Sentiment Analysis on Kindle Reviews Using Bag-of-Words, TF-IDF, and Naive Bayes

Sultan Ahmmed

1 Introduction

This project explores the sentiment expressed in Kindle reviews by classifying them as either positive or negative based on the review text. The dataset used for the analysis, `all_kindle_review.csv`, contains user-generated reviews along with their numerical ratings. The column `reviewText` provides the written feedback, while `rating` captures the original star rating given by each reviewer.

The goal of this study is to convert the textual information into meaningful numerical features and evaluate how well machine learning models can learn to distinguish positive sentiments from negative ones. Two feature extraction approaches—Bag-of-Words (BoW) and TF-IDF—were tested with a Gaussian Naive Bayes classifier to compare their effectiveness.

2 Methodology

2.1 Data Loading

The dataset was first loaded into a Pandas DataFrame for exploration and preprocessing. Initial checks were conducted to ensure the dataset was correctly structured and ready for cleaning.

2.2 Initial Exploration

A brief exploration was performed to understand the data:

- The first few rows were inspected to get a sense of the review format.
- The dataset dimensions were confirmed using `df.shape`.
- Missing values were counted across all columns.

- The distribution of star ratings was examined to understand class balance before converting them into binary sentiment labels.

2.3 Data Preprocessing

To prepare the text for modeling, several preprocessing steps were applied:

2.3.1 Converting Ratings to Sentiment Labels

The original 1–5 star ratings were grouped into binary classes. Reviews with ratings below 3 were labeled as negative (0), while ratings of 3 or higher were labeled as positive (1).

2.3.2 Text Normalization

The review text was cleaned through:

- Converting all characters to lowercase.
- Removing special characters, URLs, and HTML tags.
- Eliminating extra spaces.

2.3.3 Stopword Removal and Lemmatization

English stopwords were removed to reduce noise. **WordNet lemmatization** was applied to bring words to their base form (e.g., "running" to "run"), helping to reduce sparsity in the text features.

2.4 Train–Test Split

The cleaned dataset was split into training and testing sets using an 80:20 ratio. The `reviewText` column served as the input features, and the binary sentiment label derived from the ratings served as the target variable.

2.5 Feature Extraction

Two common text vectorization techniques were used:

- **Bag-of-Words (BoW)**: Created using `CountVectorizer`, where each unique word acts as a feature.
- **TF-IDF**: Generated using `TfidfVectorizer` to weigh words not only by frequency but also by how informative they are across the entire corpus.

2.6 Model Selection

Gaussian Naive Bayes was chosen for its simplicity and speed. Although not always ideal for sparse text data, it provides a good baseline for comparison. The model was trained separately on both BoW and TF-IDF representations.

3 Results

3.1 Accuracy Scores

- **BoW Accuracy**: 0.5867
- **TF-IDF Accuracy**: 0.5896

The difference between the two is minor, with TF-IDF performing slightly better.

3.2 Confusion Matrices

BoW

$$\begin{bmatrix} 533 & 276 \\ 716 & 875 \end{bmatrix}$$

TF-IDF

$$\begin{bmatrix} 516 & 293 \\ 692 & 899 \end{bmatrix}$$

3.3 Classification Reports

3.3.1 BoW

```
Class 0: Precision=0.43, Recall=0.66, F1=0.52
Class 1: Precision=0.76, Recall=0.55, F1=0.64
```

3.3.2 TF-IDF

```
Class 0: Precision=0.43, Recall=0.64, F1=0.51
Class 1: Precision=0.75, Recall=0.57, F1=0.65
```

4 Discussion

4.1 Model Performance Comparison

Overall, TF-IDF showed a very slight improvement over Bag-of-Words, indicating that weighting less frequent but more meaningful words helps the model capture sentiment more effectively. However, both approaches still yield relatively low accuracy, suggesting the Gaussian Naive Bayes classifier may not be the best fit for this dataset.

4.2 Challenges With Class 0 (Negative Reviews)

Both models struggled significantly when predicting negative reviews. The confusion matrices reveal high false positives and false negatives for Class 0. This difficulty is partly due to the class imbalance, where positive reviews are roughly twice as common as negative ones. As a result, the model tends to lean toward predicting the majority class.

4.3 Limitations of Gaussian Naive Bayes

Gaussian Naive Bayes assumes features follow a normal distribution, which is not the case for text-based count features (which are sparse and discrete). A more suitable variant, such as Multinomial Naive Bayes, would likely handle the data better. The mismatch between the model's assumptions and the nature of the data contributes to the modest performance scores.

4.4 Future Improvements

Several enhancements can be explored to achieve better performance:

- **Try better suited models**: Logistic Regression, SVM, Random Forest, or gradient boosting methods like XGBoost.
- **Use advanced text representations**: Word embeddings (Word2Vec, GloVe) or contextual embeddings such as BERT.
- **Address class imbalance**: Techniques such as SMOTE, undersampling, or applying class weights during training.

Applying these improvements would allow the model to better capture the nuances in user reviews and produce more reliable sentiment predictions.

5 Conclusion

This project provided a complete pipeline for sentiment analysis using Kindle review data. Although the results from Gaussian Naive Bayes with BoW and TF-IDF were modest, the experiment highlighted important insights about preprocessing, vectorization, and model selection. With more sophisticated techniques and models, the performance can be significantly improved in future work.