In this assignment, you will build a spam classifier from scratch. No training data will be provided. You are free to use whatever training data that is publicly available/does not have any copyright restrictions (You can build your own training data as well if you think that is useful). You are free to extract features as you think will be appropriate for this problem. The final code you submit should have a function/procedure which when invoked will be able to automatically read a set a emails from a folder titled test in the current directory. Each file in this folder will be a test email and will be named 'email#.txt' ('email1.txt', 'email2.txt', etc). For each of these emails, the classifier should predict +1 (spam) or 0 (non Spam). You are free to use whichever algorithm learnt in the course to build a classifier (or even use more than one). The algorithms (except SVM) need to be coded from scratch. Your report should clearly detail information relating to the data-set chosen, the features extracted and the exact algorithm/procedure used for training including hyperparameter tuning/kernel selection if any. The performance of the algorithm will be based on the accuracy on the test set.

Solution:

1. Naive Bayes

- Took dataset from kaggle, which has around 5000 unique mails and its label.
- Divided the dataset into train data (80%) and test data (20%).
- For each email of train data, first removed all the special characters (replaced it with an empty string).
- Then count the frequency of each word throughout the dataset.
- Then chose the **top k most frequent words** (tried with different values of k).
- Then for each email, check which of these words are present in each email.
- Then calculated, for each word what is the **probability that it belongs to spam email and non spam email.**
- Then for each email of test data, first remove all the special characters, then split it into words.
- For each word, check if it is **present** in the **dictionary**, if not then ignore it.

- If it is present, then use it in the formula of probability that email is spam or not.
- **If the word seems to belong to spam email**, then multiply spam probability by 0.95 and not spam probability by 0.01. So that the probability of email spam increases slightly as compared to not spam probability.
- Then compare the probability that it belongs to spam or not spam, whichever is more, predict that.
- **Accuracy** on the test dataset (20% of total emails), is around **85%.**

2.SVM

- Here we will consider discriminative classification (instead of generative classification which we used in Bayesian classification).
- In SVM, we simply try to find a line or plane which divides classes from each other.
- For SVM, I used the inbuilt library of sklearn.
- Did not use any kernel (tried to find a linear line or plane that separates the data into two classes).
- **Accuracy** of SVM on test data is around **96%.**

Accuracy of Naive Bayes is around 85%, whereas accuracy of SVM is around 96%.
SVM is performing better on test data.