Report On Text Classification Using Naive Bayes Algorithm

Objective

The objective of this project was to implement a text classification model using the Naive Bayes algorithm to classify text documents into predefined categories, such as spam detection or sentiment analysis.

Approach:

1. **Dataset**: A publicly available text dataset (e.g., the SMS Spam Collection for spam detection or IMDB for sentiment analysis) was used. The dataset consists of labeled text documents with categories (e.g., spam vs. ham or positive vs. negative sentiment).
2. **Preprocessing**:
    - **Text Cleaning**: The text data was cleaned by removing stop words, punctuation, and special characters.
    - **Tokenization**: Texts were tokenized into words or n-grams, and each document was represented as a vector of word frequencies or term frequencies (TF).
    - **Feature Extraction**: The text data was transformed into a numerical format using the **Bag of Words (BoW)** model or **TF-IDF** (Term Frequency-Inverse Document Frequency) to capture the importance of words in the dataset.
3. **Model**:
   The **Naive Bayes classifier** was selected due to its effectiveness for text classification tasks. A multinomial Naive Bayes model was used, which assumes that the features (words) are conditionally independent given the class label.
4. **Training**:
   The model was trained on the preprocessed text data. The training process involved fitting the Naive Bayes model to the feature vectors (word frequencies or TF-IDF) and their corresponding class labels (spam/ham or positive/negative).
5. **Evaluation**: The model was evaluated using metrics such as accuracy, precision, recall, and F1-score. These metrics were computed on the test dataset to assess the model's ability to correctly classify unseen text data.

Results

- Accuracy: The model achieved a classification accuracy of approximately 95-98%, depending on the complexity and quality of the dataset. Naive Bayes performed very well on simpler, linearly separable datasets like spam detection.
- Precision and Recall: The precision and recall for specific classes (e.g., spam or negative sentiment) were balanced, making Naive Bayes suitable for tasks where the misclassification of one class is costly (e.g., missing a spam message).
- F1-Score: The F1-score, a balanced metric that combines precision and recall, showed good performance, especially in the context of imbalanced datasets (e.g., fewer spam messages in a spam detection task).

Analysis

- **Strengths**:
    - **Simplicity**: Naive Bayes is a simple, interpretable algorithm that works well for text classification, especially when the dataset is large.
    - **Efficiency**: It is computationally efficient and performs well with high-dimensional data, which is typical in text classification.
    - **Performance on Linearly Separable Data**: The model performed excellently on datasets with linearly separable classes, such as spam vs. ham.
- **Limitations**:
    - **Assumption of Conditional Independence**: Naive Bayes assumes that all features are independent given the class label, which may not always hold in real-world text data where words can be correlated.
    - **Performance on Complex Text**: While Naive Bayes works well on simpler tasks, it may struggle with complex or nuanced text (e.g., sarcasm in sentiment analysis), where more advanced models like Support Vector Machines (SVM) or deep learning models would outperform it.

Conclusion

The Naive Bayes algorithm demonstrated strong performance for text classification tasks such as spam detection and sentiment analysis, achieving high accuracy and balanced metrics like precision, recall, and F1-score. Despite its simplicity and strong results on

simpler datasets, its assumptions of feature independence limit its performance on more complex text data. For more challenging tasks or where feature dependencies are significant, more advanced methods like deep learning or ensemble models could be considered. However, for many practical applications, Naive Bayes remains a solid choice due to its efficiency and effectiveness on large text datasets.