# Machine Learning Model

## Project Goal and Scope

Spam emails pose significant challenges to digital communication by cluttering inboxes, wasting user time, and sometimes posing security threats through phishing or malicious links. To address this issue, we are developing a machine learning model that detects and classifies email messages into spam or not spam by building a robust classification system that distinguishes between valid and spam messages. The final goal of the project is to implement a user-facing interface where the trained model can be used in real-time to classify new messages.

## Data

### Explaining the data structure

The dataset used in this project comprises 5,572 labeled textual messages organized in a tabular format with two key columns: class, which denotes the label of the message (spam or valid), and message, which contains the actual textual content of each email or SMS.

### Describing the preprocessing steps for text cleaning

#### Lowering case

Lowercasing is a fundamental preprocessing step in text analysis. It involves converting all characters into a message to lowercase to ensure uniformity, reduce redundancy and eliminate case sensitivity from the model's perspective. For example, the words “Free” and “free” would be treated as two different tokens if not normalized.

#### Removal of special characters

Special characters such as punctuation marks, symbols (e.g., “#”, “@”, “$”), and other non-alphanumeric characters often do not contribute meaningful information for classification tasks unless contextually important. In this project, special characters are removed to reduce noise in the data and to standardize the input for feature extraction.

#### Removal of stop words (i.e., commonly occurring words in a language like ‘the’, ‘a’, and so on)

Stop words are commonly used words in a language such as “the”, “is”, “in”, “and”, which generally do not add much meaning when analyzing text for classification. Hence, these words are removed to help reduce dimensionality, eliminate noise and improve model performance by focusing on more meaningful terms that better distinguish between spam and valid messages.

#### Removal of hyperlinks

Hyperlinks in messages often point to external websites and are a common feature in spam content. While they may indicate promotional or malicious intent, in basic text classification tasks, raw hyperlinks can add noise due to their unique and lengthy structure.

#### Removal of numbers

While numbers may sometimes carry meaning (e.g., prices or dates), they are often used in spam to attract attention or mislead users, however, numbers in text messages can introduce unnecessary complexity, especially in traditional machine learning models that do not interpret numeric values within unstructured text. Thus, in the context of this project, numbers are removed to simplify the dataset and to focus on the semantic content conveyed through words.

#### Removal of whitespaces

Unnecessary whitespaces can disrupt tokenization and skew feature extraction by treating visually similar terms as different. Removing extra whitespaces ensures consistent formatting of the message text and allows the model to better interpret the structure of the data. This step includes trimming leading/trailing spaces and reducing multiple spaces between words to a single space.

### Describe the text transformation

Once the text has been cleaned through preprocessing, it must be transformed into a structured format that can be used by machine learning algorithms. Specifically, each message is transformed into a set of variables that capture its structural and linguistic characteristics. These include the total number of characters, number of words, presence of common spam keywords (such as “free”, “win”, or “urgent”), count of special characters like exclamation marks, and message length. These numeric features effectively represent the messages in a format compatible with standard classifiers, enabling the model to learn patterns associated with spam and valid messages.

## Analysis and Modeling

### Data Splitting: training and validation

To evaluate the performance of machine learning models effectively, the dataset must be divided into separate subsets: a training set and a validation set. The training set is used to teach the model to recognize patterns and relationships in the data, while the validation set is used to test how well the model generalizes to unseen data. This process prevents overfitting and provides an unbiased estimate of model accuracy. In this project, the dataset is split 75% of the data is used for training and the remaining 25% for validation.

### Train two classification models (KNN, DT, and NB)

In this phase of the project, three classical machine learning models were trained to classify messages as either spam or valid using three models: K-Nearest Neighbors (KNN), Decision Tree (DT), and Naive Bayes (NB). Each model was trained on the structured features extracted from the preprocessed text data using the training subset created during the data splitting phase. By training these models, we aim to compare their predictive performance and identify the most accurate and reliable approach for detecting spam messages.

### Evaluate the three models and include a confusion matrix for each

Each model was trained on the same preprocessed text features derived using TF-IDF vectorization. Performance metrics including precision, recall, F1-score, and accuracy were used to evaluate the models. The Naive Bayes classifier achieved the highest accuracy at 96%, with strong performance in detecting both spam and valid messages, particularly a perfect spam precision score of 1.00. The Decision Tree classifier followed closely with a 95% accuracy, offering a slightly more balanced recall across both classes. KNN, while achieving 91% accuracy, struggled to identify spam messages effectively, with a notably low spam recall of 0.36. Confusion matrices were generated for each model to visually illustrate classification performance. Based on the combined evaluation metrics, the Naive Bayes model was selected as the most effective and reliable classifier for deployment in the spam detection interface (See figures in the appendix).

## Interface Real Demonstration

### Develop a simple interface for deploying your solution

A simple interface was developed using Streamlit. The interface allows users to input any message through a text area and classify it instantly as either “spam” or “valid” based on the trained Naive Bayes model. The system preprocesses the input message using the same steps applied during model training including lowercasing, removal of punctuation, numbers, hyperlinks, and stopwords, to ensuring consistent feature representation. Once processed, the message is transformed using the trained TF-IDF vectorizer and passed to the classifier to predict the category.

### Give two examples of you writing a message and how would the ML technique would classify each through the interface.

To validate the effectiveness of the deployed spam detection interface, two representative message examples were tested through the Streamlit application. The first example was a typical spam message: *“Congratulations! You’ve won a free iPhone. Click here to claim your prize now: www.fakeprize.com”*. When entered into the interface, the system accurately classified the message as spam. The second example tested was a valid communication: *“Hey, I’ll be a bit late to the meeting today. Can we reschedule it for 4 PM instead?”*. This message was correctly identified as valid, reflecting the model's ability to recognize casual, contextually appropriate, and non-commercial language (See Appendix) You can access it following this link: https://advbaclassproject-ykyjp5jduygxrqbytfps8u.streamlit.app/.