Spam or Ham with NLP.

Abhinav Rawat

Table of contents

Chapter 1

1. Introduction: Natural Language Processing (NLP) is an interdisciplinary subfield of computer science and artificial intelligence. It is primarily concerned with providing the ability to process data encoded in natural language and is thus closely related to information retrieval, knowledge representation and linguistics. NLP has its roots in the 1940s, the article titled 'Computing Machinery and Intelligence' which proposed what is now called the 'Turing test' as a criterion of intelligence. The proposed test includes a task that involves the automated interpretation and generation of natural language. In this project of Spam or Ham detection we focus on the detection of spam and ham (legitimate) emails using a systematic approach that includes Exploratory Data Analysis (EDA), data cleaning techniques, text tokenization, lemmatization and implementing logistic regression model.

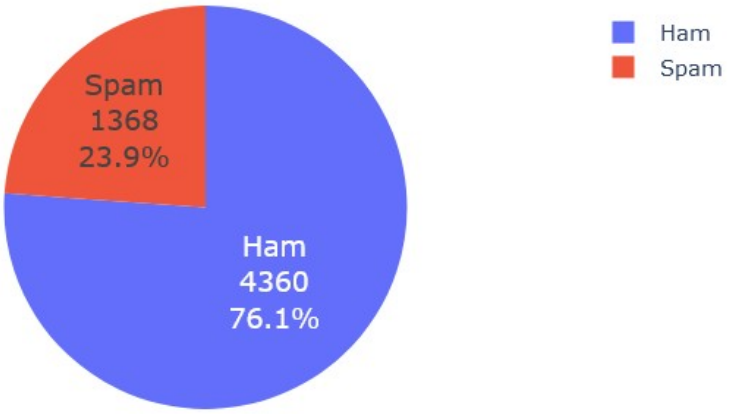
1.1 Overview: The Spam-Ham detection project is a machine learning initiative designed to enhance email communication security by automatically categorizing incoming messages as either 'Spam' or 'Ham'. Leveraging advanced NLP techniques and machine learning algorithms, this project aims to accurately identify and filter out unwanted and potentially harmful emails, thus safeguarding users from phishing attempts, scams and unsolicited content. Using a systematic approach that includes EDA, data cleaning techniques, text tokenization, lemmatization and using logistic regression model. The process involves thorough analysis, cleaning and processing of the dataset, followed by the development and training of an logistic regression classification model.

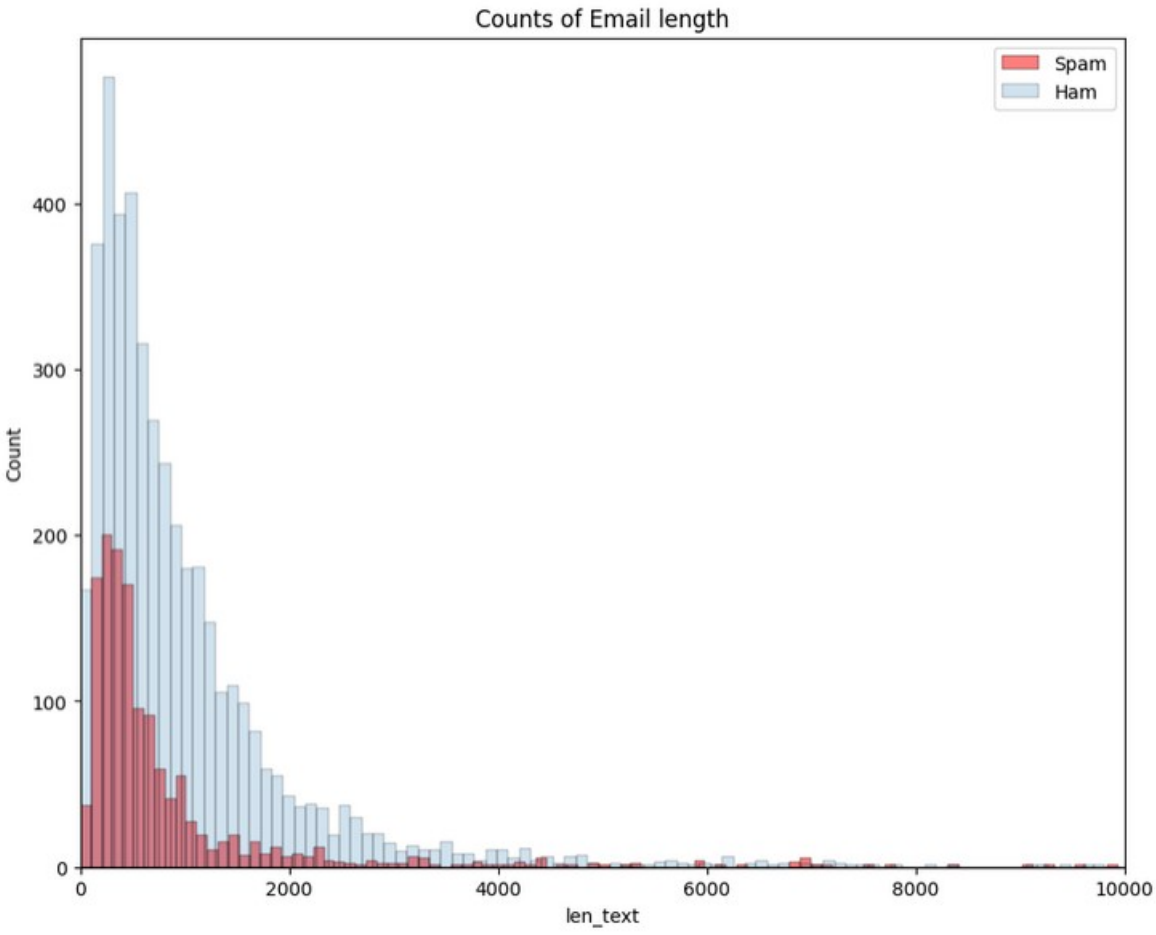
1.2 Objective of the Study: This project aims to filter and classify emails accurately into Spam or Ham (legitimate) categories.

Chapter 2

2. Methodology: The dataset consists of a collection of emails labelled as either spam or ham (legitimate). Each email sample is represented by features such as sender email address, subject line, email body content and attachments. The dataset provided a balanced distribution of spam and ham emails, enabling the development of a model that can effectively differentiate between the two classes. All the work will be done using jupyter-lab in a python kernel. The data is loaded into jupyter environment, understanding the data and handling missing values or duplicates. Exploratory Data Analysis (EDA) was performed to visualize the data statistics, Text preprocessing or text cleaning was performed, Feature engineering like numerical features from text data, Term Frequency-Inverse Document Frequency (TF-IDF) or bag of words was performed, Vectorize the text data using chosen feature representation technique. Split the data into train and test, build a model to classify the data, finally evaluate the final model on the test data.

2.1 Data Preparation: The data consists of 5728 entries of emails labelled as spam or ham, majority of data is labelled ham (4360). We label the data with 1s and 0s for Spam and Ham respectively, convert all data to lower case, removing non alphabet characters, URL's, HTML tags and Emoticons since they do not contribute in capturing the natural language. Stop words are also removed, along with punctuations since they do not contribute as features for NLP. Afterwards we tokenize the cleaned text which will save the words as lists, after tokenization most frequent words are removed to eliminate bias in features. Visualize the distribution of spam and ham emails using word clouds and pie charts as seen below.


Ham
Spam
Spam
1368
23.9%
Ham
4360
76.1%

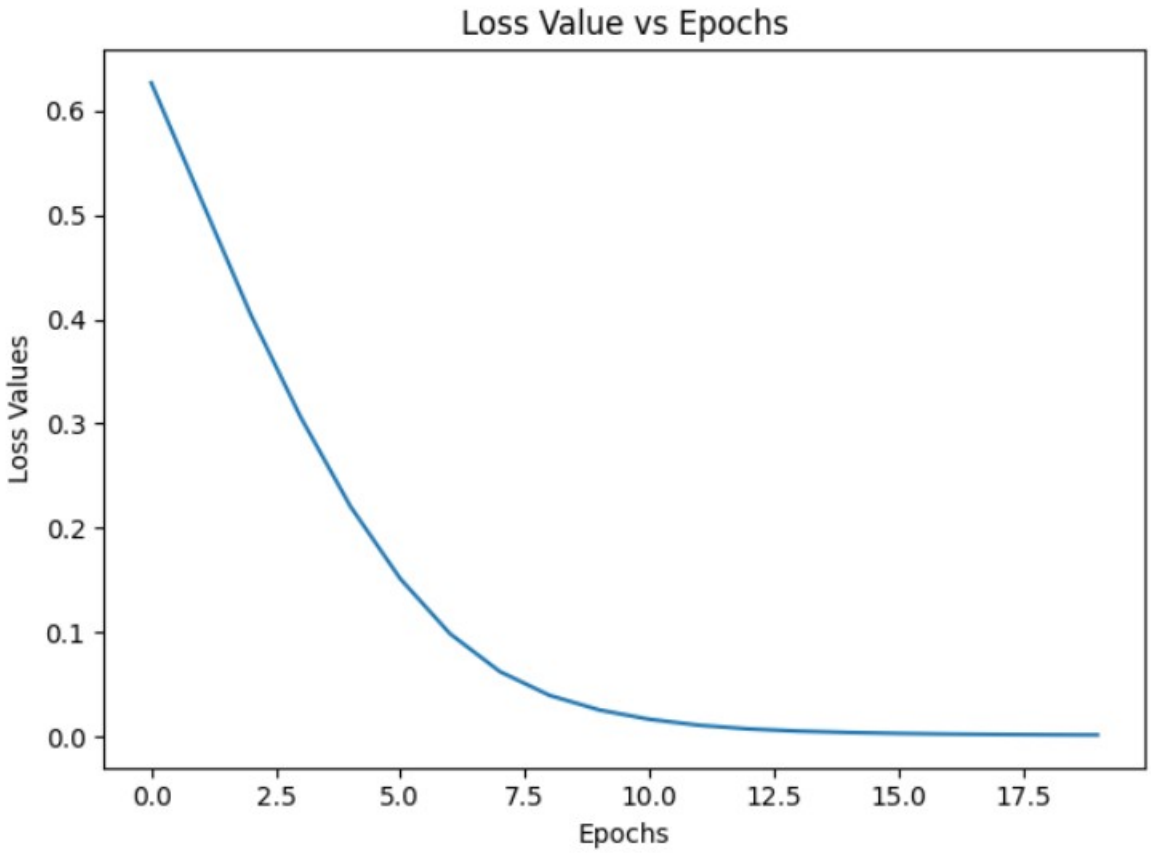

Counts of Email length
Spam
Ham
Count
0
100
200
300
400
0
2000
4000
6000
8000
10000
len_text

2.2 Model and Training: The Data is converted into a sparse matrix where each token is treated as a feature/column and if present, its indicated by a '1' and if absent its represented by '0'. We limited the word count to 10,000 since the amount of data is limited to around 6k emails. The data is split into train and test sets, where around 4600 samples are in train data and rest in the test data. A simple Logistic Regression model is build with Pytorch, with 10000 input layers and 100×10 hidden layers with 2 output layers. A learning rate of 0.01 is implemented with optimizer Adam, since its a classification model we used a basic Cross-Entropy loss function, the model is trained for 20 epochs, meaning the entire data is passed through the model 20 times.

2.3 Results: After running the model for 20 epochs we can see that the model is predicting with an accuracy of 98-99%.

```
Epoch: 0 | train Loss: 0.627 | Val Accuracy: 74.695%
Epoch: 1 | train Loss: 0.515 | Val Accuracy: 75.916%
Epoch: 2 | train Loss: 0.404 | Val Accuracy: 81.501%
Epoch: 3 | train Loss: 0.306 | Val Accuracy: 86.300%
Epoch: 4 | train Loss: 0.221 | Val Accuracy: 90.314%
Epoch: 5 | train Loss: 0.151 | Val Accuracy: 95.026%
Epoch: 6 | train Loss: 0.098 | Val Accuracy: 97.120%
Epoch: 7 | train Loss: 0.062 | Val Accuracy: 98.168%
Epoch: 8 | train Loss: 0.039 | Val Accuracy: 98.255%
Epoch: 9 | train Loss: 0.025 | Val Accuracy: 98.604%
Epoch: 10 | train Loss: 0.016 | Val Accuracy: 98.778%
Epoch: 11 | train Loss: 0.011 | Val Accuracy: 98.953%
Epoch: 12 | train Loss: 0.007 | Val Accuracy: 99.040%
Epoch: 13 | train Loss: 0.005 | Val Accuracy: 98.953%
Epoch: 14 | train Loss: 0.004 | Val Accuracy: 99.040%
Epoch: 15 | train Loss: 0.003 | Val Accuracy: 98.953%
Epoch: 16 | train Loss: 0.002 | Val Accuracy: 98.953%
Epoch: 17 | train Loss: 0.002 | Val Accuracy: 98.953%
Epoch: 18 | train Loss: 0.002 | Val Accuracy: 98.953%
Epoch: 19 | train Loss: 0.001 | Val Accuracy: 98.953%
```

At epoch 12 we can see that the model peaks to 99% and then stays around that accuracy for rest of the epochs, if we observe the loss it pretty much reaches zero after epoch 12. Looking at this we can even terminate the training after epoch 12, since much improvement is not left to make.



Chapter 3

3. Discussion and Conclusion: From the results we can conclude that a simple logistic regression model works wonder with the identifying spam emails in the given data, with predictions of 99% after 12 epochs. The total time to run the model was less than 30 seconds, this was mainly because we feed the entire data into the model at once and not feed it in batches.