Honor code: I pledge on my honor that: I have completed all steps in the below homework on my own, I have not used any unauthorized materials while completing this homework, and I have not given anyone else access to my homework.

Shubham Randive (CS20M064)
Vivek Kumar (CS20M072)
Team Members

1. **Spam or Ham?**
   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). Two sample emails your classifier will be tested on can be found in the folder test. 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:
Naive Bayes Approach:
Firstly we have to choose the data-set, so we have chosen the largest data-set available for the Emails that is Enron data-set which contains 6 directories and approximately 15-16 thousand mails in the Raw form but labeled as Spam and ham.
After reading the data-set we separate the mails in two separate Folders named Spam and Ham, but each mail is in row form,so we essentially have to do the data cleaning.

1: Data Cleaning and Prepossessing

In this step we read each mail from the .txt file, and do this steps:
step 1: Lowercase all the words
step 2: Remove the numbers, letters, punctuation, character, new line character etc. This is done by the Regular Expression that filters out each of them leaving behind with the Words.

step 3: Remove the Duplicate words.
After doing this we have a set of words for each mail, we want to make sure that each word in the mail is meaningful and adds information to the Model when trained on these mails. So we have to make sure that we remove the words that are Stop words and the words that are not in the English Dictionary. Stop words are words like a,an,the,i, me, myself etc. For this we have downloaded a list of such stop words. For the English Dictionary we have a separate list of words. So we do these steps as.
step 4: Remove Stop Words
step 5: Remove Words that are not in the English Dictionary.
After all these 5 steps we write back the words to the same .txt file for each mail. This completes the data Cleaning and Pre-processing process.

2 :Training the Model

Since our model is using the Dictionary based approach, so we have to extract out the words from each mail and make a Dictionary of words Separately for each type of mails, that is we make two Separate dictionary for Spam and Ham words.
step 1: We read the mail as spam or ham, extract out the words and add the words in their respective dictionaries, with key as the words and value as the Count number it has appeared in the spam mail for Spam dictionary and Ham mail for the Ham dictionary.
step 2: Now we count the number of spam and Ham mails, and divide each value in the Key-Value pair in the spam dictionary with the number of spam mails, and same for ham mails. This step essentially convert the value as count number to the probabilities of each word appearing in their respective type of mails.
That is we now have two Dictionary: Spam word dictionary and Ham word dictionary, with key value pair, key as the words and value as the Probability of the word appearing in their respective type of mails.
step 3: Now we normalize the Spam and Ham dictionary by including the words in spam dictionary to ham dictionary which are not present in the Ham dictionary with the minimum probability assigned to it.
This step makes sure that a word is present in the spam and ham dictionary both.
step 4: then we store the weights or the dictionaries as the weights.txt file.

3: Classifying-Using the model

step 1: input a mail to be classified
step 2: separate out the words in the Mail and get a list of words.
Now we have to calculate the probabilities of the mail being spam and ham
step 3: for each word that is present in the input mail we multiply the probabilities of words by using the spam probabilities dictionary and we know have the spam probability of the mail.

step 4: for each word that is present in the input mail we multiply the probabilities of words by using the ham probabilities dictionary and we know have the ham probability of the mail.
step 5: Spam probability>ham probability then output as spam otherwise Ham.

4: Mathematics

This is naive bayes approach of classifying the Spam and Ham mails with some modifications of filtering out words based on the Stop-words and English Dictionary. If a certain word is present in the mail, probability of the mail being spam is calculated using the formula:

$$P(Spam|Word1) = \frac{P(word1|spam) * P(spam)}{P(word1)}$$

This is Bayes Formula. For this formula we have already calculated the P(word1—Spam), and formed the dictionary for the word. Now when we look at the denominator it is equal to the :

$$P(word1) = P(word1|spam) * P(spam) + P(word1|ham) * P(ham)$$

Now when we look at this formula , in Training we have normalized the dictionary that a word is present in the ham and Spam both with different probabilities. Hence finally we have the formula as:

$$P(Spam|Word1) = \frac{P(word1|spam) * P(spam)}{P(word1|spam) * P(spam) + P(word1|ham) * P(ham)}$$

In this formula we have the $P(spam)$ as the probability of a mail being spam, $P(ham)$ as the probability of a mail being ham. $P(word1|spam)$ as the probability of the word appearing in the spam mail. $P(word1|ham)$ as the probability of the word appearing in the ham mail. Similarly we can calculate the probability of a mail being ham if the word is present in the mail as:

$$P(Ham|Word1) = \frac{P(word1|ham) * P(ham)}{P(word1|spam) * P(spam) + P(word1|ham) * P(ham)}$$

This is for one word that is word1, we can scale these probability for the whole mail of n words. To scale up the whole mail we have to multiply the probabilities of all those n words in the mail.

$$P(spam|mail) = P(spam|w1, w2...., wn) \propto P(Spam) * \prod_{i=1}^{n} P(word_i|spam)$$

similarly for the mail being ham we can calculate it as:

$$P(ham|mail) = P(ham|w1, w2...., wn) \propto P(Ham) * \prod_{i=1}^{n} P(word_i|ham)$$

This is naive bayes approach as we are not taking into account the dependency of words on each other. So we are making an assumption that the probabilities of words are independent of each other. But in reality there exist a dependency between the words.

5: Computational and Mathematical Challenges

There exist two type of Challenges in this approach. First one being:

Challenge 1: The word in the mail might not be present in the either spam or ham Dictionary. So because of this one word- $word_j$ we have the $P(Word_j|spam) = P(Word_j|ham) = 0$ so the whole probability of mail being spam or ham get zero. $P(spam|mail) = P(ham|mail) = 0$. and mail is not classified correctly so to over come this we use the factor $\alpha = 1$ which would indicate the word being present in ham and spam mail exactly one time, so it won't have zero probabilities. Moreover this won't affect the accuracy of our model.

$$P(word_j|spam) = \frac{\alpha + SpamContainingWord_j}{TotalSpam + \alpha * K}$$

IN this formula α represents the Laplace Smoothing factor, K represents the dimension or features of the data which we have chosen as 1.
This makes sure that a word which is not present in the training its probability will not be zero.

Challenge 2: Since we multiplying the probabilities which are essentially less than 1, the overall product will be a very small number approximately equal to zero, this is called as "Arithmetic Underflow" problem which will essentially turn our number equal to zero. To tackle this problem we are using the "Logarithm and Exponent" trick. Since we know the formulas as:

$$log(ab) = log(a) + log(b)$$
$$exp(log(x)) = x$$

We have used this trick to multiply the probabilities, so as a step we have modified our Spam and Ham dictionary in which we have key-value pair, with key as the word and the value as the log of the probabilities. So we can straightaway add up the probabilities.

$$log(P(spam|mail)) = log(P(spam|w1, w2.., wn)) \propto \sum_{i=1}^{n} P(word_i|spam)$$
$$log(P(ham|mail)) = log(P(ham|w1, w2.., wn)) \propto \sum_{i=1}^{n} P(word_i|ham)$$

and as the next step we implement the formula:

$$exp(log(P(spam|mail))) = P(spam|mail)$$
$$exp(log(P(ham|mail))) = P(ham|mail)$$

6: Accuracy

We have the trained our model using the Naive Bayes algorithm. Now we must test the accuracy we have achieved using the approach. We test the accuracy on the test data we have separated earlier using the Train-Test split. We have a separate test data of 300 Ham mails and 300 Spam mails.
Our model was able to correctly classify the Ham Mails with 96.5 percent accuracy and Spam mails with 89 percent accuracy.
The mails that were given as the test mail in the assignment were classified correctly as Ham and Spam.

7:References

To run the code we have the main method which calls out the classifier function to classify the mail. We have to store the mails in the test directory and the code will output the classification result as output.txt file.

8:References

1. Data-set: http://www2.aueb.gr/users/ion/data/enron-spam/
2. Stop-Words: https://gist.github.com/sebleier/554280
3. English Dictionary:https://github.com/dwyl/english-words/blob/master/words.txt