|  |
| --- |
|  |
| CLASSIFICATION OF SPAM MESSAGES USING MACHINE LEARNING TECHNIQUES |
|  |
|  |

BY V.AKASH, 2020103001, Batch P

|  |
| --- |
|  |

**ABSTRACT**

Electronic mail has eased communication methods for many organisations as well as individuals. This method is exploited for fraudulent gain by spammers through sending unsolicited emails. This project aims to present a method for detection of spam emails with machine learning algorithms. A literature review is carried to explore the efficient methods applied on different datasets to achieve good results.

An extensive research was done to implement machine learning models using Naïve Bayes, Support Vector Machine, Random Forest, Decision Tree and Multi-Layer Perceptron, along with feature extraction and pre-processing. Multinomial Naïve Bayes with Genetic Algorithm performed the best overall. The comparison of our results with other machine learning and bio-inspired models to show the best suitable model is also discussed.

**INTRODUCTION**

Machine learning models have been utilized for multiple purposes in the field of computer science from resolving a network traffic issue to detecting a malware. Emails are used regularly by many people for communication and for socialising. Security breaches that compromises customer data allows ‘spammers’ to spoof a compromised email address to send illegitimate (spam) emails. This is also exploited to gain unauthorized access to their device by tricking the user into clicking the spam link within the spam email, that constitutes a phishing attack.

Many tools and techniques are offered by companies in order to detect spam emails in a network. Organisations have set up filtering mechanisms to detect unsolicited emails by setting up rules and configuring the firewall settings. Google is one of the top companies that offers 99.9% success in detecting such emails.

There are different areas for deploying the spam filters such as on the gateway (router), on the cloud hosted applications or on the user’s computer. In order to overcome the detection problem of spam emails, methods such as content-based filtering, rule-based filtering or Bayesian filtering have been applied.

Unlike the ‘knowledge engineering’ where spam detection rules are set up and are in constant need of manual updating thus consuming time and resources, Machine learning makes it easier because it learns to recognise the unsolicited emails (spam) and legitimate emails (ham) automatically and then applies those learned instructions to unknown incoming emails.

The proposed spam detection to resolve the issue of the spam classification problem can be further experimented by feature selection or automated parameter selection for the models.

**PROBLEM STATEMENT:**

Today, spamming mails is one of the biggest issues faced by everyone in the world of the Internet. In such a world, email is mostly shared by everyone to share the information and files because of their easy way of communication and for their low cost. But such emails are mostly affecting the professionals as well as individuals by the way of sending spam emails. Every day, the rate of spam emails and spam messages is increasing. Such spam emails are mostly sent by people to earn income or for any advertisement for their benefit. This increasing amount of spam mail causes traffic congestion and waste of time for those who are receiving that spam mail. The real cost of spam emails is very much higher than one can imagine.

Sometimes, the spam emails also have some links which have malware. And also, some people will get irritated once they see their inbox which is having more spam mails. Sometimes, the users easily get trapped into financial fraud actions, by seeing the spam mails such as job alert mails and commercial mails and offer emails. It may also cause the person to have some mental stress. To reduce all these risks, the system has proposed a machine learning model which will detect spam mail and non-spam emails, and also this system will optimize the data by removing the unwanted mails which contain the advertisement mails and also some useless emails and also some fraud mails. This proposed system will detect the spam mails and ham emails with the dataset consisting of spam mails and after identifying spam mails this system will remove that spam emails and this proposed system will calculate the amount of storage before and after the removal of spam mails.

**OVERALL OBJECTIVE:**

The main objective of the project is to detect the spam mails and to optimize the data storage. We first import the emails, and count the number of data points available, frequencies and repetitions. Then, we remove punctuations and stop words. Next, we convert the sanitized data into vectors that can be fed into various classifiers. We then split the data into training set and test set, and find the accuracy of our model.

**RELATED WORK**

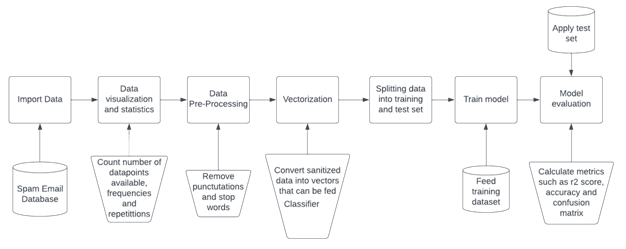
Researchers have taken a lead to implement machine learning models to detect spam emails. In the paper [3], the authors have conducted experiments with six different machine learning algorithms: Naïve Bayes (NB) classification, K-Nearest Neighbour (K-NN), Artificial Neural Network (ANN), Support Vector Machine (SVM), Artificial Immune System and Rough Sets. Their aim of the experiment was to imitate the detecting and recognising ability of humans. Tokenisation was explored and the concept provided two stages: Training and Filtering. Their algorithm consisted of four steps: Email Pre-Processing, Description of the feature, Spam Classification and Performance Evaluation. It concluded that the Naïve Bayes provided the highest accuracy, precision and recall.

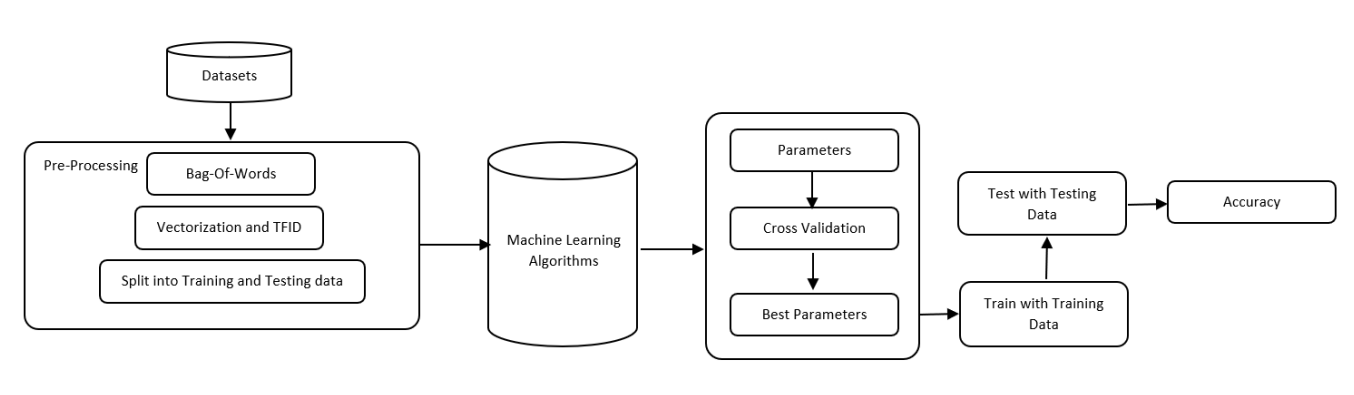
Feng et al. [1] describes a hybrid system between two machine learning algorithms i.e. SVM-NB. Their proposed method is to apply the SVM algorithm and generate the hyperplane between the given dimensions and reduce the training set by eliminating datapoints. This set will then be implemented with NB algorithm to predict the probability of the outcome. This experiment was conducted on Chinese text corpus. They successfully implemented their proposed algorithm and there was an increase in accuracy when compared to NB and SVM on their own.

Mohammed et al. [4] aimed to detect the unsolicited emails by experimenting with different classifiers such as: NB, SVM, KNN, Tree and Rule based algorithms. They generated a vocabulary of Spam and Ham emails which is then used to filter through the training and testing data. Their experiment was conducted with Python programming language on Email-1431 dataset. They concluded that NB was the best working classifier followed by Support Vector Machine.

Wijaya and Bisri [5] proposes a hybrid-based algorithm, which is integrating Decision Tree with Logistic Regression along with False Negative threshold. They were successful in increasing the performance of DT. The results were compared with the prior research. The experiment was conducted on the SpamBase dataset. The proposed method presented a 91.67% accuracy.

**BLOCK DIAGRAM:**





**TOOLS**

1. Jupyter-Notebook

The *Jupyter Notebook App* is a server-client application that allows editing and running [notebook documents](https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html#notebook-document) via a web browser. The *Jupyter Notebook App* can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet.

In addition to displaying/editing/running notebook documents, the *Jupyter Notebook App* has a “Dashboard” ([Notebook Dashboard](https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html#dashboard)), a “control panel” showing local files and allowing to open notebook documents or shutting down their [kernels](https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html#kernel)

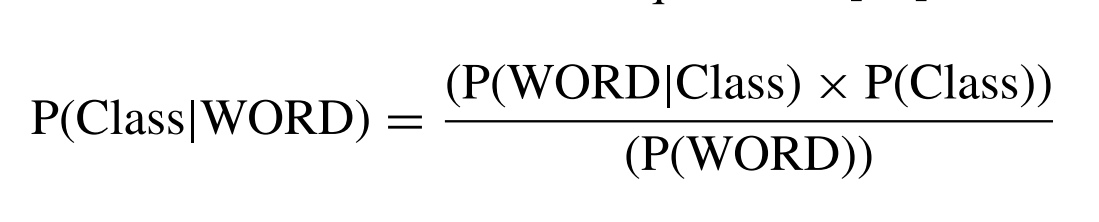
2. SCIKIT-LEARN

Scikit-Learn (SKLearn) is an environment that is incorporated with Python programming language. The library offers a wide range of supervised algorithms that will be suitable for this project. The library offers high-level implementation to train with the ’Fit’ methods and ’predict’ from an estimator (Classifier). It also offers to perform the cross validation, feature selection, feature extraction and parameter tuning.

**MODELS USED**

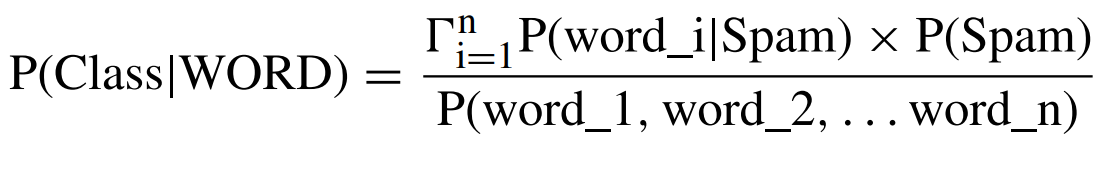
**1. Naive Bayes – Gaussian, Multinomial and Bernoulli**

Naïve Bayes model is used to resolve classification problems by using probability techniques.



where WORD is (word1,word2, . . .wordn) from within an uploaded email and ‘Class’ is either ‘Spam’ or ‘Ham’. The algorithm calculates the probability of a class from the bag of words provided by the program. Where P(Class | WORD) is a posterior probability, P(WORD | Class) is likelihood and P(Class) is the prior probability.

If ‘Class’ = Spam, the equation could be rewritten to find the spam email from the given words, and this can be further simplified as:

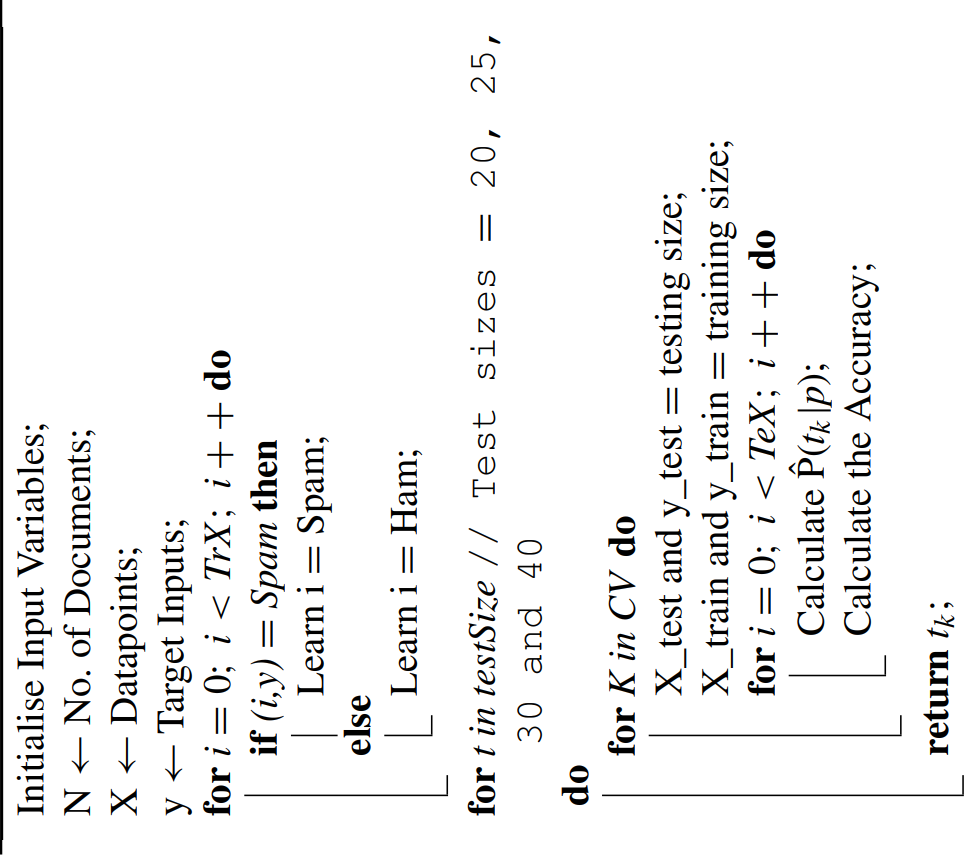


There are three types of Naïve Bayes algorithms: Multinomial, Gaussian and Bernoulli.

Multinomial Naïve Bayes considers a feature vector where a given term represents the number of times it appears or very often i.e. frequency.

 Bernoulli Naïve Bayes is a binary algorithm used when the feature is present or not.

Gaussian Naïve Bayes is based on continuous distribution.



**2. SUPPORT VECTOR MACHINE (SVM)**

Support vector machines are a set of supervised learning methods used for classification, regression, and outliers detection. All of these are common tasks in machine learning.

You can use them to detect cancerous cells based on millions of images or you can use them to predict future driving routes with a well-fitted regression model.There are specific types of SVMs you can use for particular machine learning problems, like support vector regression (SVR) which is an extension of support vector classification (SVC). The main thing to keep in mind here is that these are just math equations tuned to give you the most accurate answer possible as quickly as possible.

SVMs are different from other classification algorithms because of the way they choose the decision boundary that maximizes the distance from the nearest data points of all the classes. The decision boundary created by SVMs is called the maximum margin classifier or the maximum margin hyper plane.

There are two different types of SVMs, each used for different things:

Simple SVM: Typically used for linear regression and classification problems.

Kernel SVM: Has more flexibility for non-linear data because you can add more features to fit a hyperplane instead of a two-dimensional space.

## Kernel functions

## Linear

These are commonly recommended for text classification because most of these types of classification problems are linearly separable.

The linear kernel works really well when there are a lot of features, and text classification problems have a lot of features. Linear kernel functions are faster than most of the others and you have fewer parameters to optimize.

Here's the function that defines the linear kernel:

**f(X) = w^T \* X + b**

In this equation, w is the weight vector that you want to minimize, X is the data that you're trying to classify, and b is the linear coefficient estimated from the training data. This equation defines the decision boundary that the SVM returns.

## Polynomial

The polynomial kernel isn't used in practice very often because it isn't as computationally efficient as other kernels and its predictions aren't as accurate.

Here's the function for a polynomial kernel:

**f(X1, X2) = (a + X1^T \* X2) ^ b**

This is one of the more simple polynomial kernel equations you can use. **f(X1, X2)** represents the polynomial decision boundary that will separate your data. **X1** and **X2** represent your data.

## Gaussian Radial Basis Function (RBF)

One of the most powerful and commonly used kernels in SVMs. Usually the choice for non-linear data.

Here's the equation for an RBF kernel:

**f(X1, X2) = exp(-gamma \* ||X1 - X2||^2)**

In this equation, gamma specifies how much a single training point has on the other data points around it. **||X1 - X2||** is the dot product between your features.

## Sigmoid

More useful in neural networks than in support vector machines, but there are occasional specific use cases.

Here's the function for a sigmoid kernel:

**f(X, y) = tanh(alpha \* X^T \* y + C)**

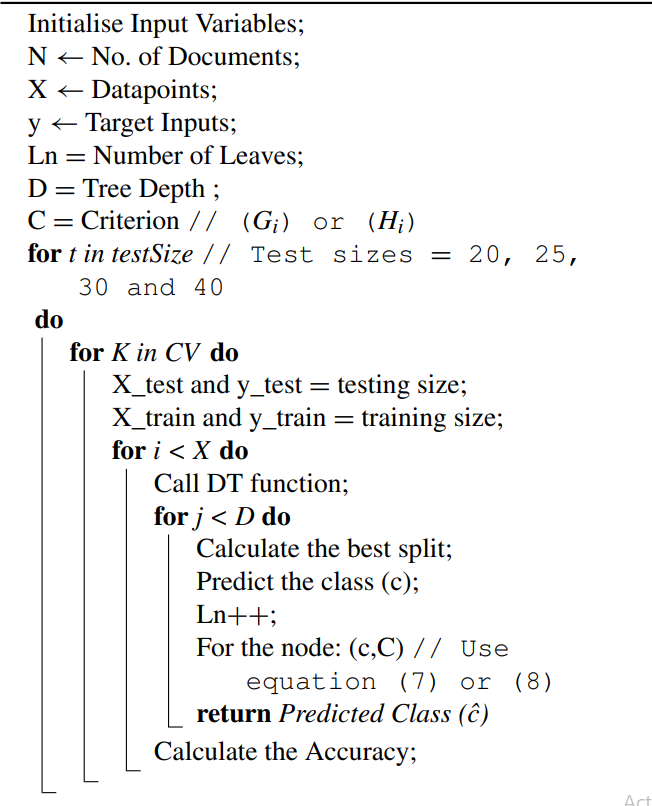
In this function, **alpha** is a weight vector and **C** is an offset value to account for some mis-classification of data that can happen.

**3. DECISION TREES**

A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning.

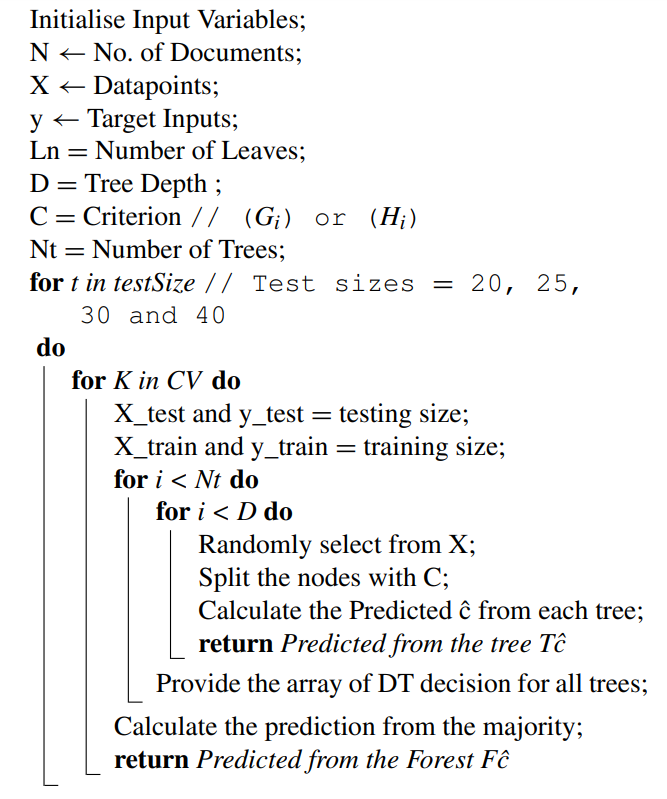
This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret. Decision Tree is a white box type of ML algorithm. It shares internal decision-making logic, which is not available in the black box type of algorithms such as Neural Network.

Its training time is faster compared to the neural network algorithm. The time complexity of decision trees is a function of the number of records and number of attributes in the given data. The decision tree is a distribution-free or non-parametric method, which does not depend upon probability distribution assumptions. Decision trees can handle high dimensional data with good accuracy.



**4. RANDOM FOREST**

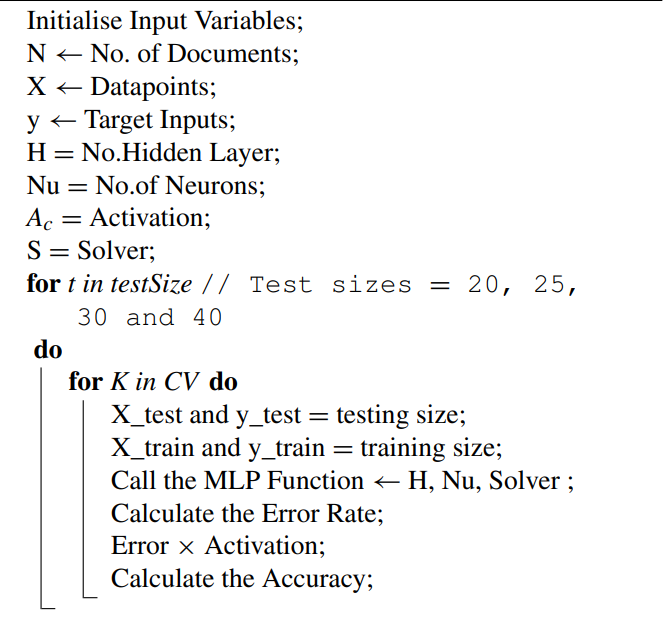
Random Forest (RF) algorithm can be used for both classification and regression. The algorithm predicts the classes by using multiple decision tree, where each tree predicts the classification class. This is evaluated by the RF model to select the high number of predicted class as an assigned prediction. The algorithm-4 explains the workings of the Random Forest classifier with the Spam Email dataset, where Fc is the outcome predicted from the entire forest. Equation-7 and equation-8 are also utilised to calculate the Gini and Entropy for Random Forest (RF) algorithm to calculate the Criterion. This module was loaded from Scikit-learn library and it is based on the depth of the tree and number of DT to be produced. These are usually considered as the termination criteria. This means the more the depth and the number of trees the more the computational time required for the algorithm.



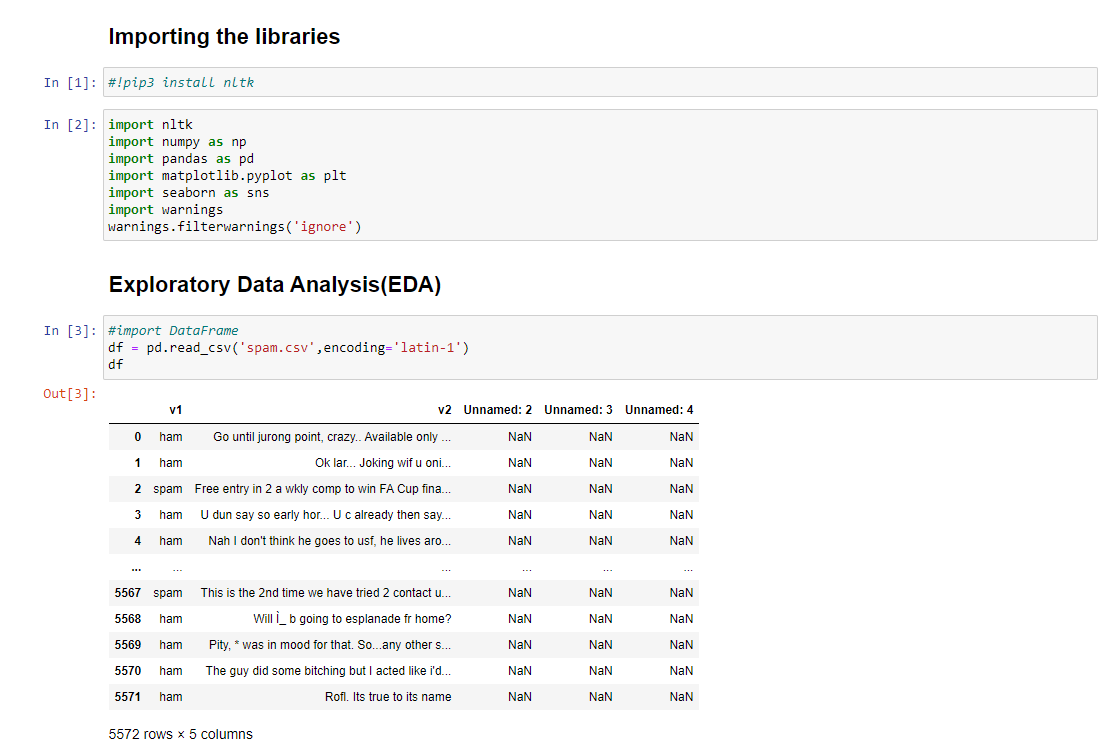
**4. MULTILAYER PERCEPTRON**

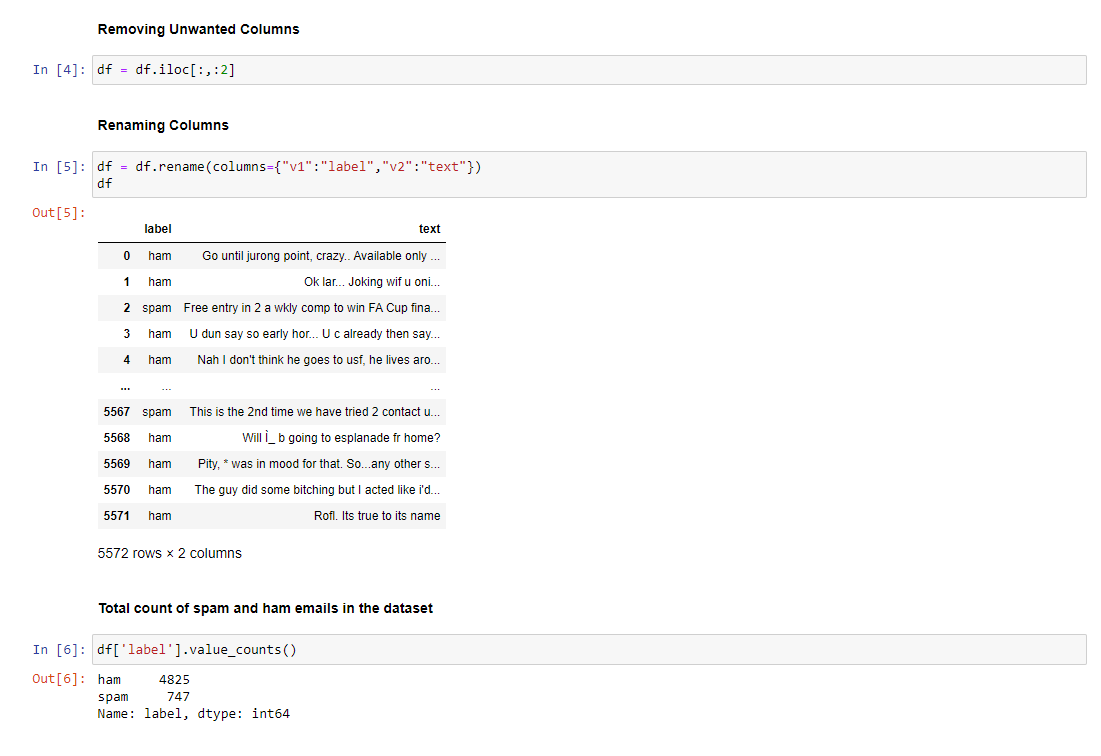
The MLP is a feed-forward Artificial Neural Network (ANN). It is a supervised method which includes non-linear hidden layers between the input and the output layer.

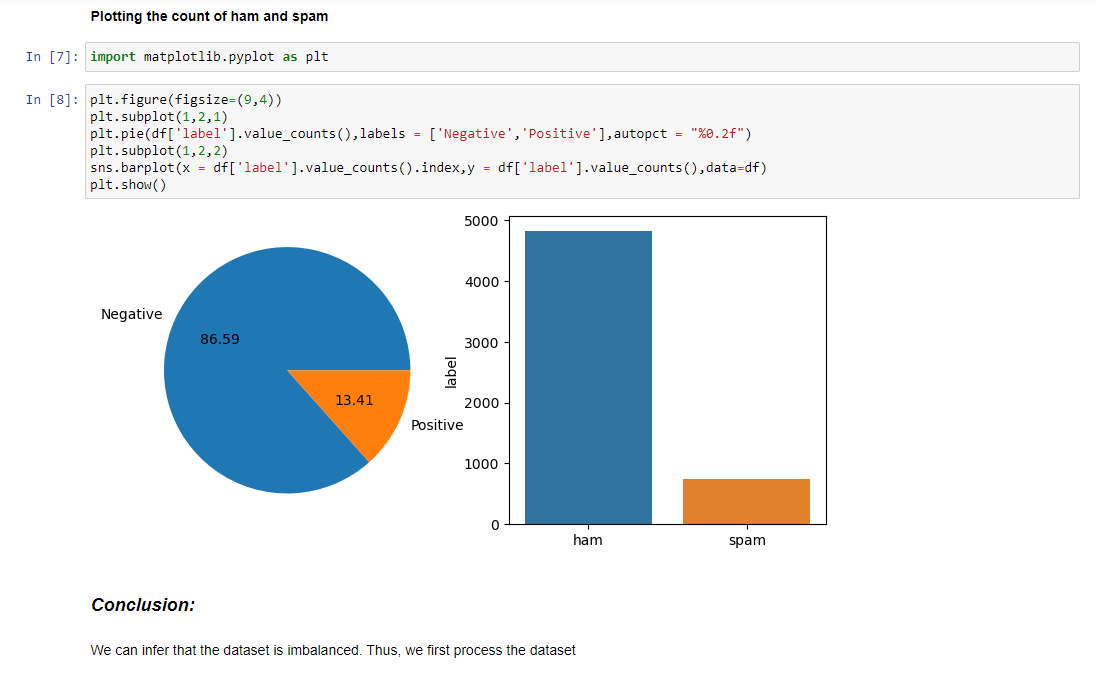
The algorithm can have one or more layers between input and output layer known as ‘Hidden Layer(s)’. The hidden layer accepts the values from the previous layer and transforms with linear summation, whereas the ‘Output’ layer provides the output values after transformation from the previous hidden layer.

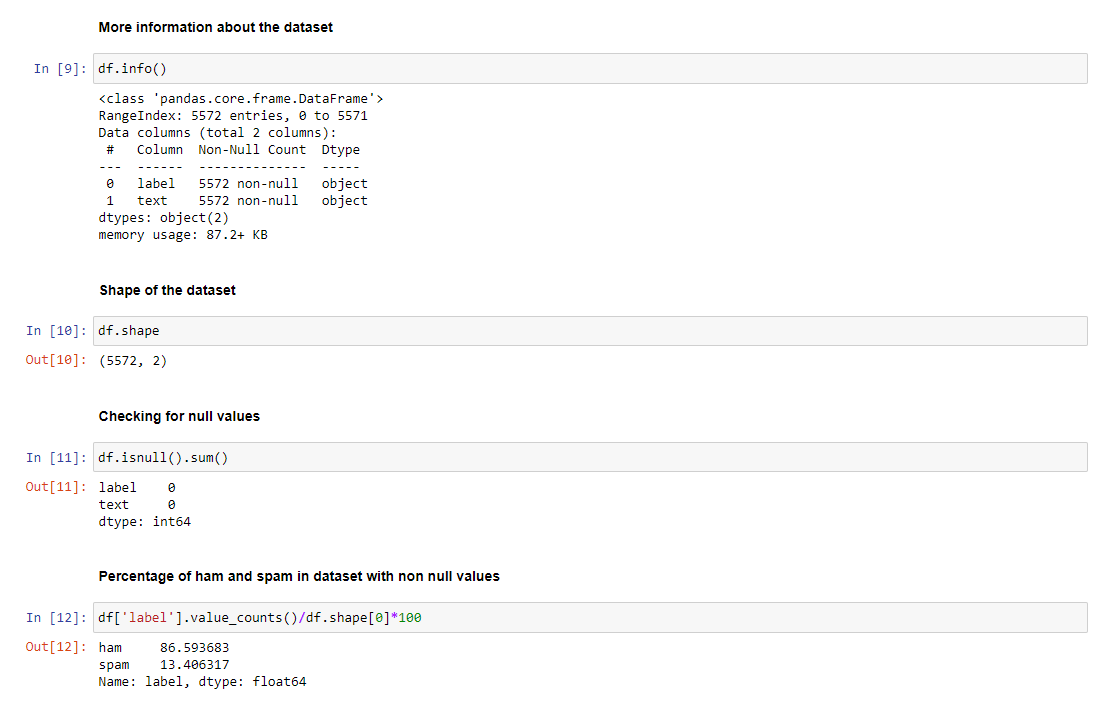


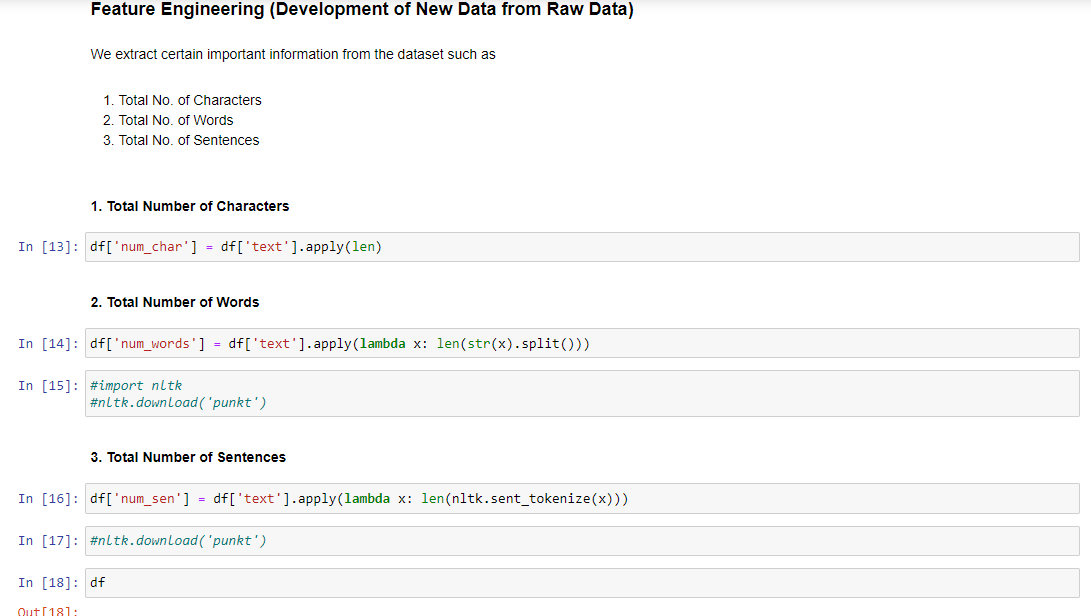
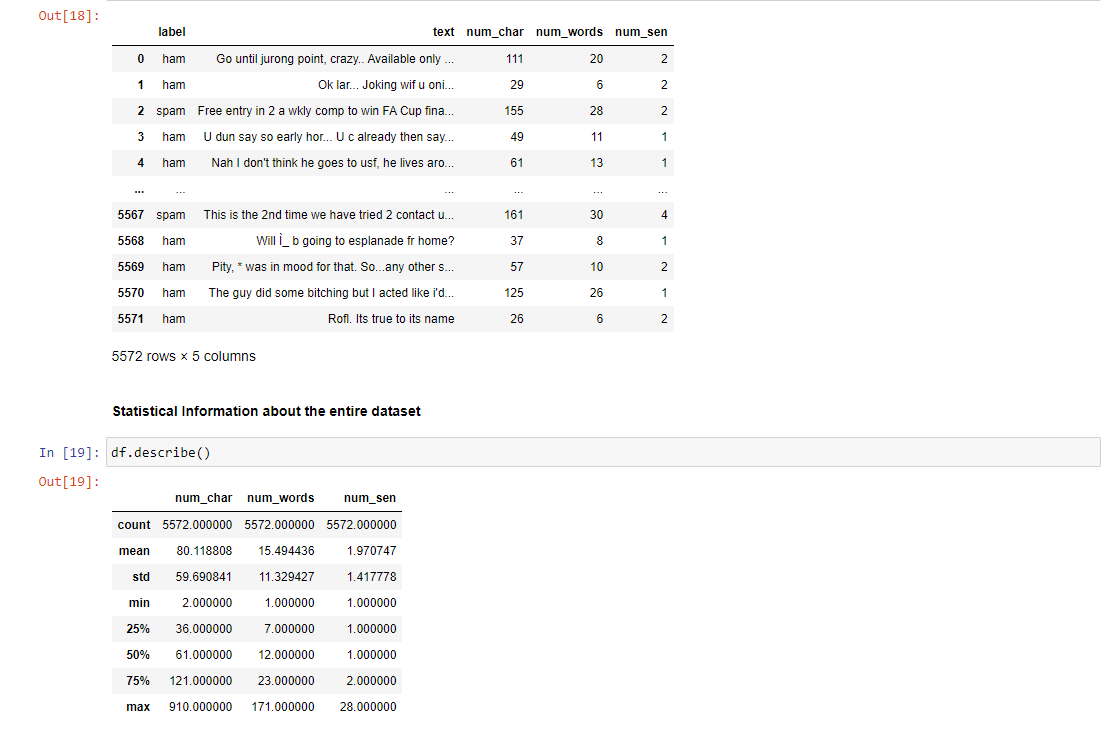
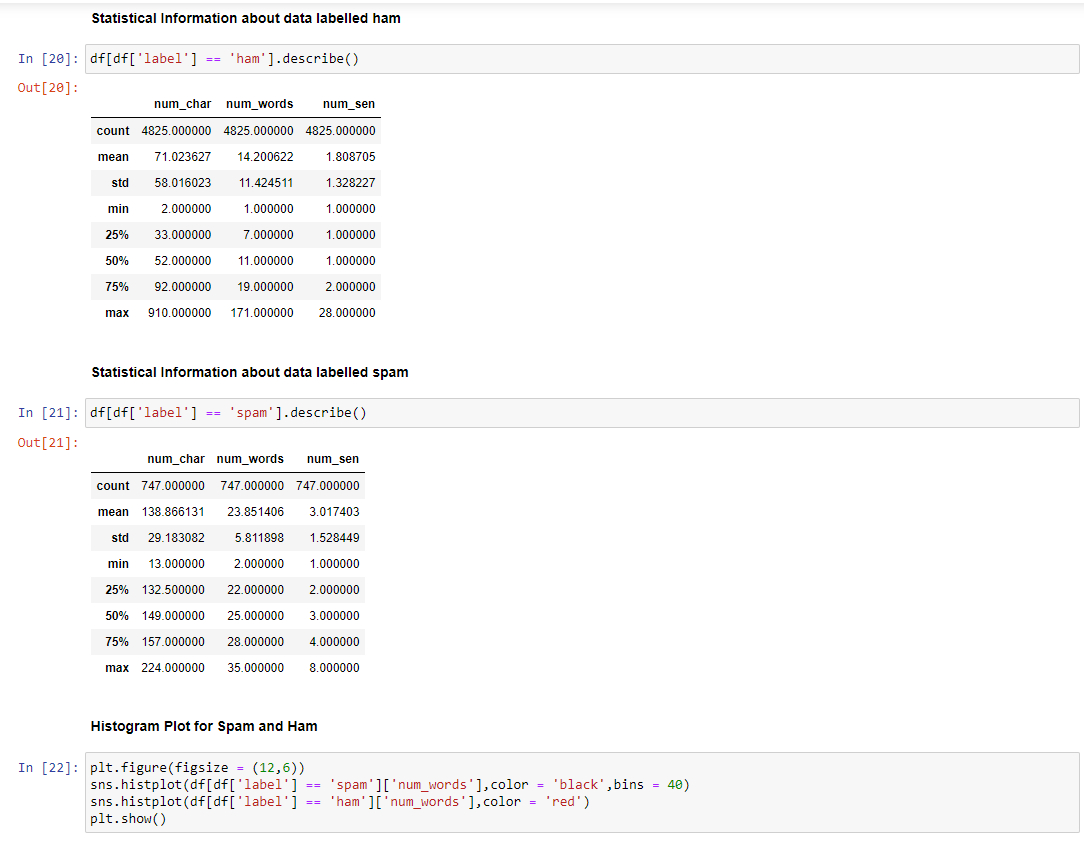
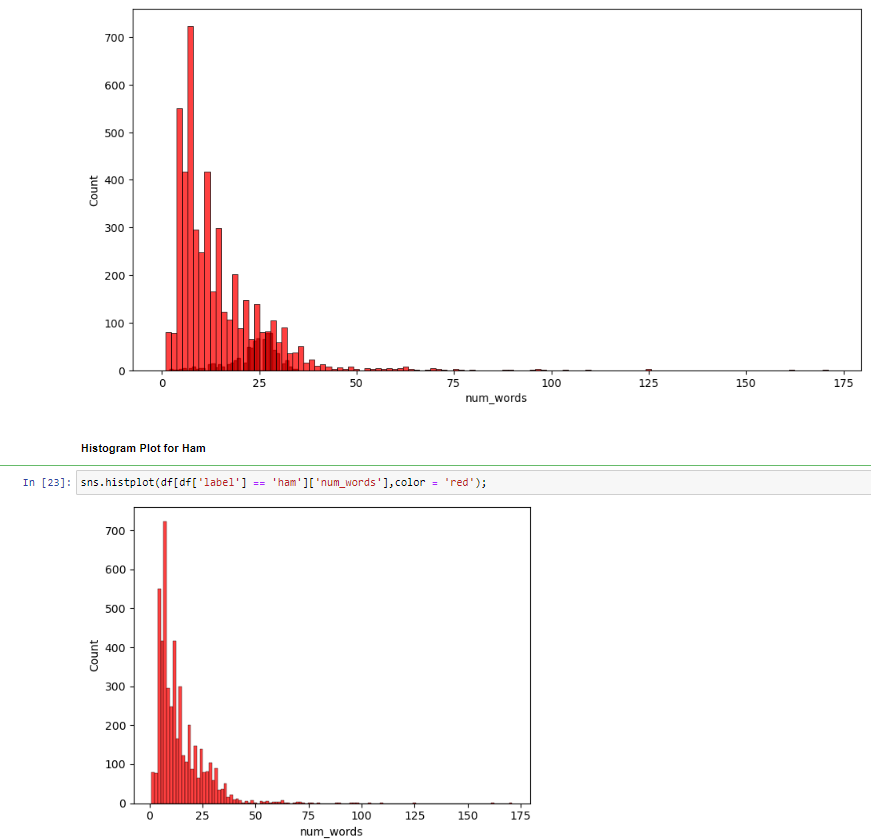
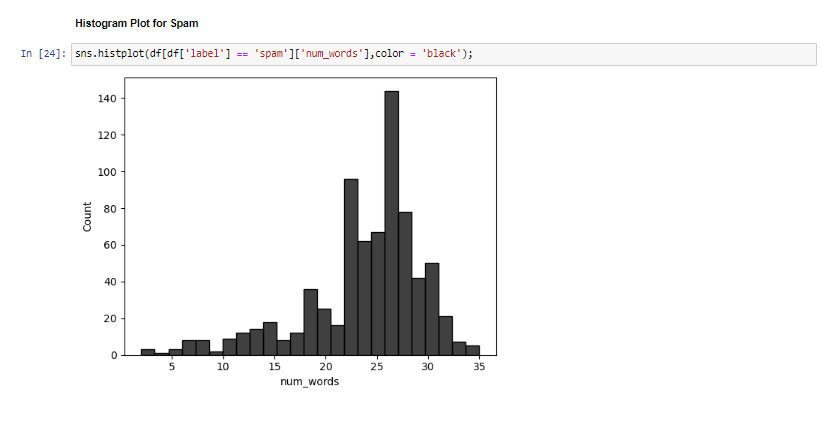
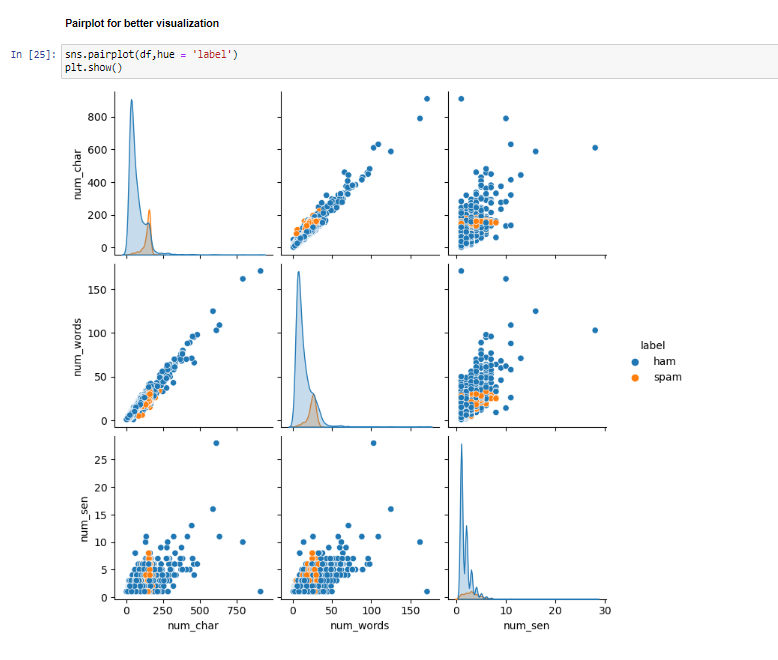
**SCREENSHOTS:**

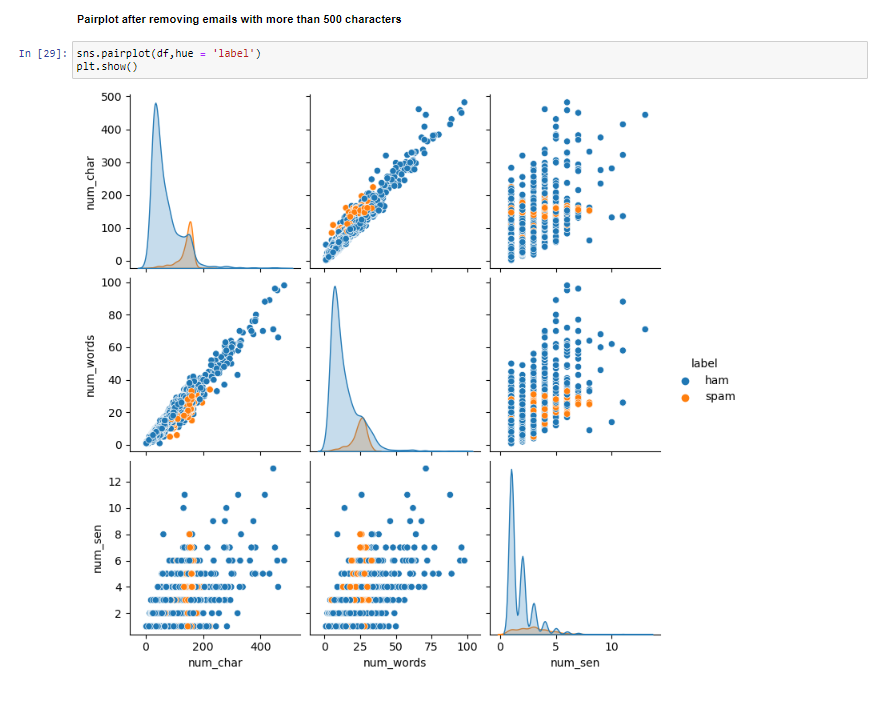


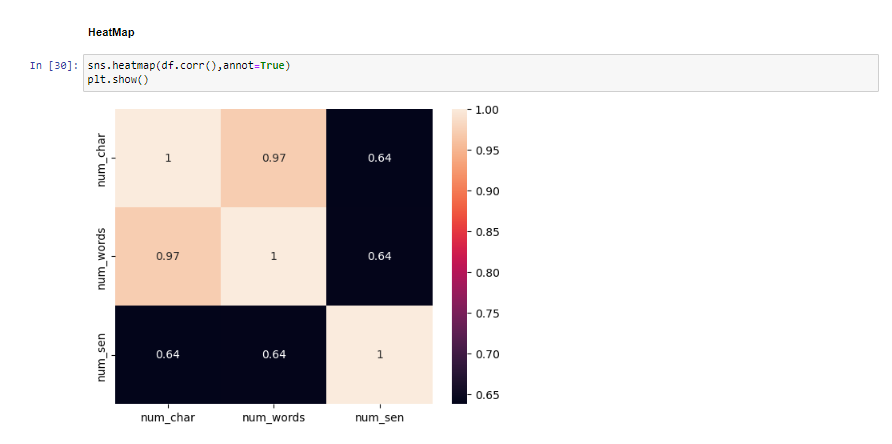
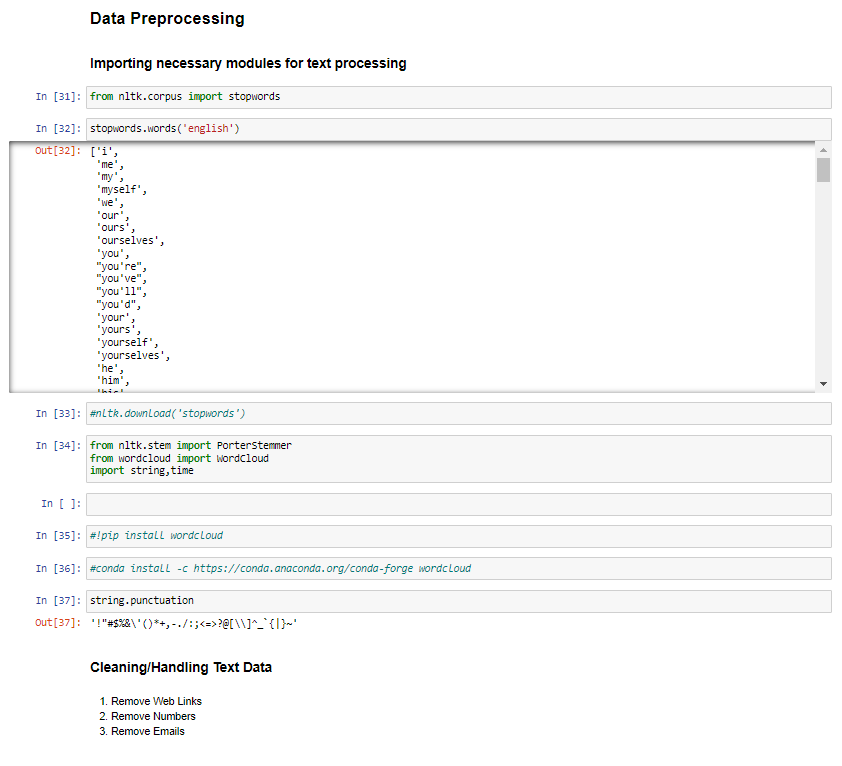
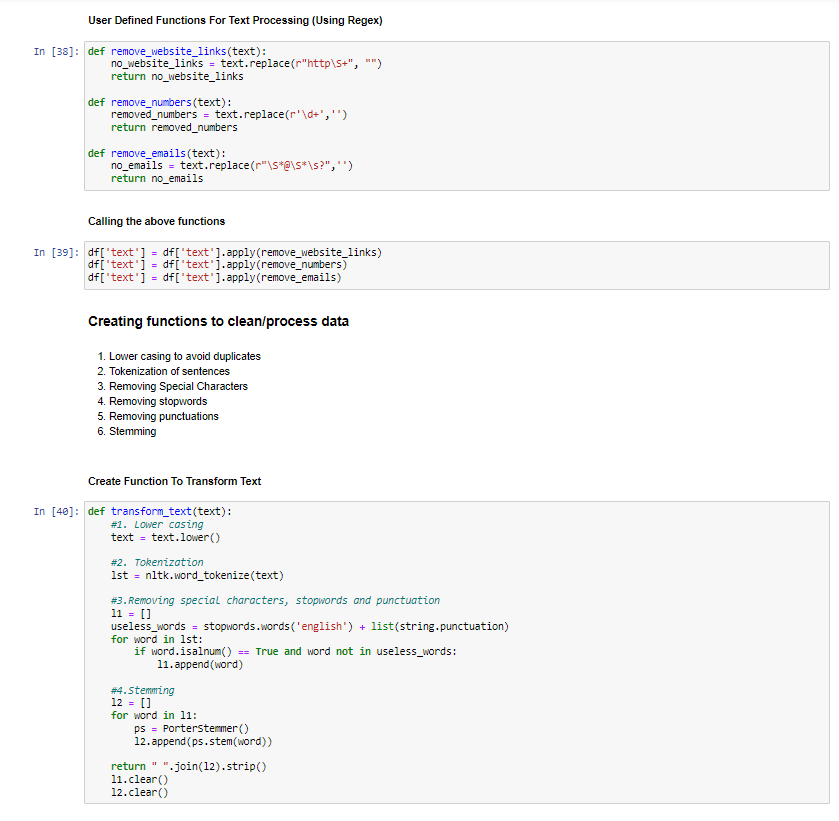
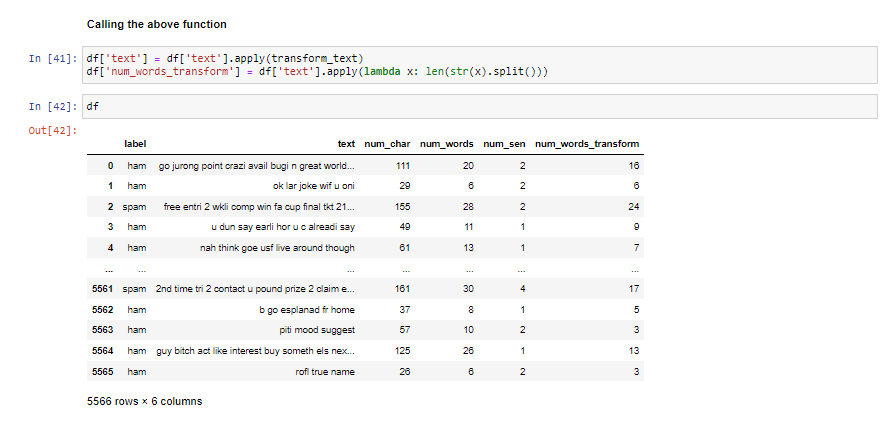
****

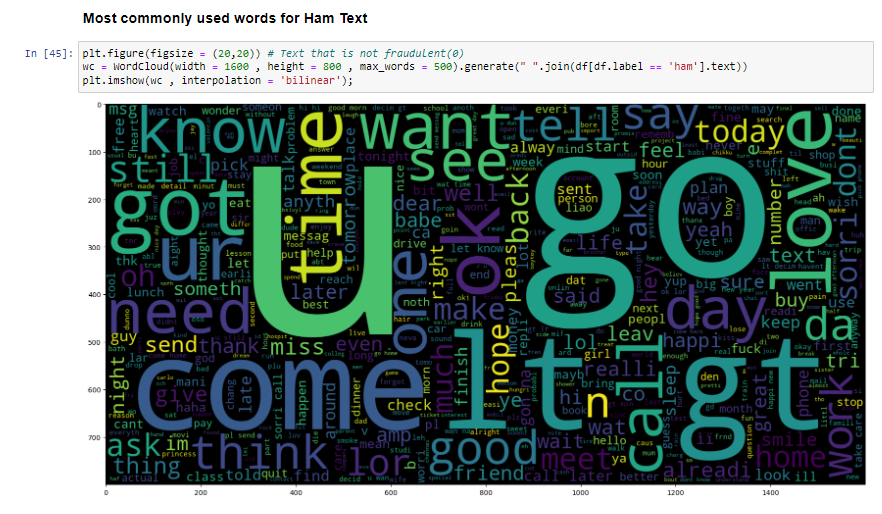
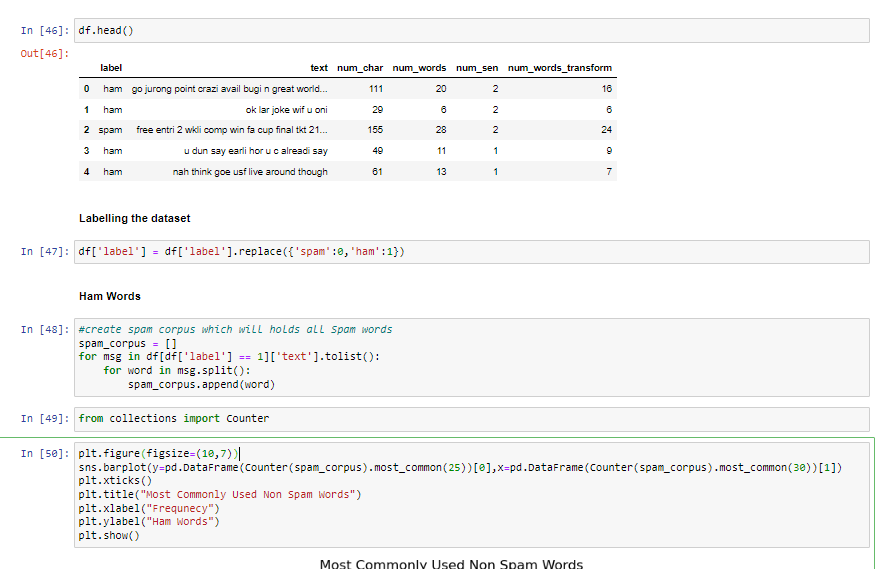
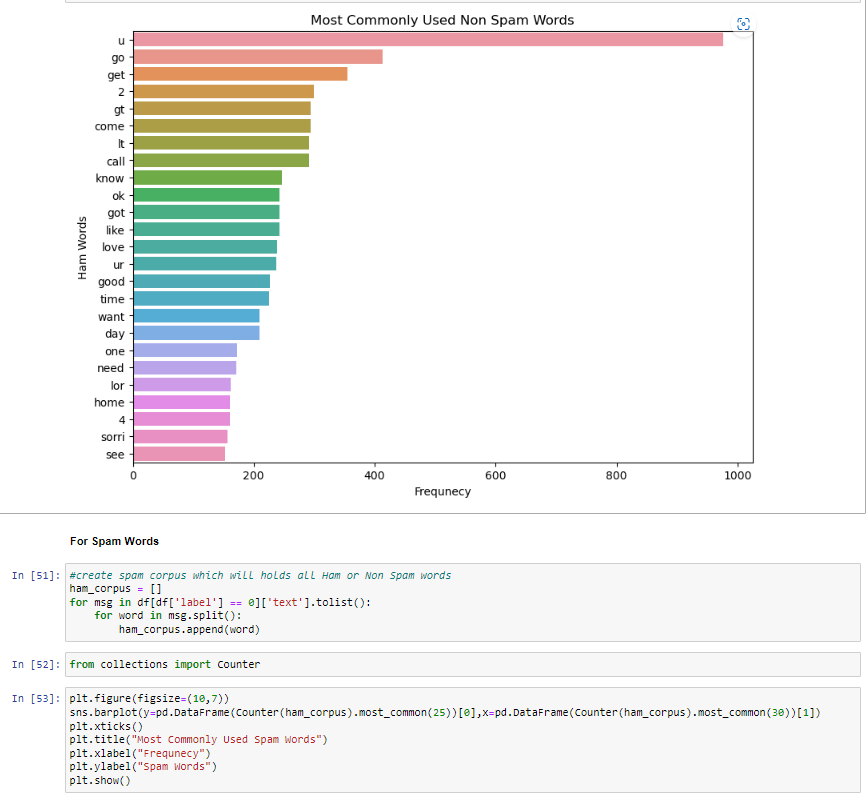
****

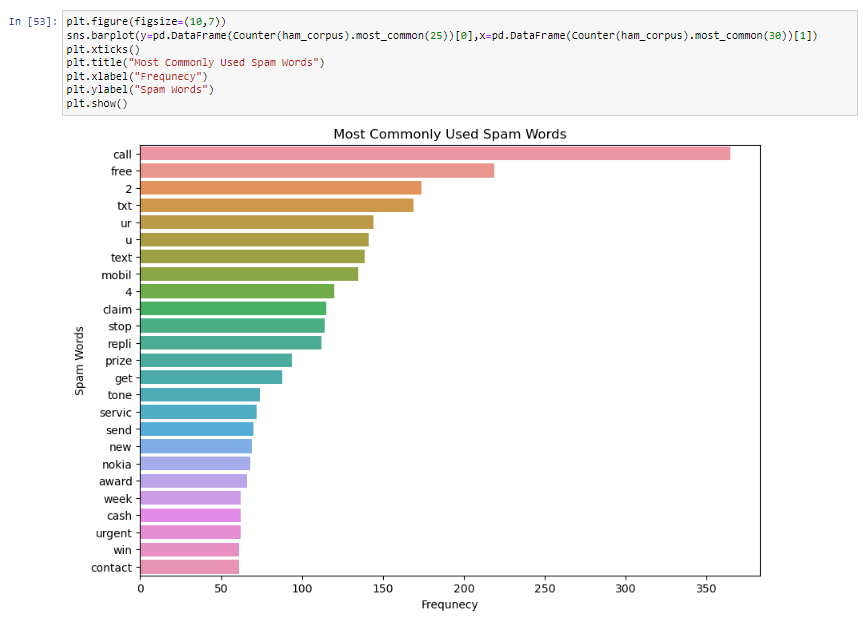
****

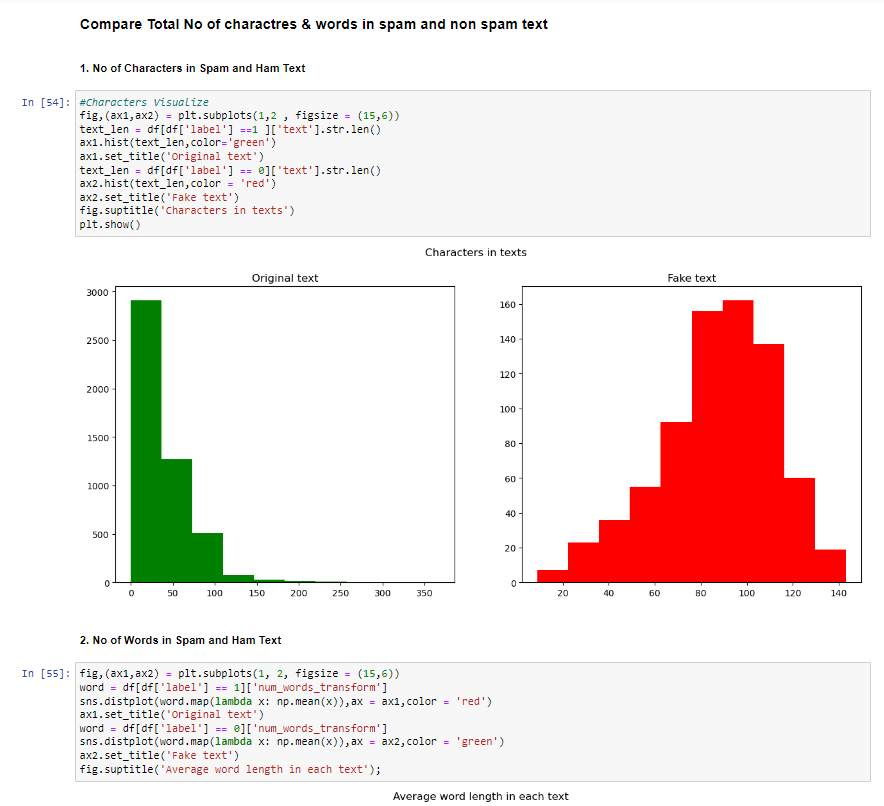
**   **

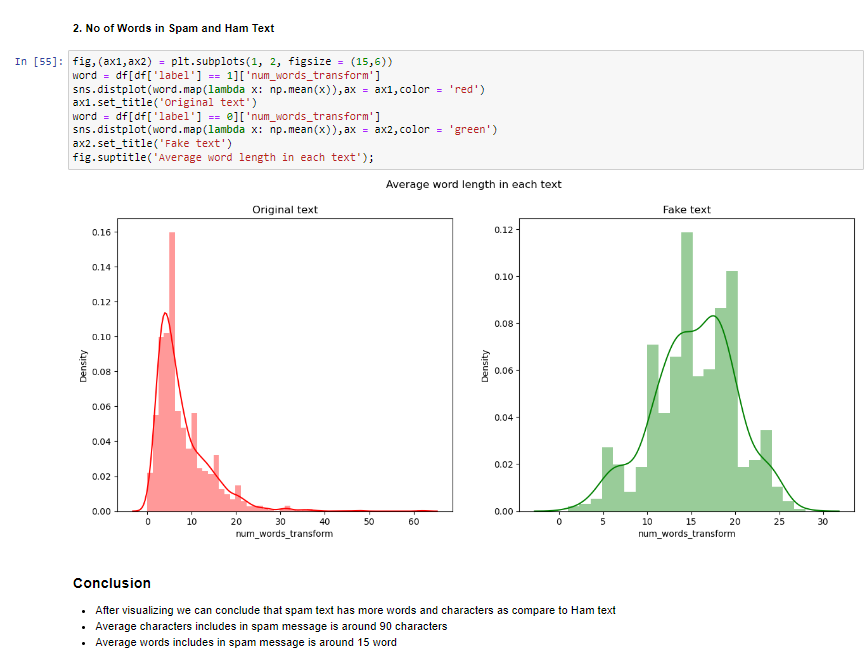
** **

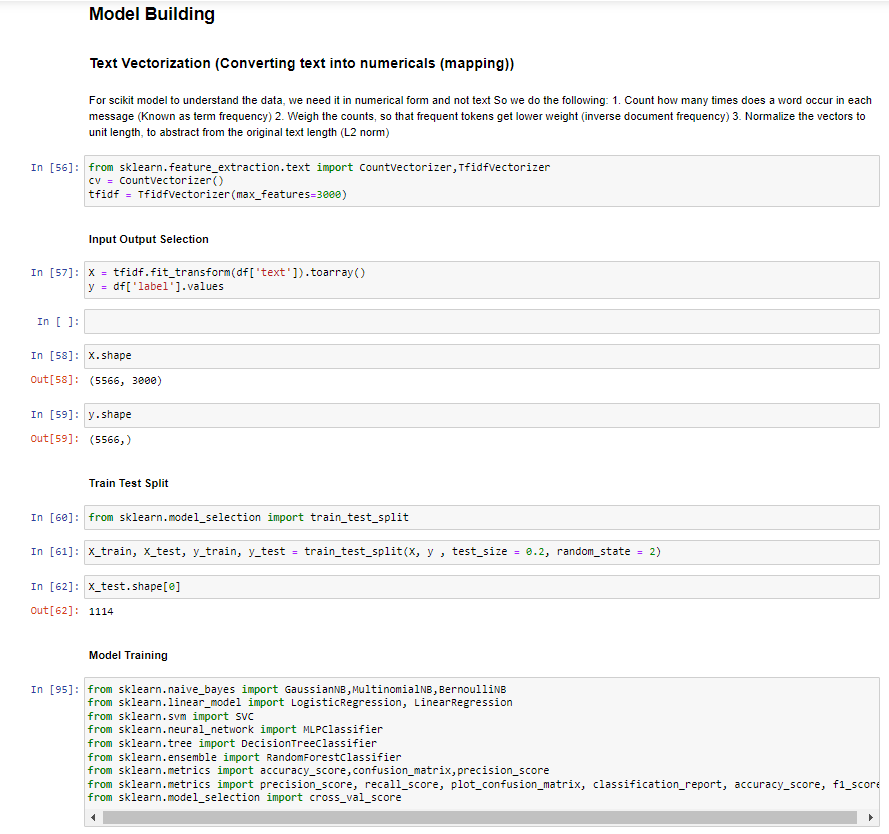
**   **

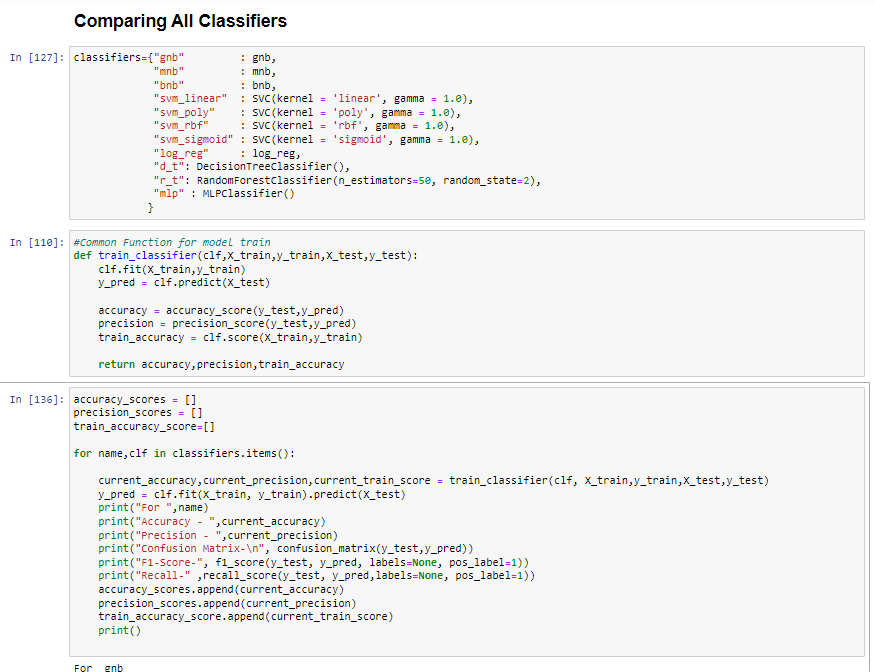
**   **

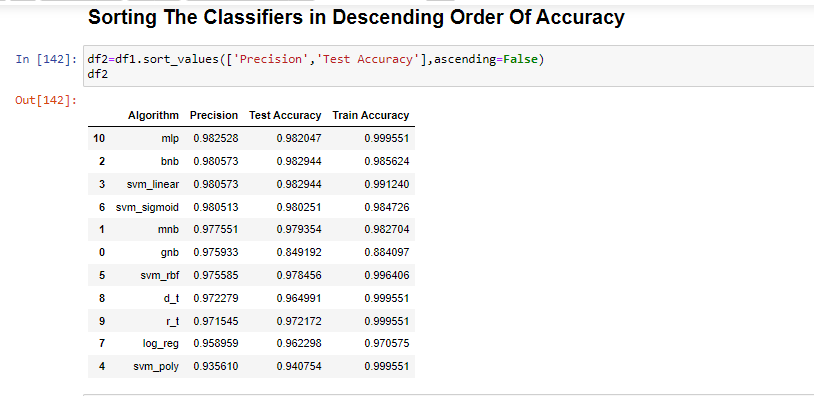
****



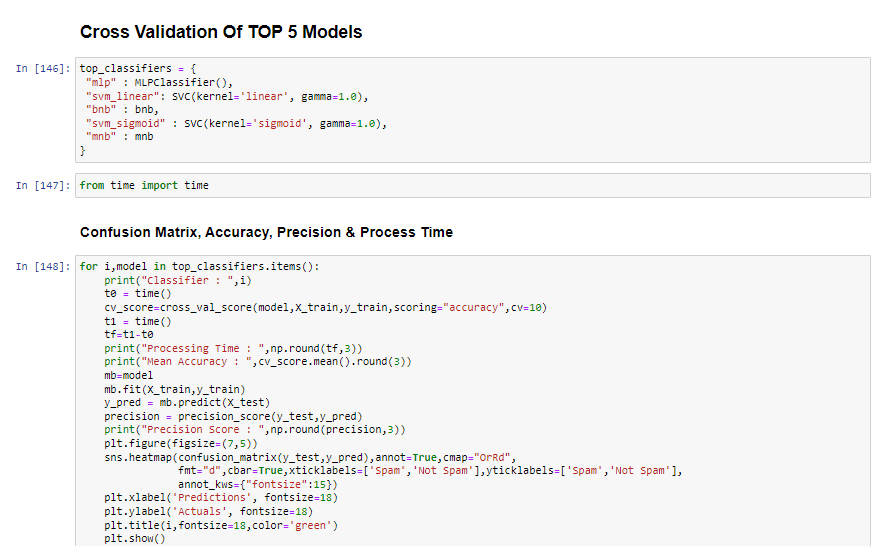
****

** **

** **

****

****

****

**PERFORMANCE MEASURES:**

1. **Confusion Matrix**

The detection of spam emails can be evaluated by different performance measures. Confusion Matrix is being used to visualise the detection of the emails for models. Confusion matrix can be defined as below: where:

1) TN = True Negative – Ham email predicted as ham

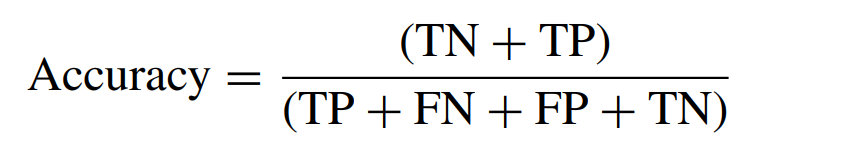
2) TP = True Positive – Spam email predicted as spam

3) FP = False Positive – Spam email predicted as ham

4) FN = False Negative – Ham email predicted as spam

1. **Accuracy**

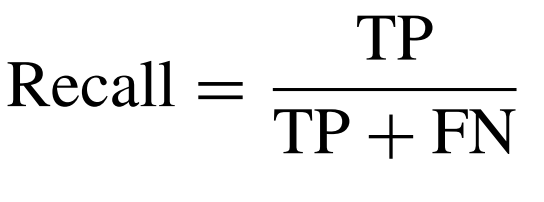
The research was aimed at finding the highest accuracy for detecting the emails correctly as ham and spam. The module from the Scikit-learn library called ‘Accuracy’ helped analyse the correct number of emails classified as ‘Spam’ and ‘Ham’. This can be measured by:

****

where the denominator of the equation is the total number of emails within the testing data.

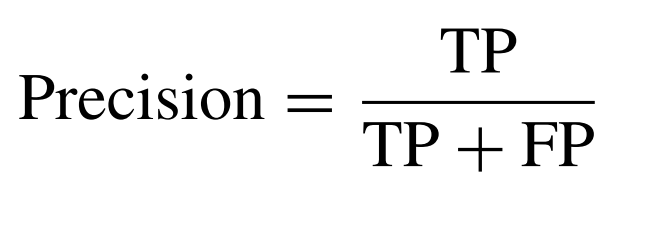
1. **Recall**

The recall measurement provides the calculation of how many emails were correctly predicted as spam from the total number of spam emails that were provided. This is defined by equation-14, where ‘TP + FN’ are the total number of spam emails within the testing data.

****

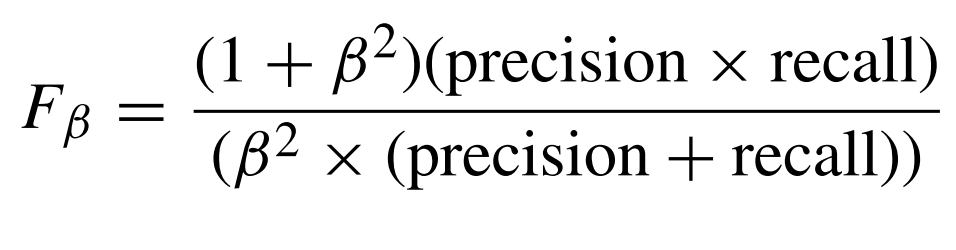
1. **Precision**

The precision measurement is to calculate the correctly identified values, meaning how many correctly identified spam emails have been classified from the given set of positive emails. This means to calculate the total number of emails which were correctly predicted as positive from amongst the total number of emails predicted positive.

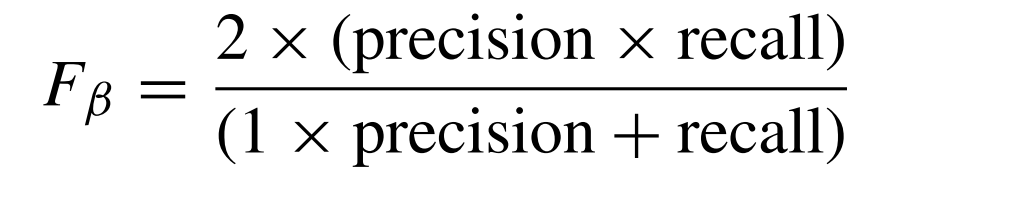


1. **F1 Score**

The F-measure or the value of Fβ is calculated with the help of precision and recall scores, where β is identified as 1, Fβ or F1 provides the F1-score. F1-score is the ‘Harmonic mean’ of the precision and recall values.

****

when the β is substituted with the value 1, the formula is simplified to:



**CONCLUSION**

Thus, we get this table after training and evaluating all the models.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Accuracy** | **Precision** | **F1-Score** | **Recall** |
| Gaussian Naive Bayes | 84.91921005 | 97.59326113 | 0.906145251 | 0.845672576 |
| Multinomial Naive Bayes | 97.93536804 | 97.75510204 | 98.81382156 | 0.998957247 |
| Bernoulli Naive Bayes | 98.29443447 | 98.05725971 | 0.990191017 | 1 |
| Support Vector Machine - Linear | 98.29443447 | 98.05725971 | 0.990191017 | 1 |
| Support Vector Machine - Polynomial | 94.07540395 | 93.56097561 | 0.966733871 | 1 |
| Support Vector Machine - RBF | 97.84560144 | 97.5584944 | 0.987641607 | 1 |
| Support Vector Machine - Sigmoid | 98.02513465 | 98.05128205 | 0.980512821 | 0.980512821 |
| Logistic Regression | 96.22980251 | 96.22980251 | 0.962298025 | 0.962298025 |
| Decision Tree | 96.67863555 | 97.62396694 | 0.976744186 | 0.98540146 |
| Random Forest | 97.21723519 | 97.15447154 | 0.984045291 | 0.996871741 |
| Multi Layer Perceptron | 98.20466786 | 98.25282631 | 0.989648033 | 0.996871741 |

We can see that Multi Layer Perceptron, which is an Artificial Neural Network, has the largest precision, followed by Bernoulli’s Naive Bayes and SVMy(Linear) with the shared second place. Thus, we can deploy the model using MLP.

**REFERENCES**

[5] A. Wijaya and A. Bisri, ‘‘Hybrid decision tree and logistic regression classifier for email spam detection,’’ in Proc. 8th Int. Conf. Inf. Technol. Electr. Eng. (ICITEE), Oct. 2016, pp. 1–4, doi: 10.1109/ICITEED.2016. 7863267.

[2] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, ‘‘Machine learning for email spam filtering: Review, approaches and open research problems,’’ Heliyon, vol. 5, no. 6, Jun. 2019, Art. no. e01802, doi: 10.1016/j.heliyon.2019.e01802.

[4] S. Mohammed, O. Mohammed, and J. Fiaidhi, ‘‘Classifying unsolicited bulk email (UBE) using Python machine learning techniques,’’ Int. J. Hybrid Inf. Technol., vol. 6, no. 1, pp. 43–55, 2013. [Online]. Available: https://www.researchgate.net/publication/236970412\_Classifying\_ Unsolicited\_Bulk\_Email\_UBE\_using\_Python\_Machine\_Learning\_ Techniques

[1] W. Feng, J. Sun, L. Zhang, C. Cao, and Q. Yang, ‘‘A support vector machine based Naive Bayes algorithm for spam filtering,’’ in Proc. IEEE 35th Int. Perform. Comput. Commun. Conf. (IPCCC), Dec. 2016, pp. 1–8, doi: 10.1109/pccc.2016.7820655.

[3] W. Awad and S. ELseuofi, ‘‘Machine learning methods for spam E-Mail classification,’’ Int. J. Comput. Sci. Inf. Technol., vol. 3, no. 1, pp. 173–184, Feb. 2011, doi: 10.5121/ijcsit.2011.3112.