News Article Classification Using Machine Learning Models

Fatima Mohsin Khan, Khadija Imran, Muhammad AbuBakar Minhas, Momina Sajid, Muhammad Ryed
LUMS
Lahore, Pakistan
26100256@lums.edu.pk,26100102@lums.edu.pk,26100156@lums.edu.pk,26100232@lums.edu.pk,26100363@lums.edu.pk

Abstract

News article classification is essential in improving the delivery of personalized content. The main goal of this project is to develop machine learning models for classifying Urdu news articles into categories, such as entertainment, business, sports, and more. The use of web scraping helped us gather more than 1,000 articles from three news websites, Express, Geo, Dunya, and Jang, and then preprocessing methods to get the data ready for analysis. Three unique machine learning models were implemented and assessed for accuracy and effectiveness. Our findings emphasize the model that performed the best in classification.

ACM Reference Format:
Fatima Mohsin Khan, Khadija Imran, Muhammad AbuBakar Minhas, Momina Sajid, Muhammad Ryed . 2024. News Article Classification Using Machine Learning Models. In . ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Methodology

1.1 Web Scraping

A class called 'NewsScraper' was used to carry out the web scraping process. It consists of four methods: 'get_express_articles()', get_geo_articles(), 'get_dunya_articles()', and 'get_jang_articles'. These methods were used to map the data from each specific news website to a dictionary with keys:

- **id** - Unique ID for each article.
- **title** - Headline of the article.
- **link** - URL to the full article.
- **content** - Article's main text.
- **gold_label** - Category of the article.

The scraping process for each website follows these steps:

(1) **Iterating over categories and pages**: The outer loop iterates over each category in the list, and the inner loop iterates over page numbers.
(2) **Fetching the webpage**: Constructs the URL for the given category and page, sends a GET request, and raises exceptions for failed responses. BeautifulSoup parses the HTML content of the page.
(3) **Finding and Parsing Articles**: Identifies articles using a specific HTML structure.
(4) **Scraping each article**: Extracts the title and link for each article. The main text is extracted from the 'span' tags with the 'story-text' class and combined into a single string.
(5) **Storing data**: Appends the extracted data to a Pandas DataFrame.
(6) **Logging progress**: Logs the number of articles successfully scraped.

1.2 Exploratory Data Analysis (EDA)

During the EDA phase, we encountered different category labels for each website. To ensure consistency and eliminate redundancy, all categories were standardized into six unique labels. For example, 'saqafat' was replaced with 'entertainment'. If columns had missing values, they were dropped, and duplicates were removed. Regex functions were used to clean the text by removing punctuation, HTML tags, special characters, and extra spaces. The final dataset contained 1,204 samples with even distribution among the categories, no skewness was observed.

1.3 Logistic Regression

For text classification, we used the 'TfidfVectorizer' to convert the 'content' column (textual data) into numerical vectors based on word frequencies. We set the vocabulary size to 11,000, as increasing the vocabulary size beyond this threshold negatively affected model accuracy. The categorical labels in the 'gold_label'column were mapped to integers (e.g., entertainment $\rightarrow$ 0). The dataset was split into training (80%) and testing (20%) sets.

The 'LogisticRegression' class was implemented to build the model. This class consists of the following features:

- **sigmoid function**: Computes the sigmoid activation function, which outputs probabilities between 0 and 1.
- **fit**: Implements batch gradient descent. The sigmoid function is used to convert the linear output into probabilities, minimizing the cost function until the number of epochs (2000) is reached.
- **predict**: Calculates the hypothesis function $h = X \cdot w + b$ and converts probabilities into binary labels (0 or 1).
- **evaluate**: Computes the confusion matrix, accuracy, and F1 score.

For multiclass classification, five separate logistic regression models were trained using the One-vs-All strategy. Each model was trained for one class versus the rest. Probabilities for all classes were collected, and the label with the highest probability was chosen as the final prediction.

1.4 Neural Network

The preprocessing steps for the neural network included tokenizing the text data, padding sequences to ensure consistent vector dimensions, and converting the processed data into PyTorch tensors. The 'TextClassifier' model consisted of:

- An embedding layer to map word indices to dense vectors.
- A linear layer followed by a ReLU activation function.

Conference'17, July 2017, Washington, DC, USA
2024. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- Another linear layer to obtain the logits for each class.

The model was trained using the Adam optimizer and the cross-entropy loss function. The training loop was run for 100 epochs, with the model parameters updated via backpropagation to minimize the cross-entropy loss. After 100 epochs, the model's performance was evaluated.

1.5 Multinomial Naive Bayes

For Multinomial Naive Bayes, the text data was preprocessed by removing digits, extra spaces, special characters, and stop words. The bag-of-words (BoW) model was used for feature extraction, where the 'CountVectorizer' transformed the text into a matrix of word counts. The 'NaiveBayesMultinomial' class computed the log-prior probabilities and log-likelihoods for each word in each class using additive smoothing. The class with the highest log-probability was selected as the predicted label.

A few articles had repeatedly shown misclassification; different approaches and multiple random states were used, but these articles were misclassified every time. After manually inspecting problematic indices, 10 articles were removed from the data set. Accuracy was increased by 2.9 percent

2 Findings

In order to categorize Urdu news articles into predetermined categories, this study investigated the effectiveness of three machine learning models: Multinomial Naive Bayes (MNB), Logistic Regression (LR), and Neural Networks (NNs). In order to shed light on the accuracy of these models on a language and dataset that are neglected by content classification techniques.. According to the results, MNB outperformed LR (96 percent) and NNs (95 percent), achieving the best classification accuracy (97.91 percent).

The compatibility of MNB with the dataset's features is the reason for its excellent performance. Because it assumes a multinomial distribution of features and effectively handles sparse, high-dimensional inputs like term frequency-inverse document frequency (TF-IDF) vectors, MNB is especially well-suited for text data. MNB was probably able to generalize more well across categories because of this alignment with the data structure. It was also successful because it is likely to overfit.

On the other hand, LR's dependence on a linear decision boundary limited its performance. The dataset may not have addressed class imbalance or included non-linear correlations between characteristics and target labels, which would have made it more difficult for LR to represent the underlying patterns accurately.

Despite its strength, the NN model has drawbacks, most likely due to feature sparsity and inadequate data quantity. Large datasets and dense embeddings are ideal for neural networks; nevertheless, the comparatively short dataset (1000 articles) and high dimensionality were challenging. Furthermore, inadequate regularization and hyperparameter tweaking strategies prevented the NN from reaching its full potential.

3 Limitations

In order to categorize Urdu news articles into predetermined categories, this study investigated the effectiveness of three machine learning models: Multinomial Naive Bayes (MNB), Logistic Regression (LR), and Neural Networks (NNs). In order to shed light on the accuracy of these models on a language and dataset that are neglected by content classification techniques.. According to the results, MNB outperformed LR (96 percent) and NNs (95 percent), achieving the best classification accuracy (97 percent).

The compatibility of MNB with the dataset's features is the reason for its excellent performance. Because it assumes a multinomial distribution of features and effectively handles sparse, high-dimensional inputs like term frequency-inverse document frequency (TF-IDF) vectors, MNB is especially well-suited for text data. MNB was probably able to generalize more well across categories because of this alignment with the data structure. It was also successful because it is likely to overfit.

On the other hand, LR's dependence on a linear decision boundary limited its performance. The dataset may not have addressed class imbalance or included non-linear correlations between characteristics and target labels, which would have made it more difficult for LR to represent the underlying patterns accurately.

Despite its strength, the NN model has drawbacks, most likely due to feature sparsity and inadequate data quantity. Large datasets and dense embeddings are ideal for neural networks; nevertheless, the comparatively short dataset (1000 articles) and high dimensionality were challenging. Furthermore, inadequate regularization and hyperparameter tweaking strategies prevented the NN from reaching its full potential.

These results highlight how well MNB performs text classification tasks, especially in settings with limited resources, such as Urdu news classification. The findings also show that careful preprocessing, feature engineering, and model selection are necessary to optimize machine learning systems' performance for natural language processing tasks. To increase the performance of NNs and LR, future research could investigate deeper neural architectures, different embeddings, and better regularization.

4 Conclusion

With Multinomial Naive Bayes performing the best with an accuracy of 97.91 percent, this study showed that it is feasible to identify Urdu news items using machine learning models. The results highlight how crucial it is to match model selection to the dataset's properties. MNB was the best option for this work because of its ease of use and compatibility with sparse text data. Despite being marginally less accurate, logistic regression and neural networks demonstrated promise with additional refinement and improvement.

The findings point to critical areas that require improvement, such as growing the dataset, adding sophisticated feature representations, and perfecting preprocessing methods. Resolving these issues could reduce the performance difference between more complicated models like neural networks and simpler models like MNB.

This project made enhancing automatic text classification in underrepresented languages more possible. It establishes the foundation for future studies to investigate more complex models, more datasets, and wider applications in natural language processing for underserved languages by aiding in creating tailored news systems.