Classification of BBC News Articles

Qize Zhang, Wentao Deng, Yifan Jiang

June 6, 2024

Abstract

The explosion in the volume of internet content, particularly news articles, has created massive volumes of unstructured text data. This project focuses on developing machine learning models to accurately classify sizable news corpora from the BBC full-text document classification dataset into five categories: business, entertainment, politics, sport, and tech. We aim to compare the efficiency of different NLP techniques and methods for accurate categorization of news content.

1 Introduction

The Digital Times has upped the ante on the volume of internet content, more so in news articles, which has created massive volumes of unstructured text data. The explosion in available information has, therefore, brought along the challenge of efficient categorization, analysis, and retrieval by individuals and organizations. Automated text classification is an application of Natural Language Processing wherein predefined categories and text documents are obtained from their content. This process is crucial for enhancing information retrieval, content filtering, sentiment analysis, and other applications.

2 Project Focus

We aim to develop machine learning models for accurately classifying sizable news corpora from the BBC full-text document classification dataset into five categories: business, entertainment, politics, sport, and tech. The content on these topics covers rather unique linguistic characteristics and peculiarities related to a peculiar context. Being the first assignment, the main objective is to compare how the highest level of efficiency of different techniques of NLP and various methods treat the problem of accurate categorization of news content.

2.1 Practical Purposes of Automated Text Classification

Automated text classification serves several important practical purposes:

- **Information Retrieval:** Improving search engines to return relevant results with greater accuracy in categorizing articles, enhancing user experience with targeted content.
- **Content Filtering:** Helping filter content and recommend news based on user preferences, especially useful for personalized news feeds and recommendations.
- **Sentiment Analysis:** Useful for businesses and organizations to understand public opinion on various topics by identifying the sentiment towards news items, helping to evaluate public response and shape strategies accordingly.

3 Objectives

The BBC dataset, therefore, poses many challenges in that it has different categories well defined. Using this dataset, we are aiming to achieve the following objectives:

- **Text Data Preprocessing:** Cleaning and standardizing the text to extract features and model—in other words, providing the models with high-quality input data.
- **Extract Meaningful Features:** Using term frequency/inverse document frequency (TF-IDF) to transform the text into a numerical representation that captures the importance of features to the respective class, aiding in determining the relative significance.

- **Training and Evaluating Models:** Implement the various machine learning models, including Naive Bayes and Support Vector Machine (SVM), to determine which model best suits this type of classification problem. This involves experimenting with different models and tuning their parameters.
- **Model Performance Assessment:** This step involves examining the model with multiple metrics, including but not limited to accuracy, precision, and recall, while employing the Kappa statistic to help ensure a more comprehensive understanding of their performance. It illuminates the strengths and weaknesses of each model.

By the end of this project, therefore, it will aim to have established the effectiveness of the selected models in implementing best practices for text classification of news articles. This report presents methodology, results, and conclusions drawn from the implementation and evaluation of this and other models in the application domain, along with possible future work.

4 Methodology

4.1 Libraries and Data Loading

For this project, we used several R libraries such as `tm`, `text2vec`, `caret`, `e1071`, `keras`, `naivebayes`, `reticulate`, `stringr`, and `dplyr`. Reading all the text files and concatenating them into a single data frame that includes two columns: one for text and another for category. The 'text' column comprised the article content, while the 'category' column had the corresponding category labels. In total, there are 2225 text files, and the amount of each category is:

- **business:** 510
- **entertainment:** 386
- **politics:** 417
- **sport:** 511
- **tech:** 401

4.2 Data Preprocessing

It was in data preprocessing that a text corpus was developed, which means a large and structured collection of written texts to apply a variety of transformations to standardize and clean the text data.

- **ASCII Conversion:** Ensured compatibility by converting from UTF-8 text to ASCII.
- **Lowercasing:** Maintained uniformity by converting all characters to lowercase.
- **Removing Punctuation and Numbers:** Cleaned the text by removing extraneous characters.
- **Removing Stop Words:** Filtered out common words that do not contribute significantly to the text's meaning.
- **Stripping Whitespace:** Removed extra spaces to standardize text format.

The preprocessed text was then converted into a document-term matrix (DTM) weighted with TF-IDF to approximate the importance of the terms relative to the dataset.

4.3 Feature Extraction

Feature Extraction represents extracted text data in numeric form that can be fed into machine learning algorithms. In this work, we have used the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm for feature extraction. TF-IDF is a statistical measure in information retrieval commonly used for evaluating the importance of a term (word) in a document relative to a collection of papers or a corpus.

TF-IDF is a statistical measure commonly used in information retrieval to evaluate the importance of a term in a document relative to a corpus. The formula for TF-IDF is given by:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t) \tag{1}$$

where $\text{TF}(t, d)$ is the term frequency with which a term occurs concerning a document of term t in document d, and $\text{IDF}(t)$ is the inverse document frequency of term t, which quantifies the importance of a term across a corpus. The IDF for terms found in many documents is lower because it is assumed they are less informative. .

5 Model Training and Evaluation

In our study, we trained and evaluated seven different machine learning models: Naive Bayes, Support Vector Machine, Random Forest, K-Nearest Neighbors, Decision Tree, Logistic Regression, Gradient Boosting, and LightGBM. These have been chosen because they are effective in text classification tasks and complementary to each other. Below are the general steps we followed for all the models:

Data Splitting:

The dataset was split into training (80%) and testing (20%) sets.

Model Training:

A Naive Bayes model was trained using the training data.

Predictions:

The model made predictions on the test data.

Evaluation:

The confusion matrix and various metrics such as accuracy, sensitivity, specificity, and Kappa statistic were computed.

5.1 Naive Bayes Model

Naive Bayes is a supervised learning algorithm under the Bayes theorem, mainly used for classification purposes. It assumes feature independence given the class label, and this supports the idea that the appearance of one feature does not affect the appearance of the others, given the class. The assumption thus simplifies the model, rendering it most effective for text classification tasks in spam detection, sentiment analysis, and categorizing documents. Naive Bayes is popular because it is simple, easy to implement, and computationally efficient for both training and prediction, even on huge datasets. It performs well with large-scale, high-dimensional, and sparse datasets, which are very common in the case of text processing. In addition, it remains robust with irrelevant features and can manage sparsity without conducting extensive preprocessing, making it a suitable choice for many realistic applications.

5.2 Support Vector Machine (SVM) Model

Support Vector Machine (SVM) is a high-powered supervised learning algorithm, which performs classification and regression tasks. It works by separating classes with an optimal hyperplane in the feature space. This way, it maximizes the margin between classes, making sure that the closest data points supporting the vectors are as far from each other as possible. The support vector machine is very robust to overfitting, and it gives very effective classification procedures for high-dimensional and sparse datasets such as text classification. More specifically, the ability of SVM in text classification lies in handling the intrinsically complicated characteristics of high dimensionality and sparsity in textual data. SVM is effective in this case as it works on the support vectors, which are the most critical elements defining the decision boundary.

5.3 Random Forest Model

Random Forest is a powerful ensemble model that trains multiple decision trees and outputs the class, taking the classes' mode in single trees. Each tree considers a random subset of features for splits, hence enhancing model robustness and generalization. Our work comprises developing a Random Forest model to classify predefined news categories, thus preprocessing data to get our dataset. The model is trained on different subsets of data, and predictions are aggregated for the final classification. We evaluate performance by metrics such as accuracy, precision, recall, and F1-score, in search of optimal accuracy for correct classification and generalization, trying to fine-tune the hyperparameters with the number of trees and tree depth. This procedure takes advantage of

Random Forest strengths for the efficient classification of news items in a test environment, thus underlining the success of ensemble learning when complex classification tasks are at stake.

5.4 K-Nearest Neighbors (KNN) Model

The K-Nearest Neighbors (KNN) model is an intuitive, instance-based learning algorithm to be used in classification. It works by finding the k nearest neighbors to a given data point and classifying the end based on the majority class among those neighbors. Here, we present how the KNN model has been implemented and what results have been achieved for classifying BBC news articles into predefined categories.

5.5 Decision Tree Model

Decision Tree is a non-parametric supervised learning method used in classification modeling. It divides a set of data into subsets based on the value of input characteristics, which are represented in tree-like decisions. Here, we describe the implementation and performance evaluation of the Decision Tree model for classifying BBC news articles into predefined categories.

5.6 Logistic Regression Model

Logistic Regression is a statistical model that uses a logistic function to model a binary dependent variable. Multinomial logistic regression is used for multiclass classification in other domains. This section describes the implementation and evaluation of the Logistic Regression model for the classification task of the BBC news articles into pre-defined categories.

5.7 Gradient Boosting Model

Gradient boosting is an effective ensemble learning algorithm: it sequentially builds an ensemble of models such that, in each stage, the new model corrects the errors of the last one. This enhances overall prediction accuracy. Given this, gradient boosting would find its best application in classification problems as one weak classifier can be combined into a robust classifier. We have implemented a Gradient Boosting classification model on categorizing BBC news articles into predefined categories such as politics, sports, technology, business, and entertainment. It begins with a lot of data preprocessing: data cleaning, stop-word removal, and text conversion into numerical forms using techniques like TF-IDF. After that, the data set is divided into the training and testing subsets in a meticulous study of model performances. The Gradient Boosting model is carefully trained for the effective classification of news articles.

5.8 LightGBM Model

LightGBM (Light Gradient Boosting Machine) is a gradient-boosting framework designed to be distributed and efficient with speed. It is best for big data and high dimensionality because it will lead to fast performance. Its execution speed includes innovations in using the Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) technique, which allows for better optimization at the end of the day, both in speed and accuracy. In this project, we use LightGBM to classify BBC news articles such as politics, sport, technology, business, and entertainment. This included preprocessing the textual data: cleaning, stop-wording, and numerical conversion of the text using TF-IDF. On splitting the data for training and testing, our LightGBM model showed a high degree of accuracy and efficiency in categorizing articles—a fact attesting to the strength of this model type in large-scale text classification problems.

6 Results

6.1 Naive Bayes Model

The Naive Bayes model achieved an overall accuracy of 75.45%, with detailed statistics provided in Table 1.

Table 1: Naive Bayes Model Results

Metric	Value
Accuracy	0.9189
95% Confidence Interval	(0.8895, 0.9426)
Kappa	0.8981
Sensitivity	Varies across classes
Specificity	High for sport (0.9804) and tech (0.9250)

6.2 SVM Model

The SVM model significantly outperformed the Naive Bayes model with an accuracy of 97.3%. Detailed statistics are provided in Table 2.

Table 2: SVM Model Results

Metric	Value
Accuracy	0.9775
95% Confidence Interval	(0.959, 0.9891)
Kappa	0.9717
Sensitivity	High across all classes
Specificity	Perfect scores for sport and tech

6.3 Random Forest Model

The Random Forest model achieved high accuracy and balanced accuracy across all classes, as shown in Table 3.

Table 3: Random Forest Model Results

Metric	Value
Accuracy	0.9482
95% Confidence Interval	(0.9233, 0.9669)
Kappa	0.9349
Sensitivity	High across all classes
Specificity	Perfect scores for business and sport

6.4 K-Nearest Neighbors (KNN) Model

The KNN model achieved moderate accuracy, with detailed statistics provided in Table 4.

Table 4: KNN Model Results

Metric	Value
Accuracy	0.7387
95% Confidence Interval	(0.6952, 0.779)
Kappa	0.6744
Sensitivity	High for politics, low for business and entertainment

6.5 Decision Tree Model

The Decision Tree model achieved moderate accuracy, with detailed statistics provided in Table 5.

Table 5: Decision Tree Model Results

Metric	Value
Accuracy	0.7590
95% Confidence Interval	(0.7165, 0.7981)
Kappa	0.6957
Sensitivity	High for sport, low for tech and entertainment

6.6 Logistic Regression Model

The Logistic Regression model achieved high accuracy and balanced accuracy across all classes, as shown in Table 6.

Table 6: Logistic Regression Model Results

Metric	Value
Accuracy	0.9752
95% Confidence Interval	(0.9561, 0.9876)
Kappa	0.9689
Sensitivity	High across all classes
Specificity	Perfect or near-perfect scores for business, sport, and entertainment

6.7 Gradient Boosting Model

The Gradient Boosting model achieved high accuracy and balanced accuracy across all classes, as shown in Table 7.

Table 7: Gradient Boosting Model Results

Metric	Value
Accuracy	0.9369
95% Confidence Interval	(0.9101, 0.9577)
Kappa	0.9208
Sensitivity	High across all classes
Specificity	High for business and sport

6.8 LightGBM Model

The LightGBM model achieved moderate accuracy, with detailed statistics provided in Table 8.

Table 8: LightGBM Model Results

Metric	Value
Accuracy	0.2072
95% Confidence Interval	(0.1704, 0.2479)
Kappa	0.0098
Sensitivity	Low across all classes
Specificity	Low across all classes

6.9 Term Explanation

Accuracy

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Number of Cases}}$$

95% Confidence Interval

The 95% Confidence Interval is typically expressed as:

$$\text{CI}_{95\%} = \left(\hat{\theta} - 1.96 \cdot \text{SE}, \hat{\theta} + 1.96 \cdot \text{SE}\right)$$

where $\hat{\theta}$ is the estimated parameter and SE is the standard error of the estimate.

Kappa (Cohen's Kappa)

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

where P_o is the observed agreement, and P_e is the expected agreement by chance.

Sensitivity (True Positive Rate or Recall)

$$\text{Sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Specificity (True Negative Rate)

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

7 Discussion

Several machine learning models were tested for classifying the BBC news articles, allowing a comprehensive assessment in this project. Of them all, the best result was shown by the Support Vector Machine model,

attaining a fantastic 97.75%. The support vector machine model is robust when dealing with high-dimensional and sparse data. It led to massive improvement because of its better performance.

The Random Forest model was also very accurate, with a performance of 94.82% accuracy thus being the best alternative to the SVM model. Logistic Regression attained an accuracy rate of 97.52%, which was just slightly behind that of the SVM model, and therefore confirms it is an excellent model to apply to the given problem of classifying texts.

Models like K-Nearest Neighbors (KNN) and Decision Tree had a moderate performance at 73.87% and 75.90% accuracy. These presented differential sensitivities and specificities across the classes, and thus, they imply that there is still space for fine-tuning in handling imbalanced datasets.

Surprisingly, among the models, the least accuracy was achieved by the LightGBM model, which stood at 20.72%. Again, a converse was expected because LightGBM has an outstanding reputation for managing large datasets with impressive efficiency and performance. This low performance is based on the poor tuning of parameters or the necessity for more elaborate preprocessing, adjusted for the LightGBM framework.

Overall, the SVM, Random Forest, and Logistic Regression models were seen to be applicable in the adequate classification of news articles, with SVM leading the way.

8 Limitations and Future Work

Despite the excellent performance of the SVM model, there are limitations and potential areas for future work:

- **Parameter Tuning:** Hyper-parameter tuning can be finer to increase the performance of the models, especially for models like LightGBM, which showed poor results because of the possibly suboptimal parameters.
- **Model Complexity:** An increase in complexity by adding more complex models, such as LSTM and BERT, to the deep learning implementation may further fuel accuracy in text data classification when hidden patterns are dealt with.
- **Feature Engineering:** While TF-IDF created a perfect basis for feature extraction, adding features like Word Embeddings—for instance, Word2Vec and GloVe—can bolster model performance through the capability to capture semantic similarities of words.
- **Dataset Expansion:** Expanding the dataset to include more diverse news articles can improve generalization for new, unseen data.
- **Cross-Validation:** Applying cross-validation techniques to estimate model performance and avoid overfitting.

9 Conclusion

This project does well in classifying BBC news articles into predefined categories by applying some different kinds of machine learning models. More importantly, the performance of the SVM model is excellent in this kind of text classification task.

The Random Forest and Logistic Regression models also gave excellent and reliable results for such tasks. We obtained a robust classification system through careful preprocessing, feature extraction using TF-IDF, and complete model evaluation to classify news articles correctly. Except for some underperformance of the models, the insights provide an excellent stepping stone for future improvement and growth in finer text classification.

We could, by responding to identified limitations in the literature and pursuing research of more advanced techniques, continue to develop the models toward higher accuracy and robustness; more sophisticated and reliable text classification systems will likely be created in the future.

A Appendices

A.1 Code

Link to our code Repositories: `https://github.com/jiangyifan0421/Math-156-Final-Project`

A.2 Raw Dataset

Kushwaha S. (n.d.). BBC Full-Text Document Classification Dataset. Retrieved from https://www.kaggle.com/datasets/shivamkushwaha/bbc-full-text-document-classification/data

A.3 References

- Greene D. & Cunningham P. (2006). Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering. Proceedings of the 23rd International Conference on Machine Learning (ICML).
- Scikit-learn. (n.d.). Support Vector Machines (SVM). Retrieved from https://scikit-learn.org/stable/modules/svm.html
- Murphy, K. P. Naive Bayes classifiers. University of British Columbia. Retrieved from https://datajobs.com/data-science-repo/Naive-Bayes-[Kevin-Murphy].pdf
- Biau, G., & Scornet, E. (2016). A Random Forest Guided Tour. Retrieved from https://arxiv.org/abs/1511.05741v1
- Rajani, N. F., Krause, B., Yin, W., Niu, T., & Socher, R. (2020). Explaining and Improving Model Behavior with k Nearest Neighbor Representations. Retrieved from https://arxiv.org/abs/2010.09030v1
- Song, Y. Y., & Ying, L. U. (2015). Decision tree methods: applications for classification and prediction. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4466856/
- Hilbe, J. M. (2009). Logistic regression models. Stata Press. Retrieved from https://www.stata.com/bookstore/pdf/HILBE_LRM_ERRATA27Mar2010.pdf
- Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2020). A Comparative Analysis of XGBoost. Retrieved from https://arxiv.org/abs/1911.01914v1