Report: Text Classification Using Naive Bayes

Objective:

The goal of this project is to build a text classification model that can classify text documents into predefined categories. In this case, the model is designed for a **spam detection task**, where the input is text, and the output is a binary label (1 for spam and 0 for not spam).

1. Dataset:

A simple dataset was used with 6 samples of text, with each text document labeled as either spam (1) or not spam (0). The dataset includes examples like promotional messages that are classified as spam and general messages classified as non-spam.

Example of Dataset:

Text	Label
"Win a free iPhone now!"	1 (Spam)
"Your order is confirmed"	0 (Not Spam)
"Congratulations, you've won!"	1 (Spam)
"Meeting at 5 PM"	0 (Not Spam)
"Free vacation offer!"	1 (Spam)
"Project deadline is tomorrow"	0 (Not Spam)

The dataset was small and consisted of only six messages, with an equal distribution of spam and non-spam messages. It was split into training and testing sets with a **80-20% split**, which left only one or two test samples.

2. Preprocessing:

Text preprocessing involved the following steps:

- **TF-IDF Vectorization**: The text data was converted into numerical vectors using **TF-IDF** (Term Frequency-Inverse Document Frequency) to represent the importance of words in the text. Stop words (like "the", "and", etc.) were removed to avoid noise in the model.

3. Model:

The classification model used in this project is the **Naive Bayes Classifier** (Multinomial Naive Bayes), which is particularly suited for text classification tasks where the input features are word counts or TF-IDF scores.

- **Multinomial Naive Bayes** is a popular algorithm for text-based tasks as it assumes that the features (words) are conditionally independent and that their occurrences follow a multinomial distribution.

4. Model Training:

The model was trained on the training portion of the dataset (80% of the total data). Since the dataset was very small, the model could quickly learn the relationship between the text features and the labels (spam or not spam).

5. Evaluation Metrics:

After training, the model was evaluated on the test set using several metrics:

- **Accuracy**: Proportion of correctly classified samples.
- **Precision**: Proportion of correctly predicted positive (spam) instances out of all predicted positives.
- **Recall**: Proportion of actual positives that were correctly predicted.
- **F1-Score**: The harmonic mean of precision and recall.

Evaluation Results:

From the results, we observed the following for the test set predictions:

Metric	Score
Accuracy	1.0000
Precision	1.0000
Recall	1.0000
F1-Score	1.0000

Given that the accuracy, precision, recall, and F1-score were all **1.00**, this indicates that the model classified all test samples correctly. The perfect scores are likely due to the simplicity and small size of the dataset.

---

6. Detailed Results Breakdown:

The classification report provides a more detailed breakdown:

	Precision	Recall	F1-Score	Support
Spam (1)	1.00	1.00	1.00	1

Not Spam (0)	1.00	1.00	1.00	1
Macro Avg	1.00	1.00	1.00	2
Weighted Avg	1.00	1.00	1.00	2

- **Precision of 1.00**: For both classes (spam and not spam), this indicates that every spam or non-spam label predicted by the model was correct.
- **Recall of 1.00**: For both classes, the model was able to retrieve all actual spam and non-spam samples correctly.
- **F1-Score of 1.00**: A perfect score, indicating that the model has a balanced and accurate prediction capability.
- **Support**: The support shows that there was only **1 sample per class** in the test set.

7. Model Save and Deployment:

- The trained Naive Bayes model was saved as `naive_bayes_text_classifier.pkl`.
- The TF-IDF vectorizer used to transform the text data was saved as `tfidf_vectorizer.pkl`.

These files can be reloaded and used to classify new, unseen text data in the future.

---

8. Limitations:

While the model achieved **100% accuracy** on this small test dataset, this result should be viewed with caution due to several limitations:

- **Small Dataset**: The dataset used was extremely small, with only 6 samples. In real-world applications, text classification tasks typically involve hundreds or thousands of samples.
- **Simple Data**: The data used here (spam detection with short phrases) was relatively simple. More complex datasets with longer text documents and more nuanced categories may not yield perfect results with Naive Bayes.
- **Class Imbalance**: With larger and more imbalanced datasets, precision and recall metrics could differ significantly, especially when one class (spam or not spam) dominates the dataset.

9. Conclusion:

- **Summary**: The Naive Bayes model performed perfectly on the test set for this small spam classification problem, achieving 100% accuracy, precision, recall, and F1-score.
- **Recommendation**: While the Naive Bayes classifier is effective for simple text classification tasks, its performance on larger and more complex datasets should be

evaluated, and other models (e.g., SVMs, neural networks) may be explored for comparison.

In a real-world setting, it is recommended to:

- Use a larger dataset.
- Test the model against more complex classification tasks.
- Employ cross-validation to better evaluate performance.