Capstone_Project 3_Report

IDENTIFY WHETHER A TEXT MESSAGE IS SPAM OR HAM

Table

List of Figures:

1. Problem Statement:

To predict the message as spam (junk) or ham (legitimate) by building a spam filter (model).

1.1 Description:

Spam text messages have become very common nowadays. It's difficult for a layman to identify whether the text message is spam or a ham since the texture of both spam and ham looks similar. They consist of inappropriate content framed by regularly used words, links to illegal websites, advertisements, blank messages etc. which may lead one to fall prey to situations like losing money, account hack etc.

To overcome these kinds of situations machine learning models come into picture. They determine the messages spam or ham based on the previously trained data and classifies them as junk messages. This will save a person from falling prey to bad situations.

2. Data Wrangling:

The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according being ham (legitimate) or spam. The file contains one message per line. Each line is composed by two columns: v1 contains the label (ham or spam) and v2 contains the raw text. A collection of 425 SMS spam messages was manually extracted from the Grumbletext Web site. A subset of 3,375 SMS randomly chosen ham messages of the NUS SMS Corpus (NSC), which is a dataset of about 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore. A list of 450 SMS ham messages collected from Caroline Tag's PhD Thesis. They have incorporated the SMS Spam Corpus v.0.1 Big which has 1,002 SMS ham messages and 322 spam messages

Since the dataset is extracted from different sources selectively it contains no null values no missing values and contains only text messages. In turn text message contains email address, website address, phone number, money symbol and digits. The dataset contains 86.59% of ham messages and 13.41% of spam messages. This is the clear indication of imbalance dataset and practically it’s true also.

3. Exploratory Data Analysis:

It’s a good idea to draw the bar graph to check the imbalance dataset.

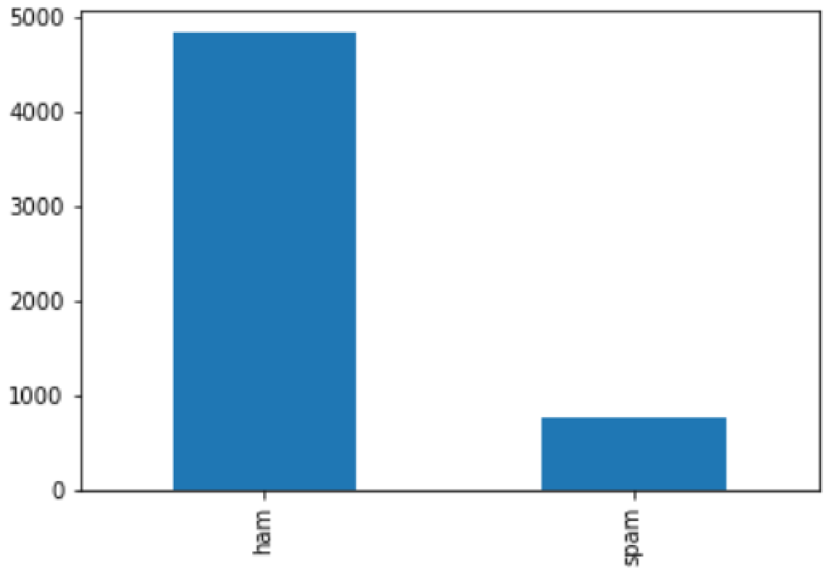

Figure 3.1: Bar graph which shows the imbalanced dataset

The above bar graph shows the number of ham messages is more in number compare to spam messages.

Let's draw the pie chart just to get the view of spam and ham message distribution.

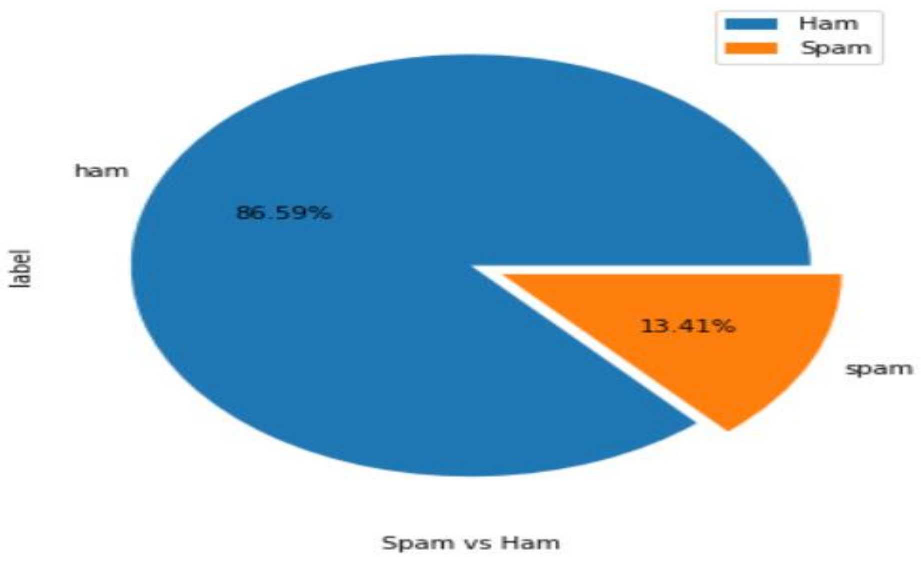

Figure 3.2: Pie Chart which shows percentage distribution

After finding the length of each message we found the messages of minimum length 2 (2 characters) maximum length of 910 and average length of 80 characters.

Let's find the average word length of messages by plotting the bar graph.

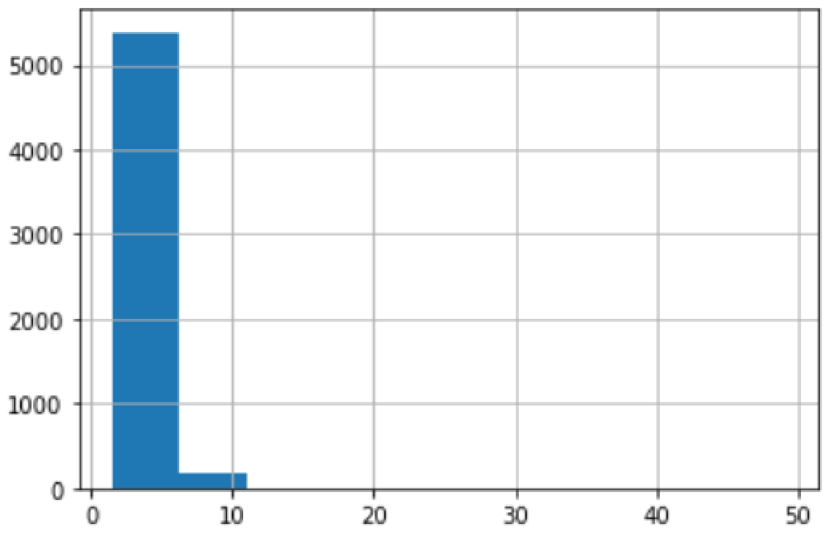

Figure 3.3.Bar graph showing the minimum and maximum word length

The average word length ranges between 2 to 10 with 5 being the most common length.

It’s important to know the each word in the message to decide whether it’s a spam or ham. Hence we are splitting the messages into words and with the help of Counter () module and most_common () function we found the most commonly occurring words. The following bar plot shows the top 10 most occurred words in messages.

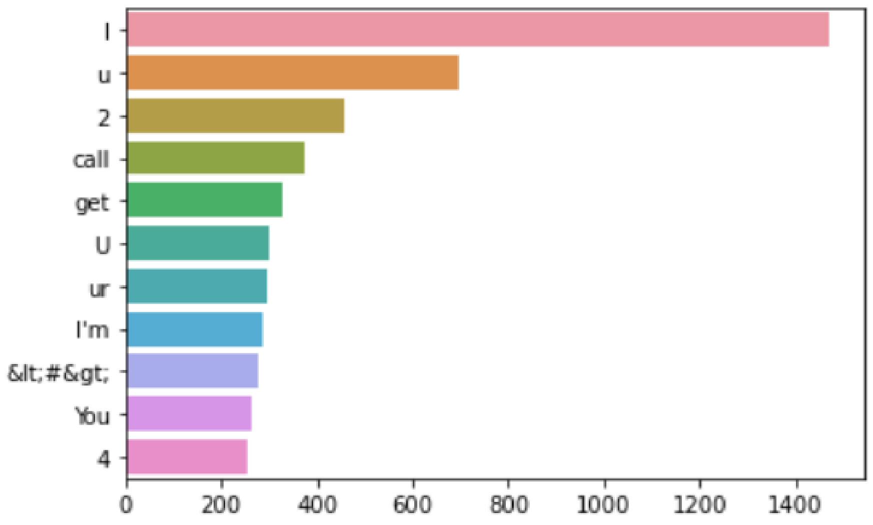

Figure 3.4 Bar plot showing the top ten most occurred words

The above bar plot shows ‘I’, ‘u’, ‘2’, ‘call’, ‘get’, ‘U’, ‘ur’, ‘I’m’, ‘you’, ‘4’, are the most commonly occurred words in the messages.

Now we categorize the messages into spam and ham based on the label column and split the messages into words. So now we have two categories of words one is spam words and other is ham words.

Let’s draw the wordcloud images for both spam and ham words which shows the most frequently occurred word with large font size and less frequently occurred word with small font size.

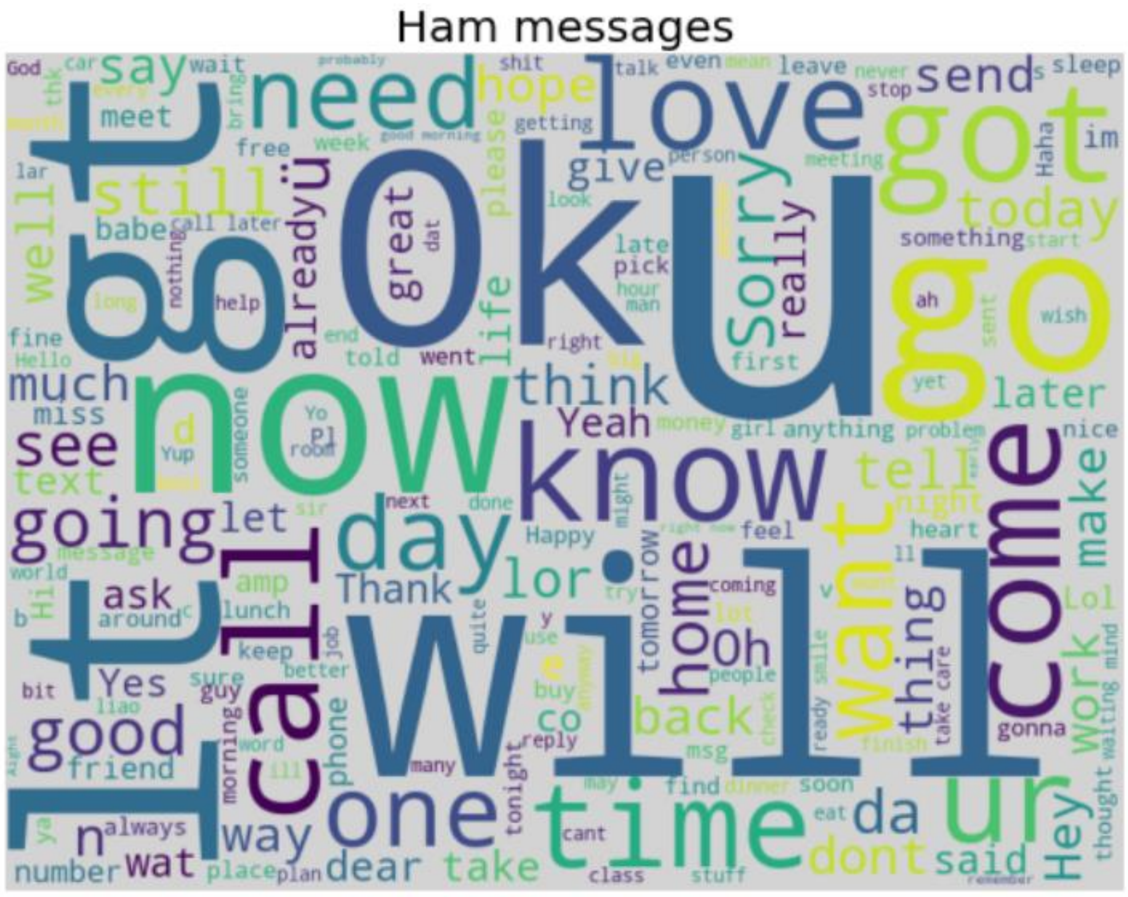

Figure 3.5 WordCloud Image for Ham words

A word cloud is an image made of words that together resemble a cloudy shape. Word clouds offer a striking way to visualize text. People typically use word clouds to easily produce a summary of large documents (reports, speeches), to create art on a topic (gifts, displays) or to visualise data (tables, surveys). Word clouds are widely used for analysing data from social network websites. Wordcloud process the text, remove punctuation, and ignore case and words that do not contain all alphabets, count the frequencies, and ignore uninteresting or irrelevant words. A dictionary is the output of the calculated frequencies of the words. The wordcloud module will then generate the image from your dictionary.

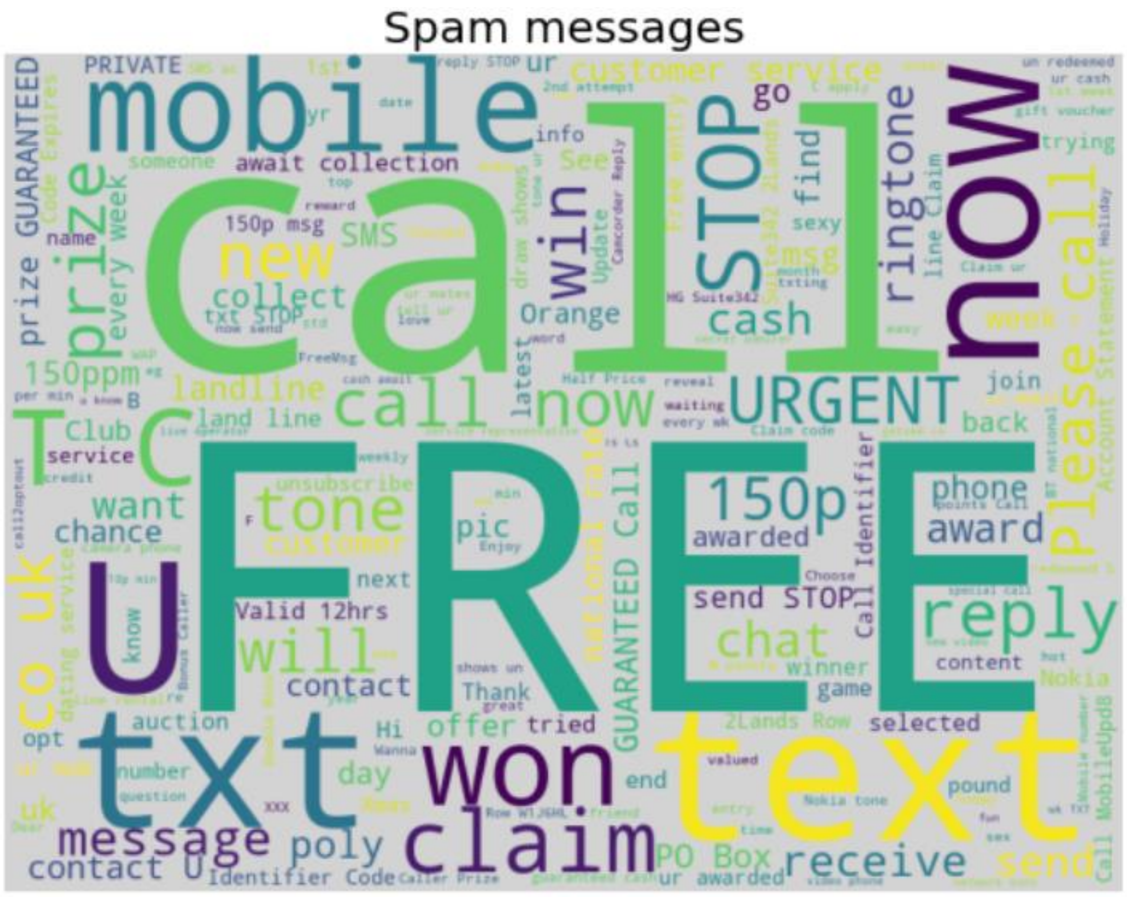

Figure 3.6 WordCloud Image for Spam words

4. Feature Engineering:

In this step we first convert the label column which contains spam and ham labels. We assign 0 to ham and 1 to spam.

We use regular expressions to replace email addresses, URLs, phone numbers, other numbers. Here we replace email addresses with 'email' , URLs with 'webaddress', money symbols (£, $) with 'moneysymb', 10 digit phone numbers with 'phonenumber' and numbers with 'numbr’, punctuation with white space, whitespace between terms with a single space, removing leading and trailing whitespace and at last changing all words to lower case.

4.1 Stopwords:

“stopwords” usually refers to the most common words in a language. These words in any language which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence. For some search engines, these are some of the most common, short function words, such as the, is, at, which, and on. Hence we remove these stopwords with the help of python library stopword.

4.2 Stemming:

Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. It is just like cutting down the branches of a tree to its stems. For example, the stem of the words eating, eats, eaten is eat. Stemming is an important part of the pipelining process in Natural language processing. The input to the stemmer is tokenized words. Tokenization involves breaking down the document into different words.

After removing stopwords and stemming we left with only those words which add a meaning to the sentence of a message.

4.3 Bag of Words Model:

We cannot pass text directly to train our models in Natural Language Processing, thus we need to convert it into numbers, which machine can understand and can perform the required modelling on it. The Bag of Words model is a fundamental way of doing this. The model is very simple as it discards all the information and order of the text and just considers the occurrences of the word; in short it converts a sentence or a paragraph into a bag of words with no meaning. It converts the documents to a fixed-length vector of numbers. A unique number is assigned to each word (generally index of an array) along with the count representing the number of occurrence of that word. This is the encoding of the words, in which we are focusing on the representation of the word and not on the order of the word.

4.4 CountVectorizer:

Since machine learning models don’t understand text, we need to convert them into vectors. CountVectorizer is one tool provided by the scikit-learn library in Python which is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text.

CountVectorizer creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix. The value of each cell is nothing but the count of the word in that particular text sample.

5. Modeling:

The dataset we use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model's prediction on this subset. Except the target variable (label) consider the dataset as one variable called X and target variable as y. We do this using the Scikit-Learn library and specifically the train_test_split method.

Since the project is related to text classification here we are implementing 8 different classification models. Following are the accuracy provided by the models.

Logistic Regression	---> Test accuracy - 98.30%
Naive Bayes classifier	---> Test accuracy - 98.48%
K-nearest neighbors	---> Test accuracy - 95.61%
Support Vector Machine	---> Test accuracy - 87.17%
Decision Tree Classifier	---> Test accuracy - 97.94%
Random Forest Classifier	---> Test accuracy - 98.39%
Gradient Boosting Classifier	---> Test accuracy - 98.12%
AdaBoost Classifier	---> Test accuracy - 98.30%

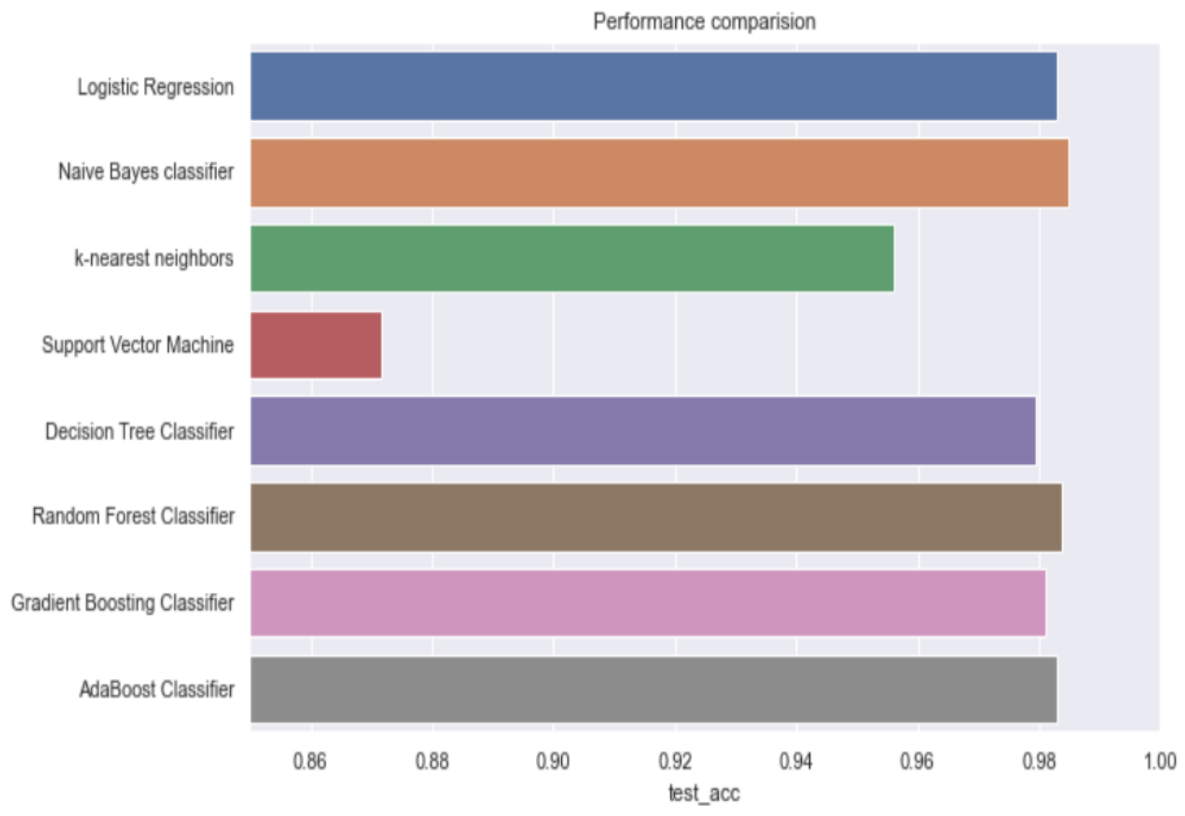

Figure 5.1 Performance comparisons of the models

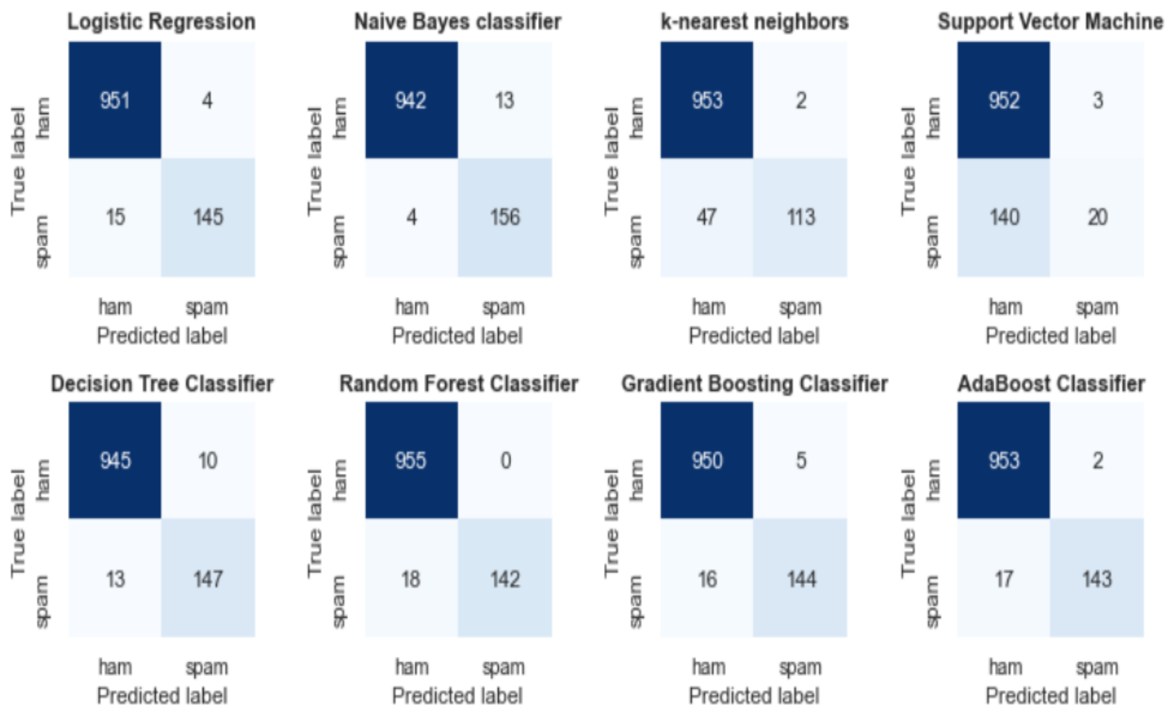

Figure 5.2 Confusion Matrix of models

Among 8 models Naïve Bayes Classifier showing the highest accuracy of 98.48%. Hence we go with this module for deployment.

6. Model Deployment:

The concept of deployment in data science refers to the application of a model for prediction using a new data. Building a model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data science process. In many cases, it will be the customer, not the data analyst, who will carry out the deployment steps.

6.1 requirements:

Applications	Virtualization
Data	Servers
Runtime	Storage
Middleware	Networking
OS	

Here we are using Heroku cloud platform to deploy our model. Heroku is PaaS (Platform as a Service) kind of platform. PaaS delivers a framework for developers that they can build upon and use to create customized applications. All servers, storage, and networking can be managed by the enterprise or a third-party provider while the developers can maintain management of the applications. This platform is delivered via the web, giving developers the freedom to concentrate on building the software without having to worry about operating systems, software updates, storage, or infrastructure.

The following are the images showing the whether a message is spam or ham after deployment

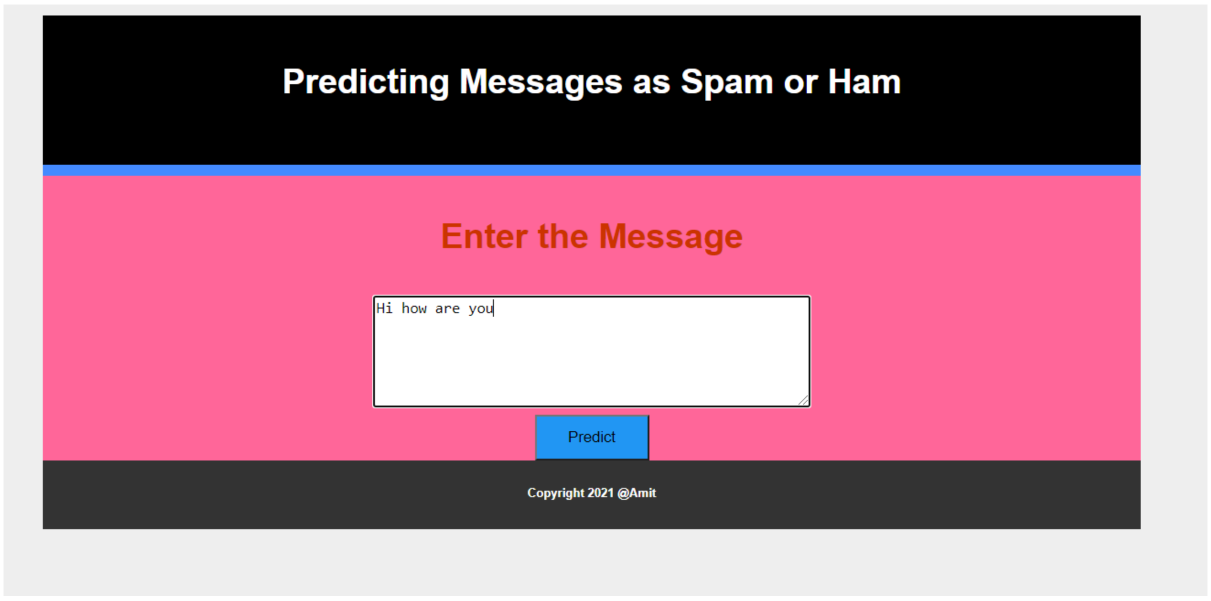

Figure 6.1 Input Message(Ham)

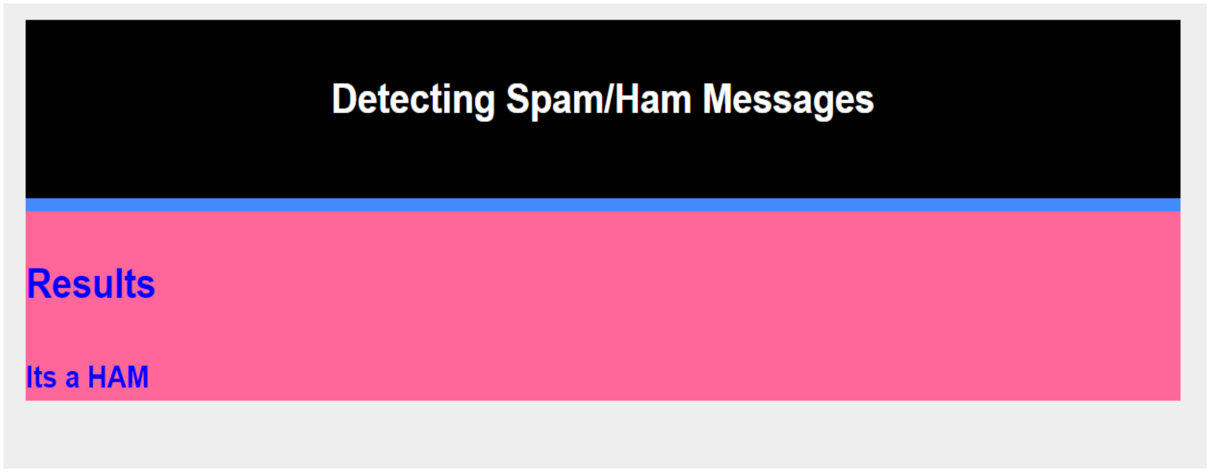

Figure 6.2 Output Result(Ham)

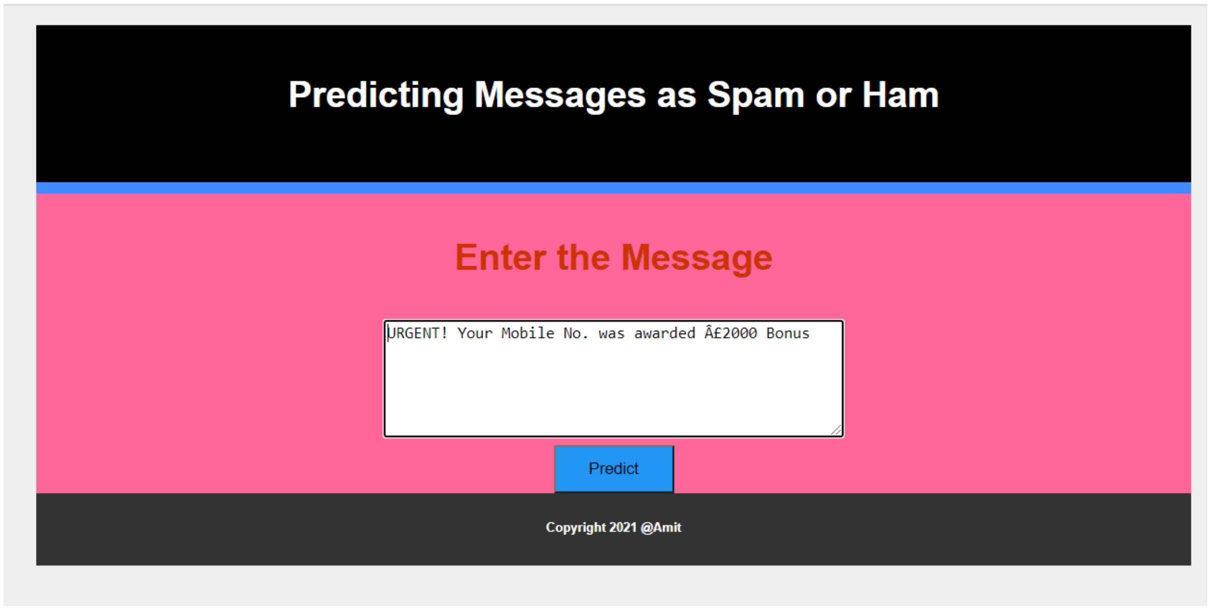
Predicting Messages as Spam or Ham
Enter the Message
URGENT! Your Mobile No. was awarded Â£2000 Bonus
Predict
Copyright 2021 @Amit

Figure 6.3 Input Message(Spam)

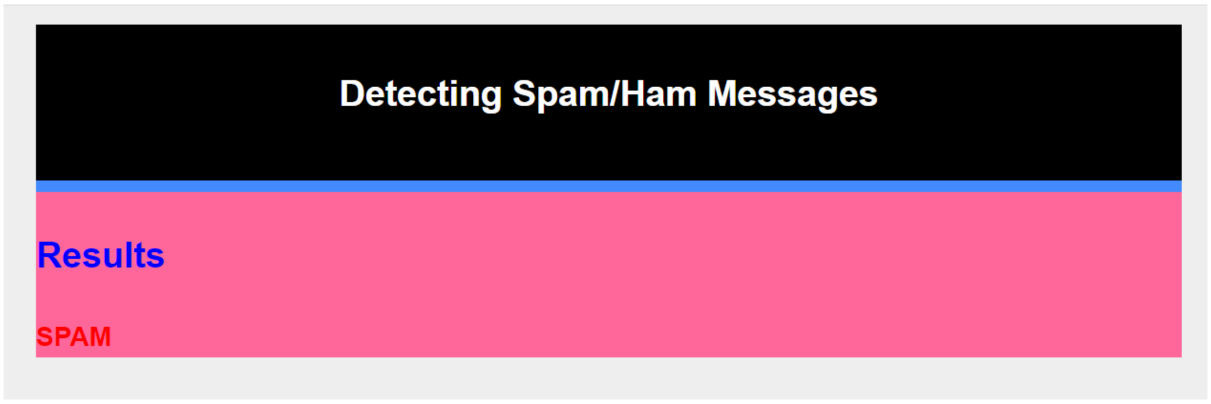

Figure 6.4 Output Result(Spam)

7. Conclusion and Future Scope:

We have used machine learning algorithms to predict the whether a message is spam or ham. We have mentioned the step by step procedure to analyse the dataset and created the vector for the document. These feature set (vector) were then given as an input to algorithms and observed their performance. Hence we calculated the performance of each model using RMSE metric and compared them and Naive Bayes classifier came out with high percentage of success hence we selected that as a final model for deployment.

For future work, we recommend that working on large dataset would yield a better and real picture about the model. We have undertaken only few Machine Learning algorithms that are actually classifiers but we need to train many other classifiers and understand their predicting behaviour. By improving the error values these models can be useful for development of applications for various situations.