International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

An Evaluation Methodology News Headline Classification using Comprehensive Pattern Learning

Sivapriya G[1], Princy Reshma R[2], Deepak S[3], Dr. C. Shanmuganathan[4]

[1,2,3]UG Student, Department of Computer Science and Engineering, SRM Institute of Science and Technology, India - 600089
[4]M.E., Ph.D., Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, India - 600089

Abstract
Text classification plays a pivotal role in numerous applications, ranging from spam detection to sentiment analysis. With the exponential growth of textual data stemming from social media interactions and news articles, the demand for accurate text classification methods has surged. This research delves into the evaluation of a novel weighting function designed to adjust the representation of texts during multilabel classification [1]. This innovative approach combines two distinct strategies: problem transformation and model adaptation. By incorporating this weighting function, the classification process undergoes a significant enhancement, ultimately improving the accuracy and efficiency of multilabel classification tasks [1]. To assess the efficacy of the proposed methodology, rigorous testing was conducted across ten referential textual datasets. Comparative analysis was performed against alternative techniques, with a keen focus on three key performance measures. The results obtained from these evaluations provide valuable insights into the effectiveness and robustness of the weighting function, shedding light on its potential to revolutionize text classification paradigms.

Keywords: News Headline Classification; Comprehensive Pattern Learning; Text Classification; Evaluation Methodology; Text Mining; Neural Network; Deep Learning; Natural Language Processing; Natural Language Toolkit.

1. Introduction

Businesses and organizations rely heavily on timely and relevant information to make informed decisions. With the rapid pace of modern communication, staying updated on new events and trends is essential. Media monitoring solutions play a critical role in this regard, providing real-time analysis and alerts for ongoing events [2]. While many existing solutions focus on entities and topics, there is a growing need for event-centric solutions. These solutions not only identify ongoing events but also analyze their characteristics, enabling event-based reasoning. By understanding the causes and consequences of events, businesses can identify potential risks and opportunities, allowing them to make proactive decisions to stay ahead in their respective markets.

In today's era of information explosion, manually processing and classifying large volumes of text data is a daunting task. Moreover, the performance of manual text classification is influenced by human factors

such as experience and fatigue, leading to inconsistent results. To address these challenges, machine learning methods are indispensable for automating text classification processes and obtaining more reliable results [3]. Automated text classification not only improves efficiency in information retrieval but also helps alleviate the problem of information overload by efficiently locating the required information amidst vast amounts of data.

The internet has become a treasure trove of valuable information, with an exponential increase in unstructured text data being generated every day. Text classification (TC) has emerged as a crucial field in text mining, aiming to organize and process this vast amount of textual data effectively. TC involves categorizing new written content into predefined conceptual groups, making it essential for various applications such as sentiment analysis, spam email filtering, news text classification, and more. Building TC systems involves several key stages, including preprocessing, document modeling, and final document classification and evaluation [4]. By leveraging machine learning algorithms, TC systems can extract valuable insights from unstructured text data, making the process of knowledge extraction effortless and efficient.

Classification, the process of assigning categories to data according to their content, plays a crucial role in data analysis and management. While various classification methods have been proposed, text classification stands out as a fundamental task in natural language processing (NLP). Text classification enables the evaluation and organization of textual data, making it easier to access and understand. It is especially important for categorizing online content, such as topic detection in news articles and spam detection in emails [4]. Although various text classification methods exist, such as the extended stochastic complexity (ESC) stochastic decision list, these methods often require deep domain knowledge and may not be suitable for analyzing unstructured text data without the use of machine learning approaches.

Machine learning-based systems offer three main types: supervised learning, unsupervised learning, and semi-supervised learning. In supervised text classification tasks, models are trained on labeled data, while unsupervised learning involves working with unlabeled data. Semi-supervised learning, a combination of supervised and unsupervised learning methods, has gained attention in recent years [5]. Positive and Unlabeled (PU) learning, a branch of semi-supervised learning, trains a two-category classifier in scenarios with only positive samples and unlabeled samples. The goal of PU learning is similar to that of traditional binary classification: to train a classifier and distinguish between positive and negative samples based on their features.

2. Existing System

With the advancement of Internet technology, online platforms have become primary sources for people to access breaking news. Filtering out the latest hot news from a vast collection of articles and delivering them to users holds significant application value. However, in supervised learning scenarios, manually labeling each news article is time-consuming and labor-intensive. From the perspective of semi-supervised learning, this paper introduces a novel algorithm named Enhanced nnPU with Focal Loss (FLPU) for news headline classification. This algorithm is based on non-negative Positive-Unlabeled (nnPU) learning and utilizes Focal Loss to calculate the empirical risk of positive and negative samples, replacing the classical nnPU method [7].

Furthermore, by incorporating Virtual Adversarial Training (VAT) from Adversarial training for large neural LangUage Models (ALUM) into FLPU, an improved algorithm called FLPU+ALUM is proposed for the same purpose. This algorithm aims to label only a small number of positive samples. Through

experiments conducted on two datasets, the superiority of both algorithms over state-of-the-art PU algorithms is demonstrated [4]. Additionally, another set of experiments shows that when enriched by these algorithms, the RoBERTa-wwm-ext model achieves better classification performance than existing state-of-the-art binary classification models included in the comparison [6].

This paper introduces the use of non-negative Positive-Unlabeled (nnPU) learning in news headline classification tasks. This approach allows for better classification performance, especially in cases where only positive samples and unlabeled samples are available. The paper proposes using Focal Loss as an optimization technique, replacing the original nnPU calculation of the empirical risk of positive and negative samples. This results in the creation of the FLPU algorithm, which enriches existing binary classifiers used for news headline classification [8]. Additionally, by incorporating Virtual Adversarial Training (VAT) from Adversarial training for large neural LangUage Models (ALUM) into FLPU, an improved algorithm called FLPU+ALUM is introduced, aiming to label only a small number of positive samples.

3. Proposed System

The goal of feature selection is to prepare cleaner and more understandable data, leading to the development of simpler and more interpretable models, thus improving the performance of data mining and machine learning [6]. Many studies have extended basic feature selection techniques by incorporating external knowledge or optimizing feature representation in natural language processing (NLP) tasks.

In the case of document classification, the representation of text significantly impacts the classification task [7]. The vector space model (VSM) is one of the most commonly used models for information retrieval due to its conceptual simplicity and the effectiveness of its underlying metaphor, which uses spatial proximity to represent semantic proximity.

In the vector space model (VSM), the content of a document is represented by a vector of terms d=w1,...,wkd=w1,...,wk, where kk is the size of the set of terms (or features). Various elements can be used to represent text, including N-grams, words, phrases, logical terms, declarations, or any other lexical, semantic, and/or syntactic unit that represents the content of the text.

The text content is processed as follows:

- **Input Layer**: Expanded text content is input into the input layer.
- **Vector Representation Layer**: All text content is represented as a zero-one binary distributed vector.
- **Convolution Layer**: The distributed vector is input into the convolution layer to extract characteristic features.
- **Pooling Layer**: The character feature vector is input into the pooling layer, and the maximum pooling method is applied to perform the pooling operation for all feature maps.
- **Fully Connected Layer**: The processed feature vectors are fed into a fully connected layer to map them to the sample label space.
- **Normalization Layer**: The results of the fully connected layer are input into the normalization layer, and the SoftMax function is used to output the normalized categories and their corresponding probability values.

$$P_{(tk)} = \frac{N\, e^{t}\, k}{\Sigma\, (e^{t}\, k)} \qquad (i) \longrightarrow$$

$$k = 1$$

where $P_{(tk)}$ represents the probability value corresponding to the k-th category, tk represents the k-th

category and N refers to the total number of categories. After the SoftMax normalization calculation step, the category corresponding to the maximum probability value can be output as the final text classification result.

Some symbols and settings are explained as follows:

- **Epoch**: One epoch refers to training once with all the samples in the training set.
- **Batch size**: The number of samples grabbed for one training iteration.
- **Learning rate**: The rate at which the model learns the parameters.
- **Model optimization**: Adam optimizer.

  In order to minimize the interference caused by randomness, five-fold cross-validation is used for model training and performance evaluation.

  The contributions of the research work are summarized as follows:
- **Text Classification Method**: We propose a novel text classification method that addresses the limitations of conventional methods by considering relationships between documents. This approach enhances the model's ability to understand the context and semantic meaning of documents.
- **Key Feature-Enhanced News Text Classification**: We introduce a key feature-enhanced news text classification method that leverages data augmentation techniques based on a portion of the original labeled data. By constructing a high-quality labeled sample set, this method effectively addresses the challenges associated with large-scale news text classification.
- **Feature Engineering Technology**: We enhance the overall classification performance and interpretability of the model through advanced feature engineering techniques. These techniques include key feature extraction and semantic information expansion, which enable the model to capture more nuanced patterns and relationships within the data.
- **Comprehensive Analysis**: Through a detailed comparison of experimental results, we provide a comprehensive analysis to demonstrate the efficiency and effectiveness of the proposed model. Our experiments showcase the model's superior performance compared to existing methods, highlighting its potential for real-world applications in text.

4. Methodology

The following figure 4.1 depicts the architecture diagram.

The methodology employs Comprehensive Pattern Learning (CPL) for news headline classification.It begins with inputting a dataset of news headlines. CPL facilitates feature extraction and model training.

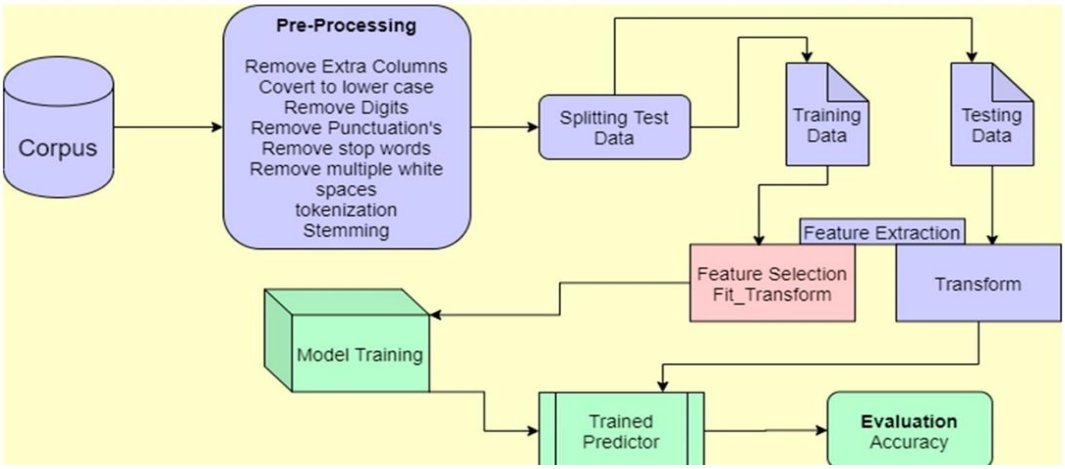

Figure 4.1 Architecture Diagram

The evaluation methodology for news headline classification employing Comprehensive Pattern Learning (CPL) encompasses several key components. Initially, a dataset of news headlines serves as input. Through CPL, the methodology undertakes feature extraction, wherein relevant patterns are discerned from the headlines. Subsequently, a CPL model is trained on these extracted features. Finally, the methodology evaluates the performance of the model, utilizing classification metrics to gauge its accuracy, precision, recall, and F1-scoree

5. Module Description

The entire process is divided into four major modules.

5.1. Module 1: News Preprocessing

In this step, several steps were performed to structure the news and remove unwanted terms that behave as noise in training the models. All characters were transformed into lowercase letters to keep the semantics independent of letter cases in the model.

In the next preprocessing stage, all the special characters and numeric values were removed from the text as they do not contribute to the classification of news; rather, they serve as noise in the data. Special characters include punctuation marks, and other characters such as $, #, @, and others. Removing special and numeric entities leaves unwanted blank spaces. These extra whitespaces were also removed and replaced with a single white space. Subsequently, stop words were also removed using a predefined list of stop words in Python. Then the news was tokenized and stemmed. Tokenization is the process where each document is divided into separate words while stemming is the process of transforming a word into its root form. As for a human being, it is easy to understand that two words go and go have the same meaning and are used in different contexts, but this is not the case with the ML algorithm. With the help of stemming, the model can get to the root. For stemming, the Porter-Stemmer algorithm was used.

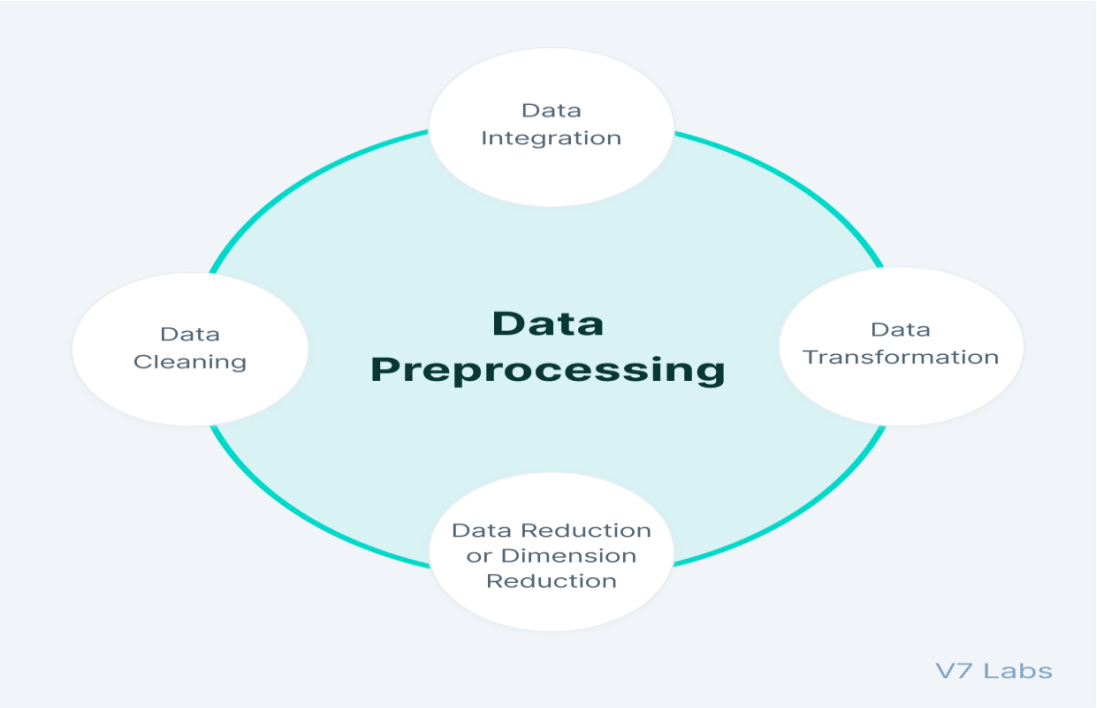

Figure 5.1.1 Data Pre-processing

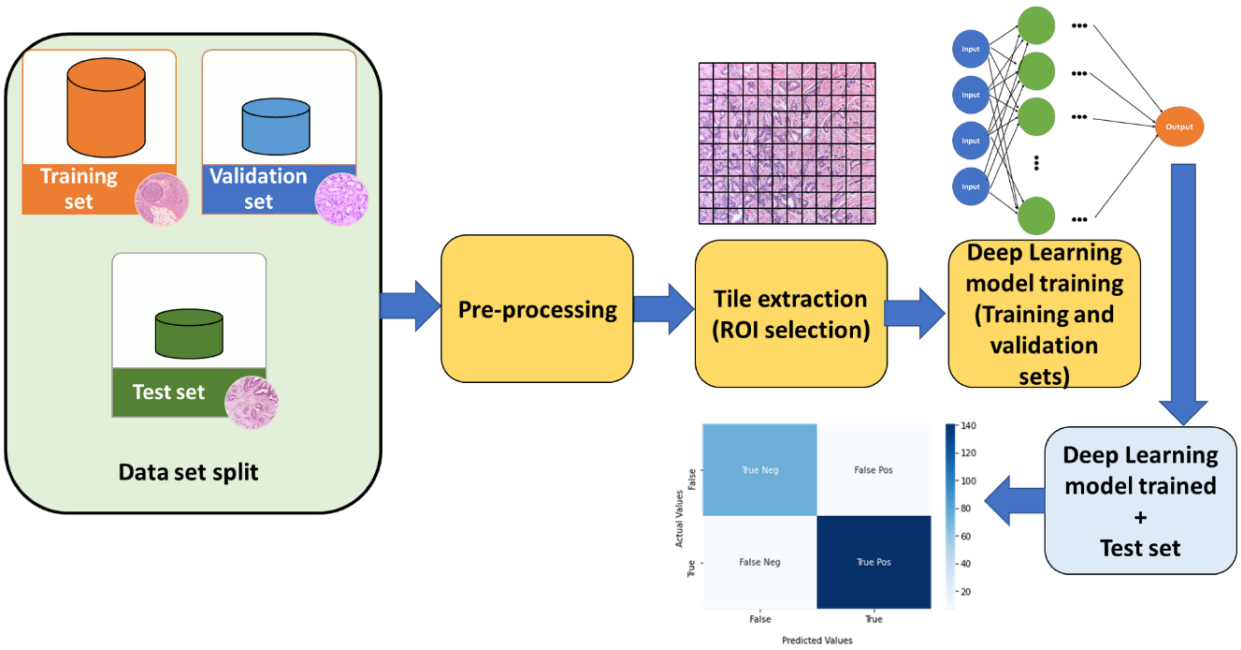

Figure 5.1.2. Pre-processing methods

5.2. Module 2: Feature Extraction

In this step, document features were converted into vector form because machine learning algorithms work with numerical data, not textual data. As these models cannot understand text features in their raw form, features need to be converted into vectors. To achieve this, the TF-IDF technique was utilized. TF-IDF is an efficient and reliable technique for categorizing news articles, where word order is not crucial. Therefore, more complex techniques such as Word2Vec and Doc2Vec were not considered.

TF-IDF utilizes two different terms: TF (Term Frequency) and IDF (Inverse Document Frequency). These two terms define two different types of frequencies for each feature, and when combined, they form each feature.

TF calculates the frequency of a word in a document by determining how many times a single term occurs in a single document relative to all the terms present in that document. Mathematically, if tftf is the word frequency in the given document and NN is the total number of documents in the corpus, TF is calculated as follows:

TF=Number of times term appears in a documentTotal number of terms in the documentTF=Total numb er of terms in the documentNumber of times term appears in a document

IDF, on the other hand, is the document frequency of a term, indicating how many documents contain a particular term. IDF is independent of the frequency of a term in the entire corpus.

$$idf\,(t, D) = \log \frac{|D|}{|d_i - t_i|} \qquad \text{(ii)} \longrightarrow$$

where $|D|$ is the number of documents in the corpus; and $|d_i - t_i|$ is the number of documents in which the term t_i is present.

The impact of IDF becomes apparent as the frequency of a term increases in the corpus, the IDF value also increases. When both TF and IDF are combined, they produce a feature that is not biased toward the frequency of a term. By considering both the document and corpus frequency, the resulting feature is an averaged vector that does not favor terms with higher frequencies in a document. This characteristic of TF-IDF has enabled researchers to achieve higher accuracies in solving problems where the word order

does not affect the results.

TF-IDF facilitates the easy identification of words that are more common across a group of documents. Words that appear more frequently in a group of documents will have higher scores. Consequently, common words can be filtered out, and rarer words can also be filtered by using a threshold for a lower score. This way, rarer words can be deemed less important for a classification model as they will have a lower weight.The mathematical operation of TF-IDF can be understood through the following equation, where D represents the corpus of documents and d is the document for which the score is being calculated. WS is the TF-IDF score for a word w in a document d. This study calculated a score for an individual word for every document using the following equation:

$$w_d = f_{w*d} * \log\left(\frac{|D|}{f_{w,d}}\right) \quad \text{(iii)} \longrightarrow$$

Module 3: Model Training

In this phase, both articles have been transformed into feature vectors, and their respective categories (class labels) have been inputted into various ML models, including hyperparameter-optimized SVM. Subsequently, these models underwent training using the provided training data. This process was iterated multiple times to refine the models by adjusting their tuning parameters, a process known as hyperparameter tuning.

Each model possesses a distinct set of parameters that can be adjusted to enhance its efficiency when trained with the given features. SVM, in particular, encompasses several parameters, with C, kernel, and decision_function_shape being significant ones. Among these, the kernel parameter notably influences the model's performance. SVM offers four different types of kernels: linear, RBF, poly, and sigmoid. The kernel function dictates the formation of the hyperplane in SVM, crucial for class distinction.

The default values for these parameters are as follows: C = 1, kernel = rbf, and decision_function_shape = ovr. The C parameter serves as a regularization function aimed at minimizing the error rate during both the training and testing phases of the model. Meanwhile, the SVM kernel determines how the hyperplane is positioned to effectively differentiate between classes.

5.3.Module 4: Evaluation Metrics

During the experimental phase, to thoroughly assess the model's overall performance, we employed not only the widely used Precision, Recall, and F1-score metrics but also Accuracy indicators. The model was optimized using cross-entropy as the loss value to enhance the network's performance. The evaluation was based on a confusion matrix, depicted in the figure below.

The calculation steps of the cross entropy for the entire model of the loss function are shown in below Formula

$$Accuracy = \frac{1}{N}\sum_{i=1}^{m}(TP_i + TN_i) \quad \text{(iv)} \longrightarrow$$

$$WP = \frac{1}{N}\sum_{i=1}^{m} n_i\left(\frac{TP_i}{TP_i+FP_i}\right) \quad \text{(v)} \longrightarrow$$

$$WR = \frac{1}{N}\sum_{i=1}^{m} n_i\left(\frac{TP_i}{TP_i+FN_i}\right) \quad \text{(vi)} \longrightarrow$$

$$WF1 = \frac{1}{N}\sum_{i=1}^{m} n_i\left(\frac{2P_iR_i}{P_i+R_i}\right) \quad \text{(vii)}$$

$$Loss = CE(\bar{y}, y) = -\sum_{k=0}^{N-1} yk \log(\bar{y}_k) \quad \text{(viii)}$$

Implementation

In an era where distinguishing between truth and falsehood in daily news has become increasingly challenging, and with a growing reliance on social media as a primary source of information, the ability to discern between genuine and fake news is more crucial than ever. To address this need, we've developed a robust model employing data analysis, machine learning, and Neural Network technology to evaluate news stories. Our user-friendly frontend, powered by Streamlit, enables users to effortlessly assess articles. By simply inputting the text and clicking submit, users receive an immediate verdict on the authenticity of the news. Our analysis is supported by a comprehensive dataset sourced from Kaggle, comprising separate CSV files containing verified and false news articles. This dataset, which includes essential information such as title, text, subject, and date of publication for each article, provides a substantial foundation for training our machine learning model. Our toolkit includes Python libraries such as pandas, numpy, and scikit-learn, as well as Streamlit for the frontend, and Tableau for advanced data visualization and analysis. We utilize applications like Jupyter Notebook, Google Colab, and VS Code for coding and development purposes.

Exploratory Data Analysis (EDA)

The EDA process involved examining and summarizing the main characteristics of the dataset, often using visual methods. It provided a better understanding of the data's distribution and uncovered patterns, anomalies, and relationships between variables.

The EDA was conducted in two distinct Jupyter notebooks: DataExploration.ipynb and SentimentAnalysis.ipynb. DataExploration.ipynb focused on initial data exploration, including data cleaning and standardization, while SentimentAnalysis.ipynb delved deeper into the emotional tone of the articles through sentiment analysis, utilizing natural language processing techniques to assess the polarity and subjectivity of the text.

Data Exploration (DataExploration.ipynb)

Utilizing Google Colab to separately inspect the fake and true CSV files, the initial phase of the process involved data cleaning, which included segregating the publisher information into a distinct column to enhance the clarity of the text column. After a thorough review of the publisher data, it became apparent that the predominant source of information was Reuters outlets in the United States, with some contributions from global sources. Further data cleansing steps were undertaken, including the removal of duplicate articles, headlines, and date information, as the latter was considered irrelevant for the analysis.

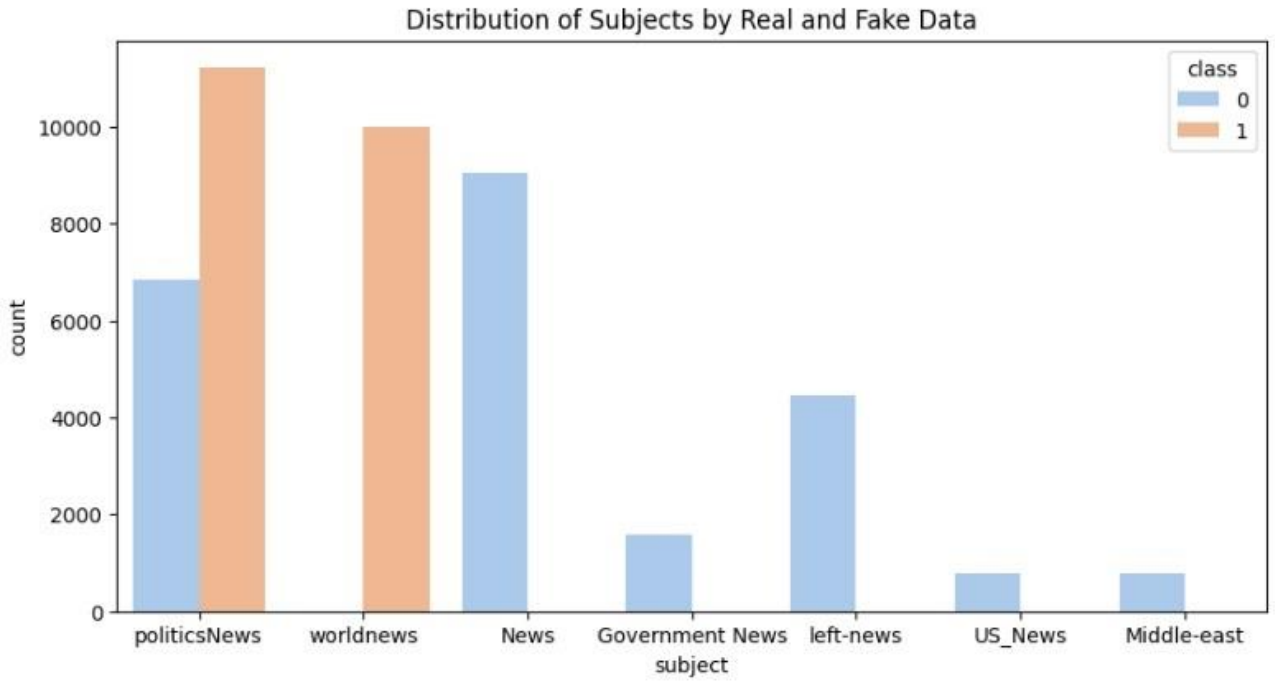

Figure 1.2 A bar graph representing distribution of subjects by real and fake data

Following these data preparation activities, the cleaned datasets were merged and produced multiple visualisations to generate insights. These visualisations encompassed:

- **comparisons of real vs. fake news article counts**

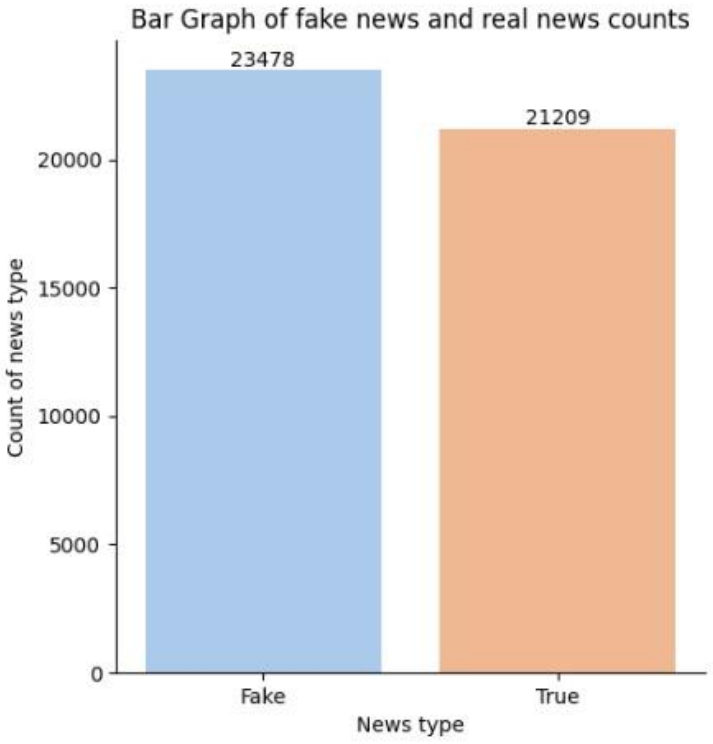

Figure 1.3 A bar graph representing fake and real news counts

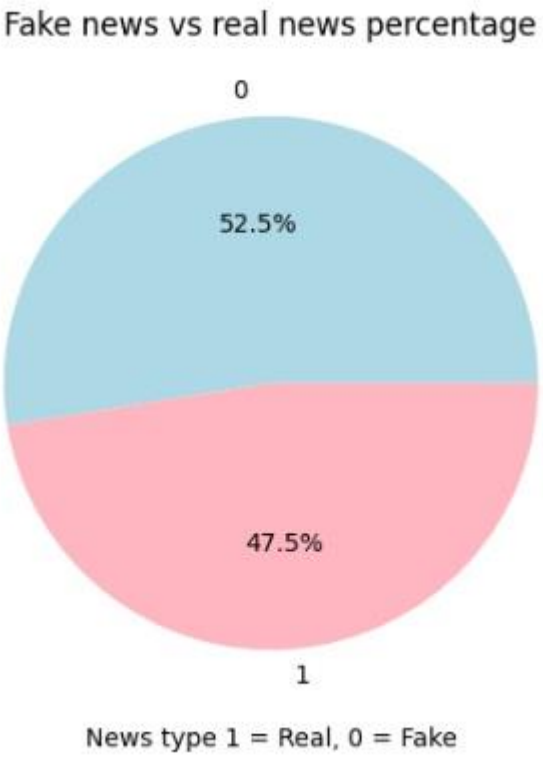

Figure 1.4 A pie chart illustrating the percentage distribution by subject

Naive Bayes

```
# fitting the training dataset on the NB classifier
Naive = naive_bayes.MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)
```

Result

	precision	recall	f1-score	support
Fake	0.59	0.89	0.71	5819
Real	0.74	0.33	0.46	5351
accuracy			0.62	11170
macro avg	0.66	0.61	0.58	11170
weighted avg	0.66	0.62	0.59	11170

Figure 1.5 Code and Result for Naive Bayes classifier

SVM

```
# fitting the training dataset on the classifier
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
```

Result

	precision	recall	f1-score	support
Fake	0.65	0.97	0.78	5819
Real	0.93	0.43	0.59	5351
accuracy			0.71	11170
macro avg	0.79	0.70	0.68	11170
weighted avg	0.79	0.71	0.69	11170

Figure 1.6 Code and Result for SVM classifier

SVM Model was observed to have a better accuracy, recall and precision than Naive Bayes Classification model.

6. Conclusion

Text classification has long been a crucial task in natural language processing. While visualization methods greatly contribute to model interpretability, there is currently limited research on visual interpretability, particularly concerning neural network models based on word embedding. This study aims to delve into this gap by employing backtracking analysis to thoroughly analyze deep learning models [8]. Additionally, it seeks to demonstrate the interpretability of these models through visualization across various dimensions.

Our research delved deeply into text classification using machine learning, specifically focusing on news categorization. We aimed to understand the significance of machine learning models in this context, particularly when optimized through hyperparameter tuning [10]. This involved optimizing and comparing various popular classifiers within text classification. Through the creation and fine-tuning of our model, we successfully achieved the highest accuracy in classifying news articles, meeting the objectives of our study.

7. Future Enhancement

In forthcoming studies, we aim to enhance the suggested model for compatibility with additional languages. Moreover, we plan to conduct further experiments to examine its suitability for various types of text. In addition, it is also possible to introduce an artificial intelligence content model similar to ChatGPT to automatically generate relevant content based on key features to achieve feature enhancement and improve the overall classification performance of the model [11].

Reference

1. Zhanlin Ji, Chengyuan Du, Jiawen Jiang, Li Zhao, Haiyang Zhang, Ivan Ganchev Improving Non-Negative Positive- Unlabeled Learning for News Headline Classification IEEE Access, 2023
2. Praboda Rajapaksha,Reza Farahbakhsh,Noel Crespi BERT, XLNet or RoBERTa: The Best Transfer Learning Model to Detect Clickbaits IEEE Access, 2021
3. Shuyin Li,Yang Liu News Video Title Extraction Algorithm Based on Deep Learning IEEE Access, 2021
4. Muhammad Umer,Zainab Imtiaz,Saleem Ullah,Arif Mehmood,Gyu Sang Choi,Byung-Won On Fake News Stance Detection Using Deep Learning Architecture (CNN-LSTM) IEEE Access, 2020

5. Piyush Ghasiya,Koji Okamura Investigating COVID-19 News Across Four Nations: A Topic Modeling and Sentiment Analysis Approach IEEE Access, 2021
6. Yi Zhu,Yun Li,Yongzheng Yue,Jipeng Qiang,Yunhao Yuan A Hybrid Classification Method via Character Embedding in Chinese Short Text With Few Words IEEE Access, 2020
7. Xinjie Sun,Xingying Huo Word-Level and Pinyin-Level Based Chinese Short Text Classification IEEE Access, 2022
8. Krzysztof Fiok, Waldemar Karwowski, Edgar Gutierrez-Franco, Mohammad Reza Davahli, Maciej Wilamowski, Tareq Ahram, Awad Al-Juaid, Jozef Zurada Text Guide: Improving the Quality of Long Text Classification by a Text Selection Method Based on Feature Importance IEEE Access, 2021
9. Muhammad Pervez Akhter, Zheng Jiangbin, Irfan Raza Naqvi, Mohammed Abdelmajeed, Atif Mehmood, Muhammad Tariq Sadiq Document-Level Text Classification Using Single-Layer Multisize Filters Convolutional Neural Network IEEE Access, 2020
10. Peng Wang, Yun Yan, Yingdong Si, Gancheng Zhu, Xiangping Zhan, Jun Wang, Runsheng Pan Classification of Proactive Personality: Text Mining Based on Weibo Text and Short-Answer Questions Text IEEE Access, 2020
11. Yanrong Huang, Rui Wang, Bin Huang, Bo Wei, Shu Li Zheng, Min Chen Sentiment Classification of Crowdsourcing Participantsâ€™ Reviews Text Based on LDA Topic Model IEEE Access, 2021