# NEWS ARTICLE CATEGORIZATION USING NAIVE BAYES MODEL

A project report submitted

in partial fulfilment of requirement for the award of degree

### BACHELOR OF TECHNOLOGY

in

### COMPUTER SCIENCE & ENGINEERING

By

### KATURI ARAVIND (2203A51140)

**POTHANA NAGAVISHNU (2203A51189)**

### SOMARTHI NIMESH (2203A51446)

**PANCHAKAM DAVAN VIKAS (2203A51380)**

Under the guidance of

**Mr. Dr. N. Venkatesh**

Associate Professor, CS & AI.



# SR UNIVERSITY

Ananthasagar, Warangal.



**CERTIFICATE**

This is to certify that this project entitled **“NEWS ARTICLE CATEGORIZATION USING NAIVES BAYES MODEL**" is the bonafied work carried out by **KATURI ARAVIND, POTHANA NAGAVISHNU, SOMARTHI NIMESH, DAVAN VIKAS** as a Capstone Phase-II project for the partial fulfilment to award the degree **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING** during the academic year 2024-2025 under our guidance and Supervision.

### Dr. N. Venkatesh Dr. M. Sheshikala

Assoc. Prof. CS & AI Assoc. Prof. & HOD(CSE),

S R University, S R University,

Ananthasagar, Warangal. Ananthasagar, Warangal.

**External Examiner**

# ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our project guide **Mr. Dr. N. Venkatesh, Assoc. Prof. CS and AI** as well as Head of the CSE Department **Dr. M. Sheshikala, Associate Professor** for guiding us from the beginning through the end of the Capstone Phase-II project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction..

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

**Table of contents**

Title Page no

1. Introduction 2
2. Problem statement 3
3. Motivation and scope of work 3
4. Literative review 5
5. Dataset 7
6. Proposed Methodology 8-9
7. Flow chart 10
8. Compared Algorithms 11
9. Hardware and software tools 12
10. Results and figures 13-19
11. Conclusion 20
12. Future work 20

**List of Figure**

|  |  |  |
| --- | --- | --- |
| **s.no** | **Figure name** | **Page no** |
| 1. | Dataset of project | 7 |
| 2. | Flow chart of project | 10 |
| 3. | Finding categories in dataset | 13 |
| 4. | Checking Null Values | 14 |
| 5. | Finding common words | 15 |
| 6. | Training and evaluating Model | 16 |
| 7. | Intializing the model | 17 |
| 8. | Applying a graphical user interface | 18 |
| 9. | Output of the project | 19 |
| 9.1 | Output fig1 | 19 |
| 9.2 | Output fig2 | 19 |
| 9.3 | Output fig3 | 19 |

# ABSTRACT

The Naive Bayes classifier, a probabilistic machine learning technique, is useful for classification tasks. It is based on the Bayes theorem, which states that the likelihood of an event occurring given some observed evidence is equal to the prior probability of the event occurring. The Naive Bayes classifier can be trained on a dataset of labelled news articles, each of which is associated with a particular class or category, for the purpose of classifying news articles. The features of the articles, such as the words used and the length of the article, can then be used by the classifier to predict the class of an unseen article. The "naive" assumption, which is one of the key assumptions of the Naive Bayes classifier, is that the articles' features are independent of one another. The classifier is able to predict outcomes without taking into account how features interact with one another because of this assumption. The Naive Bayes classifier can still perform well on many classification tasks, including the classification of news articles, despite this assumption.

# KEYWORDS

# Natural language toolkit ,python ,machine learning algorith

# INTRODUCTION:

The process of classifying a news article according to its content is known as news article classification. This is a common issue in information retrieval and natural language processing, and it can be useful for organizing and searching through large collections of news articles.

One approach to categorising news stories is to use a machine learning algorithm like the Naive Bayes classifier. A probabilistic model known as the Naive Bayes classifier makes predictions based on the likelihood that particular occurrences will occur. The events are the classes or categories to which news articles can be classified, and the features are the words or other characteristics of the articles.

A dataset of labelled news articles, each of which is associated with a distinct class, is required to train a Naïve

Bayes classifier for news article classification. The classifier would then learn the probability distribution of the

characteristics for each class. which would then use this information to predict articles that had not been seen

before. The Naive Bayes classifier is able to efficiently simplify calculations and make predictions because it

assumes that the articles' features are independent of one another.

A lot of classification tasks benefit from the Naive Bayes classifier's relative simplicity and ease of use, which

is one of its advantages. It can also do well on a variety of classification problems, such as classifying news articles.

However, in order to ensure that the classifier is effective, it is essential to evaluate its performance on your

particular dataset and problem.

# PROBLEM STATEMENT

In the digital age, with an exponential increase in online content, efficient organization and categorization of news articles have become imperative. News websites and aggregators face the challenge of quickly and accurately categorizing articles into relevant topics to enhance user experience and facilitate targeted content delivery.

Effective categorization is not just a matter of convenience; it's integral to enhancing user experience and ensuring that readers can quickly access the content most relevant to their interests. Moreover, targeted content delivery has emerged as a key strategy for retaining users and driving engagement in a competitive online environment.

By developing a Naive Bayes classification model, we aim to harness the power of machine learning to automate the process of assigning articles to predefined topics or categories. Leveraging the textual content of news articles, this model will learn to identify patterns and associations that distinguish one topic from another.

# MOTIVATION AND SCOPE OF WORK

The motivation behind implementing a Naive Bayes classification model for news article categorization stems from the pressing need to streamline content management processes and improve content discoverability for users. By automating the categorization process, we aim to alleviate the burden on human moderators and enable news platforms to efficiently classify articles into relevant topics or categories. This not only enhances the overall browsing experience for users but also enables targeted content delivery, thereby maximizing user engagement and satisfaction.

The scope of this project encompasses the development and implementation of a Naive Bayes classification model for news article categorization, focusing on the following key aspects:

**Data Collection and Preparation:** Gather a diverse dataset of news articles spanning different categories such as politics, sports, entertainment, technology, etc. Preprocess the raw text data to remove noise, perform tokenization, and apply techniques like stemming or lemmatization to normalize the text.

**Feature Engineering:** Utilize techniques such as Bag-of-Words (BoW) or Term Frequency-Inverse Document Frequency (TF-IDF) to transform the preprocessed text data into numerical feature vectors. Additionally, explore advanced feature extraction methods to capture semantic information and improve model performance.

**Model Development:** Implement the Naive Bayes algorithm, a probabilistic classifier based on Bayes' theorem, to build a classification model. Experiment with different variants of Naive Bayes, such as Multinomial Naive Bayes or Bernoulli Naive Bayes, to identify the most suitable approach for the task at hand.

**Model Evaluation and Optimization:** Assess the performance of the Naive Bayes model using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score. Employ techniques like cross-validation and hyperparameter tuning to optimize the model's performance and ensure robustness.

**Integration and Deployment:** Once the model achieves satisfactory performance, deploy it into a production environment where it can automatically categorize incoming news articles in real-time. Integrate the model with existing news platforms or APIs to facilitate seamless classification and content delivery.

By delineating the motivation and scope of the work, we aim to establish a clear roadmap for developing and implementing a Naive Bayes classification model for news article categorization, with the overarching goal of enhancing user experience and engagement in the digital news ecosystem.

# LITERATIVE REVIEW

## 4.1 Related Work

R. Siva Subhramanian and D. Prabha [22] contributed their paper in In February 2020 on research of This research seeks to identify potential customers.

They used the SBC method to modify the NB model with the goal of enhancing prediction by removing unnecessary dataset features.

According to the experimental findings, the WSNB running time is 0.03 seconds for WSNB at depth 1, 0.06 seconds for WSNB at depth 2, and 0.15 seconds for WSNB at depth 3. Running time for Standard Naive Bayes was 0.16 seconds. Which was unmistakably demonstrating that WSNB shortens the model's running time as compared to traditional Naive Bayes.

Faculty of Agriculture, University of Novi Sad [23] published their article in 2022. The effectiveness of the Naive Bayes approach for predicting water quality was studied by the author. Nine water quality factors were examined, including temperature, oxygen saturation values, and others. Five locations and 68 samples of data were used to assess the water quality using the Naive Bayes model. The testing report ranked each parameter as very good, excellent, good, or bad; after analysing the report and using the method, the author came to the conclusion that the model correctly identified water class in 64 out of 68 instances.

Disha Sharma and Sumit Chaudhary [24] They studied various sources of stress which includes 1) The surrounding Environment 2) Social Stress 3) Physiological 4) Thoughts

Authors applied four machine learning technics that are logistic Regression, Naïve Bayes, Multilayer perceptron ,Bayer’s Net.

Parameters like False Positive rate, True Positive Rate , precision, Recall considered for the performance. After comparing all the results of four methods they concluded that Baye’s Net classifiers gives longest accuracy of 88 percentage and Naive Bayes gives accuracy of 86 percentage.

Mamata Thakur and team [25] by concerning the problem of huge growth of internet and difficulty in getting relevant topic according to search. Authors chose some news websites after that the important attributes from these

The Nave Bayes algorithm was used by the authors to classify data from 10 different websites, and the results of comparative studies with other current algorithms on the same dataset demonstrate that Nave Bayes outperforms them.

Yi Ying [26] The author of this study employed a variety of news stories to research and used news categories including sports, politics, business, etc. The Confusion Matrix results show that the Sarcasm model developed using the Naive Bayes approach above achieved an accuracy level of 66%, 70% withdrawal, and 68% precision.

By summarising the literature review we can understand sometimes Navies Bayes gives good results but not able to give 100% correct results and some of other machine learning algorithms are more effective than NB, so more researches can be done increase efficiency of NB

## DATASET

Dataset consists of Article Id, Article Description and Article category which it belongs to

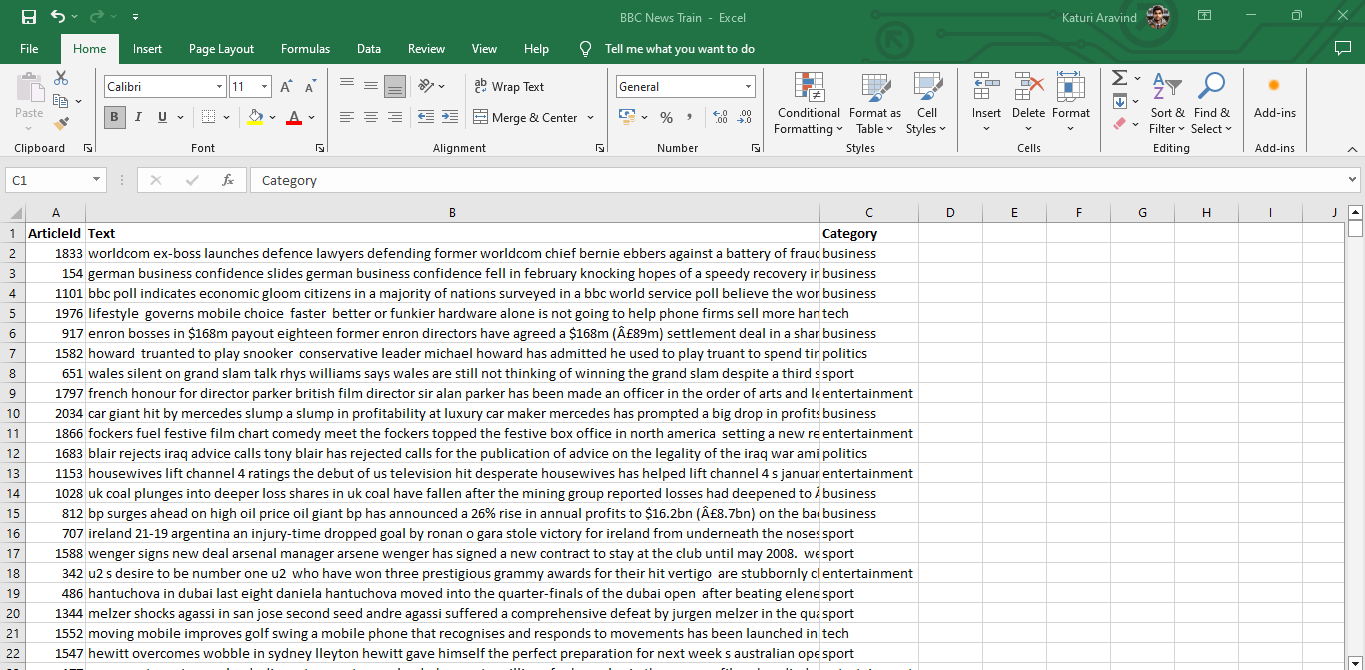


Fig 1. Dataset

# PROPOSED METHODOLOGY

# Our proposed methodology for news article categorization using a Naive Bayes model encompasses several key steps to ensure the development of an accurate and robust classification system. The first phase involves data collection and preprocessing. In this step, we gather a diverse dataset of news articles spanning different categories such as politics, sports, entertainment, technology, and more. We then preprocess the text data by removing noise, including HTML tags, punctuation, and stop words, and perform tasks like tokenization and stemming to normalize the text. Additionally, we may employ techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) or Bag-of-Words to convert the preprocessed text data into numerical feature vectors, which will serve as input to the Naive Bayes classifier.

# Following data preprocessing, the next phase entails model development and training. Here, we implement the Naive Bayes algorithm, which assumes independence between features, to build the classification model. We split the dataset into training and testing sets to train the model on labeled data and evaluate its performance. During training, the model learns the statistical relationships between features and categories, enabling it to make accurate predictions. We fine-tune model parameters and optimize performance using techniques like cross-validation. Once the model achieves satisfactory performance metrics, we proceed to deploy it into a production environment, where it can automatically categorize incoming news articles in real-time. Continuous monitoring and evaluation ensure the model's effectiveness and adaptability to evolving news trends and topics. Through this methodology, we aim to deliver a robust and efficient news article categorization system that enhances user experience and facilitates targeted content delivery.

Here is a general approach to utilising the provided code to categorise news articles:

1. Assemble and classify a dataset of news stories, each with a category tagged (e.g., sports, tech, business, entertainment). The classifier will be trained and tested using this dataset.

2. Remove all stop words from the data and lowercase each word in each article as part of the pre-processing.

3. To turn the text input into numerical feature vectors, create a TfidfVectorizer.

4. The training and test data should be converted into feature vectors using the TfidfVectorizer.

5. Making use of the training data, create a Multinomial Naive Bayes classifier.

6. Calculate the classifier's accuracy by evaluating it against the test data.

7. Make predictions for fresh, unlabelled news articles using the classifier by converting them into feature vectors and passing them into the classifiers predict method

# Flow Chart

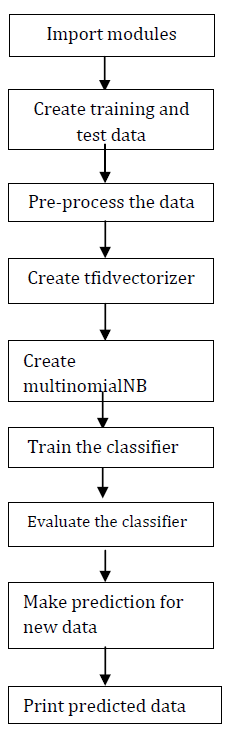


Fig .2 Flow chart

## COMPARED ALGORITHMS

### Xg boosting

XGBoost is a widespread implementation of gradient boosting. Let’s discuss some features of XGBoost that make it so attractive.

* XGBoost offers regularization, which allows you to control overfitting by introducing L1/L2 penalties on the weights and biases of each tree. This feature is not available in many other implementations of gradient boosting.
* Another feature of XGBoost is its ability to handle sparse data sets using the weighted quantile sketch algorithm. This algorithm allows us to deal with non-zero entries in the feature matrix while retaining the same computational complexity as other algorithms like stochastic gradient descent.

### SVM

Sentiment Analysis is an NLP method that works in a text to determine the author's intentions for a particular topic, product, etc. positive, negative, or neutral. The supporting vector machine analyzes the data, defines the decision parameters and uses the calculations to calculate what is done in the input field data entry for two sets of vectors of each size. Then all the data represented as vector is categorized. Next, we find the margin between the two sections away from any document. The distance defines the divider margin, increasing the limit reduces the final decisions. SVM also supports the subdivision and decline in practical mathematical learning theory and also helps to identify specific factors, which need to be considered, in order to understand them effectively.

### DECISION TREE

Decision trees are the most common way to say something. They are strong on sound data and learn divisive sayings. Decision tree is a k-array tree where each internal node displays an experiment in a few elements from a set of input element that communicates with the data. Every branch from a node is related to the unimaginable feature values determined for that node. Also, all test results in branches, refer to changed test results. The basic algorithm for decision tree imports algorithm is the decision-making tree algorithm in the form of repeated downward divisions and conquests.

**Accuracy Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **SVM** | **Decision tree** | **Xg boost** | **Multinomial Naïve bayes** |
| Business | Precision: 0.97  Recall: 0.97  F1-score: 0.97  Accuracy: 0.98 | Precision:0.82  Recall:0.82  F1-score:0.82  Accuracy:0.78 | Precision:0.95  Recall:0.97  F1-score:0.96  Accuracy:0.96 | Precision:0.96  Recall:0.98  F1-score:0.97  Accuracy:0.97 |
| Entertainment | Precision:0.98  Recall:0.98  F1-score:0.98  Accuracy:0.98 | Precision:0.68  Recall:0.75  F1-score:0.71  Accuracy:0.78 | Precision:0.98  Recall:0.98  F1-score:0.98  Accuracy:0.96 | Precision:0.98  Recall:0.92  F1-score:0.95  Accuracy:0.97 |
| Politics | Precision:0.98  Recall:0.98  F1-score:0.98  Accuracy:0.98 | Precision:0.72  Recall:0.76  F1-score:0.74  Accuracy:0.78 | Precision:0.98  Recall:0.92  F1-score:0.95  Accuracy:0.96 | Precision:0.98  Recall:0.98  F1-score:0.98  Accuracy:0.97 |
| Sports | Precision:0.97  Recall:1.00  F1-score:0.99  Accuracy:0.98 | Precision:0.82  Recall:0.80  F1-score:0.81  Accuracy:0.78 | Precision:0.97  Recall:1.00  F1-score:0.99  Accuracy:0.96 | Precision:0.96  Recall:1.00  F1-score:0.98  Accuracy:0.97 |
| Tech | Precision:0.98  Recall:0.94  F1-score:0.96  Accuracy:0.98 | Precision:  Recall:  F1-score:  Accuracy: | Precision:0.92  Recall:0.92  F1-score:0.92  Accuracy:0.96 | Precision:0.96  Recall:0.94  F1-score:0.95  Accuracy:0.97 |

## HARDWARE AND SOFTWARE TOOLS

### HARDWARE TOOLS

* System
* Hard Disk
* Ram-8 GB
* Processor

### SOFTWARE TOOLS

* Operating System-Windows 11
* Google Colab Notebook
* Python IDLE
* Pandas
* NumPy
* TensorFlow
* TPU
* NLTK

# 10.RESULTS & DISCUSSION

## Code:

# Import libraries

import pandas as pd

import re

import nltk

from nltk.tokenize import RegexpTokenizer

from collections import Counter

from nltk.tokenize import word\_tokenize

from nltk.stem import WordNetLemmatizer

from nltk.corpus import stopwords

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

from sklearn.feature\_extraction.text import TfidfVectorizer, TfidfTransformer

from sklearn.model\_selection import train\_test\_split

from sklearn import svm

from sklearn.naive\_bayes import MultinomialNB

from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier

from sklearn.tree import DecisionTreeClassifier

import xgboost

from sklearn.metrics  import classification\_report

from sklearn import metrics

import time

## Finding Categories in dataset

df1['Category'].value\_counts().plot(kind='barh')

plt.show()

## 

## Checking Null values

# Check null values

df1.isnull().sum()

ArticleId 0

Text 0

Category 0

dtype: int64

**Text Pre Processing**

# Text preprocessing

def preprocess(text):

    """

    Function: split text into words and return the root form of the words

    Args:

      text(str): the article

    Return:

      lem(list of str): a list of the root form of the article words

    """

    # Normalize text

    text = re.sub(r"[^a-zA-Z]", " ", str(text).lower())

    # Tokenize text

    token = word\_tokenize(text)

    # Remove stop words

    stop = stopwords.words("english")

    words = [t for t in token if t not in stop]

    # Lemmatization

    lem = [WordNetLemmatizer().lemmatize(w) for w in words]

    return lem

**Output:**



**Finding Common Words**

# Find the common words in each category

def find\_common\_words(df, category):

    """

    Function: find the most frequent words in the category and return the them

    Args:

      df(dataframe): the dataframe of articles

      category(str): the category name

    Return:

      the most frequant words in the category

    """

    # Create dataframes for the category

    cat\_df = df[df["Category"]==category]

    # Initialize words list for the category

    words = [word for tokens in cat\_df["Preprocessed\_Text"] for word in tokens]

    # Count words frequency

    words\_counter = Counter(words)

    return words\_counter.most\_common(10)



**Training And Evaluating Model**

# Train and evaluate model

def fit\_eval\_model(model, train\_features, y\_train, test\_features, y\_test):

    """

    Function: train and evaluate a machine learning classifier.

    Args:

      model: machine learning classifier

      train\_features: train data extracted features

      y\_train: train data lables

      test\_features: train data extracted features

      y\_test: train data lables

    Return:

      results(dictionary): a dictionary of the model training time and classification report

    """

    results ={}

    # Start time

    start = time.time()

    # Train the model

    model.fit(train\_features, y\_train)

    # End time

    end = time.time()

    # Calculate the training time

    results['train\_time'] = end - start

    # Test the model

    train\_predicted = model.predict(train\_features)

    test\_predicted = model.predict(test\_features)

     # Classification report

    results['classification\_report'] = classification\_report(y\_test, test\_predicted)

    return results

**Initializing the model**

# Initialize the models

nb = MultinomialNB()

# Fit and evaluate models

results = {}

for cls in [nb]:

    cls\_name = cls.\_\_class\_\_.\_\_name\_\_

    results[cls\_name] = {}

    results[cls\_name] = fit\_eval\_model(cls, train\_features, y\_train, test\_features, y\_test)

# Print classifiers results

for res in results:

    print (res)

    print()

    for i in results[res]:

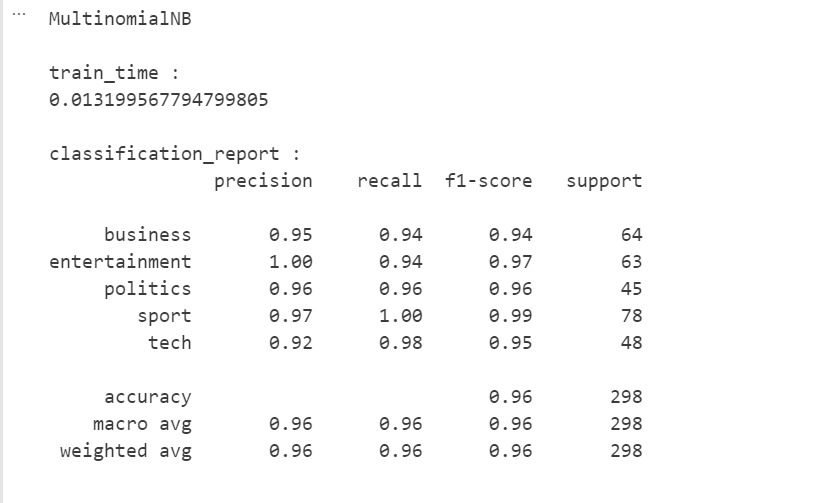
        print (i, ':')

        print(results[res][i])

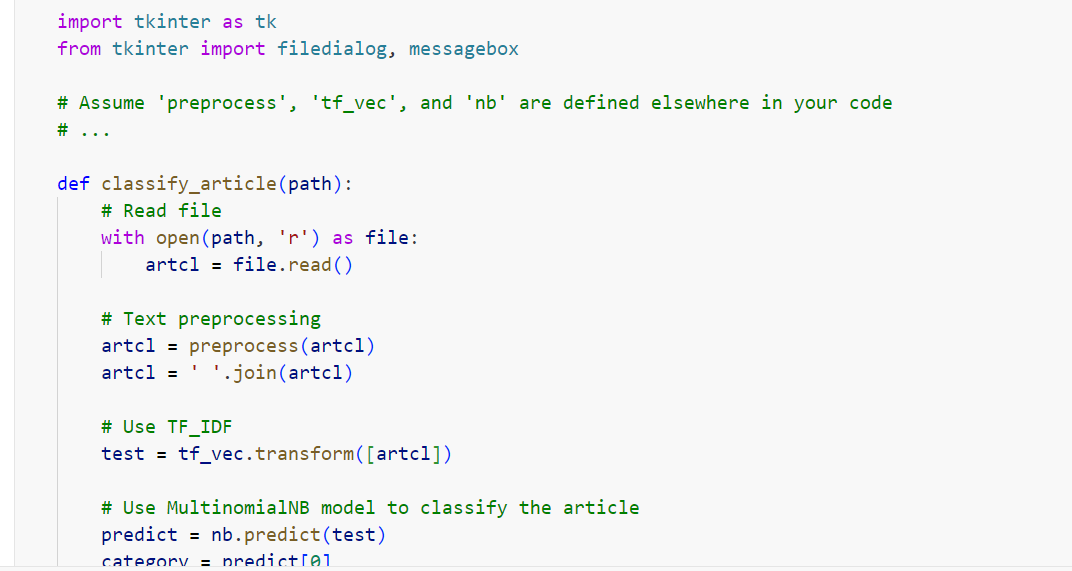
        print()

    print ('-----')

    print()

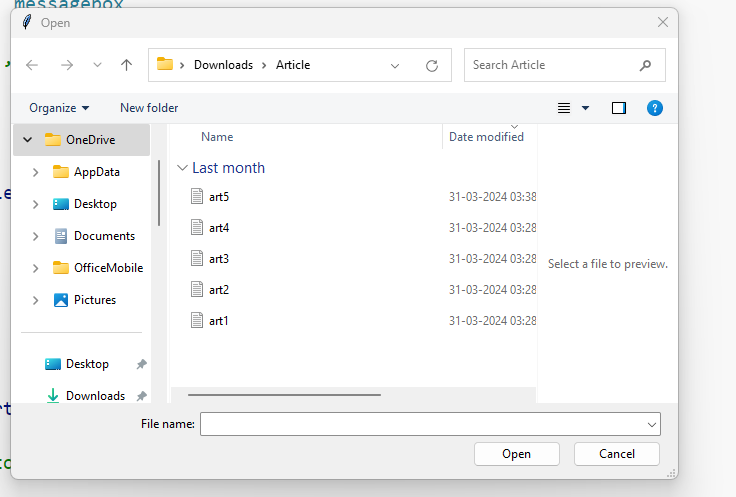


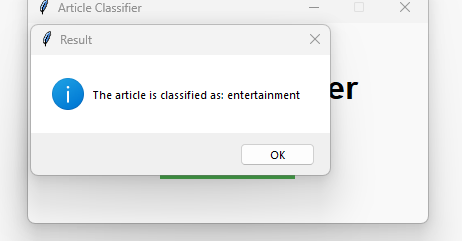
**Applying a Graphical User Interface (GUI)**



**Output of the Project**

****

****

****

# CONCLUSION:

# In conclusion, the versatility and simplicity of Naive Bayes models make them a compelling choice for news article categorization tasks in today's digital landscape. By leveraging probabilistic classification, Naive Bayes models can efficiently handle large volumes of textual data, providing a practical solution for organizing and categorizing news content across diverse topics. Their effectiveness lies in their ability to capture essential features from text data while making the assumption of feature independence, which often proves to be reasonable in practice.

# Furthermore, as the digital landscape continues to evolve, Naive Bayes models can benefit from integration with complementary techniques, such as deep learning approaches, to capture more nuanced semantic relationships within news articles. By embracing interdisciplinary approaches and collaborating across fields such as natural language processing, machine learning, and information retrieval, researchers can unlock new opportunities to enhance the effectiveness of Naive Bayes models in facilitating efficient organization and categorization of news content in the digital age.

# The Naive Bayes classifier is a popular machine learning method that can be used to categorise news stories. It has a lot going for it, like being easy to use, working well, and doing well on a lot of classification tasks. However, its performance on a particular dataset must be evaluated. The classifier's performance can be improved, more complex problems can be handled, and the classifier can be applied to new domains can all be developed further in this area. Natural language processing and information retrieval could benefit greatly from using the Naive Bayes classifier.

# FUTURE WORK:

# In the field of news article classification using Naive Bayes classifiers, there are numerous potential future directions for research and development. These are some: expanding the application of the classifier to new domains and languages, enhancing the classifier's performance, and incorporating it into news analysis systems. The Naive Bayes classifier offers a lot of potential for solving a variety of real-world issues, and more research may be done on its capabilities and restrictions when it comes to categorising news items.

# REFERENCES

[1] Kuldeep Vayadande, Aditya Bodhankar, Ajinkya Mahajan, Diksha Prasad, Shivani Mahajan, Aishwarya Pujari and Riya Dhakalkar, “Classification of Depression on social media using Distant Supervision”, ITM Web Conf. Volume 50, 2022.

[2] Kuldeep Vayadande, Rahebar Shaikh, Suraj Rothe, Sangam Patil, Tanuj Baware and Sameer Naik,” Blockchain-Based Land Record System”, ITM Web Conf. Volume 50, 2022.

[3] Kuldeep Vayadande, Kirti Agarwal, Aadesh Kabra, Ketan Gangwal and Atharv Kinage,” Cryptography using Automata Theory”, ITM Web Conf. Volume 50, 2022

[4] Samruddhi Mumbare, Kunal Shivam, Priyanka Lokhande, Samruddhi Zaware, Varad Deshpande and Kuldeep Vayadande,”Software.

[5] Preetham, H. D., and Kuldeep Baban Vayadande. "Online Crime Reporting System Using Python Django."

[6] Vayadande, Kuldeep B., et al. "Simulation and Testing of Deterministic Finite Automata Machine." International

Journal of Computer Sciences and Engineering 10.1 (2022): 13-17.

[7] Vayadande, Kuldeep, et al. "Modulo Calculator Using Tkinter Library." EasyChair Preprint 7578 (2022).

[8] VAYADANDE, KULDEEP. "Simulating Derivations of Context-Free Grammar." (2022).

[9] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.