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

The two emails are different in that the ham is much more organized while the spam is messy and has lots of interjected between text. The ham is also signed with a name and closing while the spam is not.

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`.

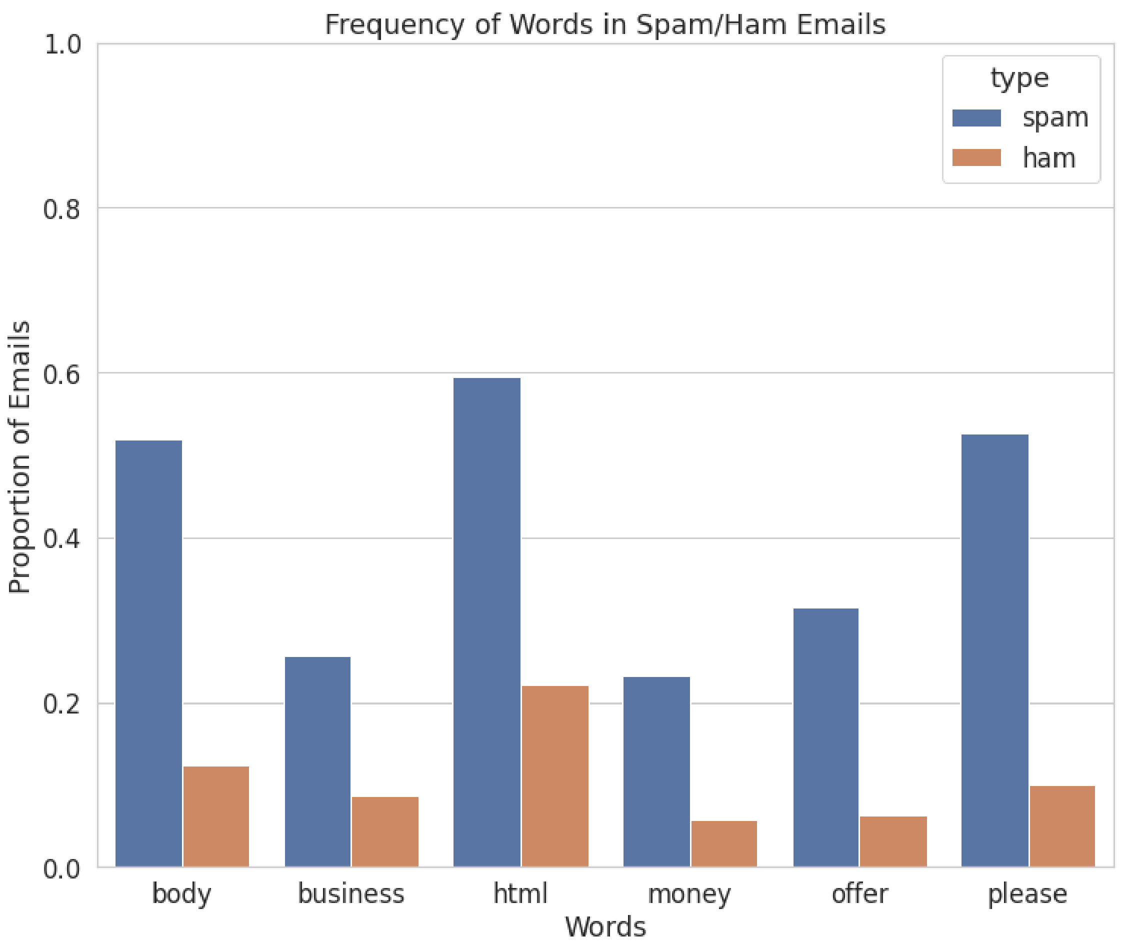
```
In [13]: train.head()
```

```
Out[13]:         id                                            subject  \
         7657  7657             Subject: Patch to enable/disable log\n
         6911  6911        Subject: When an engineer flaps his wings\n
         6074  6074  Subject: Re: [Razor-users] razor plugins for m…
         4376  4376  Subject: NYTimes.com Article: Stop Those Press…
         5766  5766  Subject: What's facing FBI's new CIO? (Tech Up…

                                                    email  spam
         7657  while i was playing with the past issues, it a…     0
         6911  url: http://diveintomark.org/archives/2002/10/…     0
         6074  no, please post a link!\n \n fox\n ----- origi…     0
         4376  this article from nytimes.com \n has been sent…     0
         5766  <html>\n <head>\n <title>tech update today</ti…     0
```

```
In [14]: train = train.reset_index(drop=True) # We must do this in order to preserve the ordering of em
         ham = train.loc[train['spam'] == 0]['email']
         spam = train.loc[train['spam'] == 1]['email']
         words = ["body", "business", "html", "money", "offer", "please"]
         propH = np.sum(words_in_texts(words, ham), axis = 0)/len(ham)
         propS = np.sum(words_in_texts(words, spam), axis = 0)/len(spam)
         data = pd.DataFrame({"word": words, "spam":propS, "ham":propH}, columns = ["word", "spam", "ha
         fin_data = data.melt('word', var_name='type', value_name='prop')
         plt.figure(figsize=(12, 10))
         sns.barplot(x = "word", y = "prop", hue = "type", data = fin_data)
         plt.ylim(0, 1)
         plt.title("Frequency of Words in Spam/Ham Emails")
         plt.xlabel("Words")
         plt.ylabel("Proportion of Emails")
```

```
Out[14]: Text(0, 0.5, 'Proportion of Emails')
```



Frequency of Words in Spam/Ham Emails
type
spam
ham
Proportion of Emails
1.0
0.8
0.6
0.4
0.2
0.0
body
business
html
money
offer
please
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.

FP in 6a would be 0 since there would be 0 false positives; all emails are classified as ham. There would be 1918 false negatives (FN) as this is the number of spam emails that were mislabeled as ham. In 6b, the accuracy is around 0.745 which means that the zero_predictor correctly classifies 74.5% of emails as ham but the recall is 0, since there are 0 false positives and spam emails are never correctly flagged as spam.

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 positives but less false negatives when using the logistic regression classifier from Question 5.

```
In [25]: print (TP, TN, FP, FN)
```

```
219 5473 122 1699
```

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.

75.76% is slightly greater accuracy than predicting 0 for every email, which is 74.48% accuracy. The words given are common, especially since both spam and ham emails can include these words at similar frequencies. Therefore, the classifier has a harder time distinguishing based on these words. I would prefer the logistic regression classifier since it is more accurate than the accuracy of predicting 0 for all emails.

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>
```