A Dissertation on

# News Articles Sorting

Submitted in partial fulfilment of the requirement for the award of the degree

of

**MASTERS OF SCIENCE**

in

**Computer Science (BIG DATA ANALYTICS)**

Submitted by

**Raushan Kumar**

2021MSBDA033

Under the Guidance of

**Ms. Sneha Patil**

**Data Analyst (PHN Technology Pvt Ltd.)**

**&**

Under the Guidance of

**Dr. Pritpal Singh**

**Assistant Professor**

**Department of Data Science & Analytics**

Department of Data Science and Analytics

School of Mathematics, Statistics and Computational Science

CENTRAL UNIVERSITY OF RAJASTHAN

August 2023

## DECLARATION

I certify that

a. The work contained in the dissertation has been done by myself under the supervision of my supervisor.

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

d. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the dissertation and giving their details in the references.

e. Whenever I have quoted written materials from other sources and due credit is given to the sources by citing them.

Date: August 16 2023 Raushan Kumar

Place: Bandarsindri , Ajmer 2021MSBDA033

**DEPARTMENT OF DATA SCIENCE AND ANLYTICS**
CENTRAL UNIVERSITY OF RAJASTHAN, INDIA

Date: August 16 2023

# CERTIFICATE

This is certify that the project titled **News Articles Sorting** is a record of the bonafide work done by **Raushan Kumar** (2021MSBDA033) submitted in partial fulfillment of the requirements for the award of the Degree of Master of Science (M.Sc.) in Data Science and Analytics of Central University of Rajasthan, during the academic year 2022-23.

**Dr. Pritpal Singh**

Supervisor,

DEPARTMENT OF DATA SCIENCE AND ANALYTICS

Central University of Rajasthan

**Dr. Vidyottama Jain**

HOD,

DEPARTMENT OF DATA SCIENCE AND ANALYTICS

Central University of Rajasthan

## Acknowledgements

I would like to take this opportunity to extend my deepest gratitude to the Central University of Rajasthan for the assistance and direction they provided throughout the course of the completion of this project. Because of his foresight and leadership, I am especially grateful to the Honorable Vice-Chancellor, Professor Anand Bhalerao.

I would also like to express my gratitude to Professor D. C. Sharma, the Dean of the School of Mathematics, Statistics, and Computational Sciences, and Associate Professor Vidyottama Jain, the Head of the Department of Data Science& Analytics, for their encouragement and support in this endeavor.

I would like to express my sincere gratitude to my internal guide, Assistant Professor Pritpal Singh, and to my external guide, Sneha Patil, for their insightful comments, helpful recommendations, and invaluable assistance.Their knowledge and insights were a significant factor in the accomplishment of bringing this project to a successful conclusion.

Throughout the entirety of this project, I would like to express my appreciation to the members of the Department of Data Science & Analytics's faculty, specifically Dr. Bhavna Saini, Dr. Nishtha kesswani, and Dr. Vidyottama Jain, for providing me with invaluable guidance, support, and encouragement. I would like to express my gratitude to Bhagirath Ashiya, who oversees the laboratory that is responsible for the Department of Data Science & Analytics, for his invaluable assistance, support, and encouragement throughout the entirety of this project.

In addition, I would like to express my gratitude to the other students in my class as well as my friends for the camaraderie, motivation, and intellectual conversations that they have contributed to, all of which have contributed to the enhancement of my educational experience and the creation of a supportive environment for the advancement of my academic career.

Lastly, but certainly not least, I would like to convey to my family the depth of my appreciation for the unwavering love, understanding, and encouragement they have shown me throughout the entirety of my educational journey. Their unwavering support has been the engine that has powered my accomplishments to this point. Place: Bandarsindri, Ajmer Pravat Patra Date: August 12, 2023 MSc IV Semester Central University of Rajasthan Bandarsindri, Ajmer

Place: Bandarsindri , Ajmer

Date: August 16 2023

Raushan Kumar

M.Sc IV Semester

Central University of Rajasthan

# Abstract

Over the past few years, text mining has grown in importance significantly. Users can now access data from a variety of sources, including electronic and digital media. There are several ways to convert this data to structured form even though it is typically only available in the least structured form. It is highly beneficial to categorise the material in a suitable set of categories in many real-life situations. One of the most crucial elements that affects many parts is the news content. The issue of categorising news stories has been taken into consideration in this work. This essay analyses the drawbacks of several algorithmic approaches and proposes algorithms for categorising news.

In comparison to short length texts, the focus on complete text classification—including entire news, large documents, long texts, etc.—is more significant. The text classification process, classifiers, and various feature extraction approaches have all been covered in this study, but only in the context of short texts, such as the classification of news articles based on their headlines. Effective comparisons are made between existing classifiers and their operating procedures.We tested an automatic method of categorising the news in our proposed Categorizor system using the Logistic Regression and Random Forest Classifier.When given a large amount of training data, model has been proved to produce accurate classification results. In our study, we used Random Forest Classifier and Logistic Regression to categorise news in a personalised way.

Our research delved into news classification, examining the intricacies of pre-processing, document indexing, feature selection, and headline classification. A frequency-based approach was particularly highlighted for filtering out stop words. In our modeling endeavors, both the Logistic Regression and Random Forest classifiers showcased exemplary performance, achieving accuracy rates of approximately 96.91% and 97.56% respectively. While both models proved potent, the Random Forest classifier slightly outperformed, suggesting its capability to deeply understand article features. Notably, the 'tech' category identified areas for potential enhancement. The culmination of our research underscores the potential of advanced machine learning techniques in the realm of news categorization, offering promising avenues for future exploration.

Central to our computational endeavors was the Python programming language, renowned for its versatility and extensive library support tailored for data science. Our modeling approach predominantly harnessed the capabilities of Scikit-learn, an industry-standard library known for its comprehensive machine learning algorithms and utilities. To manage and preprocess our datasets, the Pandas library provided robust functionalities, while natural language processing tasks were adeptly handled by either NLTK or SpaCy, both of which stand as pillars in the text analysis domain. Visual interpretations of our results were rendered using the combined graphical strengths of Matplotlib and Seaborn. Furthermore, our entire analytical workflow was streamlined within the Jupyter Notebook environment, facilitating interactive computing and an integrative presentation of code, visualizations, and annotations. Optionally, for more nuanced semantic interpretations of text, advanced word embeddings such as Word2Vec based representations might have been explored.

# Contents

# List of Figures

# CHAPTER 1
# INTRODUCTION

## 1.1 Introduction to work done

In the age of information, news serves as a pivotal mechanism to inform, enlighten, and shape the understanding of global citizens. Every day, millions of articles are published across a myriad of platforms, feeding an insatiable demand for timely and relevant information. However, as this vast sea of information continues to expand, the challenge of effectively navigating it grows proportionally. The sheer volume of news content generated daily makes it impractical, if not impossible, for an individual to manually sift through and categorize each piece. This creates a pressing need for an efficient and accurate system to classify news articles, ensuring that readers can effortlessly access the information most pertinent to their interests.This diversity manifests itself in categories such as technology, politics, entertainment, sports, and many others.

At its core, news article sorting is the systemized categorization or arrangement of news content into coherent groups or sequences, optimizing accessibility and relevance for readers. It isn't just about filing articles into basic categories like 'Sports', 'Technology', or 'Politics'. It's about ensuring that when a reader seeks information, they are presented with a logically-structured, easily navigable, and contextually relevant array of content.

News article classification is no longer just a tool for enhancing user experience; it has become a cornerstone for news organizations aiming to maintain their relevance in a fiercely competitive market. By delivering targeted and categorized content, these entities can foster deeper engagement, cultivate loyal readerships, and, crucially, ensure that essential information reaches its intended audience.

When you visit a news website, the neatly segregated sections, be it 'Tech', 'Sports', or 'Politics', are not just there for aesthetic appeal. They serve the vital function of ensuring a streamlined user experience. An individual, like myself, who gravitates towards technological updates, knows precisely where to navigate upon entering a news portal. Conversely, someone with a penchant for politics or sports would have their distinct section to dive into. These divisions, seemingly simple, are the backbone of a user-friendly and efficient news dissemination system.

Historically, the task of sorting was a manual endeavour, with editors and journalists determining where a piece should be placed within a newspaper or magazine layout. With the digital revolution and the advent of online news platforms, this process evolved. Instead of fitting news into physical pages, articles now had to be sorted into digital categories, tags, and feeds. This shift, while opening doors for more dynamic content presentation, also introduced challenges. The volume of online news is exponentially greater than print, and the speed of publication is almost instantaneous.

Machine learning and data-driven methodologies have recently begun to play a pivotal role in news article sorting. Automated systems, equipped with algorithms trained on vast datasets, can now categorize articles in real-time based on nuanced parameters ranging from content and tone to reader behaviour and trending topics. These advanced sorting techniques allow for a more personalized and responsive news consumption experience, where content is tailored to individual preferences and real-time global developments.

The importance of effective news article sorting cannot be overstated. It is the bridge between information producers and consumers, ensuring that in a world overflowing with data, readers can still find the stories that resonate with them, shaping their understanding and perspective of the world. This exploration provides an overview of the multifaceted world of news article sorting, its evolution, methodologies, and its undeniable significance in the modern information age.

This thesis delves into the realm of automated news article classification, exploring its importance, methodologies, and implications. By harnessing cutting-edge machine learning algorithms, we aim to design a system capable of effectively categorizing news content, serving both news organizations and their diverse readership. Through this exploration, we aspire to bridge the gap between vast information repositories and the discerning reader, fostering a more informed and connected global community.

### 1.1.1 Background and Motivation

In today's digital age, vast amounts of data are stored electronically, necessitating tools that can interpret, analyse, and extract valuable insights for decision-making. Data mining is one such tool that can uncover hidden details from large databases. Before the last decade, timely news access was limited, but now it's readily available through online services. The abundance of textual information across various fields offers opportunities for analysis that can be beneficial in many contexts.

Text mining, especially the classification of text, is a complex field due to the need to transform unstructured data into structured information. With the influx of news articles, it has become challenging for users to find news tailored to their interests. As a result, there's a pressing need to categorize news, enhancing user navigation and accessibility. Categorizing online news helps users find their desired content quickly and efficiently. However, classifying news is challenging, especially as new stories may emerge that don't fit established categories or could even define a new one.

This article provides an overview of news classification methods focusing on content and headlines. Past methods, their effectiveness, and associated shortcomings are also explored.

The classification of news articles into distinct categories based on their content is a critical function in today's digital news landscape. With the proliferation of online news, driven by both established media houses and independent content creators, there's a vast array of information available on a multitude of topics, from politics and sports to technology and entertainment. This expansive and diverse array of digital content has necessitated systematic categorization, mirroring the compartmentalized approach of traditional print media where news was physically separated into distinct sections. The motivations behind this classification are multifaceted. Efficient categorization enables rapid and precise information retrieval, ensuring that readers can quickly find and engage with topics that resonate with their interests. Beyond enhancing user experience, classification plays a pivotal role in facilitating personalized content delivery.

By understanding users' engagement with specific categories, digital platforms can curate tailored news feeds, optimizing reader engagement. This, in turn, improves search functionality, with algorithms prioritizing articles from categories that align more closely with a user's search intent. There's also a commercial dimension to classification; for platforms driven by ad revenues, categorizing articles ensures that advertisements are strategically placed alongside relevant content, thereby attracting advertiser trust and investment.

Furthermore, a systematic categorization provides invaluable insights for trend analysis, allowing platforms and researchers to identify shifts in news coverage and gauge public interest in specific topics. Properly classified content also acts as a bulwark against the spread of misinformation by clearly differentiating opinion pieces from factual reports.The task of news article classification, bolstered by a blend of manual efforts and advanced machine learning techniques, is indispensable in our current digital era. It serves the dual purpose of equipping readers with a focused and personalized news experience while providing platforms with tools for optimized content delivery and analysis.

Therefore, modern news platforms, equipped with advanced algorithms and technologies like natural language processing and machine learning, are working diligently to seamlessly blend personalization, relevance, and diversity, ensuring an optimal news consumption experience.

### 1.1.2 Applications

News article classification finds applications in several areas within journalism, media, information technology, and beyond. Here are detailed applications of news article classification:

- Platforms like Google News or Feedly pull articles from various sources. Classification allows these platforms to present organized sections (e.g., Technology, Health, Politics) so users can choose which categories they're most interested in.
- Based on a user’s reading habits or explicitly stated interests, platforms can offer a tailored news feed. Classification ensures that users receive articles from their

preferred categories.When users search for news on a specific topic, search engines use classification to refine and present the most relevant articles.

- Advertisers want their ads to appear next to relevant content. For example, a company advertising running shoes might want their ads to appear in or alongside sports articles.
- Platforms like news websites or apps may suggest "related articles" or "you might also be interested in" sections. Classification helps in automating this process, ensuring the relevance of suggested articles.
- Scholars and market researchers might be interested in studying articles from specific categories over time to analyze trends, sentiment, or other patterns. Classification facilitates such focused research.
- Companies and governments might use news monitoring tools to stay informed about specific topics, like mentions of their organization or topics of interest. Classification ensures they receive timely and relevant updates.
- Large media organizations with vast databases of past articles can use classification to organize their archives, making it easier to retrieve old articles when needed.
- Editors or curators, especially in platforms that curate content from various sources, can utilize classification to efficiently select and present articles fitting a particular theme or topic for their audience.
- Platforms like Facebook or Twitter, where users share news articles, can use classification to categorize shared content and improve their content recommendation algorithms.
- News platforms with multimedia content (like videos or podcasts) can suggest content based on the classification of articles a user reads. For example, after reading a science article, a user might be presented with a related science video.
- Classified news articles can be used as training datasets for various machine learning tasks, helping to develop models for tasks like sentiment analysis, topic modeling, etc.
- News platforms that offer content in multiple languages can use classification to identify which articles might be relevant to translate and share across different language sections of their platform.
- Chatbots or virtual assistants can use classification to provide users with news updates in specific categories upon request.

### 1.1.3 Advantages

- News article classification, also known as news categorization or taxonomy, involves assigning predefined categories or labels to news articles based on their content. This practice has a wide range of applications in modern journalism, media, and information retrieval. Here are some advantages of news article classification in detail:

- Classification allows for better organization of news content. By grouping articles into specific categories, publishers can streamline their websites or applications and make it easier for users to find articles of interest.
- Readers can easily navigate to their preferred news topics or sections, such as sports, politics, entertainment, etc., based on classified labels.
- Personalized news feed recommendations can be made based on users' reading habits or preferences.
- Search engines, research tools, and news aggregation platforms can utilize classification to provide more relevant results to users.It reduces information overload, presenting only the most pertinent articles to the readers.
- With classified articles, news platforms can build sophisticated recommendation engines to suggest similar articles or topics that the reader might be interested in.
- Advertisers can target specific categories or sections where they believe their ads will be most relevant. For example, sports equipment brands can target the sports section of a news website.
- Automated classification can help editorial teams in quickly sorting and prioritizing news content for publication.Journalists can identify gaps in coverage by analyzing which categories might be underrepresented during a certain period.
- By analyzing the frequency and trends of certain categories, media organizations can gain insights into readers' preferences, current events' significance, and emerging patterns in news consumption.
- News platforms that cater to a global audience in different languages can use classification to ensure that articles in all languages are correctly categorized and made accessible to readers.
- By comparing the classification of an article with its content, potential discrepancies can be flagged for further investigation. For instance, if an article is labeled as "science" but contains mostly political rhetoric, it might be subjected to additional scrutiny.
- Scholars, researchers, and students can use classified news data for various research purposes, such as studying media trends, sentiment analysis, and understanding the media's portrayal of specific events or topics.

## 1.2 Project statement

In the modern era, data holds immense value as it empowers organizations with insights that drive progress. News companies possess massive data repositories, but deriving meaningful insights from this data is a challenge. Among the many applications of data analytics, one particularly impactful use case is news article classification.

In today's digital landscape, a multitude of sources flood the internet with an overwhelming volume of daily news articles. Concurrently, users' thirst for information continues to grow unabated. To meet this demand effectively, the classification of news articles is essential. The goal is to enable users to swiftly and conveniently access information that aligns with their interests. In this context, the development of a machine learning model for automated news article classification holds immense promise. It not only aids in identifying untracked news topics but also has the potential to offer personalized suggestions based on users' historical preferences.

*Approach to News Article Classification:*

**Data Collection and Pre-processing:** Gather a comprehensive dataset of news articles across various categories.

Pre-process the text data by removing stop words, punctuation, and other non-essential elements, and then apply stemming and lemmatization.

**Feature Extraction:** Convert the pre-processed text into vectors that can be used as input for machine learning algorithms. Techniques like TF-IDF or word embeddings (Word2Vec, GloVe) can be applied.

**Exploratory Analysis and Clustering:** Before diving into supervised learning, utilize unsupervised techniques like clustering (K-Means, DBSCAN) to group similar articles together. This step helps in understanding the natural groupings in the data and can be beneficial if there's unlabelled data.

**Model Selection and Training:** Use supervised machine learning algorithms to learn the relationship between the text and its respective category. Algorithms mentioned, such as:

Support Vector Machines (SVM): Particularly effective in high-dimensional spaces.

Neural Networks: Especially deep learning models like CNNs and RNNs which can capture sequential patterns in the text.

Random Forest: An ensemble method that can capture non-linear relationships.

**Model Evaluation:**

Split the dataset into training and testing (and possibly validation) sets.

Train the chosen models on the training set and evaluate their performance on the test set using metrics like accuracy, F1-score, precision, and recall.

**Model Optimization:**

Fine-tune model hyperparameters to optimize performance.

Implement techniques like cross-validation to ensure that the model generalizes well.

### 1.2.1 Objectives of the Project

The specific objectives delineated in this study encompass the following spheres:

**Enhanced User Experience:** In the age of information overload, readers are overwhelmed with vast amounts of content. By sorting news articles, platforms can streamline the content presentation, ensuring readers aren't inundated with irrelevant information. This optimizes the user journey, making navigation more intuitive and reading more enjoyable.

**Effective Content Discovery:** A sorted news system serves as a guidepost, helping users delve deeper into topics of interest. For instance, a user reading a political piece can easily navigate to more articles in the same category, fostering a deeper engagement and exploration of content.

**Personalized Recommendations:** With articles sorted into clear categories, machine learning algorithms can better analyze user behavior patterns. Based on a user's reading habits within specific categories, platforms can provide highly tailored article suggestions, enhancing content consumption.

**Ad Targeting:** Targeted advertising relies on precise user data. By analyzing which sorted news categories a user frequents, advertisers can serve highly relevant ads, increasing the likelihood of user engagement and conversion.

**Trend Analysis:** Over time, the influx of articles within certain categories can provide insights into societal shifts, interests, and global events. For instance, a surge in health-related articles might indicate a health crisis or a new wellness trend.

**Automated Workflows:** Manual categorization can be error-prone and time-consuming. Automated sorting, often using AI and ML, can process large volumes of articles quickly and consistently, freeing up human resources for more complex tasks.

**Improved Search Functionality:** For a user, a search function's efficiency is paramount. A sorted database provides more structured data for search algorithms, ensuring users get more accurate results faster.

**Data Analytics and Insights:** Beyond user behaviour, sorted articles can provide insights into the content itself. Analytics can reveal which categories get the most traction, the average reading time per category, and even the best time to publish for each category. Such insights can shape content strategies, publication timings, and promotional efforts.

## 1.2.2 Organization of Report

**Abstract :** Brief encapsulation of the report's purpose, methodologies, findings, and recommendations.

**Introduction:**

Background on the importance of news article sorting.

Statement of objectives and the scope of the report.

**The Need for Sorting:**

Discussion on the relevance and significance of categorizing news articles.

Overview of challenges faced in the absence of effective sorting.

**Data Acquisition:**

Description of data sources from which news articles are derived.

Information on the volume, variety, and characteristics of the articles to be sorted.

**Sorting Methodology:**

Detailed explanation of the algorithms, techniques, and tools used for sorting articles.

Discussion on manual versus automated sorting mechanisms.

Introduction to any machine learning or natural language processing tools utilized.

**Category Definitions:**

Outline of the categories or labels into which articles will be sorted.

Criteria or parameters for each category, detailing what qualifies an article for a specific label.

**System Architecture:**

Description of the overall structure and components of the sorting system.

Flowchart or diagrams showcasing the flow of information and decision processes.

**Implementation and Testing:**

Steps involved in implementing the sorting system.

Testing methodologies employed and the outcomes of these tests.

**Results and Insights:**

Metrics and measurements showcasing the accuracy and efficiency of the sorting system.

Any observed patterns, anomalies, or interesting findings during the sorting process.

**Real-world Applications:**

How this sorting system can be integrated into journalism and media platforms.

Benefits for readers, journalists, and media organizations.

**Challenges and Limitations:**

Identification of challenges faced during the project's execution.

Acknowledgment of system limitations and areas where it might not perform optimally.

**Recommendations for Improvement:**

Suggestions to enhance the sorting accuracy, speed, or user interface.

Proposed research or techniques that could be integrated into future iterations.

**Conclusion:**

Recapitulation of the report's primary findings and their broader implications.

A brief look forward to the future of news article sorting and classification.

**References:**

Comprehensive list of all resources, articles, tools, and datasets referenced during the report creation.

# CHAPTER 2
# BACKGROUND MATERIAL

## 2.1 Conceptual Overview

machine learning (ML) projects often follow a structured pipeline that starts from data collection to deployment.The relevant steps, emphasizing data and features, data processing, and other ML steps and technologies are covered here as background material.

News article sorting is an important task for many applications, including news aggregators, content recommendation systems, and search engines. The goal is to efficiently categorize and present news articles to users based on various criteria. A conceptual overview of news article sorting would touch upon the following aspects:

**Data collection from trusted source**

This involves gathering news from various outlets. The choice of sources matters because it dictates the quality and breadth of information. Reliable news agencies, independent journalism outlets, blogs, and even user-generated content platforms can be sources.It's essential to be wary of potential biases in sources and strive for a balanced collection. Additionally, some sources may require permissions or partnerships for data access.

**Data Pre-Processing**

Raw articles often contain extraneous information, such as ads, HTML tags, or metadata. Pre-processing involves cleaning this data to extract just the meaningful text. This stage also includes breaking down articles into tokens (words or phrases), removing common but uninformative words (stopwords), and standardizing word forms through stemming or lemmatization.

Over-cleaning might remove context, so it's a balance. The choice between stemming and lemmatization can impact results; lemmatization, though computationally more intensive, often retains more semantic meaning.

**Feature Extraction**

For algorithms to understand text, it needs to be translated into a numerical format. TF-IDF is a measure that assigns a weight to each term in a document, indicating its importance relative to a collection of documents. Word embeddings, on the other hand, convert words into vectors in a way that captures semantic relationships.The dimensionality of the feature set (especially with word embeddings) can be high, potentially requiring dimensionality reduction techniques for computational efficiency.

**Model**

A model is a computational representation that learns from data and is capable of classifying or making predictions based on its acquired knowledge in the field of machine learning (ML). Models serve as tools to extract patterns and relationships from input data and produce insightful results, and they are at the core of ML algorithms. In essence, a model captures the learned behaviour from training data and uses it to draw conclusions about brand-new, unobserved data.

**Future Directions**

The dynamic nature of news means sorting systems must adapt quickly. There's also the challenge of avoiding algorithmic biases, which can unintentionally favor or suppress certain content. As for the future, the field is moving towards hyper-personalized sorting, where individual user preferences play a significant role. Advanced NLP models like BERT or GPT also promise better context understanding and more accurate categorization. Personalization should not compromise user privacy. Additionally, relying heavily on advanced models can be computationally expensive and requires expertise.

### 2.1.1 Introduction to Text Classification

Text classification, often referred to as text categorization, is the task of assigning predefined categories (or labels) to a text based on its content. It's a standard task in natural language processing (NLP) and has various applications including spam detection, sentiment analysis, and topic labeling.

The realm of natural language processing (NLP) introduces us to the pivotal task of text classification. This task revolves around the assignment of predefined labels to textual documents, a process that bears significance in domains ranging from sentiment analysis to spam filtering. In a closely related context, text classification finds its application in identifying and categorizing articles, facilitating efficient moderation of online discourse.

At its core, text classification entails the training of models to discern intricate patterns and attributes inherent in text data, thereby enabling them to furnish precise predictions regarding the class to which a given document belongs. The journey commences with the construction of datasets, where each document is meticulously annotated with one or more labels. In the sphere of multi-label classification, the focal point of our exploration, documents may straddle multiple categories simultaneously.

Text classification methodologies serve to allocate distinct categories to natural language documents contingent on their intrinsic content. This might entail categorizing news items by their inherent subject matter or discerning book critiques based on their sentiment—be it favourable or unfavourable. Additionally, the ambit of text classification extends to detecting specific languages, structuring consumer feedback, and pinpointing instances of fraudulent activities.

Undertaking this task manually can be a laborious and prolonged endeavor. However, with the advent of machine learning algorithms, this procedure can be expedited and optimized.

Within the realm of news categorization, this challenge is often recognized as a multi-label text classification quandary. The primary objective here is to allocate one or multiple pertinent categories to a given news piece. A prevailing approach in navigating this multi-label conundrum is the deployment of an ensemble of binary classifiers.

**Key Concepts:**

- **Text Representation:** Before text data can be used for classification, it needs to be transformed into a numerical representation that machine learning algorithms can process. Common techniques include word embedding, which map words to dense vector spaces capturing semantic relationships.

- **Feature Extraction:** Feature engineering involves selecting relevant information from text data that can aid in classification. This can include n-grams, term frequency-inverse document frequency (TF-IDF), and more recently, contextual word embedding.

- **Model Training and Inference:** The process of training a text classification model involves presenting the model with labelled examples, adjusting its internal parameters to minimize classification errors. Inference is the process of applying the trained model to new, unseen data to predict labels.

### 2.1.2 News articles sorting

In today's digital age, vast amounts of data are stored electronically, necessitating tools that can interpret, analyse, and extract valuable insights for decision-making. Data mining is one such tool that can uncover hidden details from large databases. Before the last decade, timely news access was limited, but now it's readily available through online services. The abundance of textual information across various fields offers opportunities for analysis that can be beneficial in many contexts.

Text mining, especially the classification of text, is a complex field due to the need to transform unstructured data into structured information. With the influx of news articles, it has become challenging for users to find news tailored to their interests. As a result, there's a pressing need to categorize news, enhancing user navigation and accessibility. Categorizing online news helps users find their desired content quickly and efficiently. However, classifying news is challenging, especially as new stories may emerge that don't fit established categories or could even define a new one.

This article provides an overview of news classification methods focusing on content and headlines. Past methods, their effectiveness, and associated shortcomings are also explored.

Categorizing news articles is a task where a supervised learning approach is employed, allowing articles to receive category labels according to patterns recognized from a pre-labeled set of articles. One method to facilitate this sorting is by using a structured framework tailored for organizing news pieces into predetermined categories.

The Logistic Regression and Random Forest Classifier is a tool that we intend to utilize for this classification task. It's designed specifically for categorizing content with discrete features, like word counts in text classification. Even though it's fundamentally tailored for integer count data, in real-world applications, fractional counts (like tf-idf scores) can be applied with it as well.

To determine the probability of a document belonging to a particular category, we can draw upon the known distribution of documents across categories. This is referred to as calculating prior probabilities. In other words, when we are estimating the likelihood of an occurrence based on prior observations, we term it the "prior probability." If there's a subsequent occurrence post the original event, it is termed as the "posterior probability," hence the mention of "post" in posterior.

The news article categorization can also be achieved using other algorithms like unsupervised algorithm(clustering and association rule). Logistic Regression is a statistical model that uses a logistic function to model a binary dependent variable. Meanwhile, the Random Forest method uses a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. Both algorithms have their strengths and can be very effective in classifying news articles depending on the nature and distribution of the dataset.

## 2.2 Technologies Involved

### 2.2.1 Software Technology

**Jupyter Notebook (For Experiment):** An interactive web application called Jupyter Notebook is used for things like research, coding, and data analysis. It enables us to create documents that include text explanations, live code, and visualizations. We can add formatted text to ”markdown cells” and write and execute code in various programming languages inside of ”code cells.” The platform supports interactive reports and tutorials and makes it simple to share dynamic visualizations. Data scientists, researchers, and teachers like Jupyter Notebook because of its adaptability and collaborative features.

**Visual Studio Code (For Production-Coding):** The user-friendly interface of Visual Studio Code comes with a variety of tools like syntax highlighting, code completion, and built-in Git support. Extensions that improve functionality for different programming languages, frameworks, and tools make it highly customizable. Because of its speed, flexibility, and ability to work in various coding environments, VS Code is a favourite among programmers and developers. It also offers debugging features, an integrated terminal, and a rich ecosystem of extensions

**Flask (For Production-Application):** Python's Flask web framework is nimble and adaptable, making it easier to create web applications. It offers the fundamentals, such as routing, request handling, and template rendering, for creating web applications. Flask is a well-liked option for projects where flexibility and simplicity are essential because of its minimalist design, which enables developers to quickly create web apps without imposing rigid structures. We can easily build anything with Flask, from simple websites to intricate web applications and APIs.

### 2.2.2 Hardware Technologies

| | |
|---|---|
| **Device name** | LAPTOP-4JBJG5LK |
| **Processor** | Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz |
| **Installed RAM** | 8.00 GB (7.89 GB usable) |
| **Device ID** | F163F3F2-DFF9-4737-B164-B4F44E5C2272 |
| **Product ID** | 00327-35147-63962-AAOEM |
| **System type** | 64-bit operating system, x64-based processor |
| **Edition** | Windows 11 Home Single Language |
| **Version** | 22H2 |

# CHAPTER 3

# METHODOLOGY

The systematic and organized approach or set of practices that direct the conduct of research, studies, or projects is referred to as methodology. It describes the procedures, methods, equipment, and tactics employed in the data collection, information analysis, and conclusion drawing processes. In a variety of disciplines, including science, the social sciences, engineering, and more, methodology is essential for ensuring the accuracy, dependability, and reproducibility of results. It offers a precise framework that practitioners and researchers can use to achieve well-defined goals and produce results that are reliable and valid.

## Process in Methodology

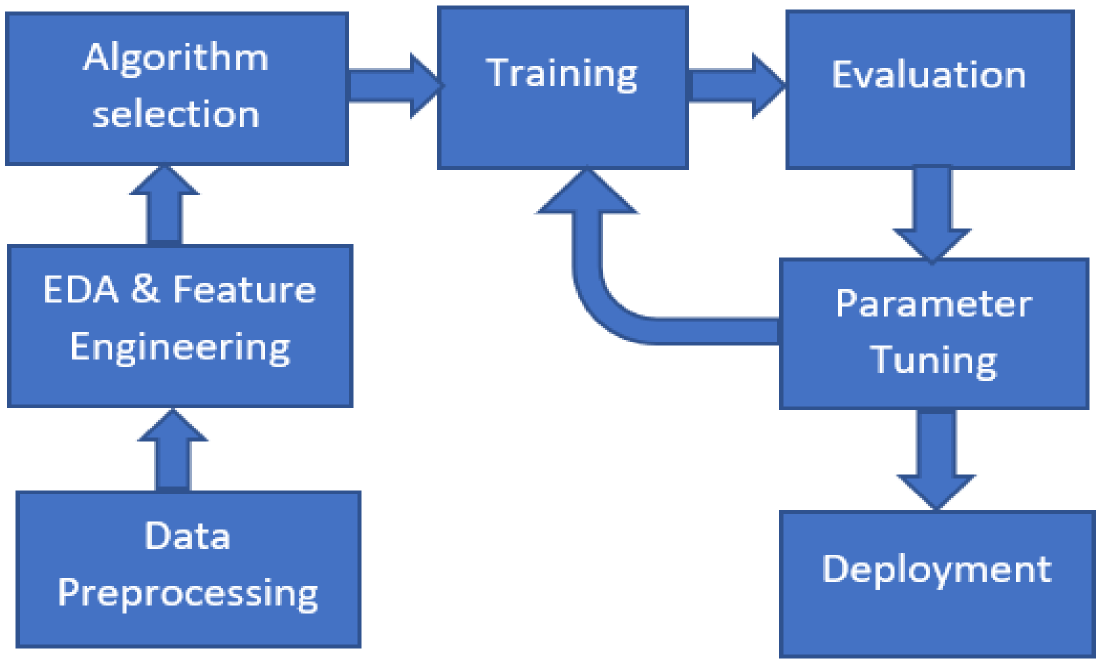


Fig No. 3.1 Steps in Methodology

## 3.1 News Article Classification: A Structured Approach

Classifying news articles is an intricate process, given the inherent challenges in transforming raw, unstructured textual data into a refined, structured format conducive for analysis. The process of text classification, particularly for news articles, entails several pivotal stages.

The process of classifying news involves several steps. Classification is a challenging task since it needs to be pre-processed in order to convert textual material from an unstructured form to a structured form. The primary steps in the text classification process for news articles are as follows. These include gathering data, pre-processing it, choosing features, applying classification techniques, and assessing performance metrics.

## News articles Classification Process Workflow :

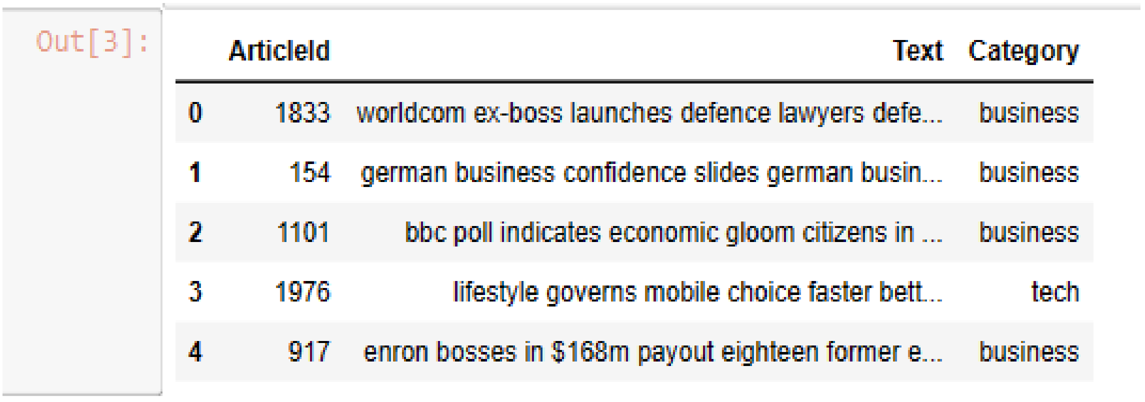
**3.1.1 Data collection :** The collection of data from multiple sources is the initial step in news classification. There are several places where you can find this information, including the World Wide Web, radio, television, newspapers, and magazines. However, with the expansion of the internet and information technology, it has become the primary source for news. Data may be made available in any format, including HTML, PDF, and Doc.

Data acquisition, often the first step in a data analysis pipeline, refers to the process of collecting, importing, and cleaning data for further use in research or analysis. The quality and method of data collection play a critical role in the outcomes of the analysis.

The data was sourced from the BBC News dataset, consisting of 1490 articles spread across categories like business, tech, politics, sports, and entertainment. The dataset had three columns: “ArticleId”, “Text”, and “Category”.

**Dataset Overview and Structure**

For the endeavor of automating news classification through machine learning, we've procured a dataset from Kaggle. This dataset, a rich compilation of news articles, offers both the headlines and the body text of these articles, making it a comprehensive source for our analysis.


Out[3]:

|   | ArticleId | Text | Category |
|---|---|---|---|
| 0 | 1833 | worldcom ex-boss launches defence lawyers defe... | business |
| 1 | 154 | german business confidence slides german busin... | business |
| 2 | 1101 | bbc poll indicates economic gloom citizens in ... | business |
| 3 | 1976 | lifestyle governs mobile choice faster bett... | tech |
| 4 | 917 | enron bosses in $168m payout eighteen former e... | business |

Fig No. 3.2 Dataset Overview

Dataset Structure:
Here's a succinct breakdown of the dataset's structure:

**Article Id:**

*Description:* A unique identifier assigned to each news article in the dataset.

*Type:* Numeric/Integer

*Usage:* Helps in differentiating articles and can be instrumental in tracking or referencing specific records.

**Article:**

*Description:* This field contains the concatenated version of the headline and the main content of the news article.

*Type:* Text/String

Usage: The primary source of data for our analysis. This field will be processed, tokenized, and vectorized to serve as the main input for our classification models.

**Category:**

*Description:* This denotes the category or genre of the news article. It's a categorical variable with predefined classes.

*Classes:* tech, business, sport, entertainment, politics.

*Type:* Categorical/String

*Usage:* Acts as the target label for our classification task. Our machine learning model will be trained to predict this field based on the content in the 'Article' field.

*Preliminary observations:*
Given the nature of the dataset, preprocessing will play a pivotal role. The 'Article' field, which is a combination of headlines and body content, will require comprehensive text processing and cleaning to ensure efficient model training. The 'Category' field, being categorical, might be encoded to a numerical format if required by specific machine learning algorithms.

In conclusion, this dataset offers a robust foundation for the news classification task. The next steps would involve data exploration, preprocessing, feature extraction, model training, and evaluation.

### 3.1.2 Text Data Pre-Processing:

After the pre-processing of the news text has been completed. Since this data is compiled from a range of sources, it must be cleaned in order to be free of errors and useless information. In order to distinguish data from unrelated terms like semicolons, commas, double quotes, full stops, and special characters, etc., discrimination is now required. Stop words, which are terms that frequently exist in text, are removed from data.

An essential step, preprocessing involves refining the raw data to make it amenable for further analysis. This phase encompasses tasks like tokenization, where text is split into individual words or terms, removal of HTML tags, discarding stopwords which might not add substantive value for classification, and purging any non-alphabetic characters.

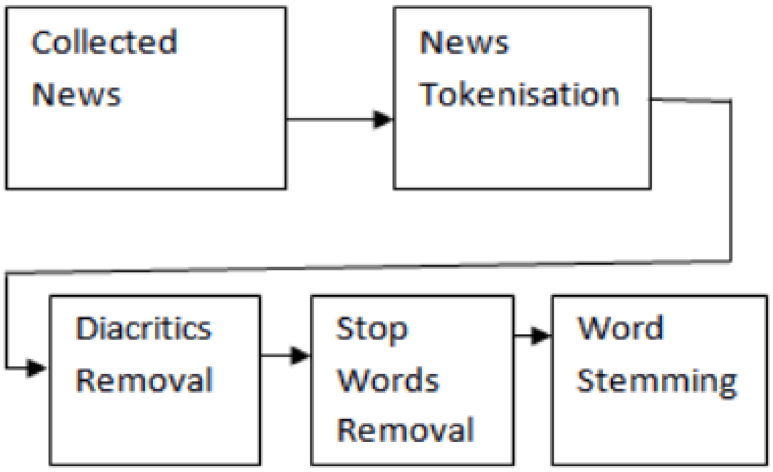


Fig No. 3.3  News Article Pre-Processing

Pre-processing is an integral step in data analysis, particularly in the domain of machine learning and natural language processing (NLP). Proper preprocessing ensures that data is ready and optimized for further analysis or feeding into algorithms. Let's delve deeper into preprocessing, focusing primarily on NLP but also touching on general data preprocessing.

## Tokenization:

Tokenization is the process of converting a sequence of text into individual tokens (typically words, subwords, or phrases). In simpler terms, it's like breaking up a sentence into the words that compose it.

Tokenization is a foundational step in many NLP tasks, transforming raw text into a format that's easier to analyze and process. It acts as a bridge, facilitating the transition from reading human language to machine-understandable format. The choice of tokenization method often depends on the specific requirements of the NLP task at hand.

Types of Tokenization:

**Word Tokenization:**

Splits text into individual words.

Example: "I love ice cream." -> ["I", "love", "ice", "cream"]

**Sentence Tokenization (or Sentence Segmentation):**

Divides text into individual sentences.

Example: "It's a sunny day. Let's go outside!" -> ["It's a sunny day.", "Let's go outside!"]

**Character Tokenization:**

Decomposes text into individual characters.

Example: "cat" -> ["c", "a", "t"]

Common Tools and Libraries:

NLTK (Natural Language Toolkit): Offers both word and sentence tokenization functions.

Spacy: A comprehensive NLP library that provides tokenization among other functionalities.

OpenNLP: Offers tokenization functionalities along with other NLP tasks.

SentencePiece: A neural-network-based tokenizer that's language-agnostic and particularly useful for subword tokenization.

## Lowercasing :

The concept of lowercasing in the context of text preprocessing, particularly within Natural Language Processing (NLP) and text analytics. Lowercasing is the process of converting all characters in a text to their lowercase equivalents. For instance, the word "Text" becomes "text", and the sentence "This Is An Example." becomes "this is an example."

Lowercasing is a foundational step in many text preprocessing pipelines, ensuring a consistent and simplified representation of text. While it offers several advantages in reducing redundancy and improving the efficiency of subsequent tasks, care must be taken in scenarios where case carries specific semantic meaning. The decision to apply lowercasing should ideally be based on the specific requirements and context of the NLP task at hand.

## Stopword Removal :

The stop words are linguistically distinctive and bear no meaning. Conjunctions, pronouns, and prepositions are typically included. They are finally deleted since they are thought to be of little value. Before data is processed, these words must percolate.

There are various ways to remove stop words from data. The removal will be of the terms that offer very little insight into classification, therefore it may be based on concepts.

Removing words off the list of English stop words is another method for getting rid of stop words. The list, which includes around 545 stop words, is made available by the Journal of Machine Learning Research. Depending on how frequently they are used, stop words may also be eliminated. With this approach, the frequency of occurrence of words is calculated before weights are given to them. The stop words are then omitted based on these weights.

Importance of Stopword Removal:

*Noise Reduction:* In many NLP tasks, stopwords can introduce unnecessary noise since they occur frequently across multiple documents but often don't carry significant meaning in relation to the content.

*Efficiency:* Removing stopwords reduces the size of the textual data, leading to faster processing and less computational resource usage.

*Dimensionality Reduction:* In vector space models like Bag of Words or TF-IDF, excluding stopwords can considerably reduce the number of dimensions (or features), making subsequent tasks more efficient.

*Improving Relevance:* In information retrieval systems, omitting stopwords can lead to more relevant search results.

**Implementation:**

Predefined Lists: Many NLP libraries provide predefined stopword lists for various languages:

NLTK: The Natural Language Toolkit in Python offers a list of stopwords for several languages.

Spacy: This NLP library also provides stopwords as part of its language models.

Scikit-learn: Offers an English stopword list as part of its feature_extraction.text module.

Custom Lists: Depending on the specificity of the domain, users might create and use their custom stopword lists.

Dynamic Lists: By analyzing the distribution of word frequencies in a corpus, it's possible to identify and define stopwords dynamically. Words that appear with extremely high frequency across documents might be treated as stopwords.

**When to Remove Stopwords:**

Information Retrieval & Search Engines: Removing stopwords can improve search efficiency.

Text Classification and Clustering: Especially when the focus is on keywords or domain-specific terms.

Topic Modeling: Helps in getting more meaningful topics.

However, for tasks like machine translation, question-answering systems, or other sequence-dependent tasks, stopwords might carry important syntactical information and shouldn't be indiscriminately removed.

Procedure for Removal:

*Tokenize the Text:* Break down the text into individual tokens (typically words).

*Filtering:* Remove the words that are present in the stopword list from the tokens.

*Reconstruct (if necessary):* If the processed text needs to be in sentence form (as opposed to a list of tokens), the filtered tokens can be joined back together.

## Word Stemming :

Stemming is the next task that is carried out once stop words are eliminated. By taking a term down to its root, this process. The purpose of stemming is to get rid of suffixes in order to reduce the amount of words. For instance, the term "USE" can be used to replace words like user, users, utilised, and using.

As a result, less time and space will be needed.

There are numerous stemmers available for stemming, including S-Stemmers, Lovins Stemmer, Porter Stemmer, and Paice/Husk Stemmer. M.F. Porter is the most frequently used of these stemmers.

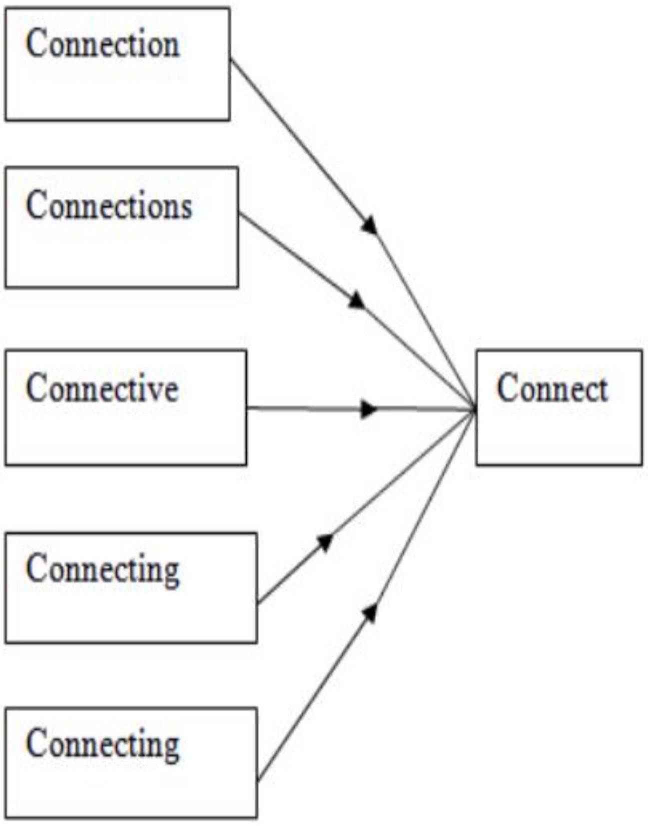


Fig No.3.4 Stemming Process

Word stemming involves reducing words to their base or root form. This means that the derived variants of a word will be transformed into their base form. For instance, stemming would transform "running", "runner", and "ran" to the common root "run".

**Common Stemming Algorithms:**

*S-Stemmer:* This stemmer is useful for words longer than three letters. This stemmer's goal is to think about both singular and plural forms of news.

*Lovins Stemmer:* The Lovins Stemmer was the initial stemmer that was suggested. The quickness of Lovins' stemmer gives it an advantage over a number of other stemmers. Compared to other stemmers, it is speedier. There are 35 transformation rules, 294 endings, and 29 conditions in this stemmer. The 35 rules are applied on the ending after the longest endings that satisfy the condition are initially found and deleted.

*Porter Stemmer:* Due to its high precision and straightforward algorithm, Porter Stemmer is the most popular stemmer. It consists of five simple procedures that can be applied to each and every word.

*Paice/Husk stemmer:* The Paice/Husk stemmer uses the same rules and suffixes for each loop of an algorithm that is based on iteration. Each rule consists of five steps, three of which must be followed without exception while the other two are optional.

## Lemmatization:

Lemmatization refers to the process of reducing a word to its base or dictionary form. For example, the words "running," "runs," and "ran" are all forms of the verb "run." The process of lemmatization would transform each of these variants back to the base word "run.

In NLP, lemmatization is a complex yet effective approach for text normalisation. Compared to stemming, it frequently provides results that are more semantically accurate. Stemming or lemmatization should be chosen depending on the particular requirements of the work at hand, striking a balance between computing efficiency and the accuracy of representation.

## Stemming vs. Lemmatization:

While stemming chops off the ends of words to find the root form, lemmatization involves looking up a word in a lexicon and returning its base or dictionary form. Lemmatization is generally more accurate but computationally expensive compared to stemming. For example, while stemming might reduce "better" to "better" or "good", lemmatization would correctly reduce it to "good".

**Common Tools and Libraries:**

*NLTK* (Natural Language Toolkit): Offers both word and sentence tokenization functions.

*Spacy:* A comprehensive NLP library that provides tokenization among other functionalities.

*OpenNLP:* Offers tokenization functionalities along with other NLP tasks.

*SentencePiece:* A neural-network-based tokenizer that's language-agnostic and particularly useful for subword tokenization.

### 3.1.3 Feature Selection

In order to overcome these problems, a process known as feature selection is used, in which only those relevant and highly effective features are chosen, which may prove more noticeable for better news classification. When there are many features and each feature is a well-known descriptive word for each class, a lot of time may be needed in classification, and it is possible that expected accuracy may not be achieved. There are numerous methods in the literature for choosing relevant characteristics, including

Boolean weighting and Class Frequency Information Gain, Term Frequency Inverse Class Frequency, and Thresh Holding

Boolean Weighting: This is a basic method where words or features are represented as binary values: present (1) or not present (0). This method doesn't consider the frequency of the term in the documents but only its presence or absence.

Class Frequency Thresholding: In this method, features are selected based on their frequency in a particular class compared to other classes. Features that are highly frequent in one class and less frequent in others can be considered as distinguishing features for that class.

Term Frequency Inverse Class Frequency (TF-ICF): It's an extension of the popular TF-IDF (Term Frequency-Inverse Document Frequency). While TF-IDF considers the rarity of a term across all documents, TF-ICF considers the rarity of a term across all classes. Features with high TF-ICF are more class-specific.

Information Gain: Information Gain measures the reduction in uncertainty or entropy when a feature is used for classification. A feature with higher information gain can provide more clarity in distinguishing between classes.

Mutual Information: This measures the dependency between the feature and the class labels. A higher mutual information value means that the presence (or absence) of that feature provides more information about the class label.

Chi-square Test: It evaluates the independence of features and class labels. Features that are highly dependent on the class labels can be considered more relevant for classification.

Correlation Coefficient: Measures the relationship between a feature and the class label. A high positive or negative correlation indicates a strong relationship, making the feature important for classification.

In feature selection, it's essential to consider the domain of application and the nature of the data. For instance, when dealing with news articles, certain features might be more telling about the content and category of the news. The goal of feature selection is not just to reduce the dimensionality of the data but to retain those features that hold the most discriminative power for the task at hand. Proper feature selection can lead to improved classification accuracy, reduced overfitting, and faster training times.

Finally, it's important to remember that the best feature selection method may vary depending on the dataset and the specific problem you're trying to solve. It's often beneficial to experiment with different methods and even combinations of methods to find the most suitable approach for your dataset.

### 3.1.4 Feature Engineering

Feature engineering is the process of transforming raw data into features that can better represent the underlying patterns in the data to predictive models. In the context of

Natural Language Processing (NLP), feature engineering plays a crucial role because raw text data cannot be directly fed into algorithms. Let's delve into the two methods you mentioned: Bag of Words and TF-IDF.

## *Bag of Words (BoW) model :*

Bag of words is a text modelling method used in natural language processing. It is a technique for feature extraction from text data, to put it technically. This method is a straightforward and adaptable way to extract features from documents.

A textual illustration of word recurrence in a document is called a "bag of words." We don't pay attention to grammatical conventions or word order; we only keep track of word counts. It is referred to as a "bag" of words because any details regarding the arrangement or structure of the words within the document are ignored. The model doesn't care where in the document recognised terms appear; it is only interested in whether they do.

Vocabulary Creation: Create a vocabulary of unique words from the entire set of documents.

Vectorization: For each document, create a vector where each position corresponds to a word in the vocabulary. The value at each position is the frequency of that word in the document.

## *Term Frequency-Inverse Document Frequency (TF-IDF) Model :*

TF-IDF is an improvement over the BoW model. While BoW gives equal weight to all words, TF-IDF gives more weight to words that are frequent in a specific document but not in all documents, capturing some of the context and meaning.

The BOW model contains a flaw, which causes it to produce subpar outcomes. Consider a scenario in which a specific word appears in every document and does so repeatedly. Over time, this word will occur more frequently and have a higher value, which will make it appear in a sentence more frequently. This is bad for our analysis.

Number of times a term has appeared in a document, or term frequency (TF). The word "frequency" describes how often or how "common" a word appears in a sentence.

The inverse document frequency (IDF) is a metric used to assess how uncommon a term is in a document. terms like "the" and "a" appear in every text, but uncommon terms won't be found in every document in the corpus.

Almost universal use of a word indicates that it is not important for classification.

A word's IDF equals log(N/n).

N stands for all documents together. n: the quantity of documents that contain a term (word).

The TF-IDF evaluates a word's relevance to its sentence within a group of sentences or documents.

### 3.1.5 Model Building

Model building, in the context of machine learning and statistical analysis, refers to the systematic process of designing, training, and validating algorithms that can discern patterns, make predictions, or take decisions based on data. It encompasses selecting the appropriate algorithm, feeding it with data, adjusting its parameters (often called hyperparameters), and evaluating its performance against known outcomes. The ultimate goal of model building is to construct a model that generalizes well to new, unseen data, striking a balance between capturing the underlying patterns (fitting) and avoiding being overly complex (overfitting). This iterative process serves as the bedrock for many applications in data science, artificial intelligence, and analytics, driving insights and actions in various domains ranging from finance and healthcare to retail and technology.

**Logistic Regression :**

Logistic Regression is a widely used statistical method in the field of machine learning, primarily employed for binary classification tasks. Despite its name, it's more of a classification algorithm than a regression one. It's particularly suitable when the dependent variable is categorical (binary), and you're interested in predicting the probability of an event occurring.

logistic regression, is classification tasks where the objective is to estimate the likelihood that a given instance belongs to a particular class. Its term is logistic regression, and it is utilised for classification methods. Regression is used because it uses a sigmoid function to estimate the probability for the given class using the output of a linear regression function as input. Logistic regression differs from linear regression in that it predicts the likelihood that an instance will belong to a specific class or not, whereas the output of the former is a continuous value that can be anything.

- In a categorical dependent variable, the output is predicted via logistic regression. As a result, the result must be a discrete or categorical value.
- Rather of providing the exact values of 0 and 1, it provides the probabilistic values that fall between 0 and 1. It can be either Yes or No, 0 or 1, true or false, etc.
- With the exception of how they are applied, logistic regression and linear regression are very similar. While logistic regression is used to solve classification difficulties, linear regression is used to solve regression problems.

- In logistic regression, we fit a "S" shaped logistic function, which predicts two maximum values (0 or 1), rather than a regression line.
- The logistic function's curve shows the possibility of several things, including whether or not the cells are malignant, whether or not a mouse is obese depending on its weight, etc.
- Because it can classify new data using both continuous and discrete datasets, logistic regression is a key machine learning algorithm.
- When classifying observations using various sources of data, logistic regression can be used to quickly identify the factors that will work well.

The core idea behind logistic regression lies in transforming the output of a linear equation into a value between 0 and 1 using the logistic function (sigmoid function). The equation looks like:

P(Y=1|X)=1/1+e−z

Where P(Y=1|X) is the probability of the positive class, X represents the input features, and z is the linear combination of feature weights and inputs:

z=β0+β1x1+β2x2+…+βnxn

The logistic function ensures the output lies between 0 and 1, thus representing probabilities.

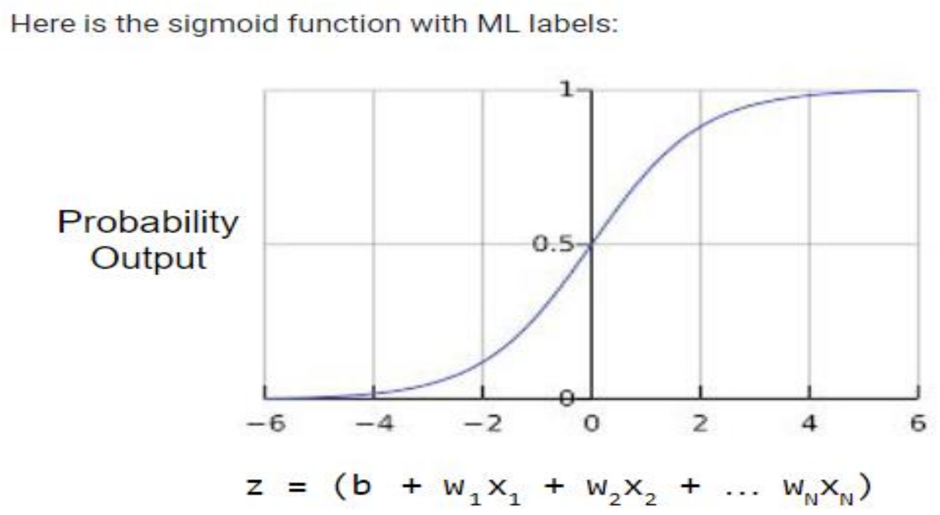


Fig No. 3.5 Sigmoidal Function

**Random Forest :**

Random Forest is a popular ensemble learning technique that builds multiple decision trees during training and outputs the mode of the classes for classification tasks or the mean/median prediction of individual trees for regression tasks. Let's delve into its mechanics, advantages, and limitations.

The idea behind Random Forest is to combine the predictions of several base estimators (in this case, decision trees) to improve generalizability and robustness over a single estimator. The term "random" arises from it introducing randomness in two ways:

Bootstrapped datasets for training individual trees.

Random subsets of features considered for splitting at each node.

The Random Forest Algorithm's ability to handle data sets with continuous variables, as in regression, and categorical variables, as in classification, is one of its most crucial qualities. For classification and regression tasks, it performs better.

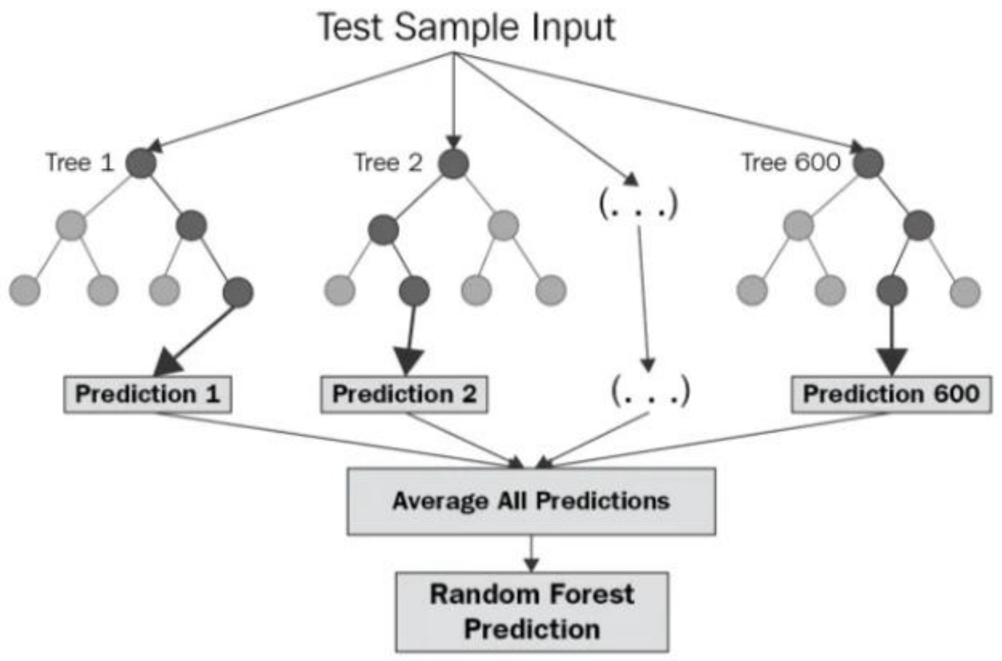


Fig No. 3.6 Random Forest workflow

Hyperparameters:

Some important hyperparameters in Random Forest include:

n_estimators: Number of trees in the forest.

max_features: Max number of features considered for splitting a node.

max_depth: Max number of levels in each decision tree.

min_samples_split: Min number of data points placed in a node before the node is split.65

min_samples_leaf: Min number of data points allowed in a leaf node.

Parameter Tuning:

Parameter tuning, also known as parameter optimization, is the process of selecting the best combination of parameters for a machine learning algorithm to achieve optimal performance on a given dataset. Parameters are parameters that are set before training a model and control aspects of the training process, model complexity, and regularization.

There're various types of parameter tuning, the most used parameter tunings are :

- Grid Search: In order to find the combination of parameters that produces the best model performance, the parameter tuning technique known as "Grid Search" is used in machine learning. It is a simple and thorough method that considers every combination within the given parameter space.

- Random Search: Another parameter-tuning method used in machine learning to identify the best parameter combinations for models is random search. Random Search selects parameter values at random from the designated ranges, in contrast to Grid Search, which thoroughly investigates every possible combination within a predefined range. While still producing competitive results, this strategy may be more resource-efficient in terms of computation.

## Deployment:

Making our trained model accessible and usable in a real-world setting where users can interact with it is what deployment of a machine learning (ML) project entails. Our ML model typically transitions from a development or testing environment to a production environment using this process.

When it comes to the lifecycle of a data-driven application, one of the most important phases is the process of deploying a machine learning (ML) project. This is because the process turns a welltrained model into a tool that is both functional and easily accessible for end-users. When you want to deploy a machine learning model, you have to move it from the controlled environment of development or testing into a production environment that is robust and scalable. In this environment, it will be able to interact with data from the real world and cater to the requirements of users and stakeholders.

In conclusion, the process of deploying a machine learning project helps bridge the gap between theoretical understanding and actual practical application. It is necessary to take a holistic approach that takes into consideration a number of different aspects, including the technical implementation, the user experience, and the ongoing maintenance. If an organization is able to pull off a successful deployment, they will be able to harness the power of machine learning and use it to make more informed decisions, streamline their processes, and improve the overall user experience.

## 3.2 Circuit Layouts

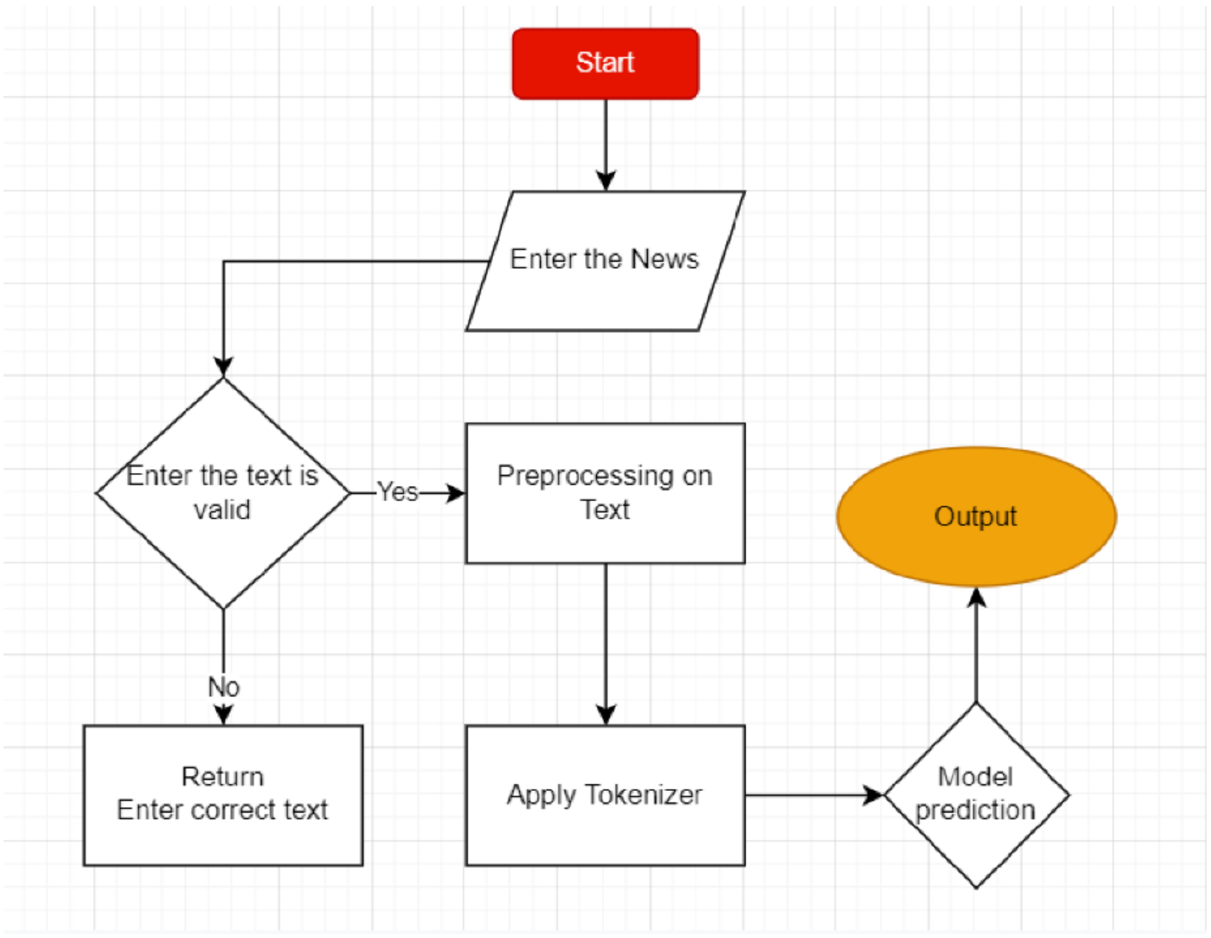


Fig No. 3.6 Circuit Layout

# CHAPTER 4

# IMPLEMENTATION

Refers to the phase in a project where theoretical designs and plans are turned into reality. It involves executing the project according to its plans and ensuring that the proposed solution or system is built and made operational. This phase often follows the design or planning phase and precedes the testing and evaluation phase.

his project’s implementation requires a number of different pipelines and components.Data ingestion, data transformation, model training, parameter tuning, and other types of components have all been used in this, as well as two different pipelines for training and prediction

## 4.1 Modules

In computing and software design, the term "module" refers to a self-contained unit that can be used to break down a larger system, making it more manageable and modular. The concept of modularity allows for improved reusability, maintainability, and clarity in software and system design.

**In Programming:** Many programming languages support the creation and use of modules. For instance, in Python, you can create separate .py files, each containing related functions and classes, and then import them using the import statement.

### 4.1.1 Environment Setup

At first one virtual environment has been created. After that some important files have been created such as: **setup.py** (for entire project), **logger.py** (for logging everything), exception.py (for handling exception) etc.

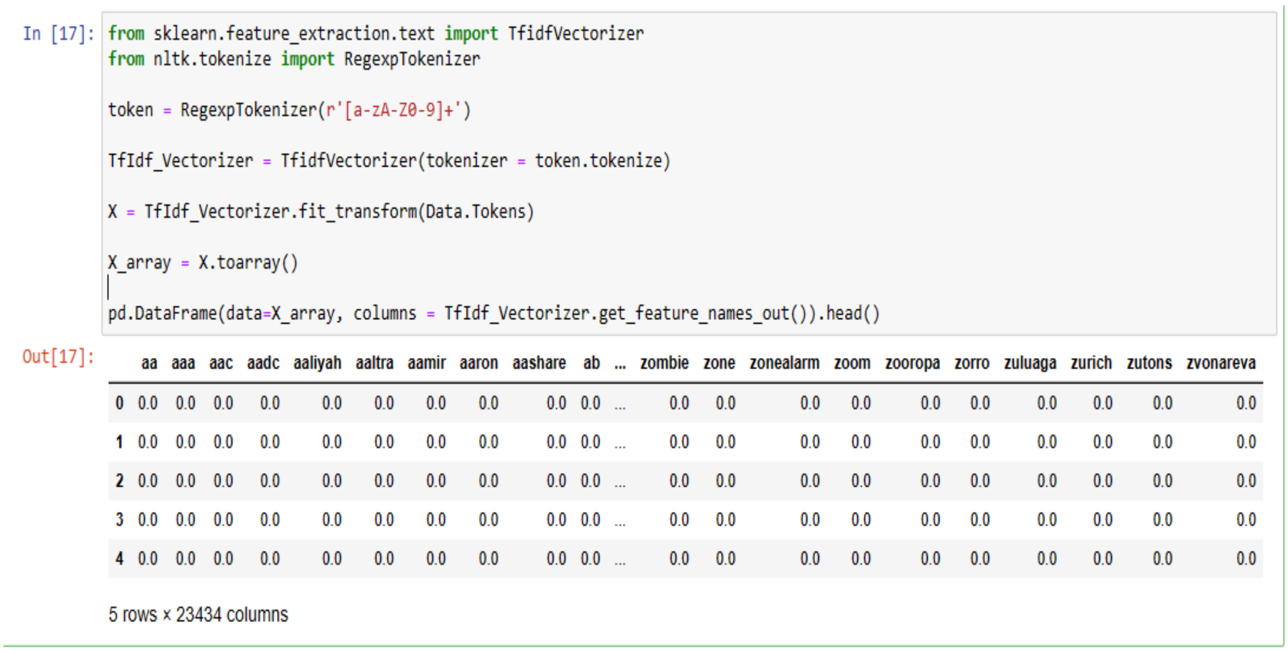
We also have to download the required libraries before starting the coding part. In this project I’ve some necessary libraries such as: **Pandas**, **Numpy**, **Matplotlib**, **flask** etc. Now we’re ready to build the core part.

### 4.1.2 Data Transformation

Here, importing necessary libraries: **TfidfVectorizer** from **sklearn.feature_extraction.text, RegexpTokenizer** from **nltk.tokenize**, and pandas for data manipulation and display.

Creating a **RegexpTokenizer** object named token. This tokenizer uses a regular expression pattern **(r'[a-zA-Z0-9]+')** to match and tokenize words composed of letters (both uppercase and lowercase) and digits.

Creating a **TfidfVectorizer** object named **TfIdf_Vectorizer**. This is a part of the Scikit-Learn library and is used to convert a collection of text documents into a matrix of TF-IDF features. The tokenizer parameter is set to the **token.tokenize** function, which will tokenize the input text data using the **RegexpTokenizer** you defined earlier.



```
In [17]: from sklearn.feature_extraction.text import TfidfVectorizer
         from nltk.tokenize import RegexpTokenizer

         token = RegexpTokenizer(r'[a-zA-Z0-9]+')

         TfIdf_Vectorizer = TfidfVectorizer(tokenizer = token.tokenize)

         X = TfIdf_Vectorizer.fit_transform(Data.Tokens)

         X_array = X.toarray()

         pd.DataFrame(data=X_array, columns = TfIdf_Vectorizer.get_feature_names_out()).head()
```

```
Out[17]:
     aa  aaa  aac  aadc  aaliyah  aaltra  aamir  aaron  aashare   ab  ...  zombie  zone  zonealarm  zoom  zooropa  zorro  zuluaga  zurich  zutons  zvonareva
  0 0.0  0.0  0.0   0.0      0.0     0.0    0.0    0.0      0.0  0.0  ...     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  0.0   0.0      0.0     0.0    0.0    0.0      0.0  0.0  ...     0.0   0.0        0.0   0.0      0.0    0.0      0.0     0.0     0.0        0.0
  2 0.0  0.0  0.0   0.0      0.0     0.0    0.0    0.0      0.0  0.0  ...     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  0.0   0.0      0.0     0.0    0.0    0.0      0.0  0.0  ...     0.0   0.0        0.0   0.0      0.0    0.0      0.0     0.0     0.0        0.0
  4 0.0  0.0  0.0   0.0      0.0     0.0    0.0    0.0      0.0  0.0  ...     0.0   0.0        0.0   0.0      0.0    0.0      0.0     0.0     0.0        0.0

5 rows × 23434 columns
```

Fig No. 4.1 Data Transformation

Using the **fit_transform** method of the **TfidfVectorizer** to transform a collection of text data. It seems that **Data.Tokens** is a data structure (perhaps a DataFrame column) containing pre-tokenized text data. This operation calculates the **TF-IDF** values for the tokens in the text data and returns a sparse matrix X representing the TF-IDF features.

Finally, creating a Pandas DataFrame using the dense array **X_array** and assigning column names using **TfIdf_Vectorizer.get_feature_names_out(),** which gets the feature (token) names corresponding to the columns. **.head()** is used to display the first few rows of the DataFrame.

In summary, this code takes pre-tokenized text data, calculates TF-IDF values for the tokens, and presents the results in a DataFrame where each row corresponds to a document and each column corresponds to a token (word) with its corresponding TF-IDF value. This is useful for various natural language processing tasks, such as text classification, clustering, and more.

### 4.1.3 Model Building

Imports several functions and classes from the scikit-learn library, a popular machine learning library in Python, and then uses them to split a dataset into training and test sets. Here's a breakdown:

**Import Statements:**

**train_test_split:** A function used to split arrays or matrices into random train and test subsets.

**cross_val_score:** A function that evaluates a model using cross-validation.

classification_report, confusion_matrix, accuracy_score: Functions used for assessing the performance of classification models.

**Splitting Data:**

X and y are the feature matrix and target vector respectively. They have not been defined in the provided snippet, but we can assume they have been defined earlier in the code.

**test_size**=.33 means that 33% of the data will be used for the test set, while the remaining 67% will be used for the training set.

**random_state**=40 ensures that the splits generate the same result every time the code is run. If you don't use the random_state parameter, every time you run the code, you'll potentially get a different split of data, which can make it hard to reproduce results.

After running this code:

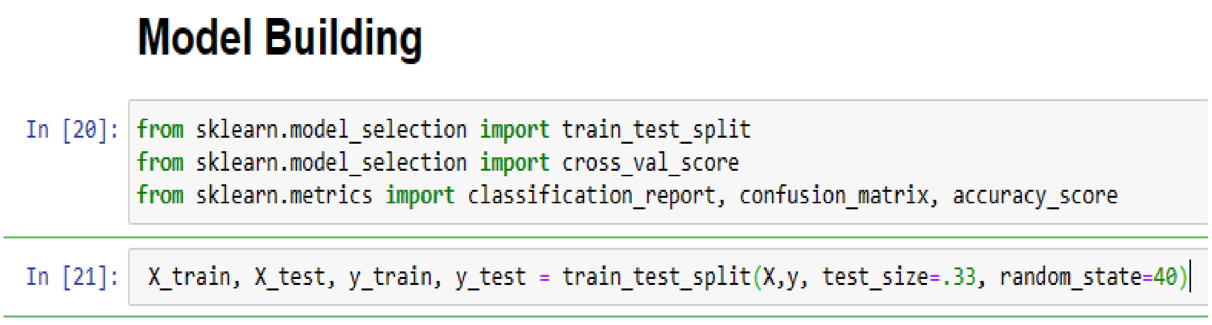
X_train will contain 67% of the samples from X.

X_test will contain the remaining 33% of the samples from X.

y_train will contain the corresponding target values for X_train.

y_test will contain the corresponding target values for X_test.

After splitting the data, you'd typically fit a model on the training set and then evaluate its performance on the test set using metrics like accuracy, confusion matrix, etc.



Model Building

```
In [20]: from sklearn.model_selection import train_test_split
         from sklearn.model_selection import cross_val_score
         from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

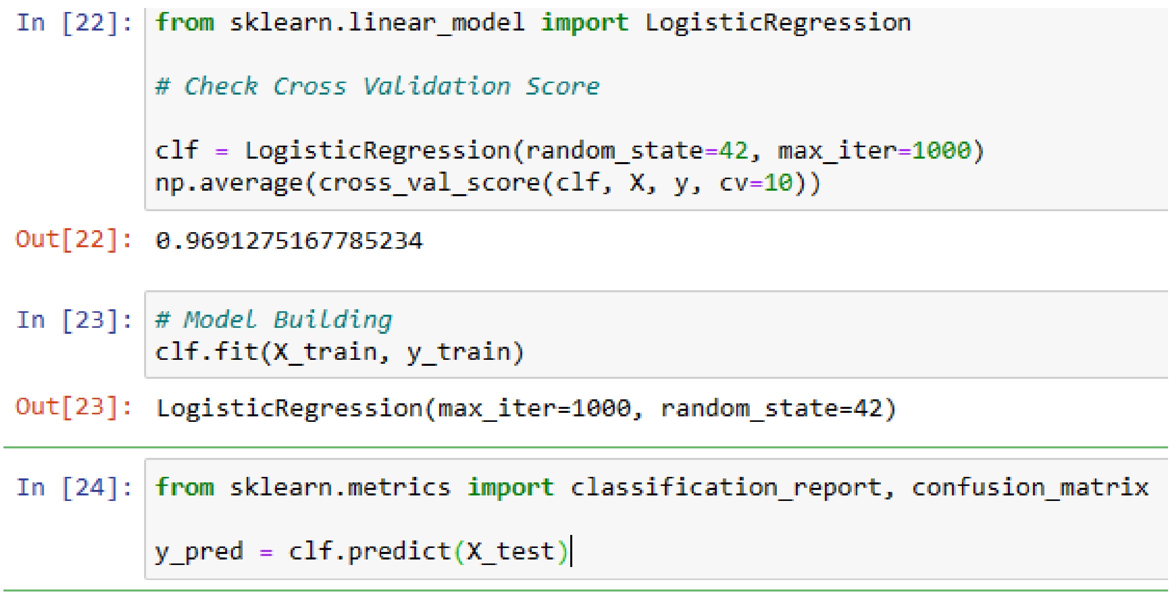
```
In [21]: X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=.33, random_state=40)
```

Fig No. 4.2  Model Building

### 4.1.4 Model Train

#### Logistic Regression

Imports the **LogisticRegression** class from scikit-learn, which is used for logistic regression modeling, a common technique for binary and multiclass classification problems.



```
In [22]: from sklearn.linear_model import LogisticRegression

         # Check Cross Validation Score

         clf = LogisticRegression(random_state=42, max_iter=1000)
         np.average(cross_val_score(clf, X, y, cv=10))

Out[22]: 0.9691275167785234

In [23]: # Model Building
         clf.fit(X_train, y_train)

Out[23]: LogisticRegression(max_iter=1000, random_state=42)

In [24]: from sklearn.metrics import classification_report, confusion_matrix

         y_pred = clf.predict(X_test)
```

Fig No. 4.5 Model Train of Logistic Regression

The **fit()** method is used to train the logistic regression model **(clf)** on the training data. The **X_train** matrix contains the features for each training example, while **y_train** contains the corresponding labels or target values.

**classification_report:** Generates a report showing the main classification metrics such as precision, recall, f1-score, etc., for each class.

**confusion_matrix:** Produces a confusion matrix which gives a more detailed breakdown of correct and incorrect classifications for each class.

### Randon Forest

Implementing and evaluating a random forest classifier using **scikit-learn.**

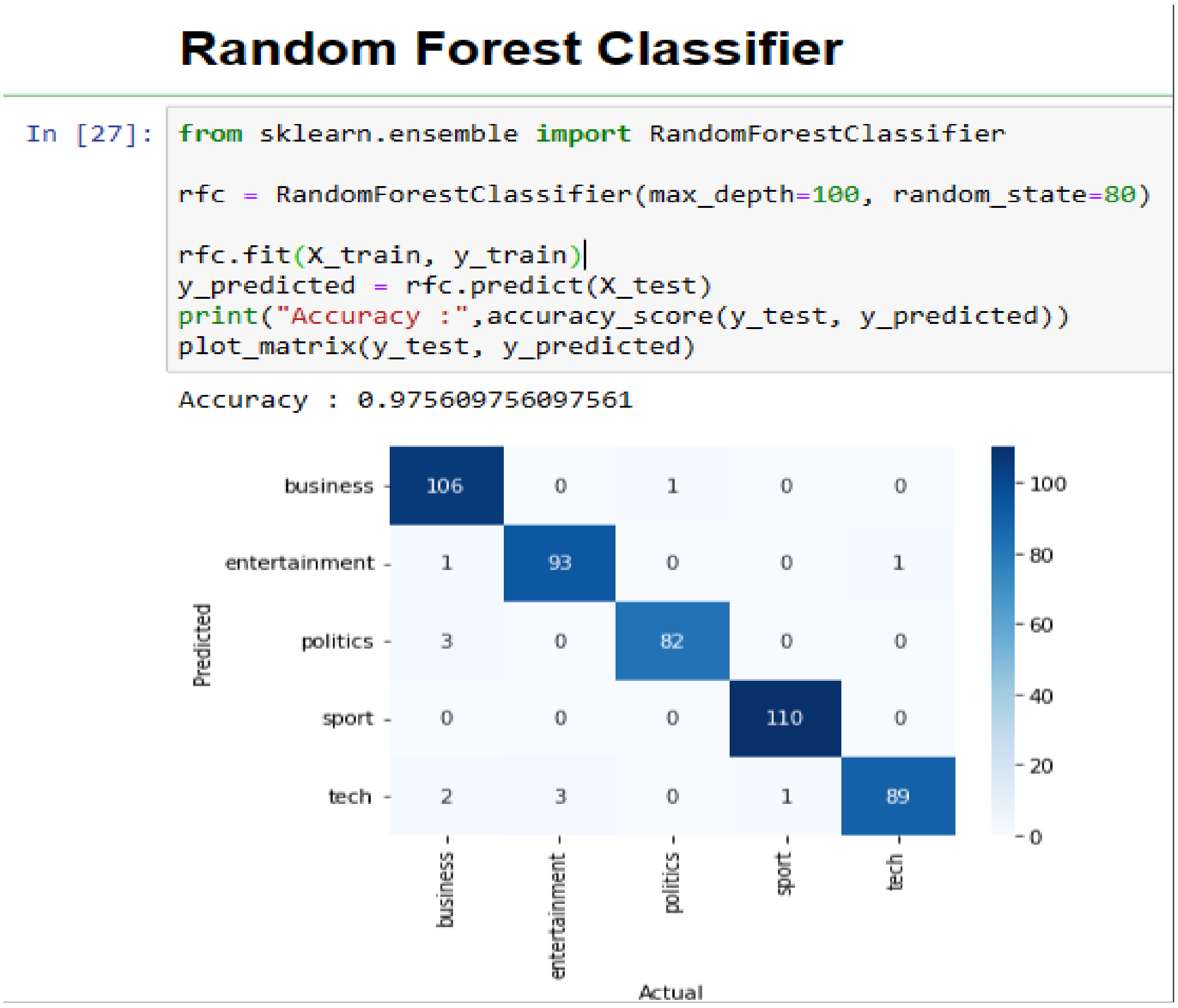
Here, a Random Forest classifier instance is being created with specified parameters.

**max_depth=100** restricts the depth of the trees in the forest to a maximum depth of 100, preventing them from growing too deep and possibly overfitting.

**random_state=80** ensures reproducibility. It controls the randomness of the bootstrap samples used when building trees, so if you run the code multiple times, the results will be the same.

The fit() method trains the Random Forest classifier using the training data (X_train and y_train).

After training, the model is used to predict the labels for the test set **X_test**. The predicted labels are stored in the **y_predicted** variable.



Random Forest Classifier

```
In [27]: from sklearn.ensemble import RandomForestClassifier

         rfc = RandomForestClassifier(max_depth=100, random_state=80)

         rfc.fit(X_train, y_train)
         y_predicted = rfc.predict(X_test)
         print("Accuracy :",accuracy_score(y_test, y_predicted))
         plot_matrix(y_test, y_predicted)

Accuracy : 0.975609756097561
```

| Predicted \ Actual | business | entertainment | politics | sport | tech |
|---|---|---|---|---|---|
| business | 106 | 0 | 1 | 0 | 0 |
| entertainment | 1 | 93 | 0 | 0 | 1 |
| politics | 3 | 0 | 82 | 0 | 0 |
| sport | 0 | 0 | 0 | 110 | 0 |
| tech | 2 | 3 | 0 | 1 | 89 |

Fig No. 4.6 Random Forest Classifier

## Gradio Interface

Gradio is an open-source Python library that allows developers to rapidly generate user interfaces (UI) for machine learning models. It's particularly useful for creating proof-of-concept interfaces without the need for extensive web development expertise.

**Features:**

Rapid Prototyping: Quickly create interfaces with just a few lines of code.

Support for Various Interfaces: It supports different types of interfaces like text, image, audio, etc.

Integration with Major ML Frameworks: Works with popular frameworks such as TensorFlow, PyTorch, and HuggingFace.

Interpretability: Provides options for visualizing model interpretations, making it easier to understand predictions.

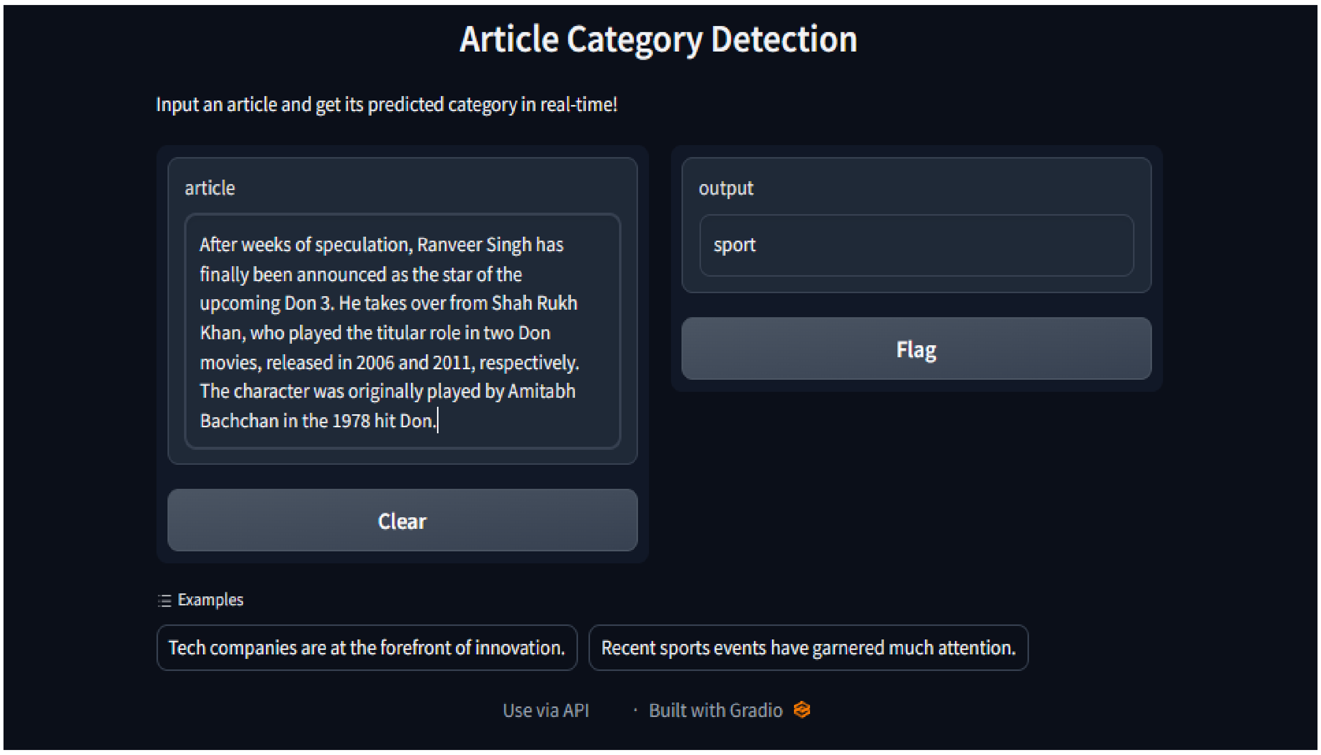
**Basic Usage:**

To create a Gradio interface, you typically follow these steps:

Define the function: This function takes in model inputs and produces outputs. For instance, for an image classifier, the function might take in an image and return predicted labels.

Choose input and output components: For instance, an "Image" input component for image classification or a "Textbox" for NLP tasks.

Launch the Interface: Call Interface() method to launch the web app.



Article Category Detection

Input an article and get its predicted category in real-time!

article: After weeks of speculation, Ranveer Singh has finally been announced as the star of the upcoming Don 3. He takes over from Shah Rukh Khan, who played the titular role in two Don movies, released in 2006 and 2011, respectively. The character was originally played by Amitabh Bachchan in the 1978 hit Don.

output: sport

Examples

Tech companies are at the forefront of innovation.

Recent sports events have garnered much attention.

Fig No. 4.7 Gradio Interface

# CHAPTER 5

# RESULTS AND ANALYSIS

In this work, a classification of news is reviewed. Pre-processing, document indexing, feature selection, and classification of news headlines are all phases that are thoroughly explored. Also mentioned is stop word filtering utilising a frequency-based stop word removal strategy. These algorithms can be evaluated on bigger corpora in the future. Additionally, these algorithms can be enhanced to increase categorization efficiency. Clustering can be done more quickly by combining several algorithms.

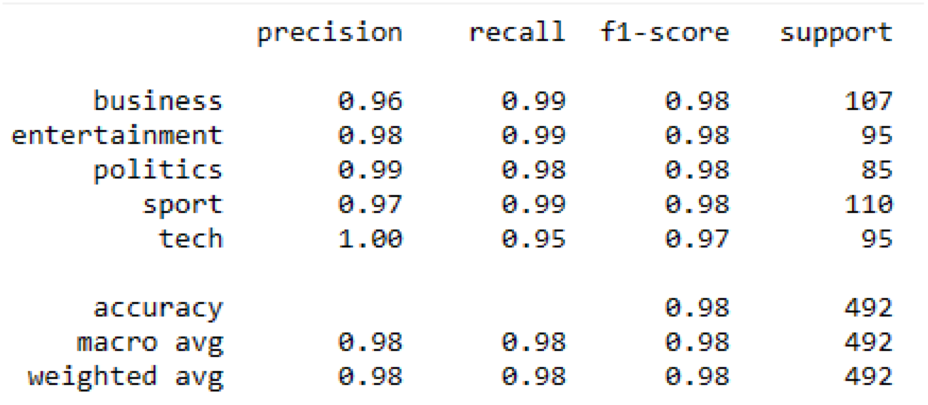
We finally obtained the accuracy scores after doing Data Cleaning and Data Preprocessing (cleaning data, train_test_split model, creation of a bag of words NLP model, and machine learning model). From these results, we can conclude that Random Forest Classification provides the highest accuracy of all machine learning models.

Finally, we forecast the genre of various news articles.

## 5.1 Model and Cross Validation of Logistic Regression

A Logistic Regression model is set up with a random_state for reproducibility and max_iter to ensure the optimization algorithm has a sufficient number of iterations to converge.

The cross_val_score function is then used to assess the model's performance using 10-fold cross-validation. The average of these scores is approximately 0.9691 or 96.91%, indicating that, on average across the ten folds, the model correctly classifies about 96.91% of the samples.


```
               precision    recall  f1-score   support

     business       0.96      0.99      0.98       107
entertainment       0.98      0.99      0.98        95
     politics       0.99      0.98      0.98        85
        sport       0.97      0.99      0.98       110
         tech       1.00      0.95      0.97        95

     accuracy                           0.98       492
    macro avg       0.98      0.98      0.98       492
 weighted avg       0.98      0.98      0.98       492
```

Fig No. 5.1 Classification Report

### 5.1.1 Classification Report

The classification report gives detailed performance metrics for each category:

Precision: The ratio of correctly predicted positive observations to the total predicted positives. High precision means that the false positive rate is low.

Recall (Sensitivity): The ratio of correctly predicted positive observations to all the actual positives.

F1-Score: The weighted average of Precision and Recall. It's useful when the class distribution is imbalanced.

Support: The number of occurrences of each class in the true response y_test.

From the provided report:

All categories (business, entertainment, politics, sport, tech) have high precision, recall, and F1-scores, typically in the high 90% range. This indicates that the model is performing well across all categories.

The tech category has the lowest recall of 0.95 or 95%, suggesting that 5% of tech articles were misclassified into other categories.

The overall accuracy is about 98%, which is quite high, suggesting the model is robust in its predictions across the dataset.

### 5.1.2 Confusion Matrix

The provided function plot_matrix visualizes the confusion matrix using seaborn.

The diagonal elements represent the number of points for which the predicted label is equal to the true label.

Off-diagonal elements are those that are misclassified by the classifier.

By examining the heatmap, you can see which classes the model is mixing up if any.

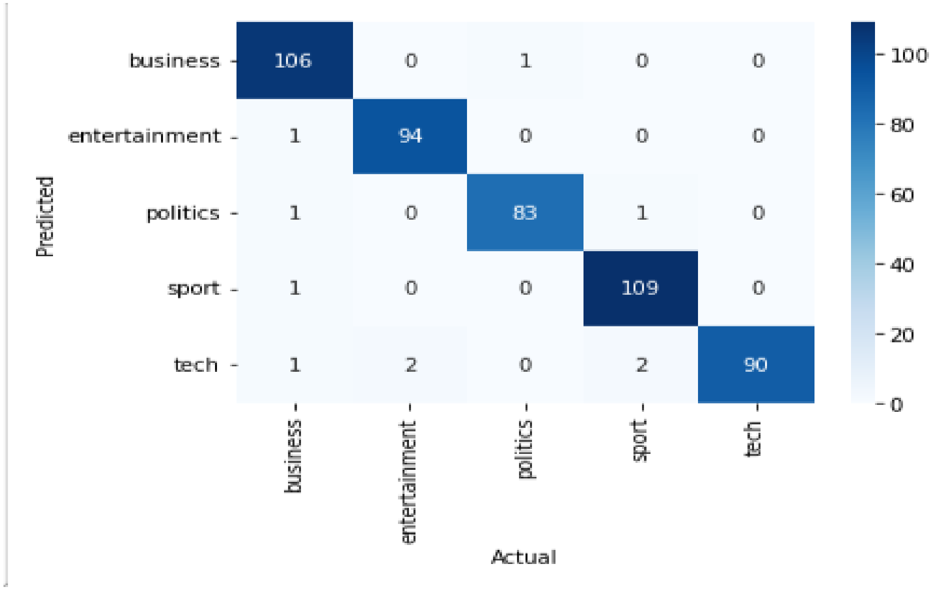


Fig No. 5.2 Confusion Matrix

The Logistic Regression model provides strong performance on this dataset, with an average accuracy of approximately 96.91% during cross-validation and about 98% on the test data.

The high scores across precision, recall, and F1-score for each category suggest that the model is both robust and reliable.

There are some misclassifications, but they are minimal given the high accuracy. Further investigation could be done to see if additional tuning or feature engineering might further reduce these misclassifications.

Overall, the model seems to be well-suited for the task of categorizing news articles into the given topics.

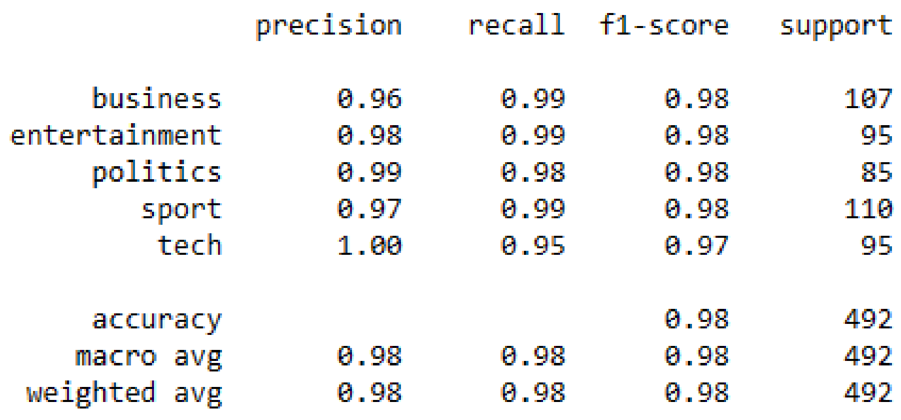
## 5.2 Classification Report of Random Forest Classifier

Precision: Of all the samples predicted in a particular category, how many were actually of that category.

Recall: Of all the actual samples of a particular category, how many were correctly predicted by the model.

F1-Score: The harmonic mean of precision and recall, a balanced measure.

Support: Number of occurrences of each category in y_test.



```
               precision    recall  f1-score   support

     business       0.96      0.99      0.98       107
entertainment       0.98      0.99      0.98        95
     politics       0.99      0.98      0.98        85
        sport       0.97      0.99      0.98       110
         tech       1.00      0.95      0.97        95

     accuracy                           0.98       492
    macro avg       0.98      0.98      0.98       492
 weighted avg       0.98      0.98      0.98       492
```

Fig No. 5.3 Classification Report of Random Forest classifier

From the provided metrics:

All the categories (business, entertainment, politics, sport, tech) have high precision, recall, and F1-scores, indicating the model performs well across all of them.

Like in the previous Logistic Regression model, the tech category still has the lowest recall, with 95%. It seems this category is slightly more challenging to predict perfectly, though a 95% recall is still quite strong.

### 5.2.1 Confusion Matrix

The plot_matrix function (assumed to have been defined as before) displays the confusion matrix. Here's what to derive from it:

Diagonal elements represent correctly classified samples. High numbers on the diagonal and low numbers elsewhere show the model's effectiveness.

Off-diagonal elements represent misclassifications. By observing which cells in the matrix have high values, you can determine which categories the model tends to confuse.

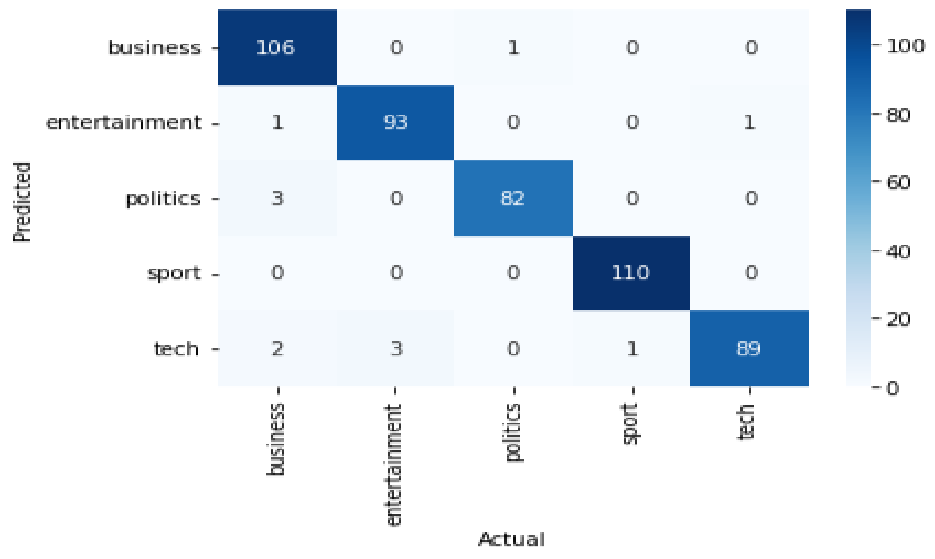


Fig No. 5.4 Confussion matrix

The Random Forest classifier performs exceptionally well on this dataset, achieving an accuracy of about 97.56% on the test set. This suggests that the features are very indicative of the article categories, and the model can capture this relationship.

While the results are quite similar to the Logistic Regression model, in a real-world scenario, you'd choose based on various factors: interpretability, training time, robustness, etc. Here, both models seem comparably effective.

There's room for improvement for the tech category, as indicated by the recall. Further analysis could provide insights: it could be due to overlapping content with other categories, or it might benefit from more labeled samples.

In summary, the Random Forest model is very well-suited for this task and offers strong predictive performance across the various news categories.

# CHAPTER 6

# CONCLUSIONS & FUTURE SCOPE

## 6.1 Conclusions

The present research offers a review of news classification. Pre-processing, document indexing, feature selection, and news headline classification are all procedures that are addressed in detail. It is also detailed how to filter out stop words using a frequency-based technique. Future testing of these algorithms can be done on larger corpora. Furthermore, these algorithms can be enhanced to increase the categorization process' effectiveness. Clustering can be accomplished more quickly by using a combination of algorithms.

The Logistic Regression model provides strong performance on this dataset, with an average accuracy of approximately 96.91% during cross-validation and about 98% on the test data.

The high scores across precision, recall, and F1-score for each category suggest that the model is both robust and reliable.

There are some misclassifications, but they are minimal given the high accuracy. Further investigation could be done to see if additional tuning or feature engineering might further reduce these misclassifications.

Overall, the model seems to be well-suited for the task of categorizing news articles into the given topics.

We finally obtained the accuracy scores after doing Data Cleaning and Data Preprocessing (cleaning data, train_test_split model, creation of a bag of words NLP model, and machine learning model). From these results, we can conclude that Random Forest Classification provides the highest accuracy of all machine learning models.

Finally, we forecast the genre of various news articles.

The Random Forest classifier performs exceptionally well on this dataset, achieving an accuracy of about 97.56% on the test set. This suggests that the features are very indicative of the article categories, and the model can capture this relationship.

While the results are quite similar to the Logistic Regression model, in a real-world scenario, you'd choose based on various factors: interpretability, training time, robustness, etc. Here, both models seem comparably effective.

There's room for improvement for the tech category, as indicated by the recall. Further analysis could provide insights: it could be due to overlapping content with other categories, or it might benefit from more labeled samples.

In summary, the Random Forest model is very well-suited for this task and offers strong predictive performance across the various news categories.

## 6.2 Future Scope of Work

Future developments in technology, customer needs, and the dynamics of news consumption will all affect how news articles are sorted. A thorough investigation of potential improvements and extensions in the field.

Use machine learning to curate personalized news feeds based on user preferences, reading history, and social signals.

Predict and suggest articles of interest, even from categories the user might not typically explore, broadening their horizons.

Use feedback from readers (likes, shares, comments) to continually refine and improve the sorting mechanism.

Similar to recommendation systems in e-commerce or movie platforms, implement collaborative filtering to sort and suggest news articles based on what similar users are reading.

## REFERENCES

1. Mazhar Iqbal Rana, Shehzad Khalid, Muhammad Usman Akbar. "News Classification Based On Their Headlines: A Review" 17th IEEE International Conference on Multi-Topic Conference (INMIC), Karachi,Pakistan,,2014,211-21

2. Rana, M. I., Khalid, S., & Akbar, M. U. (2014, December). News classification based on their headlines: A review. In *17th IEEE International Multi Topic Conference 2014* (pp. 211-216). IEEE.

3. Chan, C. H., Sun, A., & LIM, E. P. (2001). Automated online news classification with personalization.

4. Kaur, G., & Bajaj, K. (2016). News classification and its techniques: a review. *IOSR Journal of Computer Engineering*, *18*(1), 22-26.

5. CHAN, Chee-Hong; SUN, Aixin; and LIM, Ee Peng. Automated online news classification with personalization. (2001). *4th International Conference on Asian Digital Libraries*. Research Collection School Of Computing and Information Systems.