0.0.1 Question 1c

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

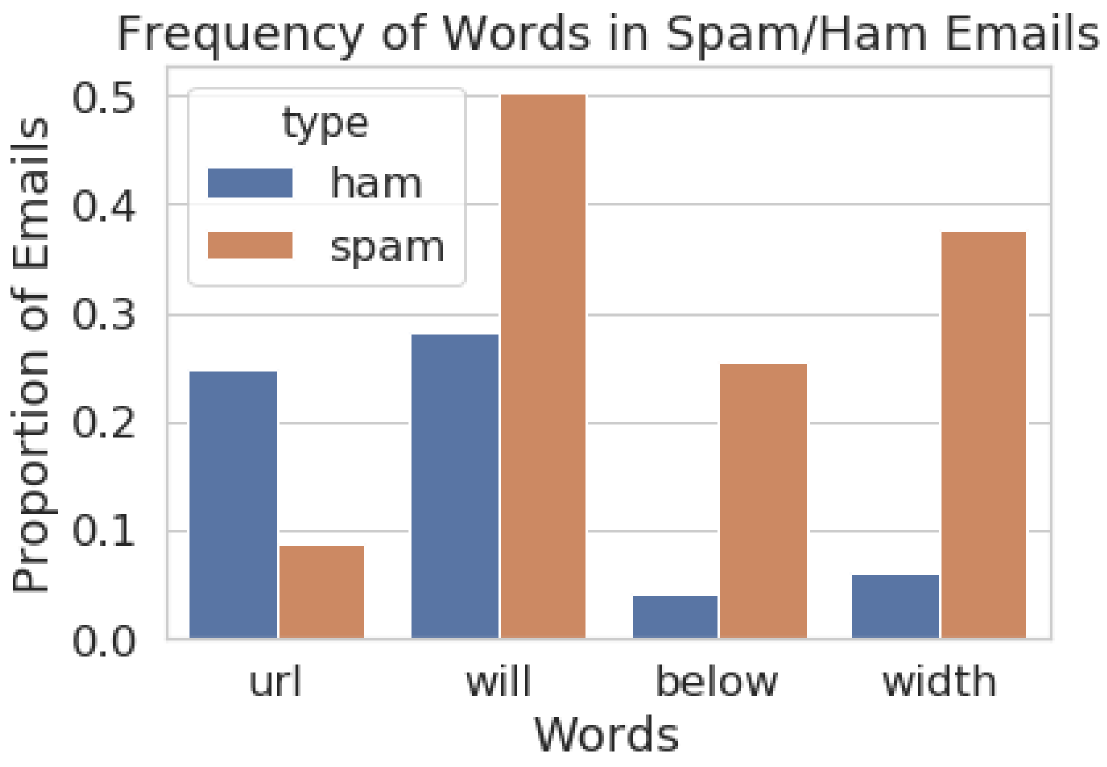
Between these two emails, I notice that the spam email contains an html while the ham email contains an url.

0.0.2 Question 3a

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 [12]: train=train.reset_index(drop=True) # We must do this in order to preserve the ordering of emai

         words = ['url', 'will', 'below', 'width']
         num = words_in_texts(words, train['email'])
         plot = np.matrix(num)
         temp = pd.DataFrame(plot).rename(columns = {0: 'url', 1: 'will', 2: 'below', 3:'width'})
         temp['type'] = train['spam']
         temp = temp.melt('type')
         temp['type'] = temp['type'].map({0:'ham', 1:'spam'})
         temp = temp.rename(columns = {'variable':'Words', 'value':'Proportion of Emails'})

         sns.barplot(x='Words', y='Proportion of Emails', hue='type', data=temp, ci=None)
         plt.title('Frequency of Words in Spam/Ham Emails')
```

```
Out[12]: Text(0.5, 1.0, 'Frequency of Words in Spam/Ham Emails')
```

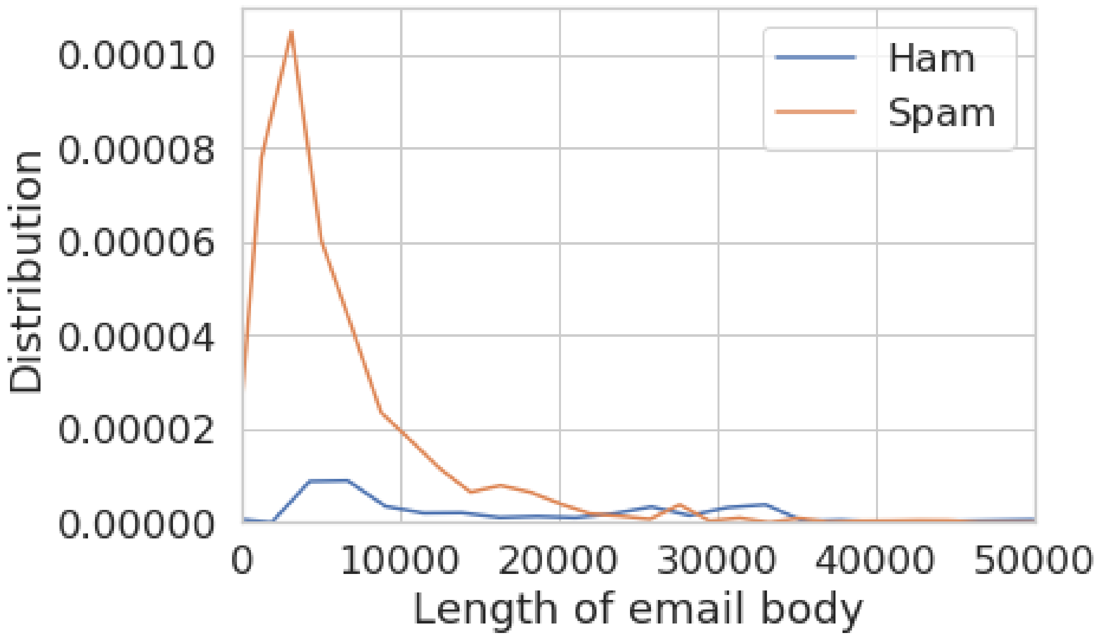


0.0.3 Question 3b

Create a *class conditional density plot* like the one above (using `sns.distplot`), comparing the distribution of the length of spam emails to the distribution of the length of ham emails in the training set. Set the x-axis limit from 0 to 50000.

```
In [13]: length = train['email'].str.len()
         temp_df = pd.DataFrame({'Length':length, 'type': train['spam']})
         temp_df = temp_df.melt('type')
         temp_df['type'] = temp_df['type'].map({0:'Ham', 1:'Spam'})
         ham = temp_df[temp_df['type'] == 'Ham']
         spam = temp_df[temp_df['type'] == 'Spam']

         plt.xlim(0, 50000)
         sns.distplot(ham['value'], label = 'Ham', hist = False)
         sns.distplot(spam['value'], label = 'Spam', hist = False)
         plt.xlabel('Length of email body')
         plt.ylabel('Distribution')
```

```
Out[13]: Text(0, 0.5, 'Distribution')
```



0.0.4 Question 6c

Provide brief explanations of the results from 6a and 6b. Why do we observe each of these values (FP, FN, accuracy, recall)?

FP would give us the number of ham email that got flagged as spam, FN would give us the number of spam being labled as ham email, accuracy would give us the percentage of ham being correctly identified, and recall would give us the proportion of spam emails that were actually labeled as spam. We see that from the results of 6a and 6b, there is no mail that is falsely classified and therefore has FP of 0. While the accuracy of this predictor is about 75% (seems good), the recall ends up being 0 because there are no mails that are classified as spam. This seesm to indicate that accuracy shouldn't be the metrics that we look at when deciding a model is good or not. This ties into why we obeserve the four values. It helps also have a wider range of tools to see if our model is reliable or not. For linear regression we use the rmse and others to see if our model was reliable. However, for logistic regression and classifiers, we use the confusion matrix and the different values given by the matrix to determine a model is accurate or not. Depending on what a person wants, you can change your classifier's threshold to maximize certain values.

0.0.5 Question 6e

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

There are far more false negatives than false positives. This contributes to the high precision and low recall

0.0.6 Question 6f

1. Our logistic regression classifier got 75.6% 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.

1) Our model was slightly higher than the all zero predictor with that predictor accuracy coming at around 74.5%

2) It may be that the words choosen aren't very distinguishly used in the two email type. The words 'drug', 'bank', 'prescription', 'memo', 'private' could be used equally often in both spam and ham emails.

3) In this case, accuracy might be the most important metric so I would use our model. However, this is subjective and depends if you rather have more spam grouped in your ham email or have more ham emails in the spam group.

0.0.7 Question 7: Feature/Model Selection Process

In this following cell, describe the process of improving your model. You should use at least 2-3 sentences each to address the follow questions:

1. How did you find better features for your model?
2. What did you try that worked / didn't work?
3. What was surprising in your search for good features?

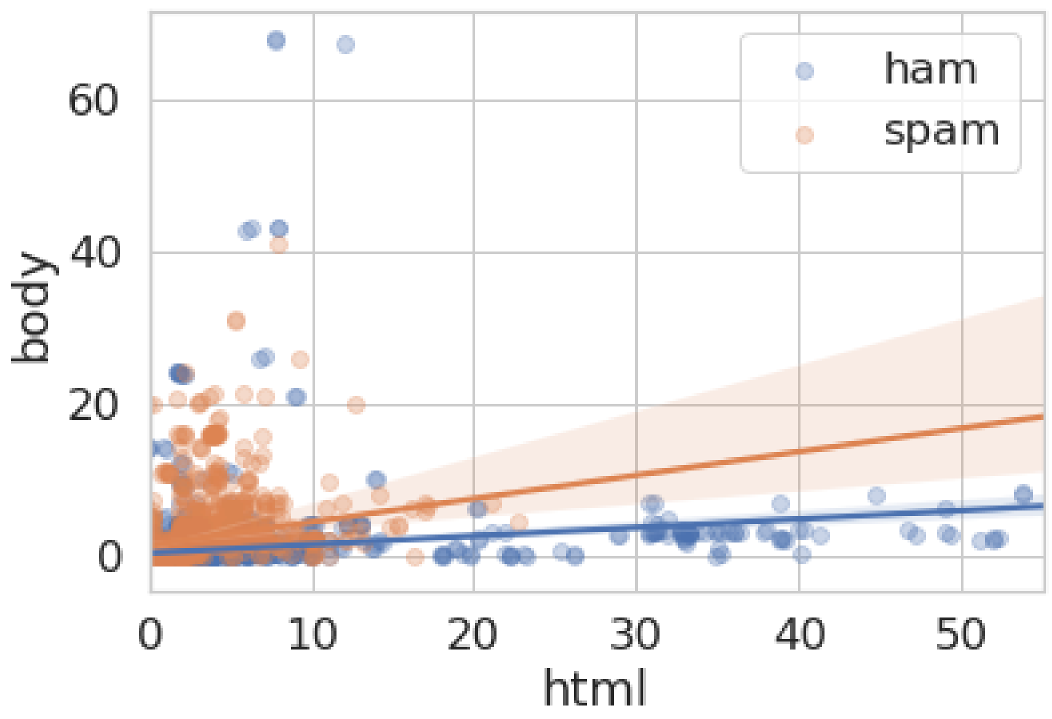
1) To help find better models, I created visual representations to see if there were any distinct trends for certain features. For example, I saw that the number of words in email followed a simialr trend for both spam and ham. Because of this I decided to leave this feature out of the final design matrix. For choosing the words, I used the graph we created earlier in the project to test words and compare their category. If I observed that a particular words was used more in ham or spam by a significant margin, I decided to use that words in my feature.

2) As mentioned previously, I found both the number of words in the email and the subject to not be helpful. I found that both the ham and spam emails had similar amount of words in their subject and email. However, I found that reply and the words (if choosen well) had a really postiive impact on the acuracy. I had huge jumps when I added the reply and gave the design matrix good words.

3) The thing that surprised me the most when searching for good feature is the fact that forward didn't have drastic affect on determing where it was spam or ham. In personal experiences, I've never experienced a forward spam and I though that forward email would give a good indication that it wasn't a spam. However, when adding the forward feature, it didn't have the degree of change I expected. I was expecting similar to reply feature.

Generate your visualization in the cell below and provide your description in a comment.

```
In [34]: # Write your description (2-3 sentences) as a comment here:
         # You can see in the graph below that the words html and body did co-occur relatively frequent
         # for spam, the html stays within the range of 0 to 10 (for most case) and the body tag stays
         # (for most case). For ham, on the other hand, has more variance when it comes to html, with t
         # from 0 to 50. This isn't the case for body as it appears to occur within the range of 0 to 2
         # Since there appears to be more body for spam, it seems to indicate that spam does have more
         # usually. This is just a correlation though and that alone does not indicate causation.

         # Write the code to generate your visualization here:

         ham_body = hams['email'].str.findall('body').str.len()
         ham_html = hams['email'].str.findall('html').str.len()
         spam_body = spams['email'].str.findall('body').str.len()
         spam_html = spams['email'].str.findall('html').str.len()

         df_ham = pd.DataFrame(columns=['body', 'html'])
         df_ham['body'] = ham_body
         df_ham['html'] = ham_html

         df_spam = pd.DataFrame(columns=['body', 'html'])
         df_spam['body'] = spam_body
         df_spam['html'] = spam_html

         plot_ham = sns.regplot(x='html', y='body', data = df_ham, x_jitter = 0.3, y_jitter = 0.3, scat
         plot_spam = sns.regplot(x='html', y='body', data = df_spam, x_jitter = 0.3, y_jitter = 0.3, sca

         plt.xlim(0,55)
         plot_ham.legend()

Out[34]: <matplotlib.legend.Legend at 0x7f91f1fd9f50>
```


ham
spam
body
html
0
20
40
60
0
10
20
30
40
50

0.0.8 Question 9: ROC Curve

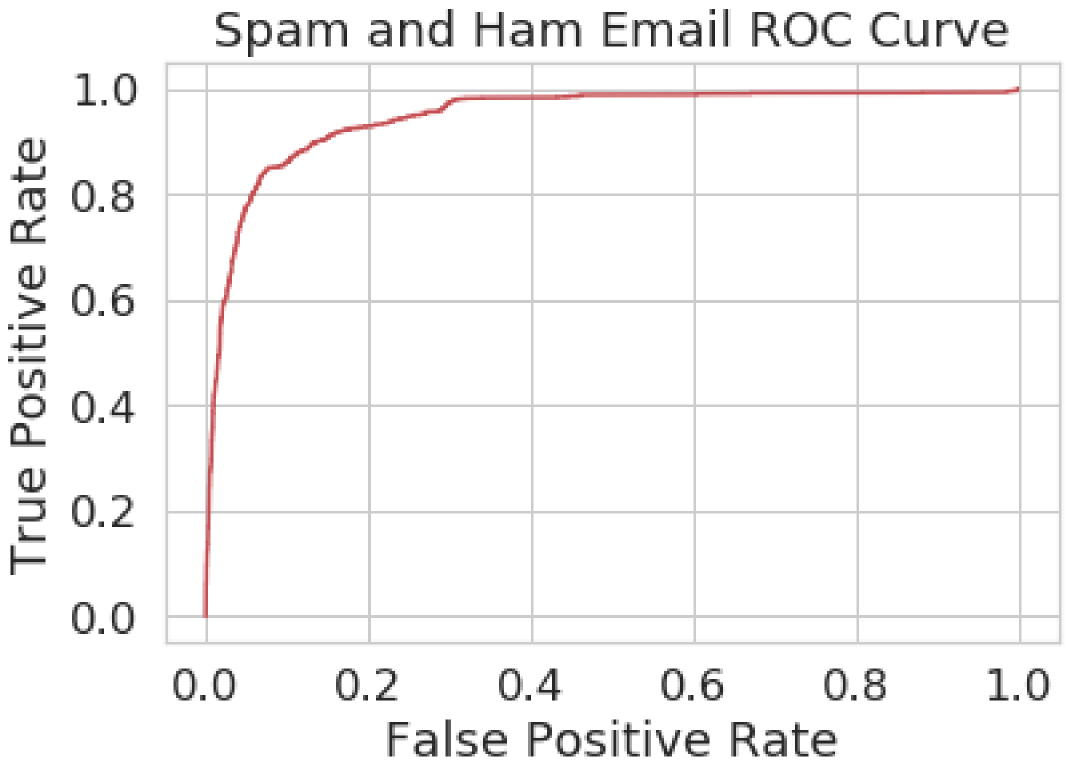
In most cases we won't be able to get no false positives and no false negatives, so we have to compromise. For example, in the case of cancer screenings, false negatives are comparatively worse than false positives — a false negative means that a patient might not discover a disease until it's too late to treat, while a false positive means that a patient will probably have to take another screening.

Recall that logistic regression calculates the probability that an example belongs to a certain class. Then, to classify an example we say that an email is spam if our classifier gives it ≥ 0.5 probability of being spam. However, *we can adjust that cutoff*: we can say that an email is spam only if our classifier gives it ≥ 0.7 probability of being spam, for example. This is how we can trade off false positives and false negatives.

The ROC curve shows this trade off for each possible cutoff probability. In the cell below, plot a ROC curve for your final classifier (the one you use to make predictions for Gradescope) on the training data. Refer to Lecture 19 or Section 17.7 of the course text to see how to plot an ROC curve.

```
In [35]: from sklearn.metrics import roc_curve

         # Note that you'll want to use the .predict_proba(...) method for your classifier
         # instead of .predict(...) so you get probabilities, not classes

         prob = model.predict_proba(XT)[:,1]

         false_positive_rate_values, sensitivity_values, thresholds = roc_curve(Y_train, prob, pos_label

         plt.step(false_positive_rate_values, sensitivity_values, color='r', alpha=1,
                  where='post')

         plt.xlabel('False Positive Rate')
         plt.ylabel('True Positive Rate')
         plt.title('Spam and Ham Email ROC Curve')
```

```
Out[35]: Text(0.5, 1.0, 'Spam and Ham Email ROC Curve')
```


Spam and Ham Email ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0