Report - Assignment 2

Nikshay Jain - MM21B044

November 2024

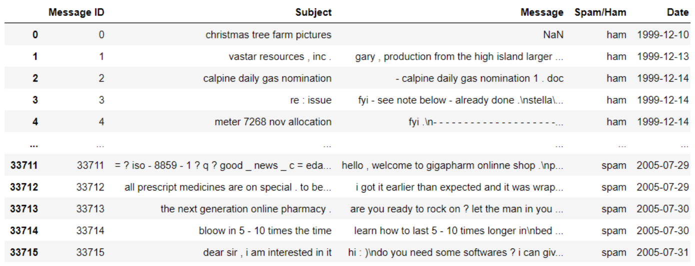
1 Introduction

The project contains 2 python files: `Model_training.py and model_testing.py`. For the entire training part, we will refer to the former only.

2 Dataset

The classifier that is trained upon the openly available **Enron dataset**, which is named as `enron_spam_dataset.csv` in the `Solutions_MM21B044.zip`.

It contains **33716 rows and 5 columns**: viz: Message ID, Subject, Message, Spam/Ham, Date. For convenience, we use only Message and Spam/Ham (renamed to labels) columns to train the classifier and start the data pre-processing before training.



	Message ID	Subject	Message	Spam/Ham	Date
0	0	christmas tree farm pictures	NaN	ham	1999-12-10
1	1	vastar resources , inc .	gary , production from the high island larger ...	ham	1999-12-13
2	2	calpine daily gas nomination	- calpine daily gas nomination 1 . doc	ham	1999-12-14
3	3	re : issue	fyi - see note below - already done .\nstella\...	ham	1999-12-14
4	4	meter 7268 nov allocation	fyi .\n- - - - - - - - - - - - - - - - - - -...	ham	1999-12-14
...	...	...	...	...	...
33711	33711	= ? iso - 8859 - 1 ? q ? good _ news _ c = eda...	hello , welcome to gigapharm onlinne shop .\np...	spam	2005-07-29
33712	33712	all prescript medicines are on special . to be...	i got it earlier than expected and it was wrap...	spam	2005-07-29
33713	33713	the next generation online pharmacy .	are you ready to rock on ? let the man in you ...	spam	2005-07-30
33714	33714	bloow in 5 - 10 times the time	learn how to last 5 - 10 times longer in\nbed ...	spam	2005-07-30
33715	33715	dear sir , i am interested in it	hi :)\ndo you need some softwares ? i can giv...	spam	2005-07-31

Figure 1: Dataset for Spam/Ham classifier

3 EDA and pre-processing

On calculating, the ratio of spam to ham emails comes out to be = 1.03, which indicates the data is quite balanced.

We delete the Message ID and Date columns and convert the labels into numeric type by 1 and 0 for spam and ham, respectively. Also, we extract a list of words from the Message after converting it to lowercase and removing all special characters from it by using the `make_usable` function.

The last step is train-test split which is facilitated by the mentioned split function, in the ratio of 80:20 for train:test.

4 Naive Bayes Model Training

We choose Naive Bayes model for this data due to its simple yet efficient architecture and start by creating dictionaries of words in spam and ham emails. This contains their respective frequencies as well.

The next step is calculating the conditional probabilities of each word, given its class. We deploy laplace smoothing to prevent error of division by zero while computing the probabilities.

We then write a function to predict the labels for given emails and name it ”predict”, which works on the principle of Maximum Likelihood Estimation (MLE).

To test the performance, we run this on the `X_test` set and get `y_pred` to be compared with `y_test`. Over the metrics of accuracy, f1-score and plot the confusion matrix as follows:

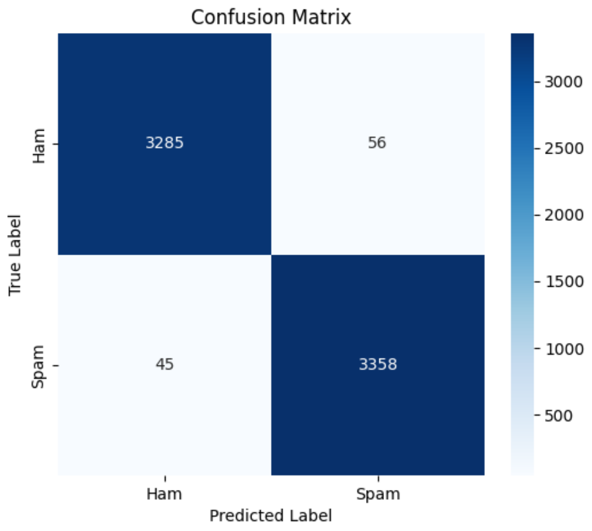


Figure 2: Confusion Matrix

The Accuracy we get = 99% and f1 score = 0.98 which is a pretty good one for binary classification. Even the area under the ROC curve states the same.

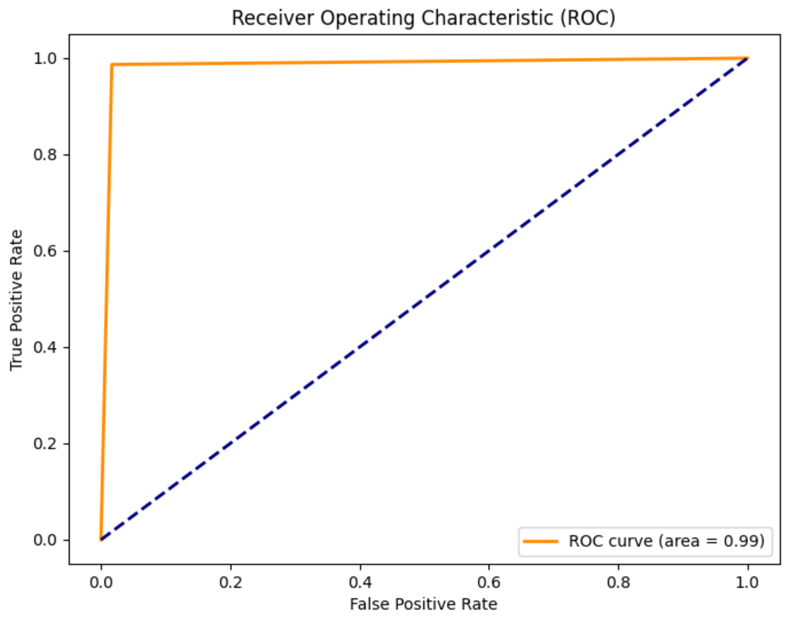


Figure 3: ROC Curve

We finally save the model parameters using pickle to be used for testing directly, without the need for retraining the model.

5 Model Testing

Here, we switch to the `model_testing.py` file and execute the code given there. It is important to add the **test folder** in the the **same working directory** and add the test emails in **.txt** format in it.

The code first loads the saved parameters (`sm_prob, hm_prob, p_spam, p_ham, tot_sm_words, tot_hm_words, vocab`) and uses the function make_usable to get the usable list from the email body and predict function, same as previously discussed.

The `test_spam` function is the one that reads the files in the folder named "test" and starts the process flow, right from pre-processing to the final prediction. The final results are added to a **.csv** file named **predictions.csv** in the format of (email-file-name, prediction). The final message that is printed is "**Predictions saved to predictions.csv**" at the end of execution.

Note that: Spam = +1 and Ham email = 0.

****