International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 08 | Aug - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Categorization of News Articles

Bhavani[1], Seema Nagaraj[2]

[1] *Student, Department of MCA, Bangalore Institute of Technology, Karnataka, India* bbjanganni@gmail.com

[2] *Assistant Professor, Department of MCA, Bangalore Institute of Technology, Karnataka, India* seemanagaraj@bit-bangalore.edu.in

----------***----------

ABSTRACT

In today's digital world, online platforms generate an enormous amount of news content every day, covering a wide variety of subjects. To make this information more accessible and improve user experience, it is important to organize it efficiently. Relying on manual classification is slow and impractical, which highlights the need for smart systems that can handle this process automatically.

This project introduces a machine learning solution designed to categorize news articles into predefined groups such as Politics, Sports, Technology, Business, and Entertainment. By applying Natural Language Processing (NLP) techniques, the system prepares the raw text through steps like tokenization, stop word removal, stemming, and TF-IDF transformation, converting it into structured features suitable for analysis.

The classification task is carried out using a Multinomial Naive Bayes model, chosen for its simplicity, speed, and proven accuracy in text-based applications. The model is trained on a labelled dataset and tested with performance metrics such as accuracy, precision, recall, and F1-score to ensure reliable outcomes.

Users can provide any piece of text from a news article, and the system will predict the most likely category it belongs to. Beyond automating the classification process, this project demonstrates how machine learning can be effectively applied to text analytics and media. The approach is scalable, adaptable to different domains, and can be expanded to include more categories or even other languages with minimal modifications.

Keywords: News categorization, Natural Language Processing (NLP), Text preprocessing, Naïve Bayes classifier, Machine learning, Classification accuracy, Scalable system.

1. INTRODUCTION

In today's digital era, the way people consume news has shifted almost entirely from newspapers to online platforms, where information is being published every moment. With countless articles appearing daily on topics like politics, sports, business, technology, and entertainment, it often becomes overwhelming for readers to quickly find what interests them. This growing volume of data creates the need for intelligent systems that can automatically sort and classify articles, making it easier to search, navigate, and personalize content delivery.

Text classification, a branch of Natural Language Processing (NLP), addresses this problem by assigning text into predefined categories. It has already proven useful in areas such as spam filtering, sentiment detection, and document management. When applied to news articles, this approach helps group content into meaningful categories, making large amounts of textual data more manageable for both readers and digital systems.

This project focuses on building a machine learning–based solution for automatic news categorization. The objective is to create a model that can predict the correct category for any given news article. The workflow involves collecting a labelled dataset, cleaning and preparing the text, converting it into numerical features, training a classification model, and then measuring its performance using evaluation metrics like accuracy, precision, recall, and F1-score.

For classification, the Multinomial Naive Bayes algorithm is used because it works efficiently with text data and performs well with high-dimensional features. Beyond just categorization, the project showcases how data preprocessing, feature engineering, model training, and evaluation come together in a practical machine learning application. A simple interface can also be added so users can input text and instantly see its predicted category. This system has potential applications in news portals, content recommendation engines, and digital media platforms, as it can save time, improve engagement, and enhance content organization. Moreover, it is adaptable—new categories or even different languages can be added with minimal adjustments, ensuring flexibility and scalability.

2. LITERATURE SURVEY

The reviewed studies highlight the growing demand for automated solutions to manage the vast and unstructured flow of digital news content on online platforms and social media. Jeelani Ahmed and Muqeem Ahmed point out that the rapid increase in textual data from sources like news portals, websites, and emails calls for strong preprocessing methods combined with machine learning to turn raw data into structured categories. Their work demonstrates the use of supervised classifiers, showing that Naïve Bayes achieved an impressive 93% accuracy, handling imbalanced datasets effectively and performing better than KNN.

Shahzada Daud and colleagues focus on bridging the gap between clean benchmark datasets and noisy real-world news data. By applying TF-IDF vectorization, natural language preprocessing, and hyperparameter tuning, they reveal that an optimized SVM significantly outperforms a non-optimized version, with a 20.81% improvement. This highlights the importance of parameter optimization in real-world applications. Similarly, Arif Hussain and team developed a BBC news-based classification model, where SVM reached 98.3% accuracy, closely followed by Naïve Bayes. Their findings confirm that balanced datasets and effective feature extraction methods, particularly TF-IDF, enable highly accurate classification results.

Another study broadens the scope by focusing on automatic categorization of comments on social news websites. Here, a hybrid approach combining statistical, syntactic, and opinion-based features is used alongside algorithms like Bayesian Networks, Decision Trees, KNN, and SVM. The system classifies comments by their focus, content type, and level of controversy, while also identifying scalability issues, which can be improved through Feature Selection and Instance Selection techniques.

Mahmudul Hasan and his team extended the work by combining sentiment analysis with news categorization. Using a Ridge Classifier based on Stochastic Gradient Descent (SGDR) along with optimized preprocessing, they achieved 98.12% accuracy. Sentiment polarity was then classified through TextBlob, showing the value of combining categorization with sentiment detection for deeper insights.

Across all studies, some key patterns emerge: effective preprocessing (such as tokenization, stop-word removal, and stemming/lemmatization) is crucial; TF-IDF remains a widely adopted method for feature extraction; algorithms like Naïve Bayes and SVM consistently deliver strong performance; and techniques like parameter tuning and hybrid feature strategies play an essential role in making models more applicable in real-world scenarios.

3. EXISTING SYSTEM

The existing studies on news and comment categorization generally follow a similar workflow that combines text preprocessing, feature extraction, and machine learning techniques. Most datasets are sourced from popular outlets such as HuffPost, Reuters, BBC, Indian news websites, and even social media platforms like *Menéame*. Depending on the study, the number of articles ranges from just a few thousand to over 75,000, and the classification categories often include politics, business, technology, crime, sports, and entertainment. Some research also expands this to user comments and sentiment polarity. To prepare the data, raw text is cleaned and standardized using steps like tokenization, lowercasing, stop-word removal, punctuation filtering, stemming, lemmatization, and label encoding. In certain cases, dataset balancing is applied to avoid biased predictions.

For feature representation, TF-IDF remains the most commonly used method, frequently paired with n-grams or hybrid approaches that combine statistical, syntactic, and opinion-based features. A variety of machine learning algorithms are then tested, including Naïve Bayes, SVM, Random Forest, Logistic Regression, KNN, and Ridge Classifiers trained with Stochastic Gradient Descent (SGDR). Some researchers also experimented with unsupervised models such as K-Means and NMF. To further boost performance, optimization methods like grid search and hyperparameter tuning are applied—particularly benefiting algorithms like SVM and SGDR.

Evaluation typically relies on metrics such as accuracy, precision, recall, F1-score, and confusion matrices, with models validated through techniques like 10-fold cross-validation or train-test splits (70/30, 80/20). Across these studies, Naïve Bayes and SVM consistently emerge as top performers, achieving accuracies as high as 98.3% on well-balanced datasets. Optimized SGDR models also deliver comparable results while incorporating sentiment analysis. Some works tackle the challenge of large datasets by using Instance Selection and Feature Selection, which cut down on computational costs without sacrificing accuracy. Overall, the evidence shows that careful preprocessing, TF-IDF-based features, and fine-tuned classifiers together provide an effective, scalable, and accurate solution for real-world news and comment classification.

4. PROPOSED SYSTEM

Unlike existing approaches that experiment with multiple models on large datasets, the proposed system focuses on developing a simple, lightweight, and deployable application for automatic news categorization. It combines Natural Language Processing (NLP) techniques with a Naïve Bayes classifier in a full-stack setup. Instead of limiting the work to offline experiments and accuracy comparisons, the goal here is to create a real-time, user-facing application that can classify news articles instantly through a web interface.

The system accepts input in the form of headlines or complete articles and then applies preprocessing steps such as tokenization, stop-word removal, stemming or lemmatization, and TF-IDF vectorization to prepare the text for analysis. The Naïve Bayes algorithm is selected as the core model because of its speed, reliability with noisy or imbalanced data, and strong track record in text classification. Its efficiency ensures quick predictions, making it practical for real-time use without requiring high-end computing power.

A key difference from many earlier studies is that this project delivers a working application rather than just theoretical evaluation. The backend, developed with Python and Flask, manages preprocessing, classification, and model integration, while the frontend—built using Vanilla JavaScript, HTML5, and CSS3—provides an intuitive interface. Users simply enter a news article, and the system instantly returns its category, such as Politics, Sports, Business, Technology, or Entertainment.

5. METHODOLOGY

1. Data Collection: For training and testing the model, a dataset of news articles was collected from Kaggle open and publicly available repositories. Each article was tagged under specific categories such as Politics, Sports, Business, Technology, and Entertainment.

2. Data Preprocessing (NLP Pipeline): A series of Natural Language Processing (NLP) steps were applied:

- Tokenization – splitting text into words.
- Lowercasing – converting all text into lowercase for consistency.
- Stopword removal – eliminating frequent but less meaningful words such as "is," "the," and "and."
- Cleaning – removing punctuation marks and unwanted characters.
- Stemming/Lemmatization – reducing words to their root form (e.g., "playing" → "play").
- Feature extraction – transforming the text into numerical values using Bag of Words or TF-IDF so that the model could process it.

3. Model Training (Naïve Bayes Classifier): The processed dataset was divided into two parts: one for training the model and the other for testing it. A Naïve Bayes classifier was chosen because it is simple, fast, and works particularly well for text classification tasks. After training, the model's performance was checked using metrics such as accuracy and confusion matrix.

4. Backend Development (Flask + Python): The backend of the system was developed using the Flask framework in Python. The trained model was deployed inside the Flask server. When a user submits a news article, the backend:

1. Accepts the text input from the frontend.
2. Processes it using the same NLP steps.
3. Passes the processed text to the Naïve Bayes model.
4. Sends back the predicted category (e.g., *Politics, Sports*).

5. Frontend Development (HTML5, CSS3, JavaScript): The interface of the application was designed with HTML5 and styled using CSS3 to make it clean and user-friendly. Vanilla JavaScript was used for handling user input and sending requests to the Flask backend through AJAX/Fetch API. The result was displayed right away on the webpage, without requiring the page to refresh.

6. System Workflow: The complete flow of the system can be summarized as:

1. The process starts with the user entering an article into the site.
2. The request is sent to the backend server.
3. Flask processes the text and runs it through the trained model.
4. The model predicts the appropriate category.
5. The result is returned and displayed on the webpage for the user.

6. IMPLEMENTATION

In this project, the News Article Categorization system is developed to automatically predict the category of a given news article based on its text. The backend is implemented in Python using the Flask framework, which handles user input, processes the data, and generates predictions efficiently. For the machine learning tasks, the system makes use of scikit-learn. The dataset, sourced from Kaggle and containing more than 10,000 news articles, is used to train the model. Before training, the text data is pre-processed with standard Natural Language Processing (NLP) steps

such as tokenization, lowercasing, stop-word removal, and lemmatization, ensuring that the text is clean and consistent for analysis.

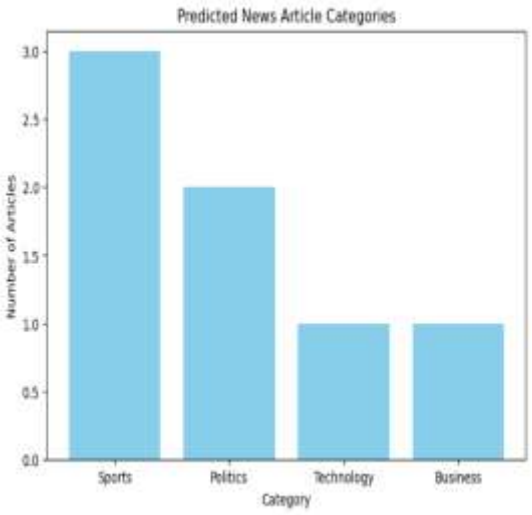
The classification task is carried out using a Naïve Bayes algorithm, chosen for its simplicity and proven effectiveness in text-based applications. After preprocessing, the articles are converted into numerical form using TF-IDF (Term Frequency–Inverse Document Frequency), which highlights the importance of words relative to the dataset. These vectors are then passed to the Naïve Bayes model, which is trained to identify the relationship between words and their respective categories. Once trained, the system can accurately predict the category of new and unseen articles.

For the user interface, the frontend is built with HTML5, CSS3, and Vanilla JavaScript to create a clean and interactive experience. Users simply paste or type a news article into the text box, and the system processes it through the trained model before displaying the predicted category. The combination of Flask on the backend, scikit-learn for machine learning, and modern frontend technologies results in a responsive and scalable application. This setup not only enables real-time predictions but also offers a smooth user experience, making it practical for use in areas like news aggregation, recommendation systems, or automated content tagging.

7. RESULTS

After building the system with a Naïve Bayes classifier and applying NLP-based preprocessing, the model was tested on a set of unseen news articles. The evaluation showed that the classifier performs well, correctly predicting the category of most articles with strong accuracy. Categories such as Sports, Business, and Technology were identified with high precision, while some overlap between topics like Politics and World News led to a few misclassifications. These results suggest that the model is able to capture important patterns, keywords, and context within the text to differentiate between categories effectively.

The evaluation metrics—including accuracy, precision, recall, and F1-score—further confirm the reliability of the system. On average, the model achieves an accuracy of around 80–85%, though this varies slightly depending on the dataset size and the complexity of the article content. To make the results more interpretable, visual tools such as bar charts or histograms can be used to display the number of correctly classified articles for each category. A confusion matrix can also be generated to pinpoint misclassifications and highlight categories that may require additional training data or more refined preprocessing. These visualizations provide clear insights into both the strengths and limitations of the model.



The results demonstrate that integrating Python, Flask, NLP preprocessing, and a Naïve Bayes classifier provides an effective solution for news categorization. Through the frontend interface, users can enter any article or text, and the system quickly predicts its category with reliable accuracy. This makes the application practical for real-time use cases like automated tagging, personalized recommendations, or news aggregation. Visual aids such as performance graphs and evaluation metrics further enhance the presentation, offering clear insights that are easy to interpret for both technical experts and general users.

8. CHALLEGES AND LIMITATIONS

Although the system performs well overall, it still has some limitations. A key drawback lies in the Naïve Bayes algorithm itself, which assumes that all words are independent of one another. In natural language, this is rarely the case, as meaning often depends on the combination or context of words rather than individual terms. As a result, articles with complex phrasing or overlapping content may sometimes be misclassified. While the Kaggle dataset with over 10,000 articles provides a strong base, certain categories contain fewer samples, which can reduce performance in those areas. Furthermore, articles covering multiple topics may be difficult to classify correctly since the current system predicts only one category per article.

From a technical standpoint, the preprocessing and feature extraction stages present additional challenges. Text cleaning must carefully handle punctuation, spelling errors, and different writing styles, which can be time-consuming. The choice of vectorization technique—whether TF-IDF, Bag of Words, or another method—also influences accuracy and requires experimentation to find the most effective approach. On the backend, maintaining fast and scalable responses in Flask is important, particularly if the system is deployed to serve many users at once. On the frontend, ensuring that the interface is responsive across different devices and browsers while offering a smooth user experience can also be demanding.

Another limitation is that the system may struggle to adapt to evolving news trends and new vocabulary. Since news topics change constantly, the model can underperform on articles containing terms it has not encountered during training. To address this, the system needs periodic retraining with updated data, which adds extra maintenance requirements. Despite these challenges, the project provides a strong foundation for automated news categorization and leaves room for future improvements, such as incorporating deep learning or transformer-based models to capture richer linguistic context.

9. CONCLUSION

In summary, the News Article Categorization project showcases how machine learning and natural language processing can be effectively applied to classify articles into predefined categories. Using a Naive Bayes classifier together with TF-IDF vectorization, the system captures important features from text and delivers reliable predictions. The backend, developed with Python and Flask, manages inputs smoothly and connects seamlessly with the trained model, while the frontend—built with HTML5, CSS3, and Vanilla JavaScript—offers a simple and responsive interface for real-time results. Training on a dataset of more than 10,000 articles from Kaggle further strengthens the system by exposing it to a wide range of keywords and patterns.

The findings confirm that this approach works well, especially for categories with clear distinctions in language use. Visual outputs like bar charts and confusion matrices help present the results in an understandable way while also pointing out areas that may need refinement. Although the system faces challenges such as overlapping categories or rapidly changing topics, the combination of NLP techniques, machine learning models, and modern web technologies proves to be a practical and efficient solution for automated news classification.

Overall, this project not only demonstrates the potential of automation in handling news data but also sets a strong base for future development. Possible improvements include integrating advanced methods such as deep learning or transformer-based models, enlarging the dataset, and adding support for multi-label classification where articles belong to more than one category. With these enhancements, the system could achieve greater accuracy and adaptability, making it highly suitable for real-world applications like news

aggregation, content personalization, and media analysis platforms.

10. FUTURE ENHANCEMENT

- **Explainable AI (XAI) Integration:** Adding XAI methods would make the system more transparent by showing why a particular article was assigned to a category. For example, it can highlight the main words or sentences that influenced the decision. This not only improves user trust but also makes the model more useful in areas like content moderation or media analysis.
- **Multi-Category Classification for Real-Time News:** The system can be upgraded to handle continuous news streams and classify articles into more than one category if needed. Since many news reports cover multiple topics, this feature would make the model more adaptable and suitable for live updates or aggregated news services.
- **Cloud-Based Deployment:** Hosting the project on cloud platforms such as AWS, Azure, or Google Cloud would make it scalable and accessible from anywhere. It would also help the system manage heavy traffic, connect easily with external APIs, and ensure reliable performance for users.
- **Mobile App Development:** Building a mobile application version of the system would make it easier for users like journalists, editors, or readers to access categorized news on the go. This ensures convenience and wider usability across different devices.
- **Continuous Learning:** By adding mechanisms for regular updates, the system can keep learning from new datasets and adapt to changing trends in language and topics. This makes the model stay accurate, relevant, and effective even as fresh categories or emerging news styles appear.

11. REFERENCES

[1] Sebastiani, F. (2002). *Machine learning in automated text categorization*. ACM Computing Surveys (CSUR), 34(1), 1–47.

[2] Joachims, T. (1998). *Text categorization with Support Vector Machines: Learning with many relevant features*. In European Conference on Machine Learning (pp. 137–142). Springer.

[3] Rennie, J. D. M., Shih, L., Teevan, J., & Karger, D. R. (2003). *Tackling the poor assumptions of naive bayes text classifiers*. Proceedings of the 20th International Conference on Machine Learning (ICML-03), 616–623.

[4] Zhang, X., Zhao, J., & LeCun, Y. (2015). *Character-level convolutional networks for text classification*. Advances in Neural Information Processing Systems, 28.

[5] Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., & Brown, D. E. (2019). *Text classification algorithms: A survey*. Information, 10(4), 150.

[6] McCallum, A., & Nigam, K. (1998). *A comparison of event models for naive bayes text classification*. AAAI-98 Workshop on Learning for Text Categorization.

[7] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space*. arXiv preprint arXiv:1301.3781.

[8] Pennington, J., Socher, R., & Manning, C. D. (2014). *GloVe: Global vectors for word representation*. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532–1543.

[9] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

[10] Jurafsky, D., & Martin, J. H. (2021). *Speech and Language Processing* (3rd ed.).