# **Naive Bayes Spam Classifier Implementation**

## **Objective**

Implement a Naive Bayes classifier to detect spam messages using the bag-of-words approach. This exercise will help you understand the fundamentals of text classification and the Naive Bayes algorithm.

## **Dataset**

You will be provided with a CSV file named 'spam.csv' containing two columns:

* 'v1': Contains labels 'spam' or 'ham' (non-spam)
* 'v2': Contains the actual text message

## **Tasks**

1. **Data Loading and Preprocessing**
   * Implement a function to load the data from the CSV file.
   * Handle any potential issues with the CSV format (e.g., extra columns, missing data).
   * Split the data into training and testing sets (suggested split: 80% training, 20% testing).
2. **Bag-of-Words Vectorizer**
   * Implement a function to tokenize text (split into words, convert to lowercase).
   * Create a vocabulary from the training data.
   * Implement a function to convert text messages into bag-of-words vectors.
3. **Naive Bayes Training**
   * Calculate prior probabilities for spam and ham classes.
   * Calculate likelihood probabilities for each word in the vocabulary.
   * Use Laplace smoothing to handle words not seen in training.
4. **Naive Bayes Classification**
   * Implement a function to classify new messages using the trained model.
5. **Model Evaluation**
   * Test your classifier on the testing set.
   * Calculate and report the accuracy of your model.
6. **Prediction**
   * Demonstrate how to use your classifier to predict whether a new, unseen message is spam or ham.

## **Implementation Guidelines**

* Use only basic Python constructs and the standard library.
* Avoid using external machine learning libraries (e.g., scikit-learn).