Lab Topic 5 - Spam Filtering Using A Naïve Bayes Classifier with Laplace Smoothing
CNM Peralta

I. Naive Bayes Classifier

A Spam Filter classifies messages (commonly emails) to remove incoming spam. This is the technology that prevents you from seeing spam in your inbox folder as much as possible; the spam instead goes to the Spam folder. A common way to implement a spam filter is to represent all received emails as in a bag-of-words and compute the probability of a message being spam given the words that make it up.

One of the ways to solve the spam filtering problem is via the Naive Bayes classifier technique. This technique is based on the Bayesian theorem which states that,

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}.$$

This theorem describes the probability of an event based on prior knowledge of conditions relating to the event. The naivety of the Naive Bayes classifier comes from its strong assumption that the conditions are **independent** of each other, that is, the features/attributes describing the event.

Using the Bayesian network, the relationship between the likelihood of the presence of a word (provided a spam or ham classification) could be modeled. That is, $P(Spam \mid message)$, where $message = w_0 w_1 w_2 \ldots w_n$. The Bayes network that describes the spam filtering problem is:

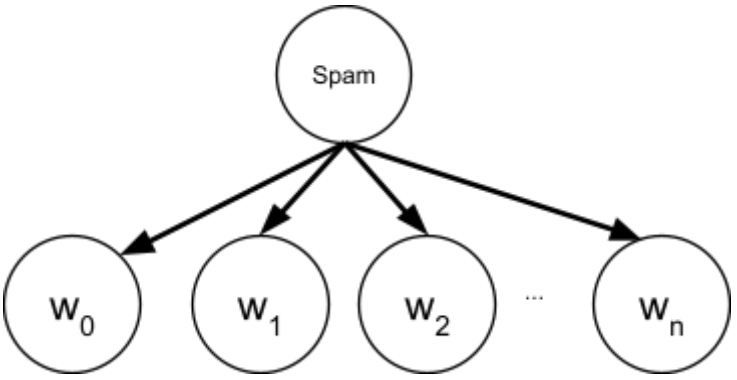


Figure 1. The Bayes* network for the Spam filtering problem.

Thus, if we apply Bayes' rule to $P(Spam \mid message)$, we get:

$$P(Spam \mid message) = \frac{P(message \mid Spam)\, P(Spam)}{P(message)}$$

Spam filtering is a **supervised learning problem**. We are initially given a data set of both Spam and Ham messages. The data set is the set of messages, and the target labels indicate whether a message is Spam/Ham. Since *we already know whether a message is Spam/Ham*, the target labels are already given, thus making spam filtering a supervised learning problem.

Lastly, it is important to note that $Ham = \neg Spam$. We will use Ham and ¬Spam interchangeably.

How do we compute $P(Spam)$?
The probability of Spam can be computed as:

$$P(Spam) = \frac{count(Spam)}{count(Spam \cup Ham)}$$

That is, the number of spam messages over the total number of messages (both Spam and Ham). Consequently, we can compute the probability of Ham as:

$$P(Ham) = \frac{count(Ham)}{count(Spam \cup Ham)} = 1 - P(Spam)$$

How do we compute $P(message \mid Spam)$**?**
Given the Bayes network in Figure 1, we can observe that if we know the value of the Spam variable, the probabilities of the words become independent. That is,

$$P(message|Spam) = P(w_0 w_1 \ldots w_n|Spam) = P(w_0|Spam)P(w_1|Spam)\ldots P(w_n|Spam)$$

The probability of each word given Spam is given by:

$$P(w|Spam) = \frac{P(w, Spam)}{P(Spam)} = \frac{count(w \text{ in } spam)}{count(totality \text{ of the words in } spam)}$$

Applying this, we get: $P(Spam \mid message) = \frac{P(w_0|Spam)P(w_1|Spam)\ldots P(w_n|Spam)\ P(Spam)}{P(message)}$

How do we compute $P(message)$**?**
The total probability of getting a message is:

$$P(message) = P(message|Spam)P(Spam) + P(message|Ham)P(Ham)$$

We already know how to compute $P(message|Spam)$, $P(Spam)$, and $P(Ham)$. So we only need to figure out how to compute $P(message|Ham)$. To do this, we only need to apply the same concept we used in computing their Spam counterpart.

We thus have:

$$P(message|Ham) = P(w_0 w_1 \ldots w_n|Ham) = P(w_0|Ham)P(w_1|Ham)\ldots P(w_n|Ham)$$

And to compute each $P(w|Ham)$, we have:

$$P(w|Ham) = \frac{P(w, Ham)}{P(H)} = \frac{count(w \text{ in } Ham)}{count(totality \text{ of the words in } Ham)}$$

Putting it all together…
We can, therefore, compute $P(Spam|message)$ as…

$$P(S|\ m) = \frac{P(w_0|S)P(w_1|S)\ldots P(w_n|S)\ P(S)}{P(w_0|S)P(w_1|S)\ldots P(w_n|S)\ P(S)+P(w_0|H)P(w_1|H)\ldots P(w_n|H)\ P(H)}$$

where S stands from Spam, m stands for the message, and H stands for Ham (to conserve space).

Lastly,
A **threshold** must be set in order to determine which among the two (2) classes does a given message fall into. For example, when a 50% probability is set in order to classify a file as a spam, once the computed P(Spam|message) falls below 50%, it is automatically classified as a ham. Otherwise, the file will be classified as spam.

Where does the bag-of-words representation come in?
Given the Spam and Ham data set, **two bags-of-words must be constructed**: **one for Spam** and **one for Ham**. Each $P(w|Spam)$ and $P(w|Ham)$is just the **frequency of the word in the data set specified by Spam/Ham**. You thus have everything you need to compute $P(Spam|message)$.

II. Augmenting the Spam Filter with Laplace Smoothing

The problem with the Naïve Bayes Classifier is that, when it encounters a word that is not in either the Spam or Ham data set, the entire message is immediately classified to the other set. For example, if we have:

Spam	Ham
Ang pogi pogi pogi ni Sir James	Ang pogi ni Sir Mir
	Ang cute ni Sir Froi
	Ang macho ni Sir JM

When we attempt to compute $P(Spam \mid 'Ang\ cute\ ni\ Sir\ James')$, we will get:

$$P(Spam \mid 'Ang\ cute\ ni\ Sir\ James') = \frac{P('ang'|Spam)P('cute'|Spam)P('ni'|Spam)P('sir'|Spam)P('james'|Spam)P(Spam)}{P('Ang\ cute\ ni\ Sir\ James')}$$

And looking at the spam data set, we can see that the word 'cute' does not occur; thus, $P('cute'|Spam) = 0$, and consequently, the entire numerator becomes 0. Since the numerator will be 0,

$$P(Spam \mid 'Ang\ cute\ ni\ Sir\ James') = \frac{0}{P('Ang\ cute\ ni\ Sir\ James')} = 0$$

Thus, if a message has a word that occurs in only one data set, it is automatically classified to that data set. Furthermore, if a word does not exist in both sets, a $\frac{0}{0}$ operation will occur, which we know to be undefined. For example:

$$P(Spam|'ang\ gwapo') = \frac{P('ang'|Spam)P('gwapo'|Spam)P(Spam)}{P('ang'|Spam)P('gwapo'|Spam)P(Spam)+P('ang'|Ham)P('gwapo'|Ham)P(Ham)}$$

$$P(Spam|'ang\ gwapo') = \frac{P('ang'|Spam)\times 0\times P(Spam)}{P('ang'|Spam)\times 0\times P(Spam)+P('ang'|Ham)\times 0\times P(Ham)}$$

$$P(Spam|'ang\ gwapo') = \frac{0}{0}$$

This is a manifestation of **overfitting**; we want to be more friendly to new words to both sets.

Enter Laplace Smoothing

Laplace smoothing handles the case of new words by introducing "phantom words" to the data set. For example:

Spam	Ham
Ang pogi pogi pogi ni Sir James	Ang pogi ni Sir Mir
	Ang cute ni Sir Froi
	Ang macho ni Sir JM

Given the data set, the bags-of-words are:

Spam Bag-of-Words	
Word	**Frequency**
ang	1
pogi	3
ni	1
sir	1
james	1

Ham Bag-of-Words	
Word	**Frequency**
ang	3
pogi	1
ni	3
sir	3
mir	1
cute	1
froi	1
macho	1

jm	1

We will modify the formulas that we introduced last meeting to accommodate new words. This is done by using a **smoothing factor**, k, that will be used in the modified formulas.

HOW DO WE COMPUTE P(Spam)?

The probability of Spam can be computed as:

$$P(Spam) = \frac{count(Spam)+k}{count(Spam \cup Ham)+2k}$$

The **probability of Ham is computed in the same way**. The effect of this smoothing is that an additional k observations of both ham and spam are taken into consideration, even if they don't really exist.

HOW DO WE COMPUTE P(message | Spam)?

Recall that the probability of each word given Spam is given by:

$$P(w|Spam) = \frac{count(w\ in\ Spam)}{count(\#\ of\ words\ in\ Spam)}$$

Applying the Laplace smoothing, we have:

$$P(w|Spam) = \frac{count(w\ in\ Spam)+k}{count(\#\ of\ words\ in\ Spam)+(k\times(dictionarySize+count(new\ words)))}$$

$dictionarySize$ = number of unique words found on both Spam and Ham
$new\ words$ = words to be classified not found on both Spam and Ham

The effect of this modification is that the frequency of all words is increased by $k$observations each. Thus, if we compute $P('gwapo'|Spam)$ from our example earlier, and $k = 2$, we will get:

$$P('gwapo'|Spam) = \frac{count('gwapo'\ in\ Spam)+k}{count(\#\ of\ words\ in\ Spam)+(k\times(dictionarySize+count(new\ words)))}$$

$$P('gwapo'|Spam) = \frac{0+2}{7+(2\times(10+1))}$$

The number of new words is 1 because 'ang gwapo' only has one new word: 'gwapo.' Thus, $P('gwapo'|Spam)$ is no longer 0, and we can avoid a $\frac{0}{0}$ operation.

Formula Cheat Sheet (Naive Bayes Classifier with Laplace smoothing)

$$P(Spam \mid message) = \frac{P\ (message \mid Spam)\ P\ (Spam)}{P(message)}$$

$$P(Spam) = \frac{count(Spam) +k}{count(Spam \cup Ham) +2k}$$

$$P(Ham) = \frac{count(Ham) +k}{count(Spam \cup Ham) +2k}$$

$$P(message \mid Spam) = P(w0|Spam)P(w1|Spam)...\ P(wn|Spam)$$

$$P(w|Spam) = \frac{count(w\ in\ Spam)+k}{count(\#\ of\ words\ in\ Spam)+(k\times(dictionarySize+count(new\ words)))}$$

$dictionarySize$: number of **unique** words found on both Spam and Ham
$new\ words$: words to be classified not found on both Spam and Ham

$$P(message) = P(message|Spam)P(Spam) + P(message|Ham)P(Ham)$$

$$P(message \mid Ham) = P(w0|Ham)P(w1|Ham)...\ P(wn|Ham)$$

$$P(w|Ham) = \frac{count(w\ in\ Ham)+k}{count(\#\ of\ words\ in\ Ham)+(k\times(dictionarySize+count(new\ words)))}$$

$$P(Spam \mid message) = \frac{P(w0|Spam)P(w1|Spam)...P(wn|Spam)\ P\ (Spam)}{P(w0|Spam)P(w1|Spam)...P(wn|Spam) + P(w0|Ham)P(w1|Ham)...P(wn|Ham)}$$

Exercise

You may use your previous bag-of-words representation exercise as a starting point for this exercise. Given the spam and ham datasets provided by your lab instructors, your program should:

- Make the bag-of-words representation of the Spam data set from the given **spam folder**.
- Make the **separate** bag-of-words representation of the Ham data set from the given **ham folder**.
- Read a **folder named *classify*** containing a series of messages; there will be **one message per file**.
- Ask the user for the value of the **smoothing factor, k**.
- Compute the probability that each message is Spam using Naive Bayes with Laplace Smoothing. Output your answers to a text file, **classify.out**. The classify.out file must have the filename of the file to be classified, its classification, and the computed probability in one line.

```
001 SPAM 0.85122
002 HAM 0.4302
099 SPAM 0.5001
```

In your terminal output, display the dictionary size and the total number of words of each bow:

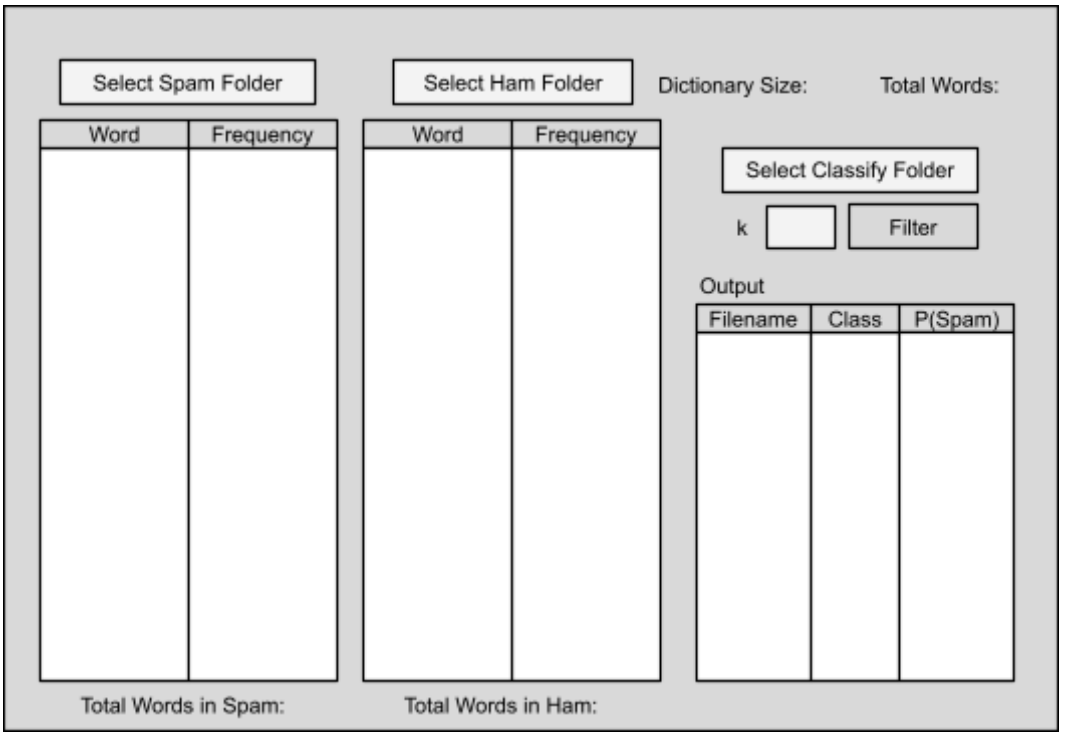
```
HAM
Dictionary Size: 2303
Total Number of Words: 14578

SPAM
Dictionary Size: 6886
Total Number of Words: 20409
```

***Hints*:**

- *Read files in Latin-1 encoding*
- *Check out decimal module in Python and Bigdecimal in Java (to avoid division by zero error)*
- *k = 5 is used in the sample data set output (for checking)*

- Bonus 1 (2pts): Make a UI for your spam filter. It should be of the following format:



- Bonus 2 (5 pts): Look for the best value for the smoothing factor (i.e. parameter tuning) using **cross-validation**:
  - Randomly partition your data (spam and ham folder) into three (3) non-overlapping subsets: training, tuning, and test subsets.
  - The training subset should be 70% of your whole data set; the tuning subset should be 20%; and the remaining 10% should be the test subset. That is, 210 spam files and 210 ham files (total of 420) will be used for the training subset; 60 spam and 60 ham files (total of 120) for the tuning subset; and the remaining files 30 spam and 30 ham files (60 files) will be for the test subset (see video).
  - Generate the spam and ham bag of words using the files in the training subset.
  - Classify the files in the **tuning subset** using Laplace smoothing and the training subset's bow. You will start with k=1, and increase the value of k by 1 until the accuracy of your agent becomes greater than or equal to 90%.
  - Once you find a k with accuracy >= 90%, you will then classify the files in the **test subset** using the k you found. Compute for the accuracy of the new model given the test subset.

  The results of the cross-validation above must be written in a report using LaTeX. The template (containing the guide questions) can be downloaded here.

- The criteria for scoring this exercise is:

Criteria	Score
Correctly generate bag-of-words for Spam	2
Correctly generate bag-of-words for Ham	2
Correctly compute P(Spam) (and P(Ham)) correctlyP(Spam) (and P(Ham))	2
Correctly compute P(word|Spam)	3
Correctly compute P(Spam|message)	5
Correctly output results to file	1
Total	15

References:

Stuart Russell and Peter Norvig. 2009. *Artificial Intelligence: A Modern Approach* (3rd ed.). Prentice Hall Press, Upper Saddle River, NJ, USA.