JADAVPUR UNIVERSITY

Project Report on

"Bengali Text Classification Using ANN"

BY

Dilip Barman

EXAMINATION ROLL NO : MCA2360031

UNDER THE SUPERVISION OF

PROF. (DR.) KAMAL SARKAR

PROJECT SUBMMITION OF FULFILLMENT FOR THE DEGREE OF

MASTER OF COMPUTER APPLICATION

IN THE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING AND TECHNOLOGY

2023

Certificate of Recommendation

This is to certify that the thesis entitled " Bengali Text Classification Using ANN" has been carried out by Dilip Barman (University Roll Number : 002010503045, Examination Roll No MCA2360031, University Registration Number : 154253 of 2020-2021), under the guidance and supervision of Prof. (Dr.) Kamal Sarkar, Department of Computer Science and Technology, Jadavpur University, Kolkata, is being presented for the partial fulfillment of the Degree of Master of Computer Applications during the academic year 2022-2023. The research results presented in the thesis have not been included in any other paper submitted for the award of any degree in any other university or institute.

Prof. (Dr.) Kamal Sarkar(Project Supervisor)
Professor, Dept. of Comp. Science & Engineering

Jadavpur University, Kolkata-700032

Prof. Dr. Nandini Mukhopadhyay

Head of the Department, Dept. of Comp. Science & Engineering

Jadavpur University, Kolkata-700032

Prof. Ardhendu Ghoshal

Dean, Faculty Council of Engineering & Technology

Jadavpur University, Kolkata-700032

Jadavpur University
Faculty of Engineering and Technology
Department of Computer Science and Engineering

CERTIFICATE OF APPROVAL

This is to clarify that the project entitled "**Bengali Text Classification Using ANN**" has been completed by **Dilip Barman.** This work is applied under the supervision of **Prof. Kamal Sarkar** in partial fulfillment for the award of the degree of **Master of Computer Applications** of the **Department of Computer Science and Engineering**, **Jadavpur University**, during the academic year **2022-2023**. The project report has been approved because it satisfies the tutorial requirements in respect of project work prescribed for the said degree.

……………………………………………………
Signature of Examiner 1

Date:

……………………………………………………
Signature of Examiner 2

Date:

Date:

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Declaration of Originality and Compliance of Academic Ethics

I hereby declare that this thesis entitled "Bengali Text Classification Using ANN" contains original research work by the undersigned candidate, as part of his degree of Master of Computer Applications.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work

Name : Dilip Barman

Roll No. : 002010503045

Examination Roll No. : MCA2360031

University Registration No. : 154253 of 2020-2021

Project Title: *"Bengali Text Classification Using ANN"*

____
(Signature of the Candidate)

ACKNOWLEDGMENT

I am delighted to convey my gratitude to my thesis supervisor Prof. (Dr.) Kamal Sarkar , Department of Computer Science & Engineering, my supervisor, for his overwhelming support and encouragement towards accomplishment throughout the duration of the project without which this work would not have been possible. His positive outlook and confidence inspired me and gave me confidence. I am grateful for his support, encouragement, and his valuable suggestions to complete this work. I feel deeply honored that I got this opportunity to work under him.

I would like to express my sincere thanks to all my teachers for providing a sound knowledge base and cooperation.

I would like to thank all the faculty members of the Department of Computer Science & Engineering of Jadavpur University for their continuous support. Last, but not the least, I would like to thank my batch mates for staying by my side when I need them.

____

Dilip Barman

Roll No: 002010503045

Reg. No. : 154253 of 2020-2021

Examination Roll No. : MCA2360031

Abstract

Bengali text classification is a challenging task due to the unique characteristics of the Bengali language. In recent years, artificial neural networks have emerged as powerful models for natural language processing tasks, including text classification. This study explores the application of artificial neural networks for Bengali text classification and investigates their performance in accurately categorizing Bengali text documents.

The research begins with the collection of a labeled dataset consisting of Bengali text documents from diverse domains. The dataset is preprocessed by tokenizing the text, removing stopwords, and applying stemming to reduce noise and enhance feature representation. Various neural network architectures, such as feedforward neural networks, recurrent neural networks (RNNs), or convolutional neural networks (CNNs), are implemented and trained using the preprocessed dataset.

During the training phase, the neural network models learn to extract relevant features and capture the underlying patterns and relationships within the Bengali text. The models are optimized using appropriate loss functions and backpropagation algorithms to minimize classification errors and improve accuracy. Hyperparameter tuning and regularization techniques are employed to prevent overfitting and enhance generalization performance.

The trained neural network models are evaluated using standard evaluation metrics, including accuracy. A separate validation or testing dataset is used to assess the models' performance on unseen Bengali text instances. The results of the experiments provide insights into the effectiveness of artificial neural networks for Bengali text classification and comparisons with other traditional machine learning algorithms.

Overall, this research contributes to the advancement of Bengali text classification using artificial neural networks, providing valuable insights and guidelines for developing robust and accurate models for automated categorization of Bengali text documents across various domains.

ANNs can effectively classify Bengali text documents into predefined categories. The findings highlight the importance of selecting appropriate network architectures, preprocessing techniques to achieve optimal performance in Bengali text classification.

The successful implementation of ANN-based Bengali text classification can have significant applications in various domains like topic categorization, document classification, and information retrieval in the Bengali language. It enables the automation of text analysis tasks, thereby facilitating efficient handling and extraction of valuable insights from large volumes of Bengali textual data.

CONTENTS

Chapter 1

Introduction

Text classification is an emerging research area in the field of Natural Language Processing (NLP), where each document needs to be assigned to a predefined class or category. In recent years, categorization of Bengali text documents is treated as a significant research issue in the domain of Bengali Language Processing (BLP). Bengali is the 7th most widely spoken language in the world. Two hundred forty-five million people in Bangladesh and some parts of India speak in this language. In recent years, the digital Bengali text contents have been growing readily on the Internet, news portals, blogs, websites, and so on due to the effortless usage of electronic gadgets. These are creating an enormous amount of unstructured data. Therefore, it is a challenging task to organize, search or manipulate such a massive amount of unstructured data by human experts manually. This manual process consumes an immense amount of time incurring the cost of money. A document categorization system can handle a massive amount of Bengali text data in which documents can be sorted, manipulated and organized expeditiously and competently.

There are no efficient tools available that have been developed till today for Bengali language processing (BLP). Therefore, there is an insistence of developing intelligent tools for BLP so that professionals, as well as the common people, can use these tools to their needs. However, developing an *intelligent* system that can categorize Bengali text documents in distinct categories is also an important research issue leading to the most use in real-world applications. A text categorization system can be used in information retrieval, text mining, and text analytic.

The proposed model can be used by the hospital management to categorize patient's reports according to their diagnosis, an e-library to arrange books according to subject categories and news agencies to classify the news based on text contents.

The key difficulty of machine learning techniques is to select useful features from high dimensional feature spaces. Moreover, a common Algorithm for selecting appropriate features is not available that can be applied to all kinds of classification task.

Artificial Neural Network (ANN) technique has been used extensively to solve complex problems in many domains, including text document categorization and NLP. Surprisingly, the potential of the deep learning techniques has not been investigated for Bengali document classification. In comparison with other machine learning-based techniques, deep learning techniques are more competent to discover complex features inherently, when the feature dimension is high. In addition to that, the deep learning is faster than the other machine learning techniques due to its usage of Graphics Processing Units (GPUs).

Bengali is a low-resource language, and no well-performed feature extractor and classification frameworks are available in Bengali. Moreover, there are huge structural variations between Bengali and English languages (or other high-resource languages). As a result, the embedding/classification models developed for one language cannot be directly applicable to another language due to their linguistic divergences. The hyperparameters of embedding and classification models must be tuned a fresh based on the language itself before performing the classification task. The hyperparameters of embedding and classification models must be newly tuned based on the language itself before performing the classification task. In order to achieve a reasonable accuracy, this model removes the Bengali stop-words, punctuation.

Chapter 2

Literature survey

This article is a literature review of various studies related to text classification approaches; therefore, this section elucidates some of the research directions observed in this regard. Statistical topic modeling is applied for multi-label document classification, where each document gets assigned to one or more classes. It became an interesting topic in the past decade as it performed well for datasets with increasing number of instances for an entity (Rubin, Chambers, Smyth, & Steyvers, 2012).

When the number of documents increased, the computational complexity also increased (Stas, Juhar,& Hladek, 2014). ML is often seen as an offshoot of statistics as far as data mining is concerned. Itemploys advanced models to make decisions based on its own cognizance (Du, 2017; Ranjan & Prasad, 2017). However, a purely statistical and purely ML approach is considered less competent, therefore a hybrid approach is usually preferred (Srivastava, 2015).

Artificial Immune System (AIS) based self-adaptive attribute weighting method for Naive Bayes classification uses immunity theory in Artificial Immune Systems to search optimal attribute weight values (Wu et al., 2015). Logistic regression is an efficient probability-based linear classifier. The problem of overfitting (data model memorizes the dataset instead of the learning procedure.) could besolved by using penalized logistic regression in active learning algorithm (Wang & Park, 2017).

A proper instance selection technique could finish half of the knowledge discovery procedure. A new instance selector based on Support Vector Machine (SVM) called, support vector oriented instance selection is suggested to remove noisy data (Tsai & Chang, 2013). Some researchers analyzed the decision trees' role in multi-valued and multi-labeled data. This type of data makes it difficult to pick a particular set of attributes. It is also difficult to calculate similarity scores multi-valued and multi-labeled data (Yi, Lu, & Liu, 2011).

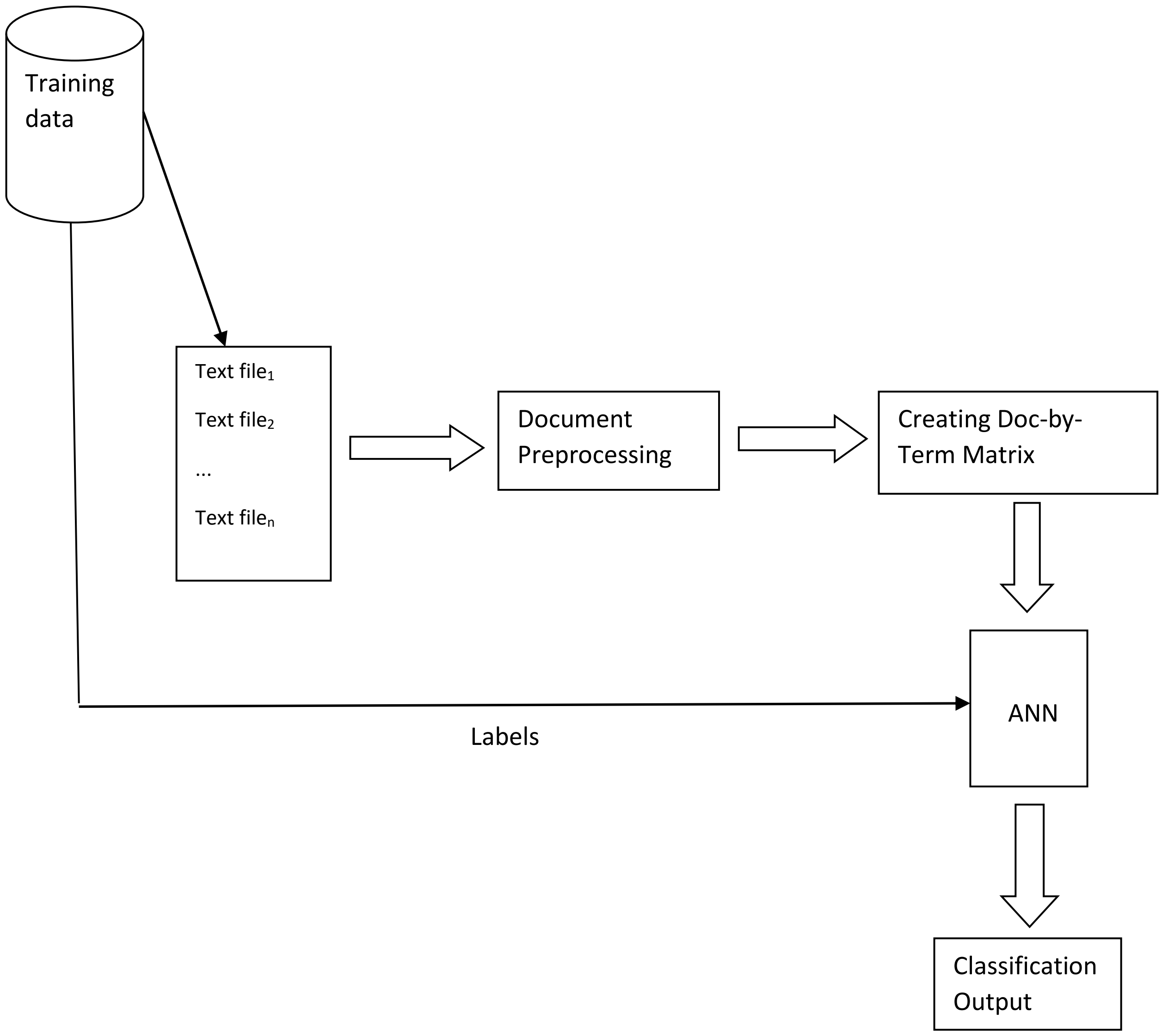
The decision tree algorithms calculate similarity scores comprehensively and accurately. It has been proven efficient for scenarios where synchronization among elements is less. To overcome the problem from the order of classes in rule learning, Complexity-based Parallel Rule Learning algorithm is suggested (Asadi & Shahrabi, 2016). In a different setting, multi-class classification is tried by com-bining kernel density estimation with k-NN (Tang & Xu, 2016). It improves the weighting principleof k-NN, thereby increasing the accuracy of classification. It has also been proven efficient for complex classification problems.

The role of ANNs in high dimensional and large data is significant. Neural classifiers such as fuzzy adaptive resonance associative maps are scalable for large volumes of data (Benites & Sapozhnikova, 2017). Unsupervised learning provides so many research opportunities in workflow management and task scheduling, particularly in the field of big data (Zhoua, Pana, Wanga, Athanasios, &Vasilakos,2017).

Chapter 3

Methodology

The method of the model is shown in follows:



3.1 Data Preprocessing:

In this section we preprocessed data by implementing tokenizing, punctuation removal and stop-word remove.

Before data preprocessed the data is shown in the given below:

[হিন্দু ধর্মে ১৮টি মহাপুরাণের উল্লেখ রয়েছে। এর মধ্যে অন্যতম হল গরুড় পুরাণ। সাধারণত পরিবারে কোনও ব্যক্তির মৃত্যু হলে গরুড় পুরাণ পাঠ করা হয়ে থাকে। গরুড় পুরাণে মৃত্যুর পর ব্যক্তির যাত্রা পথ সম্পর্কে বর্ণনা করা হয়েছে।]

After creating tokens the text data is shown in the given below:

```
['হিন্দু', 'ধর্মে', '১৮টি', 'মহাপুরাণের', 'উল্লেখ', 'রয়েছে।', 'এর',
'মধ্যে', 'অন্যতম', 'হল', 'গরুড়', 'পুরাণ।', 'সাধারণত', 'পরিবারে',
'কোনও', 'ব্যক্তির', 'মৃত্যু', 'হলে', 'গরুড়', 'পুরাণ', 'পাঠ', 'করা',
'হয়ে', 'থাকে।', 'গরুড়', 'পুরাণে', 'মৃত্যুর', 'পর', 'ব্যক্তির', 'যাত্রা',
'পথ', 'সম্পর্কে', 'বর্ণনা', 'করা', 'হয়েছে।',]
```

After removing punctuation the text data is shown in the given below:

```
['হিন্দু', 'ধর্মে', '১৮টি', 'মহাপুরাণের', 'উল্লেখ', 'রয়েছে', '', 'এর',
'মধ্যে', 'অন্যতম', 'হল', 'গরুড়', 'পুরাণ', '', 'সাধারণত', 'পরিবারে',
'কোনও', 'ব্যক্তির', 'মৃত্যু', 'হলে', 'গরুড়', 'পুরাণ', 'পাঠ', 'করা',
'হয়ে', 'থাকে', '', 'গরুড়', 'পুরাণে', 'মৃত্যুর', 'পর', 'ব্যক্তির',
'যাত্রা', 'পথ', 'সম্পর্কে', 'বর্ণনা', 'করা', 'হয়েছে',]
```

After removing stop-word the text data is shown in the given below:

```
['হিন্দু', 'ধর্মে', '১৮টি', 'মহাপুরাণের', 'উল্লেখ', 'অন্যতম', 'গরুড়',
'পুরাণ', 'সাধারণত', 'পরিবারে', 'ব্যক্তির', 'মৃত্যু', 'গরুড়', 'পুরাণ',
'পাঠ', 'গরুড়', 'পুরাণে', 'মৃত্যুর', 'ব্যক্তির', 'যাত্রা', 'পথ', 'সম্পর্কে',
'বর্ণনা',]
```

3.2 DOCUMENT REPRESENTATION

Creating Doc by Term Matrix

The Tokenizer class of Keras is used for vectorizing a text corpus. By applying this, each text input is converted into a vector .

The useful method of tokenizer class is **texts_to_matrix()** function for converting the document into a numpy matrix form.

This function works in 4 different modes –

- binary : The default value that tells us about the presence of each word in a document.
- count : As the name suggests, the count for each word in the document is known.
- tfidf : The TF-IDF score for each word in the document.
- freq : The frequency tells us about ratio of words in each document.

3.2.1 Type 1: texts_to_matrix with mode = binary

The binary mode in **texts_to_matrix()** function determines the presence of text by using ‘1’ in the matrix where the word is present and ‘0’ where the word is not present.

This mode doesn’t count the total number of times a particular word or text, but it just tells about the presence of word in each of the documents.

Doc = 'কেন্দ্রীয় সরকারের তিনটি কৃষি আইন কৃষক বিরোধী জনস্বার্থও বিরোধী আইন শুধুমাত্র পুঁজিপতিদের জন্যই মোদী সরকার অভিযোগ কেন্দ্রের কৃষি আইন বাতিলের দাবি জানাল তৃণমূল বৃহস্পতিবার তৃণমূল ভবনে সাংবাদিক বৈঠকে কেন্দ্রের কৃষি আইনের সমালোচনা তৃণমূল সাংসদ কাকলি ঘোষ দস্তিদার কেন্দ্রের কৃষি আইন কৃষক বিরোধী জনসাধারণ বিরোধীও আইনের মানুষের দৈনন্দিন জীবনে ব্যবহৃত অত্যাবশকীয় পণ্য চাল ডাল আলু পেঁয়াজকে আনা একদিকে মজুতদারদের হাত শক্ত তেমনি ওইসব দ্রব্যের কৃত্রিম চাহিদা তৈরি দাম নির্ধারিত মজুতদাররা কৃষকরা যথার্থ ফসলের দাম পাবে মূল্যবৃদ্ধির কারণে ধনী মধ্যবিত্ত দরিদ্ররাও ক্ষতিগ্রস্ত কৃষকদের বিজেপি সরকারের দায়বদ্ধতা দায়বদ্ধতা কেবল পুঁজিপতিদের দাবি কাকলি জানান আইনের কৃষি কৃষক পরিবারগুলি ক্ষতিগ্রস্ত ন্যূনতম সহায়ক মূল্যের কৃষকদের স্বার্থে স্বামীনাথন কমিটির রিপোর্ট মানেনি মোদী সরকার উপকৃত শুধুমাত্র পুঁজিপতিরা স্বার্থে পুঁজিপতিরাই আইন তৈরি বলেও আশঙ্কা প্রকাশ কৃষি আইন মধ্যস্থতার মঙ্গলবার কমিটি গঠন সুপ্রিম কোর্ট সুপ্রিম কোর্টের সিদ্ধান্তের সরাসরি বিরোধিতা

তিনটি কৃষি আইন বাতিলের দাবি কাকলি কথায় সুপ্রিম কোর্টের রায়কে মাথায় রেখেই মনে কৃষি কৃষক পরিবারের স্বার্থে ভারত সরকারের আইনগুলো প্রত্যাহার নেওয়া সংসদে আলোচনা ছাড়াই আইন আনা অভিযোগ বারাসতের সাংসদ অভিযোগ সংসদীয় রীতি নীতি উপেক্ষা শুধুমাত্র অধ্যাদেশ এনে অগণতান্ত্রিকভাবে কৃষি আইন পাশ বিজেপি সরকার ',

The above document is a content of a file. Then we apply tokenizer and text_to_matrix method to create a row of the doc-by-term matrix.

The matrix after applying texts_to_matrix with mode = binary:

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

```
<class 'numpy.ndarray'>
```

3.2.2 Type 2: texts_to_matrix with mode = count

The count mode in texts_to_matrix() function determines the number of times the words appear in each of the documents.

The matrix after applying texts_to_matrix with mode = count:

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

[0. 0. 0. ... 0. 0. 0.]
[0. 0. 0. ... 0. 0. 0.]
...
[0. 0. 3. ... 0. 0. 0.]
[0. 1. 0. ... 0. 0. 0.]
[0. 0. 0. ... 1. 1. 1.]]
```

```
<class 'numpy.ndarray'>
```

3.2.3 Type 3: **Vectorize the text data using TF-IDF**

TF-IDF or Term Frequency – Inverse Document Frequency, works by checking the relevance of a word in a given text corpus.

In this mode, a proportional score is given to words on the basis of the number of times they occur in the text corpus. In this way, this model can determine which words are worthy and which aren't.

Term Frequency (TF) is a numerical representation that indicates the frequency of a term (word) within a document or a collection of documents. It is commonly used in text mining, natural language processing, and information retrieval.

The most basic formula for calculating Term Frequency is:

TF = (Number of times a term appears in a document) / (Total number of terms in the document)

The purpose of TF is to highlight the importance or relevance of a term within a document. A higher TF value indicates that the term appears more frequently, potentially making it more significant in the context of the document.

Inverse Document Frequency (IDF) is a numerical representation used to measure the significance of a term within a collection of documents. IDF is commonly used in information retrieval, text mining, and natural language processing tasks.

The IDF of a term is calculated based on the logarithm of the ratio between the total number of documents in the collection and the number of documents that contain the term.

The formula for IDF is as follows:

$$IDF = \mathbf{log}(\frac{\textbf{Total number of documents}}{\textbf{Number of documents containing the term}})$$

The IDF value is higher for terms that appear in fewer documents in the collection, indicating that they are more unique or rare. Conversely, terms that appear in many documents have lower IDF values, as they are considered to be less informative or distinctive.

TF-IDF is a numerical representation used to evaluate the importance of a term within a document in the context of a collection of documents. TF-IDF is widely used in information retrieval, text mining, and natural language processing tasks.

The TF-IDF formula is the product of the TF and IDF values for a given term within a document. It can be represented as:

TF-IDF (t, d) = TF (t, d) * IDF (t)

The matrix after applying texts_to_matrix with mode = tfidf:

```
[[0.         0.         0.         ... 0.         0.
  0.        ]
 [0.         0.         0.         ... 0.         0.
  0.        ]
 [0.         0.         0.         ... 0.         0.
  0.        ]
 ...
 [0.         0.         2.43509931 ... 0.         0.
  0.        ]
 [0.         1.03894108 0.         ... 0.         0.
  0.        ]
 [0.         0.         0.         ... 6.77821591
  6.77821591 6.77821591]]
```

```
<class 'numpy.ndarray'>
```

3.2.4 Type 4: texts_to_matrix with mode = freq

This last mode used in texts_to_matrix() is the frequency that actually determines a score and assigning to each on the basis of the ratio of the word with all the words in the document or text corpus.

The matrix after applying texts_to_matrix with mode = freq:

```
[[0.          0.          0.          ... 0.          0.
0.        ]
[0.         0.          0.          ... 0.          0.
0.        ]
[0.         0.          0.          ... 0.          0.
0.        ]
...
[0.         0.          0.01239669 ... 0.          0.
0.        ]
[0.         0.00636943 0.          ... 0.          0.
0.        ]
[0.          0.           0.           ... 0.00363636
0.00363636 0.00363636]]

<class 'numpy.ndarray'>
```

3.3 Classifier:

An Artificial Neural Network (ANN) classifier is a machine learning model that uses a neural network architecture to perform classification tasks. It is specifically designed to learn from labeled training data and make predictions or assign class labels to unseen or future instances.

The ANN classifier consists of multiple interconnected layers of artificial neurons, which are inspired by the structure and functioning of biological neurons in the human brain. The neurons in each layer receive inputs, perform calculations, and pass the results to the next layer until the final output layer is reached.

The training process of an ANN classifier involves feeding labeled training data to the network, adjusting the weights and biases of the neurons through an optimization algorithm (e.g., backpropagation), and iteratively updating the network's parameters to minimize the prediction errors. This allows the network to learn the patterns and relationships in the training data, enabling it to classify unseen instances.

There are various types of ANN classifiers, including:

1. Feedforward Neural Networks: These are the most common type of ANN classifiers, where the information flows only in one direction, from the input layer to the output layer. They can have one or more hidden layers between the input and output layers.

2. Recurrent Neural Networks (RNNs): RNNs have connections between neurons that form directed cycles, allowing them to capture temporal dependencies in sequential data. They are suitable for tasks such as natural language processing and speech recognition.

3. Convolutional Neural Networks (CNNs): CNNs are primarily used for image classification tasks, where they perform operations such as convolution and pooling to extract features from input images. They are effective in capturing spatial patterns and structures.

4. Radial Basis Function Networks (RBFNs): RBFNs use radial basis functions as activation functions and are commonly used for pattern recognition and function approximation tasks.

ANN classifiers have shown great success in various applications, including image recognition, text classification, sentiment analysis, fraud detection, and medical diagnosis. They are capable of learning complex patterns and nonlinear relationships in the data, making them powerful tools for solving classification problems in machine learning.

The MLP classifier, also known as the Multilayer Perceptron classifier, is a type of artificial neural network (ANN) that is widely used for classification tasks. It is a feedforward neural network model that consists of multiple layers of artificial neurons, including an input layer, one or more hidden layers, and an output layer.

The MLP classifier is trained using supervised learning, where labeled training data is used to adjust the weights and biases of the neurons. During the training process, the input data is passed through the network, and the activation values of the neurons in each layer are calculated using an activation function, such as the sigmoid function or the rectified linear unit (ReLU) function. The calculated activations are then propagated through the network until reaching the output layer, which produces the final classification predictions.

The number of neurons in the input layer is determined by the dimensionality of the input features, while the number of neurons in the output layer corresponds to the number of classes or categories in the classification problem. The hidden layers can vary in size and depth, depending on the complexity of the problem and the amount of training data available.

The training of the MLP classifier involves optimizing the network's weights and biases to minimize a loss function, typically using an algorithm called backpropagation. Backpropagation computes the gradients of the loss function with respect to the network's weights and biases, allowing for iterative adjustments that improve the model's performance.

Once trained, the MLP classifier can make predictions on new, unseen instances by forwarding the input through the network and obtaining the output from the output layer. The predicted class is typically determined by selecting the neuron with the highest activation value as the predicted class label.

The MLP classifier is a versatile model that can handle various types of data and has been successfully applied to a wide range of classification problems. However, it may require careful selection of hyperparameters, such as the number of hidden layers, the number of neurons in each layer, the learning rate, and the activation functions, to achieve optimal performance.

Chapter 4

Experiment

4.1 Dataset Description

In this section we discuss about the dataset. Dataset contains 37 class and 1745 text files. The dataset contains 47398 unique words. Dataset describes in the following below:

Category	No. of Doc	Category	No. of Doc
Agriculture	50	20. Football	50
Banking	50	21. Government_Operations	50
Business	50	22. Health	50
Caste	42	23. Labor_and_Employment	50
Cinema	50	24. Law	50
Computer	49	25. Miscellaneous	50
cricket	50	26. Music	50
Crime	50	27. Politics	50
Defence	50	28.Public_lands_and_water_management	24
.Economy	50	29. Religion	50
.Education	50	30. Science	50
.election	50	31. Space	46
.Electronics	50	32. Sports_other_than_football_and_cricket	50
.Energy	35	33. Technology	50
.Entertainment	51	34. Transportation	44
.Entertainment_other_than_cinema_and_music	3	35. travel	50
.Environment	50	36. Weather	50
.Family issues	50	37. 'World_and_International	50
.Finance	50		

4.2 Parameter tuning

4.2.1 Table for Binary Based Vector Representation:

Epoch=10

Batch Size	Accuracy
32	67
36	65
40	68
44	67
48	66
52	68
56	68
60	67
64	**69**
128	**69**

Epoch=20

Batch Size	Accuracy
32	**69**
36	67
40	67
44	68
48	67
52	**69**
56	**69**
60	67
64	66
128	**69**

Epoch=30

Batch Size	Accuracy
32	68
36	68
40	**69**
44	**69**
48	66
52	68
56	66
60	67
64	67

Epoch=40

Batch Size	Accuracy
32	66
36	65
40	65
44	68
48	67
52	68
56	68
60	67
64	68

4.2.2 Table for Count Based Vector Representation:

Epoch=10

Batch Size	Accuracy
32	68
36	66
40	69
44	68
48	67
52	68
56	69
60	68
64	**70**
128	**70**

Epoch=20

Batch Size	Accuracy
32	**70**
36	68
40	67
44	68
48	68
52	**70**
56	68
60	66
64	67
128	**70**

Epoch=30

Batch Size	Accuracy
32	69
36	68
40	**70**
44	**70**
48	67
52	69
56	67
60	68
64	68

Epoch=40

Batch Size	Accuracy
32	67
36	66
40	66
44	69
48	68
52	69
56	69
60	68
64	69

4.2.3 Table For tf-idf Based Vector Representation:

Epoch=10

Batch Size	Accuracy
32	70
36	67
40	69
44	67
48	68
52	68
56	70
60	69
64	69
128	70

Epoch=20

Batch Size	Accuracy
32	70
36	68
40	70
44	68
48	69
52	70
56	69
60	66
64	66
128	70

Epoch=30

Batch Size	Accuracy
32	70
36	68
40	66
44	70
48	69
52	69
56	70
60	68
64	**72**
128	**72**

Epoch=40

Batch Size	Accuracy
32	69
36	66
40	64
44	69
48	68
52	69
56	70
60	68
64	69
128	69

4.2.4 Table for Frequency Based Vector Representation:

Epoch=10

Batch Size	**Accuracy**
32	56
36	55
40	56
44	**57**
48	55
52	56
56	55
60	54
64	56
128	**57**

Epoch=20

Batch Size	**Accuracy**
32	53
36	**57**
40	52
44	56
48	**57**
52	55
56	**57**
60	56
64	**57**
128	**57**

Epoch=30

Batch Size	**Accuracy**
32	**57**
36	56
40	55
44	**57**
48	**57**
52	56
56	**57**
60	55
64	**57**

Epoch=40

Batch Size	**Accuracy**
32	53
36	56
40	**57**
44	55
48	**57**
52	56
56	**57**
60	55
64	56

Chapter 5

Evalution and Results:

5.1 Table 1:

The following table represents the result of the classification model using ANN based on binary vector representation.

Class	precision	recall	f1-score	support
0	0.54	1.00	0.70	7
1	0.62	0.80	0.70	10
2	0.42	0.62	0.50	8
3	0.83	0.83	0.83	6
4	0.64	1.00	0.78	7
5	0.91	0.83	0.87	12
6	1.00	0.92	0.96	13
7	0.58	0.50	0.54	14
8	0.77	1.00	0.87	10
9	0.41	0.70	0.52	10
10	0.92	0.85	0.88	13
11	0.90	0.90	0.90	10
12	0.71	0.91	0.80	11
13	0.89	0.67	0.76	12
14	1.00	0.86	0.92	7
16	0.78	0.88	0.82	16
17	0.83	1.00	0.91	10
18	0.62	0.31	0.42	16
19	0.86	0.86	0.86	7
20	1.00	0.09	0.17	11
21	0.50	0.86	0.63	7
22	0.14	0.11	0.12	9
23	0.27	1.00	0.43	3
24	1.00	1.00	1.00	10
25	1.00	0.88	0.93	8
26	0.60	0.60	0.60	10
27	0.60	0.43	0.50	7
28	0.60	0.43	0.50	14
29	0.78	0.64	0.70	11
30	0.00	0.00	0.00	3
31	0.75	1.00	0.86	3
32	0.89	1.00	0.94	8
33	0.29	0.33	0.31	6
34	0.67	0.80	0.73	10
35	0.50	0.60	0.55	5
36	1.00	0.70	0.82	10
37	0.62	0.29	0.40	17
accuracy : 0.69				351
macro avg	0.69	0.71	0.67	351
weighted avg	0.72	0.69	0.68	351

5.2 Table 2:

The following table represents the result of the classification model using ANN based on count vector representation.

class	**precision**	**recall**	**f1-score**	**support**
0	0.70	0.88	0.78	8
1	0.58	0.78	0.67	9
2	0.29	0.50	0.36	8
3	0.71	0.83	0.77	6
4	0.83	0.71	0.77	7
5	0.78	0.64	0.70	11
6	0.93	1.00	0.96	13
7	0.64	0.54	0.58	13
8	0.82	1.00	0.90	9
9	0.67	0.60	0.63	10
10	0.77	1.00	0.87	10
11	1.00	0.90	0.95	10
12	0.70	0.78	0.74	9
13	0.80	0.73	0.76	11
14	0.62	1.00	0.76	8
16	0.75	0.90	0.82	10
17	0.56	0.83	0.67	12
18	0.38	0.30	0.33	10
19	0.89	1.00	0.94	8
20	0.50	0.11	0.18	9
21	1.00	0.54	0.70	13
22	0.64	0.70	0.67	10
23	0.43	0.43	0.43	7
24	0.91	0.91	0.91	11
25	0.83	0.83	0.83	12
26	0.47	0.73	0.57	11
27	1.00	0.62	0.77	8
28	0.88	0.58	0.70	12
29	0.58	0.88	0.70	8
30	0.78	1.00	0.88	7
31	0.90	1.00	0.95	9
32	0.56	0.38	0.45	13
33	1.00	0.50	0.67	6
34	0.67	0.36	0.47	11
35	0.88	0.88	0.88	8
36	0.57	0.33	0.42	12
Accuracy: 0.70				349
macro avg	0.72	0.71	0.70	349
weighted avg	0.72	0.70	0.69	349

5.3 Table -3

The following table represents the result of the classification model using ANN based on tf-idf vector representation.

Class	precision	recall	f1-score	support
0	0.71	0.62	0.67	8
1	0.53	0.89	0.67	9
2	0.38	0.38	0.38	8
3	0.67	0.67	0.67	6
4	0.83	0.71	0.77	7
5	0.77	0.91	0.83	11
6	0.92	0.92	0.92	13
7	0.75	0.69	0.72	13
8	0.90	1.00	0.95	9
9	0.62	0.50	0.56	10
10	0.83	1.00	0.91	10
11	0.88	0.70	0.78	10
12	0.64	0.78	0.70	9
13	0.86	0.55	0.67	11
14	0.73	1.00	0.84	8
16	0.73	0.80	0.76	10
17	0.67	0.83	0.74	12
18	0.50	0.40	0.44	10
19	1.00	1.00	1.00	8
20	0.18	0.22	0.20	9
21	0.57	0.62	0.59	13
22	0.89	0.80	0.84	10
23	0.50	0.71	0.59	7
24	0.77	0.91	0.83	11
25	1.00	0.83	0.91	12
26	0.57	0.73	0.64	11
27	0.86	0.75	0.80	8
28	0.71	0.42	0.53	12
29	0.70	0.88	0.78	8
30	0.86	0.86	0.86	7
31	0.90	1.00	0.95	9
32	0.57	0.31	0.40	13
33	0.57	0.67	0.62	6
34	0.86	0.55	0.67	11
35	0.78	0.88	0.82	8
36	0.62	0.42	0.50	12
accuracy : 0.72				349
macro avg	0.72	0.72	0.71	349
weighted avg	0.72	0.71	0.70	349

5.4 Table -4

The following table represents the result of the classification model using ANN based on frequency vector representation.

class	precision	recall	f1-score	support
0	0.62	0.62	0.62	8
1	1.00	0.44	0.62	9
2	0.00	0.00	0.00	8
3	0.33	0.17	0.22	6
4	0.67	0.86	0.75	7
5	0.42	0.73	0.53	11
6	0.86	0.92	0.89	13
7	0.40	0.15	0.22	13
8	1.00	1.00	1.00	9
9	0.67	0.40	0.50	10
10	0.53	1.00	0.69	10
11	1.00	0.80	0.89	10
12	0.60	0.67	0.63	9
13	0.80	0.36	0.50	11
14	0.86	0.75	0.80	8
16	0.58	0.70	0.64	10
17	0.67	0.67	0.67	12
18	0.44	0.70	0.54	10
19	1.00	0.50	0.67	8
20	0.07	0.22	0.11	9
21	0.75	0.23	0.35	13
22	0.50	0.70	0.58	10
23	0.38	0.43	0.40	7
24	0.91	0.91	0.91	11
25	1.00	0.25	0.40	12
26	0.43	0.55	0.48	11
27	1.00	0.25	0.40	8
28	0.47	0.67	0.55	12
29	0.55	0.75	0.63	8
30	0.57	0.57	0.57	7
31	0.80	0.89	0.84	9
32	0.43	0.23	0.30	13
33	0.67	0.67	0.67	6
34	0.67	0.55	0.60	11
35	0.88	0.88	0.88	8
36	0.29	0.42	0.34	12
Accuracy : 0.57				349
macro ave	0.63	0.57	0.57	349
weighted avg	0.63	0.57	0.56	349

5.5 Result:

The following table represents the accuracy of this model based on 4 types of vector representation (binary,count,tf-idf,frequency).

Vector Representation	Accuracy
Binary Based	69%
Count Based	70%
Tf-Idf Based	72%
Frequency based	57%

Tf-idf based vector representation provides best accuracy i.e 72% and Frequency based **vector representation provides worse accuracy i.e 57%.**

Chapter 6

CONCLUSION AND FUTURE WORK:

In this text classification model, we represent 4 types of vector representation for training the dataset those are binary, count, tf-idf, frequency based vector representation. We find that count and tf-idf based vector representation produce better results (70% and 72% respectively) and frequency based vector representation produce worse result (57%).

Future work on Bengali text classification using Artificial Neural Networks (ANN) can focus on several areas to improve the performance and address specific challenges. Here are some potential avenues for future research:

1. Enhanced Models: Researchers can explore more advanced ANN architectures specifically tailored for Bengali text classification. This includes investigating the use of deep learning models such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), or transformer-based models like BERT (Bidirectional Encoder Representations from Transformers) that have achieved significant success in other languages**.**

2. Data Augmentation: Data scarcity is a common challenge in Bengali text classification. Future work can explore techniques to augment the existing labeled datasets, such as using data synthesis methods, back-translation, or unsupervised pre-training on large amounts of unlabeled Bengali text data.

3. Domain Adaptation: Developing techniques for domain adaptation is important for Bengali text classification, as different domains may exhibit distinct linguistic patterns and vocabulary.

Future work can explore methods to transfer knowledge from general domains to specific domains, improving the model's ability to classify text in domain-specific contexts.

4. Interpretability and Explainability: As ANN models lack interpretability, future research can focus on developing techniques to explain the decision-making process of these models in Bengali text classification. This includes methods like attention mechanisms, model visualization, and feature importance analysis to provide insights into the model's decision process.

REFERENCES

1. https://www.sciencedirect.com
2. https://keras.io/
3. https://scikit-learn.org/stable/
4. https://www.javatpoint.com/keras
5. https://github.com/keras-team/keras
6. https://machinelearningknowledge.ai/keras-tokenizer-tutorial-with-examples-for-fit_on_texts-texts_to_sequences-texts_to_matrix-sequences_to_matrix/
7. 1. Yoon, K. J., & Seo, J. (2018). Text classification algorithms: A survey. Information, 9(4), 90.
8. 2. Goldberg, Y. (2016). A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research, 57, 345-420.
9. 3. Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.
10. 4. Kim, Y. (2014). Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1746-1751).
11. 6. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). FastText.zip: Compressing text classification models. arXiv preprint arXiv:1612.03651.
12. 7. Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. Journal of Machine Learning Research, 3(Feb), 1137-1155.