# System Architecture

The system is designed to classify spam and non-spam messages using a Naive Bayes classifier. Here's an overview of the system architecture:

1. Data Preprocessing:

- The system reads data from a CSV file containing message and category (spam or not spam) columns.

- It removes duplicate entries and updates the category labels to 'Spam' and 'Not Spam'.

2. Data Splitting:

- The dataset is split into training and testing sets with a 80:20 ratio using `train\_test\_split` from scikit-learn.

3. Feature Extraction:

- The text messages are converted into numerical features using the `CountVectorizer` from scikit-learn.

- The `CountVectorizer` converts text into a matrix where each row represents a message and each column represents a unique word in the corpus.

4. Model Training:

- A Multinomial Naive Bayes classifier is trained on the numerical features of the training data.

- The `MultinomialNB` model from scikit-learn is used for this purpose.

5. Model Evaluation:

- The trained model is evaluated on the testing dataset to compute the classification accuracy.

6. Prediction:

- The system provides a function `predict(message)` to classify new messages as spam or not spam.

- The input message is converted into numerical features using the same `CountVectorizer` used for training, and then passed to the trained Naive Bayes classifier for prediction.

# Algorithms Employed

- Multinomial Naive Bayes (MNB):

- Naive Bayes is a probabilistic classifier based on Bayes' theorem with strong independence assumptions between features.

- Multinomial Naive Bayes is suitable for text classification tasks where features are word counts or term frequencies.

- It calculates the probability of a message belonging to a particular category (spam or not spam) based on the frequency of words in the message.

# Usage Instructions

1. Setup:

- Install required packages: `pandas`, `scikit-learn`.

- Ensure that the `spam.csv` dataset is available in the same directory as the script or provide the correct path to the dataset.

2. Run the Script:

- Execute the provided Python script.

- The script will preprocess the data, train the Naive Bayes classifier, and print the classification accuracy on the test dataset.

3. Interpretation:

- Interpret the results based on the classification accuracy and predictions made by the model.

### References

<https://medium.com/@alinatabish/machine-learning-techniques-for-spam-detection-in-email-7db87eb11bc2>