Spam Counter Attack with Python

With the help of Python, we can build an app which can detect spam emails and send automatic replies to the senders. Though spam email is not our primary concern, this could be a good exercise to classify text using naive bayes classifier. Test the program using your own email addresses. Do NOT just run it as it may send unwanted message to a legitimate email address. Use at your own risk.

Here is the complete code:

```
import poplib
import email
import nltk
import smtplib
import ssl
import re

msg_data = [('U can WIN 100 of Music Gift Vouchers every week starting NOW', 'spam'),
('Show ur colours! Euro 2004 2-4-1 Offer! Get an England Flag %26; 3Lions tone on ur phone!',
'spam'),
('For ur chance to win a 250 wkly shopping spree TXT: SHOP to 80878.', 'spam'),
('Sunshine Quiz! Win a super Sony DVD recorder if you canname the capital of Australia? Text
MQUIZ to 82277.', 'spam'),
('Please call our customer service as you have WON a guaranteed £1000 cash or £5000 prize!',
'spam'),
('Do you offer a refund for your product? In case it does not work as desired.', 'ham'),
('Is the tool easy to upgrade?', 'ham'),
('Please tell me the main different between your tool and the others in the market.', 'ham'),
('Do I get a discount if I pay for a year in advance?', 'ham'),
('Just want you to know that all the features work well.', 'ham')
```

```
]

def extract_features(sentence):
    word_list = nltk.word_tokenize(sentence[0])
    return dict([word.lower(),True] for word in word_list)

def get_features():
    features_spam = [(extract_features(msg_data[i]),'spam') for i in range(len(msg_data)) if
msg_data[i][1] == 'spam']
    features_ham = [(extract_features(msg_data[i]),'ham') for i in range(len(msg_data)) if
msg_data[i][1] == 'ham']

    threshold = 0.8
    threshold_spam = int(threshold * len(features_spam))
    threshold_ham = int(threshold * len(features_ham))

    features_train = features_spam[:threshold_spam] + features_ham[:threshold_ham]
    features_test = features_spam[threshold_spam:] + features_ham[threshold_ham:]

    return features_train, features_test


account = 'name@mydomain.com'
password = 'abcd1234'
pop3_server = 'mail.mydomain.com'
pop_port = 995
smtp_server = 'mail.mydomain.com'
smtp_port = 465  # For SSL

def get_mail():
    server = poplib.POP3_SSL(pop3_server, pop_port)
    server.user(account)
    server.pass_(password)

    numMessages = len(server.list()[1])
    msg_list = []
    for i in range(numMessages):
        temp_list = []
        str = ''
        for j in server.retr(numMessages-i)[1]:
            msg = email.message_from_bytes(j)
            if msg.get('From'):
                temp_list.append(msg.get('From'))
            elif msg.get('Date'):
                temp_list.append(msg.get('Date'))
            elif msg.get('Subject'):
                temp_list.append(msg.get('Subject'))
            elif msg.get_payload():
                str = str + msg.get_payload() + ' '
            else:
                pass
        temp_list.append(str)
        msg_list.append(temp_list)

    server.quit()
    return msg_list

def send_mail(receiver,subject):
    m = re.search(r'\<(.*)\>', receiver)
    receiver = m.group(1)
    context = ssl.create_default_context()
    server = smtplib.SMTP_SSL(smtp_server, smtp_port, context=context)
    server.login(account,password)
    text = 'Subject: RE:'+subject+' Thanks for your offer, but I am busy creating humus.'
    msg = 'From: {} To: {} {} '.format(account, receiver, text)
    server.sendmail(account, receiver, msg)
    server.quit()


features_train, features_test = get_features()
classifier = nltk.classify.NaiveBayesClassifier.train(features_train)
```

```
print(nltk.classify.util.accuracy(classifier,features_test))            #Accuracy

msg_list = get_mail()

for i in range(len(msg_list)):
    probdist = classifier.prob_classify(extract_features(msg_list[i][3].split()))
    prediction = probdist.max()
    print('Prediction:',prediction)
    print('Probability:',round(probdist.prob(prediction),2))
    if prediction == 'spam':
        print('Send counter message')
        send_mail(msg_list[i][0],msg_list[i][2])
```

Explanation

The variable *msg_data* contains a set of predefined spam and non-spam email messages. It will be used by the classifier to determine whether or not an email message is a spam. The function *get_features()* detects words that are often found in each category (spam and non-spam/ham). The classifier is then built based on the return value of the function as you can see in the code below.

```
classifier = nltk.classify.NaiveBayesClassifier.train(features_train)
```

To find out the accuracy of the classifier, you can use the code below:

```
print(nltk.classify.util.accuracy(classifier,features_test))
```

Change the variables in this block of code with your own. This program uses port 995 and port 465 for secure SSL.

```
account = 'name@mydomain.com'
password = 'abcd1234'
pop3_server = 'mail.mydomain.com'
pop_port = 995
smtp_server = 'mail.mydomain.com'
smtp_port = 465  # For SSL
```

The function *get_mail()* is self explanatory. It retrieve email messages from your remote mailbox. The function returns a list (*msg_list*) that holds important email data, including the email address of the sender (spammer), email subject, and email body. This program only handles plain-text message.

The for loop below goes through the list and executes a function named *send_mail()* everytime an email is detected as spam. The function *send_mail()*, as the name implies, sends the counter message to the spammer.

```
for i in range(len(msg_list)):
    probdist = classifier.prob_classify(extract_features(msg_list[i][3].split()))
    prediction = probdist.max()
    print('Prediction:',prediction)
    print('Probability:',round(probdist.prob(prediction),2))
    if prediction == 'spam':
        print('Send counter message')
        send_mail(msg_list[i][0],msg_list[i][2])
```

Here is the output:

```
Prediction: ham
Probability: 0.83

Prediction: spam
Probability: 0.88
Send counter message
```
