Discuss one thing you notice that is different between the two emails that might relate to the identification of spam.

The format of the two emails is different. The spam emails is in html format and html tags attached to the email while the ham emails do not have anything.

0.0.1 Question 3

Create a bar chart like the one above comparing the proportion of spam and ham emails containing certain words. Choose a set of words that are different from the ones above, but also have different proportions for the two classes. Make sure to only consider emails from `train`.

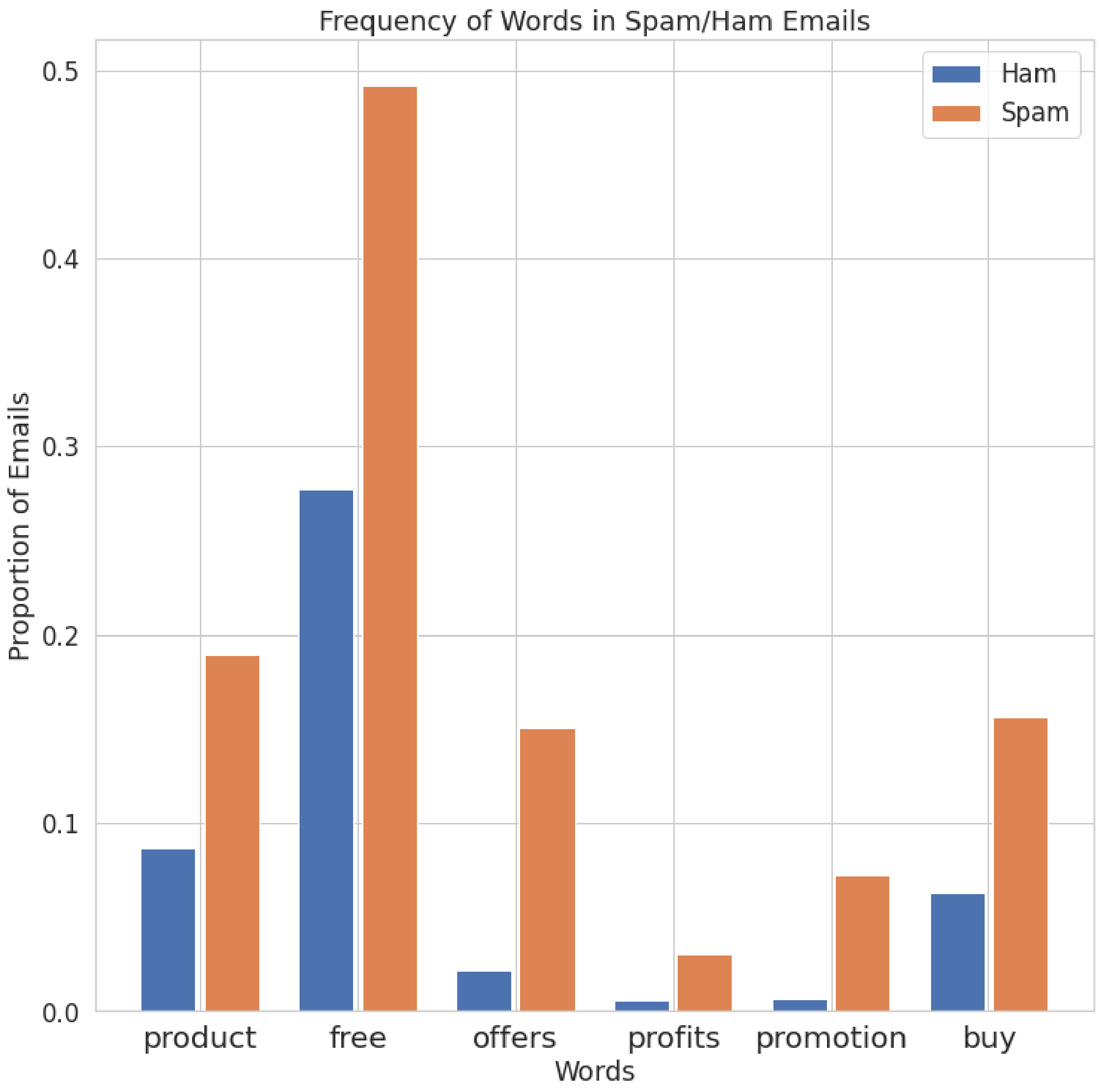
```
In [15]: #@source https://stackoverflow.com/questions/14270391/python-matplotlib-multiple-bars/14270539
         train = train.reset_index(drop=True) # We must do this in order to preserve the ordering of em
         test_words = ['product','free','offers','profits','promotion','buy']
         spam_stuff = train[train['spam'] == 1]['email']
         non_spam = train[train['spam'] == 0]['email']
         find_spam = words_in_texts(test_words, spam_stuff)
         find_normal = words_in_texts(test_words, non_spam)

         def sume(arr):
               sumt = 0
               index = 0
               while (index < len(arr)):
                     sumt += arr[index];
                     index += 1
               return sumt;

         find_proportion_spam = np.sum(find_spam,axis=0)
         find_proportion_normal = np.sum(find_normal,axis=0)
         find_total_spam = len(spam_stuff)
         find_total_normal = len(non_spam)

         widthe = 0.35
         widthy = 0.35
         f, ax = plt.subplots(figsize=(12,12))
         xe = np.arange(len(test_words))
         plt.bar(xe - 0.2,find_proportion_normal/find_total_normal,width=0.35,label='Ham');
         plt.bar(xe + 0.2,find_proportion_spam/find_total_spam,width=0.35,label='Spam');

         plt.legend();
         plt.xticks(xe, test_words)
         plt.xlabel('Words');
         plt.ylabel('Proportion of Emails');
         plt.title('Frequency of Words in Spam/Ham Emails');
         plt.xticks(size = 20)
         plt.show();
```



Frequency of Words in Spam/Ham Emails
Ham
Spam
Proportion of Emails
0.0
0.1
0.2
0.3
0.4
0.5
product
free
offers
profits
promotion
buy
Words

0.0.2 Question 6c

Comment on the results from 6a and 6b. For **each** of FP, FN, accuracy, and recall, briefly explain why we see the result that we do.

There are no false positives because nothing is labeled spam. Every spam email is mislabeled as ham, so the number of false negatives is equal to the number of spam emails in the training data. The classifier correctly labels 74.47% of observations in the training data. The classifier recalls none (0%) of the spam observations.

0.0.3 Question 6e

Are there more false positives or false negatives when using the logistic regression classifier from Question 5?

There are more false negatives, which is 1699, compared to the number of false positives 122.

0.0.4 Question 6f

1. Our logistic regression classifier got 75.76% prediction accuracy (number of correct predictions / total). How does this compare with predicting 0 for every email?
2. Given the word features we gave you above, name one reason this classifier is performing poorly. Hint: Think about how prevalent these words are in the email set.
3. Which of these two classifiers would you prefer for a spam filter and why? Describe your reasoning and relate it to at least one of the evaluation metrics you have computed so far.

An accuracy of about 75% only means we are doing somewhat better than guessing ham for every email. When tasked with implementing word features, one feature that didnt perform well was “money”. Our X_train will have too many rows with 0 in it and thus with our chosen feature it will not be present in many of the emails resulting in it being hard to distinguish ham from spam. Its better to go through the thousands of emails, eliminating spam by hand then by using a false-alarm classifier to filter out an inbox, where 2% of our legitmate emails were filtered out.

To double-check your work, the cell below will rerun all of the autograder tests.

```
In [26]: grader.check_all()
```

```
Out[26]: q2 results: All test cases passed!

         q4 results: All test cases passed!

         q5 results: All test cases passed!

         q6a results: All test cases passed!

         q6b results: All test cases passed!

         q6d results: All test cases passed!
```

0.1 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

```
In [27]: # Save your notebook first, then run this cell to export your submission.
         grader.export()
```

```
<IPython.core.display.HTML object>
```