Imran Khan: Entertainment or Sports? Classification of Urdu News Articles

Group Number: 15

ACM Reference Format:
Group Number: 15. 2024. **Imran Khan: Entertainment or Sports? Classification of Urdu News Articles** . In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 ABSTRACT

Text classification plays a pivotal role in organizing unstructured data for efficient retrieval and analysis. In this study, we address the problem of article type classification, predicting the category of newspaper articles into one of five classes: *world, business, entertainment, sports,* or *science-tech*. Using a dataset scraped from five prominent sources (*Express, Geo, Dunya, Jhang, Dawn*), we implemented three models—*Logistic Regression, Naive Bayes, and Neural Networks*—to assess and compare their performance. Each model leverages distinct algorithmic strengths, from probabilistic reasoning to linear regression and deep learning. This paper details the methodology, findings, and limitations of the study, providing insights into practical text classification challenges.

2 INTRODUCTION

With the explosion of unstructured textual data, automated text classification has become indispensable. The task involves assigning predefined labels to text, which in our case are the categories of newspaper articles. Our objective was to classify articles into five classes using three different machine learning models. The models were chosen to balance simplicity and complexity, enabling us to explore trade-offs in computational efficiency, accuracy, and interpretability.

In this work, we scraped data from multiple sources, applied preprocessing techniques, and implemented Logistic Regression, Naive Bayes, and Neural Networks. The findings from each model are analyzed to understand their respective strengths and weaknesses, ultimately contributing to the growing body of knowledge in text classification.

Before we get into the nitty-gritty of our process, we fist need to answer an obvious question. **Why these models?** the world of machine learning has introduced a large number of models each with its own unique strengths and of-course weaknesses.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The first model of choice was a multinomial Logistic Classifier. Its simplicity and interpretability provide a clear understanding of the feature-class relationships, making it an excellent baseline for comparison. Logistic regression is inherently well-suited for multi-class classification through its extension to the one-vs-all approach, and it performs efficiently in high-dimensional, sparse datasets typical of text classification tasks. Regularization techniques enhance its ability to generalize, particularly in the presence of limited data. Furthermore, logistic regression's proven track record in text classification ensures its reliability as a robust model for our article type prediction problem.

The Multinomial Naive Bayes (MNB) algorithm is a simple yet powerful tool for classifying text, widely used in tasks like news categorization and spam detection. It's based on Bayes' theorem and works by calculating the probability of a document belonging to a particular class based on the words it contains. What makes MNB unique is its ability to handle word frequencies effectively while assuming that all words are independent of each other given the class. Although this assumption doesn't always hold in real-world scenarios, MNB is incredibly efficient, easy to implement, and often delivers surprisingly accurate results, making it a popular choice for text classification tasks.

The third model of choice is a neural network, selected for its ability to automatically learn patterns from data and its flexibility in design. Its architecture can be adjusted from simple to highly complex by tuning various hyperparameters, such as the number of layers, neurons, and learning rates. This adaptability enables the model to capture subtle patterns and contextual nuances in article content that simpler models might overlook or fail to comprehend. The neural network's capacity to model intricate relationships within text data makes it particularly suited for handling the diverse and context-rich nature of newspaper articles.

3 METHODOLOGY

3.1 Data Collection

The dataset consisted of newspaper articles scraped from five major websites: *Express, Geo, Dunya, Jhang, and Dawn*. Using the *BeautifulSoup* Python library, we extracted article content from the HTML structure of each webpage. Different scraping logic was implemented for each source due to structural variations in their HTML, such as differences in `<div>` tags and class names. A single unified scraping script was not feasible as each source organized its articles uniquely.

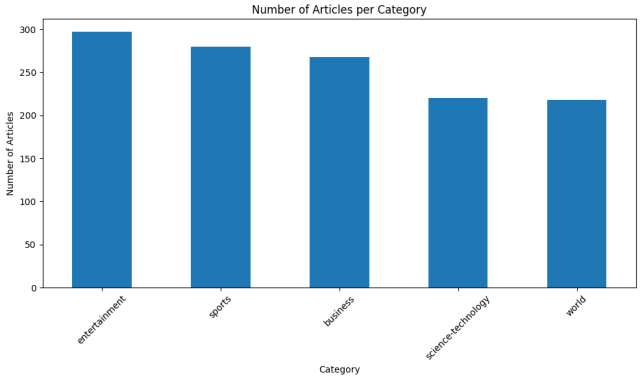

Figure 1: visualizing the class distribution

3.2 Preprocessing

After scraping, the data was consolidated into a CSV file. Data Wrangler extension was used to visualize the class distributions and the number of scraped articles. Preprocessing steps included:

- **Handling Missing and Duplicated Data:** Null entries and duplicate rows were removed.
- **Stopword Removal:** A custom stopword list of 570 terms was used to filter out high-frequency but uninformative words. These stopwords were extracted from Kaggle.
- **Regular Expressions:** Text normalization techniques were applied to remove punctuation and special characters. The Regex library from Python was used for this purpose.

The preprocessed data was then tokenized using the Bag-of-Words model (*CountVectorizer*) to represent textual content numerically. Labels were one-hot encoded for compatibility with machine learning algorithms.

3.3 Models and Implementation

3.3.1 Logistic Regression: Logistic Regression was implemented with regularization to prevent overfitting. This ensures that weights remain small, pruning features that do not contribute significantly to the model's outcome. The loss function used was cross-entropy, and the optimization employed batch gradient descent. Hyperparameter tuning involved:

- Learning rates: {0.1, 0.01, 0.001}
- Epochs: {500, 750, 1000}
- Regularization parameter (λ): 0.001

Performance was measured on both training and test sets, with comparisons drawn against Scikit-learn's implementation of Logistic Regression. Various combinations of learning rates and epochs were explored to identify the best-performing configuration for the data. The training of the model employed the One-vs-All approach, with testing conducted under the same criteria.

3.3.2 Multinomial Naive Bayes: A Multinomial Naive Bayes classifier was chosen for its probabilistic framework. The prior probabilities for each class were calculated as:

$$P(\text{class}) = \frac{\text{Number of articles in class}}{\text{Total number of articles}}$$

Conditional probabilities for words were computed using Laplace smoothing:

$$P(w \mid c) = \frac{\text{Occurrences of word } w + 1}{\text{Total words in } c + \text{Vocabulary size}}$$

The smoothing technique mitigated errors due to out-of-vocabulary words and ensured balanced class distributions. Different values of the smoothing parameter (α) were tested on the validation set, with $\alpha = 1$ yielding the best results. This value was incorporated into the model to evaluate the test set.

3.3.3 Neural Networks: The Neural Network model was implemented using PyTorch. The dataset was split into training (70%), validation (15%), and test (15%) sets using `torch.utils.randomsplit`. Data was processed in batches using PyTorch DataLoaders.

Key hyperparameters included:

- Number of hidden layers and units
- Dropout rate and weight decay
- Learning rate and batch size
- Epochs

The *itertools* library was used to generate 486 permutations of hyperparameters. For each configuration, models were trained and evaluated three times to account for randomness.

4 FINDINGS

4.1 Observations

Class imbalance emerged as a significant challenge, as minority classes (e.g., *science-tech*) were often misclassified. All models showed varying degrees of sensitivity to this imbalance.

4.2 Model-Specific Results

4.2.1 Naive Bayes: The Naive Bayes model achieved the highest accuracy of **94%** with a smoothing parameter of $\alpha = 1.0$. Misclassifications primarily occurred between categories with overlapping vocabulary, such as *business* and *science-tech*, where lexical similarities posed challenges for the model. The smoothing parameter α, which controls the extent of Laplace smoothing, played a significant role in the model's performance. Smaller values of α (e.g., 0.1) made the model rely more heavily on observed word frequencies, which increased variance but sometimes led to overfitting on less frequent words. Conversely, larger values of α (e.g., 2.0) resulted in more aggressive smoothing, reducing overfitting but occasionally underestimating the importance of significant terms. This highlights the inherent Bias-Variance tradeoff in the selection of α. fig.2 shows the relationship of alphas on the validation accuracy

4.2.2 Logistic Regression: The Logistic Regression model achieved a best accuracy of **93%** with a learning rate of 0.01 and 750 epochs. Smaller learning rates reduced overall accuracy, as the optimizer struggled to converge to the minima within the given epochs. Increasing epochs beyond 750 did not yield significant accuracy gains, as the model had already converged. Overfitting occurred with larger epoch counts, as training accuracy climbed while test accuracy suffered. A middle ground was established with a learning rate of 0.01 and 750 epochs to balance stability and generalization, emphasizing the importance of hyperparameter tuning. Refer to

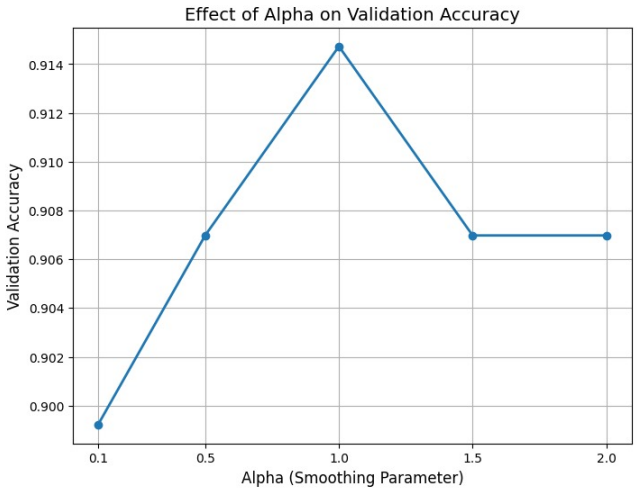

Figure 2: Bayes: alpha vs Accuracy

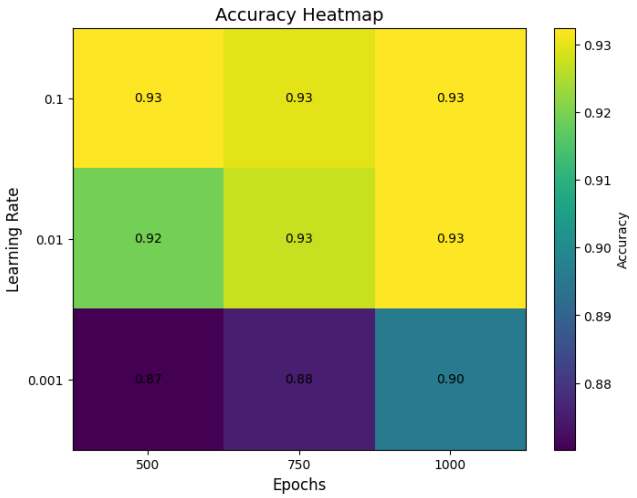

Figure 3: Logistic Classifier Results

fig.3 to see the effects of epochs and learning rates on accuracy of the model.

4.2.3 Neural Networks: Despite extensive hyperparameter tuning, the Neural Network model achieved a maximum accuracy of **93%**. This model was the most prone to overfitting, likely due to the small dataset size, causing it to memorize training data rather than generalizing effectively. I experimented with 486 hyperparameter permutations, analyzing the validation loss and accuracy for each combination. Boxplots were generated for all hyperparameters versus validation accuracy (as shown in Fig. ??), which revealed limited variation in the distributions of different hyperparameters. This indicates that the Neural Network consistently performed well, achieving approximately 93% accuracy across most parameter configurations.

For weight decay and dropout rate, their addition did not increase validation accuracy. The training accuracy often reached 100%, highlighting overfitting during training, which negatively impacted performance on the validation set. For the learning rate, a similar trend was observed. When the learning rate was decreased, it delayed overfitting and improved validation accuracy by allowing the model to generalize better. Regarding the hidden layer configuration, a simpler model (e.g., one hidden layer) performed as well as a more complex architecture, suggesting that added complexity was unnecessary. Therefore, I opted for the simpler configuration, as it reduced computational time and minimized model complexity without compromising performance.

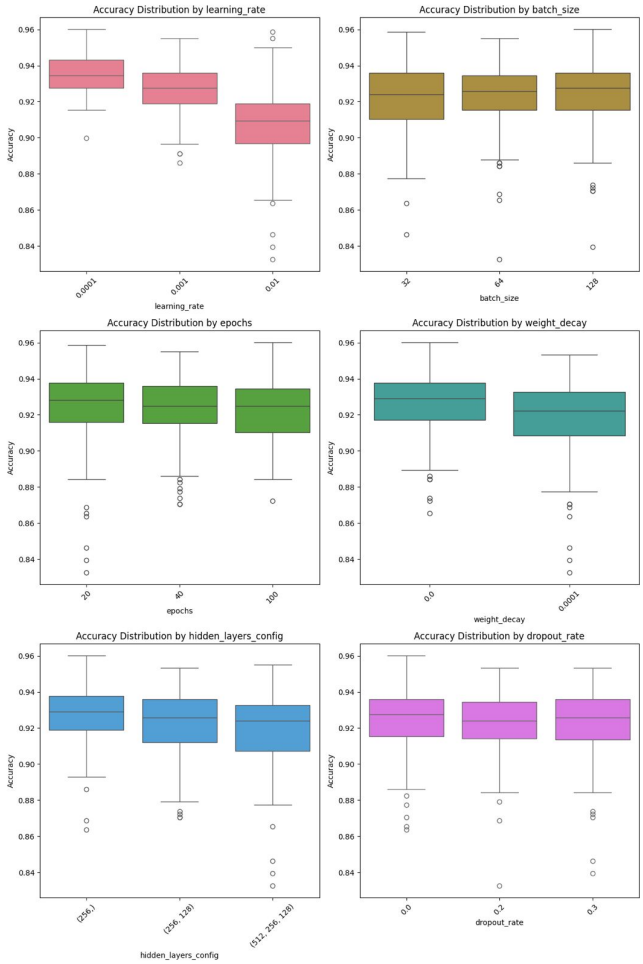

Figure 4: Neual Networks: effects of hyperparameters

5 CONCLUSION

Our study demonstrates that while Neural Networks provide greater flexibility and capacity for handling complex patterns, simpler models like Logistic Regression and Naive Bayes can achieve comparable accuracy with significantly higher interpretability and efficiency. The Naive Bayes model's probabilistic framework and Logistic Regression's linear nature make them easier to implement and analyze, particularly for smaller datasets. However, the inherent

Table 1: Model Performance Metrics

Model	Prec	Recall	F1-Score	Acc
Logistic Reg	0.92	0.92	0.92	0.93
SKlearn Logistic Reg	0.92	0.92	0.92	0.93
MNB	0.92	0.92	0.93	0.94
SKlearn MNB	0.92	0.92	0.92	0.92
Neural Network	0.93	0.93	0.93	0.93

challenges of class imbalance and limited dataset size constrained the performance of all models. Addressing these limitations through techniques like oversampling, class weighting, or collecting a more extensive dataset would significantly enhance the robustness and generalizability of the models.

Importantly, our analysis suggests that accuracy alone should not be the sole standard for determining a "better" model. Plotting a confusion matrix, for instance, provides valuable insights into specific misclassifications and helps understand the impact of imbalanced data. Such visualizations offer a more comprehensive picture of model performance, beyond a single metric like accuracy.

From an interpretability perspective, the Naive Bayes and Logistic Regression models stand out due to their higher transparency. Both models allow for straightforward interpretation of hyperparameters and their effects, making them more accessible to understand and debug. However, Neural Networks excel in post-hoc interpretability. Their complexity supports the generation of various graphs and diagrams, such as loss curves, activation heatmaps, and performance trends over epochs, providing rich insights into the learning process.

Future work could involve incorporating advanced features like word embeddings or leveraging contextual models such as transformers to enhance classification performance further. Additionally, integrating visualization tools and interpretability frameworks would make these models more transparent and actionable for real-world applications. By focusing not only on accuracy but also on interpretability, fairness, and contextual understanding, the development of text classification models can become more balanced and effective. upon further testing ,imran khan on all three models came to the similar conclusion : IMRAN KHAN : IN JAIL...