Report:

1. On executing the command **“python Spamfilter.py train test1”,** the precision of the spam filter is 90%. The Naïve Bayes theorem is applied with the smoothing parameter as 1, 2, 3.

   “3” as smoothing parameter gave the best result i.e “91%”.

2. On executing the command **“python Spamfilter.py train test2”,** the precision of the spam filter is 91%. The Naïve Bayes theorem is applied with the smoothing parameter as 1, 2, 3.

   “3” as smoothing parameter gave the best result i.e “91%”.

3. On executing the command **“python Spamfilter.py train test3”,** the precision of the spam filter is 91%. The Naïve Bayes theorem is applied with the smoothing parameter as 1, 2, 3.

   “3” as smoothing parameter gave the best result i.e “91%”.

4. From all the test data tried out, the smoothing parameter “3” worked best.

5. The precision of the spam filter is 85% if the smoothing is not considered.

Chart showing the result at a glance:

Train Data Considered	**Test Data Considered**	**Smoothing Parameter**	**Precision of segregation**
Train	Test1	1	89.5%
Train	Test2	1	90%
Train	Test3	1	90%
Train	Test1	2	89.5%
Train	Test2	2	90%
Train	Test3	2	90%
Train	**Test1**	**3**	**90%**
Train	**Test2**	**3**	**91%**
Train	**Test3**	**3**	**91%**

The concept we have used to implement the spam filter is as below:

We have used the below formula (Naïve Bayes Theorem) to calculate the probability of spam given a word.

Pr(S/W) = [Pr(W/S)*Pr(S)]/ [Pr(W/S) * Pr(S) + Pr(W/H) * Pr(H)]

where:

- Pr(S/W) is the probability that a message is a spam, knowing that the word is in it.
- Pr(S) is the overall probability that any given message is spam.
- Pr(W/S) is the probability that the word appears in spam messages.
- Pr(H) is the overall probability that any given message is not spam.
- Pr(W/H) is the probability that word appears in ham message.

Now to calculate for all the words occurring in the email, each word is considered to be occurring independently and hence the below formula is used:

To prevent underflow, we have used the log based formula instead of the below formula:

P = [p1 * p2 * p3 * p4 ……….] / [(p1 * p2 * p3 * p4 ……….) + ((1 - p1) * (1 - p2) * (1 - p3) * (1 - p4) ……….)]

Where :

- P is the probability that the suspect message is spam;
- P1 is the probability P(S/W1) that it is a spam knowing it contains a first word.
- P2 is the probability P(S/W2) that it is a spam knowing it contains a second word. etc...
- Pn is the probability P(S/Wn) that it is a spam knowing it contains an *N*th word.

Dealing with the rare words (Smoothing Parameter) :

We have used the smoothing parameter to deal with the rare words using the modified probability as shown below :

Pr'(S/W) = [s * Pr(S) + n * Pr(S/W)] / [s + n]

Where :

- Pr'(S/W) is the corrected probability for the message to be spam, knowing that it contains a given word.
- s is the *strength* we give to background information about incoming spam ;
- Pr(S) is the probability of any incoming message to be spam ;
- n is the number of occurrences of this word during the learning phase ;
- Pr(S/W) is the spamicity of this word.

From the training data, we have found out that "3" is a good value for **s.**