

# CS432 Web Science: Assignment 8

Finished on April 17, 2020

*Dr. Michele C. Weigle*

**Hannah Holloway**  
hhollowa@odu.edu

# Contents

<b>Problem 1</b>	<b>2</b>
Question . . . . .	2
Answer . . . . .	2
<b>Problem 2</b>	<b>2</b>
Question . . . . .	2
Answer . . . . .	3
<b>Problem 3</b>	<b>5</b>
Question . . . . .	5
Answer . . . . .	5
<b>Problem 4</b>	<b>5</b>
Question . . . . .	5
Answer . . . . .	5

## List of Figures

1	Dictionary . . . . .	3
2	Confusion-Matrix . . . . .	5
3	Classification Report . . . . .	6

## Problem 1

### Question

Q1. Create two datasets, Testing and Training.

The Training dataset should consist of

10 text documents for email messages you consider spam (from your spam folder) 10 text documents for email messages you consider not spam (from your inbox)

The Testing dataset should consist of:

10 text documents for email messages you consider spam (from your spam folder) 10 text documents for email messages you consider not spam (from your inbox)

Make sure that these are plain-text documents and that they do not include HTML tags. The documents in the Testing set should be different than the documents in the Training set.

Upload your datasets to your GitHub repo. Please do not include emails that contain sensitive information.

### Answer

Both inbox and spam email, I put in this format:

First, I created a directory called "datasets". Inside of datasets, I created two subdirectories. The first subdirectory is called "inbox" and the second subdirectory is called "spam".  
Subject: book : Parmanu Row Modification Request

Hi Ravi and Vinay , Hope all is well. SSG and DL have been having issues with a few applications DL has developed to work with \V\_WS\_CRSE\_SCHED\_INFO" viewer , as of while back we were not able to update a module in the server which needs to be done soon . The good news is that we have found the root cause of the issue and have tested and confirmed the solution works in PreProd DB ( Doga ) . I'd like for this change to be implemented to Parmanu \V\_WS\_CRSE\_SCHED\_INFO" viewer as well .

Thanks , Kevin Kevin Sariri Systems Engineer & Web Developer

It can be seen that the first line of the mail is subject and the 3rd line contains the body of the email.

## Problem 2

### Question

Use the provided example code (see [https://github.com/cs432-websci-spr20/assignments/blob/master/train\\_and\\_test\\_naive\\_bayes\\_classifier.py](https://github.com/cs432-websci-spr20/assignments/blob/master/train_and_test_naive_bayes_classifier.py)) to train and test the Naive Bayes classifier.



Figure 1: Dictionary

Use your Training dataset to train the Naive Bayes classifier. Use your Testing dataset to test the Naive Bayes classifier and report the classification results for each email message in the Testing dataset.

**Answer**

From my past projects and research, I have come to know that in any text mining problem, text cleaning is the first step. I removed the words from the document that do not contribute to the information needed to extract. My text cleaning involved two processes:

- a) Removing stop words
- b) Grouping together the different inflected forms of a word so they can be analysed as a single item

After text cleaning, I need to create a dictionary of words and their frequency. The training set of emails is utilized for this.

To see the dictionary I added `print dictionary` to my script. Below are the results:

The below python script will generate a feature vector matrix whose rows denote 20 files of training set and columns denote 3000 words of dictionary. The value at index ‘ij’ will be the number of occurrences of jth word of dictionary in ith file.

I will be using scikit-learn ML library for training classifiers. I have trained the model Naive Bayes classifier. Once the classifiers are trained, I checked the performance of the models on test-set. I extracted word count vector for each mail in test-set and predict its class(ham or spam) with the trained NB classifier.

```

1 import os
  import numpy as np
  from collections import Counter
  from sklearn.naive_bayes import MultinomialNB
  from sklearn.metrics import classification_report
6  from sklearn.metrics import confusion_matrix

  def make_Dictionary(train_dir):
      emails = [os.path.join(train_dir, f) for f in os.listdir(train_dir)]
      all_words = []
11     for mail in emails:
        with open(mail) as m:
            for i, line in enumerate(m):
                if i == 2:
                    words = line.split()

```

```

16         all_words += words

        dictionary = Counter(all_words)

        list_to_remove = dictionary.keys()
21     for item in list_to_remove:
        if item.isalpha() == False:
            del dictionary[item]
        elif len(item) == 1:
            del dictionary[item]
26     dictionary = dictionary.most_common(3000)
    return dictionary

def extract_features(mail_dir):
    files = [os.path.join(mail_dir, fi) for fi in os.listdir(mail_dir)]
31     features_matrix = np.zeros((len(files), 3000))
    docID = 0;
    for fil in files:
        with open(fil) as fi:
            for i, line in enumerate(fi):
36                 if i == 2:
                    words = line.split()
                    for word in words:
                        wordID = 0
                        for i, d in enumerate(dictionary):
41                             if d[0] == word:
                                wordID = i
                                features_matrix[docID, wordID] = words.count(word)
                        docID = docID + 1
    return features_matrix
46

# Create a dictionary of words with its frequency

train_dir = './datasets/train-mails'
dictionary = make_Dictionary(train_dir)
51

# Prepare feature vectors per training mail and its labels

train_labels = np.zeros(20)
train_labels[10:20] = 1
56 train_matrix = extract_features(train_dir)

# Training Naive bayes classifier and its variants

61 model1 = MultinomialNB()

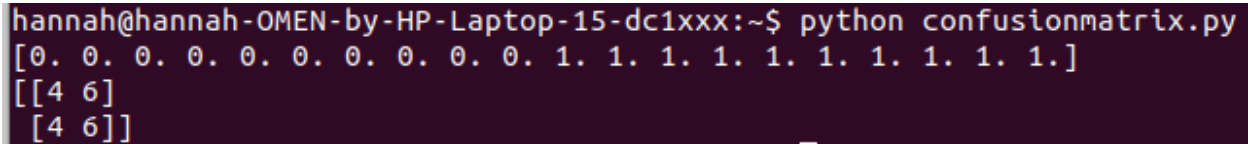
model1.fit(train_matrix, train_labels)
# Test the unseen mails for Spam

66 test_dir = './datasets/test-mails'
test_matrix = extract_features(test_dir)
test_labels = np.zeros(20)

test_labels[10:20] = 1
71

result1 = model1.predict(test_matrix)
print test_labels
print confusion_matrix(test_labels, result1)

```



```
hannah@hannah-OMEN-by-HP-Laptop-15-dc1xxx:~$ python confusionmatrix.py
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
[[4 6]
 [4 6]]
```

Figure 2: Confusion-Matrix

## Problem 3

### Question

Draw a confusion matrix for your classification results (see <https://en.wikipedia.org/wiki/Confusionmatrix> and Week-05 Searching slides for false positive/false negative examples).

### Answer

Test-set contains 10 spam emails and 10 non-spam emails. Below is the confusion matrix of the test-set. The diagonal elements represents the correctly identified (a.k.a. true identification) mails where as non-diagonal elements represents wrong classification (false identification) of mails.

## Problem 4

### Question

(Extra credit, 3 points) Report the precision and accuracy scores of your classification results (see <https://en.wikipedia.org/wiki/Precisionand> Week-05 Searching slides).

### Answer

I was able to print the precision and accuracy scores by simple adding:

```
print classification_report(test_labels,result1)
```

Here are the results:

	precision	recall	f1-score	support
0.0	0.50	0.40	0.44	10
1.0	0.50	0.60	0.55	10
micro avg	0.50	0.50	0.50	20
macro avg	0.50	0.50	0.49	20
weighted avg	0.50	0.50	0.49	20

hannah@hannah-OMEN-by-HP-Laptop-15-dc1xxx:~\$

Figure 3: Classification Report

## References

<https://stackoverflow.com/questions/2148543/how-to-write-a-confusion-matrix-in-python/30385488>  
[https://github.com/cs432-websci-spr20/assignments/blob/master/Week11\\_Cho6\\_PCI.ipynb](https://github.com/cs432-websci-spr20/assignments/blob/master/Week11_Cho6_PCI.ipynb)  
<https://hackernoon.com/how-to-build-a-simple-spam-detecting-machine-learning-classifier-4471fe6b816e>