Naive Bayes and Text Mining

Introduction

This report provides an overview of the entire codebase, covering data preprocessing, feature extraction, model training, evaluation, sentiment analysis, and overall implementation of various tasks.

Data Exploration and Preprocessing

The dataset contains raw text and labels. The preprocessing steps include:

- Loading Data: Read the dataset into a Pandas DataFrame.
- Data Cleaning: Remove punctuation, convert to lowercase, eliminate stopwords, and perform tokenization.
- Feature Extraction: Convert text data into numerical format using TF-IDF vectorization.
- Handling Missing Values: Checked and addressed missing data issues if any.

Text Classification using Naive Bayes

The classification model was implemented using Multinomial Naive Bayes.

Steps:

1. Data Splitting: Divided into 80% training and 20% testing.
2. Model Training: Trained using TF-IDF features.
3. Predictions: Applied the trained model to classify test data.
4. Evaluation: Used metrics such as accuracy, precision, recall, and F1-score.

Sentiment Analysis

Sentiment analysis was performed to determine whether blog posts express positive, negative, or neutral sentiments. The sentiment model identified patterns in text to classify the emotional tone of the content.

Steps:

1. Used a pretrained sentiment analysis model.
2. Applied sentiment classification to each blog post.
3. Analyzed the distribution of sentiments across different categories.

Model Evaluation

The performance of the Naive Bayes model was assessed using accuracy, precision, recall, and F1-score. The classifier achieved 82% accuracy, with high precision and recall in sports and technology categories but lower recall in political and religious topics due to overlapping terms. The confusion matrix revealed misclassifications in categories with similar vocabulary, highlighting areas for improvement.

Accuracy:	82%
Precision:	83.50%
Recall:	83.08%
F1 Score:	81.61%

Additional Implementations

Hyperparameter Tuning: Explored tuning parameters such as alpha in Naive Bayes to improve model performance. Adjusting this parameter helped refine probability distributions and enhance classification accuracy.

Visualization: Plotted confusion matrices to analyze misclassifications and generated sentiment distribution graphs to visualize trends in blog post emotions.

Feature Importance Analysis: Identified key terms influencing predictions, providing insights into how words impact classification outcomes.

Challenges & Improvements

Challenges:

- Class imbalance in some categories affected classification accuracy.
- Vocabulary overlap in similar topics caused misclassifications.
- Sentiment ambiguity in certain texts affected sentiment analysis results.

Improvements:

- Use bigrams/trigrams for better context understanding.
- Experiment with deep learning models for higher accuracy.
- Implement advanced sentiment models for better contextual understanding.

Conclusion

The implemented models successfully classified blog posts and analyzed sentiments. The Naive Bayes classifier achieved 82% accuracy, while sentiment analysis provided insights into content tone. Future enhancements can improve classification and sentiment accuracy, benefiting content moderation and audience engagement.