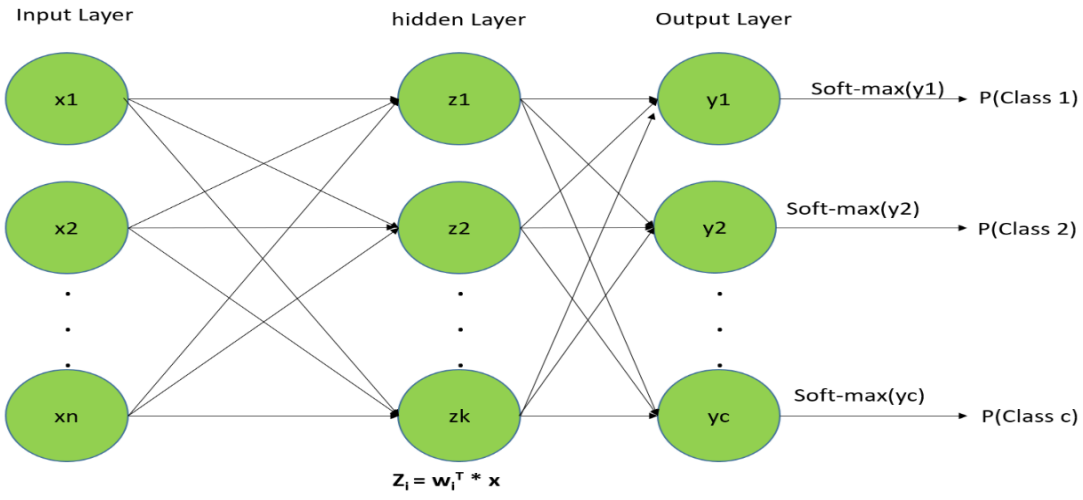

Figure 3.14: Feed forward neural network

During training, the neural network is presented with a labeled training dataset. The network calculates the predicted outputs using the current weights and compares them with the true

labels. The difference between the predicted and true labels is quantified using a loss function, such as categorical cross-entropy for multiclass classification. The network then adjusts the weights using backpropagation and an optimization algorithm, such as stochastic gradient descent, to minimize the loss and improve the model's accuracy. Once the neural network is trained, it can be used to make predictions on new, unseen data. The input data is fed through the network, and the output layer produces the predicted probabilities for each class. The class with the highest probability is selected as the predicted class label for the input data [30].

Chapter 4: Performance Analysis

4.1 Confusion Matrix

A table called a confusion matrix is used to evaluate the effectiveness of a classification model. It lists and compares the model's predictions with the actual values of a collection of data. When working with datasets that are unbalanced and have an uneven distribution of classes, the confusion matrix is particularly helpful.

A confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model [31]. In Table 4.1, we have shown the structure of a confusion matrix for a binary classification.

Table 4.1: Confusion matrix

	Predicted Negative(N)	**Predicted Positive(P)**
Actually Negative(N)	True Negative (TN)	False Positive (FP)
Actually Positive(P)	False Negative (FN)	True Positive (TP)

True Positive (TP): A collection of samples that are correctly identified as being in a positive class is said to be true positive (TP).

True Negative (TN): A collection of samples that have been correctly recognized as being in the negative class is known as a true negative (TN).

False Positive (FP): False positive samples refer to those that belong to the negative category but are mistakenly identified as being in the positive category.

False Negative (FN): False negative samples are those that belong to the positive class but are mistakenly categorized as negative samples.

We will calculate the following measures to evaluate a model's performance:

Accuracy: This term refers to how frequently the model predicts the correct outcomes. The ratio between the number of accurate predictions and the total number of predictions is known as accuracy. In a situation with equal false positives and false negatives, accuracy works well. Equation (vi) shows the formula to calculate accuracy.

Accuracy = (TP + TN) / (TP + FP + TN + FN) ……. (vi)

Precision: The ratio between accurate predictions and overall accurate predictions It measures the number of positive predictions (true positives) that are correctly made. Equation (vii) shows the formula to calculate the precision.

Precision = TP / (TP + FP) ..... (vii)

Recall: Recall is the percentage of total positive predictions that our model accurately predicts. Out of all positive cases in the data, it calculates the number of positive cases the classifier correctly predicted. Equation (viii) shows the formula to calculate recall.

Recall = TP / (TP + FN) .....(viii)

F1-Score: By using the F1-score, we may simultaneously assess recall and precision. When recall and precision are equal, the f1-score is high. F1 is considered more useful than accuracy in cases of uneven or imbalanced classes. Equation (ix) shows the formula to calculate the f1-score

F1-Score = 2 * (precision * recall) / (precision + recall) …….....(ix)

The confusion matrix provides a comprehensive view of the model's performance, allowing for the calculation of various evaluation metrics such as accuracy, precision, recall, and F1 score. These metrics provide insights into the model's ability to correctly classify instances and can help assess its strengths and weaknesses.

4.2 Confusion Matrix for Multiclass Classification

For a multi-class classification problem with N different class labels, the confusion matrix is an N*N matrix similar to figure 4.1 given below. The American news dataset [9] that we used to train our models had 21 class labels (N = 21). This means we can generate a confusion matrix of size 21*21 from the predictions of the models while using this dataset [9].

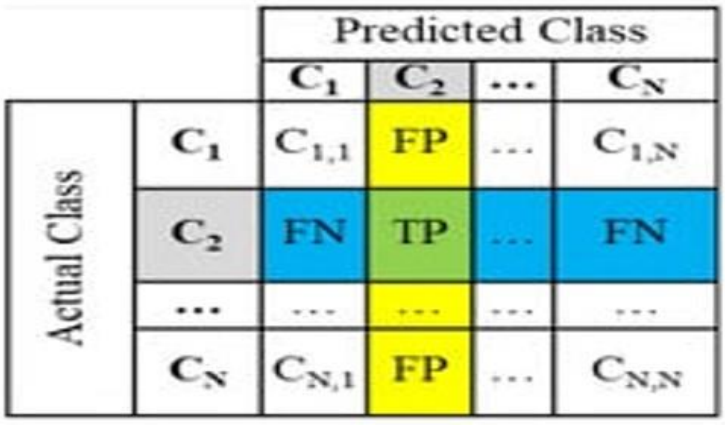

Figure 4.1: N*N confusion matrix

From figure 4.1, we can see that each cell in the matrix $C_{i,j}$ means that for a particular input, the actual class is C_i and the predicted class is C_j. We can pretty much derive any metric for a class C_k if we compute TP_k, TN_k, FP_k, and FN_k for a class using the following equations:

- $TP_k = C_{i,j}$ (i=j=k) ……..……..(x)
- $FP_k = \sum_{i=1}^{N} C_{i,j}$ (i≠k, j=k) ……(xi)
- $FN_k = \sum_{j=1}^{N} C_{i,j}$ (j≠k, i=k) ……(xii)
- $TN_k = \sum_{i=1}^{N} \sum_{j=1}^{N} C_{i,j}$ (j≠k, i≠k) ……(xiii)
- $Precision = \frac{TPk}{TPk+FPk}$ ……(xiv)
- $Recall = \frac{TPk}{TPk+FNk}$ ……..……(xv)
- $F1 = 2 * \frac{precision*recall}{precision+recall}$ ……..……..(xvi)
- $Accuracy = \frac{TPk+TNk}{TPk+TNk+FPk+FNk}$ ……..…….(xvii)

4.3 Performance of the Classifiers with Different Datasets

For building news classifier models and improving the performance of the model, we used two datasets. One of them is a pre-existing data set based on American news headlines [9]. The other dataset was created from scratch using Bangladeshi news headlines from different news websites. The following sections show the performance of each model after it was built using these datasets.

4.3.1 Performance with the American News Dataset

First, we used this news headline dataset [9] with both of our methodologies depicted in figures 3.1 and 3.2 to create classification models. This dataset has 20K news headlines and 21 categories of news. In figure 3.3, we can see the number of data points that belong to each category. After splitting the dataset into training and testing sets with a 90:10 ratio, we build our classification models. The performance figures for these models, along with precision, recall, and the f1 score of a common class in both datasets, 'ENTERTAINMENT' , are given in Table 4.2.

Table 4.2: Performance of models with the American news dataset

Model	Feature Extraction	Accuracy(%)	ENTERTAINMENT Class		
			P(%)	R(%)	F1(%)
Multinomial NB	TF-IDF Vectorization	55.3	0	0	0
Logistic Regression		70.3	79	31	45
Linear SVC		74.4	72	51	60
FFNN	Word Embedding	72	60	40	48

From Table 4.2, we can see that Linera SVC has the highest accuracy of 74.4%. On the other hand, Multinomial NB has the lowest accuracy of 55.3%. While LR and FFNN both have an accuracy of over 70%. For the same **'**ENTERTAINMENT**'** class in the dataset, linear SVC gives more balanced precision, recall, and f1-score. On the other hand, multinomial NB gives zero for precision, recall, and f1-score for the same class.

4.3.2 Performance with The Bangladeshi News Dataset

We created a dataset by gathering news from various Bangladeshi news websites. This dataset has 2K news headlines. There are 10 categories of news in the dataset. Figure 3.4 shows the frequency of every category in the dataset. This dataset has some similar categories to the previous dataset [9]. This dataset also has the same "ENTERTAINMENT" category. After splitting the dataset into training and testing sets with a 90:10 ratio, we build our classification models. The following Table 4.3 shows the performances we get by building classification

models with this dataset, along with the precision, recall, and f1-score of the 'ENTERTAINMENT' category.

Table 4.3: Performance of models with the Bangladeshi news dataset

Model	Feature Extraction	Accuracy(%)	ENTERTAINMENT Class		
			P(%)	R(%)	F1(%)
Multinomial NB	**TF-IDF Vectorization**	**72**	**87**	**65**	**74**
Logistic Regression		**70**	**58**	**70**	**64**
Linear SVC		**77**	**80**	**80**	**80**
FFNN	**Word Embedding**	**81**	**86**	**90**	**88**

From Table 4.3, we can see that FFNN has the highest accuracy of 81%. On the other hand, logistic regression has the lowest accuracy at 70%. While linear SVC has an accuracy of 77% and multinomial NB has an accuracy of over 72%. As this data set is more balanced, models built using it have much better performance. For the same 'ENTERTAINMENT' class in the dataset, linear FFNN gives the highest precision, recall, and f1-score. On the other hand, multinomial NB, which gave zero values for the previous dataset, gives much better precision, recall, and f1-score for the same class.

4.4 Individual Performance of the Classifiers

In this section, we showcase the performances of every model used in our news classification system. We can also see how these models performance varies with the two datasets that are used to train the model.

4.4.1 Multinomial Naïve Bayes

For the first news dataset [9], this classifier performed very poorly. As we can see from the classification report shown in figure 4.2 below. This model only achieved an accuracy of 55%.

	precision	recall	f1-score	support
BLACK VOICES	0.00	0.00	0.00	22
BUSINESS	0.00	0.00	0.00	41
COMEDY	1.00	0.03	0.06	34
CRIME	0.00	0.00	0.00	12
CULTURE & ARTS	0.00	0.00	0.00	20
DIVORCE	0.89	0.28	0.43	85
ENTERTAINMENT	0.00	0.00	0.00	35
ENVIRONMENT	0.00	0.00	0.00	31
FOOD & DRINK	0.90	0.29	0.44	95
HOME & LIVING	0.95	0.35	0.51	100
IMPACT	0.00	0.00	0.00	21
MONEY	0.94	0.28	0.44	102
PARENTING	0.72	0.35	0.47	193
QUEER VOICES	0.00	0.00	0.00	29
SCIENCE	0.00	0.00	0.00	22
SPORTS	0.00	0.00	0.00	11
STYLE & BEAUTY	0.57	0.92	0.70	380
TECH	1.00	0.07	0.14	27
TRAVEL	0.75	0.55	0.63	204
WEDDINGS	0.92	0.34	0.49	101
WELLNESS	0.44	0.96	0.60	454
accuracy			0.55	2019
macro avg	0.43	0.21	0.23	2019
weighted avg	0.60	0.55	0.49	2019

Figure 4.2: Classification report of Multinomial NB with American news dataset

From figure 4.2, we can see that this model gave zero and very low values to precision, recall, and the f1-score of many classes. This means this model cannot predict some classes at all for this first dataset.

The confusion matrix of this model for the American news dataset is given in the following figure 4.3:

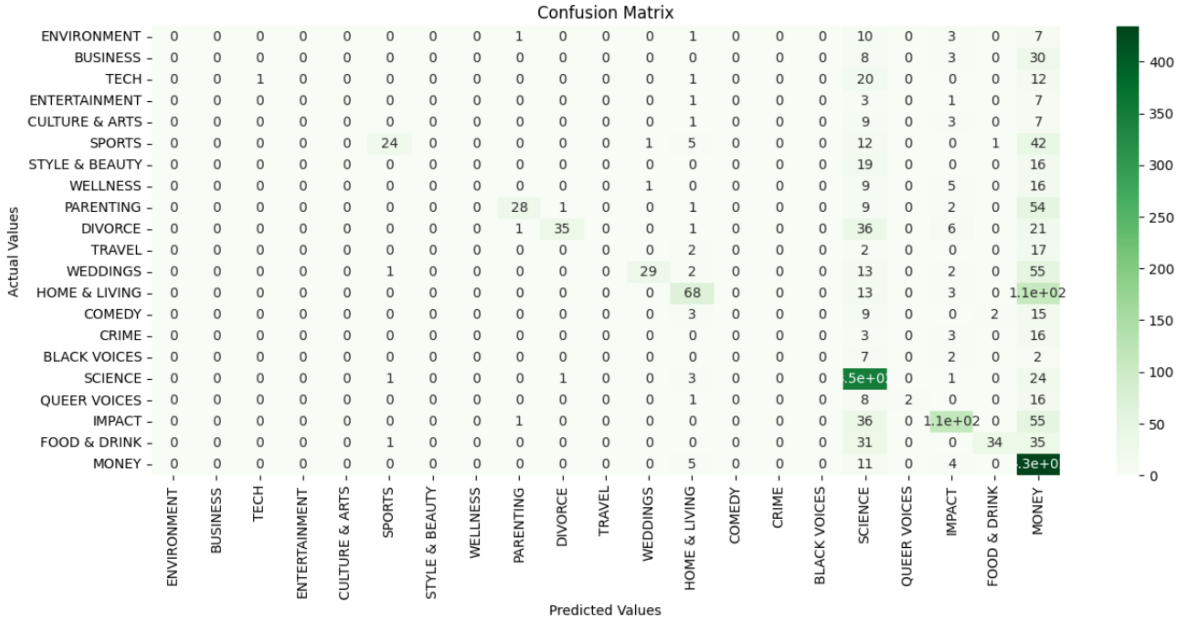

Figure 4.3: Confusion matrix of Multinomial NB with American news dataset

From the above confusion matrix, we can see why this model performs so poorly in predicting some classes.

In the following figure 4.4, we can see the classification report of Multinomial NB, which was trained using our Bangladeshi news dataset. Here we can see that the accuracy of the model has improved quite a lot from its previous accuracy. Now this model has an accuracy of 72%.

	precision	recall	f1-score	support
BUSINESS	0.60	0.90	0.72	20
ENTERTAINMENT	0.87	0.65	0.74	20
ENVIRONMENT	0.58	0.70	0.64	20
INTERNATIONAL	0.68	0.75	0.71	20
LIFESTYLE	0.81	0.65	0.72	20
NATIONAL	0.67	0.40	0.50	20
POLITICS	0.73	0.95	0.83	20
SPORTS	0.95	0.90	0.92	20
TECHNOLOGY	0.69	0.55	0.61	20
YOUTH	0.75	0.75	0.75	20
accuracy			0.72	200
macro avg	0.73	0.72	0.71	200
weighted avg	0.73	0.72	0.71	200

Figure 4.4: Classification report of Multinomial NB with proposed dataset

Now this model doesn't give any zero values to any of the precision, recall, or f1-scores of the classes. This model has the highest precision of 95% and f1-score of 92% for the "Sport" category. The following figure 4.5 shows the confusion matrix of this model after it is trained on a much more balanced dataset. This confusion has much better precision for every class in the dataset.

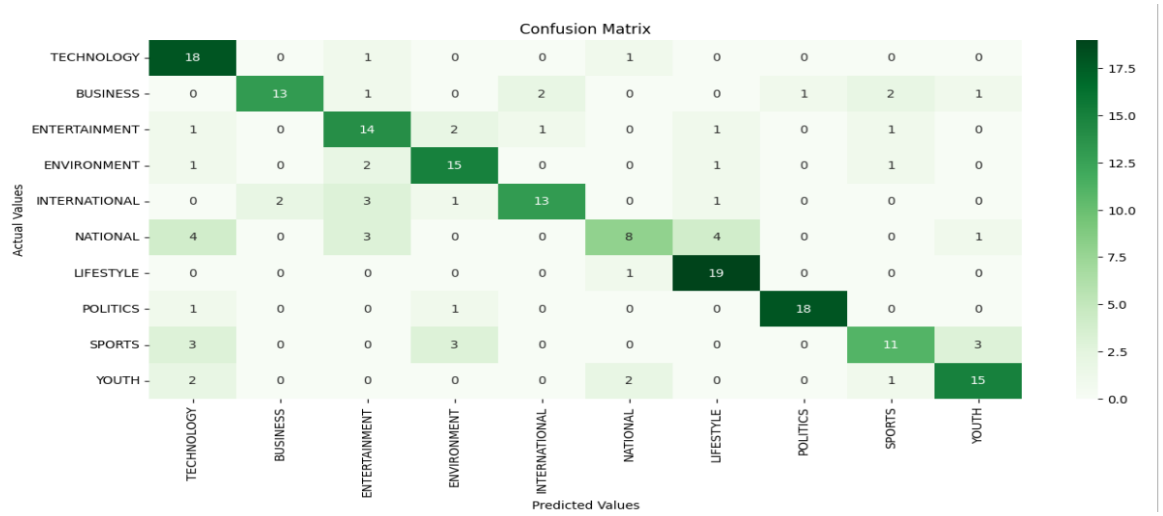

Figure 4.5: Confusion matrix of Multinomial NB with proposed dataset

4.4.2 Logistic Regression

When trained on the first dataset [9], this model had an accuracy of 70%. The classification report after building this model on the first dataset is given in the following figure:

	precision	recall	f1-score	support
BLACK VOICES	1.00	0.18	0.31	22
BUSINESS	0.88	0.17	0.29	41
COMEDY	0.85	0.50	0.63	34
CRIME	0.00	0.00	0.00	12
CULTURE & ARTS	1.00	0.15	0.26	20
DIVORCE	0.93	0.62	0.75	85
ENTERTAINMENT	0.79	0.31	0.45	35
ENVIRONMENT	0.67	0.13	0.22	31
FOOD & DRINK	0.89	0.60	0.72	95
HOME & LIVING	0.81	0.62	0.70	100
IMPACT	1.00	0.05	0.09	21
MONEY	0.74	0.60	0.66	102
PARENTING	0.69	0.64	0.66	193
QUEER VOICES	1.00	0.41	0.59	29
SCIENCE	0.00	0.00	0.00	22
SPORTS	1.00	0.36	0.53	11
STYLE & BEAUTY	0.80	0.88	0.84	380
TECH	0.92	0.44	0.60	27
TRAVEL	0.72	0.71	0.71	204
WEDDINGS	0.83	0.84	0.84	101
WELLNESS	0.57	0.93	0.70	454
accuracy			0.70	2019
macro avg	0.77	0.44	0.50	2019
weighted avg	0.73	0.70	0.68	2019

Figure 4.6: Classification report of Logistic Regression with American news dataset

Though this model performs better than the multinomial NB trained on the same dataset, it also gives zero values to the precision, recall, and f1-scores of some classes. The confusion matrix of this trained model is given in the following figure:

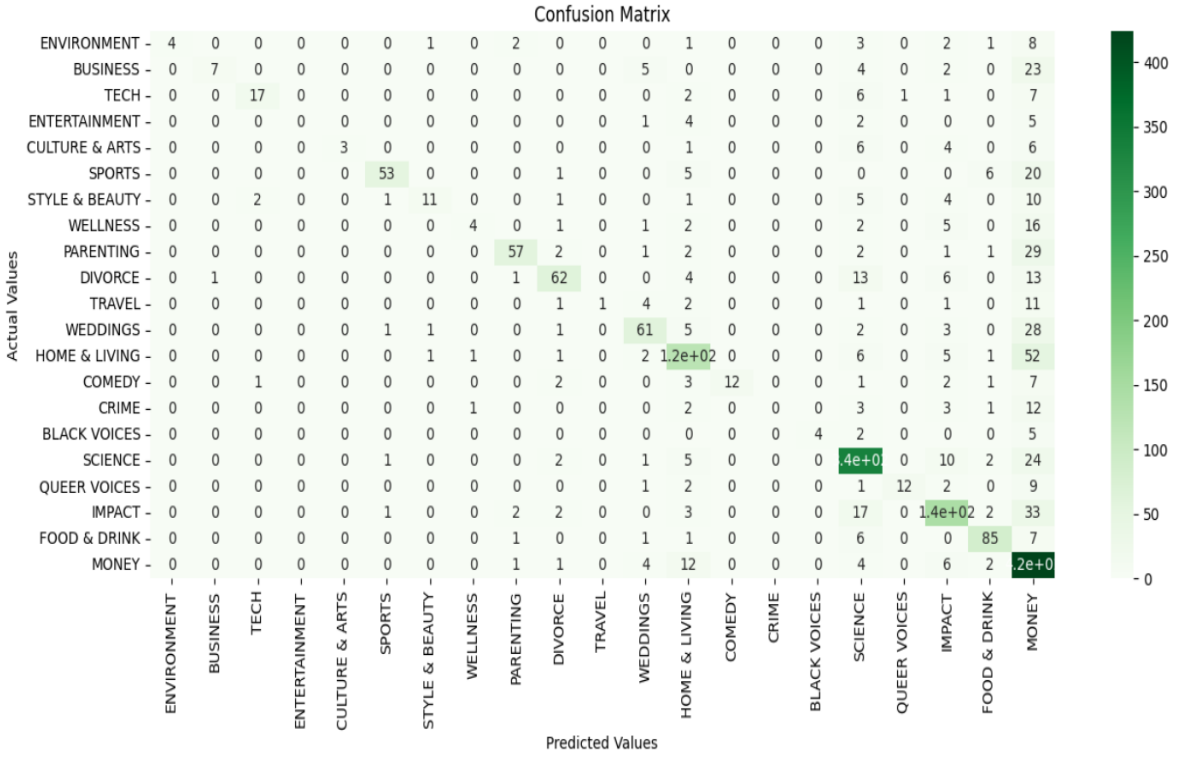

Figure 4.7: Confusion matrix of Logistic Regression with American news dataset

When we use the Bangladeshi news dataset, the LR classifier gets an accuracy score of 70%, which is the same as the previous accuracy score obtained with the other dataset. The classification report after building this model on the second dataset is given in the following figure:

	precision	recall	f1-score	support
BUSINESS	0.67	0.80	0.73	20
ENTERTAINMENT	0.58	0.70	0.64	20
ENVIRONMENT	0.68	0.75	0.71	20
INTERNATIONAL	0.67	0.70	0.68	20
LIFESTYLE	0.81	0.65	0.72	20
NATIONAL	0.54	0.35	0.42	20
POLITICS	0.89	0.85	0.87	20
SPORTS	0.86	0.90	0.88	20
TECHNOLOGY	0.58	0.55	0.56	20
YOUTH	0.76	0.80	0.78	20
accuracy			0.70	200
macro avg	0.70	0.70	0.70	200
weighted avg	0.70	0.70	0.70	200

Figure 4.8: Classification report of Logistic Regression with proposed dataset

Though the accuracy score is similar to the report in Figure 4.6, we can see that none of the classes have zeros in their precision, recall, or f1-score. The confusion matrix generated after using the second dataset is given in the following figure:

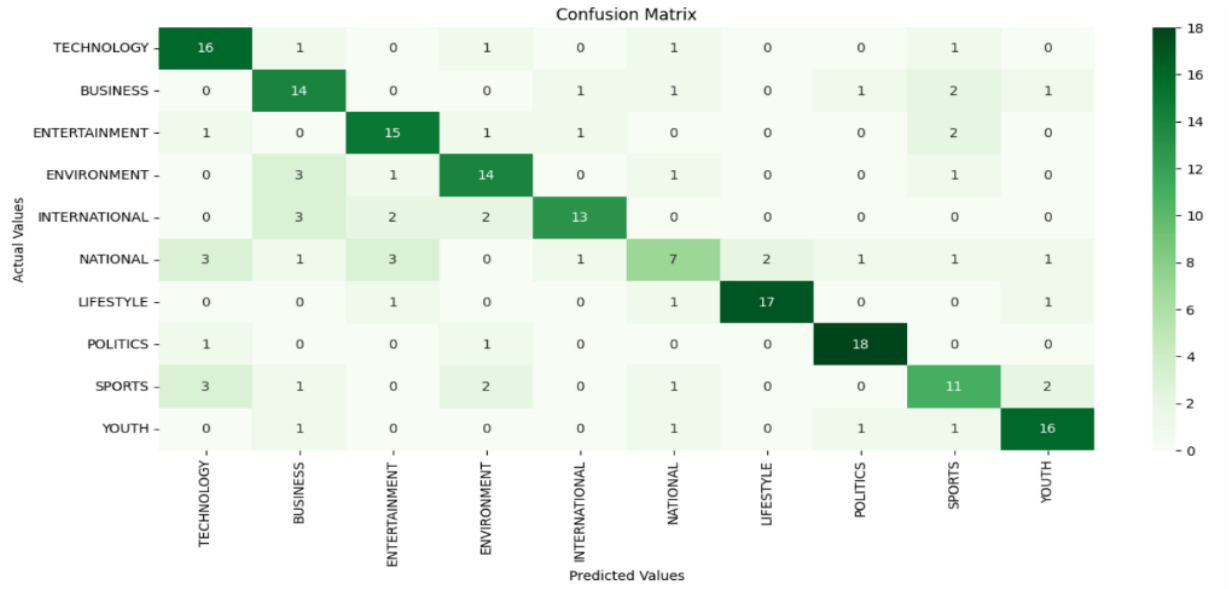

Figure 4.9: Confusion matrix of Logistic Regression with proposed dataset

This confusion matrix in figure 4.9 shows that predictions are less biased toward certain classes compared to the confusion matrix in figure 4.7.

4.4.3 Linear SVC

When trained on the first dataset [9], this model had an accuracy of 74.4%. Which is the highest score among all the models trained on this dataset [9]. The classification report after building this model on the first dataset is given in the following figure:

	precision	recall	f1-score	support
BLACK VOICES	0.83	0.23	0.36	22
BUSINESS	0.48	0.32	0.38	41
COMEDY	0.84	0.62	0.71	34
CRIME	0.67	0.50	0.57	12
CULTURE & ARTS	0.82	0.45	0.58	20
DIVORCE	0.82	0.73	0.77	85
ENTERTAINMENT	0.72	0.51	0.60	35
ENVIRONMENT	0.54	0.45	0.49	31
FOOD & DRINK	0.81	0.73	0.77	95
HOME & LIVING	0.81	0.69	0.75	100
IMPACT	0.50	0.14	0.22	21
MONEY	0.64	0.66	0.65	102
PARENTING	0.67	0.68	0.68	193
QUEER VOICES	0.74	0.59	0.65	29
SCIENCE	0.89	0.36	0.52	22
SPORTS	0.90	0.82	0.86	11
STYLE & BEAUTY	0.83	0.89	0.86	380
TECH	0.64	0.59	0.62	27
TRAVEL	0.70	0.71	0.70	204
WEDDINGS	0.85	0.85	0.85	101
WELLNESS	0.72	0.87	0.78	454
accuracy			0.74	2019
macro avg	0.73	0.59	0.64	2019
weighted avg	0.74	0.74	0.74	2019

Figure 4.10: Classification report of Linear SVC with American news dataset

From the report in Figure 4.10, we see that this model doesn't give any zero values to any of the precision, recall, or f1-scores of the classes. Thus, this model performs better than LR and multinomial NB while using the first dataset. The confusion matrix of this trained model is given in the following figure:

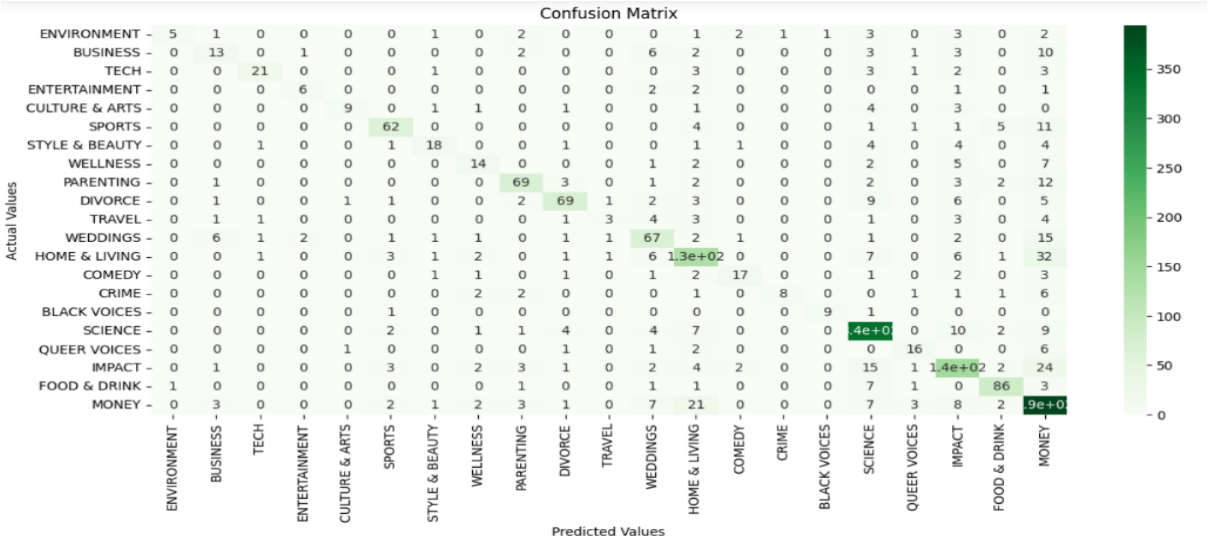

Figure 4.11: Confusion matrix of Linear SVC with American news dataset

From figure 4.11, we can see that the classifier still has biases towards some categories when making predictions. When we use the Bangladeshi news dataset to train a linear SVC classifier, it gives us an accuracy score of 77%. This is the second highest score among all the classifiers trained with that dataset. The classification report of this trained model is given in the following figure:

	precision	recall	f1-score	support
BUSINESS	0.70	0.80	0.74	20
ENTERTAINMENT	0.80	0.80	0.80	20
ENVIRONMENT	0.70	0.70	0.70	20
INTERNATIONAL	0.70	0.80	0.74	20
LIFESTYLE	0.83	0.75	0.79	20
NATIONAL	0.59	0.50	0.54	20
POLITICS	0.83	0.95	0.88	20
SPORTS	0.90	0.95	0.93	20
TECHNOLOGY	0.75	0.60	0.67	20
YOUTH	0.84	0.80	0.82	20
accuracy			0.77	200
macro avg	0.76	0.76	0.76	200
weighted avg	0.76	0.77	0.76	200

Figure 4.12: Classification report of Linear SVC with proposed dataset

This trained model in Figure 4.12 has better precision, recall, or f1-score across all the classes. The confusion matrix of this improved model is given in the following figure:

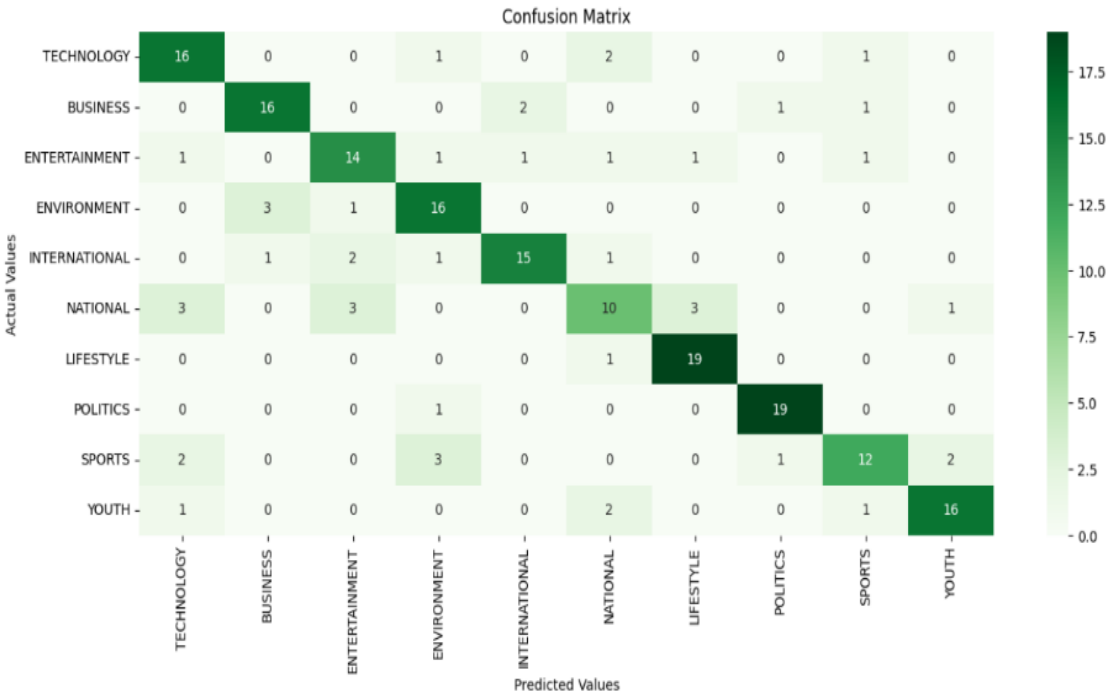

Figure 4.13: Confusion matrix of Linear SVC with proposed dataset

4.4.4 Feed Forward Neural Network

Among all the classifiers, FFNN is the only one that uses word embeddings to learn patterns of inputs and make predictions. When we trained our FFNN using the first dataset [9], this classifier gave us an accuracy score of 72%. This is the second-highest score obtained by using the first dataset [9]. The classification report after building the FFNN classifier with the help of this dataset [9] is given in the following figure:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	43
1	0.60	0.07	0.13	81
2	0.52	0.41	0.46	69
3	1.00	0.04	0.08	23
4	0.00	0.00	0.00	40
5	0.89	0.52	0.65	170
6	0.60	0.40	0.48	70
7	0.75	0.14	0.24	63
8	0.79	0.78	0.79	189
9	0.68	0.78	0.73	201
10	0.00	0.00	0.00	42
11	0.63	0.76	0.69	203
12	0.65	0.69	0.67	387
13	0.79	0.39	0.52	57
14	0.82	0.41	0.55	44
15	0.00	0.00	0.00	22
16	0.75	0.88	0.81	761
17	0.81	0.40	0.54	55
18	0.69	0.90	0.78	408
19	0.66	0.79	0.72	202
20	0.77	0.85	0.81	908
accuracy			0.72	4038
macro avg	0.59	0.44	0.46	4038
weighted avg	0.70	0.72	0.69	4038

Figure 4.14: Classification report of FFNN classifier with American news dataset

Though this model performs better than the multinomial NB trained and LR on the same dataset, it also gives zero values to the precision, recall, and f1-scores of some classes. The confusion matrix of this trained model is given in the following figure:

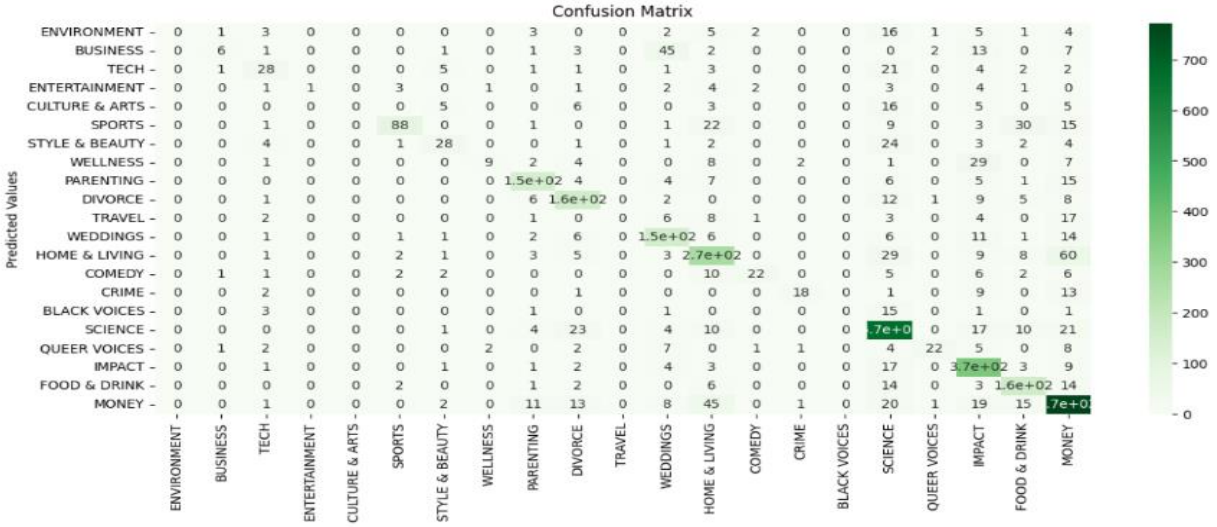

Figure 4.15: Confusion matrix of FFNN classifier with American news dataset

When we use the Bangladeshi news dataset to train our FFNN classifier, it gives us an accuracy score of 81%. Which is a good improvement from its previous performance. This is the highest score among all the classifiers trained with that dataset. The classification report of this trained model is given in the following figure:

	precision	recall	f1-score	support
BUSINESS	0.65	0.85	0.74	20
ENTERTAINMENT	0.86	0.90	0.88	20
ENVIRONMENT	0.78	0.70	0.74	20
INTERNATIONAL	0.94	0.85	0.89	20
LIFESTYLE	0.79	0.95	0.86	20
NATIONAL	0.55	0.55	0.55	20
POLITICS	0.86	0.90	0.88	20
SPORTS	1.00	0.90	0.95	20
TECHNOLOGY	0.92	0.60	0.73	20
YOUTH	0.81	0.85	0.83	20
accuracy			0.81	200
macro avg	0.82	0.80	0.80	200
weighted avg	0.82	0.81	0.80	200

Figure 4.16: Classification report of FFNN classifier with proposed dataset

From the report in Figure 4.16, we see that this classifier doesn't give any zero values to any of the precision, recall,or f1-scores of the classes. This classifier also gives higher performance metrics for multiple classes. The confusion matrix of this trained classifier is given in the following figure:

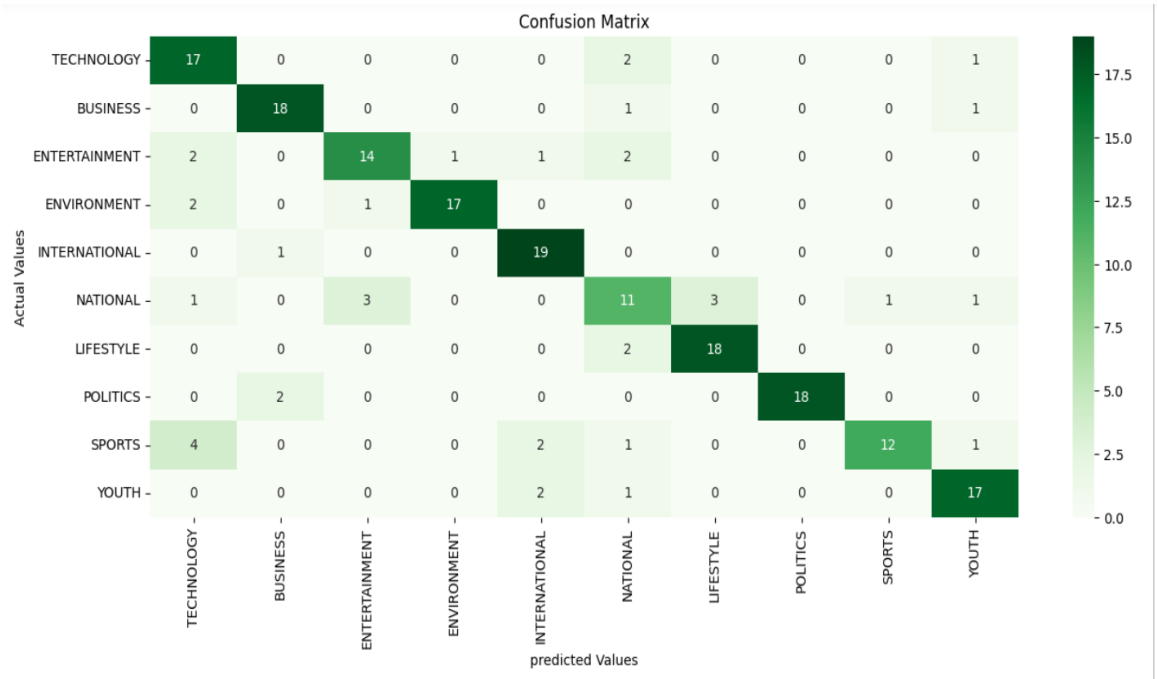

Figure 4.17: Confusion matrix of FFNN classifier with proposed dataset

4.5 Manually Testing Classifiers

For manually testing every classifier, we took a news headline from an online news website. The headline has the following text: "CPD raises flag over US turning into top remittance source.". After giving this headline to our classifiers as input, we got the outputs illustrated in figures 4.18 and 4.19.

```
news="CPD raises flag over US turning into top remittance source"
news=news.lower()
news=removePunctuations(news)
news=removePunctuations(news)
news=stemWords(news)
lsvcout=lsvc.predict([news])
lsvcout=le.inverse_transform([lsvcout])
lgrcout=lgr.predict([news])
lgrcout=le.inverse_transform([lgrcout])
mnbcout=mnb.predict([news])
mnbcout=le.inverse_transform([mnbcout])
print("According to Linear SVC news belongs to '"+lsvcout[0]+"' category.\n")
print("According to LR news belongs to '"+lgrcout[0]+"' category.\n")
print("According to Multinomial NB news belongs to '"+mnbcout[0]+"' category.\n\n")
```

```
According to Linear SVC news belongs to 'BUSINESS' category.

According to LR news belongs to 'BUSINESS' category.

According to Multinomial NB news belongs to 'BUSINESS' category.
```

Figure 4.18: Output of 3 ML classifiers

```
input_text = "CPD raises flag over US turning into top remittance source"
processed_data = prepare_data(input_text, tokenizer)
result = make_prediction(model, processed_data)
cls=np.argmax(result[0],axis=-1)
for x in d:
    if(d[x]==cls):
        print("Predicted Class is "+x)
        break
```

```
1/1 [==============================] - 2s 2s/step
Predicted Class is BUSINESS
```

Figure 4.19: Output of FFNN classifier

Chapter 5: Result Analysis

For our news headline classification task, we have implemented models with different classifiers, different preprocessing techniques, and different feature extraction techniques. We tried different approaches to find the models that gave us the most satisfactory results.

We first tried to build models using an existing dataset [9]. As this dataset [9] was unbalanced, the predictions made by the classifiers were highly biased towards the classes that were more frequent in the dataset. We can also see in Table 4.2 that Multinomial NB got an accuracy of 55.3%, while other classifiers had some respectable accuracy. But every classifier had very bad precision, recall, and f1-score for some classes due to an unbalanced dataset.

One of our goals in building a news classification model was to categorize news from various Bangladeshi news websites using their headlines. On the other hand, we needed a much more balanced dataset to properly train our models and provide acceptable classification. Thus, we created our own dataset using Bangladeshi news websites. When we used this dataset to build our models, most of the classifiers had an increase in accuracy scores. From the classification reports shown in Section 4.4, we know that when classifiers are trained using this dataset, they have better precision, recall, and f1-scores for all the classes. On the other hand, none of the classifiers had any zero value assigned to the precision, recall, or f1-score of any class while using this dataset.

From the confusion matrices shown in Section 4.4, we see that while using the improved dataset, the bias toward the majority class decreases compared to the previous dataset.

Among the three ML classifiers, Linear SVC had the best performance with an accuracy of 77%. Because Linear SVC can handle high-dimensional feature spaces efficiently. It is well suited for our news classification tasks and has a large number of features.

Among all the classifiers, FFNN performs the best. With the new dataset, it achieves 81% accuracy. This is the only classifier that uses contextual word embedding generated by the BERT model, while the other three classifiers use TF-IDF vectorization for feature extraction. Combining BERT embeddings fine-grained and contextual representations with FFNN's gradient-based optimization algorithms, such as stochastic gradient descent (SGD), to iteratively update the model parameters and minimize the prediction error leads to improved performance.

Chapter 6: Conclusion

6.1 Conclusion

Nowadays, there are a lot of news stories available online. Especially Bangladeshi news websites are becoming more popular. It is impossible to organize such large quantities of news manually. So we tried to solve this issue by applying machine and deep learning techniques through which news is categorized based on its headline content.

At the beginning, we used an existing dataset [9] to build our model. We performed various preprocessing techniques on the news data so that features could be easily extracted from it for further processing. After feature extraction, we employed different techniques for classification, including Naive Bayes, Logistic Regression, Linear SVC, and Feed-Forward Neural Networks (FFNN).

But our outcome was not satisfactory as that dataset [9] was unbalanced. To train and improve our model's performance in categorizing Bangladeshi news, we created our own dataset, which contains news headlines from different Bangladeshi news websites. After using this dataset, the performance of our model improved. The performance offered by the FFNN classifier was the most favorable.

6.2 Limitations and Future work

With our current proposed methodologies for a news classification model, we achieved a maximum of 81% accuracy. Thus, there is a lot of room for improvement. Our future plan is to find some techniques to improve the performance of the classifiers.

We would like to improve our own dataset by increasing the volume of data and including larger news headlines, which may contain more information for classification.

Finally, after we reach a standard level of performance, we would like to incorporate our news classification model with some other model, such as a fake news detection model, to perform other types of classification on a particular type of news.

References

[1] K. G, R. SB and S. KG, "A new text mining approach based on HMM-SVM for web news classification.," *International Journal of Computer Applications,* vol. 1, no. 19, pp. 98-104, 2010.

[2] R. B. Kumar, B. S. Kumar and C. S. S. Prasad, "financial news classification using SVM," *International Journal of Scientific and Research Publications,* vol. 2, no. 3, pp. 1-6, 2012.

[3] M. . A. Zaveri, and M. K. Dalal, "Automatic text classification:A technical review," *International Journal of Computer Applications,* vol. 28, no. 2, pp. 37-40, 2011.

[4] D. S. M. Hossein , M. S. Araghi and M. M. Farah, "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification.," *2016 IEEE International Conference on Engineering and Technology (ICETECH),* pp. 112-116, 2016.

[5] W. T. Chu and H. Y. Chang, "Advertisement Detection, Segmentation, and Classification for Newspaper Images and Website Snapshots," *2016 International Computer Symposium (ICS),* pp. 396-401, 2016.

[6] D. Rahmawati and M. L. Khodra, "Word2vec semantic representation in multilabel classification for Indonesian news article," *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA),,* pp. 1-6, 2016.

[7] R. M. I., . K. S. and A. M. U., "News classification based on their headlines: A review.," *17th IEEE International Multi Topic Conference 2014,* pp. 211-216, 2014.

[8] Z. . S. A. Ali and S. M. Hassan, "researchgate," December 2018. [Online]. Available: https://www.researchgate.net/publication/339029343_UrduHindi_News_Headline_Text_Classification_by_Using_Different_Machine_Learning_Algorithms. [Accessed 27 February 2023].

[9] R. MISRA, "News Category Dataset," kaggle, [Online]. Available: https://www.kaggle.com/datasets/rmisra/news-category-dataset. [Accessed 25 February 2023].

[10] R. Vickery, "An Introduction to Preprocessing Data for Machine Learning," Towards Data Science, 30 August 2022. [Online]. Available: https://towardsdatascience.com/an-introduction-to-preprocessing-data-for-machine-learning-8325427f07ab. [Accessed 27 January 2023].

[11] "Data Preprocessing in Machine learning," javatpoint, [Online]. Available: https://www.javatpoint.com/data-preprocessing-machine-learning. [Accessed 27 January 2023].

[12] Pankaj, "How To Use Python pandas dropna() to Drop NA Values from DataFrame," DigitalOcean, 3 August 2022. [Online]. Available: https://www.digitalocean.com/community/tutorials/pandas-dropna-drop-null-na-values-from-dataframe. [Accessed 27 January 2023].

[13] K. Rastogi, "Text Cleaning Methods in NLP," analyticsvidhya.com, 22 November 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/01/text-cleaning-methods-in-nlp/. [Accessed 29 January 2022].

[14] S. Singh, "NLP Essentials: Removing Stopwords and Performing Text Normalization using NLTK and spaCy in Python," analyticsvidhya, 21 August 2019. [Online]. Available:

https://www.analyticsvidhya.com/blog/2019/08/how-to-remove-stopwords-text-normalization-nltk-spacy-gensim-python/. [Accessed 30 January 2023].

[15] bistpratham, "How to convert categorical string data into numeric in Python?," geeksforgeeks.org, [Online]. Available: https://www.geeksforgeeks.org/how-to-convert-categorical-string-data-into-numeric-in-python/. [Accessed 30 January 2023].

[16] manmayi, "Python Keras | keras.utils.to_categorical()," geeksforgeeks.org, [Online]. Available: https://www.geeksforgeeks.org/python-keras-keras-utils-to_categorical/. [Accessed 31 January 2023].

[17] I. Bernardo, "Stemming Text with NLTK," 3 May 2021. [Online]. Available: https://towardsdatascience.com/stemming-corpus-with-nltk-7a6a6d02d3e5. [Accessed 2 February 2023].

[18] T. PyCoach, " 5 Simple Ways to Tokenize Text in Python," towardsdatascience.com, 14 March 2021. [Online]. Available: https://towardsdatascience.com/5-simple-ways-to-tokenize-text-in-python-92c6804edfc4. [Accessed 3 February 2023].

[19] P. Prakash, "An Explanatory Guide to BERT Tokenizer," www.analyticsvidhya.com, 9 September 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/09/an-explanatory-guide-to-bert-tokenizer/. [Accessed 22 February 2023].

[20] "Feature Extraction," deepai.org, [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/A-extraction. [Accessed 15 February 2023].

[21] V. Jayaswal, "Text Vectorization: Term Frequency — Inverse Document Frequency (TFIDF)," towardsdatascience.com, 4 October 2020. [Online]. Available: https://towardsdatascience.com/text-vectorization-term-frequency-inverse-document-frequency-tfidf-5a3f9604da6d. [Accessed 27 February 2023].

[22] J. Brownlee, "What Are Word Embeddings for Text?," machinelearningmastery.com, 11 October 2017. [Online]. Available: https://machinelearningmastery.com/what-are-word-embeddings/. [Accessed 2 March 2023].

[23] A. Pogiatzis, "NLP: Contextualized word embeddings from BERT," Towards Data Science, 20 March 2019. [Online]. Available: https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b. [Accessed 10 March 2023].

[24] P. S, "The A-Z guide to Support Vector Machine," analyticsvidhya, 16 June 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/support-vector-machine-better-understanding/. [Accessed 15 March 2023].

[25] baeldung, "Multiclass Classification Using Support Vector Machines," 11 November 2022. [Online]. Available: https://www.baeldung.com/cs/svm-multiclass-classification. [Accessed 16 March 2023].

[26] A. V. Ratz, "Multinomial Naïve Bayes' For Documents Classification and Natural Language Processing (NLP)," towardsdatascience, 17 March 2021. [Online]. Available: https://towardsdatascience.com/multinomial-na%C3%AFve-bayes-for-documents-classification-and-natural-language-processing-nlp-e08cc848ce6. [Accessed 20 March 2023].

[27] V. Kanade, "What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices," spiceworks, 18 April 2022. [Online]. Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/. [Accessed 23 March 2023].

[28] R. N. Sucky, "Multiclass Classification Using Logistic Regression from Scratch in Python: Step by Step Guide," towardsdatascience, 5 September 2020. [Online]. Available: https://towardsdatascience.com/multiclass-classification-algorithm-from-scratch-with-a-project-in-python-step-by-step-guide-485a83c79992. [Accessed 30 March 2023].

[29] S. Chandak, "BERT for Multi-class text classification," medium, 19 July 2019. [Online]. Available: https://medium.com/@chandaksumit29_15695/bert-for-multi-class-text-classification-12b66a1fc01c. [Accessed 31 March 2023].

[30] O. Knocklein, "Classification Using Neural Networks," owardsdatascience, 6 July 2019. [Online]. Available: https://towardsdatascience.com/classification-using-neural-networks-b8e98f3a904f. [Accessed 31 March 2023].

[31] A. Suresh, "What is a confusion matrix?," medium, 17 November 2020. [Online]. Available: https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5. [Accessed 10 March 2023].