ASSIGNMENT 8

CS 432 Web Science

Mackenzie Kerchner

Mackenzie Kerchner
ODU CS 432 Web science Spring 2019 Sunday, April 14, 2019

Contents

Mackenzie Kerchner
ODU CS 432 Web science Spring 2019 Sunday, April 14, 2019

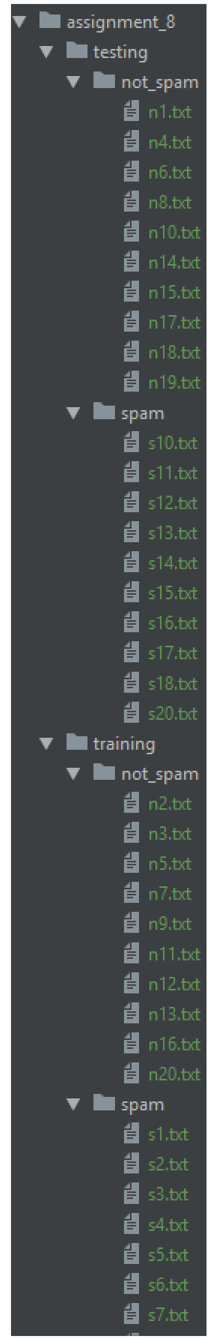
Problem 1

```
1.  Create two datasets; the first called Testing, the second called
Training. Upload your datasets on GitHub
```

Solution

Spam mail testing 11 and 12 are from my inbox, the rest of spam testing and training are downloaded from http://www.linuxfocus.org/common/src/article279/spam_samples.html as Gmail deletes spam mail over time. The training and testing non-spam emails are all from my own inbox and have been scrubbed of personal information. The document sets are on GitHub and are in the hierarchy of the rightmost image.

Originally n1-10 and s1-10 were training and n11-20 and s11-20 were testing, and while all the spam was correctly classified, all but 2 of the genuine emails were classified as spam. I swapped n1-4 and n11-14 and the results shifted to 8/10 correctly marked spam messages, and 6/10 correctly marked genuine messages. After a bit more swapping, I got the results to 9/10 correct spam classification, and 10/10 correct not spam classification.



Problem 2

```
2.  Using the PCI book modified docclass.py code and test.py
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.
```

Solution

The train function calls the docclass function of the same name for each line in every file in the specified path, with the designation of 'spam' or 'not spam'

```
def train(filepath, desig): #desig is 'spam' or 'not spam'
    file_list = os.listdir(filepath)
    for each in file_list:
        test_string = ""
        for line in open(filepath+each, 'r'):
            if line != '\n':
                train_set.train(line, desig)
```

The test function calls the classify function for each file in the specified path and prints it to the console.

```
def test(filepath):
    file_list = os.listdir(filepath)
    for each in file_list:
        test_string = ""
        for line in open(filepath+each, 'r'):
            if line != '\n':
                test_string += line
        print train_set.classify(test_string)
```

The main body and final output of the program:

```
print "training with spam"
train(training_path_spam, 'spam')

print "training with not spam"
train(training_path_not_spam, 'not_spam')

print "testing for spam-----"
test(testing_path_spam)

print "testing for not spam-----"
test(testing_path_not_spam)
```

```
training with spam
training with not spam
testing for spam-----
spam
spam
spam
spam
spam
spam
spam
spam
not_spam
spam
testing for not spam-----
not_spam
not_spam
not_spam
not_spam
not_spam
not_spam
not_spam
not_spam
not_spam
not_spam
```

Problem 3

`3. Draw a confusion matrix for your classification results`

Solution

		actual designation	
		spam	not spam
predicted designation	spam	9	0
	not spam	1	10

Problem 4

`4. Report the precision and accuracy scores of your classification results`

Solution

Of the 9 emails predicted to be spam, 9 were correct, out of the actual 10 spam messages. The precision is therefore 9/9, and the recall is 9/10.

According to the Wikipedia page listed in the assignment document, accuracy is:

$\frac{tp+tn}{tp+tn+fp+fn}$ where tp = true positive (predicted spam, actual spam), tn = true negative (predicted not spam, actual not spam), fp = false positive (predicted spam, actual not spam), and fn = false negative (predicted not spam, actual spam).

By this measurement my classification results have an accuracy rating of 95%.

Mackenzie Kerchner
ODU CS 432 Web science Spring 2019 Sunday, April 14, 2019

References

[1] http://www.linuxfocus.org/common/src/article279/spam_samples.html